Welcome To

A Practical Guide to Compute Express Link Memory Devices

A Hands-on Lab

What You Will Learn

- An Introduction to CXL Memory Devices
- CXL Memory Benefits and Use Cases
- The CXL Architecture
- Hands-On Lab: Emulating CXL Devices in QEMU
- Continue Your Learning Path

Prerequisites

- A basic understanding of Compute Express Link™ (CXL) specifications v1.x, 2.x, and 3.x
- A Laptop
- SSH Client
  - Windows: Putty (https://putty.org)
  - Windows: MobaXTerm (https://mobaxterm.mobatek.net/)
  - Linux/Mac: Use the native `ssh` client
- Internet Access
  - Conference Wifi SSID: <TODO>
  - Conference Wifi Password: <TODO>

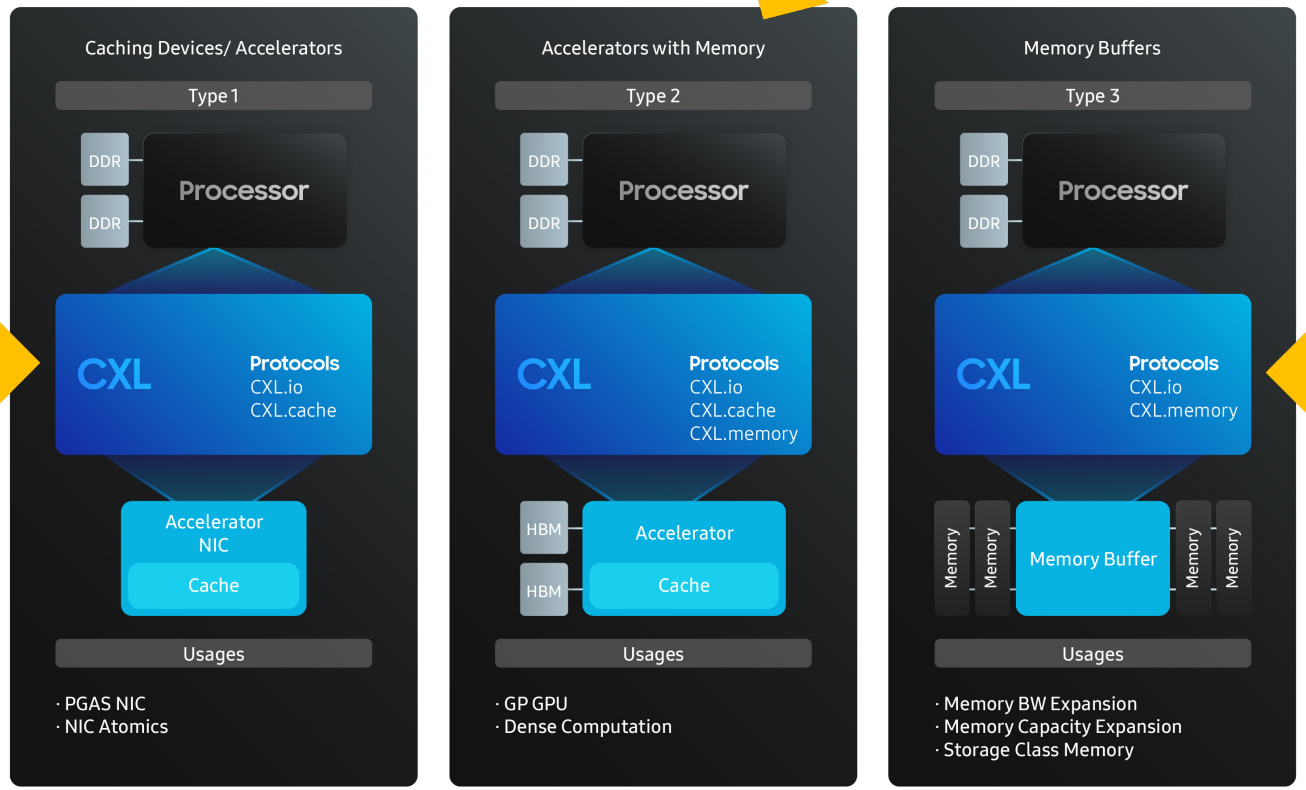
