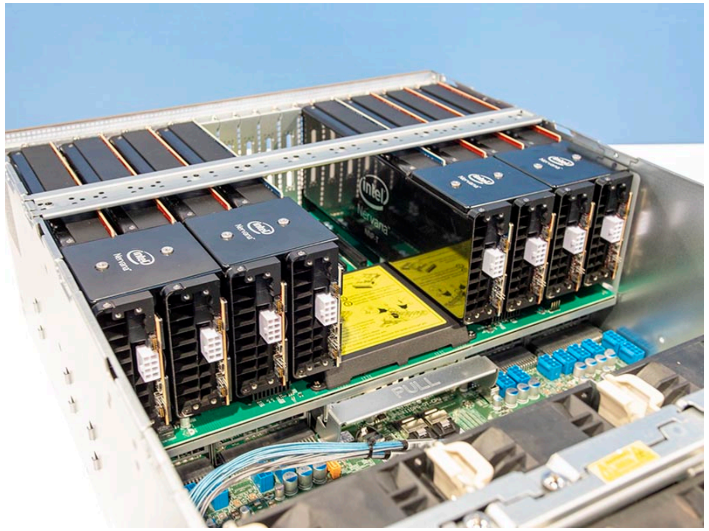
2021 OFA Virtual Workshop

TRUE ZERO MEMORY COPY DIRECT MEMORY TRANSFER FOR CLUSTERS AND HIGH-PERFORMANCE COMPUTING OVER RDMA USING P2PMEM API'S

Suresh Srinivasan

Intel Corporation

CLUSTERS AND HIGH-PERFORMANCE COMPUTING



Server with Training Accelerators on board

Server with GPUs on board

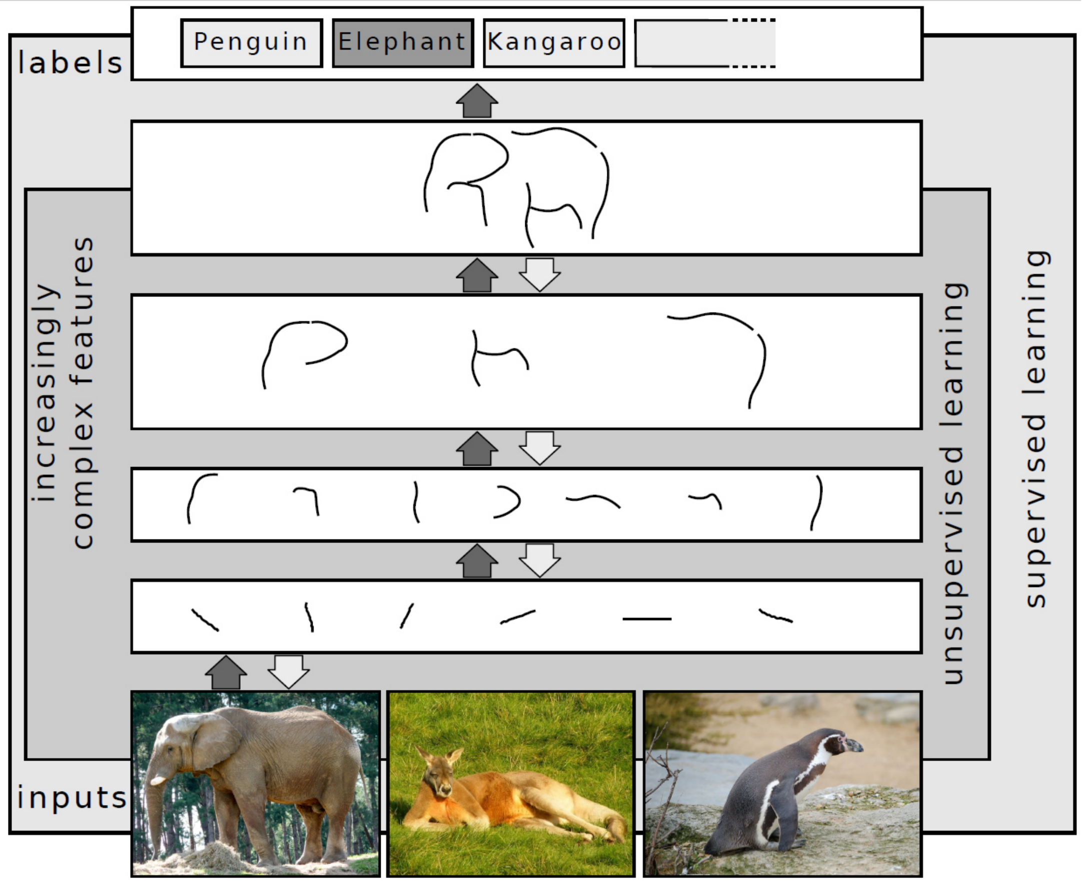
Argonne NATIONAL LABORATORY
U.S. DEPARTMENT OF ENERGY
intel
Hewlett Packard Enterprise
Aurora

Auora System at Argonne National Laboratory (HPC)

- **Cluster/HPC architecture**
  - Node: Servers with PCIe plugins such as Accelerators, Graphical Processing Unit (GPU) and Tensor Processing Unit (TPU) on board.
  - Nodes are connected using high-speed links (fabrics) such as IB, Gigabit Ethernet, etc.,
- **Cluster/HPC workloads:**
  - Split and offload their job across PCIe plugins or across nodes
  - AI workload, HPC workloads etc.,

