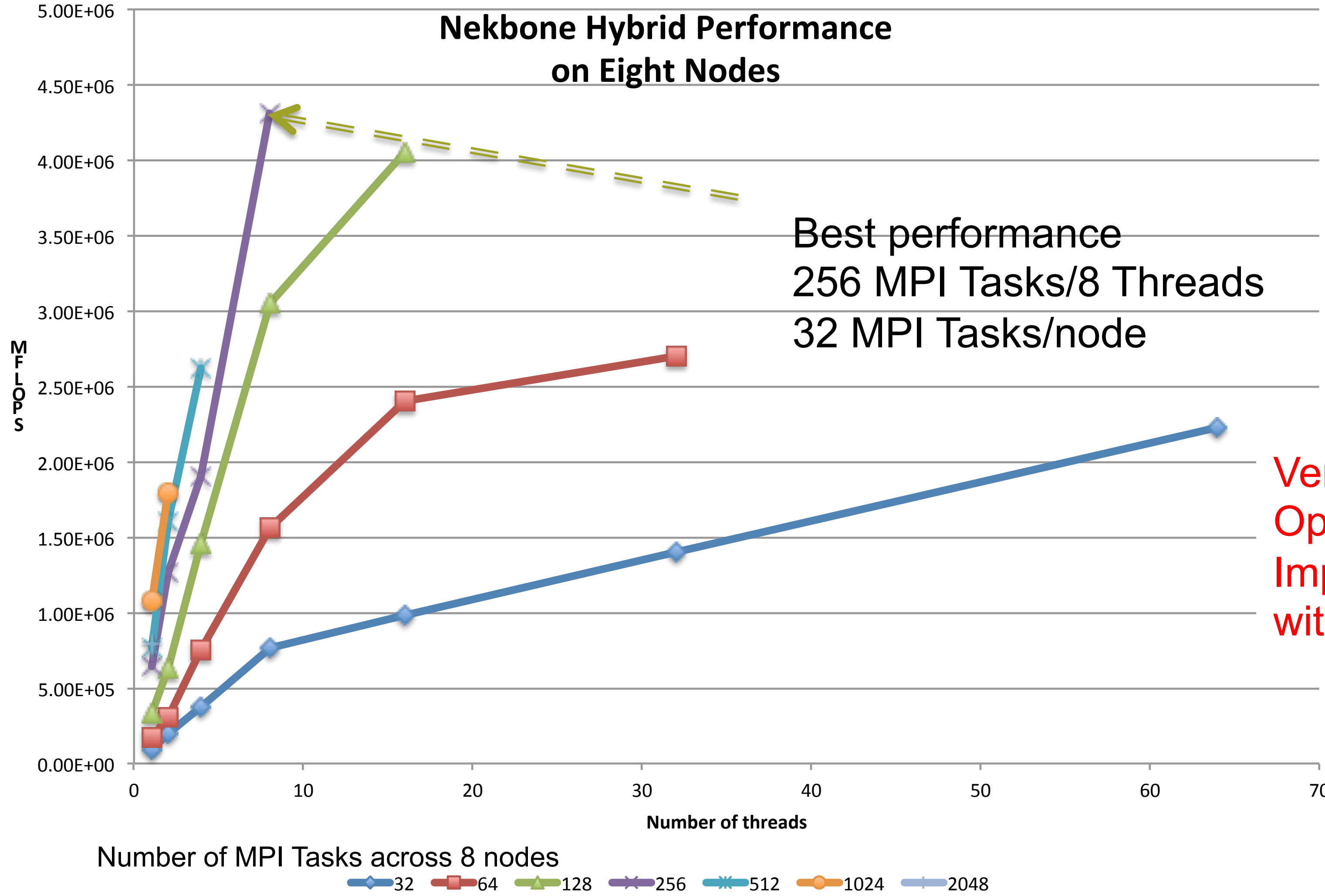
Myths about KNL

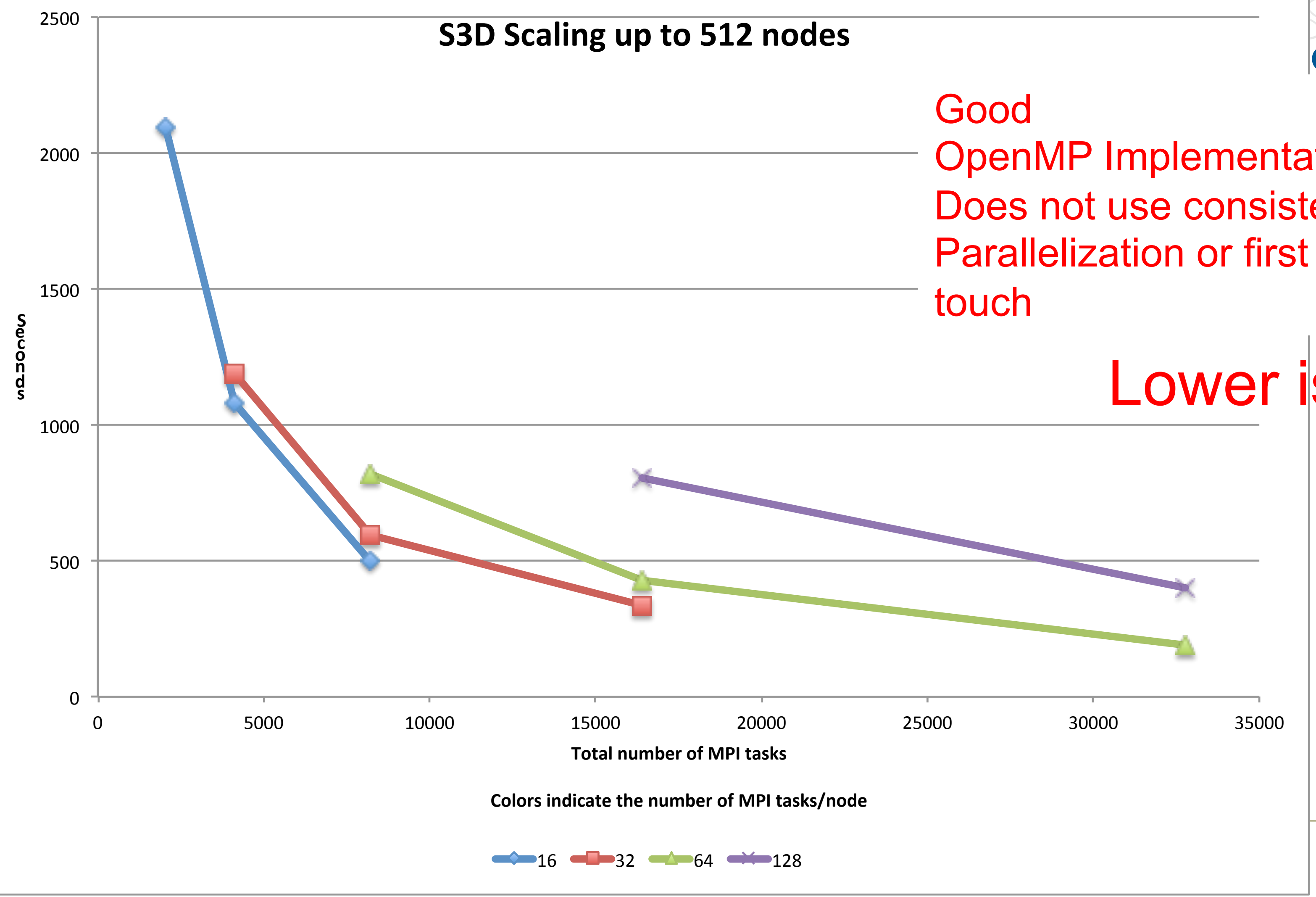
Have to use lots of threading on node



