CS321: Computer Architecture

Course web:

http://172.16.1.6/~jimson/

Computer Architecture

Basic functional blocks of a computer: CPU, memory, input-output subsystems, control unit. Instruction set architecture of a CPU - registers, instruction execution cycle, RTL interpretation of instructions, addressing modes, instruction set. Case study - instruction sets of some common CPUs; Assembly language programming for some processor; Data representation: signed number representation, fixed and floating point representations, character representation. Computer arithmetic - integer addition and subtraction, ripple carry adder, carry look-ahead adder, etc. multiplication – shift-and-add, Booth multiplier, carry save multiplier, etc. Division - non-restoring and restoring techniques, floating point arithmetic; CPU control unit design: hardwired and microprogrammed design approaches, Case study - design of a simple hypothetical CPU; Pipelining:

Basic concepts of pipelining, throughput and speedup, pipeline hazards; Memory organization: Memory interleaving, concept of hierarchical memory organization, cache memory, cache size vs block size, mapping functions, replacement algorithms, write policy; Peripheral devices and their characteristics: Input-output subsystems, I/O transfers - program controlled, interrupt driven and DMA, privileged and non-privileged instructions, software interrupts and exceptions. Programs and processes - role of interrupts in process state transitions.

Prerequisite & Text Book

- Prerequisite: **CS225 & CS226**

- Text Books
- David A. Patterson, John L. Hennessy, Computer Organization and Design, Fourth Edition:
- The Hardware/Software Interface, Morgan Kaufmann; 4 edition, 2011.
- 2. A. Tenenbaum, Structured Computer Organization, 4th Ed, Prentice-Hall of India, 1999.
- 3. W. Stallings, Computer Organization and Architecture: Designing for Performance, 6th
- Ed, Prentice Hall, 2005.
- 4. J. Hennessy and D. Patterson, Computer Architecture A Quantitative Approach, 3rd Ed,
- Morgan Kaufmann, 2002..

Evaluation Policy(CS321)

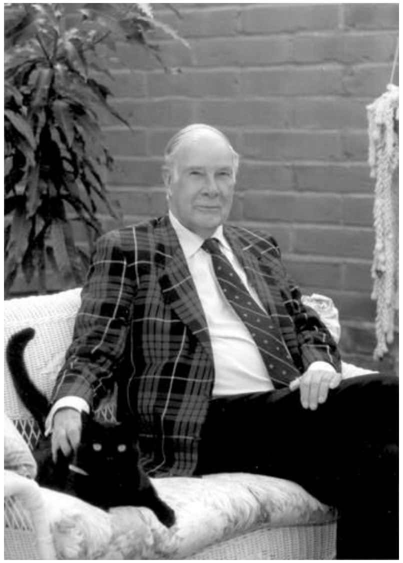
- Assignments
  & Quizzes : 40%
- Mid-Sem : 25%
- End-Sem : 25%
- Performance:10 %

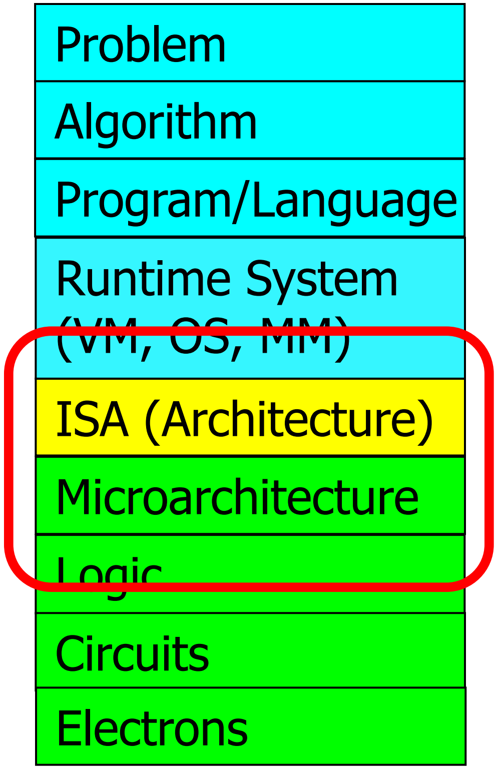
Architecture Lab(CS322)

- Regular Lab work, 40%
- Mid-Sem : 15%
- End-Sem : 25%
- Mini Project:45 %

Levels of Transformation

“The purpose of computing is insight” (*Richard Hamming*)
We gain and generate insight by solving problems
How do we ensure problems are solved by electrons?





Problem

Algorithm

Program/Language

Runtime System (VM, OS, MM)

ISA (Architecture)

Microarchitecture

Logic

Circuits

Electrons

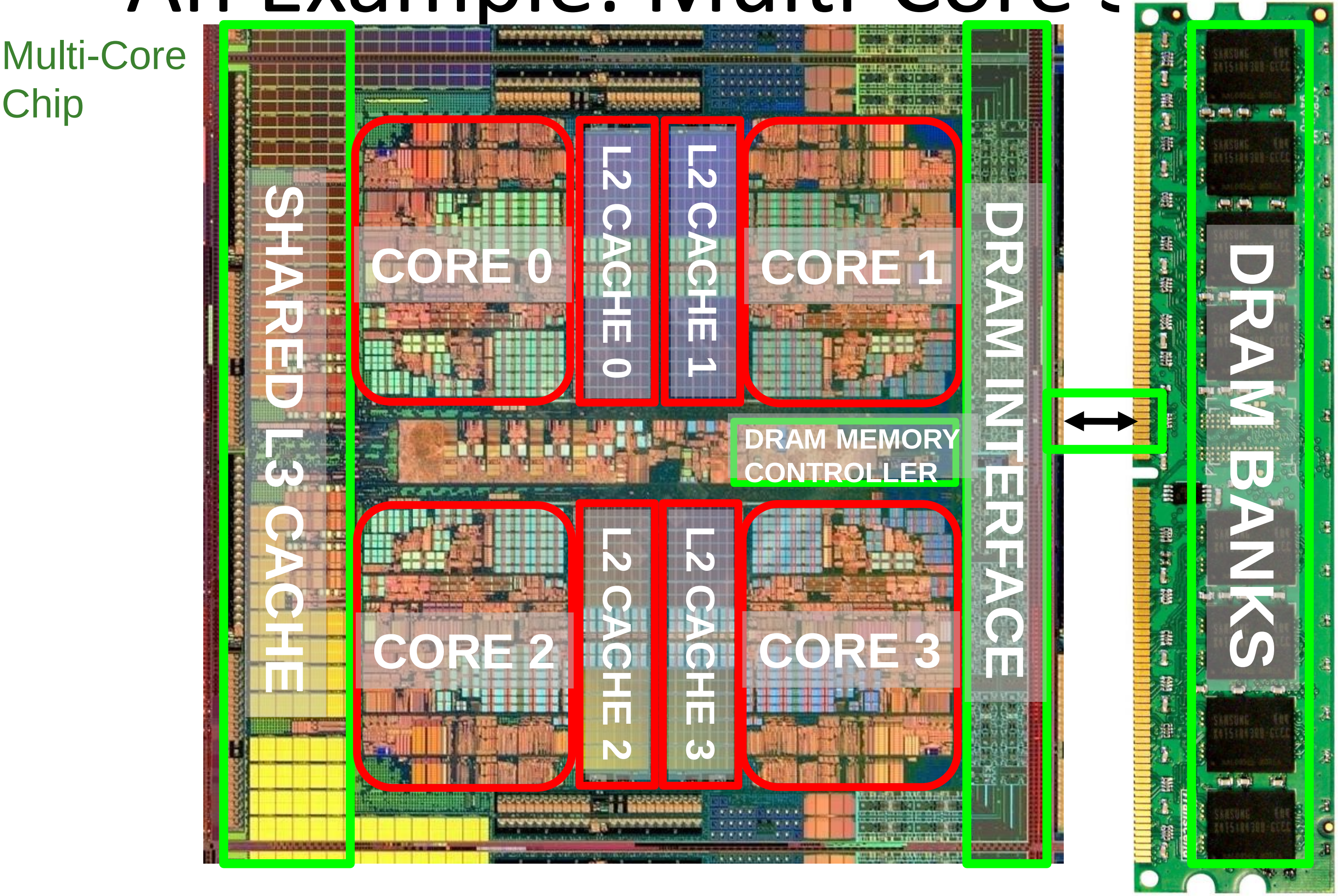
The Power of Abstraction

- Levels of transformation create abstractions
  - Abstraction: A higher level only needs to know about the interface to the lower level, not how the lower level is implemented
  - E.g., high-level language programmer does not really need to know what the ISA is and how a computer executes instructions
- Abstraction improves productivity
  - No need to worry about decisions made in underlying levels
  - E.g., programming in Java vs. C vs. assembly vs. binary vs. by specifying control signals of each transistor every cycle
- Then, why would you want to know what goes on underneath or above?

Why CS321?

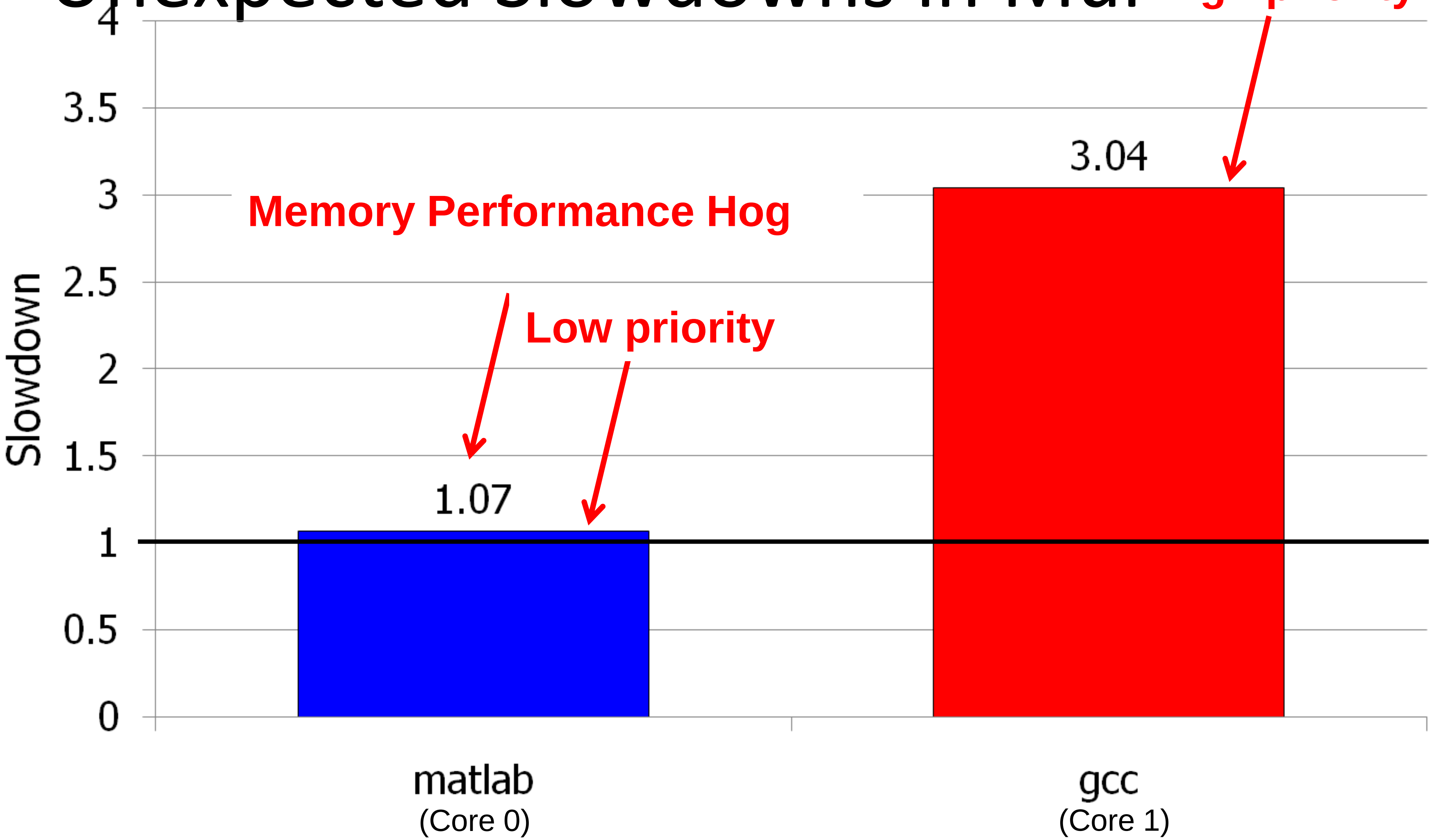
- What if
  - The program you wrote is running slow?
  - The program you wrote does not run correctly?
  - The program you wrote consumes too much energy?
- What if
  - The hardware you designed is too hard to program?
  - The hardware you designed is too slow because it does not provide the right primitives to the software?

An Example: Multi-Core Systems



*Die photo credit: AMD Barcelona

Unexpected Slowdowns in Mul



High priority

Memory Performance Hog

Low priority

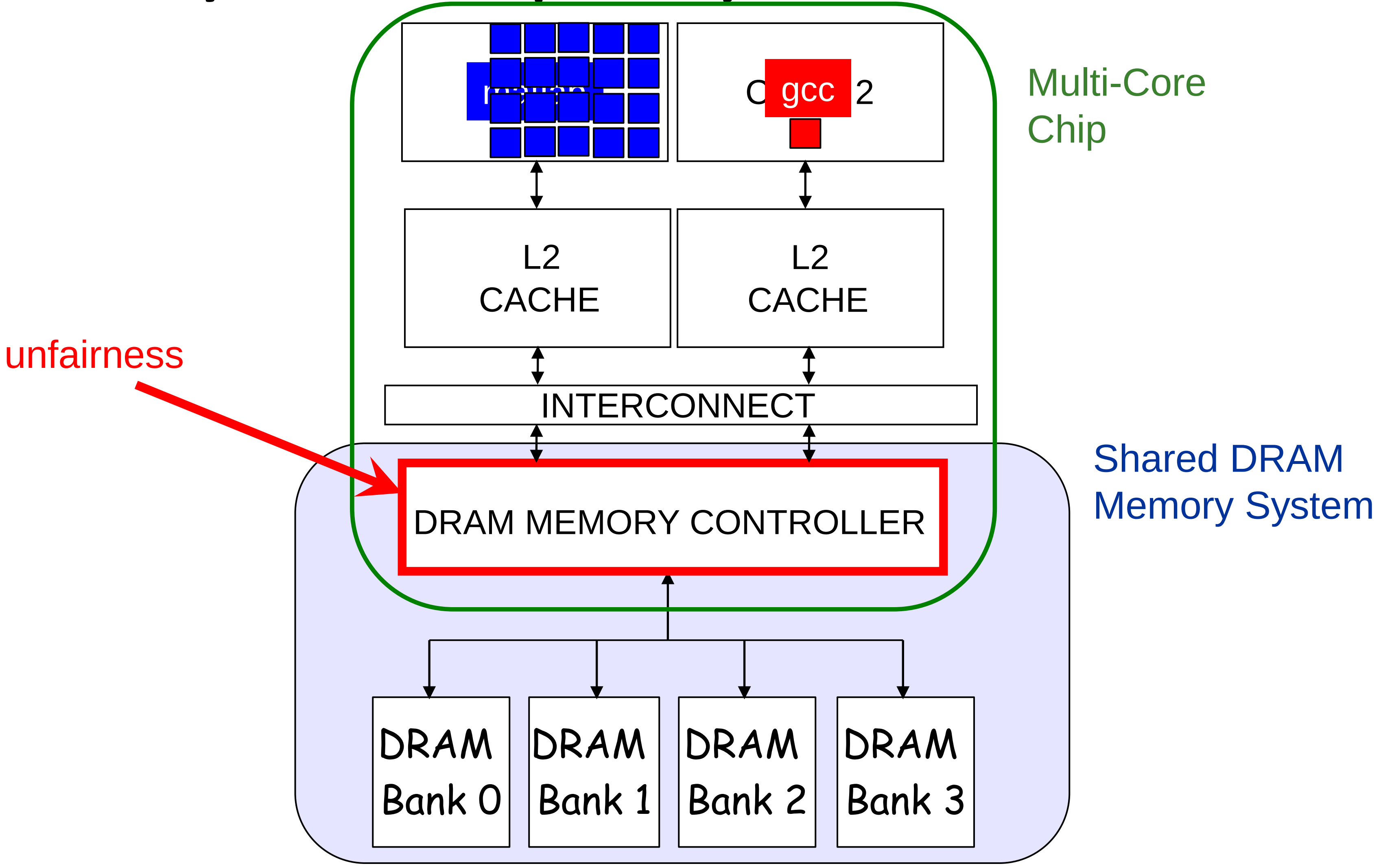
Moscibroda and Mutlu, "Memory performance attacks: Denial of memory service in multi-core systems," USENIX Security 2007.

A Question or Two

- Can you figure out why there is a disparity in slowdowns if you do not know how the processor executes the programs?

- Can you fix the problem without knowing what is happening "underneath"?

