NVM Express®

Management Interface Specification

Revision 2.1

July 30th, 2025

Please send comments to info@nvmexpress.org

The NVM Express® Management Interface Specification, Revision 2.1 available for download at https://nvmexpress.org. The NVM Express Management Interface Specification, Revision 2.1 consists of the NVM Express Management Interface Specification, Revision 2.0, TP4153, TP4163, TP4190, TP4199, TP8028, ECN124, ECN126, ECN128, ECN129, and ECN130 (refer to https://nvmexpress.org/specification/nvm-express-revision-changes for details).

SPECIFICATION DISLAIMER

LEGAL NOTICE:

This NVM Express® Management Interface Specification, Revision 2.1 is proprietary to the NVM Express, Inc. (also referred to as "Company") and/or its successors and assigns.

NOTICE TO USERS WHO ARE NVM EXPRESS, INC. MEMBERS: Members of NVM Express, Inc. have the right to use and implement this NVM Express® Management Interface Specification, Revision 2.1 subject, however, to the Member's continued compliance with the Company's Intellectual Property Policy and Bylaws and the Member's Participation Agreement.

NOTICE TO NON-MEMBERS OF NVM EXPRESS, INC.: If you are not a Member of NVM Express, Inc. and you have obtained a copy of this document, you only have a right to review this document or make reference to or cite this document. Any such references or citations to this document must acknowledge NVM Express, Inc. copyright ownership of this document. The proper copyright citation or reference is as follows: **"© 2008 to 2025 NVM Express, Inc. ALL RIGHTS RESERVED."** When making any such citations or references to this document you are not permitted to revise, alter, modify, make any derivatives of, or otherwise amend the referenced portion of this document in any way without the prior express written permission of NVM Express, Inc. Nothing contained in this document shall be deemed as granting you any kind of license to implement or use this document or the specification described therein, or any of its contents, either expressly or impliedly, or to any intellectual property owned or controlled by NVM Express, Inc., including, without limitation, any trademarks of NVM Express, Inc.

LEGAL DISCLAIMER:

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

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

The NVM Express® design mark is a registered trademark of NVM Express, Inc.

PCI-SIG®, PCI Express®, and PCIe® are registered trademarks of PCI-SIG.

NVM Express Workgroup
c/o VTM, Inc.
3855 SW 153rd Drive
Beaverton, OR 97003 USA
info@nvmexpress.org

Table of Contents

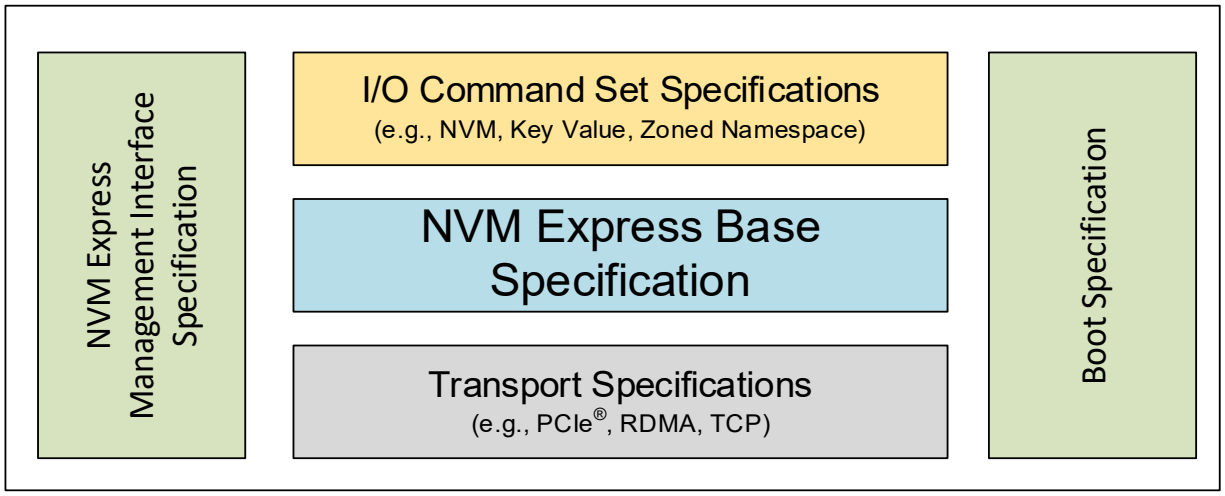
Table of Figures

1 Introduction

1.1 Overview

The NVM Express® Management Interface Specification is a member of the NVMe Family of Specifications displayed in Figure 1.

Figure 1: NVMe Family of Specifications



The NVM Express® (NVMe®) interface allows an in-band host to communicate with an NVM Subsystem. Since this specification builds on the NVM Express® Base Specification, knowledge of the NVM Express® Base Specification is assumed.

This specification defines several mechanisms to manage NVMe Storage Devices (refer to section 1.8.32) or NVMe Enclosures (refer to section 1.8.30). One mechanism allows a Management Controller to communicate out-of-band with an NVMe Storage Device or NVMe Enclosure over one or more external interfaces. Another mechanism is the in-band tunneling mechanism which allows the NVMe-MI Management Interface Command Set to be tunneled in-band via the NVMe Admin Commands NVMe-MI Send and NVMe-MI Receive to an NVMe Storage Device or NVMe Enclosure. Refer to the NVM Express Base Specification and section 4.3 of this specification for additional details on the NVMe-MI Send and NVMe-MI Receive commands.

NVMe Storage Devices and NVMe Enclosures that comply with this specification are allowed to support only the out-of-band mechanism, only the in-band tunneling mechanism, or both the out-of-band mechanism and in-band tunneling mechanism.

1.2 Scope

This specification defines an architecture and command set for out-of-band and in-band management of an NVMe Storage Device as well as an architecture and mechanisms for monitoring and controlling the elements of an NVMe Enclosure.

This specification defines the following key aspects for NVMe Storage Devices:

- Discover NVMe Storage Devices that are present and learn capabilities of each NVMe Storage Device;
- Store data about the host environment enabling a Management Controller or other entity to query the data later;
- Health and temperature monitoring;

- Multiple concurrent commands to prevent a long latency command from blocking monitoring operations;
- An out-of-band mechanism that is host processor and operating system agnostic;
- A standard format for VPD and defined mechanisms to read/write VPD contents; and
- Preserves data-at-rest security.

This specification defines the following key aspects for NVMe Enclosures:

- Discover NVMe Enclosures and learn their capabilities;
- Manage and sense the state of NVMe Enclosure elements such as power supplies, cooling devices, displays, and indicators;
- Multiple concurrent commands to prevent a long latency command from blocking monitoring operations;
- An out-of-band mechanism that is host processor and operating system agnostic;
- Discover NVMe Storage Devices that are present in Enclosure slots; and
- Preserves data-at-rest security.

1.2.1 Outside of Scope

The architecture and command set are specified apart from any usage model. This specification does not specify whether the NVMe interface is used to implement a solid-state drive, a main memory, a cache memory, a backup memory, a redundant memory, etc. Specific usage models are outside the scope of this specification.

This interface is NVM technology agnostic and is specified at a level that abstracts implementation details associated with any specific NVM technology. For example, NAND wear leveling, block erases, and other management tasks are abstracted.

The implementation or use of other published specifications referred to in this specification, even if required for compliance with the specification, are outside the scope of this specification (e.g., PCI Express, 2-Wire, and MCTP).

The management of NVMe over Fabrics is outside the scope of this specification.

This specification does not define new security mechanisms.

This specification does not cover management of non-transparent bridges or PCIe® switches. Coordination between multiple Requesters or a Requester and a device other than a Responder is outside the scope of this specification. Refer to section 1.8 for the definitions of Requester and Responder.

Coordinating concurrency resulting from operations associated with multiple Responders or between host and Management Endpoint operations is outside the scope of this specification.

The specification of specific Enclosure elements that make up an NVMe Enclosure is outside the scope of this specification. Support for cards or modules that connect to a device slot element (slot) of an NVMe Enclosure, that are not NVMe Storage Devices (e.g., GPUs or FPGAs) is outside the scope of this specification.

An enclosure may support comprehensive management capabilities using SCSI Enclosure Services, basic management capabilities using transport specific mechanisms, or no management capabilities. An example of basic enclosure management capabilities is Native PCIe Enclosure Management (NPEM) specified by the PCI-SIG® for PCI Express®. The specification of such transport specific basic management capabilities is outside the scope of this specification. This specification only defines comprehensive management using SCSI Enclosure Services.

An NVMe Enclosure may contain multiple Enclosure Services Processes. Communication and coordination between the Enclosure Services Processes that manage NVMe Enclosure elements is outside the scope of this specification.

1.3 Theory of Operation

This specification is designed to provide a common interface over multiple physical layers (i.e., PCI Express, 2-Wire) for inventory, monitoring, configuration, and change management. This specification provides the flexibility necessary to manage NVMe Storage Devices or NVMe Enclosures using an out-of-band mechanism or in-band tunneling mechanism in a variety of host environments and systems. This specification also defines a FRU Information Device that contains Vital Product Data (refer to section 1.3.1.2).

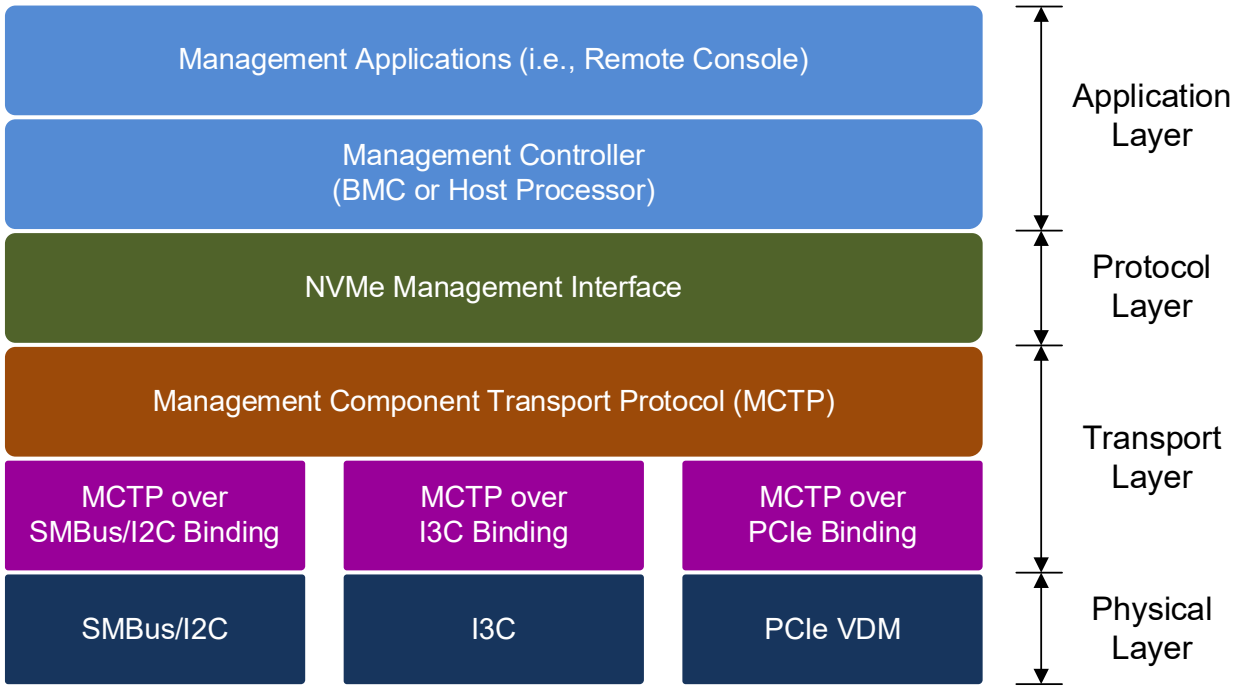
1.3.1 Out-of-Band Theory of Operation

This specification defines a mechanism for managing NVMe Storage Devices and NVMe Enclosures out-of-band via the Management Component Transport Protocol.

1.3.1.1 Management Component Transport Protocol

The out-of-band mechanism utilizes the Management Component Transport Protocol (MCTP) as the transport and utilizes existing MCTP SMBus/I2C, I3C, and PCIe bindings for the physical layer. Command Messages are submitted to one of two Command Slots associated with a Management Endpoint contained in an NVM Subsystem. Figure 2 shows the NVMe-MI out-of-band protocol layering from the Requester's point of view.

Figure 2: NVMe-MI Out-of-Band Protocol Layering



1.3.1.2 FRU Information Device

This specification defines a mechanism to access a FRU Information Device either via SMBus/I2C as defined by the IPMI Platform Management FRU Information Storage Definition specification or via the VPD Read and VPD Write commands. The data stored in the FRU Information Device is referred to as Vital Product Data (refer to section 8.2). A FRU Information Device may be implemented in a variety of ways (e.g., a serial EEPROM, one-time programmable memory in an NVMe Controller ASIC, etc.).

1.3.2 In-Band Theory of Operation

This specification defines an in-band tunneling mechanism that utilizes the NVMe Admin Commands NVMe-MI Send and NVMe-MI Receive. Refer to the NVM Express Base Specification and section 4.3 of this specification for additional details on the NVMe-MI Send and NVMe-MI Receive commands.

1.4 NVM Subsystem Architectural Model

This specification defines an interface that may be used to manage NVM Subsystems contained within an NVMe Storage Device or NVMe Enclosure.

The NVMe Storage Device (NVMESD) bit in the NVM Subsystem Report (NVMSR) field of the Identify Controller data structure shall be set to '1' for an NVMe Storage Device. The NVMe Enclosure (NVMEE) bit in the NVM Subsystem Report (NVMSR) field of the Identify Controller data structure shall be set to '1' for an NVMe Enclosure. The NVMESD bit, the NVMEE bit, or both the NVMESD bit and the NVMEE bit shall be set to '1' (refer to the NVM Express Base Specification).

Management of an NVM Subsystem using the in-band tunneling mechanism and the out-of-band mechanism consists of sending Command Messages and receiving corresponding Response Messages. Command Messages consist of:

a) standard NVMe Admin Commands that target a Controller within the NVM Subsystem;
b) commands that provide access to the PCI Express configuration, I/O, and memory spaces of a Controller in the NVM Subsystem; and
c) Management Interface specific commands for inventorying, configuring, and monitoring of the NVM Subsystem.

The Command Messages supported by an NVM Subsystem are dependent on the mechanism used to send the NVMe-MI Message (i.e., in-band tunneling mechanism or out-of-band mechanism) and whether the NVM Subsystem is contained within an NVMe Storage Device or NVMe Enclosure.

When using the in-band tunneling mechanism, the architecture and behavior of an NVM Subsystem is as defined by the NVM Express Base Specification with extensions defined by this specification. The remainder of this section describes the architecture and behavior of an NVM Subsystem when the out-of-band mechanism is used.

The PCIe ports and 2-Wire port of an NVM Subsystem may each contain zero or more Management Endpoints. A Management Endpoint is an MCTP endpoint that is the terminus and origin of MCTP packets/messages and is responsible for implementing the MCTP Base Protocol, processing MCTP Control Messages, and internal routing of Command Messages. Each Management Endpoint in an NVM Subsystem has a Port Identifier that is less than or equal to the Number of Ports (NUMP) field value in the NVM Subsystem Information data structure. If multiple Management Endpoints are supported on a port, then the NVMe Subsystem shall support MCTP bridging for MCTP endpoint ID discovery and assignment on that port (refer to the MCTP Base Specification).

Management Interface Request Messages and Response Messages are transported as MCTP messages with the Message Type set to NVM Express Management Messages over MCTP (refer to the MCTP IDs and Codes specification). All out-of-band mechanism Request Messages originate with the Management Controller and result in a Response Message from a Management Endpoint.

Each Management Endpoint advertises the unique set of capabilities supported by that Management Endpoint. All Management Endpoints may support the same commands even though PCIe ports are full duplex with much higher data rates than SMBus.

Each NVMe Controller in the NVM Subsystem shall provide an NVMe Controller Management Interface (hereafter referred to as simply Controller Management Interface). The Controller Management Interface processes Controller operations on behalf of any Controller (in-band tunneling mechanism) or Management Endpoint (out-of-band mechanism) in the NVM Subsystem. NVMe Controllers or Management Endpoints may route commands to any NVMe Controller in the NVM Subsystem. A Controller Management Interface

logically processes one operation at a time. A Controller Management Interface is not precluded from processing two or more operations in parallel; however, there shall always be an equivalent pattern of sequential operations with the same results. Responders are permitted to process Command Messages in any order. If the Requester requires Command Messages to be processed in a particular order, then the Requester waits for the Response Message of one Command Message before sending the next Command Message.

Figure 3 illustrates an example NVM Subsystem. The NVM Subsystem contains a single Controller and there is a Management Endpoint associated with the PCIe port.

Figure 3: NVM Subsystem Associated with Single PCIe Port

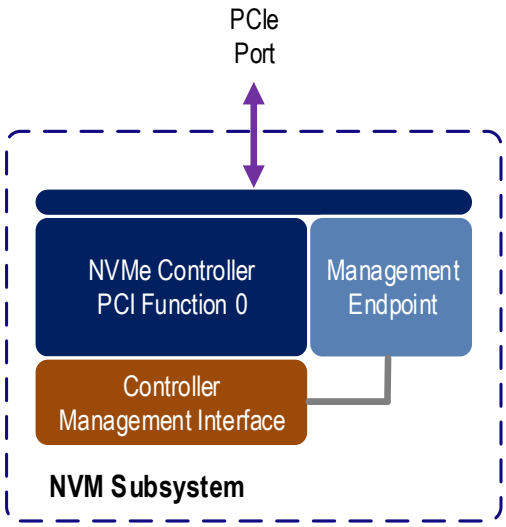
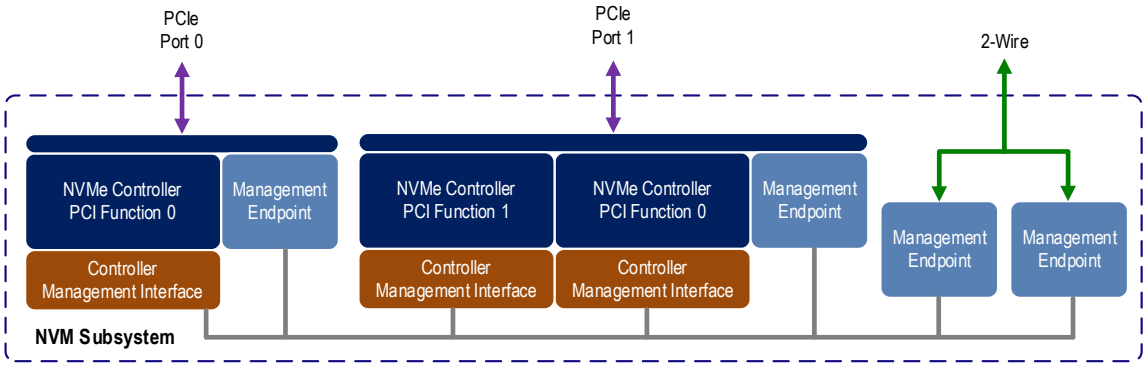


Figure 4 illustrates an example NVM Subsystem that is associated with a dual ported PCIe SSD. The NVM Subsystem contains one Controller associated with PCIe Port 0 and two Controllers associated with PCIe Port 1. There is a Management Endpoint associated with each PCIe port and the 2-Wire port. Since the NVM Subsystem contains a Management Endpoint, all Controllers have an associated Controller Management Interface.

Dual-port PCIe SSDs are typically used in systems that provide two in-band hosts and two Management Controllers for redundancy in high-availability use cases. One Management Controller is connected to the PCIe VDM Management Endpoint on PCIe port 0 and the other Management Controller is connected to the PCIe VDM Management Endpoint on PCIe port 1.

PCIe SSD connectors typically only have a single 2-Wire port. To accommodate two Management Controllers, the PCIe SSD in this example implements two Management Endpoints on the 2-Wire port using MCTP bridging for discovery and assignment of multiple MCTP endpoint IDs (refer to the MCTP Base Specification). The method for determining which Management Controller communicates with which 2-Wire Management Endpoint and the method for the Management Controllers to arbitrate for control of the 2-Wire port are outside the scope of this specification.

Figure 4: NVM Subsystem with Dual PCIe Ports and a 2-Wire Port



1.5 NVMe Storage Device Architectural Model

The architectural model for NVMe Storage Devices that support the in-band tunneling mechanism follows the architectural model defined in the NVM Express Base Specification.

An NVMe Storage Device that implements the out-of-band mechanism but not the in-band tunneling mechanism defined in this specification consists of zero or more NVM Subsystems. An NVMe Storage Device that implements the in-band tunneling mechanisms defined in this specification consists of one or more NVM Subsystems. Each NVM Subsystem that implements the out-of-band mechanism includes one or more Management Endpoints.

An NVMe Storage Device that is a Field-Replaceable Unit (FRU) is a physical component, device, or assembly that is able to be removed and replaced (e.g., by an end user or technician) without having to replace the entire system in which that NVMe Storage Device is contained. Examples of NVMe Storage Device Field-Replaceable Units include a U.2 PCIe SSD, a PCI Express Card Electromechanical (CEM) add-in card, and an M.2 module. The FRU referenced by the FRU Globally Unique Identifier (FGUID) field in the NVM Express Base Specification shall be an NVMe Storage Device Field-Replaceable Unit.

There are many variants of an NVMe Storage Device. One example is an NVMe Storage Device that only contains a single NVM Subsystem (refer to Figure 5 and Figure 6). Another example may contain no NVM Subsystems and instead have one or more Expansion Connectors for adding additional NVMe Storage Device FRUs. Such an NVMe Storage Device is referred to as a Carrier (refer to Figure 7). In another example, the NVMe Storage Device may contain one or more NVM Subsystems and one or more Expansion Connectors. NVMe Storage Devices may contain PCIe switches which connect to one or more NVM Subsystems (refer to Figure 8) or Expansion Connectors. NVMe Storage Devices may contain 2-Wire Muxes (refer to Figure 8) that connect to one or more NVM Subsystems or Expansion Connectors.

This specification defines Vital Product Data (VPD) that utilizes the format defined in the IPMI Platform Management FRU Information Storage Definition and is stored in a FRU Information Device. VPD is accessible over any port that supports the out-of-band mechanism or in-band tunneling mechanism. If the NVMe Storage Device has a 2-Wire port, then the VPD is accessible using the access mechanism over I2C as defined in the IPMI Platform Management FRU Information Storage Definition.

If an NVMe Storage Device contains multiple NVM Subsystems, then the FRU Information Device associated with each NVM Subsystem is optional since the required FRU Information Device accessible via the Upstream Connector describes the entire NVMe Storage Device (refer to section 8.2 for more information). The contents of these additional FRU Information Devices is out of scope for this specification.

Figure 5 illustrates an NVMe Storage Device that is a single-port PCIe SSD with the FRU Information Device implemented by the NVM Subsystem.

Figure 5: Single-Port PCIe SSD

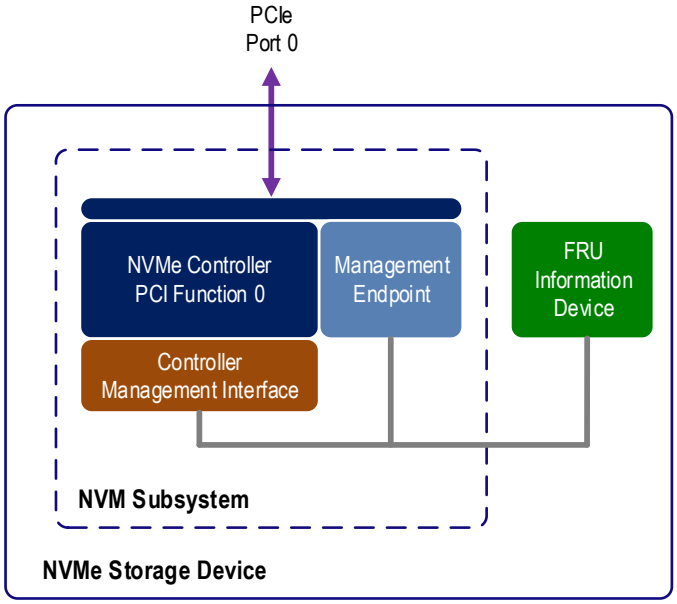
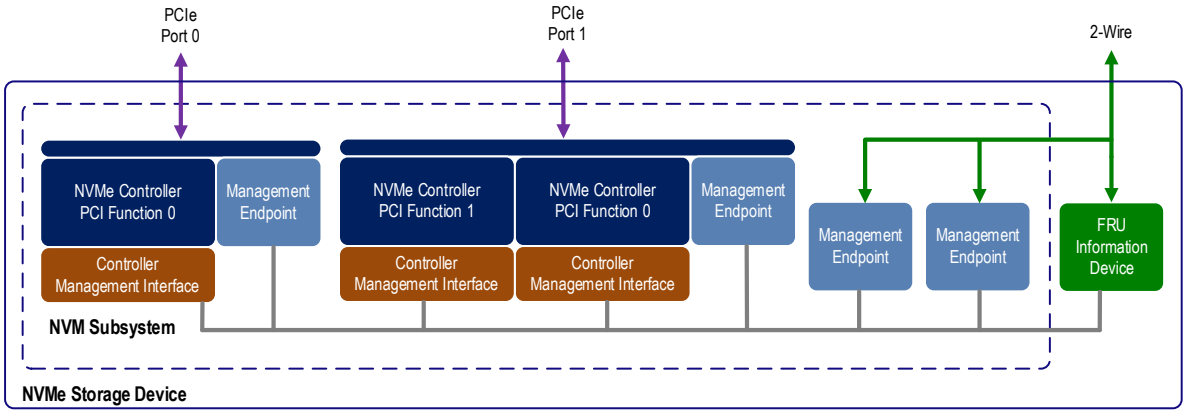


Figure 6 illustrates an NVMe Storage Device that is a dual-port PCIe SSD with a 2-Wire port and a FRU Information Device implemented using a Serial EEPROM.

Figure 6: Dual-Port PCIe SSD with 2-Wire Port



An example U.2 form factor NVMe Storage Device with Expansion Connectors (i.e., a Carrier) is shown in Figure 7. This Carrier has two M.2 Expansion Connectors for connecting two M.2 NVMe Storage Device FRUs. The Carrier and each M.2 NVMe Storage Device are separate NVMe Storage Device FRUs, each with their own FRU Information Device. As defined by Figure 16, the FRU Information Device on the Carrier is at address A4h and the FRU Information Devices on each M.2 NVMe Storage Device has a default address of A6h and supports the SMBus Address Resolution Protocol (ARP). ARP is used after power is applied to reassign the conflicting A6h addresses before the M.2 FRU Information devices are read. ARP is also used to reassign the conflicting MCTP addresses and potentially additional elements. Alternatively, I3C capable elements have an I3C defined method to setup unique bus addresses after switching from SMBus to I3C mode.

Figure 7: NVMe Storage Device with Expansion Connectors (i.e., a Carrier)

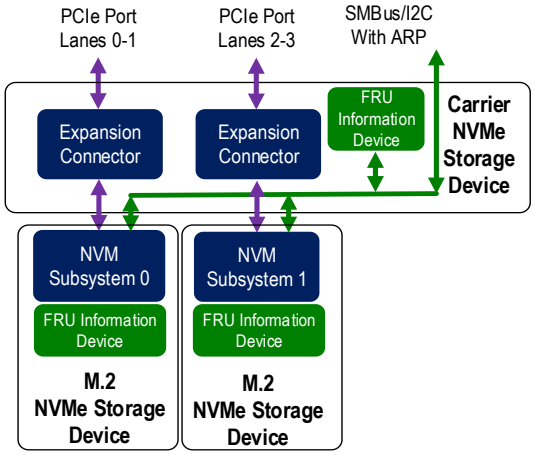
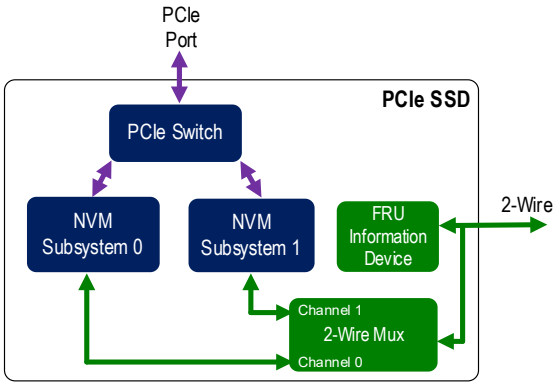


Figure 8 shows an NVMe Storage Device that contains two NVM Subsystems implemented using soldered down ball grid array (BGA) packages and a FRU Information Device at address A6h. An NVMe Storage Device without Expansion Connectors that implements a 2-Wire port always contains a FRU Information Device at address A6h directly connected to the Upstream Connector. A 2-Wire Mux is used in this example instead of ARP to eliminate 2-Wire address collisions. The 2-Wire Mux is configured by a Management Controller prior to communications with the selected NVM Subsystem. The FRU Information Device contains the details necessary to configure the 2-Wire Mux.

Figure 8: NVMe Storage Device with two NVM Subsystems and a 2-Wire Mux



1.6 NVMe Enclosure Architectural Model

An NVMe Enclosure is a platform, card, module, box, rack, or set of boxes that may provide power, cooling, and mechanical protection for one or more NVM Subsystems. These NVM Subsystems may be part of the NVMe Enclosure itself and/or may be contained in NVMe Storage Devices FRUs that connect to the NVMe Enclosure through one or more NVMe Enclosure slots. An NVMe Enclosure contains one or more NVM Subsystems. NVM Subsystems that are part of an NVMe Enclosure may support just the in-band tunneling mechanism, just the out-of-band mechanism, or both.

An NVMe Enclosure may contain elements that support operation of the NVMe Enclosure (e.g., power supplies, fans, locks, temperature sensors, current sensors, and voltage sensors). An NVMe Enclosure may also contain displays and/or indicators that indicate the state of the NVMe Enclosure (e.g., state of elements, NVM Subsystems, or RAID volumes) and/or NVMe Storage Devices that connect to the NVMe

Enclosure. Some of the elements that make up an NVMe Enclosure may be removable and replaceable while the NVMe Enclosure continues to operate normally.

SCSI Enclosure Services - 4 (SES-4) is a standard developed by the American National Standards Institute T10 committee for management of enclosures using the SCSI architecture. While the NVMe and SCSI architectures differ, the elements of an NVMe Enclosure and a SCSI enclosure are similar and the capabilities required to manage elements of an NVMe Enclosure and a SCSI enclosure are similar. Thus, this specification leverages SES for Enclosure Management. SES manages the elements of an enclosure using control and status diagnostic pages transferred using SCSI commands (refer to Enclosure Control and Enclosure Status diagnostic pages in SES-4). This specification uses these same control and status diagnostic pages but transfers them using the SES Send and SES Receive commands. This specification supports only the standalone Enclosure Services Process model as defined in SES.

A Requester manages an NVMe Enclosure using SES Send and SES Receive commands that are part of the Management Interface Command Set (refer to section 5). The SES Send command provides the functionality of the SES-4 SCSI SEND DIAGNOSTIC command and is used by a Requester to send SES control type diagnostic pages to modify the state of the NVMe Enclosure. The SES Receive command provides the functionality of the SES-4 SCSI RECEIVE DIAGNOSTIC RESULTS command and is used by a Requester to retrieve SES status type diagnostic pages that contain various status and warning information available from the NVMe Enclosure.

Refer to SES-4 for a list and description of SES control type diagnostic pages and SES status type diagnostic pages. The mapping of bytes in SES pages to NVMe-MI Request and Response Data is one-to-one where byte x of the SES page maps to byte x in the NVMe-MI Request or Response Data (e.g., byte zero of the SES control type diagnostic page corresponds to byte zero of NVMe-MI Request Data). The NVMe firmware update process is used (i.e., Firmware Image Download and Firmware Commit commands) to update NVMe firmware. Download Microcode Control and Status diagnostic pages, if supported, shall only be supported on NVMe Enclosure elements.

An Enclosure Services Process, that is logically part of the NVMe Enclosure, is responsible for managing NVMe Enclosure elements and participates in servicing SES Send and SES Receive commands issued by a Requester. Unlike the SES-4 Enclosure Services Process model that maintains state for each I_T nexus (refer to SES-4), unless otherwise noted, this specification requires an NVMe Enclosure to maintain a single global state regardless of the Requester or path used to access that state.

An NVMe Enclosure may contain of one or more Subenclosures (refer to SES-4). Each Subenclosure is identified by an SES-4 defined one-byte Subenclosure identifier. If multiple Subenclosures are present, then one of the Subenclosures is designated as the primary Subenclosure and the remaining Subenclosures are secondary Subenclosures. When an NVMe Enclosure consists of only a single Subenclosure, then that Subenclosure is the primary Subenclosure. The Enclosure Services Process associated with the primary Subenclosure is the one that provides access to NVMe Enclosure services information for all Subenclosures. Refer to SES-4 for more information.

Associated with each NVMe Enclosure slot is an SES element that may be used to manage the slot. Refer to SES-4 for more information.

Figure 9 illustrates an example NVMe Enclosure that contains one NVM Subsystem. This NVMe Enclosure has multiple ports that Requesters may use to communicate with the NVMe Enclosure. This NVMe Enclosure also has multiple slots that are used to connect NVMe Storage Devices to the NVMe Enclosure (e.g., PCIe). The mapping of NVMe Enclosure ports to NVM Subsystems, NVMe Controllers within these NVM Subsystems, and NVMe Storage Devices is vendor specific and outside the scope of this specification. An NVMe Enclosure shall contain one or more NVM Subsystems used for Enclosure Management. The NVMe Enclosure in this example may be managed using the out-of-band mechanism via the Responder (refer to the Management Endpoint in Figure 9) or using the in-band tunneling mechanism via the NVMe Controller.

Figure 9: Example NVMe Enclosure

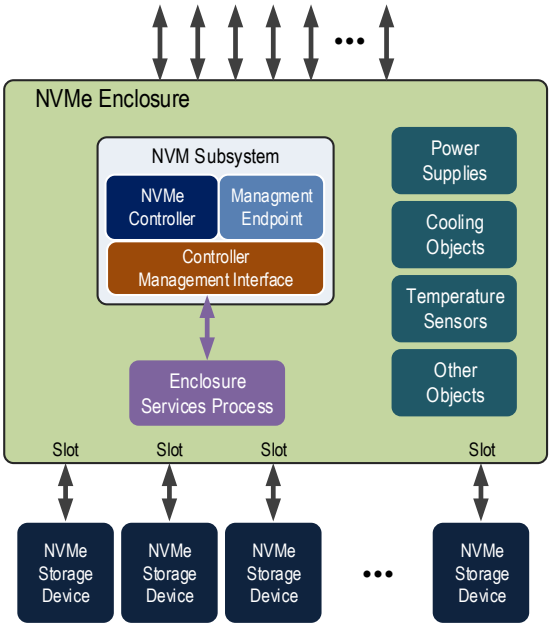


Figure 10 illustrates an example NVMe Enclosure that contains multiple NVM Subsystems and no NVMe Storage Devices. This may represent a software storage appliance. The NVM Subsystems and Controllers contained within these NVM Subsystems may be real or emulated in software. Not all Controllers within these NVM Subsystems are required to have the same capabilities. Some of the possible capability configurations are illustrated in this example. Some Controllers in this example simply provide access to Namespaces; others provide access to Namespaces and support for the in-band tunneling mechanism; and others provide access to Namespaces and support for the out-of-band mechanism.

Figure 10: Example NVMe Enclosure with Multiple NVM Subsystems

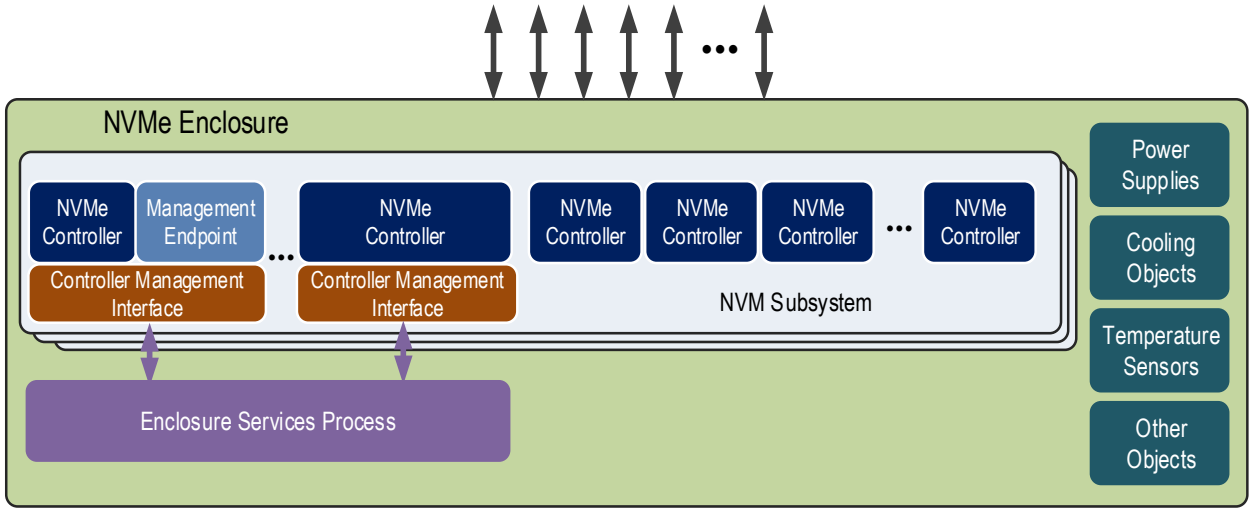


Figure 11 shows an Enclosure that supports two Enclosure Services Processes. Elements of the NVMe Enclosure may be accessible by one or both Enclosure Services Processes. The coordination of access to elements by multiple Enclosure Services Processes is outside the scope of this specification.

Figure 11: Example NVMe Enclosure with Multiple Enclosure Services Processes

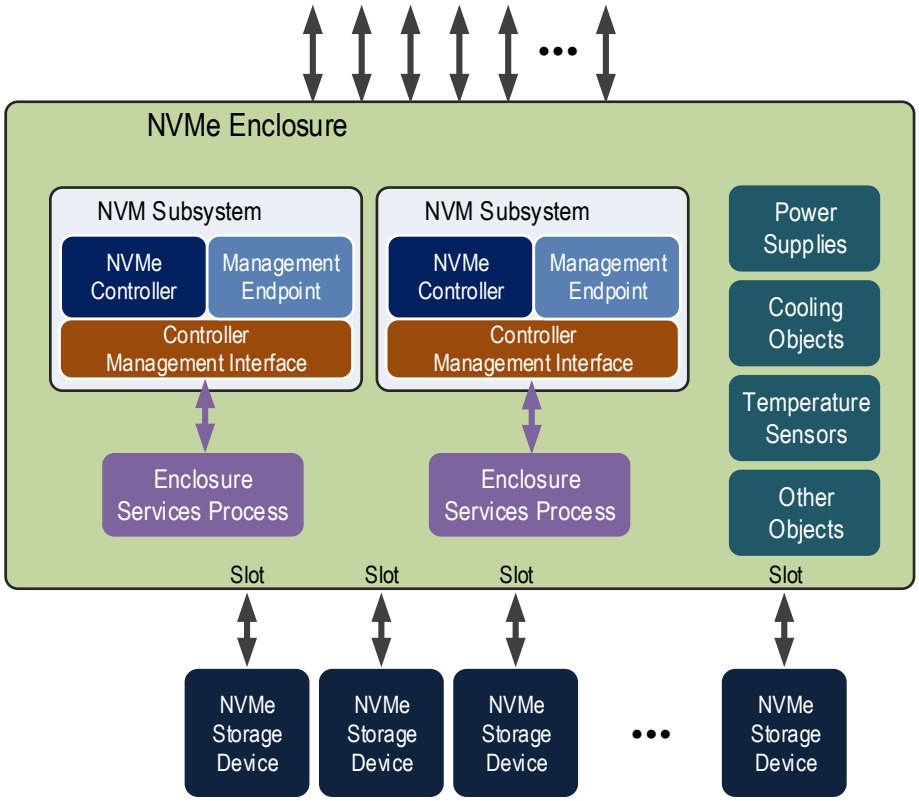
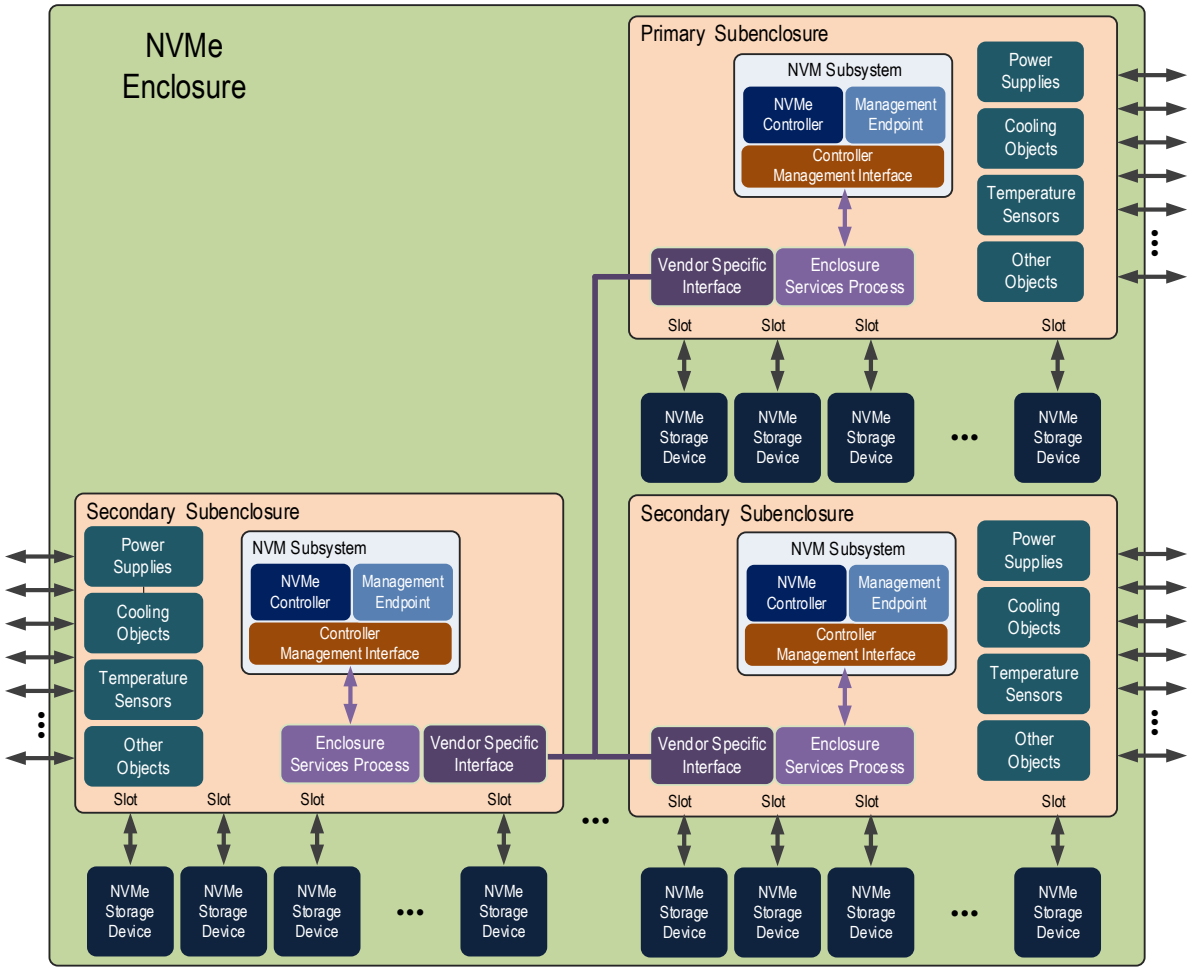


Figure 12 shows an NVMe Enclosure that consists of multiple Subenclosures. Each Subenclosure in this example contains an Enclosure Services Process. NVMe Enclosure services information from Subenclosures is combined into a single set of SES diagnostic pages by the primary Subenclosure. A Subenclosure identifier is used to distinguish from which Subenclosure the information was obtained. Refer to SES-4 for more information. A primary Subenclosure may access NVMe Enclosure services information in Subenclosures using the out-of-band mechanism, the in-band tunneling mechanism, or both; or may use a vendor specific interface. This example illustrates the use of a vendor specific interface.

Figure 12: Example NVMe Enclosure with Subenclosures



Certain NVMe Enclosure behaviors are managed by setting controls and testing status of elements within an NVMe Enclosure. An Enclosure Services Process may monitor a variety of warning and error conditions. These conditions may be communicated to the Requester through polling by the Requester (refer to Enclosure Services Management mode page in SES-4 for details).

The mapping of SES-4 sense keys and additional sense codes associated with CHECK CONDITION status to NVMe-MI Response Message Status values is shown in Figure 13. The asynchronous event notification reporting mechanism described in SES-4 is not supported by this specification.

Figure 13: Mapping of SES-4 Sense Keys and Additional Sense Codes to Response Message Status

Response Message Status Values	SES-4	
	Sense Key	Additional Sense Code
Enclosure Services Failure	HARDWARE ERROR	ENCLOSURE SERVICES FAILURE
Enclosure Services Transfer Failure		ENCLOSURE SERVICES TRANSFER FAILURE
Enclosure Failure		ENCLOSURE FAILURE

Figure 13: Mapping of SES-4 Sense Keys and Additional Sense Codes to Response Message Status

Response Message Status Values	SES-4	
	Sense Key	Additional Sense Code
Enclosure Services Transfer Refused	HARDWARE ERROR or ILLEGAL REQUEST	ENCLOSURE SERVICES TRANSFER REFUSED
Unsupported Enclosure Function	ILLEGAL REQUEST	UNSUPPORTED ENCLOSURE FUNCTION
Enclosure Services Unavailable	NOT READY	ENCLOSURE SERVICES UNAVAILABLE
Enclosure Degraded	RECOVERED ERROR	WARNING – ENCLOSURE DEGRADED

1.7 Conventions

Hardware shall return 0h for all bits, fields, and registers that are marked as reserved. The Requester should not rely on a value of 0h being returned as future revisions of this specification may contain non-zero values. The Requester should write all reserved bits and registers with the value of 0h. Future revisions of this specification may rely on a 0h value being written for backward compatibility.

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

Binary (i.e., base 2) numbers are written with a lower case “b” suffix (e.g., 1001b, 10b). Binary numbers larger than four digits are written with an underscore character dividing each group of four digits (e.g., 1000_0101_0010b).

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

a) the decimal separator (i.e., separating the integer and fractional portions of the number) is a period;
b) the thousands separator (i.e., separating groups of three decimal digits in a portion of the number) is a comma;
c) the thousands separator is used in only the integer portion of a number and not the fractional portion of a number; and
d) the decimal representation for a year does not include a comma (e.g., 2018 instead of 2,018).

2-Wire addresses are written as 8-bit hex values where bits 7:1 contain the 7-bit 2-Wire address and bit 0 is cleared to ‘0’.

When a register field is referred to in the document, the convention used is “Property Symbol.Field Symbol” (e.g., the Controller Status (CSTS) register Shutdown Status (SHST) field is referred to by the name CSTS.SHST). If the register field is an array of bits, the field is referred to as “Property Symbol.Field Symbol (array offset to element)”. When a sub-field is referred to in the document, the convention used is “Property Symbol.Field Symbol.Sub Field Symbol”.

A 0’s based value is a numbering scheme for which the number 0h represents a value of 1h, 1h represents 2h, 2h represents 3h, etc. In this numbering scheme, there is not a method for specifying the value of 0h. Values in this specification are 1-based (i.e., the number 1h represents a value of 1h, 2h represents 2h, etc.) unless otherwise specified.

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

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

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

For any ASCII string or UTF-8 string received from a Requester, a Responder shall treat that string as a binary string (e.g., it shall not perform any text processing that is specific to the character set or locale such as checks for byte values not used by UTF-8, Unicode normalization, etc.).

A range of numeric values is represented in this specification in the form “a to z”, where a is the first value included in the range, all values between a and z are included in the range, and z is the last value included in the range (e.g., the representation “0h to 3h” includes the values 0h, 1h, 2h, and 3h).

Size values are shown in binary units or decimal units. The symbols used to represent these values are as shown in Figure 14.

Figure 14: Decimal and Binary Units

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

Implementation Specific (Impl Spec) – the Responder has the freedom to choose its implementation.

Hardware Initialized (HwInit) – The value is dependent on Responder and system configuration. For the out-of-band mechanism, the value is initialized by an NVM Subsystem Reset. For the in-band tunneling mechanism, the value is initialized by a Controller Level Reset (refer to the NVM Express Base Specification).

1.8 Definitions

1.8.1 2-Wire

A generalized term from the PCI Express Base Specification for the interface port that transfers compatible protocols requiring two physical wires (i.e., SMBus, I2C, and I3C).

1.8.2 2-Wire Mux

A bidirectional 2-Wire fan-out multiplexer with one upstream channel and one or more downstream channels configured by an I2C command from a Management Controller to connect zero or more downstream channels to the upstream channel. Each downstream channel may be connected to devices with 2-Wire ports. This multiplexer permits multiple devices to use the same 2-Wire addresses if they are on separate channels. 2-Wire Mux elements may be designed to only support SMBus/I2C and such elements are unable to handle I3C traffic.

1.8.3 2-Wire Reset

A mechanism used to reset the 2-Wire elements in an NVMe Storage Device or NVMe Enclosure (refer to section 8.3.4).

1.8.4 AE (Asynchronous Event)

A condition (e.g., a health status change event, a temperature change event, etc.) that may occur in the NVM Subsystem. Refer to Figure 63 for the list of Asynchronous Events.

1.8.5 AE Arm

A condition that causes the Management Endpoint to enter the AE Armed State which occurs when a Configuration Set command for the Asynchronous Event configuration is processed that results in the AE Occurrence List Overflow bit cleared to ‘0’ and leaves one or more AEs enabled (i.e., an AE Sync or AEM Ack occurs). Refer to section 4.4.1.

1.8.6 AE Armed State

A state of the Management Endpoint where AEs that occur are permitted to be transmitted in an AEM (refer to section 1.8.9) at the next available AEM Transmission Interval. Refer to section 4.4.1 and section 4.4.3.

1.8.7 AE Disarmed State

A state of the Management Endpoint where:

a) AEs that occur are not permitted to be transmitted in an AEM; or
b) all AEs are disabled.

Refer to section 4.4.1 and section 4.4.3.

1.8.8 AE Sync

A condition that occurs when a Management Endpoint processes a Configuration Set command for an Asynchronous Event configuration that does not have the Number of AE Enable Data Structures field cleared to 0h and results in the AE Occurrence List Overflow bit cleared to ‘0’. An AE Sync:

a) may enable and/or disable one or more AEs; and
b) is used to synchronize the state of enabled AEs between a Management Endpoint and a Management Controller by causing the Management Endpoint to return the state of all enabled AEs to the Management Controller.

Refer to section 5.2.4.

1.8.9 AEM (Asynchronous Event Message)

An NVMe-MI Message transmitted from a Management Endpoint to a Management Controller containing information about one or more AEs that have occurred. Refer to section 4.4.

1.8.10 AEM Ack

A condition that occurs when a Management Endpoint processes a Configuration Set command for an Asynchronous Event configuration that has the Number of AE Enable Data Structures field cleared to 0h and results in the AE Occurrence List Overflow bit cleared to '0'. An AEM Ack is used by a Management Controller to acknowledge receipt of an AEM to the Management Endpoint that transmitted the AEM. Refer to section 5.2.4.

1.8.11 AEM Delay Interval

The time a Management Endpoint delays from the start of the AE Armed State before the Management Endpoint is permitted to enter the AEM Transmission Interval to transmit an AEM for any AEs that occurred during the AE Armed State. Refer to section 4.4.2.

1.8.12 AEM Transmission Interval

The time during which an AEM for AEs that occurred during the prior AE Armed State are transmitted or retried by the Management Endpoint. Refer to section 4.4.3.

1.8.13 Carrier

An NVMe Storage Device FRU with one or more Expansion Connectors and zero or more NVM Subsystems.

1.8.14 Command Message

A Request Message that contains an NVMe Admin Command, PCIe Command, or NVMe-MI Command.

1.8.15 Command Slot

A logical target within a Management Endpoint where a Management Controller sends a Request Message. Each Management Endpoint has exactly two Command Slots.

1.8.16 Control Primitive

Single-packet Request Messages sent from a Management Controller to a Management Endpoint to:

- affect the servicing of a previously issued Command Message; or
- get the state of a Command Slot and Management Endpoint.

Control Primitives are applicable only in the out-of-band mechanism and are prohibited in the in-band tunneling mechanism.

1.8.17 NVMe Controller (Controller)

Refer to the NVM Express Base Specification.

1.8.18 NVMe Controller Management Interface (Controller Management Interface)

An interface associated with each NVMe Controller in the NVM Subsystem that is responsible for processing management operations on behalf of a Management Endpoint.

1.8.19 Enclosure Management

The discovery, monitoring and control of elements that make up an NVMe Enclosure.

1.8.20 Enclosure Services Process

A process that implements Enclosure services for an NVMe Enclosure that supports Enclosure Management. Refer to SCSI Enclosure Services - 4 (SES-4) for more information.

1.8.21 Expansion Connector

A connector that allows an NVMe Storage Device FRU or cable to be attached or removed from a Carrier. Expansion Connectors may be empty or populated. A connector to a non-removable NVMe Storage Device is not an Expansion Connector.

1.8.22 Field-Replaceable Unit (FRU)

A physical component, device, or assembly in a system that is able to be removed and replaced (e.g., by an end user or technician) without having to replace the entire system in which it is contained. The Field-Replaceable Unit described in this specification is an NVMe Storage Device Field-Replaceable Unit (refer to section 1.8.33).

1.8.23 FRU Information Device

A logical or physical device used to hold the VPD. A FRU Information Device may be implemented in a variety of ways (e.g., a serial EEPROM, one-time programmable memory in silicon, etc.).

1.8.24 Interpacket Time

The time a Management Endpoint takes between transmitting packets. This time is measured at the Management Endpoint from the end of the successful transmission of any packet to the beginning of the transmission of the next packet. Note that the next packet may be part of the same Response Message as the prior packet or may be part of another Response Message.

1.8.25 In-Band

Per the Management Component Transport Protocol (MCTP) Overview White Paper, in-band management is management that operates with the support of hardware components that are critical to and used by the operating system. The in-band communication path defined by this specification is via the NVMe Admin Queue using the NVMe-MI Send and NVMe-MI Receive commands from a host to an NVMe Controller. Refer to the NVM Express Base Specification and section 4.3 of this specification for additional details on the NVMe-MI Send and NVMe-MI Receive commands.

1.8.26 Management Controller

A device (e.g., Baseboard Management Controller (BMC)) responsible for platform management that uses the NVM Express Management Interface to communicate to Management Endpoints.

1.8.27 Management Endpoint

An MCTP endpoint associated with an NVM Subsystem (e.g., an NVMe SSD or NVMe Enclosure) that is the terminus and origin of MCTP packets/messages and which processes Request Messages and transmits Response Messages.

1.8.28 Management Endpoint Buffer

An intermediate buffer defined by this specification to allow servicing out-of-band NVMe-MI Messages that have a Message Body that is larger than the 4,224-byte limit that is specified by the NVMe Management Messages over MCTP Binding Specification.

1.8.29 Management Endpoint Reset

A mechanism used to reset a Management Endpoint in an NVMe Storage Device or NVMe Enclosure. For more information, refer to section 8.3.3.

1.8.30 NVMe Enclosure

A platform, card, module, box, rack, or set of boxes that may provide power, cooling, mechanical protection and/or external interfaces for zero or more NVMe Storage Device FRUs. An NVMe Enclosure contains one or more NVM Subsystems and one or more Enclosure Services Processes.

1.8.31 NVMe Processing

NVMe command processing as defined by the NVM Express Base Specification. The term NVMe Processing is used to distinguish command processing as defined by the NVM Express Base Specification from the Command Message processing defined by this specification (refer to section 1.8.39).

1.8.32 NVMe Storage Device

A logical or physical component, device, or assembly that contains at least one NVM Subsystem or Expansion Connector, at least one Upstream Connector, and at least one FRU Information Device. An NVMe Storage Device that implements the out-of-band mechanism contains at least one Management Endpoint and a Controller Management Interface per Controller. An NVMe Storage Device contains zero or more PCIe switches and 2-Wire Muxes. An NVMe Storage Device shall comply with the NVM Express Base Specification. In this specification, NVMe Storage Devices shall also comply with this specification.

1.8.33 NVMe Storage Device FRU

An NVMe Storage Device that is able to be removed and replaced (e.g., by an end user or technician) without having to replace the entire system in which that NVMe Storage Device is contained. Examples of NVMe Storage Device Field-Replaceable Units include a U.2 PCIe SSD, a PCI Express Card Electromechanical add-in card, or an M.2 module.

1.8.34 NVMe Subenclosure (Subenclosure)

A portion of an NVMe Enclosure accessed through a primary NVMe Enclosure's Enclosure Services Process. Refer to SCSI Enclosure Services - 4 (SES-4) for more information.

1.8.35 NVMe-MI Message

A type of MCTP Message that is defined by this specification in sections 3.1 and 4.1. Refer to the MCTP IDs and Codes specification and the NVMe Management Messages over MCTP Binding Specification for more details on this type of MCTP Message (note that NVMe-MI Messages are referred to as NVM Express Management Messages over MCTP in these specifications).

1.8.36 NVM Subsystem

This specification extends the definition of an NVM Subsystem defined in the NVM Express Base Specification (e.g., by adding a Management Endpoint, Controller Management Interface, etc.). NVMe Enclosures and NVMe Storages devices that are not Carriers have one or more NVM Subsystems. Carriers have zero or more NVM Subsystems.

1.8.37 Out-of-Band

Per the Management Component Transport Protocol (MCTP) Overview White Paper, out-of-band management is management that operates with hardware resources and components that are independent of the operating system's control. The out-of-band communication paths supported by this specification are via MCTP over 2-Wire or MCTP over PCIe VDM from a Management Controller to a Management Endpoint. In addition, this specification supports the out-of-band access mechanism defined by the IPMI Platform Management FRU Information Storage Definition specification for accessing a FRU Information Device from a Management Controller over 2-Wire.

1.8.38 PCIe Reset

A mechanism used to reset one or more PCIe VDM Management Endpoints in an NVMe Storage Device or NVMe Enclosure. For more information, refer to section 8.3.5.

1.8.39 Process

This is the state when a Command Message is processed. Processing of a Command Message consists of checking for errors with the Command Message and performing the actions specified by the Command Message. This state is applicable in both the out-of-band mechanism and the in-band tunneling mechanism. Refer to section 4.2 for additional details on the Process state in the out-of-band mechanism. Refer to section 4.3 for additional details on the Process state in the in-band tunneling mechanism.

This specification uses the terms process/processing/processed to refer to actions performed in the Process state. These terms are distinct from the NVMe Processing term used to describe NVMe command processing as defined by the NVM Express Base Specification (refer to section 1.8.31 in this specification).

1.8.40 Request Message

An NVMe-MI Message originating from a Requester. A Request Message may be a Command Message or a Control Primitive.

1.8.41 Request-To-Response Time

The time it takes for a Management Endpoint to respond to a Request Message. This time is measured using a request-to-response timer on the Management Endpoint from the end of the reception of a Request Message to the beginning of the transmission of the first corresponding Response Message. The request-to-response timer is reset and restarted under certain conditions when the Pause Flag bit transitions from ‘1’ to ‘0’ (refer to section 4.2.2.1).

1.8.42 Requester

The entity that sends Request Messages and receives Response Messages. For the out-of-band mechanism, the Requester is a Management Controller. For the in-band tunneling mechanism, the Requester is the host.

1.8.43 Responder

The entity that receives Request Messages and sends back Response Messages. For the out-of-band mechanism, the Responder is a Management Endpoint. For the in-band tunneling mechanism, the Responder is an NVMe Controller.

1.8.44 Response Message

An NVMe-MI Message originating from a Responder in response to a Request Message. Response Messages may be used in both the out-of-band mechanism and the in-band tunneling mechanism.

1.8.45 Transmission Delay

The worst-case time it takes to transmit an NVMe-MI Message from a Requester to a Responder or vice versa. This time is measured from the end of the transmission of an NVMe-MI Message at the transmitter to the end of the reception of the NVMe-MI Message at the receiver. The Transmission Delay is dependent on the transport type and the host platform implementation.

1.8.46 Upstream Connector

A connector on the NVMe Storage Device or NVMe Enclosure to which a Requester attaches. It may be a physical connector as in U.2 form factors, solder balls as in a BGA form factor, or PCB trace fingers as in a CEM Add in Card or EDSFF form factor. An Upstream Connector may include multiple communications ports, control signals, and power supply rails.

1.8.47 Vendor ID

An identification value assigned by PCI-SIG to the PCI-SIG member company. If an NVMe Subsystem consists of parts from multiple vendors or a Vendor has multiple Vendor IDs, then for the purpose of this specification any of the Vendor IDs may be used, but the same Vendor ID should be used for all fields that require a Vendor ID.

1.8.48 VPD or Vital Product Data

Field-Replaceable Unit (FRU) Information which is stored in a FRU Information Device. This specification defines a standard VPD format for NVMe Storage Devices.

1.9 Keywords

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

1.9.1 mandatory

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

1.9.2 may

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

1.9.3 obsolete

A keyword indicating functionality that was defined in a previous version of this specification that has been removed.

1.9.4 optional

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

1.9.5 R

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

1.9.6 reserved

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

1.9.7 shall

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

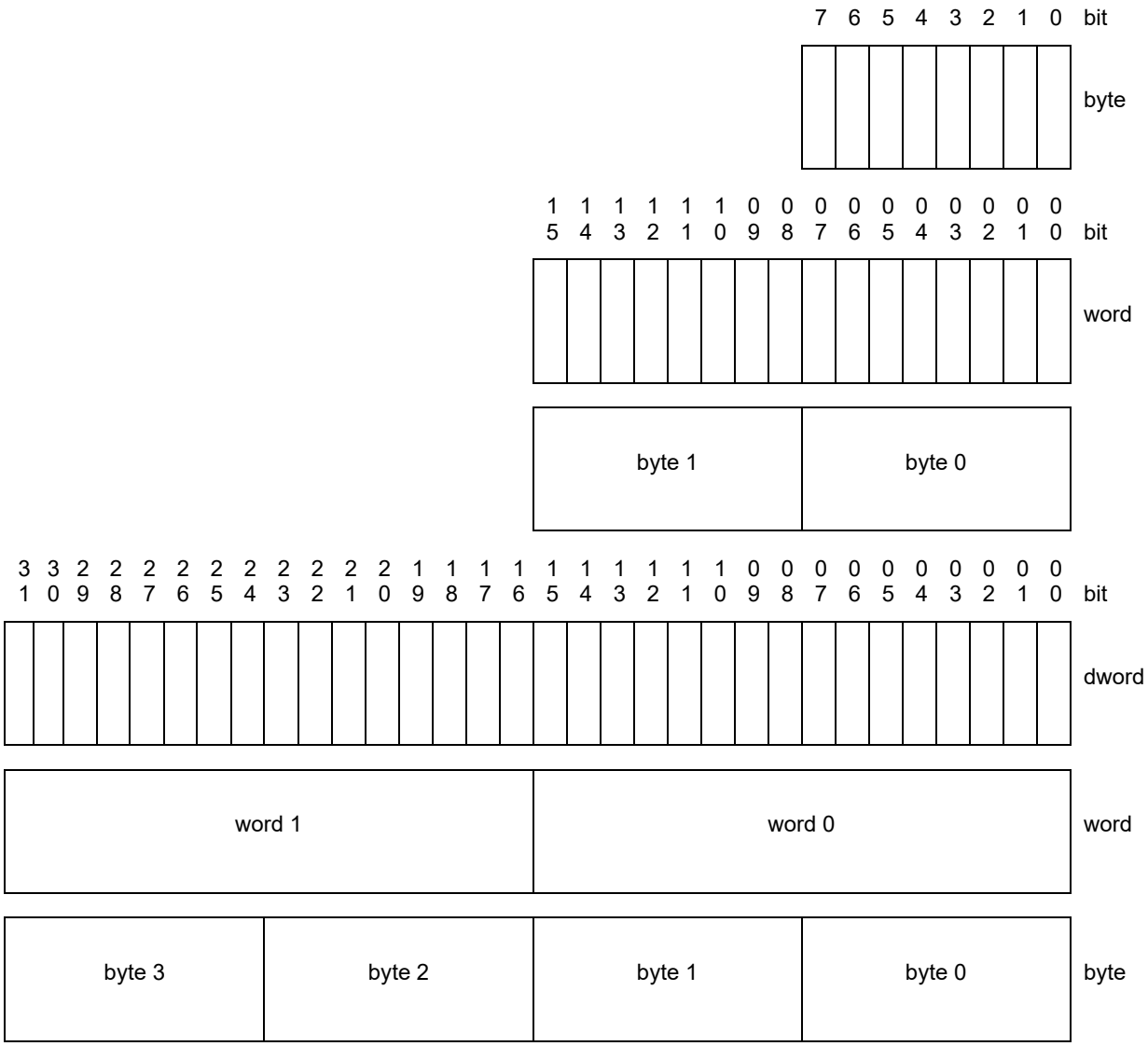
1.9.8 should

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

1.10 Byte, Word, and Dword Relationships

Figure 15 illustrates the relationship between bytes, words, and dwords. Unless otherwise stated, this specification specifies data in a little-endian format.

Figure 15: Byte, Word, and Dword Relationships



1.11 References

I2C Bus specification, revision 6.0. Available from https://www.i2c-bus.org.

IPMI Platform Management FRU Information Storage Definition 1.0, Version 1.3. Available from https://www.intel.com.

INCITS 555-2020 Information Technology – SCSI Enclosure Services – 4 (SES-4). Available from https://webstore.ansi.org.

MCTP Base Specification (DSP0236), Version 1.3.3. Available from https://www.dmtf.org/pmci.

MCTP I3C Transport Binding Specification (DSP0233), Version 1.0.1. Available from https://www.dmtf.org/pmci.

MCTP IDs and Codes (DSP0239), Version 1.11.0. Available from https://www.dmtf.org/pmci.

MCTP NVMe™ (NVM Express™) Management Messages over MCTP Binding specification (DSP0235), Version 1.0.1. Available from https://www.dmtf.org/pmci.

MCTP PCIe VDM Transport Binding Specification (DSP0238), Version 1.2.1. Available from https://www.dmtf.org/pmci.

MCTP SMBus/I2C Transport Binding Specification (DSP0237), Version 1.2.0. Available from https://www.dmtf.org/pmci.

MIPI I3C Basic℠ Specification v1.1.1. Available from https://www.mipi.org.

NVM Express Base Specification, Revision 2.3. Available from https://www.nvmexpress.org.

NVM Express Computational Programs Command Set Specification, Revision 1.2. Available from https://www.nvmexpress.org.

NVM Express Key Value Command Set Specification, Revision 1.3. Available from https://www.nvmexpress.org.

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

NVM Express NVMe over PCIe Transport Specification, Revision 1.3. Available from https://www.nvmexpress.org.

NVM Express Zoned Namespace Command Set Specification, Revision 1.4. Available from https://www.nvmexpress.org.

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

PCI-SIG PCI Express® Card Electromechanical Specification, Revision 5.1, Version 1.0. Available from https://www.pcisig.com.

PCI-SIG PCI Express® M.2 Specification, Revision 5.1. Available from https://www.pcisig.com.

PCI-SIG PCI Express® SFF-8639 Module Specification, Revision 5.0. Available from https://www.pcisig.com.

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

SNIA Native NVMe-oF™ Drive Specification, Version 1.1. Available from https://www.snia.org.

SNIA SFF-TA-1001 Universal x4 Link Definition for SFF-8639 Specification, Revision 1.1. Available from https://www.snia.org.

SNIA SFF-TA-1006 Enterprise and Datacenter 1U Short SSD Form Factor (E1.S) Specification, Revision 1.5. Available from https://www.snia.org.

SNIA SFF-TA-1007 Enterprise and Datacenter 1U Long SSD Form Factor (E1.L) Specification, Revision 1.2. Available from https://www.snia.org.

SNIA SFF-TA-1008 Enterprise and Datacenter Device Form Factor (E3) Specification, Revision 2.1. Available from https://www.snia.org.

SNIA SFF-TA-1009 Enterprise and Datacenter Standard Form Factor Pin and Signal Specification. Available from https://www.snia.org.

System Management Bus (SMBus) Specification, revision 3.2. Available from https://www.smbus.org.

2 Physical Layer

This section describes the physical layers supported by this specification for NVMe Storage Devices or NVMe Enclosures.

2.1 PCI Express

PCI Express is used as a physical layer in both the out-of-band mechanism and the in-band tunneling mechanism in this specification.

For the out-of-band mechanism, a PCIe port in an NVMe Storage Device or NVMe Enclosure may implement one or more Management Endpoints. If the PCIe port implements one or more Management Endpoints, then:

a) the PCIe port shall support MCTP over PCIe Vendor Defined Messages (VDMs) as specified by the MCTP PCIe VDM Transport Binding Specification;
b) a Management Endpoint should be associated with PCIe Function 0 on the upstream PCIe bus on the NVMe Storage Device or NVMe Enclosure; and
c) a Management Endpoint should not be associated with a PCIe virtual function (e.g., a PCIe SR-IOV Virtual Function).

For the in-band tunneling mechanism, a host issues NVMe Admin Commands (NVMe-MI Send and NVMe-MI Receive) to the NVMe Admin Queue over PCI Express. Refer to the NVM Express Base Specification and section 4.3 of this specification for additional details on the NVMe-MI Send and NVMe-MI Receive commands.

2.2 2-Wire

This section defines the requirements for an NVMe Storage Device or NVMe Enclosure that implements a 2-Wire port. The 2-Wire physical layer is only applicable for the out-of-band mechanism.

The 2-Wire port, protocols, and electricals are defined by multiple industry specifications. If an NVMe Storage Device or NVMe Enclosure implements an NVM Subsystem with a Management Endpoint associated with a 2-Wire port, then that port shall comply with the following specifications:

- PCI Express Base Specification;
- the applicable form-factor specification (e.g., U.2 or EDSFF);
- MCTP Base Specification;
- MCTP SMBus/I2C Transport Binding Specification; and
- SMBus Specification.

If the 2-Wire port also supports I3C mode, then that port shall also comply with the following specifications:

- MCTP I3C Transport Binding Specification; and
- MIPI I3C Basic Specification.

This section summarizes content from these other specifications and defines additional requirements for an NVMe Storage Device or NVMe Enclosure that implements an optional 2-Wire port.

The PCI Express Base Specification requires the 2-Wire port to default to SMBus mode after auxiliary or main power on and after specific resets. The PCI Express Base Specification also describes how to negotiate the 2-Wire Port into I3C mode for higher frequencies at lower power if supported by all elements on the bus.

An NVM Subsystem may also support the NVMe Basic Management Command for health and status polling while the 2-Wire port is in SMBus mode. The NVMe Basic Management Command is defined as an informative technical note in Appendix A, though not recommended for new designs.

Figure 16 lists 2-Wire elements that are supported on an NVMe Storage Device or NVMe Enclosure. For each 2-Wire element, the default 2-Wire address is provided as well as the conditions under which the 2-Wire element is required on an NVMe Storage Device or NVMe Enclosure. The presence or absence of Expansion Connectors on an NVMe Storage Device determines which of the two mutually exclusive 2-Wire addresses is used for the FRU Information Device. Using a different 2-Wire address for the FRU Information Device on NVMe Storage Devices that are Carriers versus non-Carriers avoids 2-Wire address conflict when Expansion Connectors are populated with NVMe Storage Devices.

Figure 16: 2-Wire Elements and Requirements

2-Wire Element	Default SMBus/I2C Address[3]		SMBus ARP Support	I3C Support	Required Element Presence
	Hex Format	Binary Format [1]			
FRU Information Device	A6h	1010_011xb	Optional	No	Required on an NVMe Storage Device with no Expansion Connectors. Undefined on NVMe Enclosures.
FRU Information Device	A4h	1010_010xb	Optional	No	Required on Carriers (i.e., an NVMe Storage Device with one or more Expansion Connectors). Undefined on NVMe Enclosures.
2-Wire Management Endpoint	3Ah	0011_101xb	Optional	Optional	Required if an NVMe Storage Device or NVMe Enclosure contains one or more 2-Wire Management Endpoints at the SMBus address. May also support I3C using a dynamically assigned address.
2-Wire Mux	E8h	1110_100xb	Optional	No	For NVMe Storage Devices, required if there is more than one 2-Wire element on any 2-Wire channel with the same 2-Wire address that does not support ARP. Undefined on NVMe Enclosures.
Basic Management Command [2]	D4h	1101_010xb	Optional	No	For NVMe Storage Devices, not recommended for new designs. Undefined on NVMe Enclosures.

Notes:
1. The x represents the 2-Wire read/write bit.
2. The NVMe Basic Management Command is defined in Appendix A as an informative technical note.
3. Per the PCI Express Base Specification, the SMBus/I2C addresses are not ACKed in I3C mode and I3C addresses are disabled in SMBus mode.

Host platforms expecting to be used with one or more Management Endpoints (e.g., data center platforms and workstations) often isolate 2-Wire ports with separate channels to avoid conflicts. Address conflicts may occur on platforms that do not isolate 2-Wire ports (e.g., some client platforms).

SMBus-capable elements may support Address Resolution Protocol (ARP) to assign dynamic addresses for SMBus communications and eliminate address conflicts. If an NVMe Storage Device or NVMe Enclosure contains more than one 2-Wire element with the same default SMBus/I2C address that are not isolated from each other, then all such elements shall support ARP. 2-Wire elements that support ARP should be implemented as DTA devices (refer to the SMBus Specification). These NVMe Storage Devices should not issue “Notify ARP Controller” commands.

