



Key Challenges in High Performance Computing

Bhanu Kapoor, Ph.D.

Consultant/Owner, Mimasic

bkapoor@mimasic.com

July 19, 2011, Chitkara University

Punjab, India

mimasic

ISSCC, Feb. 2001, Keynote

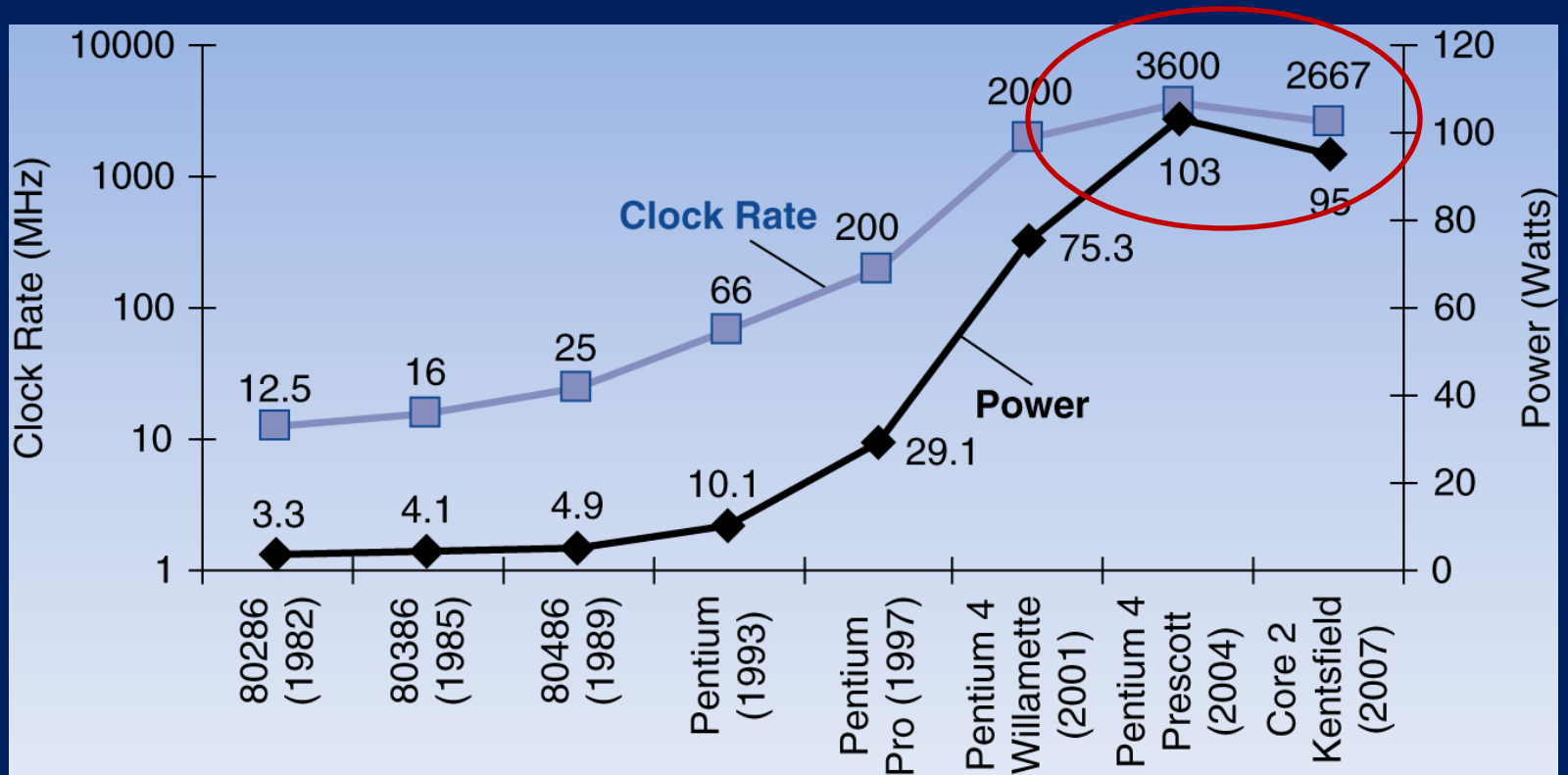


Patrick P. Gelsinger
Senior Vice President
General Manager
Digital Enterprise Group
INTEL CORP.

“Ten years from now, microprocessors will run at 10GHz to 30GHz and be capable of processing 1 trillion operations per second -- about the same number of calculations that the world's fastest supercomputer can perform now.

“Unfortunately, if nothing changes these chips will produce as much heat, for their proportional size, as a nuclear reactor. . . .”

