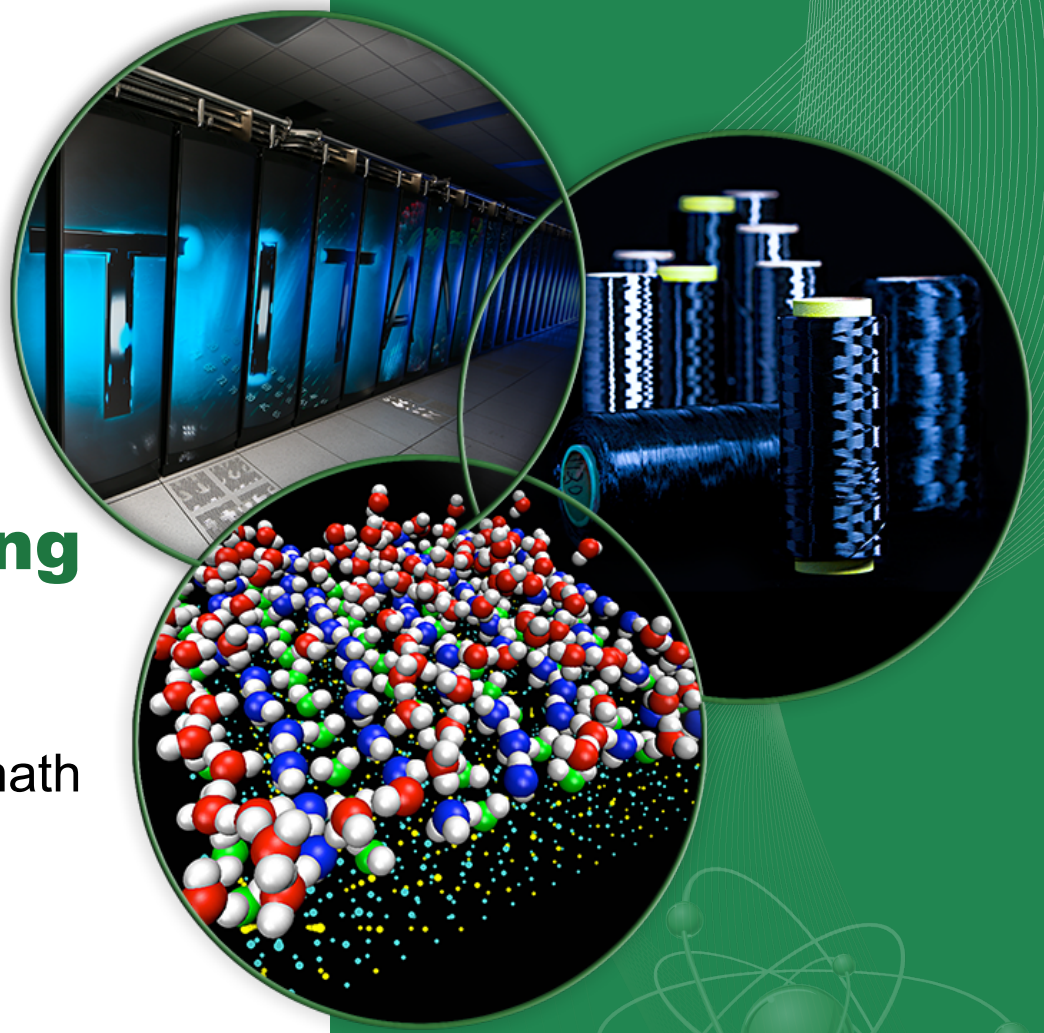
Oak Ridge National Laboratory
Computing and Computational Sciences Directorate



OpenSHMEM-UCX : Evaluation of UCX for implementing OpenSHMEM Programming Model

Languages R&D

Matthew Baker, Ferrol Aderholdt, Manjunath Gorentla Venkata, Pavel Shamis (ARM)

