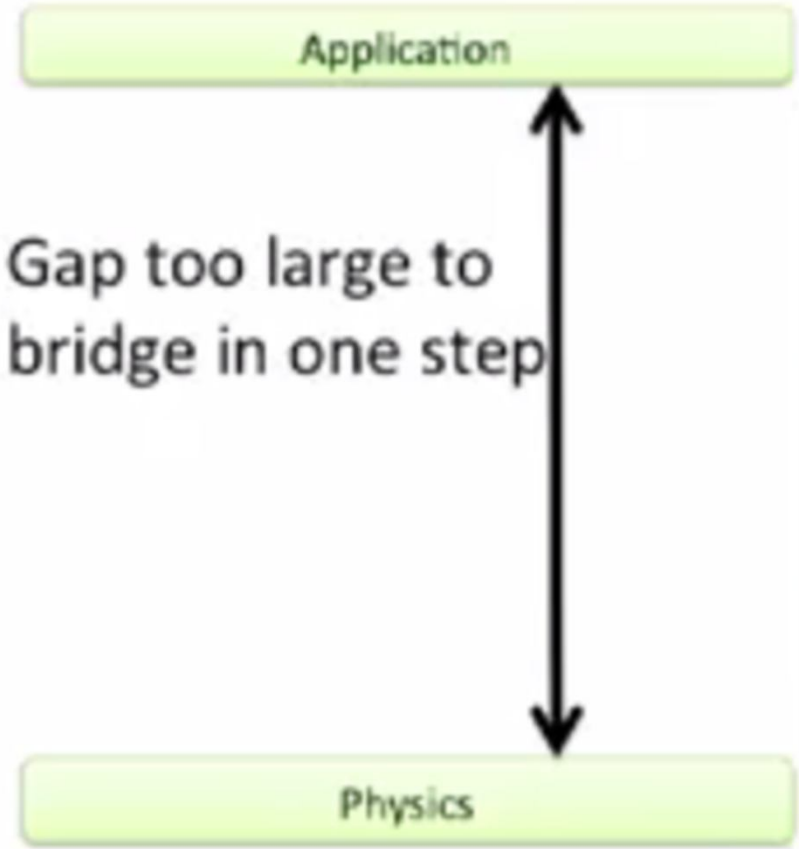
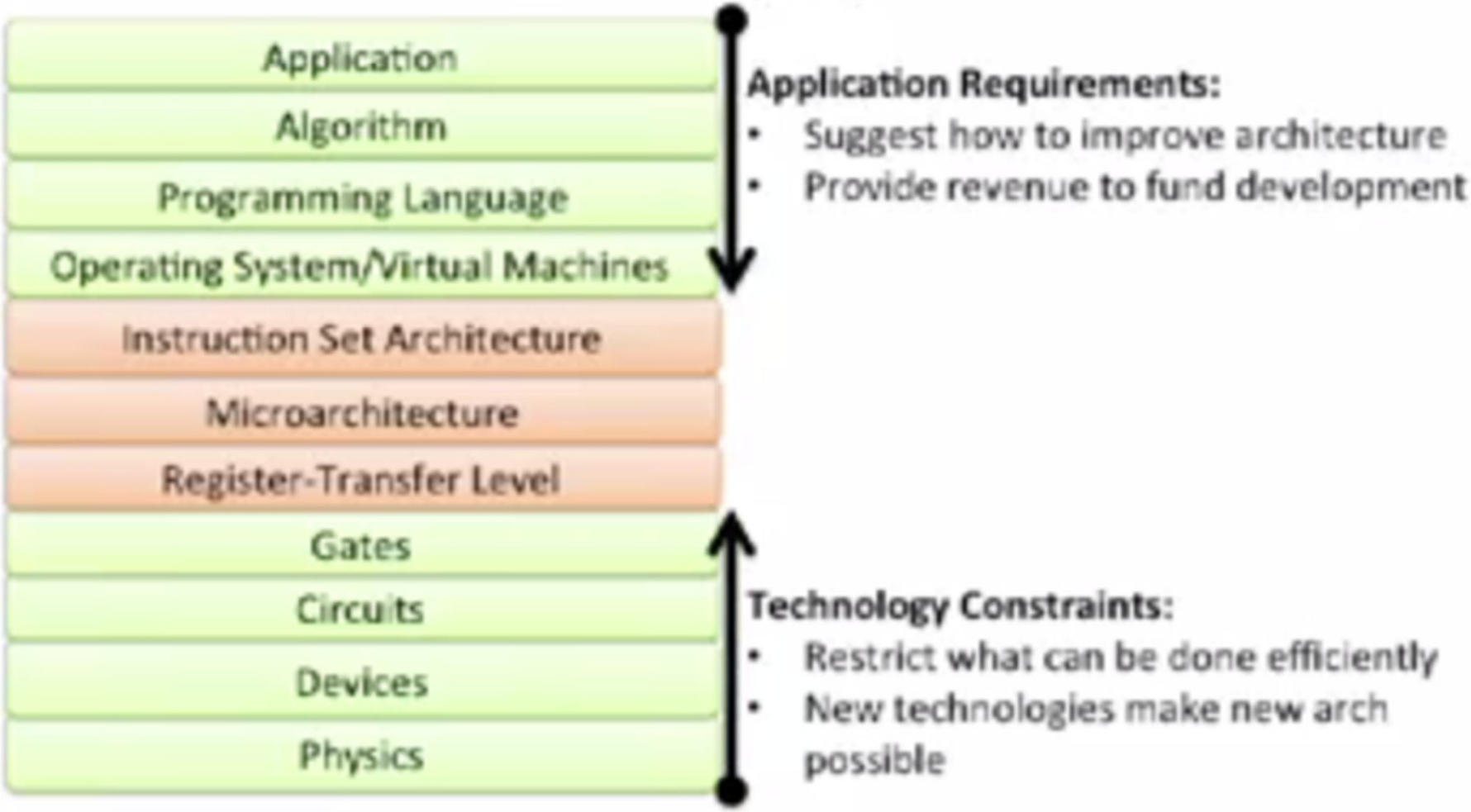
What is Computer Architecture?



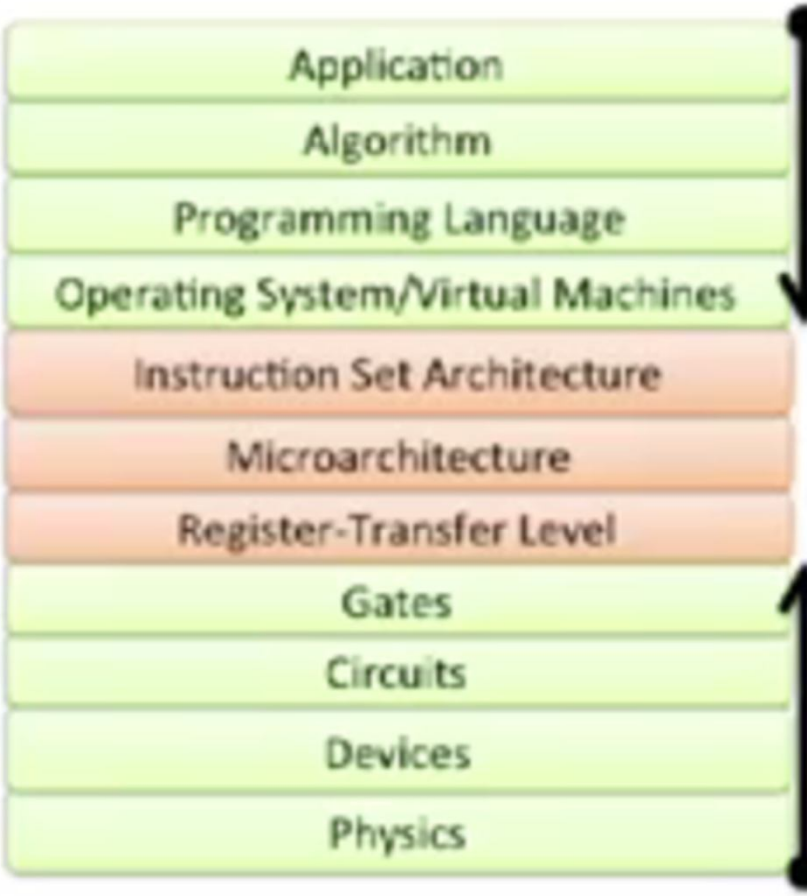
In its broadest definition, computer architecture is the design of the abstraction/ implementation layers that allow us to execute information processing applications efficiently using manufacturing technologies

Abstractions in Modern Computing Systems

Computer Architecture is Constantly Changing



Computer Architecture is Constantly Changing



Application Requirements:

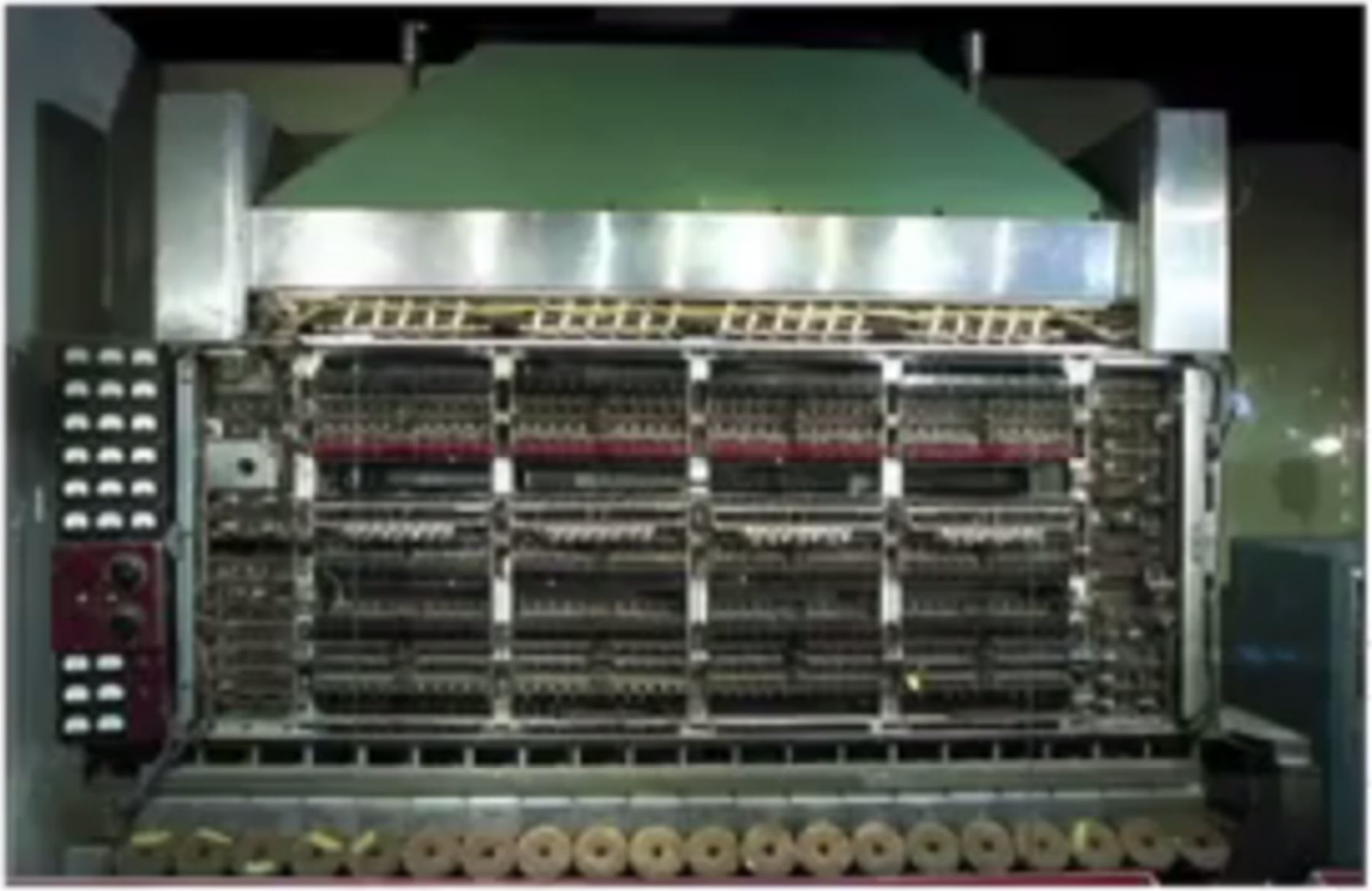
- Suggest how to improve architecture
- Provide revenue to fund development

Architecture provides feedback to guide application and technology research directions

Technology Constraints:

- Restrict what can be done efficiently
- New technologies make new arch possible

Computers Then…



IAS Machine. Design directed by John von Neumann.
First booted in Princeton NJ in 1952
Smithsonian Institution Archives (Smithsonian Image 95-06151)

Computers Now

- Sensor Networks
- Cameras
- Smartphones
- Mobile Audio Players
- Laptops
- Autonomous Cars
- Servers
- Game Players
- Routers
- Flying UAVs
- GPS
- eBooks
- Tablets
- Set-top Boxes

Architecture vs. Microarchitecture

"Architecture"/Instruction Set Architecture:

- Programmer visible state (Memory & Register)
- Operations (Instructions and how they work)
- Execution Semantics (interrupts)
- Input/Output
- Data Types/Sizes

Microarchitecture/Organization:

- Tradeoffs on how to implement ISA for some metric (Speed, Energy, Cost)
- Examples: Pipeline depth, number of pipelines, cache size, silicon area, peak power, execution ordering, bus widths, ALU widths

Computer Architecture

Logical aspects of system implementation as seen by the programmer; such as, instruction sets (ISA) and formats, opcode, data types, addressing modes and I/O.

Instruction set architecture (ISA) is different from "microarchitecture", which consist of various processor design techniques used to implement the instruction set.

Computers with different microarchitectures can share a common instruction set.

For example, the Intel Pentium and the AMD Athlon implement nearly identical versions of the x86 instruction set, but have radically different internal designs.

Computer architecture is the conceptual design and fundamental operational structure of a computer system. It is a **functional description** of requirements and design implementations for the various parts of a computer.

It is the science and art of selecting and interconnecting hardware components to create computers that meet functional, performance and cost goals.

It deals with the **architectural attributes** like physical address memory, CPU and how they should be designed and **made to coordinate with each other** keeping the goals in mind.

Analogy: "building the design and architecture of house" – architecture may take more time due to planning and then organization is building house by bricks or by latest technology keeping the basic layout and architecture of house in mind.

Computer architecture comes before computer organization.

Computer organization (CO) is how operational attributes are linked together and contribute to realise the architectural specifications.

CO encompasses all physical aspects of computer systems

e.g. Circuit design, control signals, memory types.

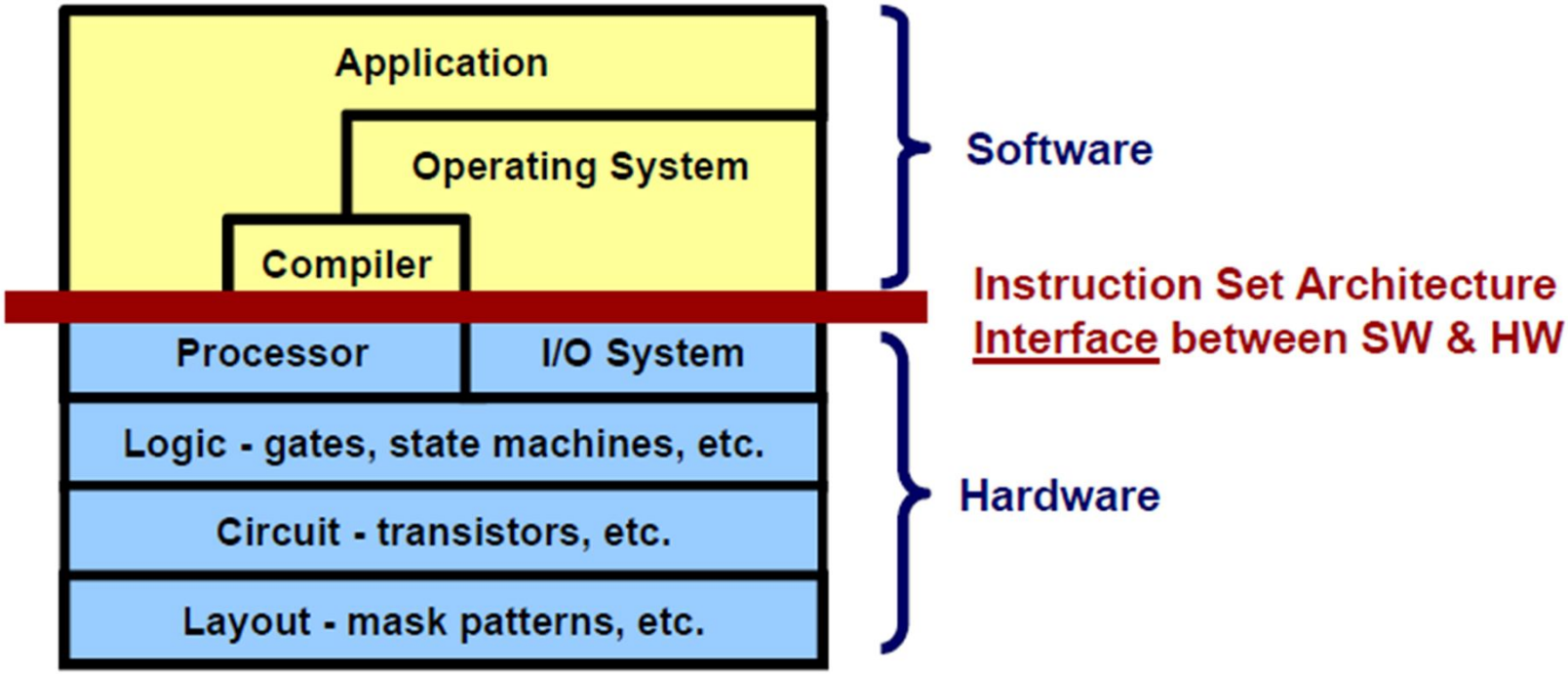
Microarchitecture, also known as **Computer organization** is a lower level, more concrete and detailed, description of the system that involves how the **constituent parts of the system are interconnected and how they interoperate** in order to implement the ISA.

The size of a computer's cache, for example, is an organizational issue that generally has nothing to do with the ISA.

