Performance of Variant Memory Configurations for Cray XT Systems

NATIONAL CENTER
FOR COMPUTATIONAL SCIENCES

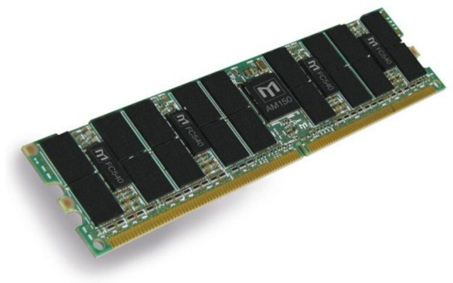
presented by

Wayne Joubert

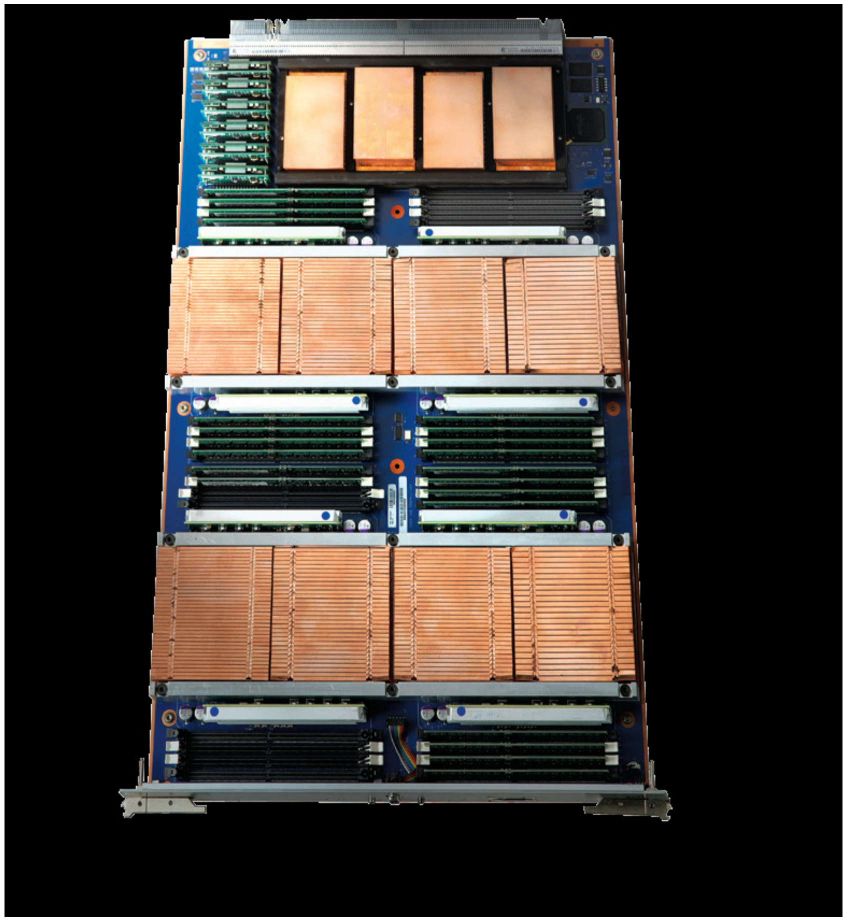
Oak Ridge National Laboratory
U.S. Department of Energy

Motivation

- Design trends are leading to non-power of 2 core counts for multicore processors, due to layout constraints – e.g., AMD Istanbul (6), Intel Dunnington (6), ...
- This complicates memory configuration choices, which often have multiple of 2 restrictions on populating the DIMM slots for compute nodes
- In 2009 NICS will upgrade from Barcelona CPU (4 cores/socket) to Istanbul (6 cores/socket), to bring Kraken to a 1 PF system
- The purpose of this study is to evaluate memory configurations for the new system, to determine how to maintain at least 1 GB memory per processor core and also give good memory performance