August 3rd, 2016

ORNL is managed by UT-Battelle
for the US Department of Energy

OAK RIDGE National Laboratory

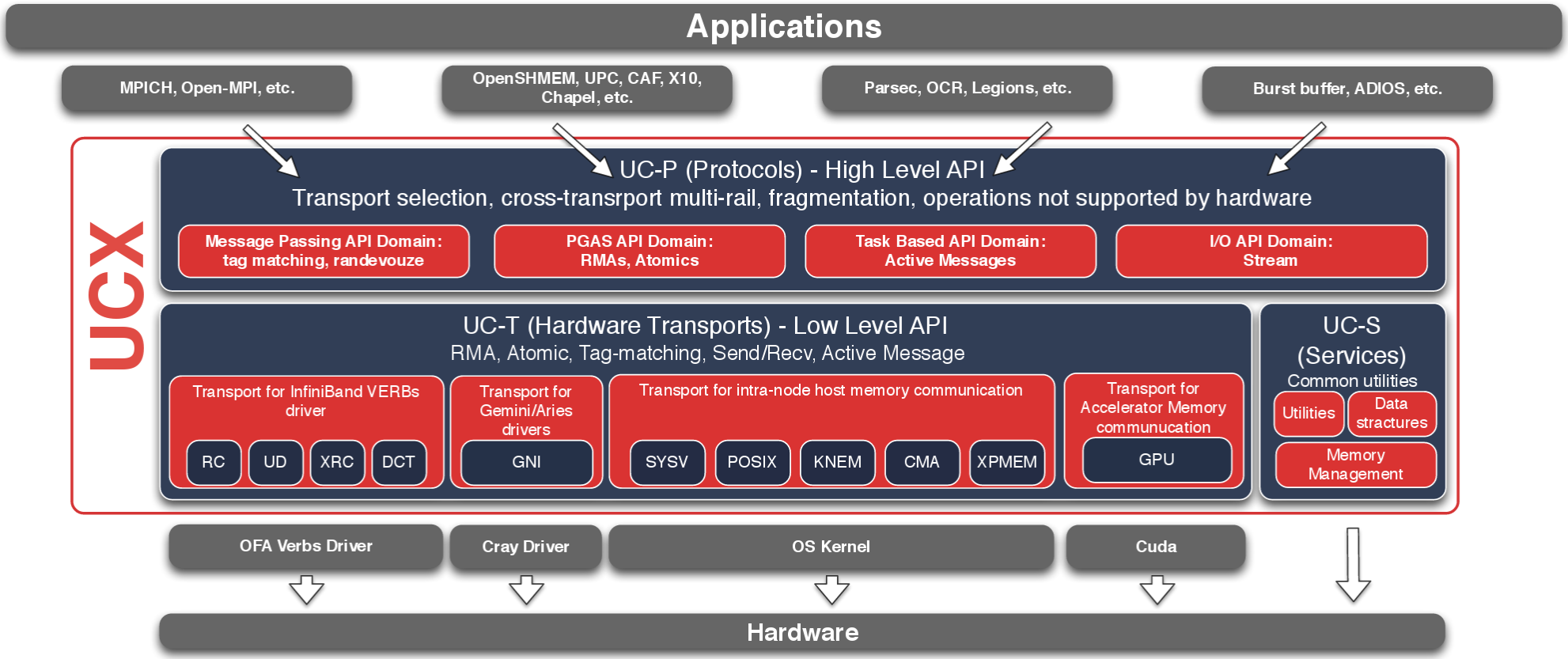
UCX

- Unified Communication layer X
- Framework for communication
- Portable performance
- Scalability

Why UCX is good for PGAS

- UCX developed in close relation with OpenSHMEM team
- Performance Oriented
  - UCX a light weight layer over hardware
  - OpenSHMEM should be a lightweight layer over UCX
- Portable
  - Targeting upper layer UCX protocol allows easy portability with low level transports

UCX Design


Applications
MPICH, Open-MPI, etc.
OpenSHMEM, UPC, CAF, X10, Chapel, etc.
Parsec, OCR, Legions, etc.
Burst buffer, ADIOS, etc.
UCX
UC-P (Protocols) - High Level API
Transport selection, cross-transrport multi-rail, fragmentation, operations not supported by hardware
Message Passing API Domain: tag matching, randevouze
PGAS API Domain: RMAs, Atomics
Task Based API Domain: Active Messages
I/O API Domain: Stream
UC-T (Hardware Transports) - Low Level API
RMA, Atomic, Tag-matching, Send/Recv, Active Message
Transport for InfiniBand VERBs driver
RC
UD
XRC
DCT
Transport for Gemini/Aries drivers
GNI
Transport for intra-node host memory communication
SYSV
POSIX
KNEM
CMA
XPMEM
Transport for Accelerator Memory communucation
GPU
UC-S (Services)
Common utilities
Utilities
Data stractures
Memory Management
OFA Verbs Driver
Cray Driver
OS Kernel
Cuda
Hardware

