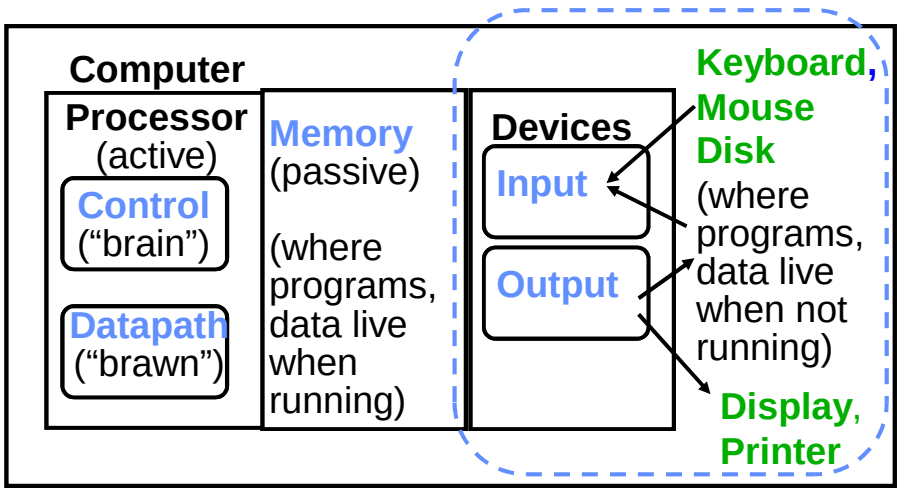
Chapter 6

Storage & Other I/O

5 components of a Computer



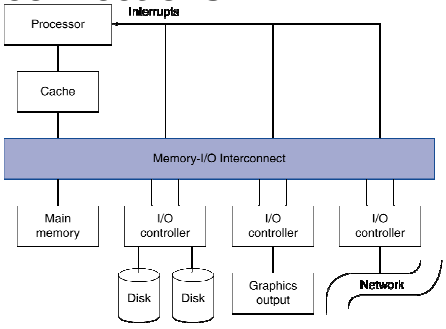
6.1 Motivation for Input/Output

- I/O is how humans interact with computers
- I/O gives computers long-term memory.
- I/O lets computers do amazing things.

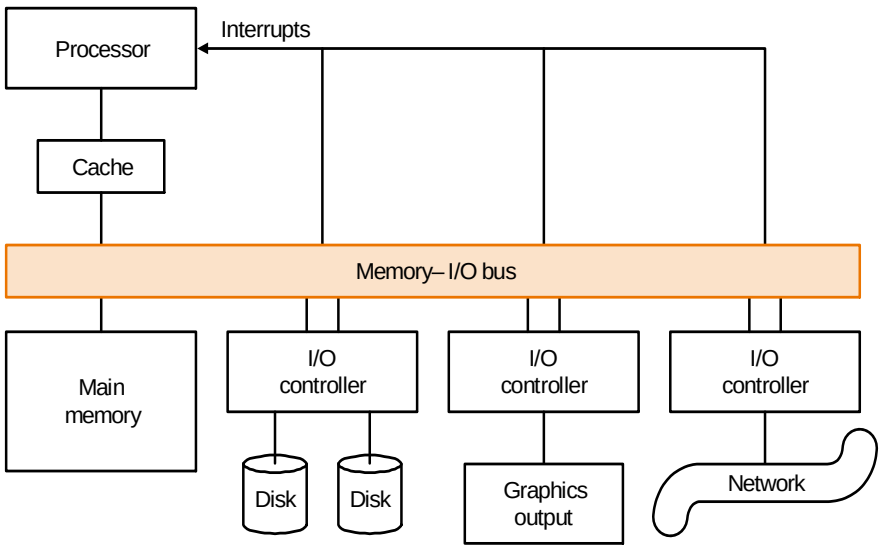
- Computer without I/O like a car without wheels; great technology, but won't get you anywhere

Introduction

- I/O devices can be characterized by
  - Behaviour: input, output, storage
  - Partner: human or machine
  - Data rate: bytes/sec, transfers/sec
- I/O bus connections



