EXTENDING AN ASYNCHRONOUS MESSAGING LIBRARY USING AN RDMA-ENABLED INTERCONNECT

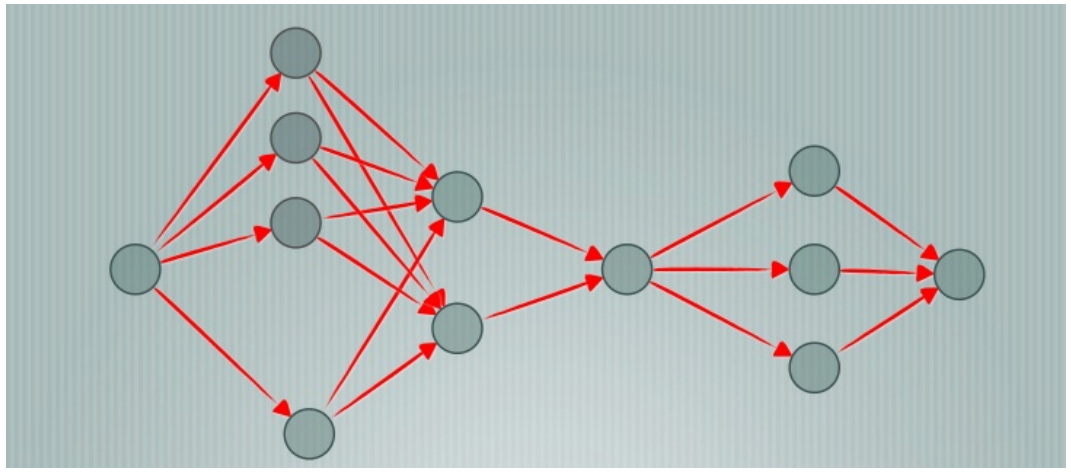
Konstantinos Alexopoulos
ECE NTUA
CSLab

MOTIVATION

- HPC, Multi-node & Heterogeneous Systems
- Communication with low latency
- Reliability
- Provide an instant employment method
- The CERN use-case

ZEROMQ

- Messaging Library
- Does not employ a messaging broker
  - Low Latency
- Asynchronous I/O
- Easy deployment of complex topologies



ZEROMQ

- Supports many platforms and language bindings
- Open source project with active development
- IPC, UDP, TCP/IP
- Port to an RDMA-enabled interconnect

COMMUNICATION PARADIGMS

TCP Socket Semantics

- High Overhead
- Decent Bandwidth
- Low implementation effort

RDMA Semantics

- Low Overhead
- Very Low Latency
- Very High Throughput
- High implementation effort

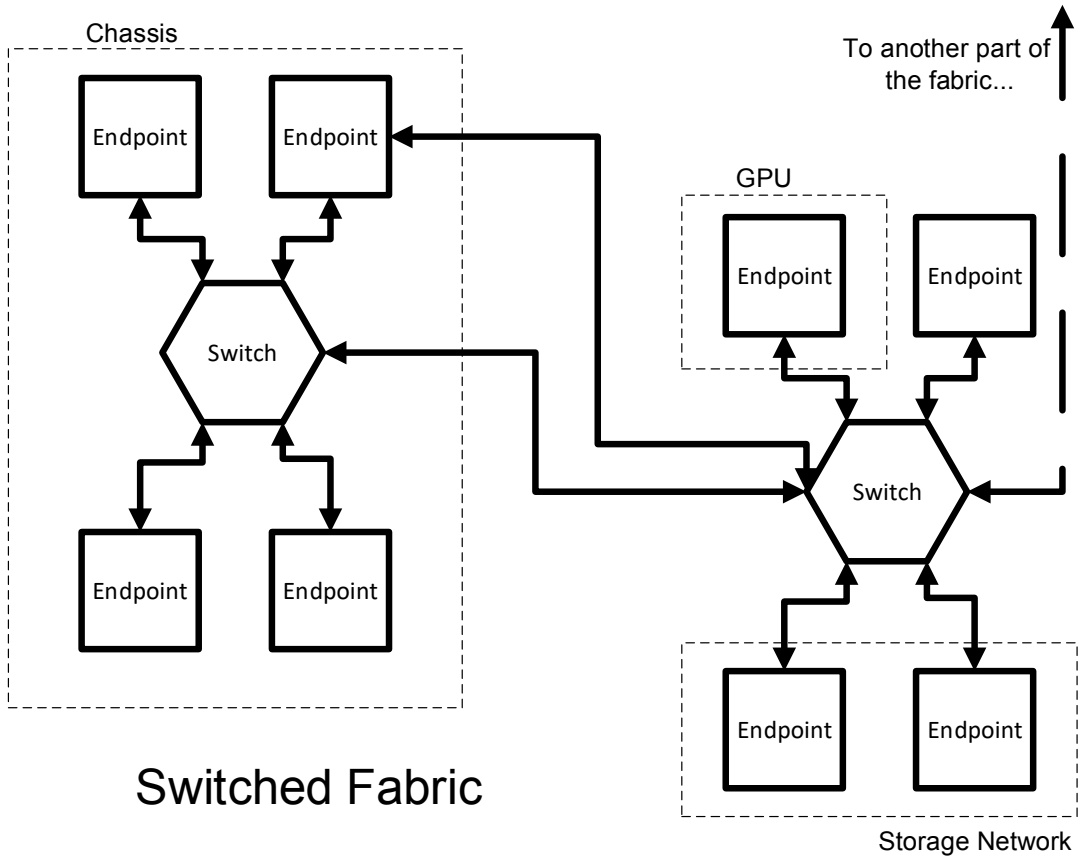
SWITCHED FABRICS

Switched Fabric Architectures

- Allow for any topology
- Reliable
- ”Inside-the-box” & “Outside-the-box”

Shared Bus Architectures

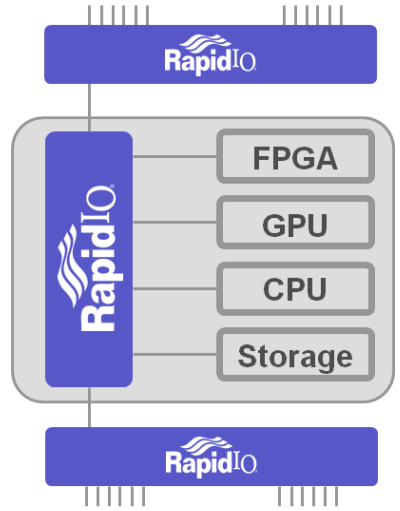
- Topology restrictions
- Bottlenecks
- ”Inside-the-box” only

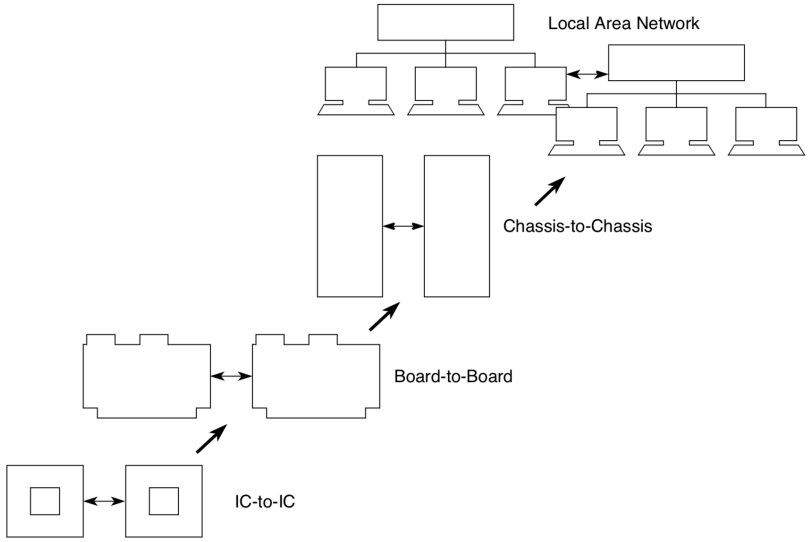


Switched Fabric

RAPIDIO

- System-level interconnect originally
- Independent from Physical Implementation
- Lately oriented towards chassis-to-chassis and SANs
- Protocol stack processed in HW
- Destination Based Routing





