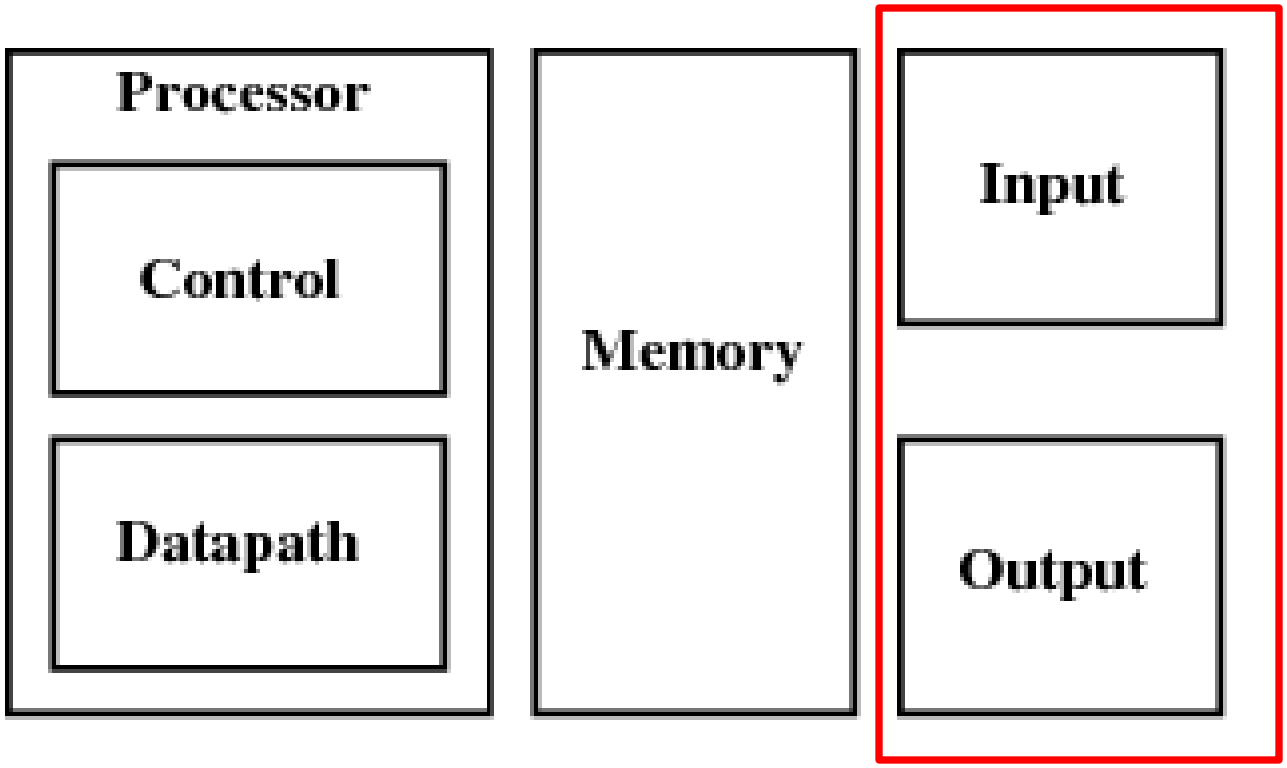
ACK: Some of these slides are based on Stallings

COMPUTER SYSTEMS ORGANIZATION

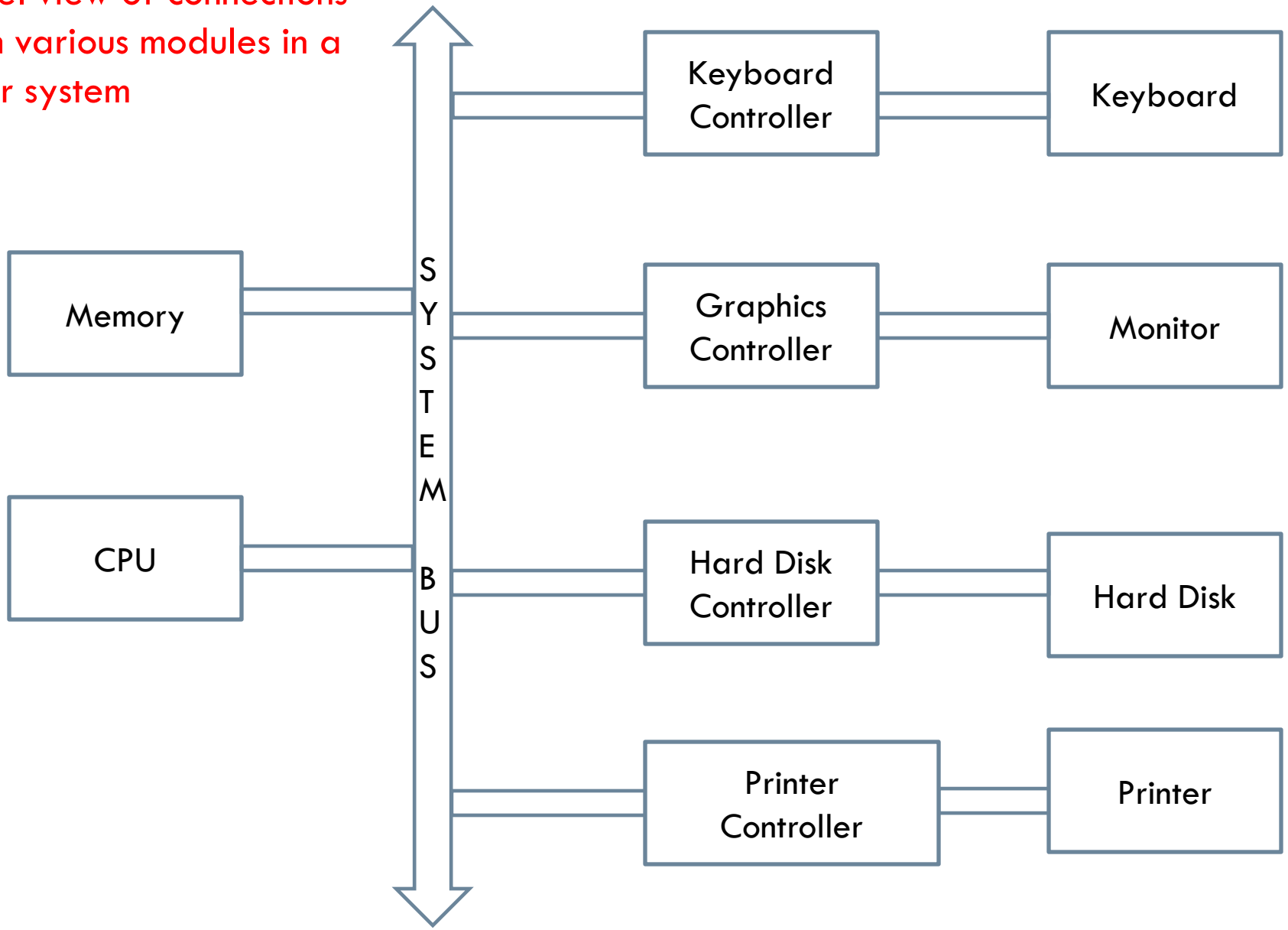
Input/Output -- Spring 2010 -- IIIT-H -- Suresh Purini

