Operating Systems: Introduction

Abhijit A. M.
abhijit.comp@coep.ac.in

Credits: Slides of “OS Book” ed10.

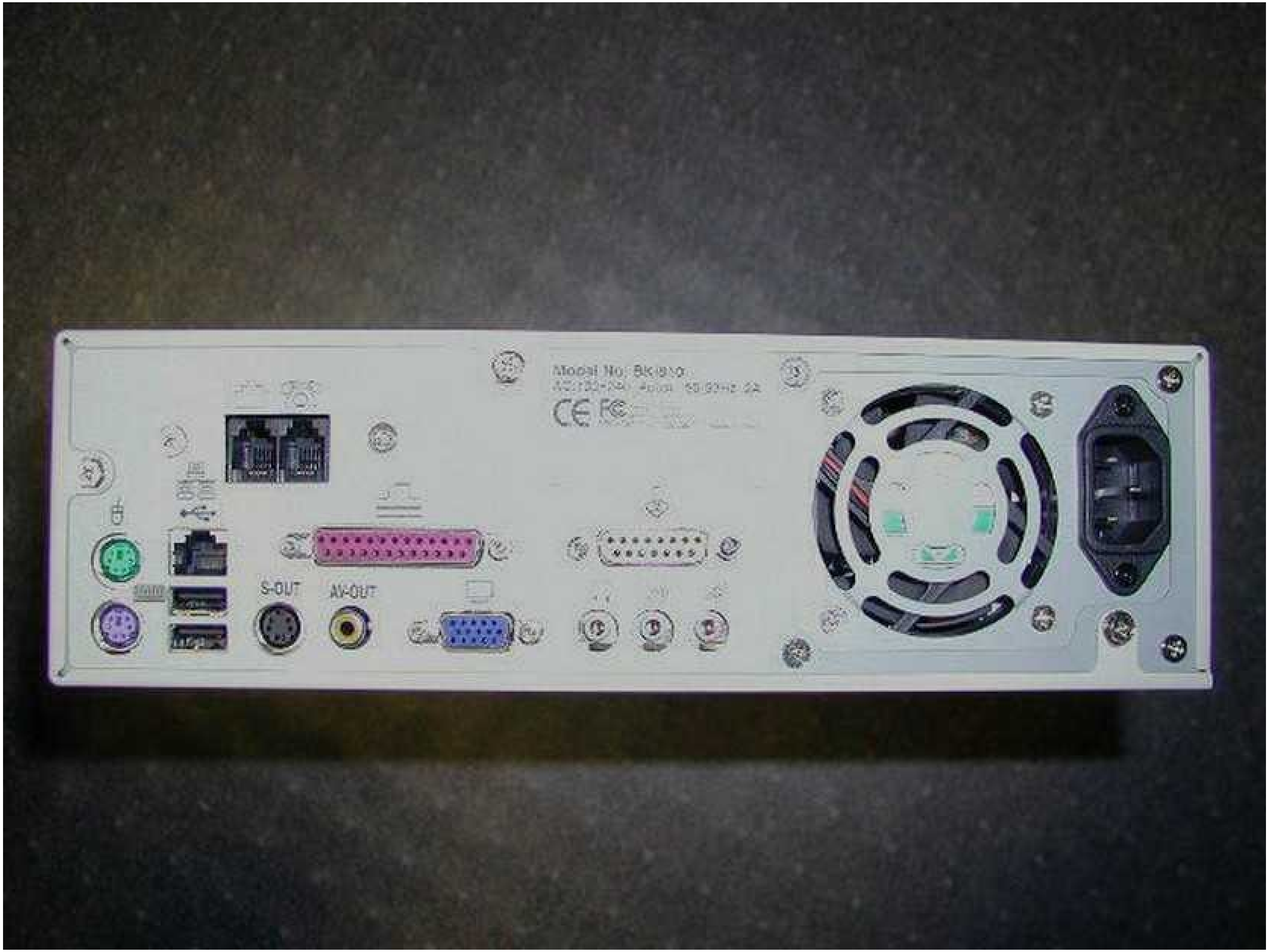
Initial lectures

We will solve a jigsaw puzzle

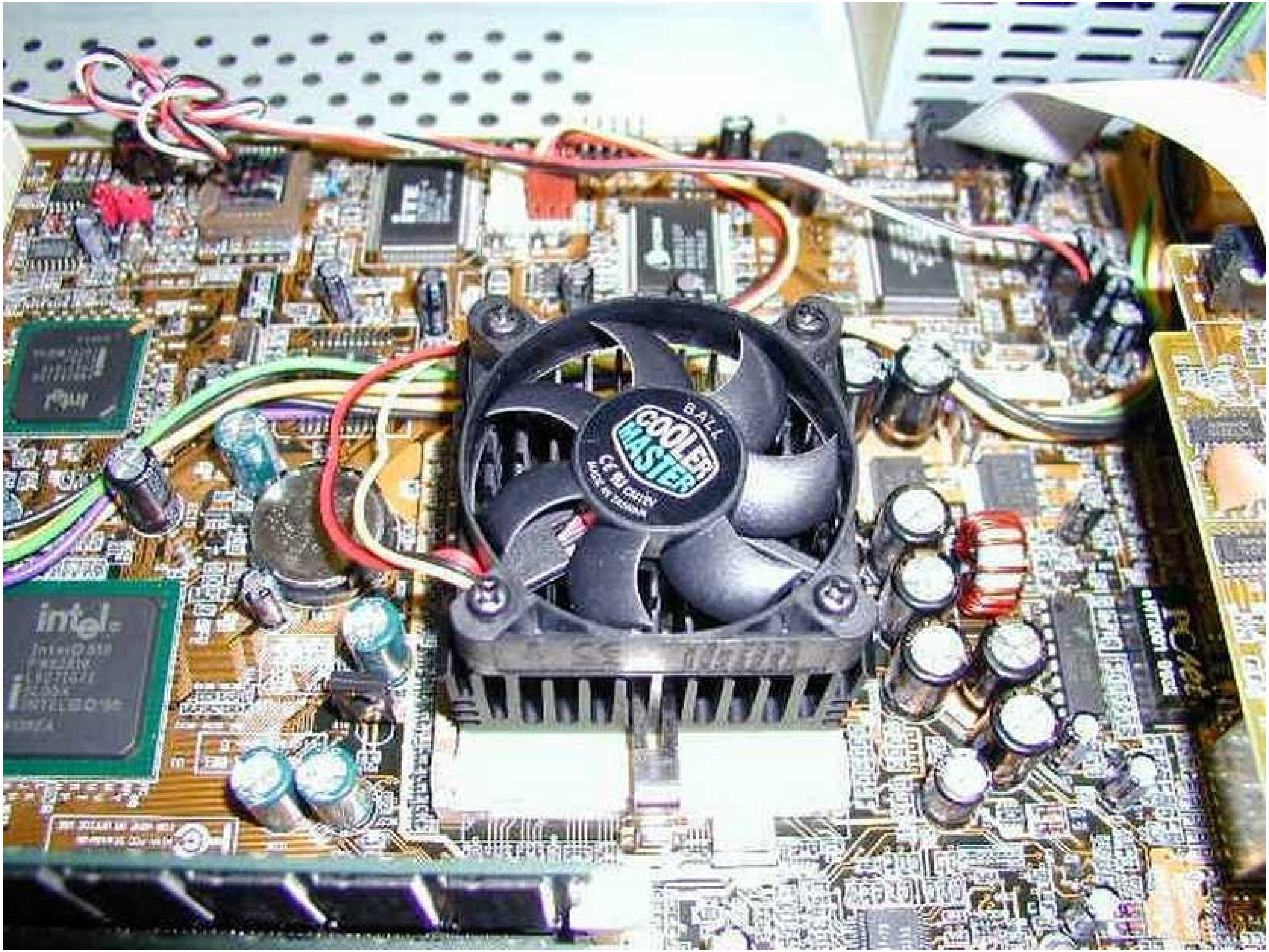
of how the computer system is built

with hardware, operating system and system programs

