The Future of Computing Performance:

Game Over or Next Level?

***Samuel H. Fuller**, Chair*

Presented with Comments by Mark D. Hill

March 22, 2011

May 12, 2011 @ U. Wisconsin

Computer Science and Telecommunications Board (CSTB)

National Research Council (NRC)

THE NATIONAL ACADEMIES

Committee On Sustaining Growth In Computing Performance

Experts Addressed the Problem

- **SAMUEL H. FULLER**, Analog Devices Inc., Chair
- **LUIZ ANDRÉ BARROSO**, Google, Inc.
- **ROBERT P. COLWELL**, Independent Consultant
- **WILLIAM J. DALLY**, NVIDIA Corporation and Stanford University
- **DAN DOBBERPUHL**, PA Semi/Apple
- **PRADEEP DUBEY**, Intel Corporation
- **MARK D. HILL**, University of Wisconsin–Madison
- **MARK HOROWITZ**, Stanford University
- **DAVID KIRK**, NVIDIA Corporation
- **MONICA LAM**, Stanford University
- **KATHRYN S. McKINLEY**, University of Texas at Austin
- **CHARLES MOORE**, Advanced Micro Devices
- **KATHERINE YELICK**, University of California, Berkeley

Staff

- **LYNETTE I. MILLETT**, Study Director
- **SHENAE BRADLEY**, Senior Program Assistant

Executive summary (Added to NA Slides)

- Highlights of National Academy Findings
  (F1) Computer hardware has transitioned to multicore
  (F2) Dennard scaling of CMOS has broken down
  (F3) Parallelism and locality must be exploited by software
  (F4) Chip power will soon limit multicore scaling
- Eight recommendations from algorithms to education

- We know all of this at some level, BUT:

Are we all acting on this knowledge or hoping for business as usual?

Thinking beyond next paper to where future value will be created?

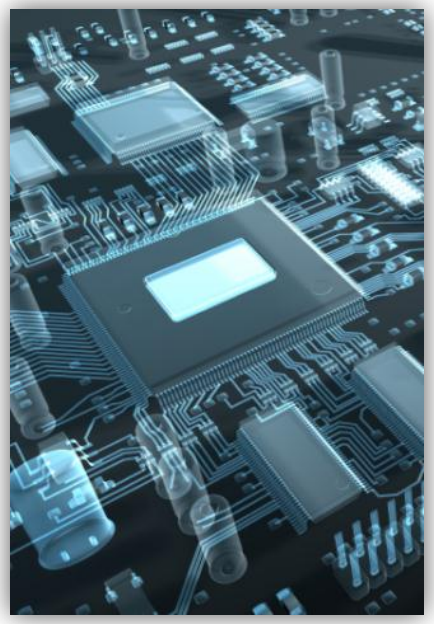
  - Questions Asked but Not Answered Embedded in NA Talk
  - Briefly Close with Kübler-Ross Stages of Grief:
    Denial ➔ … ➔ Acceptance

Project: Sustaining Growth in Computing Performance

- Proposed by **CSTB**: Computer Science and Telecommunications Board
- Sponsored by the **NSF**: National Science Foundation
- Committee was tasked to:
  - **Examine challenges to sustaining performance growth** and meeting expectations in computing across the broad spectrum of software, hardware, and architecture.
  - **Identify key problems & promising emerging technologies** and models and describe how these might fit together over time to enable continued performance scaling.
  - **Outline a research, development, and educational agenda** for meeting the emerging computing needs of the 21st century.

Sustaining Growth in Computing Performance

- What is computer performance?
- Is continued growth in computing performance needed?
- What is limiting growth in computing?
- Are programming methods that address these challenges be developed and broadly deployed ?



- Recommendations in research, practice and education

What do we mean by Computing Performance?

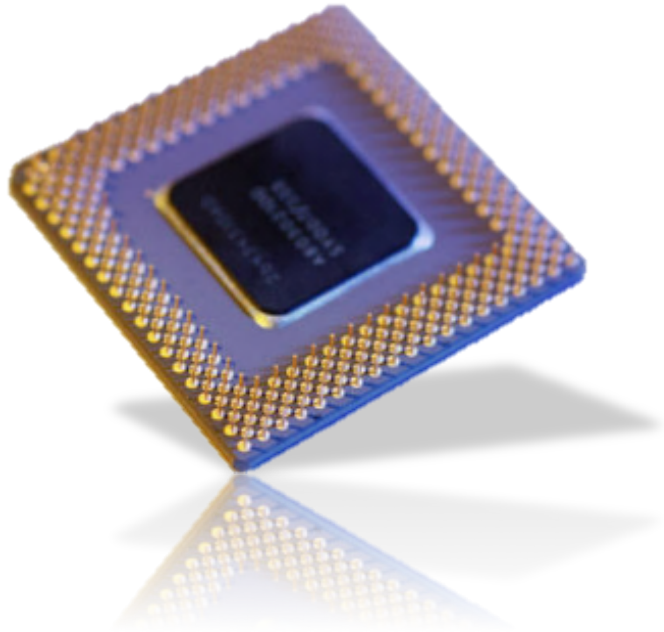
- Obvious measure of single-processor performance is raw speed
  - (clock rate) times (instructions/cycle): i.e. instructions/sec
  - An oversimplified measure, but adequate to identify dominate issue.

- Computing 'speed' is fungible and can be traded for:
  - Higher reliability, e.g., error detection/correction
  - Background operations e.g., indexing, compression, decompression
  - Redundancy
  - Near real-time translation
  - Image resolution
  - Signal fidelity
  - I/O bandwidth ….

- Delivered performance requires balance of processing performance, storage capacity and interconnect bandwidth.

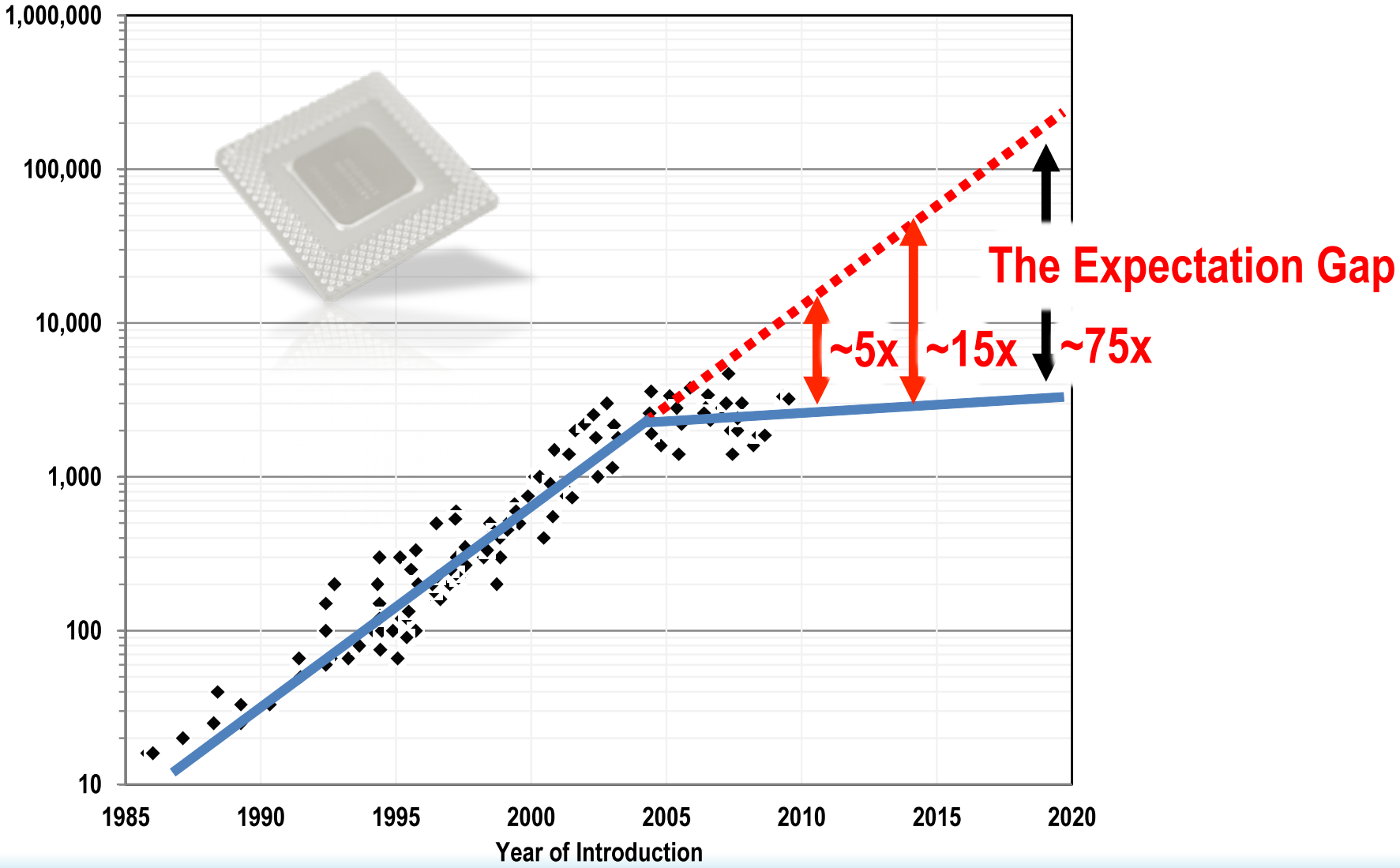
The Challenge:

Single-Processor Performance Has Plateaued



Processor Performance Plateaued about 2004 (F1)

Microprocessor Performance "Expectation Gap" over Time (1985-2020 projected)

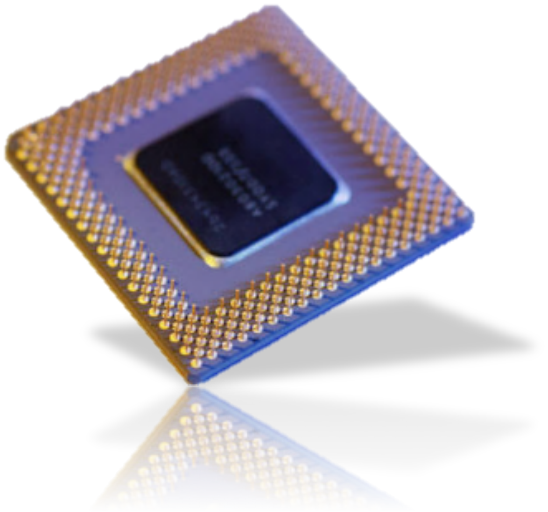


"Yes, we know."

But… have we internalized this knowledge?

Exponential Assumptions Persist

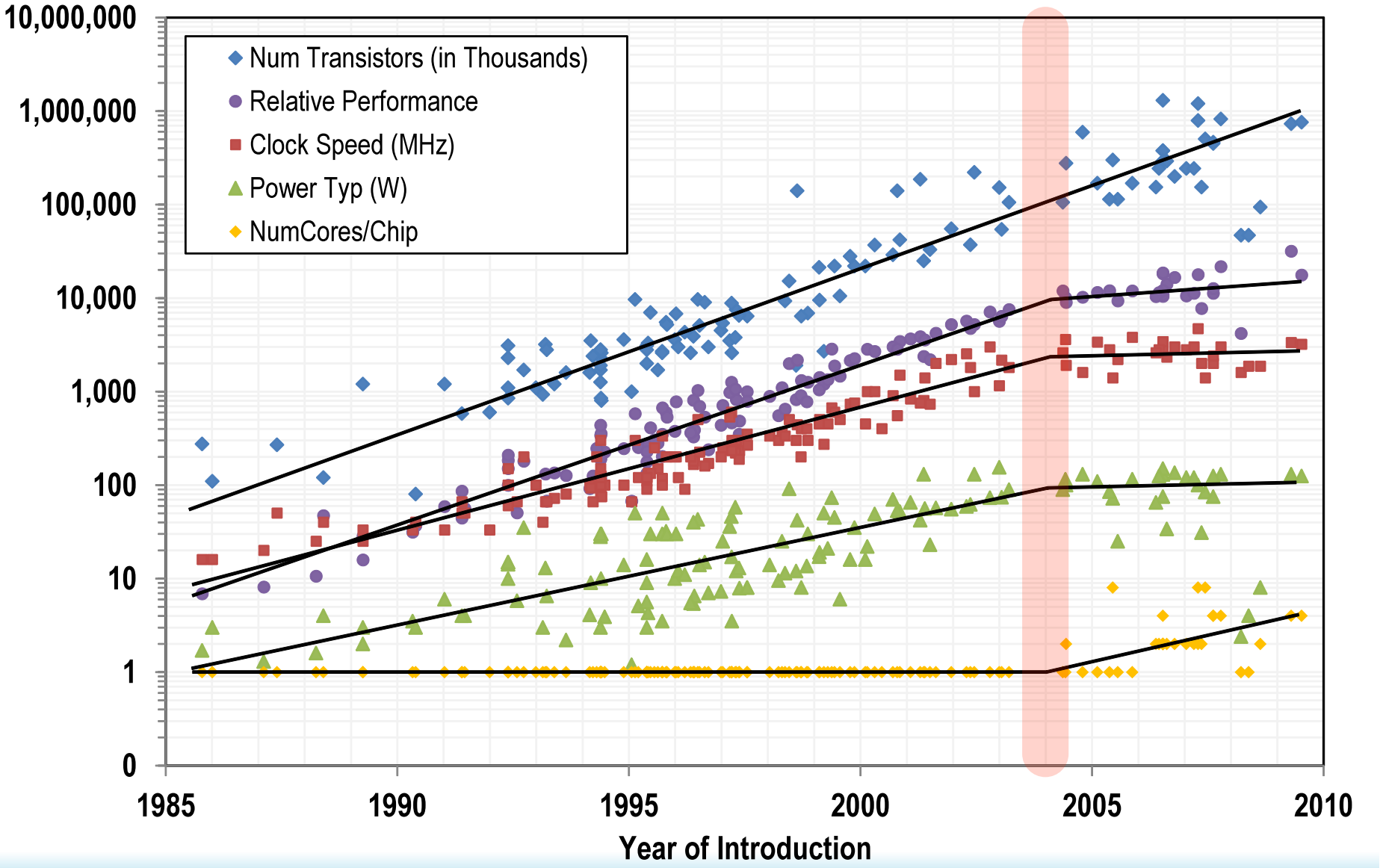
- Even among experts, hard to dislodge an implicit assumption of continuing exponential performance improvements
- *"Moore's law, which the computer industry now takes for granted, says that the processing power and storage capacity of computer chips double or their prices halve roughly every 18 months."* – The Economist, February 2010
- *"the software and other custom features become extremely important in constructing a computing system that can take advantage of the intrinsically higher speed provided by Moore's law of increasing power per chip."* – Defense Science Board, "Advanced Computing", March 2009 [arguing for parallelism, but still assuming "intrinsically higher speed"]



Question:

How best make SW contribute to performance?

Decades of exponential performance growth stalled in 2004



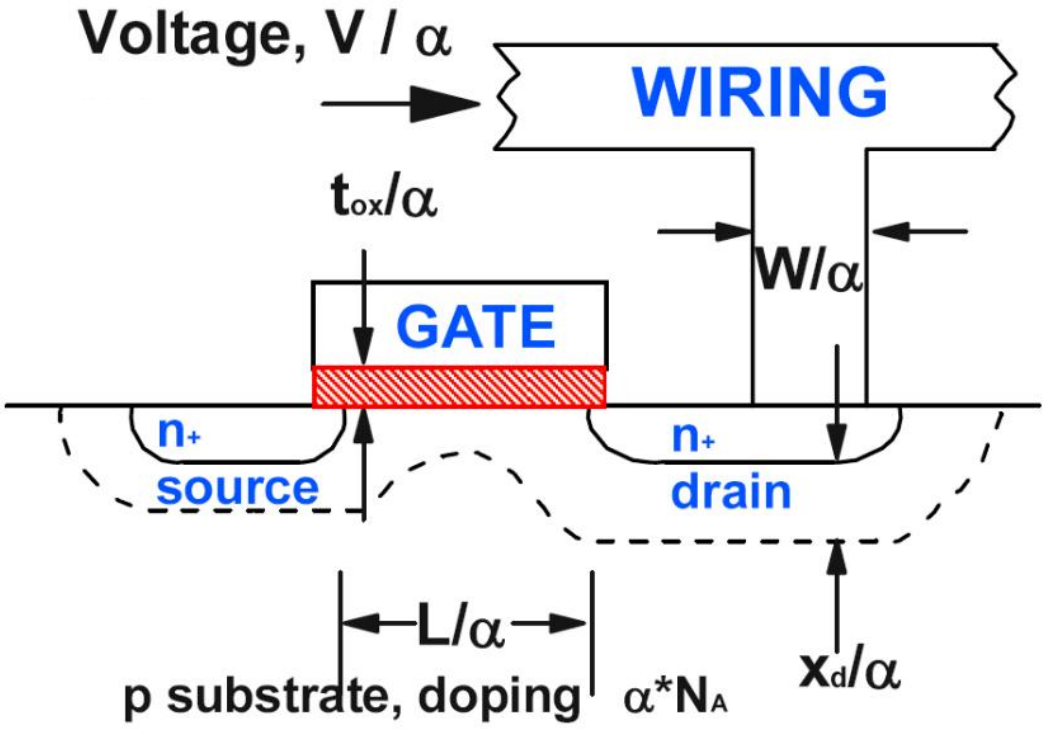
Classic CMOS Dennard Scaling: the Science behind Moore's Law

