UNH–IOL
NVMe Testing Service

Test Plan for NVMe-MI Conformance
Version 19.0
Target Specification: NVMe-MI 1.2
Technical Document

Last Updated: January 04, 2023

UNH–IOL NVMe Testing Service
21 Madbury Rd Suite 100
Durham, NH 03824

Tel: +1 603–862–0090
Fax: +1 603–862–4181
Email:
nvmelab@iol.unh.edu

Table of Contents

Table of Contents 2

Modification Record 6

Acknowledgments 12

Introduction 13

References 14

Group 1: MCTP Base Tests 15
- **Test 1.1 – MCTP Endpoint ID – (FYI)** 15
- **Test 1.1.1 – MCTP Endpoint ID Response – Reserved Bits – (FYI)** 16
- **Test 1.2 – MCTP Packet Sequence Number – (FYI)** 17
- **Test 1.3 – MCTP Tag Owner and Message Tag Bits – (FYI)** 18
- **Test 1.4.1 – MCTP Bad Packet 1 – (FYI)** 19
- **Test 1.4.2 – MCTP Bad Packet 2 – (FYI)** 20
- **Test 1.4.3 – MCTP Bad Packet 3 – (FYI)** 21
- **Test 1.4.4 – MCTP Bad Packet 4 – (FYI)** 22
- **Test 1.4.5 – MCTP Bad Packet 5 – (FYI)** 23

Group 2: MCTP Control Message Tests 24
- **Test 2.1 – MCTP Control Instance ID – (FYI)** 24

Group 3: MCTP Commands 25
- **Test 3.1 – MCTP Command Set Endpoint ID – (FYI)** 25
- **Test 3.2 – MCTP Command Get MCTP Version – (FYI)** 26
- **Test 3.3 – MCTP Command Get Message Type – (FYI)** 27
- **Test 3.4 – MCTP Command Prepare for Endpoint Discovery – (FYI)** 28
- **Test 3.5 – MCTP Command Endpoint Discovery – (FYI)** 29
- **Test 3.6 – MCTP Command Get Endpoint ID – (FYI)** 30

Group 4: NVMe Error Handling 31
- **Test 4.1 – NVMe-MI Invalid Opcode – (Mandatory)** 31
- **Test 4.2 – NVMe-MI Reserved Identifier – (Mandatory)** 32
- **Test 4.3 – NVMe-MI Health Status Change – (Mandatory)** 33
- **Test 4.4 – NVMe-MI Reserved Configuration Identifier – (Mandatory)** 34
- **Test 4.5 – NVMe-MI MAXRENT Error – (Mandatory)** 35
- **Test 4.6 – NVMe-MI Reserved Data Structure Type – (Mandatory)** 36
- **Test 4.7 – NVMe-MI Invalid VPD Read Size -(FYI)** 37
- **Test 4.8 – NVMe-MI Invalid VPD Write Status – (Mandatory)** 38
- **Test 4.9 – NVMe-MI Invalid Parameter Status – (Mandatory)** 39
- **Test 4.10 – NVMe-MI Invalid Command Size – (Mandatory)** 40

Group 5: NVMe Management Interface Tests 41
- **Test 5.1 – NVMe-MI Message Type – (Mandatory)** 41
- **Test 5.2 – NVMe-MI Message IC – (Mandatory)** 42
- **Test 5.3 – NVMe-MI CRC Check – (Mandatory)** 43
- **Test 5.4 – NVMe-MI Command Slot – (Mandatory)** 44
- **Test 5.5 – NVMe-MI MCTP packet padding – (Mandatory)** 45
- **Test 5.6 – NVMe-MI Message Integrity Check – (Mandatory)** 46

Group 6: NVMe-MI Message Processing Tests **47**
Test 6.1 – NVMe-MI Reserved Fields – (Mandatory) **47**
Test 6.2 – NVMe-MI Error Response Code – (Mandatory) **48**
Test 6.3 – Command Initiated Auto Pause – (FYI) **49**

Group 7: Control Primitives Tests **50**
Test 7.1 – NVMe-MI Response Tag – (Mandatory) **50**
Test 7.2 – NVMe-MI Response Message – (Mandatory) **51**
Test 7.3 – NVMe-MI Get State Primitive Response – (FYI) **52**
Test 7.4 – NVMe-MI Response Message Replay – (Mandatory) **52**
Test 7.5 – NVMe-MI Response Replay Offset (RRO) – (Mandatory) **54**
Test 7.6 – NVMe-MI RRO > length of Response Message – (FYI) **54**
Test 7.7 – NVMe-MI Get State, MEB=1 – (FYI) **55**

Group 8: Management Interface Commands **56**
Test 8.1 – NVMe-MI Response Header – (Mandatory) **57**
Test 8.2 – NVMe-MI Configuration Set – (Mandatory) **58**
Test 8.3 – NVMe-MI Config Get Response – (Mandatory) **59**
Test 8.4 – NVMe-MI Health Status Poll – (Mandatory) **60**
Test 8.5 – NVMe-MI Controller Health Status Poll – (M) **61**
Case 1: NVMe-MI 1.0 ECN 003 or NVMe-MI 1.0a or higher Not Implemented (M) 61
Case 2: NVMe-MI 1.0 ECN 003 or NVMe-MI 1.0a or higher Implemented (M) 61
Case 3: Controller Health Status Poll Filtering by Controller Selection (FYI) 62
Case 4: Controller Health Status Poll Filtering by Error Selection Fields (FYI) 62
Test 8.6 – NVMe-MI Read Data Structure – (Mandatory) **64**
Test 8.7 – NVMe-MI Data Length – (Mandatory) **65**
Case 1: Verify NVMSSI Data Length (M) 65
Case 2: Verify PortInfo Data Length (M) 65
Case 3: Verify CtrlrList Data Length (M) 65
Case 4: Verify CtrlrInfo Data Length (M) 65
Case 5: Verify OptCmds Data Length (M) 66
Test 8.8 – Management Endpoint Buffer Read – (FYI) **67**
Case 1: Management Endpoint Buffer Read DLEN>0 (FYI) 67
Case 2: Management Endpoint Buffer Read DLEN=0 (FYI) 67
Case 3: DOFST > Management Endpoint Buffer Size (FYI) 67
Case 4: (DOFST + DLEN) > Management Endpoint Buffer (FYI) 68
Case 5: Management Endpoint Buffer Read after Sanitize Operation(FYI) 68
Test 8.9 – Management Endpoint Buffer Write – (FYI) **69**
Case 1: Management Endpoint Buffer Write DLEN>0 (FYI) 69
Case 2: Management Endpoint Buffer Write DLEN=0 (FYI) 69
Case 3: DOFST > Management Endpoint Buffer Size (FYI) 69
Case 4: (DOFST + DLEN) > Management Endpoint Buffer (FYI) 70
Test 8.10 – SES Read – (FYI) **71**
Case 1: SES Receive (FYI) 71
Case 2: SES Receive with Reserved PCODE (FYI) 71
Case 3: SES Receive with SES Control type PCODE (FYI) 71
Case 4: SES Receive with Truncated Diagnostic Page (FYI) 72
Test 8.11 – SES Send – (FYI) **73**
Case 1: SES Send (FYI) 73
Case 2: SES Send with DLEN=0 (FYI) 73
Test 8.12 – VPD Read – (FYI) **74**

Case 1: VPD Read (FYI) 74
Case 2: VPD Read with DLEN=0 (FYI) 74
Case 3: VPD Read with DLEN + DOFST > VPD Size (FYI) 74
Test 8.13 – VPD Write – (FYI) 75
Case 1: VPD Write with DLEN=0 (FYI) 75
Case 2: VPD Write Data Check (FYI) 75
Case 3: VPD Write Cycle Information Check (FYI) 75

Group 9: NVMe Admin Command Set Tests 75
Test 9.1 – NVMe Identify Command – (FYI) 76
Case 1: Identify Command (FYI) 76
Test 9.2 – NVMe Get Log Command – (FYI) 77
Case 1: Get Log Page Command (FYI) 77
Case 2: Get Log Page Command, Retain Asynchronous Event bit cleared (FYI) 77
Case 3: Get Log Page Command, Boot Partition (FYI) 77
Test 9.3 – NVMe Get / Set Feature Command – (FYI) 79
Case 1: Get Feature Command (FYI) 79
Case 2: Get Feature Command to Host Metadata FIDs GDHM = 1 (FYI) 79
Case 3: Get Feature Command to Host Metadata FIDs GDHM = 0 (FYI) 80
Case 4: Set Feature Command to EA = 00b Non Existent Element Type (FYI) 80
Case 5: Set Feature Command to EA = 00b Element Type Exists (FYI) 80
Case 6: Set Feature Command to FID Enhanced Controller Metadata 7Dh, EA = 00b (FYI) 80
Case 7: Set Feature Command to FID Enhanced Controller Metadata 7Dh, EA = 01b (FYI) 81
Case 8: Set Feature Command to FID Enhanced Controller Metadata 7Dh, EA = 10b, Descriptor does not exist (FYI) 81
Case 9: Set Feature Command to FID Enhanced Controller Metadata 7Dh, EA = 10b, Descriptor Exists (FYI) 81
Case 10: Set Feature Command to FID 7Eh and 7FH, EA = 10b (FYI) 82
Case 11: Host Metadata Feature too Large (FYI) 82
Case 12: Enhanced Controller Metadata Feature value after Reset (FYI) 82
Case 13: Get Feature Command to FID 7Dh, SEL= 011b (FYI) 82
Case 14: Get Feature Command to FID 7Fh, SEL= 011b (FYI) 82
Case 15: Set Feature Command DLEN > 4096 (FYI) 83
Test 9.4 – Admin Commands Prohibited Out of Band – (FYI) 84
Case 1: Admin Commands Prohibited out of Band – Storage Device (FYI) 84
Case 2: Admin Commands Prohibited out of Band – Enclosure (FYI) 84
Test 9.5 – Sanitize Command – (FYI) 86
Case 1: NVMe-MI Commands During Sanitize (FYI) 86
Test 9.6 – Format NVM, More Processing Time Required – (FYI) 86
Case 1: NVMe-MI Format NVM, More Processing Required (FYI) 87

Group 10: Management Enhancement Tests 89
Test 10.1 – NVMe-MI Identify Structure ME Capabilities – (In Progress) 89
Test 10.2 – NVMe-MI Identify Capabilities – (In Progress) 90
Test 10.3 – NVMe-MI Namespace Metadata – (FYI) 91
Case 1: Perform Set/Get Feature for Namespace Metadata (FYI) 91
Case 2: Host Namespace Metadata Data Structure Too Large (FYI) 91
Test 10.4 – NVMe-MI Controller Metadata – (FYI) 93
Case 1: Perform Set/Get Feature for Controller Metadata (FYI) 93
Case 2: Host Controller Metadata Data Structure Too Large (FYI) 93

Group 11: Vital Product Data Tests 95

Test 11.1 – VPD Read Default Values – (FYI) **96**
Test 11.2 – Topology Multirecord Area – (FYI) **97**
Test 11.3 – NVMe Multirecord Area – (FYI) **98**
Test 11.4 – NVMe PCIe Port Area – (M) **99**
Test 11.5 – FRU Information Device Read via VPD Read – (M) **100**
Test 11.6 – FRU Information Device Read via I2C Read – (M) **101**
Test 11.7 – FRU Information Device Update – (M) **102**
Test 11.8 – FRU Information Device Internal Offset – (M) **103**

Group 12: Management Endpoint Reset Tests **104**
Test 12.1 – NVMe-MI PCIe Endpoint Reset – (M) **105**
Test 12.2 – NVMe-MI SMBus Endpoint Reset – (M) **105**
Test 12.3 – NVMe-MI SMBus/I2C Reset, bits and fields – (FYI) **107**
Test 12.4 – NVMe-MI Controller Level Reset – (FYI) **107**

Appendix A: Default Test Setup **110**

Appendix B: Notes on Test Procedures **111**

Appendix C: NVMe Integrators List Requirements **112**

Appendix D: Test Tools **115**

MODIFICATION RECORD

2017 February 7 (Version 7.0 r01) Initial Release
David Woolf:

2017 February 14 (Version 7.0 r02) Initial Release
David Woolf: Adjusted Group 1, 2, 3 to be FYI.

2017 February 21 (Version 7.0 r03) Initial Release
David Woolf: Adjusted Group 9, 10, 11, 12 to be In Progress.

March 22, 2017 (Version 7.0)
David Woolf: Final version published to UNH-IOL site ahead of May 2017 NVMe Plugfest #7.

April 4, 2017 (Version 7.0a)
David Woolf: Updated tests 4.7, 4.10, 5.3, 7.3 to be FYI for May 2017 NVMe Plugfest #7 per direction of NVMe-MI WG.

August 29, 2017 (Version 8.0)
David Woolf: Merged original Test Specification v0.9 from NVMe-MI Working Group into this document. Made Test 8.5 FYI.

September 12, 2017 (Version 8.0a)
David Woolf: Added Appendix D on Test Tool versions supporting the tests described in this test document.

September 19, 2017 (Version 9.0 draft)
David Woolf:

- Edited procedure and observable results for Test 8.5 to accommodate new requirements published in NVMe-MI 1.0 ECN 003.

February 1, 2018 (Version 9.0 draft)
David Woolf:

- Edited observable results for Test 4.8 to accommodate for varied error responses allowed by chapter 4.2 of the NVMe-MI 1.0 specification.
- Updated tests 4.10 and 5.3 from FYI to Mandatory.
- Modified Test 8.5 to include separate cases for devices which have or have not implemented NVMe-MI 1.0 ECN 003.

August 30, 2018 (Version 10.0 release)
David Woolf:

- Release for 10.0.

November 26, 2018 (Version 11.0 release)
David Woolf:

1. Updated Test 8.5 Case 1 and Test 8.5 Case 2 to accommodate NVMe-MI 1.0 ECN 003 changing the RENT field from being a zeroes based value to not being a zeroes based value. Products implementing NVMe-MI 1.0 may or may not implement NVMe-MI 1.0 ECN 003. Products implementing NVMe-MI 1.0a or higher must implement NVME-MI 1.0 ECN 003.
2. Fixed formatting for sub test cases in Test 8.7.

April 29, 2019 (Version 12.0 release)
David Woolf:

- Updated Test 7.3 to reflect proper use of the CESF bit, per TP2008.
- Updated section on NVMe Integrators List requirements to match the NVMe Integrators List Policy Document.

2020 March 9 (Version 13.0 release)
David Woolf:

- Fixed typo in Test 8.5 test title, which mistakenly showed the test was FYI.
- Updated Test 4.6 and 4.9 to use a better reserved value.
- Modification to tests 1.4.2 and 1.4.4. to allow for the DUT to send a NACK response.

2020 July 21 (Version 14.0 release)
David Woolf:

1. Test Cases 6.3 added to address TP 6010, Command Initiated Auto Pause requirements.
2. Test Case 7.4 updated to address NVME-MI v1.0 ECN 002 requirements around Replay and the SOM field.
3. Test Case 7.5 updated to address NVME-MI v1.0 ECN 002 requirements around Replay and the SOM field.
4. Test Case 8.5.3 added to address NVME-MI v1.0 ECN 003 requirements around filtering Controller Health Status Poll responses according to Controller Selection types.
5. Test Case 8.5.4 added to address NVME-MI v1.0 ECN 003 requirements around filtering Controller Health Status Poll responses according to error selection types.
6. Test Case 8.6 updated address TP 6007 updates.
7. Test Cases 8.8.1, 2, 3, 4, 5 added to address TP 6001 Management Endpoint Buffer Read Command requirements.
8. Test Cases 8.9.1, 2, 3, 4 added to address TP 6001 Management Endpoint Buffer Write Command requirements.
9. Test Cases 8.10.1, 2, 3, 4 added to address TP 6001 SES Receive Command requirements.
10. Test Cases 8.11.1, 2 added to address TP 6001 SES Send Command requirements.

