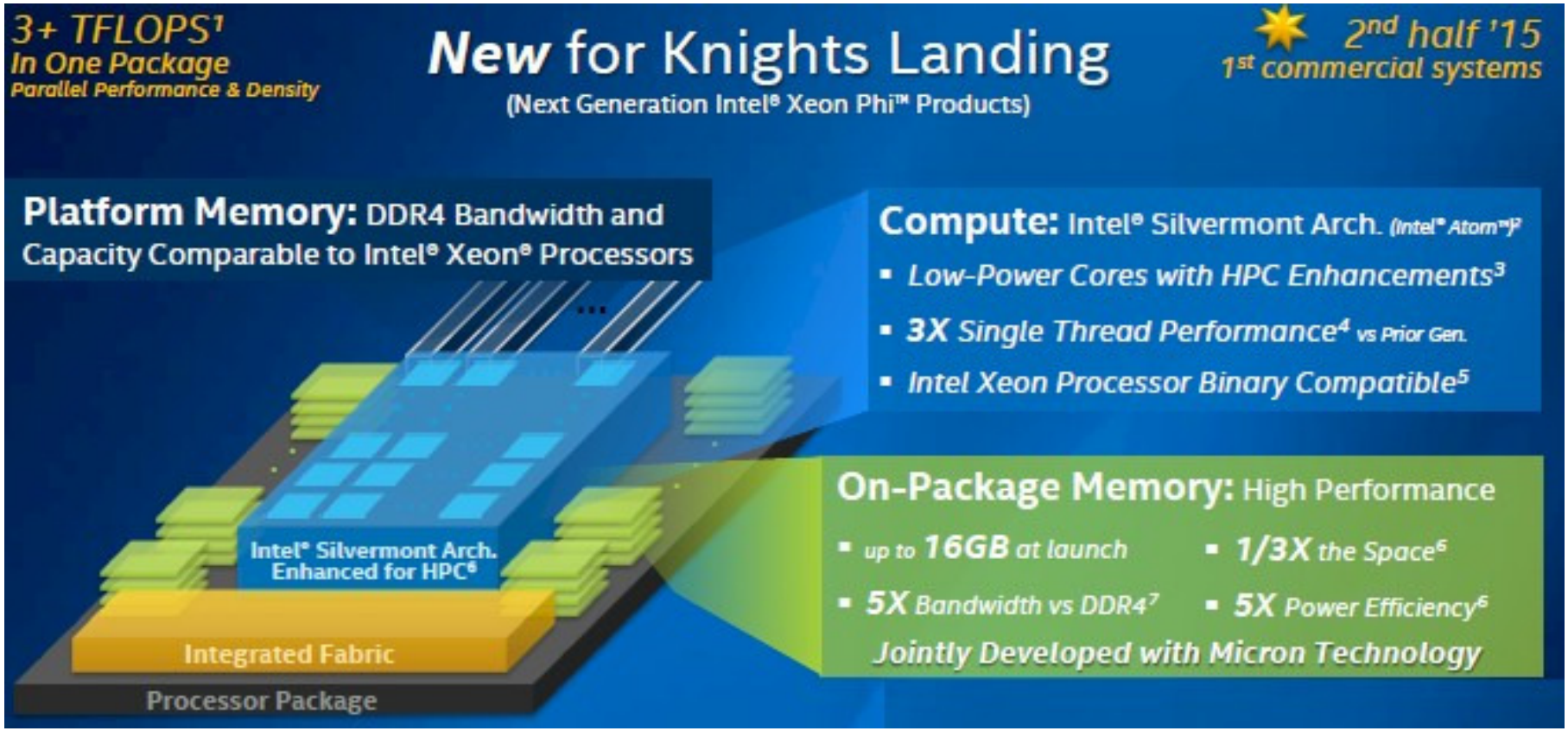
KNL EXPERIENCES

Adrian Jackson
adrianj@epcc.ed.ac.uk
@adrianjhpc

epcc

KNL


3+ TFLOPS[1]
In One Package
Parallel Performance & Density
New for Knights Landing
(Next Generation Intel® Xeon Phi™ Products)
2nd half '15
1st commercial systems
Platform Memory: DDR4 Bandwidth and Capacity Comparable to Intel® Xeon® Processors
Compute: Intel® Silvermont Arch. (Intel® Atom™)[2]
Low-Power Cores with HPC Enhancements[3]
3X Single Thread Performance[4] vs Prior Gen.
Intel Xeon Processor Binary Compatible[5]
On-Package Memory: High Performance
up to 16GB at launch
1/3X the Space[6]
5X Bandwidth vs DDR4[7]
5X Power Efficiency[6]
Jointly Developed with Micron Technology
Intel® Silvermont Arch. Enhanced for HPC[6]
Integrated Fabric
Processor Package

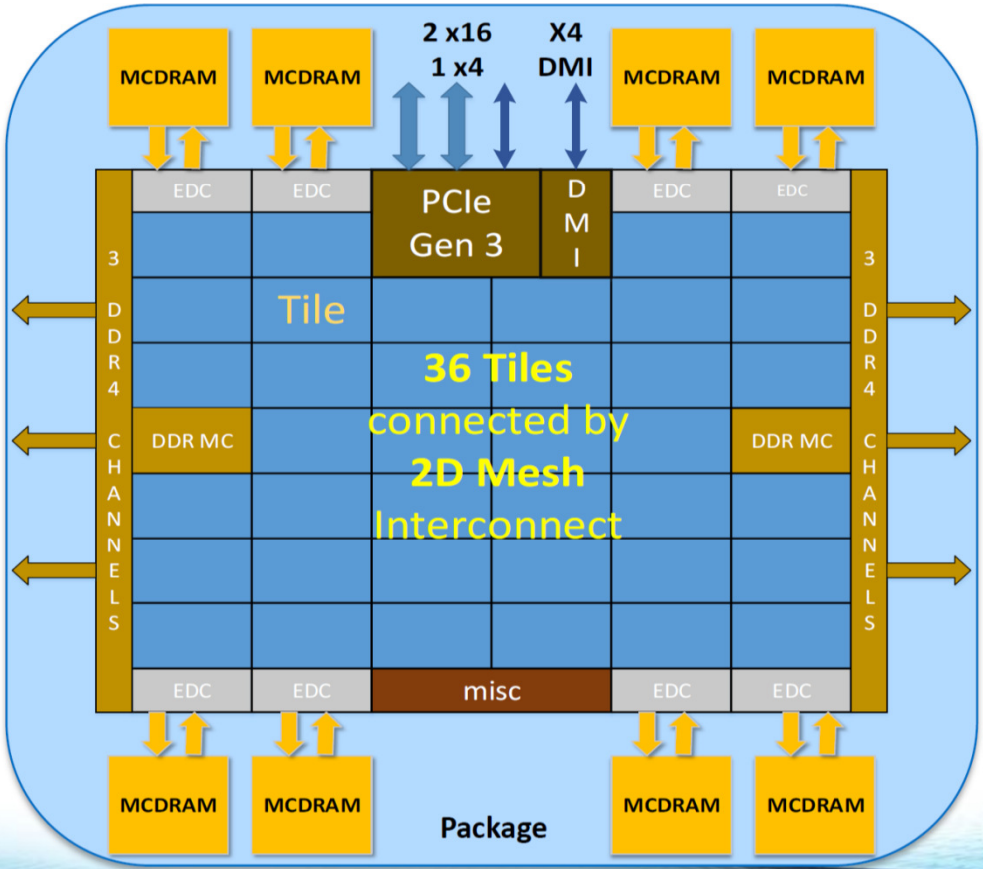
KNL


KNIGHT'S LANDING
NEXT-GEN INTEL® XEON PHI™
PROCESSOR
CPU
MCDRAM
DDR
NAND SSD
Hard Disk Drives
3X
FASTER
ST PERFORMANCE
VS. KNC
5X
FASTER
MCDRAM VS.
DDR4 DIMMs

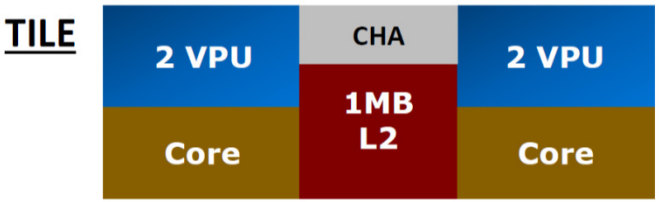
epcc

KNL

Knights Landing Overview



Avinash Sodani CGO PPoPP HPCA Keynote 2016



Chip: up to **36 Tiles** interconnected by **2D Mesh**

Tile: 2 Cores + 2 VPU/core + 1 MB L2

Memory: MCDRAM: up to 16 GB on-package; High BW

DDR4: 6 channels @ 2400 up to 384GB

IO: 36 lanes PCIe Gen3. 4 lanes of DMI for chipset

Node: 1-Socket

Fabric: Intel® Omni-Path Fabric on-package (not illustrated)

Vector Peak Perf: 3+TF DP and 6+TF SP Flops

Scalar Perf: ~3x over Knights Corner

Streams Triad (GB/s): MCDRAM : 450+; DDR: ~90

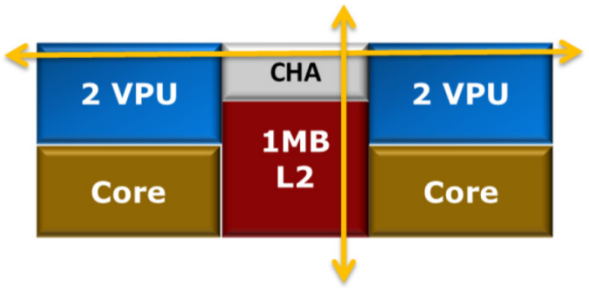
Note: not all specifications shown apply to all Knights Landing SKUs