Scaling:

Voltage:	V/α
Oxide:	t_{ox}/α
Wire width:	W/α
Gate width:	L/α
Diffusion:	x_d/α
Substrate:	αN_A

Results:

Higher Density:	$\sim\alpha^2$
Higher Speed:	$\sim\alpha$
Power/ckt:	$1/\alpha^2$
Power Density:	~Constant



R. H. Dennard *et al.*, *IEEE J. Solid State Circuits*, (1974).

Why Has Power/Chip Skyrocketed?

- CMOS threshold voltage (V_t) of at least 200 to 300 millivolts is needed to make it a good switch:
  - Drive current must be high for fast switching
  - Leakage current must be low to minimize power
- Supply voltage (V_{dd}) needs to be 3+ times V_t to enable good digital switch performance
  - Implies V_{dd} is limited to 0.8 to 0.9 volts, or higher
- **Power = C f V_{dd}^2**

Post-classic CMOS Dennard Scaling

Post Dennard CMOS Scaling Rule

Scaling:

Voltage: ~~V/α~~ V

Oxide: t_{ox}/α

Wire width: W/α

Gate width: L/α

Diffusion: x_d/α

Substrate: αN_A

Results:

Higher Density: $\sim\alpha^2$

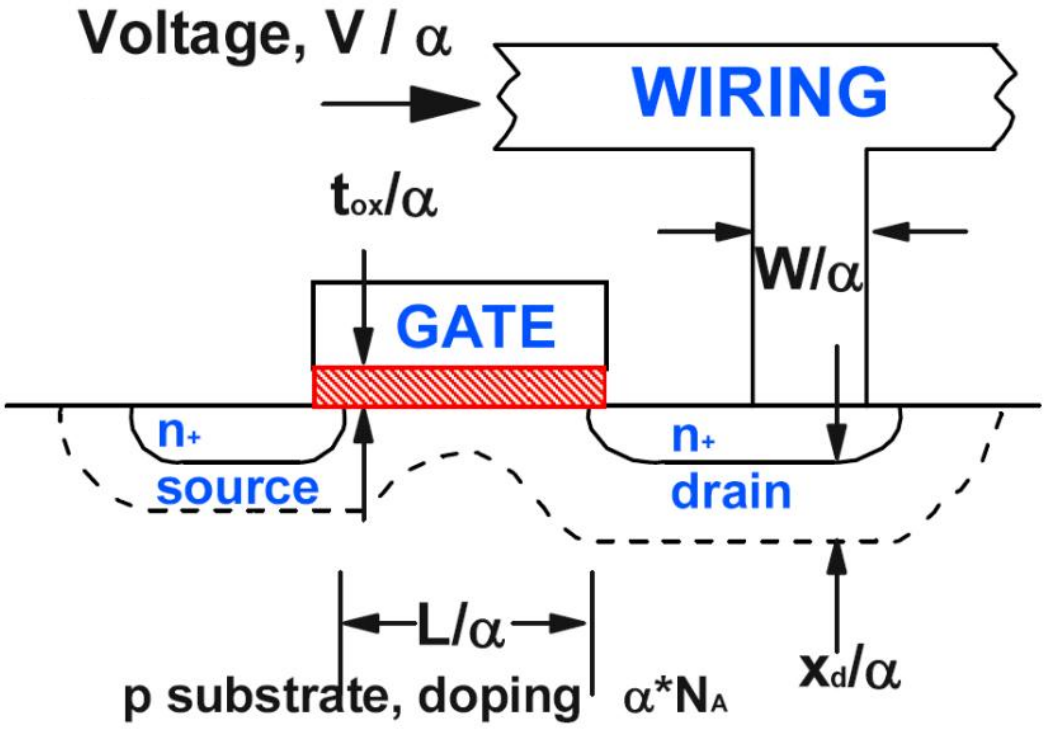
Higher Speed: $\sim\alpha$

Power/ckt: ~~$1/\alpha^2$~~ 1

Power Density: ~~~Constant~~ α^2

Question:

Chips w/ higher power (no), smaller (☹), dark silicon (☺), or other (?)



Important: Diminishing area powered up

R. H. Dennard *et al.*, *IEEE J. Solid State Circuits*, (1974).

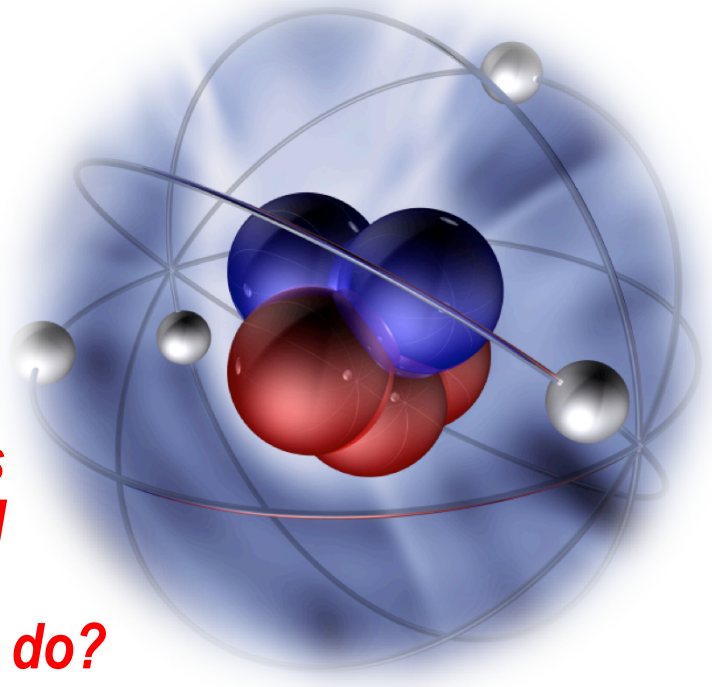
Alternatives of CMOS

Near Term – But Limited Relief to Power Constraints

- III – V materials for MOSFETs, e.g. GaAs
- Carbon nanotubes or graphene based devices

Longer Term – Much Work Remains to Bring to Commercial Reality

- Electron spin, versus electron charge., i.e. Spintronics
- Quantum devices



Important: ***No tech is as mature as was MOS when bipolar hit a power wall***

Question: ***What is an industry to do?***

Single-Processor Performance Plateau is Problematic

"Faster computers create not just the ability to do old things faster but the ability to do new things that were not feasible at all before."

