Things to Watch For in this Training

Preparing yourself for future hardware trends

- **CPU Clock rates are stalled (not getting faster)**
  - # nodes is about the same, but # processors is growing exponentially
  - Time to start thinking of parallelism from node level *(cores will drive you crazy)*
  - Go to Hybrid Parallelism to tackle intra-node parallelism so you can focus on # of nodes parallelism rather than # of cores
- **Memory capacity not growing as fast as FLOPs**
  - Memory per node is still growing, but per core is diminishing
  - Threading (OpenMP) on node can help conserve memory
- **Diminishing BW/flop makes locality essential**
  - *Vertical locality:* Careful cache-blocking and use of prefetch
  - *Horizontal locality:* NUMA effects (memory affinity: must always be sure to access data where it was first touched)

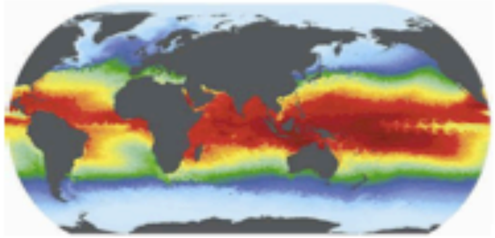
What to get from this talk

- **Hardware Trends**
  - – Exponential growth in explicit on-chip parallelism
  - – Reduced memory per core
  - – Heterogeneous computing platforms (E.G. GPU)
- **Software Response**
  - – Need to express more explicit parallelism
  - – New programming models emerging to express
  - – Increased emphasis on strong scaling
- **What you should get from this lecture**
  - – Understand emerging technology trends so that you can craft a strategy for transitioning to a stable/viable long-term programming environment

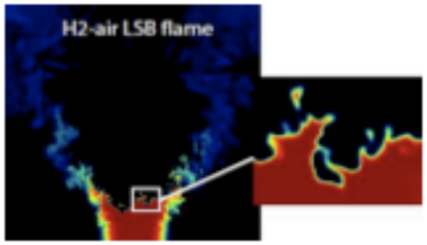
Process for identifying exascale applications and technology for DOE missions ensures broad community input

- **Town Hall Meetings April-June 2007**
- **Scientific Grand Challenges Workshops Nov, 2008 – Oct, 2009**
  - Climate Science (11/08),
  - High Energy Physics (12/08),
  - Nuclear Physics (1/09),
  - Fusion Energy (3/09),
  - Nuclear Energy (5/09),
  - Biology (8/09),
  - Material Science and Chemistry (8/09),
  - National Security (10/09)
- **Exascale Steering Committee**
  - "Denver" vendor NDA visits 8/2009
  - SC09 vendor feedback meetings
  - Extreme Architecture and Technology Workshop 12/2009
- **International Exascale Software Project**
  - Santa Fe, NM 4/2009
  - Paris, France 6/2009
  - Tsukuba, Japan 10/2009

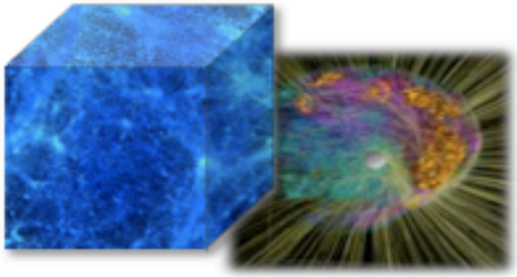
Modeling and Simulation at the Exascale for Energy and the Environment



Scientific Grand Challenges
FOREFRONT QUESTIONS IN NUCLEAR SCIENCE AND THE ROLE OF COMPUTING AT THE EXTREME SCALE
Scientific Grand Challenges
CHALLENGES FOR UNDERSTANDING THE QUANTUM UNIVERSE AND THE ROLE OF COMPUTING AT THE EXTREME SCALE
Scientific Grand Challenges
CHALLENGES IN CLIMATE CHANGE SCIENCE AND THE ROLE OF COMPUTING AT THE EXTREME SCALE
ENERGY
ENERGY
ENERGY



MISSION IMPERATIVES



FUNDAMENTAL SCIENCE

Technology Disruptions on the Path to Exascale

- **Gigaflops to Teraflops was highly disruptive**
  - Moved from vector machines to MPPs with message passing
  - Required new algorithms and software
- **Teraflops to Petaflops was *not* very disruptive**
  - Continued with MPI+Fortran/C/C++ with incremental advances
- **Petaflops to Exaflops will be highly disruptive**
  - No clock increases → hundreds of simple "cores" per chip
  - Less memory and bandwidth → cores are not MPI engines
  - x86 too energy intensive → more technology diversity (GPUs/ accel.)
  - Programmer controlled memory hierarchies likely
- **Computing at every scale will be *transformed (not just exascale)***

Systems	**2009**	**2015 +1/-0**	**2018 +1/-0**
System peak	**2 Peta**	**100-300 Peta**	**1 Exa**
Power	**6 MW**	**~15 MW**	**~20 MW**
System memory	0.3 PB	5 PB	64 PB (+)
Node performance	125 GF	0.5 TF or 7 TF	1-2 or 10TF
Node memory BW	25 GB/s	1-2TB/s	2-4TB/s
Node concurrency	12	O(100)	O(1k) or 10k
Total Node Interconnect BW	3.5 GB/s	100-200 GB/s 10:1 vs memory bandwidth 2:1 alternative	200-400GB/s (1:4 or 1:8 from memory BW)
System size (nodes)	18,700	50,000 or 500,000	O(100,000) or O(1M)
Total concurrency	225,000	O(100,000,000) *O(10)-O(50) to hide latency	O(billion) * O(10) to O (100) for latency hiding
Storage	15 PB	150 PB	500-1000 PB (>10x system memory is min)
IO	0.2 TB	10 TB/s	60 TB/s (how long to drain the machine)
MTTI	days	O(1day)	O(1 day)

Systems	2009	2015 +1/-0	2018 +1/-0
System peak	**2 Peta**	**100-300 Peta**	**1 Exa**
Power	**6 MW**	**~15 MW**	**~20 MW**
System memory	0.3 PB	5 PB	64 PB (+)
Node performance	125 GF	0.5 TF or 7 TF	1-2 or 10TF
Node memory BW	25 GB/s	1-2TB/s	2-4TB/s
Node concurrency	12	O(100)	O(1k) or 10k
Total Node Interconnect BW	3.5 GB/s	100-200 GB/s 10:1 vs memory bandwidth 2:1 alternative	200-400GB/s (1:4 or 1:8 from memory BW)
System size (nodes)	18,700	50,000 or 500,000	O(100,000) or O(1M)
Total concurrency	225,000	O(100,000,000) *O(10)-O(50) to hide latency	O(billion) * O(10) to O (100) for latency hiding
Storage	15 PB	150 PB	500-1000 PB (>10x system memory is min)
IO	0.2 TB	10 TB/s	60 TB/s (how long to drain the machine)
MTTI	days	O(1day)	O(1 day)

Exascale Roadmap Trends are already apparent today

This is not idle speculation

NERSC-6
Grace “Hopper”

Cray XE6

Performance

- 1.2 PF Peak
- 1.05 PF HPL (#5)

Processor

- AMD MagnyCours
- 2.1 GHz 12-core
- 8.4 GFLOPs/core
- 24 cores/node
- 32-64 GB DDR3-1333 per node

System

- Gemini Interconnect (3D torus)
- 6392 nodes
- 153,408 total cores

I/O

- 2PB disk space
- 70GB/s peak I/O Bandwidth

Evolution from Franklin (XT4) to Hopper (XE6)

Cray XT4: Franklin

Performance: 0.352 PF Peak
0.266 TF HPL (#26, debut@ #8)
Processor: AMD Budapest
4-core 2.3 GHz (9.2 GF/core)
4 cores/node
Memory: DDR2 667MHz
8 GB/node @ 21GB/s
2 GB/core
System
9,572 nodes (38,288 total cores)
Interconnect: SeaStar2 3D torus,
1.6GB/s measured @ 6-8usec
I/O
12GB/s peak I/O Bandwidth
0.436 PB disk space

Cray XE6: Hopper

Performance: 1.288 PF Peak
1.05 PF HPL (#5)
Processor: AMD MagnyCours
12-core 2.1 GHz (8.4 GF/core)
24 cores/node
Memory: DDR3 1333MHz
32-64 GB/node @ 84GB/s
1.3 - 2.6 GB/core
System
6,392 nodes (153,408 total cores)
Interconnect: Gemini 3D torus,
8.3GB/s measured @ 2usec
I/O
70GB/s peak I/O Bandwidth
2PB disk space

Evolution from Franklin (XT4) to Hopper (XE6)

Cray XT4: Franklin

Performance: 0.352 TF Peak
0.266 TF HPL (#26, debut@ #8)

Processor: AMD Budapest
4-core 2.3 GHz (9.2 GF/core)
4 cores/node

Memory: DDR2 667MHz
8 GB/node @ 21GB/s
2 GB/core

System
9,572 nodes (38,288 total cores)

Interconnect: SeaStar2 3D torus,
1.6GB/s measured @ 6-8usec

I/O
12GB/s peak I/O Bandwidth
0.436 PB disk space

Cray XE6: Hopper

Performance: 1.288 PF Peak
1.05 PF HPL (#5)

Processor: AMD MagnyCours
12-core 2.1 GHz (8.4 GF/core)
24 cores/node

Memory: DDR3 1333MHz
32-64 GB/node @ 84GB/s
1.3 - 2.6 GB/core

System
6,392 nodes (153,408 total cores)

Interconnect: Gemini 3D torus,
8.3GB/s measured @ 2usec

I/O
70GB/s peak I/O Bandwidth
2PB disk space

Evolution from Franklin (XT4) to Hopper (XE6)

Cray XT4: Franklin

Performance: 0.352 TF Peak

0.266 TF HPL (#26, debut@ #8)

Processor: AMD Budapest

4-core 2.3 GHz (9.2 GF/core)

4 cores/node

Memory: DDR2 667MHz

8 GB/node @ 21GB/s

2 GB/core

System

9,572 nodes (38,288 total cores)

Interconnect: SeaStar2 3D torus,

1.6GB/s measured @ 6-8usec

I/O

12GB/s peak I/O Bandwidth

0.436 PB disk space

Cray XE6: Hopper

Performance: 1.288 PF Peak

1.05 PF HPL (#5)

Processor: AMD MagnyCours

12-core 2.1 GHz (8.4 GF/core)

24 cores/node

Memory: DDR3 1333MHz

32-64 GB/node @ 84GB/s

1.3 - 2.6 GB/core

System

6,392 nodes (153,408 total cores)

Interconnect: Gemini 3D torus,

8.3GB/s measured @ 2usec

I/O

70GB/s peak I/O Bandwidth

2PB disk space

Evolution from Franklin (XT4) to Hopper (XE6)

Cray XT4: Franklin

Performance: 0.352 TF Peak
0.266 TF HPL (#26, debut@ #8)
Processor: AMD Budapest
4-core 2.3 GHz (9.2 GF/core)
4 cores/node
Memory: DDR2 667MHz
8 GB/node @ 21GB/s
2 GB/core
System
9,572 nodes (38,288 total cores)
Interconnect: SeaStar2 3D torus,
1.6GB/s measured @ 6-8usec
I/O
12GB/s peak I/O Bandwidth
0.436 PB disk space

Cray XE6: Hopper

Performance: 1.288 PF Peak
1.05 PF HPL (#5)
Processor: AMD MagnyCours
12-core 2.1 GHz (8.4 GF/core)
24 cores/node
Memory: DDR3 1333MHz
32-64 GB/node @ 84GB/s
1.3 - 2.6 GB/core
System
6,392 nodes (153,408 total cores)
Interconnect: Gemini 3D torus,
8.3GB/s measured @ 2usec
I/O
70GB/s peak I/O Bandwidth
2PB disk space

Systems	2009	2015 +1/-0	2018 +1/-0
System peak	**2 Peta**	**100-300 Peta**	**1 Exa**
Power	**6 MW**	**~15 MW**	**~20 MW**
System memory	0.3 PB	5 PB	64 PB (+)
Node performance	125 GF	0.5 TF or 7 TF	2 TF or 10TF
Node memory BW	25 GB/s	0.2TB/s or 0.5TB/s	0.4TB/s or 1TB/s
Node concurrency	12	O(100)	O(1k) or 10k
Total Node Interconnect BW	3.5 GB/s	100-200 GB/s 10:1 vs memory bandwidth 2:1 alternative	200-400GB/s (1:4 or 1:8 from memory BW)
System size (nodes)	18,700	50,000 or 500,000	O(100,000) or O(1M)
Total concurrency	225,000	O(100,000,000) *O(10)-O(50) to hide latency	O(billion) * O(10) to O (100) for latency hiding
Storage	15 PB	150 PB	500-1000 PB (>10x system memory is min)
IO	0.2 TB	10 TB/s	60 TB/s (how long to drain the machine)
MTTI	days	O(1day)	O(1 day)

Systems	2009	2015 +1/-0	2018 +1/-0
System peak	2 Peta	100-300 Peta	1 Exa
Power	6 MW	~15 MW	~20 MW
System memory	0.3 PB	5 PB	64 PB (+)
Nc le performance	125 GF	0.5 TF or 7	2 TF or 10TF
Node m BW	2	0.2TB s	0.4TB/s or 1TB/s
Node concur		O	r 10k
Total Node Interco			400GB/s y
System size (			r O(1M)
Total conc			) for latency hiding
Storage		15	00 PB (>10x sy nemory is min)
IO	0.2	10 TB/s	60 TB/s (how long to drain the machine)
MTTI	day	O(1day)	O(1 day)

All the bad stuff they are warning you about for exascale is already happening!
(its only going to get worse)

A Revolution is Underway

- **Rapidly Changing Technology Landscape**
  - **Evolutionary change between nodes** *(10x more explicit parallelism)*
  - **Revolutionary change within node** *(100x more parallelism, with diminished memory capacity and bandwidth)*
  - **Multiple Technology Paths** *(GPU, manycore/embedded, x86/PowerX)*

- **The technology disruption will be pervasive *(not just exascale)***
  - ***Assumptions that our current software infrastructure is built upon are no longer valid***
  - **Applications, Algorithms, System Software *will all break***
  - **As significant as migration from vector to MPP (early 90's)**

- **Need a new approach to ensuring continued application performance improvements**
  - **This isn't just about Exaflops – this is for all system scales**

Part I

Power Crisis in HPC

Traditional Sources of Performance Improvement are Flat-Lining

- **New Constraints**
  - 15 years of *exponential* clock rate growth has ended
- **But Moore's Law continues!**
  - How do we use all of those transistors to keep performance increasing at historical rates?
  - Industry Response: #cores per chip doubles every 18 months *instead* of clock frequency!

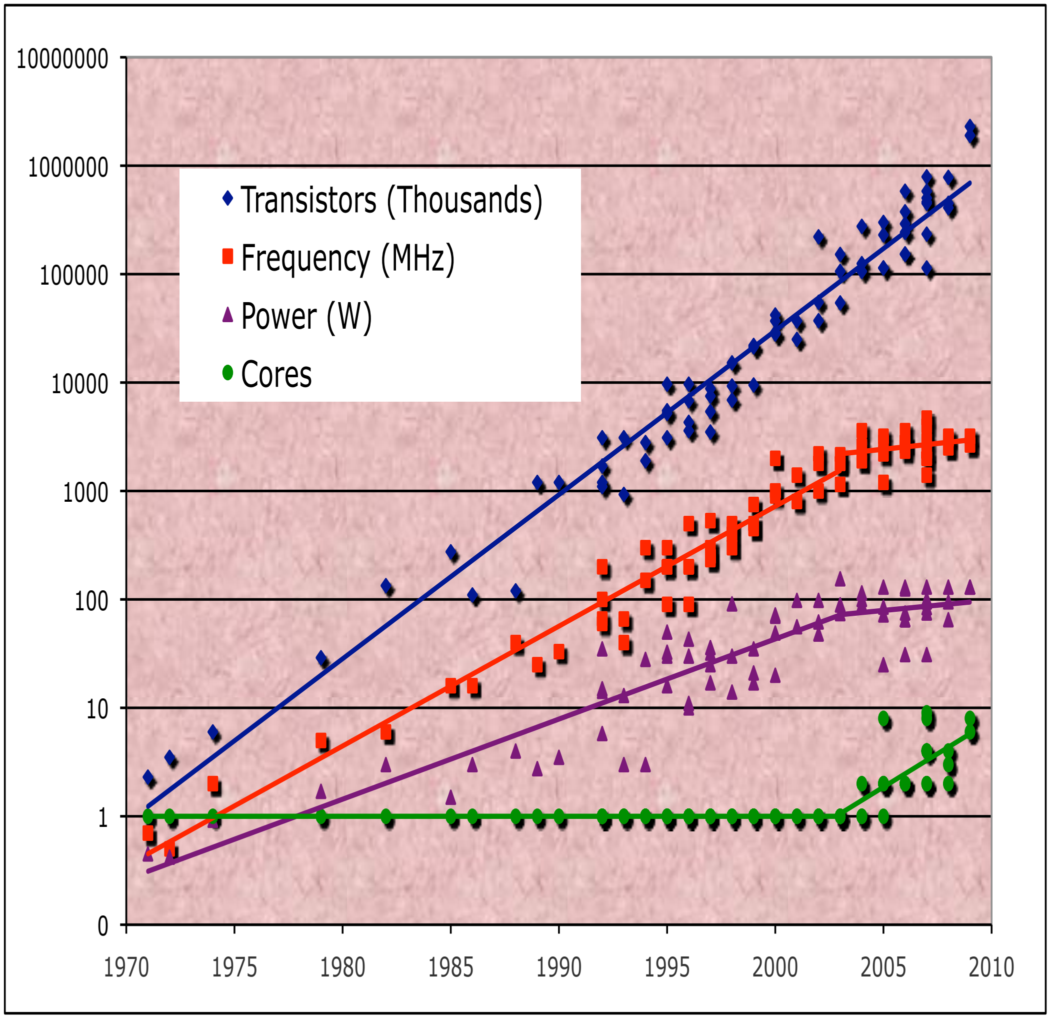
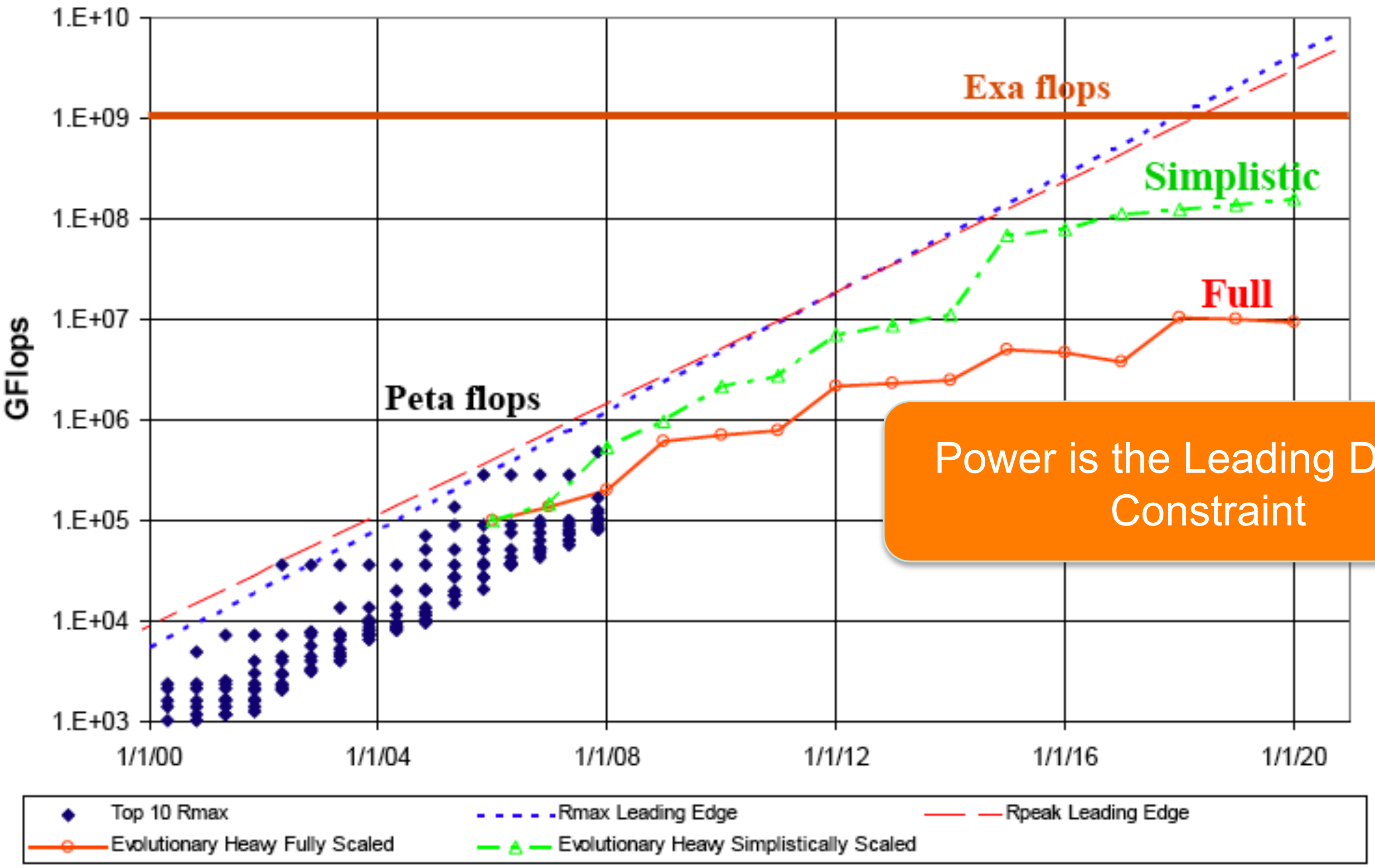


Figure adapted from Kunle Olukotun, Lance Hammond, Herb Sutter, and Burton Smith

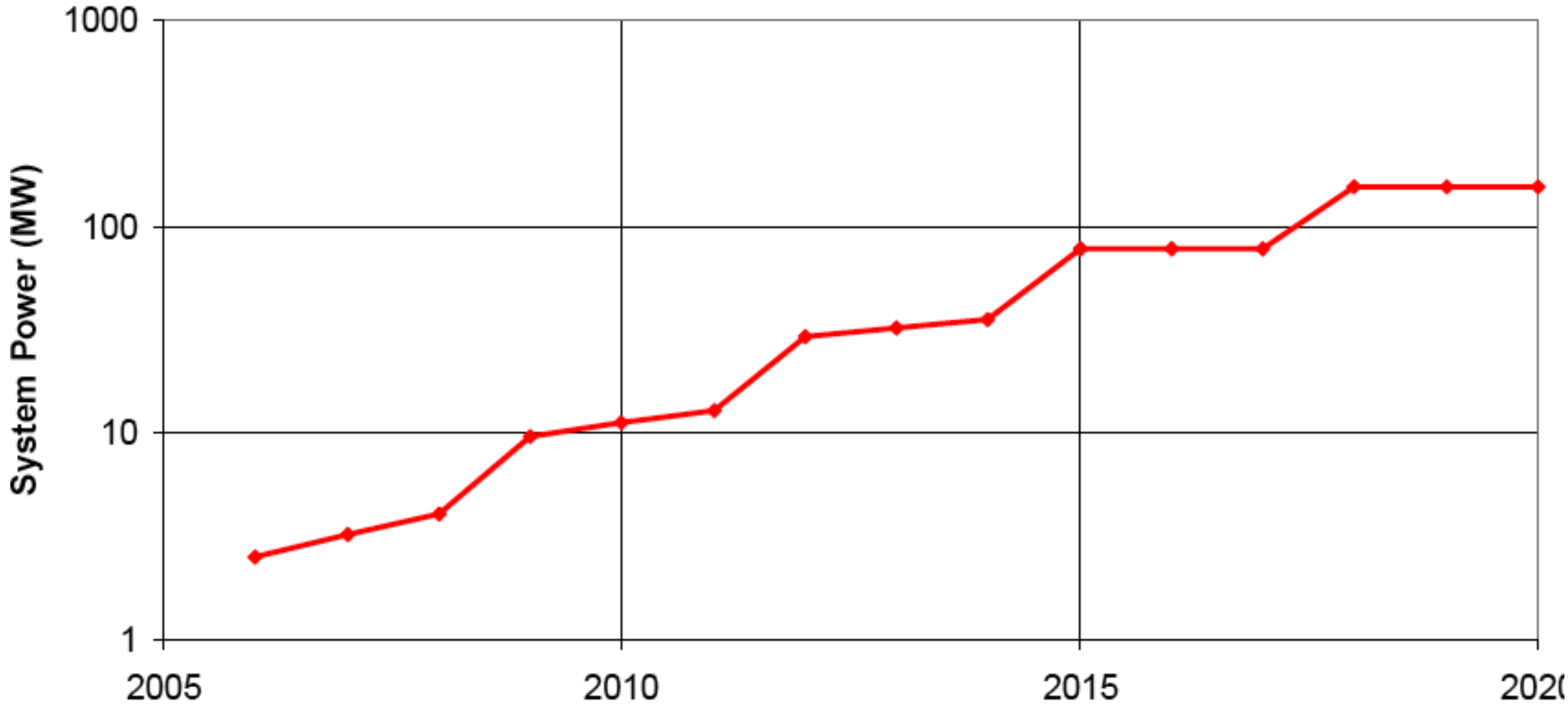
Current Technology Roadmaps will Depart from Historical Gains



Power is the Leading Design Constraint

From Peter Kogge, DARPA Exascale Study

… and the power costs will still be staggering

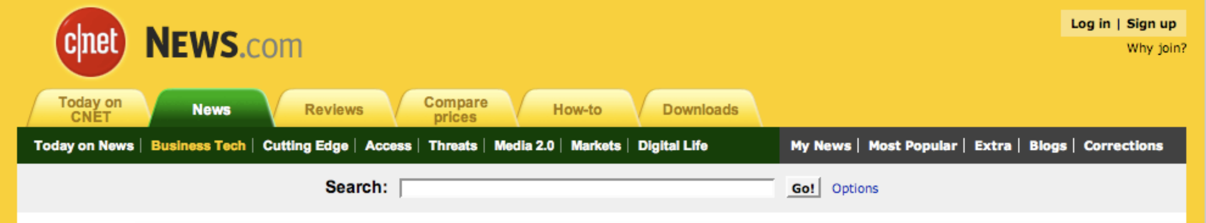


From Peter Kogge,
DARPA Exascale Study

$1M per megawatt per year! (with CHEAP power)

Power is an Industry Wide Problem

(2% of US power consumption and growing)


cnet NEWS.com
Log in | Sign up
Why join?
Today on CNET
News
Reviews
Compare prices
How-to
Downloads
Today on News | Business Tech | Cutting Edge | Access | Threats | Media 2.0 | Markets | Digital Life
My News | Most Popular | Extra | Blogs | Corrections
Search:
Go!
Options
P

The future of the entire computer industry is at stake *(clouds too)*

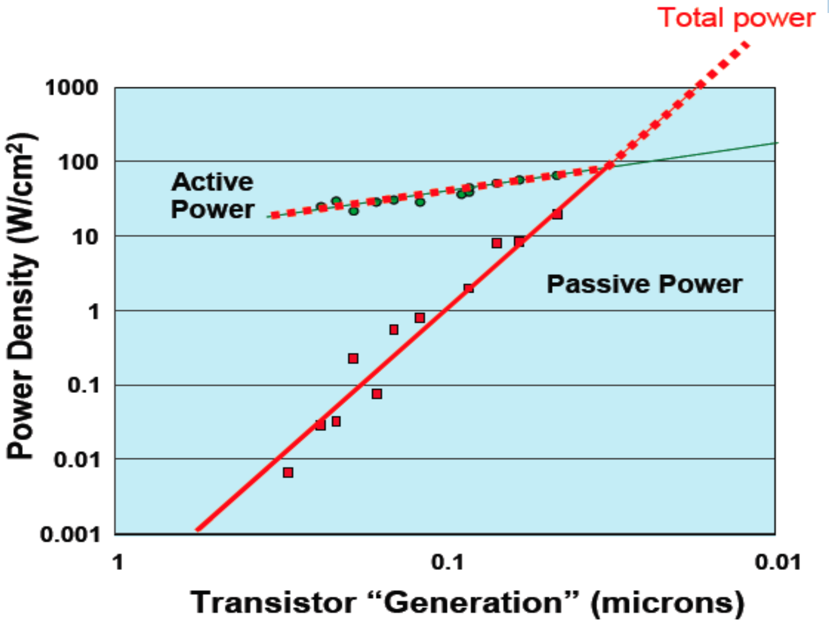
From cell phones to supercomputers

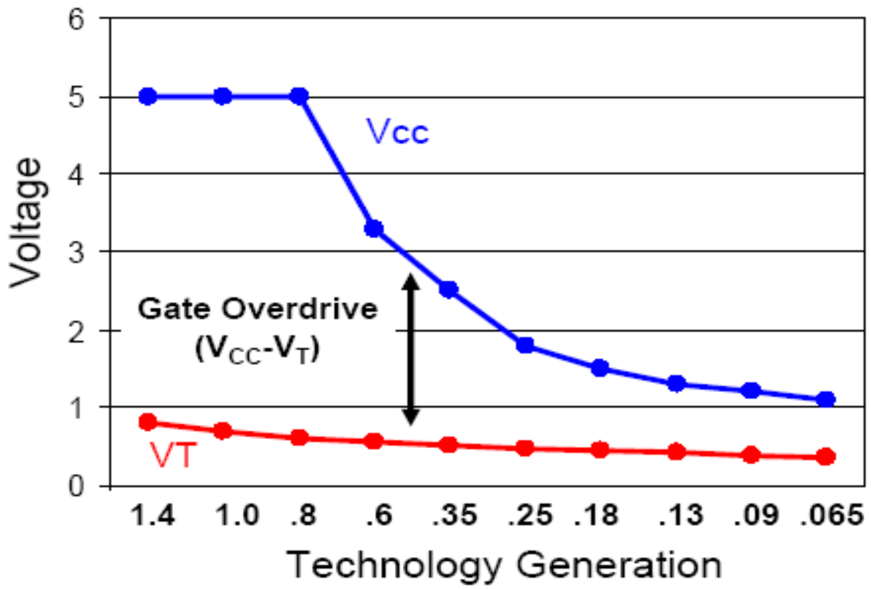
New Google Plant in The Dulles, Oregon, from NYT, June 14, 2006

Relocate to Iceland?

