NATIONAL CHENGCHI UNIVERSITY

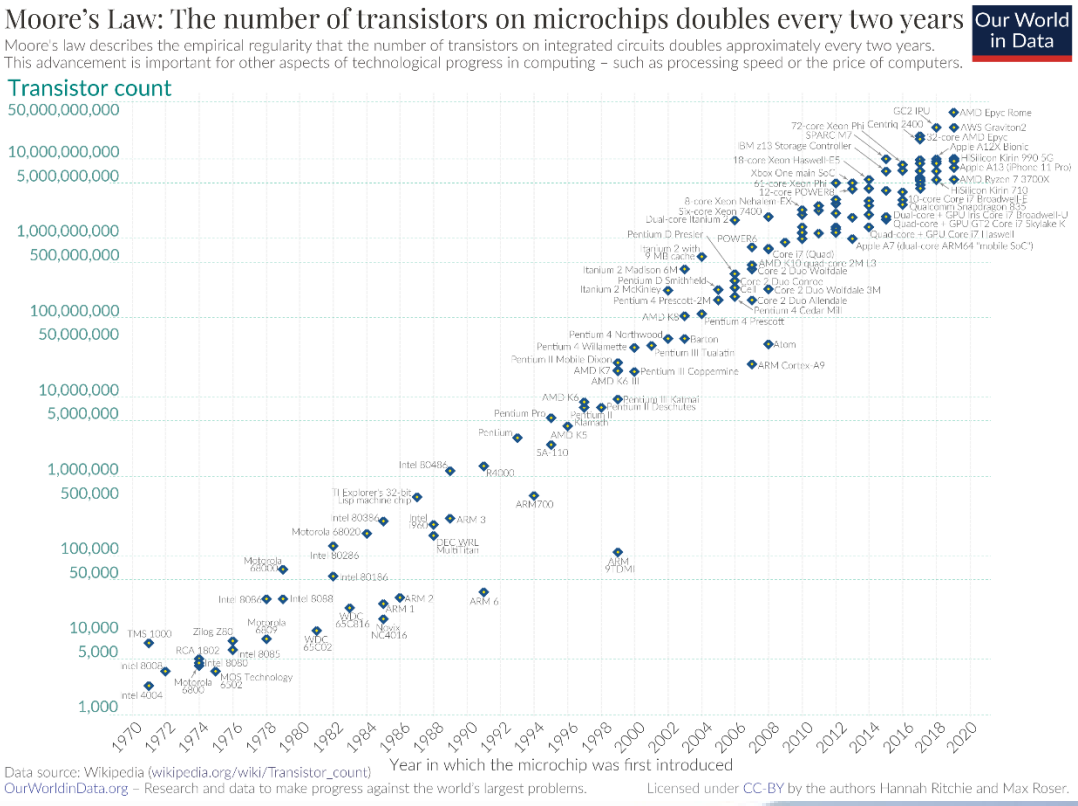
Computer Architecture and Organization

INSTRUCTOR: YAN-TSUNG PENG

DEPT. OF COMPUTER SCIENCE, NCCU

Chapter 1 – Computer Abstractions and Technology

- Moore's Law - Moore's perception
  - Named after Gordon Moore, the co-founder of Intel.
  - For every two years, the number of transistors on a microchip doubles, while the cost of computers reduces by half.
    - 2X transistors/chip every 1.5~2 years
  - In the semiconductor industry, Moore's law has served as a prediction that guide hardware and software development and long-term planning.

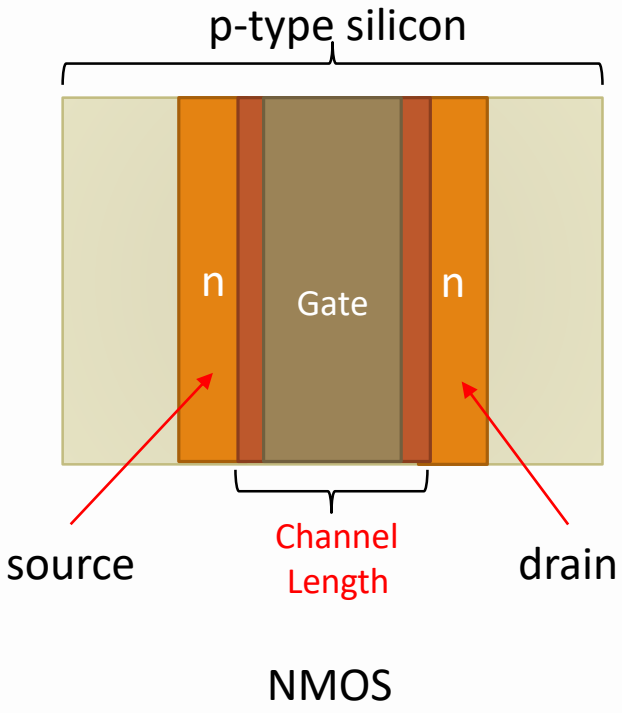


Channel Length/Feature Size

- Latest news pointed out that TSMC has made a breakthrough in developing their 2nm or even 1nm process.

$$1\mu m = 10^{-6}m$$
$$1nm = 10^{-9}m$$

Hair: 100 μm
Blood cell: 7 μm
Virus: ~0.1 μm



Eight Great Ideas

- Design for Moore's Law
- Use abstraction to simplify design
- Make the common case fast
- Performance via parallelism
- Performance via pipelining
- Performance via prediction
- Hierarchy of memories
- Dependability via redundancy

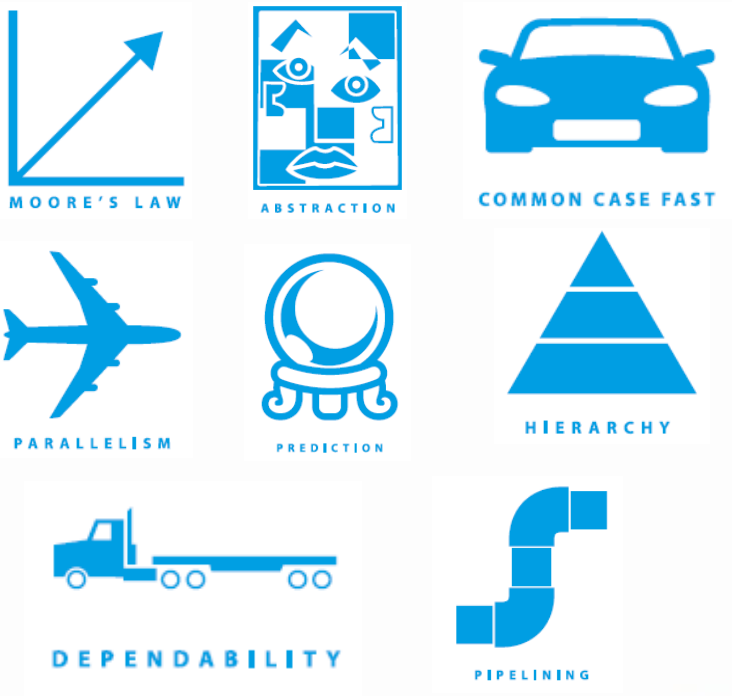


Figure is from "Computer Organization and Design MIPS Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design)" 5th Edition by David A. Patterson (Author), John L. Hennessy

