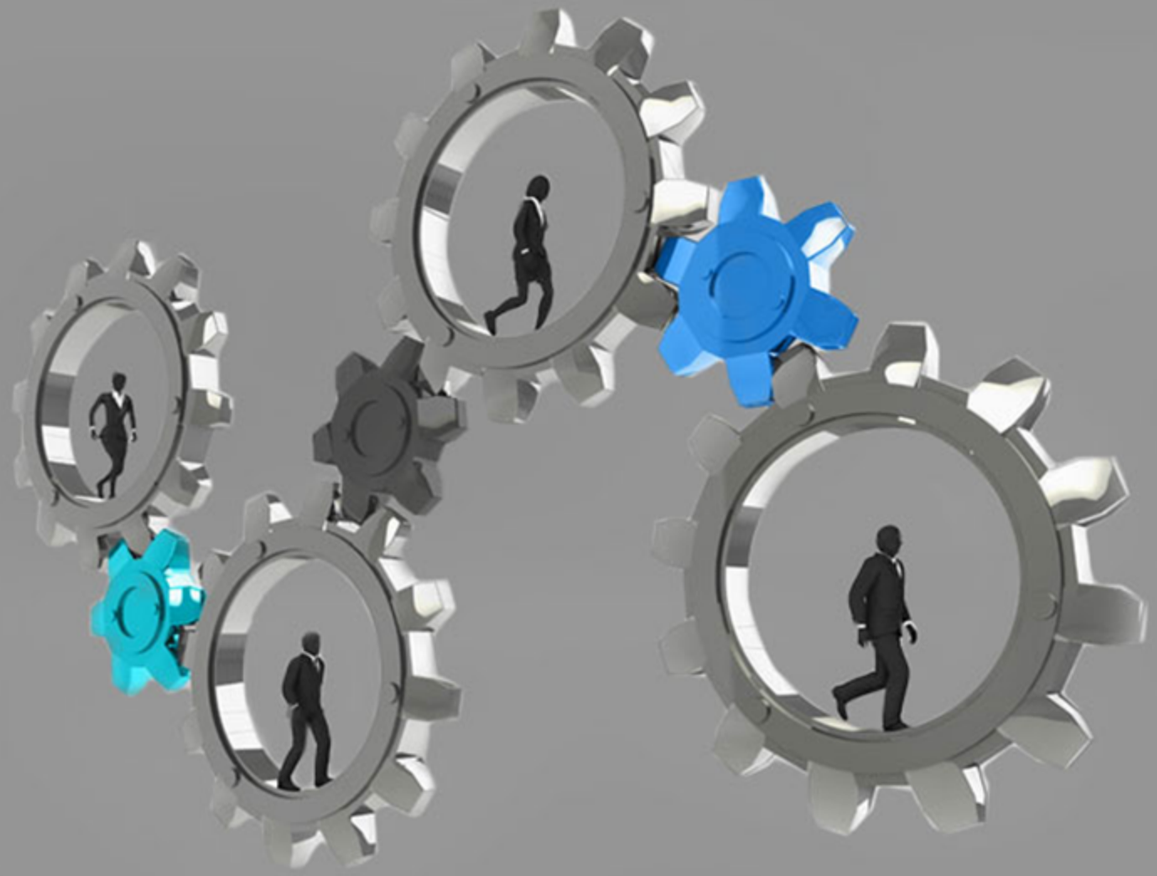
ENABLER OF CO-DESIGN

UCF



Unified Communication X (UCX)

Pavel Shamis / Pasha

ARM Research

SC'18

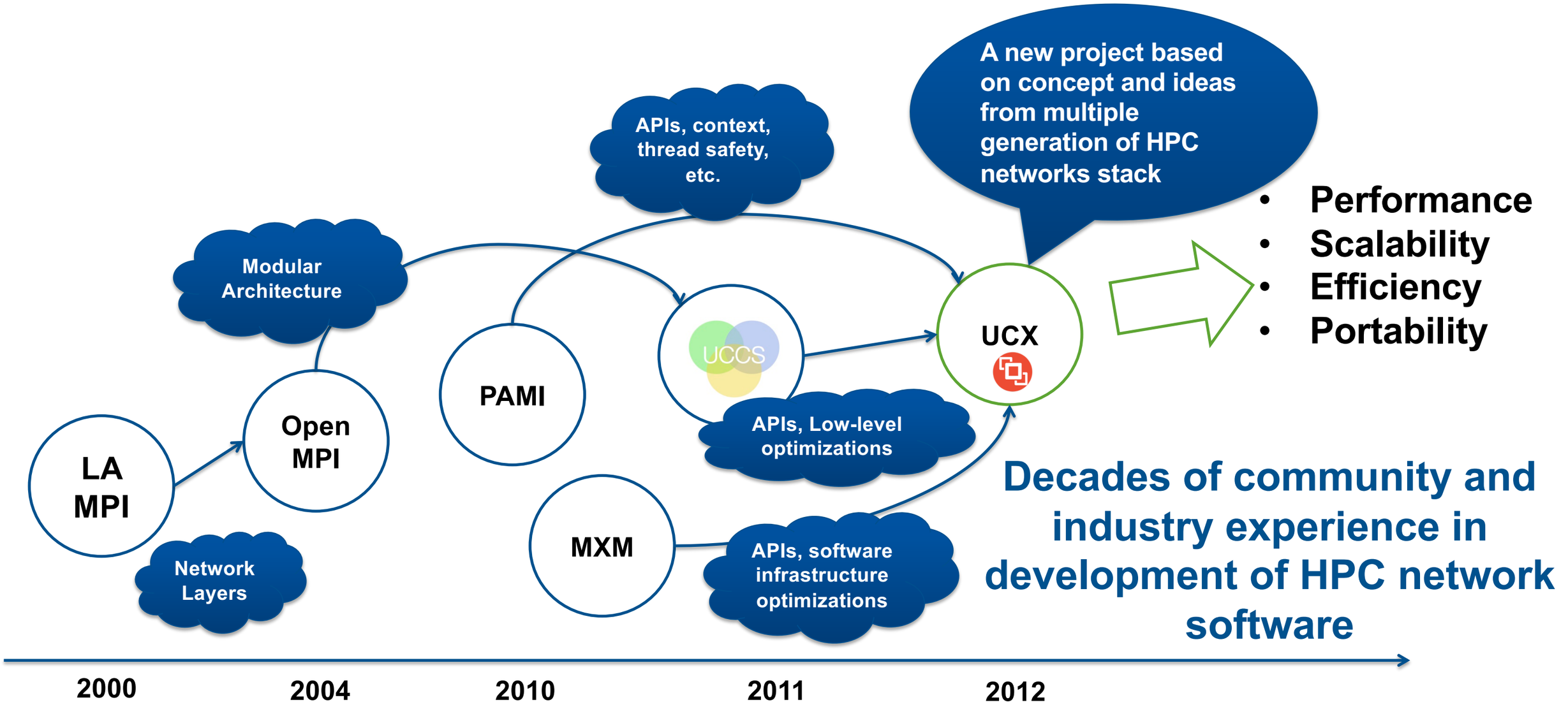
UCF Consortium

- Mission:
  - Collaboration between industry, laboratories, and academia to create production grade communication frameworks and open standards for data centric and high-performance applications