Or Utah

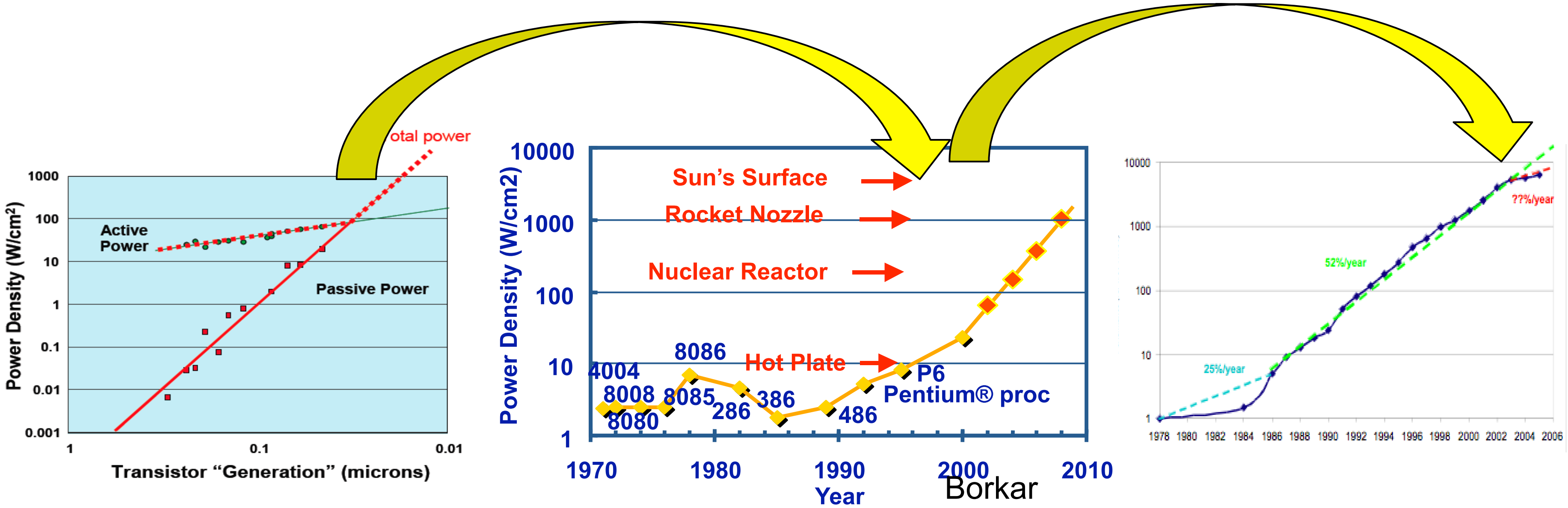
Primary Design Constraint: *POWER*





- Total Energy = Active Power + Leakage Power
- Active Power = $C * V^2 * F$
  - This is energy required to charge & discharge capacitance of transistor
  - Dennard recognized capacitance is reduced proportional to die shrink
  - Power neutral if you drop supply voltage and increase clock frequency
- Leakage Power = $V * I_{leakage}$
  - Voltage is so low that cannot turn transistor entirely on or off
  - So transistors must either "leak" or run much slower

Primary Design Constraint: *POWER*



- No room for Dennard scaling (leakage power caught up to us)

- Without changes, we will get exponential growth in power

- *So, clock frequencies stalled in 2002 (Patterson Graph)*

The Challenge of Our Decade

Where do we get a 1000x improvement in performance with only a 10x increase in power?

How do you achieve this in 10 years with a finite development budget?

Where do we get 1000x performance improvement for 10x power?

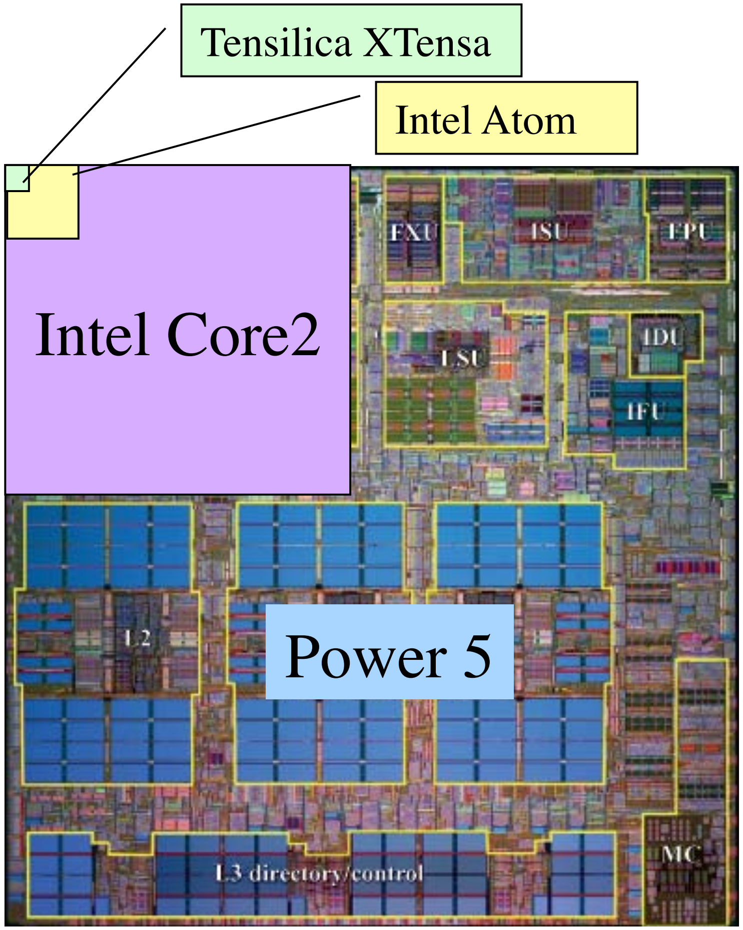
1. Processors
2. Data movement
3. Memory
4. System-wide data movement
5. Resilience Mechanisms

Processors: What are the problems?

(Lessons from the Berkeley View)

- **Current Hardware/Lithography Constraints**
  - **Power limits leading edge chip designs**
    - Intel Tejas Pentium 4 cancelled due to power issues
  - **Yield on leading edge processes dropping dramatically**
    - IBM quotes yields of 10 – 20% on 8-processor Cell
  - **Design/validation leading edge chip is becoming unmanageable**
    - Verification teams > design teams on leading edge processors
- **Solution: Small Is Beautiful**
  - **Simpler (5- to 9-stage pipelined) CPU cores**
    - Small cores not much slower than large cores
  - **Parallel is energy efficient path to performance:CV^{2}F**
    - Lower threshold and supply voltages lowers energy per op
  - **Redundant processors can improve chip yield**
    - Cisco Metro 188 CPUs + 4 spares; Sun Niagara sells 6 or 8 CPUs
  - **Small, regular processing elements easier to verify**

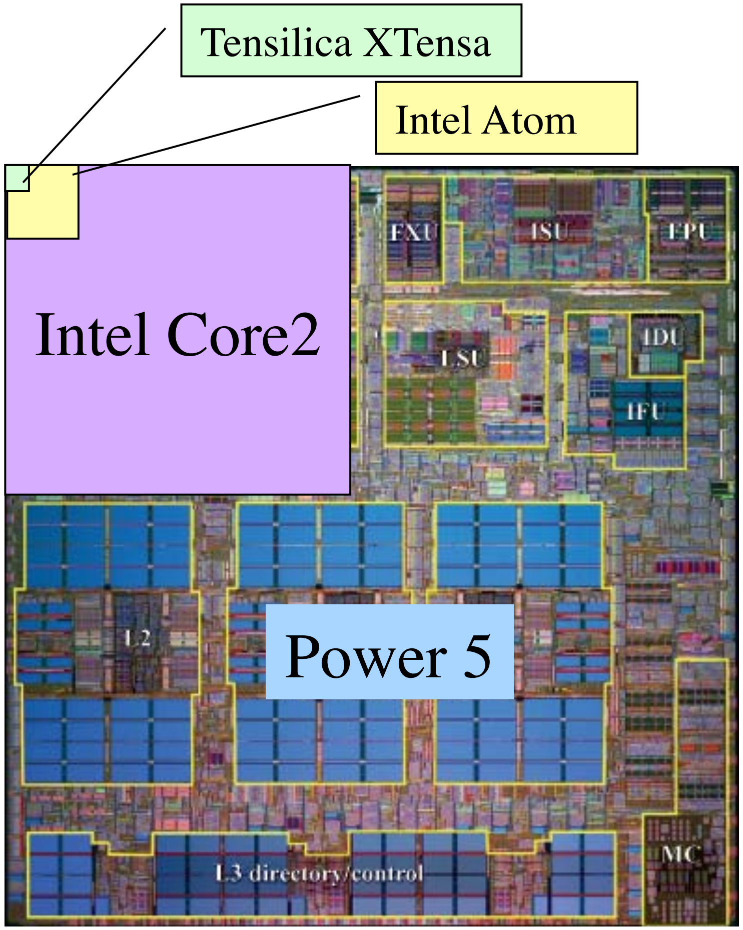
Low-Power Design Principles



- Cubic power improvement with lower clock rate due to V^2F
- Slower clock rates enable use of simpler cores
- Simpler cores use less area (lower leakage) and reduce cost
- Tailor design to application to REDUCE WASTE

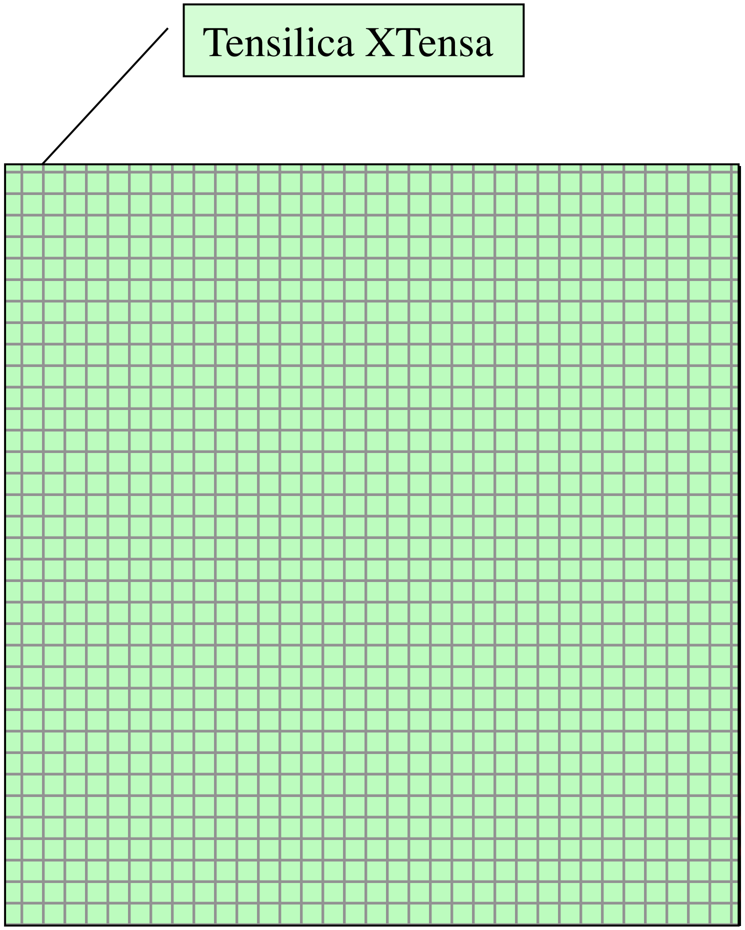
This is how iPhones and MP3 players are designed to maximize battery life and minimize cost

Low-Power Design Principles



- Power5 (server)
  - 120W@1900MHz
  - Baseline
- Intel Core2 sc (laptop) :
  - 15W@1000MHz
  - *4x more FLOPs/watt than baseline*
- Intel Atom (handhelds)
  - 0.625W@800MHz
  - 80x more
- Tensilica XTensa DP (Moto Razor) :
  - 0.09W@600MHz
  - 400x more *(80x-120x sustained)*

Low Power Design Principles

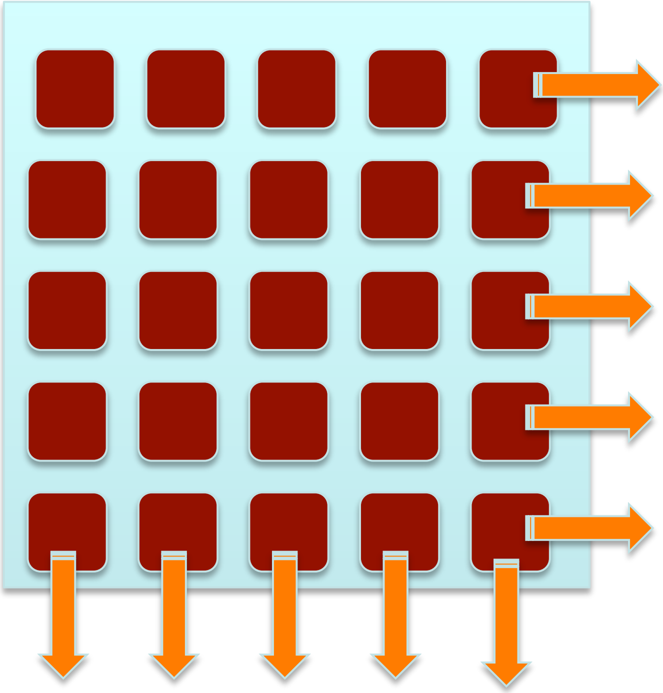


- Power5 (server)
  - 120W@1900MHz
  - Baseline
- Intel Core2 sc (laptop) :
  - 15W@1000MHz
  - *4x more FLOPs/watt than baseline*
- Intel Atom (handhelds)
  - 0.625W@800MHz
  - 80x more
- Tensilica XTensa DP (Moto Razor) :
  - 0.09W@600MHz
  - 400x more (80x-100x sustained)

Even if each simple core is 1/4th as computationally efficient as complex core, you can fit hundreds of them on a single chip and still be 100x more power efficient.

Future of On-Chip Architecture

(DOE Exascale Hardware Report, Nov 2009)



Scale-out for Planar geometry

- **~1000-10k simple cores /Chip**
  - 4-8 wide SIMD or VLIW bundles
  - Either 4 or 50+ HW threads

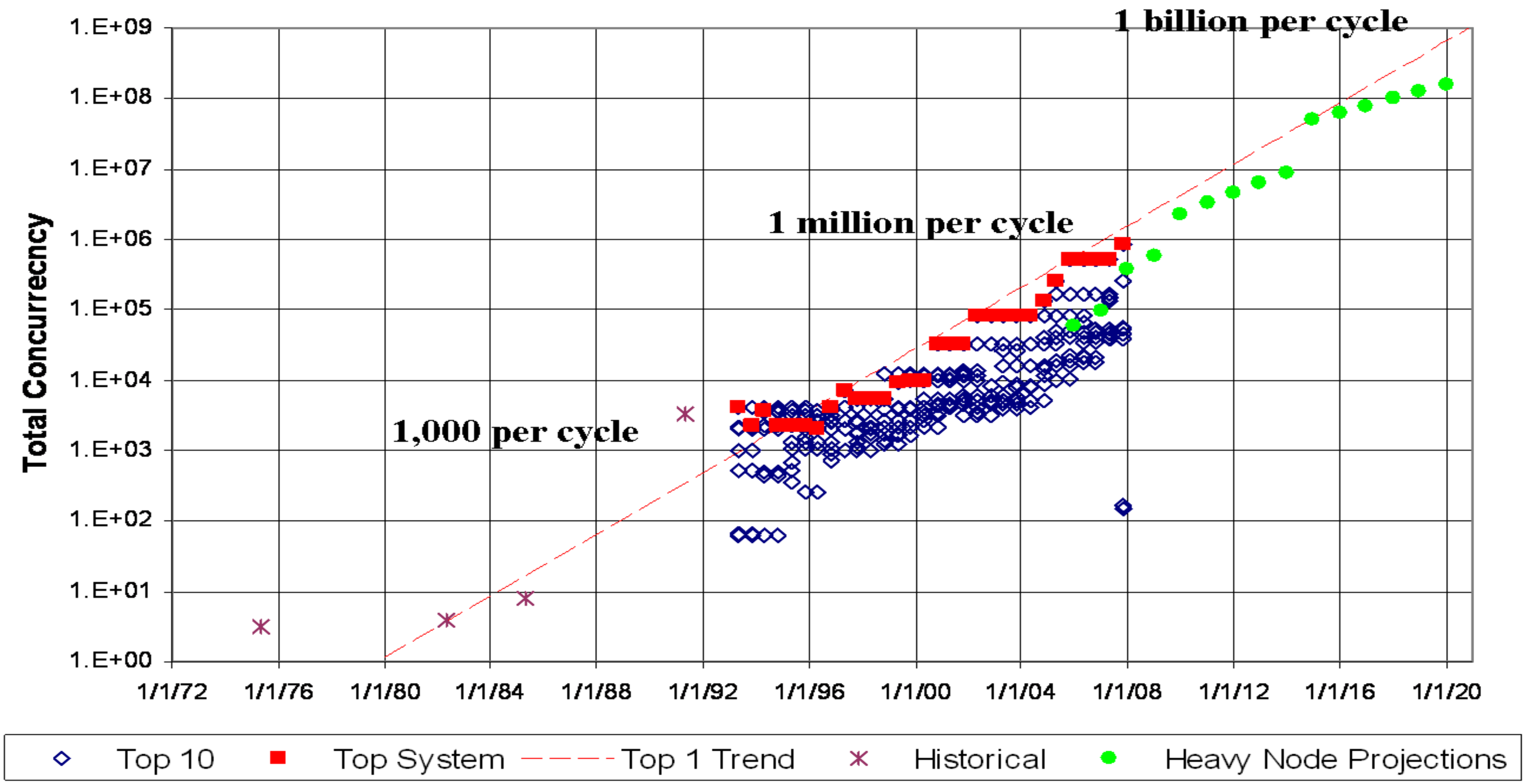
- **On-chip communication Fabric**
  - Low-degree topology for on-chip communication (torus or mesh)
  - *Scale cache coherence?*
  - Global (nonCC memory)
  - Shared register file (clusters)

- **Off-chip communication fabric**
  - Integrated directly on an SoC
  - Reduced component counts
  - Coherent with TLB (no pinning)

System on Chip (SoC) integration
Moving the NIC on Chip

- **Moore's Law continues** *(but what should we do with those transistors?)*
  - Could use it to cram more cores on chip, Or more cache
  - Or integrate other components (SoC) such as NIC
  - PCIe is wasted in cloud where nodes connected to ethernet fabric +disk in most cases (move features on chip to reduce cost)
- **Cloud and Consumer market drivers for SoC Integration**
  - Already see PCIe and 10GigE has moved on chip in commodity space (10G on BG/P, Niagara, and latest Intel Sandybridge. 100GigE by 2018??)
  - Vendors will ask you "which NIC" should we put on board?
    - cloud is pushing for ethernet (standards based interconnect)
  - *At high-end the "custom interconnect" is the "converged fabric" (e.g. Power7) with re-provisioning of pins for PCIe/Ethernet*
- ***What would you do with 100Gig NIC on each chip?***
  - *Coordinated data transfers from each node?*
  - *Is the "network the computer" or the "computer is the network?"*

Projected Parallelism for Exascale



How much parallelism must be handled by the program?

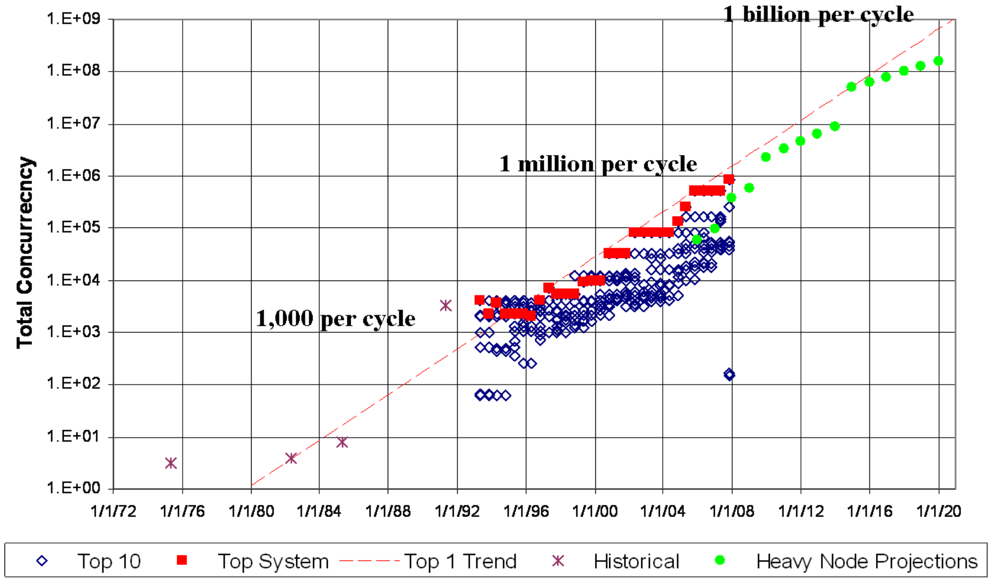
From Peter Kogge (on behalf of Exascale Working Group), “Architectural *Challenges* at the Exascale Frontier”, June 20, 2008

Need 1Million-way parallelism to reach an Exaflop . . .

. . . And possibly another 1000x just to hide latency

Conclusion: Solving Logic Power Drives Move to Massive Parallelism

- **Future HPC must move to simpler power-efficient core designs**
  - Embedded/consumer electronics technology is central to the future of HPC
  - Convergence inevitable because it optimizes both cost and power efficiency



How much parallelism must be handled by the program?

