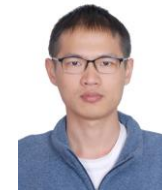




- *FpgaNIC: An FPGA-based Versatile 100Gb*
- *SmartNIC for GPUs*

Zeke Wang, Hongjing Huang, Jie Zhang, Fei Wu, Gustavo Alonso



Outline



1, Why FpgaNIC?

2, What is FpgaNIC?

3, How FpgaNIC Performs?

Why FpgaNIC?



FpgaNIC is an FPGA-based, GPU-centric SmartNIC for GPU-powered distributed applications.

Why FpgaNIC? = {

- 1, Why GPU-centric SmartNIC?**
- 2, Why FPGA-based SmartNIC?**

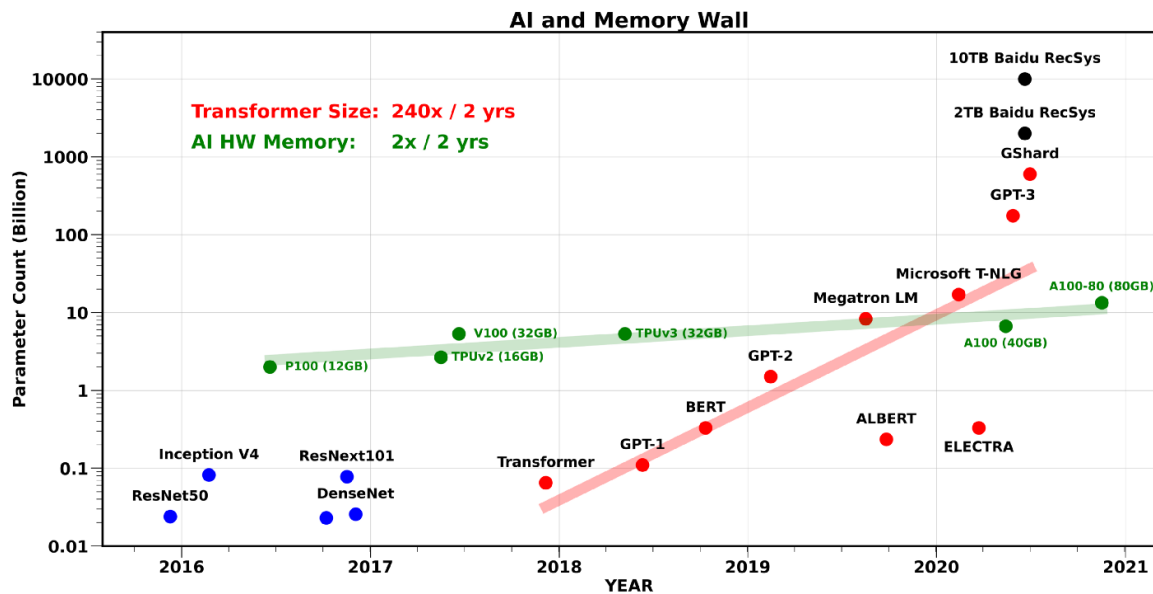
Why GPU-centric SmartNIC?



- **Main Reason:** GPU's **position** does not match its **role** in HPC and AI applications.

1, GPU Consumes Most Network Traffic

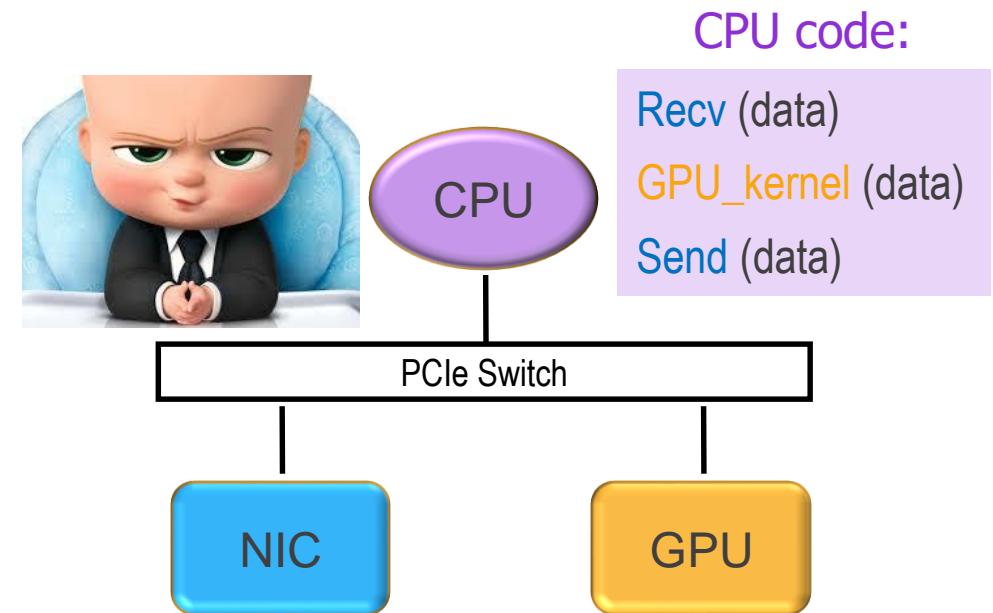
More and More



Left figure credit: Amir Gholami

2, GPU's Position: Worker

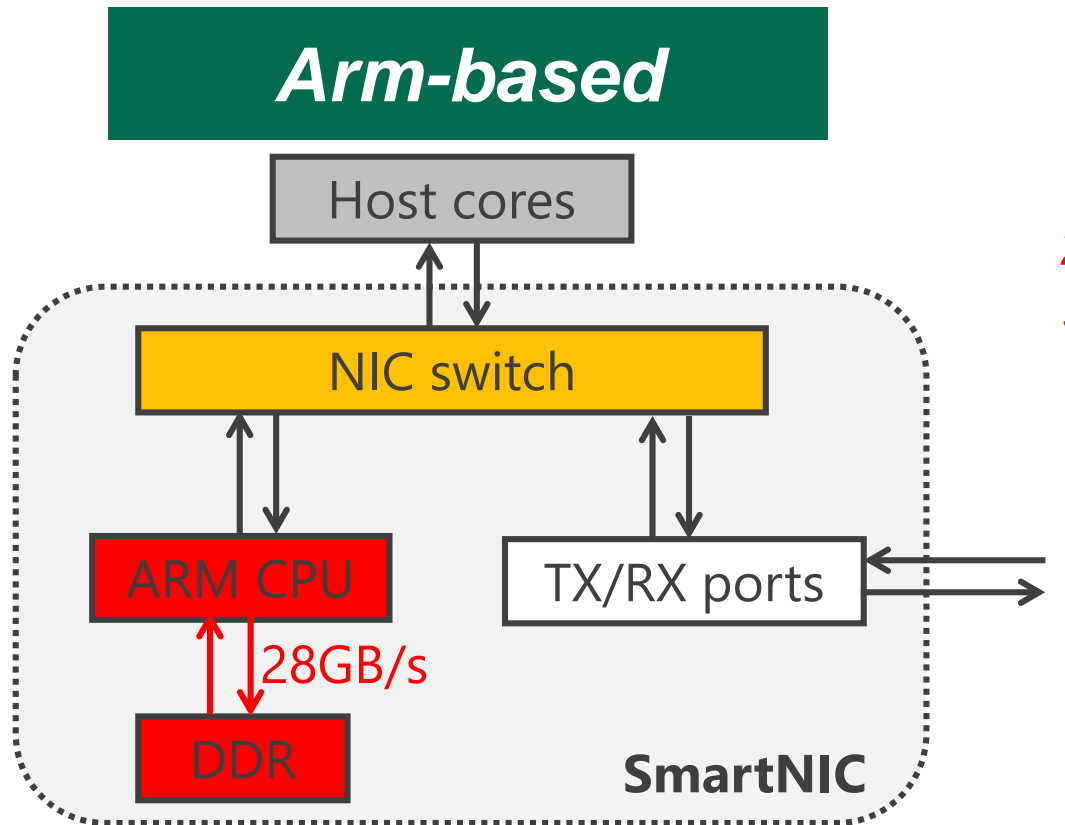
Lack of Flexibility



GPU dreams to be the boss of network data, rather than a worker of the "CPU".

