Computer Architecture

Using Tanenbaum's Modern Operating Systems (3rd edition)

The Hardware View

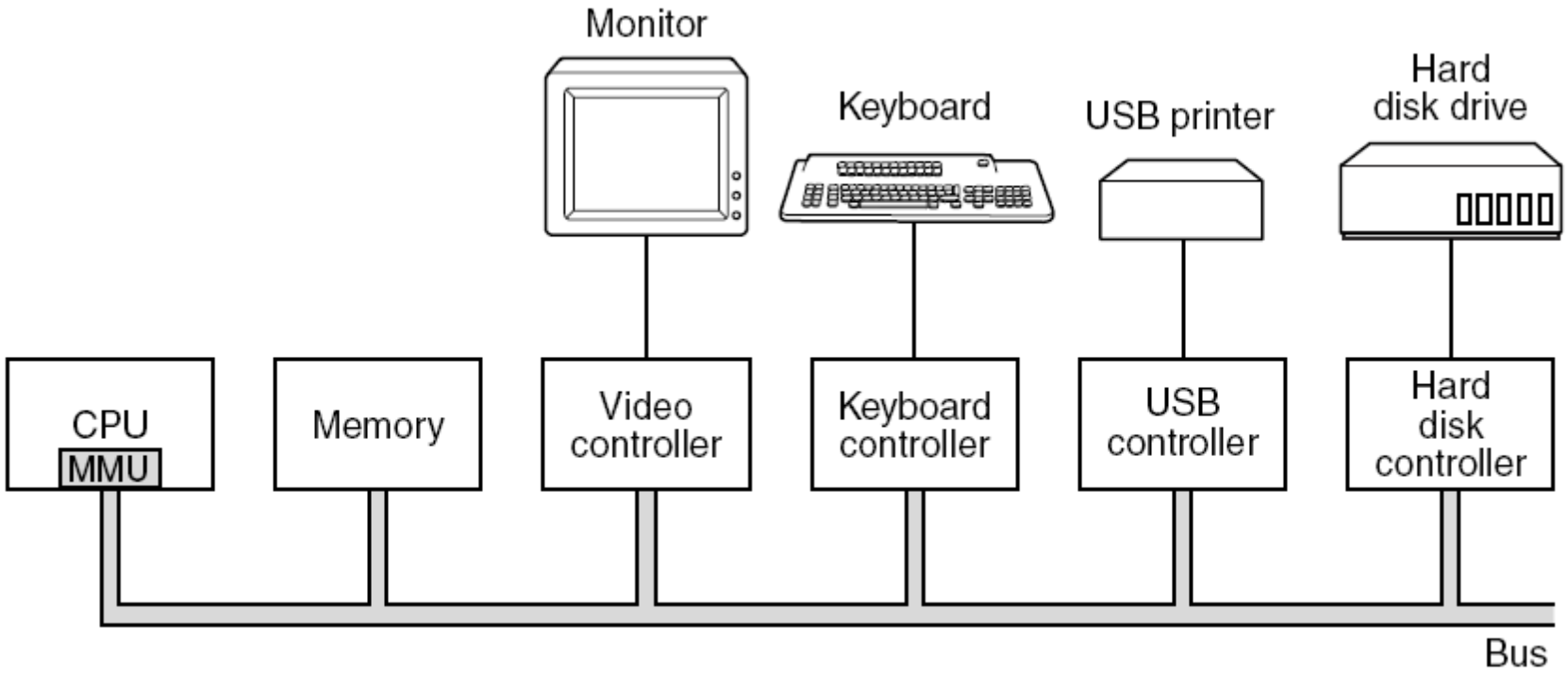
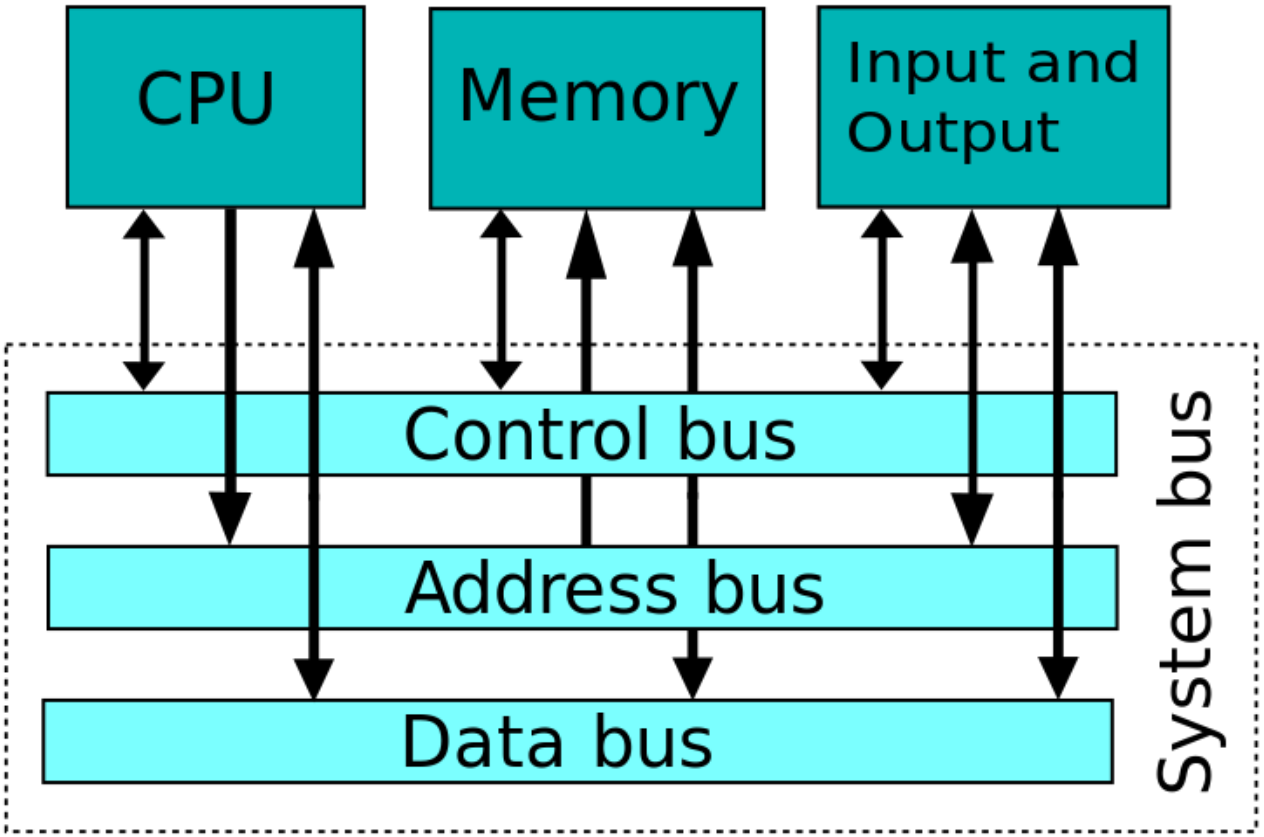


