Using NERSC for Research in High Energy Physics Theory

Richard Gerber

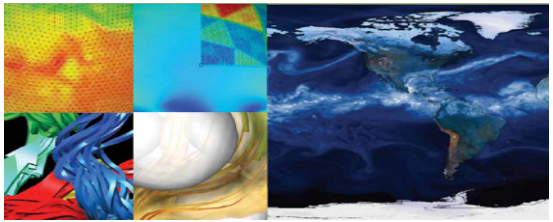
Senior Science Advisor

HPC Department Head (Acting)

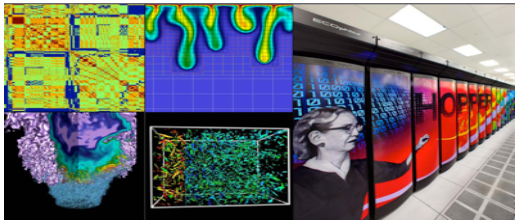
NERSC: the Mission HPC Facility for DOE Office of Science Research

U.S. DEPARTMENT OF ENERGY | Office of Science

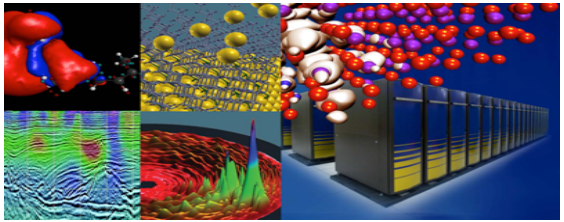
Largest funder of physical science research in U.S.



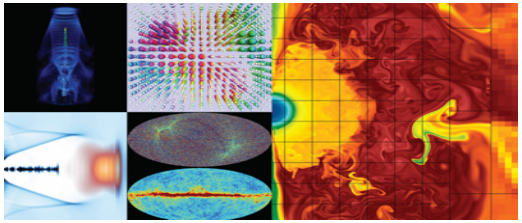
Bio Energy, Environment



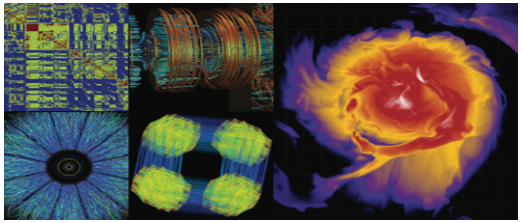
Computing



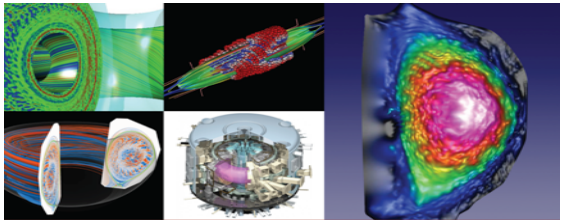
Materials, Chemistry, Geophysics



Particle Physics, Astrophysics



Nuclear Physics



Fusion Energy, Plasma Physics

6,000 users, 700 projects, 700 codes, 48 states, 40 countries, universities & national labs

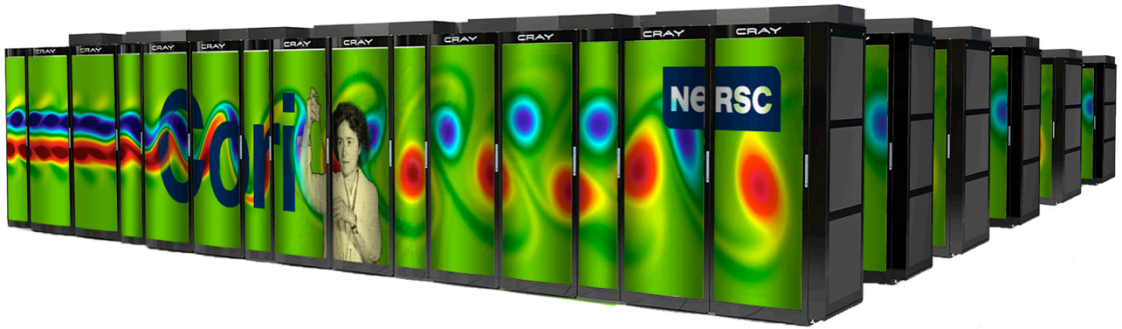
Current Production Systems

Edison

5,560 Ivy Bridge Nodes / 24 cores/node
133 K cores, 64 GB memory/node
Cray XC30 / Aries Dragonfly interconnect
6 PB Lustre Cray Sonexion scratch FS

Cori Haswell Nodes

1,900 Haswell Nodes / 32 cores/node
52 K cores, 128 GB memory/node
Cray XC40 / Aries Dragonfly interconnect
24 PB Lustre Cray Sonexion scratch FS
1.5 PB Burst Buffer



