New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies

Fred Pollack

Intel Fellow

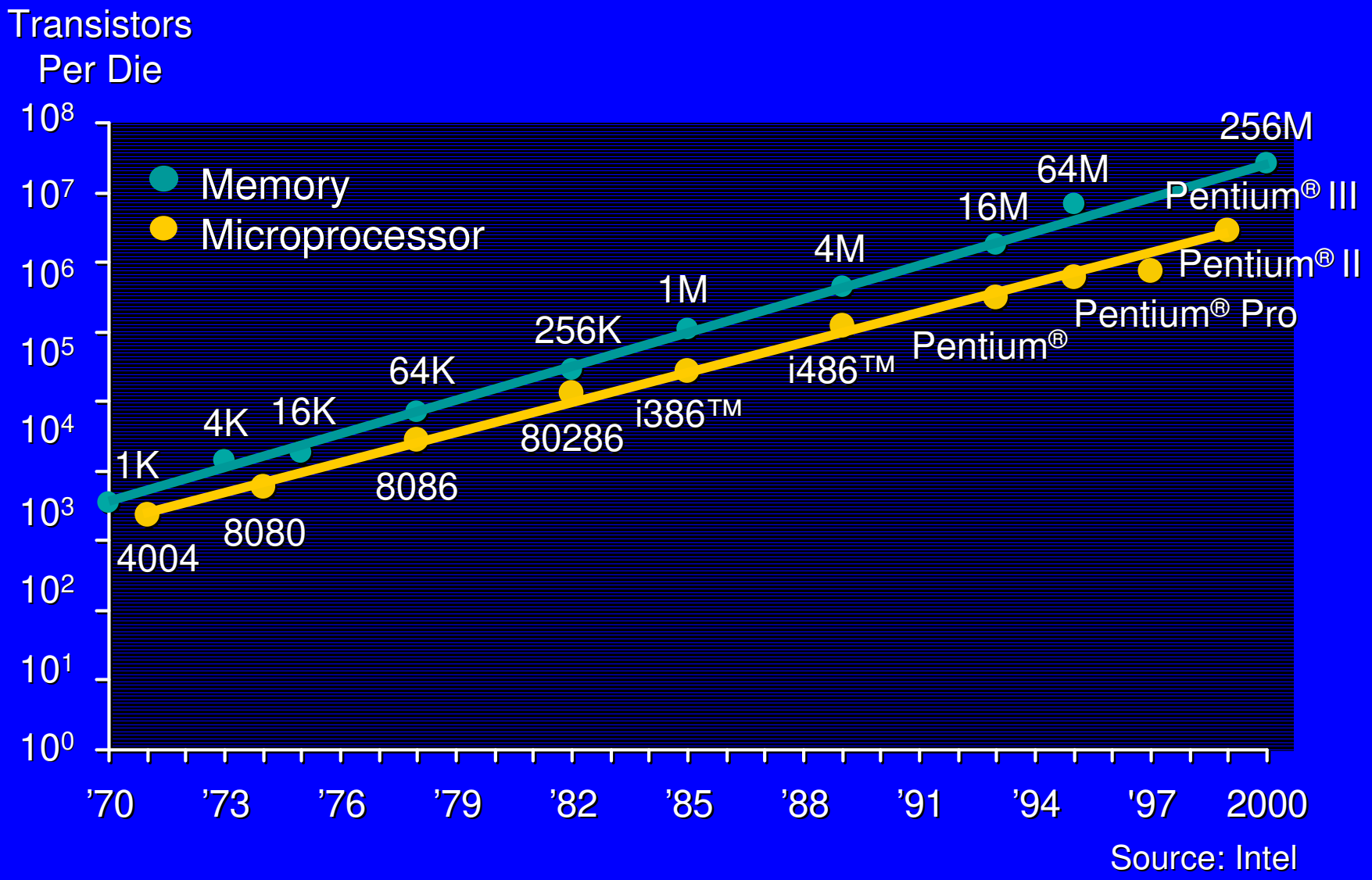
Director of Microprocessor Research Labs

fred.pollack@intel.com

Contributors: Shekhar Borkar, Ronny Ronen

intel®

Moore's Law



Source: Intel

intel®

In the Last 25 Years Life was Easy

Doubling of transistor density every 30 months

Increasing die sizes, allowed by

- Increasing Wafer Size
- Process technology moving from “black art” to “manufacturing science”

⇒ Doubling of transistors every 18 months

And, only constrained by cost & mfg limits

But how efficiently did we use the transistors?

Performance Efficiency of µarchitectures

Tech	Old µArch	mm (linear)	New µArch	mm (linear)	Area
1.0µ	i386C	6.5	i486	11.5	3.1
0.7µ	i486C	9.5	Pentium® proc	17	3.2
0.5µ	Pentium® proc	12.2	Pentium Pro®	17.3	2.1
0.18µ	Pentium III® proc	10.3	Next Gen	?	2--3

Implications: (in the same technology)

1. New µArch ~ 2-3X die area of the last µArch
2. Provides 1.5-1.7X integer performance of the last µArch

We are on the Wrong Side of a Square Law

intel®

Power Efficiency

Power is proportional to Die-area * Frequency

~2X frequency with each process generation

- Normally expect 1.5X from process technology
- Less gates per pipeline stage, e.g. due to deeper pipelines
- Pushing process technology

Examples

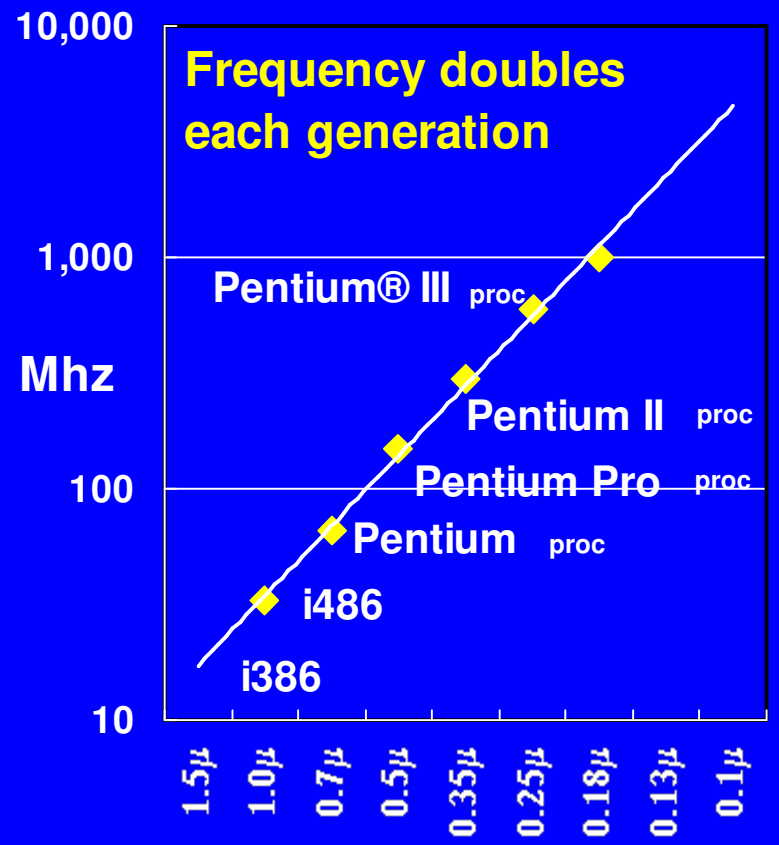
- On 0.35µ Pentium® processor at 200MHz vs Pentium II processor at 300Mhz. Difference due to pipeline depth.
- Pentium II processor at 300MHz on .35u vs. Pentium III processor at 600Mhz on 0.25u
  - » Same core µarchitecture
  - » ~50Mhz in speed-path work. The rest was pushing the process technology