If the Get UDID command is supported by an NVM Subsystem, then all 2-Wire elements associated with that NVM Subsystem shall use the Unique Device Identifier (UDID) shown in Figure 17. The UDID and the Get UDID command are defined by the SMBus Address Resolution Protocol (ARP) (refer to the SMBus Specification). The UDID Type field allows up to four 2-Wire elements to be grouped together within the same NVM Subsystem if their SMBus UDID has the same Vendor ID, Device ID, and UDID Device ID. This fact may be used by the Management Controller to associate a Management Endpoint on the 2-Wire port

with the corresponding FRU Information Device while in SMBus mode. If the 2-Wire port supports I3C mode, then the least-significant 30-bits of the I3C Device ID (refer to Figure 18) shall match the UDID Device ID. I3C uses the Device Configuration Register (DCR) to differentiate multiple device types instead of the UDID Type field. The SMBus UDID and I3C Provisioned ID shall be globally unique identifiers to prevent unreconcilable address assignment issues.

If the Upstream Connector has a 2-Wire port, then the FRU Information Device associated with that connector shall be present directly on the 2-Wire port connected to the Upstream Connector while in SMBus mode.

Clock stretching is allowed by the Management Controller, Management Endpoint, and the FRU Information Device when operating in SMBus mode. However, implementations are strongly discouraged from using clock stretching so that communications are more predictable with higher throughput. Clock stretching is prohibited when operating in I3C mode.

The PCI Express Base Specification describes how the 2-Wire port may be switched by the Management Controller from the default SMBus mode to I3C mode for higher frequencies at lower voltages. A quick summary is that only 2-Wire elements which support I3C respond with an ACK to address FCh. After this ACK, the SMBus/I2C functionality on that 2-Wire port is disabled and the Management Controller may scan for any remaining SMBus/I2C only elements. If no SMBus/I2C responders are found, the Management Controller should transmit the address FCh a second time to enter I3C mode at the lower voltage of 1.8 V. Note that the MIPI I3C Basic Specification uses the 7-bit representation of the address (i.e., 7Eh) while this specification uses the 8-bit representation of the address (i.e., FCh). If a 2-Wire port is in I3C mode, then some 2-Wire Resets (refer to section 8.3.4) shall reset the 2-Wire port into SMBus mode.

If the 2-Wire port on an NVM Subsystem is successfully switched to I3C mode, then all internal elements on the NVMe Subsystem's 2-Wire port are either switched to I3C mode or disabled. Whenever the 2-Wire port enters I3C mode, the I3C elements still ACK address FCh, but the unique address for each I3C element is disabled. Refer to the MIPI I3C Basic Specification for Common Command Codes (CCC) like ENTDAA which is used to assign a unique address. I3C capable NVM Subsystems shall support the ENTDAA CCC which is used to assign dynamic addresses for I3C communications similarly to how ARP is used to assign addresses for SMBus communications.

I3C traffic has similar bit patterns to SMBus traffic. The I3C address byte tolerates collisions as SMBus does for arbitration and prioritization. However, after the address byte the bus is driven by CMOS push/pull drivers for higher frequencies. To achieve these frequencies the bus direction does not reverse for an ACK bit on every byte like SMBus. The ACK bit is renamed to T-bit for use as bit parity on writes and as a transmission complete signal on reads. The clock is always driven by the Management Controller.

The Management Endpoint initiates traffic by asserting the data line and waiting for the Management Controller to start clocking out the Response Message or AEM. This is called an in-band interrupt (IBI) and the feature is enabled by default upon entering I3C mode. Note that because the 2-Wire port starts out in SMBus mode the Hot Join IBIs for I3C are never sent. The MCTP I3C Transport Binding Specification describes how IBIs, Private Reads, and Private Writes are used to transfer MCTP packets.

Typical I3C elements only respond to one dynamically assigned address and use a Mandatory Data Byte (MDB) to describe the traffic type. Management Endpoints shall use an MDB value of AEh and a DCR value of CCh as defined for MCTP by MIPI. The I3C MTU shall default to 64 bytes per the MCTP I3C Transport Binding Specification. Maximum bus speed and MTU negotiation for I3C traffic is described in the MIPI I3C Basic Specification. The PCI Express Base Specification indicates the required and recommended I3C features.

When a NACK is received, a Management Endpoint shall follow the appropriate MCTP transport binding specification for the current mode of operation on the 2-Wire port.–The Management Endpoint treats a STOP condition due to excessive NACKs as an implicit Pause Control Primitive. Refer to section 4.2.1.1.

Figure 17: SMBus Element UDID

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>127:120</td><td>Device Capabilities (DC): This field describes the device capabilities.

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:6</td><td>Address Type (ADDRTYP): This field describes the type of address contained in the device. Refer to the MCTP SMBus/I2C Transport Binding Specification.</td></tr>
<tr><td>5:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>PEC Supported (PECS): All MCTP transactions shall include a Packet Error Code (PEC) byte. This bit shall be set to ‘1’ to indicate support for PEC.</td></tr>
</table></td></tr>
<tr><td>119:112</td><td>Version and Revision (VERREV): This field is used to identify the UDID version and silicon revision.

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:6</td><td>Reserved</td></tr>
<tr><td>5:3</td><td>UDID Version (UDIDV): This field specifies the UDID version and shall be set to 001b.</td></tr>
<tr><td>2:0</td><td>Silicon Revision ID (SRID): This field is used to specify a vendor specific silicon revision level.</td></tr>
</table></td></tr>
<tr><td>111:96</td><td>Vendor ID (VID): This field contains the PCI-SIG Vendor ID for the NVM Subsystem.</td></tr>
<tr><td>95:80</td><td>Device ID (DID): This field contains a vendor assigned Device ID for the NVM Subsystem.</td></tr>
<tr><td>79:64</td><td>Interface (ITFC): This field defines the SMBus version and the Interface Protocols supported.

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>15:08</td><td>Reserved</td></tr>
<tr><td>07</td><td>ZONE (ZN): This bit shall be cleared to ‘0’.</td></tr>
<tr><td>06</td><td>Intelligent Platform Management Interface (IPMI): This bit shall be cleared to ‘0’.</td></tr>
<tr><td>05</td><td>Alert Standard Format (ASF): This bit shall be set to ‘1’. Refer to the MCTP SMBus/I2C Transport Binding Specification.</td></tr>
<tr><td>04</td><td>Original Equipment Manufacturer (OEM): This bit shall be set to ‘1'.</td></tr>
<tr><td>03:00</td><td>SMBus Version (SVER): This field shall be set to 4h for SMBus Version 2.0, or to 5h for SMBus Version 3.0, 3.1, and 3.2.</td></tr>
</table></td></tr>
<tr><td>63:48</td><td>Subsystem Vendor ID (SVID): This field contains the PCI-SIG Vendor ID for the NVM Subsystem.</td></tr>
<tr><td>47:32</td><td>Subsystem Device ID (SDID): This field contains a vendor assigned Device ID for the NVM Subsystem.</td></tr>
<tr><td>31:00</td><td>UDID Vendor Specific ID (UVSID): This field ensures all UDIDs from a vendor are unique and is used to associate elements implemented within an NVMe Storage Device or NVMe Enclosure.

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:30</td><td>UDID Type (UDTYP): This field distinguishes which NVM Subsystem that implements multiple SMBus elements is providing the UDID. Note that Management Controllers compliant to version 1.0 of this specification may be incompatible with NVM Subsystems using values 1h and 3h.

<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0h</td><td>FRU Information Device</td></tr>
<tr><td>1h</td><td>2-Wire Mux</td></tr>
<tr><td>2h</td><td>Management Endpoint</td></tr>
<tr><td>3h</td><td>Vendor Specific Devices</td></tr>
</table></td></tr>
<tr><td>29:00</td><td>UDID Device ID (UDDID): This field indicates a unique vendor assigned ID for the NVM Subsystem. This field shall contain a value that results in a unique UDID as specified by the SMBus Specification, and remains static during the life of the NVM Subsystem.</td></tr>
</table></td></tr>
</table>

Figure 18: I3C Provisioned ID

Bits	Description
47:33	**MIPI Manufacturer ID (MIPIMID):** This field shall indicate the lower 15 bits of the MIPI Manufacturer ID.

Figure 18: I3C Provisioned ID

Bits	Description
32	**Provisioned ID Type (PIDT):** This bit should be cleared to '0' to indicate that the Device ID is not a random value.
31:00	**Device ID (DID):** This field shall indicate a value that results in a unique I3C Provisioned ID as specified by the MIPI I3C Basic Specification and remains static during the life of the NVM Subsystem. If SMBus ARP is also supported, then the value in the least-significant 30 bits shall match UDID Device ID.

3 Message Transport

This specification defines an interface that supports multiple message transports. The message format is the same for the out-of-band mechanism and the in-band tunneling mechanism as described in section 3.1. The out-of-band message transport is described in section 3.2. The in-band tunneling message transport is described in section 3.3.

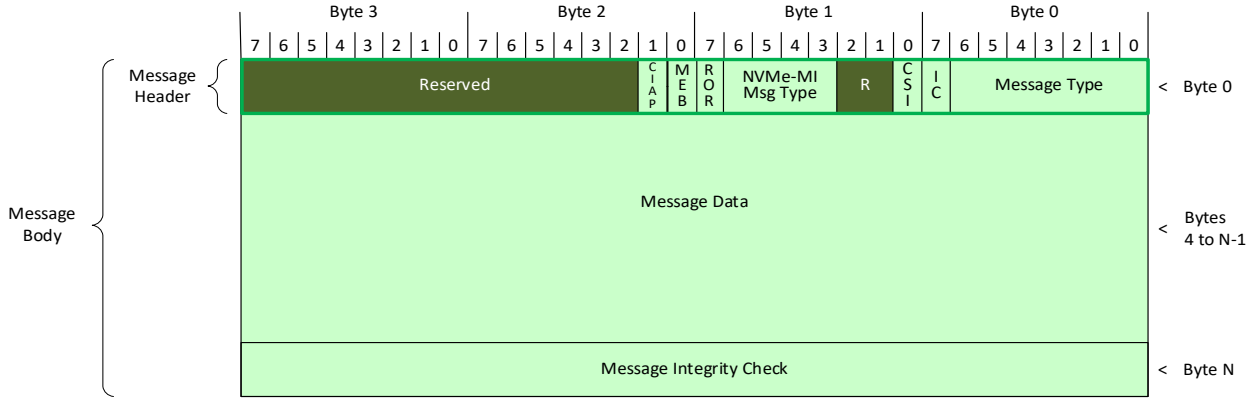
3.1 NVMe-MI Messages

NVMe-MI Messages are used in both the out-of-band mechanism and the in-band tunneling mechanism. The format of an NVMe-MI Message is shown in Figure 19 and Figure 20.

In the out-of-band mechanism, an NVMe-MI Message consists of the payload of one or more MCTP packets. The maximum sized NVMe-MI Message is 4,224 bytes (i.e., 4 KiB + 128 bytes). Refer to the NVMe Management Messages over MCTP Binding Specification. NVMe-MI Messages with lengths greater than 4,224 bytes are considered invalid NVMe-MI Messages. Refer to section 4.2 for details on how NVMe-MI Messages are used in the out-of-band mechanism.

In the in-band tunneling mechanism, NVMe-MI Messages are not split into MCTP packets and the maximum NVMe-MI message size is equal to the Maximum Data Transfer Size (refer to the NVM Express Base Specification). Refer to section 4.3 for details on how NVMe-MI Messages are used in the in-band tunneling mechanism.

Figure 19: NVMe-MI Message



3.1.1 Message Fields

The format of an NVMe-MI Message consists of a Message Header in the first dword, followed by the Message Data. If the Integrity Check (IC) bit is set to ‘1’, then the NVMe-MI Message ends with the Message Integrity Check as shown in Figure 19.

The Message Header contains a Message Type (MT) field and an Integrity Check (IC) bit that are defined by the MCTP Base Specification. The Message Type field specifies the type of payload contained in the message body and is set to 4h in all NVMe-MI Messages (refer to the MCTP IDs and Codes specification). The Integrity Check (IC) bit indicates whether the NVMe-MI Message is protected by a Message Integrity Check. All NVMe-MI Messages in the out-of-band mechanism are protected by a 32-bit CRC computed over the Message Body contents. The IC bit is set to ‘1’ in all NVMe-MI Messages in the out-of-band mechanism. The Integrity Check (IC) bit is cleared to ‘0’ in all NVMe-MI Messages in the in-band tunneling mechanism.

The Request or Response (ROR) bit in the Message Header specifies whether the NVMe-MI Message is a Request Message or a Response Message. The ROR bit is not applicable to Asynchronous Event Messages. The NVMe-MI Message Type (NMIMT) field specifies whether the Request Message is a

Control Primitive or a specific type of Command Message (refer to Figure 26). The Command Slot Identifier (CSI) bit specifies the Command Slot with which the NVMe-MI Message is associated in the out-of-band mechanism. Refer to section 4.2 for additional information about Command Slots.

If the Management Endpoint supports the Management Endpoint Buffer (i.e., the Management Endpoint Buffer Size field is set to a non-zero value), then in the out-of-band mechanism, the Management Endpoint Buffer (MEB) bit in the Message Header specifies whether Message Data is contained in the associated Message Data field of an NVMe-MI Message or in the Management Endpoint Buffer.

The Command Initiated Auto Pause Supported (CIAPS) bit in Figure 114 indicates if the port supports the Command Initiated Auto Pause (CIAP) bit in Command Messages. If the Command Initiated Auto Pause bit is supported (i.e., the CIAPS bit is set to '1'), then in the out-of-band mechanism, the Command Initiated Auto Pause (CIAP) bit in the Message Header of a Command Message specifies whether or not the Management Endpoint is automatically paused when a Command Message enters the Process state. A Command Message in the out-of-band mechanism with the CIAPS bit set to '1' and with the CIAP bit set to '1' shall be treated by the Management Endpoint as if an implicit Pause Control Primitive, as described in section 4.2.1.1, was received in the Process state with the exception that the Management Endpoint shall not transmit a Control Primitive Response Message.

Figure 20: NVMe-MI Message Fields

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>0</td><td>MCTP Data (MCTPD): This field contains the Message Type and Integrity Check fields as defined by the MCTP Base Specification.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Integrity Check (IC): If the MCTP message is covered by an overall MCTP message payload integrity check, then this bit is set to '1'. If the MCTP message is not covered by an overall MCTP message payload integrity check, then this bit is cleared to '0'.

For Request Messages in the out-of-band mechanism, this bit should be set to '1'. For Response Messages in the out-of-band mechanism and Asynchronous Event Messages, this bit shall be set to '1'.

For Request Messages in the in-band tunneling mechanism, this bit should be cleared to '0'. For Response Messages in the in-band tunneling mechanism this bit shall be cleared to '0'.</td></tr>
<tr><td>6:0</td><td>Message Type (MT): This field contains the message type. For Request Messages, this field should be set to 4h. For Response Messages and Asynchronous Event Messages, this field shall be set to 4h. Refer to the MCTP IDs and Codes specification.</td></tr>
</table></td></tr>
</table>

Figure 20: NVMe-MI Message Fields

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>1</td><td>
NVMe-MI Message Parameters (NMP): This field contains parameters applicable to the NVMe-MI Message.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Request or Response (ROR): For Request Messages, this bit should be cleared to ‘0’.
For Response Messages, this bit shall be set to ‘1’.
For Asynchronous Event Messages, this field is not applicable and shall be cleared to ‘0’.</td></tr>
<tr><td>6:3</td><td>NVMe-MI Message Type (NMIMT): For Request Messages, this field should specify the type of the NVMe-MI Message. For Response Messages, this field shall indicate the type of the NVMe-MI Message. For Asynchronous Event Messages, this field shall indicate a value of 5h.
Refer to the sections referenced in the table below for details about each NVMe-MI Message Type and whether they apply to the out-of-band mechanism, the in-band tunneling mechanism, or both.
<table>
<tr><th colspan="3">NVMe-MI Message Type</th></tr>
<tr><th>Value</th><th>Description</th><th>Reference Section</th></tr>
<tr><td>0h</td><td>Control Primitive</td><td>4.2.1</td></tr>
<tr><td>1h</td><td>NVMe-MI Command</td><td>5</td></tr>
<tr><td>2h</td><td>NVMe Admin Command</td><td>6</td></tr>
<tr><td>3h</td><td>Reserved</td><td>-</td></tr>
<tr><td>4h</td><td>PCIe Command</td><td>7</td></tr>
<tr><td>5h</td><td>Asynchronous Event</td><td>4.1.3</td></tr>
<tr><td>6h to Fh</td><td>Reserved</td><td>-</td></tr>
</table>
</td></tr>
<tr><td>2:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>Command Slot Identifier (CSI): For Request Messages in the out-of-band mechanism, this bit specifies the Command Slot with which the Request Message is associated. For Response Messages in the out-of-band mechanism, this bit shall indicate the Command Slot associated with the Request Message with which the Response Message is associated.
For Request Messages in the in-band tunneling mechanism this bit is not applicable and shall be ignored by the Management Endpoint.
For Response Messages in the in-band tunneling mechanism, this bit is not applicable and shall be cleared to ‘0’.
For Asynchronous Event Messages, this field is not applicable and shall be cleared to ‘0’.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0b</td><td>Command Slot 0</td></tr>
<tr><td>1b</td><td>Command Slot 1</td></tr>
</table>
</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:2</td><td>Reserved</td></tr>
<tr><td>1</td><td>Command Initiated Auto Pause (CIAP): If this bit is set to ‘1’ in a Command Message and the Command Initiated Auto Pause Supported (CIAPS) bit is set to ‘1’, then the Management Endpoint shall be automatically paused when the Command Message enters the Process state. If this bit is cleared to ‘0’ in a Command Message, the Management Endpoint shall not be automatically paused when the Command Message enters the Process state. The usage requirements for this bit are as follows:
<table>
<tr><th colspan="2">Mechanism</th><th>CIAP Value</th><th>Usage Requirement</th></tr>
<tr><td rowspan="4">Out-of-band</td><td rowspan="2">Command Messages</td><td>0</td><td>This value is permitted.</td></tr>
<tr><td>1</td><td>If the CIAPS bit is set to ‘1’, then a value of ‘1’ for the CIAP bit is permitted.
If the CIAPS bit is cleared to ‘0’, then:
a) a value of ‘1’ for the CIAP bit is prohibited; and
b) if a value of ‘1’ for the CIAP bit is received in a Command Message, then an Invalid Parameter Error Response with the PEL field indicating this bit shall be returned.</td></tr>
<tr><td rowspan="2">Any NVMe-MI Message other than a Command Message</td><td>0</td><td>This value is required.</td></tr>
<tr><td>1</td><td>This value is prohibited. If this value is received in a Control Primitive, then an Invalid Parameter Error Response with the PEL field indicating this bit shall be returned.</td></tr>
<tr><td colspan="2" rowspan="2">In-band Tunneling</td><td>0</td><td>This value is required.</td></tr>
<tr><td>1</td><td>This value is prohibited. If this value is received in a Request Message, then an Invalid Parameter Error Response with the PEL field indicating this bit shall be returned.</td></tr>
</table></td></tr>
<tr><td>0</td><td>Management Endpoint Buffer (MEB): This bit indicates whether the Message Data in a Command Message is contained in the Message Data field of this NVMe-MI Message or in the Management Endpoint Buffer. Refer to section 3.1.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0b</td><td>The Message Data is contained in the Message Data of this NVMe-MI Message.</td></tr>
<tr><td>1b</td><td>If the Management Endpoint supports the Management Endpoint Buffer, then the Message Data is contained in the Management Endpoint Buffer.</td></tr>
</table>
The usage requirements for this bit are as follows:
<table>
<tr><th colspan="2">Mechanism</th><th>MEB Value</th><th>Usage Requirement</th></tr>
<tr><td rowspan="3">Out-of-band</td><td rowspan="3">Command Messages</td><td>0</td><td>This value is permitted.</td></tr>
<tr><td>1</td><td>If the Management Endpoint supports the Management Endpoint Buffer, then a value of ‘1’ for the MEB bit is permitted.
If the Management Endpoint does not support the Management Endpoint Buffer, then:
a) a value of ‘1’ for the MEB bit is prohibited; and</td></tr>
</table></td></tr>
</table>

Figure 20: NVMe-MI Message Fields

Bytes	Description			
				b) if a value of '1' for the MEB bit is received in a Command Message, then an Invalid Parameter Error Response with the PEL field indicating this bit shall be returned.
		Any NVMe-MI Message other than a Command Message	0	This value is required.
			1	This value is prohibited. If this value is received in a Control Primitive, then an Invalid Parameter Error Response with the PEL field indicating this bit shall be returned.
		In-band Tunneling	0	This value is required.
			1	This value is prohibited. If this value is received in a Request Message, then an Invalid Parameter Error Response with the PEL field indicating this bit shall be returned.
3	Reserved			
N-1:4	**Message Data (DATA):** This field contains the NVMe-MI Message payload. The format of this field depends on the NVMe-MI Message Type.			
N+3:N	**Message Integrity Check (MIC):** If the Integrity Check (IC) bit is set to '1' in a Request Message, then this field should specify a CRC computed over the contents of the NVMe-MI Message. If the Integrity Check (IC) bit is set to '1' in a Response Message or an Asynchronous Event Message, then this field shall indicate a CRC computed over the contents of the NVMe-MI Message. Refer to section 3.1.1.1. If the IC bit is cleared to '0' in a Request Message, then this field should not be included in the NVMe-MI Message. If the IC bit is cleared to '0' in a Response Message or an Asynchronous Event Message, then this field shall not be included in the NVMe-MI Message. This field is byte aligned.			

3.1.1.1 Message Integrity Check

If the Integrity Check (IC) bit is set to '1', then the Message Integrity Check field contains a 32-bit CRC computed over the contents of the NVMe-MI Message. The 32-bit CRC required by this specification is CRC-32C (Castagnoli) which uses the generator polynomial 1EDC6F41h. The Message Integrity Check is calculated using the following Rocksoft™ Model CRC Algorithm parameters defined in Figure 21.

Figure 21: Rocksoft™ Model CRC Algorithm parameters

Parameter	Value
Name	"CRC-32C"
Width	32
Poly	1EDC6F41h
Init	FFFFFFFFh
RefIn	True
RefOut	True
XorOut	FFFFFFFFh
Check	E3069283h

When sending a message, the Message Integrity Check shall be calculated using the following procedure or a procedure that produces an equivalent result:

1. Initialize the CRC register to FFFFFFFFh. This is equivalent to inverting the least-significant 32 bits of the NVMe-MI Message (Dword 0 in Figure 19);
2. Append 32 bits of 0's to the end of the Message Data to allow room for the Message Integrity Check (Dword N in Figure 19). This results in the Message Body shown in Figure 19 with the Message Integrity Check field cleared to 0h;
3. Map the bits in the Message Body from step 2 to the coefficients of the message polynomial M(x). Assume the length of M(x) is Y bytes. Bit 0 of byte 0 in the Message Body is the most-significant bit of M(x), followed by bit 1 of byte 0, on through to bit 7 of byte Y - 1. Note that the bits within each byte are reflected (i.e., bit n of each byte is mapped to bit (7 - n) resulting in bit 7 to bit 0, bit 6 to bit 1, and so on);

Figure 22: Message Integrity Check Example

	Message Body (Length = Y bytes)																								
	Byte 0								Byte 1								…	Byte Y - 1							
M(x) =	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	…	0	1	2	3	4	5	6	7

4. Divide the polynomial M(x) by the generator polynomial 1EDC6F41h to produce the 32-bit remainder polynomial R(x);
5. Reflect each byte of R(x) (i.e., bit n of each byte is mapped to bit (7 - n) resulting in bit 7 to bit 0, bit 6 to bit 1, and so on) to produce the polynomial R′(x);
6. Invert R′(x) to produce the polynomial R″(x); and
7. Store R″(x) in the Message Integrity Check field of the Message Body.

Upon receipt of an NVMe-MI Message, the Message Integrity Check may be validated as follows:

1. Save the received Message Integrity Check;
2. Initialize the CRC register to FFFFFFFFh. This is equivalent to inverting the least-significant 32 bits of the NVMe-MI Message (Dword 0 in Figure 19);
3. Clear the Message Integrity Check field to 0h;
4. Map the bits in the Message Body to the coefficients of the message polynomial M(x) as described in step 3 in the Message Integrity Check calculation procedure above;
5. Divide the polynomial M(x) by the generator polynomial 1EDC6F41h to produce the 32-bit remainder polynomial R(x);
6. Reflect each byte of R(x) (i.e., bit n of each byte is mapped to bit (7 - n) resulting in bit 7 to bit 0, bit 6 to bit 1, and so on) to produce the polynomial R′(x);
7. Invert R′(x) to produce the polynomial R″(x); and
8. Compare R″(x) from step 7 to the Message Integrity Check value saved in step 1. If both values are equal, the Message Integrity Check passes.

Refer to Appendix B for artificial messages and their corresponding Message Integrity Check values.

Refer to section 4.2.1.5 for special requirements on how to construct the Response Message when the Management Controller issues a Replay Control Primitive with a non-zero Response Replay Offset.

3.2 Out-of-Band Message Transport

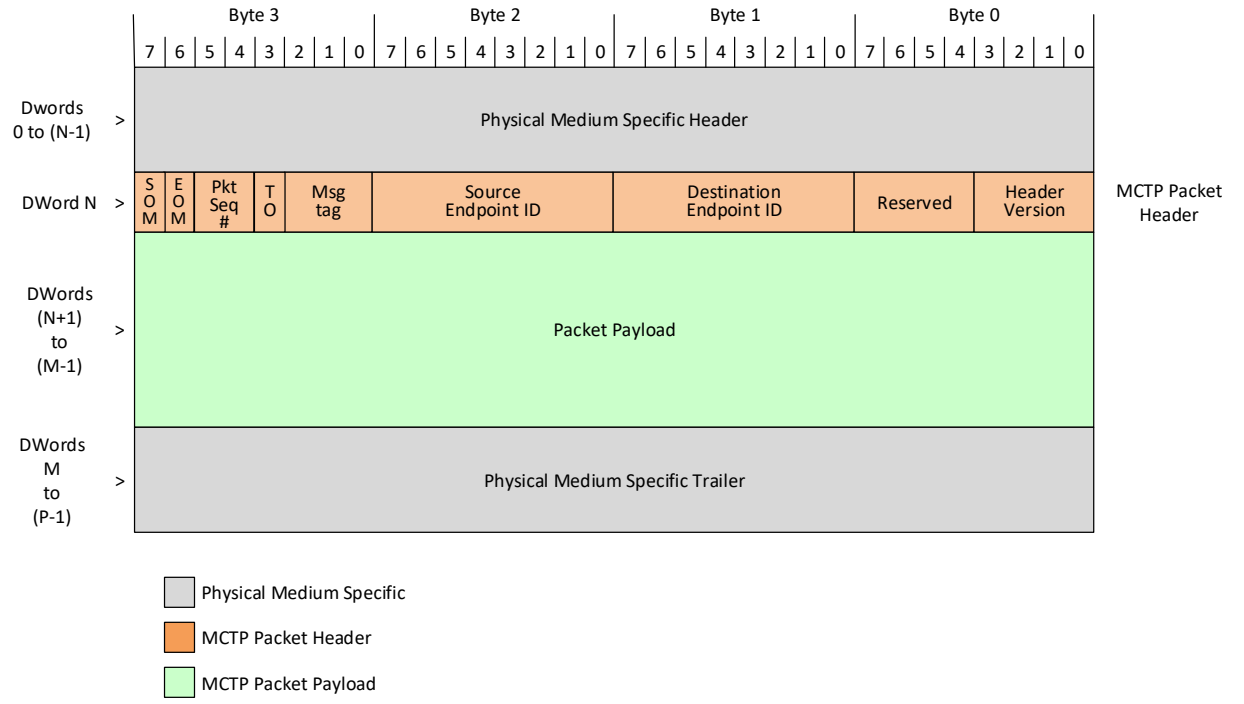
The out-of-band mechanism defined in this specification utilizes MCTP for in-order message transport between a Management Controller and a Management Endpoint.

A Management Endpoint compliant to this specification shall implement all required behaviors detailed in the Management Component Transport Protocol (MCTP) Base Specification and corresponding MCTP transport binding specification in addition to the requirements outlined in this specification (e.g., the Message Integrity Check algorithm).

3.2.1 MCTP Packet

In the MCTP Base Specification, the smallest unit of data transfer is the MCTP packet. One or more packets are combined to create an MCTP message. In this specification, the MCTP messages are referred to as NVMe-MI Messages (refer to section 1.8.35). Refer to section 3.2.1.1 for details on how MCTP packets are assembled into NVMe-MI Messages. An MCTP Packet Payload contains at least 1 byte but shall not exceed the negotiated MCTP Transmission Unit Size. The format of an MCTP Packet Payload is shown in Figure 23.

Figure 23: MCTP Packet Format



MCTP specifications use big endian byte ordering while NVM Express specifications use little endian byte ordering. All figures in this specification are illustrated with little endian byte ordering. Note that this pictorial representation does not change the order that bytes are sent out on the physical layer.

The Physical Medium-Specific Header and Physical Medium-Specific Trailer are defined by the MCTP transport binding specification utilized by the port. Refer to the MCTP transport binding specifications.

The Management Component Transport Protocol (MCTP) Base Specification defines the MCTP packet header (refer to DSP0236 for field descriptions). The fields of an MCTP Packet are shown in Figure 24.

Figure 24: MCTP Packet Fields

Field Name	Field Size
Medium-Specific Header	varies
Header Version	4 bits
Reserved	4 bits
Destination Endpoint ID	8 bits
Source Endpoint ID	8 bits
Msg tag (Message Tag)	3 bits
TO	1 bit
Pkt Seq #	2 bits

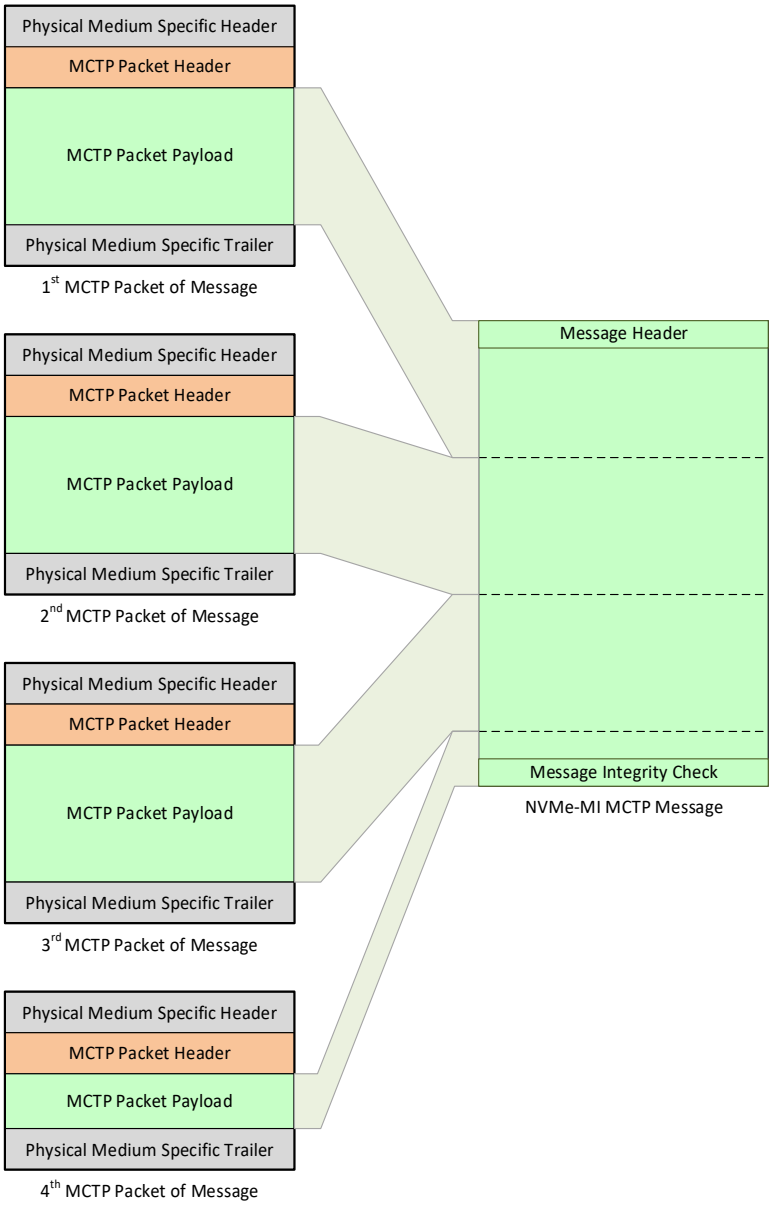
Figure 24: MCTP Packet Fields

EOM	1 bit
SOM	1 bit
Packet Payload	varies
Medium-Specific Trailer	varies

3.2.1.1 Packet Assembly into Messages

An NVMe-MI Message may be split into multiple MCTP Packet Payloads and sent as a series of packets. An example NVMe-MI Message whose contents are split across four MCTP packets is shown in Figure 25. Refer to the MCTP Base Specification for packetization and message assembly rules.

Figure 25: NVMe-MI Message Spanning Multiple MCTP Packets



In addition to the requirements outlined in the MCTP Base Specification and MCTP transport binding specifications, this specification has the following additional requirements:

- with the exception of the last packet in a Response Message or Asynchronous Event Message, the MCTP Transmission Unit size of all packets in a given Response Message or Asynchronous Event Message shall be equal to the negotiated MCTP Transmission Unit Size;
- the MCTP Transmission Unit size of the last packet in a Response Message or Asynchronous Event Message (i.e., the one with the EOM bit set in the MCTP header) shall be the smallest size required to transfer the MCTP Packet Payload for that Packet with no additional padding beyond any padding required by the physical medium-specific trailer; and
- once a complete Request Message has been assembled, the Message Integrity Check shall be verified. If the Message Integrity Check passes, then the Request Message shall be processed. If

the Message Integrity Check fails, then the Request Message shall be discarded. Refer to section 4.2.

3.2.2 Out-of-Band Error Handling

The Management Endpoint shall drop (silently discard) packets for error conditions as specified in the MCTP Base Specification. Some example conditions which result in discarding packets include unexpected middle or end packets. Silently discarded packets also cause the corresponding bit in the Get State Control Primitive Success Response field to be set to '1' (refer to Figure 45).

3.3 In-Band Tunneling Message Transport

The in-band tunneling mechanism in this specification utilizes the NVMe Admin Commands NVMe-MI Send and NVMe-MI Receive as a message transport. Refer to the NVM Express Base Specification and section 4.3 of this specification for additional details on the NVMe-MI Send and NVMe-MI Receive commands.

4 Message Servicing Model

This specification defines multiple message servicing models:

a) the out-of-band Request Message servicing model (refer to section 4.2);
b) the in-band tunneling Request Message servicing model (refer to section 4.3); and
c) the AEM servicing model (refer to section 4.4).

NVMe-MI Messages (refer to section 4.1) are used for communication in all message servicing models.

4.1 NVMe-MI Messages

Figure 26 illustrates the taxonomy of NVMe-MI Messages. The three main categories of NVMe-MI Messages are Request Messages (refer to section 4.1.1), Response Messages (refer to section 4.1.2), and Asynchronous Event Messages (AEMs, refer to section 4.1.3).

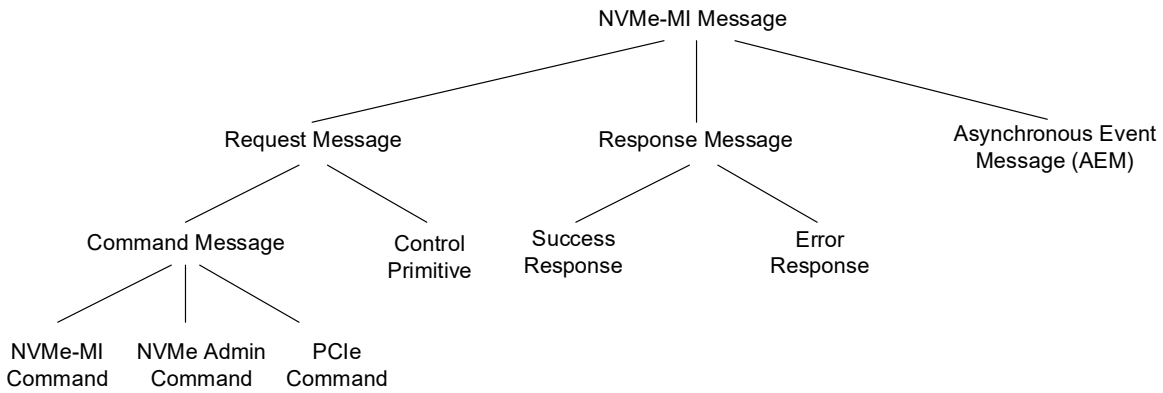
In the out-of-band Request Message servicing model, Request Messages are transmitted by a Management Controller to a Management Endpoint. In the in-band Request Message servicing model, Request Messages are transmitted by a host to an NVMe Controller. The entity transmitting the Request Message is collectively referred to as the Requester and the entity receiving the Request Message is collectively referred to as the Responder. After receiving a Request Message, the Responder processes the Request Message. When processing is complete, the Responder transmits a Response Message back to the Requester.

A Request Message is a Command Message or a Control Primitive. A Command Message specifies an operation to be performed by the Responder and is an NVMe-MI Command, an NVMe Admin Command, or a PCIe Command. Control Primitives are used in the out-of-band mechanism to affect the servicing of a previously issued Command Message or get the state of a Command Slot and Management Endpoint (refer to section 4.2.1).

A Response Message is a Success Response or an Error Response.

In the AEM servicing model, AEMs are transmitted by a Management Endpoint to a Management Controller using the out-of-band mechanism after one or more Asynchronous Events (AEs) occur. AEMs are prohibited in the in-band tunneling mechanism.

Figure 26: NVMe-MI Message Taxonomy



4.1.1 Request Messages

Request Messages specify an action to be performed by the Responder. The NMIMT field specifies the Request Message type. The format of the Message Body is determined by the Request Message types as follows:

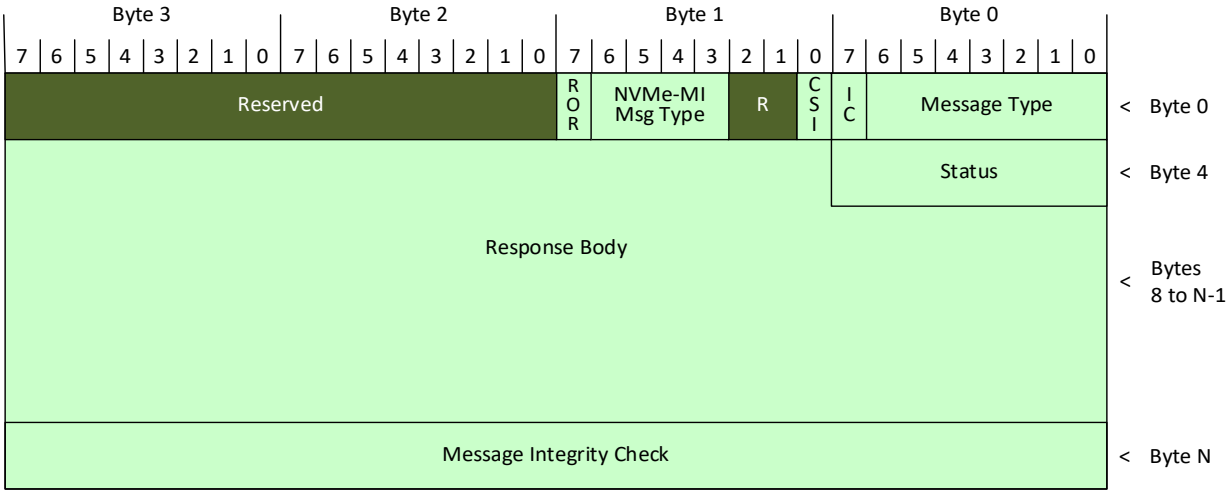
- Control Primitives (refer to section 4.2.1);
- The Management Interface Command Set (refer to section 5);
- The NVM Express Admin Command Set (refer to section 6); and
- The PCIe Command Set (refer to section 7).

4.1.2 Response Messages

Response Messages are NVMe-MI Messages that are generated when a Responder has processed a previously issued Request Message.

The format of a Response Message is shown in Figure 27 and Figure 28. The first dword contains the Message Header. The Status field encodes the status associated with the Response Message. This is followed by the Response Body whose format is NVMe-MI Message Type and Response Message Status specific. Finally, if the Integrity Check (IC) bit is set to '1', then the Response Message ends with the NVMe-MI Message Integrity Check field.

Figure 27: Response Message Format



In the out-of-band mechanism, the CSI bit in the Message Header specifies the Command Slot of the Request Message with which the Response Message is associated. In the in-band tunneling mechanism, the CSI bit in the Message Header is reserved.

The NVMe-MI Message Type (NMIMT) field contains the value from the same field in the corresponding Request Message.

Figure 28: Response Message Fields

Bytes	Description
3:0	**NVMe-MI Message Header (NMH):** Refer to section 3.1.
4	**Status (STATUS):** This field indicates the status associated with the Response Message. Response Message Status values are summarized in Figure 29.
N-1:5	**Response Body (RESPB):** This field contains response specific fields whose format is dependent on the NVMe-MI Message Type and Status field.

Figure 28: Response Message Fields

Bytes	Description
N+3:N	**Message Integrity Check (MIC):** Refer to section 3.1.

Response Message Status values are summarized in Figure 29. A Response Message Status of Success indicates that the corresponding Request Message completed successfully and that the Response Message is a Success Response. The format of the Response Body for a Success Response is dependent on the NVMe-MI Message Type (refer to Figure 20) and is described in the section defining each NVMe-MI Message Type.

A Response Message Status other than Success indicates that:

- an error occurred during servicing of the corresponding Request Message and that the Response Message is an Error Response; or
- more time is required for the processing of the corresponding Request Message and that the Response Message is a More Processing Required Response.

The format of the Response Body is dependent on the Response Message Status. Figure 29 references the section that defines the format of the Response Message for each Response Message Status value. If multiple error Response Message Status values apply, then the Responder selects one of those applicable Response Message Status values to report.

Figure 29: Response Message Status Values

Value	Description	Response Message Format Section
Status Values that do not indicate an error (i.e., Success Response).		
00h	**Success:** The command completed successfully.	4.1.2.1
01h	**More Processing Required:** The Command Message is in progress and requires more time to complete processing. When this Response Message Status is used in a Response Message, a subsequent Response Message contains the result of the Command Message. This Response Message Status shall not be sent more than once per Command Message, except for retransmission due to a Replay Control Primitive as described in section 4.2.1.5.	4.1.2.3
Status Values that indicate an error (i.e., Error Response).		
02h	**Internal Error:** The Request Message was not able to be processed due to a vendor specific internal error.	4.1.2.1
03h	**Invalid Command Opcode:** The associated command opcode field is not valid. Invalid opcodes include reserved and optional opcodes that are not implemented.	4.1.2.1
04h	**Invalid Parameter:** Invalid parameter field value. Request Messages received with reserved or unimplemented values in defined fields shall be completed with an Invalid Parameter Error Response. Other error conditions that result in Invalid Parameter Error Response are specified elsewhere in this specification.	4.1.2.2
05h	**Invalid Command Size:** The size of the Message Body of the Request Message was different than expected due to a reason other than the Command Message requiring Request Data and containing too much or too little Request Data (e.g., the Request Message did not contain all the required parameters). The expected size of the Message Body is determined by the NVMe-MI Message Type and opcode assuming no other errors are detected (e.g., Invalid Command Opcode or Invalid Parameter).	4.1.2.1
06h	**Invalid Command Input Data Size:** The Command Message requires Request Data and contains too much or too little Request Data.	4.1.2.1
07h	**Access Denied:** A Request Message was prohibited from being processed due to a vendor specific protection mechanism or the Command and Feature Lockdown feature (refer to the NVM Express Base Specification).	4.1.2.1

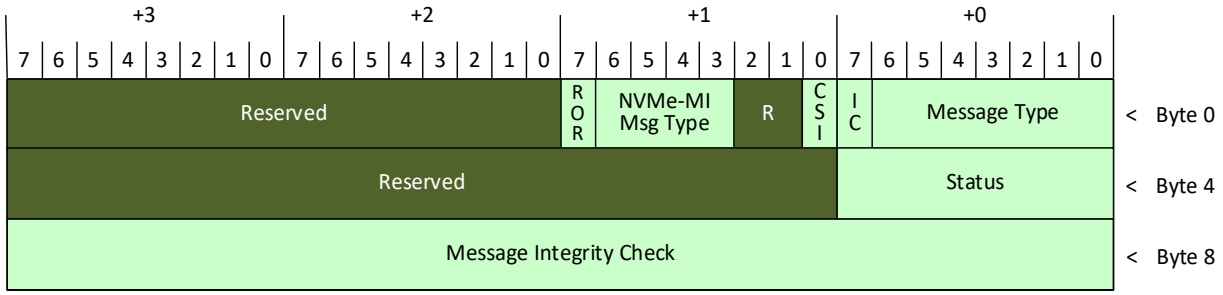
Figure 29: Response Message Status Values

Value	Description	Response Message Format Section
08h	**Unable to Abort:** The Abort Control Primitive is unable to abort a Command Message.	4.1.2.1
09h to 1Fh	Reserved	-
20h	**VPD Updates Exceeded:** More updates to the VPD are attempted than allowed.	4.1.2.1
21h	**PCIe Inaccessible:** The PCIe functionality is not available at this time.	4.1.2.1
22h	**Management Endpoint Buffer Cleared Due to Sanitize:** An attempt was made to read data as defined in section 4.2.3 in the Management Endpoint Buffer that was zeroed due to an NVM Subsystem sanitize operation.	4.1.2.1
23h	**Enclosure Services Failure:** The Enclosure Services Process has failed in an unknown manner.	4.1.2.1
24h	**Enclosure Services Transfer Failure:** Communication with the Enclosure Services Process has failed.	4.1.2.1
25h	**Enclosure Failure:** An unrecoverable enclosure failure has been detected by the Enclosure Services Process.	4.1.2.1
26h	**Enclosure Services Transfer Refused:** The NVM Subsystem or Enclosure Services Process indicated an error or an invalid format in communication.	4.1.2.1
27h	**Unsupported Enclosure Function:** An SES Send command has been attempted to a simple Subenclosure.	4.1.2.1
28h	**Enclosure Services Unavailable:** The NVM Subsystem or Enclosure Services Process has encountered an error but may become available again.	4.1.2.1
29h	**Enclosure Degraded:** A noncritical failure has been detected by the Enclosure Services Process.	4.1.2.1
2Ah	**Sanitize In Progress:** The requested command is prohibited while an NVM Subsystem sanitize operation is in progress. Refer to section 6.4.	4.1.2.1
2Bh to DFh	Reserved	-
Status Values that may or may not indicate an error.		
E0h to FFh	Vendor Specific	Vendor Specific

4.1.2.1 Generic Error Response

A Generic Error Response is generated for errors in which no additional information is provided beyond the Response Message Status. Bytes 7:5 are reserved. The format of a Generic Error Response is shown in Figure 30.

Figure 30: Generic Error Response



4.1.2.2 Invalid Parameter Error Response

An Invalid Parameter Error Response is generated for Error Responses where the Status field is set to Invalid Parameter. The format of an Invalid Parameter Error Response is shown in Figure 31 and the response specific fields are summarized in Figure 32.

Unless otherwise specified, if multiple invalid parameters errors exist in a Request Message, then the Management Endpoint selects the invalid parameter that is returned in the Invalid Parameter Error Response.

Figure 31: Invalid Parameter Error Response

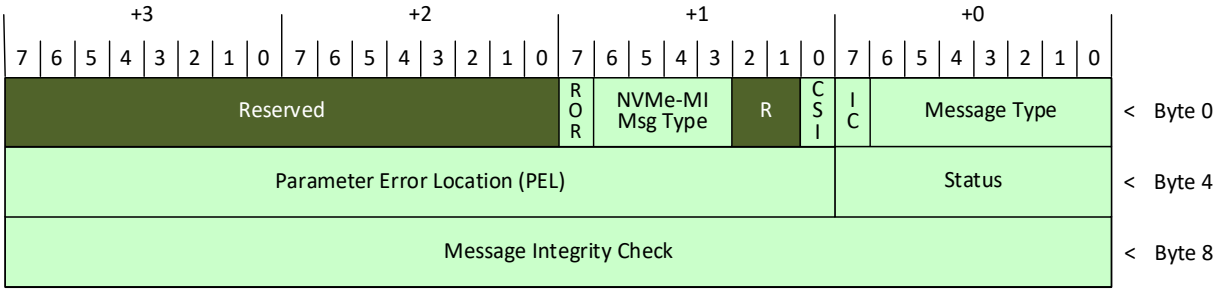


Figure 32: Invalid Parameter Error Response Fields

Bytes	Description
7:5	**Parameter Error Location (PEL):** This field indicates the request parameter within the Request Message that contains an invalid parameter. **Bits** / **Description** 23:08 — **Byte Location (BYTLOC):** Least-significant byte of the Request Message of the parameter that contained the error. If the error is beyond byte 65,535, then the value 65,535 is reported in this field. 07:03 — Reserved 02:00 — **Bit Location (BITLOC):** Least-significant bit in the least-significant byte of the Request Message of the parameter that contained the error. Valid values are 0 to 7.

4.1.2.3 More Processing Required Response

A More Processing Required Response shall be returned when the Management Endpoint requires more than the maximum Request-To-Response Time (refer to section 4.2.2.1) to complete the Process state of the Command Message unless otherwise specified (e.g., the More Processing Required Response may be discarded under certain conditions as described in section 4.2 or the Request Message may be discarded under certain conditions as described in section 8.1). If a More Processing Required Response is returned, then the Management Endpoint shall start to transmit the More Processing Required Response before the maximum Request-To-Response Time is exceeded unless otherwise specified (refer to section 4.2.2.1.1). If a Get State Control Primitive is processed while the Management Endpoint is transmitting the More Processing Required Response, then the Management Endpoint indicates a value of 2h (i.e., Process) in the Slot Command Servicing State field in the Get State Control Primitive Response Message (refer to Figure 45).

After sending a More Processing Required Response, the Command Slot shall return to the Process state to finish servicing the Command Message. The Response Message that is transmitted after processing completes is permitted to exceed the maximum Request-To-Response Time by the amount specified in the MPRT field (refer to Figure 34).

A More Processing Required Response shall only be transmitted once for a Command Message unless a Replay Control Primitive replays the More Processing Required Response. A Management Endpoint shall not transmit a More Processing Required Response for Control Primitive Response Messages.

The format of a More Processing Required Response is shown in Figure 33 and the response specific fields are summarized in Figure 34.

The following are examples of situations where a More Processing Required Response shall be returned, unless otherwise specified (e.g., the Pause Flag is set to '1' or a Firmware Commit command that results in the Firmware Activation Requires Maximum Time Violation status code):

- a Command Message is not able to be processed within the maximum Request-To-Response Time due to waiting on conditions such as the following:
  - NVM Subsystem initialization to complete (e.g., firmware initialization or hardware self-test following a reset, power on, or firmware activation without reset);
  - a resource that is not yet ready (e.g., media initialization required after power on, exiting low power mode, or reset); or
  - serialized internal queues to become free (e.g., the Command Message is not able to be processed until another Command Message completes servicing);
- the processing time of a Command Message is expected to exceed the maximum Request-To-Response Time (e.g., the Format NVM); and
- any reason other than a failure in the NVM Subsystem that is expected to cause the maximum Request-To-Response Time to be exceeded.

The Management Endpoint shall complete any steps required to be able to process the Command Message and then process the Command Message. For example, if an NVMe Admin Command targeting a Controller that is in normal operation (i.e., the value of the CSTS.SHST field is set to 00b) requires media access and media has not been initialized, then the Management Endpoint shall initialize media and then the NVMe Admin Command shall be processed.

If a Command Message is able to be processed successfully given sufficient time but the Management Endpoint instead returns an Error Response for any Command Message or returns an error status code in the Status field in CQEDW3 in an NVMe Admin Command Response, then such a response is possible to cause the Management Controller to erroneously flag the NVM Subsystem as failed.

Figure 33: More Processing Required Response

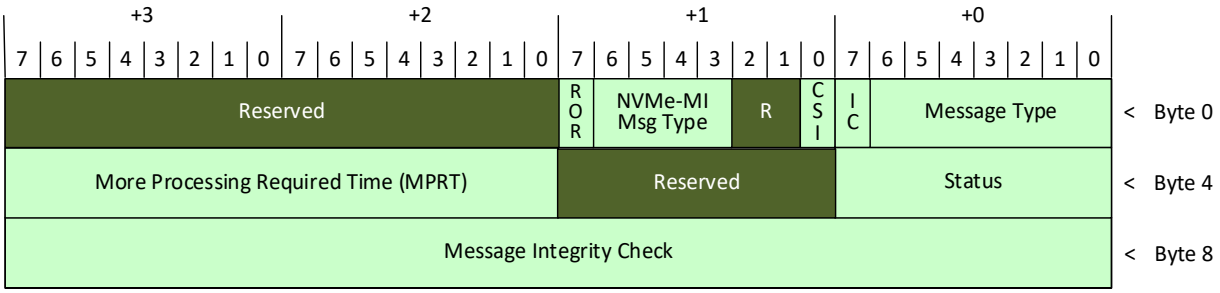


Figure 34: More Processing Required Response Fields

Bytes	Description
7:6	**More Processing Required Time (MPRT):** This field indicates the worst-case time in 100 ms units from the first attempt to start transmitting this More Processing Required Response until the first attempt to start transmitting the Response Message for the processed Command Message. A value of FFFFh in this field indicates that greater than or equal to 6,553.5 s more processing time is required.

4.1.3 Asynchronous Event Messages (AEMs) (Optional)

An Asynchronous Event Message (AEM) is an NVMe-MI Message that is transmitted by a Management Endpoint to a Management Controller after one or more Asynchronous Events (AEs) such as a health status change event, a temperature change event, or a firmware activation (refer to Figure 63) occurs. AEMs are posted (i.e., there is no NVMe-MI Message transmitted back from the Management Controller to the Management Endpoint in response to the AEM).

AEMs are permitted using the out-of-band mechanism. In-band communication uses the Asynchronous Event Request command (refer to the NVM Express Base Specification) for asynchronous events and therefore, AEMs are prohibited using the in-band tunneling mechanism. AEMs are optional for both NVMe Storage Devices and NVMe Enclosures.

AEMs are supported by Management Endpoints on a per port basis, and an implementation is permitted to support AEMs on Management Endpoints on a subset of the ports in the NVM Subsystem. Note that many host platforms are designed to connect a Management Controller to the 2-Wire Management Endpoint via a 2-Wire Mux. A Management Endpoint is not able to transmit an AEM while the 2-Wire Mux downstream channel connected to a Management Endpoint is not connected to the 2-Wire Mux upstream channel which, in turn, is connected to the Management Controller. Therefore, for example, an NVM Subsystem may choose to support AEMs on Management Endpoints on the PCIe port(s) but not on the 2-Wire port. However, host platforms that implement an I3C hub architecture resolve the problems caused by 2-Wire Mux.

If AEMs are supported on a given Management Endpoint (i.e., the Asynchronous Event Messages Supported bit is set to '1' in the Port Information data structure for the port associated with the Management Endpoint), then at least one AE shall be supported. The list of supported AEs is returned in the Response Message to a Configuration Get command for the Asynchronous Event configuration (refer to section 5.1.4).

Each AE is able to be enabled or disabled on a per Management Endpoint basis via the Configuration Set command for the Asynchronous Event configuration.

The AEM servicing model is defined in section 4.4.

4.2 Out-of-Band Request Message Servicing Model

A Management Controller sends a Request Message to a Management Endpoint, the Management Endpoint processes the Request Message, and when processing has completed, sends a Response Message back the Management Controller. Under no circumstances does a Management Endpoint generate an unsolicited Response Message (i.e., a Response Message that does not correspond to a previously received Request Message).

This specification utilizes Command Slots for Command Message servicing. Each Management Endpoint contains two Command Slots that each include state information that is unique to each Command Slot and a Pause flag that is global to the Management Endpoint. The Command Slot state information and the value of the Pause Flag is returned by the Get State Primitive (refer to section 4.2.1.4).

A Management Controller sends a Command Message to a Management Endpoint that targets a specific Command Slot in the Management Endpoint. The Management Endpoint assembles MCTP packets into Command Messages separately for each Command Slot. Each Command Slot remains allocated to the Command Message until servicing of the Command Message has completed.

If a Command Slot that is not in the Idle state receives the start of a new Command Message, then:

- the Management Endpoint shall set the CMNICS bit to '1' (refer to Figure 43); and
- perform an implicit Abort Control Primitive (refer to section 4.2.1.3) with the exception that the Management Endpoint shall not transmit the Abort Control Primitive Response Message. If command servicing is:
    - unable to be aborted, then the Management Endpoint shall:

- silently discard the new Command Message; and
- continue servicing the Command Message being serviced at the time the new Command Message was received;

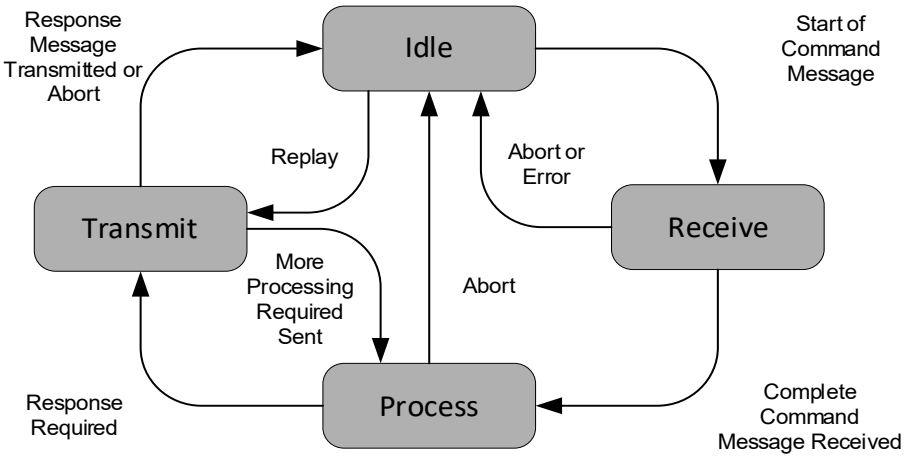
or

- aborted, then the Management Endpoint shall transition to the Receive state to service the new Command Message.

A Command Message is the only type of multi-packet NVMe-MI Message that may be received by a Management Endpoint. The maximum number of Command Messages in flight to a Management Endpoint is equal to the number of Command Slots. The operation of each Command Slot is independent, allowing a Management Controller to have two independent streams of Command Messages to a Management Endpoint. The Command Message associated with each Command Slot is serviced in parallel. If the NVM Subsystem implements multiple Management Endpoints, then command servicing of each Management Endpoint occurs in parallel. An NVM Subsystem that implements *N* Management Endpoints may have up to 2*N* Command Messages serviced in parallel using the out-of-band mechanism.

The Command Servicing State Diagram in Figure 35 is used to describe functional requirements and does not mandate an implementation.

Figure 35: Command Servicing State Diagram



Idle: A Command Slot is idle when it is not in the Receive, Process, or Transmit state. This is the initial state of the command servicing state machine following a Management Endpoint Reset (refer to section 8.3.3). Command servicing shall transition from the Idle state to the Receive state when the first MCTP packet of a Command Message is received.

If a Replay Control Primitive is received and there is a Response Message available for retransmission, then command servicing shall transition to Transmit state (refer to section 4.2.1.5).

Receive: The state when a Command Message is being received, assembled or validated. Command servicing shall transition from Receive to the Idle state when:

- an Abort Control Primitive is successful;
- an error is detected in message assembly (refer to section 3.2.1.1); or
- the Message Integrity Check fails (refer to section 3.1.1.1).

Command servicing shall transition from Receive state to the Process state when a Command Message is assembled and the message integrity check is successful.

Process: The state when a Command Message is processed. Processing of a Command Message consists of checking for errors with the Command Message and performing the actions specified

by the Command Message or aborting the Command Message. Command servicing shall transition from Process to the Transmit state when:

a) the processing of the Command Message has completed, regardless of whether the Management Endpoint is paused; or
b) a More Processing Required Response Message is required to be transmitted (refer to section 4.1.2.3) and the Management Endpoint is not paused. If the Management Endpoint is paused, then the Management Endpoint shall not transition to the Transmit state to transmit a More Processing Required Response Message.

Command servicing shall transition from the Process state to the Idle state due to an Abort Control Primitive unless the Command Message was not able to be aborted (refer to section 4.2.1.3).

Upon completion of Command Message processing, if a More Processing Required Response Message for the Command Slot is pending transmission (e.g., the More Processing Required Response Message was not able to be transmitted because the Pause Flag was set to '1' or the physical transport to the Management Controller external to the NVMe Storage Device or NVMe Enclosure is unavailable), then the More Processing Required Response Message should be discarded.

Transmit: The state in which a Response Message for the Command Message is transmitted to the Management Controller. Command servicing shall transition from the Transmit to the Idle state once the entire Response Message associated with the Command Message has been transmitted or due to an Abort Control Primitive (refer to section 4.2.1.3). Command servicing shall transition from the Transmit to the Process state after transmitting a More Processing Required Response Message.

4.2.1 Control Primitives

Control Primitives are Request Messages sent from a Management Controller to a Management Endpoint to affect the servicing of a previously issued Command Message or get the state of a Command Slot and Management Endpoint. Control Primitives are applicable only in the out-of-band mechanism and are prohibited in the in-band tunneling mechanism.

Control Primitives may target a Command Slot. Unlike Command Messages, Control Primitives may be sent while the Command Slot is in any command servicing state and are processed immediately by the Management Endpoint. Unless otherwise indicated, Control Primitives do not change the command servicing state of the Command Slot.

The format of a Control Primitive is shown in Figure 36 and the fields are described in Figure 37.

Figure 36: Control Primitive Request Message Format

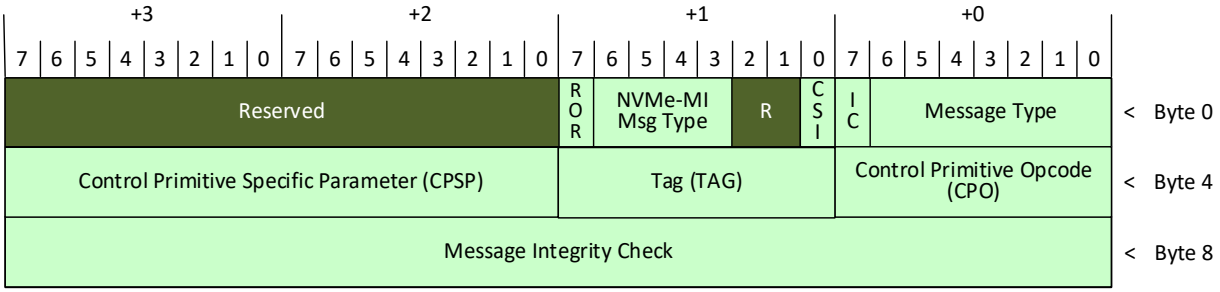


Figure 37: Control Primitive Fields

Bytes	Description
03:00	**NVMe-MI Message Header (NMH):** Refer to section 3.1.

Figure 37: Control Primitive Fields

Bytes	Description
04	**Control Primitive Opcode (CPO):** This field specifies the opcode of the Control Primitive to be processed. Refer to Figure 38.
05	**Tag (TAG):** This field specifies a tracking identifier that is returned in the TAG field of the associated Control Primitive Response Message.
07:06	**Control Primitive Specific Parameter (CPSP):** This field is used to pass Control Primitive specific parameter information.
11:08	**Message Integrity Check (MIC):** Refer to section 3.1.

Figure 38: Control Primitive Opcodes

Opcode	O/M[1]	Command
00h	M	Pause
01h	M	Resume
02h	M	Abort
03h	M	Get State
04h	M	Replay
05h to EFh		Reserved
F0h to FFh	O	Vendor Specific
Notes: 1. O/M: O = Optional, M = Mandatory.		

The format of a Success Response associated with a Control Primitive is shown in Figure 39 and the fields are described in Figure 40.

Figure 39: Control Primitive Success Response Format

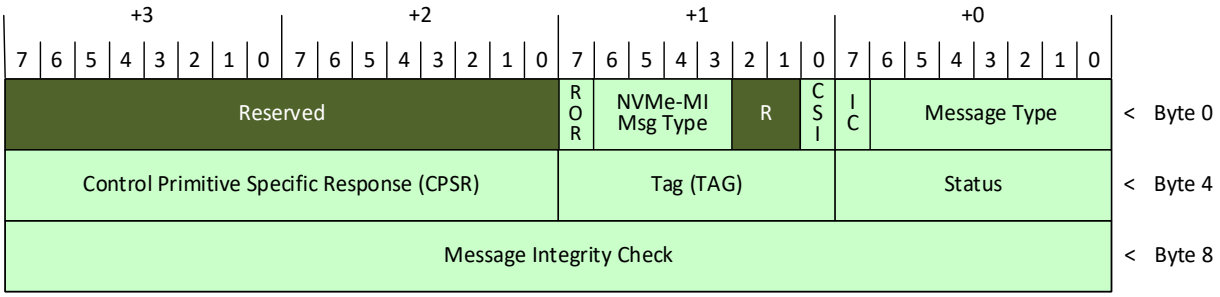


Figure 40: Control Primitive Success Response Fields

Bytes	Description
03:00	**NVMe-MI Message Header (NMH):** Refer to section 3.1.
04	**Status (STATUS):** Refer to section 4.1.2.
05	**Tag (TAG):** This field shall contain a copy of the Tag specified in the TAG field of the associated Request Message.
07:06	**Control Primitive Specific Response (CPSR):** This field is used to return Control Primitive specific status.
11:08	**Message Integrity Check (MIC):** Refer to section 3.1.

Unlike Command Messages, a Management Controller may issue a Control Primitive to a Command Slot without waiting for a response for previously issued Control Primitives to that Command Slot. If multiple Control Primitives are issued without waiting for responses from the Management Endpoint, only the actions and response associated with the last Control Primitive are guaranteed (i.e., the actions associated with

previously issued but unacknowledged Control Primitives may or may not be performed and the Response Messages for previously issued but unacknowledged Control Primitives may or may not be transmitted). Receipt of a Control Primitive never corrupts a previous Control Primitive associated with the Command Slot. The Response Message is either entirely transmitted or discarded.

The value of the Tag field in the Control Primitive Request Message is a tracking identifier specified by the Management Controller. The Management Endpoint shall copy the value of the Tag field from the Control Primitive Request Message into the Tag field of the Control Primitive Response Message.

4.2.1.1 Pause

The Pause Control Primitive shall set the Pause Flag to '1' and then suspend (i.e., pause) transmission of Response Messages for Command Messages or AEMs on a packet boundary. Response Messages for Control Primitives shall be transmitted even if the Paused Flag is set to '1' (i.e., the Pause Flag has no effect on Control Primitive Response Messages).

It is not an error to process a Pause Control Primitive while the Pause Flag is set to '1'.

The CSI bit in a Pause Control Primitive is not used and should be cleared to 0h. If the CSI bit is set to '1', then the Management Endpoint shall transmit an Invalid Parameter Error Response with the PEL field indicating the CSI bit.

The CPSP field for the Pause Control Primitive is reserved.

The format of the CPSR field in the Control Primitive Success Response is shown in Figure 41.

Figure 41: Pause Control Primitive Success Response Fields

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>07:06</td><td>Control Primitive Specific Response (CPSR): This field is used to return Control Primitive specific status.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>15:02</td><td>Reserved</td></tr>
<tr><td>01</td><td>Obsolete. Refer to NVM Express Management Interface Specification, Revision 1.2. This bit shall always be set to '1' for backwards compatibility.</td></tr>
<tr><td>00</td><td>Obsolete. Refer to NVM Express Management Interface Specification, Revision 1.2. This bit shall always be set to '1' for backwards compatibility.</td></tr>
</table></td></tr>
</table>

4.2.1.2 Resume

The Resume Control Primitive is the complement to the Pause Control Primitive. The Resume Control Primitive shall clear the Pause Flag to '0'. After transmitting the Response Message for the Resume Control Primitive, the Management Endpoint shall resume paused transmissions.

It is not an error to process a Resume Control Primitive while the Pause Flag is cleared to '0'.

The CSI bit in a Resume Control Primitive is not used and should be cleared to '0'. If the CSI bit is set to '1', then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the CSI bit.

The Resume Control Primitive shall cause a Management Endpoint to transmit the packet after the last packet the Management Endpoint finished transmitting prior to being paused. If the last Response Message packet that was transmitted was not received by the Management Controller, then the Management Controller should detect an out-of-sequence packet sequence number in the resumed Response Message and drop that Response Message. To avoid this synchronization issue, the Management Controller should issue a Replay Control Primitive specifying the packet number in the Response Replay Offset field from which the Response Message is replayed.

The CPSP field for the Resume Control Primitive is reserved. The CPSR field in the Control Primitive Success Response is reserved.

4.2.1.3 Abort

The Abort Control Primitive attempts to stop Command Message servicing and return a Command Slot to the Idle state. It may not be possible for the Abort Control Primitive to abort some Command Messages (e.g., Shutdown) during the Process state.

If a Success Response is transmitted in response to the Abort Control Primitive, then any Response Message for the Command Slot that is pending transmission, including any Response Message that is able to be replayed by a Replay Control Primitive while the Command Slot is in the Idle state, shall be discarded and is unavailable for replay.

Attempting to abort a Command Message shall clear the Pause Flag to '0' even if the Command Message is unable to be aborted. If the other Command Slot of the Management Endpoint has a paused transmission, then the paused transmission shall resume after the Abort Control Primitive Response Message has been transmitted.

The CPSP field for the Abort Control Primitive is reserved. The format of the CPSR field in the Control Primitive Success Response is shown in Figure 42.

Figure 42: Abort Control Primitive Success Response Fields

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>07:06</td><td>Control Primitive Specific Response (CPSR): This field is used to return Control Primitive specific status.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>15:02</td><td>Reserved</td></tr>
<tr><td>01:00</td><td>Command Processing Abort Status (CPAS): This field indicates the effect of the Abort Control Primitive on the processing of the Command Message associated with the Command Slot.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>Command aborted after processing completed or no command to abort.</td></tr>
<tr><td>01b</td><td>Command aborted before processing began.</td></tr>
<tr><td>10b</td><td>Command aborted after processing partially completed.</td></tr>
<tr><td>11b</td><td>Reserved</td></tr>
</table></td></tr>
</table></td></tr>
</table>

The resulting Command Slot state and CPAS value of the Abort Control Primitive Success Response is based on the command servicing state of the specified Command Slot when the Abort Control Primitive is received, as described below:

Idle: The Management Endpoint shall respond with the CPAS field cleared to 00b.

Receive: The Command Slot shall transition to the Idle state and the Management Endpoint shall respond with the CPAS field set to 01b.

Process: Results depend on the type of Command Message that is being processed and how much processing has already completed:

a) if the Management Endpoint is able to abort the Command Message before command processing affected the NVM Subsystem, then the Command Slot shall transition to the Idle state and the Management Endpoint shall respond with the CPAS field set to 01b;
b) if the Management Endpoint is able to abort the Command Message after command processing affected the NVM Subsystem, then the Command Slot shall transition to the Idle state and the Management Endpoint shall respond with the CPAS field set to 10b; or
c) if the Management Endpoint is unable to abort the Command Message (e.g., past the point-of-no-return in processing a Shutdown command), then the Management Endpoint shall respond with a Response Message Status of Unable To Abort, complete processing the Command Message, and then transition to the Transmit state to transmit the Response Message.

Transmit: Transmissions shall stop on a packet boundary as soon as possible.

If the Command Message has finished processing, then the Command Slot shall transition to the Idle state and the Management Endpoint shall respond with the CPAS field cleared to 00b.

If the Command Message has not finished processing because it is transmitting a More Processing Required Response, then the Command Slot shall transition to the Process state after transmitting the More Processing Required Response and shall behave as if an Abort Control Primitive was received in the Process state.

4.2.1.4 Get State

The Get State Control Primitive is used to get and clear the state of a Management Endpoint and the Management Endpoint's Command Slots.

The Management Endpoint shall contain Management Endpoint state as shown in Figure 43.

Figure 43: Management Endpoint State Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>1:0</td><td>Management Endpoint State (MES): This field shall indicate the Management Endpoint state. A Management Endpoint Reset of the corresponding Management Endpoint shall clear the Pause Flag bit to '0' and clear bits 13:0 to 0h.
<table>
<tr><th>Bits</th><th>Command Slot Specific[1]</th><th>Description</th></tr>
<tr><td>15</td><td>No</td><td>Pause Flag (PFLG): If the Management Endpoint is paused, then this bit shall be set to '1'. If the Management Endpoint is not paused, then this bit shall be cleared to '0'.

The request-to-response timer is reset and restarted under certain conditions when the Pause Flag transitions from '1' to '0' (refer to section 4.2.2.1).</td></tr>
<tr><td>14</td><td>No</td><td>NVM Subsystem Reset Occurred (NSSRO): If an NVM Subsystem Reset occurs due to any reason other than application of main power and does not cause activation of a new firmware image, then this bit shall be set to '1'.

