

CSE 321a

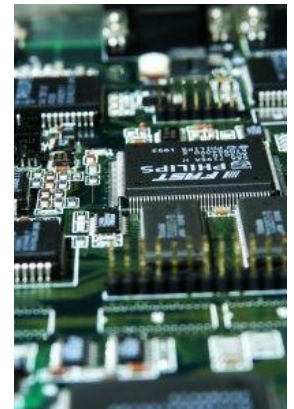
Computer Organization (1)

تنظيم الحاسبات (1)



3rd year, Computer Engineering
Fall 2016

Lecture #3



Dr. Hazem Ibrahim Shehata

Dept. of Computer & Systems Engineering

Credits to Dr. Ahmed Abdul-Monem Ahmed for the slides

Administrivia

- Lecture:
 - New lecture day/time: **Wednesday 10:15am-12:45pm**
- Tutorials:
 - Day/Time: **Sunday 2:00pm – 3:30pm**
- Assignment #1:
 - Released today.
 - Due on: **Wednesday October 19, 2016.**
 - Work in groups of two.
- Academic integrity:
 - Copying assignments will not be tolerated.
 - Involved groups get 0 points!!

Website: <http://hshehata.github.io/courses/zu/cse321a>

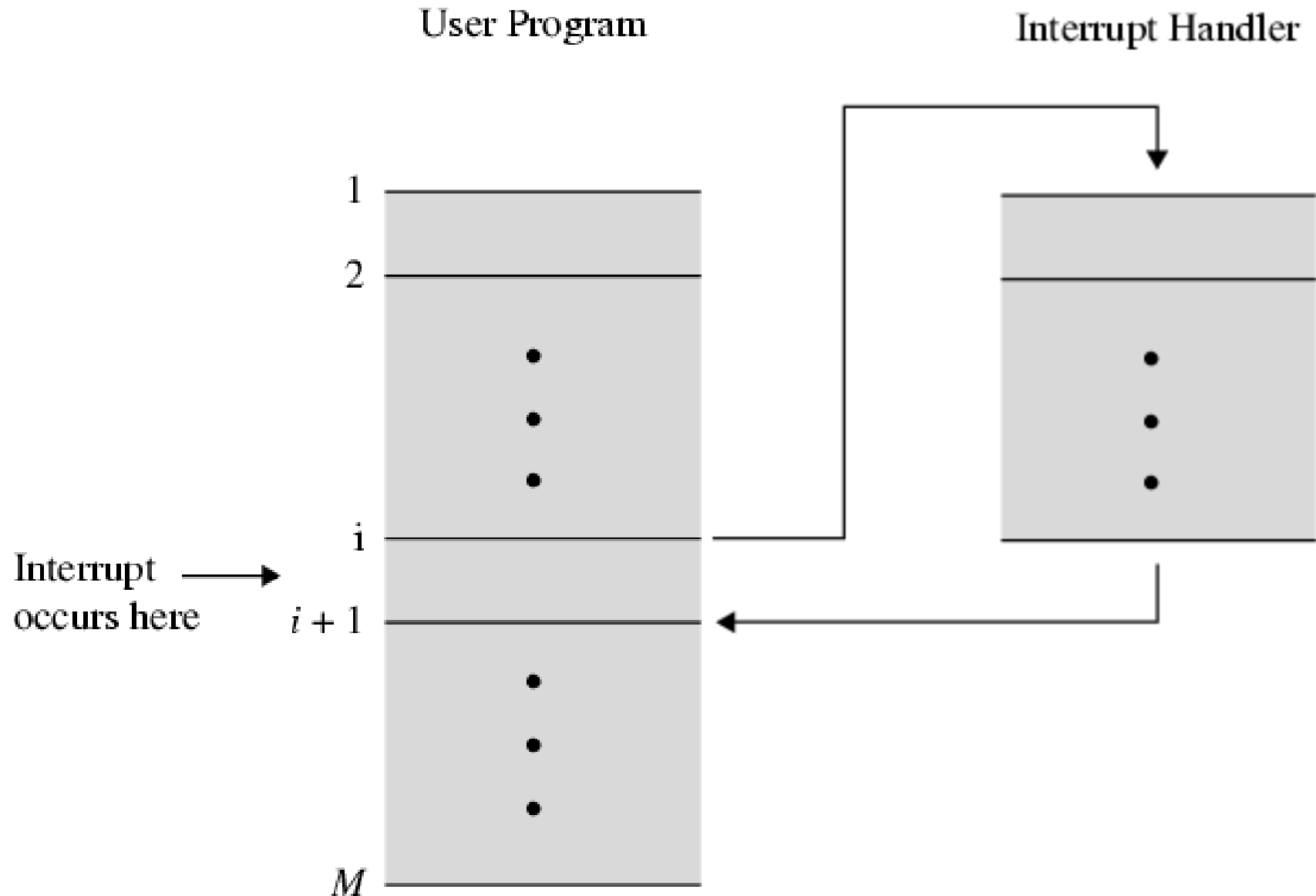
Office hours: Sunday 12:00pm-1:00pm

Chapter 3. A Top-Level View of Computer Function and Interconnection (Cont.)

Overview

- Computer Structure (Components):
 - CPU, Memory, I/O.
- Computer Function:
 - Instruction Fetch/Execute.
 - Interrupts.
- Interconnection Structure:
 - Transfer between mem. & proc., i/o & proc. ... etc.
- Bus Interconnection:
 - Bus structure: address, data, and control lines.
 - Multiple-bus hierarchies.
 - Elements of bus design.