RIO OPERATIONS – MESSAGING INTERFACE

Channelized Messages

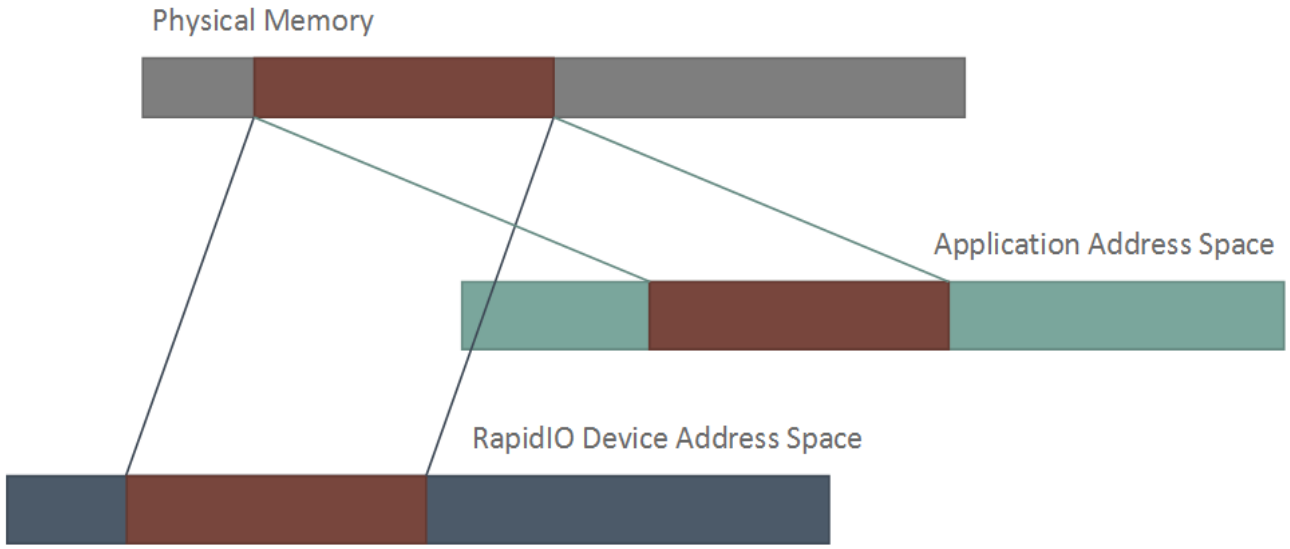
- Maximum 4K
- Socket-like interface
- Sent to a channel

Doorbells

- Hardware Signals
- 8B software-defined payload
- Have to allocate exclusive range to receive doorbells

RIO OPERATIONS – MMIO

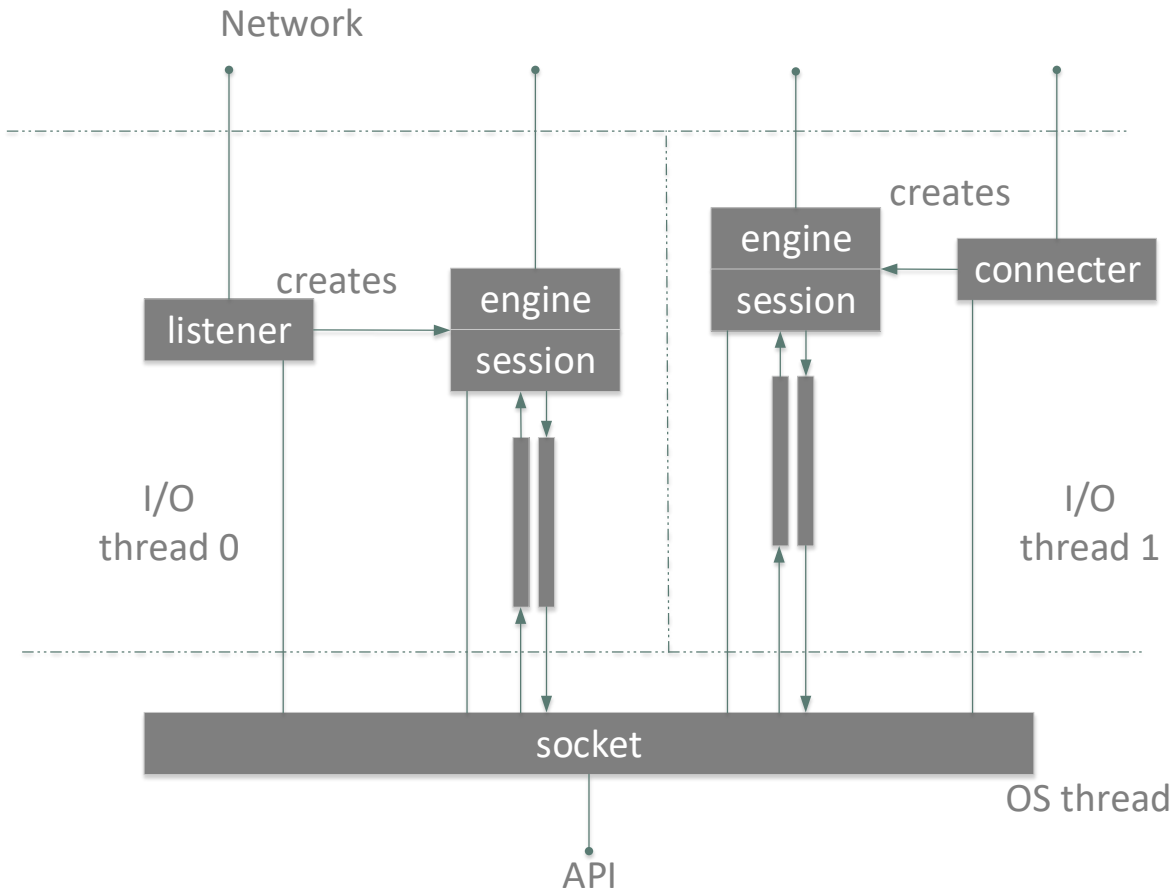
- Remote Direct Memory Access
  - Read/Write
  - Zero-copy
  - "One-way communication"
  - Device memory mapped to physical memory
    - Needs to be done at boot time
    - Kernel boot parameter
  - Physical memory mapped to process address space
    - Done through a library call in the application layer
  - Supports multi-cast



DESIGN & IMPLEMENTATION | RIOZMQ

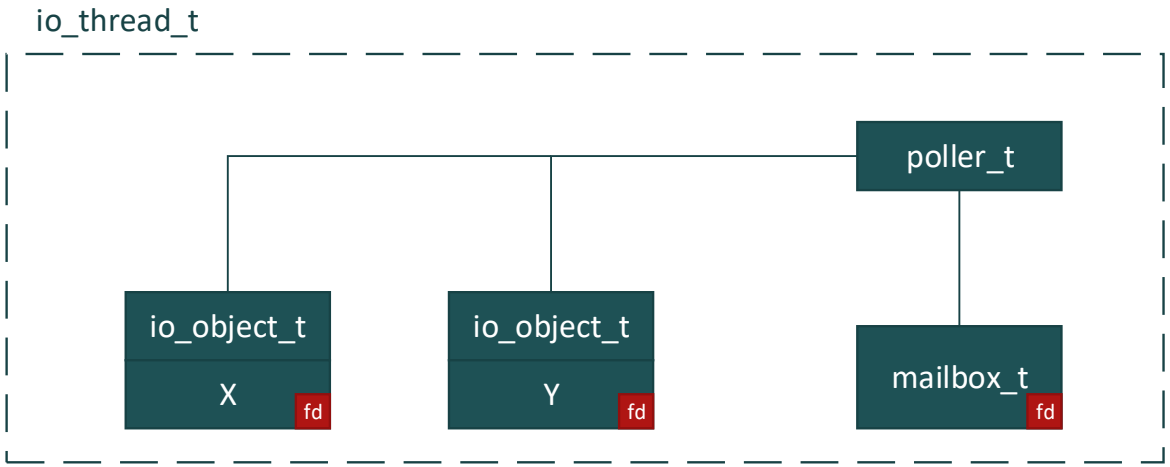
ZEROMQ INTERNAL ARCHITECTURE

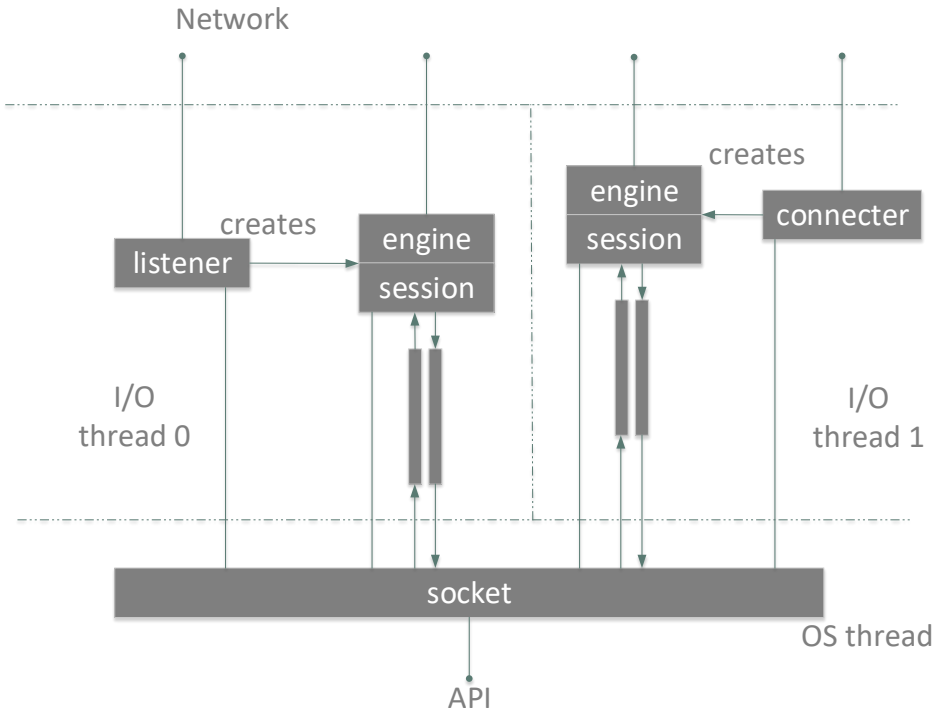
- User creates (0mq!) Socket
- Socket binds/connects
- Listener/Connecter create Session
- Session/Engine object for any new connection



