STORAGE DEVELOPER CONFERENCE
SDC 22
Fremont, CA
September 12-15, 2022
BY Developers FOR Developers
A SNIA Event

# Software Defined Memory with CXL and Tiered Memory to Enable Hyperscale Use Cases

Manoj Wadekar, Hardware Systems Technologist, Meta
Anjaneya "Reddy" Chagam, Cloud Architect, Intel

# Agenda

- Memory Challenges in Hyperscale Infrastructure
- Need for Software Defined Memory (SDM)
- SDM Use-cases
- SDM on Intel 4th Gen Xeon Processor
- Summary & Call to Action

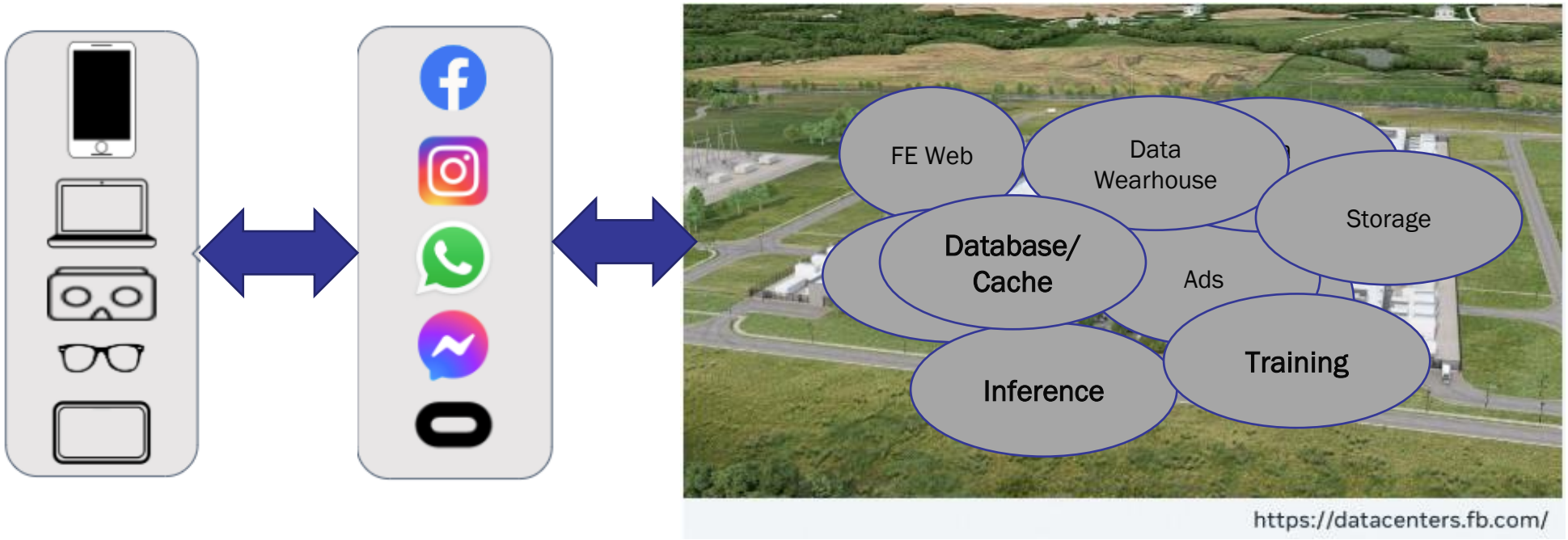
 

# Agenda

- Memory Challenges in Hyperscale Infrastructure
- Need for Software Defined Memory (SDM)
- SDM Use-cases
- SDM on Intel 4th Gen Xeon Processor
- Summary & Call to Action

# Hyperscale Infrastructure




- Application performance and growth depends on
  - DC, System, Component performance and growth
  - Compute, Memory, Storage, Network..
- Focusing on Memory discussion

# Memory Challenges



## Bandwidth and Capacity

- The Gap between bandwidth and capacity is widening
- Applications ready to trade between bandwidth and capacity

## Power

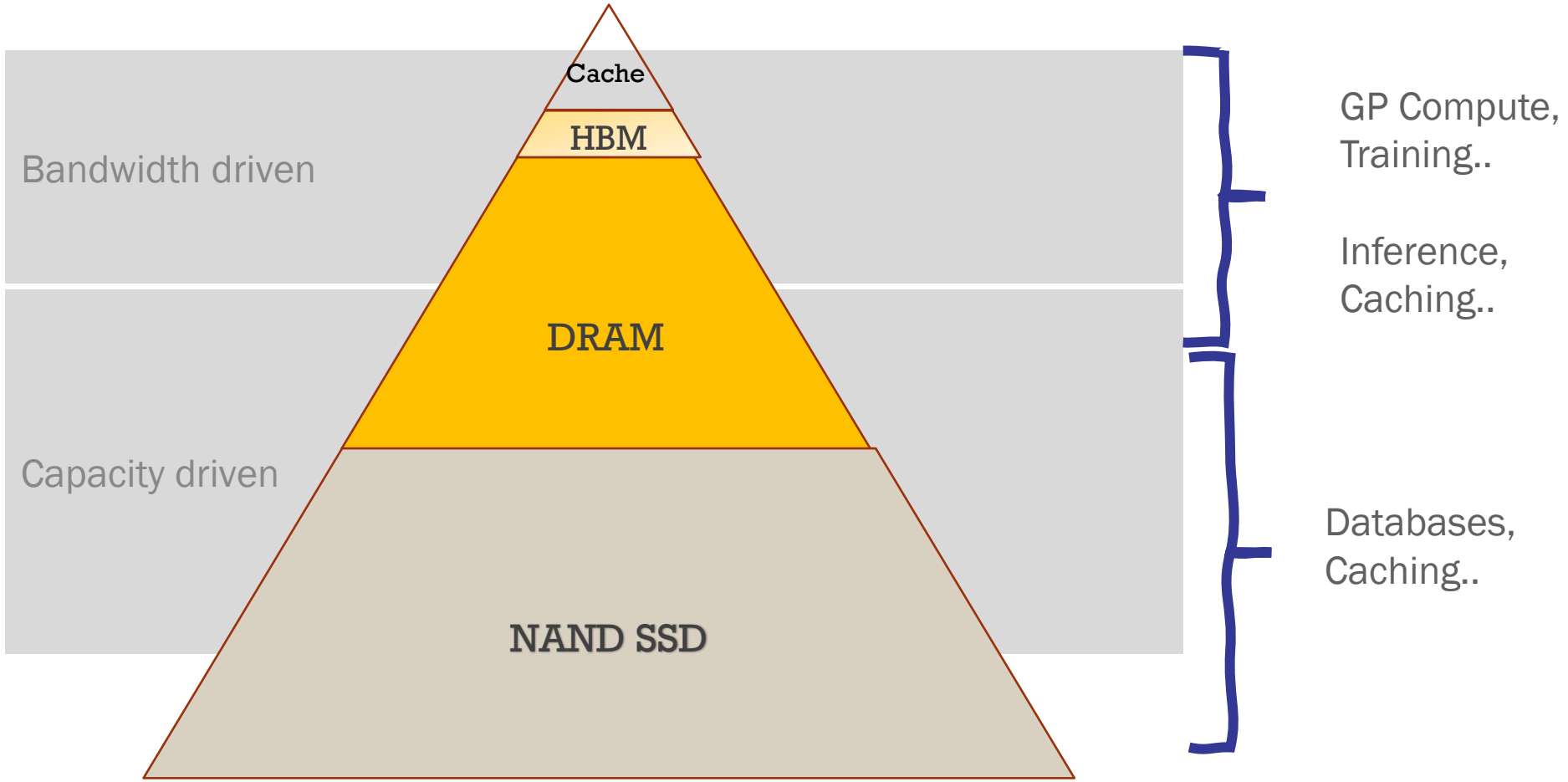
- DIMMs consume significant share of rack power
  - DDR5 exacerbates this
- Applications co-design to achieve higher capacity at optimized power

## TCO

- Cost impact of min capacity increase and Die/ECC overheads
- Applications can trade performance/capacity to achieve optimal TCO

# “Memory” Pyramid today



# Agenda

- Memory Challenges in Hyperscale Infrastructure
- Need for Software Defined Memory (SDM)
- SDM Use-cases
- SDM on Intel 4th Gen Xeon Processor
- Summary & Call to Action

# Hierarchical Memory: Need for SDM

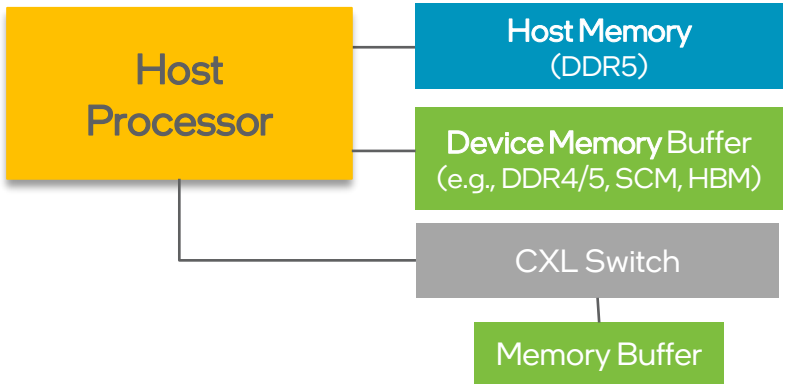
Emerging workloads driving need for **faster data processing**