Why FPGA-based SmartNIC?

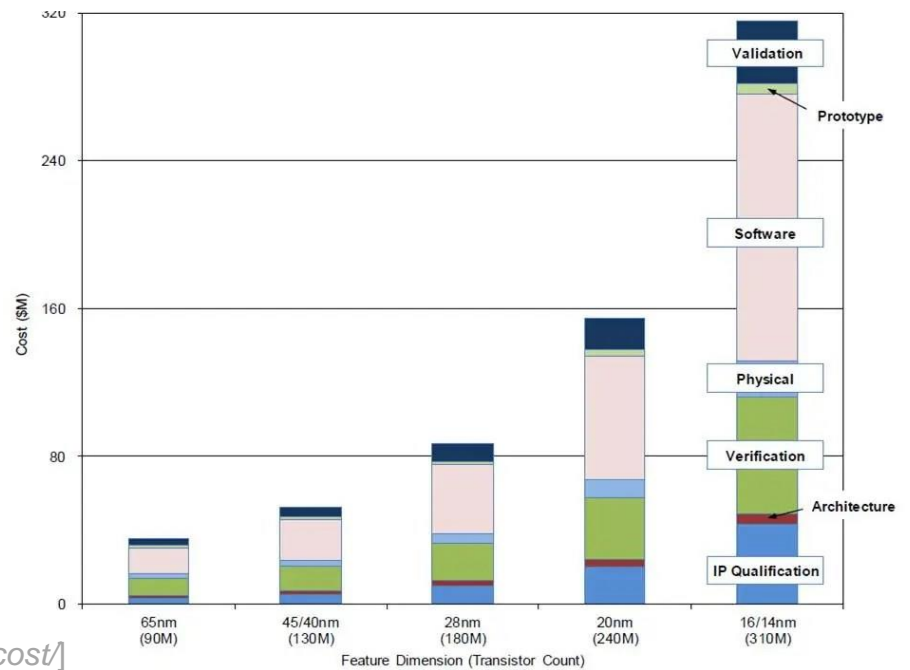
- ● ● ● ●
- **Main Reason:** ARM-based and ASIC-based SmartNICs cannot always meet two goals (**programmability** and **performance**) concurrently.



Staging 100G network data (2*100Gb≈25GB/s) already overwhelms the ARM CPU.

ASIC-based

- 1, No general architecture for various applications
- 2, Long development cycle for each typeout
- 3, Higher and higher typeout cost



[Sperling, <https://semiengineering.com/how-much-will-that-chip-cost/>]

Outline



1, Why FpgaNIC?

2, What is FpgaNIC?

3, How FpgaNIC Performs?

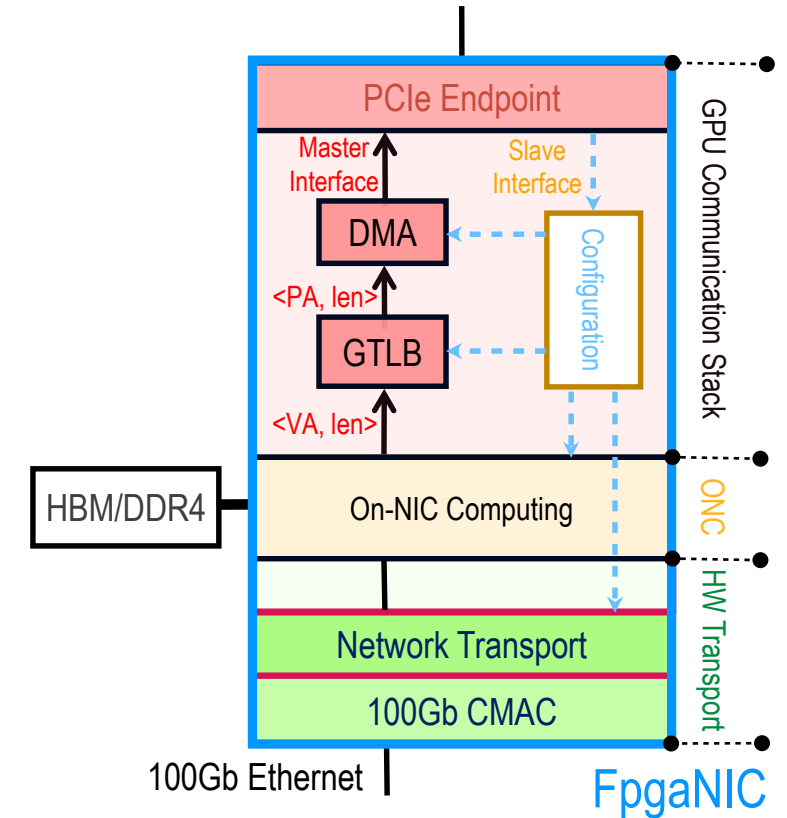
What is FpgaNIC?



■ Two Goals of FpgaNIC:

- GPU-centric SmartNIC
- **Versatility**: Flexible Design Space
Exploration around SmartNIC

■ FpgaNIC (GPU-centric SmartNIC):



FpgaNIC: GPU Communication Stack

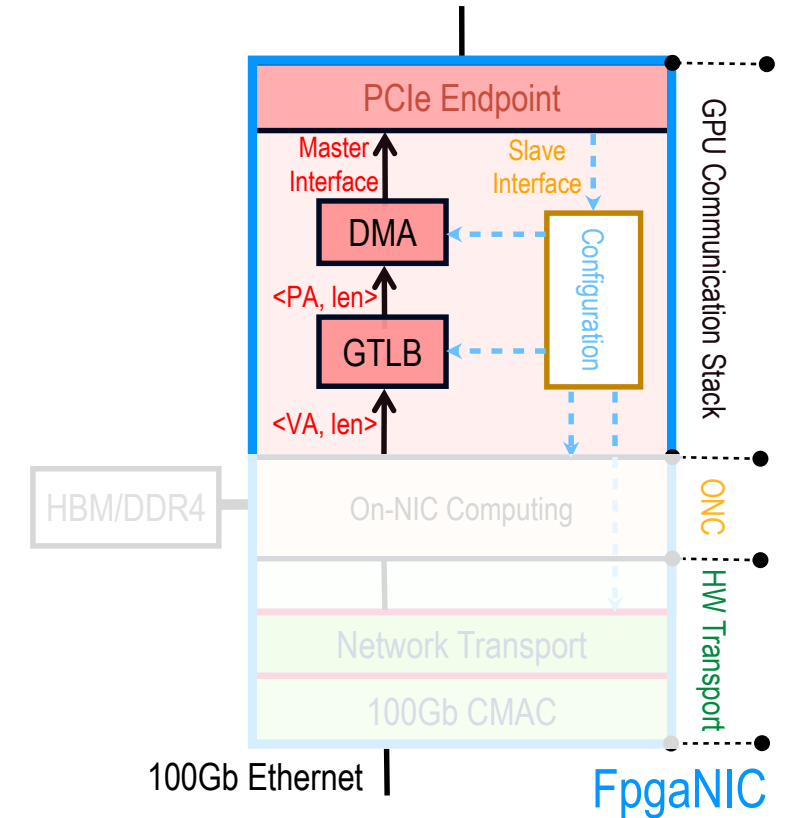


■ Two Goals of FpgaNIC:

- GPU-centric SmartNIC
- **Versatility**: Flexible Design Space
Exploration around SmartNIC

■ FpgaNIC (GPU-centric SmartNIC):

- GPU communication stack:
 - Enabling data/control plane offloading

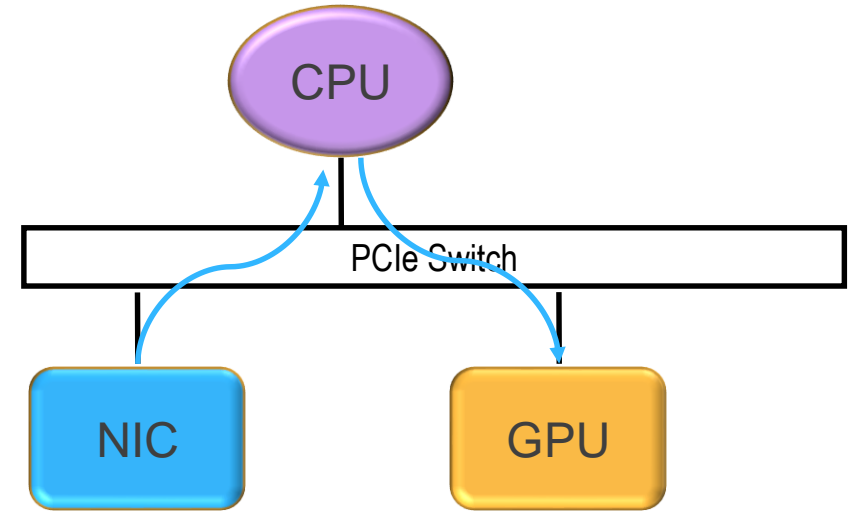


Data Plane Offloading vs. Without Offloading



- **Without Data Plane Offloading:**

- NIC → CPU, CPU → GPU

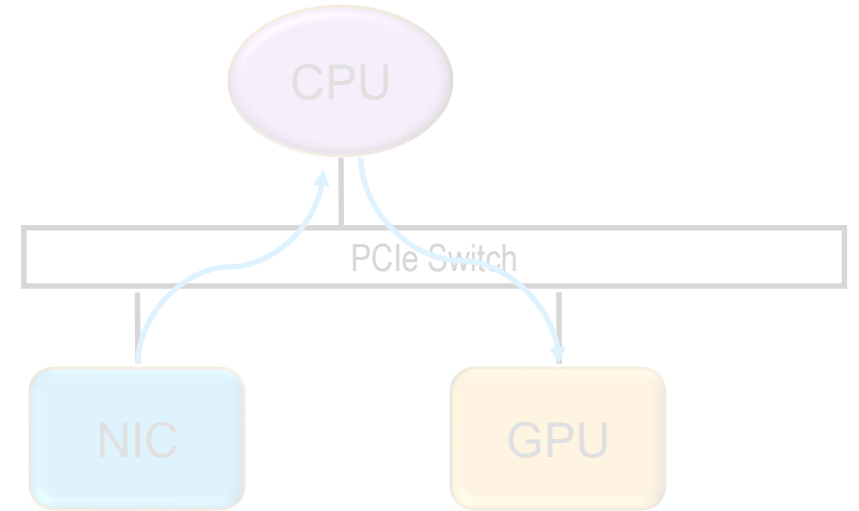


Data Plane Offloading vs. Without Offloading



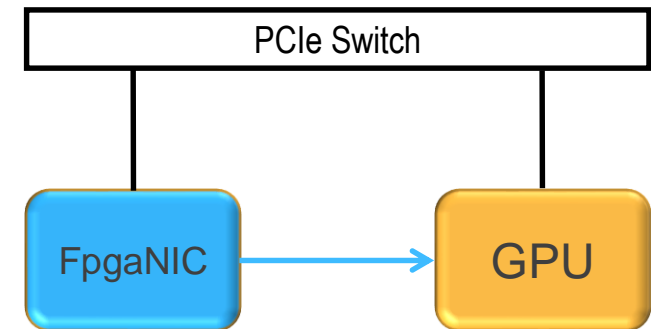
■ Without Data Plane Offloading:

- NIC → CPU, CPU → GPU



■ With Data Plane Offloading:

- NIC → GPU
- FpgaNIC uses GPU's virtual address

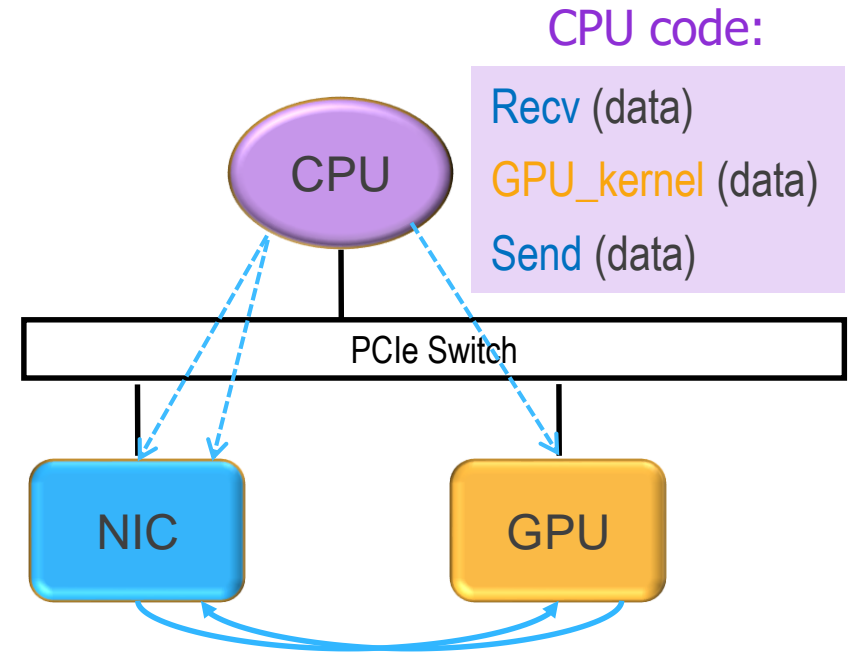


Control Plane Offloading vs. Without Offloading



Scenario: Process network data on GPU

- **Without Control Plane Offloading:**
 - ❑ Three control commands from CPU via PCIe
 - ❑ NIC and GPU operations are serialized



