CSE2021 Computer Organization

Instructor: Gulzar Khuwaja, PhD
Department of Electrical Engineering
& Computer Science
Lassonde School of Engineering
York University

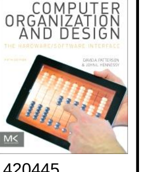
CSE2021 Computer Organization

- Instructor: Gulzar Khuwaja, PhD
  Email: khuwaja@cse.yorku.ca
  Tel: 416-736-2100 x 77874
- Couse Web:
  https://wiki.eecs.yorku.ca/course_archive/2013-14/S/2021/
- Schedule:
  Lectures: R 7:00 – 10:00 pm Room CLH M
  Labs: M 7:00 – 10:00 pm Room LAS 1006
  Office Hours: M 6:00 –7:00pm Room LAS 2018
  R 5:30 – 6:30pm Room LAS 2018

CSE2021 Computer Organization

Text books:

- Computer Organization and Design
  The Hardware/Software Interface, 5th Edition (2014)
  by David A. Patterson and John L. Hennessy
  Morgan Kaufmann Publishers (Elsevier)
  ISBN 978-0-12-407726-3
- MIPS Assembly Language Programming 2003
  by Robert Britton, Pearson Publishers ISBN-10: 0131420445



Assessment:

- Lab projects: 40%
- Midterm test: 30%
- Final exam: 30%

COMPUTER ORGANIZATION AND DESIGN
The Hardware/Software Interface
5th Edition

Chapter 1

Computer Abstractions and Technology

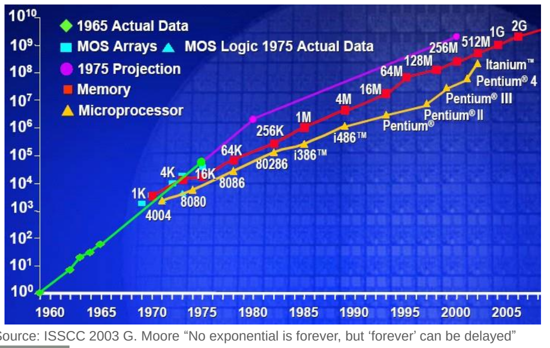
Moore's Law

- Moore's Law states that integrated circuit resources double every 18–24 months.
- Moore's Law resulted from a 1965 prediction of such growth in IC capacity made by Gordon Moore, one of the founders of Intel.
- Moore's Law graph to represent designing for rapid change

Moore's Law

	Year of introduction	Transistors
4004	1971	2,250
8008	1972	2,500
8080	1974	5,000
8086	1978	29,000
286	1982	120,000
386™	1985	275,000
486™ DX	1989	1,180,000
Pentium®	1993	3,100,000
Pentium II	1997	7,500,000
Pentium III	1999	24,000,000
Pentium 4	2000	42,000,000

Moore's Law

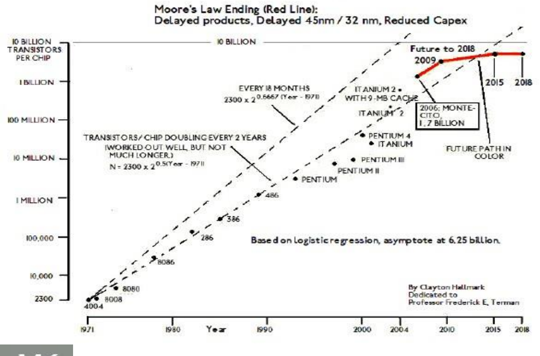


Source: ISSCC 2003 G. Moore "No exponential is forever, but 'forever' can be delayed"

Is Moore's Law Ending?

- Intel's former chief architect Bob Colwell says: Moore's law will be dead within a decade (August 2013).
- The end of Moore's Law is on the horizon, says AMD.
- Theoretical physicist Michio Kaku believes Moore's Law has about 10 years of life left before ever-shrinking transistor sizes smack up against limitations imposed by the laws of thermodynamics and quantum physics (April 2013).