From Peter Kogge (on behalf of Exascale Working Group), "Architectural *Challenges* at the Exascale Frontier", June 20, 2008

- **Consequence is massive on-chip parallelism**
  - A thousand cores on a chip by 2018
  - 1 Million to 1 Billion-way System Level Parallelism
  - *Must express massive parallelism in algorithms and pmodels*
  - *Must manage massive parallelism in system software*

The *NEW* Scaling Rules

- Old Trend
  - Clock frequency doubles every 18 months
- New Trend
  - Number of cores per chip will double every two years
  - Clock speed will not increase (possibly decrease)
- Net Result: Need to deal with systems with millions of concurrent threads
  - *No silver bullet: GPUs and FPGAs also require you to express fine-grained parallelism*
  - *GPU's require thousands of threads per chip*

- *This is a global problem for the computing industry (it affects everything from cell phones to petascale computing systems)*

Technology Paths to Exascale

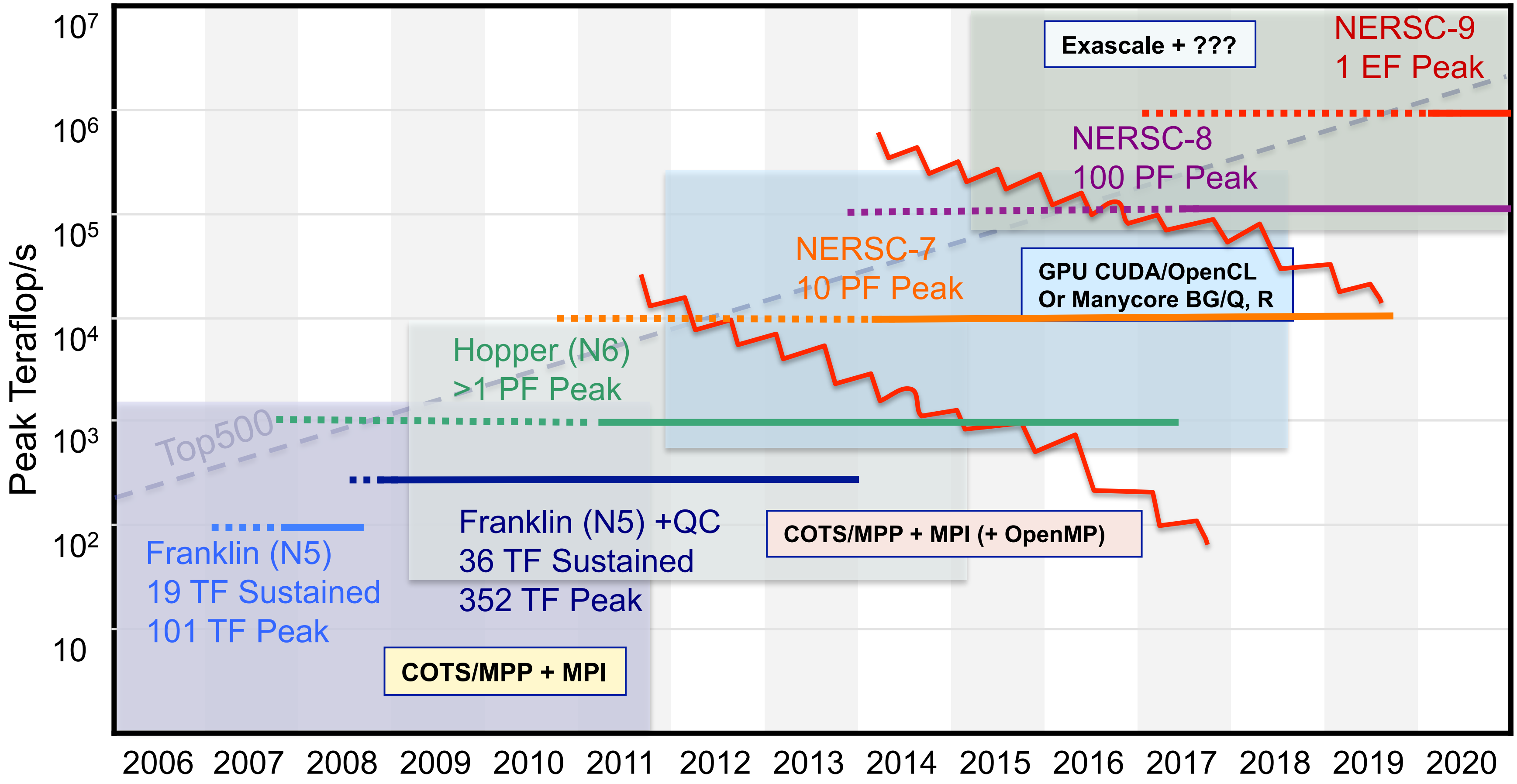
Introducing the “swim lanes”

Technology Paths to Exascale

- **Leading Technology Paths *(Swim Lanes)***
  - **Multicore:** *Maintain complex cores, and replicate (x86 and Power7)*
  - **Manycore/Embedded:** *Use many simpler, low power cores from embedded (BlueGene)*
  - **GPU/Accelerator***: Use highly specialized processors from gaming/graphics market space (NVidia Fermi, Cell)*

- **Risks in Swim Lane selection**
  - **Select too soon:** *Applications cannot follow*
  - **Select too late***: Fall behind performance curve*
  - **Select incorrectly:** *Subject application writers to multiple disruptive technology changes*

Navigating Technology Phase Transitions

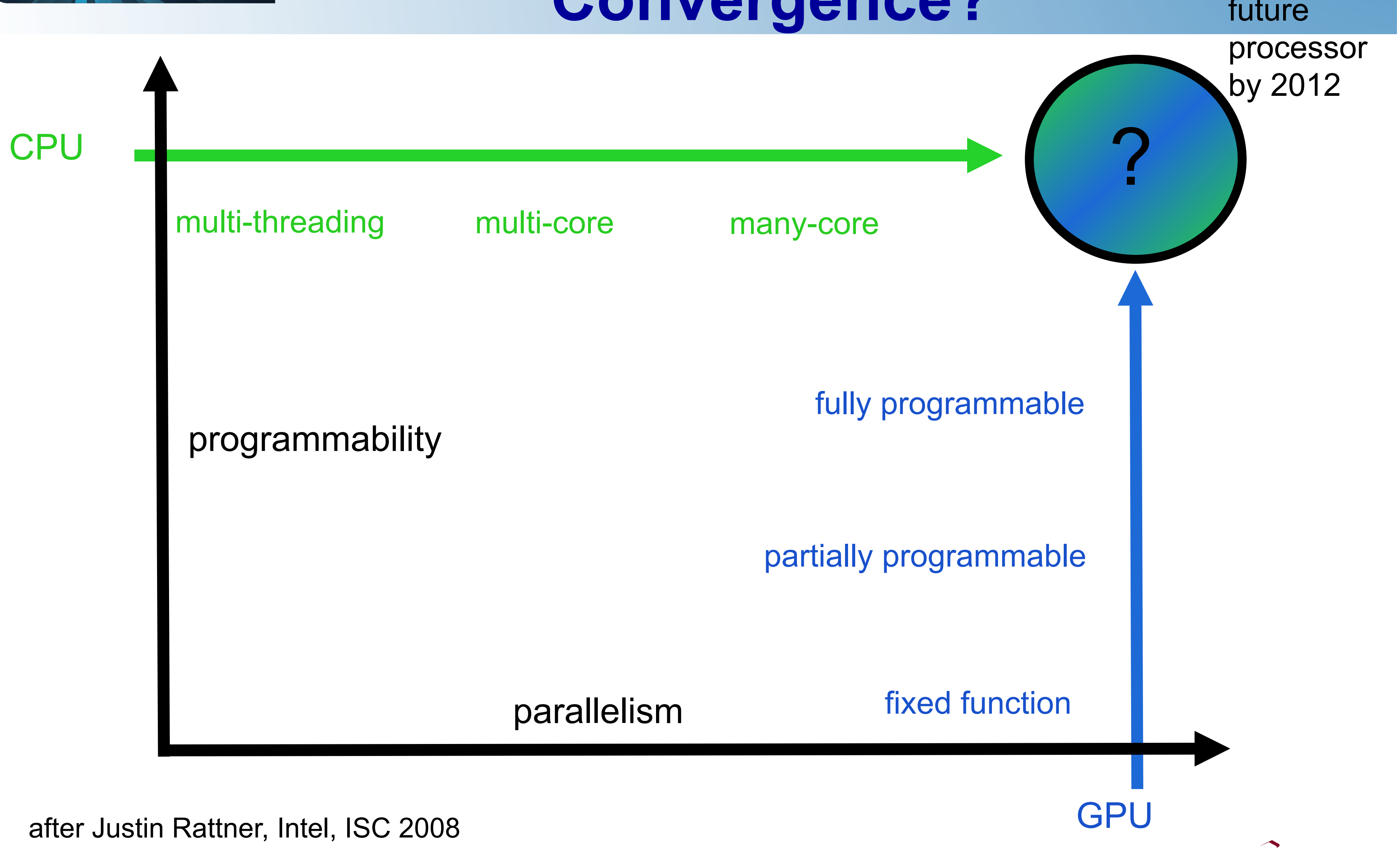


The Challenge of Heterogeneity

GPU/CPU Convergence

Or not…

A Likely Trajectory - Collision or Convergence?



after Justin Rattner, Intel, ISC 2008

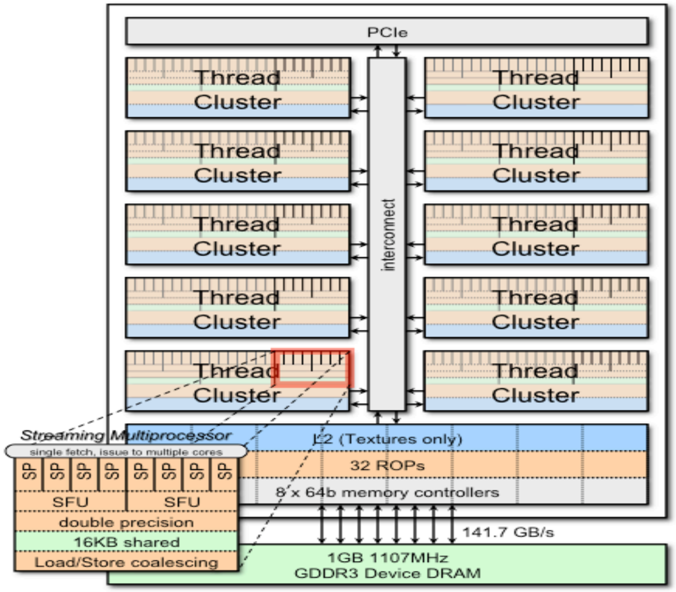
Impediments to CPU/GPU Convergence

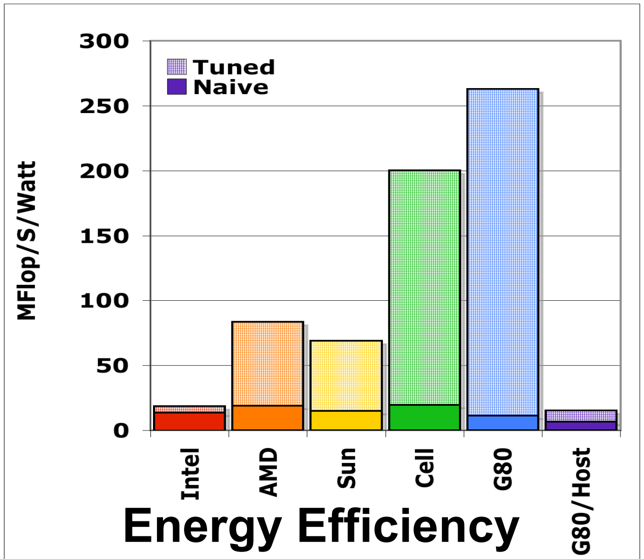
(areas where market forces do not favor architectural convergence)

- **Register files**
  - GPU: Big register files (2k+) subdivided among threads for local data
  - Embedded: small private register files
- **Design/Implementation**
  - GPU: big monolithic proprietary design ($400M multi-billion gates)
  - Embedded: Tiled design using commodity IP
- **Memory Consistency/Communication Models**
  - GPU: Streaming model (very specific to GPU)
  - Embedded: Rich variety of fine-grained inter-processor communication and sync primitives
- **Global Address Space**
  - GPU: No
  - Embedded: Yes
- **Interconnect**
  - GPU: Depend on host for wide-area communication. What market force favors integrated interconnect (IB or IP?)
  - Embedded: SoC design with integrated interconnect
- **Latency Hiding Method**
  - GPU: use many explicit hardware thread contexts (64+)
  - Embedded: Use software managed memory and DMA

Fermi GPU: NVidia’s Bet on HPC Market

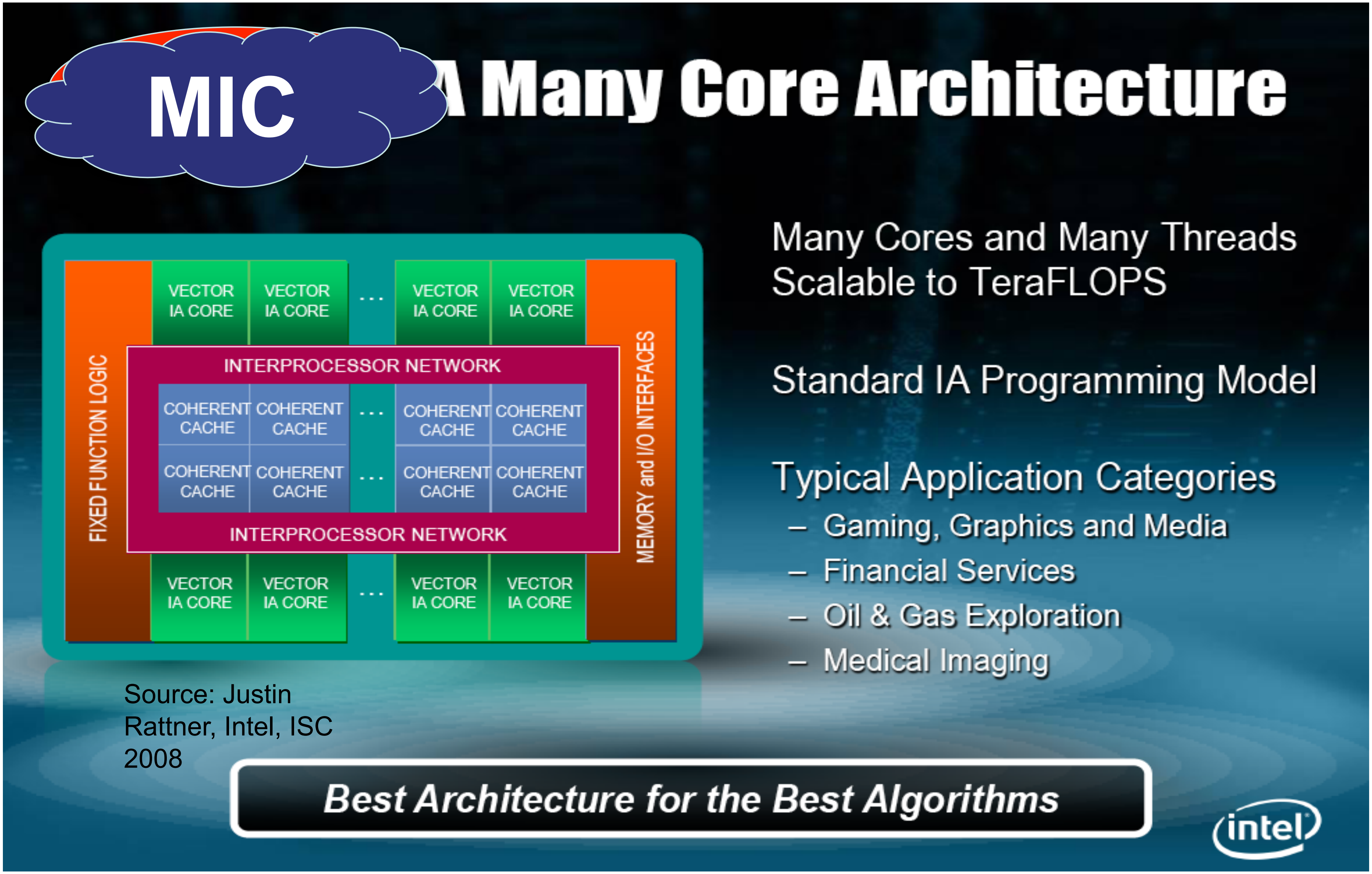
- **Fermi Features**
  - ECC protected memory (older GPUs could not even detect errors)
  - Automatically managed cache to capture temporal recurrences
  - More flexible work scheduling (coarse-grained dataflow)
  - GPU can address host memory
- **Issues**
  - Still big cost to PCIe crossing (operand references bad match for PCIe protocol)
  - Not much change in semantics of restricted GPU/CUDA programming model
  - Not as competitive for high-end graphics as less-converged solution





Energy Efficiency
(Autotuned Stencil Kernel)

Intel's Next Big Bet in Multicore



MIC

A Many Core Architecture

FIXED FUNCTION LOGIC

VECTOR IA CORE | VECTOR IA CORE | . . . | VECTOR IA CORE | VECTOR IA CORE

INTERPROCESSOR NETWORK

COHERENT CACHE | COHERENT CACHE | . . . | COHERENT CACHE | COHERENT CACHE

COHERENT CACHE | COHERENT CACHE | . . . | COHERENT CACHE | COHERENT CACHE

INTERPROCESSOR NETWORK

VECTOR IA CORE | VECTOR IA CORE | . . . | VECTOR IA CORE | VECTOR IA CORE

MEMORY and I/O INTERFACES

Many Cores and Many Threads Scalable to TeraFLOPS

Standard IA Programming Model

Typical Application Categories

- Gaming, Graphics and Media
- Financial Services
- Oil & Gas Exploration
- Medical Imaging

Source: Justin Rattner, Intel, ISC 2008

Best Architecture for the Best Algorithms

Current Status of Convergence

- **Intel Larrabee touted as harbinger for convergence of manycore and GPU**
  - Delayed because underperforms for graphics
  - GPU features put it over power budget
  - Would have been on wrong side of PCIe bus
- **NVidia Fermi also pushing to be more general purpose**
  - Added automatically managed caches and ECC memory
  - Automatically managed caches created huge headache for programming model & memory consistency model
  - Still CUDA programming *(no substantial change in semantics)*
  - Still on wrong side of PCIe (could change if market lets them)
  - Not as competitive for graphics as less “converged” ATI architecture
- **OpenCL as solution to converged CPU/GPU programming model**
  - Broad adoption (ATI, NVidia, and Intel solutions available + Apple/Microsoft support)
  - Extremely verbose compared to CUDA due to API-focused implementation
  - Very low-level exposure to hardware with explicit 4-level memory hierarchy
  - Syntactic compatibility between GPU and CPU, but **no** performance portability
- **Conclusions: there is back-pressure against convergence**
  - Language options are still not performance portable *(key requirement)*
  - Loss of specialization for GPUs hurts competitiveness
  - Graphics specialization for manycore costs power and GDDR memory (keeps on wrong side of PCIe)

Fault Resilience

Chip with FIT rate 1000 fails once every 16 years

A room full of them will fail every few minutes

Fault Tolerance/Resilience

- **Hard Errors: *proportional to component count***
  - Spare cores in design (Cisco Metro)
  - SoC design (fewer components and fewer sockets)
  - Use solder (not sockets)
  - Fewer sockets (pushes us to 10TF chip to keep # sockets const.)
- **Soft Errors: *cosmic rays randomly flip bits***
  - Simpler low-power cores expose less surface area
  - ECC for memory and caches
  - On-board NVRAM controller for localized checkpoint
  - Checkpoint to neighbor for rollback (LLNL SCR)
- **Silent errors: Sometimes RAID & ECC are not enough**
  - End-to-End protection schemes (ZFS)
  - Byzantine Fault Tolerance (BFT)

Industry Trends in Fault Resilience

- Industry must maintain constant FIT rate per node
  - ~1000 failures in time
- Moore's law gets us 100x improvement
  - But still have to increase node count by 10x
- So we will own 10x worse FIT rate
  - MTTI 1week to 1 day
  - MTTI 1 day to 1 hour

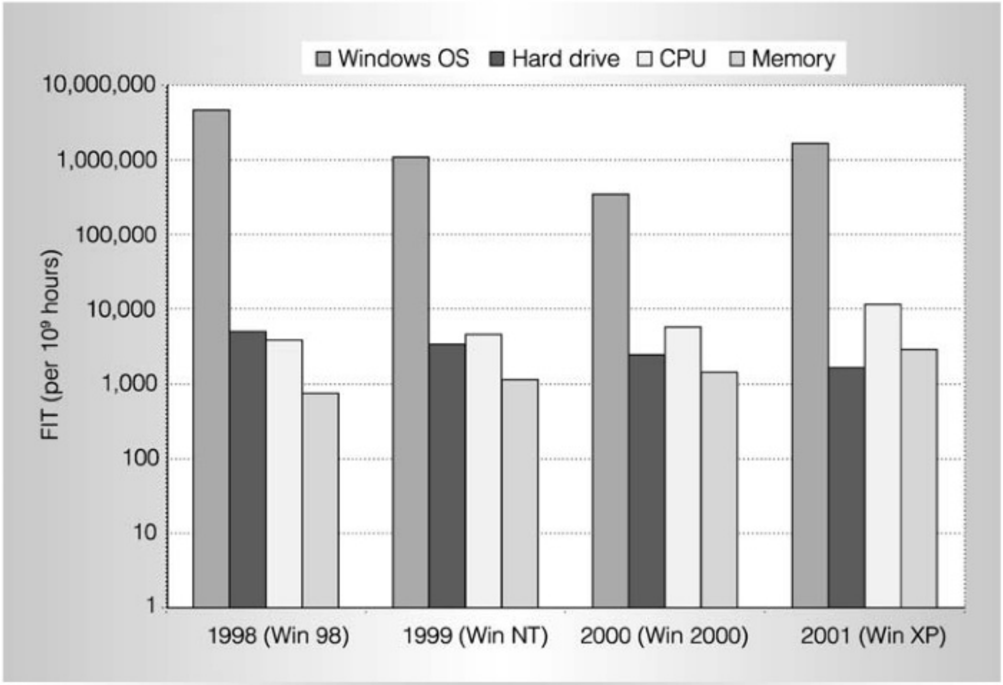


Figure 2. Failures in billions of hours of operation.[2–5]

- Localized checkpointing
  - LLNL SCR to node-local NVRAM

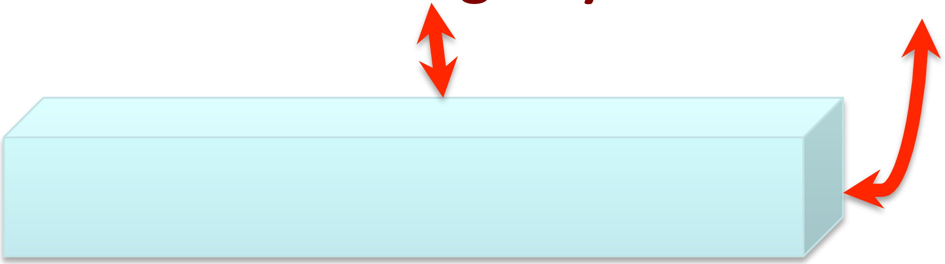
The Cost of Data Movement

How do those cores talk to each other?

The problem with Wires:

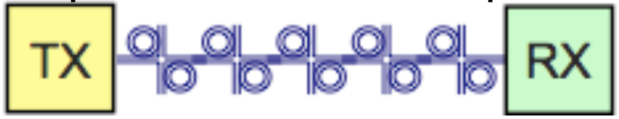
Energy to move data proportional to distance

- Cost to move a bit on copper wire:
  - **Power = bitrate * Length^2 / cross-section-area**

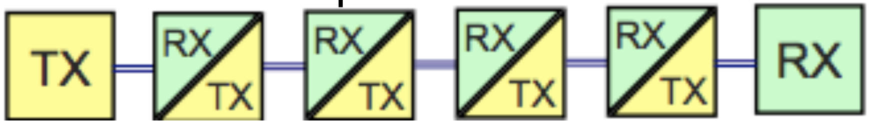


- Wire data capacity constant as feature size shrinks
- *Cost to move bit proportional to distance*
- *~1TByte/sec max feasible off-chip BW (10GHz/pin)*
- *Photonics reduces distance-dependence of bandwidth*

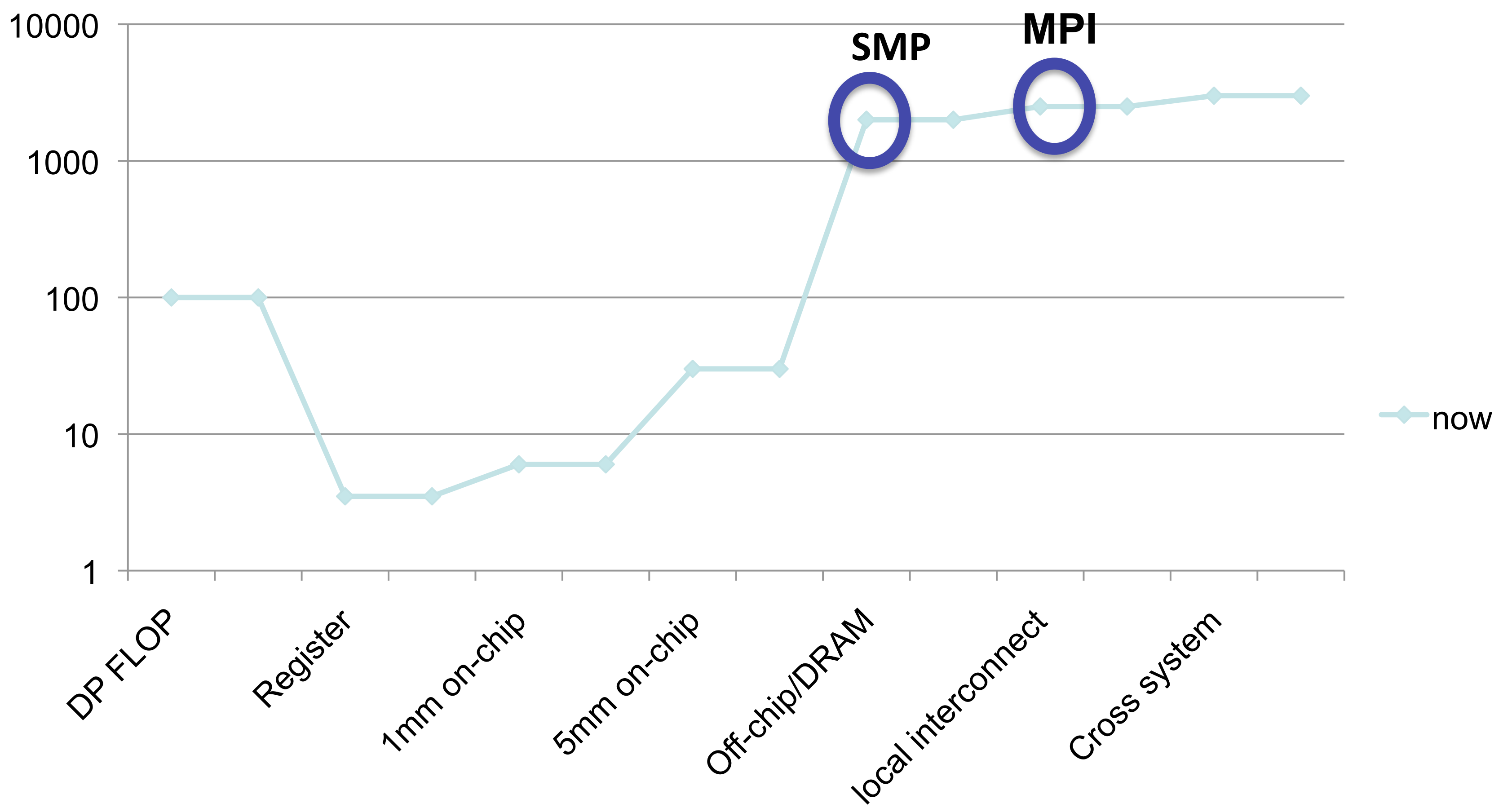
Photonics requires no redrive and passive switch little power



