LEGAL NOTICE:

© Copyright 2007 - 2016 NVM Express, Inc. ALL RIGHTS RESERVED.
This erratum to the NVM Express Management Interface Revision 1.0 specification is proprietary to the NVM Express, Inc. (also referred to as "Company") and/or its successors and assigns.

NOTICE TO USERS WHO ARE NVM EXPRESS, INC. MEMBERS: Members of NVM Express, Inc. have the right to use and implement this erratum to the NVM Express Management Interface Revision 1.0 specification subject, however, to the Member's continued compliance with the Company's Intellectual Property Policy and Bylaws and the Member's Participation Agreement.

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

LEGAL DISCLAIMER:

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

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

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

NVM Express™ Technical Errata

Errata ID	**002**
Revision Date	**9/16/2016**
Affected Spec Ver.	**NVM Express™ MI 1.0**
Corrected Spec Ver.	

Errata Author(s)

Name	Company
Austin Bolen	Dell
Myron Loewen	Intel

Errata Overview

The erratum includes clarifications for NVM Express Management Interface, Revision 1.0, including:

- Clarifications on how the replay primitive is intended to work
- New Appendix with example packets

Revision History

Revision Date	Change Description
10/29/2015	Initial draft of Appendix C
5/12/2016	Incorporated all feedback but Replay issue is still open
9/12/2016	Final revision after compromise reached on Replay primitive
9/30/2016	Reversing earlier decision to now make replay response a complete standalone MCTP message.
10/17/2016	Editorial updates. Added note about MCTP extensions that are needed by a Management Controller to handle Replay with non-zero Response Replay Offset.

Description of Specification Changes

These changes are intended to clarify grey areas of the specification of replay that was being implemented in vendor unique ways. The initial compromise is hereby improved to make the replay payload a more standard MCTP message.

Modify a Section 4.4.5 Replay paragraphs as shown below:

4.4.5 Replay

The Replay Control Primitive is used to retransmit the Response Message for the last Command Message processed in a Command Slot. 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 and shall include the Message Header of the original Response Message even if the Response Replay Offset is not zero.

Note that the Management Controller will need 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 needed to Replay with Response Replay Offset equal to zero. 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 will need extensions in 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 can then issue a Replay to get the second part of the Response Message using a non-zero Response Replay Offset. The Management Controller's NVMe-MI layer can then assemble the two partial Response Messages to create the whole Response Message. The MIC can then be validated across the whole Response Message as described in Section 3.2.1.1.

The format of the CPSP field in the Control Primitive Request Message is shown in Figure 32.

Transmit: The Management Endpoint stops transmitting response packets for the Command Slot and then transmits a Response Message with success status with the RR field set to '1'. The Management Endpoint transmits a Response Message containing the packets starting at the packet offset specified in the Response Replay Offset field of the Replay after the Control Primitive success response. The Command Slot remains in the Transmit state until retransmission is complete.

Technical input submitted to the NVM Express™ Workgroup is subject to the terms of the NVM Express™ Participant's agreement. Copyright © 2014-16 NVMe™ Corporation.

Modify Section 3.2.1.1 as shown below:

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

See section 4.4.5 for special requirements on how to construct the NVMe-MI Response Message when the Management Controller issues a Replay of a Response Message with a non-zero Response Replay Offset.

Insert Appendix C as shown below:

Appendix C – Example NVMe-MI Messages over SMBus/I2C

This section contains example NVMe-MI Messages over SMBus/I2c 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 SMBus/I2C address is 3Ah
- Management Controller SMBus/I2C address is 20h
- Management Endpoint MCTP Endpoint ID is 0, examples only use SMBus/I2C address
- Management Controller MCTP Endpoint ID is 0, examples only use SMBus/I2C address
- MCTP Transmission Unit Size is 64 bytes
- NVMe storage device Composite Temperature (CTEMP) is 30 °C
- NVMe storage device Controller ID is 1
- NVMe storage device Serial Number is AZ123456

