CS 355
Operating Systems

I/O and Devices

I/O Devices

- Block devices – stores information in fixed-sized blocks, each with its own address
  - Disks
  - USBs
- Character devices – operates on a stream of characters, is not addressable and does not have seek operations
  - Printers
  - Network interfaces
  - Mice (pointing devices)

Typical I/O Devices and Data Rates

<table>
<thead>
<tr>
<th>Device</th>
<th>Data rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard</td>
<td>10 bytes/sec</td>
</tr>
<tr>
<td>Mouse</td>
<td>100 bytes/sec</td>
</tr>
<tr>
<td>56k modem</td>
<td>7 Kbps/sec</td>
</tr>
<tr>
<td>Scanner</td>
<td>400 Kbps/sec</td>
</tr>
<tr>
<td>Digital camcorder</td>
<td>3.5 MB/sec</td>
</tr>
<tr>
<td>802.11g Wireless</td>
<td>6.75 MB/sec</td>
</tr>
<tr>
<td>50x CD-ROM</td>
<td>7.8 MB/sec</td>
</tr>
<tr>
<td>Fast Ethernet</td>
<td>12.5 MB/sec</td>
</tr>
<tr>
<td>Compact flash card</td>
<td>40 MB/sec</td>
</tr>
<tr>
<td>FireWire (IEEE 1394)</td>
<td>50 MB/sec</td>
</tr>
<tr>
<td>USB 2.0</td>
<td>480 MB/sec</td>
</tr>
<tr>
<td>SONET OC-12 network</td>
<td>78 MB/sec</td>
</tr>
<tr>
<td>SCSI Ultra 2 disk</td>
<td>80 MB/sec</td>
</tr>
<tr>
<td>Gigabit Ethernet</td>
<td>125 MB/sec</td>
</tr>
<tr>
<td>SATA disk drive</td>
<td>300 MB/sec</td>
</tr>
<tr>
<td>Ultrasound tape</td>
<td>520 MB/sec</td>
</tr>
<tr>
<td>PCI bus</td>
<td>528 MB/sec</td>
</tr>
</tbody>
</table>

Device Controllers

- I/O devices have components:
  - mechanical component
  - electronic component
- The electronic component is the device controller
  - may be able to handle multiple devices
- Controller’s tasks
  - convert serial bit stream to block of bytes
  - perform error correction as necessary
  - make available to main memory

Communication with OS

- Registers
  - OS writes to these registers to command device to deliver/accept data, turn on/off, etc.
  - OS reads from these registers to check on device status
- Data buffers
- Communication of CPU and the control registers
  - Assign port numbers – special CPU instruction
  - Map all control registers into memory space to unique addresses – must disable caching
  - Pentium provides both
    - I/O ports numbered 0 to 64k for control registers
    - Memory address from 64k to 1M for device data buffers

I/O and Memory Space Organization

- Separate I/O and memory space
- Memory-mapped I/O
- Hybrid
**Principles of I/O Programming**

- **Device independence**
  - Programs can access any I/O device without specifying device in advance
  - (floppy, hard drive, or CD-ROM)
- **Uniform naming**
  - Name of a file or device should be a string or an integer
  - Not dependent on which machine
  - On Unix, directories such as /dev/lp, /dev/tty, or arbitrarily mounted directories
- **Error handling**
  - Handle as close to the hardware as possible

**Goals of I/O Software**

- **Synchronous vs. asynchronous transfers**
  - Blocked transfers vs. interrupt-driven
- **Buffering**
  - Data coming off a device often cannot be stored in final destination directly
- **Sharable vs. dedicated devices**
  - Disks are sharable
  - CDs/DVDs would not be

**Programmed I/O**

- **Want to send n bytes to printer**
- Printer has two memory-mapped registers
  - `status` - OS reads for printer status
  - `data` - OS writes to send data
  - We busy-wait on `status`

```c
for (i=0; i<N; i++) {
    /* wait for device to be ready */
    while (*status != READY) ;
    *data = buffer[i] ;
}
```
Interrupt-Driven I/O

