Information Technology Institute

Operating Systems

Lectures 3

Difference between 32-bit and 64-bit operating systems

- there are two types of processors existing, i.e., 32-bit and 64-bit processors.
- *A 32-bit system can access 2^{32} different memory addresses, i.e 4 GB of RAM or physical memory.*
- *A 64-bit system can access 2^{64} different memory addresses, i.e actually 18-Quintillion bytes* (18 exabytes) *of RAM.*
- A computer with a 64-bit processor can have a 64-bit or 32-bit version of an operating system installed. However, with a 32-bit operating system, the 64-bit processor would not run at its full capability.

Ref: https://www.geeksforgeeks.org/difference-32-bit-64-bit-operating-systems/?ref=lbp
Ref: https://www.javatpoint.com/32-bit-vs-64-bit-operating-system

I/O Management

Content

- Need of I/O Management
- I/O Subsystem

- I/O Devices Categories:
  - Block Devices
  - Character Devices

- Synchronization and Critical Sections
  - Mutex
  - Semaphore

- Deadlocks

Course Outlines

1.2.5 Input Devices

- Which are peripherals used to **provide data and control signals** to a computer.
- Input devices allow us to enter raw data for processing. For Example:
    - Keyboard (default input device)
    - Microphone
    - Scanner (2D and 3D)
    - Mouse
    - Trackball
    - Touchpad
    - Barcode and QR Code readers
    - Digital Camera

1.2.6 Output Devices

- Which are pieces of computer hardware used to **communicate the results** of data processing performed by a computer.
- The objective of output devices is to turn computer information into a human friendly/readable form. For Example:
  - Screen (Monitor or Console) (default output device) – LED or LCD
  - Data Projectors
  - Speaker and Headphones
  - Printer (2D and 3D) – inkjet or laser
  - Plotter (wide format printer)
  - Cutter (2D or 3D)

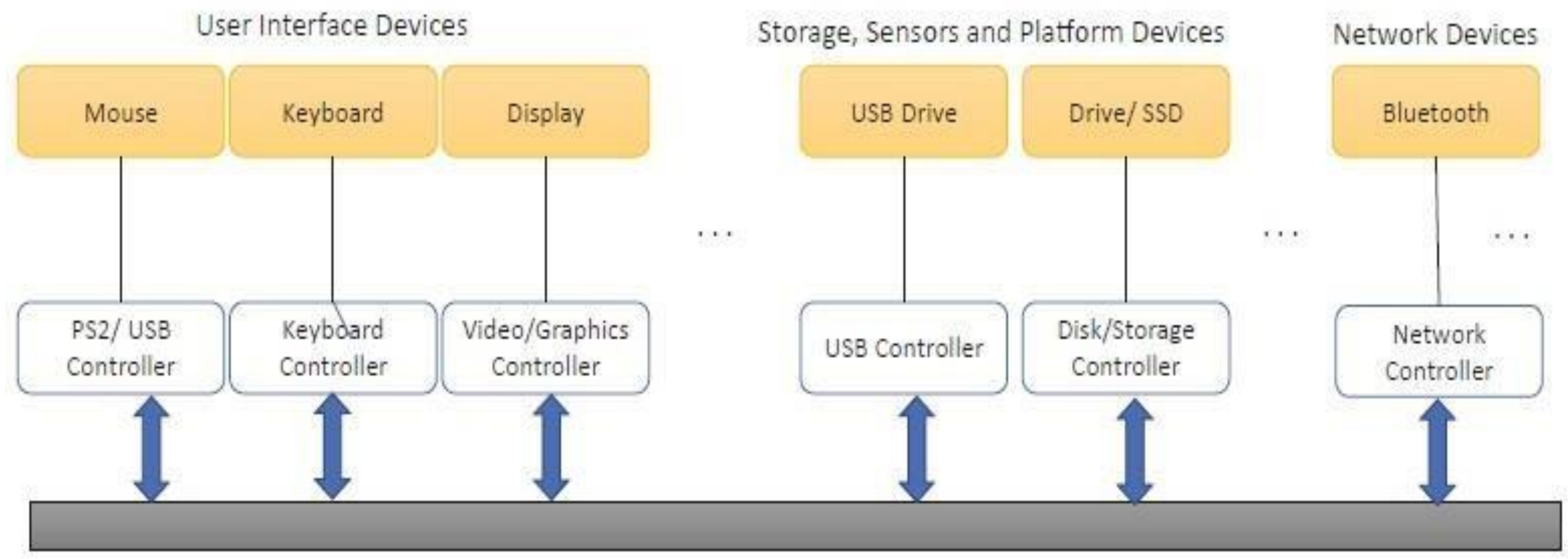
1.2.7 Input/Output Devices

- Some devices work as Input and Output devices like:
  - Touch Screen
  - Network
  - Most of I/O Ports (Both of serial and parallel)
  - New types of ports, like USB and Type-C ports
- All of the secondary storages used as I/O devices for permeant storage, like:
  - Hard disk
  - Floppy disk
  - Flash memory
  - CD and DVD

