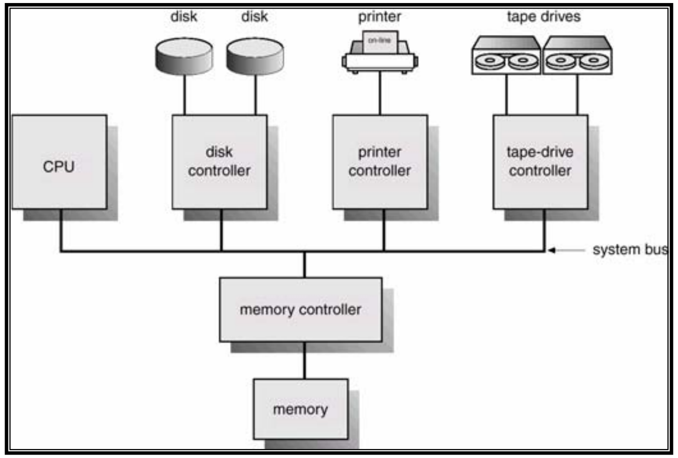
Overview of Computer Systems (Topic 1-2)

홍 성 수

서울대학교 공과대학 전기 공학부
Real-Time Operating Systems Laboratory

Computer System Architecture



Computer System Operation

- I/O devices and the CPU can execute concurrently.
- Each device controller is in charge of a particular device type.
- Each device controller has a local buffer.
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller.
- Device controller informs CPU that it has finished its operation by causing an *interrupt*.

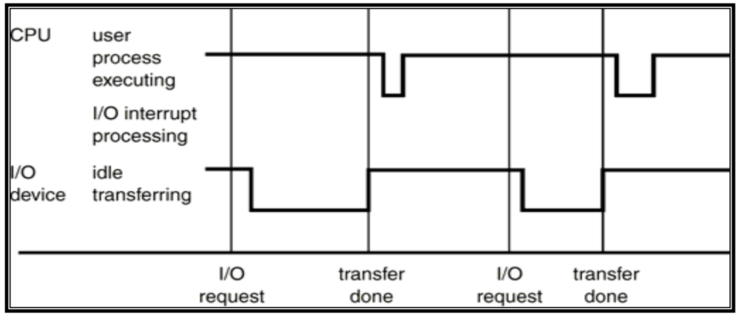
Common Functions of Interrupts

- Interrupt transfers control to the interrupt service routine generally, through the *interrupt vector*, which contains the addresses of all the service routines.
- Interrupt architecture must save the address of the interrupted instruction.
- Incoming interrupts are *disabled* while another interrupt is being processed to prevent a *lost interrupt*.
- A *trap* is a software-generated interrupt caused either by an error or a user request.
- An operating system is *interrupt* driven.

Interrupt Handling

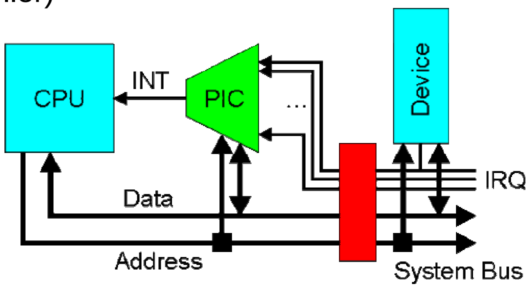
- The operating system preserves the state of the CPU by storing registers and the program counter.
- Determines which type of interrupt has occurred:
  - *polling*
  - *vectored* interrupt system
- Separate segments of code determine what action should be taken for each type of interrupt

Interrupt Timeline



General Intel Interrupt Architecture

- Connection thru PIC (programmable interrupt controller)

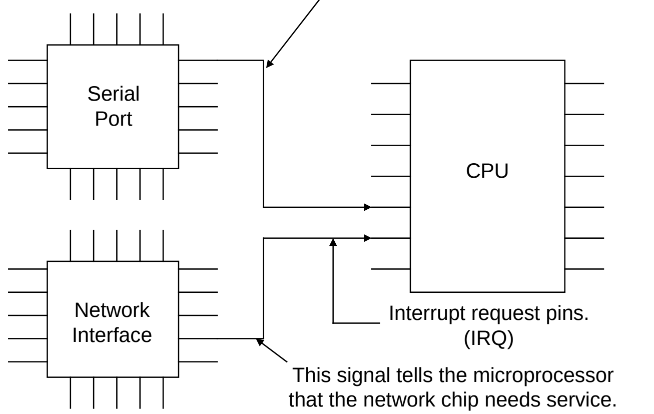


Interrupt Hardware

- Interrupts start with a signal from hardware.
- Each of most I/O chips has a pin that it asserts when it requires service.
- Microprocessor has input pins called "*interrupt request (IRQ)*" that let the microprocessor know that some other chip in the system needs attention.