- Projects
  - UCX – Unified Communication X
  - Open RDMA

- Board members
  - **Jeff Kuehn**, UCF Chairman (Los Alamos National Laboratory)
  - **Gilad Shainer**, UCF President (Mellanox Technologies)
  - **Pavel Shamis**, UCF treasurer (ARM)
  - **Brad Benton**, Board Member (AMD)
  - **Duncan Poole**, Board Member (Nvidia)
  - **Pavan Balaji**, Board Member (Argonne National Laboratory)
  - **Sameh Sharkawi**, Board Member (IBM)
  - **Dhabaleswar K. (DK) Panda**, Board Member (Ohio State University)
  - **Steve Poole**, Board Member (Open Source Software Solutions)

UCX - History



UCX Framework Mission

- Collaboration between industry, laboratories, government (DoD, DoE), and academia
- Create open-source production grade communication framework for HPC applications
- Enable the highest performance through co-design of software-hardware interfaces

API

Exposes broad semantics that target data centric and HPC programming models and applications

Performance oriented

Optimization for low-software overheads in communication path allows near native-level performance

Production quality

Developed, maintained, tested, and used by industry and researcher community

Community driven

Collaboration between industry, laboratories, and academia

Research

The framework concepts and ideas are driven by research in academia, laboratories, and industry

Cross platform

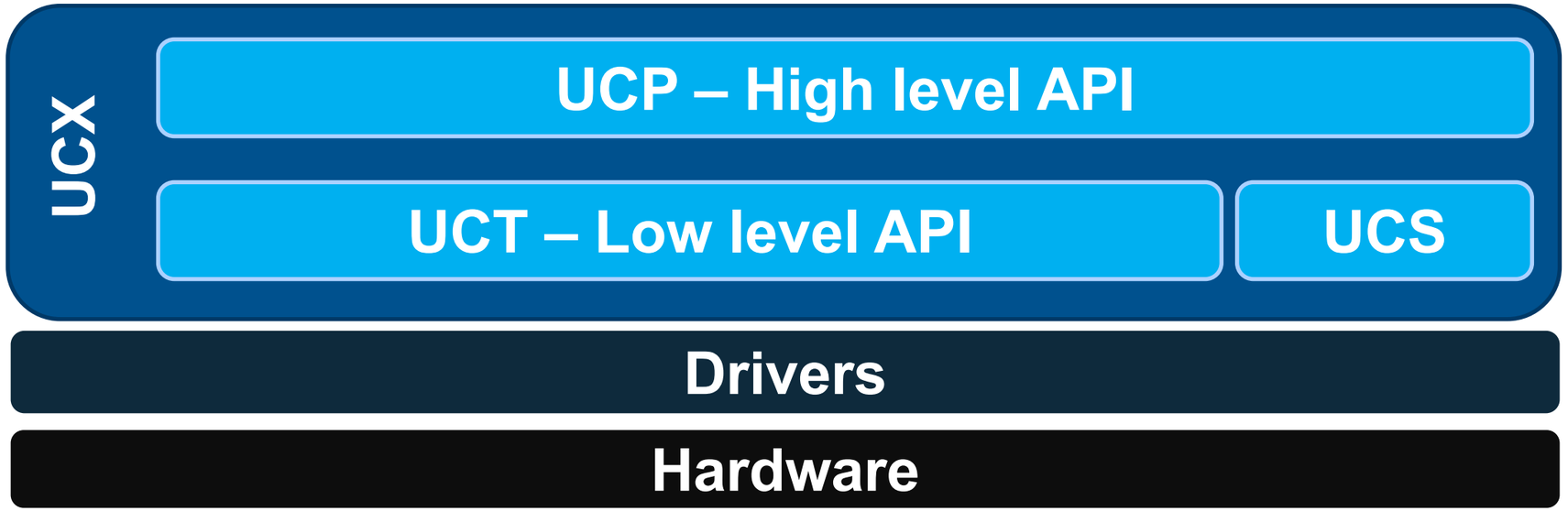
Support for Infiniband, Cray, various shared memory (x86-64, Power, ARMv8), GPUs

