LEGAL NOTICE:

This erratum to the NVMe over Fabrics revision 1.0 specification is proprietary to the NVM Express, Inc. (also referred to as "Company") and/or its successors and assigns.

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

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

LEGAL DISCLAIMER:

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

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

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

NVM Express™ Technical Errata

Errata ID	**004**
Revision Date	**08/03/2017**
Affected Spec Ver.	**NVMe over Fabrics 1.0**
Corrected Spec Ver.	

Errata Author(s)

Name	Company
Fred Knight	NetApp Inc.
Brendan Doyle	Oracle
David Black	Dell EMC

Errata

Overview

Clarifications to initial state of Dynamic controllers.

Clarifications to Discovery Log Page contents.

Clarifications to NVM subsystem port, physical fabric port and their relationship.

The requirement that the SGLS field in Identify Controller set bits 01:00 to the value 01b is now explicitly stated rather than just being inferred from other requirements.

Clarification of RKEY invalidation.

Revision History

Revision Date	Change Description
06/16/2017	Incorporate clarifications to Discovery Log Page and initial state of Dynamic Controllers
06/20/2017	Add port definition clarifications
07/12/2017	Add explicit SGLS requirement and RKEY invalidation statement
07/16/2017	Additional controller model clarifications from con-call.

Incompatible Changes

None

Description of Specification Changes

Modify a portion of section 1.4.10 (*port (NVM subsystem port)*) as shown below:

1.4.10 port (NVM subsystem port)

An NVMe over Fabrics protocol interface between an NVM subsystem and a fabric. An NVM subsystem port is a collection of one or more physical fabric interfaces that together act as a single interface ~~between the NVM subsystem and a fabric~~.

Add section 1.4.15 as shown below:

1.4.15 physical fabric interface (physical ports)

A physical connection between an NVM subsystem and a fabric.

Modify a portion of section 1.5.2 (*NVM subsystem*) as shown below:

1.5.2 NVM Subsystem

NVMe over Fabrics builds on the NVM subsystem architecture defined in the NVMe Base specification. An NVM subsystem presents a collection of one to (64K - 16) controllers which are used to access namespaces. The controllers may be associated with hosts through one to 64K NVM subsystem ports.

An NVM subsystem port (port) is a protocol interface between an NVM subsystem and a fabric. An NVM subsystem port is a collection of one or more physical fabric interfaces that together act as a single protocol interface. ~~between the NVM subsystem and a fabric.~~ When link aggregation (e.g., Ethernet) is used, the physical ports for the group of aggregated links constitute a single NVM subsystem port.

An NVM subsystem contains one or more NVM subsystem ports.

Each NVM subsystem port has a 16-bit port identifier (Port ID). An NVM subsystem port is identified by the NVM Subsystem NQN and Port ID. The ports of an NVM subsystem may support different NVMe Transports. An NVM subsystem port may support multiple NVMe Transports if more than one NVMe Transport binding specifications exist for the underlying fabric (e.g., an NVM subsystem port identified by a Port ID may support both iWARP and RoCE). An NVM subsystem implementation may bind specific controllers to specific ports or allow the flexible allocation of controllers between ports, however, once connected, each specific controller is bound to a single NVM subsystem port.

A controller is associated with exactly one host at a time. NVMe over Fabrics allows multiple hosts to connect to different controllers in the NVM subsystem through the same port. All other aspects of NVMe over Fabrics multi-path I/O and namespace sharing are equivalent to that defined in the NVMe Base specification.

An NVM subsystem may optionally include a non-volatile storage medium, and an interface between the controller(s) of the NVM subsystem and the non-volatile storage medium. Controllers expose this non-volatile storage medium to hosts through namespaces. An NVM subsystem is not required to have the same namespaces attached to all controllers. An NVM subsystem may support controllers that expose namespaces or Discovery controllers; but it does not expose a mix of controller types. A Discovery Service is an NVM subsystem that exposes Discovery controllers only.

An association is established between a host and a controller when the host connects to a controller's Admin Queue using the Fabrics Connect command (refer to section 3.3). Within the Connect command, the host specifies the Host NQN, NVM Subsystem NQN, Host Identifier, and may request a specific Controller ID or indicate that it may accept a connection to any available controller. A controller has only one association at a time.

Modify a portion of section 4.2 (Controller Model) as shown below:

4.2 Controller Model

The NVM subsystem may support a dynamic or static controller model. All controllers in the NVM subsystem shall follow the same controller model. A Discovery Controller shall support the dynamic controller model.

In a dynamic controller model, the controller is allocated by the NVM subsystem on demand. In this model, all controllers allocated to a specific host have the same state at the time the association is established, including attached namespaces and Feature settings. Changes to a controller (e.g., attached namespaces, feature settings) after the association is established do not impact other dynamic controllers. The host shall specify a Controller ID of FFFFh when using the Fabrics Connect command (refer to section 3.3) to establish an association with an NVM subsystem using the dynamic controller model.