Connection with Processor

- Simplified View



Programmable Interrupt Controller

- There are many I/O devices in a computer system.
- The PIC (Programmable Interrupt Controller) functions as an overall manager in an interrupt-driven system environment.

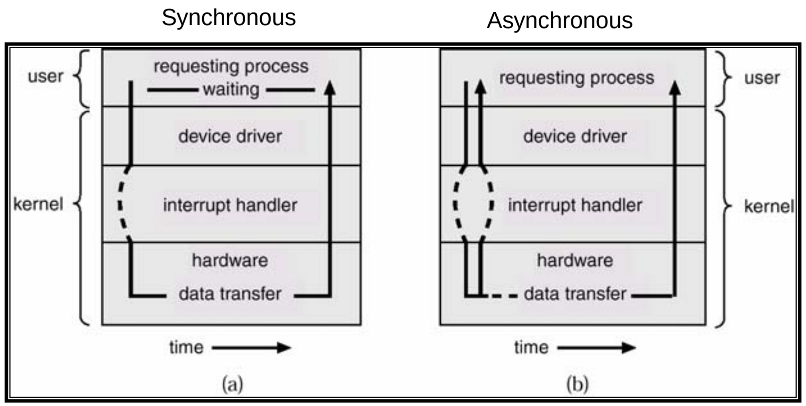
Programmable Interrupt Controller

- PIC performs the following actions
  - Monitors IRQ lines, checking for raised signals.
  - If a raised signal occurs on an IRQ line:
    - Converts the received raised signal into a corresponding vector.
    - Stores the vector in an interrupt controller I/O port, thus allowing the CPU to read it via the data bus.
    - Sends a raised signal to the processor, INTR pin – that is, issues an interrupt.
    - Waits until the CPU acknowledges the interrupt signal by writing into one of the PIC I/O ports; when this occurs, clears the INTR line.
  - Goes back to the first step.

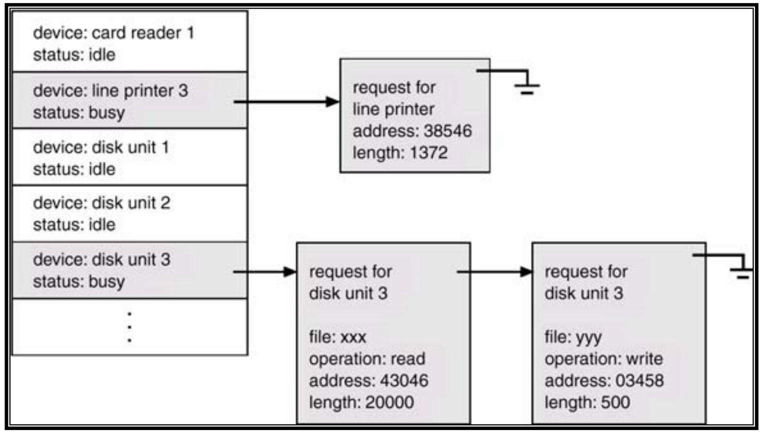
I/O Structure

- After I/O starts, control returns to user program only upon I/O completion: Synchronous I/O
  - Wait instruction idles the CPU until the next interrupt
  - Wait loop (contention for memory access).
  - At most one I/O request is outstanding at a time, no simultaneous I/O processing.
- After I/O starts, control returns to user program without waiting for I/O completion: Asynchronous I/O
  - *System call* – request to the operating system to allow user to wait for I/O completion.
  - *Device-status table* contains entry for each I/O device indicating its type, address, and state.
  - Operating system indexes into I/O device table to determine device status and to modify table entry to include interrupt.