Other Factors

- ⇩ Decreasing voltage and capacitance with each new process technology
- ⇩ Increasing use of circuit & µarch techniques for lower power
- ⇧ Increasing transistor sizing to push frequency

intel®

Trends and expectations

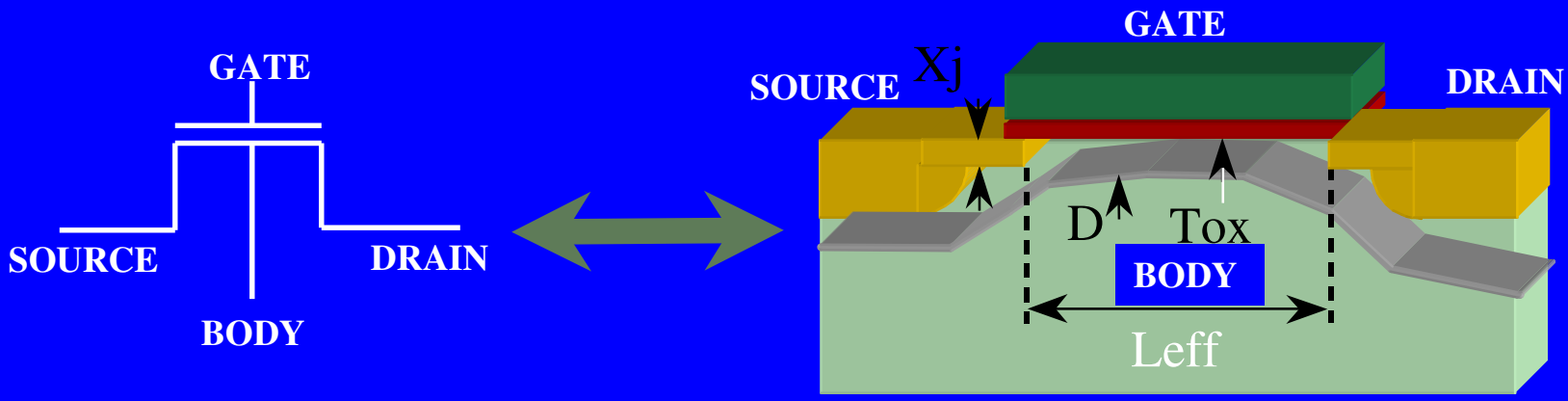


With Each Process Generation:

- Frequency increased by ~2X (not 1.5X)
- Vcc will scale by only ~0.8 (not 0.7)
- Active power will scale by ~0.9 (not 0.5)
- Active power density will increase by ~30-80% (not stay constant)
- Leakage power will make it even worse, and

intel®

As the technology scales...



$Width = W = 0.7,\ Length = L = 0.7,\ t_{ox} = 0.7$

1. Dimensions reduce 30%, this is good

$$Area\ \ Cap\ = C_a = \frac{0.7 \times 0.7}{0.7} = 0.7,$$

$$Fringing\ \ \ Cap\ = C_f = 0.7,$$

$$Total\ \ Cap\ \Rightarrow C = 0.7$$

2. Capacitance on a node reduces by 30%, this is good

$$Die\ Area = X \times Y = 0.7 \times 0.7 = 0.7^2$$

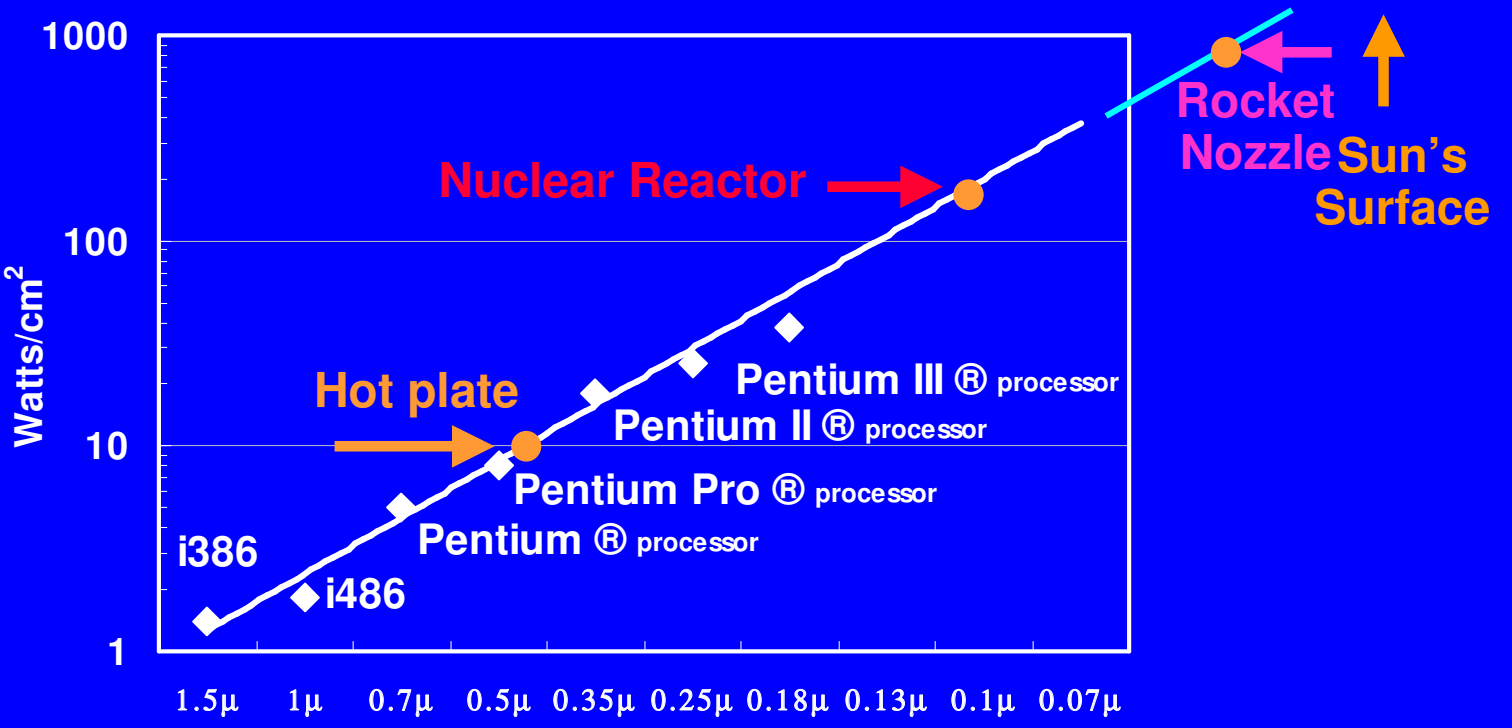
3. Transistor density (integration) doubles, this is good