Why the Disparity in Slowdowns?



unfairness

Multi-Core Chip

Shared DRAM Memory System

DRAM Bank Operation

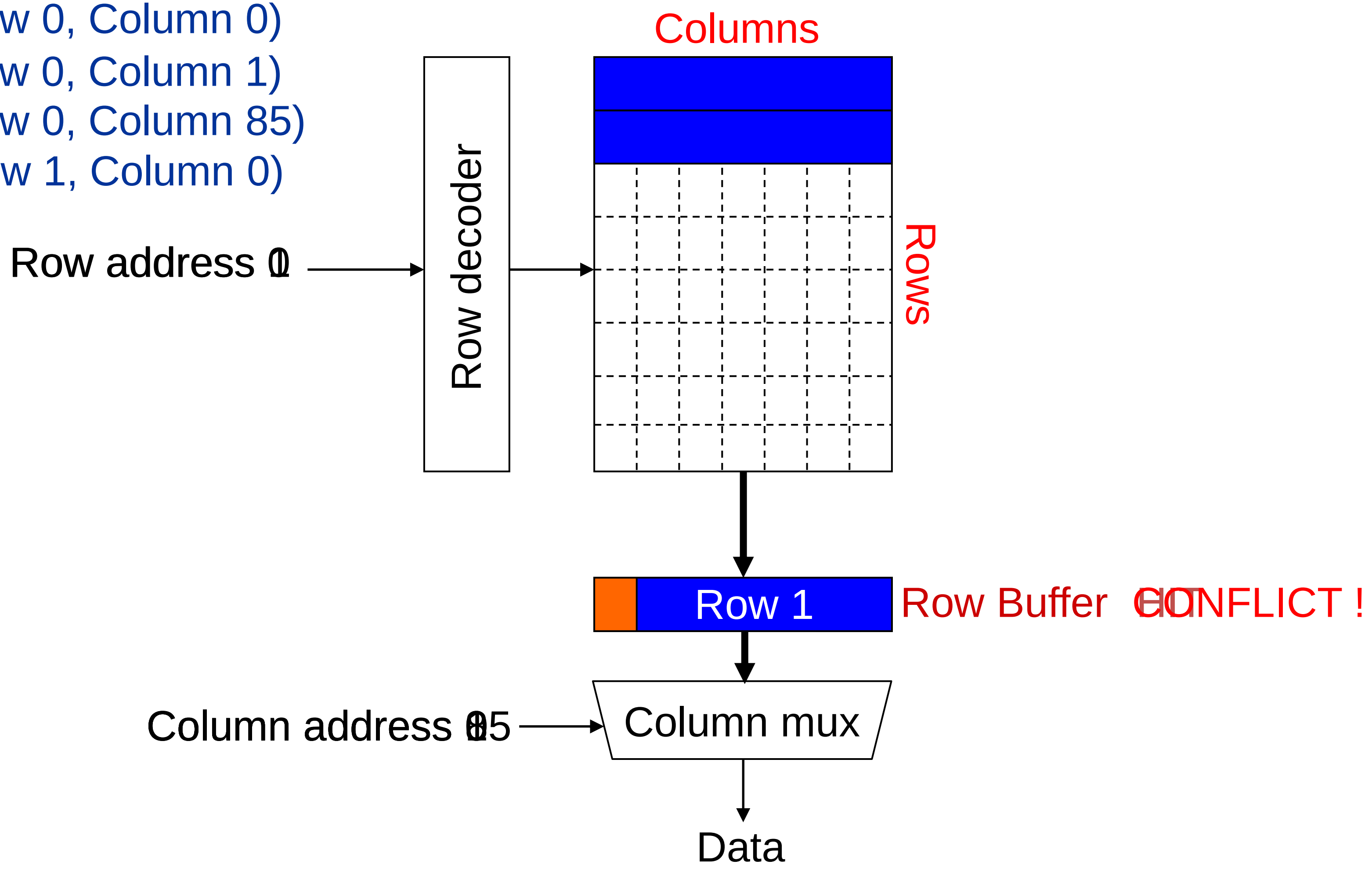
Access Address:

(Row 0, Column 0)

(Row 0, Column 1)

(Row 0, Column 85)

(Row 1, Column 0)



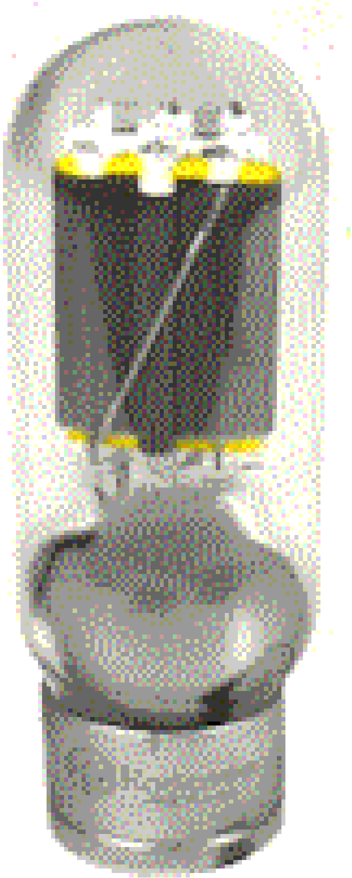
INTRODUCTION

Generations of Computer

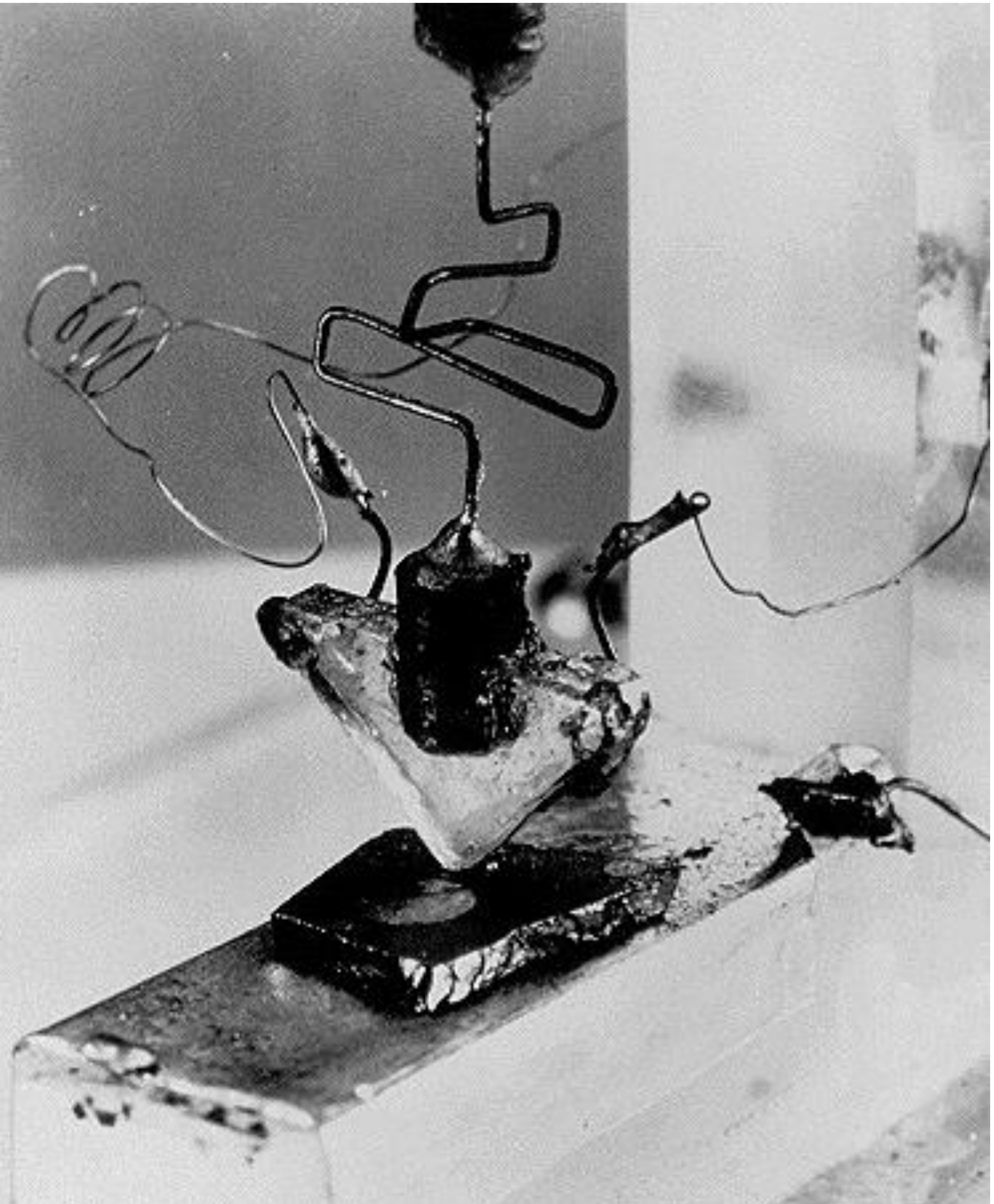
1. First Generation - 1940-1956: Vacuum Tubes
2. Second Generation - 1956-1963: Transistors
3. Third Generation - 1964-1971: Integrated Circuits
4. Fourth Generation - 1971-Present: Microprocessors
5. Fifth Generation - Present and Beyond: Artificial Intelligence

FIRST GENERATION 1940 - 1956

- First generation computers used Vacuum Tubes
- Vacuum tubes are glass tubes with circuits inside.
- The word vacuum indicates that they have no air inside, which protects the circuitry.
- Building a computer with these vacuum tubes would result in a very large machine occupying one full room.



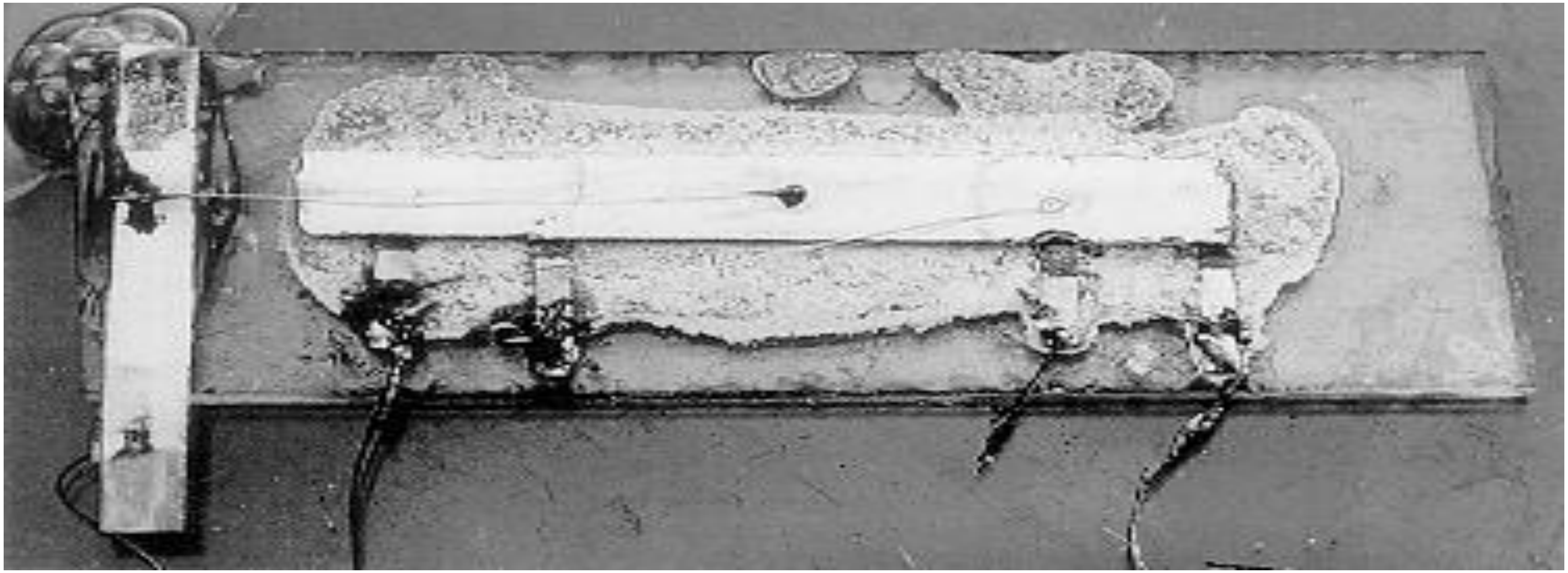
First Transistor



- Uses Silicon
- developed in 1948
- won a Nobel prize
- on-off switch

- Second Generation Computers used Transistors, starting in 1956

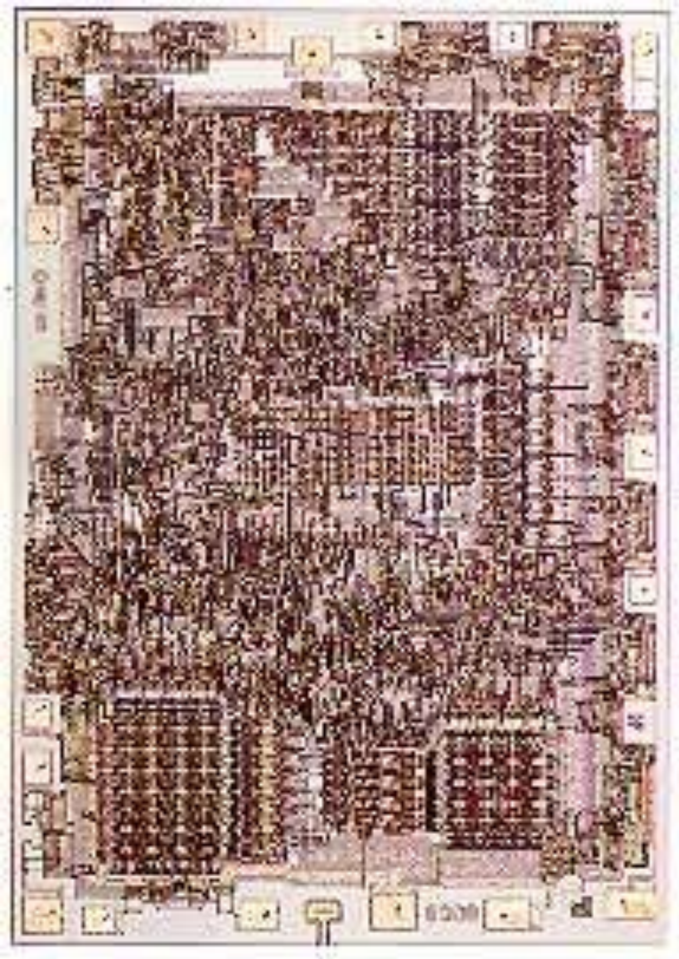
Integrated Circuits



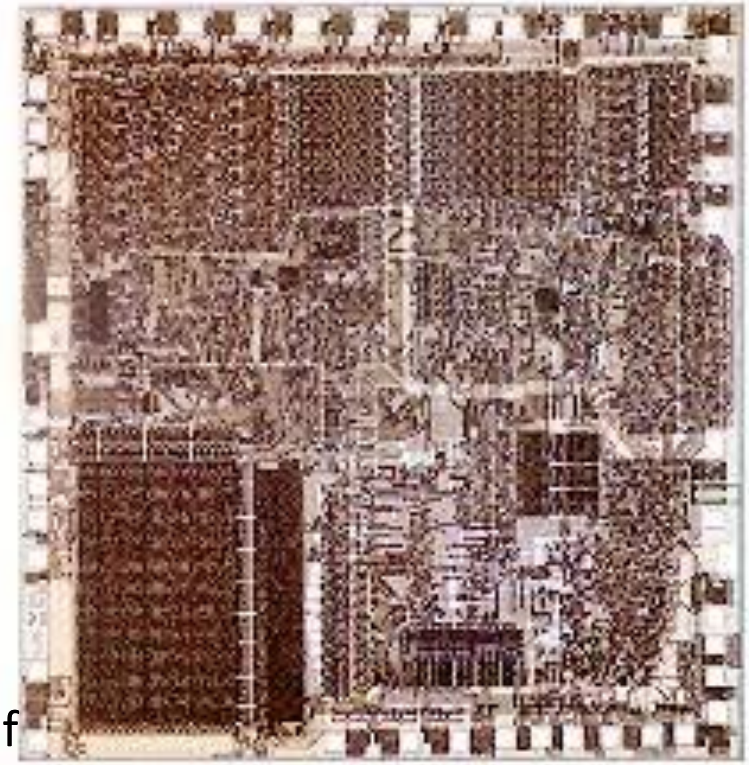
- Third Generation Computers used Integrated Circuits (chips).
- Integrated Circuits are transistors, resistors, and capacitors integrated together into a single “chip”
- Instead of punched cards and printouts, users started interacting with keyboards and mouse.

1972: 8008 Microprocessor

- The 8008 was twice as powerful as the 4004.
- According to the magazine *Radio Electronics*, Don Lancaster, a dedicated computer hobbyist, used the 8008 to create a predecessor to the first personal computer, a device *Radio Electronics* dubbed a "TV typewriter." It was used as a dumb terminal.

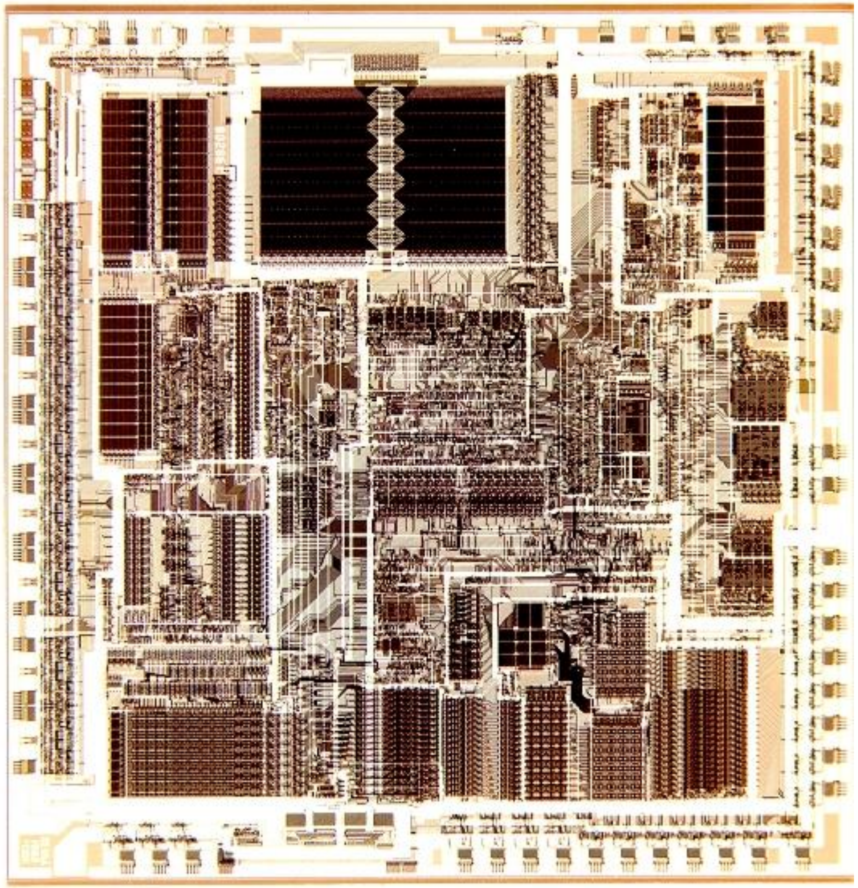


1978: 8086-8088 Microprocessor



- In 1978 intel
  - 16-bit
  - 2.5MIPS **millions of instructions per second**
  - **1M-byte memory**
  - 4- or 6-byte instruction (cache) queue that prefetch instructions
  - over 20,000 variations instructions.
  - 64K bytes of memory found in 8-bit microprocessors to execute efficiently
  - The 16-bit 8086 and 8088 provided 1M byte of memory for these applications
  - Popularity of Intel ensured in 1981 when IBM chose the 8088 in its personal computer

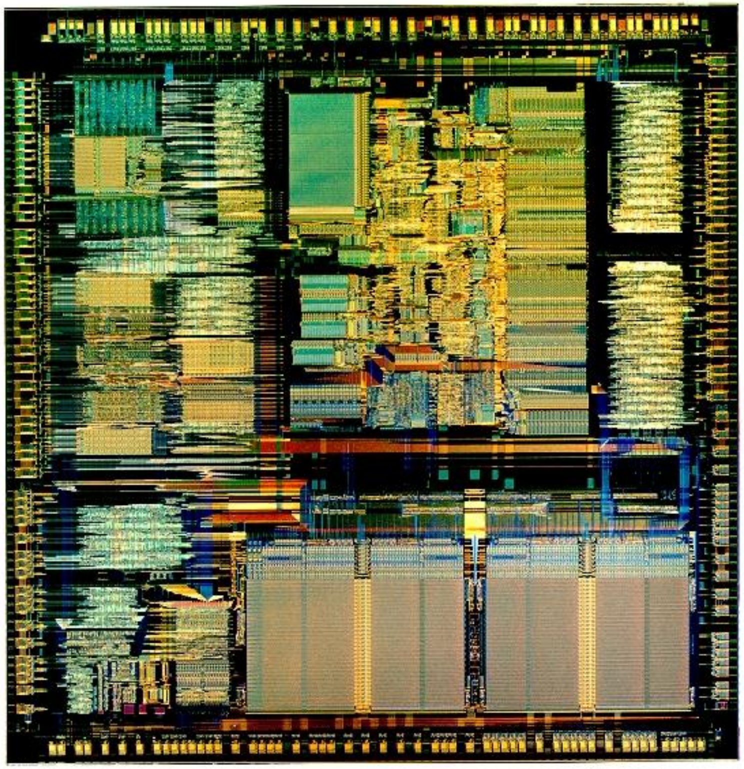
1982: 286 Microprocessor



- The 286, also known as the 80286, was the first Intel processor that could run all the software written for its predecessor.
- This software compatibility remains a hallmark of Intel's family of microprocessors.
- Within 6 years of it release, there were an estimated 15 million 286-based personal computers installed around the world.