Best performance
256 MPI Tasks/8 Threads
32 MPI Tasks/node

Product of MPI and OpenMP Threads = 64 on node

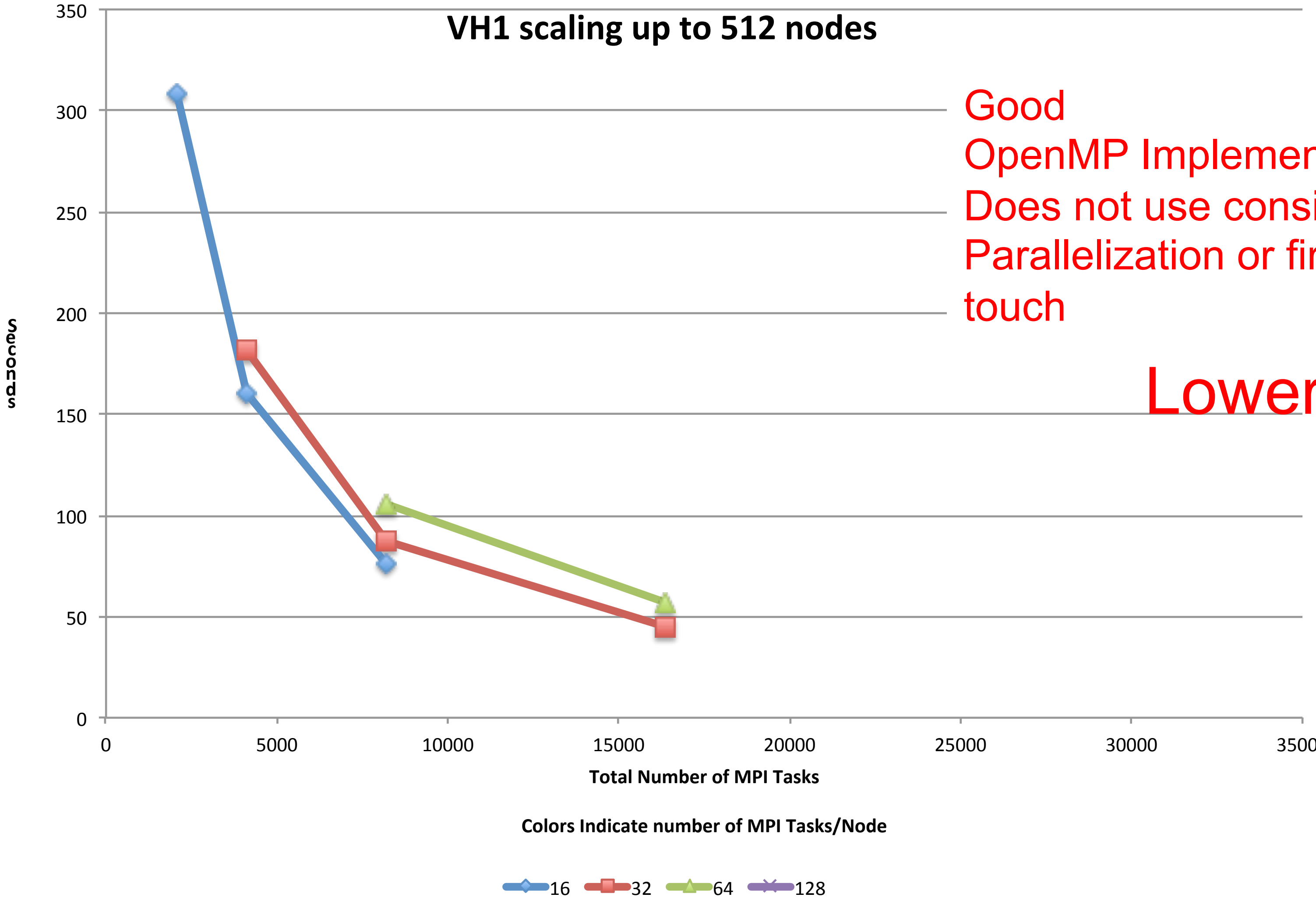
Very Good OpenMP Implementation with first touch



Good
OpenMP Implementation
Does not use consistent
Parallelization or first
touch

