Interfacing Processors and Peripherals

Chapters 1 and 11 of [OS4e], Chapters 2 and 13 of [OSC], Chapter 8 of [PH3e]:

- I/O Devices
- System Interconnection
- I/O Modules
- Hardware/Software Interface
  - Giving Commands to I/O Devices
  - Communicating with the Processor
  - Transferring Data between Device and Memory

I/O Devices

Input/output (I/O) devices enable processors

- to interact with people
e.g., mouse, keyboard, monitor
- to interact with each other
e.g., modem, LAN
- to store and retrieve data
e.g., magnetic disk, magnetic tape

Data rate

- peak rate at which data can be transferred between I/O device and main memory (or processor)

<table>
<thead>
<tr>
<th>Device</th>
<th>Behavior</th>
<th>Partner</th>
<th>Data rate (KB/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard</td>
<td>input</td>
<td>human</td>
<td>0.01</td>
</tr>
<tr>
<td>Mouse</td>
<td>input</td>
<td>human</td>
<td>0.02</td>
</tr>
<tr>
<td>Voice input</td>
<td>input</td>
<td>human</td>
<td>0.02</td>
</tr>
<tr>
<td>Scanner</td>
<td>input</td>
<td>human</td>
<td>400.00</td>
</tr>
<tr>
<td>Voice output</td>
<td>output</td>
<td>human</td>
<td>0.80</td>
</tr>
<tr>
<td>Line printer</td>
<td>output</td>
<td>human</td>
<td>1.00</td>
</tr>
<tr>
<td>Laser printer</td>
<td>output</td>
<td>human</td>
<td>250.00</td>
</tr>
<tr>
<td>Graphics display</td>
<td>output</td>
<td>human</td>
<td>50,000.00</td>
</tr>
<tr>
<td>Modem</td>
<td>input or output</td>
<td>machine</td>
<td>2.00-4.00</td>
</tr>
<tr>
<td>Network/LAN</td>
<td>input or output</td>
<td>machine</td>
<td>500.00-8000.00</td>
</tr>
<tr>
<td>Floppy disk</td>
<td>storage</td>
<td>machine</td>
<td>100.00</td>
</tr>
<tr>
<td>Optical disk</td>
<td>storage</td>
<td>machine</td>
<td>1000.00</td>
</tr>
<tr>
<td>Magnetic tape</td>
<td>storage</td>
<td>machine</td>
<td>3000.00</td>
</tr>
<tr>
<td>Magnetic disk</td>
<td>storage</td>
<td>machine</td>
<td>2000.00-10,000.00</td>
</tr>
</tbody>
</table>

Transfer rates for modems and networks are usually specified in decimal

e.g., 57.6 K bits/sec = 57600 bits/sec = 7200 Bytes/sec
= 7.03 K Bytes/sec
**System Interconnection**

*bus*
- shared communication link
  - uses one set of wires to connect multiple devices

**Buses**
- provide electronic interconnect among I/O devices, processors and memory
- define lowest level protocol for communication
  - how word or block of data should be communicated on wires

**Example** Processor, memory and I/O devices might all plug into backplane bus (e.g., VME bus)
- balances demands of processor-memory communication and demands of I/O device-memory communication

---

**Bus performance and organisation:**
- speed of bus can be severely limited for electrical reasons
  - long wires connecting large numbers of devices
- use a hierarchy of buses
  - one bus may be tapped into by other (lower-level) ones
  - lower-level buses allow for expansion
  - e.g., processor-memory bus (designed to be fast) with I/O buses (such as SCSI) tapping into it
I/O Modules

* I/O module (controller)
  - hardware (electronics) that controls a device or a bus
    e.g., disk controller, SCSI-bus controller
  - adapts device interface to bus protocol
  - smooths out differences in speed
    - contains internal buffers for temporarily holding data until it can be sent on

You can find out much more than you will ever want to know about Personal Computers at [http://www.pcguide.com](http://www.pcguide.com). Relevant to this lecture are the sections on System Buses ([ref/mbsys/buses/](http://www.pcguide.com)) and External Processor Interfaces and Operation ([ref/cpu/arch/ext.htm](http://www.pcguide.com)).

Hardware/Software Interface

Operating system has major role because
- multiple processes need to access I/O devices
- interrupts are often used to communicate information about I/O operations
- low-level control of I/O devices is complex
OS must therefore provide several functions:

- **protection**
  - i.e., control access of users to I/O devices
  - e.g., reading a file on disk requires permission
- **abstraction**
  - routines that handle low-level device operations
- **interrupt-handling**
- **scheduling**
  - equitable access to shared I/O resources
  - schedule accesses to enhance system performance

Three types of communication are required in order to perform these functions:

- OS must be able to give commands to I/O devices
  - e.g., read, write, disk seek
- I/O device must be able to notify OS of successful completion or error
  - e.g., disk has completed seek
- Data must be transferred between memory and I/O device
  - e.g., block read from disk must be written to memory

**Giving Commands to I/O Devices**

- OS must be able to address I/O device and supply one or more command words

Two methods are used to address device:

- **memory-mapped I/O**
  - portions of memory address space assigned to I/O devices, accessible only in supervisor mode
  - data written to those addresses interpreted as commands
- **dedicated I/O instructions**
  - e.g., Intel 80x86, IBM 370
  - illegal to execute when not in supervisor mode
Communicating with the Processor

- programmed I/O (polling)
  - I/O device (e.g., mouse) puts information in “status register”
  - processor periodically checks register (which can waste lot of time because processors are much faster than I/O devices)
- interrupt-driven I/O
  - I/O device causes processor to be interrupted whenever it has completed operation or needs attention
  - identity of device, etc., determined by “vectored interrupt” or “cause register”

Transferring Data between Device and Memory

Using programmed I/O or interrupt-driven I/O:

- processor (and OS) do all the work, accessing I/O device and memory for each data item transferred
  - cheap solution for low-bandwidth devices

Using direct memory access (DMA):

- specialised controller transfers data between I/O device and memory, independent of processor

1. Processor sets up DMA by supplying identity of device, operation to perform on it, memory address that is source or destination of data, and number of bytes to transfer
2. DMA starts operation on device and arbitrates for the bus
  - data transfer can begin once DMA controller has become bus master and data is available
3. Upon completion of DMA transfer, controller interrupts processor
- multiple DMA controllers are allowed
e.g., one per I/O-bus controller
- processor trying to access memory is *stalled*
  whilst DMA controller is bus master
- when virtual memory is used, either DMA transfers should be constrained to stay within one page or controller needs to know relevant page table entries
- when cache memory is used, I/O activity must be routed through cache or OS must selectively flush cache
  – latter is cheaper and avoids contention between DMA and processor on cache hit

![Diagram](image-url)

*Figure 1.19 Three Techniques for Input of a Block of Data*