

CHAPTER 1: INTRODUCTION

1

Computer Architecture and Organization

Computer Architecture and Organization

LEARNING OBJECTIVES

- Know the difference between CA & CO
- Explain the general functions and structure of a digital computer.
- Present an overview of the evolution of computer technology
- Understand the key performance issues that relate to computer design.

COMPUTER ARCHITECTURE

- Computer Architecture refers to those attributes of a system that have a direct impact on the **logical execution** of a program.
- Architectural attributes are visible to the programmer
 - Instruction set,
 - Number of bits used for data representation,
 - I/O mechanisms, addressing techniques.

COMPUTER ORGANIZATION

- Organization refer to **how** features are implemented and the **operational units** of the computer and their **interconnections** that implement the feature
 - Control signals,
 - Interfaces between computer and peripherals
 - The memory technology being used
- So, for example, the fact that a multiply instruction is available is a computer architecture issue.
- How that multiply is implemented is a computer organization issue.

DIFFERENCE BETWEEN CA & CO

Computer Architecture	Computer Organization
Higher level	Lower level (microarchitecture)
Visible and very important for programmer	Not so important for programmers
Logical components (Instruction set, Addressing modes, Data types)	Physical components (circuit design, signals, peripherals, adders)
What to do? (instruction set)	How to do? (implementation of the architecture)
Computer Architecture and Organization	

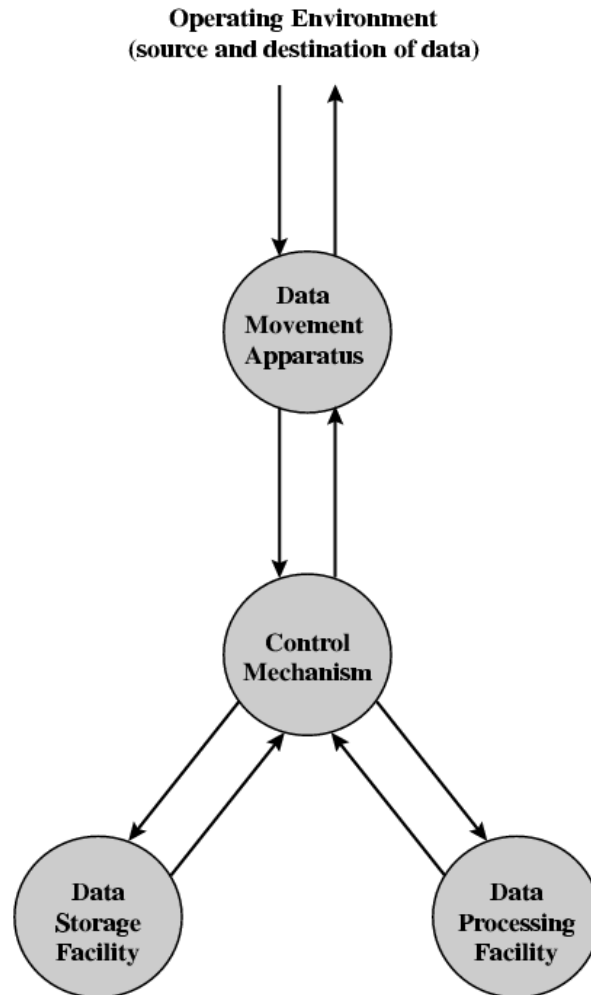
ARCHITECTURE & ORGANIZATION ...

- All **Intel x86** family share the same basic architecture
- The **IBM System/370** family share the same basic architecture
- This means the same software can be used on different models, only the hardware changes for performance improvement.
- This gives code compatibility
 - At least backwards

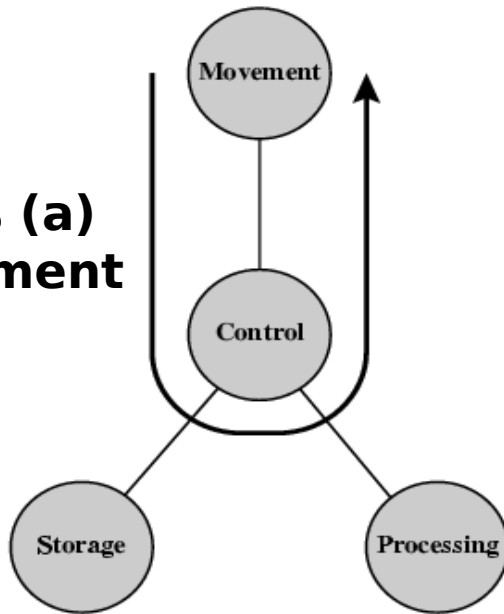
STRUCTURE & FUNCTION

- **Structure** is the way in which components relate to each other
- **Function** is the operation of individual components as part of the structure
- All computer functions are:
 - Data processing
 - Data storage
 - Data movement
 - Control

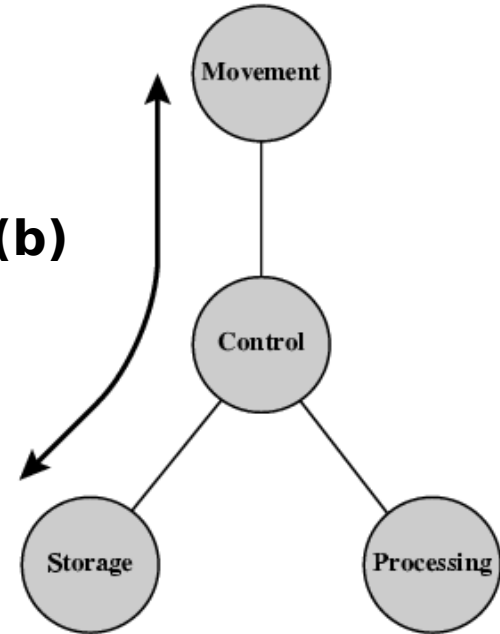
FUNCTIONAL VIEW



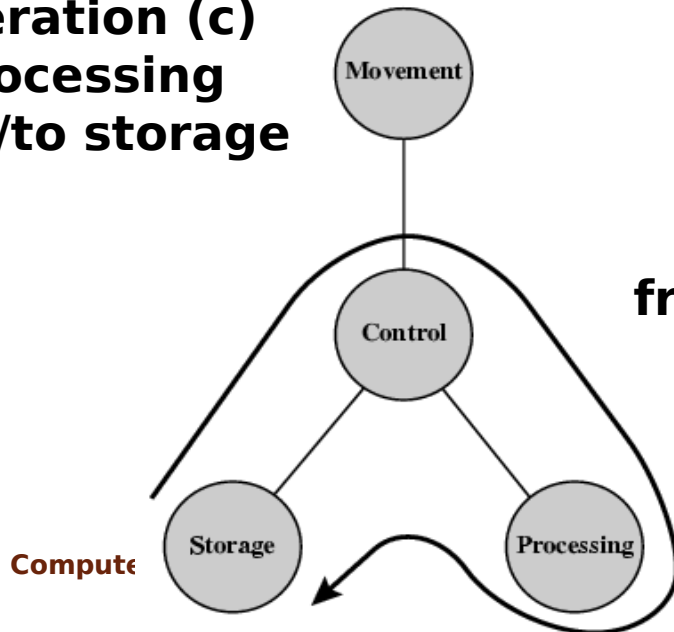
**Operations (a)
Data movement**



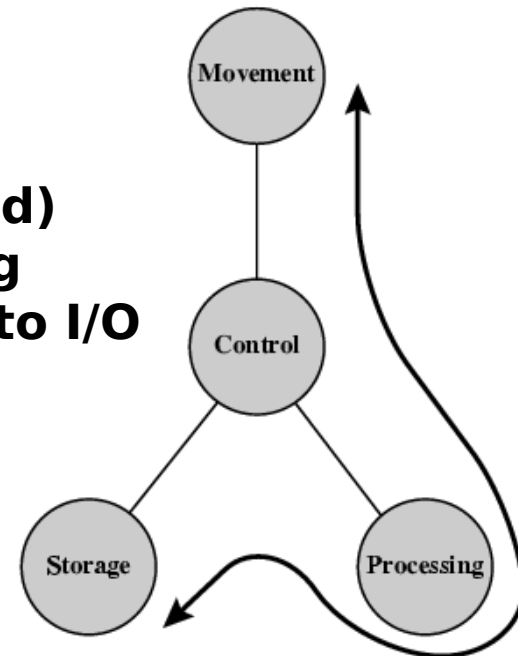
**Operations (b)
Storage**



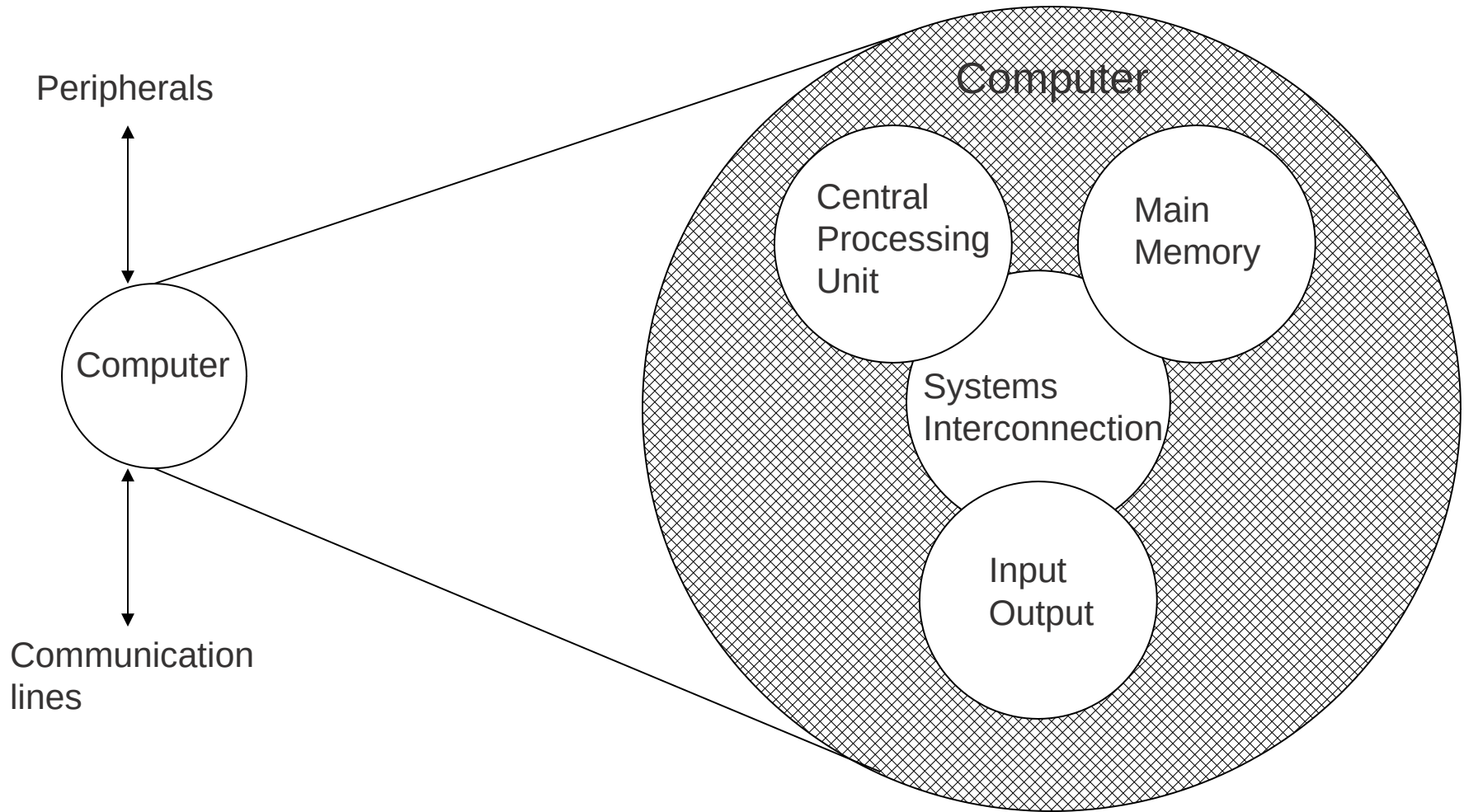
**Operation (c)
Processing
from/to storage**



