Enabling the NVMe™ CMB and PMR Ecosystem

Stephen Bates, PhD. CTO, Eideticom
Oren Duer. Software Architect, Mellanox

NVM Express Developers Day, May 1, 2018
Outline

1. Intro to NVMe™ Controller Memory Buffers (CMBs)
2. Use cases for CMBs
   a. Submission Queue Support (SQS) only
   b. RDS (Read Data Support) and WDS (Write Data Support) for NVMe p2p copies
   c. SQS, RDS and WDS for optimized NVMe™ over Fabrics (NVMe-oF™)
3. Software for NVMe CMBs
   a. SPDK (Storage Performance Developer Kit) work for NVMe copies.
   b. Linux kernel work for p2pdma and for offload.
4. Roadmap for the future
Intro to Controller Memory Buffers

- CMBs were introduced to the NVMe™ standard in 2014 in version 1.2.
- A NVMe CMB is a PCIe BAR (or part thereof) that can be used for certain NVMe specific data types.
- The main purpose of the CMB is to provide an alternative to:
  - Placing queues in host memory
  - Placing data for DMA in host memory.
- As well as a BAR, two optional NVMe registers are needed:
  - CMBLOC - location
  - CMBSZ - size and supported types
- Multiple vendors support CMB today (Intel, Eideticom, Everspin) or soon (Toshiba, Samsung, WDC etc).
Intro to Controller Memory Buffers

- **A** - This device’s manufacturer has registered its vendor ID and device IDs with the PCIe database. This means you get a human-readable description of it.

- **B** - This device has three PCIe BARs:
  - BAR0 is 16KB and is the standard NVMe™ BAR that any legitimate NVMe device must have.
  - **C** - The third BAR is the Controller Memory Buffer (CMB) which can be used for both NVMe queues and NVMe data.

- **F** - Since this device is a NVMe device it is bound to the standard Linux kernel NVMe driver.

An example of CMBLOC and CMBSZ obtained via nvme-cli:

```
NVMe> nvme list
nvme0: [10000]

NVMe> nvme list namespaces
nvme0: [1]

cmbloc : 3
Offset (c000h): 0 (see cmbsz.c20 for granularity)

nvme0: [10000]

Size: (128): 1288
Size Units: (128): 4 KB

Write Data Support: (WDS): Write Data and metadata transfer in Controller Memory Buffer is not supported
Read Data Support: (RDS): Read Data and metadata transfer in Controller Memory Buffer is not supported
Completion Queue Support: (CQS): Adm and I/O Completion Queues in Controller Memory Buffer is not supported
Submission Queue Support: (SQS): Adm and I/O Submission Queues in Controller Memory Buffer is supported
```
Some Fun Use Cases for CMBs

1. Placing some (or all) of your NVMe™ queues in CMB rather than host memory. Reduce latency [Linux Kernel¹ and SPDK¹].
2. Using the CMB as a DMA buffer allows for offloaded NVMe copies. This can improve performance and offloads the host CPU [SPDK¹].
3. Using the CMB as a DMA buffer allows RDMA NICs to directly place NVMe-oF™ data into the NVMe SSD. Reduce latency and CPU load [Linux Kernel²]

¹Upstream in the relevant tree.
²Proposed patches (see last slide for git repo).
Software for CMBs - SPDK

- Storage Performance Development Kit (SPDK) is a Free and Open Source (FOSS) user-space framework for high performance storage.
- Focus on NVMe™ and NVMe-oF™.
- Code added in Feb 2018 to enable P2P NVMe copies when CMBs allow it.
- A simple example of an application using this new API also in SPDK examples (cmb_copy).
Software for CMBs - SPDK

A - copied 9MB from SSD A to SSD B.
B - less than 1MB of data on PCIe switch Upstream Port.
C - SPDK command line

See https://asciinema.org/a/bkd32zDLyKvlq7F8M5BBvdX42
A P2P framework called p2pdma is being proposed for the Linux kernel. Much more general than NVMe™ CMBs. Any PCIe device can utilize it (NICS, GPGPUs etc.). PCIe drivers can register memory (e.g. CMBs) or request access to memory for DMA. Initial patches use p2pdma to optimize the NVMe-oF™ target code.

<table>
<thead>
<tr>
<th>Mode of Operation</th>
<th>Latency (read/write) us</th>
<th>CPU Utilization</th>
<th>CPU Memory Bandwidth</th>
<th>CPU PCIe Bandwidth</th>
<th>NVMe Bandwidth</th>
<th>Ethernet Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vanilla NVMe-oF</td>
<td>188/227</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ConnectX-5 Offload</td>
<td>129/138</td>
<td>0.02</td>
<td>2.40</td>
<td>1.03</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Eldelcom NoLoad p2pmem</td>
<td>167/212</td>
<td>0.55</td>
<td>0.09</td>
<td>0.01</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ConnectX-5 Offload + Eldelcom NoLoad p2pmem</td>
<td>142/154</td>
<td>0.02</td>
<td>0.02</td>
<td>0.04</td>
<td>1.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

The p2pdma framework can be used to improve NVMe-oF targets. Here we show results from a generic NVMe-oF system.

p2pdma can reduce CPU memory load by x50 and CPU PCIe load by x25. NVMe offload can also be employed to reduce CPU core load by x50.
Software for CMBs - The Linux Kernel

- The hardware setup for the NVMe-oF™ p2pdma testing is as shown on the right.
- The software setup consisted of a modified Linux kernel and standard NVMe-oF configuration tools (mostly nvme-cli and nvmet).
- The Linux kernel used added support for NVMe™ offload and Peer-2-Peer DMAs using an NVMe CMB provided by the Eideticom NVMe device.

This is the NVMe-oF target configuration used. Note RDMA NIC is connected to switch and not CPU Root Port.
Roadmap for CMBs, PMRs and the Software

- NVMe™ CMBs have been in the standard for a while. However it’s only now they are starting to become available and software is starting to utilize them.
- SPDK and the Linux kernel are the two main locations for CMB software enablement today.
  - SPDK: NVMe P2P copies. NVMe-oF™ updates coming.
  - Linux kernel. p2pdma framework upstream soon. Will be expanded to support other NVMe/PCIe resources (e.g. doorbells).
- Persistent Memory Regions add non-volatile CMBs and will require (lots of) software enablement too. They will enable a path to Persistent memory storage on the PCIe bus.
Further Reading, Resources and References

5. Mellanox offload driver - https://github.com/Mellanox/NVMEoF-P2P
6. SDC talk on p2pmem - https://www.youtube.com/watch?v=zEXJ549ealM.
8. Offload+p2pdma white paper link - https://github.com/Mellanox/NVMEoF-P2P
9. https://docs.google.com/document/d/1GVGCLALneyw3pyKYmRRG7VTNPzL0XqrFlYA53rx_M/edit?usp=sharing.
2018 Storage Performance Development Kit (SPDK) Summit

May 15th -16th

Dolce Hayes Mansion, San Jose
200 Edenvale Avenue, San Jose, California 95136

This will be a great opportunity to meet with other SPDK community members and listen to a new series of talks from SPDK users and developers; everything from case studies and analysis to tech tutorials and live demos.

This year we will dedicate a second day just for developers that will include a hands-on lab, as well as a few hours set aside for active contributors to tackle design issues and discuss future advancements.

Registration is free!!!!!

http://www.cvent.com/d/qgqnn3

Sponsored by Intel® Corporation