Co-design of Exascale Network APIs

UCX Framework

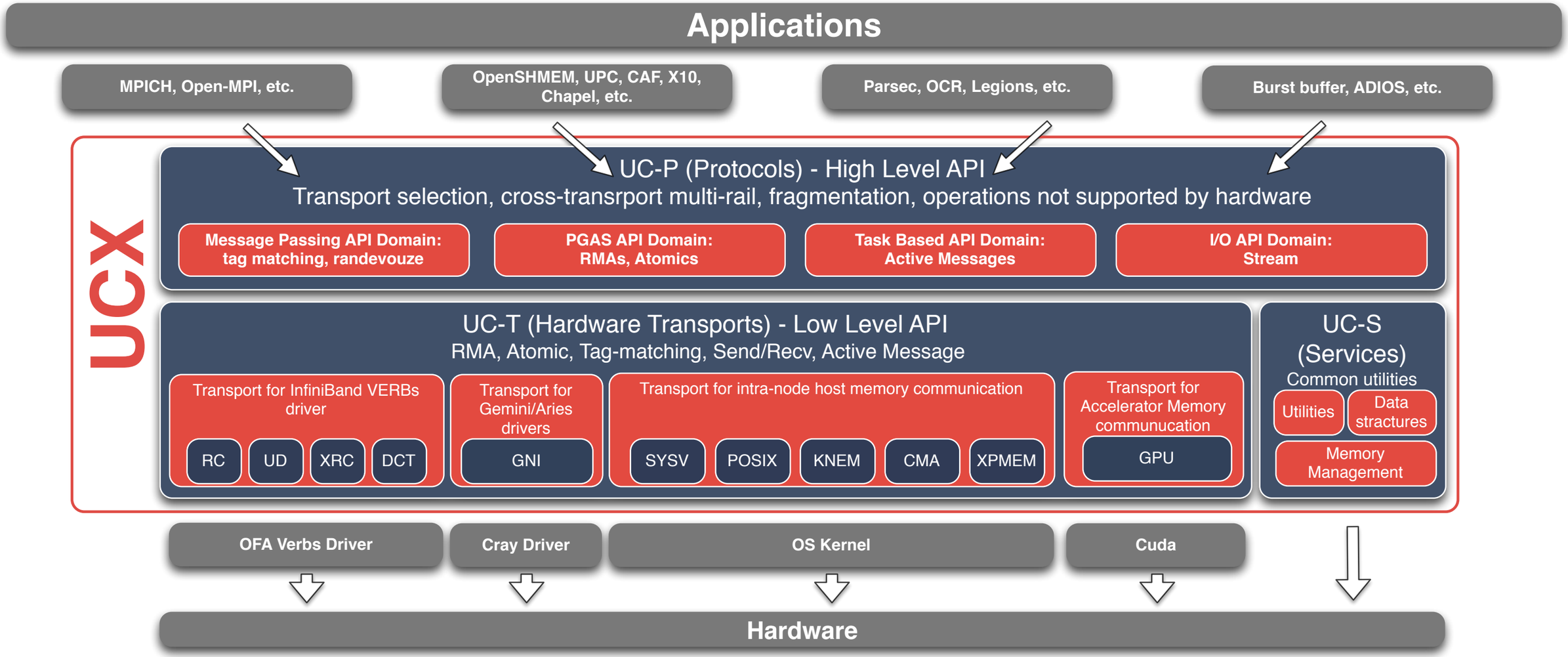
- UCX is a framework for network APIs and stacks
- UCX aims to unify the different network APIs, protocols and implementations into a single framework that is portable, efficient and functional
- UCX doesn’t focus on supporting a single programming model, instead it provides APIs and protocols that can be used to tailor the functionalities of a particular programming model efficiently
- When different programming paradigms and applications use UCX to implement their functionality, it increases their portability. As just implementing a small set of UCX APIs on top of a new hardware ensures that these applications can run seamlessly without having to implement it themselves

UCX Architecture



- **UCX framework** is composed of **three main components**.
- **UCP** layer is the protocol layer and supports all the functionalities exposed by the high-level APIs, meaning it emulates the features that are not implemented in the underlying hardware
- **UCT** layer is the transport layer that aims to provides a very efficient and low overhead access to the hardware resources
- **UCS** is a service layer that provides common data structures, memory management tools and other utilities

UCX High-level Overview



UCX Releases in 2018

- v1.3.1 - https://github.com/openucx/ucx/releases/tag/v1.3.1
  - **Multi-rail** support for eager and rendezvous protocols
  - Added **stream-based communication** API
  - Added support for **GPU** platforms: Nvidia CUDA and AMD ROCM software stacks
  - Added API for **Client-Server** based connection establishment
  - Added support for **TCP** transport (Send/Receive semantics)
  - Support for InfiniBand **hardware tag-matching** for DC and accelerated transports
  - Added support for **tag-matching communications with CUDA** buffers
  - Initial support for **Java bindings**
  - **Progress engine** optimizations
  - Improved scalability of **software tag-matching** by using a hash table
  - Added transparent **huge-pages** allocator
  - Added **non-blocking flush and disconnect** semantics
  - Added registration cache for **KNEM**
  - Support fixed-address memory allocation via ucp_mem_map()
  - Added ucp_tag_send_nbr() API to avoid send request allocation
  - Support global addressing in all IB transports
  - Add support for external epoll fd and edge-triggered events
  - Added ucp_rkey_ptr() to obtain pointer for shared memory region

UCX Releases in 2018 - continued

- V1.4.0 - https://github.com/openucx/ucx/releases/tag/v1.4.0
  - Support for installation with latest **AMD ROCm**
  - Support for latest **RDMA-CORE**
  - Support for **NVIDIA CUDA IPC** for intra-node GPU
  - Support for **NVIDIA CUDA** memory allocation cache for mem-type detection
  - Support for latest **Mellanox devices (200Gb/s)**
  - Support for **NVIDIA GPU managed memory**
  - Support for **bitwise** (OpenSHMEM v1.4) atomics operations

X86, Power8/9, arm

State-of-the-art support for GP-GPU

InfiniBand, RoCEv1/v2, Gemini/Aries, Shared Memory, TCP/Ethernet (Beta)

UCX Releases in 2018 - continued

- V1.5.0 – End of November: ADD branch here
  - New **emulation** mode enabling comprehensive UCX functionality (Atomic, Put, Get, etc) over TCP and legacy interconnects that don't implement full RDMA semantics.
  - New **non-blocking API** for all one-sided operations.
  - New **client/server connection establishment** API
  - New advanced **statistic** capabilities (tag matching queues)

UCX Roadmap

- v1.6 – Q1 2018
  - Bugfixes and optimizations
  - IWARP
  - Active Message API
- v2.0 –Q3-Q4 2019
  - Updated API – not backward compatible with 1.x
    - Cleanup (remove deprecated APIs)
    - UCP request object redesign – improves future backward compatibility
  - Binary distribution will provide v1.x version of the library (in addition for 2.x) for backward compatibility
    - All codes should work as it is