Lower is Better

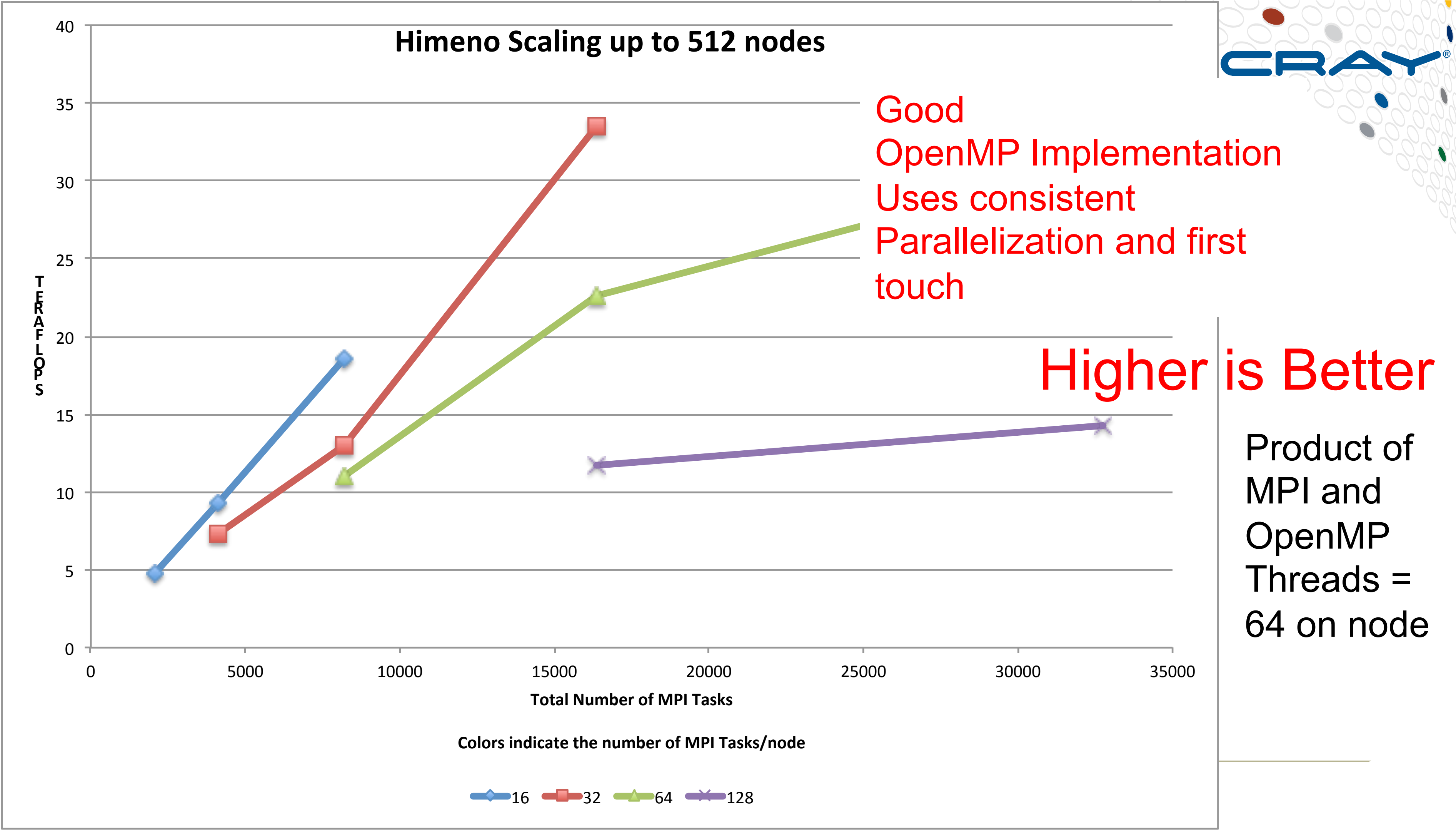
Product of
MPI and
OpenMP
Threads =
64 on node



Good
OpenMP Implementation
Does not use consistent
Parallelization or first
touch

Lower is Better

Product of
MPI and
OpenMP
Threads =
64 on node



Product of MPI and OpenMP Threads = 64 on node

Lots of usable Options for MCDRAM

Only two reasonable uses of MCDRAM

- **If code and executable fits within 16 Gbytes use numctl=1**
  - Numctl perferred is not useful
  - Why do you have all that DDR memory???
- **If code and executable is larger than 16 Gbytes use Cache**
  - Watch out for Direct Mapped Cache
  - Have not found any example where using MCDRAM as separate memory either all of partial beat using MCDRAM as CACHE

