SAND2015-9425PE

Exceptional service in the national interest

Sandia National Laboratories

Photos placed in horizontal position with even amount of white space between photos and header

HMC and Multi-level Memory

Gwen Voskuilen

U.S. DEPARTMENT OF ENERGY

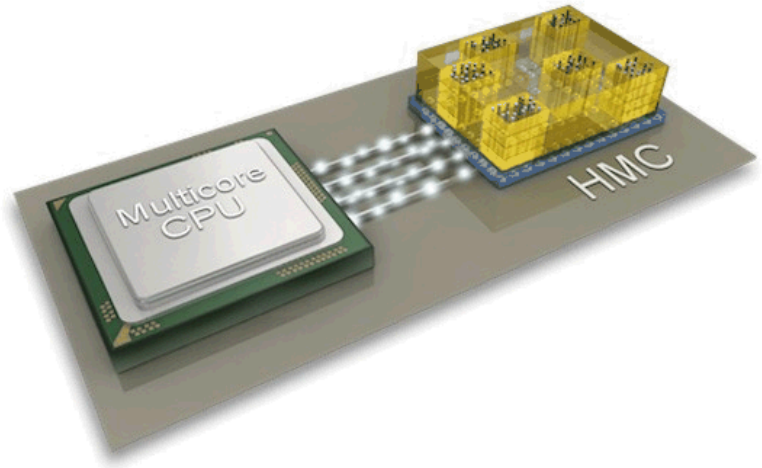
NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

Coming soon to a machine near you…

Sandia National Laboratories

- HMC is a huge opportunity for bandwidth limited applications
  - Will have limited capacity compared to DDR or non-volatile memories
    - $$$
- Machines will have multiple memories with different bandwidth, latency, and power characteristics
  - Intelligent data management becomes essential



Managing multi-level memory

Sandia National Laboratories

- Understanding the application
  - What data should go where?
  - Who is responsible for deciding?
    - Programmer? OS? Hardware? All of them?
- Understanding the HMC
  - How does the access pattern affect performance?
  - How does data layout affect performance?
- Addressing the system issues
  - Can we make use of HMC's big request sizes?
  - Where can we use HMC's atomic capabilities?
  - And what system support is needed for these?

Tools: SST (VaultSim, Sieve, etc.), HMC testbeds

Outline

- Understanding applications: multi-level memory
- Understanding HMC: testbeds

The Case of the Multi-Level Memory

Sandia National Laboratories

- For
  - Lots of potential
  - Lower cost
  - More "Effective" bandwidth
- Against
  - Potential – To waste $$$
  - How do you actually use it?
  - BW not enough – latency effects
  - Can one memory configuration "fit" all?

Sample configurations:

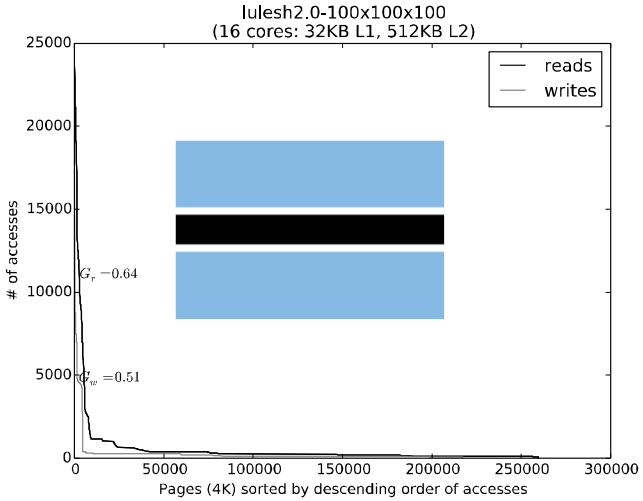
Memory	Size	Cost/bit	Total Cost
"HMC"	5%	3.0	0.15
DDR	30%	1.0	0.3
Flash	65%	0.15	0.1
Total Cost			0.54X

