14th ANNUAL WORKSHOP 2018

REMOTE PERSISTENT MEMORY THINK TANK

Report Out

Prepared by a cast of thousands

April 13, 2018

THINK TANK ABSTRACT

- **Challenge - Some people think that Remote Persistent Memory over Fabrics is the Next Big Deal. But is it really? If it is, what is needed to get it off the ground and to drive adoption?**
- **Abstract – There has been a lot of buzz in the industry over the last few years related to Persistent Memory but much of that buzz has been associated with local Persistent Memory. There have been efforts to get a discussion about remote PM off the ground, but so far they have been relatively isolated efforts. This Think Tank is focused on a) understanding if there is interest in remote persistent memory, and b) if so, what is needed to get the effort moving.**

PROCESS → ASSIGNMENT

- **Identify enough use cases for remote PM to decide if there is adequate motivation to move forward**
- **Identify current obstacles**
- **Define the missing/incomplete elements that are required to enable rapid/broad adoption of remote PM**
- **Identify appropriate forums needed to overcome those obstacles**

EXECUTIVE SUMMARY

- **Big Deal? Yes, but within reason. Much work remains to quantify and demonstrate a crisp value proposition.**
  - "There's a pony in here somewhere"
- **Use Cases – we focused on a couple, but there's a list. The list needs to be explored more deeply.**
- **Current Obstacles – Yes ☺**
- **What's Required to Get It Going – A few items were identified, there are more**
- **Process –**
  - Refine the list of use cases and identify more
  - Identify common characteristics among them
- **Appropriate Forums – Use the new SNIA/OFA Work Register. Meetings in the context of OFIWG to encourage participation in defining *Use Cases***

AN IMPORTANT CAUTION

What and Why, but NOT How

- **We need to be very careful to keep the API requirements pure and distinct from the characteristics of any given implementation**

- **To do that, we will focus on:**

- **What does the use case demand?**
- **Why does the use case demand it?**
- **~~X How does the programming model provide that function~~?**

- **We will steer clear of mechanisms**

SOME RANDOM OBSERVATIONS

- **Interesting thought: Think of RPM in terms of non-uniform memory**
  - accesses to RPM vs LPM are by definition NUMA
  - stage accesses from RPM to either local DRAM or local PM
  - Assumption of non-uniform memory → may be more flexible, has more attributes
- **Persistent memory devices have two distinct attributes:**
  - High density / capacity compared to DRAM (and at lower power compared to DRAM)
  - Persistence
- **Some usages may value one over the other**
  - e.g. Dreamworks. persistence is valuable, but the Dreamworks usage is driven by the availability of large capacity, shared database.
  - The database is Read Mostly, Write Infrequently
  - Programming model for use cases that don't depend on persistence may be significantly easier

SOME RANDOM OBSERVATIONS

- **Persistence Memory is neither memory nor storage. It is a new layer in the memory hierarchy**
- **Not all usages assume cached data**
  - e.g. Put/Get operations are non-cached

  → a cache flush/commit semantic may not be needed in all RPM use cases

CHECKPOINT RESTART

- **Problem: Checkpointing of a large machine can consume a significant fraction of the available compute time.**
- **In the limit, a machine spends all its time checkpointing and does not make forward progress on the application**
- **Checkpointing today is typically done to a burst buffer and then spooled off to Lustre in the background**
- **Objective – spend less time checkpointing and more time computing**
- **Two possible approaches:**

1. **Accelerate the existing paradigm by viewing RPM as an exquisitely fast and large burst buffer**
2. **Write a new stack based on an HA model – nontrivial**

DREAMWORKS – THE 8 SECOND PROBLEM

- **Currently each animator works with only a few seconds of an animated movie**
- **Typically, he cannot see most of the rest of the movie**
- **Objectives**
  - Enough fabric attached memory to contain the entire movie
  - Accessible by all animators
- **If it's PM, it fits more neatly into a memory hierarchy than a storage hierarchy**

MACHINE LEARNING

- **Characteristic workloads tend to be 80% random reads**
- **Probably a poor fit for a sequential storage device**

STREAMING SENSOR DATA USE CASE

- **Use case: Capturing data streaming from a sensor array**
- **Performance solutions today are small and expensive**
- **Capacity solutions may not be high enough performance**

WILL DEVELOPERS ADOPT IT?