An Introduction to CXL Memory Devices

CXL.MEM Type 3 Endpoints

CXL Device Types

A hybrid of Type 1 and Type 3

The CXL device can cache host memory



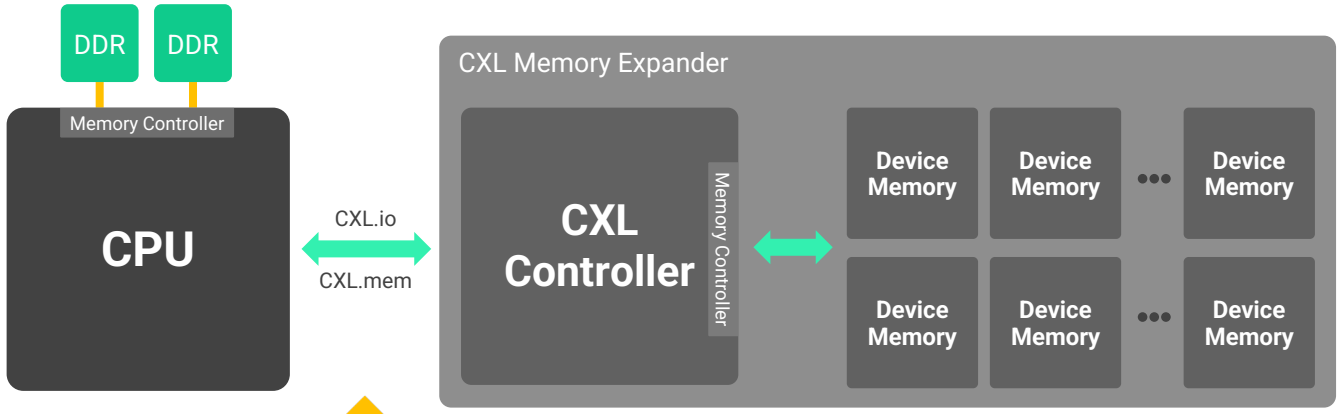
The CPU can access DRAM and CXL Memory

Figure: Representative CXL Usages (From CXL™ consortium)

Source: https://semiconductor.samsung.com/news-events/tech-blog/expanding-the-limits-of-memory-bandwidth-and-density-samsungs-cxl-dram-memory-expander/

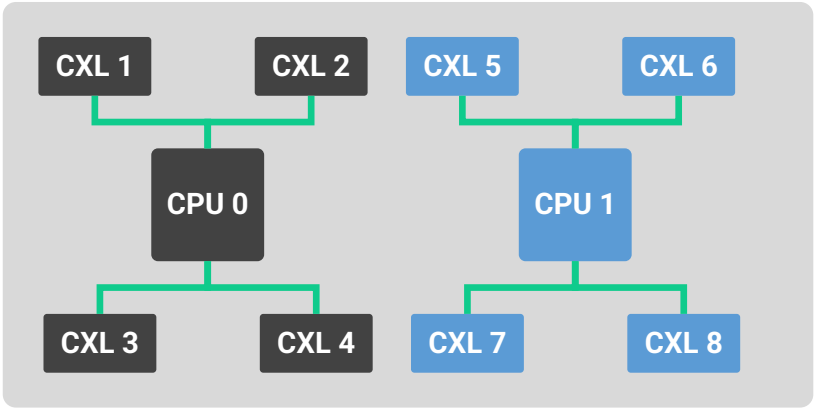
Type 3 Device Modes

Single Logical Devices (SLDs)



A **Single Port** between **CPU** and **CXL Controller**

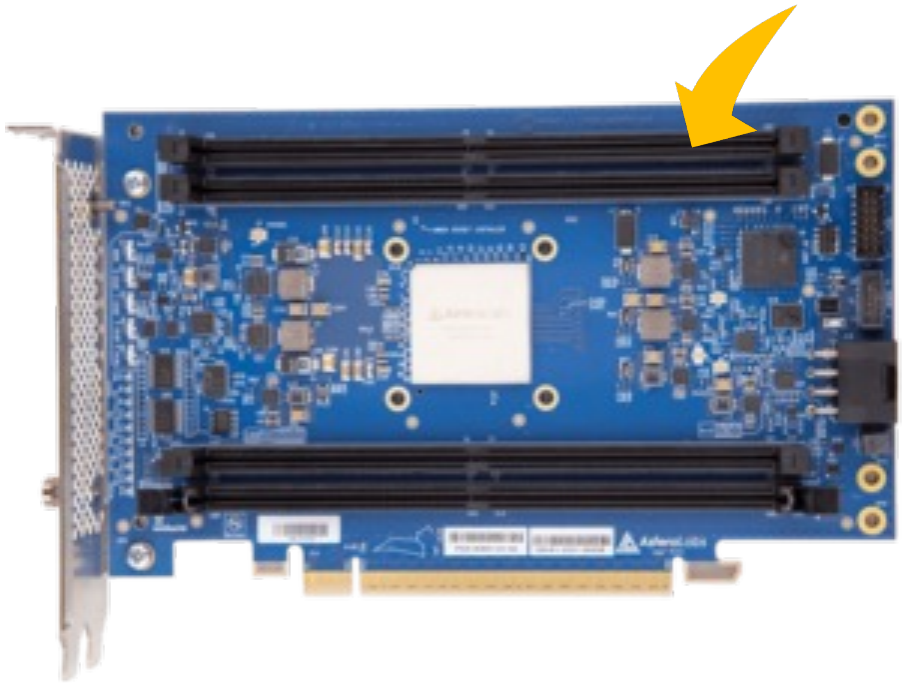
All device capacity is mapped to the host CPU



