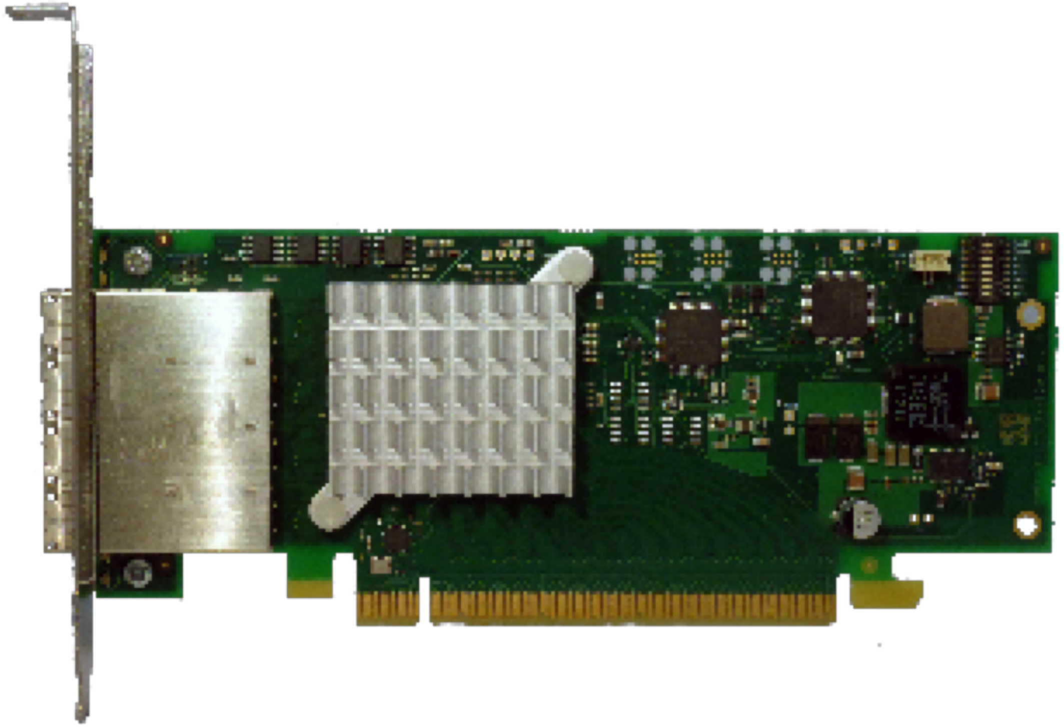
S9709
Dynamic Sharing of GPUs and IO in a PCIe Network

Håkon Kvale Stensland
Senior Research Scientist / Associate Professor
Simula Research Laboratory / University of Oslo

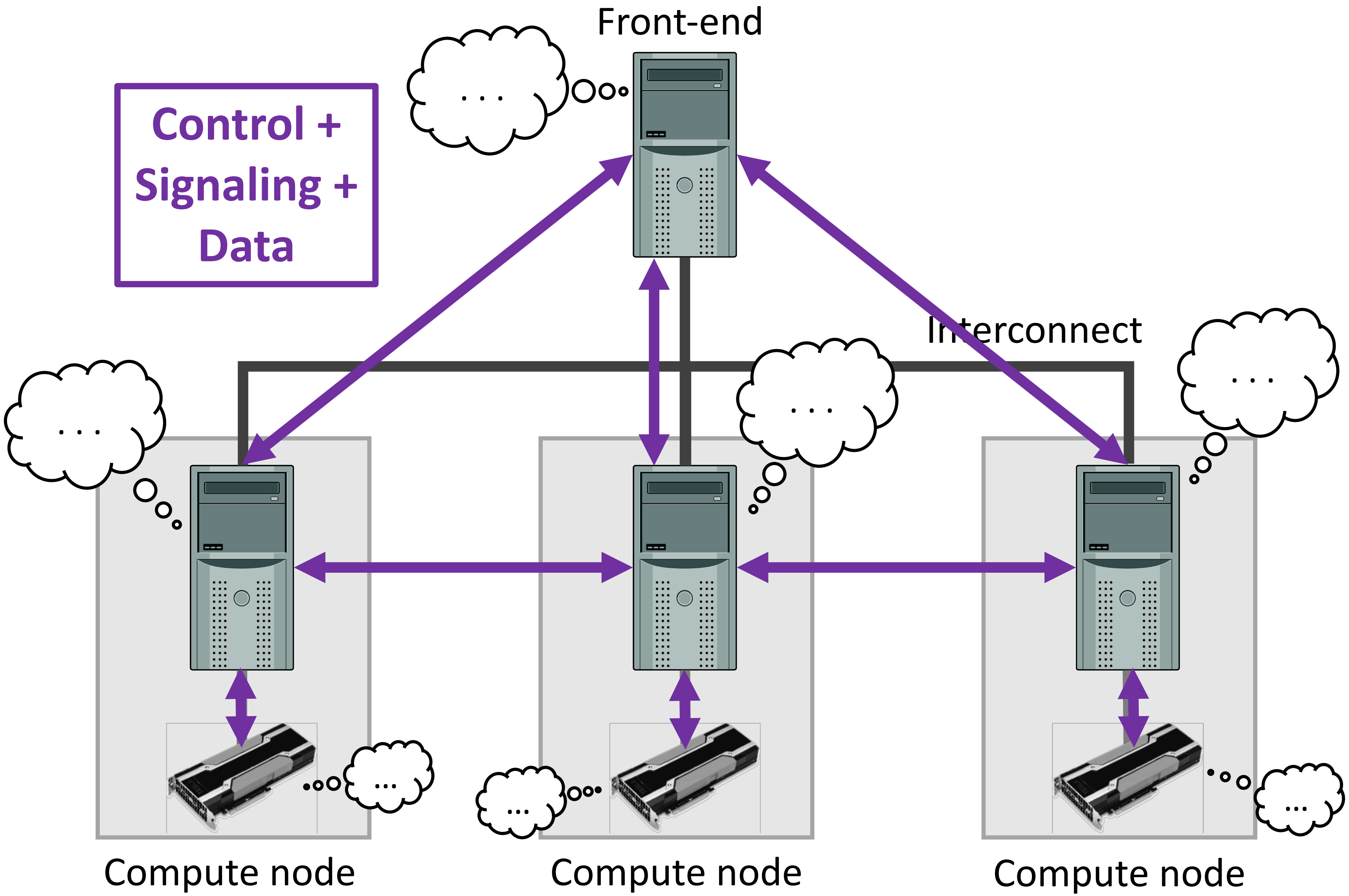


Outline

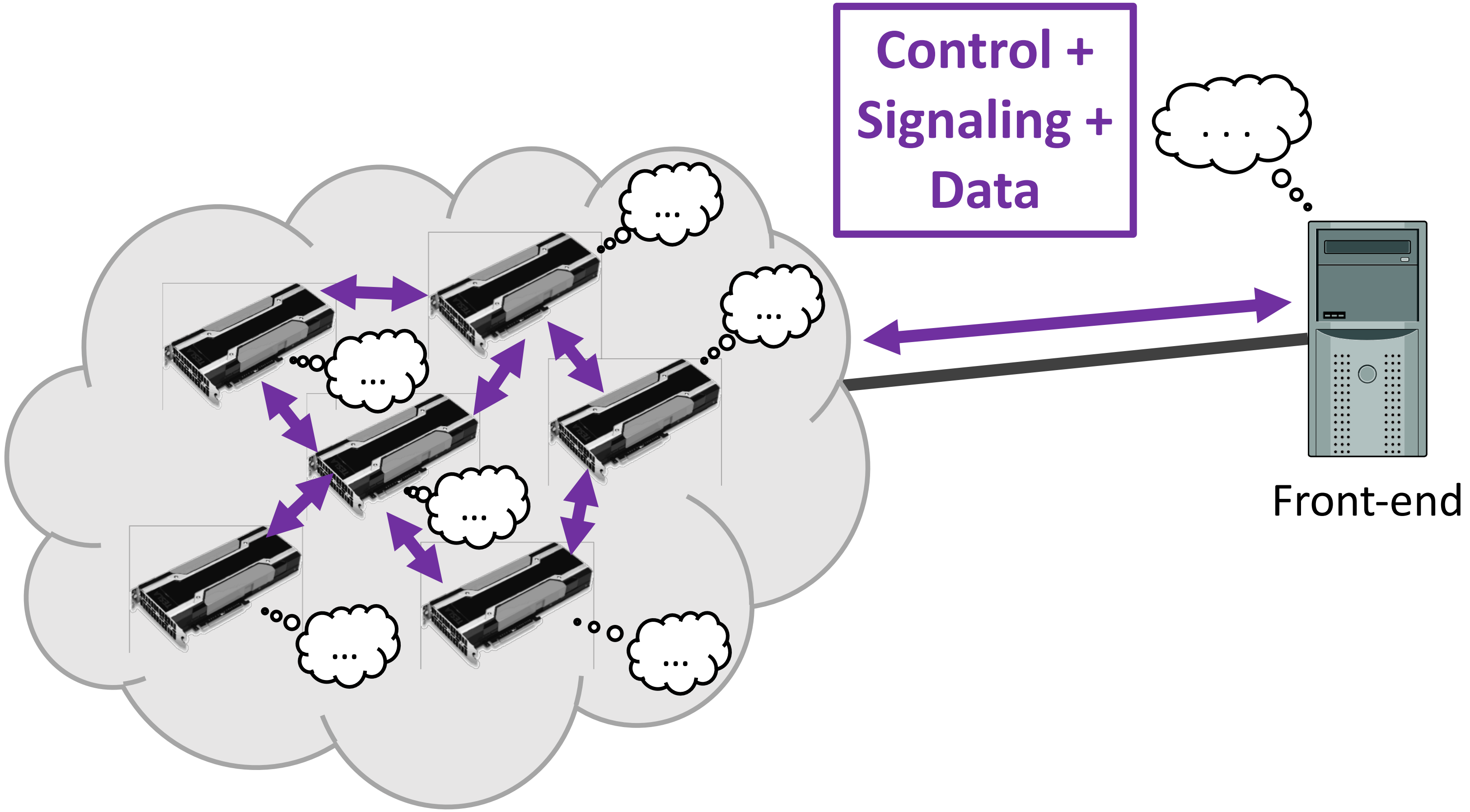
- Motivation
- PCIe Overview
- Non-Transparent Bridges
- Dolphin SmartIO
    - Example Application
    - NVMe sharing
- SmartIO in Virtual Machines

DOLPHIN INTERCONNECT SOLUTIONS
MXS824 24 PORT PCIE GEN3 SWITCH
PWR
SD CARD

Distributed applications may need to access and use IO resources that are physically located inside remote hosts



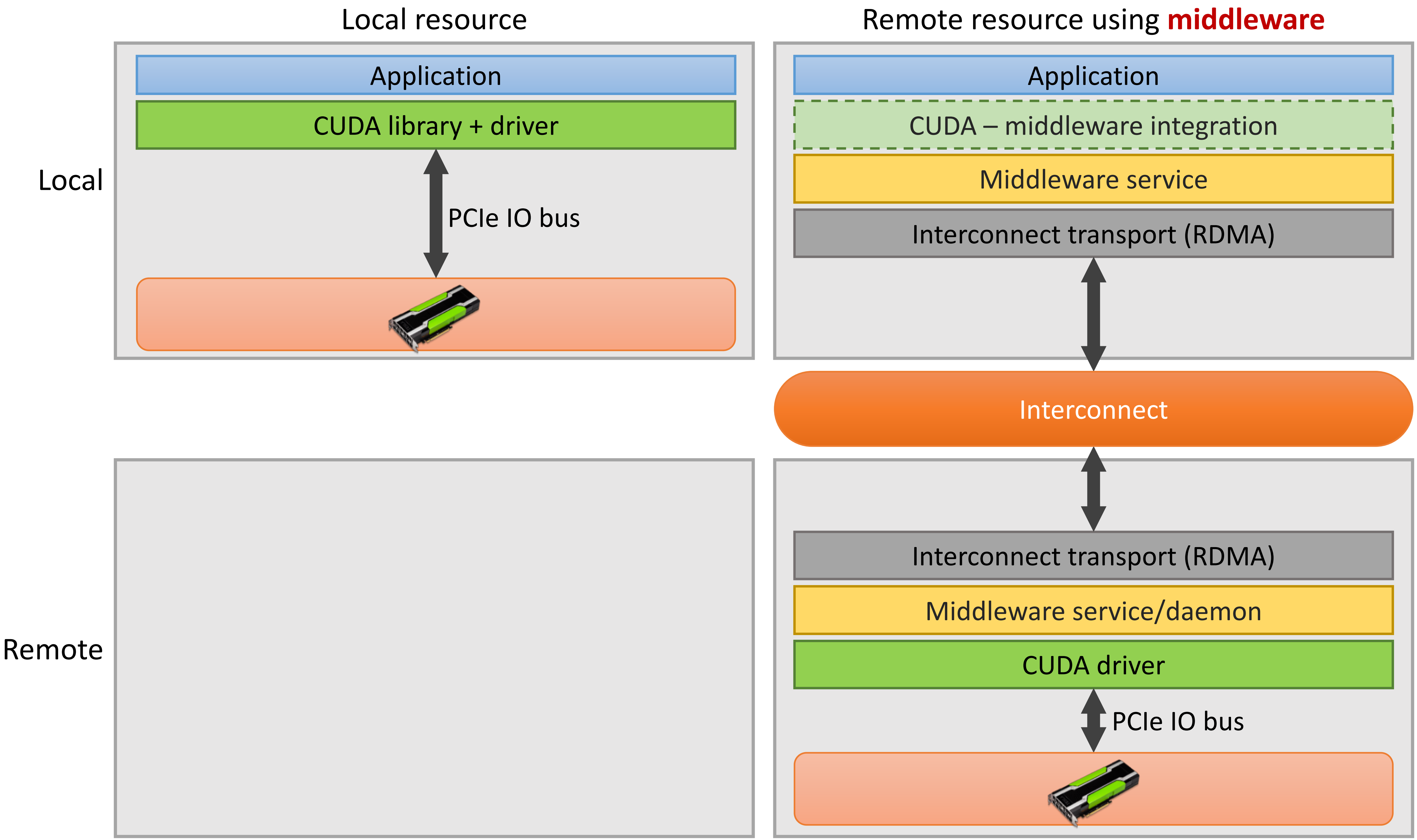
Software abstractions simplify the use and allocation of resources in a cluster and facilitate development of distributed applications



Logical view of resources

Handled in software

- rCUDA
- CUDA-aware Open MPI
- Custom GPUDirect RDMA implementation
- . . .