Source Intel: All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice. KNL data are preliminary based on current expectations and are subject to change without notice. 1Binary Compatible with Intel Xeon processors using Haswell Instruction Set (except TSX). 2Bandwidth numbers are based on STREAM-like memory access pattern when MCDRAM used as flat memory. Results have been estimated based on internal Intel analysis and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

epcc

KNL

KNL Tile: 2 Cores, each with 2 VPU
1M L2 shared between two Cores



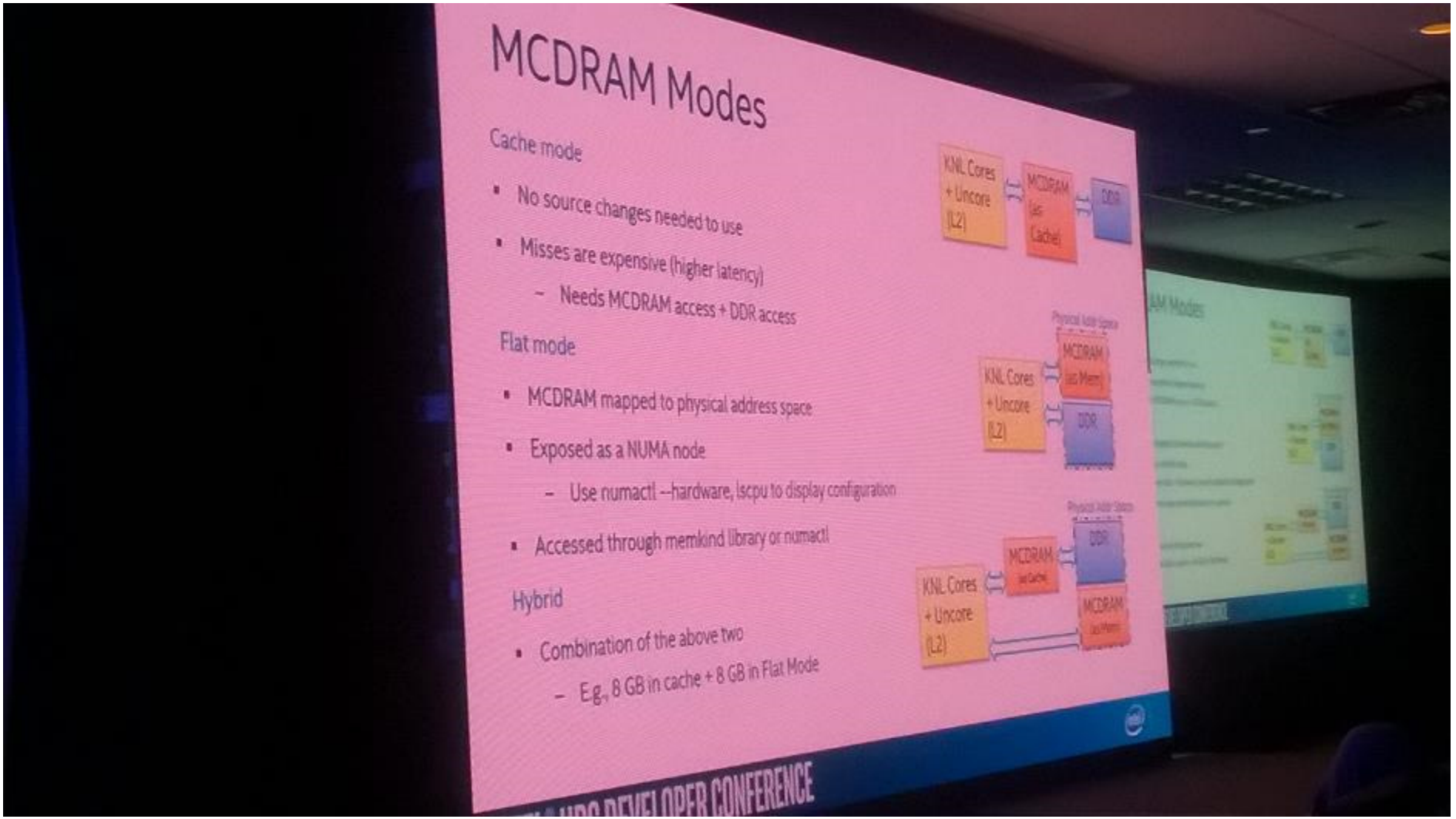
Core: New OoO Core. Balances power efficiency, parallel and single thread performance.

2 VPU: 2x AVX512 units. 32SP/16DP per unit. X87, SSE, AVX, AVX2 and EMU

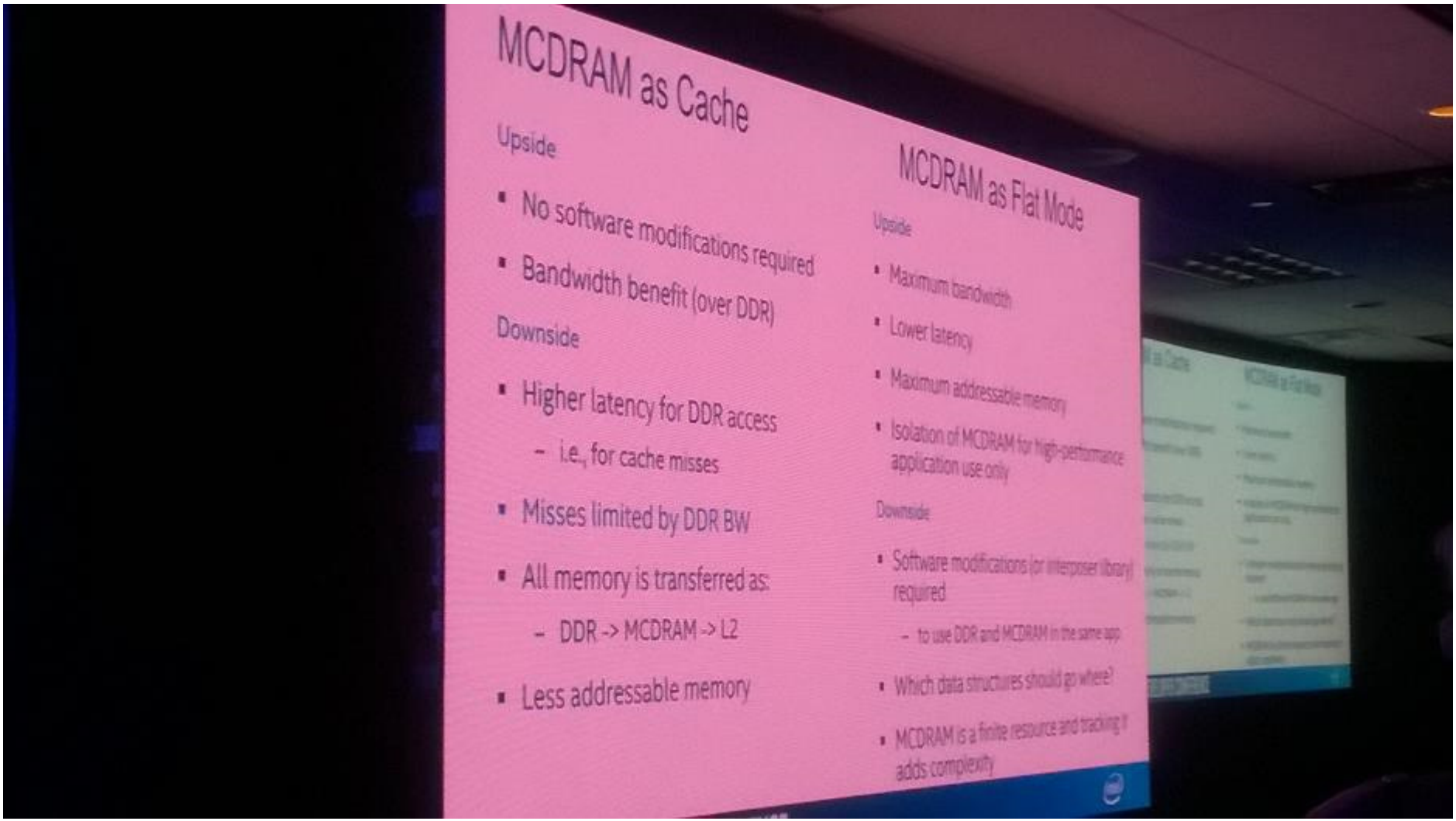
L2: 1MB 16-way. 1 Line Read and ½ Line Write per cycle. Coherent across all Tiles

CHA: **C**aching/**H**ome **A**gent. Distributed Tag Directory to keep L2s coherent. MESIF protocol. 2D-Mesh connections for Tile

KNL


MCDRAM Modes
Cache mode
No source changes needed to use
Misses are expensive (higher latency)
Needs MCDRAM access + DDR access
Flat mode
MCDRAM mapped to physical address space
Exposed as a NUMA node
Use numactl --hardware, lscpu to display configuration
Accessed through memkind library or numactl
Hybrid
Combination of the above two
E.g., 8 GB in cache + 8 GB in Flat Mode
KNL Cores + Uncore (L2)
MCDRAM (as Cache)
DDR
MCDRAM
DDR
KNL Cores + Uncore (L2)
MCDRAM
DDR
KNL Cores + Uncore (L2)
MCDRAM

KNL


MCDRAM as Cache
Upside
No software modifications required
Bandwidth benefit (over DDR)
Downside
Higher latency for DDR access
– i.e., for cache misses
Misses limited by DDR BW
All memory is transferred as:
– DDR -> MCDRAM -> L2
Less addressable memory
MCDRAM as Flat Mode
Upside
Maximum bandwidth
Lower latency
Maximum addressable memory
Isolation of MCDRAM for high-performance application use only
Downside
Software modifications (or interposer library) required
– to use DDR and MCDRAM in the same app
Which data structures should go where?
MCDRAM is a finite resource and tracking it adds complexity

epcc

KNL

- Example code:
  - Check available memory

```
[Xajacks@eln4 Mg2SiO4-geom]$ numactl --hardware
available: 2 nodes (0-1)
node 0 cpus: 0 2 4 6 8 10 12 14 16 18 20 22
node 0 size: 49090 MB
node 0 free: 32586 MB
node 1 cpus: 1 3 5 7 9 11 13 15 17 19 21 23
node 1 size: 49152 MB
node 1 free: 28820 MB
node distances:
node   0   1
  0:  10  21
  1:  21  10
```

  - Fails if exhausts memory

```
mpirun -n 64  numactl -m 1 ./castep.mpi forsterite
```

  - Tries to used preferred memory, falls back if exhausts memory

```
mpirun -n 64  numactl -p 1 ./castep.mpi forsterite
```

KNL

A Heterogeneous Memory Management Framework

MEMKIND

- Defines a plug-in architecture.
- Each plug-in is called a “kind” of memory.
- Built on top of jemalloc: the FreeBSD OS default heap manager.
- Partition is defined by functions that provide inputs for operating system calls.
- High level memory management functions can be over-ridden as well.
- https://github.com/memkind

HBWMALLOC

- Implements easy model for KNL.
- Implemented using memkind; simplifies plug-in (kind) selection.
- Provides support for 2MB and 1GB pages.
- Select fallback behavior when on package memory does not exist or is exhausted.
- Check for existence of on package memory.

Jeff Hammond
Intel Parallel Computing Lab

epcc

End Goal Usage: Code Snippets

Heap allocation in C

```
float * fv1 = malloc(sizeof(float) * 1000);
float * fv2 = hbw_malloc(sizeof(float) * 1000);
```

Allocatable arrays in Fortran

```
    REAL, ALLOCATABLE :: A(:), B(:), C(:)
!DIR$ ATTRIBUTES FASTMEM :: A
    NSIZE=1000
!   allocate array 'A' from MCDRAM
    ALLOCATE (A(1:NSIZE))
!   Allocate arrays that will come from DDR
    ALLOCATE  (B(1:NSIZE), C(1:NSIZE))
```

Automatic variables will be allocated in DDR in flat mode.

