# 2024 Computer Architecture (アーキテクチャ学)

Instructors

Hiroaki Kobayashi

Masayuki Sato

(小林広明 佐藤雅之)

Ver. 11252024

# Instructors

- Hiroaki Kobayashi
  - Professor, Graduate School of Information Sciences
  - Office:Room523, 5-th floor, Mechanical Engineering Building No.2
  - Phone: 795-7010
  - Fax: 795-7011
  - E-mail: koba@tohoku.ac.jp
- Masayuki Sato
  - Associate Professor, Graduate School of Information Sciences
  - Office: Room 525 in the same building
  - Phone: 795-7012
  - E-mail: masa@tohoku.ac.jp

Kobayashi

Sato


**Mechanical Engineering Building No.2**

# Campus Map: Where are we?

# Google Classroom

**IM2011201 (code rsbxg6g) for students of Grad. School of Information Sciences**
**TM10028000 (code bae3pof) for students of Grad. School of Engineering**

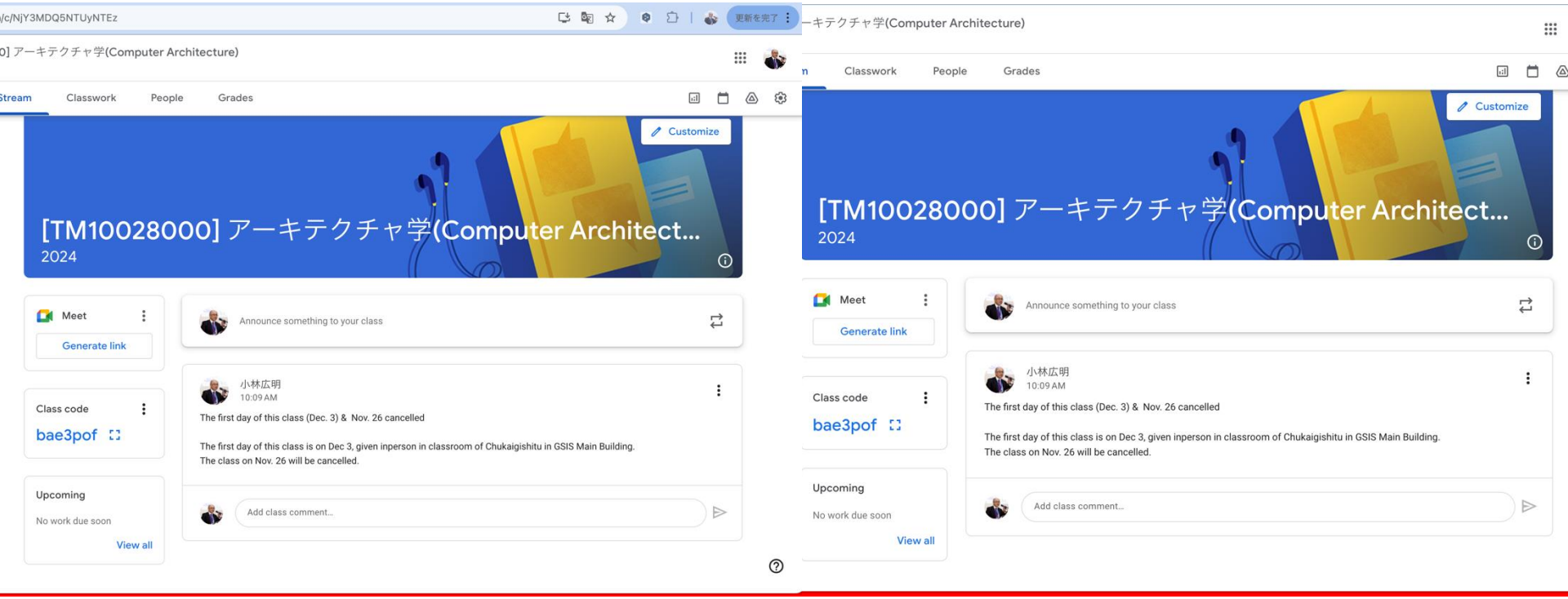
- Class Information
  - Instructors (Contact Info)
  - Syllabus
- Class Announcements
  - Schedule
  - Assignments and Exams
- Class Materials
  - Information about Reference Texts and Papers
  - Handouts

- Grading Policy
  - 50% from mid-term exam and 50% from final exam and
  - some additional home assignments for extra grading (bonus)

**Please prepare your personal copies of handouts by yourself before attending the class!**

# On-line Classes

- Classes are basically conducted in person.  If an on-line class (live) is needed, we will let you know via google classroom, and delivered at https://meet.google.com/gtx-edze-asy



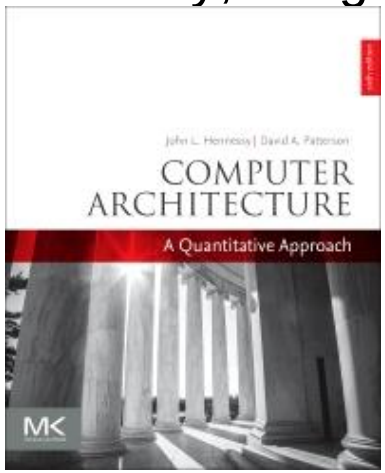
# Course Objective

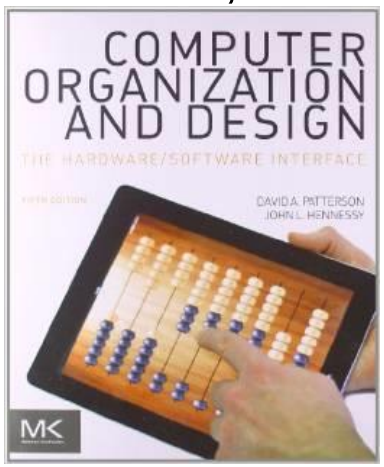
- Study the details about Computer Architecture Design, especially
  - Technology trends
  - Key factors to drive a dramatic progress in computer performance
  - Hardware and software technologies
    - ISA Design
    - Pipelining
    - Parallel processing
    - Hierarchical memory systems
    - Multicores, multiprocessors and cluster systems
    - High-performance special processors
  - Design tradeoffs between performance and cost

**This class is strongly correlated with High-Performance Computing class (by Prof. Takizawa)!**
**This class also covers the advanced topics of Class "Computer Structure (by Prof. Aoki)"**

# Reference Text

1. Computer Architecture: A Quantitative Approach, Sixth Edition, J.L.Hennessy & D.A.Patterson, Morgan Kaufmann Publishers, Inc., eBook ISBN: 9780128119068, Paperback ISBN: 9780128119051
2. Computer Organization and Design: The Hardware/Software Interface, D. Patterson and J. Hennessy, Morgan Kaufmann/Elsevier, 2013



# Contents and Schedule

1. Trends in Computer Architecture Design
2. Instruction Set Principles and Examples
3. Pipelining
4. Memory Hierarchy
5. Instruction Level Parallelism and Its Exploitation
6. Multicores, Multiprocessors, and Clusters
7. Data-Level Parallelism: SIMD, Vector, and GPU
8. Domain Specific Architecture
9. Warehouse-Scale Computers

1 by Sato, 2~4 by Kobayashi, 5~9 by Sato

Contents and schedule are subject to change.

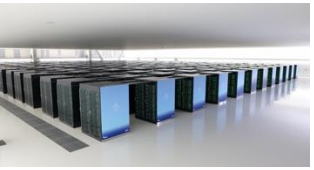
# Chapter 1
# Introduction to and Trends in Computer Architecture Design

# Contents

- Introduction
- Classes of Computers
- Defining Computer Architecture
- Trends in Technology
- Measuring, Reporting, and Summarizing Performance
- 5 Quantitative Principles of Computer Design

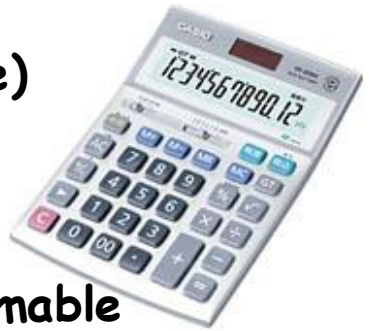
# What is a Computer…

- *A programmable electronic device that can store, retrieve and process data ---Webster*



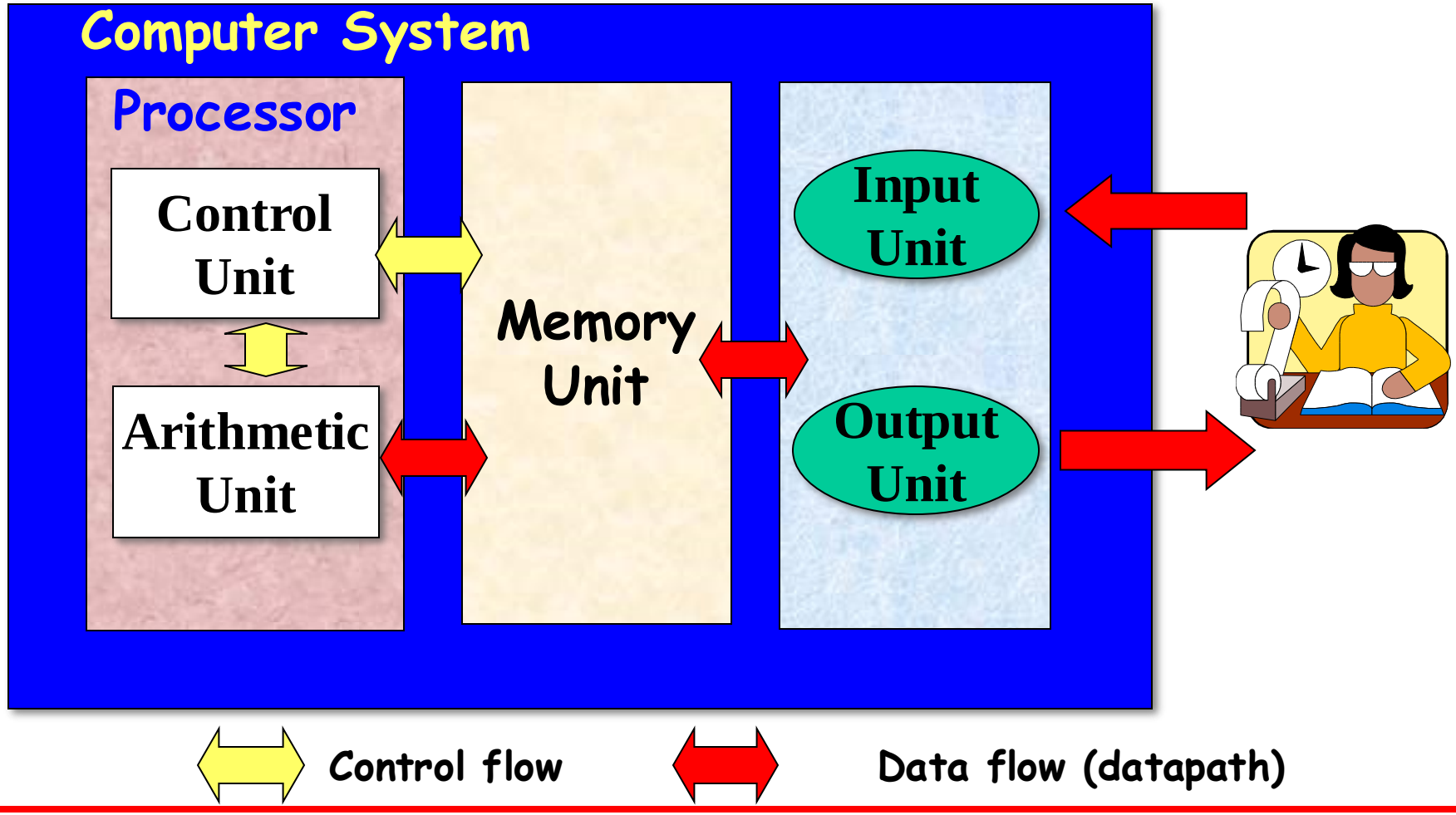
- Personal computer
- Workstation
- Game Console
- Supercomputer

**They have the following basic functions:**
**•Input**
**•Output**
**•Memory**
**•Calculation**
**•Control (programmable)**

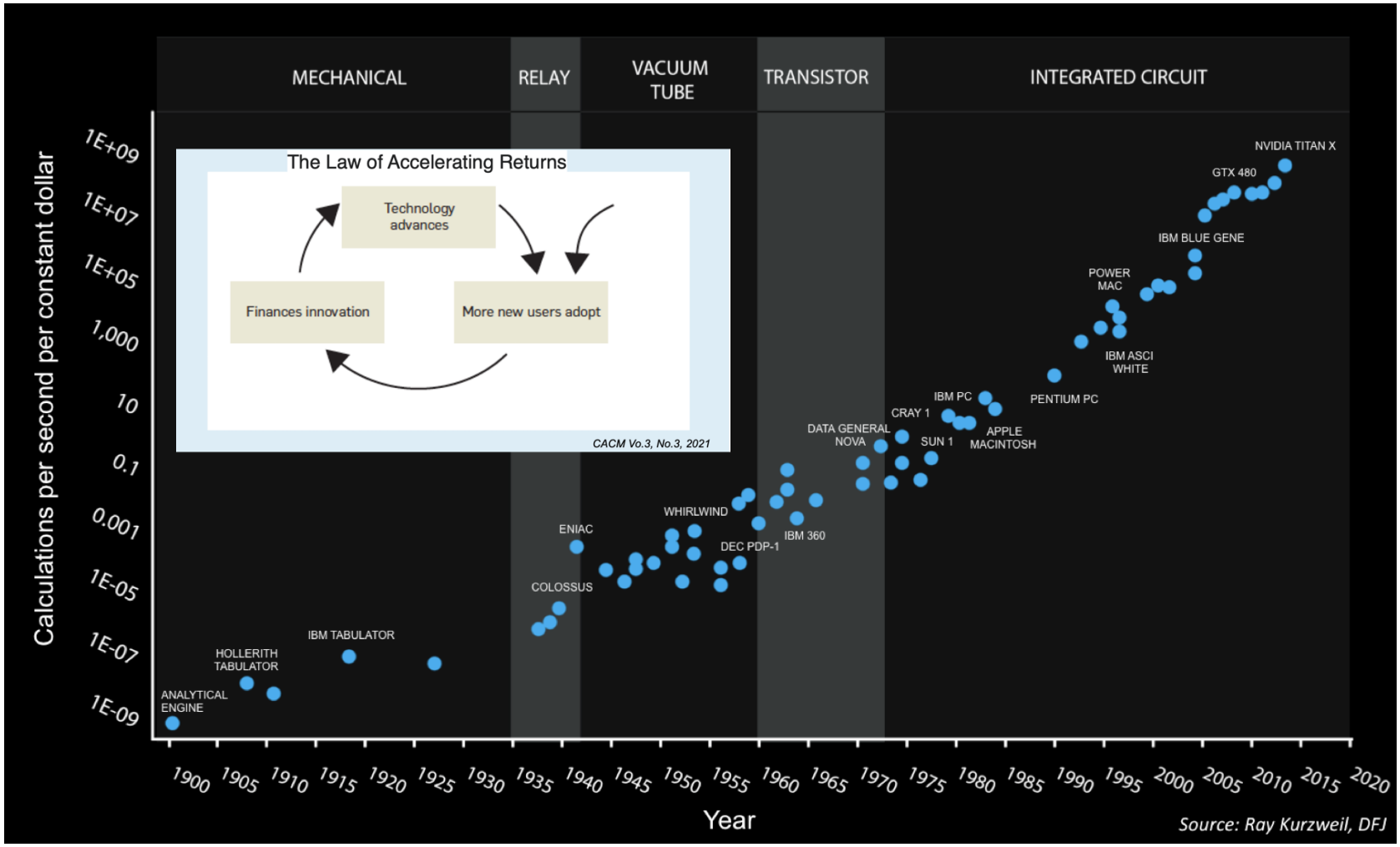


**⇔Calculator:does calculations, but is not programmable (need to control calculation sequence by operators (user)**

# Basic Structure of Computer: Von-Neumann Model

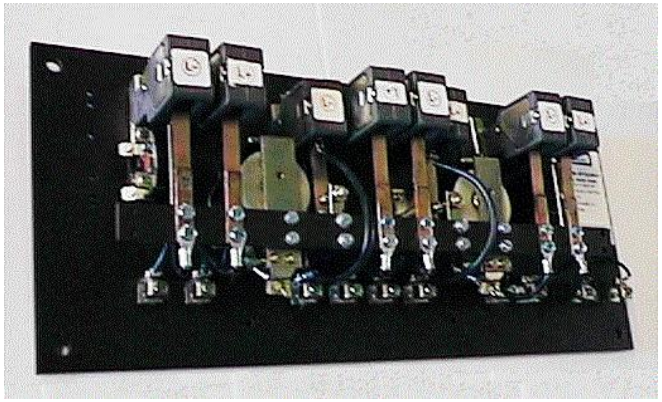


# Moore's Law

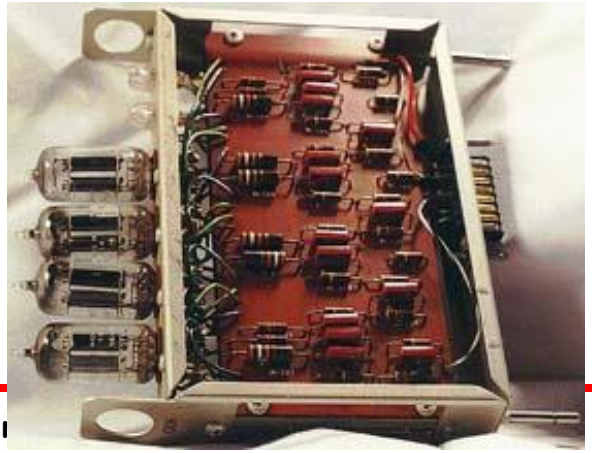

MECHANICAL
RELAY
VACUUM TUBE
TRANSISTOR
INTEGRATED CIRCUIT
Calculations per second per constant dollar
1E+09
1E+07
1E+05
1,000
10
0.1
0.001
1E-05
1E-07
1E-09
The Law of Accelerating Returns
Technology advances
Finances innovation
More new users adopt
CACM Vo.3, No.3, 2021
NVIDIA TITAN X
GTX 480
IBM BLUE GENE
POWER MAC
IBM ASCI WHITE
PENTIUM PC
IBM PC
CRAY 1
APPLE MACINTOSH
DATA GENERAL NOVA
SUN 1
WHIRLWIND
ENIAC
DEC PDP-1
IBM 360
COLOSSUS
IBM TABULATOR
HOLLERITH TABULATOR
ANALYTICAL ENGINE
1900 1905 1910 1915 1920 1925 1930 1935 1940 1945 1950 1955 1960 1965 1970 1975 1980 1985 1990 1995 2000 2005 2010 2015 2020
Year
Source: Ray Kurzweil, DFJ