A 2-Socket Server with **8 CXL Devices**

Type 3 Device Form Factors

DDR Slots



Source: https://www.asteralabs.com/product-details/leo-system-validation-board/

Add-in Card

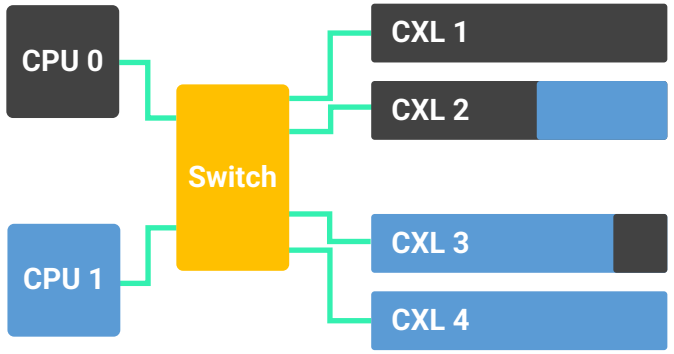
Source: https://news.samsung.com/global/samsung-electronics-introduces-industrys-first-512gb-cxl-memory-module

E3.S

Type 3 Device Modes
Multi Logical Devices (MLDs)

- A Type 3 Multi-Logical Device (MLD) can partition its capacity into isolated Logical Devices

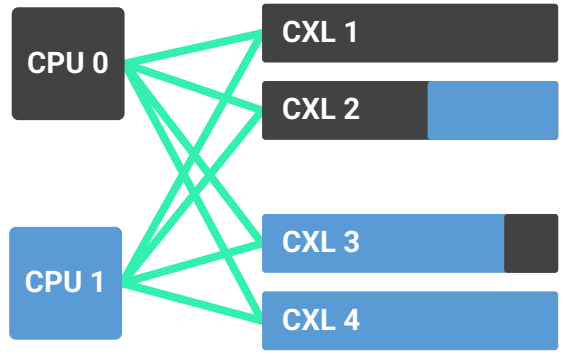
Direct Attached (CXL 1.x)



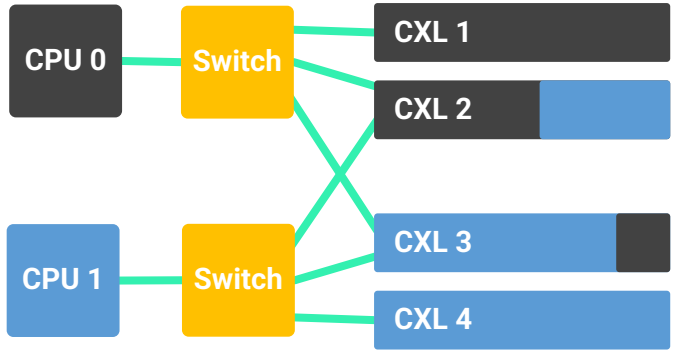
Switch Attached (CXL 2.x/3.x)

Type 3 Device Modes
Multi-Headed Logical Devices (MH-xLDs)

- A Type 3 device with multiple CXL ports is considered a Multi-Headed Device (MHD)
- Two types of Multi-Headed Devices:
  - MH-SLD, present SLDs on all heads
  - MH-MLD, may present MLDs on any of their heads