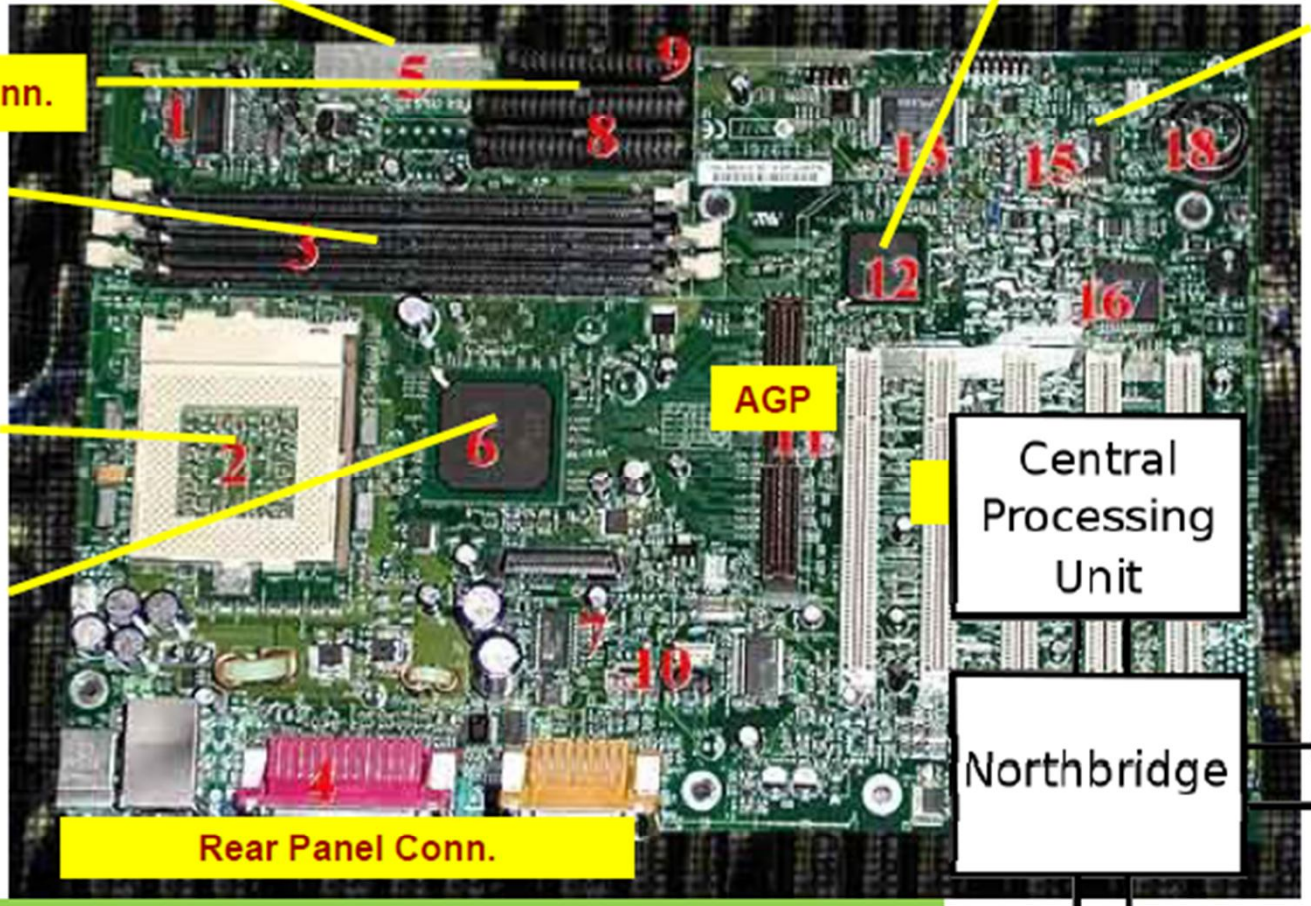
Another example: it is an architectural design issue whether a computer will have a **multiply** instruction. It is an organizational issue whether that instruction will be implemented by a special **multiply unit** or by a mechanism that makes repeated use of the **add unit** of the system.

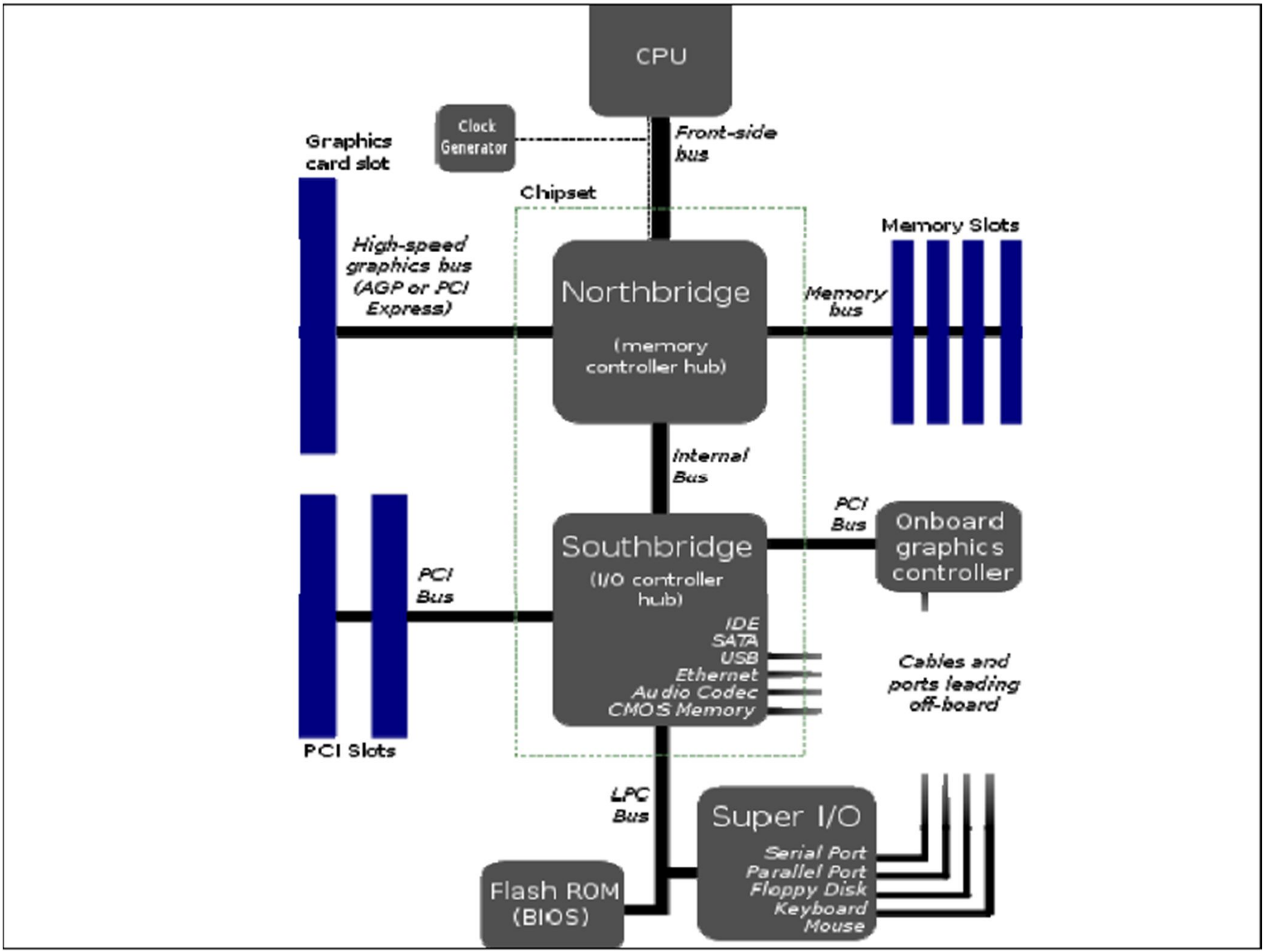
Instruction Set Architecture (ISA) - The Hardware-Software Interface

- The most important abstraction of computer design



Typical Motherboard (Pentium III)




CPU
Clock Generator
Front-side bus
Graphics card slot
Chipset
Memory Slots
High-speed graphics bus (AGP or PCI Express)
Northbridge
(memory controller hub)
Memory bus
Internal Bus
PCI Bus
Onboard graphics controller
Southbridge
(I/O controller hub)
PCI Bus
IDE
SATA
USB
Ethernet
Audio Codec
CMOS Memory
Cables and ports leading off-board
PCI Slots
LPC Bus
Super I/O
Serial Port
Parallel Port
Floppy Disk
Keyboard
Mouse
Flash ROM (BIOS)

What is Computer Architecture?

Figure 1: Courtesy: www.psychology-today.com

Computer Architecture

- The CPU is the brain of a computer system.
- It works both consciously and subconsciously.
- Consciously : Executes a program
- Sub-consciously : Runs the operating system, co-ordinates with I/O devices

Computer Architecture : Study of the CPU and the peripherals

Where does it fit in?