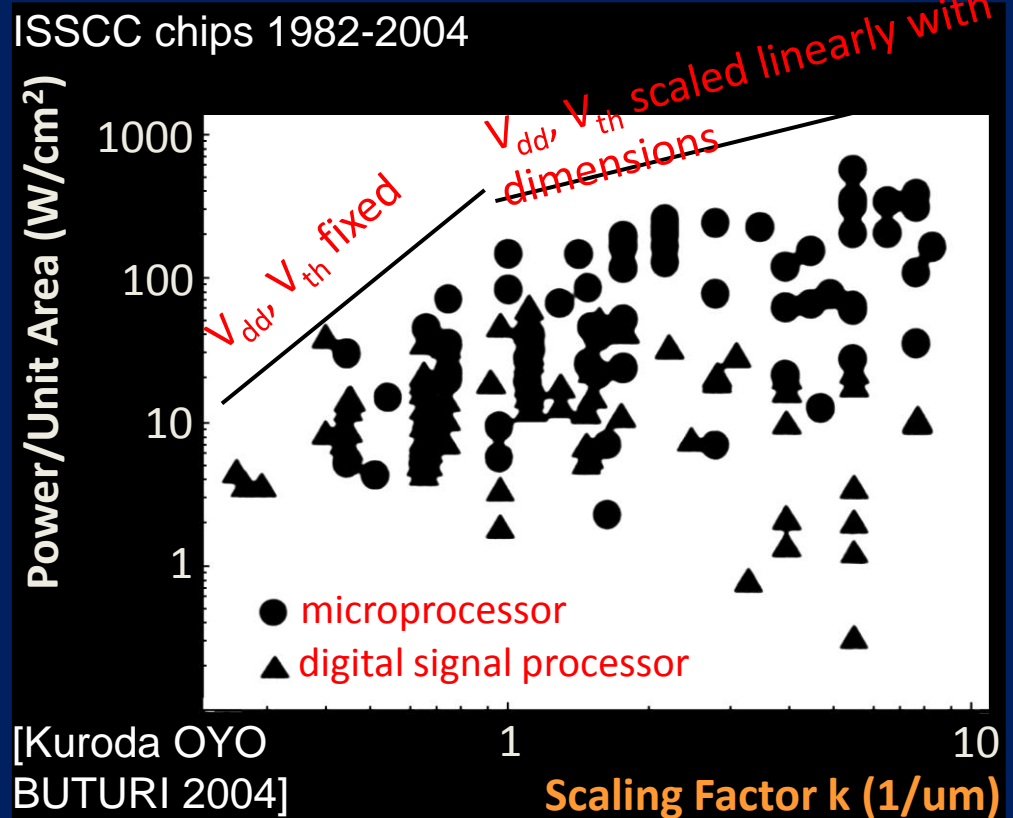
Power Trends



D. Patterson and J. Hennessey
Computer Organization

Increasing Power Density

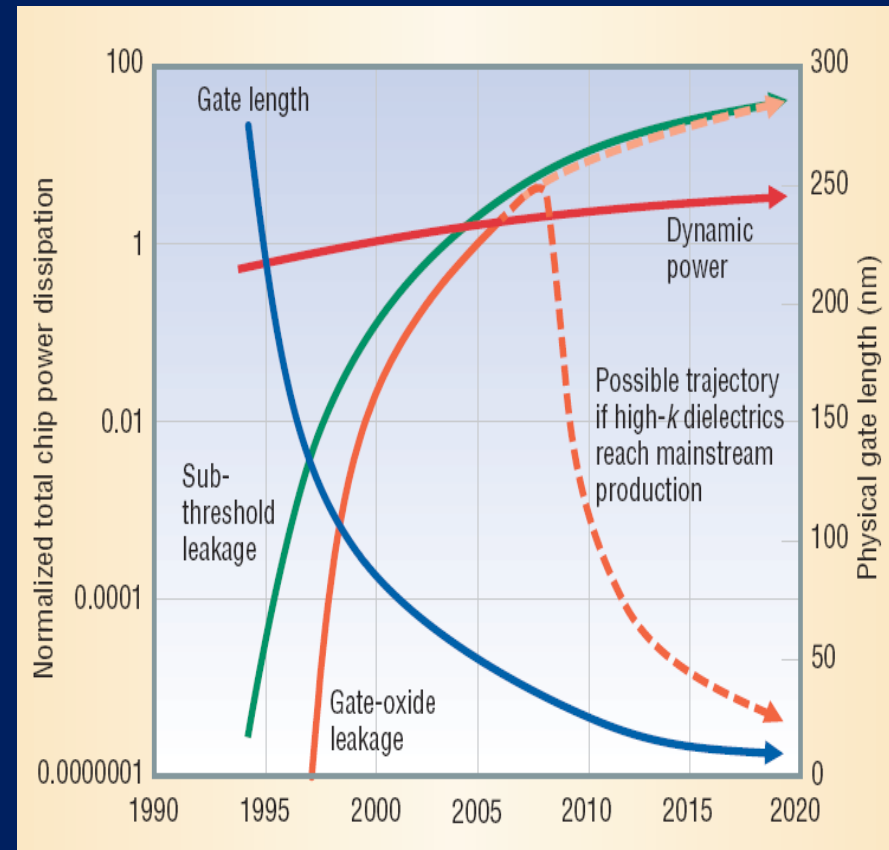
- As device dimensions (W, L, Tox) scaled down by a factor k, for high performance
- V_{dd} and V_{th} fixed \rightarrow power/unit area $\propto k^3$
- V_{dd} and V_{th} scaled down linearly \rightarrow power/unit area $\propto k^{0.7}$



Chinnery and Keutzer, DAC 2005

Power Trends

- **Power:** Dynamic , Leakage
- **Dynamic:** Activity, Frequency, Capacitance, Supply Voltage (V)
- Dynamic Power $\propto V^2$
- **Leakage :** Sub-threshold , Gate-Tunneling
- Leakage Power $\propto V, e^{-Vt}$
- **Manage:** Dynamic, Active Leakage, Standby Leakage

