An Open Solution for CXL Fabric Management & Orchestration

- Barrett Edwards – Jackrabbit Labs

Who is Jackrabbit Labs?

The open-source software services company for shared memory management

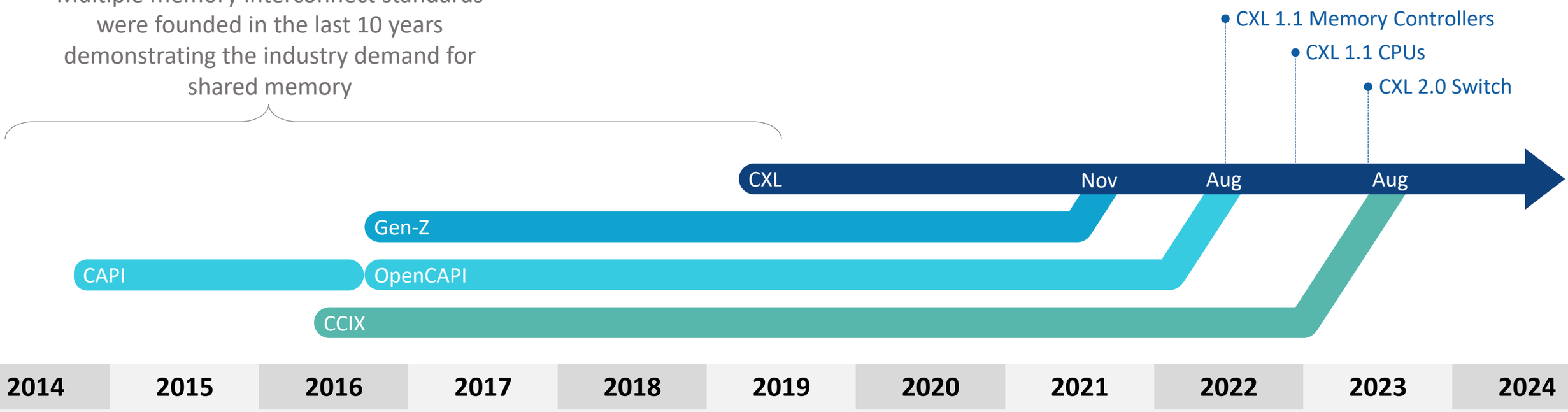
Competing technologies have merged into a single standard: CXL

CXL is the specification for next generation computer architecture both for hardware and software

Multiple memory interconnect standards were founded in the last 10 years demonstrating the industry demand for shared memory

We are at the cusp of a new paradigm of computing based on shared memory. With the hardware now available, software must be adapted to utilize it.

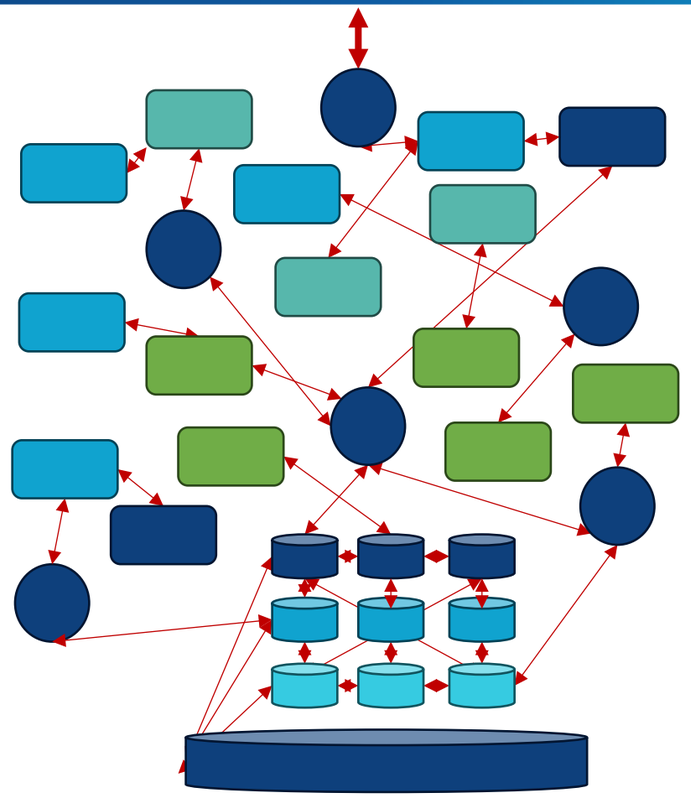
Now aligned, industry participants have designed and launched multiple CXL products.



