SAND2011-2723P

# High-Performance Networking Challenges for Exascale Computing

**Ron Brightwell**
**Scalable System Software**
**Sandia National Laboratories**
**Albuquerque, NM, USA**

**Ohio State University**

**March 4, 2011**

*Sandia is a Multiprogram Laboratory Operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy Under Contract DE-ACO4-94AL85000.*

# Sandia Massively Parallel Systems

1990

nCUBE2

- Sandia's first large MPP
- Achieved Gflops performance on applications

1993

Paragon

- Tens of users
- First periods processing MPP
- World record performance
- Routine 3D simulations
- SUNMOS lightweight kernel

1997

ASCI Red

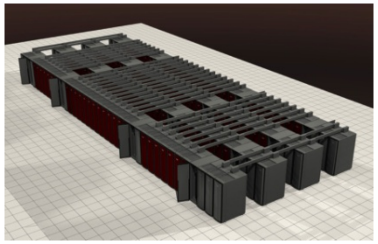
- Production MPP
- Hundreds of users
- Red & Black partitions
- Improved interconnect
- High-fidelity coupled 3-D physics
- Puma/Cougar lightweight kernel

1999

Cplant

- Commodity-based supercomputer
- Hundreds of users
- Enhanced simulation capacity
- Linux-based OS licensed for commercialization
- ~2000 nodes

2004



Red Storm

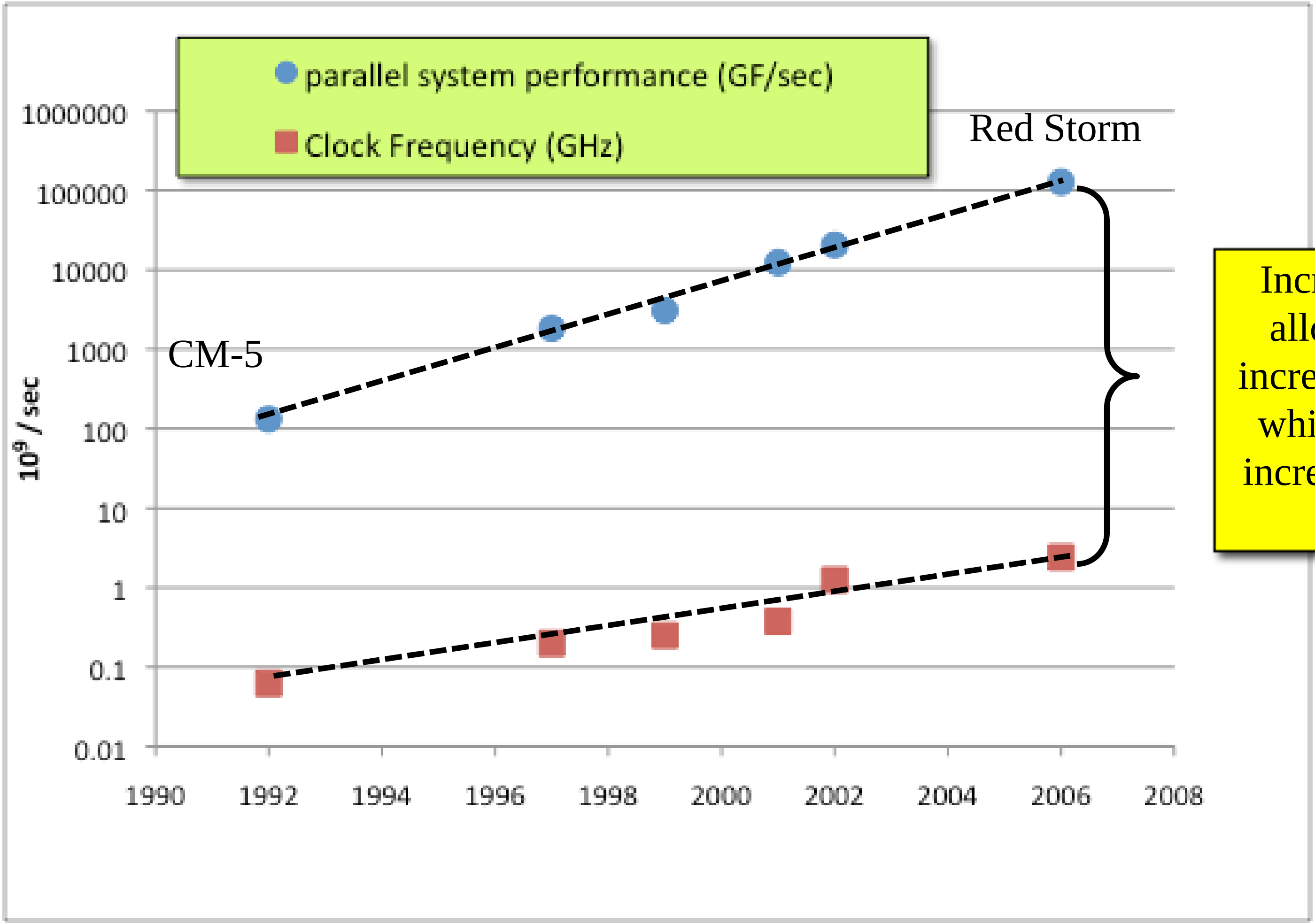
- Prototype Cray XT
- Custom interconnect
- Purpose built RAS
- Highly balanced and scalable
- Catamount lightweight kernel
- Currently 38,400 cores (quad & dual)

Sandia National Laboratories

# Potential Exascale System Targets

| System attributes | 2010 | "2015" | | "2018" | |
|---|---|---|---|---|---|
| System peak | 2 Peta | 200 Petaflop/sec | | 1 Exaflop/sec | |
| Power | 6 MW | 15 MW | | 20 MW | |
| System memory | 0.3 PB | 5 PB | | 32-64 PB | |
| Node performance | 125 GF | 0.5 TF | 7 TF | 1 TF | 10 TF |
| Node memory BW | 25 GB/s | 0.1 TB/sec | 1 TB/sec | 0.4 TB/sec | 4 TB/sec |
| Node concurrency | 12 | O(100) | O(1,000) | O(1,000) | O(10,000) |
| System size (nodes) | 18,700 | 50,000 | 5,000 | 1,000,000 | 100,000 |
| Total Node Interconnect BW | 1.5 GB/s | 20 GB/sec | | 200 GB/sec | |
| MTTI | days | O(1day) | | O(1 day) | |

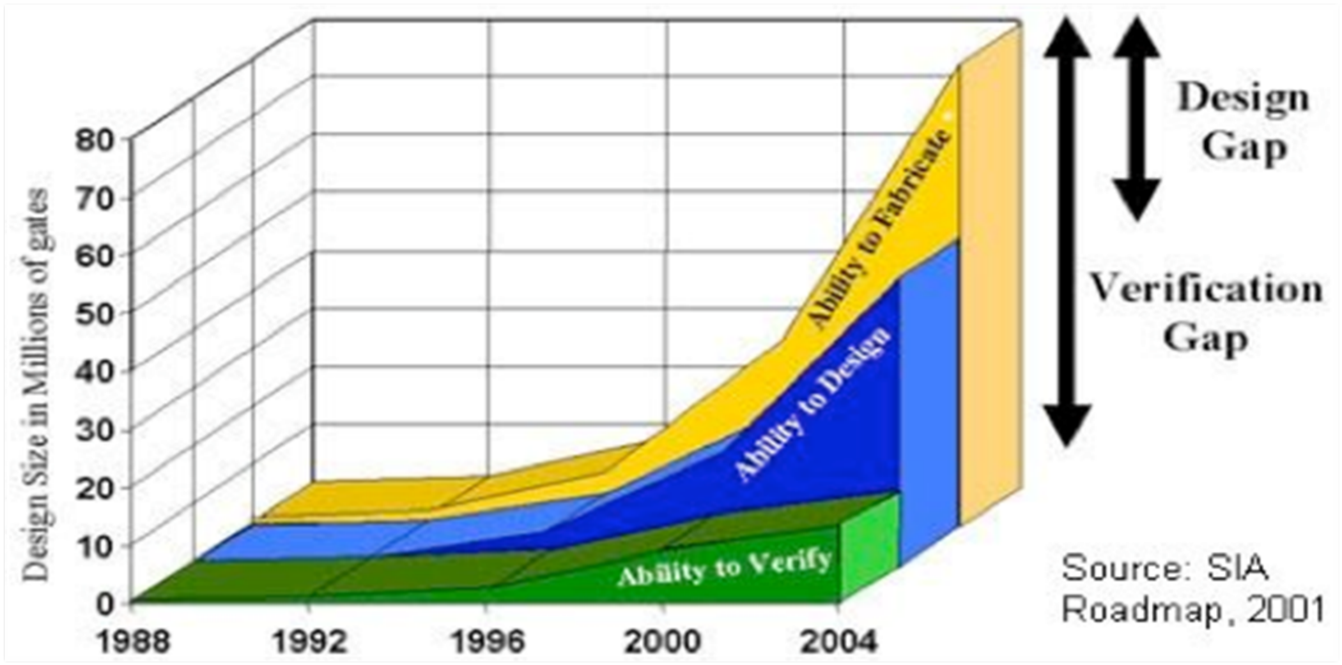
# Concurrency is one key ingredient in getting to exaflop/sec



Increased parallelism allowed a 1000-fold increase in performance while the clock speed increased by a factor of 40

*and power, resiliency, programming models, memory bandwidth, I/O, …*

# Many-core chip architectures are the future

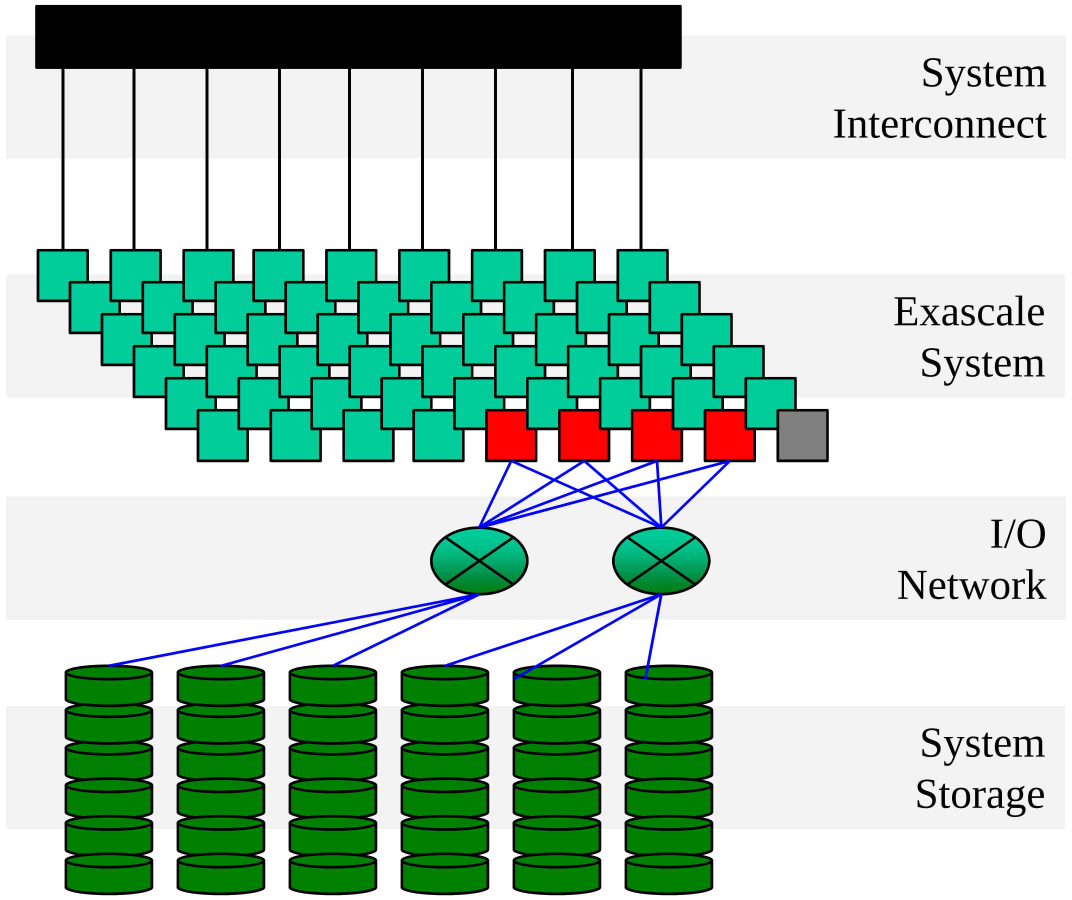


The shift toward increasing parallelism is not a triumphant stride forward based on breakthroughs in novel software and architectures for parallelism … instead it is actually a retreat from even greater challenges that thwart efficient silicon implementation of traditional uniprocessor architectures.
*Kurt Keutzer*

# What are critical exascale technology investments?

- **System power** is a first class constraint on exascale system performance and effectiveness.
- **Memory** is an important component of meeting exascale power and applications goals.
- **Programming model**.  Early investment in several efforts to decide in 2013 on exascale programming model, allowing exemplar applications effective access to 2015 system for both mission and science.
- Investment in exascale **processor design** to achieve an exascale-like system in 2015.
- **Operating System** strategy for exascale is critical for node performance at scale and for efficient support of new programming models and run time systems.
- **Reliability and resiliency** are critical at this scale and require applications neutral movement of the file system (for check pointing, in particular) closer to the running apps.
- **HPC co-design strategy** and implementation requires a set of a hierarchical performance models and simulators as well as commitment from apps, software and architecture communities.

# The high level system design may be similar to petascale systems

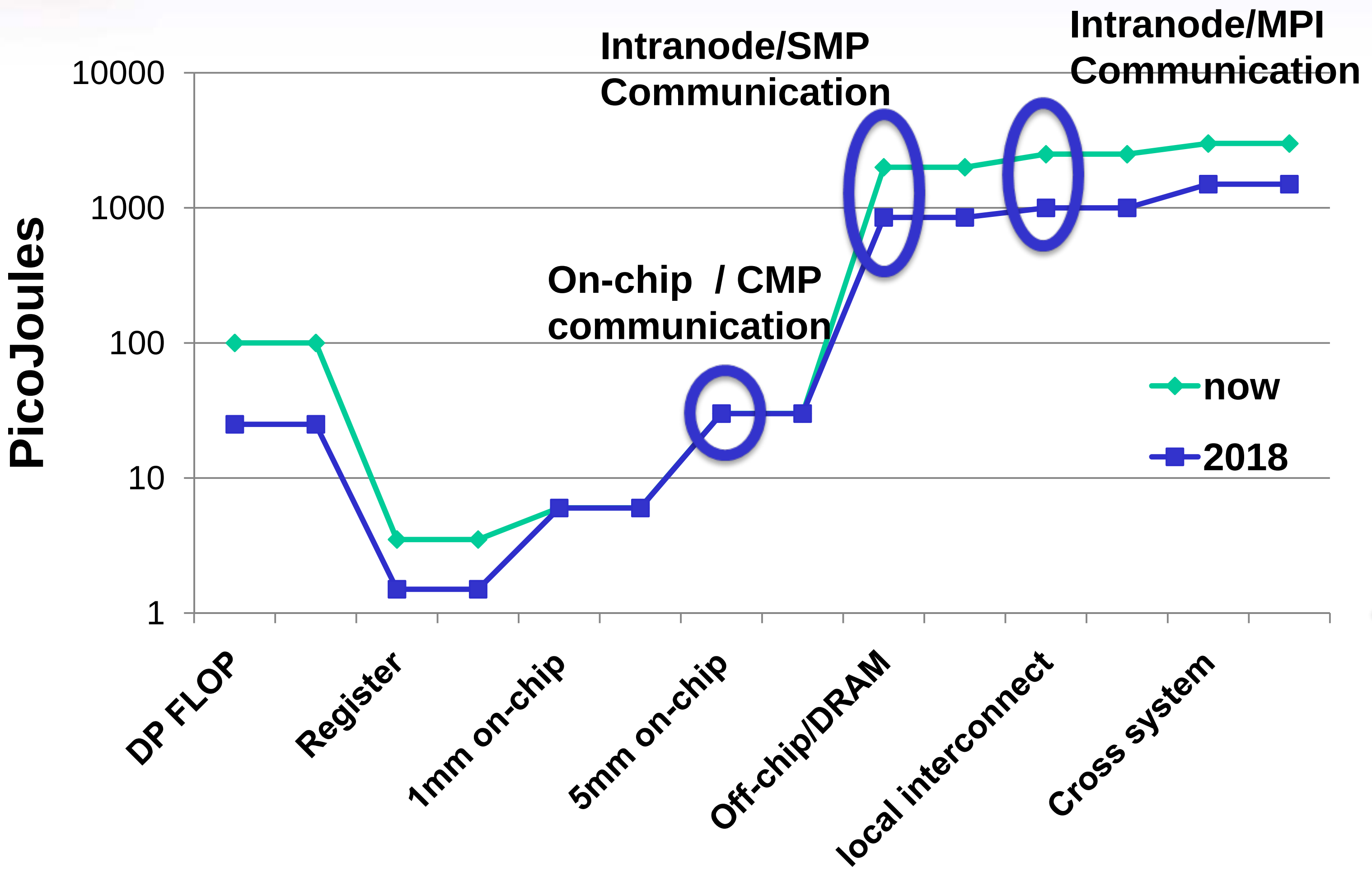


- New interconnect topologies
- Optical interconnect

- 10x – 100x more nodes
- MPI scaling & fault tolerance
- Different types of nodes

- Mass storage far removed from application data

# Investments in architecture R&D and application locality are critical



pJ → kW @Petascale

pJ → MW @Exascale

"The Energy and Power Challenge is the most pervasive … and has its roots in the inability of the [study] group to project any combination of currently mature technologies that will deliver sufficiently powerful systems in any class at the desired levels."
*DARPA IPTO exascale technology challenge report*

# Memory bandwidth and memory sizes will be >> less effective without R&D

- Primary needs are
  - Increase in bandwidth (concurrency can be used to mask latency, viz. Little’s Law)
  - Lower power consumption
  - Lower cost (to enable affordable capacity)
- Stacking on die enable improved bandwidth and lower power consumption
- Modest improvements in latency
- Commodity memory interface

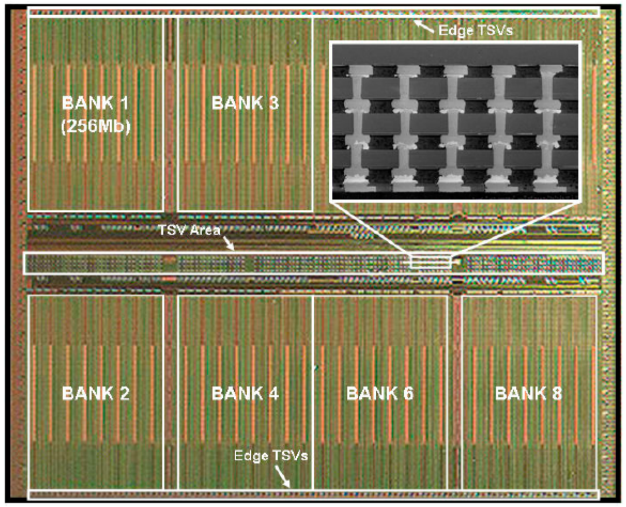


Figure 7.2.7: Die micrograph of the fabricated chip and cross-sectional view of TSVs. The chip size is 10.9×9.0mm².