Local resource

Remote resource using **middleware**

Local

Application

CUDA library + driver

PCIe IO bus

Application

CUDA – middleware integration

Middleware service

Interconnect transport (RDMA)

Interconnect

Remote

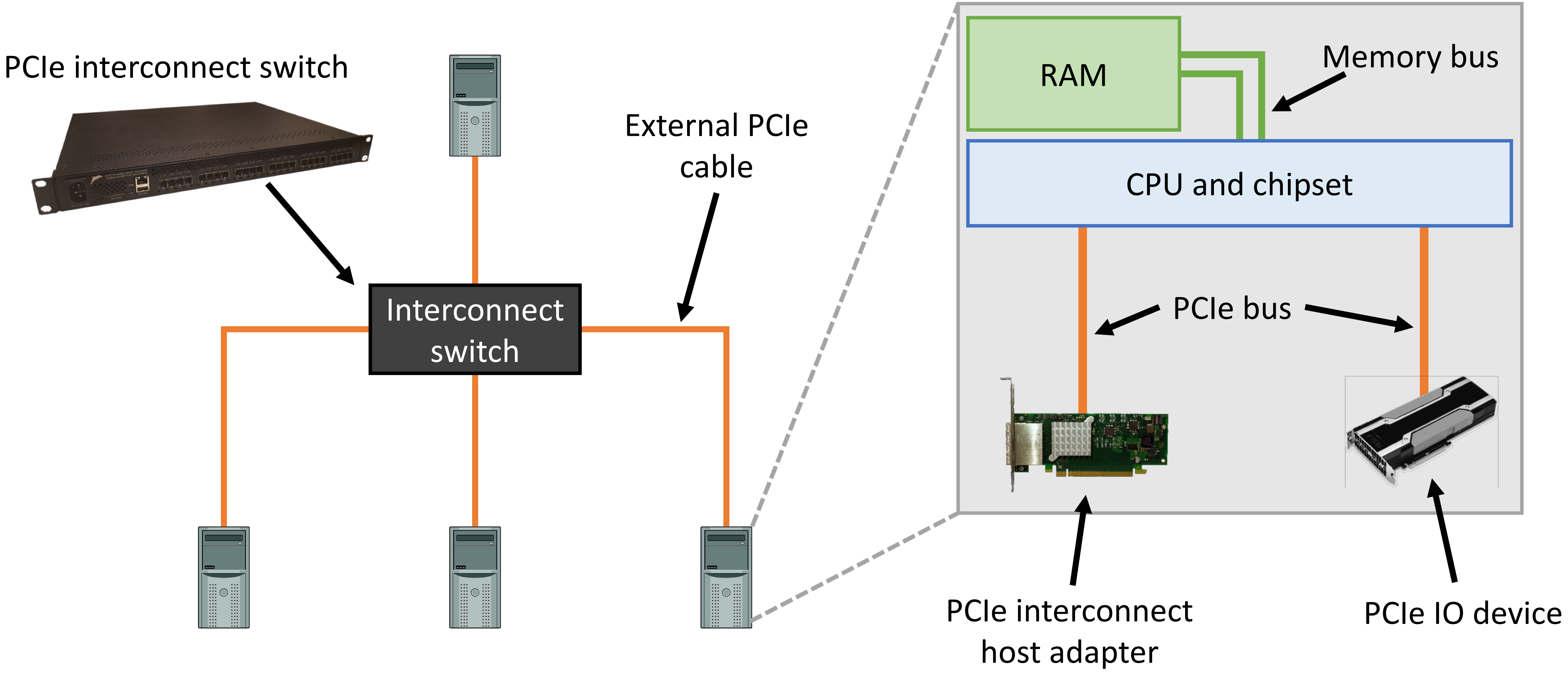
Interconnect transport (RDMA)

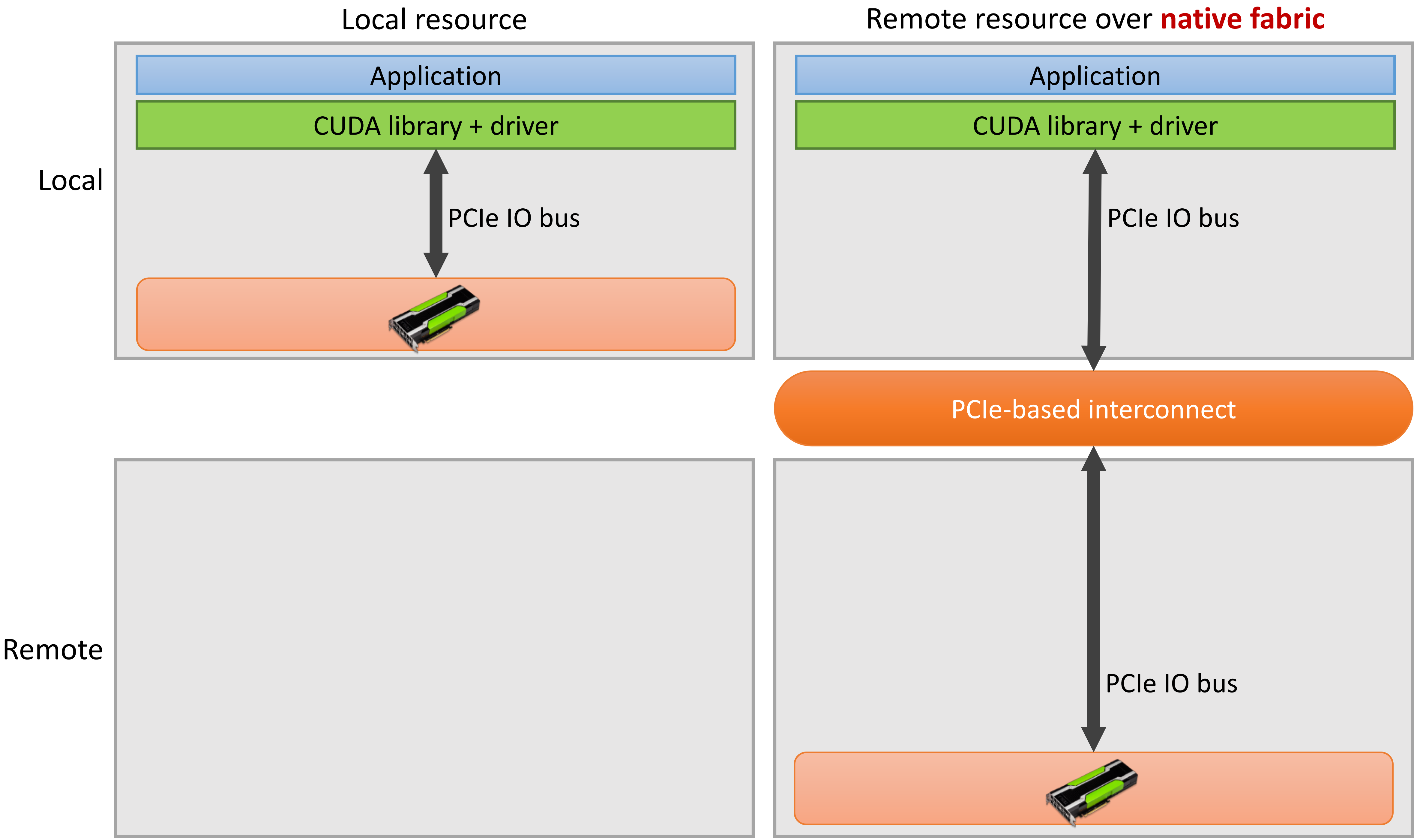
Middleware service/daemon

CUDA driver

PCIe IO bus

In PCIe clusters, the same fabric is used both as local IO bus within a single node and as the interconnect between separate nodes