Need for increased **memory capacity and bandwidth**

**Lack** of **open interconnect standard**

**CXL**

An open industry-supported cache-coherent interconnect for processors, memory expansion and accelerators



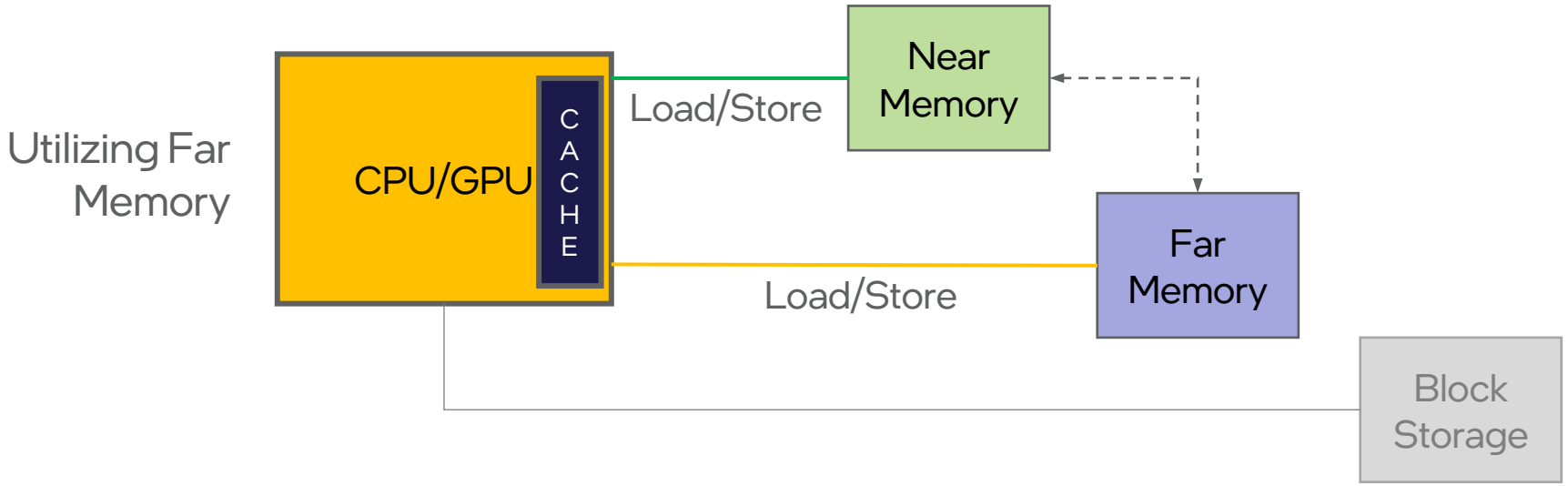

- Hierarchical memory with varying bandwidth/latency characteristics
- Late binding/on-demand provisioning
- Bare-metal v/s virtualized
- Memory Pooling (rack level)
- Kernel managed v/s application managed

**Software-Defined Memory** (SDM) is needed for optimizing resource utilization to deliver lower TCO and power-efficiency

# SDM - Introducing Far Memory

- Memory needs are growing faster than underlying memory technology
- Tiered memory can provide additional capacity at appropriate performance (with load/store) to sustain application needs
- Initial SDM opportunity is around CXL 1.1 Memory expansion for workload acceleration and lower costs with CXL connected DRAM




*CXL Memory tier can enable performance/capacity trade off to achieve TCO gains*

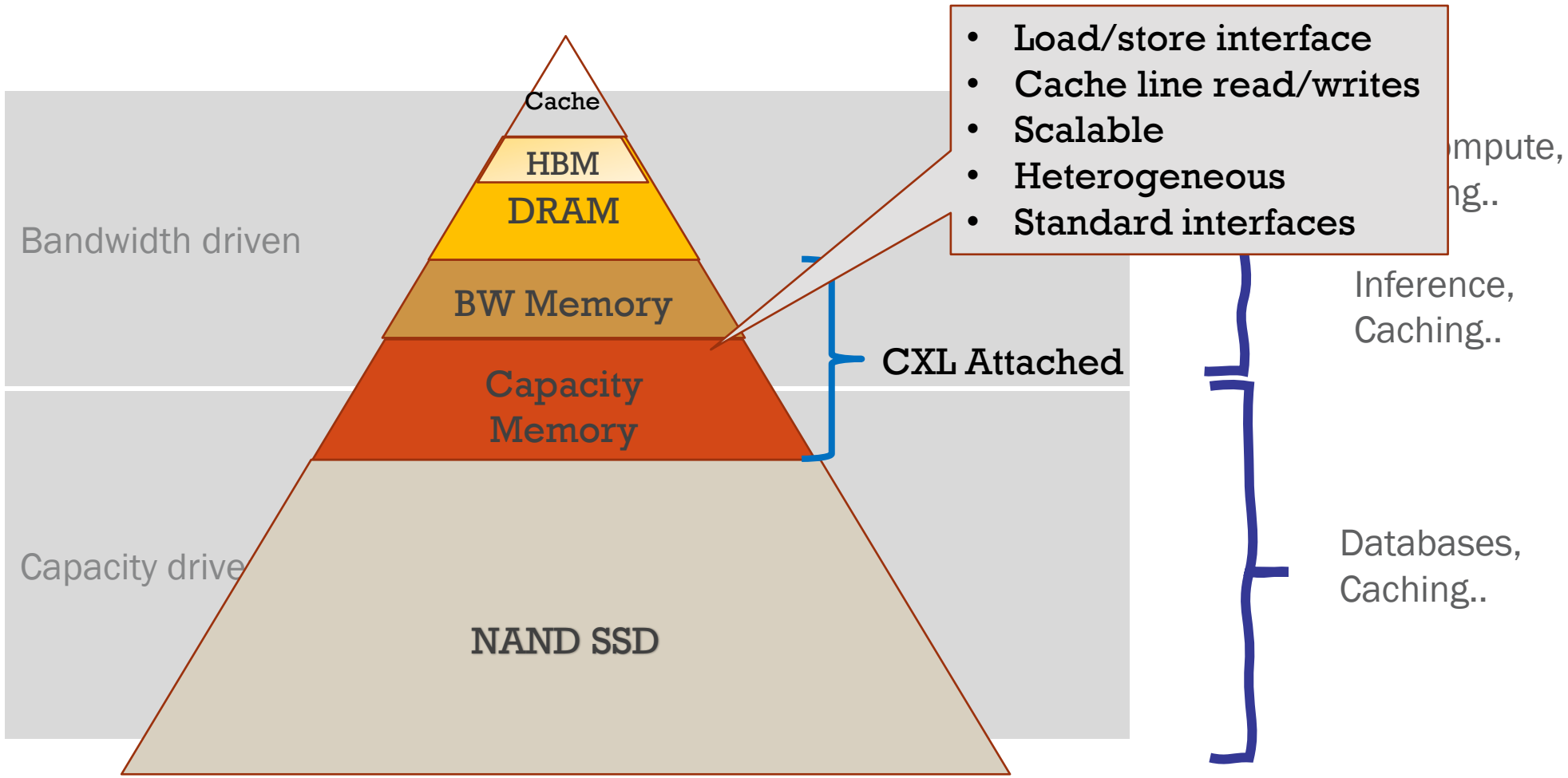
 

# Agenda

- Memory Challenges in Hyperscale Infrastructure
- Need for Software Defined Memory (SDM)
- SDM Use-cases
- SDM on Intel 4th Gen Xeon Processor
- Summary & Call to Action

# “Tiered Memory” Pyramid with CXL



# Memory Technologies

| | Compute | Storage | Training | Inference |
|---|---|---|---|---|
| DDR4 | ☑ | ☑ | ☑ | ☑ |
| DDR5 | ☑ | ☑ | ☑ | ☑ |
| HBM | | | ☑ | ☑ |
| CXL+DDR | ☑ | ☑ | | ☑ |
| SCM (PCIe/CXL) [Exploration Phase] | | ☑ | ☑ | ☑ |

# Use Case Examples

- **Caching (e.g. Memcache/Memtier (Cachelib), Redis etc.)**
  - Need to achieve higher QPS while satisfying “retention time”
  - Higher memory capacity needed
  - Current solutions include ”tiered memory” with DRAM+NAND, but need load/store
- **Databases (E.g. RocksDB/MongoDB etc.)**
  - Need to achieve efficient storage capacity per node and deliver QPS SLA
  - Higher amount of memory enables more storage per node
- **Inference (E.g. DLRM)**
  - Petaflops and Number of parameters are increasing rapidly
  - AI Models are scaling faster than the underlying memory technology
  - Current solutions include ”tiered memory” with DRAM+NAND, but need load/store