The Big Picture: Where are we now?

Input and Output

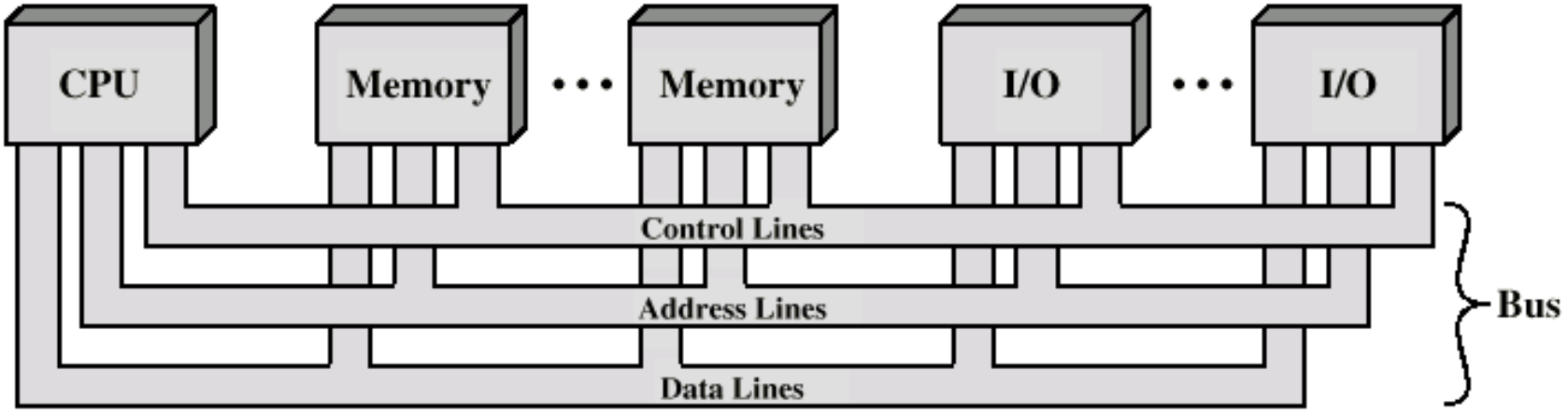


High level view of connections between various modules in a computer system



System Bus Consists of:

- Address lines
- Data lines
- Control lines



Typical Control Lines:

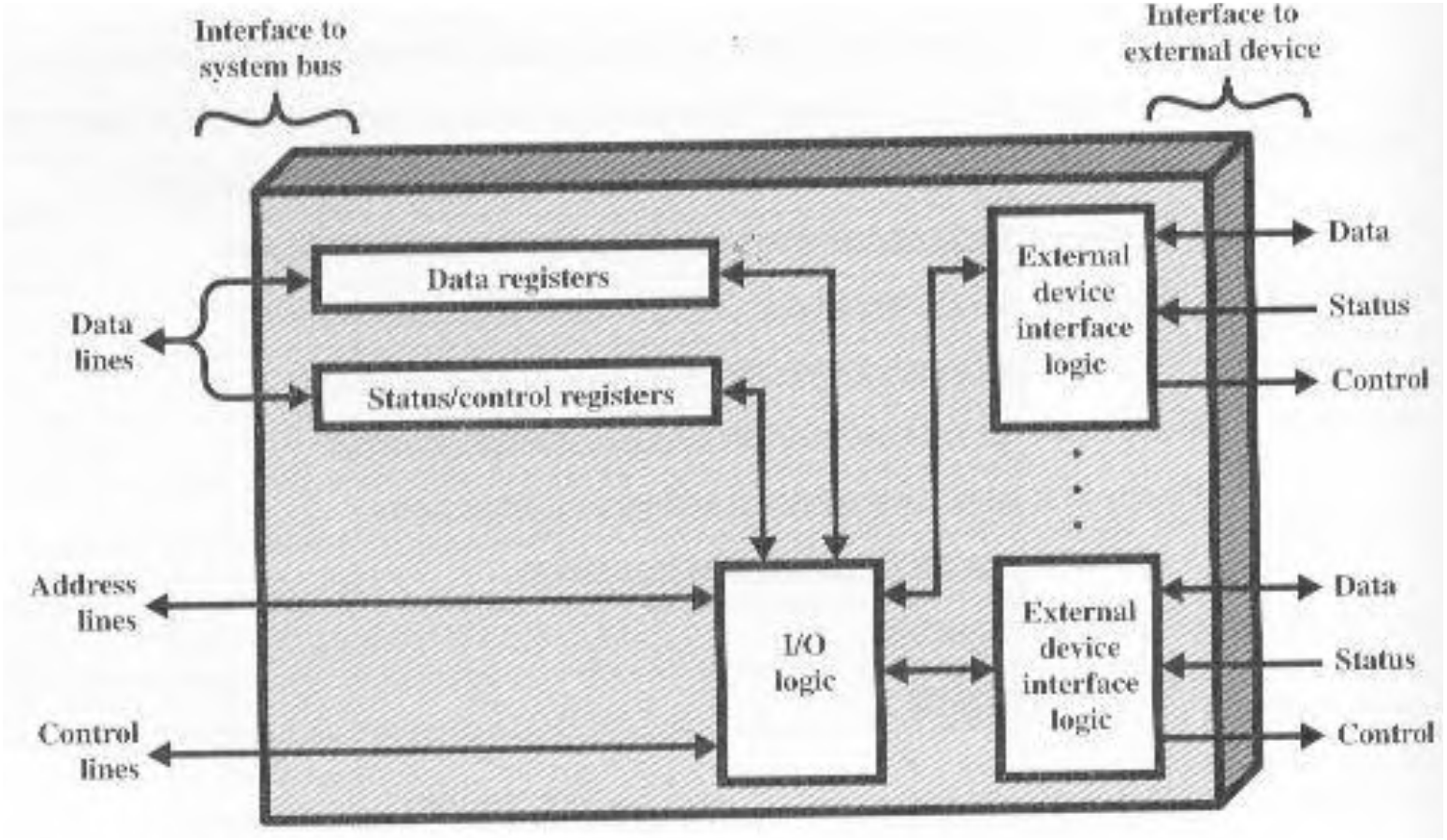
- Memory write
- Memory read
- I/O write
- I/O read
- Transfer ACK
- Bus request
- Bus grant
- Interrupt request
- Interrupt ACK
- Clock