- **In the absence of a killer app, adoption will be a function of:**
  - Price – Capacity – Persistence - Performance
- **Example: there are multiple ways to achieve high capacity**
  - RPM might be adopted if the price / capacity is reasonable
- **A killer app: one where the introduction of RPM enables functionality that is not possible today**
  - There are no killer apps (yet)
- **The checkpoint/restart problem may become a killer app**
- **Ira's killer app: Parallel address space, treating as memory not suffering DRAM cache misses. Eg: PGAS**

WHAT'S MISSING?

- **Global addressability**
- **Synchronization mechanisms**
  - multiple instances of an application must block until data has reached a persistent state
- **What is the availability model in the face of power failure?**
  - Power fail atomics
- **Data validation problem – no way to know if data arrives on the persistent media intact**
  - Same problem with storage today
- **What is equivalent of pcommit and remote PM?**
- **No use case for pcommit?**

CONSOLIDATED USE CASES - CAVEATS

- **Liberal enumeration with some attempt at application relevant framing**
- **Overlaps information on earlier slides (cross-referenced)**
- **Tried to omit technology-only statements, although some descriptions are technology centric. At least hint at "why" with some "what" and no "how".**
- **Some use cases may overlap or be relevant when used together**

CONSOLIDATED USE CASE LIST

1. **Local copy centric – data is copied from remote PM to local DRAM or PM for caching and/or manipulation, then copied back as needed**
2. **Immediate high availability - Local access to PM + remote access for HA for data recovery and failover with little to no work loss**
3. **HPC Checkpoint/Restart – Application pauses to enable rapid copy of relevant state to a checkpoint. Recovery can reference a recent checkpoint. Job suspend/resume references most recent checkpoint. (slide 8 above)**

CONSOLIDATED USE CASE LIST

4. **Distributed collaboration – Remote PM provides a central repository for the recent work of a distributed team collaborating on a large artifact such a movie. Fragments being created by individuals can be quickly viewed locally in the surrounding context. (slide 9 above)**
5. **Random byte range read after ingest – Applications such as machine learning with workloads that are mostly short random reads by parallel threads after ingesting a large body of data. (slide 10 above)**
6. **Aggregated updates – Cache line accesses such as those comprising a transaction are aggregated for communication to remote PM for visibility and/or redundancy.**

CONSOLIDATED USE CASE LIST

7. **NUMA on Steroids – Extend and merge the concepts of NUMA, caching and tiering from CPU's and storage to provide autonomous operation controlled by application informed allocation policies (slide 6 above)**
8. **Volatile memory expansion – Remote PM is used to expand volatile memory capacity with lower cost, higher density and larger (beyond local) scale than DRAM.**
9. **Mirrored transactions – Transactions using local PM are replicated in well know ways to local PM on other nodes. May or may not require remote access at the PM level.**

CONSOLIDATED USE CASE LIST

10. **GPU centric – Copy state directly between GPU memory and local or remote PM without going through DRAM.**
11. **Rehydration – Remote PM used for DB logs/checkpoints to enable rapid re-hydration of memory after failure. Cost is more aligned with large scale (e.g. SAP HANA) than scaled out open source**
12. **Metadata de-amplification – When metadata becomes larger than memory, metadata paging can cause read/write amplification relative to payload data read/write. PM density and remote PM scale can offset this type of amplification.**
13. **IoT Sensor data – Streams of information within edge or between edge and centralized repository. (slide 11)**

CONSOLIDATED GAP LIST

1. **Global addressability – means of identifying versioned data relevant to an application during restore (HA or checkpoint). Applicable to sharing as well?**
2. **Synchronization for weakened memory semantics – non cache coherent access (for recovery or sharing at scale) still requires a means of establishing weaker consistency appropriate to the application. Ordering barriers (as opposed to fine grain acknowledgement) may be one approach.**
3. **Power fail atomicity – different from current processor and network atomics, proving to have vexing subtleties**
4. **NIC persistence – Means of allowing network endpoints to participate in power fail atomicity.**
5. **Economics estimation - Tools for estimating and comparing economics of memory/storage hierarchies with and without (remote) PM**
6. **Evolved NUMA – enable use case 7 above.**