If an NVM Subsystem Reset occurs due to application of main power or causes activation of a new firmware image, then this bit shall be cleared to '0'.</td></tr>
<tr><td>13</td><td>No</td><td>Bad Packet or Other Physical Layer (BPOPL): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to '1'.</td></tr>
<tr><td>12</td><td>No</td><td>Bad, Unexpected, or Expired Message Tag (BUEMT): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to '1'.</td></tr>
<tr><td>11</td><td>No</td><td>Out-of-Sequence Packet Sequence Number (OSPSN): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to '1'.</td></tr>
<tr><td>10</td><td>No</td><td>Unexpected Middle or End of Packet (UMEP): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to '1'.</td></tr>
<tr><td>09</td><td>No</td><td>Incorrect Transmission Unit (ITU): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to '1'.</td></tr>
<tr><td>08</td><td>No</td><td>Unknown Destination ID (UDSTID): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to '1'.</td></tr>
<tr><td>07</td><td>No</td><td>Bad Header Version (BHVS): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to '1'.</td></tr>
</table></td></tr>
</table>

Figure 43: Management Endpoint State Data Structure

<table>
<tr><th>Bytes</th><th colspan="3">Description</th></tr>
<tr><td></td><td>06</td><td>No</td><td>Unsupported Transmission Unit (UTUNT): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to ‘1’.</td></tr>
<tr><td></td><td>05</td><td>No</td><td>Obsolete. Refer to the NVM Express Management Interface Specification, Revision 1.2.</td></tr>
<tr><td></td><td>04</td><td>No</td><td>Bad Message Integrity Check Error (BMICE): If the Management Endpoint detects an error of this type (refer to the MCTP Base Specification), then this bit shall be set to ‘1’.</td></tr>
<tr><td></td><td>03</td><td>No</td><td>Command Message to non-Idle Command Slot (CMNICS): If the Management Endpoint implicitly aborted one or more Command Messages due to receiving the first packet of a new Command Message to a Command Slot that was not in the Idle state, then this bit shall be set to ‘1’.</td></tr>
<tr><td></td><td>02</td><td></td><td>Reserved</td></tr>
<tr><td></td><td>01:00</td><td>Yes</td><td>Slot Command Servicing State (SSTA): This field shall indicate the current command servicing state of the Command Slot. An implementation may choose to indicate only the Idle and Process states in this field. Refer to Figure 36.

The Management Endpoint shall indicate a value of 2h (i.e., Process) while transmitting a More Processing Required Response.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Idle</td></tr>
<tr><td>1h</td><td>Receive</td></tr>
<tr><td>2h</td><td>Process</td></tr>
<tr><td>3h</td><td>Transmit</td></tr>
</table></td></tr>
<tr><td></td><td colspan="3">Notes:
1. Command Slot Specific. A ‘Yes’ in this column indicates the value of the field is independent per Command Slot within a Management Endpoint. A ‘No’ in this column indicates the same value is reported for either Command Slot.</td></tr>
</table>

The format of the CPSP field in the Control Primitive Request Message is shown in Figure 44.

Figure 44: Get State Control Primitive Request Message Fields

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>07:06</td><td>Control Primitive Specific Parameter (CPSP): This field specifies Control Primitive specific parameter information.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>15:01</td><td>Reserved</td></tr>
<tr><td>00</td><td>Clear Error State Flags (CESF): If this bit is set to ‘1’, then the Management Endpoint shall perform the following steps atomically in the order listed:
1. copy the current value of the Management Endpoint State data structure (refer to Figure 43) to the Control Primitive Specific Response field of the Response Message (refer to Figure 45); and
2. clear bits 14:03 in the Management Endpoint State data structure to 0h.

If this bit is cleared to ‘0’, then the Management Endpoint shall copy the current value of the Management Endpoint State data structure (refer to Figure 43) to the Control Primitive Specific Response field of the Response Message (refer to Figure 45) and shall not modify bits 14:03 in the Management Endpoint State data structure.</td></tr>
</table></td></tr>
</table>

The format of the CPSR field in the Control Primitive Success Response is shown in Figure 45.

Figure 45: Get State Control Primitive Success Response Fields

Bytes	Description
07:06	**Control Primitive Specific Response (CPSR):** This field shall indicate the contents of the Management Endpoint State data structure (refer to Figure 43).

4.2.1.5 Replay

The Replay Control Primitive shall cause the Management Endpoint to transmit a Response Message Status of Success and then shall clear the Pause Flag to '0' before retransmitting the Response Message for the last Command Message processed in the specified Command Slot. Packets within a given Response Message are transmitted in order as required by the MCTP Base Specification, but there are no packet ordering requirements between different Response Messages and so replayed packets from the specified Command Slot may be interleaved with packets from the other Command Slot. Control Primitive Response Messages and AEMs shall not be replayed by the Management Endpoint.

The replayed Response Message forms a new MCTP Response Message with Message Data starting from Response Replay Offset of the original Response Message and continuing to the end of the Response Message, including the original MIC. The first packet shall have SOM set to '1' and shall include the Message Header of the original Response Message even if the Response Replay Offset is not 0h. The Msg tag in each packet of the replayed Response Message shall be set to the value of the Msg tag in the associated Replay Control Primitive. Refer to the MCTP Base Specification for the definition of the Msg tag.

Note that the Management Controller requires extensions to the MCTP Base Specification in its MCTP layer in order to replay a Response Message using a non-zero Response Replay Offset. No extensions to the MCTP Base Specification are required to replay with Response Replay Offset equal to 0h. For the case where a Management Controller chooses to use a non-zero Response Replay Offset, the MCTP Base Specification requires terminating message assembly for certain errors (i.e., receiving a packet with bad packet data integrity).

If a Management Controller receives a number of packets with no errors in a Response Message and then gets an error on a packet that causes termination of message assembly, the Management Controller may extend its MCTP layer to forward the packets it received with no errors to its NVMe-MI layer prior to terminating message assembly. The Management Controller may then issue a Replay Control Primitive to get the second part of the Response Message using a non-zero Response Replay Offset. The Management Controller's NVMe-MI layer then assembles the two partial Response Messages to create the whole Response Message. The MIC may then be validated across the whole Response Message as described in section 3.1.1.1.

The format of the CPSP field in the Control Primitive Request Message is shown in Figure 46.

Figure 46: Replay Control Primitive Request Fields

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>07:06</td><td>Control Primitive Specific Parameter (CPSP): This field is used to pass Control Primitive specific parameter information.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>15:08</td><td>Reserved</td></tr>
<tr><td>07:00</td><td>Response Replay Offset (RRO): This field specifies the starting packet number from which the Response Message associated with the last Command Message processed in the Command Slot shall be replayed.

This is a 0’s based value. When this field is cleared to 0h, the first packet of the associated Response Message is the first packet replayed.

If this field specifies an offset that is beyond the length of the Response Message, then processing of the Control Primitive is aborted and the Management Endpoint shall transmit an Invalid Parameter Error Response with the PEL field indicating this field.</td></tr>
</table></td></tr>
</table>

The format of the CPSR field in the Control Primitive Success Response is shown in Figure 47.

Figure 47: Replay Control Primitive Success Response Fields

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>07:06</td><td>Control Primitive Specific Response (CPSR): This field is used to return Control Primitive specific status.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>15:01</td><td>Reserved</td></tr>
<tr><td>00</td><td>Response Replay (RR): This bit indicates if a previous Response Message is to be retransmitted. This bit is set to ‘1’ if the requested Response Message is to be retransmitted by the Management Endpoint. This bit is cleared to ‘0’ if the requested Response Message is not retransmitted (i.e., there was no Response Message to retransmit).</td></tr>
</table></td></tr>
</table>

The result of a Replay Control Primitive is based on the command servicing state of the specified Command Slot when the Replay Control Primitive is received, as described below:

Idle: The result depends on the last Command Message processed in the specified Command Slot as follows:

a) if there is no Response Message available to retransmit (i.e., a Replay Control Primitive is received after the Response Message has been discarded by an Abort Control Primitive or a Management Endpoint Reset has occurred but before any Command Messages are processed), then the Management Endpoint shall transmit a Response Message with success status with the RR bit cleared to ‘0’; or

b) if there is a Response Message available to retransmit (i.e., a Replay Control Primitive is received following the processing of one or more Command Messages but before the Response Message has been discarded by an Abort Control Primitive or a Management Endpoint Reset has occurred), then the Management Endpoint shall complete the following steps in order:

1. transmit a Replay Control Primitive Response Message with the RR bit set to ‘1’;
2. transition the Command Slot to the Transmit state; and
3. transmit a new Response Message containing the payload starting at the packet offset specified in the Response Replay Offset field of the Replay Control Primitive.

Receive: The Management Endpoint shall transmit a Replay Control Primitive Success Response with the RR bit cleared to ‘0’.

Process: If a More Processing Required Response has not been transmitted for the Command Message being processed, then a Replay Control Primitive Success Response shall be transmitted with the RR bit cleared to '0'.

If a More Processing Required Response has been transmitted, then a Replay Control Primitive Success Response shall be transmitted with the RR bit set to '1' and then the More Processing Required Response shall be retransmitted. The Management Endpoint shall update the More Processing Required Time field in the Response Message with the current worst-case amount of additional time that the Management Controller should wait for the Management Endpoint to complete processing of the Command Message.

Transmit: The Management Endpoint shall complete the following steps in order:

1. stop transmitting Response Message packets for the Command Slot;
2. transmit a Replay Primitive Success Response with the RR bit set to '1'; and
3. transmit a new Response Message containing the payload starting at the packet offset specified in the Response Replay Offset field of the Replay Control Primitive.

4.2.2 Out-of-Band Error Handling

This section describes timing requirements and a packet retry mechanism for error handling specific to the NVMe-MI out-of-band message processing model.

4.2.2.1 Response Message Timeouts

The timing parameters defined for NVMe-MI Messages are similar to the timing parameters for MCTP Control Messages in the MCTP Base Specification: Request-To-Response Time (refer to section 1.8.41), Interpacket Time (refer to section 1.8.24), and Transmission Delay (refer to section 1.8.45).

The maximum Request-To-Response Time shall be 100 ms unless otherwise specified. A request-to-response timeout occurs when a Management Controller does not receive a Response Message within the Request-To-Response Time plus two times the Transmission Delay. If the Pause Flag transitions from '1' to '0' (e.g., due to a Resume Control Primitive or a Replay Control Primitive), then the request-to-response timer shall be reset and restarted for any Command Slot where:

a) processing of the Command Message has not completed; and
b) the Management Endpoint has not started transmitting a More Processing Required Response for the Command Message being processed.

The maximum Interpacket Time shall be 40 ms while a Command Slot is in the Transmit state and the Pause Flag is cleared to '0' unless otherwise specified. An interpacket timeout occurs when a Requester does not receive a subsequent packet within the Interpacket Time plus the Transmission Delay after receiving the prior packet.

Exceptions to the timeouts in this section are specified in section 4.2.2.1.1.

4.2.2.1.1 Response Message Timeouts Exceptions

Response Message timeouts only apply while the physical transport to the Management Controller external to the NVMe Storage Device or NVMe Enclosure is available and the Management Endpoint is not paused. If the external physical transport is not available (e.g., busy or disconnected) or the Management Endpoint is paused, then all transmissions are delayed at least until the transport becomes available again and the Management Endpoint is not paused.

The Response Message timeouts defined by this specification assume that there is only one outstanding MCTP message to be transmitted from a Management Endpoint at a time since a Response Message may be delayed until other MCTP messages complete transmitting. The Management Endpoint shall not take longer than the maximum Interpacket Time to start transmitting a Response Message delayed by another Response Message, unless otherwise specified (e.g., the physical transport to the Management Controller external to the NVMe Storage Device or NVMe Enclosure is not available, or the Management Endpoint is

paused). The Management Controller should adjust its timeouts to account for this added delay whenever multiple MCTP messages from a Management Endpoint are expected.

4.2.2.2 2-Wire Management Endpoint NACK and Packet Retry

A Management Endpoint that experiences a NACK anywhere during transmission of a packet shall attempt to retransmit that packet. The Management Endpoint shall treat a NACK during the ninth transmission attempt (the original transmission attempt plus 8 retry attempts) of a packet as if an implicit Pause Control Primitive was received with the exception that the Management Endpoint shall not transmit a Pause Control Primitive Response Message.

If the packet that was NACKed nine times was a Control Primitive Response Message, then that packet shall be discarded.

4.2.3 Management Endpoint Buffer

Since the maximum size of an NVMe-MI Message is 4,224 bytes, the maximum amount of out-of-band Request Data that is able to be contained in a Request Message is 4,216 bytes (i.e., 4,224 bytes minus 4-byte Message Header and 4-byte Message Integrity Check field) and the maximum amount of out-of-band Response Data that is able to be contained in a Response Message is 4,215 bytes (i.e., 4,224 bytes minus 4-byte Message Header, 1-byte Status field, and 4-byte Message Integrity Check field). The amount of supported Request Data or Response Data is Command Message specific due to the presence of additional command specific fields. In some cases, a Management Endpoint requires the servicing of Command Messages that contain more Request Data or Response Data than is able to be transferred in an NVMe-MI Message. For example, a Management Endpoint requires the issuing of a Get Log Page command in the NVM Express Admin Command Set to transfer a log page that is greater in size than that allowed in the Response Data.

A Management Endpoint may support an optional Management Endpoint Buffer that facilitates Request Data and Response Data transfers that exceed that maximum size allowed by an NVMe-MI Message. A Management Endpoint Buffer is exclusive to one Management Endpoint and shall not be shared. Support for the Management Endpoint Buffer and its size in bytes is indicated by the Management Endpoint Buffer Size field in the Port Information data structure of the port with which the Management Endpoint is associated. Management Endpoints are not required to all have the same Management Endpoint Buffer support. For example, a subset of Management Endpoints may support a Management Endpoint Buffer and the size of each of these Management Endpoint Buffers may be different.

If a Management Endpoint supports a Management Endpoint Buffer, then all Command Messages or a subset of Command Messages supported by the Management Endpoint may support use of the Management Endpoint Buffer. A list of commands that support the use of the Management Endpoint Buffer is contained in the Management Endpoint Buffer Command Support List data structure that is retrieved using the Read NVMe-MI Data Structure command. If a Management Endpoint supports a Management Endpoint Buffer, then the Management Endpoint shall support the Management Endpoint Buffer Read and Management Endpoint Buffer Write commands.

The contents of a Management Endpoint Buffer is able to be read or written by a Management Controller by issuing Management Endpoint Buffer Read commands and Management Endpoint Buffer Write commands. The Management Endpoint Buffer is permitted to be read or written in an arbitrary manner. For example, the contents of the Management Endpoint Buffer is able to be written sequentially using a sequence of Management Endpoint Buffer Write commands or the partial contents of the Management Endpoint Buffer is able to be written in any order with gaps using these commands. Furthermore, Management Endpoint Buffer Read commands and Management Endpoint Buffer Write commands are able to be interleaved allowing a portion of the Management Endpoint Buffer to be read while another portion of the Management Endpoint Buffer is written.

If the Management Endpoint Buffer (MEB) bit is set to ‘1’ in a Command Message that normally contains Request Data and supports the Management Endpoint Buffer operation (i.e., the Command Message is

listed in the Management Endpoint Buffer Supported Command List data structure), then unless otherwise specified:

a) Request Data shall not be transferred in the Command Message itself and the required Request Data shall be transferred from the Management Endpoint Buffer;
b) the Request Data shall start at a zero offset from the start of the Management Endpoint Buffer; and
c) if the Command Message contains Request Data or does not support Request Data, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Request Data field of the Command Message.

If the Management Endpoint Buffer (MEB) bit is set to '1' in a Command Message that normally results in Response Data and supports the Management Endpoint Buffer operation, then no Response Data is transferred in the corresponding Response Message itself and the Response Data is instead transferred to the Management Endpoint Buffer. The Response Data starts at a zero offset from the start of the Management Endpoint Buffer.

A Management Endpoint Reset (refer to section 8.3.3) shall clear each byte of the corresponding Management Endpoint Buffer to 0h. The contents of the Management Endpoint Buffer are modified by the Management Endpoint Buffer Write command and by Command Messages that generate Response Data with the MEB bit set to '1'. When the Management Endpoint Buffer is updated with Response Data, the contents of the Management Endpoint Buffer that are not updated are cleared to 0h (i.e., the Message Data from previous Command Messages is not preserved). The same contents of the Management Endpoint Buffer may be used as Request Data for multiple Command Messages. Similarly, the Management Endpoint Buffer allows the use of Response Data generated by one Command Message to be used as the Request Data for a subsequent Command Message.

Since it is possible to have two out-of-band Command Messages, one associated with each of the two Command Slots, being simultaneously serviced that use the Management Endpoint Buffer, the Management Controller must comprehend and manage any possible race conditions. Updates to the Management Endpoint Buffer are not guaranteed to be atomic. Therefore, when a race condition involving two operations that update the Management Endpoint Buffer occurs, the final contents of the Management Endpoint Buffer may be an arbitrary mixture of the updates.

4.2.3.1 Interactions with Sanitize

The Management Endpoint Buffer is considered a cache in the context of NVM Subsystem sanitize operations (refer to the NVM Express Base Specification) performed in an NVM Subsystem. The Management Endpoint Buffer may contain Response Data associated with a previously processed command that is not allowed during an NVM Subsystem sanitize operation.

If an NVM Subsystem sanitize operation is initiated, then the contents of all Management Endpoint Buffers in the NVM Subsystem shall be cleared to 0h.

If a Management Endpoint processes any Command Message that accesses the Management Endpoint Buffer after that Management Endpoint Buffer has been cleared to 0h due to a NVM Subsystem sanitize operation but before any subsequent writes to the Management Endpoint Buffer from the Management Controller or NVM Subsystem, then the Management Endpoint shall abort the Command Message with a Response Message Status of Management Endpoint Buffer Cleared Due to Sanitize. Note that this Response Message Status is commonly associated with a Management Endpoint Buffer Read command but may be associated with any Command Message that uses the Management Endpoint Buffer as Request Data.

4.3 In-Band Tunneling Request Message Servicing Model

The in-band tunneling mechanism in this specification utilizes two NVMe Admin Commands (NVMe-MI Send and NVMe-MI Receive). Figure 70 specifies whether an NVMe-MI Command is tunneled via the NVMe-MI Send command or the NVMe-MI Receive command.

NVMe-MI Commands may apply to the NVM Subsystem, Controllers, and/or Namespaces. If a tunneled NVMe-MI Command applies to one or more Controllers, then the applicable Controller(s) are specified by fields in the tunneled NVMe-MI Command. Note that unlike some other NVMe Admin Commands, the Controller to which the tunneled NVMe-MI Command is issued is not used to determine which Controller the tunneled NVMe-MI Command applies to. If the tunneled NVMe-MI Command applies to one or more Namespaces, then the applicable Namespace(s) are specified by fields in the tunneled NVMe-MI Command. Note that the Namespace Identifier (NSID) field of the tunneled NVMe-MI Command (bytes 7:4 of the Submission Queue Entry) is not used and should be cleared to 0h by a host.

For details on the NVMe-MI Send command refer to:

- section 4.3.1; and
- the NVM Express Base Specification.

For details on the NVMe-MI Receive command refer to:

- section 4.3.2; and
- the NVM Express Base Specification.

4.3.1 NVMe-MI Send Command

The NVMe-MI Send command is an NVMe Admin Command as defined by this specification and the NVM Express Base Specification. It is used to tunnel an NVMe-MI Command in-band from a host to an NVMe Controller that transfers data from a host to an NVMe Controller (similar to a write operation) or to instruct the Responder to perform an action (e.g., to reset the NVM Subsystem using the Reset command). The data being transferred or action to be performed is in one or more of the following locations: the Request Data field, the NVMe Management Dword 0 field, the NVMe Management Dword 1 field. Figure 70 specifies which NVMe-MI Commands are tunneled via the NVMe-MI Send command.

4.3.1.1 NVMe-MI Send Command to NVMe Admin Command Mapping

In order to tunnel an NVMe-MI Command in-band via NVMe-MI Send, an NVMe-MI Request Message is mapped onto an NVMe Submission Queue Entry (SQE) as shown pictorially in Figure 48 and in table form in Figure 49. An NVMe-MI Response Message is mapped on to an NVMe Completion Queue Entry (CQE) as shown pictorially in Figure 50 and in table form in Figure 51. Refer to the NVM Express Base Specification for details on an NVMe Submission Queue Entry and an NVMe Completion Queue Entry.

Figure 48: NVMe-MI Send Command Request Message to NVMe Admin Command SQE Mapping Diagram

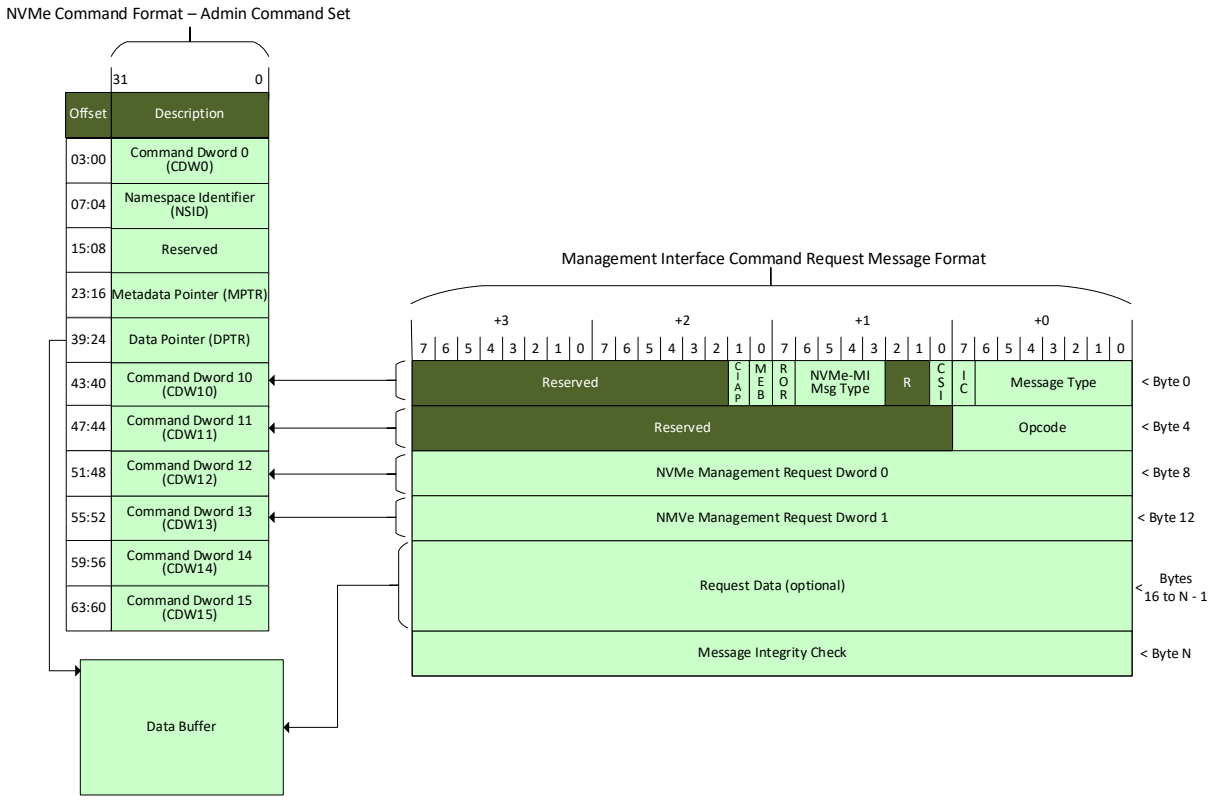


Figure 49: NVMe-MI Send Command Request Message to NVMe Admin Command SQE Mapping Table

NVMe-MI Command Request Message		NVMe Admin Command SQE Mapping	
Bytes	**Description**	**Bytes**	**Description**
Not applicable (n/a)	This field has no equivalent in this specification.	03:00	**Command Dword 0 (CDW0):** Refer to the NVM Express Base Specification.
n/a	If the tunneled NVMe-MI Command requires one or more Namespaces to be specified, then the applicable Namespace Identifiers are specified by the tunneled NVMe-MI Command.	07:04	**Namespace Identifier (NSID):** This field should be cleared to 0h by a host. Refer to the NVM Express Base Specification for more details.
n/a	These bytes have no equivalent in this specification.	23:08	Refer to the NVM Express Base Specification.
n/a	There is no equivalent of DPTR in this specification. In NVMe-MI Send, the Request Data is included in the Request Data portion of the Request Message.	39:24	**Data Pointer (DPTR):** This field contains a pointer to the start of the data buffer that contains the Request Data portion of the NVMe-MI Command that is being tunneled. If there is no Request Data for this command, then this field is ignored. Refer to the NVM Express Base Specification for the definition of this field.

Figure 49: NVMe-MI Send Command Request Message to NVMe Admin Command SQE Mapping Table

NVMe-MI Command Request Message		NVMe Admin Command SQE Mapping	
Bytes	**Description**	**Bytes**	**Description**
03:00	**NVMe-MI Message Header (NMH)**	43:40	**Command Dword 10 (CDW10):** Dword 0 of the Request Message (NMH) that is being tunneled maps to CDW10 of the SQE. The byte ordering within CDW10 is little endian (i.e., NMH byte 0 maps to CDW10 byte 0, NMH byte 1 maps to CDW10 byte 1, etc.).
04	**Opcode (OPC)**	47:44	**Command Dword 11 (CDW11):** Dword 1 of the Request Message (OPC and reserved bytes 7:5) that is being tunneled maps to CDW11 of the SQE. The byte ordering within CDW11 is little endian (i.e., OPC maps to CDW11 byte 0, the LSB of the Reserved field (NVMe-MI Command Request Message byte 5) maps to CDW11 byte 1, etc.).
07:05	Reserved		
11:08	**NVMe Management Dword 0 (NMD0)**	51:48	**Command Dword 12 (CDW12):** Dword 2 of the Request Message (NMD0) that is being tunneled maps to CDW12 of the SQE. The byte ordering within CDW12 is little endian (i.e., NMD0 byte 0 maps to CDW12 byte 0, NMD0 byte 1 maps to CDW12 byte 1).
15:12	**NVMe Management Dword 1 (NMD1)**	55:52	**Command Dword 13 (CDW13):** Dword 3 of the Request Message (NMD1) that is being tunneled maps to CDW13 of the SQE. The byte ordering within CDW13 is little endian (i.e., NMD1 byte 0 maps to CDW13 byte 0, NMD1 byte 1 maps to CDW13 byte 1).
n/a	This field has no equivalent in this specification.	59:56	**Command Dword 14 (CDW14):** Reserved.
n/a	This field has no equivalent in this specification.	63:60	**Command Dword 15 (CDW15):** Reserved.
N-1:16	**Request Data (REQD)**	n/a	Request Data is placed by a host into the data buffer pointed to by DPTR. If the Request Data is not dword granular, then the Request Data shall be padded with the minimum number of bytes cleared to 0h to make the Request Data dword granular. The byte ordering within the data buffer pointed to by DPTR is little endian (i.e., REQD byte 0 maps to byte 0 of the data buffer pointed to by DPTR, REQD byte 1 maps to byte 1 of the data buffer pointed to by DPTR, etc.).
N+3:N	**Message Integrity Check (MIC)**	n/a	The Message Integrity Check is not used in the in-band tunneling mechanism.

Figure 50: NVMe-MI Send Command Response Message to NVMe Admin Command CQE Mapping Diagram

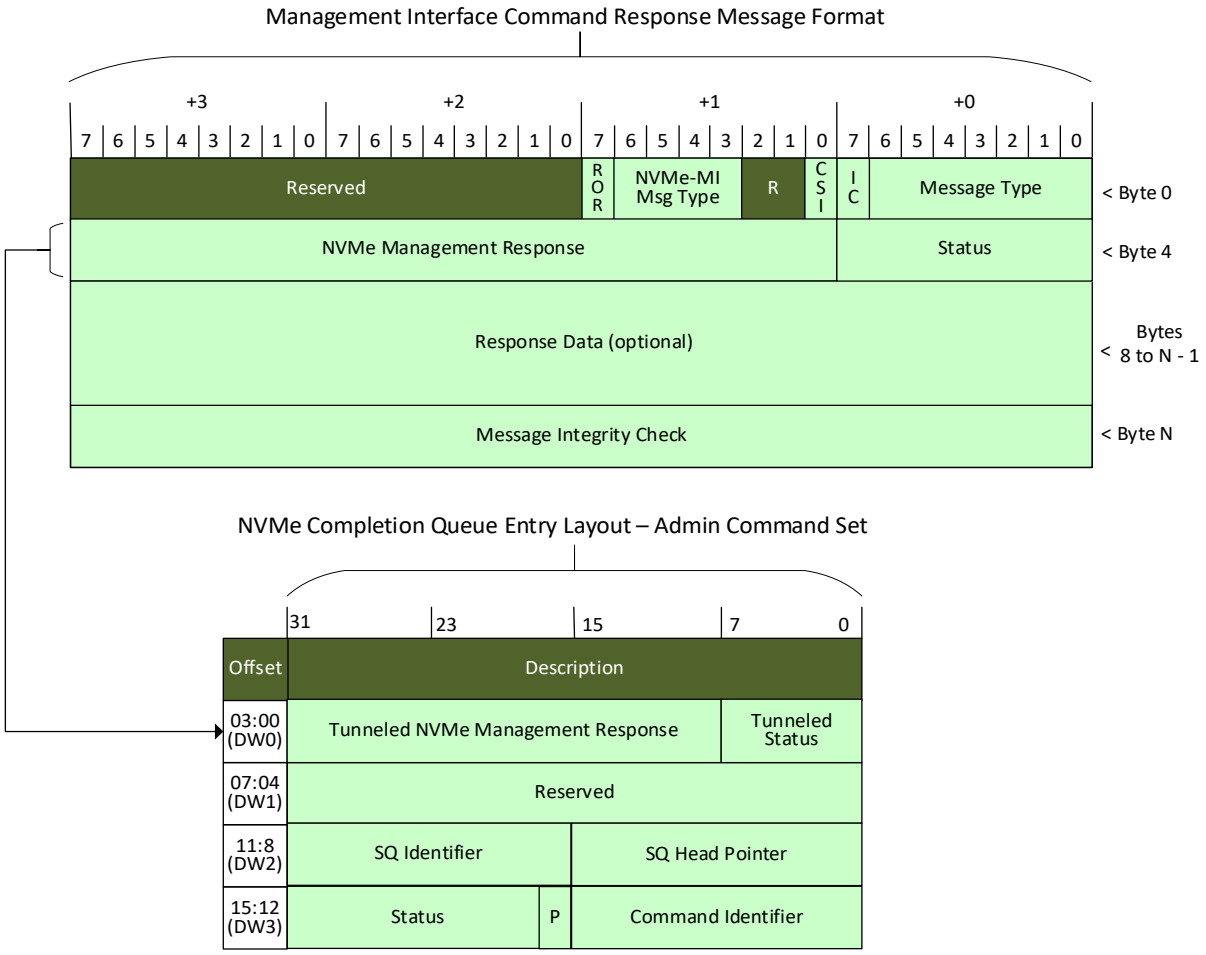


Figure 51: NVMe-MI Send Command Response Message to NVMe Admin Command CQE Mapping Table

NVMe-MI Command Response Message		NVMe Admin Command CQE Mapping	
Bytes	**Description**	**Bytes**	**Description**
00	**MCTP Data (MCTPD)**	n/a	This field has no equivalent in the NVMe Admin Command CQE.
01	**NVMe-MI Message Parameters (NMP)**	n/a	This field has no equivalent in the NVMe Admin Command CQE.
03:02	Reserved	n/a	This field has no equivalent in the NVMe Admin Command CQE.
04	**Status (STATUS)**	03:00	**Command Specific (DW0):** Dword 1 of the Response Message (STATUS and NMRESP) that is being tunneled maps to DW0 of the CQE. The byte ordering within DW0 is little endian (i.e., STATUS maps to DW0 byte 0, the LSB of the NMRESP field (NVMe-MI Command Response Message byte 5) maps to DW0 byte 1, etc.). Refer to Figure 52 for additional details on this field.
07:05	**NVMe Management Response (NMRESP)**		
N-1:8	**Response Data (RESPD)**	n/a	There is no Response Data for NVMe-MI Send.

Figure 51: NVMe-MI Send Command Response Message to NVMe Admin Command CQE Mapping Table

NVMe-MI Command Response Message		NVMe Admin Command CQE Mapping	
Bytes	**Description**	**Bytes**	**Description**
N+3:N	**Message Integrity Check (MIC)**	n/a	The Message Integrity Check is not used in the in-band tunneling mechanism.
n/a	These bytes have no equivalent in this specification.	15:04	Refer to the NVM Express Base Specification.

The definition of Dword 0 of the completion queue entry is in Figure 52.

Figure 52: NVMe-MI Send – Completion Queue Entry Dword 0 (NSCQED0)

Bits	Description
31:08	**Tunneled NVMe Management Response (TNMRESP):** This field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band. If any errors are detected in the NVMe Context as described in section 4.3.1.2, then this field shall be cleared to 0h.
07:00	**Tunneled Status (TSTAT):** This field contains the Status field from the NVMe-MI Command that is being tunneled in-band. If any errors are detected in the NVMe Context as described in section 4.3.1.2, then this field shall be cleared to 0h.

4.3.1.2 NVMe-MI Send Command Servicing Model

The NVMe-MI Send command servicing model is illustrated in Figure 53 as a series of phases and NVMe/NVMe-MI Contexts. The phases of the NVMe-MI Send command servicing model are further described in this section. The behavior of the portions of the figure in the NVMe Context are specified by the NVM Express Base Specification. The behavior of the portions of the figure in the NVMe-MI Context are specified by this specification. The phases and NVMe/NVMe-MI Contexts are logical constructs that illustrate the NVMe-MI Send command servicing model and do not mandate a particular implementation.

This section describes the NVMe-MI Send command servicing model starting at NVMe Processing as shown in phase 1 of Figure 53. In phase 1, CDW0 to CDW9 are checked for errors per the NVM Express Base Specification. If any errors are encountered in CDW0 to CDW9, then the NVMe-MI Send command is completed with an error status code in the Status Code field contained in the Status field as per the NVM Express Base Specification and the Tunneled Status and Tunneled NVMe Management Response fields shall be cleared to 0h.

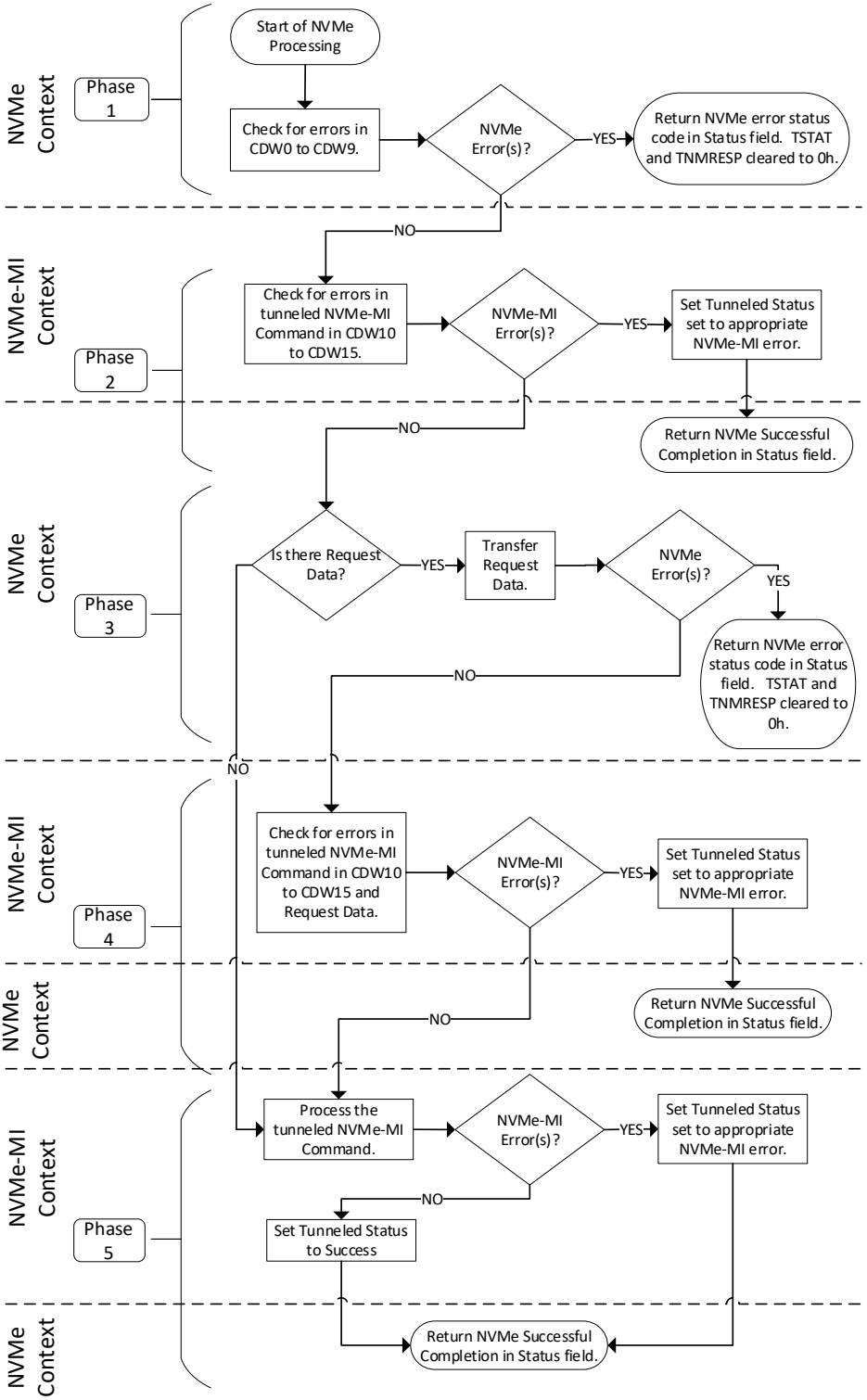
If there are no errors in CDW0 to CDW9, then command servicing enters phase 2 where the portion of the tunneled NVMe-MI Command in CDW10 to CDW15 is checked for errors. Note that if there is no Request Data, then CDW10 to CDW15 contain the entire tunneled NVMe-MI Command. If any errors are encountered in the portion of the tunneled NVMe-MI Command in CDW10 to CDW15, then the NVMe-MI Send command is completed with a status code of Successful Completion in the Status field as defined in the NVM Express Base Specification. The Tunneled Status field contains the error Response Message Status for the portion of the tunneled NVMe-MI Command in CDW10 to CDW15 and the Tunneled NVMe Management Response field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band.

If there are no errors in phase 2, then command servicing enters phase 3 where there is a check to determine if there is any Request Data for the tunneled NVMe-MI Command. If there is no Request Data for the tunneled NVMe-MI Command, then command servicing skips to phase 5. If there is Request Data, then the Request Data is transferred from the buffer pointed to by DPTR. If any errors are encountered transferring the Request Data, then the command is completed with an error status code in the Status Code field contained in the Status field as per the NVM Express Base Specification and the Tunneled Status and Tunneled NVMe Management Response fields shall be cleared to 0h.

If there are no errors transferring the data, then command servicing enters phase 4 where the whole tunneled NVMe-MI Command is constructed from CDW10 to CDW15 and the Request Data that was transferred. If any errors are encountered in the tunneled NVMe-MI Command, then the NVMe-MI Send command is completed with a status code of Successful Completion in the Status field as defined in the NVM Express Base Specification. The Tunneled Status field contains the appropriate error Response Message Status and the Tunneled NVMe Management Response field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band.

If there are no errors in phase 4, then command servicing enters phase 5 where the tunneled NVMe-MI Command finishes processing. If any errors are encountered processing the tunneled NVMe-MI Command, then the NVMe-MI Send command is completed with a status code of Successful Completion in the Status field as defined in the NVM Express Base Specification and the Tunneled Status field contains the appropriate error Response Message Status. If the tunneled NVMe-MI Command is processed successfully, then the NVMe-MI Send command is completed with a status code of Successful Completion in the Status field as defined in the NVM Express Base Specification. The Tunneled Status field contains a Response Message Status of Success for the tunneled NVMe-MI Command and the Tunneled NVMe Management Response field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band.

Figure 53: NVMe-MI Send Command Servicing Model



4.3.2 NVMe-MI Receive Command

The NVMe-MI Receive command is an NVMe Admin Command as defined by this specification and the NVM Express Base Specification. It is used to tunnel an NVMe-MI Command in-band from a host to an NVMe Controller that transfers data from an NVMe Controller to a host (similar to a read operation). The data being transferred is in one or more of the following locations: Response Data, NVMe Management Response. Figure 70 specifies which NVMe-MI Commands are tunneled via the NVMe-MI Receive command.

4.3.2.1 NVMe-MI Receive Command to NVMe Admin Command Mapping

In order to tunnel an NVMe-MI Command in-band via NVMe-MI Receive, an NVMe-MI Request Message is mapped onto an NVMe Submission Queue Entry (SQE) as shown pictorially in Figure 54 and in table form in Figure 55. An NVMe-MI Response Message is mapped on to an NVMe Completion Queue Entry (CQE) as shown pictorially in Figure 54 and in table form in Figure 56. Refer to the NVM Express Base Specification for details on an NVMe Submission Queue Entry and NVMe Completion Queue Entry.

Figure 54: NVMe-MI Receive Command Request/Response Message to NVMe Admin Command SQE/CQE Mapping Diagram

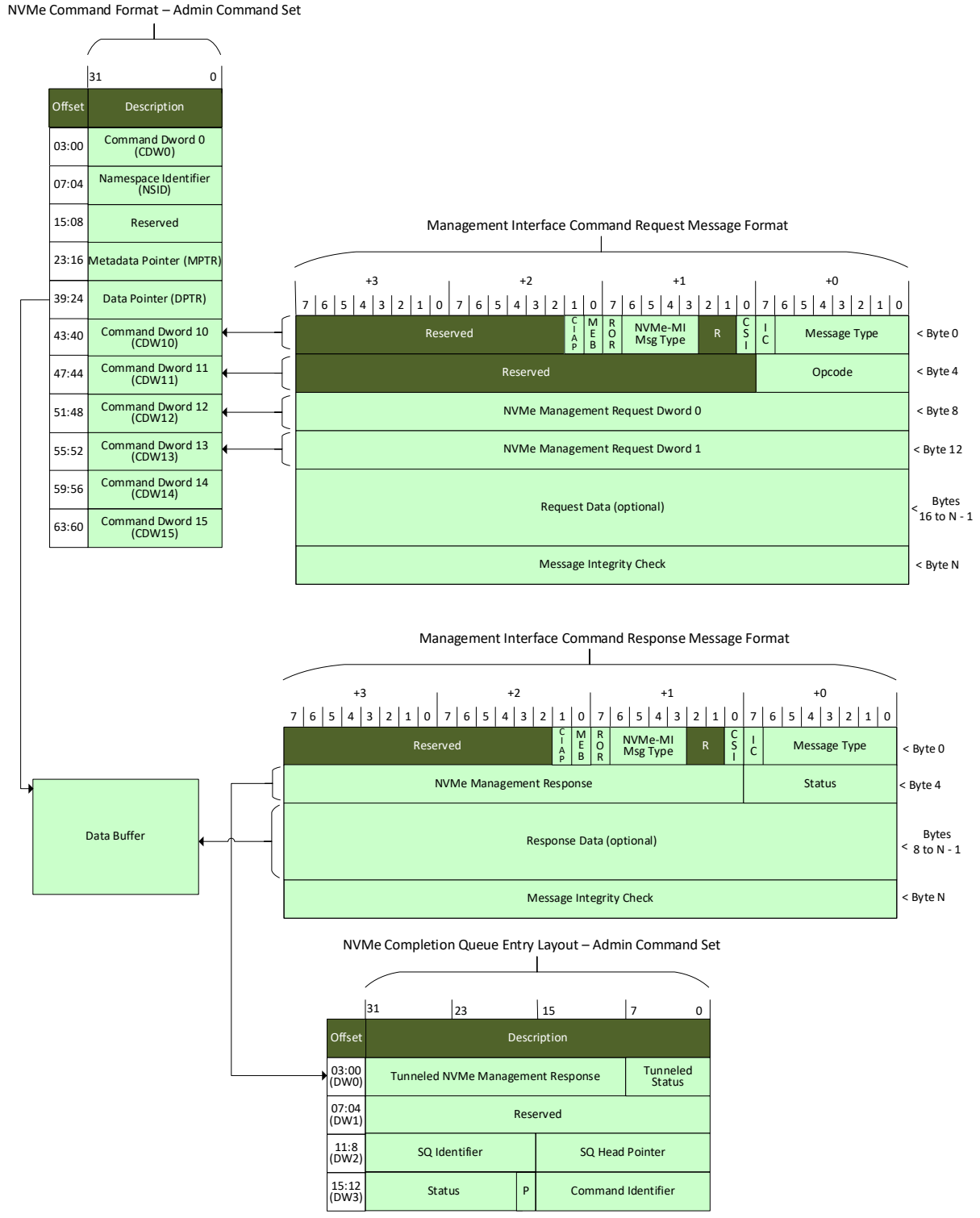


Figure 55: NVMe-MI Receive Command Request/Response Message to NVMe Admin Command SQE/CQE Mapping Table

NVMe-MI Command Request Message		NVMe Admin Command SQE Mapping	
Bytes	**Description**	**Bytes**	**Description**
n/a	This field has no equivalent in this specification.	03:00	**Command Dword 0 (CDW0):** Refer to the NVM Express Base Specification.
n/a	If the tunneled NVMe-MI Command requires one or more Namespaces to be specified, then the applicable Namespace Identifiers are specified by the tunneled NVMe-MI Command.	07:04	**Namespace Identifier (NSID):** This field should be cleared to 0h by a host. Refer to the NVM Express Base Specification for more details.
n/a	These bytes have no equivalent in this specification.	23:08	Refer to the NVM Express Base Specification.
n/a	There is no equivalent of DPTR in this specification. In NVMe-MI Receive, the Response Data is included in the Response Data portion of the Response Message.	39:24	**Data Pointer (DPTR):** This field contains a pointer to the start of the data buffer that contains the Response Data portion of the NVMe-MI Command that is being tunneled. If there is no Response Data for this command, then this field is ignored. Refer to the NVM Express Base Specification for the definition of this field.
03:00	**NVMe-MI Message Header (NMH)**	43:40	**Command Dword 10 (CDW10):** Dword 0 of the Request Message (NMH) that is being tunneled maps to CDW10 of the SQE. The byte ordering within CDW10 is little endian (i.e., NMH byte 0 maps to CDW10 byte 0, NMH byte 1 maps to CDW10 byte 1, etc.).
04	**Opcode (OPC)**	47:44	**Command Dword 11 (CDW11):** Dword 1 of the Request Message (OPC and Reserved bytes 7:5) that is being tunneled maps to CDW11 of the SQE. The byte ordering within CDW11 is little endian (i.e., OPC maps to CDW11 byte 0, the LSB of the Reserved field (NVMe-MI Command Request Message byte 5) maps to CDW11 byte 1, etc.).
07:05	Reserved		
11:08	**NVMe Management Dword 0 (NMD0)**	51:48	**Command Dword 12 (CDW12):** Dword 2 of the Request Message (NMD0) that is being tunneled maps to CDW12 of the SQE. The byte ordering within CDW12 is little endian (i.e., NMD0 byte 0 maps to CDW12 byte 0, NMD0 byte 1 maps to CDW12 byte 1).
15:12	**NVMe Management Dword 1 (NMD1)**	55:52	**Command Dword 13 (CDW13):** Dword 3 of the Request Message (NMD1) that is being tunneled maps to CDW13 of the SQE. The byte ordering within CDW13 is little endian (i.e., NMD1 byte 0 maps to CDW13 byte 0, NMD1 byte 1 maps to CDW13 byte 1).
n/a	This field has no equivalent in this specification.	59:56	**Command Dword 14 (CDW14):** Reserved.
n/a	This field has no equivalent in this specification.	63:60	**Command Dword 15 (CDW15):** Reserved.
N-1:16	**Request Data (REQD)**	n/a	There is no Request Data for NVMe-MI Receive.
N+3:N	**Message Integrity Check (MIC)**	n/a	The Message Integrity Check is not used in the in-band tunneling mechanism.

Figure 56: NVMe-MI Receive Command Response Message to NVMe Admin Command CQE Mapping Table

NVMe-MI Command Response Message		NVMe Admin Command CQE	
Bytes	**Description**	**Bytes**	**Description**
00	**MCTP Data (MCTPD)**	n/a	This field has no equivalent in the NVMe Admin Command CQE.
01	**NVMe-MI Message Parameters (NMP)**	n/a	This field has no equivalent in the NVMe Admin Command CQE.
03:02	Reserved	n/a	This field has no equivalent in the NVMe Admin Command CQE.
04	**Status (STATUS)**	03:00	**Command Specific (DW0):** Dword 1 of the Response Message (STATUS and NMRESP) that is being tunneled maps to DW0 of the CQE. The byte ordering within DW0 is little endian (i.e., STATUS maps to DW0 byte 0, the LSB of the NMRESP field (NVMe-MI Command Response Message byte 5) maps to DW0 byte 1, etc.). Refer to Figure 57 for additional details on this field.
07:05	**NVMe Management Response (NMRESP)**		
N-1:8	**Response Data (RESPD)**	n/a	Response Data is placed by the NVMe Controller into the data buffer pointed to by DPTR. If the Response Data size is not dword granular, then the Response Data shall be padded with the minimum number of bytes cleared to 0h to make the Response Data dword granular. The byte ordering within the data buffer pointed to by DPTR is little endian (i.e., RESPD byte 0 maps to byte 0 of the data buffer pointed to by DPTR, RESPD byte 1 maps to byte 1 of the data buffer pointed to by DPTR, etc.).
N+3:N	**Message Integrity Check (MIC)**	n/a	The Message Integrity Check is not used in the in-band tunneling mechanism.
n/a	These bytes have no equivalent in this specification.	15:04	Refer to the NVM Express Base Specification.

The definition of Dword 0 of the completion queue entry is in Figure 57.

Figure 57: NVMe-MI Receive – Completion Queue Entry Dword 0 (NRCQED0)

Bits	Description
31:08	**Tunneled NVMe Management Response (TNMRESP):** This field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band. If any errors are detected in the NVMe Context as described in section 4.3.2.2, then this field shall be cleared to 0h.
07:00	**Tunneled Status (TSTAT):** This field contains the Status field from the NVMe-MI Command that is being tunneled in-band. If any errors are detected in the NVMe Context as described in section 4.3.2.2, then this field shall be cleared to 0h.

4.3.2.2 NVMe-MI Receive Command Servicing Model

The NVMe-MI Receive command servicing model is illustrated in Figure 58 as a series of phases (described in this section) and NVMe/NVMe-MI Contexts. The phases of the NVMe-MI Receive command servicing model are further described in this section. The behavior of the portions of the figure in the NVMe Context are specified by the NVM Express Base Specification. The behavior of the portions of the figure in the NVMe-MI Context are specified by this specification. The phases and NVMe/NVMe-MI Contexts are logical constructs that illustrate the NVMe-MI Receive command servicing model and do not mandate a particular implementation.

This section describes the NVMe-MI Receive command servicing model starting at NVMe Processing as shown in phase 1 of Figure 58. In phase 1, CDW0 to CDW9 are checked for errors per the NVM Express Base Specification. If any errors are encountered in CDW0 to CDW9, then the command is completed with

an error status code in the Status field as per the NVM Express Base Specification and the Tunneled Status and Tunneled NVMe Management Response fields shall be cleared to 0h.

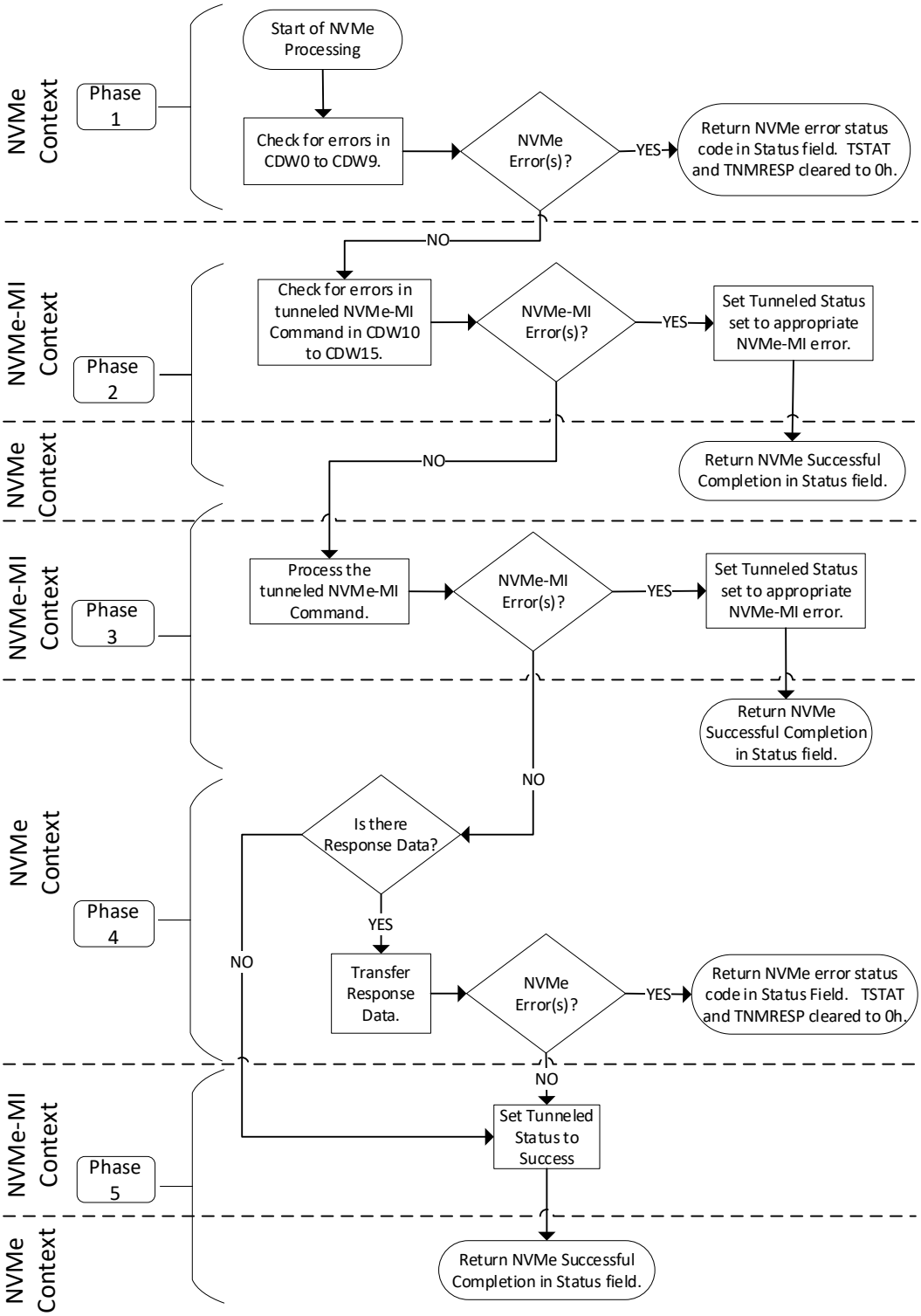
If there are no errors in CDW0 to CDW9, then command servicing enters phase 2 where the tunneled NVMe-MI Command in CDW10 to CDW15 is checked for errors. If any errors are encountered in the tunneled NVMe-MI Command in CDW10 to CDW15, then the NVMe-MI Receive command is completed with a status code of Successful Completion in the Status field as defined in the NVM Express Base Specification. The Tunneled Status field contains the appropriate error Response Message Status and the Tunneled NVMe Management Response field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band.

If there are no errors in phase 2, then command servicing enters phase 3 where the tunneled NVMe-MI Command finishes processing. If any errors are encountered processing the tunneled NVMe-MI Command, then the NVMe-MI Receive command is completed with a status code of Successful Completion in the Status field as defined in the NVM Express Base Specification. The Tunneled Status field contains the appropriate error Response Message Status and the Tunneled NVMe Management Response field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band.

If there are no errors in phase 3, then command servicing enters phase 4 where there is a check to determine if there is any Response Data for the tunneled NVMe-MI Command. If there is no Response Data for the tunneled NVMe-MI Command, then command servicing skips to phase 5. If there is Response Data, then the Response Data is transferred to the buffer pointed to by DPTR. If any errors are encountered transferring the Response Data then the command is completed with an error status code in the Status field as per the NVM Express Base Specification and the Tunneled Status and Tunneled NVMe Management Response fields shall be cleared to 0h.

If there are no errors in phase 4, then command servicing enters phase 5 where the NVMe-MI Receive command is completed with a status code of Successful Completion in the Status field as defined in the NVM Express Base Specification. The Tunneled Status field contains a Response Message Status of Success for the tunneled NVMe-MI Command and the Tunneled NVMe Management Response field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band.

Figure 58: NVMe-MI Receive Command Servicing Model



4.4 Out-of-Band AEM Servicing Model

If a Management Endpoint supports AEMs, then the Management Endpoint is in the AE Armed State or the AE Disarmed State as specified in section 4.4.1. AEs that occur (refer to section 4.4.5) are transmitted in an AE Occurrence data structure in an AEM. AEMs do not include AE Occurrence data structures for AEs that have not occurred.

AEMs are transmitted by a Management Endpoint during the AEM Transmission Interval as specified in section 4.4.3. AEMs are not transmitted outside of the AEM Transmission Interval.

AEMs shall not occupy Command Slots. Like Control Primitive Response Messages, AEMs may be transmitted while the Command Slot is in any command servicing state and should be transmitted as soon as possible by the Management Endpoint. For example, if the Management Endpoint is in the middle of receiving a multi-packet Request Message when an AE occurs, then it is recommended that the Management Endpoint transmit the AEM prior to the completion of the Request Message transfer if an AEM Transmission Interval is permitted during that time (refer to section 4.4.3). Likewise, if the Management Endpoint is in the middle of transmitting a multi-packet Response Message when an AE occurs, then it is recommended that the Management Endpoint transmit the AEM prior to the completion of the Response Message transfer if an AEM Transmission Interval is permitted during that time. AEMs shall not change the command servicing state of the Command Slots.

4.4.1 Management Endpoint AE Armed State and AE Disarmed State

A Management Endpoint that supports AEMs is either in the AE Armed State or in the AE Disarmed State.

An AE Arm (refer to section 1.8.5 and section 5.2.4.2) occurs when the Management Endpoint processes an AE Sync or AEM Ack that leaves one or more AEs enabled.

The AE Armed State shall start when an AE Arm occurs. The AE Armed State shall end when the AE Disarmed State starts.

The AE Disarmed State shall start when:

a) all supported AEs are disabled (e.g., by an AE Sync or by a Management Endpoint Reset); or
b) the AEM Transmission Interval starts.

The AE Disarmed State shall end when the AE Armed State starts.

If a Management Endpoint in the AE Armed State has one or more AEs occur, then the Management Endpoint shall transmit an AEM during the next AEM Transmission Interval (refer to section 4.4.3) unless otherwise specified (e.g., a Management Endpoint Reset occurs prior to the next AEM Transmission Interval).

4.4.2 AEM Delay Interval

The AEM Delay Interval is the time during which a Management Endpoint shall wait before transmitting an AEM for any AEs that have occurred during that AEM Delay Interval. The AEM Delay Interval shall start when the Management Endpoint enters the AE Armed State. The AEM Delay Interval shall end once the amount of time specified by the AEM Delay field has elapsed since the start of the AEM Delay Interval or the Management Endpoint enters the AE Disarmed State.

4.4.3 AEM Transmission Interval

AEMs for AEs that have occurred during an AE Armed State shall only be transmitted during the subsequent AEM Transmission Interval.

That AEM Transmission Interval shall start once:

a) the amount of time specified by the AEM Delay field has elapsed since the start of the current AE Armed State; and

b) at least one AE has occurred during the current AE Armed State (refer to section 4.4.5).

Once the AEM Transmission Interval starts, the Management Endpoint shall transmit a single AEM. Once the AEM Transmission Interval starts, the AEM should be transmitted as soon as possible. The contents of an AEM are defined in section 4.4.4.

That AEM Transmission Interval shall end once:

a) an AE Sync occurs (refer to section 5.2.4);
b) an AEM Ack occurs (refer to section 5.2.4);
c) an AEM transmission failure occurs (refer to the AEM Transmission Failure bit); or
d) a Management Endpoint Reset occurs.

Before exiting the AEM Transmission Interval, if an AEM transmission is in progress, then the Management Endpoint shall stop transmitting the AEM and should stop transmitting the AEM as soon as possible.

If the AEM Retry Delay field is not cleared to 0h and the amount of time specified by the AEM Retry Delay field elapses from:

a) the end of transmitting an AEM without receiving an AEM Ack; or
b) when a failure to transmit an AEM occurs due to the physical transport external to the NVM Subsystem being unavailable (e.g., an AEM on a PCIe VDM Management Endpoint is unable to be transmitted due to the PCIe link being down),

then the Management Endpoint shall retry the AEM transmission and should retry the AEM transmission as soon as possible until:

a) there have been 8 total attempts to transmit the AEM (the first transmission attempt and seven retries);
b) an AE Sync occurs (refer to section 5.2.4);
c) an AEM Ack occurs (refer to section 5.2.4); or
d) a Management Endpoint Reset occurs.

If an AEM transmission is retried, then the contents of the AEM shall exactly match the contents of the AEM transmitted during the first transmission attempt with the exception that the AEM Retry Count field is incremented each time the AEM transmission is retried. The retried AEM transmission shall not include an AE Occurrence data structure for any AEs that have occurred since the start of the AEM Transmission Interval. AEs that have occurred since the start of the AEM Transmission Interval are returned in the Response Message of an AEM Ack (refer to section 5.2.4).

4.4.4 AEM Format

The format of an AEM is shown in Figure 59 and the fields are described in Figure 60.

Figure 59: Asynchronous Event Message (AEM) Format

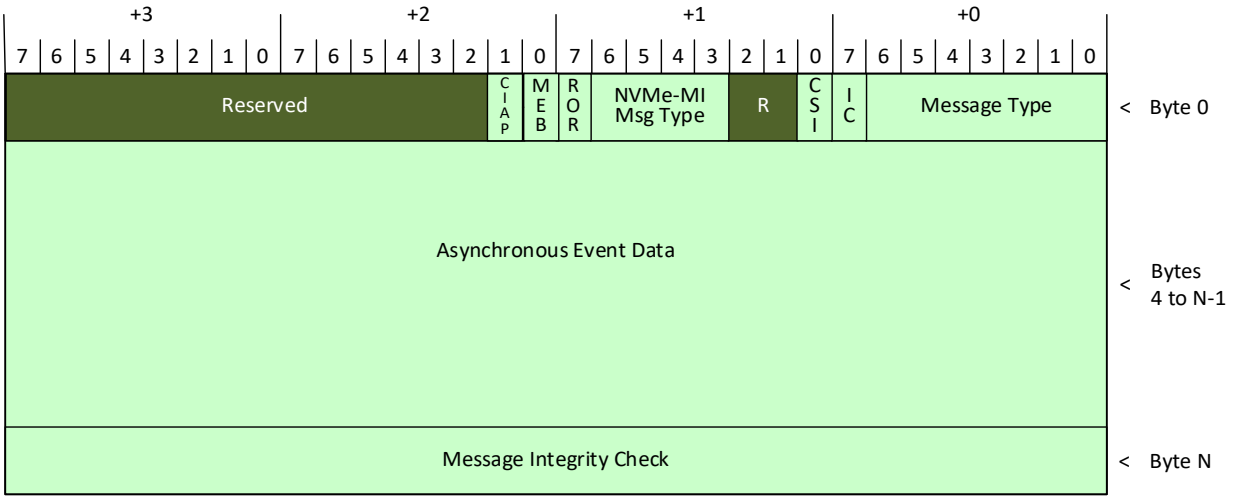


Figure 60: Asynchronous Event Message (AEM) Fields

Bytes	Description
3:0	**NVMe-MI Message Header (NMH):** Refer to section 3.1.
N-1:4	**Asynchronous Event Data (AED):** This field contains an AE Occurrence List data structure (refer to Figure 61).
N+3:N	**Message Integrity Check (MIC):** Refer to section 3.1.

An AEM shall be transmitted to the entity that most recently caused an AE Arm. All fields in the MCTP packet header (refer to Figure 24) of an AEM shall be set as specified by the MCTP Base Specification with the following additional requirements:

a) the Msg tag shall be selected by the Management Endpoint and the Tag Owner bit shall be set to '1' since the AEM originates from the Management Endpoint; and
b) the Destination Endpoint ID shall be set to the value of the Source Endpoint ID of the entity that most recently caused an AE Arm.

For an AEM originating from a PCIe VDM Management Endpoint, all PCIe VDM header fields shall be set as specified in the MCTP PCIe VDM Transport Binding Specification with the following additional requirements:

a) the PCI Target ID field shall be set to the value of the PCI Requester ID of the entity that most recently caused an AE Arm; and
b) bits 2:0 of the Type field shall be set to a value of 010b to indicate the PCIe message routing is Route by ID.

There are no special transport header field requirements to transmit an AEM from a 2-Wire Management Endpoint in either SMBus mode or I3C mode.

The Asynchronous Event Data field shall contain an AE Occurrence List data structure (refer to Figure 61) and shall be minimally sized (i.e., if there is one AE Occurrence data structure, then the AE Occurrence List data structure is the length of that AE Occurrence data structure plus the length of the AE Occurrence List Header). The AE Occurrence List data structure shall start at offset 0h of the Asynchronous Event Data field.

If the number of AEs that have occurred do not result in the length of the AE Occurrence List Body exceeding 4 KiB, then:

a) the AE Occurrence List Overflow bit shall be cleared to '0';
b) the AE Occurrence List Body shall contain an AE Occurrence data structure for the most recent occurrence of each AE of a given AE Unique ID that occurred in the prior AE Armed State;
c) the AE Occurrence List Body shall not contain an AE Occurrence data structure for any AEs that did not occur during the prior AE Armed State; and
d) each AE Occurrence data structure shall indicate the state of the AE at the time the AE occurred which is used to resynchronize the state of the AEs between the Management Controller and the Management Endpoint.

If the number of AEs that have occurred would result in the length of the AE Occurrence List Body exceeding 4 KiB if the AE Occurrence data structure for each AE that has occurred was included in the AE Occurrence List Body, then:

a) the AE Occurrence List Overflow bit shall be set to '1'; and
b) the AE Occurrence List data structure shall contain the AE Occurrence List Header and shall not contain an AE Occurrence List Body.

Figure 61: AE Occurrence List Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><th colspan="2">AE Occurrence List Header</th></tr>
<tr><td>0</td><td>Number of AE Occurrence Data Structures (NUMAEO): This field shall indicate the number of AE Occurrence data structures (refer to Figure 62) in the AE Occurrence List Body.

If there are no AE Occurrence data structures in the AE Occurrence List Body, then this field shall be cleared to 0h.</td></tr>
<tr><td>1</td><td>AE Occurrence List Version Number (AELVER): This field indicates the version number of the AE Occurrence List data structure and the AE Occurrence data structure. This field shall be cleared to 0h.</td></tr>
<tr><td>4:2</td><td>AE Occurrence List Length Info (AEOLLI): This field indicates info about the length of the AE Occurrence List data structure.

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>23</td><td>AE Occurrence List Overflow (AEOLO): This bit indicates if an AE Occurrence List data structure overflow has occurred. If an AE Occurrence List data structure overflow occurs, then an AE Sync is able to be used to resynchronize the state of the AEs between the Management Controller and the Management Endpoint.

Refer to section 5.2.4 and this section for more details.</td></tr>
<tr><td>22:00</td><td>AE Occurrence List Total Length (AEOLTL): This field indicates the length in bytes of the AE Occurrence List data structure. This field shall be set to a value equal to the value of the AE Occurrence List Header Length field plus the sum of the lengths in bytes of each AE Occurrence data structure in the AE Occurrence List Body.

If the AE Occurrence List Overflow bit is set to '1', then this field shall be cleared to 0h.</td></tr>
</table></td></tr>
<tr><td>5</td><td>AE Occurrence List Header Length (AEOLHL): This field shall indicate the length in bytes of the AE Occurrence List Header. This field shall be set to 7h.</td></tr>
</table>

Figure 61: AE Occurrence List Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td colspan="2">AE Occurrence List Header</td></tr>
<tr><td>6</td><td>AEM Transmission Info (AEMTI): This field indicates information about the AEM transmission. For the AE Occurrence List data structure in the Response Message to an AEM Ack or an AE Sync, this field is not applicable and shall be cleared to 0h.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:3</td><td>AEM Generation Number (AEMGN): This field shall indicate a value that is incremented on the first attempt to transmit an AEM (i.e., the AEMRC field is cleared to 0h) and shall not be incremented for any other reason (e.g., this field is not incremented when an AEM transmission is retried).

If the value of this field is 1Fh, then this field shall be cleared to 0h when incremented (i.e., rolls over to 0h).

Note that there are race cases where a Management Endpoint retries an AEM transmission due to a timeout at the same time the Management Controller is transmitting the AEM Ack for the AEM. In this scenario, the Management Controller may receive duplicate AEMs. This field is used to determine whether the AEM is a duplicate (e.g., if the value in the Generation Number field in the most recently received AEM is the same as the value in the Generation Number field of the prior AEM, then the AEM is a duplicate and should be ignored by the Management Controller).</td></tr>
<tr><td>2:0</td><td>AEM Retry Count (AEMRC): For an AEM, this field shall indicate the number of times the AEM has been retried. A value of 0h indicates the first attempt to transmit the AEM, a value of 1h indicates the second attempt (i.e., the first retry) to transmit the AEM, etc.</td></tr>
</table></td></tr>
<tr><td colspan="2">AE Occurrence List Body</td></tr>
<tr><td>AEOLHL+(L-1):AEOLHL</td><td>AE Occurrence 0: This field shall indicate the first AE Occurrence data structure (refer to Figure 62), if any, where L is the length in bytes of this AE Occurrence data structure.</td></tr>
<tr><td>AEOLHL+L+(M-1):
AEOLHL+L</td><td>AE Occurrence 1: This field shall indicate the second AE Occurrence data structure, if any, where L is the length in bytes of the first AE Occurrence data structure and M is the length in bytes of this AE Occurrence data structure.</td></tr>
<tr><td colspan="2">…</td></tr>
<tr><td>AEOLTL-1: AEOLTL-N</td><td>AE Occurrence N: This field shall indicate the last AE Occurrence data structure, if any, where N is the length in bytes of this AE Occurrence data structure.</td></tr>
</table>

Figure 62: AE Occurrence Data Structure

Bytes	Description
AE Occurrence Header	
0	**AE Occurrence Header Length (AELHLEN):** This field shall indicate the length in bytes of the AE Occurrence Header. This field shall be set to 9h.
1	**AE Occurrence Specific Info Length (AEOSIL):** This field shall indicate the length in bytes of the AE Occurrence Specific Info field. If there is no AE Occurrence Specific Info field for this AE, then this field shall be cleared to 0h. If this AE is vendor specific (i.e., the AE Identifier is in the range C0h to FFh), then this field shall be cleared to 0h.
2	**AE Occurrence Vendor Specific Info Length (AEOVSIL):** This field shall indicate the length in bytes of the AE Occurrence Vendor Specific Info field. If there is no AE Occurrence Vendor Specific Info field for this AE, then this field shall be cleared to 0h.

Figure 62: AE Occurrence Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td colspan="2">AE Occurrence Header</td></tr>
<tr><td>8:3</td><td>
AE Occurrence Unique ID (AEOUI): This field indicates an identifier for the AE Occurrence data structure that is unique within a given AE Occurrence List data structure.
<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>0</td><td>AE Occurrence ID (AEOI): This field shall indicate the identifier of the AE (refer to Figure 63).</td></tr>
<tr><td>4:1</td><td>AE Occurrence Scope ID Info (AEOCIDI): This field indicates info about the scope identifier associated with the AE. The format of this field is defined in Figure 64. The scope of the AE is indicated by the AE Occurrence Scope field.</td></tr>
<tr><td>5</td><td>
AE Occurrence Scope Info (AESSI): This field indicates info about the scope of the AE.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:4</td><td>Reserved</td></tr>
<tr><td>3:0</td><td>
AE Occurrence Scope (AESS): This field shall indicate the scope of the AE.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0h</td><td>Namespace</td></tr>
<tr><td>1h</td><td>Controller</td></tr>
<tr><td>2h</td><td>NVM Subsystem</td></tr>
<tr><td>3h</td><td>Management Endpoint</td></tr>
<tr><td>4h</td><td>Port</td></tr>
<tr><td>5h</td><td>Endurance Group</td></tr>
<tr><td>6h to Fh</td><td>Reserved</td></tr>
</table>
</td></tr>
</table>
</td></tr>
</table>
</td></tr>
<tr><td colspan="2">AE Occurrence Specific Info</td></tr>
<tr><td>AEOSIL+ AELHLEN-1: AELHLEN</td><td>AE Occurrence Specific Info (AEOSI): This field shall indicate info specific to the AE (refer to Figure 65), if applicable. If the value of the AEOSIL field is 0h, then this field is not included.</td></tr>
<tr><td>N:M</td><td>
AE Occurrence Vendor Specific Info (AEOVSI): This field indicates vendor-specific info specific to this AE, if applicable. If the value of the AEOVSIL field is 0h, then this field is not included.

M is equal to the value indicated by the AEOSIL field plus the value indicated by the AELHLEN field. N is equal to the value of the AELHLEN field plus the value of the AEOSIL field plus the value of the AEOVSIL field minus 1h.
<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>M</td><td>
AE Occurrence Vendor Specific Header (AEOVSH): This field indicates information about the AE occurrence vendor-specific information.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Vendor specific</td></tr>
<tr><td>6:0</td><td>AE Occurrence Vendor Specific UUID Index (AEOVSUI): If this field is set to a non-zero value, then the value of this field shall indicate the index of a UUID in the UUID List (refer to the NVM Express Base Specification) corresponding to the vendor that defined the AE Occurrence Vendor Specific Info.

If no UUID index is specified, then this field shall be cleared to 0h.</td></tr>
</table>
</td></tr>
<tr><td>N:M+1</td><td>Vendor specific</td></tr>
</table>
</td></tr>
</table>

4.4.5 AE Identifier Information

The AEs, AE Identifiers, AE scope, and the only conditions that trigger an AE occurrence are defined in Figure 63. If the AEM Delay field (refer to Figure 91) is greater than 0h, then the conditions that trigger an AE occurrence shall be checked at a frequency of less than or equal to the amount of time specified by the

AEM Delay field. If the AEM Delay field is equal to 0h, then the conditions that trigger an AE occurrence shall be checked at a frequency of less than or equal to 1 s. For example, if the AEM Delay field specifies a value of 5 s and the Composite Temperature AE is enabled, then the Management Endpoint shall check for a change in composite temperature at least once every 5 s.