11. Test Cases 8.12.1, 2, 3 added to address VPD Read requirements.
12. Test Cases 8.13.1, 2, 3 added to address VPD Write requirements and TP 6004.
13. Test Case 9.1 added to address Identify Command over NVMe-MI for NVMe Storage Device
14. Test Case 9.2 added to address Get Features Command over NVMe-MI for NVMe Storage Device
15. Test Case 9.3 added to address Get Log Page Command over NVMe-MI for NVMe Storage Device
16. Test Cases 9.4.1, 2 added to address handling of NVMe Admin commands prohibited for NVMe Enclosures.
17. Test Cases 9.5.1 added to address handling of NVMe-MI commands during a Sanitize operation.
18. Test Case 10.3 updated to address NVME-MI v1.0 ECN 003 requirements around proper to incorrect Set Features requests for the Namespace Metadata feature.
19. Test Case 10.4 added to address NVME-MI v1.0 ECN 003 requirements around proper to incorrect Set Features requests for the Controller Metadata feature.
20. Test Case 11.1 updated to address NVME-MI v1.0a ECN 001 requirements around Product Info Area field and the CPIA field.
21. Test Case 11.2 added to address TP 6003a and Topology Multirecord requirements.
22. Test Cases 11.3 and 11.4 added to address TP 6006 PCIe Port Numbering
23. Fixed typo where Observables for 8.4 and 8.6 were swapped.

2021 May 3 (Version 15.0 release)
David Woolf:

- Program Revision Update.

2021 September 23 (Version 16.0 release)
David Woolf:

1. Updated Test 9.1.1 per requirements in NVMe-MI TP 6016b that 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 and the NVMe Enclosure (NVMEE) bit shall be set to ‘1’ for an NVMe Enclosure. This test also checks that the NVMESD bit, the NVMEE bit, or both the NVMESD bit and the NVMEE bit shall be set to ‘1’. This test also checks if the VPD Write Cycle Remain Valid bit is cleared to 0, then the VPD Write Cycles Remains field shall be cleared to 0h.
2. Updated Test 9.3.1 per requirements in NVMe-MI TP 6016b that for the Set and Get Features commands, all Feature Identifiers supported shall be supported if received in-band on an NVMe controller or received out-of-band on a Management Endpoint.
3. Added Test 9.3.2 per requirements in NVMe-MI TP 6016b that Get Features commands for the Host Metadata FIDs (7Dh, 7Eh, 7Fh), has the SEL field set to 011b (i.e., Supported Capabilities), then the Saveable bit in Dword 0 of the

corresponding completion queue entry shall be cleared to '0', and the Changeable bit in Dword 0 of the corresponding completion entry shall be set to 1.

4. Added Test 9.3.3 per requirements in NVMe-MI TP 6016b that Get Features commands for the Host Metadata FIDs (7Dh, 7Eh, 7Fh), if Generate Default Host Metadata (GDHM) field is cleared to '0' then the device shall not generate any vendor specific strings for the Element Types of the specified Host Metadata feature.
5. Added Test 9.3.4 per requirements in NVMe-MI TP 6016b that for Set Features commands for the Host Metadata FIDs (7Eh, 7Fh), with the Element Action field is cleared to 00b (Add/Replace Entry) and if the Metadata Element Descriptor with the specified Element Type does not exist in the specified Host Metadata Feature value, then the Controller shall create the descriptor in the specified Host Metadata Feature value with the value in the Host Metadata data structure.
6. Added Test 9.3.5 per requirements in NVMe-MI TP 6016b that for Set Features commands for the Host Metadata FIDs (7Eh, 7Fh), with the Element Action field is cleared to 00b (Add/Replace Entry) and one Metadata Element Descriptor with the specified Element Type exists in the specified Host Metadata Feature value, then the Controller shall replace with the value in the specified Host Metadata data structure.
7. Added Test 9.3.6 per requirements in NVMe-MI TP 6016b that for Set Features commands for the Host Metadata FID 7Dh, if the Element Action field is cleared to 00b (Add/Replace Entry) and the Feature Identifier field is set to Enhanced Controller Metadata, then the controller shall abort the Set Features command with Invalid Field in Command status and shall not change any Host Metadata Feature value.
8. Added Test 9.3.7 per requirements in NVMe-MI TP 6016b that for Set Features commands for the Host Metadata features, that if the Element Action field is set to 01b (Delete Entry Multiple), then the Controller shall delete all the specified Metadata Element Descriptors from the specified Host Metadata Feature value, if any. If none of the specified Metadata Element Descriptors are present in the specified Host Metadata Feature value, then the controller shall complete the Set Features command with a status of Successful Completion and shall not change any Host Metadata Feature value.
9. Added Test 9.3.8 per requirements in NVMe-MI TP 6016b that for Set Features commands for the Host Metadata features, that if the Element Action field is set to 10b (Add Entry Multiple), the Feature Identifier field is set to Enhanced Controller Metadata, and no Metadata Element Descriptor with the specified Element Type exists in the Enhanced Controller Metadata Feature value, then the controller shall create new Metadata Element Descriptors in the Enhanced Controller Metadata Feature value with the Element Type and the value specified in the Host Metadata data structure.
10. Added Test 9.3.9 per requirements in NVMe-MI TP 6016b that for Set Features commands for the Host Metadata features, that if the Element Action field in a Set Features command, if the Element Action field is set to 10b (Add Entry Multiple),

the Feature Identifier field is set to Enhanced Controller Metadata, and one or more Metadata Element Descriptors with the specified Element Type exists in the Enhanced Controller Metadata Feature value, then the controller shall add the specified Metadata Element to the Enhanced Controller Metadata Feature value and shall not modify any existing Metadata Element Descriptors.

11. Added Test 9.3.10 per requirements in NVMe-MI TP 6016b that for Set Features commands for the Host Metadata features, that if the Element Action field is set to 10b (Add Entry Multiple) and the Feature Identifier field is not set to Enhanced Controller Metadata, then the controller shall abort the Set Features command with status Invalid Field in Command and shall not change the Host Metadata Feature value.
12. Added Test 9.3.11 per requirements in NVMe-MI TP 6016b that for a Set Features command that is submitted for a Host Metadata Feature, and host software attempts to add or replace a Metadata Element that causes the Host Metadata Feature value of the specified feature to grow larger than 4 KiB, then the controller shall abort the command with an Invalid Field In Command.
13. Added Test 9.3.12 per requirements in NVMe-MI TP 6016b that the default value for the Number of Metadata Element Descriptors of the Enhanced Controller Metadata Feature shall be 0h on a Controller Level Reset.
14. Added Test 9.3.13 per requirements in NVMe-MI TP 6016b that if a Get Features command with the SEL field set to 011b (i.e., Supported Capabilities) with the Enhanced Controller Metadata Feature value is submitted, then the NS Specific bit in Dword 0 of the corresponding completion queue entry shall be cleared to '0'.
15. Added Test 9.3.14 per requirements in NVMe-MI TP 6016b that if a Get Features command with the SEL field set to 011b (i.e., Supported Capabilities) with the Namespace Metadata Feature value is submitted, then the NS Specific bit in Dword 0 of the corresponding completion queue entry shall be set to '1'.
16. Test 11.5 added per NVMe v1.1 TP 6013, to check requirements that each NVMe Storage Device FRU shall have a FRU Information Device with a size of 256 to 4096 bytes which contains the VPD. This test also check that if the NVMe Storage Device contains an SMBus/I2C port, then the SMBus/I2C Address Info field in the FRU Information Device Element Descriptor is correct, if no SMBus/I2C port is present, then this field shall be cleared to 0h.
17. Test 11.6 added per NVMe v1.1 TP 6013, to check that FRU Information Device can be accessed by I2C/SMBus Read. This test also checks that the SMBus/I2C Address Info field in the FRU Information Device Element Descriptor is correct.
18. Test 11.7 added per NVMe v1.1 TP 6013, to check requirements that updating the VPD by writing to the FRU Information Device using I2C Writes shall not be supported if the VPD Write command is supported.
19. Test 11.8 added per NVMe v1.1 TP 6013, to check requirements that 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.

20. Test 12.1 updated per NVMe-MI V1.1 ECN 002, to check requirements that a reset of a Management Endpoint in an NVM Subsystem shall not affect any other Management Endpoint in the NVM Subsystem or any other NVM Subsystem entity.
21. Test 12.2 added per NVMe-MI v1.1 TP 6015 regarding SMBus Reset Enhancements to check that an SMBus Reset shall be treated by each Command Slot in the SMBus/I2C Management Endpoint as if an implicit Abort Control Primitive was received with the exception that the Management Endpoint does not transmit the Abort Control Primitive Response Messages.

2022 January 21 (Version 17.0 release)
Tim Sheehan:

- Program Revision Update.

2022 July 14 (Version 18.0 release)
Tim Sheehan:

1. Test 7.7 added per NVMe-MI v1.1 TP6020a.
2. Test 9.2.2 added per NVMe-MI v1.1 TP6024
3. Test 7.6 added per NVMe-MI V1.2 TP6029
4. Tests 8.8.3&4, 8.9.3&4, 8.10.2&3, 4.7, 8.12.3, and 4.8 modified per NVMe-MI V1.2 TP6029
5. Tests 11.1-8 modified version check

2023 January 04 (Version 19.0 release)
Tim Sheehan:

1. Test 9.2.3 added per TP6026
2. Tests 12.3 & 12.4 added per TP6027
3. Test 9.3.15 added per TP6025a
4. Test 9.6 added per TP6030

ACKNOWLEDGMENTS

The UNH–IOL would like to acknowledge the efforts of the following individuals in the development of this test plan:

Gordon Getty	Teledyne-LeCroy
David Woolf	UNH-IOL
Tim Sheehan	UNH-IOL

INTRODUCTION

The University of New Hampshire's InterOperability Laboratory (IOL) is an institution designed to improve the interoperability of standards–based products by providing a neutral environment where a product can be tested against other implementations of a common standard, both in terms of interoperability and conformance. This particular suite of tests has been developed to help implementers evaluate the NVMe-MI functionality of their products. This test suite is aimed at validating products in support of the work being directed by the NVMe Organization.

These tests are designed to determine if a product conforms to specifications defined in the NVMe Management Interface specification, hereafter referred to as the "NVMe-MI Specification"). Successful completion of these tests provide a reasonable level of confidence that the Device Under Test (DUT) will function properly in many NVMe-MI environments.

REFERENCES

The following documents are referenced in this text:

1. NVM Express Management Interface 1.1d, March 11 2021

Group 1: MCTP Base Tests

Test 1.1 – MCTP Endpoint ID – (FYI)

Purpose: Source Endpoint ID should be set to the assigned value

References:
MCTP Base Specification Section 8.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get Endpoint ID MCTP command
3. Wait for response.

Observable Results:
1. Ensure that the Endpoint ID returned by Get Endpoint ID command is the one assigned during MCTP initialization. f returned correctly, then result is PASS, otherwise, result is FAIL

Possible Problems: None

Test 1.1.1 – MCTP Endpoint ID Response – Reserved Bits – (FYI)

Purpose: The intent of this test is to ensure that all the reserved bits of the response are set to zero in the Get Endpoint ID Response

References:
MCTP Base Specification Section 8.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get EndPointID command
3. Wait for the Response Message

Observable Results:
1. Ensure that the Get Endpoint ID command returns with successful completion.
2. Ensure that the following bits are reserved:
    a. Bits 7:6 of byte 3 are reserved.
    b. Bits 2:0 of byte 3 are reserved.
    c. Endpoint Type (Bits 5:4 of byte 3) have values 10b and 11b reserved.

Possible Problems: None

Test 1.2 – MCTP Packet Sequence Number – (FYI)

Purpose: After the SOM packet, the packet sequence number must increment modulo 4 for each subsequent packet belonging to a given message up through the packet containing the EOM flag.

References:
MCTP Base Specification Section 8.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get Transmission Unit Size
3. Wait for Response Message
4. Execute NVMe-MI VPD Read with Data Offset = 0 and Data Length = 256.
5. Wait for Response

Observable Results:
1. Ensure that the Get Endpoint ID command returns with successful completion.
2. Ensure that the Get Transmission Unit Size returns the default value of 64
3. Ensure that the response of VPD Read is split into 5 MCTP packets.
4. Ensure that the sequence number of the packets are a modulo of 4 and are in order.

Possible Problems: None

Test 1.3 – MCTP Tag Owner and Message Tag Bits – (FYI)

Purpose: For messages that are split up into multiple packets, the Tag Owner (TO) and Message Tag bits remain the same for all packets from the SOM through the EOM.

References:
MCTP Base Specification Section 8.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get Transmission Unit Size
3. Wait for Response Message
4. Execute NVMe-MI VPD Read with Data Offset = 0 and Data Length = 256.
5. Wait for Response;

Observable Results:
1. Ensure that the Get Endpoint ID command returns with successful completion.
2. Ensure that the Get Transmission Unit Size returns the default value of 64
3. Ensure that the Tag Owner and Message Bits of the VPD Read response MCTP packets are the same,
1. Starting from the First message (SOM = 1, EOM = 0) to the last packet (SOM = 0, EOM = 1).

Possible Problems: None

Test 1.4.1 – MCTP Bad Packet 1 – (FYI)

Purpose: Individual packets are dropped (silently discarded) by an endpoint under the following conditions. These packets are discarded before being checked for acceptance or rejection for message assembly. Therefore, these packets will not cause a message assembly to be started or terminated.
Test case 1.4.1: Unexpected “middle” packet or “end” packet will be properly handled

References:
MCTP Base Specification Section 8.6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get Endpoint ID command, wait for response
3. Send MCTP packet (Get Endpoint ID command) with SOM and EOM not set ("middle" packet)
4. Wait for 126 milliseconds (MT2, response timeout)
5. Execute Get Endpoint ID command, wait for response
6. Send MCTP packet (Get Endpoint ID command) with SOM not set and EOM set ("end" packet)
7. Wait for 126 milliseconds (MT2, response timeout)
8. Execute Get Endpoint ID command, wait for response

Observable Results:
1. Ensure that the Get Endpoint ID commands with "middle" packet designation is silently dropped by the DUT.
2. Ensure that the Get Endpoint ID commands with "end" packet designation is silently dropped by the DUT.
3. Ensure that the proper responses for correct Get Endpoint ID commands are returned by the DUT.
1. If all of the above true, then result is PASS, otherwise result is FAIL

Possible Problems: None

Test 1.4.2 – MCTP Bad Packet 2 – (FYI)

Purpose:
Bad Message Integrity Check – Ensure that all Get EndPointID requests with a bad Message Integrity Check are dropped silently by the DUT
These packets are discarded before being checked for acceptance or rejection for message assembly.
Therefore, these packets will not cause a message assembly to be started or terminated.

References:
MCTP Base Specification Section 8.8

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Configuration Get command
3. Wait for the Response Message
4. Execute Configuration Get with a wrong Message Integrity Check
5. Allow for a NACK response message from the DUT.
6. Execute Configuration Get command
7. Wait for the Response Message

Observable Results:
1. Ensure that the Configuration Get returns a successful completion.
2. Ensure that the Configuration Get with wrong message integrity check is dropped silently.
3. Ensure that the Last Configuration Get returns a successful completion.

Possible Problems: None

Test 1.4.3 – MCTP Bad Packet 3 – (FYI)

Purpose: Individual packets are dropped (silently discarded) by an endpoint under the following conditions. These packets are discarded before being checked for acceptance or rejection for message assembly. Therefore, these packets will not cause a message assembly to be started or terminated.
Test 1.4.3: Bad, unexpected, or expired message tag is handled correctly.

References:
MCTP Base Specification Section 8.6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get Endpoint ID command, wait for response
3. Send MCTP packet (Get Endpoint ID command) with TO bit set to zero
4. Wait for 126 milliseconds (MT2, response timeout)
5. Execute Get Endpoint ID command, wait for response

Observable Results:
1. Ensure that the Get Endpoint ID command with bad tag owner is silently dropped by the DUT.
2.
3. Ensure that the proper responses for correct Get Endpoint ID commands are returned by the DUT. If all of the above true, then result is PASS, otherwise result is FAIL

Possible Problems: None

Test 1.4.4 – MCTP Bad Packet 4 – (FYI)

Purpose: Individual packets are dropped (silently discarded) by an endpoint under the following conditions. These packets are discarded before being checked for acceptance or rejection for message assembly. Therefore, these packets will not cause a message assembly to be started or terminated.
Test 1.4.4: Unknown Destination EID.