Is Moore's Law Ending? (2)

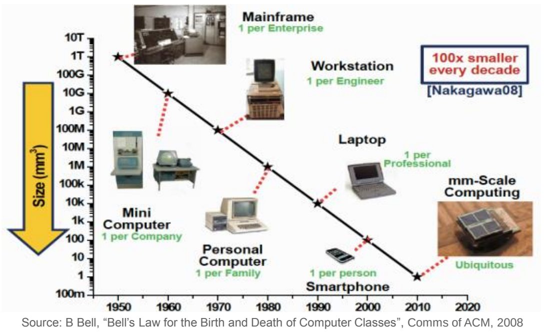


Bell's Law

Bell's Law for the birth and death of computer classes:

- Bell's law of computer classes formulated by Gordon Bell in 1972 describes how computing systems (computer classes) form, evolve and may eventually die out.
- Roughly every decade a new, lower priced computer class forms based on a new programming platform, network, and interface resulting in new industry.
- In 1951, men could walk inside a computer and now, computers are beginning to "walk" inside of us .

Bell's Law

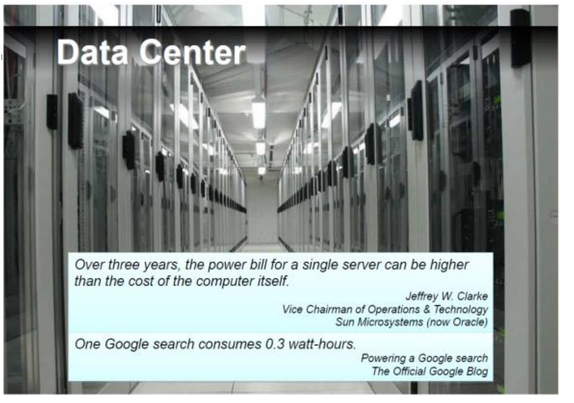


Source: B Bell, "Bell's Law for the Birth and Death of Computer Classes", Comms of ACM, 2008

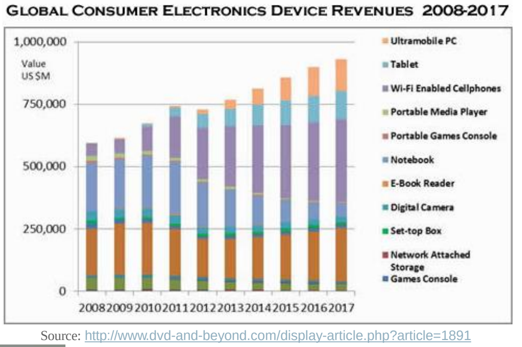
The 1st Generation Computer

Source: http://www.computerhistory.org

Computers Today



Future Direction



Source: http://www.dvd-and-beyond.com/display-article.php?article=1891

The Computer Revolution

§1.1 Introduction

- Progress in computer technology
  - Underpinned by Moore's Law
- Makes novel applications feasible
  - Computers in automobiles
  - Cell phones
  - Human genome project
  - World Wide Web
  - Search Engines
- Computers are pervasive

The Computer Revolution (2)

§1.1 Introduction

- Computers in automobiles
  - reduce pollution, improve fuel efficiency via engine controls, and increase safety through blind spot warnings, lane departure warnings, moving object detection, and air bag inflation to protect occupants in a crash
- Cell phones
  - more than half of the planet having mobile phones, allowing person-to-person communication to almost anyone anywhere in the world

The Computer Revolution (3)

§1.1 Introduction

- Human genome project
  - a global effort to identify the estimated 30,000 genes in human DNA to figure out the sequences of the chemical bases that make up human DNA to address ethical, legal, and social issues
  - The cost of computer equipment to map and analyze human DNA sequences was hundreds of millions of dollars. Since, costs continue to drop, we will soon be able to acquire our own genome, allowing medical care to be tailored to us

The Computer Revolution (4)

§1.1 Introduction

- World Wide Web
  - has transformed our society. Web has replaced libraries and newspapers
- Search Engines
  - As the content of the web grew in size and in value, many people rely on search engines for such a large part of their lives that it would be a hardship to go without them

Classes of Computers

- Personal computers
- Server computers
- Supercomputers
- Embedded computers

Classes of Computers (2)

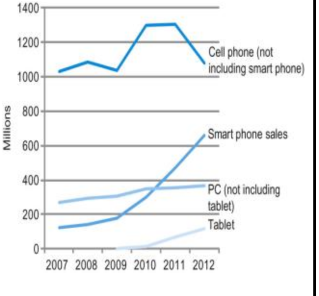
- Personal computers
  - Computers designed for use by an individual
  - General purpose, variety of software
  - Subject to cost/performance tradeoff
- Server computers
  - Computers used for running larger programs for multiple users, often simultaneously
  - Network based
  - High capacity, performance, reliability
  - Range from small servers to building sized

