Computer Architecture

Instructor: Dr. Moaath shatnawi

William Stallings
Computer Organization
and Architecture

Chapter 1
Introduction

Architecture & Organization 1

- Architecture is those attributes visible to the programmer like logical execution.

- Instruction set, number of bits used for data type representation (number, characters), I/O mechanisms, addressing memory techniques.

- Organization is how features are implemented
  - Control signals, interfaces, memory technology.

Architecture & Organization 2

- All Intel x86 family share the same basic architecture.
- The IBM System/370 family share the same basic architecture.
- This gives code compatibility.
- Organization differs between different versions.

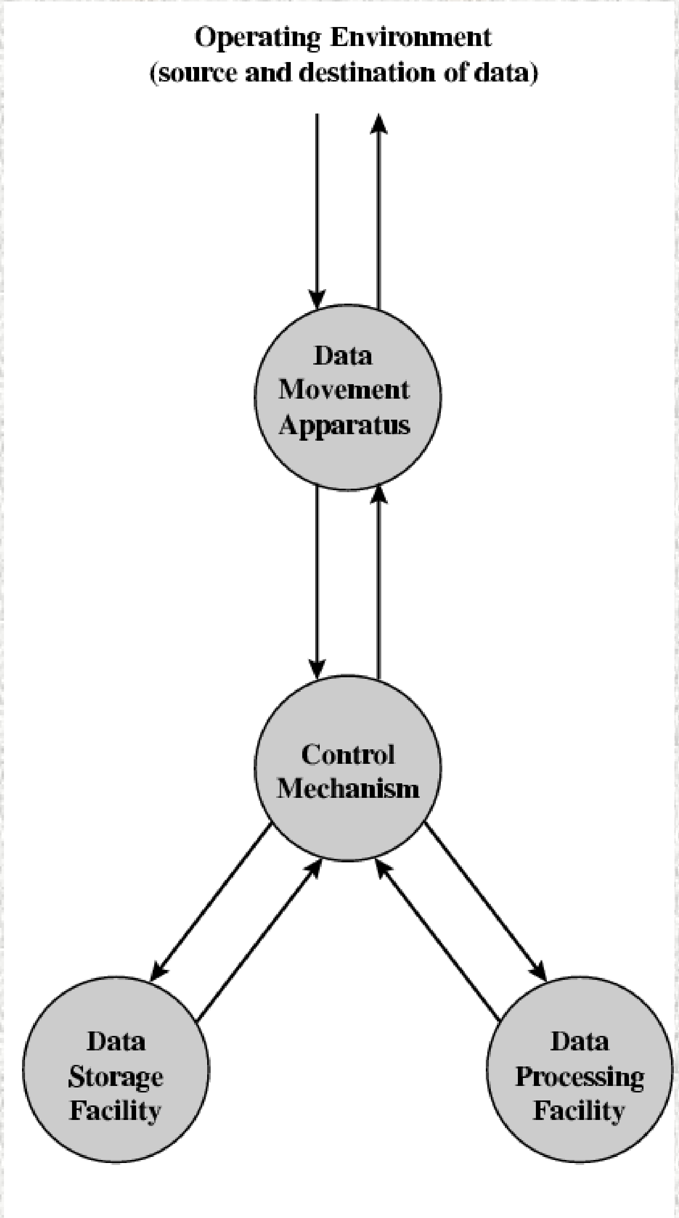
Structure & Function

- Structure is the way in which components relate to each other.
- Function is the operation of individual components as part of the structure.

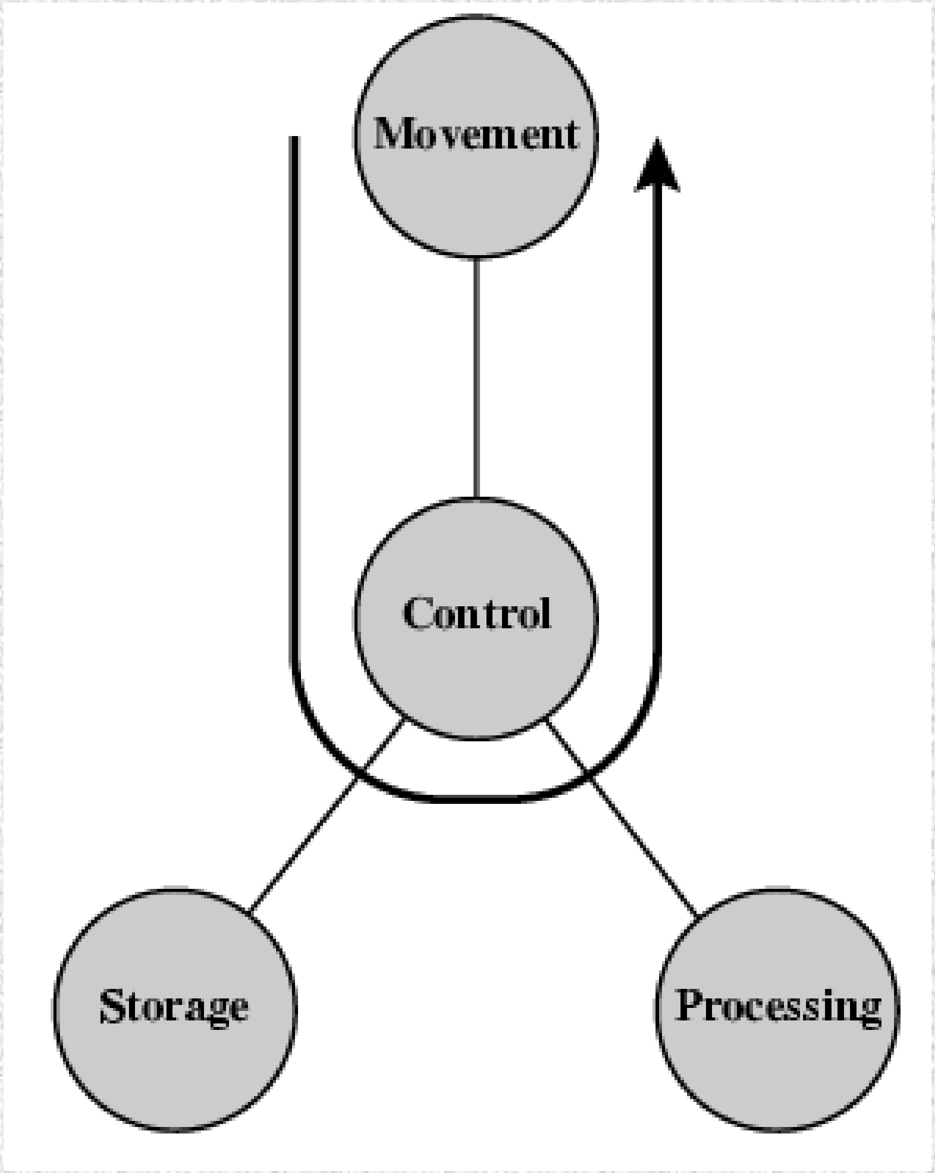
Function

- All computer functions are:
  - Data processing
  - Data storage
  - Data movement
  - Control

