SANBlaZe

White Paper

Testing NVMe-MI Over PCIe VDM

Verification of NVMe Management Interface (MI) Specification Over PCIe Vendor Defined Messages (VDM) Transport

Author:

Vince Asbridge
Founder and President, SANBlaze

Table of Contents

Introduction

SANBlaze introduces the ability to test Management Interface (MI) commands over the PCIe Vendor Defined Message Physical Layer (VDM) using Management Component Transport Protocol (MCTP) over PCIe Binding.

The Non-Volatile Memory Express specification defined by the NVM Express organization (https://nvmexpress.org/) defines a register level interface that allows in-band host software to communicate with an NVMe Subsystem.

The NVMe specification further defines several mechanisms to manage NVMe Storage Devices or NVMe Enclosures. One such mechanism, NVMe-MI (Management Interface) allows an application to communicate out-of-band with an NVMe Storage Device. Details of the MI specification are available from nvmexpress.org at the following link (NVM-Express-Management-Interface).

SANBlaze SBExpress-RM4 and SBExpress-DT4 bring the ability to test MI over VDM to the existing capabilities of the SANBlaze NVMe test system, now allowing MI to be tested in-band on PCIe with NVMe-MI Send and Receive, and out of band using MI over SMBus or PCIe VDM, rounding out a complete test system for NVMe-MI.

This white paper introduces the SANBlaze SBExpress VDM capability and provides a starting point for your own MI over VDM testing using the SBExpress system.

Transports for NVMe-MI

The NVMe-MI specification defines an architecture and command set for out-of-band and in-band management of an NVMe Storage Device.

NVMe-MI defines key capabilities for NVMe Storage Devices:

- Discover NVMe Storage Devices that are present and learn capabilities of each NVMe Storage Device
- Store data about the host environment enabling a Management Controller or other entity to query the data later
- Health and temperature monitoring
- Multiple concurrent commands to prevent a long latency command from blocking monitoring operations
- Out-of-band mechanism is host processor and operating system agnostic
- A standard format for VPD and defined mechanisms to read/write VPD contents
- Preserves data-at-rest security

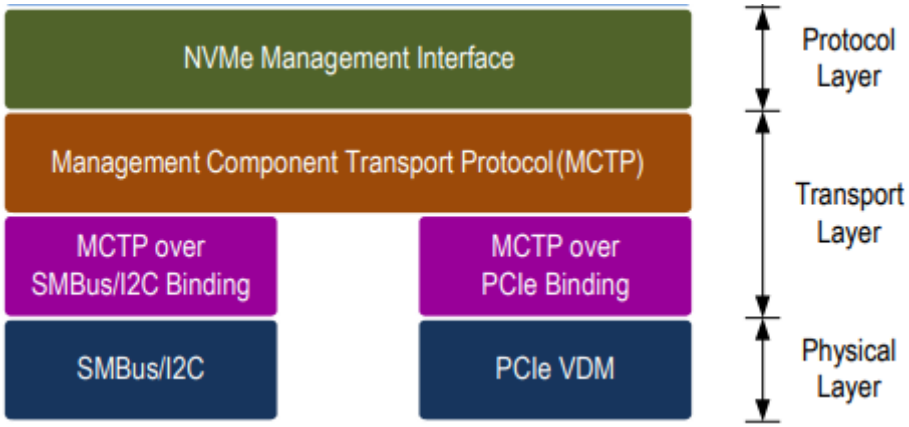

Figure 1: NVME-MI Transport Layers

This paper focuses on the verification of NVMe-MI over PCIe using the VDM Physical Layer.

MCTP over PCIe

A definition of MCTP over PCIe is available from *dmtf.org* in the following document (DSP0238_1.1.0.pdf).

The following figure describes the PCIe VDM Header and Data Formats.