This means you may need to convert from automatic to heap arrays or use hybrid mode if such data is used in a bandwidth-intensive way.

Standard containers in C++ (not documented upstream yet)

```
std::vector<float, hbwmalloc::hbwmalloc_allocator<float> > vec;
```

Jeff Hammond
Intel Parallel Computing Lab

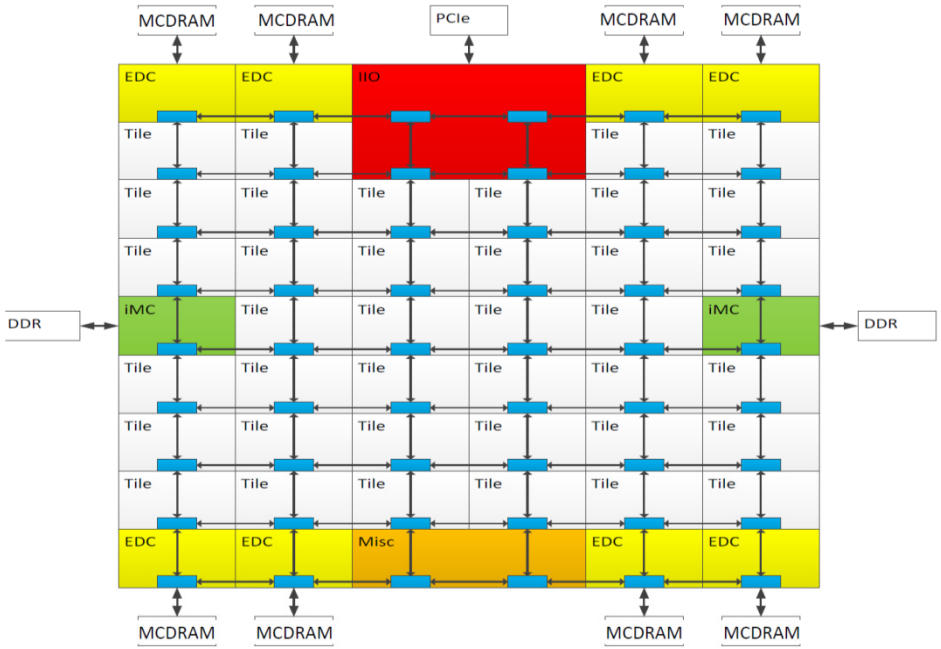
- Fortran:
  - FASTMEM is Intel directive
- Wrapped hbw_malloc
  - Call malloc directly in Fortran
  - https://github.com/jeffhammond/myhbwmalloc

```
use fortran_hbwmalloc
```

```
include 'mpif.h'
integer offset_kind
parameter(offset_kind=MPI_OFFSET_KIND)
integer(kind=offset_kind) ptr
INTEGER(C_SIZE_T) param
type(C_PTR) localptr
      real (kind=8) r8
      pointer (pr8, r8)
if (type.eq.'r8') then
        param = 8*dim
        localptr = hbw_malloc(param)
      else if (type.eq.'i4') then
        param = 4*dim
        localptr = hbw_malloc(param)
      end if
ptr = transfer(localptr,ptr)
    if (type.eq.'r8') then
      call c_f_pointer(localptr, pr8)
      call zeroall(dim,r8)
  end if
```

KNL

KNL Mesh Interconnect



Mesh of Rings

- Every row and column is a (half) ring
- YX routing: Go in Y → Turn → Go in X
- Messages arbitrate at injection and on turn

Cache Coherent Interconnect

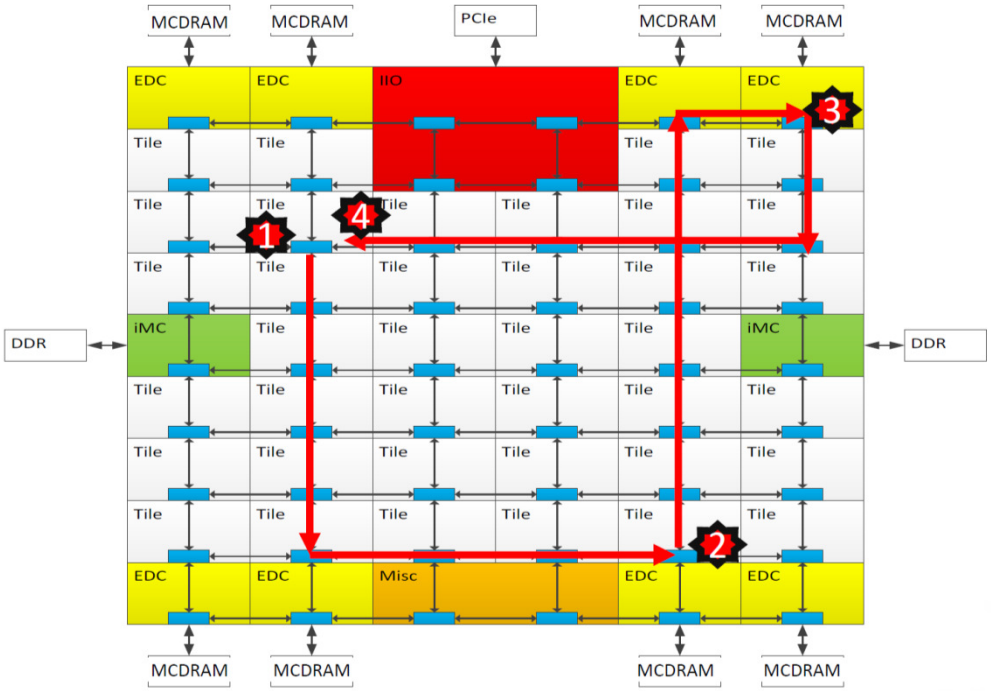
- MESIF protocol (F = Forward)
- Distributed directory to filter snoops

Three Cluster Modes

(1) All-to-All (2) Quadrant (3) Sub-NUMA Clustering

KNL

Cluster Mode: All-to-All



Address uniformly hashed across all distributed directories

No affinity between Tile, Directory and Memory

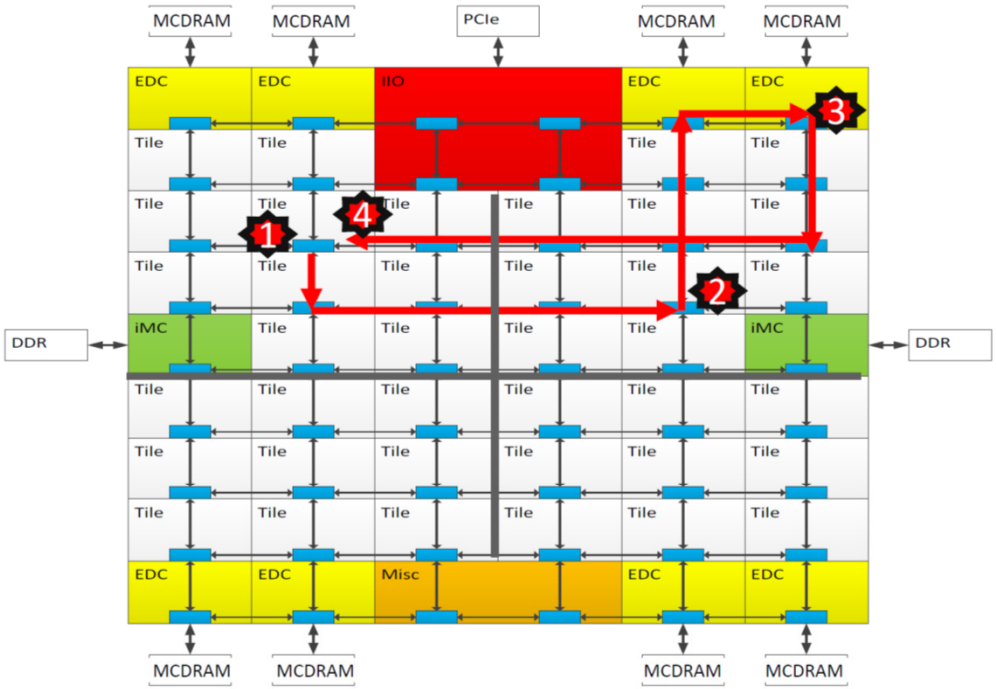
Most general mode. Lower performance than other modes.

Typical Read L2 miss

1. L2 miss encountered
2. Send request to the distributed directory
3. Miss in the directory. Forward to memory
4. Memory sends the data to the requestor

KNL

Cluster Mode: Quadrant



Chip divided into four virtual Quadrants

Address hashed to a Directory in the same quadrant as the Memory

Affinity between the Directory and Memory

Lower latency and higher BW than all-to-all. SW Transparent.

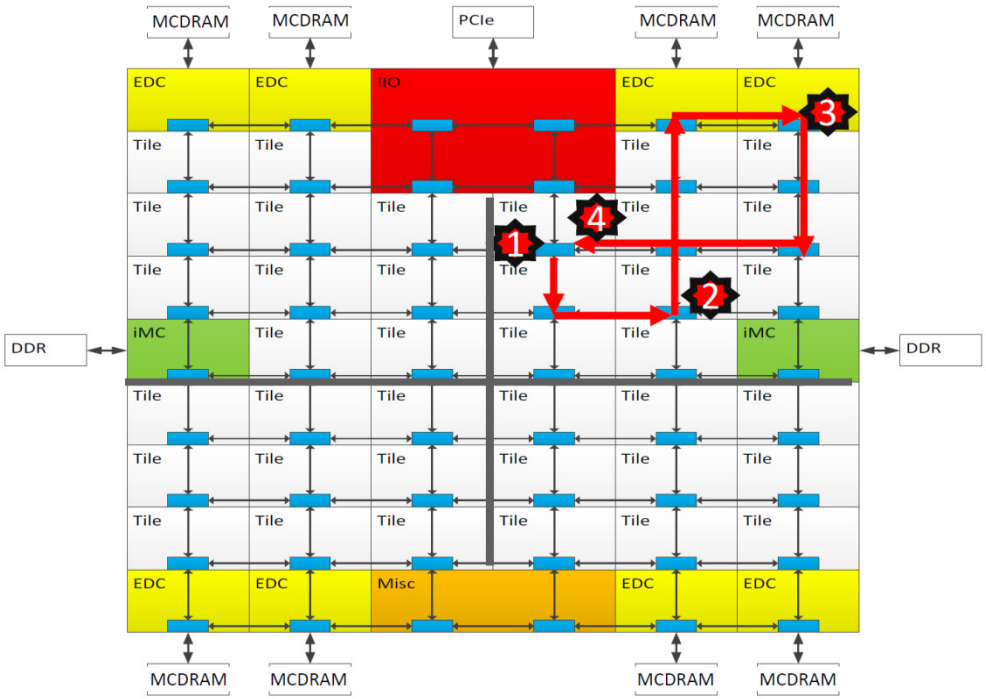
1) L2 miss, 2) Directory access, 3) Memory access, 4) Data return

Avinash Sodani CGO PPoPP HPCA Keynote 2016

epcc

KNL

Cluster Mode: Sub-NUMA Clustering (SNC)



Each Quadrant (Cluster) exposed as a separate NUMA domain to OS.