Local resource

Remote resource over **native fabric**

Local

Remote

Application

CUDA library + driver

PCIe IO bus

Application

CUDA library + driver

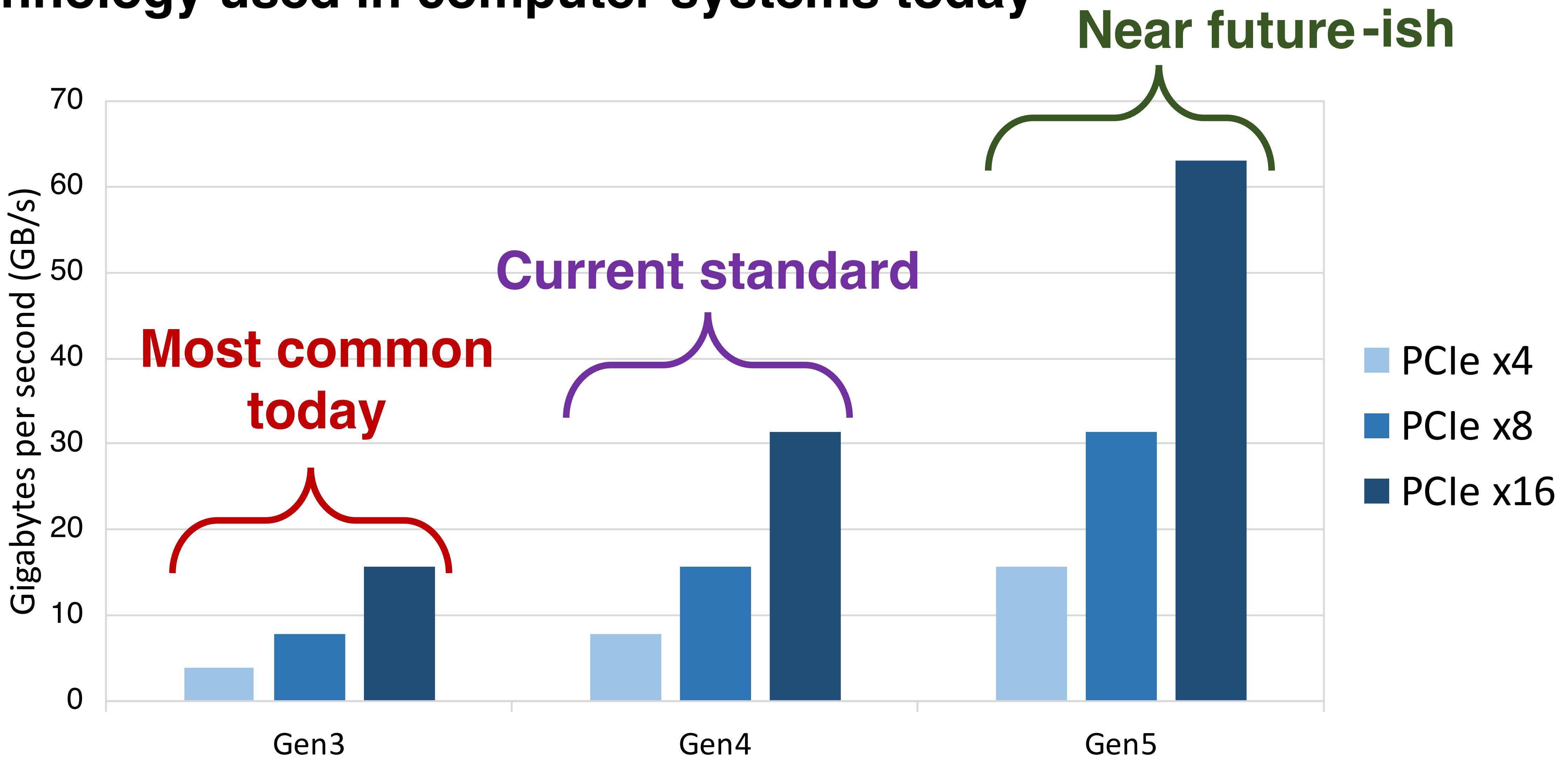
PCIe IO bus

PCIe-based interconnect

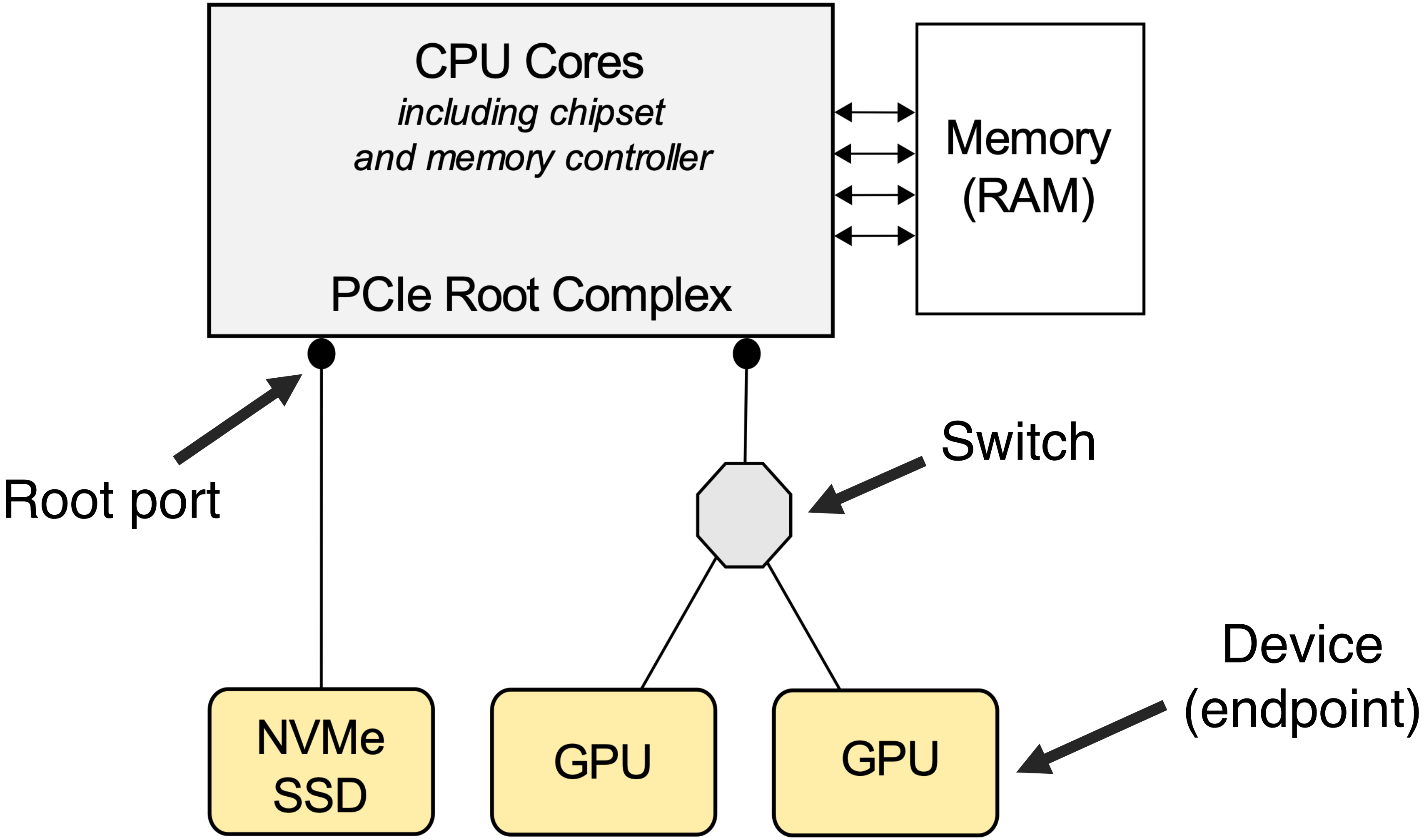
PCIe IO bus

PCIe Overview

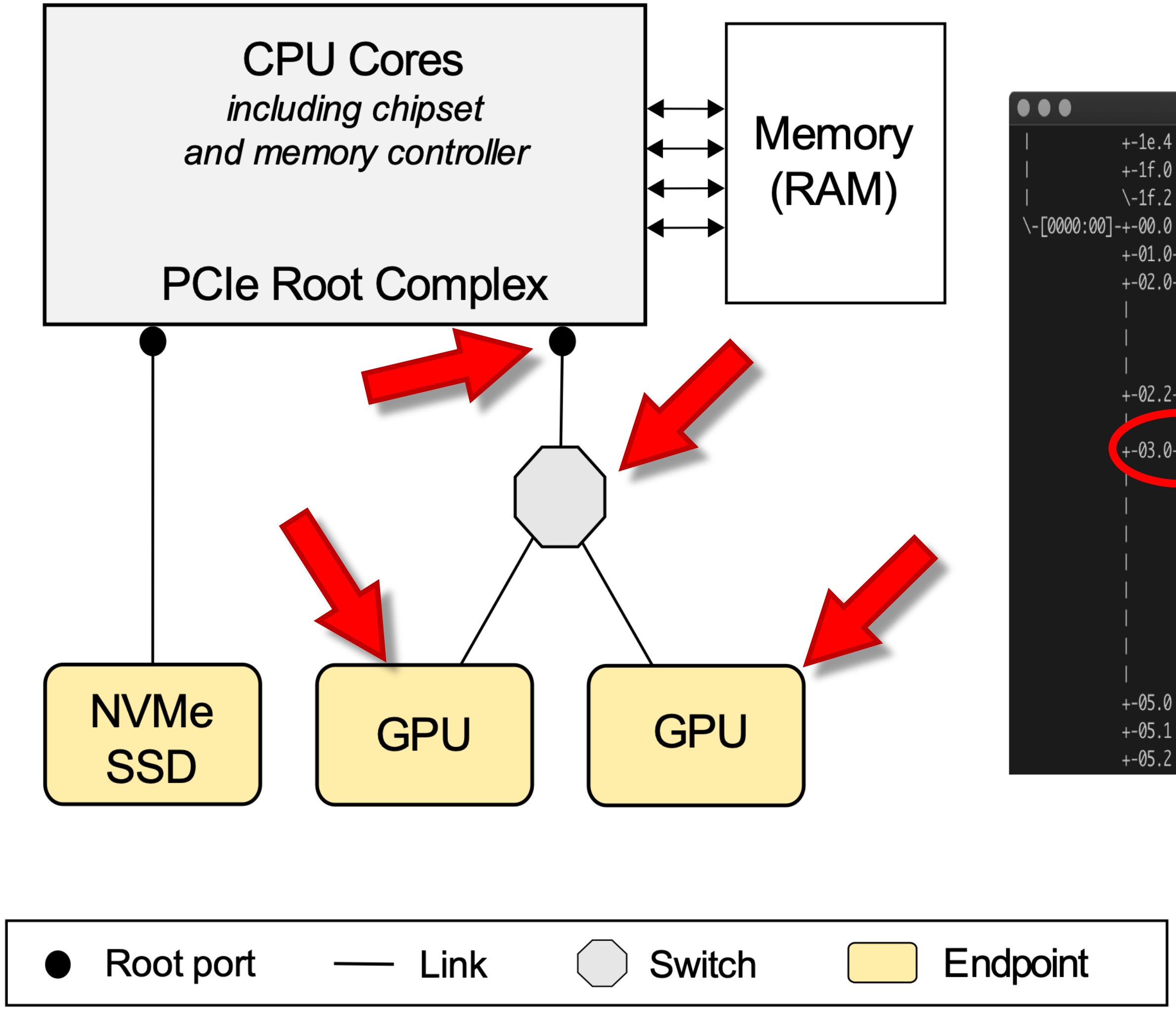
PCI Express (PCIe) is the most widely adopted I/O interconnection technology used in computer systems today



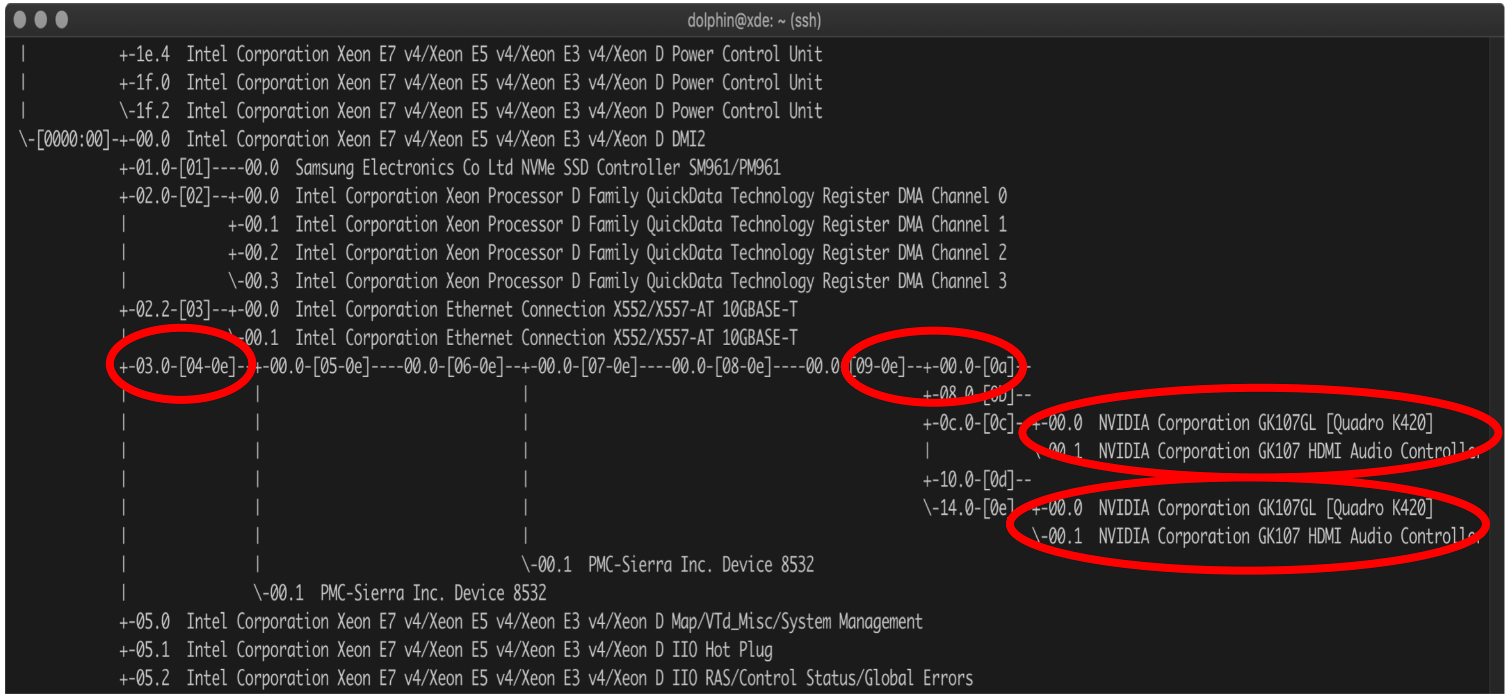
The PCIe fabric is structured as a tree, where devices form the leaf nodes (endpoints) and the CPU is on top of the root



The PCIe fabric is structured as a tree, where devices form the leaf nodes (endpoints) and the CPU is on top of the root