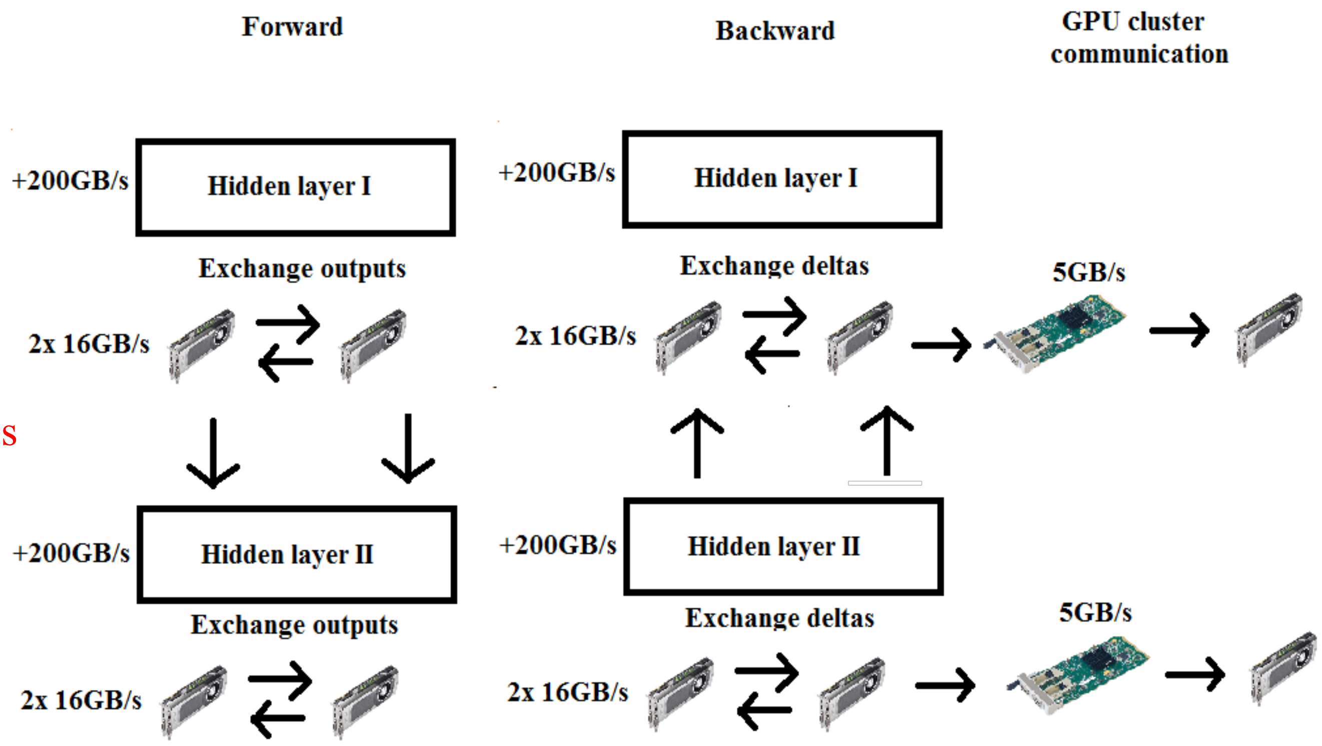
AI WORKLOAD - DEEP LEARNING

- **Progressively extracts higher-level features using multiple layers of the raw input data.**
- **In a typical DL based image processing :**
  - Lower layers may identify the edges
  - Higher layers may identify the concepts relevant to a figures such as digits or letters or faces.
- **DL may be supervised (data with labels) or unsupervised (data only).**
- **DL uses "Deep Neural Network Model" with parameters as weights – learnt through training.**
- **Deep learning training operates on a few samples of data at a time called as mini-batch.**



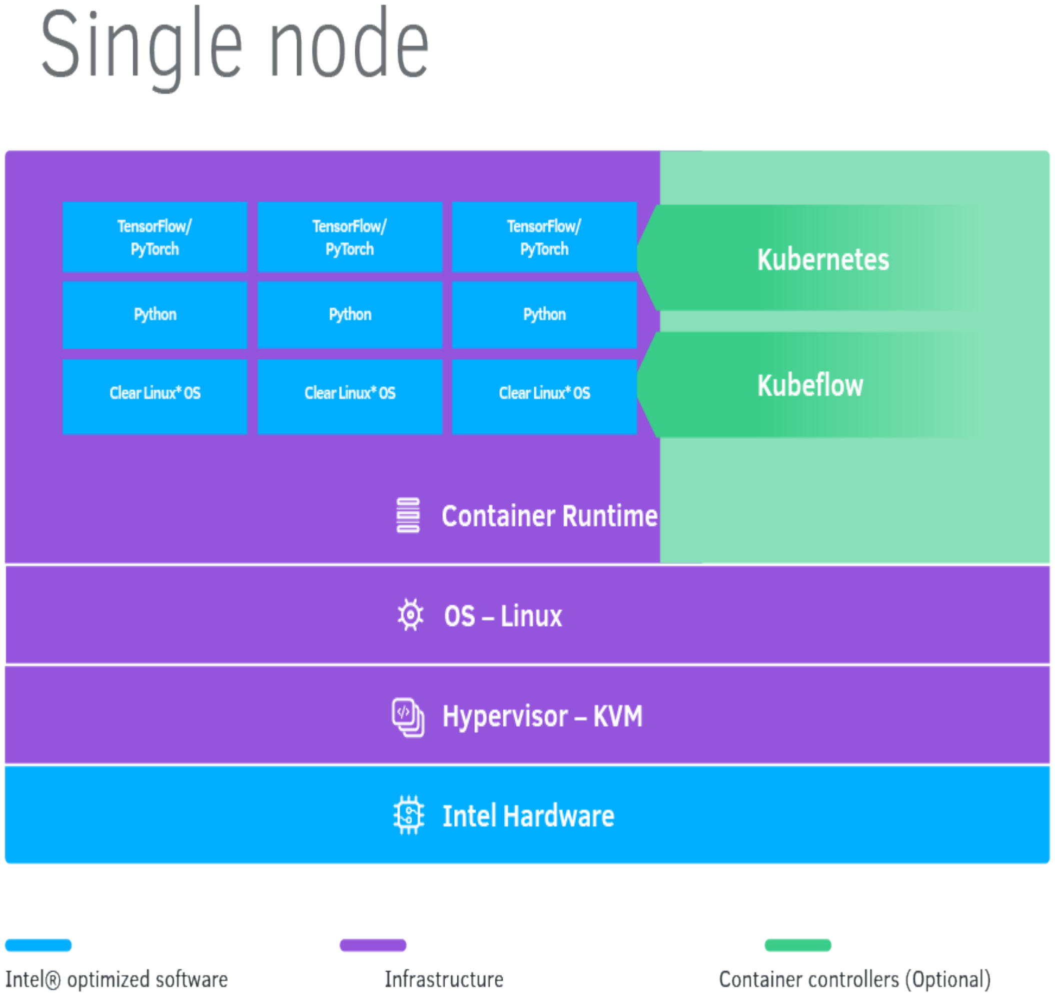
DEEP LEARNING TRAINING (DLT) JOB

- **Minibatch:**
  - Comprises of Forward Pass and Backward Pass
- **Forward Pass:**
  - Performs numerical computations on mini batch
  - Assigns set of scores based on computations for each mini batch.
- **Backward Pass:**
  - Objective function that measures the computed and desired scores
  - The error is populated via Back Pass over the model.
- **Need for Accerlators/GPU/TPU:**
  - Forward Pass and Backward Pass involve billions of floating point operations.
- **Minibatch Iteration:**
  - Millions of minibatch iterations are performed on large scale datasets to achieve high task accuracy.

