LEGAL NOTICE:

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

NOTICE TO USERS WHO ARE NVM EXPRESS, INC. MEMBERS: Members of NVM Express, Inc. have the right to use and implement this NVM Express over Fabrics revision 1.0 technical proposal 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 - 2018 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 VTM, Inc.
3855 SW 153rd Drive
Beaverton, OR 97003 USA
info@nvmexpress.org

NVM Express Technical Proposal for New Feature

Technical Proposal ID	**8005 – Fabric SQ Flow Control**
Change Date	**9/17/2018**
Builds on Specification	**NVM Express over Fabrics 1.0 and ECN 001**

Technical Proposal Author(s)

Name	Company
David Black	Dell EMC

Revision History

Revision Date	Change Description
2018/02/08	Initial version – content from Fabrics 1.0 with notes on what to change.
2018/02/28	First draft of text for changes to sections 2 and 3.
2018/03/02	Add TP summary block and Theory of Operation summary bullet Spell out details for SQ flow control enabled case. Decision: SQ flow control disablement is separately negotiated for each queue pair. Decision: Whether to disable SQ flow control (e.g., “should” vs. “may”) is transport-specific. Add a sentence to allow (“may”) the RDMA Transport to do that. Many other editorial changes.
2018/03/15	MAXCMD is not related to size of SQ when SQ flow control is disabled. Clarify meaning of SQHD value in Connect response. Minor edit to description of new bit in Discovery Log Page Entry, more to come in phase 3. Additional minor edits.
2018/04/05	Add sentence to make it clear that a controller that is capable of disabling SQ flow control may accept or reject a host request to do that. Additional editorial changes, and remove remaining Editor’s Notes.
2018/04/10	Correct specification of SQ flow control being enabled to account for SQHD update optimization that can omit some SQHD values. Additional minor edits.
2018/04/18	Prohibit FFFFh as an initial SQHD value.
2018/04/19	Revise FFFFh SQHD prohibition text, add explanation up front that this prohibition is potentially incompatible in principle, but is expected to be backwards compatible in practice.
2018/04/26	Minor edit to clarify FFFFh SQHD prohibition text.
2018/05/10	Clean version for member review.
2018/05/14	Oops – ECN 001 contains a rewrite of section 2.4. Rebase this TP’s changes to section 2.4 on ECN 001 text, reorganizing as needed. Remove section on Incompatible Changes, as ECN 001 text prohibits the potentially problematic behavior.
2018/05/18	Add specification of what happens when SQ flow control is disabled and the controller detects that there are too many outstanding commands. More editing and further reorganization of the section 2.4 text based on comments from Fred Knight and James Smart
2018/05/21	Add change to “command submission” definition. A few more minor edits
2018/05/24	Minor edits
2018/06/04	If SQ is overrun, controller “shall” terminate transport connection and association instead of “should” terminate. Add informative text on SQ sizing when SQ flow control is disabled. Additional minor edits.
2018/06/08	Describe max sizes for SQs with references to where the sizes are specified.
2018/06/13	Clean version for second member review
2018/08/04	Integration into NVMe over Fabrics 1_0a 2018.07.23 – Ratified.docx
2018/09/06	Integration edits added
2018/09/17	Ratification

This technical proposal allows Submission Queue (SQ) flow control for NVMe over Fabrics to be disabled. SQ flow control support is required in all implementations, and is disabled for a queue pair only when both the host and the controller agree to disable it via the Connect command and response.

Description of Specification Changes

Modify the section 1.4.14 definition (command submission) as shown below (changes shown here are based on changes published in ECN 001):

1.4.14 command submission

Editor's Note: The sentence removed from this definition is already covered in section 2.4 in Fabrics ECN 001 – that text is moved to section 2.4.2 by this TP, as it now only applies when SQ flow control is not disabled.

