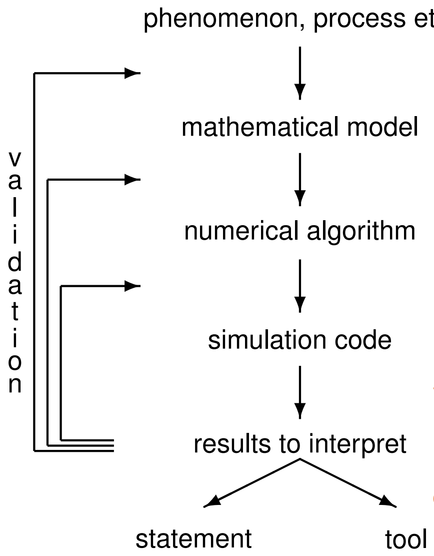
Hybrid Architectures – Why Should I Bother?

CSCS-FoMICS-USI Summer School on
Computer Simulations in Science and Engineering

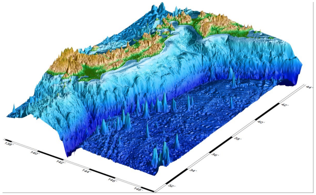
Michael Bader

July 8–19, 2013

The Simulation Pipeline



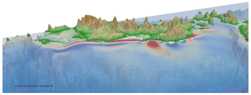
modelling



numerical treatment

parallel implementation

visualization



embedding

Parallel Computing – “Faster, Bigger, More”

Why parallel high performance computing:

- **Response time:** compute a problem in $\frac{1}{p}$ time
  - speed up engineering processes
  - real-time simulations (tsunami warning?)
- **Problem size:** compute a p-times bigger problem
  - simulation of large-/multi-scale phenomena
  - maximal problem size that “fits into the machine”
  - validation of smaller, “operational” models
- **Throughput**: compute p problems at once
  - case and parameter studies, statistical risk scenarios, etc. (hazard maps, data base for tsunami warning, . . .)
  - massively distributed computing (SETI@home, e.g.)

Part I

High Performance Computing in CSE – Past(?) and Present Trends

The Seven Dwarfs of HPC

“dwarfs” = key algorithmic kernels in many scientific computing applications

P. Colella (LBNL), 2004:

1. dense linear algebra
2. sparse linear algebra
3. spectral methods
4. N-body methods
5. **structured grids**
6. **unstructured grids**
7. Monte Carlo

Tsunami & storm-surge simulation:

→ usually PDE solvers on structured or unstructured meshes

→ **SWE**: a simple shallow water solver on Cartesian grids

Computational Science Demands a New Paradigm

Computational simulation must meet three challenges to become a mature partner of theory and experiment

(Post & Votta, 2005)

1. **performance challenge**

   $\rightarrow$ exponential growth of performance, massively parallel architectures

2. **programming challenge**

   $\rightarrow$ new (parallel) programming models

3. **prediction challenge**

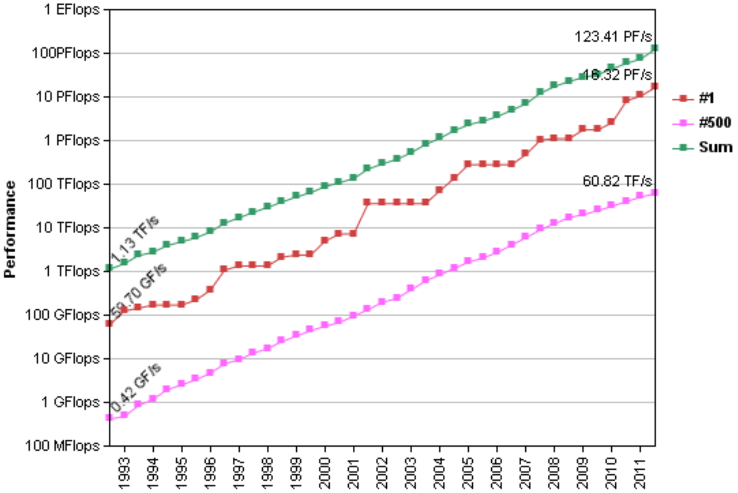
   $\rightarrow$ careful verification and validation of codes; towards **reproducible** simulation experiments

Four Horizons for Enhancing the Performance . . .