Shared Memory Improves the Efficiency of Software Architectures

Memory Fabrics Improve Optimization to Radically Reduce Cost & Complexity

Architecture Today



Eliminate Duplicate Data

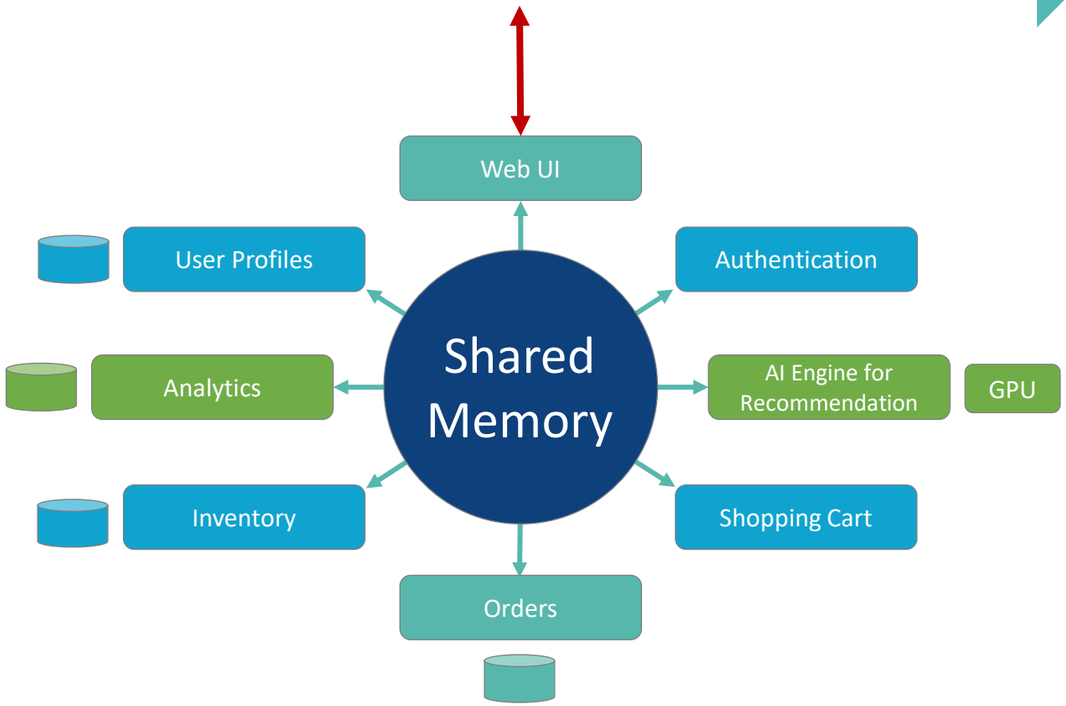
Share Memory Across Containers

Remove Extra Network Hops

Slow Ethernet

Fast Memory

Architecture Tomorrow



Kubernetes has reached a level of diminishing returns, mainly due to network and forced data duplication

Jackrabbit Labs Platform Reduces:
Storage - Memory - Latency

Example: Traditional eCommerce with duplicate services over slow Ethernet vs same services using fabrics

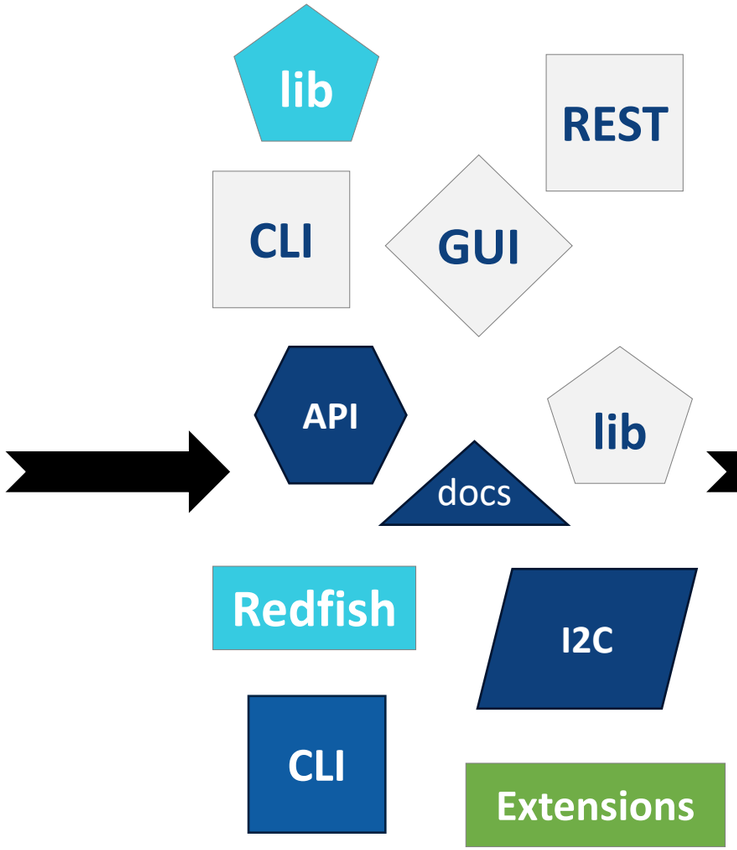
Why is Open-Source Software Needed for CXL?

Fragmentation slows adoption

Emerging Hardware Ecosystem

- Specification
- Compute
- Switches
- Memory Controllers
- Memory Modules
- Appliances

Fragmented Software Ecosystem



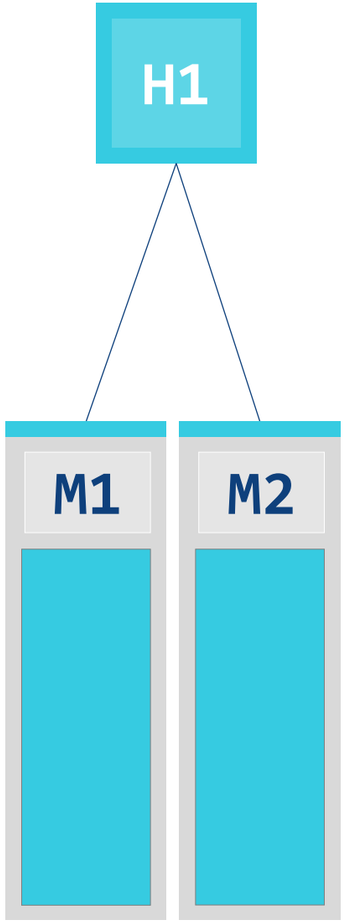
Standardized Software Stack

- App Plugins
- Management API
- Management Libraries
- Host Agent(s)
- In-band Management Protocol
- Management Endpoints
- SwitchOS
- Hardware Abstraction Layer

CXL Memory Topologies

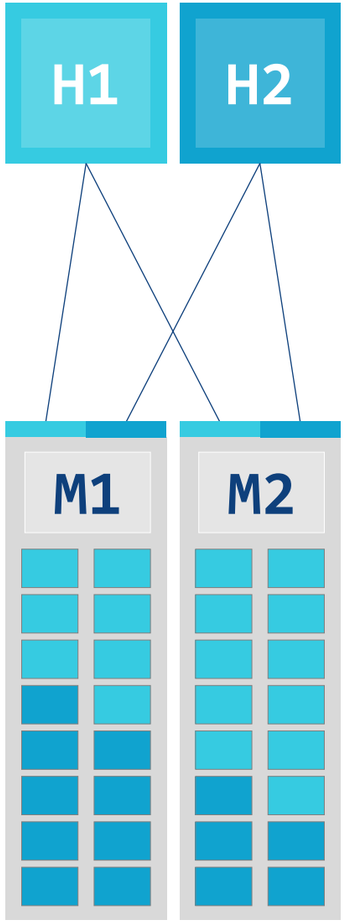
Hosts, Switches, and Devices can be connected in a Direct or Switched Topology

