The Why, Where and How of Multicore

Anant Agarwal

MIT and Tilera Corp.

What is Multicore?

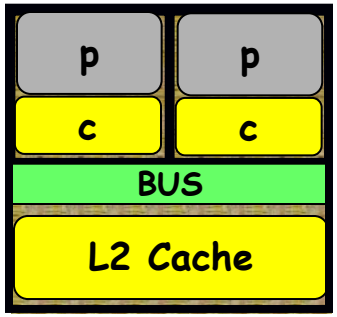
Whatever's "Inside"?

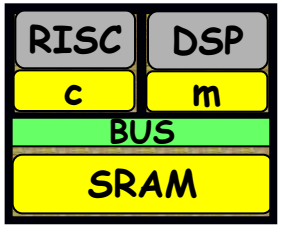
What is Multicore?

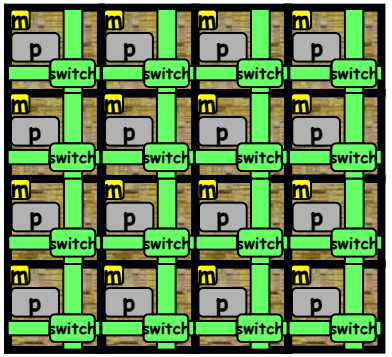
Whatever's "Inside"?

Seriously, multicore satisfies three properties

- Single chip
- Multiple distinct processing engines
- Multiple, independent threads of control (or program counters – MIMD)







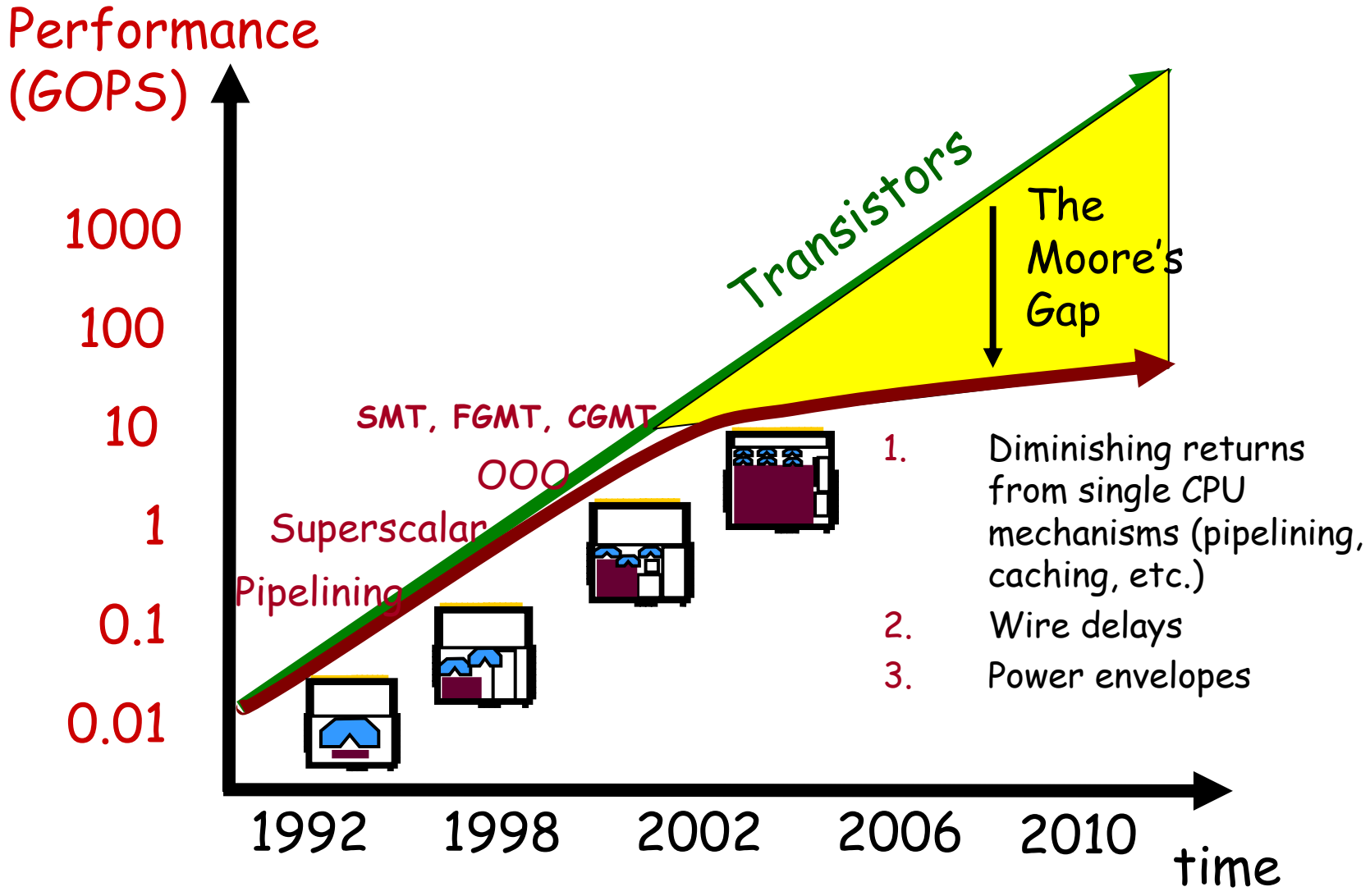
Outline

- The why
- The where
- The how

Outline

- The why
- The where
- The how

The "Moore's Gap"



Houston, we have a problem…

The Moore's Gap – Example

Pentium 3	Pentium 4
1 GHz	1.4 GHz
Year 2000	Year 2000
0.18 micron	0.18 micron
28M transistors	42M transistors
343 (Specint 2000)	393 (Specint 2000)