The “Ports”, what users see



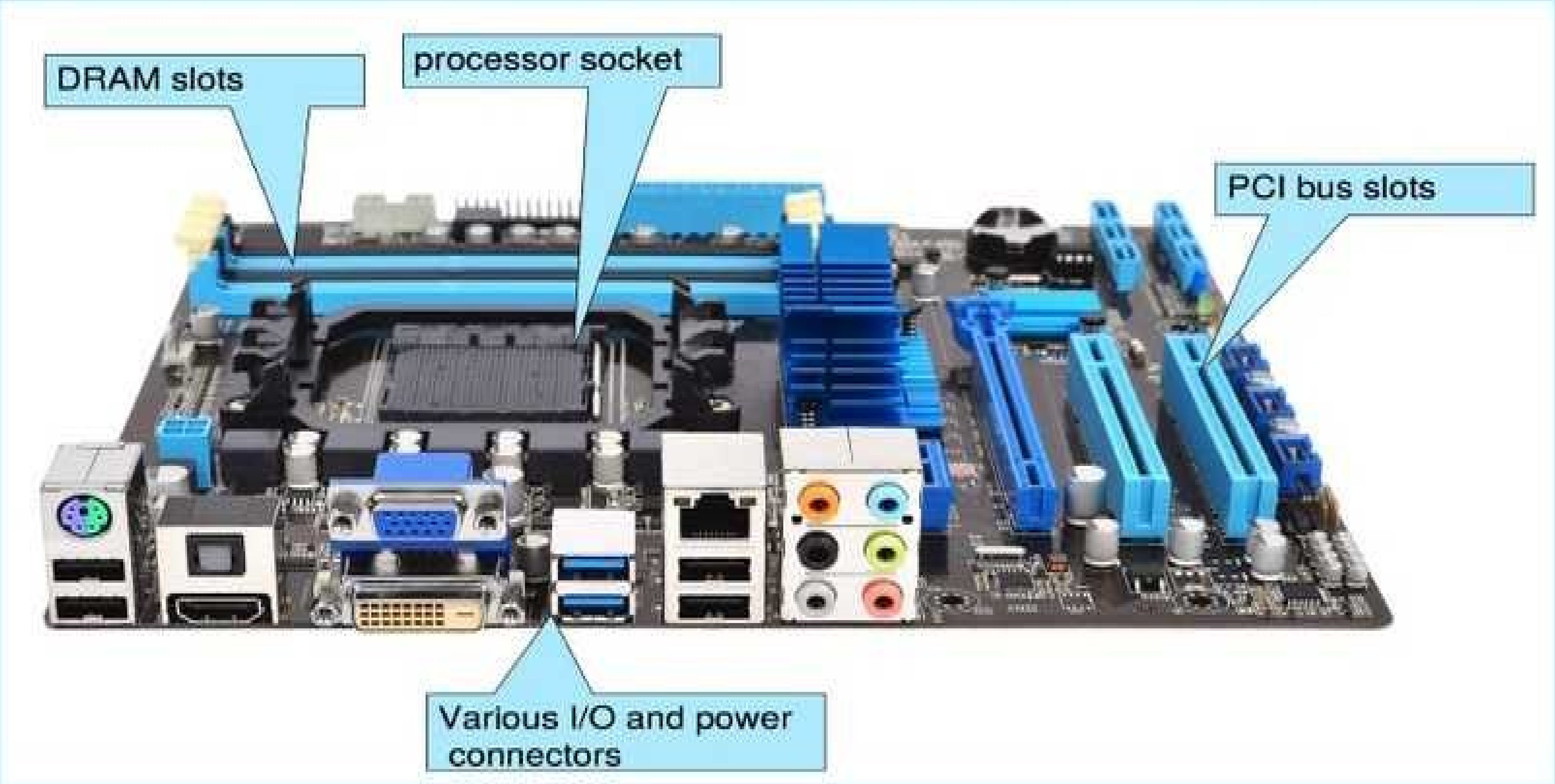
Revision: Hardware : The Motherboard



CPU/Processor

- I3,i5, etc.
- Speeds: Ghz
- “Brain”
- Runs “machine instructions”
- The actual “computer”
- Questions:
    - Where are the instructions that the processor runs?