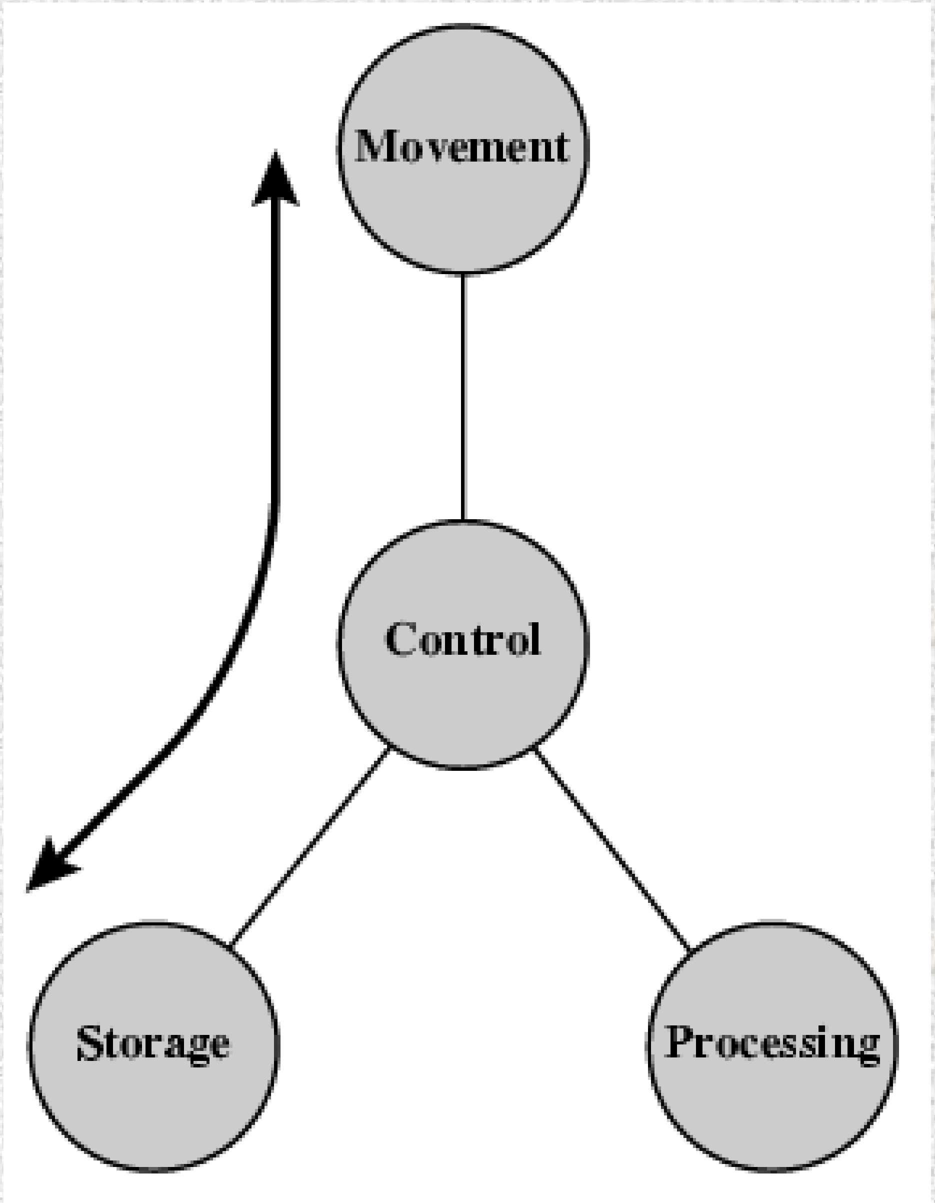
Functional view



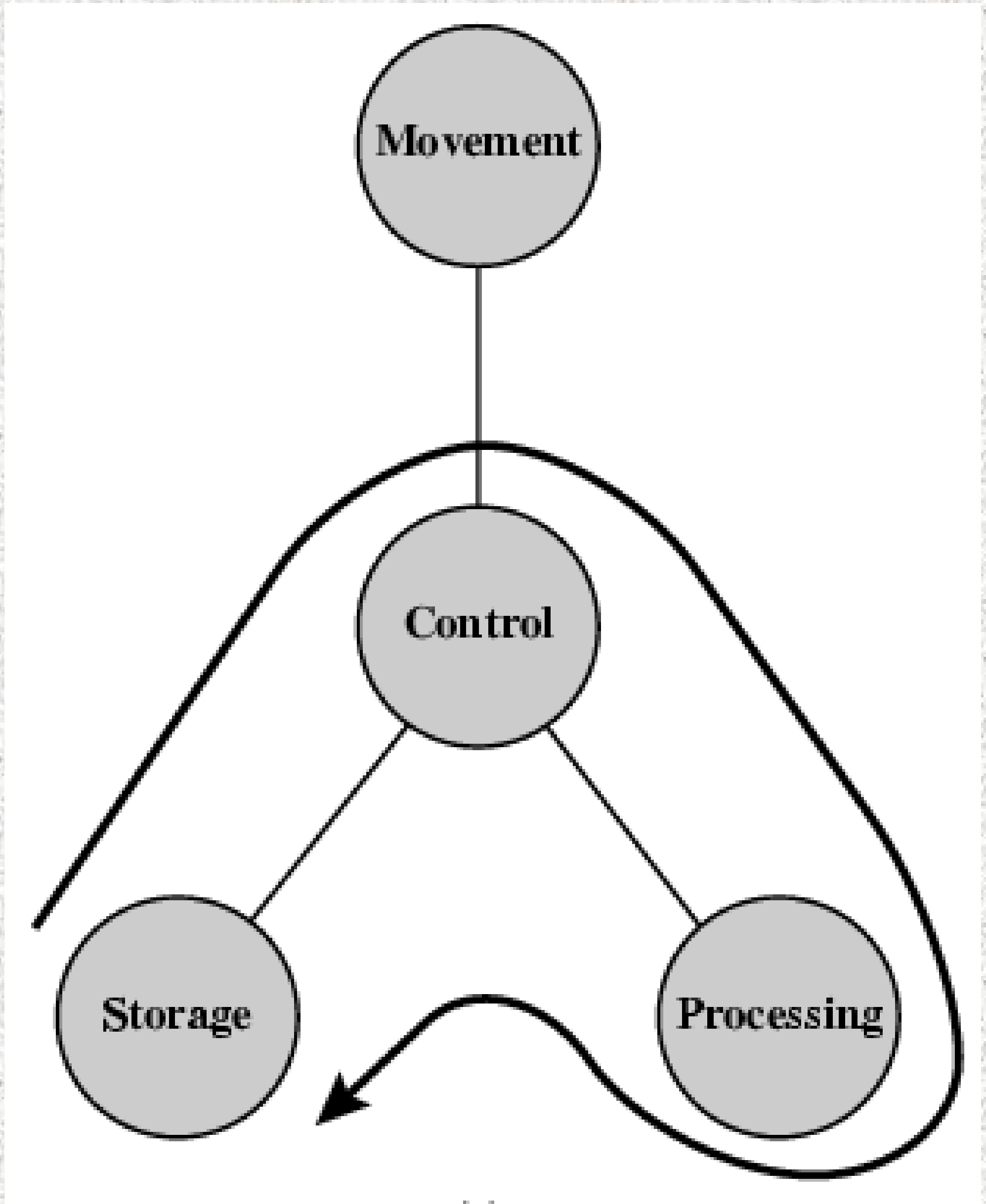
Operations (1) Data movement



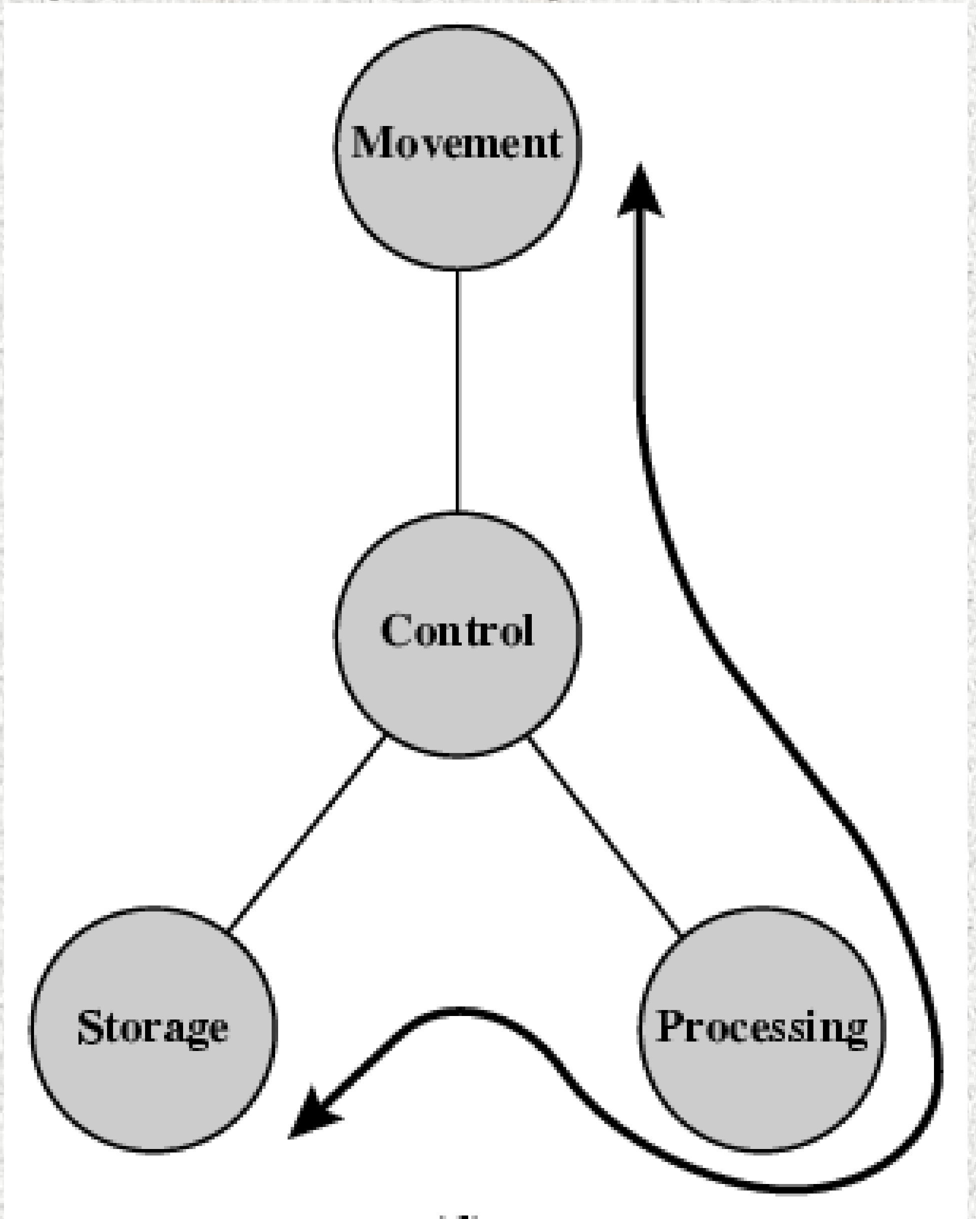
Operations (2) Storage



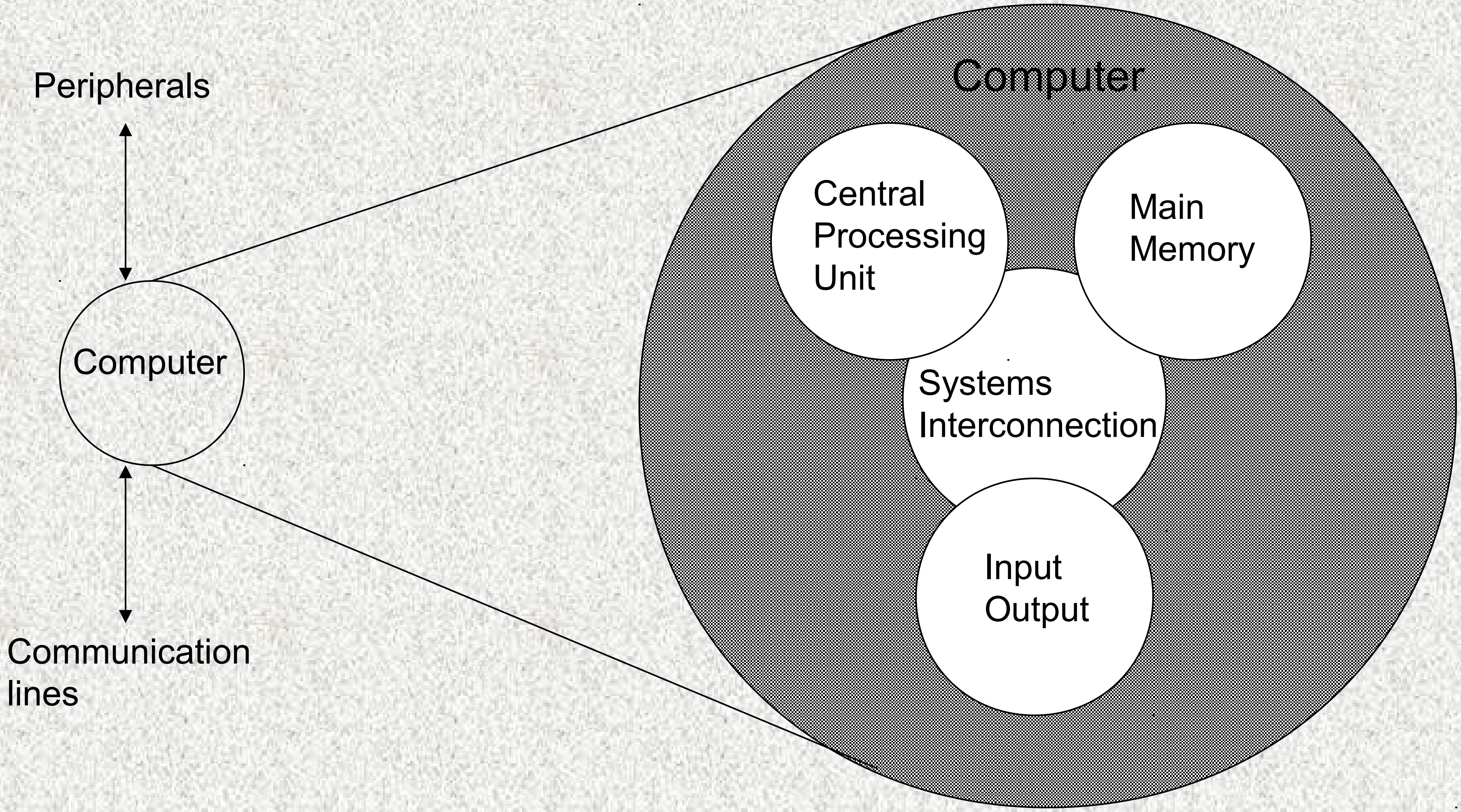
Operation (3) Processing from/to storage



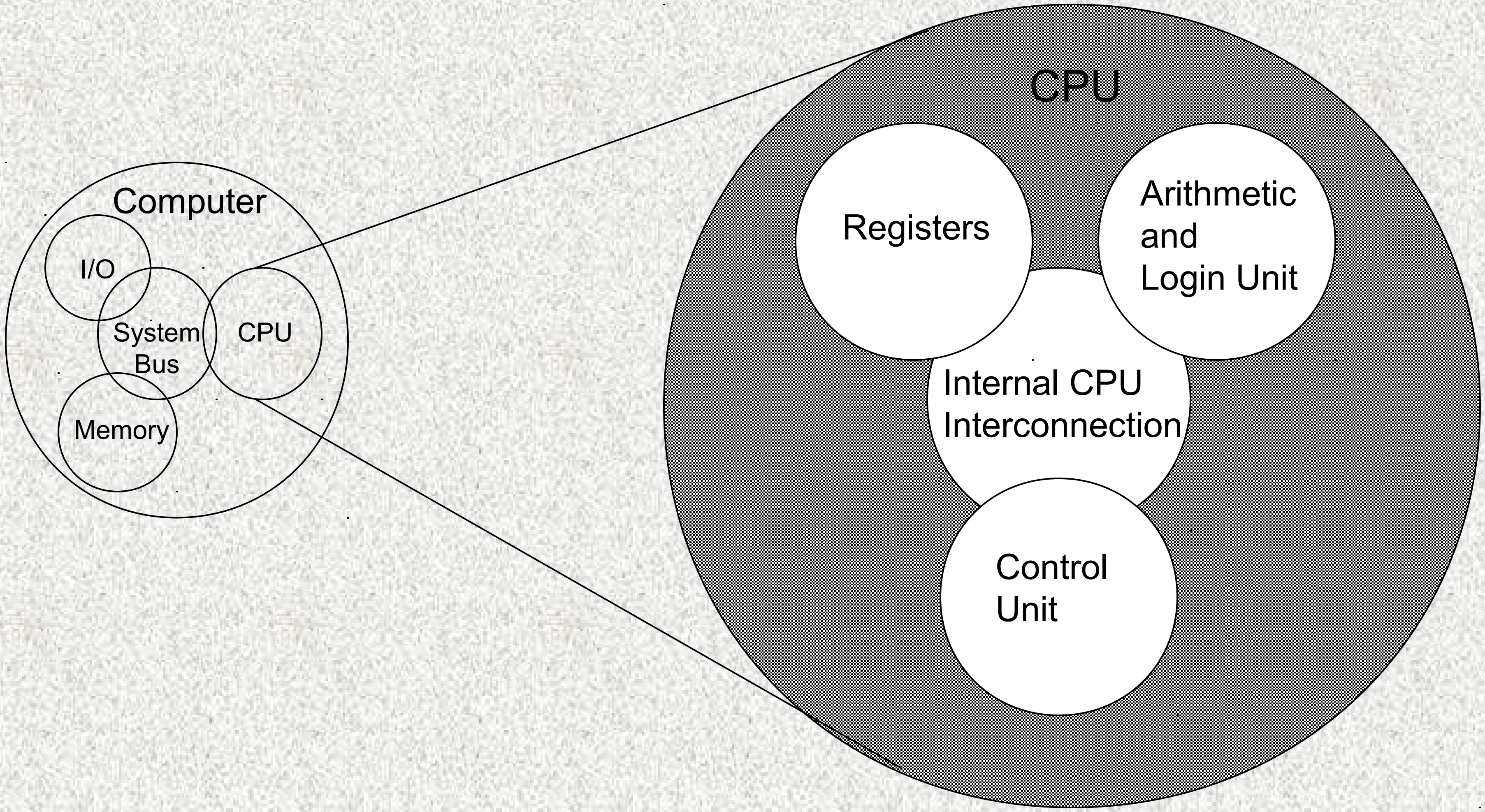
Operation (4)
Processing from storage to I/O