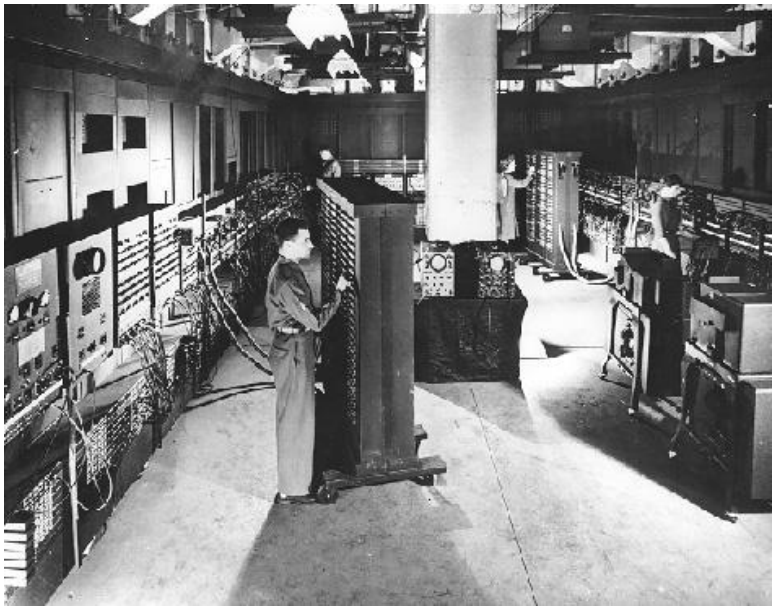
# Rapid Progress in Device Technology

**Relay (1940s)**
**（Mechanical Switch）**



**Vacuum tube(1950s)**





ENIAC（1946）
First general-purpose electric computer

**EDSAC in 1949 was the first stored-program computer**

# Rapid Progress in Device Technology

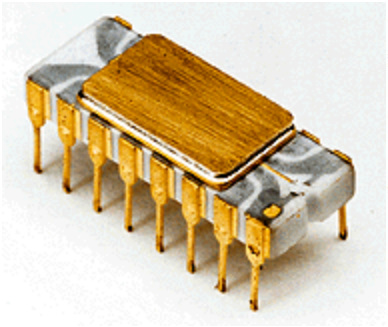
**Transister(1960s)**

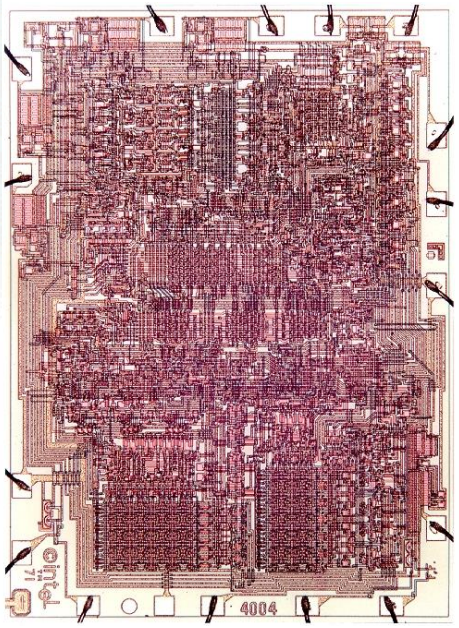
IBM 360(1964)
Commercial General-Purpose Computer family

**UNIVAC in 1952 was the first commercial computer**

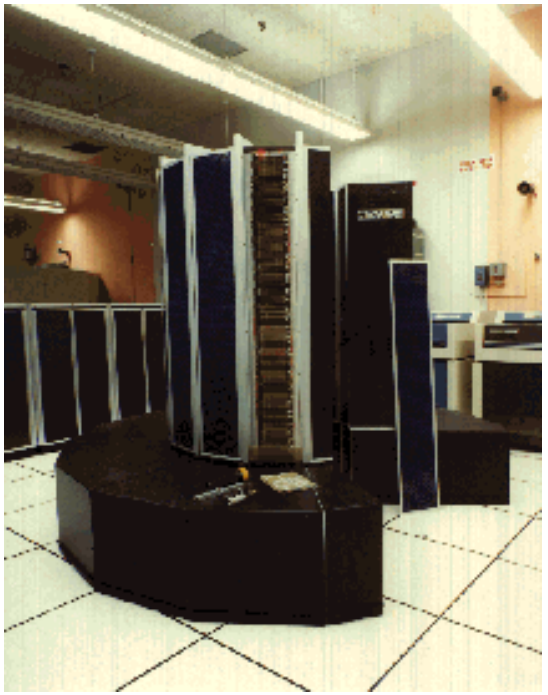
# Rapid Progress in Device Technology

IC/TTL (1970s)






Intel 4004(1971)
First microprocessor design

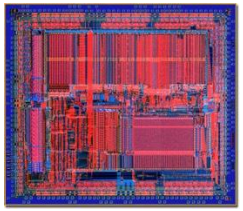


CRAY-1 (1976)
First Supercomputer

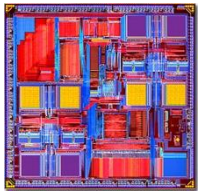
# Rapid Progress in Device Technology

**VLSI(1980~present)**


**RISC Processors**



**MIPS (1981)**



**SPARC: 1981**

Apple Computer

Apple II (1977)

Lisa (1983)

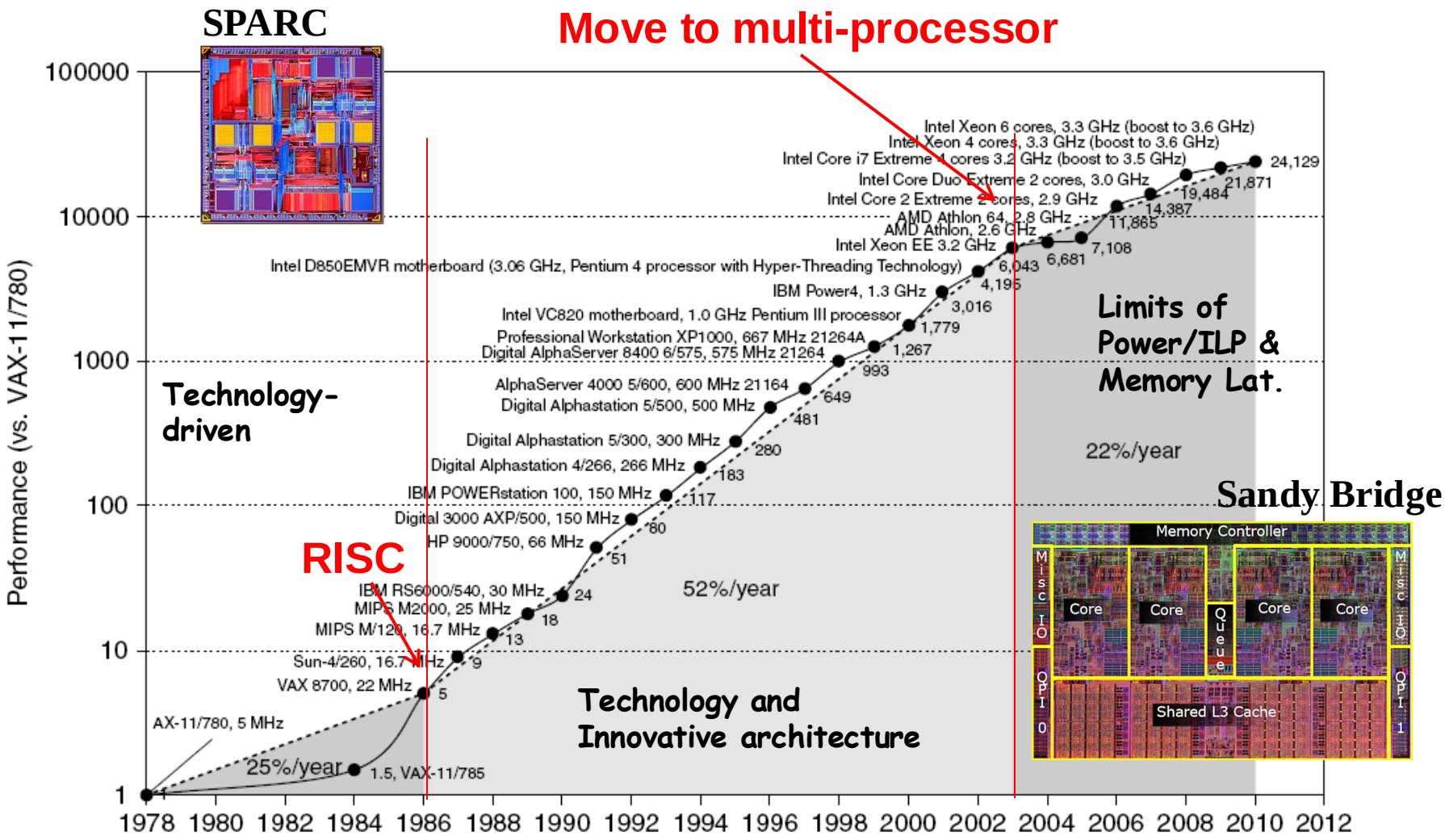
IBM PC (1981) with 8086 processor

Workstation
Sun Workstation 1 (1981)

# Drastic Cost/Performance Improvement in Last 5 decades

| | 1965 | 1977 | 2019 |
|---|---|---|---|
| |  |  |  |
| | **IBM System 360/50** | **DEC VAX11/780** | **AMD Ryzen 7 3700X (3.6GHz)** |
| Performance（ratio） | 0.15MIPS(1) | 1MIPS(6.7x) | 382,760MIPS(2.6Mx) |
| Memory（ratio） | 64KB(1) | 1MB(16x) | 16GB(256Kx) |
| Price | 360M yen | 72M yen | 100K yen |
| Price/performance | 1 | 33 times | More than 19B times! |

# Introduction: Incredible progress in computer technology over last 60 years



# Classes of Computers

- Changes in computer systems
  - Large mainframes in 1960s
  - Minicomputer and Supercomputers in 1970s
  - Desktop computers in 1980
  - Handheld computing devices (PDA, Video games) in 1990s
  - Cell phones in 2000
- These changes in computer use have led to five different computing markets

| Feature | Personal mobile device (PMD) | Desktop | Server | Clusters/warehouse-scale computer | Embedded |
|---|---|---|---|---|---|
| Price of system | $100–$1000 | $300–$2500 | $5000–$10,000,000 | $100,000–$200,000,000 | $10–$100,000 |
| Price of micro-processor | $10–$100 | $50–$500 | $200–$2000 | $50–$250 | $0.01–$100 |
| Critical system design issues | Cost, energy, media performance, responsiveness | Price-performance, energy, graphics performance | Throughput, availability, scalability, energy | Price-performance, throughput, energy proportionality | Price, energy, application-specific performance |

# Dependability is a Key Issue for Servers and Cluster/Warehouse-Scale Computers

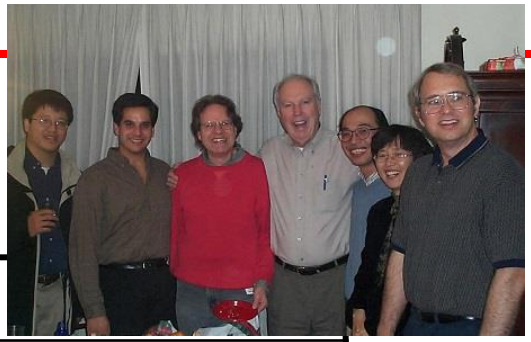
| Application | Cost of downtime per hour | Annual losses with downtime of | | |
|---|---|---|---|---|
| | | 1% (87.6 hrs/yr) | 0.5% (43.8 hrs/yr) | 0.1% (8.8 hrs/yr) |
| Brokerage operations | $6,450,000 | $565,000,000 | $283,000,000 | $56,500,000 |
| Credit card authorization | $2,600,000 | $228,000,000 | $114,000,000 | $22,800,000 |
| Package shipping services | $150,000 | $13,000,000 | $6,600,000 | $1,300,000 |
| Home shopping channel | $113,000 | $9,900,000 | $4,900,000 | $1,000,000 |
| Catalog sales center | $90,000 | $7,900,000 | $3,900,000 | $800,000 |
| Airline reservation center | $89,000 | $7,900,000 | $3,900,000 | $800,000 |
| Cellular service activation | $41,000 | $3,600,000 | $1,800,000 | $400,000 |
| Online network fees | $25,000 | $2,200,000 | $1,100,000 | $200,000 |
| ATM service fees | $14,000 | $1,200,000 | $600,000 | $100,000 |

**Figure 1.3** Costs rounded to nearest $100,000 of an unavailable system are shown by analyzing the cost of downtime (in terms of immediately lost revenue), assuming three different levels of availability and that downtime is distributed uniformly. These data are from Kembel [2000] and were collected and analyzed by Contingency Planning Research.

# Class of Parallelism and Parallel Architectures

- Two parallelism in applications
  - Data-Level Parallelism (DLP)
    - Parallelism exploited in many data items that can be operated on at the same time
  - Task-Level Parallelism (TLP)
    - Parallelism exploited in tasks of work that can operate independently and largely in parallel
- Computer hardware can use these two kinds of applications parallelism in four major way.
  - Pipelining and Superscalar use Instruction-Level Parallelism
  - SIMD/Vector Processing/GPU use data level parallelism with a single instruction for a collection of data in parallel.
  - multicore processors/ parallel processors use Thread-Level Parallelism created by using data-level parallelism or task-level parallelism
  - Request-Level Parallelism exploits parallelism among largely decoupled tasks specified by the programmer or the operating system

# Classifications Based on Instruction and Data Streams



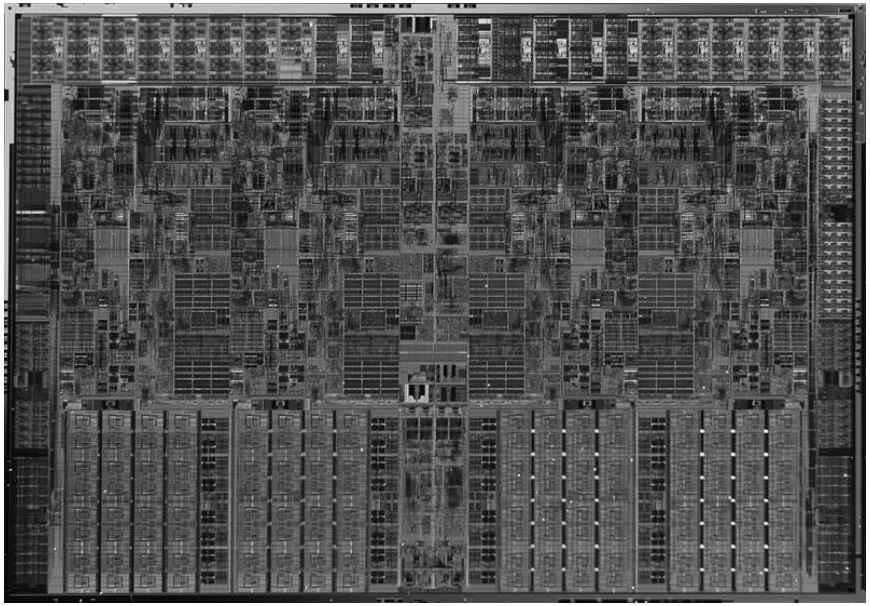
- Flynn's Classification (1966)

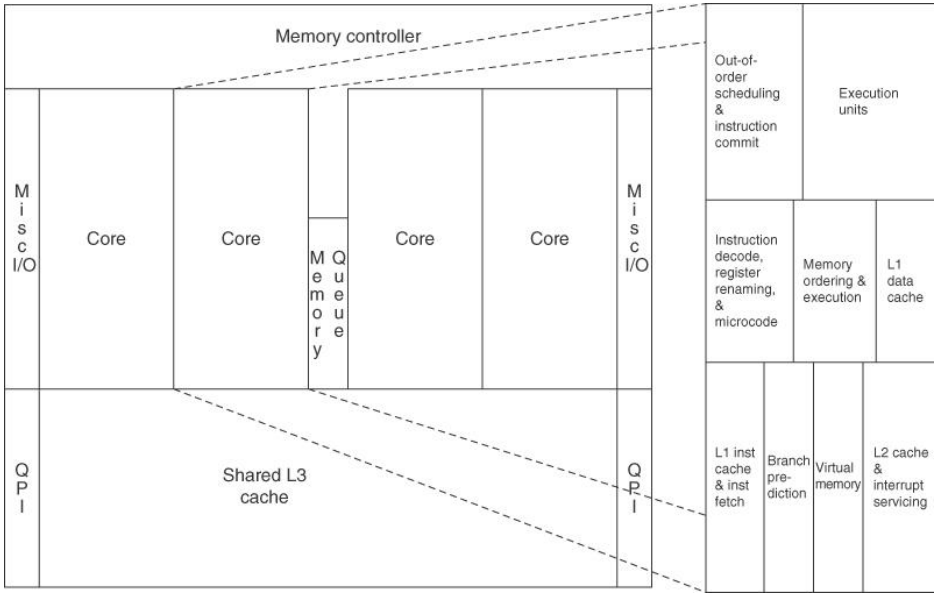
| | | Data Streams | |
|---|---|---|---|
| | | Single | Multiple |
| Instruction Streams | Single | **SISD**: Intel Pentium 4 | **SIMD**: SSE/AVX of Core i7 |
| | Multiple | **MISD**: No examples today | **MIMD**: multicore Core i7 |

- **SPMD: Single Program Multiple Data**
  - **A parallel program on a MIMD computer**
  - **Conditional code for different processors**