Figure 63: Asynchronous Events

Identifier	Scope [1]	AE	AE Occurrence Trigger when the AE is Enabled
00h	Controller	Controller Ready	The Controller ready state changes. This is the same ready state that is indicated by the CSTS.RDY bit (refer to the NVM Express Base Specification).
01h	Controller	Controller Fatal Status	The Controller fatal status changes. This is the same Controller fatal status that is indicated by the CSTS.CFS (refer to the NVM Express Base Specification).
02h	Controller or NVM Subsystem	Shutdown Status	The following conditions, as defined by the NVM Express Base Specification, trigger this AE to occur with NVM Subsystem scope: • an NVM Subsystem Shutdown starts; or • an NVM Subsystem Shutdown completes. Note that even though every Controller in the NVM Subsystem indicates when the NVM Subsystem Shutdown starts or completes, a single Shutdown Status AE occurrence is triggered for the entire NVM Subsystem when the NVM Subsystem Shutdown starts, and a single Shutdown Status AE occurrence is triggered for the entire NVM Subsystem when the NVM Subsystem Shutdown completes. The following conditions, as defined by the NVM Express Base Specification, trigger this AE to occur with Controller scope: • a Controller shutdown starts; or • a Controller shutdown completes.
03h	Controller	Controller Enable	The Controller enable state changes. This is the same Controller enable state that is reported by the CC.EN bit (refer to the NVM Express Base Specification).
04h	Controller	Namespace Attribute Changed	One or more Namespace attributes change. These are the same Namespaces attribute changes reported by the Namespace Attribute Changed bit in the Controller Status field in the Controller Health data structure.
05h	NVM Subsystem	Firmware Activated	Firmware activation status changes. This is the same firmware activation status that is reported by the Firmware Activated bit in the Composite Controller Status field in the NVM Subsystem Health data structure.
06h	NVM Subsystem	Composite Temperature	The composite temperature changes. This is the same composite temperature that is reported by the Composite Temperature field in the NVM Subsystem Health data structure.
07h	NVM Subsystem	Percentage Drive Life Used	The percentage of NVM Subsystem life used changes. This is the same NVM Subsystem life used that is reported by the Percentage Drive Life Used field in the NVM Subsystem Health data structure.
08h	NVM Subsystem	Available Spare	The amount of available spare capacity in the NVM Subsystem changes. This is the same available spare capacity that is reported by the Available Spare bit in the Composite Controller Status Flags field. If the amount of available spare capacity is not NVM Subsystem scoped, then this AE shall not be supported.

Figure 63: Asynchronous Events

Identifier	Scope [1]	AE	AE Occurrence Trigger when the AE is Enabled
09h	NVM Subsystem	SMART Warnings	The critical warning state of any Controller in the NVM Subsystem changes. This is the same critical warning state that is reported by the Critical Warning field in the SMART / Health Information log page in the NVM Express Base Specification and the SMART Warnings field in the NVM Subsystem Health data structure.
0Ah	Controller or NVM Subsystem	Telemetry Controller-Initiated Data Available	Telemetry Controller-Initiated log page generation status changes. This is the same Telemetry Controller-Initiated log page generation status that is reported by the Telemetry Controller-Initiated Data Available bit in the Composite Controller Status Flags field and the Telemetry Controller-Initiated Data Available bit in the Controller Health data structure.
0Bh	Port	PCIe Link Active	The link active state changes. This is the same link active state that is reported by the Port 0 PCIe Link Active bit in the NVM Subsystem Health data structure and the Port 1 PCIe Link Active bit in the NVM Subsystem Health data structure. The AE Occurrence Port Type bit for this AE shall be set '1' to indicate that the AE Occurrence Port ID field contains the NVMe-MI port associated with the event.
0Ch	NVM Subsystem	Sanitize Failure Mode	The sanitize NVM Subsystem failure state changes. This is the same sanitize NVM Subsystem failure state that is reported by the Sanitize Failure Mode bit in the NVM Subsystem Health data structure.
0Dh	Namespace	Sanitize Namespace Failure Mode	The sanitize namespace failure state changes. This is the same sanitize namespace failure state that is reported by the Sanitize Namespace Failure Mode bit in the NVM Subsystem Health data structure.
0Eh	Controller	Power Threshold Exceeded	The interval power measurement (i.e., average of power measurement samples over one second) is greater than or equal to a power threshold (refer to the NVM Express Base Specification).
0Fh to BFh	Reserved		
C0h to FFh	Vendor specific		

Notes:
1. The AE Occurrence Scope field of the AE Occurrence data structure indicates the scope of the AE (refer to Figure 62). The AE Occurrence Scope ID Info field of the AE Occurrence data structure contains the identifier of the entity associated with the AE in a format that matches the scope indicated in the AE Occurrence Scope field (refer to Figure 64).

Figure 64: AE Occurrence Scope ID Info Field Format

Scope	Value
Namespace	**Namespace Scope ID Info (NSIDI):** If this AE is Namespace scoped (i.e., the value of the AE Occurrence Scope field is 0h), then this field indicates information related to the AE Occurrence Scope ID of the Namespace. **Bits** / **Description** 31:00 — **AE Occurrence Namespace ID (AEONSID):** This field shall indicate the NSID of the Namespace associated with the AE.

Figure 64: AE Occurrence Scope ID Info Field Format

<table>
<tr><th>Scope</th><th>Value</th></tr>
<tr><td>Controller</td><td>Controller Scope ID Info (CSIDI): If this AE is Controller scoped (i.e., the value of the AE Occurrence Scope field is 1h), then this field indicates information related to the AE Occurrence Scope ID of the Controller.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>31:16</td><td>Reserved</td></tr><tr><td>15:00</td><td>AE Occurrence Controller ID (AEOCID): This field shall indicate the Controller ID of the Controller associated with the AE.</td></tr></table></td></tr>
<tr><td>NVM Subsystem</td><td>NVM Subsystem Scope ID Info (NSSIDI): If this AE is NVM Subsystem scoped (i.e., the value of the AE Occurrence Scope field is 2h), then this field indicates information related to the AE Occurrence Scope ID of the NVM Subsystem.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>31:00</td><td>Reserved</td></tr></table></td></tr>
<tr><td>Management Endpoint</td><td>Management Endpoint Scope ID Info (MESI): If this AE is Management Endpoint scoped (i.e., the value of the AE Occurrence Scope field is 3h), then this field indicates information related to the AE Occurrence Scope ID of the Management Endpoint.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>31:08</td><td>Reserved</td></tr><tr><td>07:00</td><td>AE Occurrence Management Endpoint ID (AEOMEID): This field shall indicate the Endpoint ID of the Management Endpoint associated with the AE.</td></tr></table></td></tr>
<tr><td>Port</td><td>Port Scope ID Info (PSI): If this AE is port scoped (i.e., the value of the AE Occurrence Scope field is 4h), then this field indicates information related to the AE Occurrence Scope ID of the port.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>31:17</td><td>Reserved</td></tr><tr><td>16</td><td>AE Occurrence Port Type (AEOPT): If the AE is associated with an NVM Subsystem port (refer to the NVM Express Base Specification), then this bit shall be cleared to ‘0’.

If the AE is associated with an NVMe-MI port, then this bit shall be set to ‘1’.</td></tr><tr><td>15:00</td><td>AE Occurrence Port ID (AEOPID): If the AEOPT field is cleared to ‘0’, then this field shall contain the Port Identifier of the NVM Subsystem port (refer to the NVM Express Base Specification) associated with the AE.

If the AEOPT field is set to ‘1’, then:

a) the least-significant byte of this field shall contain the Port Identifier of the NVMe-MI port associated with the AE; and
b) the most-significant byte of this field shall be cleared to 0h.</td></tr></table></td></tr>
<tr><td>Endurance Group</td><td>Endurance Group ID Info (EGI): If this AE is Endurance Group scoped (i.e., the value of the AE Occurrence Scope field is 5h), then this field indicates information related to the AE Occurrence Scope ID of the Endurance Group.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>31:16</td><td>Reserved</td></tr><tr><td>15:00</td><td>AE Occurrence Endurance Group ID (AEOEGID): This field shall indicate the Endurance Group ID of the Endurance Group associated with the AE.</td></tr></table></td></tr>
</table>

4.4.6 AE Occurrence Specific Information

The format of the AE Occurrence Specific Info field for each Asynchronous Event Identifier is specified in Figure 65.

Figure 65: AE Occurrence Specific Info Data Structure

<table>
<tr><th>Asynchronous Event ID</th><th>Description</th></tr>
<tr><td>00h
(Ready)</td><td>Ready Info (RI): This field defines the contents of the AE Occurrence Specific Info field for the Ready AE.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7:1</td><td>Reserved</td></tr><tr><td>0</td><td>Ready Value (RV): This bit shall indicate the value of the CSTS.RDY bit (refer to the NVM Express Base Specification).</td></tr></table></td></tr>
<tr><td>01h
(Controller Fatal Status)</td><td>Controller Fatal Status Info (CFSI): This field defines the contents of the AE Occurrence Specific Info field for the Controller Fatal Status AE.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7:1</td><td>Reserved</td></tr><tr><td>0</td><td>Controller Fatal Status Value (CFSV): This bit shall indicate the value of the CSTS.CFS bit (refer to the NVM Express Base Specification).</td></tr></table></td></tr>
<tr><td>02h
(Shutdown Status)</td><td>Shutdown Status Info (SSI): This field defines the contents of the AE Occurrence Specific Info field for the Shutdown Status AE.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7:3</td><td>Reserved</td></tr><tr><td>2</td><td>Shutdown Type Value (STV): This field shall indicate the value of the Shutdown Type bit (refer to the NVM Express Base Specification).</td></tr><tr><td>1:0</td><td>Shutdown Status Value (SSV): This field shall indicate the value of the CSTS.SHST field (refer to the NVM Express Base Specification).</td></tr></table></td></tr>
<tr><td>03h
(Controller Enable)</td><td>Controller Enable Info (CEI): This field defines the contents of the AE Occurrence Specific Info field for the Controller Enable AE.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7:2</td><td>Reserved</td></tr><tr><td>1</td><td>Controller Ready Independent of Media Enable (CRIME): This bit shall indicate the value of the CC.CRIME bit (refer to the NVM Express Base Specification).</td></tr><tr><td>0</td><td>Controller Enable Value (CEV): This bit shall indicate the value of the CC.EN bit (refer to the NVM Express Base Specification).</td></tr></table></td></tr>
<tr><td>04h
(Namespace Attribute Changed)</td><td>There shall be no AE Occurrence Specific Info field defined for this AE.</td></tr>
<tr><td>05h
(Firmware Activated)</td><td>There shall be no AE Occurrence Specific Info field defined for this AE.</td></tr>
<tr><td>06h
(Composite Temperature)</td><td>Composite Temperature Info (CTI): This field defines the contents of the AE Occurrence Specific Info field for the Composite Temperature AE.

<table><tr><th>Bytes</th><th>Description</th></tr><tr><td>0</td><td>Composite Temperature Value (CTV): This field shall indicate the value of the Composite Temperature field in the NVM Subsystem Health data structure.</td></tr></table></td></tr>
<tr><td>07h
(Percentage Drive Life Used)</td><td>Percentage Drive Life Used Info (PUI): This field defines the contents of the AE Occurrence Specific Info field for the Percentage Drive Life Used AE.

<table><tr><th>Bytes</th><th>Description</th></tr><tr><td>0</td><td>Percentage Drive Life Used Value (PUV): This field shall indicate the value of the Percentage Used field in the NVM Subsystem Health data structure.</td></tr></table></td></tr>
</table>

Figure 65: AE Occurrence Specific Info Data Structure

<table>
<tr><th>Asynchronous Event ID</th><th>Description</th></tr>
<tr><td>08h (Available Spare)</td><td>Available Spare Info (ASI): This field defines the contents of the AE Occurrence Specific Info field for the Available Spare AE.

<table><tr><th>Bytes</th><th>Description</th></tr><tr><td>0</td><td>Available Spare Value (ASV): This field shall indicate the value of the Available Spare (refer to the NVM Express Base Specification) field of any Controller in the NVM Subsystem.</td></tr></table></td></tr>
<tr><td>09h (SMART Warnings)</td><td>SMART Warnings Info (CWI): This field defines the contents of the AE Occurrence Specific Info field for the SMART Warnings AE.

<table><tr><th>Bytes</th><th>Description</th></tr><tr><td>0</td><td>SMART Warnings Value (CWV): This field shall indicate the value of the SMART Warnings field in the NVM Subsystem Health data structure.</td></tr></table></td></tr>
<tr><td>0Ah (Telemetry Controller-Initiated Data Available)</td><td>There shall be no AE Occurrence Specific Info field defined for this AE.</td></tr>
<tr><td>0Bh (PCIe Link Active)</td><td>PCIe Link Active Info (PLAI): This field defines the contents of the AE Occurrence Specific Info field for the PCIe Link Active AE.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7:1</td><td>Reserved</td></tr><tr><td>0</td><td>PCIe Link Active Value (PLAV): If the Port Scope ID Info field indicates the AE is associated with PCIe Port 0, then this bit shall indicate the value of the PCIe Port 0 PCIe Link Active bit in the NVM Subsystem Health data structure.

If the Port Scope ID Info field indicates the AE is associated with PCIe Port 1, then this bit shall indicate the value of the PCIe Port 1 PCIe Link Active bit in the NVM Subsystem Health data structure.</td></tr></table></td></tr>
<tr><td>0Ch (Sanitize Failure Mode)</td><td>Sanitize Failure Mode Info (SFMI): This field defines the contents of the AE Occurrence Specific Info field for the Sanitize Failure Mode AE.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7:1</td><td>Reserved</td></tr><tr><td>0</td><td>Sanitize Failure Mode Value (SFMV): This bit shall indicate the value of the Sanitize Failure Mode bit in the NVM Subsystem Health data structure.</td></tr></table></td></tr>
<tr><td>0Dh (Sanitize Namespace Failure Mode)</td><td>Sanitize Namespace Failure Mode Info (SNFMI): This field defines the contents of the AE Occurrence Specific Info field for the Sanitize Namespace Failure Mode AE.

<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7:1</td><td>Reserved</td></tr><tr><td>0</td><td>Sanitize Namespace Failure Mode Value (SNFMV): This bit shall indicate the value of the Sanitize Namespace Failure Mode bit in the NVM Subsystem Health data structure.</td></tr></table></td></tr>
</table>

Figure 65: AE Occurrence Specific Info Data Structure

<table>
<tr><th>Asynchronous Event ID</th><th>Description</th></tr>
<tr><td>0Eh (Power Threshold Exceeded)</td><td>
Power Threshold Exceeded Info (PTEI): This field defines the contents of the AE Occurrence Specific Info field for the Power Threshold Exceeded AE.

This field contains the interval power measurement (i.e., average of power measurement samples over one second) of the indicated power measurement type (i.e., PMT field) associated with this event (refer to the NVM Express Base Specification). The power in Watts is equal to the value in the Power Value field multiplied by the scale indicated in the Power Scale field.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:24</td><td>Reserved</td></tr>
<tr><td>23:20</td><td>Power Measurement Type (PMT): This field contains the power measurement type associated with this event.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0h</td><td>NVM Subsystem total power</td></tr>
<tr><td>1h to Bh</td><td>Reserved</td></tr>
<tr><td>Ch to Fh</td><td>Vendor Specific</td></tr>
</table></td></tr>
<tr><td>19:18</td><td>Reserved</td></tr>
<tr><td>17:16</td><td>Power Scale (PWRS): This field contains the scale for Power Threshold Exceeded Info field.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>Reserved</td></tr>
<tr><td>01b</td><td>0.0001 W</td></tr>
<tr><td>10b</td><td>0.01 W</td></tr>
<tr><td>11b</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>15:00</td><td>Power Value (PWRV): This field contains the value for the Power Threshold Exceeded Info field.</td></tr>
</table>
</td></tr>
<tr><td>0Fh to BFh</td><td>Reserved</td></tr>
<tr><td>C0h to FFh</td><td>There shall be no AE Occurrence Specific Info defined for these AEs.</td></tr>
</table>

5 Management Interface Command Set

The Management Interface Command Set defines the Command Messages that may be submitted by a Requester when the NMIMT value is set to NVMe-MI Command. The Management Interface Command Set is applicable to both the out-of-band mechanism and the in-band tunneling mechanism. The processing of commands in the Management Interface Command Set may be affected by the Command and Feature Lockdown feature (refer to the NVM Express Base Specification).

The servicing of any NVMe-MI Command over the out-of-band mechanism shall be independent of and not affected by any one or more Controllers in the NVM Subsystem being disabled or being reset by a Controller Level Reset unless the Management Endpoint servicing the NVMe-MI Command is reset (e.g., due to an NVM Subsystem Reset or due to a PCIe Reset of the PCIe VDM Management Endpoint servicing the NVMe-MI Command). Refer to section 8.1 for more details.

The NVMe-MI Message data structure with fields that are common to all NVMe-MI Messages is defined in section 3.1. The Message Body for the Management Interface Command Set is shown in Figure 66 and Figure 67. Command specific fields for the Management Interface Command Set are defined in this section. The Response Message structure for the Management Interface Command Set is defined in section 4.1.2.

Figure 66: NVMe-MI Command Request Message Format

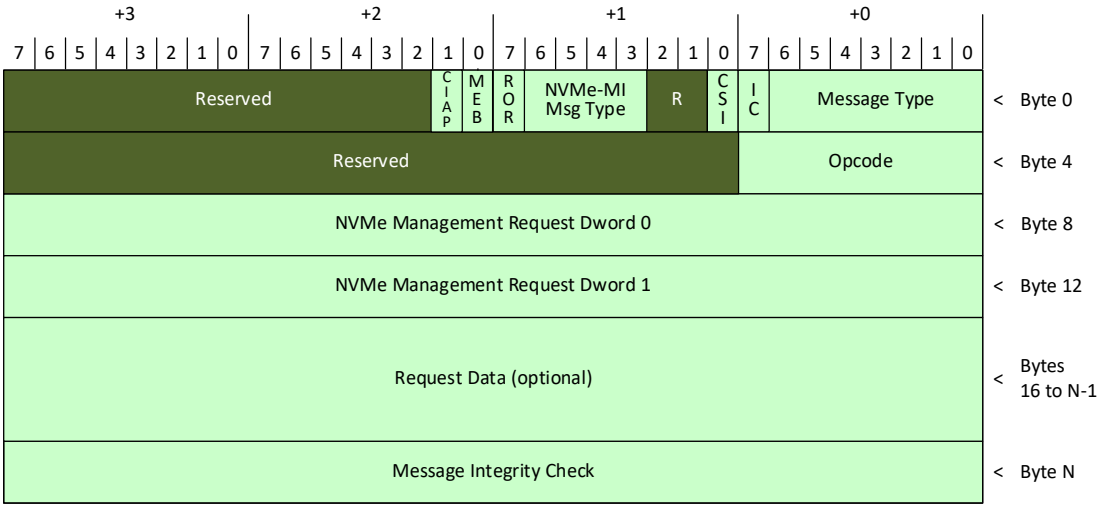


Figure 67: NVMe-MI Command Request Message Description (NCREQ)

Bytes	Description
03:00	**NVMe-MI Message Header (NMH):** Refer to section 3.1.
04	**Opcode (OPC):** This field specifies the opcode of the NVMe-MI Command to be processed. Refer to Figure 68.
07:05	Reserved
11:08	**NVMe Management Dword 0 (NMD0):** This field is command specific Dword 0.
15:12	**NVMe Management Dword 1 (NMD1):** This field is command specific Dword 1.
N-1:16	**Request Data (REQD):** This field is optional.
N+3:N	**Message Integrity Check (MIC):** Refer to section 3.1.

The Request Data field is an optional field included in some NVMe-MI Commands. If the size of the Request Data does not match the specified Data Length of the Command Message, then the Responder responds with a Generic Error Response and Invalid Command Input Data Size status.

Figure 68 defines the Management Interface Command Set opcodes.

Figure 68: Opcodes for Management Interface Command Set

Opcode	Command
00h	Read NVMe-MI Data Structure
01h	NVM Subsystem Health Status Poll
02h	Controller Health Status Poll
03h	Configuration Set
04h	Configuration Get
05h	VPD Read
06h	VPD Write
07h	Reset
08h	SES Receive
09h	SES Send
0Ah	Management Endpoint Buffer Read
0Bh	Management Endpoint Buffer Write
0Ch	Shutdown
0Dh to BFh	Reserved
C0h to FFh	Vendor specific

Figure 69 shows the Management Interface Command Set commands that are mandatory, optional, and prohibited for an NVMe Storage Device and for an NVMe Enclosure using the out-of-band mechanism. Figure 70 shows Management Interface Command Set commands that are mandatory, optional, and prohibited for an NVMe Storage Device and for an NVMe Enclosure using the in-band tunneling mechanism.

Figure 69: Management Interface Command Set Support using an Out-of-Band Mechanism

NVMe Storage Device O/M/P[1]	NVMe Enclosure O/M/P[1]	Command
M	M	Read NVMe-MI Data Structure
M	O[3]	NVM Subsystem Health Status Poll
M	O[3]	Controller Health Status Poll
M	M[2]	Configuration Set
M	M[2]	Configuration Get
M	O[3]	VPD Read
O	O[3]	VPD Write
O	O[3]	Reset
P	M	SES Receive
P	M	SES Send
O	O[3]	Shutdown
O	M	Management Endpoint Buffer Read
O	M	Management Endpoint Buffer Write

Figure 69: Management Interface Command Set Support using an Out-of-Band Mechanism

NVMe Storage Device O/M/P[1]	NVMe Enclosure O/M/P[1]	Command
O	O	Vendor specific

Notes:
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited from being supported. An NVMe Enclosure that is also an NVMe Storage Device (i.e., implements Namespaces):
   - shall implement mandatory commands required for an NVMe Enclosure and may implement optional commands allowed for an NVMe Enclosure; and
   - shall implement mandatory commands required for an NVMe Storage Device and may implement optional commands allowed for an NVMe Storage Device.
2. This command was architected for an NVMe Storage Device. The mapping of Health Status Change Configuration Identifier to an NVMe Enclosure is outside the scope of this specification.
3. This command was architected for an NVMe Storage Device. The mapping of this command to an NVMe Enclosure is outside the scope of this specification.

Figure 70: Management Interface Command Set Support using In-Band Tunneling Mechanism

NVMe Storage Device		NVMe Enclosure		Command
O/M/P[1]	NVMe-MI Send/Receive Mapping[3]	O/M/P[1]	NVMe-MI Send/Receive Mapping[3]	
O	NVMe-MI Receive	O[2]	NVMe-MI Receive	Read NVMe-MI Data Structure
M	NVMe-MI Receive	O[2]	NVMe-MI Receive	NVM Subsystem Health Status Poll
M	NVMe-MI Receive	O[2]	NVMe-MI Receive	Controller Health Status Poll
O	NVMe-MI Send	O[2]	NVMe-MI Send	Configuration Set
O	NVMe-MI Receive	O[2]	NVMe-MI Receive	Configuration Get
M	NVMe-MI Receive	O[2]	NVMe-MI Receive	VPD Read
O	NVMe-MI Send	O[2]	NVMe-MI Send	VPD Write
O	NVMe-MI Send	O[2]	NVMe-MI Send	Reset
P	n/a	M	NVMe-MI Receive	SES Receive
P	n/a	M	NVMe-MI Send	SES Send
P	n/a	P	n/a	Management Endpoint Buffer Read
P	n/a	P	n/a	Management Endpoint Buffer Write
O	NVMe-MI Send	O[2]	NVMe-MI Send	Shutdown

Figure 70: Management Interface Command Set Support using In-Band Tunneling Mechanism

NVMe Storage Device		NVMe Enclosure		Command
O/M/P[1]	NVMe-MI Send/Receive Mapping[3]	O/M/P[1]	NVMe-MI Send/Receive Mapping[3]	
O	Vendor Specific	O	Vendor Specific	Vendor specific

Notes:
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited from being supported. An NVMe Enclosure that is also an NVMe Storage Device (i.e., implements Namespaces):
   - shall implement mandatory commands required for an NVMe Enclosure and may implement optional commands allowed for an NVMe Enclosure; and
   - shall implement mandatory commands required for an NVMe Storage Device and may implement optional commands allowed for an NVMe Storage Device.
2. This command was architected for an NVMe Storage Device. The mapping of this command to an NVMe Enclosure is outside the scope of this specification.
3. This column indicates whether the NVMe-MI Command is tunneled in-band using the NVMe-MI Send or NVMe-MI Receive command.

Figure 71: NVMe-MI Command Response Message Format

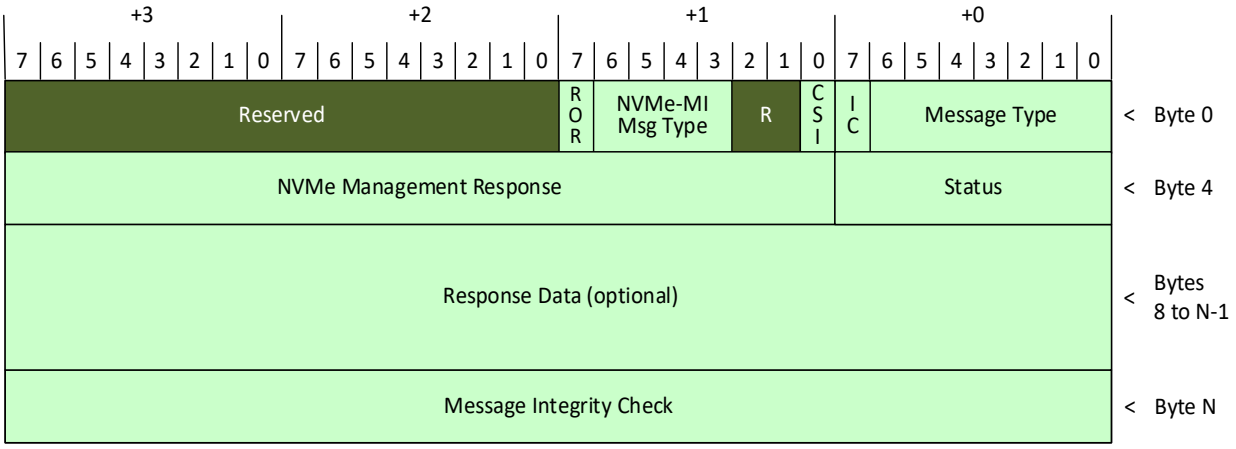


Figure 72: NVMe-MI Command Response Message Description (NCRESP)

Bytes	Description
03:00	**NVMe-MI Message Header (NMH):** Refer to section 3.1.
04	**Status (STATUS):** This field indicates the status of the NVMe-MI Command. Refer to section 4.1.2.
07:05	**NVMe Management Response (NMRESP):** This field is command specific.
N-1:08	**Response Data (RESPD):** This field is optional.
N+3:N	**Message Integrity Check (MIC):** Refer to section 3.1.

5.1 Configuration Get

The Configuration Get command allows the Requester to read the current configuration of a Responder.

The command uses the NVMe Management Dword 0 field (refer to Figure 73) and the NVMe Management Dword 1 field (refer to Figure 74). There is no Request Data included in a Configuration Get command.

Figure 73: Configuration Get – NVMe Management Dword 0

Bits	Description
31:08	**Configuration Identifier Specific (CIS):** The content of this field is based on the Configuration Identifier value.
07:00	**Configuration Identifier (CID):** This field specifies the identifier of the Configuration that is being read. Refer to Figure 75.

Figure 74: Configuration Get – NVMe Management Dword 1

Bits	Description
31:00	**Configuration Identifier Specific (CIS):** The content of this field is based on the Configuration Identifier value.

NVMe-MI Configuration Identifiers are listed in Figure 75.

Figure 75: NVMe Management Interface Configuration Identifiers

Configuration Identifier	Out-of-Band Mechanism O/M/P[1]	In-Band Tunneling Mechanism O/M/P[1]	Description
00h	-	-	Reserved
01h	M	P	SMBus/I2C Frequency
02h	M	M	Health Status Change
03h	M	P	MCTP Transmission Unit Size
04h	O[2]	P	Asynchronous Event
05h to BFh	-	-	Reserved
C0h to FFh	O	O	Vendor Specific

Notes:
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited from being supported.
2. This configuration is optional for both PCIe VDM Management Endpoints and 2-Wire Management Endpoints; however, the specifics of how this configuration works for 2-Wire Management Endpoints is outside the scope of this specification.

The NVMe Management Response field is configuration specific.

5.1.1 SMBus/I2C Frequency (Configuration Identifier 01h)

The SMBus/I2C Frequency configuration indicates the current frequency of each Management Endpoint on the SMBus port, if applicable. If the 2-Wire port is not in SMBus mode, then the indicated value is undefined.

The configuration specific fields in the NVMe Management Dword 0 field are shown in Figure 76. The configuration specific fields in the NVMe Management Dword 1 field are reserved. The current 2-Wire Frequency configuration is returned in the NVMe Management Response field as shown in Figure 77.

Figure 76: 2-Wire Frequency – NVMe Management Dword 0

Bits	Description
31:24	**Port Identifier (PORTID):** This field specifies the port whose 2-Wire Frequency is indicated.
23:08	Reserved

Figure 76: 2-Wire Frequency – NVMe Management Dword 0

Bits	Description
07:00	**Configuration Identifier (CID):** This field specifies the identifier of the Configuration that is being read. Refer to Figure 75.

Figure 77: 2-Wire Frequency – NVMe Management Response

Bits	Description
23:04	Reserved
03:00	**SMBus/I2C Frequency (SFREQ):** This field shall indicate the frequency that the Management Endpoint transmissions are clocked at while the 2-Wire port is in SMBus mode. A Management Endpoint Reset (refer to section 8.3.3) shall set this field to 1h. **Value** / **Description** 0h: Obsolete. This value is obsolete for implementations compliant with version 2.0 and later of this specification. Refer to version 1.2 of this specification for the previous definition. 1h: 100 kHz 2h: 400 kHz 3h: 1 MHz 4h to Fh: Reserved

5.1.2 Health Status Change (Configuration Identifier 02h)

The Health Status Change configuration is used to clear the selected status bits in the Composite Controller Status Flags field using a Configuration Set command. A Requester should not use a Configuration Get command for this Configuration Identifier.

The configuration specific fields in the NVMe Management Dword 0 field and the NVMe Management Dword 1 field are reserved. A Responder shall complete a Configuration Get command on this Configuration Identifier with a Success Response. The NVMe Management Response field is reserved and there is no Response Data.

5.1.3 MCTP Transmission Unit Size (Configuration Identifier 03h)

The MCTP Transmission Unit Size configuration indicates the current MCTP Transmission Unit Size of each Management Endpoint on the port corresponding to the Port Identifier specified in the NVMe Management Dword 0 field. If the 2-Wire port is in I3C mode, then the value indicated is determined with the SETMRL CCC mechanism defined by the MCTP I3C Transport Binding Specification.

The configuration specific fields in the NVMe Management Dword 0 field are shown in Figure 78. The configuration specific fields in the NVMe Management Dword 1 field are reserved. The current Transmission unit size of the specified port is returned in the NVMe Management Response field as shown in Figure 79.

Figure 78: MCTP Transmission Unit Size – NVMe Management Dword 0

Bits	Description
31:24	**Port Identifier (PORTID):** This field specifies the port whose MCTP Transmission Unit Size is indicated.
23:08	Reserved
07:00	**Configuration Identifier (CID):** This field specifies the identifier of the Configuration that is being read. Refer to Figure 75.

Figure 79: MCTP Transmission Unit Size – NVMe Management Response

Bits	Description
23:16	Reserved
15:00	**MCTP Transmission Unit Size (MTUS):** This field contains the MCTP Transmission Unit Size in bytes to be used by each Management Endpoint on the port. A Management Endpoint Reset (refer to section 8.3.3) shall cause this field to be set to 64.

5.1.4 Asynchronous Event (Configuration Identifier 04h)

The Asynchronous Event configuration indicates information about AEs for the Management Endpoint that processes the Configuration Get command.

The configuration-specific fields in the NVMe Management Dword 0 field are shown in Figure 80. The configuration-specific fields in the NVMe Management Dword 1 field are reserved.

Figure 80: Asynchronous Event – NVMe Management Dword 0

Bits	Description
31:08	Reserved
07:00	**Configuration Identifier (CID):** This field specifies the identifier of the Configuration that is being read. Refer to Figure 75.

Upon successful completion of the Configuration Get command, the data in Figure 81 shall be returned in the NVMe Management Response field and the data structure in Figure 82 shall be returned in the Response Data.

Figure 81: Asynchronous Event – NVMe Management Response

Bits	Description
23:08	Reserved
07:00	**AE Enable List Version Number (AEELVER):** This field shall indicate the version number of the AE Enable List data structure and the AE Enable data structure supported by the Management Endpoint. This field shall be cleared to 0h.

The AE Supported List data structure indicates a list of AEs that the Management Endpoint supports and shall be minimally sized (i.e., if there is one AE Supported data structure, then the length of the AE Supported List data structure is equal to the value of the AESL field plus the value of the AESLHL field). The AE Supported List data structure shall start at offset 0h of the Response Data field. The length of the AE Supported List Body shall be less than or equal to 4 KiB.

Figure 82: AE Supported List Data Structure

Bytes	Description
	AE Supported List Header
0	**Number of AE Supported Data Structures (NUMAES):** This field shall indicate the number of AE Supported data structures (refer to Figure 83) in the AE Supported List Body. This field shall be set to a value that is greater than or equal to 1h.
1	**AE Supported List Version Number (AESLVER):** This field shall indicate the version number of the AE Supported List data structure and the AE Supported data structure. This field shall be cleared to 0h.
3:2	**AE Supported Total Length (AESTL):** This field indicates the length in bytes of the AE Supported List data structure. This field shall be set to a value equal to the value of the AE Supported List Header Length field plus the sum of the lengths in bytes of each AE Supported data structure in the AE Supported List Body.

Figure 82: AE Supported List Data Structure

4	**AE Supported List Header Length (AESLHL):** This field shall indicate the length in bytes of the AE Supported List Header. This field shall be set to 5h.
AE Supported List Body	
AESLHL+(L-1):AESLHL	**AE Supported 0 (AES0):** This field shall indicate the first AE Supported data structure (refer to Figure 83), where L is the length in bytes of this AE Supported data structure.
AESLHL+L+(M-1): AESLHL+L	**AE Supported 1 (AES1):** This field shall indicate the second AE Supported data structure, if any, where L is the length in bytes of the first AE Supported data structure and M is the length in bytes of this AE Supported data structure.
…	
AESTL-1:AESTL-N	**AE Supported N (AESN):** This field shall indicate the last AE Supported data structure, if any, where N is the length in bytes of this AE Supported data structure.

Figure 83: AE Supported Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>0</td><td>AE Supported Length (AESL): This field shall indicate the length in bytes of the AE Supported data structure. This field shall be set to 3h.</td></tr>
<tr><td>2:1</td><td>AE Supported Info (AESI): This field shall indicate information about the asynchronous event.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>15</td><td>AE Supported Enable (AESE): If the AE indicated by the AE Supported ID field is enabled, then this this bit shall be set to ‘1’.

If the AE indicated by the AE Supported ID field is disabled, then this bit shall be cleared to ‘0’.

A Management Endpoint Reset shall clear this bit to ‘0’.</td></tr>
<tr><td>14:08</td><td>Reserved</td></tr>
<tr><td>07:00</td><td>AE Supported ID (AESI): This field shall indicate the identifier of the AE (refer to Figure 63).</td></tr>
</table></td></tr>
</table>

5.2 Configuration Set

The Configuration Set command allows the Requester to modify the current configuration of a Responder.

The command uses the NVMe Management Dword 0 field (refer to Figure 84) and the NVMe Management Dword 1 field (refer to Figure 85). There is no Request Data included in a Configuration Set command.

Figure 84: Configuration Set – NVMe Management Dword 0

Bits	Description
31:08	**Configuration Identifier Specific (CIS):** The content of this field is based on the Configuration Identifier value.
07:00	**Configuration Identifier (CID):** This field specifies the identifier of the Configuration that is being written. Refer to Figure 75.

Figure 85: Configuration Set – NVMe Management Dword 1

Bits	Description
31:00	**Configuration Identifier Specific (CIS):** The content of this field is based on the Configuration Identifier value.

NVMe-MI Configuration Identifiers are listed in Figure 75. Specifying a reserved identifier in the Configuration Identifier field shall cause the command to complete with an Invalid Parameter Error Response with the PEL field indicating the Configuration Identifier field.

The NVMe Management Response field is configuration Identifier specific.

5.2.1 SMBus/I2C Frequency (Configuration Identifier 01h)

The SMBus/I2C Frequency configuration specifies a new frequency for the SMBus port. If the 2-Wire port is in I3C mode, then this command shall complete with a Success Response but have no effect.

The configuration specific fields in the NVMe Management Dword 0 field are shown in Figure 86. The configuration specific fields in the NVMe Management Dword 1 field are reserved. The NVMe Management Response field is reserved.

After successful completion of this command, the SMBus/I2C frequency is updated to the specified frequency. A Management Controller should not change this configuration while there are other Command Messages outstanding.

If the specified frequency is not supported, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the SMBus/I2C Frequency field. If the Port Identifier specified is not a 2-Wire port, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Port Identifier field.

Figure 86: 2-Wire Frequency – NVMe Management Dword 0

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:24</td><td>Port Identifier (PORTID): This field specifies the port whose 2-Wire Frequency is specified.</td></tr>
<tr><td>23:12</td><td>Reserved</td></tr>
<tr><td>11:08</td><td>SMBus/I2C Frequency (SFREQ): This field specifies the new frequency for each Management Endpoint on the specified 2-Wire port.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0h</td><td>Reserved</td></tr>
<tr><td>1h</td><td>100 kHz</td></tr>
<tr><td>2h</td><td>400 kHz</td></tr>
<tr><td>3h</td><td>1 MHz</td></tr>
<tr><td>4h to Fh</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>07:00</td><td>Configuration Identifier (CID): This field specifies the identifier of the Configuration that is being written. Refer to Figure 75.</td></tr>
</table>

5.2.2 Health Status Change (Configuration Identifier 02h)

This Configuration Identifier is used to clear selected status bits in the Composite Controller Status Flags field.

The Composite Controller Status Flags field is used to report the occurrence of health and status events associated with the NVM Subsystem via the Composite Controller Status field in the Response Message for the NVM Subsystem Health Status Poll command. When a bit in this field is set to ‘1’, that bit remains set to ‘1’ until cleared to ‘0’ by a Requester or until cleared to ‘0’ by a reset as described in Figure 107.

A Configuration Set command that selects Health Status Change clears corresponding bits selected in the NVMe Management Dword 1 field of the Composite Controller Status Flags field to ‘0’.

A Configuration Set command that selects Health Status Change operates independently for each Management Endpoint in the out-of-band mechanism and each Controller in the in-band tunneling mechanism.

An NVMe Storage Device or NVMe Enclosure supporting the Health Status Change Configuration Identifier in the out-of-band mechanism shall have an independent instance of the Composite Controller Status Flags field dedicated to each Management Endpoint. In the out-of-band mechanism, a Configuration Set command that selects Health Status Change only applies to the instance of the Composite Controller Status

Flags field dedicated to the Management Endpoint to which the Configuration Set command was issued. Refer to Figure 107 for more details on the Composite Controller Status Flags field.

An NVMe Storage Device or NVMe Enclosure supporting the Health Status Change Configuration Identifier in the in-band tunneling mechanism shall have an independent instance of the Composite Controller Status Flags field dedicated to each Controller. In the in-band tunneling mechanism, a Configuration Set command that selects Health Status Change only applies to the instance of the Composite Controller Status Flags field dedicated to the Controller to which the Configuration Set command was issued.

Figure 87: Health Status Change - NVMe Management Dword 0

Bits	Description
31:08	Reserved
07:00	**Configuration Identifier (CID):** This field specifies the identifier of the Configuration that is being written. Refer to Figure 75.

Figure 88: Health Status Change – NVMe Management Dword 1

Bits	Description
31:13	Reserved
12	**Telemetry Controller-Initiated Data Available (TCIDA):** If this bit is set to '1', then the TCIDA bit (i.e., bit 13) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the TCIDA bit in the Composite Controller Status Flags field shall not be modified.
11	**Critical Warning (CWARN):** If this bit is set to '1', then the CWARN bit (i.e., bit 12) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the CWARN bit in the Composite Controller Status Flags field shall not be modified.
10	**Available Spare (SPARE):** If this bit is set to '1', then the SPARE bit (i.e., bit 11) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the SPARE bit in the Composite Controller Status Flags field shall not be modified.
09	**Percentage Used (PDLU):** If this bit is set to '1', then the PDLU bit (i.e., bit 10) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the PDLU bit in the Composite Controller Status Flags field shall not be modified.
08	**Composite Temperature (CTEMP):** If this bit is set to '1', then the CTEMP bit (i.e., bit 9) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the CTEMP bit in the Composite Controller Status Flags field shall not be modified.
07	**Controller Status Change (CSCHNG):** If this bit is set to '1', then the CSTS bit (i.e., bit 8) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the CSTS bit in the Composite Controller Status Flags field shall not be modified.
06	**Firmware Activated (FA):** If this bit is set to '1', then the FA bit (i.e., bit 7) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the FA bit in the Composite Controller Status Flags field shall not be modified.
05	**Namespace Attribute Changed (NAC):** If this bit is set to '1', then the NAC bit (i.e., bit 6) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the NAC bit in the Composite Controller Status Flags field shall not be modified.
04	**Controller Enable Change Occurred (CECO):** If this bit is set to '1', then the CECO bit (i.e., bit 5) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the CECO bit in the Composite Controller Status Flags field shall not be modified.
03	**NVM Subsystem Reset Occurred (NSSRO):** If this bit is set to '1', then the NSSRO bit (i.e., bit 4) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the NSSRO bit in the Composite Controller Status Flags field shall not be modified.
02	**Shutdown Status (SHST):** If this bit is set to '1', then the SHST bit (i.e., bit 2) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the SHST bit in the Composite Controller Status Flags field shall not be modified.
01	**Controller Fatal Status (CFS):** If this bit is set to '1', then the CFS bit (i.e., bit 1) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the CFS bit in the Composite Controller Status Flags field shall not be modified.

Figure 88: Health Status Change – NVMe Management Dword 1

Bits	Description
00	**Ready (RDY):** If this bit is set to '1', then the RDY bit (i.e., bit 0) in the Composite Controller Status Flags field shall be cleared to '0'. If this bit is cleared to '0', then the RDY bit in the Composite Controller Status Flags field shall not be modified.

5.2.3 MCTP Transmission Unit Size (Configuration Identifier 03h)

The MCTP Transmission Unit Size configuration specifies a new MCTP Transmission Unit Size for each Management Endpoint on the port corresponding to the specified Port Identifier, if applicable. This configuration is not applicable for the 2-Wire port while the port is in I3C mode. If targeting a 2-Wire port that is in I3C mode, then:

- this command shall complete successfully but have no effect; and
- the SETMWL and SETMRL CCCs are used to change the MCTP Transmission Unit Size as defined by the MCTP I3C Transport Binding Specification.

A Management Controller should check the maximum MCTP Transmission Unit Size for the port reported by the Management Endpoint using the Read NVMe-MI Data Structure command (refer to Figure 114).

The configuration specific fields in the NVMe Management Dword 0 field (refer to Figure 89) and the NVMe Management Dword 1 field (refer to Figure 90). The NVMe Management Response field is reserved.

After successful completion of this command, the MCTP Transmission Unit Size for MCTP packets on the specified port is updated to the specified size for future Command Messages. A Management Controller should not change this configuration while there are other commands outstanding. Changing this configuration while there are other Request Messages outstanding results in undefined behavior. If a Request Message is sent with a given MCTP Transmission Unit Size, then issuing a Replay Control Primitive after changing the MCTP Transmission Unit Size to a different value results in undefined behavior.

If the specified MCTP Transmission Unit Size is not supported, then the Management Endpoint shall abort the command and send a Response Message with an Invalid Parameter Error Response with the PEL field indicating the MCTP Transmission Unit Size field. If the Port Identifier specified is not valid, then the Management Endpoint shall abort the command and send a Response Message with an Invalid Parameter Error Response with the PEL field indicating the Port Identifier field.

Figure 89: MCTP Transmission Unit Size – NVMe Management Dword 0

Bits	Description
31:24	**Port Identifier (PORTID):** This field specifies the port whose MCTP Transmission Unit Size is specified.
23:08	Reserved
07:00	**Configuration Identifier (CID):** This field specifies the identifier of the Configuration that is being written. Refer to Figure 75.

Figure 90: MCTP Transmission Unit Size – NVMe Management Dword 1

Bits	Description
31:16	Reserved
15:00	**MCTP Transmission Unit Size (MTUS):** This field contains the requested MCTP Transmission Unit Size in bytes to be used by each Management Endpoint on the port.

5.2.4 Asynchronous Event (Configuration Identifier 04h)

The Asynchronous Event (AE) configuration has two variations:

a) an AE Sync; and

b) an AEM Ack.

An AE Sync shall occur when a Configuration Set command for the Asynchronous Event configuration is processed by the Management Endpoint that does not have the Number of AE Enable Data Structures field in the Configuration Set command cleared to 0h and results in the AE Occurrence List Overflow bit cleared to '0' (refer to section 5.2.4.1). The Response Message for an AE Sync includes the current state of all enabled AEs which is used to synchronize the state of the AEs between the Management Controller and the Management Endpoint.

An AEM Ack acknowledges receipt of an AEM to a Management Endpoint. An AEM Ack shall occur when a Configuration Set command for the Asynchronous Event configuration is processed by the Management Endpoint that has the Number of AE Enable Data Structures field in the Configuration Set command cleared to 0h and results in the AE Occurrence List Overflow bit cleared to '0'. An AEM Ack during the AE Disarmed State following one or more AE occurrences in the prior AE Armed State shall cause those AE occurrences to be discarded (i.e., it is not permitted to transmit those AE occurrences again once receipt of the AEM for those AE occurrences has been acknowledged by the Management Controller). The Response Message for an AEM Ack transmitted during the AE Disarmed State when one or more AEs are enabled includes an AE Occurrence data structure for each AE that occurred during the AE Disarmed State which is used to resynchronize the state of the AEs between the Management Controller and the Management Endpoint.

If the length of the AE Occurrence List Body in the Response Data is able to exceed 4 KiB, then the Management Endpoint shall support the use of the Management Endpoint Buffer to retrieve the Response Data. The size of the Management Endpoint Buffer shall be greater than or equal to the maximum possible Response Data size.

The Request Data for a Configuration Set command for the Asynchronous Event configuration is not permitted to be transferred using the MEB. The Management Endpoint shall retrieve the Request Data from the Request Message regardless of whether the MEB bit is set to '1' or cleared to '0'.

The configuration specific fields in the NVMe Management Dword 0 field are shown in Figure 91. The configuration specific fields in the NVMe Management Dword 1 field are reserved.

Figure 91: Asynchronous Event – NVMe Management Dword 0

Bits	Description
31:27	Reserved
26	**Enable SR-IOV Virtual Functions AE (ENVFA):** If this bit is set to '1' in an AE Sync, then Controller-scoped AEs shall be enabled on SR-IOV Virtual Functions. If this bit is cleared to '0' in an AE Sync, then Controller-scoped AEs shall be disabled on SR-IOV Virtual Functions. It shall not be treated as an error if this bit is set to '1' and SR-IOV Virtual Functions do not exist. This bit is not applicable and shall be ignored for an AEM Ack.
25	**Enable SR-IOV Physical Functions AE (ENPFA):** If this bit is set to '1' in an AE Sync, then Controller-scoped AEs shall be enabled on SR-IOV Physical Functions. If this bit is cleared to '0' in an AE Sync, then Controller-scoped AEs shall be disabled on SR-IOV Physical Functions. It shall not be treated as an error if this bit is set to '1' and SR-IOV Physical Functions do not exist. This bit is not applicable and shall be ignored for an AEM Ack.

Figure 91: Asynchronous Event – NVMe Management Dword 0

Bits	Description
24	**Enable PCI Functions AE (ENCFA):** If this bit is set to '1' in an AE Sync, then Controller-scoped AEs shall be enabled on non-SR-IOV PCI Functions. If this bit is cleared to '0' in an AE Sync, then Controller-scoped AEs shall be disabled on non-SR-IOV PCI Functions. It shall not be treated as an error if this bit is set to '1' and non-SR-IOV PCI Functions do not exist. This bit is not applicable and shall be ignored for an AEM Ack.
23:16	**AEM Delay (AEMD):** For an AE Sync, this field specifies the amount of time in seconds the Management Endpoint shall delay after entering the AE Armed State before the Management Endpoint is permitted to enter the AEM Transmission Interval to transmit an AEM for any AEs that occurred during the AE Armed State (refer to section 1.8.11).
15:08	**AEM Retry Delay (AERD):** If this field is not cleared to 0h in an AE Sync, then the Management Endpoint shall wait the amount of time in 100 ms units specified by this field before attempting to retry transmission of an unacknowledged or failed AEM transmission (refer to section 4.4.3). If this field is cleared to 0h in an AE Sync, then the Management Endpoint shall not attempt to retry transmission of any AEM at the MCTP layer. Note that retries may still occur at the physical layer (e.g., due to a NACK on 2-Wire) when this field is cleared to 0h. This field is not applicable and shall be ignored for an AEM Ack.
07:00	**Configuration Identifier (CID):** This field specifies the identifier of the Configuration that is being written. Refer to Figure 75.

The AE Enable List data structure is transferred in the Request Data field and may specify a list of AEs that the Management Endpoint shall configure. The AE Enable List data structure should be minimally sized (i.e., if there is one AE Enable data structure, then the AE Enable List data structure should be the length of that AE Enable data structure plus the length of the AE Enable List Header). If the AE Enable List data structure is not minimally sized, then the Management Endpoint shall ignore the additional data. The AE Enable List data structure should start at offset 0h of the Request Data field. If the AE Enable List Body is greater than 4 KiB, then the Management Endpoint shall respond with a Response Message Status of Invalid Command Input Data Size.

If the Configuration Set command initiates an AE Sync and is processed in the AE Armed State while there are AEs that have occurred but have not been transmitted in an AEM, then those AE occurrences shall be discarded (e.g., those AE occurrences shall not be transmitted during any AEM Transmission Interval or transmitted in the Response Message for an AEM Ack).

If an AE Sync or AEM Ack is occurs during the AEM Transmission Interval, then:

a) if an AEM transmission is in flight at the time the AE Sync or AEM Ack occurs, then the Management Endpoint stops the AEM transmission as defined in section 4.4.3; and
b) any AEs that have occurred in the prior AE Armed State shall be discarded (e.g., those AE occurrences shall not be transmitted during any AEM Transmission Interval or transmitted in the Response Message for an AEM Ack).

Figure 92: AE Enable List Data Structure

Bytes	Description
AE Enable List Header	
0	**Number of AE Enable Data Structures (NUMAEE):** This field specifies the number of AE Enable data structures (refer to Figure 93) in the AE Enable List Body. If there are no AE Enable data structures in the AE Enable List Body, then this field should be cleared to 0h. A value of 0h is used to initiate an AEM Ack (refer to section 1.8.10).

Figure 92: AE Enable List Data Structure

Bytes	Description
AE Enable List Header	
1	**AE Enable List Version Number (AEELVER):** This field specifies the version number of the AE Enable List data structure and the AE Enable data structure. This field should be cleared to 0h. A Management Endpoint designed to support version N of these data structures: a) shall not generate an error for any value in this field; b) shall process these data structures as defined by version N of these data structures regardless of the version of this field; c) shall ignore non-zero values in fields that are reserved in version N of these data structures; and d) shall not perform any functionality related to these data structures that is not defined by version N of these data structures.
3:2	**AE Enable Total Length (AEETL):** This field specifies the length in bytes of the AE Enable List data structure. This field should be set to a value equal to the value of the AE Enable List Header Length field plus the sum of the lengths in bytes of each AE Enable data structure in the AE Enable List Body.
4	**AE Enable List Header Length (AEELHL):** This field specifies the length in bytes of the AE Enable List Header. This field should be set to 5h.
AE Enable List Body	
AEELHL+(L-1):AEELHL	**AE Enable 0 (AEE0):** This field specifies the first AE Enable data structure (refer to Figure 93), if any, where L is the length in bytes of this AE Occurrence data structure.
AEELHL+L+(M-1): AEELHL+L	**AE Enable 1 (AEE1):** This field specifies the second AE Enable data structure, if any, where L is the length in bytes of the first AE Occurrence data structure and M is the length in bytes of this AE Enable data structure.
…	
AEETL-1:AEETL-N	**AE Enable N (AEEN):** This field specifies the last AE Enable data structure, if any, where N is the length in bytes of this AE Enable data structure.

Figure 93: AE Enable Data Structure

Bytes	Description
0	**AE Enable Length (AEEL):** This field specifies the length in bytes of the AE Enable data structure. This field should be set to 3h.
2:1	**AE Enable Info (AEEI):** This field specifies information about the asynchronous event. **Bits** / **Description** 15 — **AE Enable (AEE):** If this bit is set to '1', then the asynchronous event indicated by the AE Enable ID field shall be enabled. If this bit is cleared to '0', then the asynchronous event indicated by the AE Enable ID field shall be disabled. 14:08 — Reserved 07:00 — **AE Enable ID (AEEI):** This field specifies the identifier of the asynchronous event (refer to Figure 63).

5.2.4.1 AE Occurrence List Overflow Handling

If the length of the AE Occurrence List Body:

a) does not exceed 4 KiB; or
b) does exceed 4 KiB and the MEB bit is set to '1',

then the AE Occurrence List Overflow bit shall be cleared to '0'.

If the number of AE Occurrence data structures available to return in the AE Occurrence List Body would result in the length of the AE Occurrence List Body exceeding 4 KiB if the AE Occurrence data structure for each AE available to return was included in the AE Occurrence List Body and the MEB bit is cleared to '0', then the AE Occurrence List Overflow bit shall be set to '1'.

5.2.4.2 Asynchronous Event Response

Upon completion of the Configuration Set command, a Response Message shall be transmitted. The NVMe Management Response field is reserved. The Response Data field shall indicate an AE Occurrence List data structure (refer to Figure 61) and shall be minimally sized (i.e., if there is one AE Occurrence data structure, then the AE Occurrence List data structure is the length of that AE Occurrence data structure plus the length of the AE Occurrence List Header). The AE Occurrence List data structure shall start at offset 0h of the Response Data field.

If this command results in the AE Occurrence List Overflow bit is set to '1', then the AE Occurrence List data structure shall contain the AE Occurrence List Header and shall not contain an AE Occurrence List Body.

If a supported AE is not in the AE Enable List data structure, then that AE's configuration shall not be changed. If an unsupported AE is in the AE Enable List data structure, then the AE Enable data structure for that AE shall be ignored by the Management Endpoint.

For an AEM Ack:

a) if the Configuration Set command is processed when no AEs are enabled or during the AE Armed State, then a Success Response shall be returned; or
b) if the Configuration Set command is processed during the AE Disarmed State when one or more AEs are enabled, then:

- the AE Occurrence List Body shall contain an AE Occurrence data structure (refer to Figure 62) for each AE of a given AE Unique ID that has occurred during that AE Disarmed State;
- if multiple AEs of a given AE Unique ID occurred during that AE Disarmed State, then only the AE Occurrence data structure for the most recent occurrence of the AE associated with that AE Unique ID is included in the AEM;
- the AE Occurrence List Body shall not contain an AE Occurrence data structure for any AEs that did not occur during that AE Disarmed State; and
- each AE Occurrence data structure shall indicate the state of the AE at the time the AE occurred.

For an AE Sync:

a) the AEM Transmission Failure bit is cleared to '0' (refer to Figure 108);
b) the AE Occurrence List data structure shall contain an AE Occurrence data structure for each AE that was enabled by the Configuration Set command or that was already enabled; and
c) each AE Occurrence data structure shall indicate the state of the AE at the time the Configuration Set command was processed.

If this command leaves one or more AEs enabled and results in the AE Occurrence List Overflow bit cleared to '0', then an AE Arm shall occur. If an AE Arm occurs, then the Management Endpoint shall perform the following steps atomically:

a) for each supported AE in the AE Enable List data structure, the AE is enabled or disabled as specified by the AE Enable bit in the AE Enable data structure for the AE;
b) for each AE Occurrence data structure in the AE Occurrence List data structure, populate the current state of the AE in the AE Occurrence Specific Info field, if any, and AE Occurrence Vendor Specific Info field, if any, in the AE Occurrence data structure in the Response Data; and
c) transition the Management Endpoint to the AE Armed State.

5.3 Controller Health Status Poll

The Controller Health Status Poll command is used to efficiently determine changes in health status attributes associated with one or more Controllers in the NVM Subsystem. This command returns a list of zero or more Controller Health data structures based on various selection criteria (refer to section 5.3.1).

The Controller Health Status Poll command operates independently for each Management Endpoint in the out-of-band mechanism and each Controller in the in-band tunneling mechanism.

An NVMe Storage Device or NVMe Enclosure supporting the Controller Health Status Poll command in the out-of-band mechanism shall have an independent instance of both the Controller Health data structure (refer to Figure 97) and the Controller Health Status Changed Flags field (refer to Figure 98) for each Controller in the NVM Subsystem dedicated to each Management Endpoint. In the out-of-band mechanism, a Controller Health Status Poll command only applies to the instance of the Controller Health data structure and the Controller Health Status Changed Flags field dedicated to the Management Endpoint to which the Controller Health Status Poll command was issued.

An NVMe Storage Device or NVMe Enclosure supporting the Controller Health Status Poll command in the in-band tunneling mechanism shall have an independent instance of both the Controller Health data structure and the Controller Health Status Changed Flags field for each Controller in the NVM Subsystem dedicated to each Controller. In the in-band tunneling mechanism, a Controller Health Status Poll command only applies to the instance of the Controller Health data structure and the Controller Health Status Changed Flags field dedicated to the Controller to which the Controller Health Status Poll command was issued.

The Controller Health Status Poll command uses the NVMe Management Dword 0 field (refer to Figure 94) and the NVMe Management Dword 1 field (refer to Figure 95).

Figure 94: Controller Health Status Poll – NVMe Management Dword 0

Bits	Description
31	**Report All (ALL):** If this bit is set to '1', then the error selection bits (i.e., CWARN, SPARE, PDLU, CTEMP, and CSTS in Figure 95) shall be ignored when determining whether to return the Controller Health data structure per the selection criteria in section 5.3.1. If this bit is cleared to '0', then the error selection bits (i.e., CWARN, SPARE, PDLU, CTEMP, and CSTS in Figure 95) shall not be ignored when determining whether to return the Controller Health data structure per the selection criteria in section 5.3.1.
30:27	Reserved
26	**Include SR-IOV Virtual Functions (INCVF):** If this bit is set to '1', then a Controller Health data structure shall be returned for Controllers associated with SR-IOV Virtual Functions (VFs) unless excluded by other selection criteria as described in section 5.3.1. If this bit is cleared to '0', then a Controller Health data structure shall not be returned for Controllers associated with SR-IOV VFs. It is not an error if this bit is set to '1' and SR-IOV Virtual Functions do not exist.
25	**Include SR-IOV Physical Functions (INCPF):** If this bit is set to '1', then a Controller Health data structure shall be returned for Controllers associated with SR-IOV Physical Functions (PFs) unless excluded by other selection criteria as described in section 5.3.1. If this bit is cleared to '0', then a Controller Health data structure shall not be returned for Controllers associated with SR-IOV PFs. It is not an error if this bit is set to '1' and SR-IOV Physical Functions do not exist.
24	**Include PCI Functions (INCF):** If this bit is set to '1', then a Controller Health data structure shall be returned for Controllers associated with non-SR-IOV PCI Functions unless excluded by other selection criteria as described in section 5.3.1. If this bit is cleared to '0', then a Controller Health data structure shall not be returned for Controllers associated with non-SR-IOV PCI Functions. It is not an error if this bit is set to '1' and non-SR-IOV PCI Functions do not exist.

Figure 94: Controller Health Status Poll – NVMe Management Dword 0

Bits	Description
23:16	**Maximum Response Entries (MAXRENT):** This field specifies the maximum number of Controller Health data structure entries that may be returned in the completion. This is a 0's based field. The maximum number of entries is 255. If 256 entries are specified by this field, then an Invalid Parameter Error Response with the PEL field indicating this field shall be returned.
15:00	**Starting Controller ID (SCTLID):** This field specifies the starting Controller ID. If this field specifies a Controller ID that is less than or equal to the maximum Controller ID in the NVM Subsystem, then for each Controller in the NVM Subsystem: • if the Controller ID of the Controller is greater than or equal to the value in this field, then the Controller's Controller Health data structure shall be returned unless excluded by other selection criteria as described in section 5.3.1; or • if the Controller ID of the Controller is less than the value in this field, then the Controller's Controller Health data structure shall not be returned. If this field specifies a Controller ID that is greater than the maximum Controller ID in the NVM Subsystem, then an Invalid Parameter Error Response with the PEL field indicating this field shall be returned.

Figure 95: Controller Health Status Poll – NVMe Management Dword 1

Bits	Description
31	**Clear Changed Flags (CCF):** If this bit is set to '1', then the Management Endpoint shall perform the following steps atomically in the order listed: 1. perform the selection criteria based on the Controller Health Status Changed Flags field as described in 5.3.1.2; 2. for Controllers whose Controller Health data structure is returned, copy the instance of the Controller Health Status Changed Flags field dedicated to the Responder to which the Controller Health Status Poll command was submitted to the corresponding Controller Health Status Changed field in the Controller Health data structure; and 3. for Controllers whose Controller Health data structure is returned, clear each bit in the Controller Health Status Changed Flags field dedicated to the Responder to which the Controller Health Status Poll command was submitted to '0'. If this bit is set to '1', then the following bits in the Controller Status field in Controllers whose Controller Health data structure is returned (refer to Figure 97) shall be cleared to '0': • Namespace Attribute Changed (NAC); • Firmware Activated (FA); and • Telemetry Controller-Initiated Data Available (TCIDA). The Controller Health Status Changed Flags field and the following bits in the Controller Status field in the Controller Health data structure shall not be modified in Controllers whose Controller Health data structure is not returned: • Namespace Attribute Changed (NAC); • Firmware Activated (FA); and • Telemetry Controller-Initiated Data Available (TCIDA). If this bit is cleared to '0', then the Controller Health Status Changed Flags field and the following bits in the Controller Status field in the Controller Health data structure shall not be modified in any Controller: • Namespace Attribute Changed (NAC); • Firmware Activated (FA); and • Telemetry Controller-Initiated Data Available (TCIDA).
30:05	Reserved

Figure 95: Controller Health Status Poll – NVMe Management Dword 1

Bits	Description
04	**Critical Warning (CWARN):** If this bit is set to '1', then a Controller Health data structure shall be returned for Controllers with the Critical Warning bit set to '1' in the instance of that Controller's Controller Health Status Changed Flags field dedicated to the Responder to which the Controller Health Status Poll command was submitted unless excluded by other selection criteria as described in section 5.3.1. If this bit is set to '1', then a Controller Health data structure shall not be returned for Controllers with the Critical Warning bit cleared to '0' in their Controller Health Status Changed Flags field unless included by other selection criteria as described in section 5.3.1. If this bit is cleared to '0', then the Critical Warning bit in the Controller Health Status Changed Flags field shall not be included in the selection criteria described in section 5.3.1.
03	**Available Spare (SPARE):** If this bit is set to '1', then a Controller Health data structure shall be returned for Controllers with the Available Spare bit set to '1' in the instance of that Controller's Controller Health Status Changed Flags field dedicated to the Responder to which the Controller Health Status Poll command was submitted unless excluded by other selection criteria as described in section 5.3.1. If this bit is set to '1', then a Controller Health data structure shall not be returned for Controllers with the Available Spare bit cleared to '0' in their Controller Health Status Changed Flags field unless included by other selection criteria as described in section 5.3.1. If this bit is cleared to '0', then the Available Spare bit in the Controller Health Status Changed Flags field shall not be included in the selection criteria described in section 5.3.1.
02	**Percentage Used (PDLU):** If this bit is set to '1', then a Controller Health data structure shall be returned for Controllers with the Percent Used bit set to '1' in the instance of that Controller's Controller Health Status Changed Flags field dedicated to the Responder to which the Controller Health Status Poll command was submitted unless excluded by other selection criteria as described in section 5.3.1. If this bit is set to '1', then a Controller Health data structure shall not be returned for Controllers with the Percent Used bit cleared to '0' in their Controller Health Status Changed Flags field unless included by other selection criteria as described in section 5.3.1. If this bit is cleared to '0', then the Percent Used bit in the Controller Health Status Changed Flags field shall not be included in the selection criteria described in section 5.3.1.
01	**Composite Temperature Changes (CTEMP):** If this bit is set to '1', then a Controller Health data structure shall be returned for Controllers with the Composite Temperature bit set to '1' in the instance of that Controller's Controller Health Status Changed Flags field dedicated to the Responder to which the Controller Health Status Poll command was submitted unless excluded by other selection criteria as described in section 5.3.1. If this bit is set to '1', then a Controller Health data structure shall not be returned for Controllers with the Composite Temperature bit cleared to '0' in their Controller Health Status Changed Flags field unless included by other selection criteria as described in section 5.3.1. If this bit is cleared to '0', then the Composite Temperature bit in the Controller Health Status Changed Flags field shall not be included in the selection criteria described in section 5.3.1.
00	**Controller Status Changes (CSTS):** If this bit is set to '1', then a Controller Health data structure shall be returned for Controllers with the Controller Status Change bit set to '1' in the instance of that Controller's Controller Health Status Changed Flags field dedicated to the Responder to which the Controller Health Status Poll command was submitted unless excluded by other selection criteria as described in section 5.3.1. If this bit is set to '1', then a Controller Health data structure shall not be returned for Controllers with the Controller Status Change bit cleared to '0' in their Controller Health Status Changed Flags field unless included by other selection criteria as described in section 5.3.1. If this bit is cleared to '0', then the Controller Status Change bit in the Controller Health Status Changed Flags field shall not be included in the selection criteria described in section 5.3.1.

The Controller Health Status Poll Response Messages use the NVMe Management Response field with the format shown in Figure 96.

The Response Data field size may vary based on the number of Controllers selected using the selection criteria described in section 5.3.1. The Response Entries field indicates the number of Controller Health data structures that are contained in the Response Data.

Figure 96: Controller Health Status Poll – NVMe Management Response

Bits	Description
23:16	**Response Entries (RENT):** This field specifies the number of Controller Health data structure entries present in the Response Data for this Response Message. This is a 1-based field.
15:00	Reserved

The Controller Health data structure, shown in Figure 97, contains the health status attributes that shall be tracked for each Controller. Up to 255 Controller Health data structures for Controllers with Controller IDs greater than or equal to the Starting Controller ID (SCTLID) shall be returned starting at offset 0 of the Response Data with no padding bytes between consecutive Controller Health data structures. Controller Health data structures shall be returned in ascending order based on their Controller IDs (i.e., the Controller Health data structure for the Controller with the lowest Controller ID that matches the section criteria in section 5.3.1 shall start at offset 0, the Controller Health data structure with the second lowest Controller ID that matches the section criteria in section 5.3.1 shall start at offset 16, etc.). The Response Data size shall be equal to the number of Controller Health data structures returned multiplied by the size of the Controller Health data structure.

Figure 97: Controller Health Data Structure (CHDS)

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>01:00</td><td>Controller Identifier (CTLID): This field shall indicate the Controller Identifier of the Controller with which the data contained in this data structure is associated.</td></tr>
<tr><td>03:02</td><td>Controller Status (CSTS): This field reports the Controller status.
<table>
<tr><th>Bits</th><th>Reset [1]</th><th>Description</th></tr>
<tr><td>15:09</td><td>0</td><td>Reserved</td></tr>
<tr><td>08</td><td>HwInit</td><td>Telemetry Controller-Initiated Data Available (TCIDA): If the Telemetry Controller-Initiated Data Available field in the Telemetry Controller-Initiated log page (refer to the NVM Express Base Specification) transitions from 0h to 1h in this Controller, then this bit shall be set to ‘1’.

If the Telemetry Controller-Initiated log page is Controller in scope (refer to the Telemetry Controller-Initiated Scope (TCS) field in the NVM Express Base Specification), then:

• if this bit is set to ‘1’, then the Telemetry Controller-Initiated log page shall contain saved internal state (i.e., one or more of the Telemetry Controller-Initiated data areas shall contain valid internal data) available by issuing a Get Log Page command using the out-of-band mechanism; and
• if this bit is cleared to ‘0’, then the Telemetry Controller-Initiated log page returned in response to issuing a Get Log Page command using the out-of-band mechanism shall not contain saved internal state (i.e., Telemetry Controller-Initiated Data Area 1, Telemetry Controller-Initiated Data Area 2, Telemetry Controller-Initiated Data Area 3 and Telemetry Controller-Initiated Data Area 4 are not present).

If the Telemetry Controller-Initiated log page is NVM Subsystem in scope, then refer to the Telemetry Controller-Initiated Data Available bit in the Composite Controller Status Data Structure (refer to Figure 107) to determine the availability of the Telemetry Controller-Initiated log page.

If a Controller Health Status Poll command is processed with the Clear Changed Flags bit set to ‘1’, then this bit shall be cleared to ‘0’.</td></tr>
</table></td></tr>
</table>

Figure 97: Controller Health Data Structure (CHDS)

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td></td><td>
<table>
<tr><td></td><td></td><td>The value of this bit shall persist across all resets and power cycles.</td></tr>
<tr><td>07</td><td>HwInit</td><td>Firmware Activated (FA): If a new firmware image is activated, then this bit shall be set to ‘1’. Firmware activation is described in the NVM Express Base Specification.

If a reset caused a new firmware image to be activated, then the reset value of this bit shall be ‘1’.

If a Controller Health Status Poll command is processed with the Clear Changed Flags bit set to ‘1’, then this bit shall be cleared to ‘0’.</td></tr>
<tr><td>06</td><td>0</td><td>Namespace Attribute Changed (NAC): This bit shall be set to ‘1’ due to any condition that is capable of causing the Allocated Namespace Attribute Changed asynchronous event (e.g., exclusions to sending the Allocated Namespace Attribute Changed asynchronous event for specific Controllers do not prevent this bit from being set to ‘1’). Allocated Namespace Attribute Notices are not required to be enabled and no Asynchronous Event Request commands are required to be outstanding in order for this bit to be set to ‘1’ (refer to the NVM Express Base Specification).

If a Controller Health Status Poll command is processed with the Clear Changed Flags bit set to ‘1’, then this bit shall be cleared to ‘0’.</td></tr>
<tr><td>05</td><td>0</td><td>Controller Enable Change Occurred (CECO): This bit shall indicate the value of the Enable bit (refer to CC.EN in the NVM Express Base Specification).

Note that the name of this bit does not match the functionality, but the original name of this bit has been retained for historical continuity. Refer to version 1.2 of this specification for the original definition of this bit.</td></tr>
<tr><td>04</td><td>HwInit</td><td>NVM Subsystem Reset Occurred (NSSRO): If an NVM Subsystem Reset occurs due to any reason other than application of main power and does not cause activation of a new firmware image, then this bit shall be set to ‘1’.

If an NVM Subsystem Reset occurs due to application of main power or causes activation of a new firmware image, then this bit shall be cleared to ‘0’.</td></tr>
<tr><td>03:02</td><td>00b</td><td>Shutdown Status (SHST): This field shall indicate the value of the Shutdown Status field (refer to CSTS.SHST in the NVM Express Base Specification).</td></tr>
<tr><td>01</td><td>HwInit</td><td>Controller Fatal Status (CFS): This bit shall indicate the value of the Controller Fatal Status bit (refer to CSTS.CFS in the NVM Express Base Specification).</td></tr>
<tr><td>00</td><td>0</td><td>Ready (RDY): This bit shall indicate the value of the Ready bit (refer to CSTS.RDY in the NVM Express Base Specification).</td></tr>
</table>
</td></tr>
<tr><td>05:04</td><td>Composite Temperature (CTEMP): This field indicates a value corresponding to a temperature in Kelvins that represents the current composite temperature of the Controller and Namespace(s) associated with that Controller. The value of this field shall indicate the value of the Composite Temperature field in the Controller’s SMART / Health Information log page.</td></tr>
<tr><td>06</td><td>Percentage Used (PDLU): This field indicates an implementation-specific estimate of the percentage of NVM Subsystem life used based on the actual usage and the manufacturer’s prediction of NVM Subsystem life. The value of this field shall indicate the value of the Percent Used field in the Controller’s SMART / Health Information log page.</td></tr>
<tr><td>07</td><td>Available Spare (SPARE): This field indicates a normalized percentage (0% to 100%) of the remaining spare capacity available. The value of this field shall indicate the value of the Available Spare field in the Controller’s SMART / Health Information log page.</td></tr>
</table>

Figure 97: Controller Health Data Structure (CHDS)

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>08</td><td>Critical Warning (CWARN): This field indicates critical warnings for the Controller. The value of this field shall indicate the value of the Critical Warning field in the Controller's SMART / Health Information log page.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Reserved</td></tr>
<tr><td>6</td><td>Indeterminate Personality State (IPS): This bit shall indicate the same value as the Indeterminate Personality State (IPS) bit (i.e., bit 6) in the Critical Warning field in the Controller's SMART / Health Information log page.</td></tr>
<tr><td>5</td><td>Persistent Memory Region Error (PMRE): This bit shall indicate the same value as the Persistent Memory Region Read-Only (PMRRO) bit (i.e., bit 5) in the Critical Warning field in the Controller's SMART / Health Information log page.</td></tr>
<tr><td>4</td><td>Volatile Memory Backup Failed (VMBF): This bit shall indicate the same value as the Volatile Memory Backup Failed (VMBF) bit (i.e., bit 4) in the Critical Warning field in the Controller's SMART / Health Information log page.</td></tr>
<tr><td>3</td><td>Read Only (RO): This bit shall indicate the same value as the All Media Read-Only (AMRO) bit (i.e., bit 3) in the Critical Warning field in the Controller's SMART / Health Information log page.</td></tr>
<tr><td>2</td><td>Reliability Degraded (RD): This bit shall indicate the same value as the NVM Subsystem Degraded Reliability (NDR) bit (i.e., bit 2) in the Critical Warning field in the Controller's SMART / Health Information log page.</td></tr>
<tr><td>1</td><td>Temperature Above or Under Threshold (TAUT): This bit shall indicate the same value as the Temperature Threshold Condition (TTC) bit (i.e., bit 1) in the Critical Warning field in the Controller's SMART / Health Information log page.</td></tr>
<tr><td>0</td><td>Spare Threshold (ST): This bit shall indicate the same value as the Available Spare Capacity Below Threshold (ASCBT) bit (i.e., bit 0) in the Critical Warning field in the Controller's SMART / Health Information log page.</td></tr>
</table></td></tr>
<tr><td>10:09</td><td>Controller Health Status Changed (CHSC): This field shall indicate the value of the Controller Health Status Changed Flags field (refer to Figure 98).</td></tr>
<tr><td>15:11</td><td>Reserved</td></tr>
<tr><td colspan="2">Notes:
1. An NVM Subsystem Reset shall reset the instance of the Controller Status field dedicated to each Management Endpoint in the NVM Subsystem and the instance of the Controller Status field dedicated to each Controller in the NVM Subsystem.