Classes of Computers (3)

- Supercomputers
  - Consist thousands of processors and many terabytes of memory
  - High-level scientific and engineering calculations
  - Highest capability and cost but represent a small fraction of the overall computer market
- Embedded computers
  - Hidden as components of systems
  - Largest class of computers and span the widest range of applications
  - Strict power/performance/cost limitations

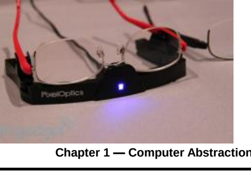
The PostPC Era

- Smart phones represent the recent growth in cell phone industry, and they passed PCs in 2011.
- Tablets are the fastest growing category, nearly doubling between 2011 and 2012.
- Recent PCs and traditional cell phone categories are relatively flat or declining.



The PostPC Era (2)

- Personal Mobile Devices (PMDs)
  - Small wireless devices
  - Battery operated
  - Connects to the Internet
  - Hundreds of dollars
  - Smart phones, tablets, electronic glasses



The PostPC Era (3)

- Cloud computing
  - Term cloud essentially used for the Internet
  - Portion of software run on a PMD and a portion run in the Cloud
  - Warehouse Scale Computers (WSC)
    - Big datacenters containing 100,000 servers
    - Amazon and Google cloud vendors
  - Software as a Service (SaaS)
    - Delivers software and data as a service over the Internet
    - Web search and social networking

What You Will Learn

- How programs are translated into machine language
  - and how hardware executes them
- The hardware/software interface
- What determines program performance
  - and how it can be improved
- How hardware designers improve performance
- What is parallel processing

Understanding Performance

- Algorithm
  - determines number of operations executed
- Programming language, compiler, architecture
  - determine number of machine instructions executed per operation
- Processor and memory system
  - determine how fast instructions are executed
- I/O system (including OS)
  - determine how fast I/O operations are executed

Eight Great Design Ideas

- Design for ***Moore's Law***
  - Computer designs can take years, resources available per chip can easily double or quadruple between start and finish of project
  - Anticipate where technology will be when design finishes
- Use ***Abstraction*** to simplify design
  - Hide lower-level details to offer a simpler model at higher levels
- Make the ***Common Case Fast***
  - To enhance performance better than optimizing the rare case

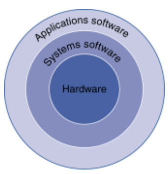
Eight Great Design Ideas (2)

- Performance *via* ***Parallelism***
  - A form of computation in which many calculations are carried out simultaneously
- Performance *via* ***Pipelining***
  - A particular pattern of parallelism
  - A set of data processing elements connected in series, so that the output of one element is the input of the next one
- Performance *via* ***Prediction***
  - It can be faster on average to guess and start working rather than wait

Eight Great Design Ideas (3)

- ***Hierarchy*** of memories
  - The closer to the top, the faster and more expensive per bit of memory
  - The wider the base of the layer, the bigger the memory
- ***Dependability*** *via* redundancy
  - Design systems dependable by including redundant components
    - to take over when failure occurs, and
    - to help detect failures

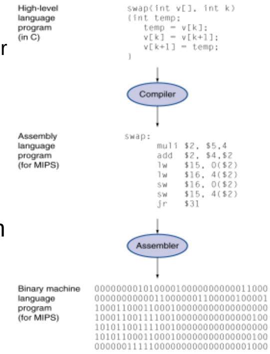
Below Your Program



- Application software
  - Written in high-level language
- System software
  - Compiler: translates HLL code to machine code
  - Operating System:
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O controllers

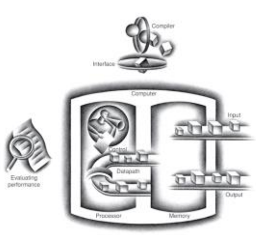
Levels of Program Code

- High-level language
  - Level of abstraction closer to problem domain
  - Provides for productivity and portability
- Assembly language
  - Textual representation of instructions
- Hardware representation
  - Binary digits (bits)
  - Encoded instructions and data



