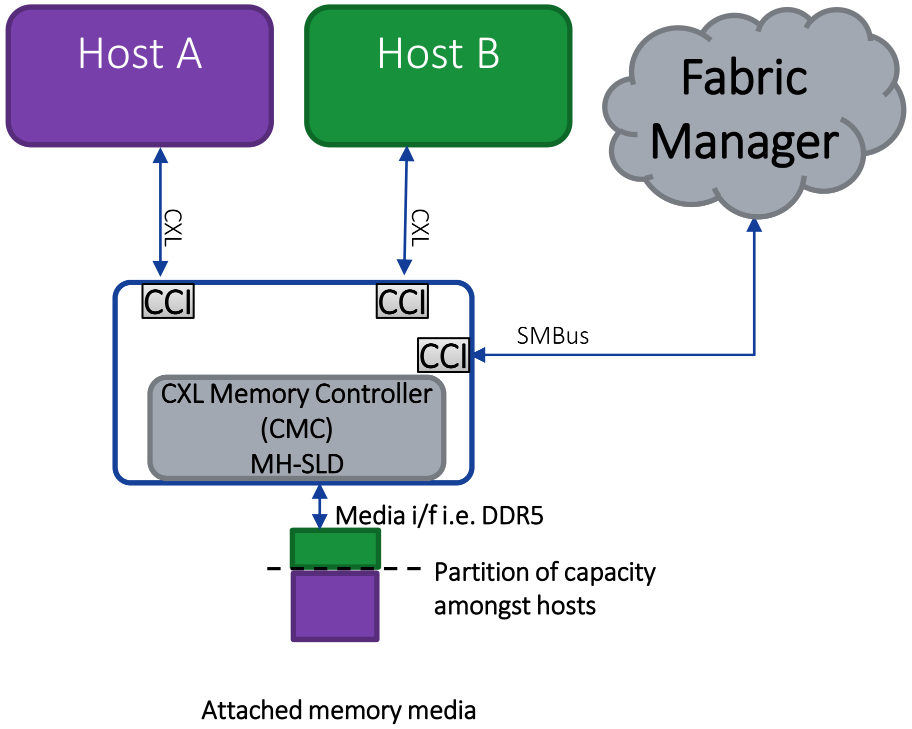
Pooling/Sharing using MH-SLD vs MLD

Sanjay Goyal/Rambus

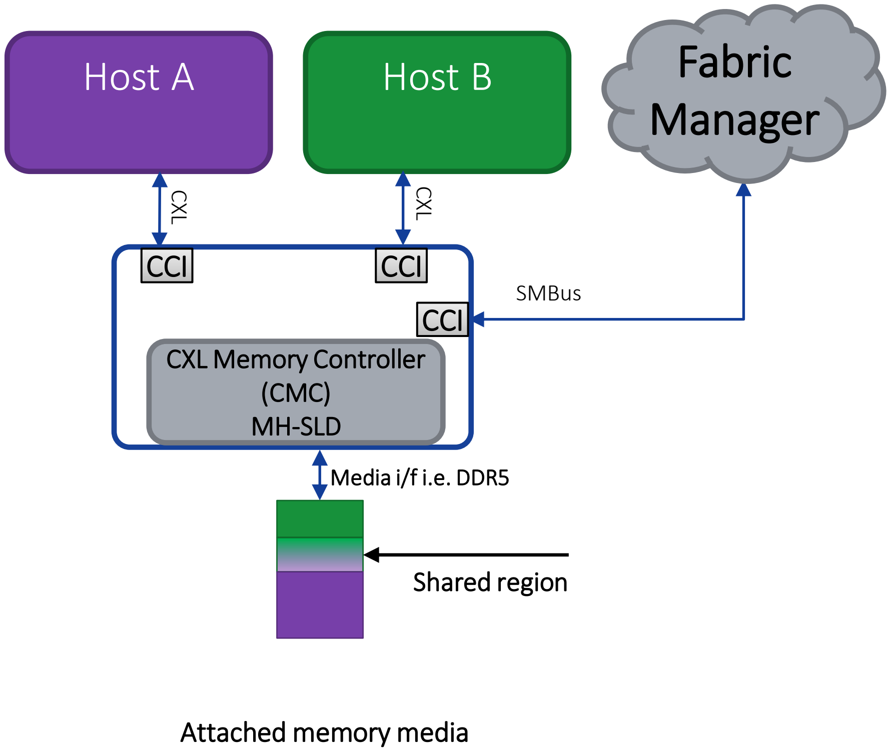
Pooling using MH-SLD

- SLD – Single Logical Device
- MLD – Multi-Logical Device
- MH-SLD – Multi-Headed Single Logical Device
- Pooling: Memory capacity is partitioned amongst hosts.
- It helps with stranded memory capacity.
- Each host experience should be as if it is connected to SLD.
- One host should not affect the BW/user-experience of other host drastically.



