So far...

- We have covered CPU and memory management.
- Computing is not interesting without I/Os.
- **Device management**: the OS component that manages hardware devices.
  - Provides a uniform interface to access devices with different physical characteristics.
  - Optimizes the performance of individual devices.
Basics of I/O Devices

- Three categories
  - A block device stores information in fixed-size blocks, each one with its own address
    - e.g., disks
  - A character device delivers or accepts a stream of characters, and individual characters are not addressable
    - e.g., keyboards, printers
  - A network device transmit data packets
**Device Controller**

- Converts between serial bit stream and a block of bytes
- Performs error correction if necessary
- Components:
  - Device registers to communicate with the CPU
  - Data buffer that an OS can read or write
Device Driver

- An OS component that is responsible for hiding the complexity of an I/O device
- So that the OS can access various devices in a uniform manner
Device Driver Illustrated

- User level
  - User applications
  - Various OS components

- OS level
  - Device drivers

- Hardware level
  - Device controllers
  - I/O devices
Device Addressing

- Two approaches
  - Dedicated range of device addresses in the physical memory
    - Requires special hardware instructions associated with individual devices
  - *Memory-mapped I/O*: makes no distinction between device addresses and memory addresses
    - Devices can be accessed the same way as normal memory, with the same set of hardware instructions
Device Addressing Illustrated

Primary memory

Device addresses

Separate device addresses

Primary memory

Device 1

Device 2

Memory addresses

Memory-mapped I/Os
Ways to Access a Device

- **Polling:** a CPU repeatedly checks the status of a device for exchanging data
  - Simple
  - Busy-waiting
Ways to Access a Device

**Interrupt-driven I/Os:** A device controller notifies the corresponding device driver when the device is available

- More efficient use of CPU cycles
- Data copying and movements
- Slow for character devices (i.e., one interrupt per keyboard input)
Ways to Access a Device

- **Direct memory access (DMA):** uses an additional controller to perform data movements
  - CPU is not involved in copying data
  - A process cannot access in-transit data
Ways to Access a Device

- **Double buffering:** uses two buffers. While one is being used, the other is being filled
  - Analogy: pipelining
  - Extensively used for graphics and animation
    - So a viewer does not see the line-by-line scanning
Overlapped I/O and CPU Processing

- Process A (infinite loop)
  - 67% CPU
  - 33% I/O

- Process B (infinite loop)
  - 33% CPU
  - 67% I/O

- SRTF
  - CPU
  - I/O
Disk as An Example Device

- 40-year-old storage technology
- Incredibly complicated
- A modern drive
  - 250,000 lines of micro code
Disk Characteristics

- Disk platters: coated with magnetic materials for recording
Disk Characteristics

- **Disk arm**: moves a comb of disk heads
  - Only one disk head is active for reading/writing
Hard Disk Trivia...

- Aerodynamically designed to fly
  - As close to the surface as possible
  - No room for air molecules
- Therefore, hard drives are filled with special inert gas
- If head touches the surface
  - Head crash
  - Scrapes off magnetic information
Disk Characteristics

- Each disk platter is divided into concentric tracks
A track is further divided into **sectors**. A sector is the smallest unit of disk storage.
A **cylinder** consists of all tracks with a given disk arm position.
Cylinders are further divided into **zones**
Disk Characteristics

- **Zone-bit recording**: zones near the edge of a disk store more information (higher bandwidth)

---

Diagram showing disk platters, sector, cylinder, track, disk arm, disk heads, and zones.
More About Hard Drives Than You Ever Want to Know

- **Track skew**: starting position of each track is slightly skewed
  - Minimize rotational delay when sequentially transferring bytes across tracks

- **Thermo-calibrations**: periodically performed to account for changes of disk radius due to temperature changes

- Typically 100 to 1,000 bits are inserted between sectors to account for minor inaccuracies
Disk Access Time