Important Data for Haswell and KNL

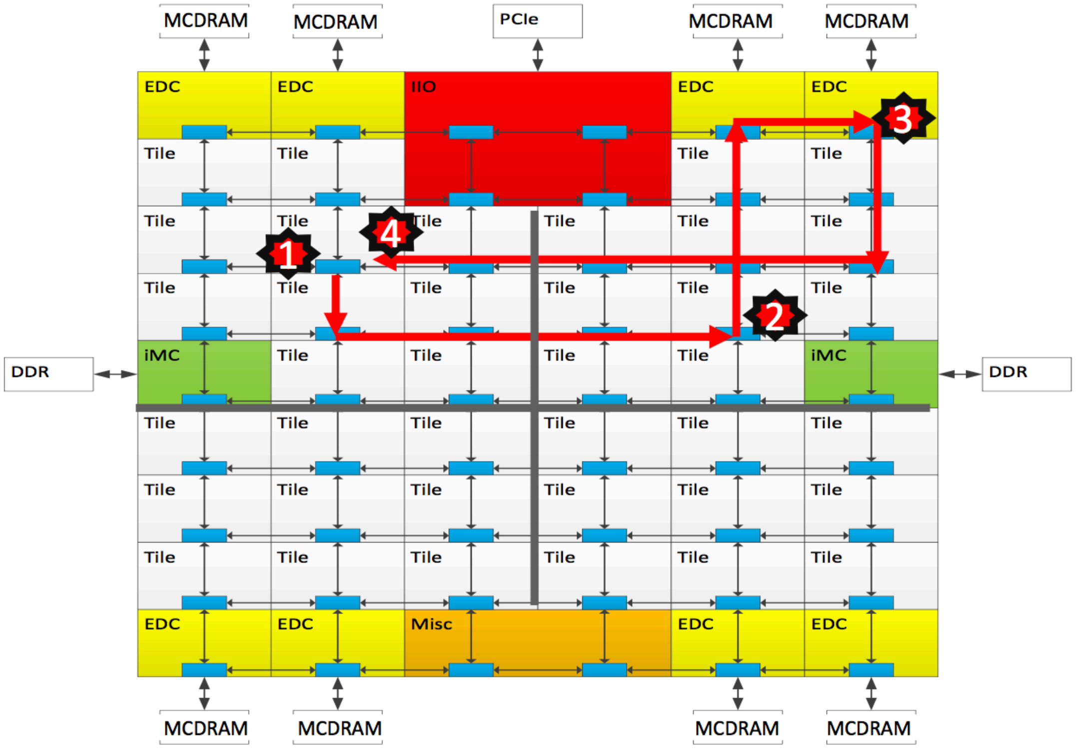
	Latency Haswell Nanosec (clocks)	KNL Size /core	Latency KNL Nanosec(clocks)	KNL Size /core	Bandwidth Haswell GB/sec Stream Triad	Bandwidth KNL GB/sec Stream Triad
L1 Cache	2.7 (6)	32KB	2.9 (4)	32KB		
L2 Cache	8.11 (19)	256KB	13.6 (19)	512KB		
MCDRAM (Cache)	24.35 (56) (Level 3)	2.5MB	173.5 (243)	242MB		329
MCDRAM (Flat)			174.2 (244)	242MB		486
DDR (Flat)			151.3 (212)		102	90
DDR (Cache)						59

What to do with all those Clustering modes

Cluster – Quad

- **Allows a collection of cores within a quadrant to get all of the memory bandwidth**
  - Could be important for a bandwidth sensitive, load imbalanced mostly-MPI application.
    - Mostly MPI = greater than/or equal to 16 MPI tasks per node
- **Introduces additional NUMA affects when threading across more than one tile**

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

Conclusions

- **KNL is very easy to port to; however, performance can always be improved**
- **For KNL to beat Xeon you must**
  - VECTORIZE
  - Use as many cores as possible – At least 16 MPI tasks – rest threads
    - OpenMP must be good OpenMP
  - Heavily use MCDRAM
- **Have seen applications that do all three and can be blocked for MCDRAM-Cache run 3-4 times faster than Xeon**
  - These usually cannot be blocked effectively for L3 cache on Xeon