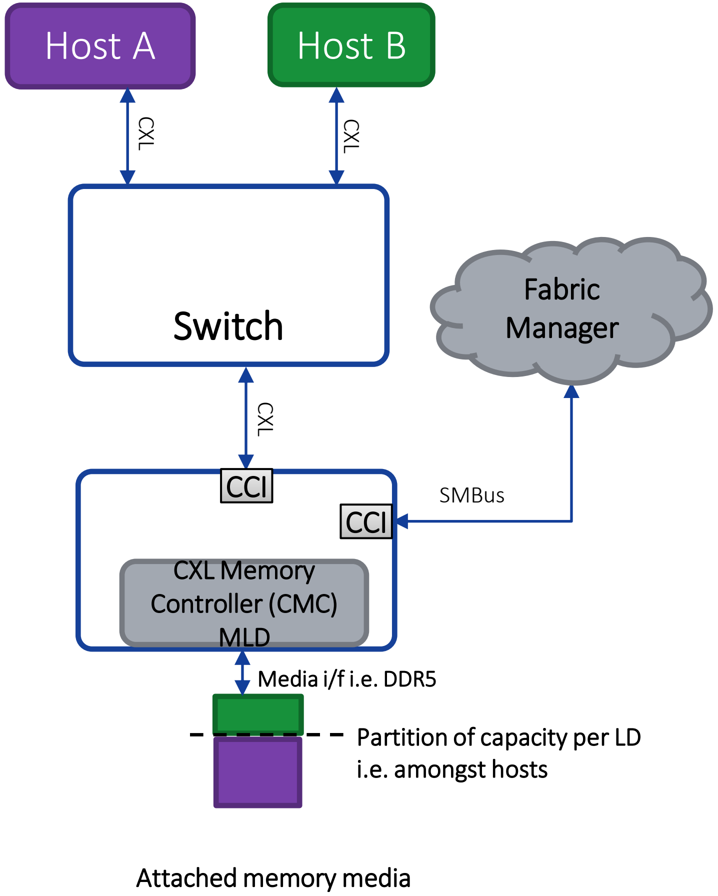
Sharing using MH-SLD

- Sharing is used when hosts need to work with common code/datasets amongst them.
- Sharing removes the need to copy the information from one host to another.
- Sharing requires coherency to be maintained amongst the requesters.
- Coherency could be done in hardware or in software.



Pooling/Sharing using MLD (Multi-Logical Device)

- MLD is connected to a switch which connects to more than one host.
- Example shows 2 hosts.



MH-SLD vs MLD

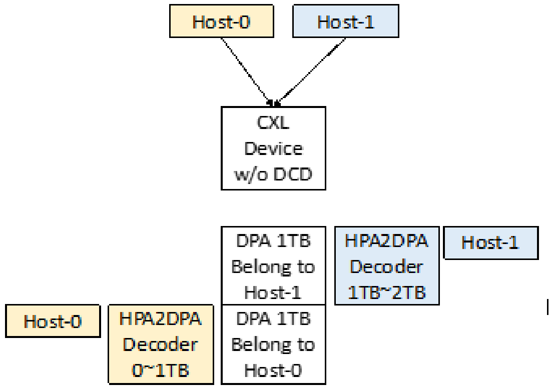
Item	MH-SLD	MLD
1	MH-SLD presents a single LD to each head – 1-1 mapping of LDs to heads	Presents a MLD to switch/FM – specification allows upto 16 LDs Host logically observes its LD as if it is directly connected to it.
2	MH-SLD works with and without switches	MLD requires a switch between a Host and a Type-3 device
3	Lower latency - as no switch is required	Higher latency due to presence of a switch
4	Provides more options, for BW sharing between different hosts i.e. Jedec CMC01 defined "Dual Port - Divided" Address Mode.	Same port's BW is divided amongst the hosts so fairness issues need to be assessed.
5	FM is not a must. Device may have fixed configuration.	FM support is must as MLD is configured using FM.
6	RAS is easier - Management is similar to SLD so easier to deploy for 1st generation of devices.	RAS is complicated - As the CXL switches will be 1st generation, need to assess the impact of MLD from host/device/switch interaction perspective for all RAS scenarios including CXL specification architected QoS for MLD. Once ecosystem matures further, MLD may be meaningful.