A command is submitted when a host adds a capsule to a Submission Queue. ~~The host increments the Tail entry pointer associated with that Submission Queue as part of submitting a command.~~

Modify a portion of section 1.5 (Theory of Operation) as shown below:

NVMe over Fabrics has the following differences from the NVMe Base specification:

…

- NVMe over Fabrics does not support PRPs but requires use of SGLs for Admin, I/O, and Fabrics commands. This differs from NVMe over PCIe where SGLs are not supported for Admin commands and are optional for I/O commands;~~ and~~
- NVMe over Fabrics does not support Completion Queue flow control. This requires that the host ensures there are available Completion Queue slots before submitting new commands;and~~.~~
- NVMe over Fabrics allows Submission Queue flow control to be disabled if the host and controller agree to disable it. If Submission Queue flow control is disabled, the host is required to ensure that there are available Submission Queue slots before submitting new commands.

…

Modify a portion of section 2.2 (Response Capsules) as shown below:

The Completion Queue Entry is 16 bytes in size and contains a two byte status field.

The definition for the Completion Queue Entry for a Fabrics command is defined in Figure 9. The definition for the Completion Queue Entry when the command is an Admin or I/O command is defined in section 4.6 of the NVMe Base specification, where the SQ Identifier and Phase Tag fields are reserved because they are not used in NVMe over Fabrics. Use of the SQHD field depends on whether SQ flow control is disabled for the queue pair, refer to section 2.4 and to section 3.3.

Figure 1: Fabrics Response Capsule – Completion Queue Entry Format

<table>
<tr><th>Byte</th><th>Description</th></tr>
<tr><td>07:00</td><td>The definition of this field is Fabrics response type specific.</td></tr>
<tr><td>09:08</td><td>SQ Head Pointer (SQHD): Indicates the current Submission Queue Head pointer for the associated Submission Queue[1].</td></tr>
<tr><td>11:10</td><td>Reserved</td></tr>
<tr><td>13:12</td><td>Command Identifier (CID): Indicates the identifier of the command that is being completed.</td></tr>
<tr><td>15:14</td><td>Status (STS): Specifies status for the associated Fabrics command.
<table>
<tr><th>Bit</th><th>Definition</th></tr>
<tr><td>15:1</td><td>Status Field as defined in section 4.6.1 of the NVMe Base specification.</td></tr>
<tr><td>00</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td colspan="2">NOTES:
1. The SQHD field is reserved if SQ flow control is disabled for the queue pair, refer to section 2.4 and to section 3.3.</td></tr>
</table>

Modify sections 2.4 (Submission Queue and Completion Queue Definition) and 2.4.1 (Submission Queue Flow Control) as shown below (changes shown here are based on ECN 001):

2.4 Submission Queue and Completion Queue Definition

NVMe over Fabrics Submission Queues and Completion Queues are message-based (refer to Figure 1) in contrast to NVMe over PCIe memory-based queues (refer to section 4.1 in NVM Express 1.2.1), Doorbells are not used by NVMe over Fabrics. ~~For the remainder of~~In this section and sections 2.4.1, 2.4.2, and 2.4.3, the terms Submission Queue, Completion Queue and queue refer to NVMe over Fabrics queues unless explicitly stated otherwise.

For NVMe over Fabrics, a queue is a unidirectional communication channel that is used to send capsules between a host and a controller. A host uses Submission Queues to send command capsules (refer to section 2.1) to a controller. A controller uses Completion Queues to send response capsules (refer to section 2.2) to a host. Submission and Completion Queues are created in pairs using the Connect command (refer to section 1.5.7).

Editor's Note: Next 3 paragraphs moved into 2.4.2, fourth paragraph deleted as already covered by 2.4.3

~~Each Submission Queue has a Head entry pointer and a Tail entry pointer that are used to manage the queue and determine the number of outstanding capsules. The Head and Tail entry pointers are initialized to zero when a queue is created. All arithmetic operations and comparisons on entry pointers are performed modulo the queue size with queue wrap conditions taken into account. The host increments the Tail entry pointer when it adds a capsule to a queue. The controller increments the Head entry pointer when it removes a capsule from the queue.~~

~~The Submission Queue is empty when the Head entry pointer equals the Tail entry pointer. A capsule consumer may continue to remove capsules from the queue as long as the empty queue condition is not met.~~

~~The Submission Queue is full when the Head entry pointer equals one more than the Tail entry pointer (i.e., incrementing the Tail entry pointer has caused it to wrap around to just behind the Head entry pointer). A full Submission Queue contains one less capsule than the queue size. A host may continue to add capsules to a Submission Queue as long as the queue is not full.~~

~~Completion Queues do not use Head entry pointers or Tail entry pointers (refer to section 2.4.2).~~

Editor's Note: Moved next two paragraphs up from below with minor edits – they both apply independent of whether or not SQ flow control is in use

The NVMe Transport is responsible for delivering command capsules to the controller and notifying the controller of capsule arrival in a transport-specific fashion.