Structure - Top Level structure



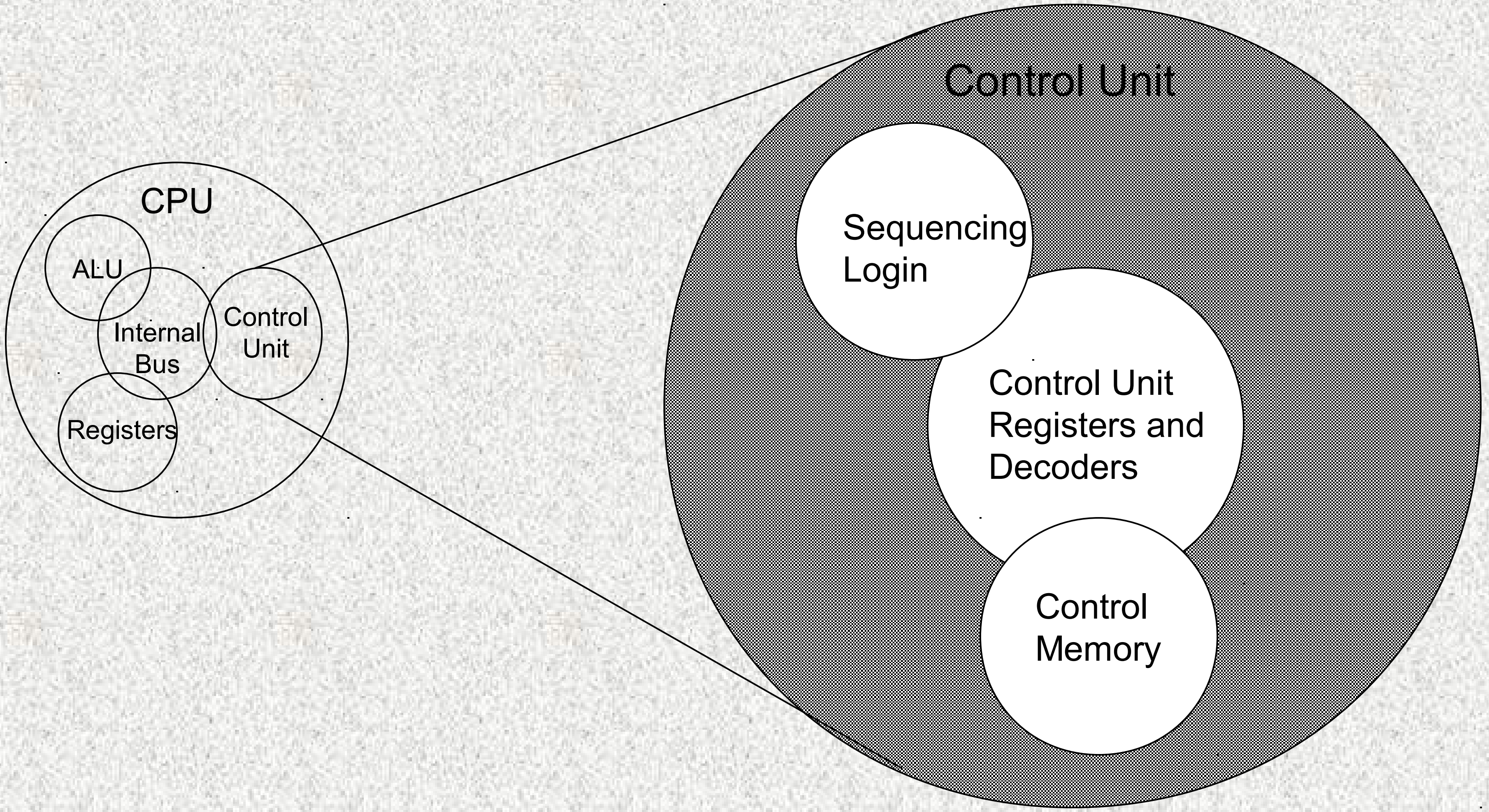
Structure - The CPU



The computer

- **Central processing unit (CPU):** controls the operation of the computer and performs its data processing functions.
- **Main memory:** stores data.
- **I/O:** moves data between the computer and its external environment.
- **System interconnection**: some mechanisim that provides for communication among CPU, main memory and I/O.

Structure - The Control Unit



- **Control unit:** controls the operation of the CPU and the Computer.
- **Arithmetic and Logic Unit (ALU):** performs the computer’s data processing funcions.
- **Registers:** provides storage internal to the CPU.
- **CPU interconnection:** its some mechanism that provides for communication among the control unit, ALU, and register

Outline of the Book (1)

- Computer Evolution and Performance
- Computer Interconnection Structures
- Internal Memory
- External Memory
- Input/Output
- Operating Systems Support
- Computer Arithmetic
- Instruction Sets

Outline of the Book (2)

- CPU Structure and Function.
- Reduced Instruction Set Computers.
- Control Unit Operation.
- Multiprocessors and Vector Processing.
- Digital Logic.

Internet Resources
- Web site for book

- http://WilliamStallings.com/COA6e.html
  - links to sites of interest
  - links to sites for courses that use the book
  - errata list for book
  - information on other books by W. Stallings

Internet Resources
- Web sites to look for

- WWW Computer Architecture Home Page
- CPU Info Center
- ACM Special Interest Group on Computer Architecture
- IEEE Technical Committee on Computer Architecture
- Intel Technology Journal
- Manufacturer’s sites
  - Intel, IBM, etc.

William Stallings
Computer Organization and Architecture

Chapter 2
Computer Evolution and Performance

ENIAC - background

- Electronic Numerical Integrator And Computer.
- Eckert and Mauchly.
- University of Pennsylvania.
- Started 1943.
- Finished 1946.
  - Too late for war effort.
- Used until 1955.

ENIAC - details

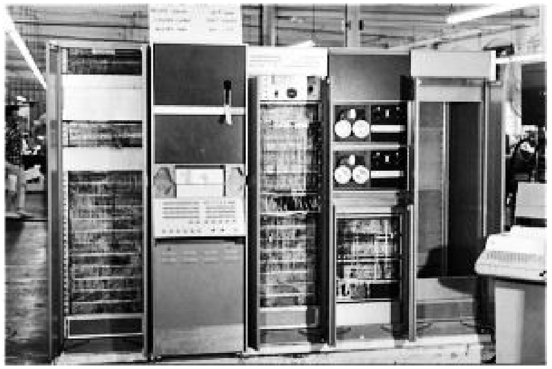
- Decimal (not binary).
- 20 accumulators of 10 digits.
- Programmed manually by switches.
- 18,000 vacuum tubes.
- 30 tons.
- 15,000 square feet.
- 140 kW power consumption.
- 5,000 additions per second.

von Neumann

- Its Stored Program concept proposed 1946.
- Main memory storing programs and data.
- ALU operating on binary data.
- Control unit interpreting instructions from memory and executing.
- Input and output equipment operated by control unit.
- Princeton Institute for Advanced Studies
  - IAS
- Completed 1952.

Designing Computers

- All computers more or less based on the same basic design, the Von Neumann Architecture!

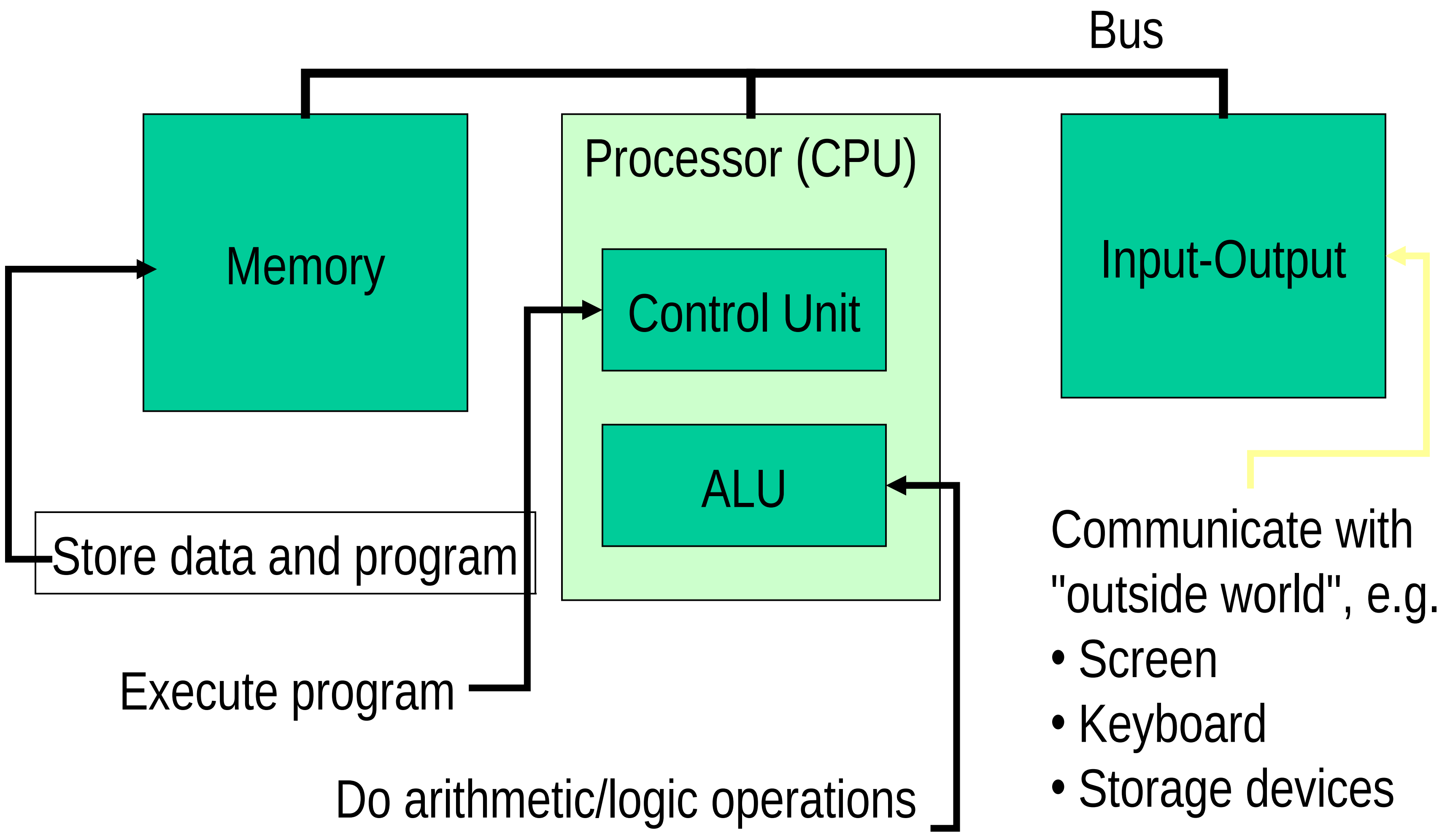


The Von Neumann Architecture

- Model for designing and building computers, based on the following three characteristics:

1) The computer consists of four main sub-systems:
   - Memory
   - ALU (Arithmetic/Logic Unit)
   - Control Unit
   - Input/Output System (I/O)
2) Program is stored in memory during execution.
3) Program instructions are executed sequentially.