The instance of the Controller Status field dedicated to a Controller shall be reset by a Controller Level Reset (refer to the NVM Express Base Specification) of that Controller. Note that a Controller Level Reset may affect the Controller Status field in the out-of-band mechanism (e.g., a Controller Level Reset causes the CECO bit in the instance of the Controller Status bits dedicated to the out-of-band mechanism to be set to '1').

The instance of the Controller Status field dedicated to a Management Endpoint shall be reset by a Management Endpoint Reset of that Management Endpoint.

No instance of the Controller Status field shall be reset by any other resets other than the resets documented by this note.</td></tr>
</table>

Associated with each Controller in the NVM Subsystem is a set of the Controller Health Status Changed Flags field shown in Figure 98. The Controller Health Status Changed Flags field shall be set as described in Figure 98 if the corresponding field/bit in the Controller Health data structure changes. Figure 99 shows a graphical representation of which field(s)/bit(s) in the Controller Health data structure shall be associated with each bit in the Controller Health Status Changed Flags field. If a bit in the Controller Health Status Changed Flags field for any Controller transitions from '0' to '1', then the corresponding bit in the Composite Controller Status Flags field shall also be set to '1'. The Controller Health Status Changed Flags field shall be cleared to 0h after the Controller selection criteria has been evaluated as described in section 5.3.1 in any Controller whose Controller Health data structure is returned in the Success Response to a Controller Health Status Poll Command Message with the Clear Changed Flags bit set to '1'.

Figure 98: Controller Health Status Changed Flags (CHSCF)

Bits	Reset[1]	Description
15:14	0	Reserved
13	HwInit	**Telemetry Controller-Initiated Data Available (TCIDA):** If the Telemetry Controller-Initiated Data Available bit in the Controller Health data structure transitions from '0' to '1', then this bit shall be set to '1'. The value of this field shall persist across all resets and power cycles.
12	0	**Critical Warning (CWARN):** If the Critical Warning field in the Controller Health data structure changes state, then this bit shall be set to '1'.
11	0	**Available Spare (SPARE):** If the Available Spare field in the Controller Health data structure changes state, then this bit shall be set to '1'.
10	0	**Percentage Used (PDLU):** If the Percentage Used field in the Controller Health data structure changes state, then this bit shall be set to '1'.
09	0	**Composite Temperature Change (CTEMP):** If the Composite Temperature field in the Controller Health data structure changes state, then this bit shall be set to '1'.
08	HwInit	**Controller Status Change (CSTS):** If any bit or field in the Controller Status field in the Controller Health data structure (e.g., the Shutdown Status field, the Ready bit, the Controller Fatal Status bit, the NVM Subsystem Reset Occurred bit, the Controller Enable Change Occurred bit, the Namespace Attribute Changed bit, the Firmware Activated bit, or the Telemetry Controller-Initiated Data Available bit) changes state, then this bit shall be set to '1'.
07	HwInit	**Firmware Activated (FA):** If the Firmware Activated bit in the Controller Health data structure transitions from '0' to '1', then this bit shall be set to '1'.
06	0	**Namespace Attribute Changed (NAC):** If the Namespace Attribute Changed bit in the Controller Health data structure transitions from '0' to '1', then this bit shall be set to '1'.
05	HwInit	**Controller Enable Change Occurred (CECO):** If the Controller Enable Change Occurred bit in the Controller Health data structure changes state, then this bit shall be set to '1'.
04	HwInit	**NVM Subsystem Reset Occurred (NSSRO):** If the NVM Subsystem Reset Occurred bit in the Controller Health data structure transitions from '0' to '1', then this bit shall be set to '1'.
03	0	Reserved
02	0	**Shutdown Status (SHST):** If the Shutdown Status field in the Controller Health data structure changes state, then this bit shall be set to '1'.
01	HwInit	**Controller Fatal Status (CFS):** If the Controller Fatal Status bit in the Controller Health data structure changes state, then this bit shall be set to '1'.
00	0	**Ready (RDY):** If the Ready bit in the Controller Health data structure changes state, then this bit shall be set to '1'.

Notes:

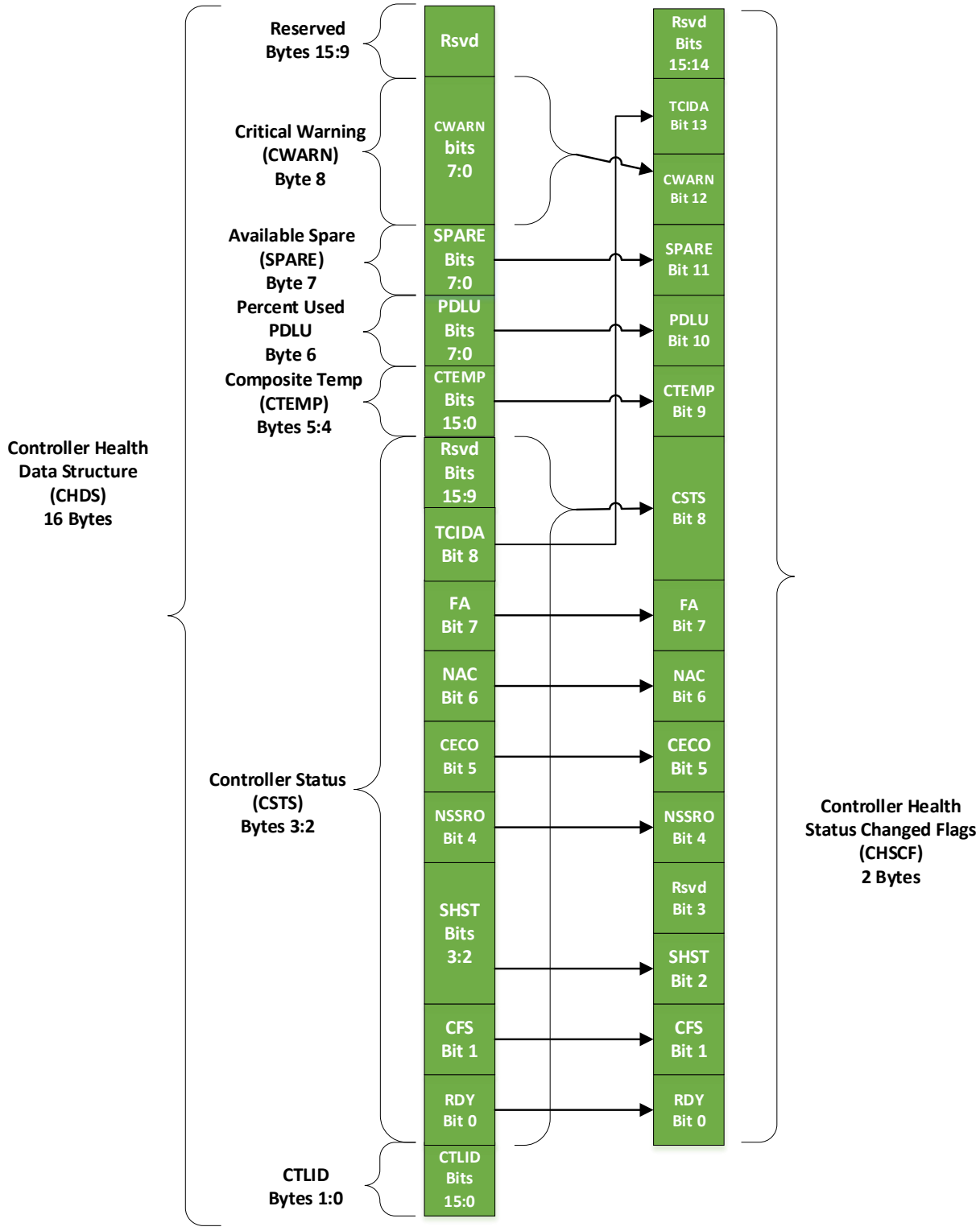
1. An NVM Subsystem Reset shall reset the instance of the Controller Health Status Changed Flags field dedicated to each Management Endpoint in the NVM Subsystem and the instance of the Controller Health Status Changed Flags field dedicated to each Controller in the NVM Subsystem.

   The instance of the Controller Health Status Changed Flags field dedicated to a Controller shall be reset by a Controller Level Reset (refer to the NVM Express Base Specification) of that Controller. Note that a Controller Level Reset may affect the Controller Health Status Changed Flags field in the out-of-band mechanism (e.g., a Controller Level Reset causes the CECO bit in the instance of the Controller Health Status Changed Flags field dedicated to the out-of-band mechanism to be set to '1').

   The instance of the Controller Health Status Changed Flags field dedicated to a Management Endpoint shall be reset by a Management Endpoint Reset of that Management Endpoint.

   No instance of the Controller Health Status Changed Flags field shall be reset by any other resets other than the resets documented by this note.

Figure 99: Controller Health Data Structure to Controller Health Status Changed Flags Mapping



5.3.1 Controller Selection Criteria

A Controller Health Status Poll response returns the Controller Health data structure for up to 255 Controllers in the Response Data field. An NVM Subsystem contains up to 64 Ki Controllers, so a method is required to limit the size of the Response Message. The Starting Controller ID field in the Command Message specifies the Controller ID of the first Controller whose Controller Health data structure may be returned in the Response Data field. The Maximum Response Entries field specifies the maximum number of Controllers whose Controller Health data structure may be returned in the Response Data field.

The Response Data field shall contain the entire Controller Health data structure for the first M Controllers in order of ascending Controller ID, where M is equal to the value in the Maximum Response Entries field, for any Controller that:

- has a Controller ID that is greater than or equal to the value in the Starting Controller ID field;
- matches the controller type selection criteria (refer to section 5.3.1.1); and
- either:
  a. matches the Controller Health Status Changed Flags field selection criteria (refer to section 5.3.1.2); or
  b. has been requested to report all changes to the Controller's Controller Health Status Changed Flags field (i.e., the Report All bit is set to '1' in the NVMe Management Dword 0 field in the Controller Health Status Poll command).

The Response Data field shall not contain the Controller Health data structure for any Controllers that do not meet the selection criteria in this section.

5.3.1.1 Selection Criteria by Controller Type

The Controllers whose Controller Health data structure are returned by the Controller Health Status Poll command are selected based on Controller type (i.e., non-SR-IOV PCI Function, SR-IOV PF, and SR-IOV VF). Controller type selection is controlled by the Include PCI Functions (INCF), Include SR-IOV PFs (INCPF), and Include SR-IOV VFs (INCVF) bits in the NVMe Management Dword 0 field. If one or more of the INCF, INCPF, or INCVF bits are set to '1', then a Controller Health data structure for Controllers corresponding to that type of PCI Function shall be included in the Response Data field unless excluded by other selection criteria as described in section 5.3.1.

5.3.1.2 Selection Criteria by Controller Health Status Changed Flags

If the Report All bit is cleared to '0', then the Controllers whose Controller Health data structure are returned by the Controller Health Status Poll command are also selected based on the Controller Health Status Changed Flags field. Selection of Controllers by the Controller Health Status Changed Flags field is controlled by the CWARN, SPARE, PDLU, CTEMP, and CSTS bits in the NVMe Management Dword 1 field (refer to Figure 95). If one or more of the CWARN, SPARE, PDLU, CTEMP, or CSTS bits in the NVMe Management Dword 1 field are set to '1' and any of the corresponding bits in the Controller Health Status Changed Flags field for the Controller are also set to '1' (refer to Figure 95 for the bits in the Controller Health Status Changed Flags field that are associated with each bit in the NVMe Management Dword 1 field), then the entire Controller Health data structure for that Controller shall be returned in the Response Data field unless excluded by other selection criteria as described in section 5.3.1.

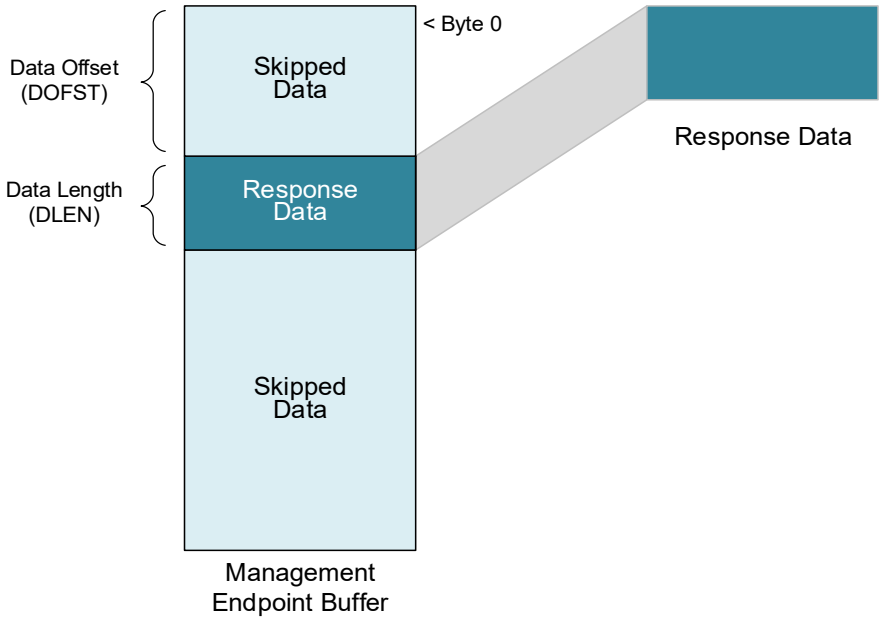
If the Report All bit is cleared to '0', then the contents returned in the Controller Health data structure for the CWARN, SPARE, PDLU, CTEMP, and CSTS fields are undefined if the corresponding CWARN, SPARE, PDLU, CTEMP, or CSTS bit in the NVMe Management Dword 1 field is cleared to '0'. If the Report All bit is set to '1', then the contents returned in the Controller Health data structure for the CWARN, SPARE, PDLU, CTEMP, and CSTS fields shall be valid regardless of the value of the corresponding CWARN, SPARE, PDLU, CTEMP, or CSTS bit in the NVMe Management Dword 1 field.

5.4 Management Endpoint Buffer Read

The Management Endpoint Buffer Read command allows the Management Controller to read the contents of the Management Endpoint Buffer. This data is returned in the Response Data.

The command uses the NVMe Management Dword 0 field (refer to Figure 101) and the NVMe Management Dword 1 field (refer to Figure 102). There is no Request Data included in a Management Endpoint Buffer Read command. The NVMe Management Response field is reserved.

Figure 100: Management Endpoint Buffer Read Response Data



If the Data Offset (DOFST) field is greater than or equal to the size of the Management Endpoint Buffer, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the DOFST field. If the DOFST field is less than the size of the Management Endpoint Buffer and the sum of the DOFST and DLEN fields is greater than the size of the Management Endpoint Buffer, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the DLEN field.

If an attempt is made to read Management Endpoint Buffer contents that were zeroed due to an NVM Subsystem sanitize operation (refer to the NVM Express Base Specification), then the Management Endpoint shall respond with a Response Message Status of Management Endpoint Buffer Cleared Due to Sanitize (refer to section 4.2.3.1).

Figure 101: Management Endpoint Buffer Read – NVMe Management Dword 0

Bits	Description
31:00	**Data Offset (DOFST):** This field specifies the starting offset, in bytes, into the Management Endpoint Buffer.

Figure 102: Management Endpoint Buffer Read – NVMe Management Dword 1

Bits	Description
31:16	Reserved

Figure 102: Management Endpoint Buffer Read – NVMe Management Dword 1

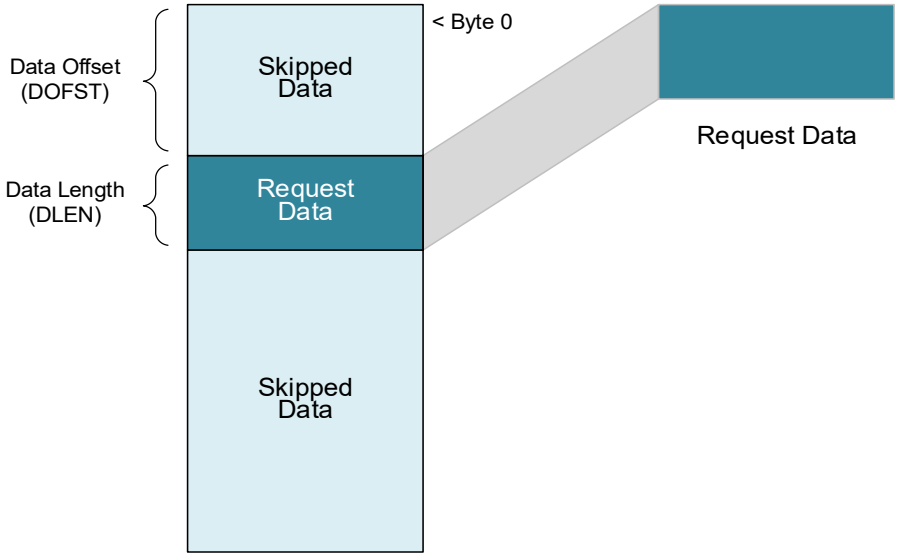
Bits	Description
15:00	**Data Length (DLEN):** This field specifies the length, in bytes, to be transferred from the Management Endpoint Buffer starting at the byte offset specified by DOFST and returned in the Response Data. Specifying a value that is greater than the maximum supported Response Data size results in an Invalid Parameter Error Response with the PEL field indicating this field. A value of 0h in this field is valid. If this field is cleared to 0h, then the Management Endpoint responds with a Success Response and no Response Data.

5.5 Management Endpoint Buffer Write

The Management Endpoint Buffer Write command allows the Management Controller to update the contents of the optional Management Endpoint Buffer. The data used to update the Management Endpoint Buffer is transferred in the Request Data included in a Management Endpoint Buffer Write command.

The command uses the NVMe Management Dword 0 field (refer to Figure 104) and the NVMe Management Dword 1 field (refer to Figure 105). The NVMe Management Response field is reserved and there is no Response Data.

Figure 103: Management Endpoint Buffer Write Request Data



If the Data Offset (DOFST) field is greater than or equal to the size of the Management Endpoint Buffer, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the DOFST field. If the DOFST field is less than the size of the Management Endpoint Buffer and the sum of the DOFST and DLEN fields is greater than the size of the Management Endpoint Buffer, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the DLEN field.

Figure 104: Management Endpoint Buffer Write – NVMe Management Dword 0

Bits	Description
31:00	**Data Offset (DOFST):** This field specifies the starting offset, in bytes, into the Management Endpoint Buffer.

Figure 105: Management Endpoint Buffer Write – NVMe Management Dword 1

Bits	Description
31:16	Reserved
15:00	**Data Length (DLEN):** This field specifies the length, in bytes, to be transferred from the Request Data to the Management Endpoint Buffer starting at the byte offset specified by DOFST. Specifying a DLEN field value that is greater than the maximum supported Response Data size results in an Invalid Parameter Error Response with the PEL field indicating this field. A value of 0h in this field specifies that no data is transferred. This condition shall not be considered an error.

5.6 NVM Subsystem Health Status Poll

The NVM Subsystem Health Status Poll command is used to efficiently determine changes in health status attributes associated with the NVM Subsystem.

The NVM Subsystem Health Status Poll command operates independently for each Management Endpoint in the out-of-band mechanism and each Controller in the in-band tunneling mechanism.

An NVMe Storage Device or NVMe Enclosure supporting the NVM Subsystem Health Status Poll command using the out-of-band mechanism shall have an independent instance of the NVM Subsystem Health data structure (refer to Figure 108) dedicated to each Management Endpoint. In the out-of-band mechanism, an NVM Subsystem Health Status Poll command only applies to the instance of the NVM Subsystem Health data structure dedicated to the Management Endpoint to which the NVM Subsystem Health Status Poll command was issued.

An NVMe Storage Device or NVMe Enclosure supporting the NVM Subsystem Health Status Poll command using the in-band tunneling mechanism shall have an independent instance of the NVM Subsystem Health data structure dedicated to each Controller. In the in-band tunneling mechanism, an NVM Subsystem Health Status Poll command only applies to the instance of the NVM Subsystem Health data structure dedicated to the Controller to which the NVM Subsystem Health Status Poll command was issued.

The NVM Subsystem Health Status Poll command uses the NVMe Management Dword 1 field as shown in Figure 106.

Figure 106: NVM Subsystem Health Status Poll - NVMe Management Dword 1

Bits	Description
31	**Clear Status (CS):** If this bit is set to ‘1’, then the Management Endpoint shall perform the following steps atomically in the order listed: 1. copy the current value of the Composite Controller Status Flags field (refer to Figure 107) to the Composite Controller Status field of the Response Message (refer to Figure 108); and 2. clear the Composite Controller Status Flags field to 0h. If this bit is cleared to ‘0’, then the Management Endpoint shall copy the current value of the Composite Controller Status Flags field (refer to Figure 107) to the Composite Controller Status field of the Response Message (refer to Figure 108) and shall not modify the Composite Controller Status Flags field.
30:00	Reserved

All other command-specific fields are reserved.

The NVM Subsystem Health data structure, shown in Figure 108, shall be returned starting at offset 0h in the Response Data of a Success Response. The NVM Subsystem Health Status Poll command Response Messages do not use the NVMe Management Response field and this field shall be reserved. The Response Data field shall be the size of the NVM Subsystem Health data structure.

Figure 107: Composite Controller Status Data Structure (CCSDS)

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>1:0</td><td>
Composite Controller Status Flags (CCSF): This field indicates the composite status of all Controllers in the NVM Subsystem.

Bits in this field are cleared to ‘0’ as described in the Clear Status field (refer to Figure 106). A Configuration Set command that specifies a Configuration Identifier value of 02h (Health Status Change) in the NVMe Management Dword 1 field clears selected bits to ‘0’ (refer to section 5.1.2).
<table>
<tr><th>Bits</th><th>Reset[1]</th><th>Description</th></tr>
<tr><td>15:14</td><td>0</td><td>Reserved</td></tr>
<tr><td>13</td><td>HwInit</td><td>Telemetry Controller-Initiated Data Available (TCIDA): If the Telemetry Controller-Initiated Data Available bit in the Controller Health Status Changed Flags field transitions from ‘0’ to ‘1’ in one or more Controllers in the NVM Subsystem, then this bit shall be set to ‘1’.

If the Telemetry Controller-Initiated log page is NVM Subsystem in scope (refer to the Telemetry Controller-Initiated Scope (TCS) field in the NVM Express Base Specification), then:
• if this bit is set to ‘1’, then the Telemetry Controller-Initiated log page shall contain saved internal state (i.e., one or more of the Telemetry Controller-Initiated data areas shall contain valid internal data) available by issuing a Get Log Page command to any Controller in the NVM Subsystem using the out-of-band mechanism; and
• if this bit is cleared to ‘0’, then the Telemetry Controller-Initiated log page returned in response to issuing a Get Log Page command using the out-of-band mechanism shall not contain saved internal state (i.e., Telemetry Controller-Initiated Data Area 1, Telemetry Controller-Initiated Data Area 2, Telemetry Controller-Initiated Data Area 3 and Telemetry Controller-Initiated Data Area 4 are not present).

If the Telemetry Controller-Initiated log page is Controller in scope, then refer to the Telemetry Controller-Initiated Data Available bit in the Controller Health Data Structure (refer to Figure 97) of each Controller in the NVM Subsystem to determine the availability of the Telemetry Controller-Initiated log page.

The value of this bit shall persist across all resets and power cycles.</td></tr>
<tr><td>12</td><td>0</td><td>Critical Warning (CWARN): If the Critical Warning bit in the Controller Health Status Changed Flags field transitions from ‘0’ to ‘1’ in one or more Controllers in the NVM Subsystem, then this bit shall be set to ‘1’.</td></tr>
<tr><td>11</td><td>0</td><td>Available Spare (SPARE): If the Available Spare bit in the Controller Health Status Changed Flags field transitions from ‘0’ to ‘1’ in one or more Controllers in the NVM Subsystem, then this bit shall be set to ‘1’.</td></tr>
<tr><td>10</td><td>0</td><td>Percentage Used (PDLU): If the Percentage Used bit in the Controller Health Status Changed Flags field transitions from ‘0’ to ‘1’ in one or more Controllers in the NVM Subsystem, then this bit shall be set to ‘1’.</td></tr>
<tr><td>09</td><td>0</td><td>Composite Temperature Change (CTEMP): If the Composite Temperature bit in the Controller Health Status Changed Flags field transitions from ‘0’ to ‘1’ in one or more Controllers in the NVM Subsystem, then this bit shall be set to ‘1’.</td></tr>
<tr><td>08</td><td>HwInit</td><td>Controller Status Change (CSTS): If the Controller Status Change bit in the Controller Health Status Changed Flags field transitions from ‘0’ to ‘1’ in one or more Controllers in the NVM Subsystem, then this bit shall be set to ‘1’.</td></tr>
<tr><td>07</td><td>HwInit</td><td>Firmware Activated (FA): If the Firmware Activated bit in the Controller Health Status Changed Flags field transitions from ‘0’ to ‘1’ in one or more Controllers in the NVM Subsystem, then this bit shall be set to ‘1’.</td></tr>
<tr><td>06</td><td>0</td><td>Namespace Attribute Changed (NAC): If the Namespace Attribute Changed bit in the Controller Health Status Changed Flags field transitions from ‘0’ to ‘1’ in one or more Controllers in the NVM Subsystem, then this bit shall be set to ‘1’.</td></tr>
</table>
</td></tr>
</table>

	05	HwInit	**Controller Enable Change Occurred (CECO):** If the Controller Enable Change Occurred bit in the Controller Health Status Changed Flags field transitions from '0' to '1' in one or more Controllers in the NVM Subsystem, then this bit shall be set to '1'.
	04	HwInit	**NVM Subsystem Reset Occurred (NSSRO):** If the value of the NVM Subsystem Reset Occurred bit in the Controller Health Status Changed Flags field transitions from a '0' to a '1' in one or more Controllers in the NVM Subsystem, then this bit shall be set to '1'.
	03	0	Reserved
	02	0	**Shutdown Status (SHST):** If the Shutdown Status bit in the Controller Health Status Changed Flags field transitions from '0' to '1' in one or more Controllers in the NVM Subsystem, then this bit shall be set to '1'.
	01	HwInit	**Controller Fatal Status (CFS):** If the Controller Fatal Status bit in the Controller Health Status Changed Flags field transitions from '0' to '1' in one or more Controllers in the NVM Subsystem, then this bit shall be set to '1'.
	00	0	**Ready (RDY):** If the Ready bit in the Controller Health Status Changed Flags field transitions from '0' to '1' in one or more Controllers in the NVM Subsystem, then this bit shall be set to '1'.

Notes:

1. An NVM Subsystem Reset shall reset the instance of the Composite Controller Status Flags field dedicated to each Management Endpoint in the NVM Subsystem and the instance of the Composite Controller Status Flags field dedicated to each Controller in the NVM Subsystem.

   The instance of the Composite Controller Status Flags field dedicated to a Controller shall be reset by a Controller Level Reset (refer to the NVM Express Base Specification) of that Controller. Note that a Controller Level Reset may affect the Composite Controller Status Flags field in the out-of-band mechanism (e.g., a Controller Level Reset causes the CECO bit in the instance of the Composite Controller Status Flags field dedicated to the out-of-band mechanism to be set to '1').

   The instance of the Composite Controller Status Flags field dedicated to a Management Endpoint shall be reset by a Management Endpoint Reset of that Management Endpoint.

   No instance of the Composite Controller Status Flags field shall be reset by any other resets other than the resets documented by this note.

Figure 108: NVM Subsystem Health Data Structure (NSHDS)

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>0</td><td>NVM Subsystem Status (NSS): This field indicates the status of the NVM Subsystem.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>AEM Transmission Failure (ATF): If there is an AEM transmission failure on any Management Endpoint in the NVM Subsystem, then this bit shall be set to ‘1’.

An AEM transmission failure on a given Management Endpoint occurs when:

a) the AEM Retry Delay field is not cleared to 0h and the amount of time specified by the AEM Retry Delay has elapsed since the end of transmission of the final attempt (refer to section 4.4.3) to transmit the AEM without processing an AEM Ack;
b) the AEM Retry Delay field is cleared to 0h and 5 s has elapsed since the end of transmission of the first attempt to transmit the AEM without processing an AEM Ack, an AE Sync, or a Management Endpoint Reset; or
c) the physical transport external to the NVM Subsystem is unavailable when the Management Endpoint attempts the final transmission of an AEM (e.g., an AEM on a PCIe VDM Management Endpoint is unable to be transmitted due to the PCIe link being down).

This bit shall be cleared to ‘0’ if:

a) an AE Sync occurs; or
b) a Management Endpoint Reset occurs.</td></tr>
<tr><td>6</td><td>Sanitize Failure Mode (SFM): If the NVM Subsystem is in the Sanitize failure mode, then this bit shall be set to ‘1’. If the NVM Subsystem is not in the Sanitize failure mode, then this bit shall be cleared to ‘0’. The NVM Subsystem is in the Sanitize failure mode while the most recent NVM Subsystem sanitize operation failed and no recovery action has been completed successfully (refer to the NVM Express Base Specification).</td></tr>
<tr><td>5</td><td>Drive Functional (DF): If the NVM Subsystem is functional, then this bit shall be set to ‘1’. If there is an unrecoverable failure detected in the NVM Subsystem, then this bit shall be cleared to ‘0’.</td></tr>
<tr><td>4</td><td>Reset Not Required (RNR): If the NVM Subsystem does not require an NVM Subsystem Reset to resume normal operation, then this bit shall be set to ‘1’.

If the NVM Subsystem does require an NVM Subsystem Reset to resume normal operation, then this bit shall be cleared to ‘0’.</td></tr>
<tr><td>3</td><td>Port 0 PCIe Link Active (P0LA): If the PCIe link on the port with the lowest Port Identifier is active (i.e., the Data Link Control and Management State Machine is in the DL_Active state as defined by the PCI Express Base Specification), then this bit shall be set to ‘1’.

If the PCIe link on the port with the lowest Port Identifier is not active (i.e., the Data Link Control and Management State Machine is not in the DL_Active state as defined by the PCI Express Base Specification), then this bit shall be cleared to ‘0’.</td></tr>
<tr><td>2</td><td>Port 1 PCIe Link Active (P1LA): If the PCIe link on the port with the second lowest Port Identifier is active (i.e., the Data Link Control and Management State Machine is in the DL_Active state as defined by the PCI Express Base Specification), then this bit shall be set to ‘1’.

If the PCIe link on the port with the second lowest Port Identifier is not active (i.e., the Data Link Control and Management State Machine is not in the DL_Active state as defined by the PCI Express Base Specification) or there is no port 1, then this bit shall be cleared to ‘0’.</td></tr>
<tr><td>1</td><td>Sanitize Namespace Failure Mode (SNFM): If any namespace in the NVM Subsystem is in the Sanitize Namespace failure mode, then this bit shall be set to ‘1’. If no namespaces in the NVM Subsystem are in the Sanitize Namespace failure mode, then this bit shall be cleared to ‘0’. A namespace is in the Sanitize Namespace failure mode while the most recent namespace sanitize operation failed and no recovery action has been completed successfully (refer to the NVM Express Base Specification).</td></tr>
<tr><td>0</td><td>Reserved</td></tr>
</table></td></tr>
</table>

Figure 108: NVM Subsystem Health Data Structure (NSHDS)

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>1</td><td>SMART Warnings (SW): This field indicates the inverted value of the Critical Warning field (i.e., byte 0) of the NVMe SMART / Health Information log page. Each bit in this field shall be inverted from the value in the Critical Warning field of the SMART / Health Information log page as defined by the NVM Express Base Specification.

If there are multiple Controllers in the NVM Subsystem, the Responder shall combine the Critical Warning field from every Controller in the NVM Subsystem. Each bit in this field is:

• cleared to ‘0’ if the corresponding bit in the Critical Warning field of the SMART / Health Information log page of any Controller in the NVM Subsystem is set to ‘1’; or
• set to ‘1’ if the corresponding bit in the Critical Warning field of the SMART / Health Information log page in all Controllers in the NVM Subsystem is cleared to ‘0’.</td></tr>
<tr><td>2</td><td>Composite Temperature (CTEMP): This field indicates information related to the composite temperature of the NVM Subsystem. The composite temperature of the NVM Subsystem shall be calculated at least every 5 s as follows:

• if there are one or more Controllers in the NVM Subsystem with a Composite Temperature in the SMART / Health Information log page that is less than or equal to an under-temperature threshold; and
• no Controllers in the NVM Subsystem with a Composite Temperature in the SMART / Health Information log page that is greater than or equal to an over-temperature threshold (refer to the NVM Express Base Specification),

then the composite temperature of the NVM Subsystem shall be the same temperature as the Composite Temperature from the SMART / Health Information log page of the coldest Controller in the NVM Subsystem; otherwise, the composite temperature of the NVM Subsystem shall be the same temperature as the Composite Temperature from the SMART / Health Information log page of the hottest Controller in the NVM Subsystem.

The reported temperature range is implementation specific. The values for this field are as follows:
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00h to 7Eh</td><td>If the composite temperature of the NVM Subsystem is greater than or equal to 0 °C and less than or equal to 126 °C, then this field shall indicate the composite temperature of the NVM Subsystem in degrees Celsius.</td></tr>
<tr><td>7Fh</td><td>If the composite temperature of the NVM Subsystem is greater than or equal to 127 °C, then this field shall indicate a value of 7Fh.</td></tr>
<tr><td>80h</td><td>If the composite temperature of the NVM Subsystem is greater than 5 s old, then this field shall indicate a value of 80h.</td></tr>
<tr><td>81h</td><td>If the composite temperature of the NVM Subsystem is not accurate due to the failure of one or more temperature sensors, then this field shall indicate a value of 81h.</td></tr>
<tr><td>82h to C3h</td><td>Reserved</td></tr>
<tr><td>C4h</td><td>If the composite temperature of the NVM Subsystem is less than or equal to -60 °C, then this field shall indicate a value of C4h.</td></tr>
<tr><td>C5h to FFh</td><td>If the composite temperature of the NVM Subsystem is less than or equal to -1 °C and greater than or equal to -59 °C, then this field shall indicate the two’s complement of the composite temperature of the NVM Subsystem in degrees Celsius.</td></tr>
</table></td></tr>
<tr><td>3</td><td>Percentage Drive Life Used (PDLU): This field shall indicate an implementation-specific estimate of the percentage of NVM Subsystem NVM life used based on the actual usage and the manufacturer’s prediction of NVM life. If an NVM Subsystem has multiple Controllers, then the highest value shall be returned. A value of 100 indicates that the estimated endurance of the NVM in the NVM Subsystem has been consumed but may not indicate an NVM Subsystem failure. The value is allowed to exceed 100. Percentages greater than 254 shall be represented as 255. This value should be updated once per power-on hour and equal the Percentage Used value in the SMART / Health Information log page.</td></tr>
<tr><td>5:4</td><td>Composite Controller Status (CCS): This field shall indicate the Composite Controller Status Flags field (refer to Figure 107).</td></tr>
<tr><td>7:6</td><td>Reserved</td></tr>
</table>

5.7 Read NVMe-MI Data Structure

The Read NVMe-MI Data Structure command requests data that describes information about the NVM Subsystem, the Management Endpoint, or the NVMe Controllers.

The command uses the NVMe Management Dword 0 and Dword 1 fields. The format of the NVMe Management Dword 0 field is shown in Figure 109 and the format of the NVMe Management Dword 1 field is shown in Figure 110. There is no Request Data included in a Read NVMe-MI Data Structure command.

Some port-specific Data Structure Types are accessible from any Responder. Other port-specific Data Structure Types (e.g., Optionally Supported Command List) are only accessible from the Responder that received the Command Message.

Figure 109: Read NVMe-MI Data Structure – NVMe Management Dword 0

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:24</td><td>Data Structure Type (DTYP): This field specifies the data structure that shall be returned.
<table>
<tr><th>Value</th><th>Definition</th><th>Reference</th></tr>
<tr><td>00h</td><td>NVM Subsystem Information</td><td>5.7.1</td></tr>
<tr><td>01h</td><td>Port Information</td><td>5.7.2</td></tr>
<tr><td>02h</td><td>Controller List</td><td>5.7.3</td></tr>
<tr><td>03h</td><td>Controller Information</td><td>5.7.4</td></tr>
<tr><td>04h</td><td>Optionally Supported Command List</td><td>5.7.5</td></tr>
<tr><td>05h</td><td>Management Endpoint Buffer Command Support List</td><td>5.7.6</td></tr>
<tr><td>06h to FFh</td><td>Reserved</td><td></td></tr>
</table></td></tr>
<tr><td>23:16</td><td>Port Identifier (PORTID): This field specifies the identifier of the port whose data structure is requested to be returned.

If the DTYP field value is 01h (i.e., Port Information) or 05h (i.e., Management Endpoint Buffer Command Support List), then this field specifies the Port Identifier of the port whose information shall be returned.

For all other non-reserved values of the DTYP field, this field shall be ignored by the Management Endpoint.</td></tr>
<tr><td>15:00</td><td>Controller Identifier (CTRLID): This field specifies the Controller Identifier of the Controller whose data structure is returned.
<table>
<tr><th>DTYP Value [1]</th><th>CTRLID Usage</th></tr>
<tr><td>02h</td><td>This field contains the Controller Identifier used to return a Controller List data structure for the NVM Subsystem as described in section 5.7.3.</td></tr>
<tr><td>03h</td><td>This field contains the Controller Identifier of the Controller for which the information is returned as described in section 5.7.4.</td></tr>
<tr><td>04h</td><td>This field contains the Controller Identifier of the Controller used to filter which optional NVM Express Admin Command Set commands (i.e., the NMIMT field is set to 02h) are returned in the Optionally Supported Command List data structure entries as described in section 5.7.5.</td></tr>
<tr><td colspan="2">Notes:
1. For all other non-reserved values of the DTYP field, this field should be ignored by the Management Endpoint.</td></tr>
</table>
For all other non-reserved values of the DTYP field, this field shall be ignored by the Management Endpoint.</td></tr>
</table>

Figure 110: Read NVMe-MI Data Structure – NVMe Management Dword 1

Bits	Description
31:08	Reserved

Figure 110: Read NVMe-MI Data Structure – NVMe Management Dword 1

Bits	Description
07:00	**I/O Command Set Identifier (IOCSI):** If the DTYP field value is 04h (i.e., Optionally Supported Command List) or 05h (i.e., Management Endpoint Buffer Command Support List), then for commands with the NMIMT field set to a value of 02h (i.e., NVMe Admin Command) in the: a) Optionally Supported Command List data structure; or b) Management Endpoint Buffer Supported Command List data structure, this field specifies the I/O Command Set that shall be used to select the optional I/O Command Set Specific Admin commands. For more information about I/O Command Sets refer to the NVM Express Base Specification. For all non-reserved values of the DTYP field, other than the values 04h and 05h, this field is not applicable and shall be ignored by the Management Endpoint. If the DTYP field is 04h or 05h, then for commands with the NMIMT field set to any non-reserved value other than 02h in the Optionally Supported Command List data structure or Management Endpoint Buffer Supported Command List data structure, this field is not applicable and shall be ignored by the Management Endpoint. The I/O Command Set specified by this field is not required to be enabled (refer to the NVM Express Base Specification).

Upon successful completion of the Read NVMe-MI Data Structure command, the NVMe Management Response field is shown in Figure 111 and the specified data structure is returned in the Response Data.

Figure 111: Read NVMe-MI Data Structure – NVMe Management Response

Bits	Description
23:16	Reserved
15:00	**Response Data Length (RDL):** The length, in bytes, of the Response Data field in this Response Message.

5.7.1 NVM Subsystem Information Response Data

The NVM Subsystem Information data structure contains information about the NVM Subsystem. The Port Identifier field and the Controller Identifier field in the NVMe Management Dword 0 field are reserved. The format of the NVM Subsystem Information data structure is shown in Figure 112.

Figure 112: NVM Subsystem Information Data Structure

Bytes	Description
00	**Number of Ports (NUMP):** This field indicates the maximum number of ports of any type supported by the NVM Subsystem. This is a 0's based value. The value of FFh is not supported because a port identifier of 256 is not able to be reported (refer to section 5.1.1).
01	**NVMe-MI Major Version Number (MJR):** An integer value indicating the major version number of this specification supported by the NVM Subsystem, as defined in Figure 113.
02	**NVMe-MI Minor Version Number (MNR):** An integer value indicating the minor version number of this specification supported by the NVM Subsystem, as defined in Figure 113.

Figure 112: NVM Subsystem Information Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>03</td><td>NVMe-MI NVM Subsystem Capabilities (NNSC): This field indicates the NVMe-MI capabilities of the NVM Subsystem.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>Status Reporting Enhancements (SRE): If the status reporting enhancements are supported, then this bit shall be set to ‘1’. If the status reporting enhancements are not supported, then this bit is reserved.

Implementations compliant to versions of this specification later than 1.2 shall set this bit to ‘1’.

The status reporting enhancements consist of the following:
• specified in the Get State Control Primitive that the NVM Subsystem Reset Occurred status bit is only set if a new firmware image is not activated to align to the behavior of NVM Subsystem Reset Occurred in the rest of this specification and the NVM Express Base Specification;
• specified that a Controller Health Status Poll command with the Clear Changed Flags bit set to ‘1’ also clears the Namespace Attribute Changed and Firmware Activated bits in Controller Health data structure to ‘0’;
• changed the Controller Enable Change Occurred bit in the Controller Health data structure from a status bit that is set when the Controller is enabled or disabled to a state bit that indicates if the Controller is currently enabled or disabled; and
• set the:
o Controller Enable Change Occurred bit;
o Critical Warning bit;
o Controller Status Change bit; or
o Controller Fatal Status bit;

in the Controller Health Status Changed Flags field whenever the corresponding bit in the Controller Health data structure changes state from ‘0’ to ‘1’ or ‘1’ to ‘0’ instead of only when it transitions from ‘0’ to ‘1’.</td></tr>
</table></td></tr>
<tr><td>31:04</td><td>Reserved</td></tr>
</table>

Published versions of this specification and the values that shall be reported by compliant implementations are defined in Figure 113.

Figure 113: Version Number Field Values

Specification Versions [1]	MJR Field	MNR Field
1.0	1h	0h
1.1	1h	1h
1.2	1h	2h
2.0	2h	0h
2.1	2h	1h

Notes:
1. The specification version listed includes lettered versions (e.g., 1.0 includes 1.0 and 1.0a, 1.1 includes 1.1 and 1.1a through 1.1d, etc.).

5.7.2 Port Information Response Data

The Port Information data structure contains information about a port within the NVM Subsystem. The Port Identifier field in the NVMe Management Dword 0 field specifies the port. The Controller Identifier field in the NVMe Management Dword 0 field is reserved. The format of the Port Information data structure is shown in Figure 114.

Figure 114: Port Information Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>00</td><td>Port Type (PRTTYP): Specifies the port type.
<table>
<tr><th>Value</th><th>Definition</th><th>Reference</th></tr>
<tr><td>0h</td><td colspan="2">Inactive</td></tr>
<tr><td>1h</td><td>PCIe</td><td>Figure 115</td></tr>
<tr><td>2h</td><td>2-Wire</td><td>Figure 116</td></tr>
<tr><td>3h to FFh</td><td colspan="2">Reserved</td></tr>
</table></td></tr>
<tr><td>01</td><td>Port Capabilities (PRTCAP): This field contains information about the capabilities of the port.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:2</td><td>Reserved</td></tr>
<tr><td>1</td><td>Asynchronous Event Messages Supported (AEMS): If this bit is set to ‘1’, then all Management Endpoints on this port shall support AEMs (refer to section 4.4) and the Asynchronous Event configuration (i.e., Configuration Identifier 04h). If this bit is cleared to ‘0’, then all Management Endpoints on this port shall not support AEMs or the Asynchronous Event configuration.</td></tr>
<tr><td>0</td><td>Command Initiated Auto Pause Supported (CIAPS): If this bit is set to ‘1’, then the Command Initiated Auto Pause (CIAP) bit is supported in Command Messages on this port. If this bit is cleared to ‘0’, then the CIAP bit is not supported in Command Messages on this port.</td></tr>
</table></td></tr>
<tr><td>03:02</td><td>Maximum MCTP Transmission Unit Size (MMTUS): The maximum MCTP Transmission Unit size that all Management Endpoints on the port are capable of sending and receiving. If:

the port does not support MCTP, then this field shall be cleared to 0h;
the Port Type is PCIe and the port supports MCTP, then this field shall be set to a value between 64 bytes and the PCIe Max Payload Size Supported (refer to the PCI Express Base Specification), inclusive. All PCIe ports within an NVM Subsystem should report the same value in this field; and
the Port Type is 2-Wire and the port supports MCTP over SMBus, then this field shall be set to a value between 64 bytes and 250 bytes, inclusive.
</td></tr>
<tr><td>07:04</td><td>Management Endpoint Buffer Size (MEBS): This field specifies the size of the Management Endpoint Buffer in bytes when a Management Endpoint Buffer is supported.

A value of 0h in this field indicates that the Management Endpoint does not support a Management Endpoint Buffer.</td></tr>
<tr><td>31:08</td><td>Port Type Specific (PTSP): Refer to Figure 115 and Figure 116.</td></tr>
</table>

Figure 115: PCIe Port Specific Data

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>08</td><td>PCIe Maximum Payload Size (PCIEMPS): This field indicates the Max_Payload_Size setting for the specified PCIe port (refer to the PCI Express Base Specification). If the link is not active, this field should be cleared to 0h.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0h</td><td>128 bytes</td></tr>
<tr><td>1h</td><td>256 bytes</td></tr>
<tr><td>2h</td><td>512 bytes</td></tr>
<tr><td>3h</td><td>1 KiB</td></tr>
<tr><td>4h</td><td>2 KiB</td></tr>
<tr><td>5h</td><td>4 KiB</td></tr>
<tr><td>6h to FFh</td><td>Reserved</td></tr>
</table>
The value reported in this field by ARI Devices and Non-ARI Multi-Function Devices (refer to the PCI Express Base Specification) whose Max Payload Size settings are identical across all Functions is the setting in Function 0. The value reported in this field by non-ARI Multi-Function Devices whose Max Payload Size settings are not identical across all Functions is implementation specific.</td></tr>
</table>

Figure 115: PCIe Port Specific Data

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>09</td><td>PCIe Supported Link Speeds Vector (PCIESLSV): This field shall indicate the Supported Link Speeds for the specified PCIe port.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:6</td><td>Reserved</td></tr>
<tr><td>5</td><td>64.0 GT/s Support (GTS64): This field is set to ‘1’ if the PCIe link supports 64.0 GT/s; otherwise, this field is cleared to ‘0’.</td></tr>
<tr><td>4</td><td>32.0 GT/s Support (GTS32): This field is set to ‘1’ if the PCIe link supports 32.0 GT/s; otherwise, this field is cleared to ‘0’.</td></tr>
<tr><td>3</td><td>16.0 GT/s Support (GTS16): This field is set to ‘1’ if the PCIe link supports 16.0 GT/s; otherwise, this field is cleared to ‘0’.</td></tr>
<tr><td>2</td><td>8.0 GT/s Support (GTS8): This field is set to ‘1’ if the PCIe link supports 8.0 GT/s; otherwise, this field is cleared to ‘0’.</td></tr>
<tr><td>1</td><td>5.0 GT/s Support (GTS5): This field is set to ‘1’ if the PCIe link supports 5.0 GT/s; otherwise, this field is cleared to ‘0’.</td></tr>
<tr><td>0</td><td>2.5 GT/s Support (GTS2P5): This field is set to ‘1’ if the PCIe link supports 2.5 GT/s; otherwise, this field is cleared to ‘0’.</td></tr>
</table></td></tr>
<tr><td>10</td><td>PCIe Current Link Speed (PCIECLS): This field shall indicate the port’s PCIe negotiated link speed.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0h</td><td>Link not active</td></tr>
<tr><td>1h</td><td>The current link speed is 2.5 GT/s.</td></tr>
<tr><td>2h</td><td>The current link speed is 5.0 GT/s.</td></tr>
<tr><td>3h</td><td>The current link speed is 8.0 GT/s.</td></tr>
<tr><td>4h</td><td>The current link speed is 16.0 GT/s.</td></tr>
<tr><td>5h</td><td>The current link speed is 32.0 GT/s.</td></tr>
<tr><td>6h</td><td>The current link speed is 64.0 GT/s.</td></tr>
<tr><td>7h to FFh</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>11</td><td>PCIe Maximum Link Width (PCIEMLW): The maximum PCIe link width for this NVM Subsystem port. This is the expected negotiated link width that the port link trains to if the platform supports it. A Requester may compare this value with the PCIe Negotiated Link Width to determine if there has been a PCIe link training issue.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>PCIe x1</td></tr>
<tr><td>2</td><td>PCIe x2</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
<tr><td>4</td><td>PCIe x4</td></tr>
<tr><td>5 to 7</td><td>Reserved</td></tr>
<tr><td>8</td><td>PCIe x8</td></tr>
<tr><td>9 to 11</td><td>Reserved</td></tr>
<tr><td>12</td><td>PCIe x12</td></tr>
<tr><td>13 to 15</td><td>Reserved</td></tr>
<tr><td>16</td><td>PCIe x16</td></tr>
<tr><td>17 to 31</td><td>Reserved</td></tr>
<tr><td>32</td><td>PCIe x32</td></tr>
<tr><td>33 to 255</td><td>Reserved</td></tr>
</table></td></tr>
</table>

Figure 115: PCIe Port Specific Data

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>12</td><td>PCIe Negotiated Link Width (PCIENLW): The negotiated PCIe link width for this port.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0</td><td>Link not active</td></tr>
<tr><td>1</td><td>PCIe x1</td></tr>
<tr><td>2</td><td>PCIe x2</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
<tr><td>4</td><td>PCIe x4</td></tr>
<tr><td>5 to 7</td><td>Reserved</td></tr>
<tr><td>8</td><td>PCIe x8</td></tr>
<tr><td>9 to 11</td><td>Reserved</td></tr>
<tr><td>12</td><td>PCIe x12</td></tr>
<tr><td>13 to 15</td><td>Reserved</td></tr>
<tr><td>16</td><td>PCIe x16</td></tr>
<tr><td>17 to 31</td><td>Reserved</td></tr>
<tr><td>32</td><td>PCIe x32</td></tr>
<tr><td>33 to 255</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>13</td><td>PCIe Port Number (PCIEPN): This field contains the PCIe port number. This is the same value as that reported in the Port Number field in the PCIe Link Capabilities Register (refer to the NVMe over PCIe Transport Specification).</td></tr>
<tr><td>31:14</td><td>Reserved</td></tr>
</table>

Figure 116: 2-Wire Port Specific Data

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>08</td><td>Current VPD Address (CVPDADDR): This field indicates the current VPD SMBus/I2C address. A value of 0h indicates there is no VPD.</td></tr>
<tr><td>09</td><td>Maximum VPD Access Frequency (MVPDFREQ): This field indicates the maximum 2-Wire frequency supported on the VPD interface.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0h</td><td>Not supported</td></tr>
<tr><td>1h</td><td>100 kHz</td></tr>
<tr><td>2h</td><td>400 kHz</td></tr>
<tr><td>3h</td><td>1 MHz</td></tr>
<tr><td>4h to FFh</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>10</td><td>Current Management Endpoint Address (CMEADDR): This field indicates the current 2-Wire address. A value of 0h indicates there is no Management Endpoint on this port.</td></tr>
<tr><td>11</td><td>2-Wire Protocols Supported (TWPRT): This field indicates which 2-Wire protocols are supported and the maximum supported SMBus/I2C frequency.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>I3C Support (I3CSPRT): If the port supports I3C mode, then this bit shall be set to ‘1’; otherwise, this bit shall be cleared to ‘0’.</td></tr>
<tr><td>6:2</td><td>Reserved</td></tr>
<tr><td>1:0</td><td>Maximum SMBus/I2C Frequency (MSMBFREQ): This field shall indicate the support for SMBus/I2C and if supported, then this field shall indicate the maximum frequency supported.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0h</td><td>Not supported</td></tr>
<tr><td>1h</td><td>100 kHz</td></tr>
<tr><td>2h</td><td>400 kHz</td></tr>
<tr><td>3h</td><td>1 MHz</td></tr>
</table></td></tr>
</table></td></tr>
</table>

Figure 116: 2-Wire Port Specific Data

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>12</td><td>NVMe Basic Management (NVMEBM):
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>NVMe Basic Management Support (NVMEBMS): If this bit is set to ‘1’, then the port implements the NVMe Basic Management Command. If this bit is cleared to ‘0’, then the port does not implement the NVMe Basic Management Command.

It is strongly recommended that implementations clear this bit to ‘0’. The NVMe Basic Management Command is included in Appendix A for information purposes only and is not a part of the standard NVMe-MI protocol.</td></tr>
</table></td></tr>
<tr><td>31:13</td><td>Reserved</td></tr>
</table>

5.7.3 Controller List Response Data

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

5.7.4 Controller Information Response Data

The Controller Information data structure shall contain information about the Controller in the NVM Subsystem that is specified in the Controller Identifier field in the NVMe Management Dword 0 field. The format of the Controller Information data structure is shown in Figure 117.

Figure 117: Controller Information Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>00</td><td>Port Identifier (PORTID): This field specifies the PCIe Port Identifier with which the Controller is associated.</td></tr>
<tr><td>04:01</td><td>Reserved</td></tr>
<tr><td>05</td><td>PCIe Routing ID Information (PRII): This field provides additional data about the PCI Express Routing ID (PRI) for the specified Controller.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>PCIe Routing ID Valid (PCIERIV): This bit is set to ‘1’ if the device has captured a Bus Number and Device Number (Bus Number only for ARI devices). This bit is cleared to ‘0’ if the device has not captured a Bus and Device number (Bus Number only for ARI devices).</td></tr>
</table></td></tr>
<tr><td>07:06</td><td>PCIe Routing ID (PRI): This field contains the PCIe Routing ID for the specified Controller.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>15:08</td><td>PCI Bus Number (PCIBN): The Controller’s PCI Bus Number.</td></tr>
<tr><td>07:03</td><td>PCI Device Number (PCIDN): The Controller’s PCI Device Number.</td></tr>
<tr><td>02:00</td><td>PCI Function Number (PCIFN): The Controller’s PCI Function Number.</td></tr>
</table>
Note: For an ARI Device, bits 7:0 represents the (8-bit) Function Number, which replaces the (5-bit) Device Number and (3-bit) Function Number fields above.</td></tr>
<tr><td>09:08</td><td>PCI Vendor ID (PCIVID): The PCI Vendor ID for the specified Controller.</td></tr>
<tr><td>11:10</td><td>PCI Device ID (PCIDID): The PCI Device ID for the specified Controller.</td></tr>
<tr><td>13:12</td><td>PCI Subsystem Vendor ID (PCISVID): The PCI Subsystem Vendor ID for the specified Controller.</td></tr>
<tr><td>15:14</td><td>PCI Subsystem Device ID (PCISDID): The PCI Subsystem Device ID for the specified Controller.</td></tr>
</table>

Figure 117: Controller Information Data Structure

Bytes	Description
16	**PCIe Segment Number (PCIESN):** The Segment Number for the specified Controller when the PCI Express Link is in Flit mode. Refer to the PCI Express Base specification for more information. If the PCI Express interface is not in Flit mode, then this field shall be cleared to 0h.
31:17	Reserved

5.7.5 Optionally Supported Command List Response Data

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

- PCIe Command Set commands (refer to Figure 148);
- Management Interface Command Set commands (refer to Figure 69); and
- NVM Express Admin Command Set commands (refer to Figure 134) supported by the Management Endpoint on the Controller specified by the Controller Identifier (CTRLID) field in the NVMe Management Dword 0 field.

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

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

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

The Optionally Supported Command List data structure shall contain no more than 2,047 commands and shall be minimally sized (e.g., the data structure size is 2 bytes if there are no optionally supported commands and the data structure size is 4 bytes if there is one optionally supported command). The format of the Optionally Supported Command List data structure is shown in Figure 118.

Figure 118: Optionally Supported Command List Data Structure

Bytes	Description
01:00	**Number of Commands (NUMCMD):** This field shall indicate the number of optionally supported commands in the list. If there are no commands in the list, then this field shall be cleared to 0h.
Command Type and Opcode List	
03:02	**Command 0:** This field shall indicate the Command Type and Opcode for the first optionally supported command, if applicable. Refer to Figure 119.
05:04	**Command 1:** This field shall indicate the Command Type and Opcode for the second optionally supported command, if applicable. Refer to Figure 119.
…	
(N*2+3): (N*2+2)	**Command N:** This field shall indicate the Command Type and Opcode for the last optionally supported command, if applicable. Refer to Figure 119. N is equal to the value of the NUMCMD field minus 1h.

Figure 119: Optionally Supported Command Data Structure

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>00</td><td>Command Type (CTYP): This field shall indicate the type of the optionally supported command.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Reserved</td></tr>
<tr><td>6:3</td><td>NVMe-MI Message Type (NMIMT): This field shall indicate the NVMe-MI Message Type of the optionally supported command. Refer to Figure 20.</td></tr>
<tr><td>2:0</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>01</td><td>Opcode (OPC): This field shall indicate the opcode of the optionally supported command.</td></tr>
</table>

5.7.6 Management Endpoint Buffer Command Support List Response Data

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

The Management Endpoint Buffer Command Support List data structure contains a list of commands that support the use of the Management Endpoint Buffer. If the DTYP field value is 05h (i.e., Management Endpoint Buffer Command Support List) and the NMIMT field value is 02h (i.e., NVMe Admin Command), then the I/O Command Set Identifier (IOCSI) field in the NVMe Management Dword 1 field selects the I/O Command Set for the I/O Command Set Specific Admin commands that are returned in the Management Endpoint Buffer Command Support List data structure. The data structure may contain up to 2,047 commands, and shall be minimally sized (i.e., if there is 1 optionally supported command, then the data structure is 4 bytes total).

The list of commands that support the Management Endpoint Buffer may be different among Management Endpoints within the NVM Subsystem. The Port Identifier (PORTID) field in the NVMe Management Dword 0 field of the Read NVMe-MI Data Structure command specifies the port of the Management Endpoint whose Management Endpoint Buffer Command Support List data structure is returned. The format of the Management Endpoint Buffer Supported Command List data structure is shown in Figure 120.

Figure 120: Management Endpoint Buffer Supported Command List Data Structure

Bytes	Description
01:00	**Number of Commands (NUMCMD):** This field contains the number of commands in the list. A value of 0h indicates there are no commands in the list.
Management Endpoint Buffer Supported Command Data Structure List	
03:02	**Command 0:** This field contains the Management Endpoint Buffer Supported Command data structure (refer to Figure 121) for the first command that supports the use of the Management Endpoint Buffer associated with the Management Endpoint, if applicable.
05:04	**Command 1:** This field contains the Management Endpoint Buffer Supported Command data structure (refer to Figure 121) for the second command that supports the use of the Management Endpoint Buffer associated with the Management Endpoint, if applicable.
…	
(N*2+3): (N*2+2)	**Command N:** This field contains the Management Endpoint Buffer Supported Command data structure (refer to Figure 121) for the last command that supports the use of the Management Endpoint Buffer associated with the Management Endpoint, if applicable. N is equal to the value of the NUMCMD field minus 1h.

Figure 121: Management Endpoint Buffer Supported Command Data Structure

Bytes	Description
00	**Command Type (CTYP):** This field specifies the type of command that supports the Management Endpoint Buffer. **Bits** / **Description** 7 – Reserved 6:3 – **NVMe-MI Message Type (NMIMT):** This field specifies the NVMe-MI Message Type. Refer to the NMIMT field shown in Figure 20. 2:0 – Reserved
01	**Opcode (OPC):** This field specifies the opcode of the command that supports the Management Endpoint Buffer.

5.8 Reset

The Reset command is used to initiate the reset defined by the Reset Type field.

The Reset command uses the NVMe Management Dword 0 field. The format of the NVMe Management Dword 0 field is shown in Figure 122. All other command specific fields in the Request Message and Response Message are reserved.

Figure 122: Reset - NVMe Management Dword 0

Bits	Description
31:24	**Reset Type (RSTTYP):** This field specifies the type of reset to be performed. **Value** / **O/M**[1] / **Description** 00h / O/M[2] / Reset NVM Subsystem 01h to FFh / - / Reserved
23:00	Reserved

Notes:

1. O/M definition: O = Optional, M = Mandatory
2. The reset type is required if the NVM Subsystem Reset feature is supported via the NSSR property as defined in the NVM Express Base Specification; else, the reset type is optional.

When a Reset command that specifies a Reset NVM Subsystem in the Reset Type field completes successfully, the NVM Subsystem Reset is initiated (refer to section 8.3). No Success Response is transmitted.

A Management Controller should shutdown all NVMe Controllers in an NVM Subsystem prior to resetting the NVM Subsystem. Refer to the Shutdown command in section 5.11.

5.9 SES Receive

The SES Receive command is used to retrieve SES status type diagnostic pages. Upon successful completion of the SES Receive command, the SES status type diagnostic page is returned in the Response Data.

The SES Receive command uses the NVMe Management Dword 0 field (refer to Figure 123) and the NVMe Management Dword 1 field (refer to Figure 124). There is no Request Data sent in the Request Message.

The Page Code (PCODE) field specifies the SES status type diagnostic page to be retrieved. Refer to SES-4 for a list and description of SES diagnostic pages. If the PCODE field specifies a reserved value, an unsupported value, or a value that only corresponds to an SES control type diagnostic page, then the

Responder responds with an Invalid Parameter Error Response with the PEL field indicating the PCODE field.

The Allocation Length (ALENGTH) field specifies the maximum length of the Response Data field in the Response Message and is used to limit the maximum amount of SES diagnostic page data that may be returned. The length of the Response Data field shall be the total length of the SES diagnostic page specified by the PCODE field or the number of bytes specified by the ALENGTH field (i.e., the SES diagnostic page is truncated), whichever is less. When the SES diagnostic page is truncated, the value of fields within the SES diagnostic page are not altered to reflect the truncation.

All errors are detected and reported while servicing the SES Receive command and reported via an Error Response. If an invalid field is detected in an SES Receive command, then the Responder responds with an Invalid Parameter Error Response with the PEL field indicating the invalid field. If a condition occurs that in SES-4 results in a CHECK CONDITION, then the Responder responds with an Error Response. The mapping of Error Response Status values to SES-4 sense keys and additional sense codes is shown in Figure 13.

If the SES Receive command is supported in the out-of-band mechanism, then the Management Endpoint Buffer shall support the use of the Management Endpoint Buffer with SES Receive command and the size of the Management Endpoint Buffer shall be greater than or equal to the maximum supported SES status type diagnostic page. This allows a Requester to retrieve an SES status type diagnostic page whose size exceed the maximum size allowed by one NVMe-MI Message.

The amount of data returned in the Response Data or transferred to the Management Endpoint Buffer is dependent on the SES status diagnostic page that is returned. The Response Data Length field in the NVMe Management Response contains the length of the Response Data.

Figure 123: SES Receive – NVMe Management Dword 0

Bits	Description
31:8	Reserved
07:00	**Page Code (PCODE):** This field specifies the SES status diagnostic page to be transferred.

Figure 124: SES Receive – NVMe Management Dword 1

Bits	Description
31:16	Reserved
15:00	**Allocation Length (ALENGTH):** This field specifies the maximum length in bytes of the Response Data field in the Response Message.

Figure 125: SES Receive – NVMe Management Response

Bits	Description
23:16	Reserved
15:00	**Response Data Length (RDL):** The length, in bytes, of the Response Data field in this Response Message or transferred to the Management Endpoint Buffer.

5.10 SES Send

The SES Send command is used to transfer SES control type diagnostic pages to an SES Enclosure Service Process. Upon successful completion of the SES Send command, the Request Data, containing an SES control type diagnostic page, is transferred by the Request Message or to the Management Endpoint Buffer.

Unlike the SES Receive command that specifies the page code of the SES status diagnostic page being retrieved, the SES Send command specifies the page code of the SES control type diagnostic page that is being transferred in the SES control type diagnostic page itself. Refer to SES-4 for a list and description of

SES control type diagnostic pages. If the Page Code (PCODE) field in the SES control type diagnostic page specifies a reserved value, an unsupported value, or a value that only corresponds to an SES status diagnostic page, then the Responder responds with an Invalid Parameter Error Response with the PEL field indicating the PCODE field.

The SES Send command does not use the NVMe Management Dword 0 field or the NVMe Management Response field. All of these are reserved.

All errors are detected and reported while processing the SES Send command and reported via an Error Response. If an invalid field is detected in the SES control type diagnostic page data transferred by an SES Send command, then the Responder responds with an Invalid Parameter Error Response with the PEL field indicating the invalid field. If a condition occurs that in SES-4 results in a CHECK CONDITION, then the Responder responds with an Error Response. The mapping of Response Message Status values to SES-4 sense keys and additional sense codes is shown in Figure 13.

The length in bytes of the Request Data field is specified in the Data Length (DLEN) field in the NVMe Management Dword 1 field. An SES Send command with DLEN equal to 0h and no data is valid, and results in a Success Response. If the DLEN field specifies a value that is greater than PAGE LENGTH field in the SES control type diagnostic page plus four, then the extra data in the Request Data field following the page is ignored. If the DLEN field specifies a value that is less than PAGE LENGTH field in the SES control type diagnostic page plus four, then the page is processed using the data contained in the Request Data field.

If the SES Send command is supported in the out-of-band mechanism, then the Responder shall support the use of the Management Endpoint Buffer with the SES Send command and the size of the Management Endpoint Buffer shall be greater than or equal to the maximum supported SES control type diagnostic page. This allows a Requester to transfer an SES control type diagnostic page whose size exceeds the maximum size allowed by one NVMe-MI Message.

Figure 126: SES Send – NVMe Management Dword 1

Bits	Description
31:16	Reserved
15:00	**Data Length (DLEN):** This field specifies the Request Data field in bytes.

5.11 Shutdown

The Shutdown command sent to one Management Endpoint initiates a shutdown on all Controllers in the NVM Subsystem.

The Shutdown command uses the NVMe Management Dword 0 field. The format of the NVMe Management Dword 0 field is shown in Figure 127. All other command specific fields in the Request Message and Response Message are reserved.

Figure 127: Shutdown - NVMe Management Dword 0

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:24</td><td>Shutdown Type (SHDNTYP): This field specifies the type of shutdown to be performed.
<table>
<tr><th>Value</th><th>O/M[1]</th><th>Definition</th></tr>
<tr><td>00h</td><td>O[2]</td><td>Normal NVM Subsystem Shutdown</td></tr>
<tr><td>01h</td><td>O[2]</td><td>Abrupt NVM Subsystem Shutdown</td></tr>
<tr><td>02h to FFh</td><td>-</td><td>Reserved</td></tr>
</table></td></tr>
</table>

Figure 127: Shutdown - NVMe Management Dword 0

Bits	Description
23:00	Reserved
Notes: 1. O/M definition: O = Optional, M = Mandatory 2. Mandatory for the out-of-band mechanism if the NVM Subsystem Shutdown feature is supported on all NVMe Controllers in the NVM Subsystem (refer to the NVM Express Base Specification).	

Upon receipt of a Shutdown command specifying a Normal NVM Subsystem Shutdown, then for each Controller in the NVM Subsystem:

- if:
  - CSTS.SHST is cleared to 00b on that Controller; and
  - An outstanding Asynchronous Event Request command exists on that Controller (refer to the NVM Express Base Specification),

  then the Controller shall issue a Normal NVM Subsystem Shutdown event prior to shutting down the Controller (refer to the NVM Express Base Specification); and
- a normal shutdown is initiated on the Controller as specified by the NVM Express Base Specification.

Upon receipt of a Shutdown command specifying an Abrupt NVM Subsystem Shutdown, then for each Controller in the NVM Subsystem an abrupt shutdown is initiated as specified by the NVM Express Base Specification.

The Shutdown command completes successfully when all NVMe Controllers in the NVM Subsystem report shutdown process complete (i.e., CSTS.SHST is set to 10b and CSTS.ST is set to ‘1’). Refer to the NVM Subsystem Shutdown section of the NVM Express Base Specification for the conditions under which the NVM Subsystem is ready to be powered off.

5.12 VPD Read

The VPD Read command is used to read the Vital Product Data described in section 8.2. Upon successful completion of the VPD Read command, the specified portion of the VPD contents is returned in the Response Data.

The VPD Read command uses the NVMe Management Dword 0 field (refer to Figure 128) and the NVMe Management Dword 1 field (refer to Figure 129). There is no Request Data sent in the Request Message.

If a VPD Read command with the DLEN field cleared to 0h is processed, then the Responder shall respond with a Success Response and no Response Data. If the Data Offset (DOFST) field is greater than or equal to the maximum size of the FRU Information Device, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the DOFST field. If the DOFST field is less than the maximum size of the FRU Information Device and the sum of the DOFST and DLEN fields is greater than the maximum size of the FRU Information Device, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the DLEN field.

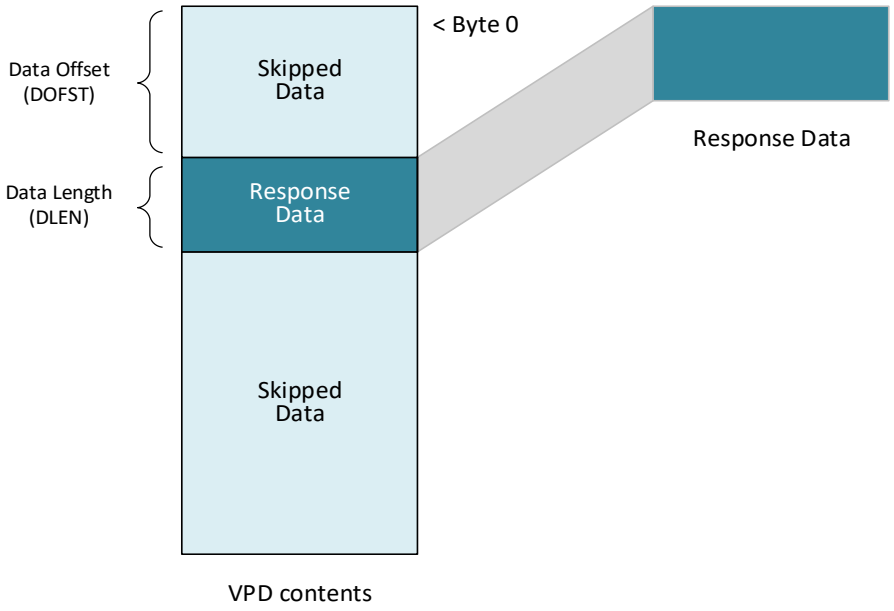
Figure 128: VPD Read NVMe Management Dword 0

Bits	Description
31:16	Reserved
15:00	**Data Offset (DOFST):** This field specifies the starting offset, in bytes, into the VPD data that is contained in the Response Message.

Figure 129: VPD Read NVMe Management Dword 1

Bits	Description
31:16	Reserved
15:00	**Data Length (DLEN):** This field specifies the length, in bytes, to be read from the VPD starting at the byte offset specified by DOFST.

Figure 130: VPD Read Response Data



5.13 VPD Write

The VPD Write command is used to update the Vital Product Data described in section 8.2.

After the VPD Write command has been processed without error, reading the contents of the FRU Information Device directly or a VPD Read command processed without error shall return the new VPD contents (i.e., those supplied with the VPD Write command). The data to be written to the VPD is specified in the Request Data field. The VPD Write command uses the NVMe Management Dword 0 field (refer to Figure 131) and NVMe Management Dword 1 field (refer to Figure 132).

The VPD contents should be capable of being updated at least 8 times using the VPD Write command[1]. If the initial value of the VPD Write Cycles Remaining field is less than 100, then the VPD Write Cycle Remaining Valid bit should be set to '1' (Refer to the VPD Write Cycle Information field in the Identify Controller data structure of the NVM Express Base Specification). If there is an error preventing update of the VPD contents, then the Responder responds with a Generic Error Response and VPD Writes Exceeded status.

A VPD Write command with the Data Length field cleared to 0h and no Request Data is valid. If the Data Length field is cleared to 0h and there is no Request Data, then the Responder responds with a Success Response.

[1] NVM Express Management Interface Specification, Revision 1.0a and prior recommended that VPD contents should be capable of being updated at least 100 times using the VPD Write command.