Figure 1-6. Some of the components of a simple personal computer

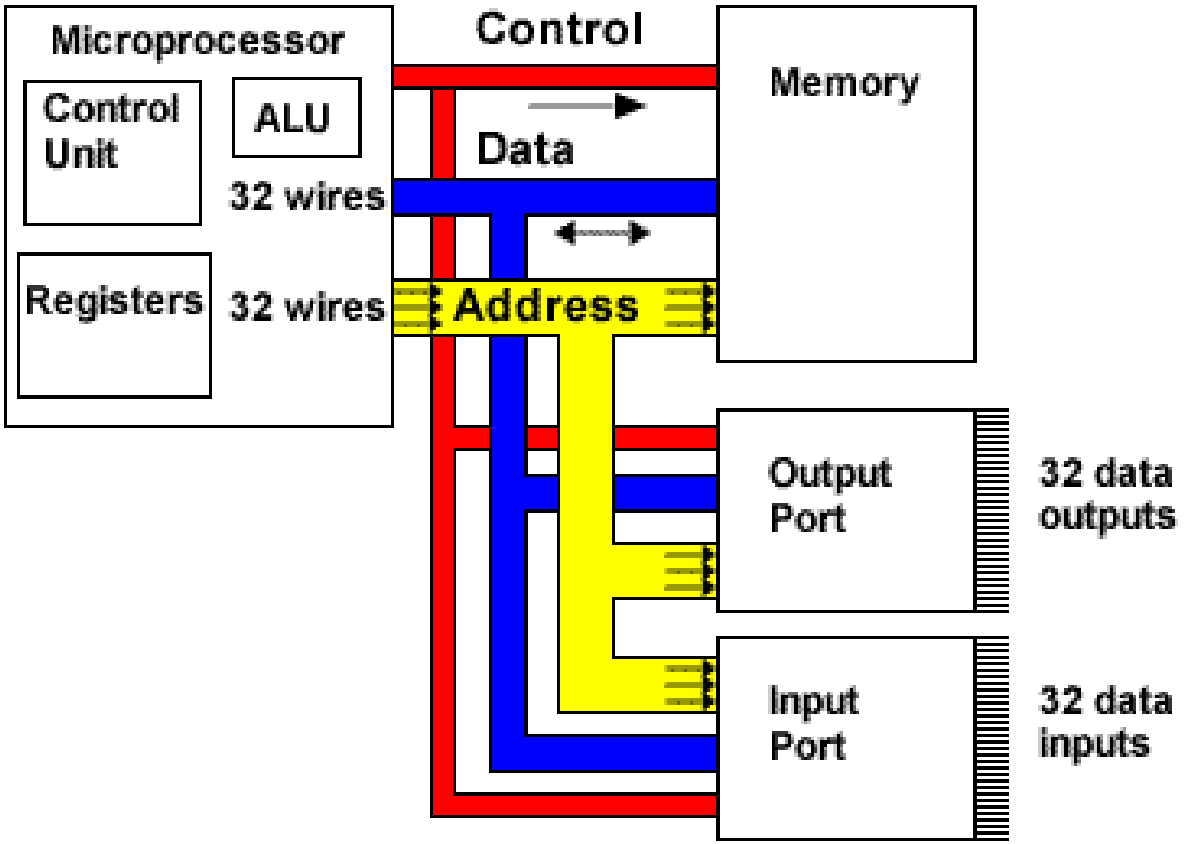
Buses

- **Data bus**
  - Carries data between CPU, memory and I/O. Note that data is anything stored or retrieved to or from memory or I/O devices.
- **Address bus**
  - Carries the address of the memory location or I/O device the CPU is accessing.
- **Control bus**
  - Carries control signals such as an interrupt has occurred, clock signals, etc.

Bus schematics (I)



Bus schematics (II)



CPU Operation

CPU Operation (I)

- Basic CPU cycles:
  - Fetch, Decode, Execute.
  - Pipeline organization.

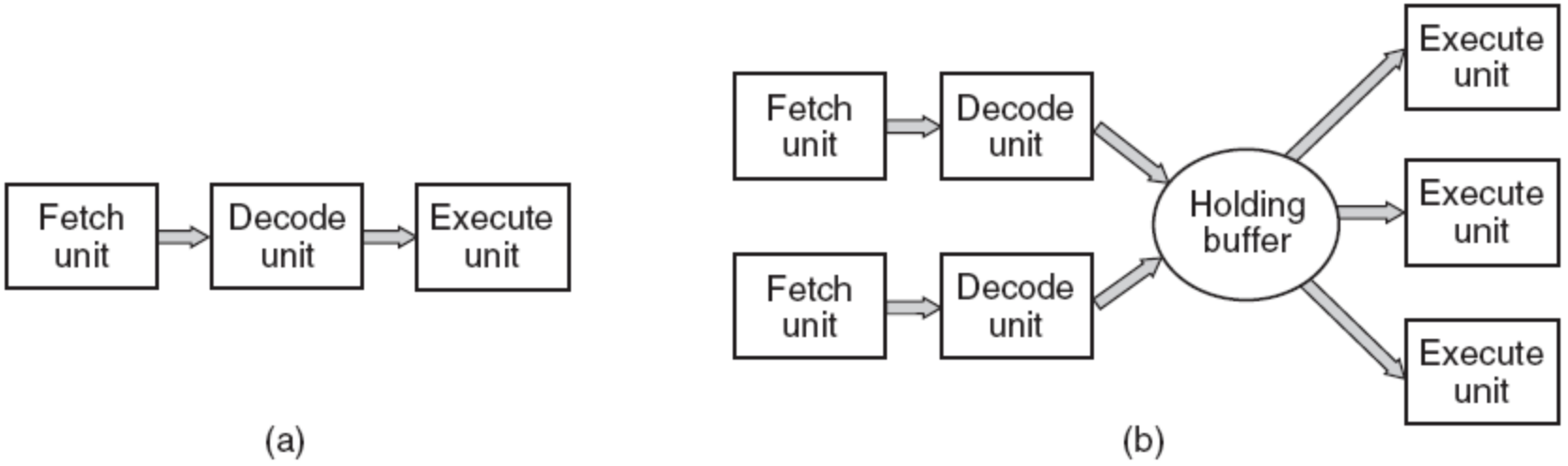


Figure 1-7. (a) A three-stage pipeline. (b) A superscalar CPU.

CPU Operation (II)

CPU Registers:

- Program Counter (**PC**) – next instruction to be fetched.
- Instruction register (**IR**) – current execution instruction.
- Program Status Word (**PSW**) – condition bits, CPU priority, user/kernel mode, …
- Stack pointer(**SP**) – top of the stack.
- Frame pointer (**FP**) – current stack frame.

MMU

- Memory management unit.
- Memory access (RAM), using virtual memory mechanism.

CPU Operation (III)