ZEROMQ INTERNAL ARCHITECTURE

- "I/O threads" for asynchronous operations
- in_event() and out_event() for every io_object_t

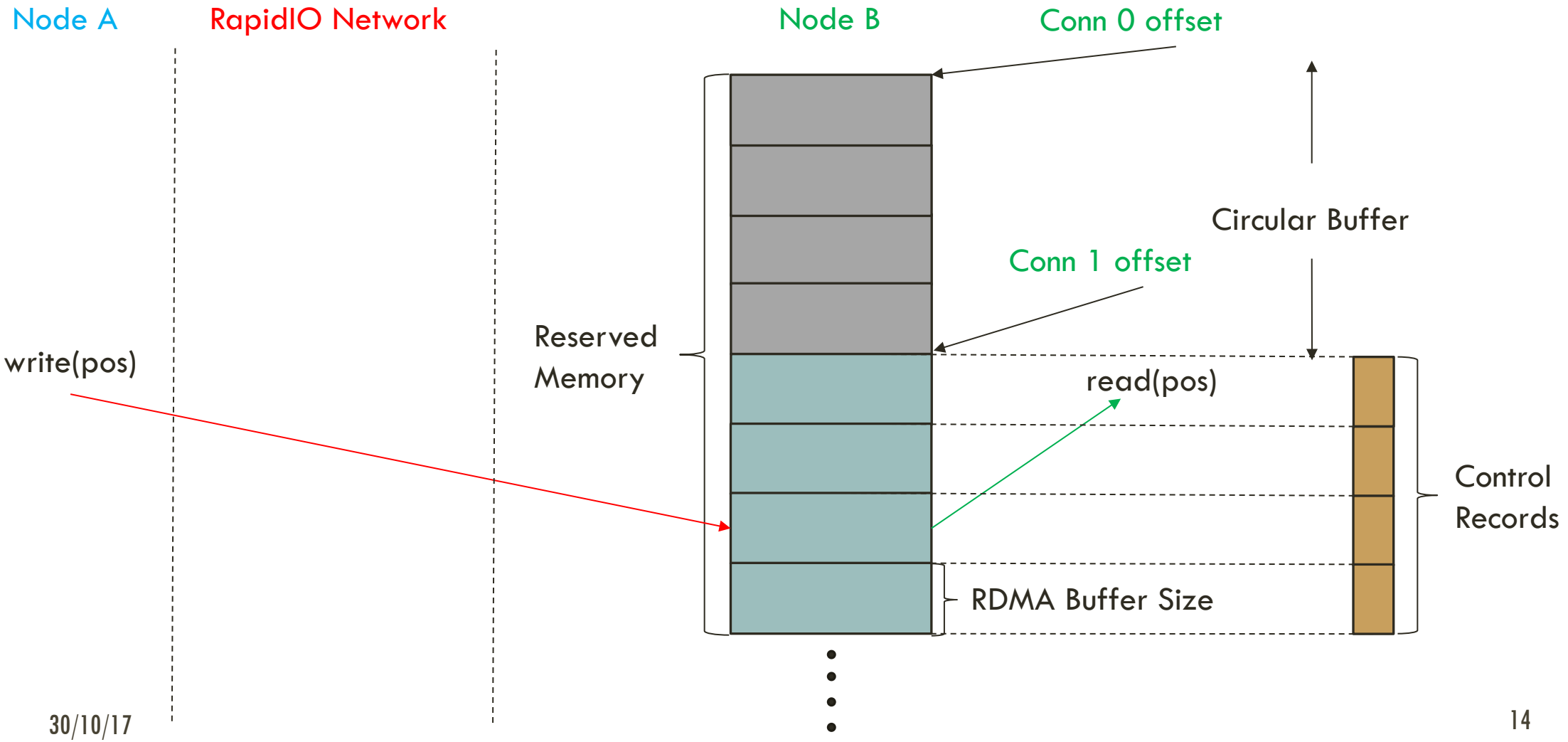




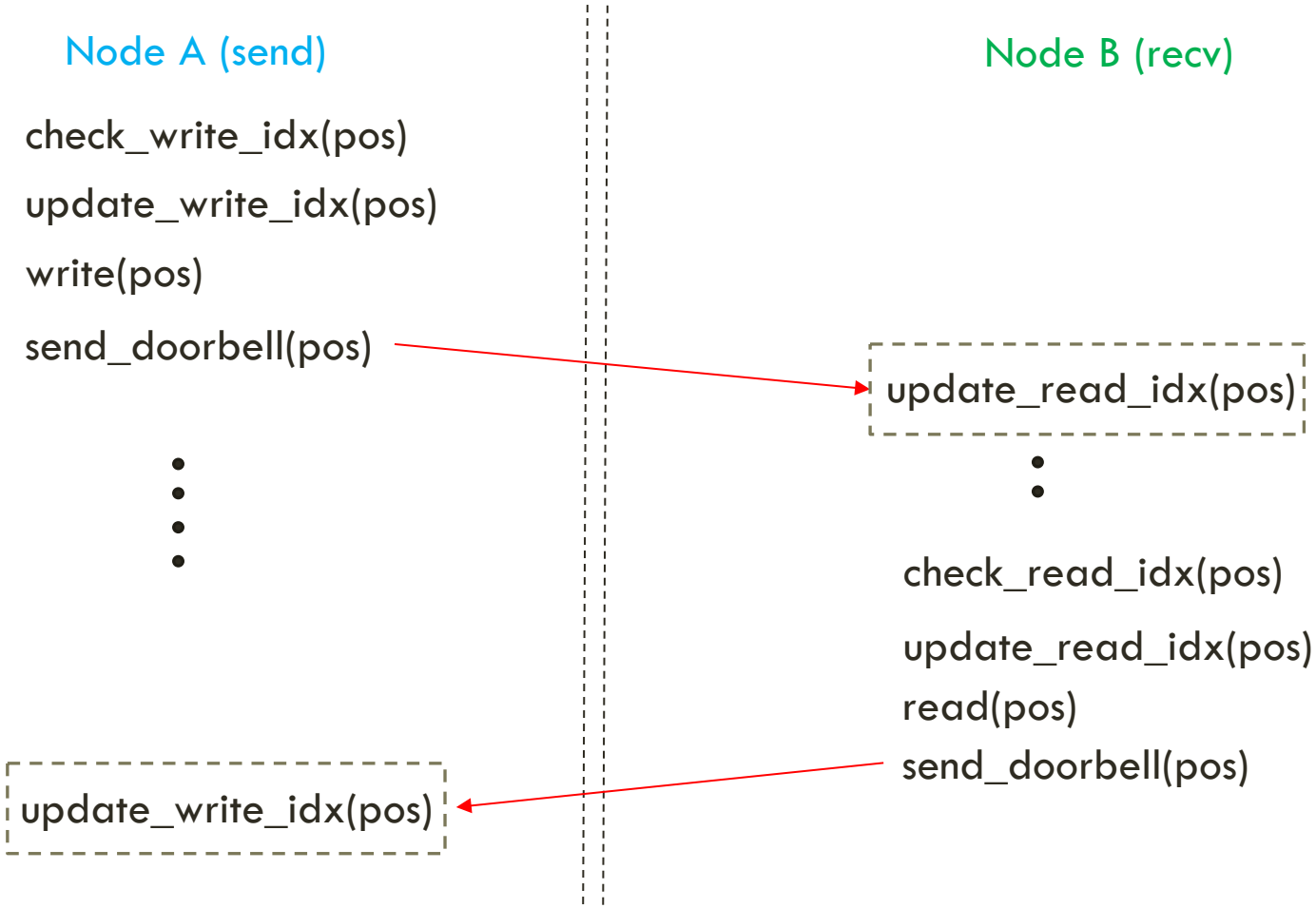
RIOZMQ EXTENSION

- rio_address
  - resolves address of rio://[destID]:[channel] format
- rio_connecter / rio_listener
  - connect()/bind()
  - RDMA target addresses exchange
  - doorbell range allotment
- rio_engine
  - RDMA write/recv
- rio_mailbox
  - doorbell operations
  - FD registered with io_object_t
- glue code in socket/session