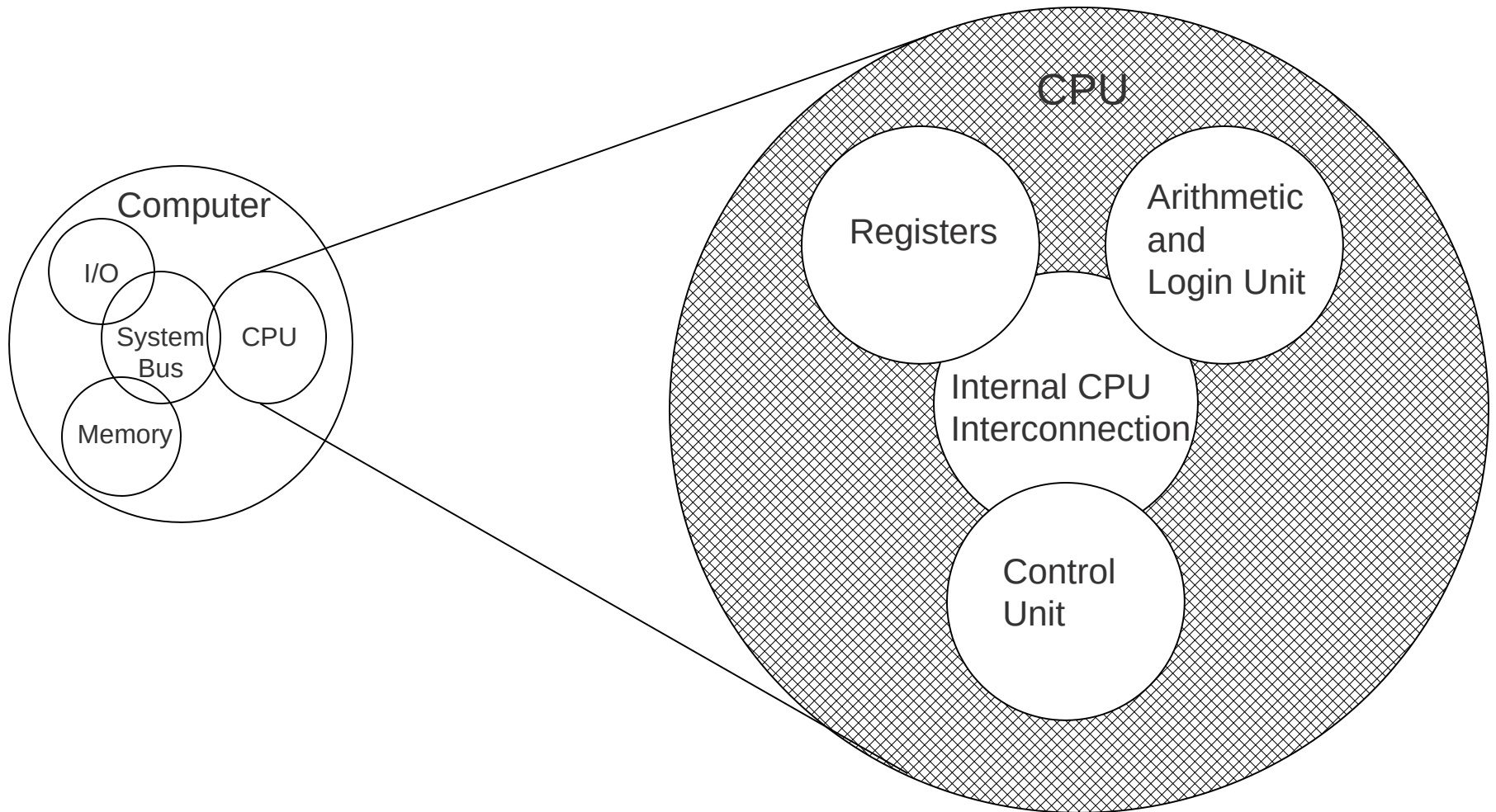
**Operation (d)
Processing
from storage to I/O**



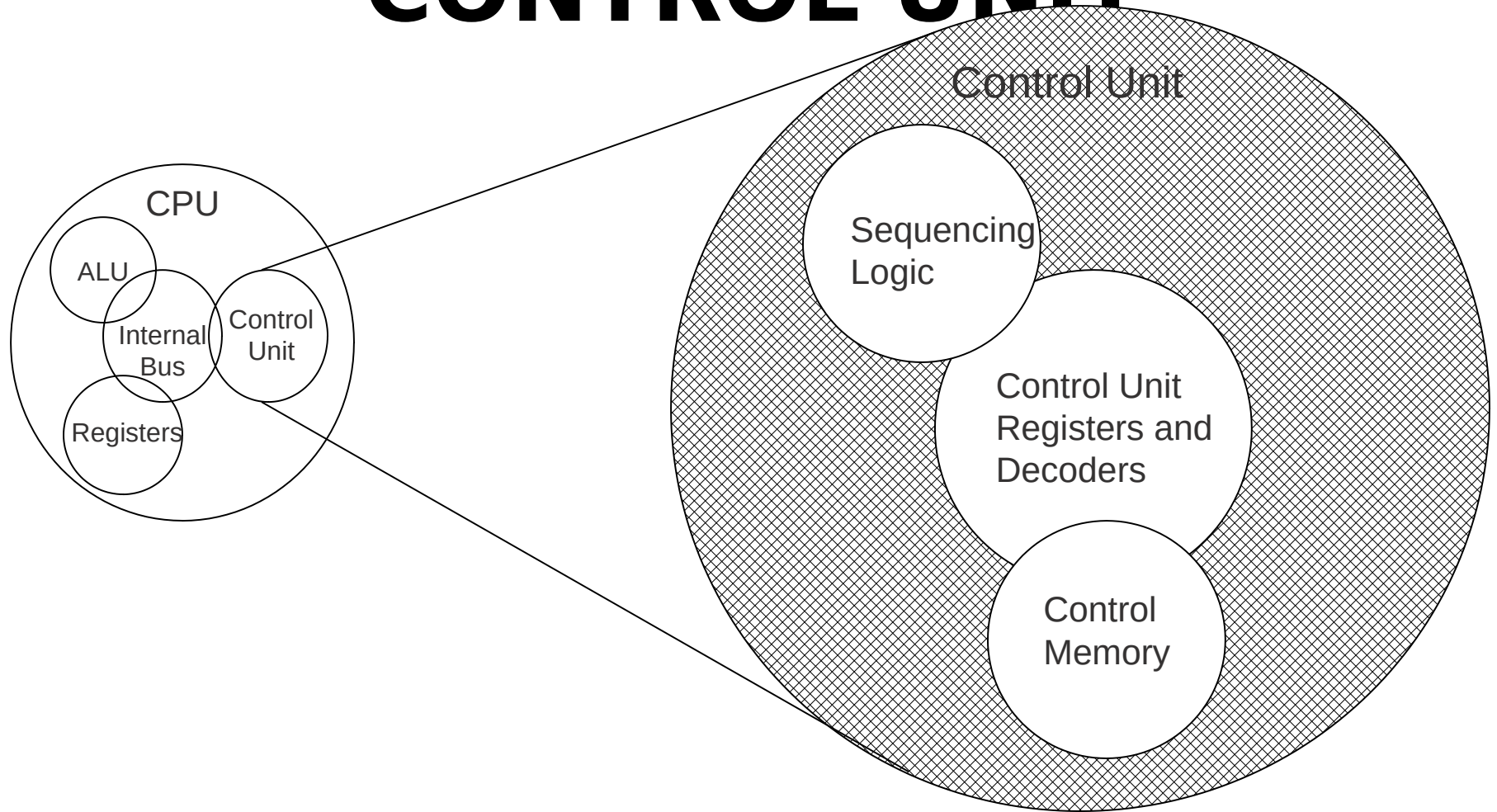
STRUCTURE - TOP LEVEL



STRUCTURE - THE CPU



STRUCTURE - THE CONTROL UNIT



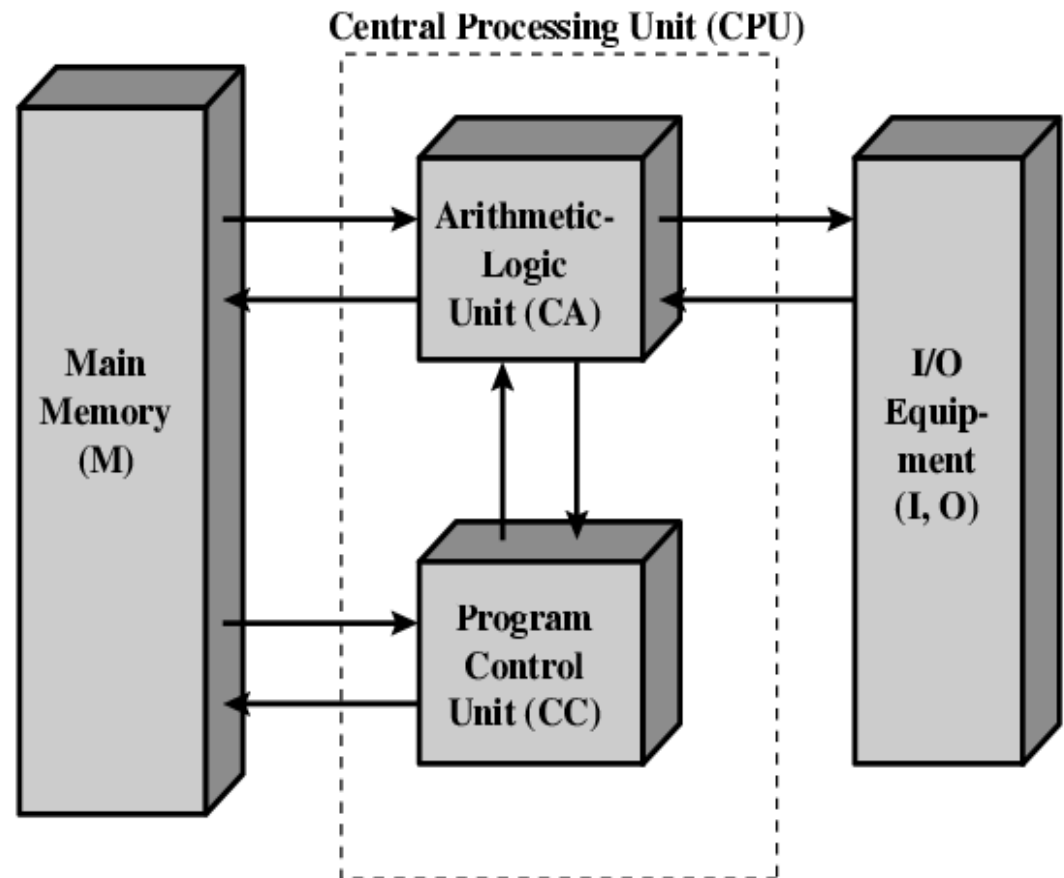
EVOLUTION OF A COMPUTER

The First Generation: Vacuum Tubes

- Computers used **vacuum tubes** for digital logic elements and memory.
- **Stored-program concept** (John von Neumann/ Alan Turing)
- The first publication of the idea was in a 1945 proposal by von Neumann for a new computer, the EDVAC (Electronic Discrete Variable Computer).
- IAS (1946-1952)
 - a new stored-Program computer by von Neumann and his colleagues
 - at the Princeton **I**nstitute for **A**dvanced **S**tudies.
 - With rare exceptions, all of today's computers have this same general structure and function

STRUCTURE OF VON NEUMANN MACHINE

- **A main memory**, which stores both data and instructions
- **An arithmetic and logic unit** (ALU) capable of operating on binary data
- **A control unit**, which interprets the instructions in memory and causes them to be executed
- **Input/output** (I/O) equipment operated by the control unit

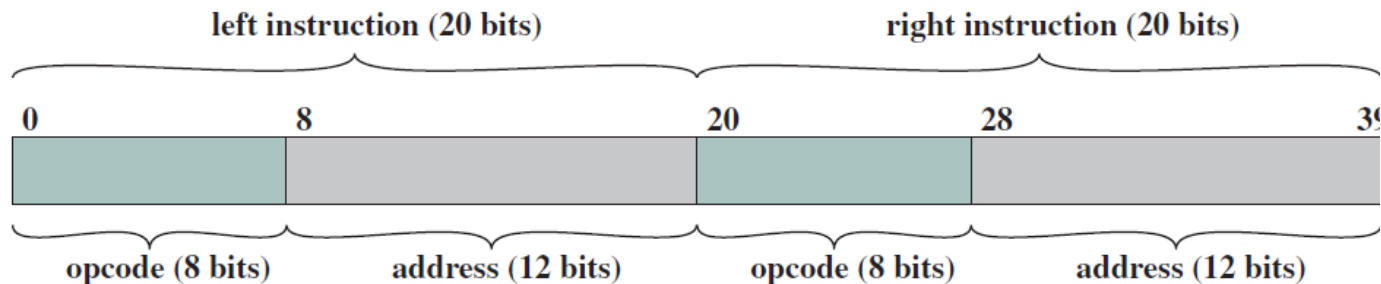
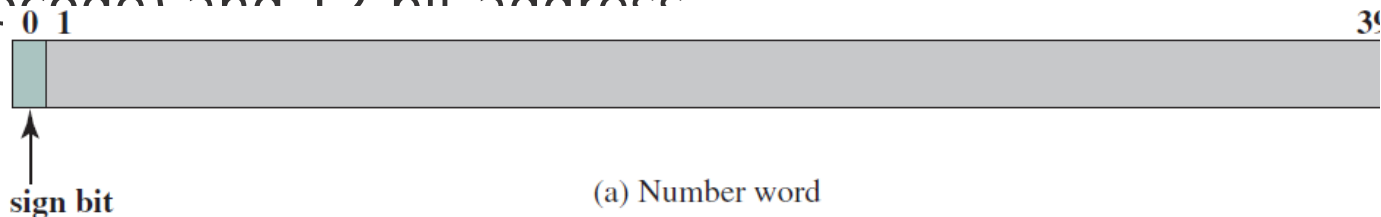


BRIEF DESCRIPTION OF THE IAS COMPUTER

- The memory of the IAS consists of storage locations, called words, of 40 binary digits (bits) each.
- Both data and instructions are stored there.
- Numbers are represented in binary form, and each instruction is a binary code.