Figure 2: courtesy: www.coolnerds.com

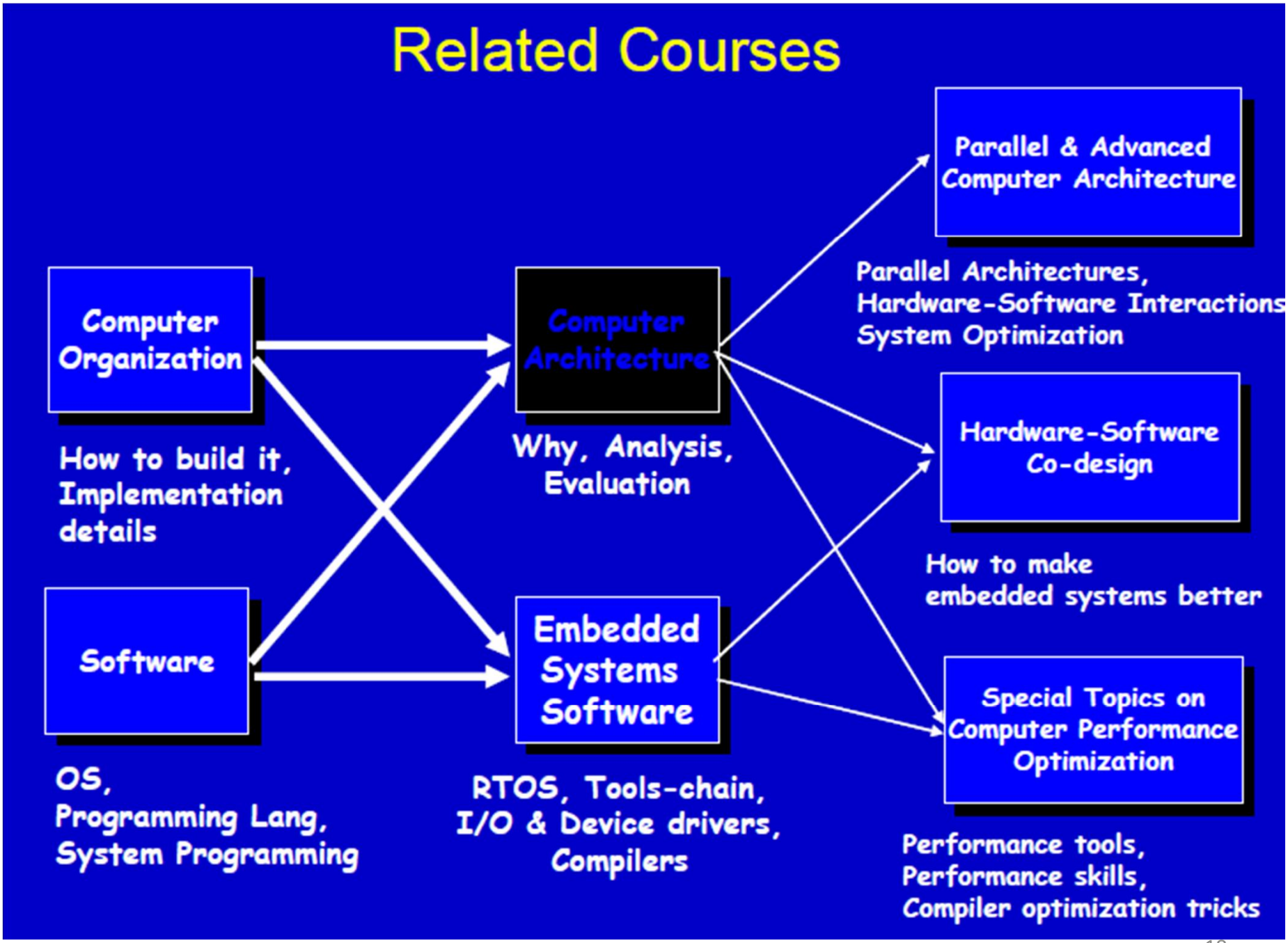
Example

- Computer Architecture –> Brain
- Networking –> Nervous and Circulatory System
- Computer Vision –> Eyes
- Operating System–> Endocrine and Immune System
- Databases –> Memory
- Algorithms –> Intelligence
- Prog. Languages –> Linguistic Center
- ...

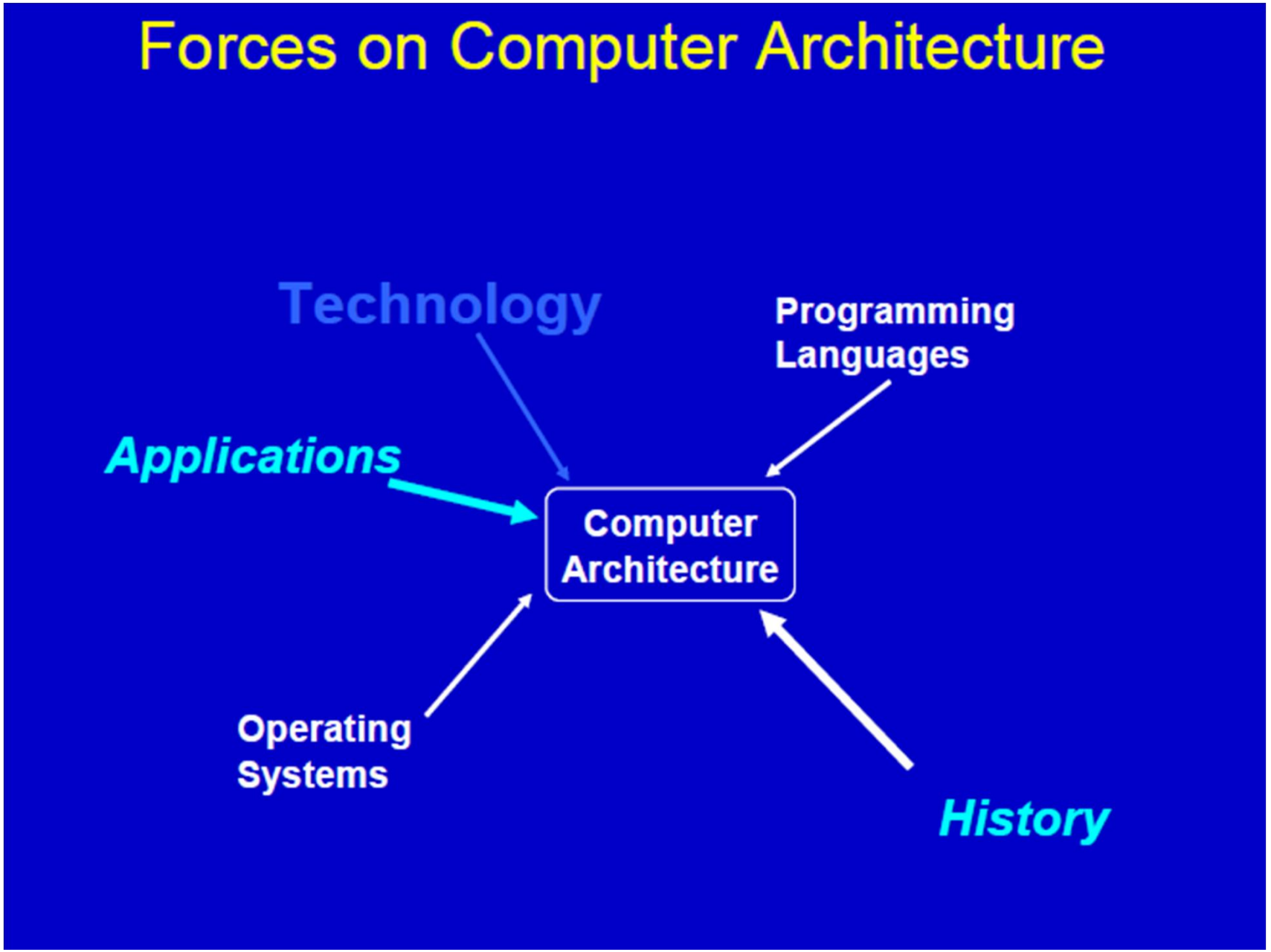
Why Study Computer Architecture?

- Understanding
  - Learn the inner workings of processors
  - Understand hardware/software interaction
  - Design better operating systems and compilers
- Career Prospects
  - Companies directly working in architecture
    - Intel, AMD, Sun/Oracle, Arm, IBM
  - Systems Software
    - Google, Samsung, VMWare, Wind River, McAfee
- Higher Studies ...