# Example: Core i7 Chip-Microarchitecture

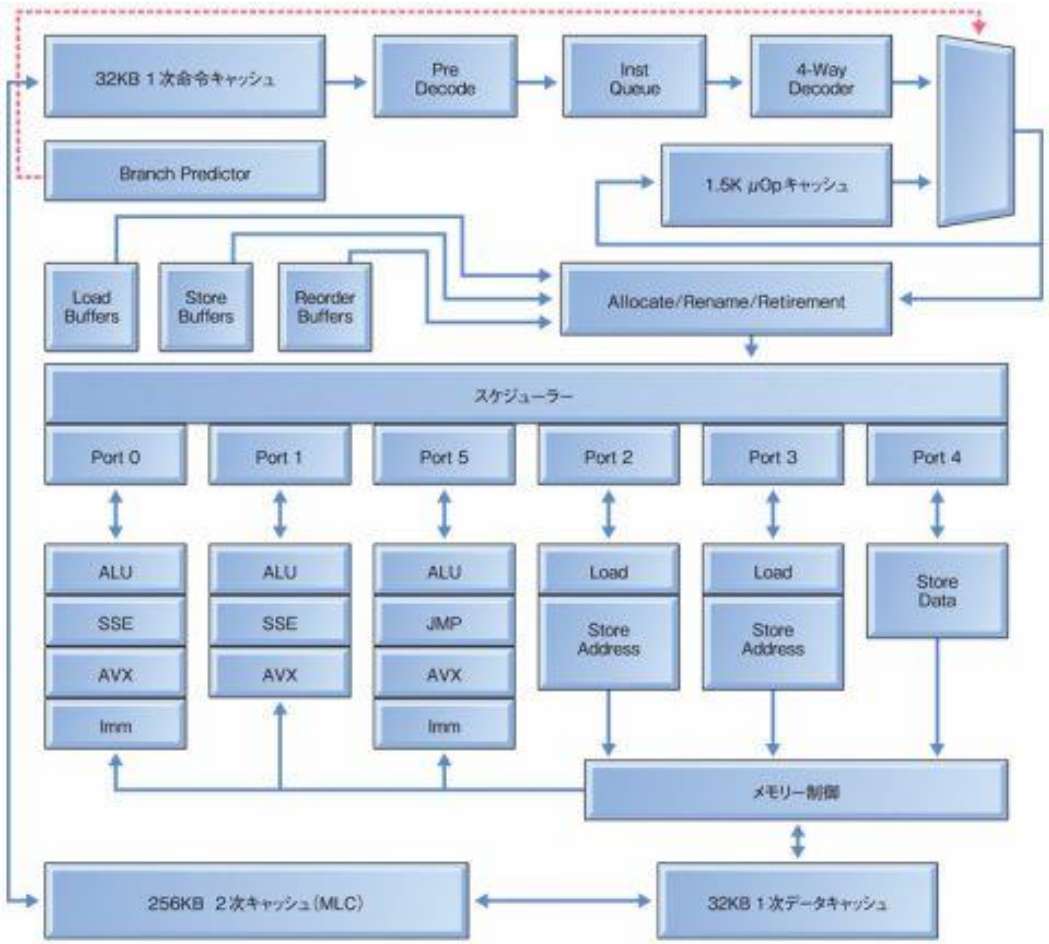
**Hierarchical integration of pipelines/superscalar for ILP, SIMD for DLP, and MIMD for thread/task parallelism**





# Example: Core i7 Core-Microarchitecture

**Hierarchical integration of pipelines/superscalar for ILP, SIMD for DLP, and MIMD for thread/task parallelism**



# Defining Computer Architecture

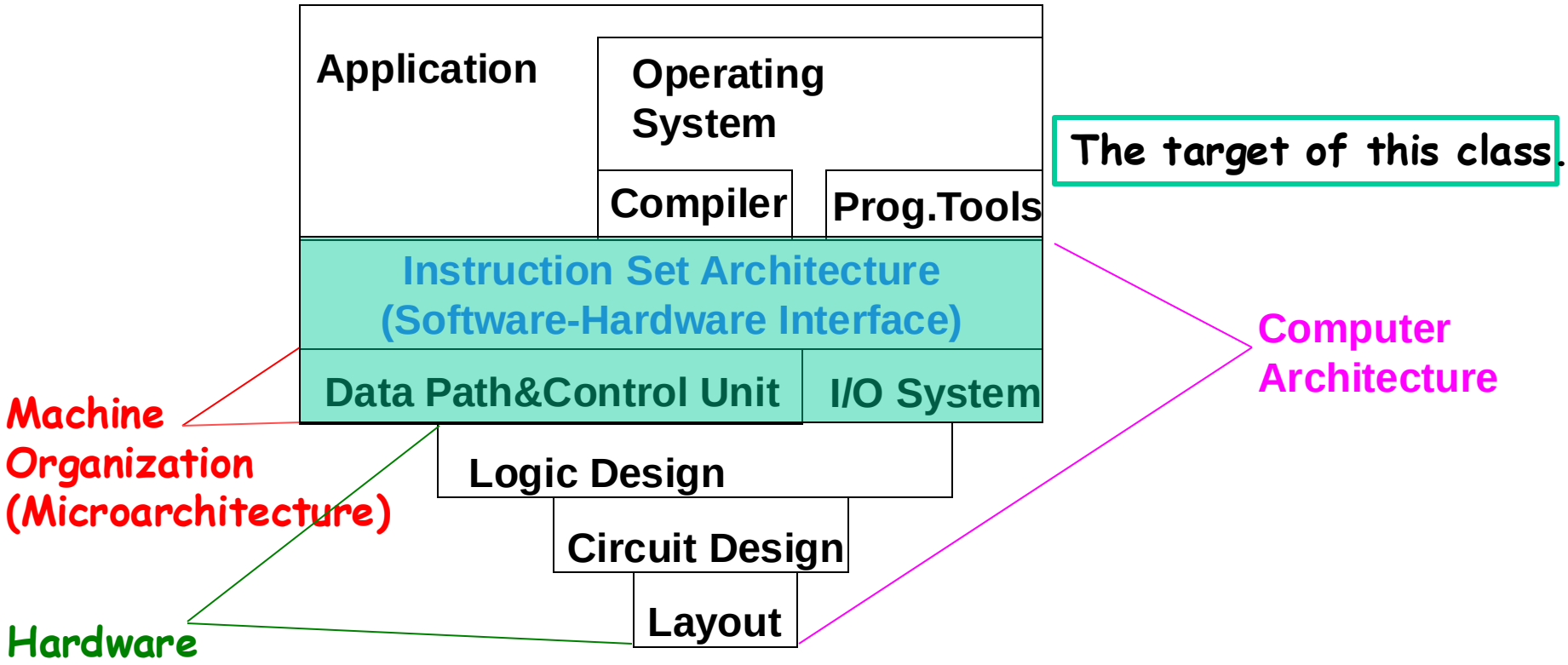
- If Computer Architecture == ISA

  then Computer architecture design == ISA design??

  - No! Computer architecture design includes ISA design and its implementation

  - What is ISA?
    - the actual programmer-visible instruction set
    - The boundary between the software and hardware

# Computer System Layers



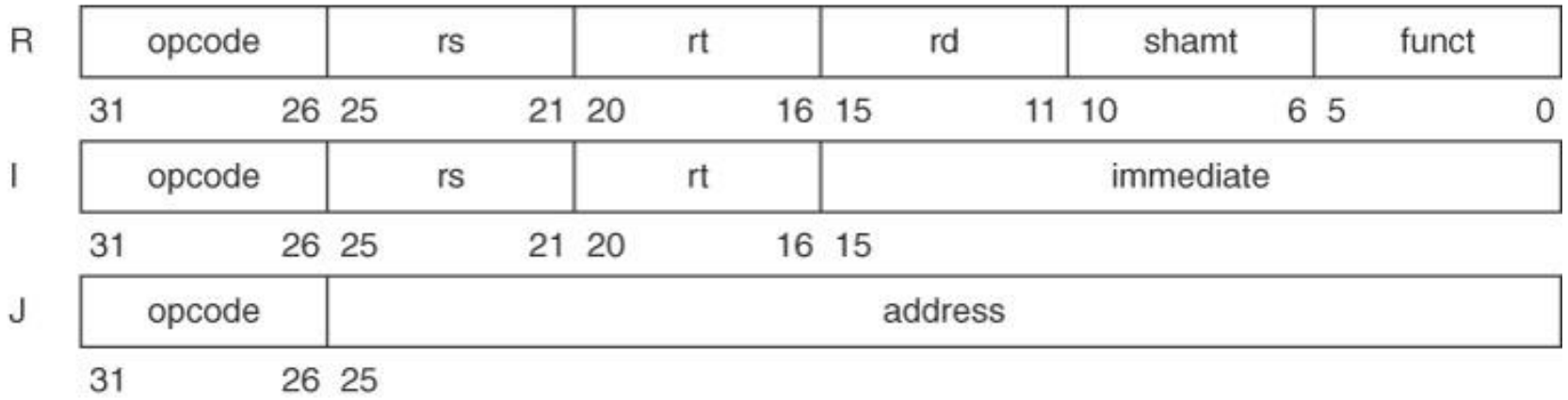

Computer Architecture ≡
Instruction Set Architecture+Machine Organization+Hardware

# Seven Dimension of an ISA

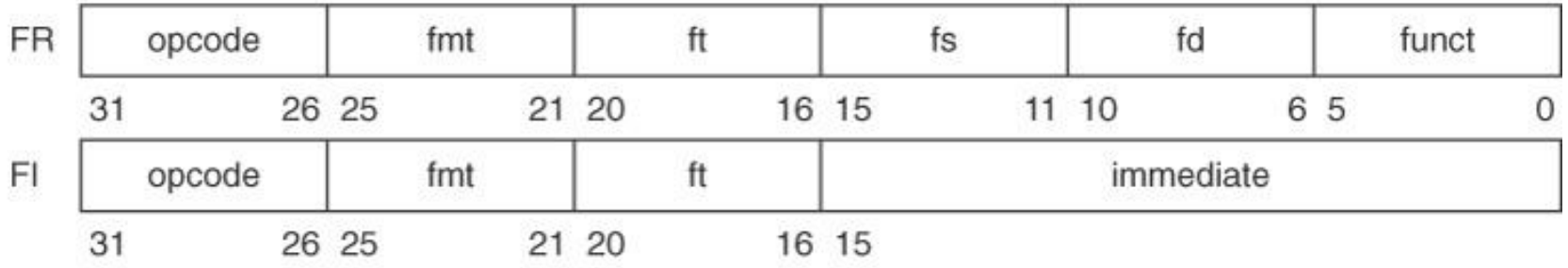
- Class of ISA
  - General-purpose register architectures
  - ✓ Register-memory ISAs of 80x86
  - ✓ Load-store ISAs of MIPS
- Memory addressing
  - Byte-addressing
  - Aligned (MIPS)
  - Not aligned (80x86)
- Addressing modes
  - register, immediate, displacement, register indirect, based with scaled index
- Types and sizes of operands
  - 8-bit (char), 16-bit (unicode char, half-word), 32-bit (integer or word), 64-bit (double word or long int), and IEEE 745 fp in 32-bit(single)&64-bit(dobule), 80-bit fp in 80x86)
- Operations
  - Data transfer, arithmetic logical, control, and floating point
  - MIPS is simple but 80x86 has a much richer and larger set
- Control flow instructions
  - Cond branch, uncond jump, procedure call and return
  - PC-relative addressing
  - MIPS cond branches test the contents of registers
  - 80x86 branches test condition code bits set as side effects of operations
- Encoding an ISA
  - Fixed length (32-bit fix in MIPS) and variable length (1 to 18 bytes in 80x86)
  - Variable length inst. can take less space, but needs complicated decoding
  - # of registers and # of addressing modes have a significant impact on size of instructions

# MIPS ISA

Basic instruction formats




Floating-point instruction formats



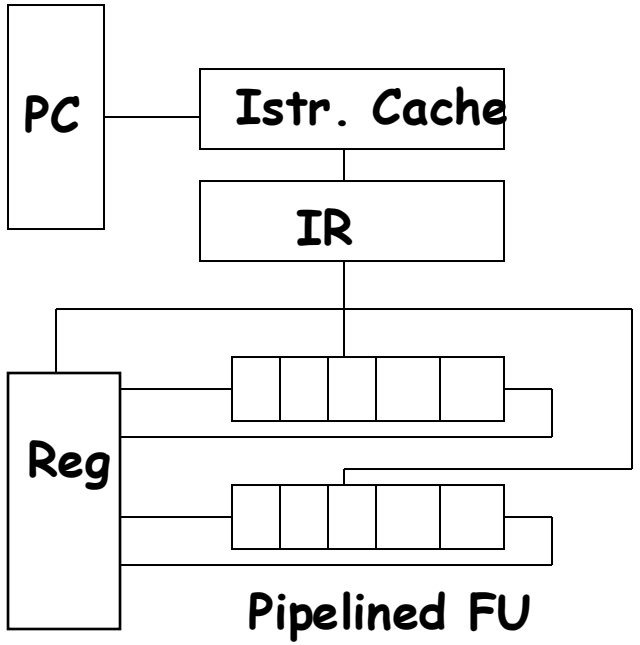
# The Rest of Computer Architecture: Designing the Organization and Hardware

- Designing the organization and hardware = implementation of an ISA
- Organization: the high-level aspects of a computer's design
  - Memory system, memory interconnect, and the design of the internal processor or CPU
  - Ex. AMD vs. Intel in X86 architecture design implementation
- Hardware: the specifics of a computer, including the detailed logic design and the packaging technology
  - Ex. Intel Core i series of i9(highest end), i7, i5, and i3 (lowest end)
    - ➢ Nearly identical, but offer different clock rates and different memory systems, high-performance oriented vs low power oriented.

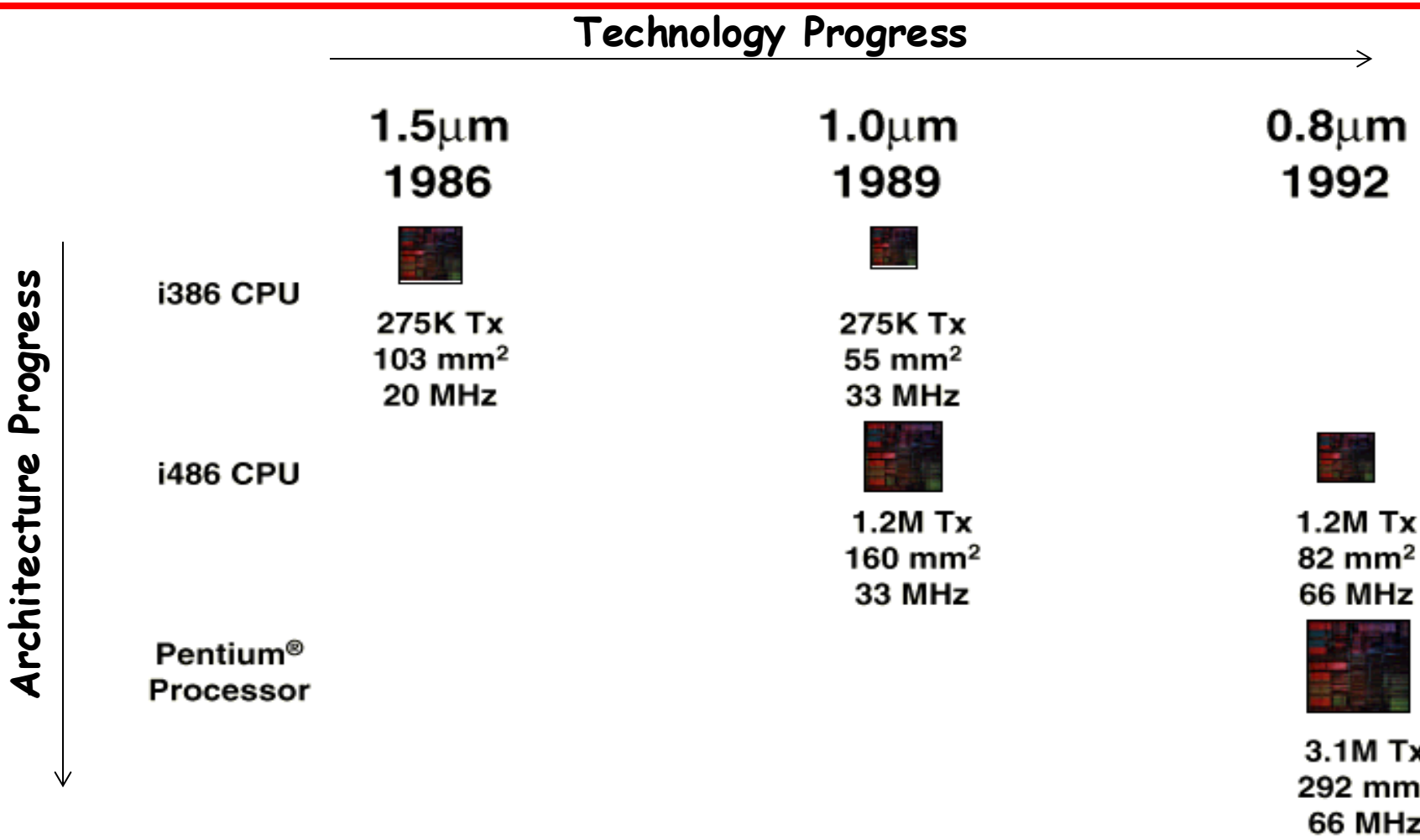
**Computer architecture = ISA, Organization, and Hardware**

# Microarchitecture Design= Data Path & Control Units Design

- Pipelining
- Instruction Level Parallel Processing
- On-chip cache and TLB
- Speculation and Branch Prediction



# Processor Generation



# Intel's TICK-TOCK Strategy, and Then Process-Architecture-Optimization Strategy







Source by Intel

# Requirements in Designing a New Computer

- Design a computer to meet functional requirements as well as price, power, performance, and availability goals.

| Functional requirements | Typical features required or supported |
|---|---|
| *Application area* | *Target of computer* |
| Personal mobile device | Real-time performance for a range of tasks, including interactive performance for graphics, video, and audio; energy efficiency (Ch. 2, 3, 4, 5; App. A) |
| General-purpose desktop | Balanced performance for a range of tasks, including interactive performance for graphics, video, and audio (Ch. 2, 3, 4, 5; App. A) |
| Servers | Support for databases and transaction processing; enhancements for reliability and availability; support for scalability (Ch. 2, 5; App. A, D, F) |
| Clusters/warehouse-scale computers | Throughput performance for many independent tasks; error correction for memory; energy proportionality (Ch 2, 6; App. F) |
| Embedded computing | Often requires special support for graphics or video (or other application-specific extension); power limitations and power control may be required; real-time constraints (Ch. 2, 3, 5; App. A, E) |

# Requirements in Designing a New Computer (cont'd)

| | |
|---|---|
| *Level of software compatibility* | *Determines amount of existing software for computer* |
| At programming language | Most flexible for designer; need new compiler (Ch. 3, 5; App. A) |
| Object code or binary compatible | Instruction set architecture is completely defined—little flexibility—but no investment needed in software or porting programs (App. A) |
| *Operating system requirements* | *Necessary features to support chosen OS (Ch. 2; App. B)* |
| Size of address space | Very important feature (Ch. 2); may limit applications |
| Memory management | Required for modern OS; may be paged or segmented (Ch. 2) |
| Protection | Different OS and application needs: page vs. segment; virtual machines (Ch. 2) |
| *Standards* | *Certain standards may be required by marketplace* |
| Floating point | Format and arithmetic: IEEE 754 standard (App. J), special arithmetic for graphics or signal processing |
| I/O interfaces | For I/O devices: Serial ATA, Serial Attached SCSI, PCI Express (App. D, F) |
| Operating systems | UNIX, Windows, Linux, CISCO IOS |
| Networks | Support required for different networks: Ethernet, Infiniband (App. F) |
| Programming languages | Languages (ANSI C, C++, Java, Fortran) affect instruction set (App. A) |

**Figure 1.7 Summary of some of the most important functional requirements an architect faces.** The left-hand column describes the class of requirement, while the right-hand column gives specific examples. The right-hand column also contains references to chapters and appendices that deal with the specific issues.

