



# **CSC2222: COMPUTER SYSTEMS II**

**M S HIRUNI PEIRIS**

**DEPARTMENT OF COMPUTER SCIENCE**

**UNIVERSITY OF RUHUNA**

# PROCESSOR STRUCTURE AND FUNCTIONS

-

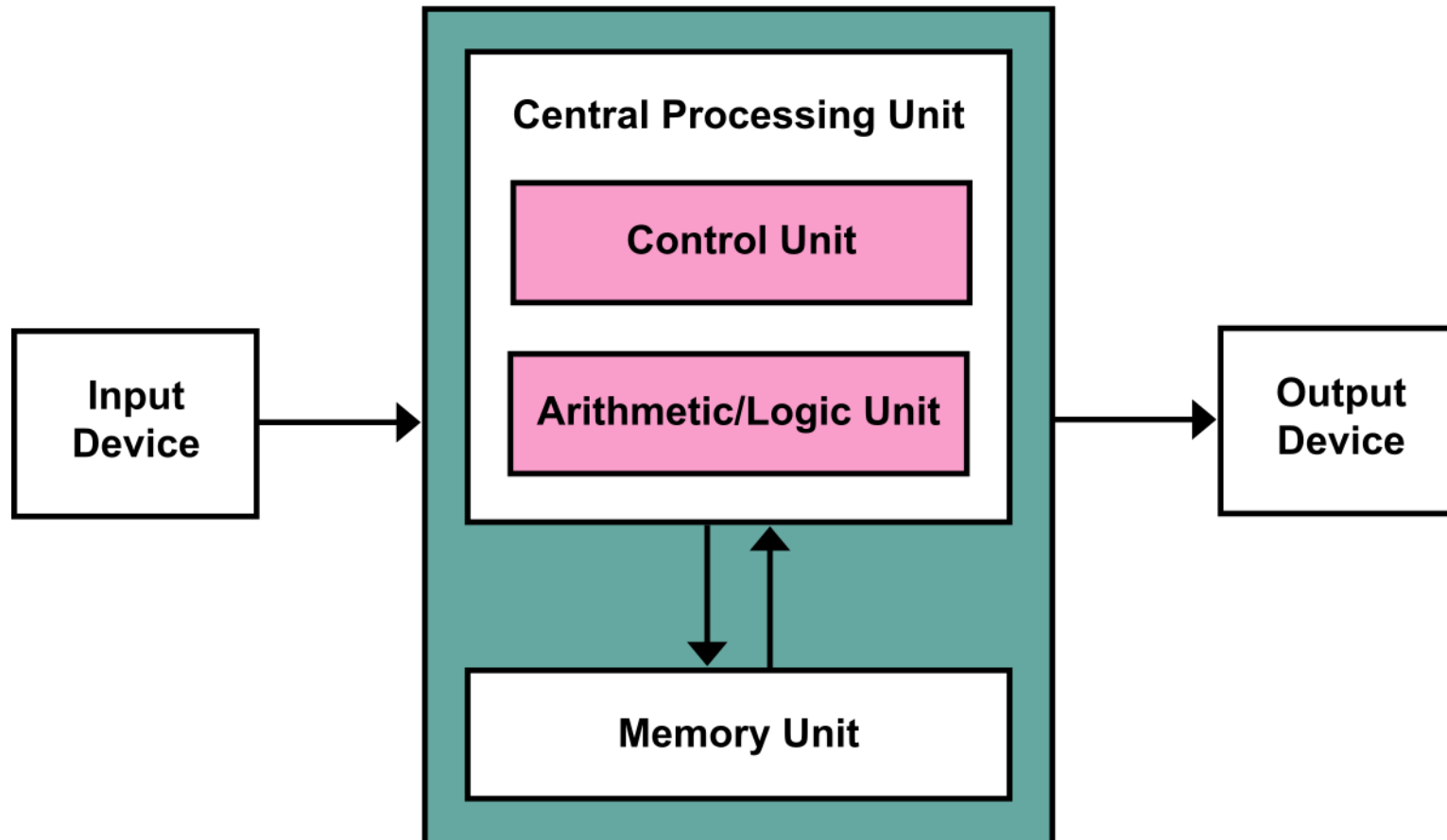
# LEARNING OBJECTIVES

- At the end of this lecture you will be able to
  - Illustrate CPU architecture
  - Illustrate register types and their structure
  - Demonstrate the complete instruction cycle

# OUTLINE

- Processor Organization
- Register Organization
- The Instruction Cycle

# VON NEUMANN ARCHITECTURE



# VON NEUMANN ARCHITECTURE

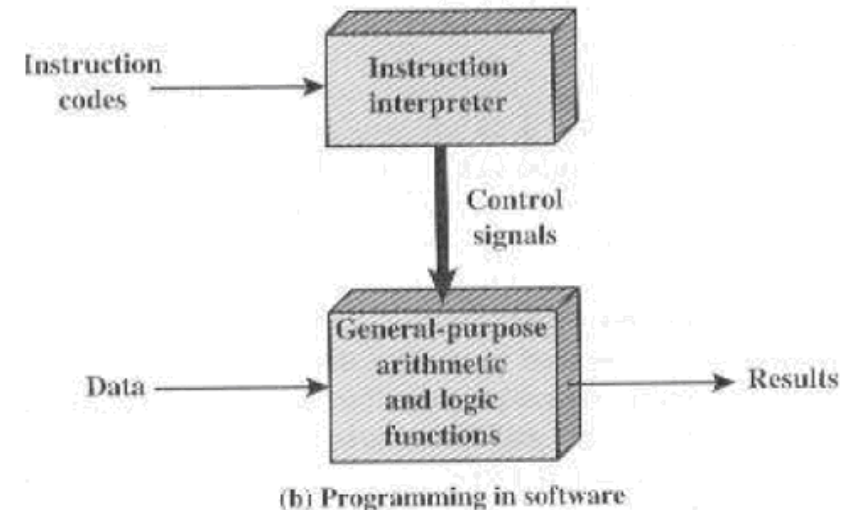
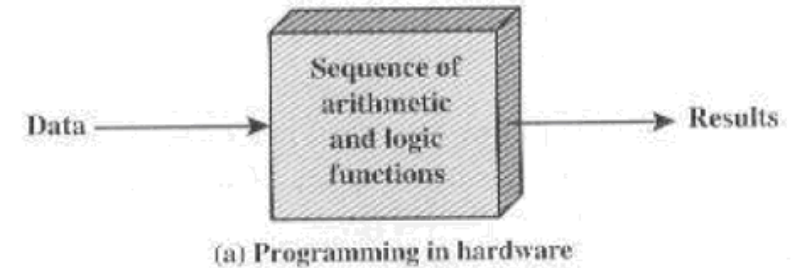
- Three key concepts:
  1. Data and instructions are stored in a single read-write memory
  2. The contents of this memory are addressable by location, without regard to the type of data contained here
  3. Execution occurs in a sequential fashion from one instruction to the next

# WHAT IS HARDWIRED PROGRAMMING?

- A set of basic logic components can be combined in a way to perform arithmetic and logical operations on the data
- The process of connecting these components produce a program in the form of hardware