Input/Output Problems

- Wide variety of peripherals
  - Delivering different amounts of data
  - At different speeds
  - In different formats
- All slower than CPU and RAM
- Need I/O modules (with their Device Drivers and the corresponding Operating System support)

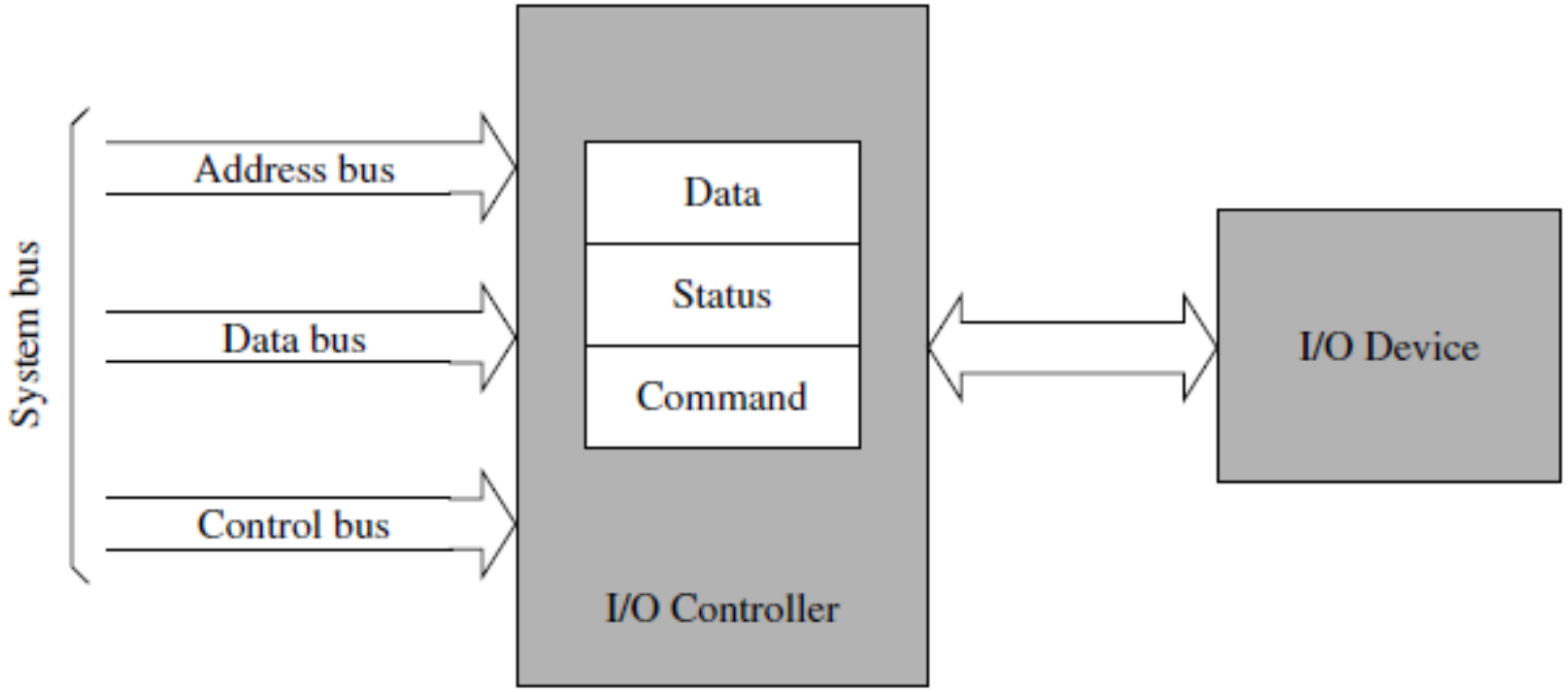
I/O Device Interface

- Interface to CPU and Memory
- Interface to one or more peripherals



I/O Device Interface

- Interface to CPU and Memory
- Interface to one or more peripherals



I/O Module Functions

- Control & Timing
- CPU Communication
- Device Communication
- Data Buffering
- Error Detection

I/O Steps

- CPU checks I/O module device status
- I/O module returns status
- If ready, CPU requests data transfer
- I/O module gets data from device
- I/O module transfers data to CPU
- Variations for output, DMA, etc.