Transfer of Control via Interrupts



Multiple Interrupts

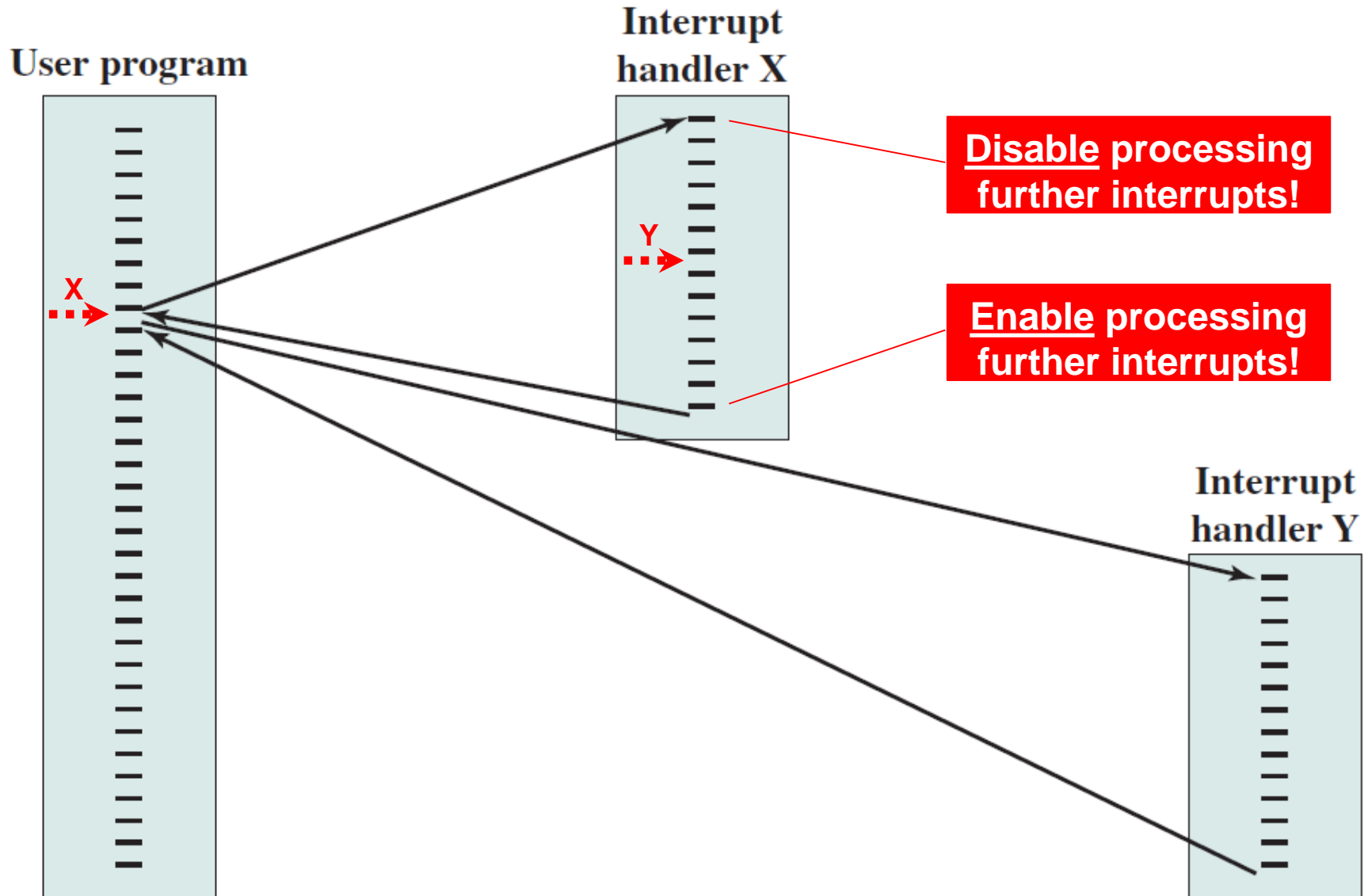
- Disable interrupts

- Processor will ignore further interrupts while processing one interrupt.
- Interrupts remain pending and are checked after first interrupt has been processed.
- Interrupts handled in **sequence** as they occur.

