Efficient and Flexible Datapaths for Fine-Grained Rack-Scale Interconnects with Elastic QP

Frank Chenxingyu Zhao[1], Yibo Wu[2], Hongtao Zhang[1], Jaehong Min[1], Ming Liu[2], Arvind Krishnamurthy[1]

University of Washington[1], University of Wisconsin–Madison[2]

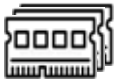
Rack-Scale Interconnects Empower Workload Scale-Up

Rack-Scale Interconnects Empower Workload Scale-Up

Workload Scale-Up

Distributed Inference

Disaggregated Storage



Memory Pooling

Rack-Scale Interconnects Empower Workload Scale-Up

Workload Scale-Up	Distributed Inference	Disaggregated Storage	Memory Pooling
Supernodes Emerge	 NVIDIA NVL72	 AMD Helios	UALink OCP ESUN $Millions per Rack!

Rack-Scale Interconnects Empower Workload Scale-Up

Workload Scale-Up	**Distributed Inference**	**Disaggregated Storage**	**Memory Pooling**
Supernodes Emerge	NVIDIA **NVL72**	AMD **Helios**	UALink, OCP ESUN **$Millions per Rack!**
Sweet Middle Spot	**Server**	**More Devices** → **Rack** ← **More Intensive Comm.**	**Cluster**

Rack-Scale Interconnects Empower Workload Scale-Up

Workload Scale-Up

Distributed Inference

Disaggregated Storage

Memory Pooling

Supernodes Emerge

NVIDIA **NVL72**

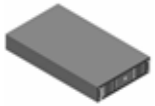
AMD **Helios**

UALink

OCP ESUN

$Millions per Rack!

Sweet Middle Spot

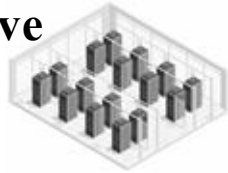


Server

More Devices →

Rack

← **More Intensive Comm.**



Cluster

How to build efficient fine-grained datapaths for rack-scale?

Background: Fine-Grained Memory-Semantic Datapath

Background: Fine-Grained Memory-Semantic Datapath

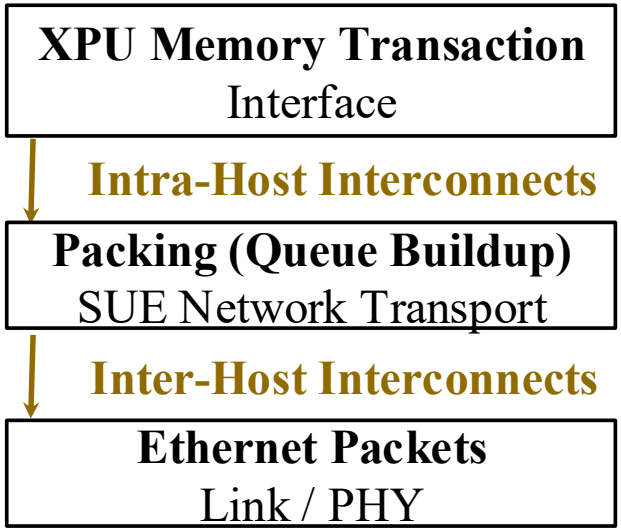
- **Remote Memory Access (RMA)** in Load/Store (*e.g.*, NVSHMEM)

Background: Fine-Grained Memory-Semantic Datapath

- **Remote Memory Access (RMA)** in Load/Store (*e.g.*, NVSHMEM)
- Datapath Specification: OCP *Scale-Up Ethernet (SUE)*

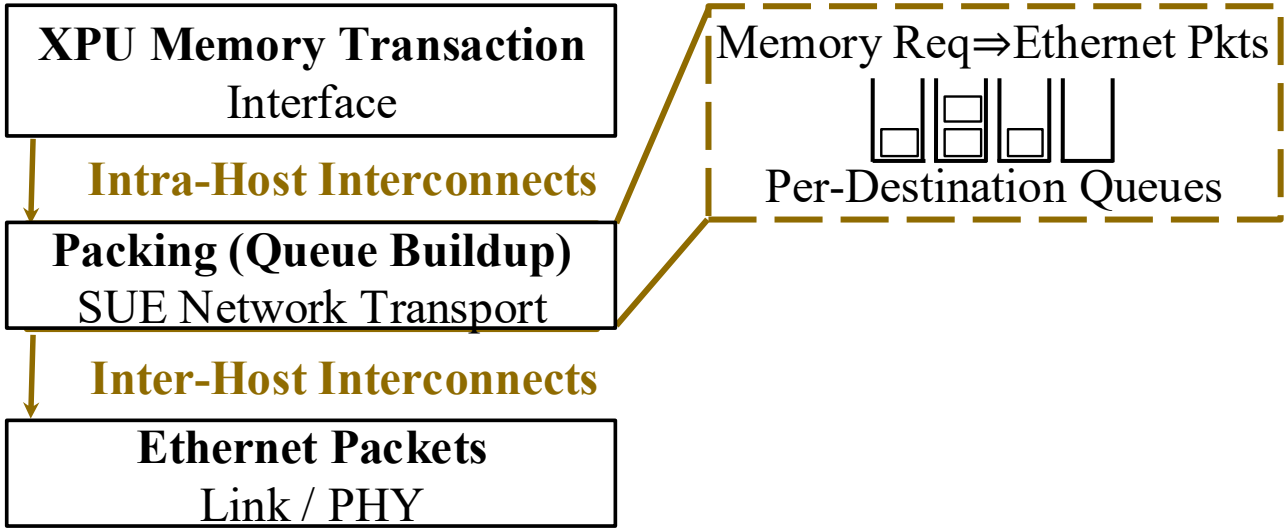
Background: Fine-Grained Memory-Semantic Datapath

- **Remote Memory Access (RMA)** in Load/Store (*e.g.*, NVSHMEM)
- Datapath Specification: OCP *Scale-Up Ethernet (SUE)*



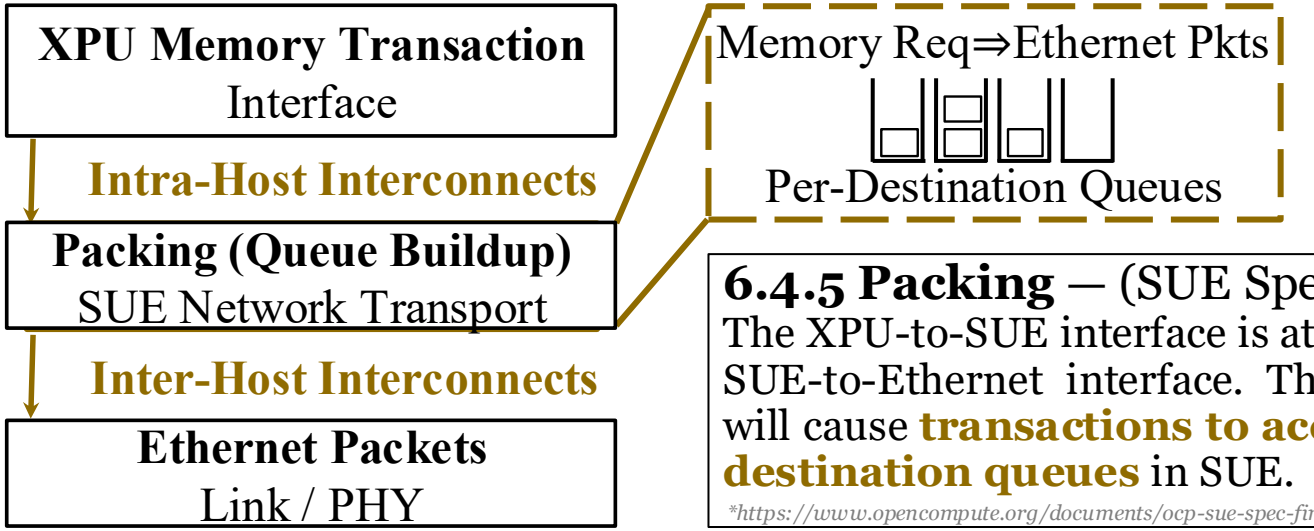
Background: Fine-Grained Memory-Semantic Datapath

- **Remote Memory Access (RMA)** in Load/Store (*e.g.*, NVSHMEM)
- Datapath Specification: OCP *Scale-Up Ethernet (SUE)*



Background: Fine-Grained Memory-Semantic Datapath

- **Remote Memory Access (RMA)** in Load/Store (*e.g.*, NVSHMEM)
- Datapath Specification: OCP *Scale-Up Ethernet (SUE)*



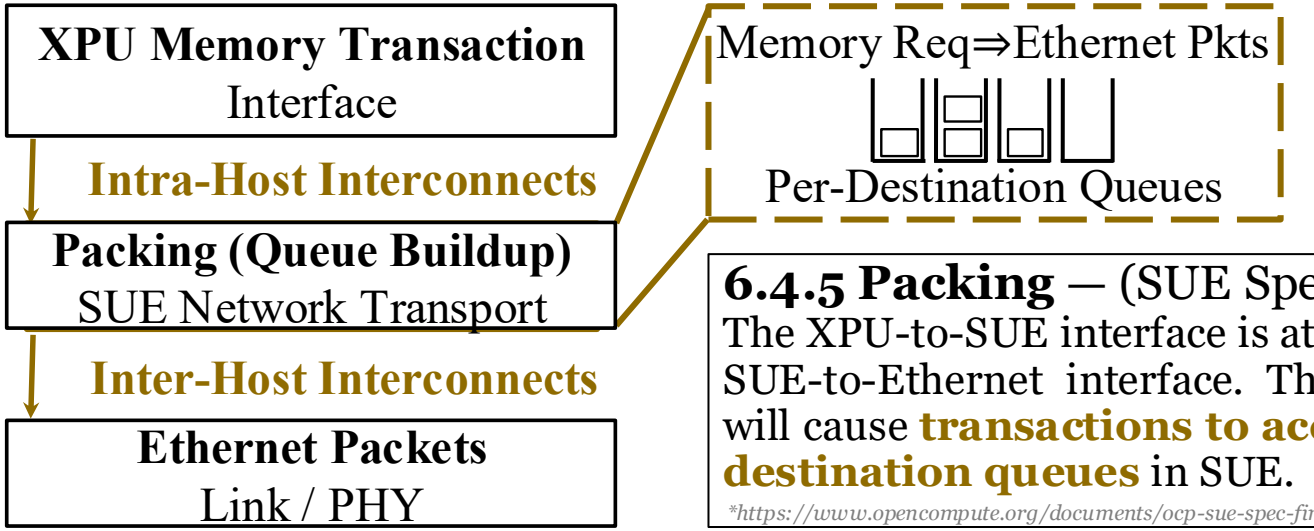
6.4.5 Packing — (SUE Spec v1.0*)

The XPU-to-SUE interface is at higher speed than the SUE-to-Ethernet interface. This **speed mismatch** will cause **transactions to accumulate in the per destination queues** in SUE.

**https://www.opencompute.org/documents/ocp-sue-spec-final-pdf-1*

Background: Fine-Grained Memory-Semantic Datapath

- **Remote Memory Access (RMA)** in Load/Store (*e.g.*, NVSHMEM)
- Datapath Specification: OCP *Scale-Up Ethernet (SUE)*



6.4.5 Packing — (SUE Spec v1.0*)

The XPU-to-SUE interface is at higher speed than the SUE-to-Ethernet interface. This **speed mismatch** will cause **transactions to accumulate in the per destination queues** in SUE.

*https://www.opencompute.org/documents/ocp-sue-spec-final-pdf-1

RMA Queue Buildup is common in fine-grained SUE

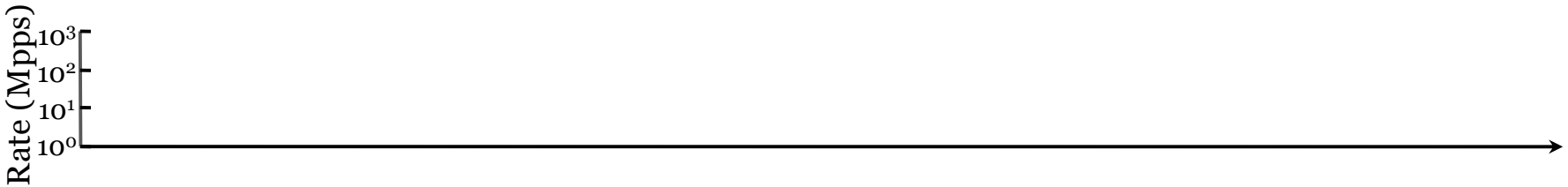
Motivation: How to Max Out Ethernet for RMA Queuing?

Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues

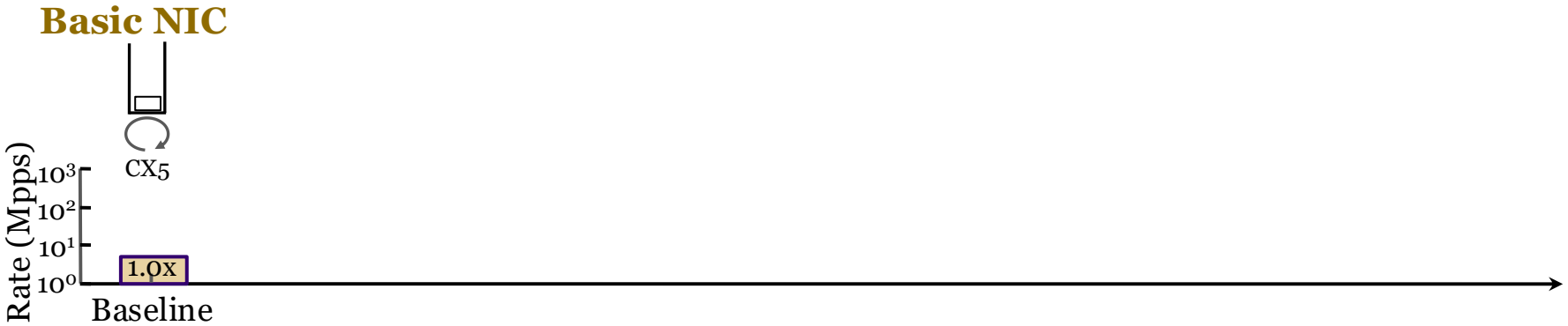
Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues



Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues

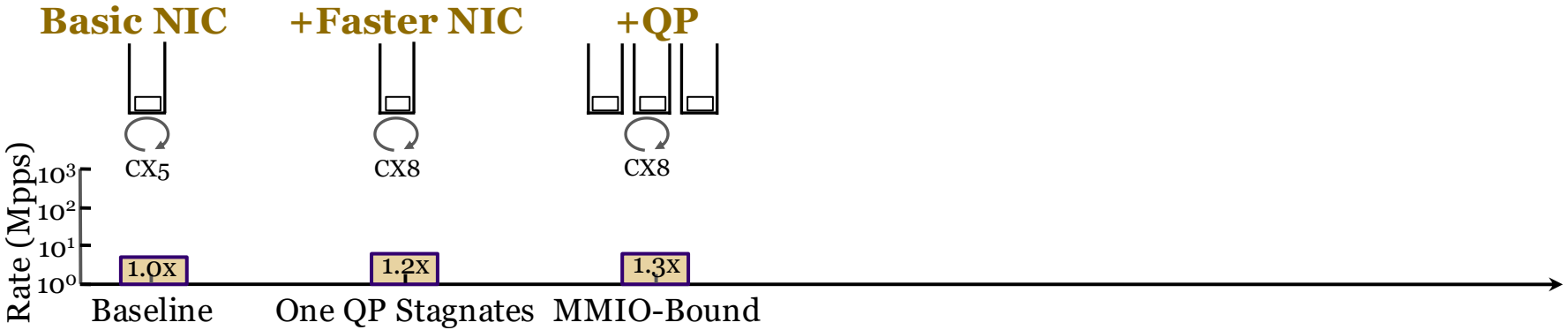


Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues

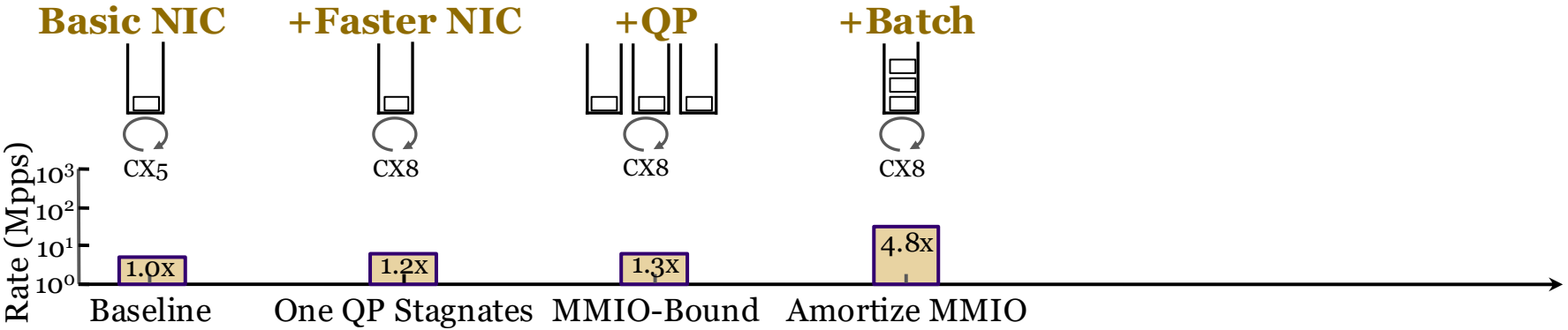
Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues



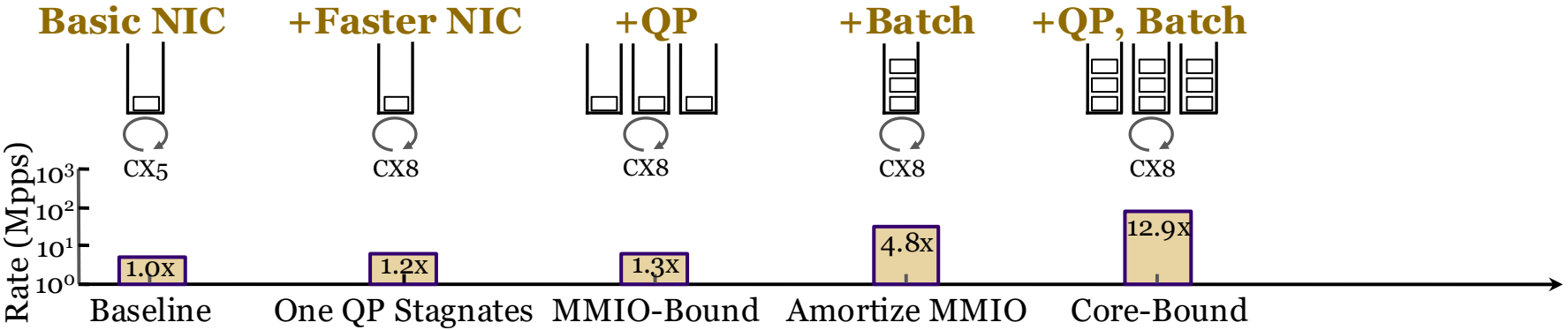
Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues



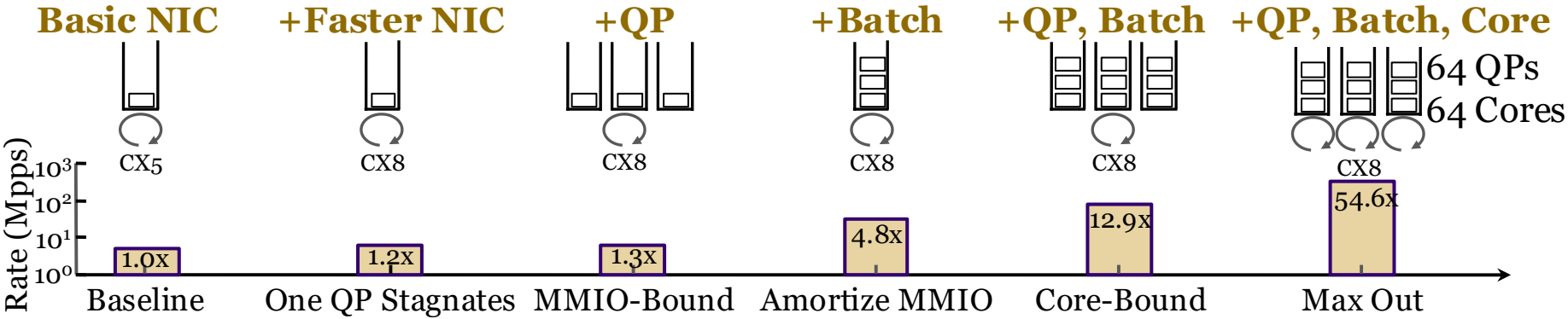
Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues



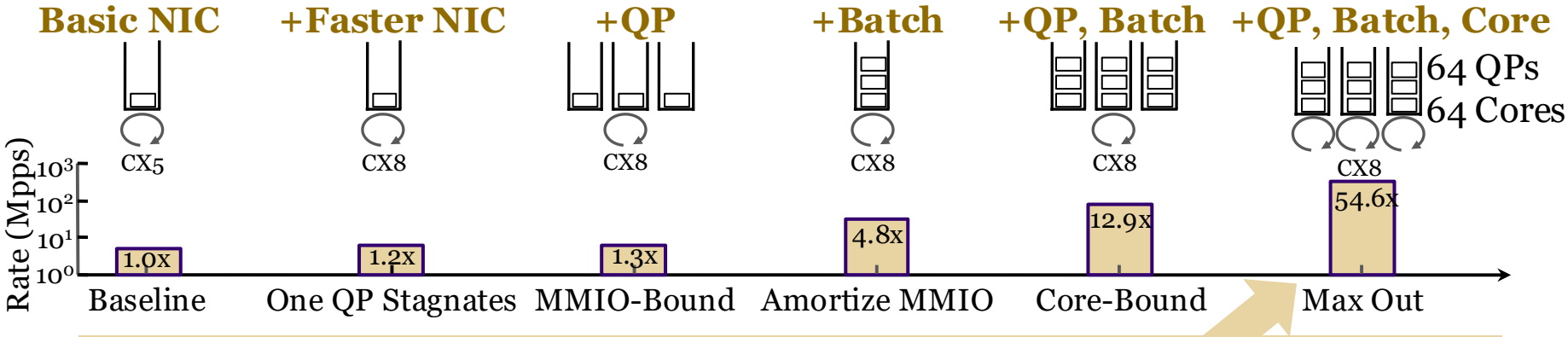
Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues



Motivation: How to Max Out Ethernet for RMA Queuing?

- **Characterization Experiments**
  - **Hardware:** RoCEv2 on NVIDIA ConnectX-8 RDMA NICs
  - **Metric:** Message Rate for Fine-Grained Access (64 B) with Deep Queues



High throughput with multiple QPs/cores comes at a cost:

- ❑ **Higher Compute Overhead**
- ❑ **Stateful Multi-QP Tracking**

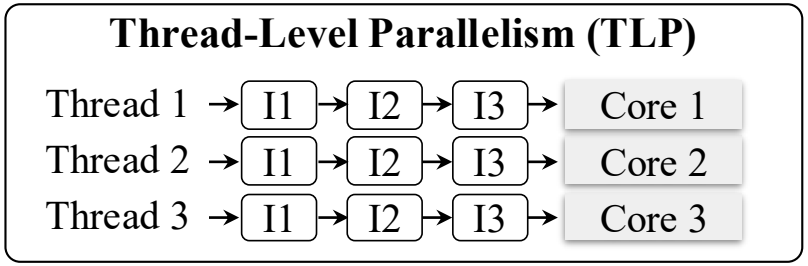
Motivation: Can We Exploit Parallelism Beyond Multi-QP?

Motivation: Can We Exploit Parallelism Beyond Multi-QP?

- **Inspiration:** Classical CPU Parallelism

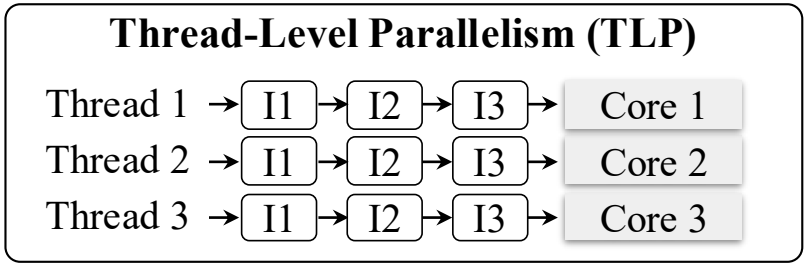
Motivation: Can We Exploit Parallelism Beyond Multi-QP?

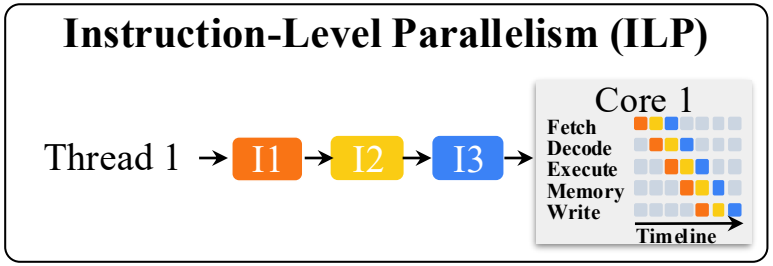
- **Inspiration:** Classical CPU Parallelism



Motivation: Can We Exploit Parallelism Beyond Multi-QP?

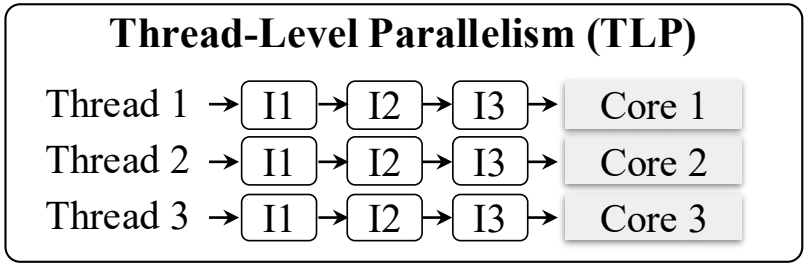
- **Inspiration:** Classical CPU Parallelism

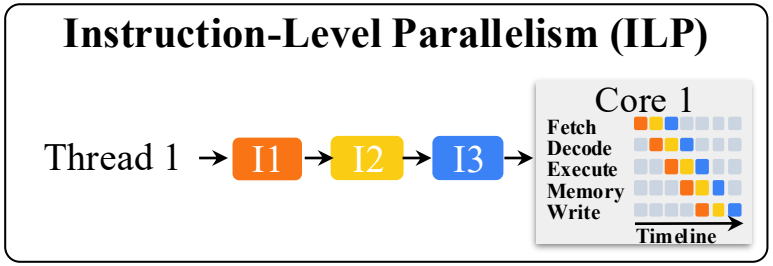




Motivation: Can We Exploit Parallelism Beyond Multi-QP?

- **Inspiration:** Classical CPU Parallelism

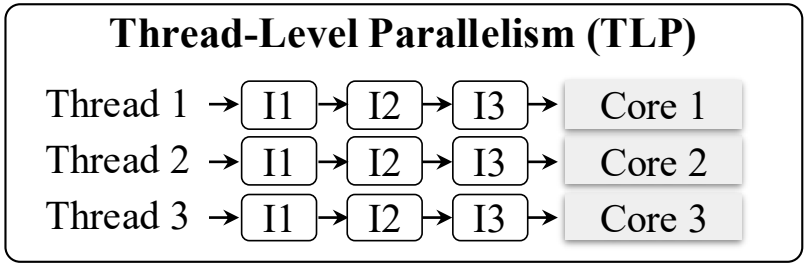


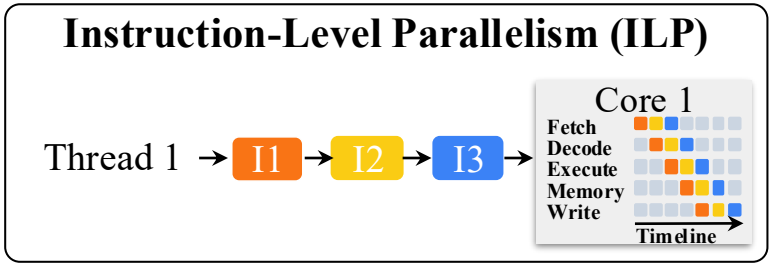


- **RDMA NIC:** QP-Level Parallelism *vs.* Intra-QP Parallelism

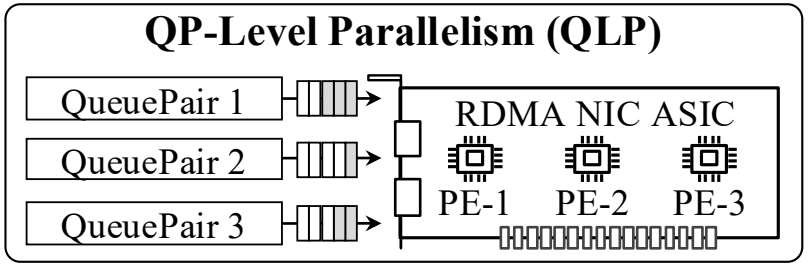
Motivation: Can We Exploit Parallelism Beyond Multi-QP?

- **Inspiration:** Classical CPU Parallelism



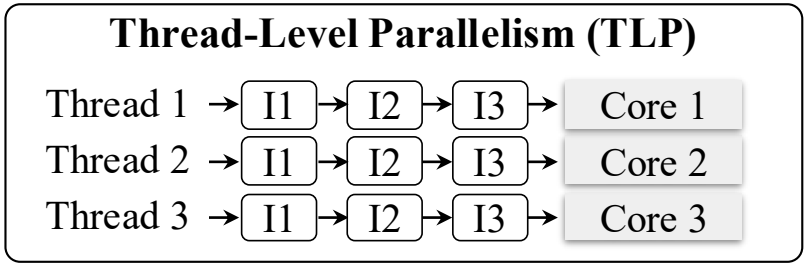


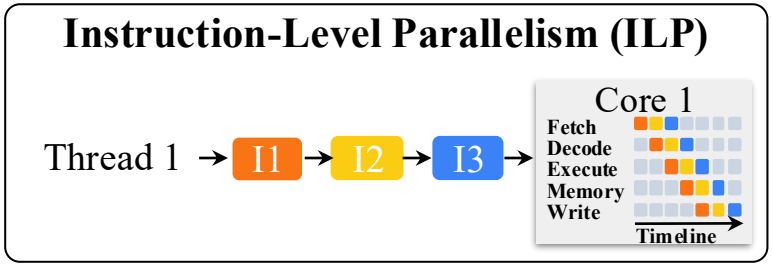
- **RDMA NIC:** QP-Level Parallelism *vs.* Intra-QP Parallelism



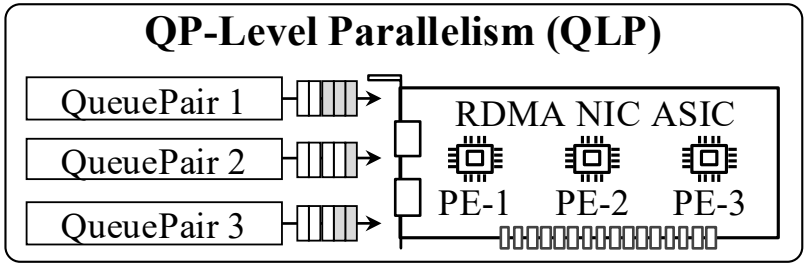
Motivation: Can We Exploit Parallelism Beyond Multi-QP?

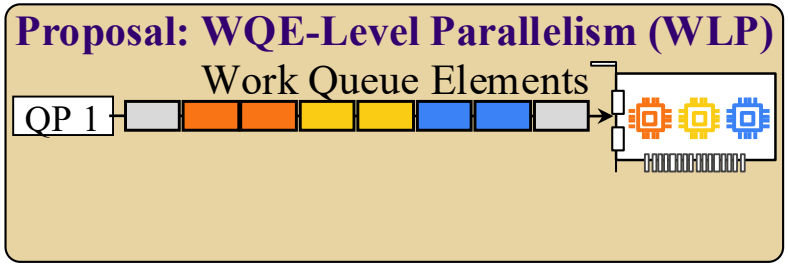
- **Inspiration:** Classical CPU Parallelism





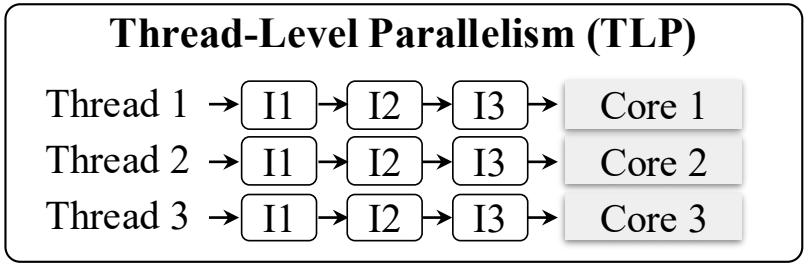
- **RDMA NIC:** QP-Level Parallelism *vs.* Intra-QP Parallelism

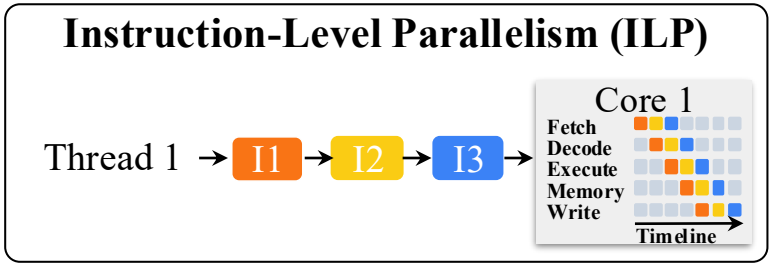




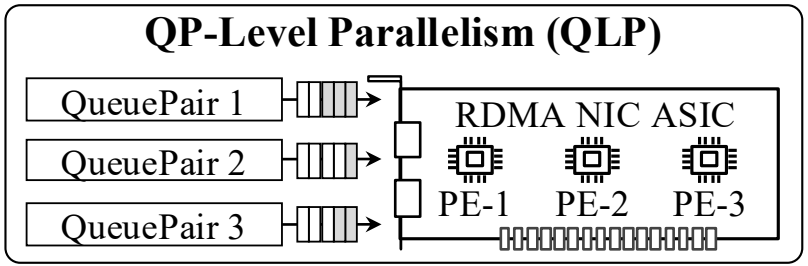
Motivation: Can We Exploit Parallelism Beyond Multi-QP?