The Von Neumann Architecture



Communicate with "outside world", e.g.

- Screen
- Keyboard
- Storage devices
- ...

Von Neuman Machine

- A main memory, which stored both of data and instruction.
- An arithmetic and logic unit (ALU) capable of operating on binary data.
- A control unit, interprets the instructions in memory and causes them to be excuted.
- Input and Output equipment operated by the control unit

IAS - details

- Institute for advanced Study.
- 1000 x 40 bit words.
    - Binary number.
    - 2 x 20 bit instructions
- Set of registers (storage in CPU).
    - Memory Buffer Register.
    - Memory Address Register.
    - Instruction Register.
    - Instruction Buffer Register.
    - Program Counter.
    - Accumulator.

IBM

- Punched-card processing equipment.
- 1953.
  - IBM’s first stored program computer.
  - Scientific calculations.
- 1955.
  - Business applications.

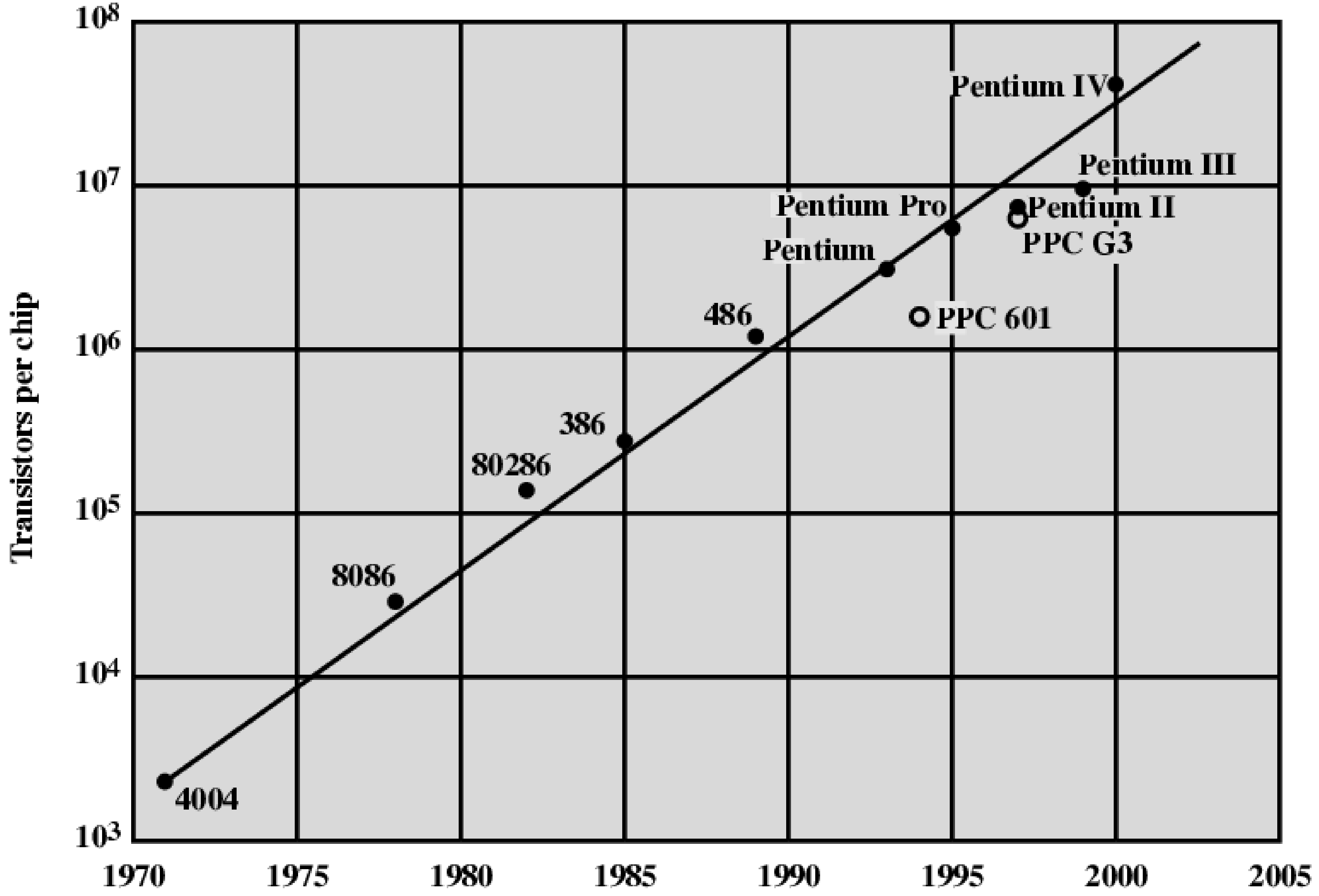
Generations of Computer

- Vacuum tube - 1946-1957
- Transistor - 1958-1964
- Small scale integration - 1965 on
  - Up to 100 devices on a chip
- Medium scale integration - to 1971
  - 100-3,000 devices on a chip
- Large scale integration - 1971-1977
  - 3,000 - 100,000 devices on a chip
- Very large scale integration - 1978
  - 100,000 - 100,000,000 devices on a chip
- Ultra large scale integration
  - Over 100,000,000 devices on a chip

Moore’s Law

- Increased density of components on chip
- Gordon Moore - cofounder of Intel
- Number of transistors on a chip will double every year
- Since 1970’s development has slowed a little
  - Number of transistors doubles every 18 months
- Cost of a chip has remained almost unchanged
- Higher packing density means shorter electrical paths, giving higher performance
- Smaller size gives increased flexibility
- Reduced power and cooling requirements
- Fewer interconnections increases reliability

Growth in CPU Transistor Count



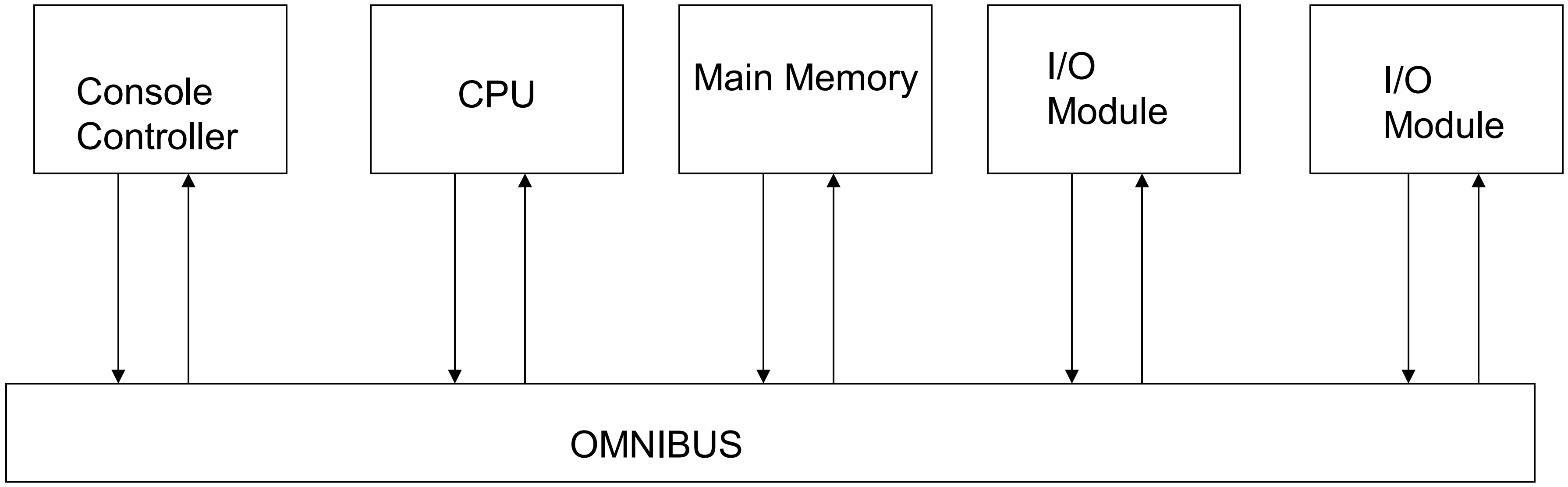
IBM 360 series

- 1964
- Replaced (& not compatible with) 7000 series
- Characteristics of First planned “family” of computers
  - Similar or identical instruction sets
  - Similar or identical O/S
  - Increasing speed
  - Increasing number of I/O ports (i.e. more terminals)
  - Increased memory size
  - Increased cost

DEC PDP-8

- 1964
- First minicomputer (after miniskirt!)
- Did not need air conditioned room
- Small enough to sit on a lab bench

DEC - PDP-8 Bus Structure



Semiconductor Memory

- 1970
- Was produced by Fairchild
- Size of a single core
  - i.e. 1 bit of magnetic core storage
- Holds 256 bits
- Non-destructive read
- Much faster than core
- Capacity approximately doubles each year

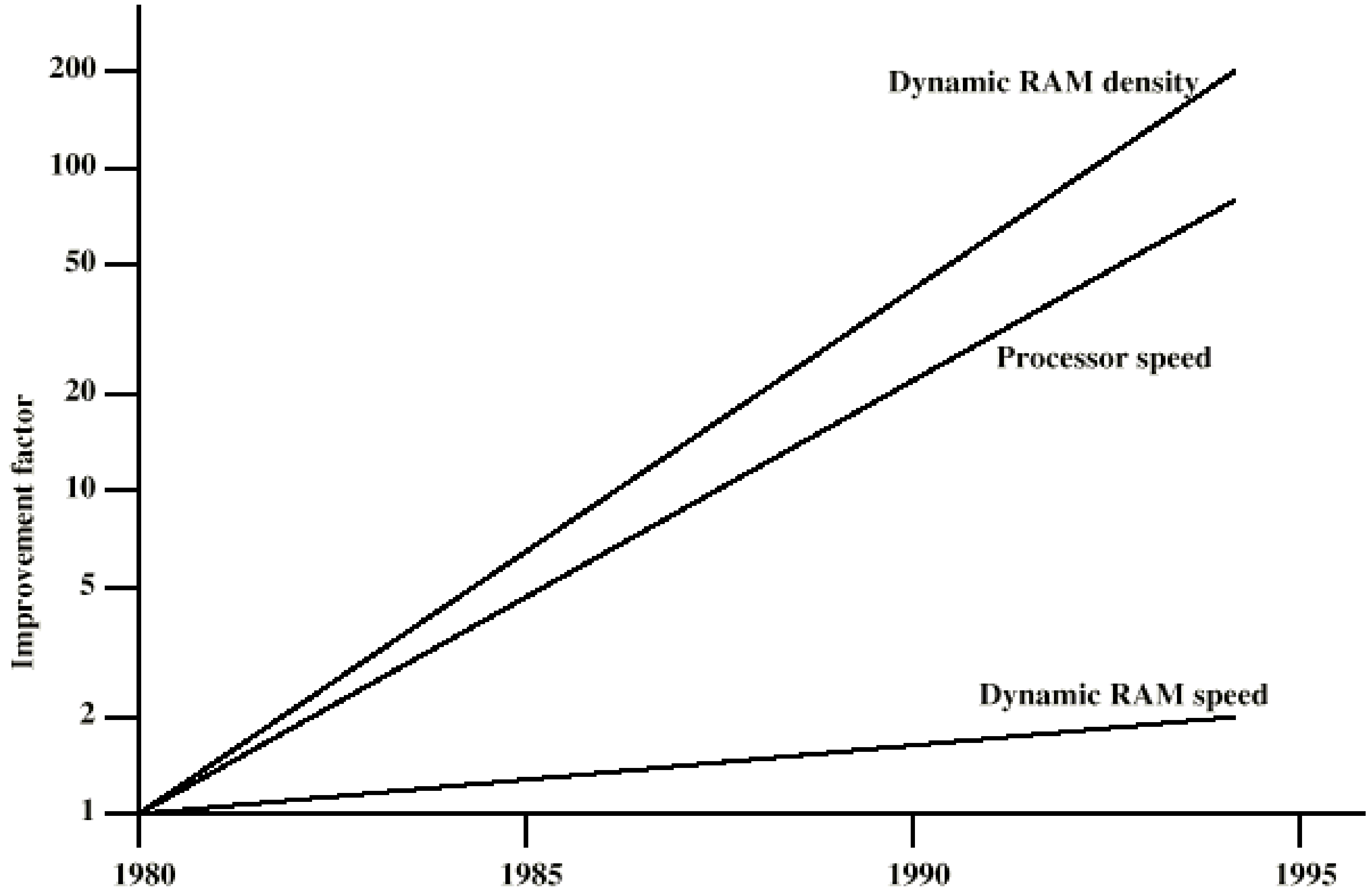
Intel

- 1971 - 4004
  - First microprocessor
  - All CPU components on a single chip
  - 4 bit
- Followed in 1972 by 8008
  - 8 bit
  - Both designed for specific applications
- 1974 - 8080
  - Intel’s first general purpose microprocessor