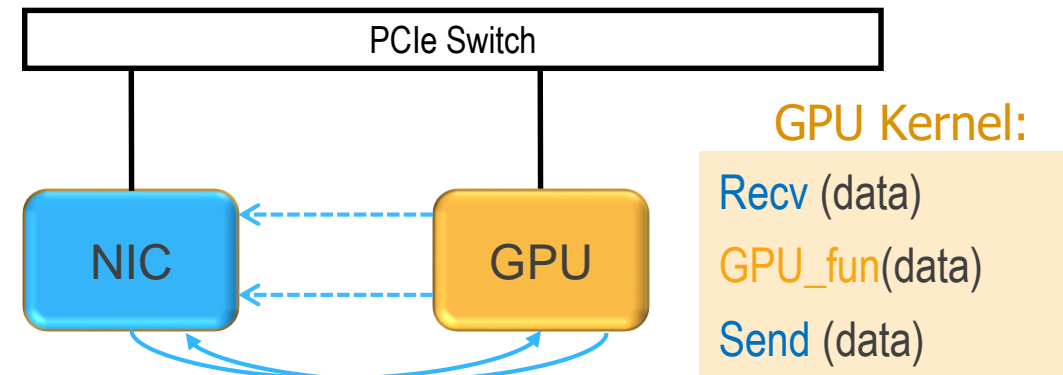
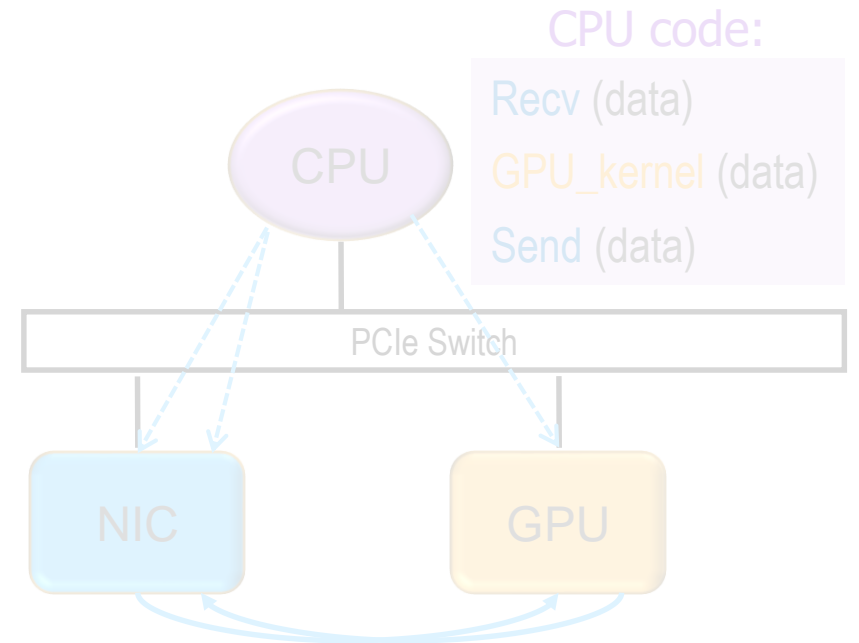
Control Plane Offloading vs. Without Offloading



Scenario: Process network data on GPU:

- **Without Control Plane Offloading:**
 - ❑ Three control commands from CPU via PCIe
 - ❑ NIC and GPU operations are serialized

- **With Control Plane Offloading:**
 - ❑ Two control commands from GPU via PCIe
 - ❑ NIC and GPU operations can be parallelized



FpgaNIC: Hardware Network Transport

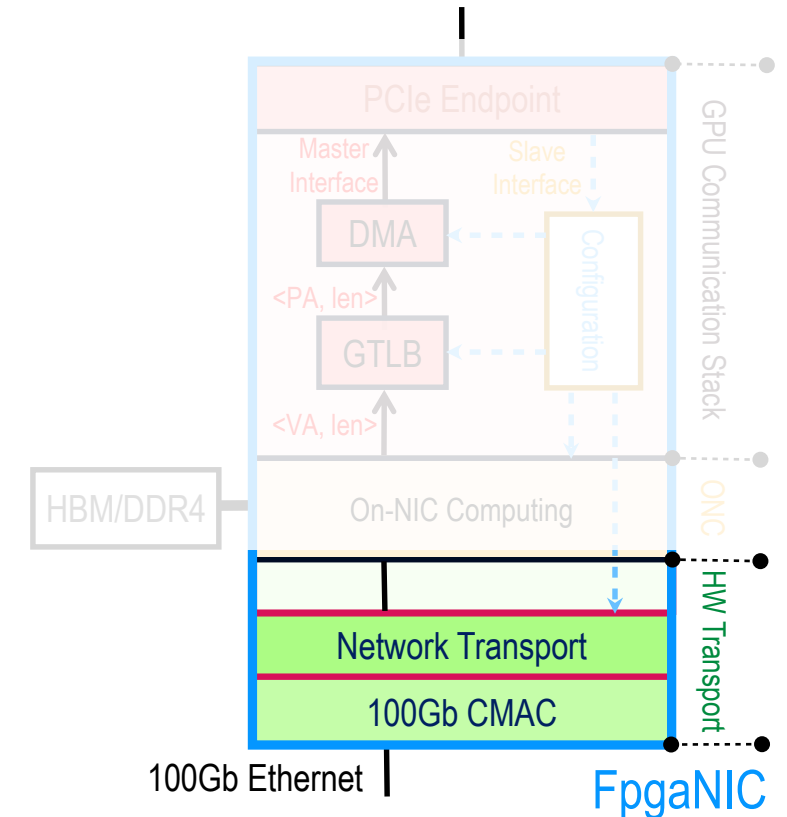


■ Two Goals of FpgaNIC:

- GPU-centric SmartNIC
- **Versatility**: Flexible Design Space
Exploration around SmartNIC

■ FpgaNIC (**GPU-centric SmartNIC**):

- GPU communication stack
 - Enabling data/control plane offloading
- **Hardware network transport**
 - Enabling fast lossless network processing



FpgaNIC: On-NIC Computing

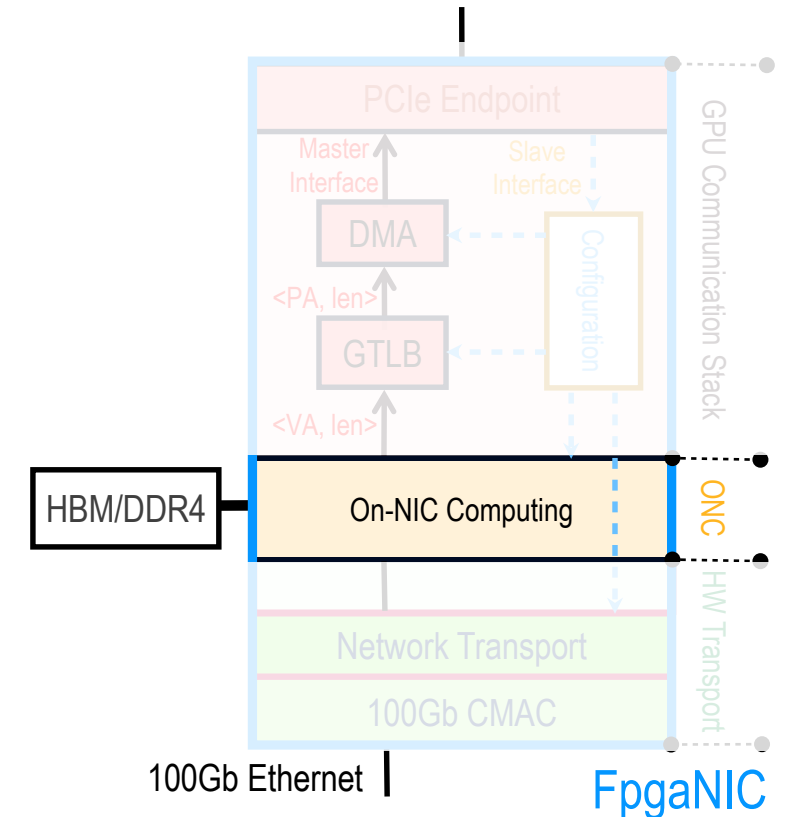


■ Two Goals of FpgaNIC:

- GPU-centric SmartNIC
- **Versatility**: Flexible Design Space
Exploration around SmartNIC

■ FpgaNIC (**GPU-centric SmartNIC**):

- GPU communication stack
 - Enabling data/control plane offloading
- **Hardware network transport**
 - Enabling fast lossless network processing
- **On-NIC computing**
 - Enabling flexible 100Gb data-path accelerator

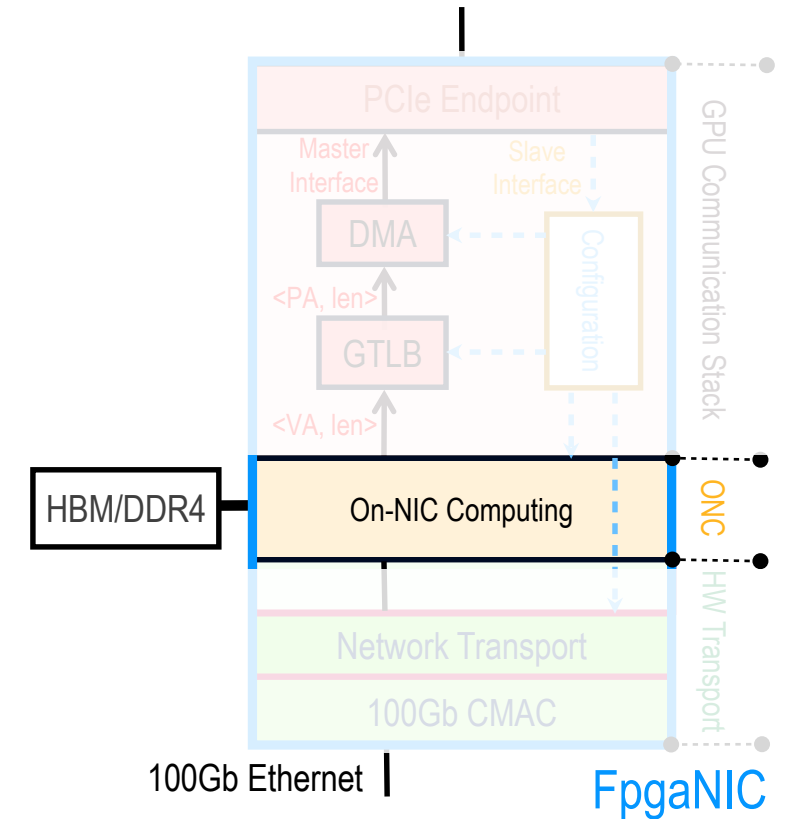


FpgaNIC: On-NIC Computing



■ On-NIC Computing (ONC):

- **Goal:** Enabling versatility (flexible 100Gb data-path accelerator)
- **Supporting Three SmartNIC Models:**
 - **Direct:** GPU-centric networking
 - Idea: GPU communication stack to HW transport.
 - **Off-path:** AllReduce
 - Idea: ONC manipulates GPU communication stack and HW transport
 - **On-path:** HyperLogLog
 - Idea: ONC as a bump between GPU communication stack and HW transport



Where is FpgaNIC?

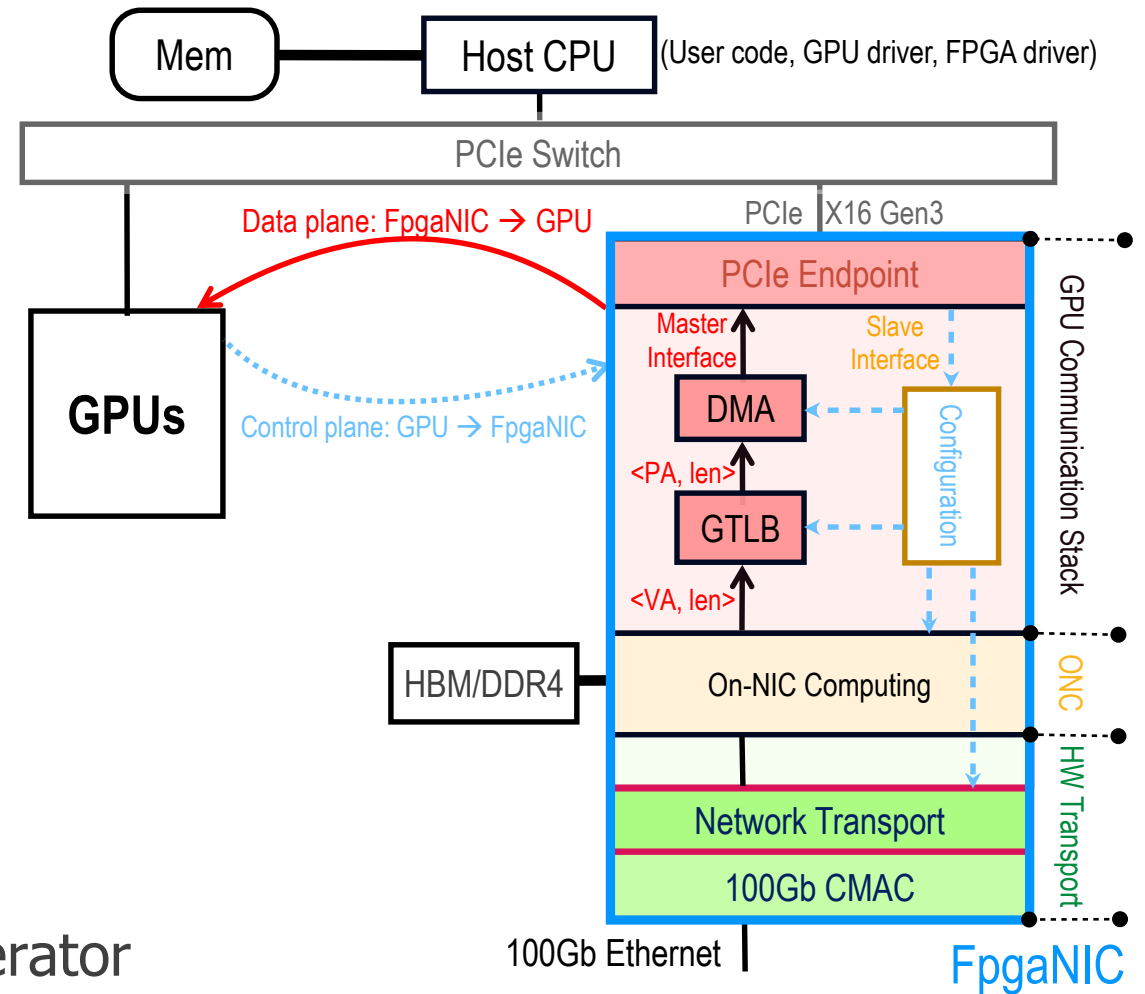


Two Goals of FpgaNIC:

- GPU-centric SmartNIC
- Versatility: Flexible Design Space Exploration around SmartNIC

FpgaNIC (GPU-centric SmartNIC):

- GPU communication stack
 - Enabling data/control plane offloading
- On-NIC computing
 - Enabling flexible 100Gb data-path accelerator
- Hardware network transport
 - Enabling fast network traffic processing