Direct Attach
Memory Expansion



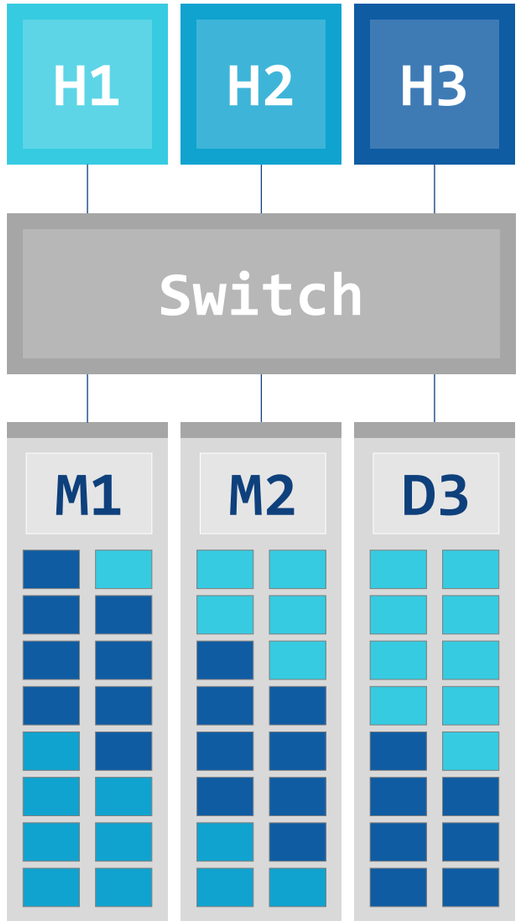
SH-SLD – DRAM Drives
Single-Headed Single Logical Devices

Direct Attach
Device Pooling / Sharing



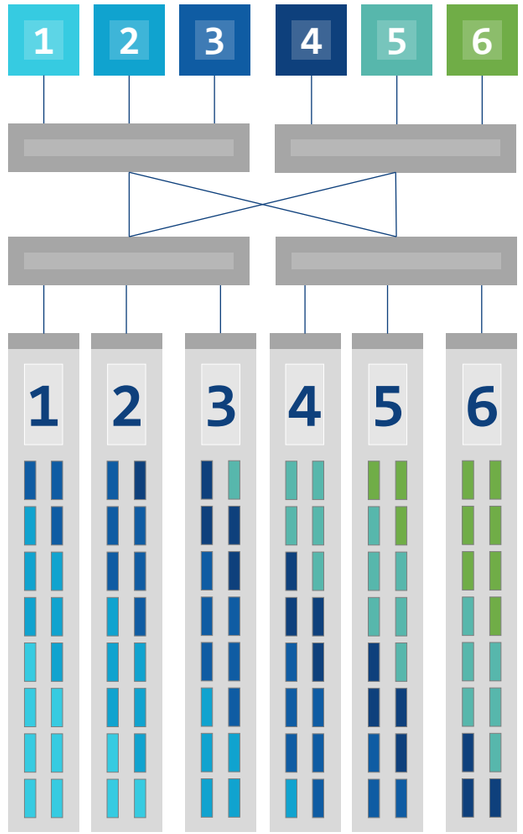
MH-MLD – DRAM Drives
Multi-Headed Multi-Logical Devices