Altering a command capsule between host submission to the Submission Queue and transport delivery of that capsule to the controller results in undefined behavior.

The ~~definition for the~~ queue attributes of Queue Size, Queue Identifier, and Queue Priority are defined in sections 4.1.3, 4.1.4, and 4.1.5 of NVM Express base specification revision 1.2.1.

NVMe Transports are not required to provide any additional end-to-end flow control. Specific NVMe Transports may require low level flow control for congestion avoidance and reliability; any such additional NVMe Transport flow control is outside the scope of this specification.

Flow control differs for Submission Queues and Completion Queues (refer to sections 2.4.1, 2.4.2, and 2.4.3~~2~~).

2.4.1 Submission Queue Flow Control Negotiation

Use of Submission Queue (SQ) flow control is negotiated for each queue pair by the Connect command and the controller response to the Connect command. SQ flow control shall be used unless it is disabled as a result of that negotiation. If SQ flow control is disabled, then the SQHD field is reserved in all Fabrics response capsules for that queue pair after the response to the Connect command (i.e., in all subsequent response capsules for that queue pair, the controller shall clear the SQHD field to 0h and the host should ignore the SQHD field).

If the host requests that SQ flow control be disabled for a queue pair, then the host should size each Submission Queue to support the maximum number of commands that the host could have outstanding at one time for that Submission Queue.

The maximum size of the Admin Submission Queue is specified in the Admin Max SQ Size (ASQSZ) field of the Discovery Log entry for the NVM subsystem (refer to section 5.3).

The maximum size of an I/O Submission Queue is specified in the Maximum Queue Entries Supported (MQES) field of the Controller Capabilities (CAP) property for the controller (refer to section 3.5.1).

The Maximum Outstanding Commands (MAXCMD) value in the Identify Controller data structure indicates the maximum number of commands that the controller processes at one time for a particular I/O Queue. The host may use this value to size I/O Submission Queues and optimize the number of commands submitted at one time per queue to achieve the best performance.

If SQ flow control is disabled, then the host should limit the number of outstanding commands for a queue pair to be less than the size of the Submission Queue. If the controller detects that the number of outstanding commands for a queue pair is greater than or equal to the size of the Submission Queue, then the controller shall:

a. stop processing commands and set the Controller Fatal Status (CSTS.CFS) bit to '1' (refer to section 9.5 in the NVMe Base specification); and
b. terminate the NVMe Transport connection and end the association between the host and the controller.

2.4.2 Submission Queue Flow Control

This section applies only to Submission Queues that use SQ flow control.

Editor's Note: Next paragraph is first paragraph moved down from 2.4 with a small edit.

~~Each~~The Submission Queue has a Head entry pointer and a Tail entry pointer that are used to manage the queue and determine the number of capsules that the queue contains ~~outstanding capsules~~. The Head and Tail entry pointers are initialized to ~~zero~~0h when a queue is created. All arithmetic operations and comparisons on entry pointers are performed modulo the queue size with queue wrap conditions taken into account. The host increments the Tail entry pointer when it adds a capsule to a queue. The controller increments the Head entry pointer when it removes a capsule from the queue.

Editor's note: Needed to reorder the paragraphs to match this TP because the Submission Queue Head entry pointer was below the Submission Queue Tail entry pointer.

The NVMe over Fabrics Submission Queue Head entry pointer is maintained by the controller and is communicated to the host in the SQHD field of Completion Queue Entries. The host uses the received SQHD values for Submission Queue management (e.g., to determine whether the Submission Queue is full).

Editor's Note: Second sentence in next paragraph moved up to 2.4.

~~An~~The NVMe over Fabrics Submission Queue Tail entry pointer is local to the host and is not communicated to the controller. ~~The NVMe Transport is responsible for promptly delivering command capsules to the controller and notifying the controller of capsule arrival in a transport-specific fashion.~~

Editor's Note: Next paragraph is third paragraph moved down from 2.4.