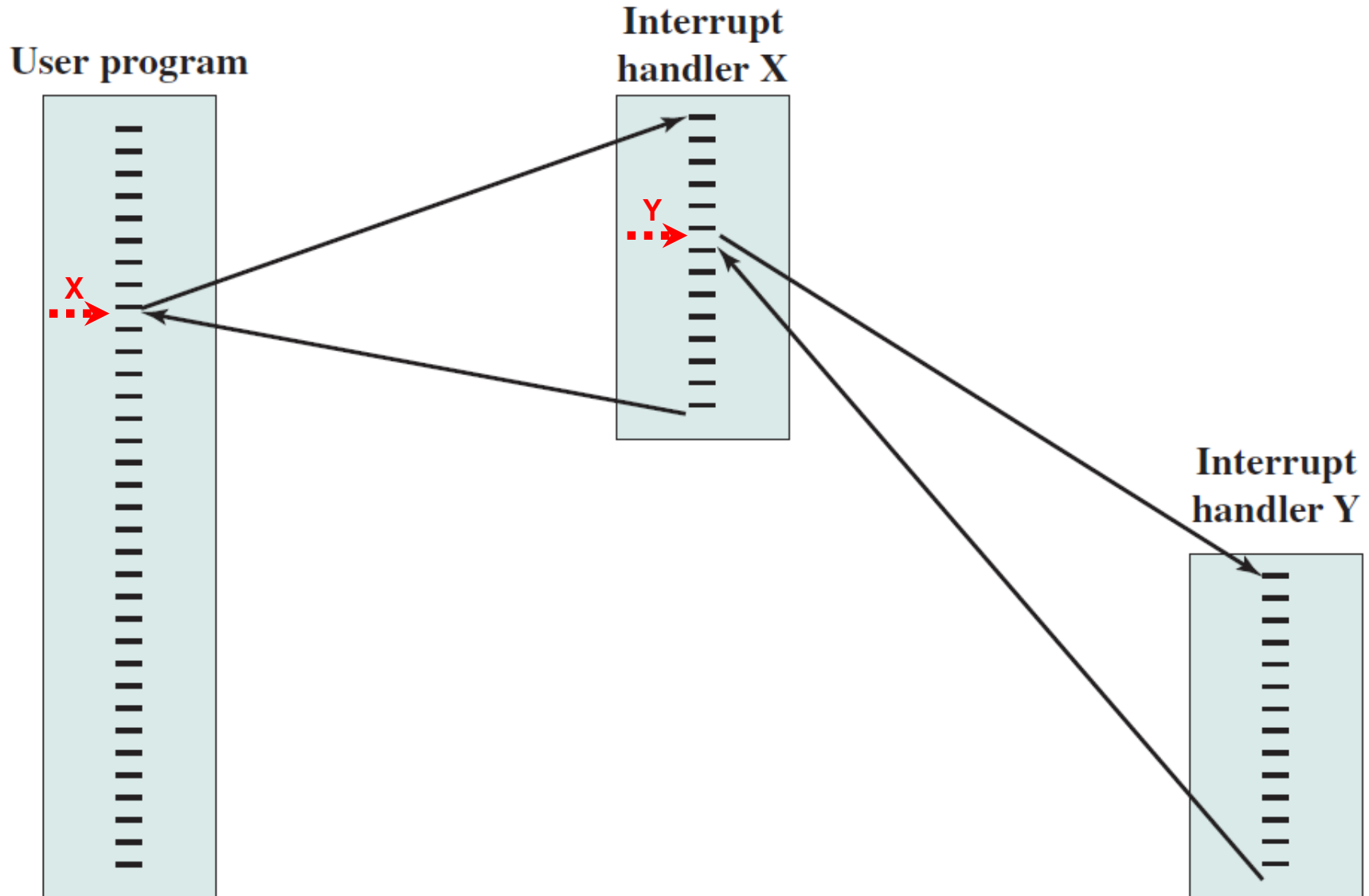
- Define priorities

- Low priority interrupts can be interrupted by higher priority interrupts.
- When higher priority interrupt has been processed, processor returns to previous interrupt.

Multiple Interrupts - Sequential



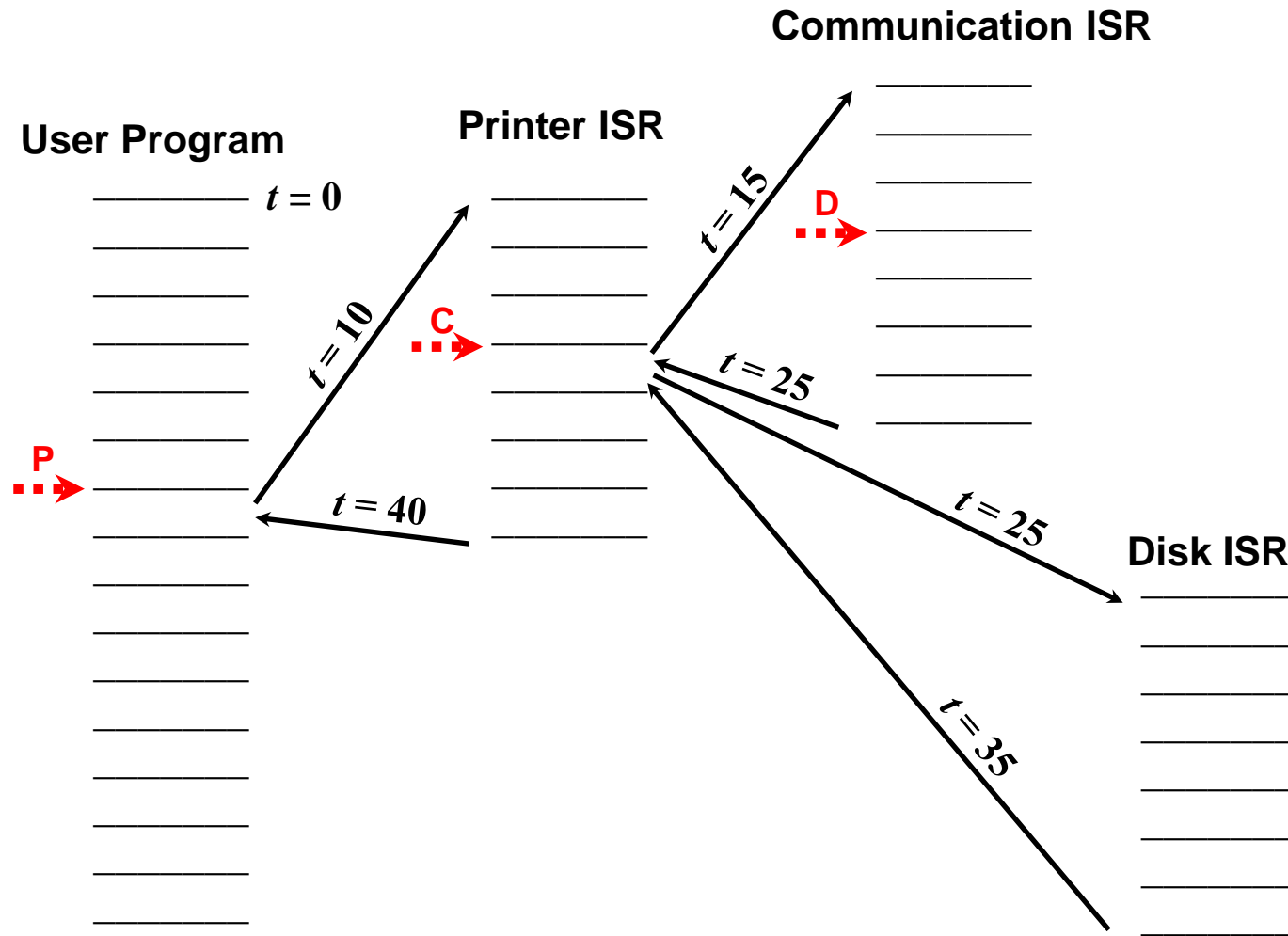
Multiple Interrupts – Nested



Multiple Interrupts – Nested (example)

