Input-output

• I/O is very much architecture/system dependent
• I/O requires cooperation between
  – processor that issues I/O command (read, write etc.)
  – buses that provide the interconnection between processor, memory
    and I/O devices
  – I/O controllers that handle the specifics of control of each device
    and interfacing
  – devices that store data or signal events

Basic (simplified) I/O architecture

Types of I/O devices

• Input devices
  – keyboard, mouse
• Output devices
  – screen, line printer
• Devices for both input and output
  – disks, network interfaces

An important I/O device: the disk

Secondary memory (disks)

• Physical characteristics
  – Platters (1 to 20) with diameters from 1.3 to 8 inches (recording on both sides)
  – Tracks (1,000 to 10,000)
  – Cylinders (all the tracks in the same position in the platters)
  – Sectors (e.g., 128-256 sectors/track with gaps and info related to sectors between them, typical sector 512 bytes)
  – Current trend: constant bit density, i.e., more info (sectors) on outer tracks

Example: IBM Ultrastar 146Z10

• Disk for server
  – 146 GB
  – 8 MB cache
  – 10,000 RPM
  – 3 ms average latency
  – Up to 6 platters, Up to 12 heads
  – Average seek latency 4.7 ms
  – Sustained transfer rate 33-66 MB/s
Disk access time

- Arm(s) with a reading/writing head
- Four components in an access:
  - Seek time (to move the arm on the right cylinder). From 0 (if arm already positioned) to a maximum of 15-20 ms. Not a linear function. Smaller disks have smaller seek times.
  - Ultrastar example: Average seek time = 4.7 ms.
  - My guess: track to track 0.5 ms; longest (innermost track to outermost track) 8 ms
  - Rotation time (on the average 1/2 rotation). At 3600 RPM, 8.3 ms.
- Current disks are 3600 (rarely now) or 5400 or 7200 or 10,000 (e.g., the Ultrastar, hence average is 3 ms) or even 15000 RPM
- Ultrastar example: Average seek time = 4.7 ms;
- Transfer time depends on rotation time, amount to transfer (minimal size a sector), recording density, disk/memory connection. Today, transfer time occurs at 10 to 100 MB/second
- Disk controller time: Overhead to perform an access (of the order of 1 ms)
- But ... many disk controllers have a cache that contains recently accessed sectors. If the I/O requests hits in the cache, the only components of access time are disk controller time and transfer time (which is then of the order of 40-100 MB/sec). Cache is also used to prefetch on read.

Improvements in disks

- Capacity (via density). Same growth rate as DRAMs
- Price decrease has followed (today $2-$10/GB?)
- Access times have decreased but not enormously
  - Higher density -> smaller drives -> smaller seek time
  - RPM has increased 3600 up to 15,000
- Transfer time has improved
- CPU speed - DRAM access is one “memory wall”
- DRAM access time - Disk access time is a “memory gap”
  - Technologies to fill the gap have not entirely succeeded (currently the most promising is more DRAM backed up by batteries and Flash memory to supersede disk cache)

Buses

- Simplest interconnect
  - Low cost: set of shared wires
  - Easy to add devices (although variety of devices might make the design more complex or less efficient — longer bus and more electrical load, hence the distinction between I/O buses and CPU/memory buses)
  - But bus is a single shared resource so can get saturated (both physically because of electrical load, and performance-wise because of contention to access it)
- Key parameters:
  - Width (number of lines: data, addresses, control)
  - Speed (limited by length and electrical load)

Connecting CPU, Memory and I/O

Memory and I/O buses

- CPU/memory bus: tailored to the particular CPU
  - Fast (separate address and data lines; of course separate control lines)
  - Often short and hence synchronous (governed by a clock)
  - Wide (64-128 and even 256 bits)
- I/O bus: follows some standard so many types of devices can be hooked on to it
  - Asynchronous (hand-shaking protocol)
  - Narrower
Bus transactions

- Consists of arbitration and commands
  - Arbitration: who is getting control of the bus
    - Commands: type of transaction (read, write, ack, etc...)
  - Read, Write, Atomic Read-Modify-Write (atomic swap)
    - Read: send address and data is returned
    - Write: send address and data
    - Read-Modify-write: keep bus during the whole transaction. Used for synchronization between processes

Bus arbitration

- Arbitration: who gets the bus if several requests occur at the same time
  - Only one master (processor): centralized arbitration
  - Multiple masters (most common case): centralized arbitration (FIFO, daisy-chain, round-robin, combination of those) vs. decentralized arbitration (each device knows its own priority)
- Communication protocol between master and slave
  - Synchronous (for short buses - no clock skew - i.e. CPU/memory)
  - Asynchronous (hand-shaking finite-state machine; easier to accommodate many devices)