ECE 4514
Digital Design II
Spring 2008

Lecture 22:
Design Economics:
FPGAs, ASICs, Full Custom

A *Tools/Methods* Lecture

Patrick Schaumont

Overview

- Wows and Woes of scaling
  - The case of the Microprocessor
  - How efficiently does a microprocessor use transistors ?
- Alternative Technologies: FPGA, ASIC, Full Custom
  - Energy Efficiency and Design Cost
- ASIC Standard Cell Design
- Full Custom Design
- Future Challenges in Digital Design

The Wows of scaling

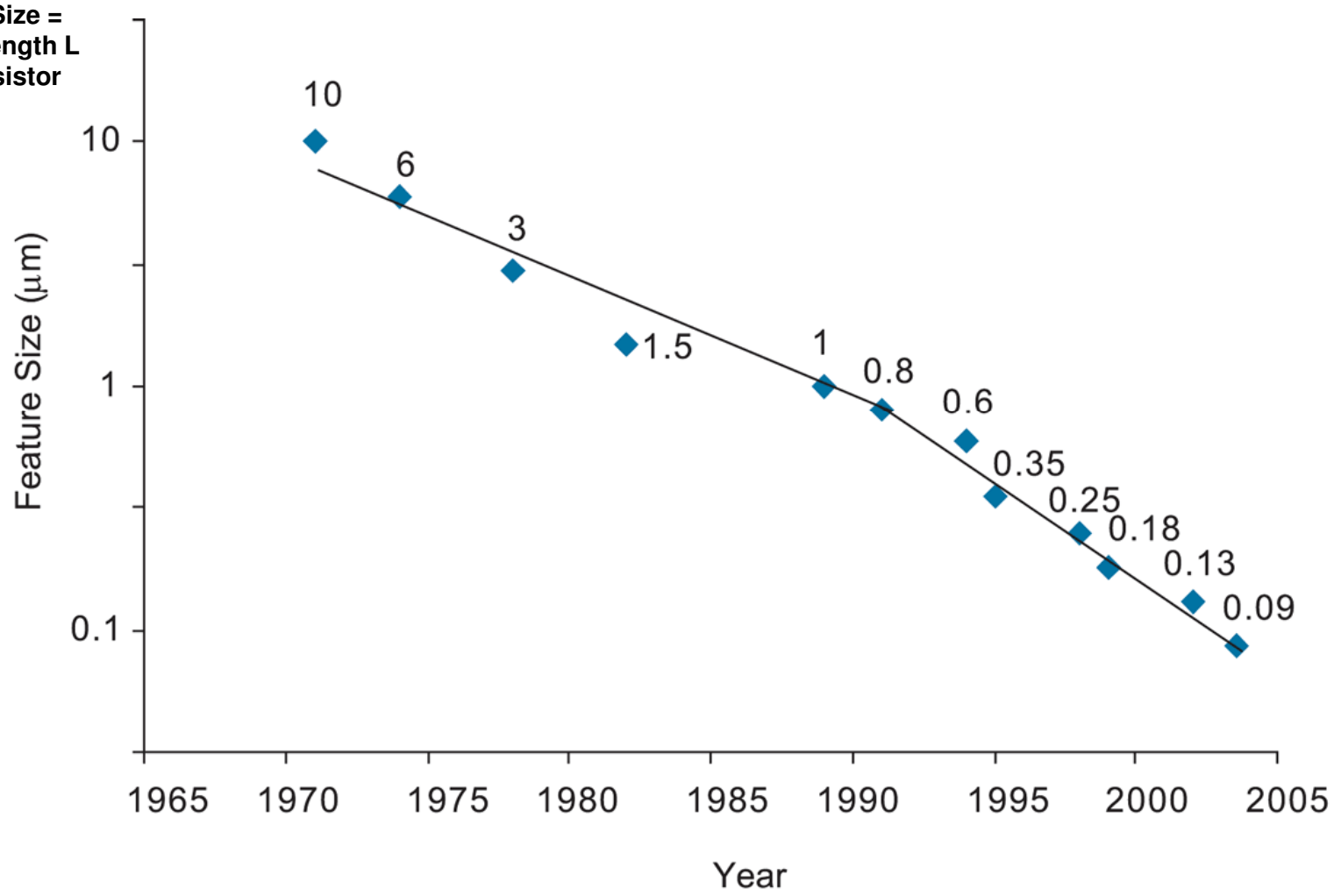
- Everybody is using Microprocessors because
  - PC's are everywhere
  - C compilers are everywhere
  - They get faster all the time
  - They get cheaper all the time

- All this is thanks to technology scaling

Scaling enabled performance improvements

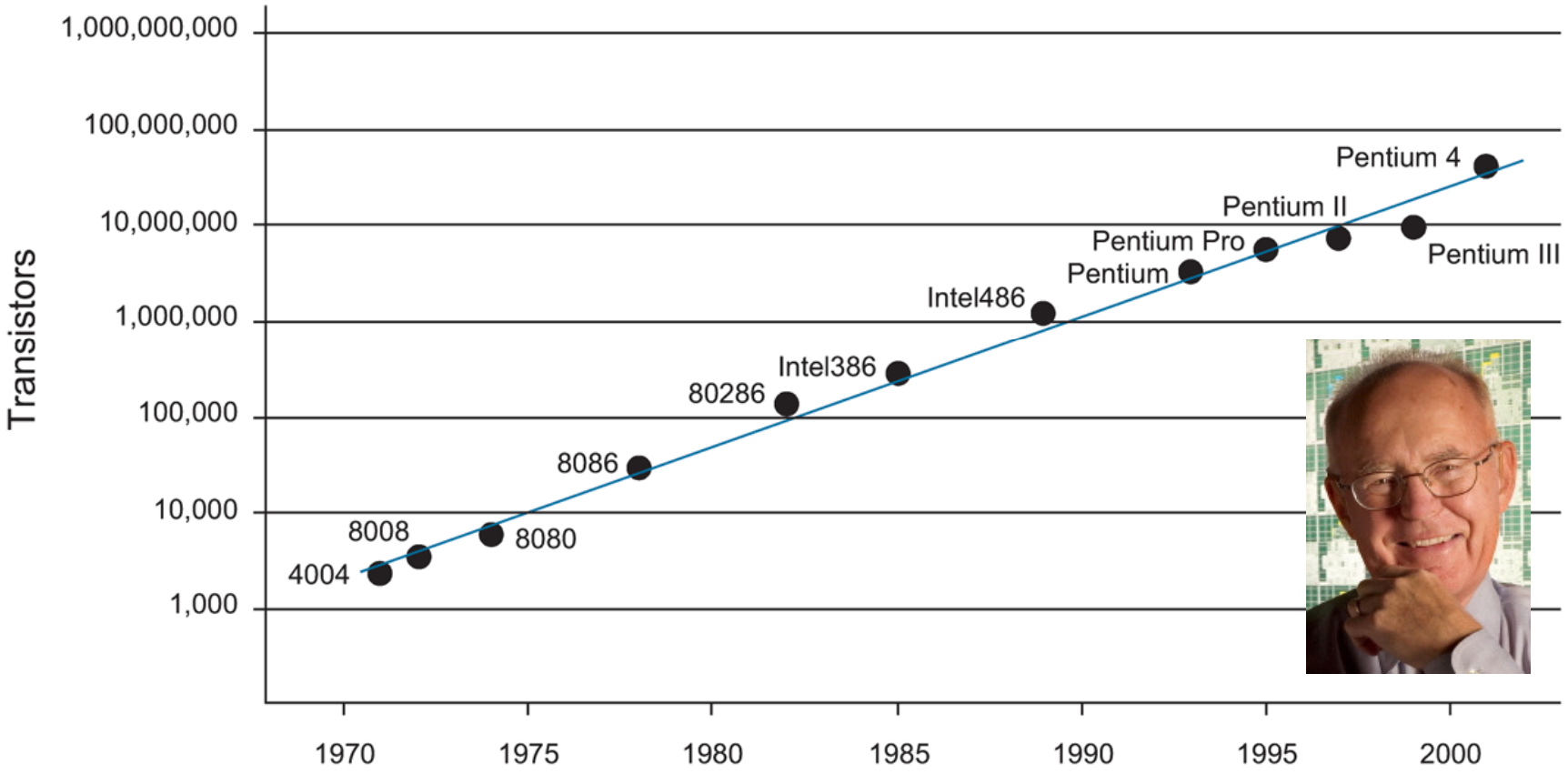
[Weste & Harris]

Feature Size = Channel length L of a transistor



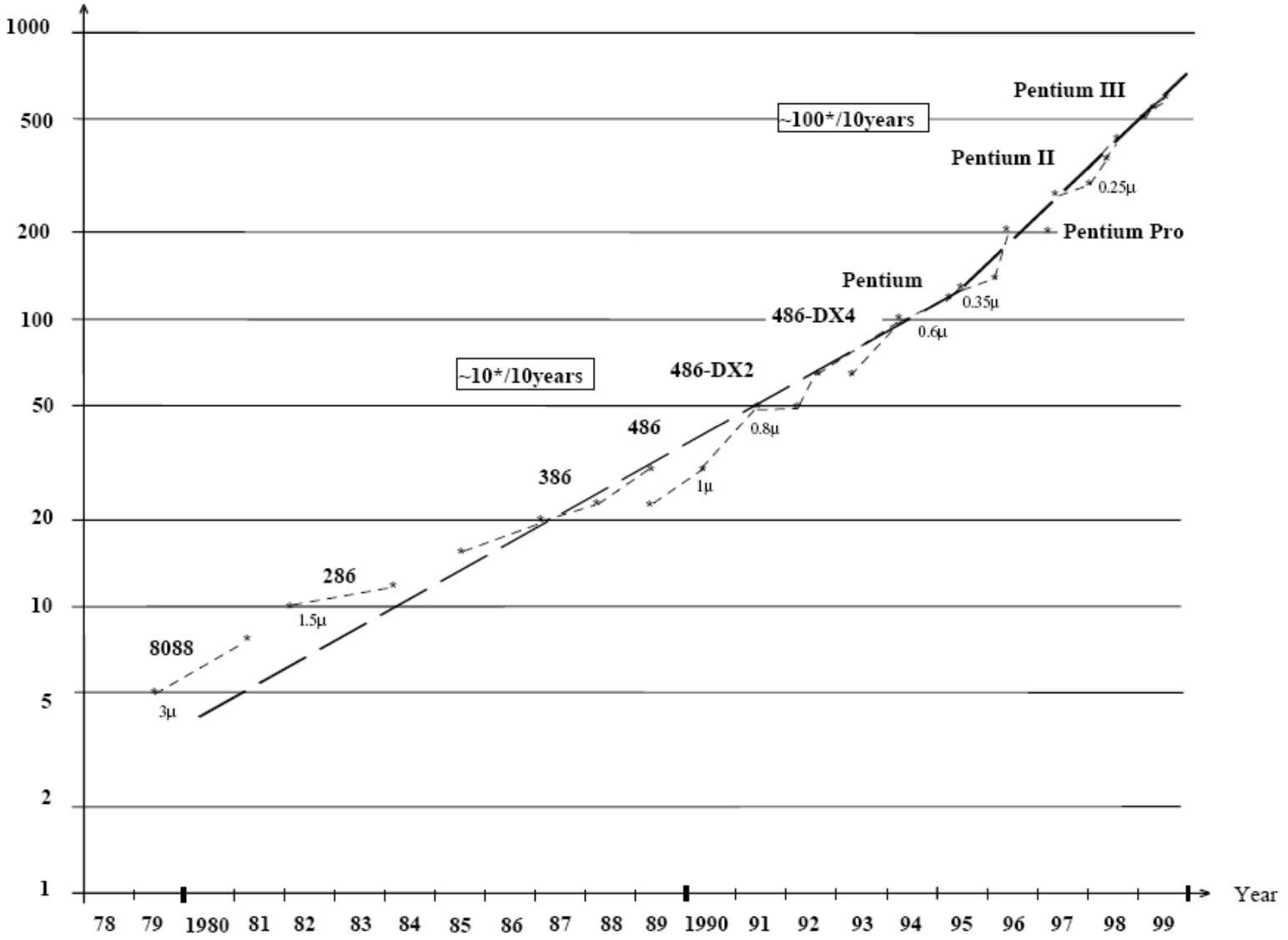
Microprocessor Transistor Count

[Weste & Harris]