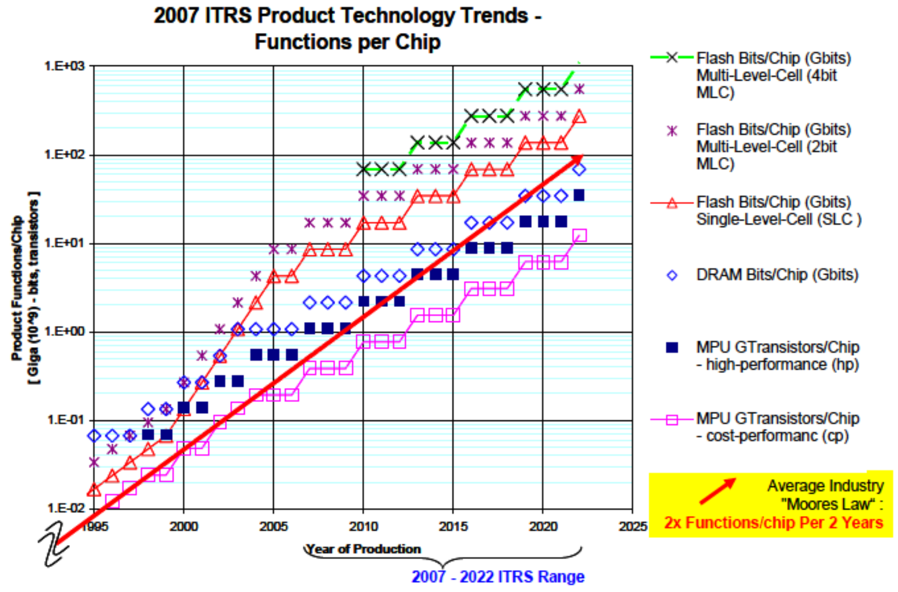


*Figure ORTC2 ITRS Product Function Size Trends: MPU Logic Gate Size (4-transistor); Memory Cell Size [SRAM (6-transistor); Flash (SLC and MLC), and DRAM (transistor + capacitor)]--Updated*

## Factors Driving up the Fault Rate

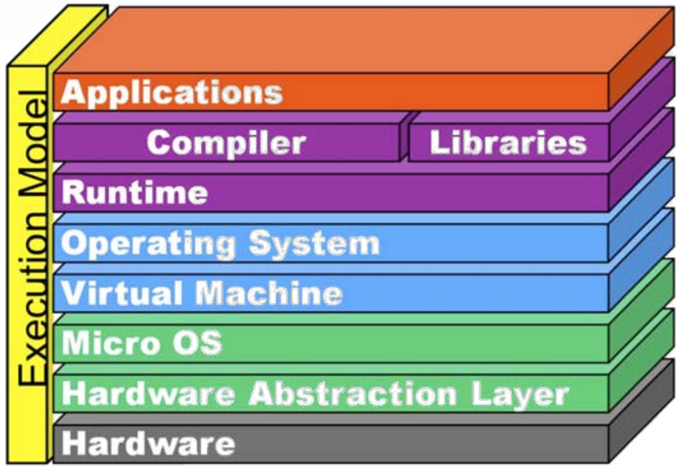
### It’s more than just an increase in components

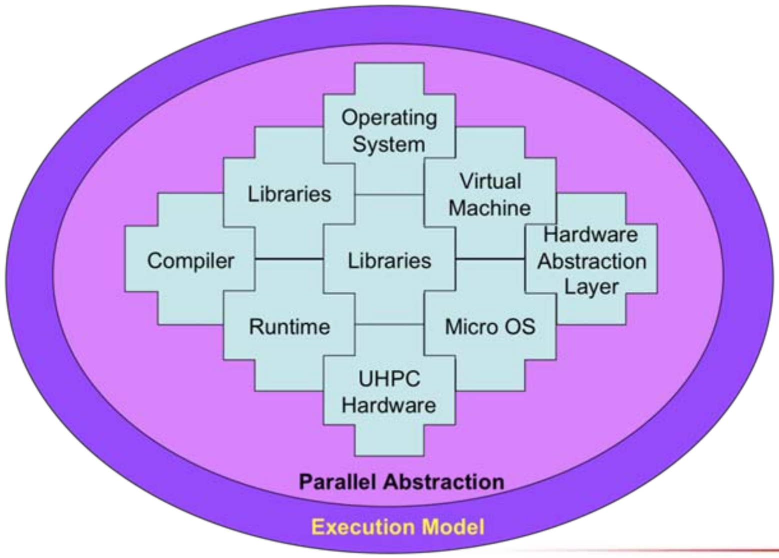
- Number of components both memory and processors will increase by an order of magnitude which will increase hard and soft errors.
- Smaller circuit sizes, running at lower voltages to reduce power consumption, increases the probability of switches flipping spontaneously due to thermal and voltage variations as well as radiation, increasing soft errors
- Power management cycling significantly decreases the components lifetimes due to thermal and mechanical stresses.
- Resistance to add additional HW detection and recovery logic right on the chips to detect silent errors. Because it will increase power consumption by 15% and increase the chip costs.
- Heterogeneous systems make error detection and recovery even harder, for example, detecting and recovering from an error in a GPU can involve hundreds of threads simultaneously on the GPU and hundreds of cycles in drain pipelines to begin recovery.
- Increasing system and algorithm complexity makes improper interaction of separately designed and implemented components more likely.
- Number of operations (1023 in a week) ensure that system will traverse the tails of the operational probability distributions.

National Nuclear Security Administration
Sandia National Laboratories

# System Software as Currently Implemented Is Not Suitable for Exascale

- Barriers
  - System management SW not parallel
  - Current OS stack designed to manage only O(10) cores on node
  - Unprepared for industry shift to NVRAM
  - OS management of I/O has hit a wall
  - Not prepared for massive concurrency
- Technical Focus Areas
  - Design HPC OS to partition and manage node resources to support massively concurrency
  - I/O system to support on-chip NVRAM
  - Co-design messaging system with new hardware to achieve required message rates
- Technical gaps
  - 10X: in affordable I/O rates
  - 10X: in on-node message injection rates
  - 100X: in concurrency of on-chip messaging hardware/software
  - 10X: in OS resource management

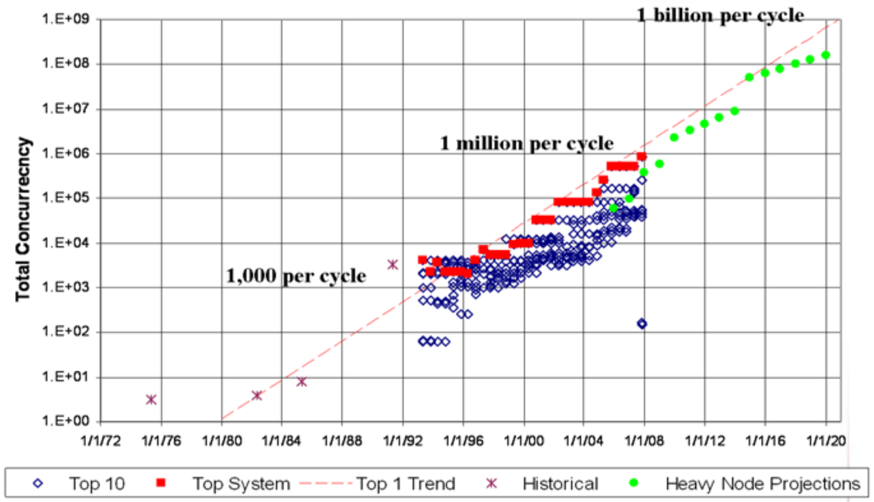




Software challenges in extreme scale systems, *Sarkar, 2010*

## Programming Models and Environments Require Early Investment

- **Barriers:** Delivering a large-scale scientific instrument that is productive and fast.
  - O(1B) way parallelism in Exascale system
  - O(1K) way parallelism in a processor chip
    - Massive lightweight cores for low power
    - Some “full-feature” cores lead to heterogeneity
  - Data movement costs power and time
    - Software-managed memory (local store)
  - Programming for resilience
  - Science goals require complex codes



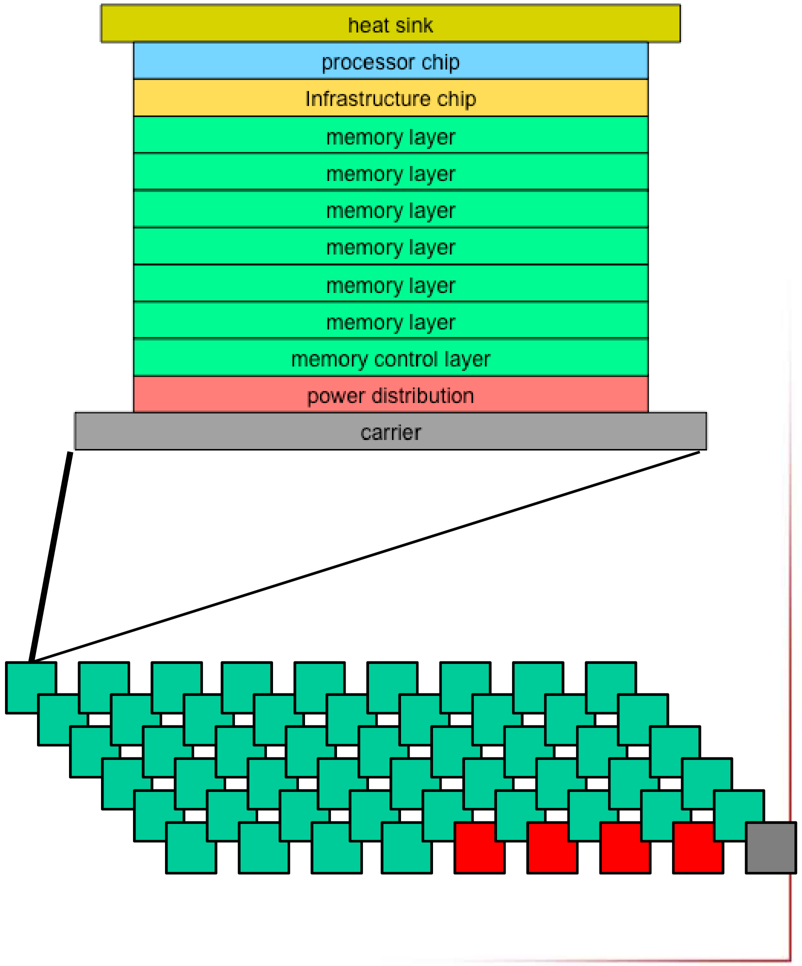
**How much parallelism must be handled by the program?**
From Peter Kogge (on behalf of Exascale Working Group), “Architectural *Challenges* at the Exascale Frontier”, June 20, 2008

- **Technology Investments**
  - Extend inter-node models for scalability and resilience, e.g., MPI, PGAS (includes HPCS)
  - Develop intra-node models for concurrency, hierarchy, and heterogeneity by adapting current scientific ones (e.g., OpenMP) or leveraging from other domains (e.g., CUDA, OpenCL)
  - Develop common low level runtime for portability and to enable higher level models

- **Technical Gap:**
  - No portable model for variety of on-chip parallelism methods or new memory hierarchies
  - Goal: Hundreds of applications on the Exascale architecture; Tens running at scale

# Programming Model Approaches

- Hierarchical approach (intra-node + inter-node)
  - Part I: Inter-node model for communicating between nodes
    - MPI scaling to millions of nodes: Importance high; risk low
    - One-sided communication scaling: Importance medium; risk low
  - Part II: Intra-node model for on-chip concurrency
    - Overriding Risk: No single path for node architecture
    - OpenMP, Pthreads: High risk (may not be feasible with node architectures); high payoff (already in some applications)
    - New API, extended PGAS, or CUDA/OpenCL to handle hierarchies of memories and cores: Medium risk (reflects architecture directions); Medium payoff (reprogramming of node code)
- Unified approach: single high level model for entire system
  - High risk; high payoff for new codes, new application domains



## Exascale Networking Challenges

## Challenge Areas for HPC Networks

- The traditional “big three”
  - Bandwidth
  - Latency
  - Message Rate (Throughput)

- Other important areas for “real applications” versus benchmarks
  - Allowable Outstanding Messages
  - Host memory bandwidth usage
  - Noise (threading, cache effects)
  - Synchronization
  - Progress
  - Topology
  - Reliability

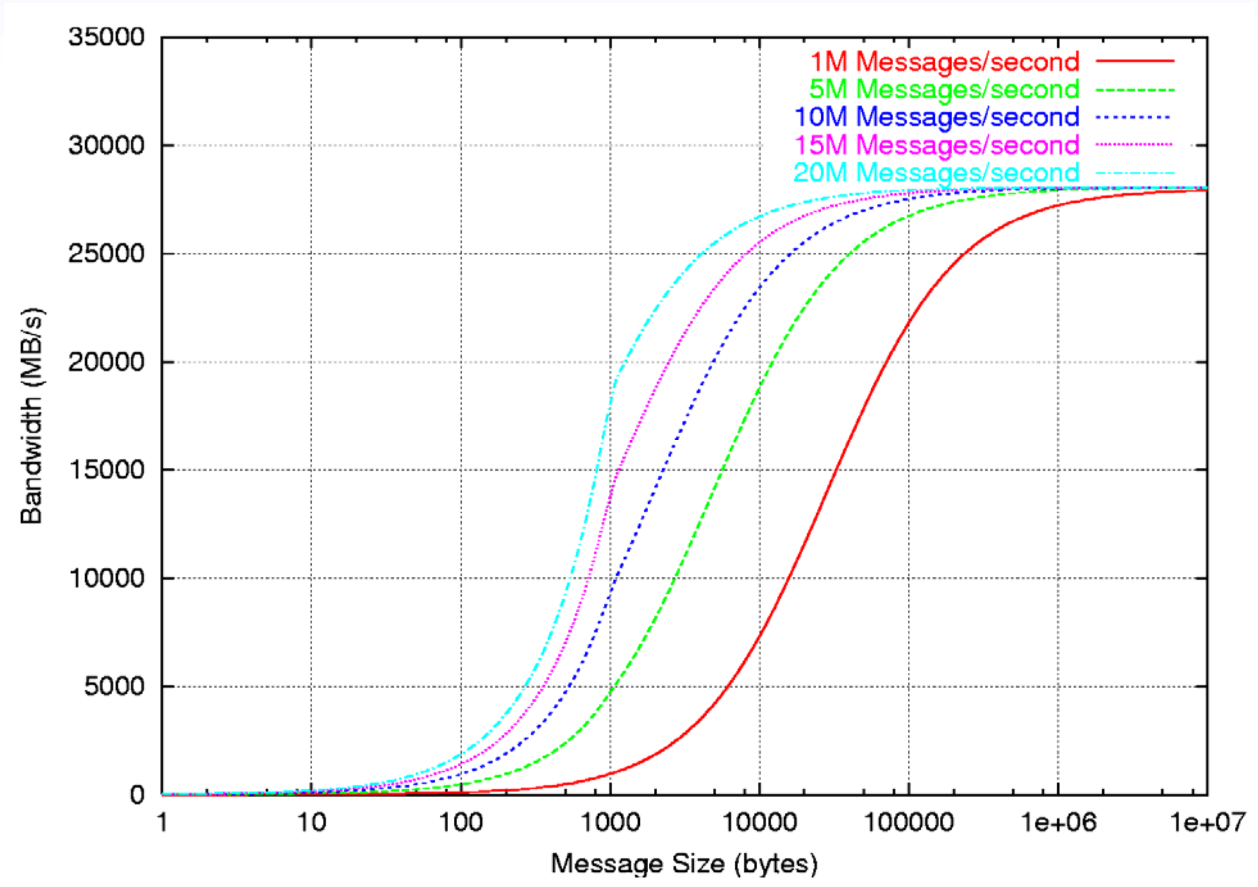
## MPI Will Likely Persist Into Exascale Era

- Number of network endpoints will increase significantly (5-50x)
- Memory and power will be dominant resources to manage
  - Networks must be power and energy efficient
  - Data movement must be carefully managed
  - Memory copies will be very expensive
- Impact of unexpected messages must be minimized
  - Eager protocol for short messages leads to receive-side buffering
  - Need strategies for managing host buffer resources
  - Flow control will be critical
  - N-to-1 communication patterns will (continue to be) disastrous
- Must preserve key network performance characteristics
  - Latency
  - Bandwidth
  - Message rate (throughput)

National Nuclear Security Administration

Sandia National Laboratories

# High Message Throughput is Vital



Message rate determines the minimum message size needed to saturate the available network bandwidth

# Current Flow Control Strategies Not Sufficient

- Credit-based
  - Limit number of outstanding send operations
  - Used credits are replenished implicitly or explicitly
  - Effectiveness limited to N-to-1 scenario
  - Potential performance penalty for well-behaved applications
- Acknowledgment-based
  - Receiver explicitly confirms receipt of every message
  - Significant per-message performance penalty
    - Round trip acknowledgment doubles latency
  - Performance penalty for well-behaved applications
- Local copying (bounce buffer) mitigates latency penalty
- Both strategies limit message rate and effective bandwidth
- Flow control implemented at user-level inside MPI library
- Network transport usually has its own flow control mechanism
  - No mechanism for back pressure from host resources to network

## Applications Must Become More Asynchronous

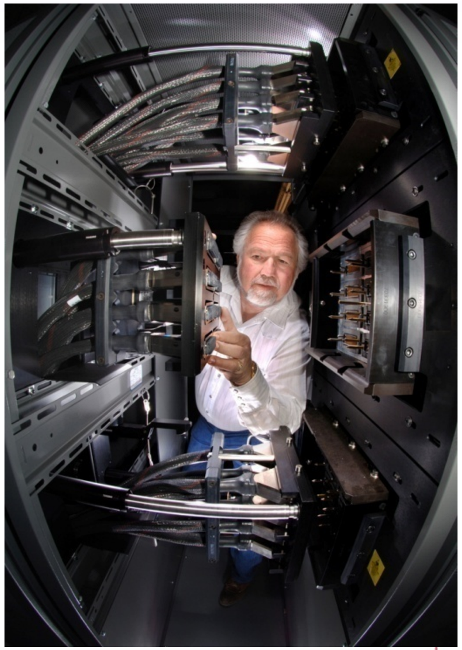
- Applications cannot continue to be bulk synchronous
  - Overhead of global synchronization will limit scaling
  - Global synchronization increases susceptibility to noise
- One-sided communication requires explicit synchronization
- Network API must provide asynchronous operations and progress
  - Data movement must be independent of host activity
- Active Messages
  - Polling is fundamental to all AM
  - Progress only when nothing else to do
  - Polling memory for message reception is inefficient
  - Needs hardware support to integrate message arrival with thread invocation
- Run-time systems will also need to communicate
  - Need to communicate evolving state of the system
  - Need a common portable API
  - Using TCP OOB connection will be infeasible

## Resiliency Will Impact Network API

- Network will need to expose errors to enable recovery
- Applications and system components will have different resiliency requirements
  - Reachability errors must be handled by run-time services
  - Graceful degradation may be appropriate for some applications
- May need OOB mechanism for recognizing network failures
  - AM or event-driven API would be ideal
  - Hardware support for network-level protection
    - RAS system invoking OS via network messages

## Topology

- No single network topology is best for all applications
- Meshes
  - Advantages: high local bandwidth, low wiring complexity, ability to easily add nodes (no distinct steps in expandability curve)
  - Disadvantages: high maximum latency, low global bandwidth
  - Works well for physical simulations which tend to talk nearest neighbor
- Trees
  - Advantages: high global bandwidth, low global latency
  - Disadvantages: high wiring complexity, lower local bandwidth, discrete steps in network topology (limiting expandability)
  - Works well for random accesses patterns and apps with lots of global communication


Red Storm
Red/Black Switching

- Hierarchical/Hybrid networks
  - Tend to inherent the strengths and weaknesses of the building blocks
- Network routers must be designed to allow different topologies with the same silicon

## Comparison of Theoretical Modern Networks

| | Dragonfly* | 3D Torus | Fat Tree |
|---|---|---|---|
| Router Radix | 64 | 7 | 32 |
| Notes | 2 links per node, group size of 256, 128 groups | 32x32x32 | Full bisection, based on 512 port switches |
| Number of Switches | 4096 | 32768 | 7168 |
| Bisection BW (Bi-directional) | 80 TB/s | 91 TB/s | 160 TB/s |
| Node BW (Bi-directional) | 9.8 GB/s | 44.6 GB/s | 9.8 GB/s |
| Max Hops | 5 | 48 | 7 |

Assumptions: 2.5Tbit/s total switch bandwidth, 32k nodes

*John Kim, William J. Dally, Steve Scott, Dennis Abts, "Technology-Driven, Highly-Scalable Dragonfly Topology." In Proceedings of the 35th International Symposium on Computer Architecture (ISCA '08).