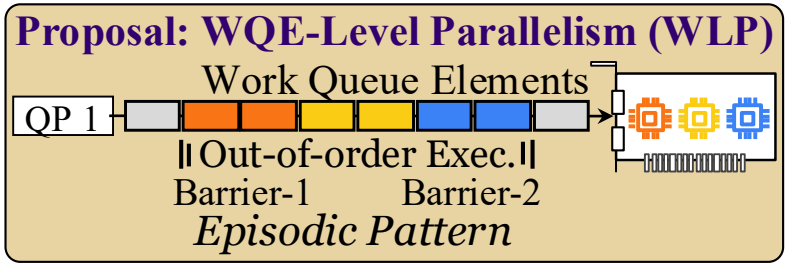
- **Inspiration:** Classical CPU Parallelism





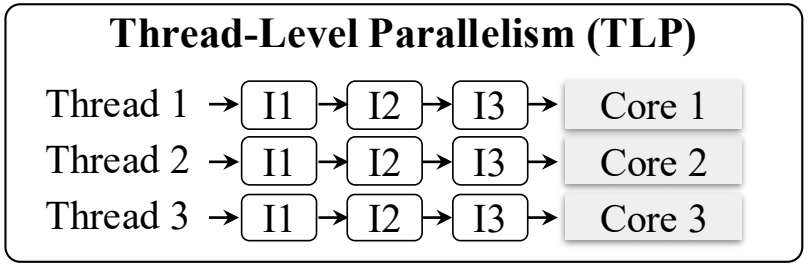
- **RDMA NIC:** QP-Level Parallelism *vs.* Intra-QP Parallelism

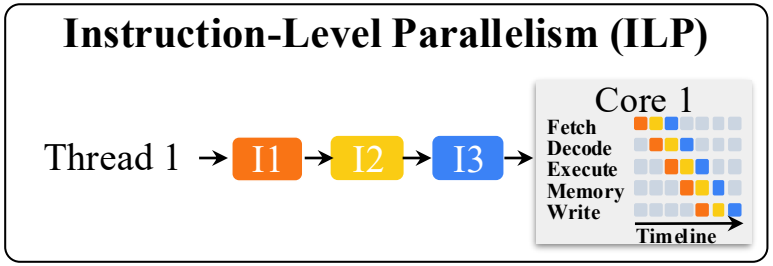




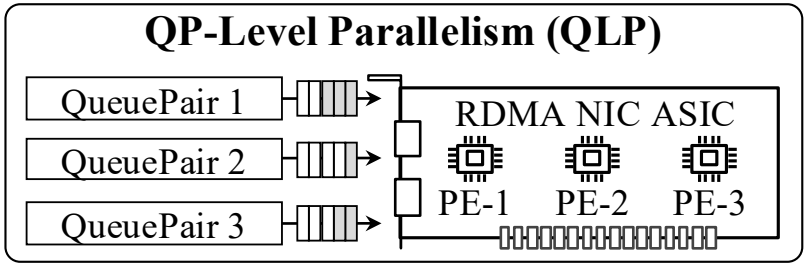
Motivation: Can We Exploit Parallelism Beyond Multi-QP?

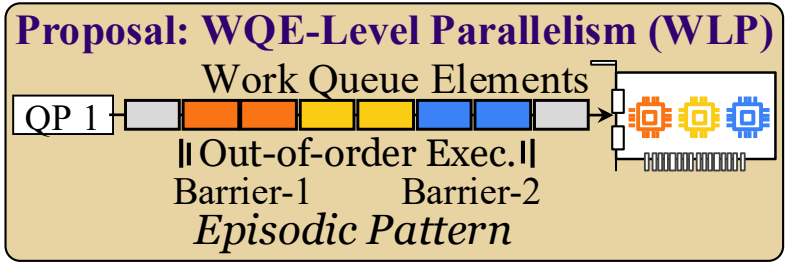
- **Inspiration:** Classical CPU Parallelism





- **RDMA NIC:** QP-Level Parallelism *vs.* Intra-QP Parallelism



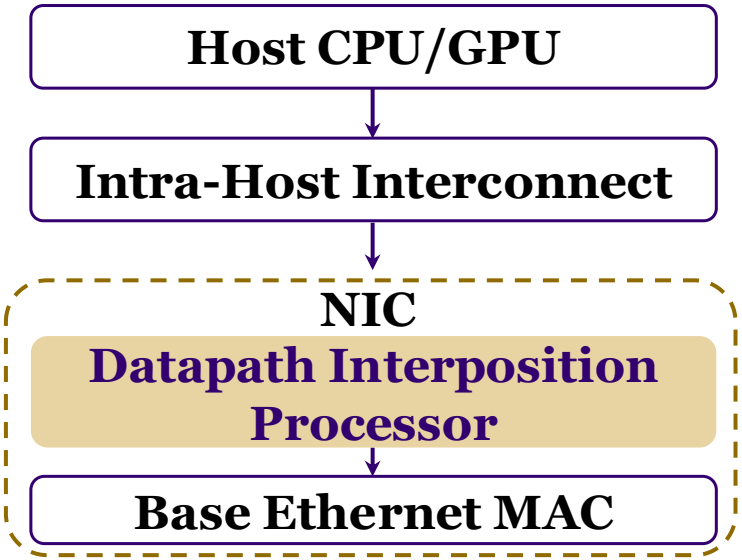


WLP enables parallel execution within a single QP

Enabler: On-NIC Interposition Processor

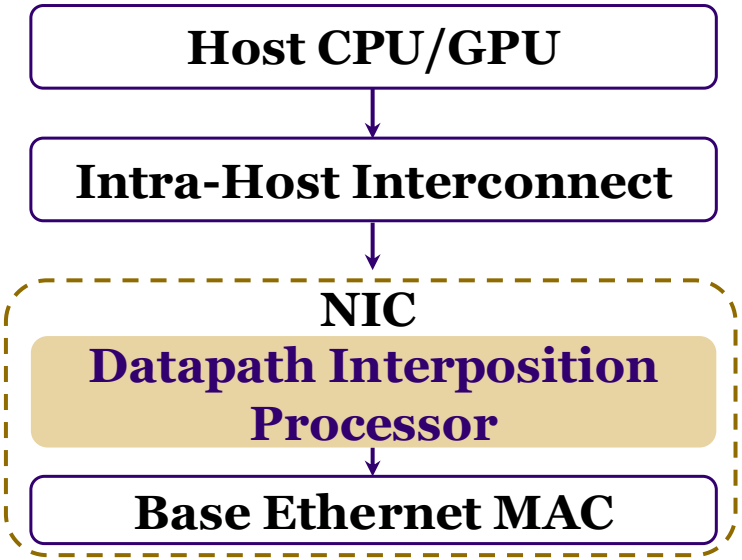
Enabler: On-NIC Interposition Processor

- **Chokepoint:** NIC bridging intra-host/inter-host interconnects



Enabler: On-NIC Interposition Processor

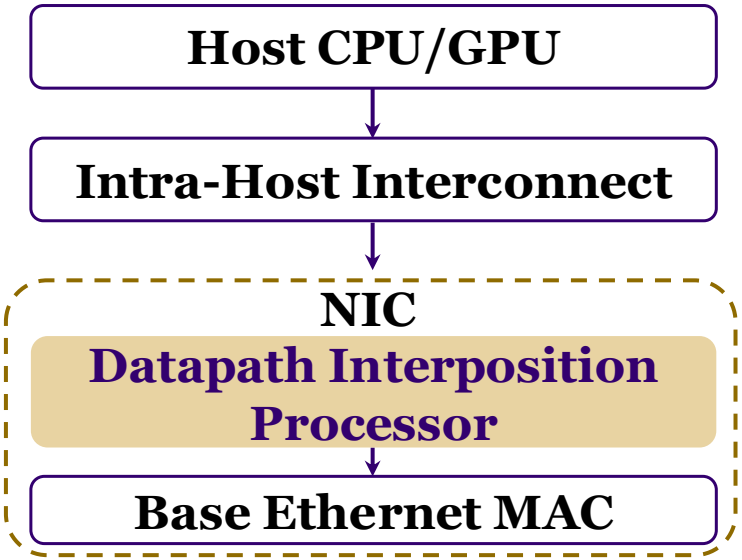
- **Chokepoint:** NIC bridging intra-host/inter-host interconnects



On-Datapath Programmability

Enabler: On-NIC Interposition Processor

- **Chokepoint:** NIC bridging intra-host/inter-host interconnects

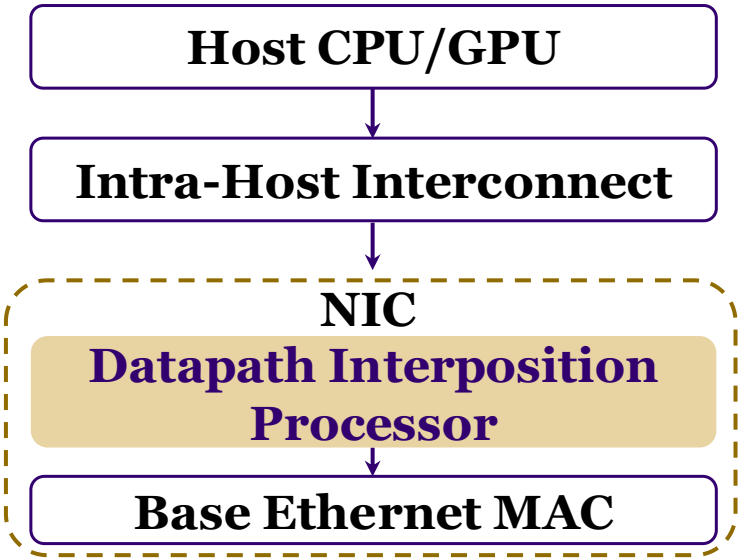


On-Datapath Programmability

LD/ST Access to Host-side Memory

Enabler: On-NIC Interposition Processor

- **Chokepoint:** NIC bridging intra-host/inter-host interconnects



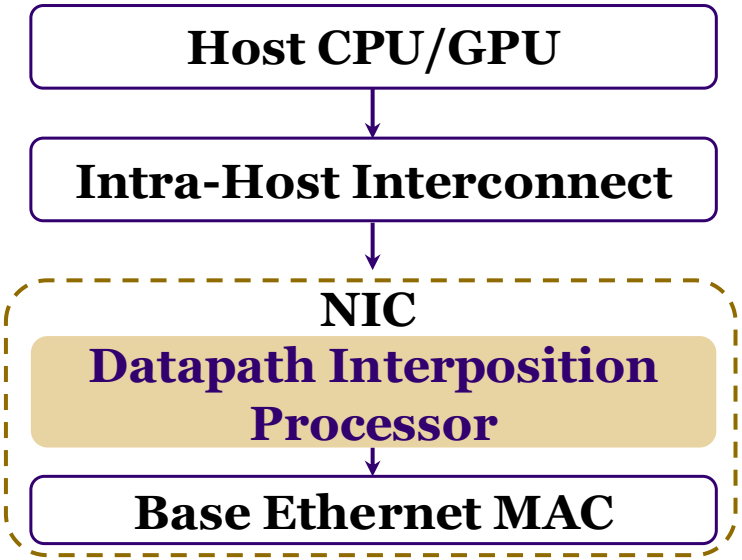
On-Datapath Programmability

LD/ST Access to Host-side Memory

Manycore Processor w/On-NIC Mem

Enabler: On-NIC Interposition Processor

- **Chokepoint:** NIC bridging intra-host/inter-host interconnects

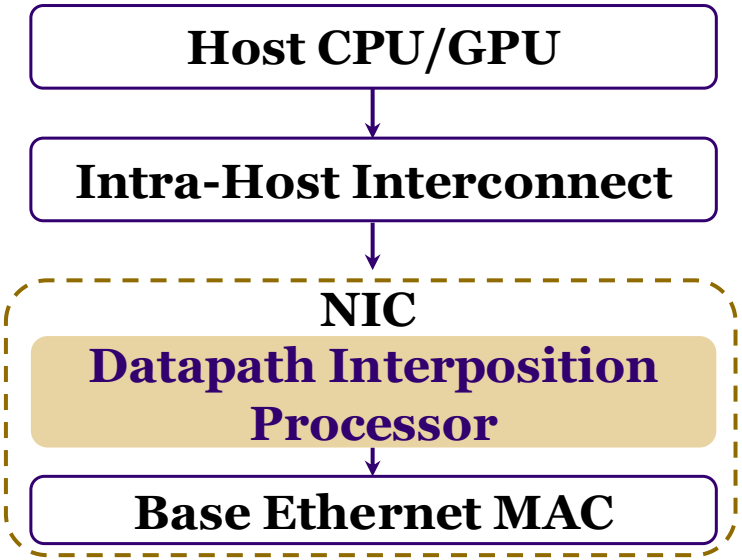


On-Datapath Programmability

- **LD/ST Access to Host-side Memory**
- **Manycore Processor w/On-NIC Mem**
- **Integration with the Ethernet MAC**

Enabler: On-NIC Interposition Processor

- **Chokepoint:** NIC bridging intra-host/inter-host interconnects



On-Datapath Programmability

- **LD/ST Access to Host-side Memory**
- **Manycore Processor w/On-NIC Mem**
- **Integration with the Ethernet MAC**

NICs are embedding on-path interposition processors

Key Idea and Design Overview

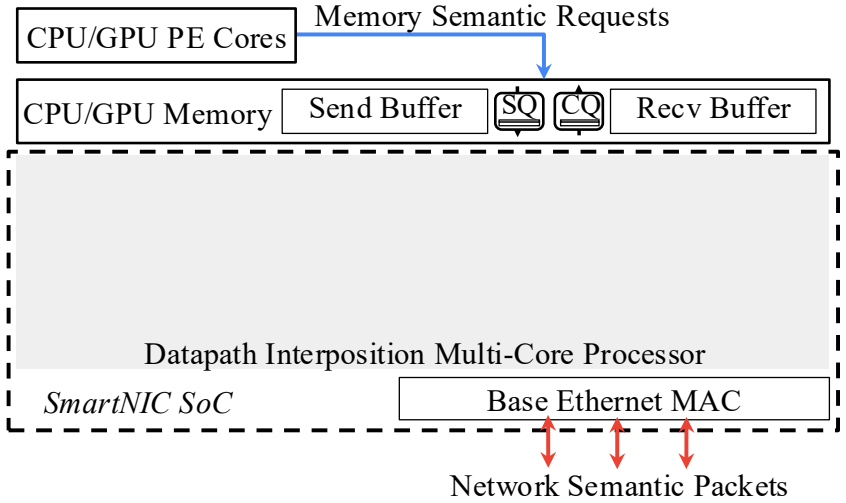
Key Idea and Design Overview

Can we enable *WQE-Level Parallelism (WLP)* via *On-NIC Interposition Processors*?

Key Idea and Design Overview

Can we enable *WQE-Level Parallelism (WLP)* via *On-NIC Interposition Processors*?

Software-Interposed Datapath (SID)



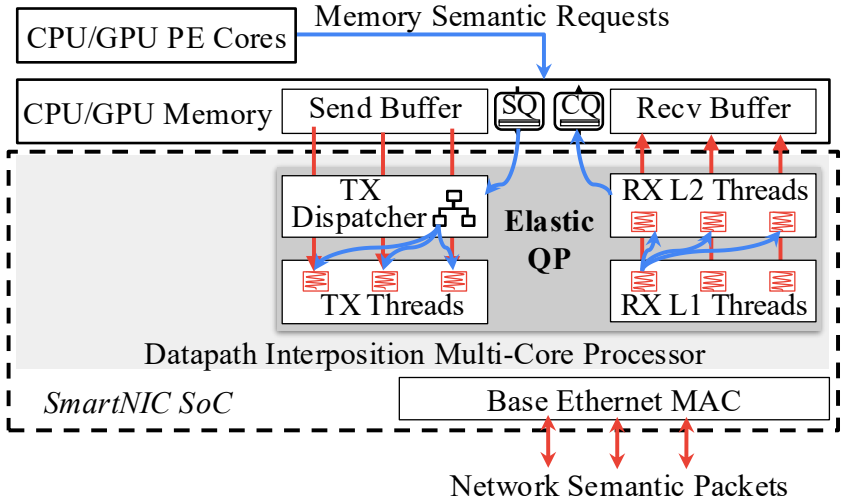
Key Idea and Design Overview

Can we enable *WQE-Level Parallelism (WLP)* via *On-NIC Interposition Processors*?

Software-Interposed Datapath (SID)

Elastic QP

- ❑ Request Ctrl Path/Packet Data Path
- ❑ TX/RX Sides



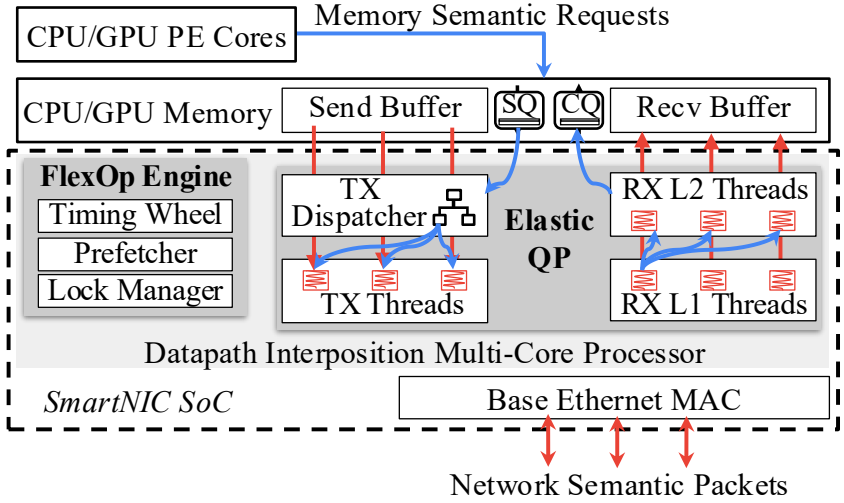
Key Idea and Design Overview

Can we enable *WQE-Level Parallelism (WLP)* via *On-NIC Interposition Processors*?