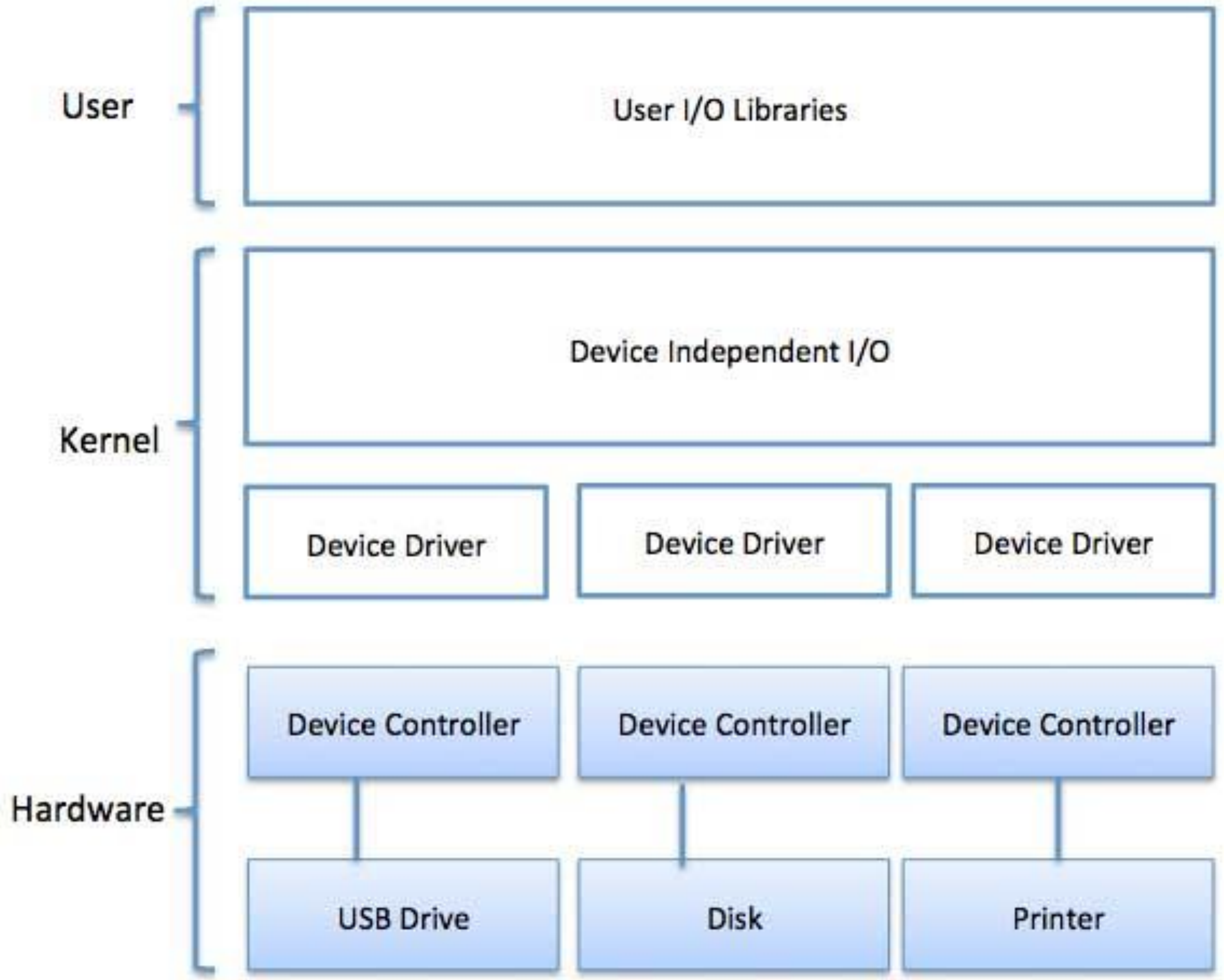
Need for I/O Management

- As part of the system, there could be **multiple** devices that are **connected** and perform different **input-output** functions.
- These I/O devices could be used for human interaction such as display panel, touch panels, keyboard, mouse, and track pads, to name a few.
- Another I/O devices could be to connect the system to storage devices, sensors, and so on. There could also be I/O devices for networking needs that implement certain parts of the networking stack. These could be Wi-Fi, Ethernet, and Bluetooth devices and so on.
- They vary from one to another in the form of protocols they use to communicate such as the data format, speed at which they operate, error reporting mechanisms, and more.
- The OS presents a unified I/O system that abstracts the complexity from applications. **The OS handles this by establishing protocols and interfaces with each I/O controller**. However, *the I/O subsystem usually forms the complex part of the operating system due to the dynamics and the wide variety of I/Os involved*.

Need for I/O Management



Device controller & device driver



Device controller & device driver

- A **device driver** **is a code** inside the OS that allows to be empowered with the specific commands needed to operate the associated device.

The code is implemented by the device manufacturer which allows the **device** to **communicate** with the computer’s OS. Without device drivers, the computer won’t be able to able to communicate properly with the hardware devices.

- **Device controller**, on the other hand, is like a bridge between the device and the operating system. It is an **electronic component** consisting of chips which control the device.

http://www.differencebetween.net/technology/difference-between-device-driver-and-device-controller/

I/O Subsystem

- Input/output devices that are connected to the computer are called ***peripheral*** devices.

- It is communicated with the system through the busses: ***Data bus***: to transfer data, ***Address bus***: used to specify address locations, and ***Control bus***: to control a device.

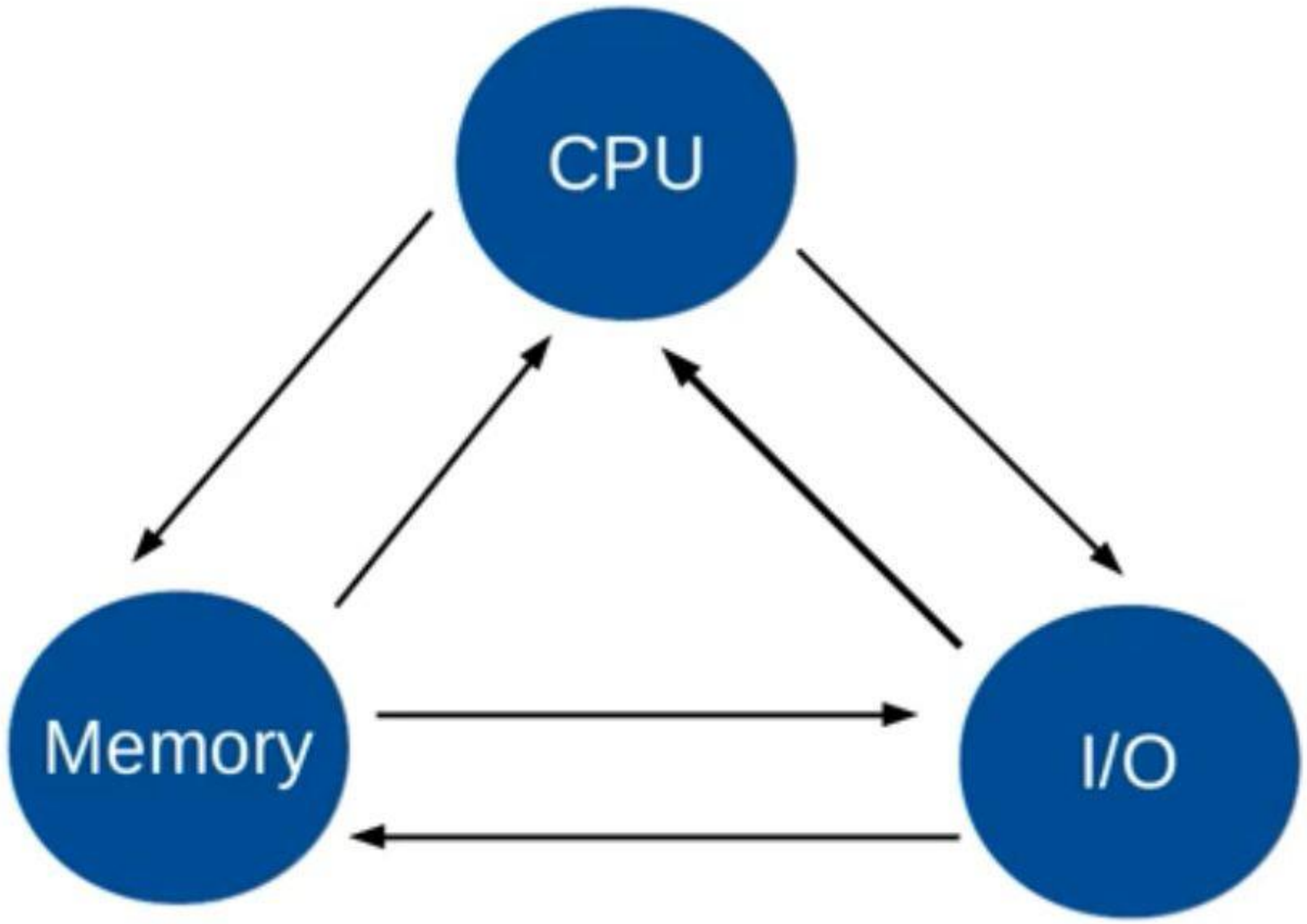
Buses

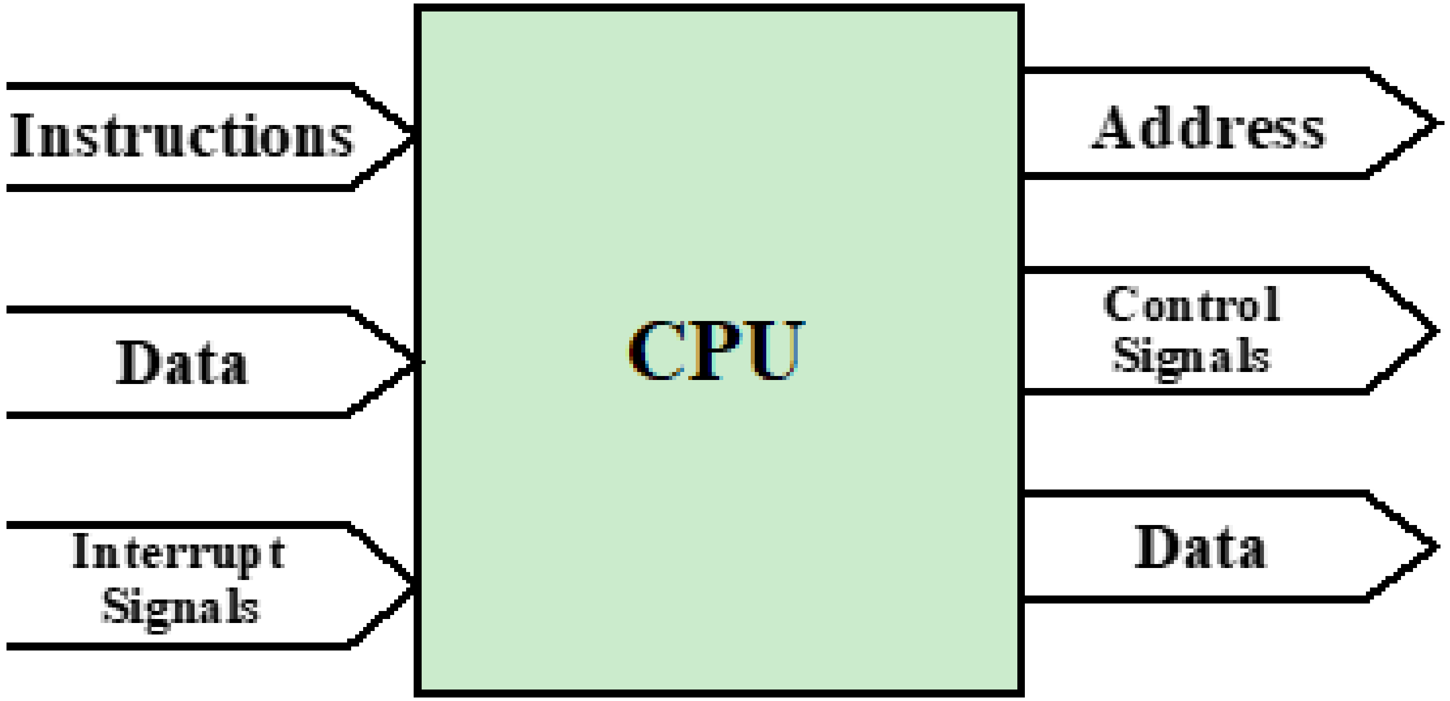
- Transfer data
- Consists of set of lines/path ways
- Each line transfer one bit
- Bits can move in parallel