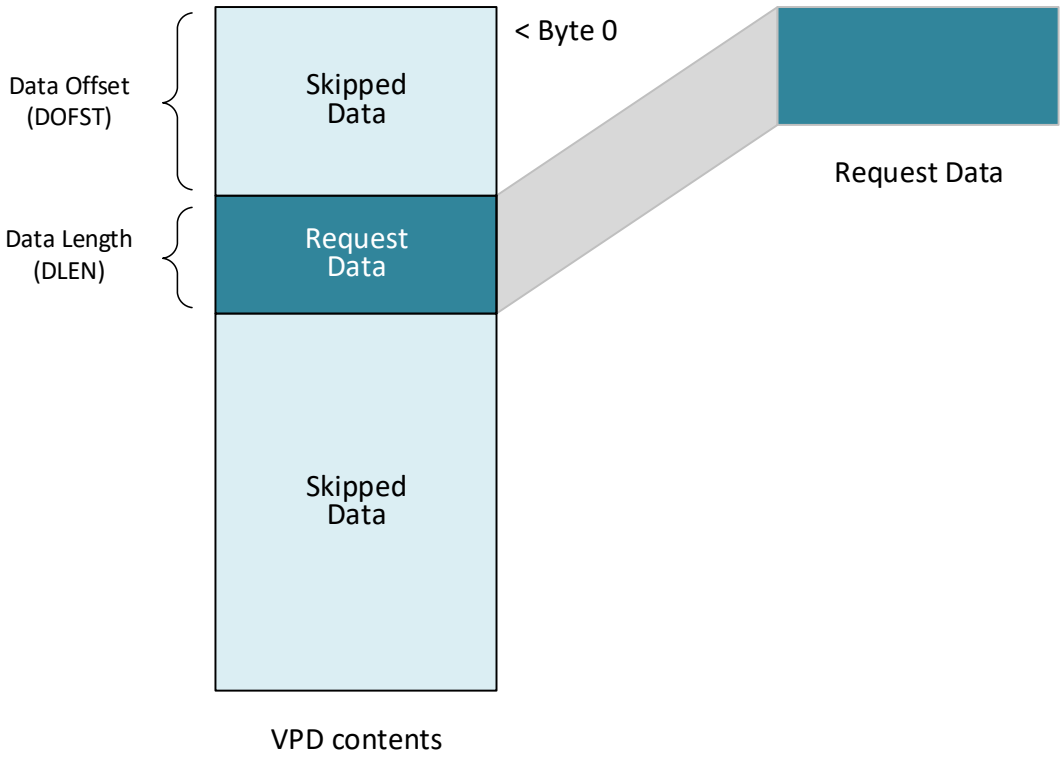
Figure 131: VPD Write – NVMe Management Dword 0

Bits	Description
31:16	Reserved
15:00	**Data Offset (DOFST):** This field specifies the starting offset, in bytes, into the VPD data that is written.

Figure 132: VPD Write – NVMe Management Dword 1

Bits	Description
31:16	Reserved
15:00	**Data Length (DLEN):** This field specifies the length, in bytes, to be written to the VPD starting at the byte offset specified by DOFST.

Figure 133: VPD Write Request Data



The Requester should not read the contents of the VPD while this command is servicing. Reading the contents of the VPD or the processing of a VPD Read command while a VPD Write command is being processed may return incorrect data as a result of the read.

If the Data Offset (DOFST) field is greater than or equal to the maximum size of the FRU Information Device, then the Management Endpoint should not write the contents of the VPD and shall respond with an Invalid Parameter Error Response with the PEL field indicating the DOFST field. If the DOFST field is less than the maximum size of the FRU Information Device and the sum of the DOFST and DLEN fields is greater than the maximum size of the FRU Information Device, then the Management Endpoint shall not write the contents of the VPD and shall respond with an Invalid Parameter Error Response with the PEL field indicating the DLEN field.

6 NVM Express Admin Command Set

The NVM Express Admin Command Set allows NVMe Admin Commands to be issued to any Controller in the NVM Subsystem using the out-of-band mechanism. Figure 134 shows NVMe Admin Commands that are mandatory, optional, and prohibited for an NVMe Storage Device and an NVMe Enclosure using the out-of-band mechanism. All NVMe Admin Commands are prohibited using the in-band tunneling mechanism. The commands are defined in the NVM Express Base Specification and the I/O Command Set specifications. If an NVMe Admin Command is issued in a Request Message that is a prohibited command in Figure 134, the Management Endpoint shall return an Invalid Command Opcode Error Response. The NVM Express Admin Command Set is supported in the out-of-band mechanism and is prohibited in the in-band tunneling mechanism.

Figure 134: List of NVMe Admin Commands Supported using the Out-of-Band Mechanism

Command	Opcode	NVMe Storage Device O/M/P [1]	NVMe Enclosure O/M/P [1]	Reference Specification
Delete I/O Submission Queue	00h	P	P	NVM Express Base Specification
Create I/O Submission Queue	01h	P	P	NVM Express Base Specification
Get Log Page[2]	02h	M	O	NVM Express Base Specification
Delete I/O Completion Queue	04h	P	P	NVM Express Base Specification
Create I/O Completion Queue	05h	P	P	NVM Express Base Specification
Identify	06h	M	O	NVM Express Base Specification
Abort	08h	P	P	NVM Express Base Specification
Set Features	09h	O	O	NVM Express Base Specification
Get Features	0Ah	M	O	NVM Express Base Specification
Asynchronous Event Request	0Ch	P	P	NVM Express Base Specification
Namespace Management	0Dh	O	P	NVM Express Base Specification
Firmware Commit	10h	O	O	NVM Express Base Specification
Firmware Image Download	11h	O	O	NVM Express Base Specification
Device Self-test	14h	O	O	NVM Express Base Specification
Namespace Attachment	15h	O	P	NVM Express Base Specification
Keep Alive	18h	P	P	NVM Express Base Specification
Directive Send	19h	P	P	NVM Express Base Specification
Directive Receive	1Ah	P	P	NVM Express Base Specification
Virtualization Management	1Ch	O	O	NVM Express Base Specification
NVMe-MI Send	1Dh	P	P	NVM Express Base Specification
NVMe-MI Receive	1Eh	P	P	NVM Express Base Specification
Capacity Management	20h	O	P	NVM Express Base Specification
Discovery Information Management	21h	P	P	NVM Express Base Specification
Fabric Zoning Receive	22h	P	P	NVM Express Base Specification
Lockdown	24h	O	O	NVM Express Base Specification
Fabric Zoning Lookup	25h	P	P	NVM Express Base Specification
Clear Exported NVM Resource Configuration	28h	O	O	NVM Express Base Specification
Fabric Zoning Send	29h	P	P	NVM Express Base Specification
Create Exported NVM Subsystem	2Ah	O	O	NVM Express Base Specification
Manage Exported NVM Subsystem	2Dh	O	O	NVM Express Base Specification
Manage Exported Namespace	31h	O	O	NVM Express Base Specification
Manage Exported Port	35h	O	O	NVM Express Base Specification

Figure 134: List of NVMe Admin Commands Supported using the Out-of-Band Mechanism

Command	Opcode	NVMe Storage Device O/M/P [1]	NVMe Enclosure O/M/P [1]	Reference Specification
Cross-Controller Reset	38h	P	P	NVM Express Base Specification
Send Discovery Log Page	39h	P	P	NVM Express Base Specification
Track Send	3Dh	P	P	NVM Express Base Specification
Track Receive	3Eh	P	P	NVM Express Base Specification
Migration Send	41h	P	P	NVM Express Base Specification
Migration Receive	42h	P	P	NVM Express Base Specification
Controller Data Queue	45h	P	P	NVM Express Base Specification
Doorbell Buffer Config	7Ch	P	P	NVM Express Base Specification
Fabrics Commands	7Fh	P	P	NVM Express Base Specification
Format NVM	80h	O	P	NVM Express Base Specification
Security Send	81h	O	P	NVM Express Base Specification
Security Receive	82h	O	P	NVM Express Base Specification
Sanitize	84h	O	O	NVM Express Base Specification
Load Program	85h	P	P	Computational Programs Command Set Specification
Get LBA Status	86h	O	P	NVM Command Set Specification
Program Activation Management	88h	P	P	Computational Programs Command Set Specification
Memory Range Set Management	89h	P	P	Computational Programs Command Set Specification
Sanitize Namespace	8Ch	O	O	NVM Express Base Specification
Vendor Specific	C0h to FFh	O	O	NVM Express Base Specification

Notes:

1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited from being supported. An NVMe Enclosure that is also an NVMe Storage Device (i.e., implements Namespaces):
    - shall implement mandatory commands required for an NVMe Enclosure and may implement optional commands allowed for an NVMe Enclosure; and
    - shall implement mandatory commands required for an NVMe Storage Device and may implement optional commands allowed for an NVMe Storage Device.
2. The Management Endpoint shall ignore the Retain Asynchronous Event (RAE) bit in the Get Log Page command (refer to the NVM Express Base Specification). If the RAE bit is supported and the log page is used with Asynchronous Events, then the Get Log Page command shall be processed as if the RAE bit is set to '1'.

NVMe Admin Commands over the out-of-band mechanism may interfere with a host. A Management Controller should coordinate with the host or issue only NVMe Admin Commands that do not interfere with a host or in-band NVMe commands (e.g., Identify). Coordination between a Management Controller and host is outside the scope of this specification.

The servicing of any NVMe Admin Command over the out-of-band mechanism shall be independent of and not affected by any one or more Controllers in the NVM Subsystem being disabled or being reset by a Controller Level Reset unless the Management Endpoint servicing the NVMe Admin Command is reset (e.g., due to an NVM Subsystem Reset or due to a PCIe Reset of the PCIe VDM Management Endpoint servicing the NVMe Admin Command). Refer to sections 8.1 and 8.5 for more details.

The Request Message format for NVMe Admin Commands is shown in Figure 135 and is described Figure 136.

Figure 135: NVMe Admin Command Request Format

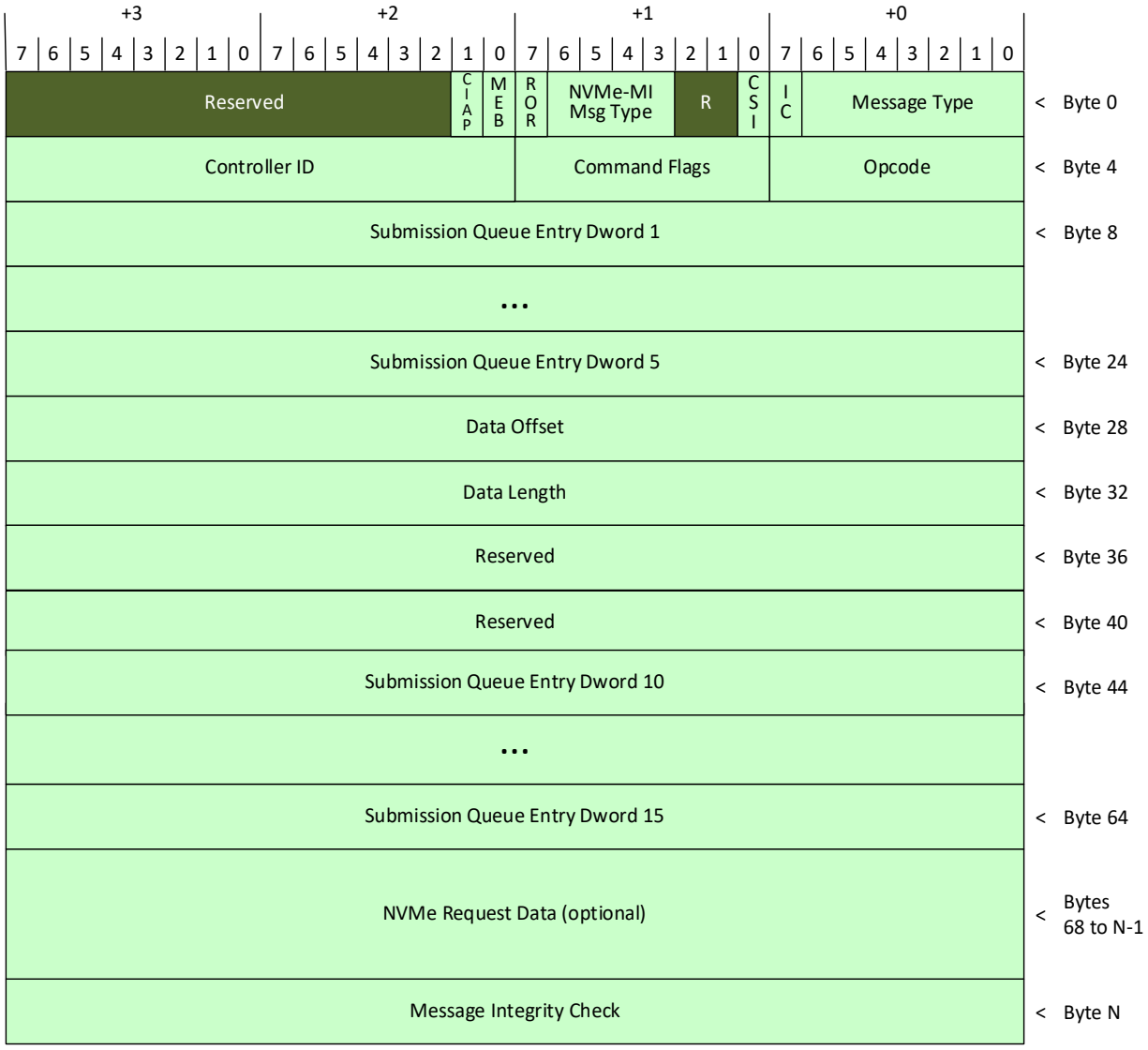


Figure 136: NVMe Admin Command Request Description

Bytes	Description
03:00	**NVMe-MI Message Header (NVMEMH):** Refer to section 3.1.
04	**Opcode (OPC):** This field specifies the opcode of the command. Refer to the NVM Express Base Specification.

Figure 136: NVMe Admin Command Request Description

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>05</td><td>Command Flags (CFLGS): This field specifies flags for the command.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:3</td><td>Reserved</td></tr>
<tr><td>2</td><td>Ignore Shutdown (ISH): The effect of this bit is specified in section 8.5.
This bit shall have no effect on the value of the CSTS.SHST field (refer to the NVM Express Base Specification).</td></tr>
<tr><td>1</td><td>DOFST Valid (DOFSTV): This bit is not used and shall be ignored by the Management Endpoint for implementations compliant with versions of this specification later than 1.1.</td></tr>
<tr><td>0</td><td>DLEN Valid (DLENV): This bit is not used and shall be ignored by the Management Endpoint for implementations compliant with versions of this specification later than 1.1.</td></tr>
</table></td></tr>
<tr><td>07:06</td><td>Controller ID (CTLID): This field specifies the Controller ID of the Controller that this command targets.</td></tr>
<tr><td>11:08</td><td>Submission Queue Entry Dword 1 (SQEDW1): Submission Queue Entry Dword 1 as defined in the NVM Express Base Specification.</td></tr>
<tr><td>15:12</td><td>Submission Queue Entry Dword 2 (SQEDW2): Submission Queue Entry Dword 2 as defined in the NVM Express Base Specification.</td></tr>
<tr><td>19:16</td><td>Submission Queue Entry Dword 3 (SQEDW3): Submission Queue Entry Dword 3 as defined in the NVM Express Base Specification.</td></tr>
<tr><td>23:20</td><td>Submission Queue Entry Dword 4 (SQEDW4): Submission Queue Entry Dword 4 as defined in the NVM Express Base Specification.</td></tr>
<tr><td>27:24</td><td>Submission Queue Entry Dword 5 (SQEDW5): Submission Queue Entry Dword 5 as defined in the NVM Express Base Specification.</td></tr>
<tr><td>31:28</td><td>Data Offset (DOFST): This field specifies the starting offset, in bytes, of the portion of data contained in the NVMe Admin Command completion data that shall be returned starting at byte offset 0h of the NVMe Response Data field in the Response Message for commands that are defined to transfer NVMe Response Data from a Management Endpoint to a Management Controller. This field should be cleared to 0h for all other commands (i.e., commands that are not defined to transmit data and commands that are defined to transfer NVMe Request Data from a Management Controller to a Management Endpoint).
The Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating this field if any of the following conditions are true:
the value of the DOFST field is greater than 0h for commands that are not defined to transfer NVMe Response Data;
the value of the DOFST field is greater than or equal to the size of the NVMe Admin Command completion data; or
bits 1:0 of the DOFST field are not cleared to 00b.</td></tr>
</table>

Figure 136: NVMe Admin Command Request Description

Bytes	Description
35:32	**Data Length (DLEN):** For commands that are defined to transfer Request Data from a Management Controller to a Management Endpoint, this field specifies the length, in bytes, of the data contained in the NVMe Request Data field in the Request Message. For commands that are defined to transfer Response Data from a Management Endpoint to a Management Controller, this field indicates the length, in bytes, of the portion of data contained in the NVMe Admin Command completion data that shall be returned in the NVMe Response Data field in the Response Message. The Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating this field if any of the following conditions are true: • bits 1:0 of the DLEN field are not cleared to 00b; • the value of the DLEN field is greater than 0h for commands that are not defined to transfer NVMe Request Data or NVMe Response Data; • the value of the DLEN field is greater than 4,096; • the sum of the value of the DLEN field plus the value of the DOFST field is greater than the size of the NVMe Admin Command completion data for commands that are defined to transfer NVMe Response Data; • the value of the DLEN field is not equal to the length of the NVMe Request Data field required by the command; or • the DLEN field is cleared to 0h for commands that are defined to transfer NVMe Request Data or NVMe Response Data.
43:36	Reserved
47:44	**Submission Queue Entry Dword 10 (SQEDW10):** Submission Queue Entry Dword 10 as defined in the NVM Express Base Specification.
51:48	**Submission Queue Entry Dword 11 (SQEDW11):** Submission Queue Entry Dword 11 as defined in the NVM Express Base Specification.
55:52	**Submission Queue Entry Dword 12 (SQEDW12):** Submission Queue Entry Dword 12 as defined in the NVM Express Base Specification.
59:56	**Submission Queue Entry Dword 13 (SQEDW13):** Submission Queue Entry Dword 13 as defined in the NVM Express Base Specification.
63:60	**Submission Queue Entry Dword 14 (SQEDW14):** Submission Queue Entry Dword 14 as defined in the NVM Express Base Specification.
67:64	**Submission Queue Entry Dword 15 (SQEDW15):** Submission Queue Entry Dword 15 as defined in the NVM Express Base Specification.
N-1:68	**NVMe Request Data (NVMERD):** This field is optional.
N+3:N	**Message Integrity Check (MIC):** Refer to section 3.1.

The Response Message contains the corresponding format for NVMe Admin Commands is shown in Figure 137 and is described in Figure 138.

Figure 137: NVMe Admin Command Response Format

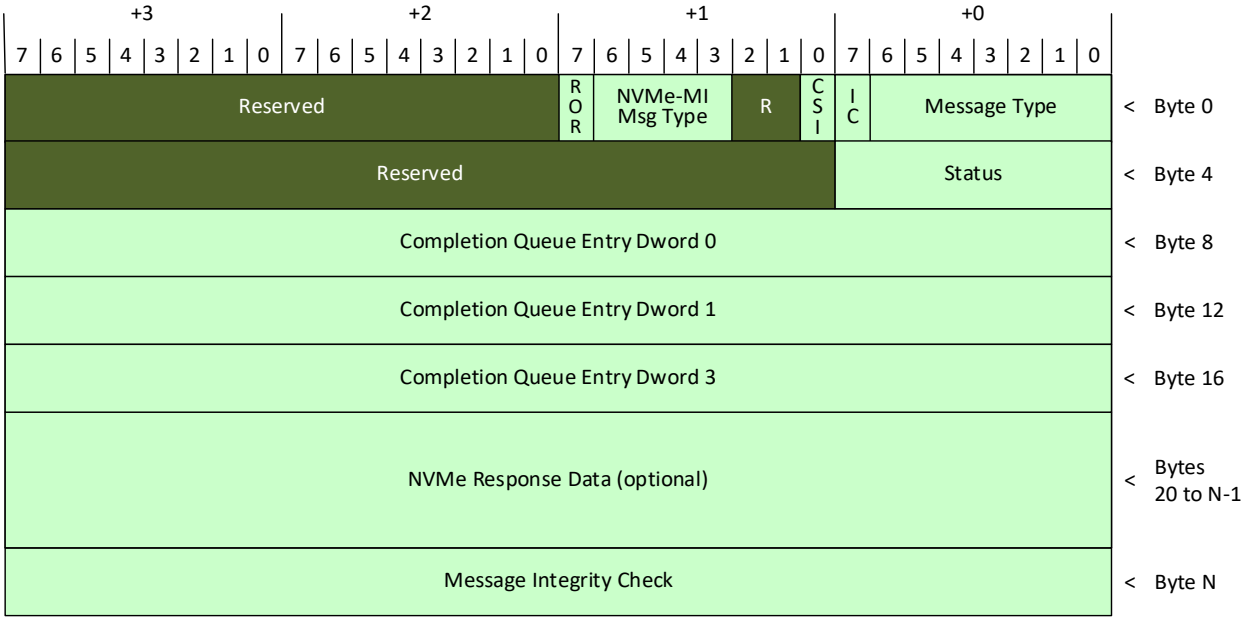


Figure 138: NVMe Admin Command Response Description

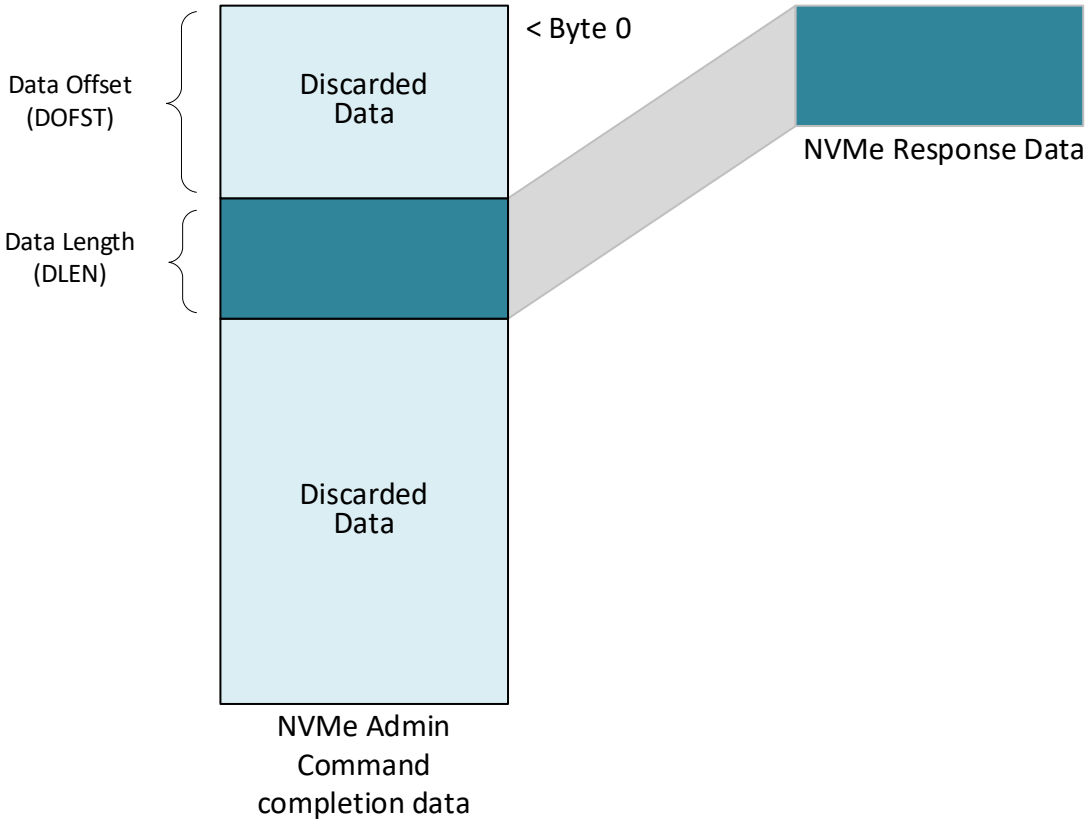
Bytes	Description
03:00	**NVMe-MI Message Header (NMH):** Refer to section 3.1.
04	**Status (STATUS):** This field indicates the status of the NVMe Admin Command. Refer to section 4.1.2.
07:05	Reserved
11:08	**Completion Queue Entry Dword 0 (CQEDW0):** Completion Queue Entry Dword 0 as defined in the NVM Express Base Specification.
15:12	**Completion Queue Entry Dword 1 (CQEDW1):** Completion Queue Entry Dword 1 as defined in the NVM Express Base Specification.
19:16	**Completion Queue Entry Dword 3 (CQEDW3):** Completion Queue Entry Dword 3 as defined in the NVM Express Base Specification. The Command ID field shall be cleared to 0h.
N-1:20	**NVMe Response Data (NVMERD):** This field is optional.
N+3:N	**Message Integrity Check (MIC):** Refer to section 3.1.

6.1 Request and Response Data

NVMe Admin Commands sent using the out-of-band mechanism may contain data as part of the Command Message. This data is passed in the NVMe Request Data field (refer to Figure 137) instead of using PRP Lists or SGL segments. The PRP Entry 2 (PRP2) and Metadata Pointer (MPTR) fields within the NVMe Admin Commands sent using the out-of-band mechanism are reserved.

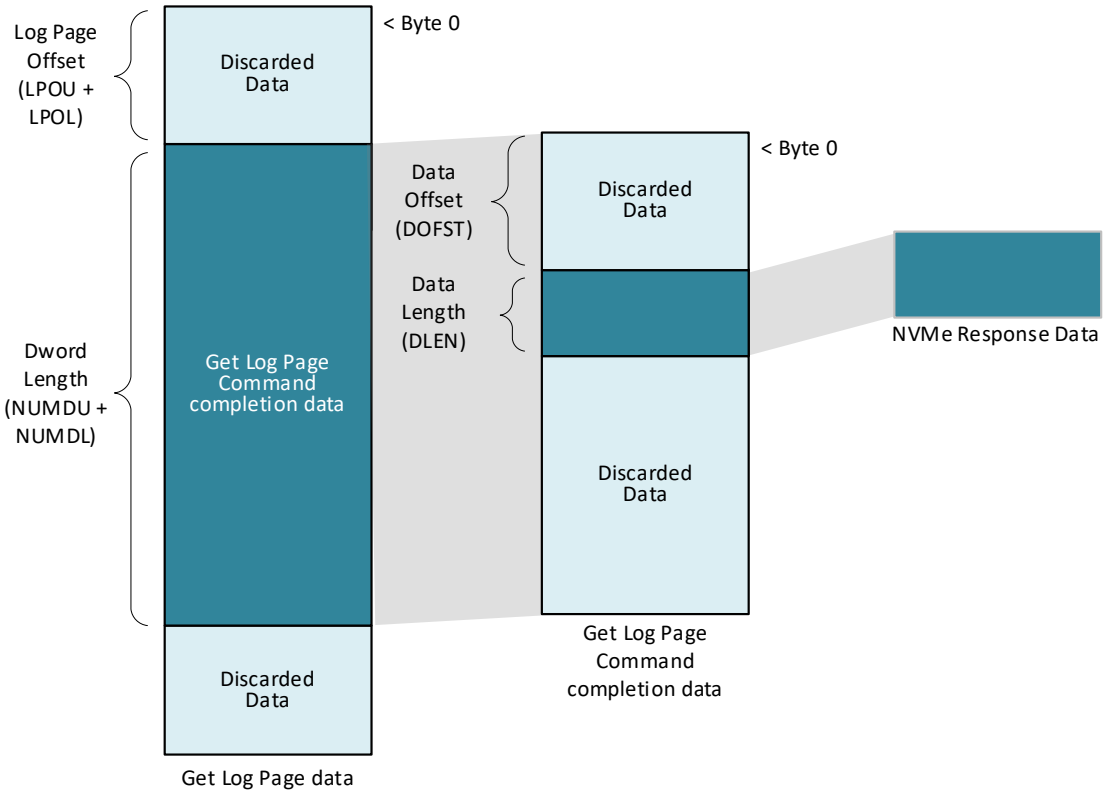
If the NVMe Admin Command is defined to transfer data in the NVMe Response Data field (refer to Figure 137) when sent using the out-of-band mechanism, then the Data Offset and Data Length fields describe the portion of the NVMe Admin Command completion data that is transferred in the NVMe Response Data field. Any remaining data not transferred in the NVMe Response Data field is discarded by the Management Endpoint as shown in Figure 139.

Figure 139: NVMe Admin Command Response Data Example



Some NVMe Admin Commands specify an offset and length which shall be applied first to create the NVMe Admin Command completion data. Then DOFST and DLEN shall be applied to that Admin Command completion data which may further reduce the amount of NVMe Response Data. Figure 140 provides an example for the Get Log Page command.

Figure 140: NVMe Get Log Page Command Response Data Example



6.2 Status

A Response Message for an NVMe Admin Command may contain two status fields. The first status field, contained in Byte 4 of the Response Message, is defined by this specification, and the second Status field, if present, is contained in the Completion Queue Entry Dword 3 field and defined in the NVM Express Base Specification.

An NVMe Admin Command Request Message is well formed if that Request Message does not result in any of the following reported errors:

- Invalid Command Opcode (e.g., the opcode is not listed in Figure 134);
- Invalid Parameter (e.g., the Controller ID field specifies a Controller ID not implemented in the NVM Subsystem);
- Invalid Command Size (e.g., the Request Message does not contain a complete command);
- Invalid Command Input Data Size (e.g., the Request Data field is larger than the size specified in the Data Length field); or
- Access Denied (e.g., the Request Message is prohibited from being executed due to the Command and Feature Lockdown feature (refer to the NVM Express Base Specification)).

If the NVMe Admin Command Request Message is well formed, then a Success Response shall be transmitted. The Success Response shall contain the status associated with NVMe Admin Command in the Status field of the Completion Queue Entry Dword 3 field. The Status field in the Completion Queue

Entry Dword 3 field shall contain any NVM Express Base Specification and I/O Command Set specifications specific status codes (e.g., Successful Completion or Invalid Field in Command).

6.3 Get Log Page

Figure 141 defines the log pages that are mandatory, optional, and prohibited for 2-Wire and PCIe VDM Management Endpoint on NVMe Storage Devices and NVMe Enclosures. The set of optional log pages supported on each Management Endpoint is allowed to differ (refer to NVM Express Base Specification).

Figure 141: Management Endpoint - Log Page Support

Log Page Name[3]	Log Identifier	Support Requirements[1]	
		NVMe Storage Device	NVMe Enclosure
Supported Log Pages	00h	M[2]	M[2]
Error Information	01h	M	M
SMART / Health Information (Controller scope)	02h	M	O
SMART / Health Information (NVM Subsystem scope)		O	O
Firmware Slot Information	03h	M	O
Changed Attached Namespace List	04h	O	O
Commands Supported and Effects	05h	O	O
Device Self-test	06h	O	O
Telemetry Host-Initiated	07h	O	O
Telemetry Controller-Initiated	08h	O	O
Endurance Group Information	09h	O	O
Predictable Latency Per NVM Set	0Ah	O	O
Predictable Latency Event Aggregate	0Bh	O	O
Asymmetric Namespace Access	0Ch	O	O
Persistent Event	0Dh	O	O
LBA Status Information[4]	0Eh	O	O
Endurance Group Event Aggregate	0Fh	O	O
Media Unit Status	10h	O	O
Supported Capacity Configuration List	11h	O	O
Feature Identifiers Supported and Effects	12h	M[2]	O
NVMe-MI Commands Supported and Effects	13h	O	O
Command and Feature Lockdown	14h	O	O
Boot Partition	15h	O	O
Rotational Media Information	16h	O	O
Dispersed Namespace Participating NVM Subsystems	17h	O	O
Management Address List	18h	O	O
Physical Interface Receiver Eye Opening Measurement	19h	O	O
Reachability Groups	1Ah	O	O
Reachability Associations	1Bh	O	O
Changed Allocated Namespace List	1Ch	O	O
Device Personalities	1Dh	O	O
Cross-Controller Reset	1Eh	P	P
Lost Host Communication	1Fh	P	P
FDP Configurations	20h	O	O
Reclaim Unit Handle Usage	21h	O	O
FDP Statistics	22h	O	O
FDP Events	23h	O	O

Figure 141: Management Endpoint - Log Page Support

Log Page Name[3]	Log Identifier	Support Requirements[1]	
		NVMe Storage Device	NVMe Enclosure
Power Measurement	25h	O	O
Discovery	70h	O	O
Host Discovery	71h	O	O
AVE Discovery	72h	O	O
Pull Model DDC Request	73h	O	O
Sanitize Namespace Status List	7Fh	O	O
Reservation Notification	80h	O	O
Sanitize Status	81h	O	O
Program List[6]	82h	O	O
Downloadable Program Types List[6]	83h	O	O
Memory Range Set List[6]	84h	O	O
Changed Zone List[5]	BFh	O	O
Vendor Specific	C0h to FFh	O	O

Notes:
1. O = Optional, M = Mandatory, P = Prohibited.
2. Optional for versions 1.1 and earlier of this specification.
3. Refer to the NVM Express Base Specification unless another footnote specifies otherwise.
4. Refer to the NVM Command Set Specification.
5. Refer to the Zoned Namespace Command Set Specification.
6. Refer to the Computational Programs Command Set Specification.

6.4 Sanitize Operation and Format NVM Command

Figure 142 specifies the Command Messages allowed during a sanitize operation (including an NVM Subsystem sanitize operation and a namespace sanitize operation) and the Command Messages that should be allowed during the processing of a Format NVM command. Refer to the NVM Express Base Specification for the definition of a sanitize operation.

Interactions between NVM Subsystem sanitize operations and the Management Endpoint Buffer are described in section 4.2.3.1.

Figure 142: Command Messages Allowed During a Sanitize Operation and During the Processing of a Format NVM Command

Command Set	Command Message	Allowed
Management Interface Command Set	Configuration Get	Yes
	Configuration Set	
	Controller Health Status Poll	
	Management Endpoint Buffer Read	
	Management Endpoint Buffer Write	
	NVM Subsystem Health Status Poll	
	Read NVMe-MI Data Structure	
	Reset	
	SES Receive	
	SES Send	
	Shutdown	

Figure 142: Command Messages Allowed During a Sanitize Operation and During the Processing of a Format NVM Command

Command Set	Command Message	Allowed
	VPD Read	
	VPD Write	
	Vendor Specific [2]	
NVMe Admin Command Set [1]	Commands listed in Figure 134 as Mandatory or Optional	Same restrictions as defined by the NVM Express Base Specification and any applicable I/O command set specification
PCIe Command Set	PCIe Configuration Read	Yes
	PCIe Configuration Write	
	PCIe I/O Read	
	PCIe I/O Write	
	PCIe Memory Read	
	PCIe Memory Write	

Notes:
1. If the Command Message accesses the Management Endpoint Buffer during a sanitize operation, then the result of that Command Message is undefined as the contents of the Management Endpoint Buffer are cleared to 0h during that sanitize operation (refer to section 4.2.3.1).
2. A vendor-specific command is allowed during a sanitize operation if that command does not alter any user data (refer to the NVM Express Base Specification); otherwise, that command is prohibited.

If an NVM Subsystem sanitize operation is in progress and a Management Endpoint implementation compliant to revision 2.1 or later of this specification processes an Admin command that is prohibited while an NVM Subsystem sanitize operation is in progress, then that Management Endpoint shall return a Success Response with a status code of Sanitize In Progress as defined by the NVM Express Base Specification in the Status field in Completion Queue Entry Dword 3 of the Response Message.

6.5 Set Features and Get Features

Figure 143 defines features that are mandatory or optional for an I/O Controller. Refer to the NVM Express Base Specification for the definition of Set Features and Get Features commands and I/O Controllers.

All Feature Identifiers supported shall be supported if received in-band on an NVMe Controller or received out-of-band on a Management Endpoint unless otherwise stated.

Figure 143: I/O Controller – Feature Support

Feature Name	Feature Support Requirements [1]	Logged in Persistent Event Log [1]
Embedded Management Controller Address	O	O
Host Management Agent Address	O	O
Enhanced Controller Metadata	M	O
Controller Metadata	M	O
Namespace Metadata	M	O

Notes:
1. O = Optional, M = Mandatory

Figure 144 defines features that are mandatory or optional for an Administrative Controller. Refer to the NVM Express Base Specification for the definition of Set Features and Get Features commands and Administrative Controller.

Figure 144: Administrative Controller – Feature Support

Feature Name	Feature Support Requirements[1]	Logged in Persistent Event Log[1]
Embedded Management Controller Address	O	O
Host Management Agent Address	O	O
Enhanced Controller Metadata	M	O
Controller Metadata	M	O
Namespace Metadata	O	O
Notes: 1. O = Optional, M = Mandatory		

Figure 145 defines the features that are mandatory, optional, and prohibited for 2-Wire and PCIe VDM Management Endpoints on NVMe Storage Devices and NVMe Enclosures. The set of optional features supported on each Management Endpoint is allowed to differ (refer to NVM Express Base Specification).

Figure 145: Management Endpoint - Feature Support

Feature Name[2]	Feature Identifier	Support Requirements[1]	
		NVMe Storage Device	NVMe Enclosure
Arbitration	01h	P	P
Power Management	02h	O	O
LBA Range Type[3]	03h	P	P
Temperature Threshold	04h	O	O
Error Recovery[3]	05h	P	P
Volatile Write Cache	06h	P	P
Number of Queues	07h	P	P
Interrupt Coalescing	08h	P	P
Interrupt Vector Configuration	09h	P	P
Write Atomicity Normal[3]	0Ah	P	P
Asynchronous Event Configuration	0Bh	P	P
Autonomous Power State Transition	0Ch	O	O
Host Memory Buffer	0Dh	P	P
Timestamp	0Eh	O	O
Keep Alive Timer	0Fh	P	P
Host Controlled Thermal Management	10h	O	O
Non-Operational Power State Config	11h	O	O
Read Recovery Level Config	12h	P	P
Predictable Latency Mode Config	13h	P	P
Predictable Latency Mode Window	14h	P	P
LBA Status Information Attributes[3]	15h	P	P
Host Behavior Support	16h	P	P
Sanitize Config	17h	O	O
Endurance Group Event Configuration	18h	P	P
I/O Command Set Profile	19h	O	P
Spinup Control	1Ah	O	O

Figure 145: Management Endpoint - Feature Support

Feature Name[2]	Feature Identifier	Support Requirements[1]	
		NVMe Storage Device	NVMe Enclosure
Power Loss Signaling Config	1Bh	O	O
Performance Characteristics	1Ch	O	P
Flexible Data Placement	1Dh	P	P
Flexible Data Placement Events	1Eh	P	P
Namespace Admin Label	1Fh	O	O
Key Value Configuration[4]	20h	O	O
Controller Data Queue	21h	P	P
Configurable Device Personality	22h	O	O
Power Limit	23h	O	O
Power Threshold	24h	O	O
Power Measurement	25h	O	O
Embedded Management Controller Address	78h	O	O
Host Management Agent Address	79h	O	O
Enhanced Controller Metadata	7Dh	M	M
Controller Metadata	7Eh	M	M
Namespace Metadata	7Fh	M	O
Software Progress Marker	80h	P	P
Host Identifier	81h	P	P
Reservation Notification Mask	82h	P	P
Reservation Persistence	83h	P	P
Namespace Write Protection Config	84h	P	P
Boot Partition Write Protection Config	85h	P	P
Vendor Specific	C0h to FFh	O	O

Notes:
1. O = Optional, M = Mandatory, P = Prohibited for Set Features/Optional for Get Features.
2. Refer to the NVM Express Base Specification unless another footnote specifies otherwise.
3. Refer to the NVM Command Set Specification.
4. Refer to the Key Value Command Set Specification.

7 PCIe Command Set (Optional)

The PCIe Command Set defines commands that a Management Controller may submit to access the memory, I/O, and configuration addresses spaces associated with a Controller in the NVM Subsystem. Only addresses mapped to the specified Controller may be accessed (e.g., these commands do not directly access memory on a host). The NMIMT field in the message header for PCIe Command Messages and Response Messages is set to 4h (PCIe Command). The PCIe Command Set is only applicable in the out-of-band mechanism and is prohibited in the in-band tunneling mechanism. The processing of commands in the PCIe Command Set may be affected by the Command and Feature Lockdown feature (refer to the NVM Express Base Specification).

PCIe Commands over the out-of-band mechanism may interfere with the host. A Management Controller should coordinate with the host or issue only PCIe Commands that do not interfere with the host or in-band NVMe commands (e.g., PCIe Configuration Read). Coordination between a Management Controller and a host is outside the scope of this specification.

The servicing of the PCIe Commands specified in Figure 148 shall be independent of and not affected by any one or more Controllers in the NVM Subsystem being (refer to section 8.1 for more details):

- disabled; or
- reset by a Controller Level Reset unless the Management Endpoint servicing the PCIe Command is reset (e.g., due to an NVM Subsystem Reset or due to a PCIe Reset of the PCIe VDM Management Endpoint servicing the PCIe Command), unless otherwise specified.

The Request Message format for PCIe Commands is shown in Figure 146 and described in Figure 147.

Figure 146: PCIe Command Request Format

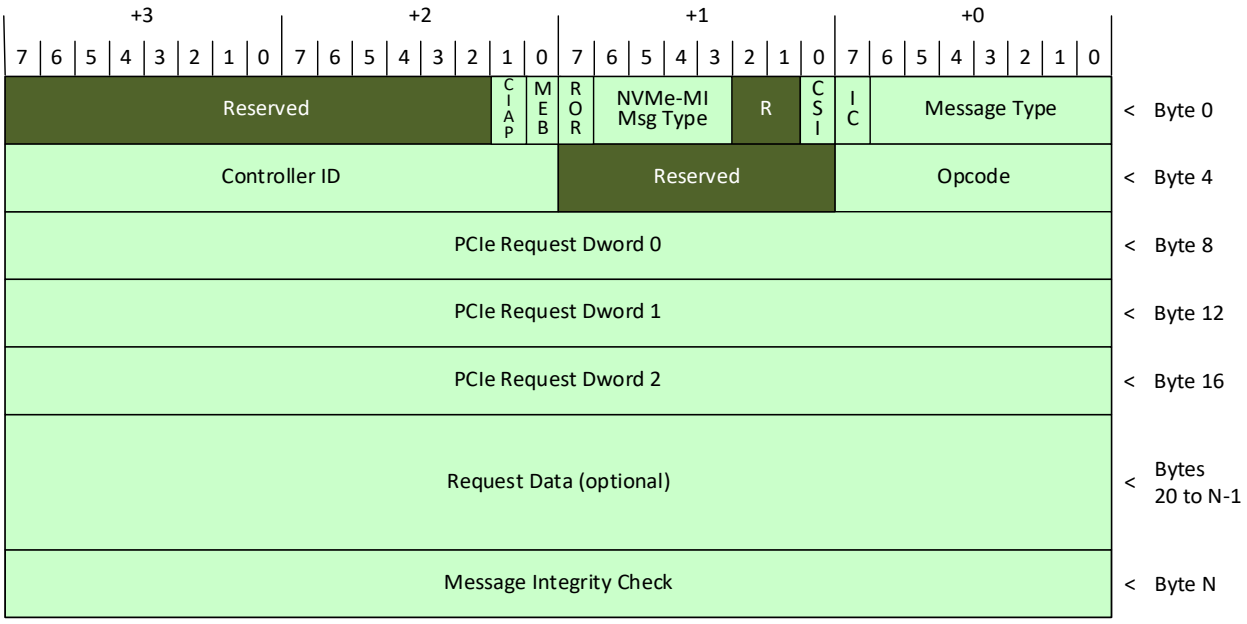


Figure 147: PCIe Command Request Description

Bytes	Description
03:00	**NVMe-MI Message Header (NMH):** Refer to section 3.1.
04	**Opcode (OPC):** This field specifies the opcode of the command to be processed. Refer to Figure 148.
05	Reserved

Figure 147: PCIe Command Request Description

Bytes	Description
07:06	**Controller ID (CTLID):** This field specifies the Controller ID of the NVMe Controller that this command targets.
11:08	**PCIe Request Dword 0 (NMD0):** This field is command specific Dword 0.
15:12	**PCIe Request Dword 1 (NMD1):** This field is command specific Dword 1.
19:16	**PCIe Request Dword 2 (NMD2):** This field is command specific Dword 2.
N-1:20	**Request Data (REQD):** This field is optional.
N+3:N	**Message Integrity Check (MIC):** Refer to section 3.1.

Figure 148 defines the PCIe Command opcodes. It also shows PCIe Commands that are mandatory, optional, and prohibited for an NVMe Storage Device and an NVMe Enclosure using the out-of-band mechanism. All PCIe Commands are prohibited using the in-band tunneling mechanism.

Figure 148: Opcodes for PCIe Commands using an Out-of-Band Mechanism

Opcode	NVMe Storage Device O/M/P[1]	NVMe Enclosure O/M/P[1]	Command
00h	O	O	PCIe Configuration Read
01h	O	O	PCIe Configuration Write
02h	O	O	PCIe Memory Read
03h	O	O	PCIe Memory Write
04h	O	O	PCIe I/O Read
05h	O	O	PCIe I/O Write
06h to FFh	-	-	Reserved

Notes:
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited from being supported. An NVMe Enclosure that is also an NVMe Storage Device (i.e., implements Namespaces):
   - shall implement mandatory commands required for an NVMe Enclosure and may implement optional commands allowed for an NVMe Enclosure; and
   - shall implement mandatory commands required for an NVMe Storage Device and may implement optional commands allowed for an NVMe Storage Device.

The Response Message for PCIe Command is shown in Figure 149 and described in Figure 150.

Figure 149: PCIe Command Response Format

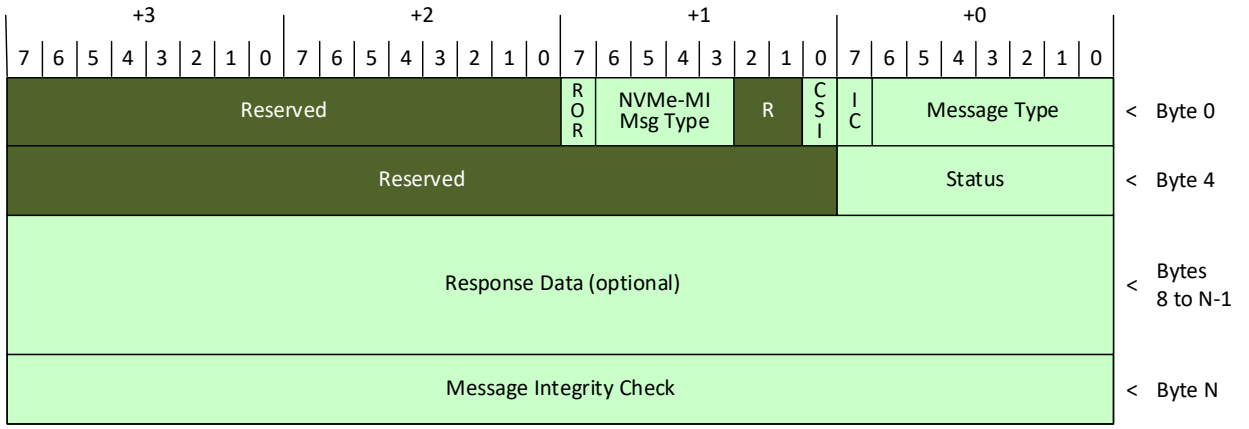


Figure 150: PCIe Command Response Description

Bytes	Description
03:00	**NVMe-MI Message Header (NMH):** Refer to section 3.1.
04	**Status (STATUS):** This field indicates the status of the PCIe Command. Refer to section 4.1.2.
07:05	Reserved
N-1:08	**Response Data (RESPD):** This field is optional.
N+3:N	**Message Integrity Check (MIC):** Refer to section 3.1.

PCIe Commands allow the Management Controller to access PCI Express configuration, I/O, and memory spaces of any Controller in the NVM Subsystem. Support for PCIe Commands is optional and indicated by the Optionally Supported Commands data structure. Refer to Figure 118.

An implementation may support a subset of the PCIe Commands. For supported commands, an implementation may block access to certain address space ranges (e.g., due to security concerns). A PCIe Command that attempts to access such a blocked address range is aborted with the Status field set to Access Denied.

It is recommended that PCIe Commands provide access to all non-blocked address spaces whenever MCTP access is supported. In some implementations, it may not be possible to access PCIe resources in certain states. A PCIe Command processed when a Controller is in one of these states may be aborted with the Status field set to PCIe Inaccessible. Refer to section 8.1.

A PCIe Command that is not well-formed results in an Error Response. A PCIe Command is well formed if that PCIe command does not result in any of the following reported errors:

- Invalid Command Opcode (e.g., the Opcode is not listed in Figure 148);
- Invalid Parameter (e.g., the Controller ID field specifies a Controller ID not implemented in the NVM Subsystem);
- Invalid Command Size (e.g., the Request Message does not contain a complete command); or
- Invalid Command Input Data Size (e.g., the NVMe Request Data field is larger than the size expected by the command).

7.1 PCIe Configuration Read

The PCIe Configuration Read command allows the Management Controller to read the contents of the PCIe configuration address space associated with an NVMe Controller in the NVM Subsystem. The Controller ID field in the Command Message specifies the Controller ID that is being accessed.

The command uses the PCIe Request Dword 0 field (refer to Figure 151) and PCIe Request Dword 1 field refer to Figure 152). The PCIe Request Dword 2 field is not used and is reserved.

Figure 151: PCIe Configuration Read – PCIe Request Dword 0

Bits	Description
31:16	Reserved
15:00	**Length (LENGTH):** This field specifies the number of bytes to be read.

Figure 152: PCIe Configuration Read – PCIe Request Dword 1

Bits	Description
31:12	Reserved
11:00	**Offset (OFFSET):** This field specifies the offset in bytes into the 4 KiB configuration space associated with the NVMe Controller at which the read begins.

When this command is completed successfully, PCI configuration space associated with the NVMe Controller specified by Controller ID is read and returned in the Response Data field. The Offset field specifies the starting read offset in PCIe configuration address space and the Length field specifies the number of bytes to be read. The Response Data field is always an integral number of dwords and is equal to the Length field rounded up to the next dword. If the Length field is not an integral number of dwords, then the read data shall be padded to the next dword boundary with bytes cleared to 0h.

If the sum of the Offset and Length fields fall outside of PCI configuration space, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Offset field.

A Management Endpoint shall support the PCIe Configuration Read command if any of the other PCIe Command Set commands are supported. Access to the BAR offsets shall not return an Access Denied Response Message Status (i.e., the correct data shall be provided).

7.2 PCIe Configuration Write

The PCIe Configuration Write command allows the Management Controller to write the contents of the PCIe configuration address space associated with an NVMe Controller in the NVM Subsystem. The Controller ID field in the Command Message specifies the Controller ID that is being accessed.

The command uses the PCIe Request Dword 0 field (refer to Figure 153) and PCIe Request Dword 1 field (refer to Figure 154). The PCIe Request Dword 2 field is not used and is reserved.

Figure 153: PCIe Configuration Write – PCIe Request Dword 0

Bits	Description
31:16	Reserved
15:00	**Length (LENGTH):** This field specifies the number of bytes to be written.

Figure 154: PCIe Configuration Write – PCIe Request Dword 1

Bits	Description
31:12	Reserved
11:00	**Offset (OFFSET):** This field specifies the offset in bytes into the 4,096B configuration space associated with the NVMe Controller at which the write begins.

When this command is completed successfully, PCI configuration space associated with the NVMe Controller specified by Controller ID is written with the data contained in the Request Data field. The Offset field specifies the starting write offset in PCIe configuration address space and the Length field specifies the number of bytes to be written. The Request Data field is always an integral number of dwords and is equal to the Length field rounded up to the next dword. If the Length field is not an integral number of dwords, then unused padding bytes are discarded.

If the sum of the Offset and Length fields fall outside of PCI configuration space, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Offset field.

7.3 PCIe I/O Read

The PCIe I/O Read command allows the Management Controller to read the contents of PCIe I/O space associated with an NVMe Controller in the NVM Subsystem. The Controller ID field in the Command Message specifies the Controller ID that is being accessed.

The command uses the PCIe Request Dword 0 field (refer to Figure 155) and PCIe Request Dword 1 field (refer to Figure 156). The PCIe Request Dword 2 field is not used and is reserved.

Figure 155: PCIe I/O Read – PCIe Request Dword 0

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:19</td><td>Reserved</td></tr>
<tr><td>18:16</td><td>Base Address Register (BAR): This field specifies the PCI Base Address Register (BAR) of the I/O space to be read. BARs are located beginning at 10h in PCI Configuration space (refer to the NVMe over PCIe Transport Specification) and the value of this field specifies the starting offset of the associated BAR. For a 64-bit BAR, this field should correspond to the least-significant 32-bits of the BAR.
<table>
<tr><th>Value</th><th>BAR Offset</th></tr>
<tr><td>0h</td><td>10h</td></tr>
<tr><td>1h</td><td>14h</td></tr>
<tr><td>2h</td><td>18h</td></tr>
<tr><td>3h</td><td>1Ch</td></tr>
<tr><td>4h</td><td>20h</td></tr>
<tr><td>5h</td><td>24h</td></tr>
<tr><td>6h to 7h</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>15:00</td><td>Length (LENGTH): This field specifies the number of bytes to be read.</td></tr>
</table>

Figure 156: PCIe I/O Read – PCIe Request Dword 1

Bits	Description
31:00	**Offset (OFFSET):** This field specifies the offset in bytes into the PCI BAR associated with the NVMe Controller at which the read begins.

When this command is completed successfully, PCI I/O space associated with the NVMe Controller specified by Controller ID is read and returned in the Response Data field. The Offset field specifies the starting read offset in PCIe I/O address space specified by the Base Address Register field. The Length field specifies the number of bytes to be read. The Response Data field is always an integral number of dwords and is equal to the Length field rounded up to the next dword. If the Length field is not an integral number of dwords, then the read data shall be padded to the next dword boundary with bytes cleared to 0h.

If the Base Address Register field does not correspond to an I/O BAR implemented by the specified NVMe Controller, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Base Address Register field.

If the sum of the Offset and Length fields fall outside the address range of the BAR specified by the Base Address Register field, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Offset field.

7.4 PCIe I/O Write

The PCIe I/O Write command allows the Management Controller to write the contents of PCIe I/O space associated with an NVMe Controller in the NVM Subsystem. The Controller ID field in the Command Message specifies the Controller ID that is being accessed.

The command uses the PCIe Request Dword 0 field (refer to Figure 157) and PCIe Request Dword 1 field (refer to Figure 158). The PCIe Request Dword 2 field is not used and is reserved.

Figure 157: PCIe I/O Write – PCIe Request Dword 0

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:19</td><td>Reserved</td></tr>
<tr><td>18:16</td><td>Base Address Register (BAR): This field specifies the PCI Base Address Register (BAR) of the I/O space to be written. BARs are located beginning at 10h in PCI Configuration space (refer to the NVMe over PCIe Transport Specification) and the value of this field specifies the starting offset of the associated BAR. For a 64-bit BAR, this field should correspond to the least-significant 32-bits of the BAR.
<table>
<tr><th>Value</th><th>BAR Offset</th></tr>
<tr><td>0h</td><td>10h</td></tr>
<tr><td>1h</td><td>14h</td></tr>
<tr><td>2h</td><td>18h</td></tr>
<tr><td>3h</td><td>1Ch</td></tr>
<tr><td>4h</td><td>20h</td></tr>
<tr><td>5h</td><td>24h</td></tr>
<tr><td>6h to 7h</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>15:00</td><td>Length (LENGTH): This field specifies the number of bytes to be written.</td></tr>
</table>

Figure 158: PCIe I/O Write – PCIe Request Dword 1

Bits	Description
31:00	**Offset (OFFSET):** This field specifies the offset in bytes into the PCI BAR associated with the NVMe Controller at which the write begins.

When this command is completed successfully, PCI I/O space associated with the NVMe Controller specified by Controller ID is written with the data contained in the Request Data field. The Offset field specifies the starting write offset in PCIe I/O address space specified by the Base Address Register field. The Length field specifies the number of bytes to be written. The Request Data field is always an integral number of dwords and is equal to the Length field rounded up to the next dword. If the Length field is not an integral number of dwords, then unused padding bytes are discarded.

If the Base Address Register field does not correspond to an I/O BAR implemented by the specified NVMe Controller, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Base Address Register field.

If the sum of the Offset and Length fields fall outside the address range of the BAR specified by the Base Address Register field, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Offset field.

7.5 PCIe Memory Read

The PCIe Memory Read command allows the Management Controller to read the contents of PCIe memory associated with an NVMe Controller in the NVM Subsystem. The Controller ID field in the Command Message specifies the Controller ID that is being accessed.

The command uses the PCIe Request Dword 0 field (refer to Figure 159), PCIe Request Dword 1 field (refer to Figure 160), and PCIe Request Dword 2 field (refer to Figure 161).

Figure 159: PCIe Memory Read – PCIe Request Dword 0

Bits	Description
31:19	Reserved

Figure 159: PCIe Memory Read – PCIe Request Dword 0

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>18:16</td><td>Base Address Register (BAR): This field specifies the PCI Base Address Register (BAR) of the memory space to be read. BARs are located beginning at 10h in PCI Configuration space (refer to the NVMe over PCIe Transport Specification) and the value of this field specifies the starting offset of the associated BAR. For a 64-bit BAR, this field should correspond to the least-significant 32-bits of the BAR.
<table>
<tr><th>Value</th><th>BAR Offset</th></tr>
<tr><td>0h</td><td>10h</td></tr>
<tr><td>1h</td><td>14h</td></tr>
<tr><td>2h</td><td>18h</td></tr>
<tr><td>3h</td><td>1Ch</td></tr>
<tr><td>4h</td><td>20h</td></tr>
<tr><td>5h</td><td>24h</td></tr>
<tr><td>6h to 7h</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>15:00</td><td>Length (LENGTH): This field specifies the number of bytes to be read.</td></tr>
</table>

Figure 160: PCIe Memory Read – PCIe Request Dword 1

Bits	Description
31:00	**Offset Lower (OFFSETL):** This field specifies the least-significant 32-bits (i.e., bits [31:0]) of the offset in bytes into the PCI BAR associated with the NVMe Controller at which the read begins.

Figure 161: PCIe Memory Read – PCIe Request Dword 2

Bits	Description
31:00	**Offset Upper (OFFSETU):** This field specifies the most-significant 32-bits (i.e., bits [63:32]) of the offset in bytes into the PCI BAR associated with the NVMe Controller at which the read begins.

When this command is completed successfully, PCI memory space associated with the NVMe Controller specified by Controller ID is read and returned in the Response Data field. The Offset field specifies the starting read offset in PCIe memory address space specified by the Base Address Register field. The Length field specifies the number of bytes to be read. The Response Data field is always an integral number of dwords and is equal to the Length field rounded up to the next dword. If the Length field is not an integral number of dwords, then the read data shall be padded to the next dword boundary with bytes cleared to 0h.

If the Base Address Register field does not correspond to one implemented by the specified NVMe Controller, or the address range specified by the Base Address Range is not a memory region, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Base Address Register field.

If the sum of the Offset and Length fields fall outside the address range specified by the Base Address Register field, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Offset field.

7.6 PCIe Memory Write

The PCIe Memory Write command allows the Management Controller to write the contents of PCIe memory associated with an NVMe Controller in the NVM Subsystem. The Controller ID field in the Command Message specifies the Controller ID that is being accessed.

The command uses the PCIe Request Dword 0 field (refer to Figure 162), PCIe Request Dword 1 field (refer to Figure 163), and PCIe Request Dword 2 field (refer to Figure 164).

Figure 162: PCIe Memory Write – PCIe Request Dword 0

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:19</td><td>Reserved</td></tr>
<tr><td>18:16</td><td>Base Address Register (BAR): This field specifies the PCI Base Address Register (BAR) of the memory space to be written. BARs are located beginning at 10h in PCI Configuration space (refer to the NVMe over PCIe Transport Specification) and the value of this field specifies the starting offset of the associated BAR. For a 64-bit BAR, this field should correspond to the least-significant 32-bits of the BAR.
<table>
<tr><th>Value</th><th>BAR Offset</th></tr>
<tr><td>0h</td><td>10h</td></tr>
<tr><td>1h</td><td>14h</td></tr>
<tr><td>2h</td><td>18h</td></tr>
<tr><td>3h</td><td>1Ch</td></tr>
<tr><td>4h</td><td>20h</td></tr>
<tr><td>5h</td><td>24h</td></tr>
<tr><td>6h to 7h</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>15:00</td><td>Length (LENGTH): This field specifies the number of bytes to be written.</td></tr>
</table>

Figure 163: PCIe Memory Write – PCIe Request Dword 1

Bits	Description
31:00	**Offset (OFFSET):** This field specifies the least-significant 32-bits (i.e., bits [31:0]) of the offset in bytes into the PCI BAR associated with the NVMe Controller at which the write begins.

Figure 164: PCIe Memory Write – PCIe Request Dword 2

Bits	Description
31:00	**Offset (OFFSET):** This field specifies the most-significant 32-bits (i.e., bits 63:32]) of the offset in bytes into the PCI BAR associated with the NVMe Controller at which the write begins.

When this command is completed successfully, PCI memory space associated with the NVMe Controller specified by Controller ID is written with the data contained in the Request Data field. The Offset field specifies the starting write offset in PCIe memory address space specified by the Base Address Register field. The Length field specifies the number of bytes to be written. The Request Data field is always an integral number of dwords and is equal to the Length field rounded up to the next dword. If the Length field is not an integral number of dwords, then unused padding bytes are discarded.

If the Base Address Register field does not correspond to one implemented by the specified NVMe Controller, or the address range specified by the Base Address Range is not a memory region, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Base Address Register field.

If the sum of the Offset and Length fields fall outside the address range of the BAR specified by the Base Address Register field, then the Management Endpoint shall respond with an Invalid Parameter Error Response with the PEL field indicating the Offset field.

8 Management Architecture

8.1 Out-of-Band Operational Times

In the out-of-band mechanism, the ability of a Management Endpoint to receive and process Request Messages outlined in this specification is dependent on the state of the Management Endpoint. This section enumerates Management Endpoint, FRU Information Device, and 2-Wire Mux operational times and the operations supported in each of these operational times.

The NVM Subsystem power state is defined by the state of main power and auxiliary power. Main power consists of one or more voltage rails as defined by form factor. When main power consists of multiple voltage rails, main power is considered “on” when power is good on all main voltage rails. Auxiliary power is optionally supported by a form factor and enables 2-Wire communications in the absence of main power. Only the Powered On and Powered Off states are applicable in form factors and platforms that do not support auxiliary power. Figure 165 defines the power states of a Management Endpoint. Note that auxiliary power is described from the perspective of the NVM Subsystem and is able to be provided by any appropriate power rail in a host platform.

The operations supported in each NVM Subsystem power state are summarized in Figure 165. SMBus/I2C VPD access consists of processing read operations to the FRU Information Device. 2-Wire Mux access consists of processing operations to the 2-Wire Mux. 2-Wire MCTP access consists of processing and responding to NVMe-MI Messages on the NVM Subsystem 2-Wire port. PCIe MCTP access consists of processing and responding to NVMe-MI Messages issued on any NVM Subsystem PCIe port. The behavior of an operation that is “Not Supported” in Figure 165 is undefined.

Figure 165: Operations Supported During NVM Subsystem Power States

Operation[4, 5]	**Powered Off**	**Powered On**	**Auxiliary Power Only**[2]	**Main Power Only**[2]
Main Power	**Off**	**On**	**Off**	**On**
Auxiliary Power[2]	**Off or Not Supported**	**On or Not Supported**	**On**	**Off**
2-Wire VPD and 2-Wire Mux Access	Not Supported	Supported[3]	Supported[3]	Implementation Specific
2-Wire MCTP Access	Not Supported	Supported[3]	Optional[1]	Implementation Specific[3]
PCIe MCTP Access	Not Supported	Supported[3]	Not Supported	Supported[3]

Notes:
1. An implementation that supports 2-Wire MCTP Access during Auxiliary Power may support a subset of the commands supported during the Power On state. If a subset of commands is supported, then the subset shall include VPD Read. Power states that support MCTP over I3C shall also support MCTP over SMBus and both transports shall support the same set of commands.
2. The form factor defines whether Auxiliary power is supported. This means that Auxiliary Power Only and Main Power Only columns are not applicable to form factors that do not support Auxiliary power.
3. For interactions with Power Loss Signaling processing, refer to section 8.1.2.
4. A 2-Wire Reset may prevent access as described in section 8.3.4. A PCIe Reset may prevent access as described in this section, section 8.3.2, and section 8.3.5.
5. Firmware activation may impact access as described in section 8.3.2.

Within 1 s after an NVM Subsystem transitions from a power state in which accesses are not supported to one where accesses are supported, the accesses listed in Figure 165 are operational.

Once operational, SMBus/I2C VPD accesses and 2-Wire Mux accesses shall be processed unless otherwise noted. SMBus/I2C VPD accesses 2-Wire Mux accesses are permitted to be unsupported once operational during the following cases:

- a firmware activation without reset (i.e., the Commit Action field in the Firmware Commit command is set to 011b) is being processed (refer to the NVM Express Base Specification);
- a VPD Read command is being processed;
- a VPD Write command is being processed; or
- while updating the Boot Failure Code field or Common Header Checksum field due to a boot failure detected while operational (refer to Figure 169).

If SMBus/I2C VPD accesses or 2-Wire Mux accesses are not supported once operational, then the FRU Information Device or 2-Wire Mux, respectively, shall be hidden from the Management Controller on the 2-Wire port (e.g., by detaching the FRU Information Device or 2-Wire Mux from the Management Controller-facing 2-Wire port). While the FRU Information Device or 2-Wire Mux is hidden from the Management Controller, SMBus/I2C VPD accesses or 2-Wire Mux accesses from the Management Controller shall be NACKed.

If SMBus/I2C VPD accesses or 2-Wire Mux accesses are not supported once operational, then it is strongly recommended that the amount of time that accesses are unsupported be minimized. If these accesses are not supported once operational, then a Management Controller may time out and report the NVM Subsystem as failed.

Once operational, all other accesses other than SMBus/I2C VPD accesses and 2-Wire Mux accesses should be processed. For example, an SMBus/I2C VPD or 2-Wire Mux access issued 1 s after transitioning from a “Powered Off” to a “Main Power” state is guaranteed to be processed and an MCTP access should be processed. The behavior of accesses prior to this 1 s time interval is undefined. For example, the behavior of a 2-Wire MCTP access issued 50 ms after transitioning from a “Powered Off” to a “Main Power” state is undefined.

If a Request Message is received greater than or equal to 1 s after entering a power state in which MCTP accesses are supported from a power state in which MCTP access are not supported and the Management Endpoint is not ready to process the Request Message, then the Management Endpoint should return a More Processing Required Response (refer to section 4.1.2.3). Upon entering a power state in which MCTP accesses are supported from a power state in which MCTP access are not supported, the maximum amount of time a Management Endpoint is ready to start processing a Request Message that:

- does not require media access is indicated by the Management Endpoint Ready Independent of Media Timeout (MERIMTO) field; and
- requires media access is indicated by the Management Endpoint Ready With Media Timeout (MERWMTO) field.

In certain cases, upon entering a power state in which MCTP accesses are supported from a power state in which MCTP access are not supported, the Management Endpoint may be unresponsive and unable to service a Request Message or return any Response Message, including a More Processing Required Response Message (e.g., due to activating a firmware image or due to executing code in a security module that prohibits execution of code outside the security module that is required to process the Request Message). Upon entering a power state in which MCTP accesses are supported from a power state in which MCTP access are not supported, the Management Endpoint shall not be unresponsive for more than the amount of time indicated by the Maximum Unresponsive Time field. Any Request Message received during the time a Management Endpoint is unresponsive shall be silently discarded.

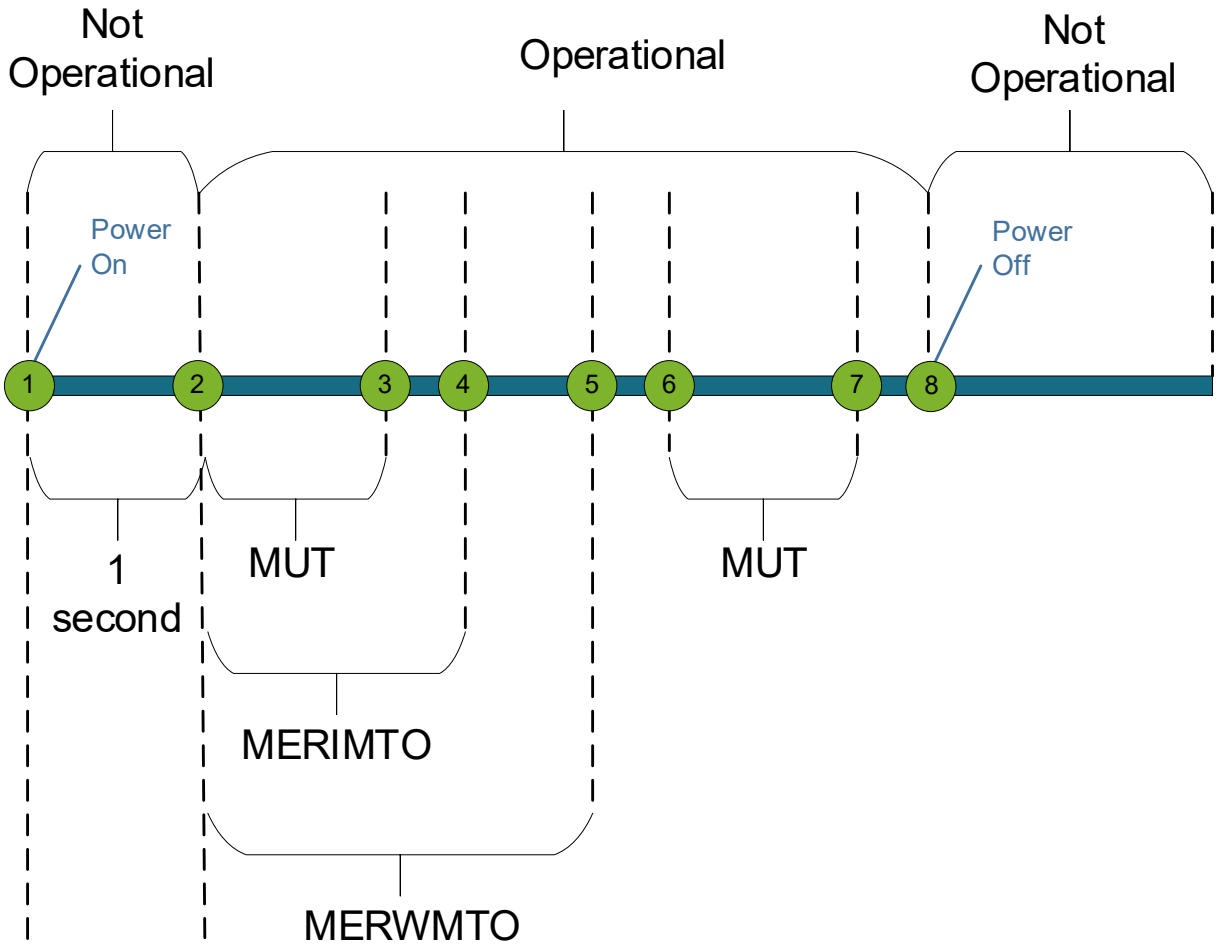
Note that it is strongly recommended that the amount of time a Management Endpoint is unresponsive be minimized. If a Management Endpoint is unresponsive, then Management Controllers compliant with revisions of this specification prior to the definition of the Maximum Unresponsive Time field may time out and report the NVM Subsystem as failed.

When transitioning between power states in which accesses are supported in both states (i.e., the state before and after the transition), there is no interruption in access processing (i.e., accesses are processed prior to the state transition, during the state transition, and immediately after entering the new power state).

An example operational timing diagram is shown in Figure 166. In this example, there are no shutdowns of any Controller or of the NVM Subsystem. And in this example, the value in the MUT field is smaller than the value in the MERIMTO field which is smaller than the value in the MERWMTO field, but this ordering is not required. The sequence of events in Figure 166 is as follows:

1. The NVM Subsystem transitions from a power state in which accesses to a Management Endpoint, FRU Information Device, and/or 2-Wire Mux are not supported to one where accesses are supported. After entering the power state in which accesses are supported, any Management Endpoint, FRU Information Device, or 2-Wire Mux in the NVM Subsystem is permitted to be non-operational for a maximum time of 1 s. While non-operational, none of the operations in Figure 165 are supported and if a Management Controller performs any of those operations, then the results are undefined.
2. Within 1 s after transitioning from a power state in which accesses are not supported to one where accesses are supported, SMBus/I2C VPD and 2-Wire Mux accesses are processed, unless otherwise specified, and MCTP accesses should be processed.
    a. If the Management Endpoint is not ready to process an MCTP access that does not require media access, then the Management Endpoint is permitted to return a More Processing Required Response for up to the amount of time indicated by the MERIMTO field.
    b. If the Management Endpoint is not ready to process an MCTP access that requires media access, then the Management Endpoint is permitted to return a More Processing Required Response for up to the amount of time indicated by the MERWMTO field.
    c. If the Management Endpoint is unresponsive and unable to service a Request Message or return any Response Message, including a More Processing Required Response Message (e.g., due to activating a firmware image or due to executing code in a security module that prohibits execution of code outside the security module that is required to process the Request Message), then the Management Endpoint discards any received Request Messages.
3. Within the amount of time indicated by the MUT field since entering the operational state, the Management Endpoint is no longer permitted to be unresponsive to MCTP accesses.
4. Within the amount of time indicated by the MERIMTO field since entering the operational state, the Management Endpoint is ready to process Request Messages that do not require access to media.
5. Within the amount of time indicated by the MERWMTO field since entering the operational state, the Management Endpoint is ready to process Request Messages that require access to media.
6. While operational, if the Management Endpoint is unresponsive and unable to service a Request Message or return any Response Message, including a More Processing Required Response Message (e.g., due to activating a firmware image without reset), then the Management Endpoint discards any received Request Messages. While operational, if SMBus/I2C VPD accesses or 2-Wire Mux accesses are not supported, then the SMBus/I2C VPD accesses or 2-Wire Mux accesses are NACKed.
7. While operational, the Management Endpoint is unresponsive to MCTP accesses, SMBus/I2C accesses, and 2-Wire Mux access for no longer than the amount of time indicated by the MUT field.
8. The NVM Subsystem transitions from a power state in which accesses to a Management Endpoint, FRU Information Device, and/or 2-Wire Mux are supported to one where accesses are not supported. The behavior of any access from the Management Controller that is not supported in the current power state is undefined.