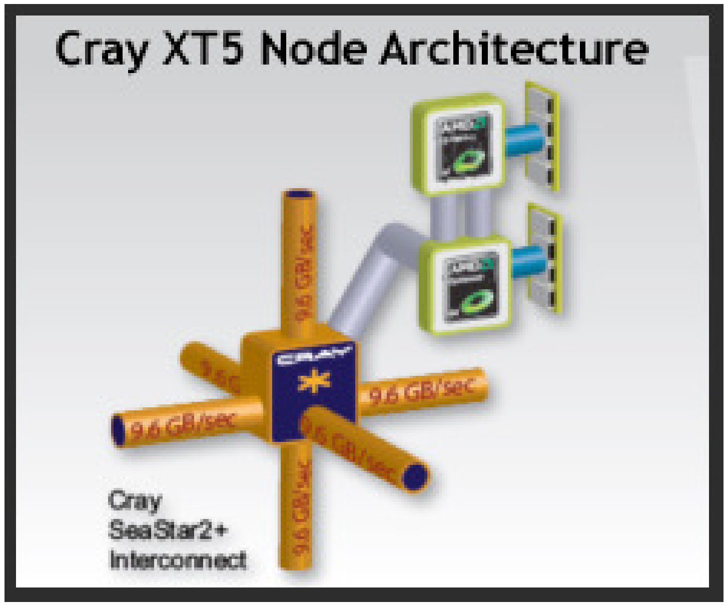


Cray XT5 Compute Blade



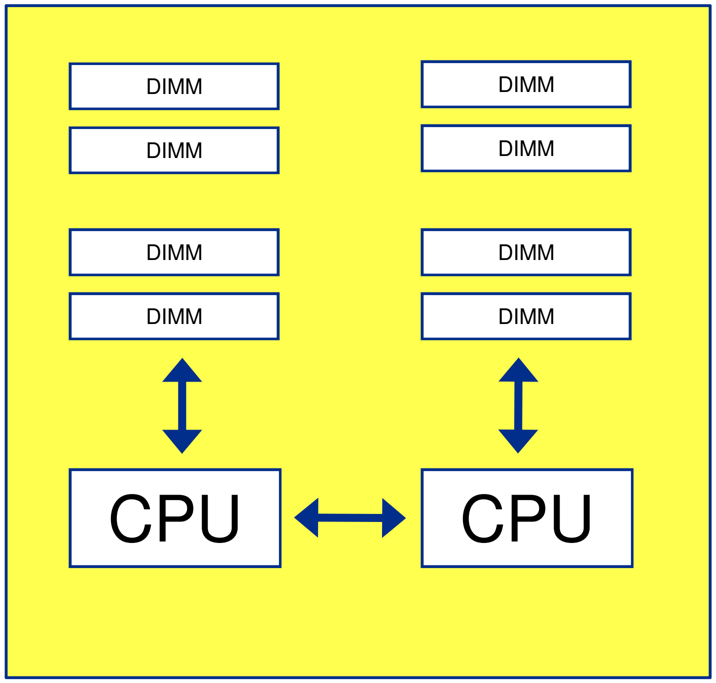
- Each blade has 4 compute nodes
- Each node has 2 CPU sockets
- The 2 CPU sockets of a node each access 8 DDR2 memory slots

Cray XT5 Compute Node Architecture



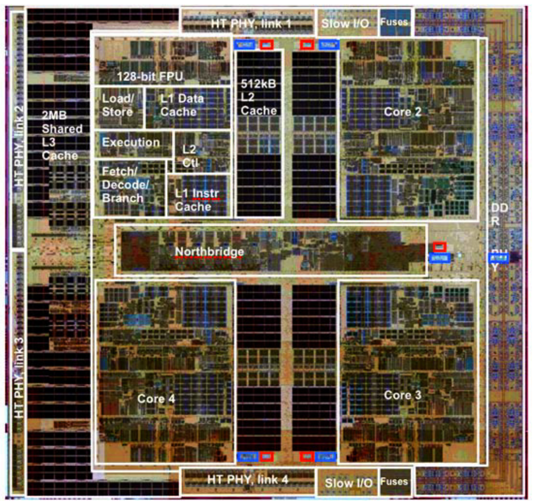
- 2 CPU sockets of the node are connected to SeaStar2+ interconnect
- 2 CPU sockets of the node are connected to each other via HyperTransport link
- Each CPU has its own on-chip memory controller to access its memory

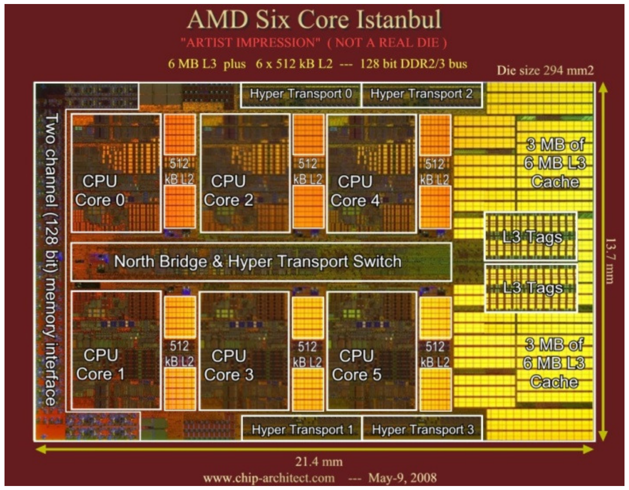
Cray XT5 Compute Node Memory



- Each CPU socket directly accesses 2 banks of 2 DIMM memory slots each, total 4 DIMM slots per socket
- NUMA configuration – each socket can access the other socket's memory, but at lower bandwidth
- Each bank "should" be either empty or fully populated with DIMMs of the same capacity
- What is the best way to populate the DIMM slots?

Processor Types Considered



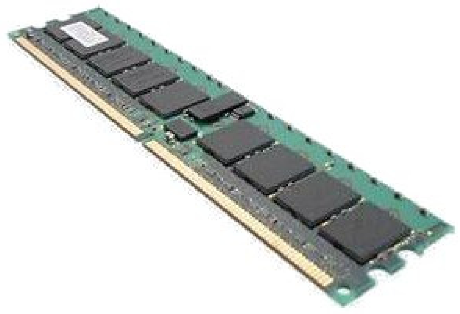


- AMD Barcelona, 4 cores/die, 2.3 GHz
- AMD Istanbul, 6 cores/die

Benchmark Systems

- For experiments, use ORNL “Chester” Cray XT5, 448 compute core TDS -- small version of JaguarPF
- Swap DIMM configurations, run experiments