Integrations

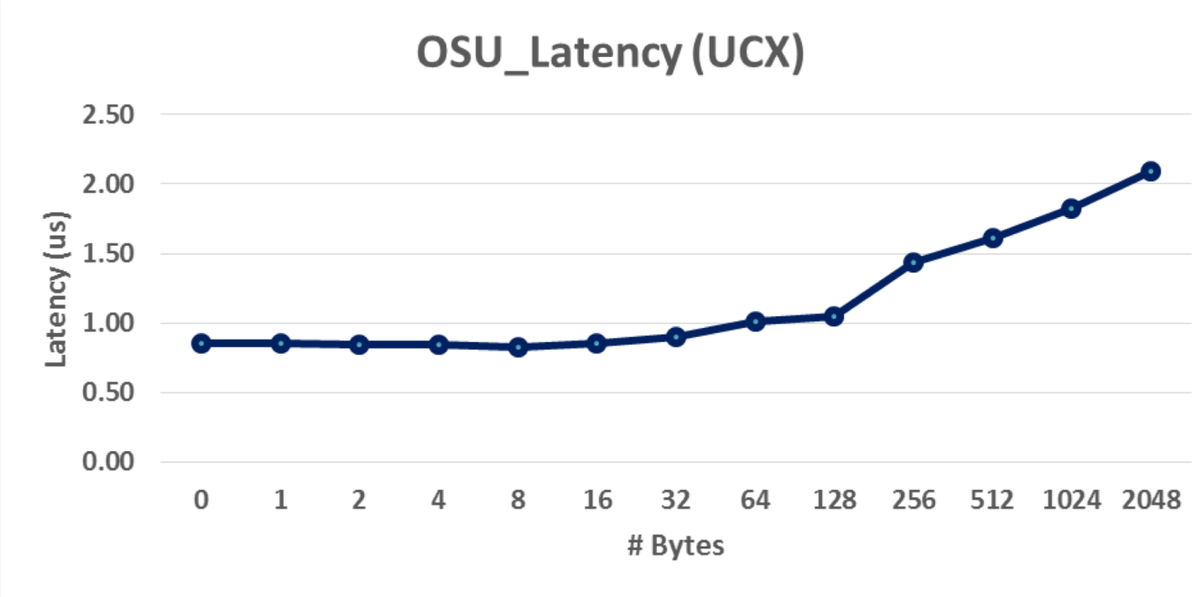
- Open MPI and OSHMEM
  - UCX replaces OpenIB BTL as default transport for InfiniBand and RoCE
  - New UCX BTL (by LANL)
- MPICH MPI
  - CH4 UCX
- OSSS SHMEM by StonyBrook and LANL
- Open SHMEM-X by ORNL
- Parsec (UTK)
- Intel Libfabrics/OFI
  - Powered by UCX !
- 3rd party commercial projects