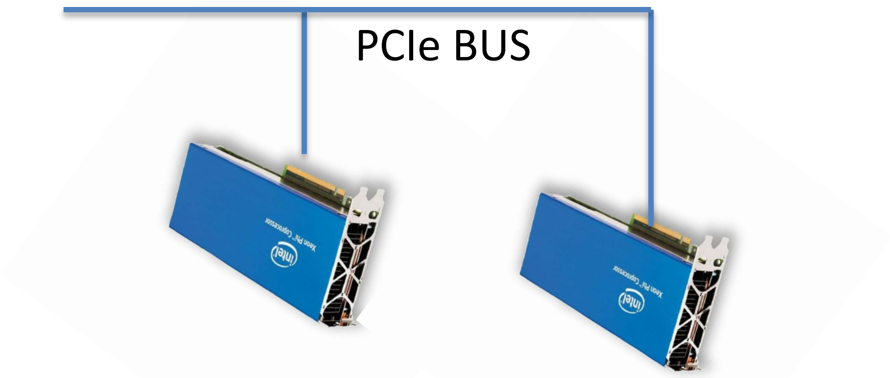


TENSORFLOW AND PYTORCH

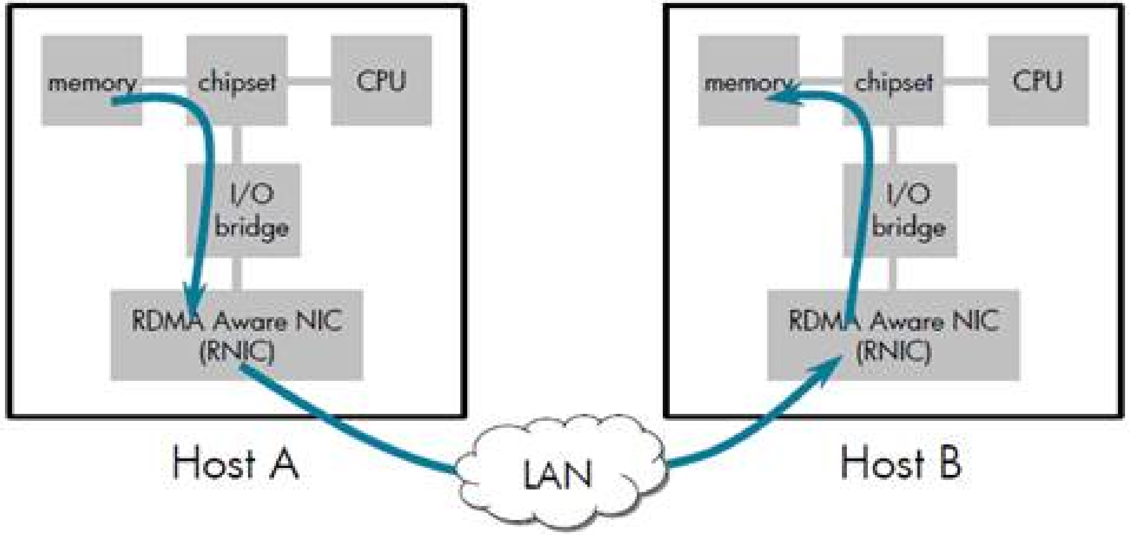
- **Tensorflow**
  - Framework developed by google for research and productions.
  - Open source library for Numerical computation and Large-Scale Machine Learning.
  - Bundles together a study of Machine Learning and Deep learning models.
  - Provides python API(s) and pip packages.
  - Underlying math operations are performed by the high performing C/C++ libraries.
- **PYTorch**
  - Framework developed and actively used by facebook
  - Open Source machine learning library based on torch library.
  - Provides python API(s) and pip packages
  - Bundles numerous deep learning software on top of PyTorch.
- **Kubernates**
  - Open source container-orchestration system for tensor flow application deployment
  - Defines a set of building blocks
  - Scales applications based on resources such as GPU(s)/Accelearators(s)/Nodes.



INTERNODE CLUSTER/HPC AND INTER ACCELERATORS/GPU COMMUNICATION



Node’s GPU to GPU communication



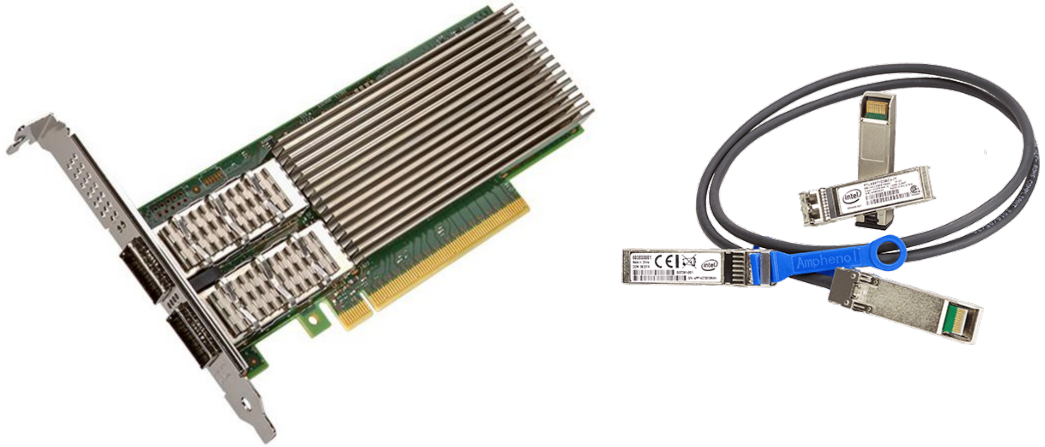
- **Inter Accelerators/GPU communication:**
  - Uses Kernel’s zero memory copy to transfer data.
- **Inter Node communication:**
  - High speed links are used for communication such as InfiniBand or 100G ethernet
  - Remote Direct Memory access (RDMA)
  - RDMA end points – memory read/write access.
  - Kernel’s zero memory copy to transfer data across node’s Accelerator/GPU

Key Bottle neck in communication

CLUSTER/HPC FABRICS AND ACHIEVABLE THROUGHPUT

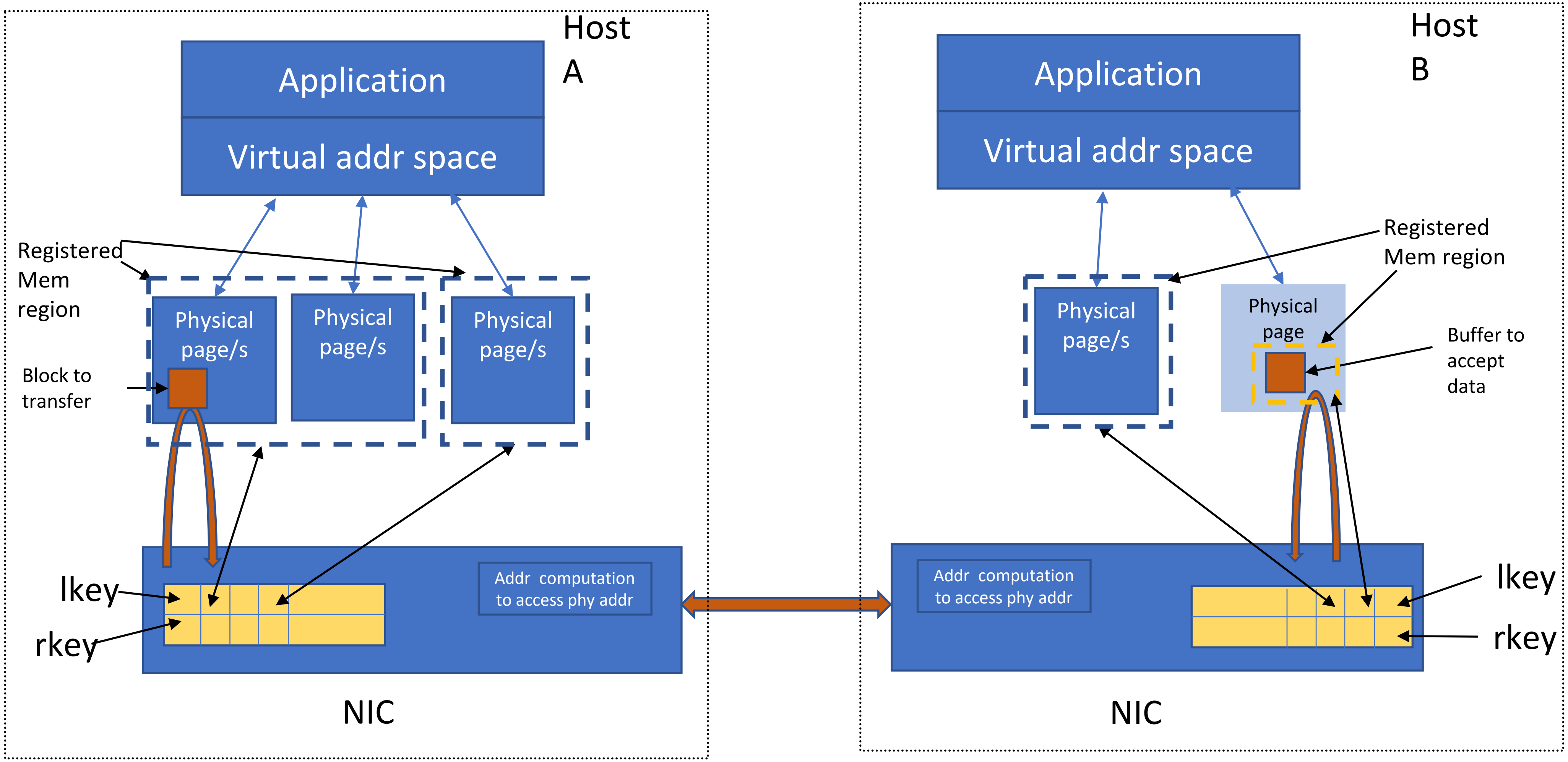
- **Remote Direct Memory Access(RDMA)**
  - Direct memory access from one node's memory to another node's memory without involving OS.
  - Supports zero-copy networking by enabling network adapter to transfer data from the wire to the application memory directly and vice versa.
- **RDMA over Converged Ethernet (RoCE)**
  - Network protocol that allows Remote Direct Memory access (RDMA) over ethernet.
  - 40 Gigabit Ethernet (40GbE) and 100 Gigabit Ethernet (100GbE) achieves effective throughput of 40 Gb/s and 100 Gb/s respectively
- **RDMA over InfiniBand (IB)**
  - Network protocol that allows RDMA over InfiniBand.
  - InfiniBand's SDR, DDR, QDR, FDR, EDR, HDR and NDR achieves effective throughput of 2Gb/s, 4 Gb/s, 8Gb/s, 10 Gb/s, 13 Gb/s, 25 Gb/s, 50 Gb/s and 100 Gb/s respectively.