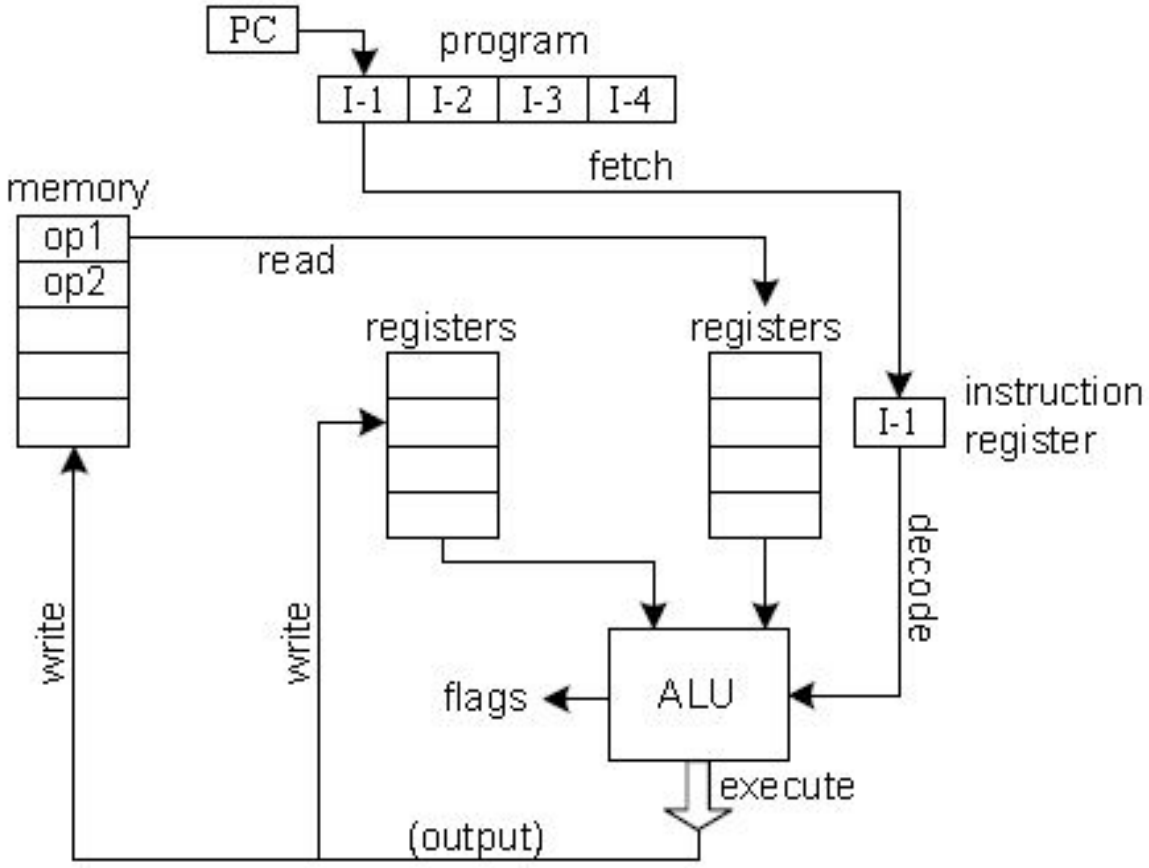


Image from: http://homepages.ius.edu/rwisman/C335/html/Chapter2.htm

CPU Operation (IV)

- At least two CPU modes of operation:
  - Kernel mode
    - Every instruction & HW feature is allowed.
  - User mode
    - Some instructions are disallowed (I/O, Memory,…)
    - Using system-call to trap the OS service.

Multiprocessor / Multicore

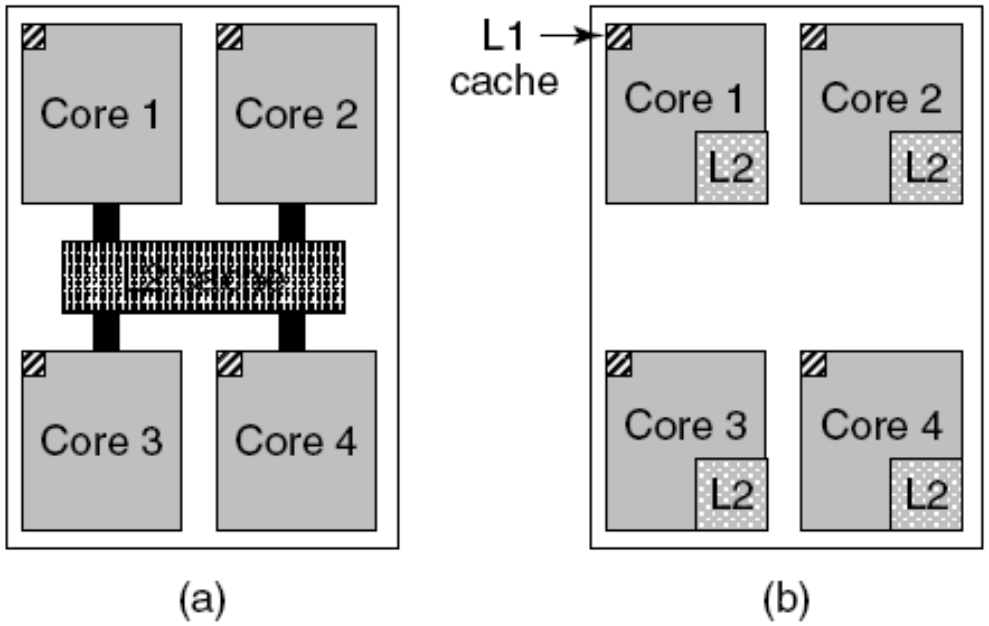


Figure 1-8. (a) A quad-core chip with a shared L2 cache.
(b) A quad-core chip with separate L2 caches.