Looks analogous to 4-Socket Xeon

Affinity between Tile, Directory and Memory

Local communication. Lowest latency of all modes.

SW needs to NUMA optimize to get benefit.

1) L2 miss, 2) Directory access, 3) Memory access, 4) Data return

Avinash Sodani CGO PPoPP HPCA Keynote 2016

epcc

Test access

- Intel(R) Xeon Phi(TM) CPU 7210 @ 1.30GHz
  - 64 core
  - 16GB MCDRAM
  - 215W TDP
  - 1.3Ghz TDP, 1.1Ghz AVX
  - 1.6Ghz Mesh
  - 6.4GT/s OPIO
  - 96GB DDR4@2133 MT/s

GS2 on KNL

- GS2 ported and run on KNL:
  - Small test cases: sweet spots: 1,2,4,8,16,32,176,352,….
  - ARCHER ~2.10 minutes (24 cores) (7% imbalance)
  - Without fast mem: KNL (64 cores) (20% imbalance)
    - Initialization 0.41 min 13.1 %
    - Advance steps 2.65 min 86.1 %
    - total from timer is: 3.08 min
  - With fast mem: KNL (64 cores)
    - Initialization 0.30 min 17.0 %
    - Advance steps 1.43 min 81.8 %
    - total from timer is: 1.74 min
  - With cache mode: KNL
    - Initialization 0.30 min 17.0 %
    - Advance steps 1.44 min 81.8 %
    - total from timer is: 1.76 min

epcc

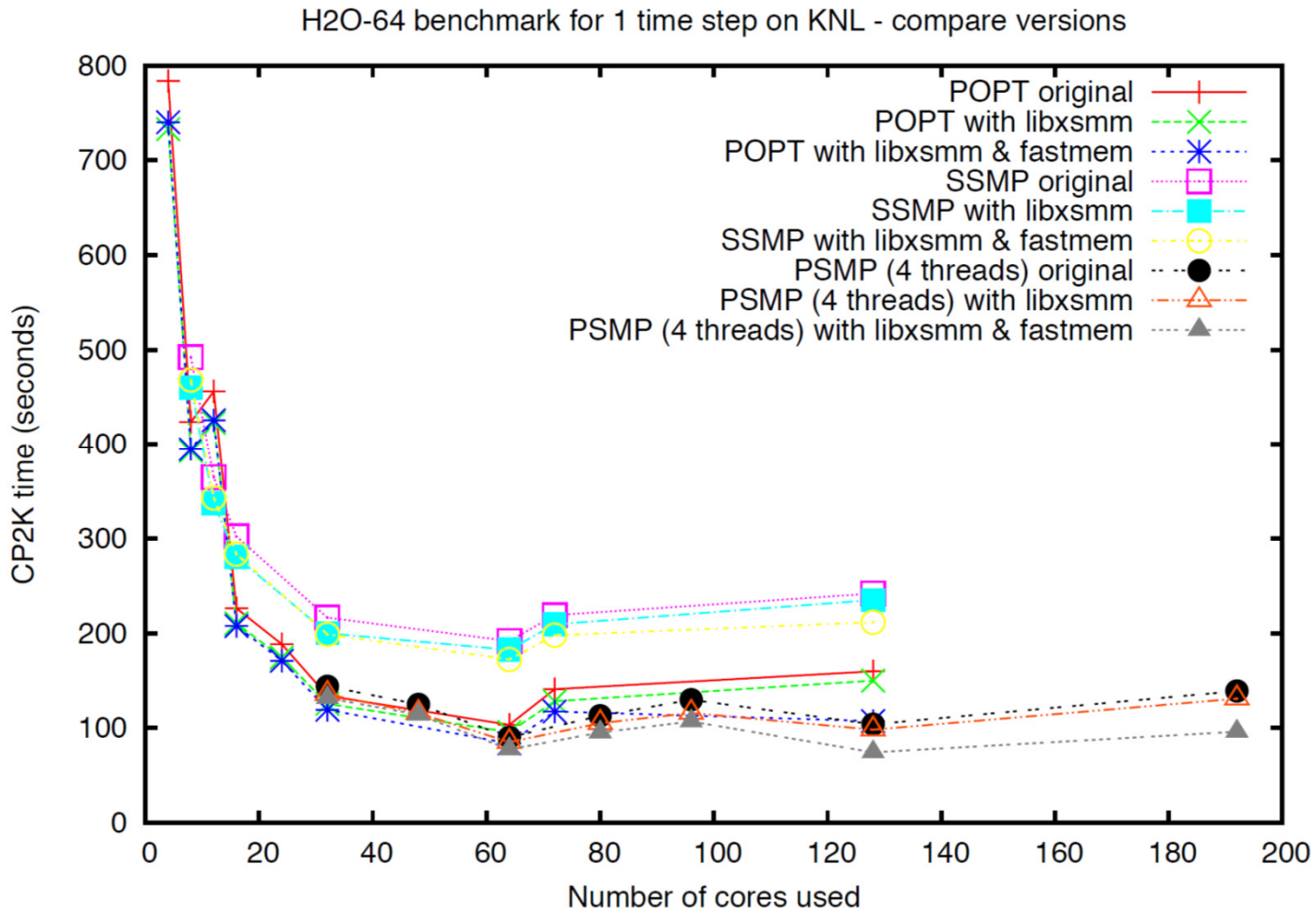
GS2 Port to KNC Xeon Phi

- Profiling of vectorisation of GS2 shows good performance
- Pure MPI code performance
  - ARCHER (2x12 core Xeon E5-2697, 16 MPI processes): 3.08 minutes
  - Host (2x8 core Xeon E5-2650, 16 MPI processes): 4.64 minutes
  - 1 Phi (176 MPI processes): 7.34 minutes
  - 1 Phi (235 MPI processes): 6.77 minutes
  - 2 Phi's (352 MPI processes): 47.71 minutes
- Hybrid code performance
  - 1 Phi (80 MPI processes, 3 threads each): 7.95 minutes
  - 1 Phi (120 MPI processes, 2 threads each): 7.07 minutes

CASTEP

- MgSiO4-Geom benchmark:
  - ARCHER: 24 cores
    - Total time = 102.27 s
  - KNL: 24 cores
    - Total time = 156.63 s
  - KNL: 64 cores
    - Total time = 149.65 s
  - KNL: 64 cores cache mode
    - Total time = 146.88 s