InfiniBand EDR NIC card and cable

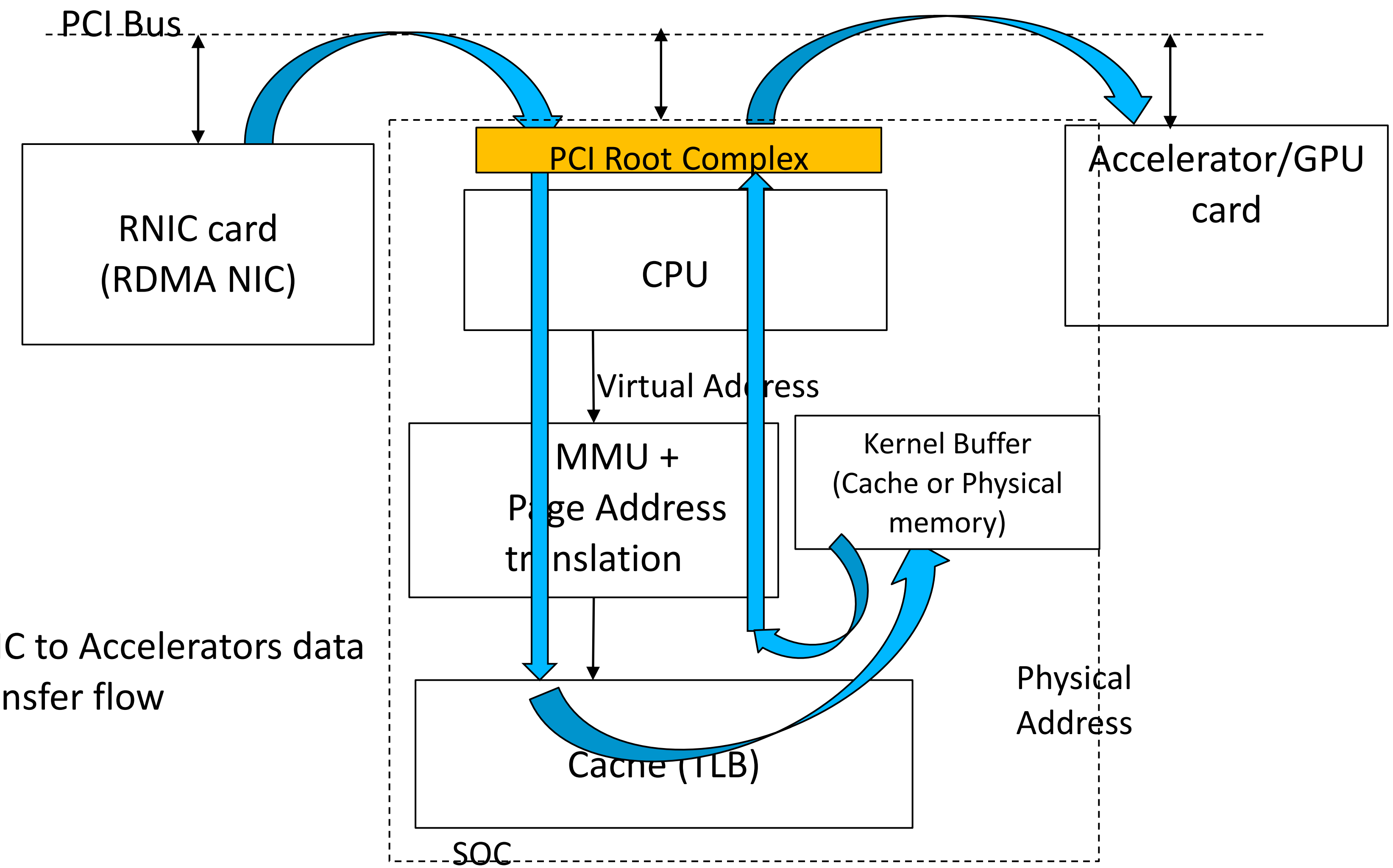


RoCE 100 Gigabit Ethernet and cable

MEMORY AND TRANSLATION IN RDMA



INTERNODE MEM SHARING USING ZERO MEMORY COPY

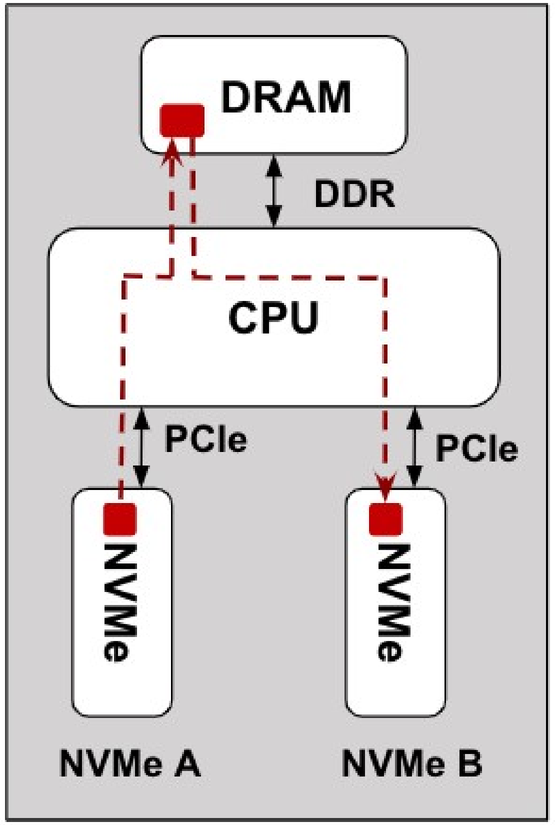


🟦 RNIC to Accelerators data transfer flow

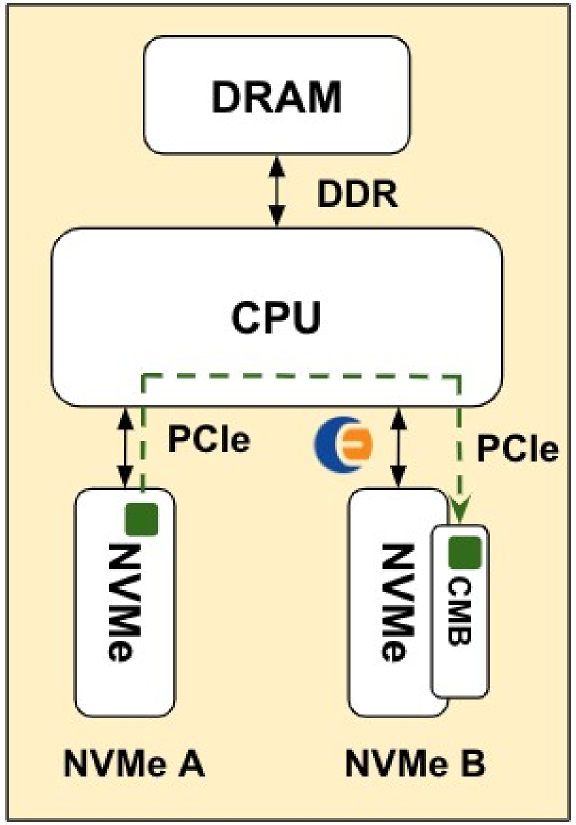
PEER TO PEER MEMORY

- **Framework proposed to bypass extra kernel/user buffer copy for optimization**

- **<u>Architecture</u> :**
  - **Shares physical address (no address translation)**
  - **Publishes desired p2pmem for clients to use**
  - **Bypasses additional kernel buffer copy and Virtual to Physical Address Translator – expected to perform better than zero buffer copy.**
  - **Supports broadcast for buffer sharing**

