ISC 2026 IXPUG Workshop

Enabling Cross-Vendor GPU Direct RDMA in UCX for Heterogeneous Systems

Buke Ao, Yihua Xu, Brian Liu, Zhongkai Zhang
Intel Corporation

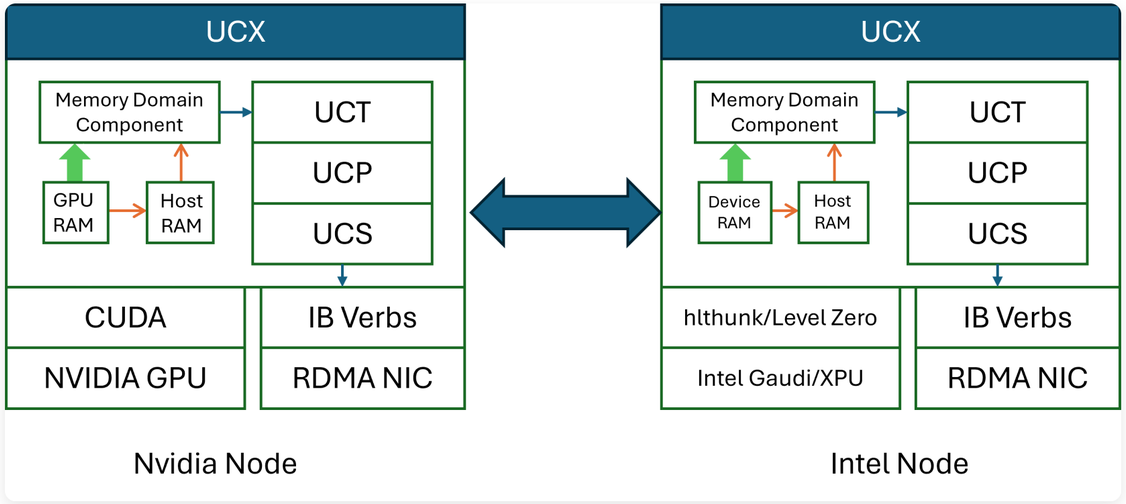
The Challenge

Heterogeneous systems with cross-vendor accelerators face a critical bottleneck

Transferring Multi-GB Data Between Devices

Cross-vendor deployments lack native GPU Direct RDMA support

- Host-mediated transfers significantly degrade performance
- Multiple copies between devices and host add latency
- CPU resources consumed unnecessarily



Real-World Impact: LLM Inference

Prefill-Decode Disaggregation

- Prefill and decode stages run on different accelerators
- Key-value cache must transfer between devices
- Cache size: several gigabytes per request for large models

Current Bottleneck

- Host staging path adds latency
- Falls back when mixing Intel Gaudi/XPU with Nvidia GPUs
- Direct device-to-device not supported

Our Solution

Extend UCX framework to enable cross-vendor GPU Direct RDMA

Memory Types

Add Intel Gaudi and XPU memory type support to UCX abstraction

Memory Domains

Implement dma-buf–based registration for device memory

Runtime Integration

Interface with SCAL and Level Zero vendor runtimes

Key benefit: Preserve UCX core logic while enabling direct device-to-device transfers

UCX Architecture Overview

UCP Layer	→	High-level communication abstractions (point-to-point, RMA, atomics)
UCT Layer	→	Transport abstraction (InfiniBand, RoCE, TCP, shared memory)
Memory Domain	→	Memory registration, type detection, dma-buf export
UCS Layer	→	Utility services (memory management, topology, config)

Runtime selection: UCX chooses appropriate transport based on buffer location and system capabilities

Key Implementation Components

Memory Domain

- Device file descriptors management
- Memory pool base address tracking
- Dma-buf descriptor maintenance
- Pointer range detection

Transport Layer

- Reuses existing IB RC transport
- No changes to transport logic
- Standard `ibv_post_send` operations
- Completion via `ibv_poll_cq`