Outline

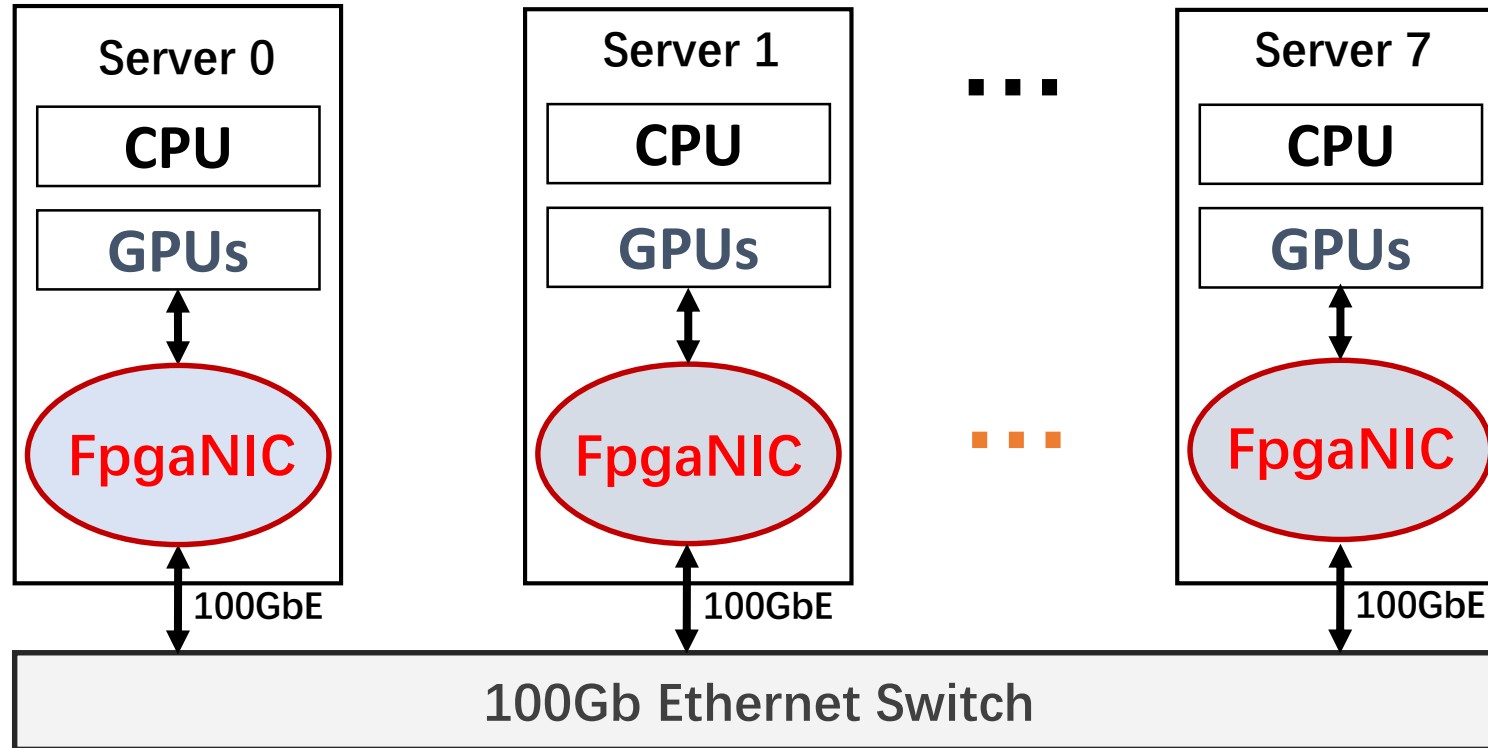


1, Why FpgaNIC?

2, What is FpgaNIC?

3, How FpgaNIC Performs?

Experimental Setup



- **Each Server in a Platform:**
 - **FpgaNIC:** Xilinx U280/U50 FPGA
 - **GPU:** Nvidia A100/RTX8000

Goal of Experiment



- **1, FpgaNIC's Functionality**
 - Latency between FpgaNIC and GPU
 - Throughput between FpgaNIC and GPU

- **2, FpgaNIC's Versatility**
 - Direct Model
 - Off-path Model
 - On-path Model