Components of a Computer

The BIG Picture



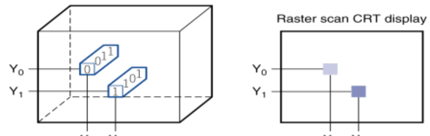
- Same components for all kinds of computer
  - Desktop, Server, Embedded
- Input/Output includes
  - User-interface devices
    - Display, keyboard, mouse
  - Storage devices
    - Hard disk, CD/DVD, flash
  - Network adapters
    - For communicating with other computers

Touchscreen

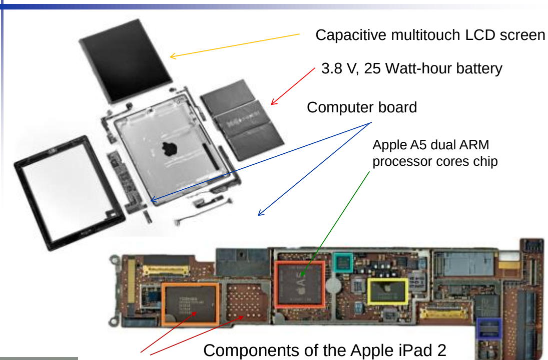
- PostPC device
- Supersedes keyboard and mouse
- Resistive and Capacitive types
  - Most tablets, smart phones use capacitive
  - Capacitive allows multiple touches simultaneously

Through the Looking Glass

- LCD screen: picture elements (pixels)
- Each coordinate in the frame buffer on the left determines the shade of the corresponding coordinate for the raster scan CRT display on the right.
- Pixel (X_0,Y_0) contains the bit pattern 0011, which is a lighter shade on the screen than the bit pattern 1101 in pixel (X_1,Y_1).



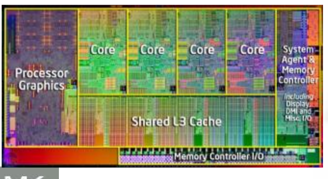
Opening the Box

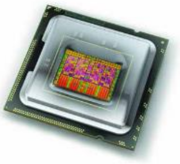


Components of the Apple iPad 2

Inside the Processor (CPU)

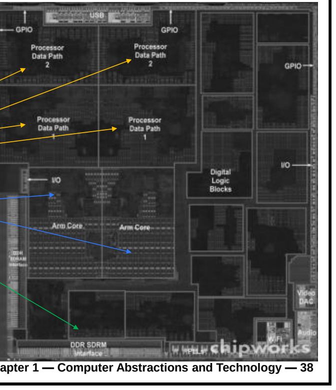
- Datapath: performs operations on data
- Control: sequences datapath, memory, ...
- Cache memory
  - Small fast SRAM memory for immediate access to data





Inside the Processor (CPU)-2

- The processor IC inside Apple A5 package
  - Size of chip is 12.1 by 10.1 mm
  - Graphical processor unit (GPU) with four datapaths
  - Two identical ARM processors
  - Interfaces to main memory (DRAM)



Abstractions

The BIG Picture

- Abstraction helps us deal with complexity
  - Hides lower-level details
- Instruction Set Architecture (ISA) or Computer Architecture
  - The hardware/software interface
  - Includes instructions, registers, memory access, I/O, and so on
- Operating system hides details of doing I/O, allocating memory from programmers

A Safe Place for Data

- Volatile main memory
  - Loses instructions and data when power off
- Non-volatile secondary memory
  - Flash memory
  - Optical disk (CDROM, DVD)
  - Magnetic disk

Networks

Backbone of computer systems

- Communication
  - Information is exchanged between computers at high speeds
- Resource sharing
  - Rather than each computer having its own I/O devices, computers on the network can share I/O devices
- Nonlocal access
  - By connecting computers over long distances, users need not be near the computer they are using

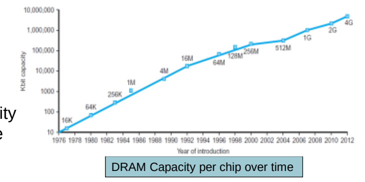
Networks (2)

- Local area network (LAN): Ethernet
  - A network designed to carry data within a geographically confined area, typically within a single building – 40 gigabits/s
- Wide area network (WAN): the Internet
  - A network extended over hundreds of kilometers that can span a continent
- Wireless network: WiFi, Bluetooth
  - transmission rates from 1 to100 million bits per second