$$\frac{Cap}{Area} = \frac{0.7}{0.7 \times 0.7} = \frac{1}{0.7}$$

4. Capacitance per unit area increases 43%, this is not good

intel®

Power density continues to get worse



Surpassed hot-plate power density in 0.5µ

Not too long to reach nuclear reactor

intel®

Some implications

We can't build microprocessors with ever increasing die sizes

The constraint is power – not manufacturability

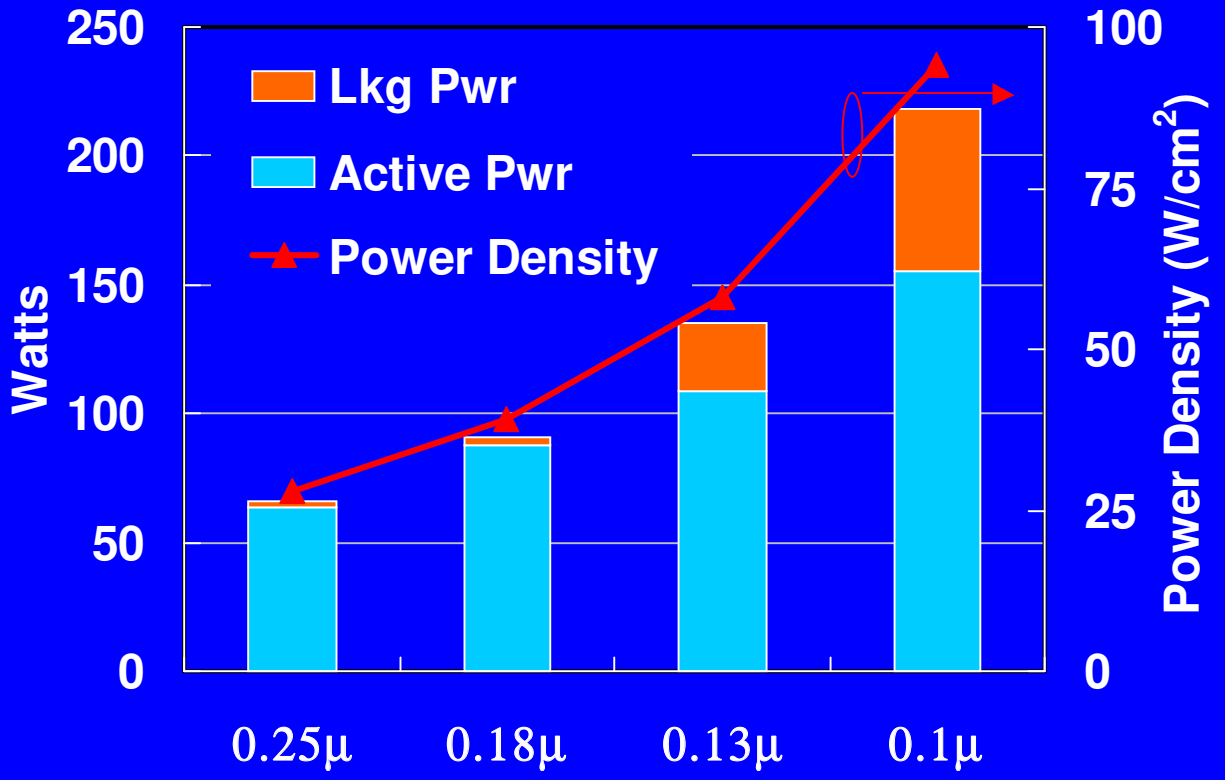
Given the trends:

- What happens to power if we hold die size constant at each generation
- What happens to die size, if we hold power constant at each generation

intel®

Constant die size

(Allows ~100% growth in transistors each generation)

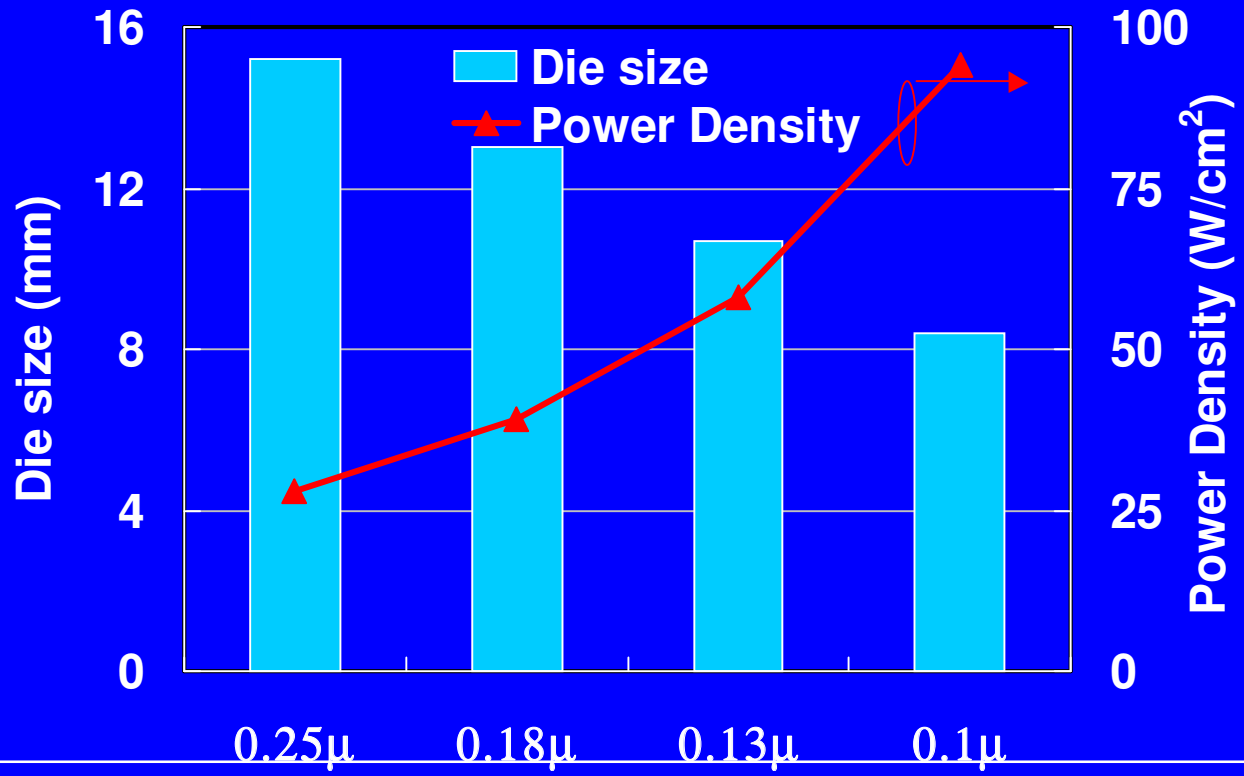


~15mm die
1.5X freq increase
each generation

Limiters:
1. Power dissipation,
2. Power delivery, and
3. Power density

intel®

If you limit die size due to power...