I/O Decisions

- Hide or reveal device properties to CPU
- Support multiple or single device
- Control device functions or leave for CPU

Input Output Techniques

- Programmed I/O
- Interrupt driven
- Direct Memory Access (DMA)

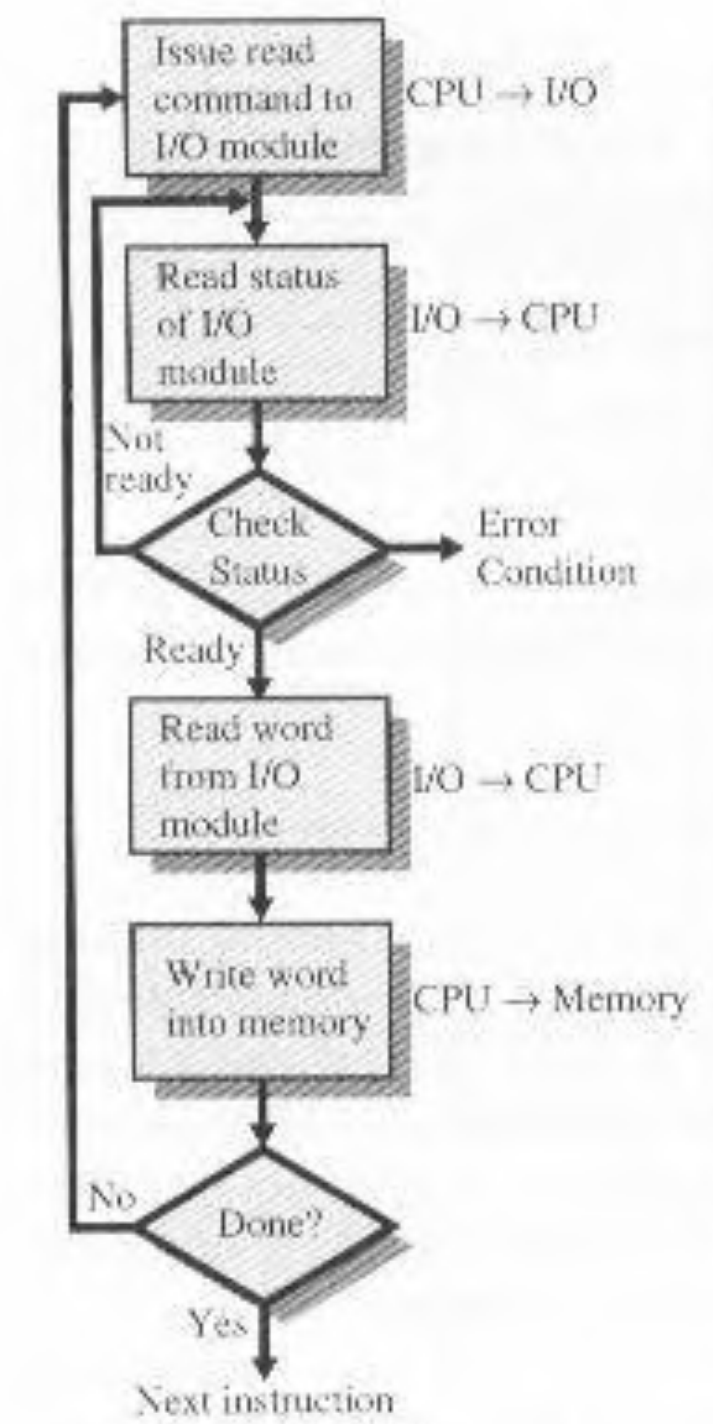
Programmed I/O

- CPU has direct control over I/O
  - Sensing status
  - Read/write commands
  - Transferring data
- CPU waits for I/O module to complete operation
- Wastes CPU time

Programmed I/O - detail

- CPU requests I/O operation
- I/O module performs operation
- I/O module sets status bits
- CPU checks status bits periodically
- I/O module does not inform CPU directly
- I/O module does not interrupt CPU
- CPU may wait or come back later

Programmed I/O



I/O Mapping

- Memory mapped I/O
  - Devices and memory share an address space
  - I/O looks just like memory read/write
  - No special commands for I/O
    - Large selection of memory access commands available
- Isolated I/O (I/O Mapped I/O)
  - Separate address spaces
  - Need I/O or memory select lines
  - Special commands for I/O
    - Limited set

