Input and Output (I/O)

- Important topic
  - A large part of any OS is concerned with I/O
- Need to move data into and out of a system (between I/O devices and memory)
- Challenges with I/O devices:
  - Different categories: e.g. storage, networking, displays
  - Many device drivers to support
  - Device drivers can crash systems

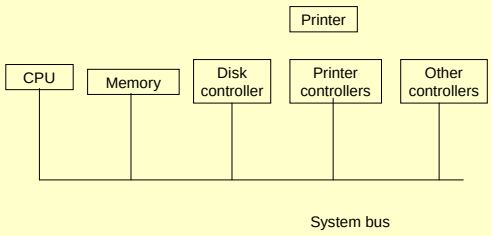
Input and Output

- OS Issues
  - Provide a generic, consistent, convenient and reliable way to access I/O devices
  - Device independence
  - Uniform naming
  - Limit impact on I/O system performance capability
  - Handle errors close to h/w
  - Synchronous (blocking) vs asynchronous (interrupt-driven)
  - Sharable vs dedicated devices

Hardware

- Computer
  - CPU, memory
- I/O hardware
  - I/O bus
  - I/O controller (adaptor)
  - I/O device
- Two types of I/O
  - Programmed I/O: CPU does the work of moving data
  - Direct Memory Access (DMA): DMA controller moves the data (not the CPU)

Connections



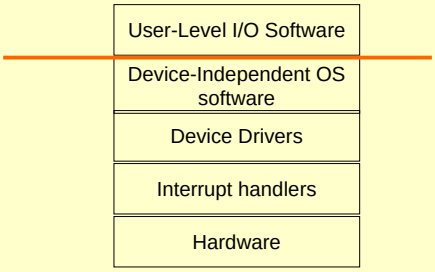
Definitions

- Overhead
  - Time that the CPU is tied up initiating/finishing an operation
- Latency
  - Time to transfer one byte (overhead + 1 byte)
- Bandwidth
  - Rate of I/O transfer (once initiated)
  - Mbytes/sec
- Method
  - use abstractions
  - Group transfers into blocks for efficiency (to lessen effects of overhead and latency)

I/O Devices

- Block devices
  - Stores information in fixed-size blocks
  - Transfers are in units of blocks
  - Blocks have addresses and data and are addressable
  - E.g. hard disks, CD-ROMs
- Character devices
  - Delivers or accepts a stream of characters
  - Not addressable
  - printers

I/O Software Levels



Input/Output Layers

- Interrupt handlers
  - Hide inside OS
  - Process is blocked until i/o request finished
  - Interrupt does its job and unblocks process
- Device drivers
  - Job is to accept abstract requests from device-independent s/w and execute that request

I/O Layers

- Device-Independent I/O software
  - Uniform interface for the device drivers
  - Device naming
  - Device protection
  - Provide a device-independent block size
  - Buffering
  - Storage allocation on block devices
  - Allocation and releasing dedicated devices
  - Error reporting
- User-Space I/O software
  - System calls via library procedures
  - E.g. printer: Spooling, daemon, spooling directory

Devices

- Programmed Input Device
  - Device controller
    - Status register (ready/busy/int (interrupt))
    - Data registers
  - Mouse design
    - Put (x, y) values in data registers
    - Interrupt
  - Input on interrupt
    - Read in values in x, y registers
    - Set ready bit
    - Wake up process/ execute a code

Devices

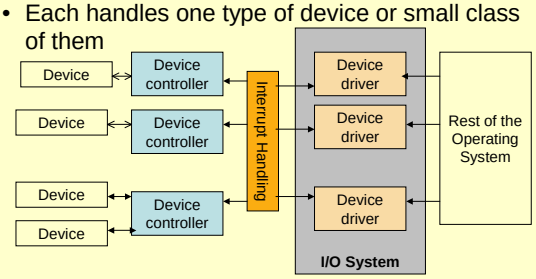
- Programmed Output Device
  - Device
    - Status registers
    - Data registers
  - Example
    - Serial output device
  - Perform on Output
    - Wait until ready bit is clear
    - Poll the busy bit
    - Writes data to registers
    - Set ready bit
    - Controller sets the busy bit and transfers data
    - Controller clears the ready bit and busy bit

Direct Memory Access (DMA)

- DMA controller
  - Status register
  - DMA command register
  - DMA register
  - DMA buffer
- CPU initiates DMA
  - Device driver call
  - Initiate DMA transaction
  - Block
- Controller performs DMA
- Interrupt handler
  - Wakes up blocked process

Device Drivers

- Manage complexity and differences among specific types of devices (disk, mouse etc.)
- Each handles one type of device or small class of them



Device Driver Design

- Driver and OS communication
  - Commands and data between drivers and OS
- Driver and hardware communication
  - Commands and data between driver and h/w
- Driver operations
  - Initialise devices
  - Interpreting commands from OS
  - Schedule multiple outstanding requests
  - Manage data transfers
  - Accept and process interrupts
  - Maintain the integrity of driver and kernel data structures

Simplified Behaviour

- Check input parameters and translate them to device-specific language
- Check if device is free
- Issue commands to control device
- Block or wait for controller to finish
- Check for errors and pass data to device-specific s/w
- Return status info
- Challenges
  - Must be reentrant (it can be interrupted)
  - Handle hot-pluggable devices and device removal while running
  - complex

Synchronous vs Asynchronous I/O

- Synchronous I/O
  - read or write will block a user process until its completion
  - OS overlaps synchronous I/O with another process
- Asynchronous I/O
  - Read or write will not block a user process
  - User process can do other things before I/O completion
  - I/O completion will notify the user process