OCP FUTURE TECHNOLOGIES SYMPOSIUM

OCP Global Summit

November 8, 2021 | San Jose, CA

Evolving Software Defined Memory for CXL Usages

Anjaneya “Reddy” Chagam
Cloud Architect, Intel Corporation

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.

.

Agenda

- CXL 1.1 Usage Models
- CXL Memory Buffer Provisioning
- SDM - Application Managed Memory
- SDM - Kernel Managed Memory
- SDM - CXL Benchmarking
- CXL Demo
- Summary

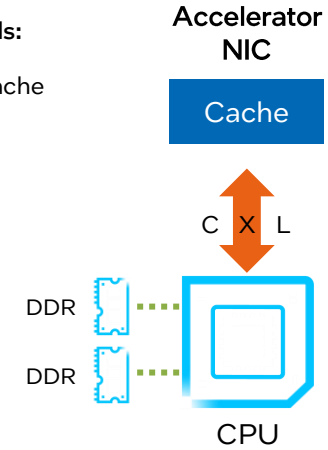
CXL 1.1 Usage Models

(Type 1 Device)

Caching Devices/Accelerators

Usages:
- PGAS NIC
- NIC atomics

Protocols:
- CXL.io
- CXL.cache

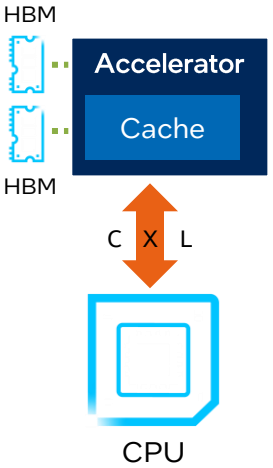


(Type 2 Device)

Accelerators with Memory

Usages:
- GPU
- FPGA
- Dense Computation

Protocols:
- CXL.io
- CXL.cache
- CXL.mem

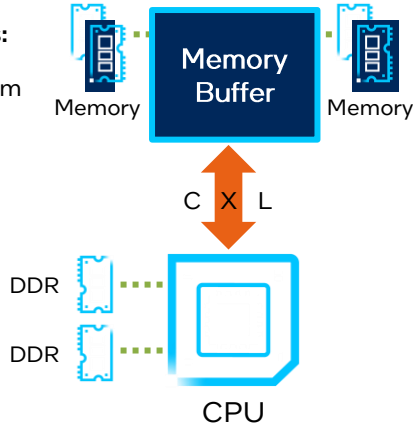


(Type 3 Device)

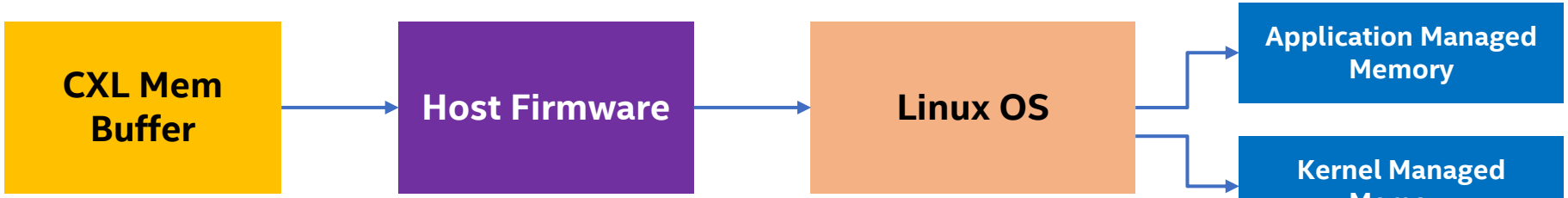
Memory Buffers

Usages:
- Memory BW expansion
- Memory capacity expansion
- 2LM

Protocols:
- CXL.io
- CXL.mem



CXL Memory Provisioning (Linux)



CXL Type2 or Type3
Host-managed Device Memory (HDM)

- **Setup Knobs**
  - 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)

