I/O Systems

Notice: The slides for this lecture have been largely based on those accompanying the textbook Operating Systems Concepts with Java, by Silberschatz, Galvin, and Gagne (2003). Many, if not all, of the illustrations contained in this presentation come from this source.
I/O Hardware

- Incredible variety of I/O devices.
- Common concepts:
  - Port,
  - Bus (daisy chain or shared direct access),
  - Controller (host adapter).
- I/O instructions control devices.
- Devices have addresses, used by
  - Direct I/O instructions,
  - Memory-mapped I/O.
**Concepts**

**port:** a connection point between a peripheral device and the computer.

**bus:** a set of wires shared by one or more devices, which communicate with the system using a rigid protocol.

**daisy chain:** every device has two ports; either one port connects directly to the system and the other to another device, or the two ports connect to other devices. The chain usually operates as a bus.
A Typical PC Bus Structure

[Diagram showing the components of a typical PC bus structure, including monitor, processor, graphics controller, bridge/memory controller, cache, memory, IDE disk controller, expansion bus interface, keyboard, parallel port, and serial port.]
CPU and I/O Controllers

The processor transfers data to and from an I/O controller to effect I/O operations on devices.
Memory-Mapped I/O

The processor reads and writes data to address in its memory space, which are associated with the registers and control lines of I/O controllers.

![Diagram of Memory-Mapped I/O]

- CPU
- Memory bus
- RAM
- I/O port:
  - 0x000: status
  - 0x004: control
  - 0x008: data in
  - 0x00C: data out
## Device I/O Port Locations on PCs (partial)

<table>
<thead>
<tr>
<th>I/O address range (hexadecimal)</th>
<th>device</th>
</tr>
</thead>
<tbody>
<tr>
<td>000-00F</td>
<td>DMA controller</td>
</tr>
<tr>
<td>020-021</td>
<td>interrupt controller</td>
</tr>
<tr>
<td>040-043</td>
<td>timer</td>
</tr>
<tr>
<td>200-20F</td>
<td>game controller</td>
</tr>
<tr>
<td>2F8-2FF</td>
<td>serial port (secondary)</td>
</tr>
<tr>
<td>320-32F</td>
<td>hard-disk controller</td>
</tr>
<tr>
<td>378-37F</td>
<td>parallel port</td>
</tr>
<tr>
<td>3D0-3DF</td>
<td>graphics controller</td>
</tr>
<tr>
<td>3F0-3F7</td>
<td>diskette-drive controller</td>
</tr>
<tr>
<td>3F8-3FF</td>
<td>serial port (primary)</td>
</tr>
</tbody>
</table>
Polling

- Determines state of device:
  - command-ready,
  - busy,
  - error.
- Busy-wait cycle to wait for I/O from device: the CPU is involved in periodically checking the status of the operation.
Interrupts

- CPU Interrupt request line triggered by I/O device.
- Interrupt handler receives interrupts.
- Maskable to ignore or delay some interrupts.
- Interrupt vector used to dispatch interrupt to correct handler:
  - Based on priority.
  - Some unmaskable.
- Interrupt mechanism also used for exceptions.
Interrupt-Driven I/O Cycle

1. Device driver initiates I/O
2. CPU executing checks for interrupts between instructions
3. I/O controller initiates I/O
4. Input ready, output complete, or error generates interrupt signal
5. CPU receiving interrupt, transfers control to interrupt handler
6. Interrupt handler processes data, returns from interrupt
7. CPU resumes processing of interrupted task
# Intel Pentium Processor Event-Vector Table

<table>
<thead>
<tr>
<th>vector number</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>divide error</td>
</tr>
<tr>
<td>1</td>
<td>debug exception</td>
</tr>
<tr>
<td>2</td>
<td>null interrupt</td>
</tr>
<tr>
<td>3</td>
<td>breakpoint</td>
</tr>
<tr>
<td>4</td>
<td>INTO-detected overflow</td>
</tr>
<tr>
<td>5</td>
<td>bound range exception</td>
</tr>
<tr>
<td>6</td>
<td>invalid opcode</td>
</tr>
<tr>
<td>7</td>
<td>device not available</td>
</tr>
<tr>
<td>8</td>
<td>double fault</td>
</tr>
<tr>
<td>9</td>
<td>coprocessor segment overrun (reserved)</td>
</tr>
<tr>
<td>10</td>
<td>invalid task state segment</td>
</tr>
<tr>
<td>11</td>
<td>segment not present</td>
</tr>
<tr>
<td>12</td>
<td>stack fault</td>
</tr>
<tr>
<td>13</td>
<td>general protection</td>
</tr>
<tr>
<td>14</td>
<td>page fault</td>
</tr>
<tr>
<td>15</td>
<td>(Intel reserved, do not use)</td>
</tr>
<tr>
<td>16</td>
<td>floating point error</td>
</tr>
<tr>
<td>17</td>
<td>alignment check</td>
</tr>
<tr>
<td>18</td>
<td>machine check</td>
</tr>
<tr>
<td>19231</td>
<td>(Intel reserved, do not use)</td>
</tr>
<tr>
<td>32D255</td>
<td>maskable interrupts</td>
</tr>
</tbody>
</table>
Direct Memory Access (DMA)