Below Your Program

- Applications
  - "If I have seen further it is by standing on the shoulders of Giants." by Isaac Newton in 1675
  - Written in high-level languages, Abstraction
- System Software
  - Compiler, operating system
  - Managing I/O, memory/storage
  - Scheduling tasks and resources
- Hardware
  - CPU, memory, I/O controllers, storage, etc.

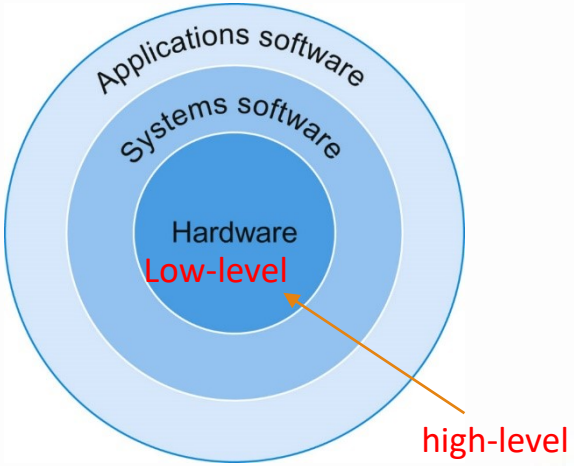


Figure is from "Computer Organization and Design MIPS Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design)" 5th Edition by David A. Patterson (Author), John L. Hennessy

Levels of Programming Languages

- High-level language
  - Closer to what human can perceive
  - Productivity and portability
- Assembly language
  - In between hardware and software
  - Dealing with registers and addresses
- Hardware representation
  - Binary code

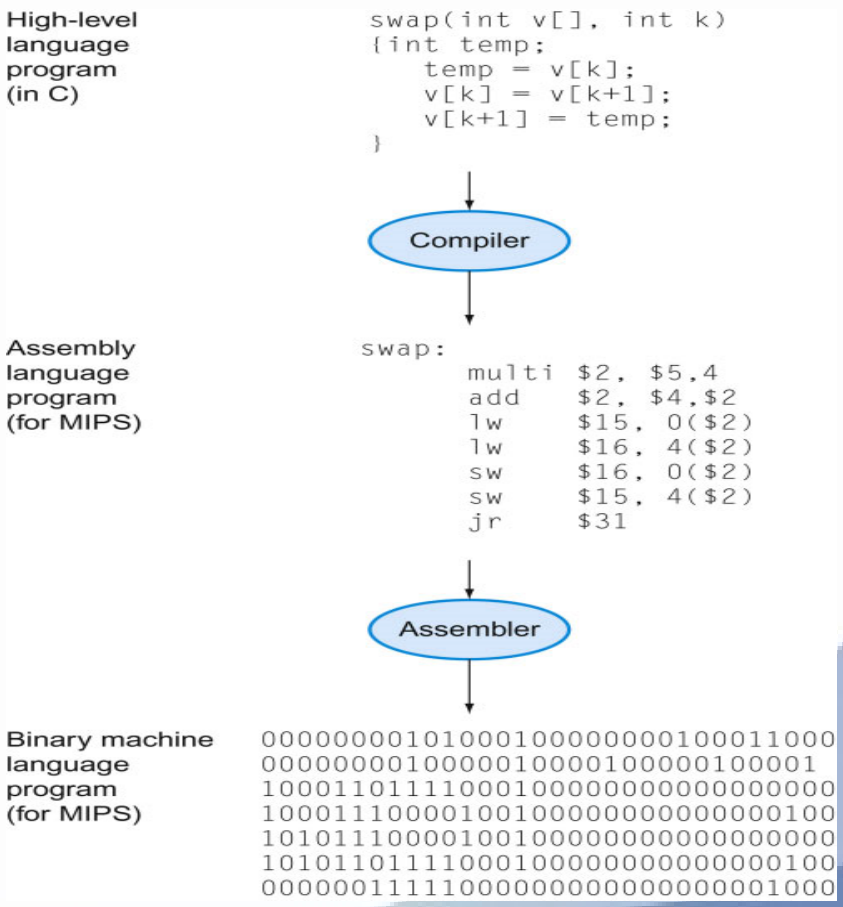
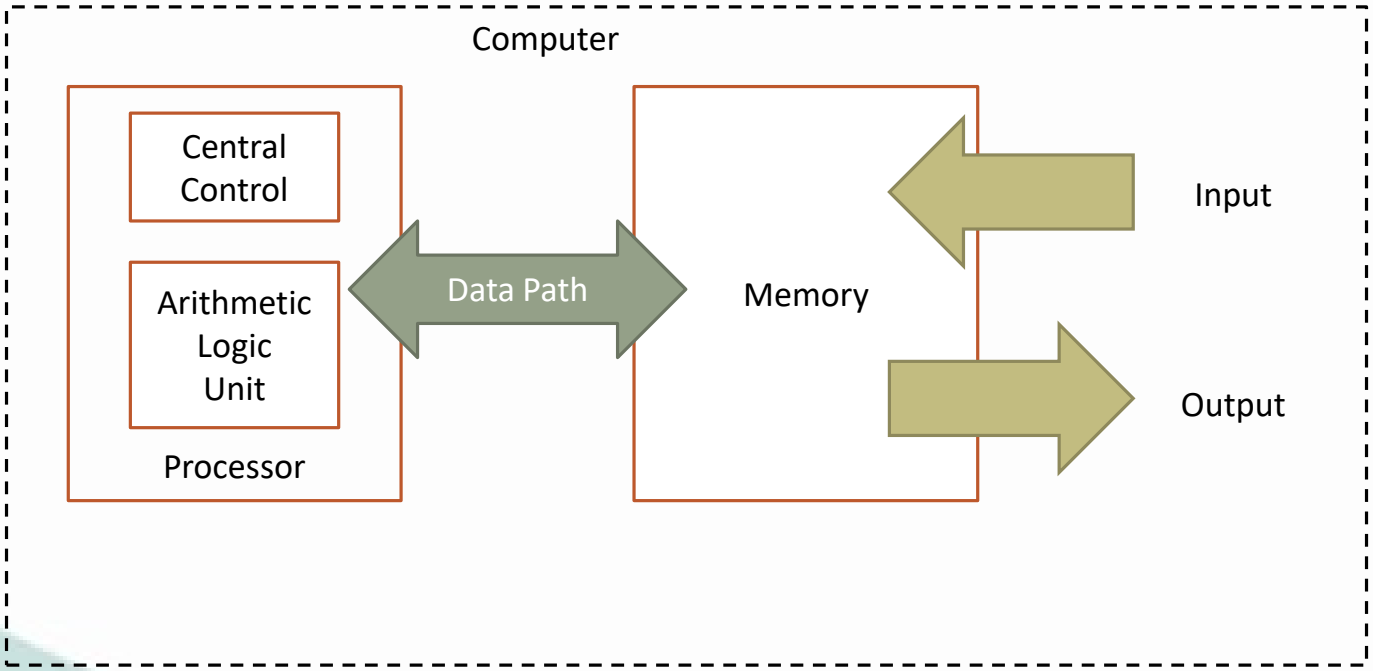