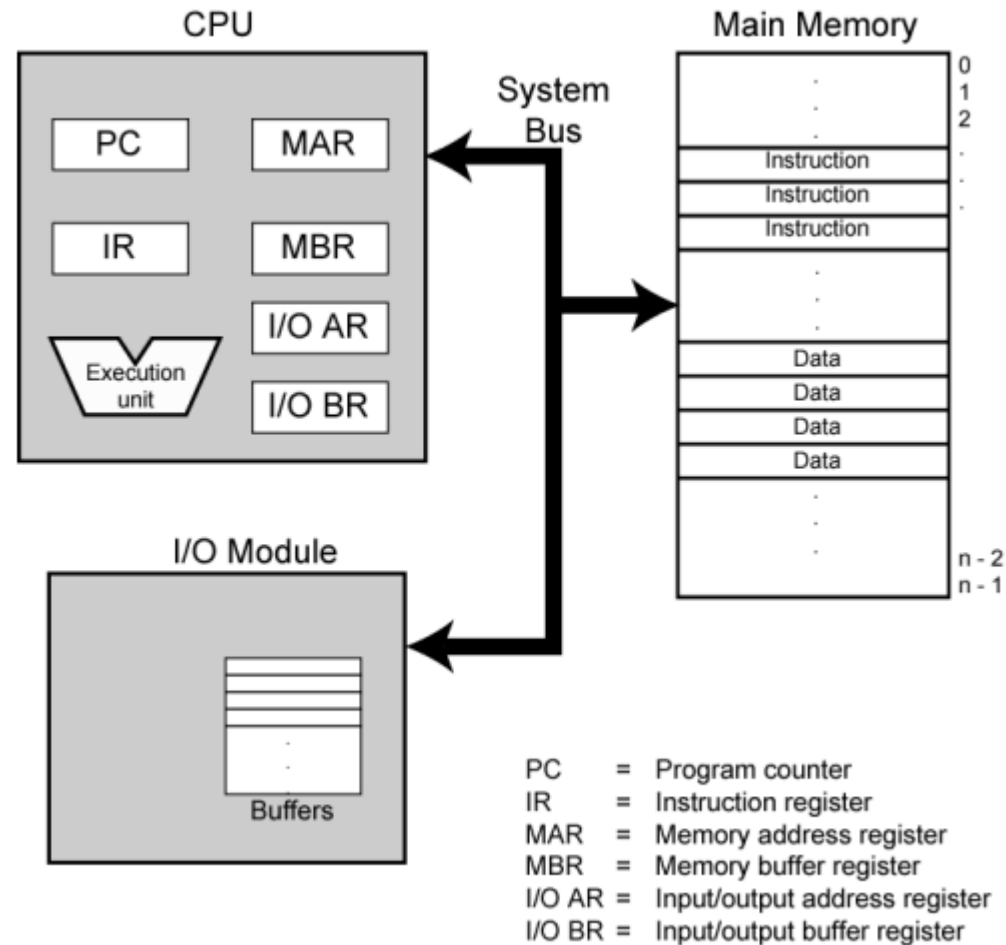
# HARDWARE AND SOFTWARE APPROACHES

- Programming in hardware means configuring a set of basic logic components specifically for a particular computation.
- Programming in software means supplying a specific set of control signals to a general purpose hardware.



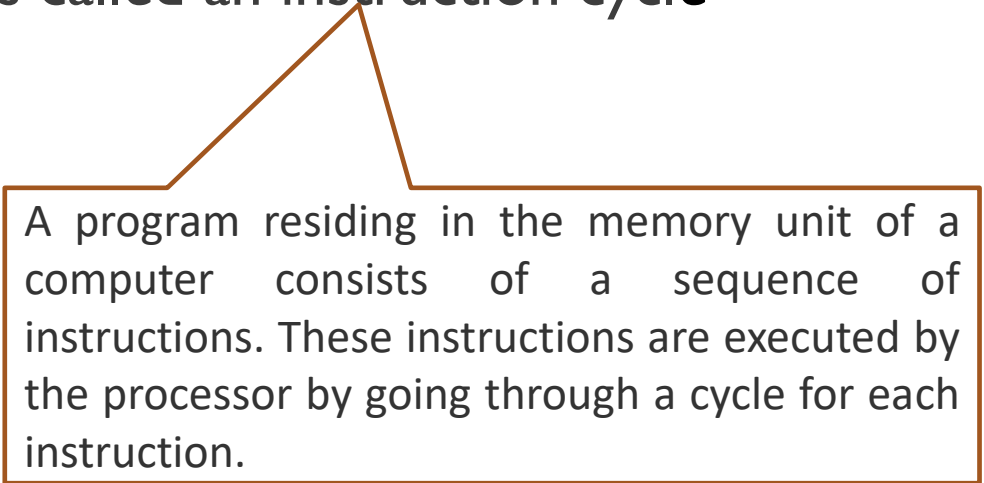


# COMPUTER COMPONENTS: TOP LEVEL VIEW



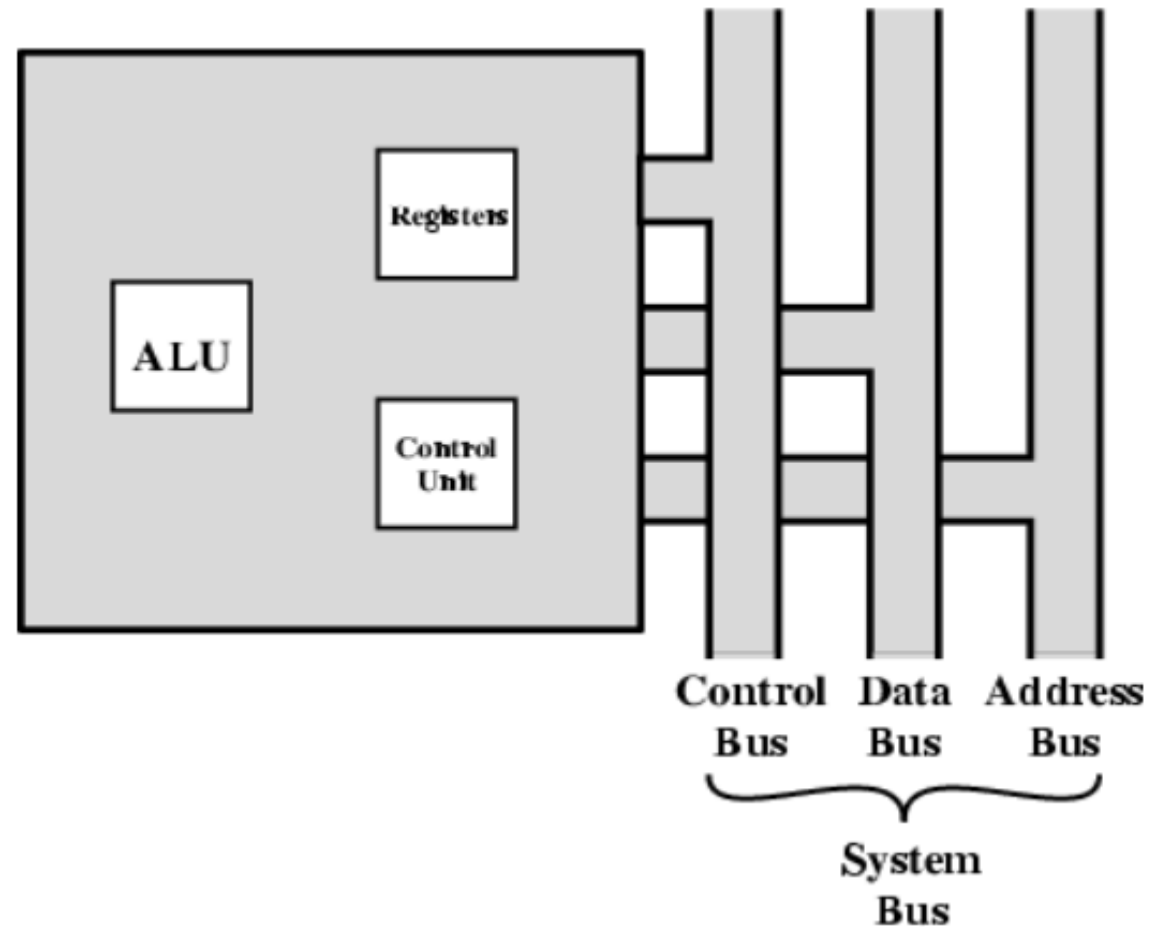
# COMPUTER FUNCTIONS

- The basic function of a computer is to execute a program, which consists of a set of instructions stored in memory
- The instruction is in the form of a binary code that specifies what action the CPU is to take
- Processing required for a single instruction is called an instruction cycle

