

263-3855-00: Cloud Computing Architecture

Section 11: Emerging Technologies in the Cloud

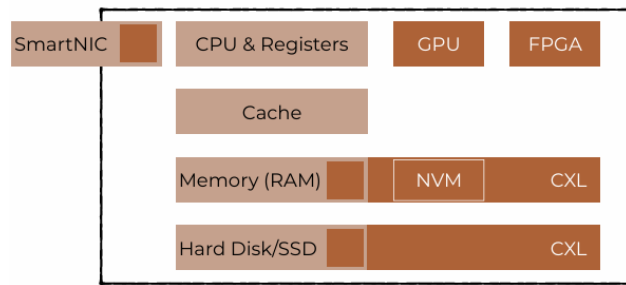
Swiss Federal Institute of Technology Zurich (ETH Zürich)

Last Edit Date: 05/31/2025

Disclaimer and Term of Use:

1. We do not guarantee the accuracy and completeness of the summary content. Some of the course material may not be included, and some of the content in the summary may not be correct. You should use this file properly and legally. We are not responsible for any results from using this file.
2. This personal note is adapted (derived) from the slides and codes from *Professor Ana Klimovic and people from ETH Zurich Systems Group*. Please [contact us](#) to delete this file if you think your rights have been violated.
3. This work is licensed under a [Creative Commons Attribution 4.0 International License](#).

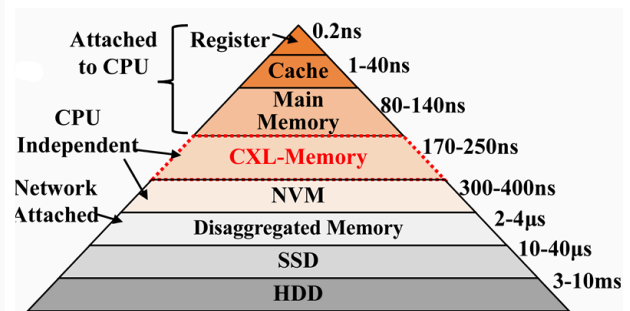
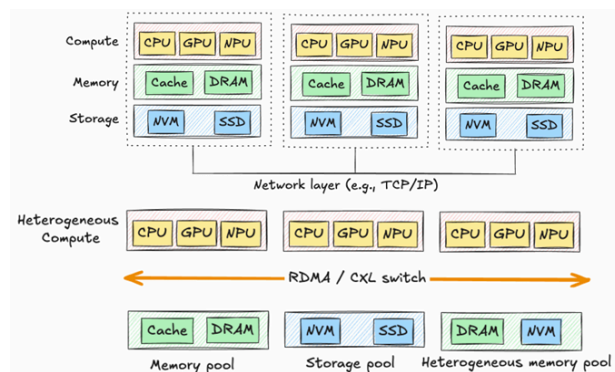
I. Modern architecture - heterogeneous



Each component - memory model, access patterns, specs

Applications - support for performance and correctness

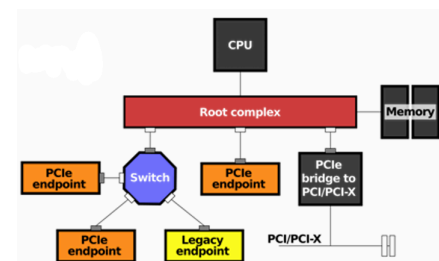
1. Disaggregation



- Shared memory pools accessed by compute node via RDMA / CXL
- Independent scaling of memory and compute
- Improved resource utilization
- Enhanced fault tolerance
- Higher latency than local memory
- Complexity in synchronization and coherence

How are each component connected? PCIe

- PCIe (Peripheral Component Interconnect Express), connects various devices to a computer
- It is a point-to-point connection
- Commonly used for GPUs, SSDs, FPGAs, network cards, etc.



