COMPUTER
ARCHITECTURE

INPUT/OUTPUT
DEVICES

Dr. Bill Yi
Santa Clara University

(Based on text: David A. Patterson & John L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, 3rd Ed., Morgan Kaufmann, 2007)
(Also based on presentation: Dr. Nam Ling, COEN210 Lecture Notes)
COURSE CONTENTS

- Introduction
- Instructions
- Computer Arithmetic
- Performance
- Processor: Datapth
- Processor: Control
- Pipelining Techniques
- Memory

⇒ Input/Output Devices, Networks, Buses
INPUT/OUTPUT DEVICES & NETWORKS

- Input/Output Devices
- Networks
- Buses
- Interfacing
I/O Devices

- Very diverse devices
- Different requirements: response time, throughput
- In I/O & networks, base 10 is used (e.g. 1 MB/sec = 10^6 B/sec) (Note: in memory base 2 is used (e.g. 1 MB = 2^{20} B)

<table>
<thead>
<tr>
<th>Device</th>
<th>Behavior</th>
<th>Partner</th>
<th>Data rate (Mbit/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard</td>
<td>input</td>
<td>human</td>
<td>0.0001</td>
</tr>
<tr>
<td>Mouse</td>
<td>input</td>
<td>human</td>
<td>0.0038</td>
</tr>
<tr>
<td>Voice input</td>
<td>input</td>
<td>human</td>
<td>0.264</td>
</tr>
<tr>
<td>Sound input</td>
<td>input</td>
<td>machine</td>
<td>3</td>
</tr>
<tr>
<td>Scanner</td>
<td>input</td>
<td>human</td>
<td>3.2</td>
</tr>
<tr>
<td>Voice output</td>
<td>output</td>
<td>human</td>
<td>0.264</td>
</tr>
<tr>
<td>Sound output</td>
<td>output</td>
<td>human</td>
<td>8</td>
</tr>
<tr>
<td>Laser printer</td>
<td>output</td>
<td>human</td>
<td>3.2</td>
</tr>
<tr>
<td>Graphics display</td>
<td>output</td>
<td>human</td>
<td>800-8000</td>
</tr>
<tr>
<td>Modem</td>
<td>input or output</td>
<td>machine</td>
<td>0.016-0.064</td>
</tr>
<tr>
<td>Network/LAN</td>
<td>input or output</td>
<td>machine</td>
<td>100-1000</td>
</tr>
<tr>
<td>Network/wireless LAN</td>
<td>input or output</td>
<td>machine</td>
<td>11-54</td>
</tr>
<tr>
<td>Optical disk</td>
<td>storage</td>
<td>machine</td>
<td>80</td>
</tr>
<tr>
<td>Magnetic tape</td>
<td>storage</td>
<td>machine</td>
<td>32</td>
</tr>
<tr>
<td>Magnetic disk</td>
<td>storage</td>
<td>machine</td>
<td>240-2560</td>
</tr>
</tbody>
</table>
I/O Example: Disk Drives

- A magnetic disk consists of a collection of platters, each has 2 recordable surfaces
- It relies on rotating platter coated with a magnetic surface and use a removable read/write head to access the disk; disk storage is non-volatile
- Platter (rotated at 5400 to 15000 RPM) --> tracks --> sectors

To access data:
- **seek**: position head over the proper track
- **rotational latency**: wait for desired sector to rotate under the head
- **transfer**: grab the data
- **Disk controller**: handles data transfer between disk & memory

RAID – redundant arrays of inexpensive disks
(to increase performance and reliability)
Networks

- Networks: provide communications between computers
  - 0.01 to 10,000 km
  - 0.001 MB/sec to 1000 MB/sec
  - different topologies, can be point-to-point or shared
- **Examples**: Internetworking, long-haul networks, local area networks (LANs), wireless LANs (WiFi – IEEE 802.11)
- See courses in networking
**Buses**

- **Bus**: a shared communication link (one or more wires) to connect multiple subsystems
- Difficult design: may be bottleneck, length of the bus, number of devices, tradeoffs, support for many different devices, cost
- A bus generally contains:
  - a set of **control** lines (e.g. signal request, acknowledgement)
  - a set of **address / data** lines
- I/O Design affected by many factors (expandability, resilience)
- Performance:
  - Data bus width
  - Separate vs multiplexed address & data lines
  - Block transfers
- Centralized control vs Bus arbitration
Advantages of Buses

- Versatility
  - New devices can be added easily
  - Peripheral devices can be moved between computer systems that use the same bus standard

- Low cost:
  - A single set of wires is shared in multiple ways

- Provides a way to manage the complexity of design
  - Device only has to implement the bus standard
Disadvantage of Buses

- It creates a communication bottleneck
  - Bus bandwidth can limit the maximum I/O throughput
- The maximum bus speed is largely limited by:
  - The length of the bus
  - The number of devices on the bus
  - The need to support a range of devices with
    - Widely varying latencies
    - Widely varying data transfer rates
The General Organization of a Bus

- Control lines
  - Signal requests and acknowledgments
  - Indicate what type of information is on the data lines, if there are errors, etc.

- Data lines carry information between the source and the destination
  - Data
  - Addresses
  - Complex commands
Types of Buses

- **Processor-memory buses** (short, high speed, custom design, maximizing CPU-memory bandwidth)

- **I/O buses** (lengthy, different devices, standardized, wide range of bandwidth). I/O buses do not interface directly to memory, but use a processor-mem or backplane bus to mem. Examples: Firewire (1394), USB 2.0.

- **Backplane buses** (high speed, often standardized). To allow CPU, memory, I/O to coexist on single bus, balance CPU-memory & device-memory communications

- Example:
Synchronous vs. Asynchronous

- **Synchronous**: use a clock and a synchronous protocol, fast and small, but every device must operate at same rate and clock skew requires the bus to be short; e.g. processor-memory bus
- **Asynchronous**: do not use a clock and instead use handshaking; scale better and support a variety of device response speeds; e.g. I/O bus. Handshaking example: to read a word from memory and receive it in an I/O device

1. I/O raises ReadReq & sends address on data bus.
2. Memory raises Ack and I/O releases ReadReq & data lines.
3. Memory drops Ack.
4. Memory has data ready, places data on data lines and raises DataRdy.
5. I/O reads data and raises Ack.
6. Memory drops DataRdy and releases data lines.
7. I/O drops Ack and transmission completes.
I/O Handling in Operating Systems

- Buses provide electrical interconnect among I/O, CPU, & memory, and also define the lowest-level protocol for communication
- Operating system handles the higher-level I/O protocol:
  - **Polling**: periodically checking the status bits (waste a lot of processor time)
  - **Interrupts**: an I/O device interrupts the processor when it needs processor’s attention
  - **Direct Memory Access (DMA)**: processor sets up DMA, DMA controller takes over to transfer data directly between memory & I/O device without involving the processor; DMA controller interrupts the processor when I/O transfer completes (higher cost)
- Polling & interrupts work best with low-bandwidth devices, DMA works best with high-bandwidth devices (like hard disks)
- CPU gives commands to I/O devices: by memory-mapped I/O or I/O instructions
- I/O system design: latency constraints and bandwidth constraints
Chapter Summary

- I/O devices
- Disk drivers
- Networks
- Buses connecting multiple subsystems
- I/O handling in operating system
Concluding Remarks

“Acceptance of hardware ideas requires acceptance by software people; therefore hardware people should learn about software. And if software people want good machines, they must learn more about hardware to be able to communicate with and thereby influence hardware engineers.”