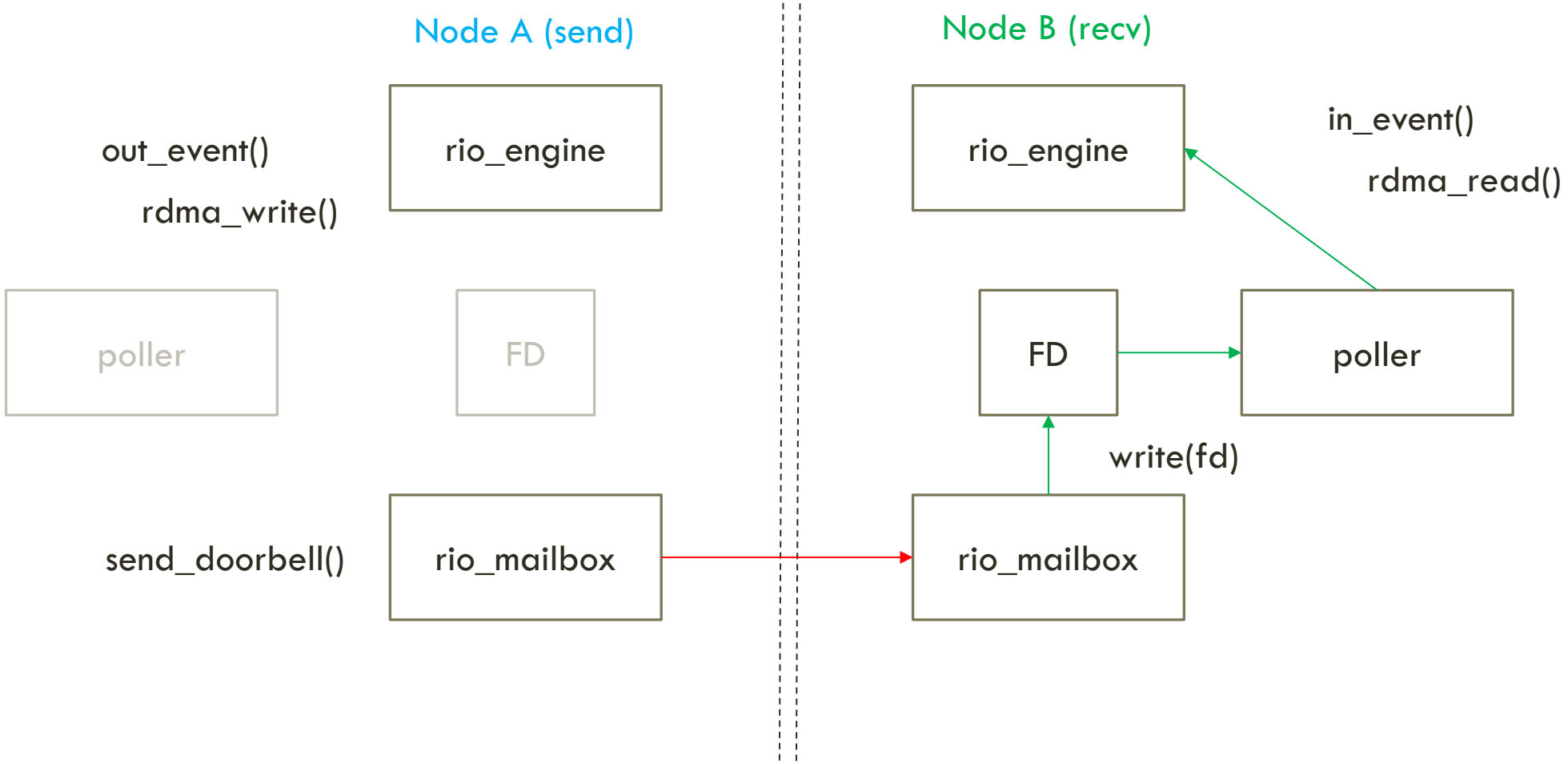
MEMORY SCHEME



DOORBELLS AS NOTIFICATIONS



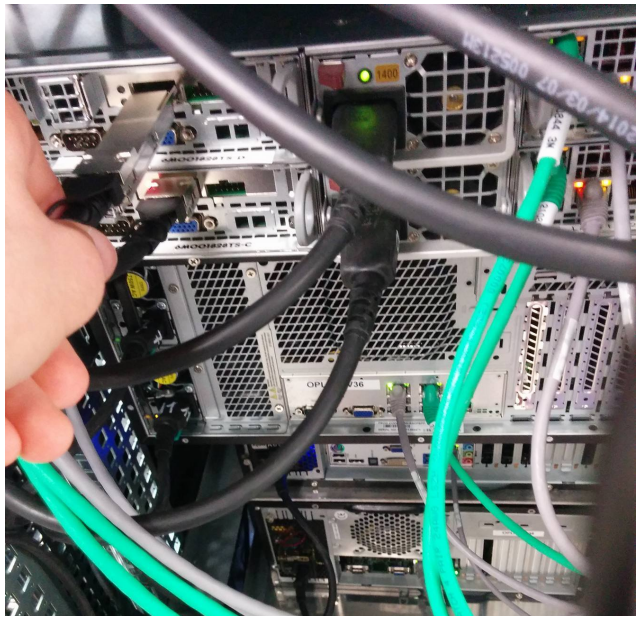
DOORBELLS - FILE DESCRIPTORS - POLLER



EVALUATION

HARDWARE SETUP

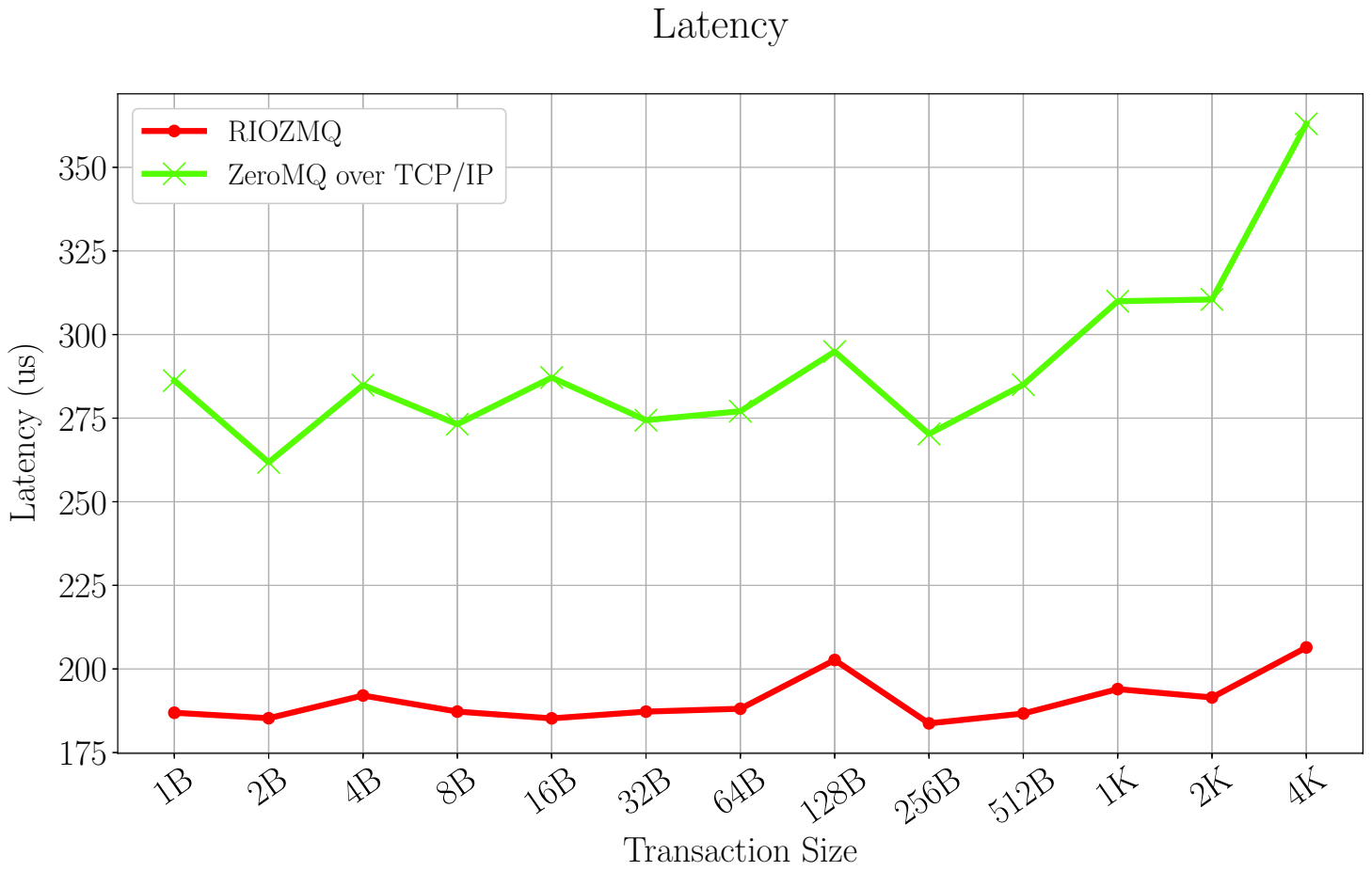
- 4x 2U Quad Units
- 4 Nodes per Unit
- Intel Xeon L5640 @ 2.27Ghz
- 48GB of DDR3 1333MHz RAM
- IDT Tsi721 RapidIO to PCIe bridge cards
- QSFP+ cables
- 38-port Top of Rack (ToR) RapidIO Gen2 switch
- CERN CentOS