References:
MCTP Base Specification Section 8.6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get EndPointID command
3. Wait for the Response Message
4. Repeat Get Endpoint Id command with Destination EID set to 0x11 (invalid), 0x22 (invalid), 0x33 (invalid),0xAB (valid)
5. Allow for a NACK response message from the DUT.

Observable Results:
1. Ensure that the Get Endpoint ID command returns with successful completion.
2. Ensure that the Get Endpoint IDs with a different Destination EPID is dropped silently by the DUT.
3. Ensure that the last Get Endpoint ID command returns a successful completion

Possible Problems: None

Test 1.4.5 – MCTP Bad Packet 5 – (FYI)

Purpose: Individual packets are dropped (silently discarded) by an endpoint under the following conditions.
These packets are discarded before being checked for acceptance or rejection for message assembly.
Therefore, these packets will not cause a message assembly to be started or terminated.
Test 1.4.5: Bad header version is handled correctly

References:
MCTP Base Specification Section 8.6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get Endpoint ID command, wait for response
3. Send MCTP packet (Get Endpoint ID command) with bad header version (set to 2 instead of 1)
4. Wait for 126 milliseconds (MT2, response timeout)
5. Execute Get Endpoint ID command, wait for response

Observable Results:
1. Ensure that the Get Endpoint ID command with bad header version is silently dropped by the DUT.
2. Ensure that the proper responses for correct Get Endpoint ID commands are returned by the DUT.
3. If all of the above true, then result is PASS, otherwise result is FAIL

Possible Problems: None

Group 2: MCTP Control Message Tests

Test 2.1 – MCTP Control Instance ID – (FYI)

Purpose: Instance ID in the MCTP Response message should be set to the same value as in the MCTP Request message

References:
MCTP Base Specification Section 10.3

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get EndPointID command
3. Wait for the Response Message
4. Execute Get EndPointID command with instance ID set to 0x6, 0xC, 0x11, 0x1F

Observable Results:
1. Ensure that the Get Endpoint ID command returns with successful completion.
2. Ensure that the Datagram Bit is set to 0 and that the instance ID of the Get EndPointID Request and the Instance id of the Get EndPointID of the Response are the same.
3. Repeat the same for all instance IDs.

Possible Problems: None

Group 3: MCTP Commands

Test 3.1 – MCTP Command Set Endpoint ID – (FYI)

Purpose: The intention of this test is to ensure that proper Completion Messages are returned for mandatory MCTP Command Control Messages

References:
MCTP Base Specification Section 11.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization

Observable Results:
1. Ensure that the response to MCTP Command Set Endpoint ID returns with successful completion.
2. Ensure that Bits 7:6 of byte 2 are reserved
3. Ensure that Bits 5:4 of byte 2 of response can only have 00b and 01b as values, 10b and 11b are reserved.
4. Ensure that Bits 3:2 of byte 2 are reserved.
5. Ensure that Bits 1:0 of Byte 2 can only have 00b, 10b, and 01b as values, 11b is reserved.
6. Ensure that if Endpoint ID Assignment Status == 00b, EID Setting (byte 3) = 0xAB

Possible Problems: None

Test 3.2 – MCTP Command Get MCTP Version – (FYI)

Purpose: The intention of this test is to ensure that proper Completion Messages are returned for mandatory MCTP Command Control Messages

References:
MCTP Base Specification Section 11.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.
2. Perform MCTP Get Version Support with Message Type Number set to 0xFF (MCTP base specification Version information).
3. Wait for Response.
4. Perform MCTP Get Version Support with Message Type Number set to 0x00(MCTP Control Protocol Message Version Information).
5. Wait for Response.
6. Perform MCTP Get Version Support with Message Type Number set to 0x10(MCTP Control Protocol Message Version Information).
7. Wait for Response.
8. Perform MCTP Get Version Support with Message Type Number set to 0x70(MCTP Control protocol message version information)
9. Wait for Response.

Observable Results:
1. Ensure that the response to MCTP Command Get MCTP version support returns with successful completion and return Message Type not supported if Completion code is 0x80.
2. Ensure that the number of Version number entries (32 bit version numbers) is equal to the Version Number entry Count (one based).
3. Repeat the above for Message Type Number = 0x00.
4. Ensure that for Message Type Number = 0x10 and 0x70, the Completion Code in Response Data is set to 0x80.

Possible Problems: None

Test 3.3 – MCTP Command Get Message Type – (FYI)

Purpose: The intention of this test is to ensure that proper Completion Messages are returned for mandatory MCTP Command Control Messages

References:
MCTP Base Specification Section 11.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.
2. Perform MCTP Get Message Type Support.
3. Wait for Response.

Observable Results:
1. Ensure that the response to MCTP Command Get Message Type returns with successful completion.

Ensure that the Number of bytes in the Message Data corresponds to the MCTP Message Type Count in the Response Data.ie.: MCTP Message Type Count = (Length of Response Data) - 1 - 1.

Possible Problems: None

Test 3.4 – MCTP Command Prepare for Endpoint Discovery – (FYI)

Purpose: The intention of this test is to ensure that proper Completion Messages are returned for mandatory MCTP Command Control Messages

References:
MCTP Base Specification Section 11.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.

Observable Results:
1. Ensure that the response to MCTP Command Prepare for EndPoint Discovery returns with successful completion.

Note: SMBus should return ERROR_UNSUPPORTED_CMD as Completion Code.

Possible Problems: None

Test 3.5 – MCTP Command Endpoint Discovery – (FYI)

Purpose: The intention of this test is to ensure that proper Completion Messages are returned for mandatory MCTP Command Control Messages

References:
MCTP Base Specification Section 11.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.

Observable Results:
1. Ensure that the response to MCTP Command EndPoint Discovery returns with successful completion.

Possible Problems: None

Test 3.6 – MCTP Command Get Endpoint ID – (FYI)

Purpose: The intention of this test is to ensure that proper Completion Messages are returned for mandatory MCTP Command Control Messages

References:
MCTP Base Specification Section 11.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization

Observable Results:
1. Ensure the response returns with Successful Completion and that all the reserved bits of the response are set to zero in the Get EndPointID Response.

Possible Problems: None

Group 4: NVMe Error Handling

Test 4.1 – NVMe-MI Invalid Opcode – (Mandatory)

Purpose: Invalid Command Opcode Status response should be sent when the request Opcode is set to 08h – BFh

References:
NVMe-MI Base Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Configuration Get MI Command (Cfg ID 0x1) to the Management Endpoint Command Slot 0, wait for response
3. Perform Configuration Get MI Command (Cfg ID 0x1) to the Management Endpoint Command Slot 1, wait for response
4. Perform Management Interface Command with opcode 0x08 to the Management Endpoint Command Slot 0, wait for response
5. Perform Management Interface Command with opcode 0x29 to the Management Endpoint Command Slot 1, wait for response
6. Perform Management Interface Command with opcode 0x5A to the Management Endpoint Command Slot 0, wait for response
7. Perform Management Interface Command with opcode 0xBE to the Management Endpoint Command Slot 1, wait for response

Observable Results:
1. Ensure that for the Configuration Get commands the Response Messages are returned properly.
2. Ensure that for the commands with reserved opcodes the Response Messages are returned with Invalid Command Opcode Status.
3. If returned correctly, then result is PASS, otherwise, result is FAIL

Possible Problems: None

Test 4.2 – NVMe-MI Reserved Identifier – (Mandatory)

Purpose: Configuration Get: Specifying a reserved identifier in the Configuration Identifier field causes the command to complete with an Invalid Parameter error status.

References:
NVMe-MI Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Configuration Get Command requesting MCTP Transmission Unit Size
3. Wait for the Response message
4. Perform Configuration Get Command with Configuration Identifier set to 0
5. Wait for the Response message
6. Repeat the above for Configuration Identifier values of 0x04, 0x56, 0xBF

Observable Results:
1. Ensure that for the first command proper response is returned, and status is set to Success;
2. Ensure that for the rest of the commands:
    a. Invalid Parameter Error Response is sent
    b. In the Parameter Error Location field of the response, the byte position is set to 8 and bit position - to 0.

Possible Problems: None

Test 4.3 – NVMe-MI Health Status Change – (Mandatory)

Purpose: For Health Status Change (Configuration Identifier 02h) - A Management Endpoint shall complete a Configuration Get command on this Configuration Identifier with a Success Response Message. The NVMe Management Response field is reserved and there is no Response Data.

References:
NVMe-MI Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization
2. Execute Configuration Get with Configuration Identifier set to 0x02(Health Status Change
3. Wait for Response message

Observable Results:
1. Ensure that Configuration get command returns with successful completion.
2. Ensure that the NVMe Management Response field is reserved and that no Response data was returned.
3. (NVMe Management Response Field - bits 8:31 of dw1 of the response)

Possible Problems: None

Test 4.4 – NVMe-MI Reserved Configuration Identifier – (Mandatory)

Purpose: Configuration Set:

- Specifying a reserved identifier in the Configuration Identifier field causes the command to complete with an Invalid Parameter error status.- SMBus/I2C Frequency (Configuration Identifier 01h). If the specified frequency is not supported or the Port Identifier specified is not an SMBus/I2C port, the Management Endpoint shall respond with an Invalid Parameter error status.
  (Can use Reserved value of 0 or 4-F)
- MCTP Transmission Unit Size (Configuration Identifier 03h). If the specified MCTP Transmission Unit Size is not supported or the Port Identifier specified is not valid, the Management Endpoint shall abort the command and send a Response Message with an Invalid Parameter error status.

References:
NVMe-MI Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:

1. Perform MCTP Initialization
2. (SMBus only) Perform Configuration Get to ensure SMBus is supported
3. Perform Configuration Set Command with Configuration Identifier 0x03 (MCTP Transmission Unit Size: Size = 64 bytes)
4. Wait for the Response message
5. Perform Configuration Set Command with Configuration Identifier set to 0
6. Wait for the Response message
7. Repeat the above for Configuration Identifier values of 0x04, 0x56, 0xBF
8. Perform Configuration Set Command with Configuration Identifier set to SMBus/I2C Frequency and Frequency Set to 100 kHz
9. Wait for Response Message
10. Repeat the above for Frequency set to 0h, 4h, and Fh and non- SMBus/I2C port

Observable Results:

1. Ensure that the Set Configuration command returns with successful completion.
2. Ensure that for the rest of the commands with invalid Configuration Identifier, Invalid Parameter Error Response is sent.
3. Ensure that Set Configuration - SMBus/I2C Frequency command returns with successful completion if Frequency is supported, otherwise should return Invalid Parameter response
4. Ensure that for the rest of the commands an Invalid Parameter error response is sent.

Possible Problems: None

Test 4.5 – NVMe-MI MAXRENT Error – (Mandatory)

Purpose: Controller Health Status Poll: For Maximum Response Entries (MAXRENT) in Request message. Specifying 256 entries is interpreted as an Invalid Field. Command to complete with an Invalid Parameter error status.

References:
NVMe-MI Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: Feb 3 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Controller Health Status Poll Command with the following parameters:
    a. -Report all set:
    b. Include PCI Functions set
    c. Controller Status Changes set
    d. -MAXRENT set to 0xFF (which results in 256 entries, as this is zero-based value), all other parameters set to zero.
3. Wait for the Response message.

Observable Results:
1. Ensure that the proper response is returned, and status is set to Invalid Parameter.
2. Ensure that for the next command:
    a. Invalid Parameter Response is set
    b. In the Parameter Error Location field of the response the byte position is set to 10 and bit position - to 0.

Possible Problems: None

Test 4.6 – NVMe-MI Reserved Data Structure Type – (Mandatory)

Purpose: Read NVMe-MI Data Structure:

- If Data Structure Type (DTYP) in Request is set to reserved value, the command shall complete with an Invalid Parameter error status. (Implicit)

References:
NVMe-MI Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: November 13, 2019

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Read NVMe-MI Data Structure Command with Data Structure Type set to a reserved value, such as 0xFF
3. Wait for the Response message
4. Repeat the above Data Structure Type values of 0x23, 0x56, 0x9A, 0xFF.

Observable Results:
1. Ensure that for each command the proper response is returned, and status is set to Invalid Parameter.
2. Ensure that for the rest of the commands:
    a. Invalid Parameter Error Response is sent
    b. In the Parameter Error Location field of the response the byte position is set to 11 and bit position - to 0.

Possible Problems: Some values that were reserved in NVMe-MI v1.0 are used in NVMe-MI 1.1, so any test implementation should pay attention to these differences.

Test 4.7 – NVMe-MI Invalid VPD Read Size -(FYI)

Purpose: VPD Read:

- If the Data Length plus Data Offset fields are greater than the size of the VPD, then the Management Endpoint does not return the VPD contents and responds with an Invalid Parameter error status response

References:
NVMe-MI Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Read the NVMe Subsystem Information Data Structure to get the MJR & MNR version supported.
3. Perform Read VPD with Data Length set to 7 bytes and Offset to 1 byte
4. Wait for the Response message
5. Perform Read VPD with Data Length set to 5 bytes and Offset to 260 bytes
6. Wait for the Response message
7. Perform Read VPD with Data Length set to 300 bytes and Offset to 5 bytes
8. Wait for the Response message

Observable Results:
1. Ensure that for First Command proper response is returned and status is set to 'success'
2. Ensure that for the rest of the commands:Invalid Parameter Error Response is sent;
3. Verify if the MJR=1 and MNR=2, then the PEL field indicates DLEN field
4. In the Parameter Error Location field of the response the byte position is set to 8 (0x08) and bit position - to 0.
    a. flagging the length field in NVMe management DWord 1 would be correct
    b. If the offset is not within range then the byte position 8 should be returned. If the offset is good, but the length causes the boundary to be exceeded then 0xc should be the byte indication

Possible Problems: None

Test 4.8 – NVMe-MI Invalid VPD Write Status – (Mandatory)

Purpose: VPD Write:
The intent of this test is to ensure that the VPD Write returns Parameter Error Response when the Data length and Data Offset Fields greater than the size of VPD.

References:
NVMe-MI Specification Chapter 4.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: January 29, 2018

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Write VPD with Data Length set to 7 bytes and Offset to 1 byte
3. Wait for the Response message
4. Perform Write VPD with Data Length set to 5 bytes and Offset to 260 bytes
5. Wait for the Response message
6. Perform Write VPD with Data Length set to 300 bytes and Offset to 5 bytes
7. Wait for the Response message

Observable Results:
1. Ensure that for First Command proper response is returned and status is set to 'success'.
2. Ensure that for the rest of the commands:
    a. Invalid Parameter Error Response is sent
    b. In the Parameter Error Location field of the response the byte position is set to 10 (0x0A) and bit position - to 0.
    c. If DOFST is out of bounds then 0x8 should be returned, If DOFST is valid and DLEN causes too large, then 0xC is returned.
1. If there are multiple error conditions detected by the DUT, then the response may be an error code other than ‘Invalid Parameter Error’. This is acceptable behavior.

Possible Problems: None

Test 4.9 – NVMe-MI Invalid Parameter Status – (Mandatory)

Purpose: Invalid Parameter Status response shall be sent when Command message contains invalid parameter. The Parameter Error Location in the Response shall be set to the proper location of the invalid parameter.

References:
NVMe-MI Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: November 13, 2019

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Start with Endpoint discovery
2. Send Configuration Get TUSize command to Command Slot 0 with Invalid Config Id = 0x04
3. Wait for response
4. Send NVMe-MI Read VPD with Offset set to more than 256
5. Wait for response
6. Send Read NVMe-MI Data Structure Command with invalid DTYP set to a reserved value such as 0xFF.
7. Wait for response

Observable Results:
1. Ensure that Invalid Parameter Status response shall be sent when Command message contains invalid parameter.

Possible Problems: Some values that were reserved in NVMe-MI v1.0 are used in NVMe-MI 1.1, so any test implementation should pay attention to these differences.