Two I/O Methods



Device Status Table



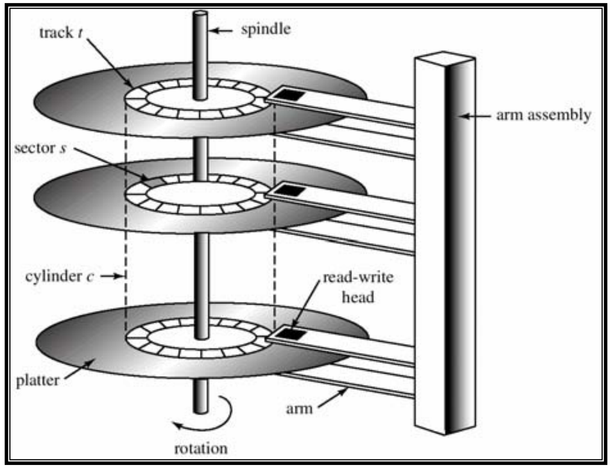
Direct Memory Access Structure

- Used for high-speed I/O devices able to transmit information at close to memory speeds.
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention.
- Only one interrupt is generated per block, rather than the one interrupt per byte.

Storage Structure

- Main memory – only large storage media that the CPU can access directly.
- Secondary storage – extension of main memory that provides large nonvolatile storage capacity.
- Magnetic disks – rigid metal or glass platters covered with magnetic recording material
  - Disk surface is logically divided into *tracks*, which are subdivided into *sectors*.
  - The *disk controller* determines the logical interaction between the device and the computer.

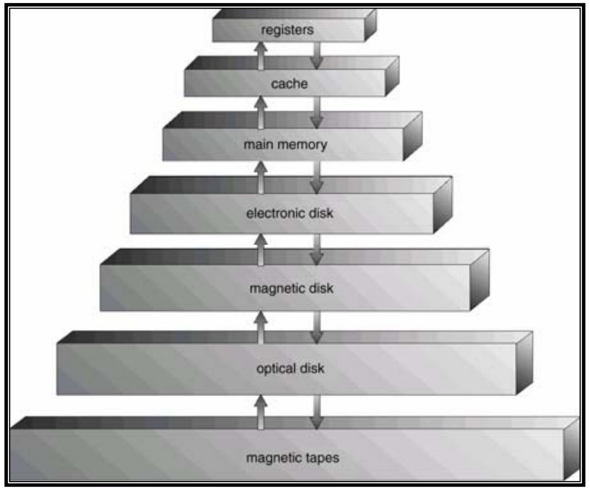
Moving-Head Disk Mechanism



Storage Hierarchy

- Storage systems organized in hierarchy.
  - Speed
  - Cost
  - Volatility
- *Caching* – copying information into faster storage system. Main memory can be viewed as a last *cache* for secondary storage.

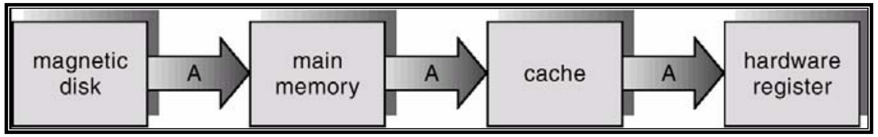
Storage Device Hierarchy



Caching

- Use of high-speed memory to hold recently-accessed data.
- Requires a *cache management* policy.
- Caching introduces another level in storage hierarchy. This requires data that is simultaneously stored in more than one level to be *consistent*.

Migration of A from Disk to register



Hardware Protection

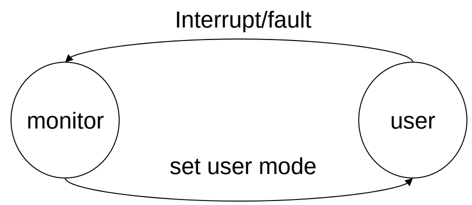
- Dual Mode Operation
- I/O Protection
- Memory Protection
- CPU Protection

Dual Mode Operation (1)

- Sharing system resources requires operating system to ensure that an incorrect program cannot cause other programs to execute incorrectly.
- Provide hardware support to differentiate between at least two modes of operations.
  1. *User mode* – execution done on behalf of a user.
  2. *Monitor mode* (also *kernel mode* or *system mode*) – execution done on behalf of operating system.

Dual Mode Operation (2)

- *Mode bit* added to computer hardware to indicate the current mode: monitor (0), user (1).
- When an interrupt or fault occurs hardware switches to monitor mode.