Interrupt priorities: Printer (P): 2, Disk (D): 4, Communication line (C): 5

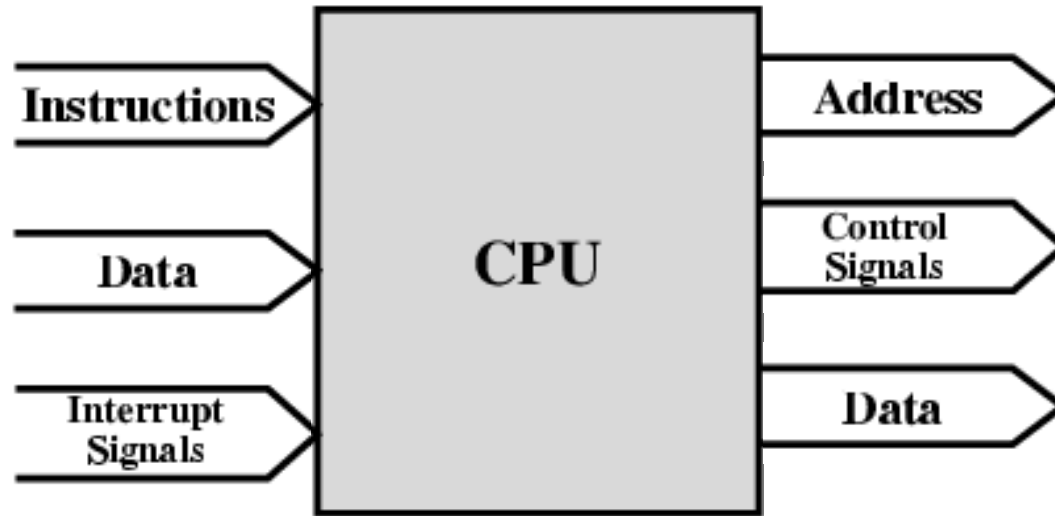
ISR execution: 10 units, Interrupt scenario: P @ $t=10$, C @ $t=15$, D @ $t=20$



Interconnection Structures

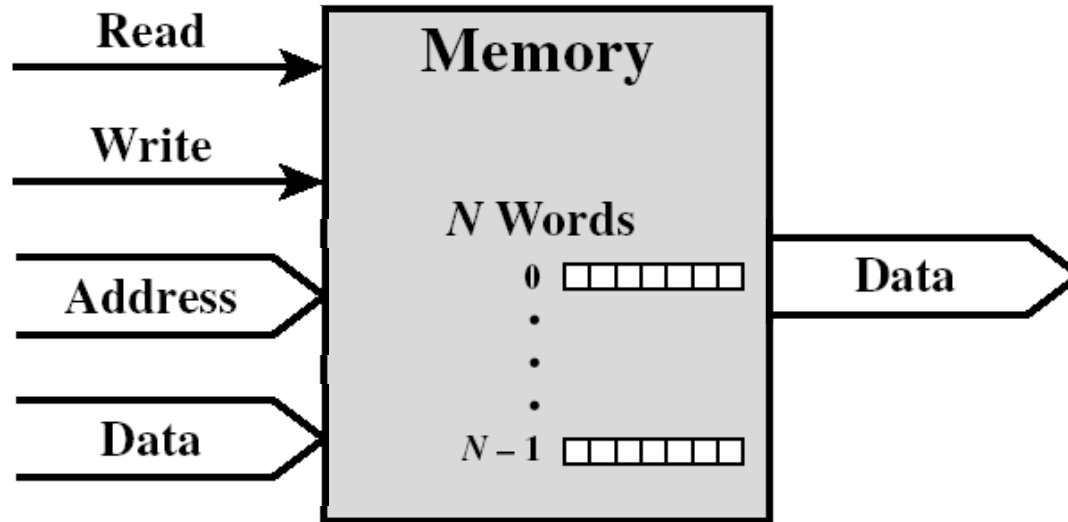
- All computer components/modules must be connected.
- Collection of paths connecting various modules is called: Interconnection Structures
- Different type of connection is required for different type of module.
 - CPU
 - Memory
 - Input/Output

CPU Connection



- Reads instructions and data
- Writes out data (after processing)
- Sends control signals to other units
- Receives (& acts on) interrupts

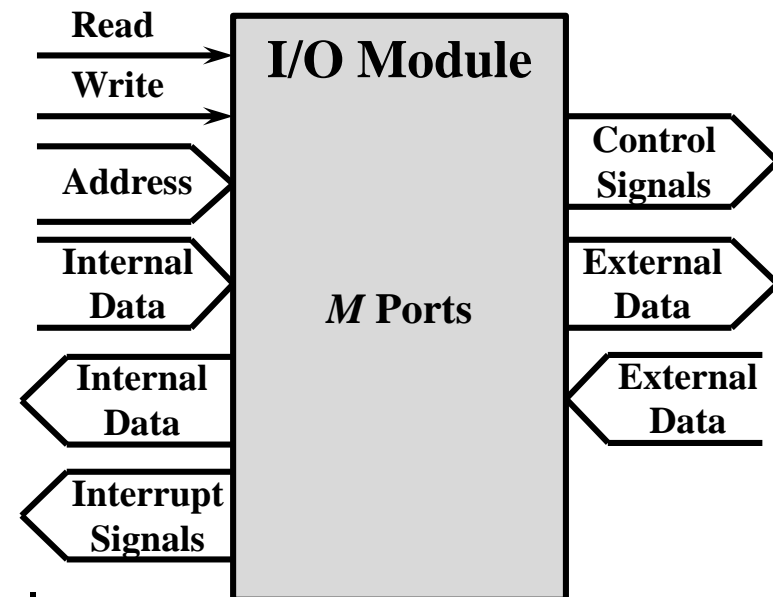
Memory Connection



- Receives control signals.
 - Read
 - Write
- Receives addresses (of locations).
- Sends data (read operation).
- Receives data (write operation).