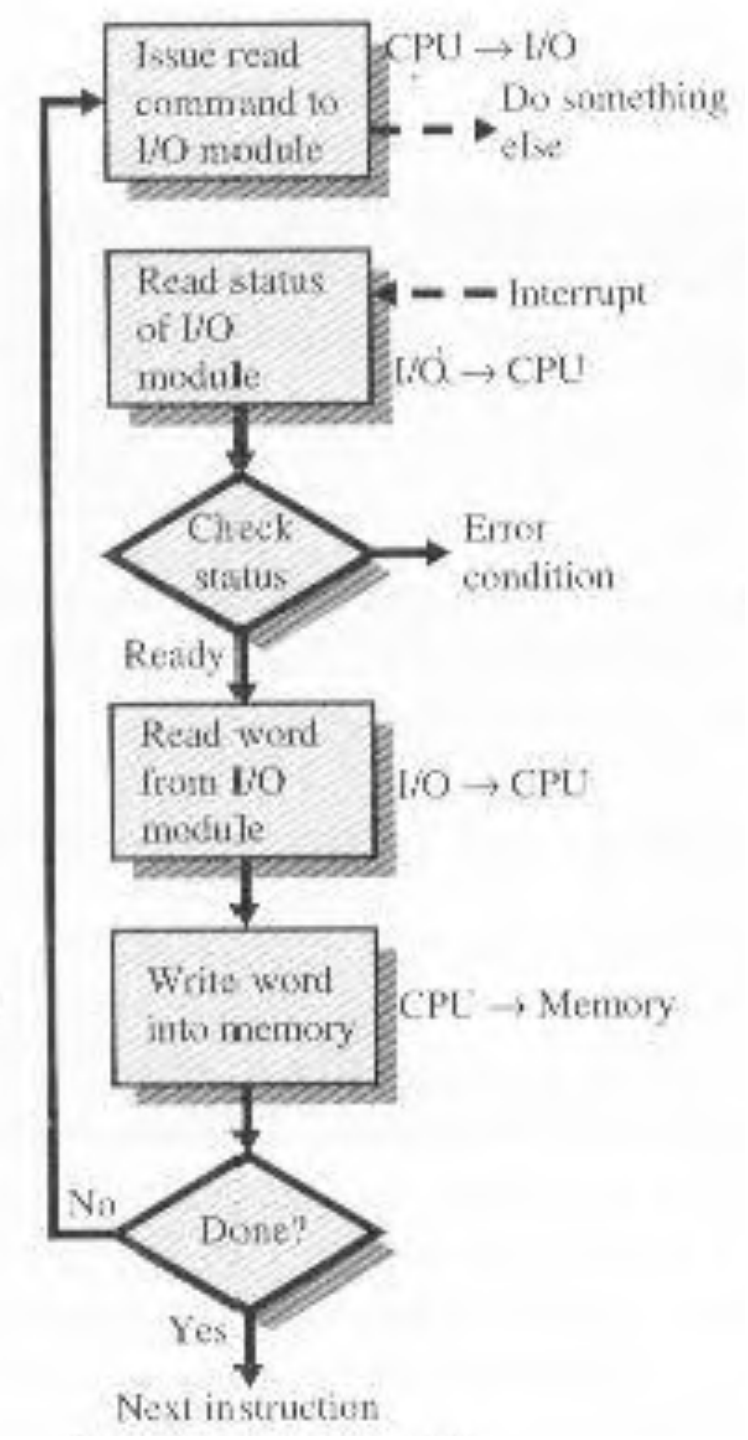
Interrupt Driven I/O

- Overcomes CPU waiting
- No repeated CPU checking of device
- I/O module interrupts when ready

Interrupt Driven I/O Basic Operation

- CPU issues read command
- I/O module gets data from peripheral whilst CPU does other work
- I/O module interrupts CPU
- CPU requests data
- I/O module transfers data

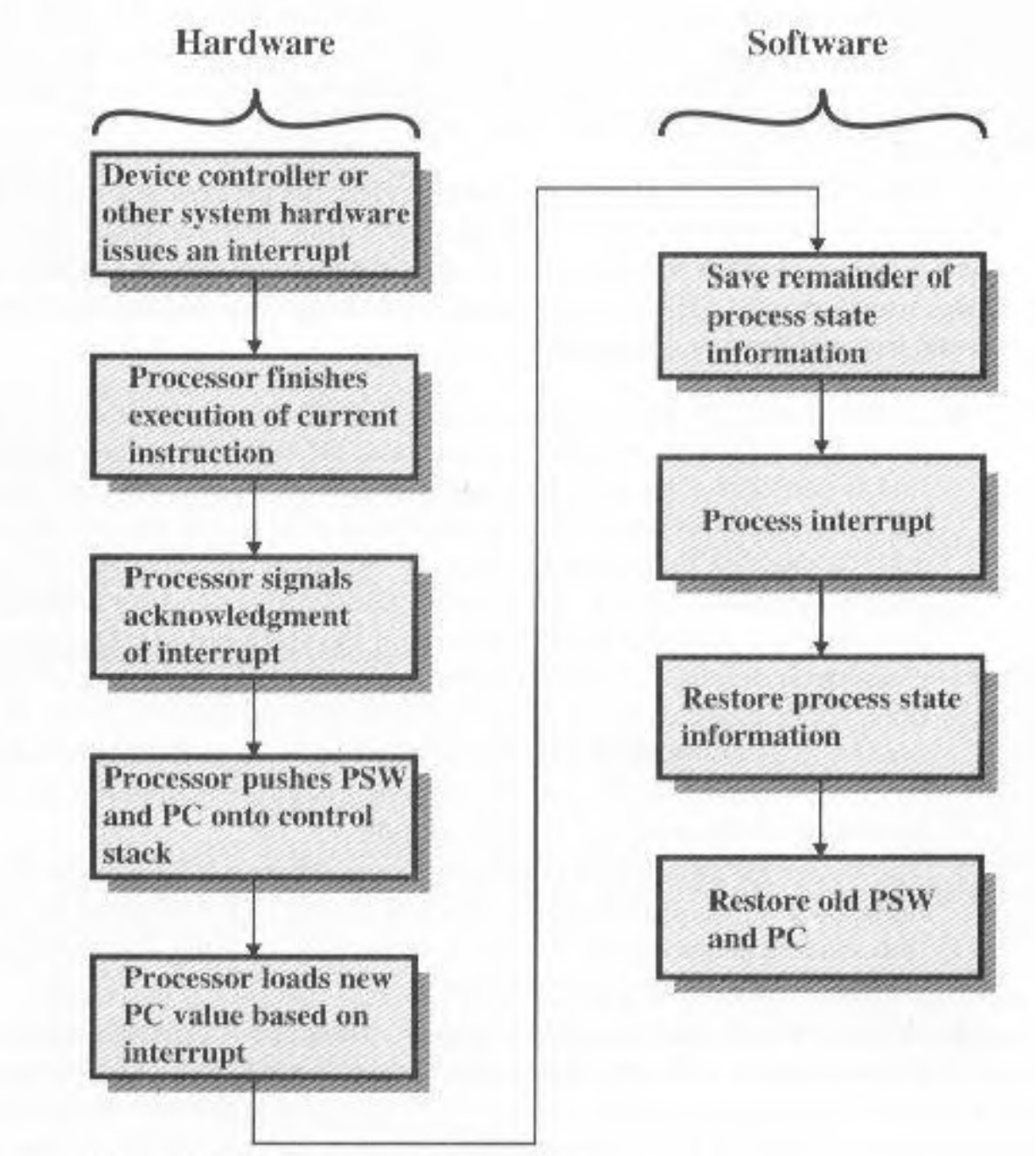
Interrupt Driven I/O



CPU Viewpoint

- Issue read command
- Do other work
- Check for interrupt at end of each instruction cycle
- If interrupted:-
  - Save context (registers)
  - Process interrupt
    - Fetch data & store