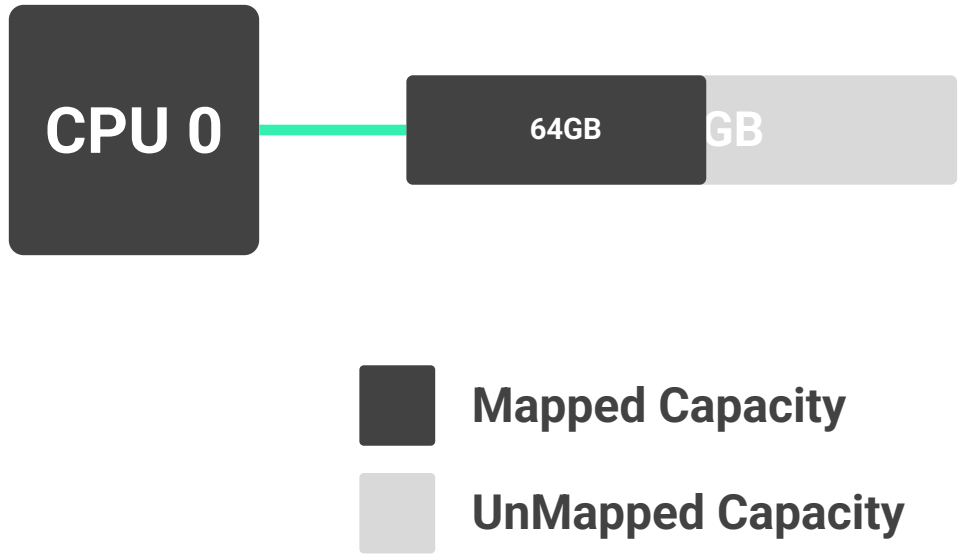
Direct Attached (CXL 1.x)



Switch Attached (CXL 2.x/3.x)

Type 3 Device Modes
Dynamic Capacity Devices (DCDs)

- **Dynamic Capacity** is a feature of a CXL memory device that allows the memory capacity to change dynamically without the need for resetting the device.
- A **DCD** is a CXL memory device that implements Dynamic Capacity.
- DCDs may be Multi- or Single-Headed



CXL Benefits and Use Cases

Why and When to use CXL Memory Expansion Devices

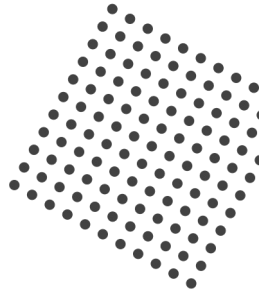
CXL Benefits & Use Cases

Benefits

- High bandwidth, low latency, and coherent interconnect
- Builds on the PCIe physical and electrical interface
- Expand memory capacity and bandwidth beyond DRAM
- Disaggregated memory can be elastically provisioned to fit application demand and growth – like storage and network
- Solve the Stranded Memory and Frigid Memory problems

