Cache Coherency & Memory Model in RDMA-Backed Software-Coherent DSM

Zhengyi Chen

January 29, 2024

Table of Contents

1. Overview

- DSM used to be constrained by NIC bandwidth & transfer rate (e.g., during the 1990s).
- The advent of high(er) transfer rate NICs allows the DSM idea to be revived.
- Orthogonally, hardware acceleration resources are scarce and highly valuable.
  - Traditional Scheduling Mechanisms within a Cluster cannot dynamically allocate hardware accelerators without high overhead.
- Ideally, via high-speed NICs, hardware accelerator could be statically allocated such that:
  - Every node have access to the hardware accelerator node in a time-shared fashion.
  - Accelerator-attached node can access remote memory much like attaching accelerator over, say, PCIe.

Heterogeneous Memory Management

- **HMM** facilitates shared address space and transparent data migration between CPU and peripherals. Specifically:
  - HMM provides interface for duplicating the CPU page table with that of the device's, which are transparently synchronized.
  - It also provides corresponding `struct page` representation of device memory pages, which are faulted between the CPU and device.
- Theoretically, this should allow for devices in remote nodes to perform HMM using the DMA-capable NIC as a "proxy HMM device".
- Details of implementation of DSM-over-HMM is beyond this thesis's scope.
  - This thesis focuses on studying and implementing cache coherency and later, memory model for the DSM subsystem of this wider, ongoing project.

Cache Coherency, and Why It Matters Here

- Cache-incoherent RDMA (e.g., mlx) performs DMA without synchronization with CPU cache.
- We cannot assume MMU to magically automatically maintain coherence.
- At transportation time:
  - Send to remote: flushes cache into memory before posting send message.
  - Receive from remote: invalidate cache entry after worked recv message.
- Example: Linux kernel tree, *smbdirect* implementation.
  - *smbdirect* opportunistically establish SMB over RDMA-capable network.
  - `smbd_post_send` cleans cache entry prior to posting send request.
  - `recv_done` invalidates cache entry after exiting softirq for recv request (as callback from RDMA driver).

Consistency Model and Protocol

- Majority of DSM literatures apply **release consistency** as the system's memory model.
- With **single-writer** protocol, however, the memory model can be strengthened with little increase in code complexity.
  - *DSPM*[1], for example, achieves a *de-facto* TSO consistency from its multi-writer release consistency counterpart – assuming correct memory barriers within each node's CPU, distributed writes are never reordered, and distributed reads can overtake writes.
  - Consequently, one can easily achieve sequential consistency by designating the entire write-access duration as a critical section.
- HMM's "CPU-or-device" data migration model also strongly implies a single-writer consistency protocol.

2. Design

- Designing a DSM necessitates designing:
  - Consistency Model.
  - Coherence Protocol and State Machine.
  - Access Control.
- Care needs to be taken to ensure that the in-kernel implementation is:
  - Correct,
  - Performant,
  - Exploits RDMA's traits.

Consistency Model

Coherence Protocol

Stateful Nodes

Progress

On-demand Coherency in ARM64

- ARMv8 defines two levels of cache coherence:
  - *Point-of-Unification*:
  - *Point-of-Coherence*:

Kernel Patch for On-demand Coherency

Proof-of-Concept Kernel Module