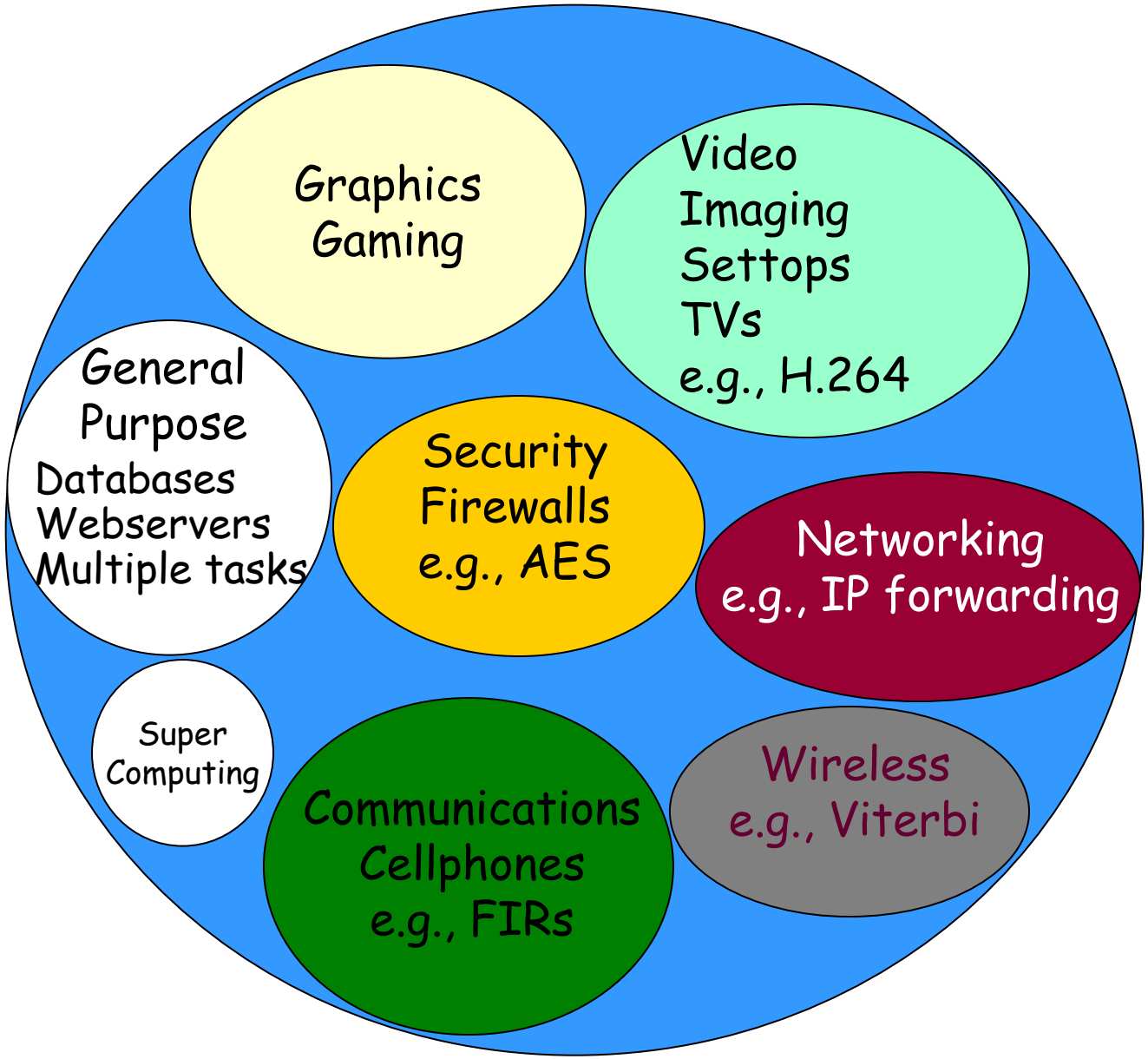
Transistor count increased by 50%
Performance increased by only 15%

Closing Moore's Gap Today

Two things have changed:

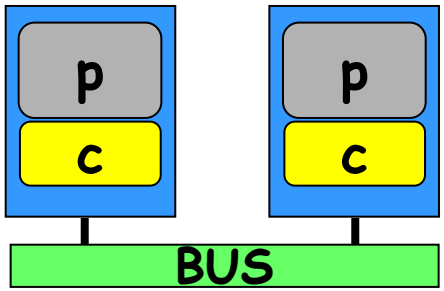
- Today's applications have ample parallelism – *and they are not PowerPoint and Word!*
- Technology: On-chip integration of multiple cores is now possible

Parallelism is Everywhere



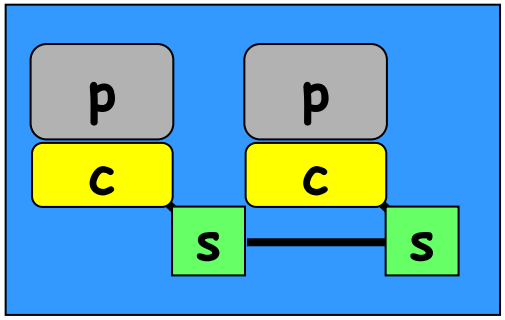
Integration is Efficient

Discrete chips



Bandwidth 2GBps
Latency 60ns
Energy > 500pJ

Multicore



Bandwidth > 40GBps*
Latency < 3ns
Energy < 5pJ

- Parallelism and interconnect efficiency enables harnessing the "power of n"
- n cores yield n-fold increase in performance
- This fact yields the multicore opportunity

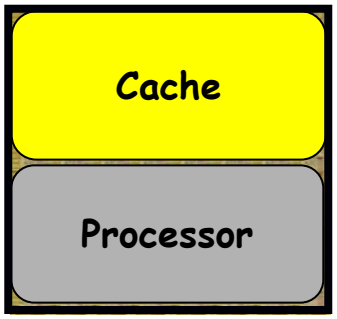
*90nm, 32 bits, 1mm

Why Multicore?