Test 4.10 – NVMe-MI Invalid Command Size – (Mandatory)

Purpose: Invalid Command Size Status response should be sent when Command message contains too much/too little data.

References:
NVMe-MI Specification

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform Endpoint Discovery
2. Send Configuration Get TUSize command to Command Slot 0
3. Wait for response
4. Send Configuration Get TUSize command to Command Slot 0 with Length = 12
5. Wait for response

Observable Results:
1. Ensure that Invalid Command size response shall be sent when Command message contains an Invalid Command Size.

Possible Problems: None

Group 5: NVMe Management Interface Tests

Test 5.1 – NVMe-MI Message Type – (Mandatory)

Purpose: The Message Type field specifies the type of payload contained in the message body and is required to be set to 4h in all messages associated with NVMe-MI

References:
NVMe-MI Specification Section 3.2.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Get State Control Primitive Request to the Management Endpoint Command Slot 0
3. Wait for the Response Message

Observable Results:
1. Ensure the Message type in the Response Message is set to 4.
2. If returned correctly, then result is PASS, otherwise, result is FAIL

Possible Problems: None

Test 5.2 – NVMe-MI Message IC – (Mandatory)

Purpose: All NVMe-MI messages are protected by a CRC and thus IC bit shall be set to '1' in all NVMe-MI messages.

References:
NVMe-MI Specification Section 3.2.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization
2. Execute NVMe-MI Configuration Get command
3. Wait for Response
4. Execute NVMe-MI Configuration Get command with IC bit set to zero
5. Wait for Response
6. Execute NVMe-MI Configuration Get command
7. Wait for Response

Observable Results:
1. Ensure that the Configuration Get command returns with successful completion.
2. Ensure that the second Get command with IC bit set to zero is dropped silently.
3. Ensure that the third Get command returns with successful completion.

Possible Problems: None

Test 5.3 – NVMe-MI CRC Check – (Mandatory)

Purpose: The Message Integrity Check field contains a 32-bit CRC computed over the contents of the NVMe-MI message. The 32-bit CRC used by NVMe-MI is CRC-32C (Castagnoli) which uses the generator polynomial 1EDC6F41h

References:
NVMe-MI Specification Section 3.2.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform Endpoint discovery
2. Perform Configuration Get - Transmission Unit Size
3. Wait for response
4. Send Control Primitive with OpCode = Replay
5. Wait for response

Observable Results:
1. Ensure that CRC check passed

Possible Problems: None

Test 5.4 – NVMe-MI Command Slot – (Mandatory)

Purpose: For Response Messages, the CSI field indicates the Command Slot associated with the Request Message with which the Response Message is associated.
(Multi Message checking – use baseline TU Size of 64 and use VPD Read Command – up to 256 bytes)

References:
NVMe-MI Specification Section 3.2.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Get State Control Primitive Request to the Management Endpoint Command Slot 0, wait for response
3. Perform Get State Control Primitive Request to the Management Endpoint Command Slot 1, wait for response
4. Perform Configuration Get MI Command (Cfg ID 0x1) to the Management Endpoint Command Slot 0, wait for response
5. Perform Configuration Get MI Command (Cfg ID 0x1) to the Management Endpoint Command Slot 1, wait for response.

Observable Results:
1. Ensure that for each command the CSI field in the Response Message matches the CSI field in the Request message.
2. If returned correctly, then result is PASS, otherwise, result is FAIL

Possible Problems: None

Test 5.5 – NVMe-MI MCTP packet padding – (Mandatory)

Purpose: The MCTP Transmission Unit size of the last packet in a Request Message or Response Message (i.e., the one with the EOM bit set in the MCTP header) shall be the smallest size needed to transfer the MCTP Packet Payload for that Packet with no additional padding beyond any padding required by the physical medium-specific trailer.

References:
NVMe-MI Specification Section 3.2.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get Transmission Unit Size
3. Wait for Response Message
4. Execute NVMe-MI VPD Read with Data Offset = 0 and Data Length = 256.
5. Wait for Response

Observable Results:
1. Ensure that the Get Endpoint ID command returns with successful completion during initialization.
2. Ensure that the Get Transmission Unit Size returns the default value of 64
3. Ensure that the MCTP packet with EOM = 1 in the VPD Response Message has the smallest size and that the start packet and every other middle packet is bigger than the end packet.

Possible Problems: None

Test 5.6 – NVMe-MI Message Integrity Check – (Mandatory)

Purpose: Once a complete NVMe-MI MCTP message has been assembled, the Message Integrity Check is verified. If the Message Integrity Check passes, then the message is processed. If the Message Integrity Check fails, then the message is discarded

References:
NVMe-MI Specification Section 3.2.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform Endpoint Discovery
2. Send Configuration Get TUSize command to Command Slot 0
3. Wait for response
4. Send Configuration Get TUSize command to Command Slot 0
5. Wait for 150 milliseconds to exceed maximum possible latency and ensure that no response was sent
6. Send Configuration Get TUSize command to Command Slot 0
7. Wait for response

Observable Results:
1. Check that message is dropped when MIC fails

Possible Problems: None

Group 6: NVMe-MI Message Processing Tests

Test 6.1 – NVMe-MI Reserved Fields – (Mandatory)

Purpose: The purpose of this test is to check the behavior of the Response Message reserved fields

References:
NVMe-MI Specification Section 6.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Get State Control Primitive Request to the Management Endpoint Command Slot 0, wait for response;
3. Perform Configuration Get MI Command (Cfg ID 0x1) to the Management Endpoint Command Slot 0, wait for response

Observable Results:
1. Ensure that for each command the Reserved fields in the first DWord of the Response Message are set to zero.
2. If returned correctly, then result is PASS, otherwise, result is FAIL

Possible Problems: None

Test 6.2 – NVMe-MI Error Response Code – (Mandatory)

Purpose: The purpose of this test is to check that the Generic Error Response with the proper code is returned for Invalid Command Opcode (0x03), and for Invalid Command Size (0x05) and Invalid Command Input Data Size (0x06)

References:
NVMe-MI Specification Section 6.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: July 17, 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform Endpoint Discovery
2. Send normal Configuration Get NVMe-MI command, Cmd slot 0
3. Wait for response
4. Send NVMe-MI command with undefined opcode 0x08
5. Wait for response
6. Send Configuration Get SmBusI2C Freq with Length = 12
7. Wait for response

Observable Results:
1. Check that the Generic Error Response with the proper code is returned for Invalid Command Opcode (0x03), and for Invalid Command Size (0x05) and Invalid Command Input Data Size (0x06)

Possible Problems: None

Test 6.3 – Command Initiated Auto Pause – (FYI)

Purpose: The purpose of this test is to check that the CPIA field is used properly for a NVMe-MI message.

References:
NVMe-MI Specification Section 3.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 30, 2020

Discussion: 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.

Test Setup: See Appendix A.

Test Procedure:
1. Perform Endpoint Discovery
2. Check Port Information Data Structure CIAPS field in the NVMe-MI Data Structure. If CIAPS is set to 0, this test is not applicable.
3. Send normal Configuration Get NVMe-MI command, with CIAP set to 1.
4. Wait for response

Observable Results:
1. Verify that the command completes successfully.

Possible Problems: None

Group 7: Control Primitives Tests

Test 7.1 – NVMe-MI Response Tag – (Mandatory)

Purpose: Tag in Response matches the Tag in Request

References:
NVMe-MI Specification Section 7.2
MCTP Base Specification Section 8.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Get State Control Primitive Request to the Management Endpoint Command Slot 0, with Tag value 0x01, wait for response
3. Repeat the above for Endpoint Command Slot 1.

Observable Results:
1. Ensure that for each command, proper response is returned, and the Tag value in Response matches the one in Request.
2. If returned correctly, then result is PASS, otherwise, result is FAIL

Possible Problems: None

Test 7.2 – NVMe-MI Response Message – (Mandatory)

Purpose: The intent of this test is to ensure that a DUT returns successful response for the Pause, Resume and Abort Control Primitives.

References:
NVMe-MI Specification Section 7.3

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform Endpoint Discovery
2. Send Control Primitive with OpCode = Pause
3. Wait for response
4. Send Control Primitive with OpCode = Resume
5. Wait for response
6. Send Control Primitive with OpCode = Abort
7. Wait for response

Observable Results:
1. Ensure that a DUT returns successful response for the Pause, Resume and Abort Control Primitives.

Possible Problems: None

Test 7.3 – NVMe-MI Get State Primitive Response – (FYI)

Purpose: Response for the GetState primitive returns the correct state based on the conditions applied. This includes the state error bits and response to Clear Error State Flags

References:
NVMe-MI Specification Section 7.4

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 2, 2019

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Execute Get EndPointID command
3. Send Get State Control Primitive Request with CESF = 1
4. Send a second Get State Control Primitive Request with CESF = 1
5. Send Get EndPoint Id with a Bad Header Version
6. Send Get State Control Primitive with CESF = 0
7. Send Get State Control Primitive with CESF = 1
8. Send Get State Control Primitive with CESF = 0
9. Repeat the above with:
10. Send Get EndPoint Id with EOM = 0 and SOM = 0 (Middle Packet)
11. Send Get EndPoint Id with a Destination Endpoint ID = 0x11
12. Send Get EndPoint Id with a Tag Owner = 01
13. Send Configuration Get - MCTP Transmission Unit Size with CRC = 0x123AB
14. Send Configuration Get - MCTP Transmission Unit Size with LCRC = 0xAB (Only for VDM)

Observable Results:
1. Ensure that the Get State Control Primitive Response sent in response to the second Get State Control Primitive Request has bytes 07:08 are all cleared.
2. Ensure that Bit 7 is 1 for Bad Header Version.
3. Ensure that Bit 10 is 1 for the Middle Packet.
4. Ensure that Bit 8 is 1 for Unknown Destination ID.
5. Ensure that Bit 12 is 1 for Bad Tag Owner.
6. Ensure that Bit 4 is 1 for Bad CRC.
7. Ensure that Bit 13 is 1 for Bad LCRC. (Only for VDM)

Possible Problems: None

Test 7.4 – NVMe-MI Response Message Replay – (Mandatory)

Purpose: Verify that DUT replays the response message according to the RRO in request, state and the rules, and that the status and RR bit in the Response control primitive is set according to the state and rules

References:
NVMe-MI Specification Section 4.4.5, 7.5

Resource Requirements:

Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 24, 2020

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:

1. Perform MCTP initialization
2. Execute Get EndPointID command
3. Perform NVMe-MI Configuration Set - Transmission Unit Size
4. Wait for Response
5. Perform NVMe-MI Control Primitive Replay
6. Wait for Response

Observable Results:

1. Ensure that the Configuration Set command returns with success.
2. Ensure that the Replay Control primitive returns with success.
3. Ensure that bits 01:15 of Byte 07:08 are Reserved
4. Ensure that the Replayed Message was received successfully.
5. Verify that the first packet has SOM set and includes the Message Header of the original Response Message.

Possible Problems: None

Test 7.5 – NVMe-MI Response Replay Offset (RRO) – (Mandatory)

Purpose: Verify that DUT replays the response message according to the RRO in request

References:
NVMe-MI Specification Section 4.4.5, 7.5

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 23, 2020

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform NVMe-MI VPD Read with Data length set to 256 Bytes.
3. Wait for Response
4. Perform NVMe-MI Control Primitive Replay with Response Replay Offset set to 2.
5. Wait for Response

Observable Results:
1. Ensure that the VPD Read command returns with success.
2. Ensure that the Replay Control primitive returns with success.
3. Ensure that the Replayed Message was received successfully with 3 packets instead of 5 because of RRO.
4. Verify that the first packet has SOM set and includes the Message Header of the original Response Message even if the Response Replay Offset is not zero.

Test 7.6 – NVMe-MI RRO > length of Response Message – (FYI)

Purpose: Verify that DUT attempts to replay the response message according to the RRO that is greater than the Length of the Response Message

References:
NVMe-MI Specification Section 4.2.1.5

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: July 13, 2022

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform NVMe-MI VPD Read with Data length set to 4096 or larger number of Bytes.

3. Wait for Response
4. Perform NVMe-MI Control Primitive Replay with Response Replay Offset set to a large enough to exceed the Response Message.
5. Wait for Response

Observable Results:

1. Ensure that the VPD Read command returns with success.
2. Ensure that the Replay Control primitive returns with success.
3. Verify that the Replayed Message was aborted with a status of Invalid Parameter Error Response

   Possible Problems: If this is an NVMe-MI v1.1 or lesser device it is possible to response with a different status.

Test 7.7 – NVMe-MI Get State, MEB=1 – (FYI)

Purpose: Verify that DUT responds with an error when MEB =1

References:
NVMe-MI Specification Section 3.1.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: July 7, 2022

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:

1. Perform MCTP initialization
2. Perform Get State Control Primitive Request to the Management Endpoint Command Slot 0, with Tag value 0x01, and Header field Byte 2, bit 0 set to 0x01 (MEB), wait for response
3. Repeat the above for Endpoint Command Slot 1.

Observable Results:

1. Ensure that a DUT returns a status of Invalid Parameter Error Response

Possible Problems: None

Group 8: Management Interface Commands

Test 8.1 – NVMe-MI Response Header – (Mandatory)

Purpose: Reserved field, Integrity Check (verify correct CRC-32)

References:
NVMe-MI Specification Section 4.4

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: July 18, 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization
2. Send an NVMe-MI Configuration Get - Transmission Unit Size

Observable Results:
1. Ensure that a successful Response was returned.
2. Ensure that Byte 3, Byte 2, and bits 1, 2 of Byte 1, are reserved.
3. Ensure that the IC bit is set to 1.

Possible Problems: None

Test 8.2 – NVMe-MI Configuration Set – (Mandatory)

Purpose: Configuration Set.
Specifying a reserved identifier in the Configuration Identifier field causes the command to complete with an Invalid Parameter error status.

References:
NVMe-MI Specification Section 5.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization
2. Perform Configuration Set Command with Configuration Identifier 0x03 (MCTP Transmission Unit Size: Size = 64 bytes)
3. Wait for the Response message
4. Perform Configuration Set Command with Configuration Identifier set to 0
5. Wait for the Response message
6. Repeat the above for Configuration Identifier values of 0x04, 0x56, 0xBF

Observable Results:
1. Ensure that the Set Configuration command returns with successful completion.
2. Ensure that for the rest of the commands:
3. In the Parameter Error Location field of the response the byte position is set to 8 and bit position - to 0.

Possible Problems: None

Test 8.3 – NVMe-MI Config Get Response – (Mandatory)

Purpose: Responses to Config Get, Health Status Polls, and VPD Read return data in proper format
Configuration Get:
- SMBus/I2C Frequency. NVMe Management Response Bits 23:4 are reserved, bits 3:0 return the frequency encoding.
- MCTP Transmission Unit Size. NVMe Management Response Bits 23:4 are reserved, bits 3:0 return the unit size.
- Specifying a reserved identifier in the Configuration Identifier field causes the command to complete with an Invalid Parameter error status.

References:
NVMe-MI Specification Section 5.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Perform Configuration Get Command requesting MCTP Transmission Unit Size
3. Wait for the Response message
4. Perform Configuration Get Command requesting SMBus/I2C Frequency
5. Wait for the Response message

Observable Results:
1. Ensure that for both commands proper response is returned, and status is set to Success
2. Ensure that for the Command requesting MCTP Transmission Unit Size:
    a. Bits 23:16 of the NVMe Management Response field are reserved (set to zero)
    b. Bits 15:0 return the default unit size of 0x40
3. Ensure that for the Command requesting SMBus/I2C Frequency Unit Size:
    c. Bits 23:4 of the NVMe Management Response field are reserved (set to zero)
    d. Bits 3:0 return the valid frequency encoding (0 to 3)

Possible Problems: None

Test 8.4 – NVMe-MI Health Status Poll – (Mandatory)

Purpose: The intent of this test is to verify the reserved bits of the Controller Health Status Poll Response are set to zero.