Use Cases

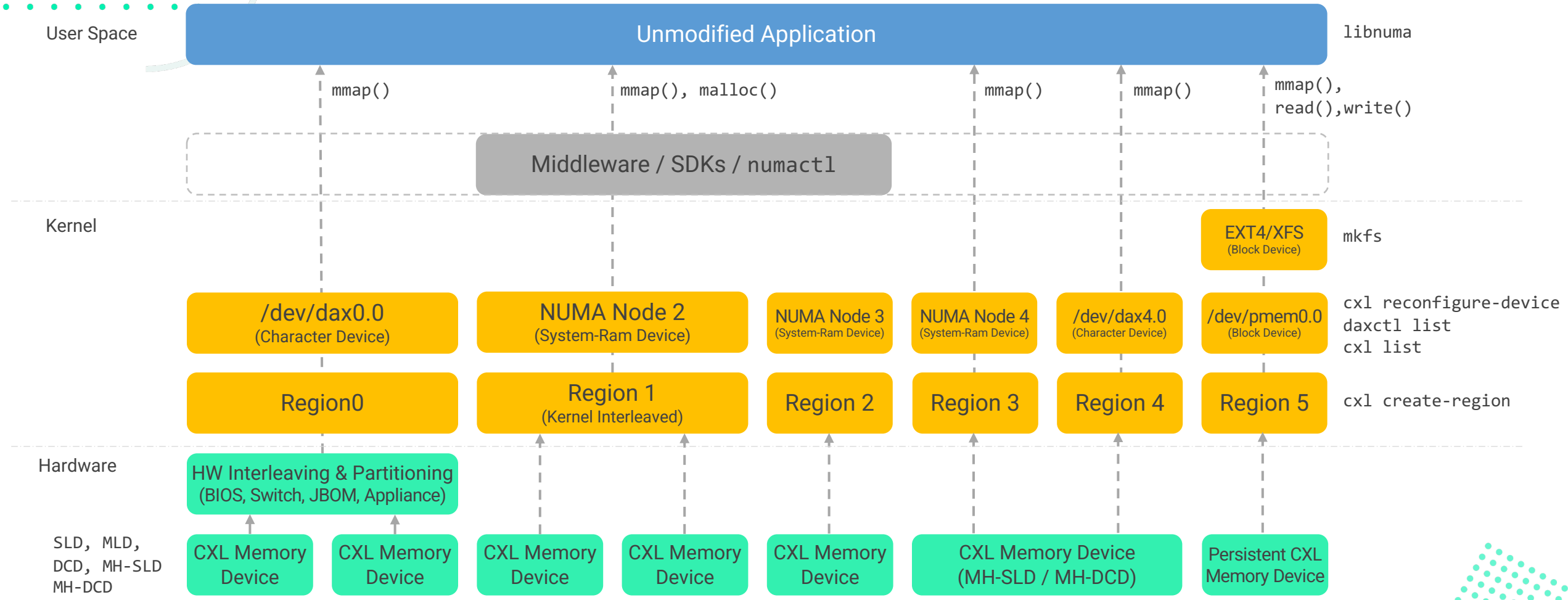
- AI/ML
- HPC
- Big Memory Databases
  - IMDB
  - RDBMS
  - Graph
  - Vector
  - …
- Streaming Analytics
- Gaming
- 3D Animation Studios
- Video Processing
- Many more …