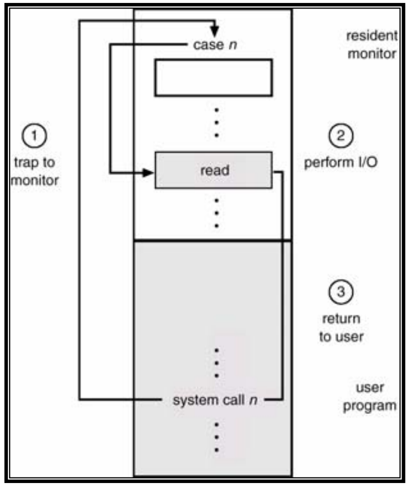


Privileged instructions can be issued only in monitor mode.

I/O Protection

- All I/O instructions are privileged instructions.
- Must ensure that a user program could never gain control of the computer in monitor mode (I.e., a user program that, as part of its execution, stores a new address in the interrupt vector).

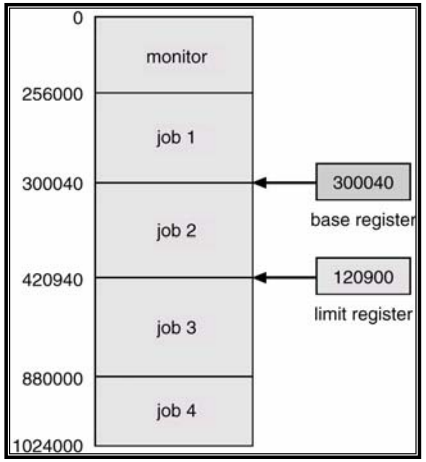
System Call to Perform I/O



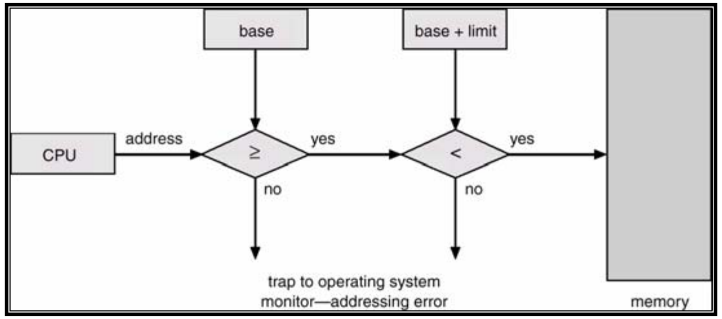
Memory Protection

- Must provide memory protection at least for the interrupt vector and the interrupt service routines.
- In order to have memory protection, add two registers that determine the range of legal addresses a program may access:
  - **Base register** – holds the smallest legal physical memory address.
  - **Limit register** – contains the size of the range
- Memory outside the defined range is protected.

Base and Limit Register



Hardware Address Protection



Hardware Protection

- When executing in monitor mode, the operating system has unrestricted access to both monitor and user's memory.
- The load instructions for the *base* and *limit* registers are privileged instructions.

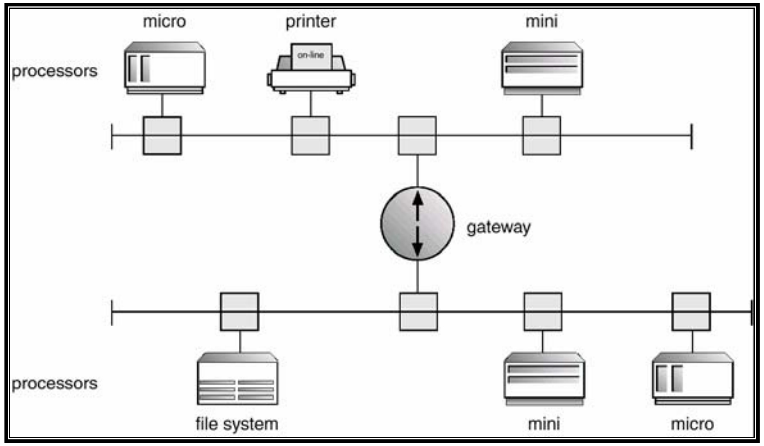
CPU Protection

- *Timer* – interrupts computer after specified period to ensure operating system maintains control.
  - Timer is decremented every clock tick.
  - When timer reaches the value 0, an interrupt occurs.
- Timer commonly used to implement time sharing.
- Time also used to compute the current time.
- Load-timer is a privileged instruction.

Network Structure

- Local Area Networks (LAN)
- Wide Area Networks (WAN)

Local Area Network Structure



Wide Area Network Structure