References:
NVMe-MI Specification Section 5.3

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: February 15 2017

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization
2. Send Configuration Get Health Status Poll
3. Wait for response

Observable Results:
1. Ensure that the proper response is returned, and status is set to Success
2. Ensure that for the following bits are reserved in the response.
    a. Bits 23:0 in DWORD 1
    b. Bits 1:0 and 7:6 in DWORD 2
    c. Bits 31:13 in DWORD 3

Possible Problems: None

Test 8.5 – NVMe-MI Controller Health Status Poll – (M)

Purpose: Controller Health Status Poll:
- The length should correspond to RENT in Management response
- For each Controller Health data structure bytes 15:9 and bits 15:8 in bytes 322 are reserved

References:
NVMe-MI Specification Section 5.3

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 24, 2020

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Case 1: NVMe-MI 1.0 ECN 003 or NVMe-MI 1.0a or higher Not Implemented (M)

Test Procedure:
1. The following procedure is only applicable to a device implementing NVMe-MI 1.0 which has not implemented NVMe.MI 1.0 ECN003.
2. Perform MCTP initialization.
3. Perform Controller Health Status Poll Command with the following parameters:Report all set, Include PCI Functions set, Controller Status Changes set,MAXRENT set to 0x01, all other parameters set to zero.Wait for the Response message

Observable Results:
1. Ensure that the proper response is returned, and status is set to Success
2. Ensure that for the following bits are reserved in the response, and cleared to 0.
    a. Bits 15:8 in DWORD 1
3. Bits 31:24 in DWORD 2Bits 31:5 in DWORD 4Bits 31:0 in DWORD 5Ensure that if a Controller Health Data Structure is included in the response, that RENT (bits 23:16 in DWORD 1) use a 0's based value (i.e a value of 0 indicated 1 Data Structure included).

Case 2: NVMe-MI 1.0 ECN 003 or NVMe-MI 1.0a or higher Implemented (M)

Test Procedure:
1. The following procedure is only applicable to a device implementing NVMe.MI 1.0 which has implemented ECN003 and to devices will implement NVMe-MI 1.0a or higher.
2. Perform MCTP initialization
3. Perform Controller Health Status Poll Command with the following parameters: Report all set, Include PCI Functions set, Controller Status Changes set, MAXRENT set to 0x01, all other parameters set to zero.
4. Wait for the Response message

Observable Results:
1. Ensure that the proper response is returned, and status is set to Success
2. Ensure that for the following bits are reserved in the response, and cleared to 0.
    a. Bits 15:8 in DWORD 1
    b. Bits 31:24 in DWORD 2
    c. Bits 31:5 in DWORD 4
    d. Bits 31:0 in DWORD 5

3. Ensure that if a Controller Health Data Structure is included in the response, that RENT (bits 23:16 in DWORD 1) is set to a non-zero value.

Case 3: Controller Health Status Poll Filtering by Controller Selection (FYI)

Test Procedure:

1. The following procedure is only applicable to a device implementing NVMe.MI 1.0 which has implemented ECN003 and to devices will implement NVMe-MI 1.0a or higher.
2. Perform MCTP initialization
3. Perform Controller Health Status Poll Command with the following parameters:
   - ALL= 0, INCVF=1, INCPF=0, INCF=0, MAXRENT=255, SCTLID=0
   - CWARN=0, SPARE=0, PDLU=0, CTEMP=0
4. Wait for the Response message
5. Perform Controller Health Status Poll Command with the following parameters:
   - ALL= 0, INCVF=0, INCPF=1, INCF=0, MAXRENT=255, SCTLID=0
   - CWARN=0, SPARE=0, PDLU=0, CTEMP=0
6. Wait for the Response message
7. Perform Controller Health Status Poll Command with the following parameters:
   - ALL= 0, INCVF=0, INCPF=0, INCF=1, MAXRENT=255, SCTLID=0
   - CWARN=0, SPARE=0, PDLU=0, CTEMP=0
8. Wait for the Response message

Observable Results:

1. Verify that for each Controller Health Status Poll Command, only the data structures matching the Controller filtering requirements were returned.
2. In each case ensure that the proper response is returned, and status is set to Success
3. Ensure that for the following bits are reserved in the response, and cleared to 0.
   a. Bits 15:8 in DWORD 1
   b. Bits 31:24 in DWORD 2
   c. Bits 31:5 in DWORD 4
   d. Bits 31:0 in DWORD 5
4. Ensure that if a Controller Health Data Structure is included in the response, that RENT (bits 23:16 in DWORD 1) is set to a non-zero value.

Case 4: Controller Health Status Poll Filtering by Error Selection Fields (FYI)

Test Procedure:

1. The following procedure is only applicable to a device implementing NVMe.MI 1.0 which has implemented ECN003 and to devices will implement NVMe-MI 1.0a or higher.
2. Perform MCTP initialization
3. Perform Controller Health Status Poll Command with the following parameters:
   - ALL= 0, INCVF=0, INCPF=0, INCF=0, MAXRENT=255, SCTLID=0
   - CWARN=1, SPARE=0, PDLU=0, CTEMP=0
4. Wait for the Response message
5. Perform Controller Health Status Poll Command with the following parameters:
   - ALL= 0, INCVF=0, INCPF=0, INCF=0, MAXRENT=255, SCTLID=0
   - CWARN=0, SPARE=1, PDLU=0, CTEMP=0
6. Wait for the Response message
7. Perform Controller Health Status Poll Command with the following parameters:
   - ALL= 0, INCVF=0, INCPF=0, INCF=0, MAXRENT=255, SCTLID=0
   - CWARN=0, SPARE=0, PDLU=1, CTEMP=0
8. Wait for the Response message
9. Perform Controller Health Status Poll Command with the following parameters:
   - ALL= 0, INCVF=0, INCPF=0, INCF=0, MAXRENT=255, SCTLID=0
   - CWARN=0, SPARE=0, PDLU=0, CTEMP=1

10. Wait for the Response message

Observable Results:

1. Verify that for each Controller Health Status Poll Command, only the data structures matching the error selection filtering requirements were returned.
2. In each case ensure that the proper response is returned, and status is set to Success
3. Ensure that for the following bits are reserved in the response, and cleared to 0.
    a. Bits 15:8 in DWORD 1
    b. Bits 31:24 in DWORD 2
    c. Bits 31:5 in DWORD 4
    d. Bits 31:0 in DWORD 5
4. Ensure that if a Controller Health Data Structure is included in the response, that RENT (bits 23:16 in DWORD 1) is set to a non-zero value.

Possible Problems: NVMe-MI 1.0 ECN 003 changed the RENT field from being a zeroes based value to not being a zeroes based value. Products implementing NVMe-MI 1.0 may or may not implement NVMe-MI 1.0 ECN 003. Products implementing NVMe-MI 1.0a or higher must implement NVME-MI 1.0 ECN 003.

Test 8.6 – NVMe-MI Read Data Structure – (Mandatory)

Purpose:
The intent of this test is to verify the response length and reserved bits of the Get Read Subsystem information response for different Data Structure Types.
References:
NVMe-MI Specification Section 5.7

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: May 27, 2021

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP initialization
2. Send Configuration Read NVMe-MI Data Structure: with DTYP = 0x00
3. Send Configuration Read NVMe-MI Data Structure: with DTYP = 0x01
4. Send Configuration Read NVMe-MI Data Structure: with DTYP = 0x02
5. Send Configuration Read NVMe-MI Data Structure: with DTYP = 0x03
6. Send Configuration Read NVMe-MI Data Structure: with DTYP = 0x04

Observable Results:
1. Ensure that a proper response is returned with status set to Success
    a. Bits 23:16 of dw0 of the NVMe Management response are reserved for all responses
    b. Check Bits 15:00 of dw0 for response length for all responses
2. Ensure that for the when DTYP is set to Subsystem Information (0x00)
    a. Check that the response length = 12 + 32 = 44 bytes
    b. BYTES 31:03 are reserved
3. Ensure that for the when DTYP is set to Port Information (0x01)
    a. Check that the response length = 12 + 32 = 44 bytes
    b. BYTES 07:04 are reserved
    c. BYTE 9 bits 7:3 are reserved
    d. BYTES 31-13 are reserved
    e. BYTE 01 is reserved
4. Ensure that for the when DTYP is set to Controller List (0x02)
    a. Check that the response length = 12 + 8 = 20 bytes (4 bytes per controller)
5. Ensure that for the when DTYP is set to Controller Information (0x03)
    a. Check that the response length = 12 + 32 = 44 bytes
    b. BYTES 7:1 are reserved
    c. BYTES 4-1 are reserved
    d. BYTE 5 bits 7:1 are reserved
    e. BYTES 31:16 are reserved
6. Ensure that for the when DTYP is set to optional Commands supported (0x04)
    a. Check that the response length = 12 + 6 = 18 bytes (CMD0 and CMD1)
    b. bits 2:0 and bit 7 of BYTE 00 are reserved for each entry in the list
    c. Verify that the PCIe Supported Link Speeds Vector bits are set properly according to the PCIe Link Speeds supported by the DUT .

Possible Problems: None

Test 8.7 – NVMe-MI Data Length – (Mandatory)

Purpose: The intent of this test is to verify the response length of specific requested parameters.

References:
NVMe-MI Specification Section 5.7

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: November 26, 2018

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Case 1: Verify NVMSSI Data Length (M)

Test Procedure:
1. Perform Endpoint Discovery
2. Send Configuration Read NVMe-MI Data Structure: with DTYP = 0x00

Observable Results:
1. Verify the response length is 32

Case 2: Verify PortInfo Data Length (M)

Test Procedure:
1. Perform Endpoint Discovery
2. Send Configuration Read NVMe-MI Data Structure: with DTYP = 0x01

Observable Results:
1. Ensure Response Data Length is 32

Case 3: Verify CtrlrList Data Length (M)

Test Procedure:
1. Perform MCTP Initialization
2. -Perform Read NVMe-MI Data Structure with DTYP set to 0x02 (Controller List)
3. -Wait for Response Message

Observable Results:
1. Ensure that a proper response is returned with status set to Success
    a. Bits 23:16 of dw0 of the response are reserved for all responses
    b. Check Bits 15:00 of dw0 for response length for all responses
    c. Ensure that for the when DTYP is set to Controller List (0x02)
2. Check that the response length = 12 + 8 = 20 bytes (4 bytes per controller)

Case 4: Verify CtrlrInfo Data Length (M)

Test Procedure:
1. -Perform MCTP Initialization

2. -Perform Read NVMe-MI Data Structure with DTYP set to 0x03 (Controller Information)
3. -Wait for Response Message

Observable Results:

1. Ensure that a proper response is returned with status set to Success
2. Bits 23:16 of dw0 of the response are reserved for all responses
3. Check Bits 15:00 of dw0 for response length for all responses
4. Ensure that for the when DTYP is set to Controller Information (0x03)
5. Check that the response length = 12 + 32 = 44 bytes
6. BYTES 4:1 are reserved
7. Bits 7:1 of byte 5
8. BYTES 31:16 are reserved

Case 5: Verify OptCmds Data Length (M)

Test Procedure:

1. -Perform MCTP Initialization
2. -Perform Read NVMe-MI Data Structure with DTYP set to 0x04 (Optional Commands)
3. -Wait for Response Message

Observable Results:

1. Ensure that a proper response is returned with status set to Success
2. Bits 23:16 of dw0 of the response are reserved for all responses
3. Check Bits 15:00 of dw0 for response length for all responses
4. Ensure that for the when DTYP is set to optional Commands supported (0x04)
5. Check that the response length = 12 + 6 = 18 bytes (CMD0 and CMD1)
6. Bits 2:0 and bit 7 of BYTE 00 are reserved

Possible Problems: None

Test 8.8 – Management Endpoint Buffer Read – (FYI)

Purpose: The purpose of this test is to verify the proper implementation of the Management Endpoint Buffer Read command.

References:
NVMe-MI Specification Section 5.4

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 28, 2020

Discussion: 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.

Test Setup: See Appendix A.

Case 1: Management Endpoint Buffer Read DLEN>0 (FYI)

Test Procedure:
1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. If Management Endpoint Buffer Size is not 0h, send a Management Endpoint Buffer Read Command with the following parameters
    a. DOFST=0
    b. DLEN=1

Observable Results:
1. Verify the response length is 1 and indicates status success

Case 2: Management Endpoint Buffer Read DLEN=0 (FYI)

Test Procedure:
1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. If Management Endpoint Buffer Size is not 0h, send a Management Endpoint Buffer Read Command with the following parameters
    a. DOFST=0
    b. DLEN=0

Observable Results:
1. Verify no response data is sent, and response indicates status success

Case 3: DOFST > Management Endpoint Buffer Size (FYI)

Test Procedure:

1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. Read the NVMe Subsystem Information Data Structure to get the MJR & MNR version supported.
4. If Management Endpoint Buffer Size is not 0h, send a Management Endpoint Buffer Read Command with the following parameters
    a. DOFST>Management Endpoint Buffer Size
    b. DLEN=1

Observable Results:
1. Verify response indicates Invalid Parameter Error
2. Verify if the MJR=1 and MNR=2, then the PEL field indicates DLEN field

Case 4: (DOFST + DLEN) > Management Endpoint Buffer (FYI)

Test Procedure:
1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. Read the NVMe Subsystem Information Data Structure to get the MJR & MNR version supported.
4. If Management Endpoint Buffer Size is not 0h, send a Management Endpoint Buffer Read Command with the following parameters
    a. DOFST = Management Endpoint Buffer Size - 1
    b. DLEN=2

Observable Results:
1. Verify response indicates Invalid Parameter Error
2. Verify if the MJR=1 and MNR=2, then the PEL field indicates DLEN field

Case 5: Management Endpoint Buffer Read after Sanitize Operation(FYI)

Test Procedure:
1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. If Management Endpoint Buffer Size is not 0h, perform a Sanitize operation to clear the Management Endpoint Buffer.
4. Send a Management Endpoint Buffer Read Command with the following parameters
    a. DOFST = 0
    b. DLEN=1

Observable Results:
1. Verify response indicates Management Endpoint Buffer Cleared Due to Sanitize.

Possible Problems: None

Test 8.9 – Management Endpoint Buffer Write – (FYI)

Purpose: The purpose of this test is to verify the proper implementation of the Management Endpoint Buffer Write command.

References:
NVMe-MI Specification Section 5.5

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 28, 2020

Discussion: 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.

Test Setup: See Appendix A.

Case 1: Management Endpoint Buffer Write DLEN>0 (FYI)

Test Procedure:
1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. If Management Endpoint Buffer Size is not 0h, send a Management Endpoint Buffer Write Command with the following parameters
    a. DOFST=0
    b. DLEN=1

Observable Results:
1. Verify the response indicates status success

Case 2: Management Endpoint Buffer Write DLEN=0 (FYI)

Test Procedure:
1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. If Management Endpoint Buffer Size is not 0h, send a Management Endpoint Buffer Write Command with the following parameters
    a. DOFST=0
    b. DLEN=0

Observable Results:
1. Verify the response indicates status success

Case 3: DOFST > Management Endpoint Buffer Size (FYI)

Test Procedure:

1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. Read the NVMe Subsystem Information Data Structure to get the MJR & MNR version supported.
4. If Management Endpoint Buffer Size is not 0h, send a Management Endpoint Buffer Write Command with the following parameters
    a. DOFST>Management Endpoint Buffer Size
    b. DLEN=1

Observable Results:

1. Verify response indicates Invalid Parameter Error
2. Verify if the MJR=1 and MNR=2, then the PEL field indicates DLEN field

Case 4: (DOFST + DLEN) > Management Endpoint Buffer (FYI)

Test Procedure:

1. Perform Endpoint Discovery
2. Read the Port Information Data Structure of the Management Endpoint. Bytes 07:04 indicate 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, and this test is not applicable.
3. Read the NVMe Subsystem Information Data Structure to get the MJR & MNR version supported.
4. If Management Endpoint Buffer Size is not 0h, send a Management Endpoint Buffer Write Command with the following parameters
    a. DOFST = Management Endpoint Buffer Size - 1
    b. DLEN=2

Observable Results:

1. Verify response indicates Invalid Parameter Error
2. Verify if the MJR=1 and MNR=2, then the PEL field indicates DLEN field

Possible Problems: None

Test 8.10 – SES Read – (FYI)

Purpose: The purpose of this test is to verify the proper implementation of the SES Receive command.