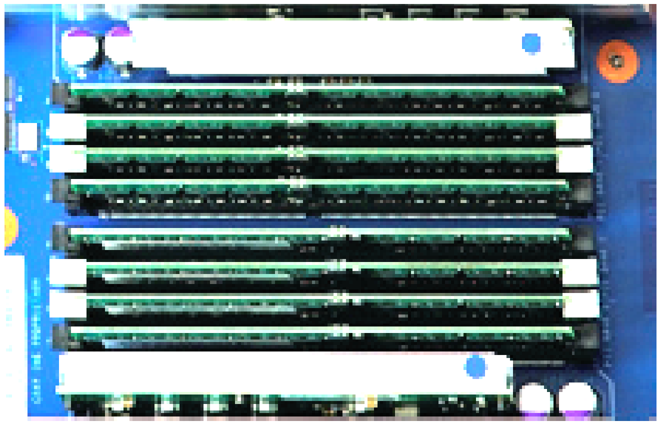
Memory Types



- Chester:
  - DDR2-800 4GB DIMMs
  - DDR2-667 2GB DIMMs
  - DDR2-533 8GB DIMMs

- Goal: evaluate performance of XT5 system for different configurations of memory DIMMs for each socket of a compute node

Memory Configurations



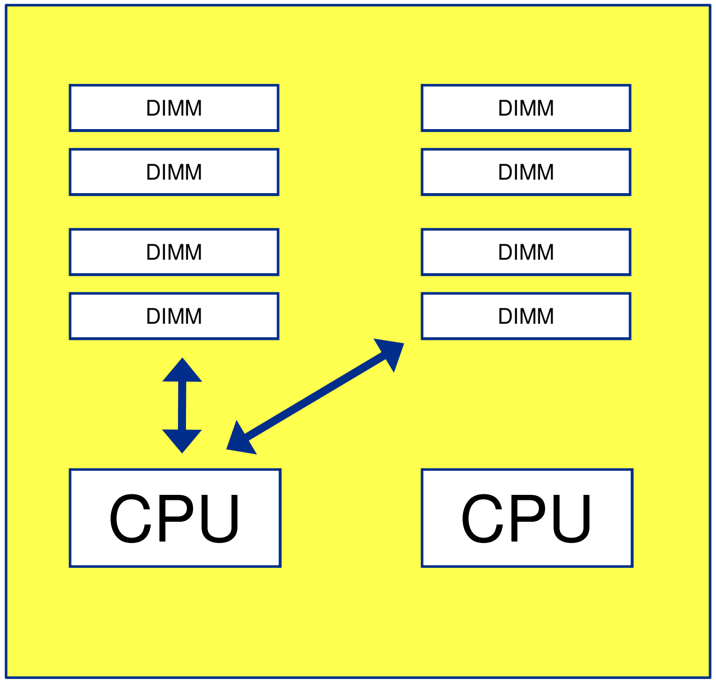
- Chester node:
  - 4-4-0-0, 4-4-0-0, (2 GB/core balanced)
  - 4-4-0-0, 2-2-0-0, (1.5 GB/core unbalanced)
  - 2-2-2-2, 2-2-0-0, (1.5 GB/core unbalanced)
  - 2-2-2-0, 2-2-2-0, (1.5 GB/core balanced)
  - 4-4-2-2, 4-4-2-2, (3 GB/core balanced)
  - 8-8-0-0, 8-8-0-0, (4 GB/core balanced)

Our particular concern: What is the penalty of using off-socket memory for the unbalanced cases?

Codes for Benchmark Tests

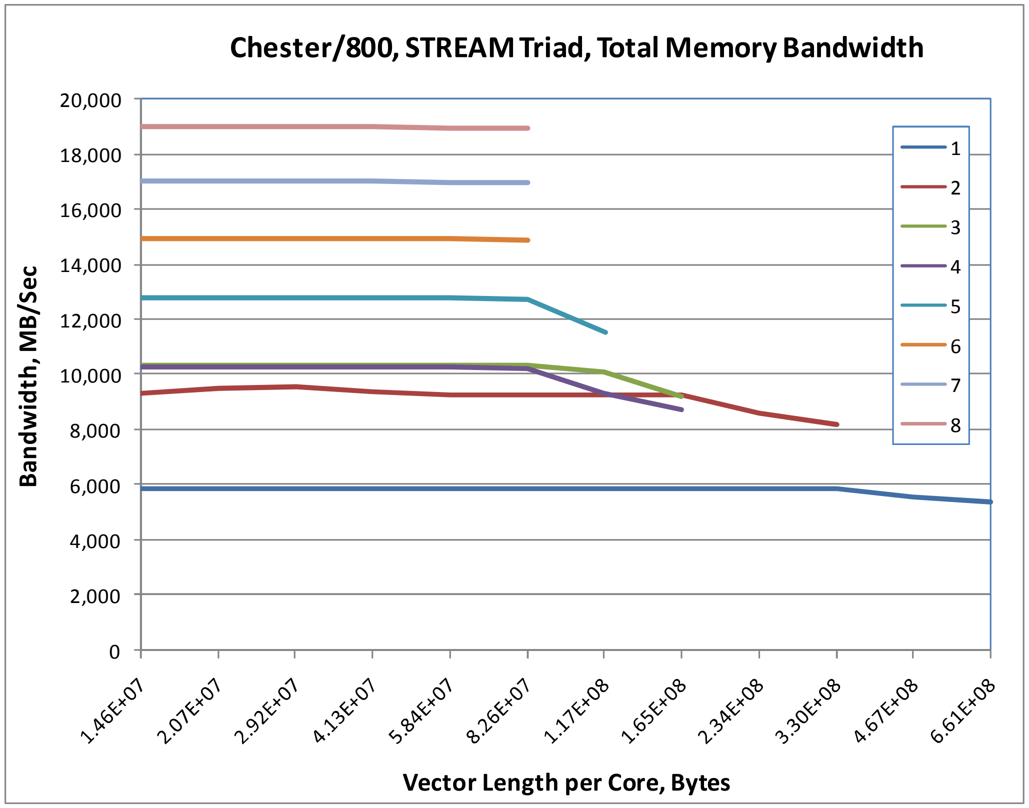
1. STREAM Benchmark.
    - Measures memory bandwidth for several kernels
    - Use TRIAD kernel $z = y + a\,x$
2. DAXPY kernel $y = y + a\,x$.
3. LMBENCH – measures memory latency.
4. S3D application code – petascale combustion application that uses structured 3-D meshes. Performance is typically memory-bound.