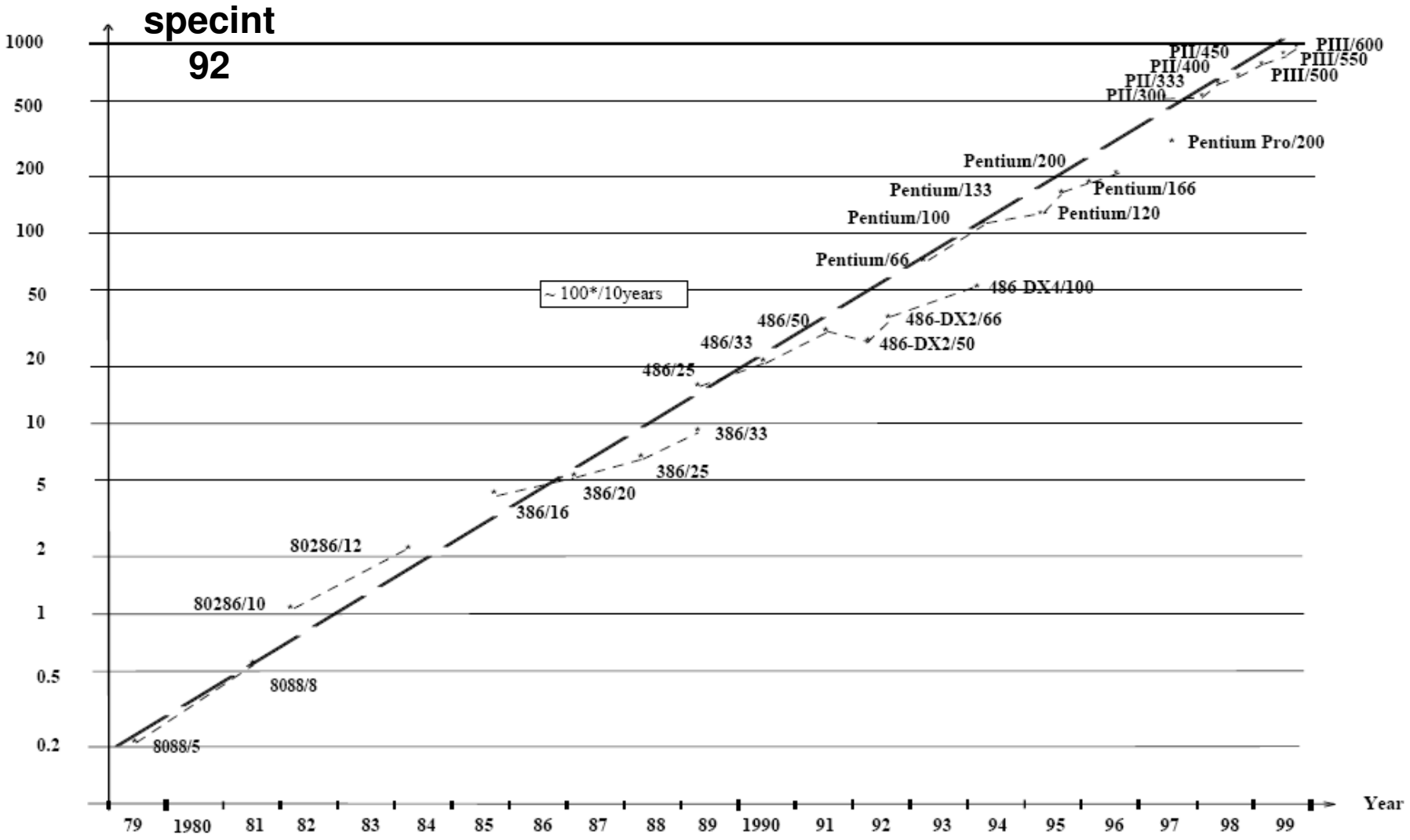
Microprocessor Clock Frequency

[Sima]



Microprocessor Integer Performance

[Sima]

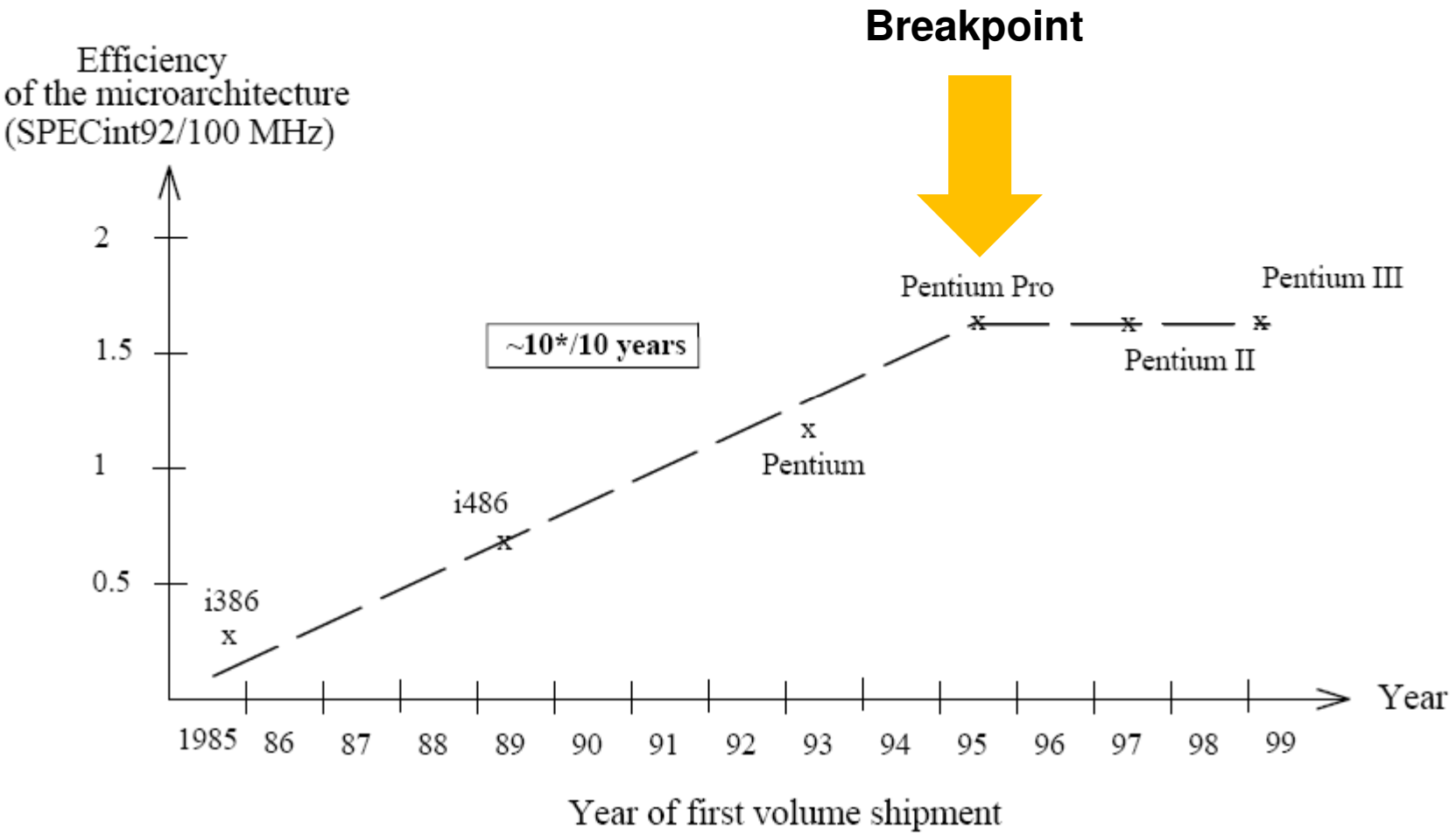


Date of first volume shipments
(P denotes Pentium)

EC
Le

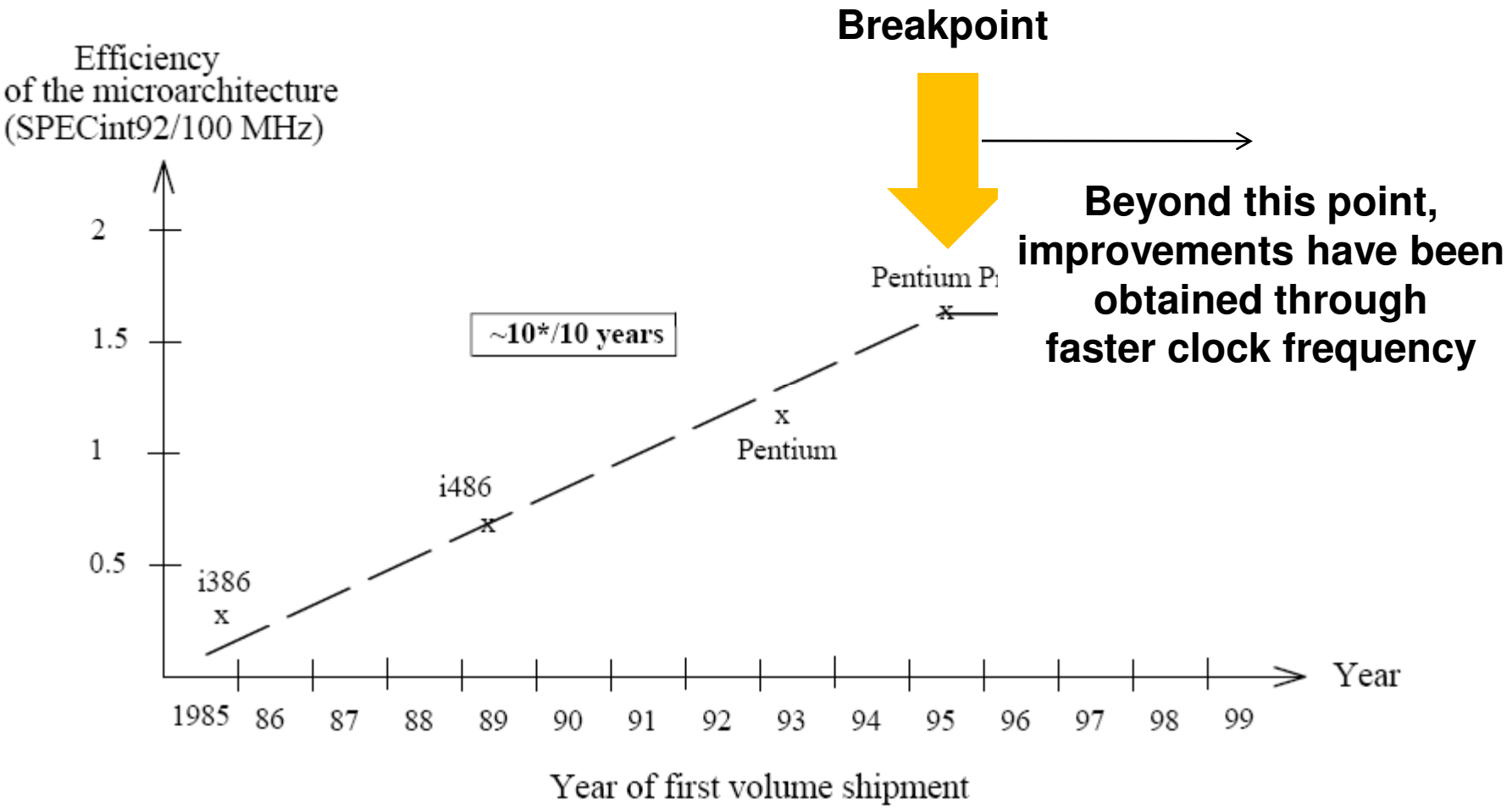
Microprocessor Architecture Efficiency

[Sima]