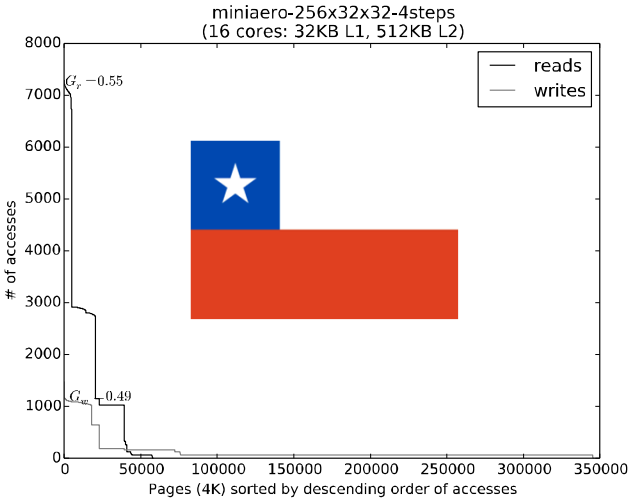
Memory	Accesses	Bandwidth
"HMC"	65%	14.1
DDR	30%	1.0
Flash	5%	0.18
Weighted Harmonic Mean		1.6X

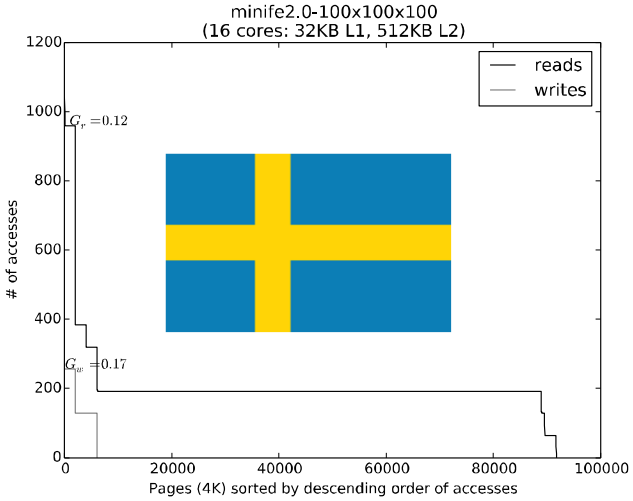
