**OFI WG Data Storage / Data Access Subteam Weekly telecom – 11/04/2014**

**OFIWG Download Site:** [www.openfabrics.org](http://www.openfabrics.org) 🡪OFED/OFA Resources 🡪 OpenFabrics Interfaces WG

**Agenda**

* role call,
* Bernard Metzler’s presentation on byte addressable memory architecture

Reprising Bernard’s materials from last week on byte addressable memory.

Reviewed some basic, general requests:

- byte granular access,

- RDMA Read/Write, atomics, Expose NVM specifics

Ordering requirements

- strict ordering

- lazy ordering,

- explicit unordered

- ordering selectable

- fencing

Write Completion level

- write/send with lazy completion – data has reached the peer, but may not have been written

- write/send committed – data has been written to target device

Read and Write Acceleration

- read ahead

- write more

Memory Registration and Addressing

- zero-based addressing – support zero based addressing (same as physical addressing, but of course there is only a key, no physical address. ZBA gives an offset into physical memory space.

-Registration mapped by VA – allow NVM registration using VA from resource mapping

- registration by opaque resource ID (e.g. memory handle, as opposed to a VA)

- re-registration by reservation key – allow re-registration of previously registered persistent memory object.

- re-size memory registration

NVM Specific Commands Support

- TRIM support – a reservation stays up, but the current content of the registration is not considered usable. It only becomes valid again if it is re-written. If you have persistent memory, you are creating/reading/writing files, but device doesn’t know that a file has potentially been closed. Useful for e.g. garbage collection, but allows the resource to be re-used, with the previous contents having been lost, but you may not be able to guarantee that the previous contents have been wiped.

Accessing local NVM

- single EndPoint for operations on local storage. allows access to local NVM.

Mixed DRAM and NVM access

- work request may reference any memory type. SGL may reference a mix of storage and DRAM.

Any performance requirements? Many people have a requirement that the client of an RDMA operation is not involved in the operation, even though it is NVM memory being accessed.

New slide deck – “Accessing byte-addressable NVM using OFI”

- Source of Requirements: requirements arise due to access delay, persistency…

- Lazy Ordered vs Explicitly Unordered – an example showing the difference. Compensates for media access delays, enhances access efficiency by merging, mix access to different media types (DRAM/NVM)

- Grouping (“Extended Atomics”)

- Write/Read completion level – write into NVM with speculative completion (sufficient that data has reached the peer), write into NVM with commit (on write completion).

- VA-based addressing of NVM memory –

- Key/Object-based addressing – a key already exists from a previous registration. Avoids maintaining VA, but requires zero-based addressing (physical memory address offset

- Access local NVM using a single Endpoint

All the above has been prototyped by IBM, which intends to open source shortly.

**Agenda for next meeting**

Chet Douglas – some challenges being faced by Intel (and other processor vendors) in making data persistent.

**Next regular telecom**

Next meeting: Tuesday, 12/2/14

8am-9am Pacific daylight time

**NOTE:** We have switched over to using Webex (courtesy of Cisco). The URL for joining meetings is:

<https://cisco.webex.com/cisco/j.php?J=200935598&PW=67935ad6df07030d5f05044a5b0f>