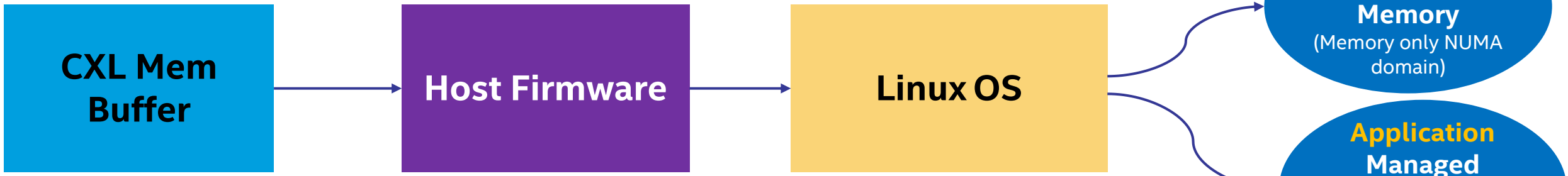
 

# Agenda

- Memory Challenges in Hyperscale Infrastructure
- Need for Software Defined Memory (SDM)
- SDM Use-cases
- SDM on Intel 4th Gen Xeon Processor
- Summary & Call to Action

# Intel 4th Gen Xeon Processor CXL Memory Buffer Provisioning




**CXL Type2 Device**
w/ Host-managed Device Memory (HDM)

- **Setup Knobs**
  - **DfxCxlType3LegacyEn (Enable)**
  - Port bi-furcation
  - CXL security level etc.
- **CXL Training**
- **SAD Programming**
- **ACPI Table Setup**
  - CEDT, SRAT, HMAT etc.
- **CXL Memory Map**
  - Special Purpose (EFI_MEMORY_SP)
  - Conventional (EFI_MEMORY_WB)

- **CXL Memory Configs**
  - Conventional (Memory only NUMA domain)
  - HMEM/DAX device (/dev/dax<x>.<y>)
- **DAX Memory Config Knobs**
  - **efi=nosoftreserve** kernel command line
  - **daxctl** to promote/demote to system ram
- **Kernel Tiering for Conventional Memory**
- **CXL Mailbox Driver**
- **CXL Command Line Tools**

***ACPI SLIT (2S, CXL)***

```
[02Ch 0044   3] Locality   0 : 0A 15 18
[02Fh 0047   3] Locality   1 : 15 0A 0E
[032h 0050   3] Locality   2 : 18 0E 0A
```

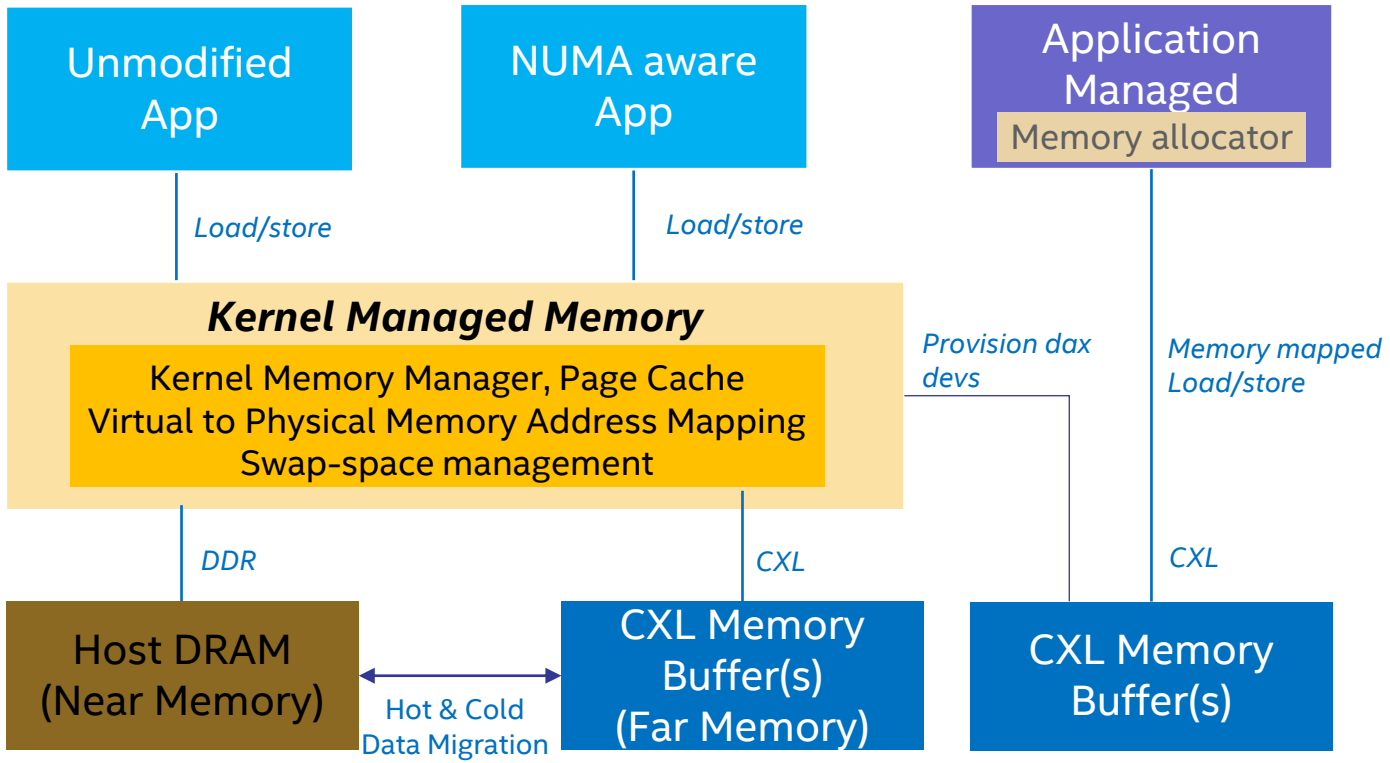
***numactl -H***

```
node   0   1   2
  0:  10  21  24
  1:  21  10  14
  2:  24  14  10
```

**mm/migrate.c:** builds **node_demotion** list based on SLIT. Critical data structure used for data migration decisions

# SDM - Kernel & Application Managed Memory




## Kernel Managed

- OS can **map far memory** into application's **virtual address space**
- Applications **can execute from pages in far memory** (albeit more slowly)
- Kernel memory manager can implement **varying policies for migrating hot & cold pages** between tiers

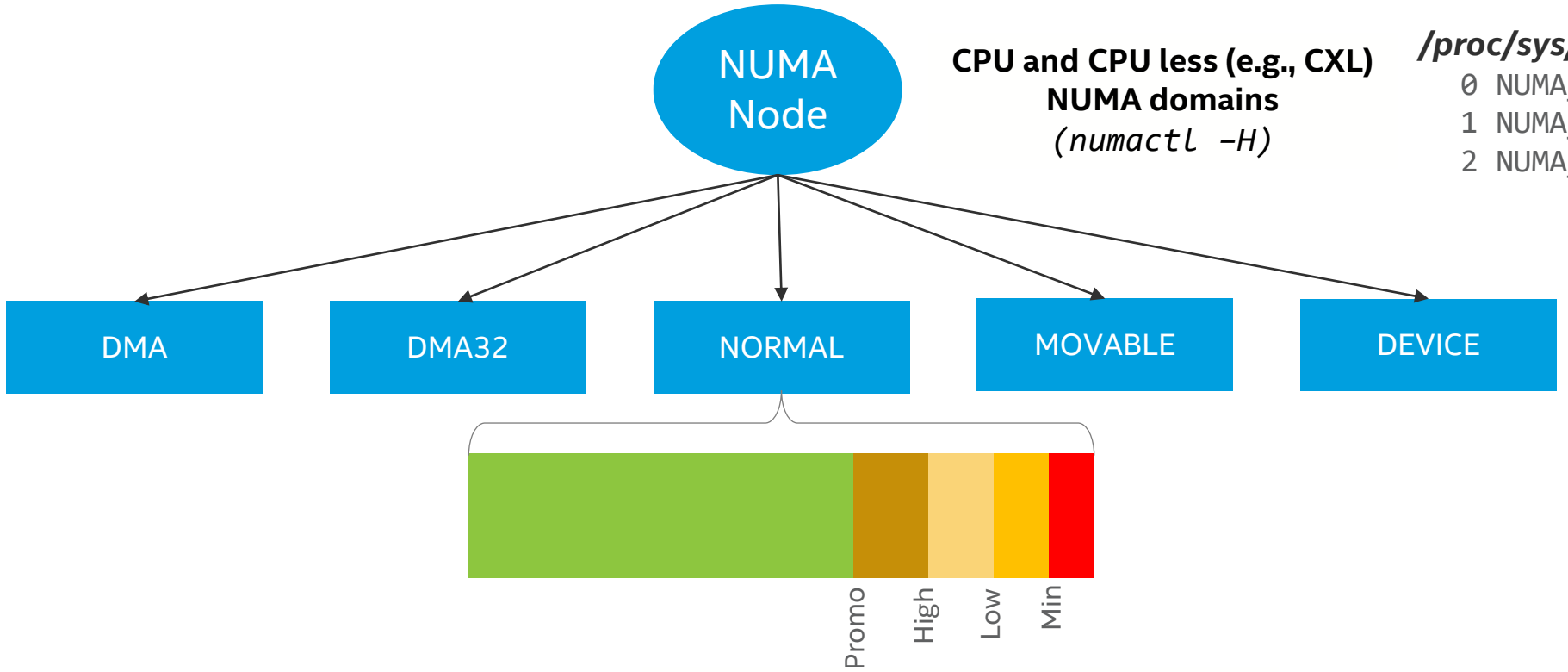
## Application Managed

- **Allows apps** to access CXL **memory as memory-mapped files**
- **Built on top of DAX** (Direct Access) file system