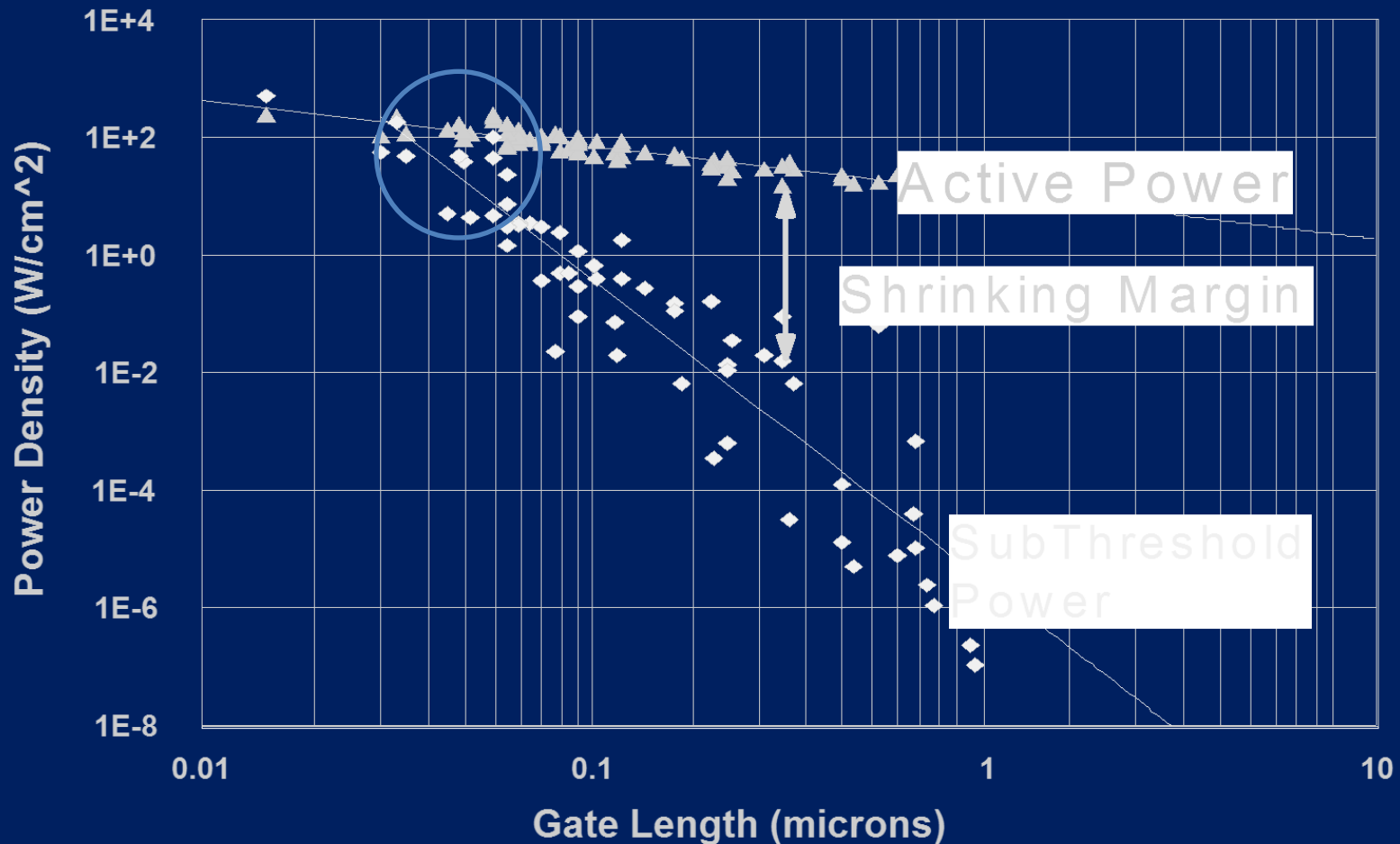


Power Trends, IEEE
D&T, 2003

Scaling in Recent Past

- Transistor dimensions are scaled by 30% (0.7x)
- Area shrinks by 50%, transistor density 2x
- Scaled transistor's delay 0.7x, 1.4x frequency
- Constant field scaling => voltage reduction
- Supply voltage scaled down by 30%
- Dynamic power reduces by 50%
- Leakage Power = Supply Voltage X L. Current
- Leakage Current growing exponentially

So, What's Going On ?



- At 65nm node Static Power is equal to Active Power
 - Clock distribution accounts for half of active power

Increasing Parallelism/ Concurrency

- Chandrakasan first showed that concurrency can be used to reduce power instead of increasing performance
- Primary idea is to reduce the frequency of operation and/or voltage to meet a certain throughput
- Power consumed by additional logic required to distribute computation and multiplex results needs to be accounted for

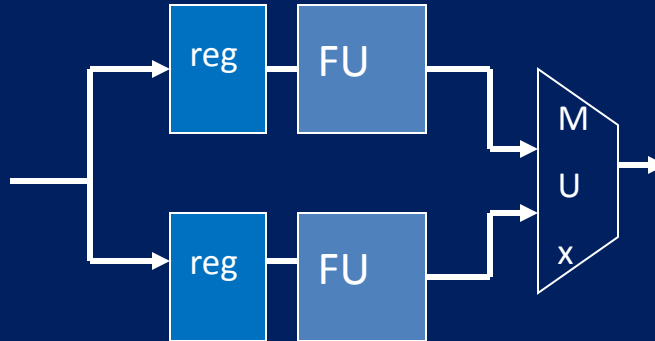
Effect of Parallelism

Case1:
Single FU



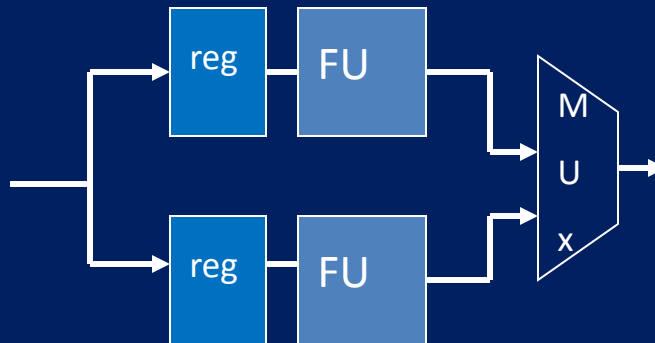
freq: f_0
voltage: v_0
throughput: T_0

Case2:
Two FUs for
enhanced
performance



$f_1 = f_0$
 $v_1 = v_0$
 $T_1 > T_0$

Case 3:
Two FUs for
reducing
power



$f_2 < f_0$
 $v_2 < v_0$
 $T_2 = T_0$

Today



2nd Generation Intel® Core™ i7 Mobile Processor Extreme Edition

Print

Processor Number	Cache	Clock Speed	# of Cores / # of Threads	Max TDP	Memory Types	Intel® HD Graphics
32 nm						
i7-2920XM	8.0 MB	2.50 GHz	4 / 8	55 W	DDR3-1066/1333/1600	✓