Cori Xeon Phi KNL Nodes

Cray XC40 system with 9,300 Intel Knights Landing compute nodes

68 cores / 96 GB DRAM / 16 GB HBM

Support the entire Office of Science research community

Begin to transition workload to energy efficient architectures

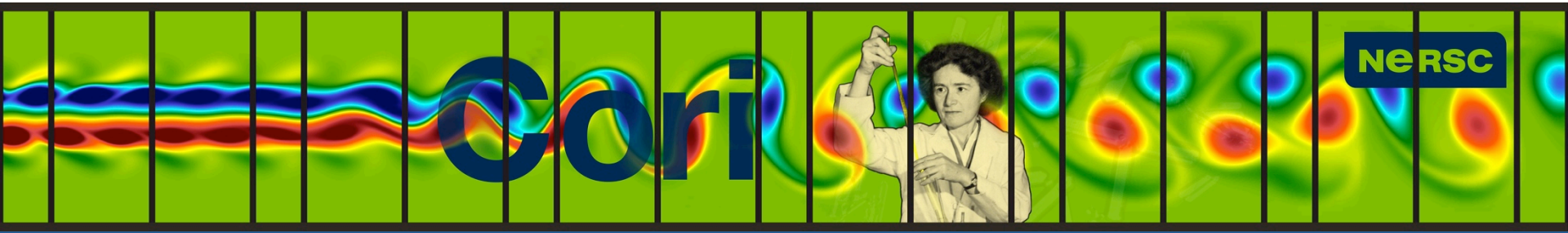
Data Intensive Science Support

10 Haswell processor cabinets (Phase 1)

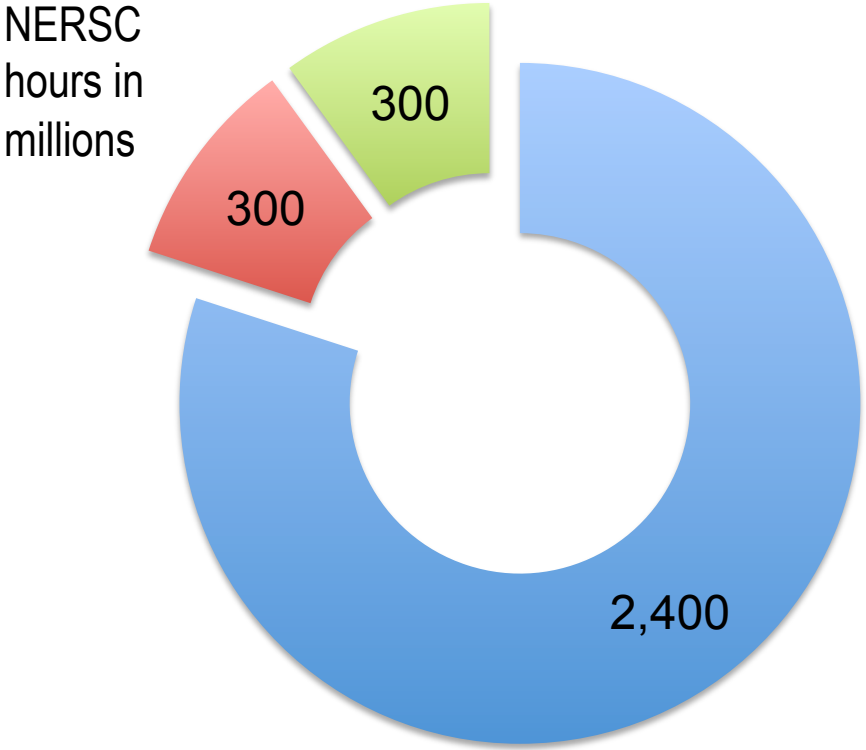
NVRAM Burst Buffer 1.5 PB, 1.5 TB/sec

30 PB of disk, >700 GB/sec I/O bandwidth

Integrated with Cori Ha swell nodeson Aries network for data / simulation / analysis on one system

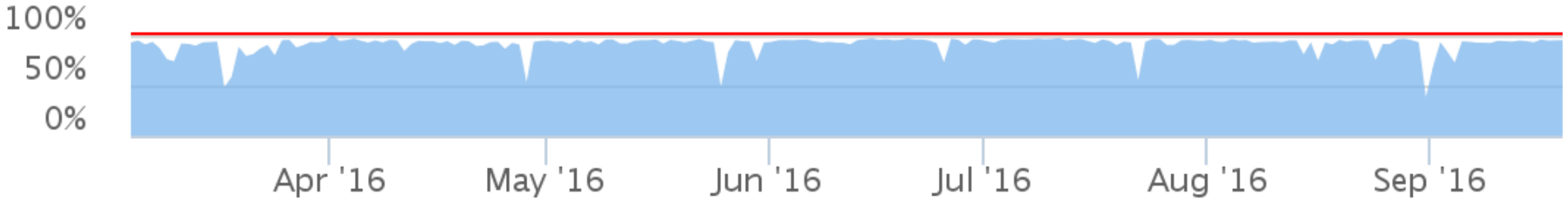


NERSC Allocation of Computing Time



- DOE Mission Science **80%**
  - Distributed by DOE SC program managers
- ALCC **10%**
  - Competitive awards run by DOE ASCR
- Directors Discretionary **10%**
  - Strategic awards from NERSC

NERSC has ~100% utilization

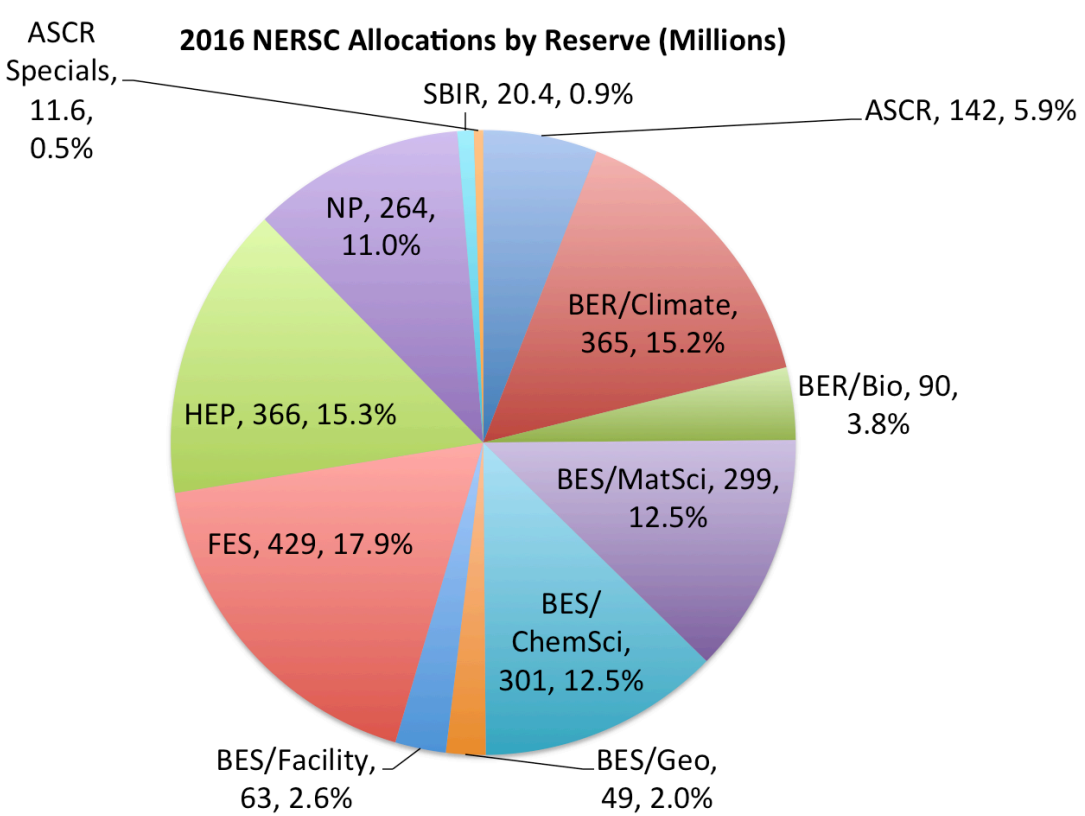


Important to get support and allocation from DOE program manager (L. Chatterjee) or through ALCC!

They are supportive.