Microprocessor Architecture Efficiency

[Sima]



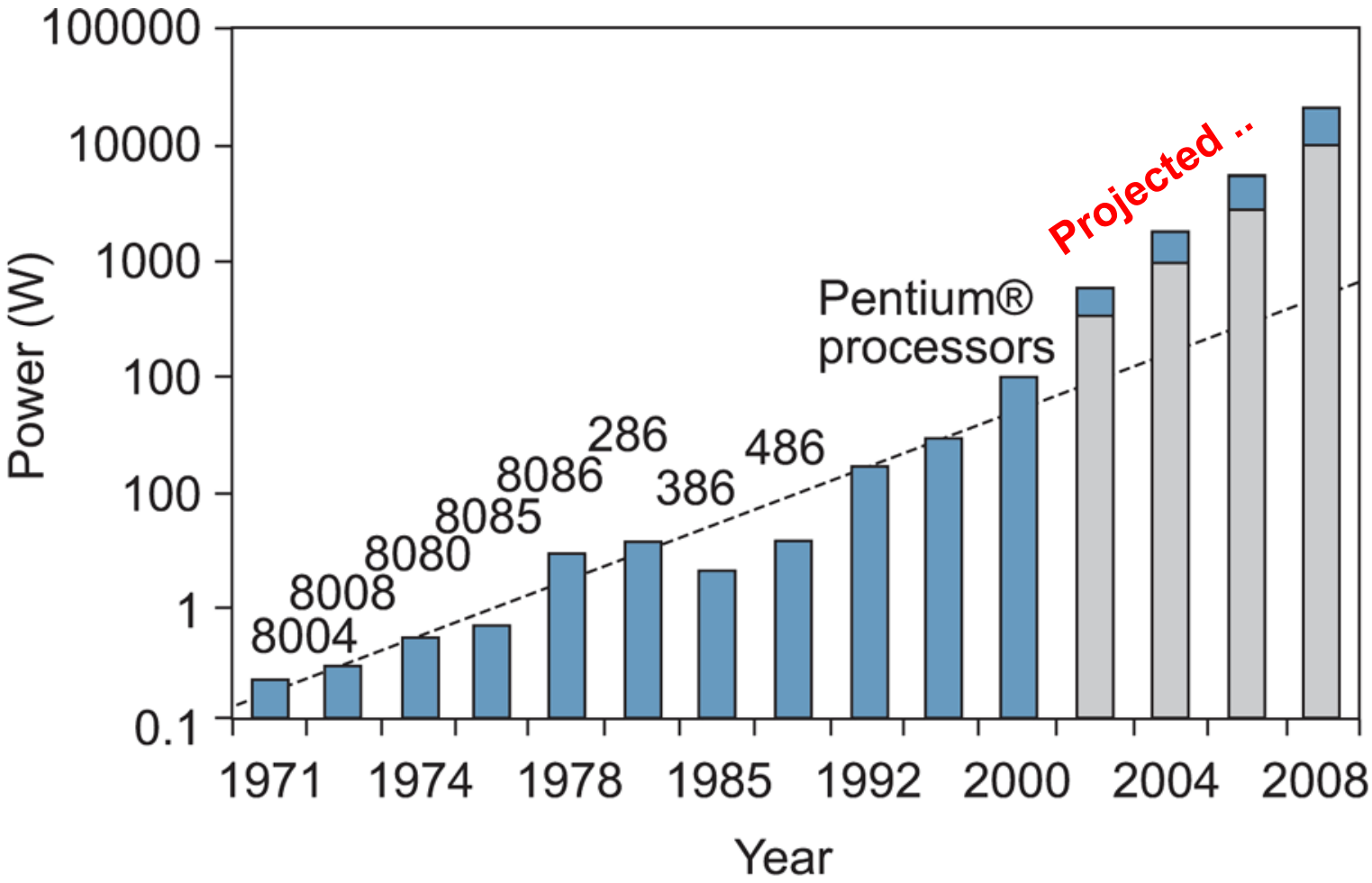
The Woes of scaling

- Scaling provides more transistors that operate faster
  - Can do more operations per second
- However
  - More transistors working faster will consume more power (even when they're smaller)
  - Low-level electrical effects become dominant
  - Wires do not improve with scaling

Processor Power Consumption

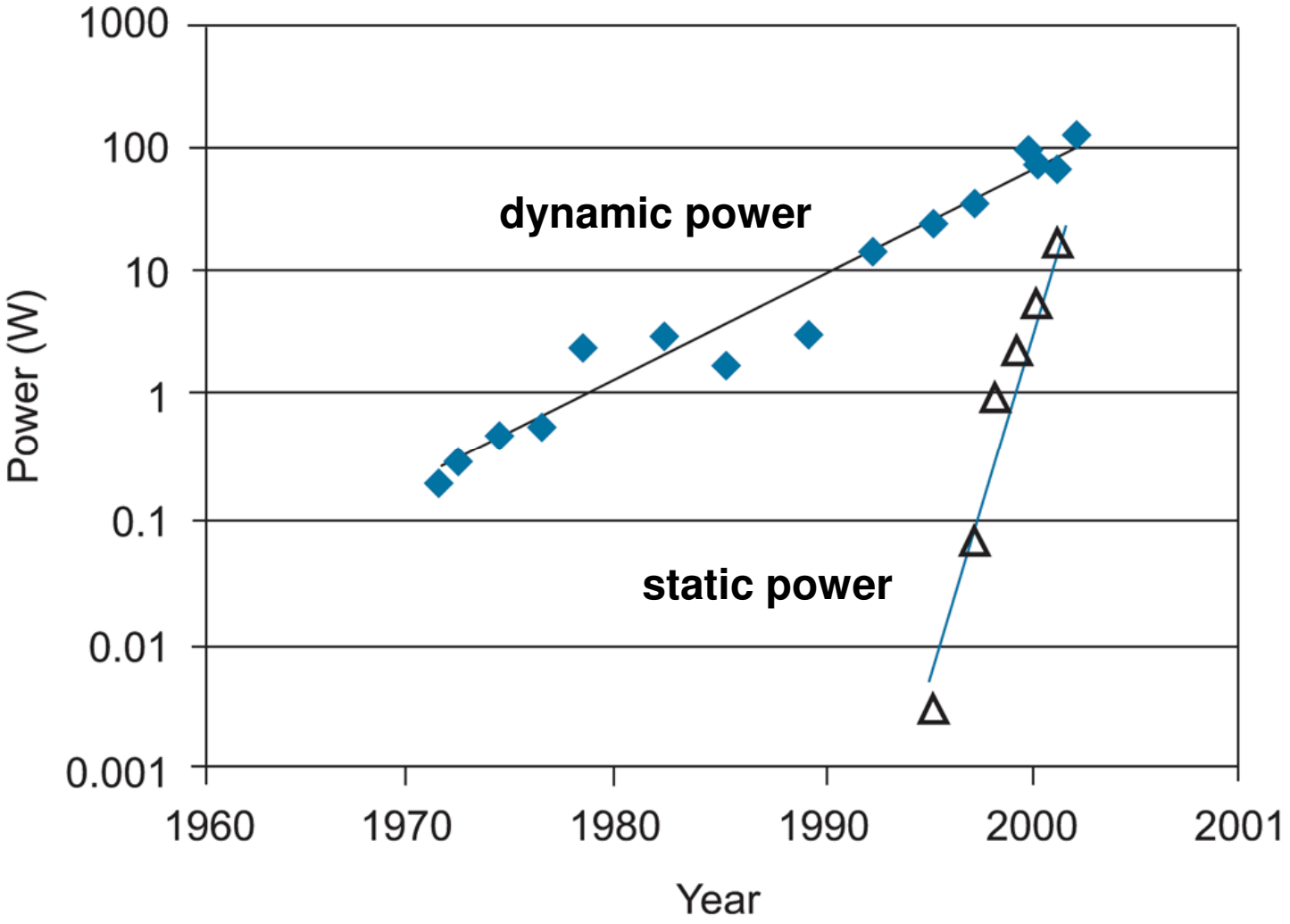
[Weste & Harris]

$$P_{dyn} = a.\ C\ .\ V^2\ .\ f_{clk}$$

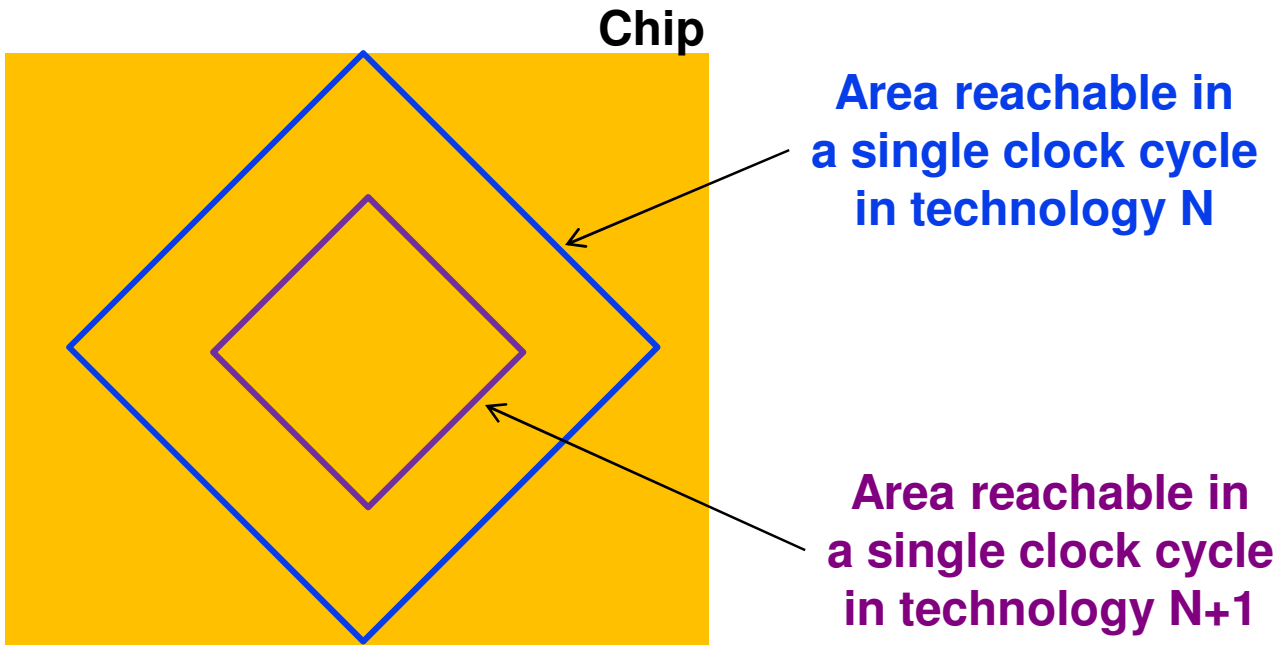


Low level electrical effects become dominant

[Weste & Harris]