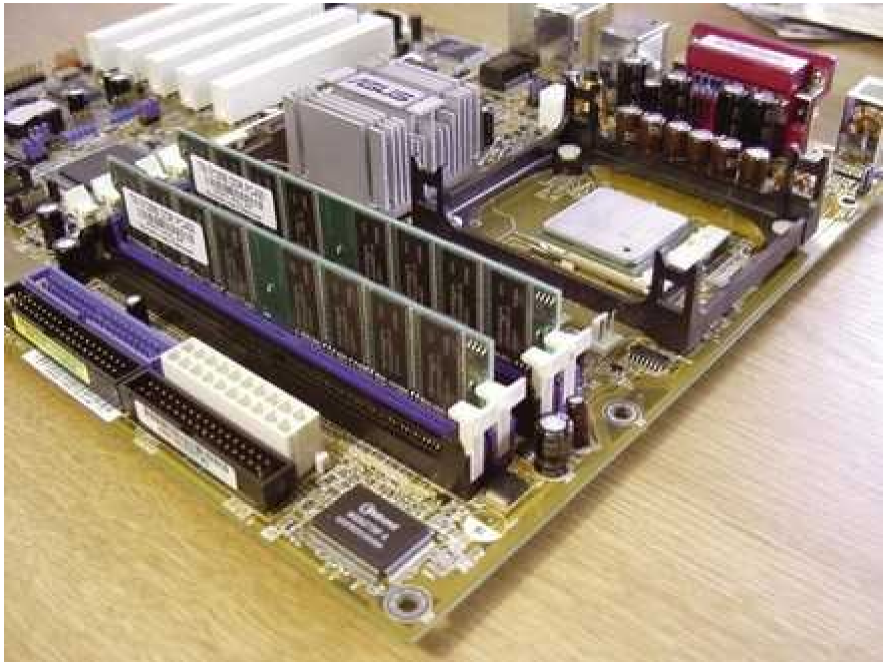
What’s on the motherboard?

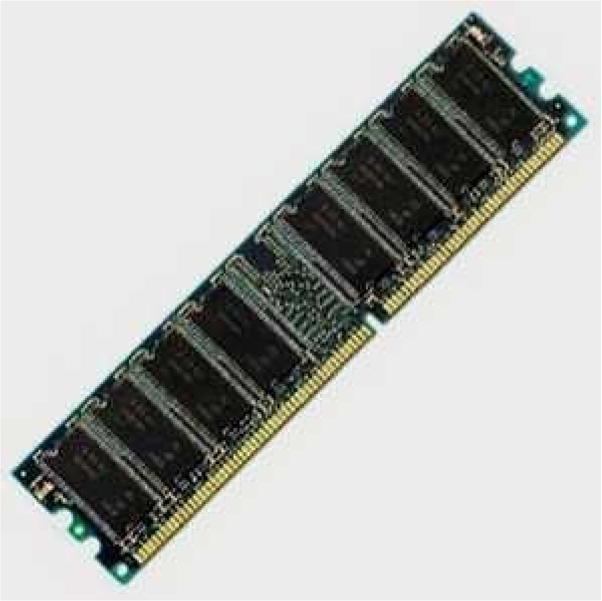


This board is a fully-functioning computer, once its slots are populated.

Memory

- Random Access Memory (RAM)
  - Same time of access to any location – randomly accessible
  - Semiconductor device

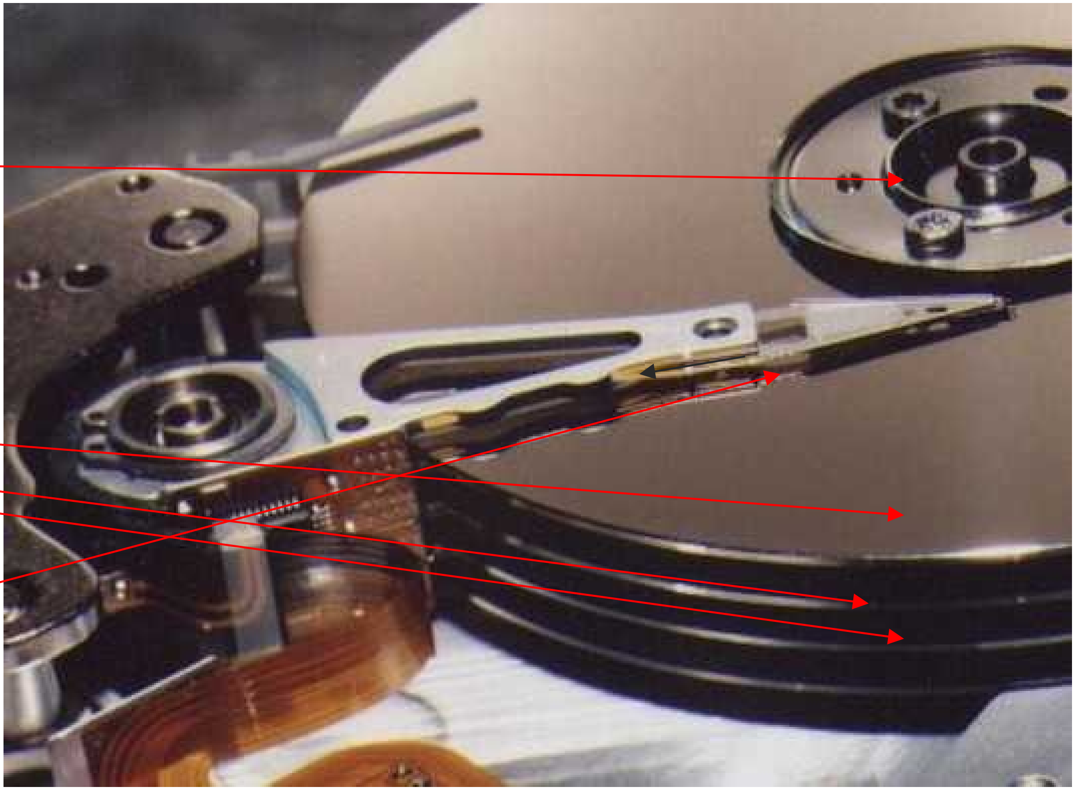




Question:
Can we add more RAM to a computer?