I/O Connection

- Similar to memory from computer's viewpoint.
- Receives control signals from computer.
 - e.g., read and write.
- Receives addresses from computer
 - e.g., port no. to identify peripheral.
- Output
 - Receives data from computer
 - Sends data to peripheral
- Input
 - Receives data from peripheral
 - Sends data to computer
- Sends control signals to peripherals.
 - I/o commands (e.g., spin disk).
- Sends interrupt signals to computer

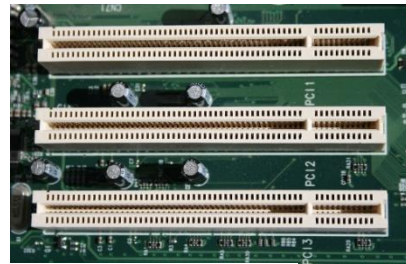


Types of Transfers

- CPU to memory.
 - Memory to CPU.
 - CPU to I/O.
 - I/O to CPU.
 - I/O to memory.
 - Memory to I/O.
- } **Direct Memory Access (DMA)**

Bus Interconnection

- There are a number of possible interconnection systems.
- Single and multiple Bus structures are most common.
- Examples of a bus:
 - Internal: PCI, SATA
 - External: USB
 - Internal/external: SCSI



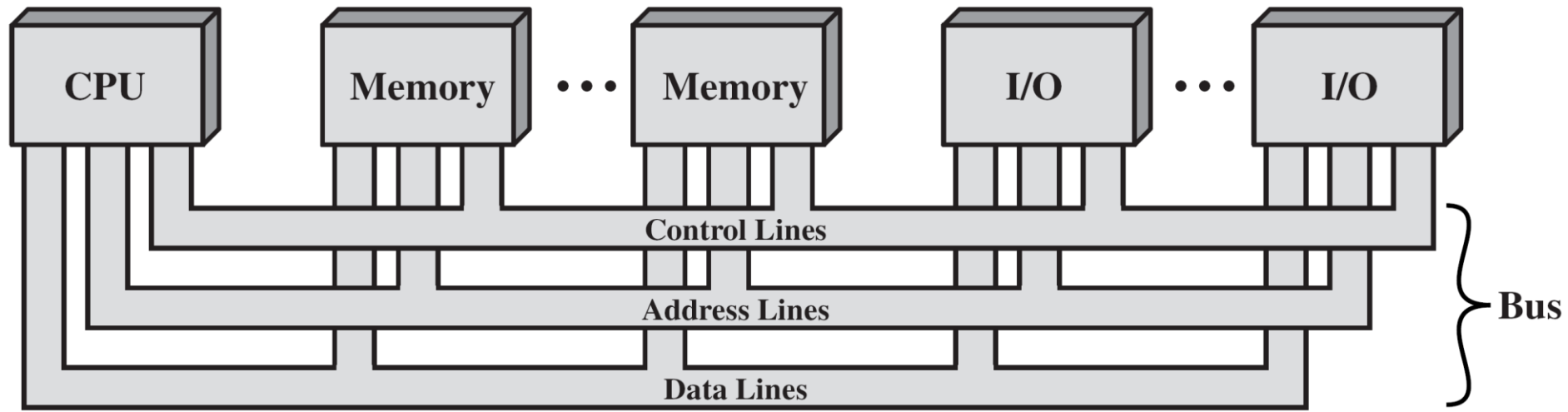
What is a Bus? (1)

- A communication pathway connecting two or more devices.
- Usually by **broadcast**.
- Consists of multiple lines (each carries 1 bit)
 - e.g. 32 lines are needed to transfer 32 bits of data in parallel.
- Bus lines typically:
 - carry address, data, and/or control information
 - supply power (VCC, GND, ... etc.)
- Power lines may not be shown.

What is a Bus? (2)

- Data lines
 - Carries data: Remember that there is no difference between “data” and “instruction” at this level!
 - Width is a key determinant of performance: 8, 16, 32, 64 bit.
- Address lines
 - Identify the source or destination of data.
 - e.g. CPU needs to read an instruction (data) from a given location in memory.
 - Bus width determines maximum memory capacity of system.
 - e.g. 8080 has 16 bit address bus giving 64k address space.
- Control lines
 - Control and timing information: memory read/write signal, interrupt request, clock signals