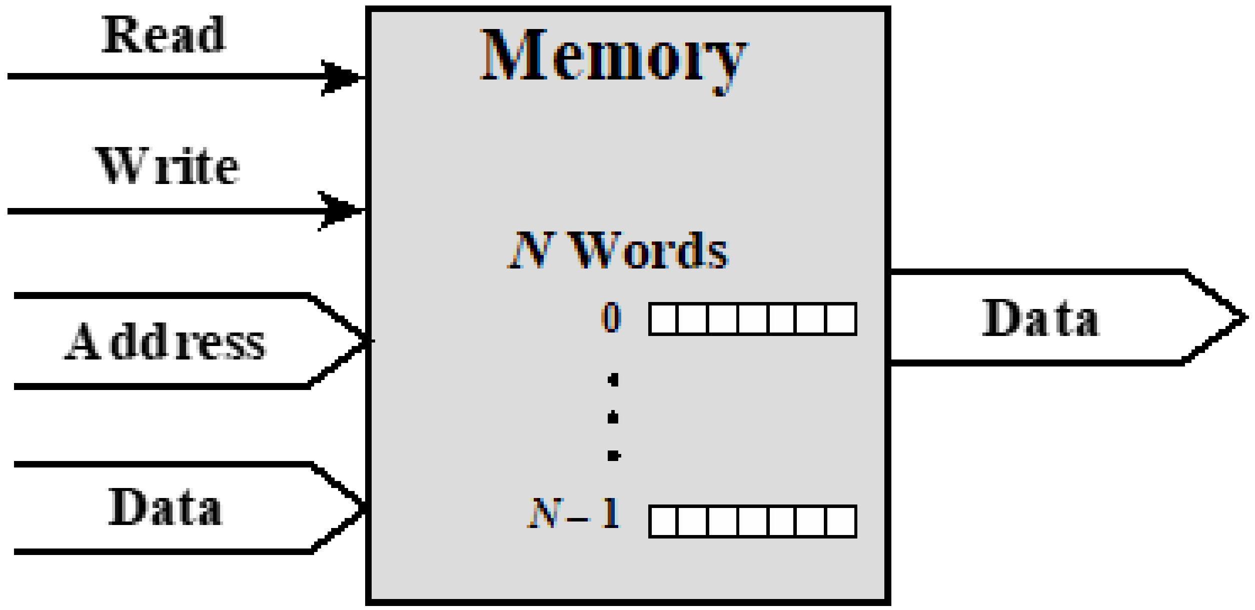
I/O Subsystem (Buses)

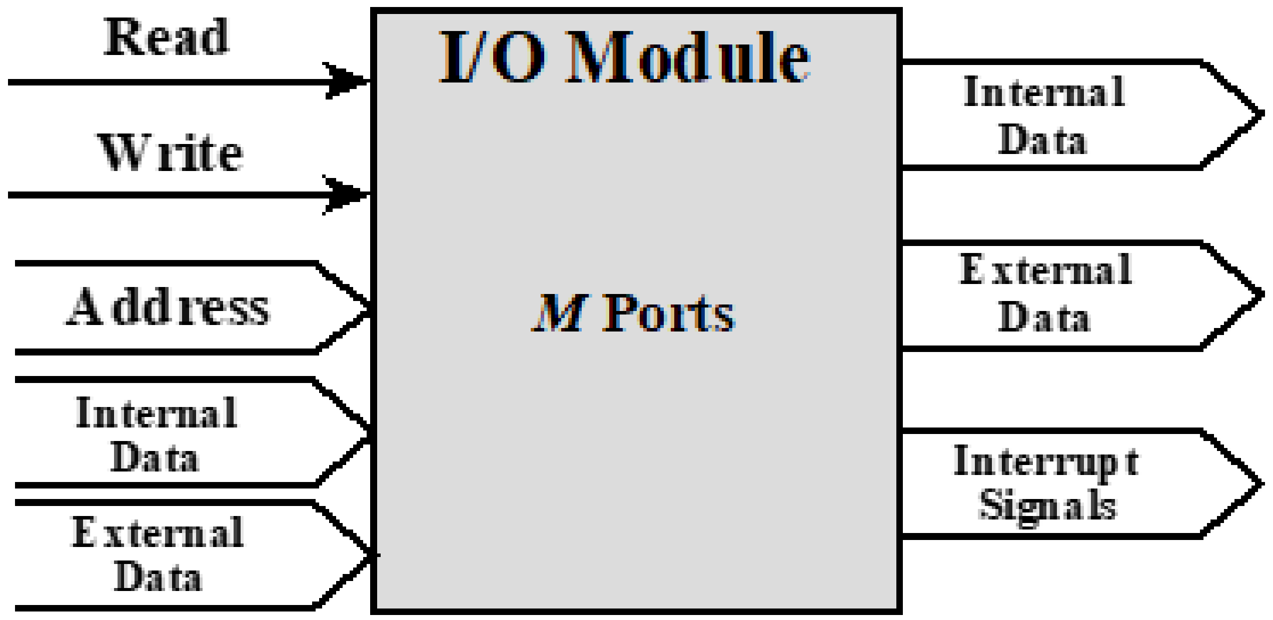
- Buses are the means by which data is transmitted from one part of a computer to another, connecting all major internal components to the CPU and memory.
- A standard CPU system bus is comprised of a control bus, data bus and address bus.