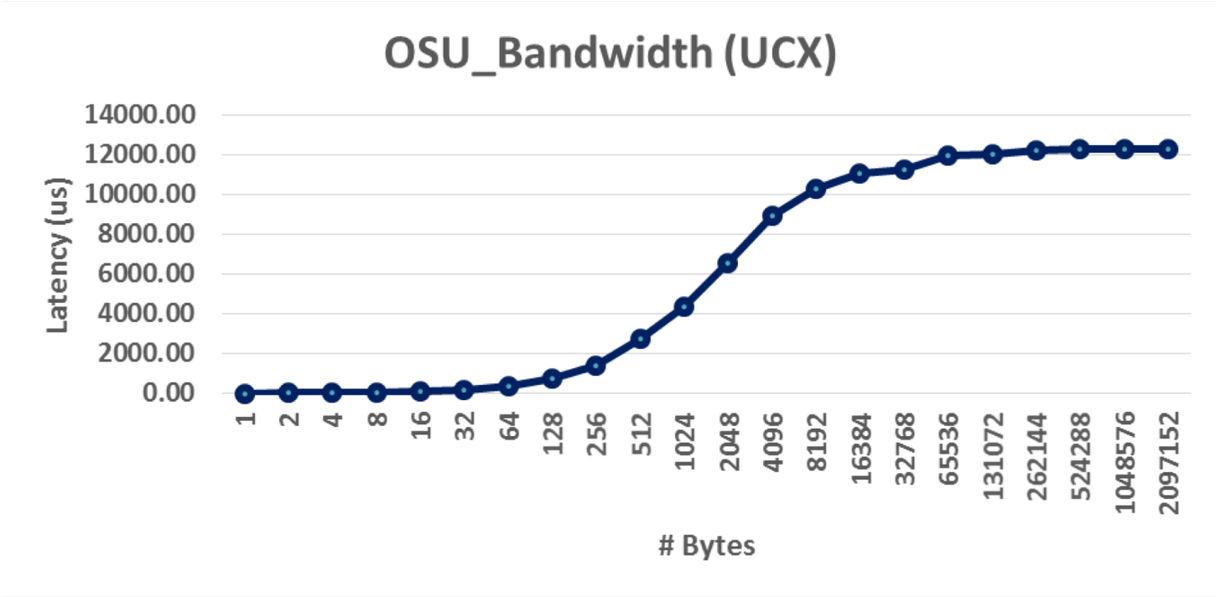
We Love Testing

Over 100,000 tests per commit

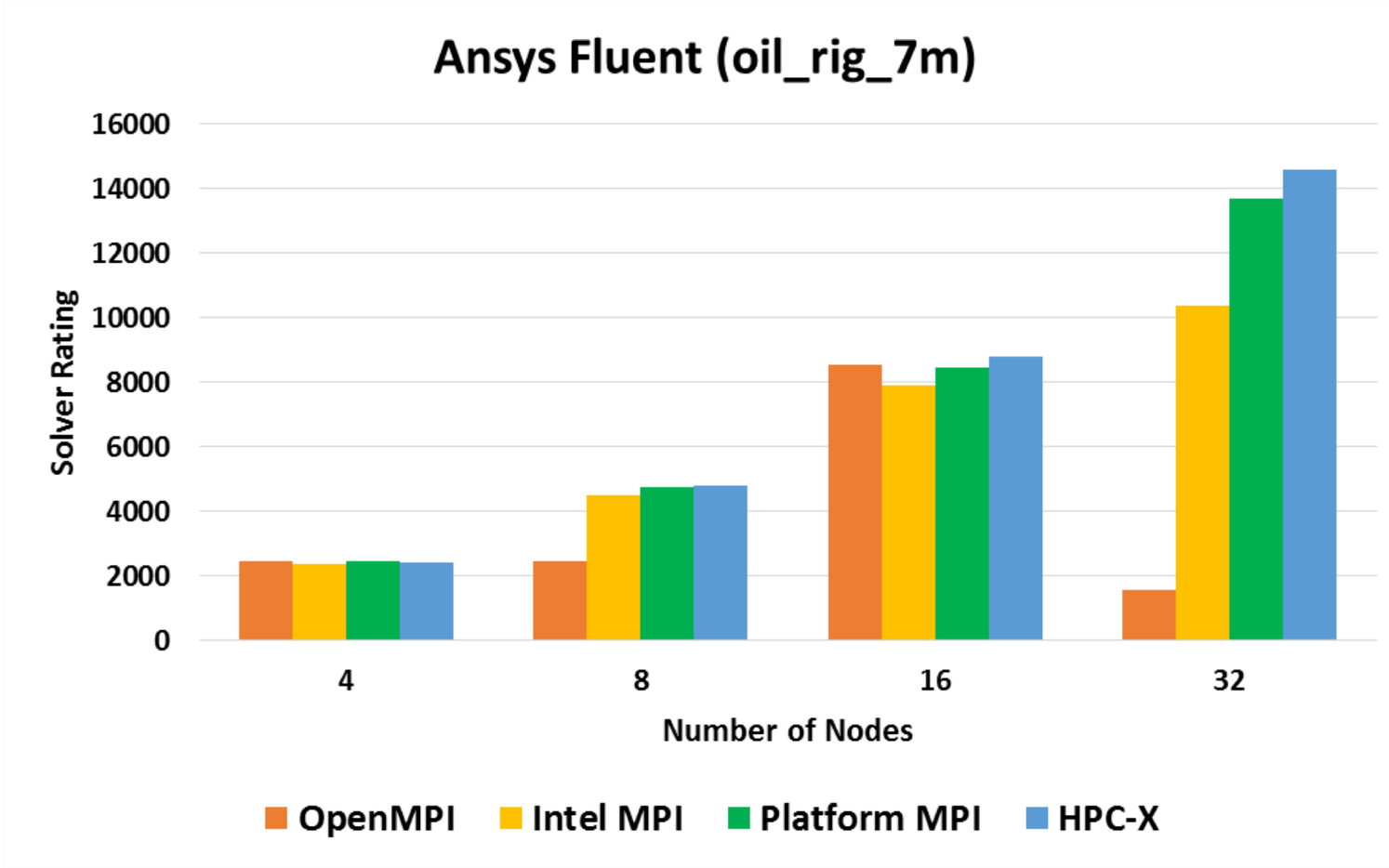
220,000 CPU hours per release

Performance – MPI Latency and Bandwidth

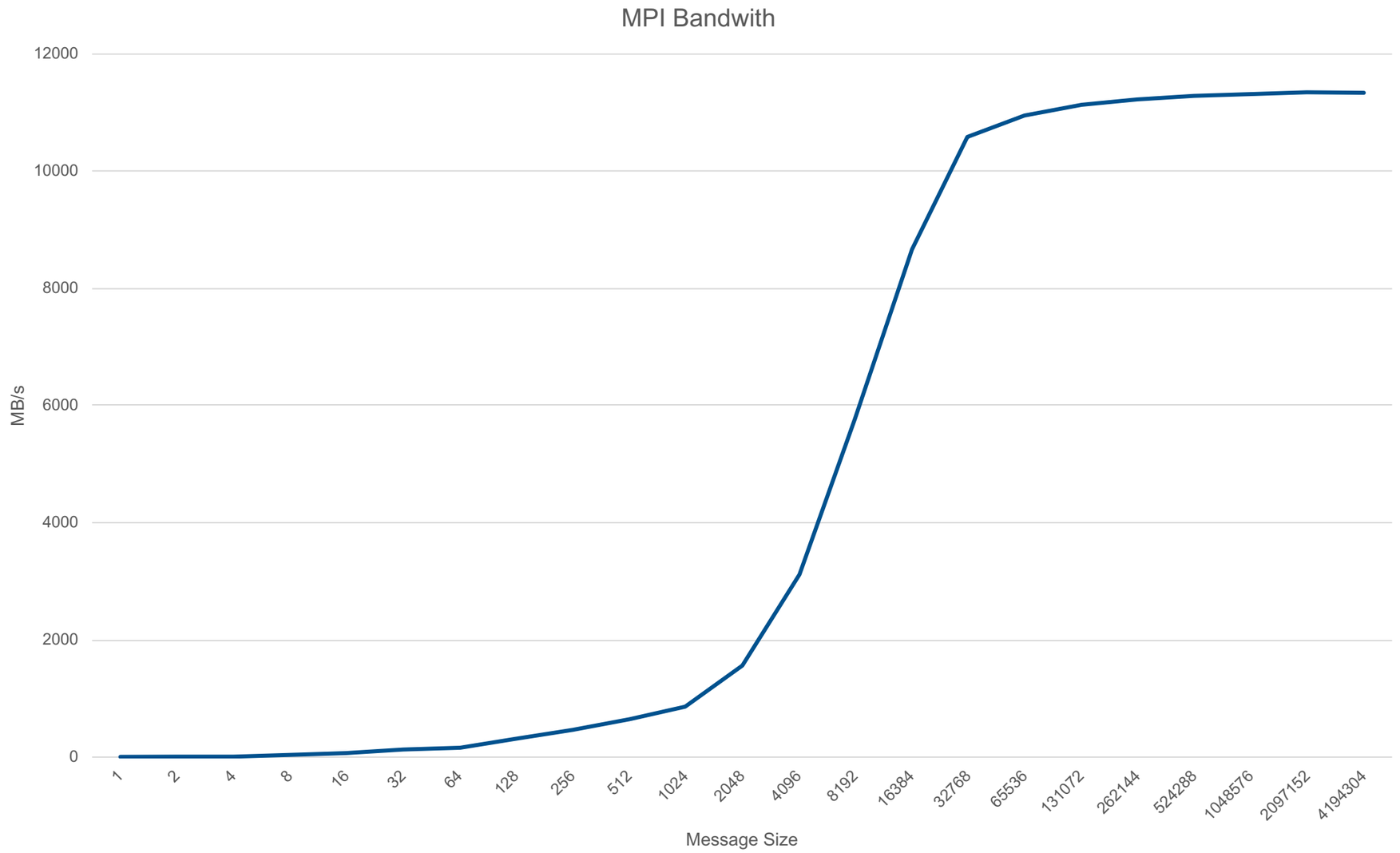




Performance - Fluent

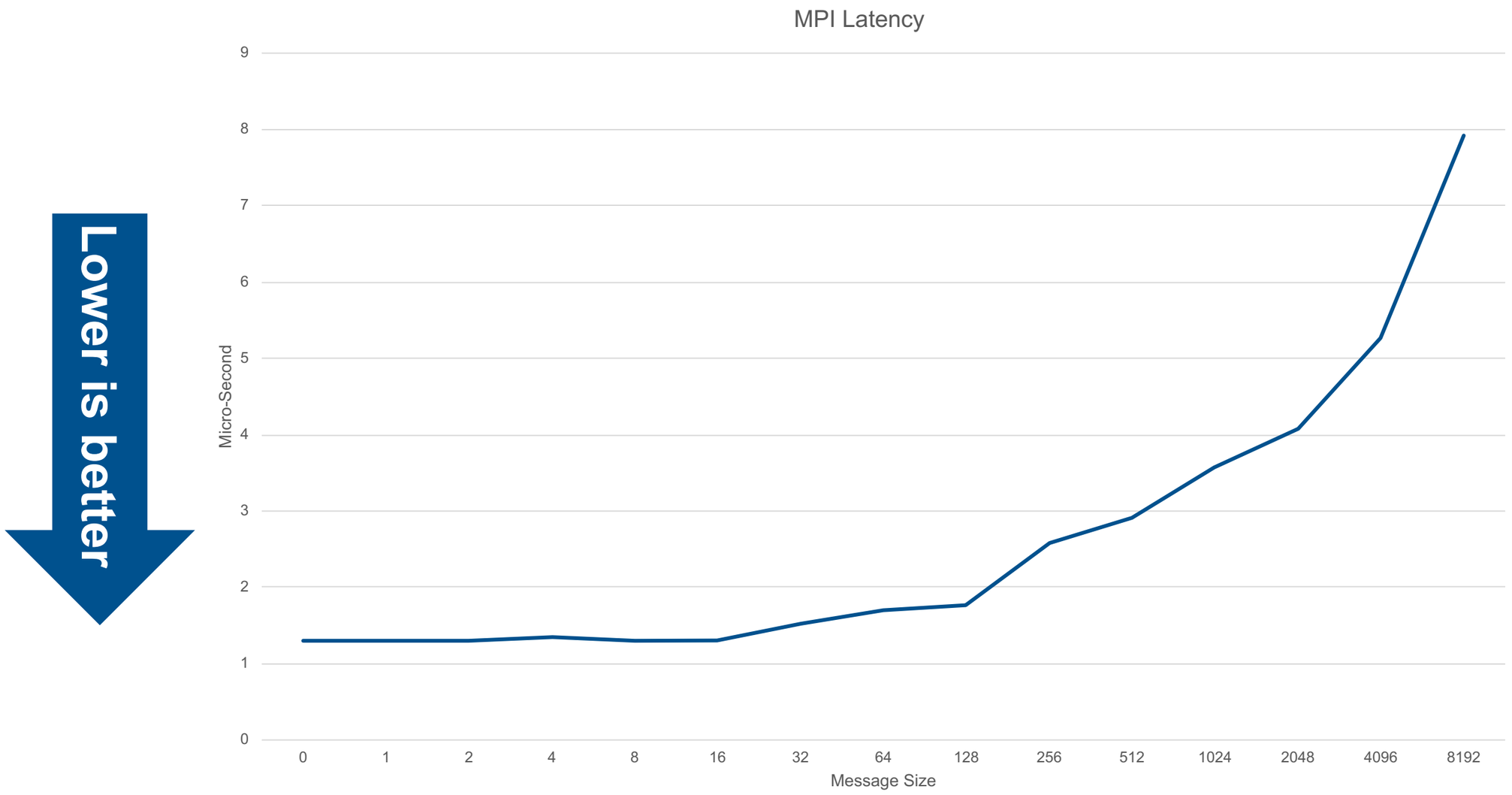


Cavium Thunder X2 SINGLE core InfiniBand Bandwidth

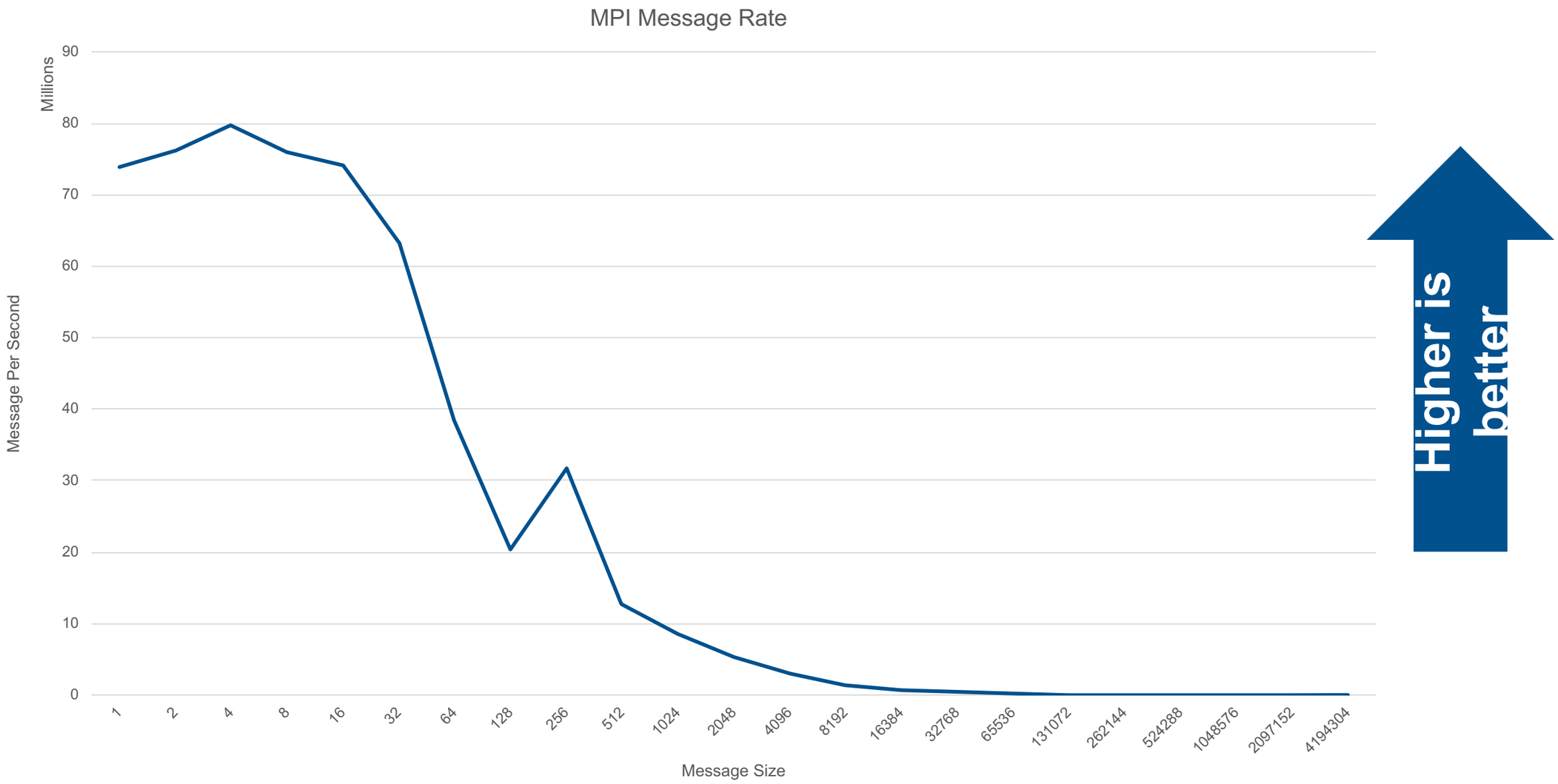


Higher is better