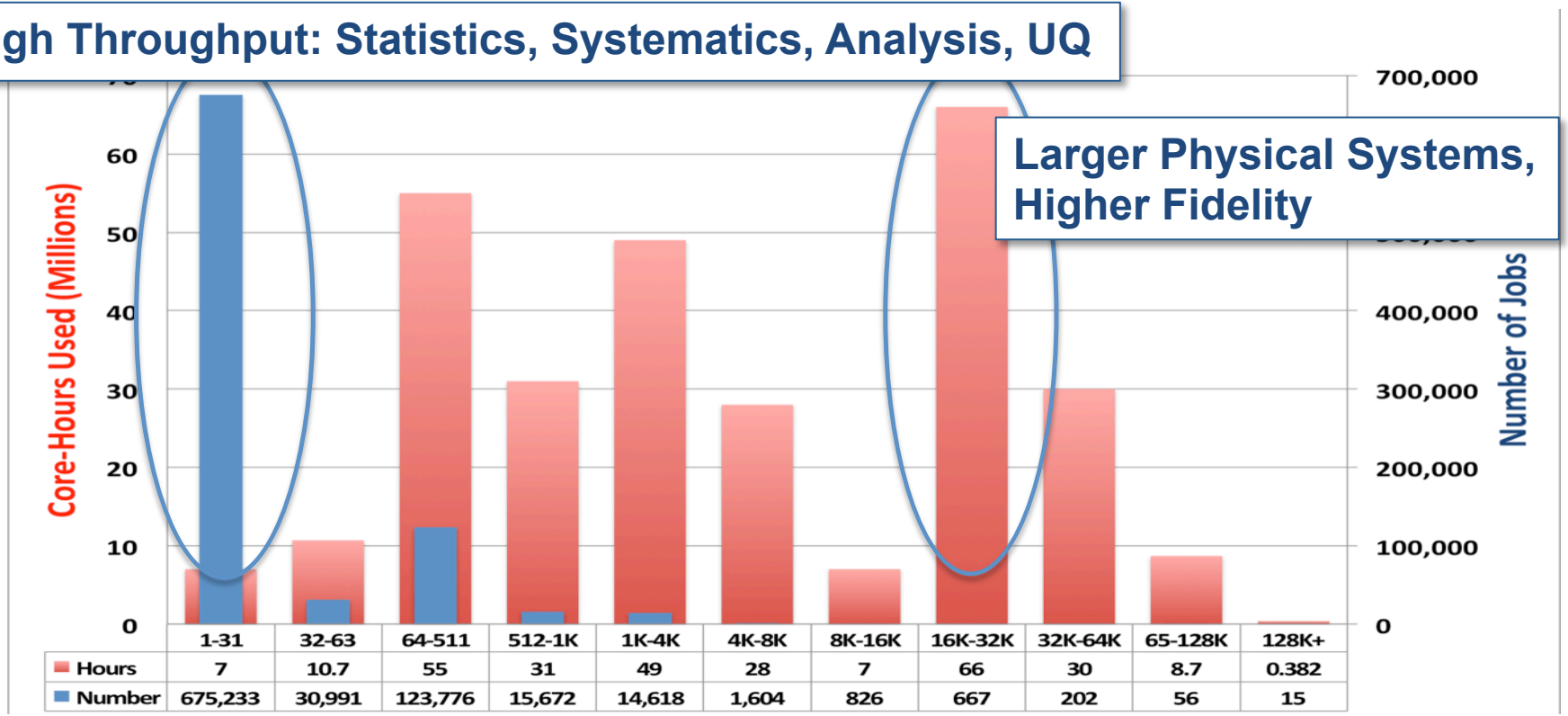
PI	Allocation (Hrs)	Program
Childers/Lecompte	18,600,000	ALCC
Hoeche	9,000,000	DOE Production
Hinchliffe	800,000	DOE Production
Ligeti	2,800,000	DOE Production
Piperov	1,500,000	DOE Production

Initial Allocation Distribution Among Offices for 2016



NERSC Supports Jobs of all Kinds and Sizes

High Throughput: Statistics, Systematics, Analysis, UQ



	1-31	32-63	64-511	512-1K	1K-4K	4K-8K	8K-16K	16K-32K	32K-64K	65-128K	128K+
Hours	7	10.7	55	31	49	28	7	66	30	8.7	0.382
Number	675,233	30,991	123,776	15,672	14,618	1,604	826	667	202	56	15

Cori Integration Status

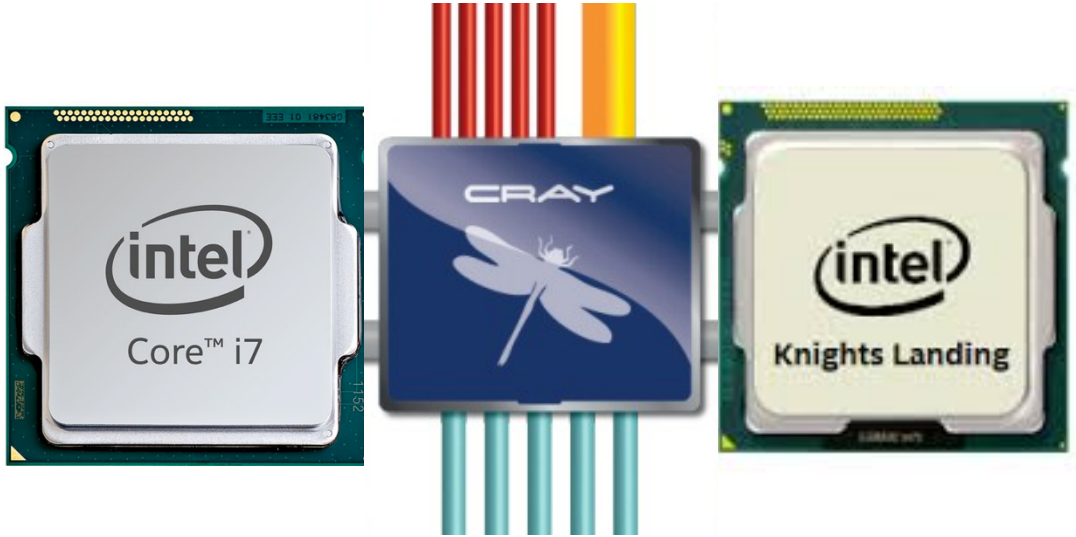
July-August
- 9300 KNL nodes arrive, installed, tested

Monday
- P1 shut down, P2 stress test

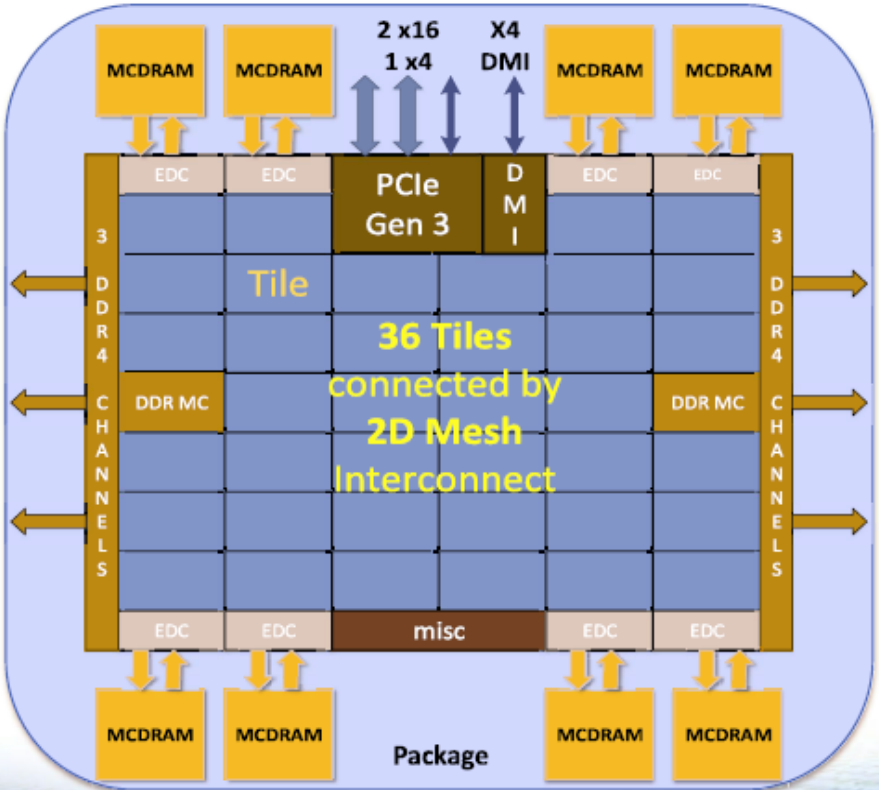
This week
- Move I/O, network blades
- Add Haswell to P1 to fill holes
- Cabling/Re-cabling
- Aries/LNET config
- Cabinet reconfigs

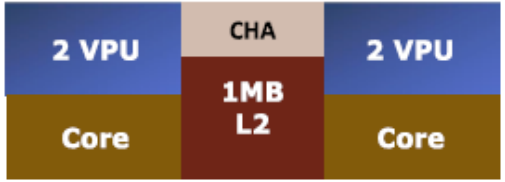
Now to now+6 weeks
- …continue, test, resolve issues
- configure SLURM

NESAP code team access ASAP!