Software-Interposed Datapath (SID)

Elastic QP
- ❑ Request Ctrl Path/Packet Data Path
- ❑ TX/RX Sides

FlexOp Engine
- ❑ Operation Set/Transport/PCIe



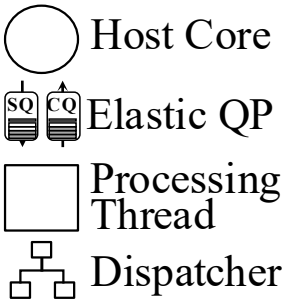
Technique #1: TX-Side Elastic Queue Pair (1/2)

Technique #1: TX-Side Elastic Queue Pair (1/2)

- **Dispatching** requests to multiple parallel processing threads

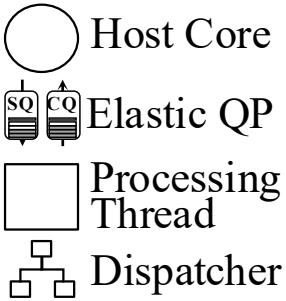
Technique #1: TX-Side Elastic Queue Pair (1/2)

- **Dispatching** requests to multiple parallel processing threads

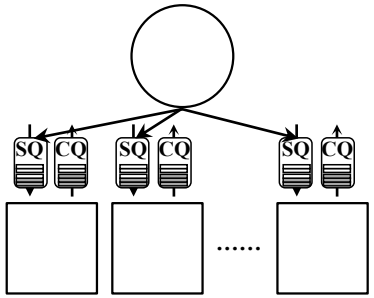


Technique #1: TX-Side Elastic Queue Pair (1/2)

- **Dispatching** requests to multiple parallel processing threads



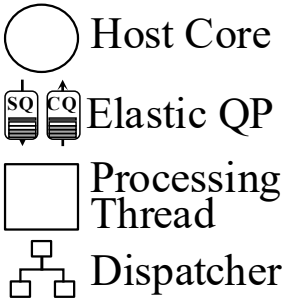
#1 Multi SPSC
Single-Producer
Single-Consumer



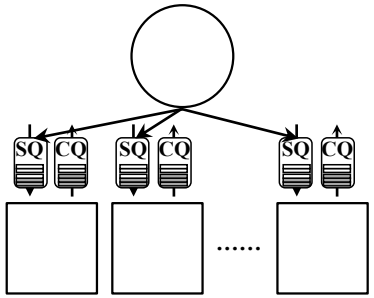
High Overhead for State Tracking

Technique #1: TX-Side Elastic Queue Pair (1/2)

- **Dispatching** requests to multiple parallel processing threads

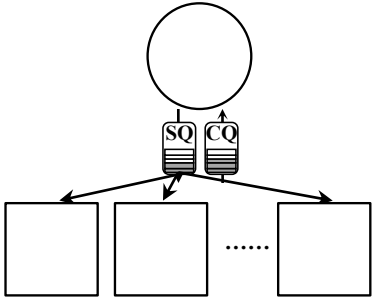


#1 Multi SPSC
Single-Producer
Single-Consumer



High Overhead for State Tracking

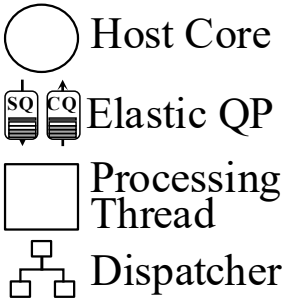
#2 Central SPMC
Single-Producer
Multi-Consumer

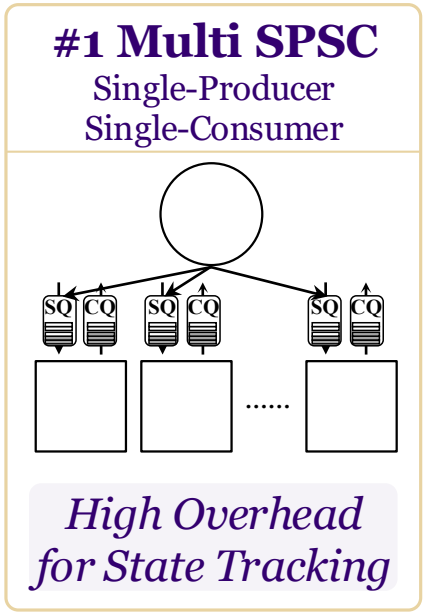


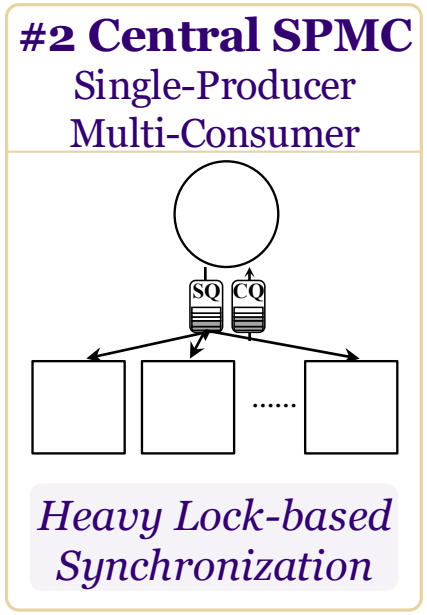
Heavy Lock-based Synchronization

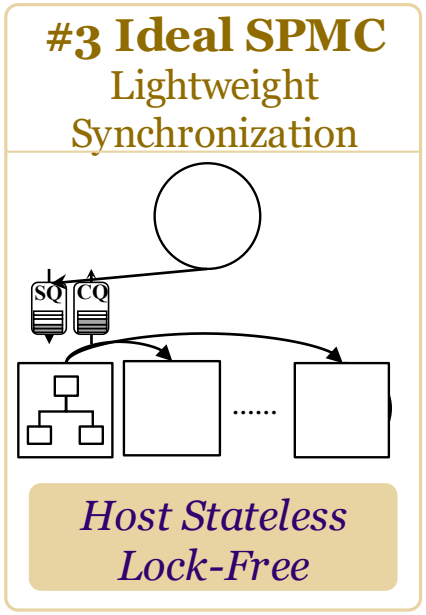
Technique #1: TX-Side Elastic Queue Pair (1/2)

- **Dispatching** requests to multiple parallel processing threads



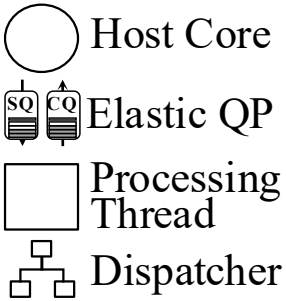


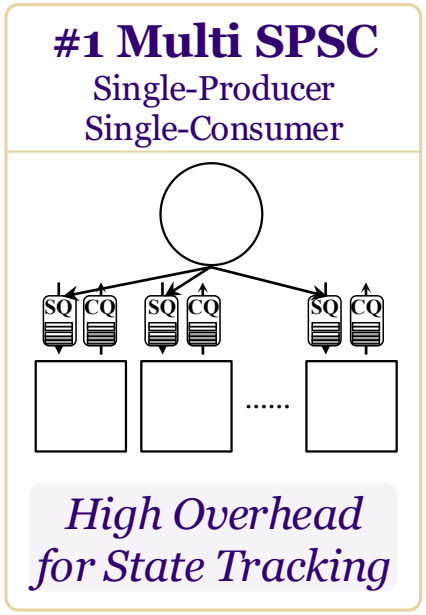


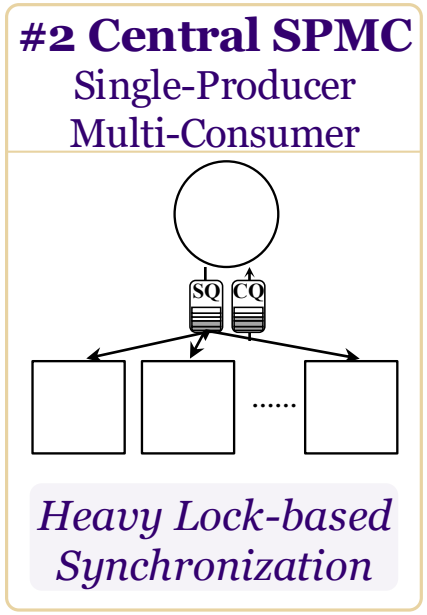


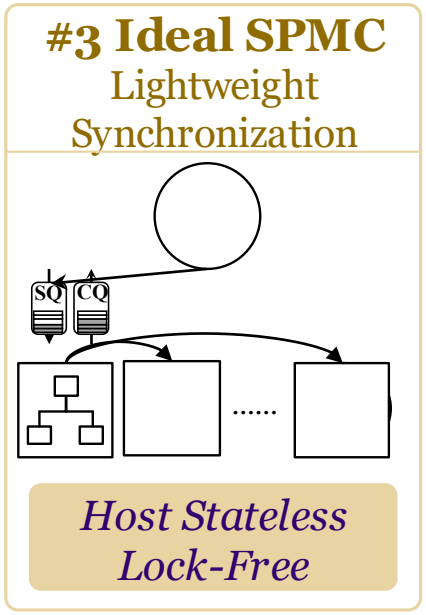
Technique #1: TX-Side Elastic Queue Pair (1/2)

- **Dispatching** requests to multiple parallel processing threads









Challenge: preserve ordering and efficiently dispatch requests

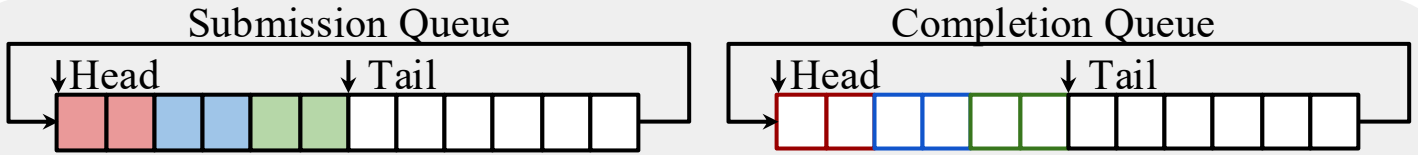
Technique #1: TX-Side Elastic Queue Pair (2/2)

Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free

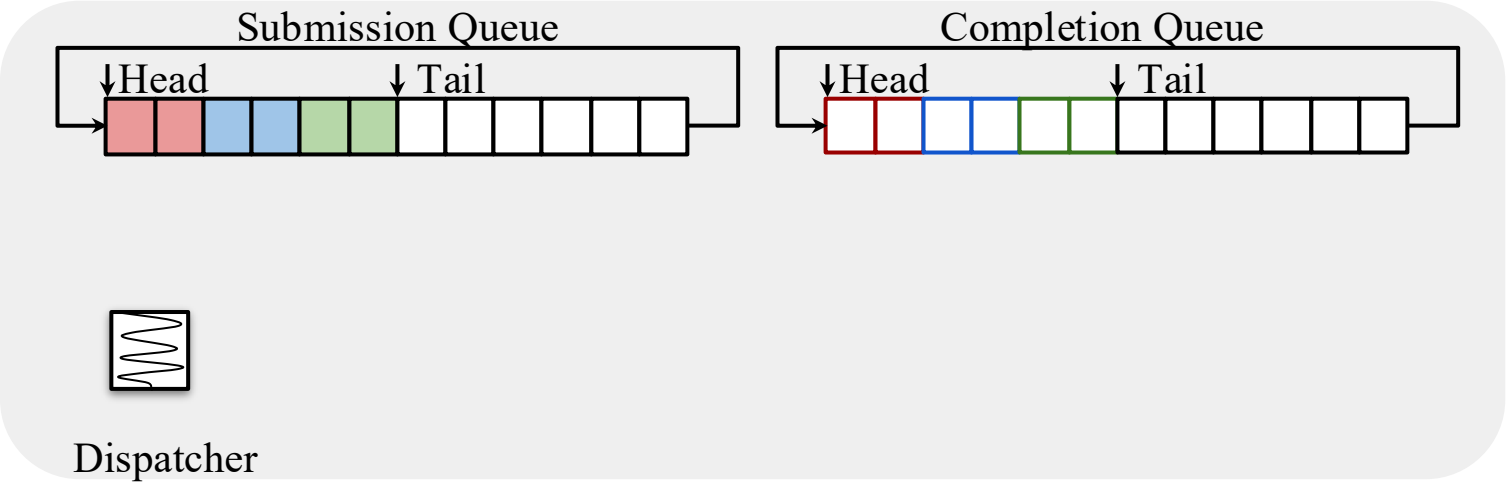
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



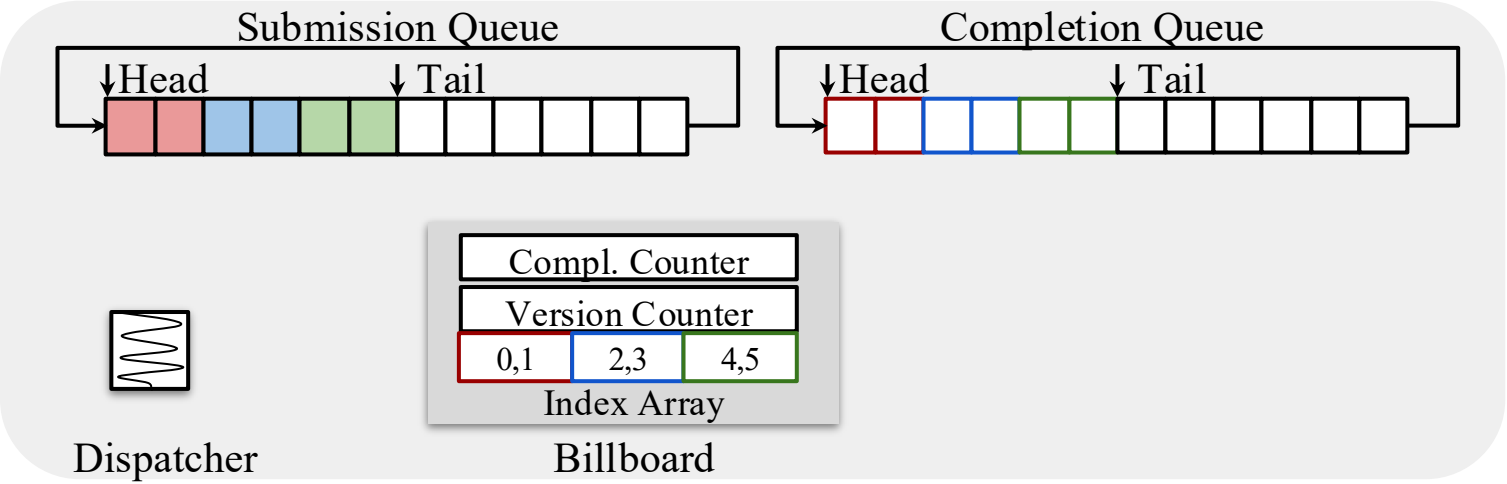
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



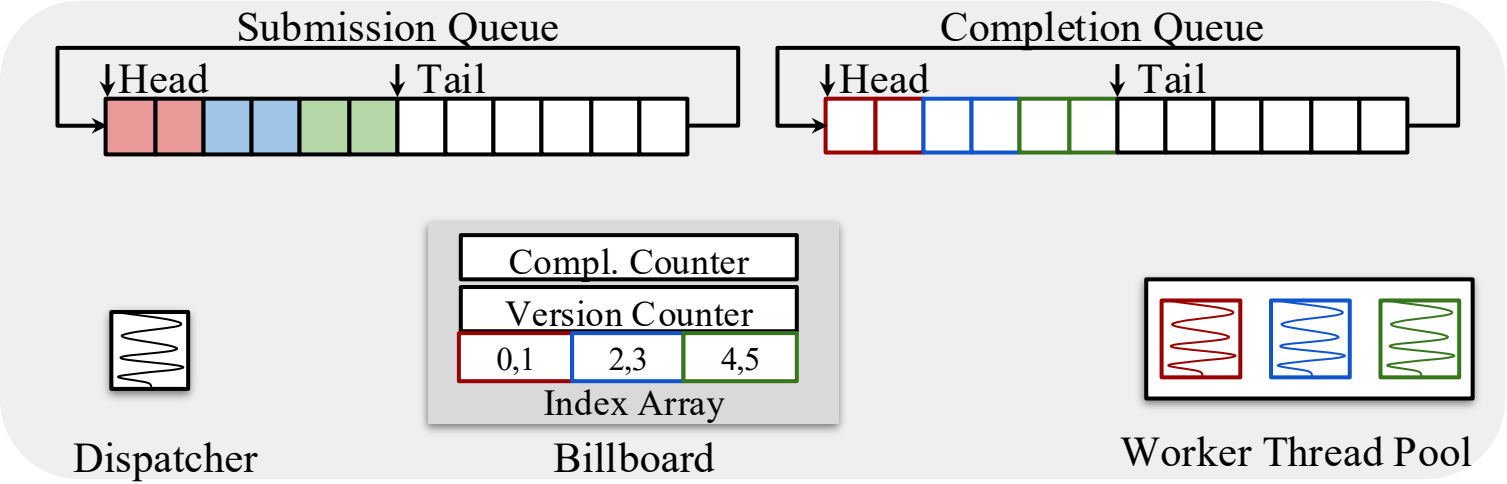
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



