Documenting the Heterogeneous Memory Model Architecture

Felix Kuehling
Background

• HMM provides powerful tools for driver developers
• Motivated by earlier attempts to partially implement in drivers
  • Using get_user_pages, MMU notifiers
  • Prone to race conditions, lock dependency issues
  • Different (buggy) solutions in different drivers
• New functionality not previously possible
  • ZONEDEVICE represents device pages in VMAs
  • Page-based migration of anonymous memory to ZONEDEVICE
• Existing documentation written mostly for driver developers
Questions raised recently

Architectural questions raised by MM and FS experts:

• Should ZONE_DEVICE pages be pfn_valid?
• Does page_lock guarantee exclusive access to ZONE_DEVICE pages
• How does FS know when pages are dirtied by a device
• How does demand-paging in and out of device memory work
• How does I/O to/from ZONE_DEVICE pages work

• Where is the architecture documentation?
Immediate goal

- Add DEVICE_PUBLIC support for Frontier
  - Aiming for v5.16
- Need consensus from affected MM and FS maintainers
  - Add any new documentation needed to establish consensus
Future areas of interest

• Migration of file-backed pages to ZONEDEVICE
  • Demand-paging of large data sets
  • Loading compute shader code with mmap

• Replication of shared, read-only pages on multiple devices
  • Optimize access performance, memory usage for compute shader code

• Improve THP handling
  • We’ve seen migrations of huge pages fail in some cases
  • GPU page tables can handle huge pages (2MB, 1GB)

• P2P with other HMM clients
  • Avoid CPU faults on peer access to DEVICE_PRIVATE (e.g. RDMA)