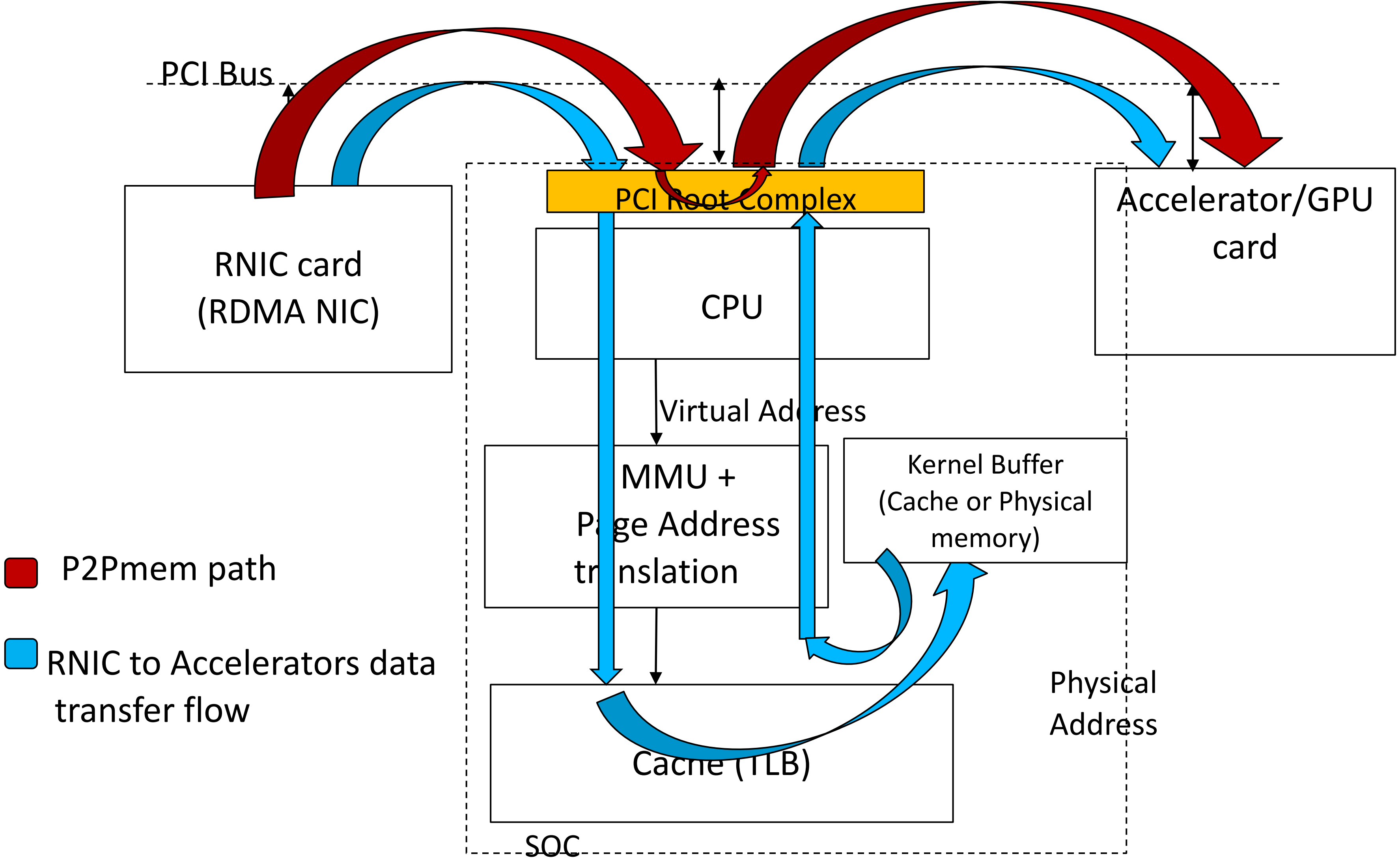


Legacy Datapath



Peer-2-Peer Datapath

KERNEL BUFFER COPY VERSUS P2PMEM FLOW COMPARISON



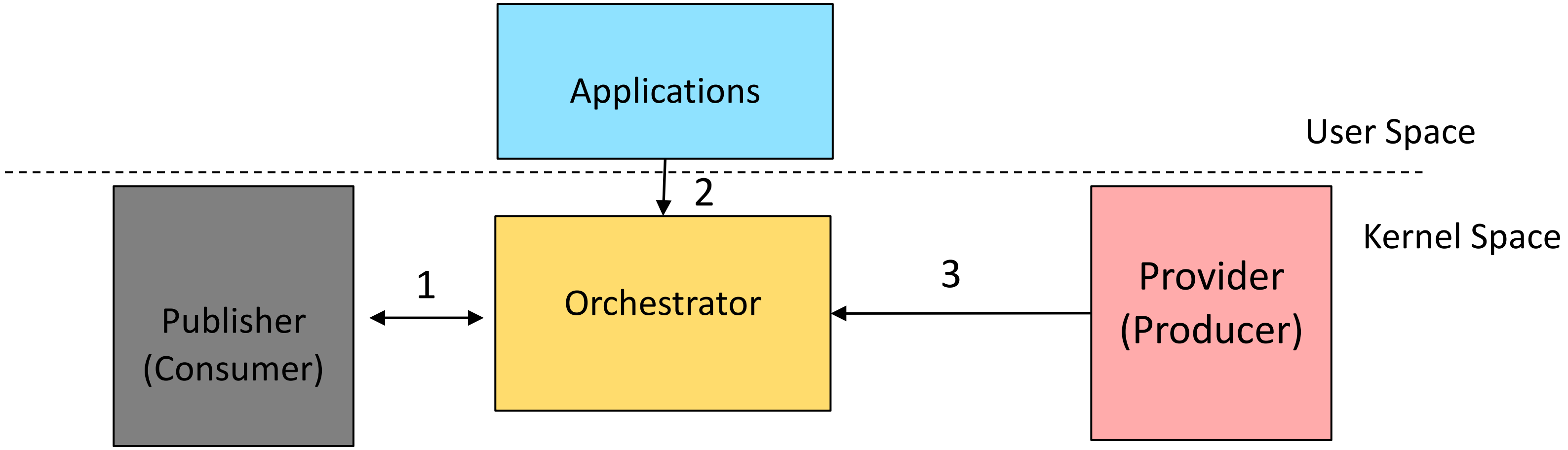
FACTORS INFLUENCING KERNEL BUFFER COPY & P2PMEM PERFORMANCE

- $\text{Latency}_{\text{Kernel-Buf}}$ = PCI-root Complex delay + Virtual Address translation + Kernel Page translation + Kernel Buffer Copy + RDMA to Kernel Page translation + GPU to kernel page translation

- $\text{Latency}_{\text{p2pmem}}$ = PCI-root Complex delay

- **Kernel-Buf incurs additional penalty of cache hit /miss factors, which flushes the cache to reload the cache with appropriate addresses**
- **Common Issue: Multiple PCI cards with multiple DMA access could worsen cache flush incurring delays (To be quantified with practical data)**

P2PMEM CONTROL AND DATA FLOW



- **Publisher exposes their intended memory bar for P2Pmem transfer through P2Pmem-publisher API(s)**
- **Providers consult orchestrator to use the memory.**
- **Orchestrator manages the published memory, and allocates appropriate memory to the providers**