- 80286: updated 8086
  - 16M byte memory addressing
  - Instructions identical to 8086 few more added
  - 4MIPS
  - 8Mhz clock speed

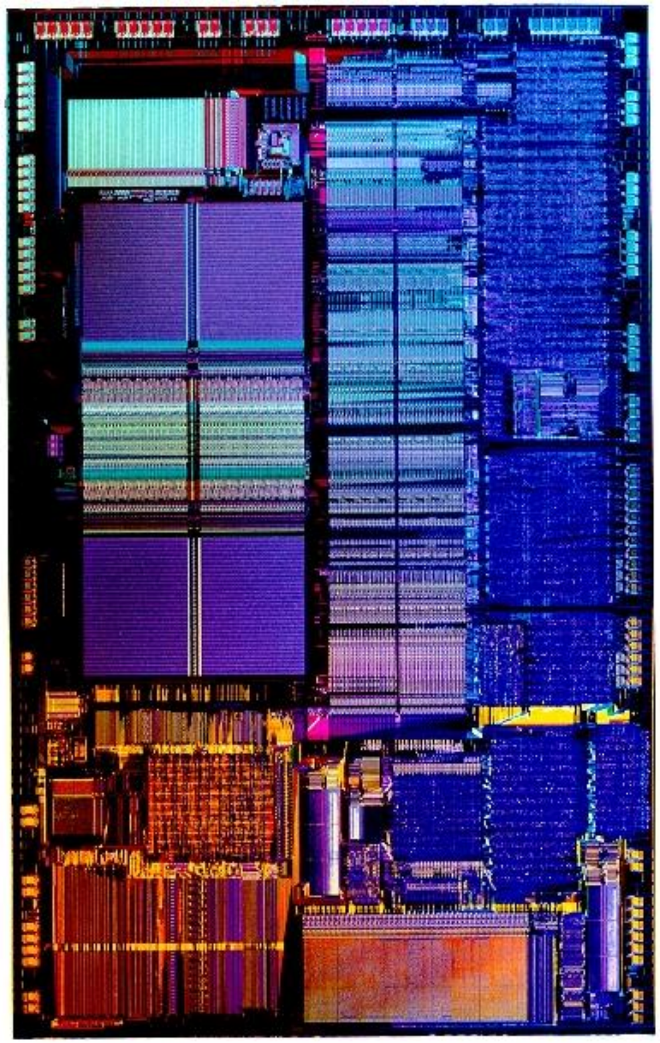
1985: Intel 386(TM) Microprocessor



- The Intel 386^{TM} microprocessor featured 275,000 transistors--more than 100times as many as the original 4004.
- It was a 32-bit chip and was "multi tasking," meaning it could run multiple programs at the same time.
- 80386 (1985)
  - 32-bit address bus and 32-bit data bus
  - 4GB memory
  - Hardware circuitry for memory management
  - Additional instructions referenced 32-bit registers and managed the memory system
-

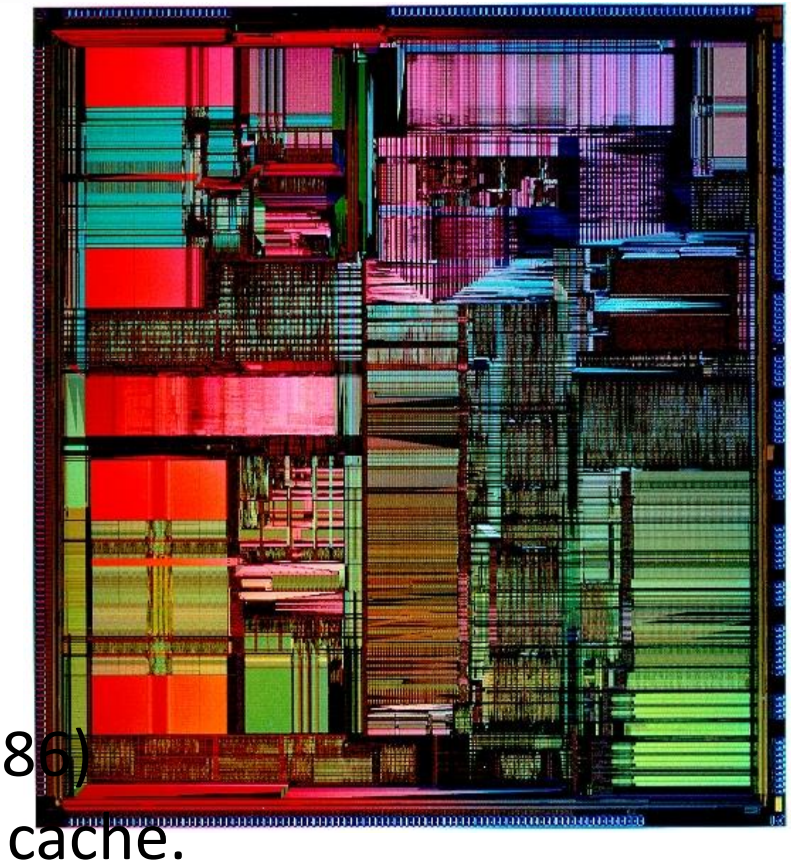
1989: Intel 486(TM) DX CPU

- The 486TM generation really allowed the shift from command-level computing into point-and-click computing.
- The Intel 486TM processor was the first to offer a built-in math coprocessor, which speeds up computing because it offers complex math functions from the central processor.



- 80486
  - Highly integrated package.
  - 80386-like microprocessor.80387-like numeric coprocessor.
  - 50 MIPS
  - 8K-byte cache memory system
  - Half of its instructions executed in 1 clock cycle rather 2

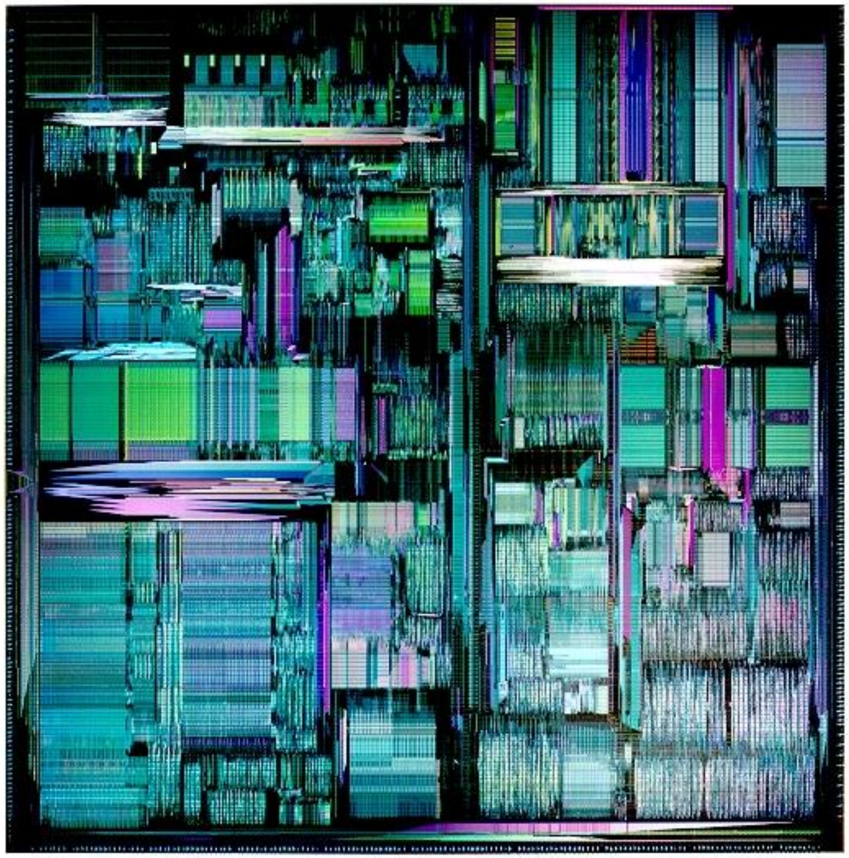
1993: Pentium® Processor



- The Pentium® processor allowed computers to more easily incorporate "real world" data such as speech, sound, handwriting and photographic images.
- The name Pentium®, mentioned in the comics and on television talk shows, became a household word soon after introduction.
  - Originally named P5 or 80586
  - Clock speed of 60MHZ
  - Executes 110MIPS
- Cache size: 16K bytes (8K cache in 80486)
  - 8K-byte instruction cache and data cache.
- Memory system up to 4G bytes.
- Data bus width increased to a full 64 bits.
- Data bus transfer speed 60 MHz or 66 MHz.
  - depending on the version of the Pentium

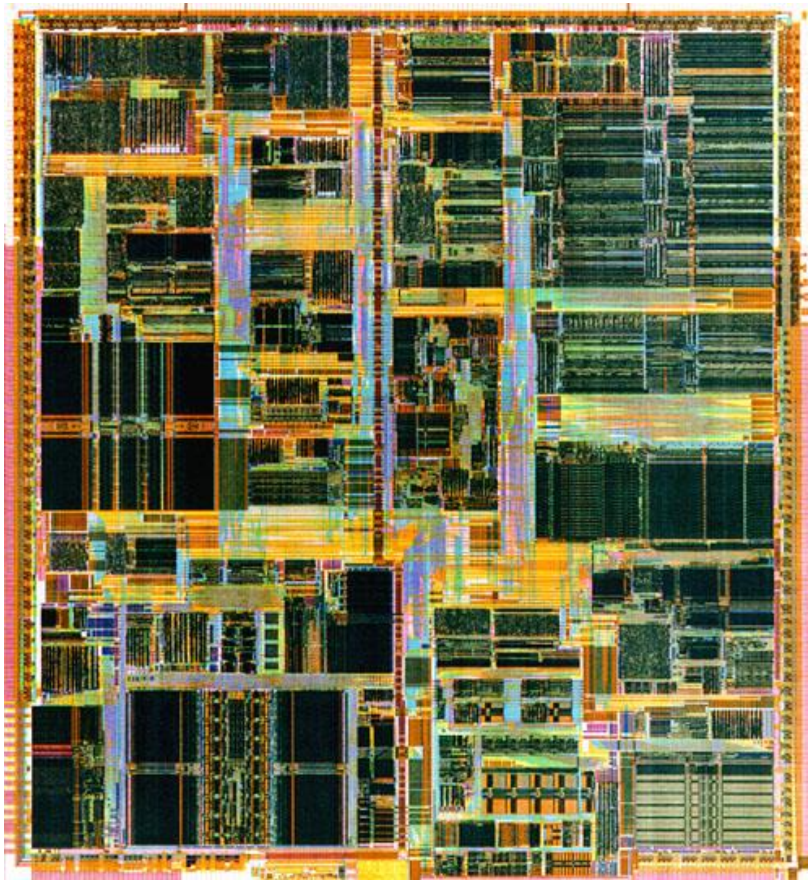
1995: Pentium® Pro Processor

- Released in the fall of 1995 the Pentium® Pro processor is designed to fuel 32-bit server and workstation-level applications, enabling fast computer-aided design, mechanical engineering and scientific computation.
- Each Pentium® Pro processor is packaged together with a second speed-enhancing cache memory chip.
- The powerful Pentium® Pro processor boasts 5.5 million transistors.



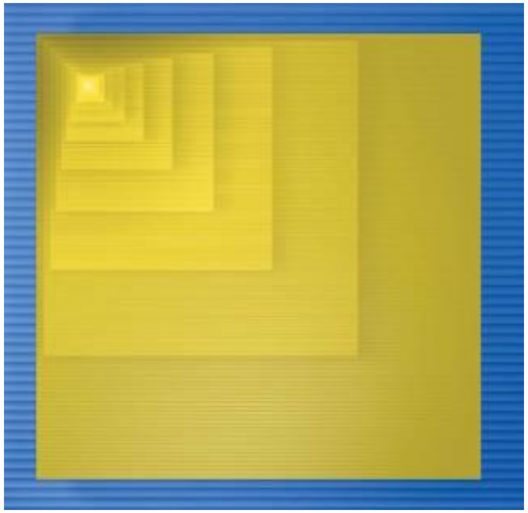
1997: Pentium® II Processor

- The 7.5 million-transistor Pentium® II processor incorporates Intel MMX™ technology, which is designed specifically to process video, audio and graphics data efficiently.
- It is packaged along with a high-speed cache memory chip in an innovative Single Edge Contact (S.E.C.) cartridge that connects to a motherboard via a single edge connector, as opposed to multiple pins.
- With this chip, PC users can capture, edit and share digital photos with friends and family via the Internet; edit and add text, music or between-scene transitions to home movies; and, with a video phone, send video over standard phone lines and the Internet.

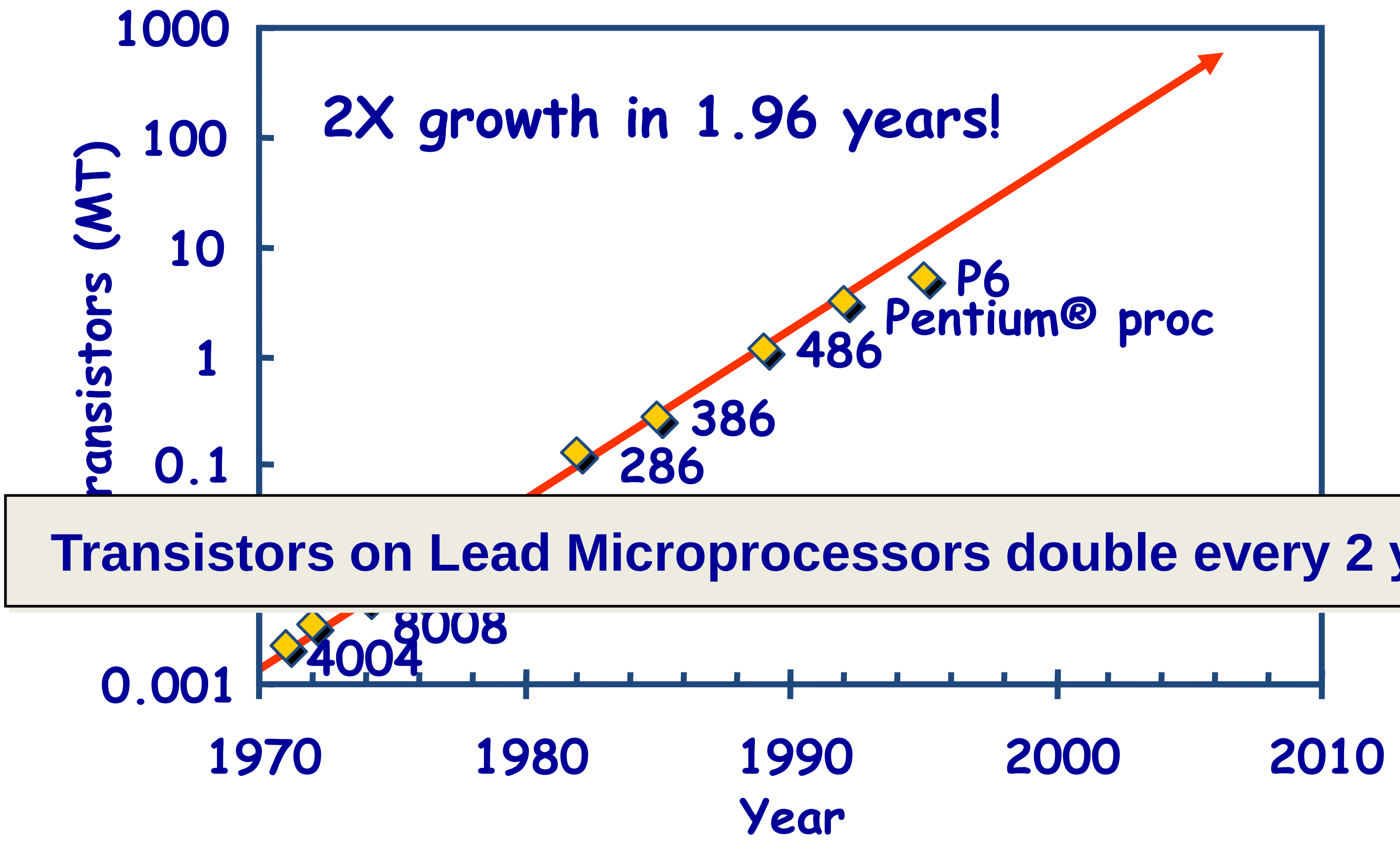


Moore’s Law

- **IC capacity doubling about every 18 months for several decades**
  - **Known as “Moore’s Law” after Gordon Moore, co-founder of Intel**
    - **Predicted in 1965 predicted that components per IC would double roughly every year or so**
  - **Picture depicts related phenomena**
    - **For a particular number of transistors, the IC shrinks by half every 18 months**
      - **Notice how much shrinking occurs in just about 10 years**
      - **Enables incredibly powerful computation in incredibly tiny devices**
  - **Today’s ICs hold *billions* of transistors**
    - **The first Pentium processor (early 1990s) needed only 3 million**