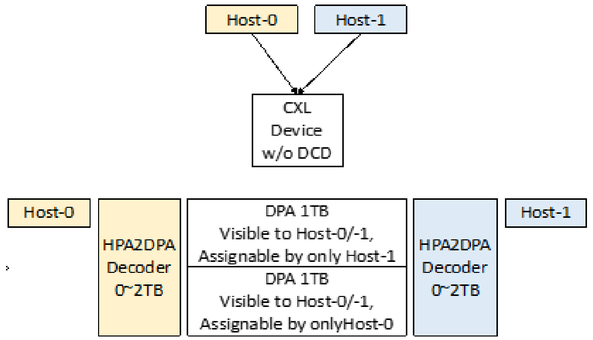
CXL 3.1 High Level Change List

Features	CXL 2.0	CXL 3.0	CXL 3.1	
Memory Pooling	✓			
3/6/12/16 memory interleaving		✓		CXL 2.0 ECN
CXL.cachemem IDE Establishment Flow		✓		CXL 2.0 ECN
NULL CXL Capability ID		✓		CXL 2.0 ECN
Mailbox Ready Time		✓		CXL 2.0 ECN
Vendor Specific Extension to Register Locator DVSEC		✓		CXL 2.0 ECN
CXL Features		✓		CXL 2.0 ECN
Component State Dump Log		✓		CXL 2.0 ECN
Devices Operating in CXL 1.1 mode with no RCRB		✓		CXL 2.0 ECN
surprise hot remove (Error Isolation for .cache/.mem)/ CXL Error Isolation		✓		CXL 2.0 ECN
Type 3 Management Using MCTP CCI" for device management baseline		✓		CXL 2.0 ECN
CXL maintenance command for PPR operation		✓		CXL 2.0 ECN
256-Byte Flit (upto 64 GT/s)		✓		PCIe Gen6
Multi level switching (PBR flit) - Fabric capabilities		✓		
Memory Sharing (256-Byte flit only)		✓		
Back Invalidate (HDM-DB)		✓		
late poison injection		✓		
256 Byte Lopt Flit		✓		
DCD (Dynamic Capacity Device)		✓		
Performance Monitoring		✓		
FM API over mailbox		✓		
Compliance Mode DOE is now required		✓		
TSP (Trusted Security Protocol) for HDM-H			✓	CXL 3.0 ECN
"Device Built-In Test" for Media testing			✓	CXL 3.0 ECN
"Memory Scrub Control" For patrol scrub/ECS control and status logging			✓	CXL 3.0 ECN
Extended Meta Data			✓	CXL 3.0 ECN
Direct P2P CXL.mem for accelerators			✓	CXL 3.0 ECN
Capacity Reduction at boot time			✓	CXL 3.0 ECN

Pooling/sharing using MH-SLD with DCD

- DCD (section 9.13.3) of CXL 3.1 specification
  - Allows for dynamic allocation of memory capacity from one host to another without the need to reprogram the HDM decoders
  - Prior to the definition of DCD
    - Allocating and deallocating memory was very disruptive
    - For the new capacity to be utilized by the host, traffic must be quiesced; HDM decoder changes would be done to access newly added capacity
  - Requires device/FM/orchestrator coordination for optimized usage of memory capacity
  - Initial use-cases can start with simple configurations instead of full DCD implementation
    - A simple algorithm could be implemented in the device itself for DCD's capabilities and for host memory allocations




Host-0
Host-1
CXL Device w/o DCD
Host-0
HPA2DPA Decoder 0~2TB
DPA 1TB Visible to Host-0/-1, Assignable by only Host-1
DPA 1TB Visible to Host-0/-1, Assignable by onlyHost-0
HPA2DPA Decoder 0~2TB
Host-1

DCD feature makes Pooling/Sharing easier

- DCD region has Flags, which make it easier for host to know the memory capacity attributes i.e. is it Sharable, is it Read-Only region