~66 Watts total, 1.5X freq increase each generation

Die size has to reduce ~25% in area each generation

– Implies ~50% vs. the 200+% historical growth in transistors

Limits performance

Power density does not improve

intel®

Therefore

Business-as-usual won't work

We need to look at alternatives – and we all are

intel®

Current Directions

*Low-power circuit and µarch techniques**

*SIMD ISA extensions**

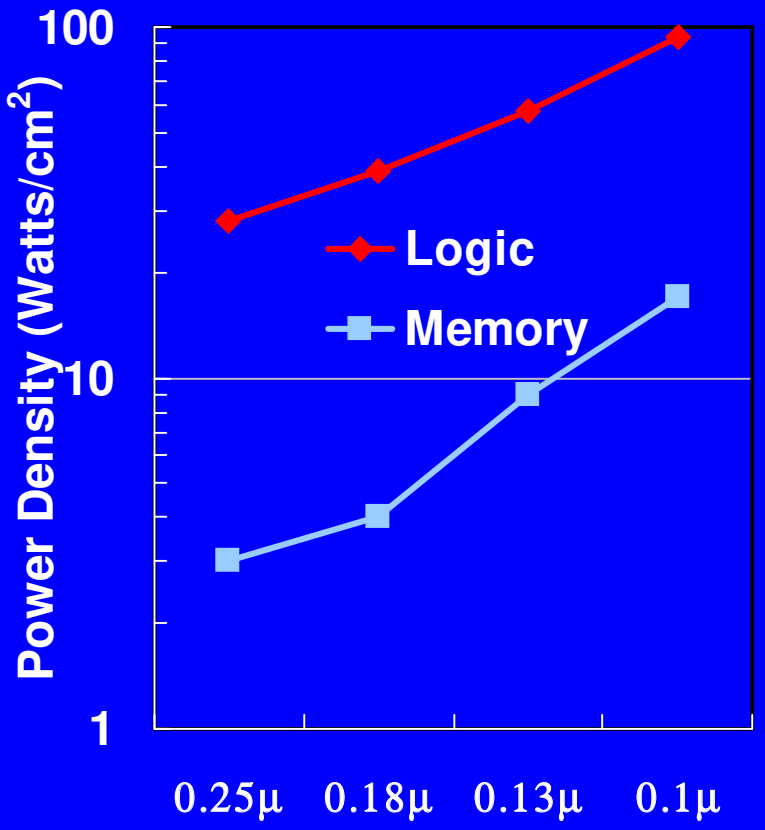
On-die L2 caches

Multiple CPU cores on die

Multithreaded CPU On-Die L2 Caches

* Not discussed

Memory is more power efficient



Static memory has 10X lower active power density

Lower leakage than logic

Leakage control is also easier to implement than logic

Integrated L2 provides:

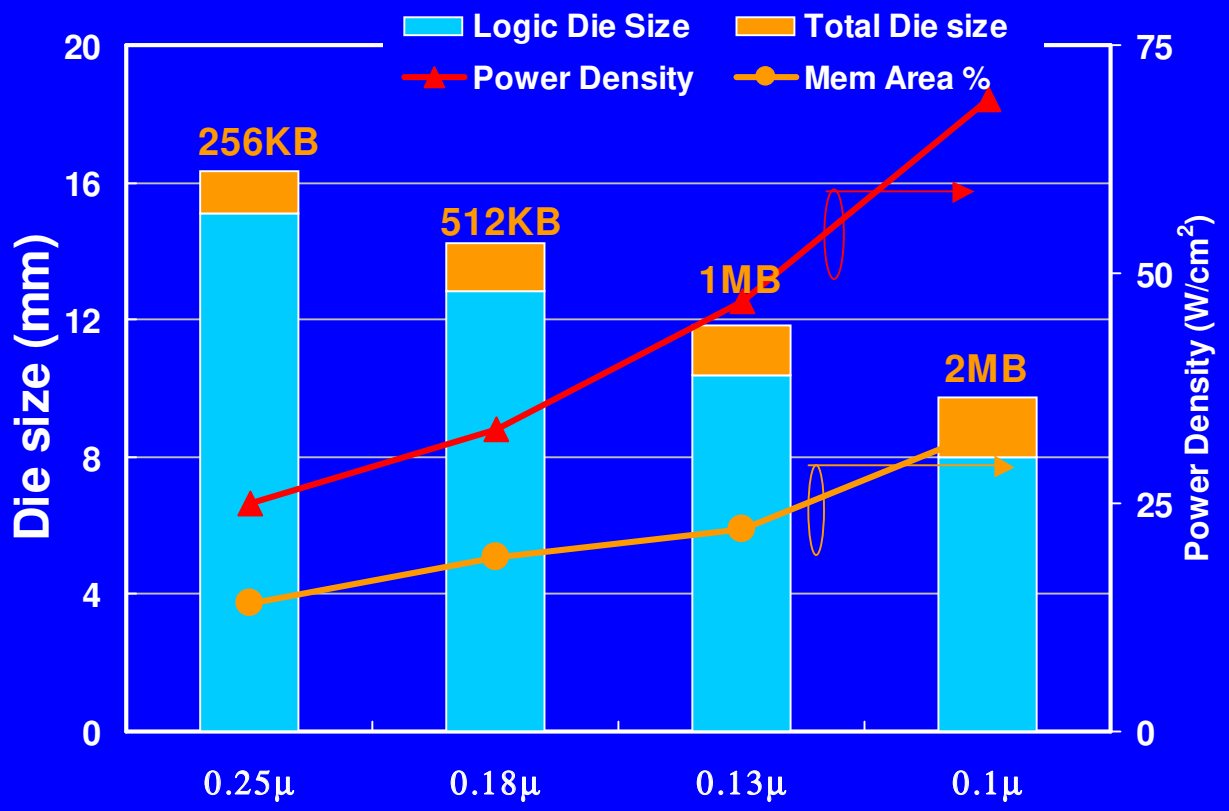
1. Higher bandwidth

2. Lower latency

So on-die L2 caches make sense

intel®

Can easily double the on-die L2 ...



66 Watts constant, 1.5X freq increase each generation

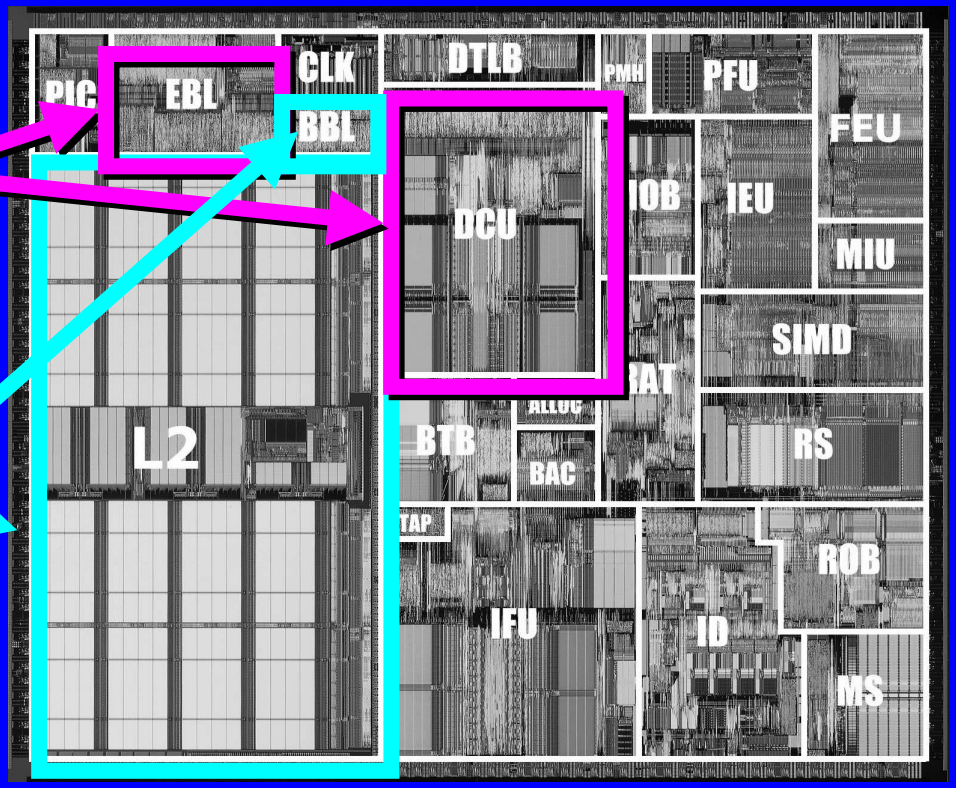
- Die space not used for logic can be used for on-die L2 cache
  - To improve performance with <10% increase in max power
- >512KB L2 good for server performance, but small impact on PC desktop performance
- And little help in "real" power density