Gini Analysis: Diverse Distribution

Sandia National Laboratories

Pages sorted by access counts

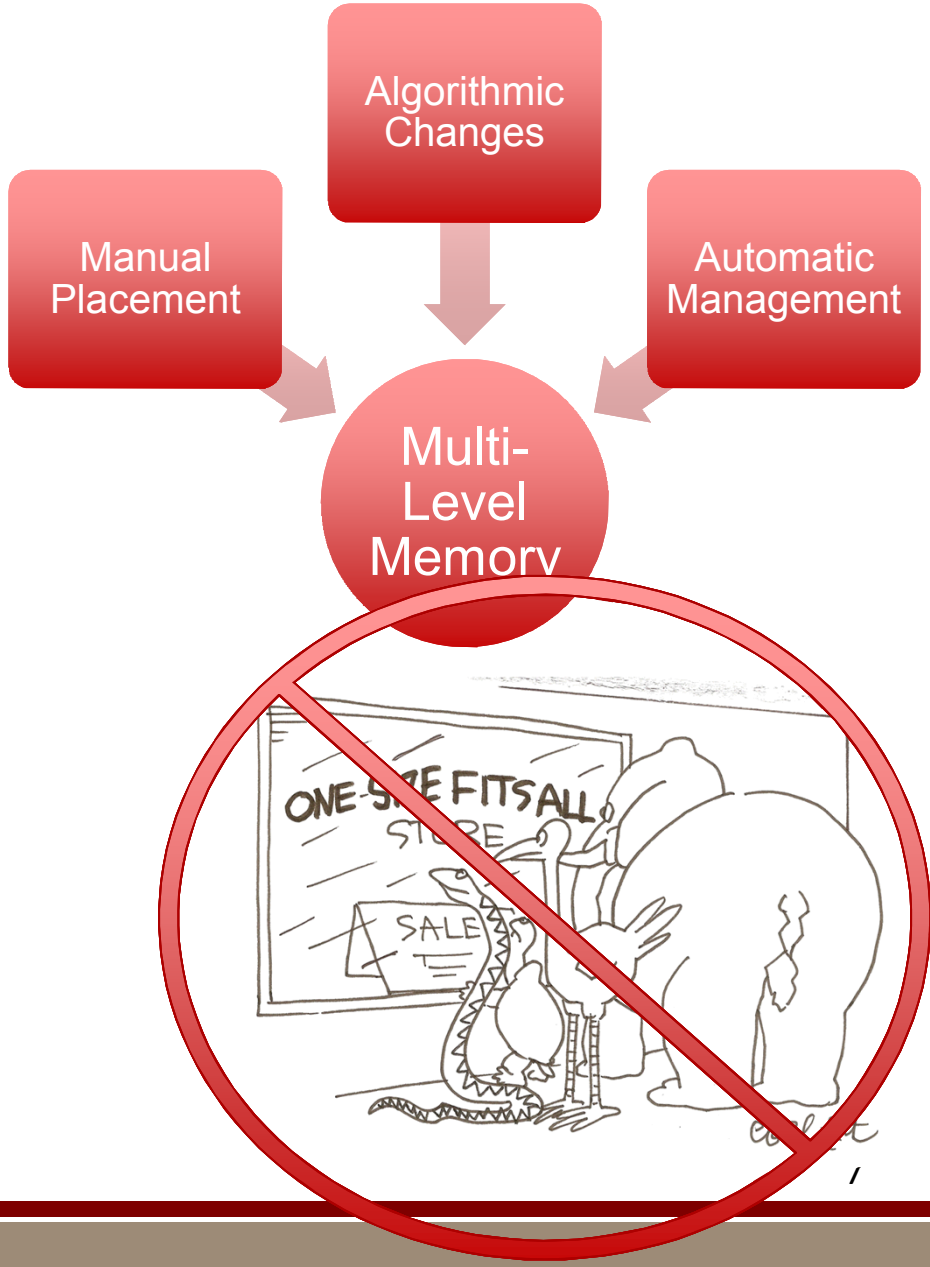






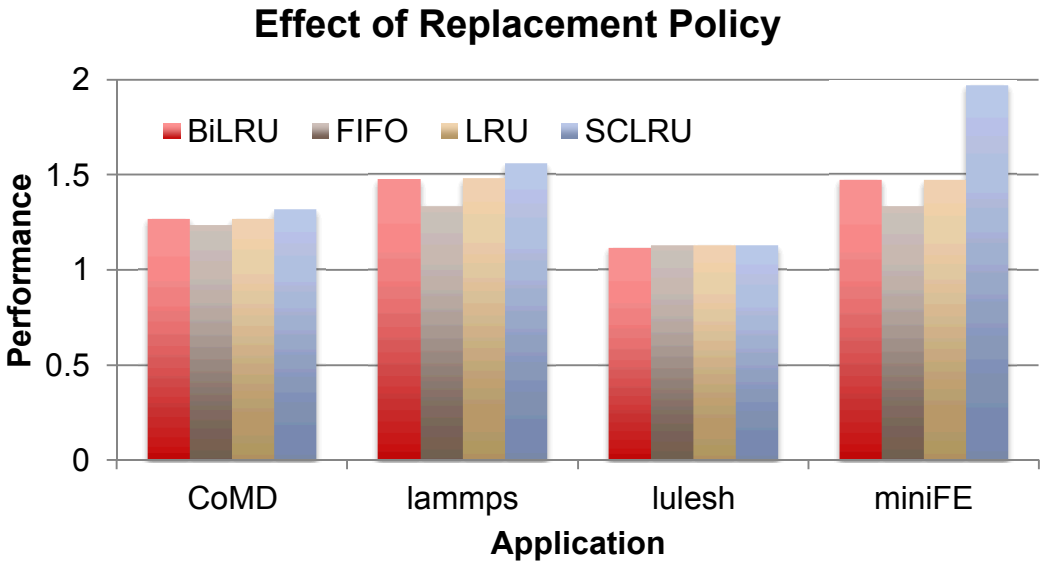
No "One-Size-Fits-All"