Experiments: Memory Spillage Effects



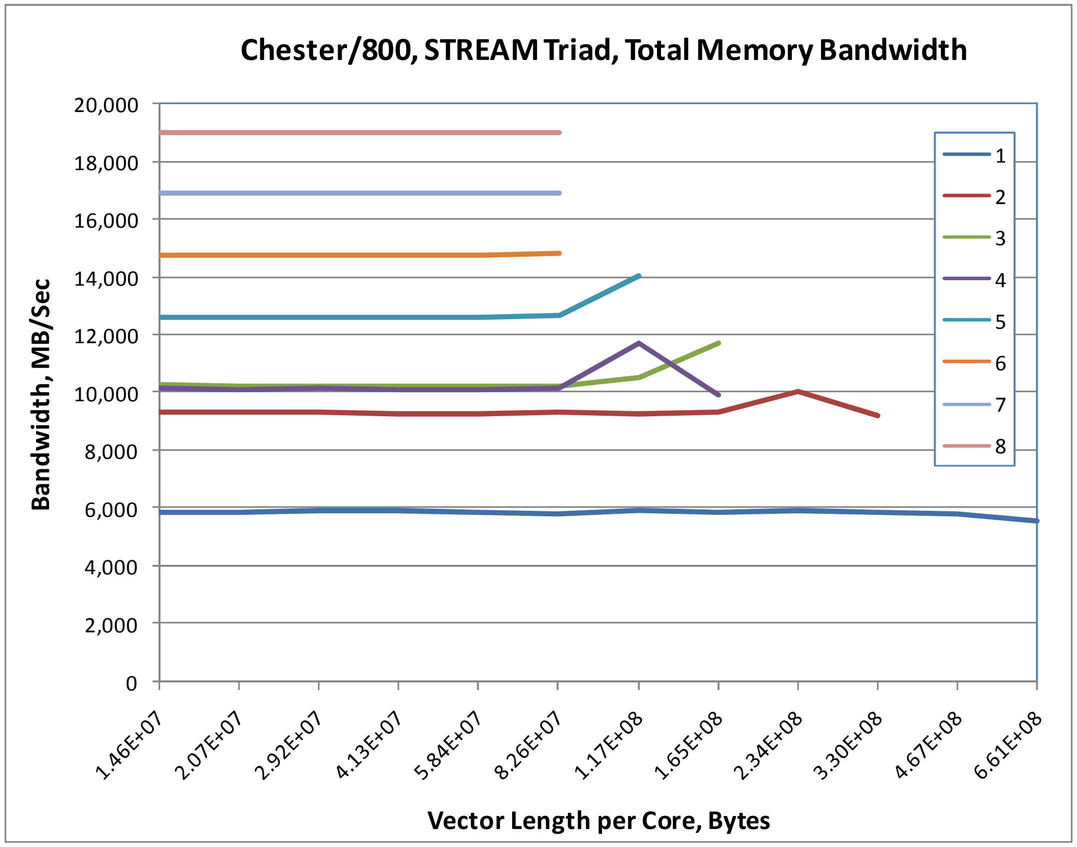
- Execute benchmark on one CPU socket
- Ramp up problem size / memory usage until memory spills off-socket
- Measure effects of using off-socket memory

Memory Spillage Effects: STREAM



- Run on 1-8 cores
- Chester, balanced config -- 4-4-0-0, 4-4-0-0
- Peak memory bandwidth:
  - 25.6 GB/sec theoretical
  - 21.2 GB/sec actual
- See up to 15% decrease in performance when spilling memory references off-socket
- Note: STREAM puts related array entries z(i), x(i), y(i) all on same memory page – Linux first touch policy