Multicore in Products

- “We are dedicating all of our future product development to multicore designs. ... This is a sea change in computing”

Paul Otellini, President, Intel (2005)

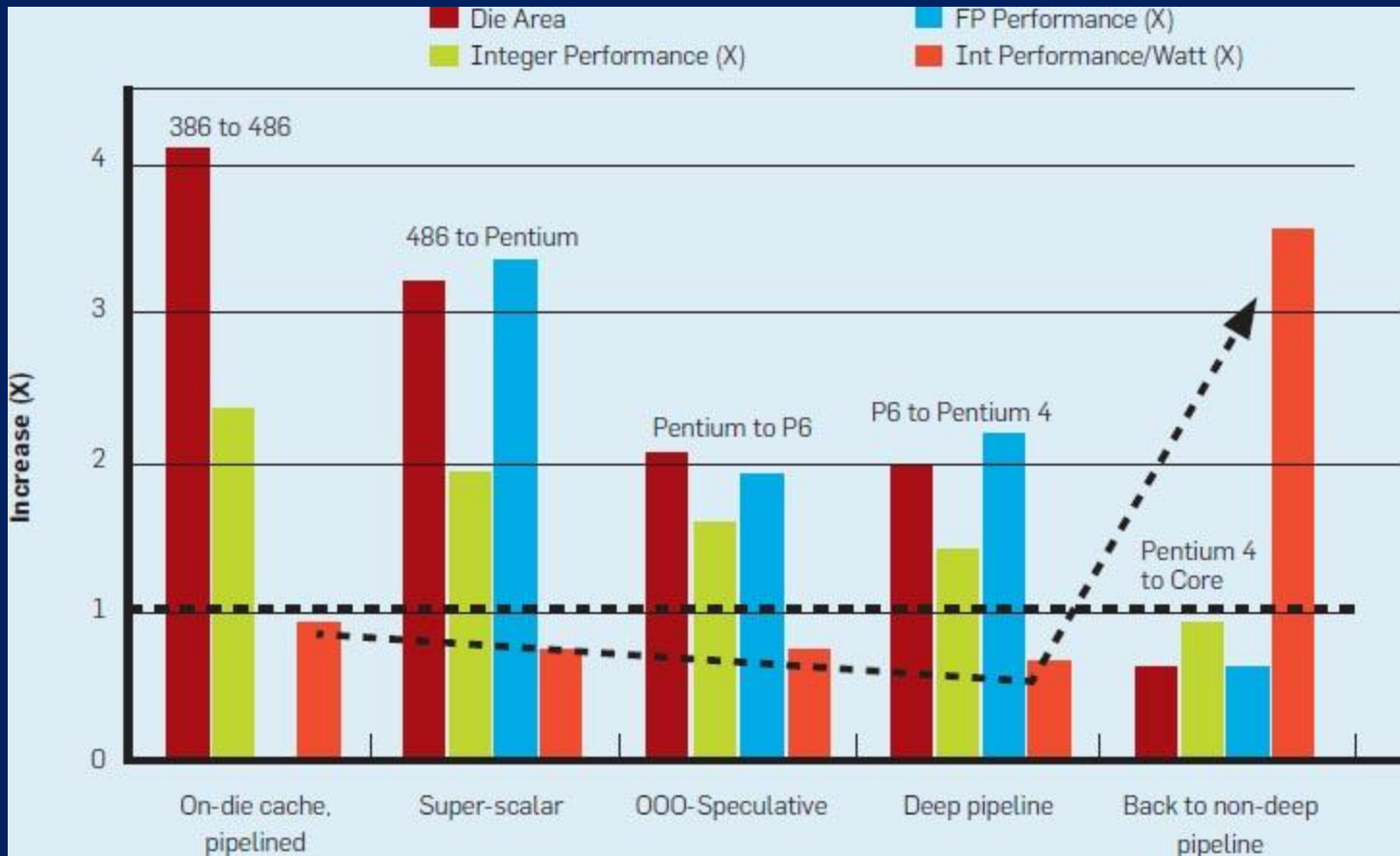
- All microprocessor companies switch to MP (2X CPUs / 2 yrs)
⇒ Procrastination penalized: 2X sequential perf. / 5 yrs

Manufacturer/Year	AMD/'05	Intel/'06	IBM/'04	Sun/'07
Processors/chip	2	2	2	8
Threads/Processor	1	2	2	16
Threads/chip	2	4	4	128

And at the same time,

- The STI Cell processor (PS3) has 8 cores
- The latest NVidia Graphics Processing Unit (GPU) has 128 cores
- Intel has demonstrated an 80-core research chip