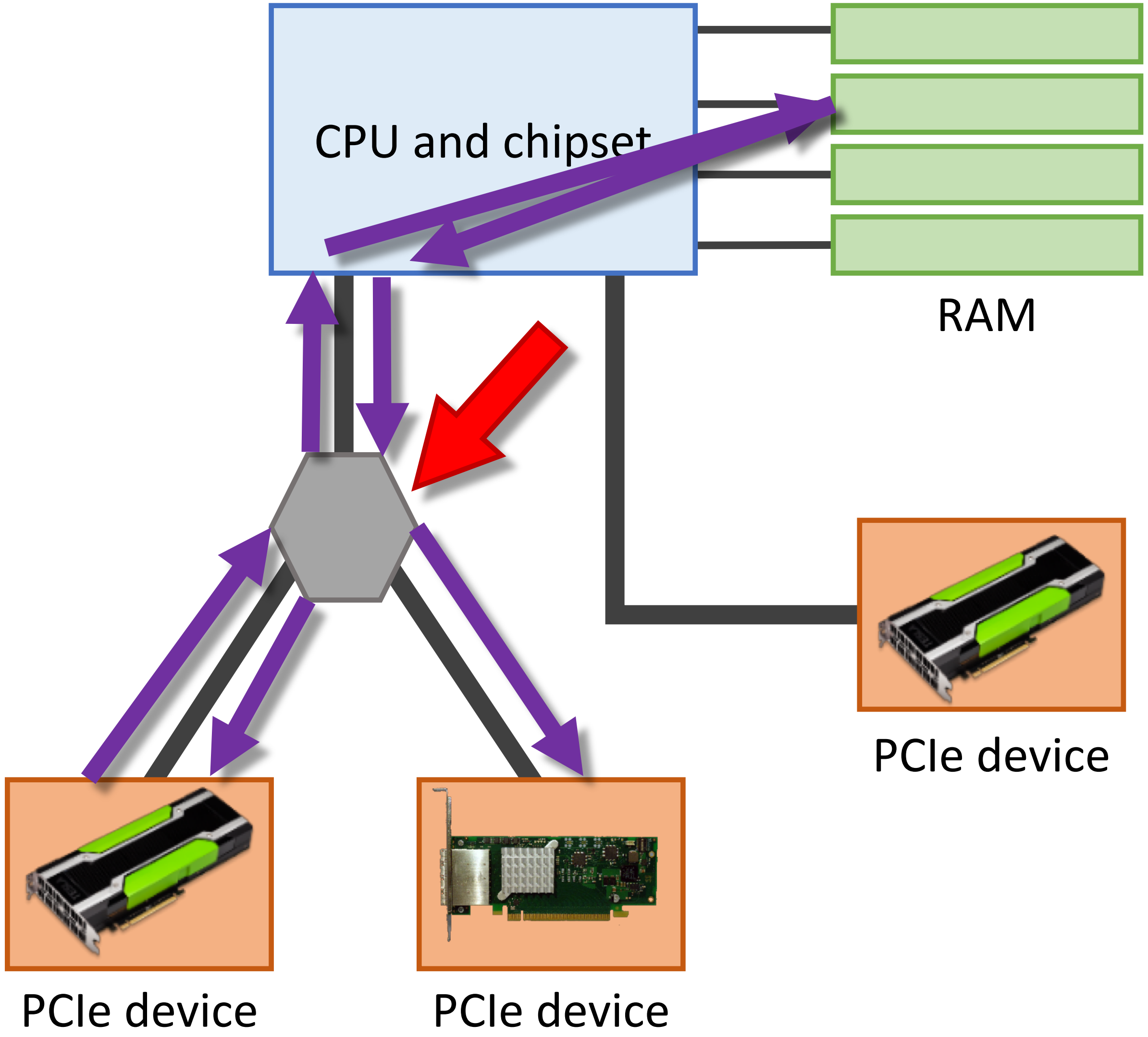


`$ lspci -tv`



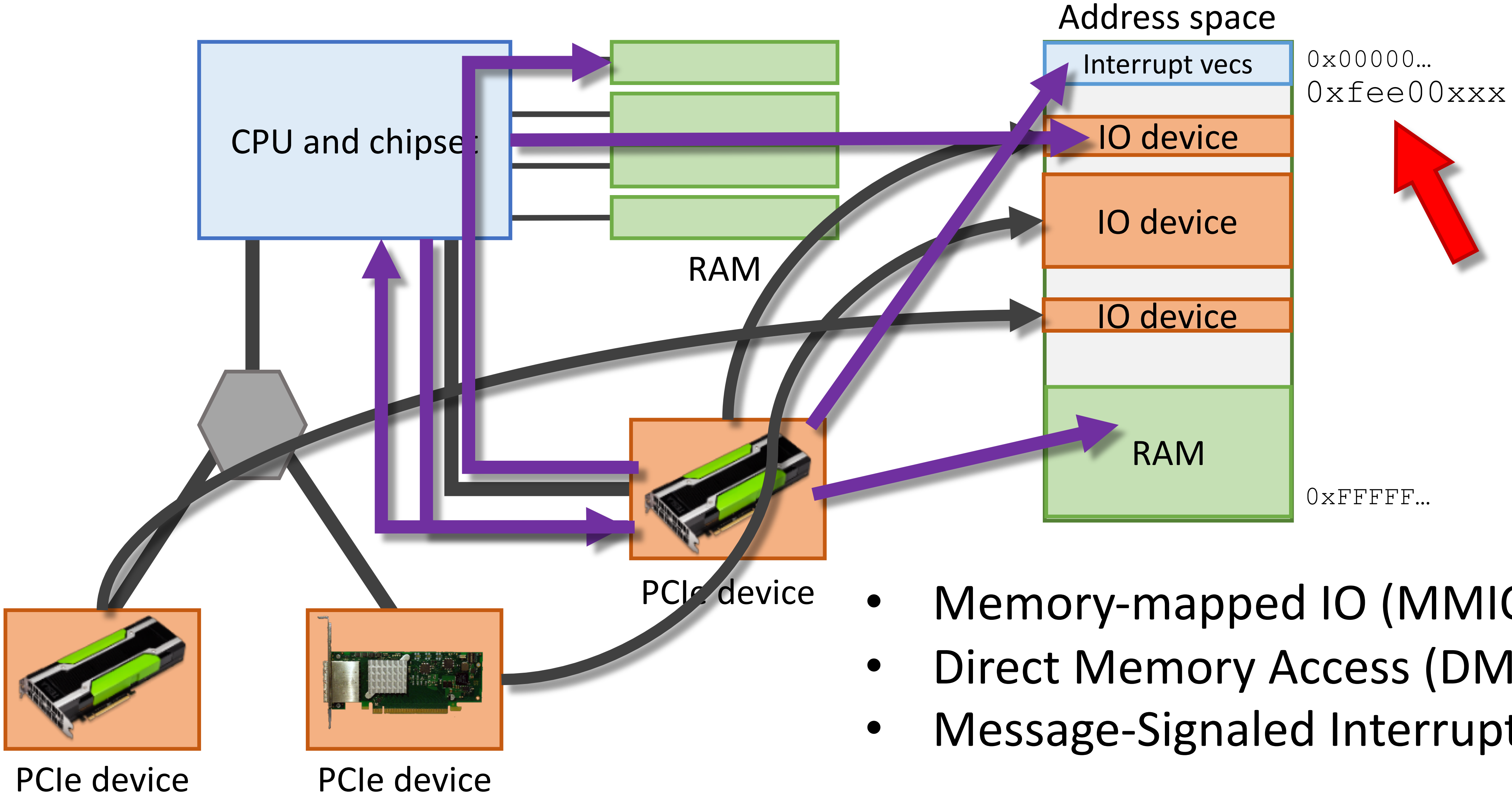
```
|           +-1e.4  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D Power Control Unit
|           +-1f.0  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D Power Control Unit
|           \-1f.2  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D Power Control Unit
\-[0000:00]-+-00.0  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D DMI2
            +-01.0-[01]----00.0  Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961
            +-02.0-[02]--+-00.0  Intel Corporation Xeon Processor D Family QuickData Technology Register DMA Channel 0
            |            +-00.1  Intel Corporation Xeon Processor D Family QuickData Technology Register DMA Channel 1
            |            +-00.2  Intel Corporation Xeon Processor D Family QuickData Technology Register DMA Channel 2
            |            \-00.3  Intel Corporation Xeon Processor D Family QuickData Technology Register DMA Channel 3
            +-02.2-[03]--+-00.0  Intel Corporation Ethernet Connection X552/X557-AT 10GBASE-T
            |            \-00.1  Intel Corporation Ethernet Connection X552/X557-AT 10GBASE-T
            +-03.0-[04-0e]--+-00.0-[05-0e]----00.0-[06-0e]--+-00.0-[07-0e]----00.0-[08-0e]----00.0-[09-0e]--+-00.0-[0a]--
            |               |                               |                                               +-08.0-[0b]--
            |               |                               |                                               +-0c.0-[0c]--+-00.0  NVIDIA Corporation GK107GL [Quadro K420]
            |               |                               |                                               |            \-00.1  NVIDIA Corporation GK107 HDMI Audio Controlle
            |               |                               |                                               +-10.0-[0d]--
            |               |                               |                                               \-14.0-[0e]--+-00.0  NVIDIA Corporation GK107GL [Quadro K420]
            |               |                               |                                                            \-00.1  NVIDIA Corporation GK107 HDMI Audio Controlle
            |               |                               \-00.1  PMC-Sierra Inc. Device 8532
            |               \-00.1  PMC-Sierra Inc. Device 8532
            +-05.0  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D Map/VTd_Misc/System Management
            +-05.1  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D IIO Hot Plug
            +-05.2  Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeon D IIO RAS/Control Status/Global Errors
```

Memory reads and writes are handled by PCIe as transactions that are packet-switched through the fabric depending on the address



- Upstream
- Downstream
- Peer-to-peer (shortest path)

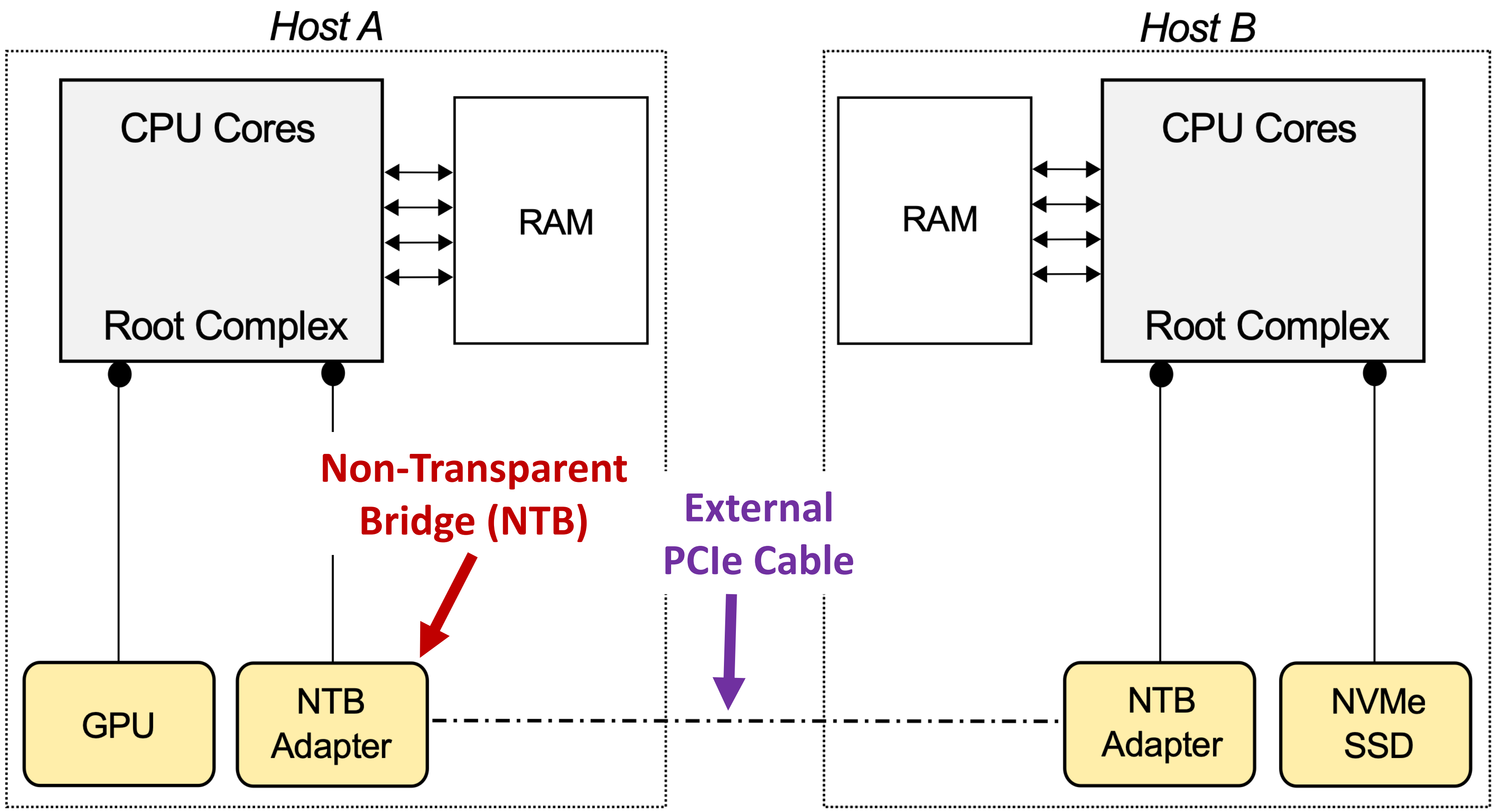
IO devices and the CPU share the same physical address space, allowing devices to access system memory and other devices



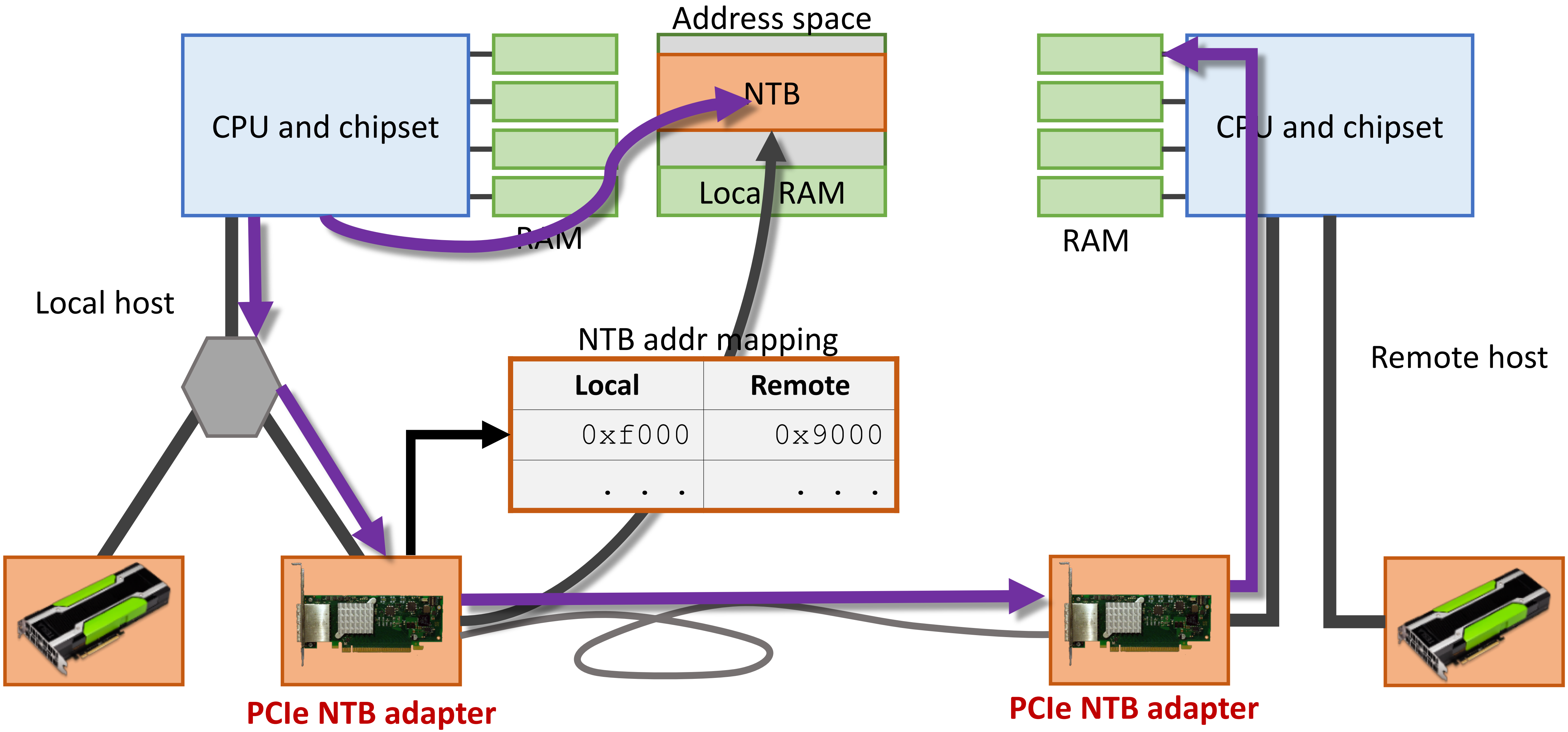
- Memory-mapped IO (MMIO / PIO)
- Direct Memory Access (DMA)
- Message-Signaled Interrupts (MSI-X)

Non-Transparent Bridges

We can interconnect separate PCIe root complexes and translate addresses between them using a non-transparent bridge (NTB)