The Submission Queue is full when the Head entry pointer equals one more than the Tail entry pointer (i.e., incrementing the Tail entry pointer has caused it to wrap around to just behind the Head entry pointer). A full Submission Queue contains one less capsule than the queue size. A host may continue to ~~add capsules~~ submit commands to a Submission Queue as long as the queue is not full.

Editor's Note: Next paragraph moved up to 2.4.
~~Altering a command capsule between host submission to the Submission Queue and transport delivery of that capsule to the controller results in undefined behavior.~~

If the controller detects that the host has submitted a command capsule to a full Submission Queue, then the controller shall:

a. stop processing commands and set the Controller Fatal Status (CSTS.CFS) bit to '1' (refer to section 9.5 in the NVMe Base specification); and
b. terminate the NVMe Transport connection and end the association between the host and the controller.

Editor's Note: Next paragraph is second paragraph moved down from 2.4, but second sentence is removed, as it's covered by the "delivering command capsules" text in 2.4..

The Submission Queue is empty when the Head entry pointer equals the Tail entry pointer. ~~A capsule consumer may continue to remove capsules from the queue as long as the empty queue condition is not met.~~

2.4.~~2~~3 Completion Queue Flow Control Considerations

Editor's Note: No change in this text, but section is renumbered, full text included for complete context.

Completion Queue flow control is not used in NVMe over Fabrics. NVMe over Fabrics Completion Queues do not use either Head entry pointers or Tail entry pointers.

The host should size each Completion Queue to support the maximum number of commands that ~~it~~ the host could have outstanding at one time for a particular Submission Queue. The Completion Queue size may be larger than the size of the corresponding Submission Queue to accommodate responses for commands that are being processed by the controller in addition to responses for commands are still in the Submission Queue.

If the size of a Completion Queue is too small for the number of outstanding commands and the controller submits a response capsule to a full Completion Queue, then the results are undefined.

The Maximum Outstanding Commands (MAXCMD) value in the Identify Controller data structure indicates the maximum number of commands that the controller processes at one time for a particular I/O ~~q~~Queue. The host may use this value to size I/O Completion Queues and optimize the number of commands submitted at one time per queue to achieve the best performance.

Altering a response capsule between controller submission to the Completion Queue and transport delivery of that capsule to the host results in undefined behavior.

Modify a portion of section 3.3 (Connect Command and Response) as shown below:

If the Host Identifier, Host NQN, NVM Subsystem NQN, and Controller ID values specified for an I/O Queue are not the same as the values specified for the associated Admin Queue in which the association between the host and controller was established then a status value of Connect Invalid Parameters is returned. If the Host NQN or NVM Subsystem NQN values do not match the values that the NVM subsystem is configured to support, then a status value of Connect Invalid Parameters is returned. If there is a syntax error in the Host NQN or NVM Subsystem NQN value (refer to section 7.9 in the NVMe Base specification), then a status value of Connect Invalid Parameters is returned. If the Host Identifier is cleared to 0h, then a status value of Connect Invalid Parameters is returned.

Submission Queue (SQ) flow control based on the SQ Head Pointer (SQHD) field in Fabrics response capsules (refer to section 2.2) shall be supported by all hosts and controllers. Use of SQ flow control is negotiated by the Connect command and response. A host requests that SQ flow control be disabled by setting bit 2 of the Connect Attributes field to '1' in a Connect command. A controller that agrees to disable SQ flow control shall set the SQHD field to FFFFh in the response to that Connect command. A controller that does not agree to disable SQ flow control shall set the SQHD field to a value other than FFFFh in the response to that Connect command.

If the Connect command did not request that SQ flow control be disabled, then the controller shall not set the SQHD field to FFFFh in the response to that Connect command.

SQ flow control is disabled and shall not be used for a created queue pair only if:

a. bit 2 is set to '1' in the Connect Attributes field of the Connect command that creates the queue pair; and

b. the SQHD field is set to FFFFh in the response to that Connect command.

If SQ flow control is disabled, then the SQHD field is reserved in Fabrics response capsules for all command completions on that queue pair after the response that completes the Connect command.