Knights Landing Overview





Chip: 36 Tiles interconnected by **2D Mesh**
Tile: 2 Cores + 2 VPU/core + 1 MB L2

Memory: MCDRAM: 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 only
Fabric: Omni-Path on-package (not shown)

Vector Peak Perf: 3+TF DP and 6+TF SP Flops
Scalar Perf: ~3x over Knights Corner
Streams Triad (GB/s): MCDRAM : 400+; DDR: 90+

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. †Binary Compatible with Intel Xeon processors using Haswell Instruction Set (except TSX). ††Bandwidth 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.

Key Intel Xeon Phi (KNL) Features

NERSC

Single socket self-hosted processor

- (Relative!) ease of programming using portable programming models and languages (MPI+OpenMP)
- Evolutionary coding model on the path to manycore exascale systems

Low-power manycore (68) processor with up to 4 hardware threads

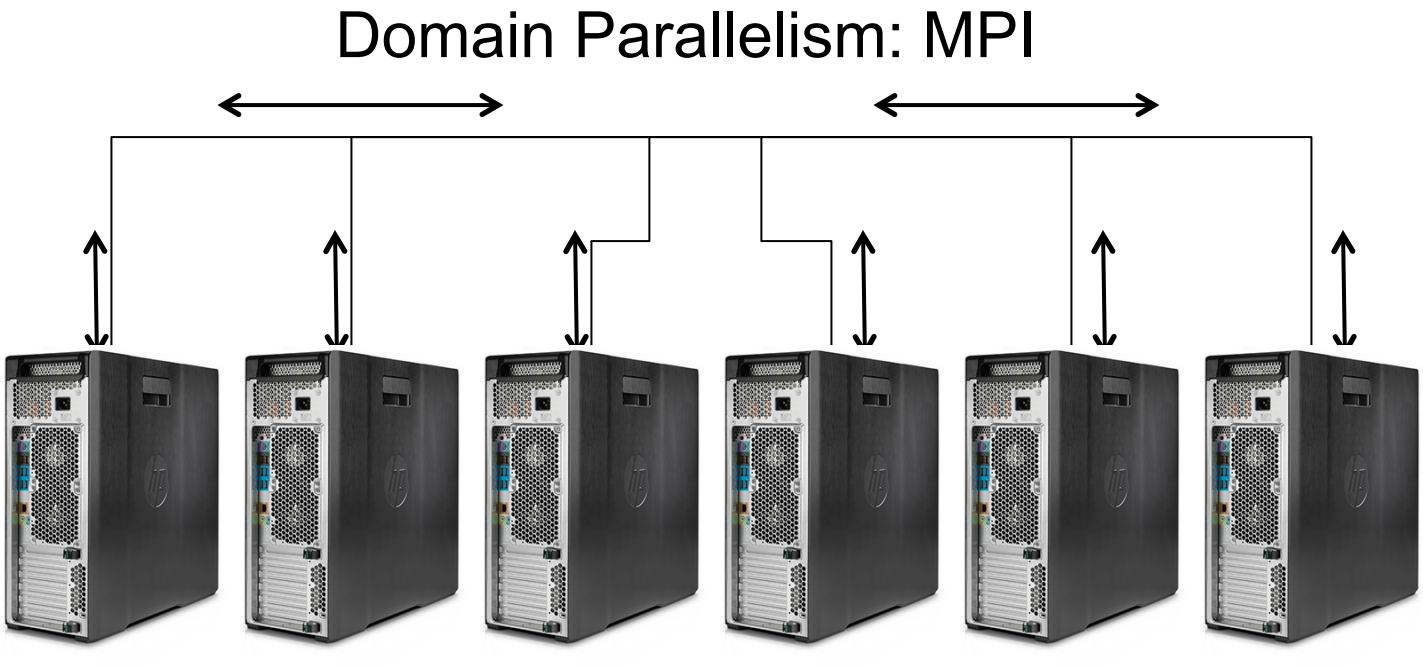
512b vector units

- Opportunity for 32 DP flops / clock (2 VPU * 64b * FMA)

16 GB High bandwidth on-package memory

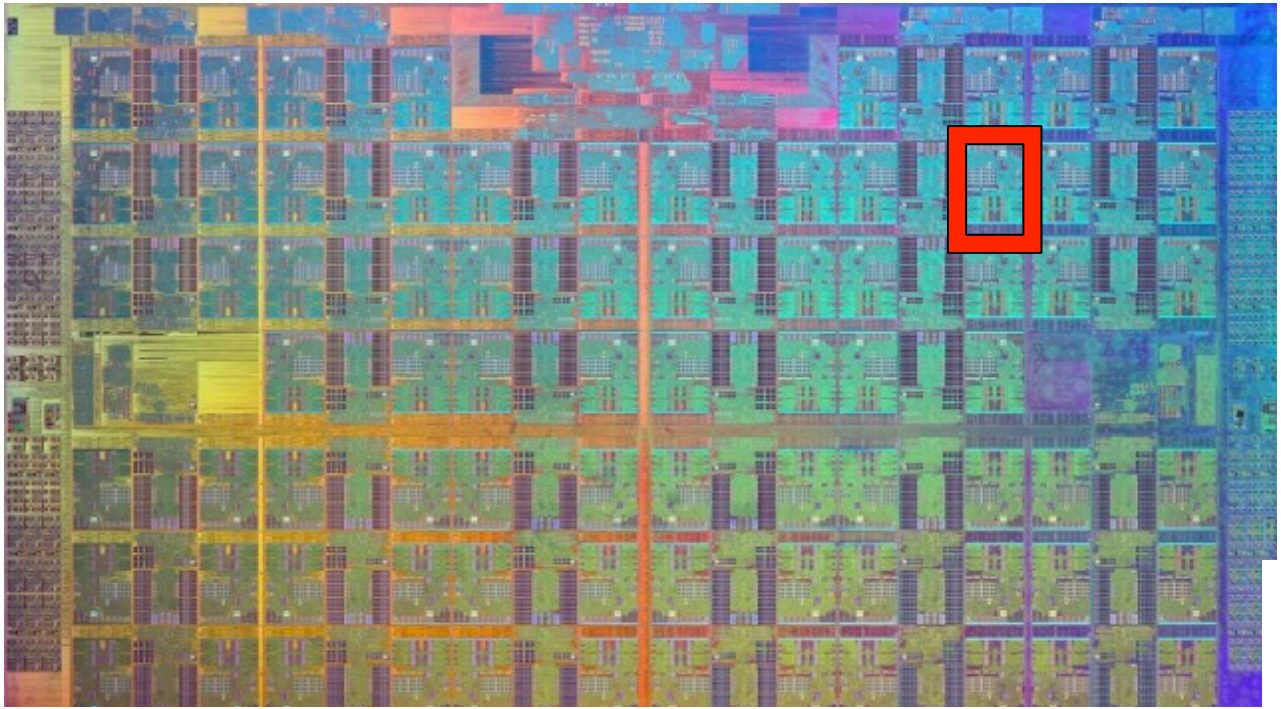
- Bandwidth 4-5X that of DDR4 DRAM memory
- Many scientific applications are memory-bandwidth bound