IAS MEMORY FORMAT

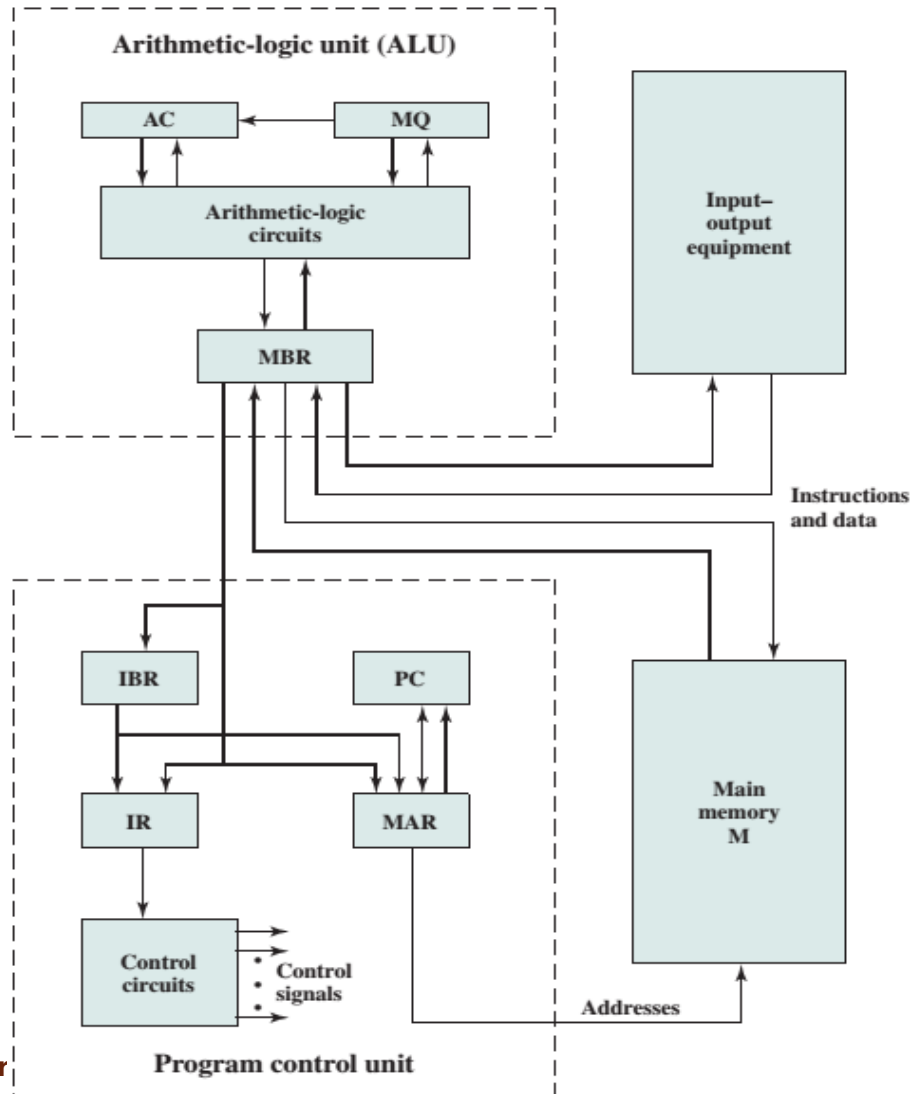
- Each number is represented by a sign bit and a 39-bit value.
- A word may also contain two 20-bit instructions, with each instruction consisting of 8-bit operation code (opcode) and 12 bit address



IAS MAIN REGISTERS

- **Memory buffer register (MBR):** Contains a word to be stored in memory or sent to the I/O unit, or is used to receive a word from memory or from the I/O unit.
- **Memory address register (MAR):** Specifies the address in memory of the word to be written from or read into the MBR.
- **Instruction register (IR):** Contains the 8-bit opcode instruction being executed.
- **Instruction buffer register (IBR):** Employed to hold temporarily the right hand instruction from a word in memory.
- **Program counter (PC):** Contains the address of the next instruction pair to be fetched from memory.
- **Accumulator (AC) and multiplier quotient (MQ):** Employed to hold temporarily operands and results of ALU operations. For example, the result of multiplying two 40-bit numbers is an 80-bit number; the most significant 40 bits are stored in the AC and the least significant in the MQ.

EXPANDED STRUCTURE OF IAS COMPUTER



IAS OPERATION

- Operates by repetitively performing an ***Instruction cycle***.
- Each instruction cycle consists of two subcycles. ***Instruction Fetch*** and ***Instruction Execute Cycle***
- In fetch cycle the opcode of the next instruction is loaded into the IR and the address portion is loaded into the MAR.
- This instruction may be taken from the IBR or from memory by loading a word into the MBR, and then down to the IBR, IR, and MAR.
- There is only one register that is used to specify the address in memory for a read or write and One register is used for the source or destination.
- Once the opcode is in the IR, the execute cycle is performed.
- Control circuitry interprets the opcode and executes the instruction.

IAS INSTRUCTION SET

The IAS computer had a total of 21 instructions, which can be grouped as follows:

- **Data transfer:** Move data between memory and ALU registers or between two ALU registers.
- **Unconditional branch:** Normally, the control unit executes instructions in sequence from memory. This sequence can be changed by a branch instruction, which facilitates repetitive operations.
- **Conditional branch:** The branch can be made dependent on a condition, thus allowing decision points.
- **Arithmetic:** Operations performed by the ALU.
- **Address modify:** Permits addresses to be computed in the ALU and then inserted into instructions stored in memory. This allows a program considerable addressing flexibility.

THE SECOND GENERATION : TRANSISTORS

- Transistor Based Computers
- More complex arithmetic and logic units and control units,
- The use high-level programming languages and software provided the ability to load programs,(beginning of OSeS)
- **Data channels**:-independent I/O module with its own processor & Instruction set. -relieves the CPU of a considerable processing burden.
- **Multiplexor** :- termination point for data channels, the CPU, and memory.
 - schedules access to the memory from the CPU and data channels

TRANSISTORS



- Replaced vacuum tubes
 - Smaller
 - Cheaper
 - Less heat dissipation
- Solid State device
- Made from Silicon
- Invented 1947 at Bell Labs
- William Shockley et al.



IBM

- Punched-card processing equipment
- 1953 - the 701
 - IBM's first Electronic stored program computer
 - Scientific calculations
- 1955 - the 702
 - Business applications
- Lead to 700/7000 series



An IBM 704 mainframe at NACA in 1957



THE THIRD GENERATION: INTEGRATED CIRCUITS

Microelectronic

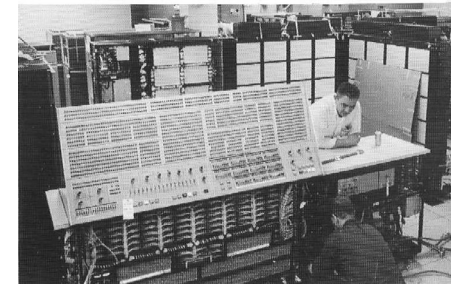
- Literally - “small electronics”
- A computer is made up of gates, memory cells and interconnections
- These can be manufactured on a semiconductor material. e.g. silicon wafer
- Jack Kilby invented the **integrated circuit** at Texas Instruments in 1958 creating a transistor , resistor and capacitor from a piece of Germanium
- Robert Noyce also independently invented IC

MOORE'S LAW

- Gordon Moore – co-founder 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
- Consequence of Moore's Law
 - Higher packing density means shorter electrical paths, giving higher performance
 - Smaller size gives increased flexibility
 - Reduced power and cooling requirements
 - Fewer interconnections increases reliability

IBM 360 SERIES

- 1964
- Replaced (& not compatible with) 7000 series
- 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!)
- Small enough to
sit on a lab bench
- \$16,000
 - \$100k+ for IBM 360



LATER GENERATIONS

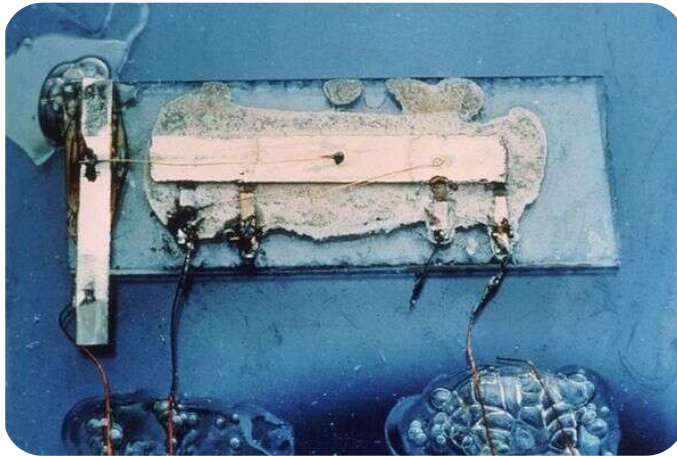
- Beyond the third generation there is less general agreement on defining generations of computers.

Table 1.2 Computer Generations

Generation	Approximate Dates	Technology	Typical Speed (operations per second)
1	1946–1957	Vacuum tube	40,000
2	1957–1964	Transistor	200,000
3	1965–1971	Small- and medium-scale integration	1,000,000
4	1972–1977	Large scale integration	10,000,000
5	1978–1991	Very large scale integration	100,000,000
6	1991–	Ultra large scale integration	>1,000,000,000

EVOLUTION OF INTEL MICROPROCESSORS

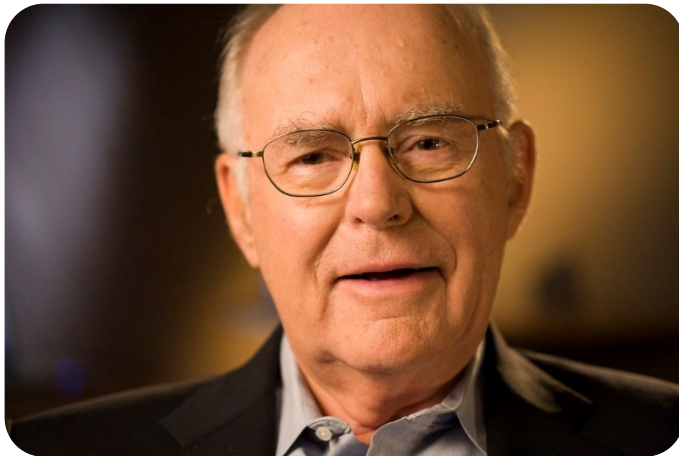
1958: First Semiconductor Integrated Circuit (IC),



- *Jack S. Kilby demonstrated the first working integrated circuit to managers at Texas Instruments. USA*
- *This was the first time electronic components were integrated onto a single substrate.*

CONT...

1965: Moore's Law



- *Gordon Moore, cofounder of Intel*
- *"The number of transistors on a microchip doubles every two years, though the cost of computers is halved."*

CONT...

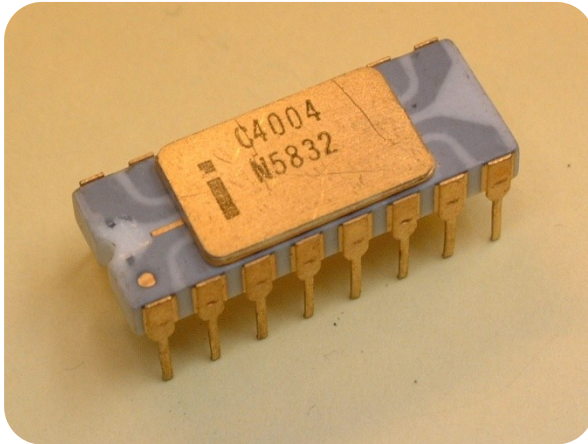
1968-1970: “Tomcat”



- *The World's First Microprocessor*
- *Was designed for the US Navy F14A “TomCat” fighter jet*

CONT...

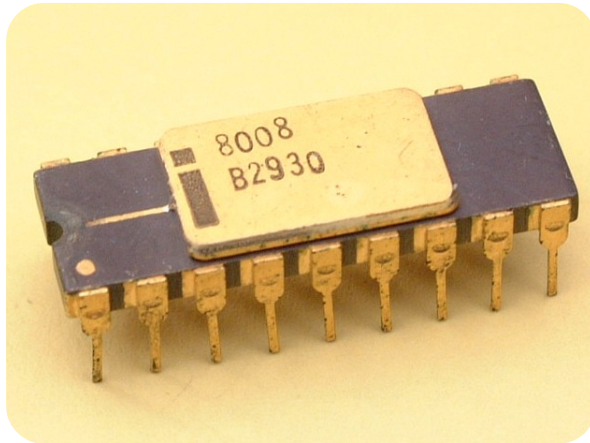
1971: Intel 4004



- *The first commercially available μP*
- *4 bit processor, made by Intel,*
- *had 2300 transistors,*
- *speed up to 740 KHz*

CONT...