References:
NVMe-MI Specification Section 5.9

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 28, 2020

Discussion: 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. This test is only applicable to NVMe Enclosures.

Test Setup: See Appendix A.

Case 1: SES Receive (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller Data Structure command. Check the NVMSR field. If NVMEE is set to 0, this test is not applicable.
3. Perform a SES Receive command with valid PCODE and ALENGTH fields.

Observable Results:
1. Verify the response indicates status success and includes the requested SES Diagnostic page.

Case 2: SES Receive with Reserved PCODE (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller Data Structure command. Check the NVMSR field. If NVMEE is set to 0, this test is not applicable.
3. Read the NVMe Subsystem Information Data Structure to get the MJR & MNR version supported.
4. Perform a SES Receive command with valid ALENGTH field and a reserved PCODE value.

Observable Results:
1. Verify the response indicates status Invalid Parameter Error.
2. Verify if the MJR=1 and MNR=2, then the PEL field indicates PCODE field

Case 3: SES Receive with SES Control type PCODE (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command. Check the NVMSR field. If NVMEE is set to 0, this test is not applicable.
3. Read the NVMe Subsystem Information Data Structure to get the MJR & MNR version supported.
4. Perform a SES Receive command with valid ALENGTH field and a PCODE value that corresponds to a SES Control type diagnostic page.

Observable Results:

1. Verify response indicates Invalid Parameter Error
2. Verify if the MJR=1 and MNR=2, then the PEL field indicates PCODE field

Case 4: SES Receive with Truncated Diagnostic Page (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command. Check the NVMSR field. If NVMEE is set to 0, this test is not applicable.
3. Perform a SES Receive command with a valid PCODE value and an ALENGTH value that is less than the expected length of the requested diagnostic page.

Observable Results:

1. Verify response length corresponds to the requested ALENGTH value.

Possible Problems: None

Test 8.11 – SES Send – (FYI)

Purpose: The purpose of this test is to verify the proper implementation of the SES Send command.

References:
NVMe-MI Specification Section 5.10

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 28, 2020

Discussion: 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.

Test Setup: See Appendix A.

Case 1: SES Send (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller Data Structure command. Check the NVMSR field. If NVMEE is set to 0, this test is not applicable.
3. Perform a SES Send command with a valid SES Diagnostic page and DLEN field.

Observable Results:
1. Verify the response indicates status success.

Case 2: SES Send with DLEN=0 (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller Data Structure command. Check the NVMSR field. If NVMEE is set to 0, this test is not applicable.
3. Perform a SES Send command with DLEN=0.

Observable Results:
1. Verify the response indicates status Success.

Possible Problems: None

Test 8.12 – VPD Read – (FYI)

Purpose: The purpose of this test is to verify the proper implementation of the VPD Read command.

References:
NVMe-MI Specification Section 5.11

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 28, 2020

Discussion: The VPD Read command is used to read the Vital Product Data described in section 9.2. Upon successful completion of the VPD Read command, the specified portion of the VPD contents is returned in the Response Data.

Test Setup: See Appendix A.

Case 1: VPD Read (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform a VPD Read command with a valid DOFST and non-zero DLEN fields.

Observable Results:
1. Verify the response indicates status success and VPD Response data is returned.

Case 2: VPD Read with DLEN=0 (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform a VPD Read command with a valid DOFST and DLEN=0.

Observable Results:
1. Verify the response indicates status success and no VPD Response data is returned.

Case 3: VPD Read with DLEN + DOFST > VPD Size (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Read the NVMe Subsystem Information Data Structure to get the MJR & MNR version supported.
3. Perform a VPD Read command with a DOFST + DLEN > VPD Size.

Observable Results:
2. Verify that the DUT does not return the VPD contents and responds with an Invalid Parameter Error Response
3. Verify if the MJR=1 and MNR=2, then the PEL field indicates DLEN field

Possible Problems: None

Test 8.13 – VPD Write – (FYI)

Purpose: The purpose of this test is to verify the proper implementation of the VPD Write command.

References:
NVMe-MI Specification Section 5.11

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 28, 2020

Discussion: The VPD Write command is used to update the Vital Product Data.

Test Setup: See Appendix A.

Case 1: VPD Write with DLEN=0 (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform a VPD Write command with a valid DOFST and DLEN=0.

Observable Results:

1. Verify the response indicates status success.

Case 2: VPD Write Data Check (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform a VPD Write command with a valid DOFST and non-zero DLEN.
3. Perform a VPD Read command to the same offset and length as the previous VPD Write.

Observable Results:

1. Verify the response to each command indicates status success.
2. Verify that the returned VPD Read data matched the data written in the VPD Write.

Case 3: VPD Write Cycle Information Check (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify command to retrieve the Identify Controller Data Structure.

Observable Results:

1. Verify if the VWCRV bit is set to 0, that the VWCR is also set to 0.

Possible Problems: None

Group 9: NVMe Admin Command Set Tests

Test 9.1 – NVMe Identify Command – (FYI)

Purpose: Check operation mandatory NVMe Admin Commands over NVMe-MI.

References:
NVMe-MI Specification 6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: June 15 ,2021

Discussion: 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. Certain NVM Express Admin Commands that are mandatory, optional, and prohibited for an NVMe Storage Device and an NVMe Enclosure using the out-of-band mechanism.

Test Setup: See Appendix A.

Case 1: Identify Command (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command. Record the NVMSR field.

Observable Results:
1. Verify that a response indicating status success is return along with the Identify Controller data structure.
2. Verify that if the DUT is an NVMe Storage Device, the NVMESD bit is set to 1.
3. Verify that if the DUT is an NVMe Enclosure, the NVMEE bit is set to 1.
4. Verify that the NVMSR field is not cleared to 0.
5. Verify that if the VPD Write Cycle Remaining Valid bit is cleared to ‘0’, then VPD Write Cycles Remaining field is cleared to a value of 0h.

Possible Problems: None

Test 9.2 – NVMe Get Log Command – (FYI)

Purpose: Check operation mandatory NVMe Admin Commands over NVMe-MI.

References:
NVMe-MI Specification 6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: June 15, 2021

Discussion: 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. Certain NVM Express Admin Commands that are mandatory, optional, and prohibited for an NVMe Storage Device and an NVMe Enclosure using the out-of-band mechanism.

Test Setup: See Appendix A.

Case 1: Get Log Page Command (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command. Check the NVMSR field. If NVMSD is set to 0, this test is not applicable.
3. Perform a Get Log Page command in-band over the NVMe Interface supported Log Pages.

Observable Results:
1. Verify that a response indicating status success is returned along with requested Log Page for each supported Log Page.

Case 2: Get Log Page Command, Retain Asynchronous Event bit cleared (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command. Check the NVMSR field. If NVMSD is set to 0, this test is not applicable.
3. Perform a Get Log Page command in-band over the NVMe Interface supported Log Pages, with the Retain Asynchronous Event bit cleared to 0.

Observable Results:
1. Verify that a response indicating status Invalid Field in Command for each supported Log Page.

Case 3: Get Log Page Command, Boot Partition (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Configure the NVMe host to read the CAP.BPS register field, if the value is set to zero this test is not applicable.
3. Perform a Get Log Page command Boot Partition, Log Identifier 15h, with the Boot Partition Identifier equal to the value of CAP.BPS.

Observable Results:

1. Verify that a response indicating the Boot Partition log page, that the Log Identifier is set to 15h and Active Boot Partition ID and Boot Partition Size field are displayed.

Possible Problems: None

Test 9.3 – NVMe Get / Set Feature Command – (FYI)

Purpose: Check operation mandatory NVMe Admin Commands over NVMe-MI.

References:
NVMe-MI Specification 6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: June 15, 2021

Discussion: 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. Certain NVM Express Admin Commands that are mandatory, optional, and prohibited for an NVMe Storage Device and an NVMe Enclosure using the out-of-band mechanism.

Test Setup: See Appendix A.

Case 1: Get Feature Command (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command in-band over the NVMe Interface to the NVMe Controller to determine all supported Features.
4. Perform a Set Feature command in-band over the NVMe Interface to the NVMe Controller to determine all supported Features.
5. Perform a Get Feature command out-of-band over the NVMe-MI Interface to the Management Endpoint for all supported Features as determined in the earlier steps.
6. Perform a Set Feature command out-of-band over the NVMe-MI Interface to the Management Endpoint for all supported Features as determined in the earlier steps.

Observable Results:
1. Verify that a response indicating status success is returned for each supported Set/Get Feature command.
2. Verify that all Features that were supported when requested in-band were also supported when requested out-of-band.

Case 2: Get Feature Command to Host Metadata FIDs GDHM = 1 (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FIDs 7Dh, 7Eh, 7Fh with the SEL field set to 011b and GDHM set to 1.

Observable Results:
1. Verify that the Saveable bit in Dword 0 is cleared to 0 for the corresponding completion queue entry for each Get Features Commands.
2. Verify that the Changeable bit in Dword 0 is set to 1 for the corresponding completion queue entry for each Get Features Commands.
3. Verify that if the generated vendor specific string's Metadata Element Descriptor does not exist for the Host Metadata Data Structure that contains the default value of the specified Host Metadata Feature value, then

the controller shall create the Metadata Element Descriptor in the Host Metadata Data Structure that contains the default value with the generated vendor specific string.

4. Verify that if the generated vendor specific string's Metadata Element Descriptor does exist for the Host Metadata Data Structure that contains the default value of the specified Host Metadata Feature value, then the controller shall replace the Metadata Element Descriptor with the generated vendor specific string.

Case 3: Get Feature Command to Host Metadata FIDs GDHM = 0 (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FIDs 7Dh, 7Eh, 7Fh with the SEL field set to 011b and GDHM set to 0.

Observable Results:

1. Verify that the Saveable bit in Dword 0 is cleared to 0 for the corresponding completion queue entry for each Get Features Commands.
2. Verify that the Changeable bit in Dword 0 is set to 1 for the corresponding completion queue entry for each Get Features Commands.
3. Verify that the device does not generate any vendor specific strings for the Element Types of the specified Host Metadata feature.

Case 4: Set Feature Command to EA = 00b Non Existent Element Type (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FIDs 7Eh, 7Fh each with the GDHM field set to 1b
4. Perform a Set Feature command for FIDs 7Eh, 7Fh each with the EA field set to 00b and the Element Type set to a value that does not exist in the specified Host Metadata Feature value. The Get Feature responses can be used to determine a non-existent Element Type.

Observable Results:

1. Verify that the Controller creates the descriptor in the specified Host Metadata Feature value with the value in the Host Metadata data structure.

Case 5: Set Feature Command to EA = 00b Element Type Exists (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FIDs 7Eh, 7Fh each with the GDHM field set to 1b
4. Perform a Set Feature command for FIDs 7Eh, 7Fh each with the EA field set to 00b and the Element Type set to a value that does exist in the specified Host Metadata Feature value. The Get Feature responses can be used to determine an existent Element Type.

Observable Results:

1. Verify that the Controller replaces the Metadata Element Descriptor with the value in the specified Host Metadata data structure.

Case 6: Set Feature Command to FID Enhanced Controller Metadata 7Dh, EA = 00b (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Set Feature command for FID 7Dh and the EA field set to 00b.

Observable Results:

1. Verify that the Controller aborts the Set Features command with status 'Invalid Field in Command'.

Case 7: Set Feature Command to FID Enhanced Controller Metadata 7Dh, EA = 01b (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FID 7Dh, 7Eh, 7Fh and the GDHM = 1.
4. Perform a Set Feature command for FID 7Dh, 7Eh, 7Fh and the EA field set to 01b for existing Metadata Element Descriptors.
5. Perform a Get Feature command for FID 7Dh, 7Eh, 7Fh and the GDHM = 1.
6. Perform a Set Feature command for FID 7Dh, 7Eh, 7Fh and the EA field set to 01b for non-existent Metadata Element Descriptors.
7. Perform a Get Feature command for FID 7Dh, 7Eh, 7Fh and the GDHM = 1.

Observable Results:

1. Verify that the Set Feature command completes with status success and the controller deletes all the specified Metadata Element descriptors when performed for existing Metadata Element Descriptors.
2. Verify that the Set Feature command completes with status success and the controller does not change or update any existing Metadata Element Descriptors.

Case 8: Set Feature Command to FID Enhanced Controller Metadata 7Dh, EA = 10b, Descriptor does not exist (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FID 7Dh and the GDHM = 1.
4. Perform a Set Feature command for FID 7Dh and the EA field set to 10b for Metadata Element Descriptors that do not exist.
5. Perform a Get Feature command for FID 7Dh and the GDHM = 1.

Observable Results:

1. Verify that the Set Feature command completes with status success and the controller created new Metadata Element Descriptors which were returned in response to the second Get Feature command.

Case 9: Set Feature Command to FID Enhanced Controller Metadata 7Dh, EA = 10b, Descriptor Exists (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FID 7Dh and the GDHM = 1.
4. Perform a Set Feature command for FID 7Dh and the EA field set to 10b for Metadata Element Descriptors that do exist.
5. Perform a Get Feature command for FID 7Dh and the GDHM = 1.

Observable Results:

1. Verify that the Set Feature command completes with status success and the controller added the specified Metadata Element to the Enhanced Controller Metadata Feature value and did not modify any existing Metadata Element Descriptors.

Case 10: Set Feature Command to FID 7Eh and 7FH, EA = 10b (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Set Feature command for FID 7Eh and the EA field set to 10b.
4. Perform a Set Feature command for FID 7Fh and the EA field set to 10b.

Observable Results:

1. Verify that each Set Feature command was aborted with status 'Invalid Field in Command'.

Case 11: Host Metadata Feature too Large (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FID 7Dh and the GDHM = 1.
4. Perform a Set Feature command for FID 7Dh and the EA field set to 10b for Metadata Element Descriptors that do exist.
5. Perform a Get Feature command for FID 7Dh and the GDHM = 1.
6. Repeat steps 4 and 5 until the Host Metadata Feature value exceeds 4 KiB.

Observable Results:

1. Verify that the Set Feature command which causes the Host Metadata Feature value to exceed 4 KiB completes with status status 'Invalid Field in Command'.

Case 12: Enhanced Controller Metadata Feature value after Reset (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FID 7Dh and the GDHM = 1.
4. Perform a Controller Level Reset.
5. Perform a Get Feature command for FID 7Dh and the GDHM = 1.

Observable Results:

1. Verify that the value for the Number of Metadata Element Descriptors of the Enhanced Controller Metadata Feature returned in response ot the second get Feature command is set to 0h.

Case 13: Get Feature Command to FID 7Dh, SEL= 011b (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FID 7Dh and the GDHM = 1, SEL=011b.

Observable Results:

1. Verify that the NS Specific bit in Dword 0 of the corresponding completion queue entry for the Get Feature command is cleared to '0'.

Case 14: Get Feature Command to FID 7Fh, SEL= 011b (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FID 7Fh and SEL=011b.

Observable Results:

1. Verify that the NS Specific bit in Dword 0 of the corresponding completion queue entry for the Get Feature command is set to '1'.

Case 15: Set Feature Command DLEN > 4096 (FYI)

Test Procedure:

1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. Perform a Get Feature command for FIDs 7Eh, 7Fh each with the GDHM field set to 1b
4. Perform a Set Feature command for FIDs 7Eh, 7Fh each with the EA field set to 00b and the Element Type set to a value that does not exist in the specified Host Metadata Feature value. The DLEN of the Set Feature should be set to a value greater than 4096. The Get Feature responses can be used to determine a non-existent Element Type.

Observable Results:

1. Verify that the Set Feature causes the Controller to respond with a status of 'Invalid Parameter Error Response' with the PEL field indicating the DLEN field.

Possible Problems: None

Test 9.4 – Admin Commands Prohibited Out of Band – (FYI)

Purpose: Check that NVMe Admin commands issued using the out of band mechanism are handled properly .