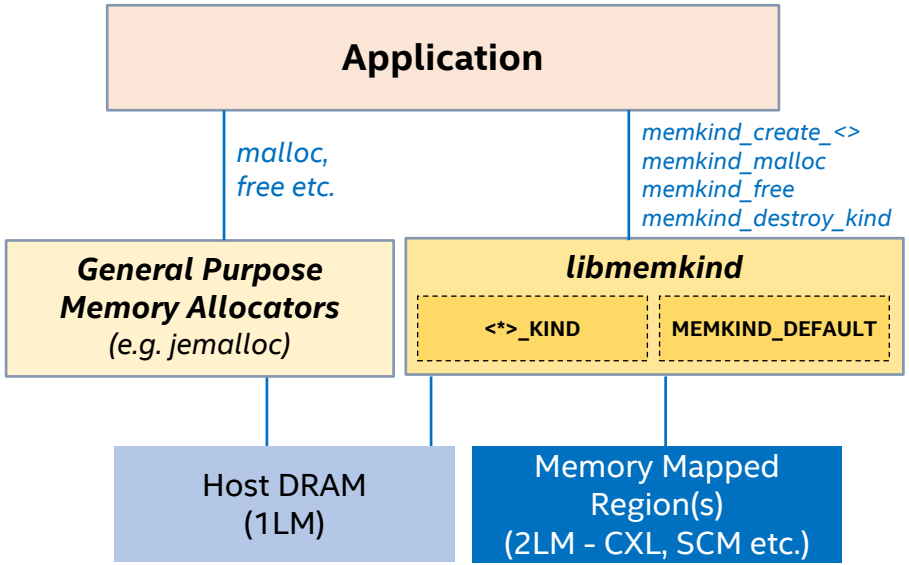
- **CXL Memory Configs**
  - Conventional (Memory only NUMA domain)
  - HMEM/DAX device (/dev/dax<x>.<y>)
- **DAX Memory Config Knobs**
  - efi=nosoftreserve kernel command line
  - daxctl to promote/demote to system ram
- **Kernel Tiering for Conventional Memory**
- **CXL Mailbox Driver**
- **CXL Command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

Application Managed CXL Memory – Basic

```
int fd = open(device, O_RDWR, S_IRWXU); //e.g., device=/dev/dax0.0
if (fd < 0) {
    printf("%s open failed with error %s\n", device, strerror(errno));
    return 1;
}
char *addr = (char *) mmap(NULL, size, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);
if (addr == MAP_FAILED) {
    close(fd);
    return 1;
}

/* write to CXL Memory */
for (int i=0; i < size; ++i)
      addr[i] = 'C';

/* read from CXL Memory */
for (int i=0; i < size; ++i)
      printf("%c", addr[i]);

munmap(addr, size);
close(fd);
```

libndctl, sysfs to enumerate HMEM/DAX devices (size etc.)