- **Seek time**: the time to position disk heads (~4 msec on average)
- **Rotational latency**: the time to rotate the target sector to underneath the head
  - Assume 7,200 rotations per minute (RPM)
  - $7,200 / 60 = 120$ rotations per second
  - $1/120 = \sim 8$ msec per rotation
  - Average rotational delay is ~4 msec
Disk Access Time

- **Transfer time:** the time to transfer bytes
  - Assumptions:
    - 58 Mbytes/sec
    - 4-Kbyte disk blocks
    - Time to transfer a block takes 0.07 msec

- **Disk access time**
  - Seek time + rotational delay + transfer time
Disk Performance Metrics

- **Latency**
  - Seek time + rotational delay

- **Bandwidth**
  - Bytes transferred / disk access time
Examples of Disk Access Times

- If disk blocks are randomly accessed
  - Average disk access time = \( \sim 8 \) msec
  - Assume 4-Kbyte blocks
  - \( 4 \text{ Kbyte} / 8 \text{ msec} = \sim 500 \text{ Kbyte/sec} \)

- If disk blocks of the same cylinder are randomly accessed without disk seeks
  - Average disk access time = \( \sim 4 \) msec
  - \( 4 \text{ Kbyte} / 4 \text{ msec} = \sim 1 \text{ Mbyte/sec} \)
Examples of Disk Access Times

- If disk blocks are accessed sequentially
  - Without seeks and rotational delays
  - Bandwidth: 58 Mbytes/sec

- Key to good disk performance
  - Minimize seek time and rotational latency
Disk Tradeoffs

- Larger sector size → better bandwidth
- Wasteful if only 1 byte out of 1 Mbyte is needed

<table>
<thead>
<tr>
<th>Sector size</th>
<th>Space utilization</th>
<th>Transfer rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 byte</td>
<td>8 bits/1008 bits (0.8%)</td>
<td>125 bytes/sec (1 byte / 8 msec)</td>
</tr>
<tr>
<td>4 Kbytes</td>
<td>4096 bytes/4221 bytes (97%)</td>
<td>500 Kbytes/sec (4 Kbytes / 8 msec)</td>
</tr>
<tr>
<td>1 Mbyte</td>
<td>(~100%)</td>
<td>58 Mbytes/sec (peak bandwidth)</td>
</tr>
</tbody>
</table>
Disk Controller

- Few popular standards
  - IDE (integrated device electronics)
  - ATA (advanced technology attachment interface)
  - SCSI (small computer systems interface)
  - SATA (serial ATA)

- Differences
  - Performance
  - Parallelism
Disk Device Driver

- Major goal: reduce seek time for disk accesses
  - Schedule disk request to minimize disk arm movements
Disk Arm Scheduling Policies

- **First come, first serve (FCFS):** requests are served in the order of arrival
  + Fair among requesters
  - Poor for accesses to random disk blocks
- **Shortest seek time first (SSTF):** picks the request that is closest to the current disk arm position
  + Good at reducing seeks
  - May result in starvation
Disk Arm Scheduling Policies

- **SCAN:** takes the closest request in the direction of travel (an example of elevator algorithm)
  + no starvation
  - a new request can wait for almost two full scans of the disk
Disk Arm Scheduling Policies

- **Circular SCAN (C-SCAN):** disk arm always serves requests by scanning in one direction.
  - Once the arm finishes scanning for one direction
  - Returns to the 0th track for the next round of scanning
First Come, First Serve

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
First Come, First Serve

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
- Total seek distance: $1 + 3 + 5 + 1 + 7 = 17$
Shortest Seek Distance First

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
Shortest Seek Distance First

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
- Total seek distance: $1 + 2 + 1 + 6 + 1 = 10$
- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
SCAN

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
- Total seek distance: $1 + 1 + 3 + 3 + 1 = 9$
C-SCAN

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
C-SCAN

- Request queue: 3, 6, 1, 0, 7
- Head start position: 2
- Total seek distance: $1 + 3 + 1 + 7 + 1 = 13$