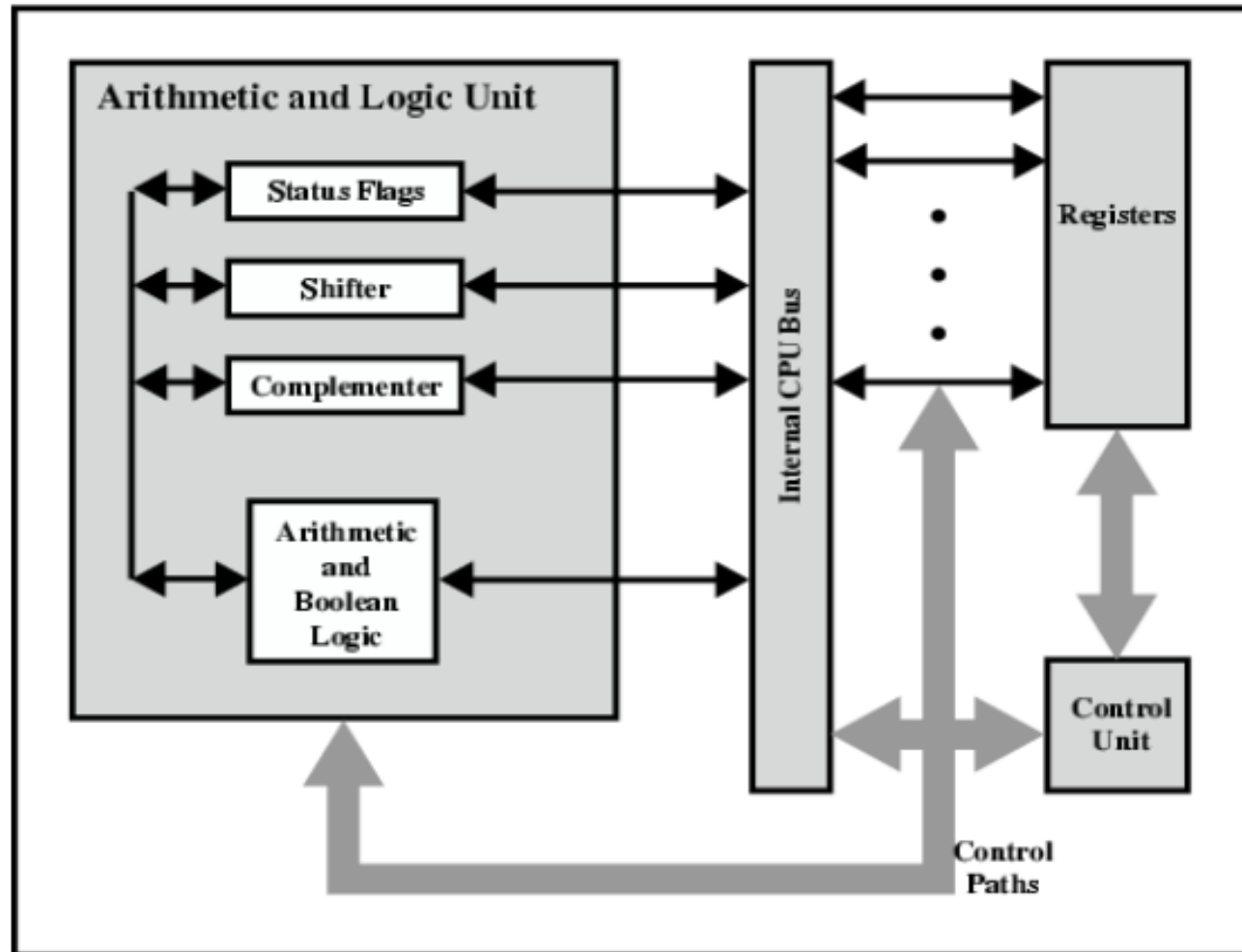


A program residing in the memory unit of a computer consists of a sequence of instructions. These instructions are executed by the processor by going through a cycle for each instruction.

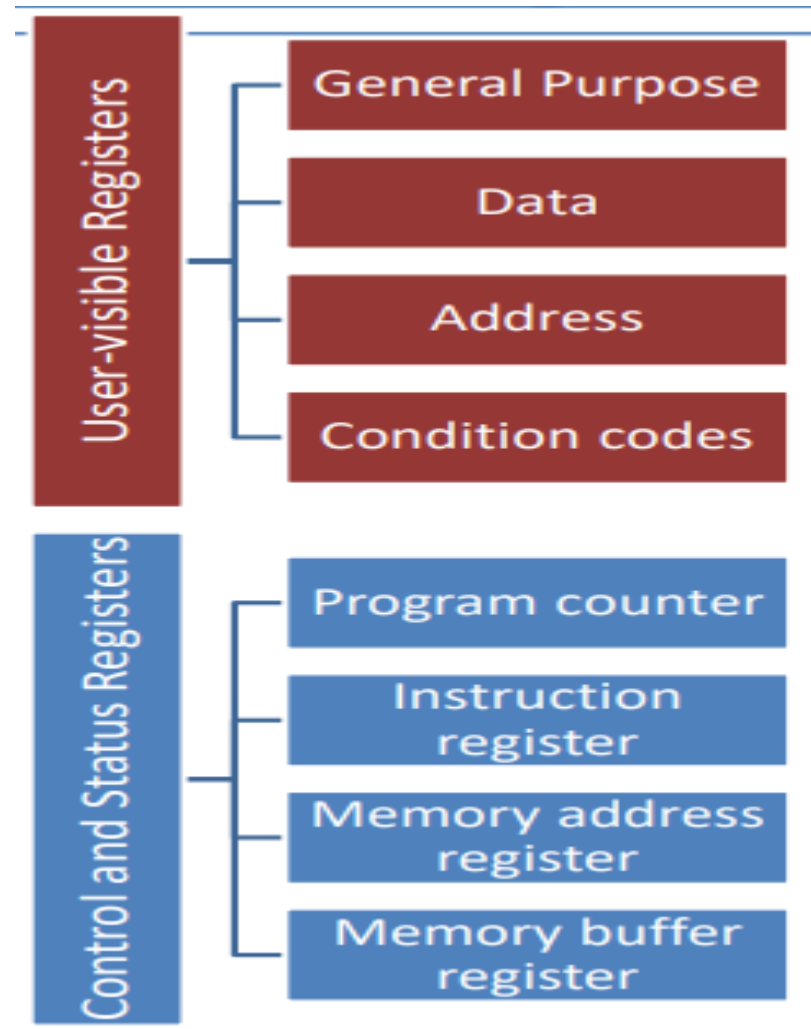
# CPU WITH SYSTEM BUS



# CPU INTERNAL STRUCTURE



# REGISTER ORGANIZATION



# REGISTER ORGANIZATION

- 1. User Visible Registers
  - Can be accessed via assembly instructions
  - Optimizing the use of registers can minimize the main memory references
- 2. Control and Status Registers
  - Control unit uses to control the operations of the processor
  - Also some privileged operating system programs use to control the execution of programs

# I. USER VISIBLE REGISTERS

- General Purpose:
  - can be assigned to a variety of functions by the programmer.
- Data registers:
  - may be used only to hold data and cannot be employed in the calculation of an operand address.