Let's look at the opportunity from two viewpoints

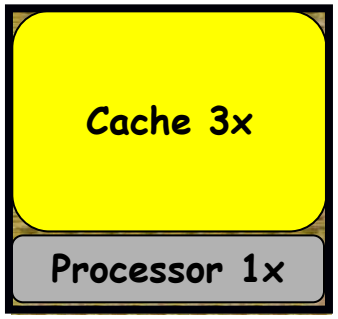
- Performance
- Power efficiency

The Performance Opportunity



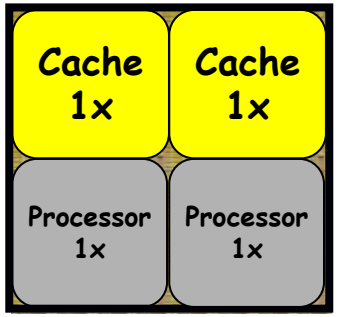
90nm
CPI: 1 + 0.01 x 100 = 2

Push single core



65nm
CPI: 1 + 0.006 x 100 = 1.6

Go multicore

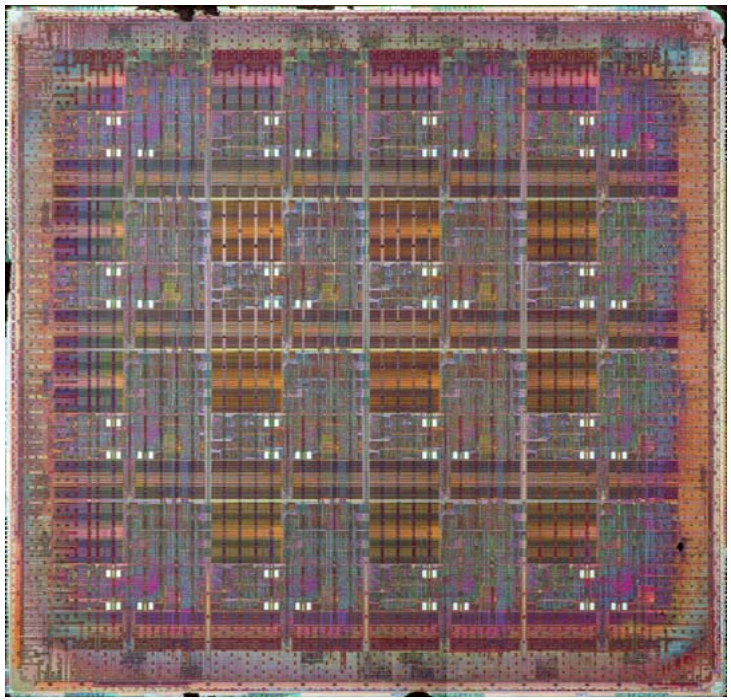


65nm
CPI: (1 + 0.01 x 100)/2 = 1

Smaller CPI (cycles per instruction) is better

Single processor mechanisms yielding diminishing returns
Prefer to build two smaller structures than one very big one