References:
NVMe-MI Specification 6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 28, 2020

Discussion: 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. NVM Express Admin Commands that are mandatory, optional, and prohibited for an NVMe Storage Device or NVMe Enclosure using the out-of-band mechanism are defined in the NVMe-MI specification.

Test Setup: See Appendix A.

Case 1: Admin Commands Prohibited out of Band – Storage Device (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command. Check the NVMSR field. If NVMSD is set to 0, this test is not applicable.
3. Perform the following commands using an out of band mechanism (the NVMe-MI transport)
    a. Abort Command
    b. Asynchronous Event Request
    c. Create I/O Completion Queue
    d. Create I/O Submission Queue
    e. Delete I/O Completion Queue
    f. Delete I/O Submission Queue
    g. Keep Alive
    h. NVMe-MI Send
    i. NVMe-MI Receive

Observable Results:
1. Verify that a response indicating status Invalid Parameter Error Response with Parameter Error Location pointing to the NVMe opcode is returned for each prohibited command,

Case 2: Admin Commands Prohibited out of Band – Enclosure (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command. Check the NVMSR field. If NVMEE is set to 0, this test is not applicable.
3. Perform the following commands using an out of band mechanism (the NVMe-MI transport)
    a. Abort Command
    b. Asynchronous Event Request
    c. Create I/O Completion Queue
    d. Create I/O Submission Queue
    e. Delete I/O Completion Queue

f. Delete I/O Submission Queue
g. Keep Alive
h. NVMe-MI Send
i. NVMe-MI Receive
j. Format NVM

Observable Results:

1. Verify that a response indicating status Invalid Parameter Error Response with Parameter Error Location pointing to the NVMe opcode is returned for each prohibited command,

Possible Problems: None

Test 9.5 – Sanitize Command – (FYI)

Purpose: Check operation of NVMe-MI commands during a the Sanitize operation.

References:
NVMe-MI Specification 6

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 28, 2020

Discussion: 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. Certain NVM Express Admin Commands that are mandatory, optional, and prohibited for an NVMe Storage Device and an NVMe Enclosure using the out-of-band mechanism.

Test Setup: See Appendix A.

Case 1: NVMe-MI Commands During Sanitize (FYI)

Test Procedure:
1. Perform Endpoint Discovery.
2. Perform an Identify Controller command.
3. On the NVMe Interface, initiate a Sanitize operation using supported parameters.
4. While the Sanitize Operation is being performed, perform each of the following Management Interface commands (if supported) on the NVMe-MI Interface:
    a. Configuration Get
    b. Configuration Set
    c. Controller Health Status Poll
    d. Management Endpoint Buffer Read
    e. Management Endpoint Buffer Write
    f. NVM Subsystem Health Status Poll
    g. Read NVMe-MI Data Structure
    h. Reset
    i. SES Receive
    j. SES Send
    k. VPD Read
    l. VPD Write

Observable Results:
1. Verify that each of the NVMe-MI commands supported by the DUT completed successfully.

Possible Problems: None

Test 9.6 – Format NVM, More Processing Time Required – (FYI)

Purpose: To perform a Format NVM command, possibly get a More Processing required Time Response.

References:
NVMe-MI Specification 4.1.2.3 & 4.2.2.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: December 8, 2022

Discussion: A Management Endpoint should only return a More Processing Required Response for a Command Message that are expected to take longer than the required time (e.g. Format NVM).

Test Setup: See Appendix A.

Case 1: NVMe-MI Format NVM, More Processing Required (FYI)

Test Procedure:
1. Perform Endpoint Discovery
2. Perform an Identify Controller command.
3. On the NVMe Interface, initiate a NVM Format operation using supported parameters.

Observable Results:
1. Verify that the Format NVM command completed successfully.
2. If a More Processing Required Time Response is received wait the amount of time in the More Processing Required Time (MPRT) field unless it is unusually large.

Possible Problems: None

Group 10: Management Enhancement Tests

Test 10.1 – NVMe-MI Identify Structure ME Capabilities – (In Progress)

Purpose: New Identify Controller structure fields (ME Capabilities)

References:
TBD

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
TBD

Observable Results:
TBD

Possible Problems: None

Test 10.2 – NVMe-MI Identify Capabilities – (In Progress)

Purpose: In NVMe Identify Controller data structure bytes 254:240 and byte 255 bits 7:2 are reserved. Bits 0 and 1 in byte 255 identify presence of the Management Endpoint on the port (SMBus/I2c and PCIe) – should probably use this information from the start to decide what can be tested for the port.

References:
TBD

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: No further discussion available at time of writing

Test Setup: See Appendix A.

Test Procedure:
TBD

Observable Results:
TBD

Possible Problems: None

Test 10.3 – NVMe-MI Namespace Metadata – (FYI)

Purpose: Verify proper handling of Namespace Metadata Elements.

References:
NVMe-MI Specification Section 8.2.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 24, 2020

Discussion: This feature is used to store metadata about a namespace associated with a Controller in the NVM Subsystem for later retrieval. The values stored in the Namespace Metadata Feature do not modify Controller behavior on the namespace. This feature is namespace specific.

If a Set Features command is submitted for this Feature, a Host Metadata data structure is transferred in the data buffer for the command. The Host Metadata data structure is 4 KiB in size and contains zero or more Metadata Element Descriptors. If host software attempts to add or update a Metadata Element Descriptor that causes the stored Host Metadata data structure to grow larger than 4 KiB, the Controller shall abort the command with an Invalid Parameter Error Response. The Host Metadata structure for this feature is independent of the Host Metadata data structure for the Controller Metadata feature.

Test Setup: See Appendix A.

Case 1: Perform Set/Get Feature for Namespace Metadata (FYI)

Test Procedure:

1. Perform MCTP initialization.
2. Perform a Get Feature Operation for Feature Value 7Fh Namespace Metadata. Wait for the Get Feature Response.
3. Perform a Set Feature Operation for Feature Value 7Fh Namespace Metadata providing a new supported value. Wait for the Get Feature Response.
4. Perform a Get Feature Operation for Feature Value 7Fh Namespace Metadata. Wait for the Get Feature Response.

Observable Results:

1. Verify that for each request the proper response is returned, and status is set to Success.
2. Verify that the new values provided in the Set Feature operation are returned in the Get Feature operation.

Case 2: Host Namespace Metadata Data Structure Too Large (FYI)

Test Procedure:

1. Perform MCTP initialization.
2. Perform a Get Feature Operation for Feature Value 7Fh Namespace Metadata. Wait for the Get Feature Response.
3. Perform a Set Feature Operation for Feature Value 7Fh Namespace Metadata that adds or update a Metadata Element that causes the stored Host Metadata data structure to grow larger than 4096 bytes.
4. Perform a Get Feature Operation for Feature Value 7Fh Namespace Metadata. Wait for the Get Feature Response.

Observable Results:

1. Verify that for the initial Get Feature request the proper response is returned, and status is set to Success.
2. Verify that for the Set Feature request the proper response is returned, and status is set to Invalid Parameter.
3. Verify that the new values provided in the Set Feature operation are not returned in the Get Feature operation.

Possible Problems: None

Test 10.4 – NVMe-MI Controller Metadata – (FYI)

Purpose: Verify proper handling of Controller Metadata Elements.

References:
NVMe-MI Specification Section 8.2.1

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 20 2016

Discussion: This feature is used to store metadata about the host platform in an NVM Subsystem for later retrieval. The values stored in the Controller Metadata Feature do not modify Controller behavior.

If a Set Features command is submitted for this Feature, a Host Metadata data structure is transferred in the data buffer for the command. The Host Metadata data structure is 4 KiB in size and contains zero or more Metadata Element Descriptors. If host software attempts to add or update a Metadata Element that causes the stored Host Metadata data structure to grow larger than 4 KiB, the Controller shall abort the command with an Invalid Parameter Error Response. The Host Metadata Data Structure for this feature is independent of the Host Metadata data structure for the Namespace Metadata feature

Test Setup: See Appendix A.

Case 1: Perform Set/Get Feature for Controller Metadata (FYI)

Test Procedure:
1. Perform MCTP initialization.
2. Perform a Get Feature Operation for Feature Value 7EFh Controller Metadata. Wait for the Get Feature Response.
3. Perform a Set Feature Operation for Feature Value 7Eh Controller Metadata providing a new supported value. Wait for the Get Feature Response.
4. Perform a Get Feature Operation for Feature Value 7Eh Controller Metadata. Wait for the Get Feature Response.

Observable Results:
1. Verify that for each request the proper response is returned, and status is set to Success.
2. Verify that the new values provided in the Set Feature operation are returned in the Get Feature operation.

Case 2: Host Controller Metadata Data Structure Too Large (FYI)

Test Procedure:
1. Perform MCTP initialization.
2. Perform a Get Feature Operation for Feature Value 7Eh Controller Metadata. Wait for the Get Feature Response.
3. Perform a Set Feature Operation for Feature Value 7Eh Controller Metadata that adds or update a Metadata Element that causes the stored Host Metadata data structure to grow larger than 4096 bytes.
4. Perform a Get Feature Operation for Feature Value 7Eh Controller Metadata. Wait for the Get Feature Response.

Observable Results:
1. Verify that for the initial Get Feature request the proper response is returned, and status is set to Success.
2. Verify that for the Set Feature request the proper response is returned, and status is set to Invalid Parameter.

3. Verify that the new values provided in the Set Feature operation are not returned in the Get Feature operation.

Possible Problems: None**.**

Group 11: Vital Product Data Tests

Test 11.1 – VPD Read Default Values – (FYI)

Purpose: Verify that the DUT sets NVMe-MI VPD default values properly.

References:
NVMe-MI Specification Section 9.2.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 23, 2020

Discussion: 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 1.8 of the NVMe-MI specificaiton. The Product Info Area factory default values shall be set to the values defined below.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization
2. Verify that the device claims support for NVMe-MI v1.1 and beyond, if not, this test is not applicable.
3. Perform VPD Read
4. Wait for Response

Observable Results:
1. Ensure that VPD Read returned with Success.
5. Ensure that the Following values were reserved in the Response.
    a. Common Header:
        i. Byte 01 = 0x01 (IPMIVER)
        ii. Byte 04 = 0x01 (PIAOFF)
        iii. Byte 05 = 0x0F (MRIOFF)
        iv. Byte 06 = 0x00 (Reserved)
    b. Product Info Area:
        i. IPMIVER = 01h
        ii. LCODE = 19h
        1. EOR = C1h
        iii. CPIA field is preceded by a Type/Length field
        iv. Padding of 0h to make the area size a multiple of 8 bytes.

Possible Problems: None.

Test 11.2 – Topology Multirecord Area – (FYI)

Purpose: Verify that the DUT sets Topology Multirecord values properly.

References:
NVMe-MI Specification Section 9.2.5

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 23, 2020

Discussion: This MultiRecord describes an NVMe Storage Device's architectural elements and their connections. It is required on all NVMe Storage Device FRUs.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.
2. Verify that the device claims support for NVMe-MI v1.1 and beyond, if not, this test is not applicable.
3. Perform VPD Read requesting the Topology Multirecord Area.
4. Wait for Response

Observable Results:
1. Verify that that VPD Read returned with Success.
2. Verify that Topology Record Type ID = 0Dh
3. Verify that Record Format = 02h, or 82h for the last record in list.
4. Verify that the RLEN field is correct.
5. Verify that Record and Header Checksum are correct
6. Verify that Version Number = 0h
7. Verify that Element Count is correct and not set to 0.

Possible Problems: None

Test 11.3 – NVMe Multirecord Area – (FYI)

Purpose: Verify that the DUT sets NVMe Multirecord values properly.

References:
NVMe-MI Specification Section 9.2.3

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 23, 2020

Discussion: The NVMe MultiRecord is used to describe the form factor, power requirements, and capacity of NVMe Storage Devices with a single NVM Subsystem.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.
2. Verify that the device claims support for NVMe-MI v1.1 and beyond, if not, this test is not applicable.
3. Perform VPD Read requesting the NVMe Multirecord Area.
4. Wait for Response

Observable Results:
1. Verify that that VPD Read returned with Success.
2. Verify that NVMe Record Type ID = 0Bh
3. Verify that Record Format = 02h, or 82h for the last record in list.
4. Verify that the RLEN field =3Bh
5. Verify that Record and Header Checksum are correct
6. Verify that NVMe Multirecord Area Version Number = 0h

Possible Problems: None

Test 11.4 – NVMe PCIe Port Area – (M)

Purpose: Verify that the DUT sets NVMe PCIe Port Multirecord values properly.

References:
NVMe-MI Specification Section 9.2.4

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: April 23, 2020

Discussion: The NVMe PCIe Port MultiRecord is used to describe the PCIe connectivity for NVMe Storage Devices with a single NVM Subsystem.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.
2. Verify that the device claims support for NVMe-MI v1.1 and beyond, if not, this test is not applicable.
3. Perform VPD Read requesting the NVMe PCIe Port Multirecord Area.
4. Wait for Response

Observable Results:
1. Verify that that VPD Read returned with Success.
2. Verify that NVMe Record Type ID = 0Ch
3. Verify that Record Format = 02h, or 82h for the last record in list.
4. Verify that the RLEN field =0Bh
5. Verify that Record and Header Checksum are correct
6. Verify that NVMe PCIe Port Multirecord Area Version Number = 1h

Possible Problems: None.

Test 11.5 – FRU Information Device Read via VPD Read – (M)

Purpose: Verify that the VPD properly contains the FRU Information Device.

References:
NVMe-MI Specification Section 9.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: June 15, 2021

Discussion: 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 4096 bytes which contains the VPD.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.
2. Verify that the device claims support for NVMe-MI v1.1 and beyond, if not, this test is not applicable.
3. Perform VPD Read requesting the FRU Information Device.
4. Wait for Response

Observable Results:
1. Verify that the VPD Read returned with Success and the FRU Information Device is between 256 to 4096 bytes.
2. Verify if the DUT contains an SMBus/I2C Port, that the FRU Information Device Element Descriptor contains a valid SMBus/I2C Address Info field.
3. Verify if the DUT does not contain an SMBus/I2C Port, that the FRU Information Device Element Descriptor SMBus/I2C Address Info field is cleared to 0h.

Possible Problems: None

Test 11.6 – FRU Information Device Read via I2C Read – (M)

Purpose: Verify that the VPD properly contains the FRU Information Device.

References:
NVMe-MI Specification Section 9.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: June 15, 2021

Discussion: 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 4096 bytes which contains the VPD.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.
2. Verify that the device claims support for NVMe-MI v1.1 and beyond, if not, this test is not applicable.
3. Perform I2C Read requesting the FRU Information Device.
4. Wait for Response

Observable Results:
1. Verify that the I2C Read returned with Success and the FRU Information Device is between 256 to 4096 bytes.
2. Verify that the FRU Information Device Element Descriptor contains a valid SMBus/I2C Address field.

Possible Problems: None

Test 11.7 – FRU Information Device Update – (M)

Purpose: Verify that the VPD can properly update the FRU Information Device.

References:
NVMe-MI Specification Section 9.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: June 15, 2021

Discussion: Updating the VPD by writing to the FRU Information Device using I2C Writes shall not be supported if the VPD Write command is supported.

Test Setup: See Appendix A.

Test Procedure:
1. Determine if the DUT supports the VPD Write command. If the DUT does not support the VPD Write command this test is not applicable.
2. Verify that the device claims support for NVMe-MI v1.1 and beyond, if not, this test is not applicable.
3. Perform VPD Read requesting the FRU Information Device.
4. Wait for Response
5. Perform VPD Write updating the FRU Information Device.
6. Perform VPD Read requesting the FRU Information Device to verify that the update was successful.
7. Perform an I2C/SMBus Write to update the FRU Information Device.

Observable Results:
1. Verify that the VPD Write operation was successful.
2. Verify that the I2C/SMBus Write to update the FRU Information Device was not successful.

Possible Problems: None

Test 11.8 – FRU Information Device Internal Offset – (M)

Purpose: Verify that the DUT properly contains the FRU Information Device internal offset.