Cavium Thunder X2 MPI Ping-Pong Latency with InfiniBand



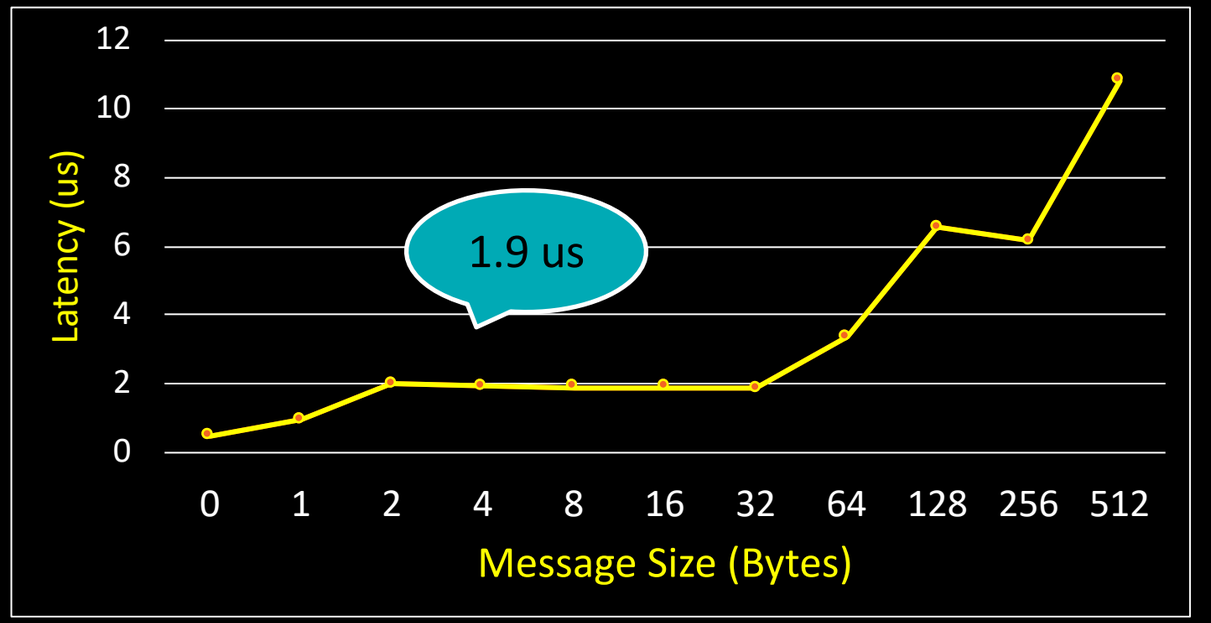
Cavium Thunder X2 MPI Message Rate with InfiniBand (28 cores)



- Open MPI + UCX full scale !

UCX over ROCm: Intra-node support

- Zero-copy based design
  - uct_rocm_cma_ep_put_zcopy
  - uct_rocm_cma_ep_get_zcopy
- Zero-copy based implementation
  - Similar to the CMA UCT code in UCX
  - ROCm provides similar functions to the original CMA for GPU memories