Memory

Memory Hierarchy (I)

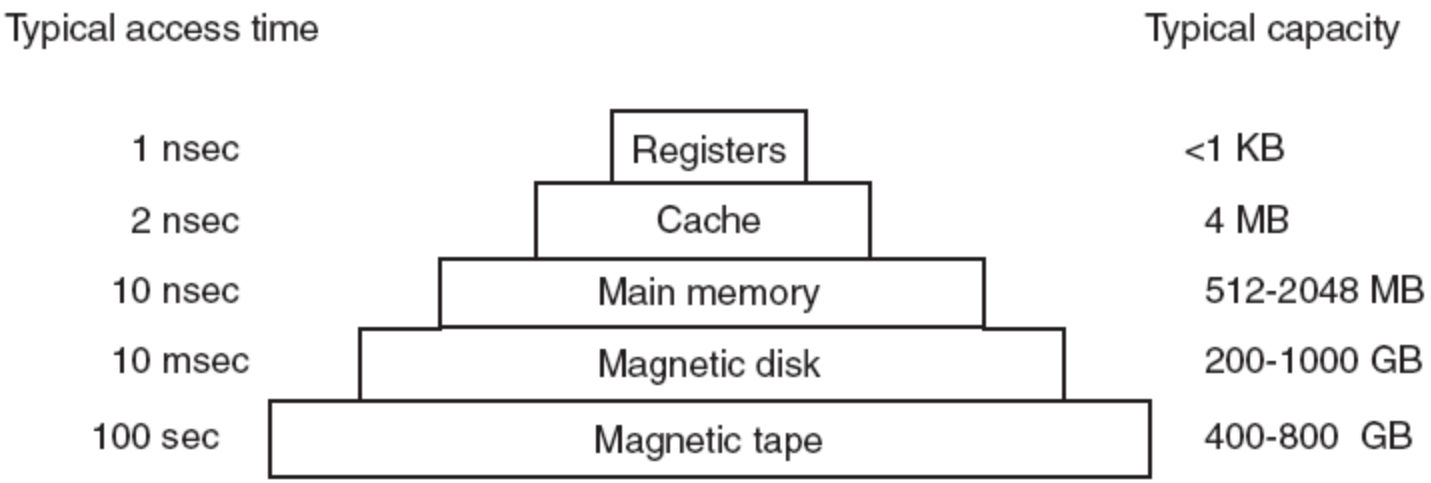


Figure 1-9. A typical memory hierarchy.
The numbers are very rough approximations.

Memory Hierarchy (II)

- Cache:
  - Level 1 (L1) – Inside the CPU, 16 KB, No delay.
  - Level 2 (L2) – Outside the CPU (AMD/Intel), several MB, 1-2 clock cycles delay.
- RAM – Main memory:
  - Random access memory, several GB.
- ROM, EPROM:
  - Nonvolatile, R/O, inexpensive, fast.
  - Bootstrap loader, low-level device control.
- E^2PROM (Flash):
  - Nonvolatile, R/W memory, slow writing.

I/O Devices – Storage

Magnetic Disk (Persistent Storage)

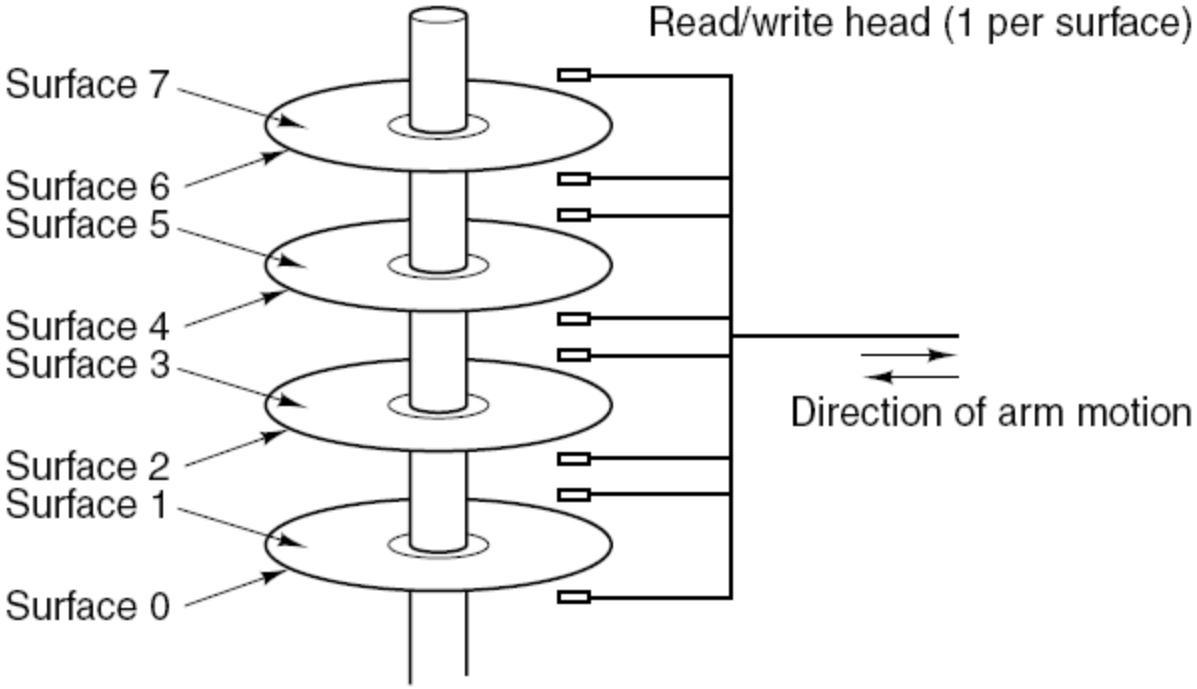


Figure 1-10. Structure of a disk drive.