. . . of Parallel Simulations Based on Partial Differential Equations
(David Keyes, 2000)

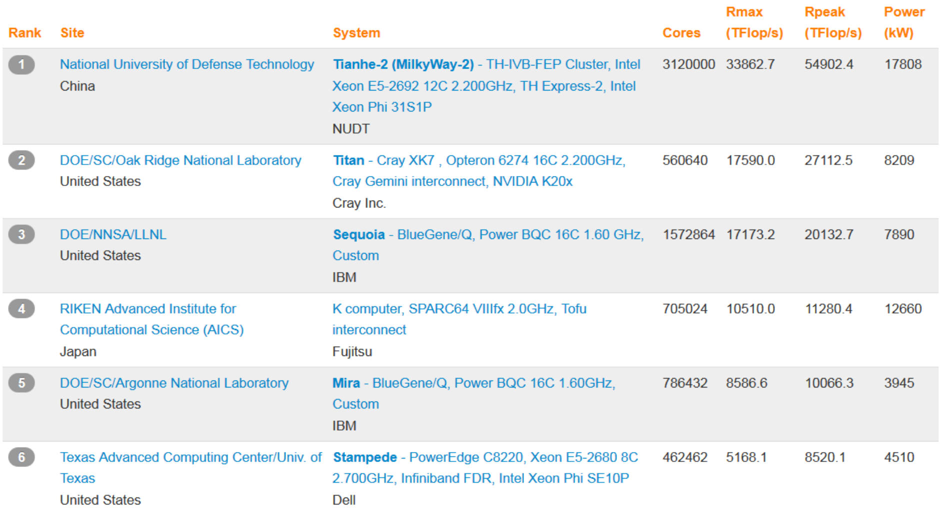
1. Expanded Number of Processors
   $\rightarrow$ in 2000: 1000 cores; in 2010: 200,000 cores
2. More Efficient Use of Faster Processors
   $\rightarrow$ PDF working-sets, cache efficiency
3. More Architecture-Friendly Algorithms
   $\rightarrow$ improve temporal/spatial locality
4. Algorithms Delivering More "Science per Flop"
   $\rightarrow$ adaptivity (in space and time), higher-order methods, fast solvers

Performance Development in Supercomputing



(source: www.top500.org)

Top 500 (www.top500.org) – June 2013



Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	National University of Defense Technology China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3120000	33862.7	54902.4	17808
2	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560640	17590.0	27112.5	8209
3	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1572864	17173.2	20132.7	7890
4	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705024	10510.0	11280.4	12660
5	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786432	8586.6	10066.3	3945
6	Texas Advanced Computing Center/Univ. of Texas United States	**Stampede** - PowerEdge C8220, Xeon E5-2680 8C 2.700GHz, Infiniband FDR, Intel Xeon Phi SE10P Dell	462462	5168.1	8520.1	4510

Top 500 Spotlights – Tianhe-2 and K Computer

Tianhe-2/MilkyWay-2 $\rightarrow$ Intel Xeon Phi (NUDT)

- 3.1 mio cores(!) – Intel Ivy Bridge and Xeon Phi
- Linpack benchmark: 33.8 PFlop/s
- $\approx$ 17 MW power(!!)
- Knights Corner / **Intel Xeon Phi** / Intel MIC as accelerator
- 61 cores, roughly 1.1–1.3 GHz

Titan $\rightarrow$ Cray XK7, NVIDIA K20x (ORNL)