## High Radix: The Wave of the Future?

- High radix routers are definitely in our future
  - Available bandwidth on a chip continues to increase
  - Core clock speeds are not increasing
  - But, we have a few more years where low radix networks are feasible, and for certain workloads possibly favorable
    - Still need to study the energy/performance trade-offs of newer networks

- New work into hybrid network topologies is a good start in how to best use high radix routers
  - However, networks seem to be optimized for global random traffic
  - Need a better feel for how these types of topologies will impact performance of traditional scientific applications
  - What is the right radix??

## System View is Vital for Energy

- Interconnect is not an isolated system and only accounts for a portion of the total system power
- Higher interconnect power can actually lead to lower energy
- Understanding the true impact of the interconnect trade-offs can lead to more energy efficient systems

- Areas were wrong assumptions can lead to less energy efficient solutions
  - Microbenchmarks
  - High Radix Routers
  - Tight Integration

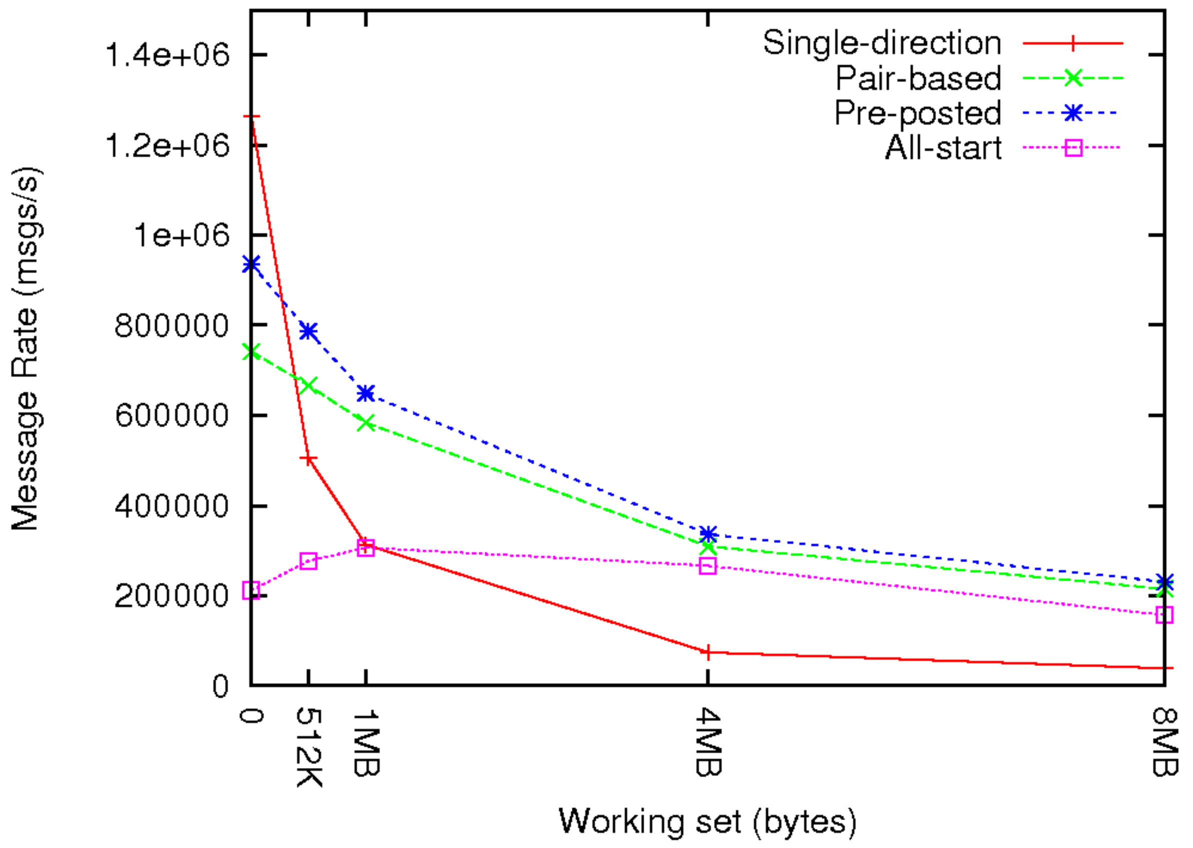
## Microbenchmarks

- Fallacy:  Optimizing interconnects and MPI implementations to microbenchmarks will necessarily improve application performance (or at least won't hurt it).
- Any optimization that reduces performance without reducing power will lead to less energy efficient system
  – Conversely, any optimization that increases performance without increasing power will lead to more energy efficient systems

- Removing useful advanced features to improve NetPipe latency and bandwidth will not generally translate to improved application performance (and may actually make it worse)
- Coalescing identical zero-byte messages will not help any application of which I am aware
- Measuring message rate under ideal conditions does not provide useful information about message rate achievable by an application

## Sandia Message Throughput Benchmark

- Measures message rate using communication patterns mimicking those of scientific applications
  - Simulation of computation/communication phase with variable working set sizes (compute stage modeled by touching data to invalidate some portion of cache)
  - **Each MPI rank both sends and receives**
  - **Variable number of peers**

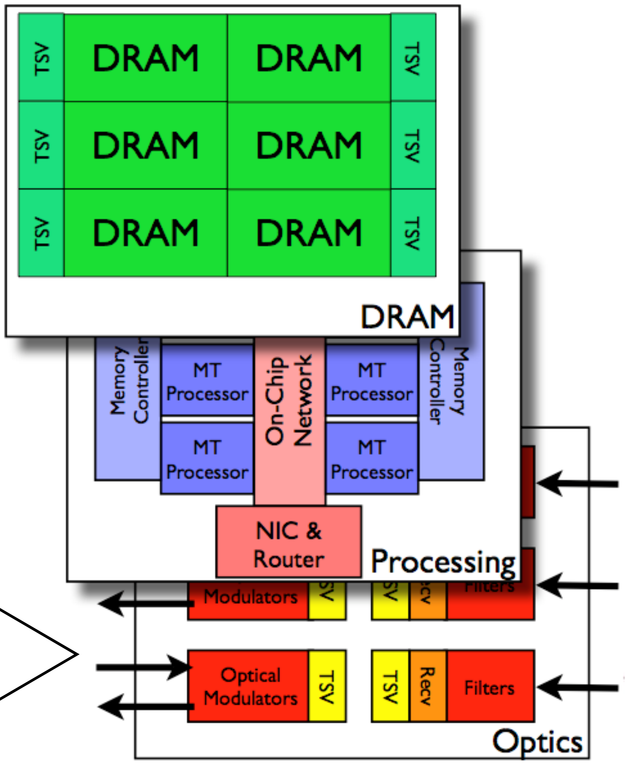


# Tight Integration

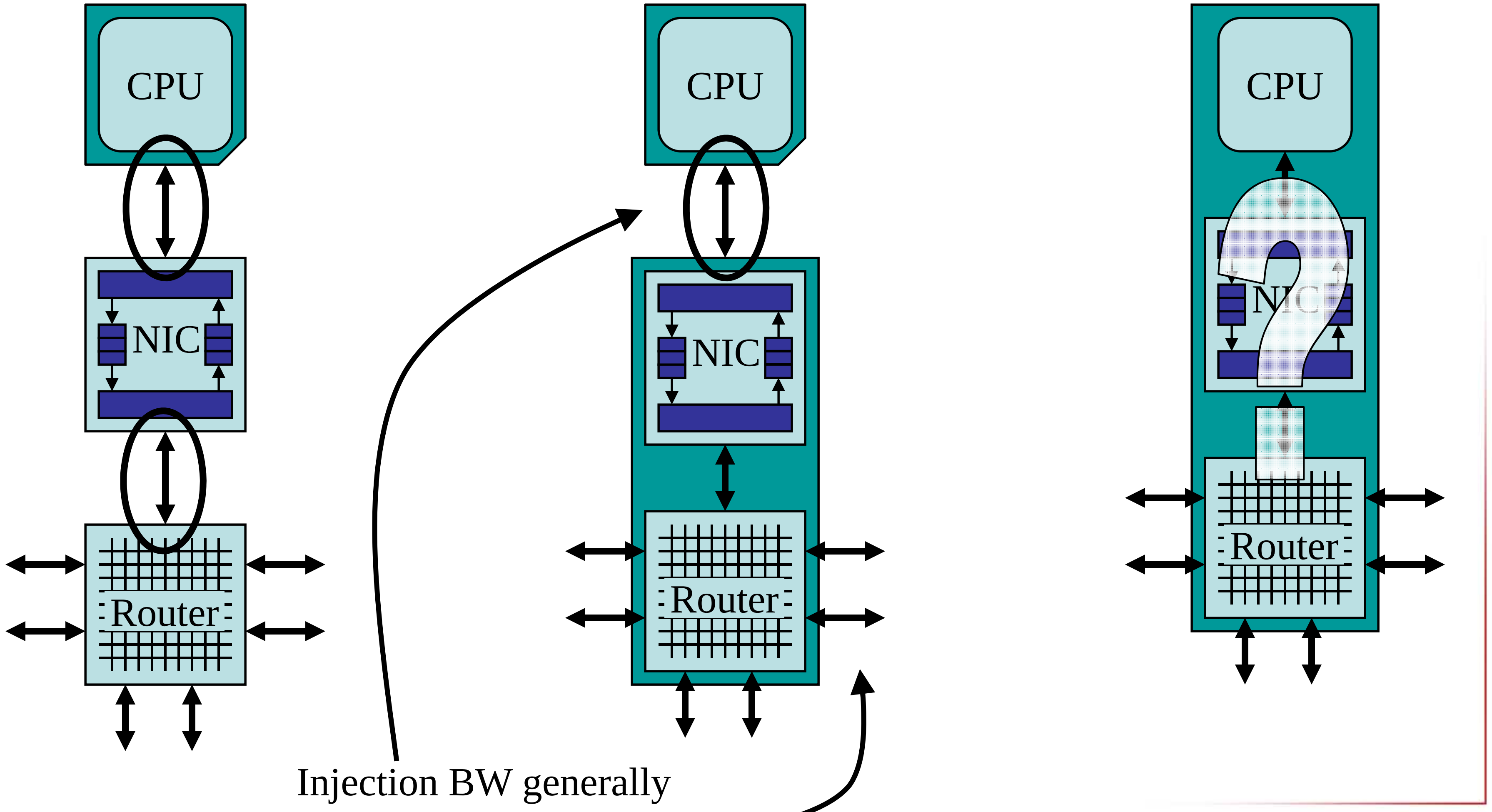
- Fallacy: Integrating router, NIC and processor onto same package magically provides access to more usable interconnect bandwidth.

- **The problem is not with tight integration, it's understanding how that integration affects the balance of the system**

Million dollar question:
How much of this bandwidth is genuinely available to the processors?

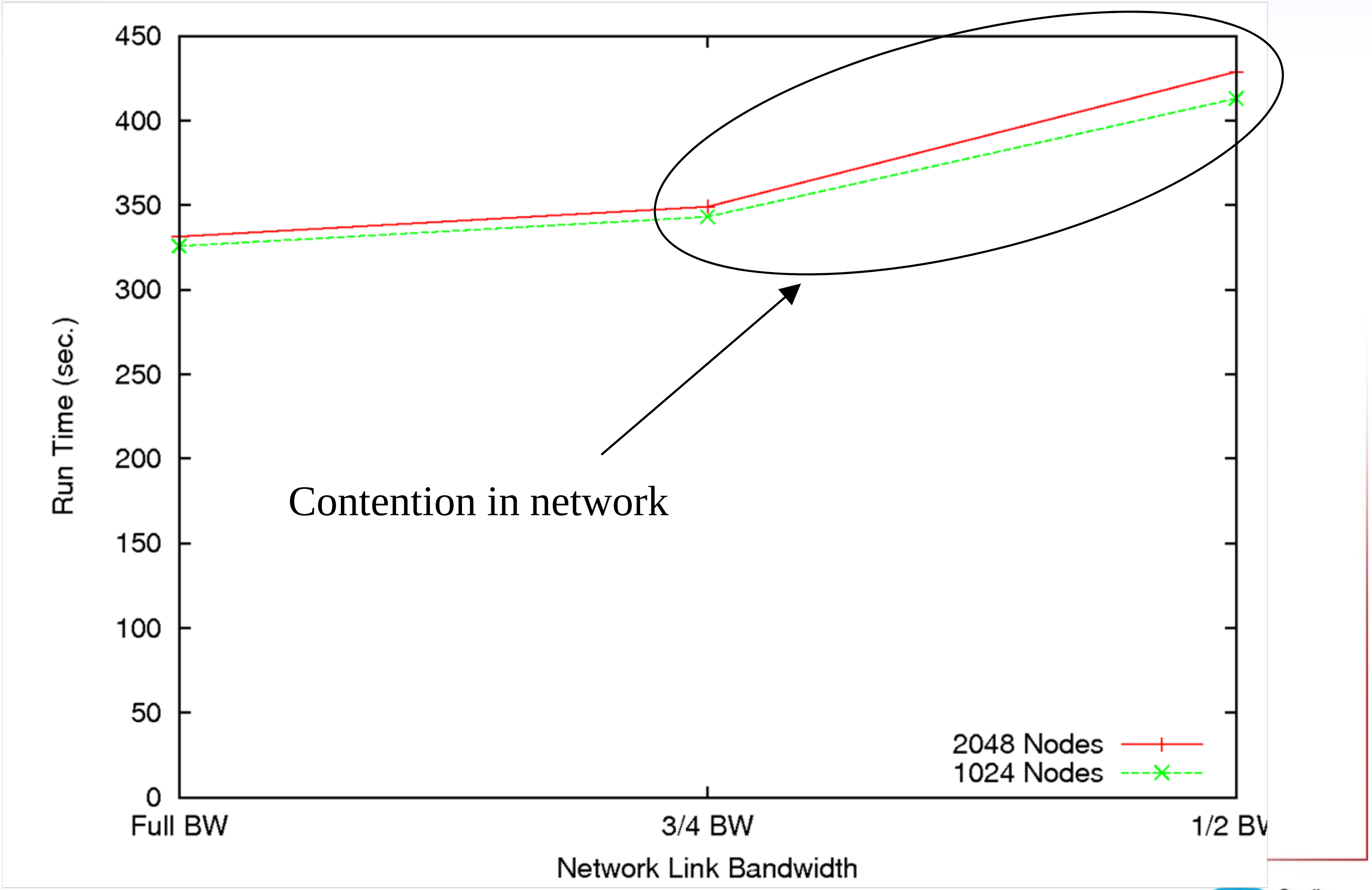


# Tight Integration and Injection Bandwidth



Injection BW generally
smaller than link BW.
2:1 ratio for Red Storm

## CTH Example:  Network Contention



## Realizations

- Most of the bandwidth into the stack is not typically usable by the cores in the stack
  - Most of the bits flowing in are not destined for that node
  - Most of the bandwidth going out is already being used by other traffic

- Expect to get the same utilization as when the router is off-chip

- Two approaches in the end:
  - Marketing approach:  Count all the bandwidth
    - Detrimental impact on application performance/energy due to poor balance
  - Technical approach:  Properly balance the system based on usable bandwidth

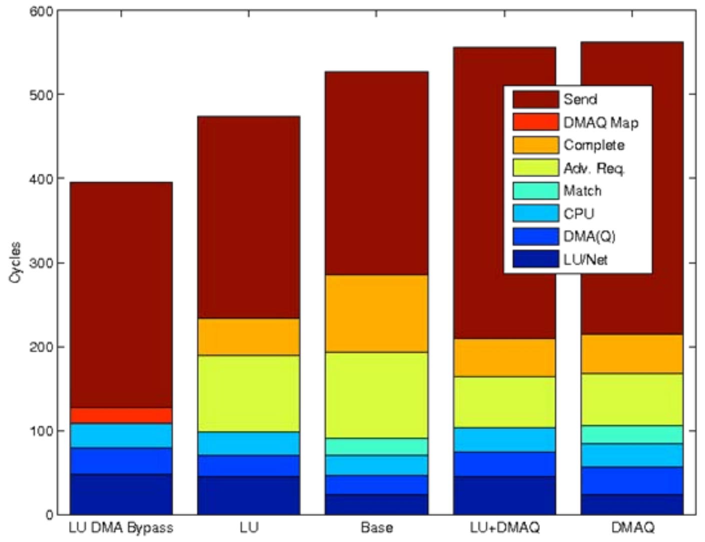
## Thoughts

- It's not necessarily about power, it's about energy to solution
  - Higher power systems can actually lead to lower energy to solution
  - When peak power is a limiter, likely better off with a "smaller", more balanced system, than a larger, unbalanced system
- It's not about peak FLOPS/Watt, it's about the percent of peak that can be sustained
  - We pay an energy penalty for unused operations
  - With rising awareness of energy-efficient computing, FLOPS/Watt threatens to become the new HPL
- This talk is on interconnects, but other areas are equally important
  - What's the application impact of slower, less complex cores
    - Can in-order cores use wide floating-point units?
    - Can applications scale to the dramatically increased number of cores?
- Components should be designed with a system view and understanding of the application needs