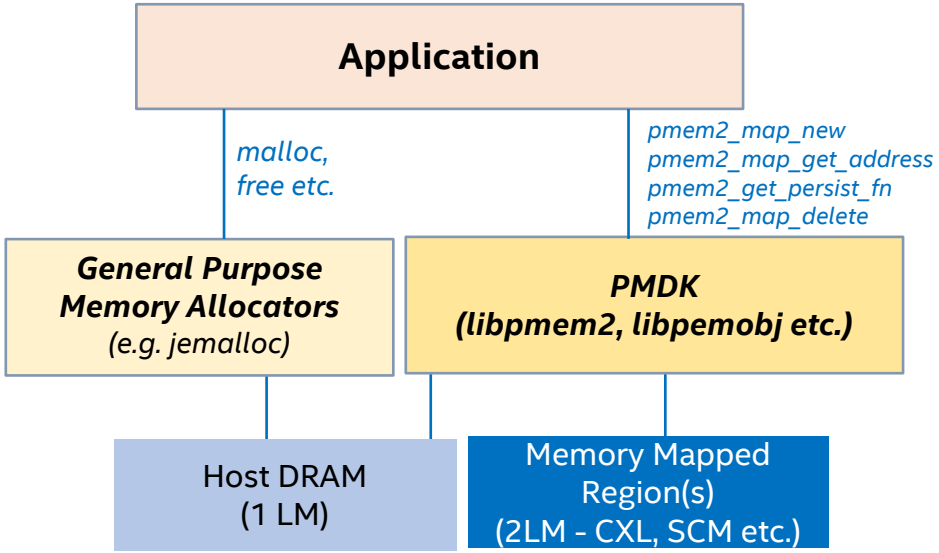
Application Managed CXL Memory – Heap Manager




```
// create memkind partition with specific size
err = memkind_create_pmem(path, size, &mem_kind);
if (err) {
    fprintf(stderr, “create partition error\n");
    return 1;
}

// allocate
str1 = (char *) memkind_malloc(mem_kind, 512);
if (str1 == NULL) {
    fprintf(stderr, “alloc error\n");
    return 1;
}

sprintf(str1, "Hello CXL.\n");

memkind_free(mem_kind, str1);

memkind_destroy_kind(mem_kind);
```

- **Memkind** library is a **user extensible** heap manager built on top of **jemalloc**
- Multiple pools to allocate from (DRAM, HBM, PMEM etc.)
- Need simple modifications to the applications

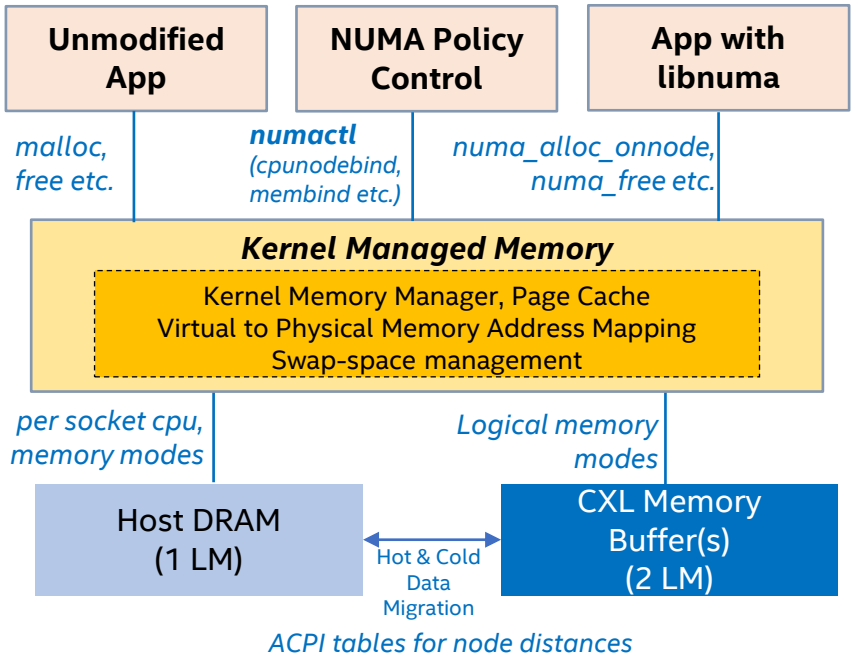
Application Managed CXL Memory – Persistence




```
if ((fd = open(argv[1], O_RDWR)) < 0) { // /dev/dax0.0
    perror("open");
    exit(1);
}
..
if (pmem2_source_from_fd(&src, fd)) {
    pmem2_perror("pmem2_source_from_fd");
    exit(1);
}
..
if (pmem2_map_new(&map, cfg, src)) {
    pmem2_perror("pmem2_map_new");
    exit(1);
}

char *addr = pmem2_map_get_address(map);
..

strcpy(addr, "hello, persistent memory");

persist = pmem2_get_persist_fn(map);
persist(addr, size);

pmem2_map_delete(&map);
..
close(fd);
```