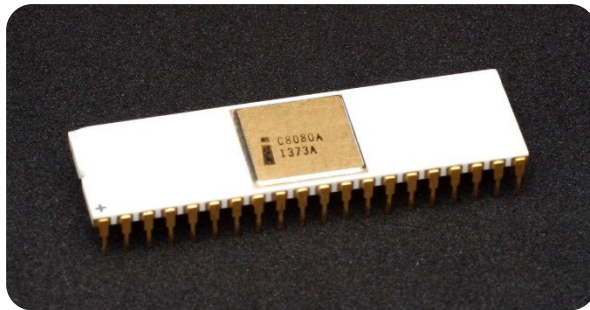
1972: Intel 8008



- *8 bit processor,*
- *14 Bit Address width*
- *Had 3,500 Transistors*
- *Originally designed for Datapoint Corp. as a CRT display controller*

CONT...

1974: Intel 8080



- *8 bit processor with 16 bit address bus*
- *Speed: 2 MHz to 3.125 MHz*
- *6000 transistors*
- *Apple II -- Steve Jobs and Steve Wozniak 1976, Apple inc.*
- *Bill Gates and Allen Paul: BASIC, 1975 --> Microsoft corp*

CONT...

1978: Intel 8086/8088



- *16 bit processor with 20 bit address bus*
- *Speed: 5 MHz to 10 MHz*
- *29,000 Transistors*
- *1979: **8088** a slightly modified chip with an external 8-bit data bus*
- *8088 (used in the IBM PC)*
- *8086 gave rise to the x86 architecture, which eventually became Intel's most successful line of processors*

CONT...

1982: Intel 80286



- *16 bit processor with 24 bit address bus*
- *Speed: 4 MHz to 25 MHz*
- *134,000 Transistors*
- *16 MB of physical MEM and 1 GB of virtual mem*
- *IBM PC/AT in 1984, IBM PS/2 Model 50 and 60*

CONT...

1985: Intel 80386



- *32 bit processor with 32 bit address bus*
- *Speed: 12 MHz to 40 MHz*
- *275,000 Transistors*
- *up to 4 GB of memory.*
- *Memory paging and enhanced I/O permission features*

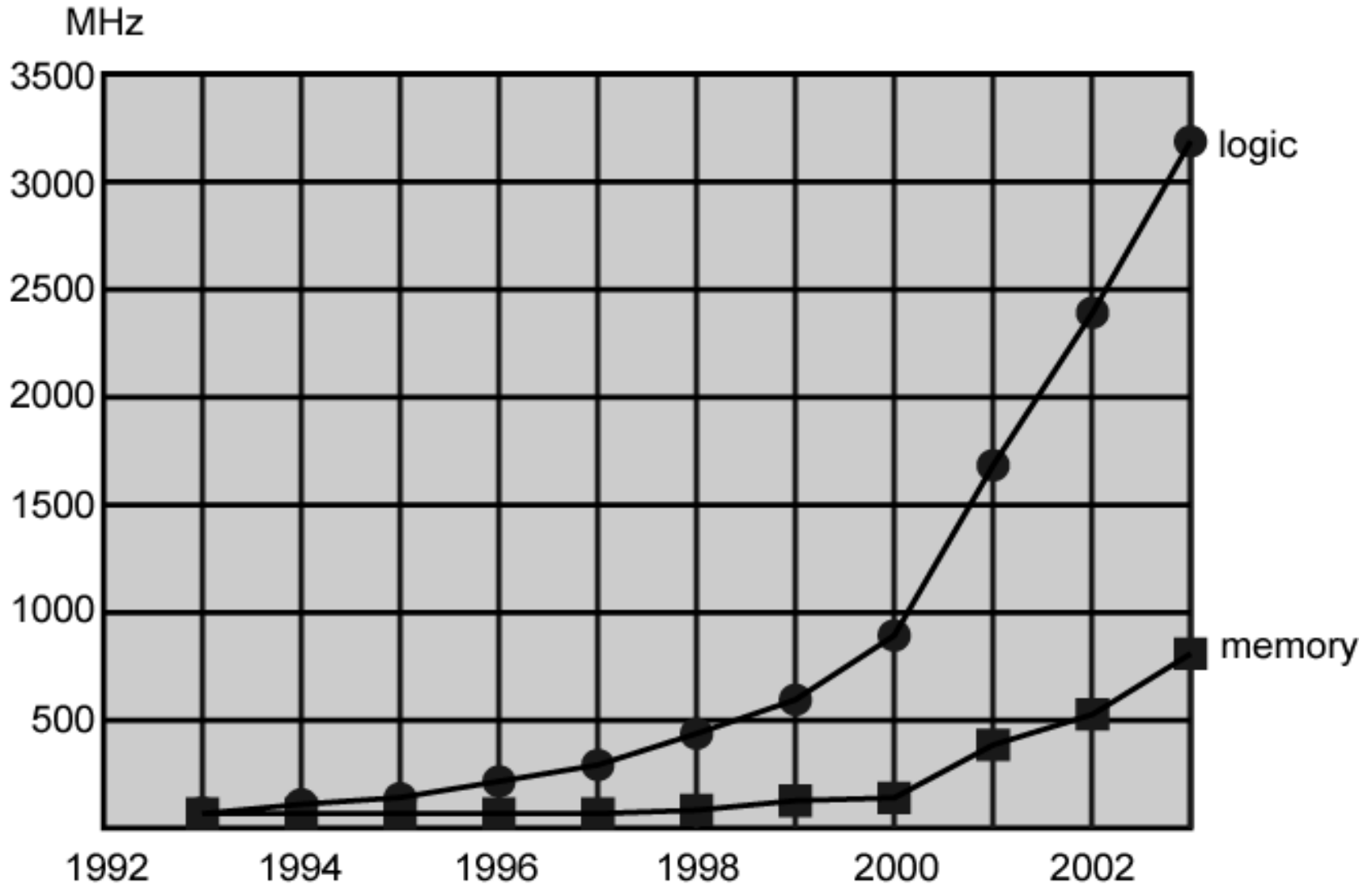
CONT. . .

- Intel 80486
- Intel Pentium , Pentium II, Pentium III, Pentium 4,
- Core, Core 2,
- Core i3, i5, i7, i9

PERFORMANCE BALANCE

- Processor speed increased
- Memory capacity increased
- Memory speed lags behind processor speed

LOGIC AND MEMORY



SOLUTIONS

- Increase number of bits retrieved at one time
 - Make DRAM “wider” rather than “deeper” by using wide bus data paths
- Change DRAM interface
 - including a cache or other buffering scheme on the DRAM chip
- Reduce frequency of memory access
 - This includes the incorporation of **one or more caches on the processor** chip as well as on an **off-chip cache** close to the processor chip.
- Increase interconnection bandwidth
 - High speed buses and Hierarchy of buses

HOW TO INCREASE PROCESSOR SPEED

- Increase the hardware speed of the processor.
 - Shrinking the size of the logic gates and packed together more tightly and increase clock rate
 - An increase in clock rate means that individual operations are executed more rapidly.
- Increase the size and speed of caches
 - Dedicating a portion of the processor chip itself to the cache, cache access times drop significantly.
- Make changes to the processor organization and architecture
 - Using parallelism (instruction pipelining and superscalar)

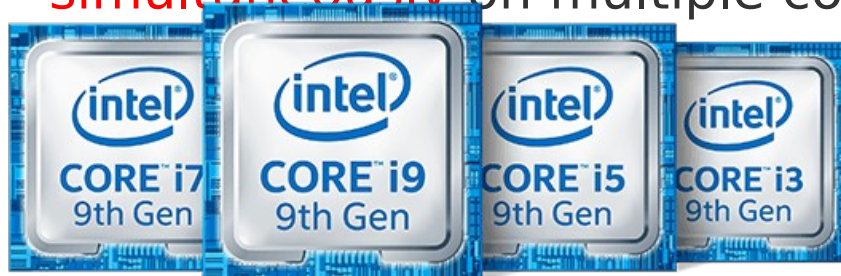
HOW TO INCREASE PROCESSOR SPEED

- Instruction pipelining works like assembly line
 - Different stages of execution of different instructions at same time along pipeline
- A superscalar processor is a CPU that implements a form of parallelism called instruction-level parallelism within a single processor.
 - Scalar processor:- execute at most one single instruction per clock cycle,
 - Superscalar processor can execute more than one instruction during a clock cycle by simultaneously dispatching multiple instructions to different duplicate functional units on the processor.
 - Each functional unit is just an execution resource inside the CPU core, like an arithmetic logic unit (ALU), floating point unit (FPU), a bit shifter, or a multiplier.
 - Most superscalar CPUs are also pipelined,

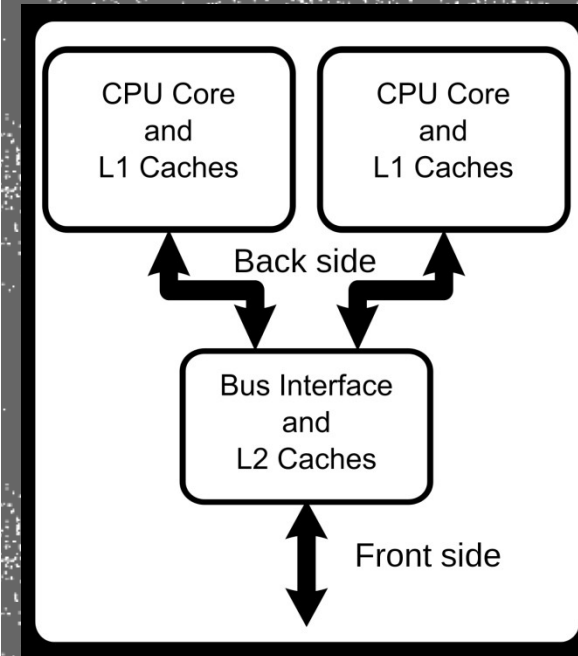
PROBLEMS WITH OVERCLOCKING AND LOGIC DENSITY

- Power
 - Power density increases with density of logic and clock speed
 - Dissipating large amount of heat and controlling the heat is becoming difficult which will lead to permanent damage of transistor
- RC delay
 - Speed at which electrons flow **limited** by **resistance** and **capacitance** of metal wires connecting them
 - Delay increases as RC product increases
 - Wire interconnects thinner, increasing resistance
 - Wires closer together, increasing capacitance
- Solution:
 - More emphasis on other organizational and architectural approaches

- A multi-core processor is a computer processor **on a single IC** with two or **more separate processing units**, called **cores**, each of which reads and executes program instructions.
- Single processor can run instructions on separate cores at the same time, increasing overall speed for programs.
- In particular, possible gains are limited by the fraction of the software **that can run in parallel simultaneously** on multiple cores



NEW APPROACH - MULTIPLE CORES





PART 2

Chapter 1

Computer Architecture and Organization

LEARNING OBJECTIVES

- Understand the basic elements of an instruction cycle and the role of interrupts.
- Describe the concept of interconnection within a computer system.

WHAT WE ALREADY KNOW?

- The von Neumann architecture and is based on three key concepts:
 - **Data** and **instructions** are stored in a single read-write memory.
 - The contents of this memory are **addressable by location**, without regard to the type of data contained there.
 - Execution occurs in a **sequential fashion** (unless explicitly modified) from one instruction to the next.
- For each instructions has a unique operation code
- CU accepts the Opcode decode it then issues the control signals