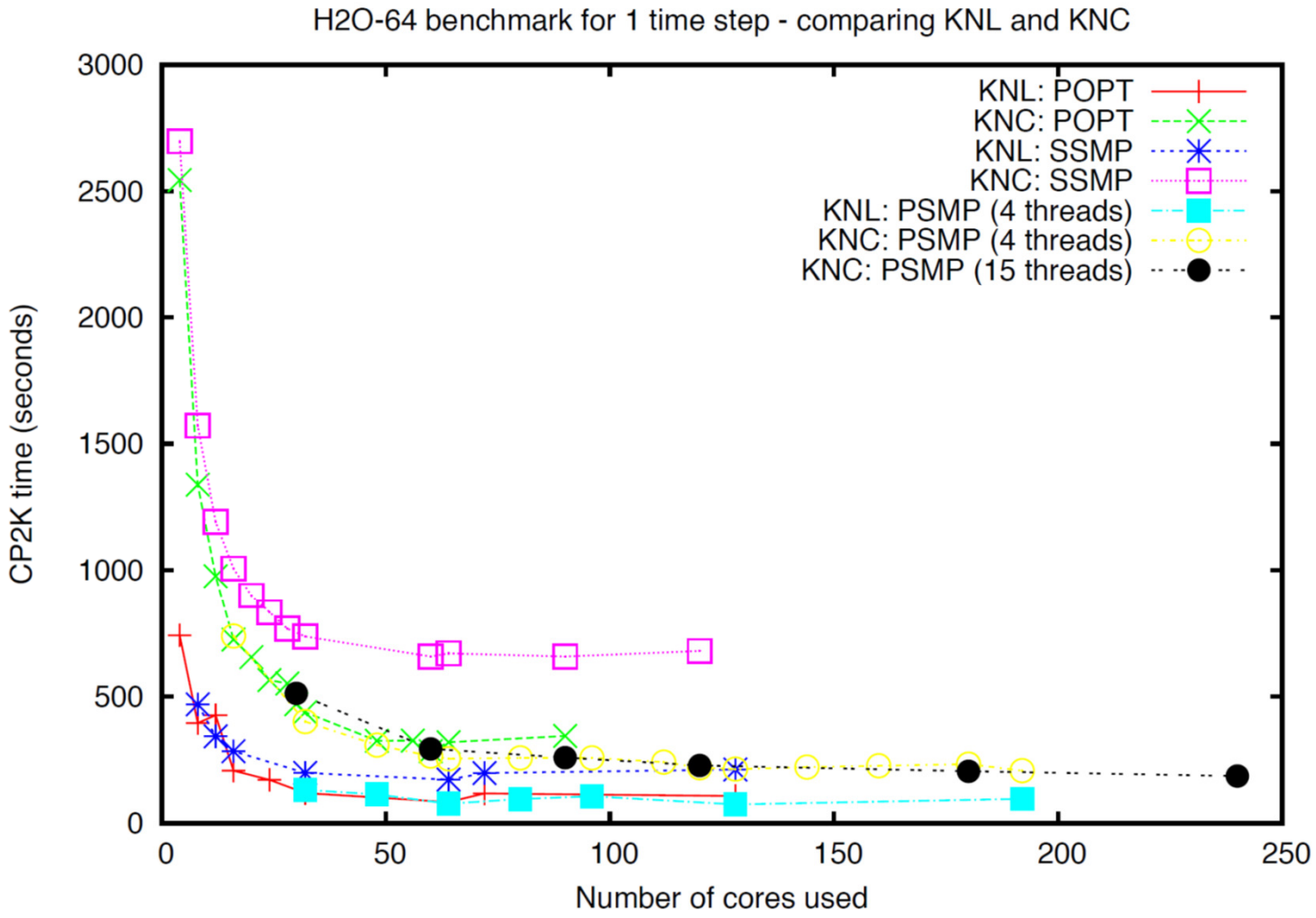
CP2K



Results courtesy of Fiona

epcc

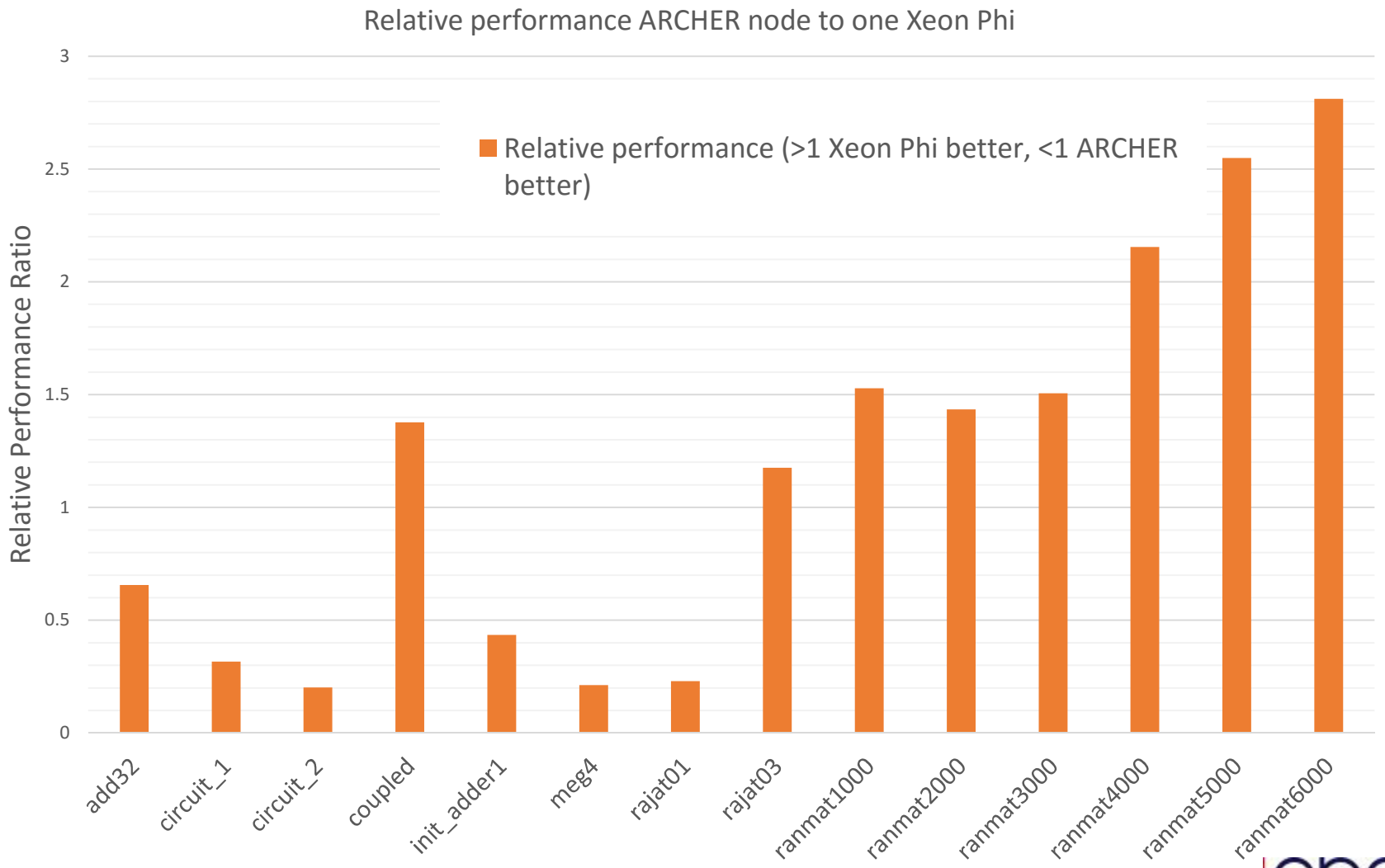
CP2K



Results courtesy of Fiona

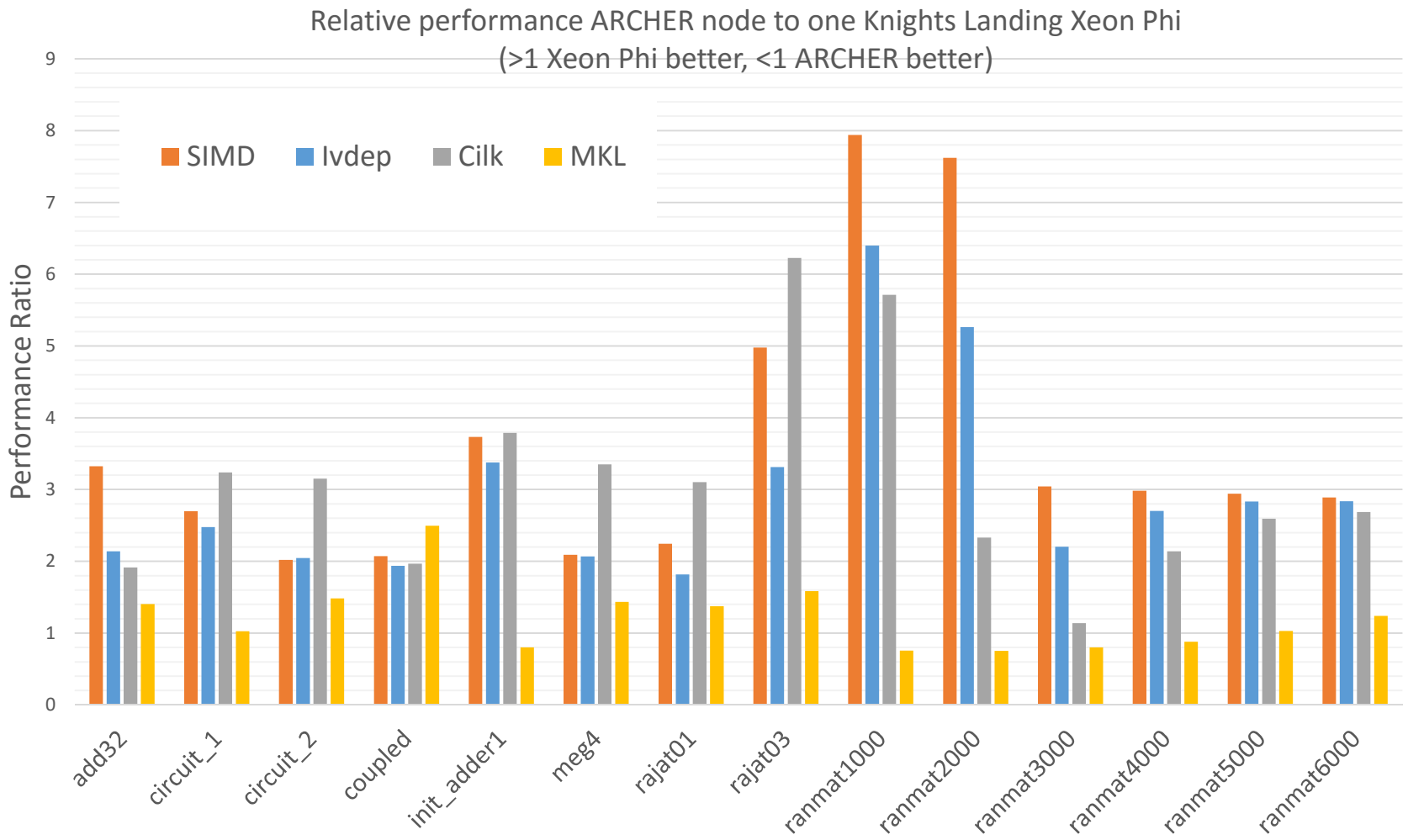
epcc

LU factorisation (KNC)