Goal of Experiment



■ **1, FpgaNIC's Functionality**

- Latency between FpgaNIC and GPU
- Throughput between FpgaNIC and GPU

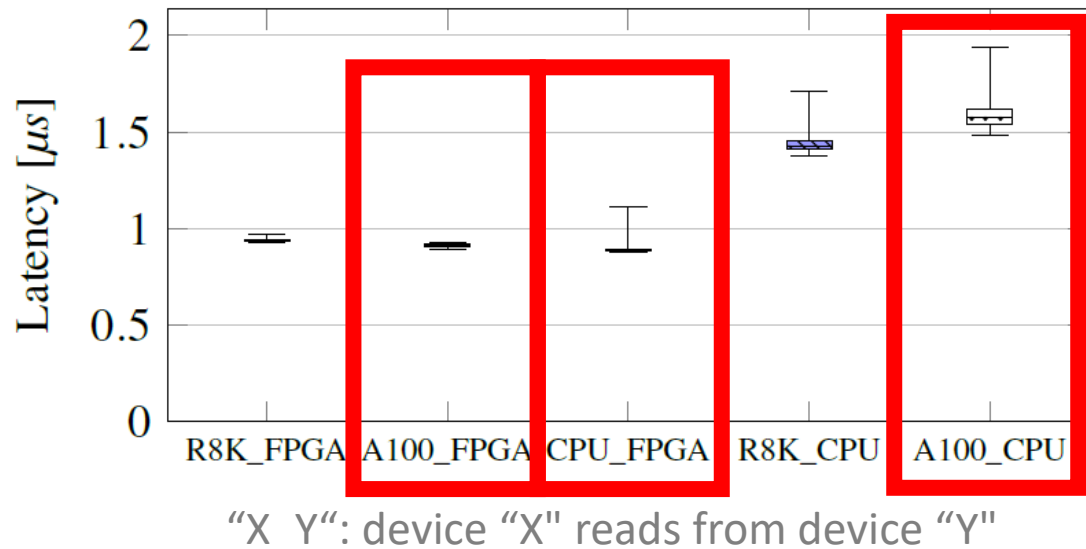
■ **2, FpgaNIC's Versatility**

- Direct Model
- Off-path Model
- On-path Model

FpgaNIC: Functionality

• • • • •

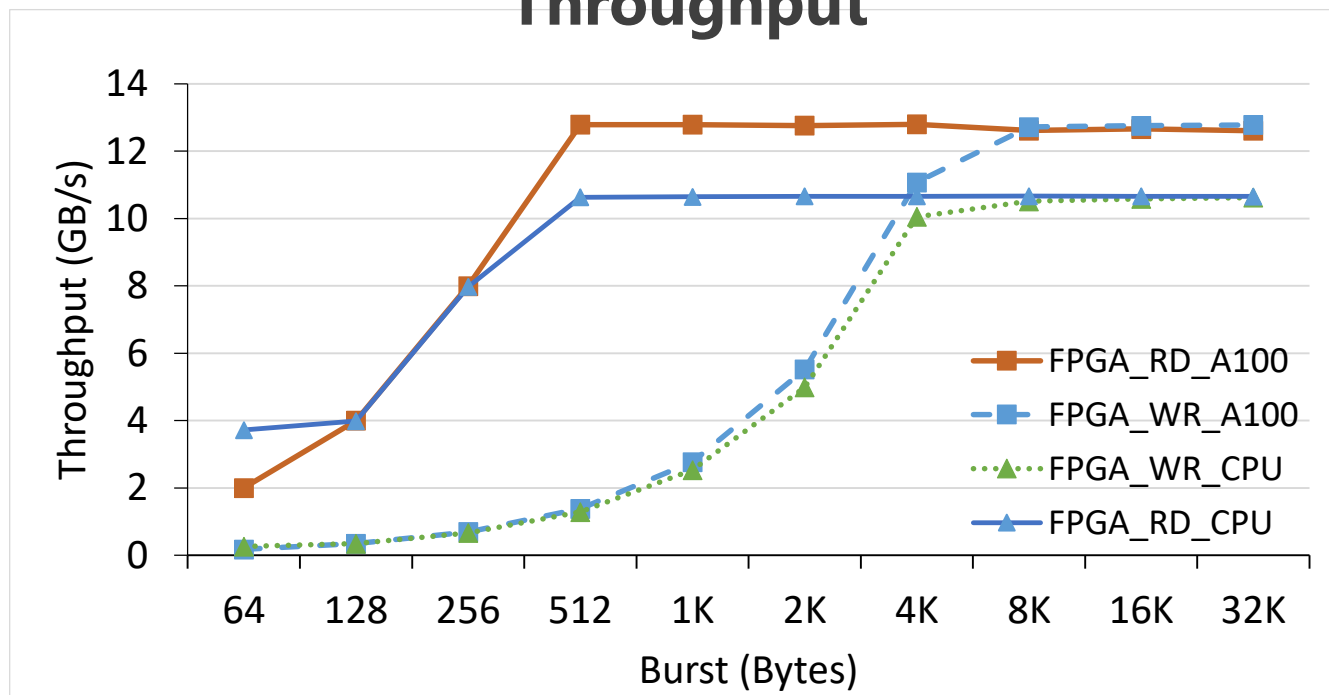
Latency:



Observation:

- 1, GPU-FPGA has stable latency.
- 2, "GPU_FPGA" < "GPU_CPU" + "CPU_FPGA".

Throughput



Observation:

FpgaNIC achieves similar throughput when accessing GPU memory and CPU memory.

Goal of Experiment



■ **1, FpgaNIC's Functionality**

- Latency between FpgaNIC and GPU
- Throughput between FpgaNIC and GPU

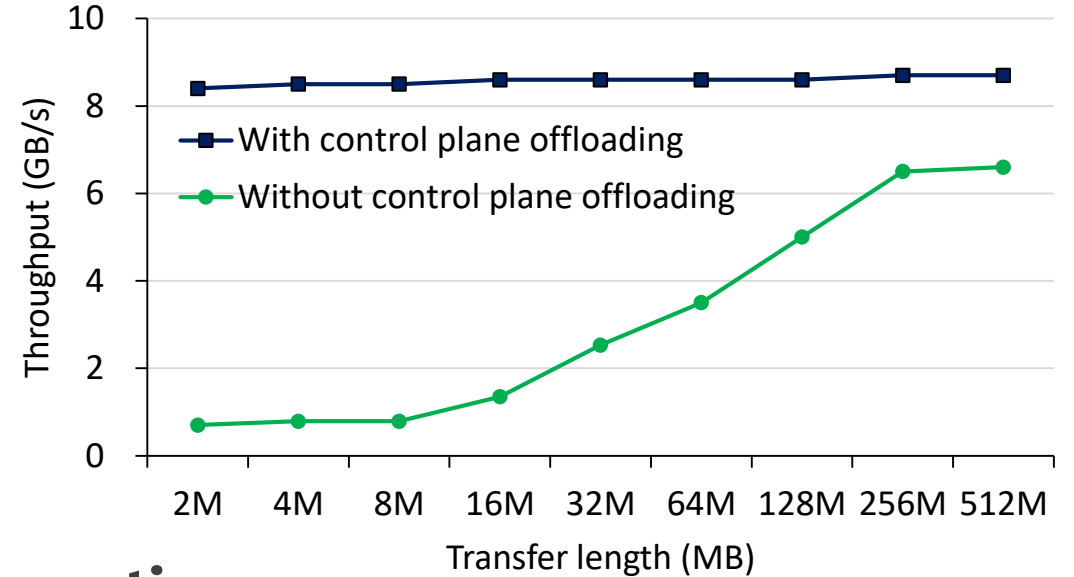
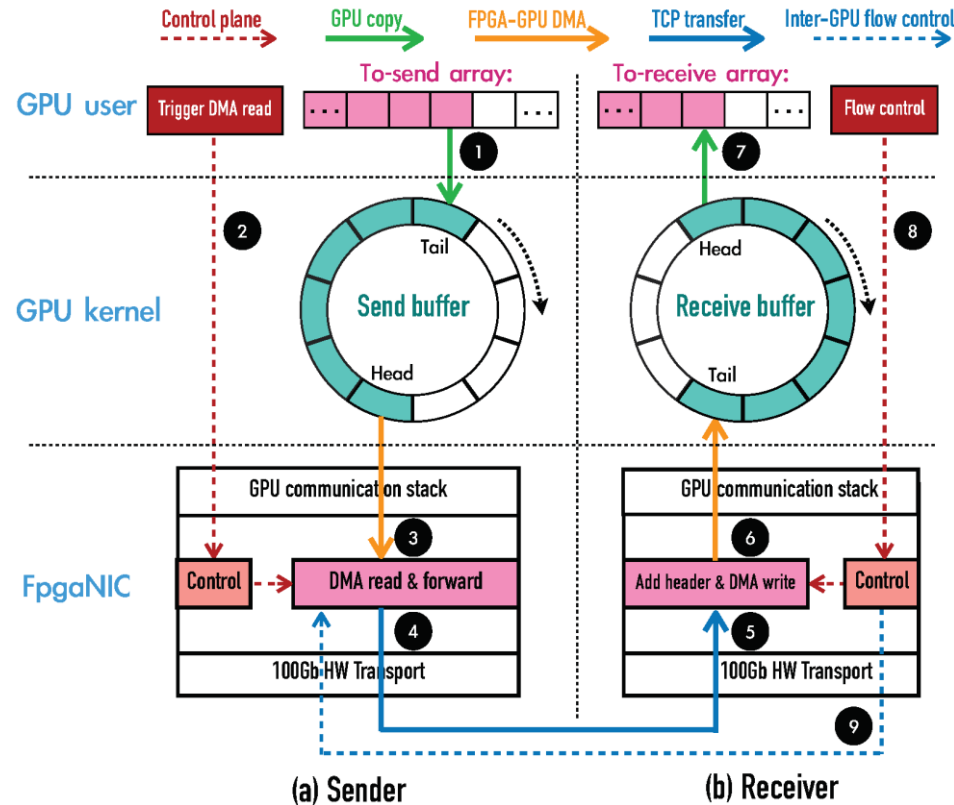
■ **2, FpgaNIC's Versatility**

- Direct Model
- Off-path Model
- On-path Model

FpgaNIC: GPU-centric Networking (Direct Model)



Architecture of GPU-centric Networking



Observations:

- 1, Control plane offloading significantly increases throughput no matter the transfer length is.
- 2, When the transfer length is smaller, control plane offloading gets higher speedup.