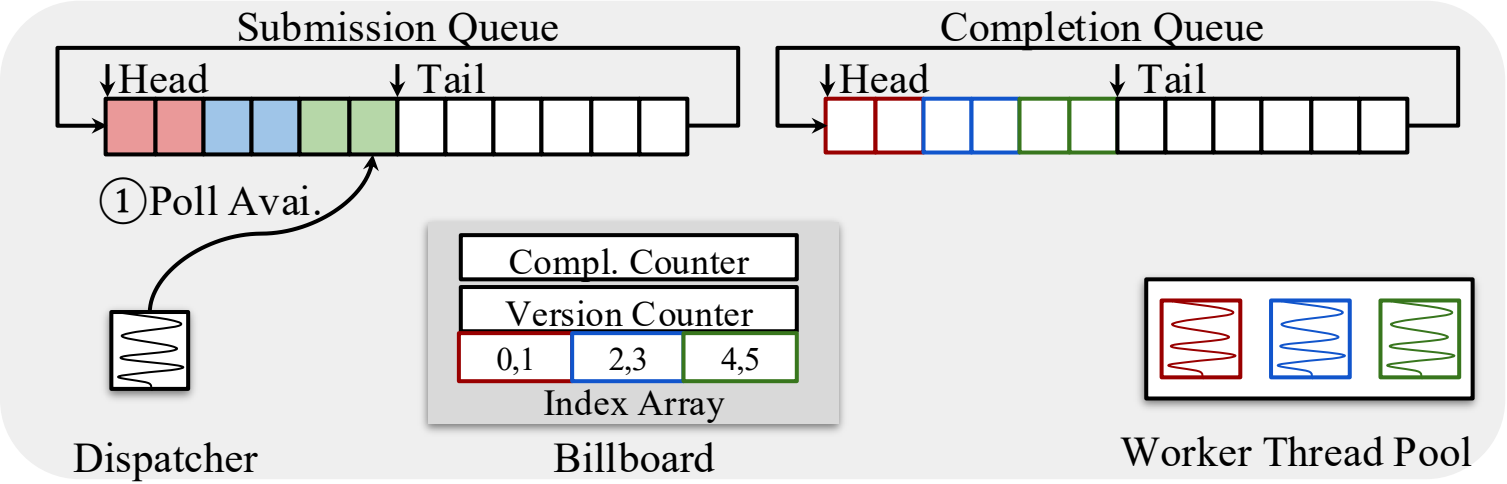
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



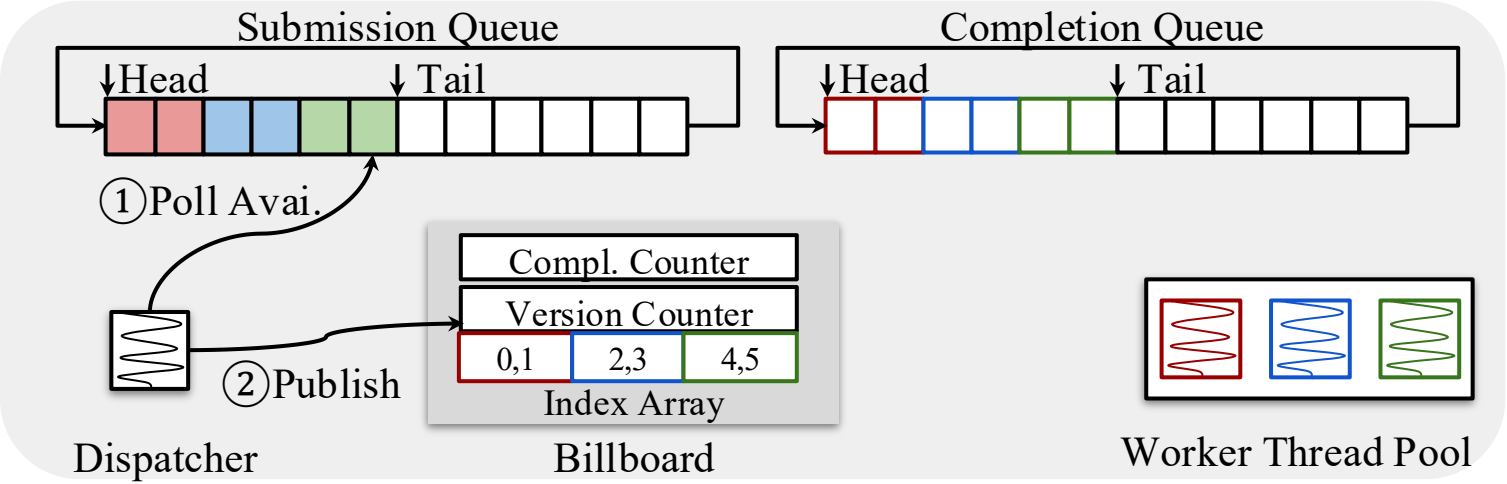
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



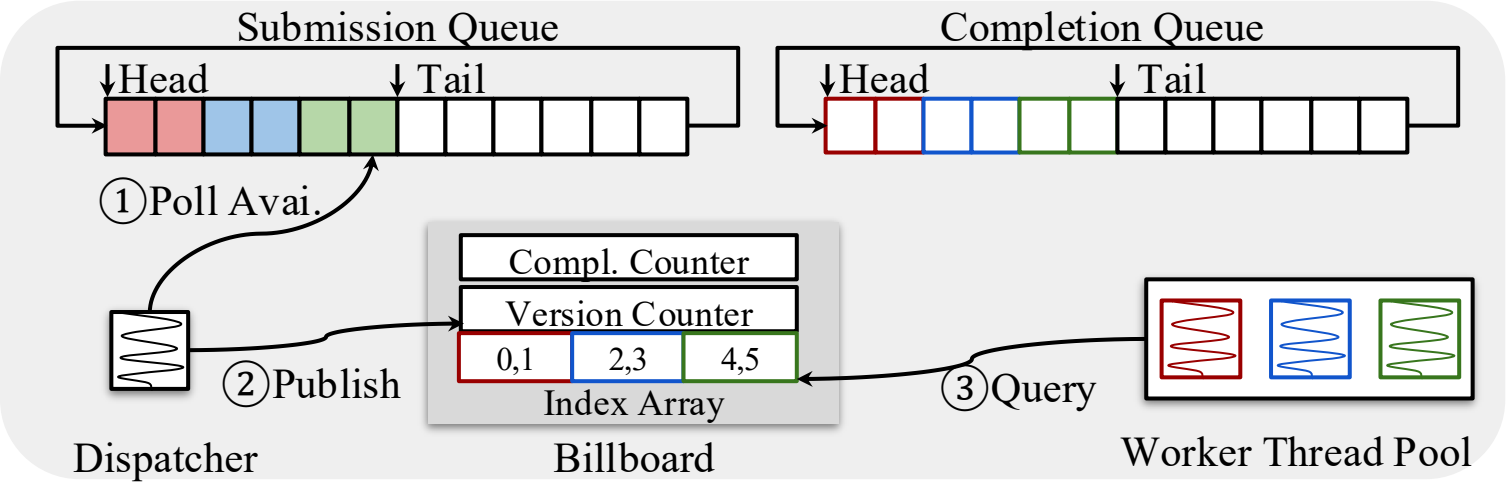
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



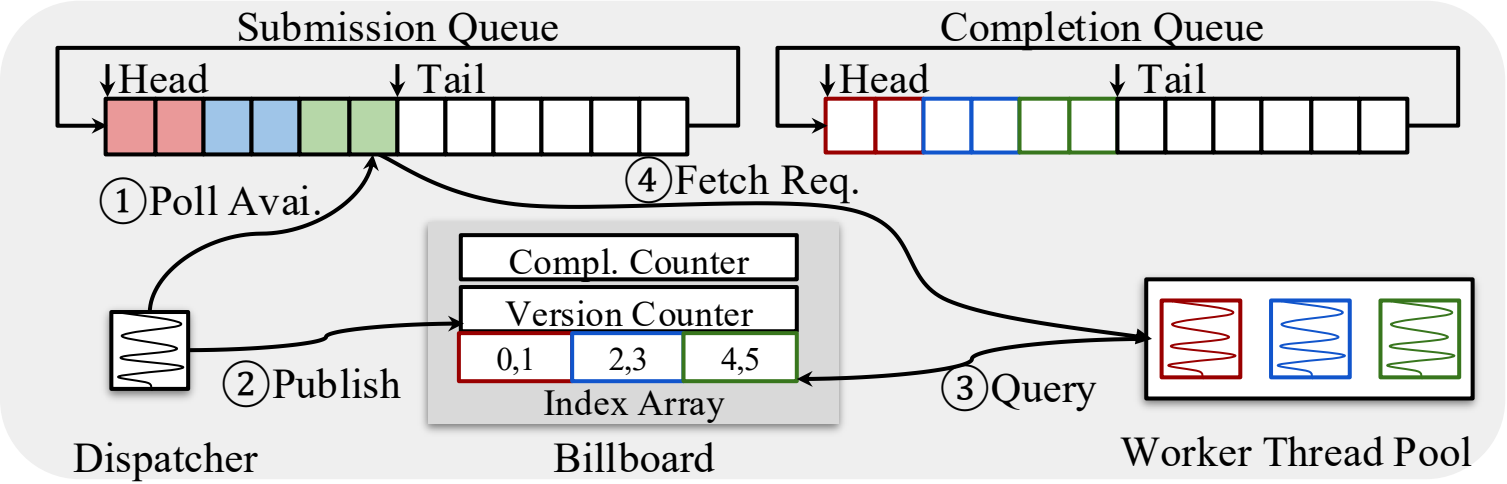
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



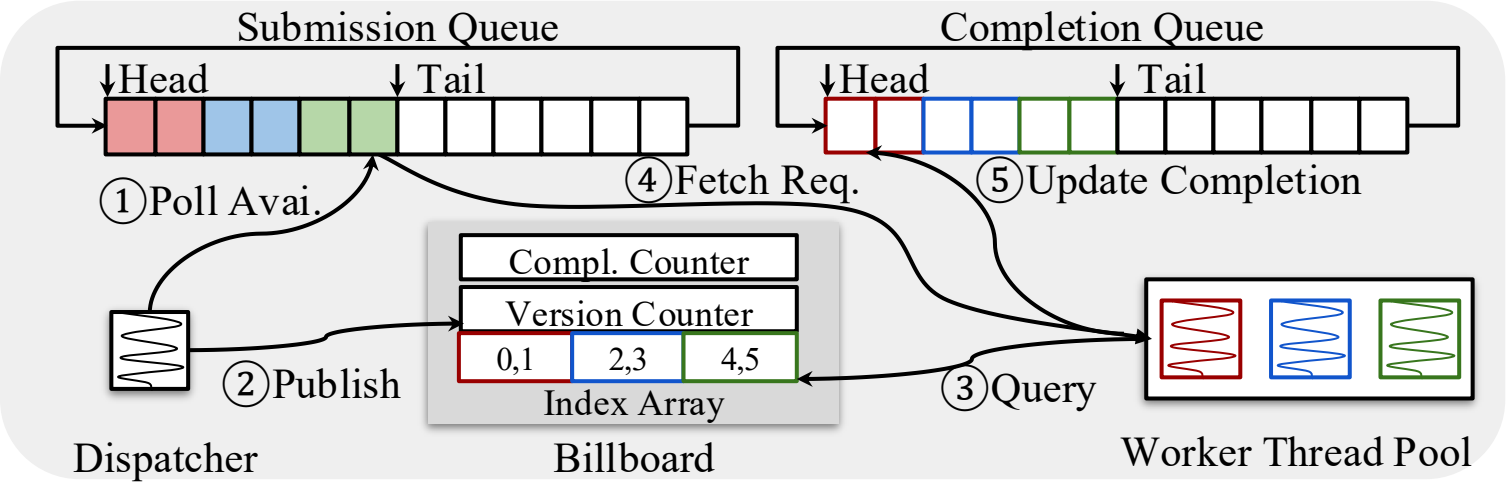
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



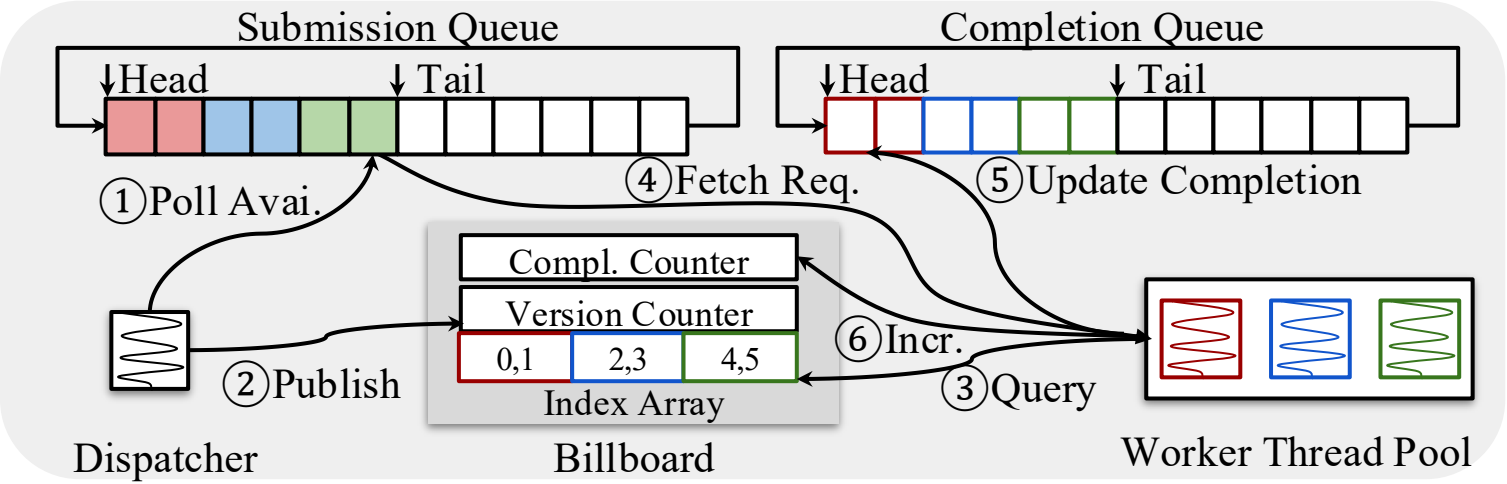
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



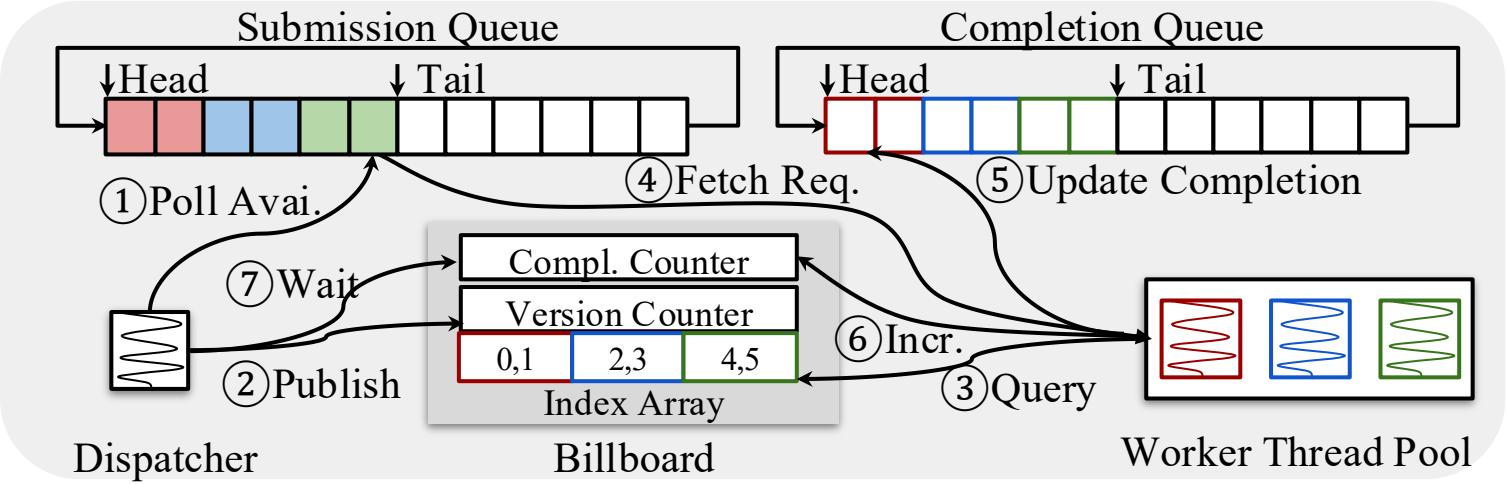
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