Address Bus	**Carries the addresses of data (but not the data) between the processor and memory**
Data Bus	**Carries data between the processor, the memory unit and the input/output devices**
Control Bus	**Carries control signals/commands from the CPU (and status signals from other devices) in order to control and coordinate all the activities within the computer**


CPU
Memory
I/O


Instructions
Data
Interrupt Signals
CPU
Address
Control Signals
Data


Read
Write
Address
Data
Memory
N Words
0
N – 1
Data


Read
Write
Address
Internal Data
External Data
I/O Module
M Ports
Internal Data
External Data
Interrupt Signals

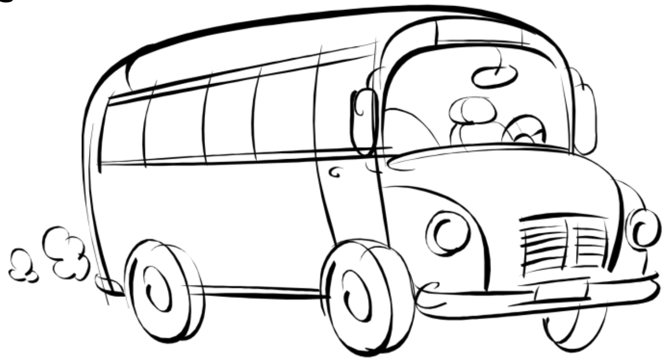
Type of buses

- Data bus
- Address bus
- Control bus

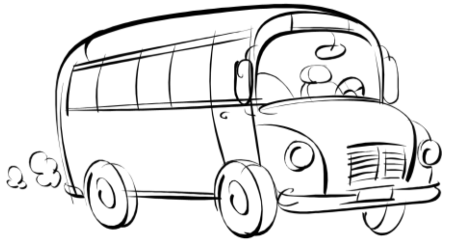
Data Bus

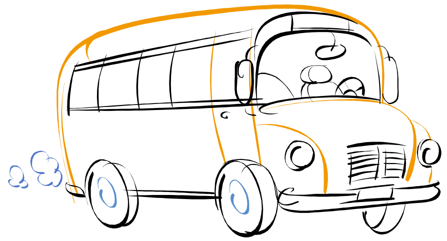
- Consist of set of lines (data bus width)
- If data = 64 bit & bus width = 32 bit

  you need 2 cycles



Address Bus Control Bus





Width represents max accessible memory address
Can hold memory or I/O device address
Higher order bits for module to connect with (memory, I/O)
The remaining part is the address

Module	Address

X1010011
X=0 memory
X= 1 I/O

Used to control the access and the use of the data and address lines
Because the data and address lines are shared by all components there must be a means of controlling their use

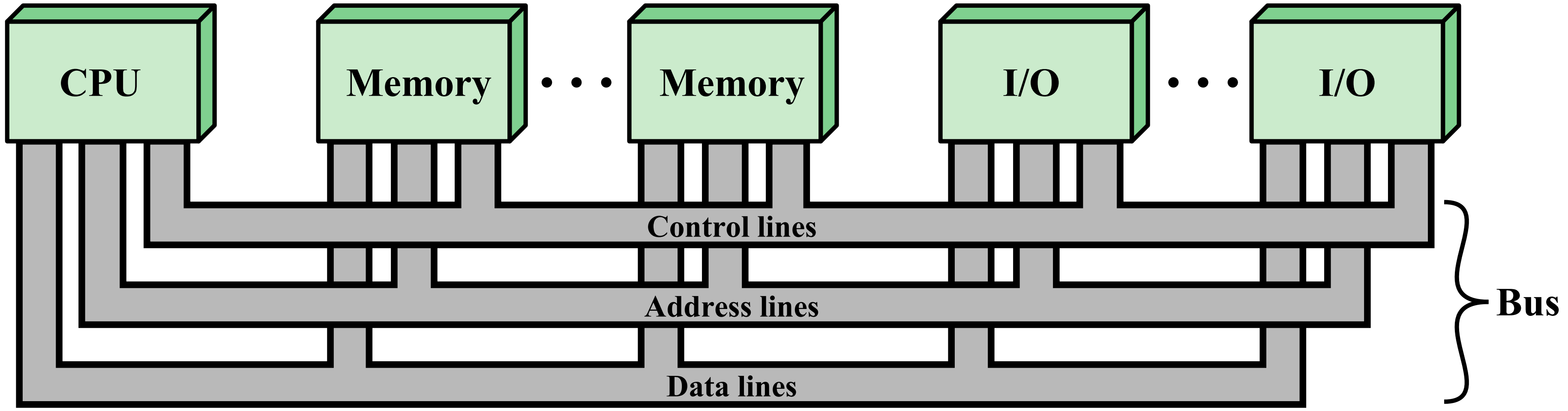


Figure 3.16 Bus Interconnection Scheme

I/O Subsystem

- Typically, there is a software component in kernel mode called as the "***device driver***" that handles all **interfaces** with a device.
- It helps with communicating between the device and the OS and abstracts the device specifics.
- Similarly, there could be a **driver** at the **bus** level usually referred to as
  the bus driver. (Ex: USB Bus driver)
- Most OSs include an inbox driver that implements the bus driver.
- There is usually a driver for each controller and each device.

I/O Devices Categories

- The I/O devices can be broadly divided into two categories called:

block devices and ***character*** devices.

Character Devices

- Another class of devices are character devices, the subtle difference is that the communication happens by sending and receiving single characters, which is usually a byte or an octet.
- Many serial port devices like ***keyboards***, some ***sensor devices***, and microcontrollers follow this mechanism.

Block Devices

- These are devices with which the I/O device controller communicates by sending blocks of data.
- A block is referred to as a group of bytes that are referred together for Read/Write purposes.
- Example: ***flash memory***, ***digital camera***
- The device driver would access by specifying the size of Read/Writes which is varying from device to another.

Synchronization and Critical Sections

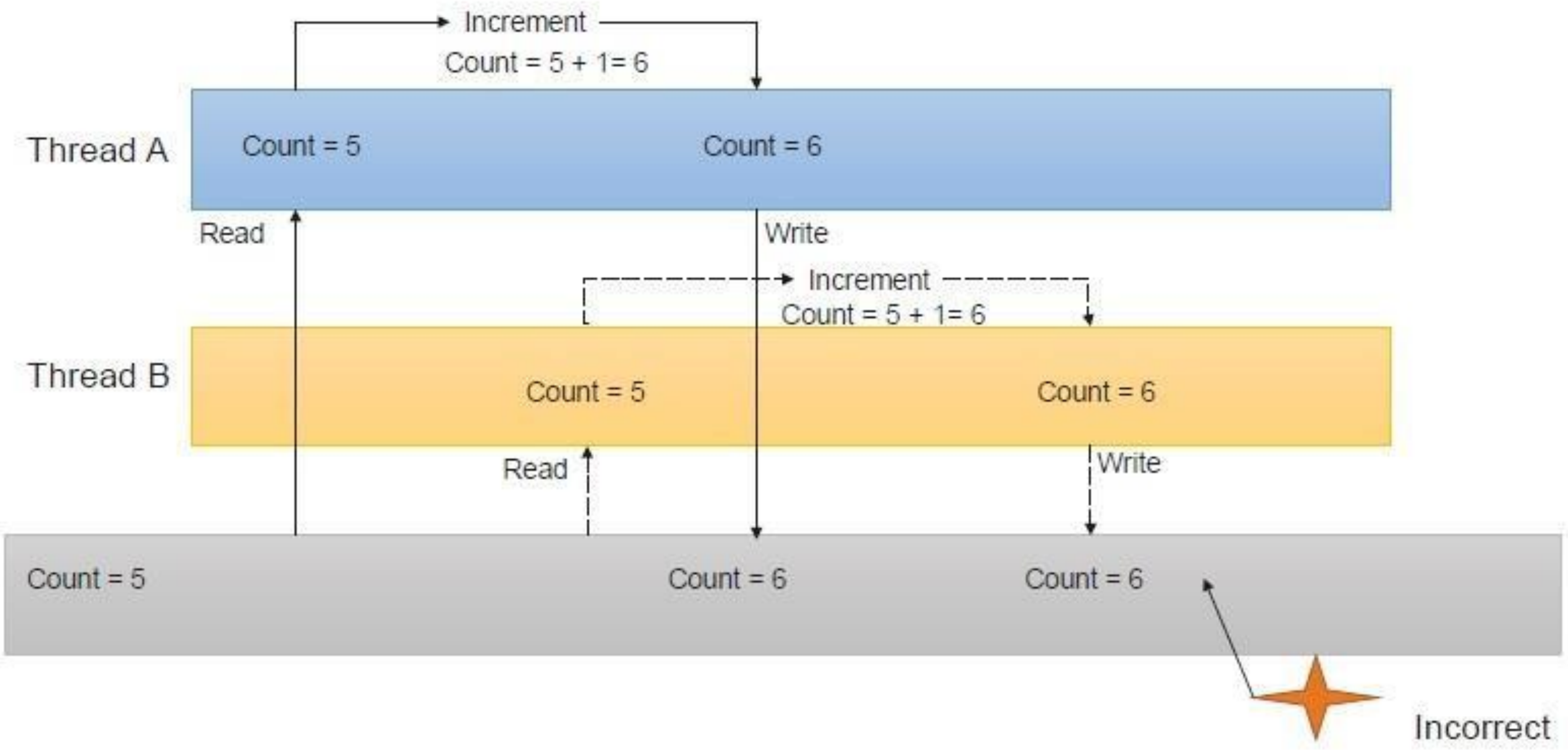
- In multi-threaded applications, if one **thread tries** to **change** the **value** of **shared data** at the same time as another thread tries to read the value, there could be a race condition across threads.