Figure is from "Computer Organization and Design MIPS Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design)" 5th Edition by David A. Patterson (Author), John L. Hennessy

Under the Covers

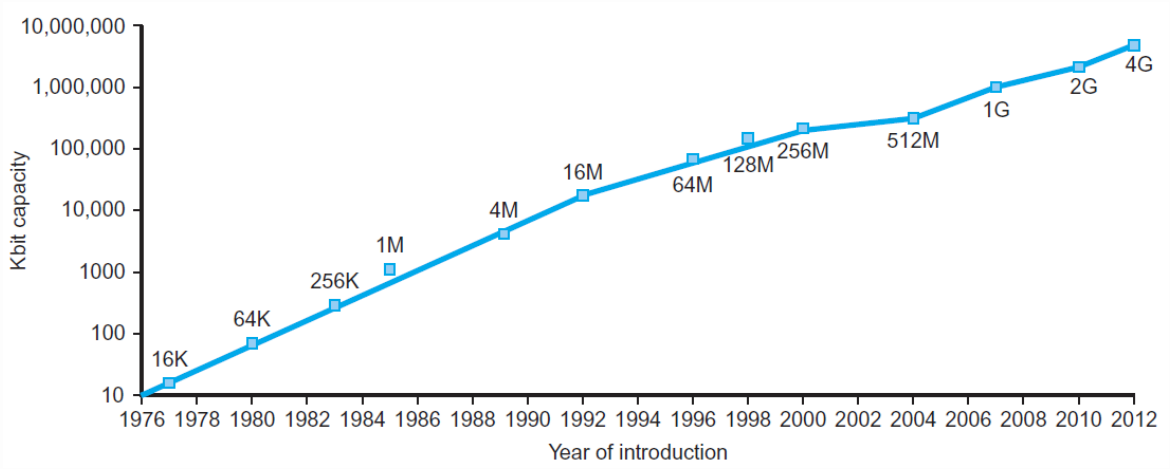


I/O devices:

- Display
- Keyboard
- Mouse
- HD, flash
- Network devices

Technology Trends

- For Memory



- For Processors

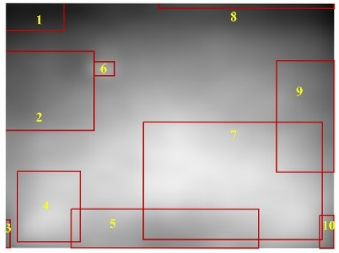
Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated circuit (IC)	900
1995	Very large scale IC (VLSI)	2,400,000
2013	Ultra large scale IC	250,000,000,000

Figure is from "Computer Organization and Design MIPS Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design)" 5th Edition by David A. Patterson (Author), John L. Hennessy

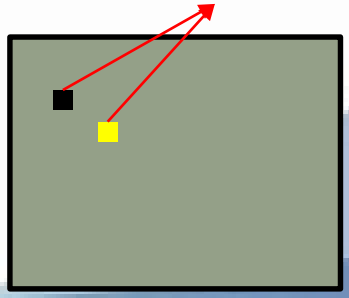
Output Devices

- Liquid Crystal Display (LCD)
  - Non-emissive – light produced behind the screen
  - Backlight provided for the entire screen even when it's presenting black.
  - May have "MURA" effect (**clouding**) due to uneven brightness levels.
- Plasma Display Panel (PDP)
  - Emissive – turn electrical energy into light
  - Each pixel is controlled by three cells with the primary colors of light.
  - Almost died out due to lower-cost LCDs and higher-contrast OLEDs.
- Organic Light-Emitting Diodes (OLED)
  - Emissive
  - Fast response time
  - No backlighting - light produced on individual pixels so as to control color, brightness, black levels, etc.
  - Power-efficient and vivid color

MURA effect



Pixels



Ngo *et al.* A new algorithm on the automatic TFT-LCD mura defects inspection based on an effective background reconstruction, Journal of the Society for Information Dispay, 2018.

Input Devices

- Optical mouse
  - Using light-emitting diodes (LEDs) to detect motions.
  - Using an optoelectronic sensor to take images for detecting motions
- Keyboard
- Touch screen
  - I/O device
  - LCD or OLED
  - Resistive/Capacitive
  - Smartphones often use capacitive sensing

A Microsoft wireless optical mouse (from wiki)