Related Courses

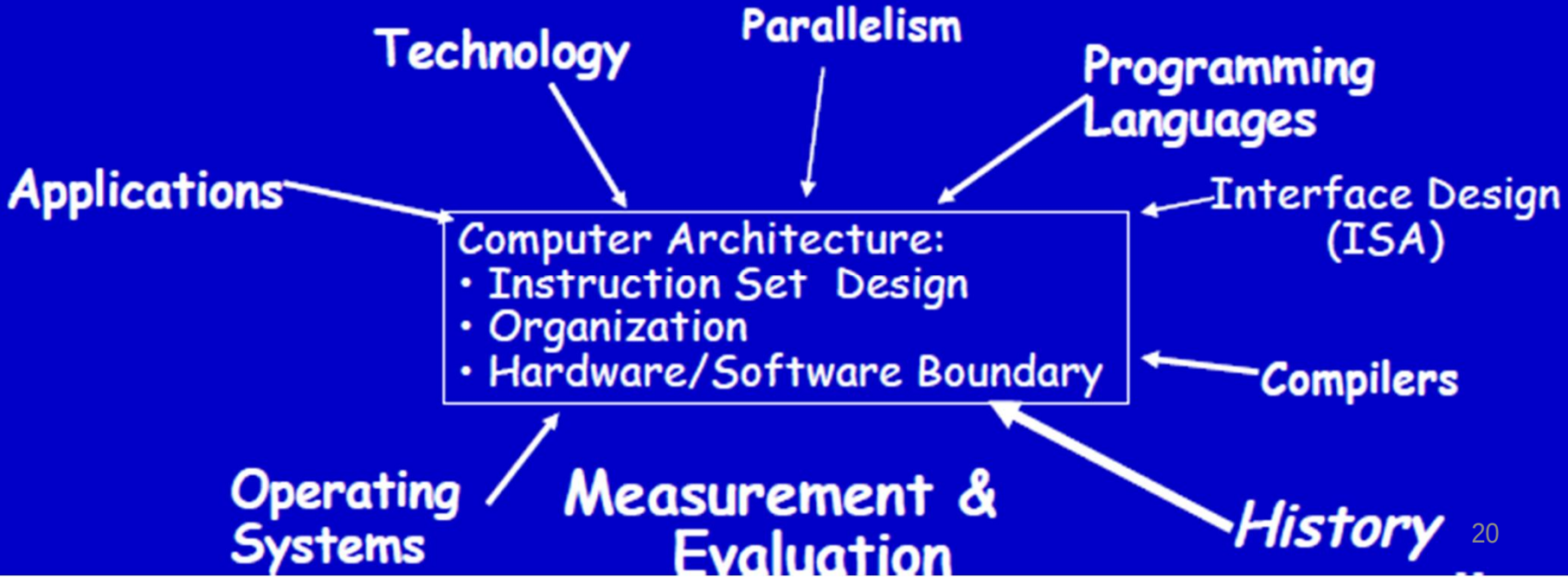


Forces on Computer Architecture



Course Focus

Understanding the design techniques, machine structures, technology factors, evaluation methods that will determine the form of computers in 21st Century



Input/ Output Unit Overview

- Input units
  - Keyboard, mouse, microphone, CDROM, etc.
- Output units
  - Graphical display, printer, etc.
- The collective term input/ output (I/O) units
  - Input units, output units, disk drives, etc.

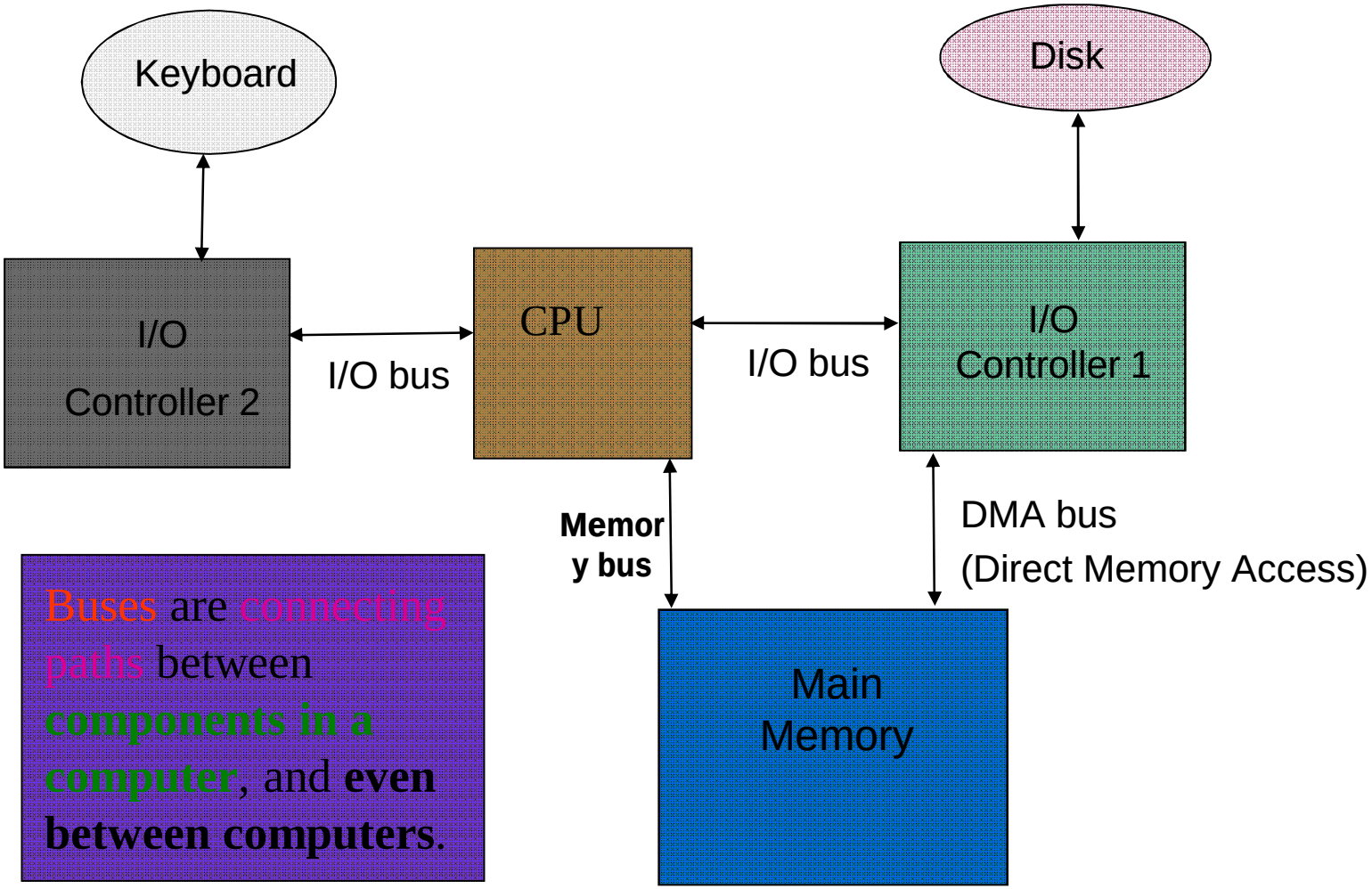
Memory Unit Overview

- Memory is used to store **programs** and **data**
- [Low-level] Unit of access is an **n**-bit word
  - Unique location is its address
  - Retrieval is in units of words
  - Commonly 32-bit today, moving to 64-bits
  - Typically 16-bit – 64-bit machines nowadays
- Primary storage: random-access memory (RAM)
- Secondary storage: hard disk, CDROM, etc.

Processor Overview

- Registers
  - Small but fast storage of intermediate values in a computation
- Arithmetic logic unit (ALU): performs computations
  - e.g. arithmetic operations: add, subtract, multiply, divide, etc.
  - e.g. logical operations: and, or, not, xor, etc.
  - c.f. calculator
  - Operands taken from registers
- Control
  - Orchestrates the transfer of data and sequencing of operations between memory, registers, ALU, I/O devices

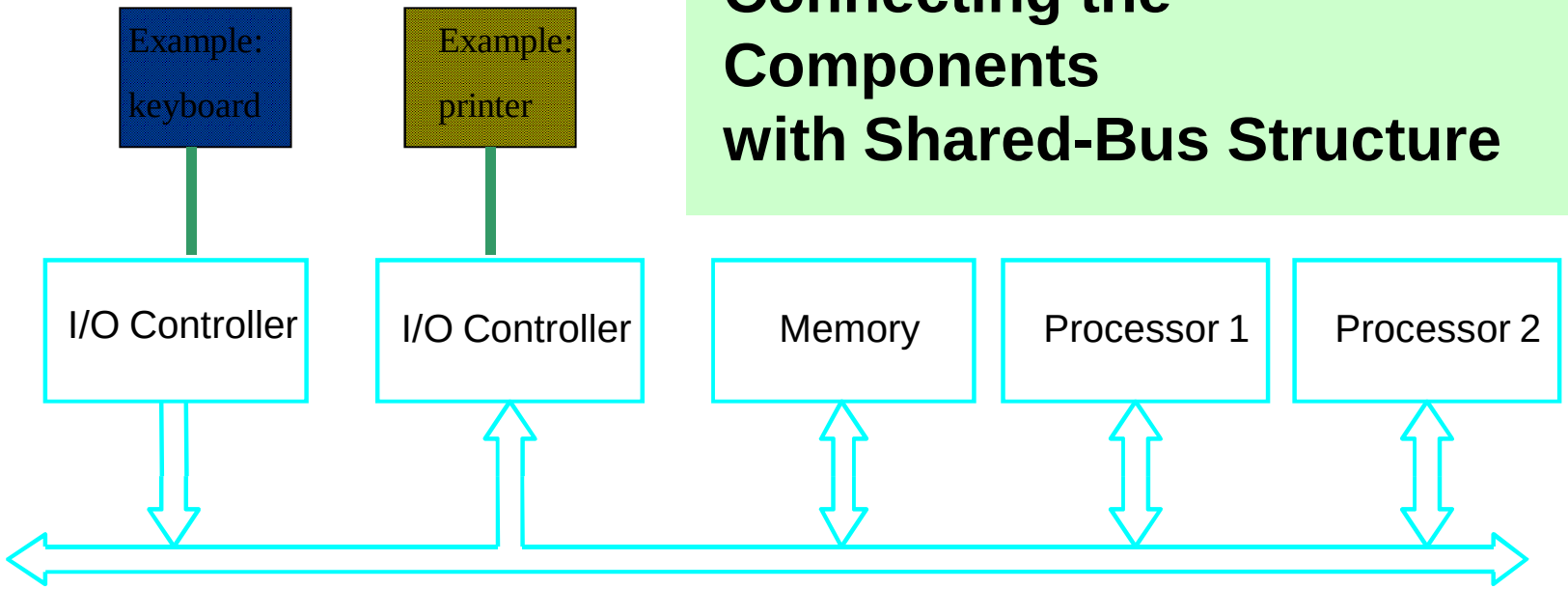
Connecting the Components with Dedicated/ Multiple Buses