intel®

Example: Pentium® III processor on .18µ process technology

Advanced System Buffering

Advanced Transfer Cache



- 256KB L2
- 28 million transistors
- 106 mm² die size
- Multi-voltage capability: 1.1V-1.7V
- On-die GTL+ termination

intel®

Advanced System Buffering

Balanced increase in buffers to minimize bottlenecks

- Buffer sizes maximize utilization of the 133MHz system bus bandwidth

6 Fill buffers (vs 4)

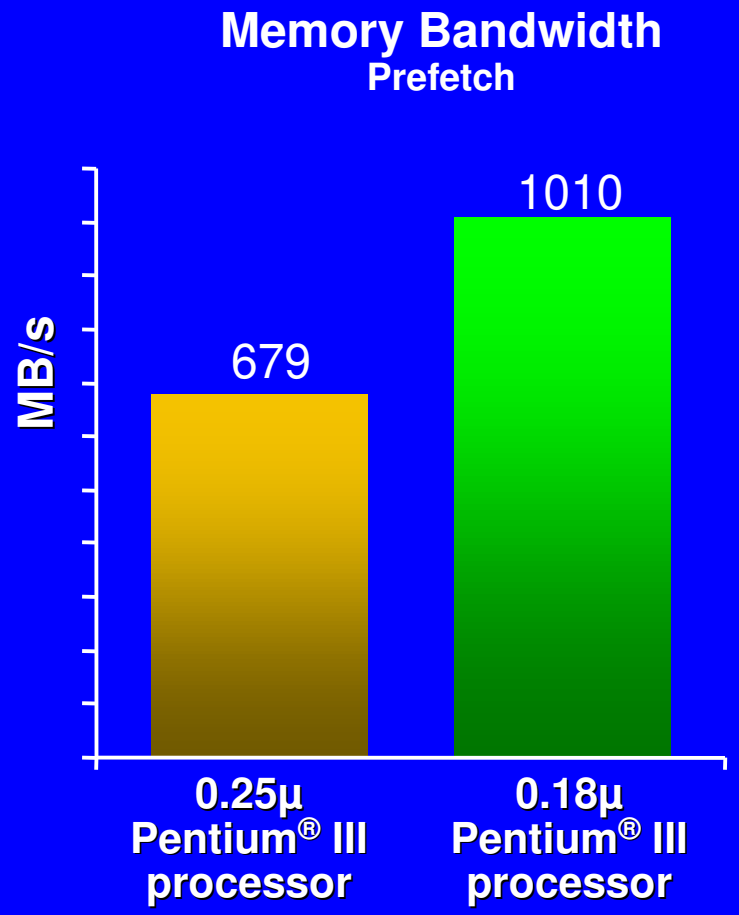
- 50% increase in concurrent non-blocking data cache operations

8 Bus queue entries (vs 4)

- Allows more outstanding memory/bus operations

4 Writeback buffers (vs 1)

- Reduced blocking during cache replacement operations
- Faster deallocation time for fill buffers



0.18µ Pentium® III processor 600 MHz (with ATC, ASB) vs. 0.25µ Pentium III processor 600 MHz System configuration: Pre-production Intel VC820 board with 133 MHz system bus, 128MB RDRAM, Seagate Barracuda SCSI, STB4400 Velocity AGP 2X. Intel internal design analysis tools used to obtain measurement data.

intel®

Advanced Transfer Cache

Size

- 256KB on-die level 2 cache

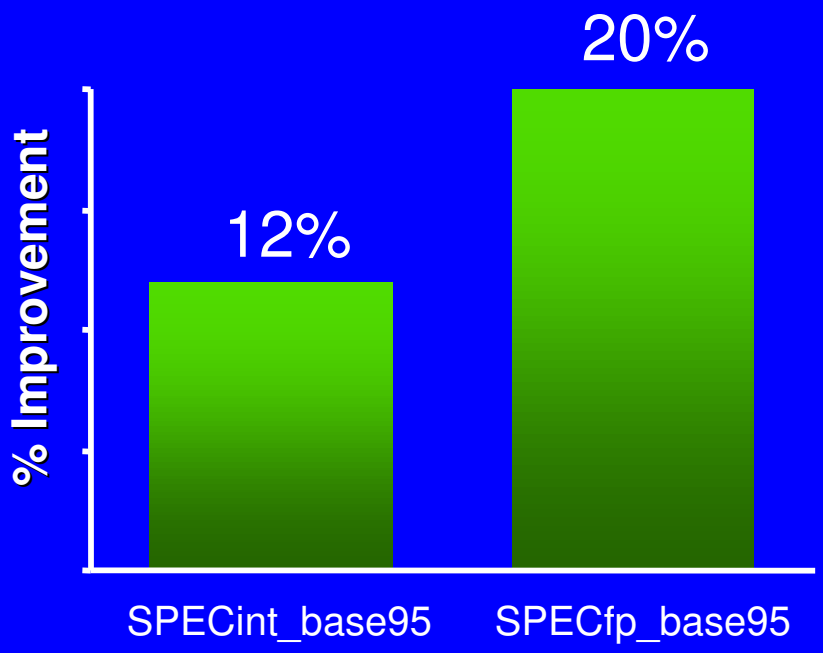
Organization

- 8-way set associative, 1024 sets
- 32 byte line (32 bytes data, 4 bytes ECC)
- 36-bit physical address space

Cache Bus

- Full speed, scaleable with core frequency
- 288-bit transfer width (256 data, 32 ECC)
- 2 cycle back-to-back throughput
- >4x reduction in latency (as compared to 0.25µ Pentium® III processor)

Performance Gain at Equal MHz (600MHz)

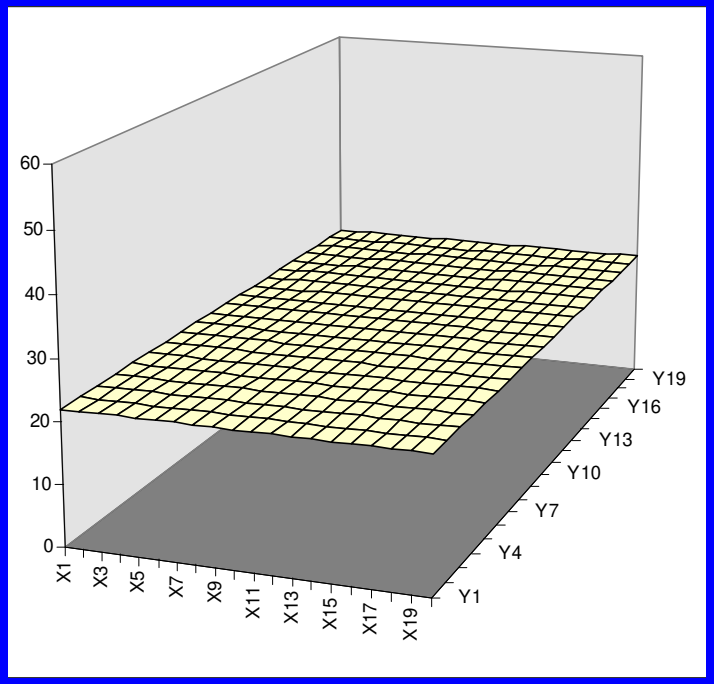


0.18µ Pentium III processor 600 MHz (with ATC, ASB) vs. 0.25µ Pentium III processor 600B MHz
Source: Intel MAP; Results estimated using Intel C/C++ Compiler 4.5 and Intel Fortran Compiler 4.5
System configuration: Pre-production Intel VC820 board with 133 MHz system bus, 256MB RDRAM, IBM371800 ATA-66, Diamond Viper 770 Ultra TNT2 AGP4X