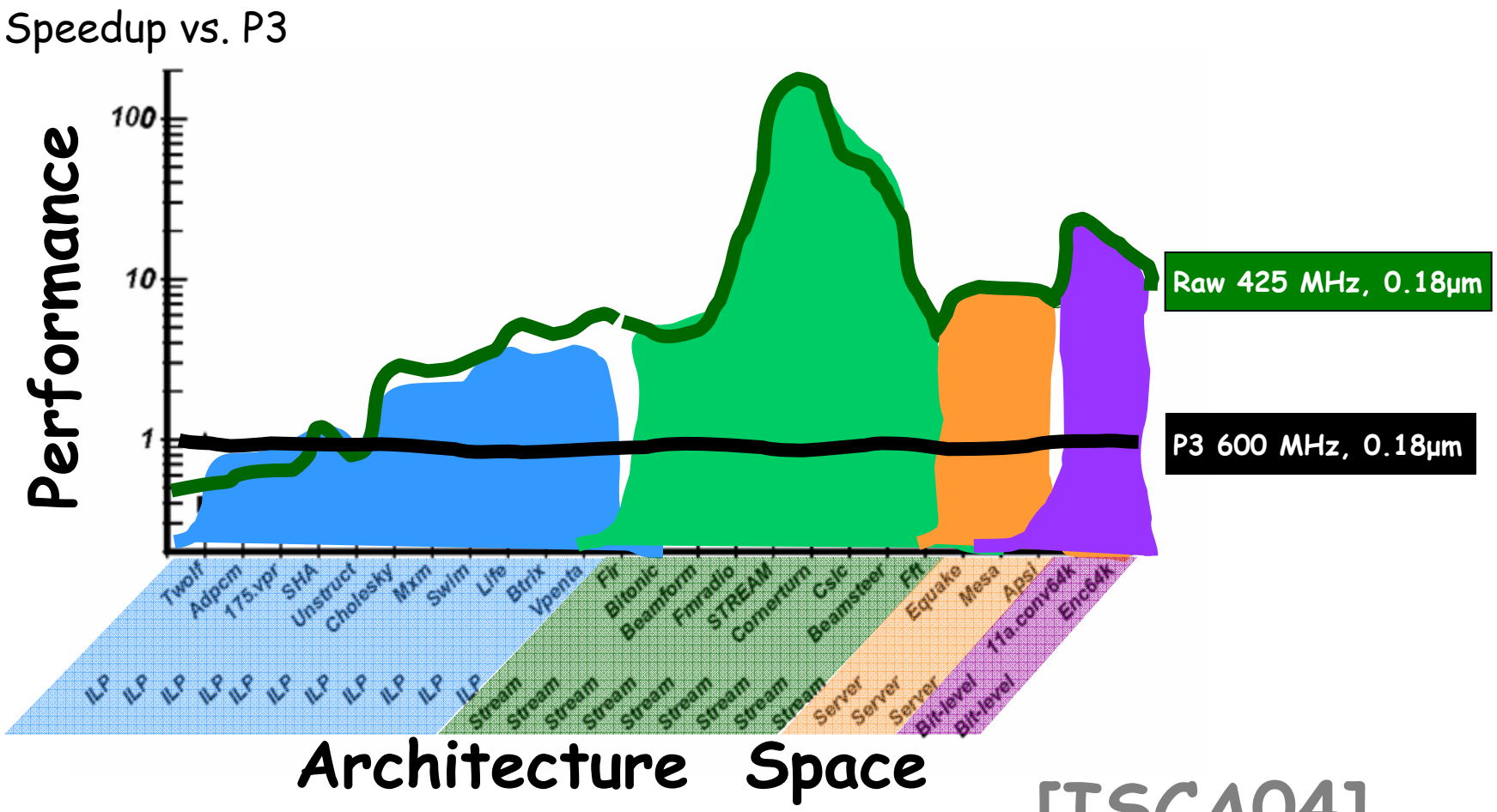
Multicore Example: MIT's Raw Processor



- 16 cores
- Year 2002
- 0.18 micron
- 425 MHz
- IBM SA27E std. cell
- 6.8 GOPS

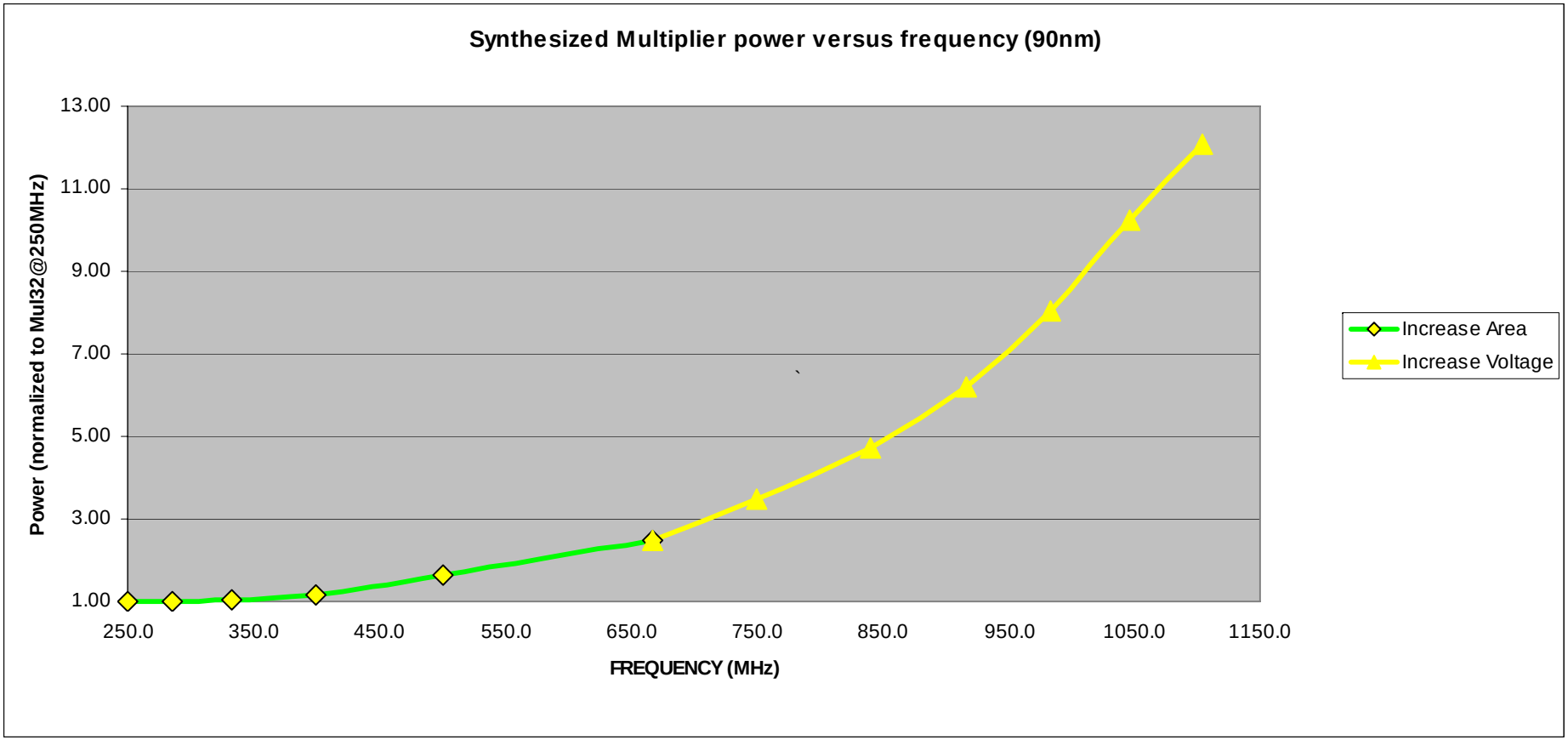
Google MIT Raw

Raw's Multicore Performance



[ISCA04]

The Power Cost of Frequency



Frequency $\propto V$

Power $\propto V^3$ (V^2F)

For a 1% increase in freq, we suffer a 3% increase in power

Multicore's Opportunity for Power Efficiency

	Cores	V	Freq	Perf	Power	PE (Bops/watt)
Superscalar	1	1	1	1	1	1
"New" Superscalar	1X	1.5X	1.5X	1.5X	3.3X	0.45X
Multicore	2X	0.75X	0.75X	1.5X	0.8X	1.88X

(Bigger # is better)

50% more performance with 20% less power

Preferable to use multiple slower devices, than one superfast device

Outline

- The why
- The where
- The how

The Future of Multicore

Number of cores will double every 18 months

	'02	'05	'08	'11	'14
Academia	**16**	**64**	**256**	**1024**	**4096**
Industry	**4**	**16**	**64**	**256**	**1024**

But, wait a minute…

Need to create the "1K multicore" research program ASAP!

Outline

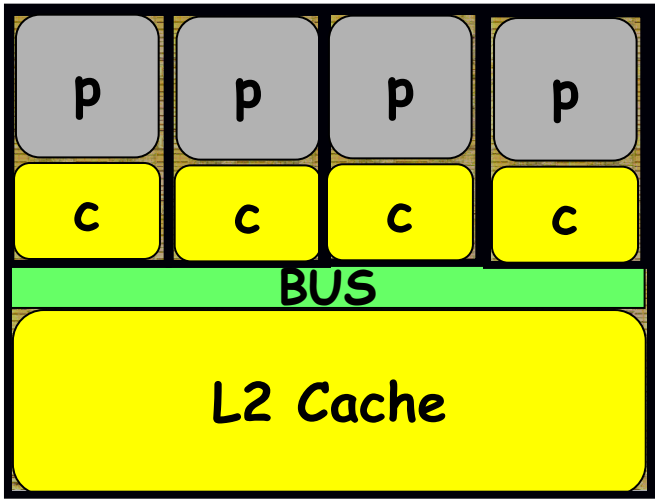
- The why
- The where
- The how

Multicore Challenges
The 3 P's

- Performance challenge
- Power efficiency challenge
- Programming challenge

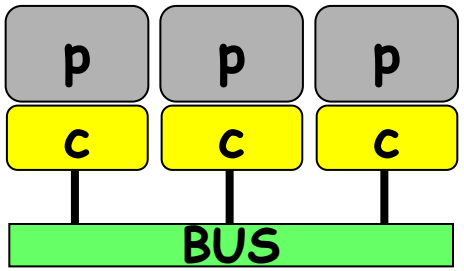
Performance Challenge

- Interconnect
- It is the new mechanism
- Not well understood
- Current systems rely on buses or rings
- Not scalable - will become performance bottleneck
- Bandwidth and latency are the two issues