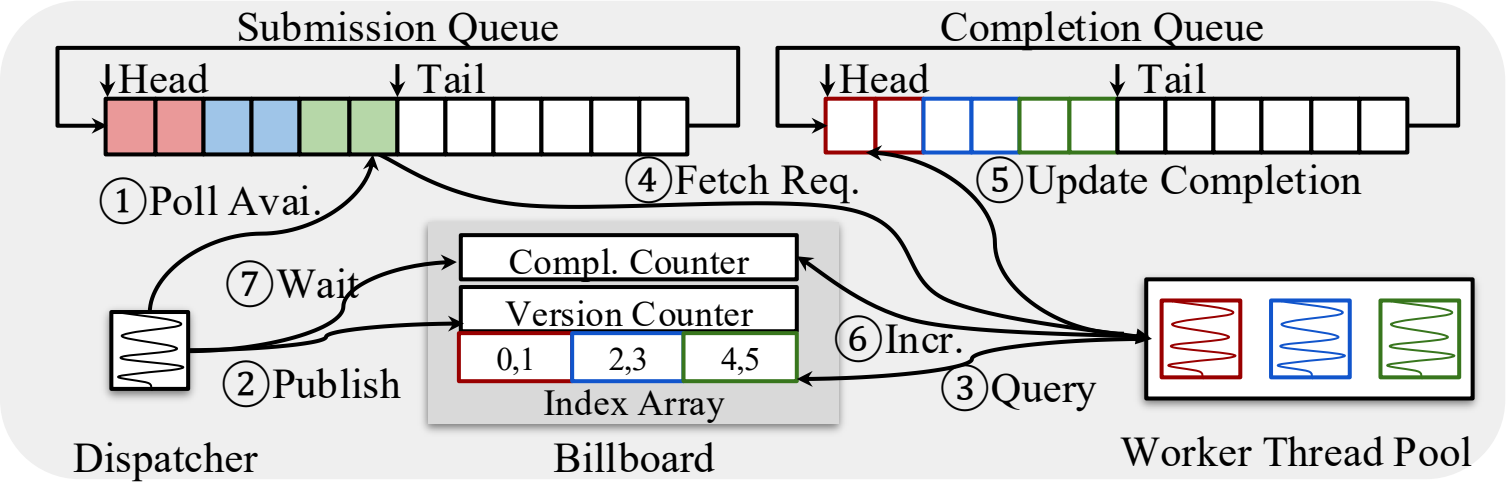
Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



Technique #1: TX-Side Elastic Queue Pair (2/2)

- **Ours: Elastic QP (E-QP)**
  - Single-Producer Multi-Consumer (SPMC)
  - Stateless Host & Lock Free



Dedicated dispatcher decouples coordination from workers

Technique #2: RX-Side Elastic Queue Pair

Technique #2: RX-Side Elastic Queue Pair

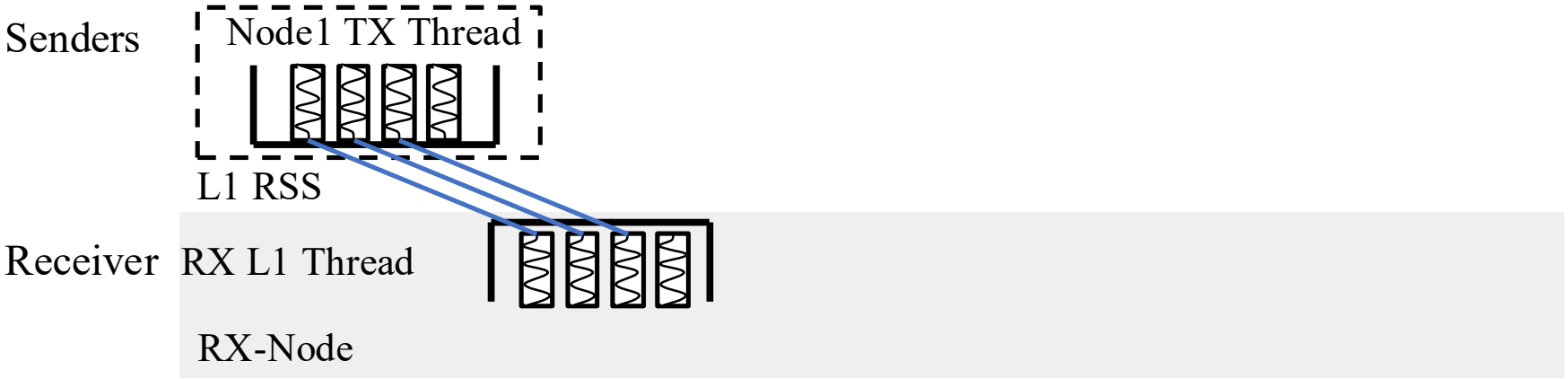
- **Difference**: TX-Side Host-to-NIC vs. RX-Side Wire-to-NIC

Technique #2: RX-Side Elastic Queue Pair

- **Difference**: TX-Side Host-to-NIC vs. RX-Side Wire-to-NIC

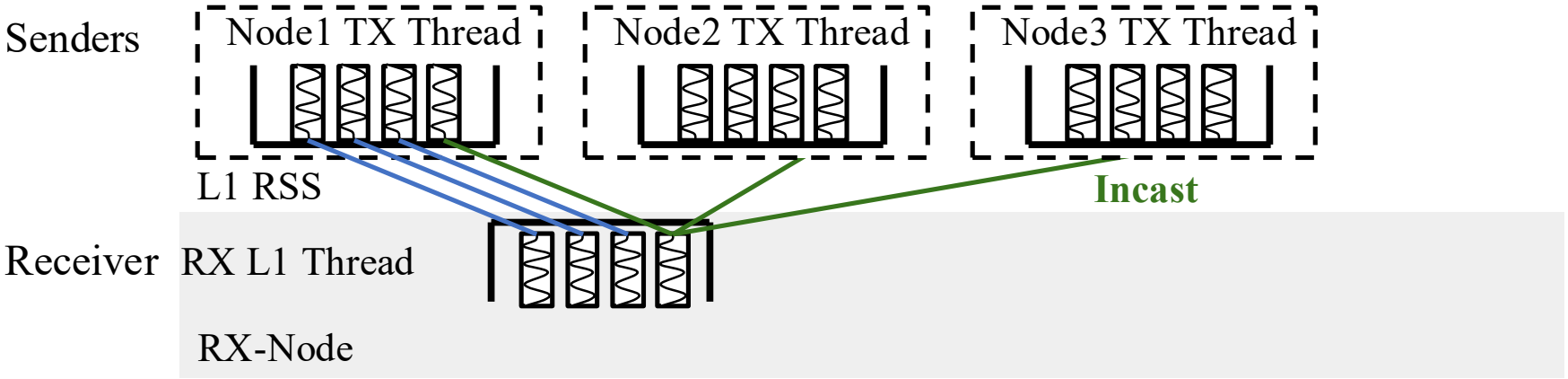
Technique #2: RX-Side Elastic Queue Pair

- **Difference**: TX-Side Host-to-NIC vs. RX-Side Wire-to-NIC
- **Level-1 Scaling (RSS):** one-to-one traffic



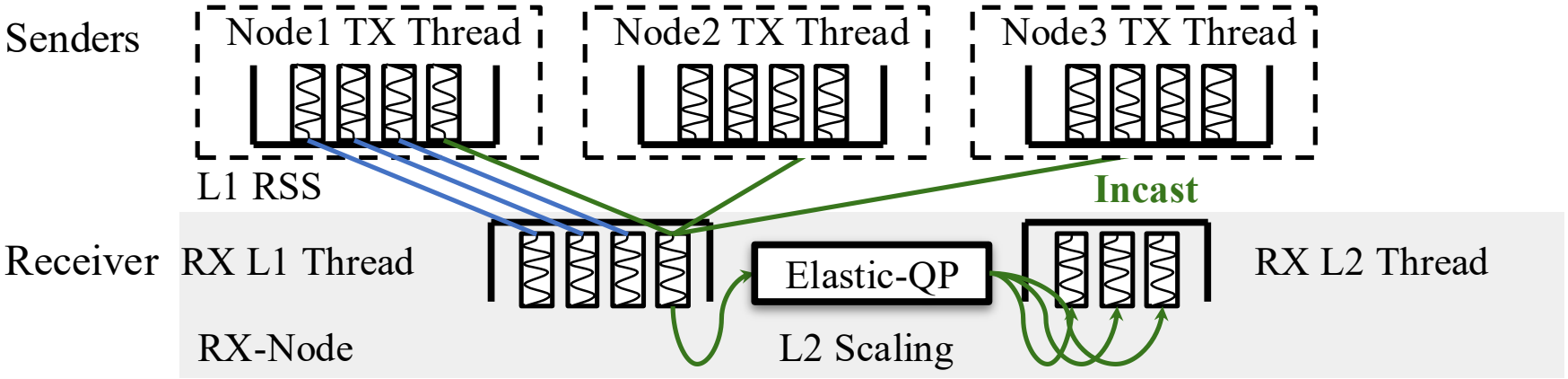
Technique #2: RX-Side Elastic Queue Pair

- **Difference**: TX-Side Host-to-NIC vs. RX-Side Wire-to-NIC
- **Level-1 Scaling (RSS):** one-to-one traffic
- **Level-2 Scaling (E-QP):** many-to-one incast



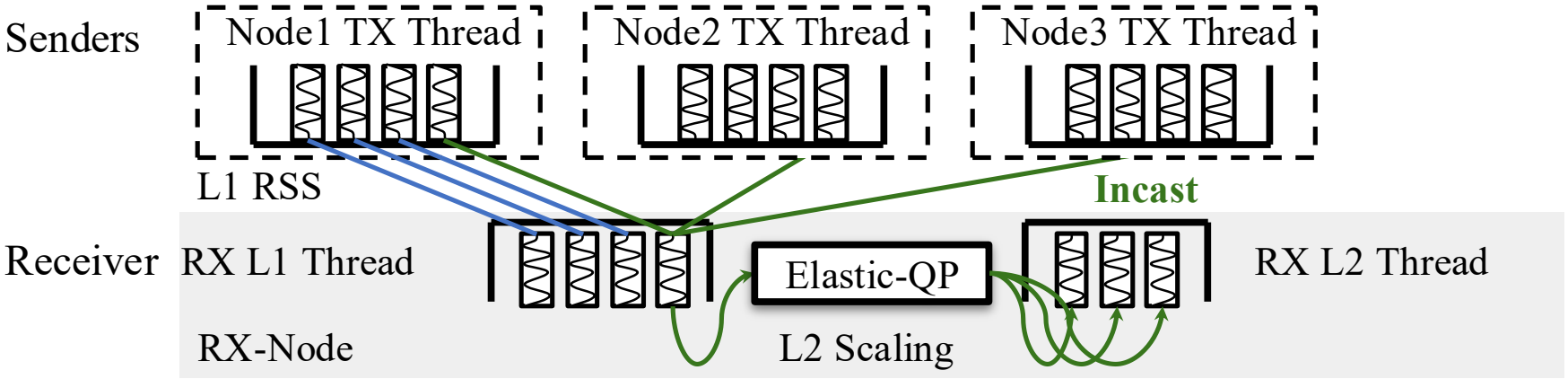
Technique #2: RX-Side Elastic Queue Pair

- **Difference**: TX-Side Host-to-NIC vs. RX-Side Wire-to-NIC
- **Level-1 Scaling (RSS):** one-to-one traffic
- **Level-2 Scaling (E-QP):** many-to-one incast



Technique #2: RX-Side Elastic Queue Pair

- **Difference**: TX-Side Host-to-NIC vs. RX-Side Wire-to-NIC
- **Level-1 Scaling (RSS):** one-to-one traffic
- **Level-2 Scaling (E-QP):** many-to-one incast

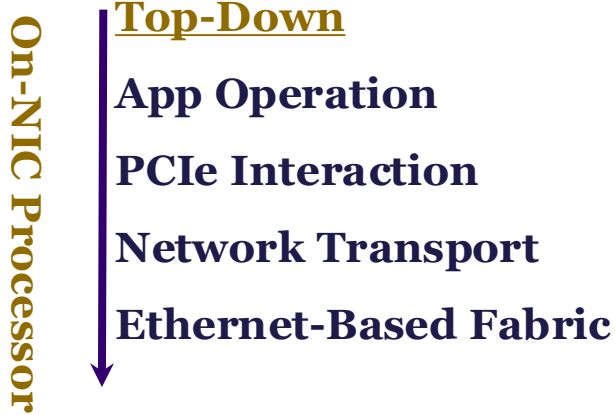


Besides TX, RX-Side E-QP can scale processing power for incast

Technique #3: Flexible Operation (FlexOp) Engine

Technique #3: Flexible Operation (FlexOp) Engine

- **Vertical Integration** Enables *Flexibility*, Cooperating with E-QP *Parallelism*



Technique #3: Flexible Operation (FlexOp) Engine

- **Vertical Integration** Enables *Flexibility*, Cooperating with E-QP *Parallelism*

On-NIC Processor

Top-Down

App Operation

PCIe Interaction

Network Transport

Ethernet-Based Fabric

Flex. Op. Set

- OpenSHMEM
- Message Queue

Technique #3: Flexible Operation (FlexOp) Engine

- **Vertical Integration** Enables *Flexibility*, Cooperating with E-QP *Parallelism*

On-NIC Processor

Top-Down

App Operation

PCIe Interaction

Network Transport

Ethernet-Based Fabric

Flex. Op. Set

- OpenSHMEM
- Message Queue

Flex PCIe I/O

- MMIO Batching
- WQE Fetching
- Payload Inlining

Technique #3: Flexible Operation (FlexOp) Engine

- **Vertical Integration** Enables *Flexibility*, Cooperating with E-QP *Parallelism*

On-NIC Processor

Top-Down

App Operation

PCIe Interaction

Network Transport

Ethernet-Based Fabric

Flex. Op. Set	**Flex PCIe I/O**	**Flex. Transport**
• OpenSHMEM	• MMIO Batching	• Reliability
• Message Queue	• WQE Fetching	• Flow Control
	• Payload Inlining	• **Packet Format**

Technique #3: Flexible Operation (FlexOp) Engine

- **Vertical Integration** Enables *Flexibility*, Cooperating with E-QP *Parallelism*

On-NIC Processor

Top-Down

App Operation

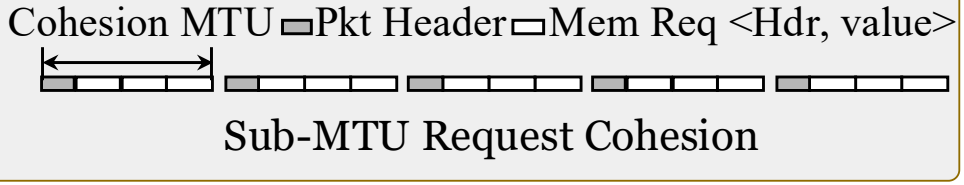
PCIe Interaction

Network Transport

Ethernet-Based Fabric

Flex. Op. Set	**Flex PCIe I/O**	**Flex. Transport**
• OpenSHMEM	• MMIO Batching	• Reliability
• Message Queue	• WQE Fetching	• Flow Control
	• Payload Inlining	• **Packet Format**

Use Case: Customizing Pkt Format



Technique #3: Flexible Operation (FlexOp) Engine

- **Vertical Integration** Enables *Flexibility*, Cooperating with E-QP *Parallelism*

On-NIC Processor

Top-Down

App Operation

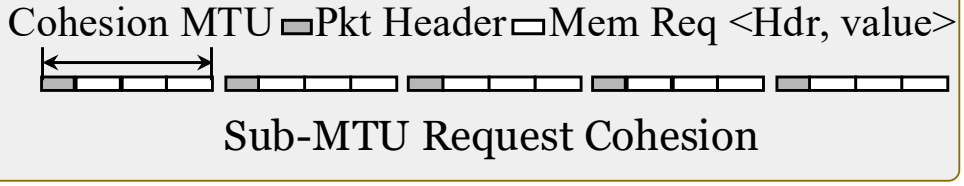
PCIe Interaction

Network Transport

Ethernet-Based Fabric

Flex. Op. Set	**Flex PCIe I/O**	**Flex. Transport**
• OpenSHMEM	• MMIO Batching	• Reliability
• Message Queue	• WQE Fetching	• Flow Control
	• Payload Inlining	• **Packet Format**

Use Case: Customizing Pkt Format



SID is open to evolving protocols and extending operations

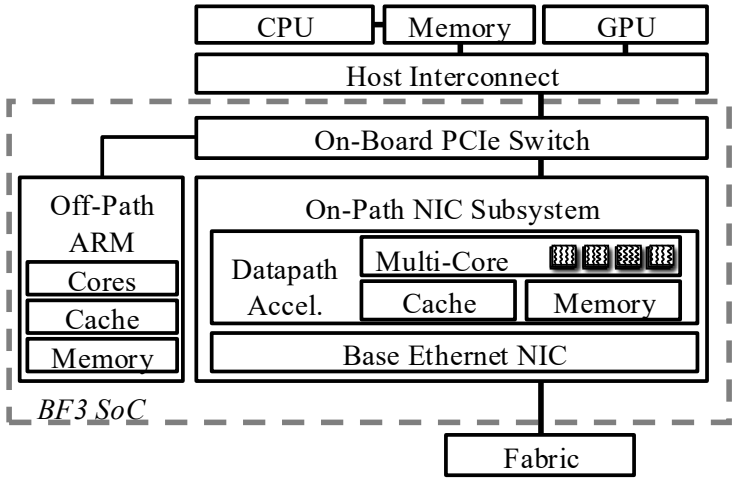
Implementation and Evaluation Setup

Implementation and Evaluation Setup

- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)

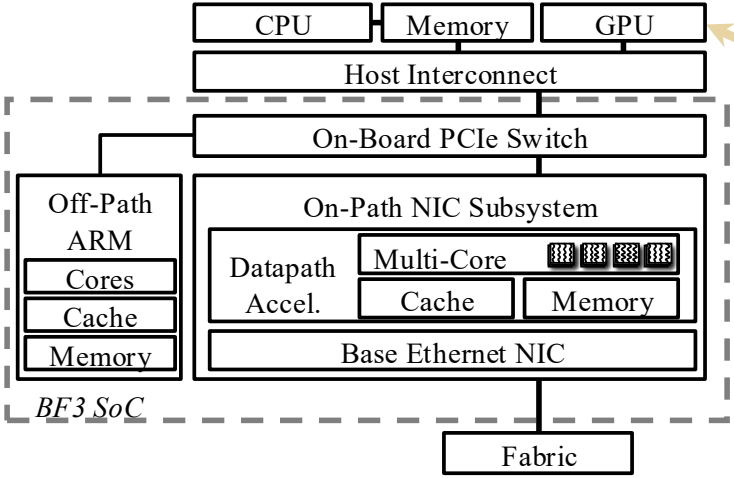
Implementation and Evaluation Setup

- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)