CXL Architecture

Understand the Hardware and Software Stack

CXL Architecture



Hands-On Lab

Emulate CXL Memory Devices using QEMU

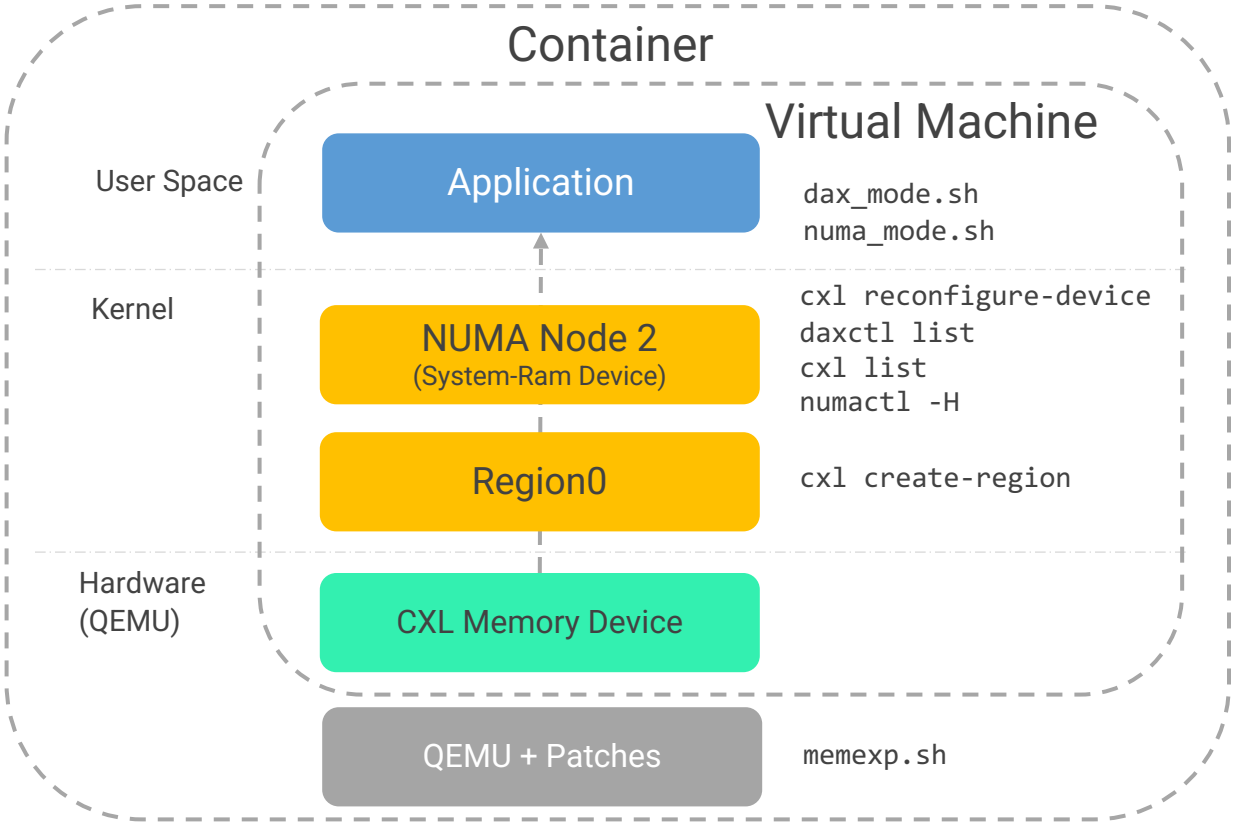
Lab Objectives

1. Provide early exposure to CXL Memory Devices and Features before hardware is generally available
2. Give you hands-on experience with the Linux utilities (`cxl`, `daxctl`, `numactl`, etc)
3. Provide a learning environment you can take home with you