Top Level Parallelism



Opportunity cost: 9300X

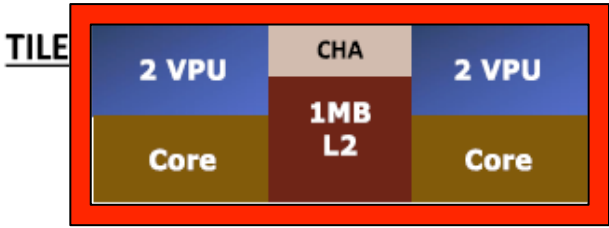
Thread-Level Parallelism for Xeon Phi Manycore



Xeon Phi "Knights Landing"

68 Cores with 1-4 threads

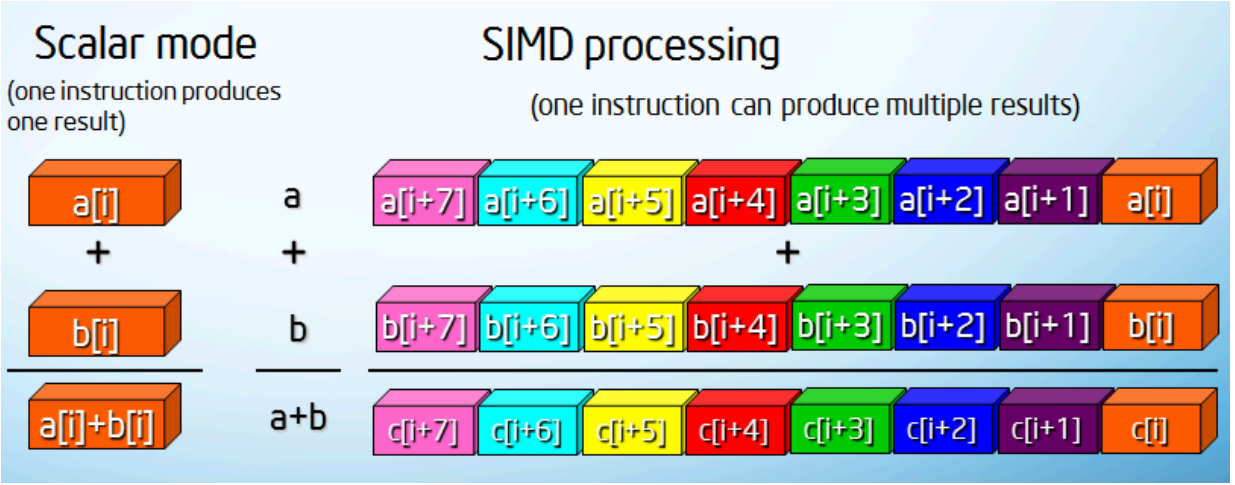
Commonly using OpenMP to express threaded parallelism



On-Chip Parallelism – Vectorization (SIMD)

Single instruction to execute up to 16 DP floating point operations per cycle per VPU.

32 Flop / cycle / core
44 Gflops / core
3 TFlops / node

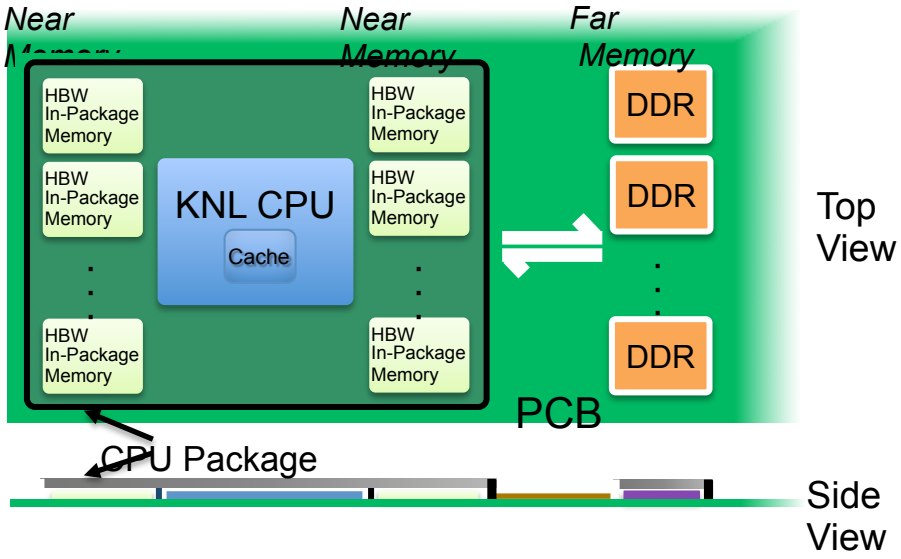


Knights Landing Integrated On-Package Memory

Cache Model — Let the hardware automatically manage the integrated on-package memory as an "L3" cache between KNL CPU and external DDR

Flat Model — Manually manage how your application uses the integrated on-package memory and external DDR for peak performance

Hybrid Model — Harness the benefits of both cache and flat models by segmenting the integrated on-package memory