*Linux Kernel tiering in early development stages*

# Linux Kernel – Memory Layout




**CPU and CPU less (e.g., CXL) NUMA domains**
*(numactl -H)*

***/proc/sys/kernel/numa_balancing***

```
0 NUMA_BALANCING_DISABLED
1 NUMA_BALANCING_NORMAL
2 NUMA_BALANCING_MEMORY_TIERING
```

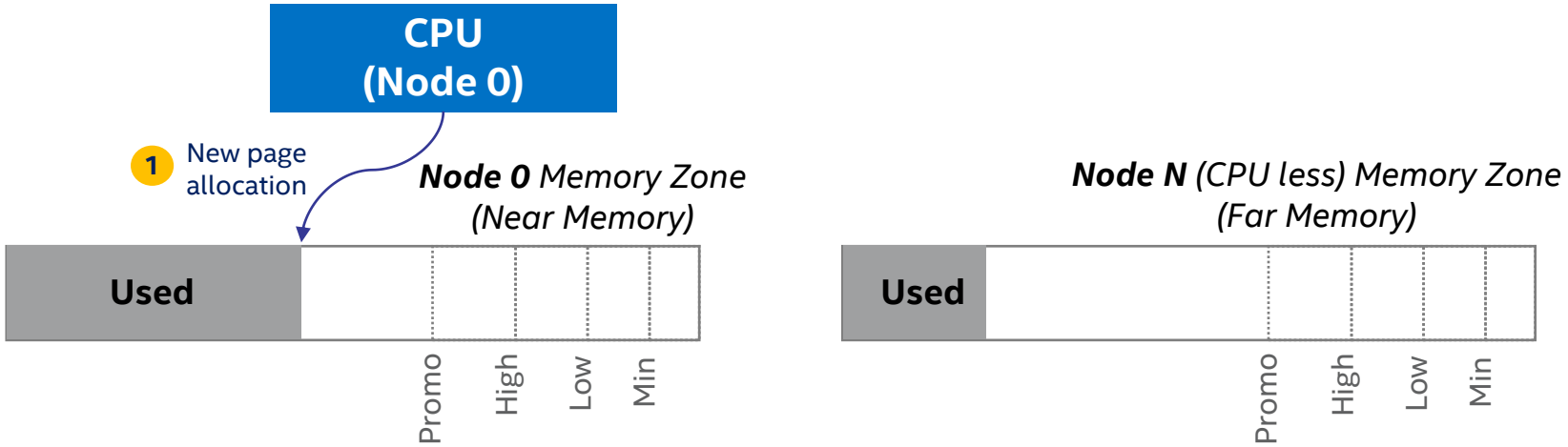
**ZONES**
*(/proc/zoneinfo)*

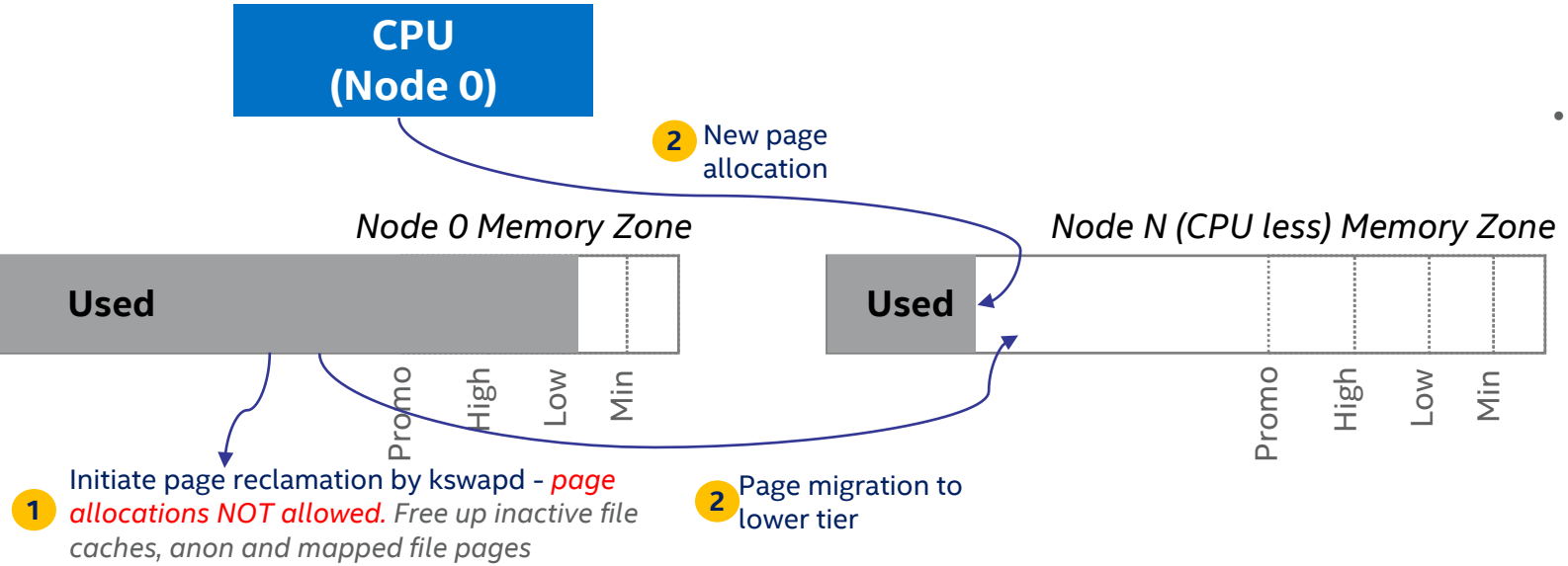
**Per-Zone Watermarks**
(for placement/reclamation)
*(/proc/sys/vm/watermark_scale_factor)*

- **Low** - when #of free pages reaches **low** watermark, kswapd is woken up to start freeing pages
- **Min** - when #of free pages reaches **min** watermark, page allocator will do the kswapd work in a synchronous fashion (i.e. direct-reclaim path)
- **High** – zone is not "balanced" until #of free pages reaches high watermark and considered for placement
- **Promo** - reclamation watermark to keep enough buffer for placement and promotion from lower tiers. Once the watermark has been reached, kswapd will go back to sleep. *NOTE: Enabled only when /proc/sys/kernel/numa_balancing is NUMA_BALANCING_MEMORY_TIERING (2)*