Buses are connecting paths between **components in a computer**, and **even between computers**.

http://en.wikipedia.org/wiki/Computer_bus

Connecting the Components with Shared-Bus Structure



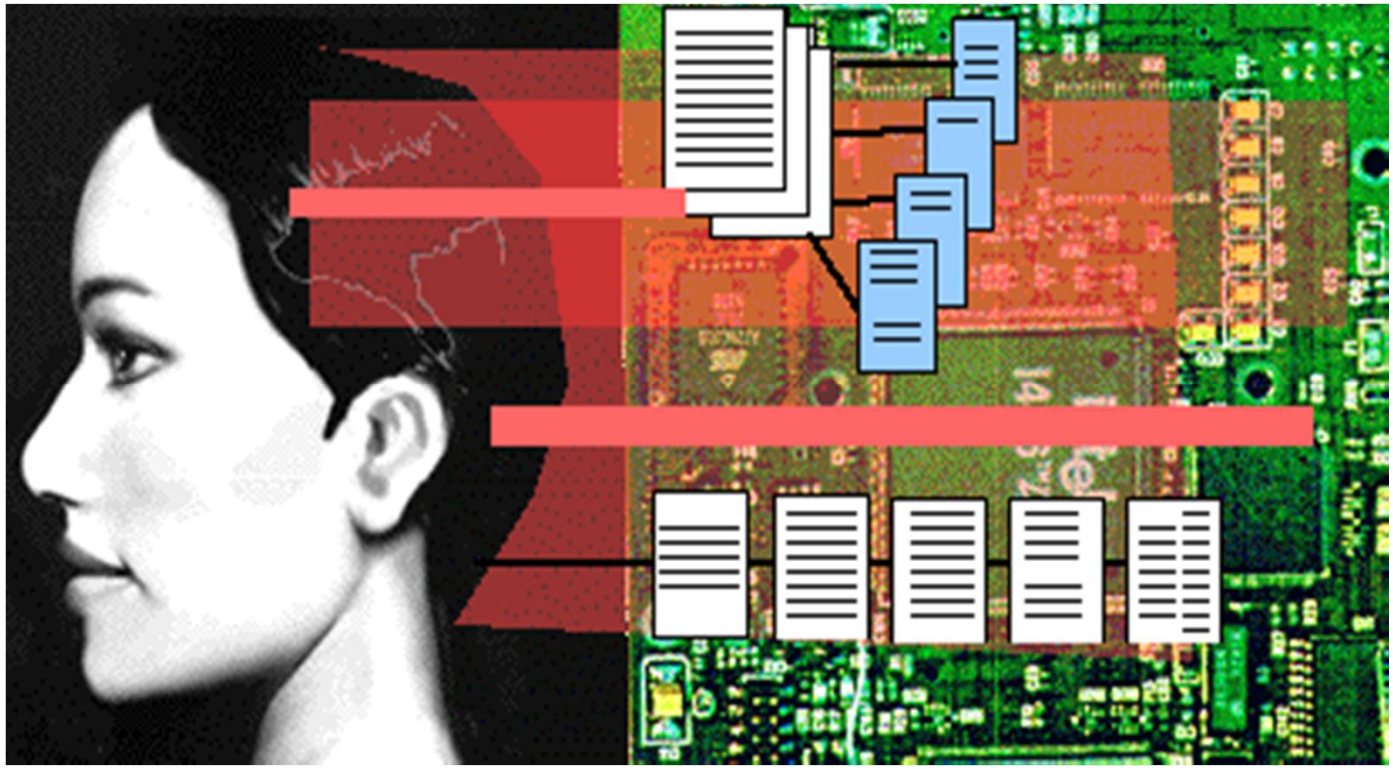
Information carried along a bus: address, data , control

- **All devices have same address structure**
- **All devices can be controlled by common machine instructions**
- **Only two devices can do data communication simultaneously**

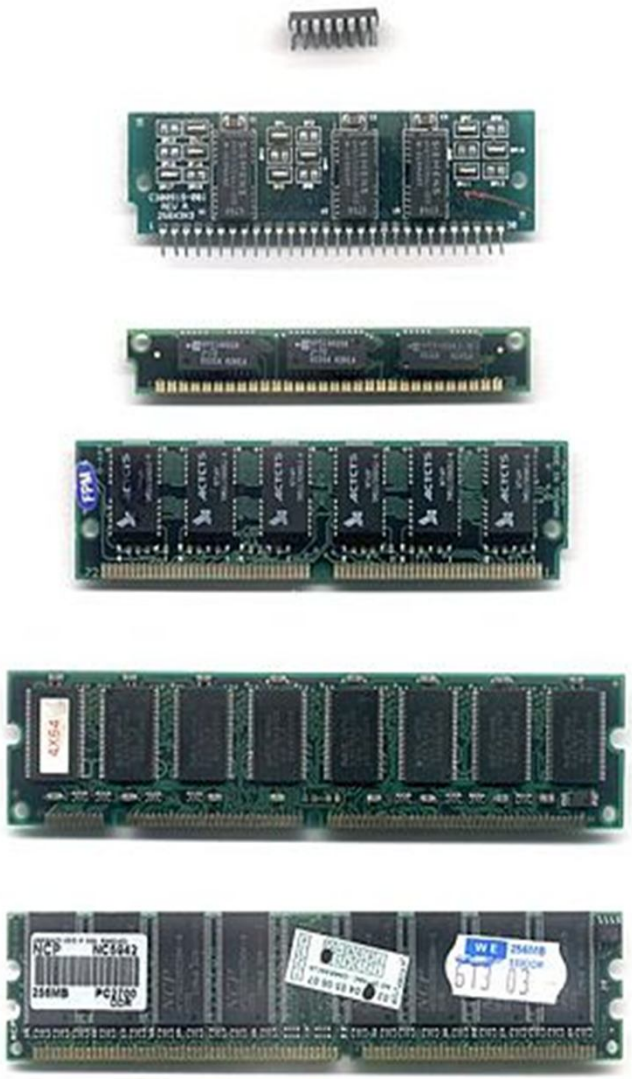
Bus

- Allows computer to be customized for different applications e.g. different peripherals
- Design criteria – speed, cost, etc.
- Word length can be different depending on application
  - E.g. USB is serial (1-bit), PCI bus is 32-bit

Computer Memory



Memory Hardware: Chips and Modules



Digital (Binary) Memory

- **Primary storage, main storage, main memory**
- **Fast Access (when compared to I/O)**

Units

bit (1 binary digit, a value of 0 or 1)

Byte (1 byte = 8 bits)

Word

Manipulation of data by CPU is in words.
A single access results in one word of data being transferred.

Word length is specified in number of bits/ bytes:
for example, 8-bit, 16-bit, 32-bit, 64-bit, 4-byte, etc.

Main Memory (MM) Organization

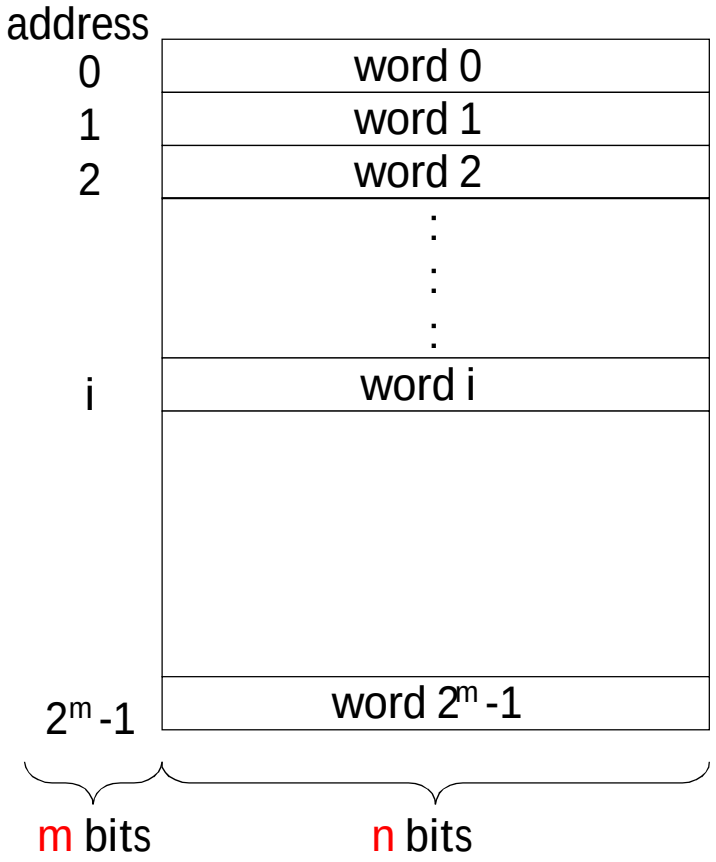
•In a main memory with m-bit addresses,
0 to 2^m-1 words are available.