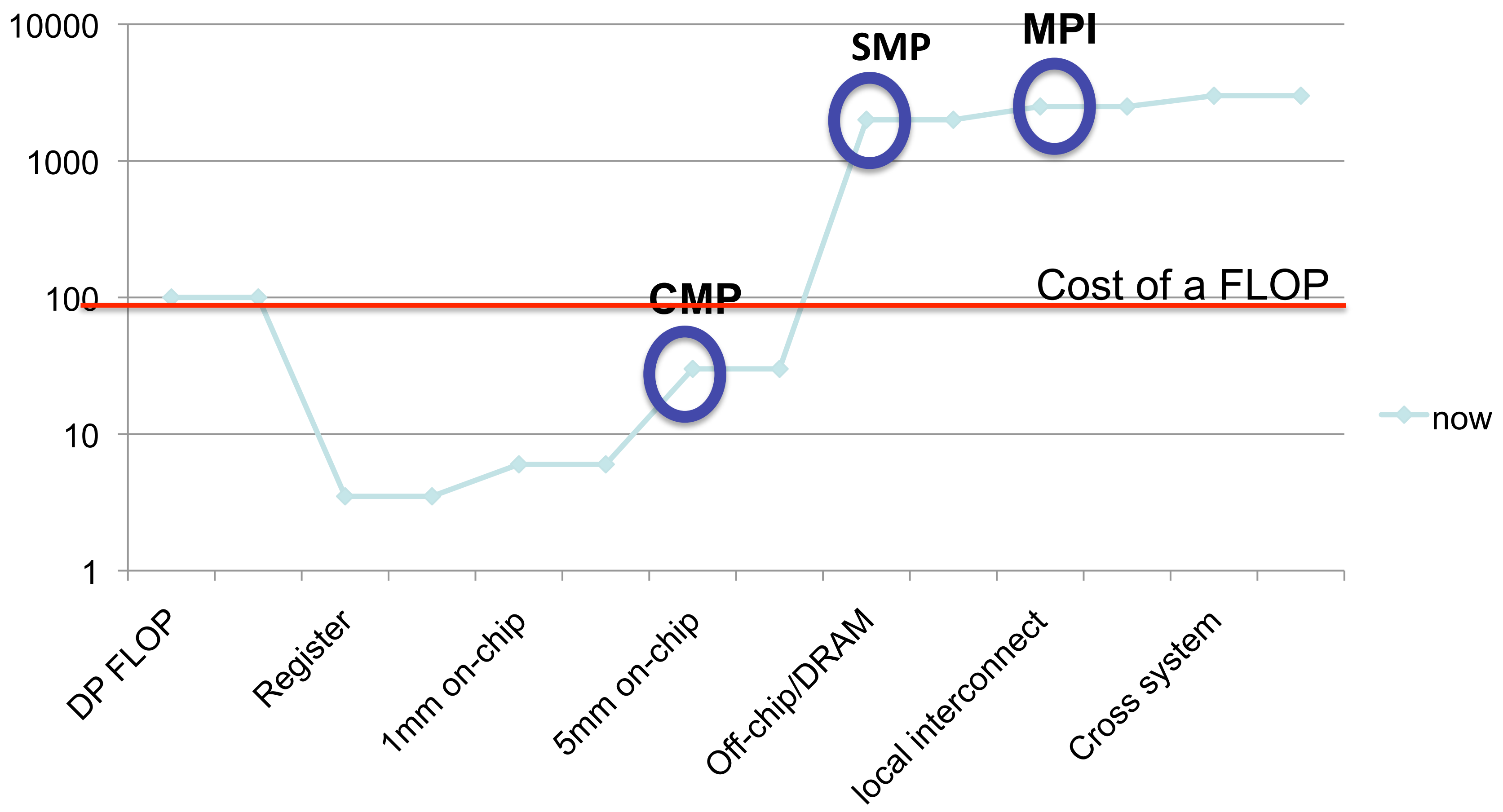
Copper requires to signal amplification even for on-chip connections



The Cost of Data Movement

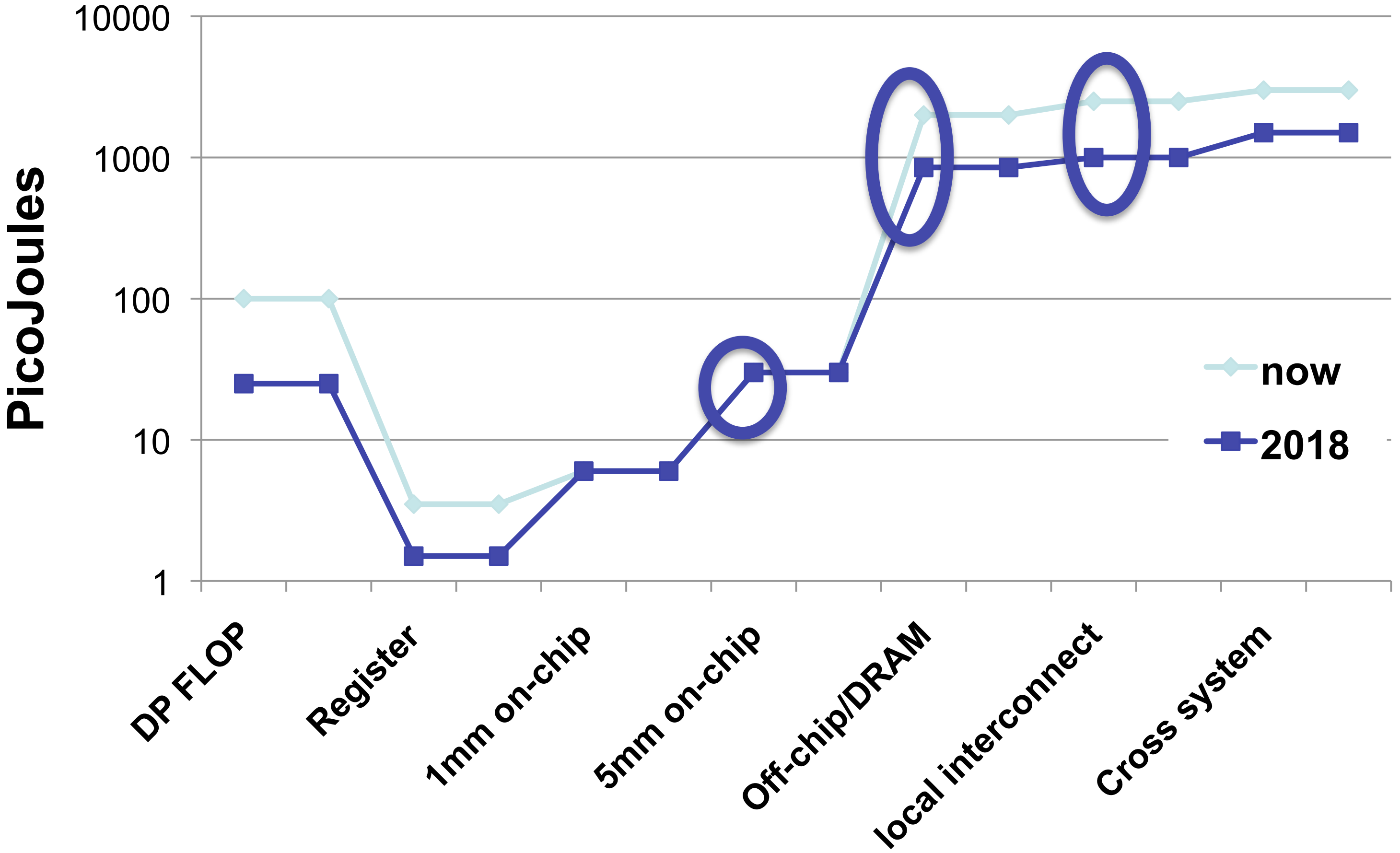


The Cost of Data Movement



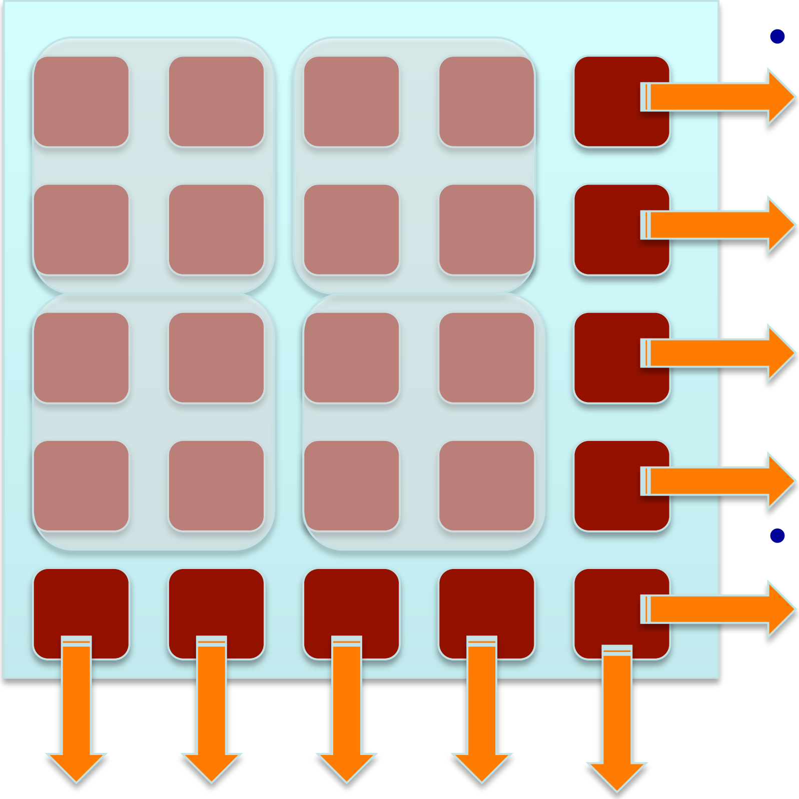
The situation will not improve in 2018

Energy Efficiency will require careful management of data locality



Important to know when you are on-chip and when data is off-chip!

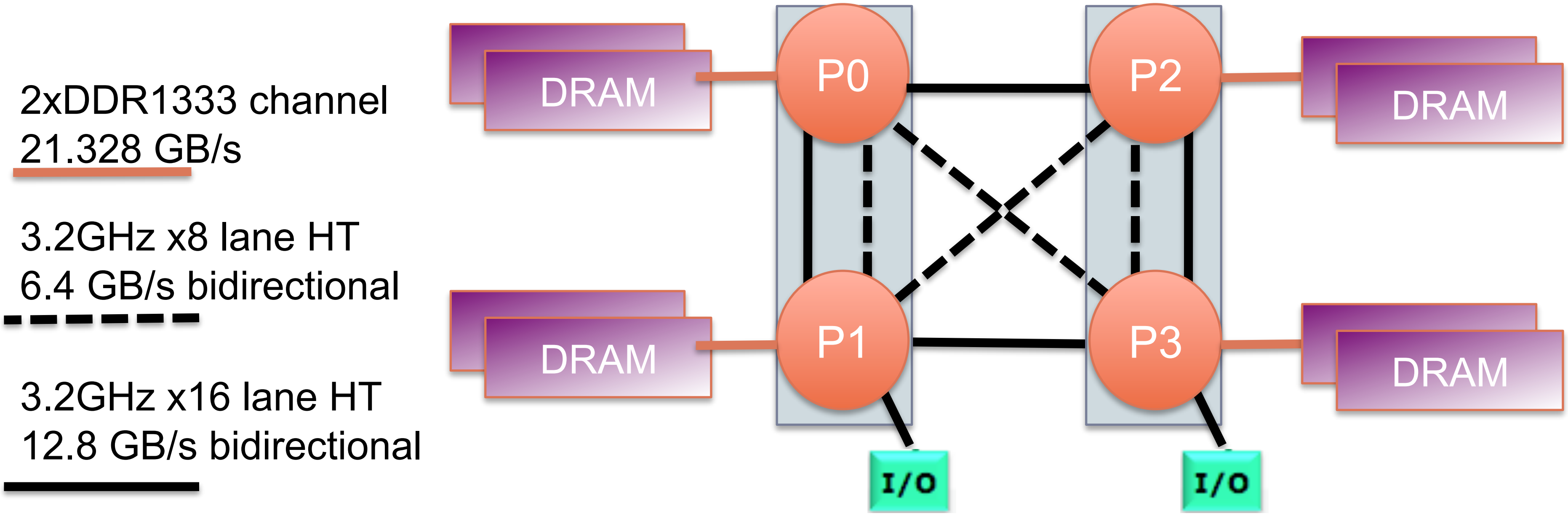
Cost of Data Movement (manifests as NUMA effect)



- Cost of moving long-distances on chip motivates clustering on-chip
  - 1mm costs ~6pj (today & 2018)
  - 20mm costs ~120 pj (today & 2018)
  - FLOP costs ~100pj today
  - FLOP costs ~25pj in 2018
- Different Architectural Directions
  - GPU: WARPs of hardware threads clustered around shared register file
  - CMP: limited area cache-coherence
  - CMT: hardware multithreading clusters

Hopper Node Topology
Understanding NUMA Effects

- **Heterogeneous Memory access between dies**
- **“First touch” assignment of pages to memory.**



2xDDR1333 channel
21.328 GB/s

3.2GHz x8 lane HT
6.4 GB/s bidirectional

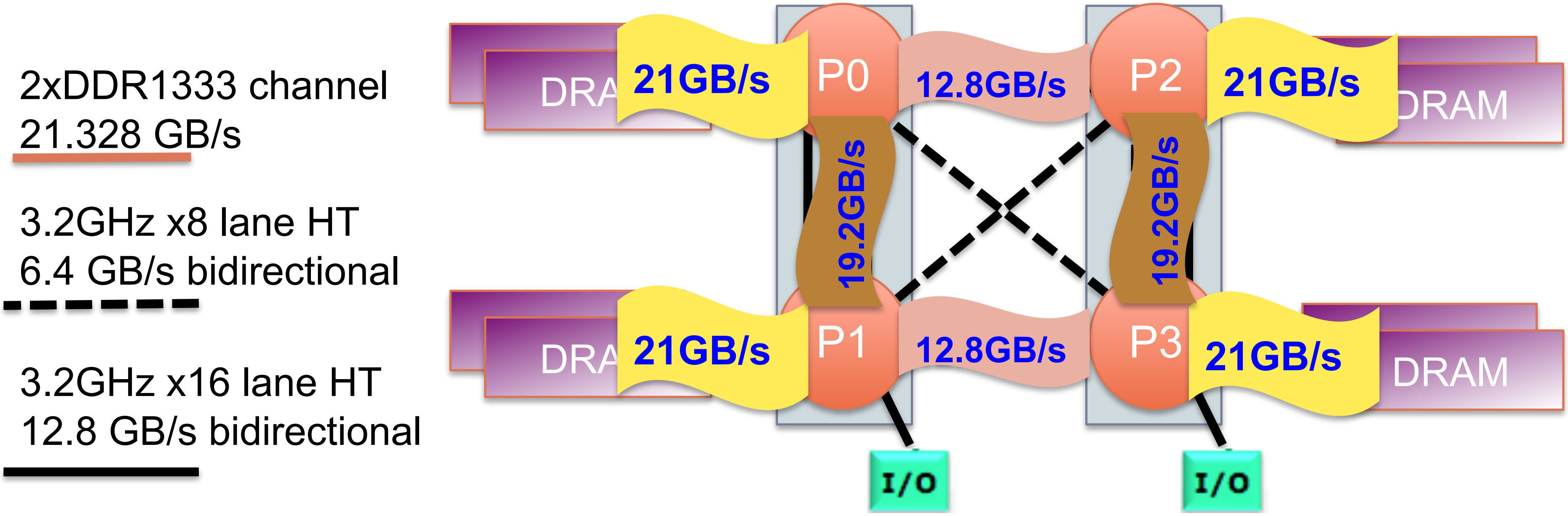
3.2GHz x16 lane HT
12.8 GB/s bidirectional

- **Locality is key** *(just as per Exascale Report)*
- **Only *indirect* locality control with OpenMP**

Hopper Node Topology

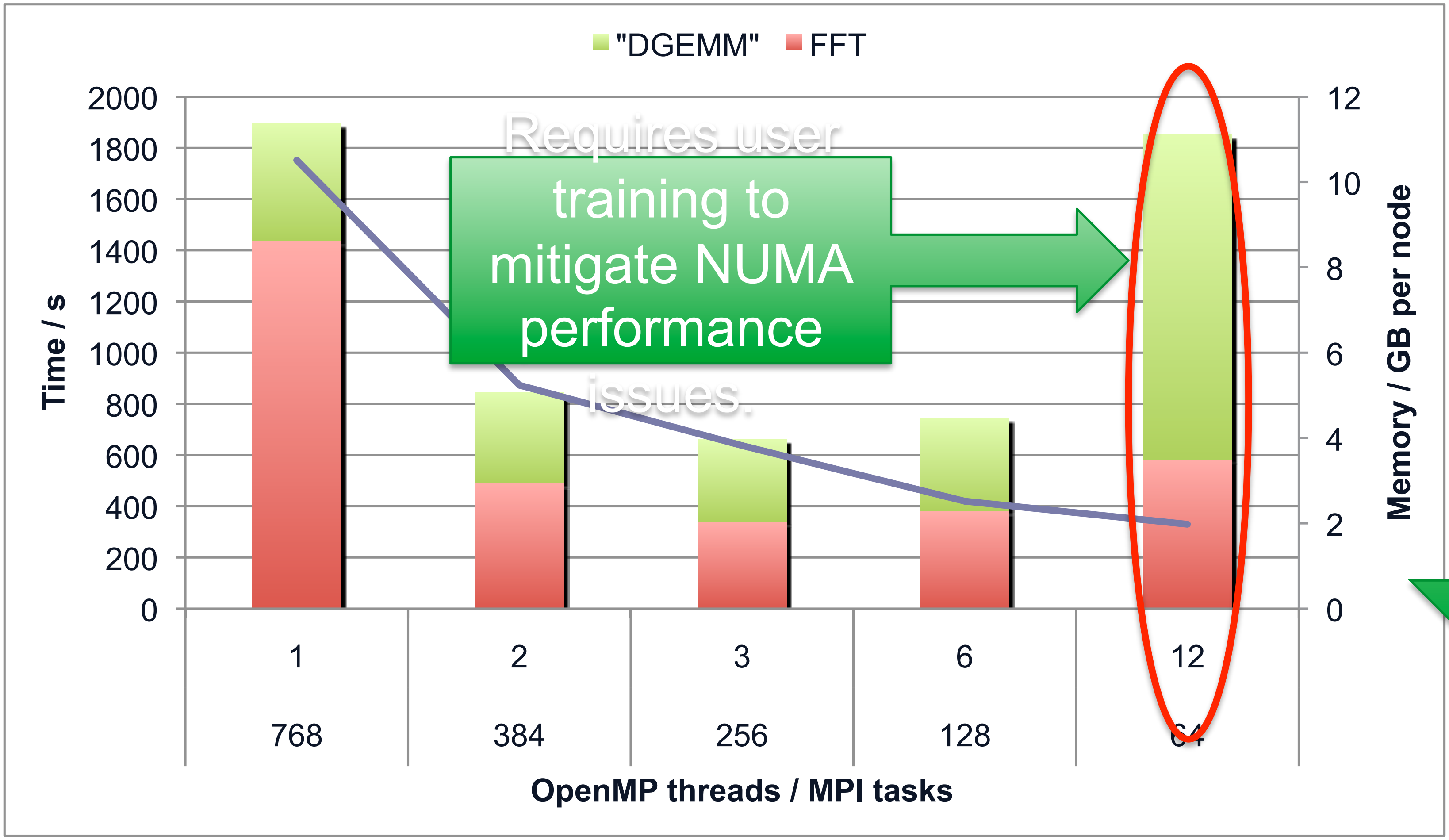
NUMA effects are manifest today

- **Heterogeneous Memory access between dies**
- **“First touch” assignment of pages to memory.**



- **Locality is key** *(just as per Exascale Report)*
- **Only *indirect* locality control with OpenMP**

Example OpenMP for PARATEC

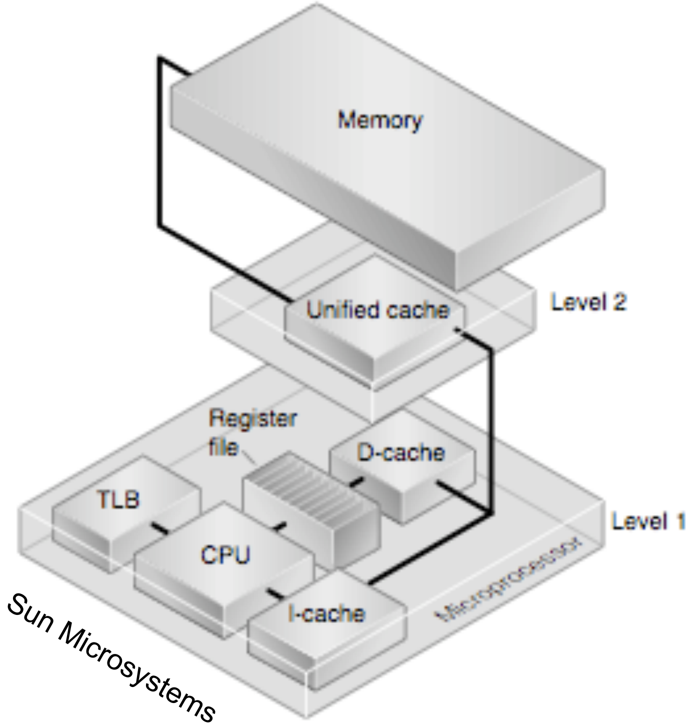


Requires user training to mitigate NUMA performance issues.

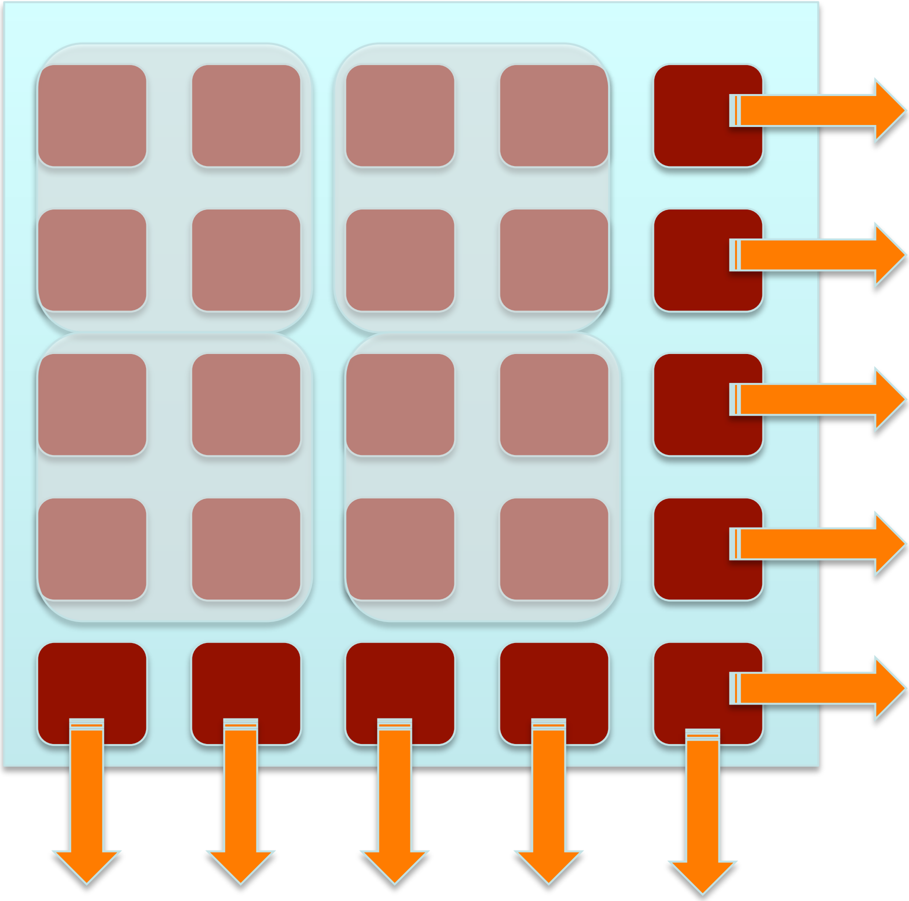
GOOD

Locality Management is Key

Vertical Locality Management



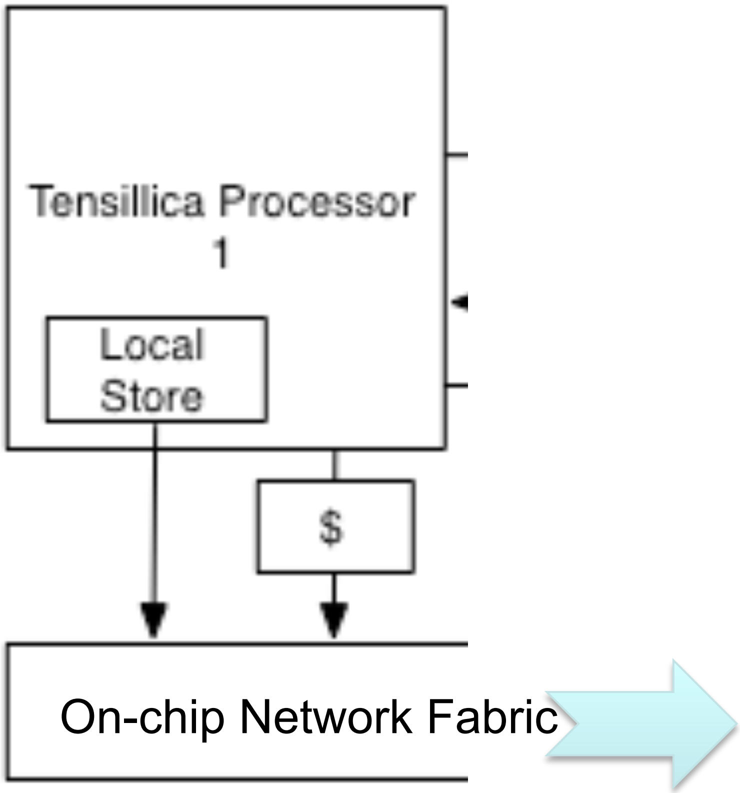
Horizontal Locality Management



Locality Management is Key

Vertical Locality Management

- **Movement of data up and down cache hierarchy**
  - Cache virtualizes notion of on-chip off-chip
  - Software managed memory (local store) is hard to program (cell)
- **Software Managed Memories**
  - Use conventional cache for portability
  - Only use SW managed memory only for performance critical code
  - Repartition as needed



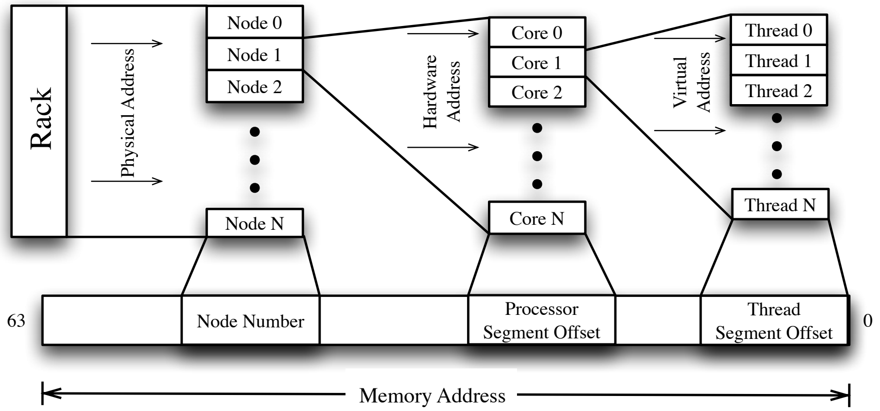
The Problem with Caches

(why local-store will be increasingly prevalent)

- **Automatic cache virtualizes the notion of on-chip vs. off-chip memory**
  - Makes on-chip memory indistinguishable from off-chip to pmodel
  - But energy cost is ~100x is data is off-chip
  - But if you have explicit on-chip memory, then what does that mean for cache-coherence?
- **If you want performance, you really need to know the difference between on & off chip**
  - *You can ignore it and be correct, but penalty is ~100x (reduces urgency from standpoint of applications)*

This is why flat models for parallelism are NOT in the solution space (what about cache-coherence?)