- **WO control plane offloading** (baseline): CPU manipulates NIC, hard to parallel NIC and GPU operations
- **With control plane offloading:** GPU directly manipulates NIC, allowing fine-grained coprocessing

FpgaNIC: AllReduce (Off-path Model)



■ **NCCL** [1]: baseline

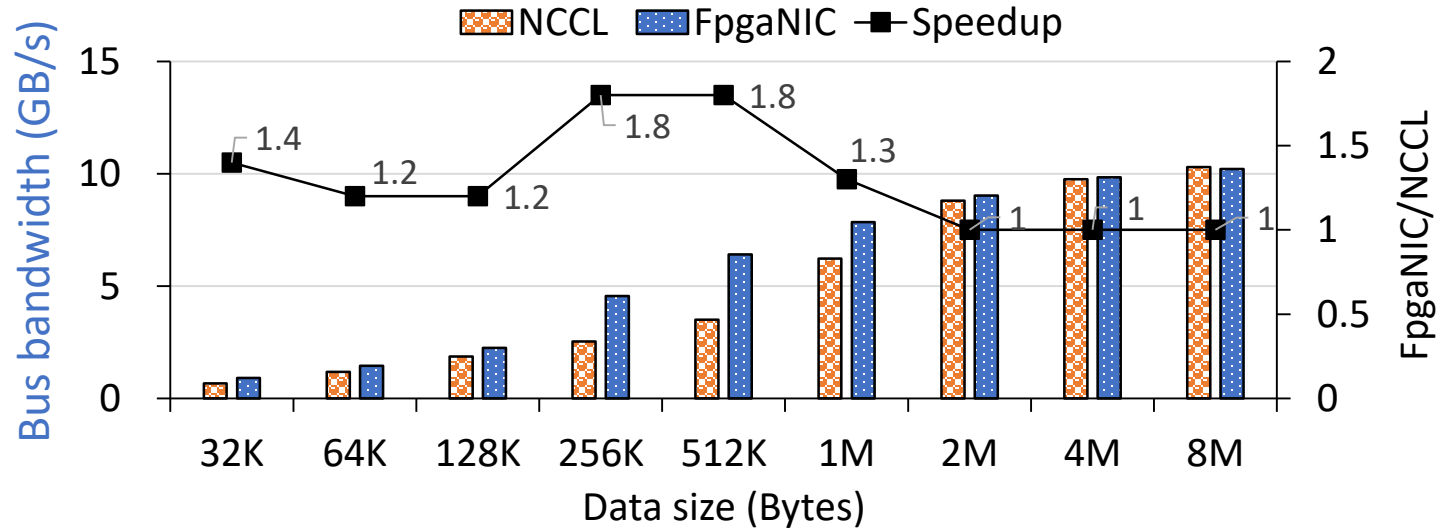
- ❑ RDMA and GPUDirect enabled
- ❑ CPU/GPU computing cycles needed
- ❑ More GPU memory footprint for intermediate states

■ **FpgaNIC**

- ❑ Pure hardware implementation
- ❑ No CPU/GPU computing cycles needed
- ❑ No GPU memory footprint for intermediate states

[1] NVIDIA NCCL <https://developer.nvidia.com/nccl>

AllReduce on eight nodes:



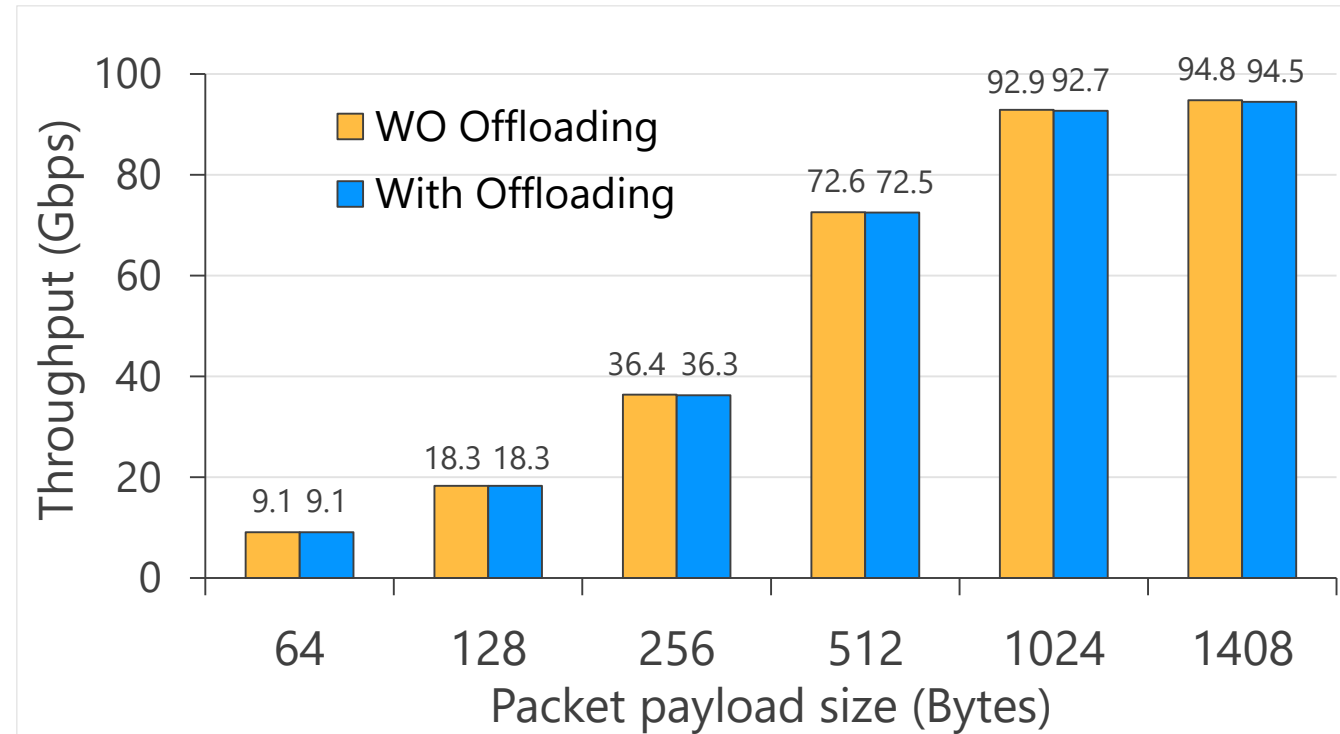
Observations:

- 1, FpgaNIC-enhanced AllReduce reaches theoretical bus bandwidth as NCCL, when data size is big enough.
- 2, FpgaNIC-enhanced AllReduce achieves higher bus bandwidth as NCCL, when data size is less than 1M.

FpgaNIC: HyperLogLog (On-path Model)

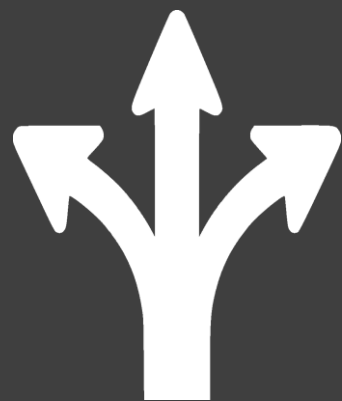


- **WO offloading:** baseline
 - Role: Estimating cardinality on GPU
 - **Need 8 A100 Streaming Multiprocessors**
- **With offloading**
 - Role: Estimating cardinality on FpgaNIC
 - **No GPU cycles**



Observation:

FpgaNIC-enhanced HyperLogLog offloading does not affect achievable throughput.



*We are happy to apply **FpgaNIC** to more **applications**.*

***FpgaNIC** is open-source:*

<https://github.com/RC4ML/FpgaNIC>.