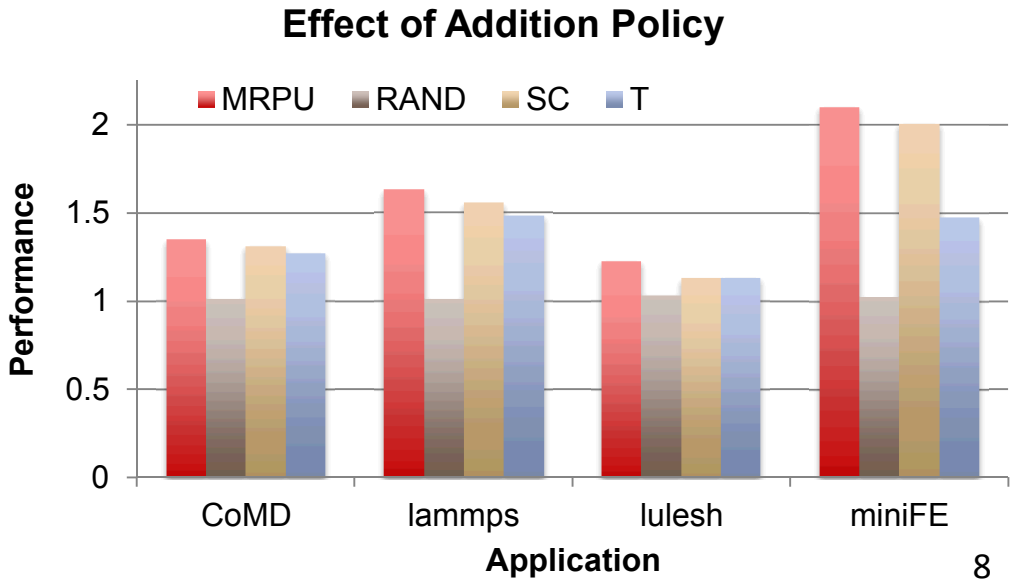
- Diverse applications require diverse MLM approaches
- Algorithmic: extensive program changes
- Manual: Use-directed placement
- **Automatic**: OS/RT/HW manages movement of data
- Tradeoffs
  - Performance / User effort



Automatic Management

- How do you decide what "goes in" to fast memory?
- How do you decide what gets "kicked out"?
- Traditional OS/cache research focuses on 2nd question (replacement)
- MLM performance is more sensitive to the 1st (addition)





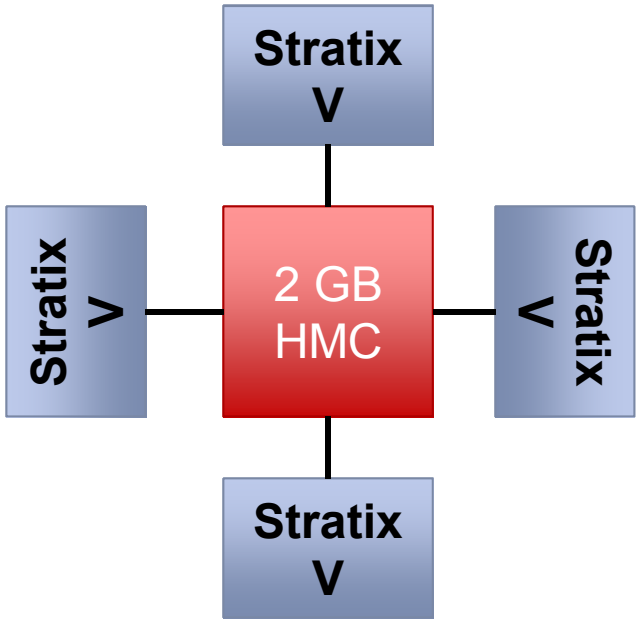
Understanding HMC

Sandia National Laboratories

- Likely that applications will have more data that could benefit from being put in HMC than will fit
- So…how do we decide what makes the cut?
  - Are all “hot pages” created equal?
- Use testbeds to characterize HMC performance
  - How does access pattern affect bandwidth utilization?
  - How do quads/vaults/banks respond to contention?
    - How does this affect data layout?
  - How does link-quad locality affect performance?
- Simulation/testbed

Sandia testbeds

- EX-800
  - Four Stratix V FPGAs
  - One four-link, 2GB HMC
  - PCIe host->board interface
  - Controller on each FPGA handles translation of requests to HMC flits
    - Command, size, address, tag, (data)
- Merlin (new!)
  - "Shared-memory" host->board interface
  - CAPI
  - More tomorrow



EX-800 Architecture

Characterize HMC performance

Sandia National Laboratories

- Characterize performance as a function of:
  - Request size
  - Access contention
  - Read-write ratio
  - Local vs. remote quad accesses

Evaluating access patterns

Sandia National Laboratories

- Trace-based
  - Memory access traces from MiniApps
    - Post-cache access trace (LLC misses only)
  - Practical limitation on trace size
    - Probably ~10Ks (have done 20K/FPGA → 80K access trace)
- Pattern generators
  - Scales to large # of requests
  - Generates pre-cache accesses
    - May differ significantly from post-cache access pattern
  - OK for patterns with low cache hit rates
  - Linear, SpMV, random, stream, etc.
  - Pulled from mini-apps