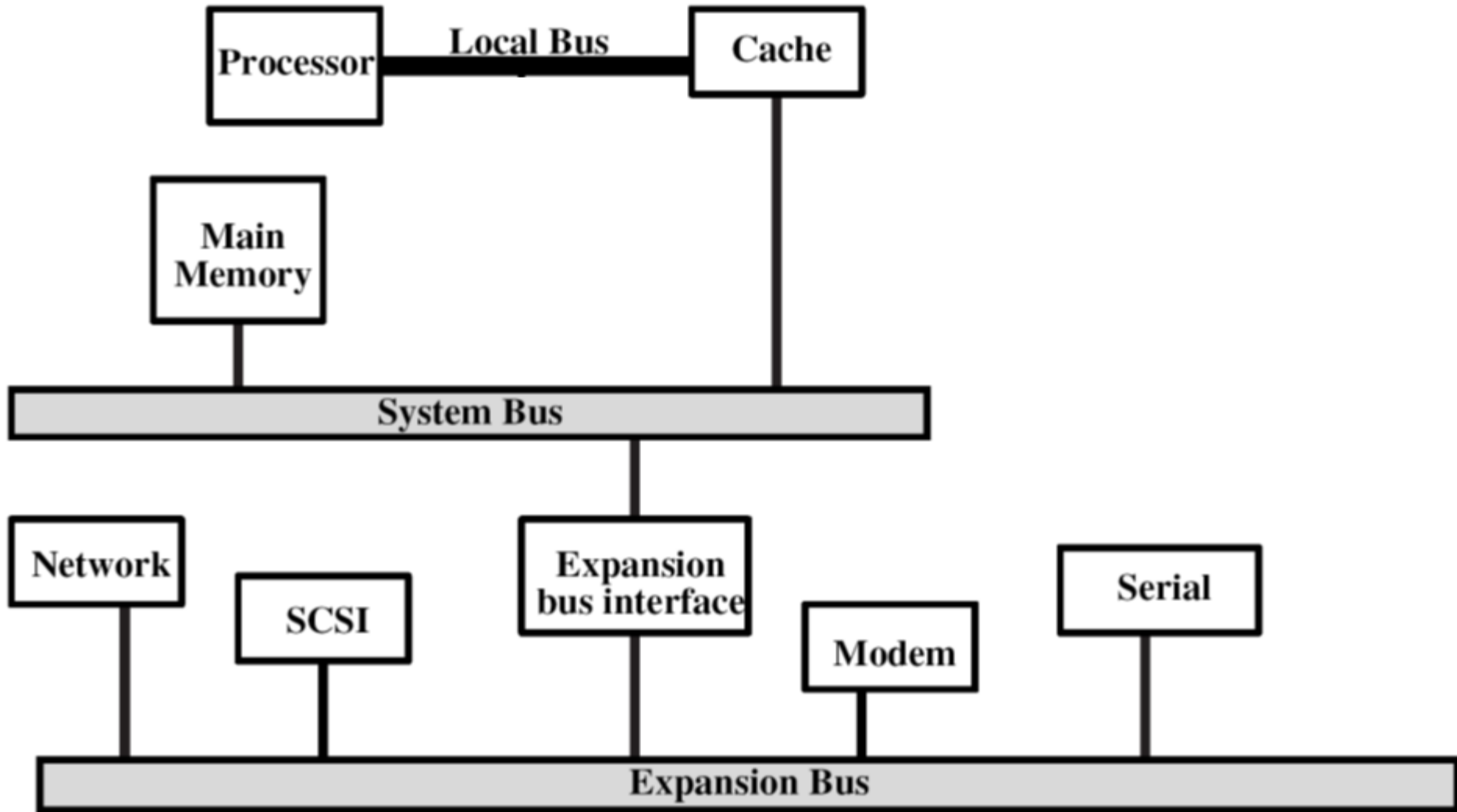
Bus Interconnection Scheme



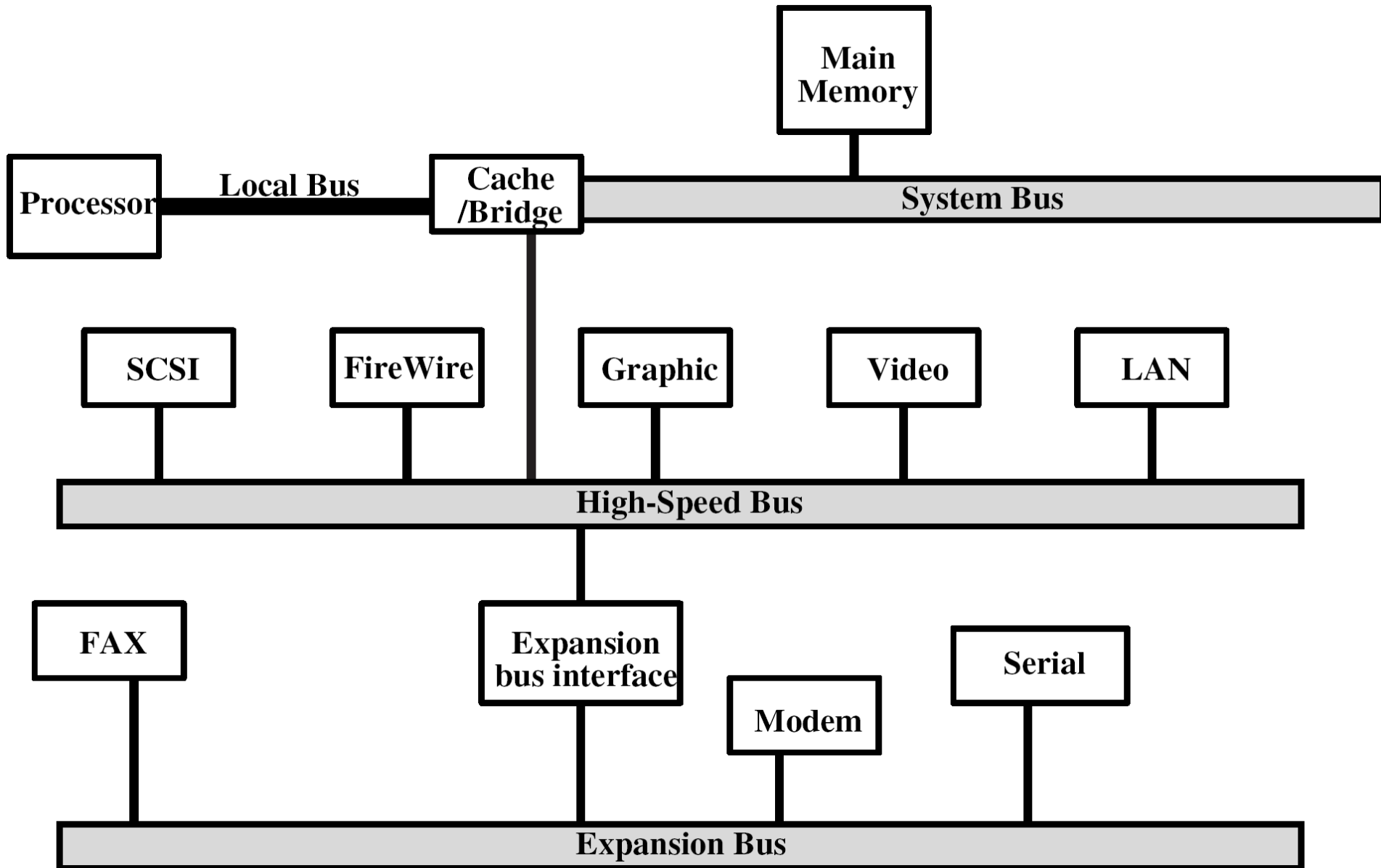
Single Bus Problems

- If great number of devices are connected to bus, performance will suffer. Why?
 - Greater bus length → greater **propagation** delay → more time required for device coordination
 - Aggregate data transfer demand approaches bus capacity → bus **speed** becomes a bottleneck.
- To overcome this problem:
 - Most systems use multiple buses laid out hierarchically.