Maximum performance through higher memory bandwidth and flexibility

Data layout crucial for performance

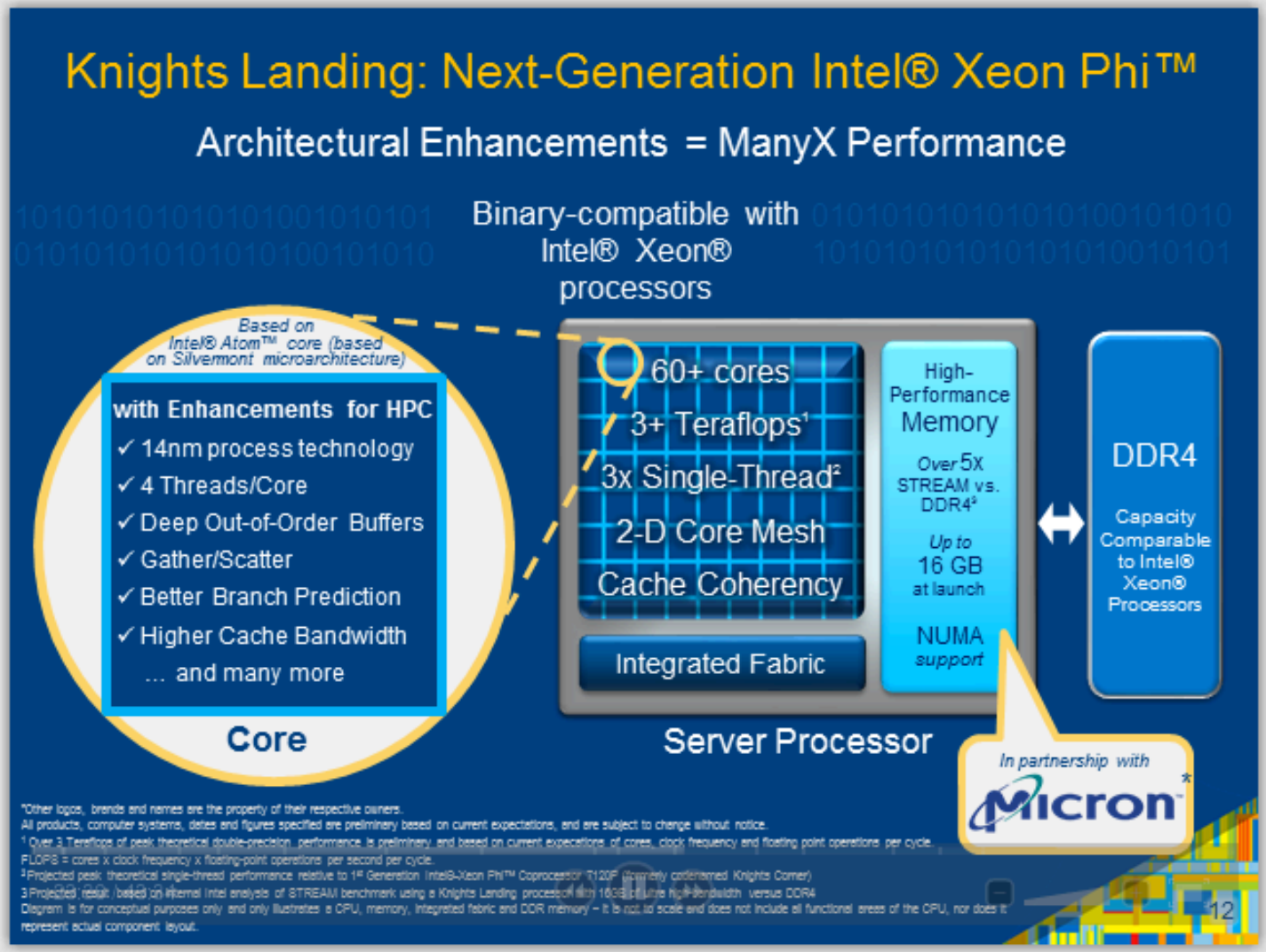
Enables efficient vectorization

Cache "blocking"

Fit important data structures in 16 GB of MCDRAM

MCDRAM memory/core = 235 MB

DDR4 memory/core = 1.4 GB

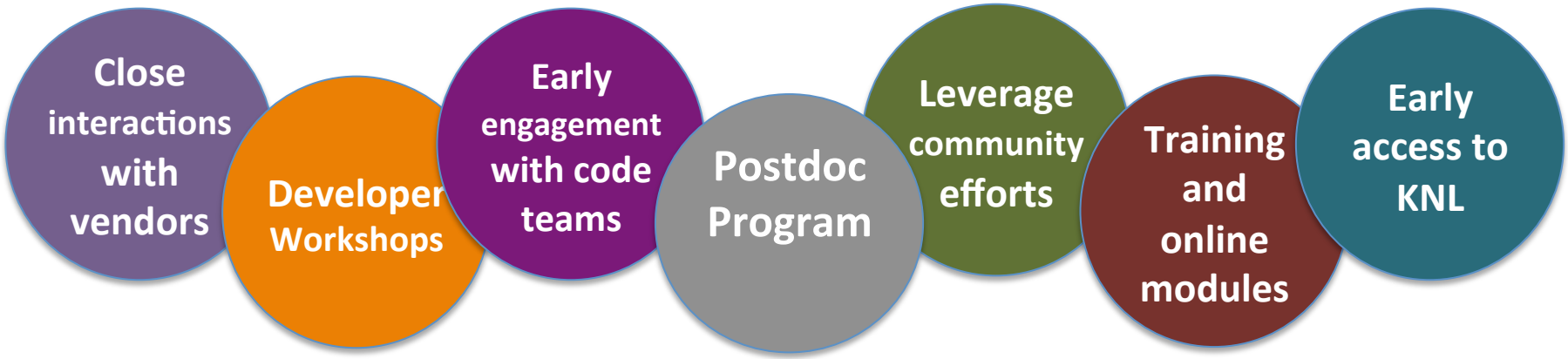


NERSC Exascale Scientific Application Program (NESAP)

Goal: Prepare DOE Office of Science users for many core

Partner closely with ~20 application teams and apply lessons learned to broad NERSC user community

NESAP activities include:



Resources for Code Teams

- **Early access to hardware**
  - Early “white box” test systems and testbeds
  - Early access and significant time on the full Cori system
- **Technical deep dives**
  - Access to Cray and Intel staff on-site staff for application optimization and performance analysis
  - Multi-day deep dive (‘dungeon’ session) with Intel staff at Oregon Campus to examine specific optimization issues
- **User Training Sessions**
  - From NERSC, Cray and Intel staff on OpenMP, vectorization, application profiling
  - Knights Landing architectural briefings from Intel
- **NERSC Staff as Code Team Liaisons (Hands on assistance)**
- **8 Postdocs**