Figure 166: Operational Time Example Timing Diagram



Key:
MUT: Maximum Unresponsiveness Time field value
MERIMTO: Management Endpoint Read Independent of Media Timeout field value
MERWMTO: Management Endpoint Read With Media Timeout field value

8.1.1 Controller Disable and Reset Interactions

The enable/disable state of any Controller in the NVM Subsystem (refer to the CC.EN bit in the NVM Express Base Specification) shall have no effect on any operations in Figure 165. For example, in a power state where 2-Wire MCTP access or PCIe MCTP access is supported, it is not an error to submit a Request Message to a disabled Controller, including Request Messages such as NVMe Admin Commands that target the disabled Controller (e.g., the Management Endpoint processes an NVMe Admin Command the same as if the Controller were enabled). Likewise, disabling a Controller while one or more operations from Figure 165 are being serviced, including Request Messages targeting the Controller being disabled, shall have no effect on the servicing of those operations.

Controller Level Resets shall have no impact on any operations in Figure 165 except for the impacts to PCIe MCTP accesses to PCIe VDM Management Endpoints due to PCIe Resets (i.e., Conventional Reset or Function Level Reset) described in section 8.3.5 and the impacts to PCIe Command processing described in this section. For example, in a power state where 2-Wire MCTP access is supported and unless otherwise specified, it is not an error to submit a Request Message via MCTP over 2-Wire to a Controller in a PCIe Reset asserted state including Request Messages that target the Controller in the PCIe

Reset asserted state such as NVMe Admin Commands (e.g., the Management Endpoint processes an NVMe Admin Command the same as if the Controller was not in a PCIe Reset asserted state). Likewise, unless otherwise specified, asserting Controller Level Reset to a Controller while one or more Request Messages are being serviced, including Request Messages targeting the Controller being reset, shall have no effect on the servicing of any Request Message.

Unless otherwise specified, if the NVM Subsystem is in a power state in which an operation in Figure 165 is supported, then the Management Endpoint shall complete any steps required to be able to successfully handle the operation and then handle the operation. For example, if an NVMe Admin Command targeting a Controller that is in normal operation (i.e., the value of the CSTS.SHST field is cleared to 00b) and has not been shut down requires media access and media has not been initialized, then the Management Endpoint shall initialize media and then the NVMe Admin Command shall be processed. If an operation is able to be processed successfully but the Management Endpoint instead returns an error (e.g., an Error Response for any Command Message or an error status code in the Status field in CQEDW3 in an NVMe Admin Command Response) due to any reason including not attempting to complete any steps necessary to successfully complete the operation (e.g., initializing media), then the Management Controller may erroneously flag the NVM Subsystem as failed.

Although not recommended, an implementation may choose not to support processing of PCIe Commands that target a Controller in the NVM Subsystem that is in any of the following states:[1]

- Controller Level Reset that does not reset the Management Endpoint servicing the PCIe Command;
- SR-IOV Virtual Function is not enabled;
- during any type of PCI Express Conventional Reset, for PCIe Commands received via MCTP over 2-Wire;
- during a PCI Express Function Level Reset (FLR);
- when the PCI Express Function is in a non-D0 power D-state; or
- when the PCI Express link is down (i.e., not in the DL_Active state).

If a PCIe Command is received that targets a Controller in one of these states and the implementation does not support processing of PCIe Commands in that state, then the PCIe command is completed with status PCIe Inaccessible. The Controller shall not complete PCIe Commands with a status of PCIe Inaccessible in all other Controller states.

If a PCIe Command is received that targets a Controller whose corresponding PCIe link is in a low power state (i.e., PCIe ASPM), then processing of the command may cause the link to temporarily exit the low power state.

8.1.2 Power Loss Signaling Interactions

A Controller which responds to a Power Loss Signaling notification performs either Forced Quiescence Processing or Emergency Power Fail Processing (refer to the Power Loss Signaling section of the NVM Express Base Specification).

If one or more Controllers in an NVM Subsystem are in the:

a) FQ Processing state;
b) FQ Complete state;
c) EPF Processing Port Enabled state;
d) EPF Complete Port Enabled state;
e) EPF Processing Port Disabled state; or

[1] A Management Controller should only send these commands using SMBus/I2C or another PCIe port since the link associated with the PCIe port and Controller is down in these states.

f) EPF Complete Port Disabled state,

then:

a) all Command Slots in all Management Endpoints in that NVM Subsystem should:
    1. behave as if an implicit Abort Control Primitive (refer to section 4.2.1.3) was received with the exception that the Management Endpoint shall not transmit the Abort Control Primitive Response Messages; and
    2. drop (silently discard) Control Primitives;
b) access to an SMBus/I2C VPD in that NVM Subsystem may or may not be supported; and
c) access to the 2-Wire Mux in that NVM Subsystem may or may not be supported.

When all Controllers in an NVM Subsystem have transitioned out of the FQ Complete state because the PLN variable transitioned to Deasserted, then:

a) the Management Endpoint shall service Request Messages;
b) access to the SMBus/I2C VPD shall be supported; and
c) access to the 2-Wire Mux shall be supported.

8.2 Vital Product Data

The Vital Product Data (VPD) is FRU Information (refer to the IPMI Platform Management FRU Information Storage Specification) describing an NVMe Storage Device. Each NVMe Storage Device FRU shall have a FRU Information Device with a size of 256 to 65,536 bytes which contains the VPD. The VPD for NVMe Storage Device FRUs shall contain the required elements defined in Figure 167. The VPD and FRU Information Device are optional for:

a) NVMe Storage Devices that are not FRUs (e.g., NVMe Storage Devices with a Form Factor type of Integrated per Figure 180); and
b) NVMe Enclosures.

The VPD contents for these optional use cases is outside the scope of this specification.

Figure 167: VPD Elements

Bytes	Name
7:0	Common Header
Vendor Specific	Product Info Area (Optional)
Vendor Specific	MultiRecord Info Area
Vendor Specific	Internal Use Area (Optional)
Vendor Specific	Chassis Info Area (Optional)
Vendor Specific	Board Info Area (Optional)

The VPD shall be accessible using the VPD Read command on all Management Endpoints on the NVMe Storage Device FRU. The entire contents of the VPD may be updated using the VPD Write command.

If the NVM Subsystem has a 2-Wire port and the 2-Wire port is in SMBus mode, then the VPD shall be accessible at the SMBus/I2C address of the FRU Information Device using I2C Reads. Updating the VPD using I2C Writes shall not be supported if the VPD Write command is supported. Refer to the IPMI Platform Management FRU Information Storage Definition for more information about the FRU Information Device access mechanisms (I2C Reads/I2C Writes).

If the NVM Subsystem has a 2-Wire port and the 2-Wire port is in I3C mode, then the FRU Information Device is not accessible using I2C Reads. In this case, the VPD is accessible with the VPD Read command if the VPD Read command is supported in the current NVM Subsystem power state.

Figure 168: I2C Read from a FRU Information Device

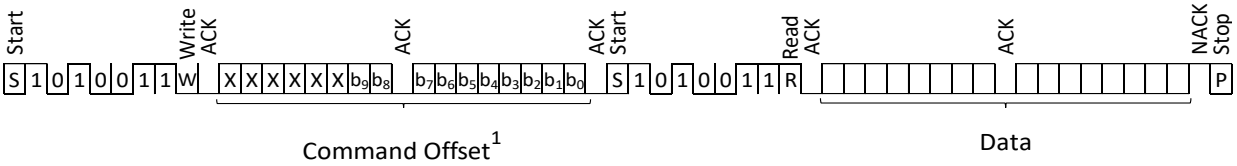


Figure 168 shows an I2C Read where the A6h addresses and Command Offset are provided by the Management Controller followed by data being returned from the Management Endpoint. The Command Offset as shown in Figure 168 is stored internal to the NVMe Storage Device (i.e., the internal offset).

The number of valid bits in the Command Offset is dependent on the Maximum FRU Information Size. Command Offset bits that contain an "X" in Figure 168 are not valid and shall be ignored. This example shows a FRU Information Device with 10 valid Command Offset bits which corresponds to a Maximum FRU Information Device Size of 1 KiB.

If an I2C Read is issued, then data is returned from the internal offset within the FRU Information Device and then the internal offset is incremented by 1h. If the Management Controller reads the last byte of the FRU Information Device (refer to Maximum FRU Information Size) via an I2C Read, then the internal offset shall be cleared to 0h (i.e., rolls over to 0h). If only one byte of the Command Offset is provided by the Management Controller, then the least-significant byte of the internal offset shall be set to that value and the most-significant byte of the internal offset shall be cleared to 0h.

The internal offset shall be cleared to 0h by a power on of the FRU Information Device. All other SMBus Resets should clear the internal offset value to 0h.

8.2.1 Common Header

The fields that make up the VPD Common Header are shown in Figure 169.

Figure 169: Common Header

Bytes	Factory Default	Description
0	01h	**IPMI Format Version Number (IPMIVER):** This field indicates the IPMI Format Version.
1	Impl Spec	**Internal Use Area Starting Offset (IUAOFF):** This field indicates the starting offset in multiples of 8 bytes for the Internal Use Area. A value of 0h may be used to indicate the Internal Use Area is not present.
2	Impl Spec	**Chassis Info Area Starting Offset (CIAOFF):** This field indicates the starting offset in multiples of 8 bytes for the Chassis Info Area. A value of 0h may be used to indicate the Chassis Info Area is not present.
3	Impl Spec	**Board Info Area Starting Offset (BIAOFF):** This field indicates the starting offset in multiples of 8 bytes for the Board Info Area. A value of 0h may be used to indicate the Board Info Area is not present.
4	Impl Spec	**Product Info Area Starting Offset (PIAOFF):** This field indicates the starting offset in multiples of 8 bytes for the Product Info Area. A value of 0h may be used to indicate the Product Info Area is not present.
5	Impl Spec	**MultiRecord Info Area Starting Offset (MRIOFF):** This field indicates the starting offset in multiples of 8 bytes for the MultiRecord Info Area.

Figure 169: Common Header

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>6</td><td>00h</td><td>Boot Failure Code (BFC): If the Boot Failure Code Support bit (refer to Figure 193) is set to ‘1’, then this field shall indicate the applicable boot failure code from the following table. A boot failure is a failure to load or initialize the NVM Subsystem firmware. If this field is updated, then the Common Header Checksum field shall be updated.
If the Boot Failure Code Support bit is cleared to ‘0’, then this field is reserved.
<table>
<tr><th>Boot Failure Code</th><th>Description</th></tr>
<tr><td>0h</td><td>No boot failure has occurred</td></tr>
<tr><td>1h</td><td>Unrecoverable Hardware Issue</td></tr>
<tr><td>2h</td><td>Self-test Failure</td></tr>
<tr><td>4h</td><td>Corrupted Critical Data</td></tr>
<tr><td>6h</td><td>Corrupted Key Manifest</td></tr>
<tr><td>8h</td><td>Corrupted Firmware Image</td></tr>
<tr><td>Ah</td><td>Corrupted Security Data</td></tr>
<tr><td>Ch</td><td>Corrupted Recovery Firmware</td></tr>
</table></td></tr>
<tr><td>7</td><td>Impl Spec</td><td>Common Header Checksum (CHCHK): Checksum computed over byte 0 to byte 6. The checksum is computed by adding the 8-bit value of the bytes modulo 256 and then taking the 2’s complement of this sum. When the checksum and the sum of the bytes module 256 are added, the result should be 0h.</td></tr>
</table>

8.2.2 Product Info Area (offset 8 bytes)

The optional Product Info Area shall have the same format and conventions as the Product Info Area Format as defined by the IPMI Platform Management FRU Information Storage Definition. Therefore, all fields within the Product Info Area shall not follow the conventions defined in section 1.7. The Product Info Area factory default values shall be set to the values defined in Figure 171. The Type/Length bytes use the format shown in Figure 170.

Figure 170: Type/Length Byte Format

Bits	Description
7:6	**Type Code (TCODE):** Specifies field encoding 11b – Always corresponds to ASCII in this specification
5:0	**Number of Data Bytes (NDB):** Specifies field length 0h indicates that the field is empty

Figure 171: Product Info Area Factory Default Values

Factory Default	Description
01h	**IPMI Format Version Number (IPMIVER):** This field indicates the IPMI Format Version.
Impl Spec	**Product Info Area Length (PALEN):** This field indicates the length of the Product Info Area in multiples of 8 bytes.
19h	**Language Code (LCODE):** This field indicates the language used. A value of 19h is used to indicate English.
Impl Spec	**Manufacturer Name Type/Length (MNTL):** This field indicates the type and length of the Manufacturer Name field. The maximum length is 8.
Impl Spec	**Manufacturer Name (MNAME):** This field indicates the manufacturer name in 8-bit ASCII. The manufacturer name in this field should correspond to that in the PCI Subsystem Vendor ID (SSVID) field and the IEEE OUI Identifier field in the Identify Controller data structure and should not be padded. If padded, then those pad bytes should be NULL characters.

Figure 171: Product Info Area Factory Default Values

Factory Default	Description
Impl Spec	**Product Name Type/Length (PNTL):** This field indicates the type and length of the Product Name field. The maximum length is 24.
Impl Spec	**Product Name (PNAME):** This field indicates the product name in 8-bit ASCII and should not be padded. If padded, then those pad bytes should be NULL characters.
Impl Spec	**Product Part/Model Number Type/Length (PPMNNTL):** This field indicates the type and length of the Product Part/Model Number field. The maximum length is 40.
Impl Spec	**Product Part/Model Number (PPMN):** This field indicates the product part/model number in 8-bit ASCII. This field should contain the same value as the Model Number (MN) field in the Identify Controller data structure (refer to the NVM Express Base Specification) with the exclusion of any spaces (i.e., ASCII character 20h) added in that MN field for padding. If padding is added to this field, then those pad bytes should be NULL characters.
Impl Spec	**Product Version Type/Length (PVTL):** This field indicates the type and length of the Product Version field. The maximum length is 2.
Impl Spec	**Product Version (PVER):** This field indicates the Product Version in 8-bit ASCII and should not be padded. If padded, then those pad bytes should be NULL characters.
Impl Spec	**Product Serial Number Type/Length (PSNTL):** This field indicates the type and length of the Product Serial Number field. The maximum length is 20.
Impl Spec	**Product Serial Number (PSN):** This field indicates the product serial number in 8-bit ASCII. This field should contain the same value as the Serial Number (SN) field in the Identify Controller data structure (refer to the NVM Express Base Specification) with the exclusion of any spaces (i.e., ASCII character 20h) added in that SN field for padding. If padding is added to this field, then those pad bytes should be NULL characters.
Impl Spec	**Asset Tag Type/Length (ATTL):** This field indicates the type and length of the Asset Tag field. A value of 0h may be used to indicate an Asset Tag is not present.
Impl Spec	**Asset Tag (AT):** This field indicates the asset tag.
Impl Spec	**FRU File ID Type/Length (ATTL):** This field indicates the type and length of the FRU File ID field. A value of 0h may be used to indicate a FRU File ID is not present.
Impl Spec	**FRU File ID (FFI):** This field provides manufacturing aid for verifying the file that was used during manufacture or field update to load the FRU information.
Impl Spec	**Custom Product Info Area (CPIA):** This optional field allows for the addition of custom Product Info Area fields that shall be proceeded with a Type/Length field.
C1h	**End of Record (EOR):** A value of C1h in this field indicates the end of record.
0h	Zero or more bytes of value 0h that are used to pad the size of the Product Info Area to a multiple of 8 bytes.
Impl Spec	**Product Info Area (PICHK):** Checksum computed over all bytes in the Product Info Area excluding this field. The checksum is computed by adding the 8-bit value of the byes modulo 256 and then taking the 2's complement of this sum. When the checksum and the sum of the bytes module 256 are added, the result should be 0h.

8.2.3 NVMe MultiRecord Area

This MultiRecord is used to describe the form factor, power requirements, and capacity of NVMe Storage Devices with a single NVM Subsystem. Implementations compliant to version 1.1 and later of this specification should implement the Topology MultiRecord (refer to section 8.2.5). For backwards compatibility, the NVMe MultiRecord and the NVMe PCIe Port MultiRecord (refer to section 8.2.4) should both be included in the VPD in addition to the Topology MultiRecord unless:

a) the NVMe Storage Device FRU has:

   1. Expansion Connectors; or
   2. more than one NVM Subsystem;

   or

b) including both this MultiRecord and the NVMe PCIe Port MultiRecord would extend the size of the VPD beyond 256 bytes.

If either the NVMe MultiRecord or NVMe PCIe Port MultiRecord is not included, then neither MultiRecord should be included.

Figure 172: NVMe MultiRecord Area

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>00</td><td>0Bh</td><td>Record Type Identifier (RTI): This field indicates the type of MultiRecord. This field shall be set to 0Bh (i.e., NVMe MultiRecord).</td></tr>
<tr><td>01</td><td>02h or 82h</td><td>Record Format (RFMT): This field indicates format attributes for the record.
<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7</td><td>Last Record (LREC): If this is the last record in the list, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.</td></tr><tr><td>6:0</td><td>Record Version (RVER): This field indicates the NVMe MultiRecord format version. This field shall be set to 2h.</td></tr></table></td></tr>
<tr><td>02</td><td>20h or 3Bh</td><td>Record Length (RLEN): This field indicates the length of the MultiRecord Area in bytes without including the first 5 bytes that are common to all MultiRecords.</td></tr>
<tr><td>03</td><td>Impl Spec</td><td>Record Checksum (RCSUM): This field is used to give the record data a zero checksum (i.e., the modulo 256 sum of the record data bytes from byte offset 05 to the end of this record plus this checksum byte equals 0h).</td></tr>
<tr><td>04</td><td>Impl Spec</td><td>Header Checksum (HSUM): This field is used to give the record header a zero checksum (i.e., the modulo 256 sum of the least-significant byte of the header through this checksum byte equals 0h).</td></tr>
<tr><td>05</td><td>0h</td><td>NVMe MultiRecord Area Version Number (NMAVN): This field indicates the version number of this NVMe MultiRecord. This field shall be cleared to 0h in this version of the specification.</td></tr>
<tr><td>06</td><td>Impl Spec</td><td>Form Factor (FF): This field indicates the form factor of the Management Endpoint. Refer to the values in Figure 180.</td></tr>
<tr><td>12:07</td><td>0h</td><td>Reserved</td></tr>
<tr><td>13</td><td>Impl Spec [1]</td><td>Initial 1.8 V Power Supply Requirements (I1P8PSR): This field specifies the initial 1.8 V power supply requirements in Watts prior to receiving a Set Slot Power message.</td></tr>
<tr><td>14</td><td>Impl Spec [1]</td><td>Maximum 1.8 V Power Supply Requirements (M1P8PSR): This field specifies the maximum 1.8 V power supply requirements in Watts.</td></tr>
<tr><td>15</td><td>Impl Spec [1]</td><td>Initial 3.3 V Power Supply Requirements (I3P3PSR): This field specifies the initial 3.3 V power supply requirements in Watts prior to receiving a Set Slot Power message.</td></tr>
<tr><td>16</td><td>Impl Spec [1]</td><td>Maximum 3.3 V Power Supply Requirements (M3P3PSR): This field specifies the maximum 3.3 V power supply requirements in Watts.</td></tr>
<tr><td>17</td><td>0h</td><td>Reserved</td></tr>
<tr><td>18</td><td>Impl Spec [1]</td><td>Maximum 3.3 V aux Power Supply Requirements (M3P3APSR): This field specifies the maximum 3.3 V power supply requirements in 10 mW units.</td></tr>
<tr><td>19</td><td>Impl Spec [1]</td><td>Initial 5 V Power Supply Requirements (I5PSR): This field specifies the initial 5 V power supply requirements in Watts prior to receiving a Set Slot Power message.</td></tr>
<tr><td>20</td><td>Impl Spec [1]</td><td>Maximum 5 V Power Supply Requirements (M5PSR): This field specifies the maximum 5 V power supply requirements in Watts.</td></tr>
</table>

Figure 172: NVMe MultiRecord Area

Bytes	Factory Default	Description
21	Impl Spec [1]	**Initial 12 V Power Supply Requirements (I12PSR):** This field specifies the initial 12 V power supply requirements in Watts prior to receiving a Set Slot Power message.
22	Impl Spec [1]	**Maximum 12 V Power Supply Requirements (M12PSR):** This field specifies the maximum 12 V power supply requirements in Watts.
23	Impl Spec	**Maximum Thermal Load (MTL):** This field specifies the maximum thermal load from the NVM Subsystem in Watts.
36:24	Impl Spec	**Total NVM Capacity (TNVMCAP):** This field indicates the total NVM capacity of the NVM Subsystem in bytes. If the NVM Subsystem supports Namespace Management, then this field should correspond to the value reported in the TNVMCAP field in the Identify Controller data structure (refer to the NVM Express Base Specification). A value of 0h may be used to indicate this feature is not supported.
63:37	0h	**Pad (PAD):** If the RLEN field is set to 3Bh, then this field is reserved. If the RLEN field is set to 20h, then this field is not present.

Notes:
1. Power supply requirements shall be set to the smallest integer value which fully supplies the necessary power to the NVMe Storage Device. A value of 0h indicates that the power supply voltage is not used.

8.2.4 NVMe PCIe Port MultiRecord Area

This MultiRecord is used to describe the PCIe connectivity for NVMe Storage Devices with a single NVM Subsystem. Implementations compliant to version 1.1 and later of this specification should implement the Topology MultiRecord (refer to section 9.2.5). For backwards compatibility, the NVMe PCIe Port MultiRecord and the NVMe MultiRecord (refer to section 8.2.3) should both be included in the VPD in addition to the Topology MultiRecord unless:

a) the NVMe Storage Device FRU:

   1. has Expansion Connectors; or
   2. more than one NVM Subsystem;

   or

b) if including both this MultiRecord and the NVMe MultiRecord would extend the size of the VPD beyond 256 bytes.

If either the NVMe MultiRecord or NVMe PCIe Port MultiRecord is not included then neither MultiRecord should be included.

Figure 173: NVMe PCIe Port MultiRecord Area

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>00</td><td>0Ch</td><td>Record Type Identifier (RTI): This field indicates the type of MultiRecord. This field shall be set to 0Ch (i.e., NVMe PCIe Port MultiRecord).</td></tr>
<tr><td>01</td><td>02h or 82h</td><td>Record Format (RFMT): This field indicates format attributes for the record.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Last Record (LREC): If this is the last record in the list, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.</td></tr>
<tr><td>6:0</td><td>Record Version (RVER): This field indicates the NVMe PCIe Port MultiRecord format version. This field shall be set to 2h.</td></tr>
</table>
</td></tr>
</table>

Figure 173: NVMe PCIe Port MultiRecord Area

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>02</td><td>08h or 0Bh</td><td>Record Length (RLEN): This field indicates the length of the MultiRecord Area in bytes without including the first 5 bytes that are common to all MultiRecords.</td></tr>
<tr><td>03</td><td>Impl Spec</td><td>Record Checksum (RCSUM): This field is used to give the record data a zero checksum (i.e., the modulo 256 sum of the record data bytes from byte offset 05 to the end of this record plus this checksum byte equals 0h).</td></tr>
<tr><td>04</td><td>Impl Spec</td><td>Header Checksum (HSUM): This field is used to give the record header a zero checksum (i.e., the modulo 256 sum of the least-significant byte of the header through this checksum byte equals 0h).</td></tr>
<tr><td>05</td><td>1h</td><td>NVMe PCIe Port MultiRecord Area Version Number (NPCIEPMAVN): This field indicates the version number of this NVMe PCIe Port MultiRecord. This field shall be set to 1h in this version of the specification.</td></tr>
<tr><td>06</td><td>Impl Spec</td><td>PCIe Port Number (PCIEPN): This field contains the PCIe port number. This is the same value as that reported in the Port Number field in the PCIe Link Capabilities Register.</td></tr>
<tr><td>07</td><td>Impl Spec</td><td>Port Information (PINFO): This field indicates information about the PCIe Ports in the device.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>Common PCIe Port Capabilities (CPPC): If this bit is set to ‘1’, then all PCIe ports within the device have the same capabilities (i.e., the capabilities listed in this structure are consistent across each PCIe port). If this bit is cleared to ‘0’, then all PCIe ports within the device do not have the same capabilities.</td></tr>
</table></td></tr>
<tr><td>08</td><td>Impl Spec</td><td>PCIe Link Speed (PCIELS): This field indicates a bit vector of link speeds supported by the PCIe port.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:6</td><td>Reserved</td></tr>
<tr><td>5</td><td>64.0 GT/s Support (GTS64): Set to ‘1’ if the PCIe link supports 64.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>4</td><td>32.0 GT/s Support (GTS32): Set to ‘1’ if the PCIe link supports 32.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>3</td><td>16.0 GT/s Support (GTS16): Set to ‘1’ if the PCIe link supports 16.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>2</td><td>8.0 GT/s Support (GTS8): Set to ‘1’ if the PCIe link supports 8.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>1</td><td>5.0 GT/s Support (GTS5): Set to ‘1’ if the PCIe link supports 5.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>0</td><td>2.5 GT/s Support (GTS2P5): Set to ‘1’ if the PCIe link supports 2.5 GT/s, otherwise cleared to ‘0’.</td></tr>
</table></td></tr>
</table>

Figure 173: NVMe PCIe Port MultiRecord Area

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>09</td><td>Impl Spec</td><td>PCIe Maximum Link Width (PCIEMLW): The maximum PCIe link width for this NVM Subsystem port. This is the expected negotiated link width that the port link trains to if the platform supports it. A Requester may compare this value with the PCIe Negotiated Link Width to determine if there has been a PCIe link training issue.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>PCIe x1</td></tr>
<tr><td>2</td><td>PCIe x2</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
<tr><td>4</td><td>PCIe x4</td></tr>
<tr><td>5 to 7</td><td>Reserved</td></tr>
<tr><td>8</td><td>PCIe x8</td></tr>
<tr><td>9 to 11</td><td>Reserved</td></tr>
<tr><td>12</td><td>PCIe x12</td></tr>
<tr><td>13 to 15</td><td>Reserved</td></tr>
<tr><td>16</td><td>PCIe x16</td></tr>
<tr><td>17 to 31</td><td>Reserved</td></tr>
<tr><td>32</td><td>PCIe x32</td></tr>
<tr><td>33 to 255</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>10</td><td>Impl Spec</td><td>MCTP Support (MCTPS): This field indicates a bit vector that specifies the level of support for the NVMe Management Interface.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>MCTP PCIe VDM Support (MCTPPCIEVS): If MCTP-based NVMe-MI Messages are supported on this PCIe port, then this bit shall be set to ‘1’; otherwise, this bit shall be cleared to ‘0’.</td></tr>
</table></td></tr>
<tr><td>11</td><td>Impl Spec</td><td>Ref Clk Capability (RCCAP): This field contains a bit vector that specifies the PCIe clocking modes supported by the port.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:4</td><td>Reserved</td></tr>
<tr><td>3</td><td>RefClk Support (RCS): Set to ‘1’ if the device automatically uses RefClk if provided and otherwise uses SRIS, otherwise cleared to ‘0’.</td></tr>
<tr><td>2</td><td>Separate RefClk with SSC Support (SRCSS): Set to ‘1’ if the PCIe link supports Separate RefClk with SSC (SRIS), otherwise cleared to ‘0’.</td></tr>
<tr><td>1</td><td>Separate RefClk with No SSC Support (SRCNSS): Set to ‘1’ if the PCIe link supports Separate RefClk with no SSC (SRNS), otherwise cleared to ‘0’.</td></tr>
<tr><td>0</td><td>Common RefClk Support (CRCS): Set to ‘1’ if the PCIe link supports common RefClk, otherwise cleared to ‘0’.</td></tr>
</table></td></tr>
<tr><td>12</td><td>Impl Spec</td><td>Port Identifier (PORTID): This field contains the NVMe-MI Port Identifier.</td></tr>
<tr><td>15:13</td><td>0h</td><td>Pad (PAD): If the RLEN field is set to 0Bh, then this field is reserved. If the RLEN field is set to 08h, then this field is not present.</td></tr>
</table>

8.2.5 Topology MultiRecord Area

This MultiRecord describes an NVMe Storage Device’s architectural elements and their connections. It is required on all NVMe Storage Device FRUs.

The Topology MultiRecord consists mainly of a list of Element Descriptors as shown in Figure 174. Element Descriptors are used to describe the architectural elements that make up an NVMe Storage Device such as NVM Subsystems, Upstream Connectors, Expansion Connectors, 2-Wire elements, and PCIe elements.

Each architectural element has an Element Descriptor Type. The format of an Element Descriptor is shown in Figure 176 and Element Descriptor Types are listed in Figure 177.

Element Descriptors may have fields that are used to point to other Element Descriptors. When an Element Descriptor contains a pointer to another Element Descriptor, then the Element Descriptor containing the pointer is called the parent and the Element Descriptor pointed to by the parent is called the child. An Element Descriptor may be both a child and a parent.

An Element Descriptor pointer is either populated with an index of the child or 0h to indicate that there is no child. The index is a logical construct that indicates the position of an Element Descriptor in the VPD. The Element Descriptor at the lowest byte offset in the VPD has an index of 0, the Element Descriptor at the second lowest byte offset has an index of 1, and so on. A child may have an index that is higher or lower than its parent. The Element Descriptor at the lowest byte offset (i.e., index 0) shall be an Upstream Connector Element Descriptor. Some Element Descriptors use indexes in a similar manner to select a Port from a list of Ports.

Figure 174: Topology MultiRecord

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>00</td><td>0Dh</td><td>Record Type Identifier (RTI): This field indicates the type of MultiRecord. This field shall be set to 0Dh (i.e., Topology MultiRecord).</td></tr>
<tr><td>01</td><td>2h or 82h</td><td>Record Format (RFMT): This field indicates format attributes for the record.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Last Record (LREC): If this is the last record in the list, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.</td></tr>
<tr><td>6:0</td><td>Record Version (RVER): This field indicates the Topology MultiRecord format version. This field shall be set to 2h.</td></tr>
</table></td></tr>
<tr><td>02</td><td>Impl Spec</td><td>Record Length (RLEN): This field indicates the length of the MultiRecord Area in bytes without including the first 5 bytes that are common to all MultiRecords.</td></tr>
<tr><td>03</td><td>Impl Spec</td><td>Record Checksum (RCSUM): This field is used to give the record data a zero checksum (i.e., the modulo 256 sum of the record data bytes from byte offset 05 to the end of this record plus this checksum byte equals 0h).</td></tr>
<tr><td>04</td><td>Impl Spec</td><td>Header Checksum (HSUM): This field is used to give the record header a zero checksum (i.e., the modulo 256 sum of the least-significant byte of the header through this checksum byte equals 0h).</td></tr>
<tr><td>05</td><td>0h</td><td>Version Number (VNUM): This field indicates the version number of this Topology MultiRecord. This field shall be cleared to 0h in this version of the specification.</td></tr>
<tr><td>06</td><td>0h</td><td>Reserved</td></tr>
<tr><td>07</td><td>Impl Spec</td><td>Element Count (ELEMC): This field indicates the number of Element Descriptors in this Topology MultiRecord. The value of 0h is reserved.</td></tr>
<tr><td colspan="3">Element Descriptor List</td></tr>
<tr><td>Impl Spec: 08</td><td>Impl Spec</td><td>Element Descriptor 0: This field contains the first Element Descriptor in this Topology MultiRecord.</td></tr>
<tr><td>Impl Spec</td><td>Impl Spec</td><td>Element Descriptor 1: This field contains the second Element Descriptor in this Topology MultiRecord if the Element Count field is greater than 1h; otherwise this field is not present.</td></tr>
<tr><td>…</td><td>…</td><td>…</td></tr>
<tr><td>Impl Spec</td><td>Impl Spec</td><td>Element Descriptor N: This field contains the last Element Descriptor in this Topology MultiRecord if the Element Count field is greater than N; otherwise, this field is not present. N is equal to the value of the ELEMC field minus 1h.</td></tr>
</table>

The VPD may contain more than one Topology MultiRecord only when the list of required Element Descriptors is too large to fit into a single Topology MultiRecord. If there is more than one Topology MultiRecord, then the index associated with Element Descriptors continues to increment sequentially across Topology MultiRecord instances. Figure 175 illustrates multiple Topology MultiRecords where Index

0 is at the lowest byte offset of any Element Descriptor in the VPD. Parent Element Descriptors may be in different Topology MultiRecords from their Child Element Descriptors.

Figure 175: Indexing Across Extended MultiRecords

Index	Topology Multi Record Instance	Element Descriptors	Child Indices
0	0	Element Descriptor 0, parent of 2, 3, 5	2, 3, 5
1		Element Descriptor 1, child of 5	
2		Element Descriptor 2, child of 0	
3		Element Descriptor 3, child of 0	
4	1	Element Descriptor 0 [1]	
5		Element Descriptor 1, child of 0, parent of 1	1

Notes:
1. This Element Descriptor is an Extended Element Descriptor that extends the preceding Element Descriptor at index 3. Extended Element Descriptors are further detailed in section 8.2.5.1.

Figure 176: Element Descriptor

Bytes	Factory Default	Description
00	Impl Spec	**Type (TYP):** This field indicates the type of the Element Descriptor. Values are defined in Figure 177.
01	Impl Spec	**Revision (REV):** This field indicates the revision of the Element Descriptor.
02	Impl Spec	**Length (LEN):** Number of bytes in the Element Descriptor.
LEN - 1:03	Impl Spec	**Type-Specific Information (TSINFO):** This area contains the Type-specific information associated with the Element Descriptor. Type-specific information is defined for each Element Descriptor Type in the subsections below.

Element Descriptor Types, fields, and bits in the VPD that are defined as reserved should be ignored by Requesters to ensure forward and backward compatibility. Extra trailing bytes in an Element Descriptor should be treated as reserved in order to tolerate the Length of an Element Descriptor increasing as new fields are appended in future revisions of the Element Descriptor.

Element Descriptor Types are defined in Figure 177. Subsequent sections define the details associated with each Element Descriptor Type.

Figure 177: Element Descriptor Types

Value	Element Descriptor Type	Reference Section
0	Reserved	-
1	Extended Element Descriptor	8.2.5.1
2	Upstream Connector Element Descriptor	8.2.5.2
3	Expansion Connector Element Descriptor	8.2.5.3
4	Label Element Descriptor	8.2.5.4
5	2-Wire Mux Element Descriptor	8.2.5.5
6	PCIe Switch Element Descriptor	8.2.5.6
7	NVM Subsystem Element Descriptor	8.2.5.7
8	FRU Information Device Element Descriptor	8.2.5.8
9 to 239	Reserved	-
240 to 255	Vendor specific	8.2.5.9

8.2.5.1 Extended Element Descriptor

The Extended Element Descriptor is shown in Figure 178. This Element Descriptor Type shall only be used when an Element Descriptor spans across more than one Topology MultiRecord. Extended Element Descriptors shall not be the children of other Element Descriptors.

If an Element Descriptor causes the maximum size of a Topology MultiRecord to be exceeded, then that Element Descriptor is truncated so that the non-truncated portion of the Element Descriptor fits into the Topology MultiRecord. The truncated portion of the Element Descriptor forms the contents of the Extended Content field in an Extended Element Descriptor. That Extended Element Descriptor is the first Element Descriptor in the next Topology MultiRecord. If the truncated portion of the Element Descriptor does not fit into a single Topology MultiRecord, then two or more Extended Element Descriptors are required, each in subsequent Topology MultiRecords.

An example is shown in Figure 175 where the Element Descriptor at index 4 is an Extended Element Descriptor that extends the Element Descriptor at index 3. Element Descriptor 3 is the child of Element Descriptor 0 and Element Descriptor 4 is not the child of any parent Element Descriptor.

Figure 178: Extended Element Descriptor

Bytes	Factory Default	Description
00	01h	**Type (TYP):** This field indicates the type of the Element Descriptor. The field shall be set to the Extended Element Descriptor Type (i.e., 1h). Refer to Figure 177.
01	00h	**Revision (REV):** This field indicates the revision of the Extended Element Descriptor. This field shall be cleared to 0h.
02	Impl Spec	**Length (LEN):** This field indicates the length of the Extended Element Descriptor in bytes.
LEN - 1:03	Impl Spec	**Extended Content (EXTC):** This field extends the content of the Element Descriptor at the immediately preceding index.

8.2.5.2 Upstream Connector Element Descriptor

The Upstream Connector Element Descriptor is shown in Figure 179 and is used to describe an Upstream Connector (i.e., a connector through which a Requester communicates with the NVMe Storage Device). Upstream Element Descriptors are always a parent and never a child.

Figure 179: Upstream Connector Element Descriptor

Bytes	Factory Default	Description
00	02h	**Type (TYP):** This field indicates the type of the Element Descriptor. This field shall be set to the Upstream Connector Element Descriptor Type (i.e., 2h). Refer to Figure 177.
01	00h	**Revision (REV):** This field indicates the revision of the Upstream Connector Element Descriptor. This field shall be cleared to 0h.
02	Impl Spec	**Length (LEN):** This field indicates the length of the entire Upstream Connector Element Descriptor in bytes.
03	Impl Spec	**Form Factor (FF):** This field indicates the Form Factor of the NVMe Storage Device. Refer to Figure 180 for a list of defined values.
04	Impl Spec	**Label Pointer (LPTR):** If the Upstream Connector has a label, then this field shall contain the index of a Label Element Descriptor that contains the label. The value 0h indicates there is no associated label.
06:05	00h	Reserved

Figure 179: Upstream Connector Element Descriptor

Bytes	Factory Default	Description
07	Impl Spec	**Maximum Auxiliary Power (MAXAPWR):** This field specifies the maximum auxiliary power supply requirements in 10 mW increments consumed by the NVMe Storage Device. A value of 0h indicates that auxiliary power is not used from this Upstream Connector.
09:08	Impl Spec	**Maximum Power (MAXPWR):** This field specifies the maximum power in Watts consumed by the NVMe Storage Device.
10	Impl Spec	**Upstream Port Descriptor Count (UPDC):** This field indicates the number of Upstream Port Descriptors associated with this Upstream Connector Element Descriptor. The permitted range of values is 1 to 64.
Upstream Port Descriptor List		
Impl Spec: 11	Impl Spec	**Upstream Port Descriptor 0:** This field contains the first Upstream Port Descriptor.
Impl Spec	Impl Spec	**Upstream Port Descriptor 1:** This field contains the second Upstream Port Descriptor if the Port Descriptor Count field is greater than 1h; otherwise this field is not present.
…	…	…
Impl Spec	Impl Spec	**Upstream Port Descriptor N:** This field contains the last Upstream Port Descriptor if the Port Descriptor Count field is greater than N; otherwise, this field is not present. N is equal to the value of the UPDC field minus 1h.

The value of the Form Factor field indicates the NVMe Storage Device's form factor. Figure 180 lists the NVMe Storage Device's Form Factor values.

Figure 180: Form Factors

Value	Definition	
	Interface	Form Factor Description
0	Unspecified	Other – unknown
1	PCIe	Integrated
2	PCIe	Other - unknown
3 to 15	Reserved	
16	PCIe	2.5" Form Factor – unknown
17	PCIe	2.5" Form Factor – PCI Express SFF-8639 Module (U.2) 15 mm
18	PCIe	2.5" Form Factor – PCI Express SFF-8639 Module (U.2) 7 mm
19	PCIe	2.5" Form Factor – (SFF-TA-1001) 15 mm
20	PCIe	2.5" Form Factor – (SFF-TA-1001) 7 mm
21 to 31	Reserved	
32	PCIe	CEM add in card – unknown
33	PCIe	CEM add in card – Low Profile (HHHL)
34	PCIe	CEM add in card – Standard Height Half Length (FHHL)
35	PCIe	CEM add in card – Standard Height Full Length (FHFL)
36 to 47	Reserved	
48	PCIe	M.2 module – unknown
49	PCIe	M.2 module – 2230
50	PCIe	M.2 module – 2242
51	PCIe	M.2 module – 2260
52	PCIe	M.2 module – 2280
53	PCIe	M.2 module – 22110
54 to 63	Reserved	
64	PCIe	BGA SSD – unknown
65	PCIe	BGA SSD – 16 x 20mm (M.2 Type 1620)
66	PCIe	BGA SSD – 11.5 x 13mm (M.2 Type 1113)

Figure 180: Form Factors

Value	Definition	
	Interface	Form Factor Description
67 to 79	Reserved	
80	PCIe	Enterprise & Datacenter SSD Form Factor – unknown
81	PCIe	E1.S - (SFF-TA-1006) 5.9 mm
82	PCIe	E1.S - (SFF-TA-1006) 8 mm
83	PCIe	E1.L - (SFF-TA-1007) 9.5 mm
84	PCIe	E1.L - (SFF-TA-1007) 18 mm
85	PCIe	E3.S - (SFF-TA-1008) 7.5 mm
86	PCIe	E3.S - (SFF-TA-1008) 16.8 mm
87	PCIe	E3.L - (SFF-TA-1008) 7.5 mm
88	PCIe	E3.L - (SFF-TA-1008) 16.8 mm
89	PCIe	E1.S - (SFF-TA-1006) 9.5 mm
90	PCIe	E1.S - (SFF-TA-1006) 15 mm
91	PCIe	E1.S - (SFF-TA-1006) 25 mm
92 to 95	Reserved	
96	Ethernet	Other – unknown
97	Ethernet	2.5” Form Factor – (Native NVMe-oF Drive) 15 mm
98	Ethernet	2.5” Form Factor – (Native NVMe-oF Drive) 7 mm
99	Ethernet	E3.S – (Native NVMe-oF Drive) 7.5 mm
100	Ethernet	E3.S – (Native NVMe-oF Drive) 16.8 mm
101 to 239	Reserved	
240 to 255	Vendor Specific	Vendor Specific

The Upstream Connector may have an associated label, such as silk-screened text on the printed circuit board. If the Upstream Connector has a label, then the Label Pointer may contain the index of the associated Label Element Descriptor.

The Upstream Connector Element Descriptor contains a list of the Upstream Port Descriptors that are ports through which a Requester communicates with the NVMe Storage Device. Each Upstream Port Descriptor has a type. The types defined in this specification are 2-Wire Upstream Port Descriptor and PCIe Upstream Port Descriptor.

A 2-Wire Upstream Port Descriptor is shown in Figure 181. It contains a list of pointers to child Element Descriptors whose 2-Wire port is directly connected to the Upstream Connector.

Figure 181: 2-Wire Upstream Port Descriptor

Bytes	Factory Default	Description
00	00h	**Type (TYP):** This field indicates the type of the Port Descriptor. This field shall be cleared to 0h.
01	Impl Spec	**Length (LEN):** This field indicates the length of the 2-Wire Upstream Port Descriptor in bytes.
02	Impl Spec	**Count (CNT):** This field indicates the number of 2-Wire Pointers in the 2-Wire Upstream Port Descriptor. The permitted range of values is 1 to 32.
2-Wire **Pointer List**		
03	Impl Spec	**2-Wire Pointer 0:** This field contains the child index of the first Element Descriptor whose 2-Wire port is connected to this 2-Wire port.
04	Impl Spec	**2-Wire Pointer 1:** This field contains the child index of the second Element Descriptor whose 2-Wire port is connected to this 2-Wire port if the Count field is greater than 1h; otherwise, this this field is not present.
…	…	…

Figure 181: 2-Wire Upstream Port Descriptor

Bytes	Factory Default	Description
CNT+2	Impl Spec	**2-Wire Pointer N:** This field contains the child index of the last Element Descriptor whose 2-Wire port is connected to this 2-Wire port if the Count field is greater than 2h; otherwise, this this field is not present. N is equal to the value of the CNT field minus 1h.

A PCIe Upstream Port Descriptor is shown in Figure 182. It indicates the starting and ending PCIe lane numbers on the Upstream Connector that make up a PCIe Upstream Port. The PCIe Upstream Port Descriptor contains a single pointer to a child Element Descriptor connected to this PCIe Upstream Port. The Destination Port field of the PCIe Upstream Port Element Descriptor specifies which port of the child is connected to this Upstream Connector. The Destination Port value is an index into the child Element Descriptor's list of Port Descriptors.

Figure 182: PCIe Upstream Port Descriptor

Bytes	Factory Default	Description
00	01h	**Type (TYP):** This field indicates the type of Upstream Port Descriptor. This field shall be set to 1h.
01	06h	**Length (LEN):** This field indicates the length of the PCIe Upstream Port Descriptor in bytes.
02	Impl Spec	**Starting Lane (SL):** This field indicates the first PCIe lane (i.e., lane 0) of the port from the Upstream Connector.
03	Impl Spec	**Ending Lane (EL):** This field indicates the ending PCIe lane of the port from the Upstream Connector.
04	Impl Spec	**PCIe Pointer (PCIEPTR):** This field contains the child index of the Element Descriptor whose PCIe port is connected to this PCIe Upstream Port.
05	Impl Spec	**Destination Port (DPORT):** This field contains the index of the Port Descriptor in the child Element Descriptor. If the child Element Descriptor has one PCIe upstream port (i.e., a PCIe Switch Element Descriptor) this field shall be cleared to 0h.

The PCIe lanes associated with a PCIe Upstream Connector may be organized as a single large port or subdivided into multiple ports. Each of these ports is described with its own PCIe Upstream Port Descriptor. The PCIe Upstream Port Descriptors may be listed in any order. A form factor specific mechanism, such as the U.2 Dual Port Enable signal, may be used to determine which of the listed PCIe Upstream Port Descriptors are currently applicable. These form factor specific mechanisms are outside the scope of this specification.

For example, a U.2 NVMe Storage Device capable of running in either single-port mode or dual-port mode based on the Dual Port Enable signal is required to have three PCIe Upstream Port Descriptors describing PCIe ports on the following PCIe Lanes:

1. PCIe lanes 0 to 3 (single-port mode);
2. PCIe lanes 0 to 1 (dual-port mode); and
3. PCIe lanes 2 to 3 (dual-port mode).

In the example above, if the U.2 NVMe Storage Device is only capable of running in single-port mode, then only the PCIe Upstream Port Descriptor describing the single-port mode (item 1 in the list above) shall be included in the Upstream Connector Element Descriptor. And if the U.2 NVMe Storage Device is only capable of running in dual-port mode, then only the two PCIe Upstream Port Descriptors describing the dual-port mode (items 2 and 3 in the list above) shall be included in the Upstream Connector Element Descriptor.

In another example, consider a x16 CEM add-in card Upstream Connector that is subdivided into four x4 PCIe ports, also referred to as bifurcation. Each of these x4 PCIe Upstream Ports may connect to different

elements on the NVMe Storage Device. The Upstream Connector in this example shall contain four PCIe Upstream Port Descriptors describing the four PCIe ports:

1. PCIe lanes 0 to 3;
2. PCIe lanes 4 to 7;
3. PCIe lanes 8 to 11; and
4. PCIe lanes 12 to 15.

8.2.5.3 Expansion Connector Element Descriptor

The Expansion Connector Element Descriptor is shown in Figure 183 and is used to describe the form factor, label, and port configurations for Expansion Connectors on a Carrier. The Expansion Connector Element Descriptor shall be a child Element Descriptor.

Figure 183: Expansion Connector Element Descriptor

Bytes	Factory Default	Description
00	03h	**Type (TYP):** This field indicates the type of the Element Descriptor. This field shall be set to the Expansion Connector Element Descriptor Type (i.e., 3h). Refer to Figure 177.
01	00h	**Revision (REV):** This field indicates the revision of the Expansion Connector Element Descriptor. This field shall be cleared to 0h.
02	Impl Spec	**Length (LEN):** This field indicates the length of the Expansion Connector Element Descriptor in bytes.
03	Impl Spec	**Form Factor (FF):** This field indicates the Form Factor of the NVMe Storage Device FRU that plugs into the Expansion Connector. Refer to Figure 180 for a list of defined values.
04	Impl Spec	**Label Pointer (LPTR):** If the Upstream Connector has a label, then this field shall contain the index of a Label Element Descriptor that contains the label. The value 0h indicates there is no associated label.
05	Impl Spec	**Expansion Connector Port Descriptor Count (ECPDC):** This field indicates the number of Expansion Port Descriptors associated with this Expansion Connector Element Descriptor. The permitted range of values is 1 to 64.
Expansion Connector Port Descriptor List		
Impl Spec: 06	Impl Spec	**Expansion Connector Port Descriptor 0:** This field contains the first Expansion Connector Port Descriptor.
Impl Spec	Impl Spec	**Expansion Connector Port Descriptor 1:** This field contains the second Expansion Connector Port Descriptor in this Expansion Connector Descriptor if the Expansion Connector Port Descriptor Count field is greater than 1h; otherwise this field is not present.
…	…	…
Impl Spec	Impl Spec	**Expansion Connector Port Descriptor N:** This field contains the last Expansion Connector Port Descriptor in this Expansion Connector Descriptor if the Expansion Connector Port Descriptor Count field is greater than 2h; otherwise, this field is not present. N is the equal to the value of the ECPDC field minus 1h.

In a manner similar to the PCIe Upstream Connector, the Expansion Connector Element Descriptor's PCIe lanes may support one or more PCIe ports for connecting to external NVMe Storage Device FRUs. The PCIe ports have a starting and ending PCIe lane number on the Expansion Connector that are determined by the external NVMe Storage Device FRU's form factor's lane numbering.

The Expansion Connector Element Descriptor holds the list of Expansion Connector PCIe Port Descriptors. Each PCIe port is described by an Expansion Connector PCIe Port Descriptor whose format is shown in Figure 184. Parent Element Descriptors, such as Upstream Connectors and PCIe Switches, contain Port Descriptors that point to Expansion Connectors. The Destination Port field of the parent Port Descriptor contains an index to the specific Expansion Connector PCIe Port Descriptor instance to which the Port

Descriptor is connected. Each Expansion Connector PCIe Port Descriptor is the destination of exactly one pointer from a parent Element Descriptor.

Figure 184: Expansion Connector PCIe Port Descriptor

Bytes	Factory Default	Description
00	00h	**Type (TYP):** This field indicates the type of Expansion Connector Port Descriptor. This field shall be cleared to 0h.
01	Impl Spec	**Length (LEN):** This field indicates the length of the Expansion Connector PCIe Port Descriptor in bytes.
02	Impl Spec	**Starting Lane (SL):** This field indicates first PCIe lane (i.e., lane 0) of the port on the Expansion Connector PCIe Port Descriptor.
03	Impl Spec	**Ending Lane (EL):** This field indicates the ending PCIe lane of the port on the Expansion Connector PCIe Port Descriptor.

8.2.5.4 Label Element Descriptor

The Label Element Descriptor is shown in Figure 185 and is used to store text strings in the VPD for Element Descriptors that have a label. A Label Element Descriptor shall be a child Element Descriptor.

Figure 185: Label Element Descriptor

Bytes	Factory Default	Description
00	04h	**Type (TYP):** This field indicates the type of the Element Descriptor. This field shall be set to the Label Element Descriptor Type (i.e., 4h). Refer to Figure 177.
01	00h	**Revision (REV):** This field indicates the revision of the Label Element Descriptor. This field shall be cleared to 0h.
02	Impl Spec	**Length (LEN):** This field indicates the length of the Label Element Descriptor in bytes including the null termination.
Length - 1:03	Impl Spec	**Label String (LSTR):** This field contains a null-terminated UTF-8 string used to identify the parent Element Descriptor.

8.2.5.5 2-Wire Mux Element Descriptor

The 2-Wire Mux Element Descriptor is shown in Figure 186 and is used to describe a 2-Wire multiplexor element that connects a single upstream 2-Wire channel to zero or more downstream 2-Wire channels. This Element Descriptor contains the address and capabilities of the 2-Wire Mux followed by a list of 2-Wire Mux Channel Descriptors that describe 2-Wire Mux downstream channel connections. The 2-Wire Mux shall be compatible with the industry standard PCA9542/45/48 family of 2-Wire multiplexors and may be extended to support ARP, error detection, and additional downstream channels as defined below.

Figure 186: 2-Wire Mux Element Descriptor

Bytes	Factory Default	Description
00	05h	**Type (TYP):** This field indicates the type of the Element Descriptor. This field shall be set to the 2-Wire Mux Element Descriptor Type (i.e., 5h). Refer to Figure 177.
01	00h	**Revision (REV):** This field indicates the revision of the 2-Wire Mux Element Descriptor. This field shall be cleared to 0h.
02	Impl Spec	**Length (LEN):** This field indicates the length of the 2-Wire Mux Element Descriptor in bytes.

Figure 186: 2-Wire Mux Element Descriptor

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>03</td><td>E8h or E9h</td><td>2-Wire Address Info (TWADDRI): This field indicates the 2-Wire address and whether or not ARP is supported.
<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7:1</td><td>2-Wire Address (TWADDR): This field contains the 7-bit 2-Wire address. Refer to Figure 16 for requirements.</td></tr><tr><td>0</td><td>ARP Capable (ARPC): This bit is set to ‘1’ if SMBus ARP is supported, else it is cleared to ‘0’. Refer to Figure 16 for requirements.</td></tr></table></td></tr>
<tr><td>04</td><td>Impl Spec</td><td>2-Wire Capabilities (TWCAP): This field indicates the 2-Wire Mux capabilities.
<table><tr><th>Bits</th><th>Description</th></tr><tr><td>7</td><td>Form Factor Reset (FFR): This bit is set to ‘1’ if all of the 2-Wire reset mechanisms are supported as defined by the associated form factor specification. This bit is cleared to ‘0’ if the form factor does not define 2-Wire Reset or the NVMe Storage Device does not support all of the 2-Wire reset mechanisms defined in the specification for the Form Factor in the Host Connector Element Descriptor.</td></tr><tr><td>6</td><td>Packet Error Code Support (PECS): This bit is set to ‘1’ if Packet Error Code (PEC) is supported by the 2-Wire Mux. This bit is cleared to ‘0’ if PEC is not supported.</td></tr><tr><td>5:2</td><td>Reserved</td></tr><tr><td>1:0</td><td>Maximum Speed (MSPD): This field is set to the highest supported SMBus/I2C clock speed by the 2-Wire Mux.
<table><tr><th>Value</th><th>Definition</th></tr><tr><td>0</td><td>100 kHz</td></tr><tr><td>1</td><td>400 kHz</td></tr><tr><td>2</td><td>1 MHz</td></tr><tr><td>3</td><td>Reserved</td></tr></table></td></tr></table></td></tr>
<tr><td>05</td><td>Impl Spec</td><td>2-Wire Mux Channel Descriptor Count (TWMCDC): This field indicates the number of downstream channels listed for this 2-Wire Mux. Each channel has a corresponding 2-Wire Channel Descriptor in the list below. The permitted range of values is 1 to 64. The value of this field may be less than the actual number of Channels implemented by the 2-Wire Mux if the truncated 2-Wire Mux Channel Descriptors are not connected to anything.</td></tr>
<tr><th colspan="3">2-Wire Mux Channel Descriptor List</th></tr>
<tr><td>Impl Spec: 06</td><td>Impl Spec</td><td>2-Wire Mux Channel Descriptor 0: This field contains the first 2-Wire Mux Channel Descriptor.</td></tr>
<tr><td>Impl Spec</td><td>Impl Spec</td><td>2-Wire Mux Channel Descriptor 1: This field contains the second 2-Wire Mux Channel Descriptor in this 2-Wire Mux Element Descriptor if the 2-Wire Mux Channel Descriptor Count field is greater than 1h; otherwise this field is not present.</td></tr>
<tr><td>…</td><td>…</td><td>…</td></tr>
<tr><td>Impl Spec</td><td>Impl Spec</td><td>2-Wire Mux Channel Descriptor N: This field contains the last 2-Wire Mux Channel Descriptor in this 2-Wire Mux Element Descriptor if the 2-Wire Mux Channel Descriptor Count field is greater than 2h; otherwise, this field is not present. N is equal to the value in the TWMCDC field minus 1h.</td></tr>
</table>

A 2-Wire Mux Channel Descriptor is shown in Figure 188. 2-Wire Mux Channel Descriptors that are not connected to anything have a value 0h in the Count field and contain no 2-Wire Mux Channel Descriptors. Unconnected 2-Wire Mux Channel Descriptors at the end of the list in Figure 186 may be truncated unless they are required to position the optional Packet Error Code (PEC).

Writing to a 2-Wire Mux configures the 2-Wire Mux and reading from an SMBus Mux returns its current configuration. Figure 187 shows the protocol for reading and writing a 2-Wire Mux configuration. The white

background blocks are transmitted by a Management Controller and the grey background blocks are transmitted in response by the 2-Wire Mux. The first byte sent or received is the 2-Wire Mux address followed by one or more channel bytes. Each channel byte has eight channel bits that are set to '1' for connecting the corresponding downstream channel to the upstream channel or cleared to '0' for disconnecting the corresponding downstream channel from the upstream channel.

The first channel byte sent or received represents channels 0 to 7, the second channel byte sent or received represents channels 8 to 15, and so on. Within each channel byte the least-significant bit in the byte that is transmitted or received represents the lowest numbered channel. Bits for channels exceeding the 2-Wire Mux Channel Descriptor Count are reserved.

Figure 187: 2-Wire Mux Read and Write Command Format

The minimum number of channel bytes are read or written to reach all the channels specified in the 2-Wire Mux Channel Descriptor Count field. Thus, a 2-Wire Mux with one to eight downstream channels has one channel byte while a 2-Wire Mux with 25 to 32 downstream channels has 4 channel bytes. In the example shown in Figure 187, the 2-Wire Mux has 16 downstream channels that require 2 bytes. In this example, channels 1 and 8 are being connected while all others are being disconnected.

A 2-Wire Mux may also protect communications with an optional Packet Error Code (PEC) that is appended after sufficient channel bytes have been read or written to satisfy the 2-Wire Mux Channel Descriptor Count value. If the write command includes a PEC byte and the PEC byte is incorrect, then the entire command shall be ignored by the 2-Wire Mux, otherwise the actions associated with the write command take place after the STOP condition is received. Write commands with insufficient channel bytes shall be accepted with truncated channel bytes having an implied value of 0h. Bytes beyond the size required for the number of channels and PEC are reserved.

Multiple downstream channels may be simultaneously connected to the upstream channel to bridge them together. All downstream channels shall be disconnected when the NVMe Storage Device is powered off (refer to Figure 165) or by an SMBus Reset (refer to section 8.3.4). Connecting or disconnecting channels while they are active is strongly discouraged and results in undefined behavior.

Figure 188: 2-Wire Mux Channel Descriptor

Bytes	Factory Default	Description
00	00h	**Type (TYP):** This field indicates the type of the 2-Wire Mux Channel Descriptor. This field shall be cleared to 0h.
01	Impl Spec	**Length (LEN):** This field indicates the length of the 2-Wire Mux Channel Descriptor in bytes.
02	Impl Spec	**Count (CNT):** This field indicates the number of 2-Wire Pointers in the 2-Wire Mux Channel Descriptor. The permitted range of values is 0 to 32.
2-Wire Pointer List		
03	Impl Spec	**2-Wire Pointer 0:** This field contains the child index of the first Element Descriptor whose 2-Wire is connected to this channel.
04	Impl Spec	**2-Wire Pointer 1:** This field contains the child index of the second Element Descriptor whose 2-Wire is connected to this channel if the Count field is greater than 1h; otherwise, this field is not present.
…	…	…
CNT+2	Impl Spec	**2-Wire Pointer N:** This field contains the child index of the last Element Descriptor whose 2-Wire is connected to this channel if the Count field is greater than 2h; otherwise, this field is not present. N is equal to the value of the CNT field minus 1h.

8.2.5.6 PCIe Switch Element Descriptor

The PCIe Switch Element Descriptor is shown in Figure 189 and is used to describe a PCIe switch. This Element Descriptor is the child of a single parent and the parent of one or more children.

Figure 189: PCIe Switch Element Descriptor

Bytes	Factory Default	Description
00	06h	**Type (TYP):** This field indicates the type of the Element Descriptor. This field shall be set to the PCIe Switch Element Descriptor Type (i.e., 6h). Refer to Figure 177.
01	Impl Spec	**Revision (REV):** This field indicates the revision of the PCIe Switch Element Descriptor. This field shall be cleared to 0h.
02	Impl Spec	**Length (LEN):** This field indicates the length of the PCIe Switch Element Descriptor in bytes.
03	Impl Spec	**Upstream Switch Port Descriptor (USPD):** This field contains the PCIe Switch Port Descriptor that describes the upstream switch port.
04	Impl Spec	**Downstream Switch Port Descriptor Count (DSPDC):** This field indicates the number of PCIe Port Descriptors associated with downstream switch ports.
PCIe Switch Port Descriptor List		
12:05	Impl Spec	**Downstream Switch Port Descriptor 0:** This field contains the PCIe Switch Port Descriptor associated with the first downstream port.
20:13	Impl Spec	**Downstream Switch Port Descriptor 1:** This field contains the PCIe Switch Port Descriptor associated with the second downstream port if the Downstream Switch Port Descriptor Count field is greater than 1h; otherwise, this field is not present.
…	…	…
(DSPDC*LEN)+4: (DSPDC*LEN)-3	Impl Spec	**Downstream Switch Port Descriptor N:** This field contains the PCIe Switch Port Descriptor associated with the last downstream port if the Downstream Switch Port Descriptor Count field is greater than 2h; otherwise, this field is not present. N is equal to the value of the DSPDC field minus 1h and LEN is the length of the PCIe Switch Port Descriptor.

The PCIe Switch Element Descriptor consists of a list of PCIe Switch Port Descriptors. There is an Upstream Switch Port Descriptor that describes the upstream port and is the child of exactly one parent Element Descriptor. A variable length list of Downstream Switch Port Descriptors describes the downstream ports.

The format of a PCIe Switch Port Descriptor is shown in Figure 190. It describes the PCIe port's supported PCIe link speeds, PCIe maximum link width, reference clock capabilities, and PCIe Port Number. Downstream ports also have a child Element Descriptor and its Destination Port index value.

Figure 190: PCIe Switch Port Descriptor

Bytes	Factory Default	Description
00	00h	**Type (TYP):** This field indicates the type of PCIe Switch Port Descriptor. This field shall be cleared to 0h.
01	Impl Spec	**Length (LEN):** This field indicates the length of the PCIe Switch Port Descriptor in bytes.

Figure 190: PCIe Switch Port Descriptor

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>02</td><td>Impl Spec</td><td>PCIe Link Speed (PCIELS): This field indicates a bit vector of link speeds supported by the PCIe port.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:6</td><td>Reserved</td></tr>
<tr><td>5</td><td>64.0 GT/s Support (GTS64): Set to ‘1’ if the PCIe link supports 64.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>4</td><td>32.0 GT/s Support (GTS32): Set to ‘1’ if the PCIe link supports 32.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>3</td><td>16.0 GT/s Support (GTS16): Set to ‘1’ if the PCIe link supports 16.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>2</td><td>8.0 GT/s Support (GTS8): Set to ‘1’ if the PCIe link supports 8.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>1</td><td>5.0 GT/s Support (GTS5): Set to ‘1’ if the PCIe link supports 5.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>0</td><td>2.5 GT/s Support (GTS2P5): Set to ‘1’ if the PCIe link supports 2.5 GT/s, otherwise cleared to ‘0’.</td></tr>
</table></td></tr>
<tr><td>03</td><td>Impl Spec</td><td>PCIe Maximum Link Width (PCIEMLW): The maximum PCIe link width for this port.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>PCIe x1</td></tr>
<tr><td>2</td><td>PCIe x2</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
<tr><td>4</td><td>PCIe x4</td></tr>
<tr><td>5 to 7</td><td>Reserved</td></tr>
<tr><td>8</td><td>PCIe x8</td></tr>
<tr><td>9 to 11</td><td>Reserved</td></tr>
<tr><td>12</td><td>PCIe x12</td></tr>
<tr><td>13 to 15</td><td>Reserved</td></tr>
<tr><td>16</td><td>PCIe x16</td></tr>
<tr><td>17 to 31</td><td>Reserved</td></tr>
<tr><td>32</td><td>PCIe x32</td></tr>
<tr><td>33 to 255</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>04</td><td>Impl Spec</td><td>RefClk Capability (RCCAP): This field contains a bit vector that specifies the PCIe clocking modes supported by the port.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:4</td><td>Reserved</td></tr>
<tr><td>3</td><td>RefClk Support (RCS): Set to ‘1’ for upstream ports that automatically use RefClk if provided and otherwise uses SRIS, otherwise, cleared to ‘0’. Reserved for downstream ports.</td></tr>
<tr><td>2</td><td>Separate RefClk with SSC Support (SRCSS): Set to ‘1’ if the PCIe port supports Separate RefClk with SSC (SRIS), otherwise cleared to ‘0’.</td></tr>
<tr><td>1</td><td>Separate RefClk with No SSC Support (SRCNSS): Set to ‘1’ if the PCIe port supports Separate RefClk with no SSC (SRNS), otherwise cleared to ‘0’.</td></tr>
<tr><td>0</td><td>Common RefClk Support (CRCS): Set to ‘1’ if the PCIe port supports common RefClk, otherwise cleared to ‘0’.</td></tr>
</table></td></tr>
<tr><td>05</td><td>Impl Spec</td><td>Port Number (PN): This field indicates the PCIe Port Number, as defined by the PCI Express Base Specification, associated with this port.</td></tr>
<tr><td>06</td><td>Impl Spec</td><td>PCIe Pointer (PCIEPTR): In downstream ports this field contains the child index of the Element Descriptor that has a PCIe port connected to this PCIe port. In upstream ports this field is cleared to 0h.</td></tr>
</table>

Figure 190: PCIe Switch Port Descriptor

Bytes	Factory Default	Description
07	Impl Spec	**Destination Port (DPORT):** This field contains the index of the Port Descriptor in the child Element Descriptor. If the child Element Descriptor has one PCIe upstream port (i.e., a PCIe Switch Element Descriptor), this field shall be cleared to 0h.

8.2.5.7 NVM Subsystem Element Descriptor

The NVM Subsystem Element Descriptor is shown in Figure 191 and is used to describe an NVM Subsystem contained in the NVMe Storage Device.

Figure 191: NVM Subsystem Element Descriptor

Bytes[1]	Factory Default	Description
00	07h	**Type (TYP):** This field indicates the type of the Element Descriptor. This field shall be set to the NVM Subsystem Element Descriptor Type (i.e., 7h). Refer to Figure 177.
01	01h	**Revision (REV):** This field indicates the revision of the NVM Subsystem Element Descriptor. This field shall be set to 1h.
02	Impl Spec	**Length (LEN):** This field indicates the length of the NVM Subsystem Element Descriptor in bytes.
03	3Ah or 3Bh	**SMBus/I2C Address Info (SADDRI):** If the NVM Subsystem supports MCTP on all SMBus/I2C Management Endpoints on the 2-Wire port, then this field indicates the 2-Wire address for the MCTP over 2-Wire port and whether or not SMBus ARP is supported; otherwise, this field shall be cleared to 0h. **Bits** / **Description** 7:1 — **SMBus/I2C Address (SADDR):** This field contains the 7-bit 2-Wire address. Refer to Figure 16 for requirements. 0 — **ARP Capable (ARPC):** If SMBus ARP is supported, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'. Refer to Figure 16 for requirements.

Figure 191: NVM Subsystem Element Descriptor

<table>
<tr><th>Bytes[1]</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>04</td><td>Impl Spec</td><td>
2-Wire Capabilities (TWCAP): If the NVM Subsystem supports a 2-Wire port then this field indicates the 2-Wire capabilities; otherwise, this field shall be cleared to 0h.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Reset (RST): This bit is set to ‘1’ if all of the 2-Wire reset mechanisms are supported as defined by the associated form factor specification. This bit is cleared to ‘0’ if the form factor does not define SMBus Reset or the NVMe Storage Device does not support all of the 2-Wire reset mechanisms defined by the specification for the Form Factor in the Host Connector Element Descriptor.</td></tr>
<tr><td>6:5</td><td>MCTP over 2-Wire In Aux Power Support (M2WAS): This field indicates the 2-Wire port support during the Auxiliary Only power state (refer to Figure 165).

This field shall not be cleared to 00b on implementations that are compliant with revision 2.0 or later of this specification.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>Support for this capability is not indicated.</td></tr>
<tr><td>01b</td><td>The 2-Wire port does not support MCTP during the Auxiliary Only power state.</td></tr>
<tr><td>10b</td><td>The 2-Wire port supports MCTP during the Auxiliary Only power state.</td></tr>
<tr><td>11b</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>4</td><td>I3C Support (I3CS): If the 2-Wire port supports MCTP during the Auxiliary Only power state (refer to Figure 165), then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.</td></tr>
<tr><td>3:2</td><td>Reserved</td></tr>
<tr><td>1:0</td><td>Maximum Speed (MSPD): This field is set to the highest supported 2-Wire clock speed.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0</td><td>100 kHz</td></tr>
<tr><td>1</td><td>400 kHz</td></tr>
<tr><td>2</td><td>1 MHz</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
</table>
</td></tr>
</table>
</td></tr>
<tr><td>05</td><td>Impl Spec</td><td>NVM Subsystem Port Descriptor Count (NVMSPDC): This field indicates the number of NVM Subsystem Port Descriptors associated with the NVM Subsystem. The permitted range of values is 1 to 64.</td></tr>
<tr><td colspan="3">NVM Subsystem Port Descriptor List</td></tr>
<tr><td>6+M-1:6</td><td>Impl Spec</td><td>NVM Subsystem Port Descriptor 0: This field contains the NVM Subsystem Port Descriptor associated with the first NVM Subsystem port.</td></tr>
<tr><td>6+(2*M)-1: 6+M</td><td>Impl Spec</td><td>NVM Subsystem Port Descriptor 1: This field contains the NVM Subsystem Port Descriptor associated with the second NVM Subsystem port if the NVM Subsystem Port Descriptor Count field is greater than 1h; otherwise this field is not present.</td></tr>
<tr><td>…</td><td>…</td><td>…</td></tr>
<tr><td>6+(N*M)-1: 6+((N-1)*M)</td><td>Impl Spec</td><td>NVM Subsystem Port Descriptor N: This field contains the NVM Subsystem Port Descriptor associated with the last NVM Subsystem port if the NVM Subsystem Port Descriptor Count field is greater than 2h; otherwise, this field is not present. N is equal to the NVMSPDC field minus 1h.</td></tr>
</table>

Figure 191: NVM Subsystem Element Descriptor

Bytes[1]	Factory Default	Description
X+1:X	Impl Spec	**Management Endpoint Ready Independent of Media Timeout (MERIM)**: This field shall indicate the maximum time in 100 ms units required by a Management Endpoint after entering a power state in which accesses are supported from a power state in which accesses are not supported (refer to section 8.1), to be ready to start processing a Request Message that does not require media access. A value of 0h indicates that no time is indicated. This field shall not be cleared to 0h on implementations that are compliant with revision 2.0 or later of this specification that support SMBus/I2C VPD accesses.
X+3:X+2	Impl Spec	**Management Endpoint Ready With Media Timeout (MERWMTO):** This field shall indicate the estimated maximum time in 100 ms units required by a Management Endpoint after entering a power state in which accesses are supported from a power state in which accesses are not supported (refer to section 8.1), to be ready to start processing a Request Message that requires media access. A value of 0h indicates that no time is indicated. This field shall not be cleared to 0h on implementations that are compliant with revision 2.0 or later of this specification that support SMBus/I2C VPD accesses.
X+5:X+4	Impl Spec	**Maximum Unresponsive Time (MUT):** This field shall indicate the estimated maximum time in 100 ms units once operational (refer to section 8.1) that: • the Management Endpoint is permitted to be unable to service Request Messages or AEMs for any reason (e.g., due to activating a firmware image or due to executing code in a security module that prohibits execution of code outside the security module that is required to process a Request Message); and • SMBus/I2C VPD accesses or 2-Wire Mux accesses are permitted to be unsupported due to the conditions listed in section 8.1. The ability for accesses to the Management Endpoint, SMBus/I2C VPD, and 2-Wire Mux to be unresponsive are independent (e.g., an NVM Subsystem may be in a state where accesses to the Management Endpoint are unresponsive and accesses to the SMBus/I2C VPD are responsive or vice versa). This field shall not include the time to ready for the Management Endpoint indicated by the MERIMTO and MERWMTO fields. A value of 0h indicates that no time is indicated. This field shall not be cleared to 0h on implementations that are compliant with revision 2.0 or later of this specification that support SMBus/I2C VPD accesses.

Notes:
1. When used in this column:
   - N is equal to the value of the NVMSPDC;
   - M is equal to the value of the Length field in the NVM Subsystem Port Descriptor data structure (refer to Figure 173); and
   - X is equal to the starting byte offset of the MERIMTO field (i.e., 6+(NVMSPDC*M)).

Each upstream port is described by an NVM Subsystem Port Descriptor as shown in Figure 192. The NVM Subsystem Port Descriptor describes the PCIe port's supported PCIe link speeds, PCIe max link width, RefClk capabilities, PCIe Port Identifier, and MCTP support. Each NVM Subsystem Port Descriptor should be the child of exactly one parent Element Descriptor.