- Used to avoid programmed I/O for large data movement.

- Requires DMA controller.

- The controller allows for data to be transferred directly between I/O device and memory without CPU intervention.
DMA Transfer

1. Device driver is told to transfer disk data to buffer at address X
2. Device driver tells disk controller to transfer C bytes from disk to buffer at address X
3. Disk controller initiates DMA transfer
4. Disk controller sends each byte to DMA controller
5. Disk controller transfers bytes to buffer X, increasing memory address and decreasing C until C = 0
6. When C = 0, DMA interrupts CPU to signal transfer completion
Application I/O Interface

- I/O system calls encapsulate device behaviors in generic classes.
- Device-driver layer hides differences among I/O controllers from kernel.
- Devices vary in many dimensions:
  - Character-stream or block.
  - Sequential or random-access.
  - Sharable or dedicated.
  - Speed of operation.
  - Read-write, read only, or write only.
A Kernel I/O Structure

- Kernel
- Kernel I/O Subsystem
  - SCSI device driver
  - Keyboard device driver
  - Mouse device driver
  - PCI bus device driver
  - Floppy device driver
  - ATAPI device driver

- Hardware
  - SCSI devices
  - Keyboard
  - Mouse
  - PCI bus
  - Floppy disk drives
  - ATAPI devices (disks, tapes, drives)
## Characteristics of I/O Devices

<table>
<thead>
<tr>
<th>aspect</th>
<th>variation</th>
<th>example</th>
</tr>
</thead>
<tbody>
<tr>
<td>data-transfer mode</td>
<td>character block</td>
<td>terminal disk</td>
</tr>
<tr>
<td>access method</td>
<td>sequential</td>
<td>modern CD-ROM</td>
</tr>
<tr>
<td>transfer schedule</td>
<td>synchronous</td>
<td>tape keyboard</td>
</tr>
<tr>
<td>sharing</td>
<td>dedicated</td>
<td>tape keyboard</td>
</tr>
<tr>
<td>device speed</td>
<td>latency</td>
<td></td>
</tr>
<tr>
<td></td>
<td>seek time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>transfer rate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>delay between operations</td>
<td></td>
</tr>
<tr>
<td>I/O direction</td>
<td>read only</td>
<td>CD-ROM graphics controller disk</td>
</tr>
<tr>
<td></td>
<td>write only</td>
<td></td>
</tr>
<tr>
<td></td>
<td>read+write</td>
<td></td>
</tr>
</tbody>
</table>
Block and Character Devices

- Block devices include disk drives.
  - Commands include read, write, seek.
  - Raw I/O or file-system access.
  - Memory-mapped file access possible.

- Character devices include keyboards, mice, serial ports.
  - Commands include get, put.
  - Libraries layered on top allow line editing.
Network Devices

- Different enough from block and character to have their own interface.

- Unix and Windows NT/9x/2000 include socket interface:
  - Separates network protocol from network operation.
  - Includes `select` functionality.

- Approaches vary widely (pipes, FIFOs, streams, queues, mailboxes).
Clocks and Timers

- Provide current time, elapsed time, timer.
- If programmable interval time used for timings, periodic interrupts.
- `ioctl` (on UNIX) covers odd aspects of I/O such as clocks and timers.
Blocking and Nonblocking I/O

- **Blocking** - process suspended until I/O completed.
  - Easy to use and understand.
  - Insufficient for some needs.

- **Nonblocking** - I/O call returns as much as available.
  - User interface, data copy (buffered I/O).
  - Implemented via multi-threading.
  - Returns quickly with count of bytes read or written.

- **Asynchronous** - process runs while I/O executes.
  - Difficult to use.
  - I/O subsystem signals process when I/O completed.