- **User application will use p2pmem allocator from orchestrator (To be developed) to allocate p2pmemory for RDMA transfer or p2pmem transfer.**
- **User Applications can set rules to orchestrator in managing published memory bars.**
- **Orchestrator supports list of devices publishing memory bar and the distances between provider and publisher.**

P2PMEM API(S) SUPPORTED BY UPSTREAM KERNEL

P2Pmem API(s)
pci_alloc_p2pmem(): Allocate P2P DMA memory
pci_p2pdma_add_resource(): Add memory for use as P2P memory
pci_p2pmem_publish(): Publish p2pmem for use by other devices with pci_p2pmem_find()
pci_p2pmem_find_many(): Find a P2P DMA memory device compatible with a specified list of clients with shortest distance
pci_has_p2pmem(): Check if a given PCI has published any p2pmem
pci_p2pmem_alloc_sgl(): Allocate P2P DMA memory in a SGL

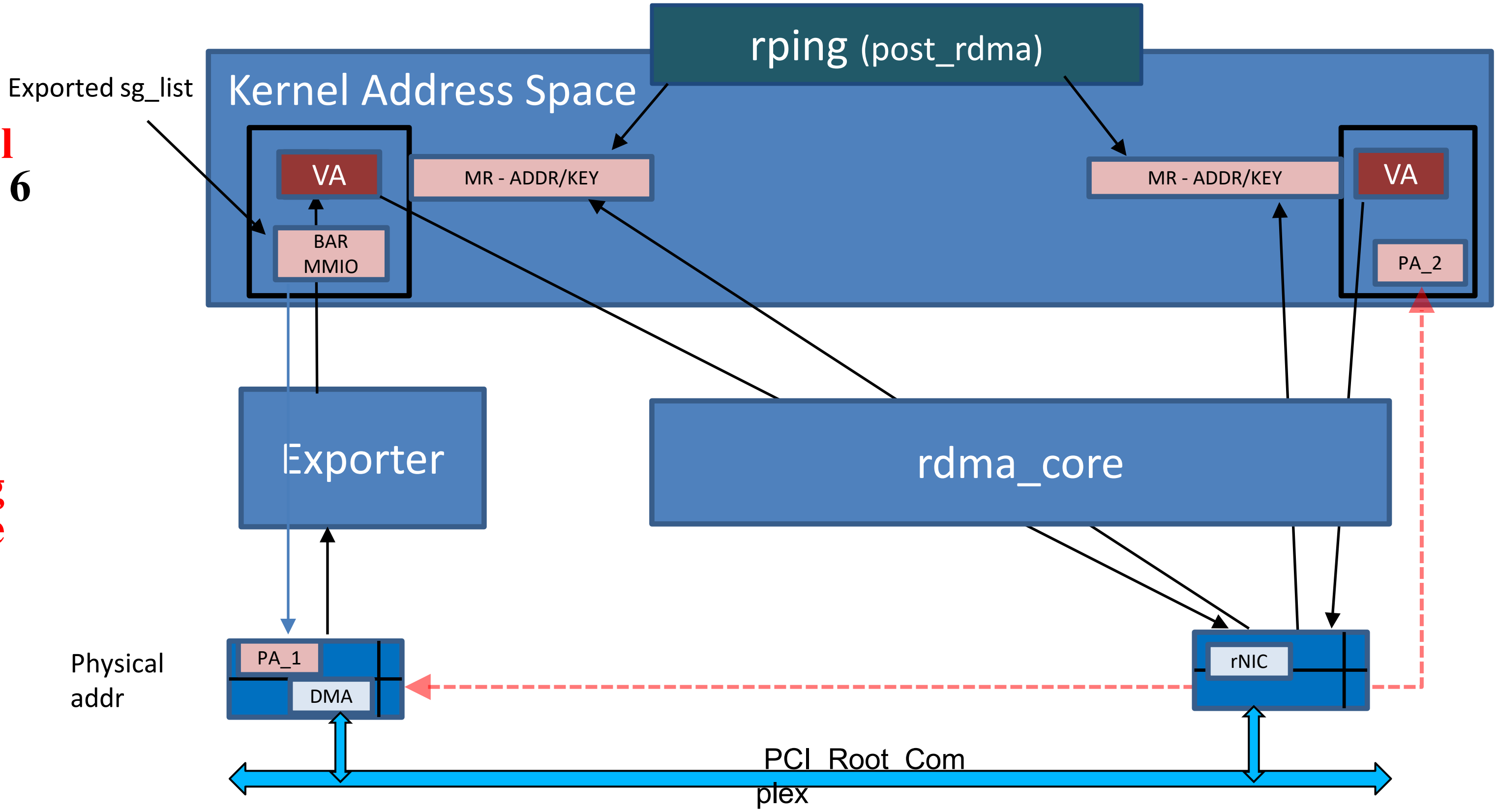
INTEL RDMA (ROCE) SUPPORT FOR RDMA

- Device memory are defined as windows in physical address space BAR (Base Address Register), up to 6 per device