# I.USER VISIBLE REGISTERS

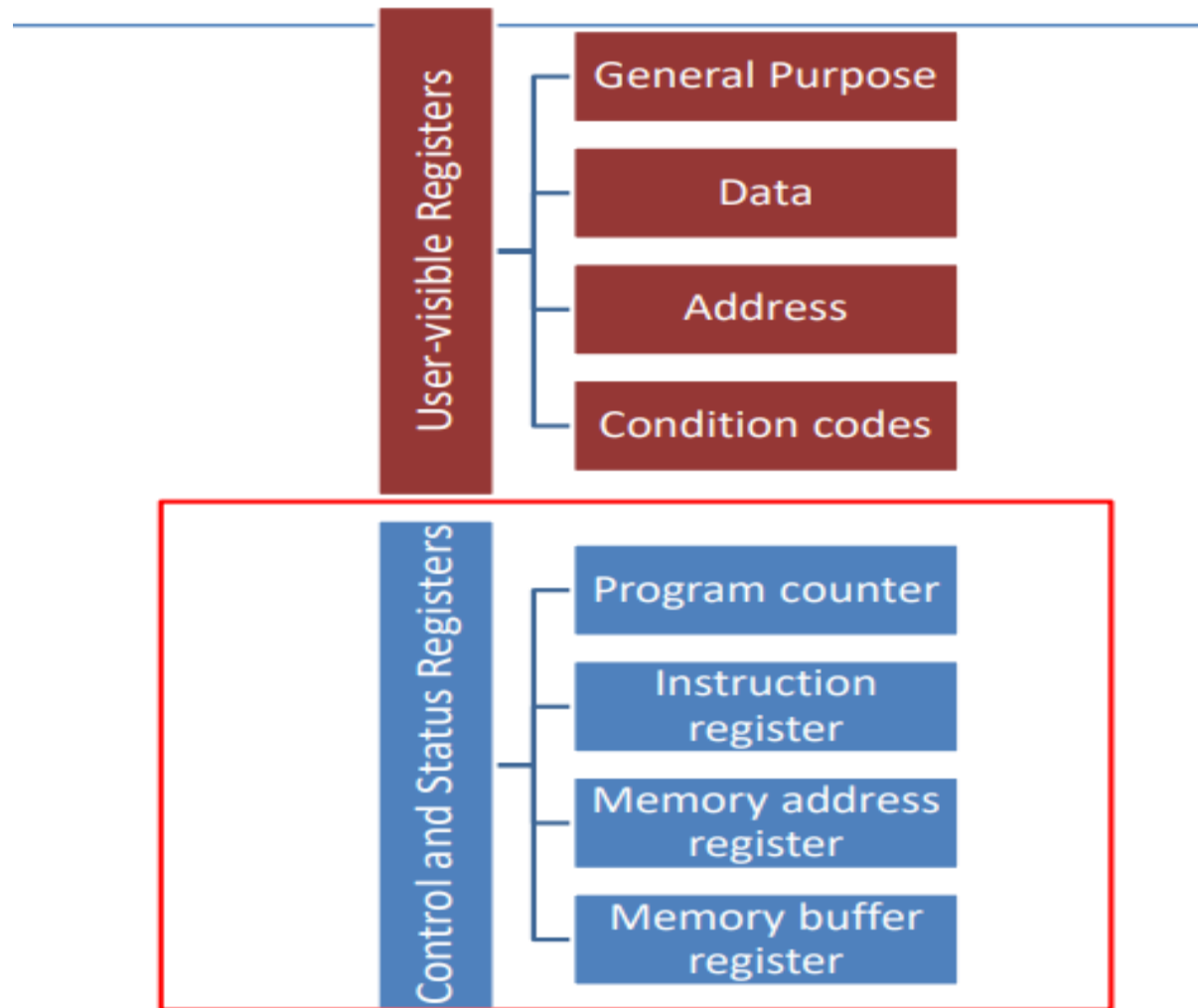
- Address registers:
  - somewhat general purpose, or they may be devoted to a particular addressing mode.
  - Segment pointers: In a machine with segmented addressing, a segment register holds the address of the base of the segment.
  - Index registers: These are used for indexed addressing and may be auto indexed.
  - Stack pointer: A dedicated register that points to the top of the stack.



# I.USER VISIBLE REGISTERS

- Condition codes / flags:
    - Partially visible to the user, holds condition codes.
    - Condition codes are bits set by the processor hardware as the result of operations.
- For example, an arithmetic operation

# REGISTER ORGANIZATION



## 2.CONTROL AND STATUS REGISTERS

### 1.Program counter (PC)

- Contains the address of an instruction to be fetched

### 2. Instruction register (IR)

- Contains the instruction most recently fetched

### 3. Memory address register (MAR)

- Contains the address of a location in memory

### 4. Memory buffer register (MBR)

- Contains a word of data to be written to memory or the word most recently read

# TYPES OF DATA TRANSFERS

## 1. Memory to processor:

- The processor reads an instruction or a unit data from memory

## 2. Processor to memory:

- The processor writes a unit of data to memory

## 3. I/O to processor:

- The processor reads data from an I/O device via an I/O module

## 4. Processor to I/O:

- The processor sends data to the I/O device

## 5. I/O to or from memory:

- For these two cases, an I/O module is allowed to exchange data directly with memory, without going through the processor, using direct memory access (DMA)

# BUS INTERCONNECTION

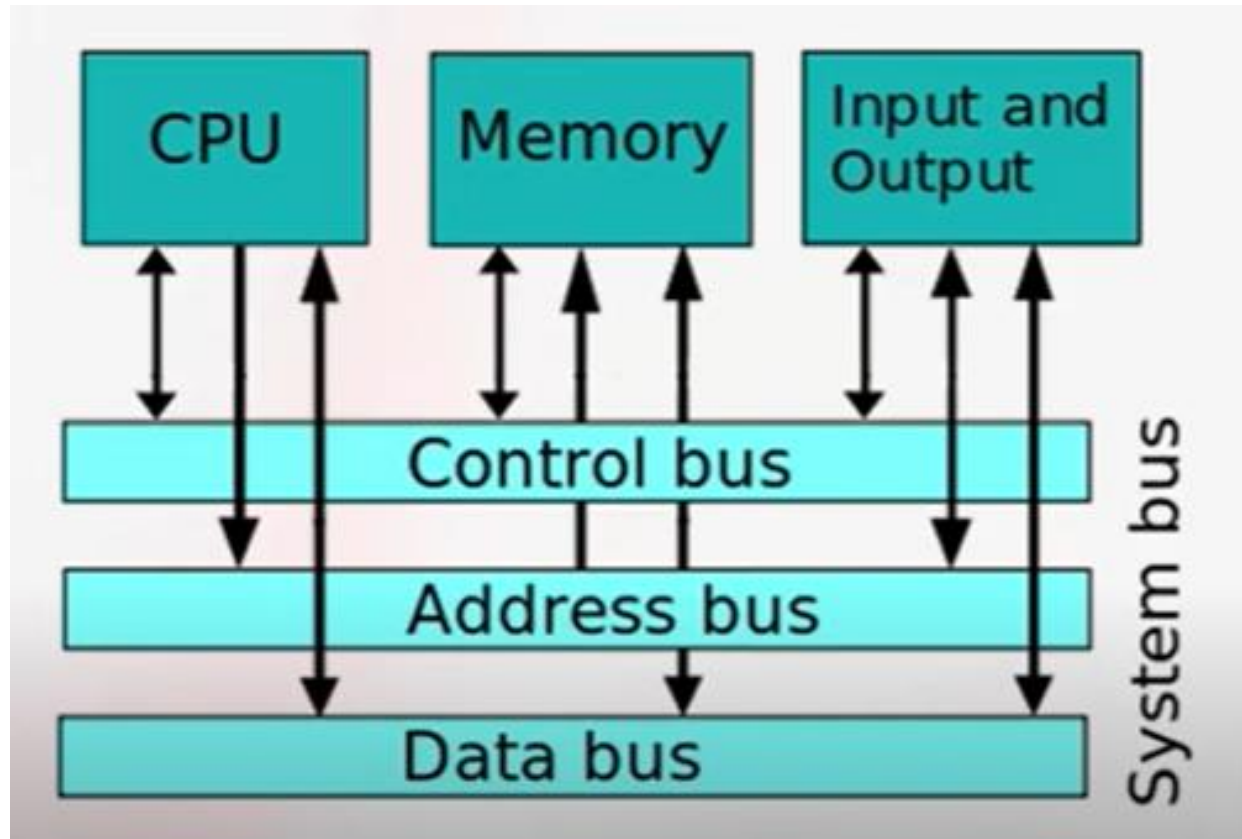
There are a number of possible interconnection systems