- 18,688 compute nodes; 300,000 Opteron cores
- 18,688 **NVIDIA Tesla K20 GPUs**
- Linpack benchmark: 17.6 PFlop/s
- $\approx$ 8.2 MW power

Top 500 Spotlights – Sequoia and K Computer

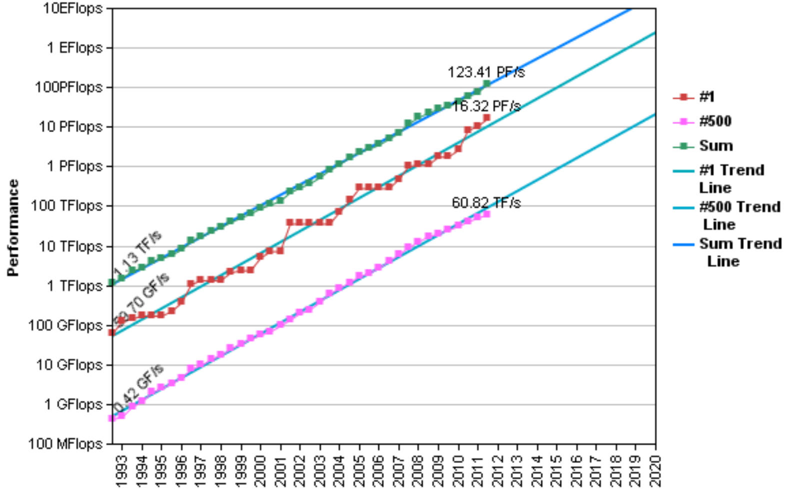
Sequoia $\rightarrow$ IBM BlueGene/Q (LLNL)

- 98,304 compute nodes; 1.6 mio cores
- Linpack benchmark: 17.1 PFlop/s
- $\approx$ 8 MW power

K Computer $\rightarrow$ SPARC64 (RIKEN, Kobe)

- 88,128 processors; 705,024 cores
- Linpack benchmark: 10.51 PFlop/s
- $\approx$ 12 MW power
- SPARC64 VIIIfx 2.0GHz (8-core CPU)

Performance Development in Supercomputing



(source: www.top500.org)

International Exascale Software Project Roadmap

Towards an Exa-Flop/s Platform in 2018 (www.exascale.org):

1. **technology trends**
   - → concurrency, reliability, power consumption, . . .
   - → blueprint of an exascale system: 10-billion-way concurrency, 100 million to 1 billion cores, 10-to-100-way concurrency per core, hundreds of cores per die, . . .
2. **science trends**
   - → climate, high-energy physics, nuclear physics, fusion energy sciences, materials science and chemistry, . . .
3. **X-stack** (software stack for exascale)
   - → energy, resiliency, heterogeneity, I/O and memory
4. **Polito-economic trends**
   - → exascale systems run by government labs, used by CSE scientists

Exascale Roadmap

"Aggressively Designed Strawman Architecture"

Level	What	Perform.	Power	RAM
FPU	FPU, regs,. instr.-memory	1.5 Gflops	30mW	
Core	4 FPUs, L1	6 Gflops	141mW	
Proc. Chip	**742 cores**, L2/L3, Interconn.	4.5 Tflops	214W	
Node	Proc. chip, DRAM	4.5 Tflops	230W	**16 GB**
Group	12 proc. chips, routers	54 Tflops	3.5KW	192 GB
rack	32 groups	1.7 Pflops	116KW	6.1 TB
System	**583 racks**	1 Eflops	67.7MW	3.6 PB

approx. 285,000 cores per rack; 166 mio cores in total

Source:

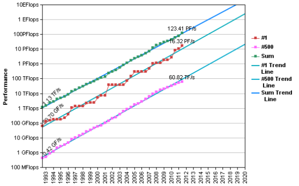
ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems

Exascale Roadmap – Should You Bother?

Your department's compute cluster in 5 years?