Magnetic Disk Single Plate

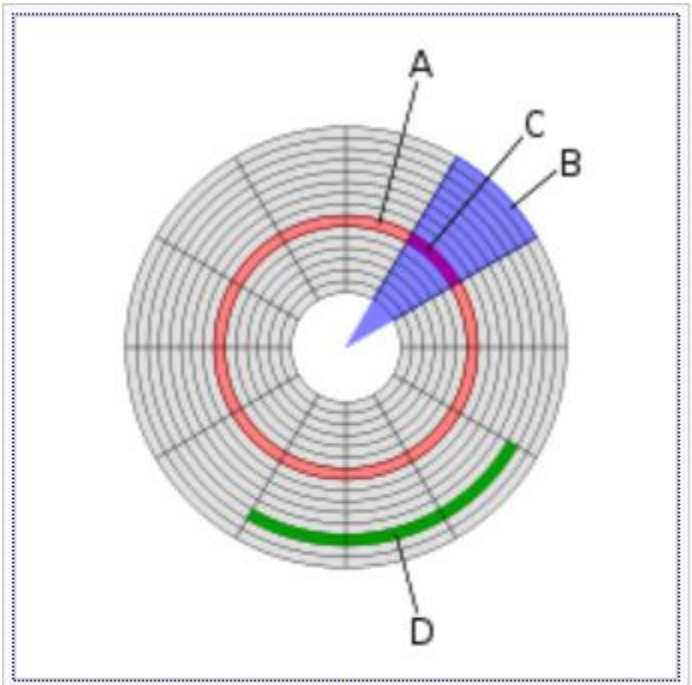


Figure 1. Disk structures:
(A) Track
(B) *Geometrical* **sector**
(C) Track sector
(D) Cluster

- Disk might be formatted with 10,000 sectors of 512 Bytes per track
- Modern disks sector may be more than 4KB

Image from:
http://en.wikipedia.org/wiki/Disk_sector

Disk Performance

- **Bandwidth**
  - Number of bytes transferred divided by time between the request for service and the completion of the transfer
  - Depends on Access time that depends on:
    - Seek time - Time to move the head (≃ 10 msec).
    - Latency time -Time for disk to rotate to desired place (≃ 4 msec)
    - To make it clear: disk write takes only ≃ 50µsec

Solid-State Disk (SSD)

- Solid state (electronic disk) is
  - Essentially a Flash memory formatted as disk in blocks
  - Have no moving (mechanical) components
  - Uses integrated circuit assemblies as memory to store data persistently
- SSDs are typically:
  - More resistant to physical shock
  - Run silently
  - Have lower access time, and less latency
  - Six to seven times more expensive per unit of storage than HDDs (for now)
- SSHD
  - Combines the features of SSD and HDD in the same unit, containing a large HD and SSD cache to improve performance

Figure 1-12. Modern 2.5-inch SSD
http://en.wikipedia.org/wiki/Solid-state_drive

I/O Devices – Operation

I/O Devices

- Device ↔ Controller
- Device Driver
- I/O Modes of Operation:
  - Polling, Busy waiting
    - Application→ System Call→ Device Driver → Device.
    - Driver sits in a tight loop – poling the device, to see if done.
  - Interrupt
    - Application→ System Call→ Device Driver → Device.
    - The caller is blocked - the OS is free to look for other task.
    - The device generates an interrupt to signal completion.

Interrupts (I)