Consumer Needs and Applications

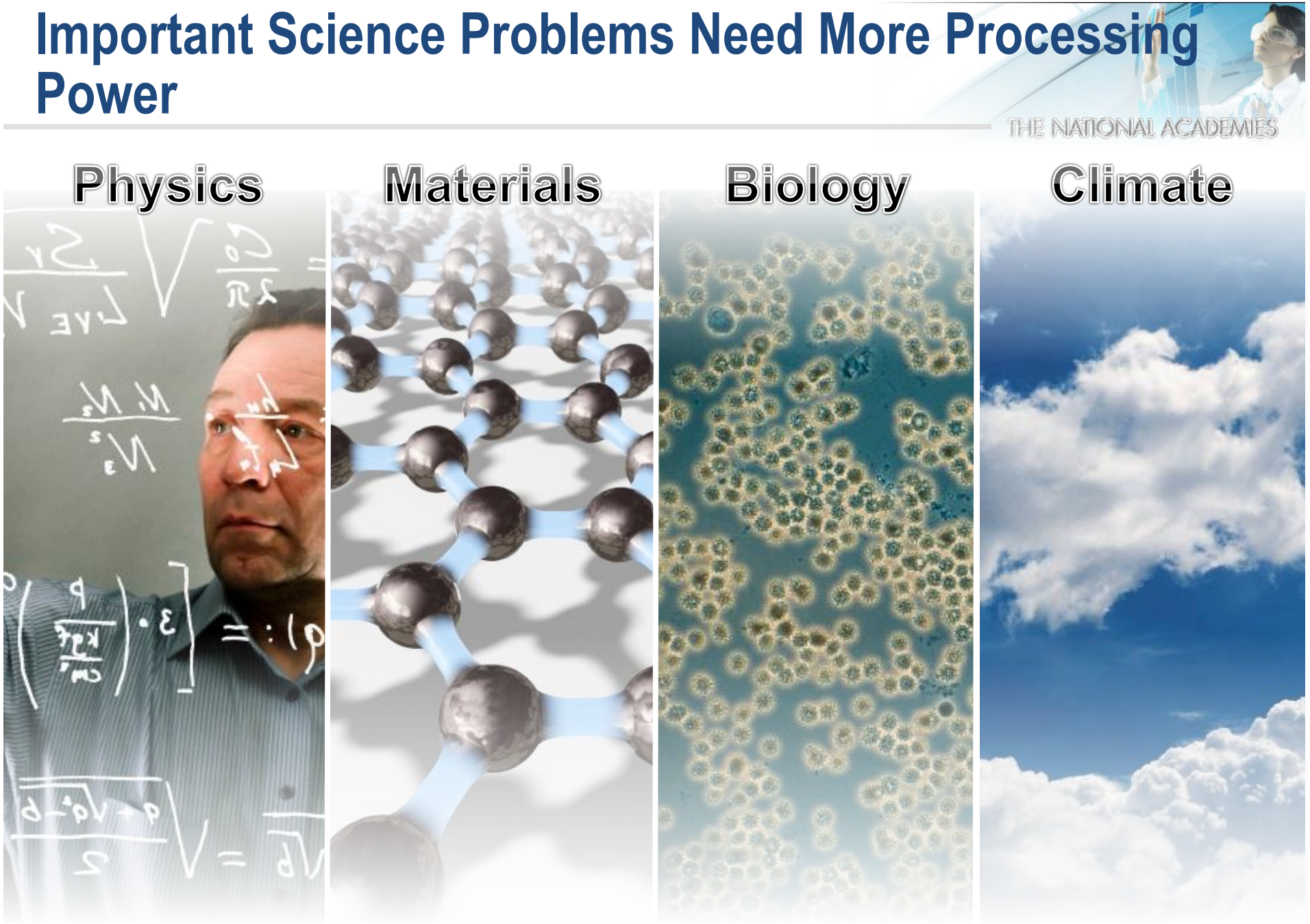
User Demands for Continued Growth in Performance

- **Digital Content Creation** — express creative skills and be entertained through various forms of electronic arts, such as animated films, digital photography, and video games.
- **Search and Mining** — ability to search and recall objects, events, and patterns well beyond the natural limits of human memory.
- **Real-Time Decision-Making** — computational assistance for complex problem-solving tasks, such as speech transcription and language translation.
- **Collaboration Technology** — more immersive and interactive 3D environment for realtime collaboration and telepresence.
- **Machine-Learning Algorithms** — filter e-mail spam, supply reliable telephone-answering services, and make book, music and other purchasing recommendations.

Question:

As computing gets cheaper, how to get people to still "buy" more?

Important Science Problems Need More Processing Power



National Defense Needs Much More Processing Power than Our Adversaries

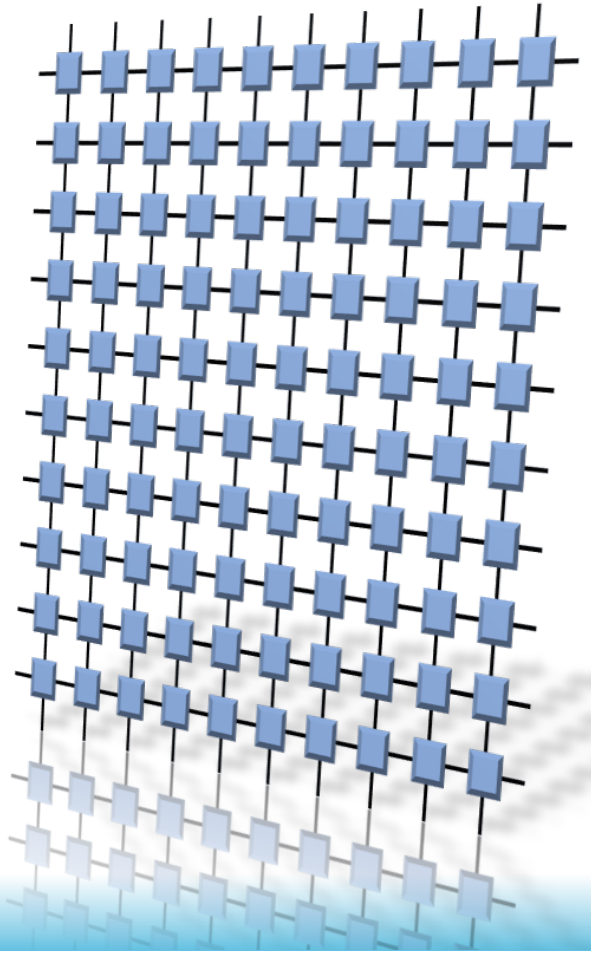
Computing Performance in the Enterprise

- **Advances in computing technology** — communication and info sharing have increased the productivity of enterprises.
- **Huge improvements in storage technology** -- affordable to keep all an organization's information online & searchable
- **Use of computation to understand data from core lines of business**— analytics has improved dramatically as computer performance has increased
- **Massive amounts of data and computational capability accessible on the Internet** have increased the demand and opportunity to develop new products and services.
- **Computing in a typical worker's day** is undergoing a momentous transformation from being useful yet nonessential to being the foundation for around-the-clock relied-on vital services.

Parallel Systems Are the Only Way to Continue the Exponential Growth in Computing Power

Not just for HPC, but for mainstream computing:
mobile devices to server farms

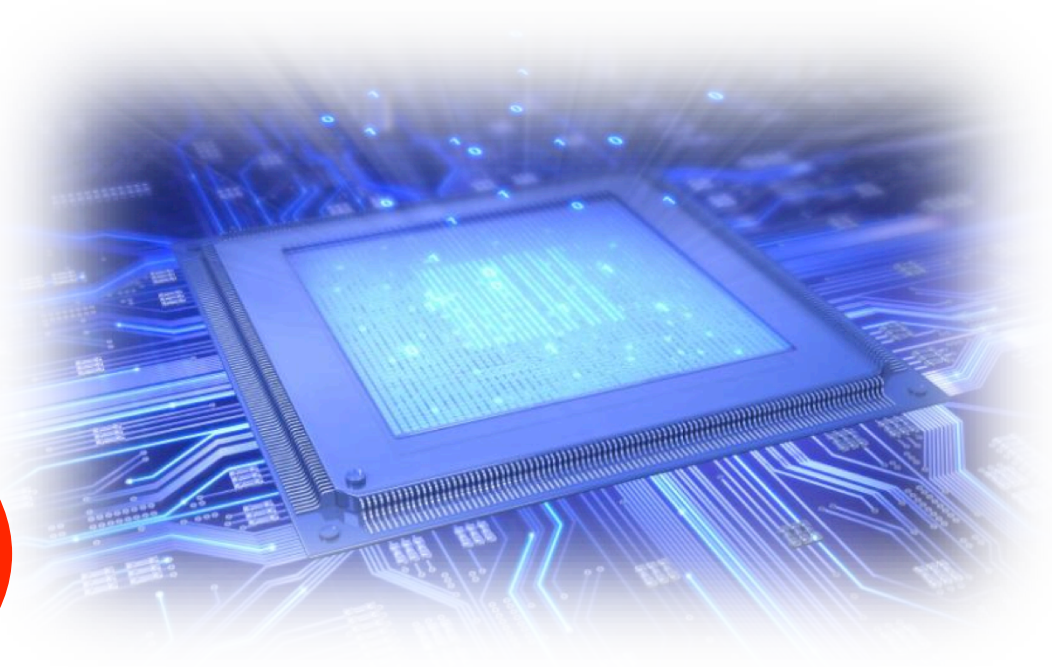
"In the future, all software must be able to exploit multiple processors to enter into a new virtuous cycle with successive generations of parallel hardware that expands software capabilities and generates new applications."