- Single and multiple BUS structures are most common
  - – e.g. Control/Address/Data bus (PC)
  - – e.g. Unibus (DEC-PDP)
- A key characteristic of a bus is that it is a shared transmission medium
- A bus consists of multiple communication pathways, or lines. Each line is capable of transmitting signals representing binary 1 and binary 0

# WHAT IS A BUS?

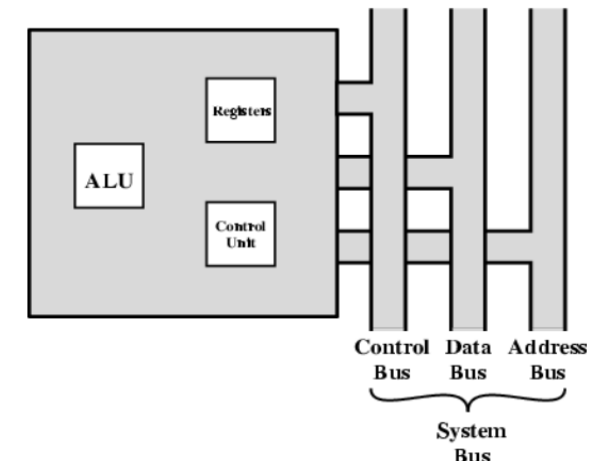
- Bus is a communication pathway connecting two or more devices
- Features:
  - – Shared transmission medium
  - – Multiple devices connected to the bus
  - – Signal transmitted by any device is available for reception by all other devices attached to the bus
  - – Only one device at a time can successfully transmit
  - – Signals will overlap if two devices transmit

# WHAT IS A BUS?



# SYSTEM BUS

- A bus connecting major components such as processor, memory and I/O is called a system bus
  - – Most interconnection structures use one or more system buses
  - – A system bus consists of about 50 to hundreds of separate lines, and each line is assigned a particular meaning or function
  - – Three functional groups: data, address and control





# DATA BUS

- Paths for moving data among system modules
  - – At this level, no difference between data and instructions
- The number of lines in a data bus is referred as width
  - – 32, 64, 128 or more separate lines
  - – Number of lines determines how many bits can be transferred at a time as each line can carry only one bit at a time
  - – The width of the data bus is a key factor in determining overall system performance
- Example:
  - • Data bus is 32 bits wide
  - • Each instruction is 64 bits long
  - • Processor must access the memory module twice during each instruction cycle

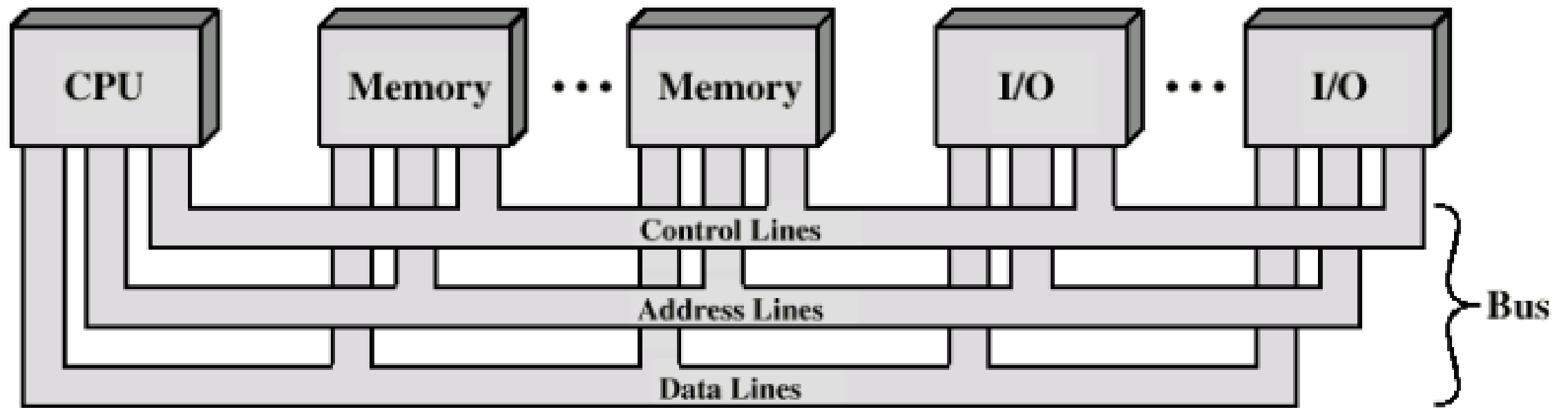
# ADDRESS BUS

- Used for identifying the source or destination of the data on the data bus
- Example:
  - Processor wants to read a word 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
  - Eg. 8080 has 16bit address bus giving 64K address space

# CONTROL BUS

- Used to control the access to and the use of the data and address lines
- Control signals transmit both command and timing information
  - – Timing Signals : Validity of data and address information
  - – Command Signals : Specify the operations to be performed
- Control and Timing information:
  - 1. Memory read/write signal
  - 2. Interrupt request/ACK
  - 3. Bus request/grant
  - 4. Clock signals

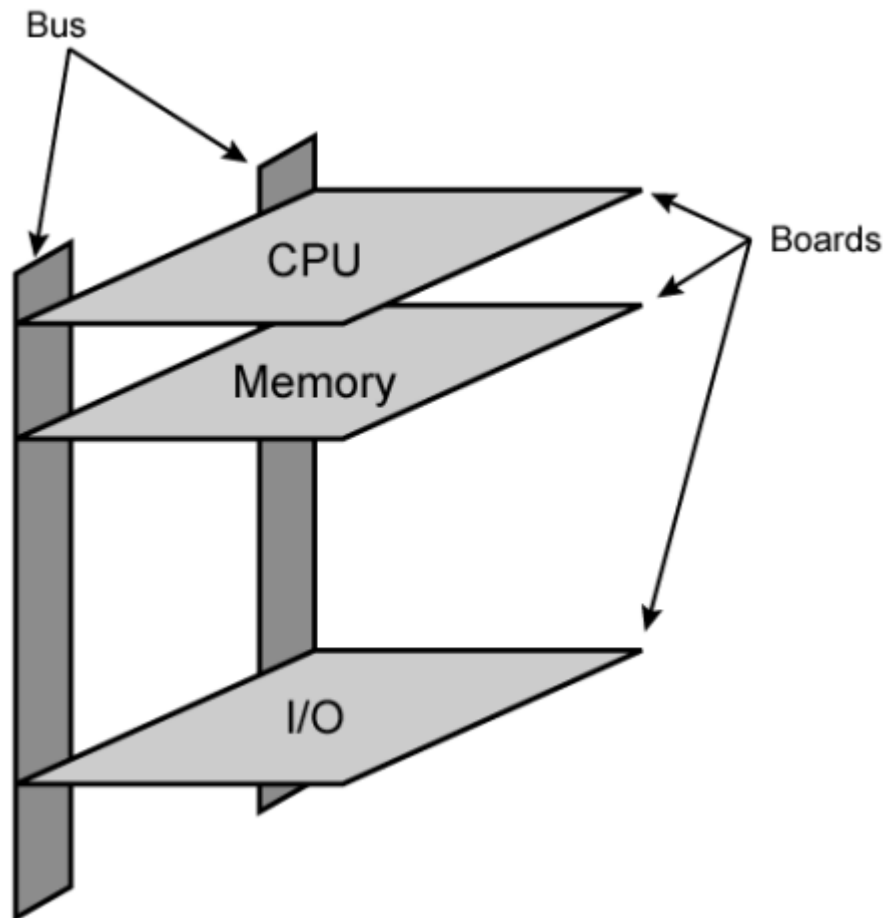
# BUS INTERCONNECTION SCHEME



# BUS OPERATION

- Module that sends data to another:
  - 1. Obtain the use of the bus
  - 2. Transfer data via the bus
- • Module requesting data:
  - 1. Obtain the use of the bus
  - 2. Transfer a request to the other module over the appropriate control and address lines. Module must wait for that second module to send the data