BENCHMARKS

- Standard ZeroMQ Benchmarks
- Measures round trip time (RTT) between 2 nodes

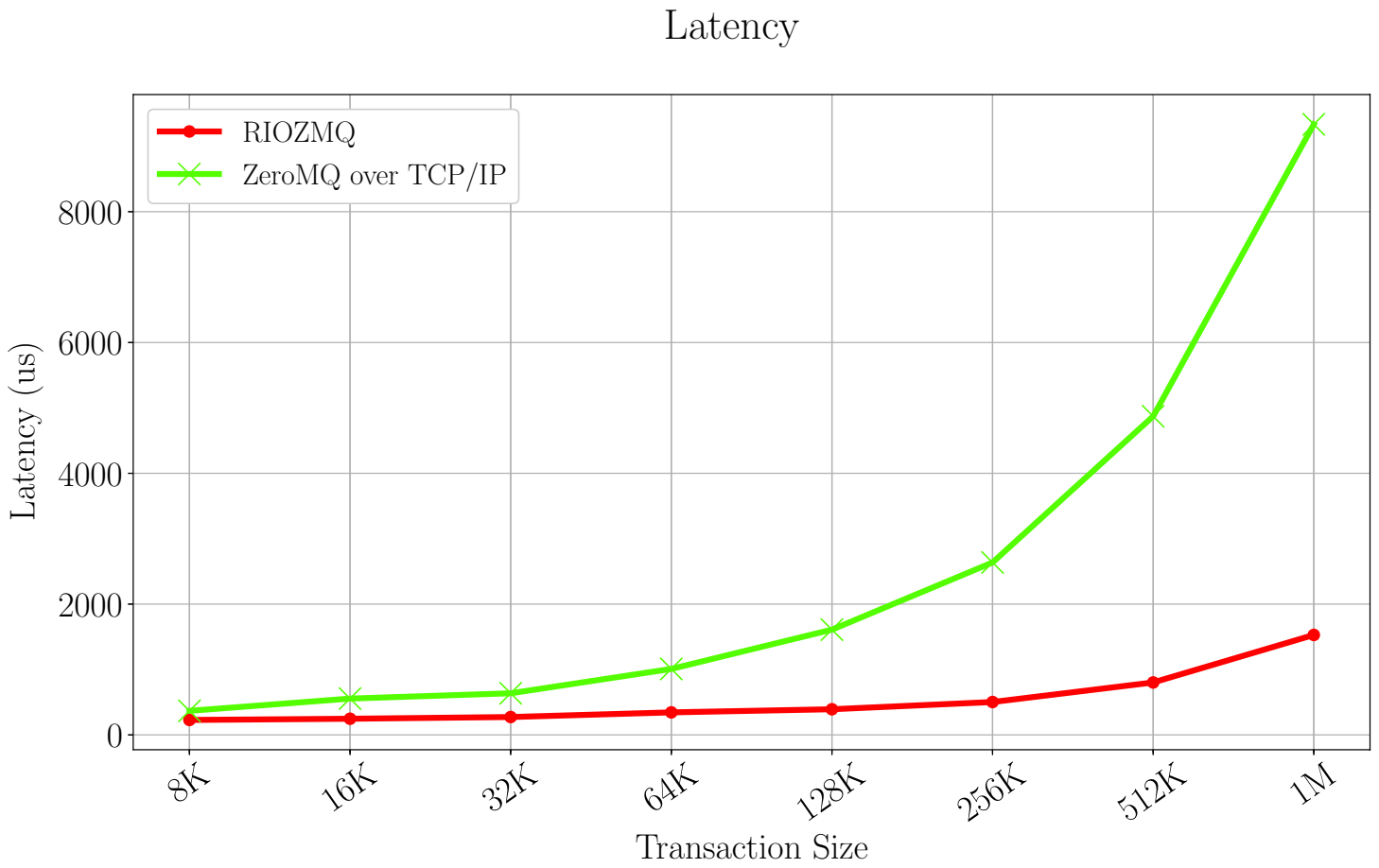
EVALUATION – LATENCY SMALL



- RDMA Buffer Size : 4K
- Circular Buffer Length : 1

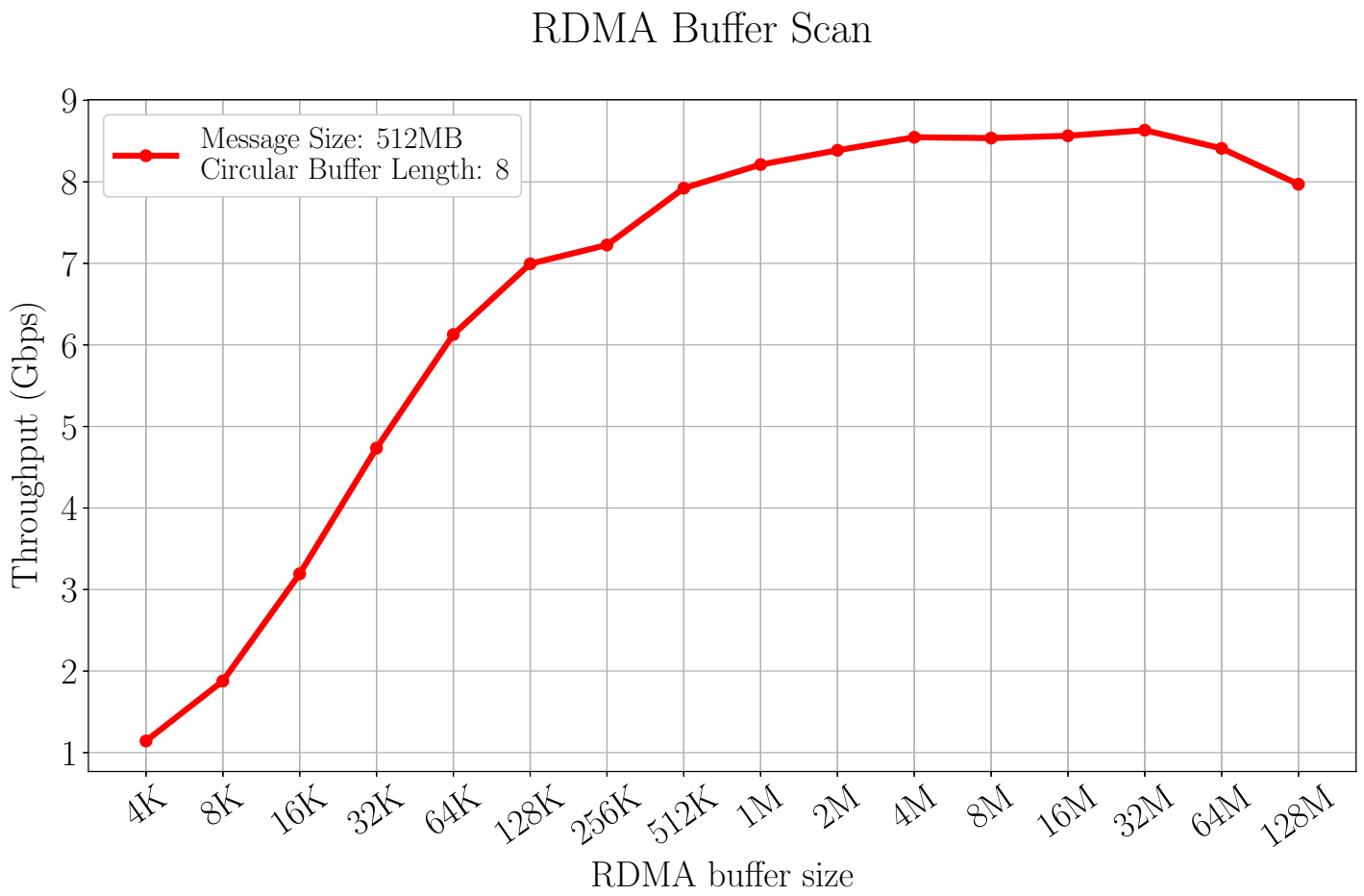
- RIOZMQ 65% faster

EVALUATION – LATENCY LARGE



- RDMA Buffer Size : 1M
- Circular Buffer Length : 1

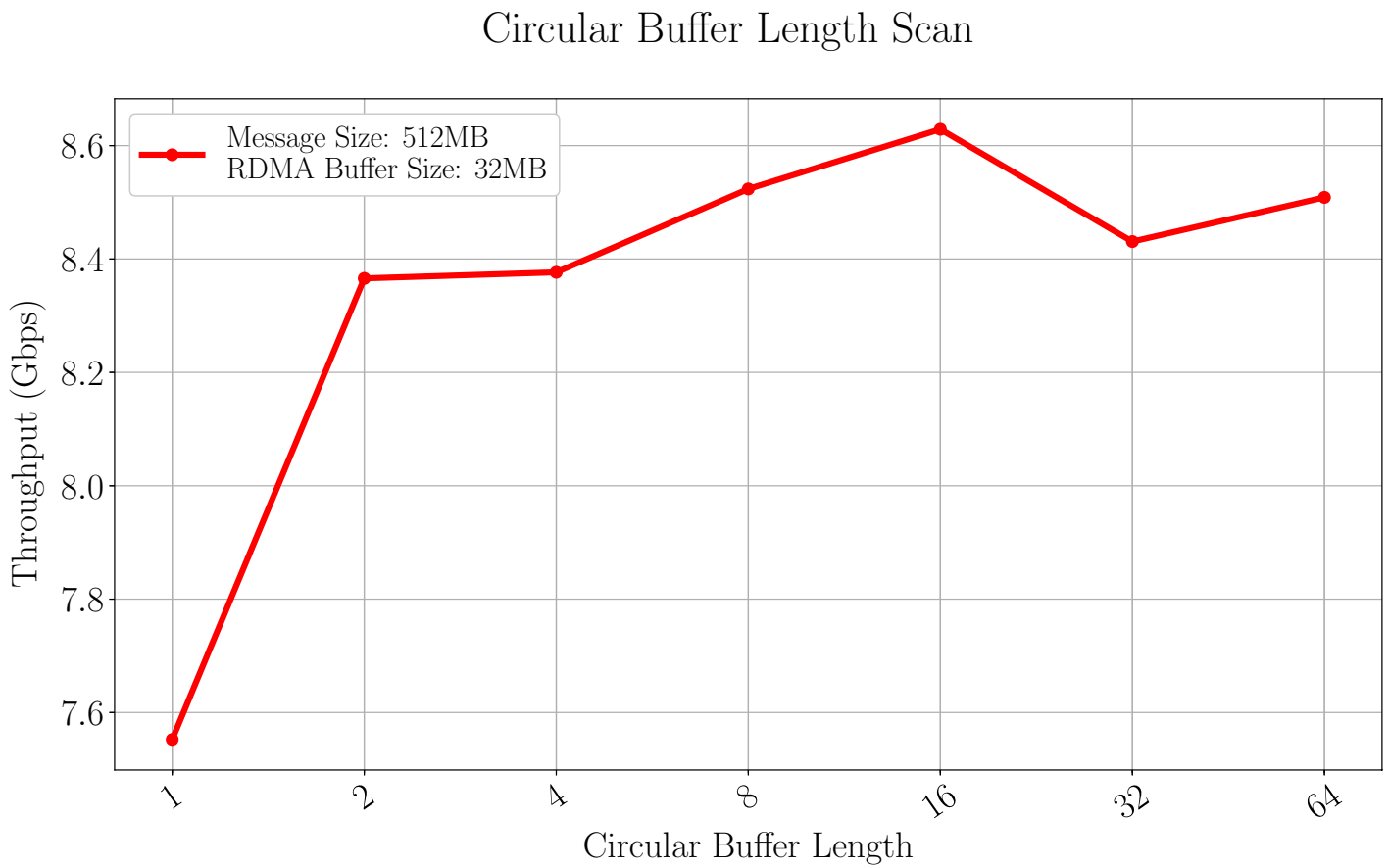
EVALUATION – RDMA BUFFER SIZE SCAN