**Market (applications), and technology decide the requirements.**

# Trends in Technology

- If an ISA is to be successful, it must be designed to survive rapid changes in computer technology
  - IBM mainframe core has been in use for 50 years
- Rapid changes in implementation technology
  - Integrated circuit logic technology
    - Tr.count/year 40-55%: Tr density:35%/year, Die size: 10-20%/year
  - Semiconductor DRAM
    - Capacity:25%～40%/year, doubling roughly every two to three years
  - Semiconductor Flash: nonvolatile semiconductor memory
    - 50% ~60% per year, doubling roughly every two years, 15x to 20x cheaper per bit than DRAM
  - Magnetic disk technology
    - 30%/year(~1990), 60%/year (1990~2000, 100%/year 1996), 40%/year (2003~), 15x～20x cheaper per bit than Flash, 300x-500x than DRAM
  - Network technology
    - Depends both on the performance of switches and on the performance of the transmission system

**Although technology improves continuously, the impact of these improvements can be in discrete leaps.**

# Performance Milestones over 20 to 25 Years Microprocessor and Memory Module

| Microprocessor | 16-bit address/ bus, microcoded | 32-bit address/ bus, microcoded | 5-stage pipeline, on-chip I & D caches, FPU | 2-way superscalar, 64-bit bus | Out-of-order 3-way superscalar | Out-of-order superpipelined, on-chip L2 cache | Multicore OOO 4-way on chip L3 cache, Turbo |
|---|---|---|---|---|---|---|---|
| Product | Intel 80286 | Intel 80386 | Intel 80486 | Intel Pentium | Intel Pentium Pro | Intel Pentium 4 | Intel Core i7 |
| Year | 1982 | 1985 | 1989 | 1993 | 1997 | 2001 | 2010 |
| Die size ($mm^2$) | 47 | 43 | 81 | 90 | 308 | 217 | 240 |
| Transistors | 134,000 | 275,000 | 1,200,000 | 3,100,000 | 5,500,000 | 42,000,000 | 1,170,000,000 |
| Processors/chip | 1 | 1 | 1 | 1 | 1 | 1 | 4 |
| Pins | 68 | 132 | 168 | 273 | 387 | 423 | 1366 |
| Latency (clocks) | 6 | 5 | 5 | 5 | 10 | 22 | 14 |
| Bus width (bits) | 16 | 32 | 32 | 64 | 64 | 64 | 196 |
| Clock rate (MHz) | 12.5 | 16 | 25 | 66 | 200 | 1500 | 3333 |
| Bandwidth (MIPS) | 2 | 6 | 25 | 132 | 600 | 4500 | 50,000 |
| Latency (ns) | 320 | 313 | 200 | 76 | 50 | 15 | 4 |
| **Memory module** | **DRAM** | **Page mode DRAM** | **Fast page mode DRAM** | **Fast page mode DRAM** | **Synchronous DRAM** | **Double data rate SDRAM** | **DDR3 SDRAM** |
| Module width (bits) | 16 | 16 | 32 | 64 | 64 | 64 | 64 |
| Year | 1980 | 1983 | 1986 | 1993 | 1997 | 2000 | 2010 |
| Mbits/DRAM chip | 0.06 | 0.25 | 1 | 16 | 64 | 256 | 2048 |
| Die size ($mm^2$) | 35 | 45 | 70 | 130 | 170 | 204 | 50 |
| Pins/DRAM chip | 16 | 16 | 18 | 20 | 54 | 66 | 134 |
| Bandwidth (MBytes/s) | 13 | 40 | 160 | 267 | 640 | 1600 | 16,000 |
| Latency (ns) | 225 | 170 | 125 | 75 | 62 | 52 | 37 |

# Performance Milestones over 20 to 25 Years LAN and Disk

| Local area network | Ethernet | Fast Ethernet | Gigabit Ethernet | 10 Gigabit Ethernet | 100 Gigabit Ethernet | |
|---|---|---|---|---|---|---|
| IEEE standard | 802.3 | 803.3u | 802.3ab | 802.3ac | 802.3ba | |
| Year | 1978 | 1995 | 1999 | 2003 | 2010 | |
| Bandwidth (Mbits/sec) | 10 | 100 | 1000 | 10,000 | 100,000 | |
| Latency (µsec) | 3000 | 500 | 340 | 190 | 100 | |
| Hard disk | 3600 RPM | 5400 RPM | 7200 RPM | 10,000 RPM | 15,000 RPM | 15,000 RPM |
| Product | CDC WrenI 94145-36 | Seagate ST41600 | Seagate ST15150 | Seagate ST39102 | Seagate ST373453 | Seagate ST3600057 |
| Year | 1983 | 1990 | 1994 | 1998 | 2003 | 2010 |
| Capacity (GB) | 0.03 | 1.4 | 4.3 | 9.1 | 73.4 | 600 |
| Disk form factor | 5.25 inch | 5.25 inch | 3.5 inch | 3.5 inch | 3.5 inch | 3.5 inch |
| Media diameter | 5.25 inch | 5.25 inch | 3.5 inch | 3.0 inch | 2.5 inch | 2.5 inch |
| Interface | ST-412 | SCSI | SCSI | SCSI | SCSI | SAS |
| Bandwidth (MBytes/s) | 0.6 | 4 | 9 | 24 | 86 | 204 |
| Latency (ms) | 48.3 | 17.1 | 12.7 | 8.8 | 5.7 | 3.6 |

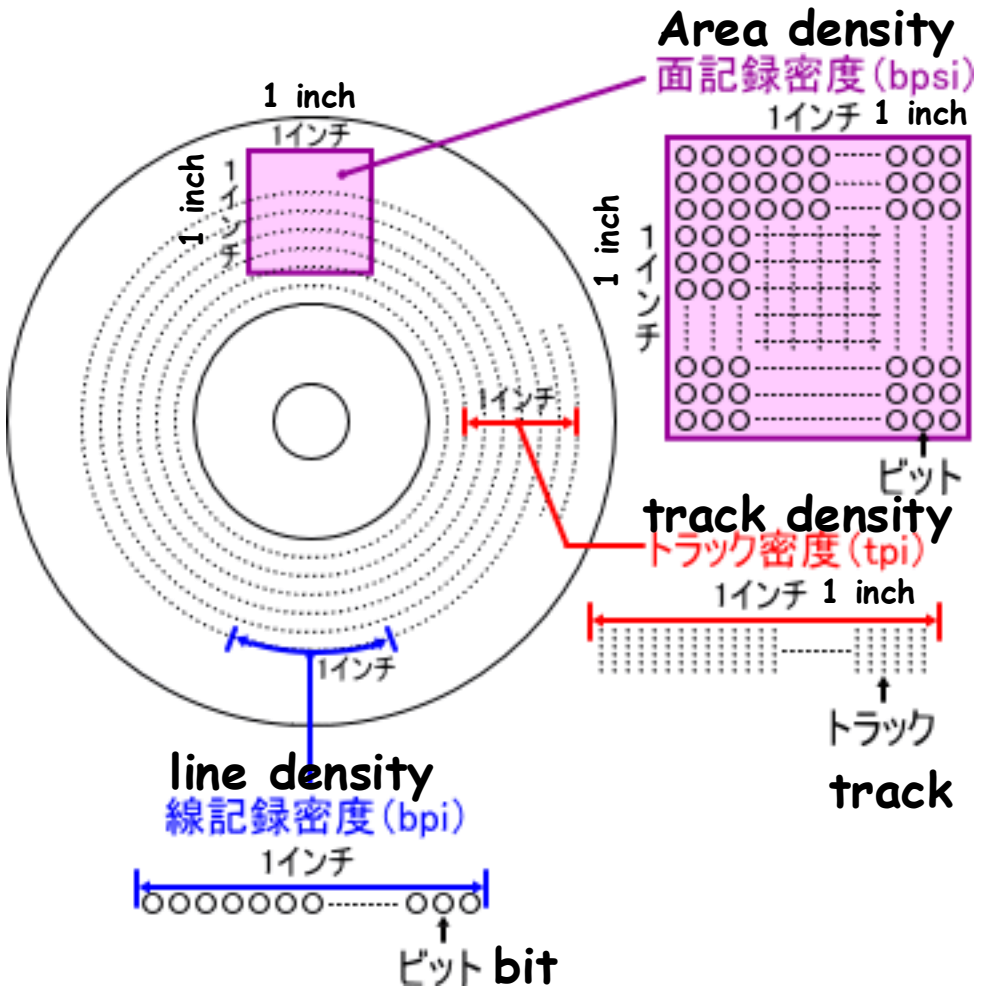
# Tracking Technology Performance Trends

- Drill down into 4 technologies:
  - Disks,
  - Memory,
  - Network,
  - Processors
- Compare ~1980 Archaic (Nostalgic) vs. ~2010 Modern (Newfangled)
  - Performance Milestones in each technology
- Compare for Bandwidth vs. Latency improvements in performance over time
- Bandwidth,Throughput: number of events per unit time
  - E.g., MIPS (million instructions/second), M bits / second over network, M bytes / second from disk
- Latency: elapsed time for a single event
  - E.g., one-way network delay in microseconds, average disk access time in milliseconds

# Disks: Archaic(Nostalgic) v. Modern(Newfangled)

- CDC Wren I, 1983
- 3600 RPM
- 0.03 GBytes capacity
- Tracks/Inch: 800
- Bits/Inch: 9550
- Three 5.25” platters

- Bandwidth:
  0.6 MBytes/sec
- Latency: 48.3 ms
- Cache: none



# Disks: Archaic(Nostalgic) v. Modern(Newfangled)

- CDC Wren I, 1983
- 3600 RPM
- 0.03 GBytes capacity
- 5.25” platters
- Bandwidth: 0.6 MBytes/sec
- Latency: 48.3 ms
- Cache: none

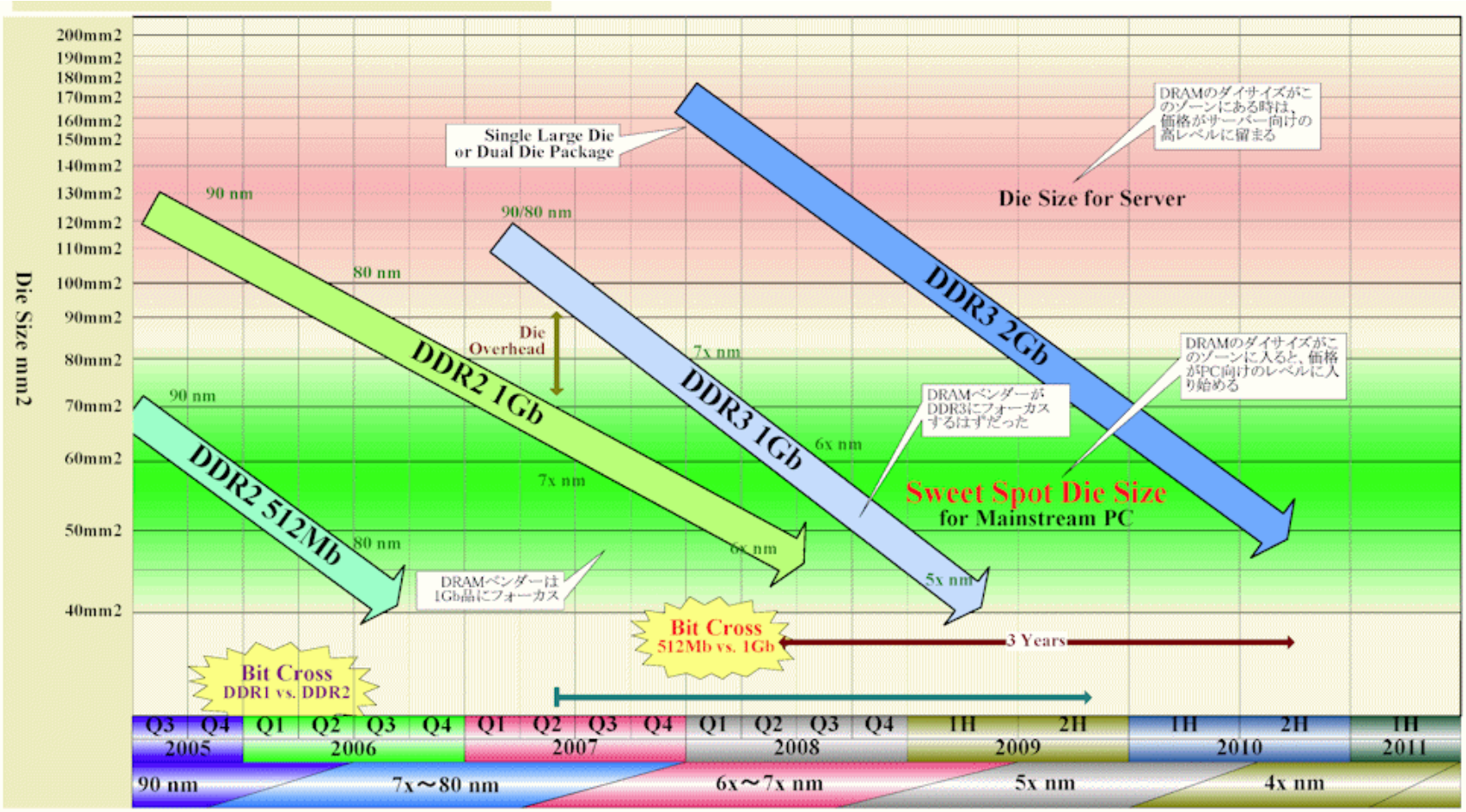
- Seagate 3600057, 2010
- 15,000 RPM (4X)
- 600 GBytes (20,000X)
- 2.5” platters (in 3.5” form factor)
- Bandwidth: 204 MBytes/sec (340X)
- Latency: 3.6 ms (13X)
- Cache: 8 MBytes

# Memory: Archaic (Nostalgic) v. Modern (Newfangled)

- 1980 DRAM (asynchronous)
- 0.06 Mbits/chip
- Die size 35 $mm^2$
- 16 pins/chip
- 13 Mbytes/sec
- Latency: 225 ns

- 2010 DDR3 SDRAM
- 2048 Mbits/chip (34,000X)
- Die size 50$mm^2$ (1.6X)
- 134 pins/chip (8X)
- 16,000 Mbytes/sec (1230X)
- Latency: 37 ns (6X)

# DRAM Process and Die Size

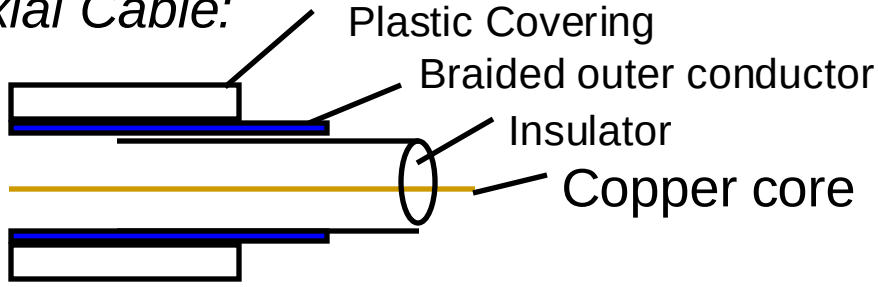



Source by http://pc.watch.impress.co.jp/docs/2009/0121/kaigai_01b.gif

# LANs: Archaic (Nostalgic)v. Modern (Newfangled)

- Ethernet 802.3
- Year of Standard: 1978
- 10 Mbits/s link speed
- Latency: 3000 μsec
- Shared media
- Coaxial cable

*Coaxial Cable:*




- 100G Ethernet 802.3ba
- Year of Standard: 2010
- 100,000 Mbits/s (10,000X) link speed
- Latency: 100 μsec (30X)
- Switched media
- Category 5 copper wire