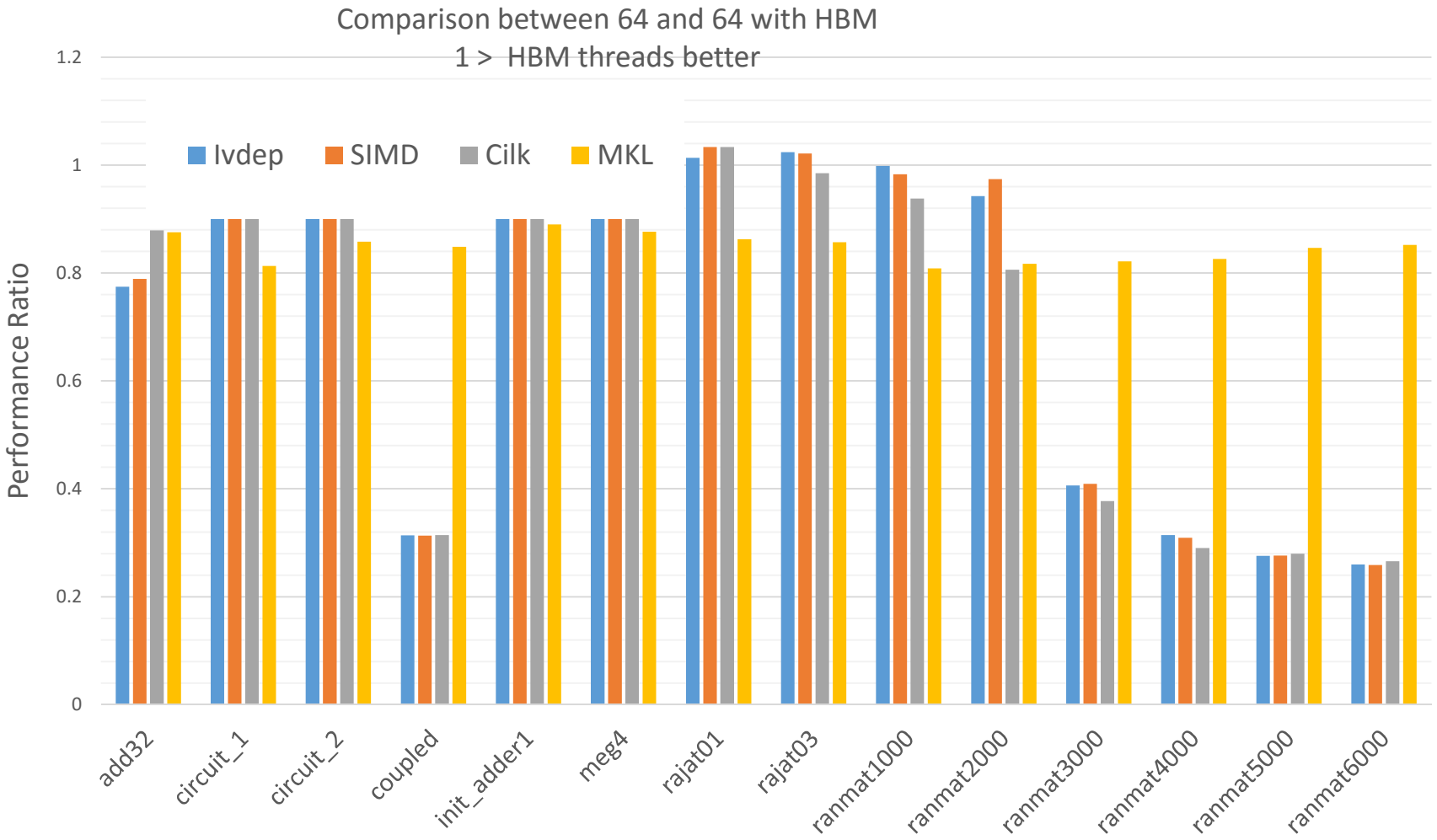
epcc

LU Factorisation



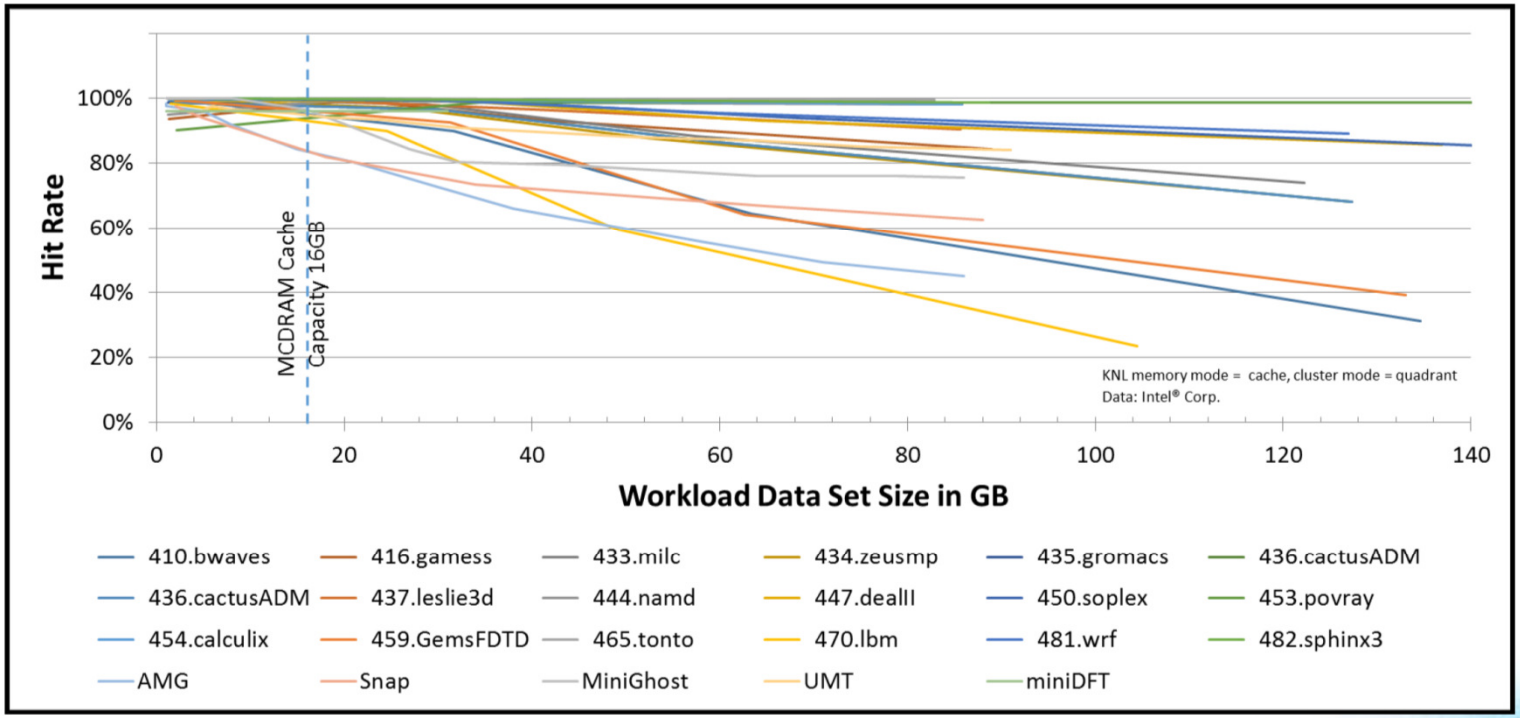
epcc

LU factorisation



KNL

MCDRAM Cache Hit Rate

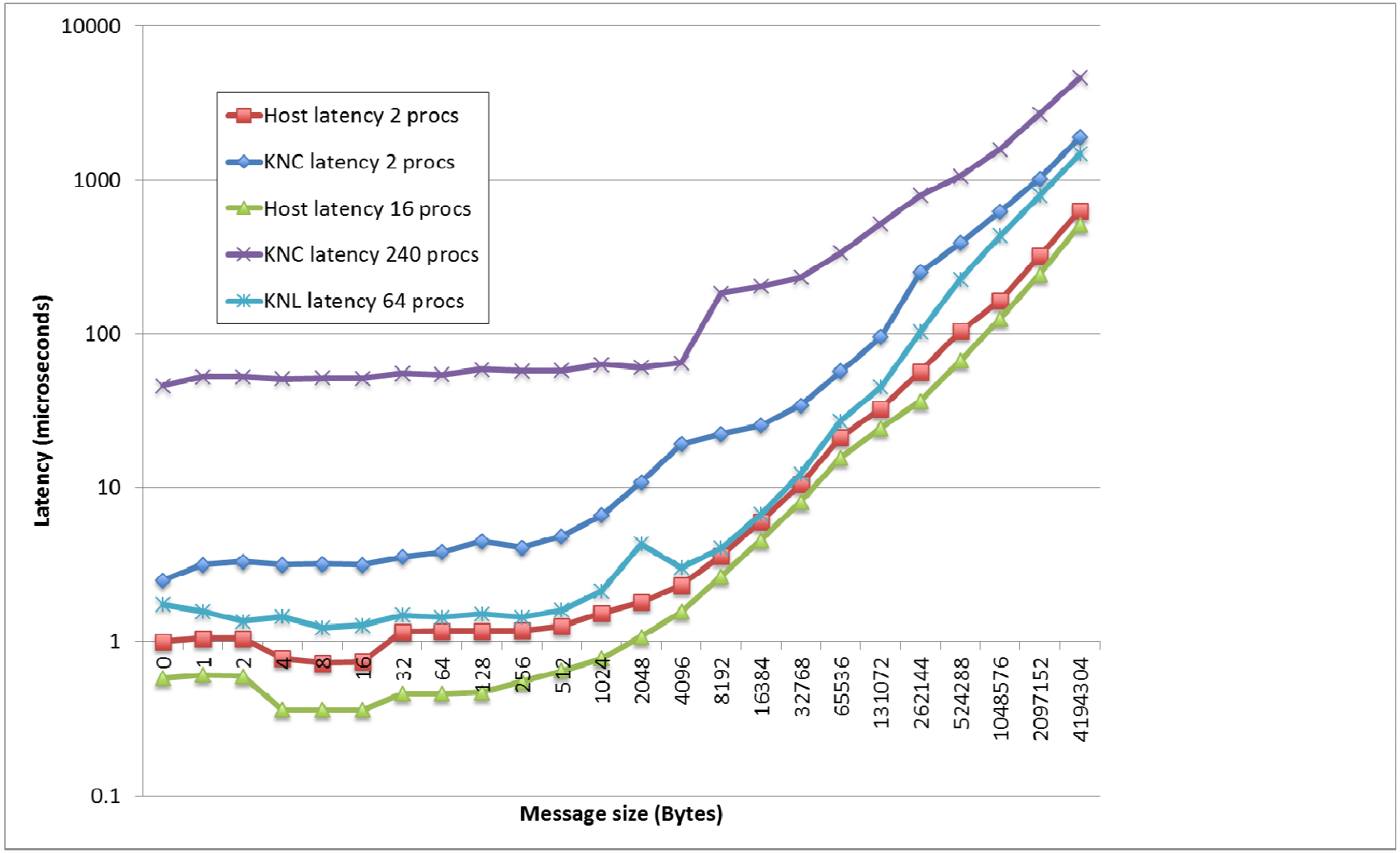


Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you purchases, including the performance of that product when combined with other products. KNL results measured on pre-production parts. Any difference in system hardware or software design or configuration may affect actual performance. For more information go to http://www.intel.com/performance *Other names and brands may be claimed as the property of others

MCDRAM performs well as cache for many workloads
Enables good out-of-box performance without memory tuning