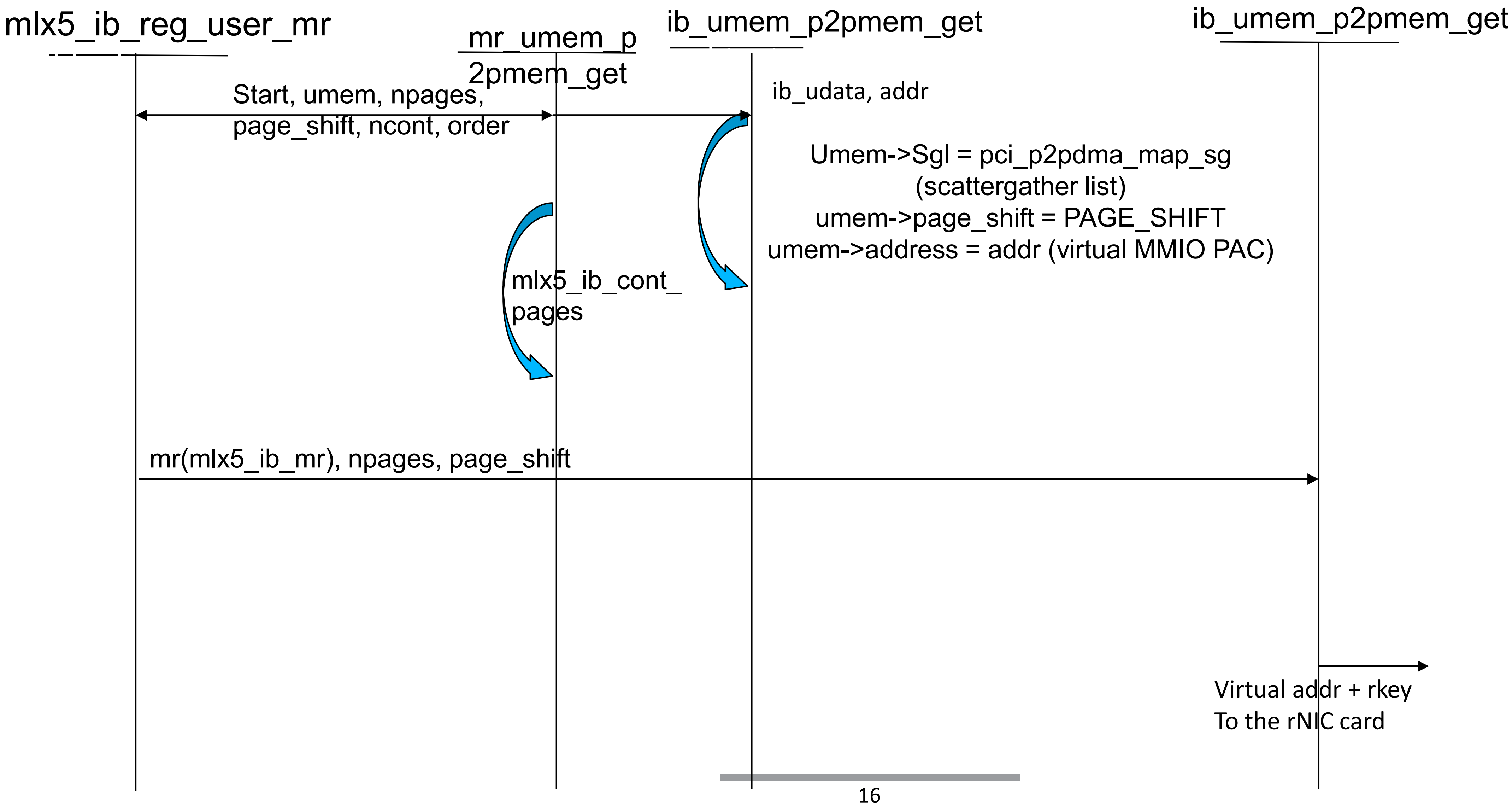
- BAR window is usually mapped to kernel/user address space as MMIO addresses

- DMA uses physical address. Some address sharing mechanism is needed to allow one driver to get the physical address information of the peer buffer

- A few options are available for this purpose peer_mem, pci_p2pdma(p2pmem), dma_buf



ROCE/IB MR REGISTRATION FLOW



mlx5_ib_reg_user_mr

mr_umem_p2pmem_get

ib_umem_p2pmem_get

ib_umem_p2pmem_get

Start, umem, npages, page_shift, ncont, order

ib_udata, addr

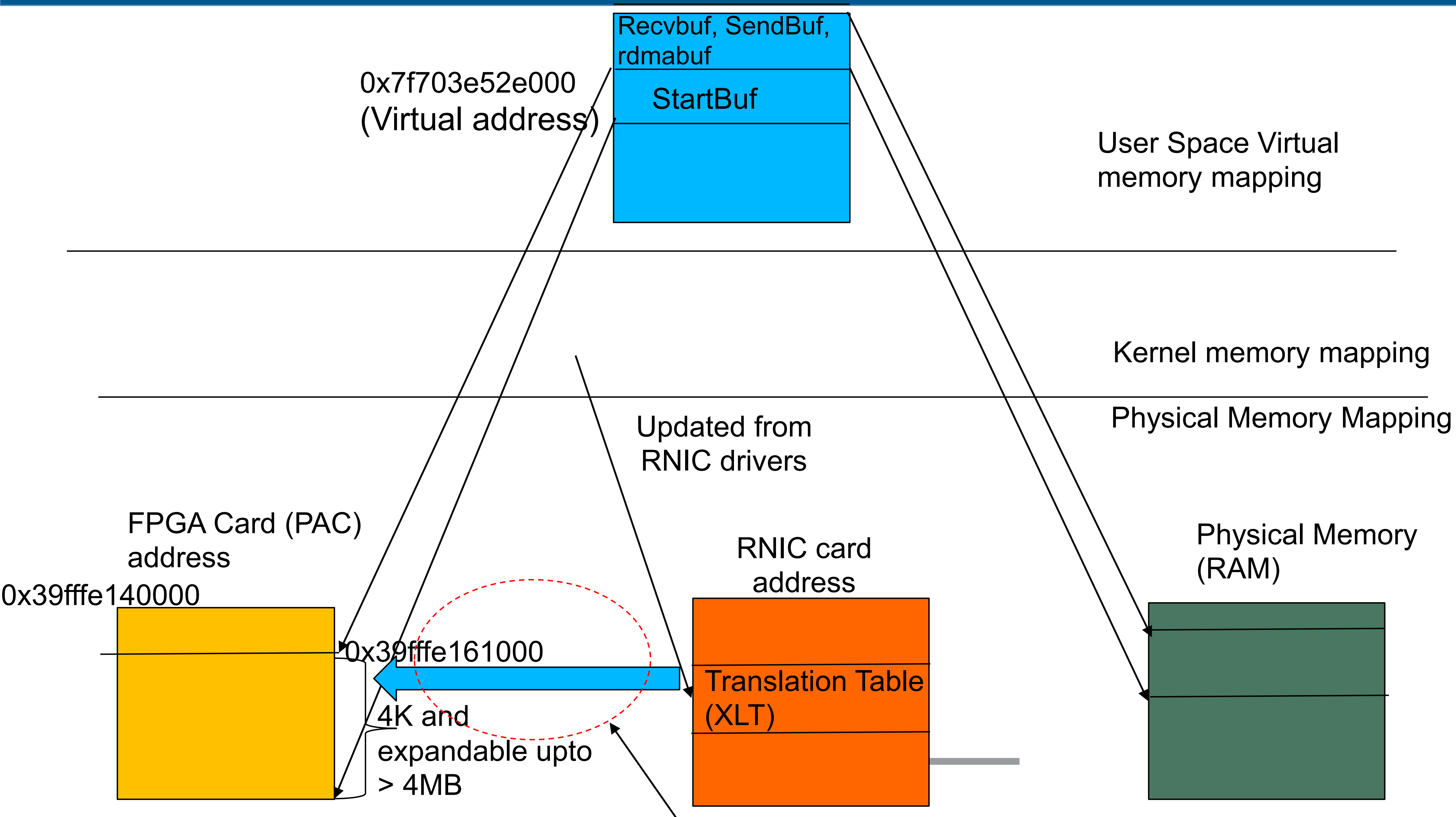
Umem->Sgl = pci_p2pdma_map_sg
(scattergather list)
umem->page_shift = PAGE_SHIFT
umem->address = addr (virtual MMIO PAC)