Single Layer Switch
Device Pooling / Sharing



SH-MLD – DRAM Drives
Single-Headed Multi-Logical Devices

Multi-Layer Switch
Device Pooling / Sharing

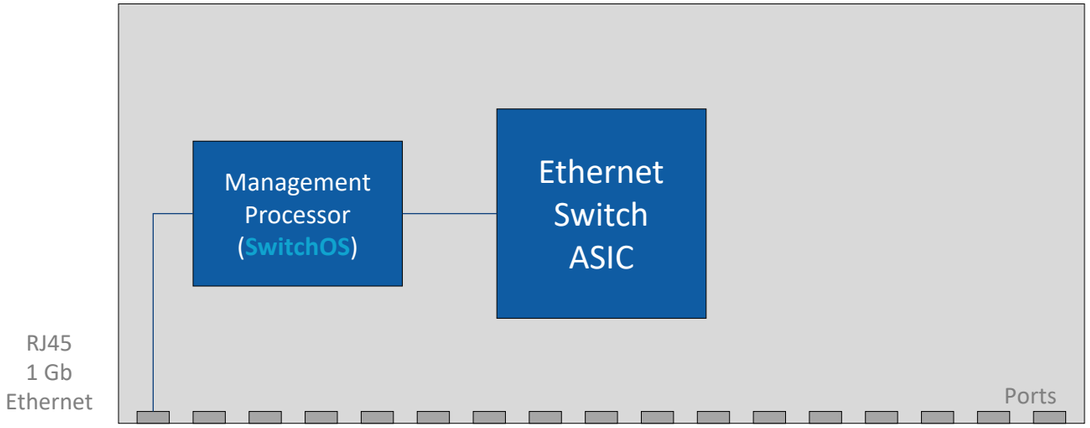


SH-MLD – DRAM Drives
Single-Headed Multi-Logical Devices

SwitchOS

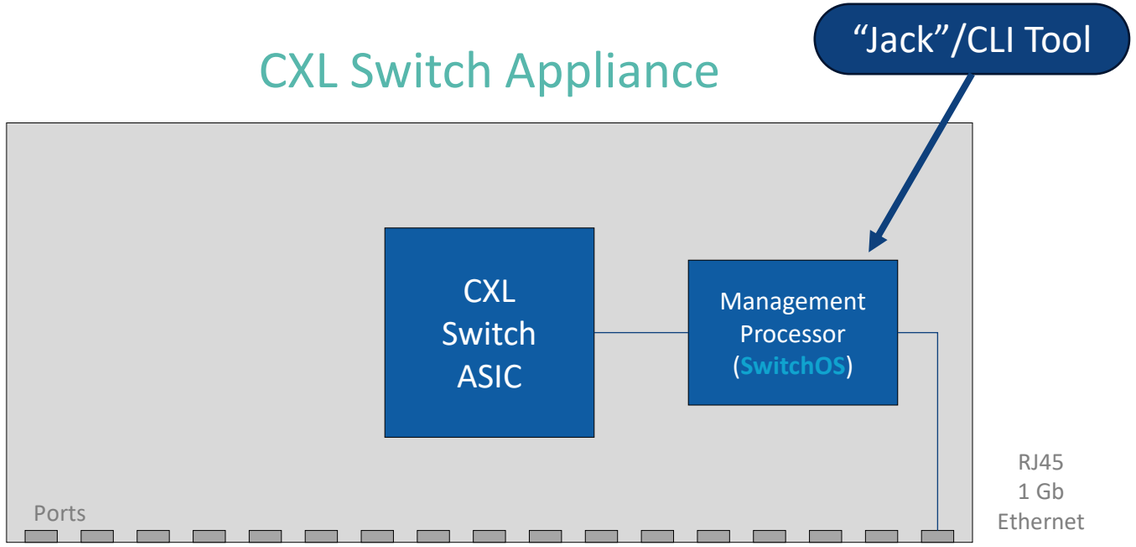
The Management Abstraction Layer to the Silicon

Ethernet Switch Appliance



- Managed Ethernet switches run a SwitchOS
- e.g. SONiC, Cumulus, FBOSS, EOS, NX-OS
- Managed through in-band / out-of-band Ethernet links
- Hardware Abstraction Layer (HAL)
- Can be run on a low-end BMC or larger x86 processor
- SONiC = Debian + Ethernet Management Containers
- Typically has a CLI shell + Web API / GUI

CXL Switch Appliance

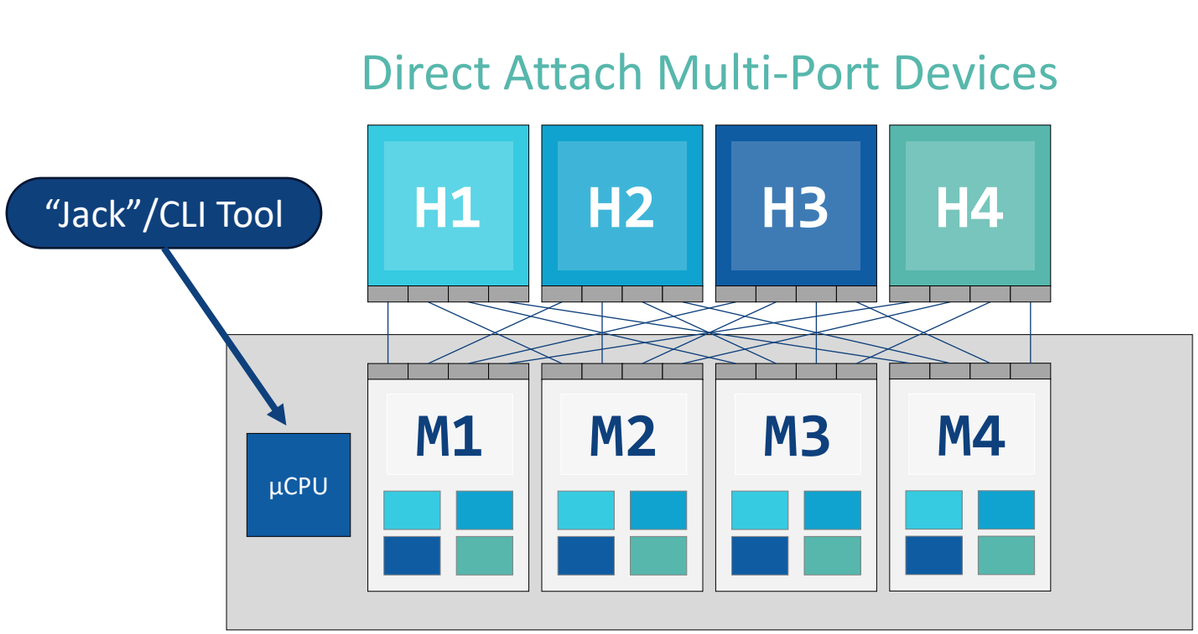


- CXL Switch appliances are equivalent to Ethernet switches
- Will run a SwitchOS to manage CXL switch silicon
- The "Fabric Manager" lives in this SwitchOS
  (Or at least a software agent of a larger orchestration system)
- Has a Hardware Abstraction Layer (HAL) for CXL switch silicon
- External interface can be REST, GUI over Ethernet or an in-band protocol over CXL links

SwitchOS

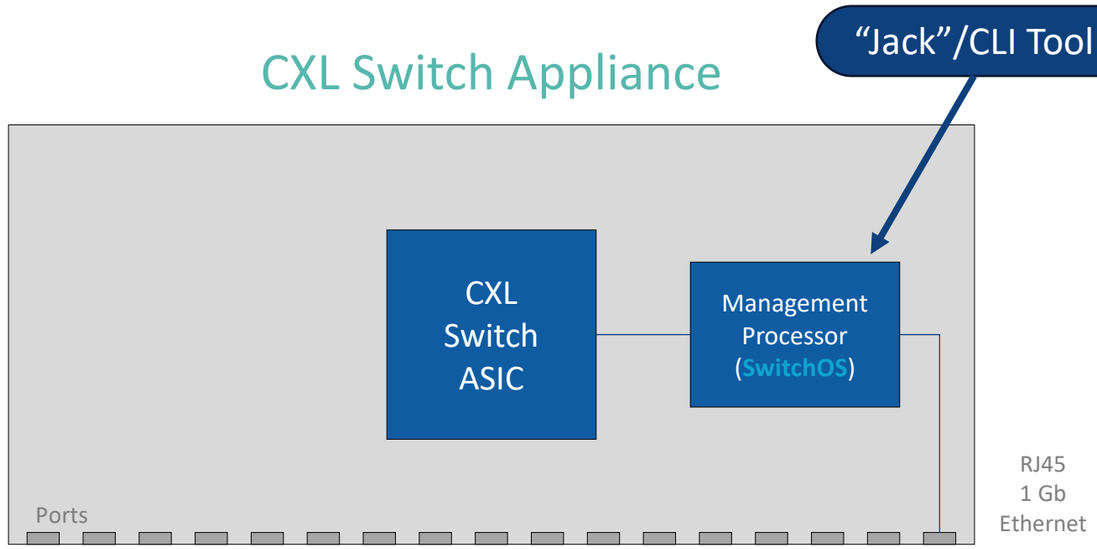
The Management Abstraction Layer to the Silicon