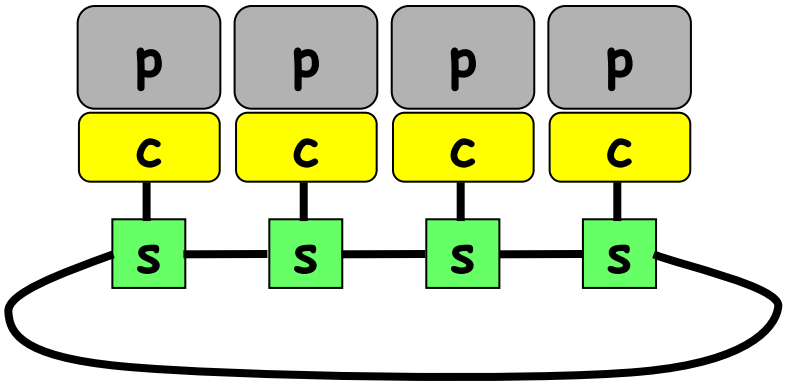


Interconnect Options

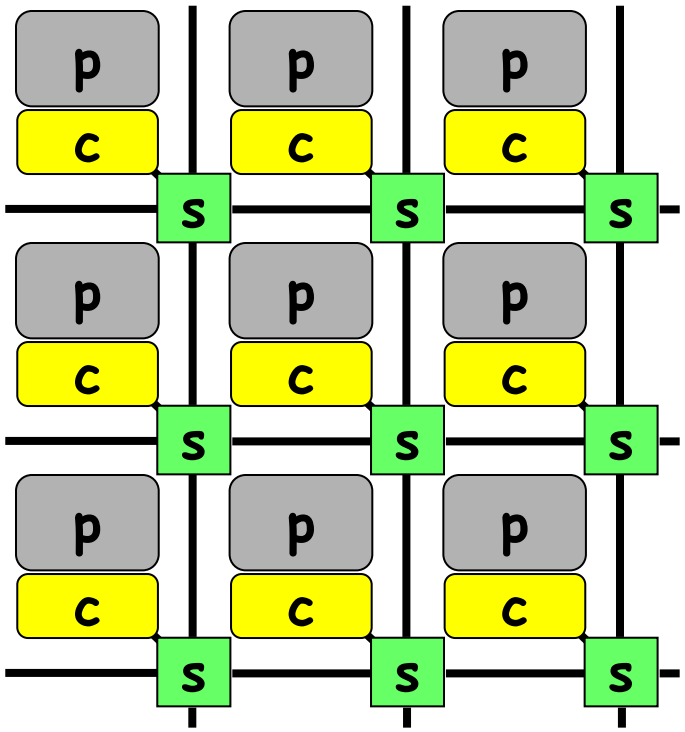
Bus Multicore



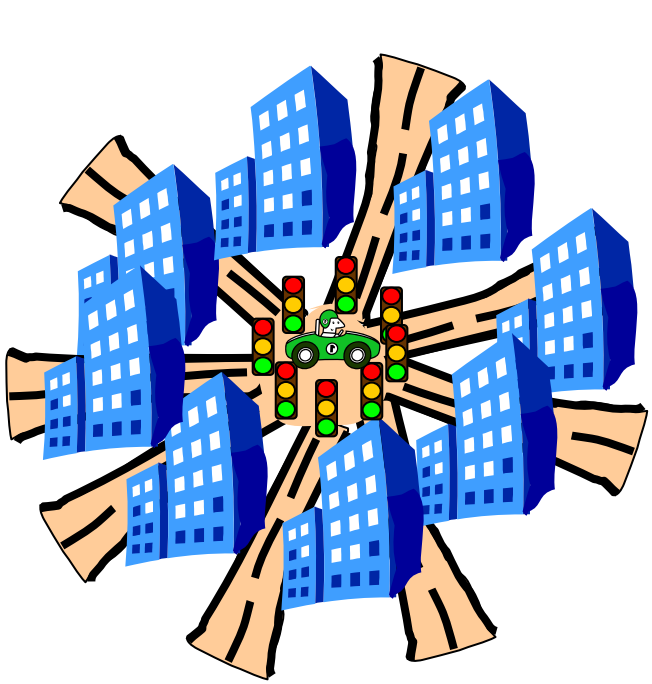
Ring Multicore

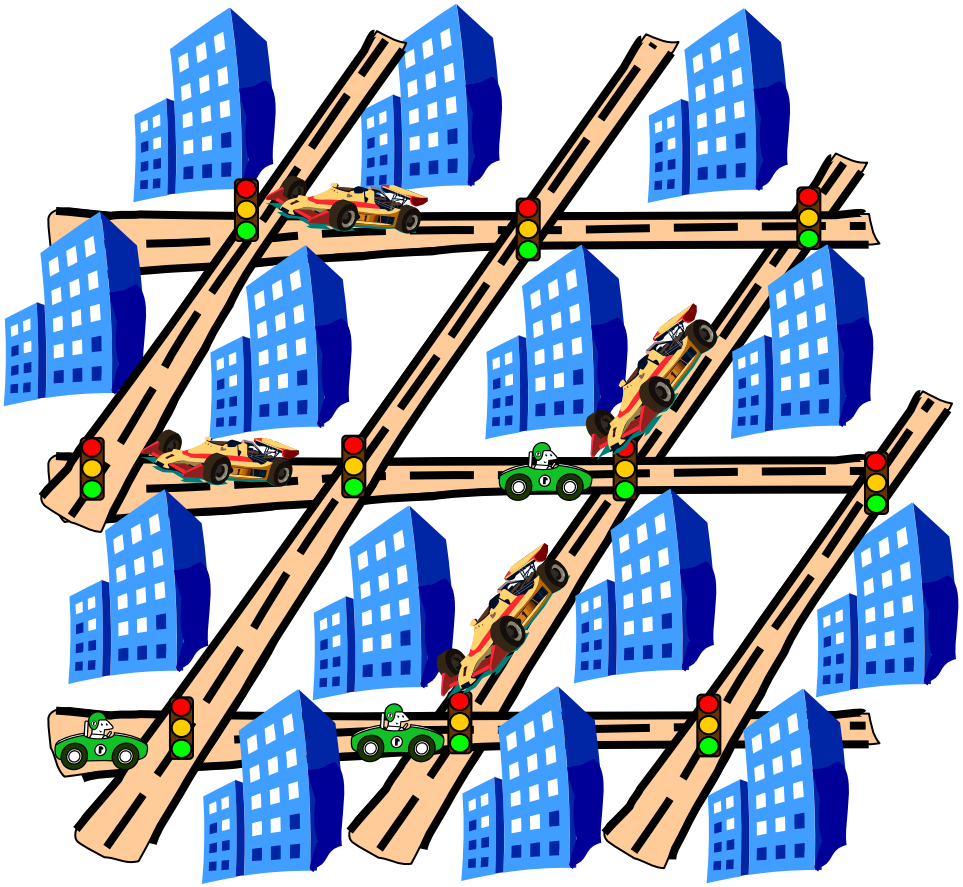


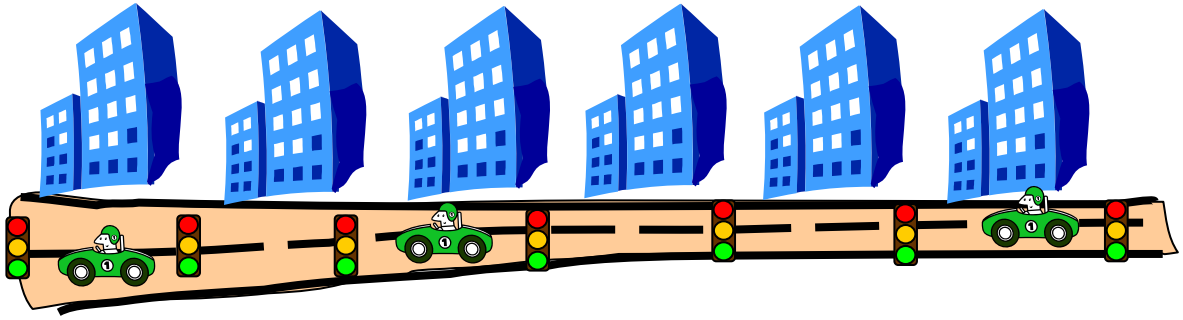
Mesh Multicore



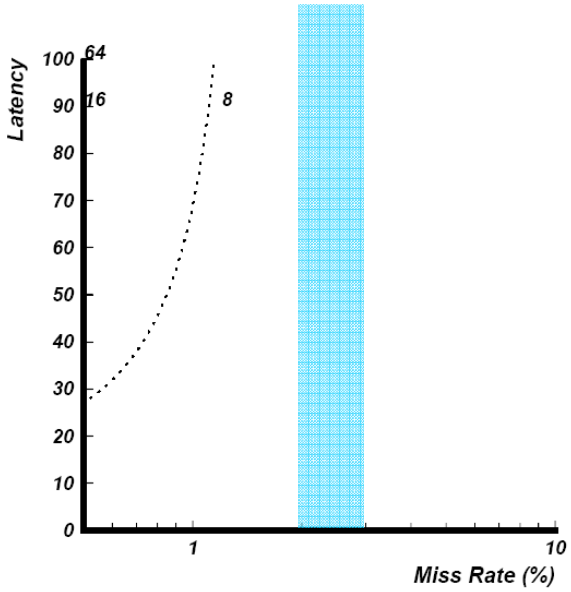
Imagine This City…

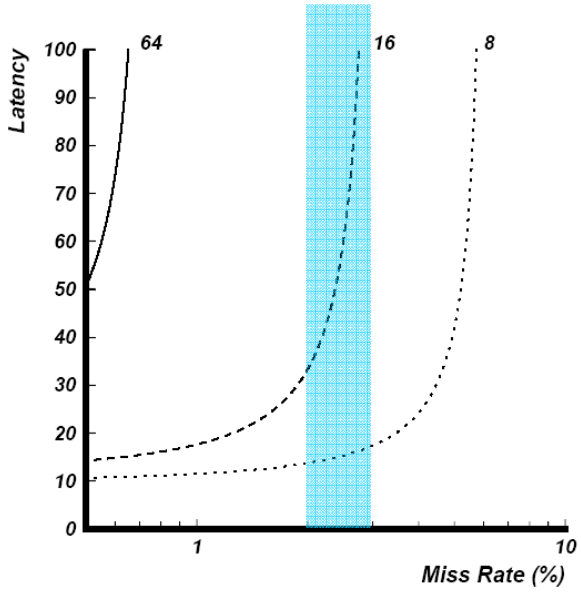


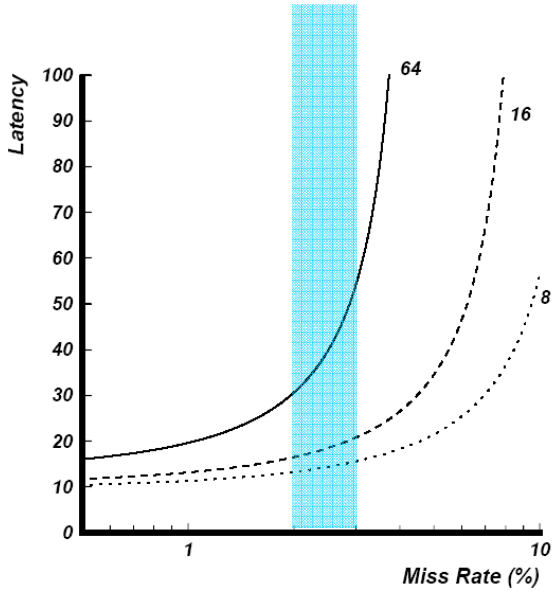