UCX layers

- UCT
  - Supports specific hardware
  - Currently supported: Cray UGNI, Mellanox IB, shared memory
  - Abstractions
    - Memory domain (md)
      - Memory registration
    - Interface (iface)
      - A particular interface to a network device
    - End Point (ep)
      - A particular point to point communication object
    - Worker
      - ensures that underlying network hardware makes progress

UCX layers

- UCP
  - Provides protocol abstractions.
  - Abstracts away common high level tasks
    - Selects best mechanism for transfers
    - Wiring up end points
    - Selecting transports
    - Tagged messages
  - Implements missing features
    - 32 bit atomics on Gemini
  - Workers
    - Provides application contexts, one application can have multiple progress threads

UCX layers

- UCS
  - Common platform services
    - Local atomics
    - Thread safety
  - Data structures
  - Debug and logging output
- UCM
  - Memory management
  - Manages registration caches

UCX on Cray hardware

- UCT on Cray has 3 different transports
  - FMA and BTE APIs in RDMA transport
  - SMSG short messages in SMSG transport
  - Datagrams in UDT transport
- Each exposes different capabilities in UCT
  - RDMA exposes RMA bcopy/zcopy
  - SMSG exposes active messages
  - UDT exposes active messages on interfaces

OpenSHMEM on UCX on Cray

- OpenSHMEM only requires the RDMA transport.
  - UCX will, on its own, only initialize RDMA and skip SMSG
- Nothing special required, other than passing in correct feature flags

OpenSHMEM on UCX

- OpenSHMEM becomes thin on UCX
- Atomics and put/gets done directly through UCP.
- Collective operations and handled by OpenSHMEM
- OpenSHMEM handles memory, but UCX has hardware specific knowledge
  - OpenSHMEM registers memory, receives keys
  - UCX worries about how to register memory with driver