mlx5_ib_cont_pages

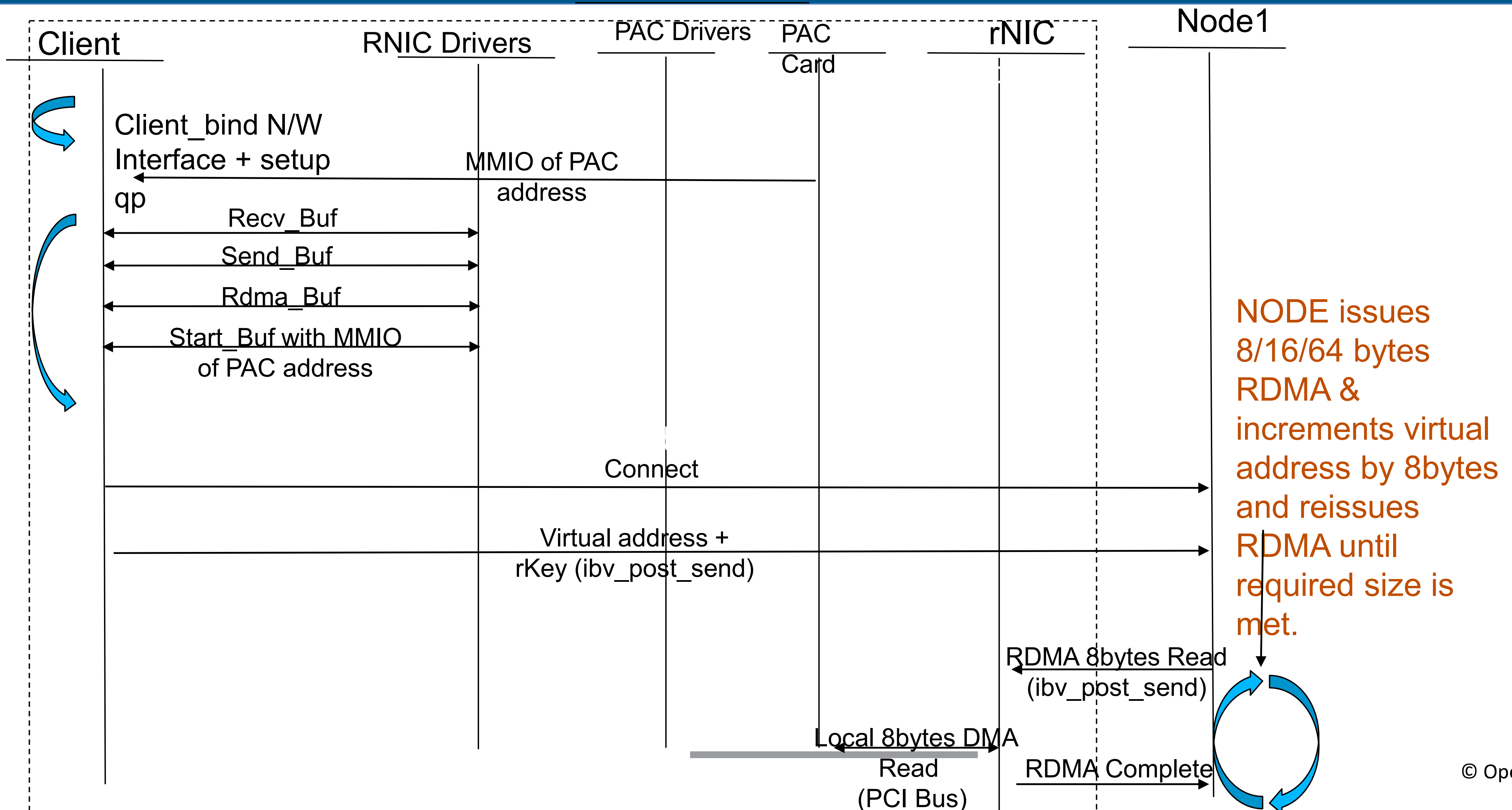
mr(mlx5_ib_mr), npages, page_shift

Virtual addr + rkey
To the rNIC card

ROCE/IB AND PAC SYSTEM MEMORY MAP



END TO END MR REGISTRATION AND P2PMEM TRANSFER



Client

RNIC Drivers

PAC Drivers

PAC Card

rNIC

Node1

Client_bind N/W Interface + setup qp

MMIO of PAC address

Recv_Buf

Send_Buf

Rdma_Buf

Start_Buf with MMIO of PAC address

Connect

Virtual address + rKey (ibv_post_send)

RDMA 8bytes Read (ibv_post_send)

Local 8bytes DMA Read (PCI Bus)

RDMA Complete

NODE issues 8/16/64 bytes RDMA & increments virtual address by 8bytes and reissues RDMA until required size is met.

REFERENCES

- Xiao, Wencong, et al. "Gandiva: Introspective cluster scheduling for deep learning." *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*. 2018.
- Narayanan, Deepak, et al. "Heterogeneity-Aware Cluster Scheduling Policies for Deep Learning Workloads." *14th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 20)*. 2020.
- Biswas, Rajarshi, Xiaoyi Lu, and Dhabaleswar K. Panda. "Accelerating tensorflow with adaptive rdma-based grpc." *2018 IEEE 25th International Conference on High Performance Computing (HiPC)*. IEEE, 2018.
- https://www.tensorflow.org/
- https://en.wikipedia.org/wiki/PyTorch
- https://timdettmers.com/2014/11/09/model-parallelism-deep-learning/
- **P2pmem: Enabling PCI p2pmem in Linux by Stephen Bates**
- https://www.kernel.org/doc/html/latest/driver-api/pci/p2pdma.html
- **https://en.wikipedia.org/wiki/Deep_learning**

OPENFABRICS ALLIANCE

2021 OFA Virtual Workshop

THANK YOU

Suresh Srinivasan

Intel Corporation

intel