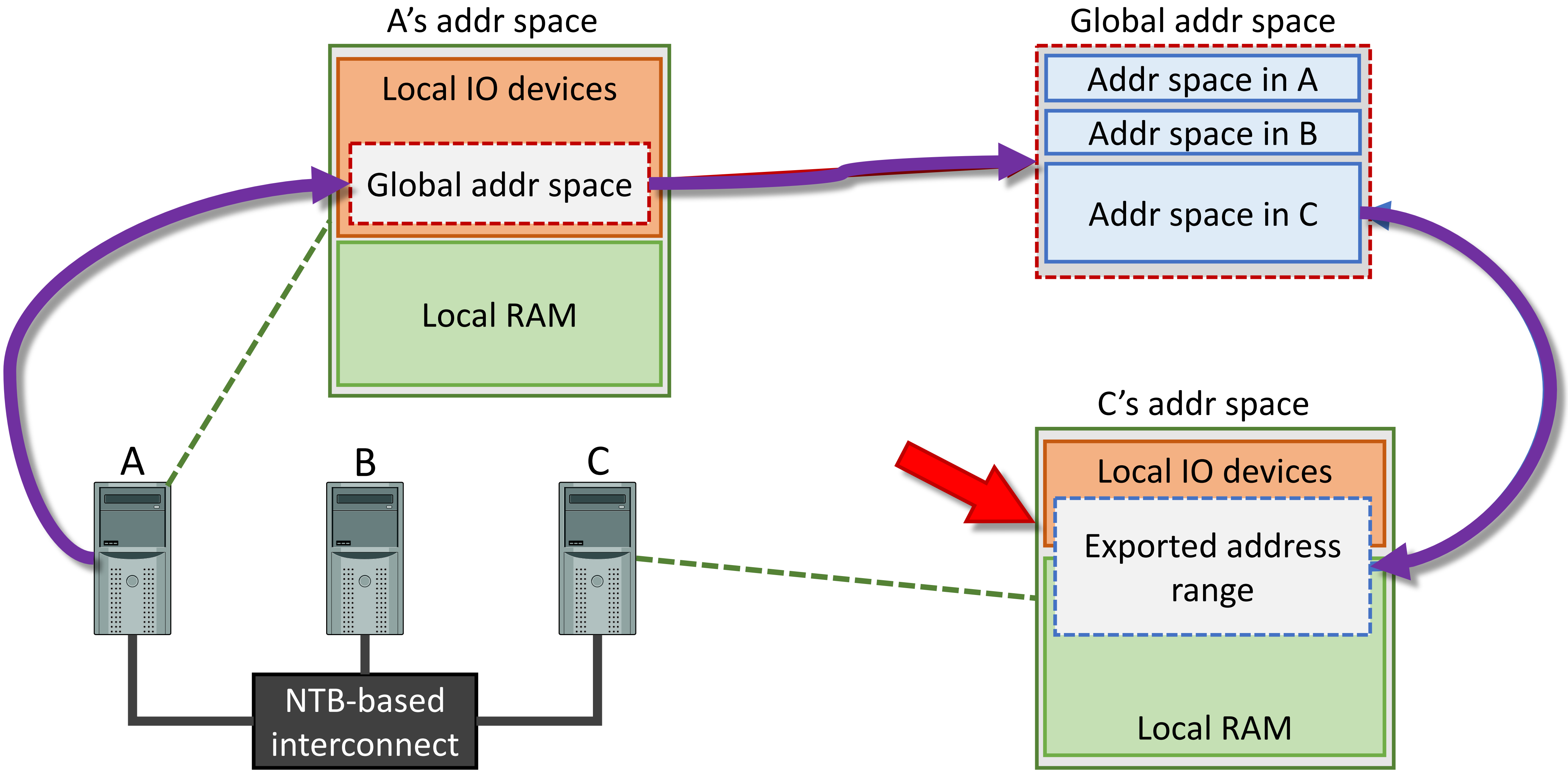


Remote address space can be mapped into local address space by using PCIe Non-Transparent Bridges (NTBs)



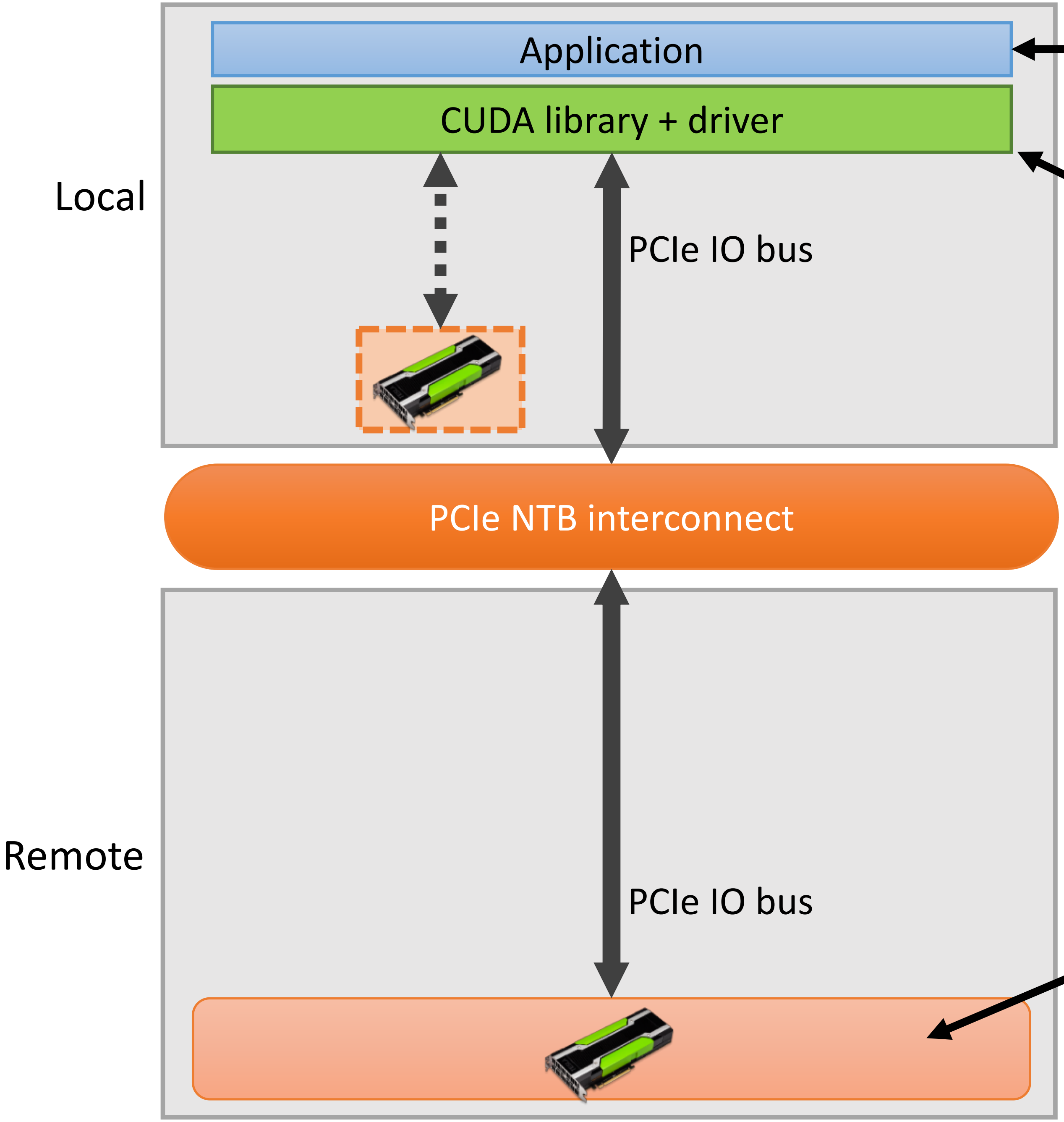
Local	Remote
0xf000	0x9000
. . .	. . .

Using NTBs, each node in the cluster take part in a shared address space and have their own "window" into the global address space



SmartIO

Borrowed remote resource



Local

Application

CUDA library + driver

PCIe IO bus

PCIe NTB interconnect

Remote

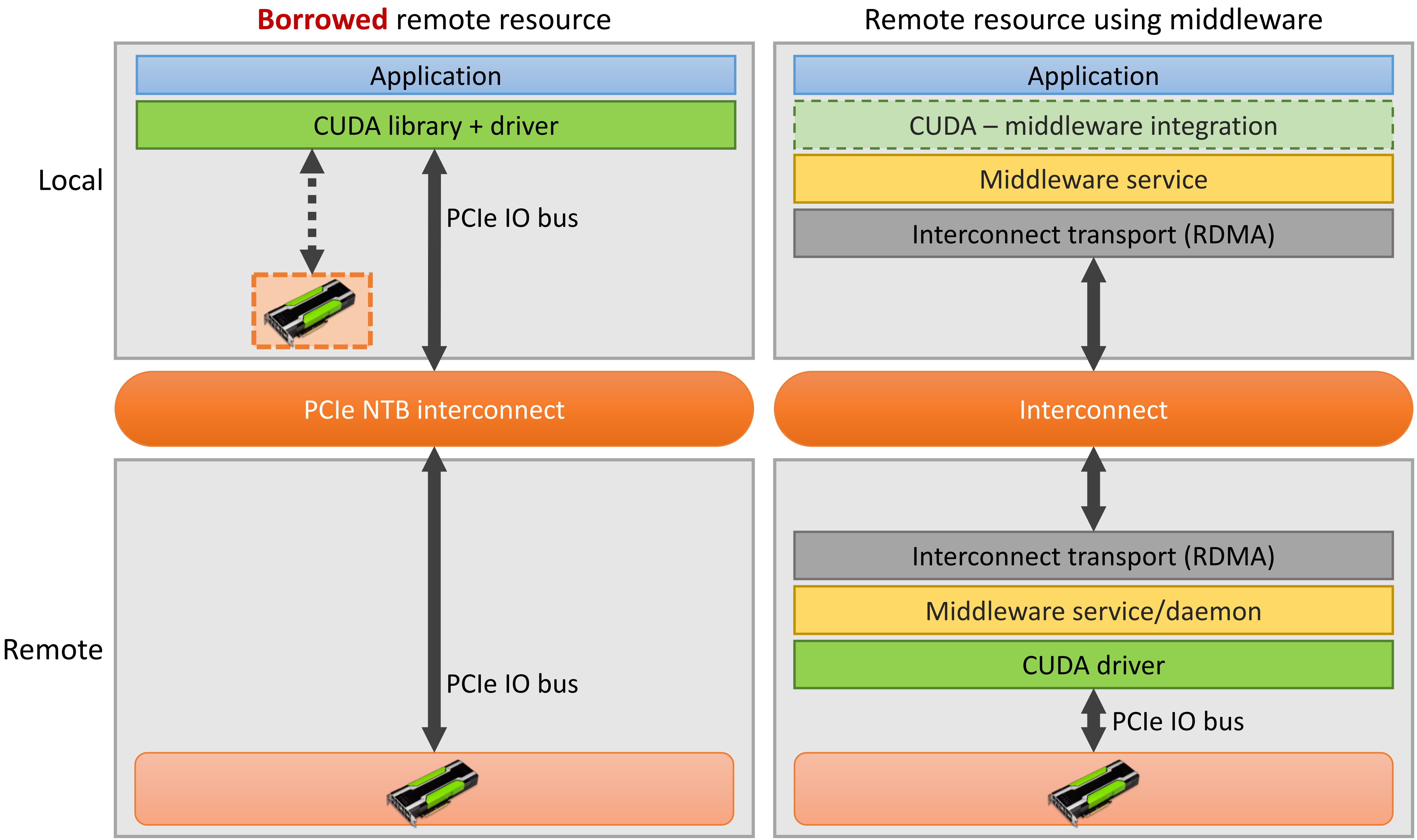
PCIe IO bus

Resource appears local to OS, driver, and app

Unmodified local driver
(with hot-plug support)

Hardware mappings ensure **fast data path**

Works with **any PCIe device**
(even individual SR-IOV functions)



Borrowed remote resource

Remote resource using middleware

Local

Remote

Application

CUDA library + driver

PCIe IO bus

PCIe NTB interconnect

PCIe IO bus

Application

CUDA – middleware integration

Middleware service

Interconnect transport (RDMA)

Interconnect

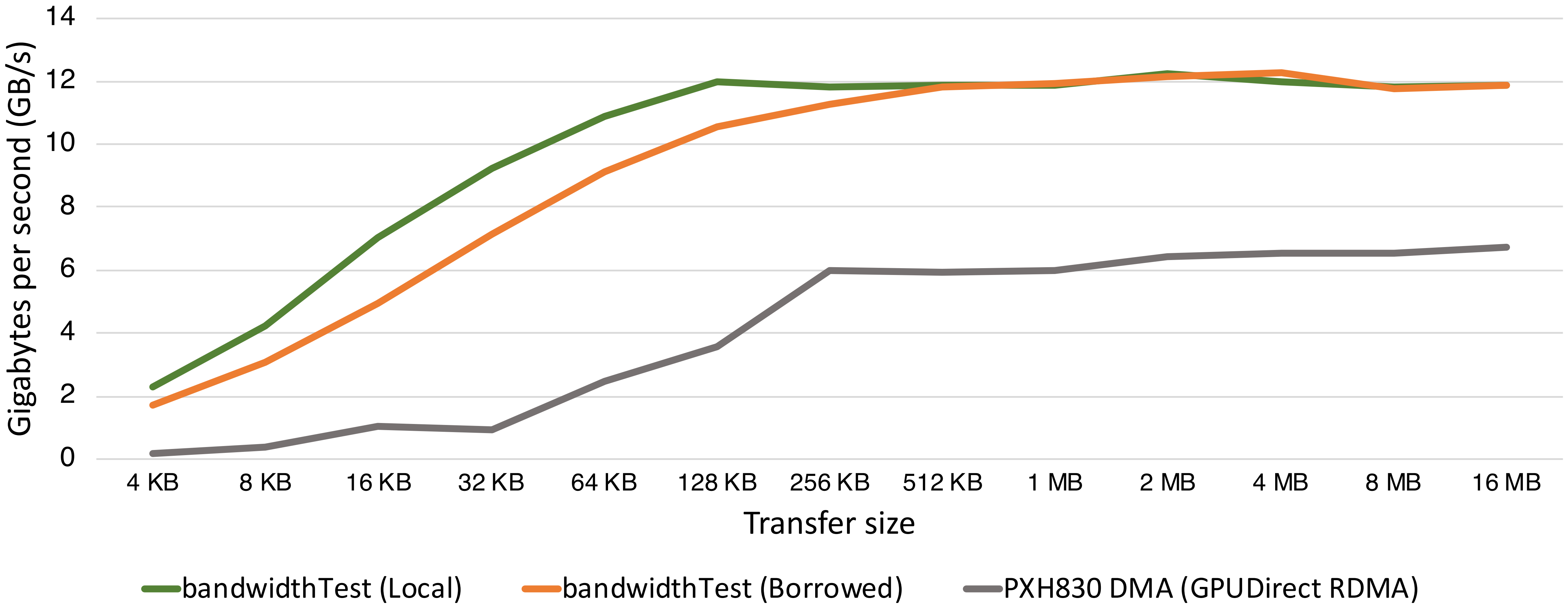
Interconnect transport (RDMA)