- Back-Invalidate feature allows for HW managed coherency between multiple requestors
  - Even If BI is not used, sharing using DCD still allows for interesting use-cases i.e. Read-Only regions for large database processing

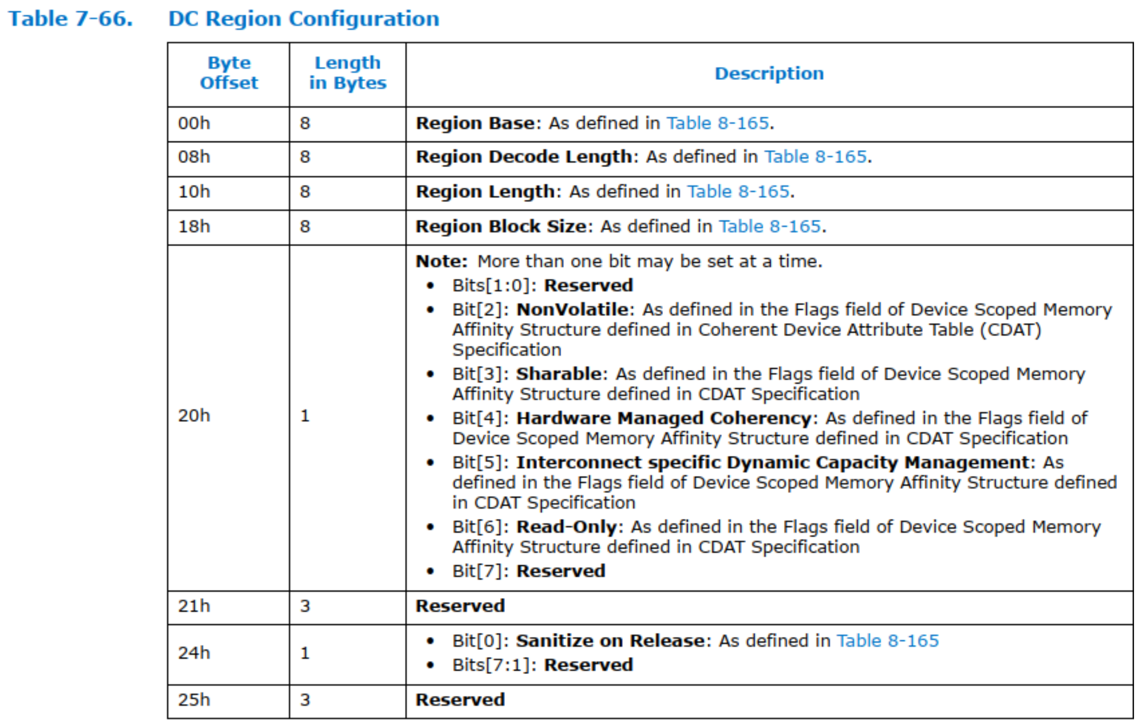


Table 7-66. DC Region Configuration

Byte Offset	Length in Bytes	Description
00h	8	**Region Base**: As defined in Table 8-165.
08h	8	**Region Decode Length**: As defined in Table 8-165.
10h	8	**Region Length**: As defined in Table 8-165.
18h	8	**Region Block Size**: As defined in Table 8-165.
20h	1	**Note:** More than one bit may be set at a time. • Bits[1:0]: **Reserved** • Bit[2]: **NonVolatile**: As defined in the Flags field of Device Scoped Memory Affinity Structure defined in Coherent Device Attribute Table (CDAT) Specification • Bit[3]: **Sharable**: As defined in the Flags field of Device Scoped Memory Affinity Structure defined in CDAT Specification • Bit[4]: **Hardware Managed Coherency**: As defined in the Flags field of Device Scoped Memory Affinity Structure defined in CDAT Specification • Bit[5]: **Interconnect specific Dynamic Capacity Management**: As defined in the Flags field of Device Scoped Memory Affinity Structure defined in CDAT Specification • Bit[6]: **Read-Only**: As defined in the Flags field of Device Scoped Memory Affinity Structure defined in CDAT Specification • Bit[7]: **Reserved**
21h	3	**Reserved**
24h	1	• Bit[0]: **Sanitize on Release**: As defined in Table 8-165 • Bits[7:1]: **Reserved**
25h	3	**Reserved**

Thank you

Back-up

Advantages of Pooling -> many papers have been published like this Pond/ASPLOS23 paper