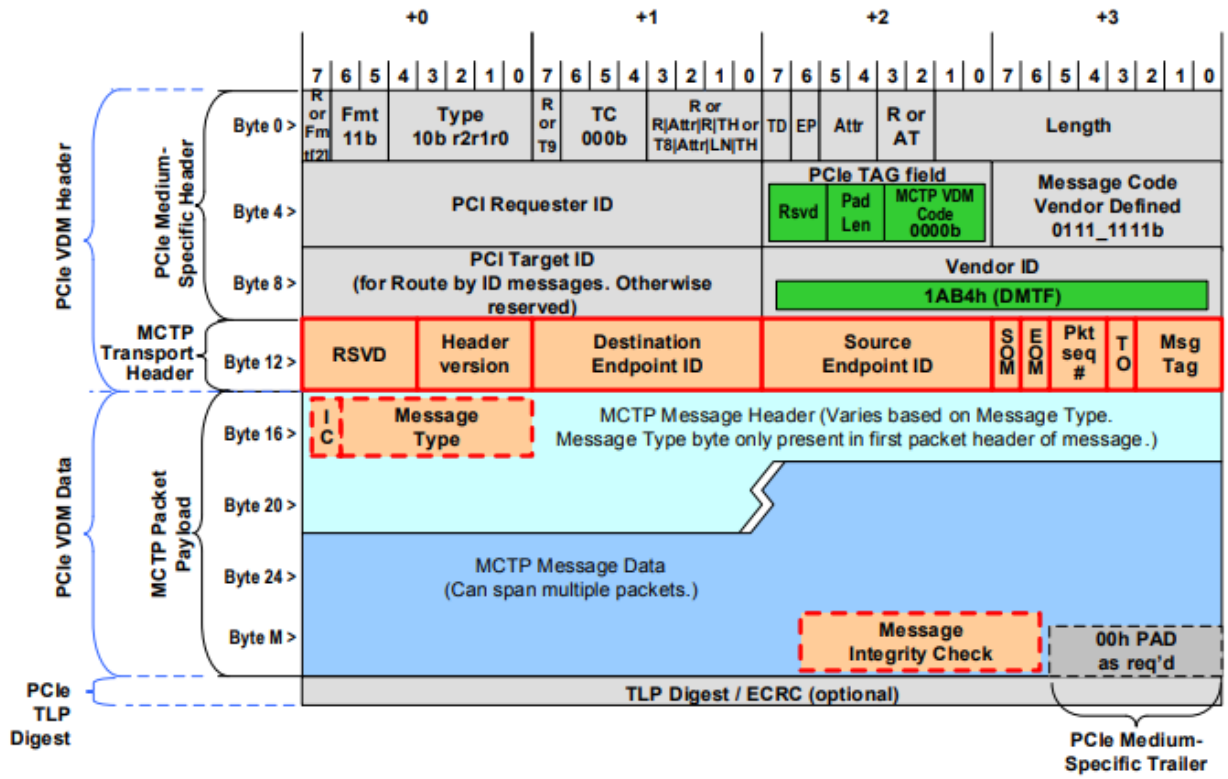

Figure 2: MCTP over PCI Express Vendor Defined Message (VDM) packet format

Hardware Setup

In order to achieve testing of NVMe-MI over PCIe VDM, a hardware configuration capable of sending and receiving PCIe Vendor Defined Messages (VDM) with and without data is required.

SANBlaze offers a VDM controller which, when coupled with an SBExpress-RM4 or SBExpress-DT4 system, provides a mechanism for transporting PCIe VDM messages generated by the mi program to the device under test and receiving the reply from the device under test.

The SANBlaze VDM controller is compatible with all existing Gen4 SBExpress systems, and is available at time of purchase or as a field installable add-on to your current Gen4 SBExpress system.

Once installed, the SBExpress system will be capable of NVMe-MI over PCIe VDM. The hardware and software are seamlessly integrated into the "Certified by SANBlaze" program as described below.

Software Setup

NVMe-MI over PCIe VDM is a licensed feature of the SBExpress NVMe test system and can be added to any existing Gen4 SBExpress NVMe test system starting with SANBlaze software release V8.2.

Once installed, the application layer program simply called "mi" talks directly to the NVMe Management Interface layer as shown in Figure 1.

The mi program builds an MCTP encapsulated message containing the MI command and passes it via a PCIe VDM message using the SANBlaze VDM controller. This out-of-band message is routed over PCIe using VDM and "Route by ID" to the controller under test.

The VDM capable device under test responds to the SANBlaze VDM controller with the MI information wrapped in an MCTP packet using a VDM message with data. The message is unpacked by the mi program and the NVMe-MI data is returned to the user.

Determining Supported MI Transports

Supported MI transports can be found at the following links (see also Figure 4 below):

https://nvmexpress.org/wp-content/uploads/NVM-Express-1_4-2019.06.10-Ratified.pdf

https://nvmexpress.org/wp-content/uploads/NVM-Express-Management-Interface-1.1-Ratified.pdf

Management Endpoint Capabilities (MEC): This field indicates the capabilities of the Management Endpoint in the Controller.