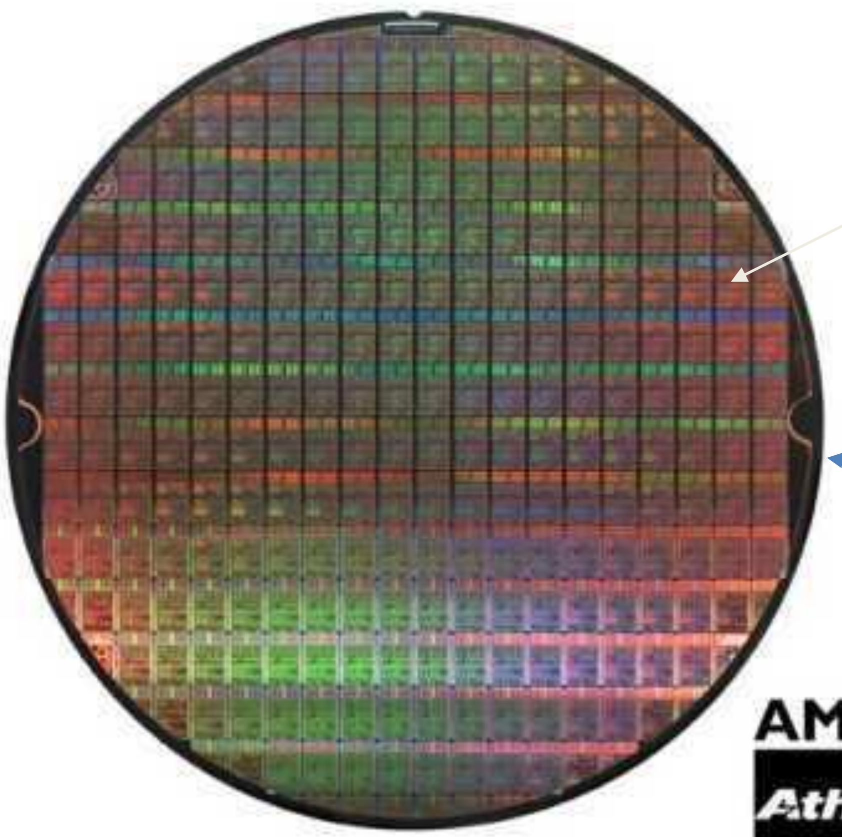
Moore's law in Microprocessors



Transistors on Lead Microprocessors double every 2 years

Courtesy, Intel

Die

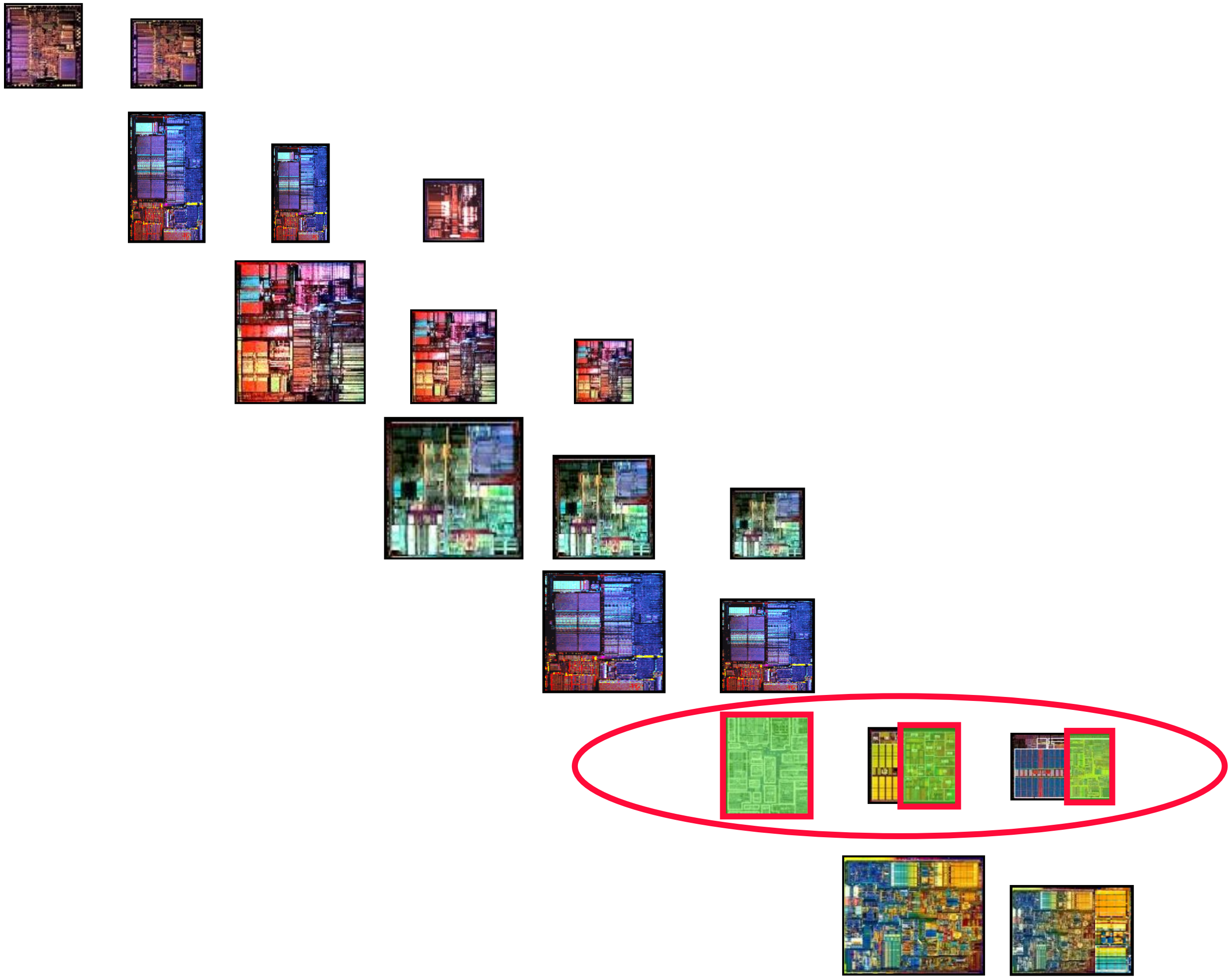


Going up to 12" (30cm)