Interfacing Processors and Peripherals



I/O System Characteristics

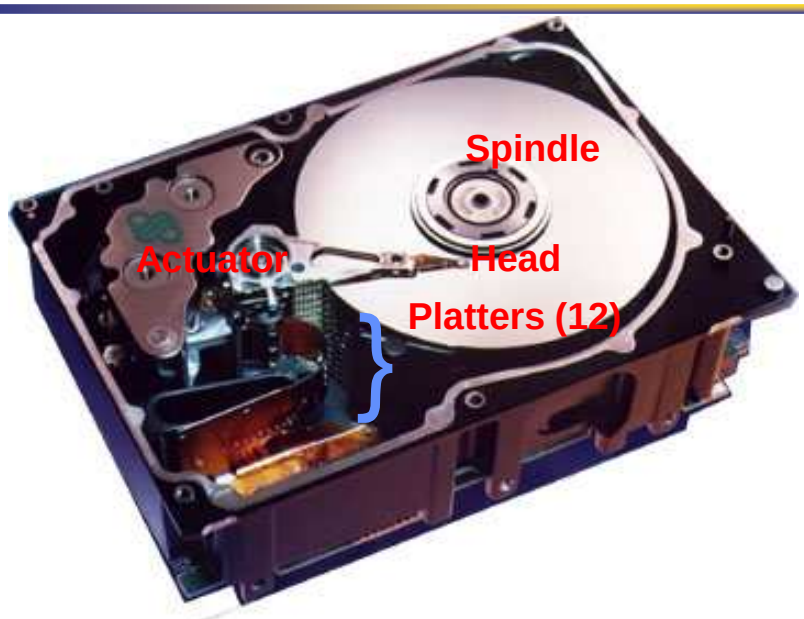
- Dependability is important
  - Particularly for storage devices
- Performance measures
  - Latency (response time)
  - Throughput (bandwidth)
  - Desktops & embedded systems
    - Mainly interested in response time & diversity of devices
  - Servers
    - Mainly interested in throughput & expandability of devices

I/O Devices - Very diverse

- I/O Speed: bytes transferred per second
  (from mouse to display: million-to-1)

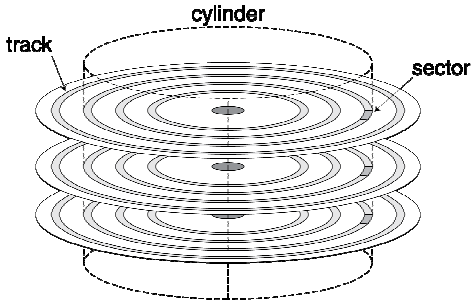
Device	Behavior	Partner	Data Rate (KBytes/s)
Keyboard	Input	Human	0.01
Mouse	Input	Human	0.02
Voice output	Output	Human	5.00
Floppy disk	Storage	Machine	50.00
Laser Printer	Output	Human	100.00
Magnetic Disk	Storage	Machine	10,000.00
Network-LAN	I or O	Machine	10,000.00
Graphics Display	Output	Human	30,000.00

6.3 Disk Storage



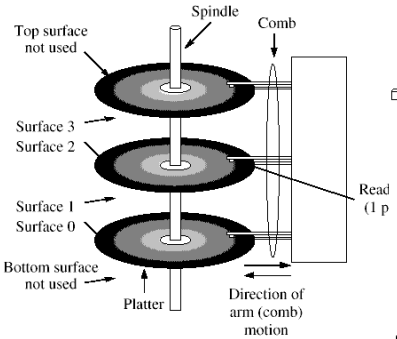
6.3 Disk Storage

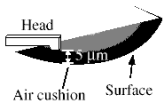
- **Nonvolatile, rotating magnetic storage**