Bits	Description
7:2	Reserved
1	**PCIe Port Management Endpoint (PCIEME):** If set to '1', then the NVM Subsystem contains a Management Endpoint on a PCIe port.
0	**SMBus/I2C Port Management Endpoint (SMBUSME):** If set to '1', then the NVM Subsystem contains a Management Endpoint on an SMBus/I2C port.

Figure 3: Management Endpoint Capabilities (MEC)

Optional Admin Command Support (OACS): This field indicates the optional Admin commands and features supported by the controller. Refer to section 5.

Bits 15:10 are reserved.

Bit 9 if set to '1', then the controller supports the Get LBA Status capability (refer to section 8.22). If cleared to '0', then the controller does not support the Get LBA Status capability.

Bit 8 if set to '1', then the controller supports the Doorbell Buffer Config command. If cleared to '0', then the controller does not support the Doorbell Buffer Config command.

Bit 7 if set to '1', then the controller supports the Virtualization Management command. If cleared to '0', then the controller does not support the Virtualization Management command.

Bit 6 if set to '1', then the controller supports the NVMe-MI Send and NVMe-MI Receive commands. If cleared to '0', then the controller does not support the NVMe-MI Send and NVMe-MI Receive commands.

Figure 4: Optional Admin Command Support (OACS)

From the SANBlaze CLI

Since MI implementation for NVMe is optional and since there are three possible transports, the following commands can be used to determine if the device under test supports the desired transport.

From the SANBlaze CLI:

```
grep MEC= /iport0/target102
MEC=3
grep OACS= /iport0/target102
OACS=1f
```

From the appropriate specifications, it is determined that the unit under test at target102 supports VDM and SMBus as transports, but not Inband NVMe-MI Send and Receive.

SANBlaze mi Program CLI Example

The mi program is capable of using PCIe VDM, SMBus or inband as a transport for NVMe-MI messages. The same message can be sent via any transport to verify the device under test can support the given method of receiving and sending MI messages.

VDM support in the mi program is available starting with V8.2 software and can be accessed via the GUI, CLI or Python API interfaces.

Now that the groundwork has been laid, let's get to the fun part. The SANBlaze mi program in action at the CLI:

1. Establish an ssh session to the SANBlaze NVMe test system with root privileges:

   Username: **vlun**
   Password: **SANBlaze**

   At the prompt, request super user access, with password SANBlaze (case-sensitive):

   su
   SANBlaze

2. Issue the MI command using the -T flag to specify transport. Valid values for -T are vdm, smbus and inband (see examples below.)

mi Example Read Basic Status Transport=VDM

mi -T vdm -d 2 -t 1 0 0
PCIe path is 0f:00.00

Read NVMe-MI Data Structure
command is 'vdm -a 0f:00.00 -k 0 840800000000000000000000000000000e2000607'

Response Data Length: 32
NVM Subsystem Information Data Structure:
 Number of Ports: 2
 NVMe-MI Major Version Number: 1
 NVMe-MI Minor Version Number: 0

mi Example Read Basic Status Transport=smbus

Since the device under test supports both SMBus and VDM as a transport for MI, the device functionality can be verified by changing the transport to SMBus.

[root@DTMini3 ~]# **mi -T smbus -d 2 -t 1 0 0**
Read NVMe-MI Data Structure
command is 'sb_i2c -n 1 -d 2 -z -k 0 -w 84 08 00 00 00 00 00 00 00 00 00 00 00 00 00 00 e2 00 06 07'

Response Data Length: 32

```
NVM Subsystem Information Data Structure:
  Number of Ports: 2
  NVMe-MI Major Version Number: 1
  NVMe-MI Minor Version Number: 0
```