- In this case, the result can be unpredictable.

- The **access** to such **shared variables** via shared memory, files, ports, and other I/O resources needs to be **synchronized** to protect it from being corrupted.

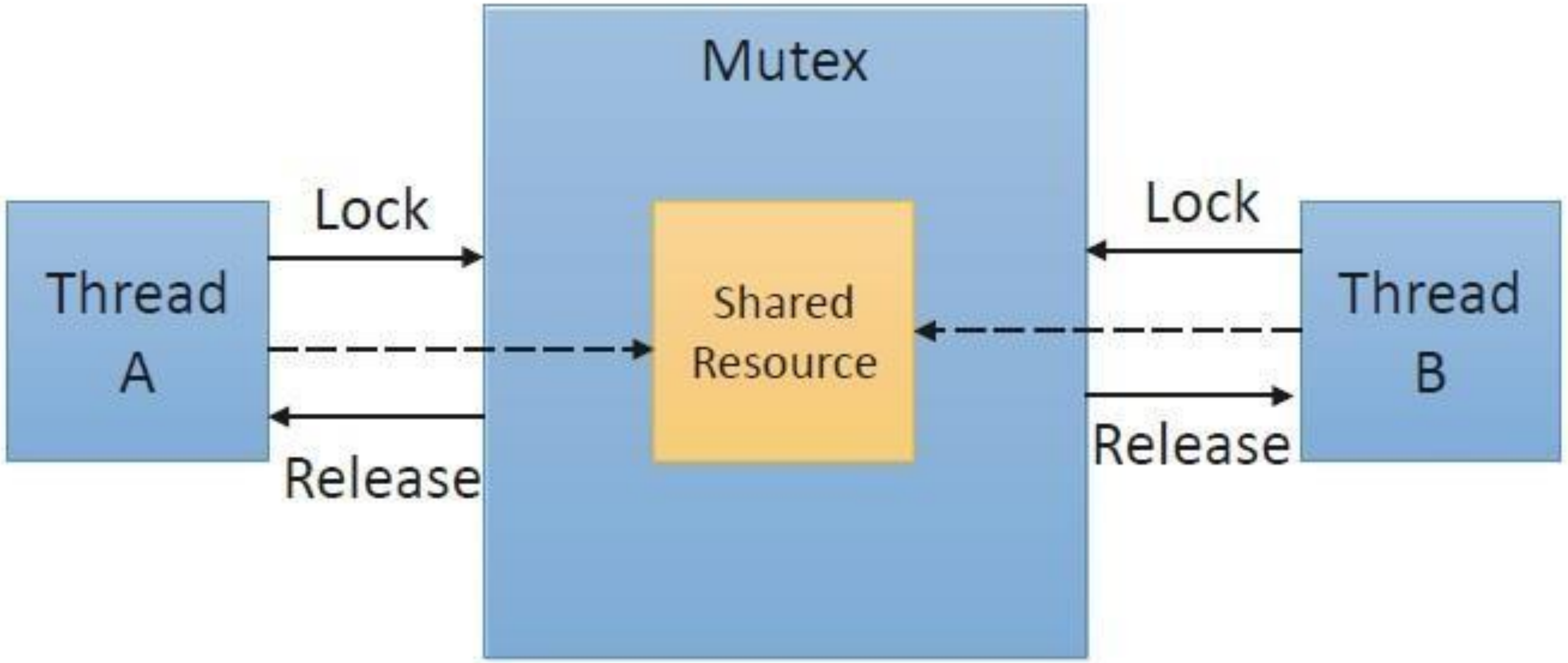
- order to support this, the operating system provides mutexes and semaphores to coordinate access to these shared resources.

Synchronization and Critical Sections



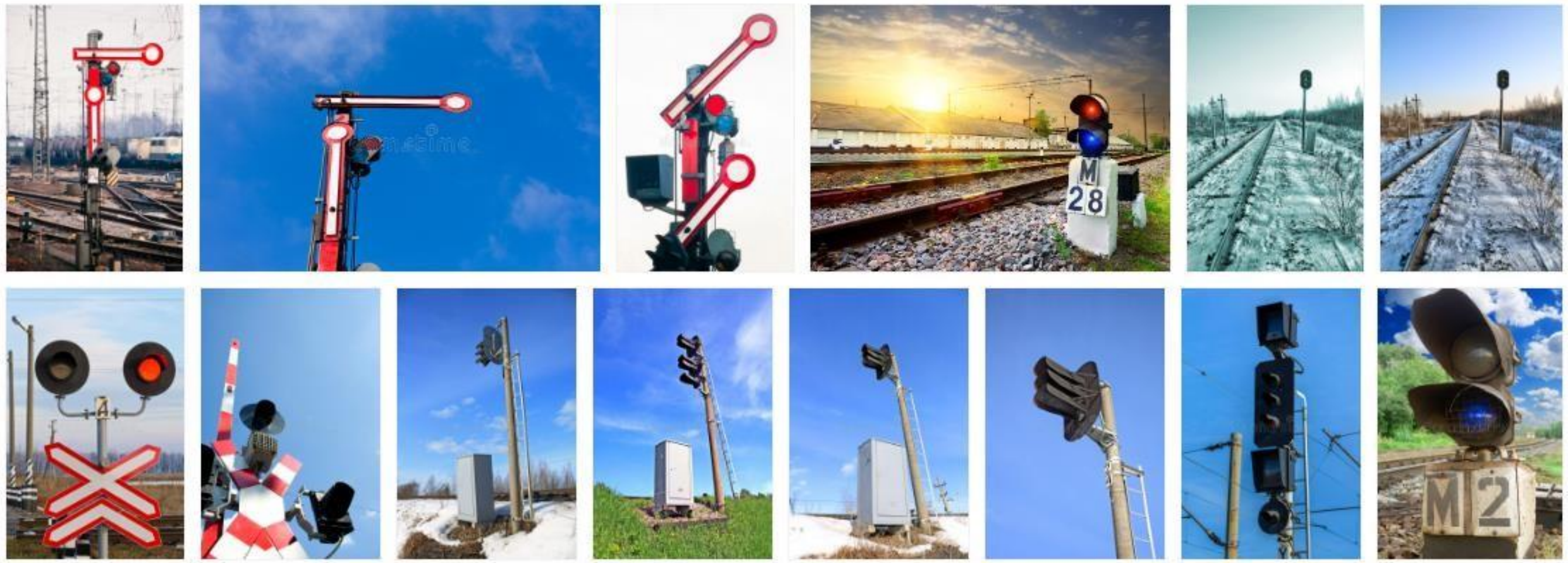
Mutex

- A mutex is used for implementing ***mutual exclusion***: either of the participating processes or threads can have the key (mutex) and proceed with their work.
- The other one would have to wait until the one holding the mutex finishes.