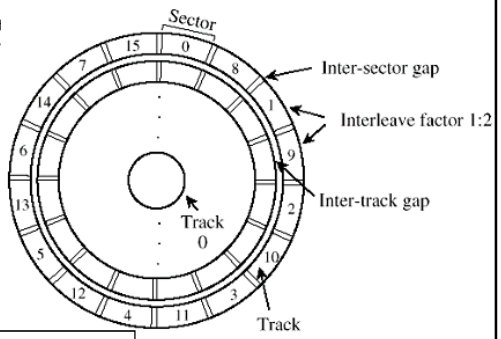


Disk Drives

A Magnetic Disk with Three Platters



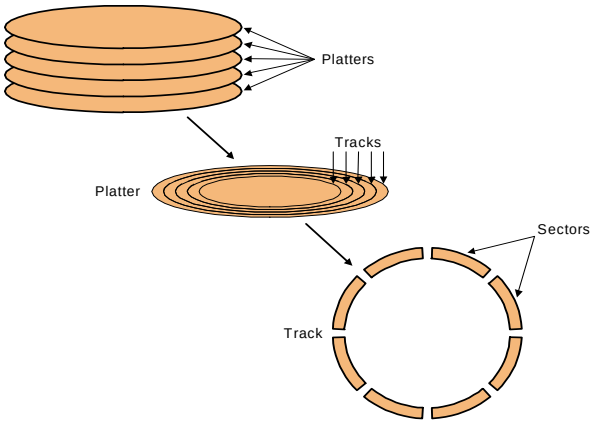




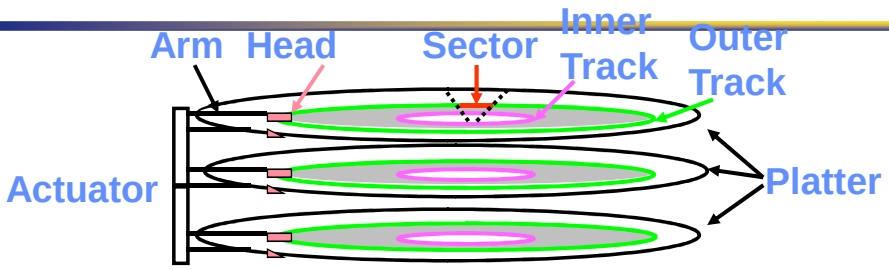
Sector org.

10 bytes header	512 Bytes data	12 bytes ECC

Disk Drives



Disk Device Terminology



- Several platters, with information recorded magnetically on both surfaces (usually)
- Bits recorded in tracks, which in turn divided into sectors (e.g., 512 Bytes); error correction code per sector to find and correct errors
- Actuator moves head (end of arm) over track ("seek"), wait for sector rotate under head, then read or write

Disk Sectors and Access

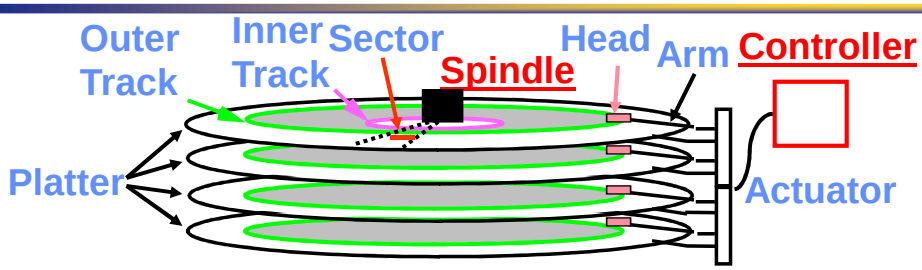
- Each sector records
  - Sector ID
  - Data (512 bytes, 4096 bytes proposed)
  - Error correcting code (ECC)
    - Used to hide defects and recording errors
  - Synchronization fields and gaps
- Access to a sector involves
  - Queuing delay if other accesses are pending
  - Seek: move the heads
  - Rotational latency
  - Data transfer, Controller overhead

Disk drive capacities

- A high-capacity disk drive may have 512 bytes per sector, 1,000 sectors per track, 5,000 tracks per surface, and 8 platters. The total capacity of this drive is

  C = 512 bytes/sector* 1000 sectors/track * 5000 tracks/surface * 8 platters *2surfaces/platter = 38 GB.

Disk Device Performance



- Disk Latency = Seek Time + Rotation Time + Transfer Time + Controller Overhead
  - –Seek Time depends no. tracks move arm, seek speed of disk
  - –Rotation Time depends on speed disk rotates, how far sector is from head
  - –Transfer Time depends on data rate (bandwidth) of disk (bit density), size of request

Disk drive speeds

- To access data:
  - — seek: position head over the proper track (8 to 20 ms. avg.)
  - — rotational latency: wait for desired sector (.5 / RPM)
  - — transfer: grab the data (one or more sectors) 2 to 15 MB/sec
  - – Controller time: overhead the controller imposes in I/O access.
- Disk Read Time = Average seek time + average rotational delay + transfer time + controller overhead
- What is the average time to read or write a 512-byte sector for a typical disk rotating at 5400 RPM? The average seek time is 12 ms, the transfer rate 5MB/sec, and the controller overhead is 2ms.
- Rotational delay = 0.5 rotation/5400RMP = 5.6 ms
- 12ms + 5.6ms + 0.5KB/5MB/sec + 2ms = 19.7 ms