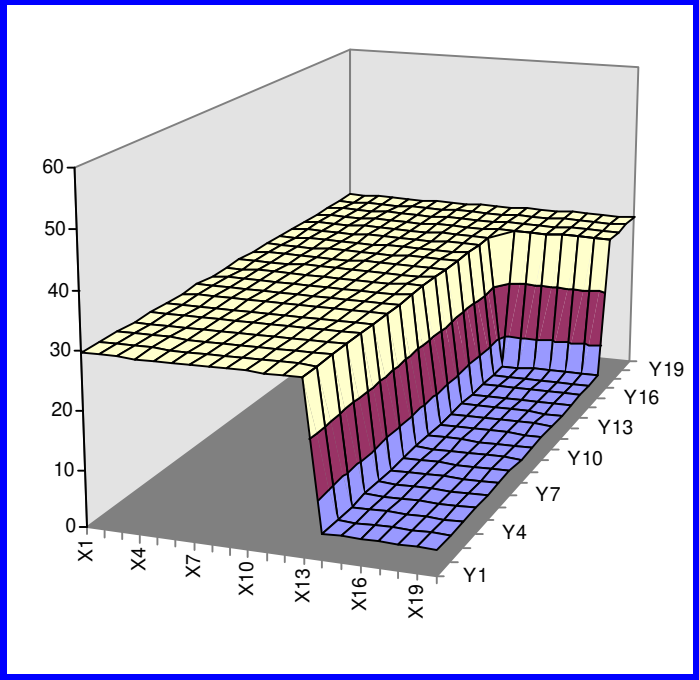
intel®

Power Density: Cache vs. Logic

Past: Thermal Uniformity



Present: Logic vs. Cache



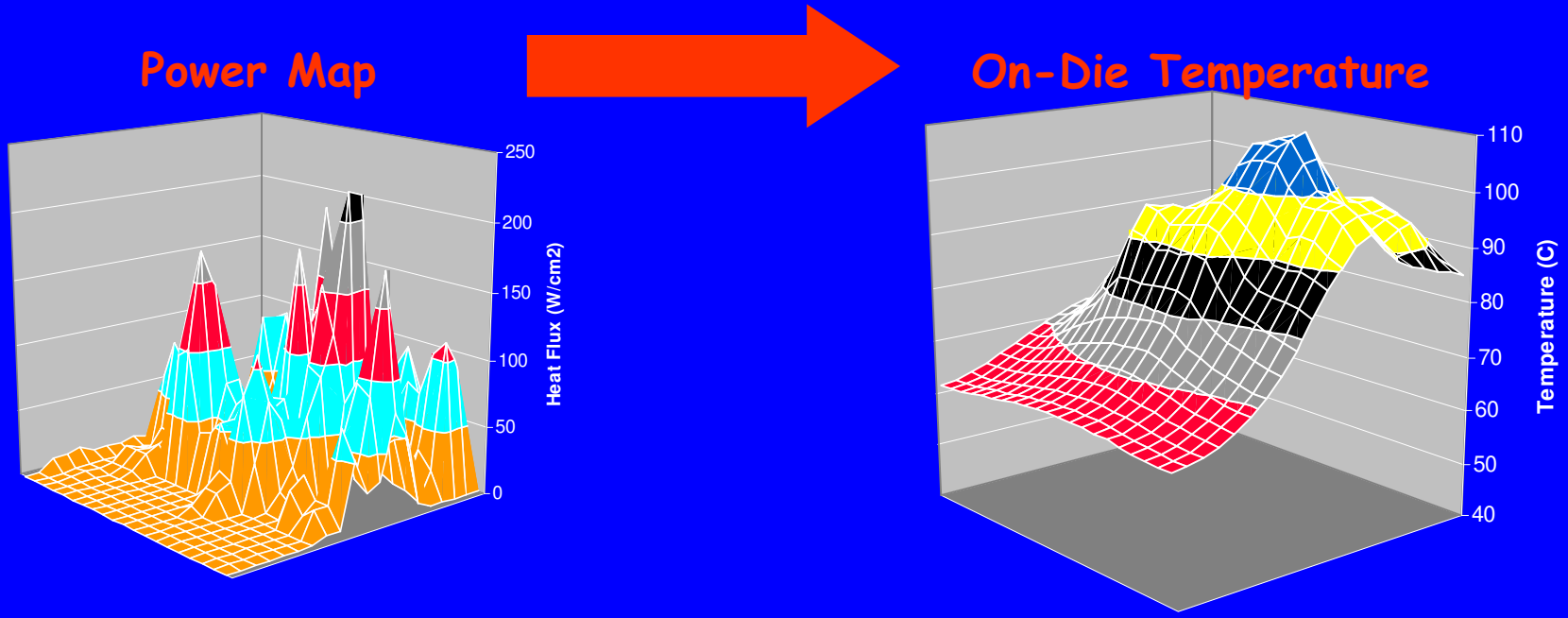
As die temperature increases, CMOS logic slows down

With low power density (past), can assume uniformity

With increasing power density and on-die L2 cache, need to consider simplistic non-uniformity

intel®

Power Density: The Future



With high power density, cannot assume uniformity

- As die temperature increases, CMOS logic slows down
- At high die temperatures, long-term reliability can be compromised