Implementation and Evaluation Setup

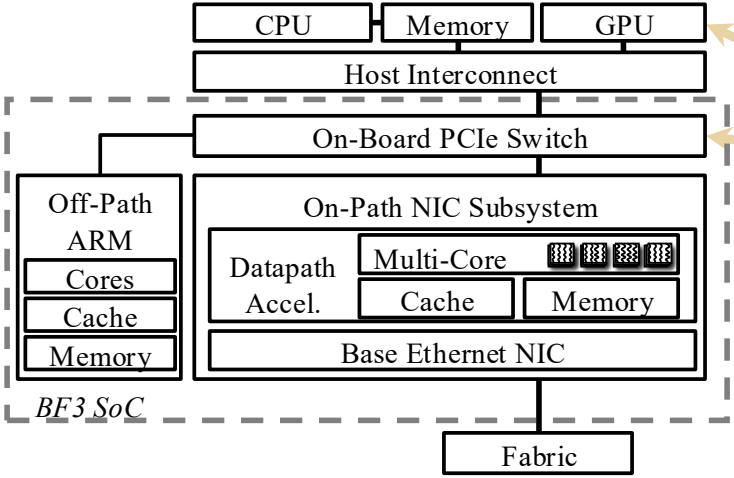
- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)



Both CPU & GPU

Implementation and Evaluation Setup

- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)

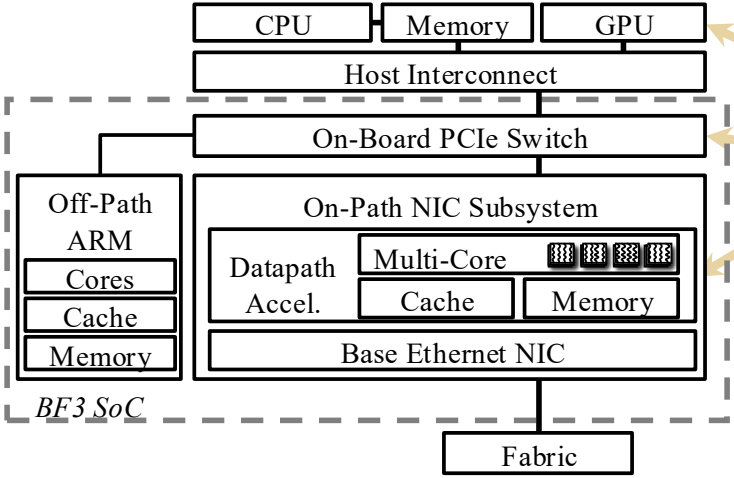


Both CPU & GPU

Memory Aperture

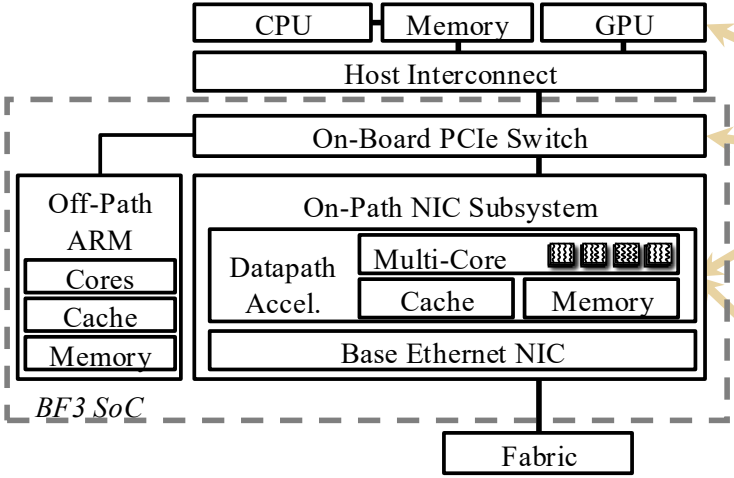
Implementation and Evaluation Setup

- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)



Implementation and Evaluation Setup

- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)



Both CPU & GPU

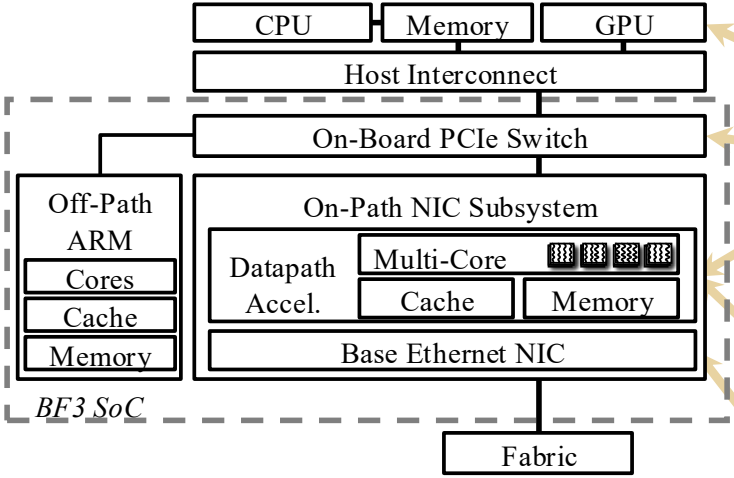
Memory Aperture

256 RISC-V Threads

C Programmability

Implementation and Evaluation Setup

- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)



Both CPU & GPU

Memory Aperture

256 RISC-V Threads

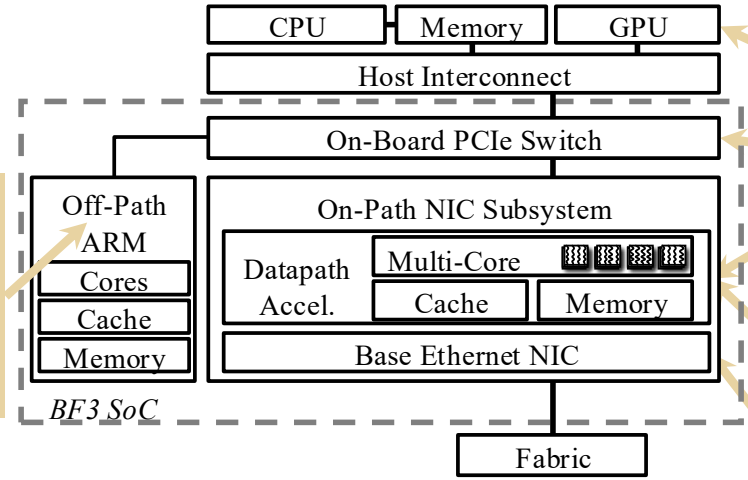
C Programmability

Ethernet SND/RCV

Implementation and Evaluation Setup

- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)

Why DPA? Not ARM
- **Availability on CX8**
- **Massive Threading**
- **Host-Mem LD/ST**



Both CPU & GPU

Memory Aperture

256 RISC-V Threads

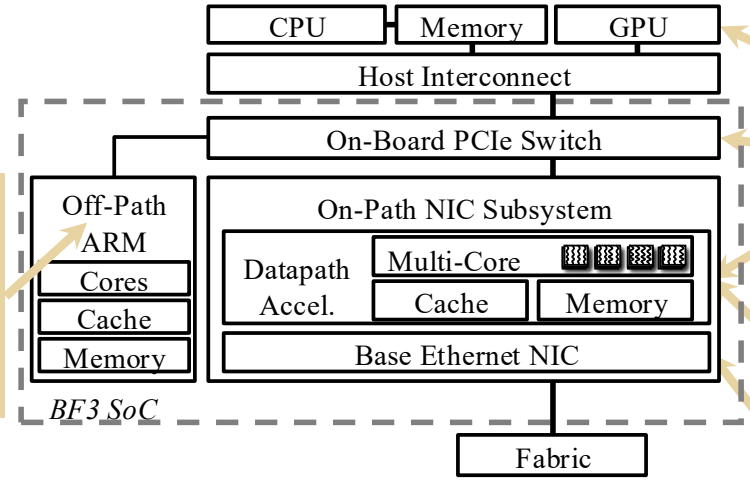
C Programmability

Ethernet SND/RCV

Implementation and Evaluation Setup

- **Implementation:** NVIDIA BlueField-3 & ConnectX-8
  - Datapath Accelerator (DPA)

Why DPA? Not ARM
- **Availability on CX8**
- **Massive Threading**
- **Host-Mem LD/ST**



Both CPU & GPU

Memory Aperture

256 RISC-V Threads

C Programmability

Ethernet SND/RCV

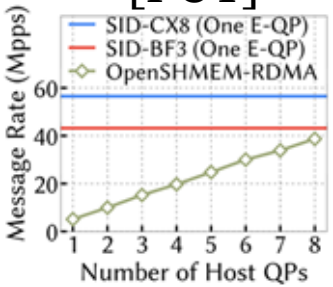
- **Evaluation Setup**
  - Baseline: RDMA-Based OpenSHMEM
  - Metric: Message Rate (Mpps) for Fine-Grained Access (64B)

Evaluation: Critical Operations and Application Workloads

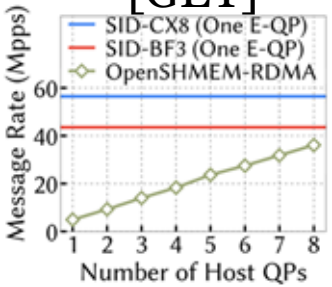
Evaluation: Critical Operations and Application Workloads

- **Critical Operations**
  - **`PUT`**: 56.50 Mpps for a single E-QP, 11.03× higher than RDMA
  - **`GET`**: 56.45 Mpps for a single E-QP, 11.49× higher than RDMA

[PUT]

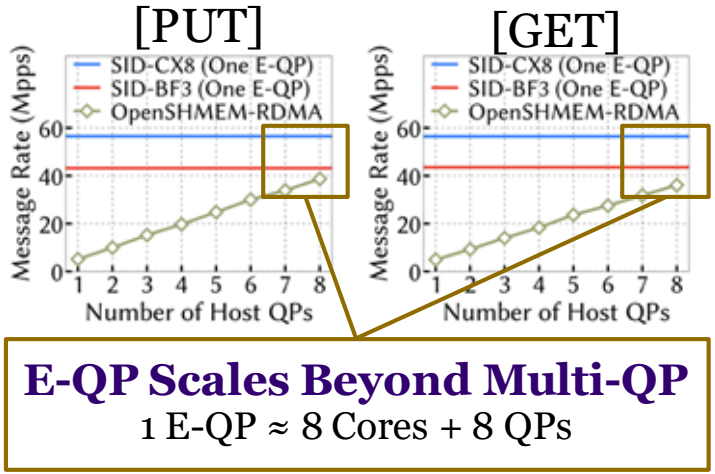


[GET]



Evaluation: Critical Operations and Application Workloads

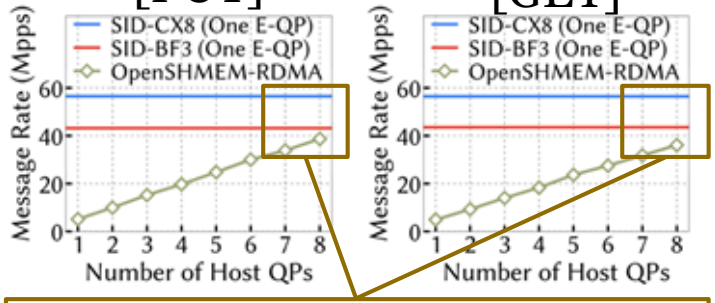
- **Critical Operations**
  - **`PUT`**: 56.50 Mpps for a single E-QP, 11.03× higher than RDMA
  - **`GET`**: 56.45 Mpps for a single E-QP, 11.49× higher than RDMA



Evaluation: Critical Operations and Application Workloads

- **Critical Operations**
  - **PUT**: 56.50 Mpps for a single E-QP, 11.03× higher than RDMA
  - **GET**: 56.45 Mpps for a single E-QP, 11.49× higher than RDMA

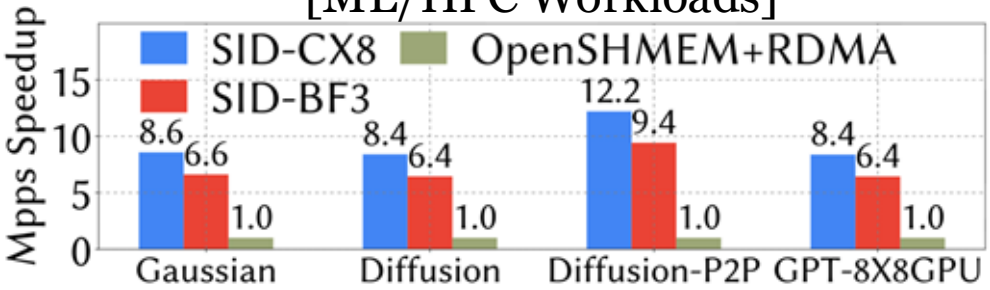
[PUT] [GET]



E-QP Scales Beyond Multi-QP

1 E-QP ≈ 8 Cores + 8 QPs

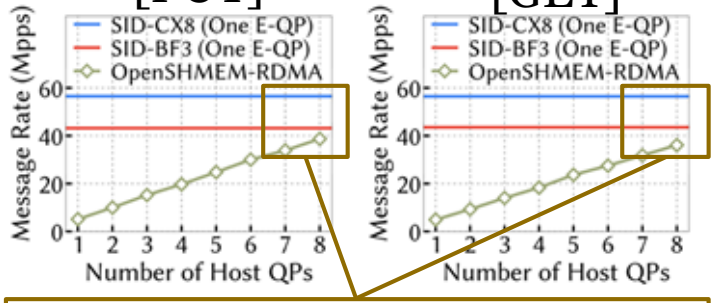
[ML/HPC Workloads]



Evaluation: Critical Operations and Application Workloads

- **Critical Operations**
  - **PUT**: 56.50 Mpps for a single E-QP, 11.03× higher than RDMA
  - **GET**: 56.45 Mpps for a single E-QP, 11.49× higher than RDMA

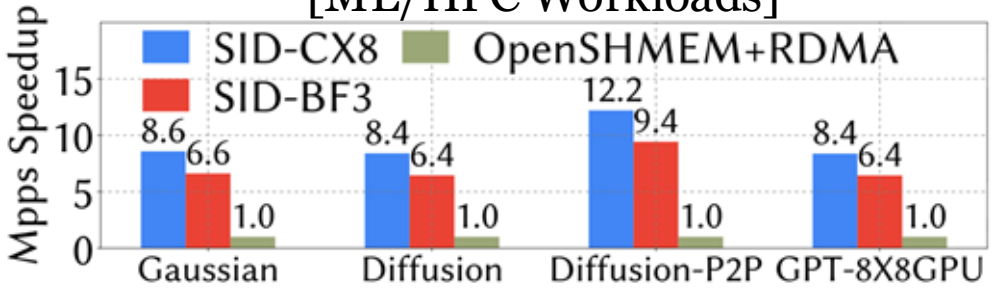
[PUT] [GET]



E-QP Scales Beyond Multi-QP

1 E-QP ≈ 8 Cores + 8 QPs

[ML/HPC Workloads]



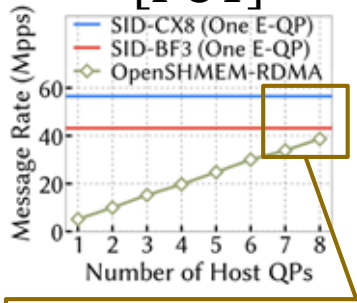
Speed Up Broad Applications

Compatible with OpenSHMEM (*e.g.*, NVSHMEM)

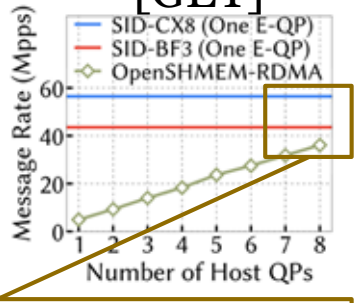
Evaluation: Critical Operations and Application Workloads

- **Critical Operations**
  - **`PUT`**: 56.50 Mpps for a single E-QP, 11.03× higher than RDMA
  - **`GET`**: 56.45 Mpps for a single E-QP, 11.49× higher than RDMA

[PUT]



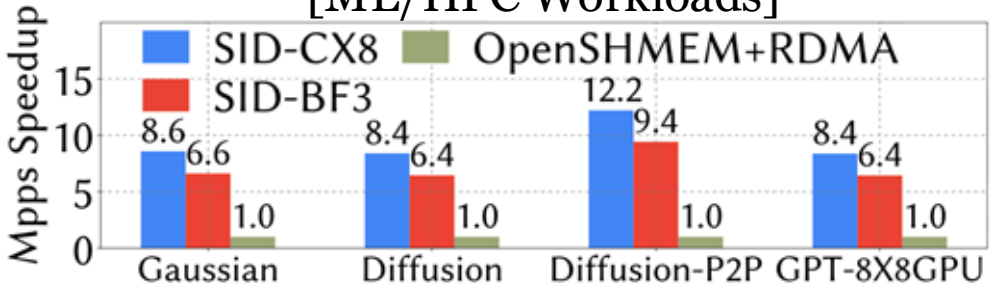
[GET]