- a **Petaflop System**!
- "one rack of the Exaflop system"
  $\rightarrow$ using the same/similar hardware
- extrapolated example machine:
  - peak performance: 1.7 PFlop/s
  - 6 TB RAM, 60 GB cache memory
  - "total concurrency": $1.1 \cdot 10^6$
  - number of cores: $280,000$
  - number of chips: 384

Source: ExaScale Software Study: Software Challenges in Extreme Scale Systems



Your Department's PetaFlop/s Cluster in 5 Years?

Tianhe-1A (Tianjin, China; Top500 # 10)

- 14,336 Xeon X5670 CPUs
- **7,168 Nvidia Tesla M2050 GPUs**
- Linpack benchmark: $\approx$ 2.6 PFlop/s
- $\approx$ 4 MW power

Stampede (Intel, Top500 # 6)

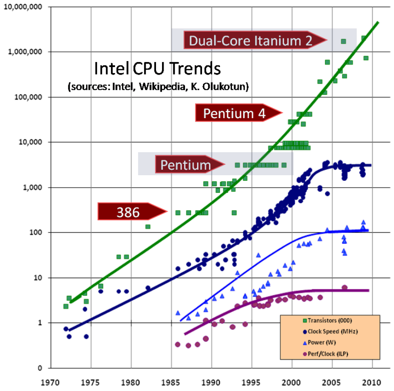
- 102,400 cores (incl. Xeon Phi: MIC/"many integrated cores")
- Linpack benchmark: $\approx$ 5 PFlop/s
- Knights Corner / **Intel Xeon Phi** / Intel MIC as accelerator
- 61 cores, roughly 1.1–1.3 GHz
- wider vector FP units: 64 bytes (i.e., 16 floats, 8 doubles)
- $\approx$ 4.5 MW power

Free Lunch is Over$^{(*)}$

. . . actually already over for quite some time!

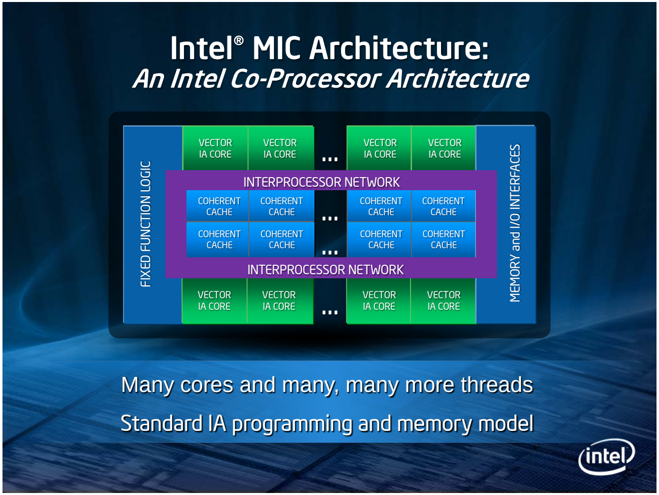
Speedup of your software can only come from parallelism:

- clock speed of CPU has stalled
- instruction-level parallelism per core has stalled
- number of cores is growing
- size of vector units is growing



$^{(*)}$ Quote and image taken from: H. Sutter, *The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software*, Dr. Dobb's Journal 30(3), March 2005.

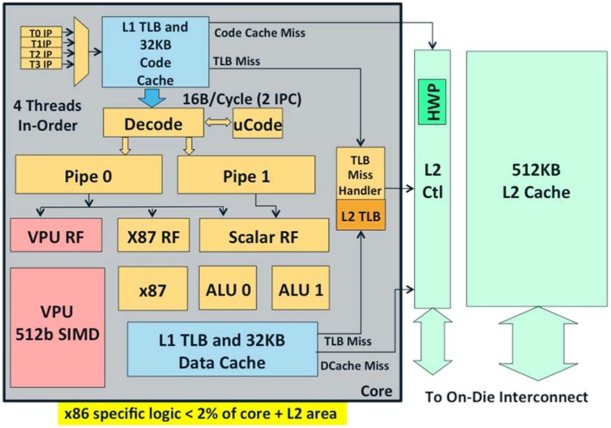
Manycore CPU – Intel MIC Architecture



(source: Intel/K. Skaugen – SC'10 keynote presentation)