Direct Attach Multi-Port Devices



- Directly connected Multi-Headed (Multi-Port) devices
- No switch architecture
- Memory devices housed in separate / bladed enclosure
- Lower latency – more cables / complex enclosure
- Still requires separate management entity

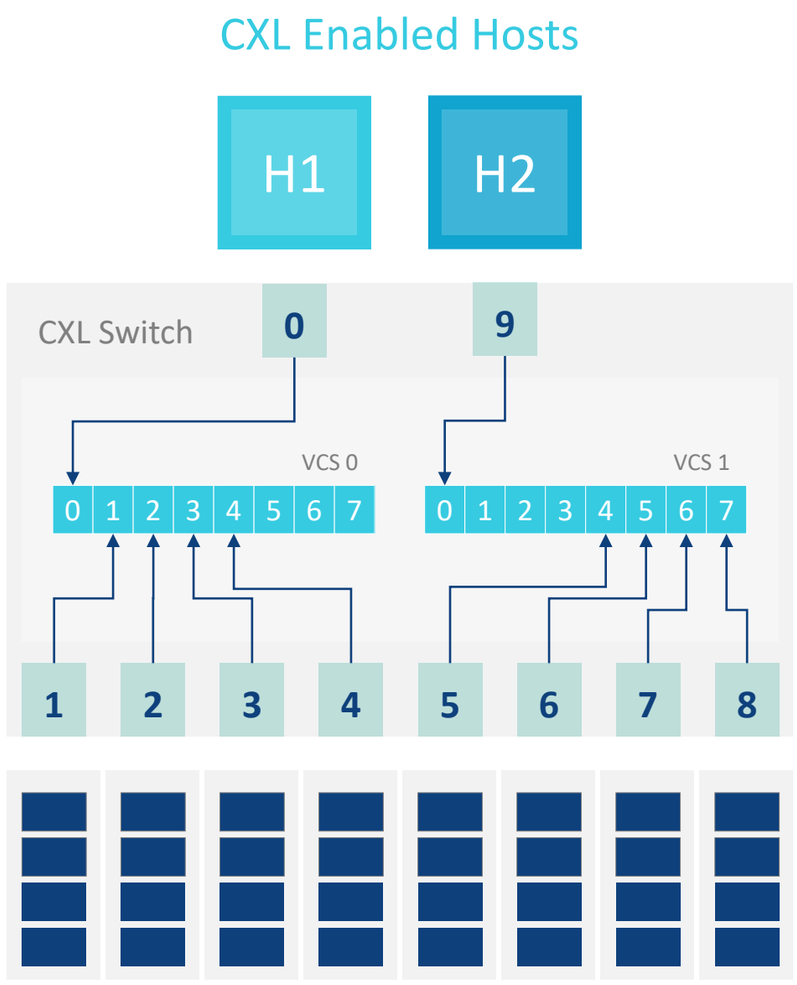
CXL Switch Appliance



- CXL Switch appliances are equivalent to Ethernet switches
- Will run a SwitchOS to manage CXL switch silicon
- The "Fabric Manager" lives in this SwitchOS
  (Or at least a software agent of a larger orchestration system)
- Has a Hardware Abstraction Layer (HAL) for CXL switch silicon
- External interface can be REST, GUI over Ethernet or an in-band protocol over CXL links