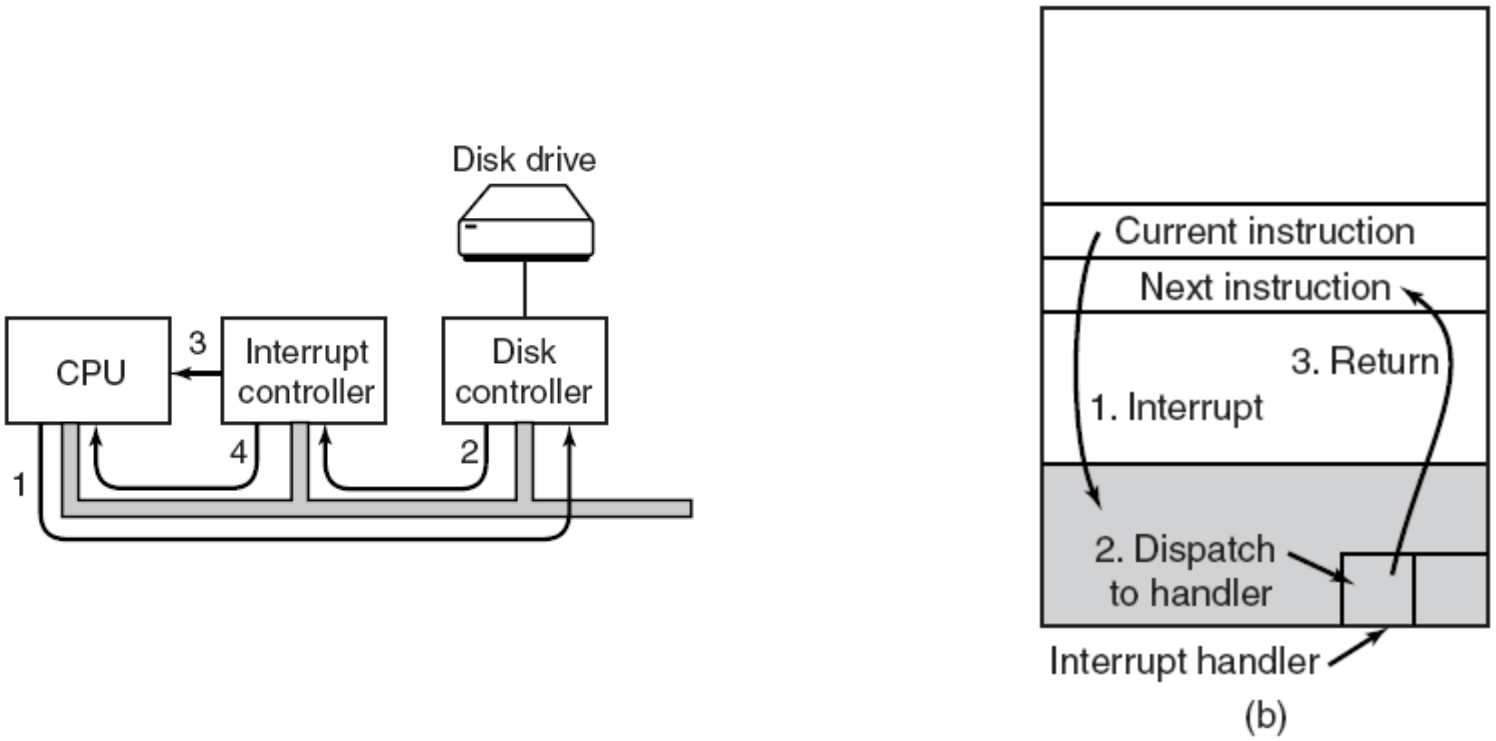


Figure 1-13. (a) The steps in starting an I/O device and getting an interrupt. (b) Interrupt processing involves taking the