    - hsaKmtProcessVMWrite
    - hsaKmtProcessVMRead
- IPC for intra-node communication
  - Working on providing ROCm-IPC support in UCX
- Test-bed:
  - AMD FIJI GPUs, Intel CPU, Mellanox Connect-IB
  - OMB latency benchmark



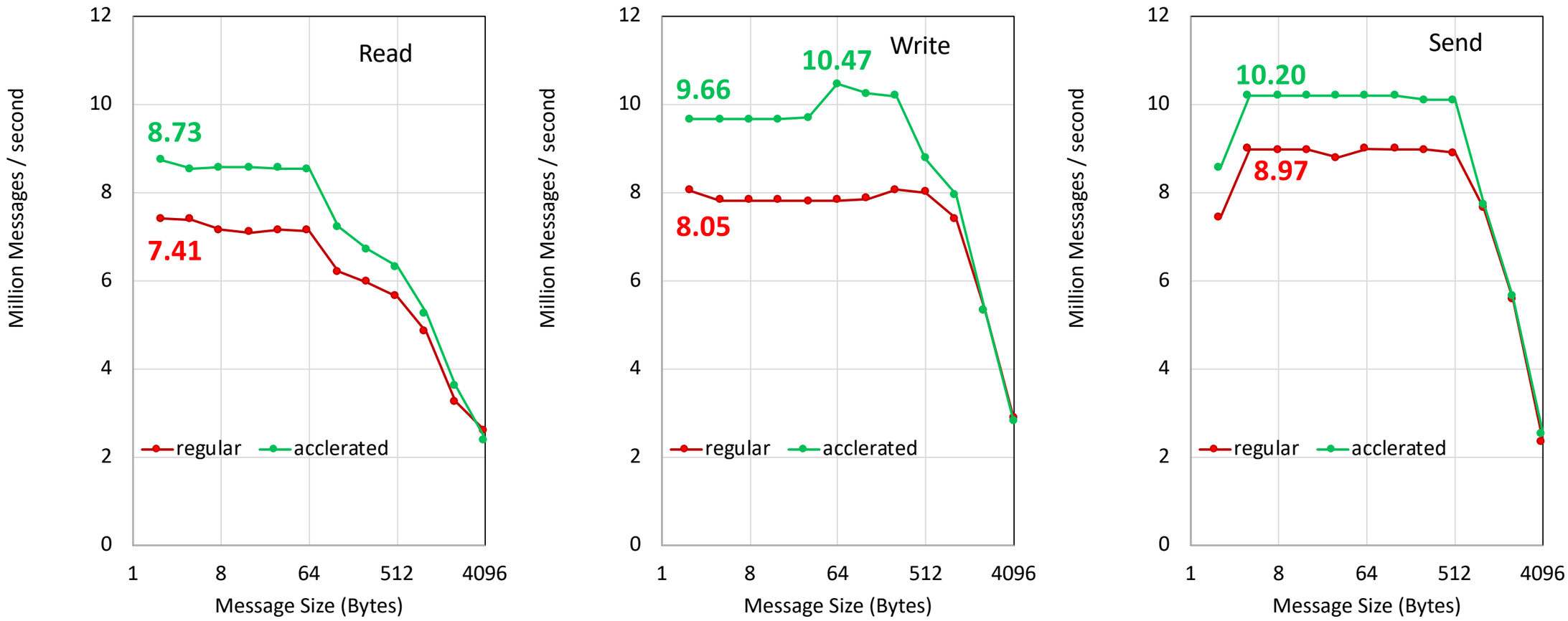
- ROCM-CMA provides efficient support for large messages
- 1.9 us for 4 Bytes transfer for intra-node D-D
- 43 us for 512KBytes transfer for intra-node