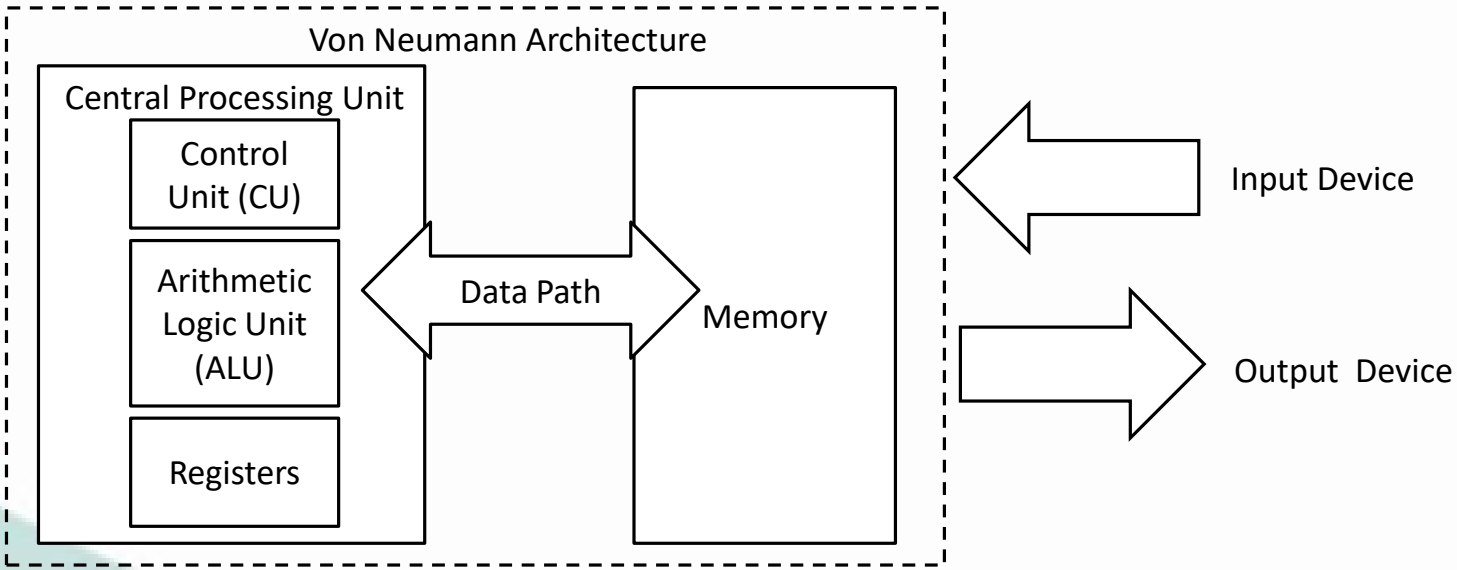
https://ee.ntu.edu.tw/upload/hischool/doc/2013.05.pdf

Abstraction

- Abstraction can help make complex things simpler by
  - Breaking down things into different levels of hierarchy to hide lower-level detail
- Instruction set architecture (ISA)
  - The hardware/software interface
  - The ISA plus system software interface

Von Neumann Architecture

- First published by John von Neumann in 1945



Von Neumann Architecture

- Stored-program Computer Concept
  - Instructions and data are both stored in the memory
  - ALU processes binary data
- Central Processing Unit (CPU)
  - Executing instructions
  - Containing ALU, CU, and Registers
- Arithmetic Logic Unit (ALU)
  - Arithmetic (add, subtract etc) and logic (AND, OR, NOT etc) operations
- Control Unit (CU)
  - Controls the operation of the ALU, memory and input/output devices based on instructions read
- Von Neumann Bottleneck
  - The overall computer performance is limited to the **rate of data transfer** allowed by the bottleneck

How to Define Performance

- Different aspects of performance of an airplane

Airplane	Passenger Capacity	Cruising Range (miles)	Cruising Speed (m.p.h.)
Boeing 777	375	4630	610
Boeing 747	470	4150	610
BAC/Sud Concorde	132	4000	1350
Douglas DC-8-50	146	8720	544