Jack – CXL Fabric Management CLI Tool

Implements the CXL Fabric Management API

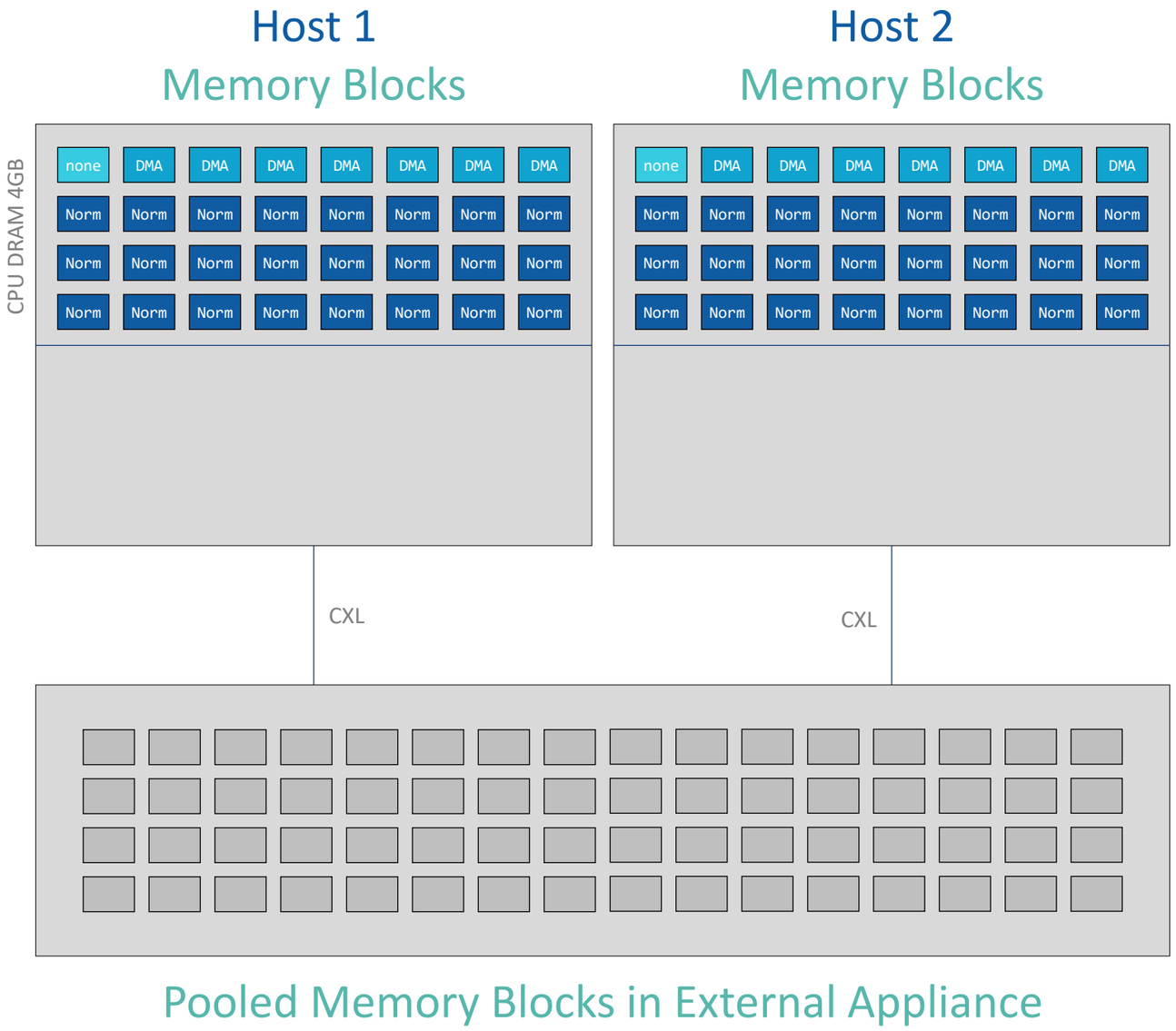



```
# jack show port

#    @  Port State  Type    LD  Ver  CXL Ver   MLW  NLW  MLS  CLS  Speeds    LTSSM     LN  Flags
---  -  ----------  ------  --  ---  --------  ---  ---  ---  ---  --------  --------  --  ------------
0    +  Upstream    T1      -   2.0  AB        16   16   5.0  5.0       45    L0        0     P
1    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0       45    L0        0     P
2    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0       45    L0        0     P
3    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0       45    L0        0     P
4    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0       45    L0        0     P
5    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0       45    L0        0     P
6    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0       45    L0        0     P
7    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0       45    L0        0     P
8    +  Downstream  T3-MLD  16  2.0  AB        16   16   5.0  5.0       45    L0        0     P
9    +  Upstream    T1      -   2.0  AB        16   16   5.0  5.0       45    L0        0     P

# jack show vcs 0

Show VCS:
VCS ID  : 0
State   : Enabled
USP ID  : 0
vPPBs   : 8

vPPB  PPID LDID Status
----  ---- ---- -----------
   0:    0    - Bound Physical Port
   1:    1    0 Bound LD
   2:    2    0 Bound LD
   3:    3    0 Bound LD
   4:    4    0 Bound LD
   5:    -    - Unbound
   6:    -    - Unbound
   7:    -    - Unbound
```