Benchmark results

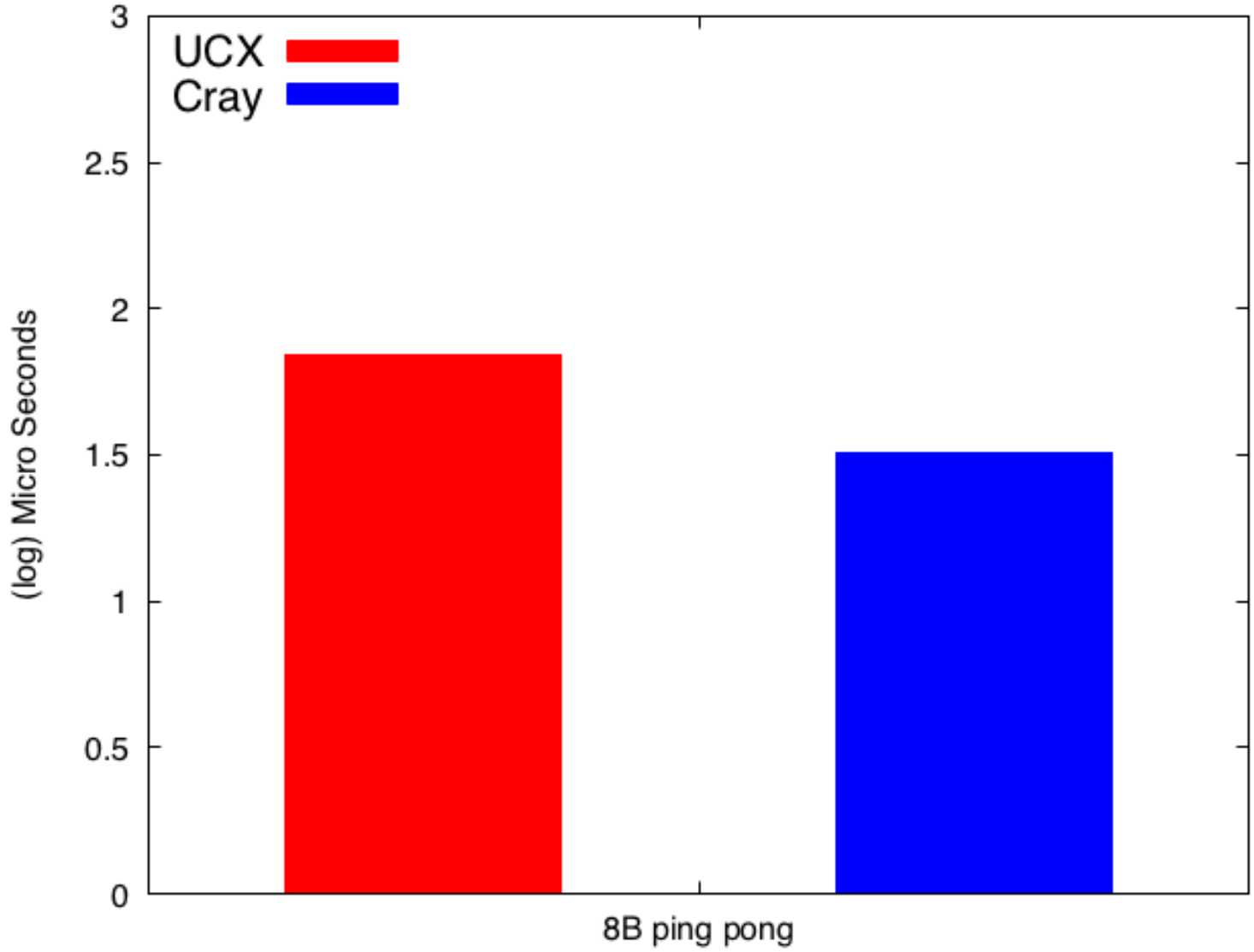
Test System

- Titan
  - 18,688 nodes
  - 16 core AMD cpu per node
  - 32 GB of memory per node
  - Nvidia K20X accelerator

Modified OSU benchmark

- Ping pong test
  - Bouncing puts between two nodes

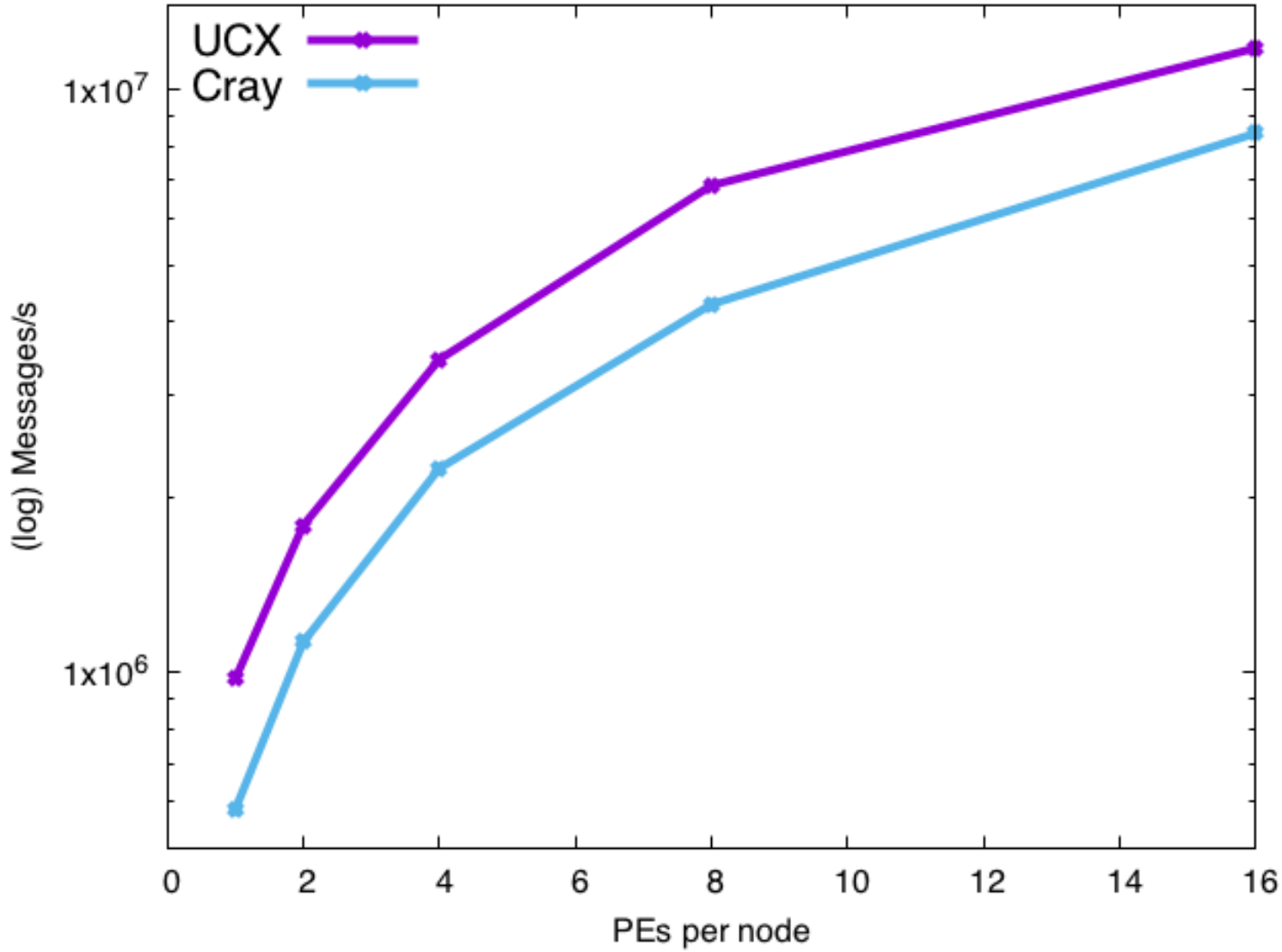
Ping pong test (lower is better)