Interconnect Bandwidth



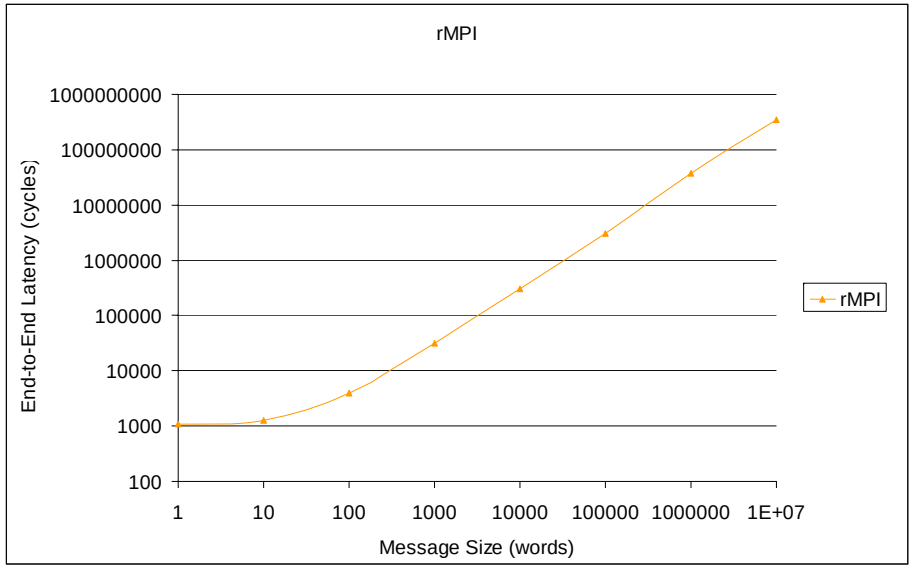




	Bus	Ring	Mesh
Cores:	2-4	4-8	> 8

Communication Latency

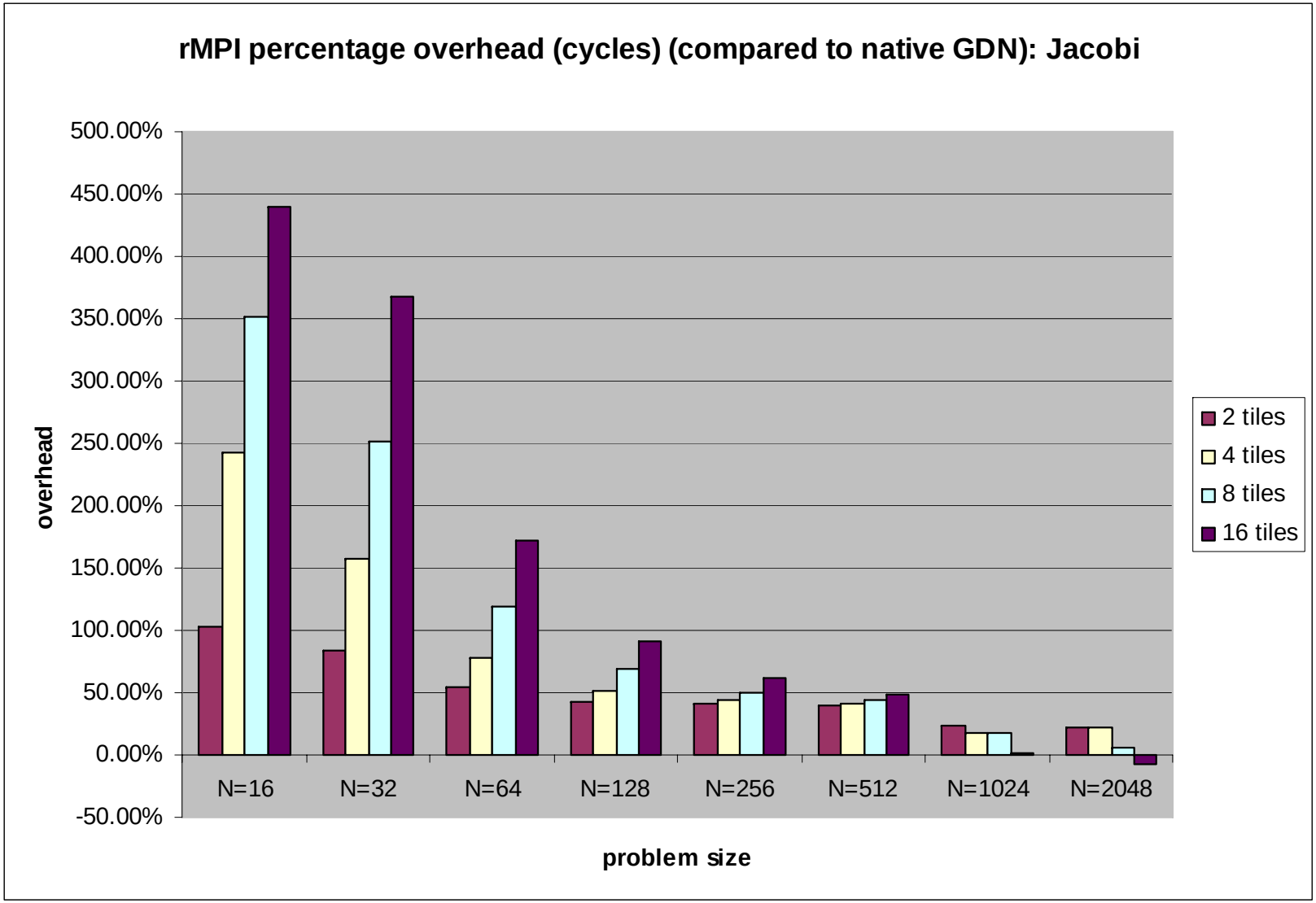
- Communication latency not interconnect problem! It is a "last mile" issue
- Latency comes from coherence protocols or software overhead



Highly optimized MPI implementation on the Raw Multicore processor