Another Disk Access Example

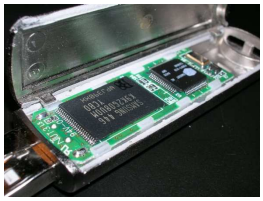
- Given
  - 512B sector, 15,000rpm, 4ms average seek time, 100MB/s transfer rate, 0.2ms controller overhead, idle disk
- Average read time
  - 4ms seek time
    + ½ / (15,000/60) = 2ms rotational latency
    + 512 / 100MB/s = 0.005ms transfer time
    + 0.2ms controller delay
    = 6.2ms
- If actual average seek time is 1ms
  - Average read time = 3.2ms

Disk Performance Issues

- Manufacturers quote average seek time
  - Based on all possible seeks
  - Locality and OS scheduling lead to smaller actual average seek times
- Smart disk controller allocate physical sectors on disk
  - Present logical sector interface to host
  - SCSI, ATA, SATA
- Disk drives include caches
  - Prefetch sectors in anticipation of access
  - Avoid seek and rotational delay

6.4 Flash Storage

- Nonvolatile semiconductor storage
  - is a type of EEPROM chip (Electronically Erasable Programmable Read Only Memory).
  - 100× – 1000× faster than disk
  - Smaller, lower power, more robust
  - But more $/GB (between disk and DRAM)



Example Use

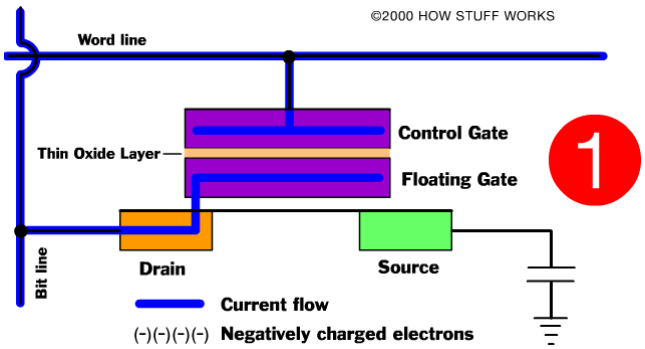
- Your computer's BIOS chip
- CompactFlash (most often found in digital cameras)
- SmartMedia (most often found in digital cameras)
- Memory Stick (most often found in digital cameras)
- PCMCIA Type I and Type II memory cards (used as solid-state disks in laptops)
- Memory cards for video game consoles, Cellular phone, Video/Music player
- Computer replacing hard drive?

Characteristics

Characteristics	Kingston SecureDigital (SD) SD4/8 GB	Transend Type I CompactFlash TS16GCF133	RiDATA Solid State Disk 2.5 inch SATA
Formatted data capacity (GB)	8	16	32
Bytes per sector	512	512	512
Data transfer rate (read/write MB/sec)	4	20/18	68/50
Power operating/standby (W)	0.66/0.15	0.66/0.15	2.1/—
Size: height × width × depth (inches)	0.94 × 1.26 × 0.08	1.43 × 1.68 × 0.13	0.35 × 2.75 × 4.00
Weight in grams (454 grams/pound)	2.5	11.4	52
Mean time between failures (hours)	> 1,000,000	> 1,000,000	> 4,000,000
GB/cu. in., GB/watt	84 GB/cu.in., 12 GB/W	51 GB/cu.in., 24 GB/W	8 GB/cu.in., 16 GB/W
Best price (2008)	~ $30	~ $70	~ $300

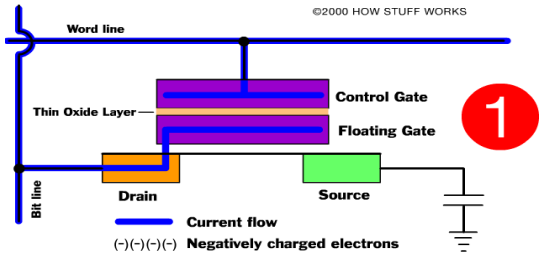
A Flash Memory Cell

- A cell has two transistors (Control and Floating Gates) at each intersection. The floating gate's only link to the row, or wordline, is through the control gate.
- As long as this link is in place, the cell has a value of 1. To change the value to a 0 requires a curious process called Fowler-Nordheim tunneling