## NIC Architecture Co-Design

# NIC Architecture Co-design

- Prevailing architectural constraints have driven many applications to highly bursty communication patterns
- In a power constrained world this trend will be unsustainable due to inefficient use of the system interconnect
- Design Goal: Produce a NIC architecture that enables overlap through high message rates and independent progress

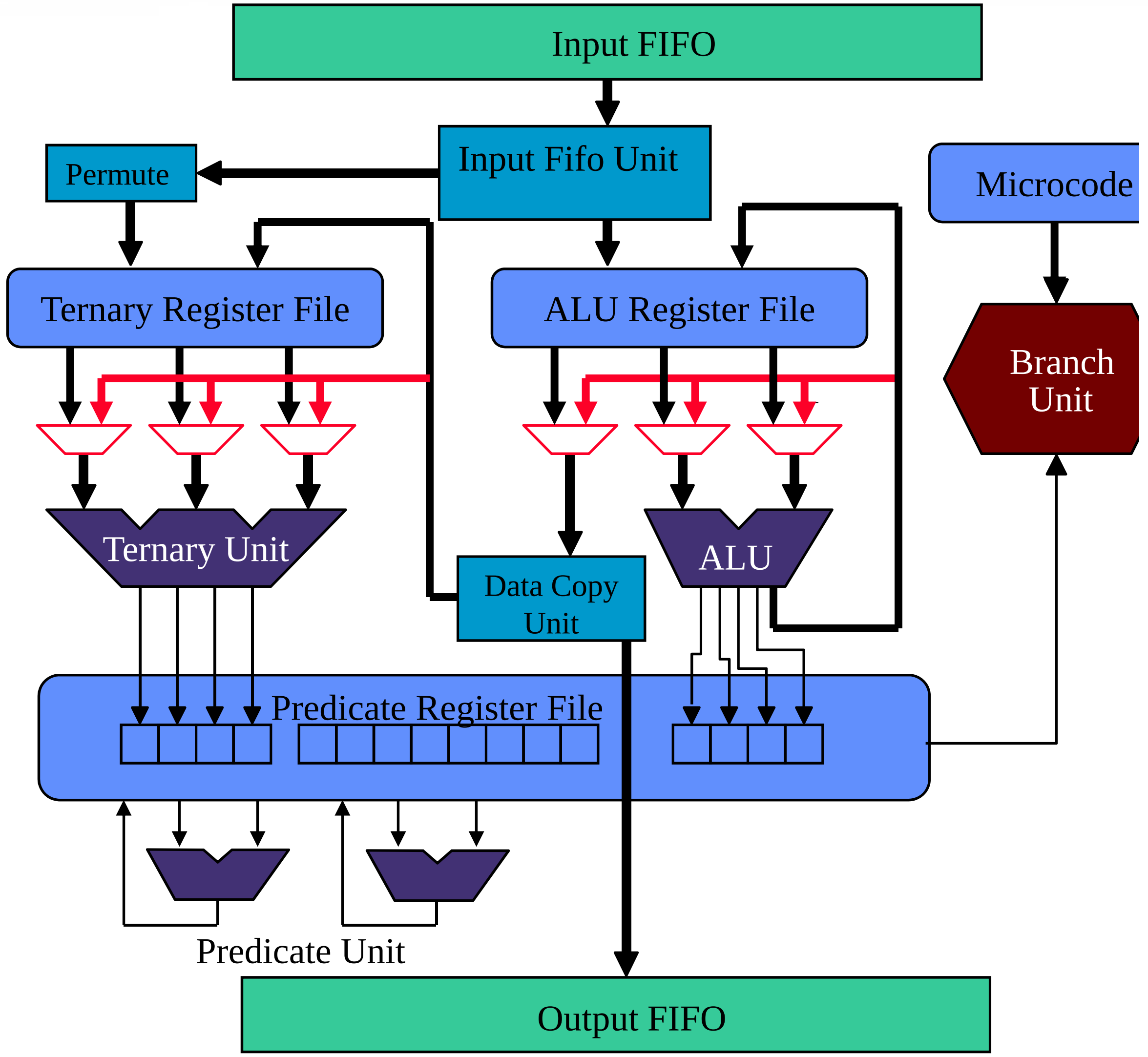


- Using simulation, NIC hardware & software and host driver software were simultaneously profiled for various architecture choices
- Trade-offs:
  - Which architectural features provide performance advantages
  - What software bottlenecks need to be moved to hardware
  - Which functions can be left to run on NIC CPU or in the host driver
- Next step: rework applications (or portions) to take advantage of the new features and provide feedback for more architectural improvements

## Network Interface Controller

- Power will be number one constraint for exascale systems
- Current systems waste energy
  - Using host cores to process messages is inefficient
  - Only move data when necessary
  - Move data to final destination
    - No intermediate copying due to network
- Specialized network hardware
  - Atomic operations
  - Match processing
- Addressing and address translation
  - Virtual address translation
    - Avoid registration cache
  - Logical node translation
    - Rank translation on a million nodes
- Hardware support for thread activation on message arrival

## Match Unit Architecture

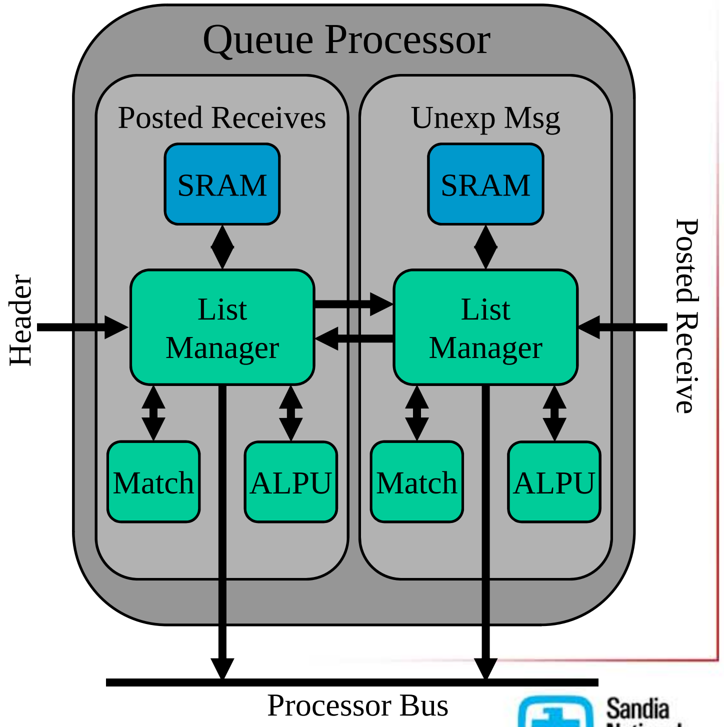


Architecture Drivers

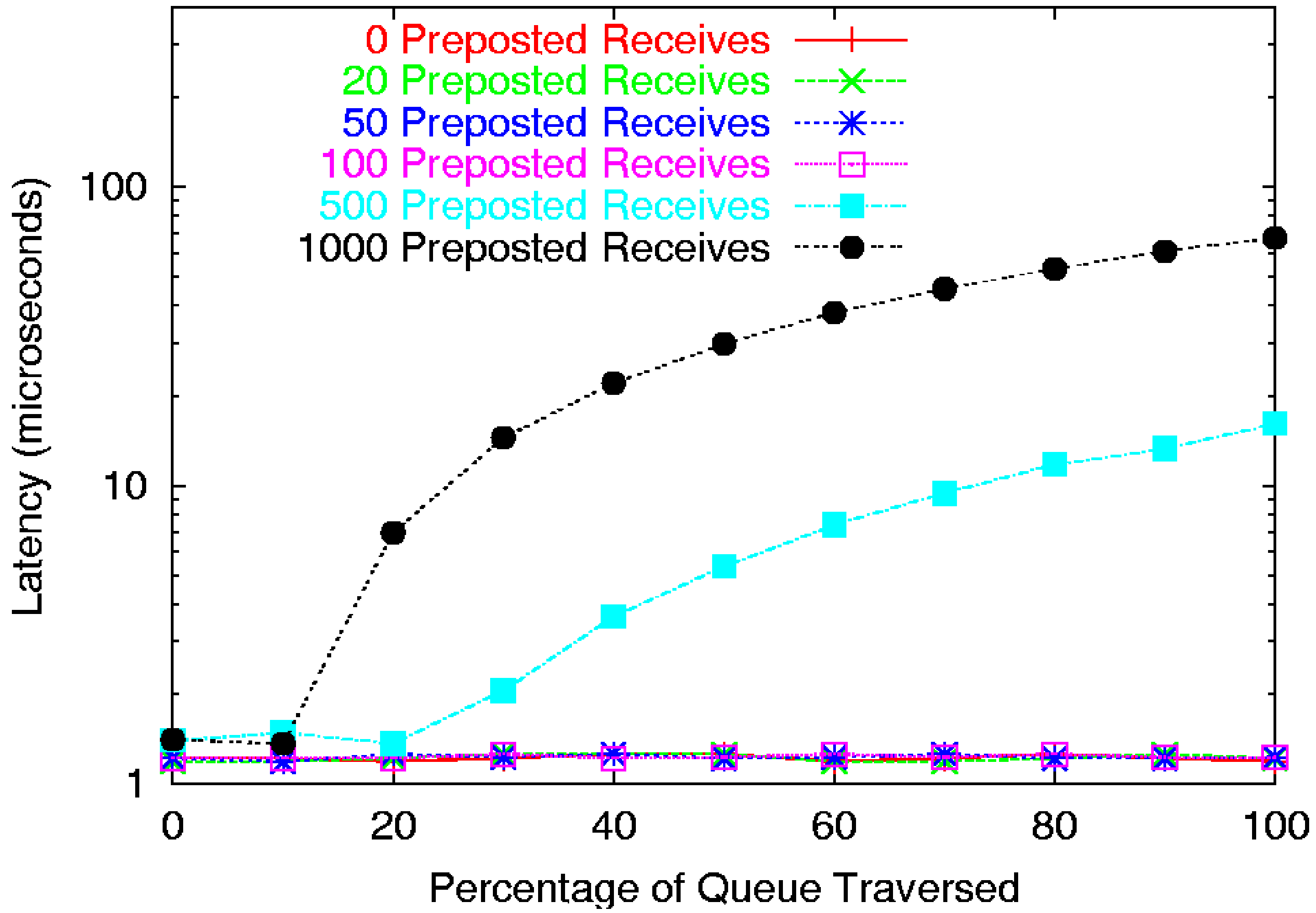
- **High throughput**
  - **3 stage pipeline**
- **Irregular data alignment**
  - **SIMD operation**
  - **Permute units**
- **Program Consistency**
  - **Forwarding in datapath**
  - **Read before write in register file**

## High Message Throughput Challenges

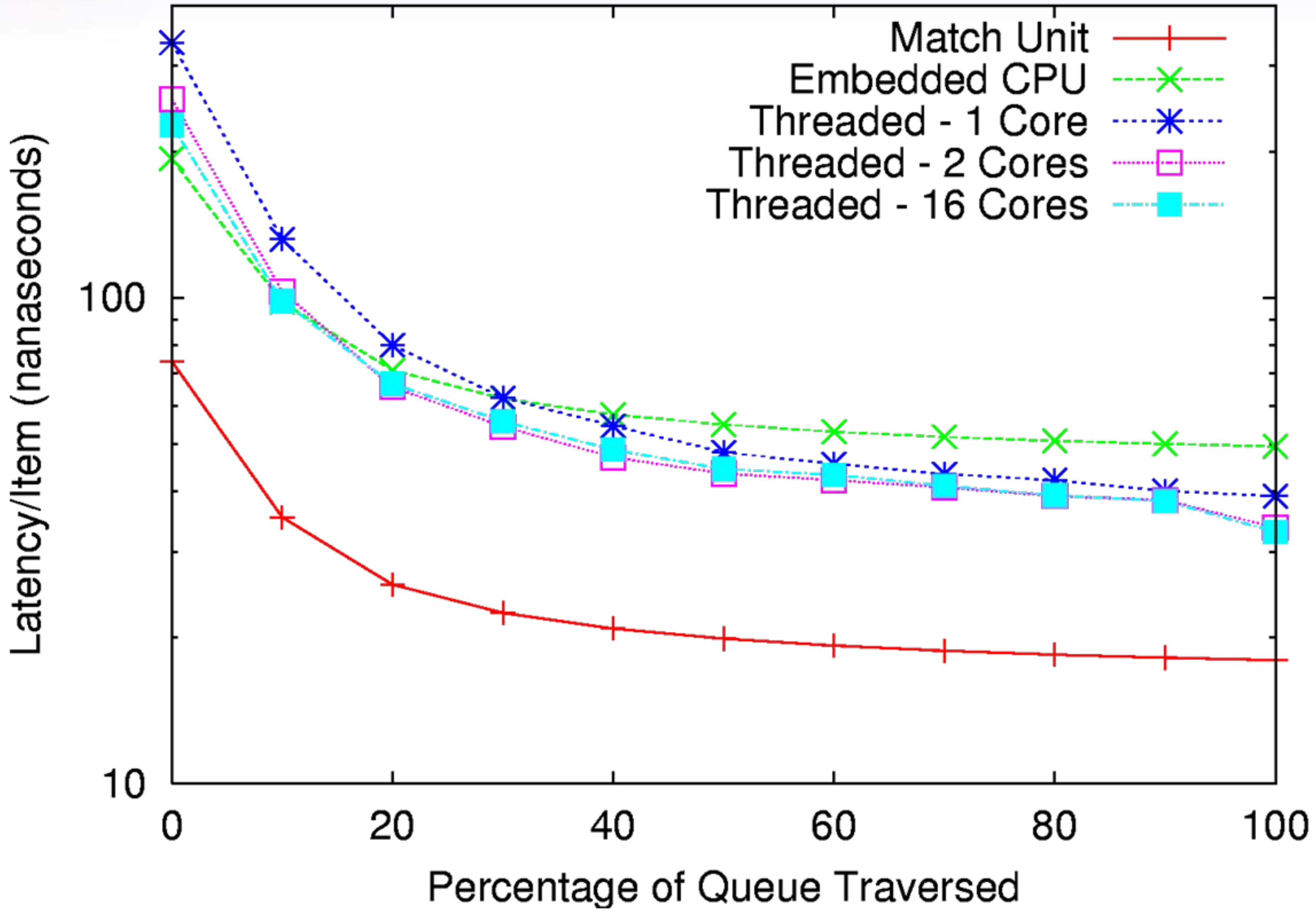
- 20M messages per second implies a new message every 50ns
- Significant constraints created by MPI semantics
- On-load approach
  - Roadblocks
    - Host general purpose processors are inefficient for list management
    - Caching (a cache miss is 70-120ns latency)
      - Microbenchmarks are cache friendly, real life is not
  - Benefits
    - Easier & cheaper
- Off-load approach
  - Roadblocks
    - Storage requirements on NIC
    - NIC embedded processor is worse at list management (than the host processor)
  - Benefits
    - Opportunity to create dedicated hardware
    - Macroscale pipelining



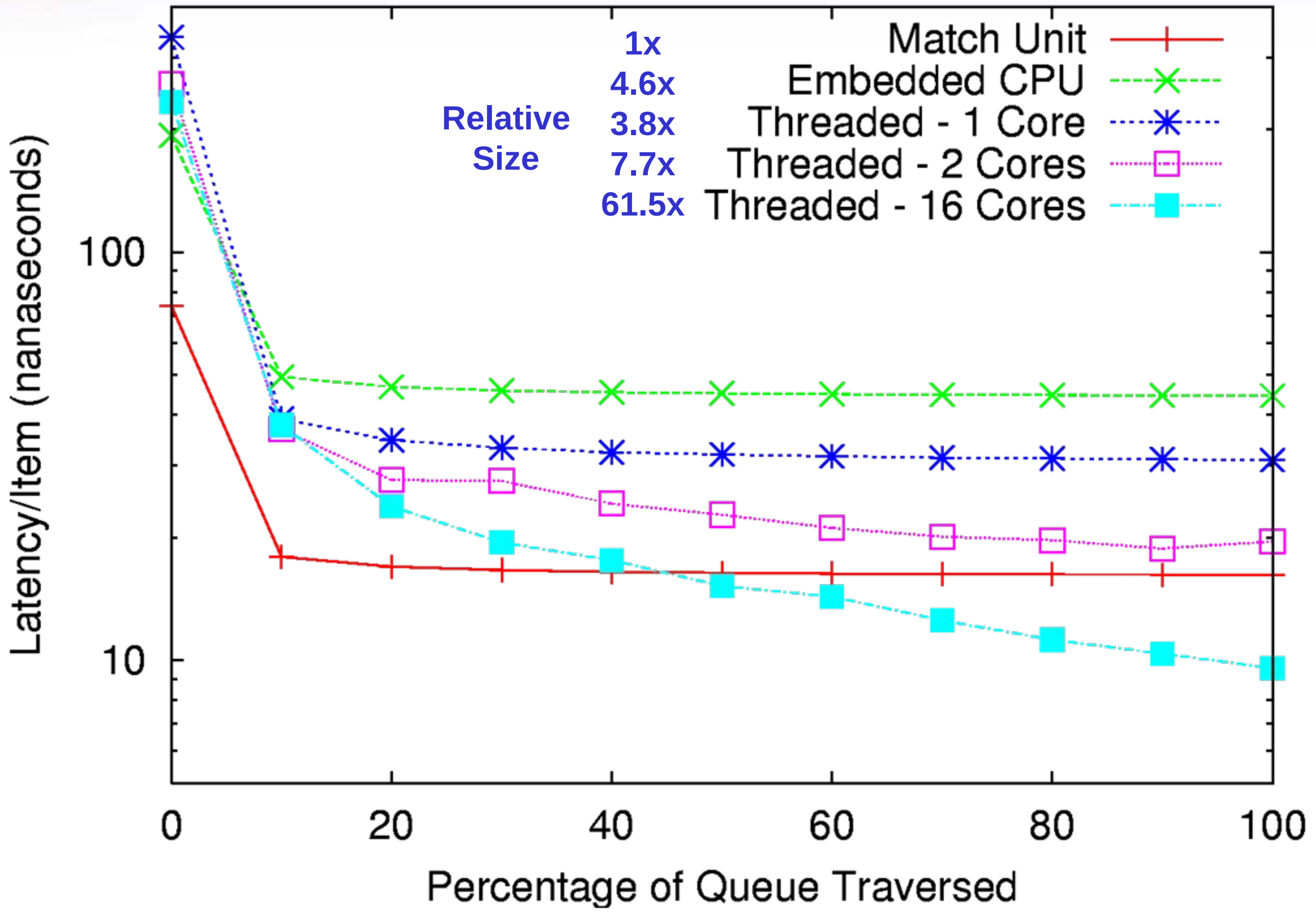
# Posted Queue Results – 128 Entry ALPU



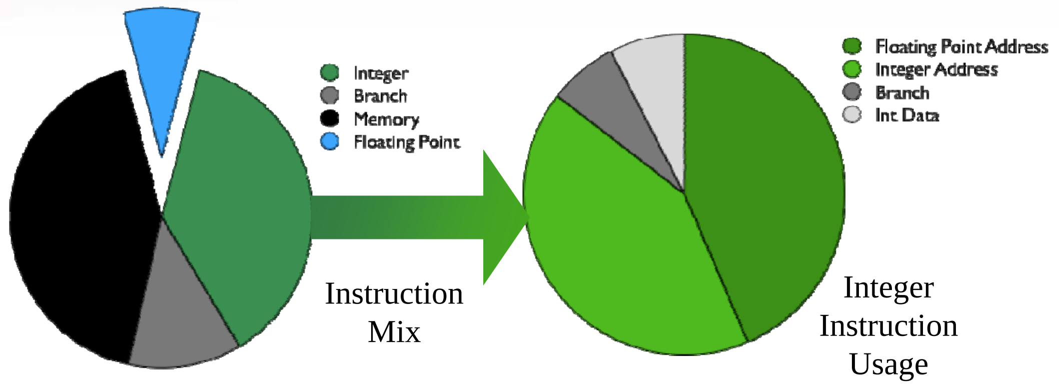
# Match Time Results (30 items)