- Challenge: Reduce overhead to a few cycles
- Avoid memory accesses, provide direct access to interconnect, and eliminate protocols

rMPI vs Native Messages



Power Efficiency Challenge

- Existing CPUs at 100 watts
- 100 CPU cores at 10 KWatts!
- Need to rethink CPU architecture

The Potential Exists

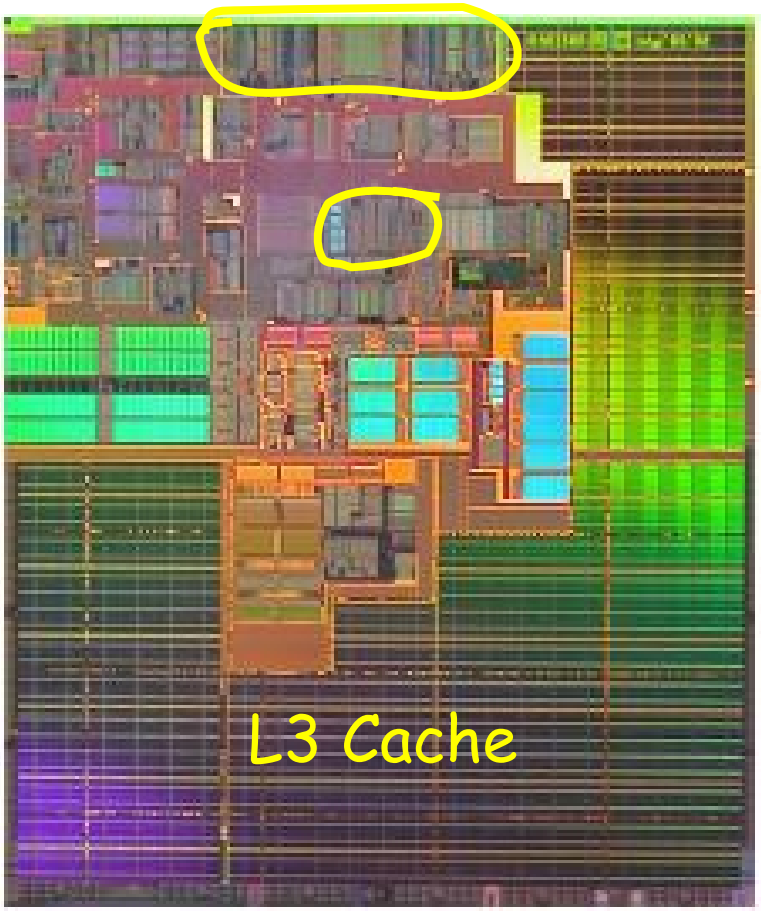
Processor	Power	Perf	Power Efficiency
Itanium 2	100W	1	1
RISC*	1/2W	1/8X**	25X