From http://www.amd.com

In Picture…

Silicon Process Technology 1.5μ 1.0μ 0.8μ 0.6μ 0.35μ 0.25μ 0.18μ 0.13μ



Intel386™ DX Processor

Intel486™ DX Processor

Pentium® Processor

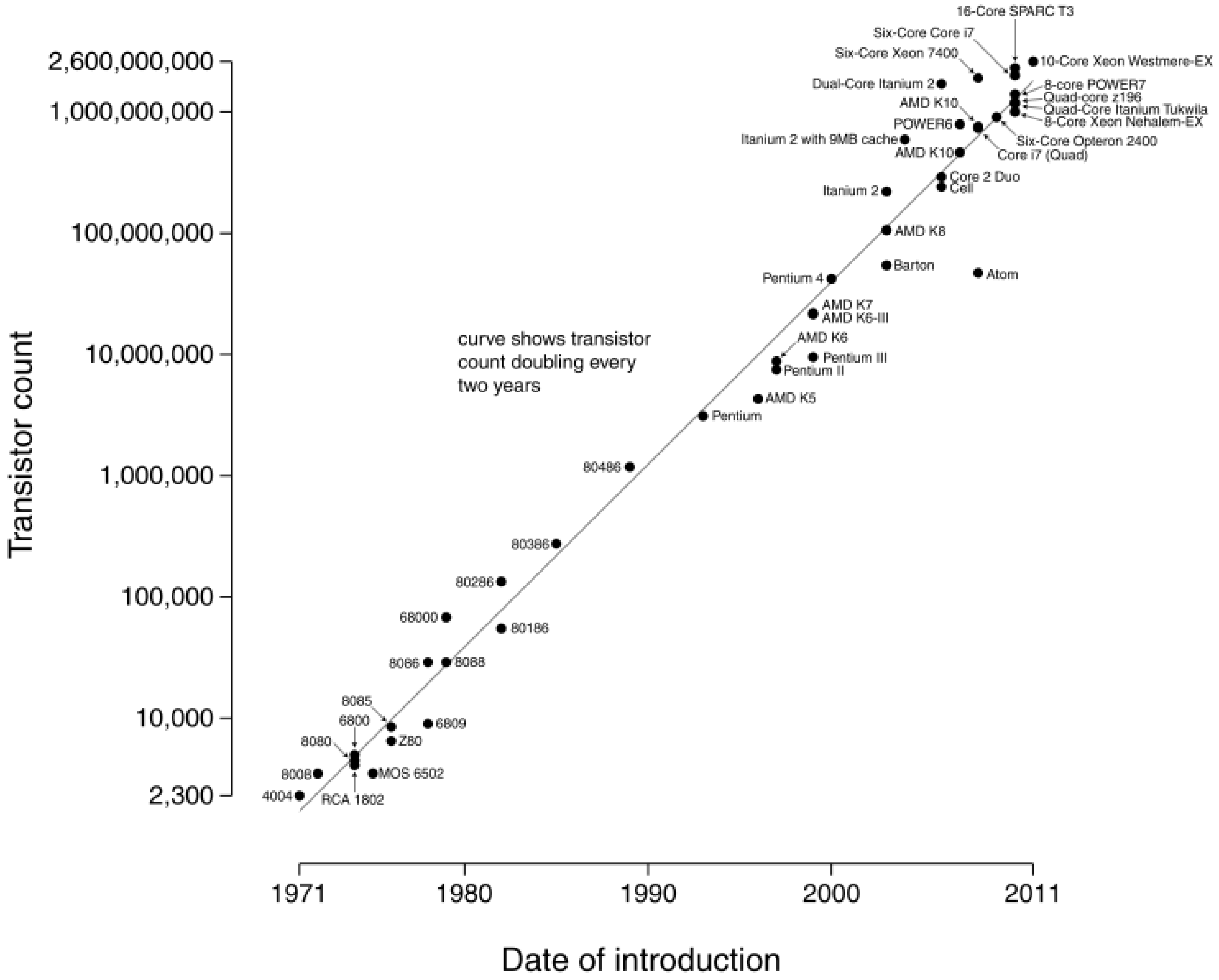
Pentium® Pro Processor

Pentium® II Processor

Pentium® III Processor

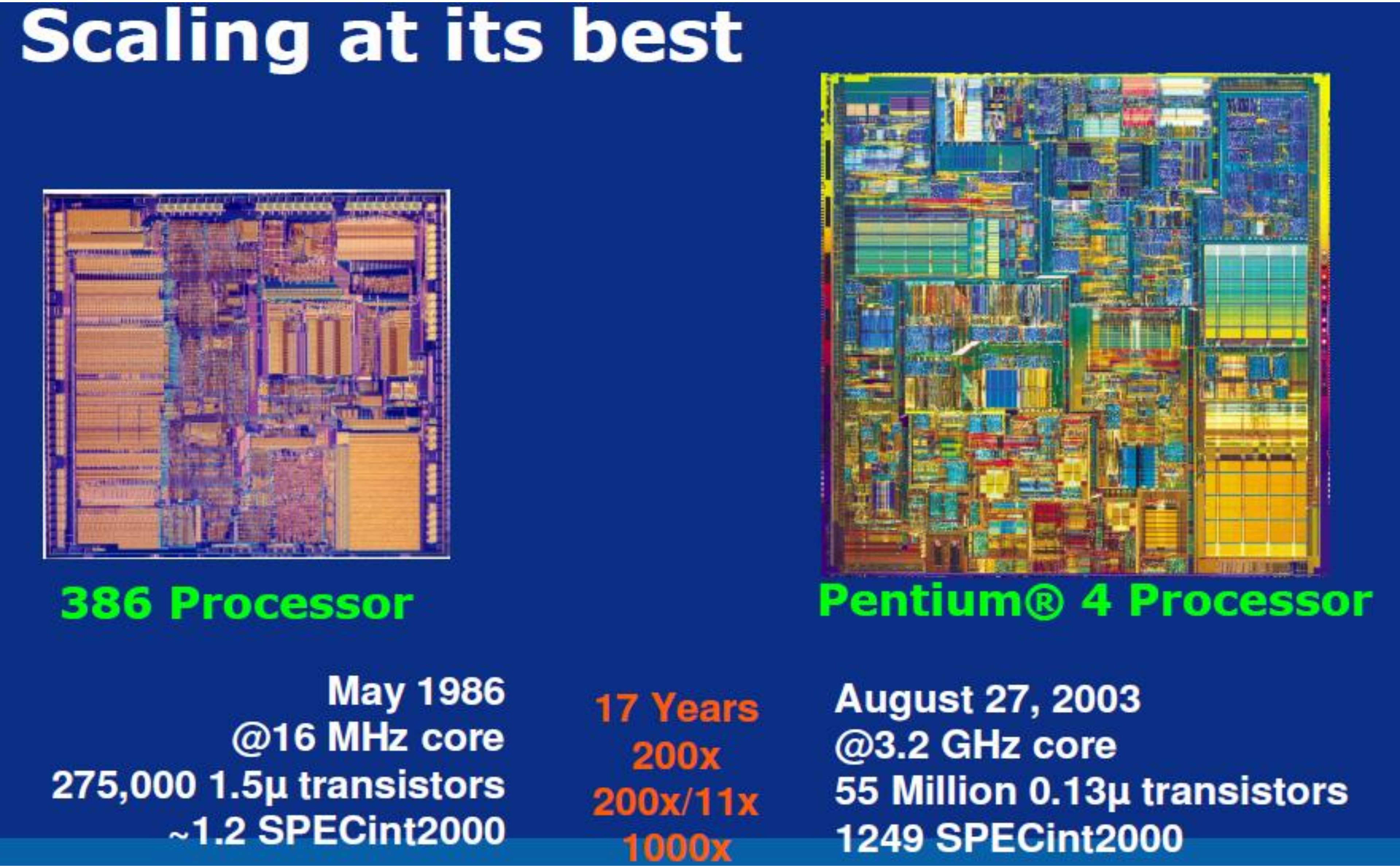
Pentium® 4 Processor

Microprocessor Transistor Counts 1971-2011 & Moore's Law



Number of transistors on an integrated circuit doubles ~ every two years

Image source: Wikipedia



Scaling at its best

386 Processor

Pentium® 4 Processor

May 1986
@16 MHz core
275,000 1.5µ transistors
~1.2 SPECint2000

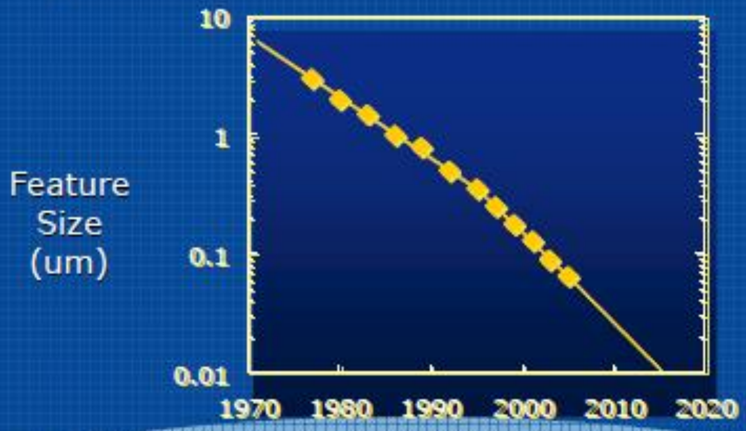
17 Years
200x
200x/11x
1000x

August 27, 2003
@3.2 GHz core
55 Million 0.13µ transistors
1249 SPECint2000

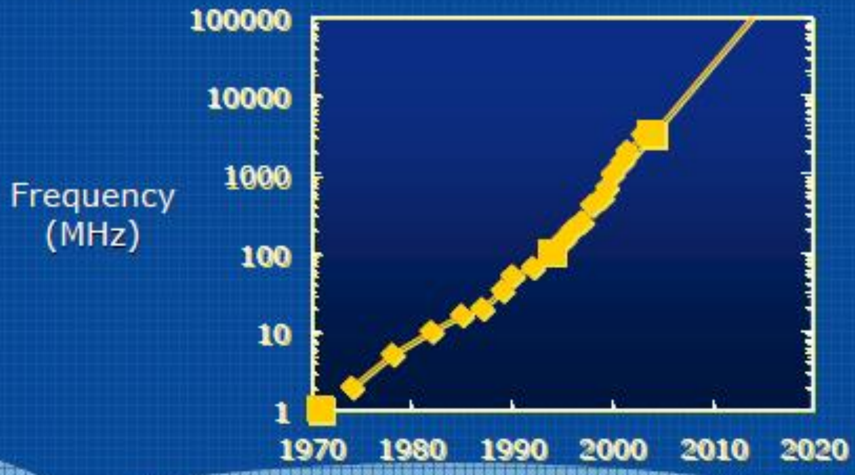
Courtesy, Intel

Historical Driving Forces

Shrinking Geometry

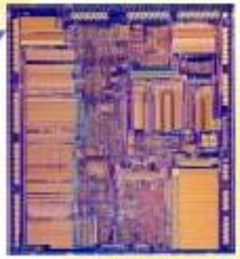


Increased Frequency

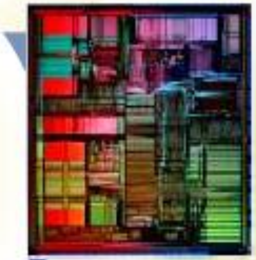


1971
4004 Processor
2300 Transistors

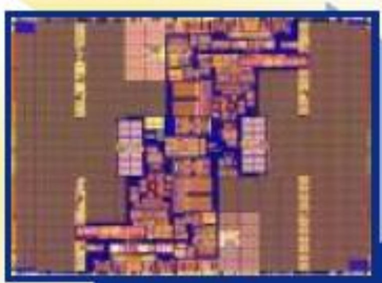
1978
8008 Processor
IBM PC



1986
I386 Processor
32-bit



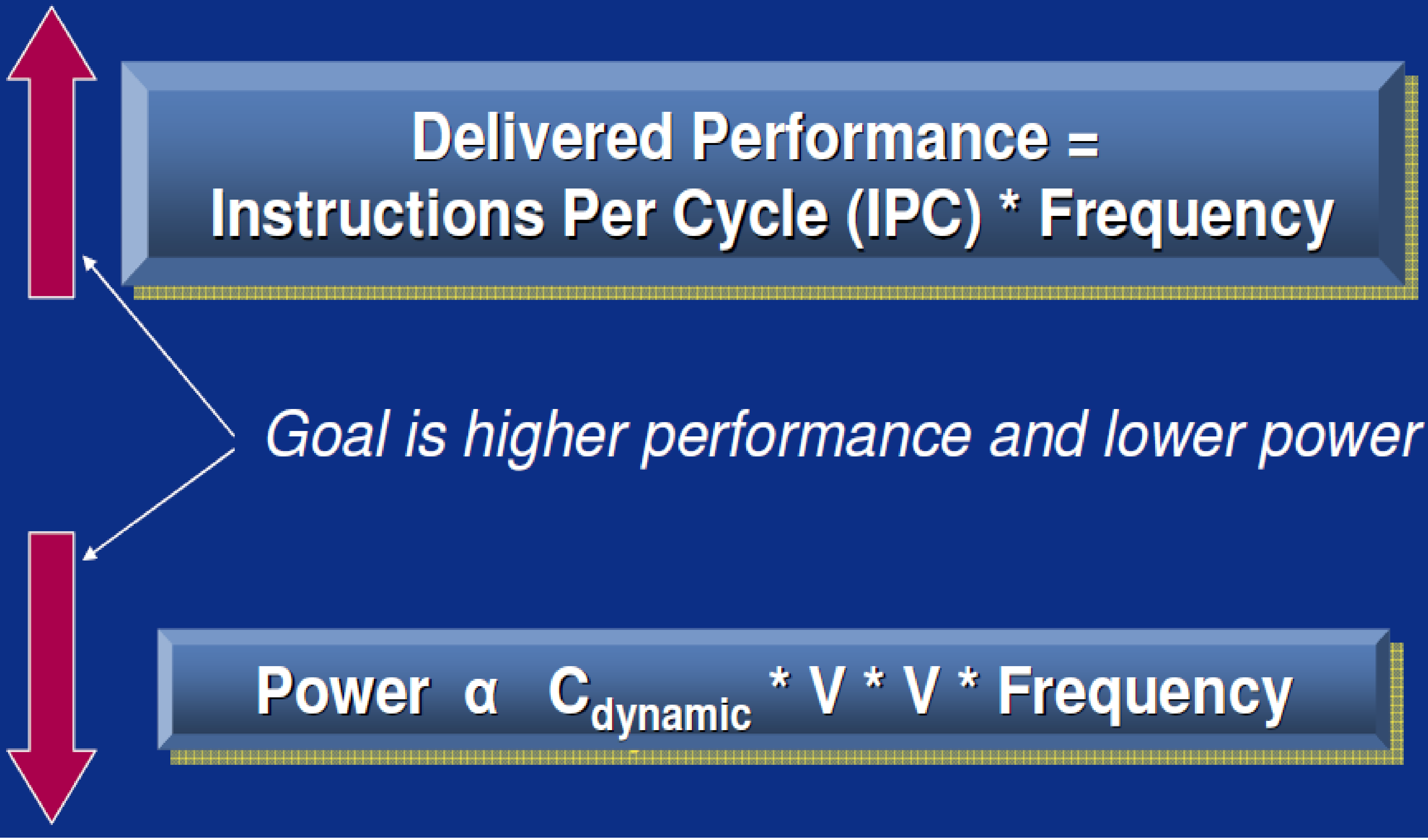
1993
Pentium Processor
3.1M transistors



2005
Montecito
1.7B Transistors

Courtesy, Intel

The Act of Balancing



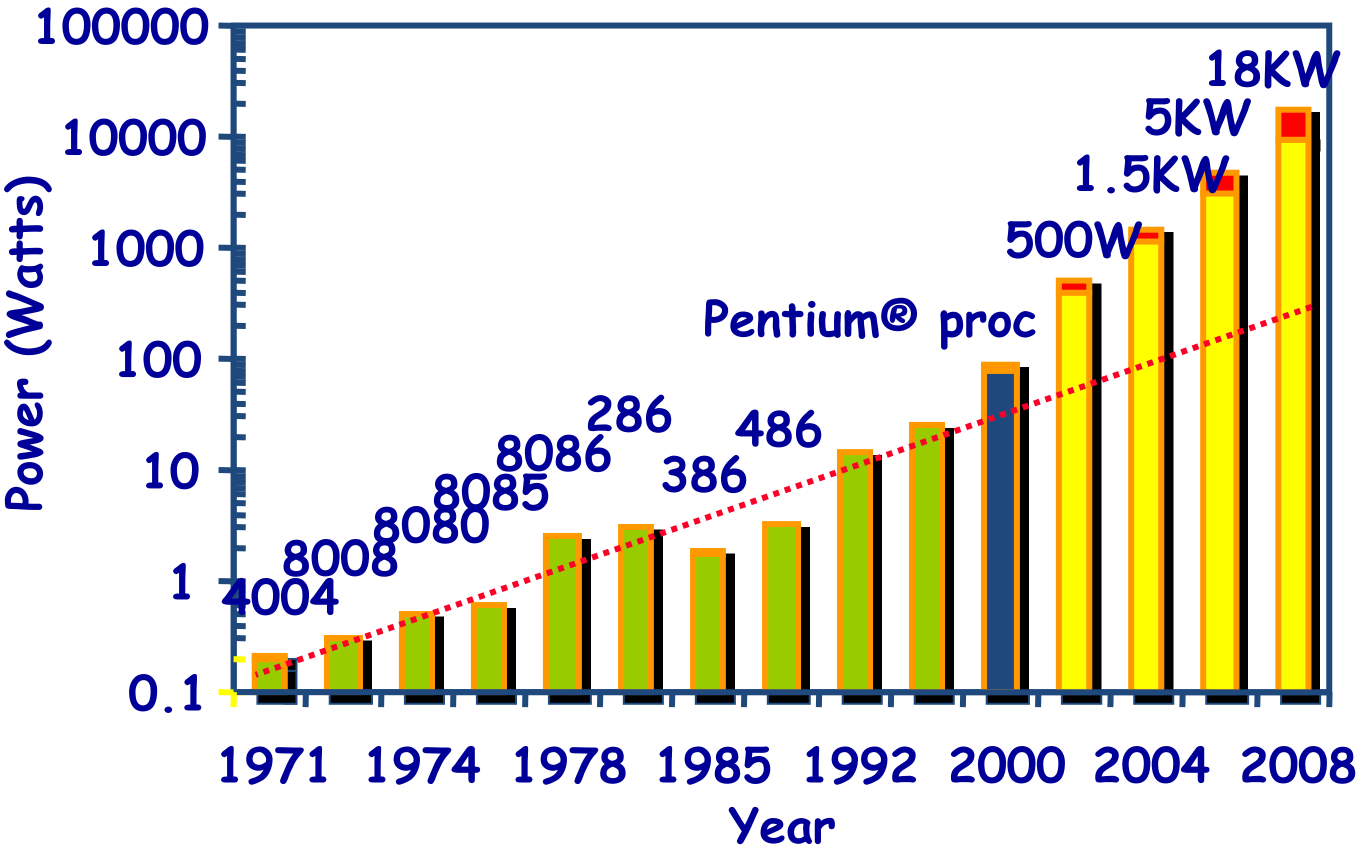
Delivered Performance =
Instructions Per Cycle (IPC) * Frequency

Goal is higher performance and lower power

Power α $C_{dynamic}$ * V * V * Frequency

Courtesy, Intel

Power will be a major problem



Power delivery and dissipation will be prohibitive

Courtesy, Intel

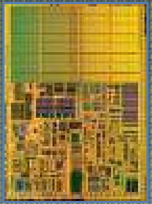
Dual core with voltage scaling

RULE OF THUMB

A 15% Reduction In Voltage Yields

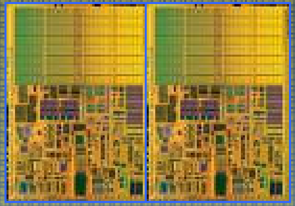
Frequency Reduction	Power Reduction	Performance Reduction
15%	45%	10%

SINGLE CORE



Area = 1
Voltage = 1
Freq = 1
Power = 1
Perf = 1

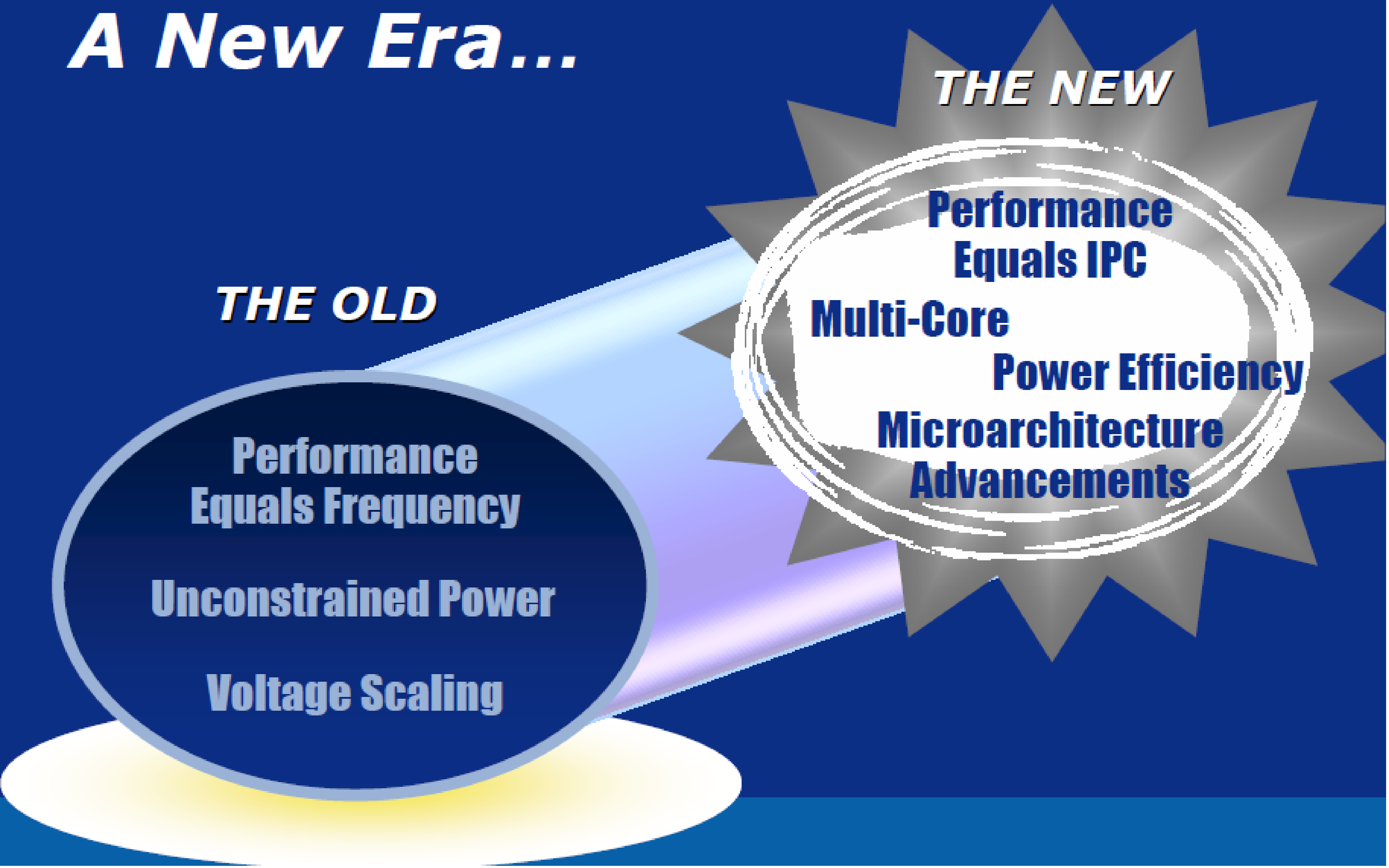
DUAL CORE



Area = 2
Voltage = 0.85
Freq = 0.85
Power = 1
Perf = ~1.8

Courtesy, Intel

A New Era…



THE OLD

Performance Equals Frequency

Unconstrained Power

Voltage Scaling

THE NEW

Performance Equals IPC

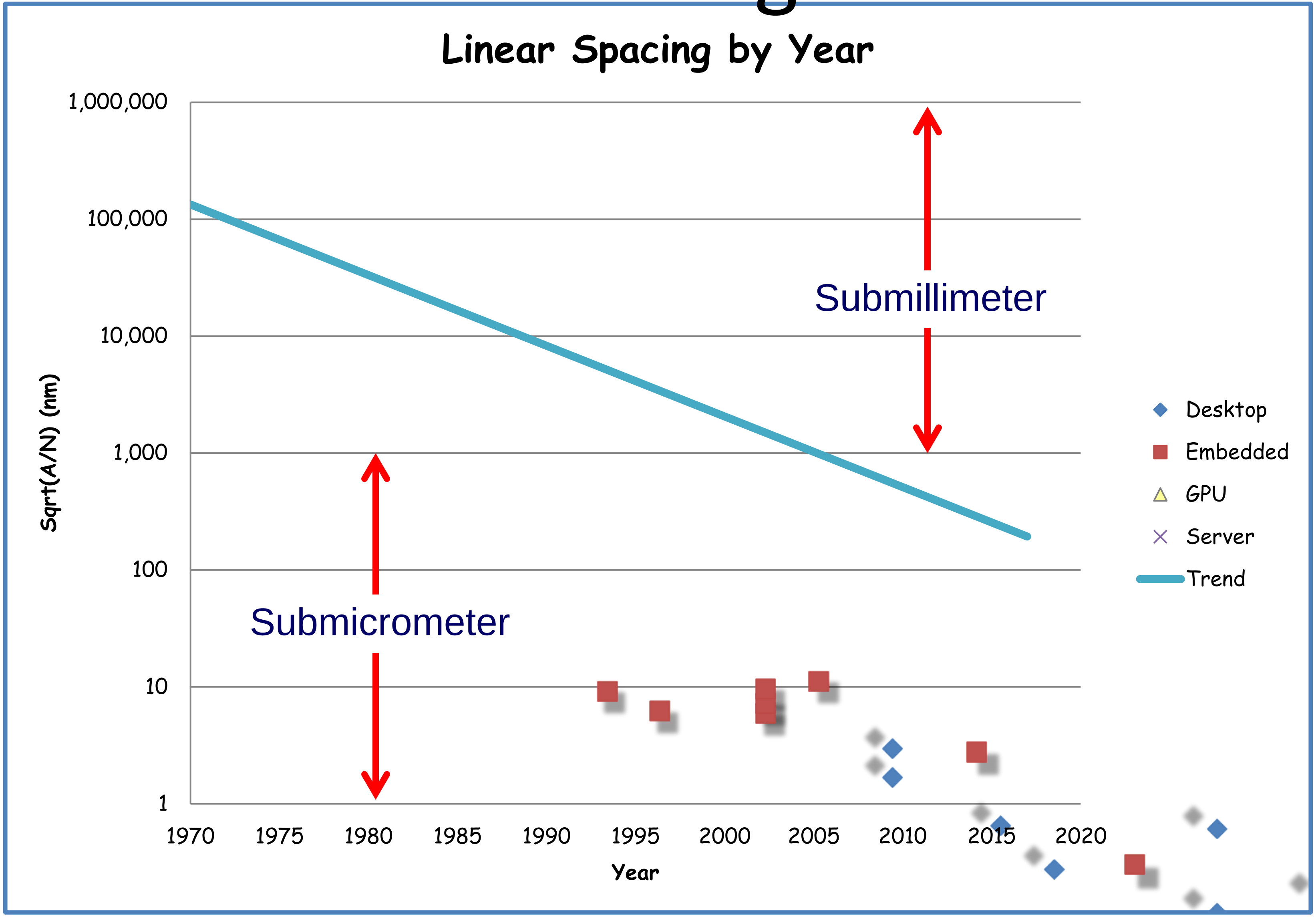
Multi-Core

Power Efficiency

Microarchitecture Advancements

Courtesy, Intel

Linear Scaling Trend

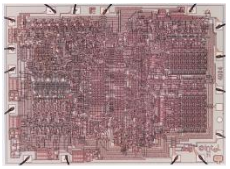


Submillimeter Dimensions

10^{-3} 1 millimeter (mm)

500µm: Length of amoeba

10^{-4}



72µm: Intel 4004 linear scale

50µm: Average size of cell in human body

10^{-5}

10µm: Thickness of sheet of plastic food wrap

5µm: Spider silk thickness

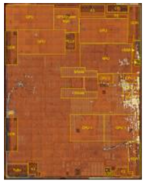
2µm: E coli bacterium length

10^{-6} 1 micrometer (µm)

Submicrometer Dimensions

10^{-6} 1 micrometer (µm)

400-700nm: Visible light wavelengths



144nm: Apple A11 linear scale

10^{-7}

30nm: Minimum cooking oil smoke particle diameter

10^{-8}

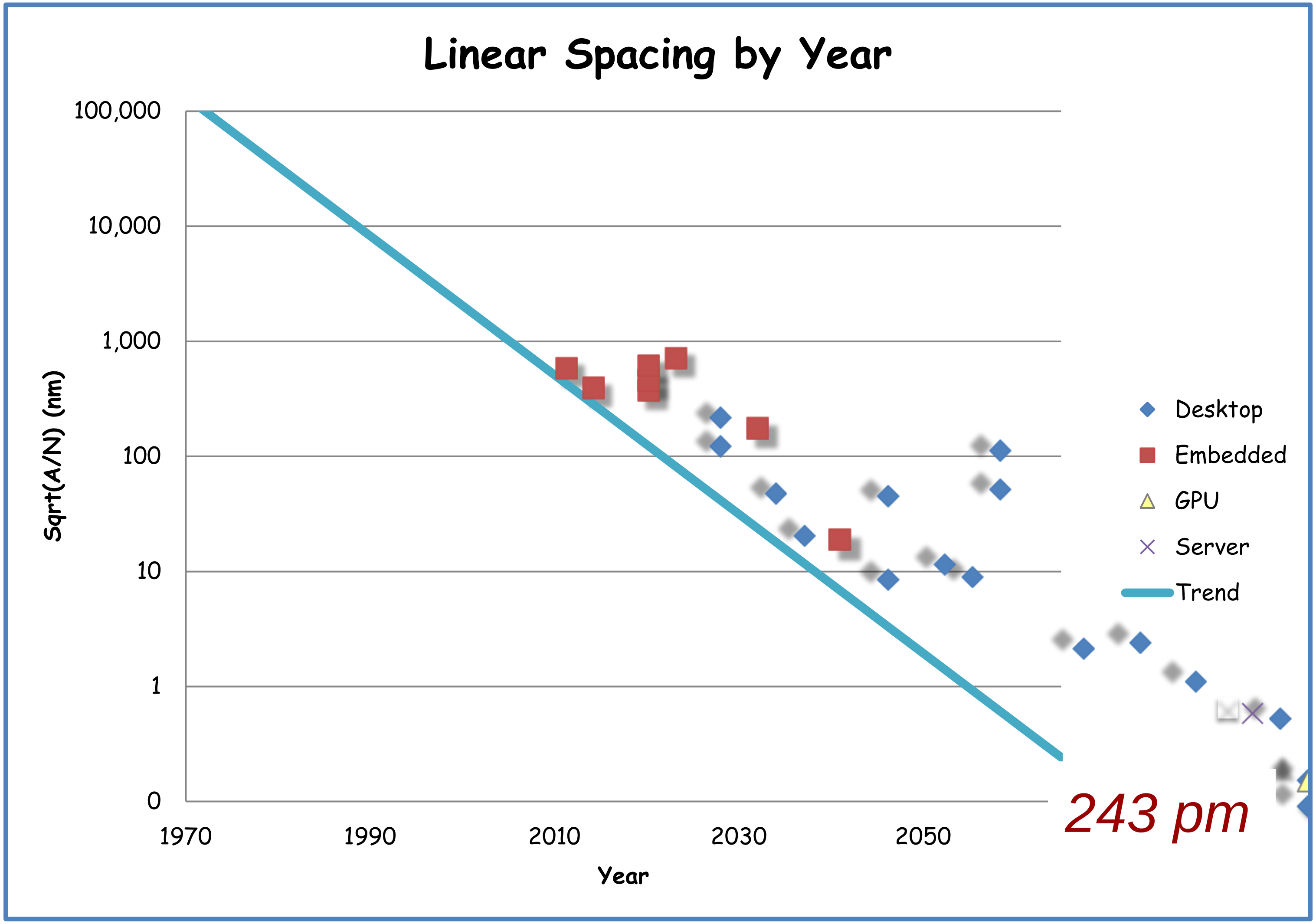
9nm: Cell membrane thickness

2nm: DNA helix diameter

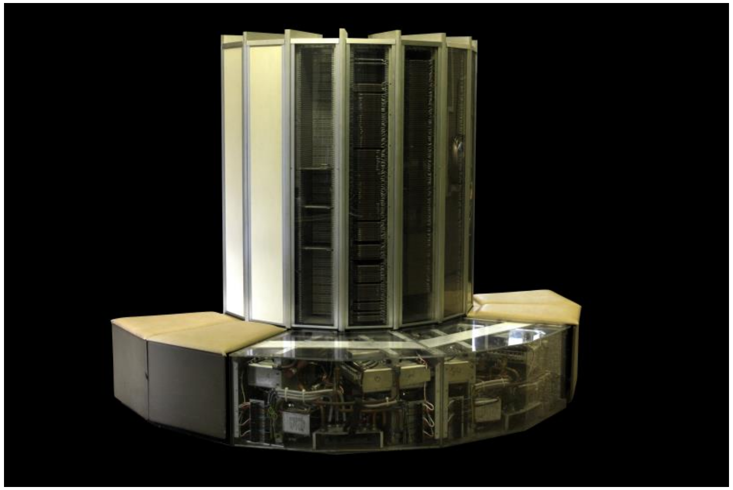
10^{-9} 1 nanometer (nm)