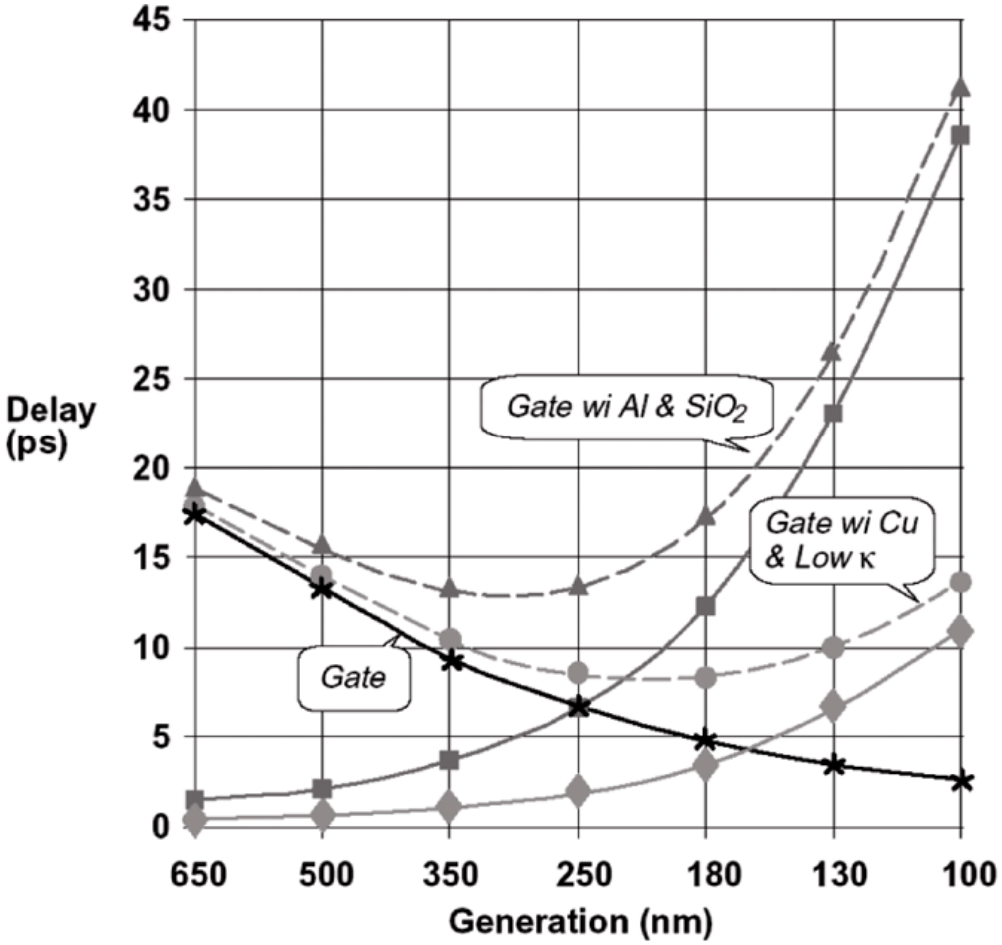
Wires do not improve with scaling



Caused by scaled wire geometry (R increases)

Wires even become slower than gates

[SIA97]



Al	*3.0 μΩ –cm*
Cu	*1.7 μΩ –cm*
SiO_2	*κ = 4.0*
Low κ	*κ = 2.0*
Al & Cu	*.8 μ Thick*
Al & Cu Line	*43 μ Long*

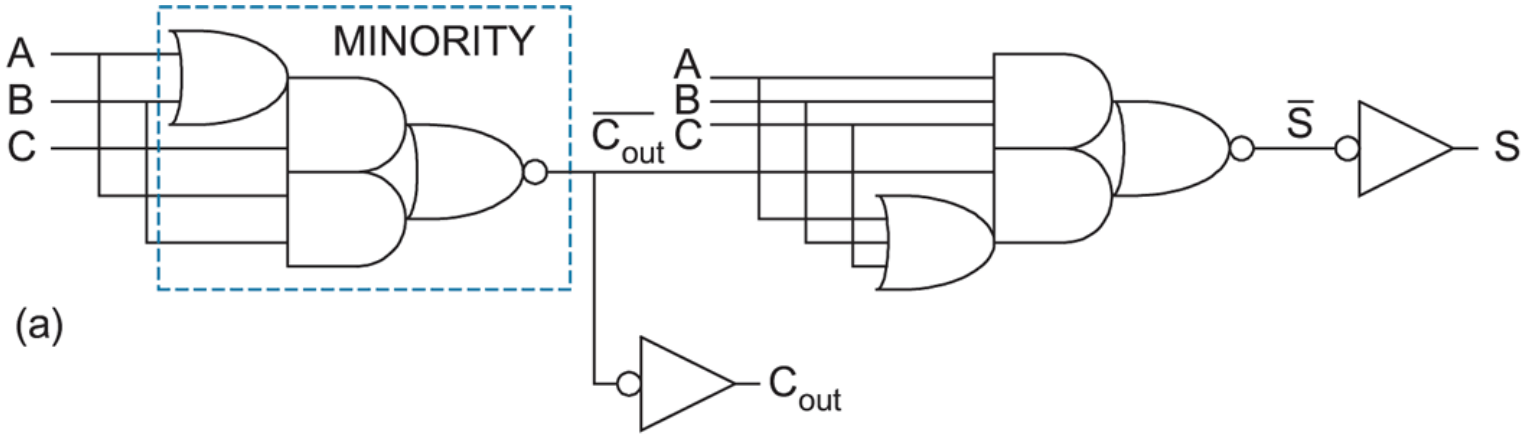
medium-long wire .. (43u @ 90nm = 500 transistors long)

Can Microprocessors keep up with scaling?

- Let's first see how efficient microprocessors really are.
- We will try to evaluate how well a microprocessor makes use of transistors, i.e. how performance and number of transistors relate
- We will compare that to the performance of a dedicated hardware circuit
- Just a back-of-the-envelope calculation ..

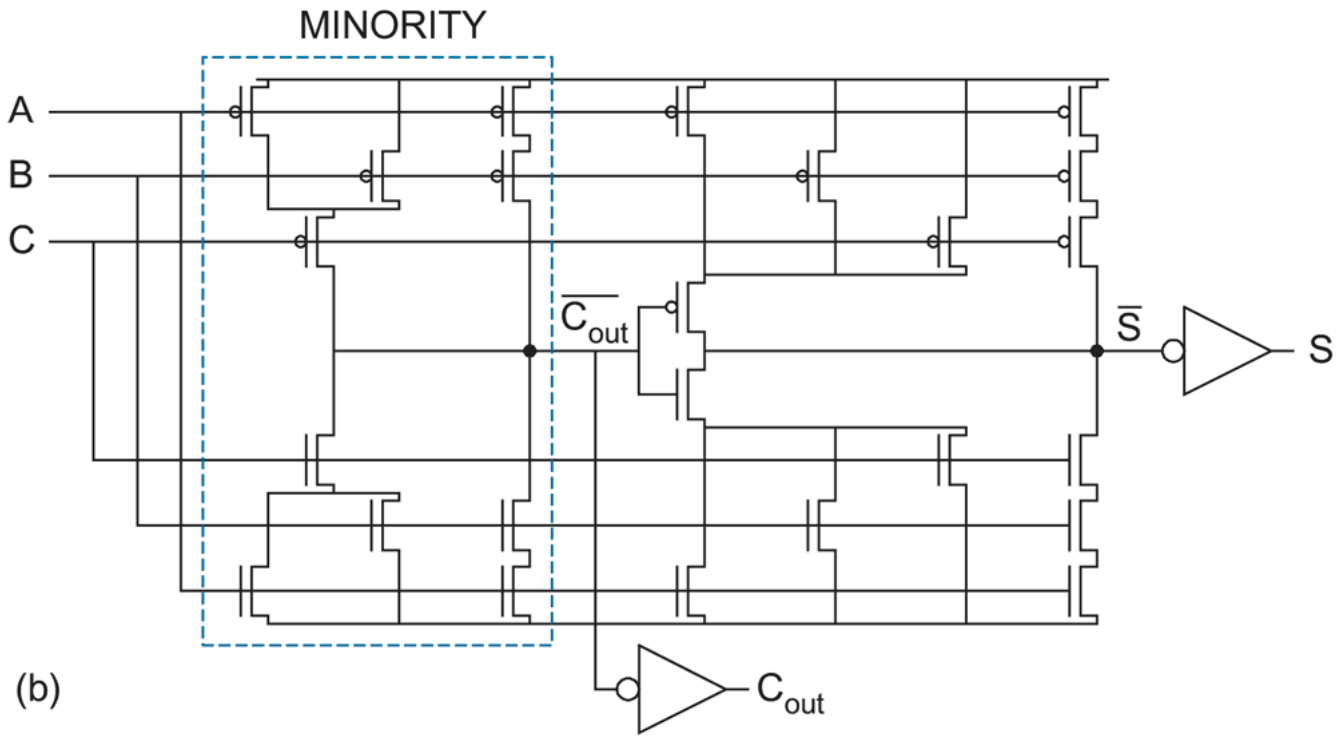
Performance

- Performance = [Operations / Second]
- Here's a full adder cell
  - Input A, B, C (carry in)
  - Output Cout, S
  - 10 gates