Memory Spillage Effects: STREAM

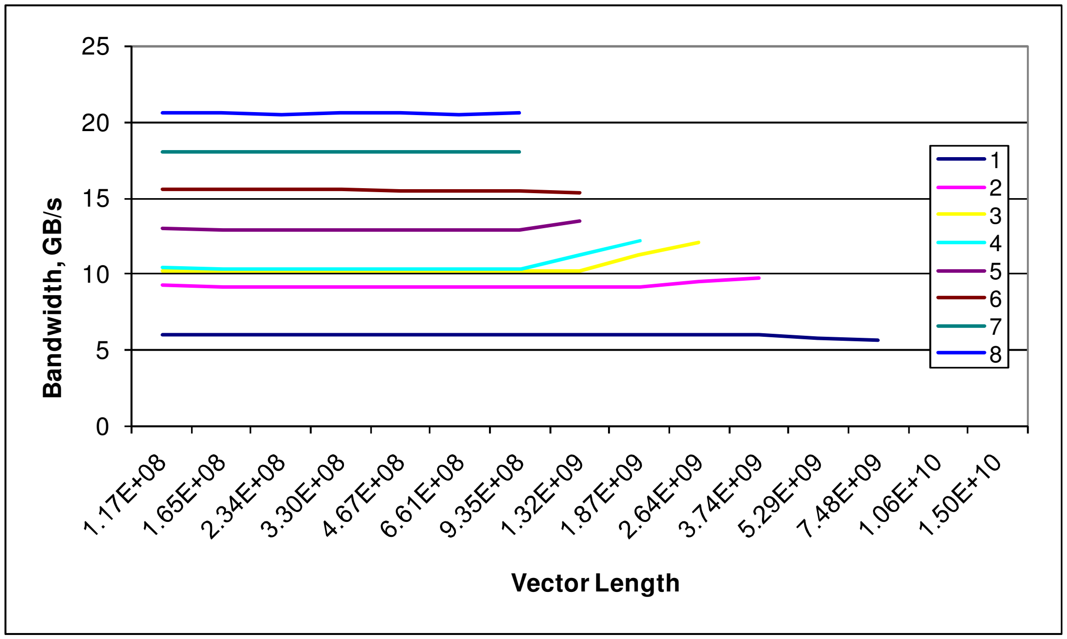


- Same experiment, change STREAM memory initialization to put z(i), x(i), y(i) for same i on different pages, potentially different sockets

- Performance uptick – can get higher performance from accessing on-socket and off-socket memory concurrently

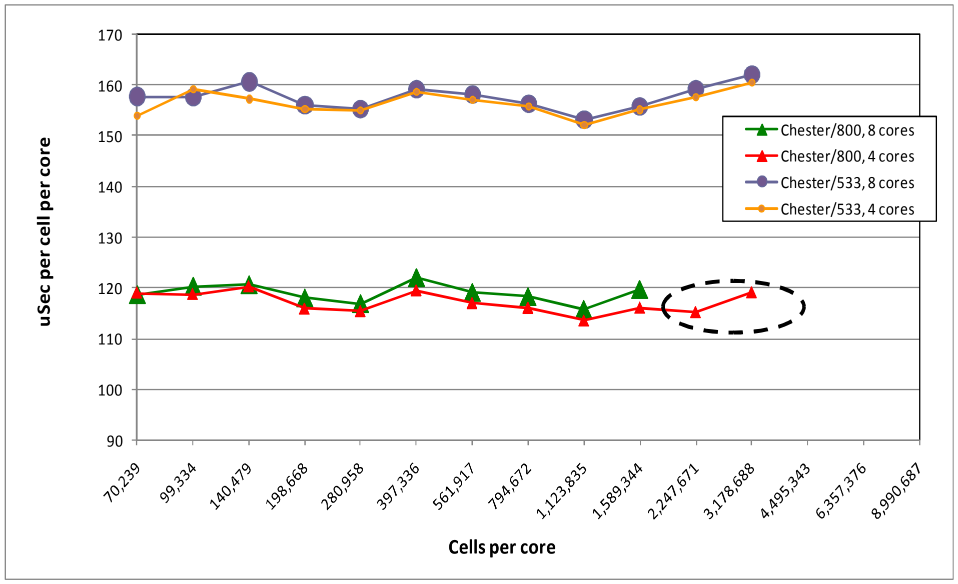
- Not helpful for typical use case of using all cores for computation

Memory Spillage Effects: DAXPY



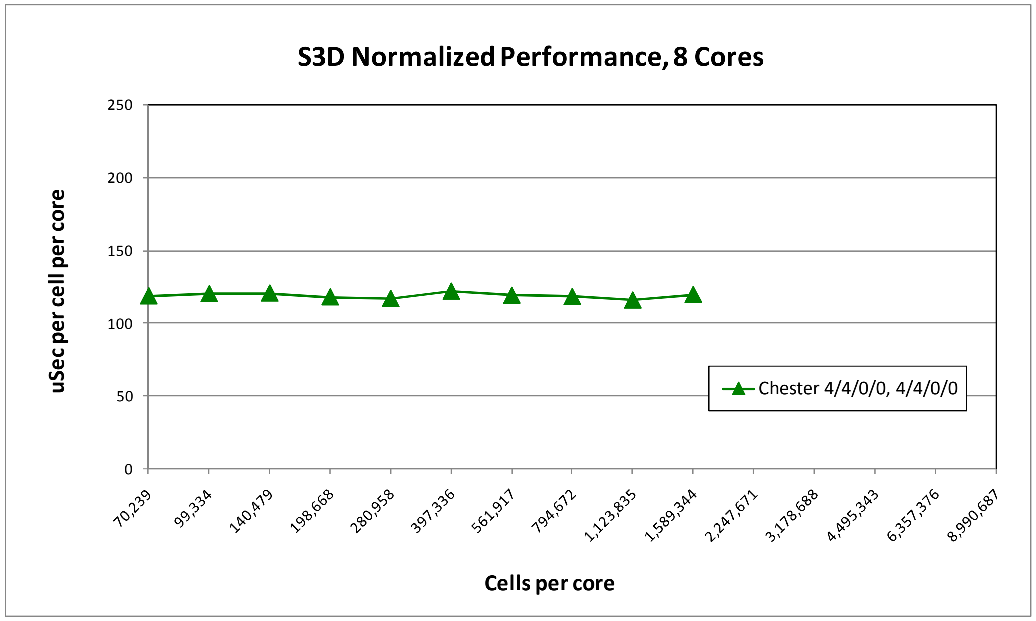
- Memory for y(i), x(i) on different pages
- Similar uptick in bandwidth for off-socket memory references