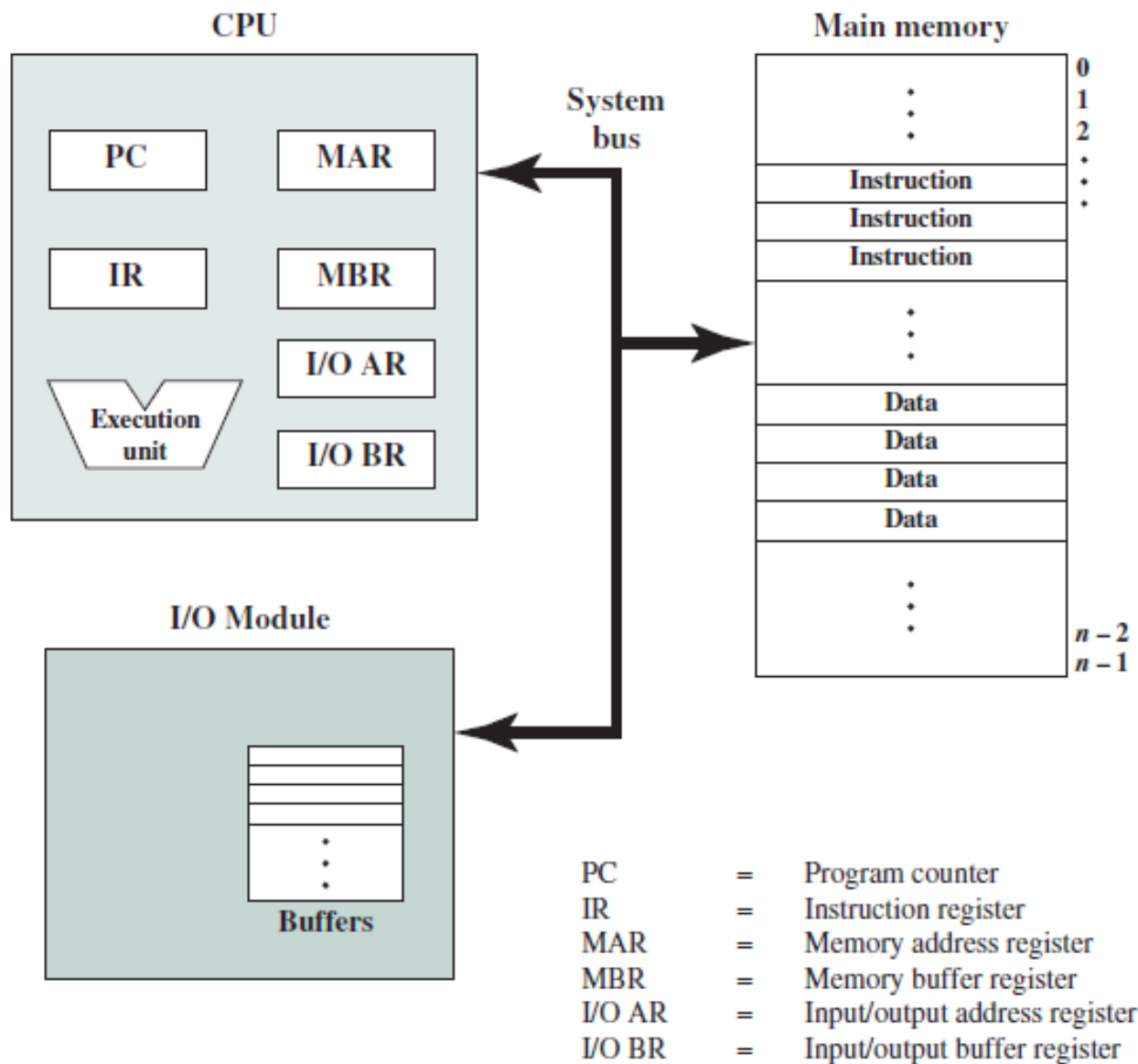
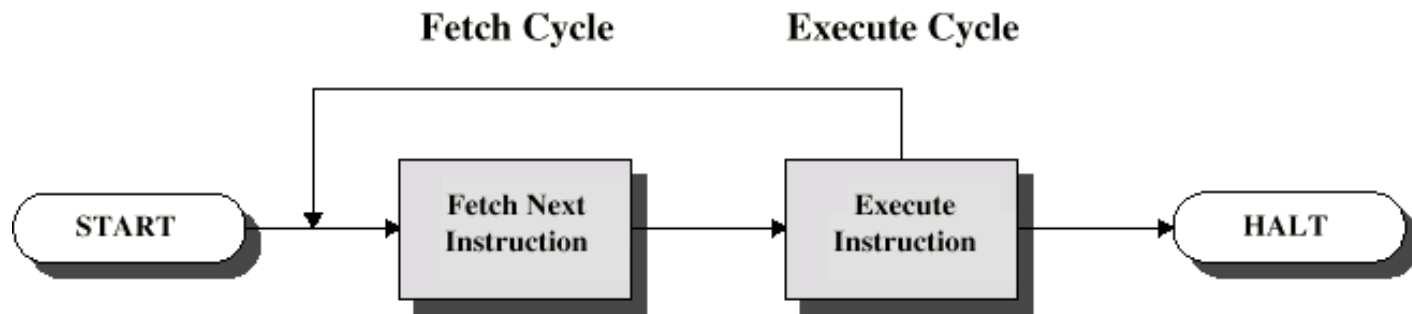


Figure 3.2 Computer Components: Top-Level View

HOW COMPUTER EXECUTE INSTRUCTIONS?

- Two steps:
 - Fetch
 - Execute
- The instruction execution may involve several operations and **depends on the nature of the instruction**



INSTRUCTION CYCLE

- The processing required for a single instruction is called an **instruction cycle**
- Fetch cycle
 - Program Counter (PC) holds address of next instruction to fetch
 - Processor fetches instruction from memory location pointed to by PC
 - Increment PC
 - Unless told otherwise
 - The Opcode of the Instruction loaded into Instruction Register (IR)
- Execute cycle
 - Processor interprets instruction and performs required actions

CONT...

In general, these actions fall into four categories

- **Processor-memory**

- data transfer between CPU and main memory

- **Processor I/O**

- Data transfer between CPU and I/O module

- **Data processing**

- Some arithmetic or logical operation on data

- **Control**

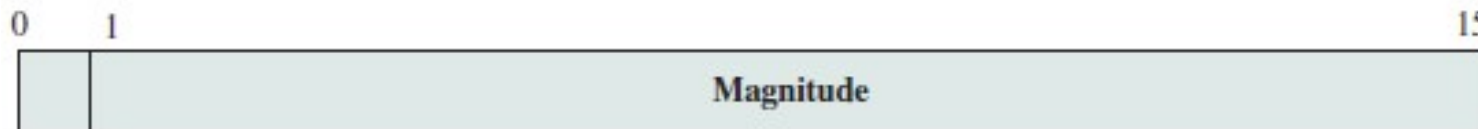
- Alteration of sequence of operations
- e.g. jump

Combination of above

EXAMPLE FOR A SIMPLE HYPOTHETICAL MACHINE



(a) Instruction format



(b) Integer format

Program counter (PC) = Address of instruction
 Instruction register (IR) = Instruction being executed
 Accumulator (AC) = Temporary storage

(c) Internal CPU registers

0001 = Load AC from memory
 0010 = Store AC to memory
 0101 = Add to AC from memory

(d) Partial list of opcodes

Memory				
300	1	9	4	0
301	5	9	4	1
302	2	9	4	1
⋮				
940	0	0	0	3
941	0	0	0	2

Figure 3.4 Characteristics of a Hypothetical Machine
 Computer Architecture and Organization

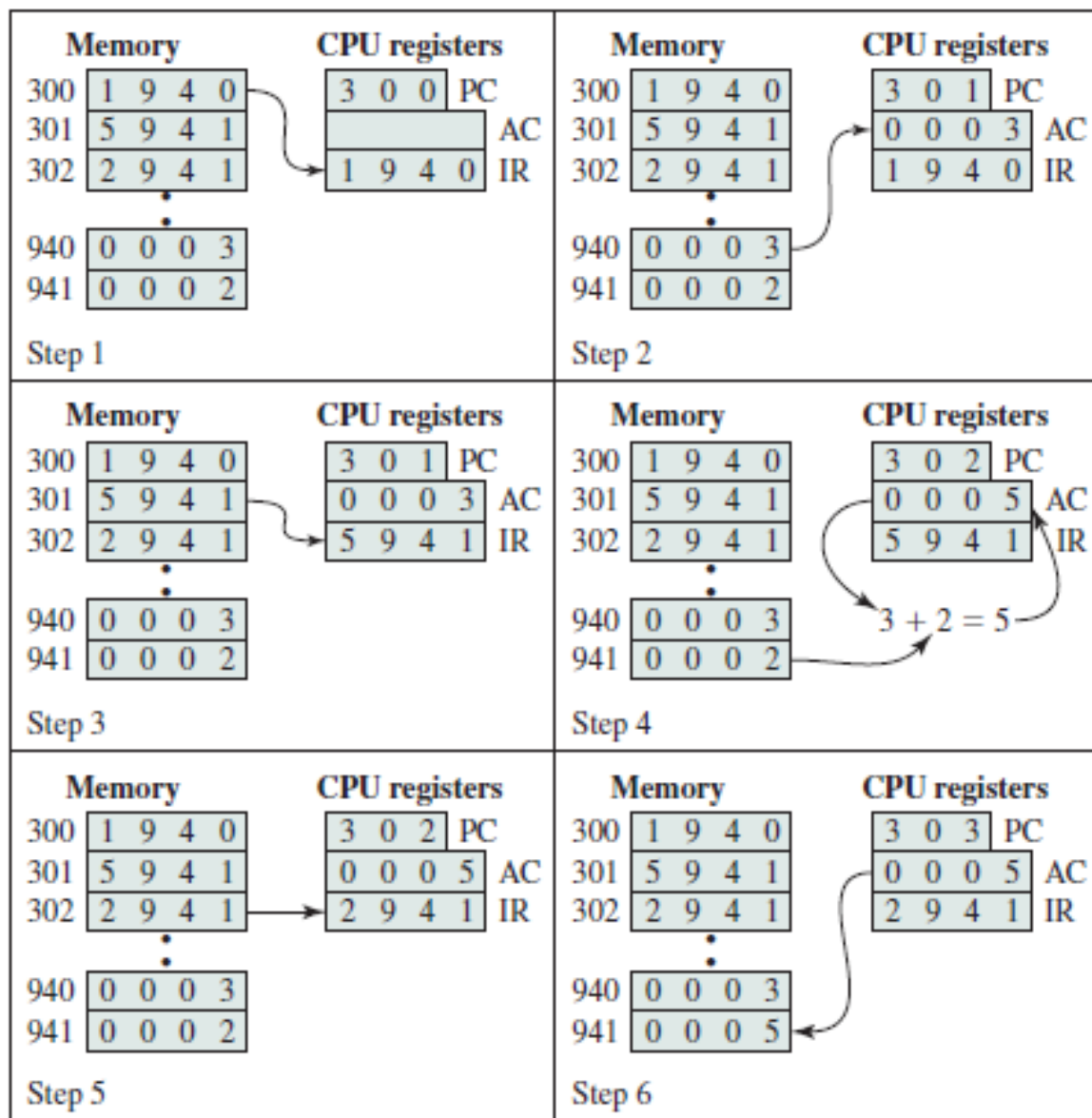


Figure 3.5 Example of Program Execution (contents of memory and registers in hexadecimal)

CONT...

- Some older processors, included instructions that contain **more than one memory address** in a single instructions
- Thus, the execution cycle for a particular instruction on such processors could involve **more than one reference** to memory.
 - For example, the PDP-11 processor **ADD B,A** instruction
- Also, instead of memory references, an instruction may specify an **I/O operation**.
- With these additional considerations in mind, **Instruction cycle state diagram** provides a more detailed look at the basic instruction cycle
 - For any given instruction cycle, some states may be **null** and others may be visited **more than once**.

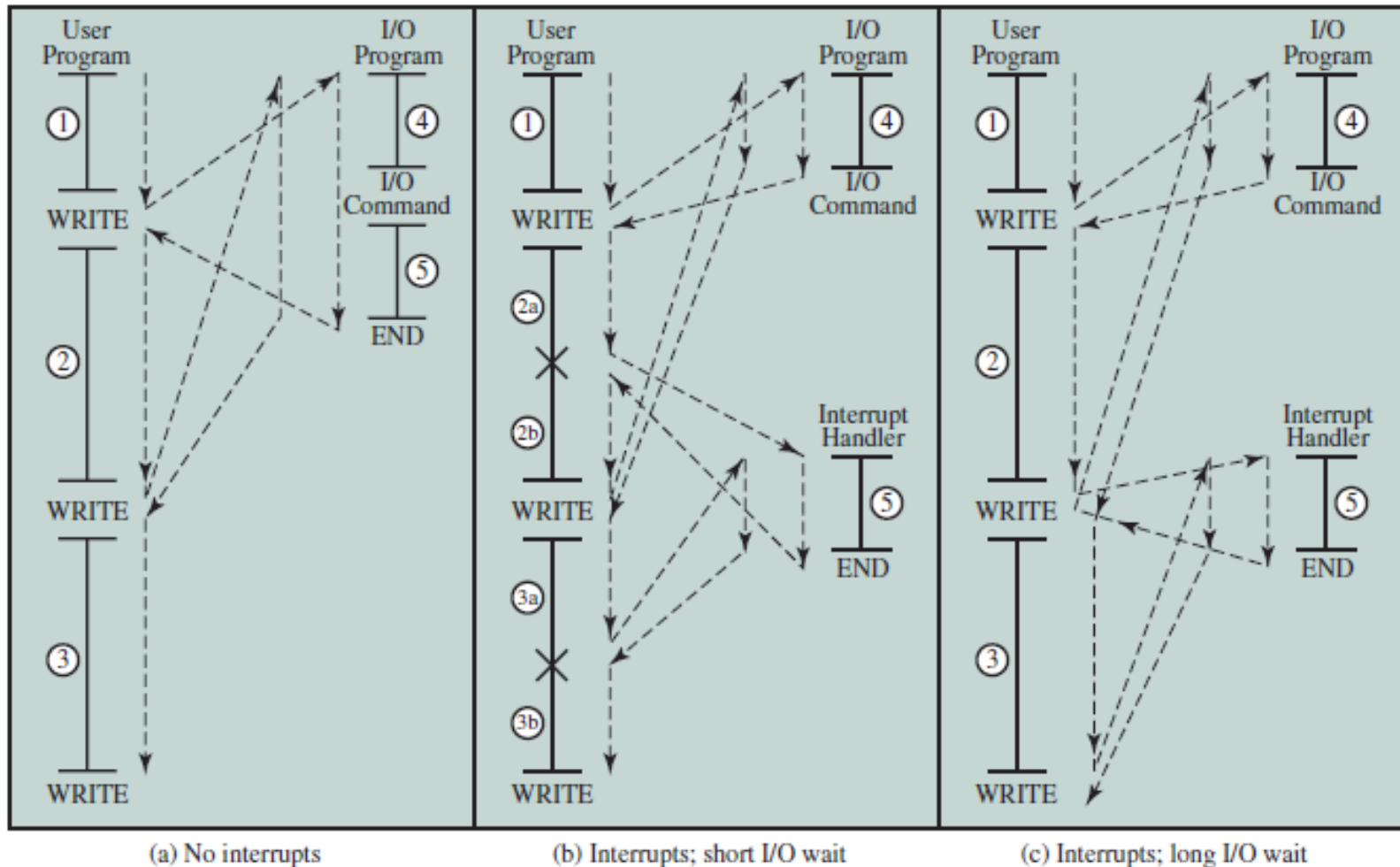


Computer Architecture and Organization

INTERRUPTS

- Mechanism by which other modules (e.g. I/O) may interrupt normal sequence of processing
- Program
 - e.g. overflow, division by zero
- Timer
 - Generated by internal processor timer
 - Used in pre-emptive multi-tasking
- I/O
 - from I/O controller
- Hardware failure
 - e.g. memory parity error