#newhardwareproblems

Sandia National Laboratories

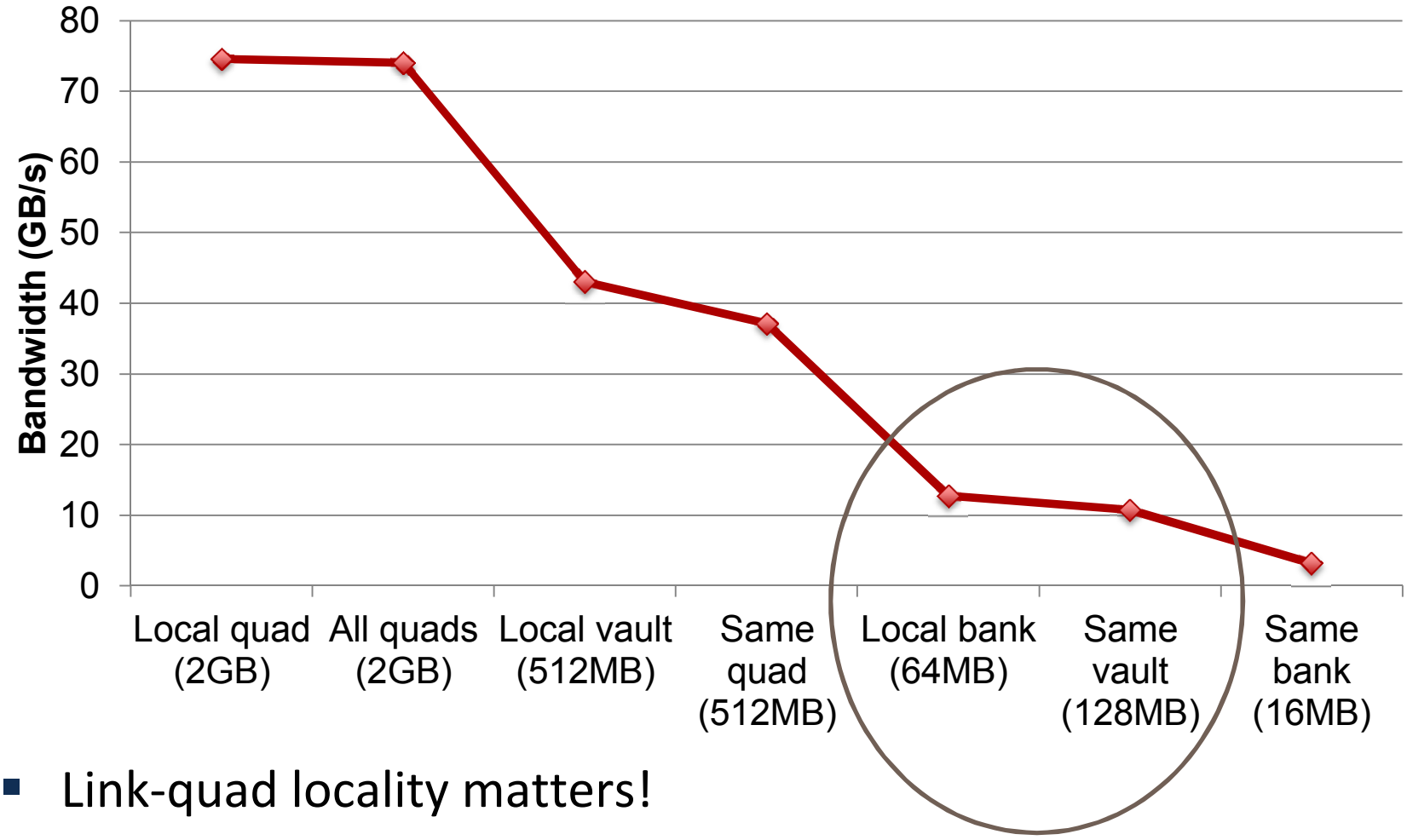
- Early hiccups
  - Socketed HMCs…
    - Good for upgrades
    - (really) Bad for reliability
  - Some (FPGA-side) controller bugs caused certain configurations to fail
    - Write-only, request sizes > 64B, etc.
- Micron/Pico did some upgrades on the boards over the past several months
  - No more sockets
  - Updated controller with bug fixes