Parallelism Is Now A Necessity

- Software has not only taken advantage of Moore's bounty, but **assumed and depended** on hardware to provide ever-increasing performance of sequential processing

- Now it's up to continuing innovations in algorithms and software systems to enable ongoing performance growth



Question:

But most only know sequential programming, even CS professors?

Compatibility of Processors Generations has Postponed the Need for More Efficient Software

The Past:

- Key driver in the virtuous cycle of exploiting the effects of Moore's law is that applications benefited from processor performance improvements without those applications having to be adapted to changes in hardware.
- Programs ran faster on successive generations of hardware, allowing new features to be added without slowing the application performance.

Now and the Future:

- Computers now contain multiple processors yet most existing software is sequential.
- The shift in the hardware industry has broken the performance-portability connection in the virtuous cycle:
  - *Sequential programs will not automatically benefit from increases in system performance due to the use of multiple processors*

Cracks in the Virtuous Cycle

Costs associated with fabrication increasing dramatically

Innovation

Diminishing improvements in architecture, power efficiency

Demand → Revenue

Improved Architecture

Diminished uptake w/o sustained performance improvements

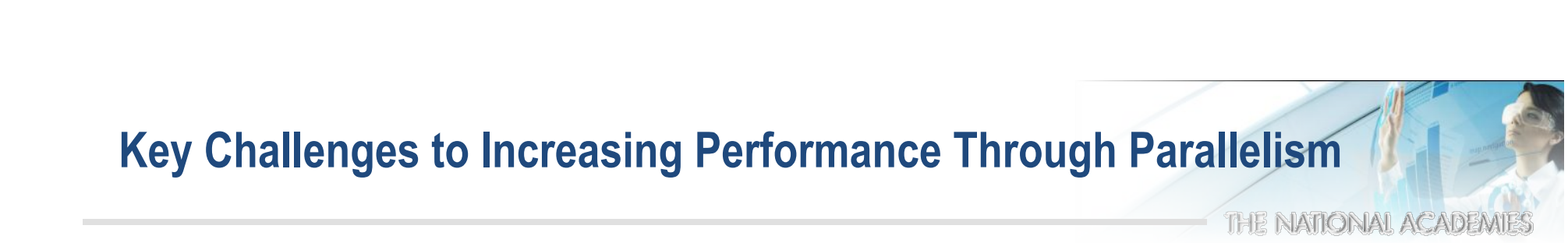
Less dramatic improvements

Improved architectures no longer transparent to developers

Visible Improvements & Innovations

Software Leveraging

Key Challenges to Increasing Performance Through Parallelism



1. Finding independent operations

2. Communicating between operations

3. Preserving locality between data and operations

4. Synchronizing concurrent operations

5. Balancing the load represented by the operations among the system resources

Locality's Increasing Importance

- Main-memory bandwidth, access energy, and latency have all scaled at a lower rate than the corresponding characteristics of processor chips for many years
- Increased use of chip multiprocessors means that the bandwidth gap will probably continue to widen
- To keep memory from severely impacting system power and performance: minimize the distance data must travel to get to the compute unit
- A key challenge in exploiting locality is developing abstractions for locality, independent of any particular target machine

Software Abstractions and Hardware Trends

- Successful software abstractions are needed to enable programmers to express
  - The parallelism that is inherent in a program
  - The dependences between operations
  - Structure a program to enhance locality
  - Avoid being tied to a specific hardware configuration
  - *All without being bogged down in low-level architectural details*

Question:

How get more than one of (a) good performance, (b) general, & (c) easy to use?

- Examples
  - MPI: used in scientific programming with large FE simulations
  - MapReduce: range of applications in search and data fusion
  - Cilk: minimum extension to C++ for parallel programming
  - CUDA: high level language for applications that can map to GPU arrays

Software Abstractions and Hardware Trends (cont)

- Successful mechanisms will enable
  - low-overhead communication and synchronization
  - migration of data and operations to maximize locality and balance load.

- Some trends in hardware architecture:
  - Multiple processors sharing a memory, e.g. multicore/multiprocessors
  - Multiple computers interconnected via a high-speed comm network.
  - Single processor containing multiple execution units, e.g. DSPs, VLIW processors
  - Array of specialized processors, e.g. GPGPU.

- Current versions of these architectures will likely evolve substantially to support the most promising programming systems

Software Abstractions and Hardware Trends (cont)

- We may see entirely new hardware architectures in support of not yet-developed programming approaches to parallel computing.
  - FPGA fabrics integrated with execution units
  - **Heterogeneous** multiprocessor / multicore systems
  - Application specific accelerators, with varying degrees of flexibility

Rethinking The Software Stack

- A key part of modern programming systems is the modern software stack
  - Libraries
  - Compilers
  - Runtime system
  - Virtual machines
  - Operating system.

Questions:

Is the future integrated designs (like iPad)?

Only few programmers go to lower level?

- The future stack must enable the optimization of the **five key challenges** to scalable and efficient performance:
  - Independent threads
  - Communication
  - Locality
  - Synchronization, and
  - Load-balancing.

Question:

How expose this to higher levels?

Parallel Computing also faces a Power Challenge (F4)

- Initial approach to managing power
  - In the past, to double performance in a given technology required quadrupling the number of gates -- and hence power
  - As multiple processors are put on a single chip, processor complexity is reduced to keep overall chip power within bounds

- However, this only works over a limited range of processor complexity
  - Report compares low end ARM to a high end X86 processors

- Potential future directions
  - Heterogeneous processors
  - Specialized array of very simple processing elements: e.g. GPU's
  - FPGA fabrics with embedded compute units
  - Processors specialized for application areas.

Important: Power will stop multicore & GPU scaling

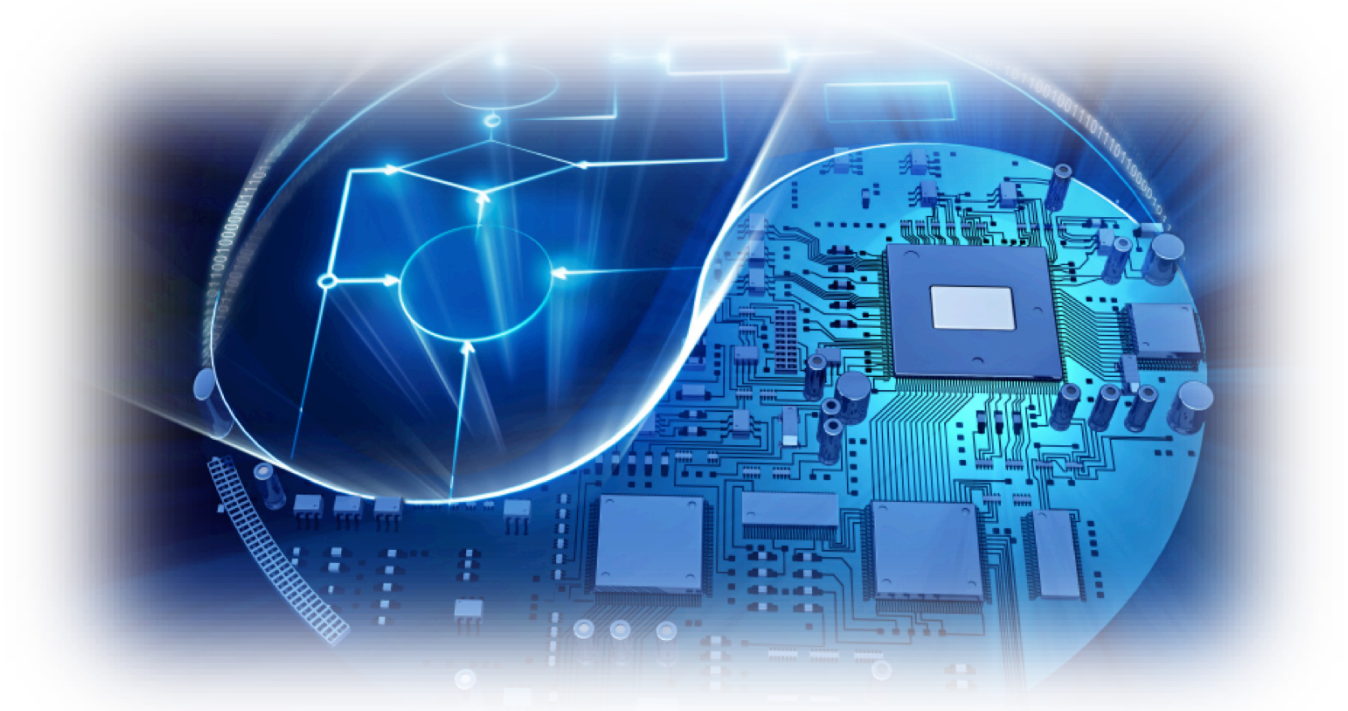
Question:

When multicore & GPU scaling crawls/stops, what next?