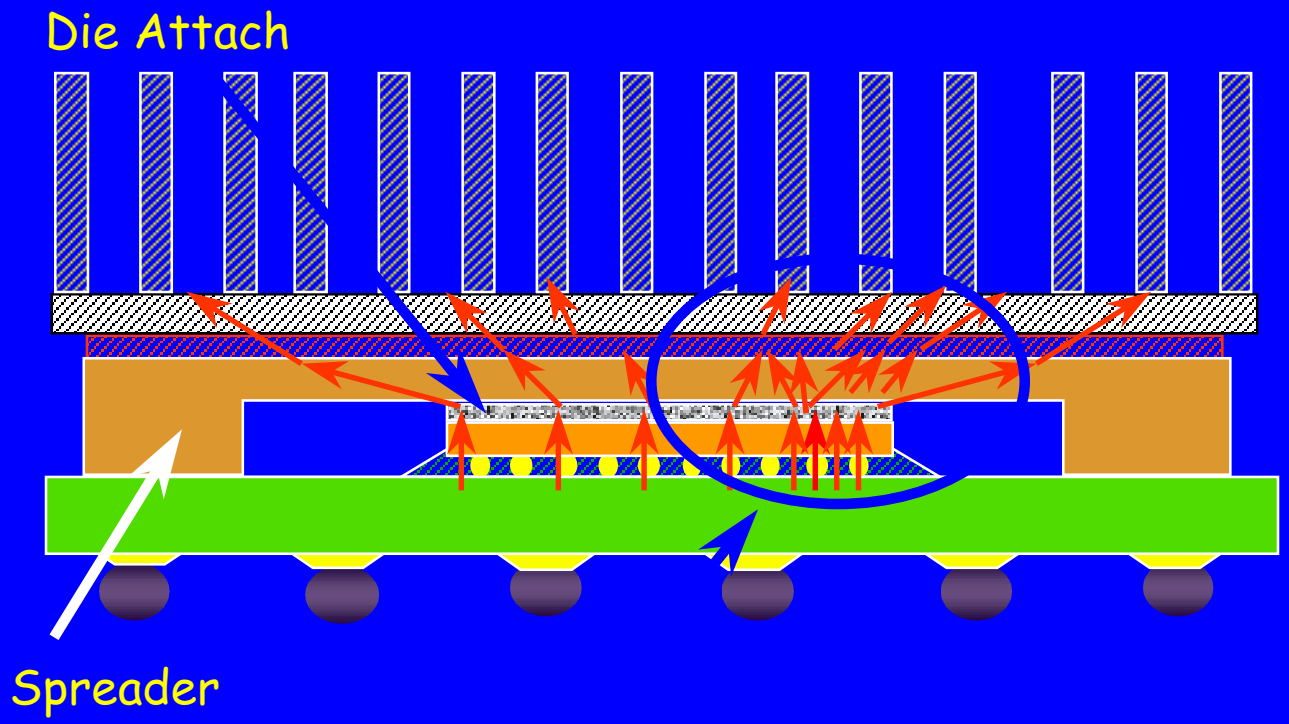
Silicon is not a good heat conductor

- Impact on packaging, w.r.t. cooling

intel®

Packaging Implications

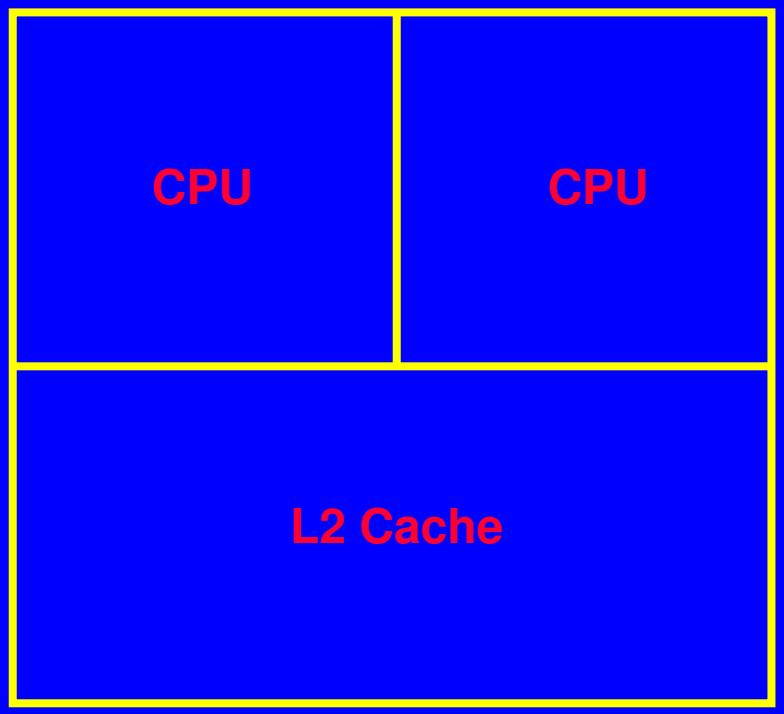
- Power removal for non-uniform heating is a big challenge since we need to
  - spread the heat (smooth local concentrations) &
  - then dissipate it in the ambient
  - Hot spots created on die since we cannot completely smooth them away



Need to spread out local concentrations

intel®

Multiple CPU on Die



Shared L2 more efficient than separate L2's of ½ size

About linear performance with die size vs. historical square law

Unlikely that both CPUs are at Max Power at same time

- Typical application power << max power on each CPU
- Can throttle performance if both CPUs approach max power at same time.

Can simplify interconnect in SMP system

Also, can be used to build highly reliable system via FRC

intel®

Multithreading

Single CPU µArch augmented to look as 2 or more CPUs to software

Adds ~10% logic to CPU

Max Power increases <10%

Can increase throughput by 30+%

Helps to address increasing overhead of cache misses

intel®

Key Challenges for future Microarchitectures

Special Purpose Performance

Increased Execution Efficiency

Breaking the Dataflow Barrier

- With efficiency

intel®

Special vs. General Purpose Performance

Special purpose performance can deliver more MIPS/mm².

To Date: SIMD integer and floating-point instructions added to several ISAs

- <10% in die area and power increase, and 1.5-4x increase in multimedia/3D kernels

With future silicon budgets approaching 100M transistors, we need to consider:

- Integration of other platform components (e.g. Memory controller, graphics)
- Special purpose logic, programmable logic, & separately programmable engines
- But all have very complex/costly software issues

Challenge: Design for "Valued Performance"

intel®

Increased Execution Efficiency

Max Power Occurs on Code that Keeps Pipeline full and all superscalar units busy

- Thermal solution designed for Max Power
- Power delivery designed for Max Power

But few apps spend any significant time operating at Max Power

And the wider and deeper the execution core, the greater the inefficiency

Thus, with power constraints, need to focus on techniques that increase execution core efficiency and only add modest additional logic and power

Challenges to Increased Efficiency

Improved prediction for less unused speculation

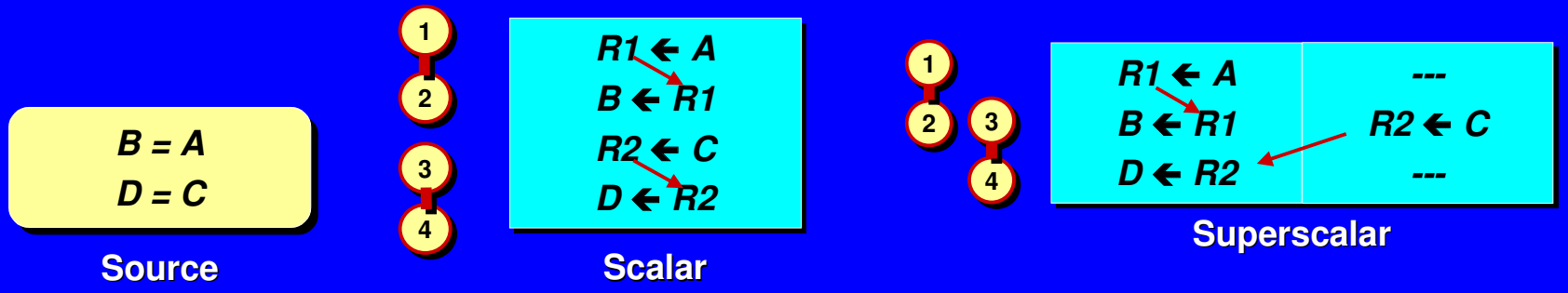
Establish & use *Confidence* measures

– For example, don't speculate on a flaky branch if another thread can better use the execution resources

intel®

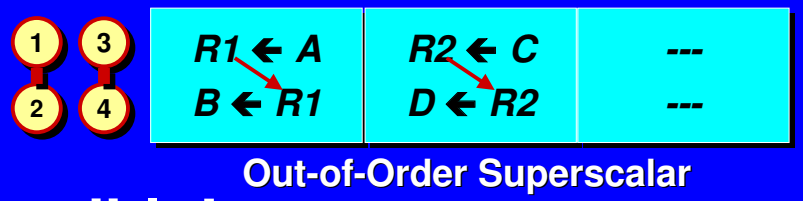
Data Flow Execution

In Order Processors



Out-of-Order Processors

– Order is not important, **data flow** (dependencies) matters



The Goal: Shortest length as possible!