OSU message rate

- Message rate benchmark from OSU benchmark suite
- Graphed results for 8 byte puts between two nodes, going from 1 PE per node to 16

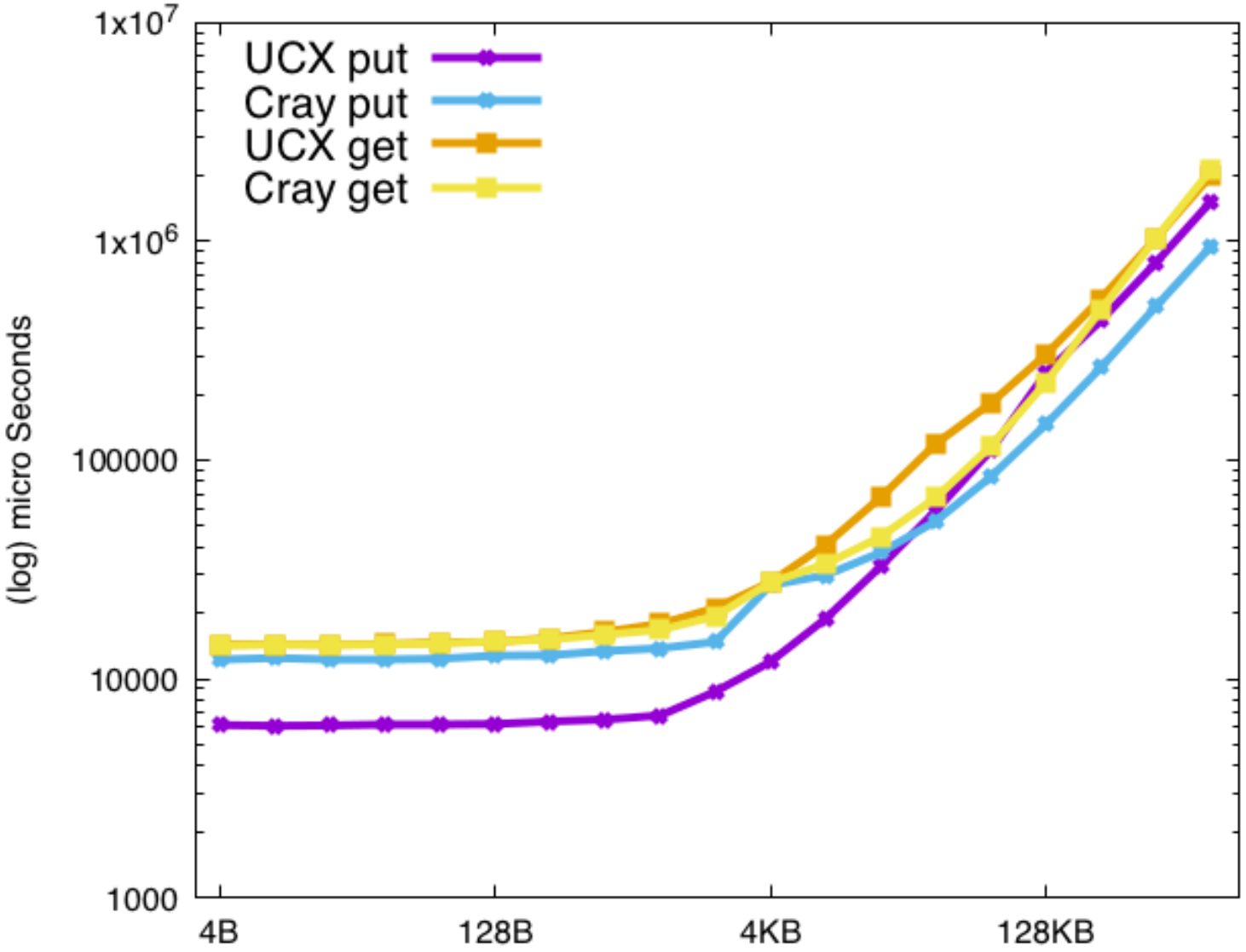
Message Rate