Interrupts are provided primarily as a way to improve processing efficiency



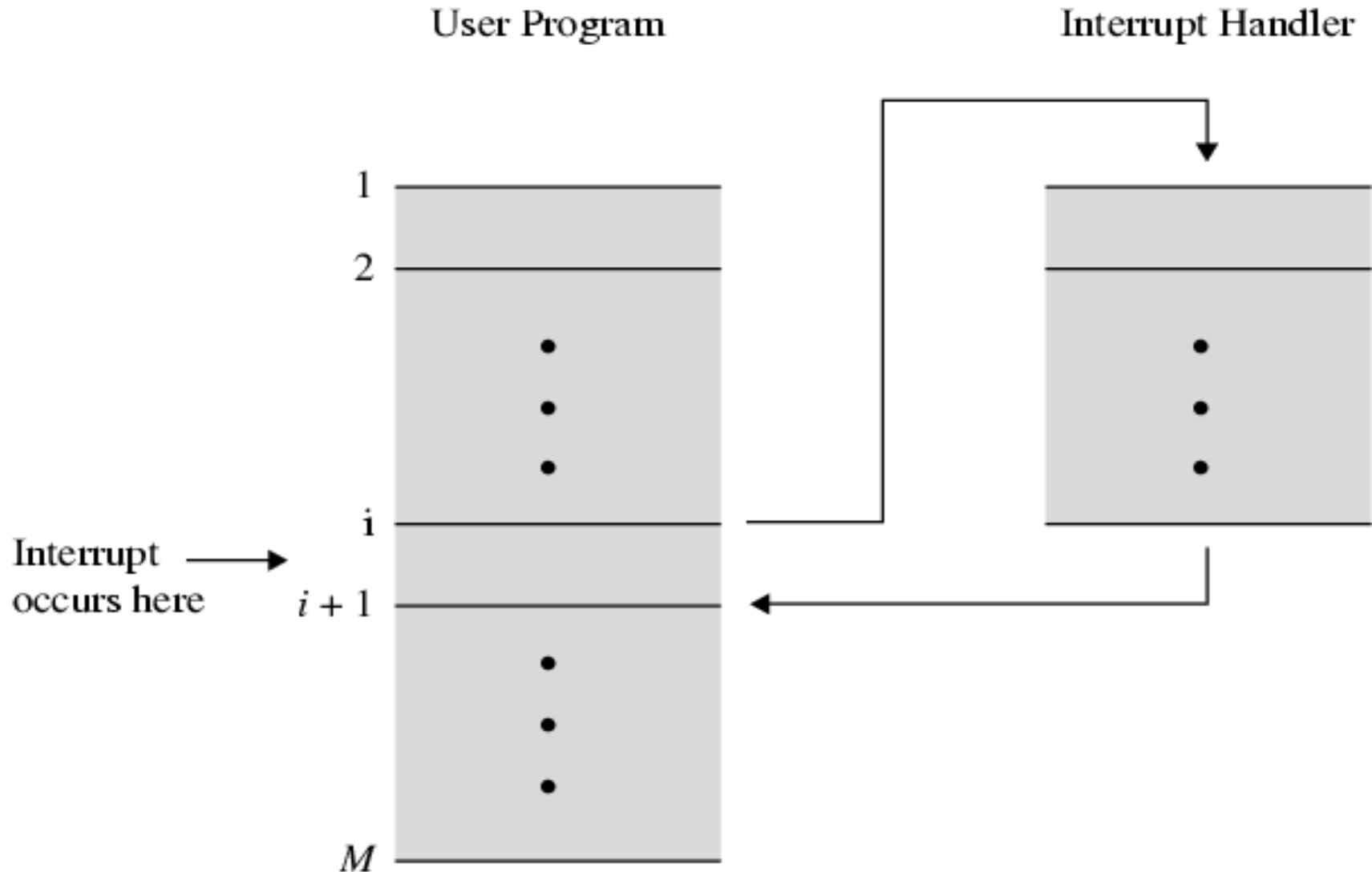
✕ = interrupt occurs during course of execution of user program

Figure 3.7 Program Flow of Control without and with Interrupts

INTERRUPT CYCLE

- Added to instruction cycle to accommodate **interrupts**
- Processor checks for interrupt
 - Indicated by an interrupt signal
- If no interrupt, fetch next instruction
- If interrupt pending:
 - Suspend execution of current program
 - Save context
 - Set PC to start address of interrupt handler routine
 - Process interrupt
 - Restore context and continue interrupted program

TRANSFER OF CONTROL VIA INTERRUPTS



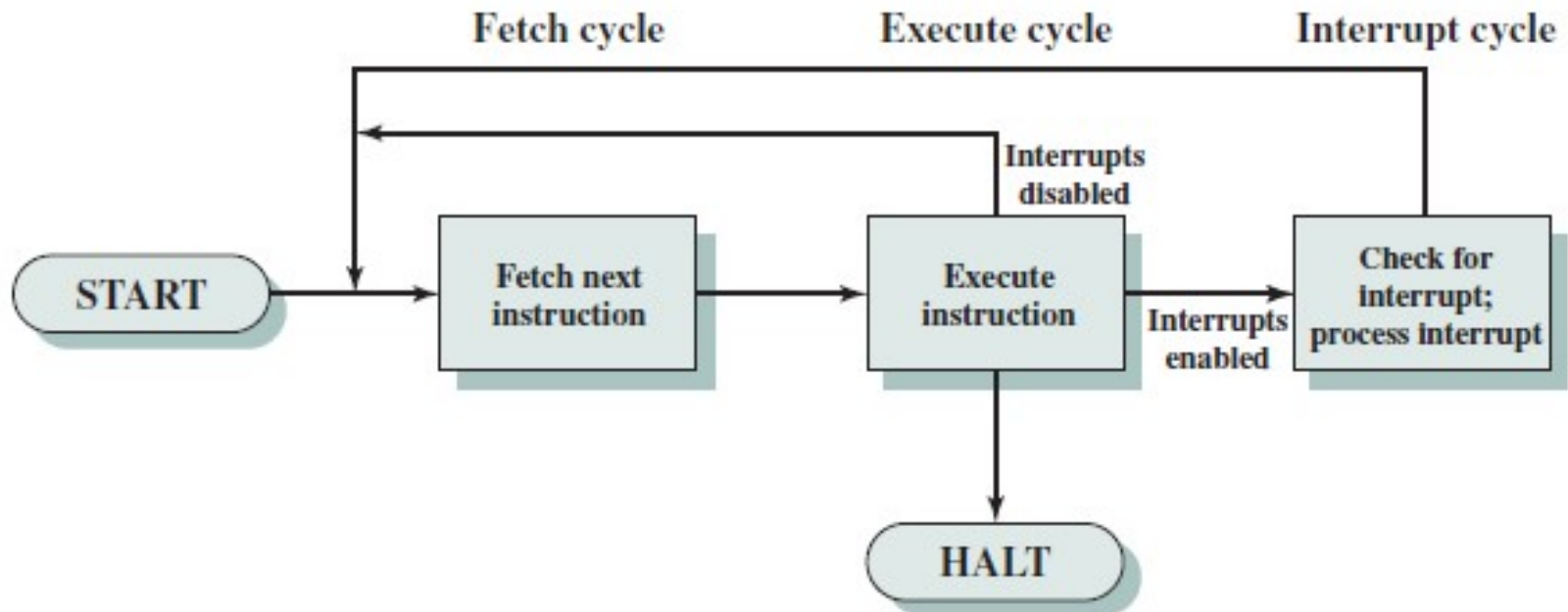
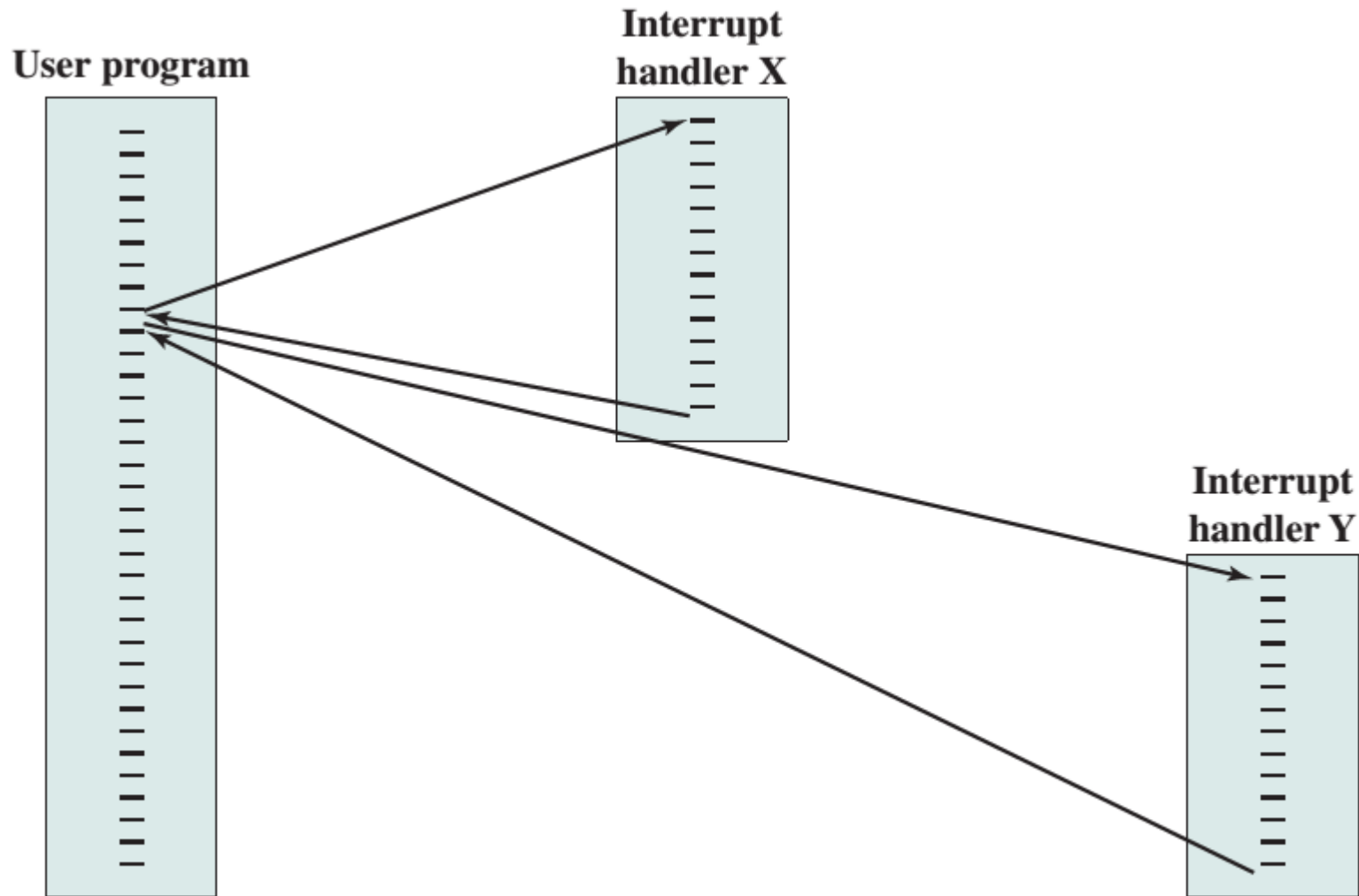


Figure 3.9 Instruction Cycle with Interrupts

MULTIPLE INTERRUPTS

- Two approaches
- A disabled interrupt
 - Simply means that the processor can and will ignore that interrupt request signal. If an interrupt occurs during this time, it generally remains pending and will be checked by the processor after the processor has enabled interrupts.
 - The drawback to the preceding approach is that it does not take into account relative priority or time-critical needs.
- A second approach is to define priorities for interrupts and to allow an interrupt of higher priority to cause a lower-priority interrupt handler to be itself interrupted