E-QP Scales Beyond Multi-QP

1 E-QP ≈ 8 Cores + 8 QPs

[ML/HPC Workloads]



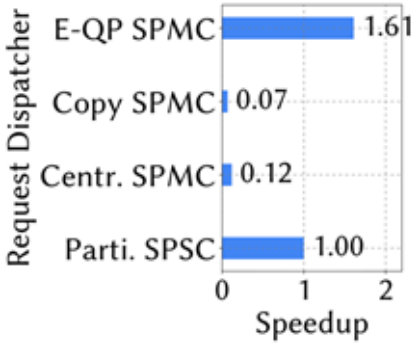
Speed Up Broad Applications

Compatible with OpenSHMEM (*e.g.*, NVSHMEM)

Please see our paper for more operations and more workloads

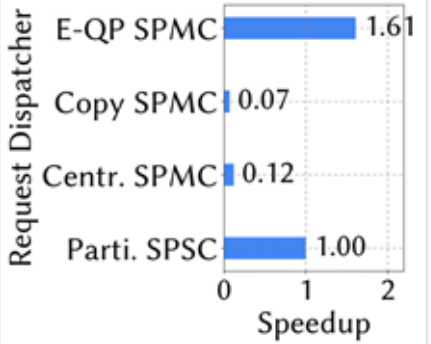
Evaluation: Effectiveness of SID Key Techniques

Evaluation: Effectiveness of SID Key Techniques

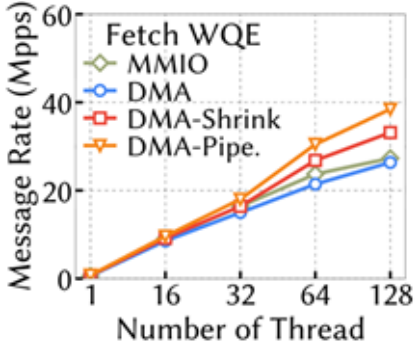


E-QP Achieves Efficient Dispatching

Evaluation: Effectiveness of SID Key Techniques

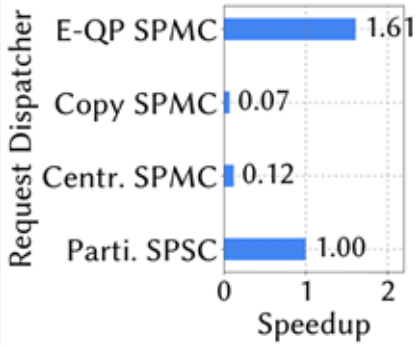


E-QP Achieves Efficient Dispatching

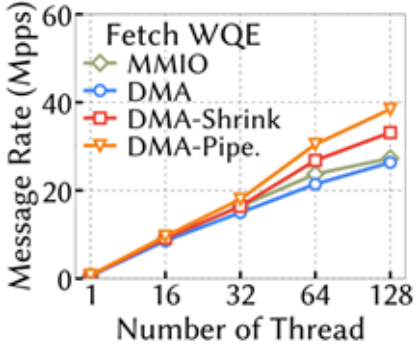


E-QP Scales to Many Processing Threads

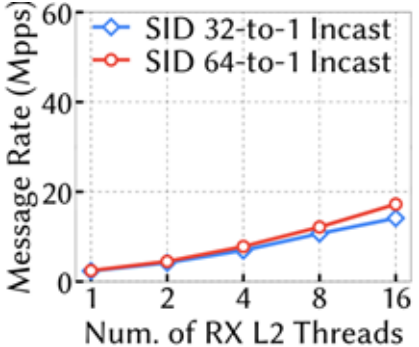
Evaluation: Effectiveness of SID Key Techniques



E-QP Achieves Efficient Dispatching

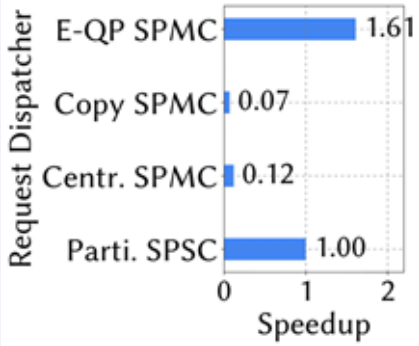


E-QP Scales to Many Processing Threads

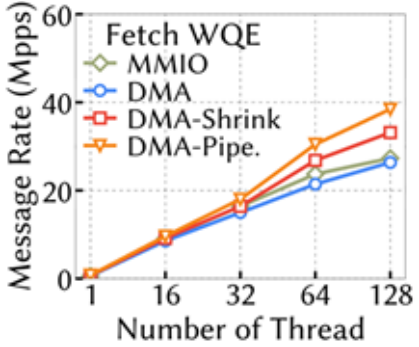


RX-Side Two-Level Scaling is Effective

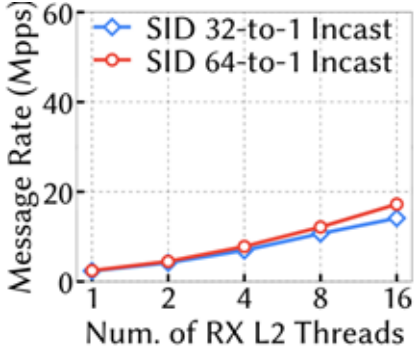
Evaluation: Effectiveness of SID Key Techniques



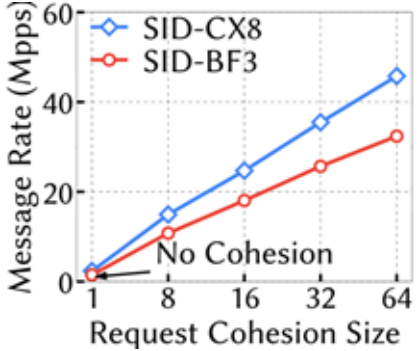
E-QP Achieves Efficient Dispatching



E-QP Scales to Many Processing Threads

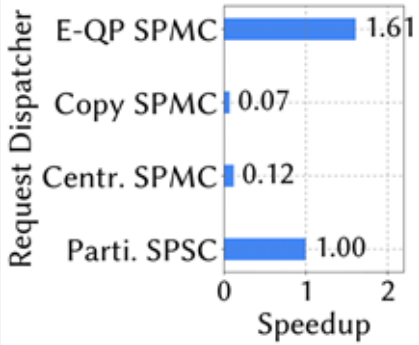


RX-Side Two-Level Scaling is Effective

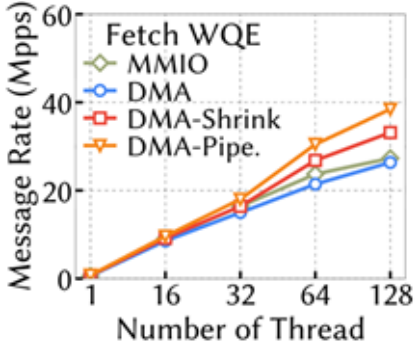


Sub-MTU Request Cohesion is Effective

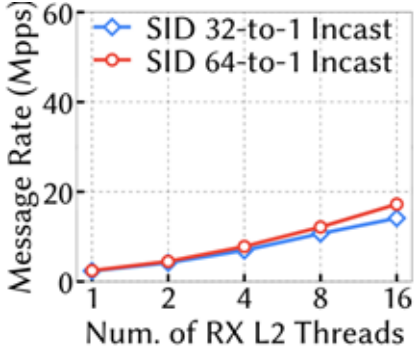
Evaluation: Effectiveness of SID Key Techniques



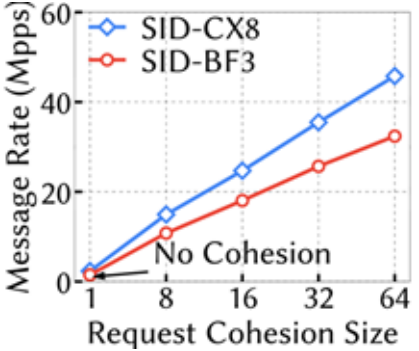
E-QP Achieves Efficient Dispatching



E-QP Scales to Many Processing Threads



RX-Side Two-Level Scaling is Effective

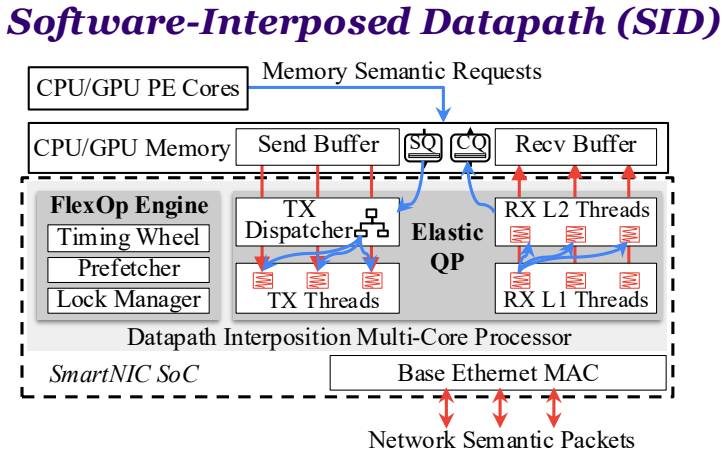


Sub-MTU Request Cohesion is Effective

Please see our paper for more experiments (*e.g.*, GPU Comm.)

Summary

Summary

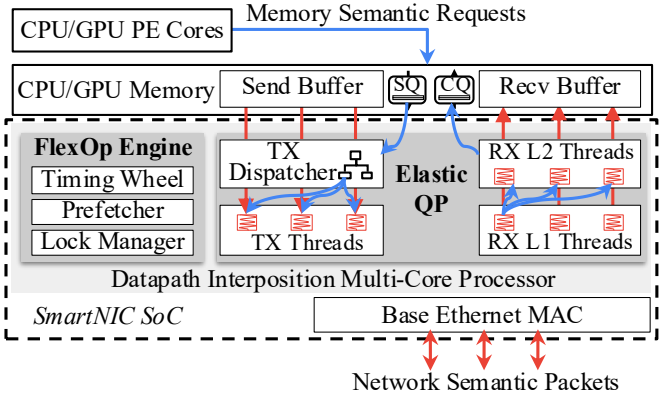

Software-Interposed Datapath (SID)
CPU/GPU PE Cores
Memory Semantic Requests
CPU/GPU Memory
Send Buffer
SQ
CQ
Recv Buffer
FlexOp Engine
Timing Wheel
Prefetcher
Lock Manager
TX Dispatcher
TX Threads
Elastic QP
RX L2 Threads
RX L1 Threads
Datapath Interposition Multi-Core Processor
SmartNIC SoC
Base Ethernet MAC
Network Semantic Packets

Summary

Fine-Grained Rack-Scale Interconnects

Efficient, software-flexible, and low-cost solution

Software-Interposed Datapath (SID)



Summary

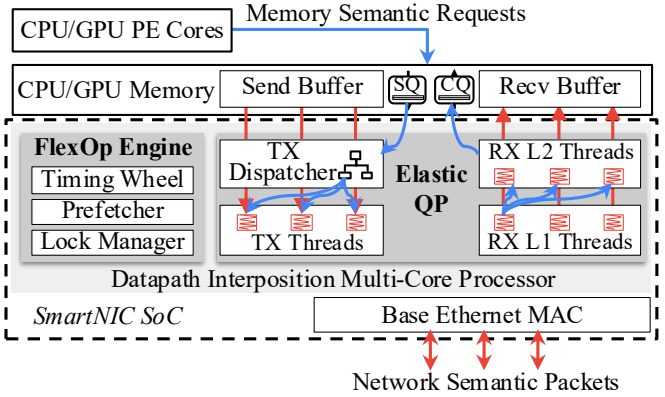
Fine-Grained Rack-Scale Interconnects

Efficient, software-flexible, and low-cost solution

On-NIC WQE-level Parallelism

Exploiting WLP via TX-Side and RX-Side E-QP

Software-Interposed Datapath (SID)



Summary

Fine-Grained Rack-Scale Interconnects

Efficient, software-flexible, and low-cost solution

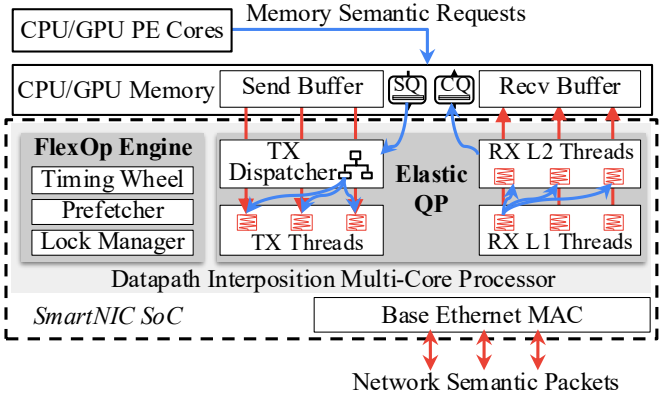
On-NIC WQE-level Parallelism

Exploiting WLP via TX-Side and RX-Side E-QP

Software-Interposed Flexibility

Provides flexibility across the full vertical stack

Software-Interposed Datapath (SID)



Summary

Fine-Grained Rack-Scale Interconnects

Efficient, software-flexible, and low-cost solution

On-NIC WQE-level Parallelism

Exploiting WLP via TX-Side and RX-Side E-QP

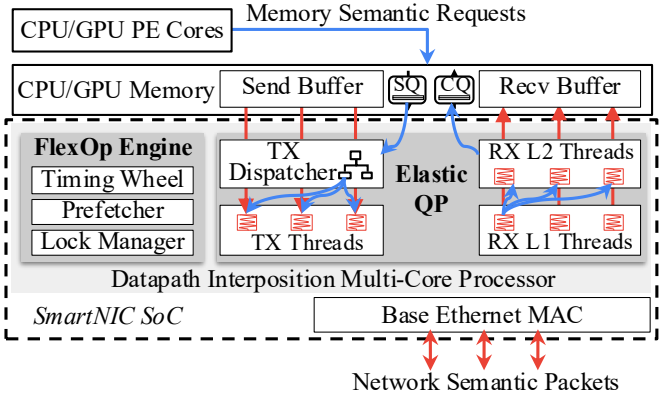
Software-Interposed Flexibility

Provides flexibility across the full vertical stack

Speed Up ML/HPC Applications

OpenSHMEM for ML; Message Queues for Cloud

Software-Interposed Datapath (SID)



Summary

Fine-Grained Rack-Scale Interconnects

Efficient, software-flexible, and low-cost solution

On-NIC WQE-level Parallelism

Exploiting WLP via TX-Side and RX-Side E-QP

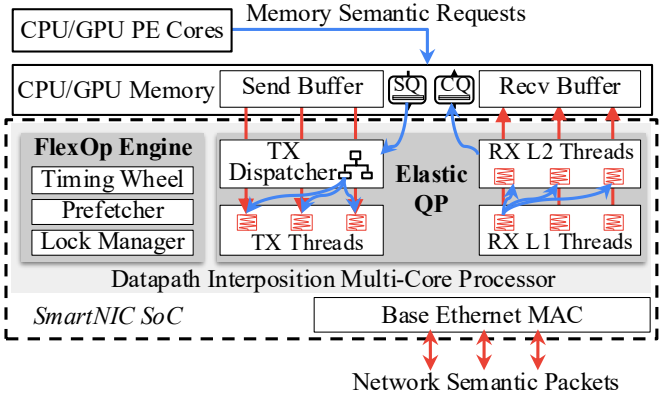
Software-Interposed Flexibility

Provides flexibility across the full vertical stack

Speed Up ML/HPC Applications

OpenSHMEM for ML; Message Queues for Cloud

Software-Interposed Datapath (SID)



Enabling WQE-Level Parallelism with On-NIC Interposition Processor

Summary

Fine-Grained Rack-Scale Interconnects

Efficient, software-flexible, and low-cost solution

On-NIC WQE-level Parallelism

Exploiting WLP via TX-Side and RX-Side E-QP

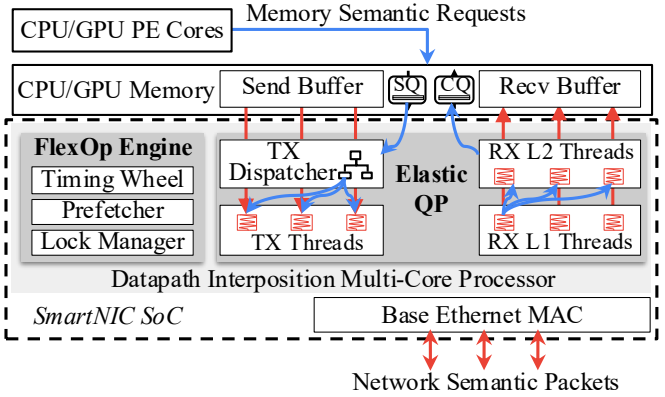
Software-Interposed Flexibility

Provides flexibility across the full vertical stack

Speed Up ML/HPC Applications

OpenSHMEM for ML; Message Queues for Cloud

Software-Interposed Datapath (SID)



Enabling WQE-Level Parallelism with On-NIC Interposition Processor

SIGCOMM 2026 DENVER

Thanks for listening!

Frank Chenxingyu Zhao (cxyzhao@cs.washington.edu)
University of Washington

UNIVERSITY of WASHINGTON

Efficient and Flexible Datapaths for Fine-Grained Rack-Scale Interconnects with Elastic QP.

https://doi.org/10.1145/3789240.3829160