1nm: Carbon nanotube diameter

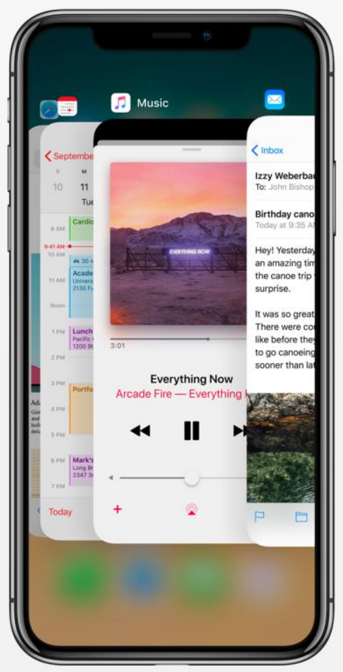
Linear Scaling Extrapolation



What Moore’s Law Has Meant



- 1976 Cray 1
  - 250 M Ops/second
  - ~170,000 chips
  - 0.5B transistors
  - 5,000 kg, 115 KW
  - $9M
  - 80 manufactured

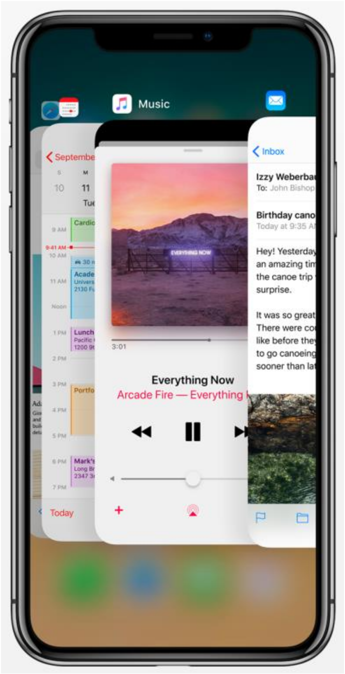


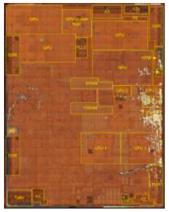
- 2017 iPhone X
  - > 10 B Ops/second
  - 16 chips
  - 4.3B transistors (CPU only)
  - 174 g, < 5 W
  - $999
  - ~3 million sold in first 3 days

What Moore's Law Has Meant

- 1965 Consumer Product

- 2017 Consumer Product



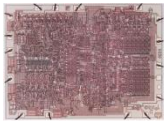


Apple A11 Processor
4.3B transistors

Visualizing Moore’s Law to Date

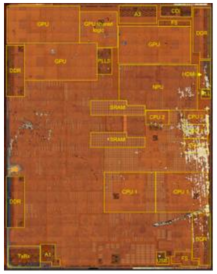
If transistors were the size of a grain of sand

Intel 4004
1970
2,300 transistors



0.1 g

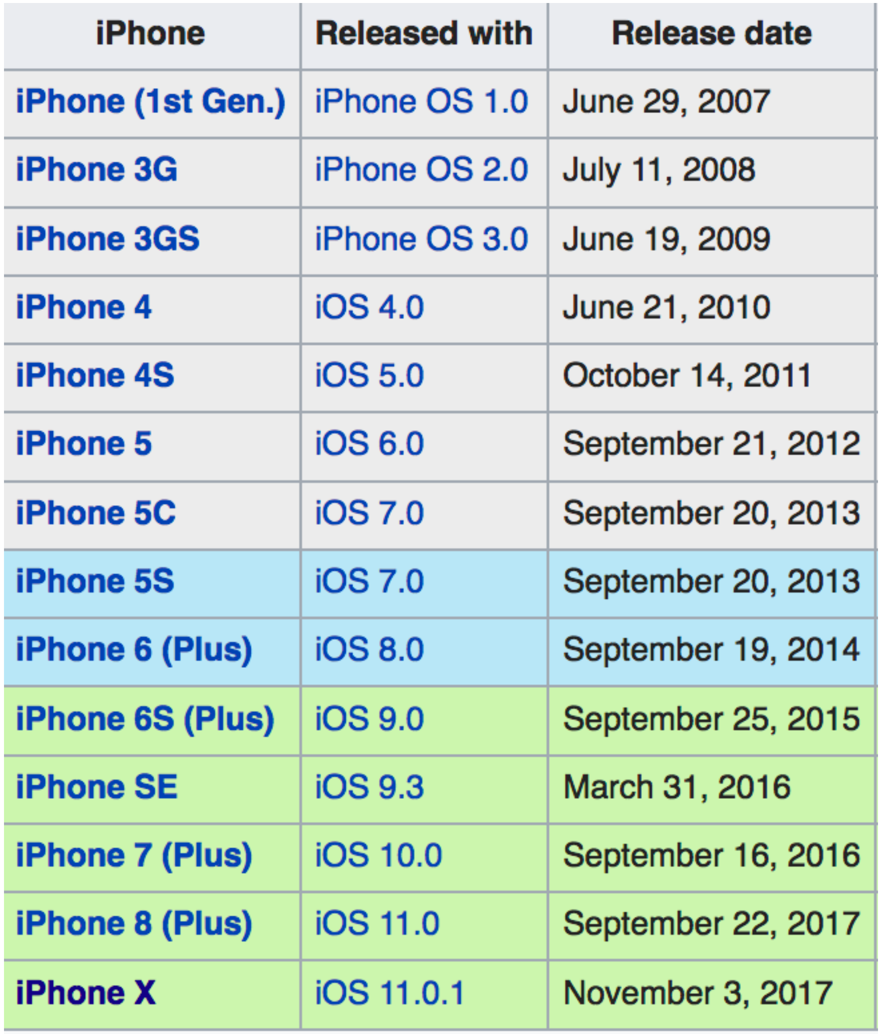
Apple A11
2017
4.3 B transistors



QUIKRETE
PREMIUM
PLAY SAND
Ideal For Building & Molding
Screened • Washed • Dried
NET WT. 50 LB APPROX 1/2 CU. FT. (22.7kg)

189 kg

What Moore’s Law Has Meant

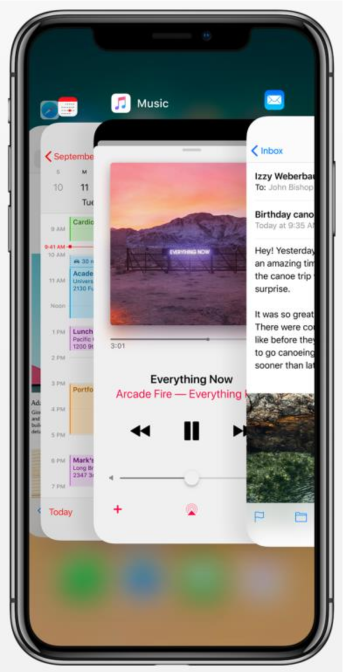


iPhone	Released with	Release date
iPhone (1st Gen.)	iPhone OS 1.0	June 29, 2007
iPhone 3G	iPhone OS 2.0	July 11, 2008
iPhone 3GS	iPhone OS 3.0	June 19, 2009
iPhone 4	iOS 4.0	June 21, 2010
iPhone 4S	iOS 5.0	October 14, 2011
iPhone 5	iOS 6.0	September 21, 2012
iPhone 5C	iOS 7.0	September 20, 2013
iPhone 5S	iOS 7.0	September 20, 2013
iPhone 6 (Plus)	iOS 8.0	September 19, 2014
iPhone 6S (Plus)	iOS 9.0	September 25, 2015
iPhone SE	iOS 9.3	March 31, 2016
iPhone 7 (Plus)	iOS 10.0	September 16, 2016
iPhone 8 (Plus)	iOS 11.0	September 22, 2017
iPhone X	iOS 11.0.1	November 3, 2017

12 generations of iPhone since 2007

What Moore's Law Could Mean

- 2017 Consumer Product



- 2065 Consumer Product

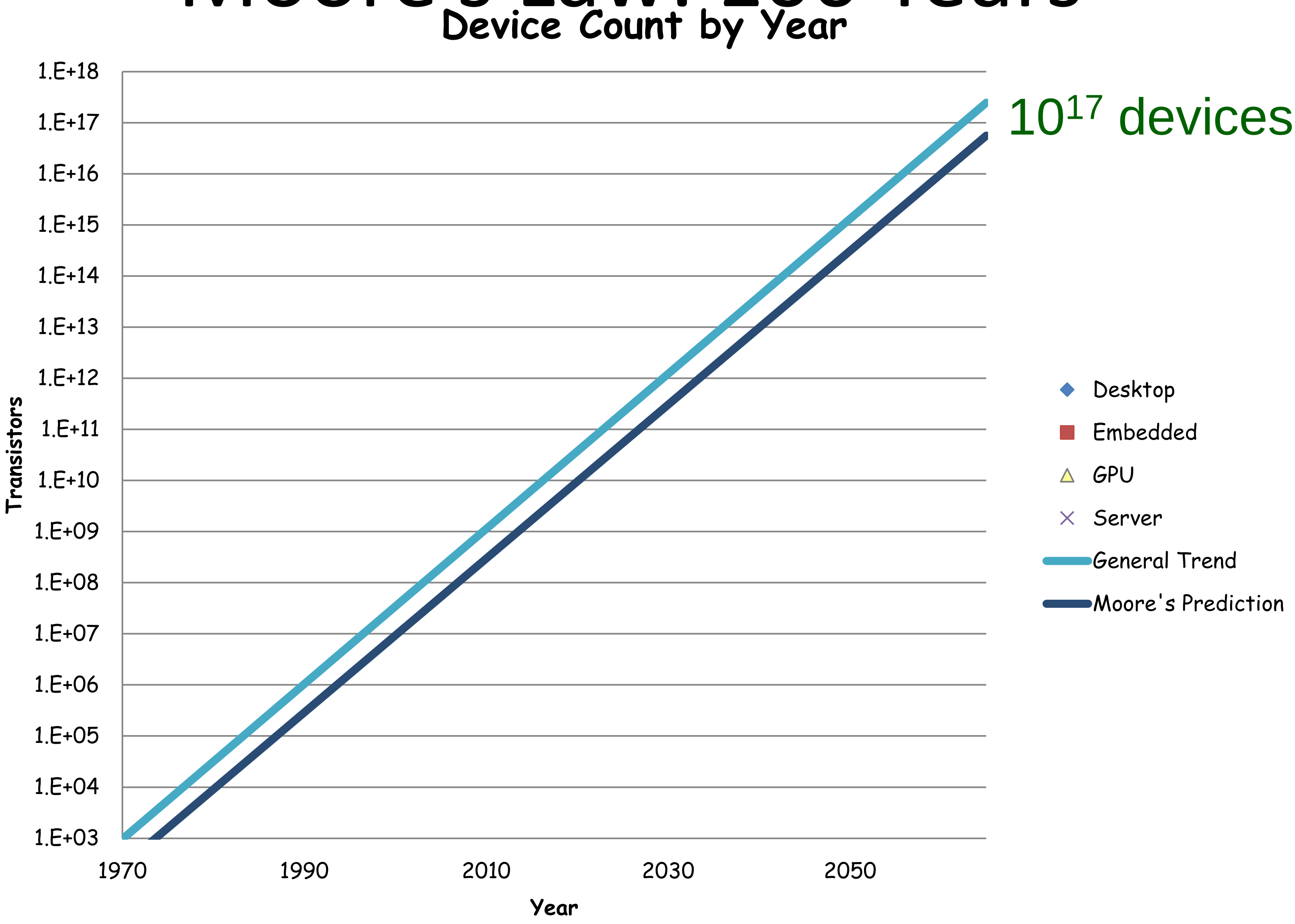
?

  – Portable
  – Low power
  – Will drive markets & innovation

Requirements for Future Technology

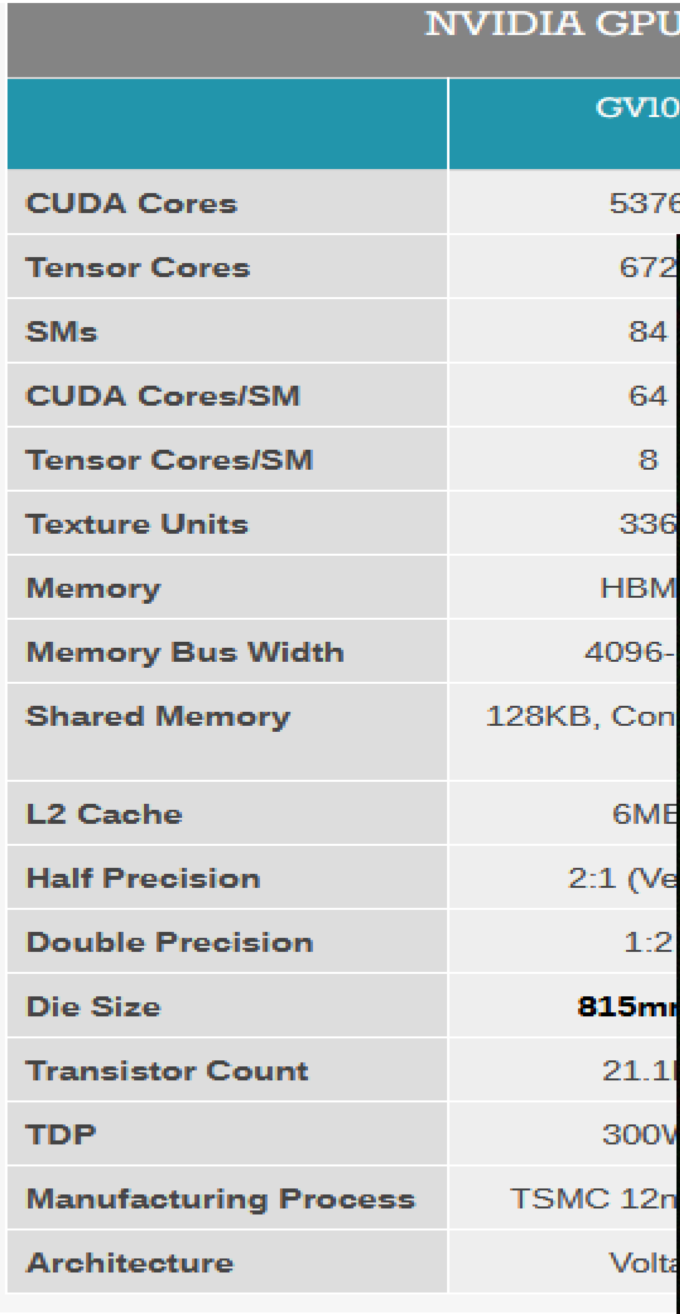
- Must be suitable for portable, low-power operation
  - Consumer products
  - Internet of Things components
- Must be inexpensive to manufacture
  - Comparable to current semiconductor technology
    - $O(1)$ cost to make chip with $O(N)$ devices
- Need not be based on transistors
  - Memristors, carbon nanotubes, DNA transcription, ...
  - Possibly new models of computation
  - But, still want lots of devices in an integrated system

Moore’s Law: 100 Years



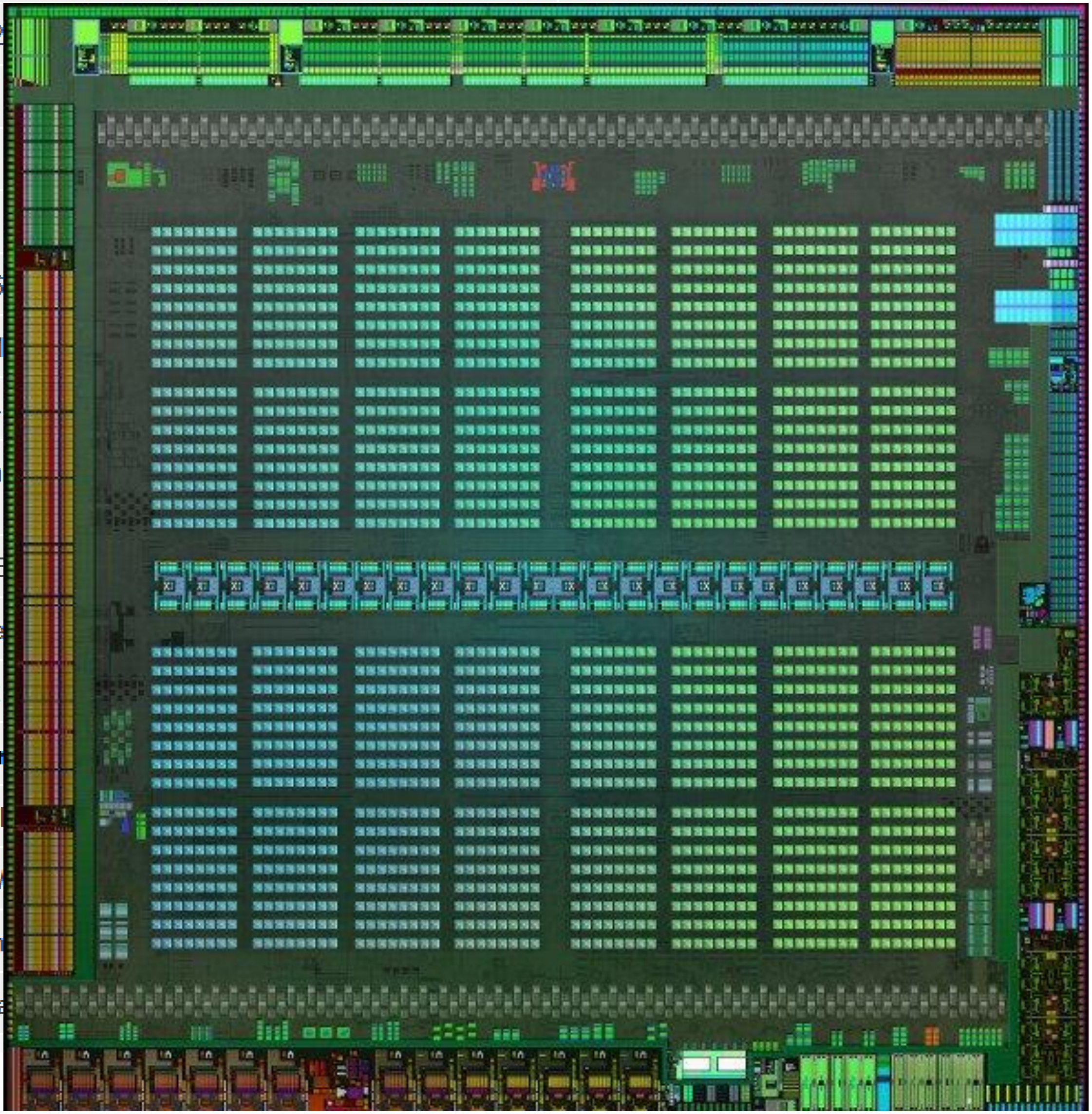
10^{17} devices!