NERSC NESAP Staff


Katie Antypas


Nick Wright


Richard Gerber


Brian Austin


Zhengji Zhao


Helen He


Stephen Leak


Woo-Sun Yang

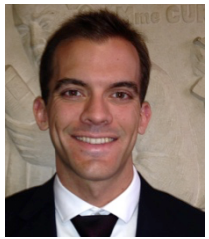

Rebecca Hartman-Baker


Doug Doerfler


Jack Deslippe


Brandon Cook


Thorsten Kurth


Brian Friesen

NESAP Postdocs

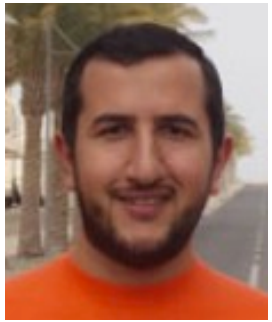
Taylor Barnes
Quantum ESPRESSO

Zahra
Ronaghi

Andrey Ovsyannikov
Chombo-Crunch

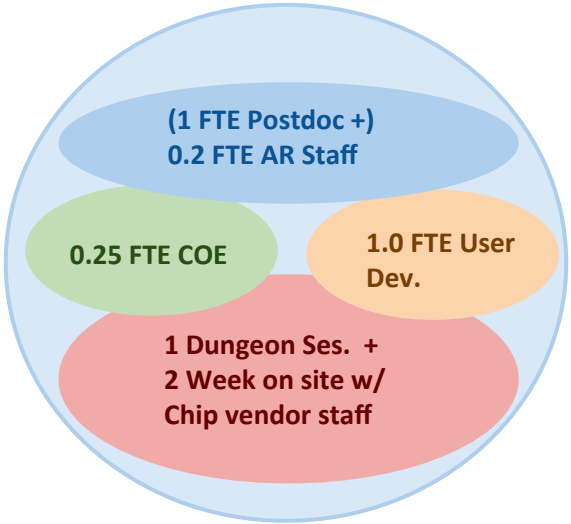
Mathieu Lobet
WARP

Tuomas Koskela
XGC1



Tareq Malas
EMGeo

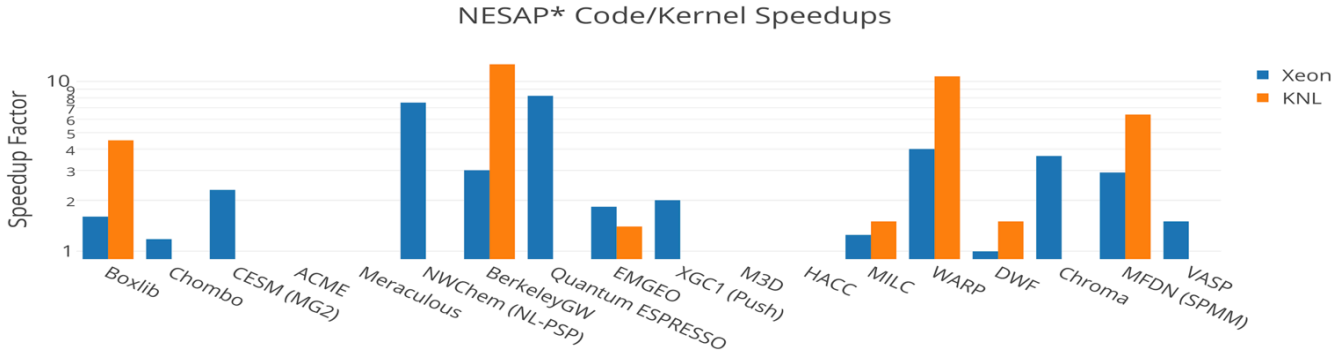
Target Application Team Concept



NESAP Code Status (Work in Progress)

	GFLOP/s KNL	Speedup HBM / DDR	Speedup KNL / Haswell
Chroma (QPhiX)	388 (SP)	4	2.71
MILC	117.4	3.8	2.68
CESM (HOMME)			1.8
MFDN (SPMM)	109.1	3.6	1.62
BGW Sigma	279	1.8	1.61
HACC	1200		1.41
EMGEO (SPMV)	181.0	4.2	1.16

	GFLOP/s KNL	Speedup HBM / DDR	Speedup KNL / Haswell
DWF	600 (SP)		0.95
WARP	60.4	1.2	1.0
Meraculous			0.75
Boxlib		1.13	1.1
Quantum ESPRESSO			1
XGC1 (Push-E)	8.2	0.82	0.2-0.5
Chombo			0.5-1.5



*Speedups from direct/indirect NESAP efforts as well as coordinated activity in NESAP timeframe

What has gone well

Setting requirements for Dungeon Session (Dungeon Session Worksheet).

Engagement with IXPUG and user communities (DFT, Accelerator Design for Exascale Workshop at CRT)

Learned a massive amount about tools and architecture

Large number of NERSC and vendor training events (Vectorization, OpenMP, Tools/Compilers)

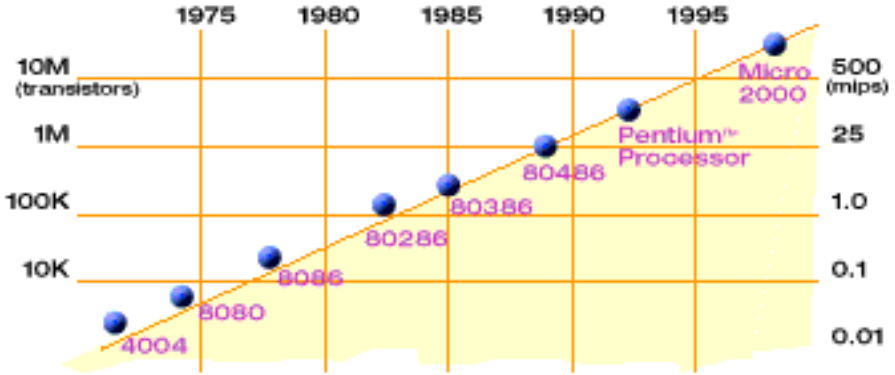
Cray COE VERY helpful to work with. Very pro-active.

Pipelining code work via Cray and Intel experts

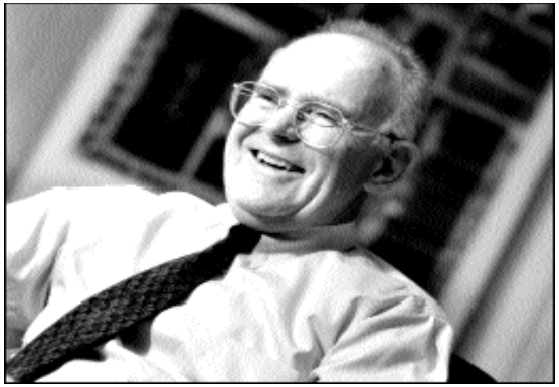
Case studies on the web to transfer knowledge to larger community


EXTRA SLIDES

Why You Need Parallel Computing: The End of Moore's Law?



2X transistors/Chip Every 1.5 years
Called "Moore's Law"



Gordon Moore (co-founder of Intel) predicted in 1965 that the transistor density of semiconductor chips would double roughly every 18 months.

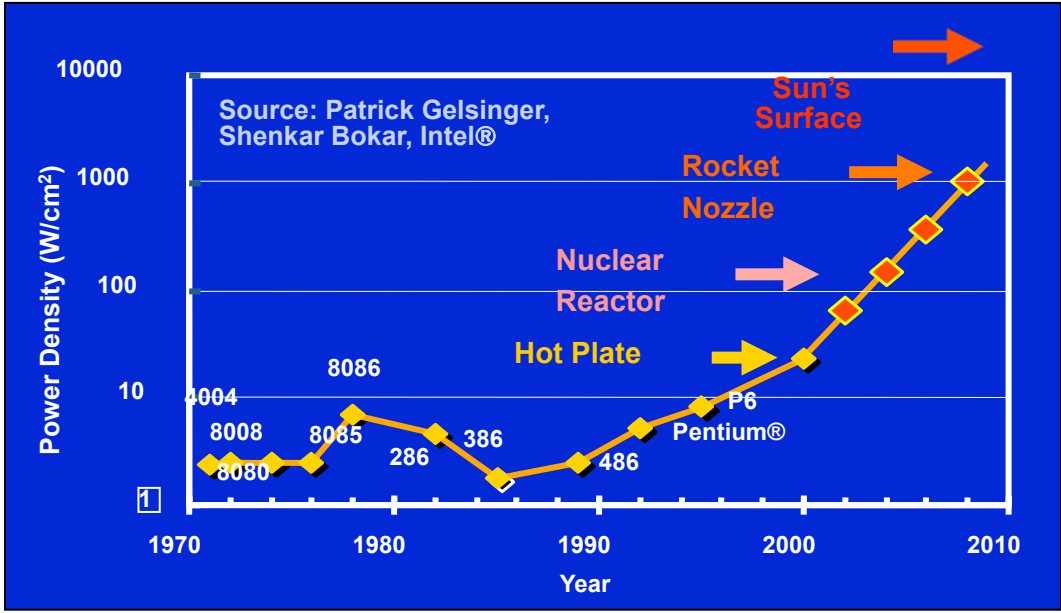
Microprocessors have become smaller, denser, and more powerful.