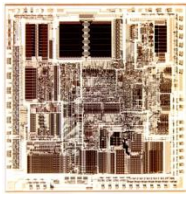
"Cat 5" is 4 twisted pairs in bundle

***Twisted Pair:***

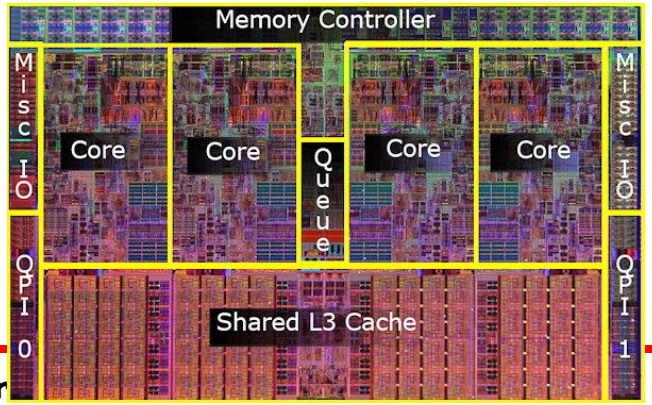
Copper, 1mm thick, twisted to avoid antenna effect

# CPUs: Archaic (Nostalgic) v. Modern (Newfangled)

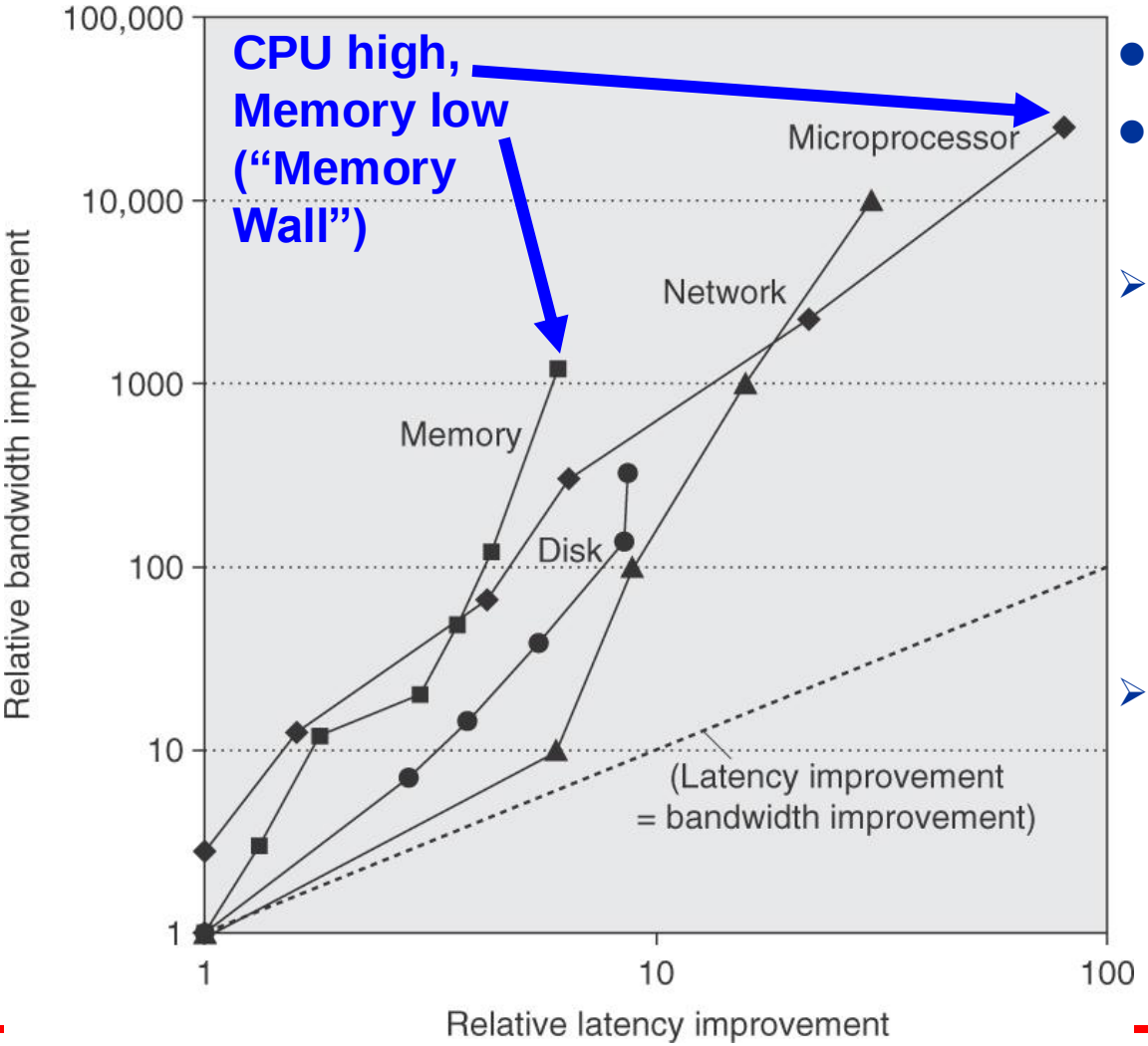
- 1982 Intel 80286
- 12.5 MHz
- 2 MIPS (peak)
- Latency 320 ns
- 134,000 Transistors, 47 $mm^2$
- 16-bit data bus, 68 pins
- Microcode interpreter, separate FPU chip
- (no caches)



- 2010 Intel CoreI7
- 3333MHz (267X)
- 50,000 MIPS (peak) (25,000X)
- Latency 4 ns (80X)
- 1,170,000,000 Transistors, 240 $mm^2$
- 196-bit data bus, 1366pins
- 4 cores, Dynamic translate to RISC, Out-of-Order execution
- On-chip 32KB Data caches, 256KB L2 cache, 8M L3 Share cache



# Latency Lags Bandwidth




- **Latency 6X～80X**
- **Bandwidth 300X～25,000X**

- **In the time that bandwidth doubles, latency improves by no more than a factor of 1.2 to 1.4**
  - ✓ **and capacity improves faster than bandwidth**
- **Stated alternatively: Bandwidth improves by more than the square of the improvement in Latency**

# Scaling of Transistor Performance and Wires

- Integrated circuit processes are characterized by feature size
  - the minimum size of a transistor or a wire in either the x or y dimension
  - 10 microns in 1971 to 0.014 microns (14nano) in 2014 , 7 nano in 2019
- The density of transistors increases quadratically with a linear decrease in feature size.
- Transistor performance improves linearly with decreasing feature size
- But wire delay scales poorly compared to transistor performance
  - the signal delay for a wire increases in proportion to the product of its resistance and capacitance
  - wire also get shorter, but the resistance and capacity per unit length get worse,
  - Larger and larger fractions of the clock cycle have been consumed by the propagation delay of signals on wires.
    - 2 stages of the Pentium 4 20+-stage pipeline just for signal propagation

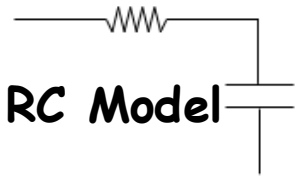
# Trends in Power and Energy in Integrated Circuits

- Power is the biggest challenge as devices are scaled
  - Power must be brought in and distributed around the chip, and modern microprocessors use hundreds of pins and multiple interconnect layers just for power and ground
  - Power is dissipated as heat and must be removed.
- For CMOS chips, traditional dominant energy consumption has been in switching transistors, called *dynamic power*

$$Power_{dynamic} = 1/2 \cdot CapacitiveLoad \cdot Voltage^2 \cdot FrequencySwitched$$

- And energy is given by

$$Energy_{dynamic} = CapacitiveLoad \cdot Voltage^2$$

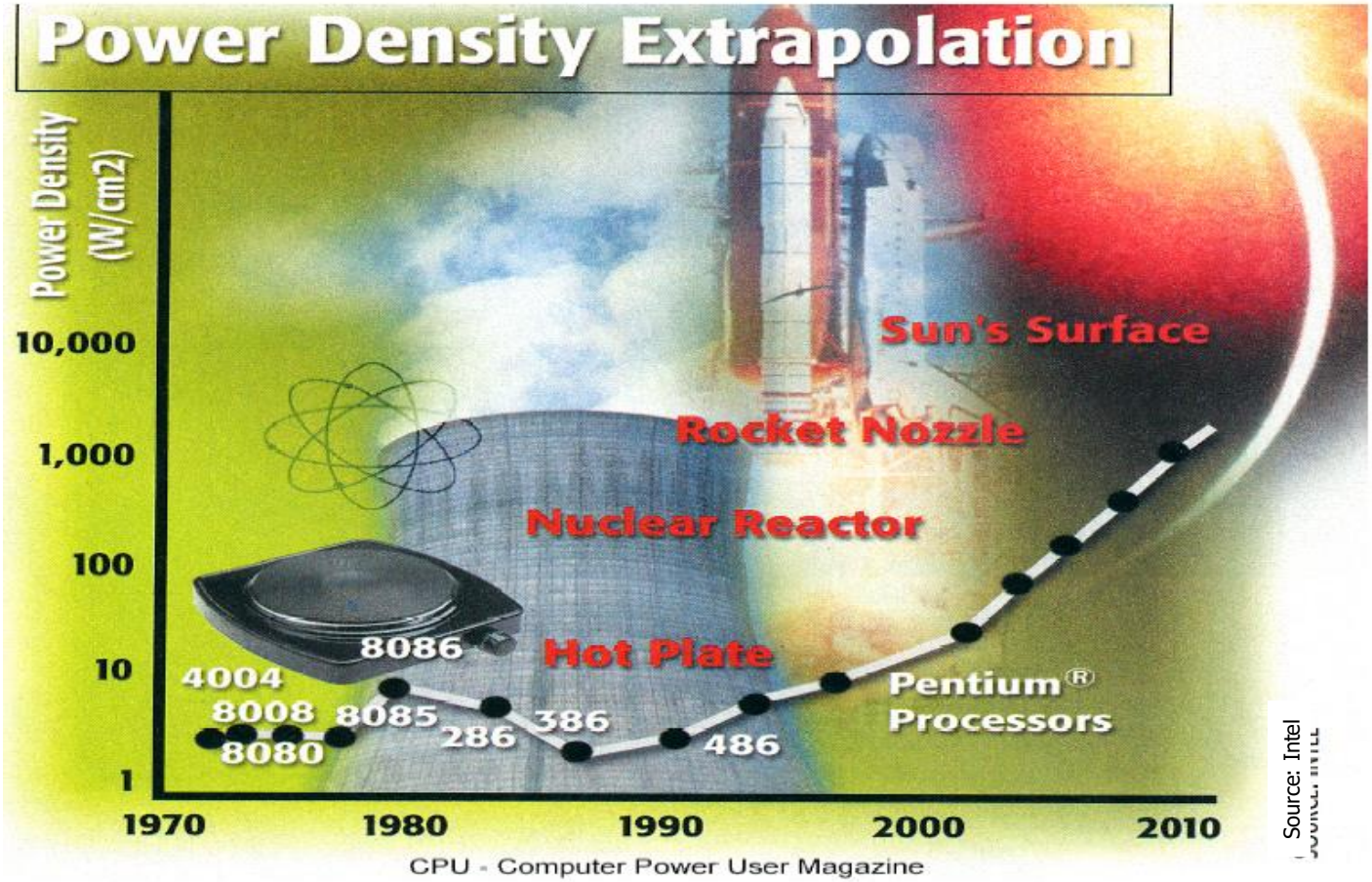



- For a fixed task, slowing clock rate (frequency switched) reduces power, but not energy
- Capacitive load is a function of number of transistors connected to output and technology, which determines capacitance of wires and transistors
- Dropping voltage helps both, so went from 5V to 1V
- To save energy & dynamic power, most CPUs now turn off clock of inactive modules (e.g. Fl. Pt. Unit)

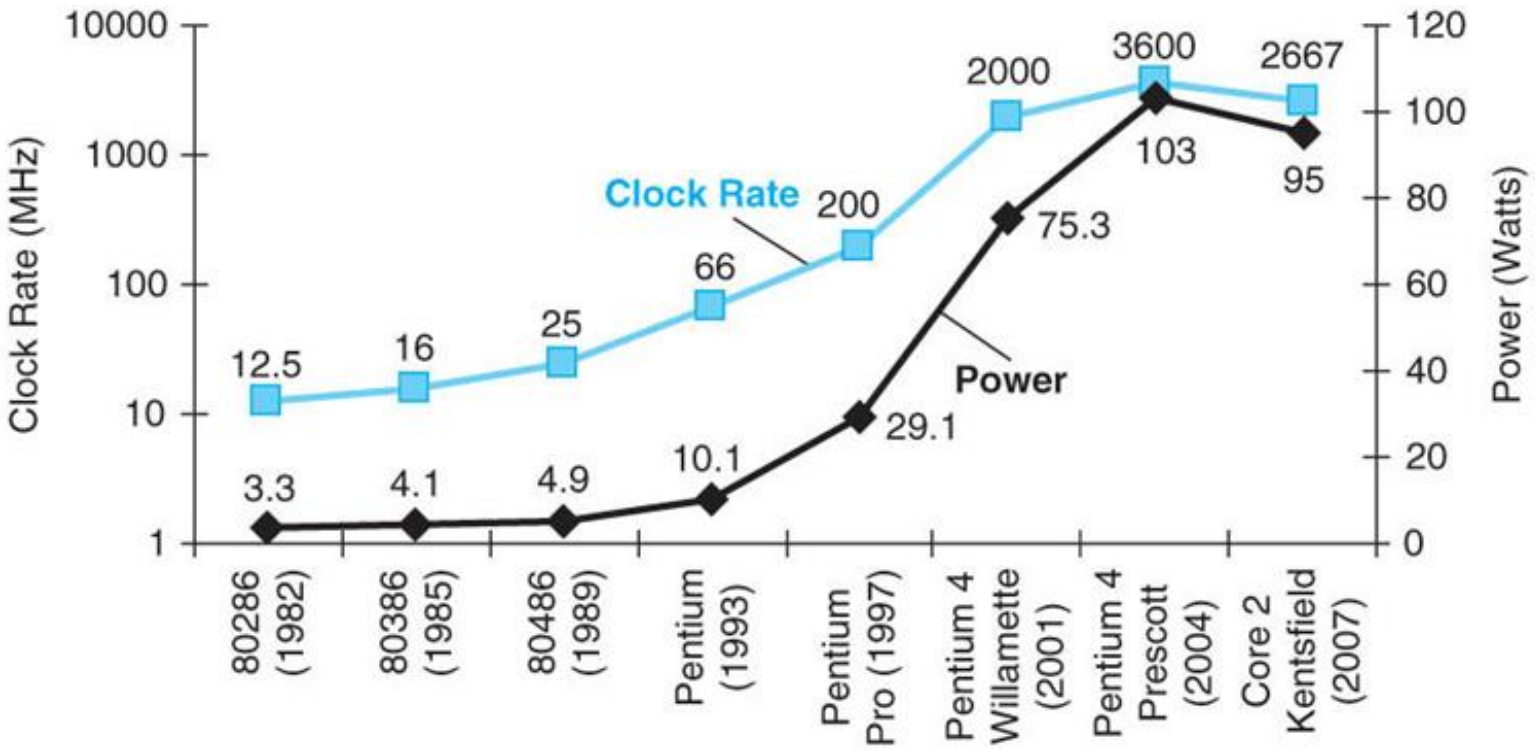
# Power Estimation vs Energy Estimation

- Power: energy per unit time
  - 1 watt = 1 joule per second
- So, which is the right one for comparing processors: energy or power?
  - ➢ Energy is always a better metric because it is tied to a specific task and the time required for that task.
    - ✓ Energy = (average power) × (execution time)
  - Lower energy means lower electricity bill and longer battery lifetime
  - ❖ Power is used as a constraint: A chip might be limited to 100 watt.
- Example:
  - Processor A has a 20% higher power than B, but executes the task in only 70% of that of B.
  - Energy: 0.84 (=1.2x0.7) vs. 1

# Critical Problem in Scalable Processor Design
# High Power and Heat Dissipation



# In Reality…: Switching from Uniprocessors at the Higher Clock to Multiprocessors at the Lower Clock



# Example of Quantifying Power

- Suppose 15% reduction in voltage results in a 15% reduction in frequency. What is impact on dynamic power?

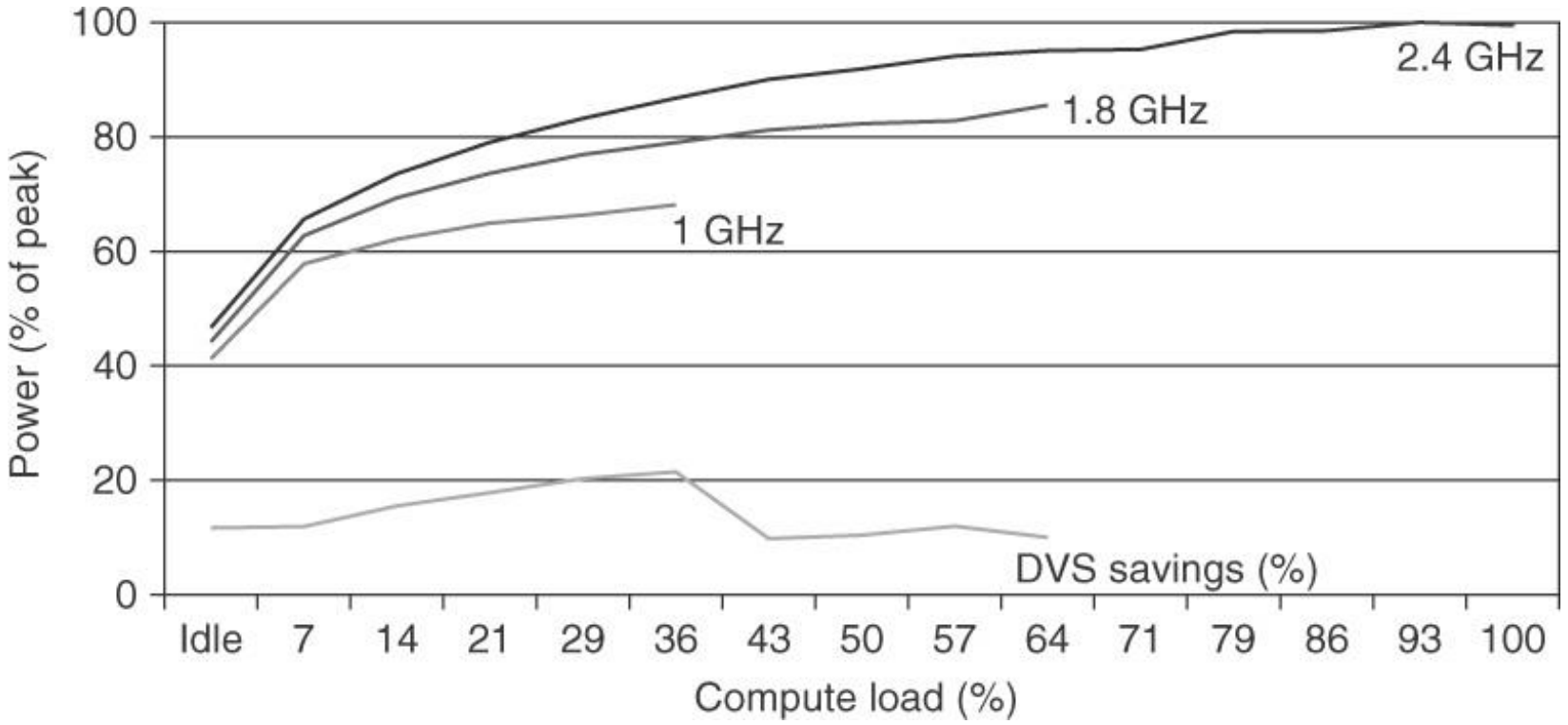
$$
\begin{aligned}
Power_{dynamic} &= 1/2 \cdot CapacitiveLoad \cdot Voltage^2 \cdot FrequencySwitched \\
&= 1/2 \cdot .85 \cdot CapacitiveLoad \cdot (.85 \cdot Voltage)^2 \cdot FrequencySwitched \\
&= (.85)^3 \cdot OldPower_{dynamic} \\
&\approx 0.6 \cdot OldPower_{dynamic}
\end{aligned}
$$

# Techniques for Reducing Power

- Do nothing
  - Most microprocessor today turn off the clock of inactive modules to save energy and dynamic power.
- Dynamic Voltage-Frequency Scaling (DVFS)
  - Dynamically changing operating frequency and voltage according to the amount of the workload.
    - Lowering  voltage and frequency in the periods of low activity
- Design for typical case
  - Mobile devices (smartphones, tablet) and laptop uses low power devices (processors, memory, disks…) to save energy, and server and high-end PC use performance-oriented devices with the "*emergency slowdown*" mechanism.
- Overclocking and turning off cores
  - The chip decides that it is safe to run at a higher clock rate for a short time possibly on just a few core until temperature starts to rise
    - In Intel's Turbo mode, the 3.3 GHz Core i7 can run in short bursts for 3.6GHz.