References:
NVMe-MI Specification Section 9.2

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: June 15, 2021

Discussion: 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 following a power cycle of the FRU Information Device. Implementations are allowed to maintain the current internal offset value or clear it to 0h following a reset of the FRU Information Device.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP Initialization.
2. Verify that the device claims support for NVMe-MI v1.1 and beyond, if not, this test is not applicable.
3. Perform I2C Read requesting the FRU Information Device.
4. Wait for Response
5. Perform I2C Read requesting the FRU Information Device.
6. Wait for Response
7. Continue performing I2C reads until the last byte of the FRU Information Device is read. Perform 1 additional I2C read.
8. Power cycle the device, and allow it to restart and return to the Idle state.
9. Perform I2C Read requesting the FRU Information Device.
10. Wait for Response

Observable Results:
1. Verify that each I2C Read returned with Success and after each Read the internal offset was incremented by 1h.
2. Verify that the internal offset rolls over to 0h when the last FRU Information Device byte is read.
3. Verify that the I2C read after the power cycle returned data from offset 0h.

Possible Problems: None

Group 12: Management Endpoint Reset Tests

Test 12.1 – NVMe-MI PCIe Endpoint Reset – (M)

Purpose: Verify that Management Endpoint resets are properly isolated.

References:

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: May 27, 2021

Discussion: A reset of a Management Endpoint in an NVM Subsystem shall not affect any other Management Endpoint in the NVM Subsystem or any other NVM Subsystem entity. Note that for implementations compliant to NVMe-MI version 1.1 and earlier, during a PCI Express conventional reset of a PCIe Management Endpoint, MCTP accesses may not be supported on other PCIe or SMBus/I2C Management Endpoints in the NVM Subsystem.

This test is only applicable to products with multiple Management Endpoints within the NVM Subsystem.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP accesses to all management endpoints in the NVM Subsystem.
2. Perform a PCIe Convention Reset on a single PCIe port

Observable Results:
1. Verify that MCTP accesses on any management endpoints not associated with the reset PCIe port completed without error.

Possible Problems: None

Test 12.2 – NVMe-MI SMBus Endpoint Reset – (M)

Purpose: Verify that Management Endpoint resets are properly isolated.

References:

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: June 15, 2021

Discussion: An SMBus Reset shall be treated by each Command Slot in the SMBus/I2C Management Endpoint as if an implicit Abort Control Primitive was received with the exception that the Management Endpoint does not transmit the Abort Control Primitive Response Messages.

This test is only applicable to products with multiple Management Endpoints within the NVM Subsystem.

Test Setup: See Appendix A.

Test Procedure:
1. Perform MCTP accesses to all management endpoints in the NVM Subsystem.
2. Perform a SMBus Reset on a single SMBus port on the Management Endpoint with the Command Slot in each of the following states:

a. Idle
b. Receive
c. Process
d. Transmit

Observable Results:

1. Verify that MCTP accesses on any management endpoints not associated with the reset Command Slot completed without error.
2. Verify that in each case, the Command Slot entered the Idle state after the SMBus reset was executed.
3. Verify that the DUT does not transmit the Abort Control Primitive Response.

Possible Problems: None

Test 12.3 – NVMe-MI SMBus/I2C Reset, bits and fields – (FYI)

Purpose: Verify that Management Endpoint resets are properly handled and that the reset will reset associated bits and fields.

References:

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: Dec 8, 2022

Discussion: An SMBus Reset shall be treated by each Command Slot in the SMBus/I2C Management Endpoint as if an implicit Abort Control Primitive was received with the exception that the Management Endpoint does not transmit the Abort Control Primitive Response Messages. An SMBus Reset shall cause a Management Endpoint Reset of the SMBus/I2C Management Endpoint. A Management Endpoint Reset: resets bits and fields dedicated to the out-of-band mechanism.

Test Setup: See Appendix A.

Test Procedure:
1. This test is only applicable to a device supporting NVMe-MI 1.2 or greater, if not skip this test
2. Perform MCTP accesses to a management endpoint in the NVM Subsystem.
3. Perform a Controller Health Status Poll Command on the Controller Health Data Structure (CHDS) with the following parameters: Report all set, Include PCI Functions set, Controller Status Changes set, MAXRENT set to 0x01, for the NVMe controller.
4. Perform a Controller Health Status Poll on the Controller Health Status Changed Flags for the NVMe controller.
5. Perform a Subsystem Health Status Poll on the NVM Subsystem Health Data structure (NSHDS).
6. Perform a Configuration Get with Configuration Identifier set to 0x01 (SMBus/I2C Frequency).
7. Wait for Response message
8. Perform a SMBus Reset on the controller in the CHDS response.
9. Perform the same Controller Health Status Poll Commands and NVM Subsystem Health Status Poll Commands as in steps 3&4
10. Perform the same NVM Subsystem Health Status Poll as step 5.
11. Perform a Configuration Get with Configuration Identifier set to 0x01 (SMBus/I2C Frequency).

Observable Results:
1. Verify that MCTP access on any management endpoints not associated with the reset Command Slot completed without error.
2. Verify that all Controller and Subsystem Health Status Polls complete successfully
3. Verify that all fields and bits in the Out-of-band mechanism are reset.
4. Verify that the Configuration Get Commands complete successfully and that the SMBus/I2C Frequency is set to 1h (100kHz).

Possible Problems: None

Test 12.4 – NVMe-MI Controller Level Reset – (FYI)

Purpose: Verify that Controller Level resets are properly handled and that the reset will reset associated bits and fields.

References:

Resource Requirements:
Tools capable of stimulus and decoding of traffic on the MCTP and NVMe-MI interface.

Last Modification: Dec 8, 2022

Discussion: A Controller Level Reset : resets bits and fields dedicated to the in-band tunneling mechanisms.

Test Setup: See Appendix A.

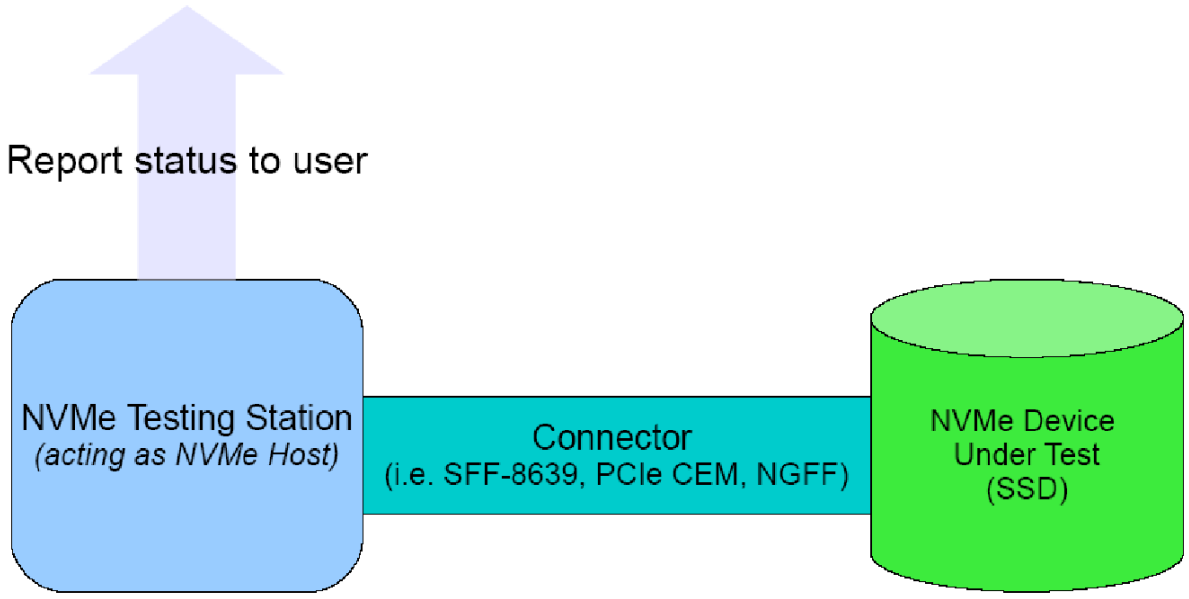
Test Procedure:
1. This test is only applicable to a device supporting NVMe-MI 1.2 or greater, if not skip this test
2. Perform MCTP accesses to a management endpoint in the NVM Subsystem.
3. Perform a Controller Health Status Poll Command on the Controller Health Data Structure (CHDS) with the following parameters: Report all set, Include PCI Functions set, Controller Status Changes set, MAXRENT set to 0x01, for the NVMe controller.
4. Perform a Controller Health Status Poll on the Controller Health Status Changed Flags for the NVMe controller.
5. Perform a Subsystem Health Status Poll on the NVM Subsystem Health Data structure (NSHDS).
6. Perform a Controller Level Reset on the controller in the CHDS response.
7. Perform the same Controller Health Status Poll Commands and NVM Subsystem Health Status Poll Commands as in steps 3&4
8. Perform the same NVM Subsystem Health Status Poll as step 5.

Observable Results:
1. Verify that MCTP access on any management endpoints not associated with the reset Command Slot completed without error.
2. Verify that all Controller and Subsystem Health Status Polls complete successfully
3. Verify that all fields and bits in the in-band tunneling mechanisms are reset.

Possible Problems: None

Appendix A: Default Test Setup

Except where otherwise specified, all tests will require the DUT to have one of the following default physical configuration at the beginning of each test case:



Appendix B: Notes on Test Procedures

There are scenarios where in test procedures it is desirable to leave certain aspects of the testing procedure as general as possible. In these cases, the steps in the described test procedure may use placeholder values, or may intentionally use non–specific terminology, and the final determination of interpretation or choice of values is left to the discretion of the test technician. The following is an attempt to capture and describe all such instances used throughout the procedures.

Ports on Testing Station and Device Under Test

In general, any PCIe Port on the Testing Station or Device Under Test may be used as an interface with a test station or interoperability partner. There is *assumed* to be no difference in behavior, with respect to the protocols involved in this test suite, between any two PCIe ports on the Testing Station or Device Under Test. Hence, actual ports used may be chosen for convenience. However, it is recommended that the PCIe port used in the test configuration is recorded by the test technician.

Use of "various"

To maintain generality, some steps will specify that "various other values" (or the like) should be used in place of a given parameter. Ideally, all possible values would be tested in this case. However, limits on available time may constrain the ability of the test technician to attempt this. Given this, a subset of the set of applicable values must generally be used.
When deciding how many values should be used, it should be noted that the more values that are tested, the greater the confidence of the results obtained (although there is a diminishing return on this).
When deciding which specific values to use, it is generally recommended to choose them at pseudo–randomly yet deterministically. However, if there exist subsets of the applicable values with special significance, values from each subset should be attempted.

Appendix C: NVMe Integrators List Requirements

Purpose: To provide guidance on what tests are required for NVMe Integrators List Qualification for products that advertise support for NVMe-MI on the NVMe Integrators List.

References:
UNH-IOL NVMe Integrators List Policy Document
NVMe-MI Conformance Test Specification

Resource Requirements:
NVMe-MI Device.

Last Modification: April 29, 2019

Discussion: The table below outlines what tests are to be considered Mandatory, FYI, or In Progress for devices being qualified for the NVMe Integrators List that advertise support for NVMe-MI on the NVMe Integrators List. Further descriptions of each test can be found in the NVMe-MI Conformance Test Specification.

MCTP Base Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 1.1 – MCTP Endpoint ID	FYI
Test 1.1.1 – MCTP Endpoint ID – Reserved Bits	FYI
Test 1.2 – MCTP Packet Sequence Number	FYI
Test 1.3 – MCTP Multiple Packets	FYI
Test 1.4.1 – MCTP Bad Packet 1	FYI
Test 1.4.1 – MCTP Bad Packet 2	FYI
Test 1.4.1 – MCTP Bad Packet 3	FYI
Test 1.4.1 – MCTP Bad Packet 4	FYI
Test 1.4.1 – MCTP Bad Packet 5	FYI

MCTP Control Message Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 2.1 – MCTP Control Instance ID	FYI

MCTP Commands

Test Name and Number	Mandatory, FYI, In Progress?
Test 3.1 – MCTP Command Set Endpoint ID	FYI
Test 3.2 – MCTP Command Get MCTP Version	FYI
Test 3.3 – MCTP Command Get Message Type	FYI

Test 3.4 – MCTP Command Prepare for Endpoint Discovery	FYI
Test 3.5 – MCTP Command Endpoint Discovery	FYI
Test 3.6 – MCTP Command Get Endpoint ID	FYI

NVMe Error Handling

Test Name and Number	Mandatory, FYI, In Progress?
Test 4.1 - NVMe-MI Invalid Opcode	Mandatory
Test 4.2 - NVMe-MI Reserved Identifier	Mandatory
Test 4.3 - NVMe-MI Health Status Change	Mandatory
Test 4.4 - NVMe-MI Reserved Identifier 2	Mandatory
Test 4.5 - NVMe-MI MAXRENT Error	Mandatory
Test 4.6 - NVMe-MI Reserved Data Structure Type	Mandatory
Test 4.7 - NVMe-MI Invalid VPD Read Size	FYI
Test 4.8 - NVMe-MI Invalid VPD Write Status	Mandatory
Test 4.9 - NVMe-MI Invalid Parameter Status	Mandatory
Test 4.10 - NVMe-MI Invalid Command Size	FYI

NVMe Management Interface Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 5.1 – NVMe-MI Message Type	Mandatory
Test 5.2 – NVMe-MI Message IC	Mandatory
Test 5.3 – NVMe-MI CRC Check	FYI
Test 5.4 – NVMe-MI Command Slot	Mandatory
Test 5.5 – NVME-MI Request Data size mismatch	Mandatory
Test 5.6 – NVMe-MI MCTP packet padding	Mandatory
Test 5.7 – NVMe-MI Message Integrity Check	Mandatory

NVMe-MI Message Processing Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 6.1 – NVMe-MI Reserved Fields	Mandatory
Test 6.2 – NVMe-MI Error Response Code	Mandatory

Control Primitives Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 7.1 – NVMe-MI Response Tag	Mandatory
Test 7.2 – NVMe-MI Response Message	Mandatory

Test 7.3 – NVMe-MI GetState Primitive Response	FYI
Test 7.4 – NVMe-MI Response Message Replay	Mandatory

Management Interface Commands

Test Name and Number	Mandatory, FYI, In Progress?
Test 8.1 – NVMe-MI Response Header	Mandatory
Test 8.2 – NVMe-MI Configuration Set	Mandatory
Test 8.3 – NVMe-MI Config Get Response	Mandatory
Test 8.4 – NVMe-MI Health Status Poll	Mandatory
Test 8.5 – NVMe-MI Controller Health Status Poll	Mandatory
Test 8.6 – NVMe-MI Read Data Structure	Mandatory
Test 8.7.1 – NVMe-MI NVMSSI Data Length	Mandatory
Test 8.7.2 – NVMe-MI PortInfo Data Length	Mandatory
Test 8.7.3 – NVMe-MI CtrlrList Data Length	Mandatory
Test 8.7.4 – NVMe-MI CtrlrInfo Data Length	Mandatory
Test 8.7.5 – NVMe-MI OptCmds Data Length	Mandatory

NVMe Admin Commands Set Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 9.1 – NVMe-MI Mandatory Admin Commands	In Progress

Management Enhancement Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 10.1 – NVMe-MI Identify Structure ME Capabilities	In Progress
Test 10.2 – NVMe-MI Identify Capabilities	In Progress
Test 10.3 – NVMe-MI Namespace Metadata	In Progress

Vital Product Data Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 11.1 – NVMe-MI VPD Default Values	In Progress

Management Endpoint Reset Tests

Test Name and Number	Mandatory, FYI, In Progress?
Test 12.1 – NVMe-MI PCIe Reset	In Progress

Appendix D: Test Tools

The Tests described in this document can be performed using available Teledyne-LeCroy Test Tools. These tests are supported in v8.77 or higher of the Teledyne-LeCroy PC Edition software available at:
http://teledynelecroy.com/support/softwaredownload/documents.aspx?standardid=18