(a) Sequential interrupt processing

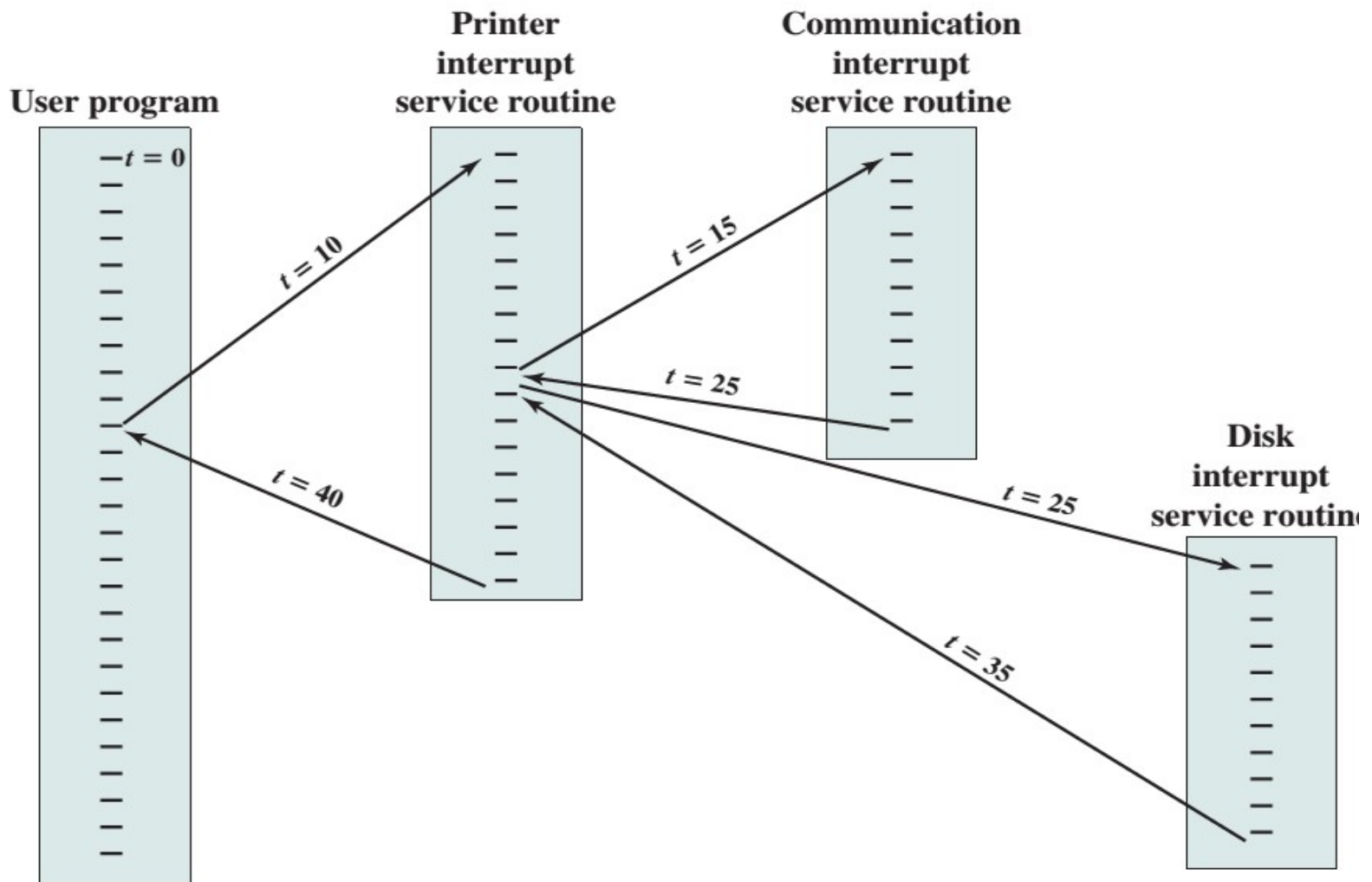


Figure 3.14 Example Time Sequence of Multiple Interrupts

Communication Priority > Disk Priority > Printer Priority

INSTRUCTION CYCLE STATE DIAGRAM, WITH INTERRUPTS

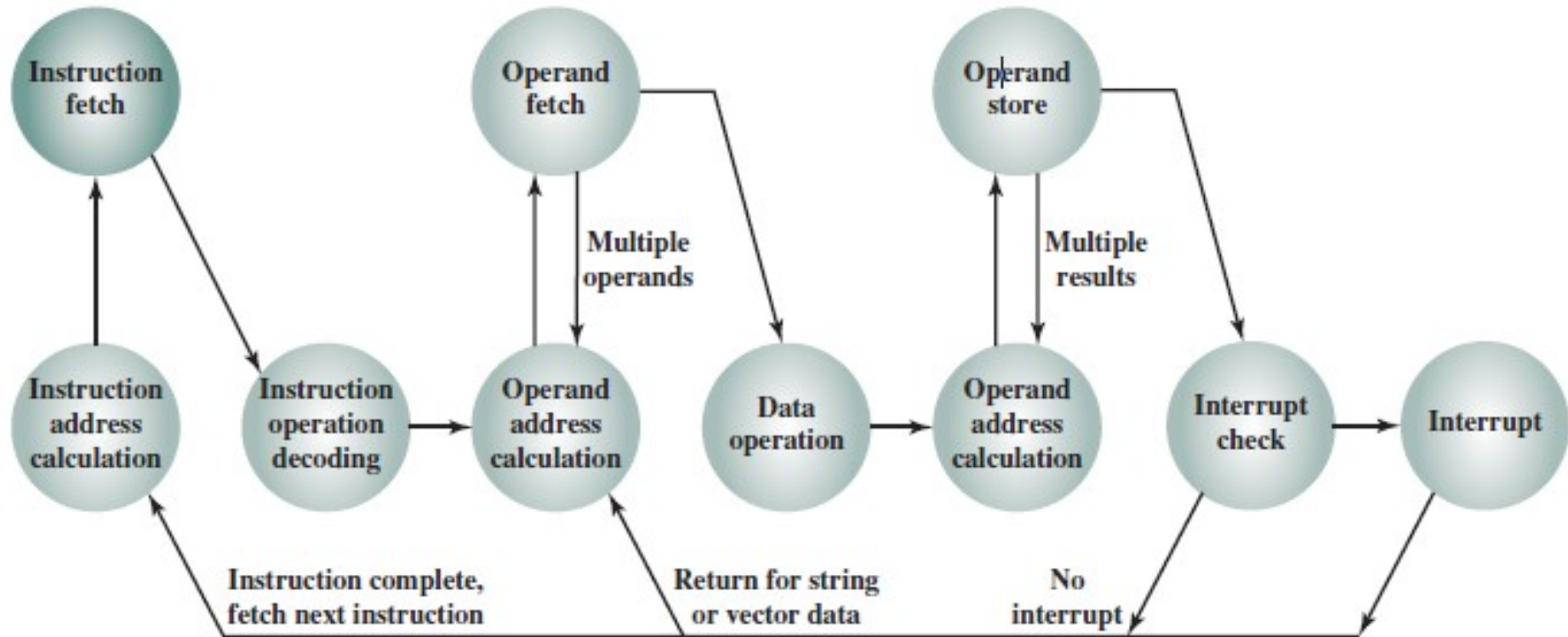
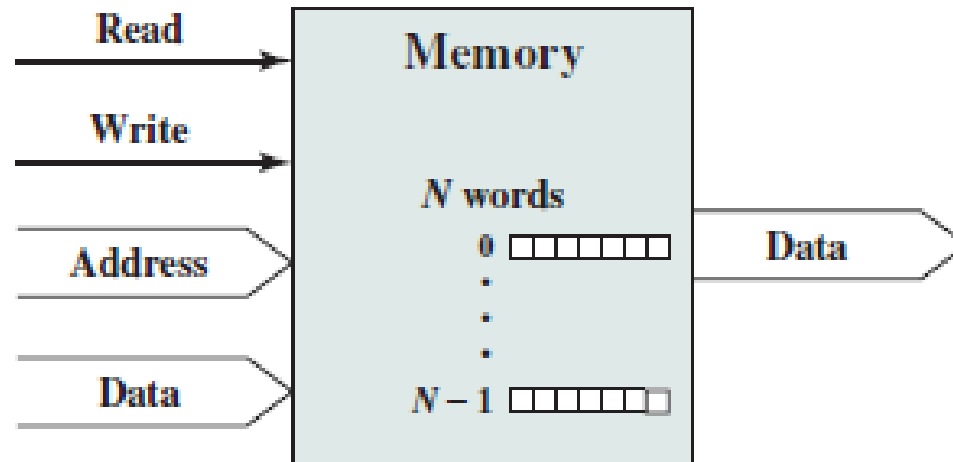


Figure 3.12 Instruction Cycle State Diagram, with Interrupts

INTERCONNECTION STRUCTURES

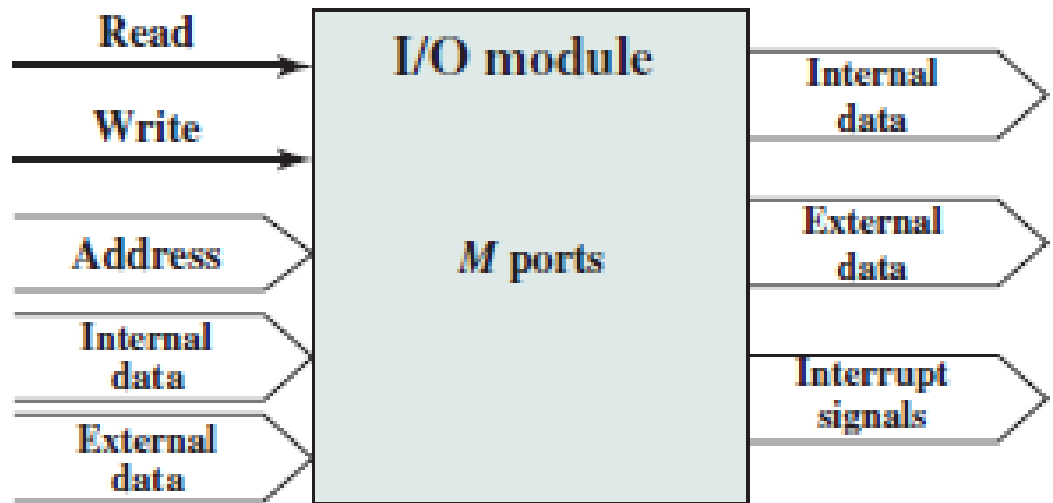
- A computer consists of a set of components or modules of three basic types (processor, memory, I/O) that communicate with each other.
- The collection of paths connecting the various modules is called the **interconnection structure**.
- The design of this structure **will depend on the exchanges** that must be made among modules.

MEMORY



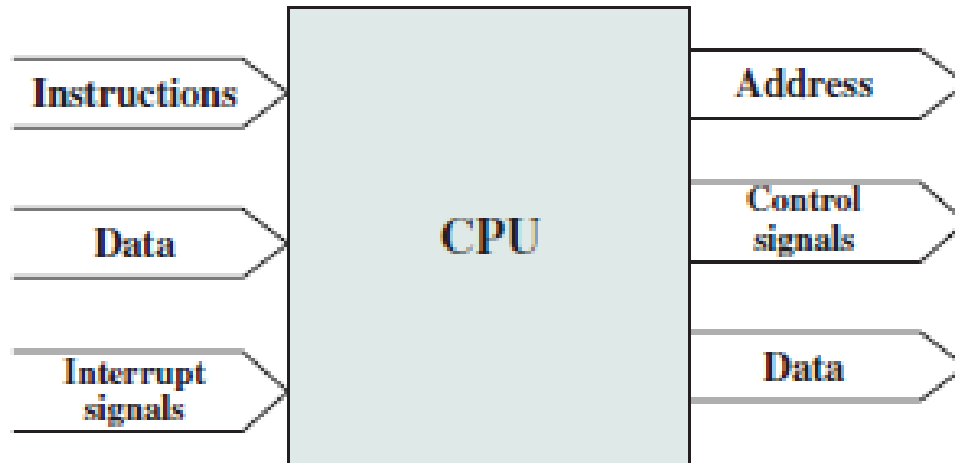
- Typically, a memory module will consist of N words of equal length.
- Each word is assigned a unique numerical address (0, 1, ..., $N-1$).
- A word of data can be **read** from or **written** into the memory.
- The nature of the operation is indicated **by read and write control signals**.
- The location for the operation is specified by an **address**.

I/O MODU



- There are two operations; **read and write**.
- Further, an I/O module may control more than one external device. We can refer to each of the interfaces to an external device as a **port** as a unique address (e.g., 0, 1, c , $M-1$).
- In addition, there are external data paths for the input and output of data with an external device.
- Finally, an I/O module may be able to **send interrupt signals** to the processor.