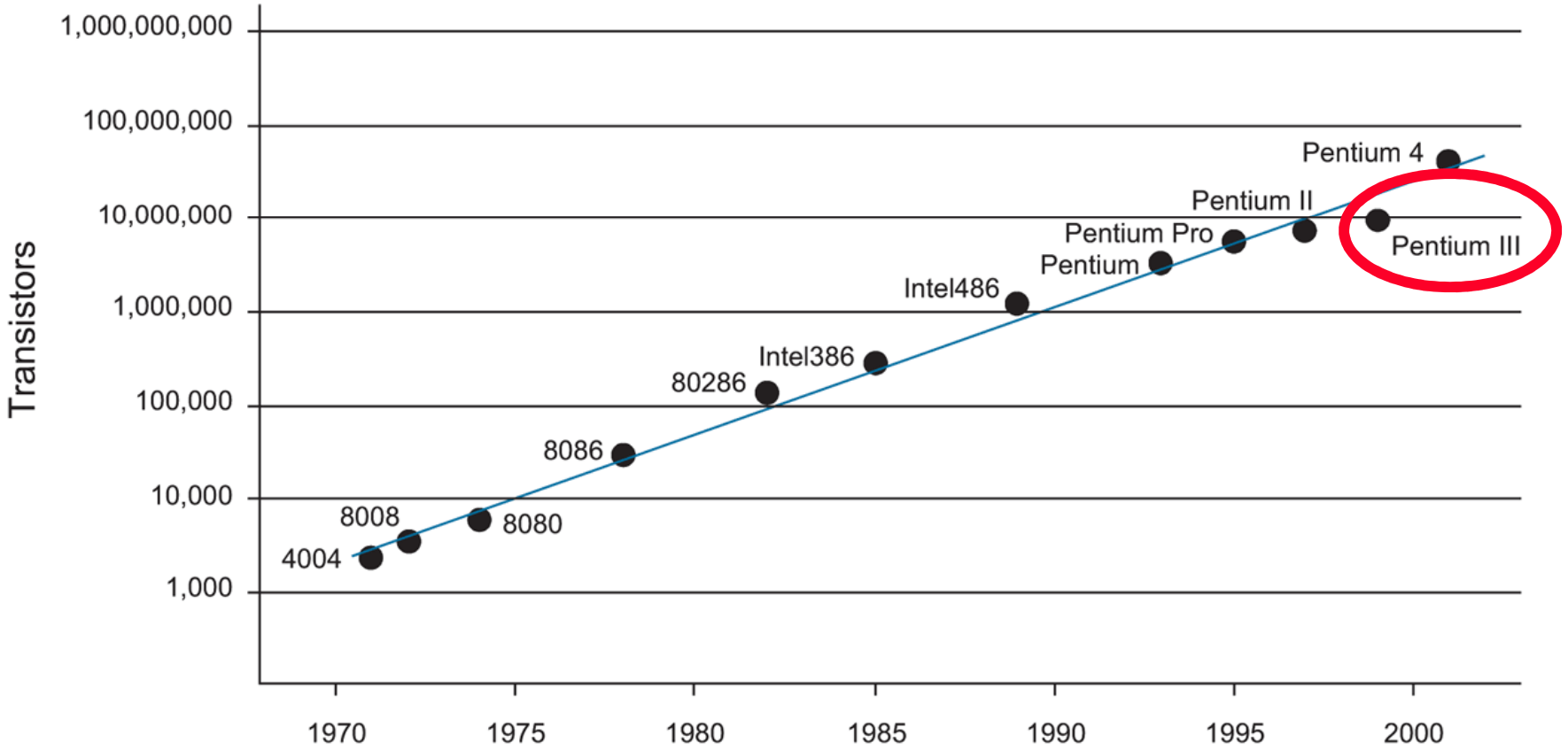
Performance

- Here's the same full adder cell, with transistors
  - Input A, B, C (carry in)
  - Output Cout, S
  - 28 transistors (~ 2.8 / gate)



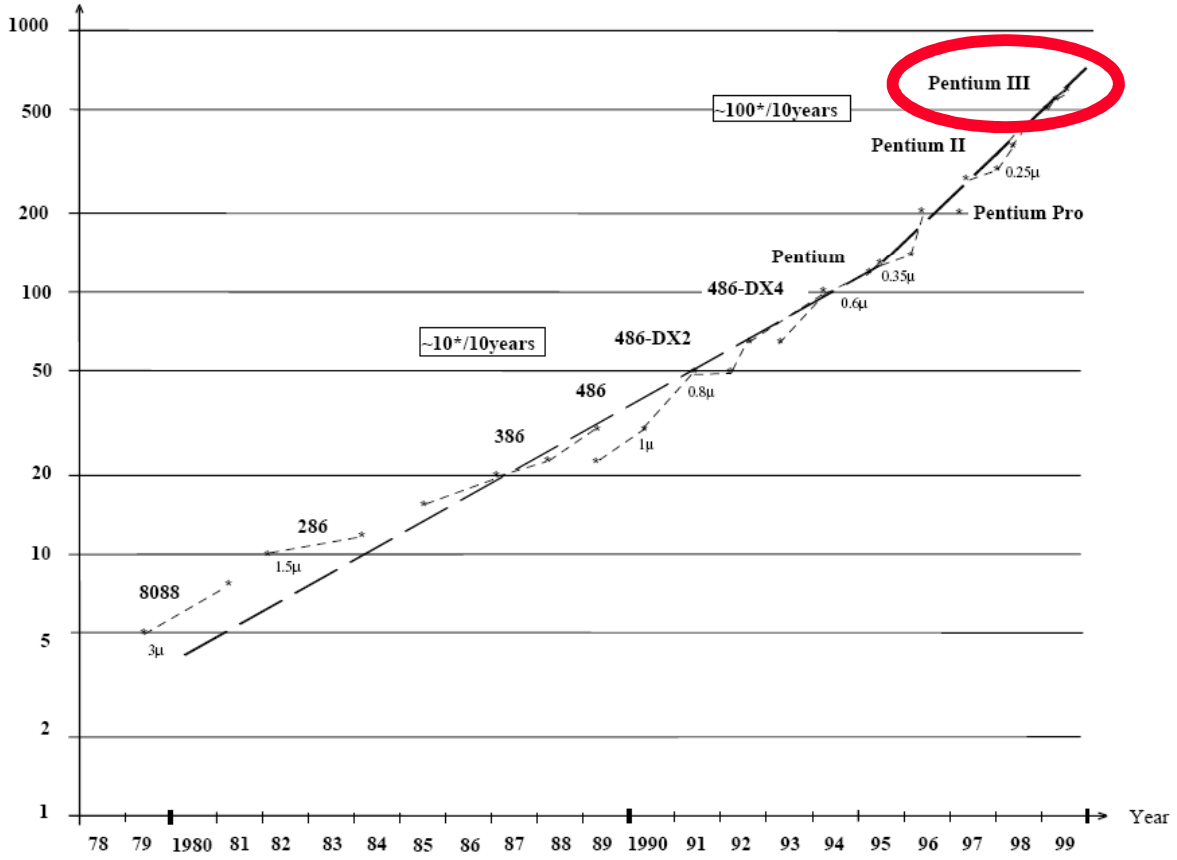
Performance

- Let's see how well the transistors in a Pentium III operate
  - Approx 10 million transistors (note: a Core2 has ~ 400 million..)



Performance

- Let's see how well the transistors in a Pentium III operate
  - Approx 10 million transistors
  - Chip frequency is 500 MHz



Performance

- Let's see how well the transistors in a Pentium III operate
  - Approx 10 million transistors
  - Chip frequency is 500 MHz
  - So a 'perfect' P-III could do N 1-bit additions per second
    - N = # gates / (gates/addition) x fclk
    - N = 10.10^6 / 28 x 500 x 10^6
  - We need 32 additions for an ALU operation
  - So the P-III has enough transistors to do N/32 = **5580 Giga Operations per Second**!

But a 500 MHz P-III gets you about 500 MIPS ..

Family	Trade Name (Code Name for Future Chips)	Clock Frequency in MegaHertz**	Millions of Instructions per Second	Date of Introduction	Number of Transistors	Design Rule (Pixel Size)	Address Bus Bits
80986	Projected Roadmap	24,000.0 MHz	+125,000. MIPS	2007	1 billion	0.045 micron	64 bit
80886	(Northwood)	3,000.0 MHz	TBA	2003	TBA	0.13 micron	64 bit
80886	(Madison)	TBA	TBA	2003	TBA	0.13 micron	64 bit
80886	(Deerfield)***	TBA	TBA	2002Q2	TBA	0.13 micron	64 bit
80886	(McKinley)	1,000.0 MHz	TBA	2002Q1	TBA	0.18 micron	64 bit
80786	Itanium (Merced)	800.0 MHz	+2,500.00 MIPS	May 29, 2001	30 / 300 M	0.18 micron	64 bit
80686	Pentium 4	1,500.0 MHz	*1,500.00 MIPS	November 20, 2000	42 million	0.18 micron	32 bit
80686	Pentium III	1,000.0 MHz	*1,000.00 MIPS	March 1, 2000	28.1 million	0.18 micron	32 bit
80686	P III Xeon	733.0 MHz	*733.00 MIPS	October 25, 1999	28.1 million	0.18 micron	32 bit
80686	Mobile P II	400.0 MHz	*400.00 MIPS	June 14, 1999	27.4 million	0.18 micron	32 bit
80686	P III Xeon	550.0 MHz	*550.00 MIPS	March 17, 1999	9.5 million	0.25 micron	32 bit
80686	Pentium III	500.0 MHz	*500.00 MIPS	February 26, 1999	9.5 million	0.25 micron	32 bit
80686	P II Xeon	400.0 MHz	*400.00 MIPS	June 29, 1998	7.5 million	0.25 micron	32 bit
80686	Pentium II	333.0 MHz	*333.00 MIPS	January 26, 1998	7.5 million	0.25 micron	32 bit
80686	Pentium II	300.0 MHz	*300.00 MIPS	May 7, 1997	7.5 million	0.35 micron	32 bit
80586	Pentium Pro	200.0 MHz	*200.00 MIPS	November 1, 1995	5.5 million	0.35 micron	32 bit
90586	Pentium	133.0 MHz	*133.00 MIPS	June 1995	3.3 million	0.35 micron	32 bit
80586	Pentium	90.0 MHz	*90.00 MIPS	March 7, 1994	3.2 million	0.60 micron	32 bit
80586	Pentium	60.0 MHz	*60.00 MIPS	March 22, 1993	3.1 million	0.80 micron	32 bit
80486	80486 DX2	50.0 MHz	*50.00 MIPS	March 3, 1992	1.2 million	0.80 micron	32 bit
80486	486 DX	25.0 MHz	20.00 MIPS	April 10, 1989	1.2 million	1.00 micron	32 bit
80386	386 DX	16.0 MHz	5.00 MIPS	October 17, 1985	275,000	1.50 micron	16 bit
80286	80286	6.0 MHz	0.90 MIPS	February 1982	134,000	1.50 micron	16 bit
8086	8086	5.0 MHz	0.33 MIPS	June 8, 1978	29,000	3.00 micron	16 bit
8080	8080	2.0 MHz	0.64 MIPS	April 1974	6,000	6.00 micron	8 bit
8008	8008	.2 MHz	0.06 MIPS	April 1972	3,500	10.00 micron	8 bit
4004	4004	.1 MHz	0.06 MIPS	November 15, 1971	2,300	10.00 micron	4 bit

Performance

- So, the potential parallellism of the P-III silicon is 10,000X higher then the achieved parallellism
  - Theoretically we should get about 5000 Gops
  - Practically we get 500 Mops
- Where do we waste all this performance?
  - Sequential Execution of Instructions
  - Memory-Processor Bottleneck
- Bottom line: Raw hardware can be *far more* computationally efficient then what can be achieved with software on a Microprocessor
  - However, making use of all that parallelism is the core of the problem

Alternatives over a Microprocessor ?

