12th ANNUAL WORKSHOP 2016

PERSISTENT MEMORY BOF UPDATE

Tom Talpey, Microsoft

Stephen Bates, Microsemi

April 8, 2016

"ADDING PERSISTENCE TO RDMA"

- **BoF held Thursday evening**
- **50+ attendees!**
- **Lively discussion**
- **~5:00pm - ~7:15pm**
  - Cut short by facilities issue
- **Led by Tom Talpey**
  - Stephen Bates unable to attend ☹

MAIN DISCUSSION POINTS

- **All agree on need for explicit remote commit operation**
- **Much discussion on:**
  - Commit scope could encompass:
    - “everything” (system global)
    - “everything from this connection”
    - single region/offset/length
    - multiple region/offset/length *(preferred? With limits)*
    - explicitly tagged
  - Ordering and fencing
    - Does commit impose an explicit fence?
    - Or should a fence be specified by the initiator?
    - What other ordering is desirable?
      - E.g. “Log writer” scenario – durably write a log record, atomically/durably update pointer
  - Can RDMA Writes be decorated instead?
    - Discussion of pros and cons (consensus more cons than pros)

FURTHER DISCUSSION POINTS

- **Discussion continued:**
  - Piggybacking/aggregating commit responses
    - Seen as potential optimization, but doesn't fundamentally alter the model
  - Ordering across ranges
    - How does commit(region a) affect non-overlapping commit(region b)?
    - How can an upper layer use multiple connections for write and commit?
    - Consensus that these points are important to explore
  - Ordering on an unordered transport
    - E.g. can this be supported over a datagram service?
  - Error reporting/recovery
    - Meaning of Commit returning a "status"
    - Implications of supporting wide/multiple commit range
  - Will upper layer "push mode" contribute to in-cast congestion at the PM?
    - Possibly – important area to explore
    - Crediting and QoS policies still relevant
    - Note however – push mode is only one model for using Commit

RELATIONSHIP TO EXISTING APIS

- **SNIA NVM Programming Library**
- **Windows and Linux "mapped files"**
  - Windows: MapViewOfFile/FlushViewOfFile
  - Linux/Posix: mmap/msync
  - Both have a Load/Store (native instruction) paradigm, with explicit flush
    - Natural mapping of flush/sync to OptimizedFlush and RDMA Commit
    - Unnatural mapping of load/store to decorated write
  - Is an asynchronous commit useful?
    - Note: SNIA NVM TWG is exploring this, answer appears to be yes
- **Higher-layer application semantics**
  - Databases
  - Transaction libraries
  - Language/compiler extensions
- **Desire broader engagement and dialog with developers**
  - With a goal to provide fundamental network primitives
  - Layered support, phased utilization

NEXT STEPS

- **Consensus desire to have coordinated discussions**
- **In and among relevant groups:**
  - OFA
  - IBTA
  - IETF
  - SNIA
  - NVMe Consortium?
  - ???
- **No conclusion whether a single organization can shepherd**
- **But strong desire to have one!**