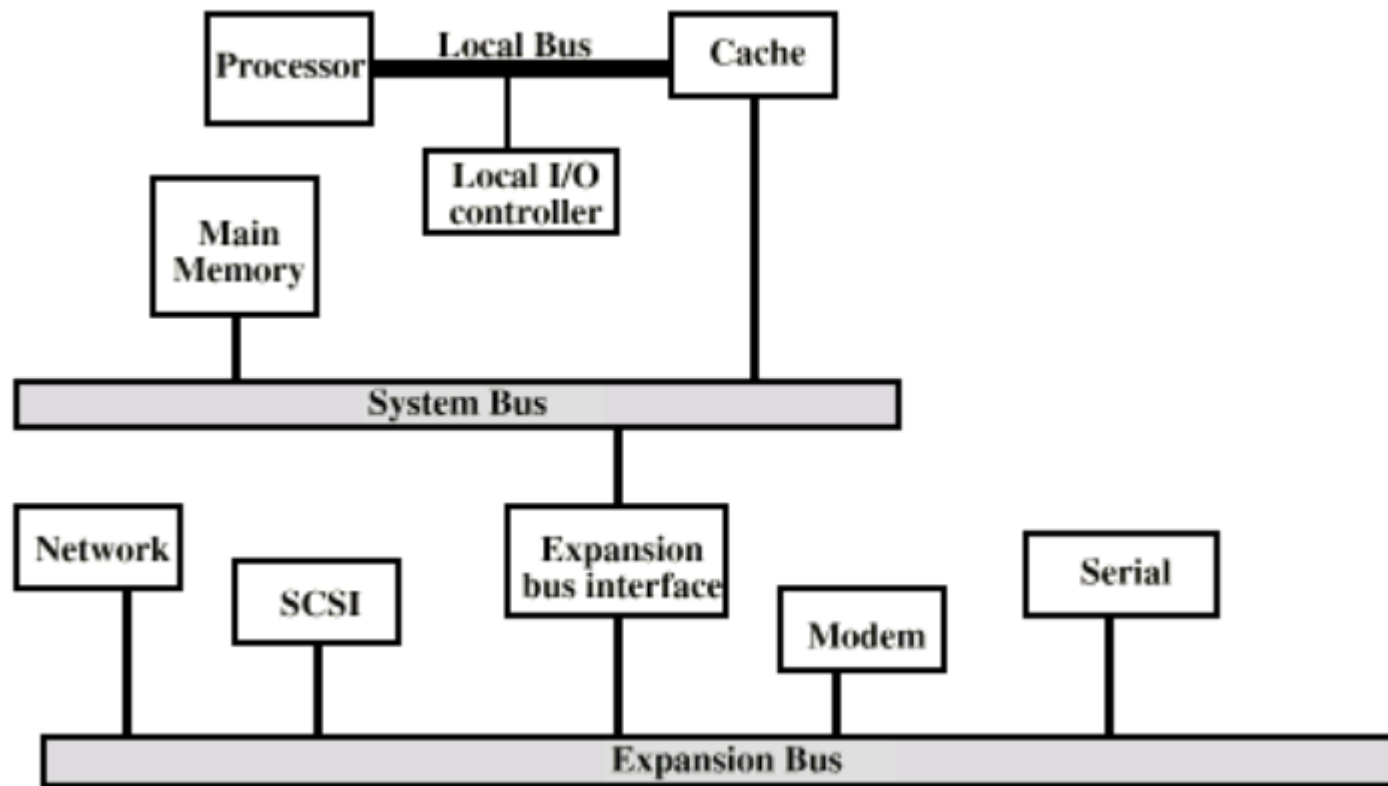
# PHYSICAL REALIZATION OF BUS ARCHITECTURE



# SINGLE BUS PROBLEMS

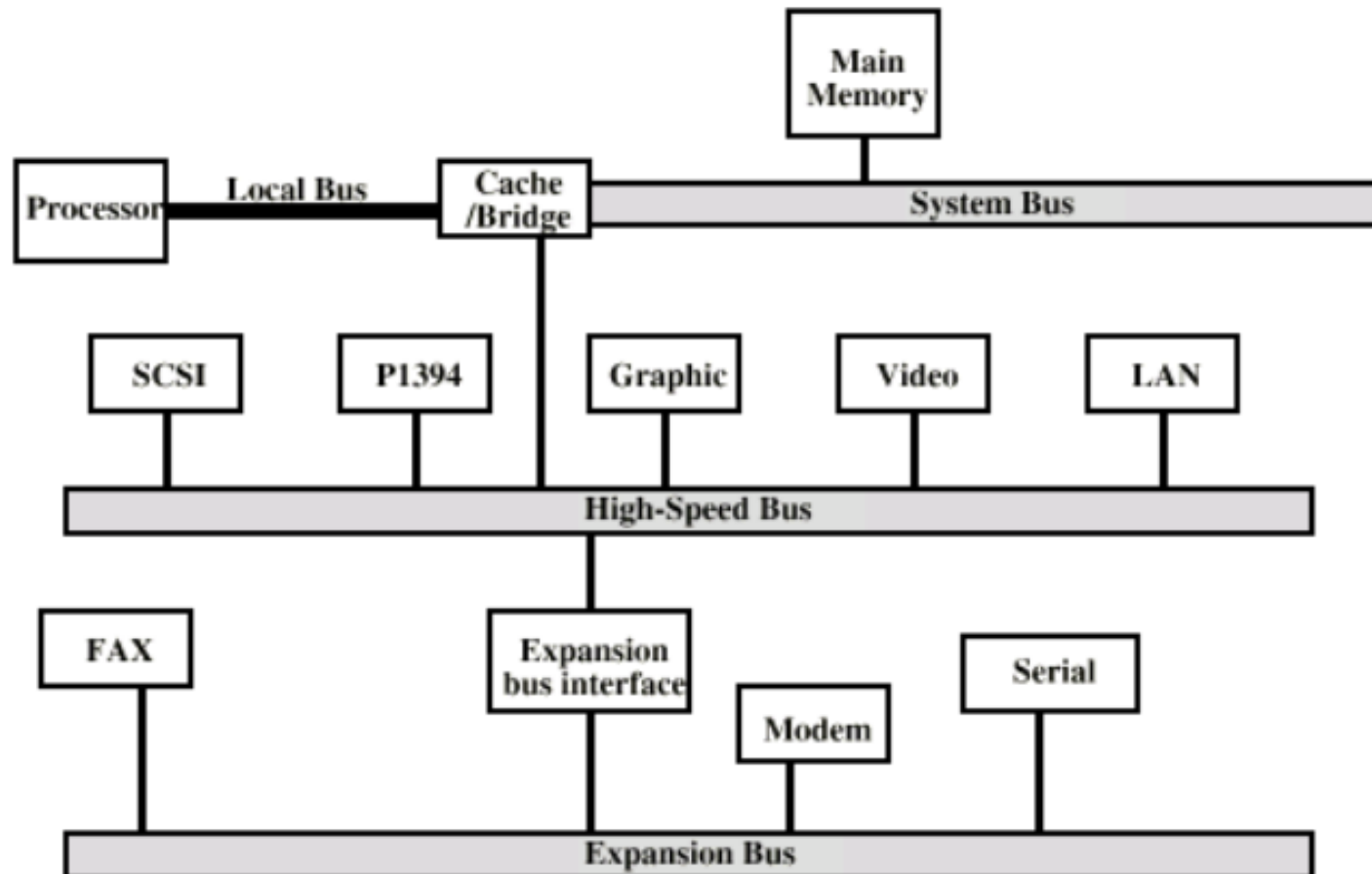
- Lots of devices on one bus leads to:
  - Propagation delays
    - Long data paths mean that co-ordination of bus use can adversely affect performance
    - If aggregate data transfer approaches bus capacity
- Most systems use multiple buses to overcome these problems