Speeding it up

- Pipelining
- On board cache
- Branch prediction
- Data flow analysis

DRAM and Processor Characteristics



Solutions

- Increase number of bits retrieved at one time
  - —Make DRAM “wider” rather than “deeper”
- Change DRAM interface
  - —Cache
- Reduce frequency of memory access
  - —More complex cache and cache on chip
- Increase interconnection bandwidth
  - —High speed buses

Pentium Evolution (1)

- 8080
  - first general purpose microprocessor
  - 8 bit data path
  - Used in first personal computer – Altair
- 8086
  - much more powerful
  - 16 bit
  - instruction cache, prefetch few instructions
  - 8088 (8 bit external bus) used in first IBM PC
- 80286
  - 16 Mbyte memory addressable
  - up from 1Mb
- 80386
  - 32 bit
  - Support for multitasking

Pentium Evolution (2)

- 80486
  - sophisticated powerful cache and instruction pipelining
  - built in maths co-processor
- Pentium
  - Superscalar
  - Multiple instructions executed in parallel
- Pentium Pro
  - Increased superscalar organization
  - Aggressive register renaming
  - branch prediction
  - data flow analysis
  - speculative execution

Pentium Evolution (3)

- Pentium II
  - —MMX technology
  - —graphics, video & audio processing
- Pentium III
  - —Additional floating point instructions for 3D graphics
- Pentium 4
  - —Note Arabic rather than Roman numerals
  - —Further floating point and multimedia enhancements
- Itanium
  - —64 bit
  - —see chapter 15
- See Intel web pages for detailed information on processors

Memory Subsystem

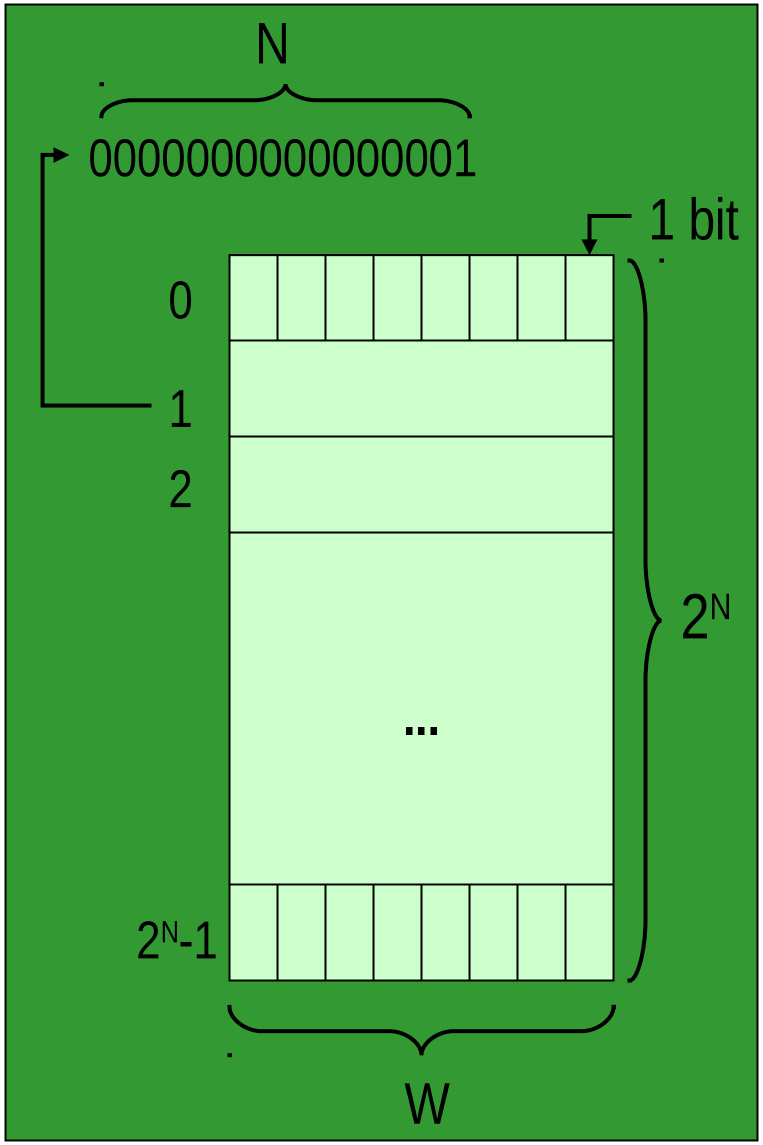
- Memory, also called RAM (Random Access Memory),
  - Consists of many memory cells (storage units) of a fixed size.
    Each cell has an address associated with it: 0, 1, …
  - All accesses to memory are to a specified address.
    A cell is the minimum unit of access (fetch/store a complete cell).
  - The time it takes to fetch/store a cell is the same for all cells.

- When the computer is running, both
  - Program
  - Data (variables)

  are stored in the memory.

RAM

- Need to distinguish between
  - the <u>address</u> of a memory cell and the <u>content</u> of a memory cell
- Memory width (W):
  - How many bits is each memory cell, typically one <u>byte</u> (=8 bits)
- Address width (N):
  - How many bits used to represent each address, determines the maximum memory size = <u>address space</u>
  - If address width is N-bits, then address space is 2^N $(0,1,\ldots,2^N-1)$



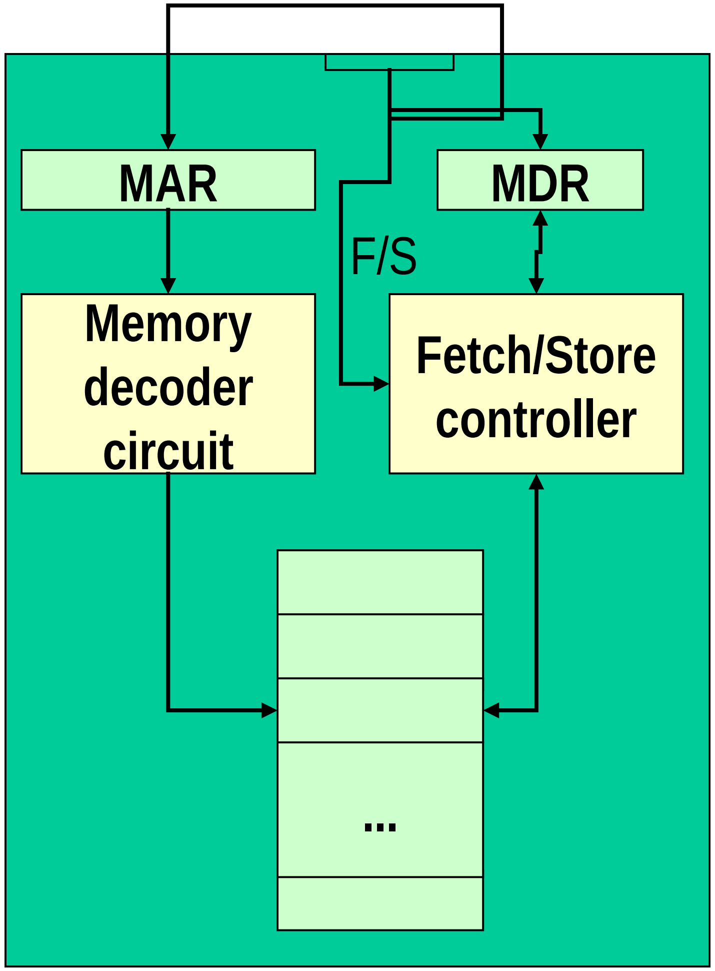
Memory Size / Speed

- Typical memory in a personal computer (PC):
  - 64MB - 256MB
- Memory sizes:
  - Kilobyte (KB)= 2^{10} = 1,024 bytes ~ 1 thousand
  - Megabyte(MB) = 2^{20} = 1,048,576 bytes ~ 1 million
  - Gigabyte(GB) = 2^{30} = 1,073,741,824 bytes ~ 1 billion
- Memory Access Time (read from/ write to memory)
  - 50-75 nanoseconds (1 nsec. = 0.000000001 sec.)
- RAM is
  - volatile (can only store when power is on)
  - relatively expensive

Operations on Memory

- Fetch (address):
  - Fetch a copy of the content of memory cell with the specified address.
  - Non-destructive, copies value in memory cell.
- Store (address, value):
  - Store the specified value into the memory cell specified by address.
  - Destructive, overwrites the previous value of the memory cell.
- The memory system is interfaced via:
  - Memory Address Register (MAR)
  - Memory Data Register (MDR)
  - Fetch/Store signal

Structure of the Memory Subsystem



- Fetch(address)
  - Load address into MAR.
  - Decode the address in MAR.
  - Copy the content of memory cell with specified address into MDR.
- Store(address, value)
  - Load the address into MAR.
  - Load the value into MDR.
  - Decode the address in MAR
  - Copy the content of MDR into memory cell with the specified address.

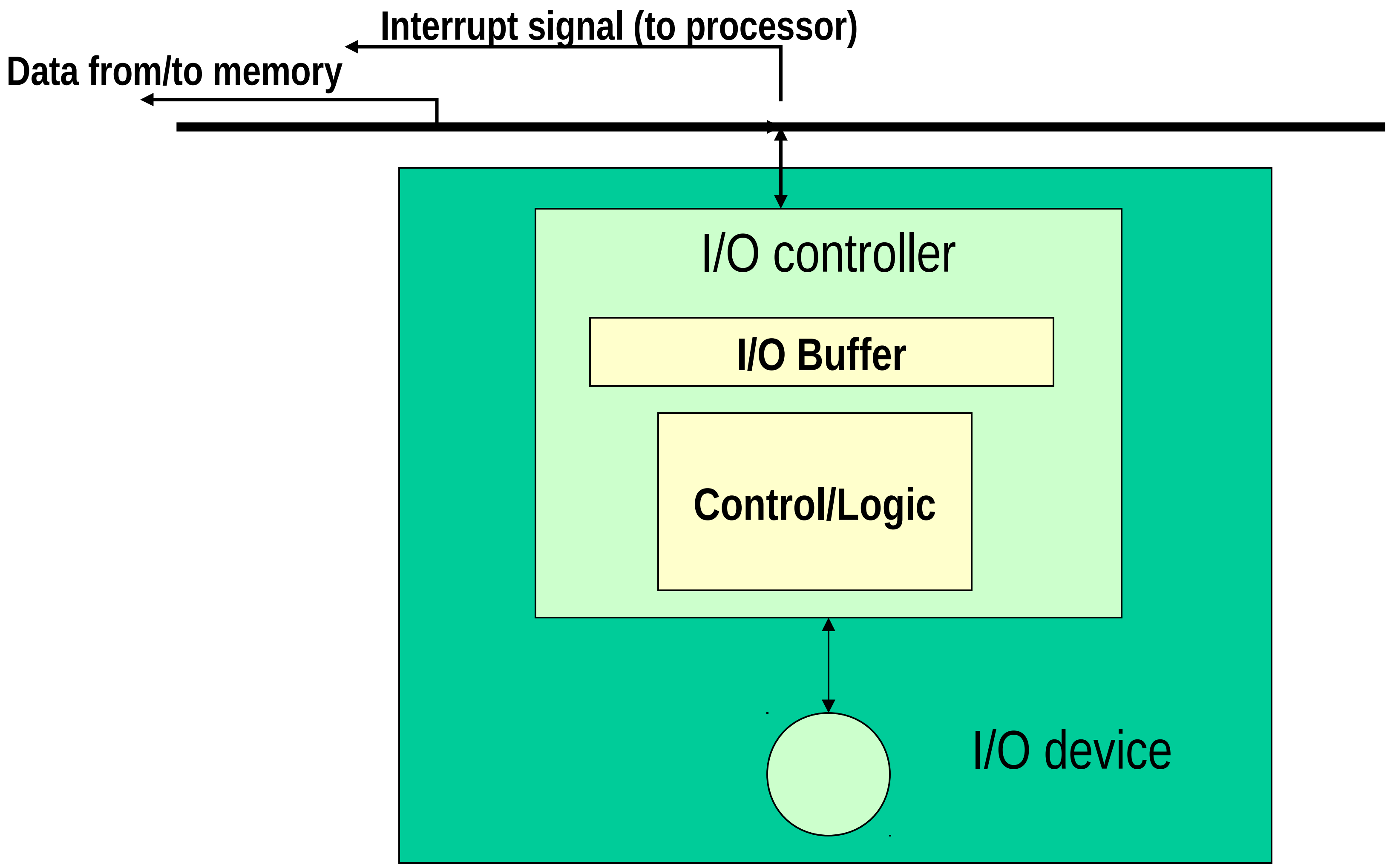
Input/Output Subsystem

- Handles devices that allow the computer system to:
  - Communicate and interact with the outside world
    - Screen, keyboard, printer, ...
  - Store information (mass-storage)
    - Hard-drives, floppies, CD, tapes, …
- Mass-Storage Device Access Methods:
  - Direct Access Storage Devices (DASDs)
    - Hard-drives, floppy-disks, CD-ROMs, ...
  - Sequential Access Storage Devices (SASDs)
    - Tapes (for example, used as backup devices)