NVIDIA GV100 Volta
2017
21.1 B transistors
815 mm^2

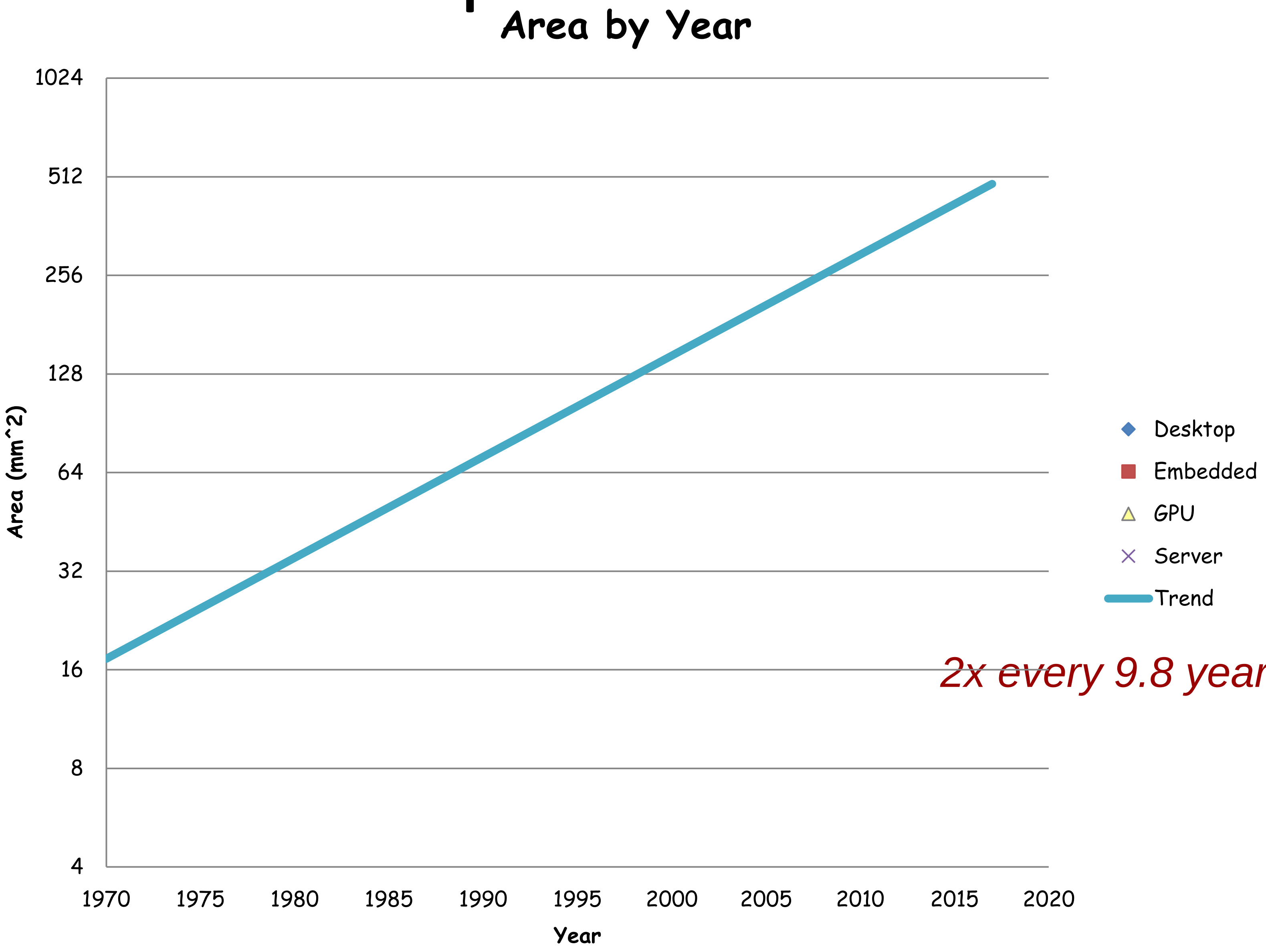


NVIDIA GPU Specification Comparison

	GV100	GP100	GK110
CUDA Cores	5376	3840	2880
Tensor Cores	672		
SMs	84		
CUDA Cores/SM	64		
Tensor Cores/SM	8		
Texture Units	336		
Memory	HBM		
Memory Bus Width	4096-		
Shared Memory	128KB, Con		
L2 Cache	6MB		
Half Precision	2:1 (Ve		
Double Precision	1:2		
Die Size	**815mm**		
Transistor Count	21.1B		
TDP	300W		
Manufacturing Process	TSMC 12n		
Architecture	Volta		

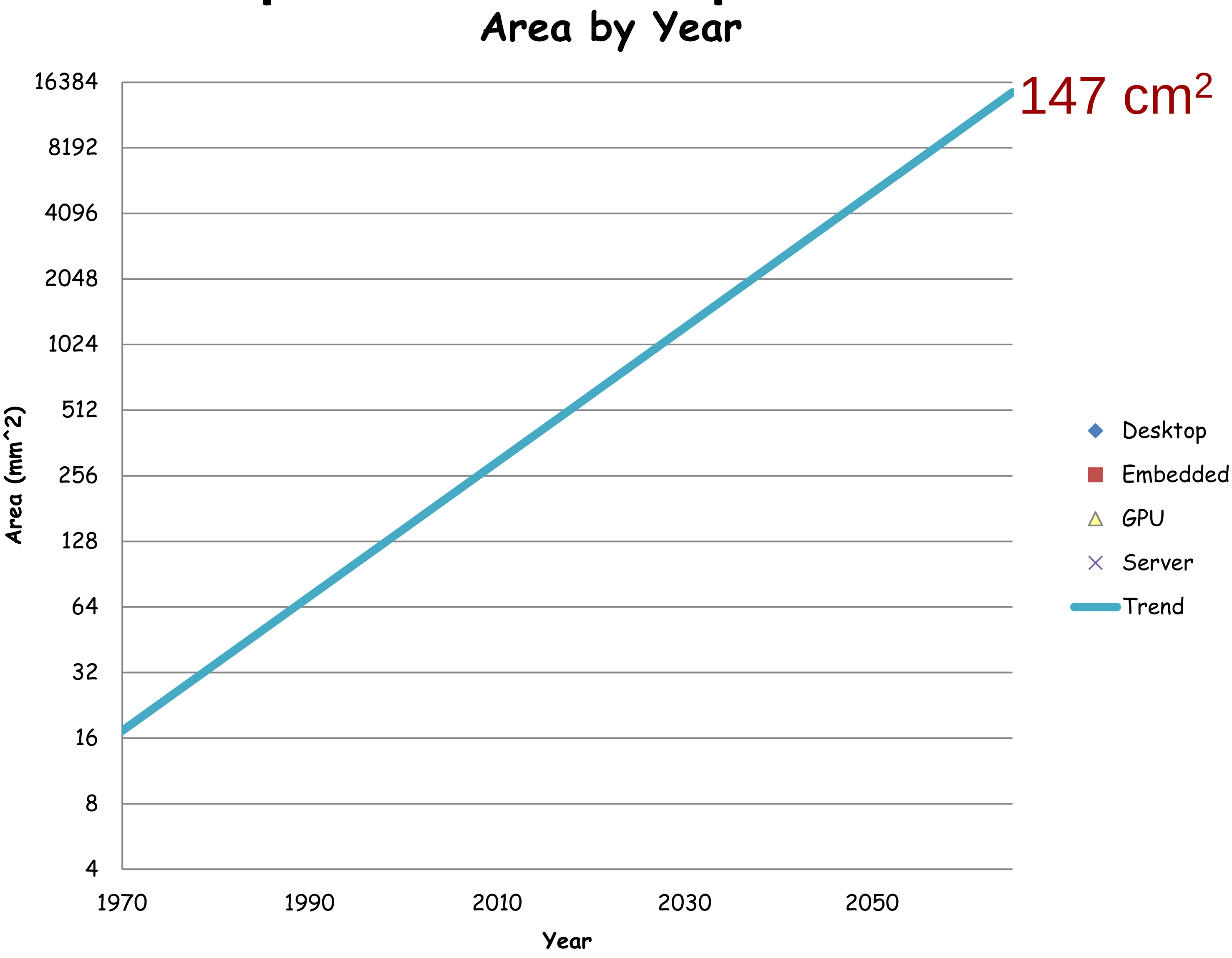


Chip Size Trend



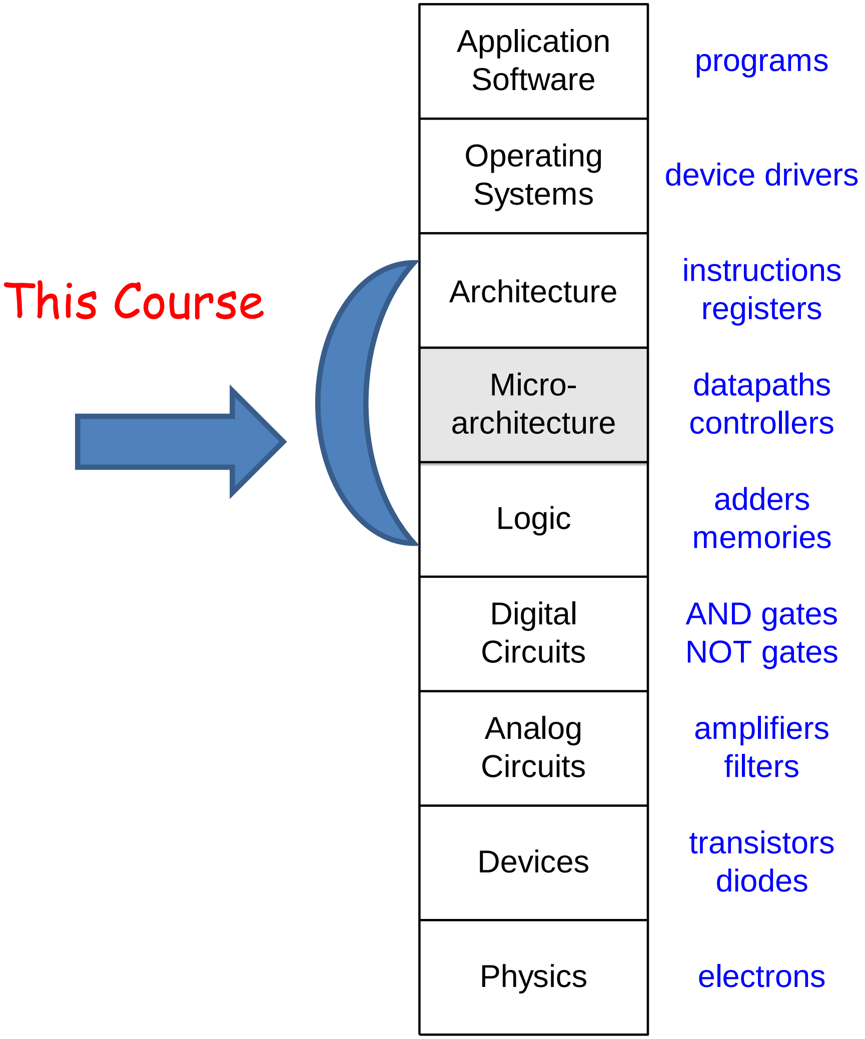
2x every 9.8 years

Chip Size Extrapolation



This Course

- Micro-architecture: how to implement an architecture in hardware
- Processor:
  - Datapath: functional blocks
  - Control: control signals



What is Computer Architecture?

- The science and art of designing, selecting, and interconnecting hardware components and designing the hardware/software interface to create a computing system that meets functional, performance, energy consumption, cost, and other specific goals.

- We will soon distinguish between the terms *architecture*, and *microarchitecture*.

Why Study Computer Architecture?

- Enable better systems: make computers faster, cheaper, smaller, more reliable, …
  - By exploiting advances and changes in underlying technology/circuits

- Enable new applications
  - Life-like 3D visualization 20 years ago?
  - Virtual reality?

- Enable better solutions to problems
  - Software innovation is built into trends and changes in computer architecture
    - > 50% performance improvement per year has enabled

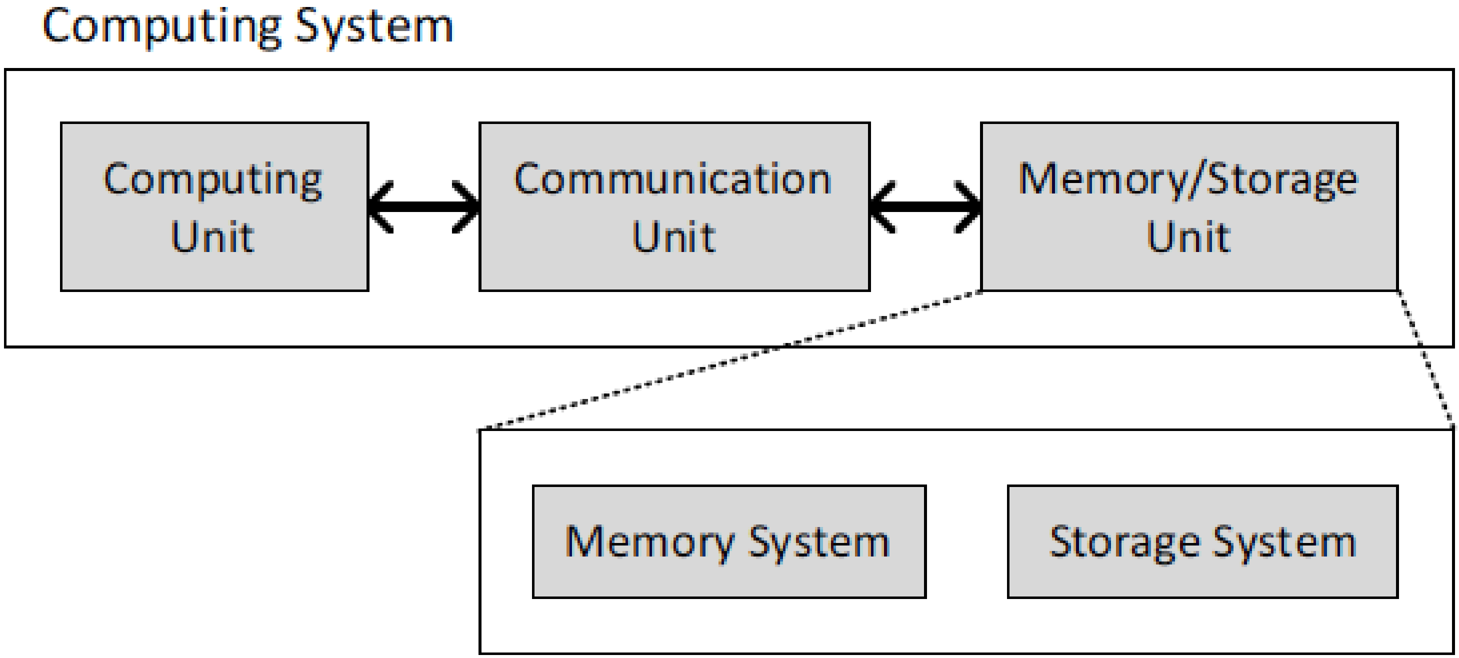
- Understand why computers work the way they do

Computer Architecture Today

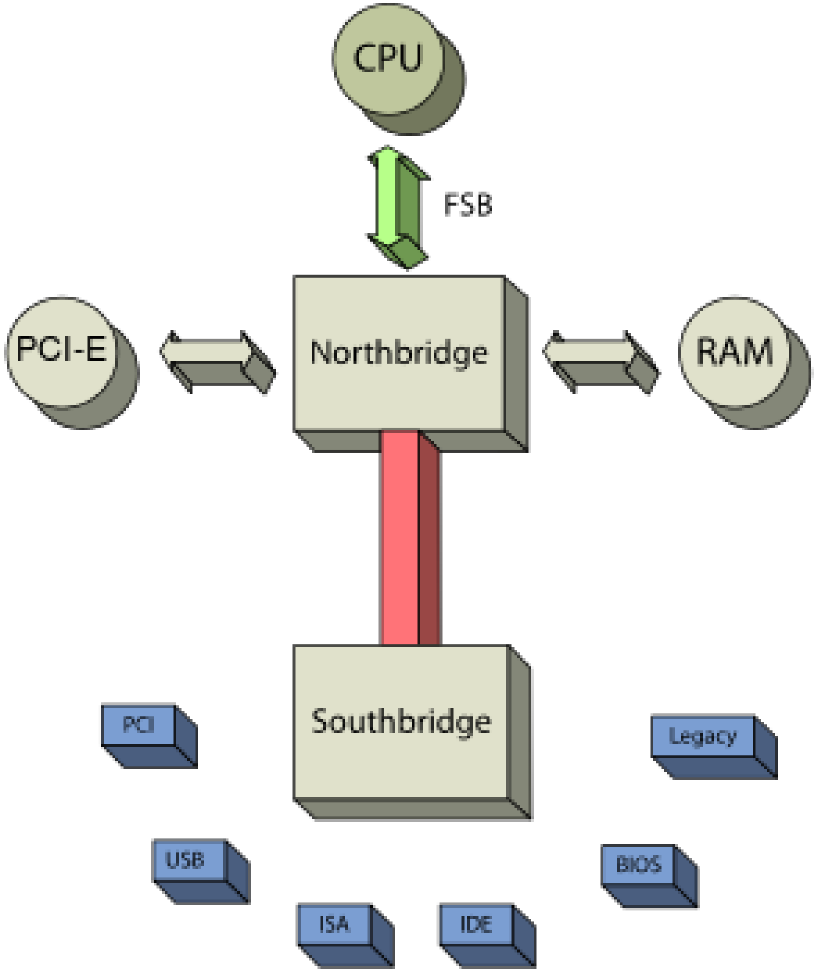
- Today is a very exciting time to study computer architecture
- Industry is in a large paradigm shift (to multi-core)
- Many problems motivating and caused by the shift
  - Power/energy constraints
  - Complexity of design → multi-core
  - Technology scaling → new technologies
  - Memory wall/gap
  - Reliability wall/issues
  - Programmability wall/problem

What is A Computer?