- Possibly many, but let's just consider
  - Field Programmable Gate Array
  - ASIC
  - Full-custom

- Note that
  - FPGA = design + program ready-made chip
  - ASIC = design + fabricate custom chip
  - Full-custom = design + fabricate custom chip

Micro, FPGA, ASIC, Full Custom

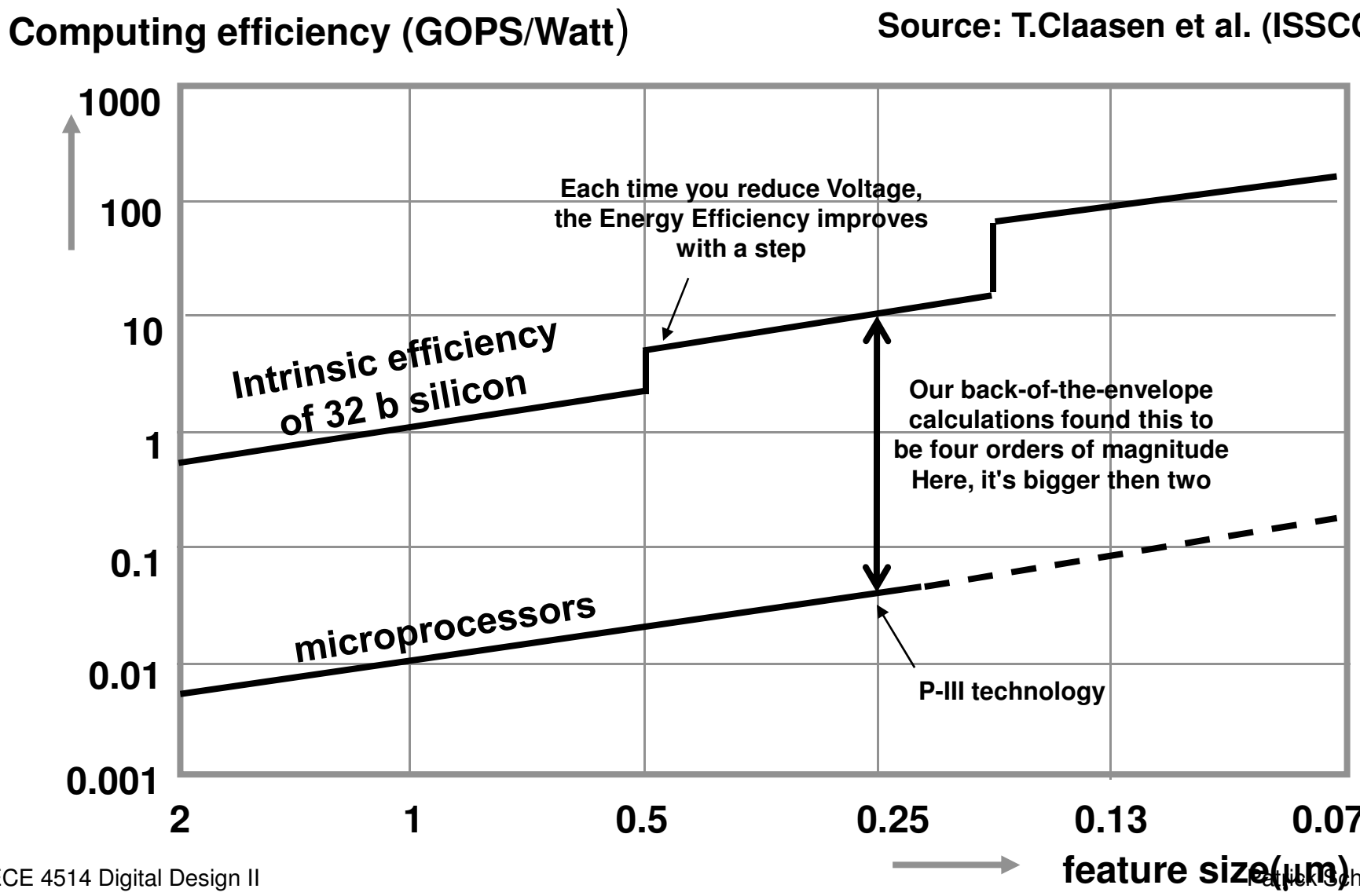
- Let's compare these four target technologies on the basis of two arguments (which is incomplete ..)
  - Energy Efficiency: the amount of operations per unit of energy
  - Design Cost: the amount of $ per chip

Energy Efficiency

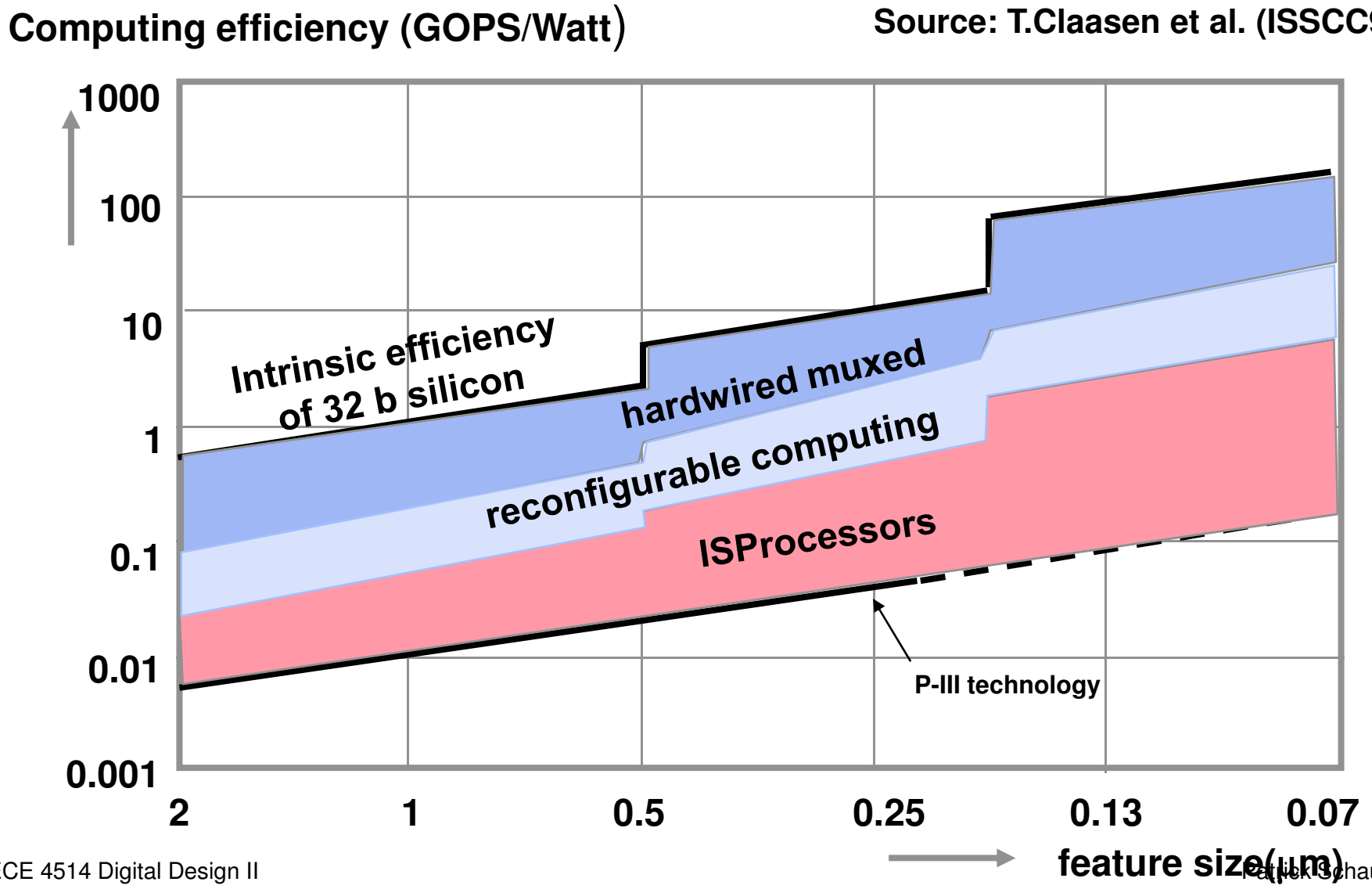
- Many applications need high performance but cannot afford high power consumption: cell phones, ipods, laptops, portable electronics, ..

- Energy Efficiency = [Operations / Joule]
  - = [Operations / Second] / [Joule / Second]
  - = Performance / Power