•Each word stores n bits

•m, n are independent

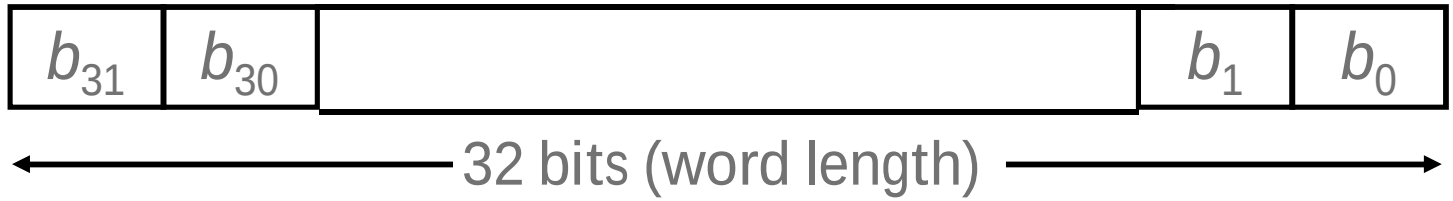
•m specifies the number of units;
n specifies the unit size

•What is the total number of bits?



Memory: Contents of a Word (I)

Here is an example of a 32-bit word.



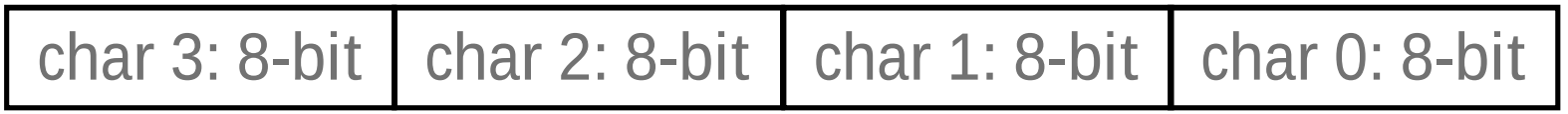
A word can store information. For example,

1. Four English characters, each encoded in a common 8-bit code

ASCII: American Standard Code for Information Interchange

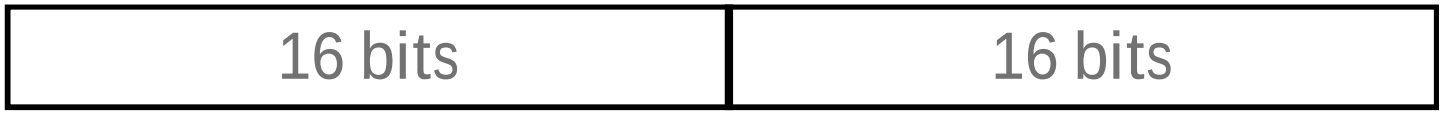
or

EBCDIC: Extended Binary Coded Decimal Interchange Code



Memory: Contents of a Word (II)

2. Two Chinese characters



Big5/GB code

3. A 32-bit Signed integer

sign bit

b_{31}=0 means positive integer, b_{31}=1 means negative integer

magnitude = $b_{30} \cdot 2^{30} + \ldots + b_0 \cdot 2^0$

4. A 32-bit machine instruction

operation code

address field

address field

Addresses

- Use addresses to store or retrieve a single item of information
- For some k, memory consists of 2^k unique addresses which range from 0 – (2^k-1)
  - The ***possible*** addresses are the address space of the computer
  - e.g. 28-bit address → 2^{28} = 268435456 locations
    - Talk about word address: we use words
    - Talk about byte address: we use bytes
    - Talk about memory size: we use bytes/ sometimes bits

Byte addresses

- Information quantities: bit, byte, word
- 1 Byte = 8 bits
- Word typically varies 16-64 bits (the IA-32 architecture has 32-bit words which we will assume from now on)

  Intel Architecture, 32-bit
- Most machines address memory in units of bytes (byte-addressable)
  - Implies for a 32-bit machine, successive words are at addresses 0, 4, 8, 12 …

More/ Less Significant Bytes

- Consider the hexadecimal (base 16) 32-bit number

  $12342A3F_{16}$

  $=1x16^7+2x16^6+3x16^5+4x16^4+2x16^3+\mathbf{10}x16^2+3x16^1+\mathbf{15}x16^0$

  $=305408575_{10}$

- This 32-bit number has four bytes 12, 34, 2A, 4F (4 bytes x 8 bits/byte = 32 bits)

- Bytes/bits with higher weighting are " more significant "
  - e.g. the byte 34 is more significant than 2A

- Bytes/bits with lower weighting are " less significant "

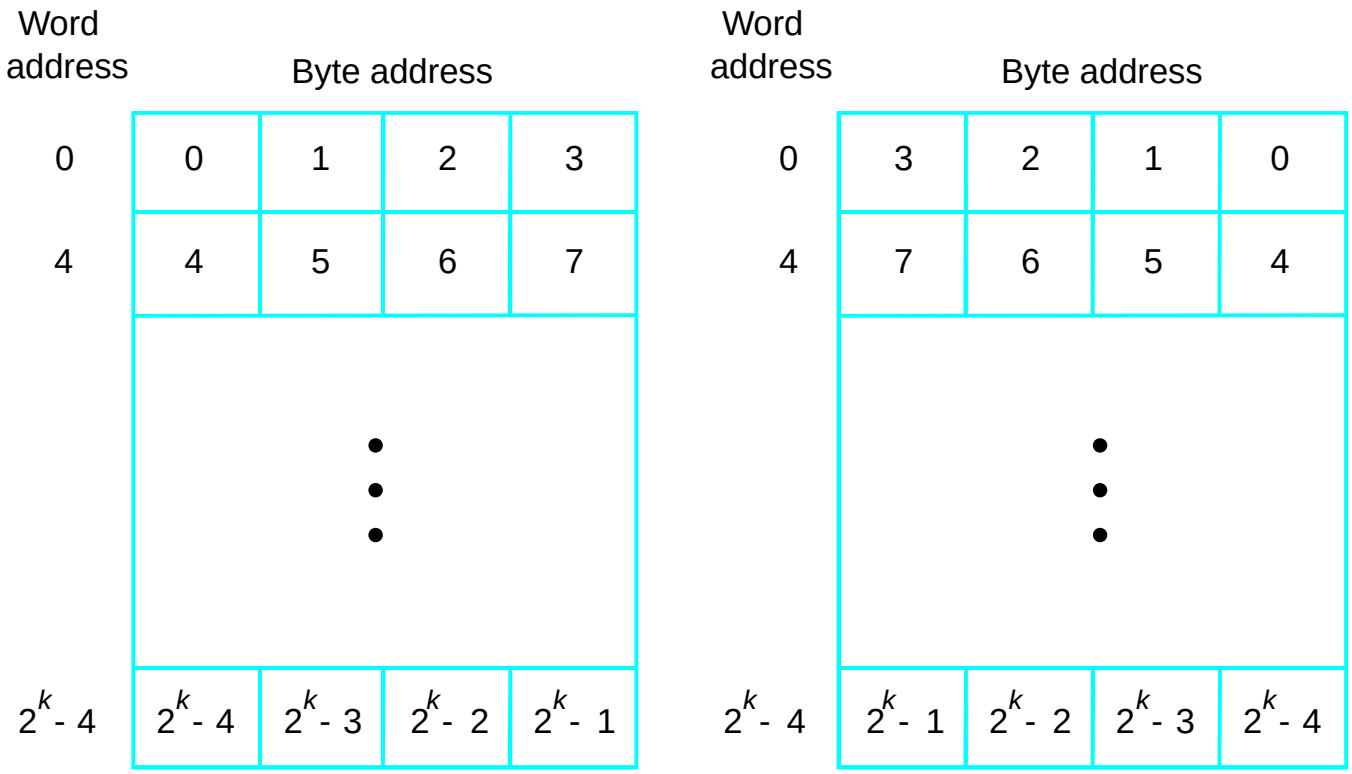
- We also use terms " most significant byte/ bit " and " least significant byte/ bit "

Big/ Little Endian

- Two ways byte addresses can be assigned/ arranged across words
  - more significant bytes first (big endian)
    - e.g. Motorola
  - less significant bytes first (little endian)
    - e.g. Intel

Big/ Little Endian

What would $12342A3F_{16}$ look like in these two endianship assignments?



(a) Big-endian assignment
e.g. Motorola

(b) Little-endian assignment
e.g. Intel

Assume k-bit address, 4-byte (32-bit) word

Word alignment

- 32-bit words ***align*** naturally at addresses 0, 4, 8 ...
  - These are aligned addresses

- Unaligned accesses are either not allowed or slower, why?
  - e.g. read a 32-bit word from address 00000001

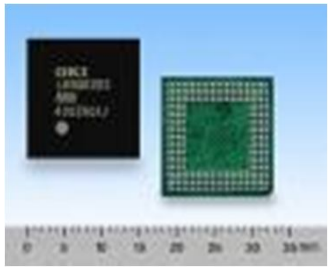
- What about for 16- and 64-bit words?