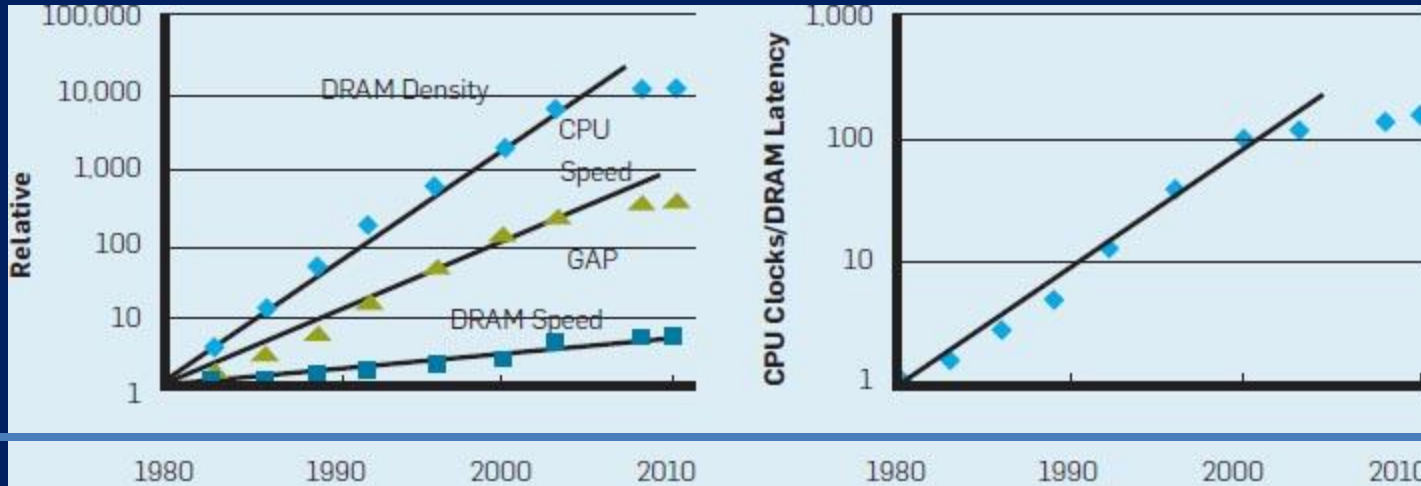
Impact of Architecture on Power



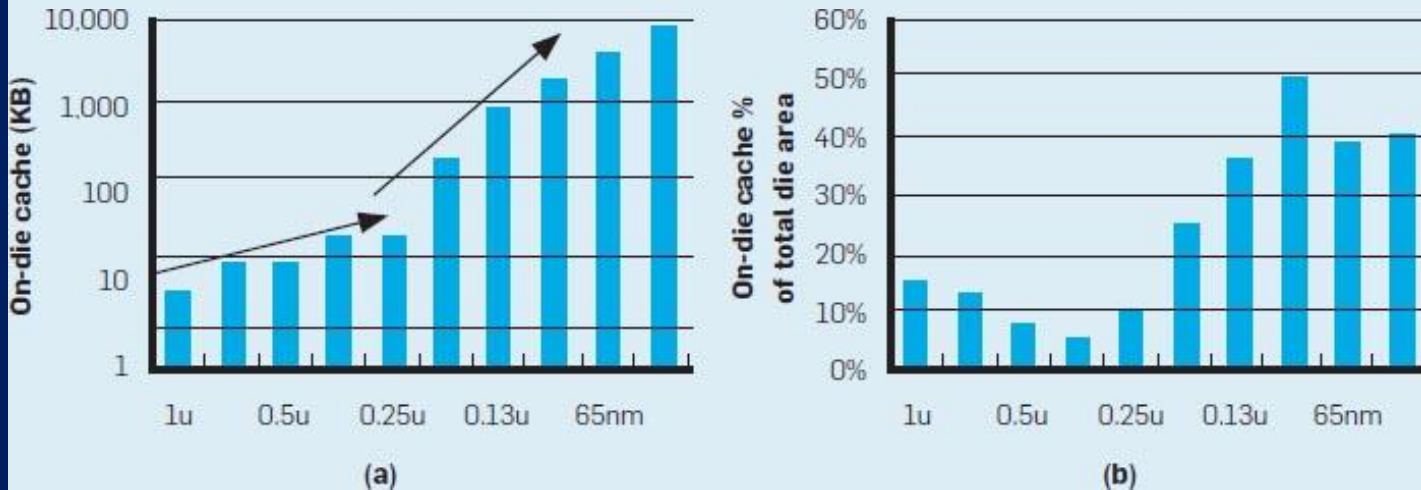
Source: Borkar, Intel®

Memory Advances

DRAM