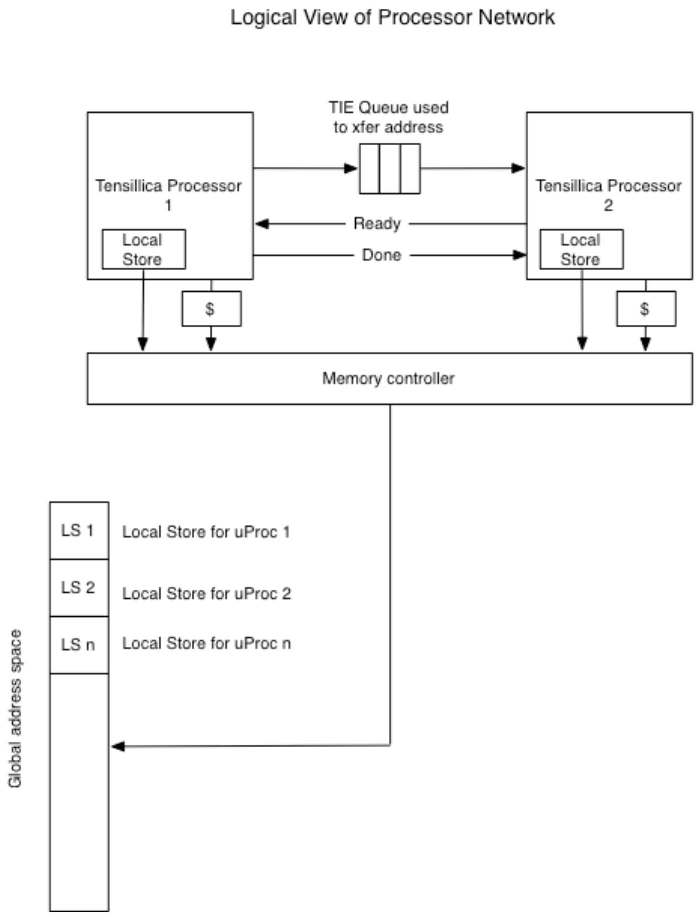
Managing Data Locality



Horizontal Locality Management

- **Movement of data between processors**
  - 10x lower latency and 10x higher bandwidth on-chip
  - Need to minimize distance of horizontal data movement
- **Encode Horizontal locality into memory address**
  - Hardware hierarchy where high-order bits encode cabinet and low-order bits encode chip-level distance

Managing Data Locality



Horizontal Locality Management

- **Movement of data between processors**
  - 10x lower latency and 10x higher bandwidth on-chip
  - Need to minimize distance of horizontal data movement
- **Encode Horizontal locality into memory address**
  - Hardware hierarchy where high-order bits encode cabinet and low-order bits encode chip-level distance
- **Map local-store into global address space**
  - Hierarchical Partitioned Global Address space

Managing Data Locality

Vertical Locality Management

- **Movement of data up and down cache hierarchy**
  - Cache virtualizes notion of on-chip off-chip
  - Software managed memory (local store) is hard to program (cell)
- **Virtual Local store**
  - Use conventional cache for portability
  - Only use SW managed memory only for performance critical code
  - Repartition as needed

Horizontal Locality Management

- **Movement of data between processors**
  - 10x lower latency and 10x higher bandwidth on-chip
  - Need to minimize distance of horizontal data movement
- **Encode Horizontal locality into memory address**
  - Hardware hierarchy where high-order bits encode cabinet and low-order bits encode chip-level distance
- **Map local-store into global address space**
  - Hierarchical Partitioned Global Address space

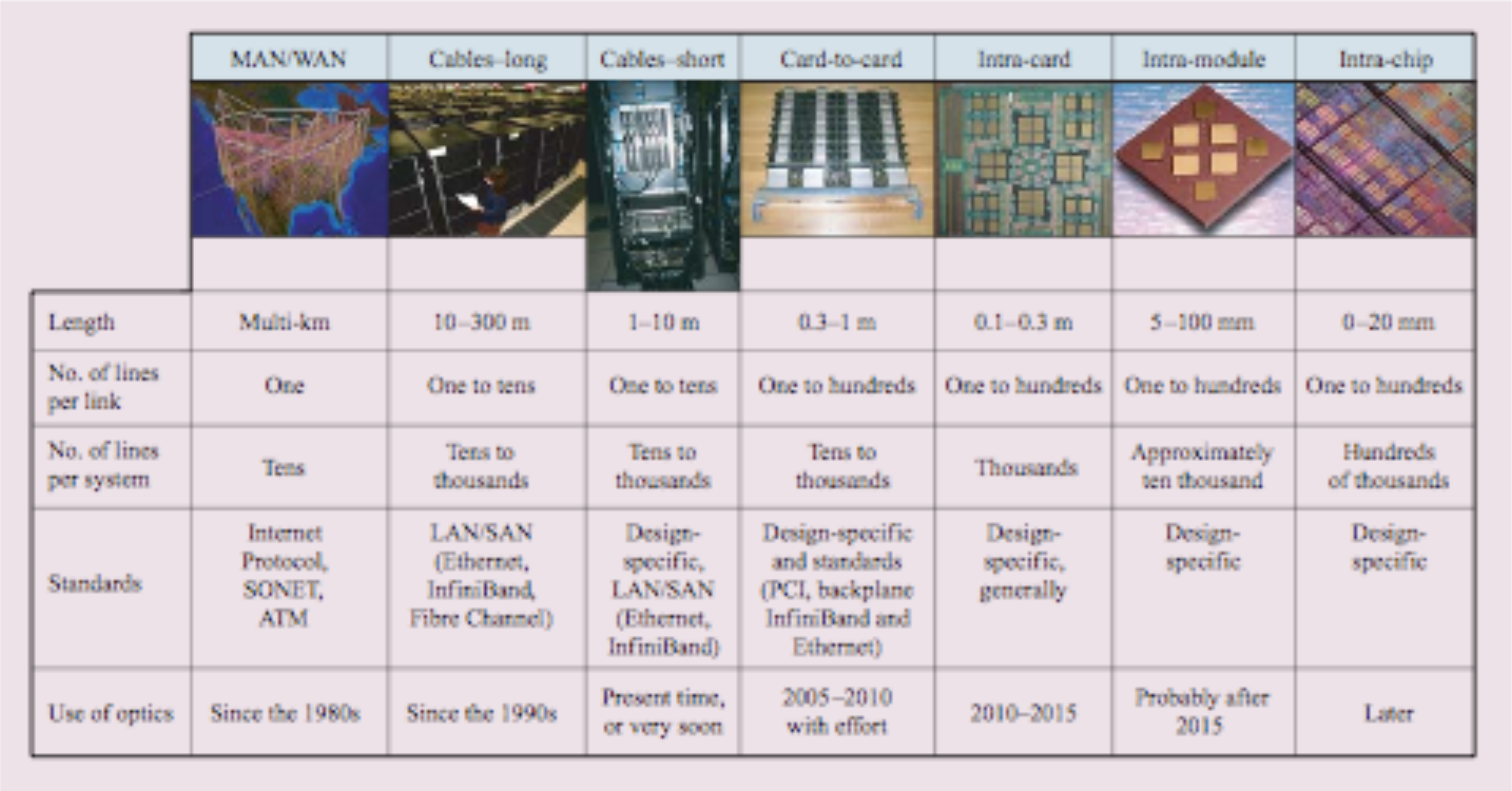
Why use Hierarchical (hybrid) model for parallelism?

- **Our current programming models assume all communicating elements are equidistant (PRAM)**
  - OpenMP, and MPI each assume flat machine at their level of parallelism
- **But the machine is not flat!!!**
  - Lose performance because expectation and reality are mismatched
  - *Pmodel does not match underlying machine model!!*
- **Target: Get Strong scaling on-chip and weak-scaling off-chip**
  - 100x higher bandwidth between cores on chip
  - 100x lower latency between cores on chip
  - If you pretend that every core is a peer (each is just a generic MPI rank) you are leaving a lot of performance on the table
  - You cannot domain-decompose things forever
- **MPI between nodes and *X* within node is short term solution**
  - Where X could be OpenMP, UPC, OpenCL, CUDA, etc…
  - *But a new language and model of computation may be worth considering*

Optics: Potential alternative path

Kash & Benner (2005)

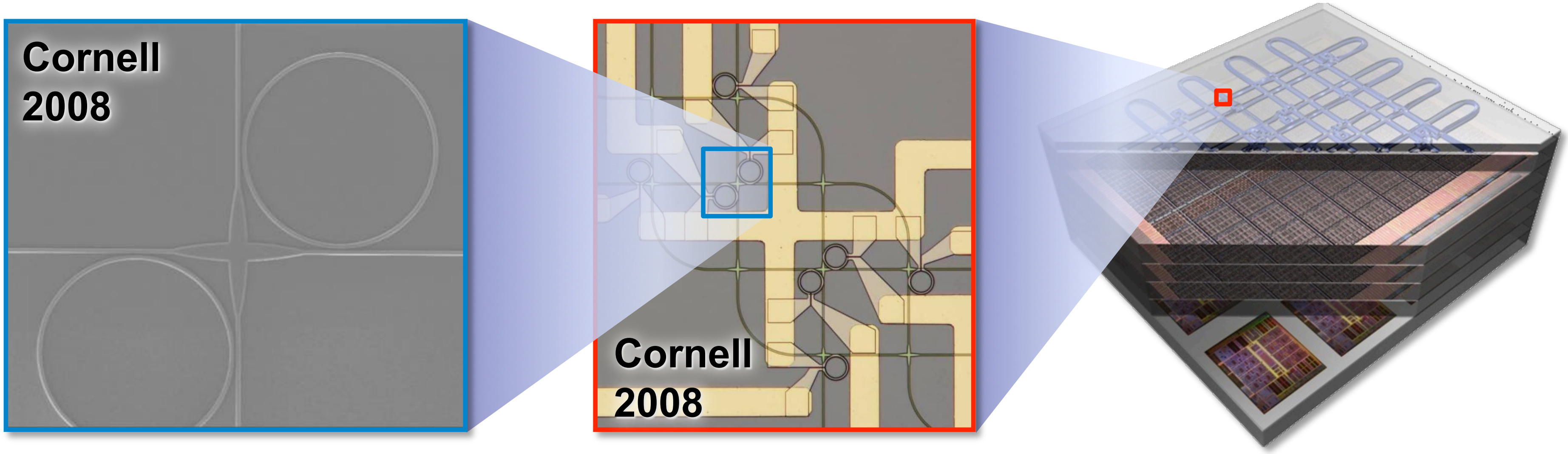
progression towards on-chip optics



	MAN/WAN	Cables–long	Cables–short	Card-to-card	Intra-card	Intra-module	Intra-chip
Length	Multi-km	10–300 m	1–10 m	0.3–1 m	0.1–0.3 m	5–100 mm	0–20 mm
No. of lines per link	One	One to tens	One to tens	One to hundreds	One to hundreds	One to hundreds	One to hundreds
No. of lines per system	Tens	Tens to thousands	Tens to thousands	Tens to thousands	Thousands	Approximately ten thousand	Hundreds of thousands
Standards	Internet Protocol, SONET, ATM	LAN/SAN (Ethernet, InfiniBand, Fibre Channel)	Design-specific, LAN/SAN (Ethernet, InfiniBand)	Design-specific and standards (PCI, backplane InfiniBand and Ethernet)	Design-specific, generally	Design-specific	Design-specific
Use of optics	Since the 1980s	Since the 1990s	Present time, or very soon	2005–2010 with effort	2010–2015	Probably after 2015	Later

Optical Switches, Routers, and Networks-on-Chip

On-chip photonics enable ultrahigh-bandwidth, low-power communications for both on- and off-chip signaling, allowing the maximized performance for chip-scale parallel processing systems



photonic switching device:
2×2 switch composed of 2 waveguides, 2 micro-ring resonators, and a crossing

photonic routing subsystem:
integrated photonic and electronic devices providing multi-wavelength, non-blocking, low-power photonic routing

CMP system vision:
3DI stack with dedicated communications plane (top layer) housing a photonic NoC

With Keren Bergman
Columbia University

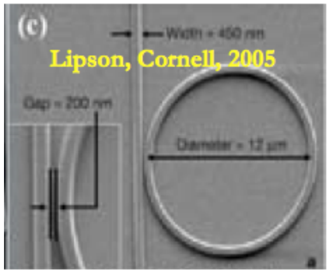
Micro-Ring Switches

- 375 fJ/transition possible
- 400 µW ON-state static power possible
- 250 Gb/s throughput bandwidth demonstrated
- > 1 Tb/s possible

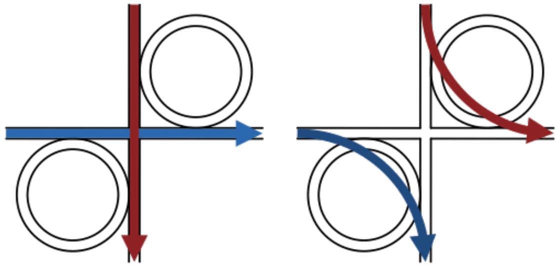
U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

Silicon Photonics for Energy-Efficient Communication

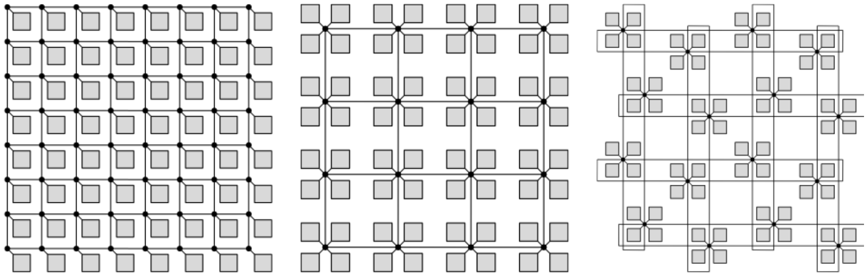


Silicon Photonic Ring Resonator



(a) Off state

(b) On state

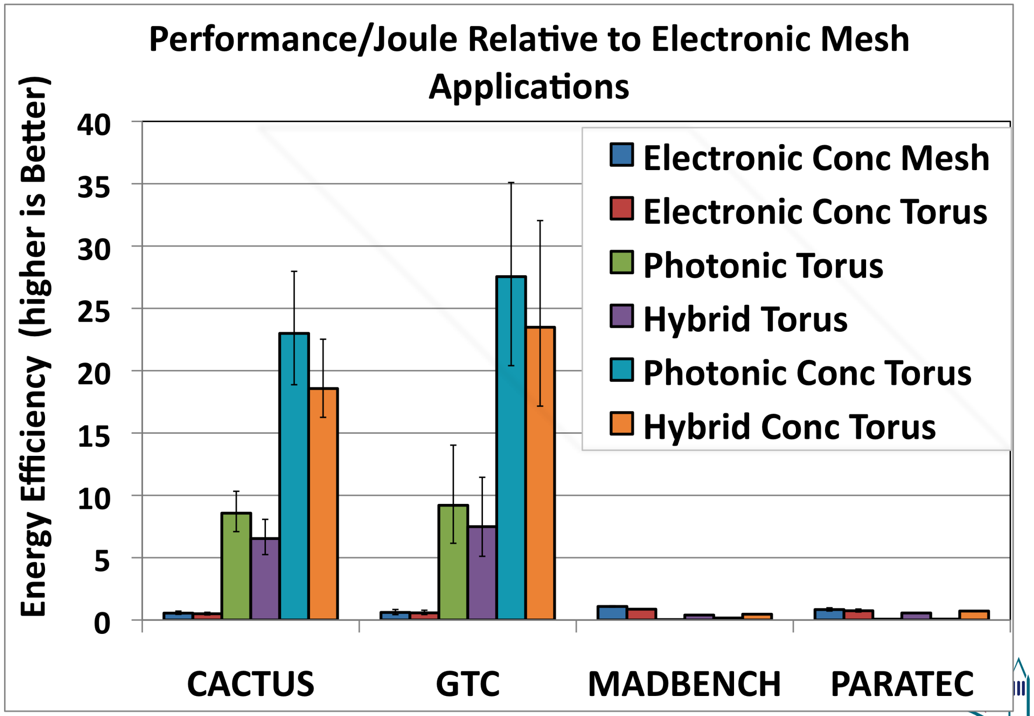


(a) Mesh

(b) Concentrated Mesh

(c) Concentrated Torus

- Silicon photonics enables optics to be integrated with conventional CMOS
- Enables up to 27x improvement in communication energy efficiency!



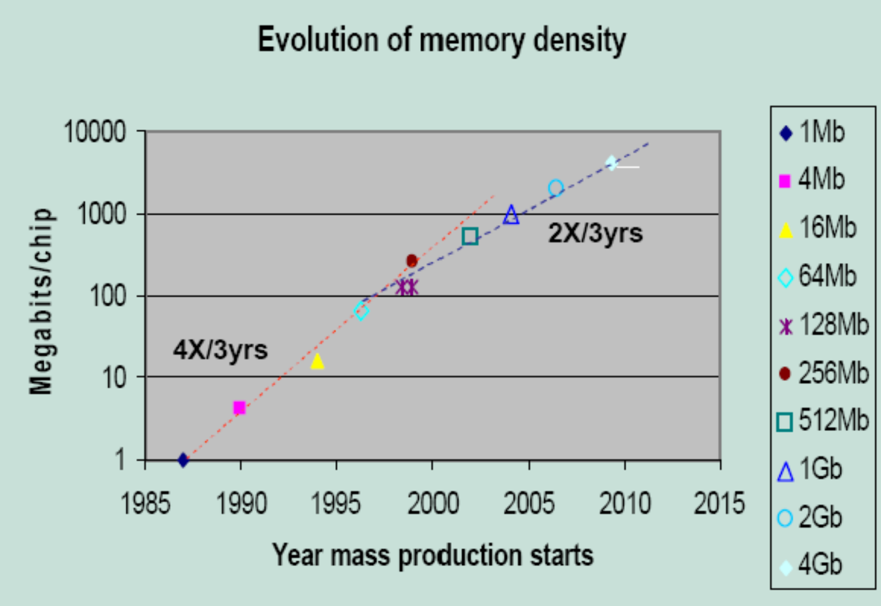
Optics Conclusions

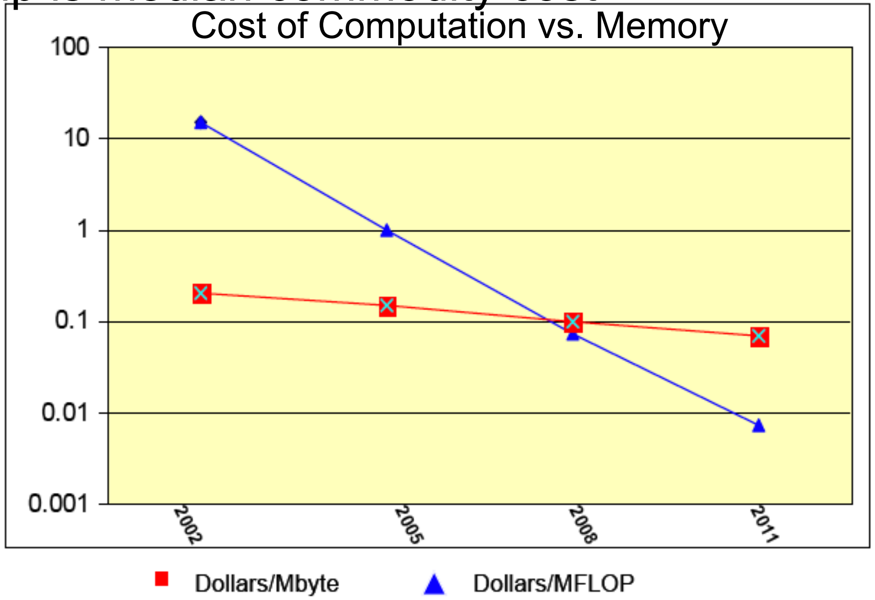
- **Optics could drastically reduce distance dependence of bandwidth**
  - Would have huge effect on programming if we don’t have to worry about bandwidth localization (just latency hiding)
- **But its not a sure bet**
  - Still expensive
  - Nanophotonics not yet mature
  - Mechanical engineering (optics alignment, cabling and connectors) does not benefit from Moore’s Law
- **Hope for the best and plan for the worst**
  - **We hope that with investment, silicon photonics will be ready in time for exascale**

Memory

Projections of Memory Density Improvements

- Memory density is doubling every three years; processor logic is every two
    - Project 8Gigabit DIMMs in 2018
    - 16Gigabit if technology acceleration (or higher cost for early release)
- Storage costs (dollars/Mbyte) are dropping gradually compared to logic costs
    - Industry assumption: $1.80/memory chip is median commodity cost

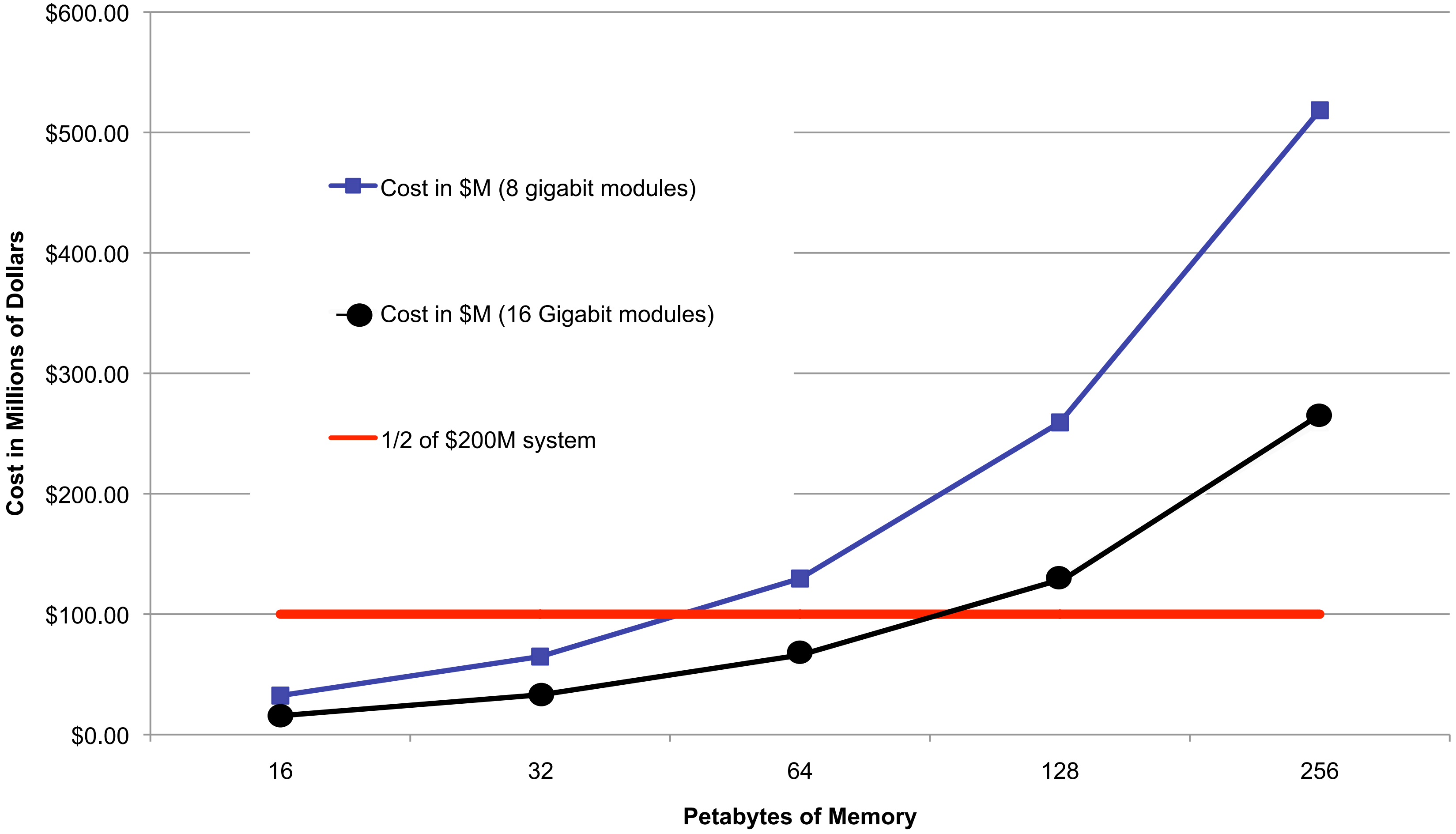




The cost to sense, collect, generate and calculate data is declining much faster than the cost to access, manage and store it