# TRADITIONAL BUS ARCHITECTURE





# HIGH PERFORMANCE BUS



# ELEMENTS OF A BUS DESIGN

## Reference;

- Type
  - 1.Dedicated
  - 2.Multiplexed
- Method of Arbitration
  - 1.Centralized
  - 2.Distributed
- Timing
  - 1.Synchronous
  - 2.Asynchronous
- Bus Width
  - 1.Address
  - 2.Data
- Data transfer Type
  - Read
  - Write

# BUS TYPES

- Dedicated
  - Separate data & address lines
- Multiplexed
  - Shared lines
  - Address valid or data valid control line
  - Advantage
    - - fewer lines
  - Disadvantages
    - More complex control
    - Ultimate performance

# BUS ARBITRATION

- A device that initiates data transfers on the bus at any given time is called a bus master.
- (In all but the simplest systems, more than one module can require control of the bus.)
- Therefore only one unit at a time can strongly transfer over the bus, some method of arbitration is required.
- Bus arbitration is a process by which next device becomes the bus controller by transferring bus mastership to another bus.
  - e.g. CPU and DMA controller
- Only one module may control bus at one time
- Arbitration may be centralized or distributed

# CENTRALIZED OR DISTRIBUTED ARBITRATION

- Centralized
  - Only single bus arbiter performs the required arbitration and it can be either a processor or a separate DMS controller.
    - Bus Controller
    - Arbiter
  - May be part of CPU or separate

## Reference:

three arbitration schemes which run on centralized arbitration.

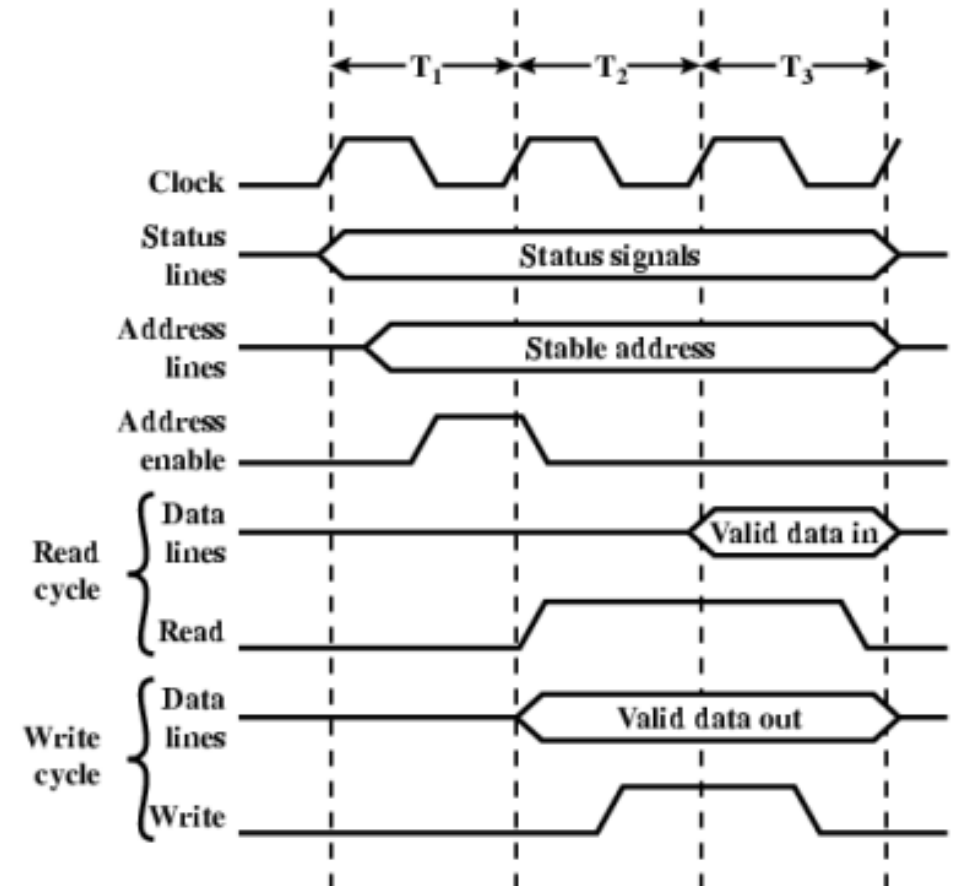
1. Daisy chaining
2. Polling method
3. Independent request

# CENTRALIZED OR DISTRIBUTED ARBITRATION

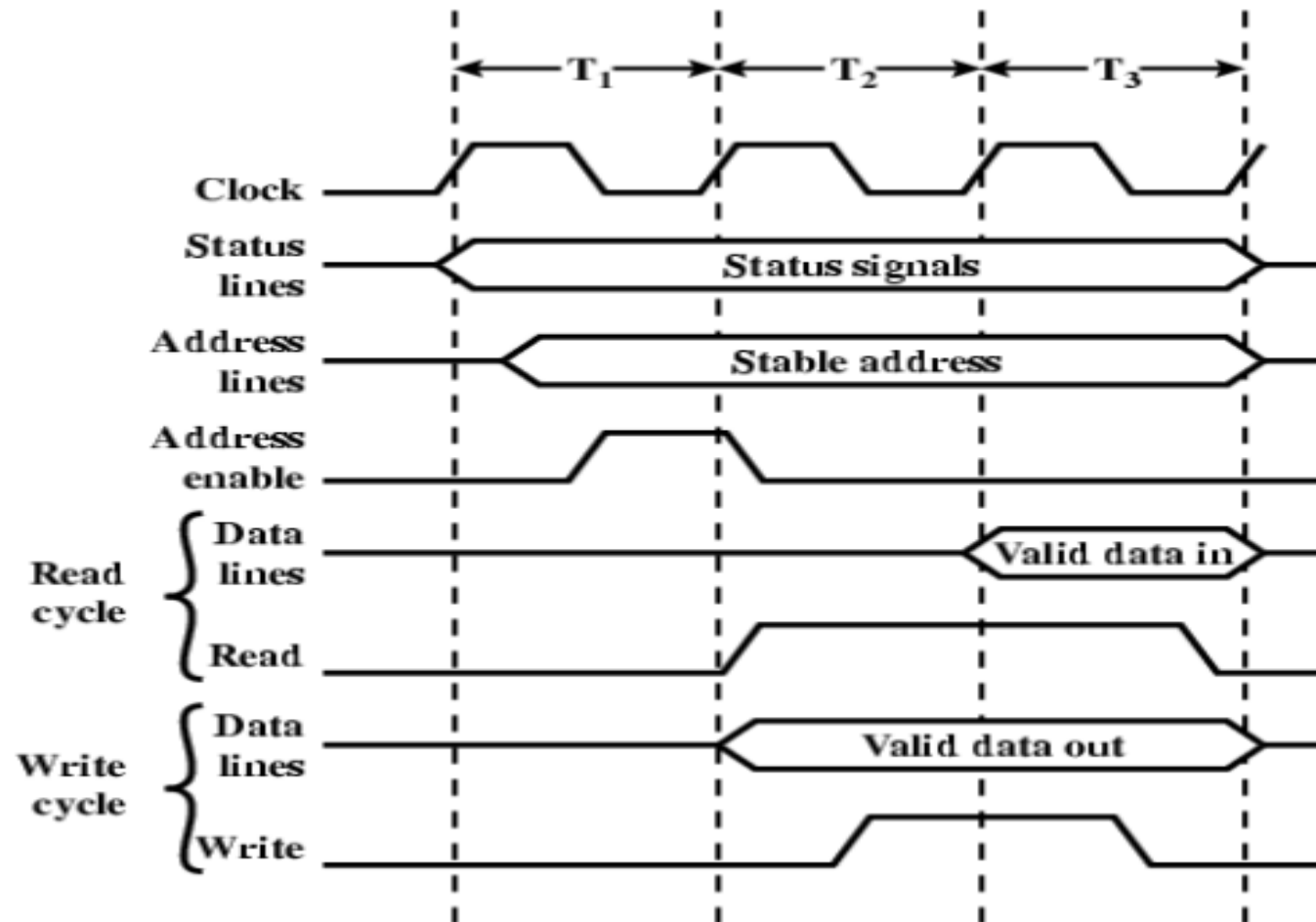
- Distributed
  - Here, all the devices participate in the selection of the next bus master.
  - Control logic on all modules