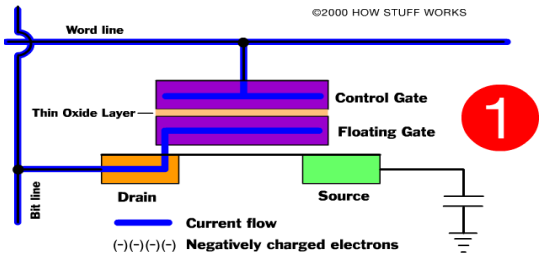
How Flash Memory works?

- Tunneling is used to alter the placement of electrons in the floating gate. An electrical charge, is applied to the floating gate. The charge comes from the column, or bitline, enters the floating gate and drains to a ground. This charge causes the floating-gate transistor to act like an electron gun. The excited electrons are pushed through and trapped on other side of the thin oxide layer, giving it a negative charge.



How Flash Memory works?

- A special device called a cell sensor monitors the level of the charge passing through the floating gate. If the flow through the gate is above the 50 percent threshold, it has a value of 1. When the charge passing through drops below the 50-percent threshold, the value changes to 0. A blank EEPROM has all of the gates fully open, giving each cell a value of 1.



Flash Types

- NOR flash: bit cell like a NOR gate
  - Random read/write access
  - Used for instruction memory in embedded systems
- NAND flash: bit cell like a NAND gate
  - Denser (bits/area), but block-at-a-time access
  - Cheaper per GB
  - Used for USB keys, media storage, …
- Flash bits wears out after 1000's of accesses
  - Not suitable for direct RAM or disk replacement
  - Wear leveling: remap data to less used blocks

Characteristics of NOR and NAND

Characteristics	NOR Flash Memory	NAND Flash Memory
Typical use	BIOS memory	USB key
Minimum access size (bytes)	512 bytes	2048 bytes
Read time (microseconds)	0.08	25
Write time (microseconds)	10.00	1500 to erase + 250
Read bandwidth (MBytes/second)	10	40
Write bandwidth (MBytes/second)	0.4	8
Wearout (writes per cell)	100,000	10,000 to 100,000
Best price/GB (2008)	$65	$4

6.5 Interconnecting Components

- Need interconnections between
  - CPU, memory, I/O controllers
- Bus: shared communication channel
  - Parallel set of wires for data and synchronization of data transfer
  - Can become a bottleneck
- Performance limited by physical factors
  - Wire length, number of connections
- More recent alternative: high-speed serial connections with switches
  - Like networks

Bus Types

- Processor-Memory buses
  - Short, high speed
  - Design is matched to memory organization
- I/O buses
  - Longer, allowing multiple connections
  - Specified by standards for interoperability
  - Connect to processor-memory bus through a bridge

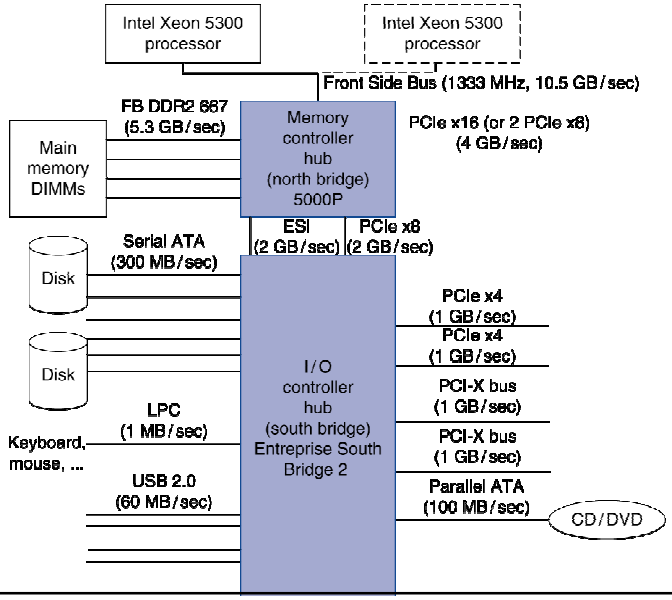
Bus Signals and Synchronization

- **Data lines**
  - **Carry address and data**
  - **Multiplexed or separate**
- **Control lines**
  - **Indicate data type, synchronize transactions**
- **Synchronous**
  - **Uses a bus clock**
- **Asynchronous**
  - **Uses request/acknowledge control lines for handshaking**