# Linux Kernel – Placement/Reclamation

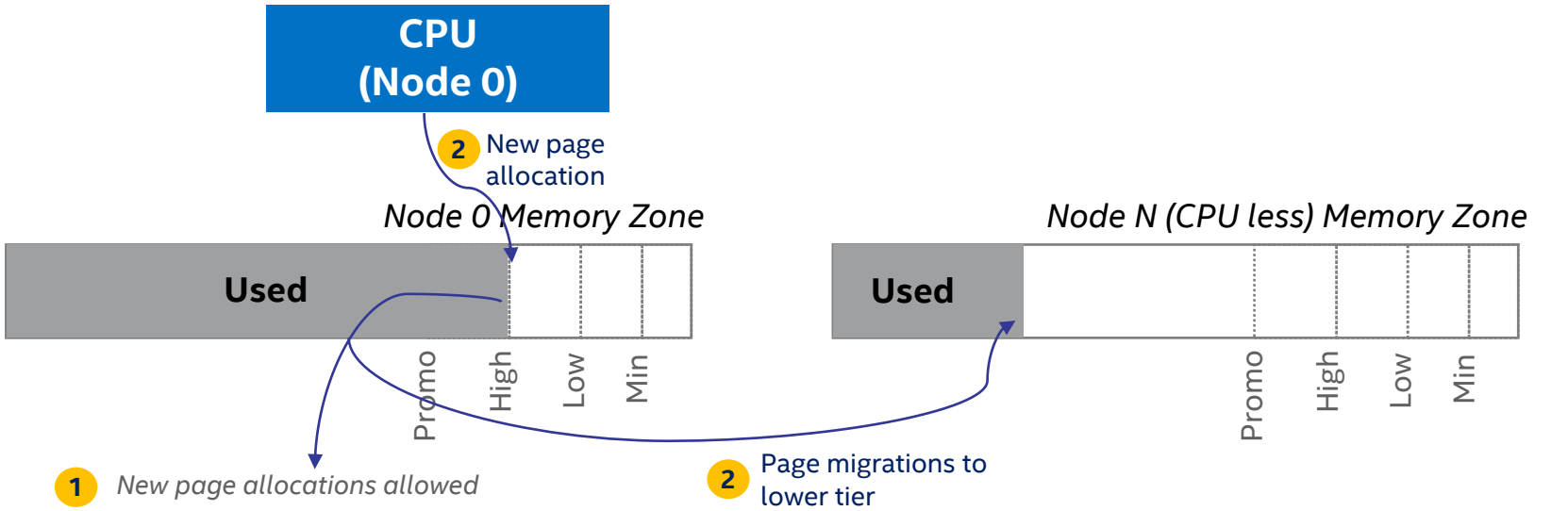


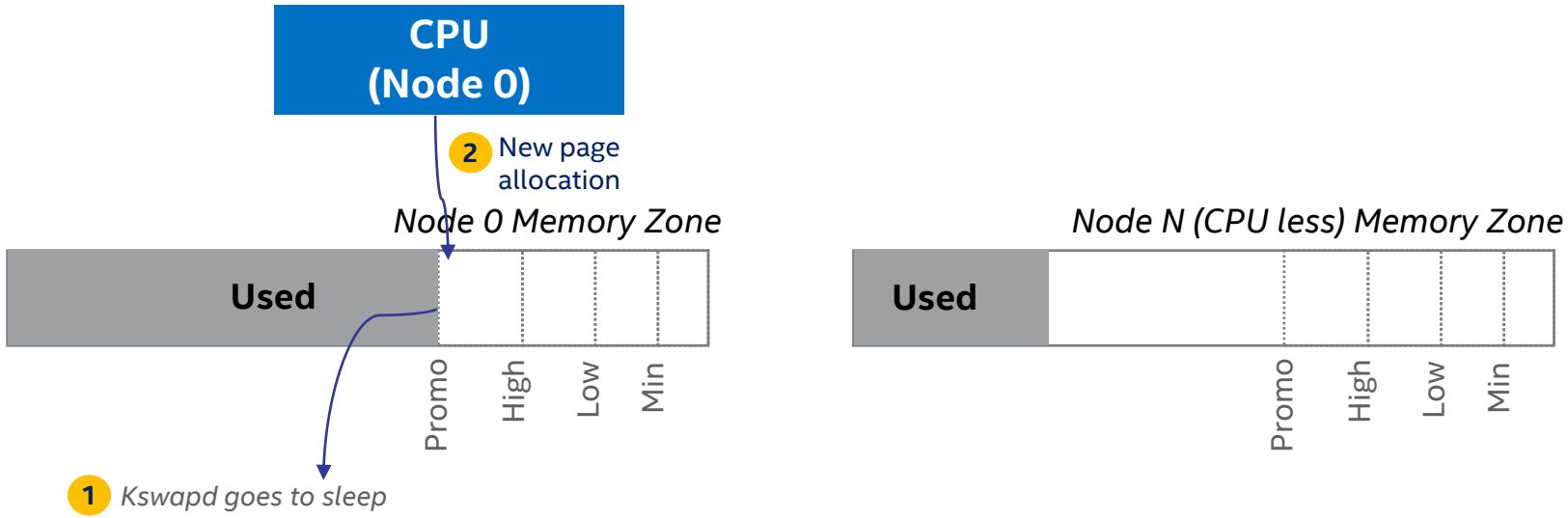




- **/sys/kernel/mm/numa/demotion_enabled** needs to be enabled (default is false) to demote pages to lower tier during reclamation
- **numastat** to show per-NUMA-node memory statistics for processes and the operating system

# Linux Kernel – Placement/Reclamation (Contd)

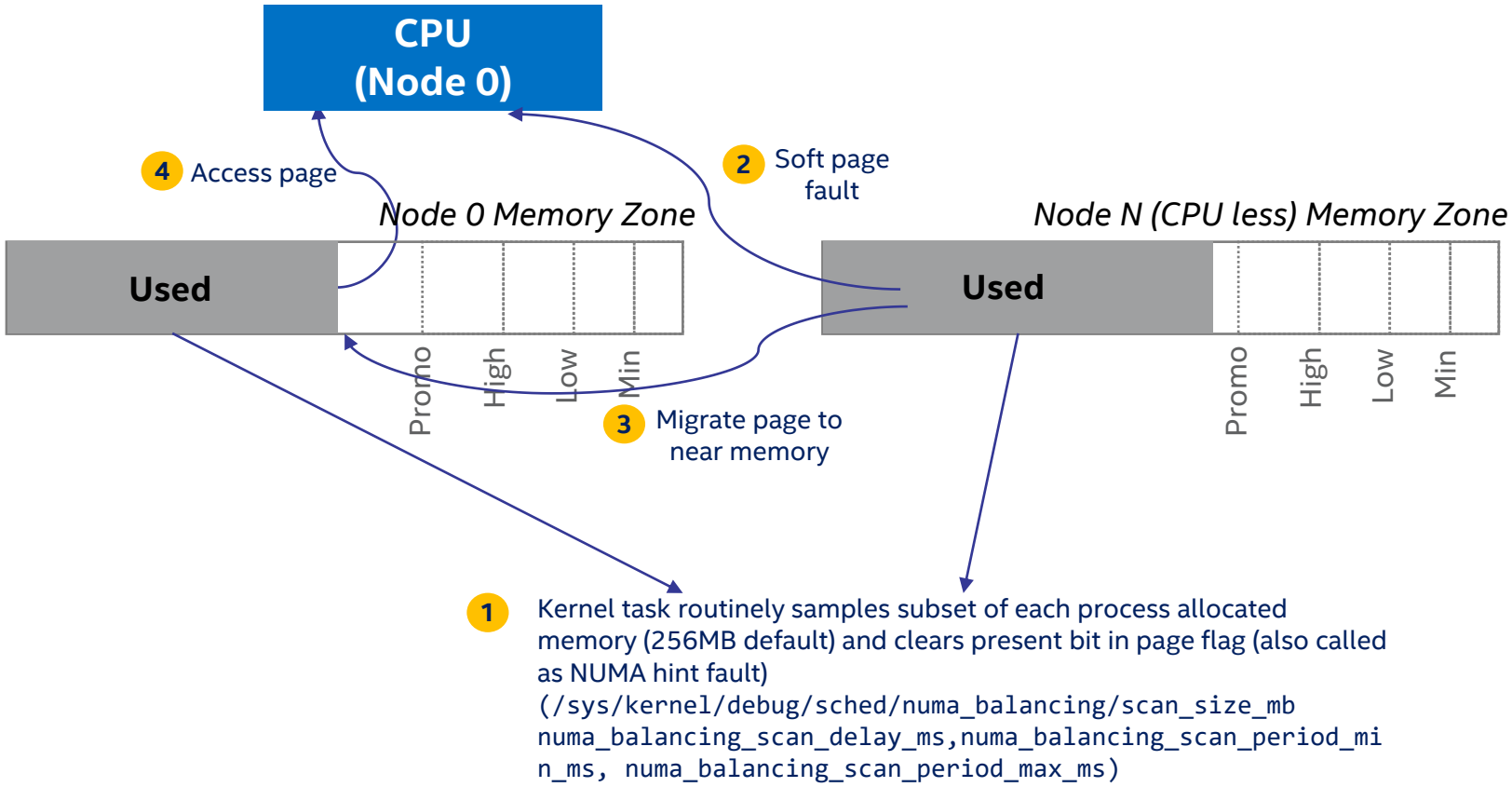



- **stats: /proc/vmstat**
  - **numa_pages_migrated** - # of pages migrated among tiers which includes inter-socket NUMA balancing as well
  - **numa_pte_updates**: # of pages marked as inaccessible which are cleared later by a NUMA hinting faults
  - **numa_huge_pte_updates** - # of huge pages marked for NUMA hinting faults
  - **numa_hint_faults** - #of NUMA hint faults)
  - **numa_hint_faults_local** - #of NUMA hint faults to local nodes
  - **pgdemote_kswapd/pgdemote_direct** - #of pages demoted to lower node