Source: David Turek, IBM

Cost of Memory Capacity
2 different potential Memory Densities



Forces us to strong scaling

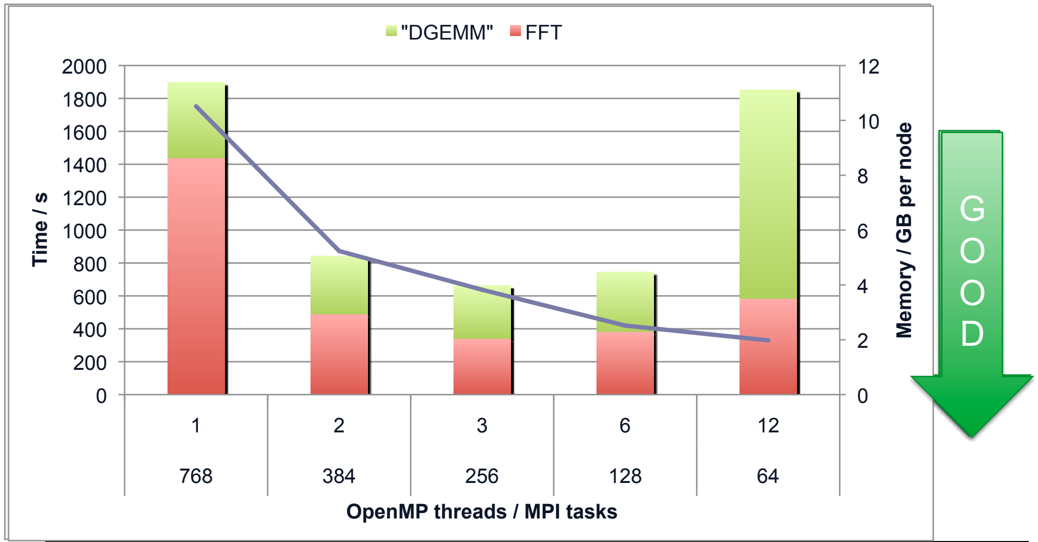
Forces us to memory conservative communication (GAS)

Future of Memory Scaling

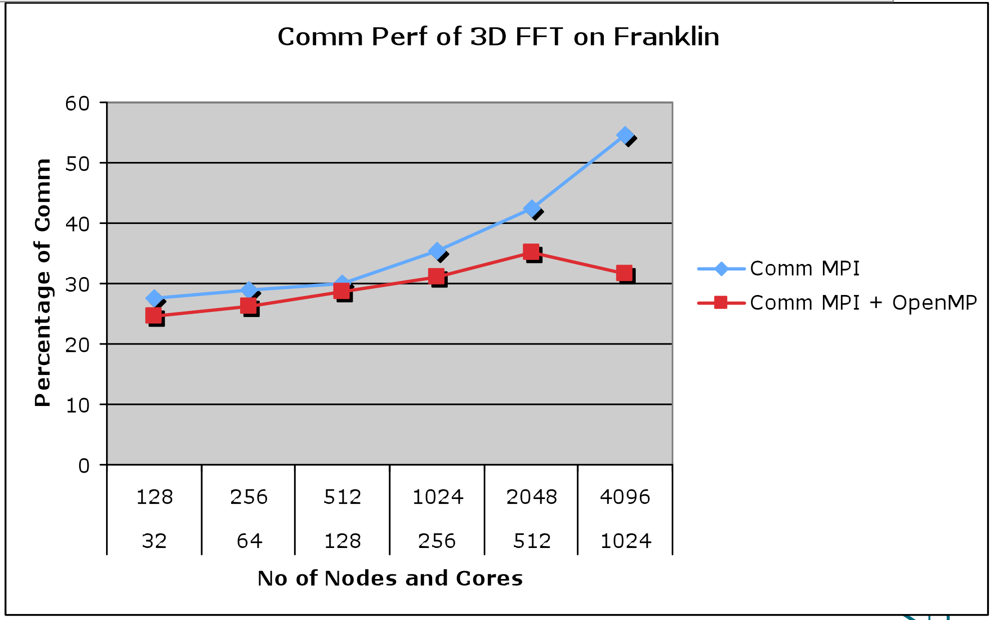
- **Old Trend**
  - Memory increased proportional to CPU performance (more memory per core)
  - Scale-up problem proportional to system parallelism (weak scaling)
- **New Trend**
  - Memory per core will *decrease* (slow increase per node)
  - Can no longer scale problem with increased parallelism
  - Forces to strong-scaling (increase parallelism with fixed problem size)
- **Net Result: Caught between a rock and a hard place**
  - Need strong-scaling to keep runtimes from growing exponentially with increased problem size given fixed clock frequency
  - Even if you don’t care about increased runtime, you have less memory per core (*so you still end up with strong-scaling*)

Example OpenMP for PARATEC

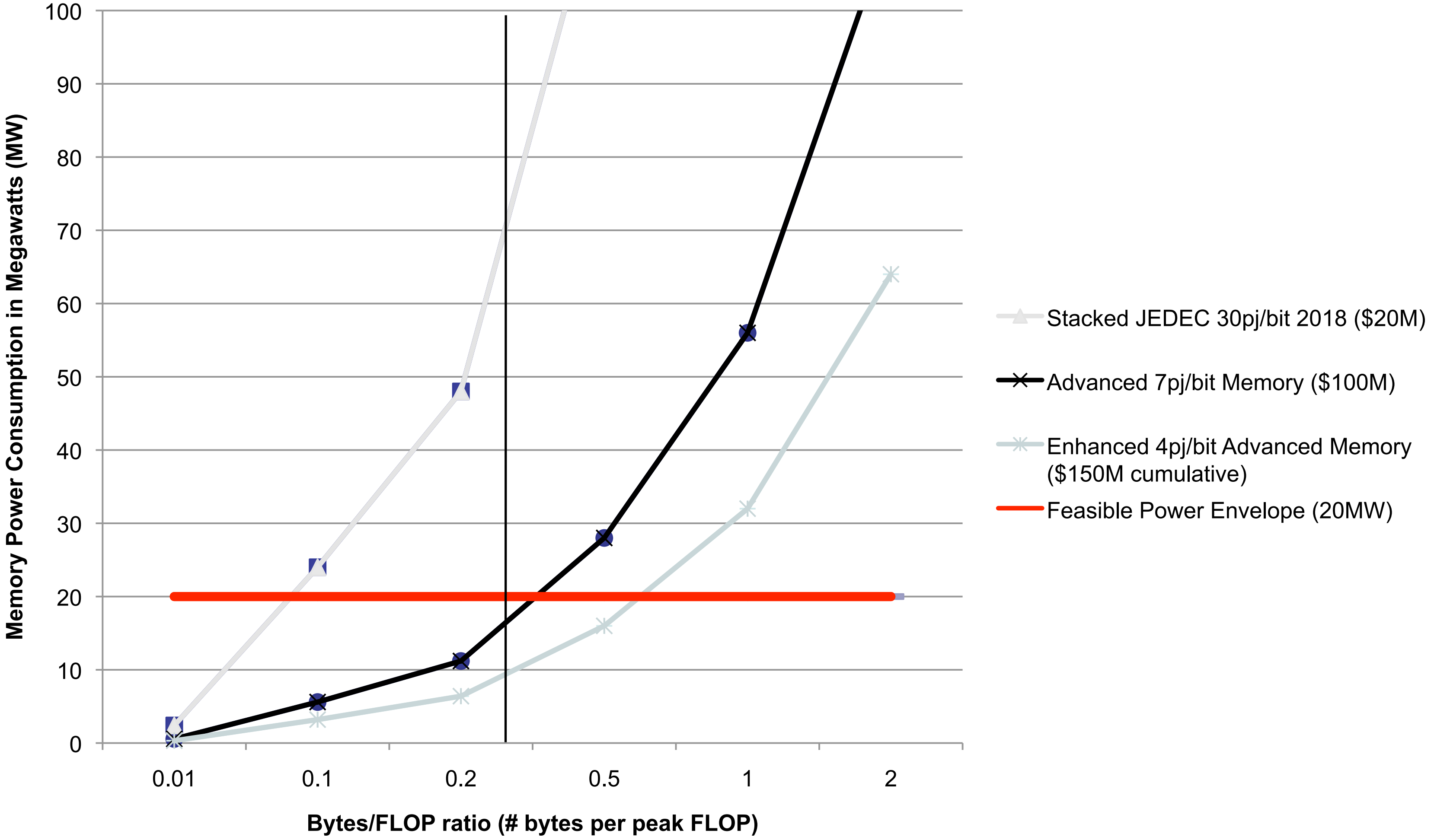
- **MPI+OMP Hybrid**
  - **Reduces memory footprint**
  - **Increases performance up to NUMA-node limit**



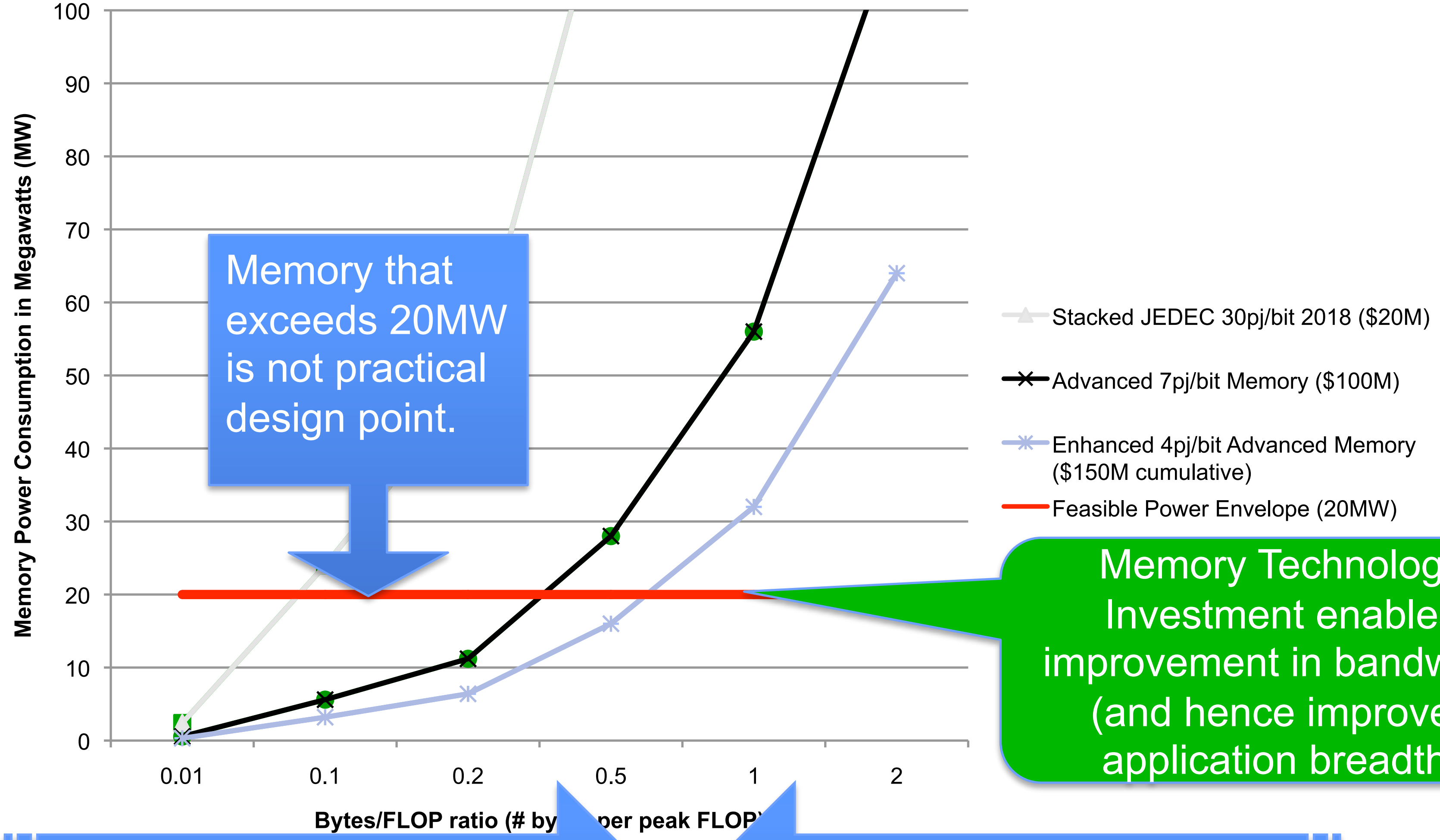
- **Hybrid Model improves 3D FFT communication performance**
  - Enables node to send larger messages
  - Substantial improvements in communications efficiency



Memory Technology Bandwidth costs power



Limiting Memory Bandwidth Limits System Scope



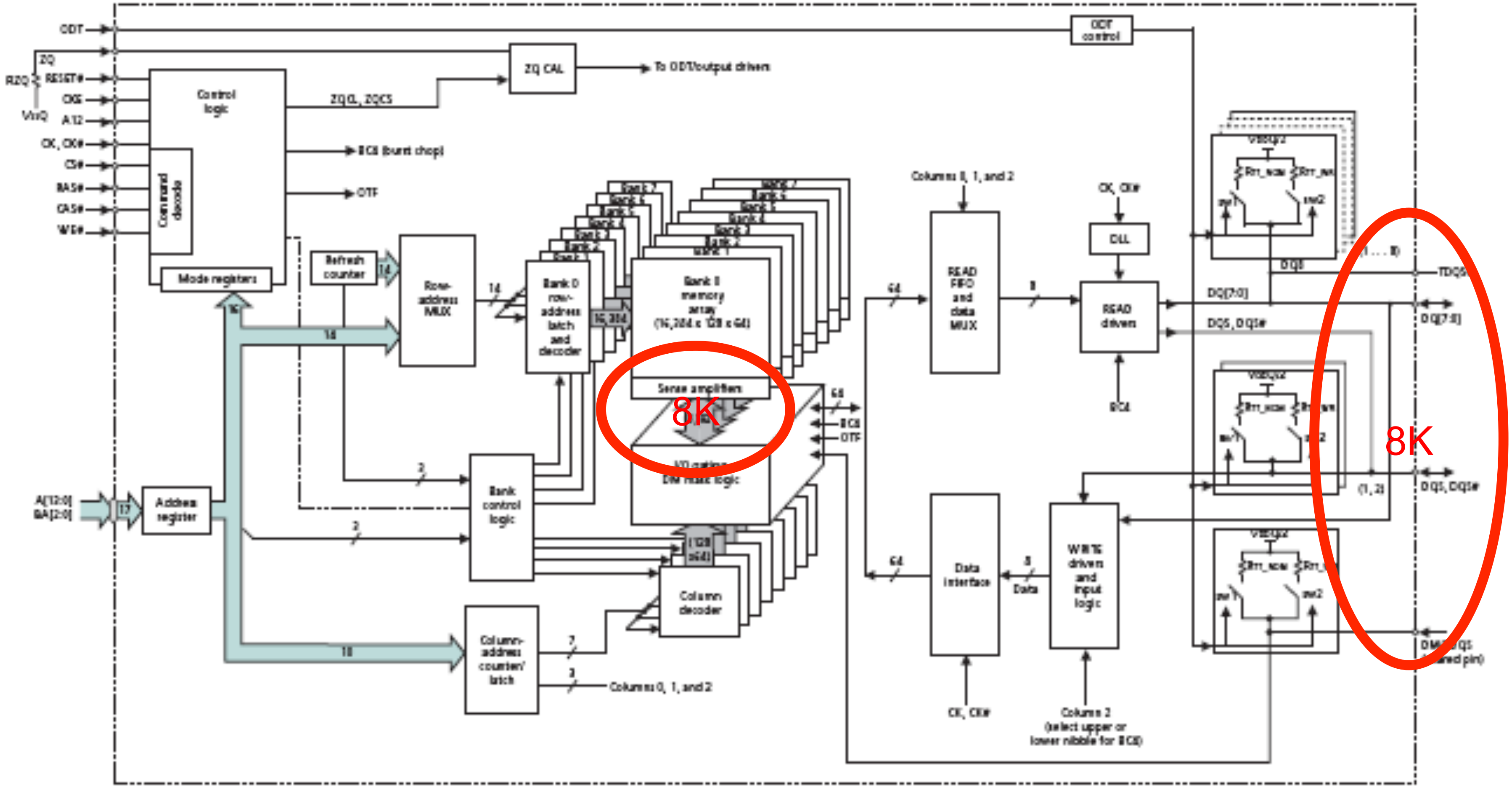
Memory that exceeds 20MW is not practical design point.

Memory Technology Investment enables improvement in bandwidth (and hence improves application breadth)

Application performance and breadth pushes us to higher BW

Power pushes us to lower bandwidth

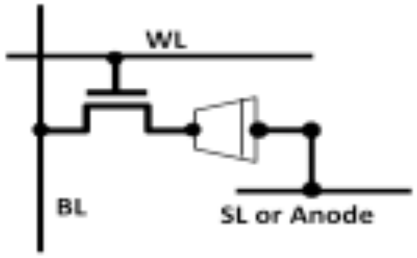
1Gbit DDR3 Architecture

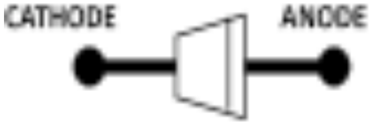


Slide from Dean Klein (Micron Technology)

Looking Beyond DRAM

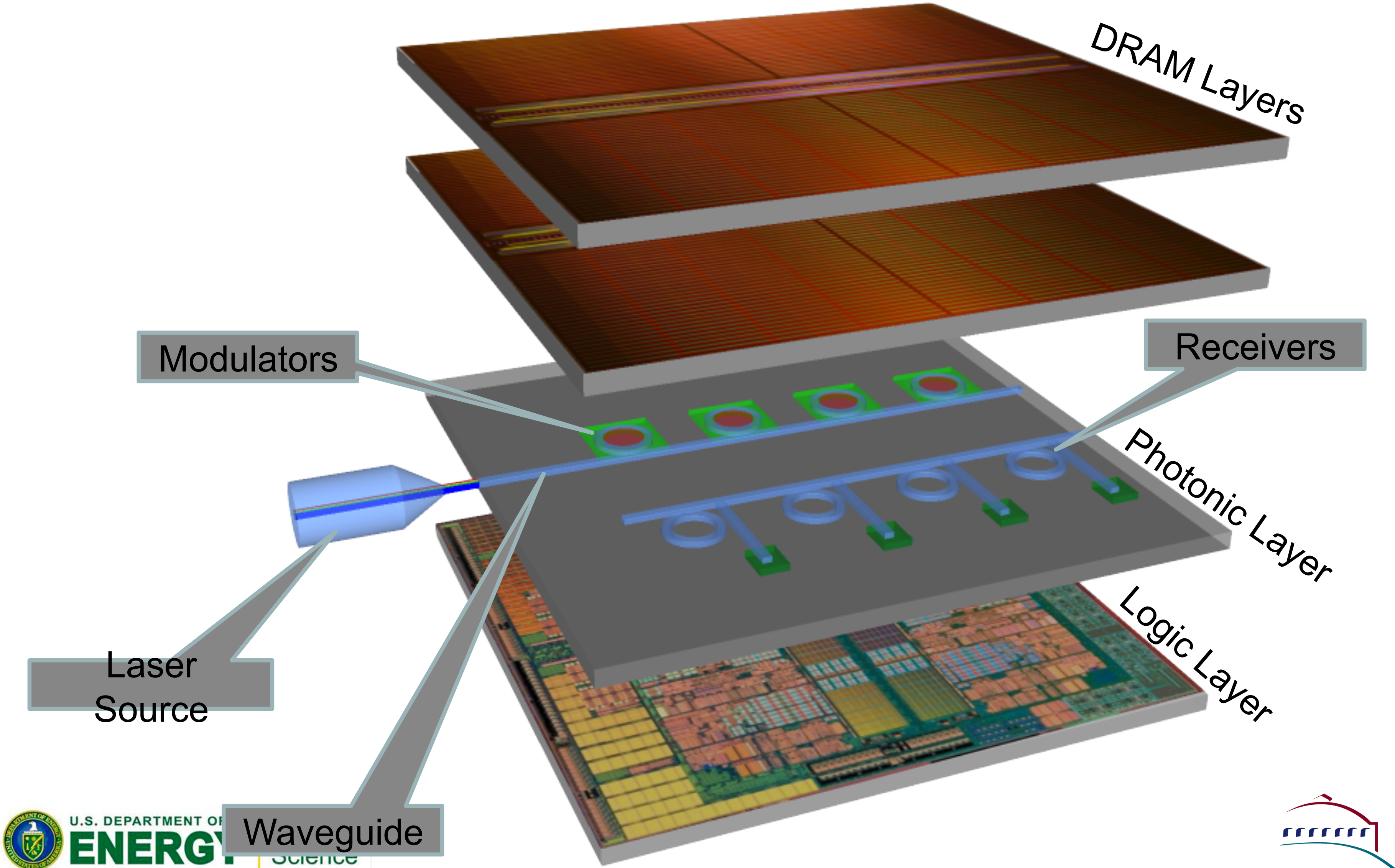
- Resistive Change RAM (ReRAM)
  - Nonvolatile - no refresh required!
  - No high-voltage requirement
  - Less energy / write (compared to FLASH)
  - More robust than FLASH
    - More cycles to cell wear out
  - Lower read energy than DRAM
    - < 1V read-out voltage
  - Similar density to flash
    - MLC capable
    - 2-4x DRAM
  - Read / write speeds comparable (or better!) than DRAM
  - Integrates very well with existing CMOS processes





Overall 10x reduction in power with a 4x increase in density

Memory Stacking and Photonics *(advanced technology solutions)*



Conclusions on Memory Technology

- **Memory technology requires major reorganization** (if domestic industry stays alive)
  - More ranks/banks, Less over-fetch, new drivers
  - Chip stacking or optical memory interfaces
  - New nonvolatile memory technologies

- **Failure to invest in memory technology means**
  - We will have to live with less memory (more emphasis on strong scaling)
  - We will have lower memory bandwidth/ computational performance (< 0.01 bytes/flop)

Exascale I/O

I/O Technology
(HEC-FSIO Discussion)

- **Mechanical Disk storage: spindle limited**
  - Requires exponentially more devices (more subject to failure)
  - Need to purchase more capacity than we want to get bandwidth
- **NVRAM/FLASH: way faster than disk, but expensive**
  - Can easily purchase sufficient bandwidth
  - But cannot afford the capacity that we need
- ***Grider's "Reese's Peanut Butter Cup" solution: Hybrid I/O with NVRAM for defensive I/O that bleeds off to disk***
- **Shared Filesystems vs. Distributed Filesystems**
  - Difficult to scale POSIX consistency model to exascale
  - Consider how to integrate node-localized storage into hierarchy
  - How does one manage a distributed filesystem?

Other I/O Issues

- Defensive I/O (for ~10x higher MTTI)
  - **Localized Checkpointing:** SCR to local NVRAM could supply required bandwidth
  - *How does one manage node-distributed persistent storage?*
- Analysis I/O
  - **In-situ (locality aware) data analysis**: e.g. MapReduce: Layout data across cluster and ship computation to the storage (functional semantics)
  - **Object database storage** (HDF, NetCDF) pushed into the storage infrastructure (interoperate with locality-aware storage)
- Data provenance
  - As we move to analysis of experimental data, need to know who touched the data and when (NASA example)
  - Requires coordination with data transport infrastructure

System on Chip (SoC) integration Moving the NIC on Chip

- **Moore's Law continues** *(but what should we do with those transistors?)*
  - **Could use it to cram more cores on chip, Or more cache**
  - **Or integrate other components (SoC) such as NIC**
  - **PCIe is wasted in cloud where nodes connected to ethernet fabric +disk in most cases (move features on chip to reduce cost)**
- **Cloud and Consumer market drivers for SoC Integration**
  - **Already see PCIe and 10GigE has moved on chip in commodity space (10G on BG/P, Niagara, and latest Intel Sandybridge. 100GigE by 2018??)**
  - **Vendors will ask you "which NIC" should we put on board?**
    - **cloud is pushing for ethernet (standards based interconnect)**
  - ***At high-end the "custom interconnect" is the "converged fabric" (e.g. Power7) with re-provisioning of pins for PCIe/Ethernet***
- ***What would you do with 100Gig NIC on each chip?***
  - ***Coordinated data transfers from each node?***
  - ***Is the "network the computer" or the "computer is the network?"***

Interconnects

Interconnects

(How do we determine appropriate interconnect requirements?)

- **Topology:** *will the apps inform us what kind of topology to use?*
  - Crossbars: Not scalable
  - Fat-Trees: Cost scales superlinearly with number of processors
  - Lower Degree Interconnects: *(n-Dim Mesh, Torus, Hypercube, Cayley)*
    - Costs scale linearly with number of processors
    - Problems with application mapping/scheduling fault tolerance

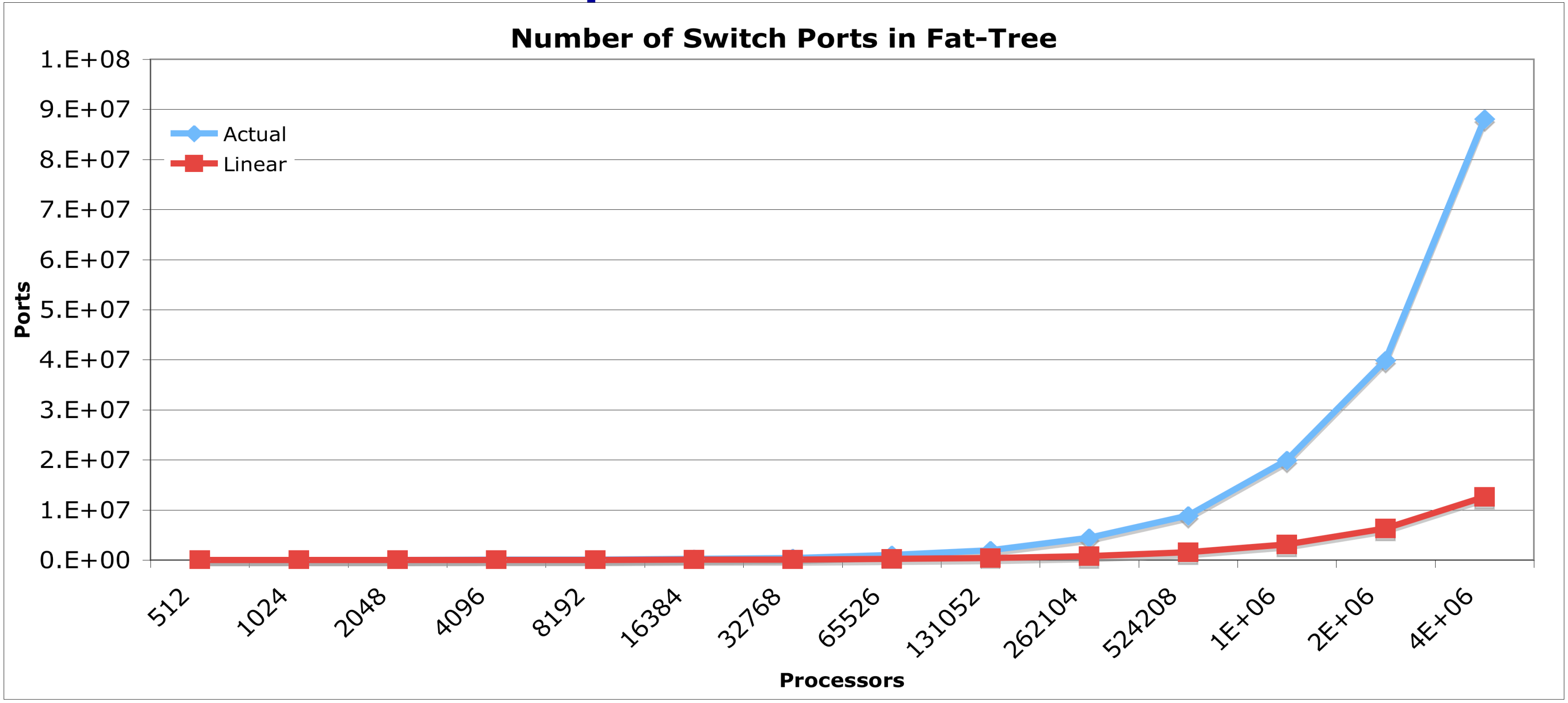
- **Bandwidth/Latency/Overhead**
  - Which is most important? *(trick question: they are intimately connected)*
  - Requirements for a “balanced” machine? *(eg. performance is not dominated by communication costs)*

- **Collectives**
  - How important/what type?
  - Do they deserve a dedicated interconnect?
  - Should we put floating point hardware into the NIC?