https://github.com/JackrabbitLabs/jack

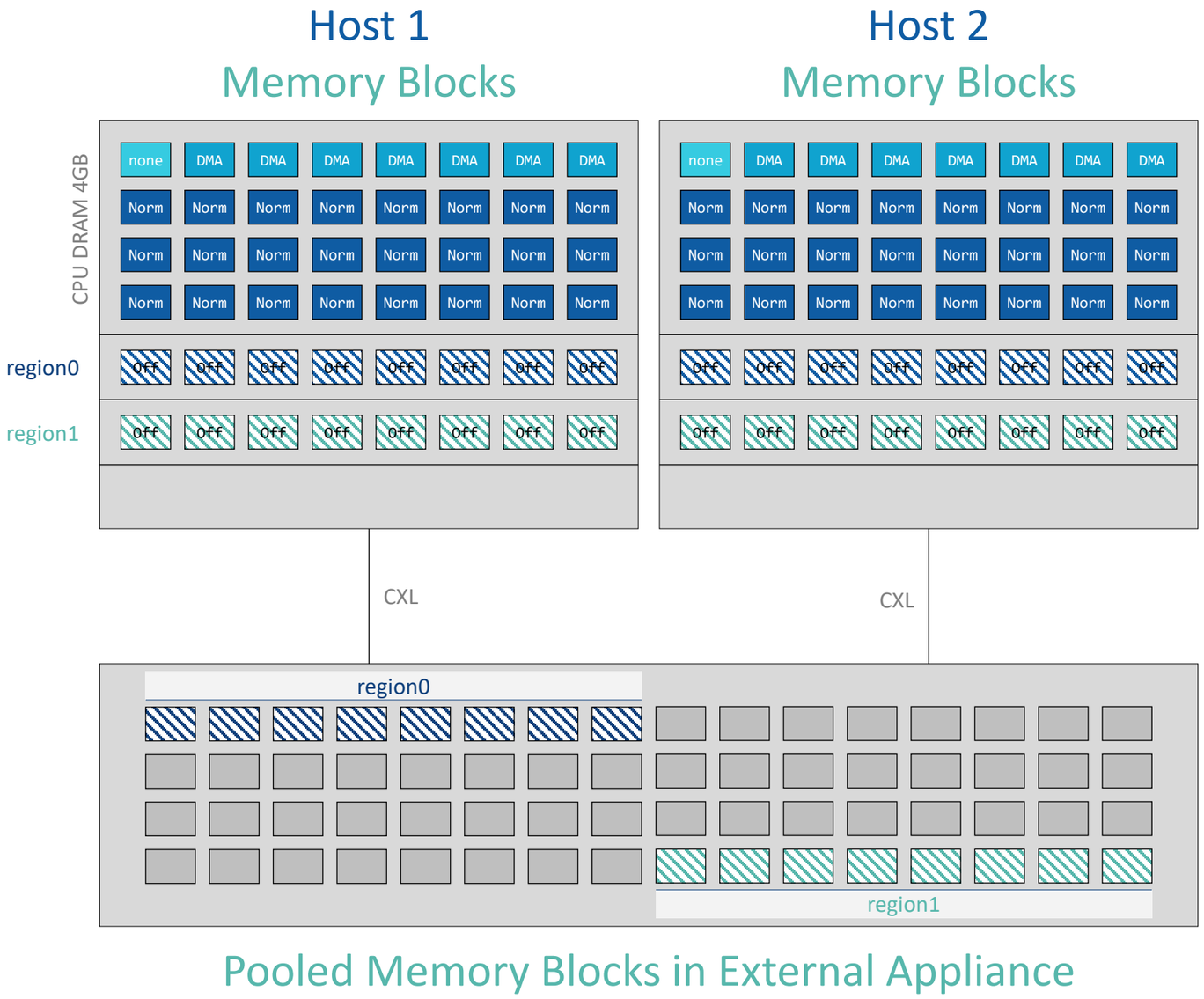
Live Demo

libmem – Library for Managing Memory Blocks



- Host memory is divided into blocks (typically 128 MiB)
- Each memory block is assigned a zone

libmem – Library for Managing Memory Blocks

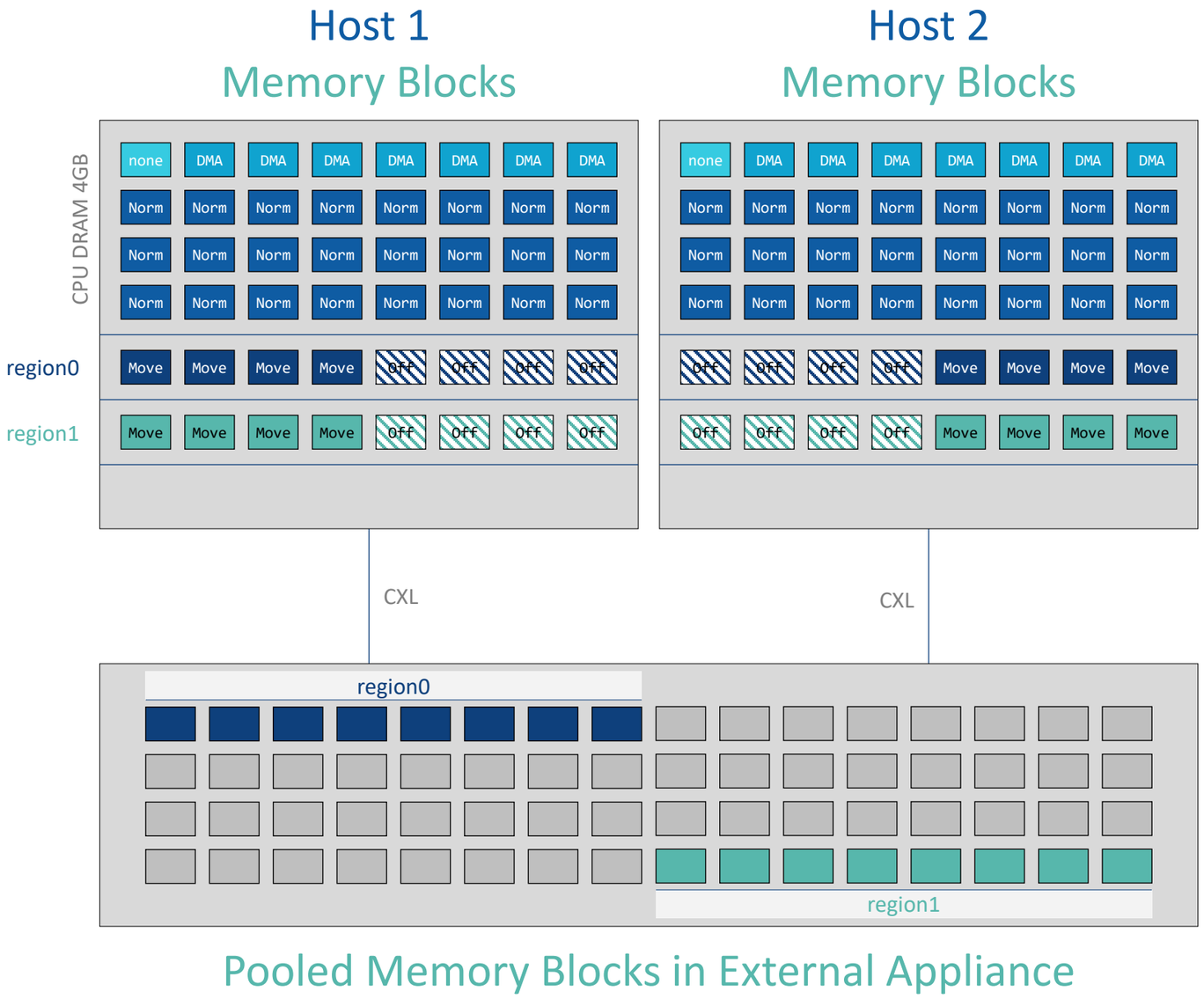


Pooled Memory Blocks in External Appliance

- Host memory is divided into blocks (typically 128 MiB)
- Each memory block is assigned a zone