# Match Time Results (300 items)



# Minimizing Memory Bandwidth Usage in Network Stack



Memory Usage in Sandia Applications

- Memory bandwidth is most often the limiting factor for on node performance
- We must minimize the use of host memory bandwidth in the network stack

- Bounce buffers (or any other copying) incur a 2x memory bandwidth penalty
- A fast off-load approach can minimize host memory bandwidth utilization
  - Allows the NIC to determine where received messages need to be put in host memory and DMA the data directly there, eliminating the need for bounce buffers
  - High message rate can reduce the need for buffering of non-contiguous data

## Portals

## Portals Network Programming Interface

- Network API developed by Sandia, U. New Mexico, Intel
- Previous generations of Portals deployed on several production massively parallel systems
  - 1993: 1800-node Intel Paragon (SUNMOS)
  - 1997: 10,000-node Intel ASCI Red (Puma/Cougar)
  - 1999: 1800-node Cplant cluster (Linux)
  - 2005: 10,000-node Cray Sandia Red Storm (Catamount)
  - 2009: 18,688-node Cray XT5 – ORNL Jaguar (Linux)
- Focused on providing
  - Lightweight “connectionless” model for MPP systems
  - Low latency
  - High bandwidth
  - Independent progress
  - Overlap of computation and communication
  - Scalable buffering semantics
- Supports MPI, Cray SHMEM, ARMCI, GASNet, Lustre, etc.

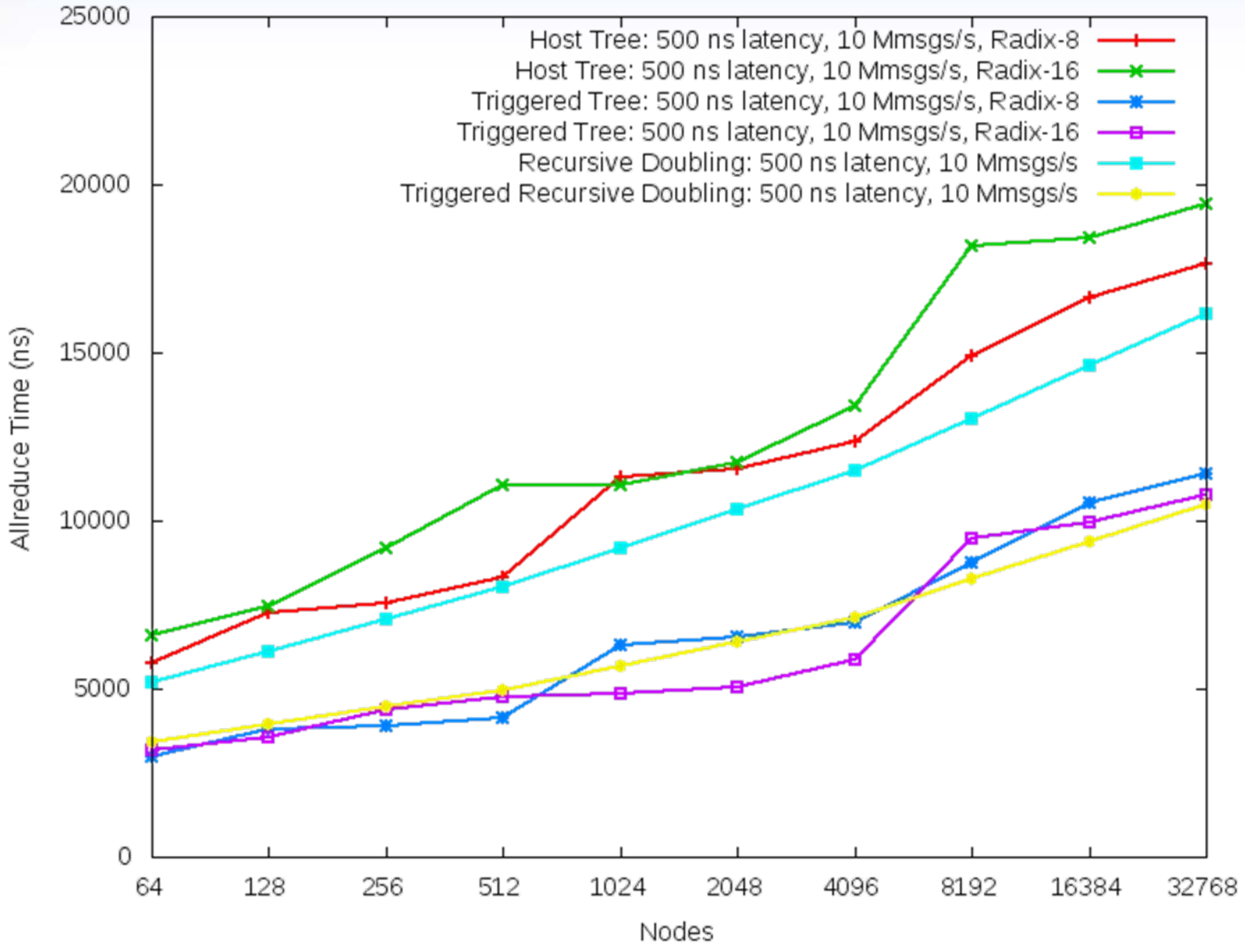
## Portals 4.0: Applying Lessons Learned from Cray SeaStar

- High message rate
  - Atomic search and post for MPI receives required round-trip across PCI
  - Eliminate round-trip by having Portals manage unexpected messages
- Flow control
  - Encourages well-behaved applications
    - Fail fast
    - Identify application scalability issues early
  - Resource exhaustion caused unrecoverable failure
  - Recovery doesn’t have to be fast
  - Resource exhaustion will disable Portal
  - Subsequent messages will fail with event notification at initiator
  - Applies back pressure from network
    - Performance for scalable applications
    - Correctness for non-scalable applications

## Portals 4.0 (cont'd)

- Hardware implementation
  - Designed for intelligent or programmable NICs
  - Arbitrary list insertion
  - Unneeded symmetry on initiator and target objects
- New functionality for one-sided operations
  - Eliminate matching information
    - Smaller network header
    - Minimize processing at target
  - Scalable event delivery
    - Lightweight counting events
  - Triggered operations
    - Chain sequence of data movement operations
    - Build asynchronous collective operations
      - Mitigate OS noise effects

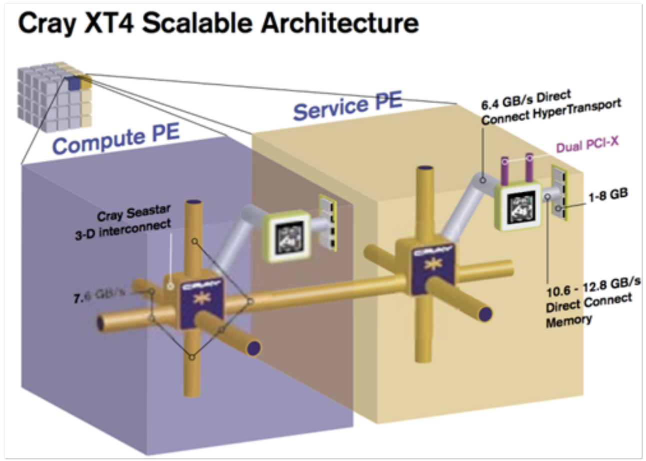
## Triggered Operations



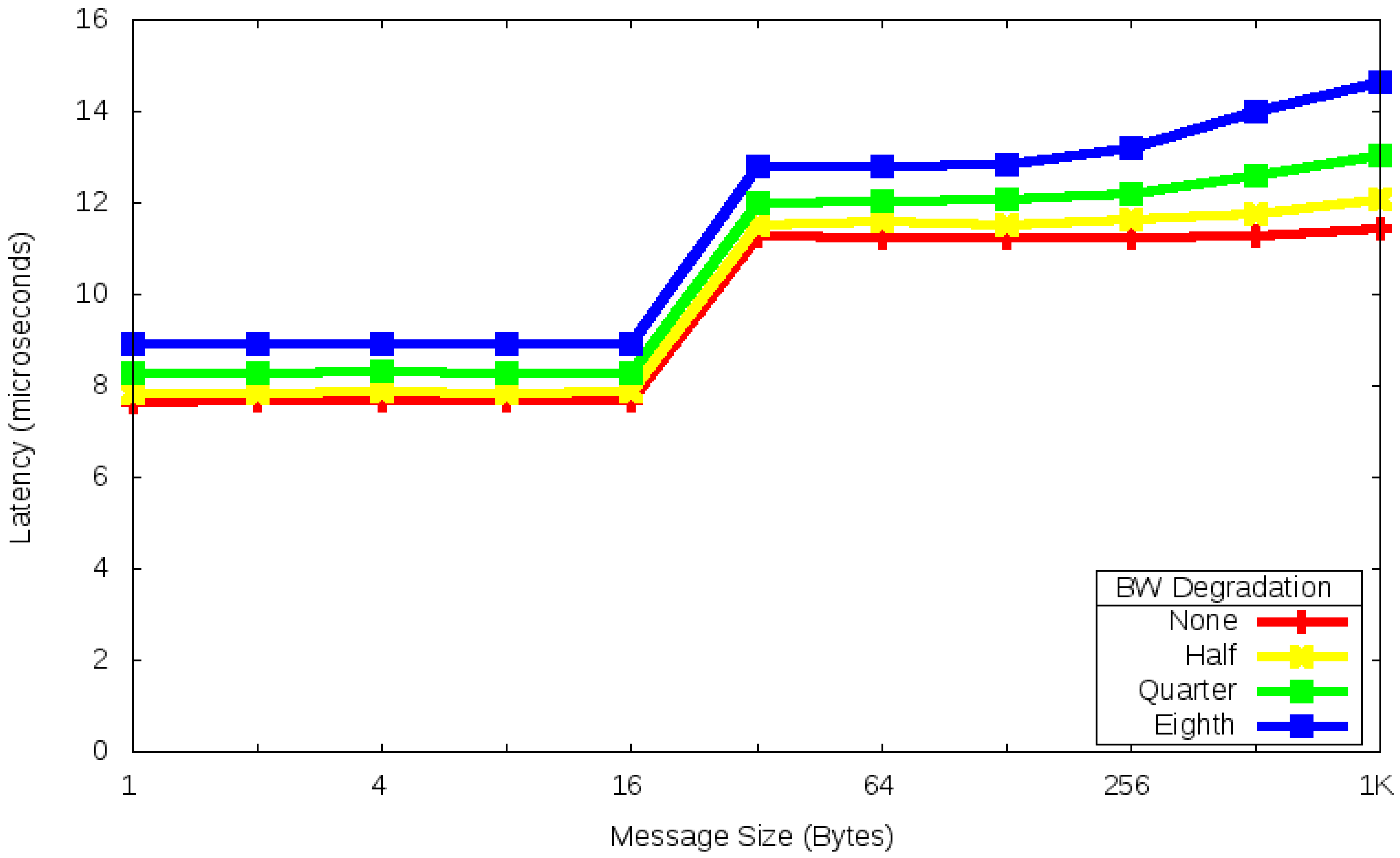
## Injection Bandwidth Detuning

# HyperTransport Detuning

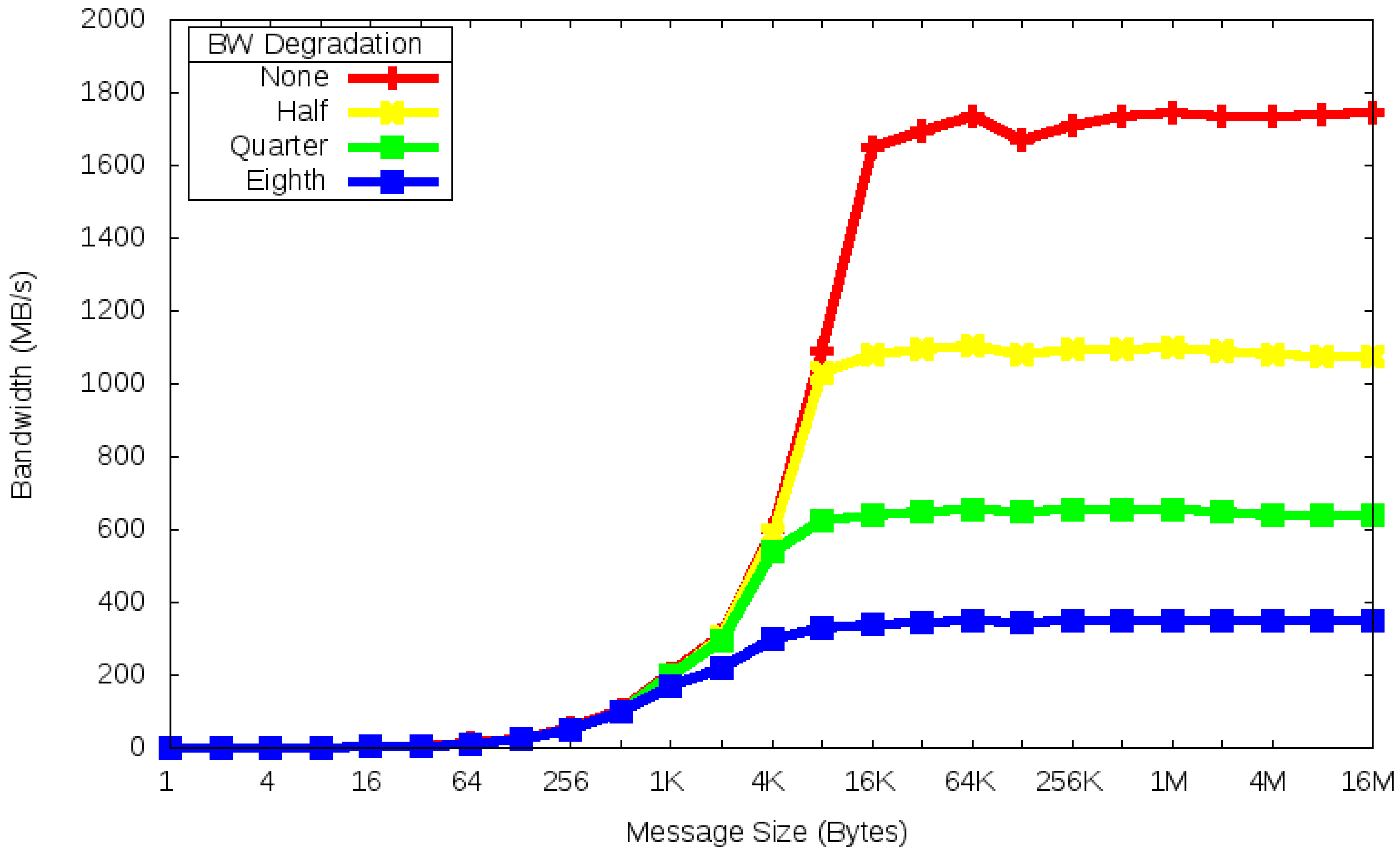
- HyperTransport link between Opteron and SeaStar setup at boot by Coldstart on Cray XT, BIOS on standard PCs
- Anyone may query HT widths and frequencies supported by SeaStar via the PCI config space:
  - 8 or 16 bits wide
  - 200, 400, or 800 MHz
- HT link config currently hard-coded in Coldstart


Cray XT4 Scalable Architecture
Compute PE
Service PE
6.4 GB/s Direct Connect HyperTransport
Dual PCI-X
1-8 GB
Cray Seastar 3-D interconnect
7.6 GB/s
10.6 - 12.8 GB/s Direct Connect Memory