- Need both driver and interrupt handler
- Driver copies a single byte into buffer and calls scheduler:
  ```c
  while (*status != READY) ;
  *data = buffer[0]; /* 1 byte */
  invoke_scheduler();
  ```
- Printer generates interrupt after printing one char. Interrupt handler does the rest:
  ```c
  if (N == 0) unblock_user();
  else (*data = buffer[i++]; N-- ; i++)
  ResetInterrupt(); /* notify device */
  ```

I/O Using DMA

- Printing a string using DMA
  - code executed when the print system call is made
  - interrupt service procedure

I/O Software Layers

- User-level I/O software
- Device-independent operating system software
- Device drivers
- Interrupt handlers
- Hardware

Device Drivers

- Device Drivers
- User interface
- Kernel interface
- Hardware
- Devices
- Device drivers: Printer driver, Keyboard driver, Disk driver

Device-Independent I/O Software

- Uniform interfacing for device drivers
- Buffering
- Error reporting
- Allocating and releasing dedicate devices
- Providing a device-independent block size

OS with and without Standard Driver Interface

- Disk driver
- Printer driver
- Keyboard driver
- Operating system
Buffering

- (a) Unbuffered input
- (b) Buffering in user space
- (c) Buffering in the kernel followed by copying to user space
- (d) Double buffering in the kernel

Networking and Many Buffers

Disks

<table>
<thead>
<tr>
<th>Parameter</th>
<th>IBM 360-KB floppy disk</th>
<th>WD 18300 hard disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of cylinders</td>
<td>40</td>
<td>10601</td>
</tr>
<tr>
<td>Tracks per cylinder</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>Sectors per track</td>
<td>9</td>
<td>281 (avg)</td>
</tr>
<tr>
<td>Sectors per disk</td>
<td>720</td>
<td>35742000</td>
</tr>
<tr>
<td>Bytes per sector</td>
<td>512</td>
<td>512</td>
</tr>
<tr>
<td>Disk capacity</td>
<td>360 KB</td>
<td>18.3 GB</td>
</tr>
<tr>
<td>Seek time (adjacent cylinders)</td>
<td>6 msec</td>
<td>0.8 msec</td>
</tr>
<tr>
<td>Seek time (average case)</td>
<td>77 msec</td>
<td>6.9 msec</td>
</tr>
<tr>
<td>Rotation time</td>
<td>200 msec</td>
<td>8.33 msec</td>
</tr>
<tr>
<td>Motor stop/start time</td>
<td>250 msec</td>
<td>20 sec</td>
</tr>
<tr>
<td>Time to transfer 1 sector</td>
<td>22 msec</td>
<td>17 msec</td>
</tr>
</tbody>
</table>

Disk Virtual Geometry

Disk Sector

Preamble | Data | ECC
Cylinder Skew

Interleaving
- No interleaving
- Single interleaving
- Double interleaving

Disk Arm Scheduling
- Physical address on disk
  - cylinder number, head number, sector number
- Time required to read or write a disk block
determined by 3 factors
  1. Seek time
  2. Rotational delay
  3. Actual transfer time
- Seek time dominates
- Error checking is done by controllers

Shortest Seek First (SSF)

The Elevator Algorithm

Error Handling
- A disk track with a bad sector
- Substituting a spare for the bad sector
- Shifting all the sectors to bypass the bad one
Read Error

- After reading a sector, what if you get an error?
  - ECC may allow recovery
  - Read again
  - Sector is bad, replace with spare
- OS may track bad sectors, or the controller can handle transparently
- Back up process has to manage bad sectors

RAID

- Redundant Array of Independent Disks
- Basic idea: spread work over several physical disks, but appear like a single disk to OS
- RAID controller manages a suite of disks to improve reliability and performance
- Striping
  - Organize logically consecutive chunks of data across different disks
  - Can be bit-level, byte-level or sector level
  - Incoming request is split into all relevant disks by RAID controller thus they can operate in parallel

RAID 0-2

- Backup and parity drives are shaded