- **The Persistent Memory Development Kit** (PMDK) is a collection of libraries and tools
- **Built on top of DAX** (Direct Access) file system
- **Allows apps** to access **persistent memory as memory-mapped files**

Kernel Managed CXL Memory



Benefits

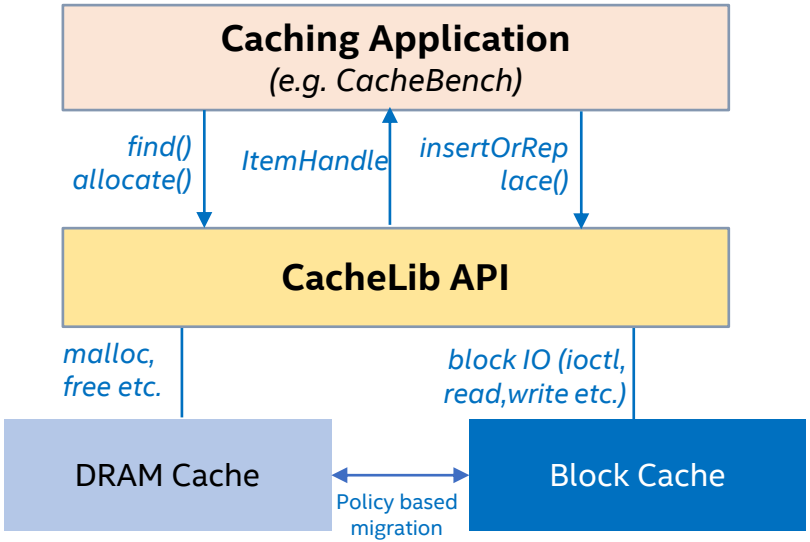
- OS can **map 2nd level memory** into application's **virtual address space**
- **Cooler pages copied to 2nd level mem** instead of 'swap out' to disk.
- Applications **can execute from pages in 2nd level** mem (albeit more slowly) avoiding Page Fault traps into the kernel.
- Kernel memory manager can implement **varying policies for migrating hot & cold pages** between tiers

Downside

- Page copying uses CPU and can impact performance
- Page copies require TLB flushes which impacts performance

Linux Kernel tiering in early development stages

SDM CXL Benchmarking – CacheBench



Github: https://github.com/facebook/CacheLib

CacheLib

- **Pluggable in-process 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

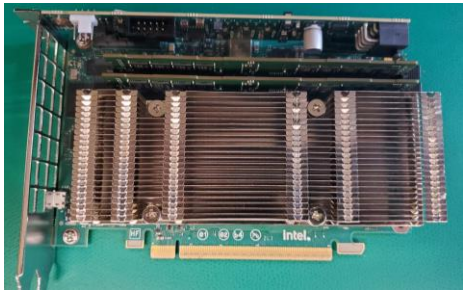
CacheBench

- **Benchmarking tool** for evaluating caching performance
- **numactl or Kernel tiering** as DRAM Cache for **CXL memory buffer benchmarking**

Need memory tiering support

Intel Xeon Sapphire Rapids/CXL Enabling - Demo

Intel Pre-production CXL FPGA Memory Buffer



Intel Sapphire Rapids Pre-production Platform

CXL Memory in Linux OS

Intel Xeon Sapphire Rapids Pre-Production OCP Platforms and FPGA memory buffer interop demonstrated

Summary

- **Software-Defined Memory** (SDM) initiative is focused to assist adoption of **Hierarchical/Hybrid memory solutions**
- **Newer memory technologies** (e.g., SCM, HBM) and **industry standard interconnects** (e.g., CXL) are key components of SDM
- **Kernel tiering**, application libraries such as **CacheLib** provide basic abstraction to underlying memory and storage resources
- **Industry wide effort needed to drive SDM** from concept to reality


OCP
TOTAL SOLUTIONS

OCP FUTURE TECHNOLOGIES SYMPOSIUM

2021 OCP Global Summit | November 8, 2021, San Jose, CA