Interconnect Cost

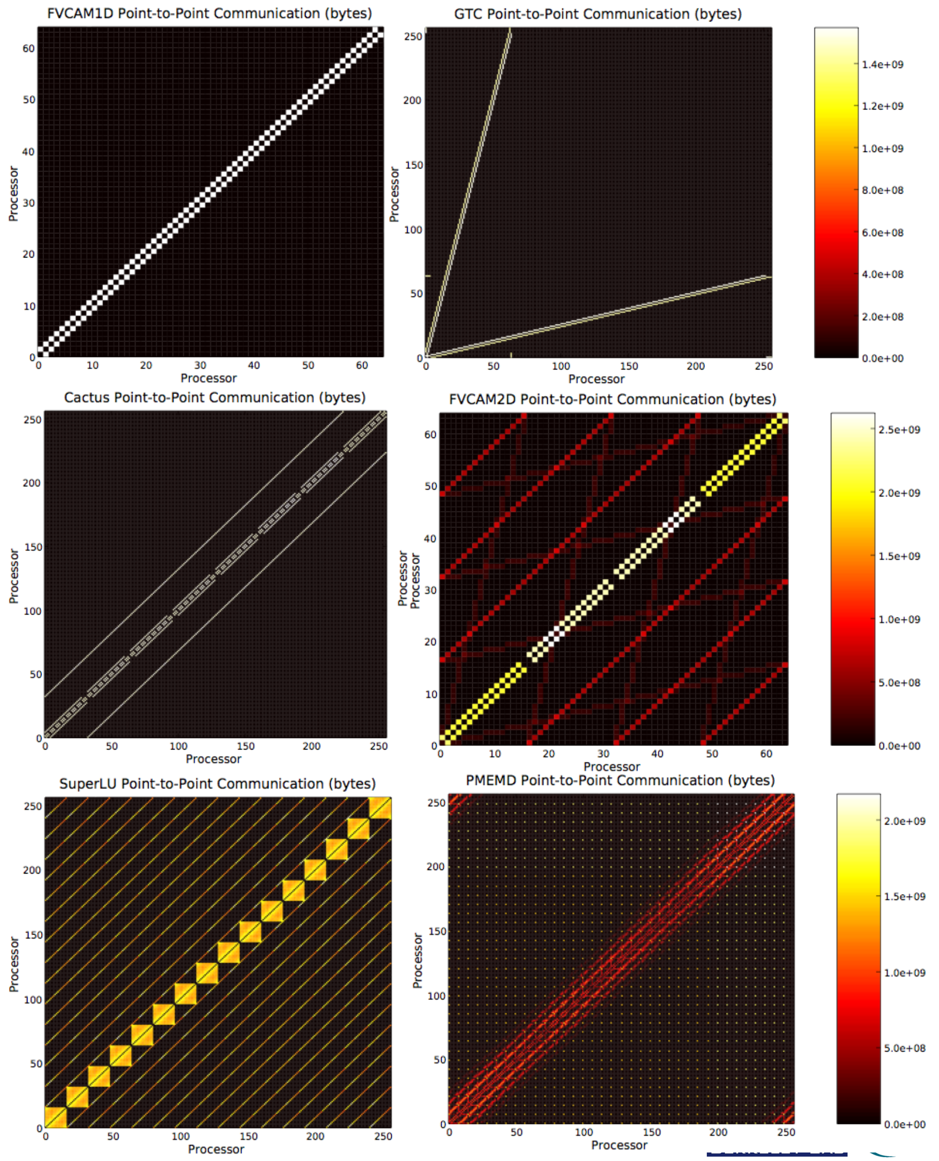
(Scalable Topologies)

- **Fully-connected networks scale superlinearly in cost, but perform the best**
- **Limited-connectivity networks scale linearly in cost, but introduce new problems**

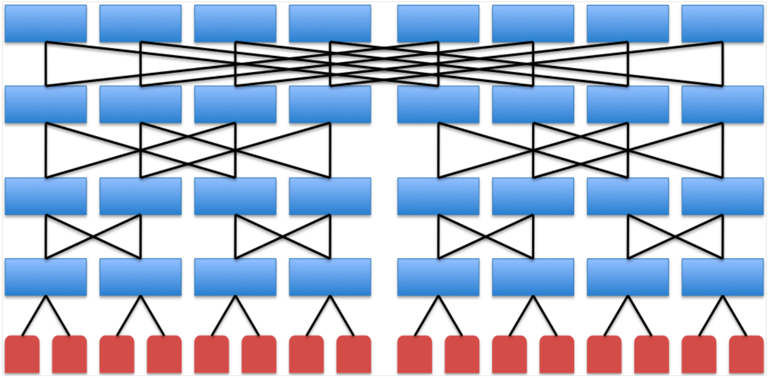


Interconnect Design Considerations for Message Passing Applications

- **Application studies provide insight to requirements for Interconnects (both on-chip and off-chip)**
  - On-chip interconnect is 2D planar (crossbar won't scale!)
  - Sparse connectivity for most apps.; crossbar is overkill
  - No single best topology
  - Most point-to-point message exhibit sparse topology + often bandwidth bound
  - Collectives tiny and primarily latency bound
- **Ultimately, need to be aware of the on-chip interconnect topology in addition to the off-chip topology**
  - Adaptive topology interconnects (HFAST)
  - Intelligent task migration?



Topology Optimization
(turning Fat-trees into Fit-trees)



A 2-ary 4-tree with 16 nodes.

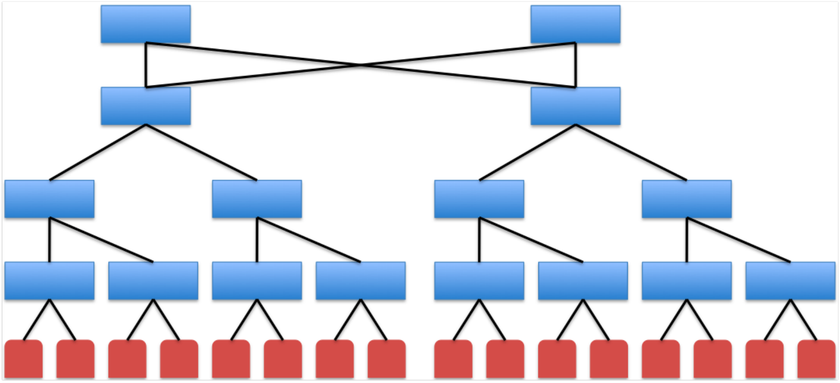
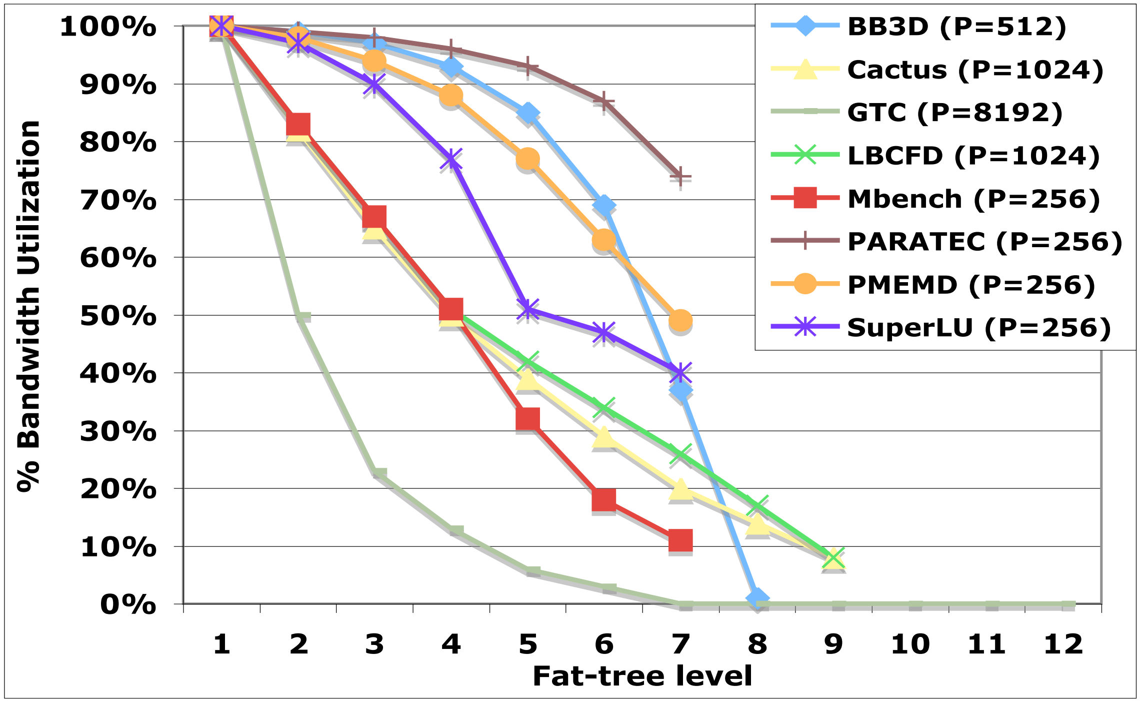


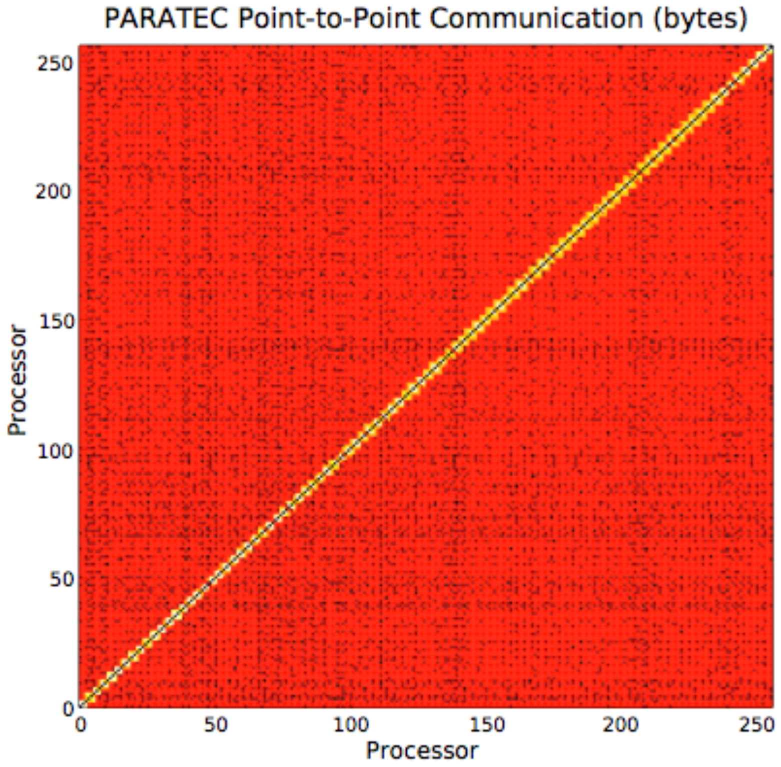
Figure 2: A $(2, 2, 4)$-TL fit-tree with 16 nodes.

- **A Fit-tree uses OCS to prune unused (or infrequently used) connections in a Fat-Tree**
- **Tailor the interconnect to match application data flows**



More on Bisection Bandwidth

- **3D FFT easy-to-identify as needing high bisection**
  - **Each processor must send messages to all PE's! (all-to-all) for 1D decomposition**
  - **However, most implementations are currently limited by overhead of sending small messages!**
  - **2D domain decomposition (required for high concurrency) actually requires sqrt(N) communicating partners! *(some-to-some)***
  - ***The issue is OVERHEAD (more of a limit than latency)***
- **Same Deal for AMR**
  - **AMR communication is sparse, but limited by message overhead**

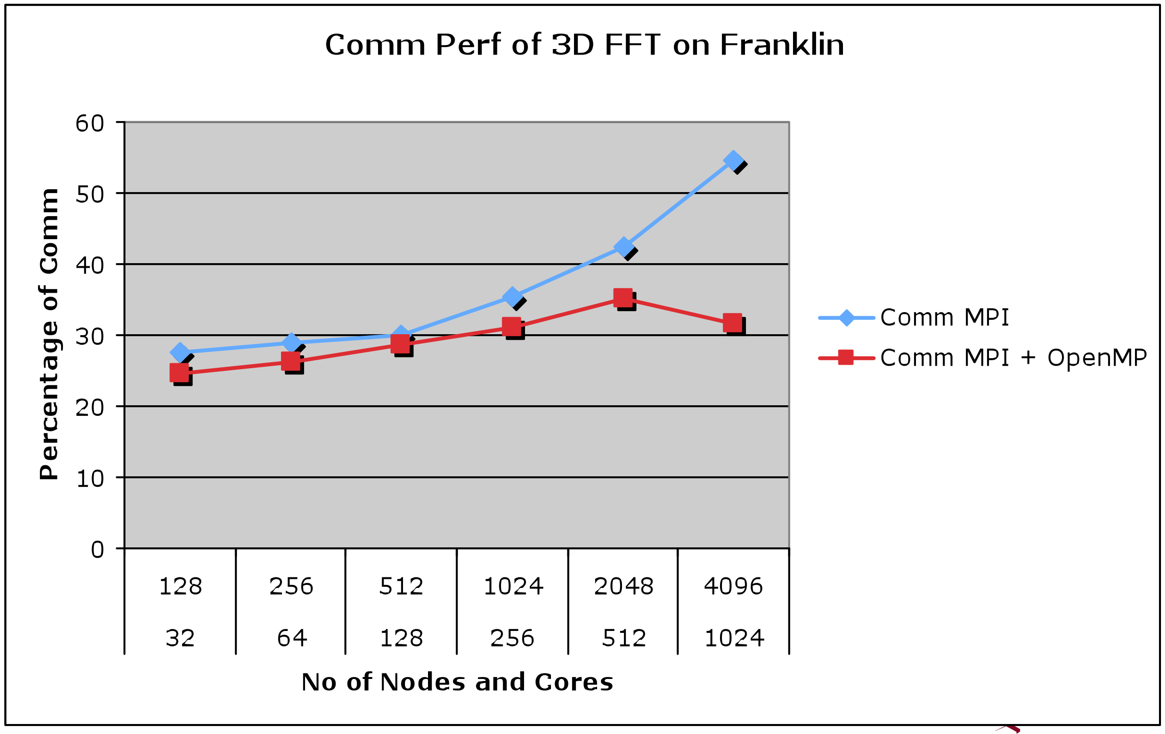


Strong Scaling of Communication Performance (message aggregation)

- **Efficient Lightweight Messaging: *Technology trends will push point-to-point messaging towards smaller message sizes.***

- ***Hybrid/hierarchical model allows increased msg sizes***

- **Hybrid Model improves 3D FFT communication performance**
  - Enables node to send larger messages
  - Substantial improvements in communications efficiency



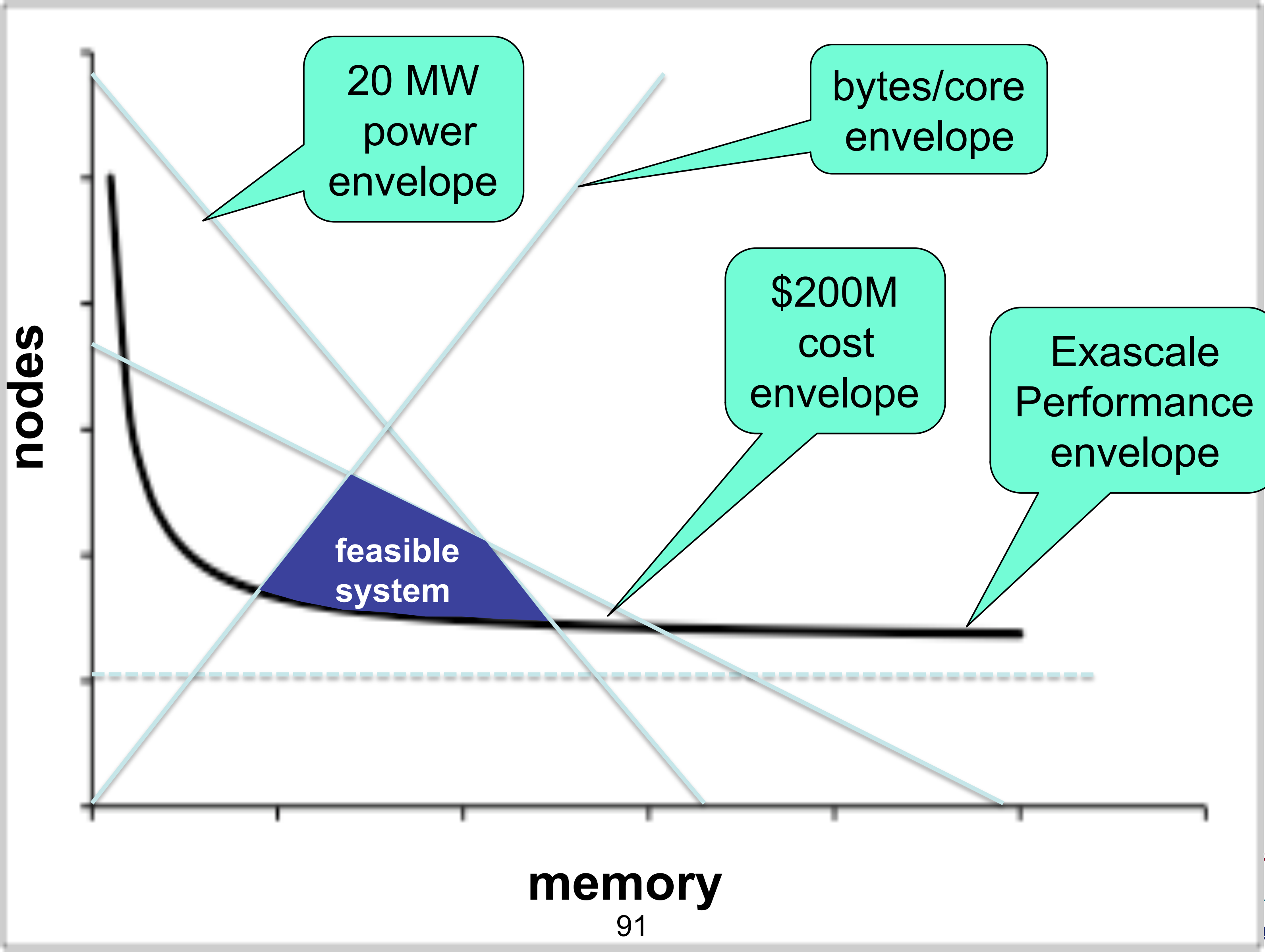
Co-Design

How do we optimize a system given all of these complicated trade-offs

Changing Notion of “System Balance”

- If you pay 5% more to double the FPUs and get 10% improvement, it’s a win (despite lowering your % of peak performance)
- If you pay 2x more on memory BW (power or cost) and get 35% more performance, then it’s a net loss (even though % peak looks better)
- *Real example: we can give up ALL of the flops to improve memory bandwidth by 20% on the 2018 system*
- We have a fixed budget
  - Sustained to peak FLOP rate is *wrong* metric if FLOPs are cheap
  - Balance involves balancing your checkbook & balancing your power budget
  - Requires a application co-design make the right trade-offs

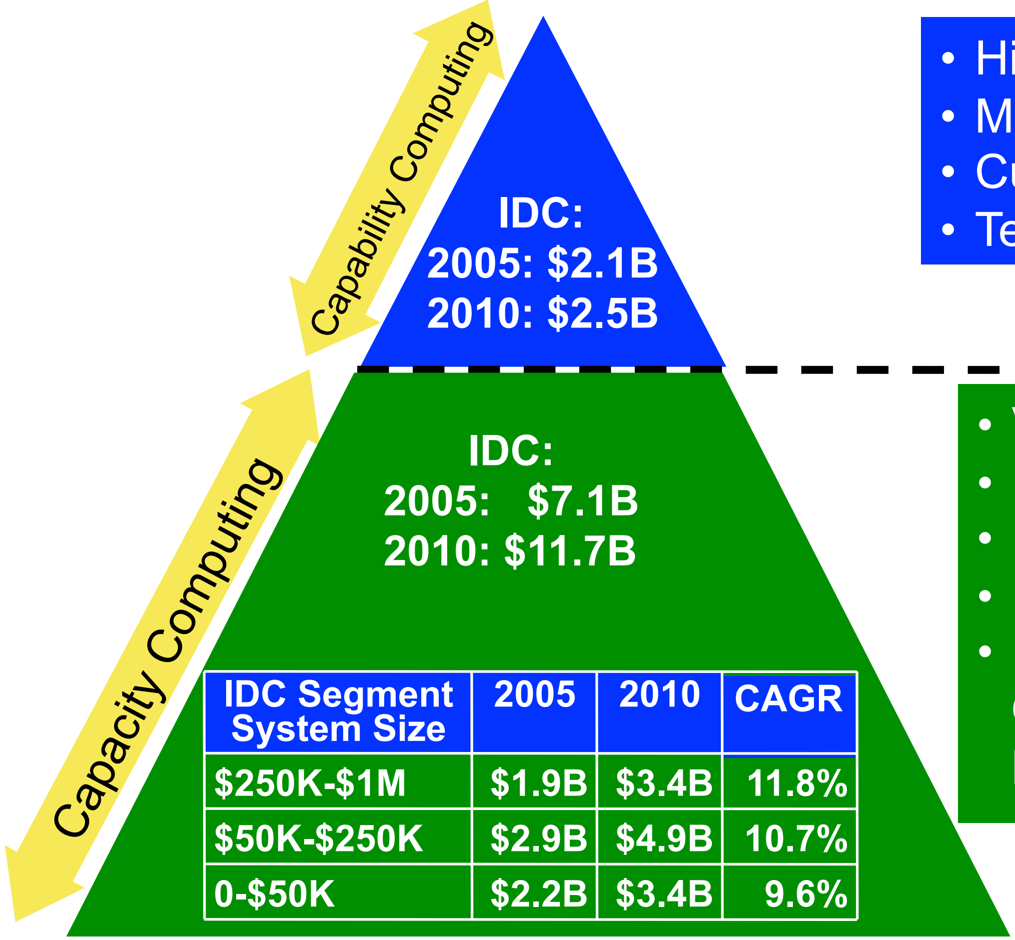
DOE Roadmap: The Trade Space for Exascale is Very Complex.



How Can We Achieve our Goals Cost Effectively?

How do we maximally leverage market forces and research investments?

Intel HPC Market Overview



Capability Computing

IDC:
2005: $2.1B
2010: $2.5B

Capacity Computing

IDC:
2005: $7.1B
2010: $11.7B

IDC Segment System Size	2005	2010	CAGR
$250K-$1M	$1.9B	$3.4B	11.8%
$50K-$250K	$2.9B	$4.9B	10.7%
0-$50K	$2.2B	$3.4B	9.6%

- High End Systems (>$1M)
- Most/all Top 500 systems
- Custom SW & ISV apps
- Technology risk takers & early adopte

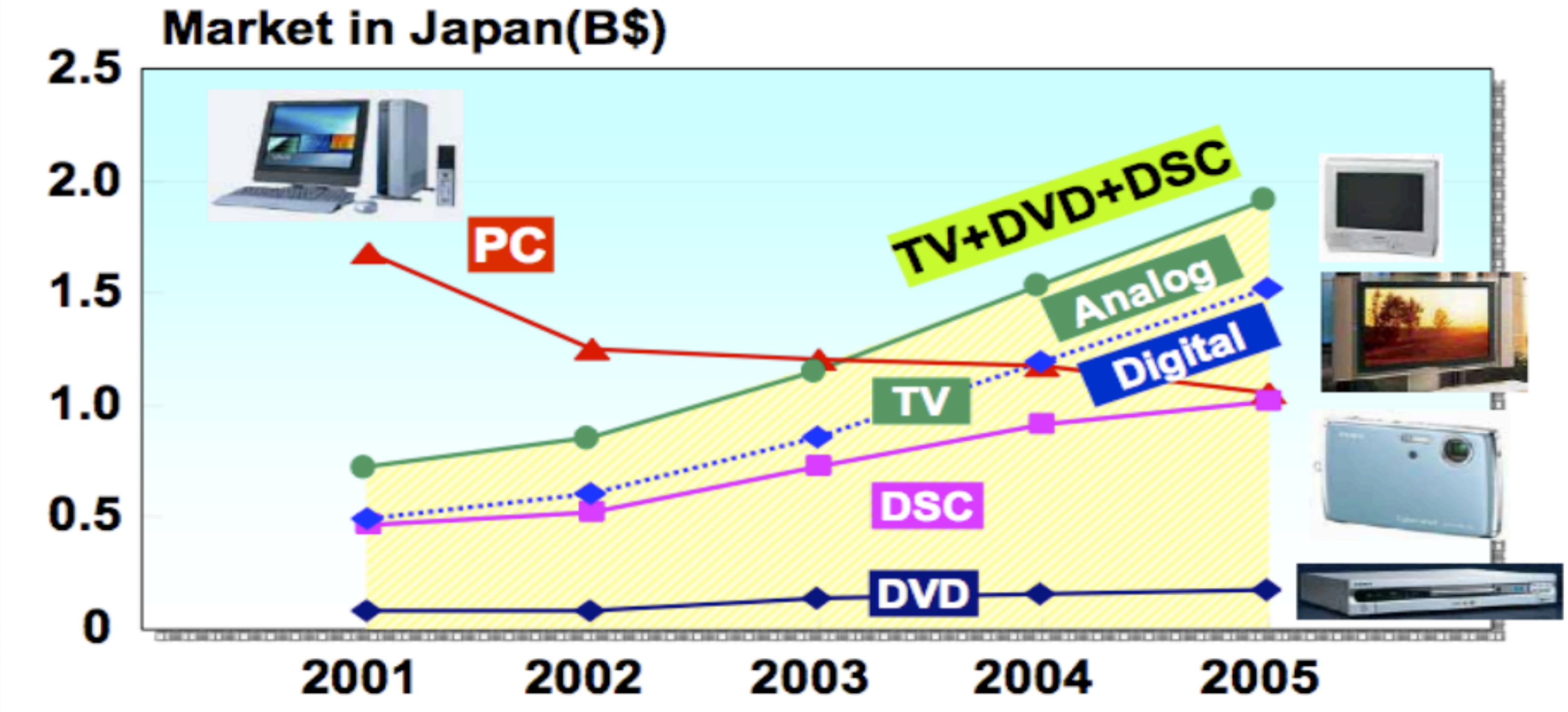
- Volume Market
- Mainly capacity; <~150 nodes
- Mostly clusters; >50% & growing
- Higher % of ISV apps
- Fast growth from commercial HPC; Oil &Gas, Financial services, Pharma, Aerospace, etc.