- External CXL attached memory added as regions
- Regions can be dynamically added
- Regions must be removed in reverse order (LIFO)

libmem – Library for Managing Memory Blocks

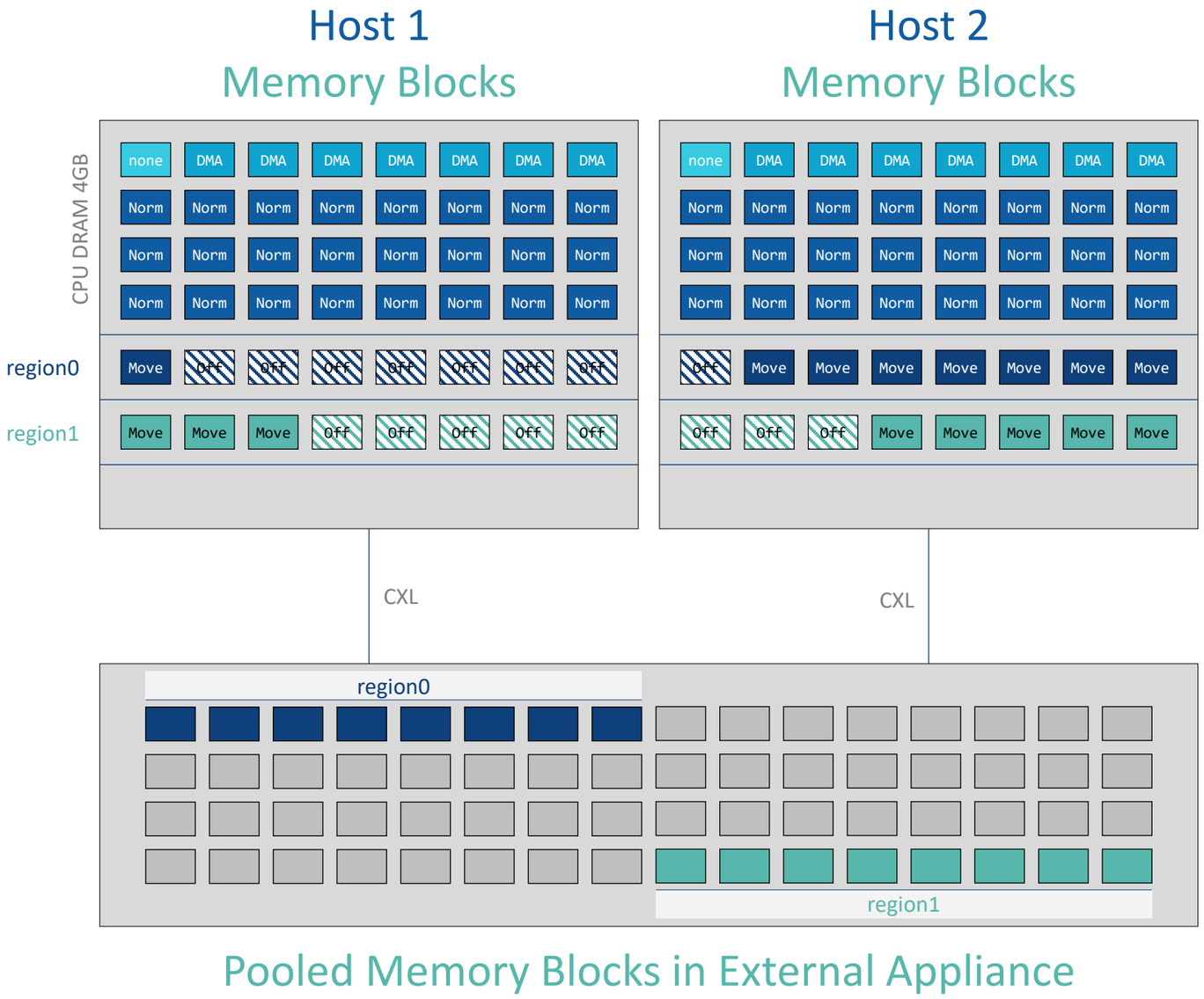


- Host memory is divided into blocks (typically 128 MiB)
- Each memory block is assigned a zone

- External CXL attached memory added as regions
- Regions can be dynamically added
- Regions must be removed in reverse order (LIFO)

- Only online a memory block in one of the hosts at a time

libmem – Library for Managing Memory Blocks



- Host memory is divided into blocks (typically 128 MiB)
- Each memory block is assigned a zone

- External CXL attached memory added as regions
- Regions can be dynamically added
- Regions must be removed in reverse order (LIFO)

- Only online a memory block in one of the hosts at a time

- Memory blocks can be dynamically offlined in one host, and then onlined in another host

libmem is a library to manage the state (online/offline) of memory blocks of a CXL memory region

Review

- Challenges
  - Potential fragmentation of the shared memory software ecosystem will delay adoption
  - Dynamically removing memory from the kernel requires shared access across multiple hosts (until DCD)
  - ZONE_MOVABLE not a 100% reliable way to hot-remove memory from the kernel
- Call to Action
  - Experiment with QEMU today! QEMU supports more CXL features (i.e. CXL 3.0+) than CPU HW today
  - Software application development doesn't have to wait until hardware (i.e. switches) are available
  - Evaluate where open-source tools / libraries / APIs can be used in your projects

Intel	libcxl	https://github.com/pmem/ndctl
Jackrabbit Labs	libmem	https://github.com/JackrabbitLabs/libmem
Jackrabbit Labs	Jack - CXL FM API CLI Tool	https://github.com/JackrabbitLabs/jack
Jackrabbit Labs	CXL Switch Emulator	https://github.com/JackrabbitLabs/cse
Samsung	Scalable Memory Development Kit (SMDK)	https://github.com/OpenMPDK/SMDK
Micron	CXL Memory Resource Kit (CMRK)	https://github.com/cxl-micron-reskit/cxl-reskit
SK Hynix	Heterogeneous Memory Software Development Kit (HMSDK)	https://github.com/skhynix/hmsdk
Micron	CXL Library CLI	https://github.com/cxl-micron-reskit/mxcli
Micron	FAMFS	https://github.com/cxl-micron-reskit/famfs
Intel	Unified Memory Framework	https://github.com/oneapi-src/unified-memory-framework
QEMU	QEMU	https://github.com/qemu/qemu
Samsung	libcxlmi	https://github.com/computexpresslink/libcxlmi

JACKRABBIT LABS

Driving CXL Adoption with Open Source