> **Design Principle:** Extend UCX memory abstraction without modifying transport mechanisms — preserves compatibility and reduces maintenance burden

Technical Implementation

Intel Gaudi Integration

- New memory type: `UCS_MEMORY_TYPE_GAUDI`
- Dma-buf export via `hl-thunk`
- Memory pool queries via `libscal`
- Registration: `ibv_reg_dmabuf_mr`

Intel XPU Integration

- Leverage existing Level Zero support
- Dma-buf-based registration
- Runtime: `libze_loader`
- Minimal changes required

InfiniBand Verbs Integration: `ibv_reg_mr` for host memory, `ibv_reg_dmabuf_mr` for device memory enables zero-copy data transfers

Experimental Setup

Nvidia Node

2× Intel Xeon Platinum 8380

8× Nvidia A100

2× ConnectX-5 NICs (100 Gbps, RoCE)

Intel Gaudi Node

2× Intel Xeon Platinum 8480+

8× Intel Gaudi 3 (128 GB HBM)

2× ConnectX-6 Dx (100 Gbps, RoCE)

Intel XPU Node

2× Intel Xeon Sapphire Rapids

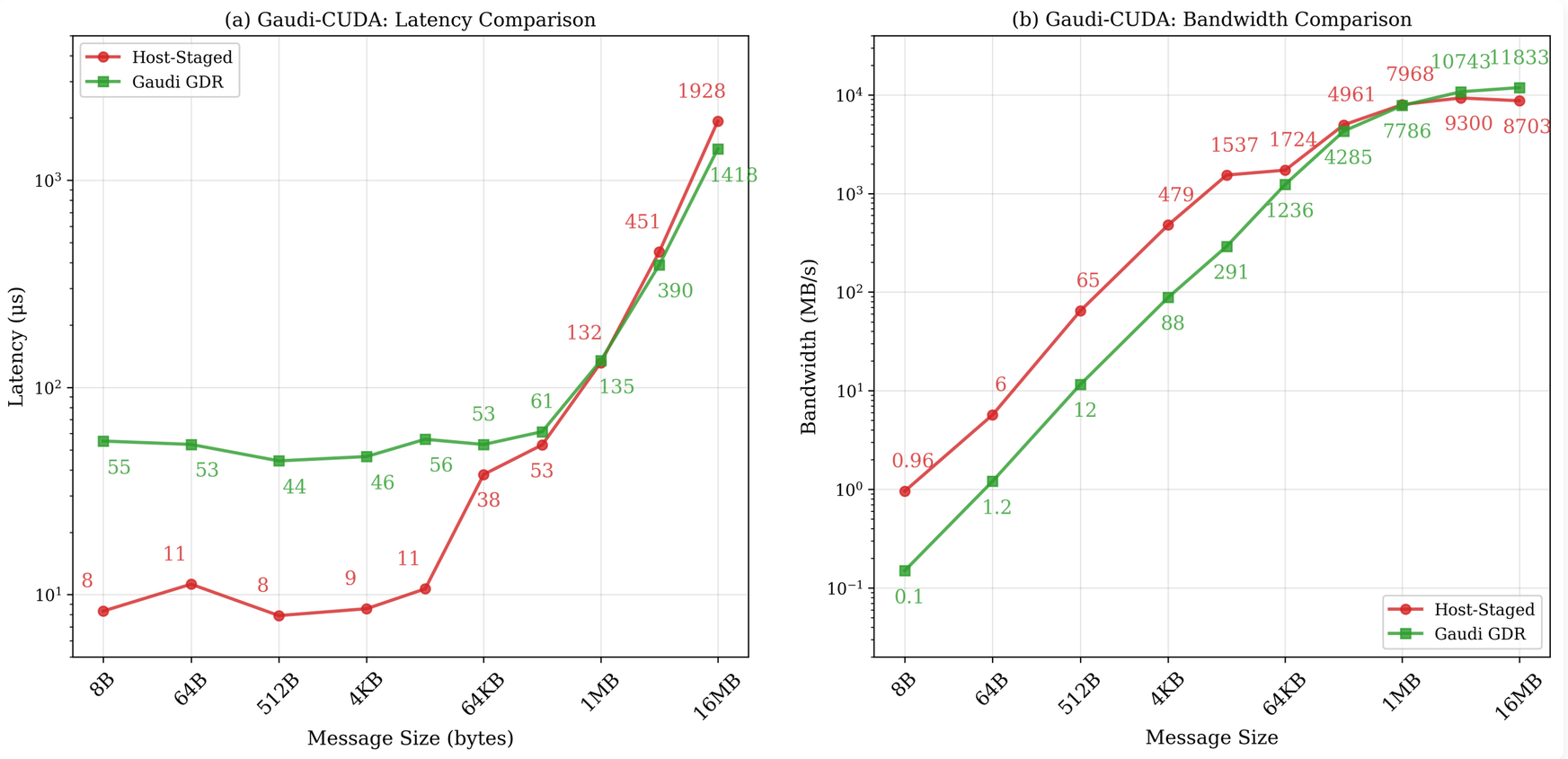
2× Intel Arc B580

1× ConnectX-5 (100 Gbps, RoCE)

Network: 100 Gigabit Ethernet with RoCE support

Software: UCX 1.22 with Intel Gaudi/XPU patches, Synapse AI 1.18, CUDA 13.0, Level Zero 1.26.2

Performance: Gaudi-CUDA



Latency Behavior

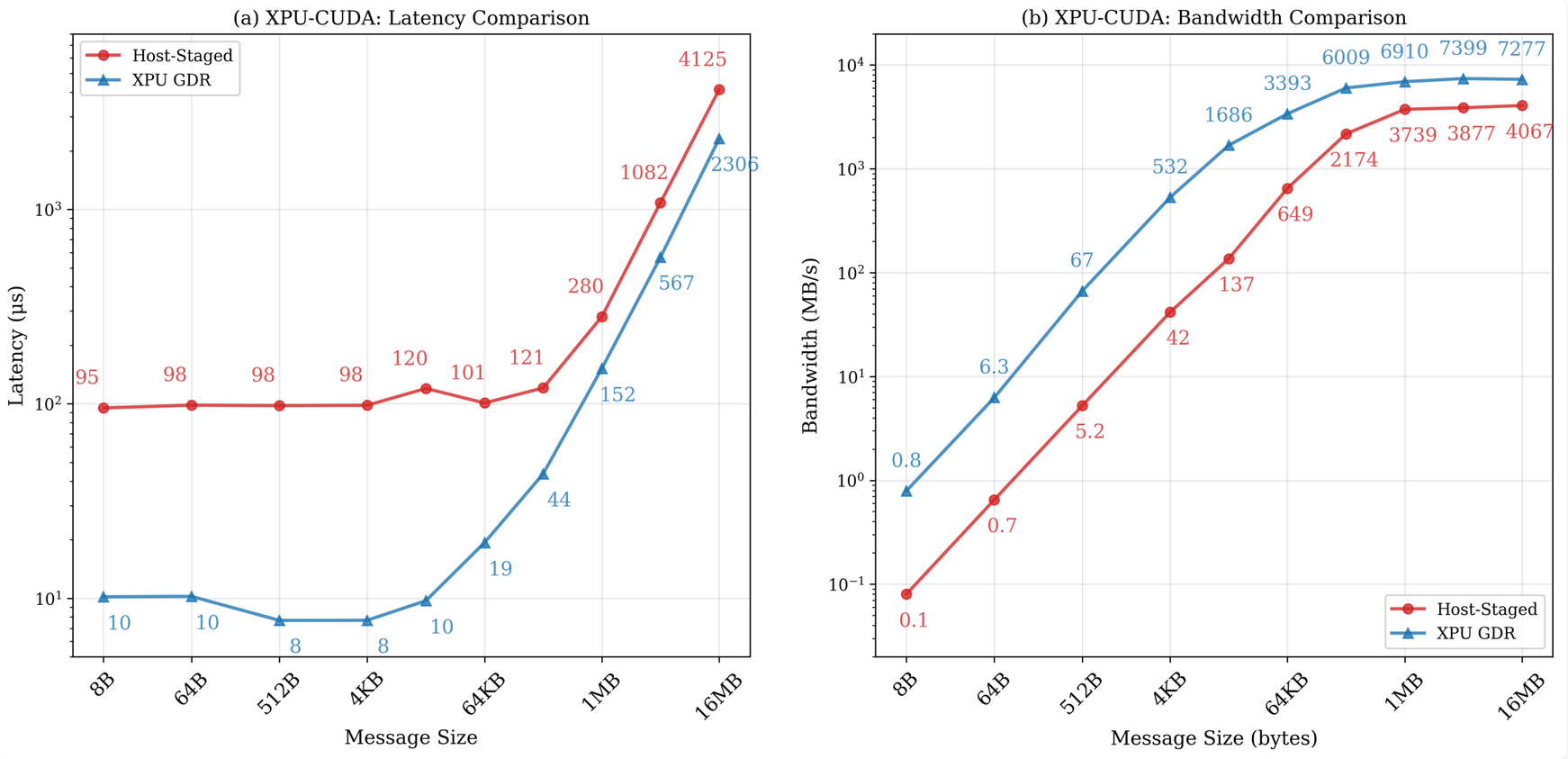
- Small messages (≤16 KB): host staging lower (8–11 µs vs 44–55 µs)
- Large transfers (≥1 MB): GDR becomes faster as overhead is amortized