Boeing 777

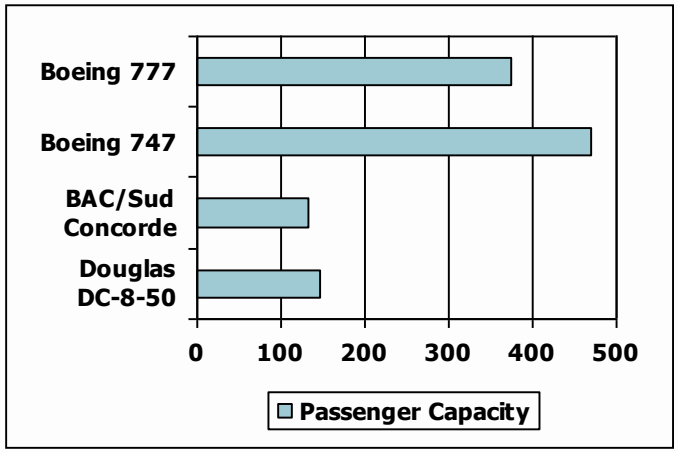
Boeing 747

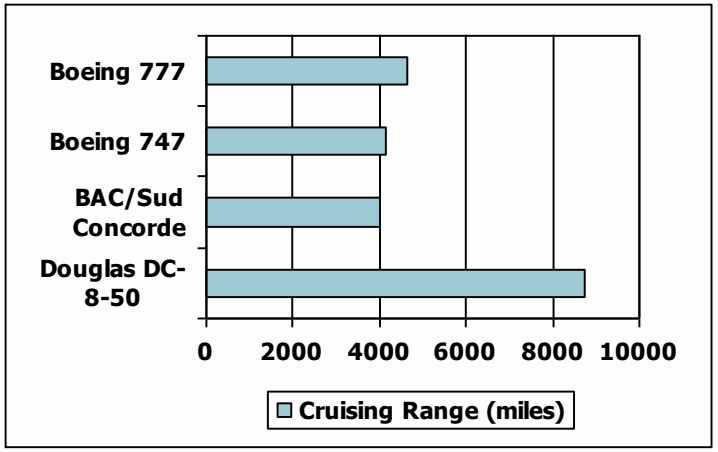
Concorde

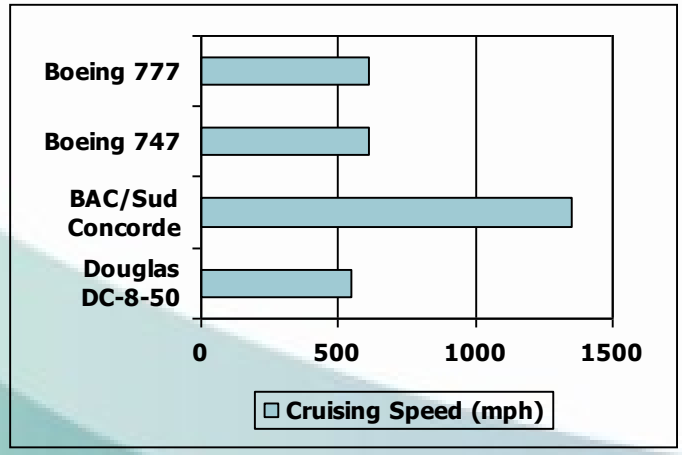
Douglas DC-8-50

Image source: Wiki

Performance Definition







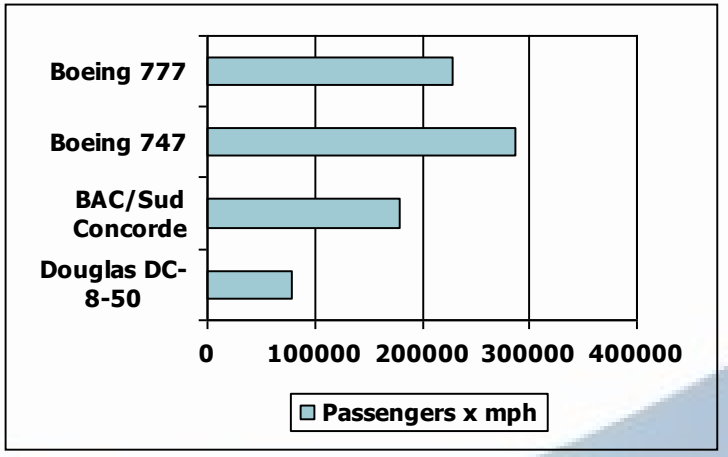


Figure is from "Computer Organization and Design MIPS Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design)" 5th Edition by David A. Patterson (Author), John L. Hennessy

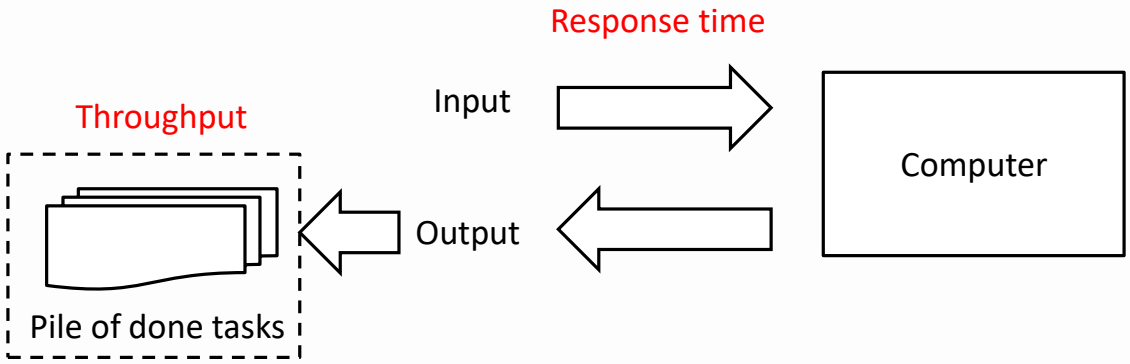
Computer Performance

- **Response Time** (execution time)
  - The total time required for the computer to complete a task
- **Throughput** (bandwidth)
  - The total amount of work done in a given time. (work/time)
- Example

  How are response time and throughput affected by
  - Replacing the processor with a faster version?
  - Adding more processors?

Response Time and Throughput

- Decreasing response time almost always improves throughput.



- Replacing the processor with a faster version — Response time, Throughput — both will be affected !!!
- Adding more processors — Throughput

Measuring Performance (Execution Time)

- Program execution time is measured in **seconds** per program.
- The total time to complete a task: wall clock time, response time, or elapsed time
  - Including disk accesses, memory accesses, input/output activities, operating system overhead, and so on
- **CPU execution time** (CPU time)
  - Time spent for doing a given task by CPU
  - Not including I/O or running other programs

User CPU time
The CPU time spent in a program itself

System CPU time
The CPU time spent in the operating system performing tasks for a program

Elapsed time (system performance) ≠ **CPU time (CPU performance)**

Performance

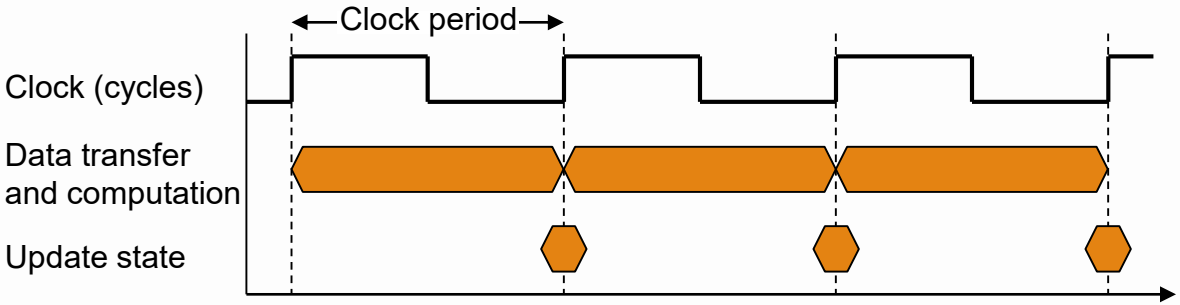
- Mainly focusing on response time for now
- Relative Performance

$$\text{Performance} = \frac{1}{\text{Execution Time}}$$

X is **N** times faster than Y -> $\frac{\text{Execution Time of Y}}{\text{Execution Time of X}} = \frac{\text{Performance of X}}{\text{Performance of Y}} = N$

CPU Clock Cycles

- Using a constant-rate clock to trigger operations of computers



- Clock period: length of a complete clock cycle
  - e.g., 250ps = 0.25ns = 250×10^{-12}s (ps: picoseconds)
- Clock frequency (rate): cycles per second
  - e.g., 4.0GHz = 4000MHz = 4.0×10^{9}Hz

$$\text{Clock rate} = \frac{1}{\text{Clock Period}}$$

CPU Performance

- CPU execution time (CPU time)

$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$

$$= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}$$

$$= \text{CPU Clock Cycles} \times \text{Clock Period}$$

So, reducing number of clock cycles or increasing clock rate would boost performance.

CPU Time Example

- Computer A: 2GHz clock rate, A program runs 10s CPU time on it
- One would like to designing Computer B to run the program for 6s CPU time
  - This design will require 1.2 times as many clock cycles as A
  - Target clock rate for B ?
- Clock rate: cycles per second

$$\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6\text{s}}$$

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$

$$= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$$

$$\text{Clock Rate}_B = \frac{1.2 \times 20 \times 10^9}{6\text{s}} = \frac{24 \times 10^9}{6\text{s}} = 4\text{GHz}$$

Instruction Performance

- Instruction counts a program needs
- A program -> Compiler -> Instructions -> CPU to run
- CPU execution time depends upon how many instructions a program needs
- Clock cycles per instruction (CPI)
  - Average number of clock cycles each instruction takes to execute for a program
  - Determined by CPU hardware
  - Different instructions may take different amounts of time

- Clock rate: cycles per second

Clock Cycles = Instruction Count * Cycles per Instructions (CPI)

CPU Time = Instruction Count * CPI * Clock Cycle Time

$$= \frac{\text{Instruction Count} * \text{CPI}}{\text{Clock Rate}}$$

$$\text{CPU Time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instruction Count}}{\text{Program}} \times \frac{\text{Clock Cycle}}{\text{Instruction Count}} \times \frac{\text{Seconds}}{\text{Clock Cycle}}$$

CPI Example

- Assume we have two implementations of the same instruction set architecture
- Computer A: Cycle Time = 250ps, CPI = 2.0 for some program P
- Computer B: Cycle Time = 500ps, CPI = 1.2 for P
- Which is **faster**, and by how much?