mi Example Read Vital Product Data

```
mi -T vdm -d 2 -t 1 5
PCIe path is 0f:00.00

VPD Read
command is 'vdm -a 0f:00.00 -k 0 840800000500000000000000000100006c16f156'

Vital Product Data:

Common Header
  IPMI Version Number: 1
  Internal Use Area Starting Offset: 0h
  Chassis Info Area Starting Offset: 0h
  Board Info Area Starting Offset: 1Dh
  Product Info Area Starting Offset: 1h
  MultiRecord Info Area Starting Offset: Fh
  Common Header Checksum: D2h (correct)
Product Info Area
  Product Area Format Version: 1
  Product Info Area Length: 14
  Language Code: 19h
  Manufacturer Name: C8h 'ManufacturerABC'
  Product Name: D8h 'ESSDRIVE              '
  Product Part: E8h 'Dummy                           '
  Product Version: C2h '  '
  Product Serial Number: D4h 'A041CB1A          '
  Asset Tag: 0h
  FRU File ID: 0h
  Product Info Area End: C1h
  Product Info Area Checksum: BBh (correct)
MultiRecord Info Area
  Record Type: Bh
  Record Format: 2h
  Record Length: 28h
  Record Checksum: C9h (correct)
  Header Checksum: 02h (correct)
  NVMe MultiRecord Area
    Record Version Number: 0
    Form Factor: 12h
    Initial 1.8 V Power Supply Requirements: 0
    Maximum 1.8 V Power Supply Requirements: 0
```

```
    Initial 3.3 V Power Supply Requirements: 0
    Maximum 3.3 V Power Supply Requirements: 0
    Maximum 3.3 V Aux Power Supply Requirements: 1
    Initial 5 V Power Supply Requirements: 0
    Maximum 5 V Power Supply Requirements: 0
    Initial 12 V Power Supply Requirements: 10
    Maximum 12 V Power Supply Requirements: 10
    Maximum Thermal Load: 12
    Total NVM Capacity: 6FC7D256000h, 7681501126656 bytes (7153.9 GB)
  MultiRecord Info Area
   Record Type: Ch
   Record Format: 2h
   Record Length: 10h
   Record Checksum: F2h (correct)
   Header Checksum: F0h (correct)
   PCIe Port MultiRecord Area
    Record Version Number: 0
    Port Number: 0h
    Port Information: 1h
    Link Speeds: 7h
    Maximum Link Width: 4h
    MCTP Support: 1h
    Ref Clk Capability: 1h
    Port Identifier: 0h
  MultiRecord Info Area
   Record Type: C0h
   Record Format: 82h
   Record Length: 20h
   Record Checksum: F9h (correct)
   Header Checksum: A5h (correct)
   MR B0-BF: C0 82 20 F9 A5 00 36 00 C0 00 00 11 00 00 00 00       6
   MR C0-CF: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  Board Info Area
   Board Area Format Version: 1
   Board Info Area Length: 2
   Language Code: 19h
   Board Manufacturing Date/Time: 0h
   Board Manufacturer: C0h
   Board Product Name: C0h
   Board Serial Number: C0h
   Board Part Number: C0h
   FRU File ID: C0h
   Board Info Area End: C1h
   Board Info Area Checksum: 63h (correct)
```

Similarly all supported MI commands can be sent and received through any of the three supported transports: VDM, SMBus and inband.

VDM MI Command on the PCIe Bus

The MI message and response are exchanged using Msg Routing by ID as seen in the following PCIe trace.

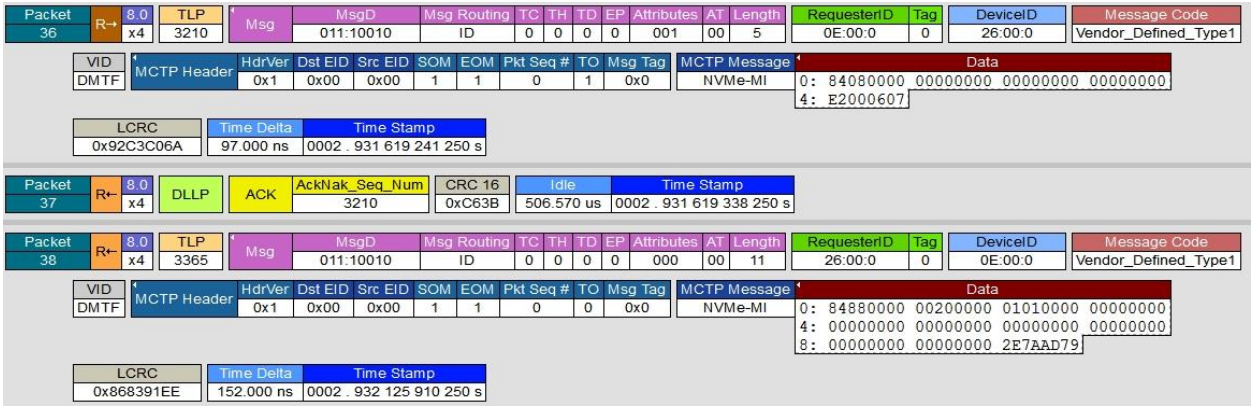

Figure 5: VDM MI Message Exchange on PCIe

SANBlaze mi Program GUI Example

The mi program is also available at the GUI, which makes building mi commands simple. Use the drop down menus on the Namespace/GenericIO web page to select the MI command, transport and options.