The Hard Drive

The Hard Drive

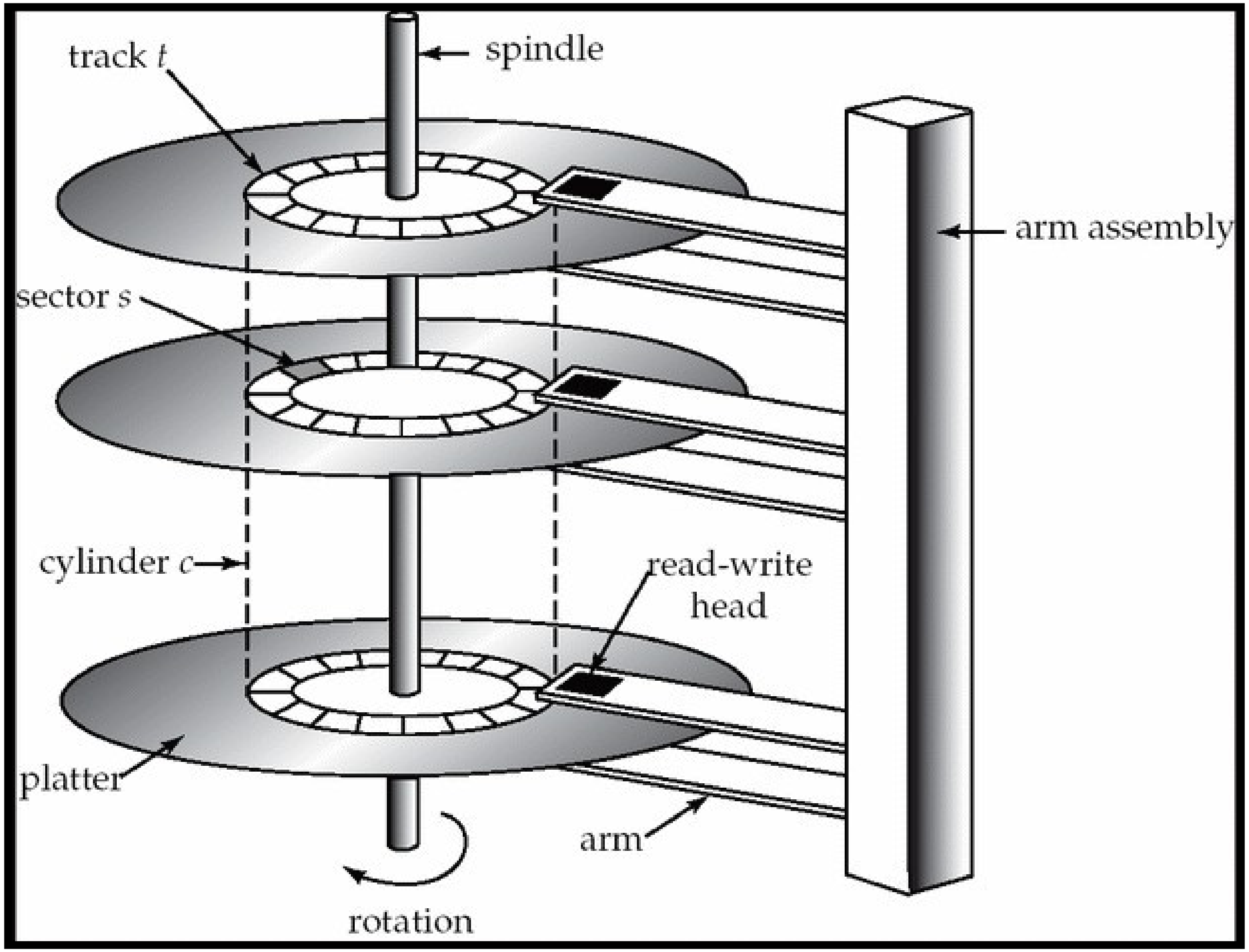


Spindle

Magnetic disks

Head

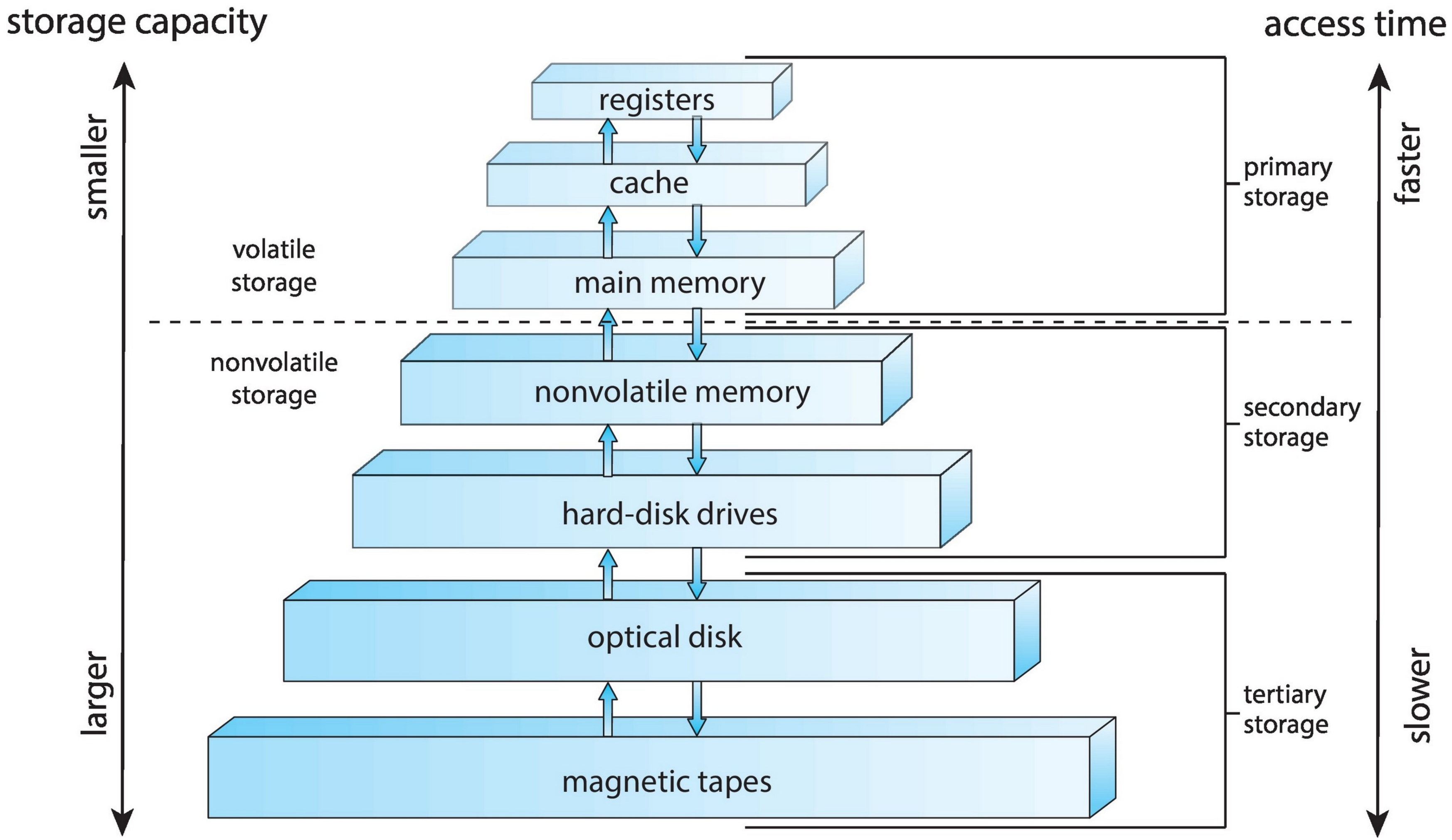
The hard Drive

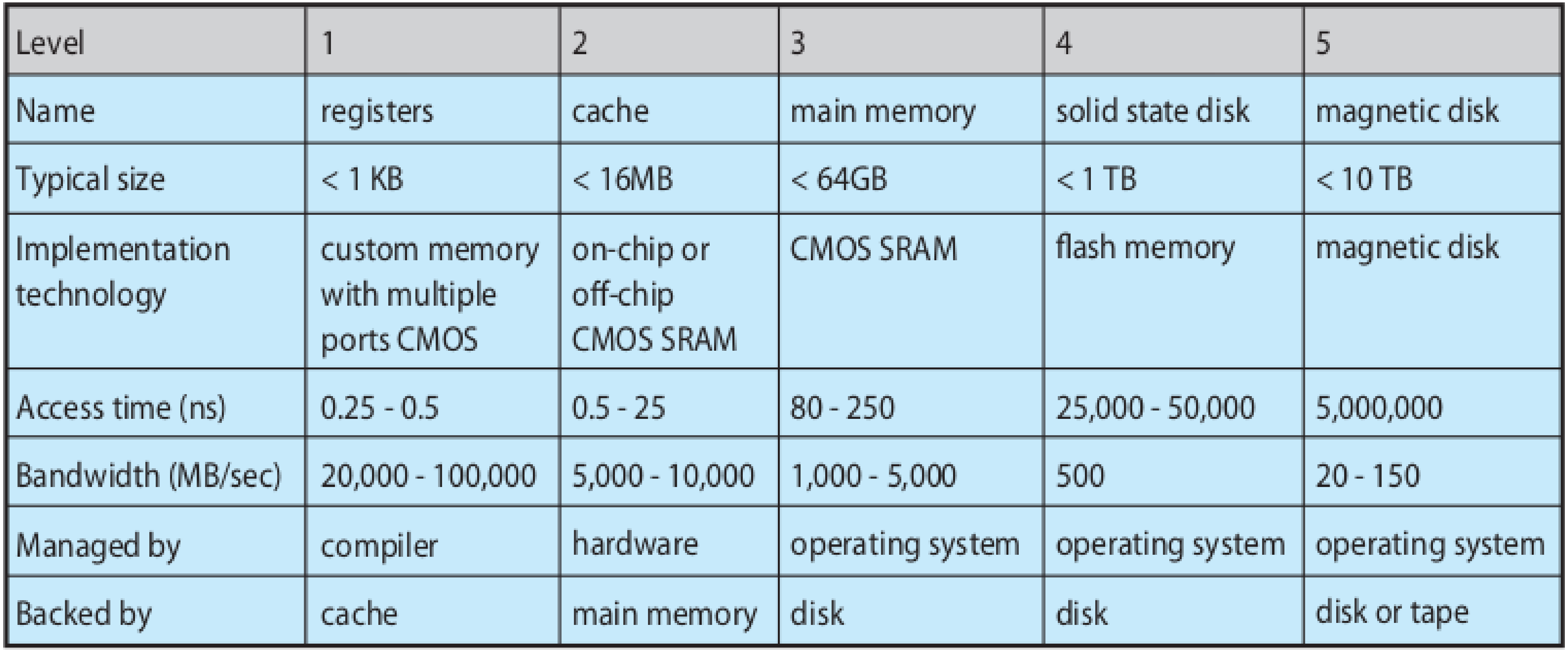


The Hard Drive

- Is a Magnetic device
- Each disk divided into tiny magnetic spots, each representing 1 or 0
  - What's the physics ?
  - Two orientations of a magnet
- Is “persistent”
  - Data stays on powering-off
- Is slow
- IDE, SATA, SCSI, PATA, SAS, …

Storage-Device Hierarchy





Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

Figure 1.11 Performance of various levels of storage.