(CPU Time)

Assuming the Instruction Count P needs = I

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$
$$= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$
$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$
$$= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$
$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{I \times 600\text{ps}}{I \times 500\text{ps}} = 1.2$$

Multiple Instruction Classes

- If different instruction classes take different numbers of cycles

$$\text{Clock cycles} = \sum_{k=1}^{n} \text{CPI}_k \times \text{Instruction Count}_k$$

- $$\text{CPI} = \frac{\text{Clock cycles}}{\text{Instruction}} = \frac{\sum_{k=1}^{n} \text{CPI}_k \times \text{Instruction Count}_k}{\sum_{j=1}^{n} \text{Instruction Count}_j}$$

$$= \sum_{k=1}^{n} \text{CPI}_k \times \frac{\text{Instruction Count}_k}{\sum_{j=1}^{n} \text{Instruction Count}_j}$$

Relative frequency

Another Example of CPI

- If you were to design a compiler based on the following information

	CPI for each instruction class		
	A	B	C
CPI	1	2	3

Class	Instruction Counts		
	A	B	C
Sequence 1	2	1	2
Sequence 2	4	1	1

- Which code sequence executes more instructions?
- Which one is faster?
- What is the CPI for each sequence?

Sequence 1: IC = 5

- Clock Cycles
  = 2×1 + 1×2 + 2×3
  = 10
- CPI = 10/5 = 2.0

Sequence 2: IC = 6

- Clock Cycles
  = 4×1 + 1×2 + 1×3
  = 9
- CPI = 9/6 = 1.5

Summary

$$\text{CPU Time} = \frac{\textbf{Seconds}}{\textbf{Program}} = \frac{\textbf{Instruction Count}}{\textbf{Program}} \times \frac{\textbf{Clock Cycle}}{\textbf{Instruction Count}} \times \frac{\textbf{Seconds}}{\textbf{Clock Cycle}}$$

- Computer performance depends on
  - Algorithm: affects instruction counts, CPI (diff implementation)
  - Programming language: affects instruction counts, CPI (diff instruction)
  - Compiler: affects instruction counts, CPI
  - Instruction set architecture: affects instruction counts, CPI, (Clock Rate, single-clock instruction)
  - Computer organization: CPI (carry ripple adder vs look-ahead adder), clock rate
  - Technology: clock rate

CPU Performance

- Computer performance depends on

HW/SW Factors	Impact on	How
Algorithm	ICs, CPI	It determines high-level ICs, and thus low-level ICs. It may opt for instructions that take more clocks.
Programming Language	ICs, CPI	For the same purpose, different languages affect translated ICs. Java/C++ support data abstraction, which need indirect calls that take more clocks.
Compiler	ICs, CPI	It directly affects translated ICs and CPI due to different compiling strategies it may take.
ISA	ICs, CPI, Clock Rate	It determines what instructions a function needs and how many clocks it takes. For a single-clock processor, a complex instruction decides a critical path, thus clock period.
Computer Organization	CPI, Clock Rate	Hardware design affect the clock rate and how many clocks it takes to execute an instruction.
Technology	Clock Rate	It affects how a CPU is designed.

Amdahl's Law

- The maximum possible improvement obtained by improving a particular part of a system

$$\text{Overall Speedup} = \frac{\text{Total time required}}{\frac{\text{Time accounts for the part}}{\text{Speedup factor}} + \text{Time for the other parts}}$$

- Example: Runs a program for 100s where "divide" accounts for 80s
  - The maximum possible improvement based on optimizing "divide"

Speedup factor for "divide" = p

Overall speedup = $\frac{100}{\frac{80}{p}+20}$

make the common case fast

Millions of Instructions Per Second (MIPS)

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

$$= \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

- Problems of using MIPS to measure computer performance
  - Not considering ISA among computers
  - Not considering differences in complexity between instructions

CPU Time: IC, CPI, Clock Rate
MIPS: CPI, Clock Rate

$$\text{CPU Time} = \frac{\textbf{Seconds}}{\textbf{Program}} = \frac{\textbf{Instruction Count}}{\textbf{Program}} \times \frac{\textbf{Clock Cycle}}{\textbf{Instruction Count}} \times \frac{\textbf{Seconds}}{\textbf{Clock Cycle}}$$

Example of MIPS

Measurement	Computer A	Computer B
Instruction count	10 billion	8 billion
Clock rate	4 GHz	4 GHz
CPI	1.0	1.1

a. Which computer has the higher MIPS rating?
b. Which computer is faster?