# Effects of DVFS

**AMD Opteron microprocessor with 8GB DRAM and one ATA Disk**



# Static Power Consumption

- Because leakage current flows even when a transistor is off, now *static power* important too

$$Power_{static} = Current_{static} \cdot Voltage$$

- Leakage current increases in processors with smaller transistor sizes
  - Increasing the number of transistors increases power even if they are idle.
- In 2011, goal for leakage is 25% of total power consumption; high performance designs need more.
- Very low power systems even turn off the power supply *(power gating)* to inactive modules to control loss due to leakage

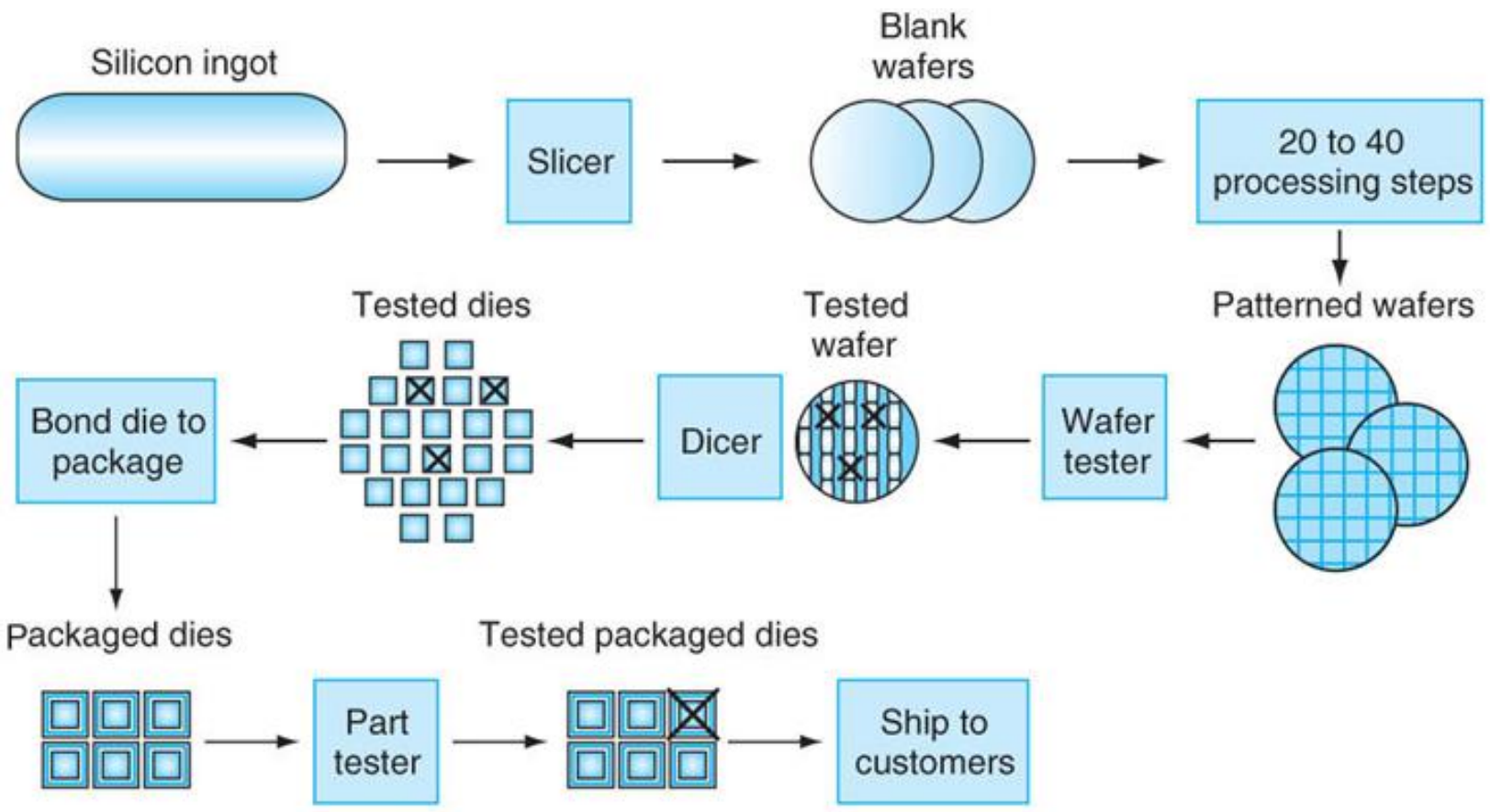
# Trends in Cost

- Cost driven down by learning curve
  - The cost of a manufactured computer component decreases over time even without major improvements in the basic implantation technology.
  - Yield: the percentage of manufactured devices that survives the testing procedure.
    - Doubling the yield will have half the cost in design of chip, board or system

  ✓ DRAM: price closely tracks cost

  ✓ Microprocessors: price depends on volume
    - 10% less for each doubling of volume
    - Not so competitive in the market compared with the DRAM market!

# Chip Manufacturing Process

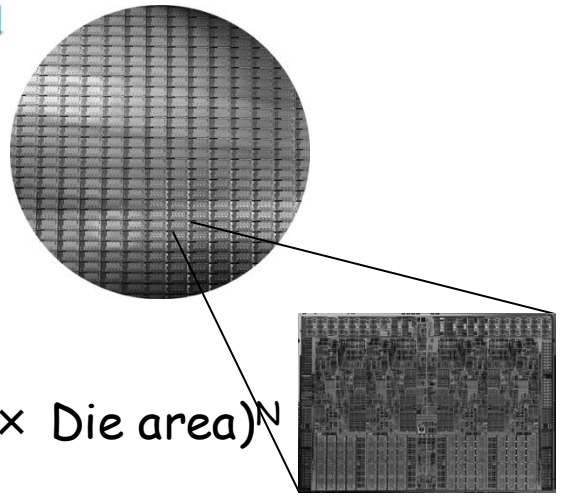


# Cost of an Integrated Circuit

$$\text{Cost of integrated circuit} = \frac{\text{Cost of die} + \text{Cost of testing die} + \text{Cost of packaging and final test}}{\text{Final test yield}}$$

$$\text{Cost of die} = \frac{\text{Cost of wafer}}{\text{Dies per wafer} \times \text{Die yield}}$$

$$\text{Dies per wafer} = \frac{\pi \times (\text{Wafer diameter}/2)^2}{\text{Die area}} - \frac{\pi \times \text{Wafer diameter}}{\sqrt{2 \times \text{Die area}}}$$



$$\text{Die yield} = \text{Wafer yield} \times 1/(1 + \text{Defects per unit area} \times \text{Die area})^N$$

- Die yield: the fraction of good dies on a wafer number
  - Inversely proportional to the complexity of the fabrication process
- Wafer yield accounts for wafers that are completely bad and so need not to be tested (100% wafer yield assumed here)
- Defects per unit area is a measure of the random manufacturing defects that occur.
  - 0.1～0.3 defects per $cm^2$ for 40nm in 2011
- N is a parameter called the process-complexity factor, a measure of manufacturing difficulty. N ranged from 11.5 to 15.5 for 40nm processes in 2010.

# Example of Evaluating Die Yield

1. Find the number of dies per 300mm wafer for a die that is 1.5cm on a side (2.25cm$^2$) and for a die that is 1.0cm on a side (1.00cm$^2$)
2. Find the die yield for dies of the 2.25cm$^2$ chip and the 1.00cm$^2$ chip , assuming a defect density of 0.031 per cm$^2$ and N=13.5.

**2.25cm$^2$ chip**

$$\text{Dies per wafer} = \frac{\pi \times (30/2)^2}{2.25} - \frac{\pi \times 30}{\sqrt{2 \times 2.25}} = \frac{706.9}{2.25} - \frac{94.2}{2.12} = 270$$

$$\text{Die yield} = 1/(1 + 0.031 \times 2.25)^{13.5} = 0.40$$

**1.00cm$^2$ chip**

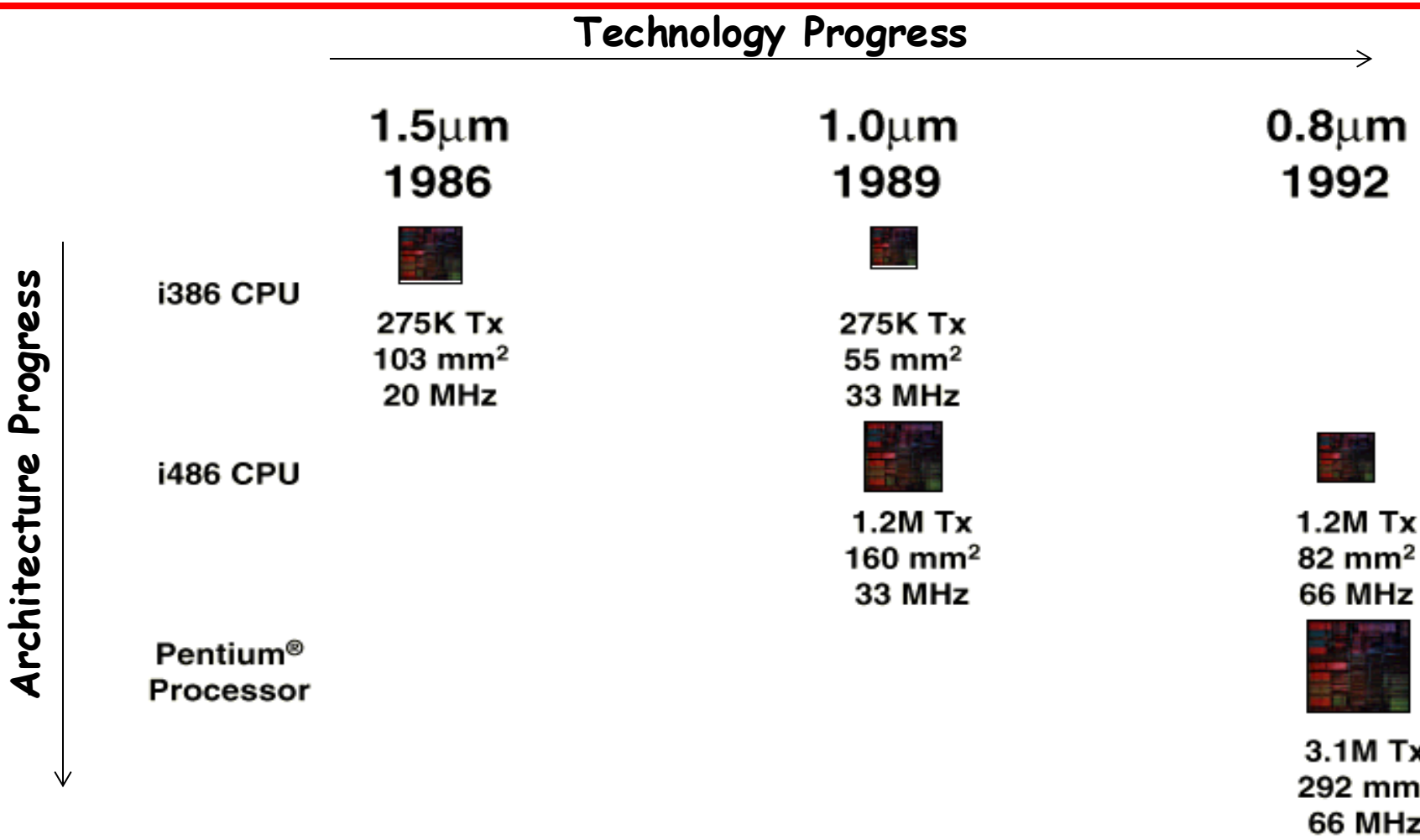
$$\text{Dies per wafer} = \frac{\pi \times (30/2)^2}{1.00} - \frac{\pi \times 30}{\sqrt{2 \times 1.00}} = \frac{706.9}{1.00} - \frac{94.2}{1.41} = 640$$

$$\text{Die yield} = 1/(1 + 0.031 \times 1.00)^{13.5} = 0.66$$

Less than half of all the large die are good but more than two-thirds of the small die are good.

108 good 2.25 cm$^2$ dies and 422 good 1.00 cm2 dies from the 300mm wafer

# Processor Generation

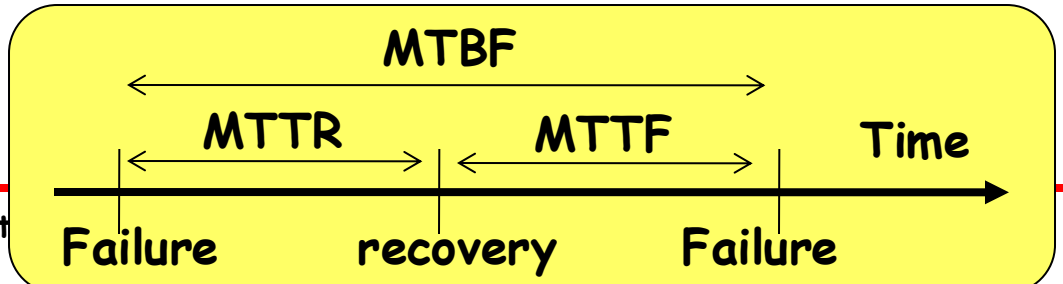


# Define and Quantity Dependability (1/2)

- How decide when a system is operating properly?
- Infrastructure providers now offer Service Level Agreements (SLA) to guarantee that their networking or power service would be dependable
- Systems alternate between 2 states of service with respect to an SLA:
  1. Service accomplishment, where the service is delivered as specified in SLA
  2. Service interruption, where the delivered service is different from the SLA
- Failure = transition from state 1 to state 2
- Restoration = transition from state 2 to state 1

# Define and Quantity Dependability (2/2)

- *Module reliability* = measure of continuous service accomplishment (or time to failure).
  2 metrics
  1. *Mean Time To Failure* (*MTTF*) measures reliability
  2. *Failures In Time* (*FIT*) = 1/MTTF, the rate of failures
     - Traditionally reported as failures per billion hours of operation
- *Mean Time To Repair* (*MTTR*) measures Service Interruption
  - *Mean Time Between Failures* (*MTBF*) = MTTF+MTTR
- *Module availability* measures service as alternation between the 2 states of accomplishment and interruption (number between 0 and 1, e.g. 0.9)
- *Module availability = MTTF / ( MTTF + MTTR)*



# Example Calculating Reliability

- If modules have *exponentially distributed lifetimes* (age of module does not affect probability of failure), overall failure rate is the sum of failure rates of the modules
- Calculate FIT and MTTF of a disk subsystem with 10 disks (1M-hour MTTF per disk), 1 ATA disk controller (0.5M-hour MTTF), 1 power supply (0.2M-hour MTTF), 1 fan, 0.2M-hour MTTF, and 1ATA cable (1M-hour MTTF):

The sum of the failure rates is

$$\text{Failure rate}_{\text{system}} = 10 \times \frac{1}{1{,}000{,}000} + \frac{1}{500{,}000} + \frac{1}{200{,}000} + \frac{1}{200{,}000} + \frac{1}{1{,}000{,}000}$$

$$= \frac{10 + 2 + 5 + 5 + 1}{1{,}000{,}000 \text{ hours}} = \frac{23}{1{,}000{,}000} = \frac{23{,}000}{1{,}000{,}000{,}000 \text{ hours}}$$

or 23,000 FIT. The MTTF for the system is just the inverse of the failure rate:

$$\text{MTTF}_{\text{system}} = \frac{1}{\text{Failure rate}_{\text{system}}} = \frac{1{,}000{,}000{,}000 \text{ hours}}{23{,}000} = 43{,}500 \text{ hours}$$

or just under 5 years.