(Boeing provides value even though airplanes no longer get faster)

Game Over or Next Level?

This Is A Golden Time for Innovation in Computing Architectures and Software



Recommendations

From the Committee on Sustaining Growth in Computing Performance

Recommendations

- **Overarching theme:**

  Much greater focus on improvements and innovations in parallel processing …

  and* on making the transition to *computing centered on parallelism

Algorithms and Software Recommendations

1. **Invest in research in and development of algorithms** that can exploit parallel processing
2. **Invest in research in and development of programming methods** that will enable efficient use of parallel systems not only by parallel-systems experts but also by typical programmers
3. **Focus long-term efforts on rethinking of the canonical computing "stack"** in light of parallelism and resource-management challenges
   - Applications
   - Programming language
   - Compiler
   - Runtime
   - Virtual machine
   - Operating system
   - Hypervisor
   - Architecture

Question: Does this require intimate cooperation with HW? By whom?

Architecture Recommendations

4. **Invest in research on and development of parallel architectures driven by applications**, including enhanced chip multiprocessors, massively data-parallel architectures, application-specific architectures, and more radical approaches

Important: CLOUD is inflection point

K. Lowery crawled 100M pages for $200

In THREE days starting with no servers

1 server-month

$0 inbound BW

750GB storage

w/ Amazon

Power Efficiency Recommendation

5a. **Invest in research and development to make computer systems more power efficient** at **all levels of the system**

R&D efforts should address ways in which software and system architectures can improve power efficiency, such as by **exploiting locality** and the use of **domain-specific execution units**.

5b. R&D to make the fundamental **logic gate more power-efficient**. Such efforts will need to address alternative physical devices beyond incremental improvements in today's CMOS circuits.

Questions:

Make GPUs more power savvy?

Exploit near-threshold operation?

Practice and Education Recommendations

6. **To promote cooperation and innovation** by sharing and encouraging development of **open interface standard**s for parallel programming.

7. **Invest in the development of tools and methods** to transform legacy applications to parallel systems.

8. **Incorporate in computer science education** an increased emphasis on parallelism, and use a variety of methods and approaches to better prepare students for computing resources that they will encounter in their careers.

Questions:

Where TEACH parallelism?

(a) nowhere, (b) senior/grad, (c) early, (d) pervasive?

Easy parallelism first or only: data parallelism (e.g., map-reduce)?

Summary of Recommendations

Invest in:

1. Algorithms to exploit parallel processing
2. Programming methods to enable efficient use of parallel systems
3. Long-term efforts on rethinking of the canonical computing "stack"
4. Parallel architectures driven by applications
   - enhancements of chip multiprocessor systems
   - data-parallel architectures
   - application-specific architectures
   - radically different approaches
5. Make computer systems more power efficient
6. Cooperation & innovation of open interfaces for parallel programming
7. Tools and methods to transform legacy apps to parallel systems
8. Increased emphasis on parallelism in computer science education

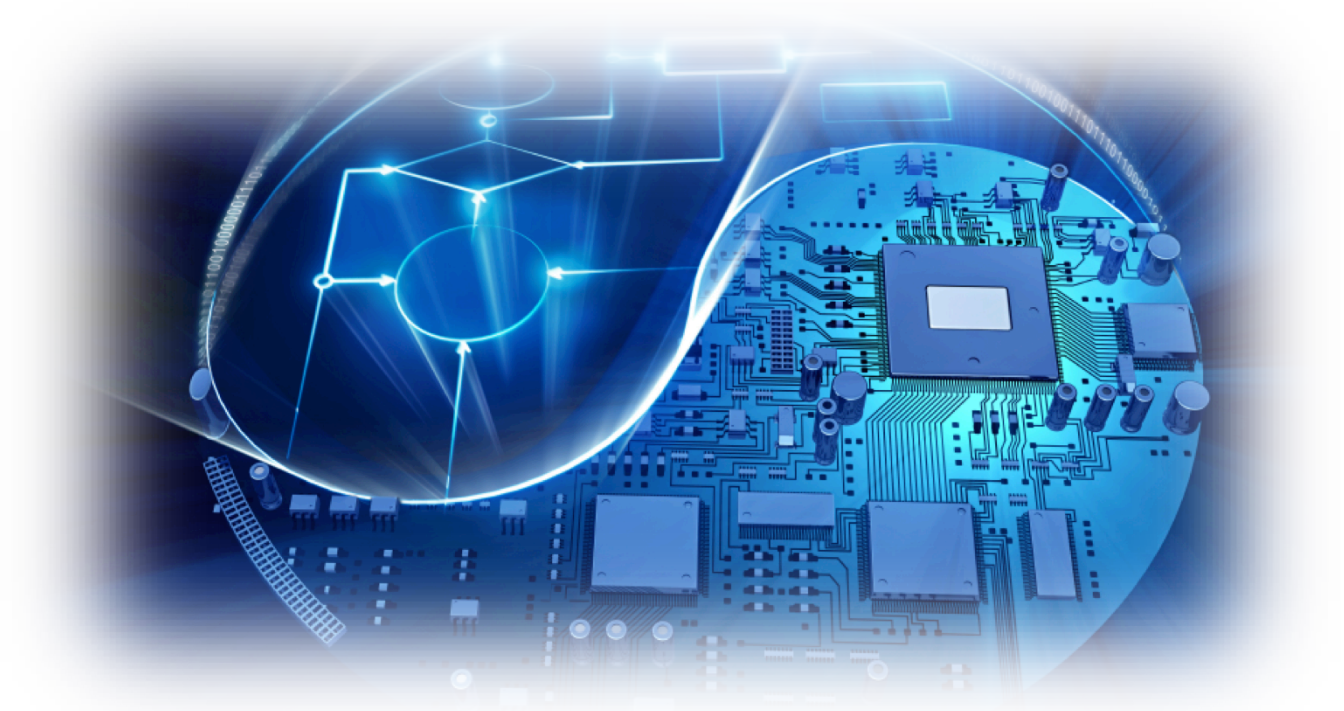
Questions:

Enough to do research within one layer?

Find ways to INNOVATE across layers?

Game Over or Next Level?

This Is A Golden Time for Innovation
in Computing Architectures and Software



The Future of Computing Performance:

Game Over or Next Level?

Computer Science and Telecommunications Board (CSTB)
National Research Council (NRC)

THE NATIONAL ACADEMIES

Challenges in Software Development for Parallelism

- Parallelism introduces new problems in testing and debugging: program behavior can depend on the scheduling of different processes.

- Finding a way to allow performance-increasing reuse of the large software-code base that has been developed over many years would be a huge success.
  - Automatic parallelization has some (very limited) successes,
  - But, most programs, once coded sequentially, have many data dependences that prevent automatic parallelization.

Executive summary (Added to NA Slides)

- Highlights of National Academy Findings
  (F1) Computer hardware has transitioned to multicore
  (F2) Dennard scaling of CMOS has broken down
  (F3) Parallelism and locality must be exploited by software
  (F4) Chip power will soon limit multicore scaling
- Eight recommendations from algorithms to education

- We know all of this at some level, BUT:

Are we all acting on this knowledge or hoping for business as usual?

Thinking beyond next paper to where future value will be created?

  - Questions Asked but Not Answered Embedded in NA Talk
  - Briefly Close with Kübler-Ross Stages of Grief …

Kübler-Ross Stages of Grief (Added to NA Slides) (http://changingminds.org/disciplines/change_management/kubler_ross/kubler_ross.htm)

1. Denial stage: Trying to avoid the inevitable.
2. Anger stage: Frustrated outpouring of bottled-up emotion.
3. Bargaining stage: Seeking in vain for a way out.
4. Depression stage: Final realization of the inevitable.
5. Acceptance stage: Finally finding the way forward.