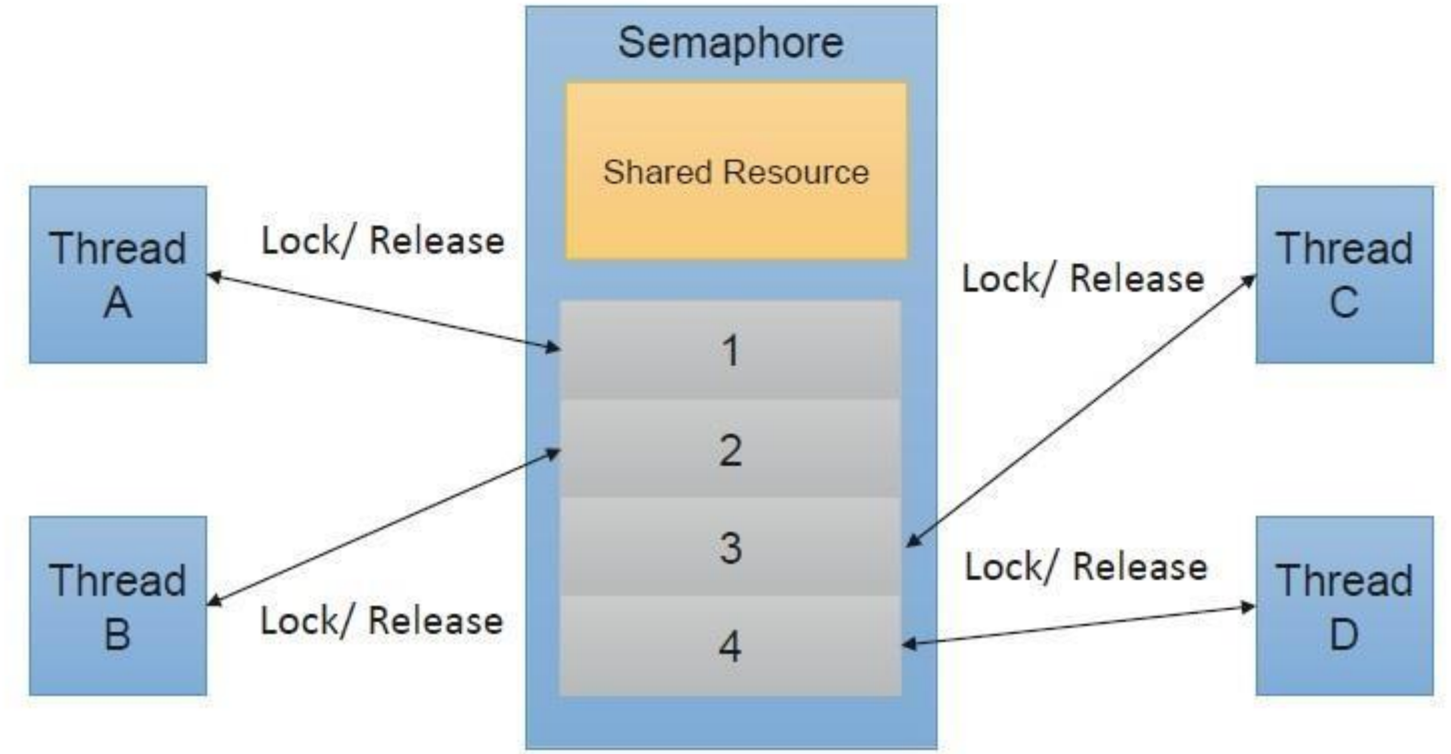
Semaphore

- A semaphore is a generalized mutex. A binary semaphore can assume a value of 0/1 and can be used to perform locks to certain critical sections.



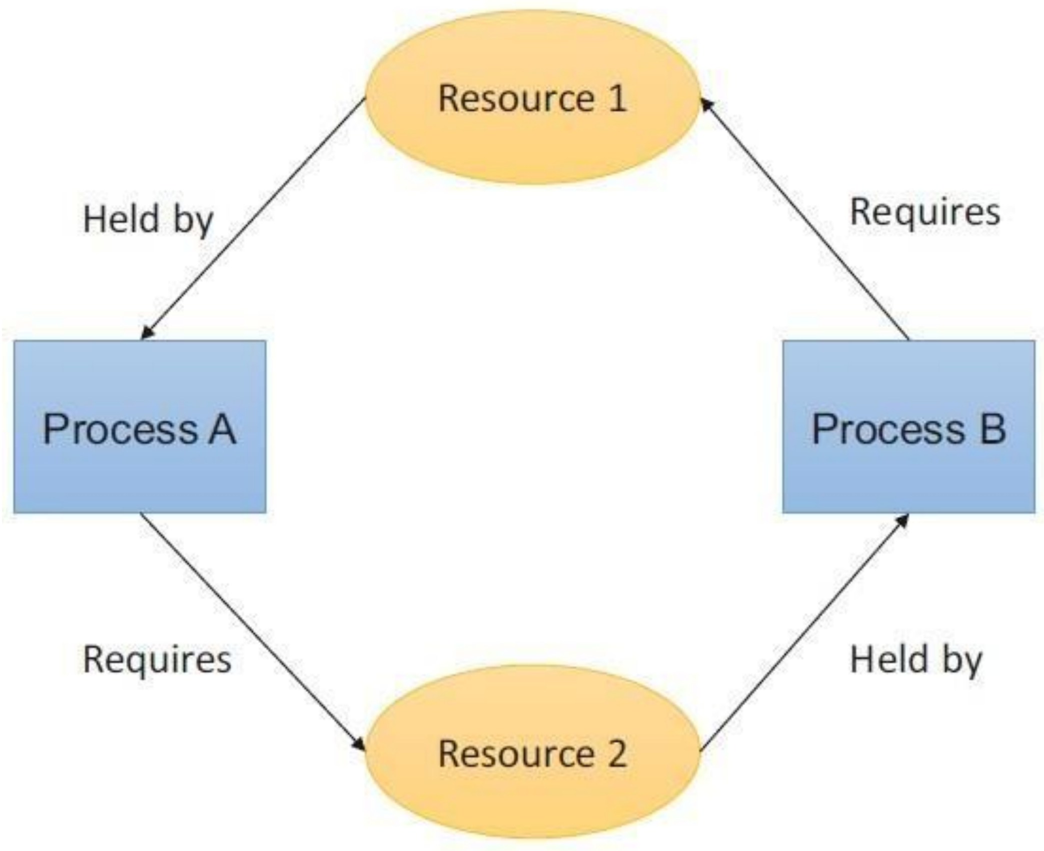
Semaphore

- A semaphore is a generalized mutex. A binary semaphore can assume a value of 0/1 and can be used to perform locks to certain critical sections.



Deadlocks

- When a set of processes become blocked because each process is holding a resource and waiting for another resource acquired by some other process. This is called as a ***deadlock***.
- Process A holds Resource 1 and requires Resource 2. However, Process B already is holding Resource 2, but requires Resource 1.Unless either of them releases their resource, neither of the processes may be able to move forward with the execution.



