LEGAL NOTICE:

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 Proposal for New Feature

Technical Proposal ID	**4008 –Transport SGL**
Change Date	**8/22/2017**
Builds on Specification	**NVM Express 1.3**

Technical Proposal Author(s)

Name	Company
David Black	Dell EMC

This technical proposal adds a new SGL Descriptor type to indicate that the NVMe Transport handles data transfer using Transport specific buffering.

Revision History

Revision Date	Change Description
2017/06/14	Initial version
2017/06/15	Change name of new descriptor to Transport SGL Data Block descriptor, add a descriptive sentence to 4.4 and edit other text to align with existing Data Block descriptors. Add cross-references to Identify Controller data structure
2017/06/29	Edits to add in Identify and Figure 11 changes, plus a few questions to close with the team.
2017/08/22	Ratified.

Description of Specification Changes

Modify 4.4 Scatter Gather List (SGL) as shown below

4.4 Scatter Gather List (SGL)

A Scatter Gather List (SGL) is a data structure in memory address space used to describe a data buffer. The controller indicates the SGL types that it supports in the Identify Controller data structure. A data buffer is either a source buffer or a destination buffer. An SGL contains one or more SGL segments. The total length of the Data Block and Bit Bucket descriptors in an SGL shall be equal to or exceed the amount of data required by the number of logical blocks transferred.

An SGL segment is a Qword aligned data structure in a contiguous region of physical memory describing all, part of, or none of a data buffer and the next SGL segment, if any. An SGL segment consists of an array of one or more SGL descriptors. Only the last descriptor in an SGL segment may be an SGL Segment descriptor or an SGL Last Segment descriptor.

A last SGL segment is an SGL segment that does not contain an SGL Segment descriptor, or an SGL Last Segment descriptor.

A controller may support byte or Dword alignment and granularity of Data Blocks. If a controller supports only Dword alignment and granularity as indicated in the SGL Support field of the Identify Controller data structure (refer to Figure 109), then the values in the Address and Length fields of all Data Block descriptors shall have their lower two bits cleared to 00b. This requirement applies to Data Block descriptors that indicate data and/or metadata memory regions.

A Keyed SGL Data Block descriptor is a Data Block descriptor that includes a key that is used as part of the host memory access. The maximum length that may be specified in a Keyed SGL Data Block descriptor is (16 MB – 1).

A Transport SGL Data Block descriptor is a Data Block descriptor that specifies a data block that is transferred by the NVMe Transport using a transfer mechanism and data buffers that are specific to the NVMe Transport.

The SGL Identifier Descriptor Sub Type field may indicate additional information about a descriptor. As an example, the Sub Type may indicate that the Address field is an offset rather than an absolute address. The Sub Type may also indicate NVMe Transport specific information.

The controller shall abort a command if:

- an SGL segment contains an SGL Segment descriptor or an SGL Last Segment descriptor in other than the last descriptor in the segment;
- a last SGL segment contains an SGL Segment descriptor, or an SGL Last Segment descriptor;
- an SGL descriptor has an unsupported format; or
- an SGL Data Block descriptor contains Address or Length fields with either of the two lower bits set to 1b and the controller supports only Dword alignment and granularity as indicated in the SGL Support field of the Identify Controller data structure. Refer to Figure 109.

Modify Figure 18: SGL Descriptor Type as shown below

Figure 18: SGL Descriptor Type

Code	Descriptor
0h	SGL Data Block descriptor
1h	SGL Bit Bucket descriptor
2h	SGL Segment descriptor
3h	SGL Last Segment descriptor
4h	Keyed SGL Data Block descriptor
5h	Transport SGL Data Block descriptor
~~5h~~6h – Eh	Reserved
Fh	Vendor specific

Insert a new Figure after Figure 24 Keyed SGL Data Block descriptor (this will become Figure 25 after figure renumbering)

Figure 24a: Transport SGL Data Block descriptor

<table>
<tr><th>Bytes</th><th>Description</th></tr>
<tr><td>7:0</td><td>Reserved</td></tr>
<tr><td>11:8</td><td>Length: The Length field specifies the length in bytes of the data block. A Length field set to 00000000h specifies that no data is transferred. A Transport SGL Data Block descriptor specifying that no data is transferred is a valid Transport SGL Data Block descriptor. If the controller requires Dword alignment and granularity as specified in the SGL Support field of Identify Controller (refer to Figure 109) then the lower two bits shall be cleared to 00b.

The data transfer mechanism and data buffers for data specified by a Transport SGL Data Block descriptor are defined by the binding section for the associated NVMe Transport.</td></tr>
<tr><td>14:12</td><td>Reserved</td></tr>
<tr><td>15</td><td>SGL Identifier: The definition of this field is described in the table below.