II. CXL - Compute Express Link

- Extend PCIe for heterogeneous computing and server disaggregation usages
- CXL provides a better connectivity to devices

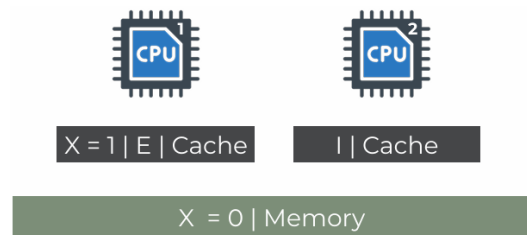
- Protocols
 - CXL.io: based on PCIe, used for device discovery, configuration, initialization, I/O virtualization, and DMA
 - CXL.cache: provides coherent access and caching to the CPU's memory
 - CXL.mem: enables CPU coherent access to device memory

1. CXL Devices

- **Type 1** (CXL.io and CXL.cache): accelerators rely on the CPU memory for operating
- **Type 3** (CXL.io and CXL.mem): memory expansions, persistent memory, and memory of other devices
- **Type 2** (CXL.io, CXL.cache, and CXL.mem): accelerators that have memory and can access the CPU memory or grant access to the CPU on their own memory

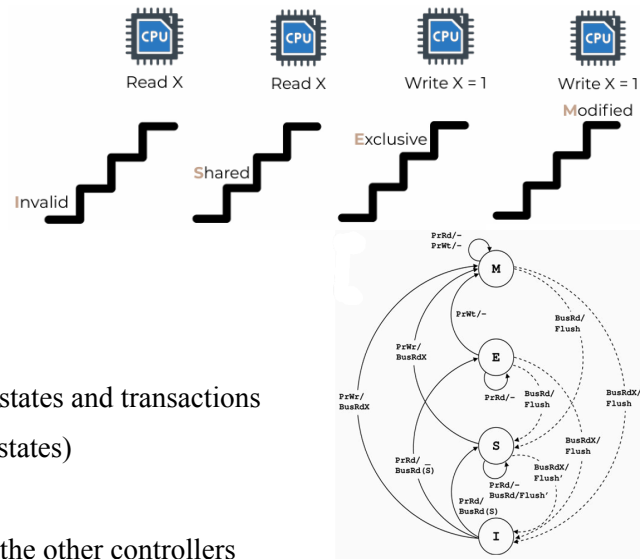
What does coherent mean?

- A CPU that requests X sees the result of the previous modification of X
- Cache Coherence = what coordinates access to memory addresses



The MESI protocol

- Each cache line has a state
- CPUs access cause lines to change states:
 - Modified
 - Exclusive
 - Shared
 - Invalid

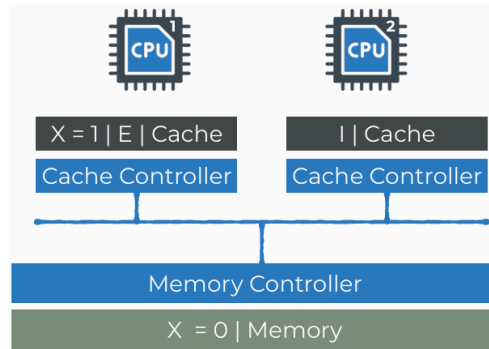


A protocol diagram

- A state machine describing all the possible states and transactions
- More complex (with multiple intermediate states)

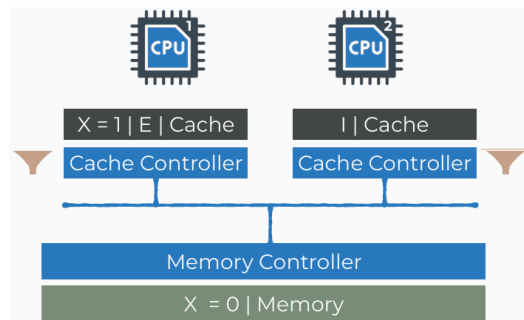
Implementation - Snooping

- To access data, a controller broadcasts it to the other controllers
- Each controller snoops to know what is happening and react accordingly
- The bus serializes all access - big disadvantage



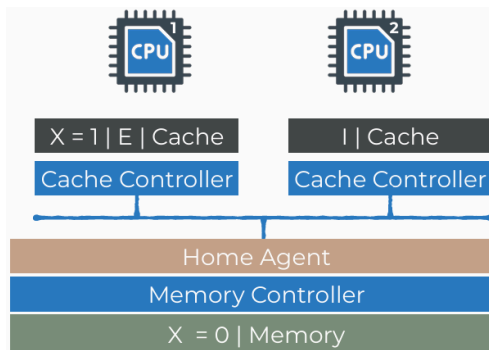
Implementation - Snoop filters

- A snoop filter acts as a “gatekeeper” for bus snooping
- It tracks which caches hold copies of specific cache blocks and only broadcasts snoop requests when they are truly needed