Figure 192: NVM Subsystem Port Descriptor

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>00</td><td>00h</td><td>Type (TYP): This field indicates the type of an NVM Subsystem Port Descriptor. This field shall be cleared to 0h.</td></tr>
<tr><td>01</td><td>Impl Spec</td><td>Length (LEN): This field indicates the length of the NVM Subsystem Port Descriptor in bytes.</td></tr>
<tr><td>02</td><td>Impl Spec</td><td>PCIe Link Speed (PCIELS): This field indicates a bit vector of link speeds supported by the PCIe port.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:6</td><td>Reserved</td></tr>
<tr><td>5</td><td>64.0 GT/s Support (GTS64): Set to ‘1’ if the PCIe link supports 64.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>4</td><td>32.0 GT/s Support (GTS32): Set to ‘1’ if the PCIe link supports 32.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>3</td><td>16.0 GT/s Support (GTS16): Set to ‘1’ if the PCIe link supports 16.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>2</td><td>8.0 GT/s Support (GTS8): Set to ‘1’ if the PCIe link supports 8.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>1</td><td>5.0 GT/s Support (GTS5): Set to ‘1’ if the PCIe link supports 5.0 GT/s, otherwise cleared to ‘0’.</td></tr>
<tr><td>0</td><td>2.5 GT/s Support (GTS2P5): Set to ‘1’ if the PCIe link supports 2.5 GT/s, otherwise cleared to ‘0’.</td></tr>
</table></td></tr>
<tr><td>03</td><td>Impl Spec</td><td>PCIe Maximum Link Width (PCIEMLW): The maximum PCIe link width for this NVM Subsystem port.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>PCIe x1</td></tr>
<tr><td>2</td><td>PCIe x2</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
<tr><td>4</td><td>PCIe x4</td></tr>
<tr><td>5 to 7</td><td>Reserved</td></tr>
<tr><td>8</td><td>PCIe x8</td></tr>
<tr><td>9 to 11</td><td>Reserved</td></tr>
<tr><td>12</td><td>PCIe x12</td></tr>
<tr><td>13 to 15</td><td>Reserved</td></tr>
<tr><td>16</td><td>PCIe x16</td></tr>
<tr><td>17 to 31</td><td>Reserved</td></tr>
<tr><td>32</td><td>PCIe x32</td></tr>
<tr><td>33 to 255</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>04</td><td>Impl Spec</td><td>RefClk Capability (RCCAP): This field contains a bit vector that specifies the PCIe clocking modes supported by the port.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:4</td><td>Reserved</td></tr>
<tr><td>3</td><td>RefClk Support (RCS): Set to ‘1’ if the device automatically uses RefClk if provided and otherwise uses SRIS, otherwise cleared to ‘0’.</td></tr>
<tr><td>2</td><td>Separate RefClk with SSC Support (SRCSS): Set to ‘1’ if the PCIe link supports Separate RefClk with SSC (SRIS), otherwise cleared to ‘0’.</td></tr>
<tr><td>1</td><td>Separate RefClk with No SSC Support (SRCNSS): Set to ‘1’ if the PCIe link supports Separate RefClk with no SSC (SRNS), otherwise cleared to ‘0’.</td></tr>
<tr><td>0</td><td>Common RefClk Support (CRCS): Set to ‘1’ if the PCIe link supports common RefClk, otherwise cleared to ‘0’.</td></tr>
</table></td></tr>
<tr><td>05</td><td>Impl Spec</td><td>Port Identifier (PORTID): This field contains the NVMe-MI Port Identifier associated with this port.</td></tr>
</table>

Figure 192: NVM Subsystem Port Descriptor

<table>
<tr><th>Bytes</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>6</td><td>Impl Spec</td><td>MCTP Support (MCTPS): This field indicates a bit vector that specifies the level of support for the NVMe Management Interface.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:1</td><td>Reserved</td></tr>
<tr><td>0</td><td>MCTP PCIe VDM Support (MCTPPCIEVS): If MCTP-based NVMe-MI Messages are supported on this PCIe port, then this bit shall be set to ‘1’; otherwise, this bit shall be cleared to ‘0’.</td></tr>
</table></td></tr>
</table>

8.2.5.8 FRU Information Device Element Descriptor

The FRU Information Device Element Descriptor is shown in Figure 193 and is used to describe a FRU Information Device contained in the NVMe Storage Device.

Figure 193: FRU Information Device Element Descriptor

<table>
<tr><th>Byte Offset</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>00</td><td>08h</td><td>Type (TYP): This field indicates the type of the Element Descriptor. This field shall be set to the FRU Information Device Element Descriptor Type (i.e., 8h). Refer to Figure 177.</td></tr>
<tr><td>01</td><td>00h</td><td>Revision (REV): This field indicates the revision of the FRU Information Device Element Descriptor. This field shall be set to 1h.</td></tr>
<tr><td>02</td><td>06h</td><td>Length (LEN): This field indicates the length of the FRU Information Device Element Descriptor in bytes.</td></tr>
<tr><td>03</td><td>A6h/A7h or 0h for NVM Storage Devices A4h/A5h or 0h for Carriers</td><td>2-Wire Address Info (TWADDRI): If the NVMe Storage Device contains a 2-Wire port, then this field indicates the default 2-Wire addressing per the table below; else, this field shall be cleared to 0h.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7:1</td><td>2-Wire Address (TWADDR): This field contains the 7-bit 2-Wire address. Refer to Figure 16 for requirements.</td></tr>
<tr><td>0</td><td>ARP Capable (ARPC): If this bit is set to ‘1’, then SMBus ARP is supported. If this bit is cleared to ‘0’, then SMBus ARP is not supported. Refer to the SMBus Specification for additional details.</td></tr>
</table></td></tr>
</table>

Figure 193: FRU Information Device Element Descriptor

<table>
<tr><th>Byte Offset</th><th>Factory Default</th><th>Description</th></tr>
<tr><td>04</td><td>Impl Spec</td><td>2-Wire Capabilities (TWCAP): If the NVM Storage Device contains a 2-Wire port, then this field indicates the 2-Wire capabilities per the table below; else, this field shall be cleared to 0h.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>Reset (RST): If this bit is set to ‘1’, then all of the 2-Wire reset mechanisms are supported as defined by the specification for the Form Factor in the Host Connector Element Descriptor.

If this bit is cleared to ‘0’, then the FRU Information Device does not support all of the 2-Wire reset mechanisms defined by the specification for the Form Factor in the Host Connector Element Descriptor.</td></tr>
<tr><td>6</td><td>I2C Writes Allowed (I2CWA): If this bit is set to ‘1’, then the FRU Information Device is allowed to be written using an I2C Write operation.

If this bit is cleared to ‘0’, then the FRU Information Device is not allowed to be written using an I2C Write operation.</td></tr>
<tr><td>5</td><td>Boot Failure Code Support (BFCS): If SMBus/I2C VPD accesses are supported (refer to section 8.1) and the Boot Failure Code field (refer to Figure 169) is supported, then this bit shall be set to ‘1’; otherwise, this bit shall be cleared to ‘0’.

This bit shall not be cleared to ‘0’ on implementations that are compliant with revision 2.0 or later of this specification that support SMBus/I2C VPD accesses.</td></tr>
<tr><td>4:2</td><td>Reserved</td></tr>
<tr><td>1:0</td><td>Maximum Speed (MSPD): This field is set to the highest supported 2-Wire clock speed supported by the FRU Information Device.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0</td><td>100 kHz</td></tr>
<tr><td>1</td><td>400 kHz</td></tr>
<tr><td>2</td><td>1 MHz</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
</table></td></tr>
</table></td></tr>
<tr><td>05</td><td>8h to 10h inclusive</td><td>Maximum FRU Information Device Size (MFIDS): The maximum size of the FRU Information Device is 2^N bytes where N is the value in this field (e.g., a value of 8 in this field indicates a maximum FRU Information Device size of 2^8 or 256 bytes).</td></tr>
</table>

8.2.5.9 Vendor-Specific Element Descriptors

The Vendor-Specific Element Descriptor is shown Figure 194.

Figure 194: Vendor-Specific Element Descriptors

Bytes	Factory Default	Description
00	Impl Spec	**Type (TYP):** This field indicates the type of the Element Descriptor. Vendor-Specific Types have a value in the range of F0h to FFh.
01	Impl Spec	**Revision (REV):** This field indicates the revision of the Element Descriptor. The Vendor-Specific Element Descriptor Revision is determined by the Vendor.
02	Impl Spec	**Length (LEN):** This field indicates the length of the Vendor-Specific Element Descriptor in bytes.
04:03	Impl Spec	**PCI Vendor ID (PCIVID):** This field indicates PCI-SIG assigned vendor identifier.
LEN-1: 05	Impl Spec	**Vendor Specific (VS):** Vendor-specific information.

8.3 Reset Architecture

This section describes the reset architecture defined by this specification that is applicable to NVMe Storage Devices and NVMe Enclosures. Additional requirements and recommendations for resets are specified elsewhere in this specification.

8.3.1 NVM Subsystem Reset

An NVM Subsystem Reset is initiated under the conditions outlined in the NVM Express Base Specification (e.g., when main power is applied to the NVM Subsystem) and section 5.8 of this specification.

An NVM Subsystem Reset initiated via the out-of-band mechanism may interfere with a host. A Management Controller should coordinate with the host. Coordination between a Management Controller and a host are outside the scope of this specification.

When an NVM Subsystem Reset is initiated, the entire NVM Subsystem shall be reset. This includes all NVM Subsystem ports (PCIe and 2-Wire), 2-Wire elements (e.g., 2-Wire Management Endpoints, FRU Information Devices, 2-Wire Muxes, etc.), PCIe VDM Management Endpoints, and Controller Management Interfaces. If an NVMe Storage Device does not contain a 2-Wire port, then a NVM Subsystem Reset should reset the FRU Information Device if the FRU Information Device supports a reset mechanism. On an NVM Subsystem Reset, any internal state of the NVM Subsystem should be returned to its power-on condition.

8.3.2 Controller Level Reset

A Controller Level Reset is initiated under the conditions outlined in the NVM Express Base Specification.

A Controller Level Reset initiated via the out-of-band mechanism may interfere with a host. A Management Controller should coordinate with the host. Coordination between a Management Controller and a host are outside the scope of this specification.

The actions performed on a Controller Level Reset are outlined in the NVM Express Base Specification. A Controller Level Reset shall have no effect on the Controller Management Interface associated with that Controller, the PCI Express port associated with that Controller, or a Management Endpoint associated with that port. The servicing of any Management Interface Command Set commands, NVM Express Admin Command Set commands, or Control Primitives shall be independent of and not affected by any one or more Controllers in the NVM Subsystem being disabled or being reset by a Controller Level Reset unless the Management Endpoint servicing the NVMe-MI Request is reset (e.g., due to an NVM Subsystem Reset or due to a PCIe Reset of the PCIe VDM Management Endpoint servicing the NVMe-MI Request). A Controller Level Reset shall have no effect on the AEM servicing model (refer to section 4.4).

A Controller Level Reset may prevent PCIe Commands from being processed on that Controller (refer to section 8.1). If a PCIe Command is prevented from being processed due to a Controller Level Reset that does not reset the Management Endpoint, then that PCIe Command shall be completed with status PCIe Inaccessible.

On a Controller Level Reset, any internal state of the Controller should be returned to its power-on condition.

A Controller Level Reset that causes a new firmware image to activate is considered a special event and may impact the operation of the Controller Management Interface associated with one or more Controllers, servicing of NVMe-MI Messages, or Management Endpoints within an NVM Subsystem. This impact is unspecified and vendor specific. The Management Controller and host should coordinate the activation of a new firmware image. Coordination between a Management Controller and a host are outside the scope of this specification.

Additional requirements and recommendations for Controller Level Resets are specified elsewhere in this specification. For example, bits and fields that are dedicated to each Controller in the in-band tunneling mechanism are reset as defined in Figure 97, Figure 98, and Figure 107.

8.3.3 Management Endpoint Reset

The following shall cause a Management Endpoint Reset:

- an NVM Subsystem Reset of the NVM Subsystem containing the Management Endpoint; or
- the conditions for resetting an MCTP endpoint outlined in the MCTP Base Specification or the associated MCTP transport binding specifications.

In addition to these conditions, a Management Endpoint Reset shall be initiated on the Management Endpoint associated with a PCI Express port when that PCI Express port undergoes a PCIe Reset (refer to section 8.3.5) or is powered on. A Management Endpoint Reset shall be initiated on the Management Endpoint associated with a 2-Wire port when that 2-Wire port undergoes an SMBus Reset (refer to section 8.3.4) or is powered on.

If a Management Endpoint Reset is initiated, then:

- each Command Slot in that Management Endpoint shall behave as if an implicit Abort Control Primitive (refer to section 4.2.1.3) was received with the exception that the Management Endpoint shall not transmit any Abort Control Primitive Response Messages;
- any Control Primitives being processed by that Management Endpoint shall be dropped (silently discarded); and
- any internal state of that Management Endpoint should be returned to its power-on condition.

A Management Endpoint Reset of a Management Endpoint shall not affect any other Management Endpoint or entity in the NVM Subsystem. Note that for implementations compliant to version 1.1 and earlier of this specification, implementations may block MCTP accesses on additional Management Endpoints during a PCI Express conventional reset of a PCIe VDM Management Endpoint.

Additional requirements and recommendations for Management Endpoint Resets are specified elsewhere in this specification. For example, a Management Endpoint Reset:

- resets bits and fields that are dedicated to each Management Endpoint in the out-of-band mechanism as defined in Figure 97, Figure 98, and Figure 107;
- disables all supported AEs as defined by Figure 83;
- stops transmission of any AEM as defined in section 4.4.3;
- causes the Management Endpoint to transition to the AE Disarmed State as defined in section 4.4.1;
- clears the AEM Transmission Failure bit as defined in Figure 108;
- resets the value of the Composite Controller Status Flags field as defined by Figure 107;
- resets the value of the SMBus/I2C Frequency field as defined by Figure 78;
- resets the value of the MCTP Transmission Unit Size field as defined by Figure 79; and
- clears the Control Primitive Specific Response field to 0h as defined in Figure 45.

8.3.4 2-Wire Resets

The 2-Wire port may be reset in multiple ways to restore 2-Wire communications. Some 2-Wire Resets are only applicable for specific 2-Wire modes or device form factors. Some 2-Wire Resets also reset the associated 2-Wire Management Endpoint.

Clock-low recovery is the ability to reset communication on all 2-Wire Management Endpoints on a 2-Wire port when the 2-Wire clock on that 2-Wire port is low for longer than $t_{TIMEOUT,MIN}$ (refer to the SMBus Specification). 2-Wire Management Endpoints shall support clock-low recovery. It is strongly recommended that any 2-Wire element other than the 2-Wire Management Endpoint (refer to Figure 16 for a list of 2-Wire elements) should support clock-low recovery. Clock-low recovery:

- shall cause a Management Endpoint Reset;
- shall not reset ARP-assigned addresses to their default values; and
- shall switch the 2-Wire port to SMBus mode if the 2-Wire port was in I3C mode.

Some form factor specifications may also specify one or more form factor-specific mechanisms to reset the 2-Wire port (e.g., SMRST# as defined in the SNIA SFF-TA-1009 Enterprise and Datacenter Standard Form Factor Pin and Signal Specification, or the rising edge of the +3.3 Vaux rail as defined in the PCI Express SFF-8639 Module Specification). Any form factor-specific mechanisms to reset the 2-Wire port supported by an NVMe Storage Device or NVMe Enclosure shall cause:

- a Management Endpoint Reset; and
- shall switch the 2-Wire port to SMBus mode if the 2-Wire port was in I3C mode.

An NVM Subsystem Reset also causes a 2-Wire Reset and a Management Endpoint Reset. If the NVM Subsystem Reset is due to application of main power, then the 2-Wire port shall switch to SMBus mode; otherwise, the NVM Subsystem Reset shall not change the 2-Wire port's current SMBus or I3C mode.

The Target Reset Pattern as defined by the MIPI I3C Basic Specification may also cause a 2-Wire Reset for 2-Wire ports in I3C mode. The default case for the Target Reset Pattern shall reset the I3C physical layer. The default case shall not cause a Management Endpoint Reset and shall not reset the I3C Dynamically Assigned Address. Target Reset Patterns shall not change the 2-Wire port's current SMBus or I3C mode.

2-Wire ports in I3C mode may optionally support the RSTACT CCC as defined by the MIPI I3C Basic Specification. RSTACT uses Defining Bytes to modify the behavior of the immediately subsequent Target Reset Pattern as follows:

- Defining Byte value 0 shall not cause a 2-Wire Reset;
- Defining Byte value 1 shall behave the same as a default Target Reset Pattern; and
- Defining Byte value 2 shall cause a Management Endpoint Reset and shall reset the I3C Dynamically Assigned Address.

If a 2-Wire Reset causes a Management Endpoint Reset, then the 2-Wire physical layer shall also be reset. For any 2-Wire element other than the 2-Wire Management Endpoint (refer to Figure 16 for a list of 2-Wire elements), it is strongly recommended that an SMBus Reset should reset that 2-Wire element.

An 2-Wire Reset shall cause a 2-Wire reset mechanism defined for the Expansion Connector to be applied to each Expansion Connector in the NVMe Storage Device.

If a 2-Wire Management Endpoint is transmitting a Response Message, then an SMBus Reset shall cause the 2-Wire port to attempt to generate a STOP condition (refer to the SMBus Specification) within 5 ms from the assertion of a 2-Wire Reset. The 2-Wire port's transmitter shall remain in the bus idle condition (refer to the SMBus Specification and MIPI I3C Basic Specification) and the 2-Wire port's receiver shall ignore any incoming traffic for the remainder of the 2-Wire Reset assertion. A 2-Wire port shall support 2-Wire accesses starting from the de-assertion of 2-Wire Reset within the same timing constraints as are applicable to transitioning from an unsupported to a supported power state as defined in section 8.1.

Additional requirements and recommendations for 2-Wire Resets are specified elsewhere in this specification. For example, a 2-Wire Reset:

- resets the value of the SMBus/I2C Frequency field as defined in Figure 77; and
- should clear the internal Command Offset for the VPD to 0h as defined in section 8.2.

8.3.5 PCIe Reset

A PCIe Reset is generated by:

a) a Conventional Reset (refer to the PCI Express Base Specification); or
b) a Function Level Reset (refer to the PCI Express Base Specification).

PCIe Resets have additional impacts on in-band traffic and NVMe Controller operations which are outside the scope of this specification.

A Conventional Reset shall cause a Management Endpoint Reset of all PCIe VDM Management Endpoints associated with the PCI Express port being reset. A Function Level Reset shall cause a Management Endpoint Reset of the PCIe VDM Management Endpoint associated with the Function being reset.

A PCIe VDM Management Endpoint shall support PCIe MCTP accesses after a PCIe Reset is de-asserted within the same timing constraints as are applicable to transitioning from an unsupported to a supported NVM Subsystem power state as defined by section 8.1.

8.4 Security

The Responder may respond with a Response Message Status of Access Denied in an Error Response. While a drive is in an unlocked state, this mechanism shall not be used for the Management Interface Command Set or the NVMe Admin Command Set.

The commands and the times at which such a response is generated is vendor specific. The mechanism used to lock a drive is outside the scope of this specification.

8.5 Shutdown Impacts

If shutdown processing is reported as in progress or is reported as complete (i.e., the value of the CSTS.SHST field as defined by the NVM Express Base Specification is set to 01b or is set to 10b) on a Controller, then media may be in the shutdown state. If the value of the CSTS.SHST field is cleared to 00b (i.e., normal operation), then the media is able to be in the shutdown state under specific conditions defined by the NVM Express Base Specification (e.g., after initial power application or after the value of the CSTS.SHST field has been cleared to 00b by a Controller Level Reset while shutdown processing is reported as complete).

If the media is in the shutdown state, then Controller processing of any NVMe Admin Command that is received over the out-of-band mechanism and that requires access to media (refer to the Admin Commands Permitted to Return a Status Code of Admin Command Media Not Ready section in the NVM Express Base Specification) may be impacted by the Ignore Shutdown bit (ISH) as specified in Figure 195.

If an NVMe Admin Command that requires access to media and specifies the ISH bit set to ‘1’ is processed by any Controller in the NVM Subsystem while shutdown processing is reported as in progress or is reported as complete (i.e., the value of the CSTS.SHST field is set to 01b or 10b), then the NVM Subsystem may transition the media out of the shutdown state. If the NVM Subsystem loses main power while the media is not in the shutdown state, then the Unexpected Power Losses field in the SMART / Health Information log page is incremented (refer to the NVM Express Base Specification).

If the media is transitioned out of the shutdown state by any Controller in the NVM Subsystem processing an NVMe Admin Command that specifies the ISH bit set to ‘1’, then the NVM Subsystem is permitted to transition the media back into the shutdown state after processing of that NVMe Admin Command is completed. If the media is not transitioned back into the shutdown state and main power is lost while the media is not in the shutdown state, then the Unexpected Power Losses field is incremented. Whether or not an NVM Subsystem transitions the media back into the shutdown state is implementation specific.

In all cases where an NVMe Admin Command that requires access to media is processed by any Controller in the NVM Subsystem while shutdown processing is reported as in progress or is reported as complete (i.e., the value of the CSTS.SHST field is set to 01b or 10b) in Figure 195, if the NVMe Admin Command is aborted with the Status field in Completion Queue Entry Dword 3 set to a value of Commands Aborted due to Power Loss Notification or Admin Command Media Not Ready, then media shall not be transitioned out of the shutdown state.

If shutdown processing is reported as in progress or is reported as complete (i.e., the value of the CSTS.SHST field is set to 01b or 10b), then:

- there shall be no impact on access to the FRU Information Device; and
- there shall be no impact on the out-of-band mechanism other than for NVMe Admin Commands that access media as described in this section.

If the Controller is in normal operation (i.e., the value of the CSTS.SHST field is cleared to 00b), then:

- there shall be no impact on access to the FRU Information Device; and
- there shall be no impact on the out-of-band mechanism.

If an NVMe Admin Command does not require access to media, then the ISH bit shall have no effect on the processing of that NVMe Admin Command.

Figure 195: Shutdown Interactions with NVMe Admin Commands that Access Media

<table>
<tr><th rowspan="2">CSTS.SHST Field Value at the Time the Controller Started Processing the NVMe Admin Command</th><th colspan="2">ISH Bit Value</th></tr>
<tr><th>ISH Bit Cleared to ‘0’</th><th>ISH Bit Set to ‘1’</th></tr>
<tr><td rowspan="2">00b: Normal operation (e.g., no shutdown has been requested, or the CSTS.SHST field has been cleared to 00b by a Controller Level Reset after shutdown processing is reported as complete)</td><td colspan="2">If a shutdown is requested (i.e., the value of the CC.SHN field as defined by the NVM Express Base Specification transitions from 00b to 01b for normal shutdown or 10b for abrupt shutdown) while the NVMe Admin Command is being processed, then the NVMe Admin Command may be aborted by the Controller. If the NVMe Admin Command is aborted, then the Status field in Completion Queue Entry Dword 3 shall be set to a value of Commands Aborted due to Power Loss Notification.</td></tr>
<tr><td>If media is in the shutdown state when the NVMe Admin Command is processed, then the NVMe Admin Command may be aborted by the Controller and the Status field in Completion Queue Entry Dword 3 shall be set to a value of Admin Command Media Not Ready.</td><td>The NVMe Admin Command shall not be aborted by the Controller with the Status field in Completion Queue Entry Dword 3 set to Commands Aborted due to Power Loss Notification or Admin Command Media Not Ready. If media is in the shutdown state when the NVMe Admin Command that requires media access is processed, then the media shall be transitioned out of the shutdown state and then the NVMe Admin Command shall be processed. Since the media is required to be transitioned out of the shutdown state, the NVMe Admin Command processing may take longer than normal. If the processing takes longer than the maximum Request-To-Response time, then a More Processing Required Response is transmitted as specified in section 4.1.2.3.</td></tr>
</table>

Figure 195: Shutdown Interactions with NVMe Admin Commands that Access Media

CSTS.SHST Field Value at the Time the Controller Started Processing the NVMe Admin Command	ISH Bit Value	
	ISH Bit Cleared to '0'	**ISH Bit Set to '1'**
01b: Shutdown processing in progress	The NVMe Admin Command may be aborted by the Controller. If the NVMe Admin Command is aborted, then the Status field in Completion Queue Entry Dword 3 shall be set to a value of Commands Aborted due to Power Loss Notification.	The NVMe Admin Command shall not be aborted by the Controller with the Status field in Completion Queue Entry Dword 3 set to Commands Aborted due to Power Loss Notification or Admin Command Media Not Ready. If media is in the shutdown state when the NVMe Admin Command that requires media access is processed, then the media shall be transitioned out of the shutdown state and then the NVMe Admin Command shall be processed. Since the media is required to be transitioned out of the shutdown state, the NVMe Admin Command processing may take longer than normal. If the processing takes longer than the maximum Request-To-Response time, then a More Processing Required Response is transmitted as specified in section 4.1.2.3. If the value of the CSTS.SHST field at the time the Controller started processing the NVMe Admin Command was set to 01b (i.e., shutdown processing in progress), then the value of the CSTS.SHST field shall transition to a value of 10b (i.e., shutdown processing complete) within the same amount of time as if no NVMe Admin Command that specifies the ISH bit set to '1' had been processed (e.g., for controller shutdown, within the amount of time indicated by the RTD3 Entry Latency field as defined by the NVM Express Base Specification if RTD3 Entry Latency is reported).
10b: Shutdown processing complete	The NVMe Admin Command may be aborted by the Controller. If the NVMe Admin Command is aborted, then the Status field in Completion Queue Entry Dword 3 shall be set to a value of Admin Command Media Not Ready.	

Appendix A Technical Note: NVM Express Basic Management Command

This appendix describes the NVMe Basic Management Command and is included here for informational purposes only. The NVMe Basic Management Command is not formally a part of this specification and its features are not tested by the NVMe Compliance program. No further enhancements to the NVMe Basic Management Command are planned, and it is strongly recommended that any consumers of the NVMe Basic Management Command transition to using the standard NVMe-MI protocol.

This specification utilizes Management Component Transport Protocol (MCTP) messages. The NVMe Basic Management Command does not utilize MCTP. Support for the NVMe Basic Management Command is optional. The NVMe Basic Management Command only works while the 2-Wire port is in SMBus mode.

This command does not provide any mechanism to modify or configure the NVMe device. Modifying or configuring the NVMe device requires the more capable MCTP protocol rather than this command's SMBus Block Read. The host may reuse existing SMBus or FRU Information Device read subroutines for this read.

The block read protocol is specified by the SMBus Specification which is available online at www.smbus.org. First SMBus address write and command code bytes are transmitted by the host, then a repeated start and finally a SMBus address read. The host keeps clocking as the drive responds with the selected data. The command code is used as a starting offset into the data block shown in Figure 173, like an address on a serial EEPROM.

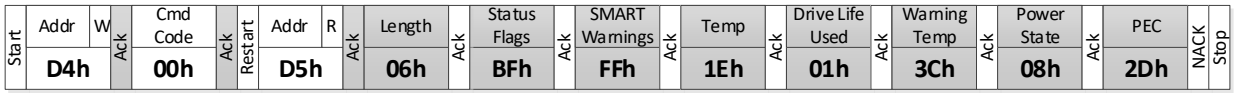
The offset value increments on every byte read and is reset to 0h on a stop condition. A read command without a repeated start is permissible and starts transmission from offset zero. Reading more than the block length with an I2C read is also permissible and these reads continue into the first byte in the next block of data. The Packet Error Code (PEC) accumulates all bytes sent or received after the start condition and the current value is inserted whenever a PEC field is reached.

Blocks of data are packed sequentially. The first 2 blocks are defined by the NVMe-MI workgroup. The first block is the dynamic host health data. The second block includes the Vendor ID (VID) and serial number of the drive. Additional blocks of data may be defined by the owner of the VID. Reading past the end of the vendor defined blocks shall return zeros.

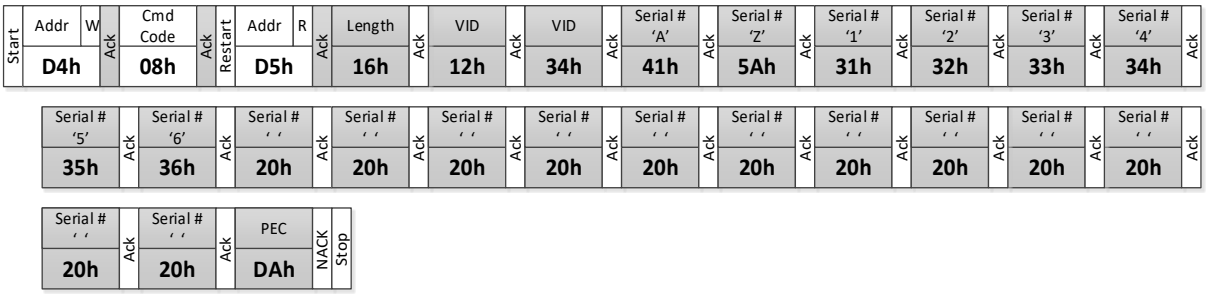
The SMBus address to read this data structure defaults to D4h. After the Management Controller successfully assigns the MCTP UDID to D4h using ARP, then the Basic Management Command may track and respond to reads at future ARP assigned MCTP addresses. This method of changing the Basic Command address is optional and does not persist through power cycles. Interleaved MCTP and block read traffic is permissible and neither command type shall disturb the state of the other commands.

Here are a few example reads from an NVMe drive at 30 °C, no alarms, VID=1234h, serial number is AZ123456 using the format defined in Figure 173. Host transmissions are shown in white blocks and drive responses are shown in grey blocks:

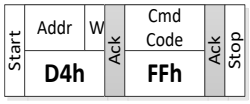
Example 1: SMBus block read of the drive's status (status flags, SMART warnings, temperature):



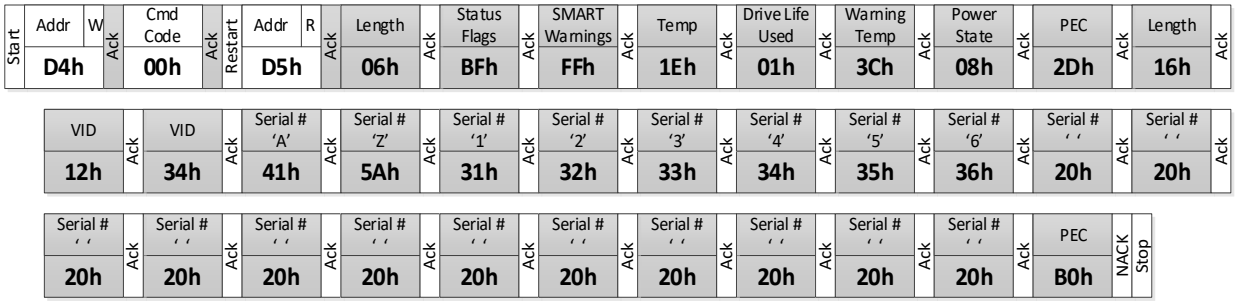
Example 2: SMBus block read of the drive's static data (VID and serial number):



Example 3: SMBus send byte to reset Arbitration bit:



Example 4: I2C read of status and vendor content, I2C allows reading across SMBus block boundaries:



The SMBus Arbitration bit may be used for simple arbitration on systems that have multiple drives on the same SMBus channel without ARP or muxes to separate them. To use this mechanism, the host performs the following process to handle collisions for the same SMBus address:

1. The host does an SMBus byte write to send byte FFh which clears the SMBus Arbitration bit on all listening Management Endpoints at this SMBus address;
2. The host does an I2C read starting from offset 0h and continuing at least through the serial number in the second block. The drive transmitting a '0' when other drives sent a '1' wins arbitration and sets the arbitration bit to '1' upon read completion to give other drives priority on the next read;
3. Repeat step 2 until all drives are read, host receiving the Arbitration bit as a '1' indicates loop is done; and
4. Sort the responses by serial number since the order of drive responses varies with health status and temperatures.

Be careful that there are no short reads of similar data between steps 1 and 3. If the read data is exactly the same on multiple drives, then all these drives set the arbitration bit. After that a new send byte FFh is required to restart the process.

The logic levels were intentionally inverted to normally high in the bytes 1 and 2. This is an additional mechanism to assist systems that do not have ARP or muxes. Since '0' bits win arbitration on SMBus, a drive with an alarm condition is prioritized over healthy drives in the above arbitration scheme. A single I2C read of byte of two bytes starting at offset one from an array of drives detects alarm conditions. Note that only one drive with an alarm may be reliably detected because drives without the same alarm stop

transmitting once the bus contention is detected. For this reason, the bits are sorted in order of priority. Continuing to read further provides the serial number of the drive that had the alarm.

Figure 196: Subsystem Management Data Structure

<table>
<tr><th>Command Code</th><th>Offset (byte)</th><th>Description</th></tr>
<tr><td rowspan="3">0</td><td>00</td><td>Length of Status (LOS): Indicates number of additional bytes to read before encountering PEC. This value should always be 6 (06h) in implementations of this version of the spec.</td></tr>
<tr><td>01</td><td>Status Flags (SFLGS): This field indicates the status of the NVM Subsystem.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>SMBus Arbitration (SARB): This bit is set to ‘1’ after an SMBus block read is completed all the way to the stop bit without bus contention and this bit is cleared to ‘0’ if an SMBus Send Byte FFh is received on this SMBus address.</td></tr>
<tr><td>6</td><td>Drive Not Ready (DNR): This bit is set to ‘1’ when the NVM Subsystem is not capable of processing NVMe management commands, and the rest of the transmission may be invalid. If this bit is cleared to ‘0’, then the NVM Subsystem is fully powered and ready to respond to management commands. This logic level intentionally identifies and prioritizes powered up and ready drives over their powered off neighbors on the same SMBus channel.</td></tr>
<tr><td>5</td><td>Drive Functional (DF): This bit is set to ‘1’ to indicate an NVM Subsystem is functional. If this bit is cleared to ‘0’, then there is an unrecoverable failure in the NVM Subsystem and the rest of the transmission may be invalid. Note that this bit may default to ‘0’ after reset and transition to ‘1’ after the NVM Subsystem has completed initialization and this case should not be considered an error.</td></tr>
<tr><td>4</td><td>Reset Not Required (RNR): This bit is set to ‘1’ to indicate the NVM Subsystem does not require a reset to resume normal operation. If this bit is cleared to ‘0’, then the NVM Subsystem has experienced an error that prevents continued normal operation. A Controller Level Reset is required to resume normal operation.</td></tr>
<tr><td>3</td><td>Port 0 PCIe Link Active (P0PCIELA): This bit is set to ‘1’ to indicate the first port’s PCIe link is up (i.e., the Data Link Control and Management State Machine is in the DL_Active state). If this bit is cleared to ‘0’, then the PCIe link is down.</td></tr>
<tr><td>2</td><td>Port 1 PCIe Link Active (P1PCIELA): This bit is set to ‘1’ to indicate the second port’s PCIe link is up. If this bit is cleared to ‘0’, then the second port’s PCIe link is down or not present.</td></tr>
<tr><td>1:0</td><td>Not Used (NUSED): These bits shall be set to 11b.</td></tr>
</table></td></tr>
<tr><td>02</td><td>SMART Warnings (SMTW): This field shall contain the Critical Warning field (byte 0) of the NVMe SMART / Health Information log page. Each bit in this field shall be inverted from the NVMe definition (i.e., the management interface shall indicate a ‘0’ value while the corresponding bit is ‘1’ in the log page). Refer to the NVM Express Base Specification for bit definitions.

If there are multiple Controllers in the NVM Subsystem, the Management Endpoint shall combine the Critical Warning field from every Controller such that a bit in this field is:
Cleared to ‘0’ if any Controller in the NVM Subsystem indicates a critical warning for that corresponding bit.
Set to ‘1’ if all Controllers in the NVM Subsystem do not indicate a critical warning for the corresponding bit.</td></tr>
</table>

Figure 196: Subsystem Management Data Structure

<table>
<tr><th>Command Code</th><th>Offset (byte)</th><th>Description</th></tr>
<tr><td rowspan="3"></td><td>03</td><td>Composite Temperature (CTemp): This field indicates the current temperature in degrees Celsius. If a temperature value is reported, it should be the same temperature as the Composite Temperature from the SMART / Health Information log page of the hottest Controller in the NVM Subsystem. The reported temperature range is vendor specific, and shall not exceed the range -60 °C to +127°C.

This field should not report a stale temperature, which means that it was sampled more than 5 s prior. If recent data is not available, the Management Endpoint should indicate a value of 80h for this field.

The field values are shown below.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00h to 7Eh</td><td>Temperature is measured in degrees Celsius (0 °C to 126 °C)</td></tr>
<tr><td>7Fh</td><td>127 °C or higher</td></tr>
<tr><td>80h</td><td>No temperature data or temperature data is more the 5 s old.</td></tr>
<tr><td>81h</td><td>Temperature sensor failure</td></tr>
<tr><td>82h to C3h</td><td>Reserved</td></tr>
<tr><td>C4h</td><td>Temperature is -60 °C or lower</td></tr>
<tr><td>C5h to FFh</td><td>Temperature measured in degrees Celsius is represented in two’s complement (-1 °C to -59 °C)</td></tr>
</table></td></tr>
<tr><td>04</td><td>Percentage Drive Life Used (PDLU): Contains a vendor specific estimate of the percentage of NVM Subsystem NVM life used based on the actual usage and the manufacturer’s prediction of NVM life. If an NVM Subsystem has multiple Controllers the highest value is returned. A value of 100 indicates that the estimated endurance of the NVM in the NVM Subsystem has been consumed but may not indicate an NVM Subsystem failure. The value is allowed to exceed 100. Percentages greater than 254 shall be represented as 255. This value should be updated once per power-on hour and equal the Percentage Used value in the SMART / Health Information log page.</td></tr>
<tr><td>05</td><td>Current Over Temperature Warning Threshold (COTWT): This optional field indicates the composite temperature over temperature warning threshold in degrees Celsius. This is intended to initially match the temperature reported in the WCTEMP field in the Identify Controller data structure (refer to the NVM Express Base Specification). If the Over Temperature threshold for Composite Temperature is modified with set features, then the most recent value should be reported. The data format should match the same single byte format as the CTemp field with a range from -60 C to 127 C. A value of 0h means that this field is not reported or that the threshold is set to 0 C.</td></tr>
</table>

Figure 196: Subsystem Management Data Structure

<table>
<tr><th>Command Code</th><th>Offset (byte)</th><th>Description</th></tr>
<tr><td rowspan="2"></td><td>06</td><td>Current Power (CPWR): This optional field reports the current NVM Subsystem power consumption. If both bit mapped fields are cleared to 0h, then this field is not reported.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>7</td><td>NVM Subsystem Idle (NVMSI): This bit is set to ‘1’ when the NVM Subsystem is idle and has been idle for at least 5 s. Refer to the NVMe Idle Power (IDLP) definition.</td></tr>
<tr><td>6:0</td><td>NVM Subsystem Power (NVMSP): This field reports the ceiling function of the power consumed by the NVM Subsystem in watts. If the power consumed by the NVM Subsystem in watts is greater than or equal to 127 W, then 127 W is reported in this field.

Power reported by the NVM Subsystem is determined in the following manner. If the NVMSI bit is set to ‘1’, then the value returned in this field is:
• equal to the value reported in the Idle Power (IDLP) field in the Power State Descriptor data structure for the corresponding NVMe power state if the IDLP field is set to a non-zero value; or
• equal to the value reported in the Maximum Power (MP) field in the Power State Descriptor data structure for the corresponding NVMe power state, if the IDLP field is cleared to 0h.

If the NVMSI bit is cleared to ‘0’, then the value returned in this field is:
• equal to the power reported by the Active Power (ACTP) field in the Power State Descriptor Structure for the corresponding NVMe power state if the ACTP field is set to a non-zero value; or
• equal to the value reported in the Maximum Power (MP) field in the Power State Descriptor data structure for the corresponding NVMe power state, if the APW field is cleared to 0h.</td></tr>
</table></td></tr>
<tr><td>07</td><td>Packet Error Code (PEC): An 8 bit CRC calculated over the SMBus address, command code, second SMBus address, and returned data. The algorithm is defined in the SMBus Specification.</td></tr>
<tr><td rowspan="4">8</td><td>08</td><td>Length of Identification (LENID): Indicates number of additional bytes to read before encountering PEC. This value should always be 22 (16h) in implementations of this version of the spec.</td></tr>
<tr><td>10:09</td><td>Vendor ID (VID): The 2 byte Vendor ID, assigned by the PCI-SIG. Should match VID in the Identify Controller command response. Note the MSB is transmitted first.</td></tr>
<tr><td>11:30</td><td>Serial Number (SN): 20 characters that match the serial number in the NVMe Identify Controller command response. Note the first character is transmitted first.</td></tr>
<tr><td>31</td><td>Packet Error Code (PEC): An 8 bit CRC calculated over the SMBus address, command code, second SMBus address, and returned data. The algorithm is defined in the SMBus Specification.</td></tr>
<tr><td>32+</td><td>32:255</td><td>Vendor Specific (VS): These data structures shall not exceed the maximum read length of 255 specified in the SMBus version 3 specification. Preferably their lengths are not greater than 32 for compatibility with SMBus 2.0.</td></tr>
</table>

Appendix B Example MCTP Messages & Message Integrity Check

Below are artificial MCTP Messages with their corresponding Message Integrity values. Figure 199 shows an example where the message is not an even number of dwords and the MIC spans Dwords 7 and 8. The contents of the messages listed below should be used for reference and do not correspond to valid MCTP messages.

Figure 197: MIC Example 1 – 32 Bytes of 0's

	3	2	1	0
Dword 0	00h	00h	00h	00h
…	...	...	...	...
Dword 7	00h	00h	00h	00h
Dword 8 (MIC)	8Ah	91h	36h	AAh

Figure 198: MIC Example 2 – 32 Bytes of 1's

	3	2	1	0
Dword 0	FFh	FFh	FFh	FFh
...	...	...	...	...
Dword 7	FFh	FFh	FFh	FFh
Dword 8 (MIC)	62h	A8h	ABh	43h

Figure 199: MIC Example 3 – 30 Incrementing Bytes from 00h to 1Dh

	3	2	1	0
Dword 0	03h	02h	01h	00h
...	...	...	...	...
Dword 7 (MIC)	92h	D7h	1Dh	1Ch
Dword 8 (MIC)	<unused>		1Eh	05h

Figure 200: MIC Example 4 – 32 Decrementing Bytes from 1Fh to 00h

	3	2	1	0
Dword 0	1Ch	1Dh	1Eh	1Fh
...	...	...	...	...
Dword 7	00h	01h	02h	03h
Dword 8 (MIC)	11h	3Fh	DBh	5Ch

Appendix C Example NVMe-MI Messages

This section contains example NVMe-MI Messages between a Management Controller (e.g., a Baseboard Management Controller) and a Management Endpoint. The Request Messages are sent from the Management Controller to the Management Endpoint and the corresponding Response Messages are sent back from the Management Endpoint to the Management Controller.

The examples assume the following:

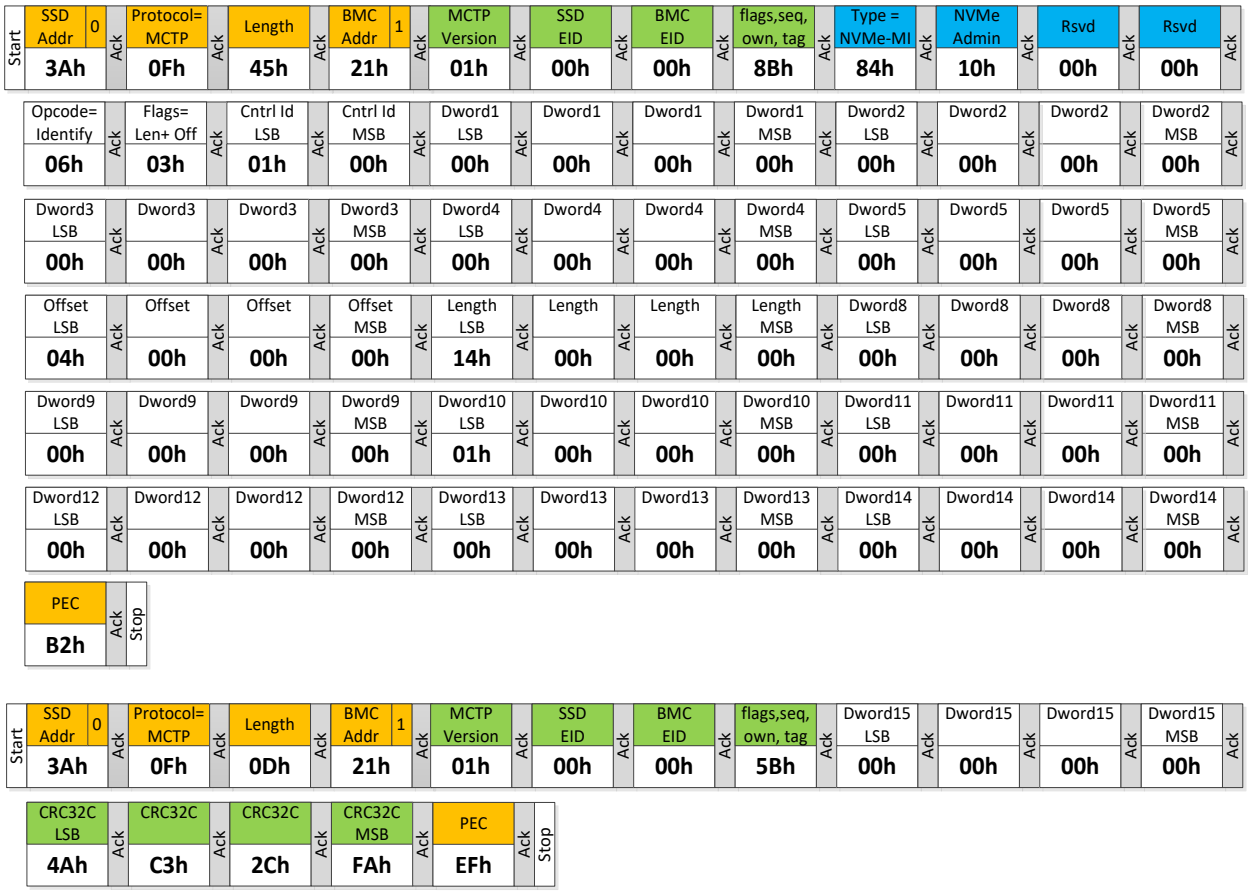
- Management Endpoint 2-Wire address is 3Ah;
- Management Controller 2-Wire address is 20h;
- Management Endpoint MCTP Endpoint ID is 0, examples only use 2-Wire address;
- Management Controller MCTP Endpoint ID is 0, examples only use 2-Wire address;
- MCTP Transmission Unit Size is 64 bytes;
- NVMe Storage Device Composite Temperature (CTEMP) is 30 °C;
- NVMe Storage Device Controller ID is 1; and
- NVMe Storage Device Serial Number is AZ123456.

The first 4 bytes and the last byte of each packet (shown in orange in the examples below) are defined by the MCTP SMBus/I2C Transport Binding Specification. Bytes 4 to 7 of each packet and the Message Integrity Check (**green**) are defined by the MCTP Base Specification. The CRC-32C algorithm and the NVMe-MI Message Header (**blue**) are defined in section 3.1.1.1. Management Controller transmission bytes are shown in white blocks and Management Endpoint transmission bytes are shown in grey blocks. The MCTP endpoint sending the messages drives the clock pin so the signal direction changes between commands and responses as described in the MCTP binding specification.

Example 1: In this example, a Management Controller issues an Identify Command to read the Serial Number (bytes 23:04 of the Identify Controller data structure) of an NVMe Storage Device. The NVMe Storage Device's response is shown in the Example 2.

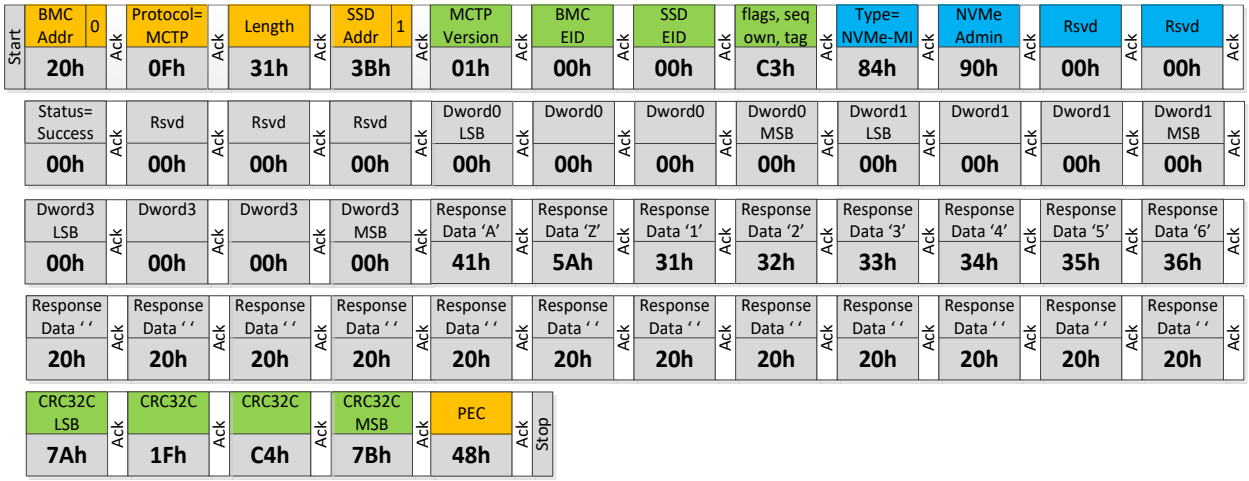
The Request Message is longer than the default 64-byte MCTP Transmission Unit Size and thus spans two MCTP packets. The NVMe-MI Message Type (NMIMT) field specifies that this is an NVMe Admin Command. The NVMe Opcode 06h specifies that this is an Identify Command. This NVMe Opcode and the required values for Dwords 1 to 15 are defined in the NVM Express Base Specification for the Identify Command. The Data Offset of 00000004h skips the first 4 bytes of the Identify Controller data structure response. The Data Length of 00000014h limits the response to 20 bytes.

Notice that the blue header is only present in the first packet of a message. The MCTP packet sequence number is incremented from 0 for the first packet to 1 for the second packet. The SMBus PEC is calculated per packet and includes every byte sent. The Message Integrity Check is calculated across both packet payloads but skips all orange and green bytes. The value for SMBus Length field (Byte 2) is the number of bytes following it in the packet, not including the SMBus PEC field per the SMBus Specification.

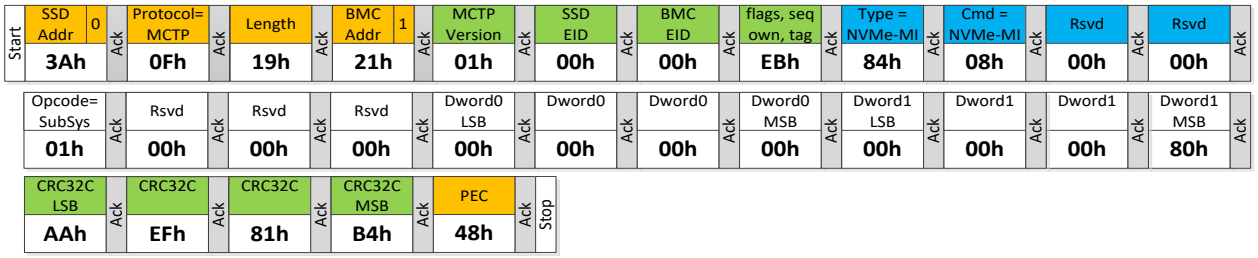


Example 2: This example shows an NVMe Storage Device's Response Message to the Identify Command from Example 1. This message is small enough to fit in a single packet so both MCTP SOM and EOM flags are set. The NVM Express Base Specification defines the format (Dwords 0, 1, and 3) of the Identify Controller data structure bytes that are returned.

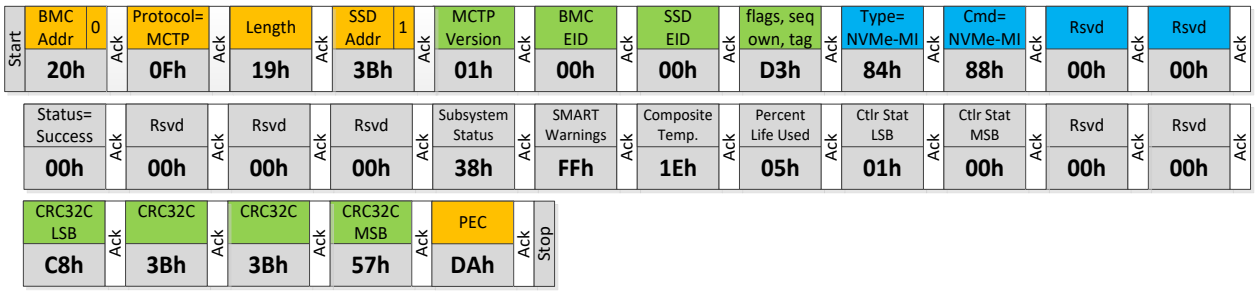
Note that the 2-Wire addresses and MCTP Endpoint IDs in the Response Message are swapped from their order in the Request Message. Also note that the incrementing MCTP packet sequence number for the Management Endpoint is independent from the Management Controller's MCTP packet sequence number.



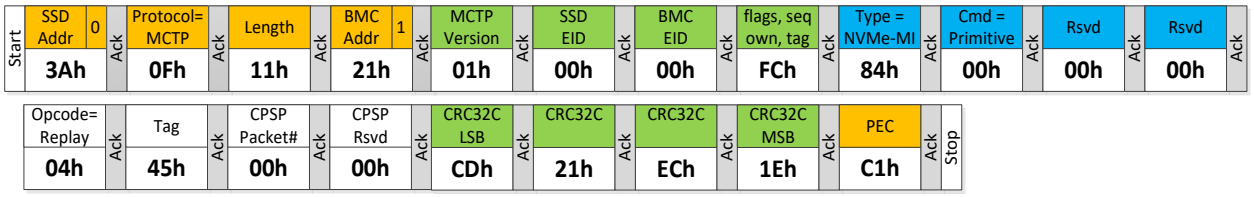
Example 3: In this example, a Management Controller issues an NVM Subsystem Health Status Poll command and clears the Composite Controller Status. Note that the MCTP packet sequence number is incremented from the last packet the Management Controller sent in Example 1. The NVMe-MI Message Type value of 08h with Opcode 01h makes this an NVM Subsystem Health Status Poll command. Bit 31 of Dword1 set to '1' clears the Composite Controller Status Flags field after preparing the response. Only the first non SR-IOV PCI function with any of the trigger able changes is requested.



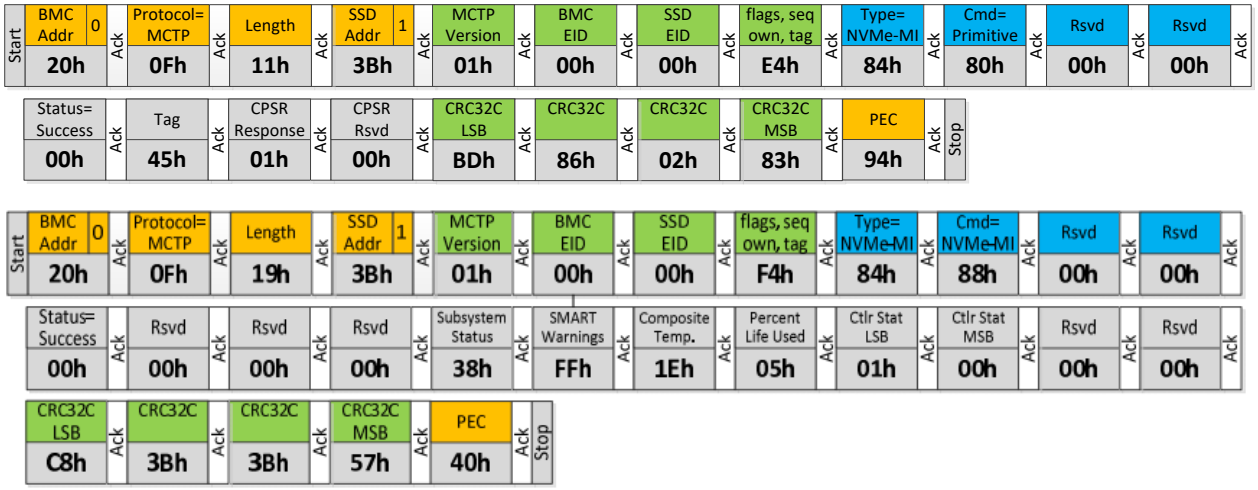
Example 4: This example shows an NVMe Storage Device's response to the NVM Subsystem Health Status Poll command from Example 3. Note that the MCTP packet sequence number is incremented from the last packet the NVMe Storage Device sent in Example 2. Controller ID 0 had a reportable trigger due to its composite temperature change.



Example 5: This example shows a Management Controller issuing a Replay Control Primitive. The Management Controller may choose to replay an entire Response Message if, for example, the Message Integrity Check failed on the initial Response Message. Or the Management Controller may choose to replay a partial message starting at a specified MCTP Transmission Unit Size boundary if, for example, the SMBus PEC failed on an individual packet. The Control Primitive Tag is arbitrarily set to 45h and remembered by the Management Controller to match response packets to the correct Control Primitives. The MCTP Tag is also modified for this example to show the effect on the replayed packet.



Example 6: This example shows an NVMe Storage Device sending an acknowledgement Response Message to the Replay Control Primitive and then sending a second Response Message that replays the previous Response Message from specified offset of 0h. Note that the previous command is not reissued because that may or may not return different data after having the Composite Controller Status Flags field cleared.



If each Request Message fits into a single packet, then the Management Controller's MCTP stack may be simplified to transmit fixed strings that are pre-coded with the correct headers and MIC. Receiving Response Messages also does not require a full MCTP stack as the Management Controller may ignore the Pkt Seq # field for single packet Response Messages but should compute CRCs to test the Response Message's PEC and MIC fields. Underlined bytes in the example Response Messages change with Pkt Seq #and thus may not match the example. Bold italic shadows mark bytes in the example Response Messages that change if an Error Response is returned.

The rest of this appendix follows the same color scheme for headers and footers as in the prior examples but byte labeling was eliminated to for easier copying of the byte strings into test tools. Each Request Message is a single packet and each Response Messages is a single packet. If implementations do not use a Management Controller's physical address of 20h (21h in the command string), then the physical address and the PEC field should be updated. Some implementations may also require the Port Identifier to be a value other than 0. Online tools like https://www.crccalc.com may be used to calculate the PEC and MIC fields.

Example 7: Set SMBus packet size to 250 bytes to make most Response Messages single packet:

3Ah 0Fh 19h 21h 01h 00h 00h 8Bh 84h 08h 00h 00h 03h 00h 00h 00h 03h 00h 00h 00h FAh 00h 00h 00h D2h 88h B4h 01h 0Ch

Successful Response:

20h 0Fh 11h 3Bh 01h 00h 00h D3h 84h 88h 00h 00h ***00h 00h 00h 00h 24h 55h 77h 22h*** 21h

Example 8: VPD Read command to fetch first 232 bytes of VPD on MCTP over SMBus (same data as I2C reads from offset 0):

3Ah 0Fh 19h 21h 01h 00h 00h 8Bh 84h 10h 00h 00h 05h 00h 00h 00h 00h 00h 00h 00h F0h 00h 00h 00h 51h 22h 32h 62h 9Ah

Successful Response (the 232 bytes of VPD are shown with '…' and CRCs are marked with 'xxh' since they depend on VPD content):

20h 0Fh **F9h** 3Bh 01h 00h 00h D3h 84h 88h 00h 00h ***00h 00h 00h 00h … xxh xxh xxh xxh*** xxh

Example 9: Read NVMe-MI Data Structure command on MCTP over SMBus to retrieve data similar to what is returned by NVMe Basic Management Command:

3Ah 0Fh 19h 21h 01h 00h 00h 8Bh 84h 10h 00h 00h 02h 00h 00h 00h 00h 00h 00h 00h 00h 00h 00h AFh 3Dh 20h 33h 41h

Successful Response (the 8 bytes of NSHDS are shown with '…' and CRCs are marked with 'xxh' since they depend on NSDHDS content, the first 4 bytes of NSHDS match Basic data):

20h 0Fh **19h** 3Bh 01h 00h 00h D3h 84h 88h 00h 00h ***00h 00h 00h 00h … xxh xxh xxh xxh*** xxh

Refer to the PCI Express Base Specification for the algorithm to switch the 2-Wire port from SMBus mode into I3C mode. The next 4 examples show how the prior 3 examples are done in I3C mode. In the I3C interface, the clock signal is always driven by the Management Controller so there is no mechanism to negotiate the bus frequency but a Management Endpoint may advertise the max timings supported.

Example 10: Assign address 30h to the I3C element using the ENTDAA CCC (07h). The NVMe Storage vendor's MIPI Manufacturer ID is 1234h and the NVMe Storage Device's unique Device Id is 01020304h. The BCR for this element is 06h and the DCR is CCh for all MCTP elements. Red bytes are MIPI CCCs from the Management Controller with orange responses from the Management Endpoint. A 7Eh read becomes FCh, a 7Eh write becomes FDh, the ENTDAA CCC is 07h, and a repeated start is indicated with 'Sr'. Interrupts are enabled by default so the NVMe Storage Device with an address is now ready to send and receive MCTP packets over I3C. Changes for error conditions are similar to MCTP over SMBus but not indicated for I3C.

FCh 07h Sr FDh 24h 68h 01h 02h 03h 04h 06h CCh 60h

Example 11: Configure the I3C device at address 30h to receive and transmit packets of 250 (FAh) bytes. Follow up reads are used to confirm that both settings were accepted. Direct CCCs for SETMWL=89h, SETMRL=8Ah, GETMWL=8B, and GETMRL=8Ch:

FCh 89h Sr 60h 00h FAh

FCh 8Ah Sr 60h 00h FAh

FCh 8Bh Sr 61h 00h FAh

FCh 8Ch Sr 61h 00h FAh

Example 12: VPD Read command to fetch first 8 bytes of VPD on MCTP over I3C (smaller read used to show CRC values):

60h 01h 00h 00h 8Bh 84h 10h 00h 00h 05h 00h 00h 00h 00h 00h 00h 00h 08h 00h 00h 00h D9h 72h 31h 53h D3h

IBI notification that an MCTP packet is ready for Management Controller to read:

61h AEh

Successful Response privately read by Management Controller (VPD is last 8 bytes in black font):

61h 01h 00h 00h D3h 84h 88h 00h 00h 00h 00h 00h 00h 01h 00h 00h 00h 01h 0Bh 00h F3h E9h 1Fh 3Dh 8Bh EFh

Example 13: Read NVMe-MI Data Structure command on MCTP over I3C to retrieve data similar to what is returned by NVMe Basic Management Command:

60h 01h 00h 00h 8Bh 84h 10h 00h 00h 02h 00h 00h 00h 00h 00h 00h 00h 00h 00h 00h 00h AFh 3Dh 20h 33h A7h

IBI notification that an MCTP packet is ready for Management Controller to read:

61h AEh

Successful Response privately read by Management Controller (data is last 8 bytes in black font):

61h 01h 00h 00h D3h 84h 88h 00h 00h 00h 00h 00h 00h 34h FFh 25h 0Ah 00h 01h 00h 00h 1Ah 75h 4Ah 30h F2h

Appendix D AEM Example Timing Diagrams

Figure 201 shows an example where the AEM Retry Delay field is not cleared to 0h and with:

a) multiple unique AEs occurring during the AEM Delay Interval and during the AEM Transmission Interval;
b) a failed AEM transmission; and
c) a successful retry of the failed AEM transmission.

The sequence of events in Figure 201 is as follows:

1. At the end of Management Endpoint Reset, the Management Endpoint is in the AE Disarmed State.
2. AEs are enabled via the Configuration Set command for the Asynchronous Event configuration.
    a. The Management Controller specifies the duration of the AEM Delay Interval in the AEM Delay field.
    b. The Management Controller specifies the amount of time to delay before retrying an AEM transmission due to transmission failure in the AEM Retry Delay field.
    c. The Management Endpoint transitions to the AE Armed State.
    d. The AEM Delay Interval starts.
3. The first AE occurs but an AEM is not yet transmitted since it occurred during the AEM Delay Interval.
4. The second AE occurs but an AEM is not yet transmitted since it occurred during the AEM Delay Interval.
5. After the AEM Delay Interval ends, the AEM Transmission Interval starts immediately since there are AEs that have occurred during the AEM Delay Interval.
    a. When the AEM Transmission Interval started, the Management Endpoint transitioned to the AE Disarmed State.
6. The Management Endpoint transmits a single AEM containing an AE Occurrence data structure for each unique AE that occurred during the prior AE Armed State (i.e., AE #1 and AE #2 in this example) with the AEM Generation Number field cleared to 0h and the AEM Retry Count field cleared to 0h.
    a. The Management Endpoint should minimize the amount of time between entering the AEM Transmission Interval and transmitting the AEM.
    b. After transmitting the AEM, the Management Endpoint waits the amount of time specified by the AEM Retry Delay field to receive an AEM Ack.
7. The third AE occurs but an AEM is not yet transmitted since it occurred during the AEM Transmission Interval.
8. After waiting the amount of time specified AEM Retry Delay field for the AEM Ack, the Management Endpoint times out waiting for an AEM Ack.
9. The Management Endpoint retries the AEM transmission from step 6 with the AEM Generation Number field cleared to 0h and the AEM Retry Count field set to 1h.
    a. Since this is a retry of the AEM transmission in step 6, an AE Occurrence data structure for AE #3 is not included.
    b. The Management Endpoint should minimize the amount of time between the end of the AEM Retry Delay interval and retrying the AEM transmission.
    c. After retrying the AEM transmission, the Management Endpoint waits the amount of time specified by the AEM Retry Delay field to receive an AEM Ack.

10. The Management Endpoint receives an AEM Ack to acknowledge receipt of the AEM containing AE #1 and AE #2 prior to the AEM Retry Delay interval ending.
    a. The Response Message for the AEM Ack contains any AEs that have occurred since the start of the AEM Transmission Interval (i.e., AE #3 in this example).
    b. The AEM Transmission Interval Ends.
    c. The Management Endpoint transition back to the AE Armed State.
    d. The next AEM Delay Interval starts.
    e. The Management Endpoints waits for the next AE to occur.

Figure 201: AEM Example 1

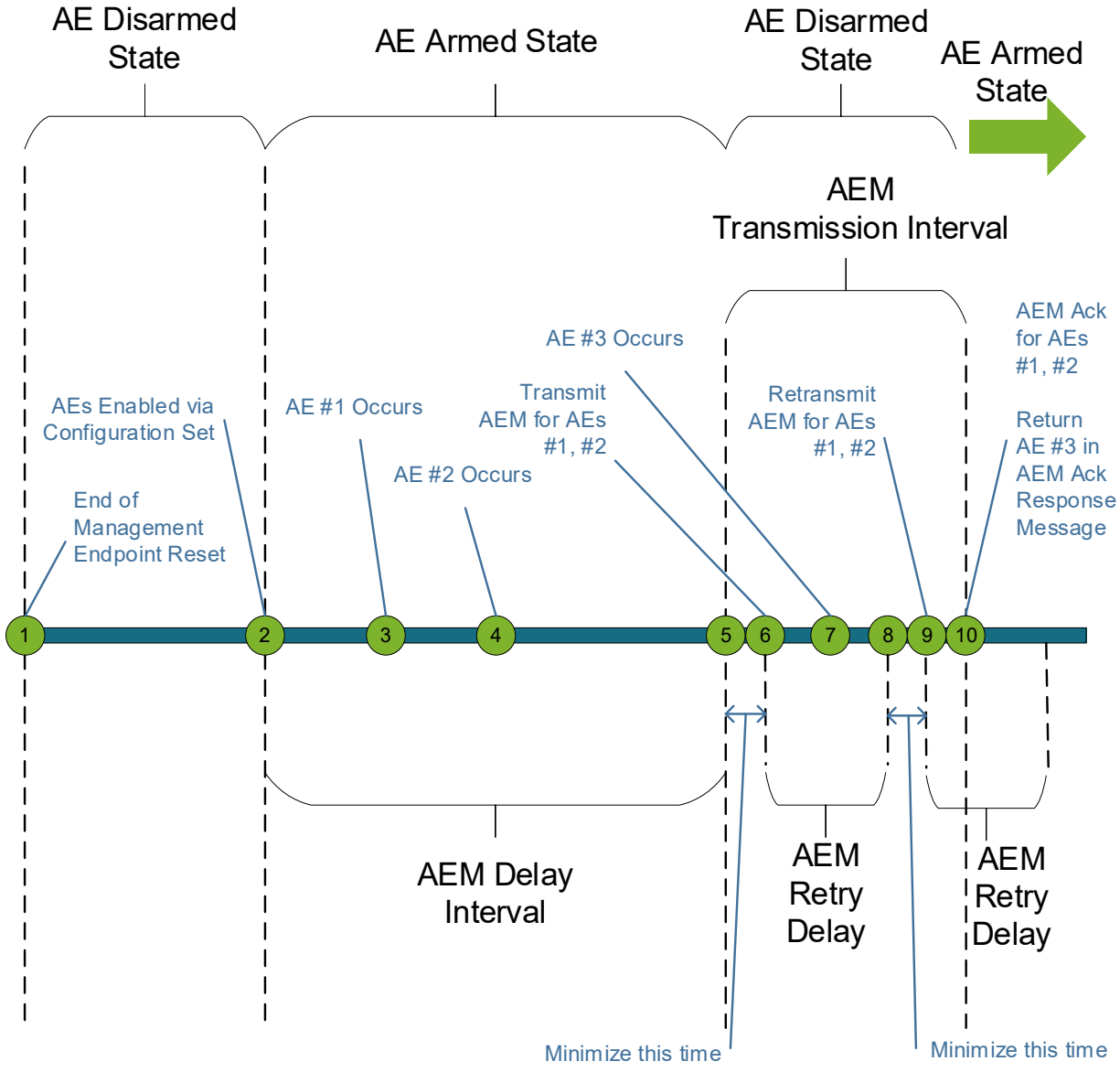


Figure 202 shows an example where an AE occurs after the AEM Delay Interval, the Management Endpoint exhausts its AEM transmission retries without getting an AEM Ack assuming the AEM Retry Delay field is not cleared to 0h, then another AE occurs after AEM Transmission Interval.

The sequence of events in Figure 202 is as follows:

1. AEs are enabled via the Configuration Set command for the Asynchronous Event configuration.
    a. The Management Controller specifies the duration of the AEM Delay Interval in the AEM Delay field.
    b. The Management Controller specifies the amount of time to delay before retrying an AEM transmission due to transmission failure in the AEM Retry Delay field.
    c. The Management Endpoint transitions to the AE Armed State.
    d. The AEM Delay Interval starts.
2. The first AE occurs after the AEM Delay Interval has ended.
    a. Since the AEM Delay Interval has already ended, the AEM Transmission Interval starts as soon as the first AE occurs.
    b. When the AEM Transmission Interval started, the Management Endpoint transitioned to the AE Disarmed State.
3. The Management Endpoint transmits a single AEM containing an AE Occurrence data structure for each AE that occurred during the prior AE Armed State (i.e., AE #1 in this example) with the AEM Generation Number field cleared to 0h and the AEM Retry Count field cleared to 0h.
    a. The Management Endpoint should minimize the amount of time between entering the AEM Transmission Interval and transmitting the AEM.
    b. After transmitting the AEM, the Management Endpoint waits the amount of time specified by the AEM Retry Delay field to receive an AEM Ack.
4. After waiting the amount of time specified AEM Retry Delay field for the AEM Ack, the Management Endpoint times out waiting for an AEM Ack.
5. The Management Endpoint retries the AEM transmission from step 3 with the AEM Generation Number field cleared to 0h and the AEM Retry Count field incremented by one.
    a. The Management Endpoint should minimize the amount of time between the end of the AEM Retry Delay interval and retrying the AEM transmission.
    b. After retrying the AEM transmission, the Management Endpoint waits the amount of time specified by the AEM Retry Delay field to receive an AEM Ack.
6. The Management Controller remains unresponsive (e.g., due to being reset) and does not transmit an AEM Ack and so steps 4 and 5 are repeated until all AEM transmission attempts have been exhausted (8 total transmission attempts).
7. After waiting the amount of time specified by the AEM Retry Delay field after the final AEM transmission attempt, no AEM Ack has been received and the AEM Transmission Interval Ends.
    a. The Management Endpoint sets the AEM Transmission Failure bit in the NVM Subsystem Health data structure.
8. The second AE occurs but an AEM containing an AE Occurrence data structure for the second AE is not permitted to be transmitted since the Management Endpoint is in the AE Disarmed State.
    a. As long as the Management Endpoint is in the AE Disarmed State, no AEMs are permitted to be transmitted.
    b. If the Management Controller times out waiting for an AEM, the Management Controller may issue an NVM Subsystem Health Poll command and check the AEM Transmission Failure bit to determine if an AEM transmission failure has occurred.
9. Once the Management Controller becomes responsive, the Management Controller resyncs with the Management Endpoint to get the current state of the AEs.
    a. If the Management Controller never received any of the AEMs containing an AE Occurrence data structure for the first AE, then note that an AEM Ack at this point is not

able to be used to resync since that AEM Ack only returns an AE Occurrence data structure for the second AE and not the first AE.

b. If the Management Controller does not know which AEs are enabled, then the Management Controller may issue a Configuration Get command for the Asynchronous Event configuration to get the enable/disable state of all supported AEs.
c. The Management Controller may issue a Configuration Set command for the Asynchronous Event configuration to disable or enable AEs.
d. In response to the Configuration Set command in step 9c, the Management Endpoint returns an AE Occurrence data structure for each AE that was enabled by the Configuration Set command or that was already enabled which resyncs the Management Endpoint and Management Controller.
e. The Management Endpoint transitions to the AE Armed State.
f. The next AEM Delay Interval starts.
g. The Management Endpoints waits for the next AE to occur.

Figure 202: AEM Example 2