I/O Controllers

- Speed of I/O devices is slow compared to RAM
  - RAM ~ 50 nsec.
  - Hard-Drive ~ 10msec. = (10,000,000 nsec)
- Solution:
  - I/O Controller, a special purpose processor:
    - Has a small memory buffer, and a control logic to control I/O device (e.g. move disk arm).
    - Sends an interrupt signal to CPU when done read/write.
  - Data transferred between RAM and memory buffer.
  - Processor free to do something else while I/O controller reads/writes data from/to device into I/O buffer.

Structure of the I/O Subsystem

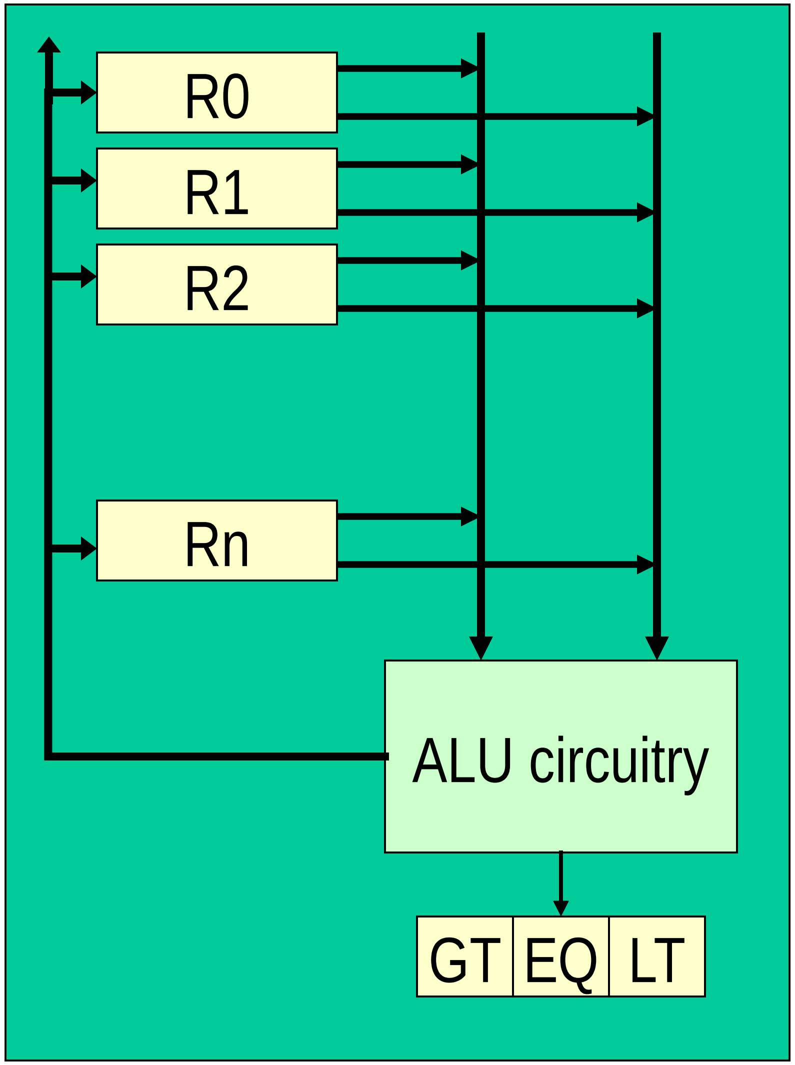


The ALU Subsystem

- The ALU (Arithmetic/Logic Unit) performs
  - mathematical operations (+, -, x, /, …)
  - logic operations (=, <, >, and, or, not, ...)
- In today's computers integrated into the CPU
- Consists of:
  - Circuits to do the arithmetic/logic operations.
  - Registers (fast storage units) to store intermediate computational results.
  - Bus that connects the two.

Structure of the ALU

- Registers:
  - Very fast local memory cells, that store operands of operations and intermediate results.
  - CCR (condition code register), a special purpose register that stores the result of <, = , > operations
- ALU circuitry:
  - Contains an array of circuits to do mathematical/logic operations.
- Bus:
  - Data path interconnecting the registers to the ALU circuitry.



The Control Unit

- Program is stored in memory
  - as machine language instructions, in binary
- The task of the <u>control unit</u> is to execute programs by repeatedly:
  - <u>Fetch</u> from memory the next instruction to be executed.
  - <u>Decode</u> it, that is, determine what is to be done.
  - <u>Execute</u> it by issuing the appropriate signals to the ALU, memory, and I/O subsystems.
  - Continues until the HALT instruction

Instruction Set Design

- Two different approaches:
  - Reduced Instruction Set Computers (RISC)
    - Instruction set as small and simple as possible.
    - Minimizes amount of circuitry --> faster computers
  - Complex Instruction Set Computers (CISC)
    - More instructions, very complex
    - Each instruction can do more work, but require more circuitry.

Typical Machine Instructions

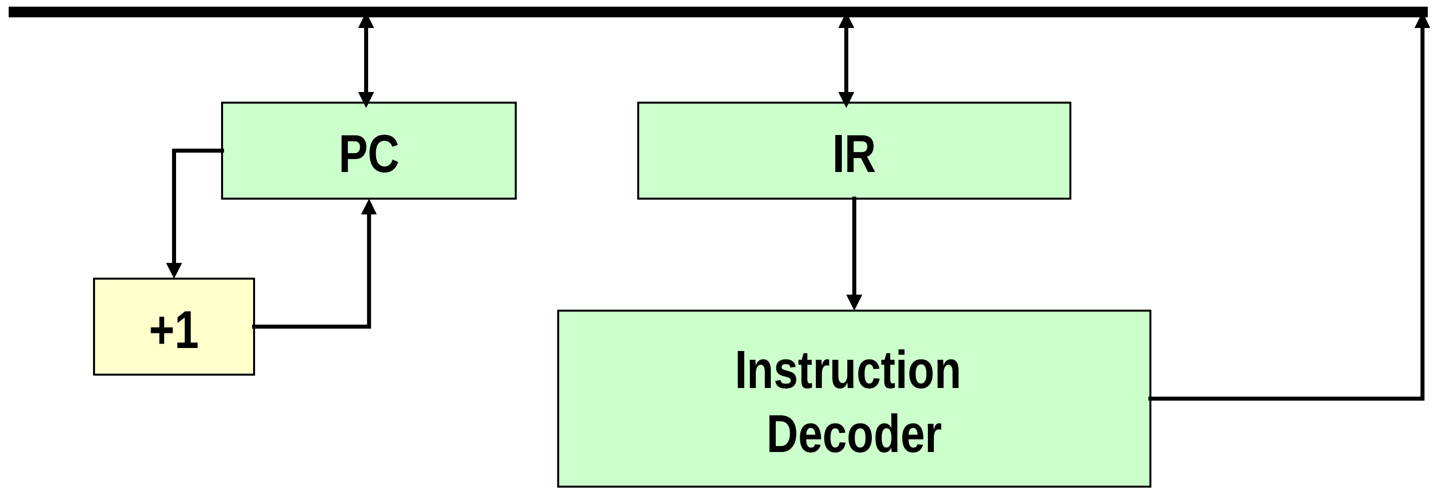
- Notation:
  - We use X, Y, Z to denote RAM cells
  - Assume only one register R (for simplicity)
  - Use English-like descriptions (should be binary)
- Data Transfer Instructions
  - LOAD X Load content of memory location X to R
  - STORE X Load content of R to memory location X
  - MOVE X, Y Copy content of memory location X to loc. Y

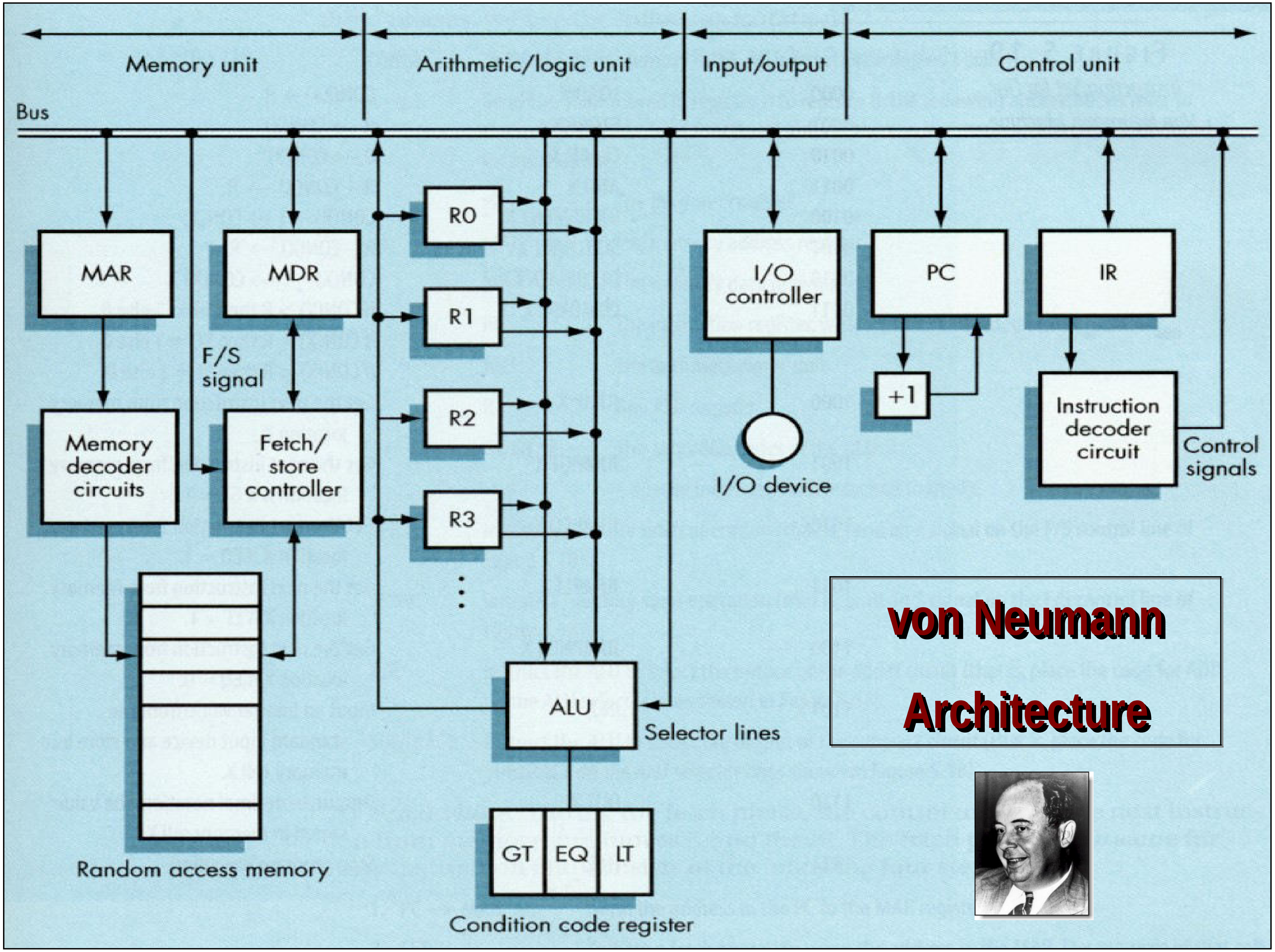
Example

- Pseudo-code: Set A to B + C
- Assuming variable:
  - A stored in memory cell 100, B stored in memory cell 150, C stored in memory cell 151
- Machine language (really in binary)
  - LOAD 150
  - ADD 151
  - STORE 100
  - or
  - (ADD 150, 151, 100)

Structure of the Control Unit

- PC (Program Counter):
  - stores the address of next instruction to fetch
- IR (Instruction Register):
  - stores the instruction fetched from memory
- Instruction Decoder:
  - Decodes instruction and activates necessary circuitry





von Neumann Architecture

How does this all work together?