An **RDMA Buffer** is the smallest possible data size to transmit

The **Circular Buffer** consists of the number of RDMA-enabled blocks assigned to a connection

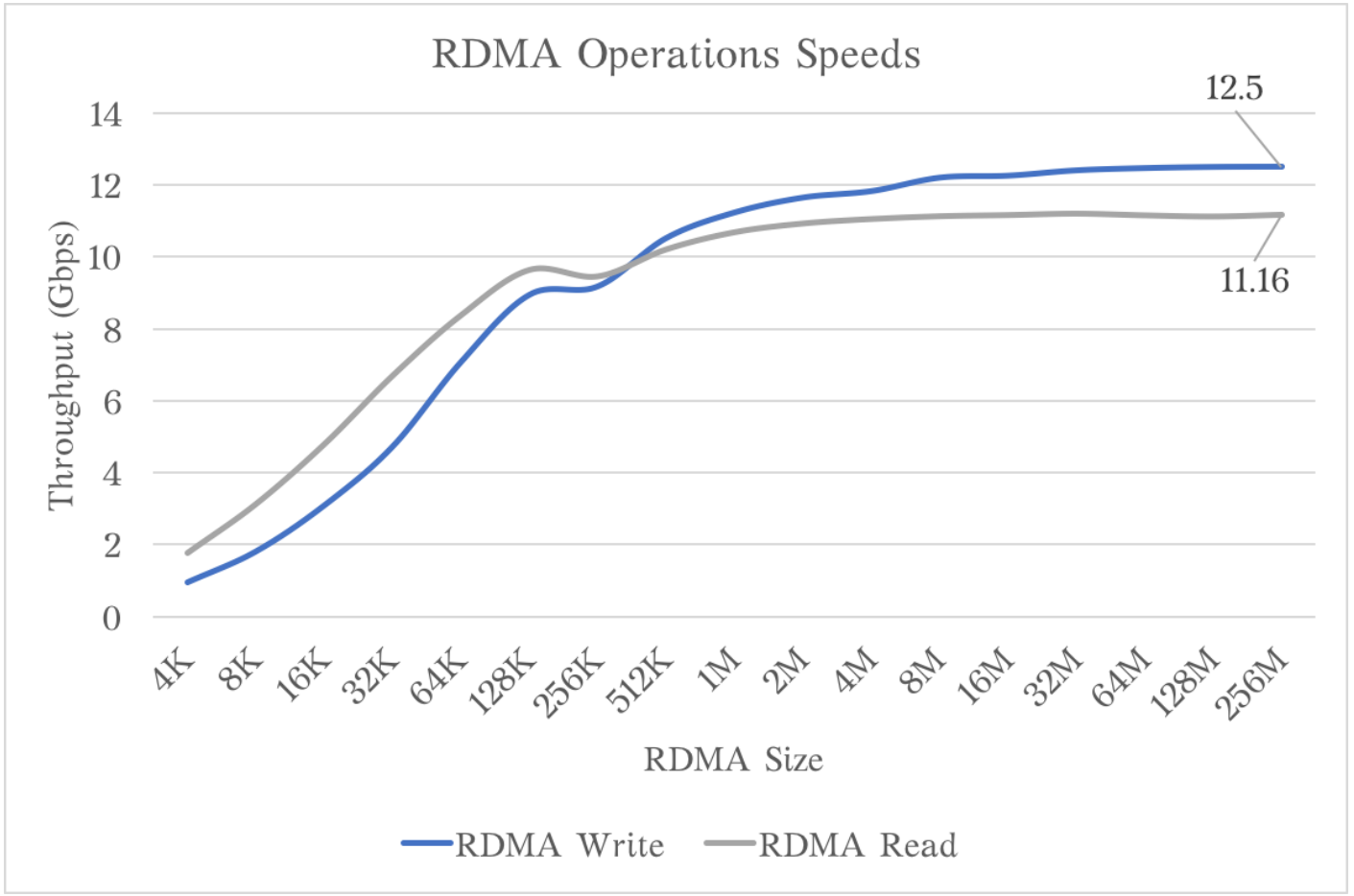
EVALUATION – CIRCULAR BUFFER LENGTH SCAN



An **RDMA Buffer** is the smallest possible data size to transmit

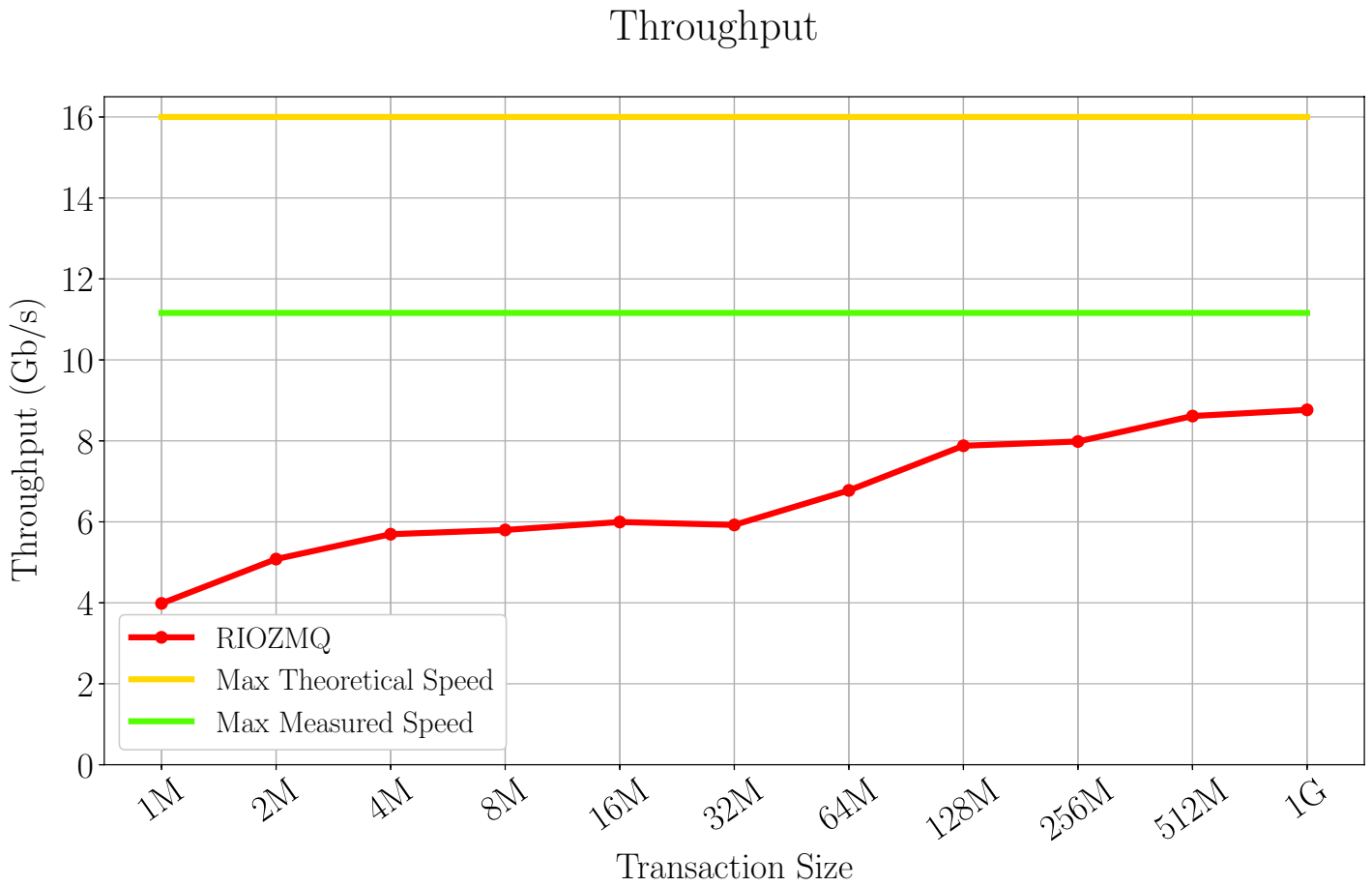
The **Circular Buffer** consists of the number of RDMA-enabled blocks assigned to a connection