Technology Trends

§1.5 Technologies for Building Processors and Memory

- Electronics technology continues to evolve
  - Increased capacity and performance
  - Reduced cost



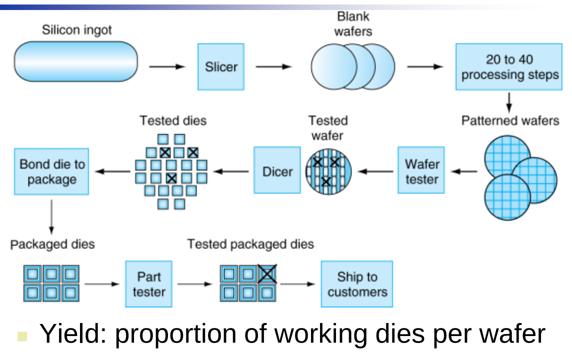
DRAM Capacity per chip over time

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

Semiconductor Technology

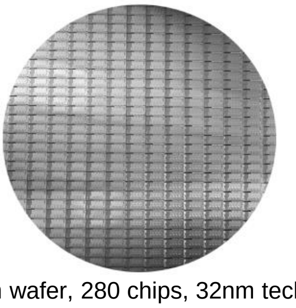
- Silicon: Semiconductors
- Add materials to transform properties:
  - Conductors
    - adding copper or aluminum wire
  - Insulators
    - like plastic or glass
  - Switches (transistors)
    - conduct or insulate under special conditions

Manufacturing ICs



- Yield: proportion of working dies per wafer

Intel Core i7 Wafer



- 300mm wafer, 280 chips, 32nm technology
- Each chip is 20.7 by 10.5 mm

Integrated Circuit Cost

$$\text{Cost per die} = \frac{\text{Cost per wafer}}{\text{Dies per wafer} \times \text{Yield}}$$

$$\text{Dies per wafer} \approx \text{Wafer area}/\text{Die area}$$

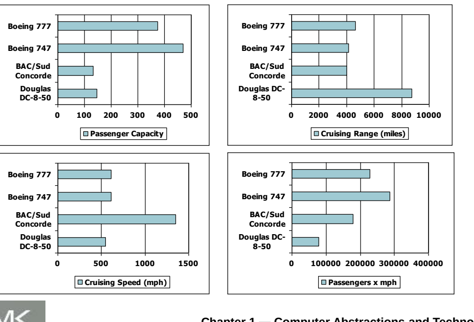
$$\text{Yield} = \frac{1}{(1 + (\text{Defects per area} \times \text{Die area}/2))^2}$$

- Nonlinear relation to area and defect rate
  - Wafer cost and area are fixed
  - Defect rate determined by manufacturing process
  - Die area determined by architecture and circuit design

Defining Performance

§1.6 Performance

- Which airplane has the best performance?



Response Time and Throughput

- Response time
  - How long it takes to do a task
- Throughput
  - Total work done per unit time
    e.g., tasks/transactions/… per hour
- How are response time and throughput affected by
  - Replacing the processor with a faster version?
  - Adding more processors?
- We'll focus on response time for now…

Relative Performance

- Define Performance = 1/Execution Time
- "X is n times faster than Y"

$$\text{Performance}_X / \text{Performance}_Y = \text{Execution time}_Y / \text{Execution time}_X = n$$

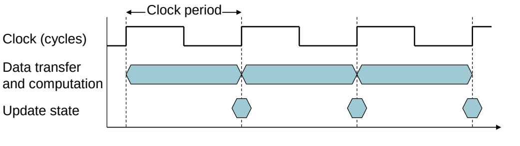
- Example: time taken to run a program
  - 10s on A, 15s on B
  - Execution Time$_B$ / Execution Time$_A$ = 15s / 10s = 1.5 = 1½
  - So A is 1½ times faster than B

Measuring Execution Time

- Elapsed time
  - Total response time, including all aspects
    - Processing, I/O, OS overhead, idle time
  - Determines system performance
- CPU time
  - Time spent processing a given job
    - Minus I/O time, other jobs' shares
  - Includes user CPU time and system CPU time
  - Different programs are affected differently by CPU and system performance
    - Running on servers – I/O performance – hardware and software
    - Total elapsed time is of interest
    - Define performance metric and then proceed

CPU Clocking

- Operation of digital hardware governed by a constant-rate clock



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

CPU Time

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

- Performance can be improved by
  - Reducing <u>number</u> of clock cycles
  - Increasing clock rate
  - Hardware designer must often trade off clock rate against cycle count

CPU Time Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B
  - Aim for 6s CPU time
  - Can do faster clock, but causes 1.2 × clock cycles
- How fast must Computer B clock be?