Memory Spillage Effects: S3D



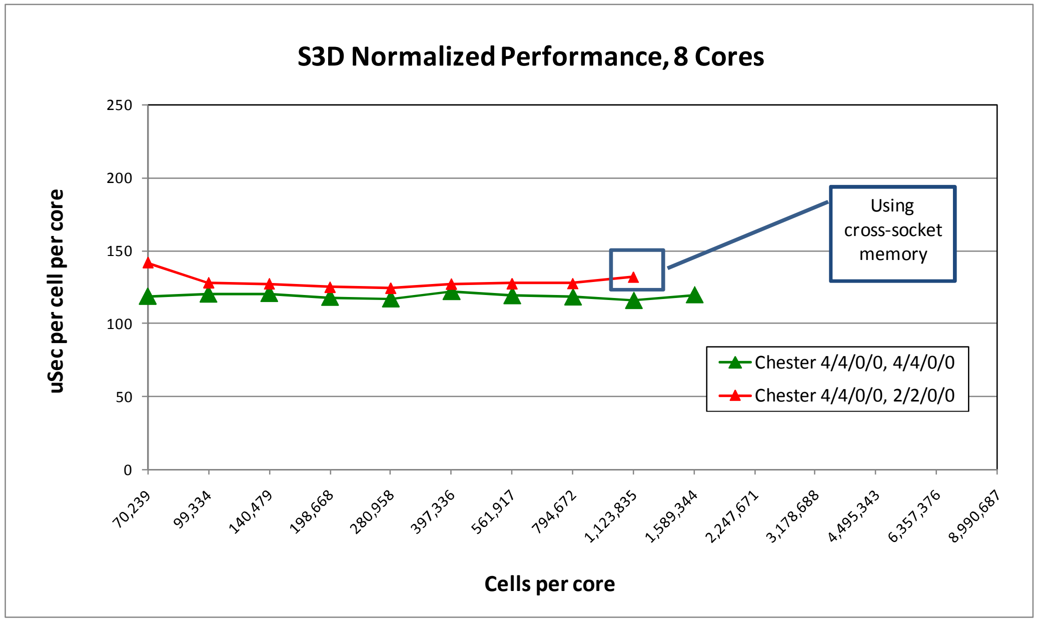
- S3D application code
- Chester 4-4-0-0, 4-4-0-0 (DDR2-800)
- Chester 8-8-0-0, 8-8-0-0 (DDR2-533)
- Vary grid cells per core
- Graph: wallclock time microseconds per gridcell per core
- Run on 1 socket or 2 sockets of node
- Observe spillage effects for 1 socket case
- Effects of memory spillage minimal

Memory Configuration Effects: S3D: Chester 4-4-0-0, 4-4-0-0



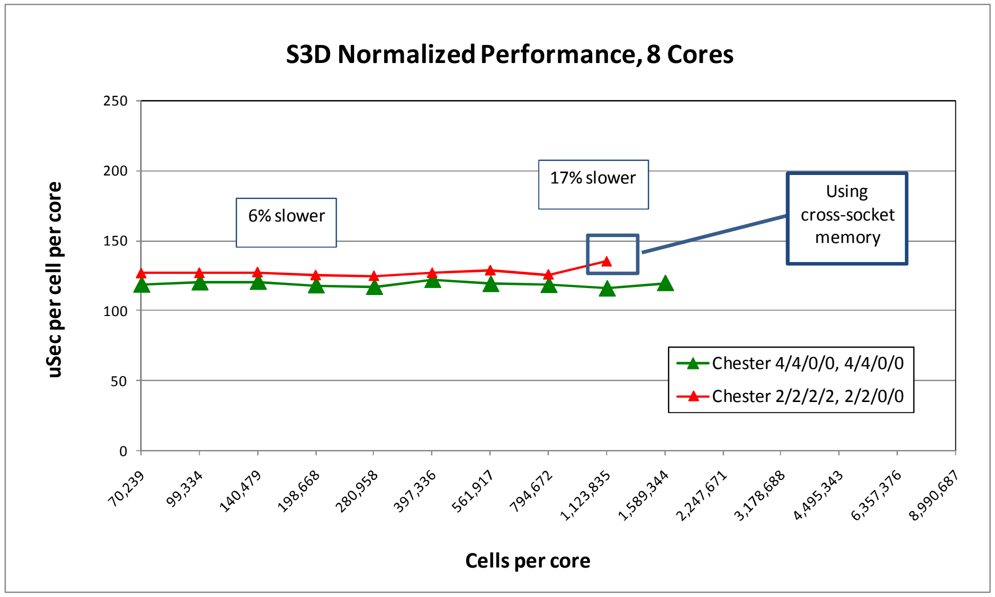
- Now compare different memory configurations
- Run on both sockets
- Baseline case: 4-4-0-0, 4-4-0-0, balanced between sockets
- All memory references are on-socket