Slide source: Jack Dongarra

Power Density Limits Serial Performance

Concurrent systems are more power efficient

- Dynamic power is proportional to V^2fC
- Increasing frequency (f) also increases supply voltage (V) → cubic effect
- Increasing cores increases capacitance (C) but only linearly
- Save power by lowering clock speed

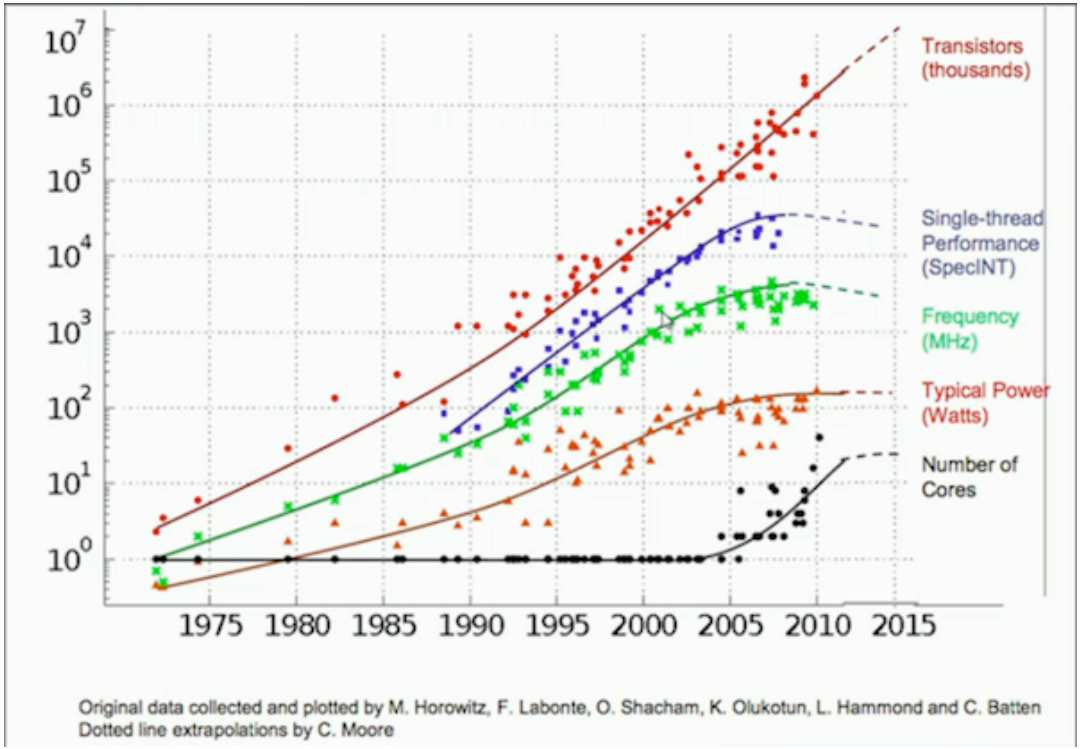


High performance serial processors waste power

- Speculation, dynamic dependence checking, etc. burn power
- Implicit parallelism discovery

More transistors, but not faster serial processors

Processor design for performance and power



Exponential performance continues

Single-thread performance flat or decreasing

Power under control ($P \sim f^{2\text{-}3}$)

Number of cores / die grows

Moore's Law Reinterpreted

Number of cores per chip will increase

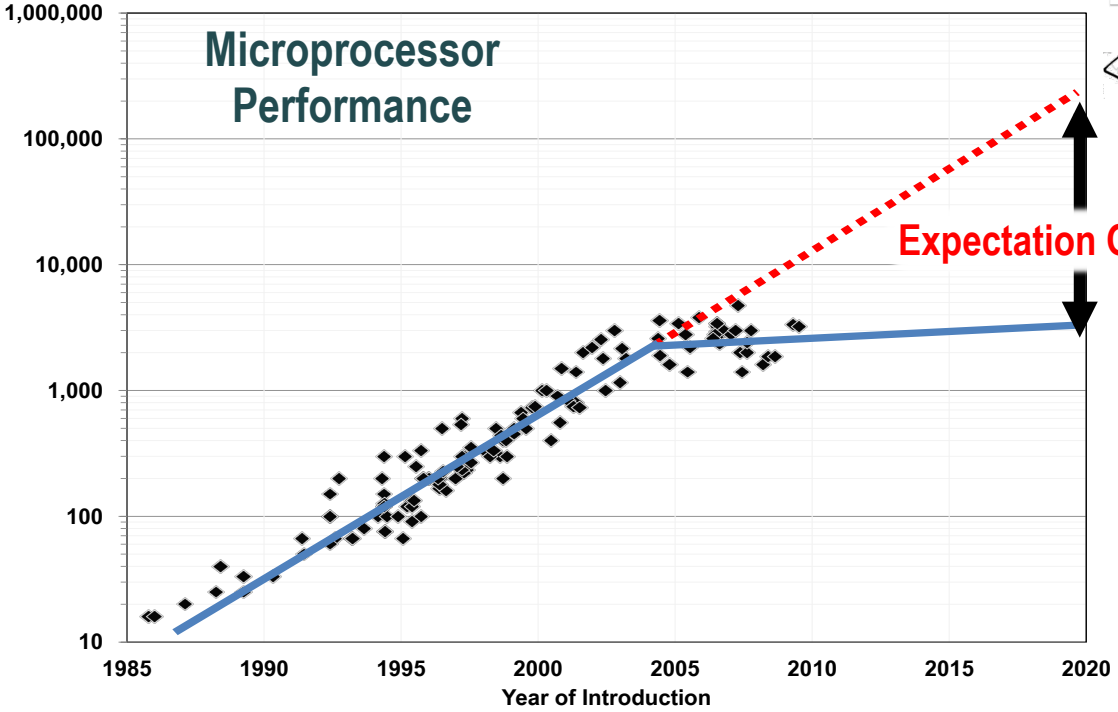
Clock speed will not increase (possibly decrease)

Need to deal with systems with millions of concurrent threads

Need to deal with intra-chip parallelism (OpenMP threads) as well as inter-chip parallelism (MPI)

Any performance gains are going to be the result of increased parallelism, not faster processors

Un-optimized Serial Processing = Left Behind



Modern software users

Do nothing

Application Portability

- DOE Office of Science will have at least two HPC architectures
  - NERSC and ALCF will deploy Cray-Intel Xeon Phi many core based systems in 2016 and 2018
  - OLCF will deploy and IBM Power/NVIDIA based system in 2017
- Question: Are there best practices for achieving performance portability across architectures?
- What is “portability”?
  - ! #ifdef
  - Could be libraries, directives, languages, DSL,
  - Avoid vendor-specific constructs, directives, etc?

Application Portability

- Languages
  - Fortran?
  - Python?
  - C, C++?
  - UPC?
  - DSL?
  - Frameworks (Kokkos, Raja, Tida)