… But still limited by instruction dependencies

1. How much limited?

2. Can we break the Data Flow Barrier?

intel®

Breaking the Barrier: Beyond Data-Flow Execution

Driving Idea:

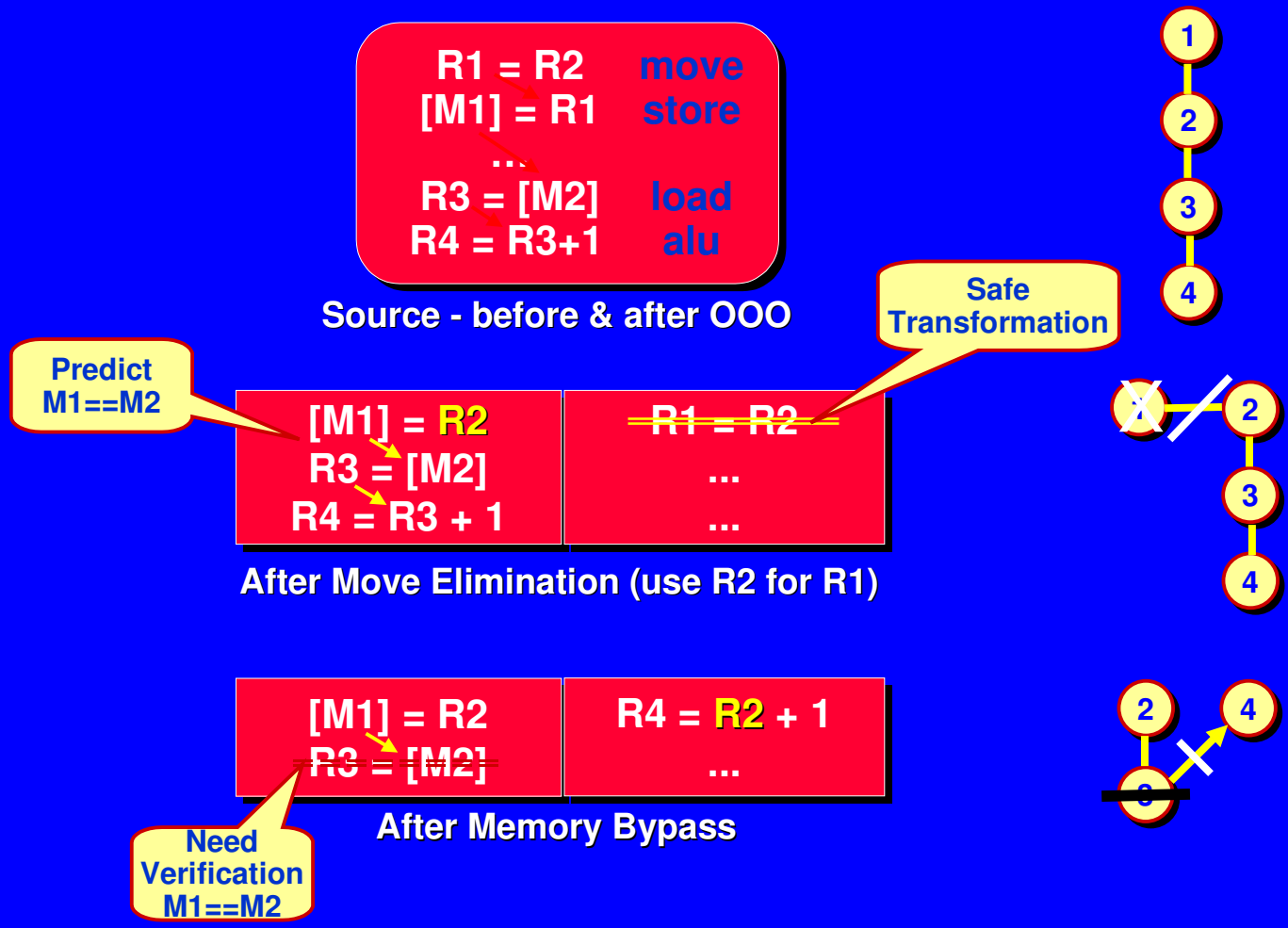
- Transform the DFG into an improved DFG which:
  - » Has a shorter critical path (higher parallelism)
  - » Has less instructions

Families of Transformations

- Safe transformations
  - » **Like compilers do, but using dynamic information**
- Speculative transformations
  - » **Guess intermediate values (results, addresses, flags,…)**
  - » **Ignore dependencies**
  - » **Verify and redo if wrong**
  - » **Do it smart to reduce unused speculation (use confidence factor)**

Simple Transformation Examples

Move Elimination & Memory Bypass



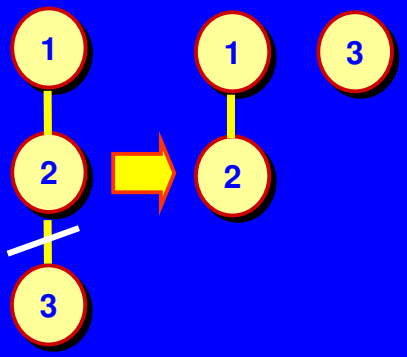
intel®

Value Prediction

Predict the outcome value of an instruction instead of waiting for it to be ready/produced

- **But confidence factor is key to avoiding unused speculation**

Enables ***Dependency Elimination*** thus collapsing the DFG



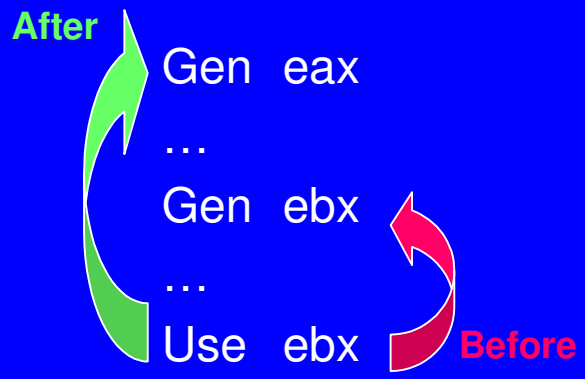
Don't wait for the outcome of "2" in order to execute "3" but predict it's value

intel®

Value Identity Predictor

If you can't predict the actual value, try to predict if it is identical to a value produced by a prior instruction

Enables ***Dependency Redirection*** thus collapsing the DFG



But with confidence

intel®

Summary

Logic Transistor growth constrained by power – not mfg

– At constant power, 50% per process generation vs. over 200% in past

Current Directions in microarchitecture that help

– SIMD ISA extensions
– On-die L2 caches
– Multiple CPU cores on die
– Multithreaded CPU

Key Challenges for future Microarchitectures

– Special purpose performance
– Increased execution efficiency: improved prediction and *confidence*
– Break the data-flow barrier, but in a power efficient manner

CMOS Challenges beyond thermal power

– Increasing power density
– Leakage Power becoming a significant factor
– Increasing and quickly-changing current with lower voltage (di/dt)
– SER (soft error rate) – not just a memory problem

Conclusion

Power is the key challenge

- Need to address at all levels (process, circuits, architecture, compiler)
- And, use multi-disciplinary approach

New Goal:

Double *Valued Performance* every 18 months, at the same power level

intel®