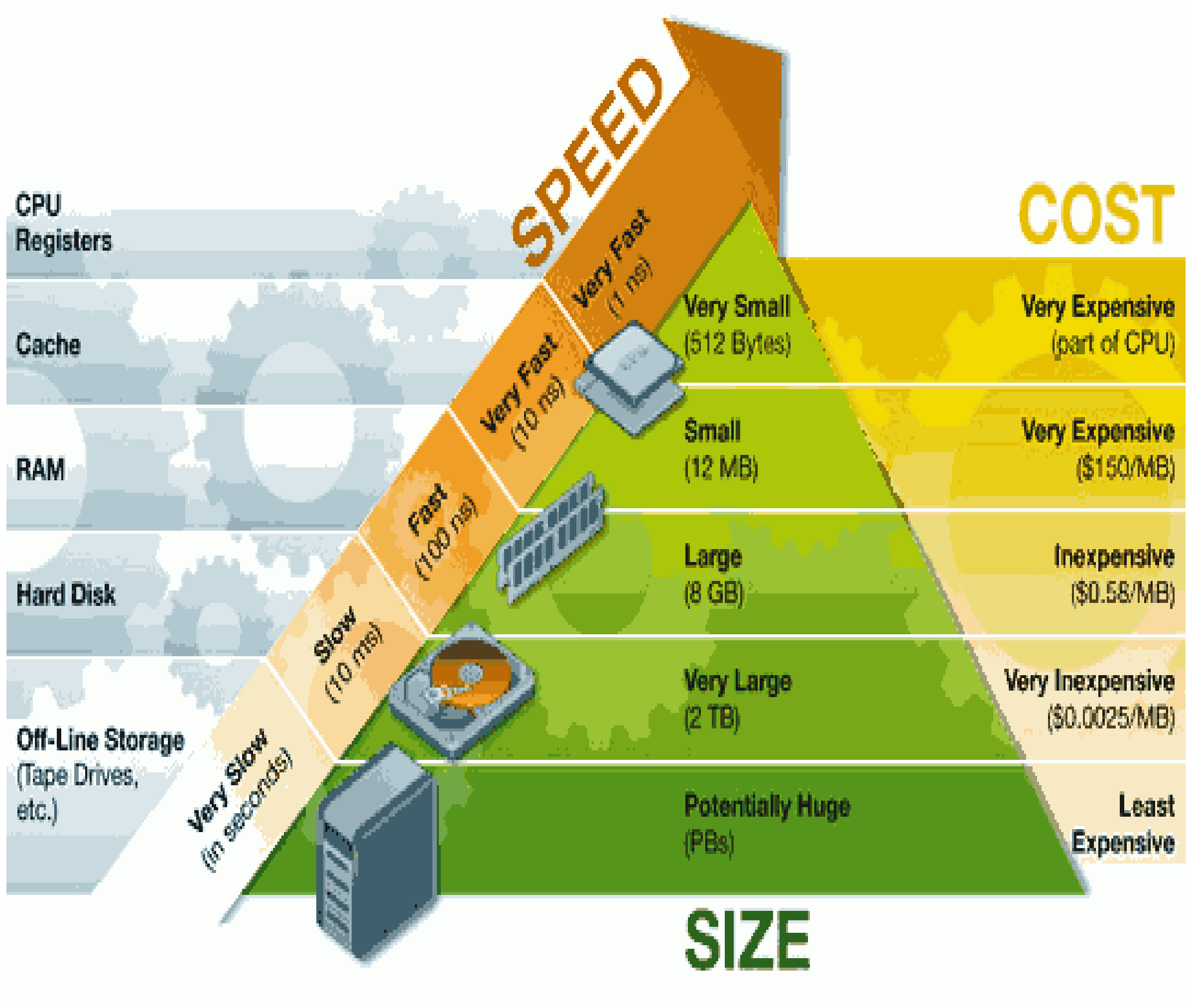
Deadlocks

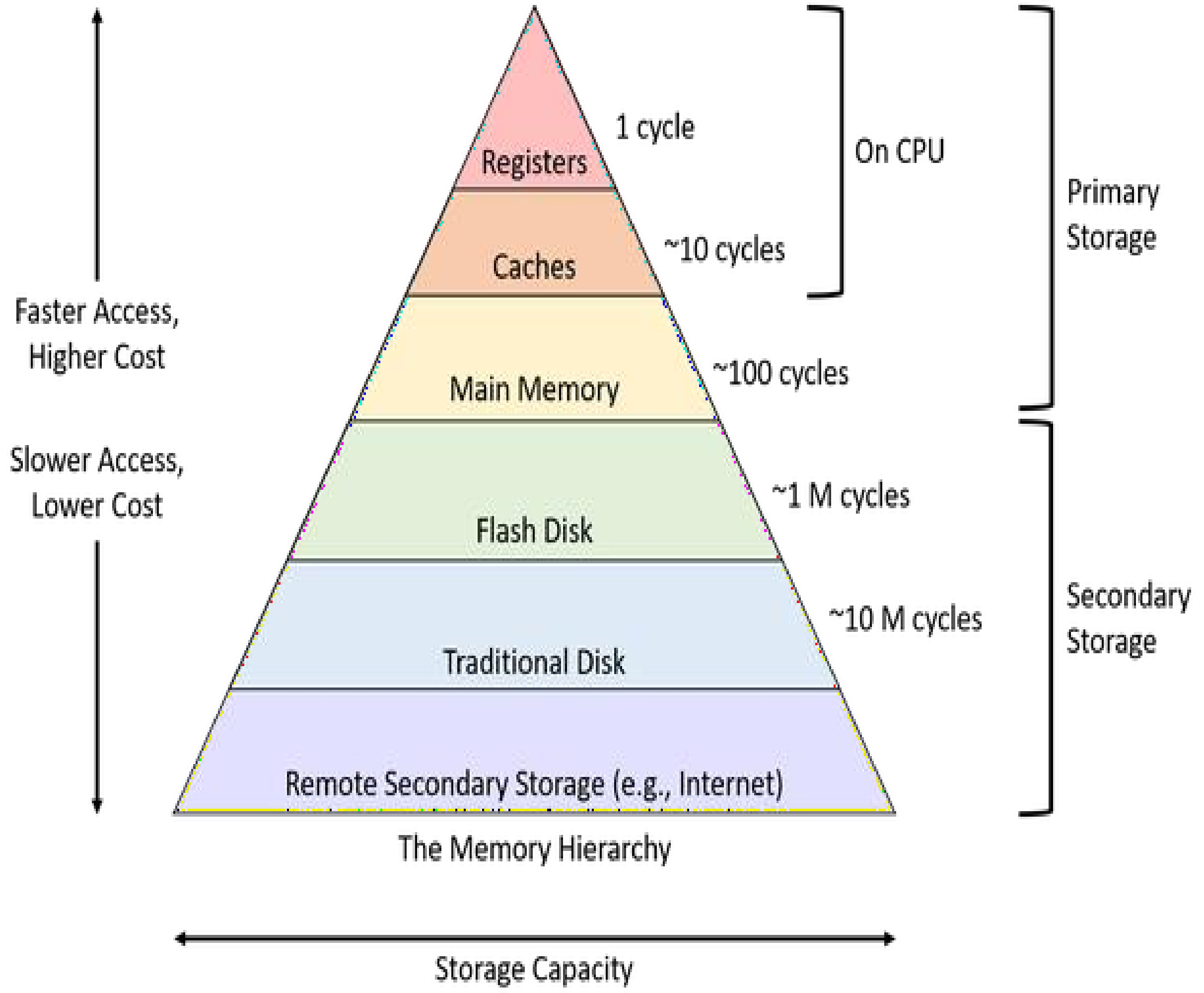
- A deadlock can arise if the following four conditions hold:
  - **Mutual Exclusion**: There is at least one resource on the system that is not shareable. This means that only one process can access this at any point in time. In the preceding example, Resources 1 and 2 can be accessed by only one process at any time.
  - **Hold and Wait**: A process is holding at least one resource and is waiting for other resources to proceed with its action. In the preceding example, both Processes A and B are holding at least one resource.
  - **No Preemption**: A resource cannot be forcefully taken from a process unless released automatically.
  - **Circular Wait**: A set of processes are waiting for each other in circular form.

Memory Management

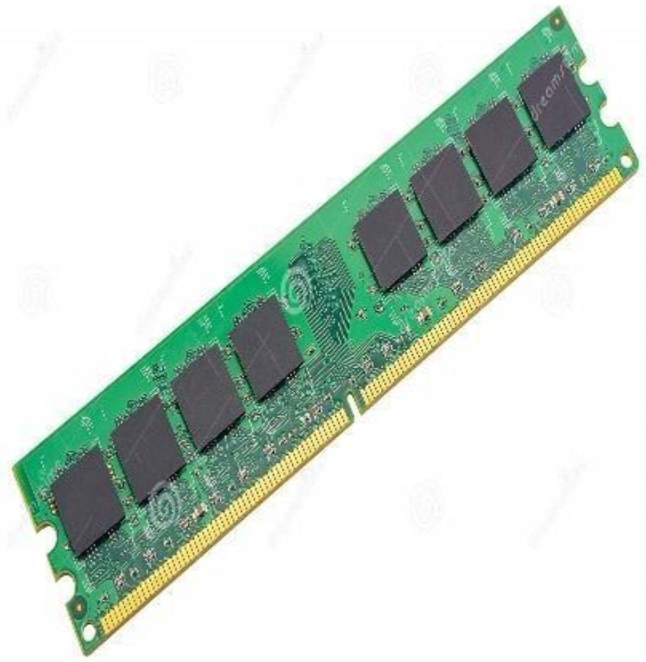
RAM is the main memory in a computer.