Computer Organization

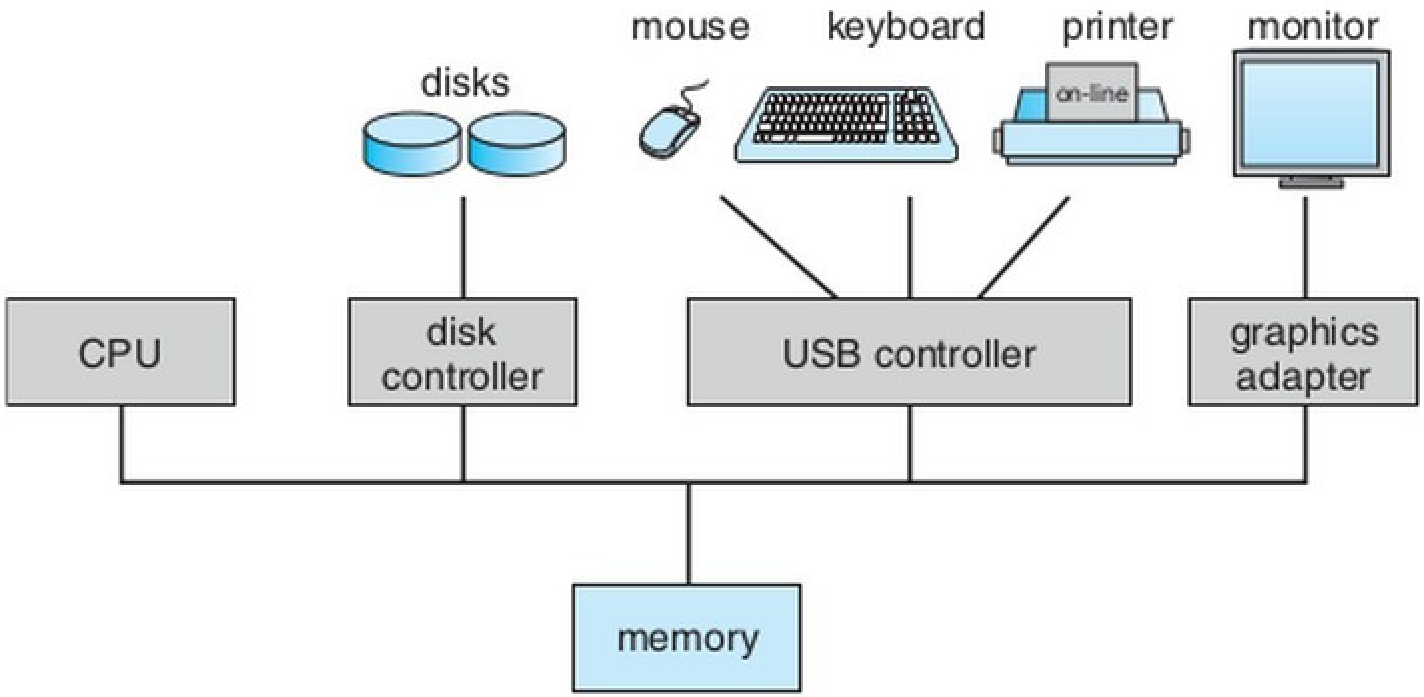


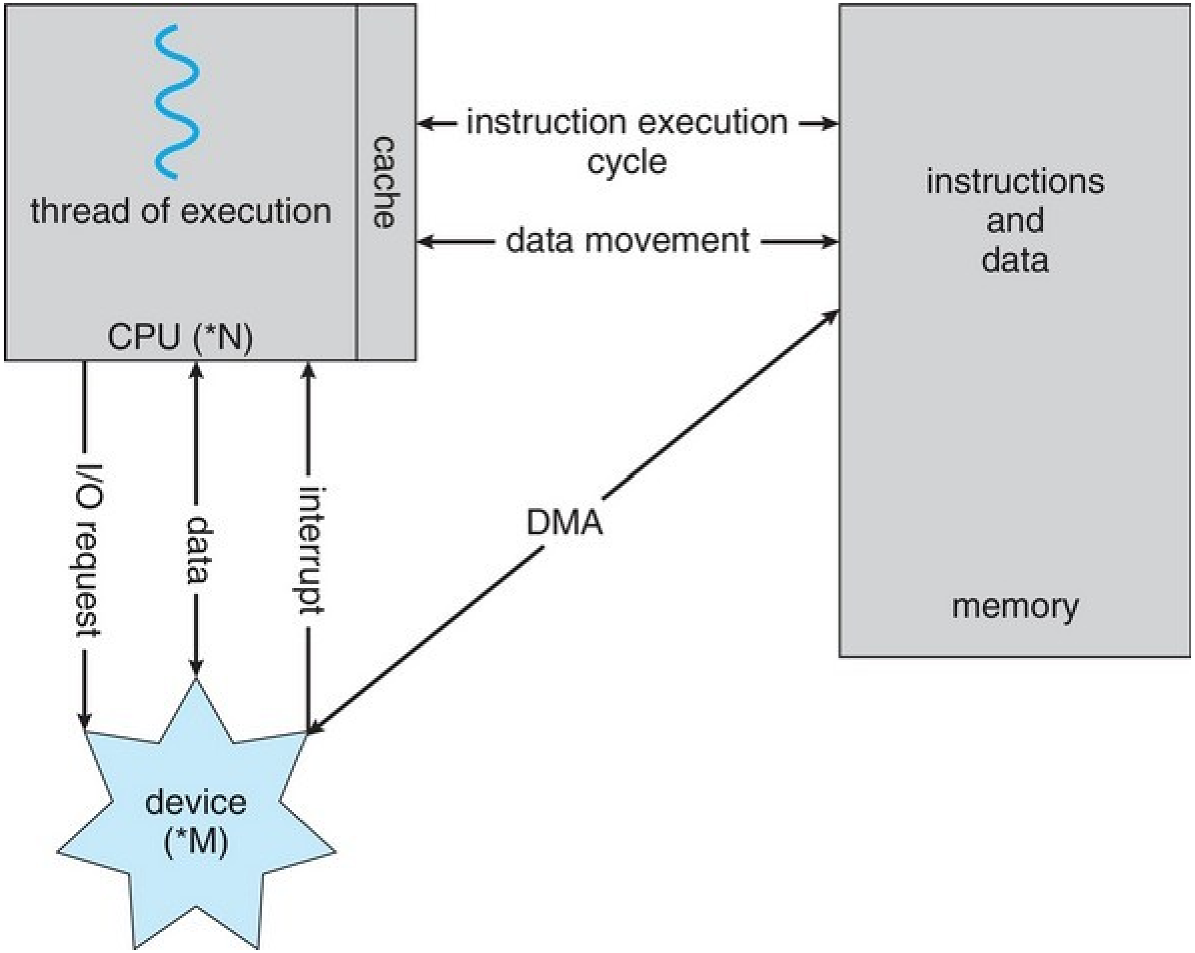
Figure 1.2 A modern computer system.

https://aycaakcay.medium.com/computer-system-organization-6f04c5a314fd

Important Facts

- Processor (CPU) transfers data between itself and main memory(RAM) only!
- No data transfer between CPU and Hard Disk, CPU and Keyboard, CPU and Mouse, etc.
- I/O devices transfer (how?) data to memory(RAM) and CPU instructions access data from the RAM

How a Modern Computer Works



A von Neumann architecture

What does the processor do?