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>03:00</td><td>SGL Descriptor Sub Type field. Valid values are specified in Figure 19.</td></tr>
<tr><td>07:04</td><td>SGL Descriptor Type: 5h as specified in Figure 18.</td></tr>
</table>
</td></tr>
</table>

Modify Figure 109 as shown below:

<table>
<tr>
<td>539:536</td>
<td>O</td>
<td>
SGL Support (SGLS): This field indicates if SGLs are supported for the NVM Command Set and the particular SGL types supported. Refer to section 4.4.
<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td><s>31:21</s>
31:22</td><td>Reserved</td></tr>
<tr><td>21</td><td>If set to ‘1’, then the controller supports the Transport SGL Data Block descriptor. If cleared to ‘0’, then the controller does not support the Transport SGL Data Block descriptor.</td></tr>
<tr><td>20</td><td>If set to ‘1’, then the controller supports the Address field in SGL Data Block, SGL Segment, and SGL Last Segment descriptor types specifying an offset. If cleared to ‘0’ then the Address field specifying an offset is not supported.</td></tr>
<tr><td>19</td><td>If set to ‘1’, then use of a Metadata Pointer (MPTR) that contains an address of an SGL segment containing exactly one SGL Descriptor that is Qword aligned is supported. If cleared to ‘0’, then use of a MPTR containing an SGL Descriptor is not supported.</td></tr>
<tr><td>18</td><td>If set to ‘1’, then the controller supports commands that contain a data or metadata SGL of a length larger than the amount of data to be transferred. If cleared to ‘0’, then the SGL length shall be equal to the amount of data to be transferred.</td></tr>
<tr><td>17</td><td>If set to ‘1’, then use of a byte aligned contiguous physical buffer of metadata (the Metadata Pointer field in Figure 11) is supported. If cleared to ‘0’, then use of a byte aligned contiguous physical buffer of metadata is not supported.</td></tr>
<tr><td>16</td><td>If set to ‘1’, then the SGL Bit Bucket descriptor is supported. If cleared to ‘0’, then the SGL Bit Bucket descriptor is not supported.</td></tr>
<tr><td>15:03</td><td>Reserved</td></tr>
<tr><td>02</td><td>If set to ‘1’, then the controller supports the Keyed SGL Data Block descriptor. If cleared to ‘0’, then the controller does not support the Keyed SGL Data Block descriptor.</td></tr>
<tr><td>01:00</td><td>This field is used to determine the SGL support for the NVM Command Set. Valid values are shown in the table below.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>SGLs are not supported.</td></tr>
<tr><td>01b</td><td>SGLs are supported. There is no alignment nor granularity requirement for Data Blocks.</td></tr>
<tr><td>10b</td><td>SGLs are supported. There is a Dword alignment and granularity requirement for Data Blocks (refer to section 4.4).</td></tr>
<tr><td>11b</td><td>Reserved</td></tr>
</table>
</td></tr>
</table>
</td>
</tr>
</table>

Modify a portion of Figure 11 as shown below:

<table>
<tr><td>39:24</td><td>
Data Pointer (DPTR): This field specifies the data used in the command.

If CDW0.PSDT is set to 00b, then the definition of this field is:

<table>
<tr><td>39:32</td><td>PRP Entry 2 (PRP2): This field:

a) is reserved if the data transfer does not cross a memory page boundary.

b) specifies the Page Base Address of the second memory page if the data transfer crosses exactly one memory page boundary. E.g.,:

i. the command data transfer length is equal in size to one memory page and the offset portion of the PBAO field of PRP1 is non-zero or

ii. the Offset portion of the PBAO field of PRP1 is equal to zero and the command data transfer length is greater than one memory page and less than or equal to two memory pages in size.

c) is a PRP List pointer if the data transfer crosses more than one memory page boundary. E.g.,:

i. the command data transfer length is greater than or equal to two memory pages in size but the offset portion of the PBAO field of PRP1 is non-zero or

ii. the command data transfer length is equal in size to more than two memory pages and the Offset portion of the PBAO field of PRP1 is equal to zero.</td></tr>
<tr><td>31:24</td><td>PRP Entry 1 (PRP1): This field contains the first PRP entry for the command or a PRP List pointer depending on the command.</td></tr>
</table>

If CDW0.PSDT is set to 01b or 10b, then the definition of this field is:

<table>
<tr><td>39:24</td><td>SGL Entry 1 (SGL1): This field contains the first SGL segment for the command. If the SGL segment is an SGL Data Block or Keyed SGL Data Block or Transport SGL Data Block descriptor, then it describes the entire data transfer. If more than one SGL segment is needed to describe the data transfer, then the first SGL segment is a Segment, or Last Segment descriptor. Refer to section 4.4 for the definition of SGL segments and descriptor types.

The NVMe Transport may support a subset of SGL Descriptor types and features as defined in the NVMe Transport binding specification.</td></tr>
</table>
</td></tr>
</table>