Bandwidth Results

- Host staging saturates at 8.7–9.3 GB/s
- GDR reaches **11.8 GB/s** at 16 MB (near hardware limit)
- GDR optimal for bandwidth-dominated workloads

Note: Goal is not to reach peak performance, but to demonstrate GDR works better than CPU-staged transfers.

Performance: XPU-CUDA



Consistent GDR Advantage

- Small messages (≤16 KB): ~10 µs vs ~100 µs latency
- Large transfers (16 MB): 2.3 ms vs 4.1 ms (44% reduction)
- GDR outperforms across all message sizes

Bandwidth Gains

- GDR reaches **7.4 GB/s**
- CPU staging limited to ~4.0 GB/s
- PCIe Gen3 ×8 bottleneck (7.88 GB/s theoretical)

Note: Goal is not to reach peak performance, but to demonstrate GDR works better than CPU-staged transfers.

Contributions & Impact

Open Source

PR merged into UCX project for reproducibility and community benefit

Compatibility

No modifications required on Nvidia nodes — full backward compatibility

Performance

Near-hardware bandwidth limits achieved across all device combinations

Enables Heterogeneous Deployments

- Mix Intel Gaudi, Intel XPU, and Nvidia GPUs in same cluster
- Seamless integration with HPC communication frameworks
- Eliminates host staging overhead
- Practical for LLM prefill-decode disaggregation

Future Directions

Topology Optimization

Topology-aware device-to-NIC mapping for multi-socket systems to minimize cross-socket traffic

Large-Scale Evaluation

Distributed LLM training and inference workloads on production clusters

PD Disaggregation Systems

Apply cross-vendor GDR to prefill-decode disaggregation architectures for real-world LLM serving systems

Thank You

Key Takeaways

- Cross-vendor GDR is practical and performant
- UCX extensibility enables heterogeneous systems
- Up to 35% bandwidth improvement vs host staging

Resources

GitHub: UCX open-source project

Contact: aobuke@hotmail.com

Intel Gaudi SDK & Level Zero