- Program Execution:
  - PC is set to the address where the first program instruction is stored in memory.
  - Repeat until HALT instruction or fatal error

    Fetch instruction

    Decode instruction

    Execute instruction

  End of loop

CH3: CISC and RISC Architecture

Dr. Moaath shatnawi

Instruction Format

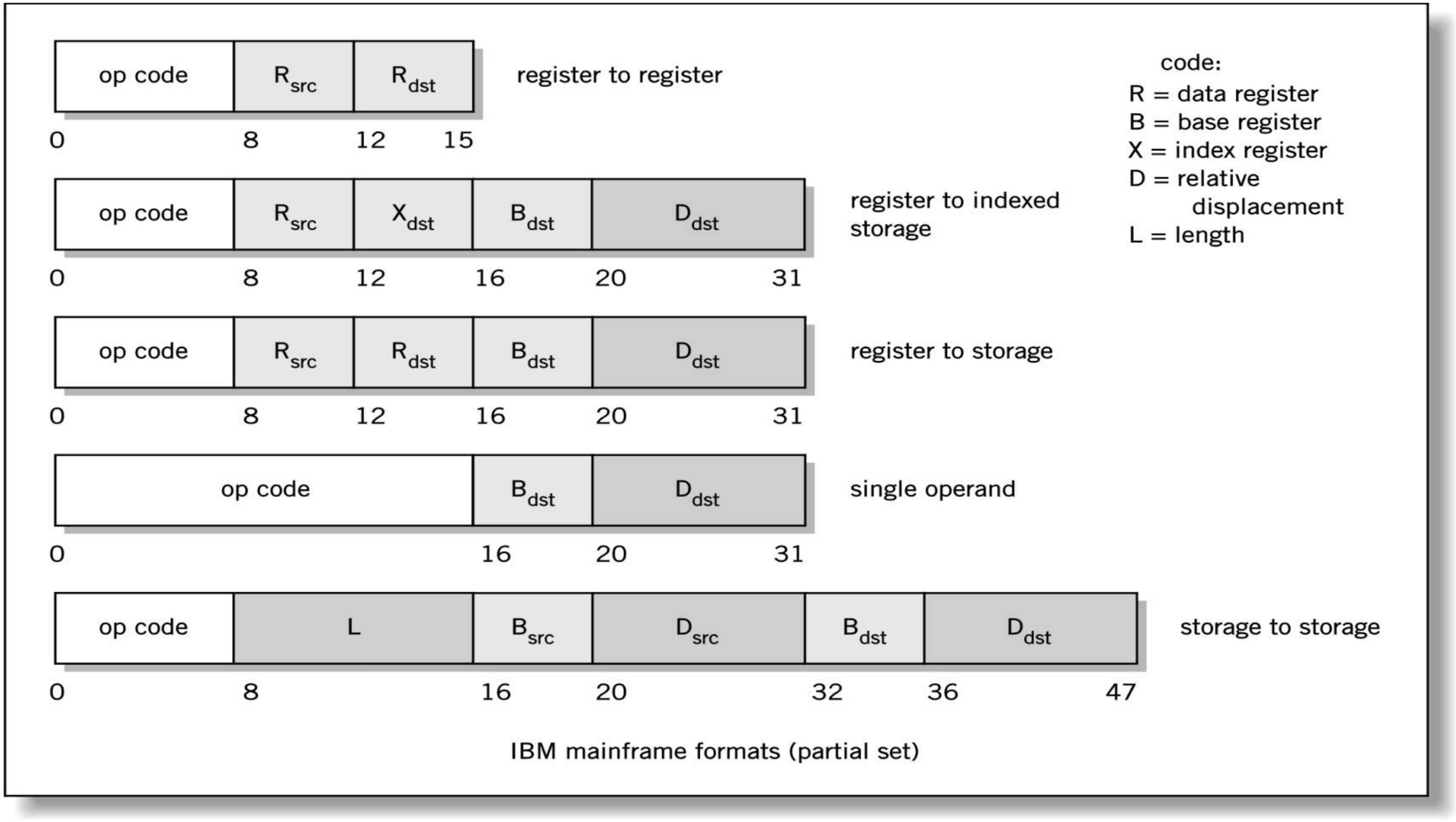
What is CISC?

- CISC is an acronym for Complex Instruction Set Computer and are chips that are easy to program and which make efficient use of memory. Since the earliest machines were programmed in assembly language and memory was slow and expensive, the CISC philosophy made sense
- Most common microprocessor designs such as the Intel 80x86 and Motorola 68K series followed the CISC philosophy.
- But recent changes in software and hardware technology have forced a re-examination of CISC and many modern CISC processors are hybrids, implementing many RISC principles.
- CISC was developed to make compiler development simpler. It shifts most of the burden of generating machine instructions to the processor. For example, instead of having to make a compiler write long machine instructions to calculate a square-root, a CISC processor would have a built-in ability to do this.

Most CISC hardware architectures have several characteristics in common:

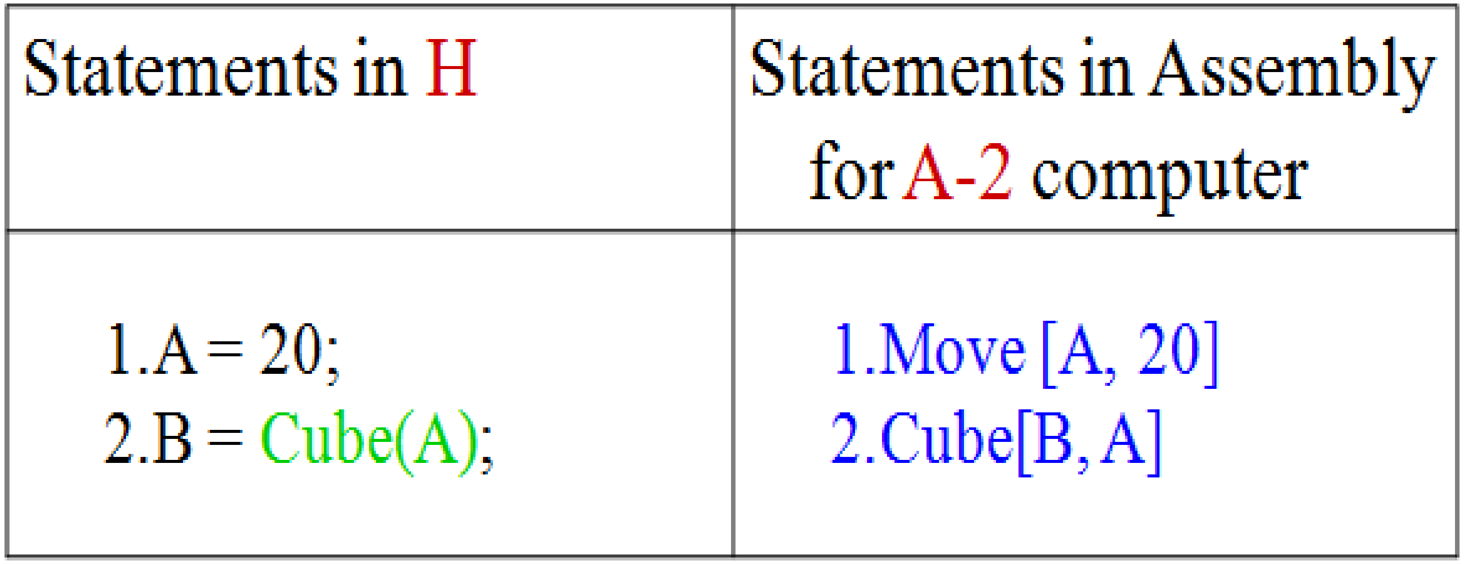
- Complex instruction-decoding logic, driven by the need for a single instruction to support multiple addressing modes.
- A small number of general purpose registers. This is the direct result of having instructions which can operate directly on memory and the limited amount of chip space not dedicated to instruction decoding, execution, and microcode storage.
- Several special purpose registers. Many CTSC designs set aside special registers for the stack pointer, interrupt handling, and so on. This can simplify the hardware design somewhat, at the expense of making the instruction set more complex.
- A 'Condition code" register which is set as a side-effect of most instructions. This register reflects whether the result of the last operation is less than, equal to, or greater than zero and records if certain error conditions occur.

Instruction Formats: CISC



IBM mainframe formats (partial set)

Instruction Formats: CISC example1



Statements in H	Statements in Assembly for A-2 computer

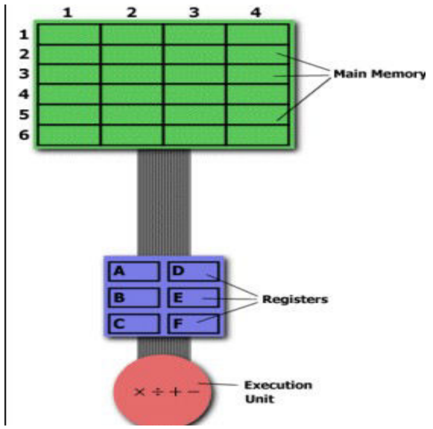
1.A = 20;
2.B = Cube(A);
	1.Move [A, 20]
	2.Cube[B, A]

Instruction Formats: CISC example2

- **The CISC Approach**
  The primary goal of CISC architecture is to complete a task in as few lines of assembly as possible. This is achieved by building processor hardware that is capable of understanding and executing a series of operations. For this particular task, a CISC processor would come prepared with a specific instruction (we'll call it "MULT"). When executed, this instruction loads the two values into separate registers, multiplies the operands in the execution unit, and then stores the product in the appropriate register. Thus, the entire task of multiplying two numbers can be completed with one instruction:

MULT 2:3, 5:2

MULT is what is known as a "complex instruction."



What is RISC?