# TIMING

- Co-ordination of events on bus
- Synchronous
  - Events determined by clock signals
  - Control Bus includes clock line
  - A single I-O is a bus cycle
  - All devices can read clock line
  - Usually sync on leading edge
  - Usually a single cycle for an event

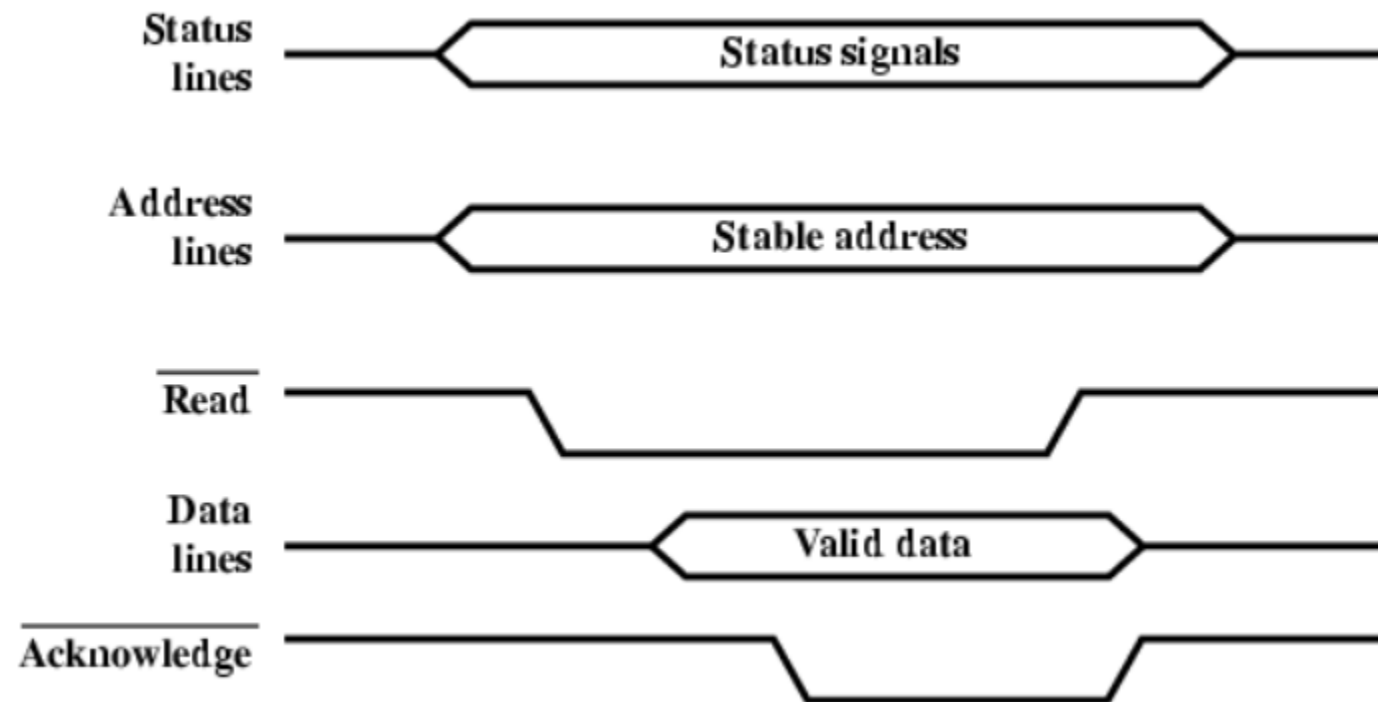


# SYNCHRONOUS TIMING DIAGRAM

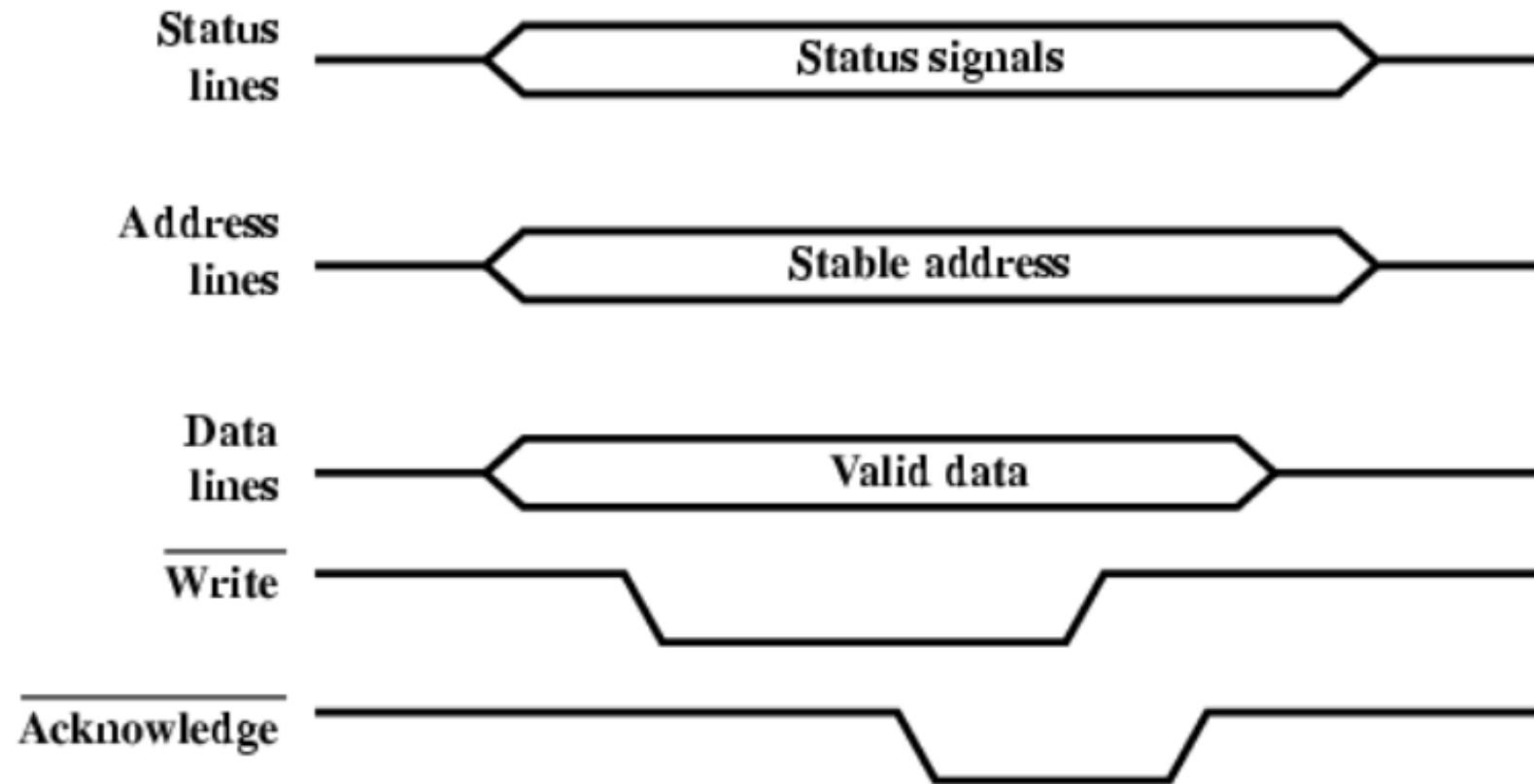




# ASYNCHRONOUS TIMING DIAGRAM- READ DIAGRAM



# ASYNCHRONOUS TIMING DIAGRAM- WRITE DIAGRAM



- 
- Questions?