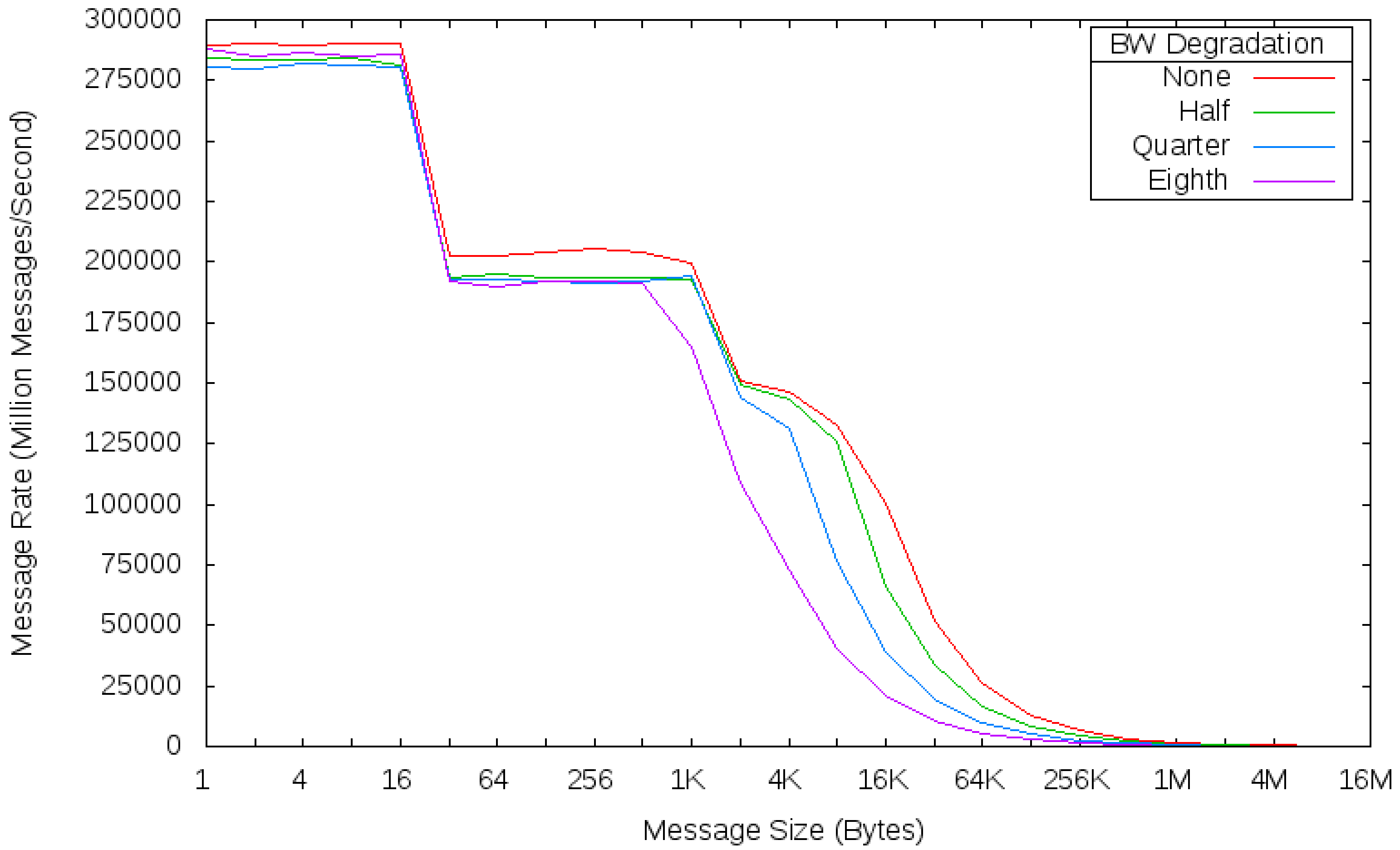
# OSU Latency


Latency (microseconds)
16
14
12
10
8
6
4
2
0
1
4
16
64
256
1K
Message Size (Bytes)
BW Degradation
None
Half
Quarter
Eighth

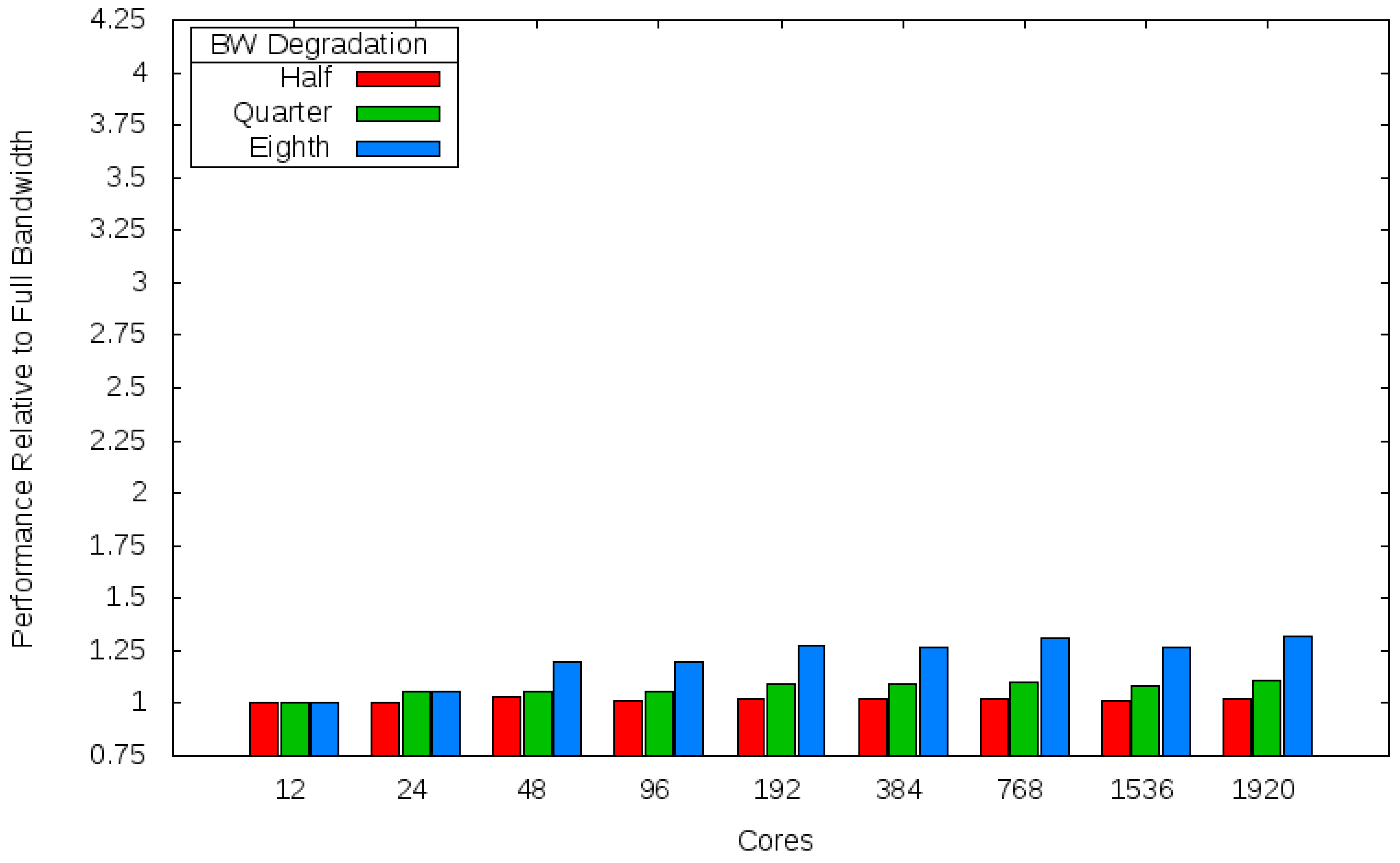
# OSU Bandwidth



# OSU Message Rate

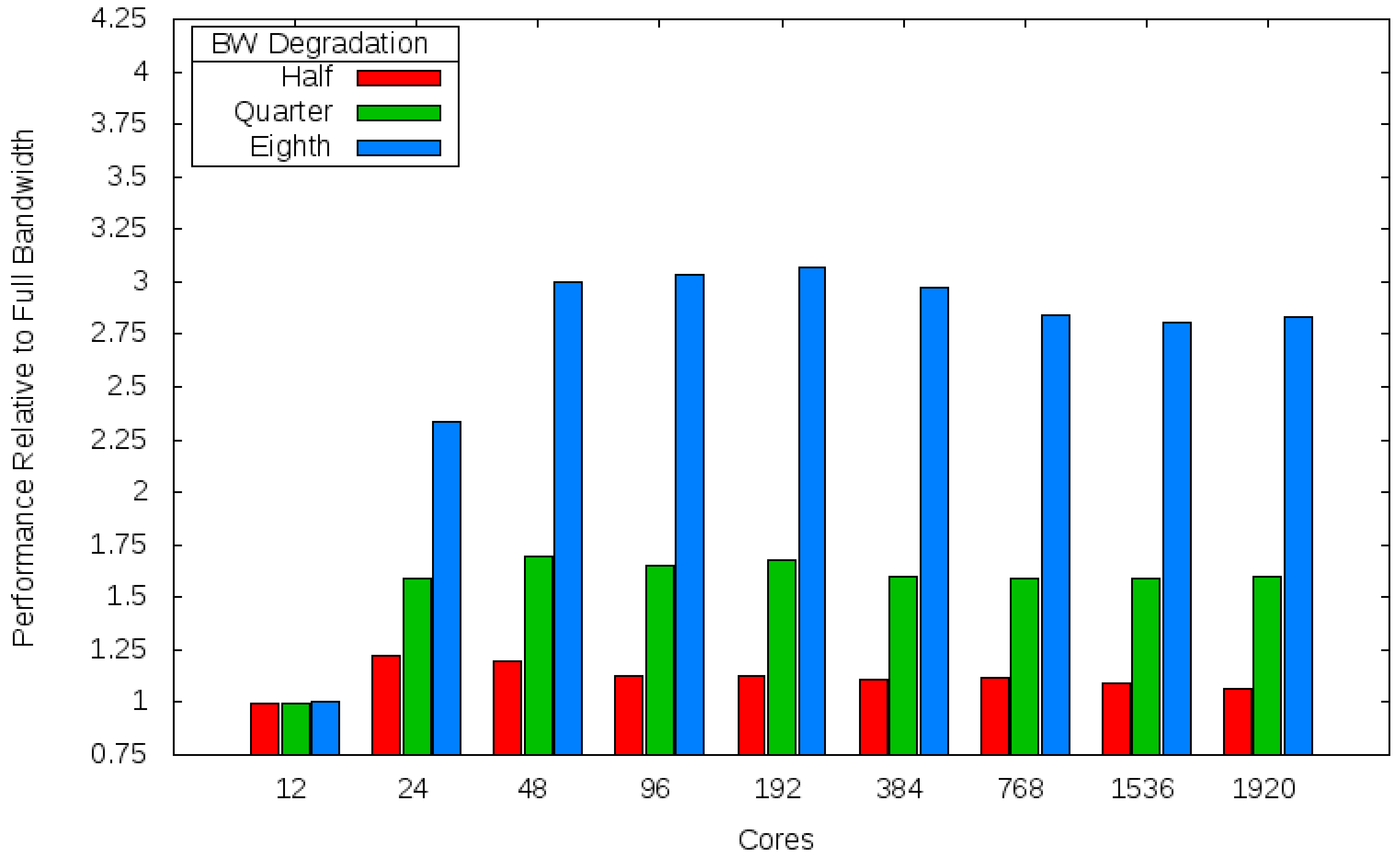


# HPCC - HPL

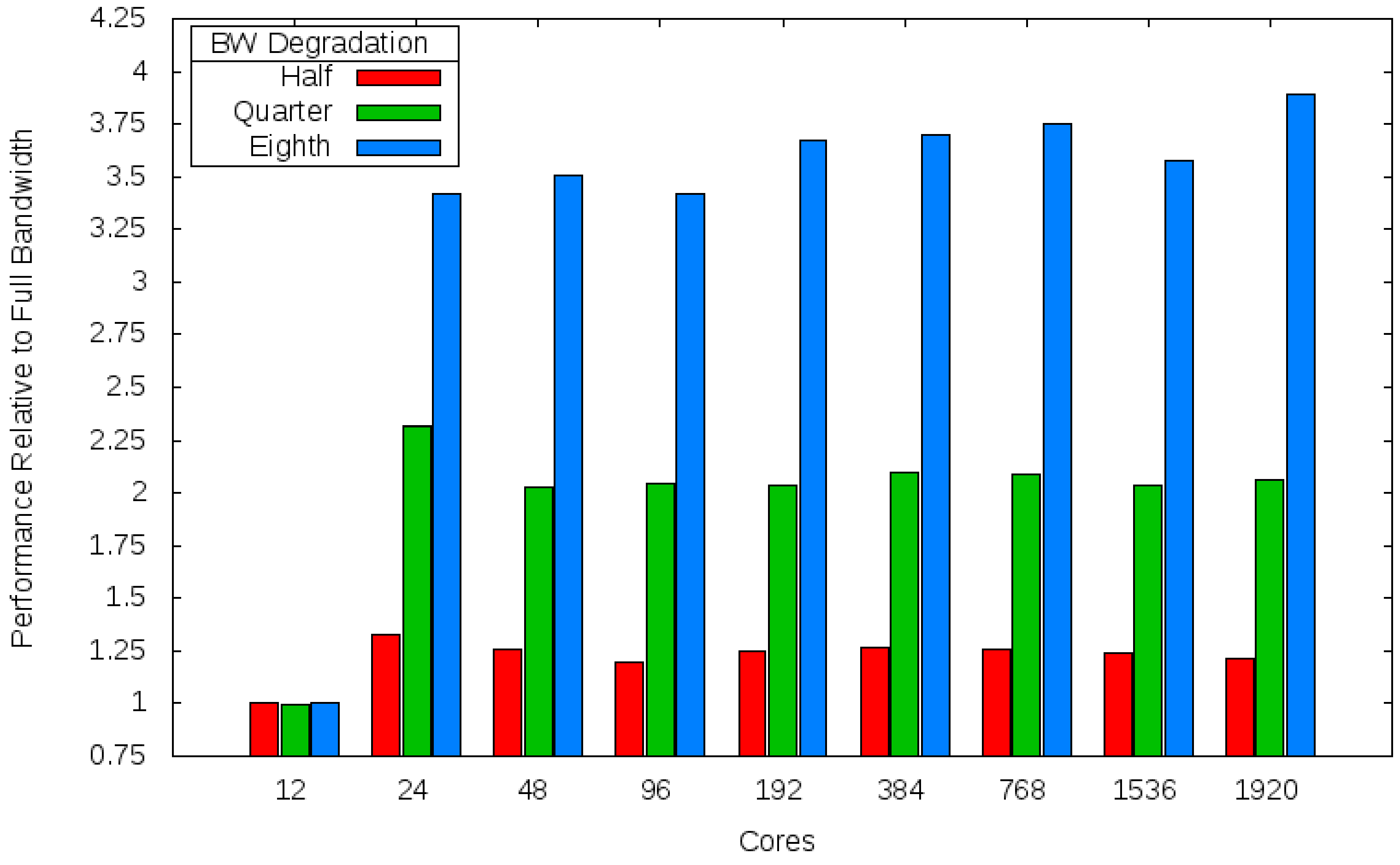


BW Degradation
Half
Quarter
Eighth
Performance Relative to Full Bandwidth
4.25
4
3.75
3.5
3.25
3
2.75
2.5
2.25
2
1.75
1.5
1.25
1
0.75
12
24
48
96
192
384
768
1536
1920
Cores

## HPCC - FFT

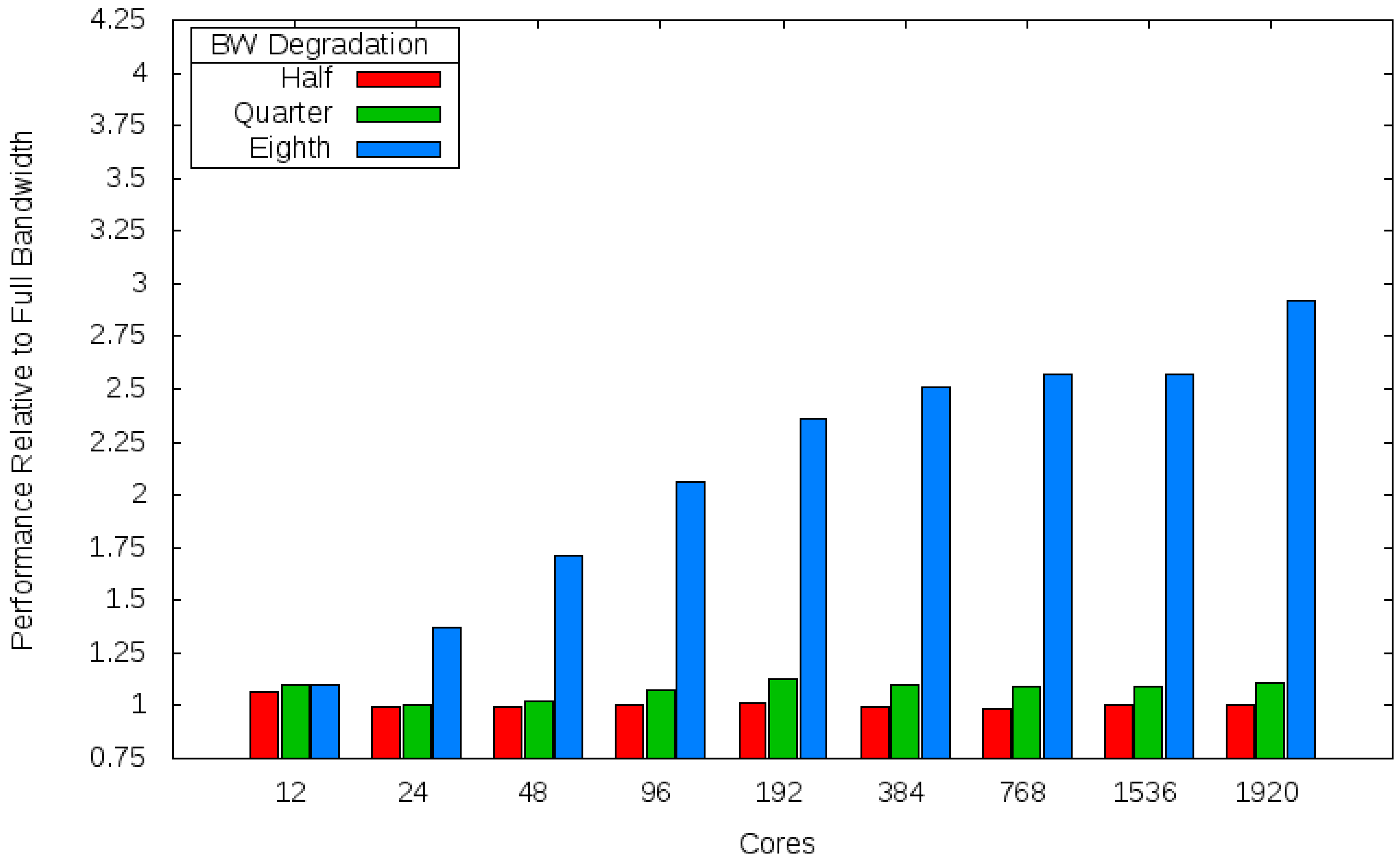

BW Degradation
Half
Quarter
Eighth
Performance Relative to Full Bandwidth
4.25
4
3.75
3.5
3.25
3
2.75
2.5
2.25
2
1.75
1.5
1.25
1
0.75
12
24
48
96
192
384
768
1536
1920
Cores