- **RISC?**
  RISC, or *Reduced Instruction Set Computer*. is a type of microprocessor architecture that utilizes a small, highly-optimized set of instructions, rather than a more specialized set of instructions often found in other types of architectures.
- **History**
  The first RISC projects came from IBM, Stanford, and UC-Berkeley in the late 70s and early 80s. The IBM 801, Stanford MIPS, and Berkeley RISC 1 and 2 were all designed with a similar philosophy which has become known as RISC. Certain design features have been characteristic of most RISC processors:
  - *one cycle execution time*: RISC processors have a CPI (clock per instruction) of one cycle. This is due to the optimization of each instruction on the CPU and a technique called PIPELINING
  - *pipelining*: a techique that allows for simultaneous execution of parts, or stages, of instructions to more efficiently process instructions;
  - *large number of registers*: the RISC design philosophy generally incorporates a larger number of registers to prevent in large amounts of interactions with memory

RISC Attributes

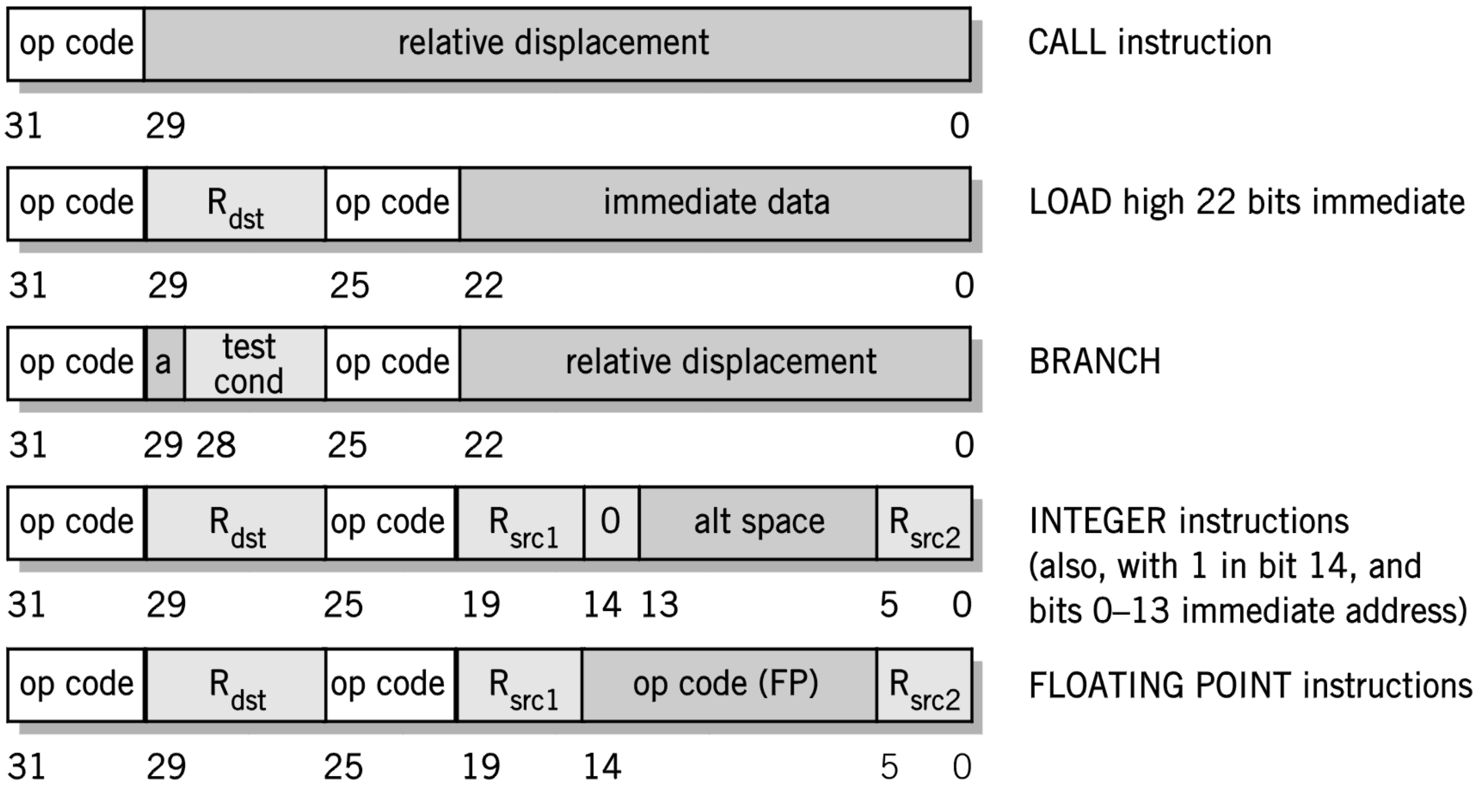
The main characteristics of CISC microprocessors are:

- Extensive instructions.
- Complex and efficient machine instructions.
- Microencoding of the machine instructions.
- Extensive addressing capabilities for memory operations.
- Relatively few registers.

In comparison, RISC processors are more or less the opposite of the above:

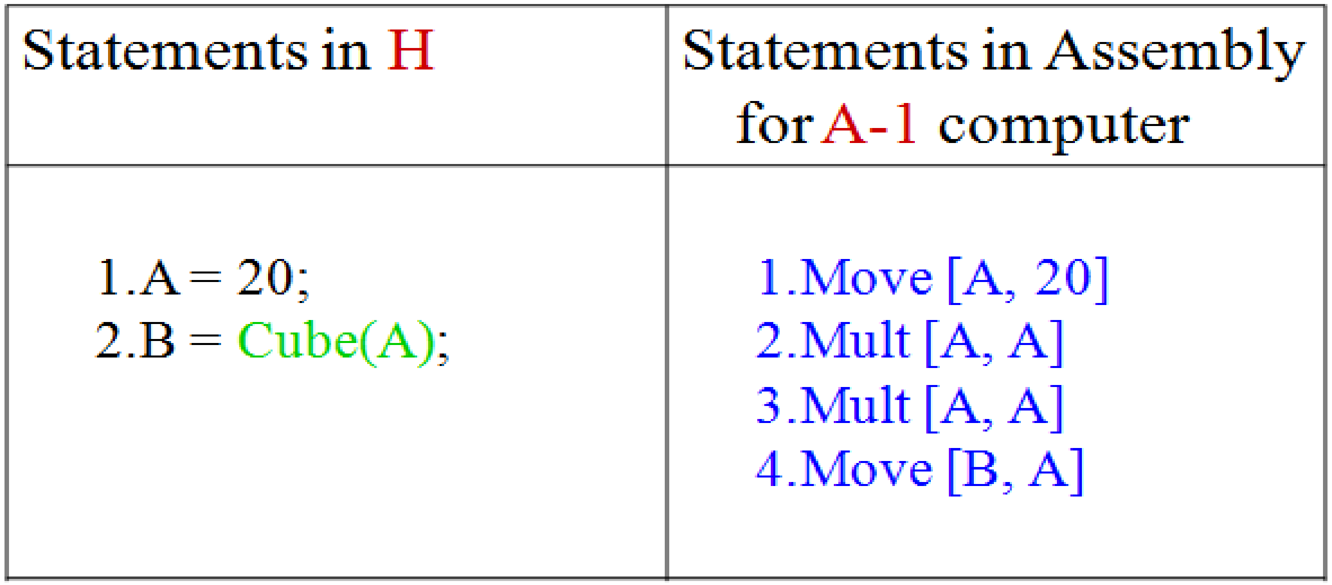
- Reduced instruction set.
- Less complex, simple instructions.
- Hardwired control unit and machine instructions.
- Few addressing schemes for memory operands with only two basic instructions, LOAD and
- STORE
- Many symmetric registers which are organised into a register file.

Instruction Formats: RISC



SPARC RISC formats (complete set)

Instruction Formats: RISC example1



Statements in H	Statements in Assembly for A-1 computer

1.A = 20;
2.B = Cube(A);
	1.Move [A, 20]
	2.Mult [A, A]
	3.Mult [A, A]
	4.Move [B, A]

Instruction Formats: RISC example2

RISC processors only use simple instructions that can be executed within one clock cycle. Thus, the "MULT" command described above could be divided into three separate commands: "LOAD," which moves data from the memory bank to a register, "PROD," which finds the product of two operands located within the registers, and "STORE," which moves data from a register to the memory banks. In order to perform the exact series of steps described in the CISC approach, a programmer would need to code four lines of assembly:

```
LOAD A, 2:3
LOAD B, 5:2
PROD A, B
STORE 2:3, A
```

CISC versus RISC

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

Chip Types

- Complex Instruction Set Computer (CISC)
  - Many forms of instructions (some special purpose, but chip must support all of them)
  - x86, Pentium
- Reduced Instruction Set Computer (RISC)
  - Specific list of supported instructions (Want to do something else? Find a combination of instructions to accomplish the task)
  - Power PC (Motorola), Alpha, IBM RISC System/6000, Sun SPARC, MIPS
- Difference is speed of execution: RISC is faster, but may require longer instruction combinations to accomplish the same task as a CISC

CISC Architecture

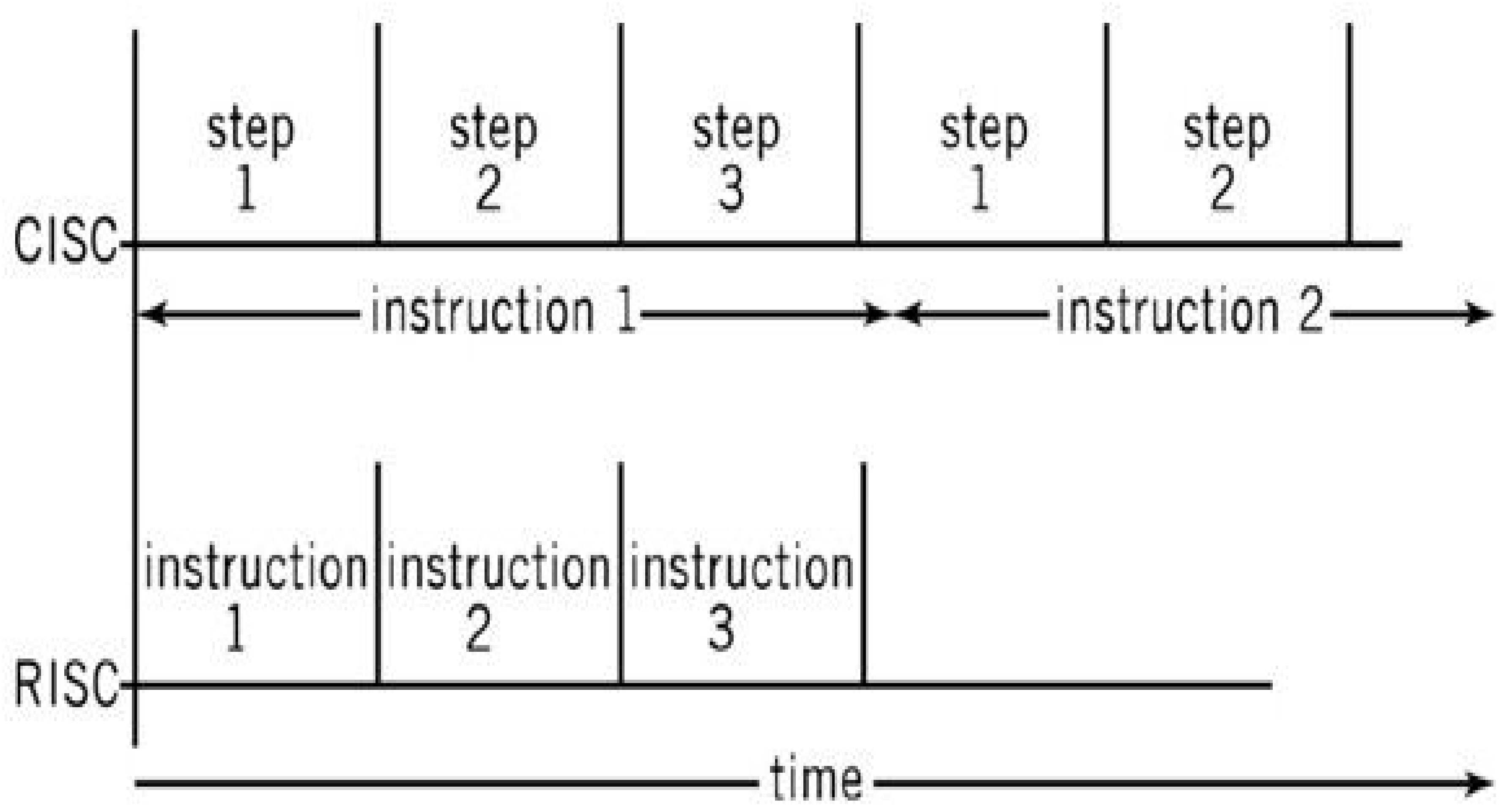
Characteristics

- Few general purpose registers
- Many addressing modes
- Large number of specialized, complex instructions
- Instructions are of varying sizes

RISC Features

- Limited and simple instruction set
- Fixed length, fixed format instruction words
  - Enable pipelining, parallel fetches and executions
- Limited addressing modes
  - Reduce complicated hardware
- Register-oriented instruction set
  - Reduce memory accesses
- Large bank of registers
  - Reduce memory accesses
  - Efficient procedure calls

CISC vs. RISC Processing


step 1
step 2
step 3
step 1
step 2
CISC
instruction 1
instruction 2
instruction 1
instruction 2
instruction 3
RISC
time