The GUI creates the appropriate MI command, which can be captured for later use at the CLI or in a custom script.

Built-In "Certified by SANBlaze" tests are available for automatic verification of your MI implementation and are covered in the next section.

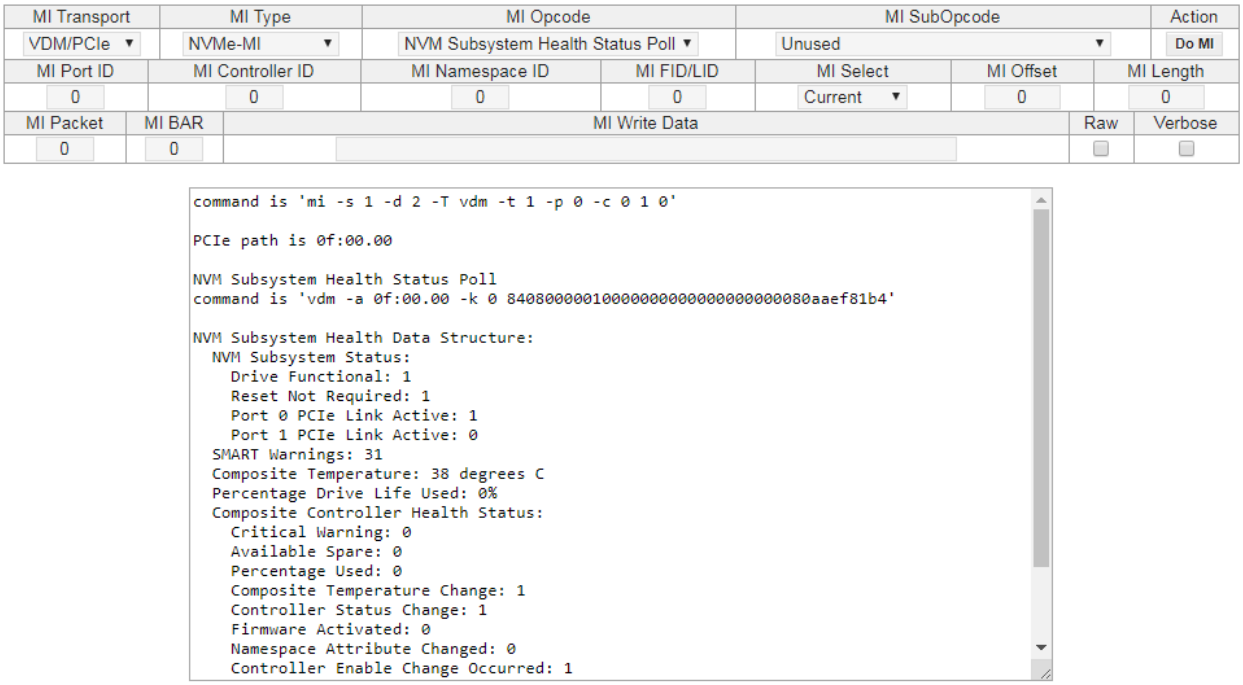

Figure 6: Accessing MI from the SANBlaze GUI

As shown above, the GUI creates an MI command which can be used at the CLI or in a custom script.

VDM MI Certified by SANBlaze

Last, but perhaps most interesting, is the built-in verification of VDM MI by the built-in "Certified by SANBlaze" test suite.

All MI functionality including adherence to the applicable specifications can be verified by selecting the "NVMe MI Full Command Set" as shown in the following figure:

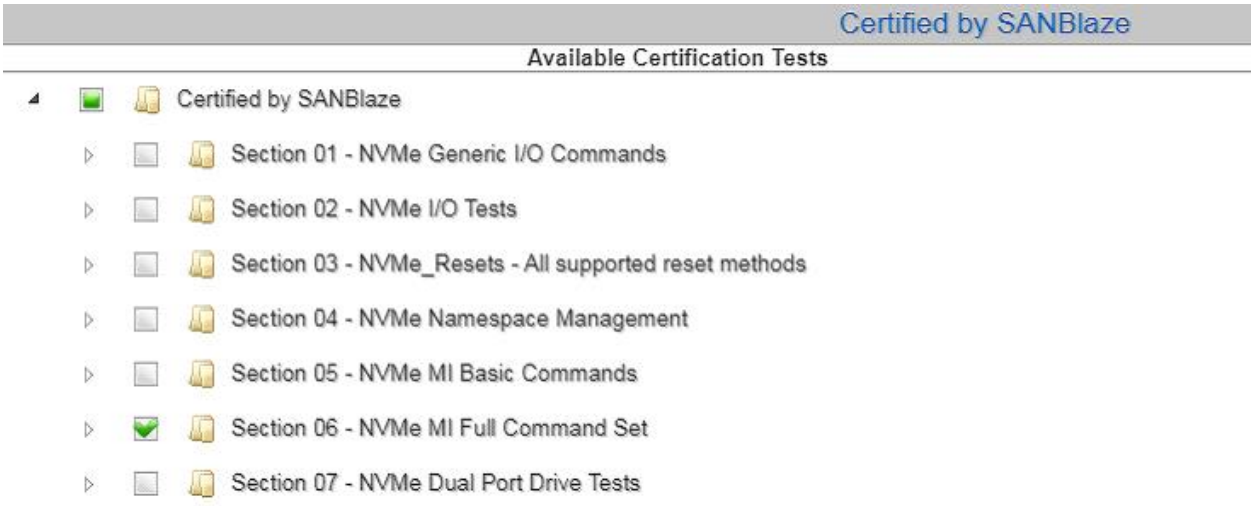

Figure 7: Certified by SANBlaze Testing MI over VDM or SMBus

The tests in the selected groups will be run against the unit under test and a comprehensive report will be generated for the device under test.

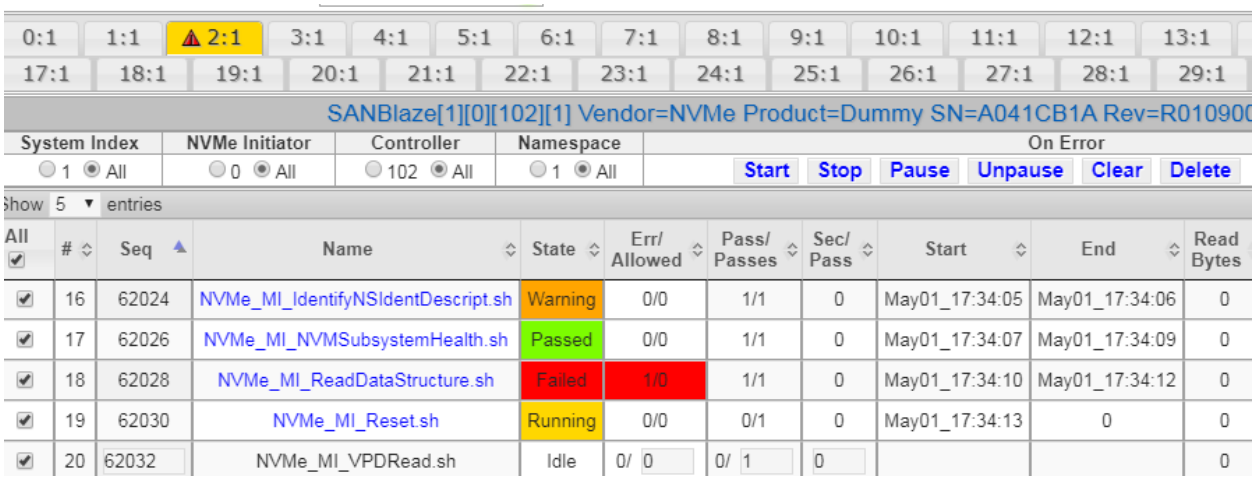

Figure 8: Running "Certified by SANBlaze" MI Qualification Tests

Conclusion

Until now testing of MI via PCIe VDM required complicated hardware and software configurations and produced results that were difficult to achieve and hard to duplicate.

SANBlaze has bundled a VDM hardware module with a simple to use MI packet generator which allows engineers to build custom test scenarios or run any combination of built-in "Certified by SANBlaze" MI tests.

Contact SANBlaze sales for more information at sales@sanblaze.com.

SANBlaze Technology, Inc.
One Monarch Drive, Suite 204
Littleton, MA 01460
(978) 679-1400
Help Center
Storage Testing Sales
Storage Testing Support

For General Inquiries:

Email Sales | Email Support