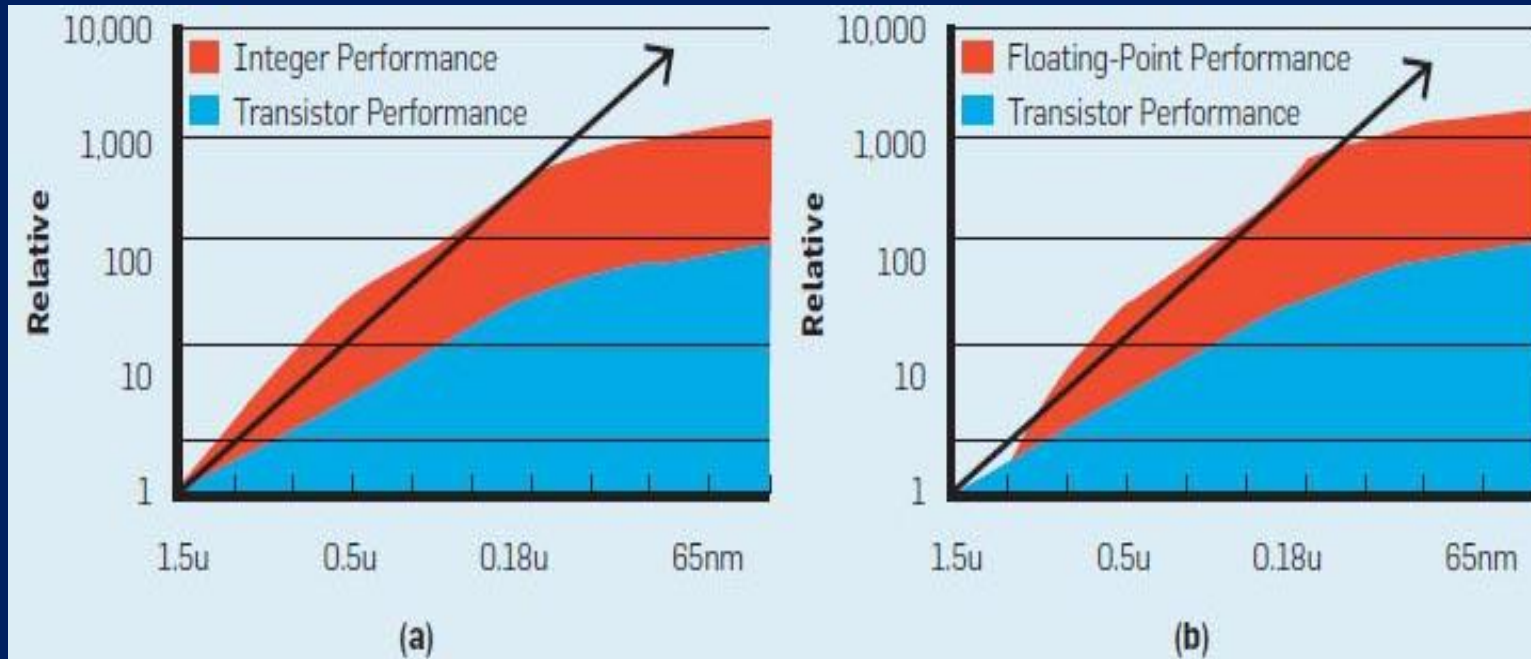


Cache



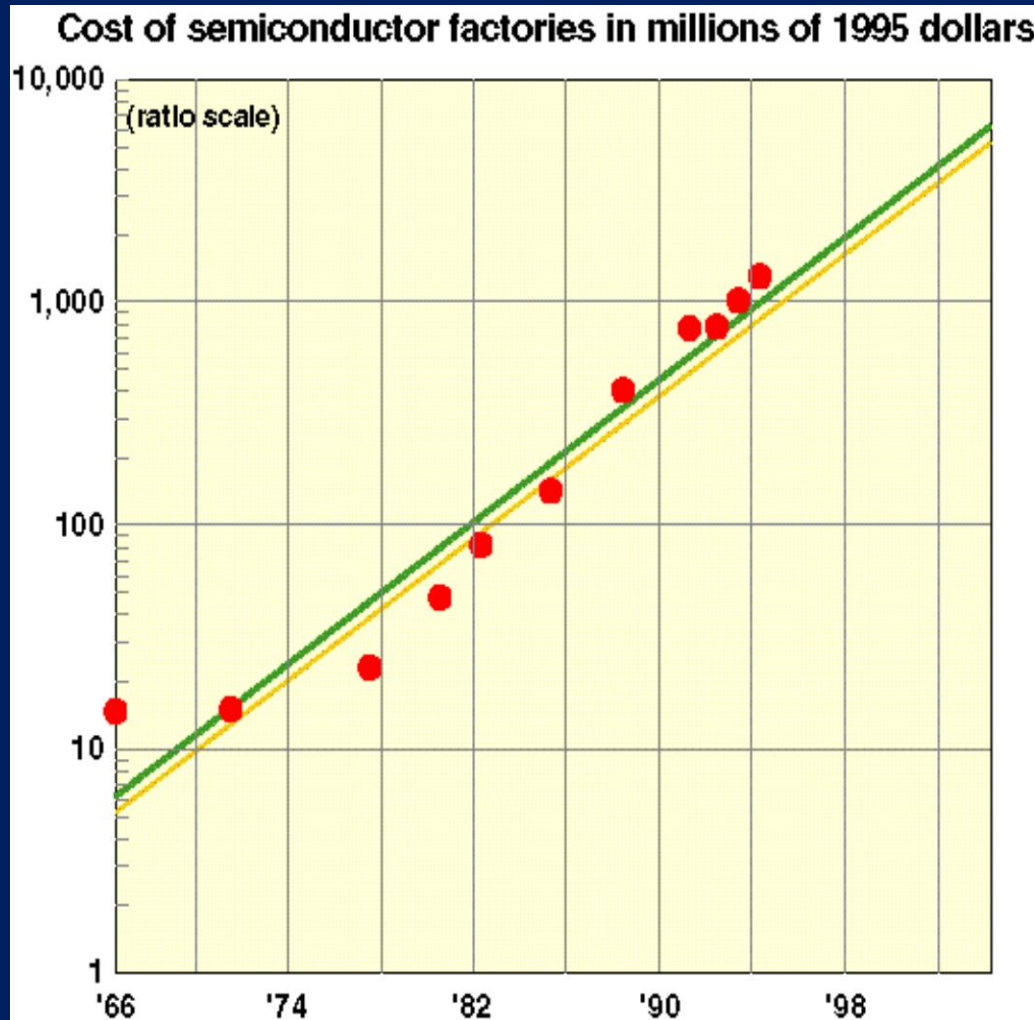
Source: Borkar, Intel®

Where has performance gains come from?