Interrupt Processing



Design Issues

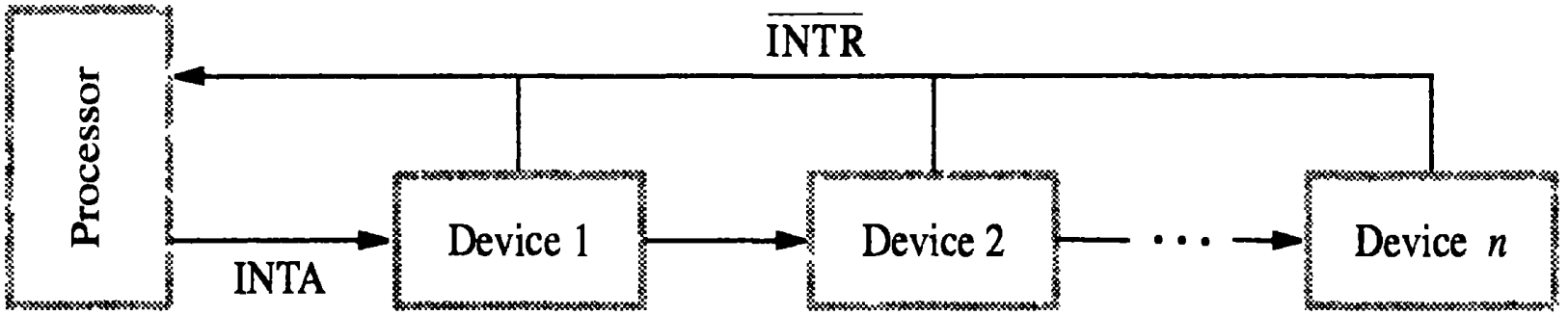
- How do you identify the module issuing the interrupt?
- How do you deal with multiple interrupts?
  - i.e. an interrupt handler being interrupted

Identifying Interrupting Module

- Different line for each module
  - PC
  - Limits number of devices
- Software poll
  - CPU asks each module in turn
  - Slow

Identifying Interrupting Module

- Daisy Chain or Hardware poll
  - Interrupt Acknowledge sent down a chain
  - Module responsible places vector on bus
  - CPU uses vector to identify handler routi



- Bus Master
  - Module must claim the bus before it can raise interrupt

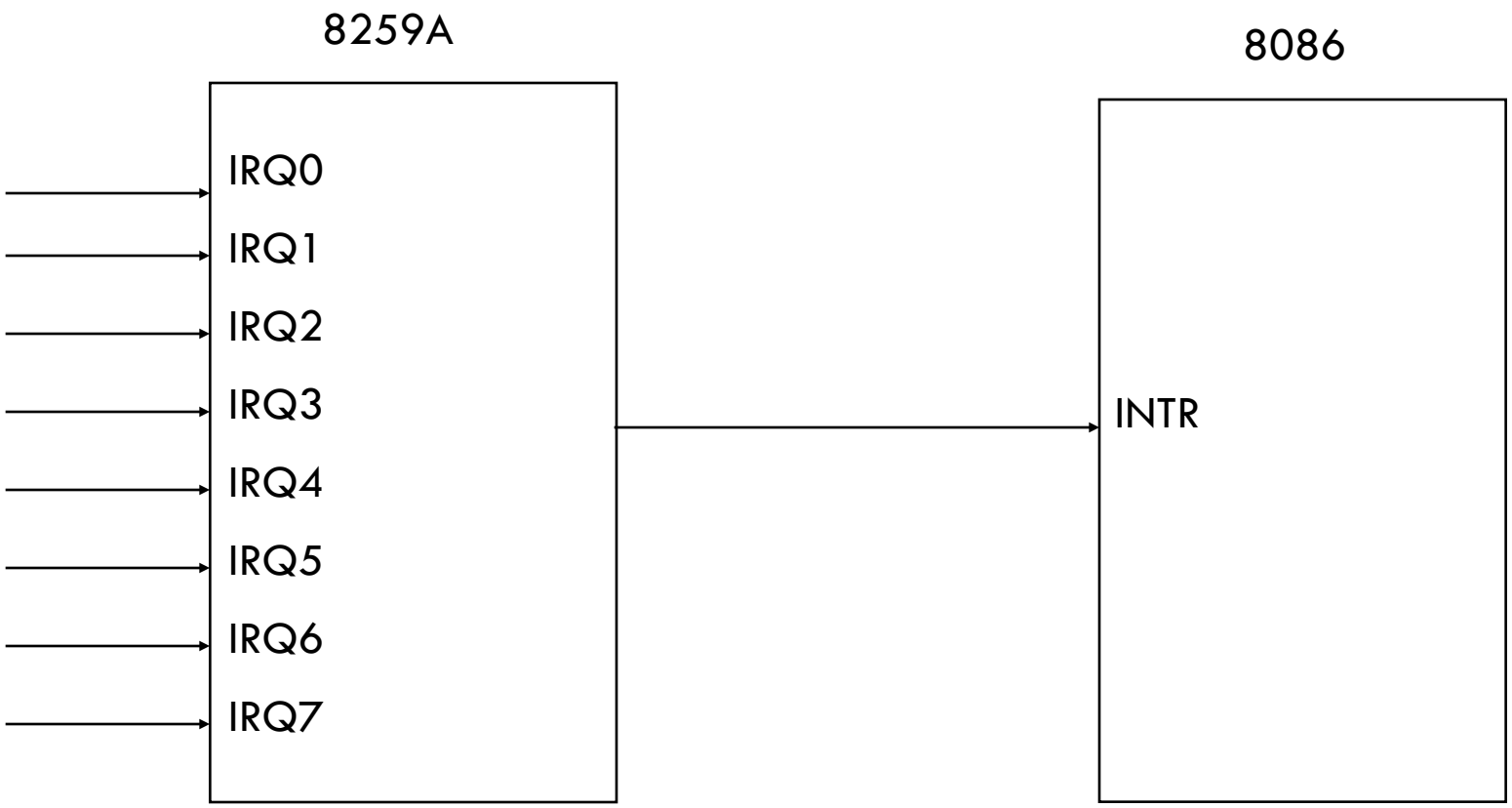
Example

- 80x86 has one interrupt line
- 8086 based systems use one 8259A interrupt controller
- 8259A has 8 interrupt lines