# Definition: Performance

- Performance is in units of things per sec
  - bigger is better
- If we are primarily concerned with response time

$$\text{performance}(x) = \frac{1}{\text{execution\_time}(x)}$$

" X is n times faster than Y" means

$$n = \frac{\text{Performance}(X)}{\text{Performance}(Y)} = \frac{\text{Execution\_time}(Y)}{\text{Execution\_time}(X)}$$

# Performance: What to Measure

- Usually rely on benchmarks vs. real workloads
- To increase predictability, collections of benchmark applications, called *benchmark suites*, are popular
- SPECCPU: popular desktop benchmark suite
  - CPU only, split between integer and floating point programs
  - SPECint2000 has 12 integer prgms, SPECfp2000 has 14 floating point pgms
  - SPECCPU2006 12 integer prgs (CINT2006) & 17 floating point prgs (CFP2006)
    - http://www.spec.org/cpu2006/
  - SPECSFS (NFS file server) and SPECWeb (WebServer) added as server benchmarks
- Transaction Processing Council measures server performance and cost-performance for databases
  - TPC-C Complex query for Online Transaction Processing
  - TPC-H models ad hoc decision support
  - TPC-W  a transactional web benchmark
  - TPC-App application server and web services benchmark

## How Summarize Suite Performance (1/3)

- Arithmetic average of execution time of all pgms?
  - But they vary by 4X in speed, so some would be more important than others in arithmetic average
- Could add a weights per program, but how pick weight?
  - Different companies want different weights for their products
- SPECRatio: Normalize execution times to reference computer, yielding a ratio proportional to performance

$$= \frac{\text{time on reference computer}}{\text{time on computer being rated}}$$

## How Summarize Suite Performance (2/3)

- If program SPECRatio on Computer A is 1.25 times bigger than Computer B, then

$$1.25 = \frac{SPECRatio_A}{SPECRatio_B} = \frac{\frac{ExecutionTime_{reference}}{ExecutionTime_A}}{\frac{ExecutionTime_{reference}}{ExecutionTime_B}}$$

$$= \frac{ExecutionTime_B}{ExecutionTime_A} = \frac{Performance_A}{Performance_B}$$

  - Note that when comparing 2 computers as a ratio, execution times on the reference computer drop out, so choice of reference computer is irrelevant

## How Summarize Suite Performance (3/3)

- Since ratios, proper mean is geometric mean (SPECRatio unitless, so arithmetic mean meaningless)

$$GeometricMean = \sqrt[n]{\prod_{i=1}^{n} SPECRatio_i}$$

1. Geometric mean of the ratios is the same as the ratio of the geometric means
2. Ratio of geometric means
   = Geometric mean of performance ratios
   $\Rightarrow$ choice of reference computer is irrelevant!

- These two points make geometric mean of ratios attractive to summarize performance

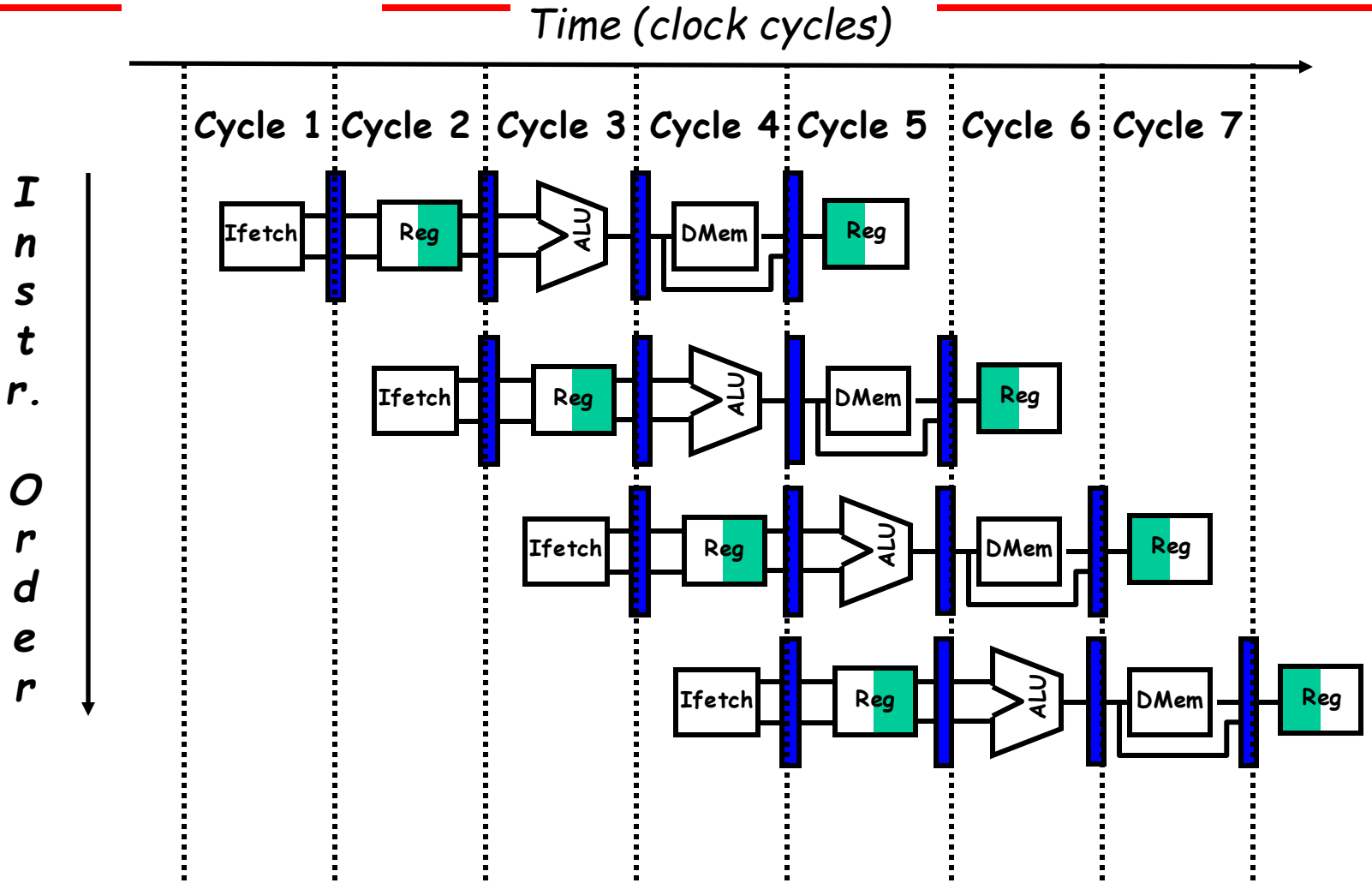
# 5 Quantitative Principles of Computer Design

1. Take Advantage of Parallelism
2. Principle of Locality
3. Focus on the Common Case
4. Amdahl's Law
5. The Processor Performance Equation

# 1) Taking Advantage of Parallelism

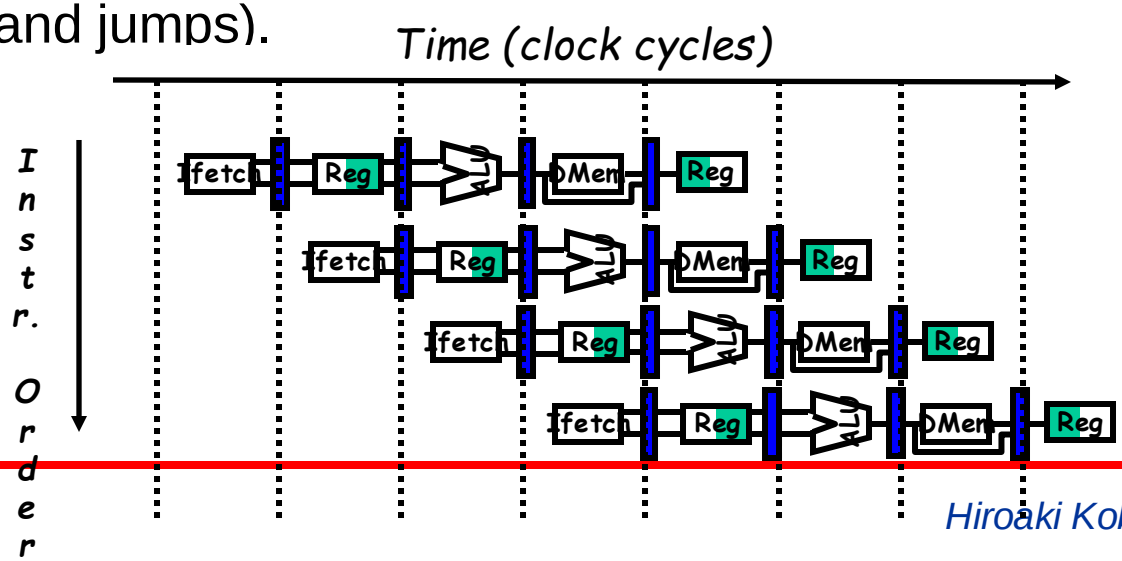
- Increasing throughput of server computer via multiple processors or multiple disks (at the system level)
- At the processor level, Pipelining: overlap instruction execution to reduce the total time to complete an instruction sequence.
  - Not every instruction depends on immediate predecessor ⇒ executing instructions completely/partially in parallel possible
  - Classic 5-stage pipeline:
    1) Instruction Fetch (Ifetch),
    2) Register Read (Reg),
    3) Execute (ALU),
    4) Data Memory Access (Dmem),
    5) Register Write (Reg)
- Detailed HW design level
  - Carry lookahead adders uses parallelism to speed up computing sums from linear to logarithmic in number of bits per operand
  - Multiple memory banks searched in parallel in set-associative caches

# Pipelined Instruction Execution



# Limits to Pipelining

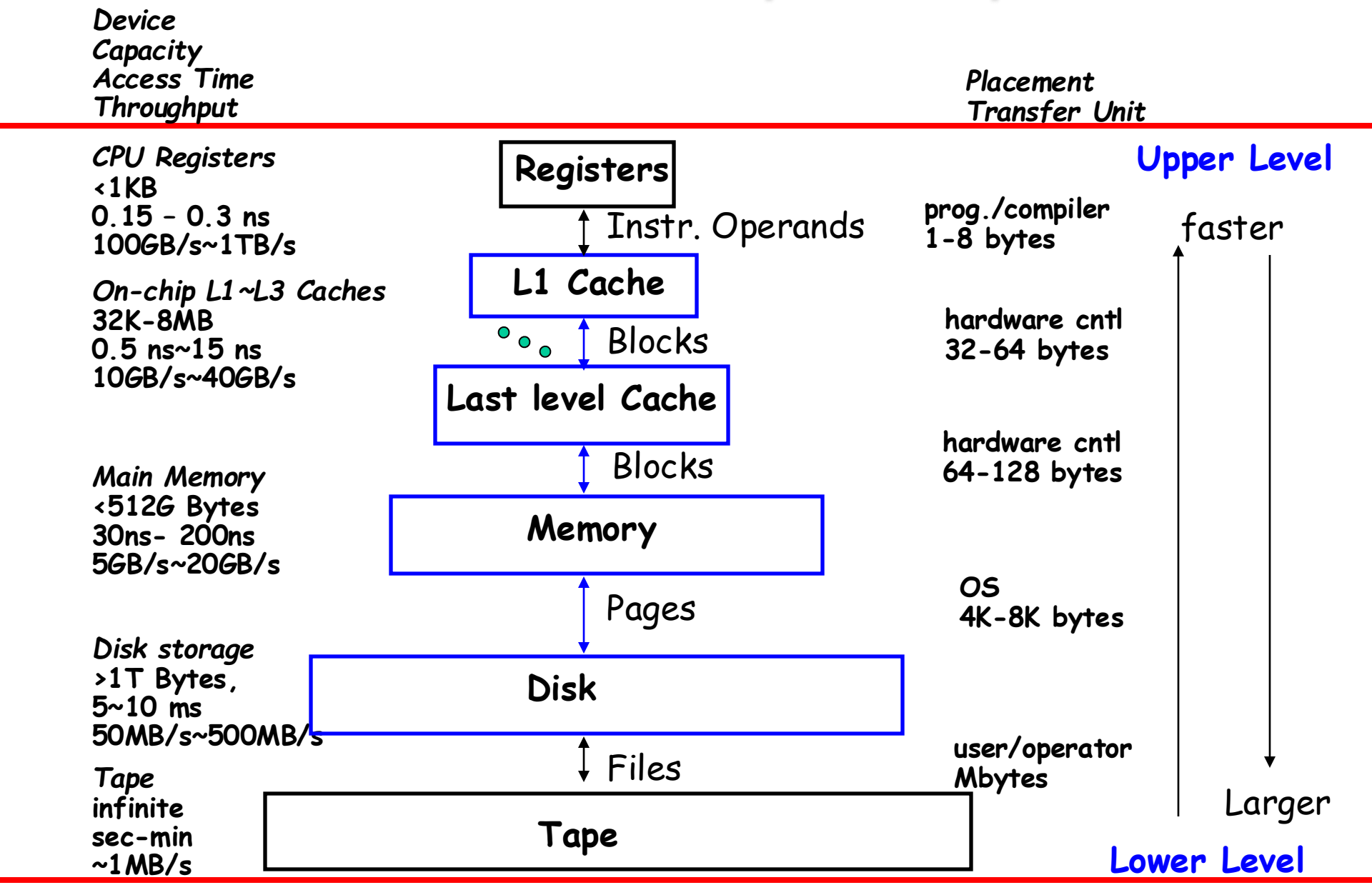
- Hazards prevent next instruction from executing during its designated clock cycle
  - Structural hazards: attempt to use the same hardware to do two different things at once
  - Data hazards: Instruction depends on result of prior instruction still in the pipeline
  - Control hazards: Caused by delay between the fetching of instructions and decisions about changes in control flow (branches and jumps).



# 2) The Principle of Locality

- The Principle of Locality:
  - Program access a relatively small portion of the address space at any instant of time.
- Two Different Types of Locality:
  - Temporal Locality (Locality in Time): If an item is referenced, it will tend to be referenced again soon (e.g., loops, reuse)
  - Spatial Locality (Locality in Space): If an item is referenced, items whose addresses are close by tend to be referenced soon (e.g., straight-line code, array access)

# Levels of the Memory Hierarchy



# 3) Focus on the Common Case

- Common sense guides computer design
  - Since its engineering, common sense is valuable
- In making a design trade-off, favor the frequent case over the infrequent case
  - E.g., Instruction fetch and decode unit used more frequently than multiplier, so optimize it 1st
  - E.g., If database server has 50 disks / processor, storage dependability dominates system dependability, so optimize it 1st
- Frequent case is often simpler and can be done faster than the infrequent case
  - E.g., overflow is rare when adding 2 numbers, so improve performance by optimizing more common case of no overflow
  - May slow down overflow, but overall performance improved by optimizing for the normal case
- What is frequent case and how much performance improved by making case faster => Amdahl's Law

# 4) Amdahl's Law

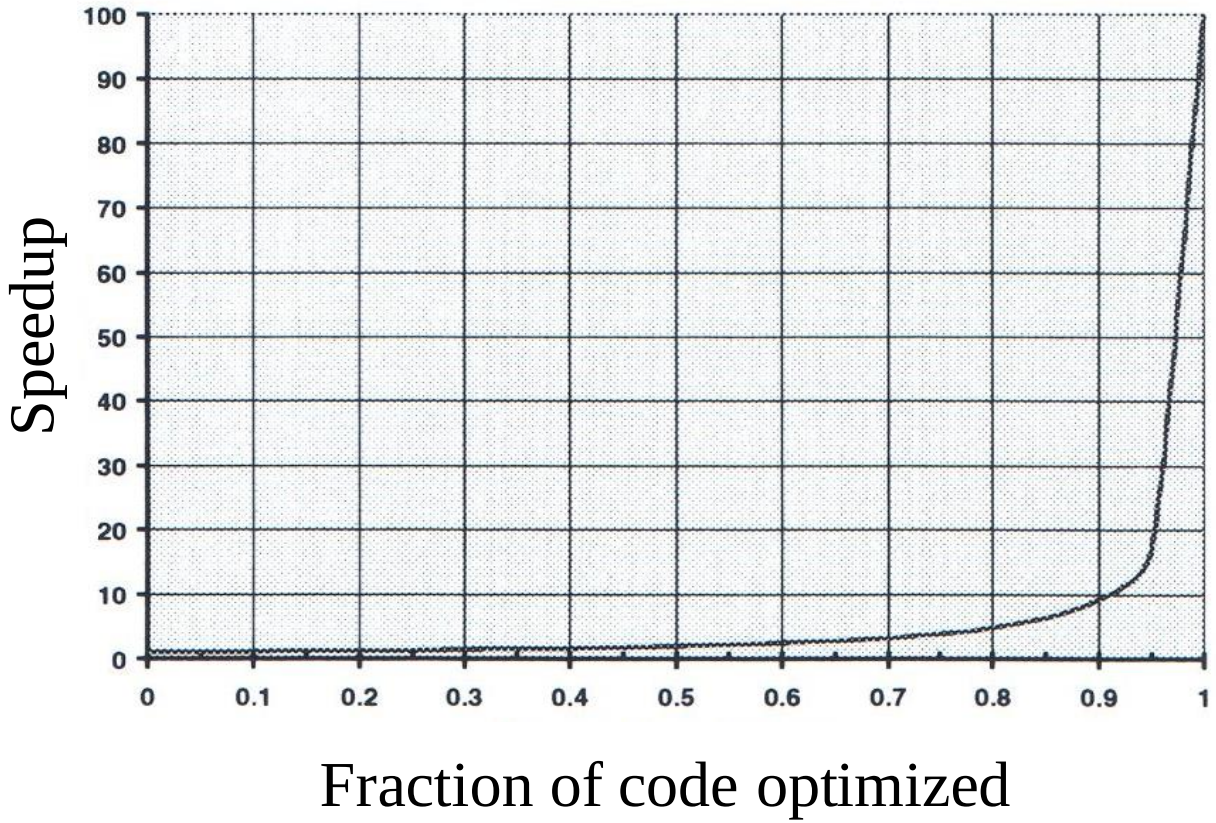
$$ExTime_{new} = ExTime_{old} \times \left[ \left(1 - Fraction_{enhanced}\right) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}} \right]$$

$$Speedup_{overall} = \frac{ExTime_{old}}{ExTime_{new}} = \frac{1}{\left(1 - Fraction_{enhanced}\right) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}}}$$