Central Processing Unit (CPU)

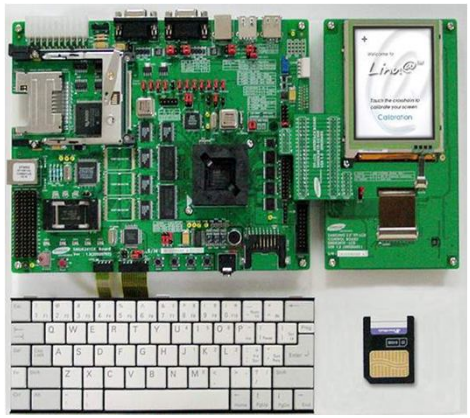
IBM Power PC

Intel Pentium

Digital signal processor IC



ARM 9



Some Intel CPUs

4004

8008

8080

8088/ 8086 [x86]

80186

80286 [286]

80386 [386]

80486 [486]

Pentium [586]

Pentium MMX

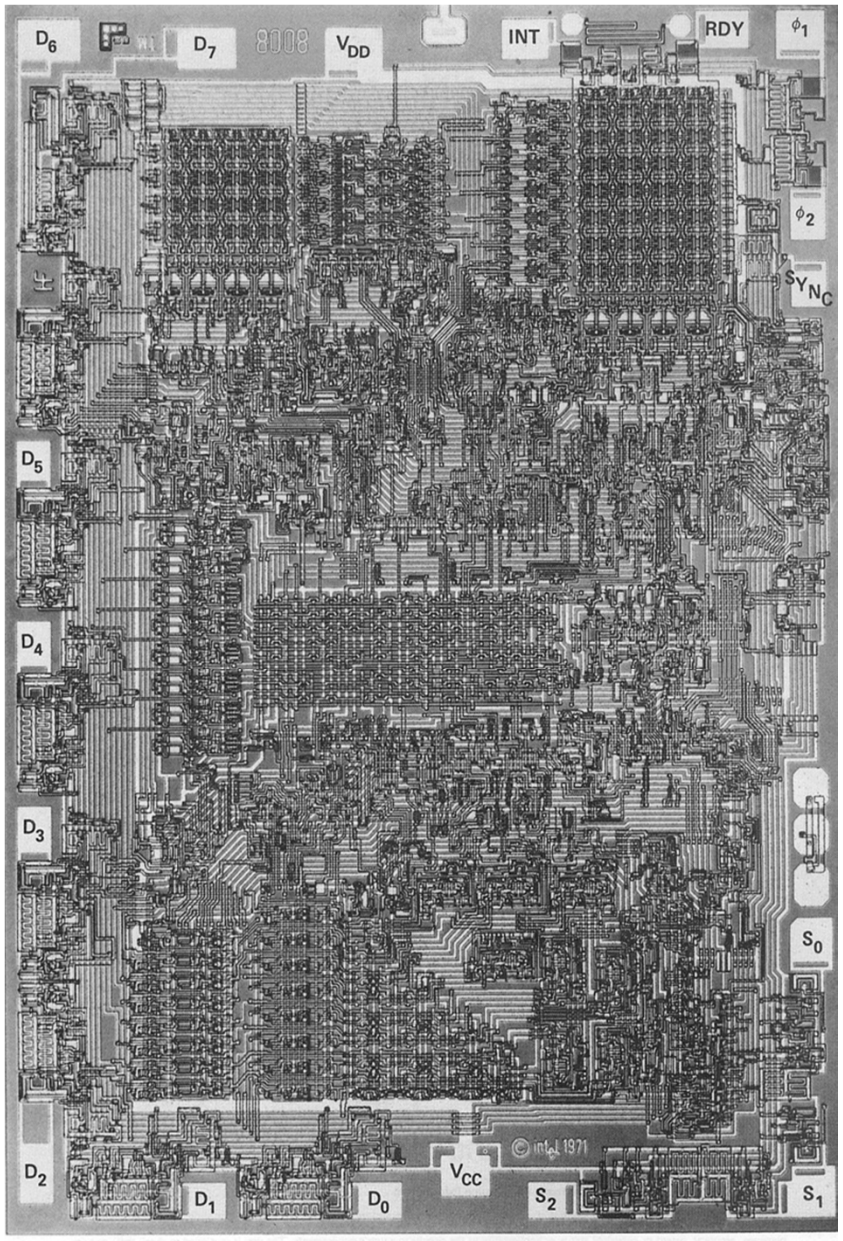
Pentium PRO [686]

Pentium II

Pentium III

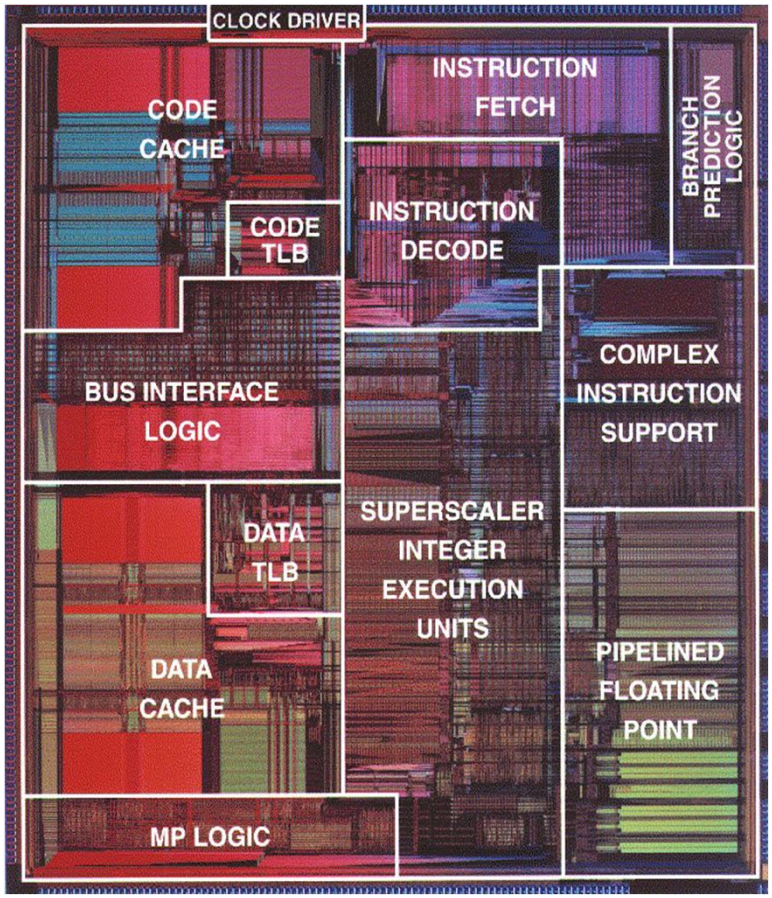
Pentium 4

Core2 Duo



8008 Photomicrograph With Pin Designations

CPU on a Chip → Microprocessor



http://micro.magnet.fsu.edu/chipshots/index.html

CPU

- What does a CPU do?

  It executes programs.

- What is a program?

  Program = Instruction + Data

- Where are the instructions and data?

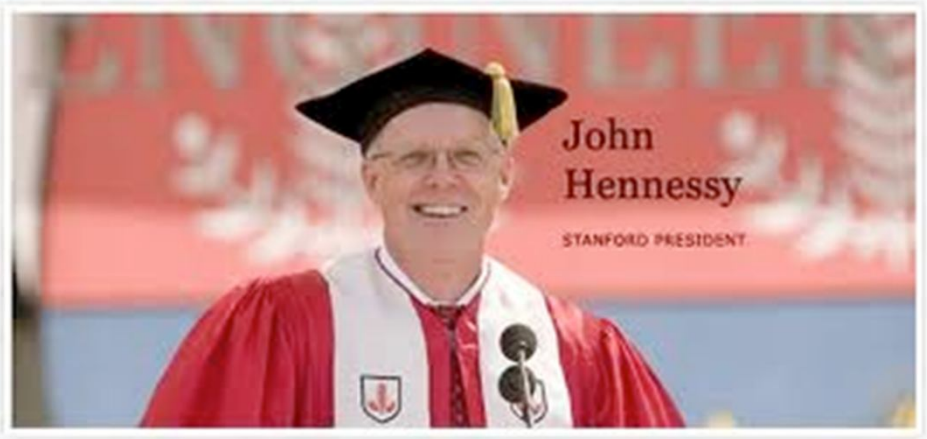
  In Memory – when they are not being processed

  In CPU – when they are being processed

Reduced Instruction Set Computer (RISC)

- Dave Patterson
  - RISC Project, 1982
  - UC Berkeley
  - RISC-I: ½ transistors & 3x faster
  - Influences: Sun SPARC, namesake of industry

- John L. Hennessy
  - MIPS, 1981
  - Stanford
  - Simple pipelining, keep full
  - Influences: MIPS computer system, PlayStation, Nintendo



Reduced Instruction Set Computer (RISC)

- MIPS Design Principles
- Simplicity favors regularity
  - 32 bit instructions
- Smaller is faster
  - Small register file
- Make the common case fast
  - Include support for constants
- Good design demands good compromises
  - Support for different type of interpretations/classes

Reduced Instruction Set Computer

- MIPS = Reduced Instruction Set Computer (RISC)
  - ≈ 200 instructions, 32 bits each, 3 formats
  - all operands in registers
    - almost all are 32 bits each
  - ≈ 1 addressing mode: Mem[reg + imm]

- x86 = Complex Instruction Set Computer (CISC)
  - > 1000 instructions, 1 to 15 bytes each
  - operands in dedicated registers, general purpose registers, memory, on stack, …
    - can be 1, 2, 4, 8 bytes, signed or unsigned
  - 10s of addressing modes
    - e.g. Mem[segment + reg + reg*scale + offset]

RISC vs CISC

- RISC Philosophy
- Regularity & simplicity
- Leaner means faster
- Optimize the common case

CISC Rebuttal

Compilers can be smart

Transistors are plentiful

Legacy is important

Code size counts

Micro-code!

- Energy efficiency
- Embedded Systems
- Phones/Tablets

Desktops/Servers