Sequence of Events

- 8259A accepts interrupts
- 8259A determines priority
- 8259A signals 8086 (raises INTR line)
- CPU Acknowledges
- 8259A puts correct vector on data bus
- CPU processes interrupt

PC Interrupt Layout



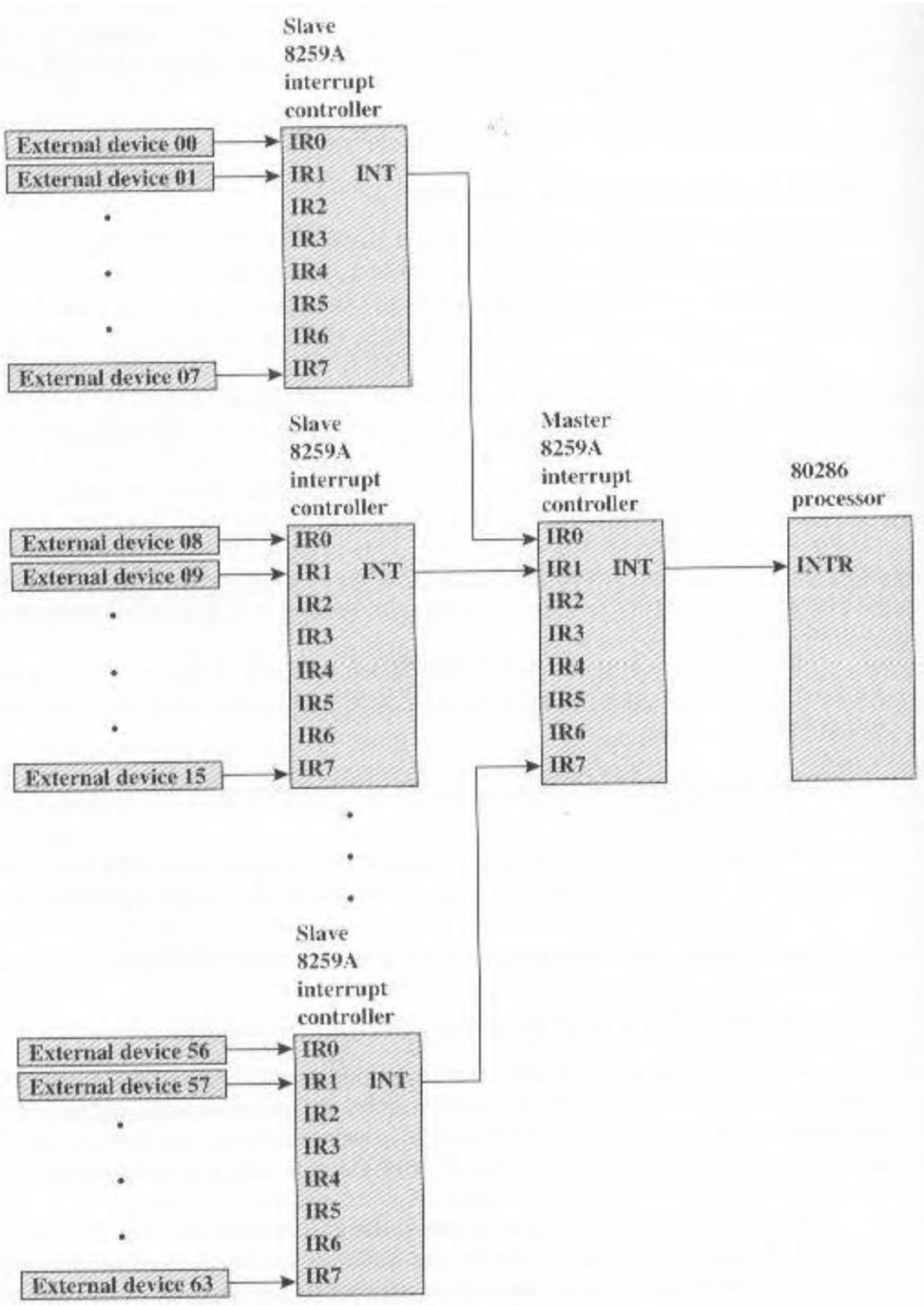
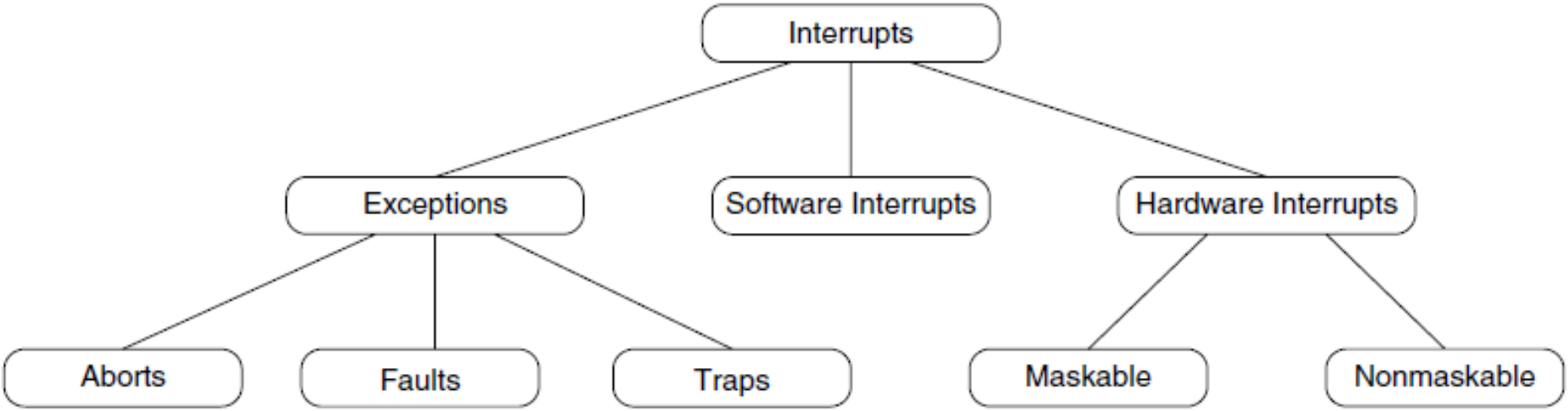