- From the moment it’s turned on until it’s turned off, the processor simply does this

  1) Fetch the instruction from RAM (Memory).

     Location is given by Program Counter (PC) register

  2) Decode the instruction and execute it

     While doing this may fetch some data from the RAM

  3) While executing the instruction change/update the Program Counter

  4) Go to 1

Immediate questions

- What’s the initial value of PC when computer starts ?
- Who puts “this” value in PC ?
- What is there at the initial location given by PC ?

A critical question you need to keep thinking about ...

- Throughtout this course, With every concept that you study, Keep asking this question
- Which code is running on the processor?
  - Who wrote it?
  - Which code ran before it
  - Which code can run after it
- Basically try to understand the flow of instructions that execute on the processor

Few terms

- BIOS
  - The code “in-built” into your hardware by manufactuerer
  - Runs “automatically” when you start computer
  - Keeps looking for a “boot loader” to be loaded in RAM and to be executed
- Boot Loader
  - A program that exists on (typically sector-0 of) a secondary storage
  - Loaded by BIOS in RAM and passed over control to
  - E.g. “Grub”
  - It’s job is to locate the code of an OS kernel, load it in RAM and pass control over to it

Kernel, System Programs, Applications

- **Kernel**
  - The code that is loaded and given control by BIOS initially when computer boots
  - Takes control of hardware (how?)
  - Creates an environment for "applications" to execute
  - Controls access to hardware by applications,
  - Etc.
- **Everything else is "applications"**
  - System programs: applications that depend heavily on the kernel and processor
  - E.g. Compiler, linker, loader, etc.
  - Other applicatiosn: GUI, Terminal, Libreoffice, Firefox, VLC, ... Your own programs from data structures, etc.

How is a modern day
Desktop system
built
on top of
this type of hardware?

Components of a computer system

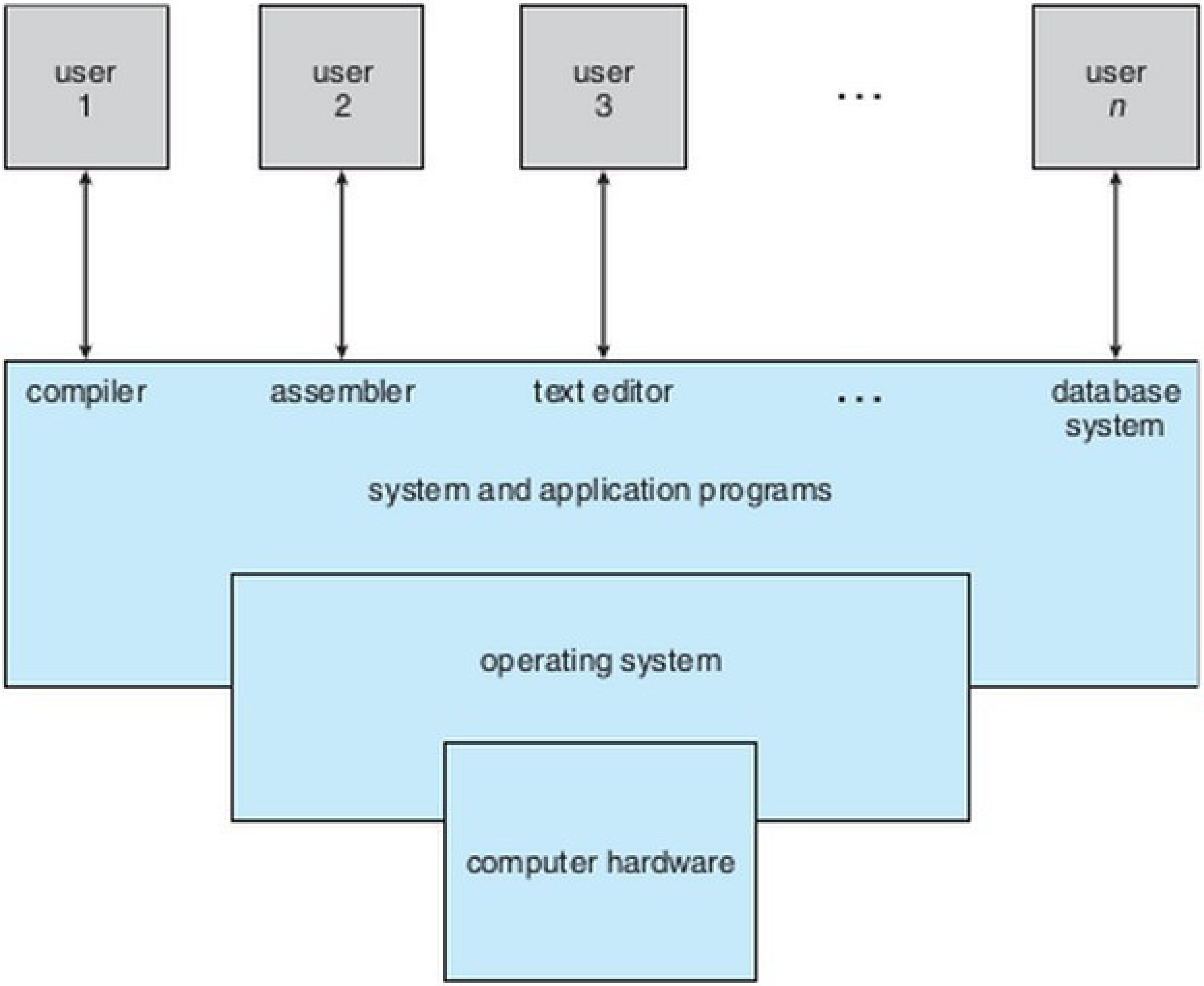


Figure 1.1 Abstract view of the components of a computer system.