Memory Configuration Effects: S3D: Chester 4-4-0-0, 2-2-0-0



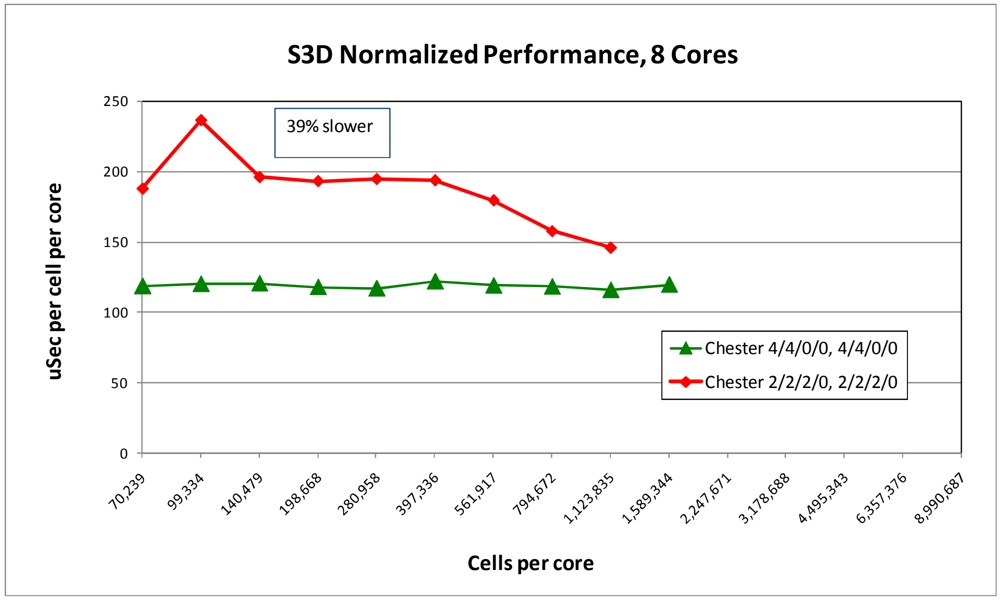
- Chester 4-4-0-0, 2-2-0-0
- Unbalanced between sockets
- For large memory cases, socket with less memory takes memory from other socket
- Memory performance slightly worse overall
- Memory performance slightly worse when thin-memory socket uses off-socket memory

Memory Configuration Effects: S3D: Chester 2-2-2-2, 2-2-0-0



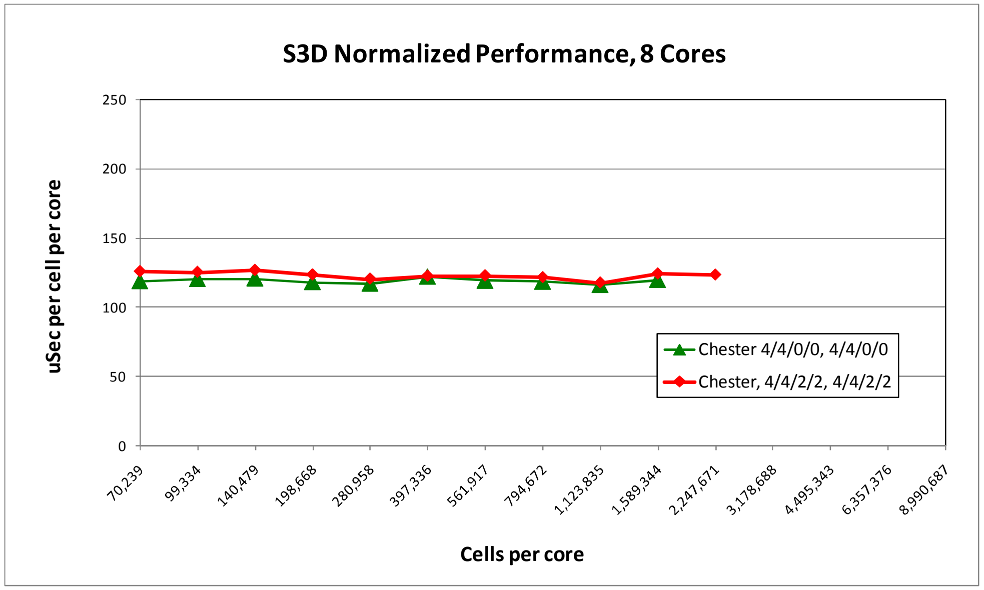
- Chester 2-2-2-2, 2-2-0-0
- Unbalanced between sockets
- Memory performance slightly worse overall
- Memory performance slightly worse when thin-memory socket uses off-socket memory

Memory Configuration Effects: S3D: Chester 2-2-2-0, 2-2-2-0



- Chester 2-2-2-0, 2-2-2-0
- Balanced between sockets
- Unsupported memory configuration – one bank is half-full
- Significantly worse memory performance
- Believed to be due to the way memory is striped across DIMMs in the bank

Memory Configuration Effects: S3D: Chester 4-4-2-2, 4-4-2-2

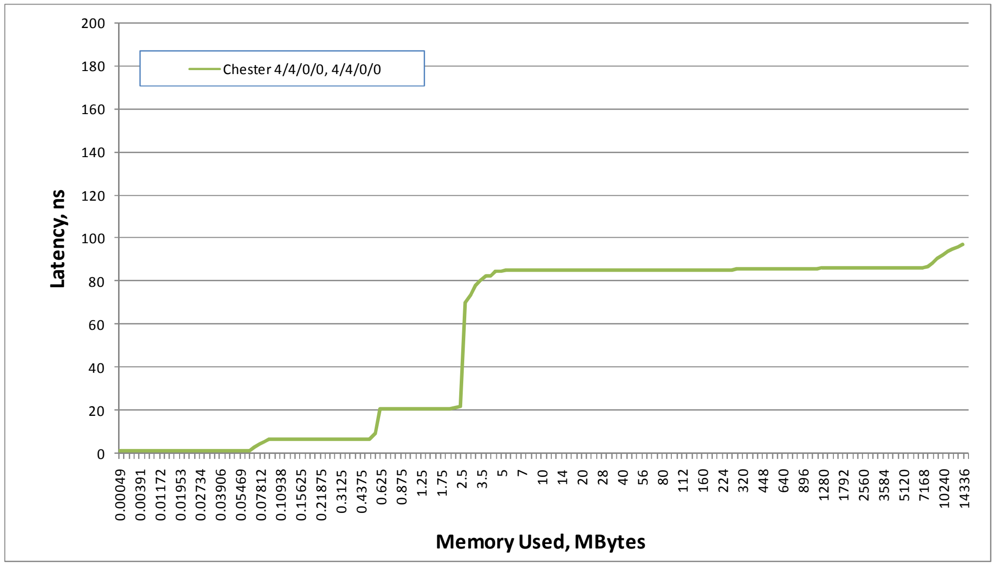


- Chester 4-4-2-2, 4-4-2-2
- Balanced between sockets
- Fat-memory configuration
- Similar performance to baseline case