I/O Bus Examples

	Firewire	USB 2.0	PCI Express	Serial ATA	Serial Attached SCSI
Intended use	External	External	Internal	Internal	External
Devices per channel	63	127	1	1	4
Data width	4	2	2/lane	4	4
Peak bandwidth	50MB/s or 100MB/s	0.2MB/s, 1.5MB/s, or 60MB/s	250MB/s/lane 1×, 2×, 4×, 8×, 16×, 32×	300MB/s	300MB/s
Hot pluggable	Yes	Yes	Depends	Yes	Yes
Max length	4.5m	5m	0.5m	1m	8m
Standard	IEEE 1394	USB Implementers Forum	PCI-SIG	SATA-IO	INCITS TC T10

Typical x86 PC I/O System



I/O Management

- I/O is mediated by the OS
  - Multiple programs share I/O resources
    - Need protection and scheduling
  - I/O causes asynchronous interrupts
    - Same mechanism as exceptions
  - I/O programming is fiddly
    - OS provides abstractions to programs

6.6 Interfacing I/O to Memory, CPU

1. **Polling**
   - Check each I/O device in turn, schedule I/O work on appropriate idle devices
   - *Ask for device status* (busy, wait, idle, dead…)
2. **Interrupt-Driven**
   - On-demand request of I/O services
   - *Needs queue to save waiting I/O requests*
3. **Direct Memory Access (DMA)**
   - Direct I/O Device to Memory Transfer
   - *Very fast, used for large amounts of data* (e.g., video, imagery, audio)

Polling

- Periodically check I/O status register
  - If device ready, do operation
  - If error, take action
- Common in small or low-performance real-time embedded systems
  - Predictable timing
  - Low hardware cost
- In other systems, wastes CPU time

Interrupts

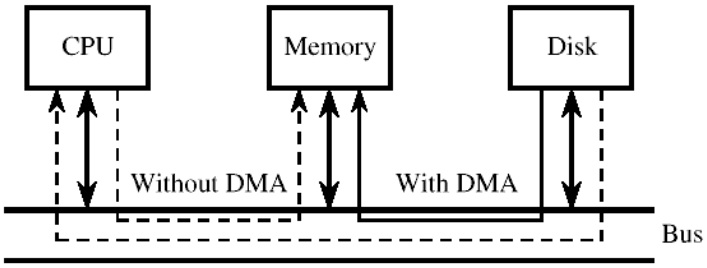
- When a device is ready or error occurs
  - Controller interrupts CPU
- Interrupt is like an exception
  - But not synchronized to instruction execution
  - Can invoke handler between instructions
  - Cause information often identifies the interrupting device
- Priority interrupts
  - Devices needing more urgent attention get higher priority
  - Can interrupt handler for a lower priority interrupt

I/O Data Transfer

- Polling and interrupt-driven I/O
  - CPU transfers data between memory and I/O data registers
  - Time consuming for high-speed devices
- Direct memory access (DMA)
  - OS provides starting address in memory
  - I/O controller transfers to/from memory autonomously
  - Controller interrupts on completion or error

DMA

DMA Transfer from Disk to Memory Bypasses the CPU



I/O Commands

- I/O devices are managed by I/O controller hardware
  - Transfers data to/from device
  - Synchronizes operations with software
- Command registers
  - Cause device to do something
- Status registers
  - Indicate what the device is doing and occurrence of error
- Data registers
  - Write: transfer data to a device
  - Read: transfer data from a device

Instruction Set Architecture for I/O

- What must the processor do for I/O?
  - Input: reads a sequence of bytes
  - Output: writes a sequence of bytes
- Memory mapped I/O
  - Registers are addressed in same space as memory
  - Address decoder distinguishes between them
  - OS uses address translation mechanism to make them only accessible to kernel
- I/O instructions
  - Separate instructions to access I/O registers
  - Can only be executed in kernel mode
  - Example: x86