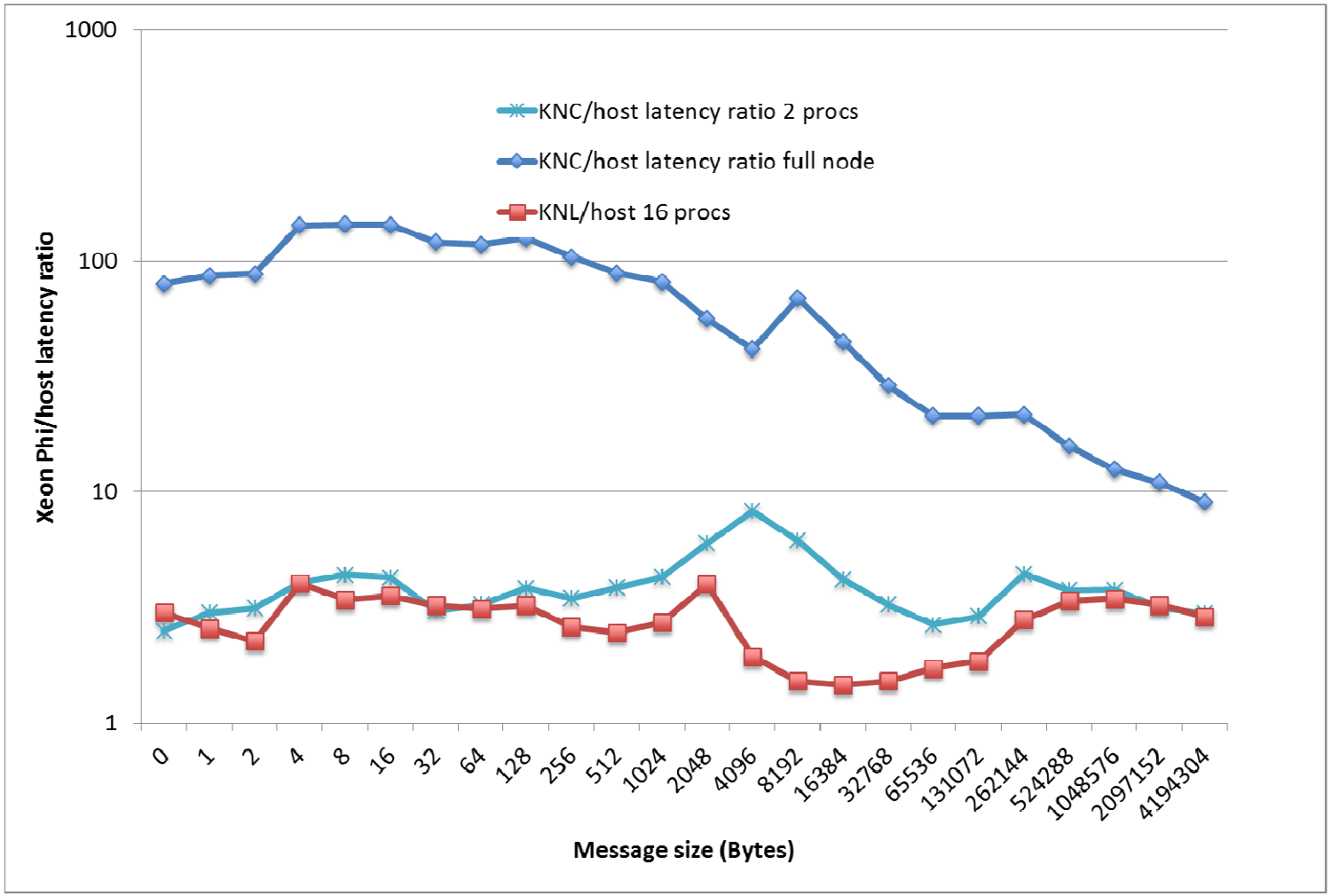
epcc

MPI Performance - PingPong



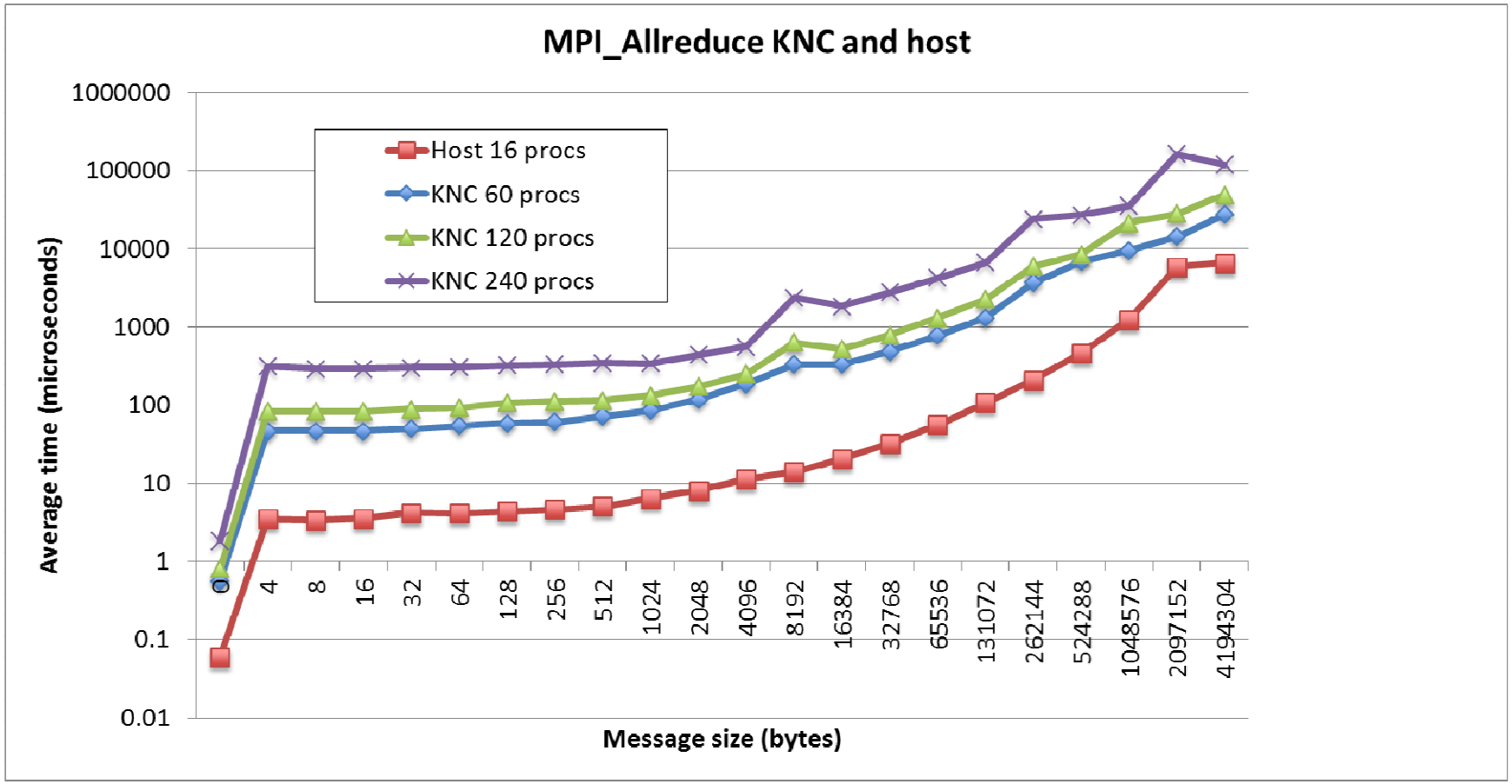
epcc

MPI Performance - PingPong



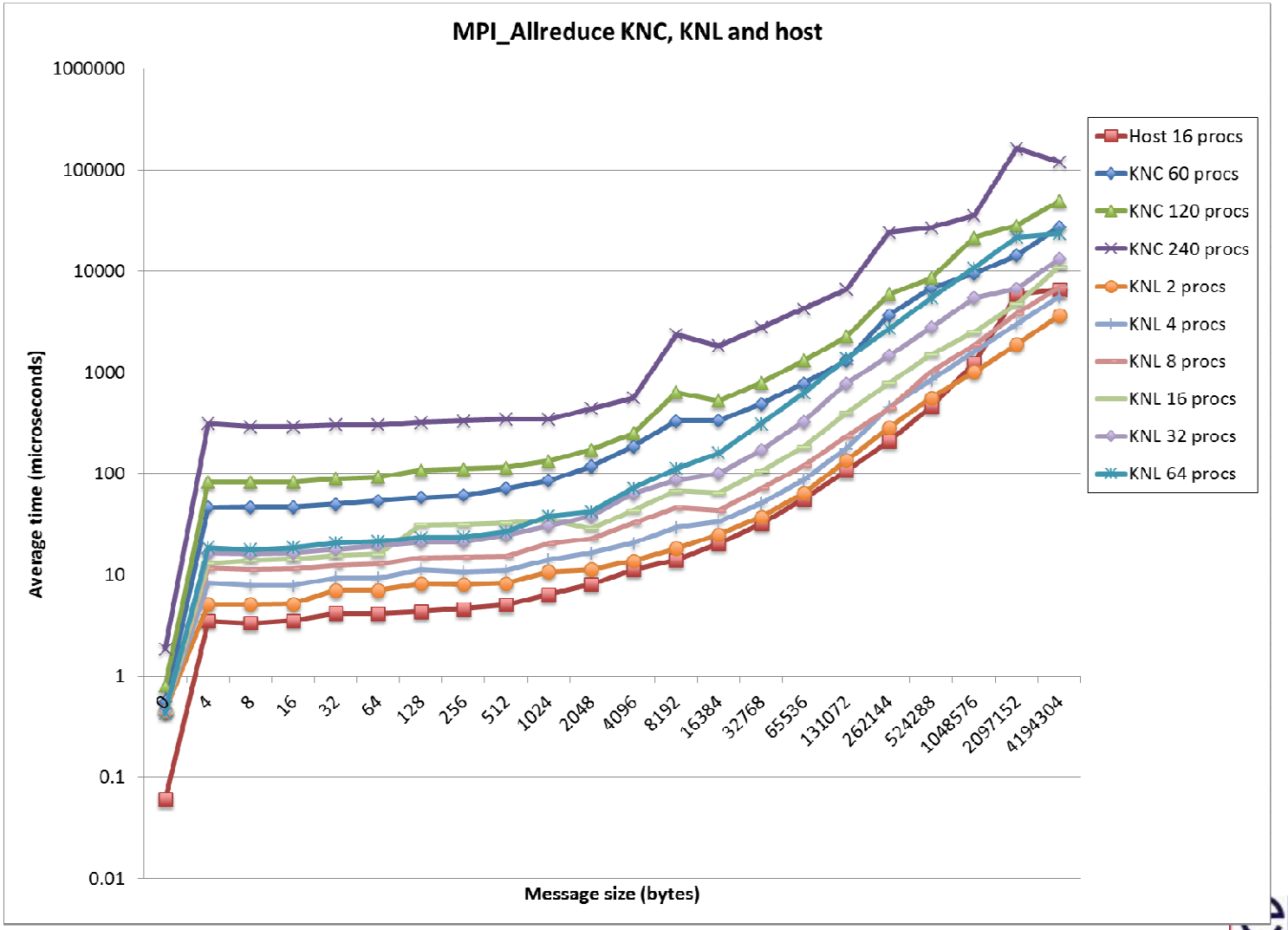
epcc

MPI Performance - Allreduce

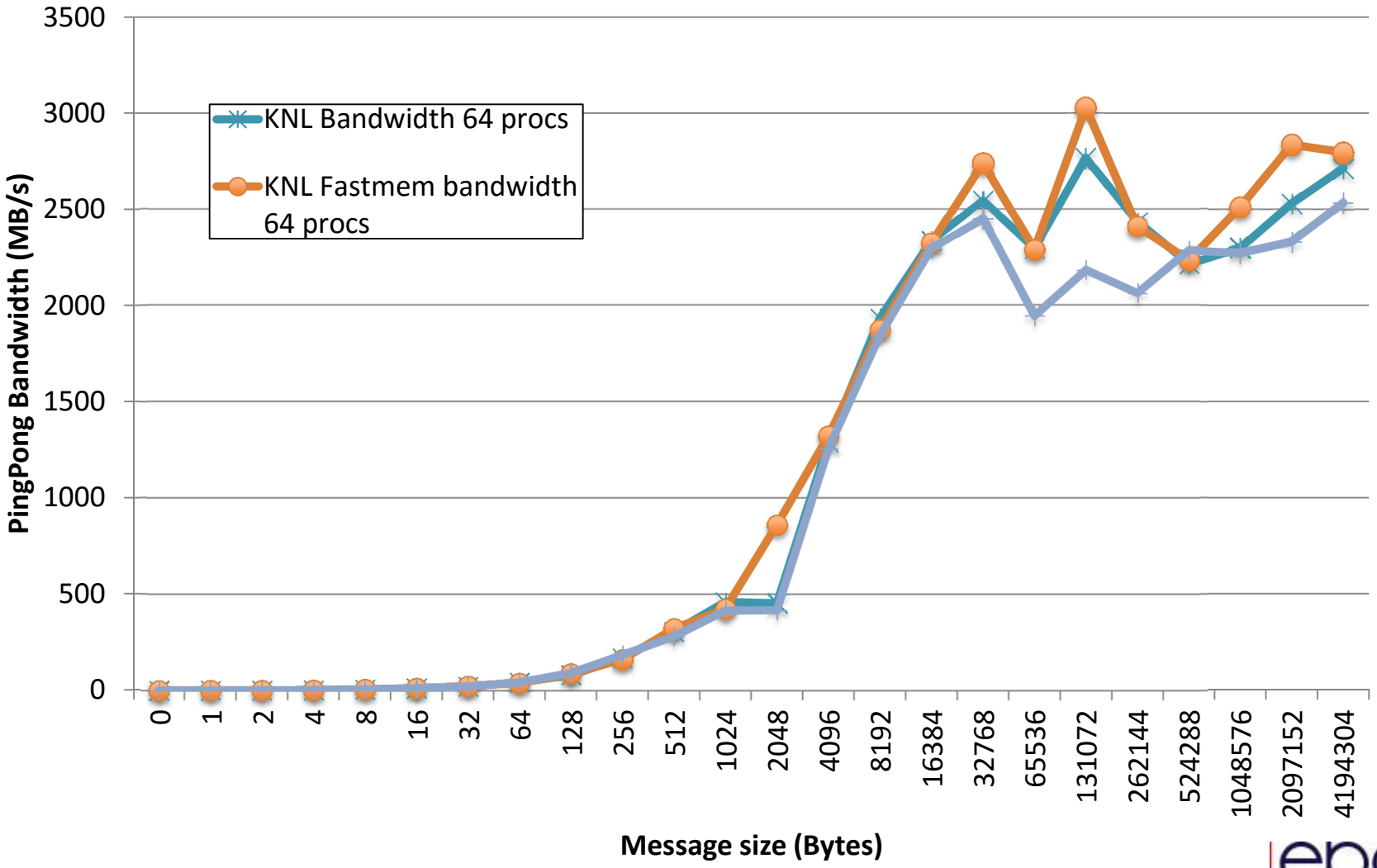


epcc