EVALUATION – RDMA SPEEDS



- Maximum Measured Speeds
- Around rdma_write() call
- RapidIO <--> PCIe Translations
- Library Overhead

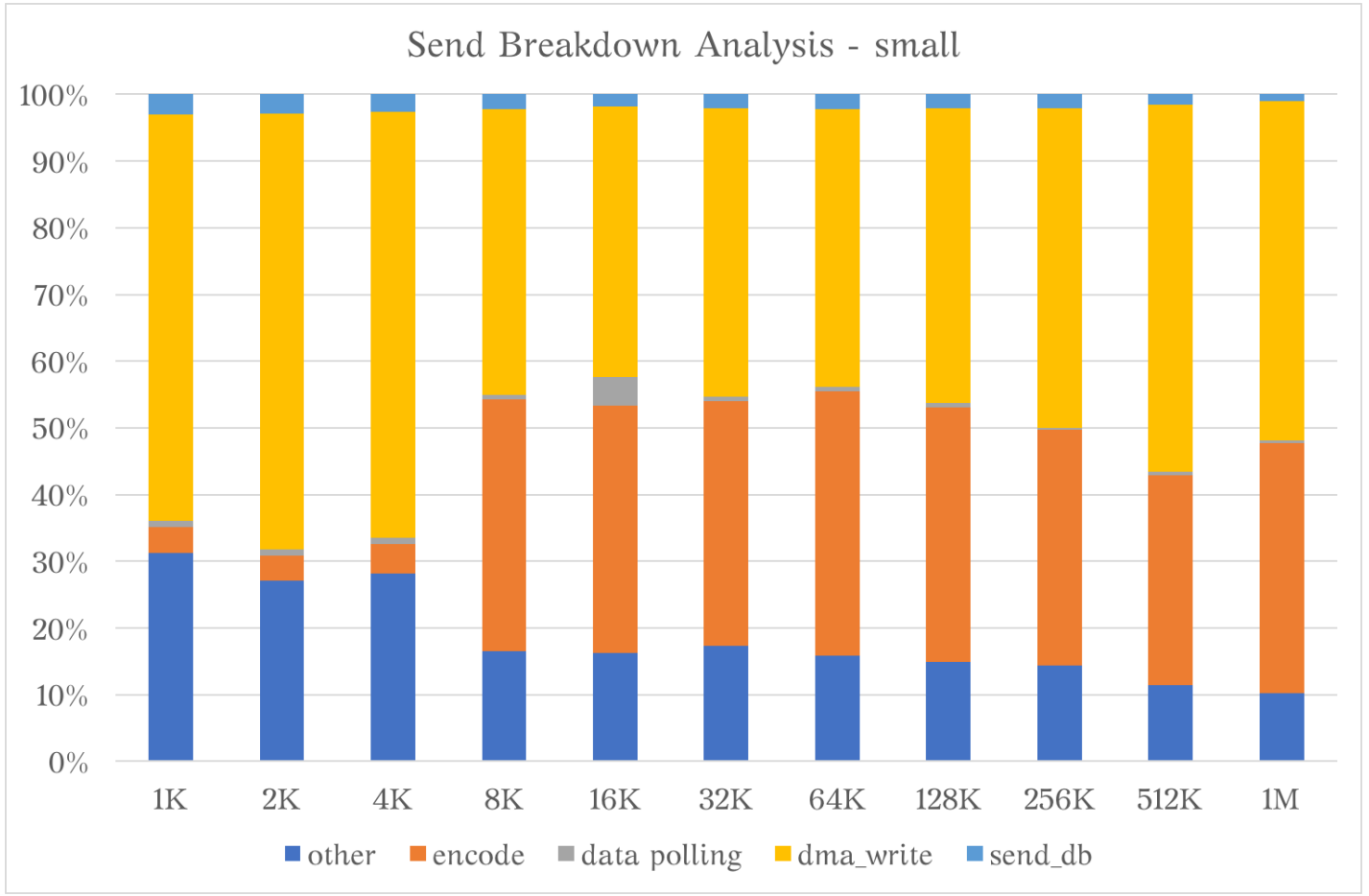
EVALUATION – THROUGHPUT



- RDMA Buffer Size : 32M
- Circular Buffer Length : 16

- Achieved ~75% saturation

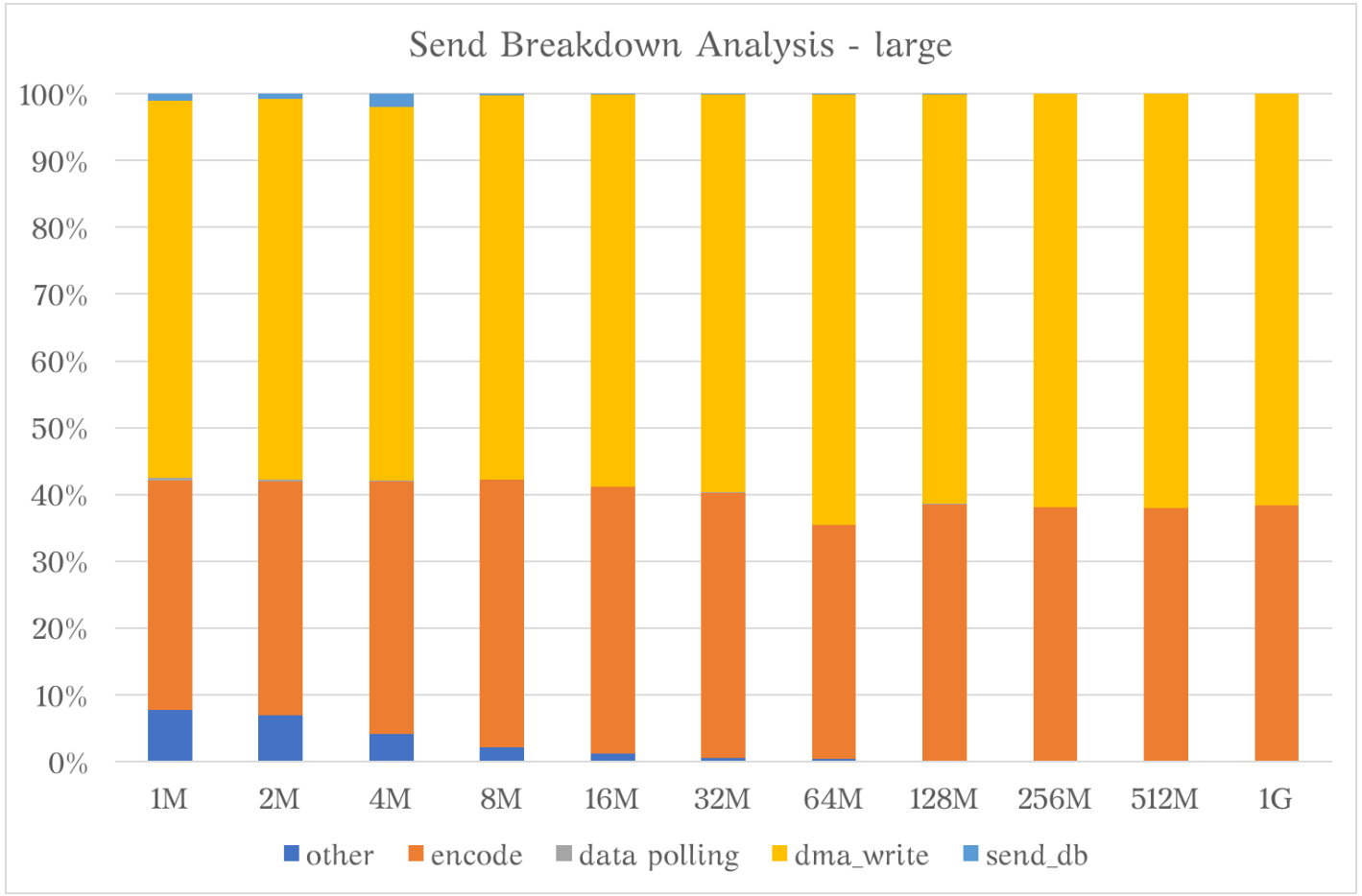
EVALUATION – BREAKDOWN ANALYSIS SEND SMALL