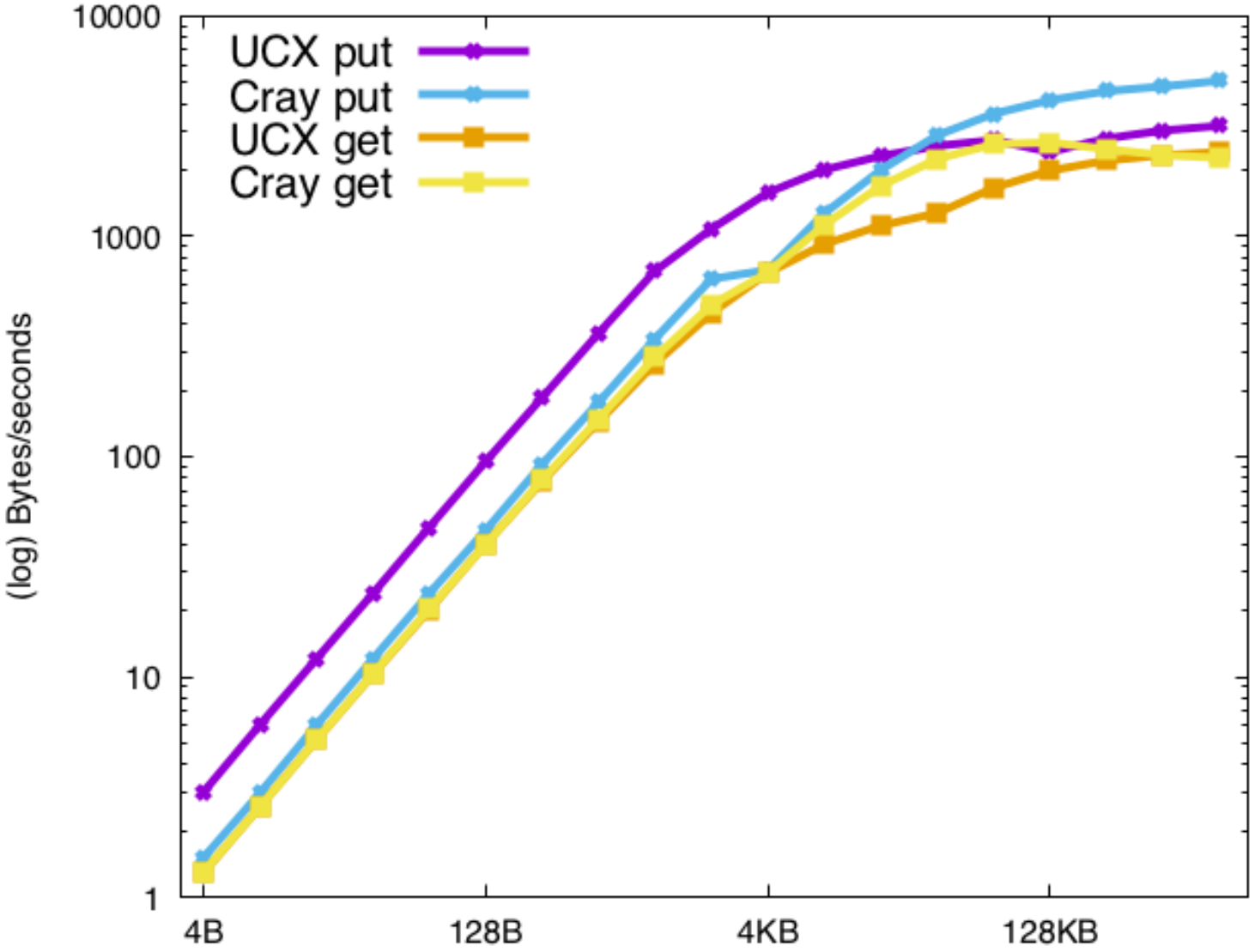
SHOMS tests

- SHOMS is a OpenSHMEM microbenchmark suite
  - Tests cover entire OpenSHMEM API
- Test results
  - One origin PE sending to 4095 other PEs
  - Latency from 4B to 1MB
  - Bandwidth from 4B to 1MB