**Best you could ever hope to do:**

$$Speedup_{maximum} = \frac{1}{\left(1 - Fraction_{enhanced}\right)}$$

# Amdahl's Law:
# Non-Optimized Portion Slows the Speedups!



# Amdahl's Law example

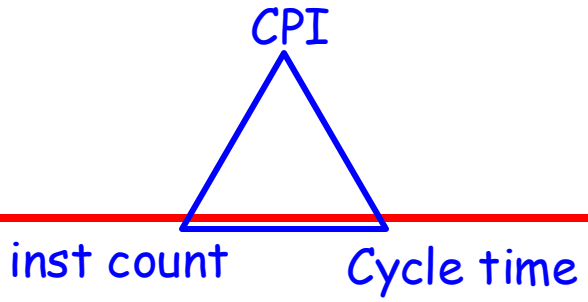
- New CPU 10X faster
- I/O bound server, so 60% time waiting for I/O

$$\text{Speedup}_{\text{overall}} = \frac{1}{(1 - \text{Fraction}_{\text{enhanced}}) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}$$

$$= \frac{1}{(1 - 0.4) + \frac{0.4}{10}} = \frac{1}{0.64} = 1.56$$

- Apparently, its human nature to be attracted by 10X faster, vs. keeping in perspective its just 1.6X faster

# 5) Processor performance equation




$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

| | Inst Count | CPI | Clock Rate |
|---|---|---|---|
| Program | X | | |
| Compiler | X | | |
| Inst. Set. | X | X | |
| Organization | | X | X |
| Technology | | | X |

# Processor Performance Equation

- CPU time =(CPU clock cycles for a program) × Clock cycle time
  - CPU Clock cycles=$\Sigma IC_i \times CPI_i$

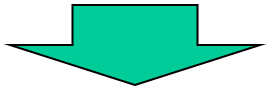
    $ic_i$: instruction count for instruction i, $CPI_i$ : Clocks per instruction i

    $= \Sigma IC_i \times CPI_i \times$ Clock cycle time

  - Overall CPI =$\Sigma IC_i \times CPI_i$/(Instruction count)

    $=\Sigma(IC_i$ /Instruction count) $\times CPI_i$

    = (Instruction Count) × (Overall CPI) × (Clock cycle time)



$$\text{CPU time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Cycle}}$$

# Example of Evaluating Design Altarnatives

- Suppose we have made the following measurements:
  - Frequency of FP operations =25%
  - Average CPI of FP operations=4.0
  - Average CPI of other instructions=1.33
  - Frequency of FPSQR=2%
  - CPI of FPSQR=20
- Assume that the two design alternatives are to decrese the CPI of FPSQR to 2 or to decrease the average CPI of all FP operations to 2.5. Compare these two design alternatives using the processor performance equation

$$\text{CPI}_{\text{original}} = \sum_{i=1}^{n} \text{CPI}_i \times \left( \frac{\text{IC}_i}{\text{Instruction count}} \right)$$
$$= (4 \times 25\%) + (1.33 \times 75\%) = 2.0$$

$$\text{CPI}_{\text{with new FPSQR}} = \text{CPI}_{\text{original}} - 2\% \times (\text{CPI}_{\text{old FPSQR}} - \text{CPI}_{\text{of new FPSQR only}})$$
$$= 2.0 - 2\% \times (20 - 2) = 1.64$$

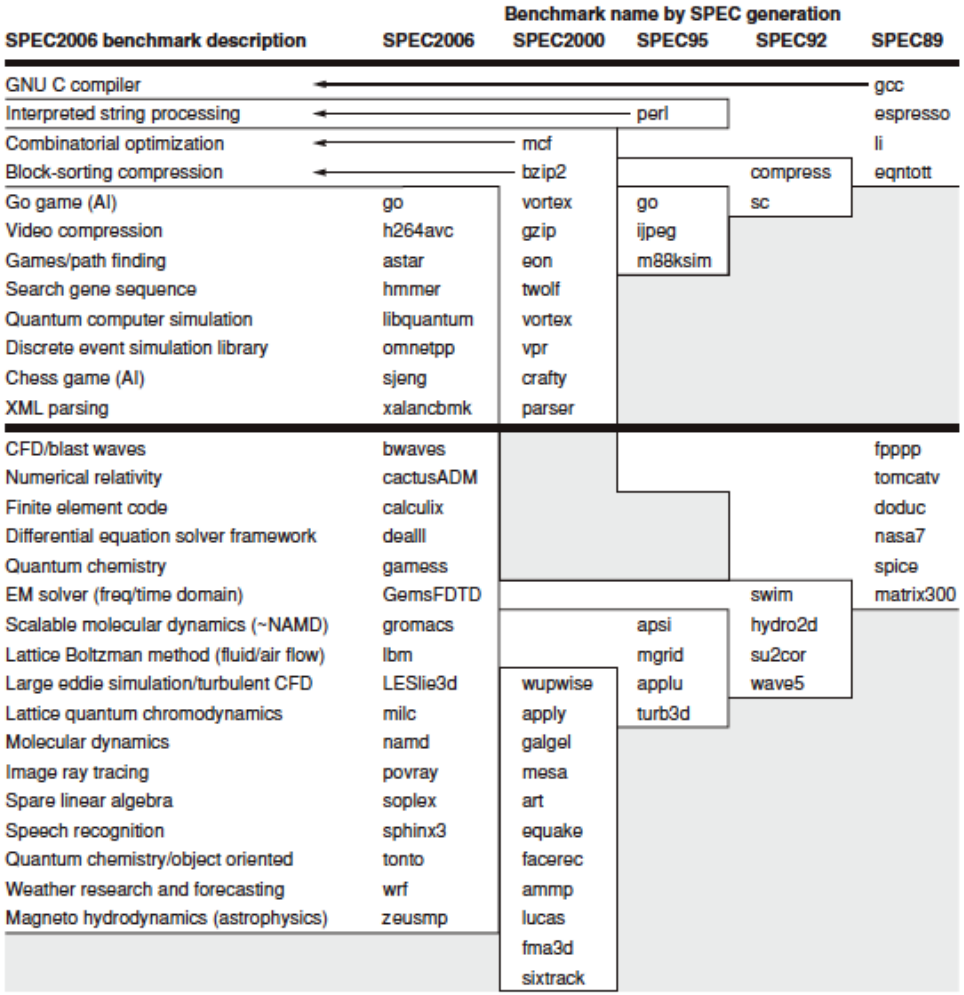
$$\text{CPI}_{\text{new FP}} = (75\% \times 1.33) + (25\% \times 2.5) = 1.62$$

$$\text{Speedup}_{\text{new FP}} = \frac{\text{CPU time}_{\text{original}}}{\text{CPU time}_{\text{new FP}}} = \frac{\text{IC} \times \text{Clock cycle} \times \text{CPI}_{\text{original}}}{\text{IC} \times \text{Clock cycle} \times \text{CPI}_{\text{new FP}}}$$
$$= \frac{\text{CPI}_{\text{original}}}{\text{CPI}_{\text{new FP}}} = \frac{2.00}{1.625} = 1.23$$

# Fallacies and Pitfalls

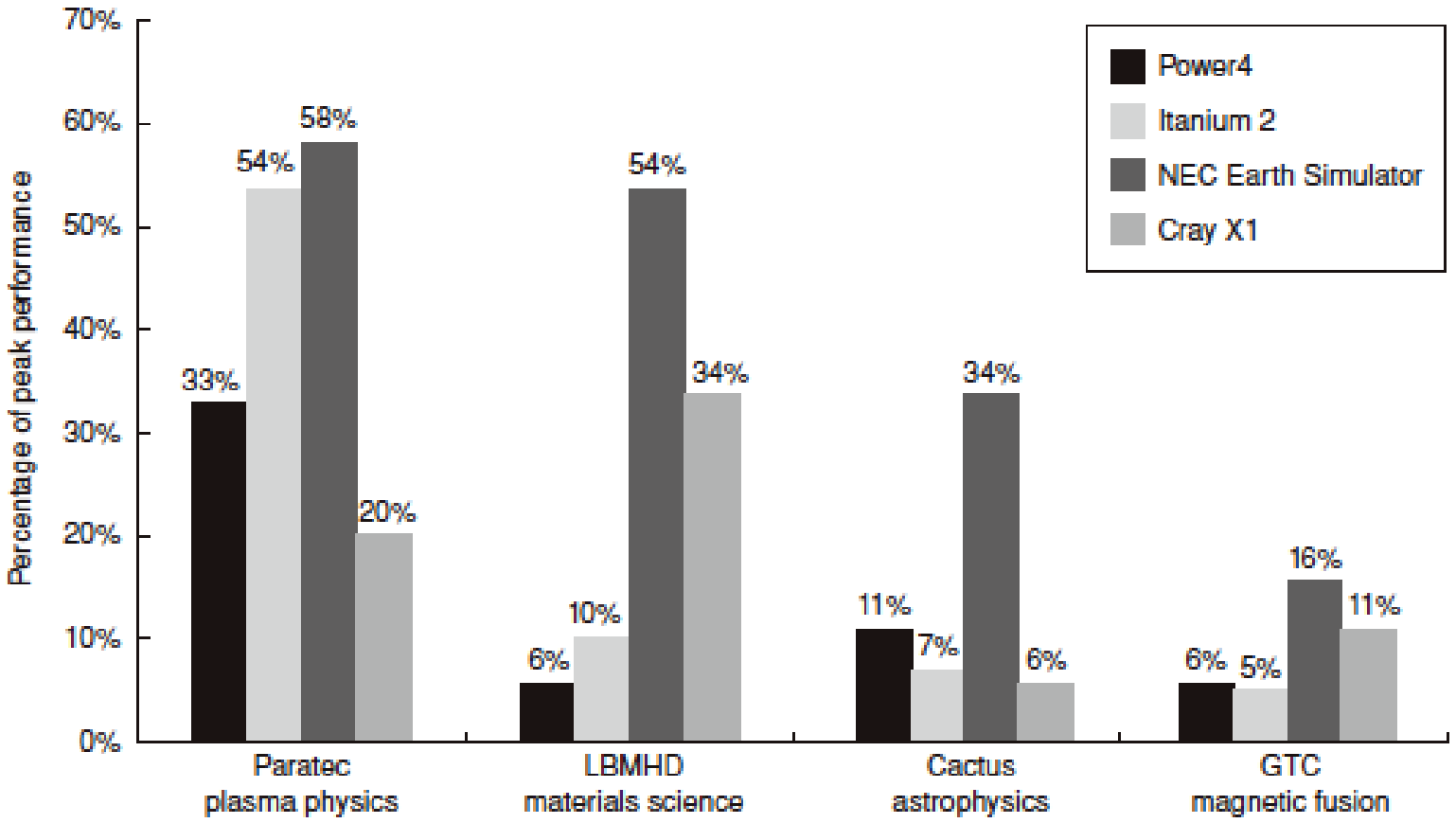
- **Fallacy**: Multiprocessors are a silver bullet
- **Pitfall**: Falling prey to Amdahl's heartbreaking Law
- **Pitfall**: A single point of failure
- **Fallacy**: Hardware enhancements that increase performance improve energy efficiency or are at worst energy neutral.
- **Fallacy**: Benchmarks remain valid indefinitely.
- **Fallacy**: The rated mean time to failure of disks is 1,200,000 hours or almost 140 years, so disks practically never fail.
- **Fallacy**: Peak performance tracks observed performance.
- **Pitfall**: Fault detection can lower availability.

# Evolution of the SPEC Benchmarks over Time

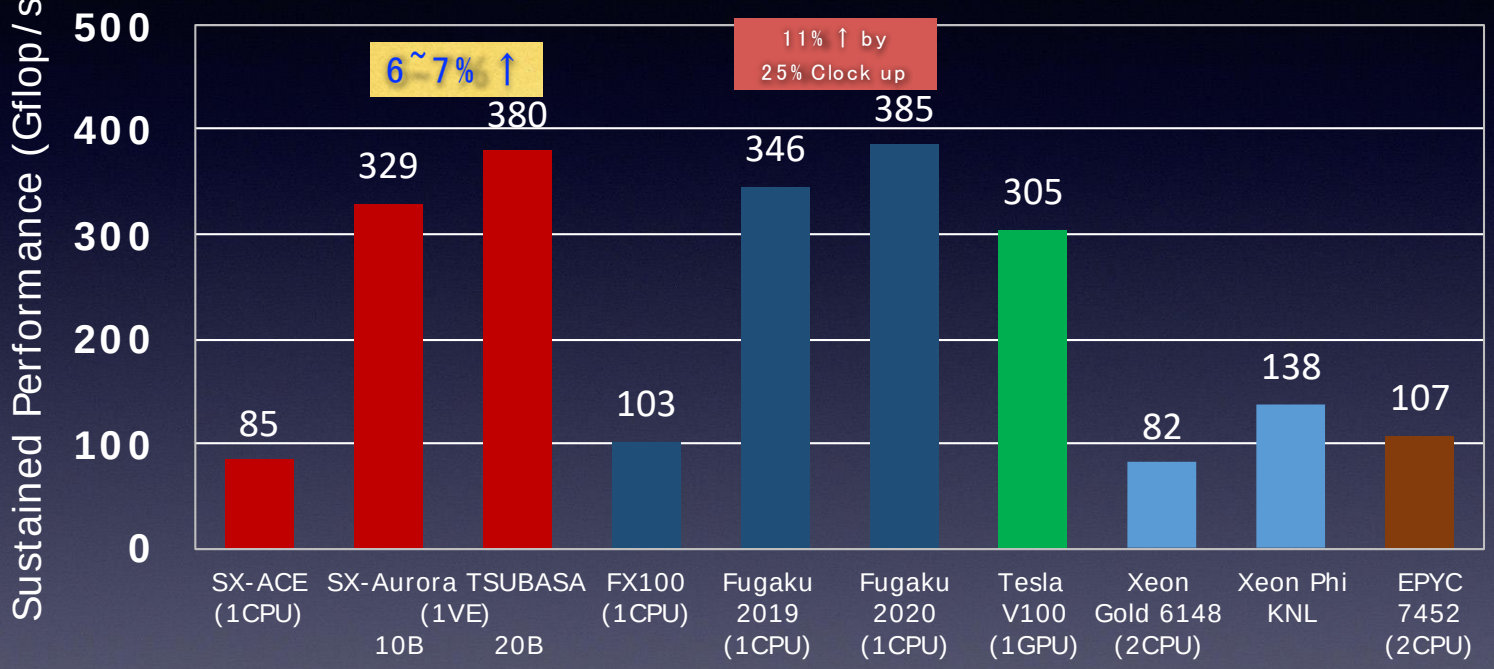


| SPEC2006 benchmark description | Benchmark name by SPEC generation | | | | |
|---|---|---|---|---|---|
| | SPEC2006 | SPEC2000 | SPEC95 | SPEC92 | SPEC89 |
| GNU C compiler | | | | | gcc |
| Interpreted string processing | | | perl | | espresso |
| Combinatorial optimization | | mcf | | | li |
| Block-sorting compression | | bzip2 | | compress | eqntott |
| Go game (AI) | go | vortex | go | sc | |
| Video compression | h264avc | gzip | ijpeg | | |
| Games/path finding | astar | eon | m88ksim | | |
| Search gene sequence | hmmer | twolf | | | |
| Quantum computer simulation | libquantum | vortex | | | |
| Discrete event simulation library | omnetpp | vpr | | | |
| Chess game (AI) | sjeng | crafty | | | |
| XML parsing | xalancbmk | parser | | | |
| CFD/blast waves | bwaves | | | | fpppp |
| Numerical relativity | cactusADM | | | | tomcatv |
| Finite element code | calculix | | | | doduc |
| Differential equation solver framework | dealII | | | | nasa7 |
| Quantum chemistry | gamess | | | | spice |
| EM solver (freq/time domain) | GemsFDTD | | | swim | matrix300 |
| Scalable molecular dynamics (~NAMD) | gromacs | | apsi | hydro2d | |
| Lattice Boltzman method (fluid/air flow) | lbm | | mgrid | su2cor | |
| Large eddie simulation/turbulent CFD | LESlie3d | wupwise | applu | wave5 | |
| Lattice quantum chromodynamics | milc | apply | turb3d | | |
| Molecular dynamics | namd | galgel | | | |
| Image ray tracing | povray | mesa | | | |
| Spare linear algebra | soplex | art | | | |
| Speech recognition | sphinx3 | equake | | | |
| Quantum chemistry/object oriented | tonto | facerec | | | |
| Weather research and forecasting | wrf | ammp | | | |
| Magneto hydrodynamics (astrophysics) | zeusmp | lucas | | | |
| | | fma3d | | | |
| | | sixtrack | | | |

# Percentage of Peak Performance for Four Programs on Four Multiprocessors



# Single-VE Performance of the HIMENO Benchmark (2020 Results)




| | SX-ACE (1CPU) | SX-Aurora TSUBASA (1VE) 10B | SX-Aurora TSUBASA (1VE) 20B | FX100 (1CPU) | Fugaku 2019 (1CPU) | Fugaku 2020 (1CPU) | Tesla V100 (1GPU) | Xeon Gold 6148 (2CPU) | Xeon Phi KNL | EPYC 7452 (2CPU) |
|---|---|---|---|---|---|---|---|---|---|---|
| Peak DP Perf. (TF) | 0.256 | 2.15 | 2.45 | 1.12 | 2.7 | 3.38 | 7 | 3.07 | 3.456 | 2.4 |
| Peak Mem BF (TB/s) | 0.256 | 1.22 | 1.53 | R:0.24 W:0.24 | 1.024 | 1.024 | 0.90 | 0.256 | 0.446* | 0.410 |
| B/F | 1.00 | 0.57 | 0.62 | 0.43 | 0.38 | 0.30 | 0.13 | 0.08 | 0.13 | 0.17 |
| Efficency (%) | 33.2 | 15.3 | 15.5 | 9.2 | 12.8 | 11.4 | 4.4 | 2.7 | 4.0 | 4.5 |

*Stream benchmark result

# Conclusions

- Computer architects must design a computer to meet functional requirements as well as price, power, performance, and availability goals
- Architects must also be aware of important trends in both the technology and the use of computers
- Expect Bandwidth in disks, DRAM, network, and processors to improve by at least as much as the square of the improvement in Latency
- Quantify dynamic and static power
  - Capacitance x Voltage$^2$ x frequency, Energy vs. power
- Quantify and summarize performance
  - Ratios, Geometric Mean, Multiplicative Standard Deviation
- 5 Quantitative Principles of Computer Design