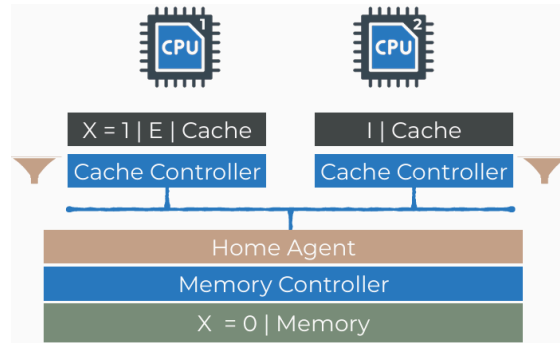
Implementation - directory controller

- Keeps a directory with addresses that are cached
- Useful for direct messages



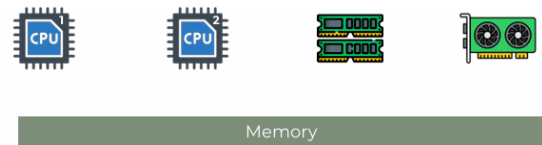
Implementation nowadays

- CPUs nowadays are hybrid



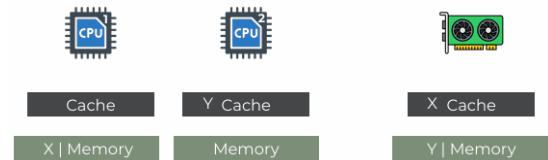
Cache coherence in CXL

- A protocol: tracks memory addresses access
- An interconnect and agents: used to share CPUs requests
- CXL extends the coherence domain beyond CPUs



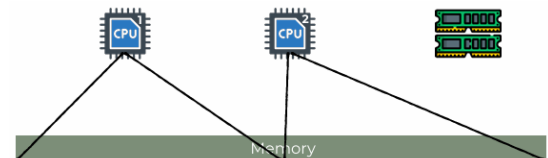
Devices

- Type 1 (CXL.io and CXL.cache)
- Type 2 (CXL.io and CXL.mem)
- Type 3 (CXL.io, CXL.cache, and CXL.mem)



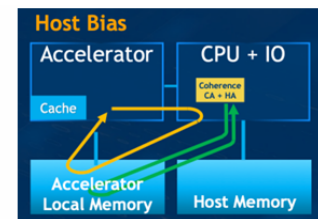
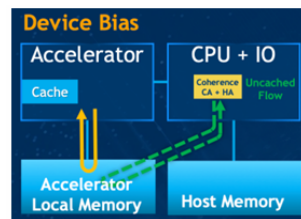
CXL.mem

- Version 1.0: memory expansion
- Version 2.0: memory pooling
- Version 3.2: memory sharing

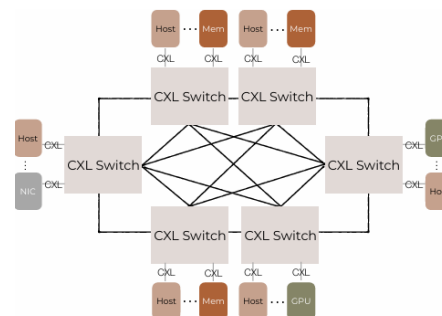


Asymmetric protocol

- The host (CPU) manages the cachelines
- Why? All CPU does coherence anyways
- Keep it simple, backwards compatible and open to multiple devices

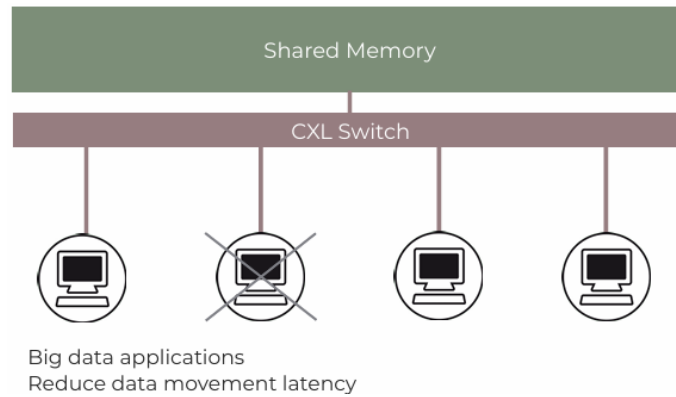


CXL coherent memory sharing



CXL transactions

- Commands for managing cache line states and moving data between host devices
- Cache line granularity for low latency



Challenges

- Incoherent view between the local and shared memory
- Correctness depends on local memory
- Flush data to maintain consistency
- Hard to reason and expensive