Total market >$10.0B in 2006
Forecast >$15.5B in 2011

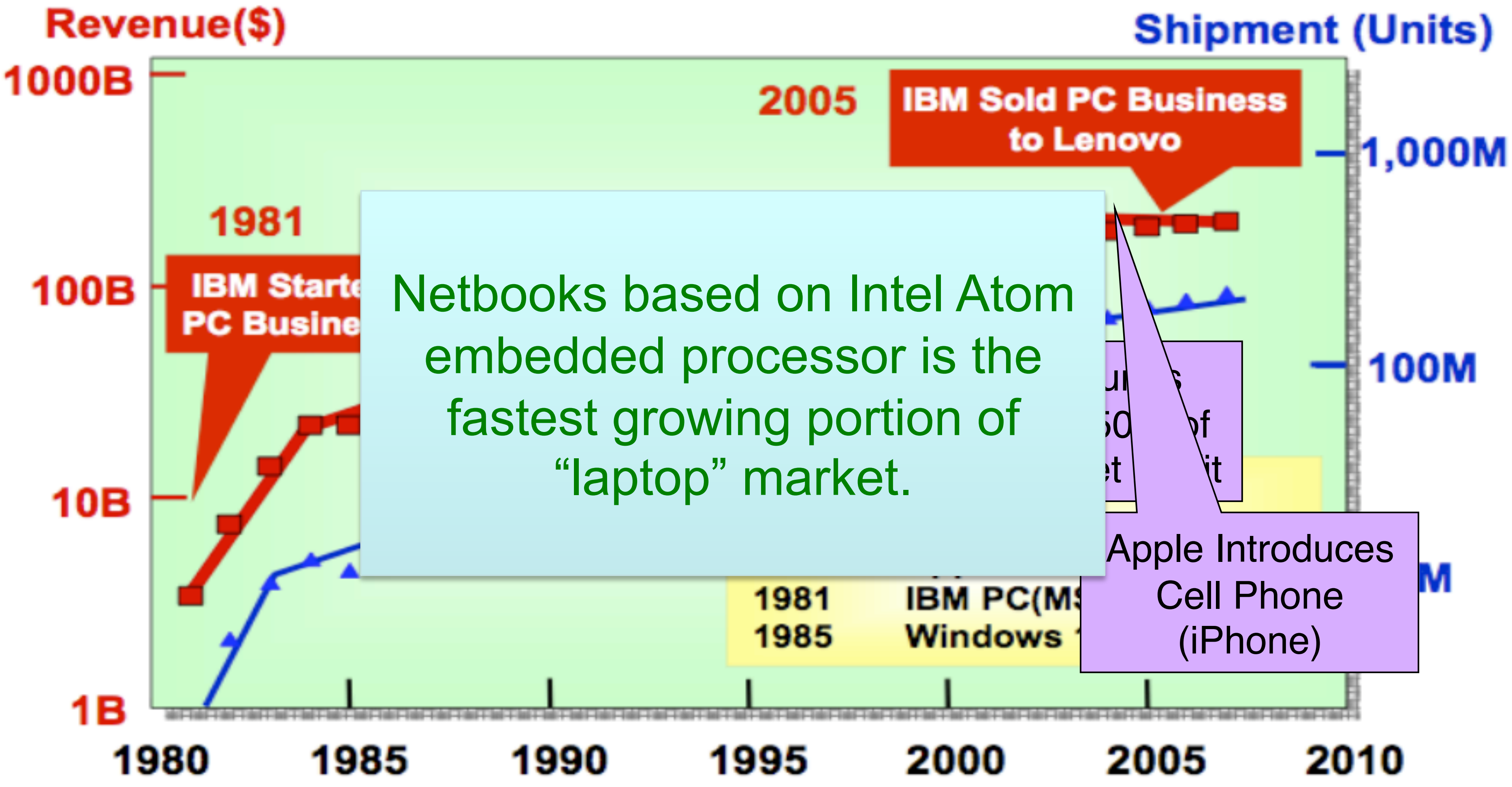
HPC is built with of pyramid investment model

Processor Technology Trends

- 1990s - R&D computing hardware dominated by desktop/COTS
  - Had to learn how to use COTS technology for HPC
- 2010 - R&D investments moving rapidly to consumer electronics/ embedded processing
  - Must learn how to leverage embedded processor technology for future HPC systems



Consumer Electronics has Replaced PCs as the Dominant Market Force in CPU Design!!



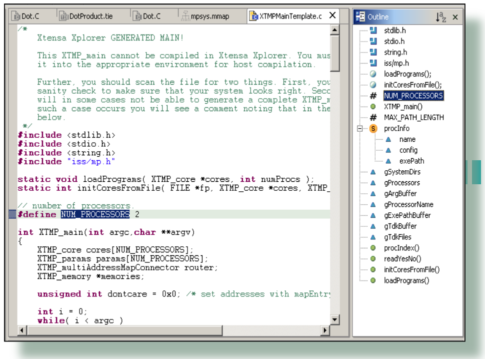
Netbooks based on Intel Atom embedded processor is the fastest growing portion of “laptop” market.

Apple Introduces Cell Phone (iPhone)

Source: IDC

Embedded Design Automation

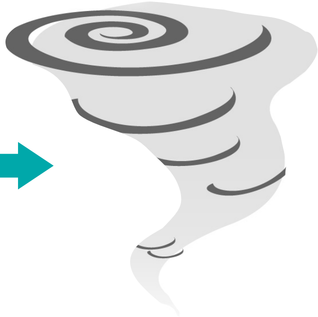
(Example from Existing Tensilica Design Flow)



```
/*
    Xtensa Xplorer GENERATED MAIN!

    This XTMP_main cannot be compiled in Xtensa Xplorer. You mus
    it into the appropriate environment for host compilation.

    Further, you should scan the file for two things. First, you
    sanity check to make sure that your system looks right. Secc
    will in some cases not be able to generate a complete XTMP_m
    such a case occurs you will see a comment noting that in the
    below.
 */
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "iss/mp.h"

static void loadPrograms( XTMP_core *cores, int numProcs );
static int initCoresFromFile( FILE *fp, XTMP_core *cores, XTMP_

// number of processors.
#define NUM_PROCESSORS 2

int XTMP_main(int argc,char **argv)
{
    XTMP_core cores[NUM_PROCESSORS];
    XTMP_params params[NUM_PROCESSORS];
    XTMP_multiAddressMapConnector router;
    XTMP_memory *memories;

    unsigned int dontcare = 0x0; /* set addresses with mapEntry

    int i = 0;
    while( i < argc )
```

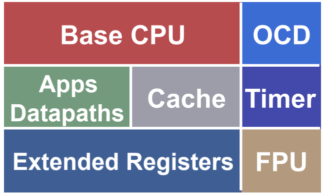
Processor configuration

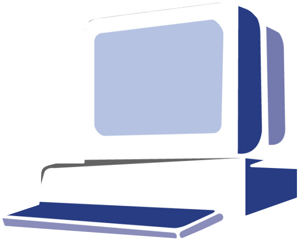
1. Select from menu
2. Automatic instruction discovery (XPRES Compiler)
3. Explicit instruction description (TIE)



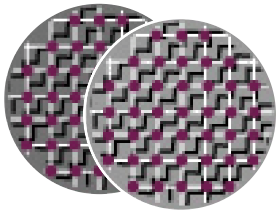
Processor Generator (*Tensilica*)

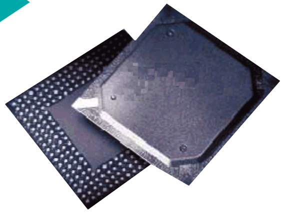
Application-optimized processor implementation (RTL/Verilog)





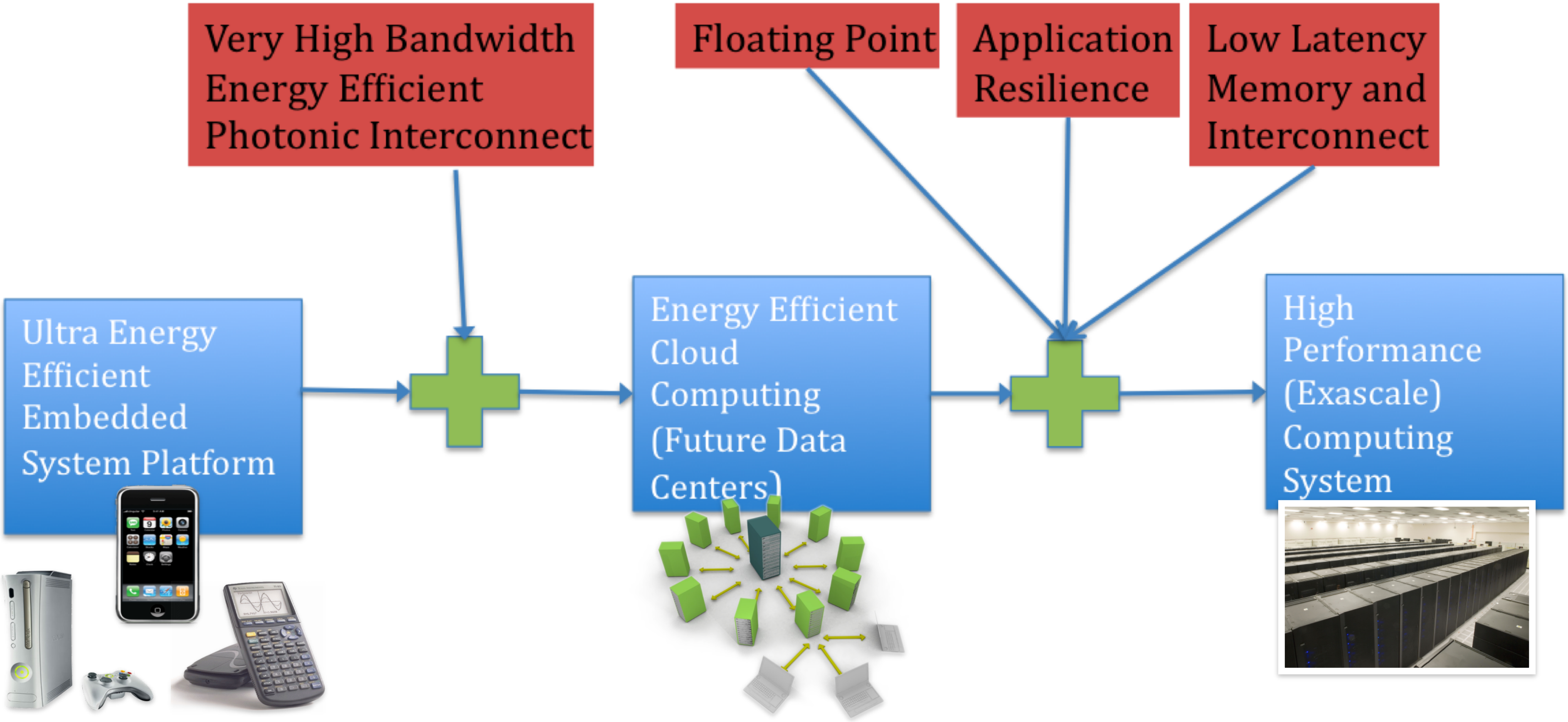
Tailored SW Tools: Compiler, debugger, simulators, Linux, other OS Ports *(Automatically generated together with the Core)*





Build with any process in any fab

Technology Continuity for A Sustainable Hardware Ecosystem



Very High Bandwidth Energy Efficient Photonic Interconnect

Floating Point

Application Resilience

Low Latency Memory and Interconnect

Ultra Energy Efficient Embedded System Platform

Energy Efficient Cloud Computing (Future Data Centers)

High Performance (Exascale) Computing System

Need building blocks for a compelling environment at all scales

Future HPC Technology Building Blocks

- **Previous Decade**
  - Optimization target: minimize price to buy more hardware
  - COTS: Redirect off-the-shelf components designed for mass market
  - This leveraged “Moore’s Law” density improvements

- **Next Decade**
  - Optimization target: minimize power consumed for work performed
  - Specialize and integrate: Embedded + SoC is proven design point
  - This leverages “Bells Law” cost efficiency: *Commodity* not *COTS*

Future HPC Technology Building Blocks

- **Previous Decade**
  - Optimization target: minimize price to buy more hardware
  - COTS: Redirect off-the-shelf components designed for mass market
  - This leveraged "Moore's Law" density improvements

- **Next Decade**
  - Optimization target: minimize power consumed for work performed
  - Specialize and integrate: Embedded + SoC is proven design point
  - This leverages "Bells Law" cost efficiency: *Commodity* not *COTS*

- **Interim solution: Accelerators**
  - Demonstrate huge efficiency potential of manycore
  - Demonstrate we have failed to learn from CM5 (PCIe)
  - Stepping stone to convergence (merge manycore with host memory)
  - But also points to benefits of some specialization

Overall Conclusions

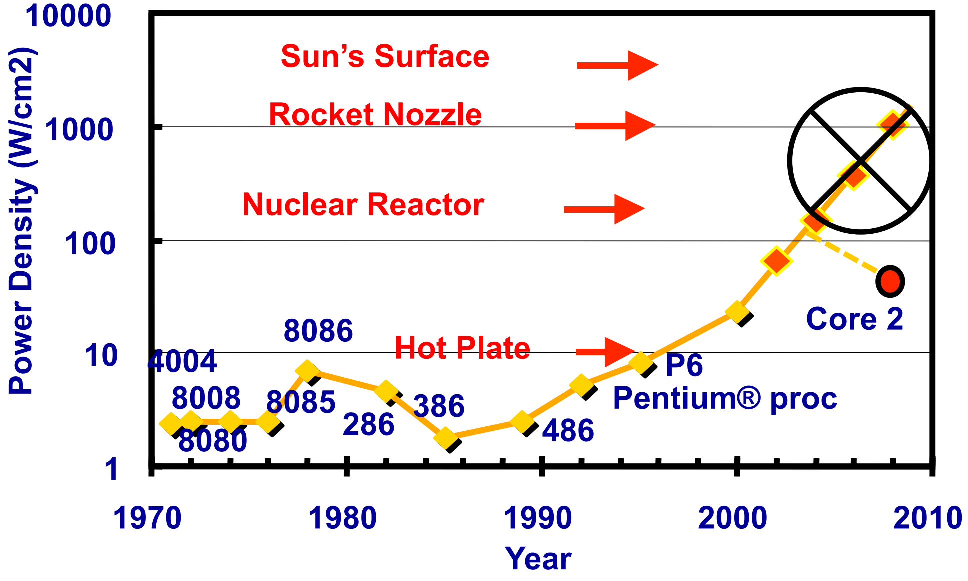
- **Supercomputers are power limited**
  - Limited by end of Dennard scaling for logic
  - Limited by energy cost of moving bits
- **Primary growth in explicit parallelism is on-chip**
  - *100x growth in parallelism on-chip*
  - *10x growth in parallelism off-chip*
- **Need a new abstract machine model that reflects hierarchical power costs**
  - *Current abstract machine model has flat or 2-level costs, which do not match technology trends*
  - *Will require fundamental advances in technology and system architecture*
  - *Will result in disruptive changes to our entire software and programming environment (see Kathy's talk!)*

More Info

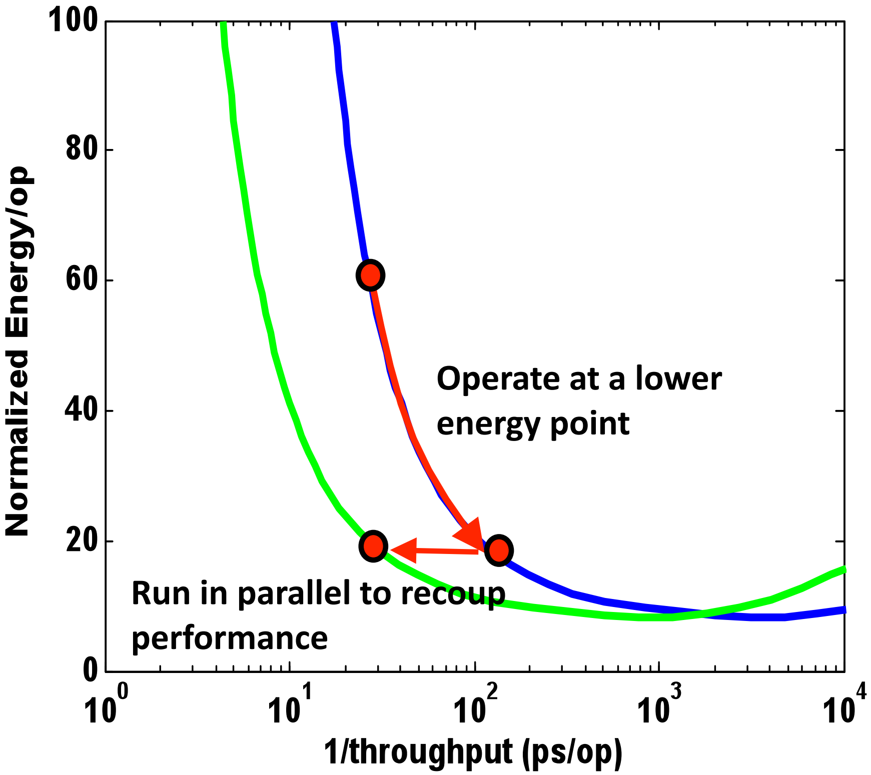
- **Green Flash**
  - http://www.lbl.gov/CS/html/greenflash.html
  - http://www.lbl.gov/CS/html/greenmeetings.html

- **NERSC Advanced Technology Group**
  - http://www.nersc.gov/projects/SDSA

Parallelism to Recover Performance



Source: S. Borkar (Intel)



- Computing performance is now limited by power dissipation. This has forced the move to parallelism as principal means of increasing performance without increasing energy per operation.

Strategies for Strong-Scaling

Strong-Scaling Drives Change in Algorithm Requirements

- Parallel computing has thrived on weak-scaling for past 15 years
- Flat CPU performance increases emphasis on strong-scaling
- Focus on Strong Scaling will dramatically change computational requirements in the future!
  - Concurrency: *Will double every 18 months (cannot partition)*
  - Implicit Methods*: Improve time-to-solution (pay for allreduce)*
  - Multiscale/AMR methods: *Only apply computation where it is required – (need better approach to metadata +load balancing)*
  - Efficient Lightweight Messaging: *All of these trends will push point-to-point messaging towards smaller message sizes.*
- *Hybrid/hierarchical model allows us to increase msg size*

Where to Find 12 Orders in 10 years?

Jardin & Keyes Example of Strong Scaling from FES

Hardware: 3

- 1.5 orders: increased processor speed and efficiency
- 1.5 orders: increased concurrency

Software: 9

- 1 order: higher-order discretizations
  - Same accuracy can be achieved with many fewer elements
- 1 order: flux-surface following gridding
  - Less resolution required along than across field lines
- 4 orders: adaptive gridding
  - Zones requiring refinement are <1% of ITER volume and resolution requirements away from them are ~10^2 less severe
- 3 orders: implicit solvers
  - Mode growth time 9 orders longer than Alfven-limited CFL

Programming Strategies for Strong-Scaling

Why MPI will persist

- Obviously MPI will not disappear in five years

- By 2014 there will be 20 years of legacy software in MPI

- Thus far, new systems are not sufficiently different to lead to new programming model

Why use Hierarchical (hybrid) model for parallelism?

- **The machine is not flat**
  - We lose a lot of performance by lying to ourselves
- **Target: Get Strong scaling on-chip and weak-scaling off-chip**
  - 100x higher bandwidth between cores on chip
  - 100x lower latency between cores on chip
  - If you pretend that every core is a peer (each is just a generic MPI rank) you are leaving a lot of performance on the table
  - You cannot domain-decompose things forever (cannot weak-scale forever)
- **MPI between nodes and *X* within node**
  - Where *X* could be OpenMP, UPC, OpenCL, CUDA, etc…

What is X?

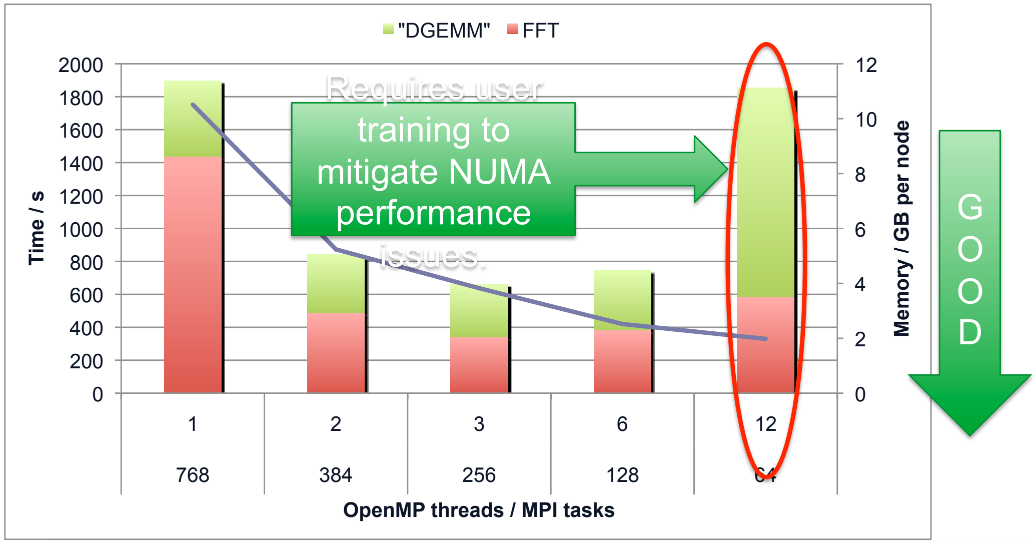
- X is probably not OpenMP
  - Too much synchronization
  - Poor expression of locality (will not scale)
- X might be UPC or PGAS language
  - Explicit definition of local vs. remote
  - Very lightweight communication
- X might be CUDA or OpenCL
  - OpenCL is very CUDA-like cross-platform extension to C language
  - CUDA is also being extended to also taret multicore
- For all X
  - Define better way to express fine-grained parallelism on-chip
  - must rigorously determine semantics for interoperation with MPI

MPI+X: Requirements for X

- **Must be able to write once and run everywhere**
  - Cannot develop architecture-specific code
  - Don't want to write code for each target! (just once please)
- **Needs to be ubiquitous**
  - Most people start a new code on a laptop and graduate to HPC systems
  - The complete development environment must be in both places (freely available)
- **Must emphasize ability to deliver strong-scaling on-chip to replace clock-frequency scaling**
  - Data parallelism might not be sufficient
  - We cannot rely on domain-decomposition for speed-up ad-infinitum (nothing to take up slack for CFL)
  - Functional partitioning (Happening at macro-scale with frameworks At micro-scale, requires bounded side-effects! its not magic)

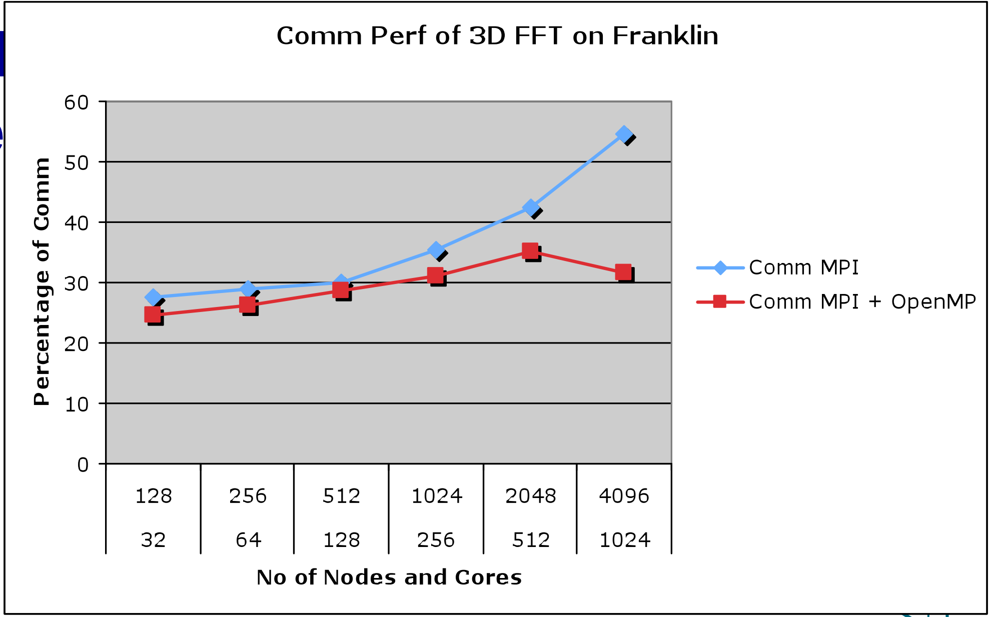
Example OpenMP for PARATEC

- **MPI+OMP Hybrid**
  - **Reduces memory footprint**
  - **Increases performance up to NUMA-node limit**



Requires user training to mitigate NUMA performance issues.

- **Hybrid Model improves 3D FI communication performance**
  - Enables node to send larger messages
  - Substantial improvements in communications efficiency


Comm Perf of 3D FFT on Franklin
Percentage of Comm
No of Nodes and Cores
Comm MPI
Comm MPI + OpenMP