Traditional Bus Architecture



High Performance Bus



Elements of Bus Design

- Bus Type
- Method of Arbitration
- Timing
- Data Transfer Type

Bus Types

- Dedicated
 - Functional: separate data & address lines.
 - Physical: multiple buses connecting different subsets of modules.
- Multiplexed
 - Shared lines.
 - Address valid or data valid control line.
 - Advantages:
 - fewer lines.
 - Disadvantages
 - More complex control.
 - Lower performance.

Bus Arbitration (I)

- More than one module controlling the bus.
 - e.g. CPU and DMA controller.
- Only one module may control bus at a time.
- Need a special h/w for arbitration.
- Arbitration may be centralized or distributed.

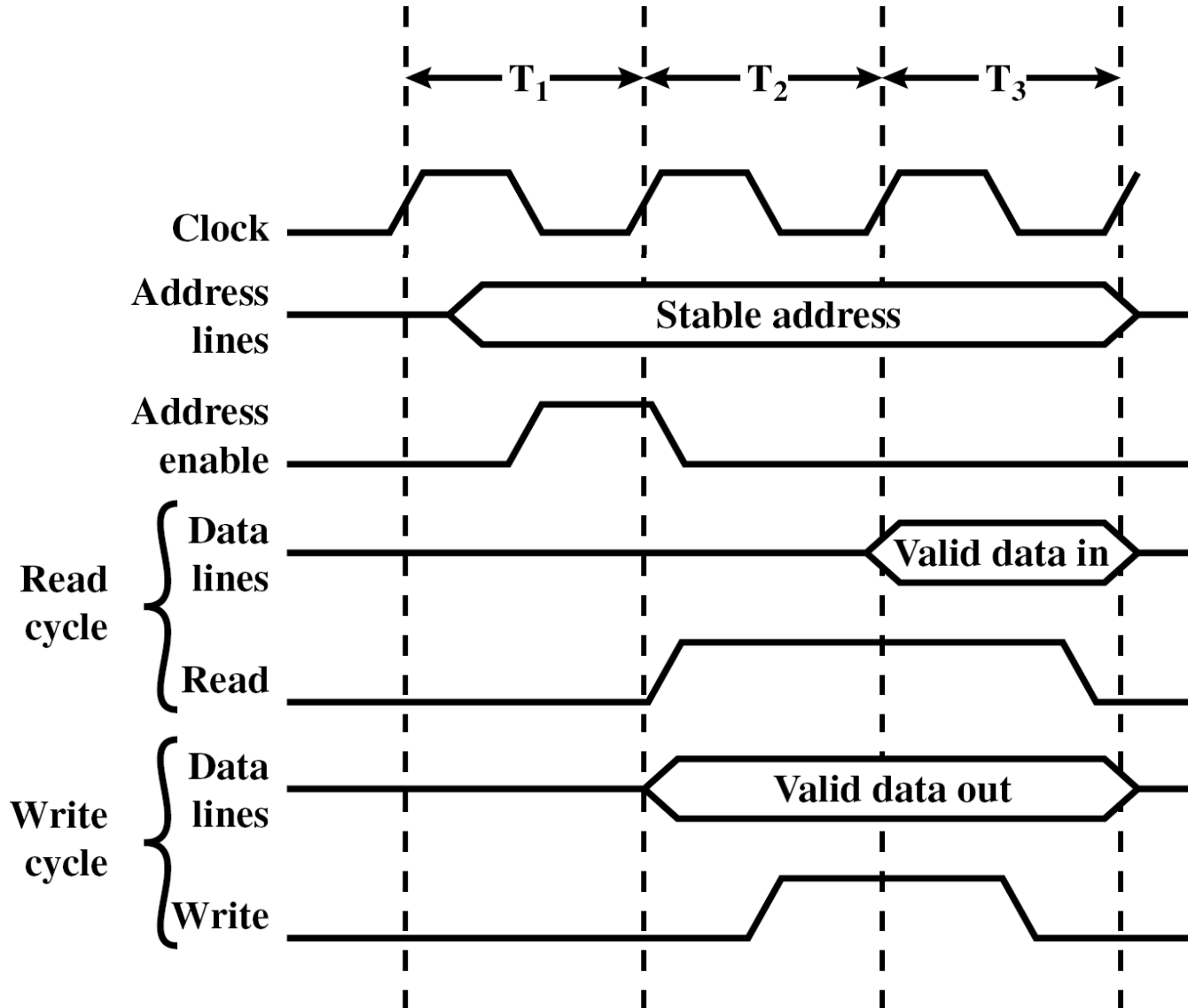
Bus Arbitration (II)