SQ flow control is enabled and shall be used for a created queue pair if:

a. bit 2 is cleared to '0' in the Connect Attributes field of the Connect command that creates the queue pair; or

b. the SQHD field is not set to FFFFh in the response to that Connect command.

If SQ flow control is enabled, then the controller shall use the SQHD field in Fabrics response capsules for all command completions on that queue pair, except for command completions that omit the SQHD value due to use of the SQHD pointer update optimization described in section 7.1.1.

Figure 2: Connect Command – Submission Queue Entry

Byte	Description
00	**Opcode (OPC):** Set to 7Fh to indicate a Fabrics command.
[… snipped …]	
46	**Connect Attributes (CATTR):** This field indicates attributes for the connection. Bits 7:~~32~~ are reserved. Bit 2 if set to '1', then the host is requesting that SQ flow control be disabled. If cleared to '0', then SQ flow control shall not be disabled. Bits 1:0 indicate the priority class to use for commands within this Submission Queue. This field is only used when the weighted round robin with urgent priority class is the arbitration mechanism selected, the field is ignored if weighted round robin with urgent priority class is not used. Refer to section 4.11 of the NVMe Base specification. This field is only valid for I/O Queues. It shall be set to 00b for Admin Queue connections.
[… snipped …]	

Value	Definition
00b	Urgent
01b	High
10b	Medium
11b	Low

The Connect response provides status for the Connect command. If a connection is established, then the Controller ID allocated to the host is returned. The Connect response is defined in Figure 21.

For a Connect command that fails:

- the controller shall not return a status value of Invalid Field in Command; and
- the controller shall not add an entry to the Error Information Log.

Figure 3: Connect Response

Byte	Description
03:00	**Status Code Specific:** The value is dependent on the status returned. Refer to Figure 22.
07:04	Reserved
09:08	**SQ Head Pointer (SQHD):** If the Connect command requested that SQ flow control be disabled, then a value of FFFFh in this field indicates that SQ flow control is disabled for the created queue pair. Otherwise, this field ~~i~~indicates the current Submission Queue Head pointer for the associated Submission Queue and also indicates that SQ flow control is enabled for the created queue pair.
11:10	Reserved
13:12	**Command Identifier (CID):** Indicates the identifier of the command that is being completed.
15:14	**Status (STS):** Specifies status for the command. Refer to section 2.2.1 for values specific to the Connect command.

Modify figure 34 (Get Log Page – Discovery Log Page Entry) as shown below:

<table>
<tr>
<td>03</td>
<td>Transport Requirements (TREQ): Indicates requirements for the NVMe Transport.

Bits 7:3~~2~~ are reserved.

Bit 2 if set to ‘1’ indicates that the controller is capable of disabling SQ flow control. A controller that is capable of disabling SQ flow control may accept or reject a host request to disable SQ flow control. If cleared to ‘0’, then the controller requires use of SQ flow control.

Bits 1:0 indicate whether connections shall be made over a fabric secure channel.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>Not specified</td></tr>
<tr><td>01b</td><td>Required</td></tr>
<tr><td>10b</td><td>Not required</td></tr>
<tr><td>11b</td><td>Reserved</td></tr>
</table>
</td>
</tr>
</table>

The following text is the last paragraph in Section 7.1 (Transport Requirements). It is included for reference, as no change is necessary when SQ flow control is disabled:

The NVMe Transport shall provide reliable delivery of response capsules from an NVMe subsystem to a host over each connection. The NVMe Transport shall deliver response capsules that include an SQ Head Pointer (SQHD) value to the host in-order; this includes all Connect response capsules.

Modify section 7.1.1 (Submission Queue Head Pointer Update Optimization) as shown below:

7.1.1 Submission Queue Head Pointer Update Optimization

This optimization does not apply to queue pairs for which Submission Queue (SQ) flow control is disabled, as the SQHD field is reserved if SQ flow control is disabled, refer to section 2.4 and to section 3.3.

…

Add a new section 7.3.6.3 (Submission Queue Flow Control) as shown below:

7.3.6.3 Disabling Submission Queue Flow Control

Hosts and controllers that use the RDMA Transport may disable Submission Queue flow control as part of creating a Submission Queue and Completion Queue pair, refer to section 2.4 and to section 3.3.