PROCESSOR



- The processor reads in **instructions** and **data**, writes out **data** after processing, and **uses control signals** to control the overall operation of the system.
- It also receives interrupt signals

BUS INTERCONNECTION

- Bus is a **shared** communication pathway connecting two or more devices
- Usually **broadcast**
- Often grouped
 - A number of channels in one bus
 - e.g. 32 bit data bus is 32 separate single bit channels
- Only one device at a time can successfully transmit
- Power lines may not be shown
- Computer systems contain a number of different buses
- A bus that connects major computer components (processor, memory, I/O) is called a **system bus**.
- Bus lines can be classified into three functional groups: **data, address, and control lines**.

DATA BUS

- The data lines (Data Bus) provide a path for **moving data** among system modules.
- The width of the data bus may consist of 32, 64, 128, or even more separate lines.
- The number of lines determines how many bits can be transferred at a time.
- The width of the data bus is a **key factor** in determining overall system performance.
- For example, if the data bus is **32 bits wide** and each instruction is **64 bits long**, then the processor must access the memory module **twice during each instruction cycle**.

ADDRESS BUS

- The address lines are used to designate the **source** or **destination** of the data on the data bus.
 - For example, if the processor wishes to read a word (8, 16, or 32 bits) of data from memory, it puts the address of the desired word on the address lines.
- The width of the address bus **determines the maximum possible memory capacity of the system**
- Furthermore, the address lines are generally also used to address **I/O ports**.
- Typically, the higher-order bits are used to select a particular module on the bus, and the lower-order bits select a memory location or I/O port within the module.
- For example, on an 8-bit address bus, address 01111111 and below might reference locations in a memory module (module 0) with 128 words of memory, and address 10000000 and above refer to devices attached to an I/O module (module 1).

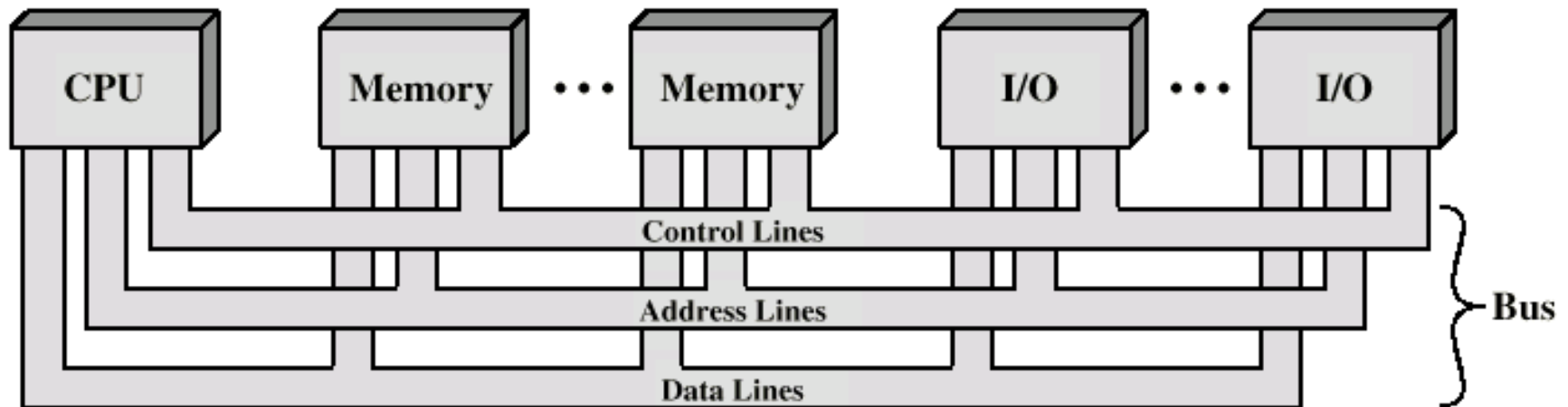
CONTROL BUS

- The control lines are **used to control the access** to and **the use of the data and address lines**.
- Because the data and address **lines are shared** by all components, there must be a means of controlling their use.
- Control signals transmit both **command** and **timing information** among system modules.
 - Timing signals indicate the validity of data and address information.
 - Command signals specify operations to be performed

TYPICAL CONTROL LINES

- **Memory write:** causes data on the bus to be written into the addressed location
- **Memory read:** causes data from the addressed location to be placed on the bus
- **I/O write:** causes data on the bus to be output to the addressed I/O port
- **I/O read:** causes data from the addressed I/O port to be placed on the bus
- **Transfer ACK:** indicates that data have been accepted from or placed on the bus
- **Bus request:** indicates that a module needs to gain control of the bus
- **Bus grant:** indicates that a requesting module has been granted control of the bus
- **Interrupt request:** indicates that an interrupt is pending
- **Interrupt ACK:** acknowledges that the pending interrupt has been recognized
- **Clock:** is used to synchronize operations
- **Reset:** initializes all modules.

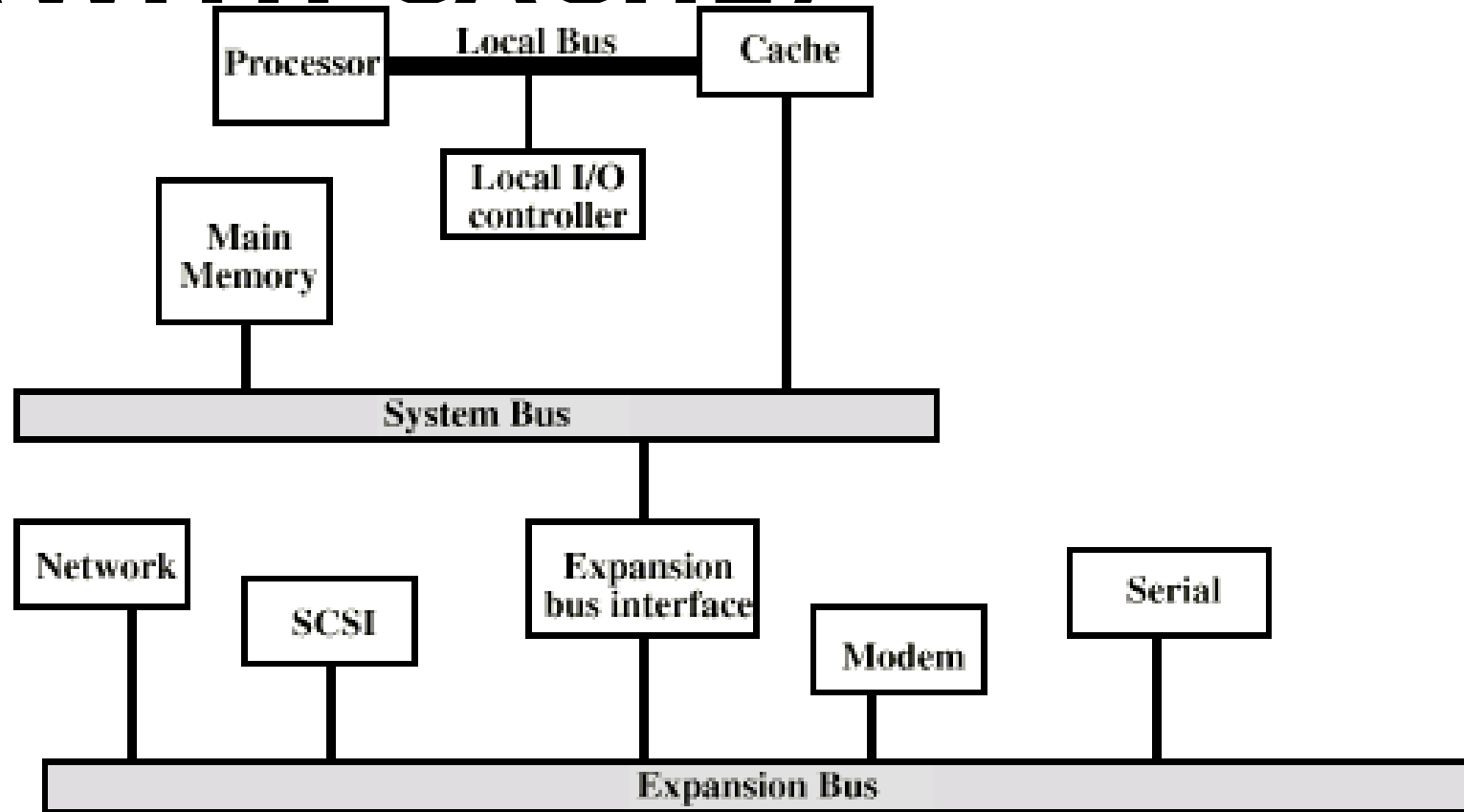
BUS INTERCONNECTION SCHEME



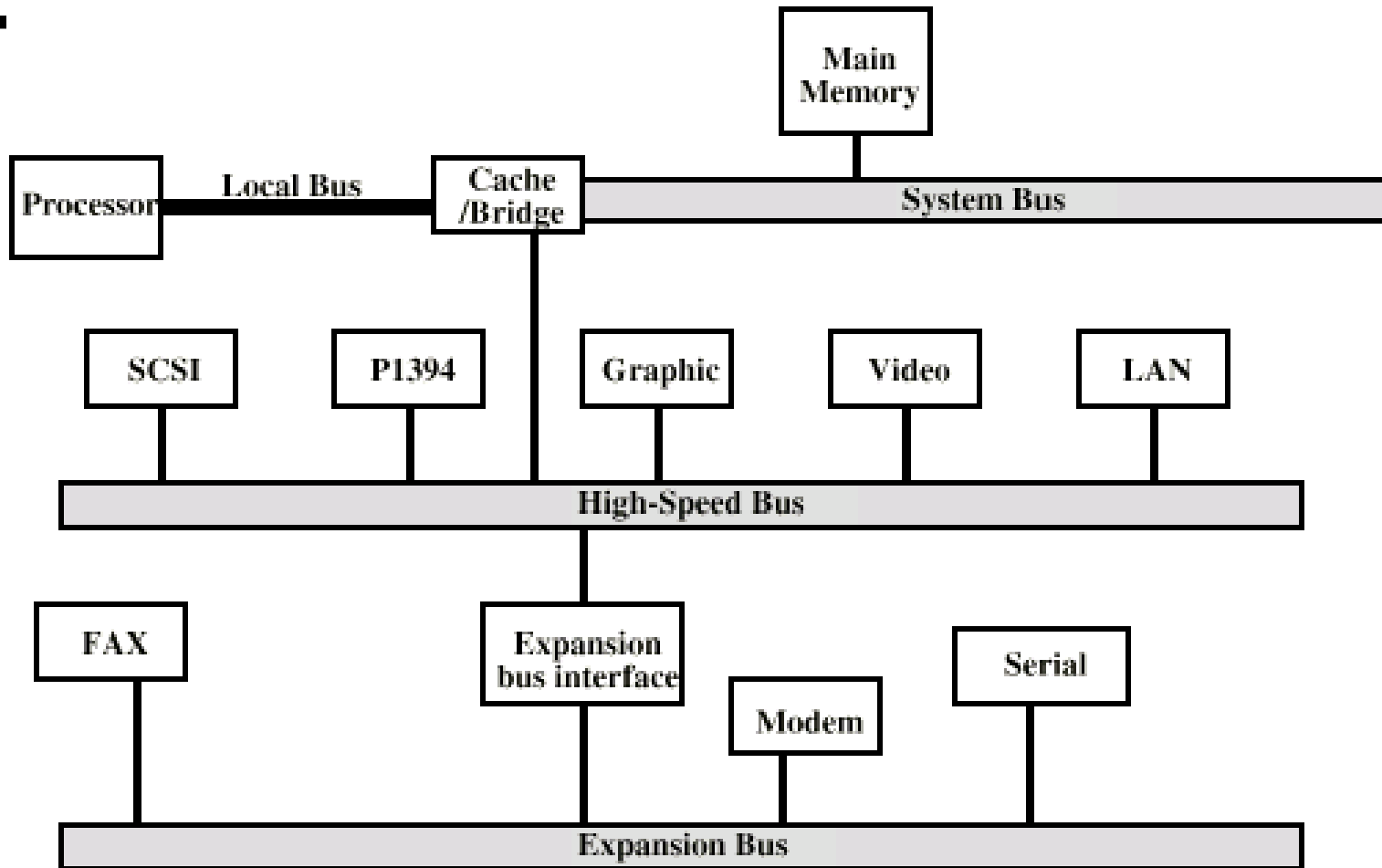
SINGLE BUS PROBLEMS

- Lots of devices on one bus leads to:
 - Propagation delays
 - Long data paths mean that co-ordination of bus use can adversely affect performance
- Most systems use multiple buses to overcome these problems

TRADITIONAL (ISA) (WITH CACHE)



HIGH PERFORMANCE PISC



BUS TYPES

- Dedicated
 - Separate data & address control lines
- Multiplexed
 - Shared lines
 - Time multiplexed
 - Advantage - fewer lines
 - Disadvantages
 - More complex control
 - Reduction performance

BUS ARBITRATION

- Centralized
 - Single hardware device controlling bus access
 - Bus Controller
 - Arbiter
 - May be part of CPU or separate
- Distributed
 - Each module may claim the bus
 - Control logic on all modules