Final Questions:

Regarding the long term (beyond next few papers) …

- ***Does this model apply to your reaction to NA findings?***
- ***What stage are you in?***
- ***What is needed to move to Acceptance?***

BACKUP SLIDES

- The slides that follow provide good data and were used by Mark Horowitz of Stanford used to introduce this panel when this National Academy study was unveiled

Computing in an Energy Constrained World

Mark Horowitz, Bob Dennard, Dan Dobberpuhl,
Kevin Nowka, Partha Ranganathan

History: The Triple Play

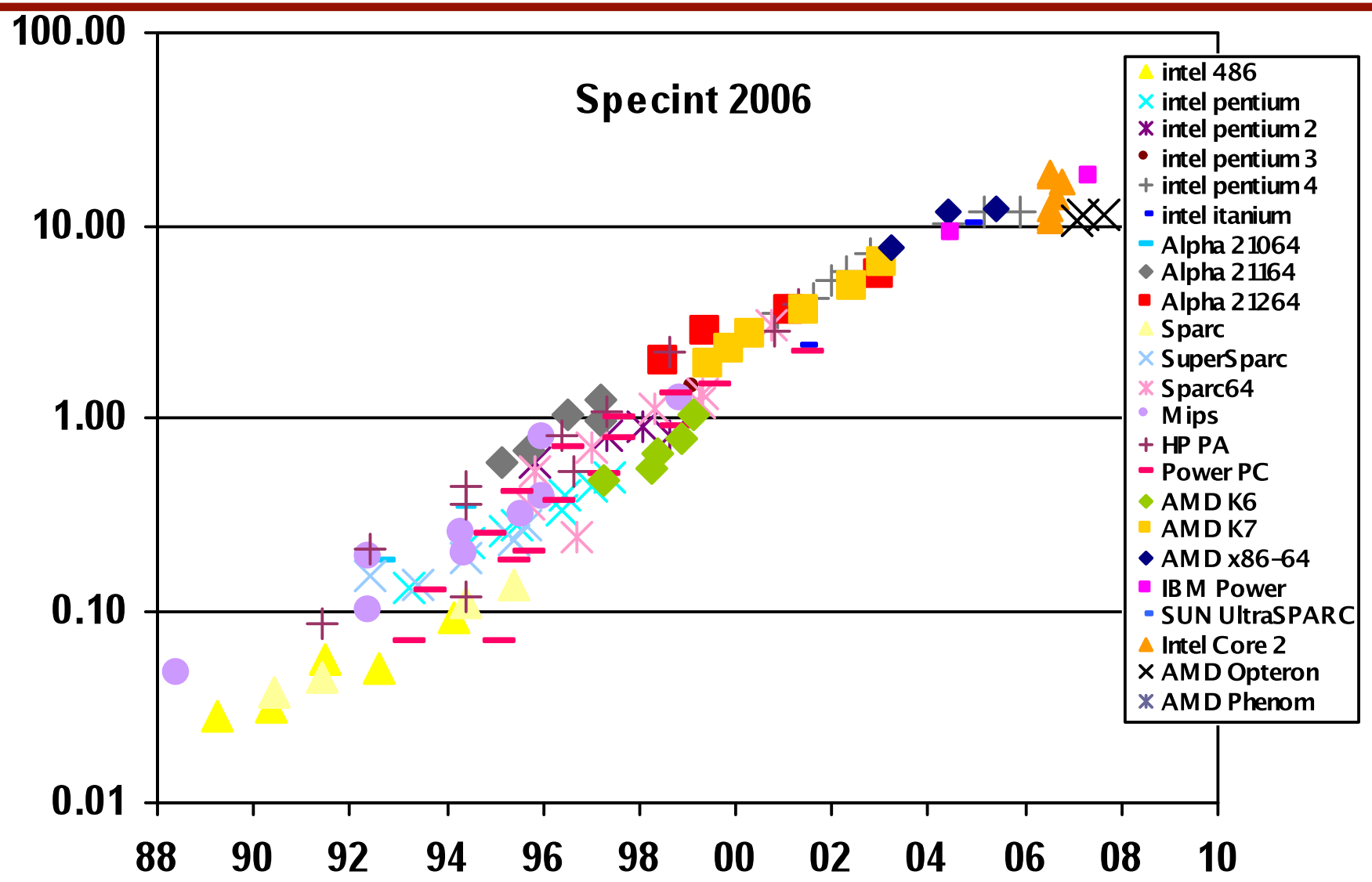
Using Dennard scaling rules

- Get more transistors, gates, 1/α^2
- Gates get faster, delay scales as α
- Energy per switch is reduced α^3

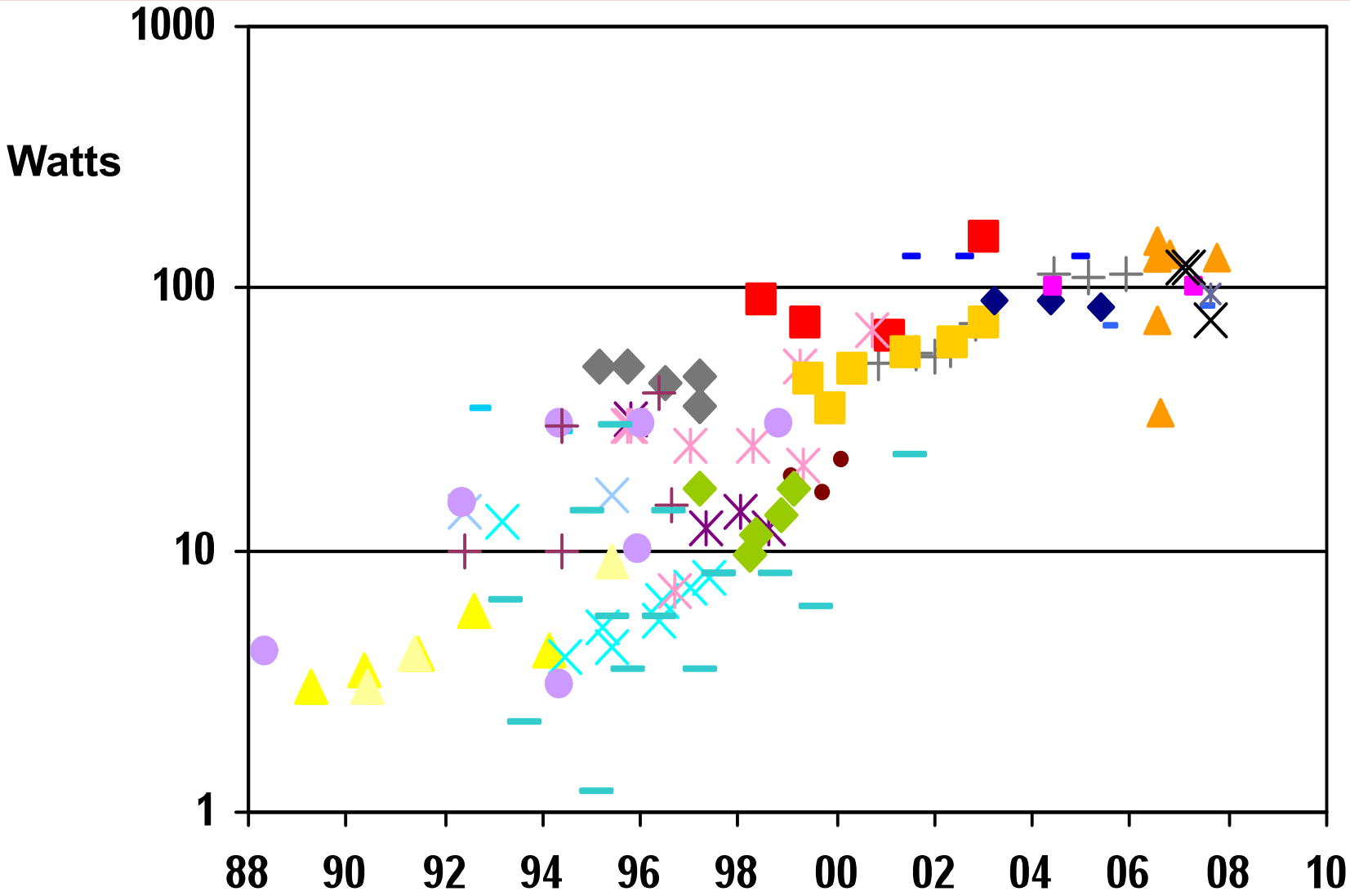
So we can compute 1/α^3 as many gate evals/sec

