CXL Dynamic Capacity Device / Memory Sharing
What we are after?

• Scare people into paying attention.
• Discussion of some aspects.
• Unlikely we’ll get to any conclusions – no code yet!

• Broad strategy question:
  “Solve for simple first, or try for unified solution?”
Overview

• Not enough time!
• Basic DCD introduction.
• Use cases
• Sharing
• ‘Plan’.
Simple Viewpoint – at boot

- At ‘boot’ mapping from Device Physical Address to Host Physical Address established.
- No ‘extents’ present
  Extents are contiguous region of memory {base, size}
Add some extents

- OoB agent tells devices to add some extents (don’t care how this happens!)
- Driver notified of new extents (interrupt plus description via mailbox)
Add some more extents

- Driver notified of more extents.
Delete some extents.

- Driver notified of request to release extents
- There is a scary force release path to handle (ignored for now!)
Delete some extents.

- Driver notified of request to release extents
- If Linux can release extent... Stop using it and...
- Notify device.
Use Cases

• TCO Reduction, DCD can fulfill temporary need of additional memory in the workload from the memory pool.
• Dynamic aspect brings repurpose of the pool memory.

Node-0  Node-1  Node-N

DCD Extents  DCD Extents  DCD Extents

CXL -Dynamic capacity device

Memory Pool  DRAM
• Make use of Idle compute by provisioning more memory using DCD.

• Memory sharing among compute hosts.
Dev-Dax?

- Dynamic Capacity Devices:
  - Allocate new physical capacity to a host
  - Request that capacity back.
  - The “get it back” guarantee can be non-deterministic.
  - Makes it hard to manage as hotplug, DAX more suitable.

- Regions:
  - Concept already exists in DAX
  - Allows carving out an arbitrary number of mappable devices.
  - Sparse memory and resizing of DAX regions matches to the DCD.
Challenges

• Event flow
  • Many inbound capacity events related to a single memory allocation requests.
  • Policy control to add to existing DAX device or add a new one.

• Common with existing CXL type 3: Need to decide whether to provide to OS as normal memory (more complex as sparse!).
Not so simple.

- Two types of allocation.
  - Untagged – general purpose (KMEM?)
  - Tagged – intended for use by specific application (DAX?)
    - Free based on tag or extent.
- Handling the Interleaving case.
- Memory Sharing.
• Hosts see different DPA extents for same physical memory!
• Could be remapped to contiguous VA
Plan...

- Emulation – need a platform.
- Simple first?
  - Not shared
  - Volatile (ordering does matter)
  - Tagged – so application specific
  - Whole region allocated or freed
- Generalize to dynamic
  - Addition + removal of extents to existing DAX region (ideas from virtio-mem?)
- Hit the complex cases
  - Shared / non-volatile.
- Profit