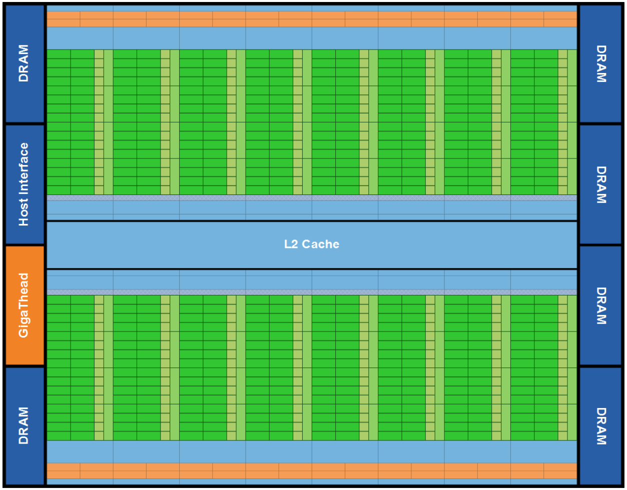
Manycore CPU – Intel MIC Architecture (2)

Diagram of a Knights Corner core:



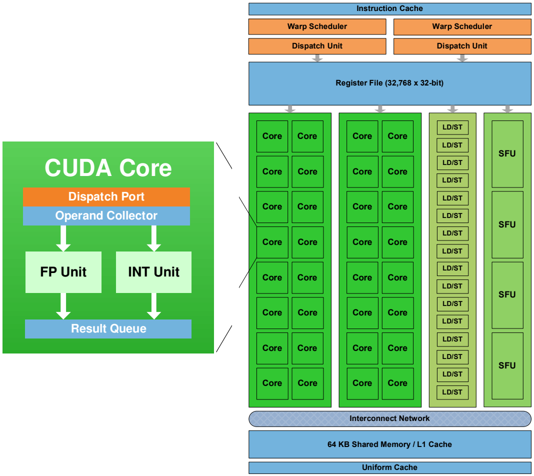
(source: An Overview of Programming for Intel Xeon processors and Intel Xeon Phi coprocessors)

GPGPU – NVIDIA Fermi



(source: NVIDIA – Fermi Whitepaper)

GPGPU – NVIDIA Fermi (2)



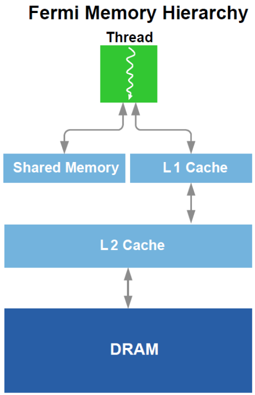
(source: NVIDIA – Fermi Whitepaper)

Fermi Streaming Multiprocessor (SM)

GPGPU – NVIDIA Fermi (3)

General Purpose Graphics Processing Unit:

- 512 CUDA cores
- improved double precision performance
- shared vs. global memory
- new: L1 und L2 cache (768 KB)
- trend from GPU towards CPU?



Parallel Computing Paradigms

Not exactly sure how the hardware will look like . . .
(CPU-style, GPU-style, something new?)

However: massively parallel programming required

- revival of vector computing
  $\rightarrow$ several/many FPUs performing the same operation
- hybrid/heterogenous achitectures
  $\rightarrow$ different kind of cores; dedicated accelerator hardware
- different access to memory
  $\rightarrow$ cache and cache coherency
  $\rightarrow$ small amount of memory per core
- our concern in this course: **data parallelism**
  $\rightarrow$ vectorisation (and a look into GPU computing)