Preliminary results: Contention

Sandia National Laboratories

- Vary access domain → which area of the HMC a link accesses
  - Measure effect of contention
  - Using GUPS/random access pattern
  - Using read-only accesses versus reads and writes
  - 128B accesses
  - Domains
    - All quads: any link to any quad
    - Own quad: each link to local quad
    - Own vault: each link to local vault
    - Own bank: each link to local bank
    - Same quad: all links to one quad
    - Same vault: all links to one vault
    - Same bank: all links to one bank

Effect of contention on bandwidth



- Link-quad locality matters!
  - But full bandwidth also depends on using all links

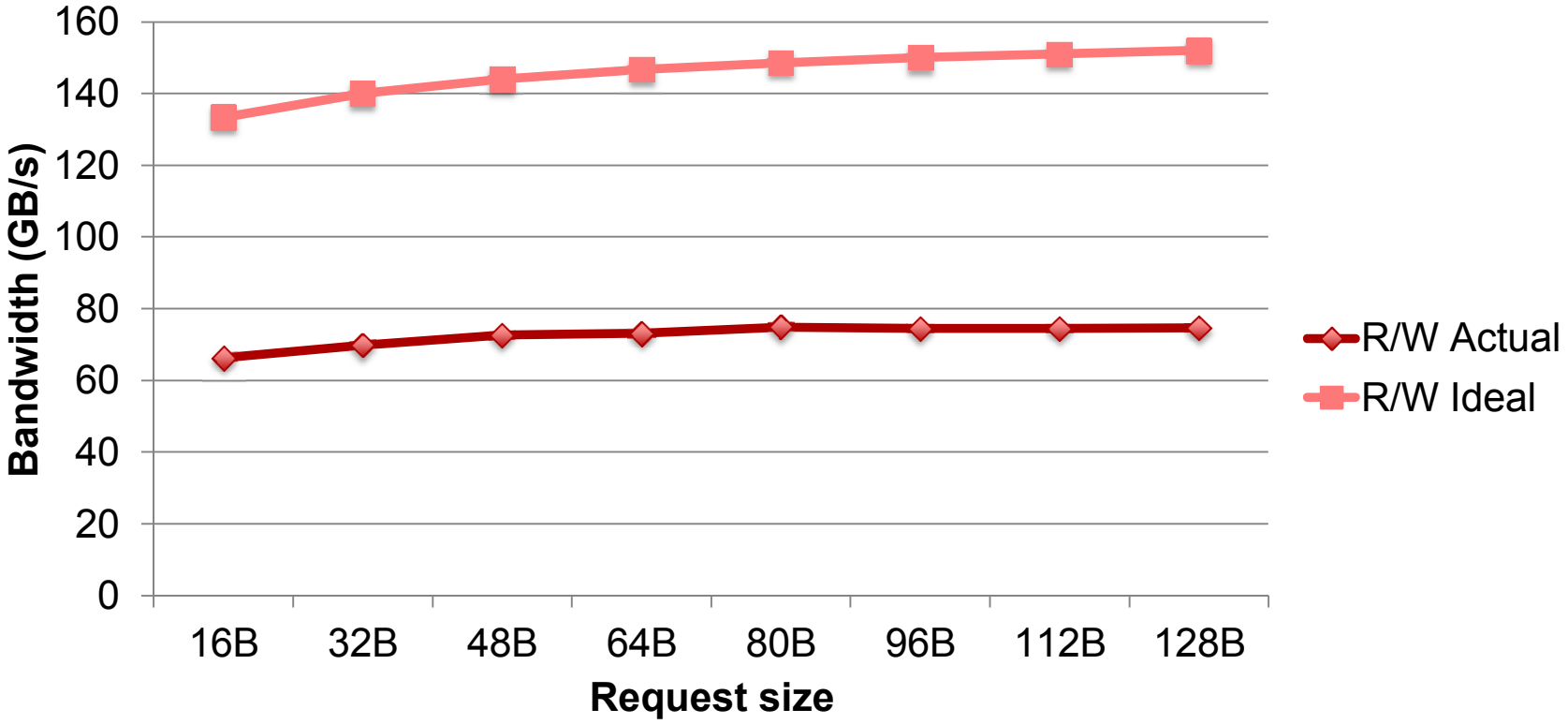
Preliminary results: Request size

Sandia National Laboratories

- Measure achievable bandwidth as a function of request size
  - Request size ranging from 16B to 128B
    - HMC configured for 128B blocks
  - Patterns
    - GUPS (random read-write addresses)
    - Random read-only addresses