Middleware service/daemon

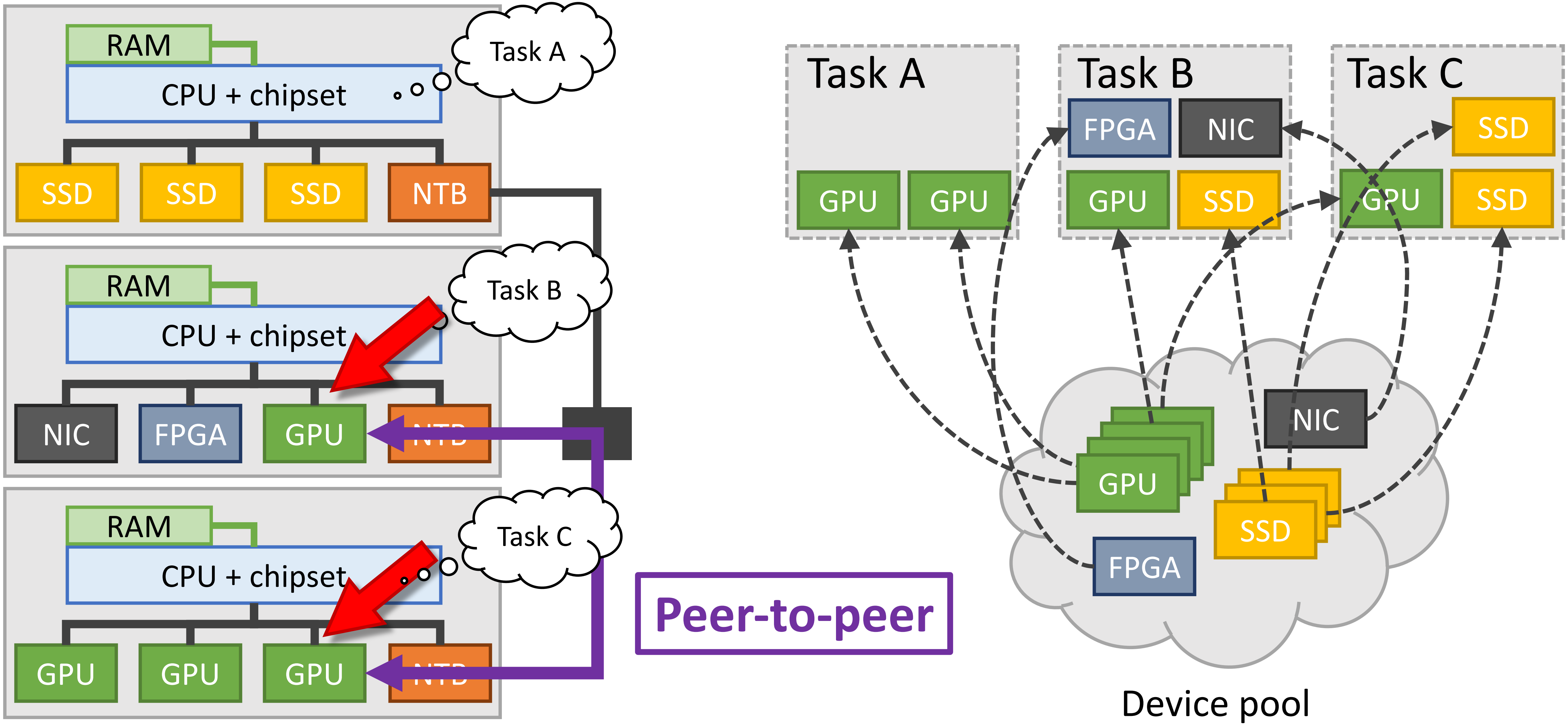
CUDA driver

PCIe IO bus

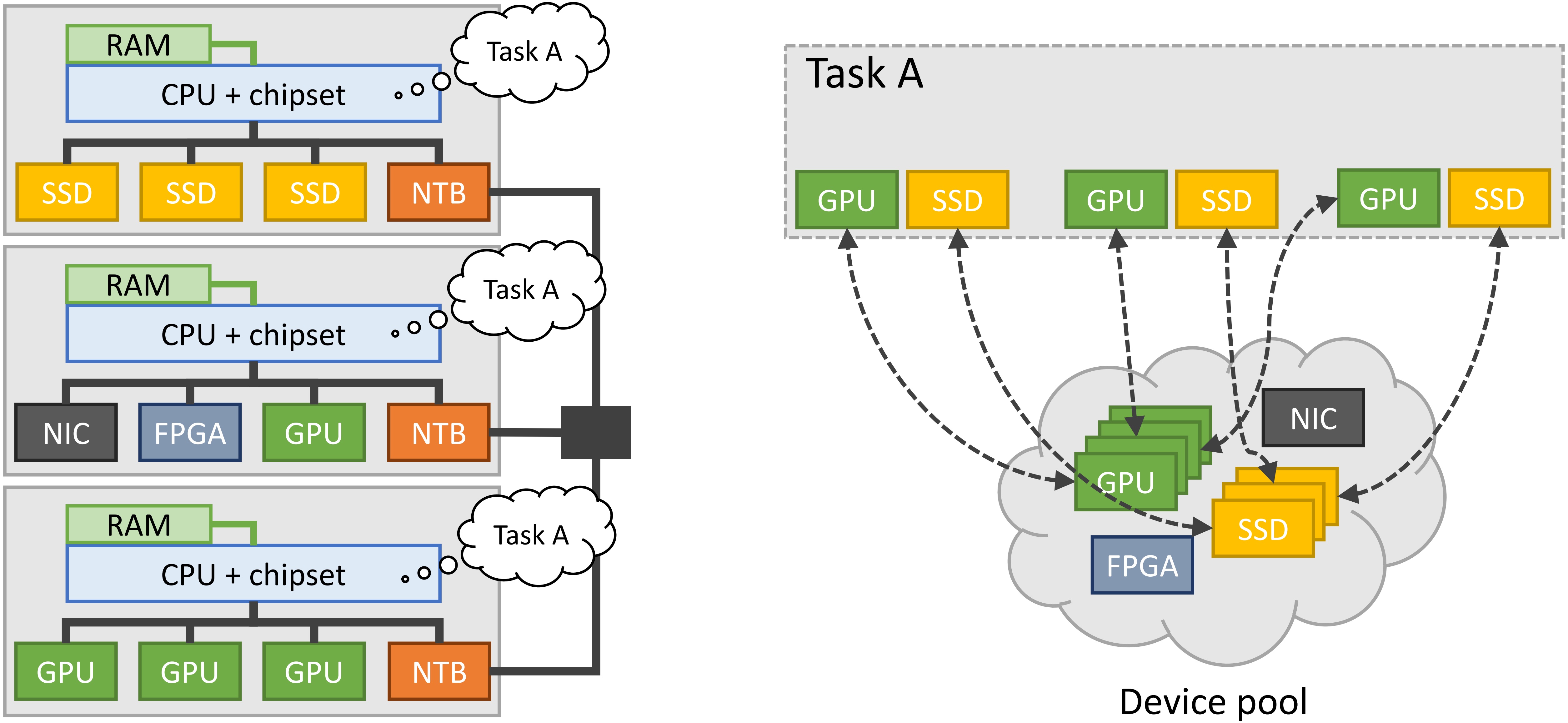
Device to host transfers: Comparing local to borrowed GPU



Using Device Lending, nodes in a PCIe cluster can share resources through a process of borrowing and giving back devices



Using Device Lending, nodes in a PCIe cluster can share resources through a process of borrowing and giving back devices

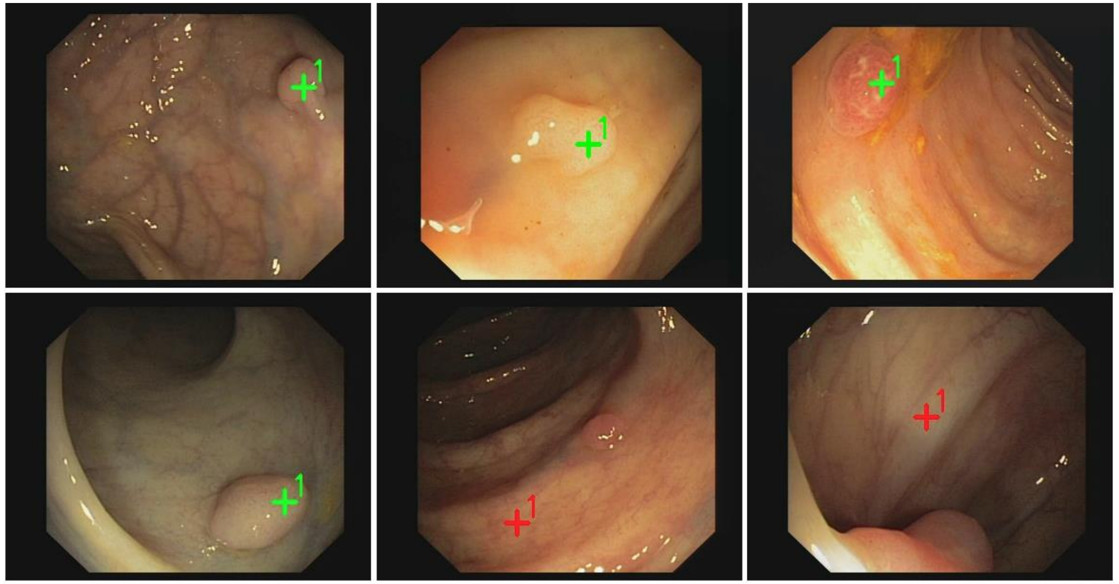


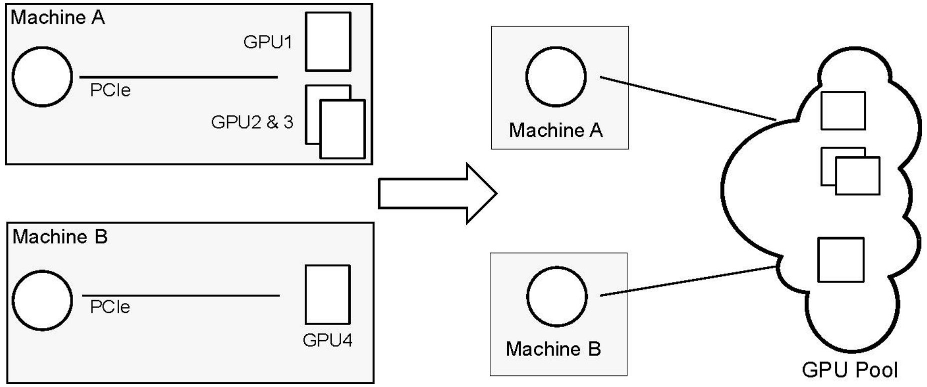
Example Application

Processing of Medical Videos

P9258 - Efficient Processing of Medical Videos in a Multi-auditory Environment Using GPU Lending

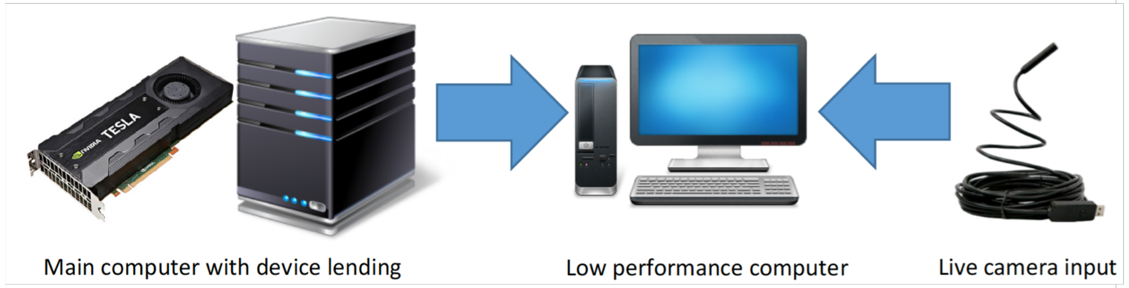
Scenario: Real-time computer-aided polyp detection

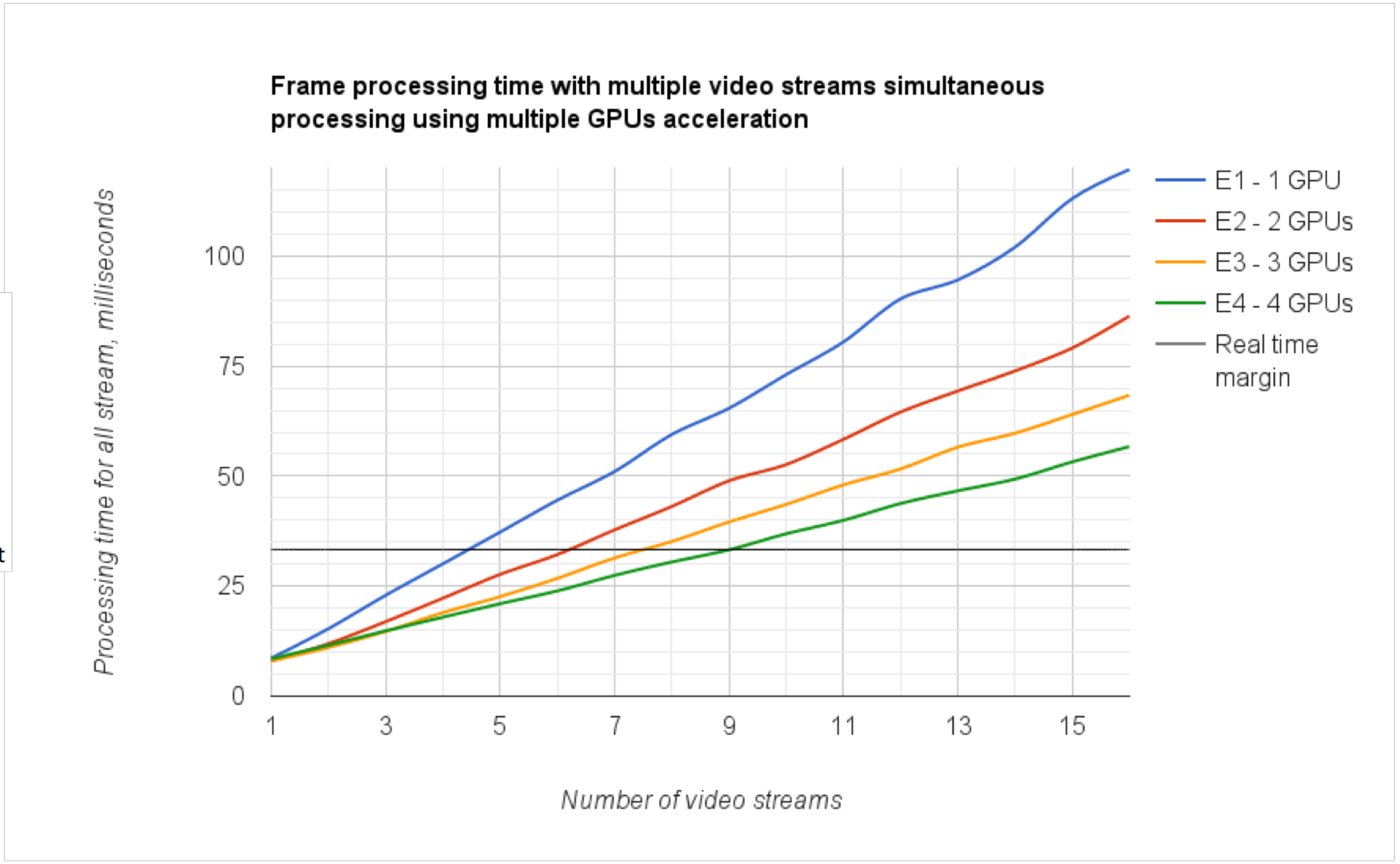




- PCIe fiber cables can be up to 100 meters.
- Enable ”thin clients” to use GPUs in remote machine room

Flexible sharing of GPU resources between multiple examination rooms




Frame processing time with multiple video streams simultaneous processing using multiple GPUs acceleration

- System uses a combination of classic computer vision algorithms and machine learning.
- Research prototype since 2016.