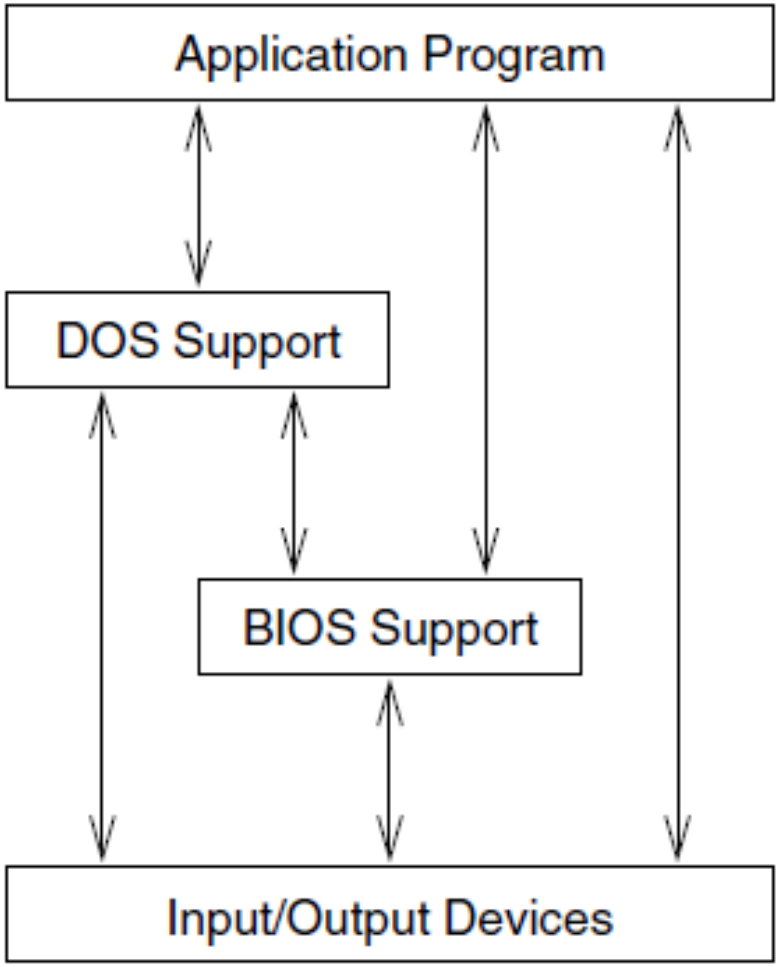
Figure 7.9 Use of the 82C59A Interrupt Controller

Taxonomy of Interrupts in Pentium



Software Interrupts are Synchronous Events whereas Hardware Interrupts are Asynchronous Events.

Interacting with I/O Devices: MS-Dos on an x86 Machine

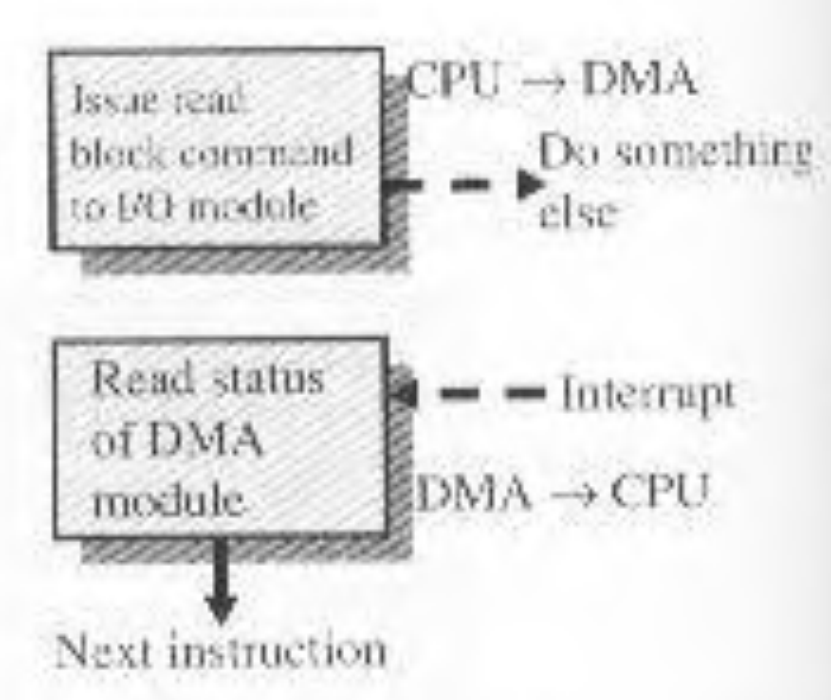


Pentium Modes

- Real Mode
- Protected Mode

Pentium

Direct Memory Access



I/O Techniques Summary

	No Interrupts	Use of Interrupts
I/O-to-memory Transfer through Processor	Programmed I/O	Interrupt-driven I/O
Direct I/O-to-memory transfer		Direct Memory Access