SHOMS latency test



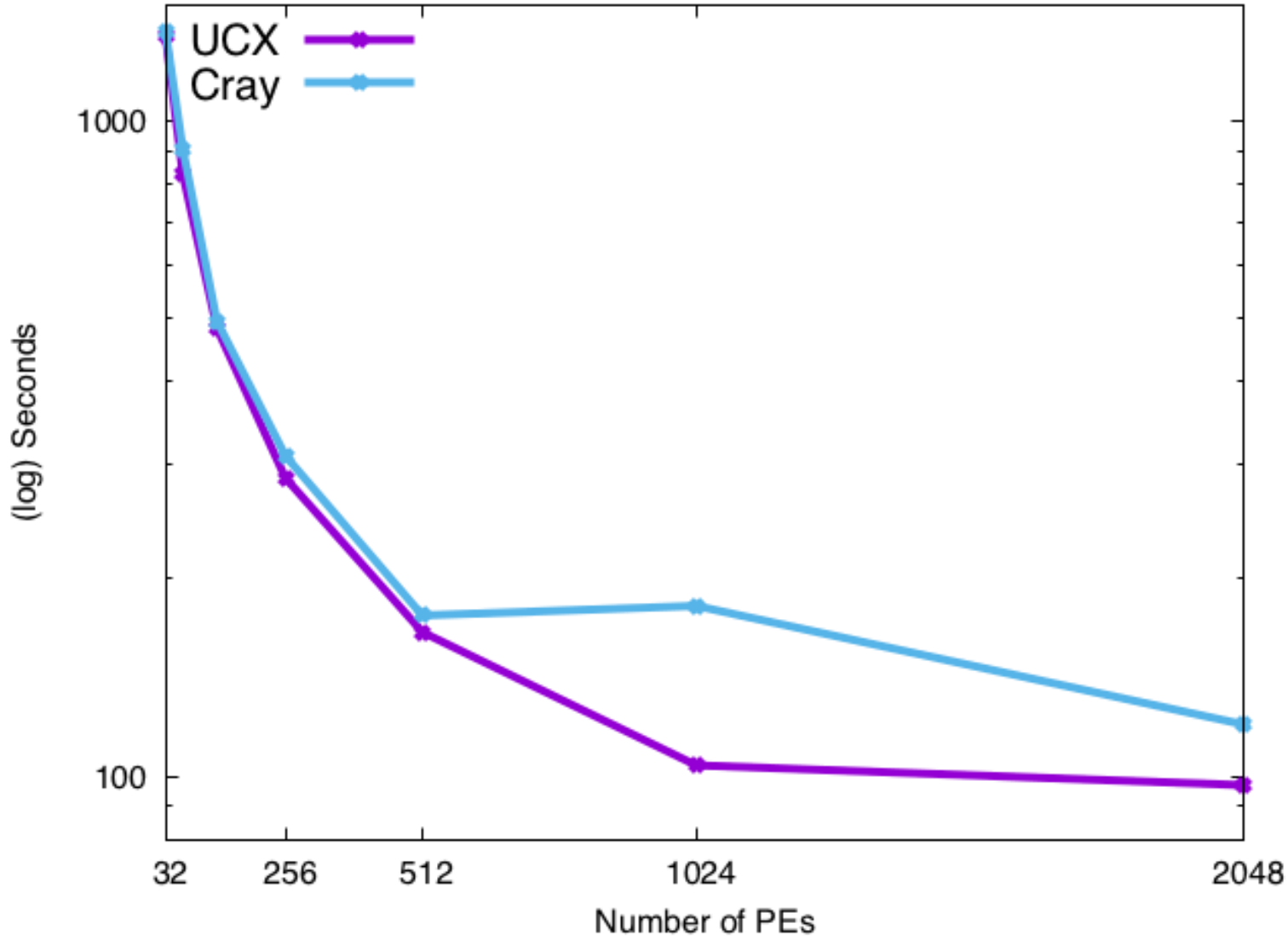
SHOMS bandwidth



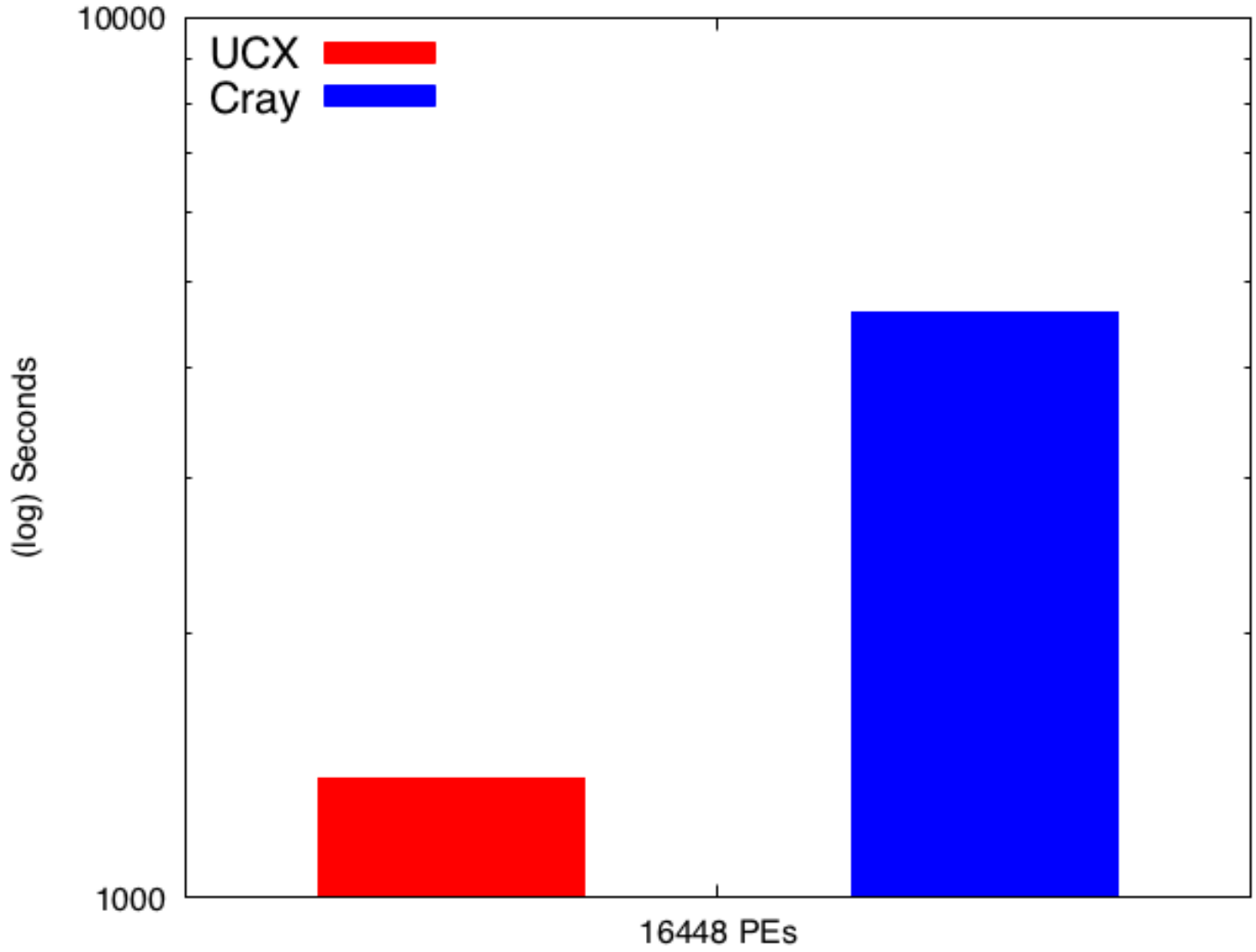
SSCA1 Smith-Waterman

- Genetic local alignment benchmark
- Large parallel inner loop with many short puts and gets
  - 7 gets and 5 puts
- Extremely sensitive to message rates

SSCA1 up to 2048 PEs



SSCA1 16k PEs



Conclusions

- Implementing OpenSHMEM over UCX is light weight
- UCX provides useful abstractions for high performance and high through put
  - Workers, interfaces, endpoints arbiters
- Evaluation of microbenchmarks and kernels prove usefulness
  - Short put and get message rates
  - Still work to be done (improved management of memory registrations)

Acknowledgements

This work was supported by the United States Department of Defense (DoD) and used resources of the Computational Research and Development Programs at Oak Ridge National Laboratory.