Sharing of NVMe drives

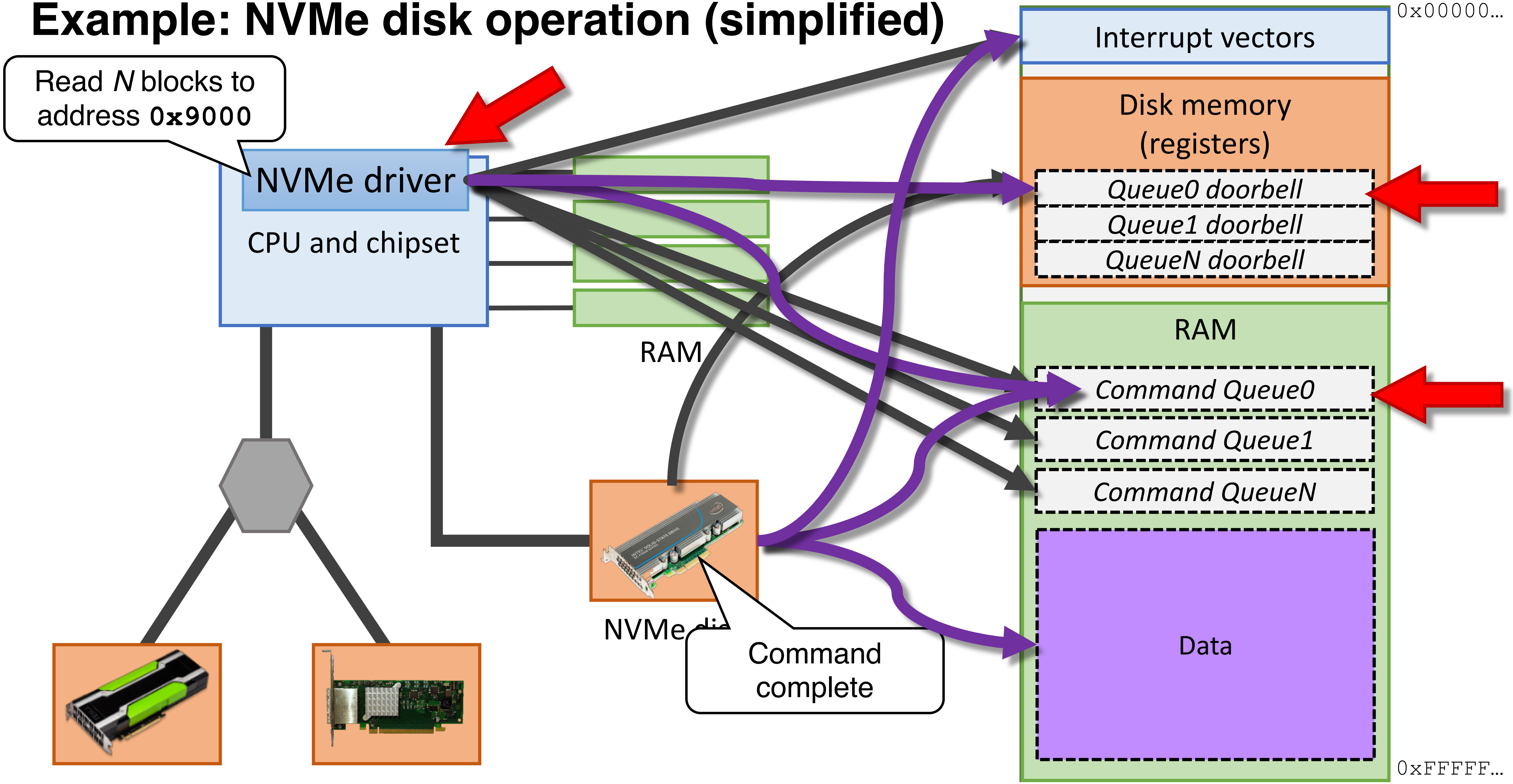
For more details:

S9563 - Efficient Distributed Storage I/O using NVMe and GPU Direct in a PCIe Network

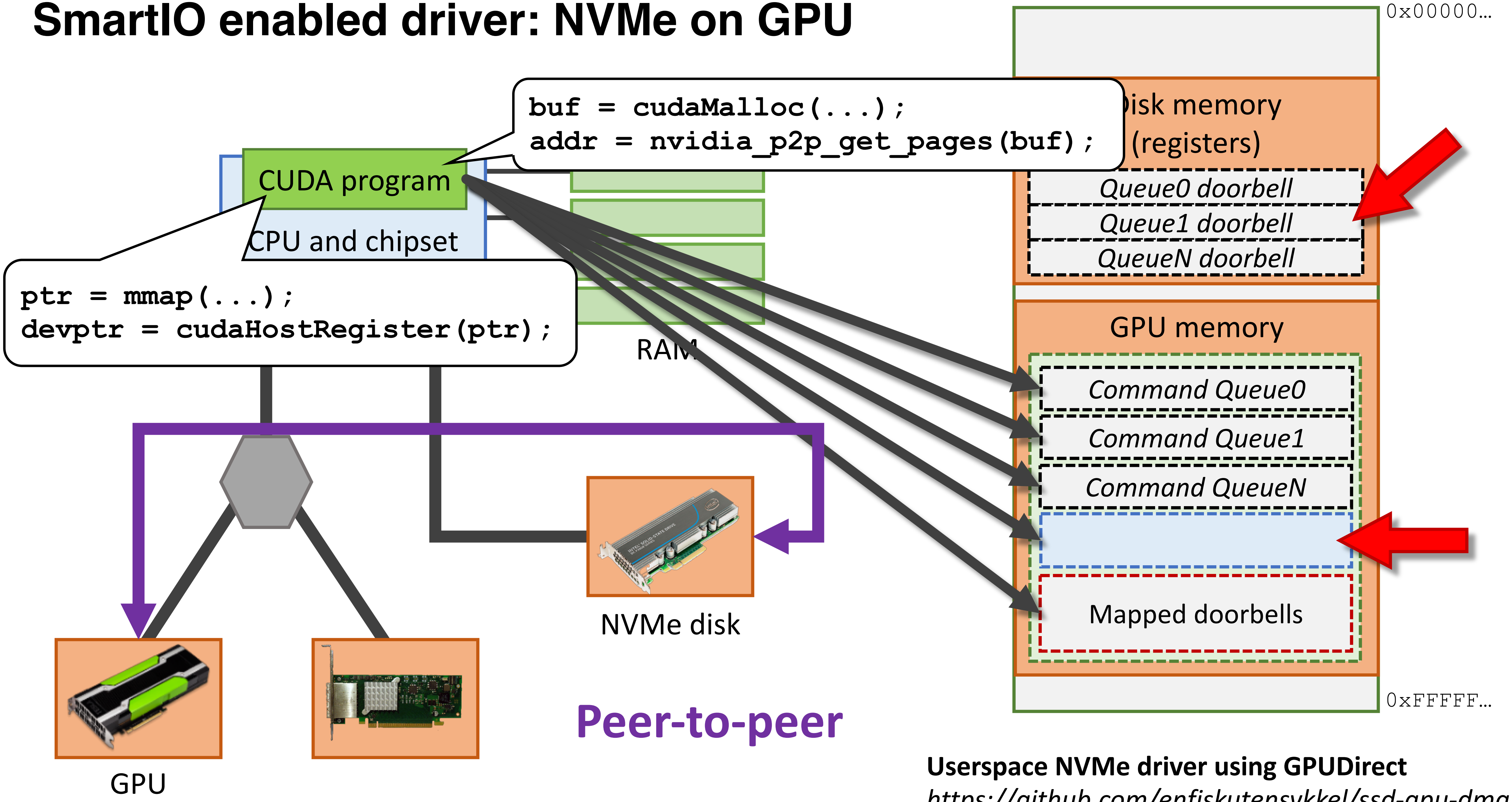
or

Visit Dolphin Interconnect Solutions in booth 1520

Example: NVMe disk operation (simplified)



SmartIO enabled driver: NVMe on GPU

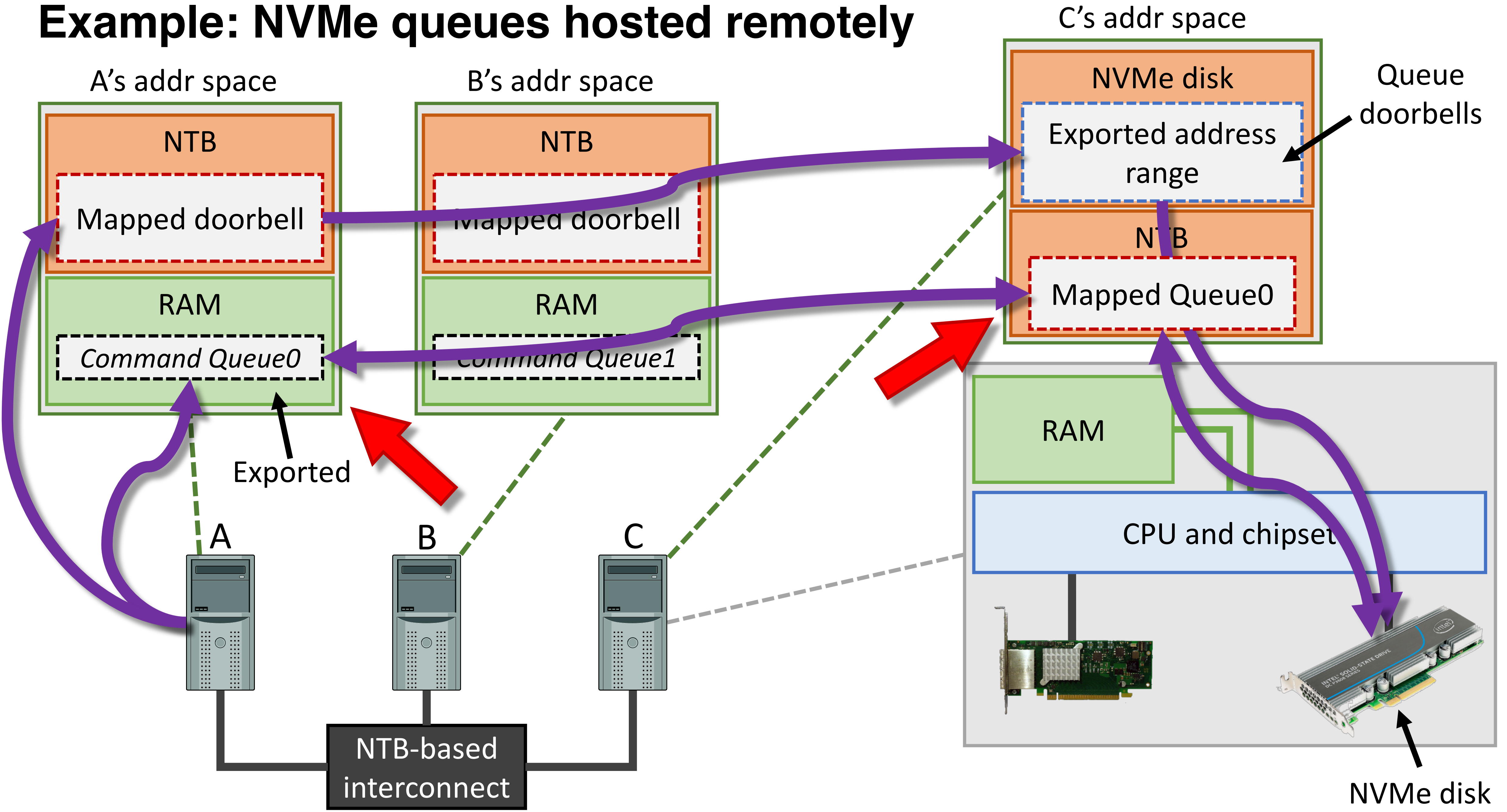


```
buf = cudaMalloc(...);
addr = nvidia_p2p_get_pages(buf);
```