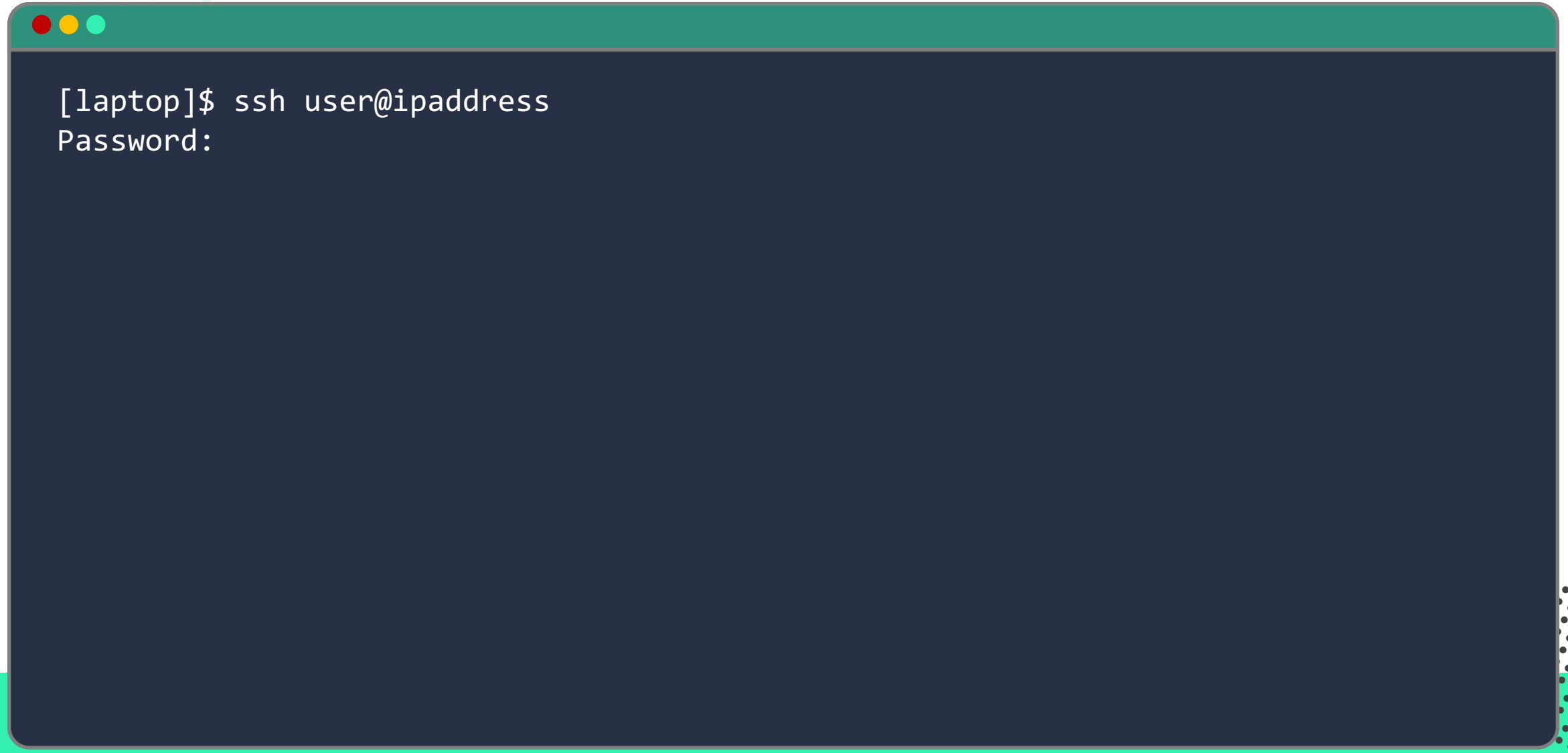
Lab Overview

1. ssh to your assigned Cloud Instance
2. Install `podman`
3. Pull the MemVerge CXL Expansion container image
4. Run the Container
5. Start a Guest VM with a single CXL Memory Expander
6. Login to the Guest VM
7. Create a Region
8. Explore the CXL Device using Linux tools
9. Start an Application using CXL and DRAM memory

Lab Environment



SSH to Your Assigned Lab Host



Install Podman

```
[host]$ sudo apt install -y podman
```

Start the Container & Virtual Machine

```
// Pull the Container image
[host]$ podman pull docker.io/mvpool/qemu_cxl_memexp

// Start the container as a daemon
[host]$ podman run -d --name cxllab qemu_cxl_memexp

// Connect to the container
[host]$ podman exec -it cxllab bash

// Wait a few minutes for the VM to start and initialize

// Connect to the Virtual Machine (Please be patient!)
[container]$ ssh -p 2222 fedora@localhost
Password: password
```

Create a Region

```
// Create a new Region and a devdax in ‘System-Ram’ mode
[vm]$ ./create_region.sh
```

Explore the CXL Tools

```
// List the CXL devices
[vm]$ cxl list
[vm]$ cxl list -vvv

// List the DAX devices
[vm]$ daxctl list

// Install pciutils
[vm]$ sudo dnf install -y pciutils

// Check the PCI device(s)
[vm]$ lspci | grep -i cxl
35:00.0 CXL: Intel Corporation Device 0d93 (rev 01)

// Get more info about the PCI/CXL device
[vm]$ lspci -s 35:00.0 -vvv
```

Explore the CXL Tools

```
// List the NUMA Nodes
[vm]$ numactl -H
available: 2 nodes (0-1)
node 0 cpus: 0 1 2 3
node 0 size: 3901 MB
node 0 free: 3062 MB
node 1 cpus:                <-- A CXL.mem Device has no CPUs
node 1 size: 4096 MB
node 1 free: 4096 MB
node distances:
node   0   1
  0:  10  20
  1:  20  10 <-- CXL Device
```

Explore the CXL Tools

```
// List the Memory Blocks
[vm]$ lsmem
[vm]$ lsmem -o+ZONES,NODE
```

Start an Application

```
// Allocate memory from DRAM and CXL using a 50:50 Interleave policy
$ numactl –interleave=0,1 memhog 1g

// Allocate memory entirely from CXL
$ numactl –-membind 1 memhog 1g
```

Explore the CXL Tools

```
// List the NUMA Node statistics
[vm]$ numastat
                           node0           node1
numa_hit                1154086          262175
numa_miss                     0               0
numa_foreign                  0               0
interleave_hit              980               0
local_node              1154086               0
other_node                    0          262175
```

Explore the CXL Tools

```
// Convert the ‘System-Ram’ device to a ‘devdax’
[vm]$ cd
[vm]$ ./dax_mode.sh

// Convert the ‘devdax’ device to a ‘System-Ram’ node
[vm]$ cd
[vm]$ ./numa_mode.sh
```

Explore the Kernel

```
// Investigate /dev
[vm]$ ls /dev/dax*
[vm]$ ls /dev/cxl/

// sysfs has a lot of useful information. Explore and have fun.
[vm]$ ls /sys/bus/node/devices/node1/
[vm]$ ls /sys/bus/cxl/devices/
[vm]$ ls /sys/bus/acpi/devices/
```

Continue Your Learning Path

- CXL Consortium: https://www.computeexpresslink.org/
- Linux Kernel CXL Mailing List: https://lore.kernel.org/linux-cxl/
- Linux Kernel CXL Driver: https://github.com/torvalds/linux/tree/master/drivers/cxl
- NDCTL Linux Tools: https://github.com/pmem/ndctl
  - Includes `cxl` & `daxctl`
- QEMU: https://www.qemu.org/
  - Join the Community: https://www.qemu.org/contribute/