# HPCC - PTRANS

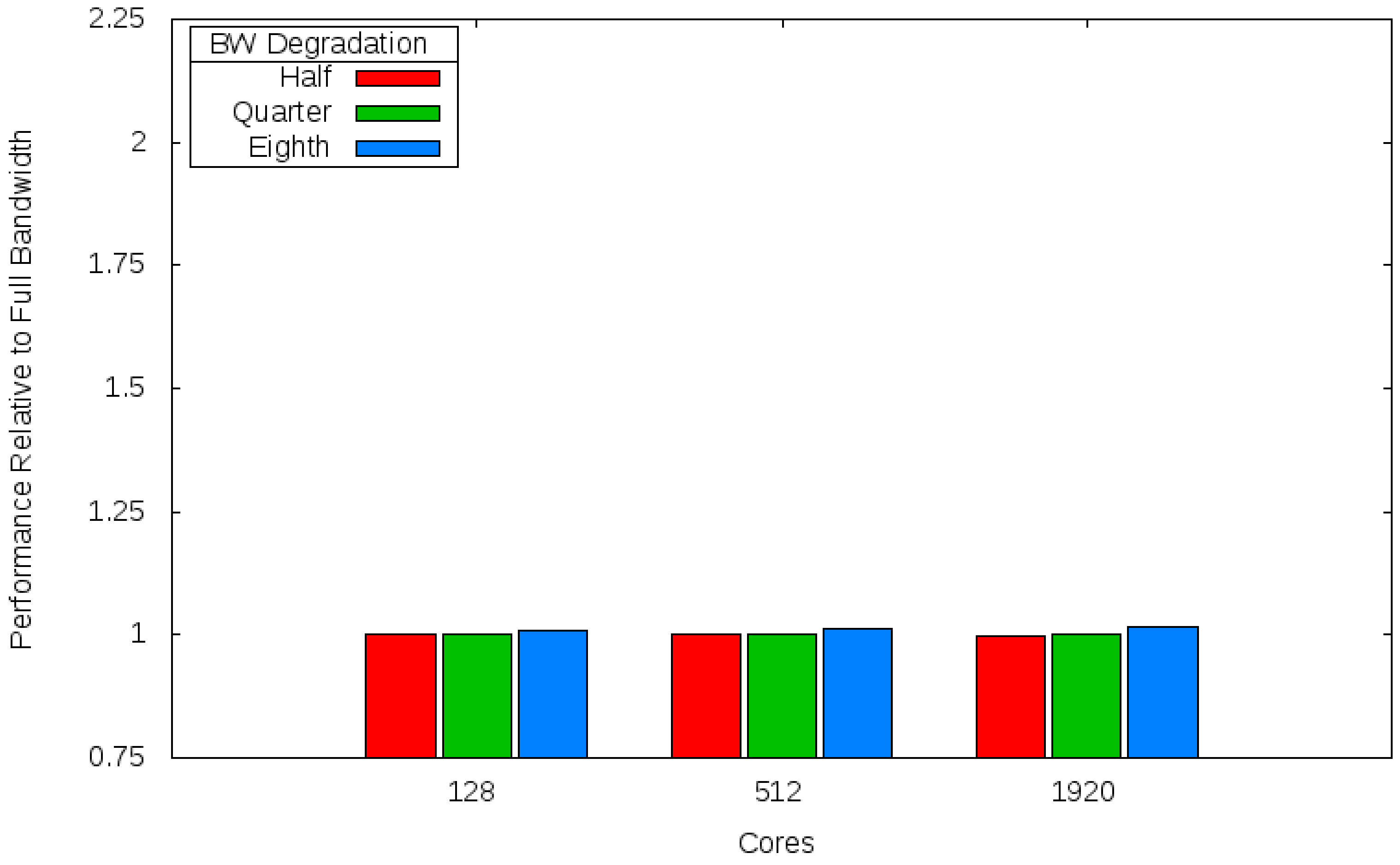


BW Degradation
Half
Quarter
Eighth
Performance Relative to Full Bandwidth
4.25
4
3.75
3.5
3.25
3
2.75
2.5
2.25
2
1.75
1.5
1.25
1
0.75
12
24
48
96
192
384
768
1536
1920
Cores

# HPCC - RandomAccess

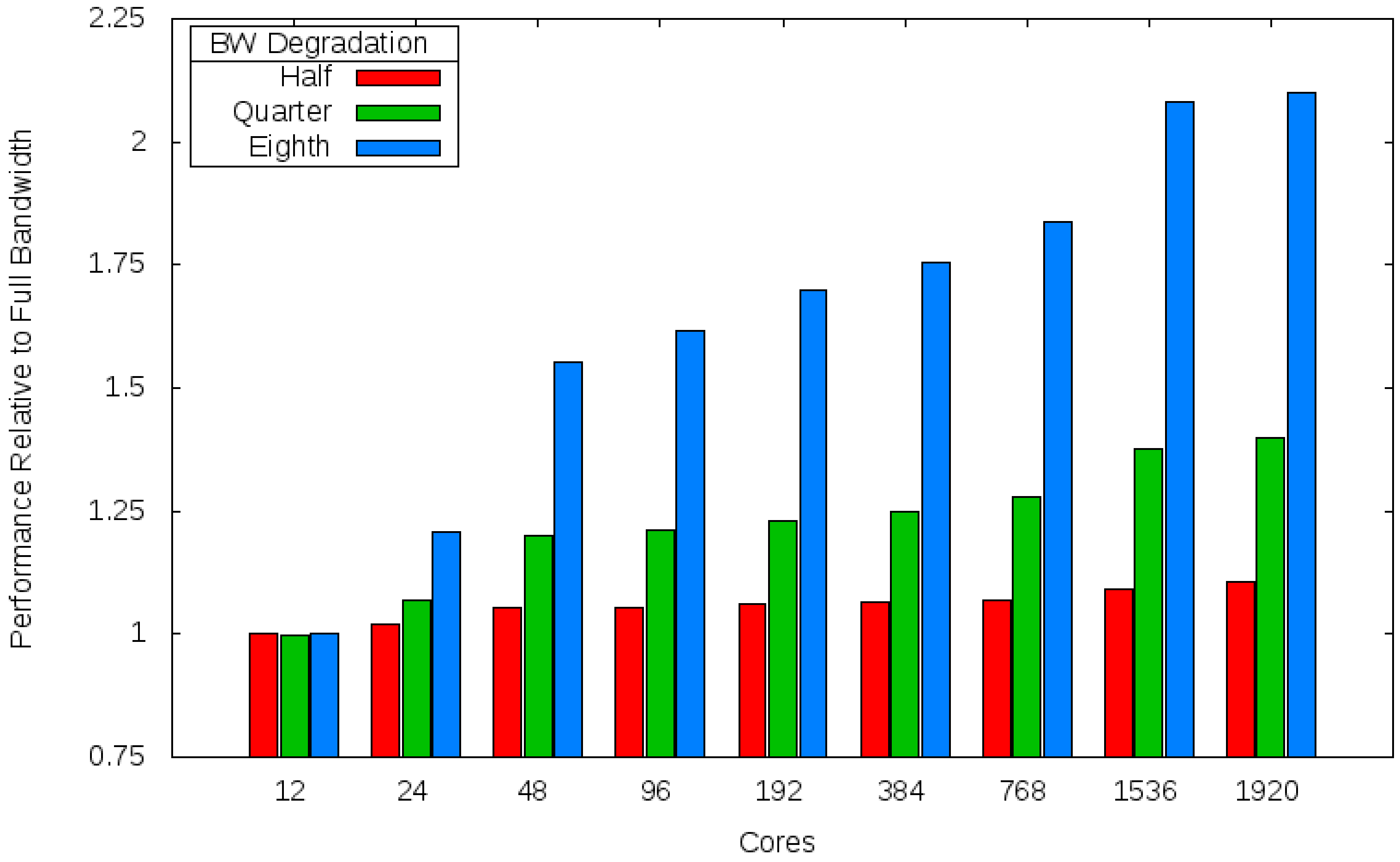


# Charon



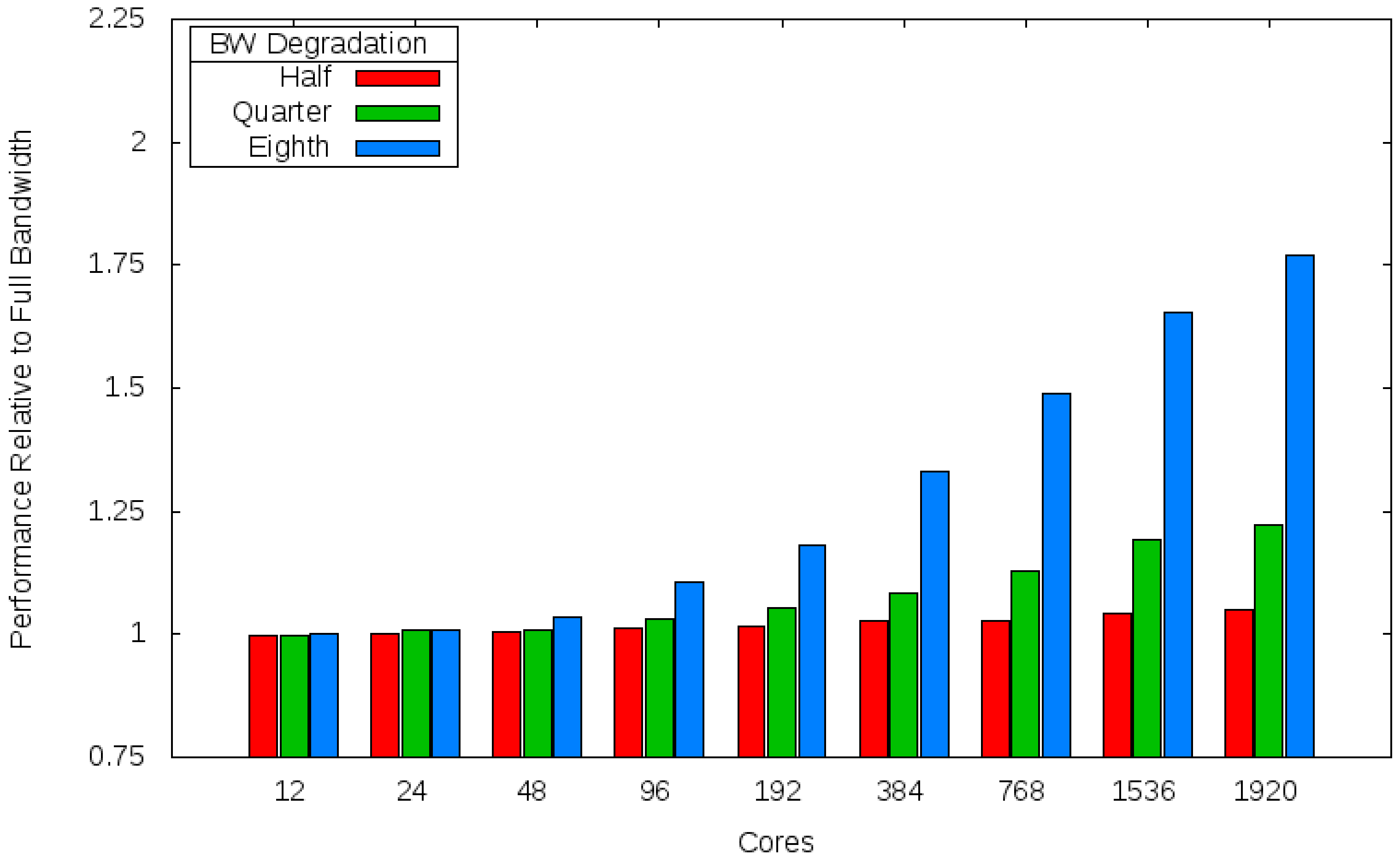
BW Degradation
Half
Quarter
Eighth
Performance Relative to Full Bandwidth
2.25
2
1.75
1.5
1.25
1
0.75
128
512
1920
Cores

# CTH



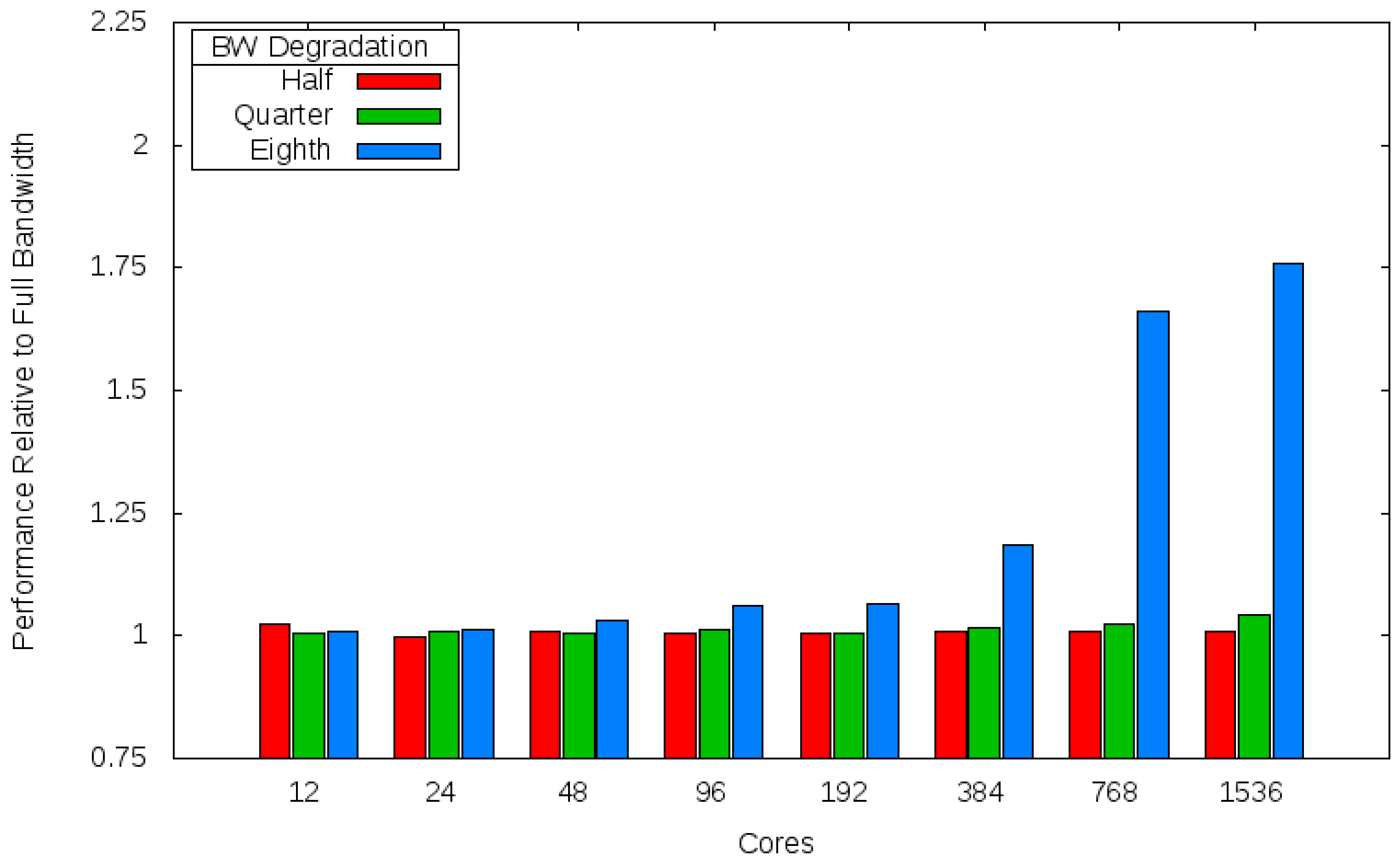
BW Degradation
Half
Quarter
Eighth
Performance Relative to Full Bandwidth
2.25
2
1.75
1.5
1.25
1
0.75
12
24
48
96
192
384
768
1536
1920
Cores

# SAGE



BW Degradation
Half
Quarter
Eighth
Performance Relative to Full Bandwidth
2.25
2
1.75
1.5
1.25
1
0.75
12
24
48
96
192
384
768
1536
1920
Cores