- Three key components
- Computation
- Communication
- Storage (memory)



Typical PC Architecture: **northbridge** or **host bridge**

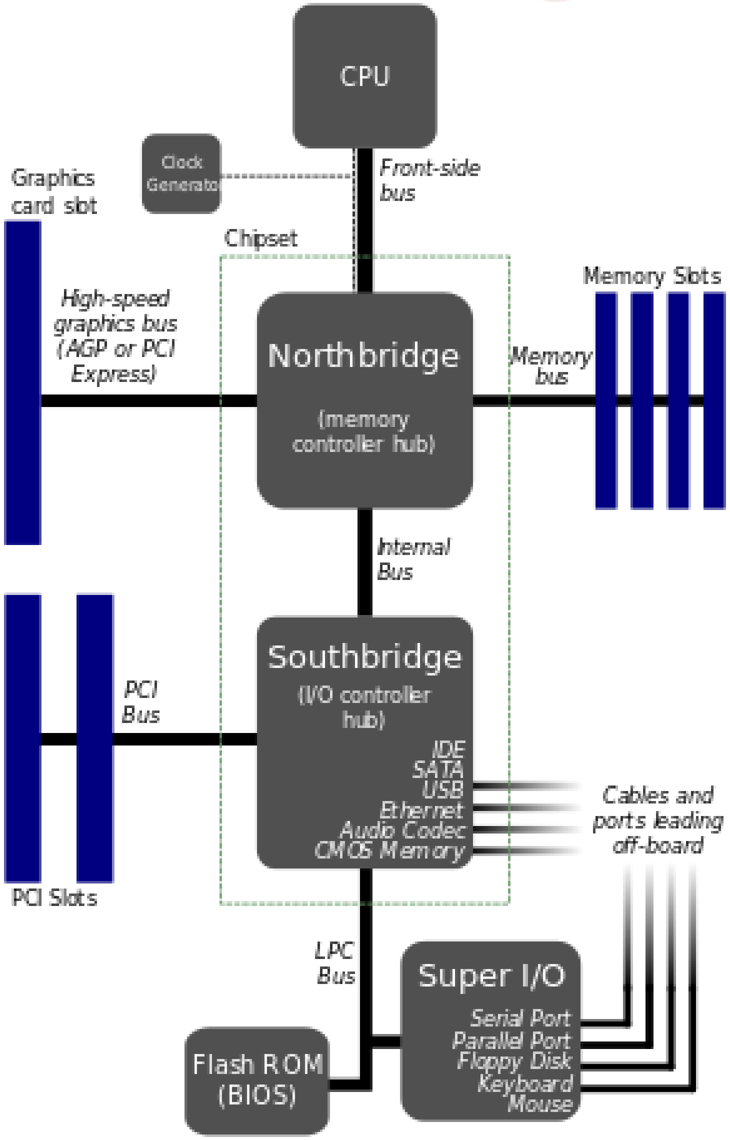


front-side bus (FSB)

I/O controller hub

(Typical PC Architecture)

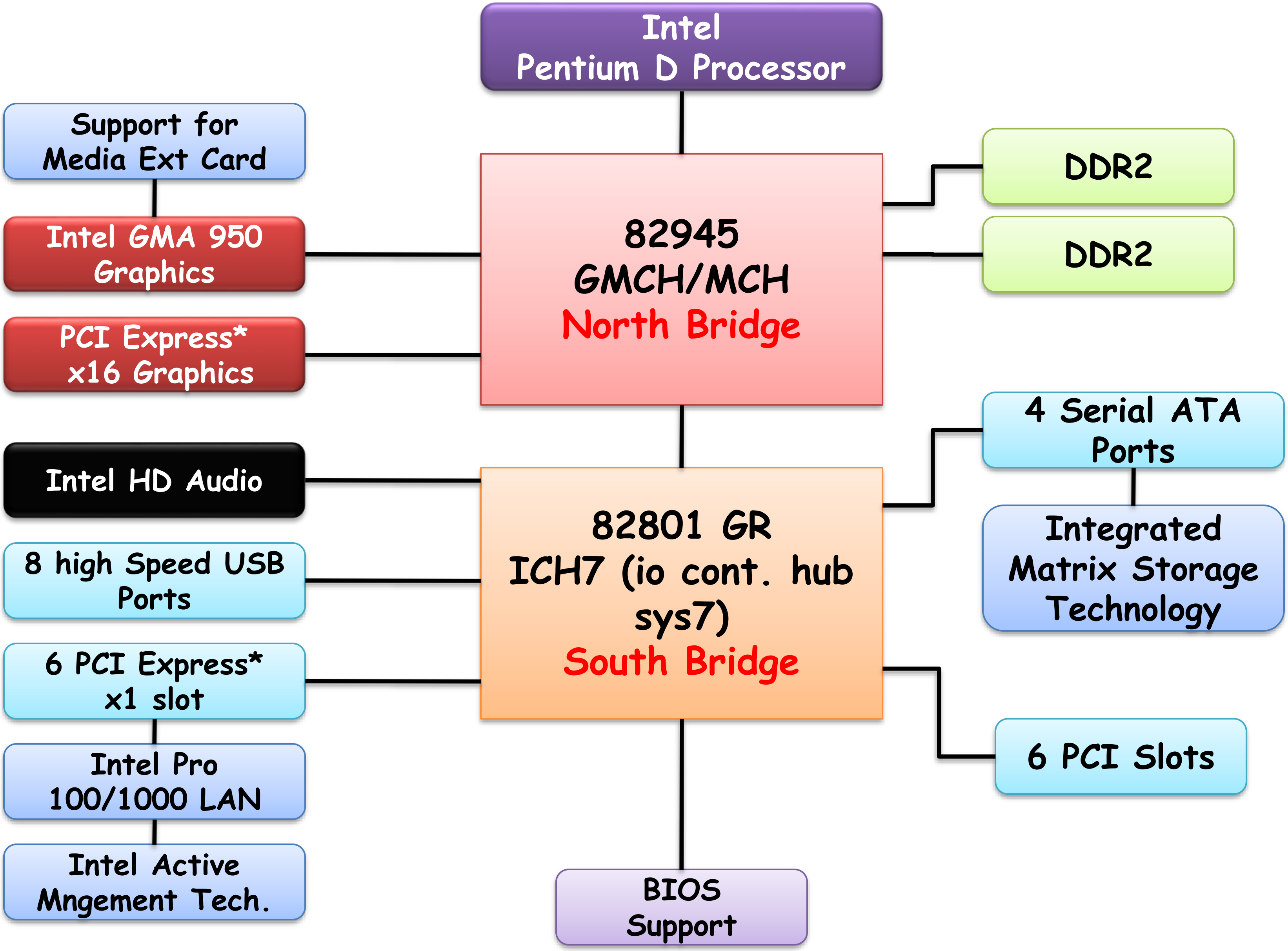
northbridge or **host bridge**



Graphics and Memory Controller Hu
(GMCH)

I/O Controller Hub (ICH)

Intel 945 Express Chipset



Intel Pentium D Processor

82945 GMCH/MCH North Bridge

Support for Media Ext Card

Intel GMA 950 Graphics

PCI Express* x16 Graphics

DDR2

DDR2

82801 GR ICH7 (io cont. hub sys7) South Bridge

Intel HD Audio

8 high Speed USB Ports

6 PCI Express* x1 slot

Intel Pro 100/1000 LAN

Intel Active Mngement Tech.

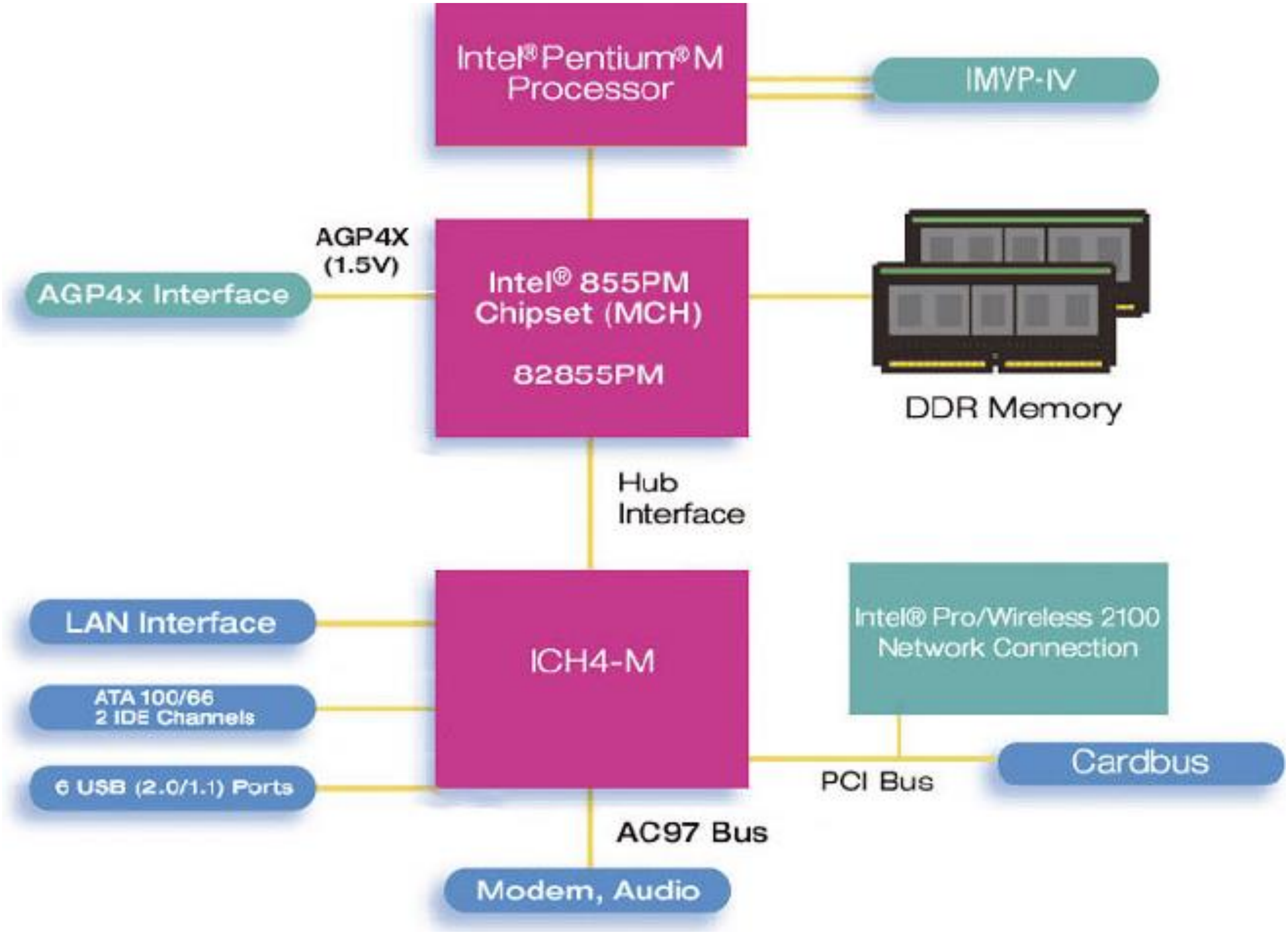
4 Serial ATA Ports

Integrated Matrix Storage Technology

6 PCI Slots

BIOS Support

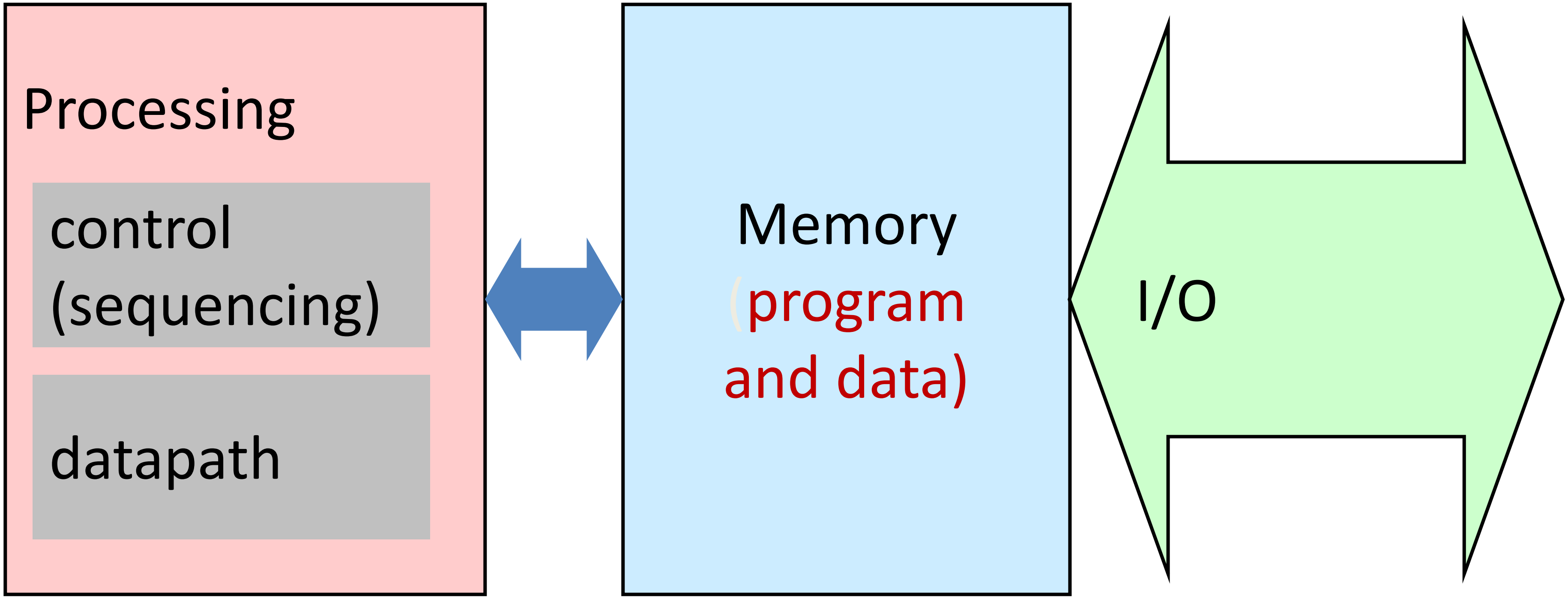
What is A Computer?



Courtesy, Intel

What is A Computer?

- We will cover all three components



Processing

control (sequencing)

datapath

Memory (program and data)

I/O

The Von Neumann Model/Architecture

- Also called *stored program computer* (instructions in memory). Two key properties:

- Stored program
  - Instructions stored in a linear memory array
  - Memory is unified between instructions and data
    - The interpretation of a stored value depends on the control signals

When is a value interpreted as an instruction?

- Sequential instruction processing
  - One instruction processed (fetched, executed, and completed) at a time
  - Program counter (instruction pointer) identifies the current instr.
  - Program counter is advanced sequentially except for control transfer instructions

Dataflow Model (of a Computer)

- Von Neumann model: An instruction is fetched and executed in control flow order
  - As specified by the instruction pointer
  - Sequential unless explicit control flow instruction

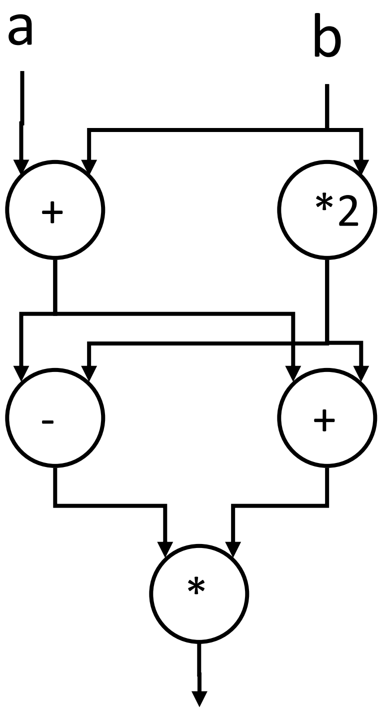
- Dataflow model: An instruction is fetched and executed in data flow order
  - i.e., when its operands are ready
  - i.e., there is no instruction pointer
  - Instruction ordering specified by data flow dependence
    - Each instruction specifies "who" should receive the result
    - An instruction can "fire" whenever all operands are received
  - Potentially many instructions can execute at the same time
    - Inherently more parallel

Von Neumann vs Dataflow

- Consider a von Neumann program
  - □ What is the significance of the program order?
  - □ What is the significance of the storage locations?

```
v <= a + b;
w <= b * 2;
x <= v - w
y <= v + w
z <= x * y
```

Sequential



Dataflow

- Which model is more natural to you as a programmer?

ISA vs. Microarchitecture Level Tradeoff

- A similar tradeoff (control vs. data-driven execution) can be made at the microarchitecture level

- ISA: Specifies how the programmer sees instructions to be executed
  - Programmer sees a sequential, control-flow execution order vs.
  - Programmer sees a data-flow execution order

- Microarchitecture: How the underlying implementation actually executes instructions
  - Microarchitecture can execute instructions in any order as long as it obeys the semantics specified by the ISA when making the instruction results visible to software
    - Programmer should see the order specified by the ISA

ISA vs. Microarchitecture Level

- ISA: Specifies how the programmer sees instructions to be executed

- Microarchitecture: How the underlying implementation actually executes instructions
  - Microarchitecture can execute instructions in any order as long as it obeys the semantics specified by the ISA when making the instruction results visible to software
    - Programmer should see the order specified by the ISA

The Von-Neumann Model

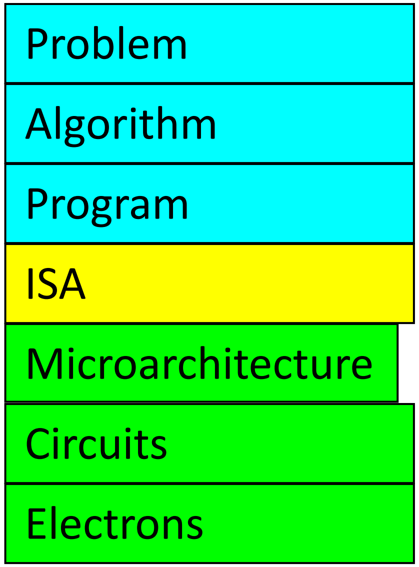
- All major *instruction set architectures* today use this model
  - x86, ARM, MIPS, SPARC, Alpha, POWER

- Underneath (at the microarchitecture level), the execution model of almost all *implementations (or, microarchitectures)* is very different
  - Pipelined instruction execution: *Intel 80486 uarch*
  - Multiple instructions at a time: *Intel Pentium uarch*
  - Out-of-order execution: *Intel Pentium Pro uarch*
  - Separate instruction and data caches

- But, what happens underneath that is *not* consistent with the von Neumann model is *not* exposed to software
  - Difference between ISA and microarchitecture

ISA vs. Microarchitecture

- ISA
  - Agreed upon interface between software and hardware
    - SW/compiler assumes, HW promises
  - What the software writer needs to know to write and debug system/user programs
- Microarchitecture
  - Specific implementation of an ISA
  - Not visible to the software
- Microprocessor
  - **ISA, uarch**, circuits
  - "Architecture" = ISA + microarchitecture



CISC versus RISC

CISC	**RISC**
Emphasis on hardware	Emphasis on software
Includes multi-clock complex instructions	Single-clock, reduced instruction only
Memory-to-memory: "LOAD" and "STORE" incorporated in instructions	Register to register: "LOAD" and "STORE" are independent instructions
Small code sizes, high cycles per second	Low cycles per second, large code sizes
Transistors used for storing complex instructions	Spends more transistors on memory registers