- RDMA Buffer Size : 32M
- Circular Buffer Length : 16

- Main bottleneck dma_write
- Encode also consumes time

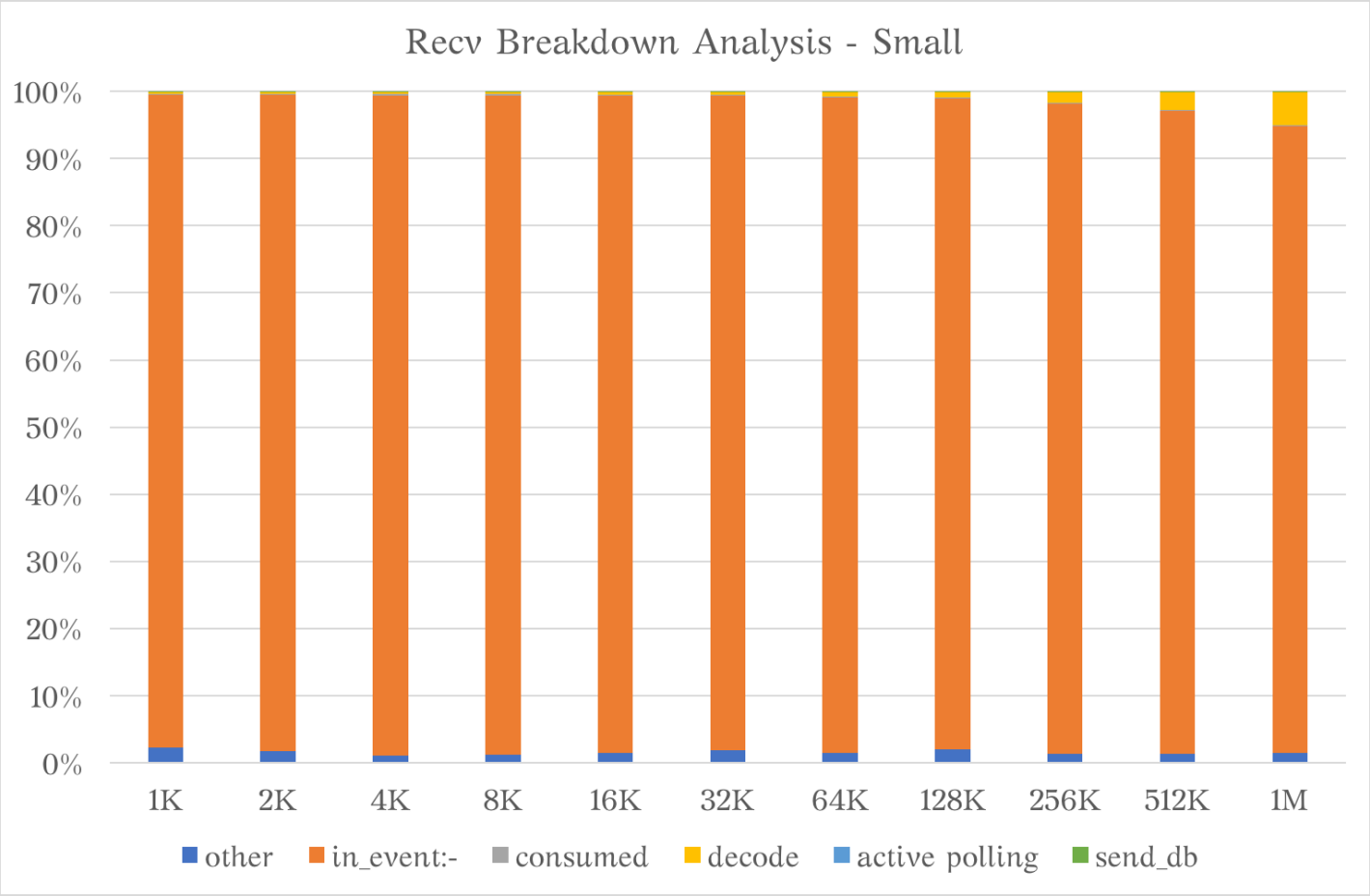
EVALUATION – BREAKDOWN ANALYSIS SEND LARGE



- RDMA Buffer Size : 32M
- Circular Buffer Length : 16

- Main bottleneck dma_write
- Encode also consumes time

EVALUATION – BREAKDOWN ANALYSIS RECV SMALL

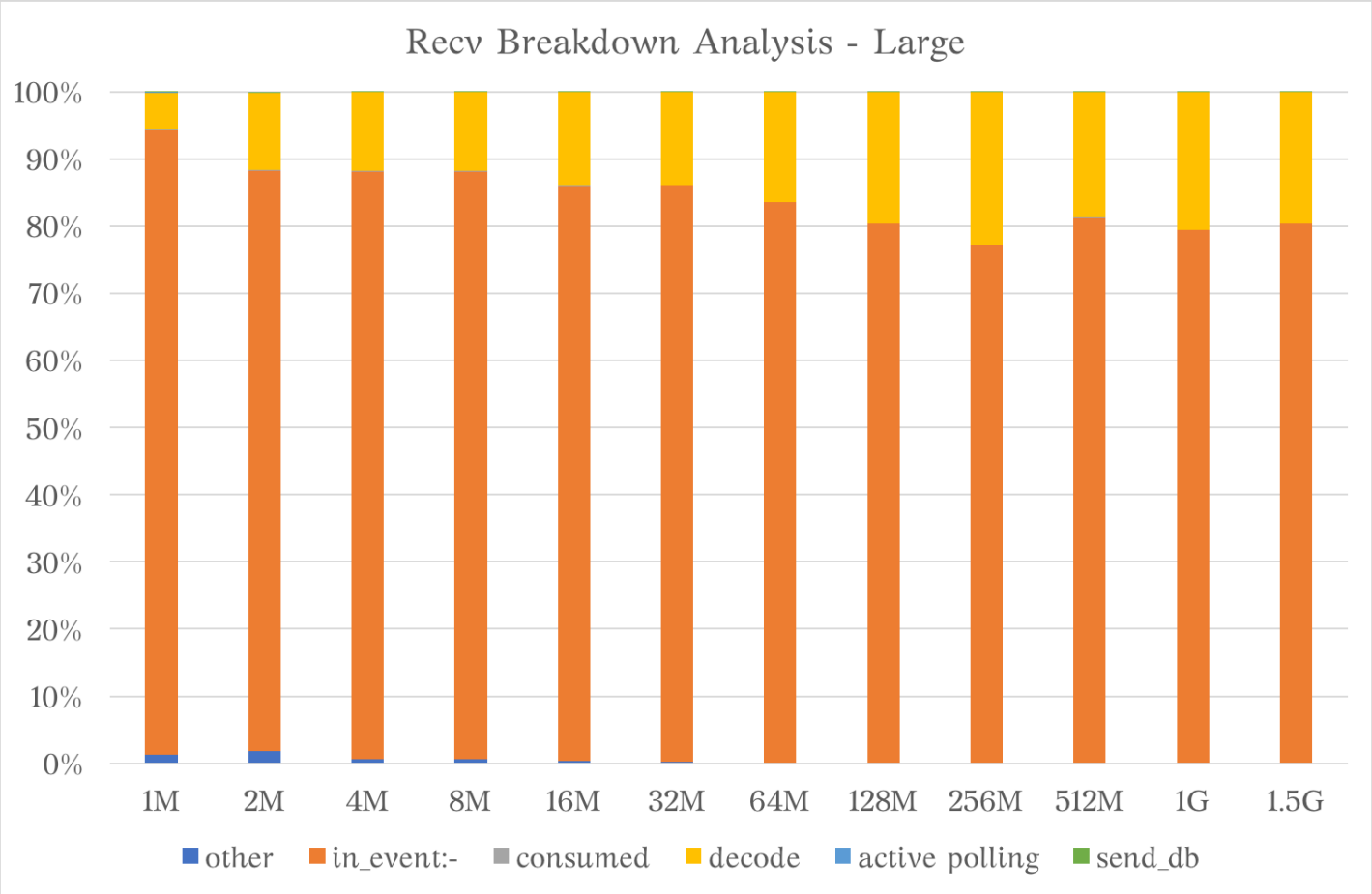


- RDMA Buffer Size : 32M
- Circular Buffer Length : 16

- in_event: between calls
- poll operations
- fd operations
- doorbell handling

- Can't measure asynchronous operations

EVALUATION – BREAKDOWN ANALYSIS RECV LARGE



- RDMA Buffer Size : 32M
- Circular Buffer Length : 16

- in_event: between calls
- poll operations
- fd operations
- doorbell handling

- Can't measure asynchronous operations
- Decode for larger transactions

CONCLUSIONS

- ZeroMQ extended to use the RapidIO transport
- Achieved better latency for small messages compared to TCP/IP
- Designed for use with arbitrary number of nodes
- Use in existing setups by changing the address from tcp* to rio*
- Used RDMA semantics within ZeroMQ
- Same scheme for other RDMA-enabled interconnects
- Work is open-source - can be found at github.com/kostorr/libzmq (soon…)

FUTURE WORK

Implementation

- Optimize circular buffer
- Zero-Copy in the critical path
- Possible removal of file descriptor use

Evaluation

- More extensive breakdown on recv() performance
- Employ on a system with more than 16 nodes
- Run a real-life benchmark on a distributed system

THANK YOU!

THE PROBLEM (1)

Moore's Law

Semiconductor performance increases at an exponential rate

The conjunction of these laws leads to an imbalance, limiting performance

Amdahl's Law – Law of diminishing returns

The performance of a system can only be assessed as the balance between:

- CPU
- Memory Bandwidth
- I/O Performance

New Interconnects Technologies!