Memory Configuration Effects: S3D: Conclusions

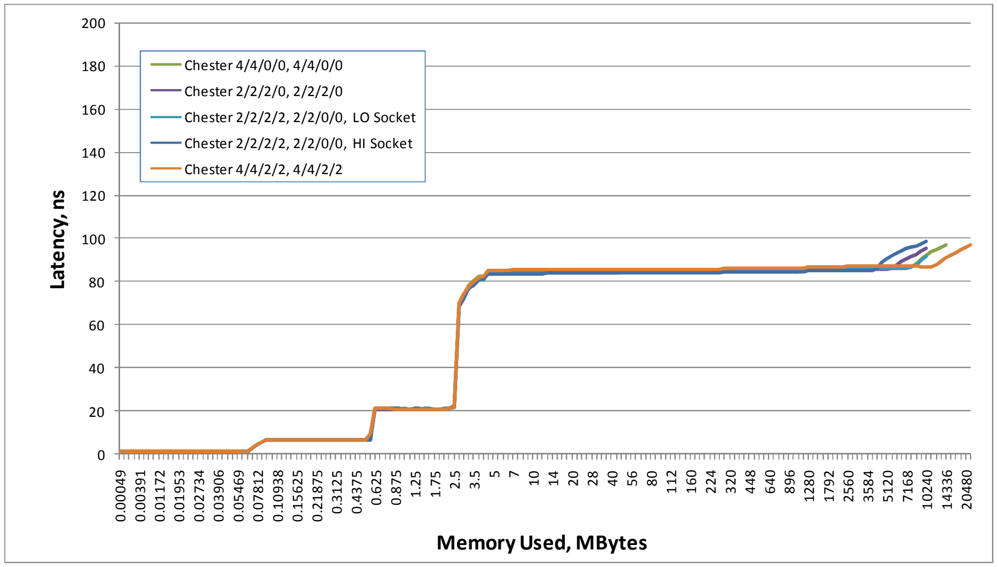
- Performance loss from imbalanced configurations is at most ~ 17%
- Balanced (unsupported) memory configuration has much worse performance

Memory Configuration Effects: LMBENCH: Chester 4-4-0-0, 4-4-0-0



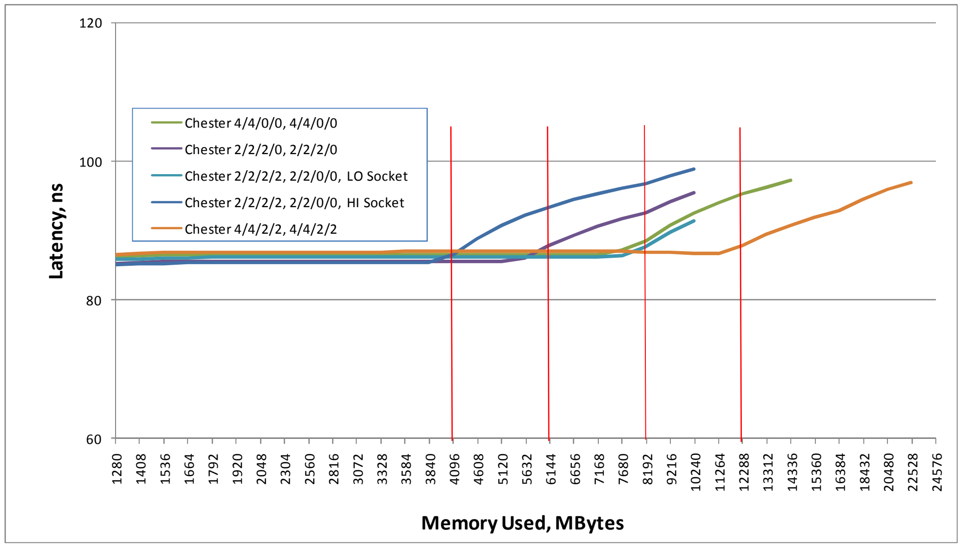
- Measures array load latency based on array length
- Run on 1 core
- Clearly see L-1/2/3 cache effects
- Higher latency when some of array is off-socket
- Baseline case: 4-4-0-0, 4-4-0-0

Memory Configuration Effects: LMBENCH



- Balanced and unbalanced memory configurations
- All cases have similar performance
- Memory configuration has no significant impact on latency

Memory Configuration Effects: LMBENCH



- Detail of previous graph
- On-socket latency ~ 86 ns
- Off-socket latency ~ 102-108 ns depending on configuration

Conclusions

- Impact of unbalanced memory configuration on memory bandwidth less than expected: ~ 20% at worst
  - Would not affect apps that don't use much memory
  - Would not affect apps that are not memory-bound
- Balanced (but unsupported) memory configuration performs very poorly – half-empty memory bank appears to run at half speed
- Memory latency is unaffected by any change in memory configuration
- In some rare cases there could be advantage to using on- and off-socket memory in parallel