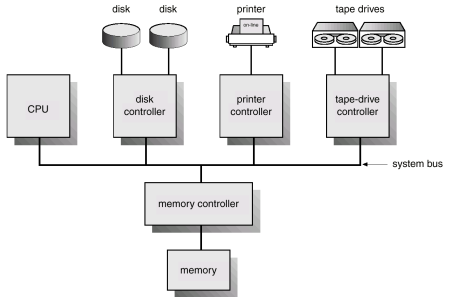
Module 2: Computer-System Structures

- Computer System Operation
- I/O Structure
- Storage Structure
- Storage Hierarchy
- Hardware Protection
- General System Architecture

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

- Interrupts 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*
- An operating system is *interrupt* driven

Trap

- A trap is a software-generated interrupt caused either by an error or a user request
- A trap is typically a **synchronous** event, for example
  - Division by zero
  - Memory access violation
- The program can restart from the interrupted operation, rather than from the following instruction

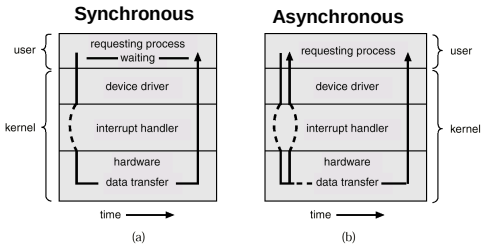
Interrupt Handling

- 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

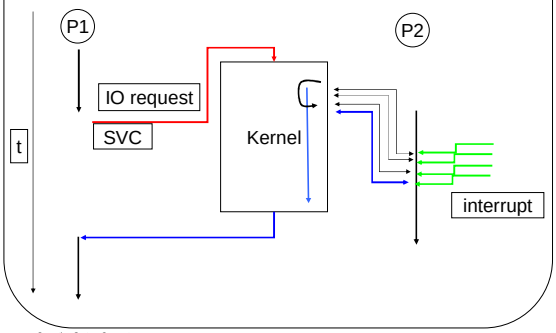
I/O Structure

- After I/O starts, control returns to user program only upon I/O completion.
  - 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.

Two I/O methods



I/O interrupt managment



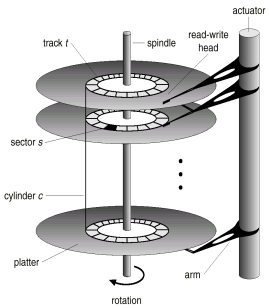
Direct Memory Access (DMA) Structure

- 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.

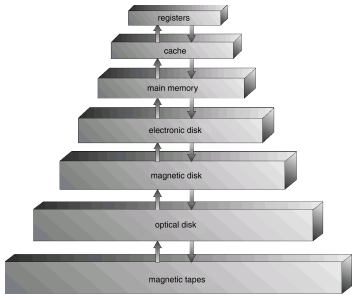
Moving-Head Disk Mechanism



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



Hardware Protection

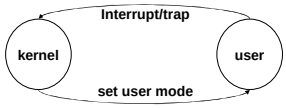
- Dual-Mode Operation
- I/O Protection
- Memory Protection
- CPU Protection

Dual-Mode Operation

- 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. Kernel mode (also supervisor mode or system mode) – execution done on behalf of operating system.

Dual-Mode Operation (Cont.)

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

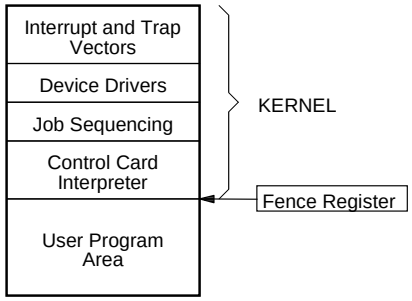


- **Privileged instructions**, that can be issued only in kernel mode.

I/O Protection

- All I/O instructions are privileged instructions.
- Must ensure that a user program could never gain control of the computer in kernel mode
- I.e., a user program cannot store a new address in the interrupt vector

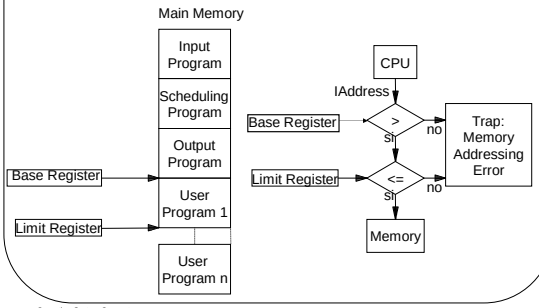
Memory layout for a Batch System



Memory Protection

- Must provide memory protection at least for the interrupt vector and the interrupt service routines.
- In order to have memory protection, add a fence register
  - **fence register** – holds the smallest legal physical memory address.
- Memory addresses less than the content of the fence register are protected.

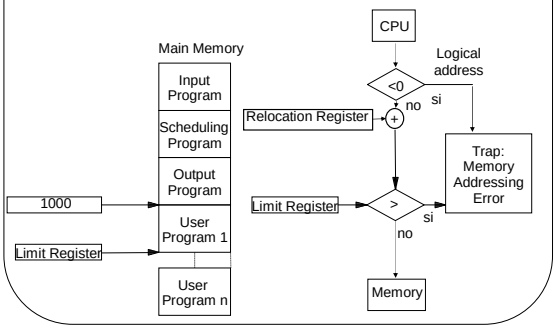
Memory Protection Hardware



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.

Memory Protection Hardware - Relocation



Protection Hardware

- When executing in monitor mode, the operating system has unrestricted access to both monitor and user's memory.
- The load instructions for the base (relocation) 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
- Timer also used to compute the current time
- Load-timer is a privileged instruction

General-System Architecture

- Given the I/O instructions are privileged, how does the user program perform I/O?
- More generally, how does a user program call a kernel function?

General-System Architecture

- System call – the method used by a process to request action by the operating system.
  - Usually takes the form of a trap to a specific location in the interrupt vector.
  - Control passes through the interrupt vector to a service routine in the OS, and the mode bit is set to kernel mode.
  - The monitor verifies that the parameters are correct and legal, executes the request, and returns control to the instruction following the system call.

Use of a System Call to Perform I/O