In a static controller model, controllers that may be allocated to a particular host may have different state at the time the association is established. The controllers within an NVM subsystem are distinguished by their Controller ID. The state that persists across associations is any state that persists across a Controller Level Reset. In a static controller model, different controllers may present different Feature settings or namespace attachments to the same host. The NVM subsystem may allocate particular controllers to specific hosts.

…

If an NVM subsystem is dynamic, then multiple Discovery Loge Page entries (refer to Figure 34) with the Controller ID set to FFFFh may be returned for that NVM subsystem (e.g., to indicate multiple NVM subsystem ports) ~~the Controller ID returned~~ in the Discovery Log ~~shall be FFFFh~~. If an NVM subsystem is static, then multiple Discovery Log Page entries that indicate different Controller ID values may be

returned for that NVM subsystem in the Discovery Log ~~specifying different Controller IDs~~. If an NVM subsystem that is static includes any Discovery Log entries that indicate ~~entry that specifies~~ a Controller ID of FFFEh, then the host should remember the Controller ID returned from the Fabrics Connect command and re-use the allocated Controller ID for future associations to that particular controller.

Modify a portion of section 5 (Discovery Service) as shown below:

5 Discovery Service

NVMe over Fabrics defines a discovery mechanism that a host uses to determine the NVM subsystems that expose namespaces that the host may access. The Discovery Service provides a host with the following capabilities:

- The ability to discover a list of NVM subsystems with namespaces that are accessible to the host.
- The ability to discover multiple paths to an NVM subsystem.
- The ability to discover controllers that are statically configured.

…

The Keep Alive command is reserved for Discovery controllers. A transport may specify a fixed Discovery controller activity timeout value (e.g., 2 minutes). If no commands are received by a Discovery controller within that time period, the controller may perform the actions for Keep Alive Timer expiration defined in section 7.1.2.

A Discovery Log Page with multiple entries for the same NVM subsystem indicates that there are multiple fabric paths to the NVM subsystem, and/or that multiple static controllers may share a fabric path. The host may use this information ~~these paths~~ to form multiple associations to controllers within an NVM subsystem.

Modify a portion of section 5.3 (Discovery Log Page) as shown below:

5.3 Discovery Log Page (Log Identifier 70h)

The Discovery Log Page shall only be supported by Discovery controllers. The Discovery Log Page shall not be supported by controllers that expose namespaces for NVMe over PCIe or NVMe over Fabrics. The Discovery Log Page provides an inventory of NVM subsystems with which a host may attempt to form an association. The Discovery Log may be specific to the host requesting the log. The Discovery Log page is persistent across power cycles.

…

If an NVM subsystem supports the dynamic controller model, then all entries for that NVM subsystem shall have the Controller ID field set to FFFFh. For ~~an NVM subsystem with~~ a particular NVM subsystem port and NVMe Transport address in an NVM subsystem, there shall be no more than ~~only~~ one entry with ~~a~~ the Controller ID field set to:

- FFFFh ~~(for~~ if that NVM subsystem supports the dynamic controller model~~);~~ or
- FFFEh ~~(for~~ if that NVM subsystem supports the static controller model~~)~~.

Modify a portion of section 7.3.2 (Capsules and SGLs) as shown below:

7.3.2 Capsules and SGLs

…

Admin command data is transferred using host-resident data buffers specified in Keyed SGL Data Block descriptor entries. I/O command data is transferred using host-resident data buffers specified in Keyed SGL Data Block descriptor entries or within the capsule. The RDMA Transport supports the SGL Data Block, SGL Last Segment, and Keyed SGL Data Block descriptors only. The RDMA Transport does not support SGLs in host memory; all SGLs shall be contained in the command capsule. Fabrics and Admin commands have one (Keyed) SGL Data Block descriptor (i.e., there are no SGL descriptors following the Submission Queue Entry). I/O commands may have more than one SGL descriptor.

The controller shall set bits 01:00 of the SGLS field to 01b in the Identify Controller data structure (refer to the NVMe Base specification) (i.e., the controller shall support SGLs and impose no alignment or granularity requirements for data blocks).

There are SGL Descriptor Sub Type values that are specific to RDMA operation as defined in Figure 40.

Modify a portion of section 7.3.7 (Key Management) as shown below:

7.3.7 Key Management

…

Commands that require data transfers between a host memory buffer and the controller shall use SGLs that contain a full RDMA host address tuple consisting of an RKEY, Offset, and Length. The host NVMe RDMA Transport is responsible for allocating this tuple by registering the associated data buffers with the appropriate RNIC to produce the RKEY and then subsequently inserting the RKEY, Offset, and Length into the SGL entries associated with the command. The same RKEY may be used in multiple SGL entries associated with the same Fabrics, Admin, or I/O command. The RKEY shall be invalidated only after all ~~any~~ RDMA_READ or RDMA_WRITE operations have been completed that use the RKEY.