interrupt, running the interrupt handler, returning to the user program.

Interrupts (II)

Legend*: *HW
SW

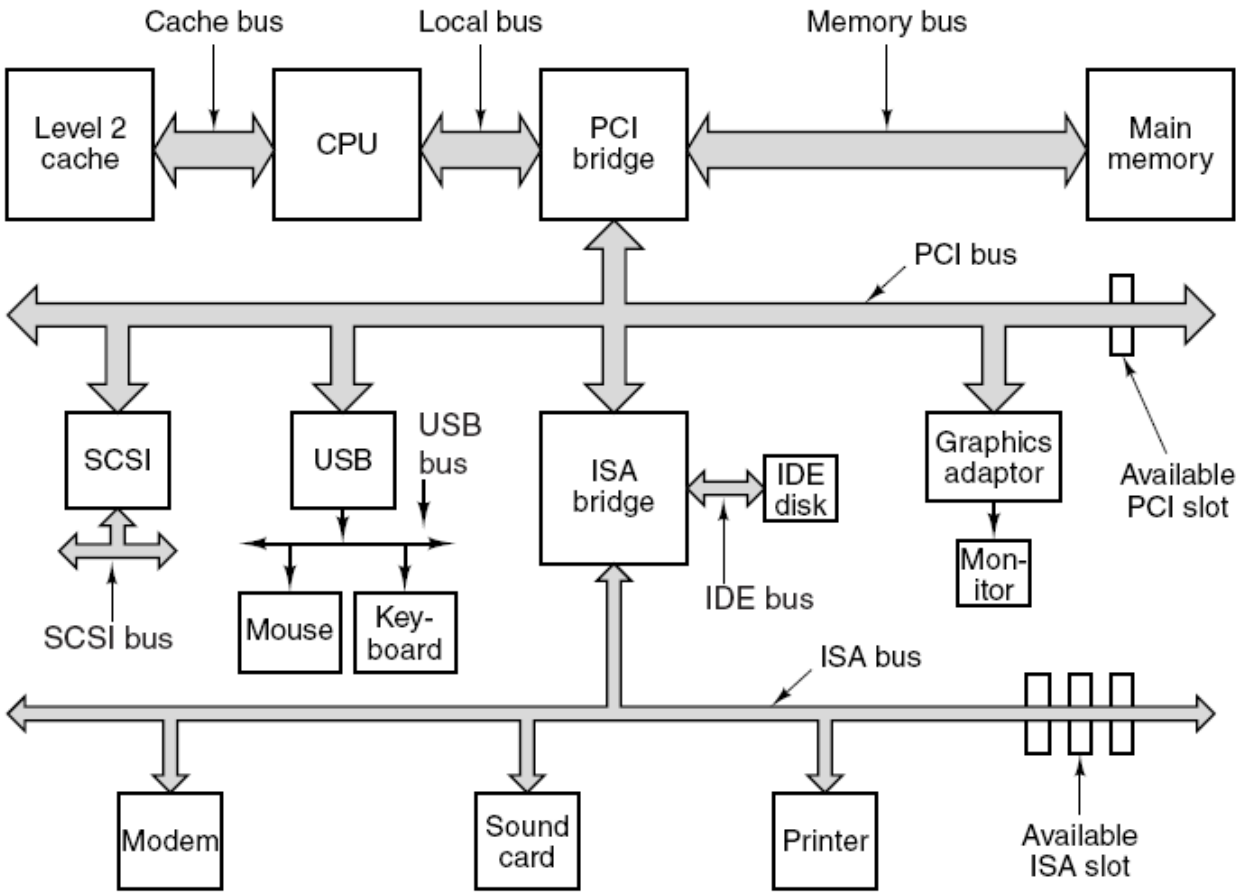
I/O using interrupt mechanism:

1. The driver writes to device registers (Using the controller).
2. The controller signals – the job done.
3. If the interrupt is accepted by the interrupt controller - the CPU is signaled.
4. The CPU itself ***mask*** the interrupt – if accepted:
    I. Push the PC & PSW of the current program.
    II. Switch to kernel mode.
    III. Extract the ***interrupt handler*** from the ***interrupt vector*** using the device index.
    IV. Run the ***interrupt handler*** **.**
    V. Return to the pushed program.

Interrupts (III)

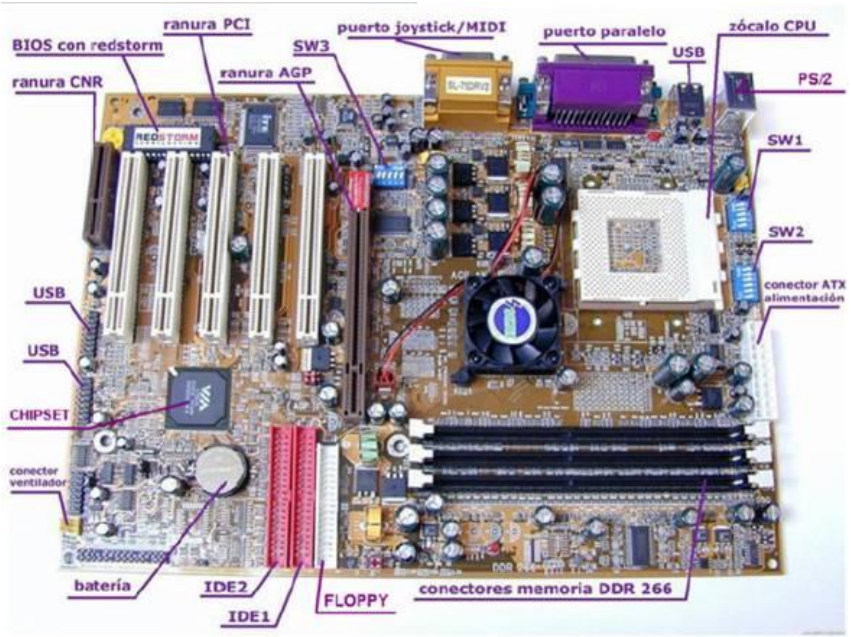
- CPU can be interrupted by 3 types of events
  - HW Interrupt – asynchronous events
  - Trap (a.k.a. SW Interrupt) – synchronous event
  - Command execution Exception – error event

Pentium architecture (I)



Pentium architecture (II)

- Two main buses:
  - ISA for b/w compatible, old and slow I/O cards.
  - PCI high-speed I/O devices.
- Three secondary busses:
  - IDE for Disk controller interface.
  - USB to attach all slow I/O devices.
  - SCSI (SATA today) for fast disks, scanners, etc.
- Two working methods:
  - Fixed: each I/O had a fixed interrupt # and registers.
  - Plug & Play: collect info and assign interrupt # and registers for each device.



Boot Sequence

BIOS (Basic Input Output System):

- Held in a flash RAM
- Started while booting
- Checks:
  - How much RAM?
  - Are basic devices installed?
  - Detect other devices (Legacy / Plug&Play)
  - Determine the boot device
  - Secondary boot loader is read
  - The OS is read from the active partition and started
    - Performs its own checking (drivers, etc.), login…

HW + Interview Questions

- What are the 3 components of the BUS? Explain the R&R of each one.
- Explain the following:
  - Kernel Mode
  - User Mode
  - Differences between them
- Explain the R&R of the following registers:
  - PC
  - IR
  - PSW
  - SP
  - FP