GUPS bandwidth

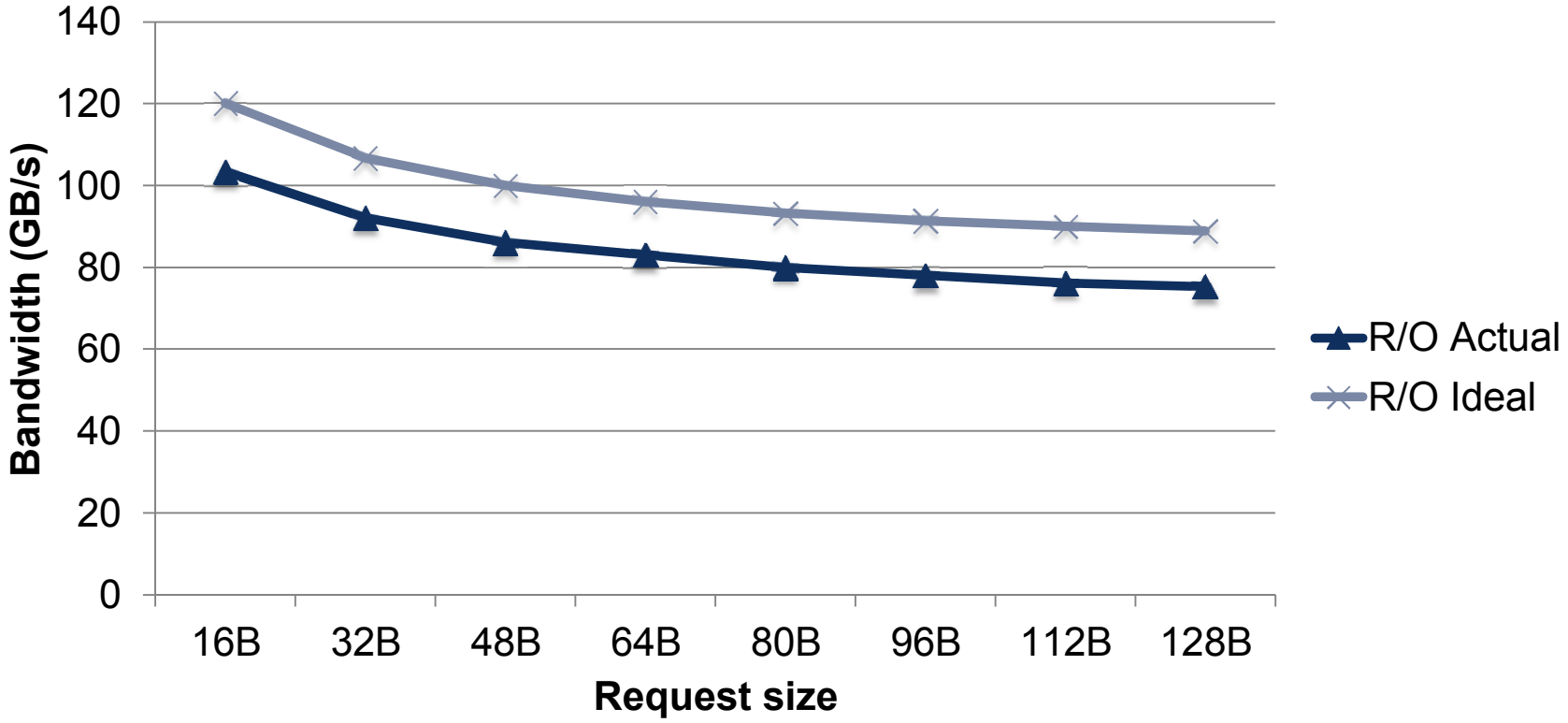
Including packet overhead



- Trends match
- Measured < ideal

Read-only random access bandwidth

Including packet overhead



- Trends match
- Measured close to ideal

Where we're headed

Sandia National Laboratories

- No one-size solution to MLM for our applications
  - Huge design space
  - Replacement/addition policies
    - Dynamic, static, programmer-assisted?
  - If programmer-assisted, can we assist programmer?
- HMC testbeds
  - Assessing performance with trace & pattern generators
  - Data layout, atomics