Multiprocessor system: SMP

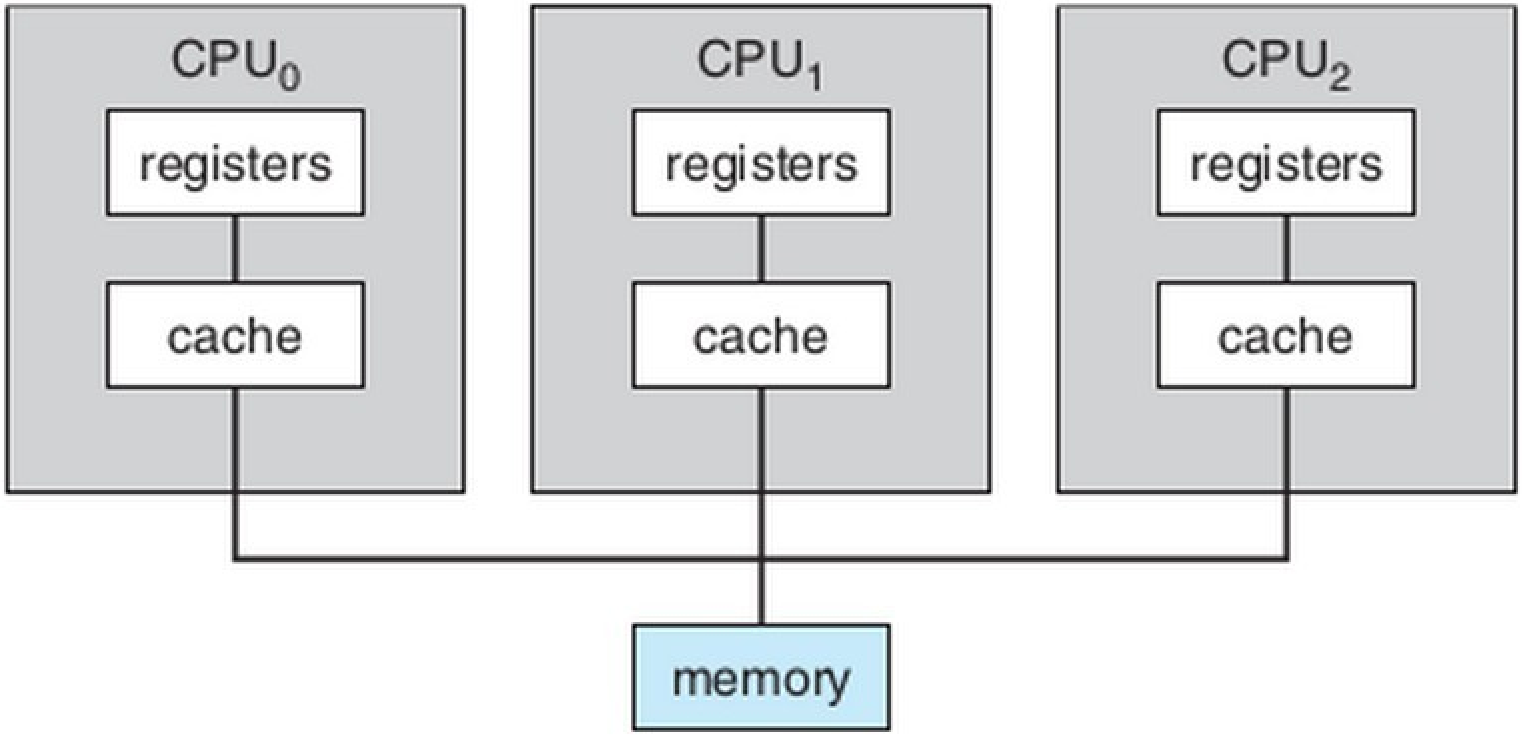


Figure 1.6 Symmetric multiprocessing architecture.

Dual Core: what's that?

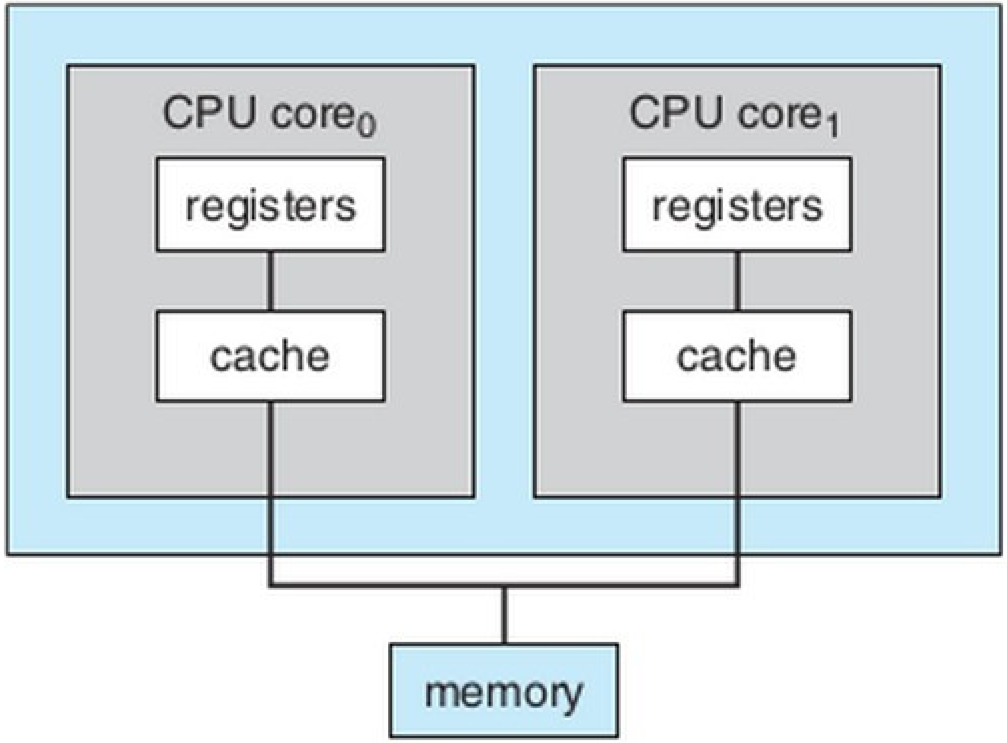


Figure 1.7 A dual-core design with two cores placed on the same chip.

The very important question:

Who does what?

What is done in hardware, by OS, by compiler, by linker, by loader, by human end-user?

There is no magic!

A very intelligent division of work/labour between different components of the computer system makes a system