MPI Performance - Allreduce

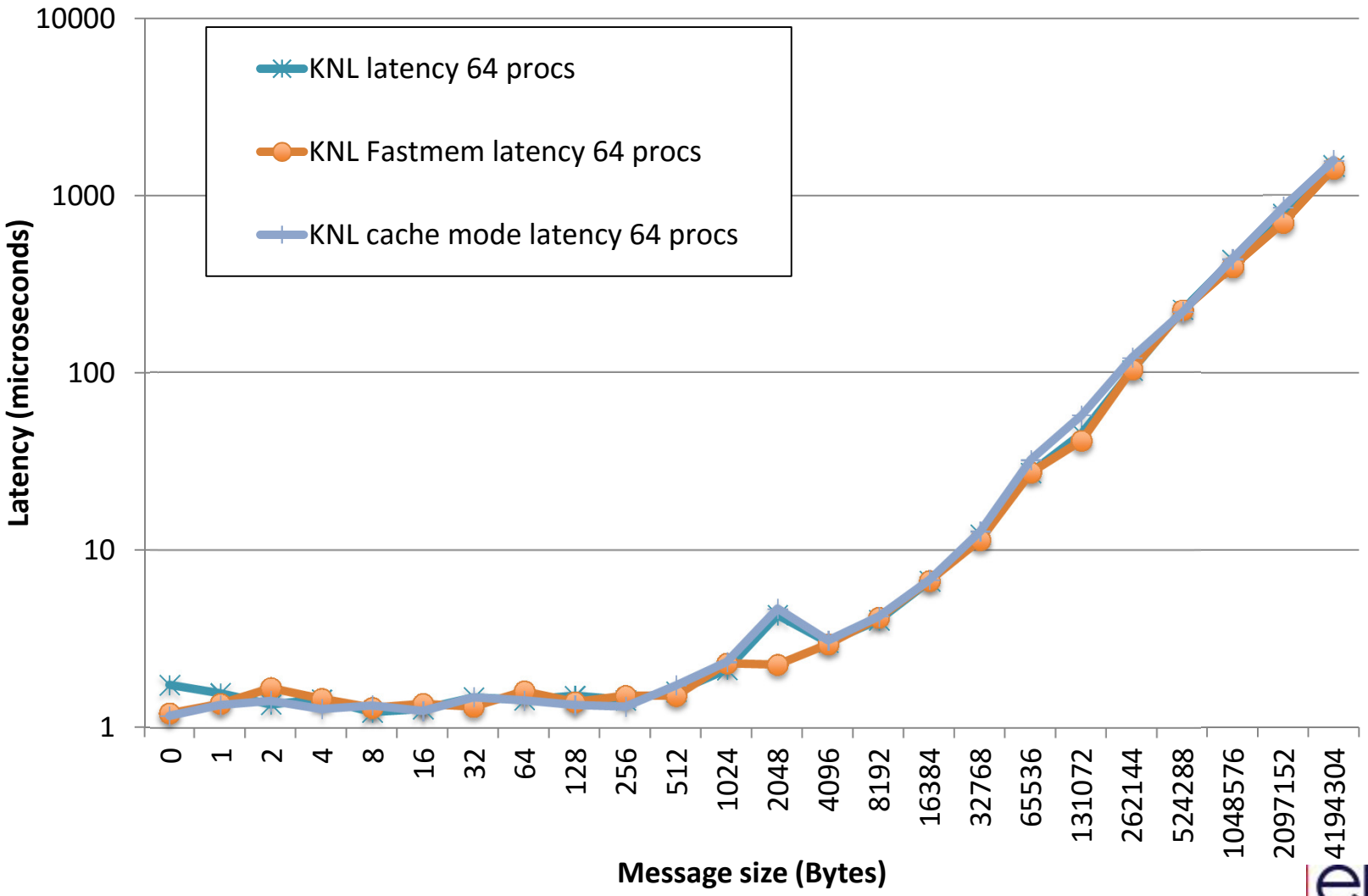


MPI Performance – PingPong – Memory modes



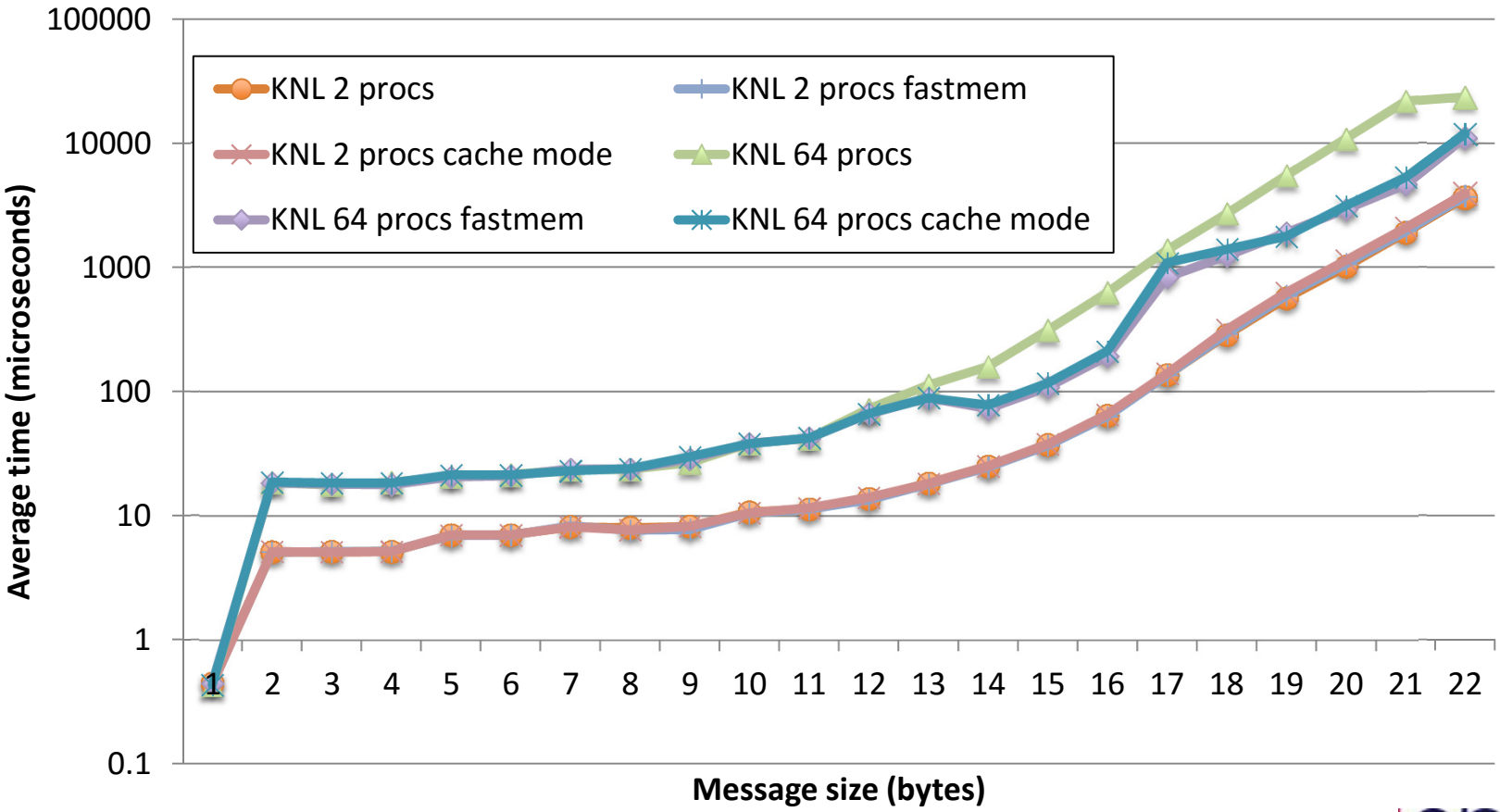
epcc

MPI Performance – PingPong – Memory modes



epcc

MPI_Allreduce KNL different memory modes for 2 and 64 processor benchmarks



epcc