# Linux Kernel – Promotion



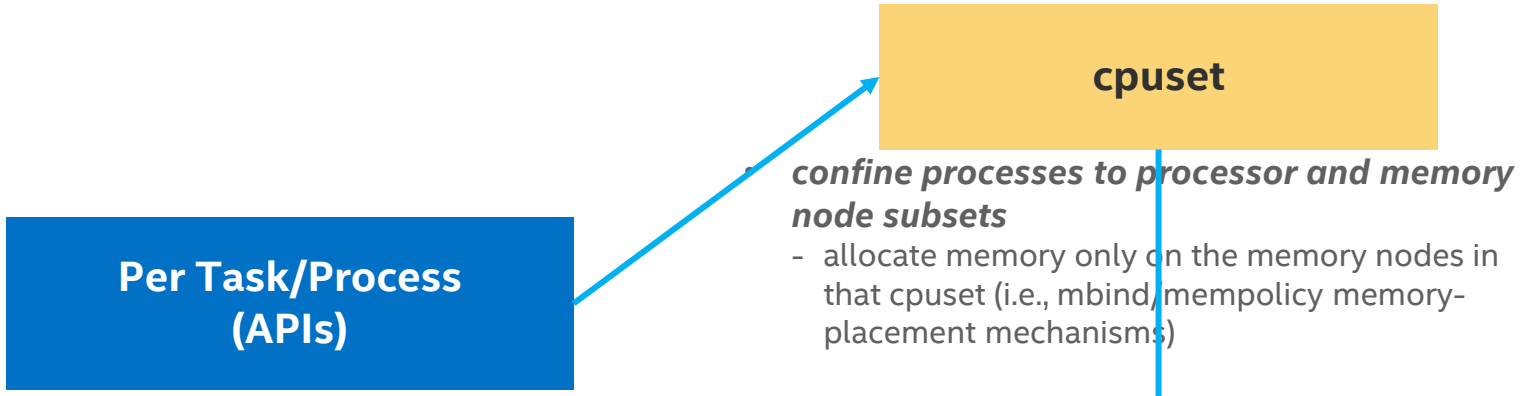

- **stats: /proc/vmstat**
  - **pgpromote_success** - #of promoted pages to top tier node

**Meta TPP patches (pending)**:

- **LRU-filtered promotion** to avoid ping-pong. If the faulted page is in inactive LRU, we do not consider the page as the promotion candidate instantly as it might be an infrequently accessed page. We consider the faulted page as a promotion candidate only if it is found in the active LRUs (either of anon or file active LRU)
  https://github.com/hnaz/linux/commit/8040b0b808b69ba680f53979ca9c5db800b2c8a6
- **Promotion and demotion related statistics:**
  https://lore.kernel.org/lkml/e3db05f4bd7dd9363c2a895875505088d3273bb8.1637778851.git.hasanalmaruf@fb.com/#r

# Linux Kernel – NUMA Memory Policy




- ***set_mempolicy*** *to set NUMA memory policy for a thread and its children*
  - ***MPOL_DEFAULT*** *(local allocation)*
  - ***MPOL_BIND*** *(restrict memory allocation to the nodes specified)*
  - ***MPOL_INTERLEAVE*** *(interleave page allocations among nodes specified)*
  - ***MPOL_PREFERRED*** *(sets preferred node for allocation)*
  - ***MPOL_LOCAL*** *(local allocation)*
- **get_mempolicy** to get memory policy for thread
- **mbind** to set memory policy for a memory range
- **move_pages/ migrate_pages** to move individual or all pages of a process to another node (candidate for DMA offload capability)

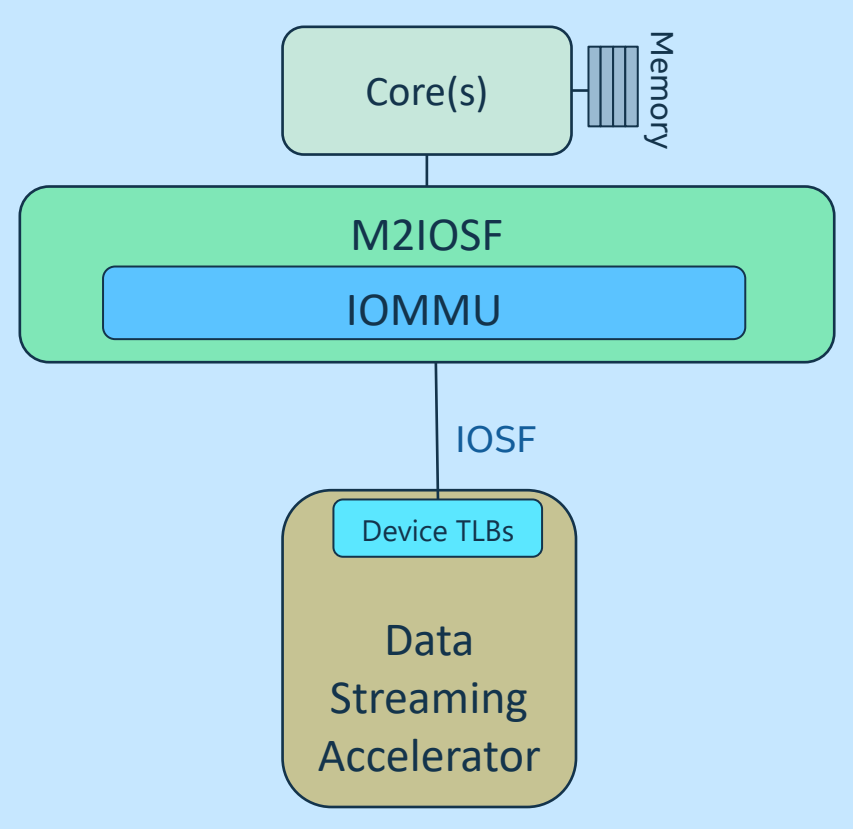
- ***command line option that controls numa policy for processes (NUMA scheduling & memory placement policy)***
  - ***Membind*** *(restrict memory allocation to the nodes specified)*
  - ***Interleave*** *(interleave page allocations among nodes specified)*
  - ***preferred*** *(sets preferred node for allocation)*
  - ***localalloc*** *(local allocation)*

- ***Default policy that governs page allocations not controlled by more specific policies***
  - ***interleave*** *(during boot) &* ***local allocation*** *(OS is up & running)*

**Meta TPP patch (pending) - mm: mempolicy: N:M interleave policy for tiered memory nodes**
*(https://github.com/hnaz/linux/commit/efb59df4f19ec0da7a89131cc23b55658437e72d)*