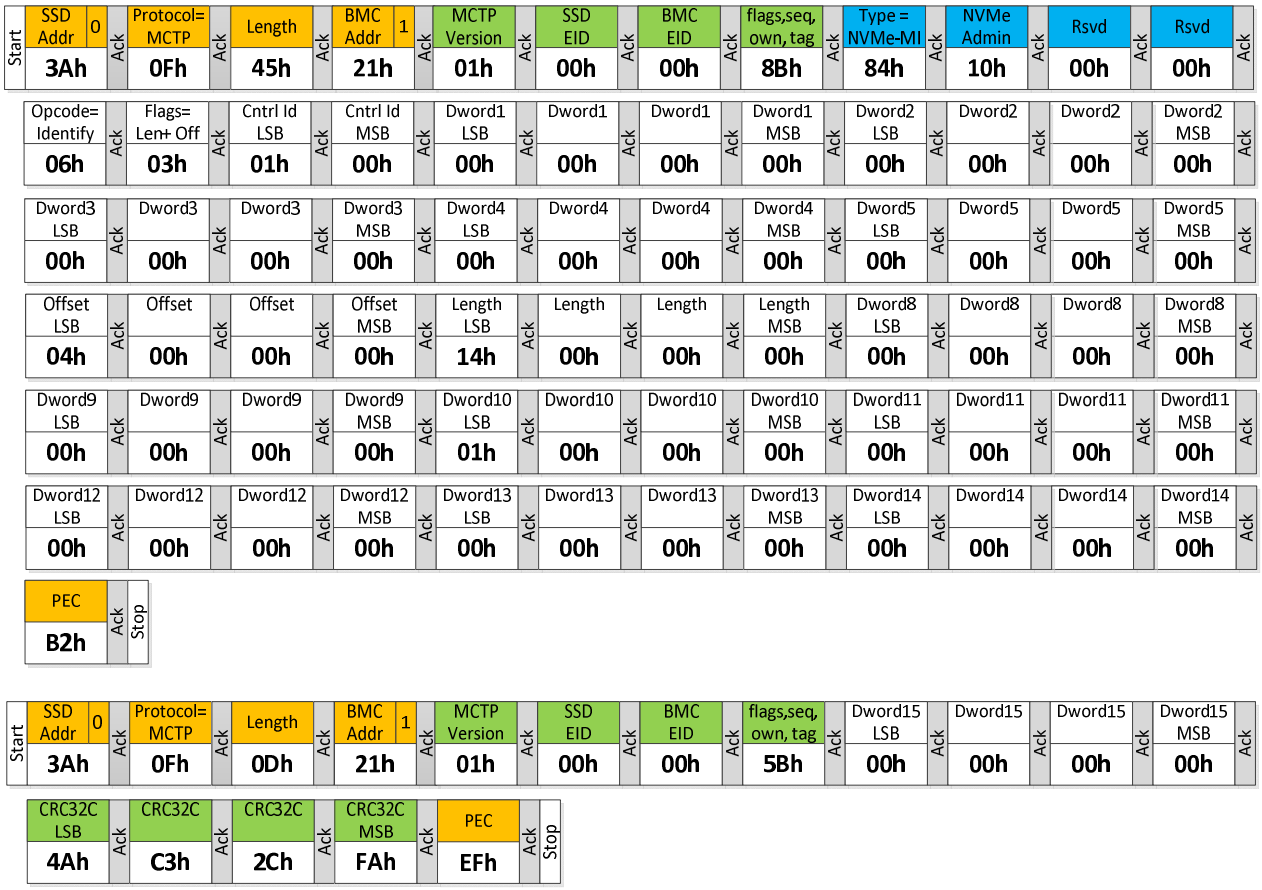
The first 4 bytes and the last byte of each packet (shown in orange in the examples below) are defined by the MCTP SMBus/I2C Transport Binding Specification. Bytes 4 to 7 of each packet and the Message Integrity Check (**green**) are defined by the MCTP Base Specification. The CRC-32C algorithm and the NVMe-MI Message Header (**blue**) are defined in Section 3.2.1.1. Management Controller transmission bytes are shown in white blocks and Management Endpoint transmission bytes are shown in grey blocks. All messages are sent in SMBus master mode and received in slave mode so both sides must reconfigure SMBus between commands and responses.

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 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.

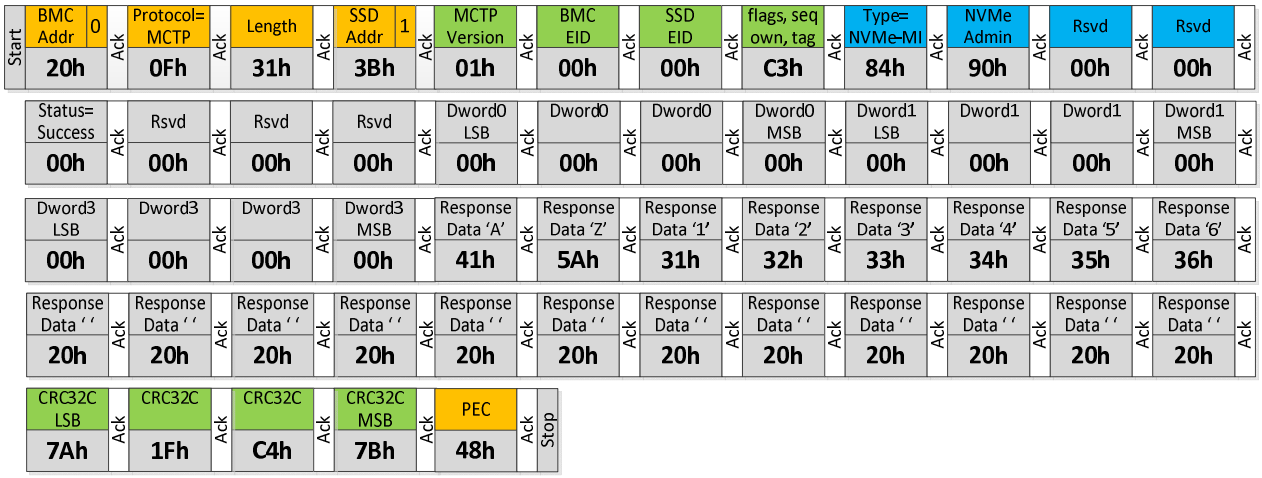
Notice that the blue header is only present in the first packet of a message. The MCTP packet sequence number is incremented from 0 for the first packet to 1 for the second packet. The SMBus PEC is calculated per packet and includes every byte sent. The Message Integrity Check is calculated across both 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

Technical input submitted to the NVM Express™ Workgroup is subject to the terms of the NVM Express™ Participant's agreement. Copyright © 2014-16 NVMe™ Corporation.