Source: Borkar, Intel®

Reasons for Parallel Computing : Chip Yield



- Moore's (Rock's) 2nd law: fabrication costs go up
- Yield (% usable chips) drops
- Parallelism can help
 - More smaller, simpler processors are easier to design and validate
 - Can use partially working chips:
 - E.g., Cell processor (PS3) is sold with 7 out of 8 "on" to improve yield

Reasons for Parallel Computing: Speed of Light (Fundamental)

1 Tflop/s, 1 Tbyte
sequential machine



$r = 0.3 \text{ mm}$

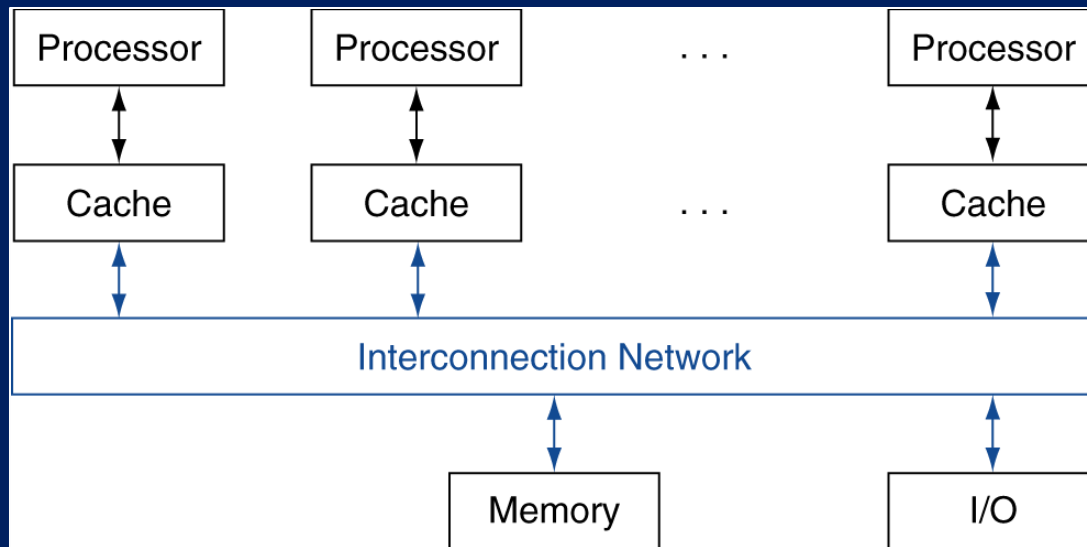
- Consider the 1 Tflop/s sequential machine:
 - Data must travel some distance, r , to get from memory to CPU.
 - To get 1 data element per cycle, this means 10^{12} times per second at the speed of light, $c = 3 \times 10^8 \text{ m/s}$. Thus $r < c/10^{12} = 0.3 \text{ mm}$.
- Now put 1 Tbyte of storage in a $0.3 \text{ mm} \times 0.3 \text{ mm}$ area:
 - Each bit occupies about 1 square Angstrom, or the size of a small atom.
- No choice but parallelism

Parallel Computing Challenges: Amdahl's Law

- Sequential part can limit speedup
- Example: 100 processors, 90× speedup?
 - $T_{\text{new}} = T_{\text{parallelizable}}/100 + T_{\text{sequential}}$
 - $\text{Speedup} = \frac{1}{(1 - F_{\text{parallelizable}}) + F_{\text{parallelizable}}/100} = 90$
 - Solving: $F_{\text{parallelizable}} = 0.999$
- Need sequential part to be 0.1% of original time
- A 40% parallel program can never be sped by more than a factor of 1.7!

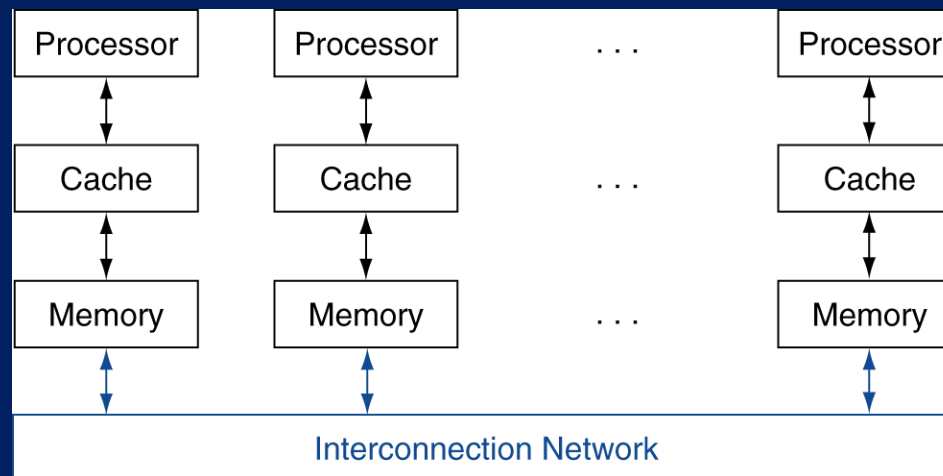
Parallel Computing Challenges: Shared Memory Processor Programming

- SMP: shared memory multiprocessor
 - Hardware provides single physical address space for all processors
 - Synchronize shared variables using locks
 - OpenMP, Pthreads, CUDA



Parallel Computing Challenges: Programming Message Passing Architecture

- Each processor has private physical address space
- Hardware sends/receives messages between processors
- Programming: MPI



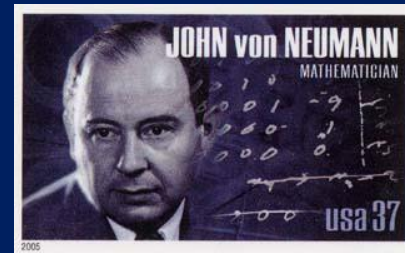
Why is Concurrency Difficult?