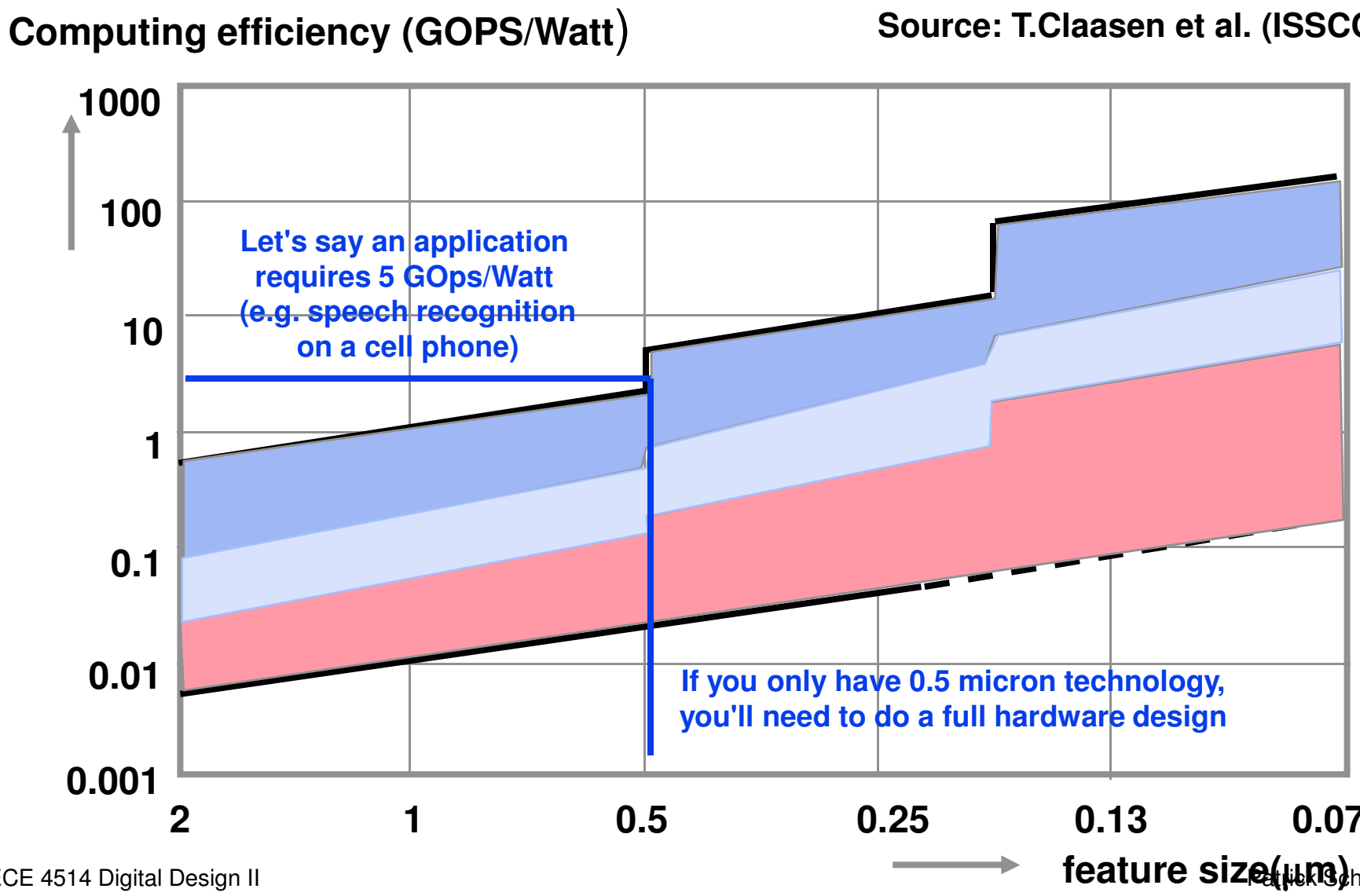
Energy Efficiency



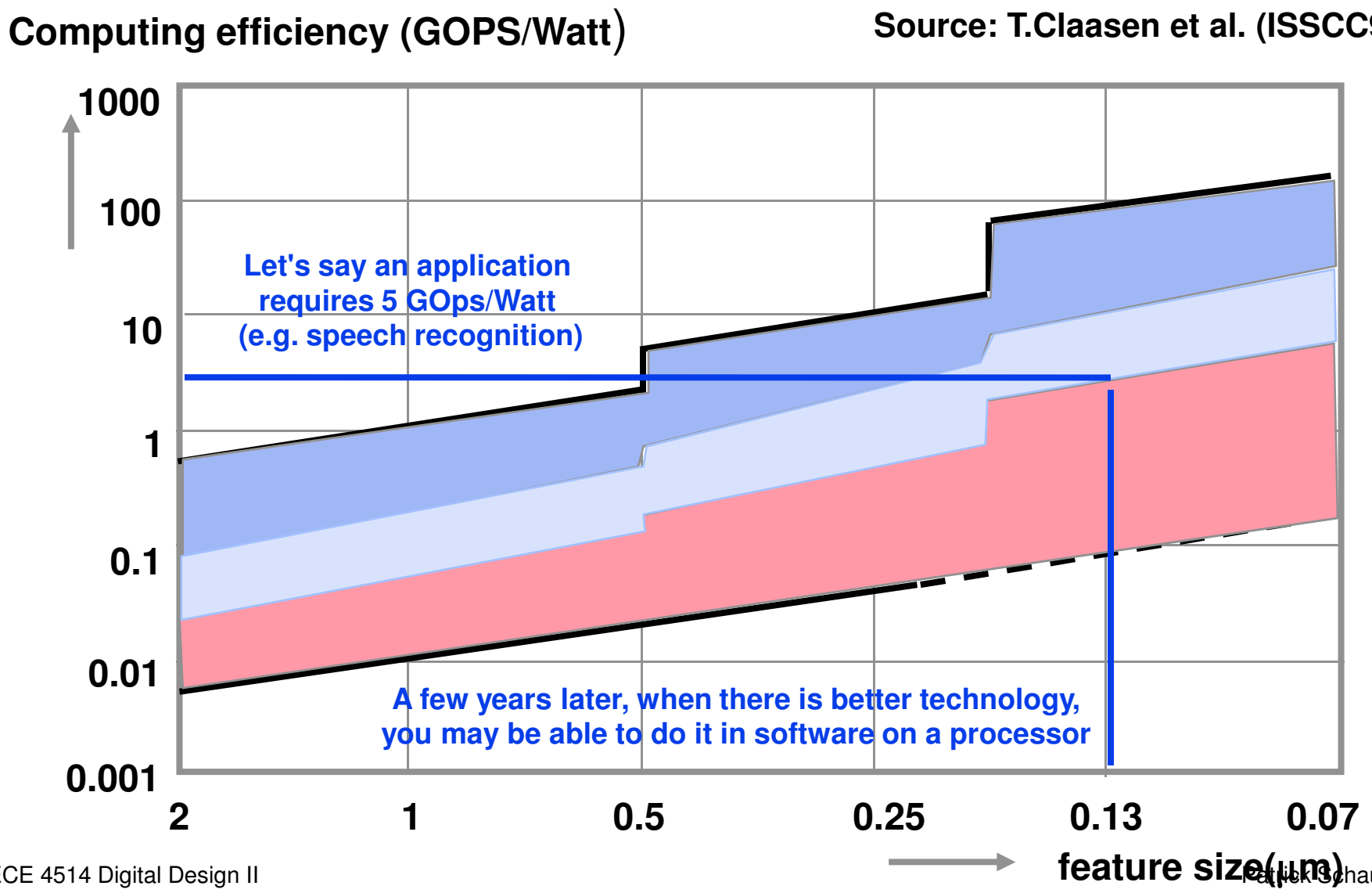
The Energy-Flexibility Conflict



The Energy-Flexibility Conflict



The Energy-Flexibility Conflict



Cost

- The cost of a microprocessor is just a per-part cost, maybe with some additional tools (GUI, compiler, etc, ..)
  - Eg. typical 32-bit microprocessor ~ $15 in small quantities
- The cost of an FPGA is also just a per-part cost.
  - Eg. Spartan3E S500 ~ $30 in small quantities

Cost of ASIC and Full-Custom Design

- Total Cost of Design has two components:
  - Non-recurring engineering costs (NRE): Costs independent of the number of components sold
  - Recurring costs (R): Costs proportional to the number of components sold

- Total Cost of Design = NRE + n . R, with n the number of components sold.
  - For high-volume products (e.g. RAM, Processors), n is very high, and the role of NRE is less important
  - For low-volume products (e.g. Application-specific designs), n is low, and the role of NRE is more important

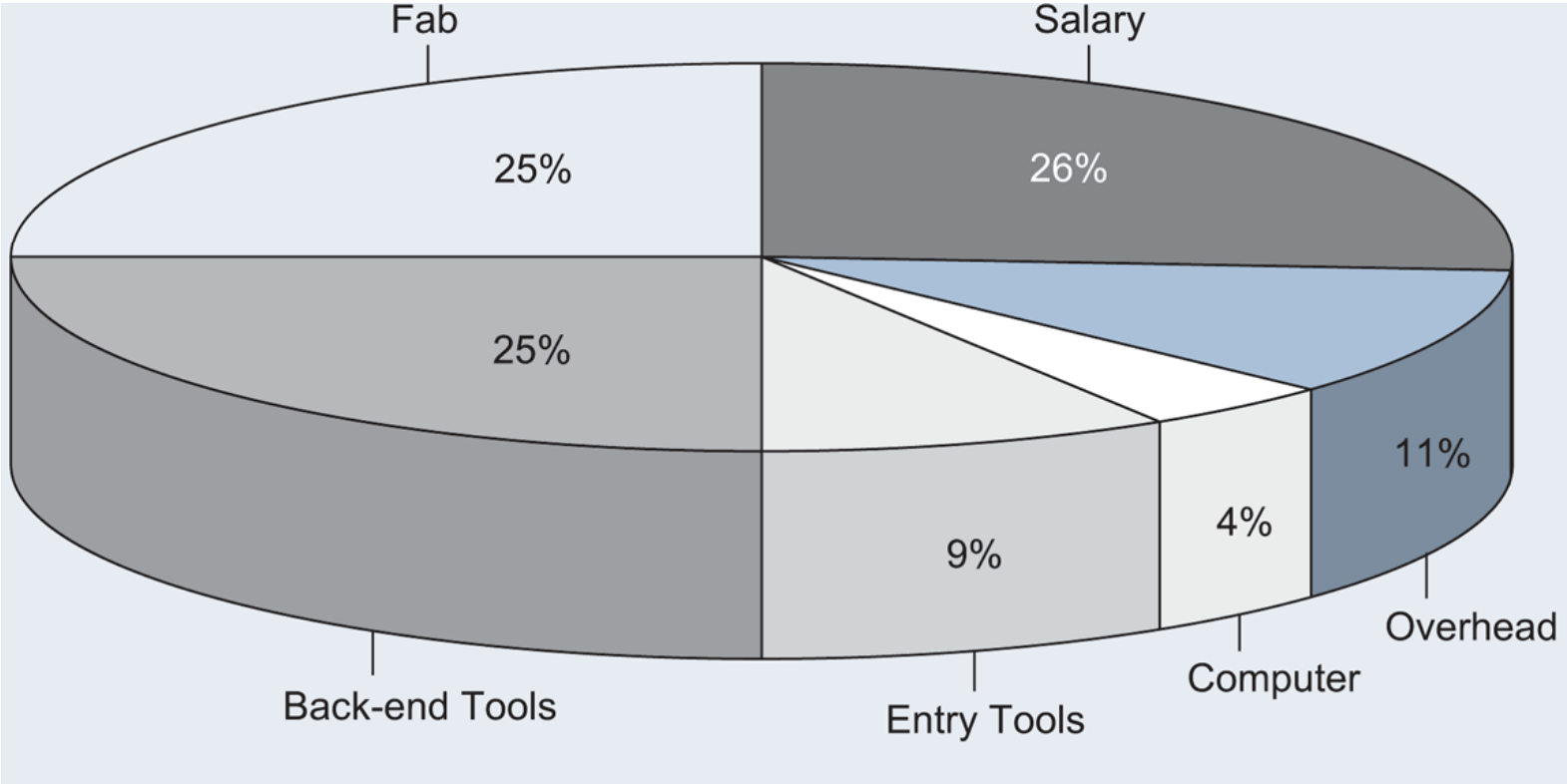
Cost

- NRE = Design + Prototype Creation
  - Design
    - Personnel and Engineering (~ $80K per MY)
    - Computers (~$5K per MY)
    - CAD Software ($10K .. $1M per seat)
  - Prototype Creation
    - For Custom chips (ASIC/Full Custom) ~ $1M
    - PCB Development
- R = Production Cost per component
  - Depends on process, package, testing complexity, yield