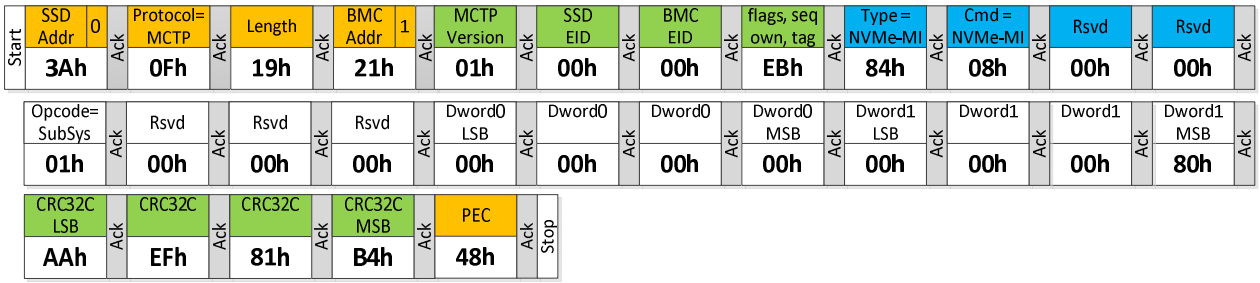


Example 2: This example shows an NVMe storage device's Response Message to the Identify Command from Example 1. This message is small enough to fit in a single packet so both MCTP SOM and EOM flags are set. The NVM Express Specification defines the format (Dwords 0, 1, and 3) of the Identify Controller Data Structure bytes that are returned.

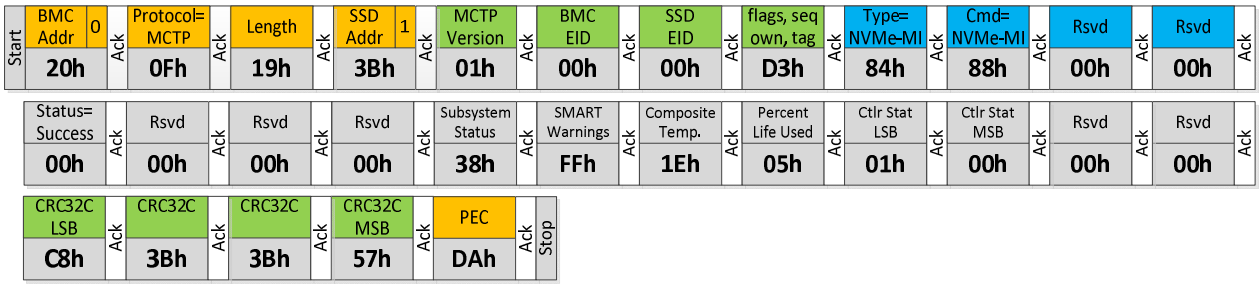
Note that the SMBus/I2C addresses and MCTP Endpoint IDs in the Response Message are swapped from their order in the Request Message. Also note that the incrementing MCTP packet sequence number for the Management Endpoint is independent from the Management Controller's MCTP packet sequence number.



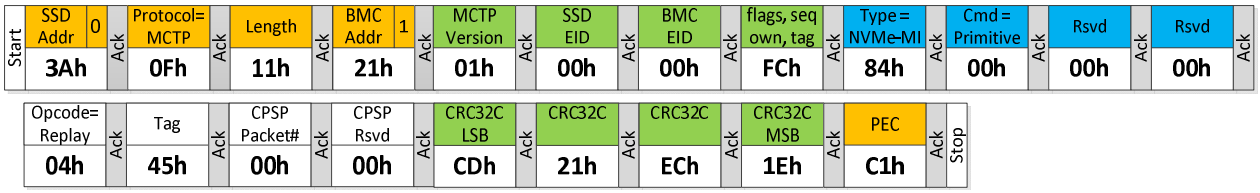
Example 3: In this example, a Management Controller issues an NVM Subsystem Health Status Poll command and clears the Composite Controller Status. Note that the MCTP packet sequence number is incremented from the last packet the Management Controller sent in Example 1. The NVMe-MI Message Type value 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 after preparing the response. Only the first non SR-OV 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.



Technical input submitted to the NVM Express™ Workgroup is subject to the terms of the NVM Express™ Participant's agreement. Copyright © 2014-16 NVMe™ Corporation.

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 zero. Note that the previous command is not reissued because that could return different data after having the Composite Controller Status cleared. .