- People think sequentially
 - partial orders problematic
 - arbitrary interleavings overlooked
 - worst case scenarios too complex to reason about
- Lack of agreement on programming model
 - shared memory, message passing, SIMD, MIMD, data parallel, functional, ...
- Common abstractions are low level primitives
 - threads and explicit synchronization == assembly language
- Poorly supported by languages and tools
 - libraries with threads and synchronization are not the answer
- Concurrent programs provably harder to analyze
 - *Context-sensitive Synchronization-sensitive Analysis is Undecidable*
[Ramalingam 2000]

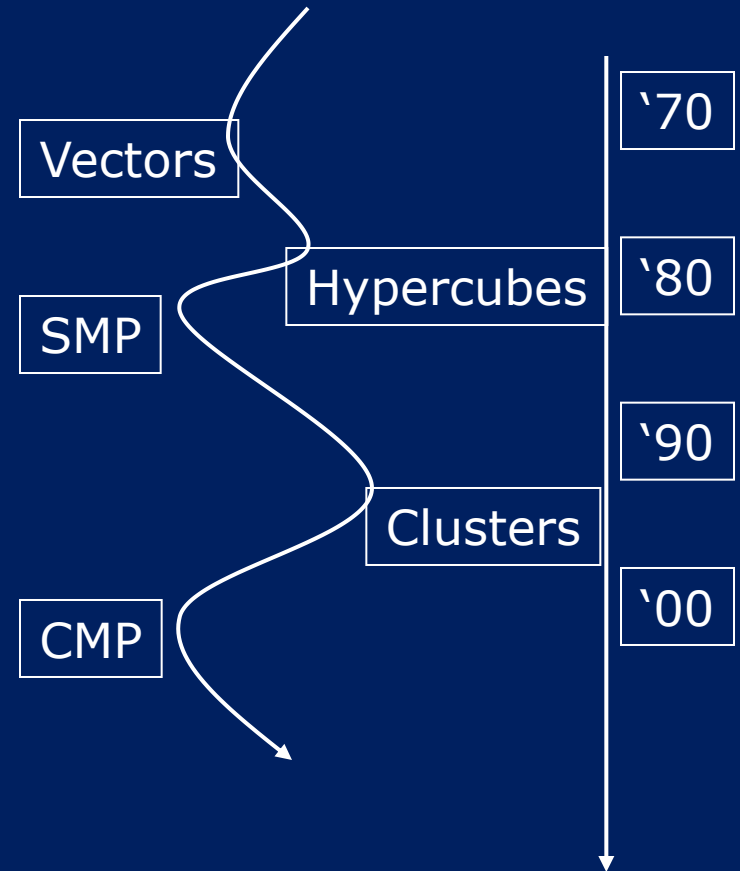
James Larus
Microsoft Research

mimasic

Lack a Common Parallel Programming Model



- Computing without John Von Neumann!
 - threads vs event-driven
 - shared memory vs message passing
 - SIMD vs MIMD
 - data parallelism, actors, dataflow, ...
- Continually reinventing the wheel
 - education & training
 - programming languages & tools
 - poor libraries
 - lack of software reuse



James Larus
Microsoft Research

Successful Applications of Concurrency

- Success \Rightarrow normal mode
 - routinely used to write software
 - profitably build parallel hardware for domain
- Server
 - stream of (mostly) independent requests
- Desktop
 - graphics
- Scientific
 - “emerging”

James Larus
Microsoft Research

Graphics is Naturally Concurrent

- Abundance of “embarrassingly parallel tasks”
 - lots of vertices, pixels, frames, ...
 - increased computation resources enables higher resolution, which requires more computation, ...
 - operations have few data or control dependencies
- Practical programming model
 - abstract datatypes (points, lines, triangles, polygons, light sources, ...)
 - data parallel operations
- Decouple programming model from hardware
 - video cards from on 6-month release cycle
 - game programmers do not care (except at leading edge)

James Larus
Microsoft Research

Single Concurrent Model Unlikely

- Maybe three:
 - shared memory
 - message passing
 - data parallelism
- Can they co-exist and inter-operate?
- Can we teach developers when each is appropriate?

James Larus
Microsoft Research

HW Solution: Small is Beautiful

- Expect modestly pipelined (5- to 9-stage) CPUs, FPUs, vector, SIMD PEs
 - Small cores not much slower than large cores
- Parallel is energy efficient path to performance: CV^2F
 - 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 CPUs
- Small, regular processing elements easier to verify
- One size fits all?
 - Amdahl's Law \Rightarrow Heterogeneous processors
 - Special function units to accelerate popular functions

Berkeley Par Lab

Energy Efficient Software

- Parallel programming difficult
- Should applications scale voltage along with?
- Accurate Power/Energy models are difficult to create
- This can work well in server like scenario i.e., application-based scaling for energy efficiency

Cell Phone Processors: Infineon Baseband Processor – ISSCC06

Application	Settings	ARM926 MHzX26	DSP MHzX26	Combo*	Peripheral Bus 1	Peripheral Bus 2	Analog	Standby Domain
GSM Voice	6.60-AMR	f1	f2	On	Off	Off	On	On
GSM Voice	6.60AMR+Handsfree	f1	f2a	Off	Off	Off	Off	On
GSM Idle	Sleep Mode	0	0	On	Off	Off	On	On
GSM Idle	Paging	f1	f1	On	Off	Off	On	On
Data Down.	HSDPA 3.6Mbps	f4	0	On	On	Off	Off	On
Data Down.	UMTS PS 384 kbps	f2	0	On	On	Off	Off	On
Data Down.	E-GPRS	f2	f3	On	On	Off	On	On
Video Enc.	MPEG-4 20fps QVGA	f5	f2	On	On	On	On	On
Video Tel.	MPEG-4 15fps QCIF	f3	f2b	On	On	On	On	On
Music Repl.	MP3	f1	f1	On	Off	Off	On	On
Music Repl.	MP3 + Paging	f2	f1	On	Off	Off	On	On

Summary

- Parallel Computing was the last resort: Our industry has bet its future on parallelism
- None of us have ever lived in a world in which computers don't run sequential code faster
- Most programs and programmers just entered this world
- We must make concurrency usable
 - programming models that are accessible and generally applicable