Assuming 130nm

* 90's RISC at 425MHz

** e.g., Timberwolf (SpecInt)

Area Equates to Power



Madison Itanium2
0.13μm

Less than 4% to
ALUs and FPUs

Photo courtesy Intel Corp.

Less is More

- Resource size must not be increased unless the resulting percentage increase in performance is at least the same as the percentage increase in the area (power)
- Remember power of n: n → 2n cores doubles performance
- 2n cores have 2X the area (power)
- e.g., increase a resource only if for every 1% increase in area there is at least a 1% increase in performance

"KILL Rule" for Multicore

Kill If Less than Linear

Communication Cheaper than Memory Access

Action	Energy
Network transfer (1mm)	3pJ
ALU add	2pJ
32KB cache read	50pJ
Off-chip memory read	500pJ

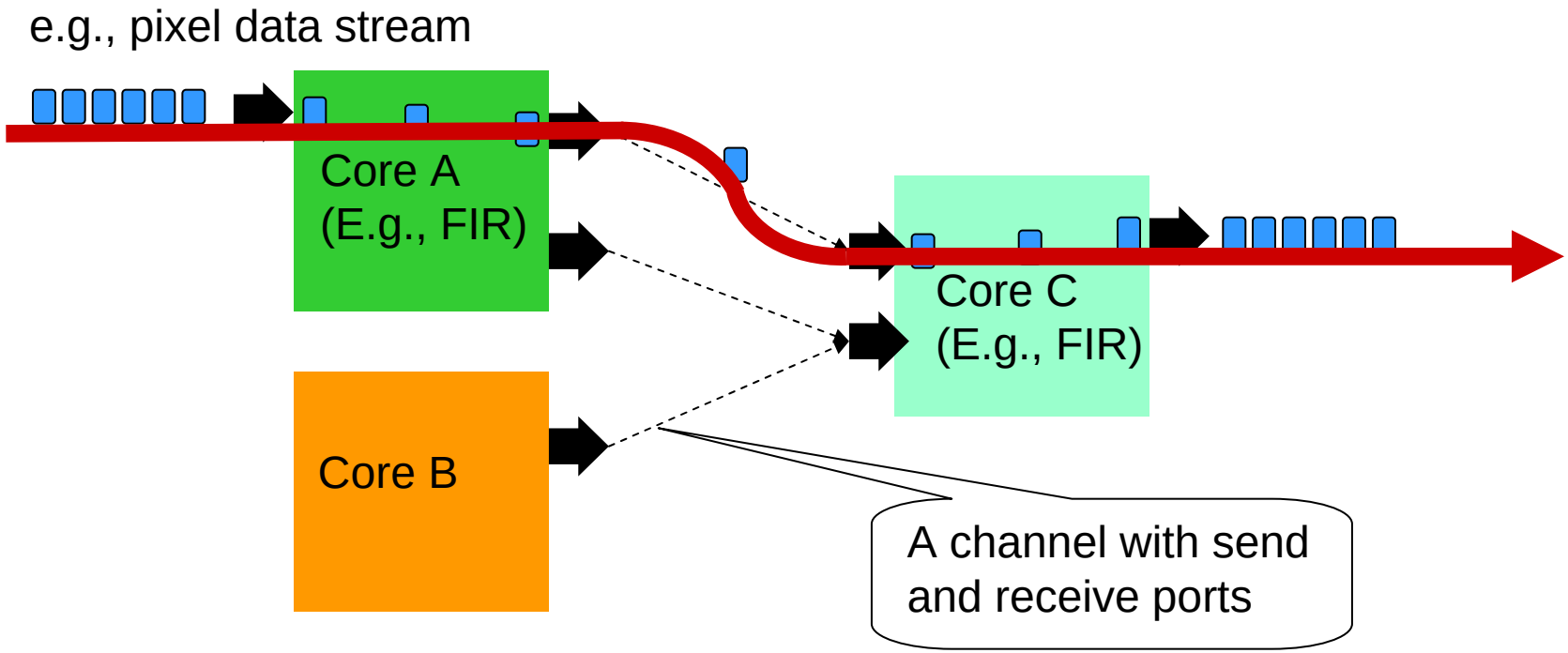
90nm, 32b

Migrate from memory oriented computation models to communication centric models

Multicore Programming Challenge

- Traditional cluster computing programming methods squander multicore opportunity
  - Message passing or shared memory, e.g., MPI, OpenMP
  - Both were designed assuming high-overhead communication
    - Need big chunks of work to minimize comms, and huge caches
  - Multicore is different – Low-overhead communication that is cheaper than memory access
    - Results in smaller per-core memories
- Must allow specifying parallelism at any granularity, and favor communication over memory

Stream Programming Approach



- ASIC-like concept
- Read value from network, compute, send out value
- Avoids memory access instructions, synchronization and address arithmetic

e.g., Streamit, StreamC

Conclusion

- Multicore can close the "Moore's Gap"
- Four biggest myths of multicore
  - Existing CPUs make good cores
  - Bigger caches are better
  - Interconnect latency comes from wire delay
  - Cluster computing programming models are just fine
- For multicore to succeed we need new research
  - Create new architectural approaches; e.g., "Kill Rule" for cores
  - Replace memory access with communication
  - Create new interconnects
  - Develop innovative programming APIs and standards

Vision for the Future

- The 'core' is the LUT (lookup table) of the 21st century

- If we solve the 3 challenges, multicore could replace all hardware in the future