UCX Support in CH4

- UCX Netmod Development
  - MPICH Team
  - Tommy Janjusic (Mellanox)
- MPICH 3.3rc1 just released
  - Includes an embedded UCX 1.4.0
- Native path
  - pt2pt (with pack/unpack callbacks for non-contig buffers)
  - contiguous put/get rma for win_create/win_allocate windows
- Non-native path is CH4 active messages (hdr + data)
  - Layered over UCX tagged API
- Not yet supported
  - MPI dynamic processes

OSU Latency: 0.99us
OSU BW: 12064.12 MB/s
Argonne JLSE Gomez Cluster
- Intel Haswell-EX E7-8867v3 @ 2.5 GHz
- Connect-X 4 EDR
- HPC-X 2.2.0, OFED 4.4-2.0.7

Verbs-level Performance: Message Rate



ConnectX-5 EDR (100 Gbps), Intel Broadwell E5-2680 @ 2.4 GHz
MOFED 4.2-1, RHEL-7 3.10.0-693.17.1.el7.x86_64

Unified Communication - X Framework

WEB:

www.openucx.org

https://github.com/openucx/ucx

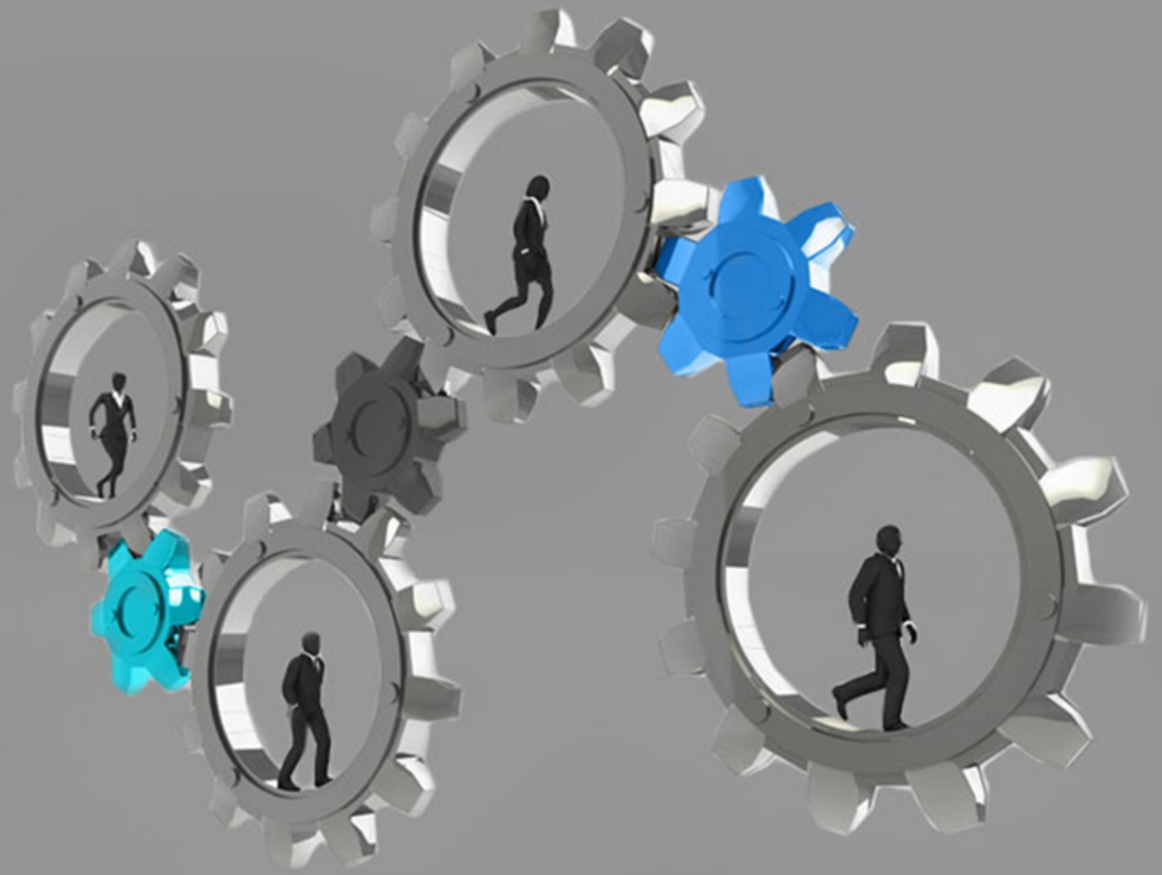
Mailing List:

https://elist.ornl.gov/mailman/listinfo/ucx-group

ucx-group@elist.ornl.gov

ENABLER OF CO-DESIGN

UCF



Thank You

The UCF Consortium is a collaboration between industry, laboratories, and academia to create production grade communication frameworks and open standards for data centric and high-performance applications.