# Intel 4th Gen Xeon Processor - Data Streaming Accelerator (DSA) Overview

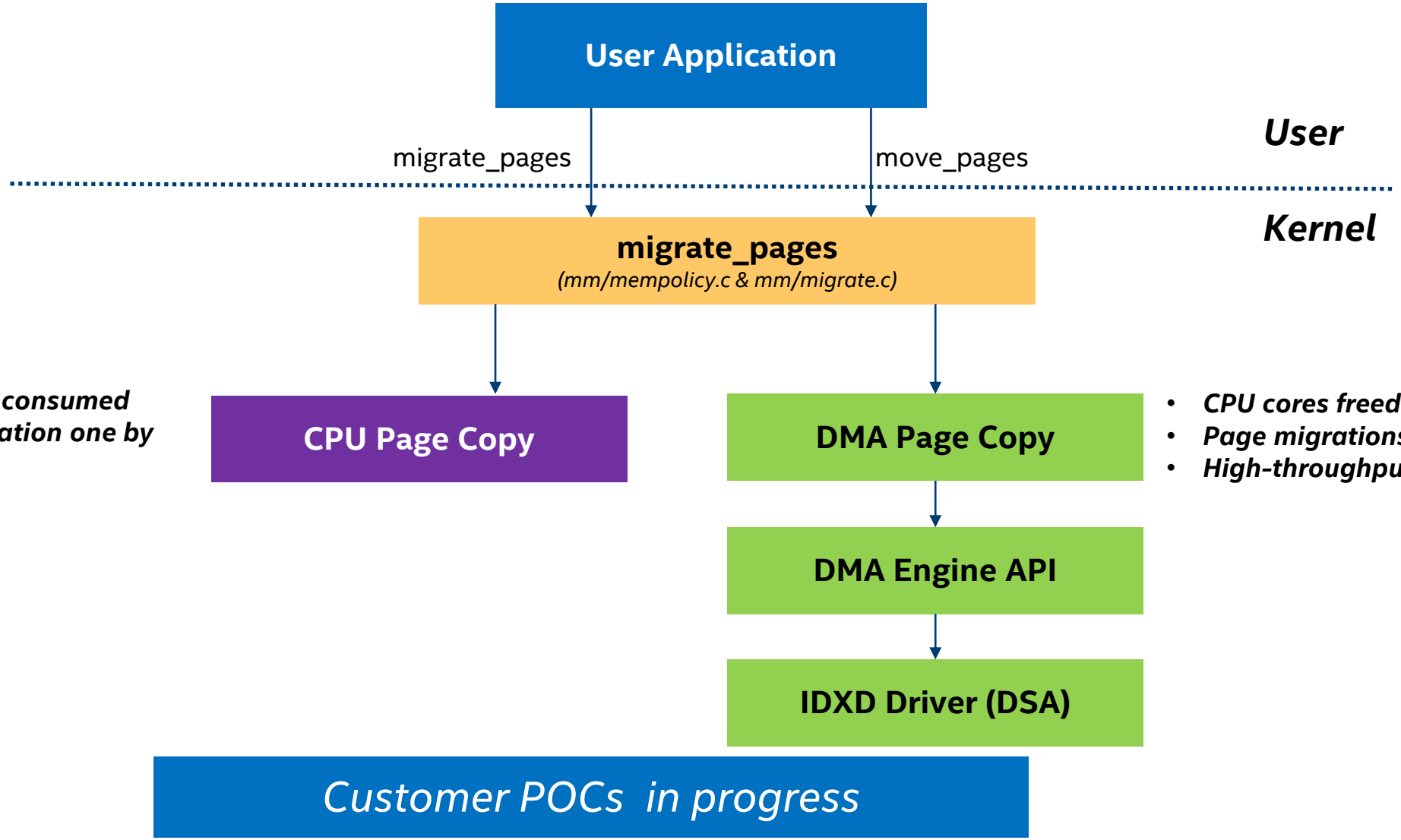



**Logical Representation**

| Capability | DSA 1.0 (SPR) |
|---|---|
| Max Instances per socket | 4 |
| Max Throughput (each direction) | 32 GB/s |
| Shared Virtual Memory support | Yes |
| Low latency dispatch (AIA ISA) | Yes |
| Recoverable page faults | Yes |
| DDIO caching controls | Yes |
| Persistent memory support | Yes |
| Cross-address space support | No |
| Num. WQs & Engines per device | 8 WQs; 4 Engines |
| Virtualization support with SIOV | Yes |
| VM Migration support | Yes |
| Confidential compute (TDX-IO) | No |
| Quality of Service | WQ-Engine Groups, Read buffer controls, VCs for ATS/DRAM/PMEM |
| Data targets supported | DDR, PCI-E, NTB, CXL |

# Linux Kernel – Page Copy Offload using DSA

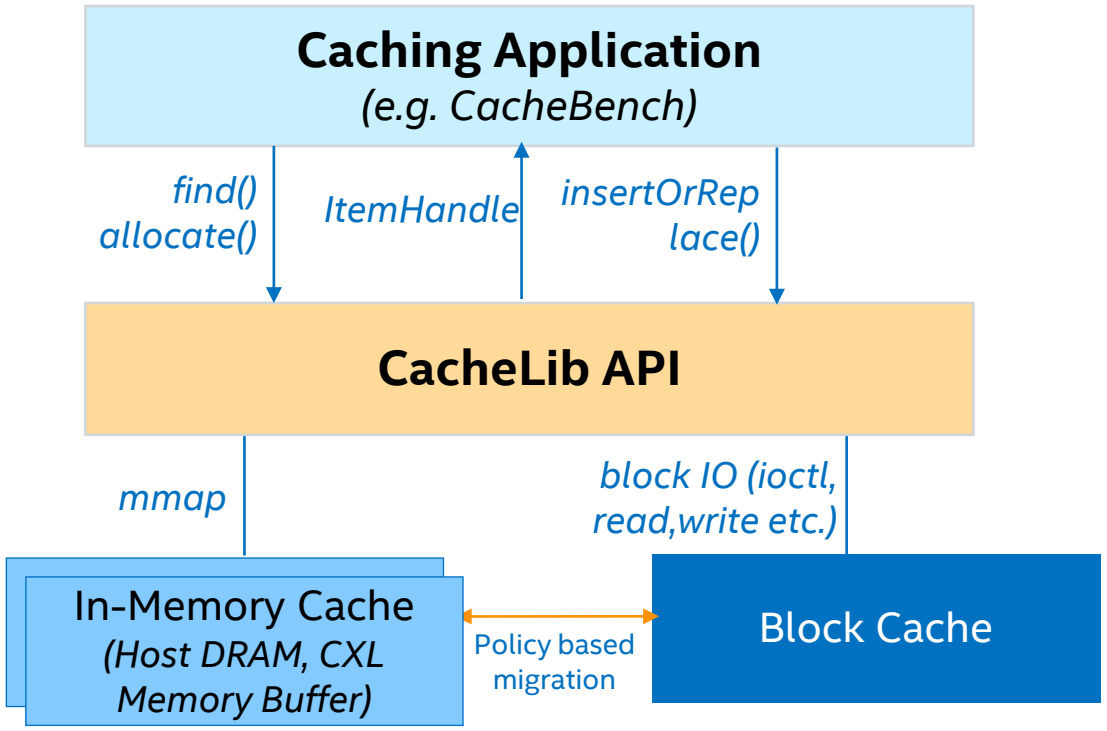



- *CPU cores consumed*
- *Page migration one by one*

- *CPU cores freed up*
- *Page migrations in batch*
- *High-throughput*

# CacheLib - Overview




*Github:* https://github.com/facebook/CacheLib
*Intel development fork:* *https://github.com/intel/CacheLib*

## CacheLib

- **Pluggable in-process open-source caching engine** to build and scale high-performance services
- C++ Library
- Thread-safe API
- Manages DRAM and Block Caching transparently
- Decoupled from underlying medium
- Policy based

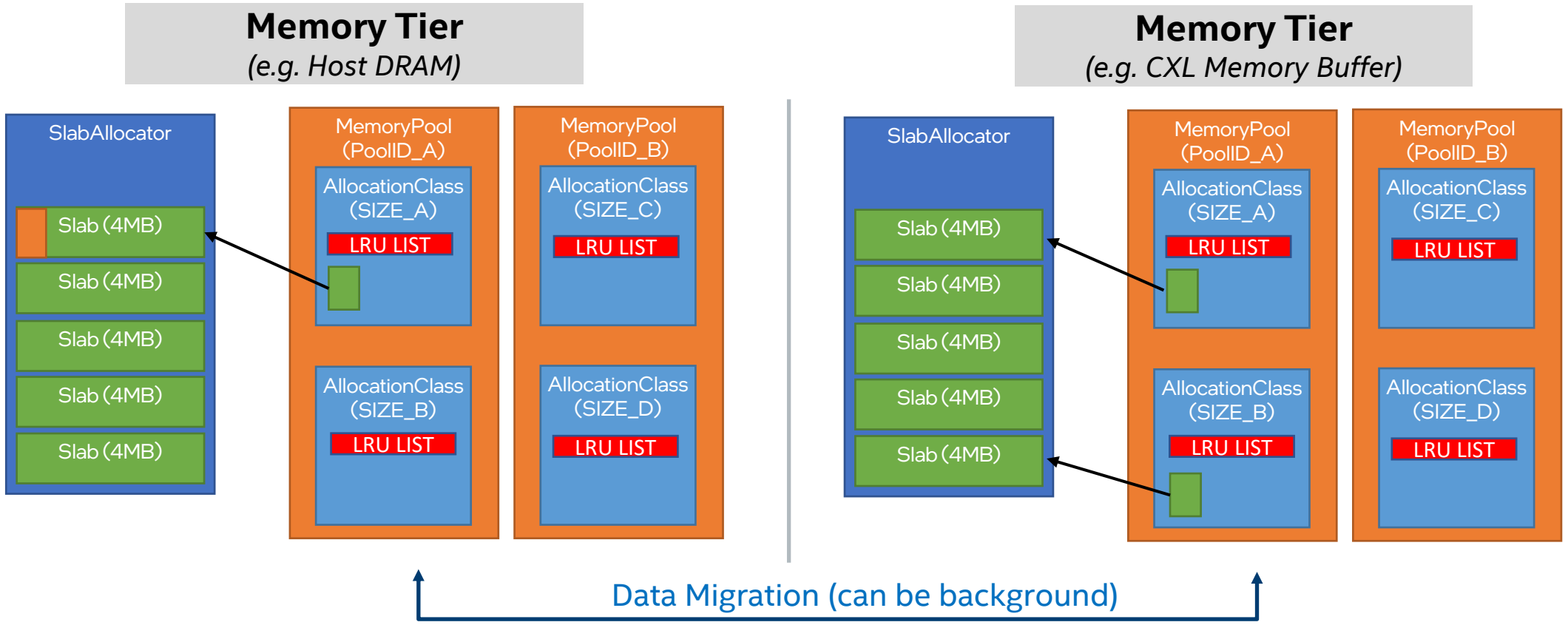
## CacheBench

- **Benchmarking tool** for evaluating caching performance
- **NUMA aware cachelib tiering or Kernel tiering** for **CXL memory buffer benchmarking**