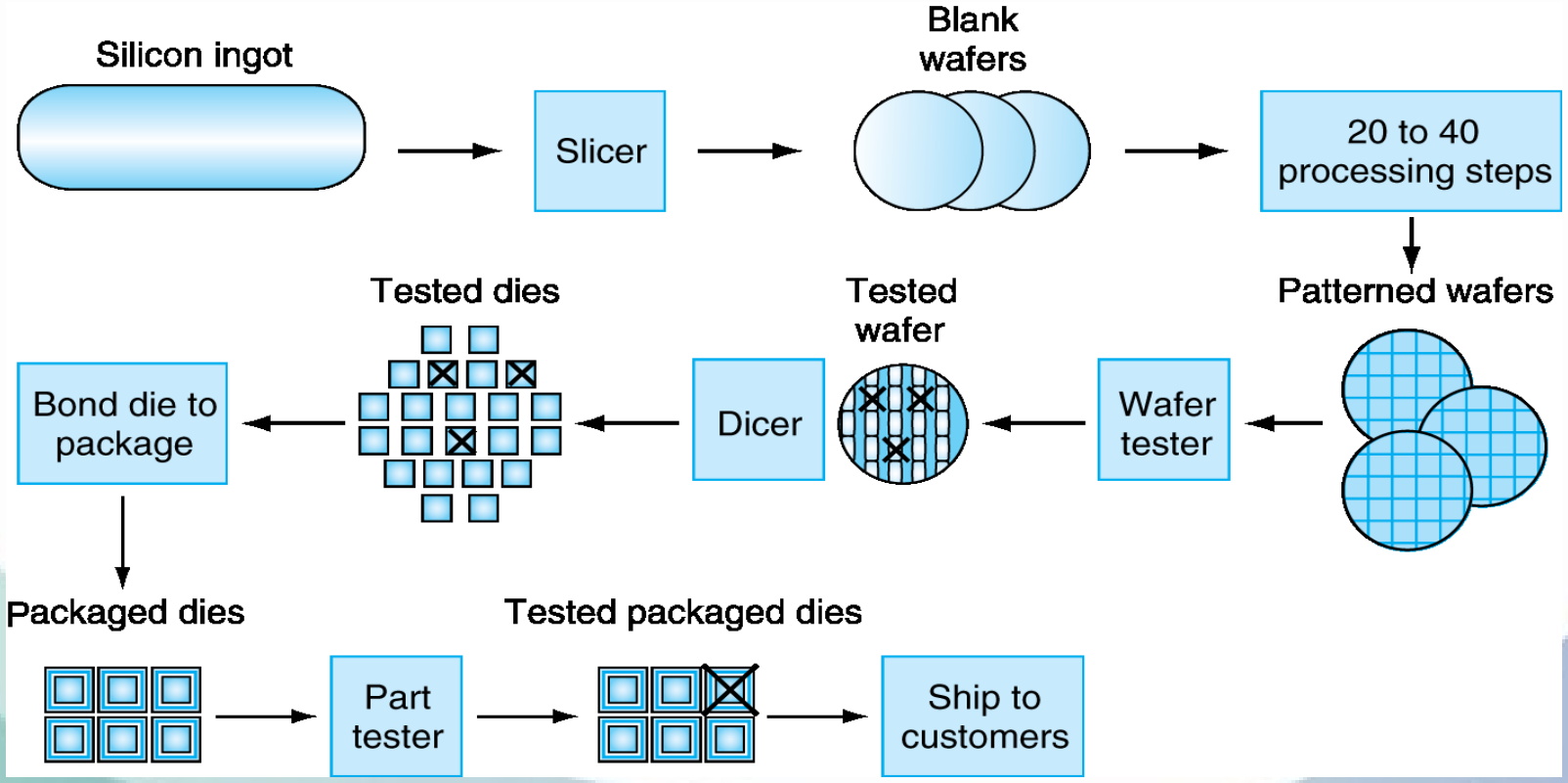
$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6} = \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

$$\text{CPU Time} = \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instruction Count}}{\text{Program}} \times \frac{\text{Clock Cycle}}{\text{Instruction Count}} \times \frac{\text{Seconds}}{\text{Clock Cycle}}$$

Practices

- (Yes/No) ISA (instruction set architecture) is an abstraction that enables different implementations for the processor, for example, a pipelined implementation or a non-pipelined one.
- In evaluating the performance of a computer, two performance metrics response time and throughput are usually used. Explain the meanings of these two terms.
- If computer A runs a program in 15 seconds and computer B runs the same program in 25 seconds, how much faster is A than B?

IC Manufacturing Process



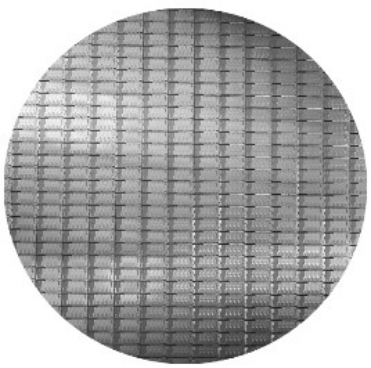
https://www.youtube.com/watch?app=desktop&v=xftnhfa-Dmo

IC Cost

$$\frac{Cost}{die} = \frac{\frac{cost}{wafer}}{\frac{dies}{wafer} \times Yield}$$

$$\frac{Dies}{wafer} \approx \frac{wafer\ area}{die\ area}$$

$$Yield = \frac{1}{\left(1 + \left(\frac{defects}{area} \times \frac{die\ area}{2}\right)\right)^2}$$



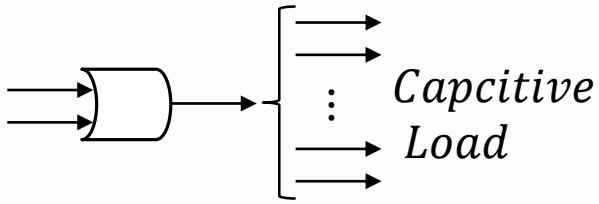
Intel Core i7 Wafer
w/ 32nm tech.

- Yield: # of working dies per wafer
- Architecture and circuit design determine Die area