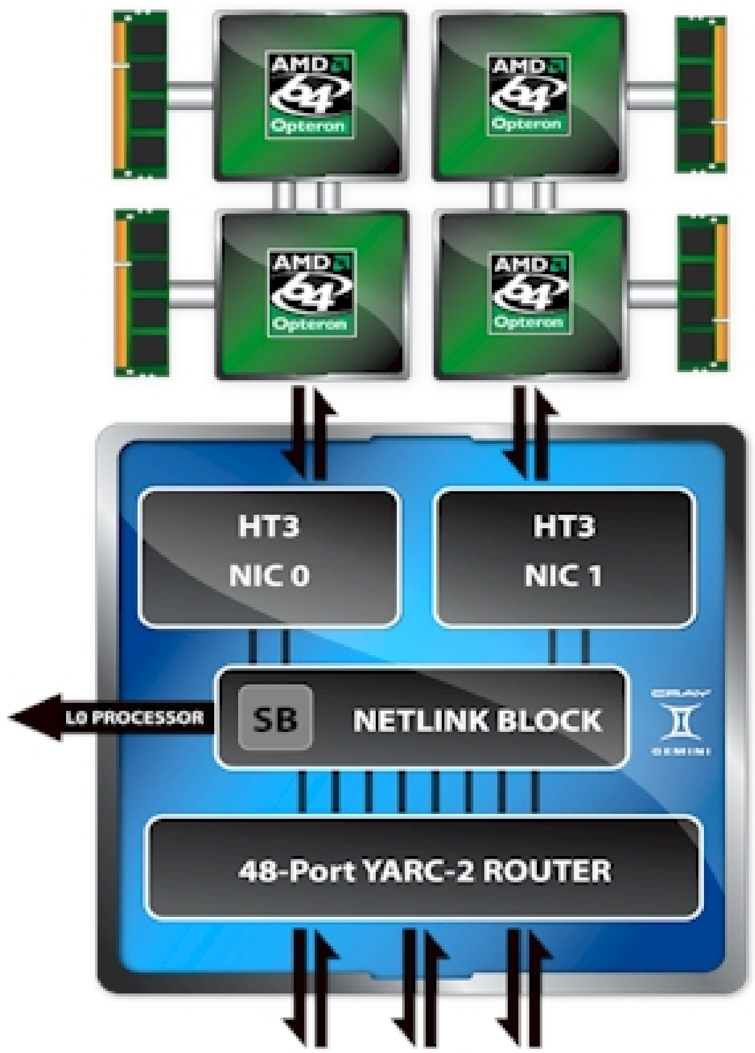
# xNOBEL



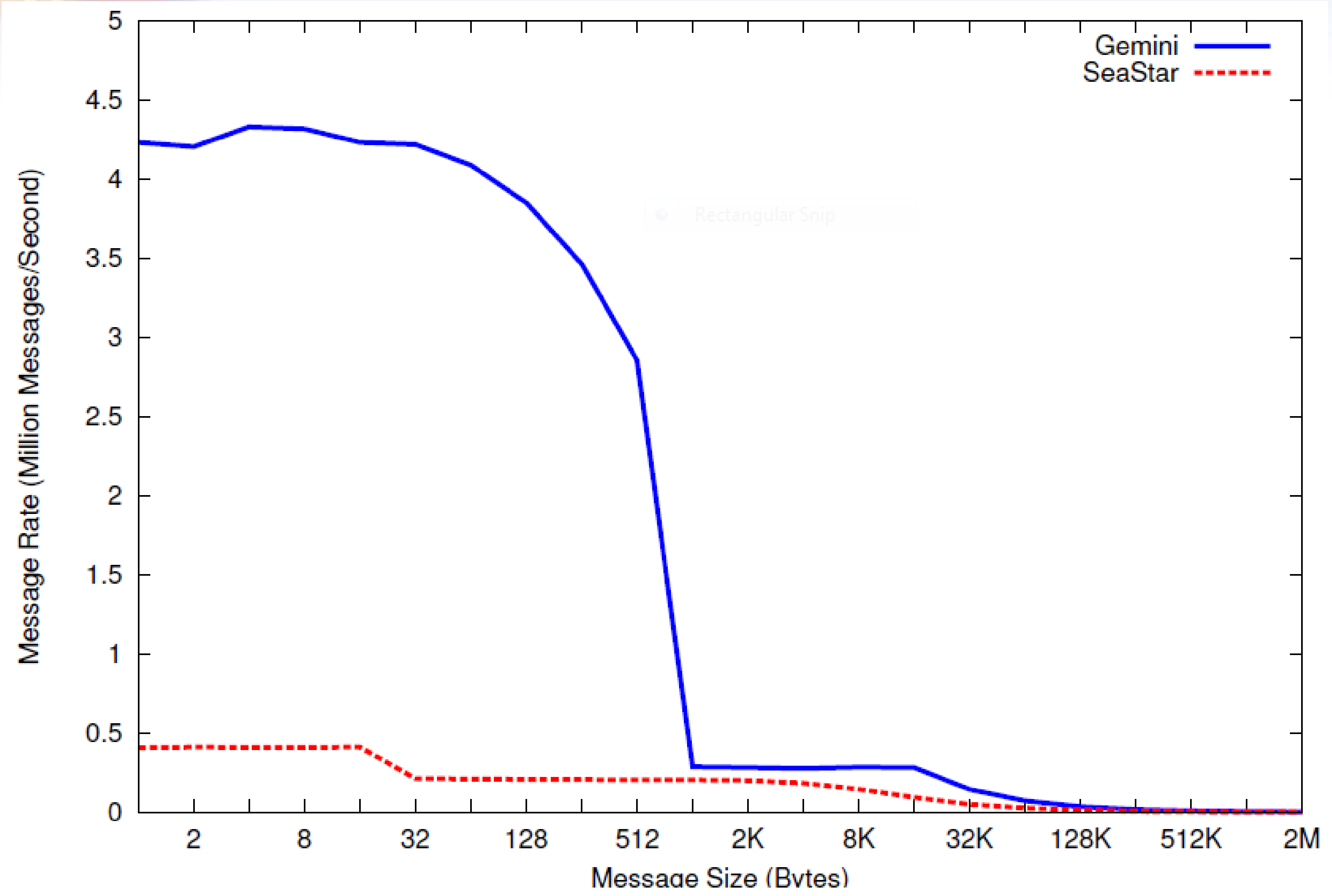
BW Degradation
Half
Quarter
Eighth
Performance Relative to Full Bandwidth
Cores
2.25
2
1.75
1.5
1.25
1
0.75
12
24
48
96
192
384
768
1536

## Gemini
## Memory Affinity

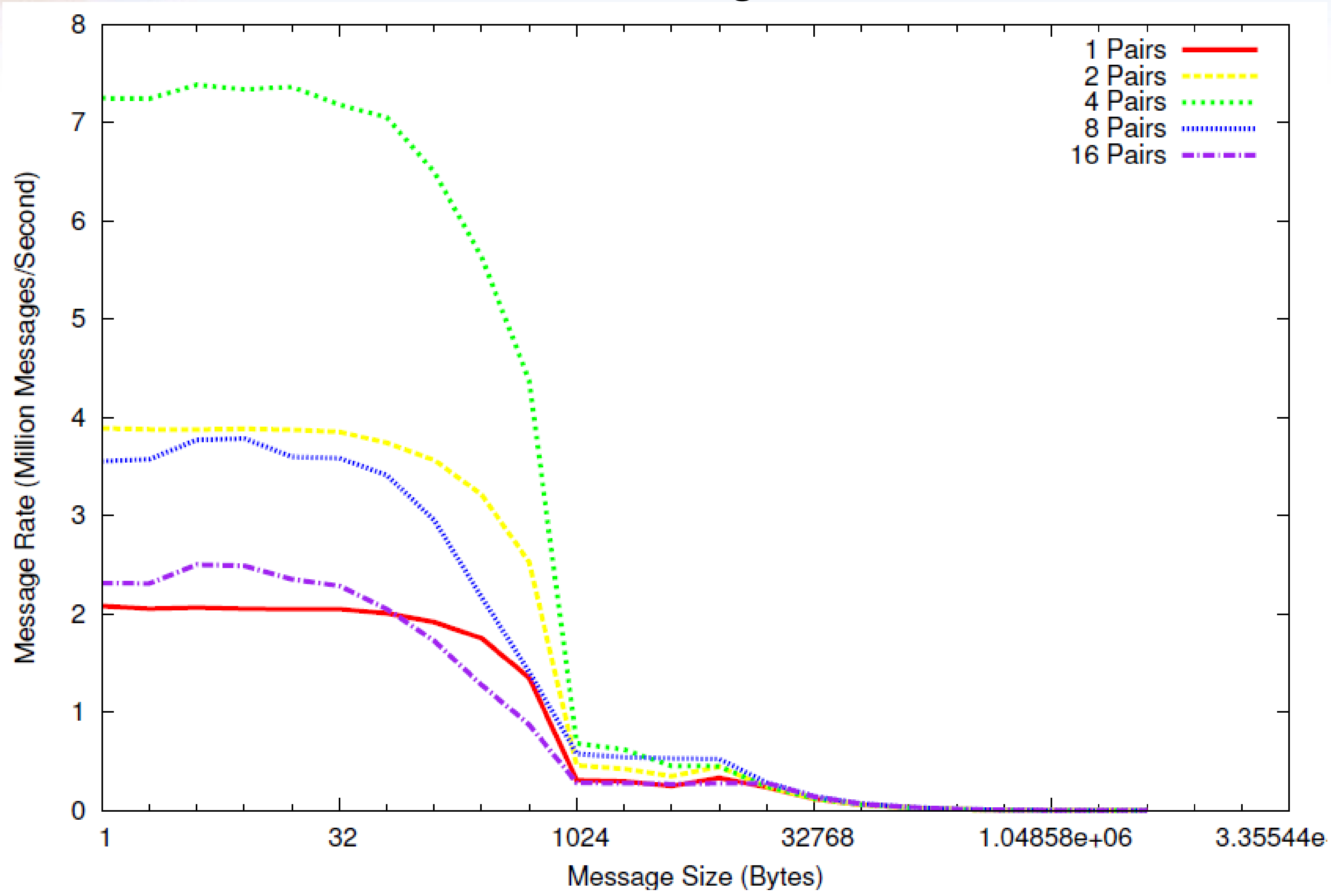
# Cray XE6 Node



# SeaStar vs Gemini



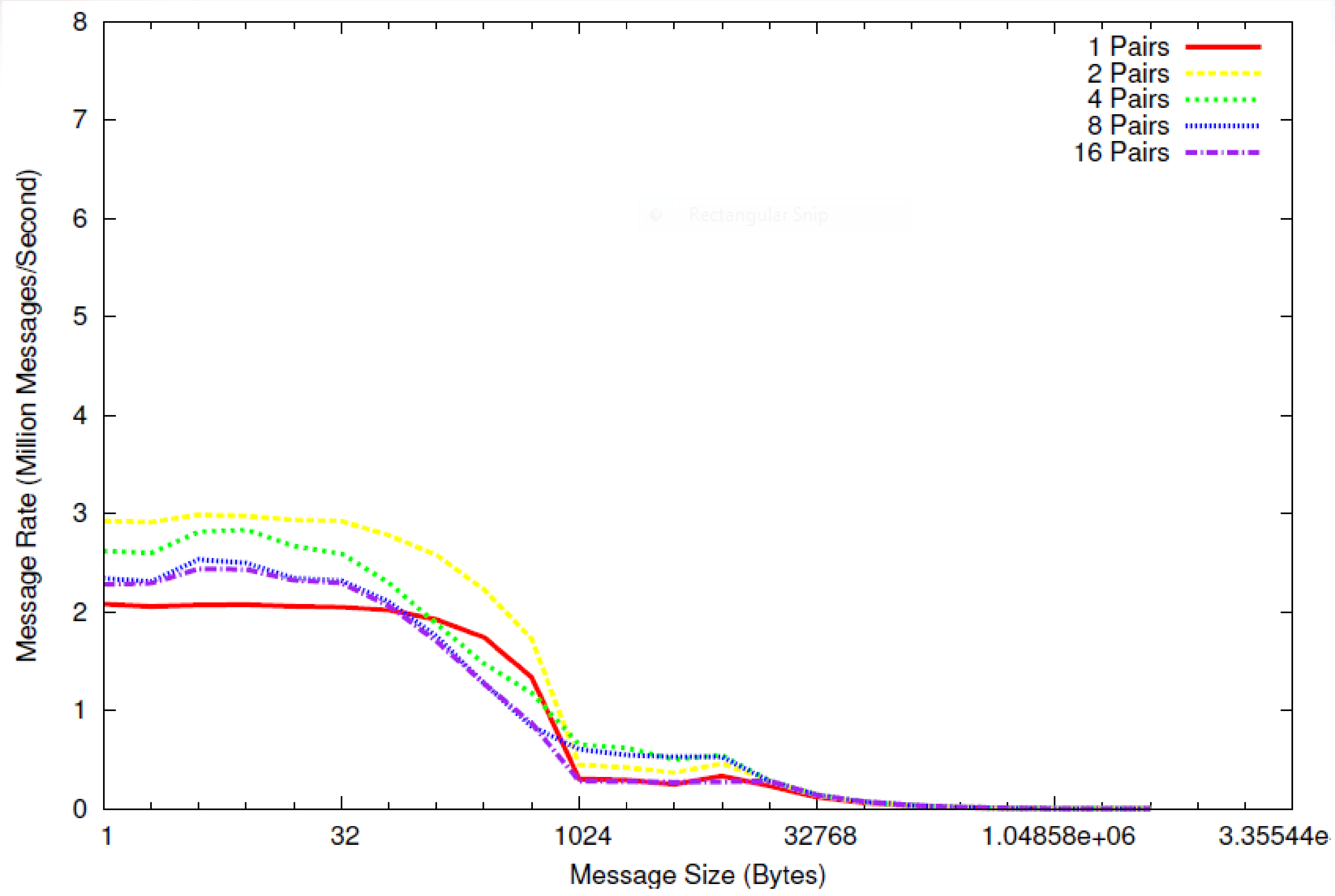
# Gemini
# Default Configuration



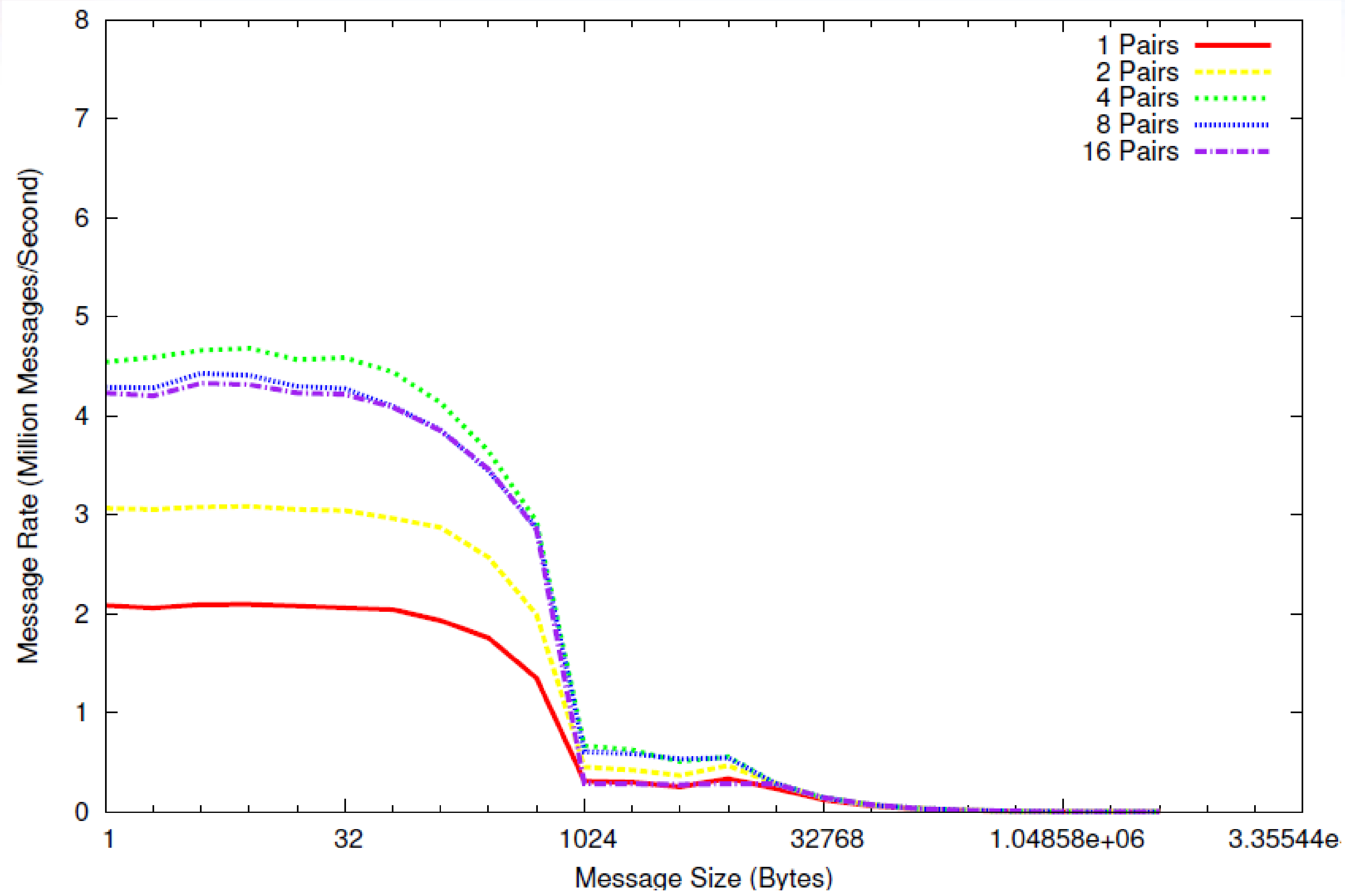
National Nuclear Security Administration

Sandia National Laboratories

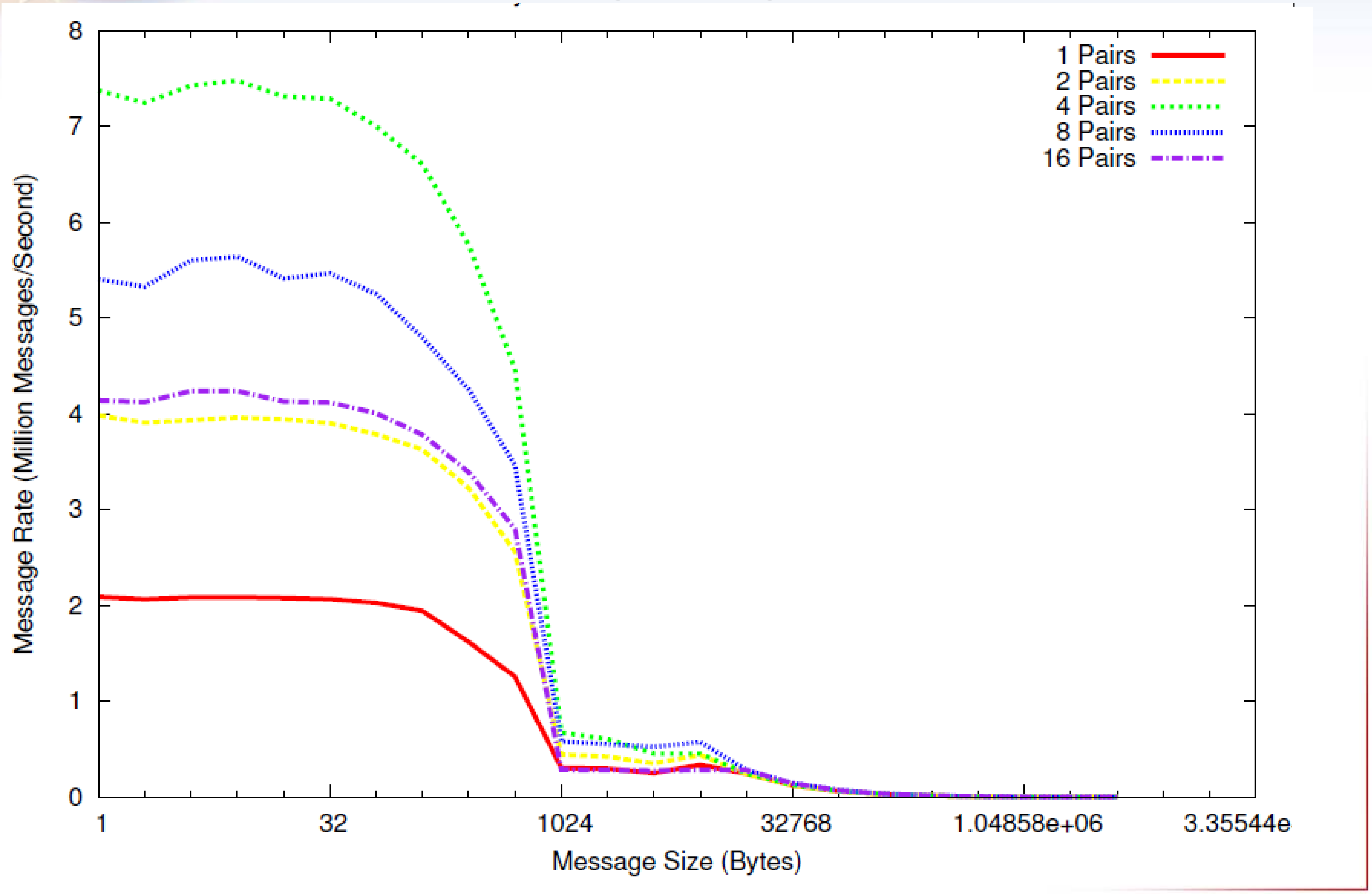
# Gemini
# Balanced NUMA Distribution



# Gemini
## MBOX=NIC, Balanced NUMA Distribution



# Gemini
MBOX=NIC



# Acknowledgments

- Brian Barrett
- Kurt Ferreira
- Scott Hemmert
- Jim Laros
- Sue Kelly
- Kevin Pedretti
- Courtenay Vaughan
- Keith Underwoord (Intel)