POINTS OF CONTACT

- **OFA – OFIWG Co-chairs**
  - Paul Grun grun@cray.com
  - Sean Hefty sean.hefty@intel.com
- **SNIA – NVMP TWG Co-chairs**
  - Doug Voigt doug.voigt@hpe.com
  - Alan Bumgarner alan.bumgarner@intel.com
- **Mail reflectors**
  - ofa_remotepm@lists.openfabrics.org
  - ofiwg@lists.openfabrics.org
  - To subscribe go to http://lists.openfabrics.org/mailman/listinfo/
- **Stay tuned for meeting announcements**

BACKUP SLIDES AND RAW NOTES FROM OUR DISCUSSION

WHAT IS PMOF?

- **Is this distributed file system or shared memory?**
  - Andy: Oracle exascale showed clients could access memory in remote server without impacting server CPU cycles.
- **How to create persistence? No flushing needed in future?**

USE CASES

- **HA usage – first among others. What's driving the HA usage?**
- **Identify early adopters or usages for PM.**
- **How are these getting used?**
- **Scale out – use case where PM has storage.**
- **Fast resume-usage**
- **What about Cost?**
- **Scott Ashley – Assume its faster than flash, assume its byte addressable. There is no need for byte addressable → requires new S/W stack.**
- **Use case for scott: Checkpoint restart.**
  - Why not use local PM? Requires S/W to refer to pointer in PM
  - Could save the state then theoretically can refer to that state. Similar to VM pause and resume. Pause just before the the crash

NEW SW STACK DEFINITION

- **On top of current HA usage? – Yes, HA is required.**
  - Replace Lustre
- **Acknowledging Checkpoint as a usage**
- **Regular memory in 2 nodes works? We cannot use memory for storage; should be dedicated to computation**
- **Why checkpoint on PM? Why doesn't existing solutions work?**
  - Higher bandwidth, less time checkpointing, more time computing
- **If CPUs can support then it will be awesome**
- **Local pooled memory → acts as shared memory. Eg: Dreamworks**

MACHINE LEARNING USAGE

- **90% read and a lot of random access**
- **Two kinds of PM**
  - Expensive than DRAM; doesn't work for making copies
  - Cheaper than DRAM
- **Assumption of non-uniform memory → may be more flexible, has more attributes**
- **Could RPM be non-uniform memory**
- **Two attributes for PM= Capability, persistence. Which are important?**
  - Both

- **Persistence Memory is a new tier of memory.**
- **SAP Hana solving the problem that their memory is now persistent**
- **High bandwidth but capacity is pretty limited but cannot afford**
- **At some point, capacity will be impacted when heuristics come into play**
- **NVDIMM-T; mixed media in DIMM Form factor**

SO IS IT A BIG DEAL?

- **100 of linux cheap boxes → still don't know PM relevance in DBs**
- **From DB POV don't see case for RPM, may be cheaper if we use NVMe SSD for replication**
- **Its interesting, another layer that is farther from local memory**
- **Storage is easier to pool; what about pool of PM? Issue: cost & capacity**
- **Front end is RPM but back end is different usages like battery backup**

WHY AREN'T DEVELOPERS DEVELOPING APPS?

- **Local PM accesses are enabled by JEDEC, Oses…**
- **HA, price, density is relevant. Perf can be secondary**
  - Byte addressability is not an issue
  - Lower latency, improved bandwidth works if price allows.
- **Ira's killer app: Parallel address space, treating as memory not suffering DRAM cache misses. Eg: PGAS**

PM ROLE

- **Byte addressable PM is fast; what is the cost replicated? Because this is the limiting factor**
- **Checkpoints can be done today but compute is sacrificed. Machine fails. Machine approx 20,000 cluster nodes (2S, IB)**

BIG QUESTIONS (DURING CONCLUSION)

- **PM is here, well and alive. But Data center loses power then what happens? Who maintains state. …and restore**
- **Univ of Santa Cruz → address PM so that it is globally available? – Project name Twizzler (LINK)**
- **Can multiple LPMs work as RPM? Pre-fetch, NUMA idea…Full PM is somewhere else but some pieces of it is pre-fetched into LPM**