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

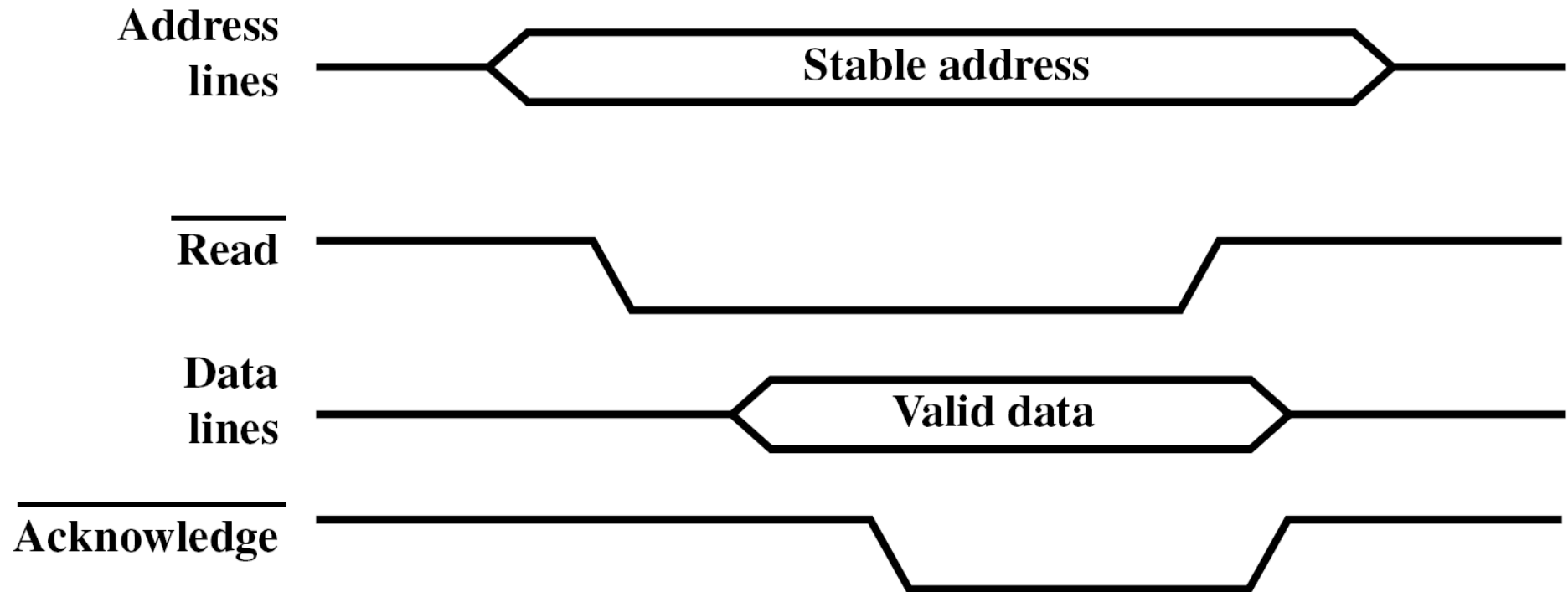
Timing

- Coordination of events on bus.
- Synchronous
 - Events determined by clock signals.
 - Clock line belongs to the control bus.
 - A single 1-0 is a bus cycle.
 - All devices can read clock line.
 - Usually sync on leading edge.
 - Usually a single cycle for an event.
- Asynchronous
 - Occurrence of one event on a bus follows and depends on the occurrence of a previous events.

Synchronous Timing Diagram



Asynchronous Timing Diagram - Read



Asynchronous Timing Diagram - Write

**Address
lines**



**Data
lines**



Write



Acknowledge



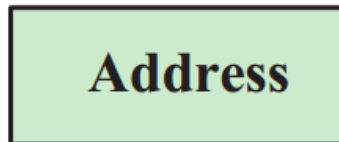
Data Transfer Type (I)

Time →



Read (non-multiplexed) operation

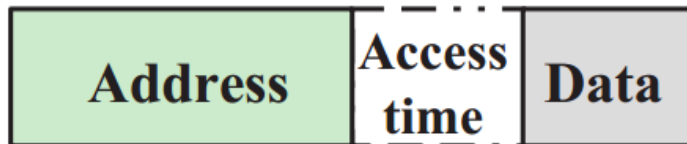
Time →



Data and address sent by master in same cycle over separate bus lines.

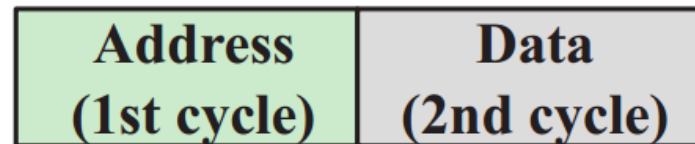
Write (non-multiplexed) operation

Time →



Read (multiplexed) operation

Time →

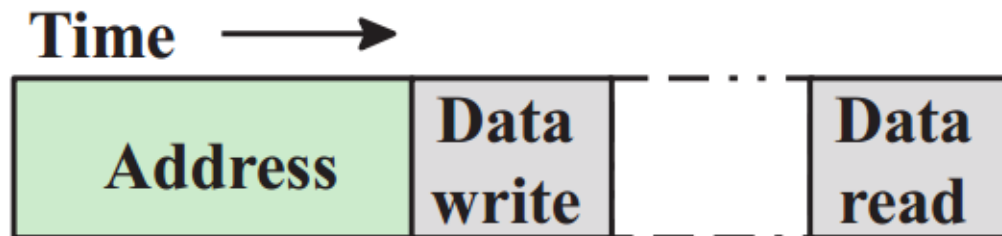


Write (multiplexed) operation

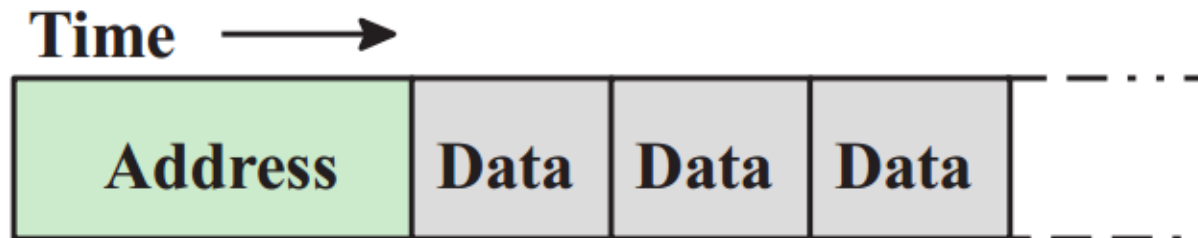
Data Transfer Type (II)



Read-modify-write operation



Read-after-write operation



Block data transfer

Reading Material

- Stallings, Chapter 3:
 - Pages 80 – 93