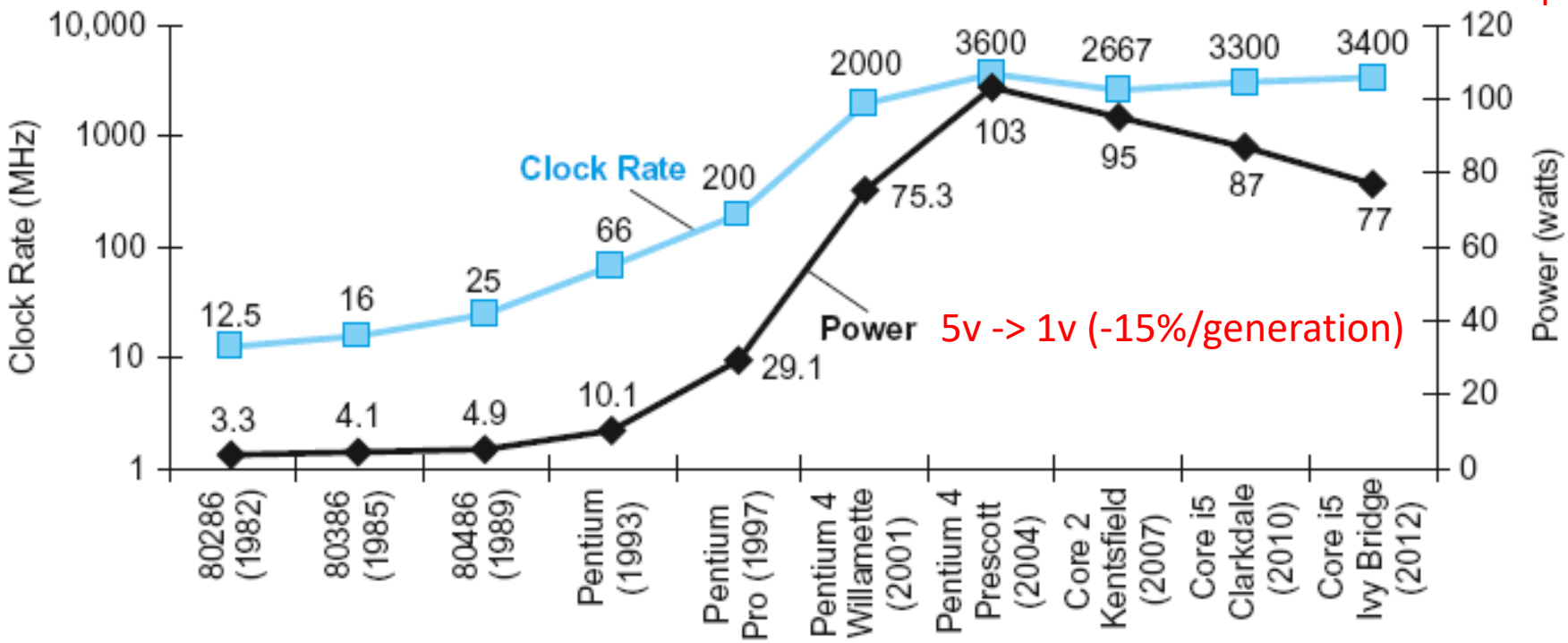
Trend - Power vs. Clock Rate

- As for CMOS ICs,

$$Power \propto Capcitive\ Load \times Voltage^2 \times Frequency$$



number of transistors connected to an output



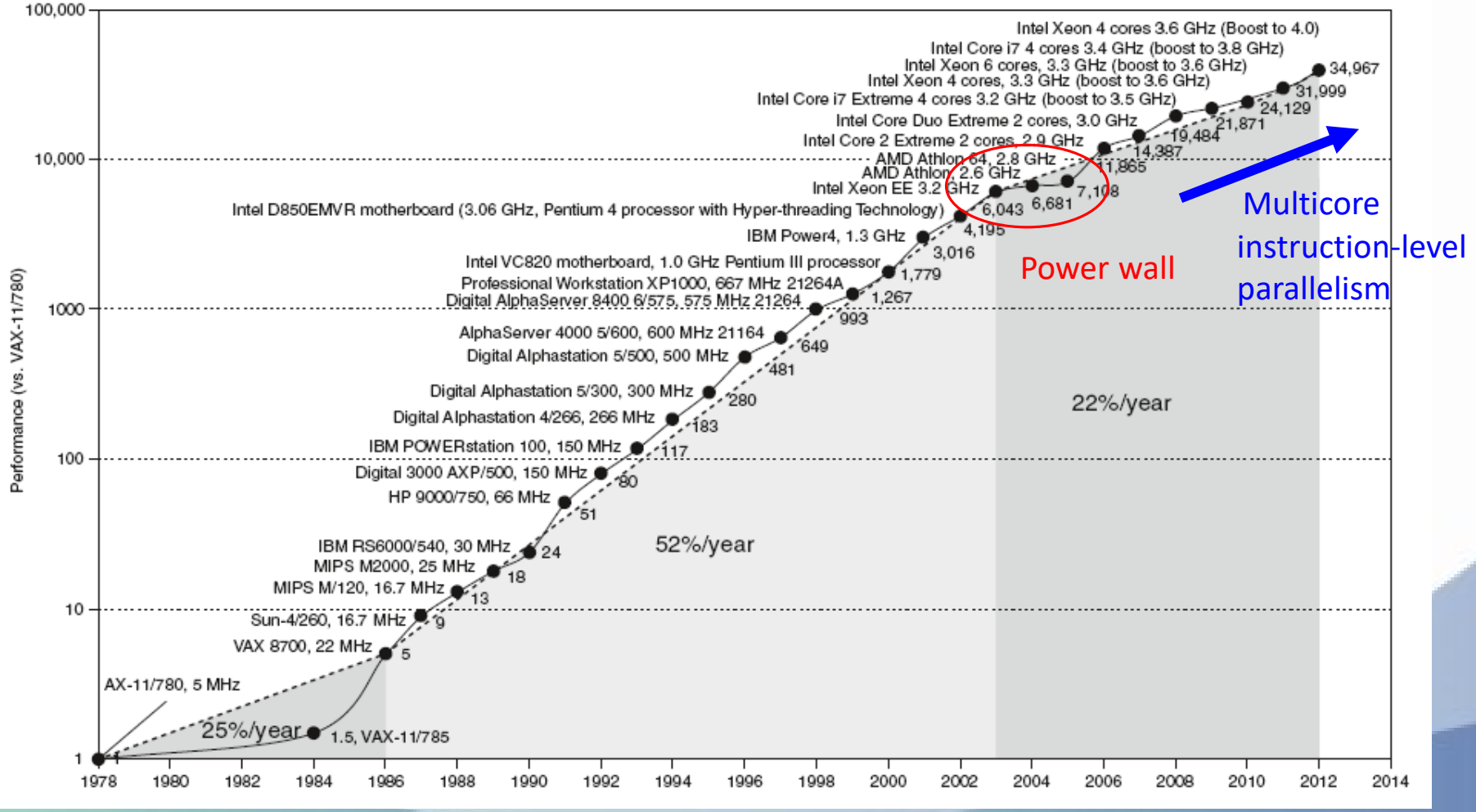
Power Reduction

- Suppose a new CPU has
  - 85% of capacitive load of old CPU
  - 15% voltage and 15% frequency reduction

$$\frac{P_{new}}{P_{old}} = \frac{C_{old} \times 0.85 \times (V_{old} \times 0.85)^2 \times F_{old} \times 0.85}{C_{old} \times V_{old}^{\ 2} \times F_{old}} = 0.85^4 = 0.52$$

- The power wall -> leakage power (can't reduce voltage more) and heat generation (can't reduce more heat)

Uniprocessor/Multiprocessor Performance



Transition from Uniprocessor to Multiprocessor

- Multicore microprocessors
- Parallel programming
  - Instruction-level parallelism - Hardware handles the work and execute multiple instructions, there is no extra efforts made by programmers.
  - Multicore parallelism – programmers need to optimize communication and synchronization

Measuring and Evaluating CPU Performance

- For different purposes, it should provide different measuring criteria.
  - Image processing/computer graphics
  - Video editing
  - Office software
  - etc.
- Benchmark
  - Measuring the performance of a CPU depends on code used to test it
  - Need industry standards (benchmark programs)
    - For different processors to be compared fairly