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

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$
$$= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$$

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

Instruction Count and CPI

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles Per Instruction}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time} = \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

- Instruction Count for a program
  - Determined by program, ISA, and compiler
- Average cycles per instruction
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI gets affected by instruction mix (dynamic frequency of instructions)

CPI Example

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same ISA
- Which is faster? by how much?

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

A is faster…

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

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{I \times 600\text{ps}}{I \times 500\text{ps}} = 1.2$$

…by this much

CPI in More Detail

- If different instruction classes take different numbers of cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

- Weighted average CPI

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}} = \sum_{i=1}^{n} \left(\text{CPI}_i \times \frac{\text{Instruction Count}_i}{\text{Instruction Count}} \right)$$

Relative frequency

CPI Example

- Alternative compiled code sequences using instructions in classes A, B, C

Class	A	B	C
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

- Sequence 1: IC = 5
  - Clock Cycles = 2×1 + 1×2 + 2×3 = 10
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles = 4×1 + 1×2 + 1×3 = 9
  - Avg. CPI = 9/6 = 1.5

Performance Summary

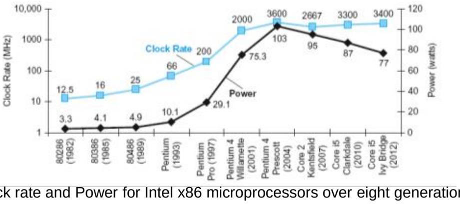
The BIG Picture

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

- Performance depends on
  - Algorithm: affects IC, possibly CPI
  - Programming language: affects IC, CPI
  - Compiler: affects IC, CPI
  - Instruction set architecture: affects IC, CPI, T_c

Power Trends

§1.7 The Power Wall



Clock rate and Power for Intel x86 microprocessors over eight generations

- In CMOS IC technology

$$\text{Power} = \text{Capacitive load} \times \text{Voltage}^2 \times \text{Frequency}$$

×30 | 5V → 1V | ×1000

Reducing Power

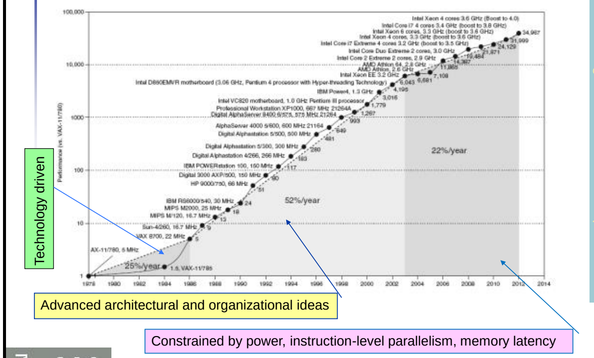
- Suppose a new CPU has
  - 85% of capacitive load of old CPU
  - 15% voltage and 15% frequency reduction

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

- The power wall
  - We can't reduce voltage further
  - We can't remove more heat
- How else can we improve performance?

Uniprocessor Performance

§1.8 The Sea Change: The Switch to Multiprocessors



Multiprocessors

- Multicore microprocessors
  - More than one processor <u>per chip</u>
- Requires <u>explicitly</u> parallel programming
  - Compare with instruction level parallelism
    - Hardware executes multiple instructions at once
    - Hidden from the programmer
  - Hard to do (Why?)
    - Programming for performance
    - Load balancing
    - Optimizing communication and synchronization

Concluding Remarks

§1.9 Concluding Remarks

- Cost/performance is improving
  - Due to underlying technology development
- Hierarchical layers of abstraction
  - In both hardware and software
- Instruction set architecture
  - The hardware/software interface
- Execution time: the best performance measure
- Power is a limiting factor
  - Use parallelism to improve performance

Acknowledgement

The slides are adopted from Computer Organization and Design, 5th Edition
by David A. Patterson and John L. Hennessy
2014, published by MK (Elsevier)