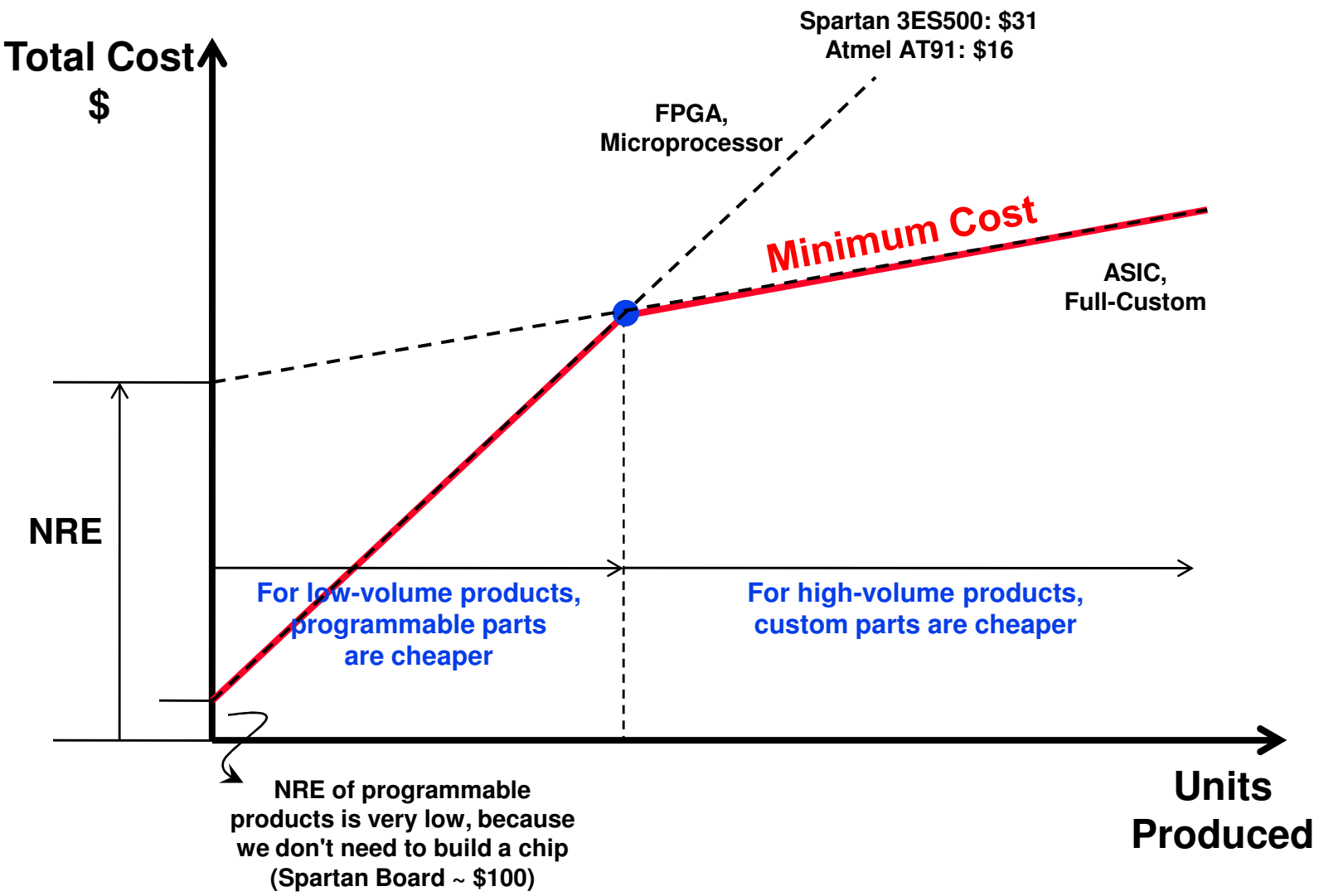
Typical NRE for medium-size custom chip

[Weste & Harris]

- (7 digital designers, 3 analog designers, 5 support)

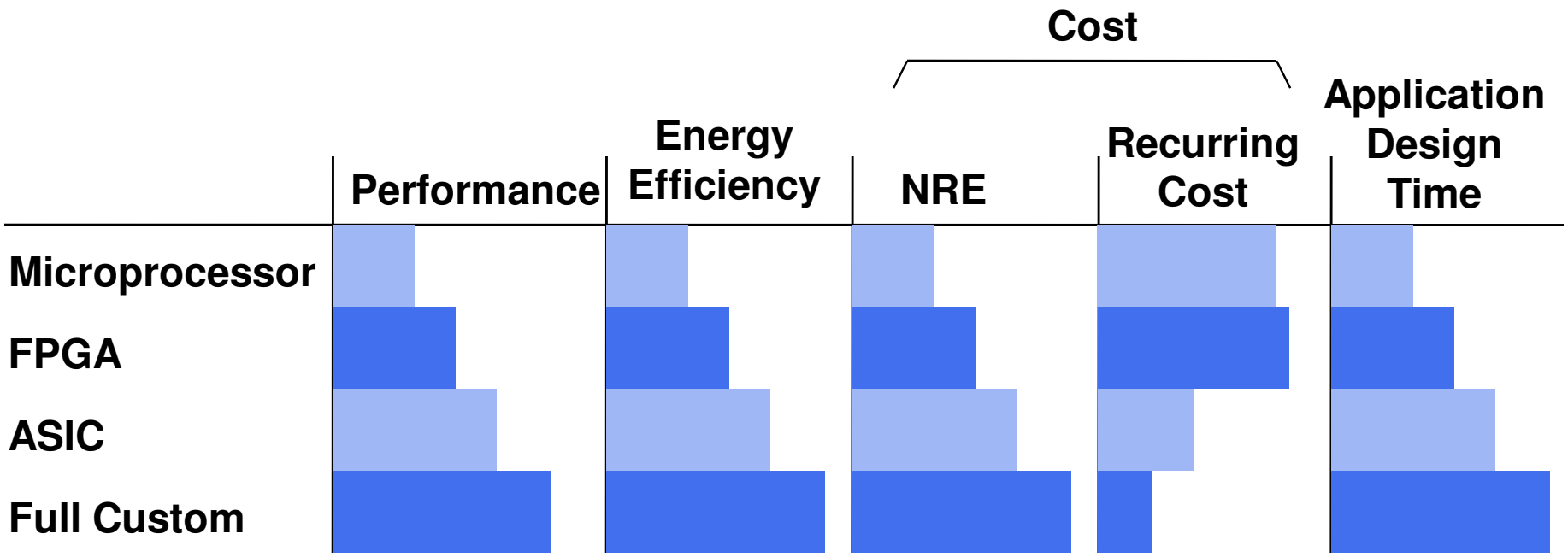


Cost


Spartan 3ES500: $31
Atmel AT91: $16
Total Cost
$
FPGA,
Microprocessor
Minimum Cost
ASIC,
Full-Custom
NRE
For low-volume products,
programmable parts
are cheaper
For high-volume products,
custom parts are cheaper
NRE of programmable
products is very low, because
we don't need to build a chip
(Spartan Board ~ $100)
Units
Produced

Performance, Efficiency, Cost, Design Time

- Given a single application implemented on Microprocessor, FPGA, ASIC, Full Custom, then a relative appreciation of these factors is as follows



FPGA vs Standard Cell ASIC

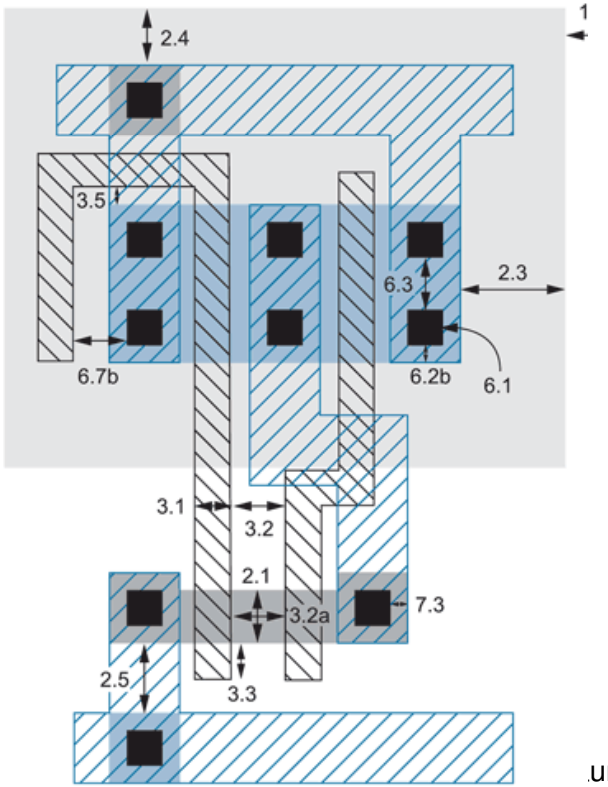
- Compared to FPGA, ASIC uses custom wiring pattern, and dedicated standard cells

- Comparison 90nm FPGA with 90nm ASIC [Kuon 06] :
  - Area difference **40X** (no custom blocks), **21 X** (with custom)
  - Performance (f_{max}) **4X**
  - Dynamic Power **12X**

- Note that FPGA technology tends to be 1 - 2 generations ahead of ASIC designs
  - E.g. Now (04/08) Virtex5 FPGA uses 65nm, while standard cell kit from IBM uses 130nm

Full Custom

- Instead of using standard cells, we can design a full chip at transistor level.
- Certain regular structures (memory cells, datapath bit-slices) can be made very dense in this manner
- Designers have to work according to 'design rules': specific topological constraints to guarantee that a resulting chip will work



Full Custom vs ASIC

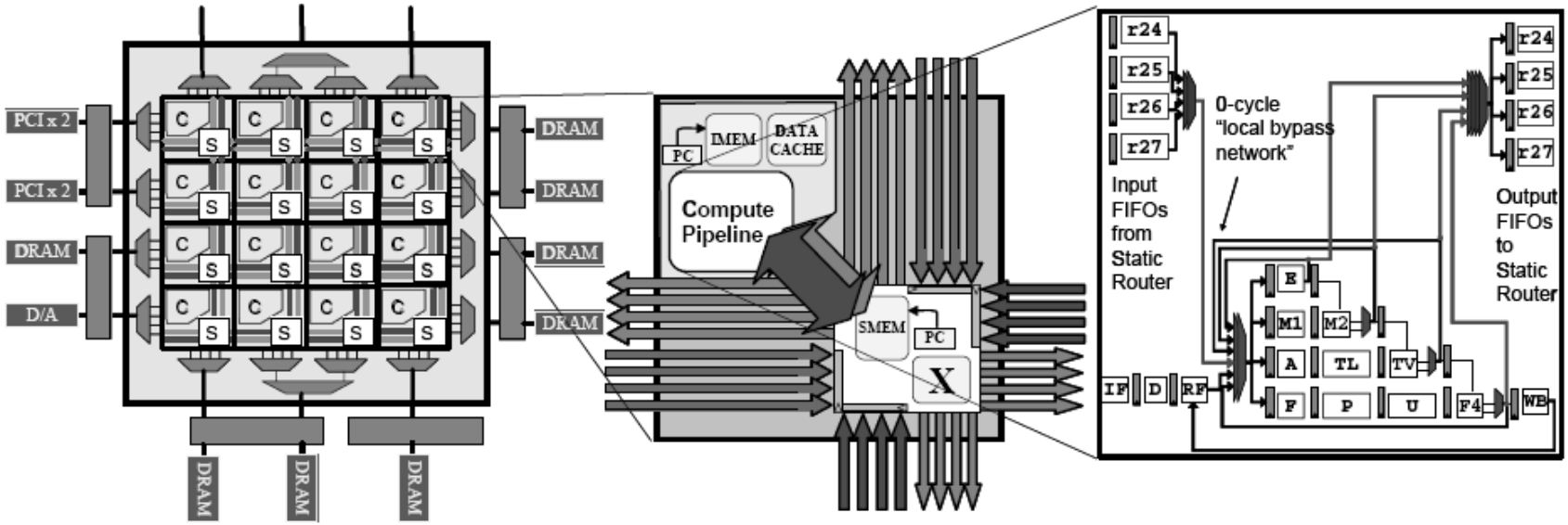
- Additional effort of full custom over ASIC yields better performance, area, and power consumption:
  - Speed: **3X .. 8X** times faster
  - Area: **15X** times smaller (depending on application)
  - Power: **3X .. 10X** times smaller

Technology Challenges for Digital Design

- How to design faster ?
  - Raise the abstraction level
  - Reuse
- How to build efficient on-chip communications?
  - On-chip busses
  - On-chip networks
- How to reduce power consumption ?

A general trend: more parallellism

- Example: RAW Microprocessor
  - Multiple CPU work in parallel, reduce f_{max}
  - CPUs pass around 'packets' among each other, chip traversal takes multiple clock cycles
  - Addresses Challenge #1 and #2
  - Needs 16 parallel programs to run ... (ouch)



Summary

- Wows and Woes of scaling
  - The case of the Microprocessor
  - How efficient does a microprocessor use transistors ?
- Alternative Technologies: FPGA, ASIC, Full Custom
  - Energy Efficiency and Design Cost
- ASIC Standard Cell Design
- Full Custom Design
- Future Challenges in Digital Design