*Memory tiering and DMA (DSA) offload development in progress*

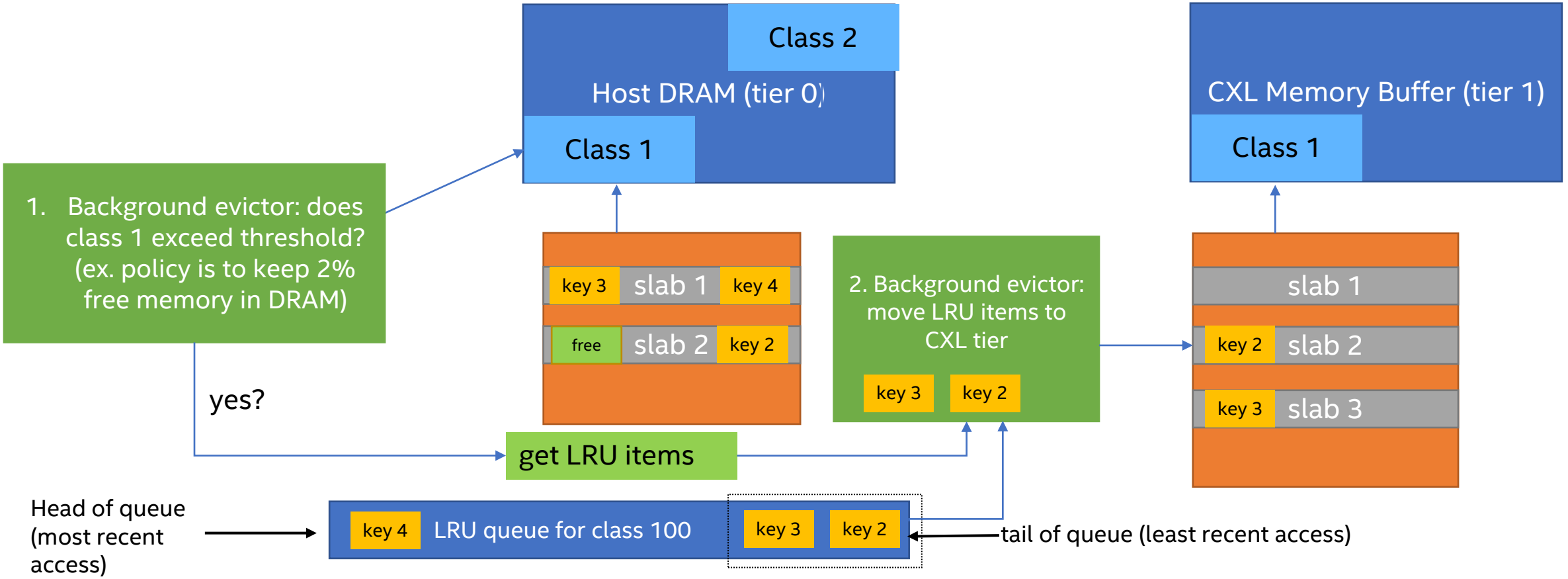
# CacheLib – Memory Tiering (Data Structures)




- Items are initially allocated in tier 1 (DRAM tier) and then migrated to tier 2 (CXL) via eviction (which can be during new item allocation or happen in background)
- Slabs are contiguous chunks of virtual address space
- Slabs are assigned to an allocation class to store items of a given size range
- Pools are independent allocators – an abstraction to support multi-tenancy
- For details refer to Cachelib architecture guide: https://cachelib.org/docs/Cache_Library_Architecture_Guide/overview_a_random_walk

# CacheLib – Memory Tiering (Demotion Example)



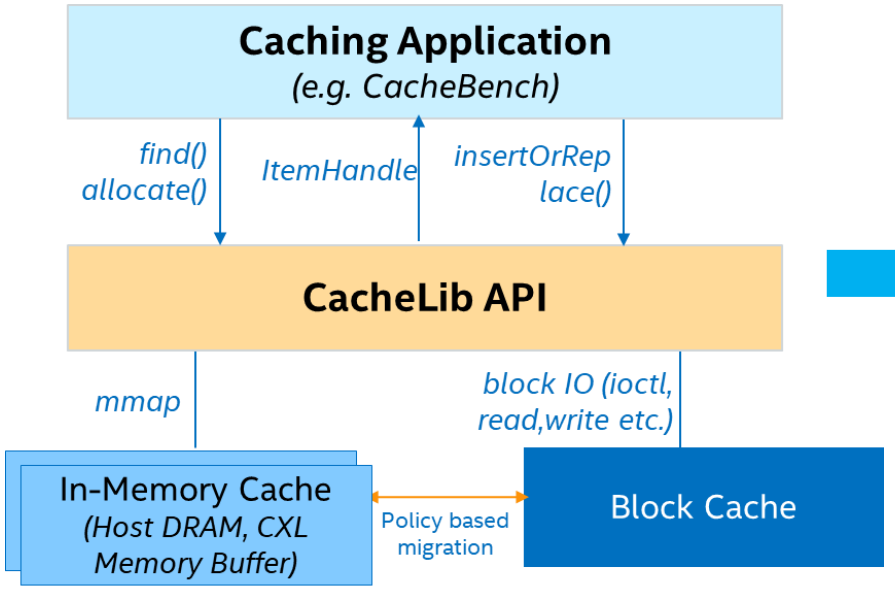

*NOTE: Promotion works the same way but in reverse direction*

*CPU and DSA offload options will be available*

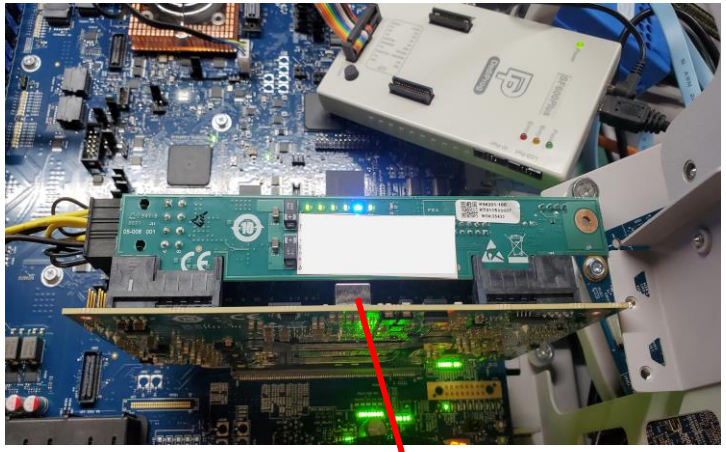
 

# Intel 4th Gen Xeon Processor: Caching Use Case

**CacheBench**

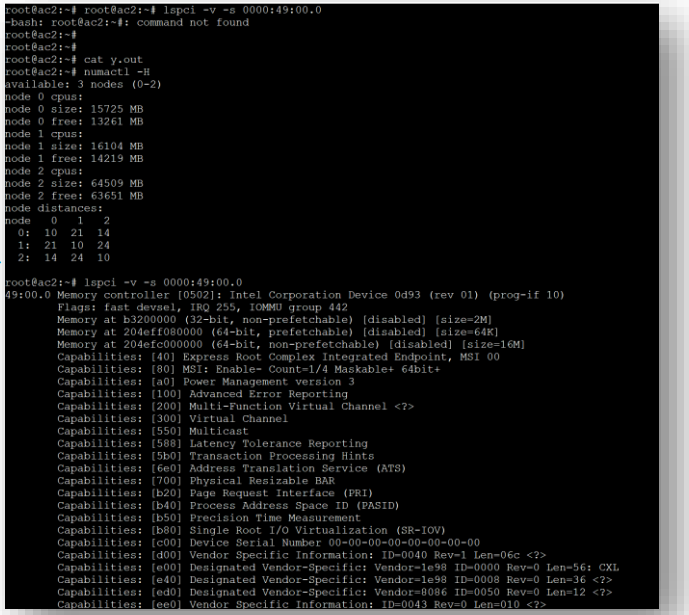



**Intel 4th Gen Xeon Processor Pre-production Platform**



*Intel Pre-production CXL FPGA Memory Buffer*

**CXL Memory in Linux OS**



# Agenda

- Memory Challenges in Hyperscale Infrastructure
- Need for Software Defined Memory (SDM)
- SDM Use-cases
- SDM on Intel 4th Gen Xeon Processor
- Summary & Call to Action

# Summary and Call to Action

- OCP Foundation SDM initiative is focused on applying emerging memory technologies for cloud use cases
- Newer memory technologies (e.g., HBM) and industry standard interconnects (e.g., CXL) are key components of SDM
- Kernel and application tiering provide basic abstraction to underlying memory and storage resources
- Vendors are demonstrating CXL Capable CPUs and devices
- Meta and others are investigating solutions to real world memory problems

**Call to Action: Join OCP SDM workstream!**

# Notices & Disclaimers

Intel technologies may require enabled hardware, software or service activation. Your costs and results may vary.

No product or component can be absolutely secure.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. For more complete information about performance and benchmark results, visit http://www.intel.com/benchmarks .

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit http://www.intel.com/benchmarks .

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.



.

# Please take a moment to rate this session.

Your feedback is important to us.