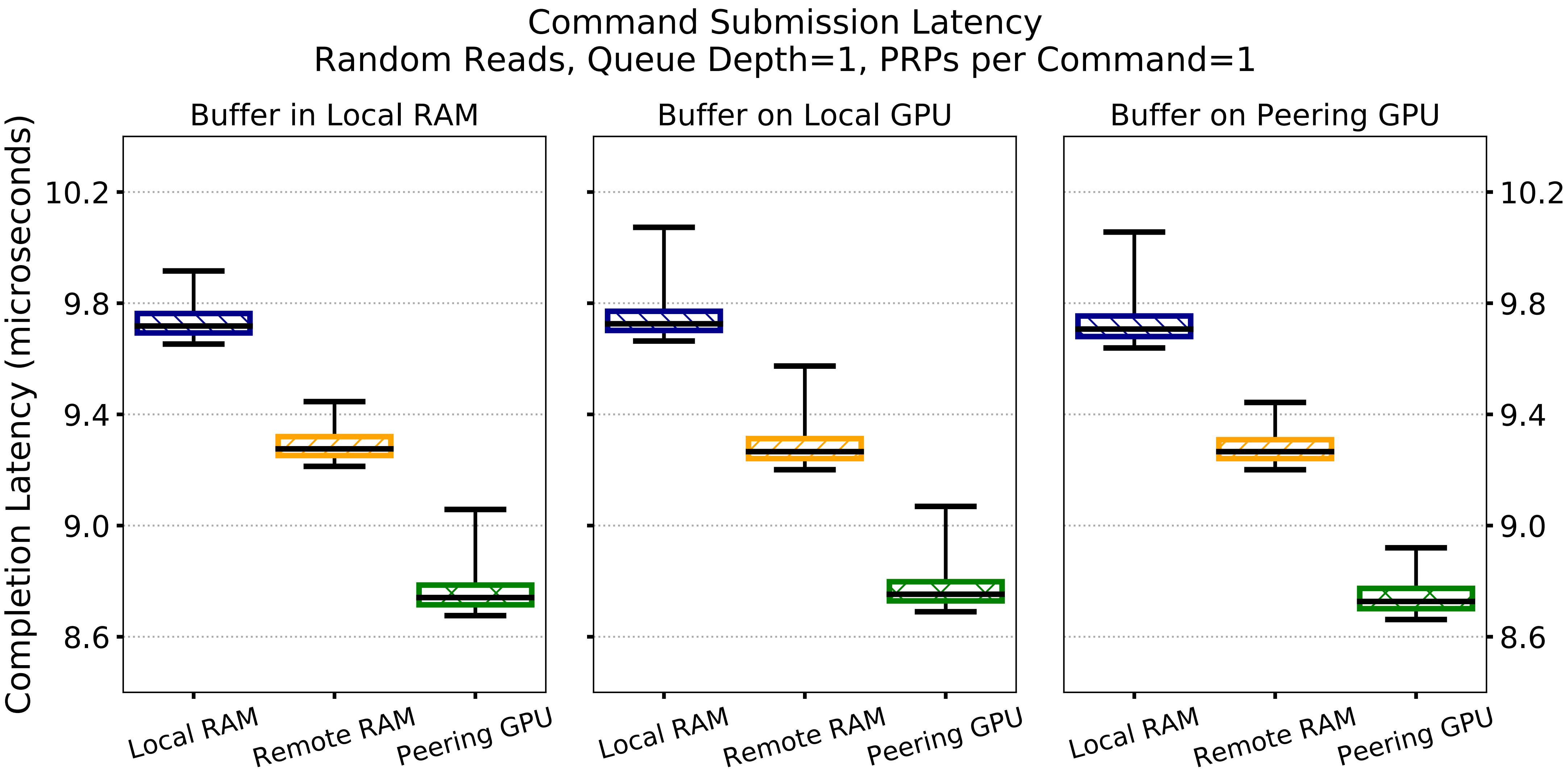
```
ptr = mmap(...);
devptr = cudaHostRegister(ptr);
```

Userspace NVMe driver using GPUDirect
https://github.com/enfiskutensykkel/ssd-gpu-dma

Example: NVMe queues hosted remotely

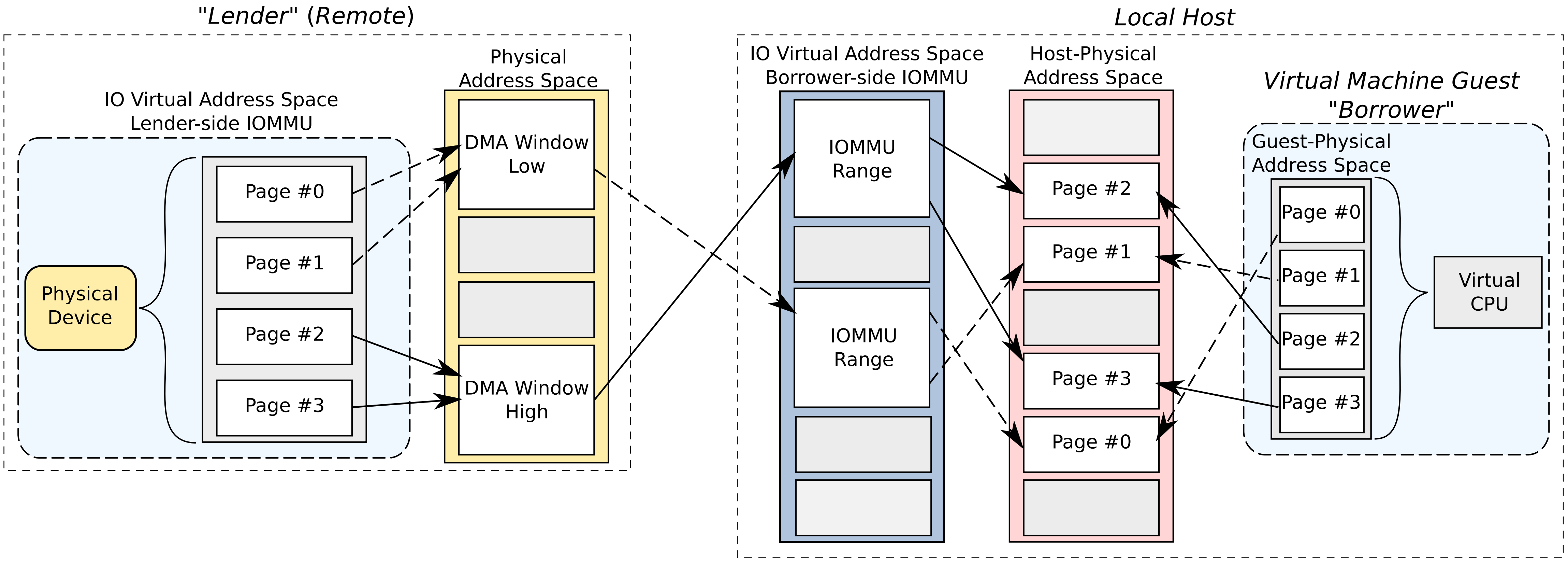


Read latency for reading blocks from a NVMe disk into a GPU: Local versus borrowed disk

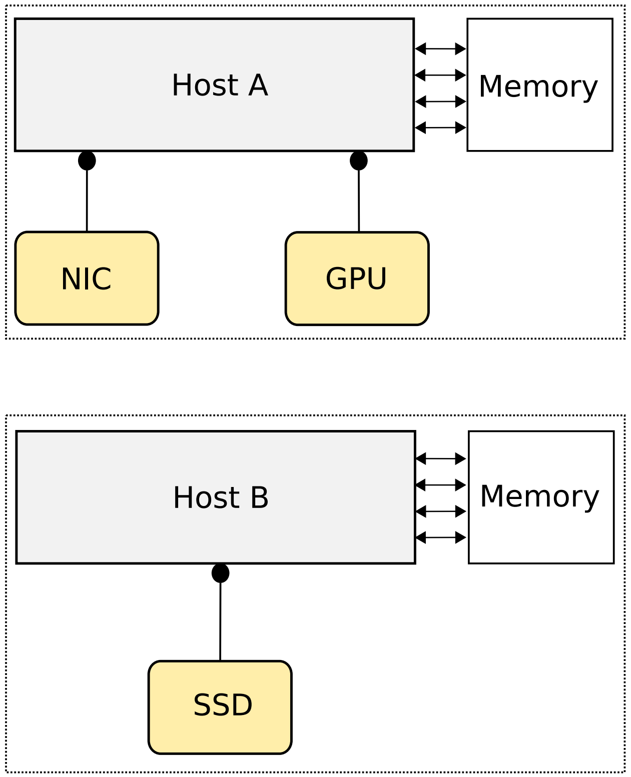


SmartIO in Virtual Machines

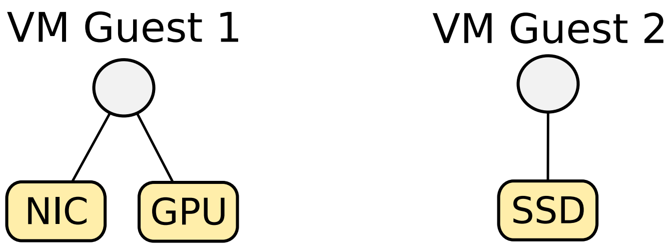
SmartIO fully supports to lend devices to virtual machines running in Linux KVM uning Virtual Function IO API (VFIO)



Pass-through allows physical devices to be used by VMs with minimal overhead, but is not as flexible as resource virtualization



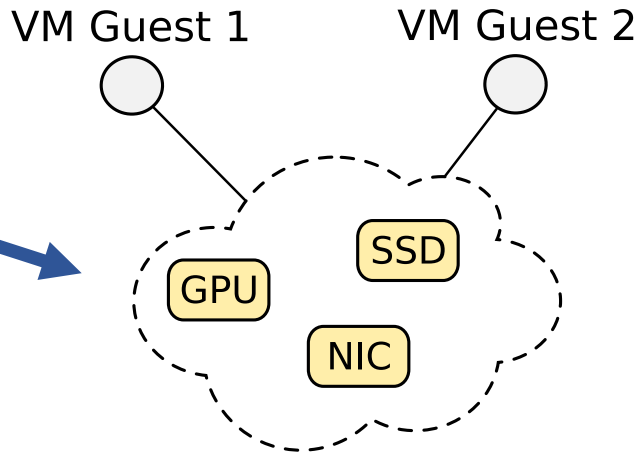
Physical View



Minimal Virtualization Overhead

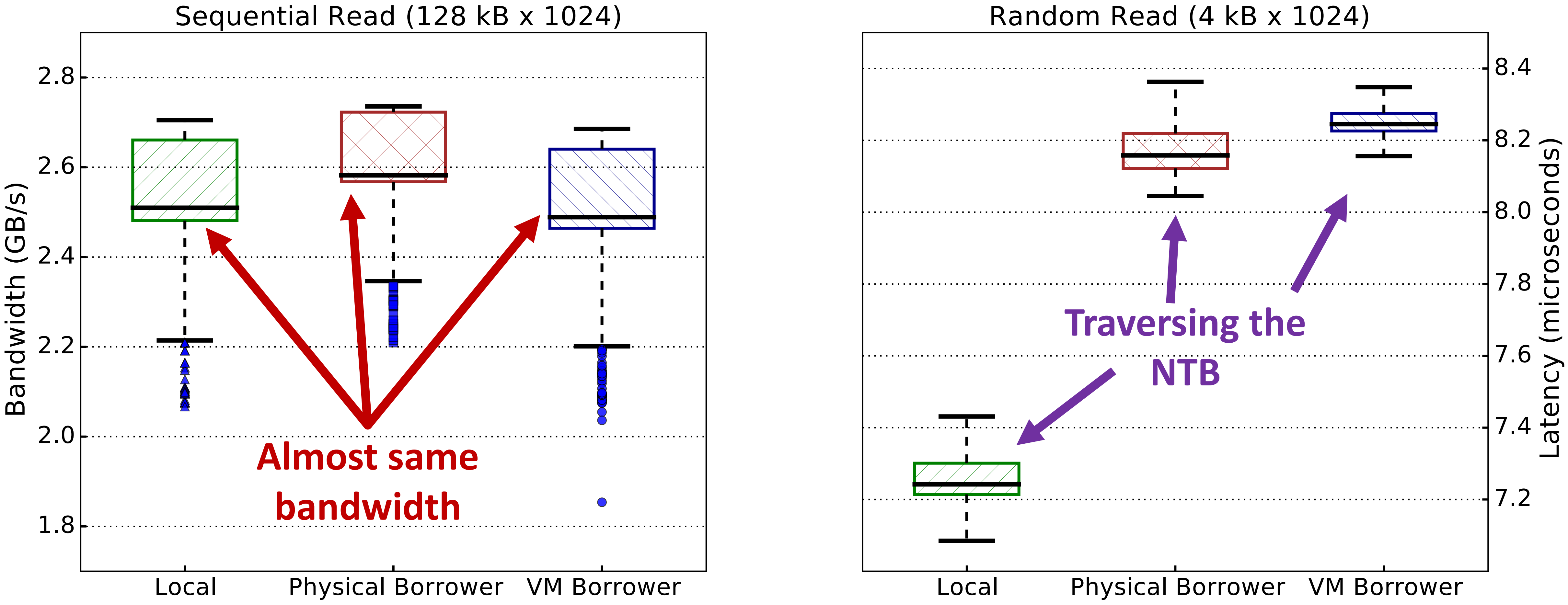
Pass-through of Physical Resources

Dynamic Provisioning & Flexible Composition



Virtual or Paravirtualized Resources

Passing through a remote NVMe disk to a VM only adds the latency of traversing the NTB and is comparable to a physical borrower



Almost same bandwidth

Traversing the NTB

Guest OS: Ubuntu 17.04, Host OS: CentOS 7
VM: Qemu 2.17 using KVM
NVMe Disk: Intel 900P Optane (PCIe x4 Gen3)

Thank you!

Selected publications

“Device Lending in PCI Express Networks”
ACM NOSSDAV 2016

“Efficient Processing of Video in a Multi Auditory Environment using Device Lending of GPUs”
ACM Multimedia Systems 2016 (MMSys’16)

“Flexible Device Sharing in PCIe Clusters using Device Lending”, International Conference on Parallel Processing Companion (ICPP'18 Comp)

haakonks@simula.no

SmartIO & Device Lending demo with GPUs, NVMe and more

Visit Dolphin in the exhibition area (booth 1520)