- At the same power and area as the previous design
- Architects take this to improve computer performance

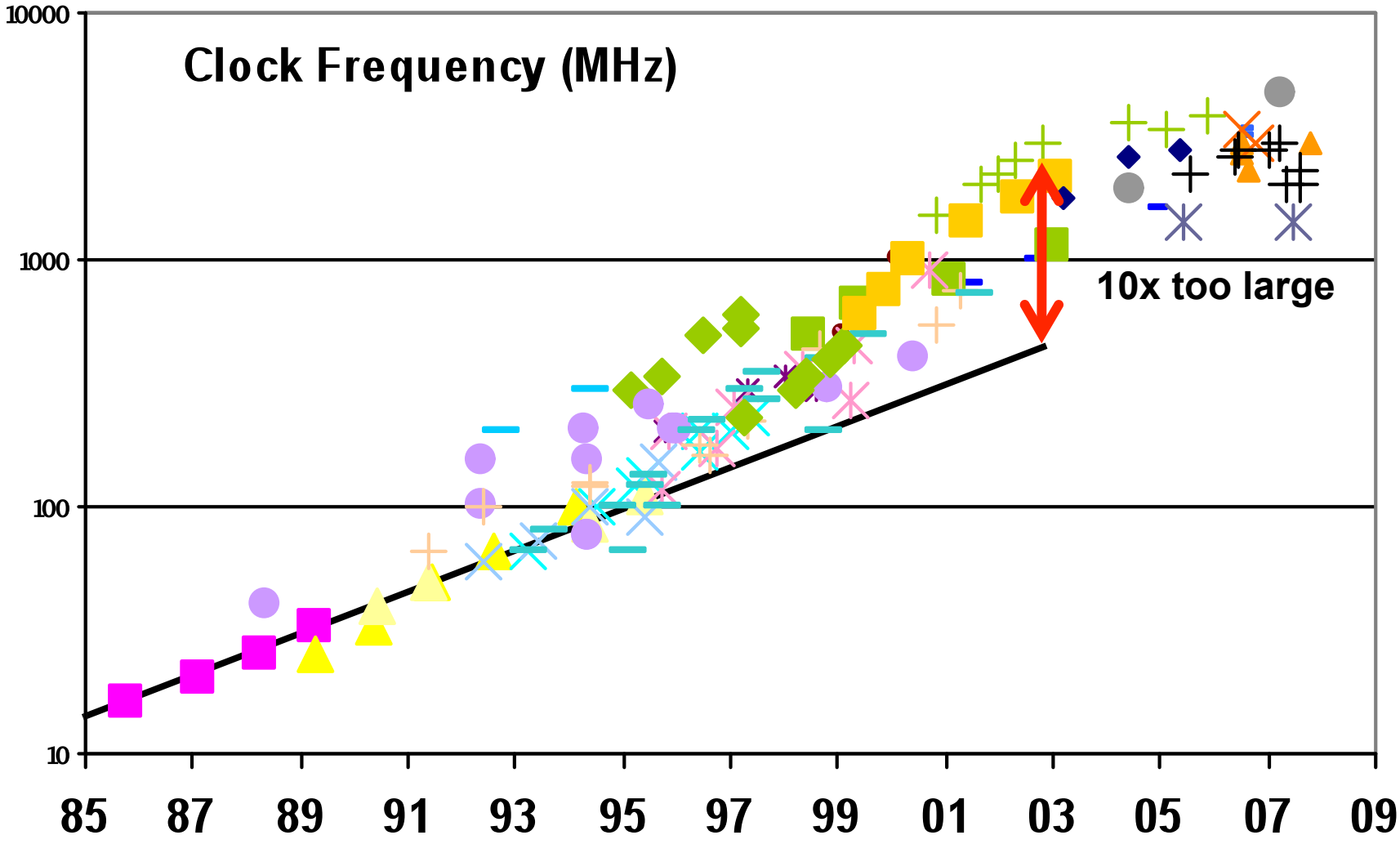
Or At Least We Used To



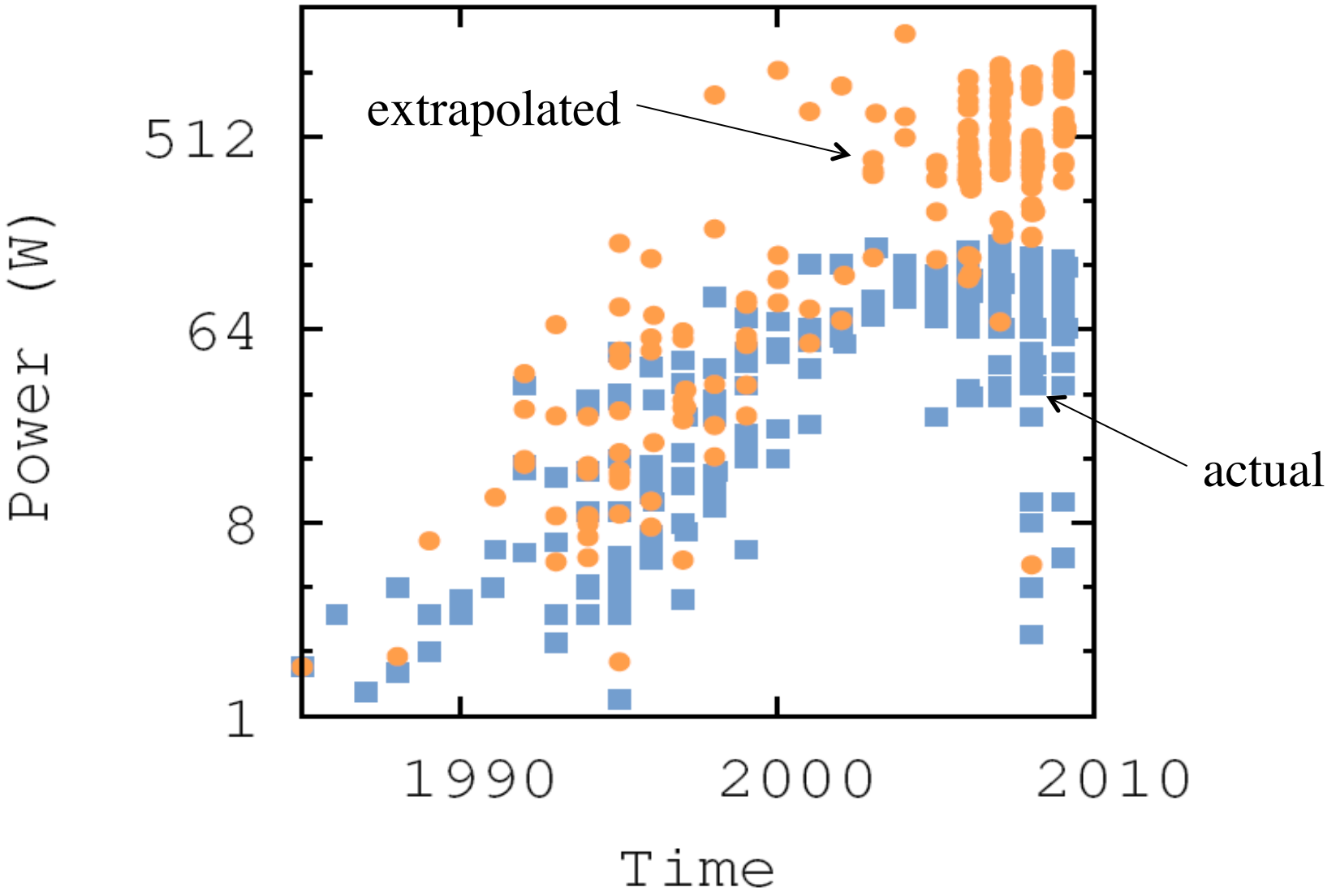
The Power Limit



Power Increased Because We Were ~~Greedy~~ Clever



Circuit Designers Were Working Hard Too



It is Simple Math

$$Power = Energy_{Op} \times \frac{Ops}{second}$$

The Push For Parallelism