Storage-Device (memory) Hierarchy





RAM is the main memory in a computer.



- Array of memory words
- Each byte has an address
- Memory Address:
  - Physical
  - Logical
- CPU Instructions:
  - Load: moves a word from main memory to CPU register
  - Store: move a word from CPU register to main memory

Memory Unit

- The memory unit consists of RAM (Random Access Memory) and ROM (Read Only Memory), sometimes referred to as primary or main memory .
- Unlike a hard drive (secondary memory), this memory is fast and also directly accessible by the CPU.
- RAM is split into partitions (bytes). Each partition consists of an address and its contents (both in binary form).
- The address will uniquely identify every location (byte) in the memory.
- Loading data from permanent memory (secondary storage or hard drive), into the faster and directly accessible temporary memory (RAM), allows the CPU to operate much quicker.

Main-Memory Management

- Memory is a large array of bytes, each with its own address. It is a repository of quickly accessible data shared by the CPU and I/O devices.
- Main memory **is a volatile storage device**. It loses its contents in the case of system failure.
- **The operating system is responsible for** the following activities in connections with memory management:
  - Keep track of which parts of memory are currently being used and by whom.
  - Decide which processes to load when memory space becomes available.
  - Allocate and de-allocate memory space as needed

Need of Memory Management

- **In systems with multiple programs running in parallel**, there could be **many processes in memory at the same time**, and **each process may have specific memory needs**.

- **The OS (kernel component) may also need to be loaded in memory**. Additionally, there may be a specific portion of memory needed for specific devices (Ex: printer spooling).

Spooling in Operating System

- The CPU executes the instructions and deal with many processes, and we know that the time taken in the I/O operation is very large compared to the time taken by the CPU for the execution of the instructions.

- SPOOLING: **is used for the purpose of copying data between different devices**. In modern systems it is usually used for mediating between a computer application and a slow peripheral, such as a printer.

Ref: https://computersolve.com/what-does-spooling-mean-on-printer/
Ref: https://www.youtube.com/watch?v=Nk08Tb9qDkQ
Ref: https://www.javatpoint.com/spooling-in-operating-system

Printer spooling

- the CPU executes the instructions and deal with many processes, and we know that the time taken in the I/O operation is very large compared to the time taken by the CPU for the execution of the instructions.
- SPOOLING: is used for the purpose of copying data between different devices. In modern systems it is usually used for mediating between a computer application and a slow peripheral, such as a printer.

- **Spooling refers to** the **time that it takes to move information from a program or document to your printer**, which must have been turned on and connected

- The spooling process usually happens in the background, and can sometimes **take several minutes** if you're **printing** an especially **large document** or image.

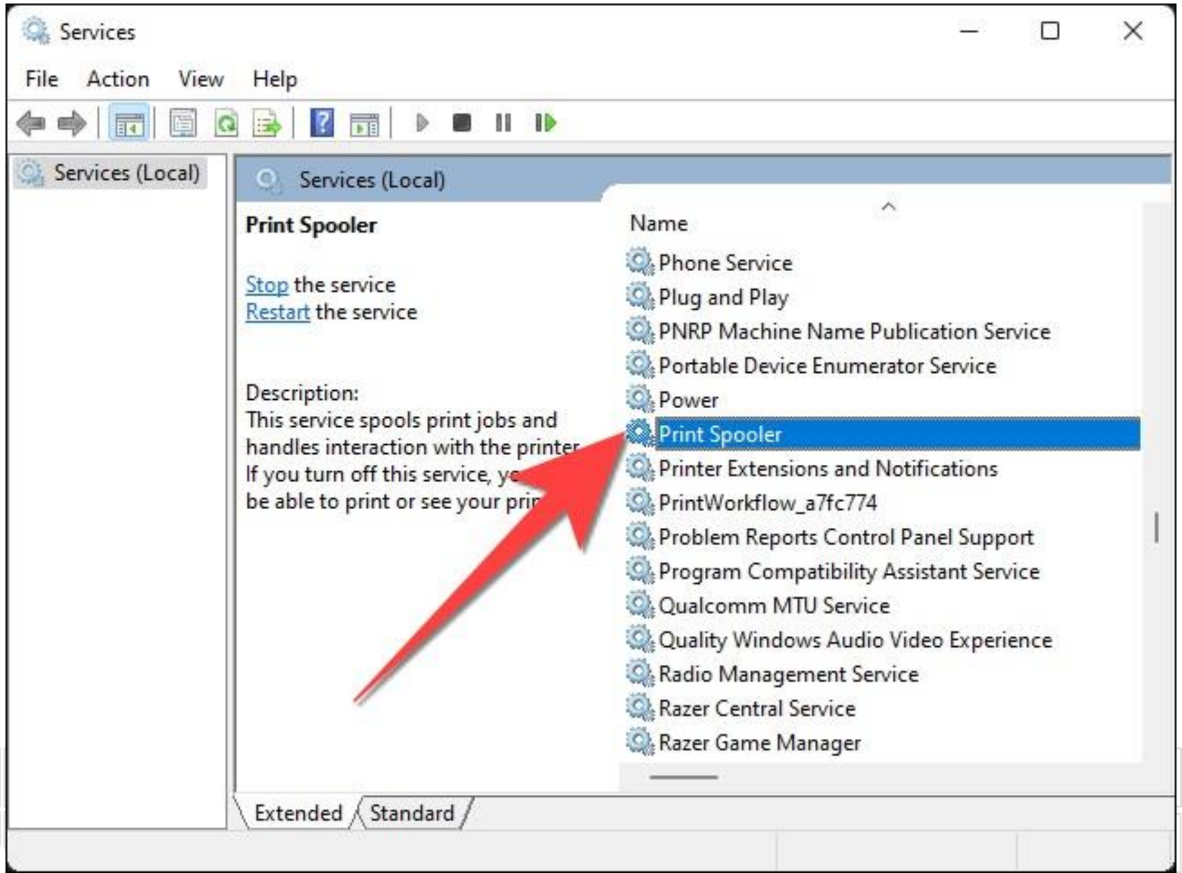
Ref: https://computersolve.com/what-does-spooling-mean-on-printer/
Ref: https://www.youtube.com/watch?v=Nk08Tb9qDkQ
Ref: https://www.javatpoint.com/spooling-in-operating-system

Printer spooling (Advantages)

- **No Waste if Printing Many Documents at Once**: It also allows you to print more than one document at a time (i.e. multiple files).

- **Productivity Increases**: Users can continue working as print jobs are running in the background. This helps increase productivity because users don't have to wait for a job to complete before moving on to another task.

- **Minimizing Errors**: Spooling prevents errors that arise during printing if you send another document or command before the printer finishes its current task.



Print Spooler

Stop the service
Restart the service

Description:
This service spools print jobs and handles interaction with the printer. If you turn off this service, yo be able to print or see your prin

Phone Service
Plug and Play
PNRP Machine Name Publication Service
Portable Device Enumerator Service
Power
Print Spooler
Printer Extensions and Notifications
PrintWorkflow_a7fc774
Problem Reports Control Panel Support
Program Compatibility Assistant Service
Qualcomm MTU Service
Quality Windows Audio Video Experience
Radio Management Service
Razer Central Service
Razer Game Manager

Logical Address Vs. Physical Address

- **The program each time executed it is loaded in a different memory location** (according to the available spaces at time loading).
- And so, when the process in waiting state for I/O operation it may be swapped out from main memory to virtual memory (part from secondary storage), and when the waiting state changed to ready, the process must swapped in to main memory which – almost cases- another location in the main memory.
- That means the addresses of the variables which used in the program, may be changed many times at the runtime!!!
- To solve this problem, the common solution is to *map* the program's *compiled* addresses to the actual address in *physical* memory.

Logical Address Vs. Physical Address

- A group of **several** logical address is referred to a **logical address space**. The logical address is basically used as a reference to access the physical memory locations.

- The **logical** address of a program is **visible** to the computer user.

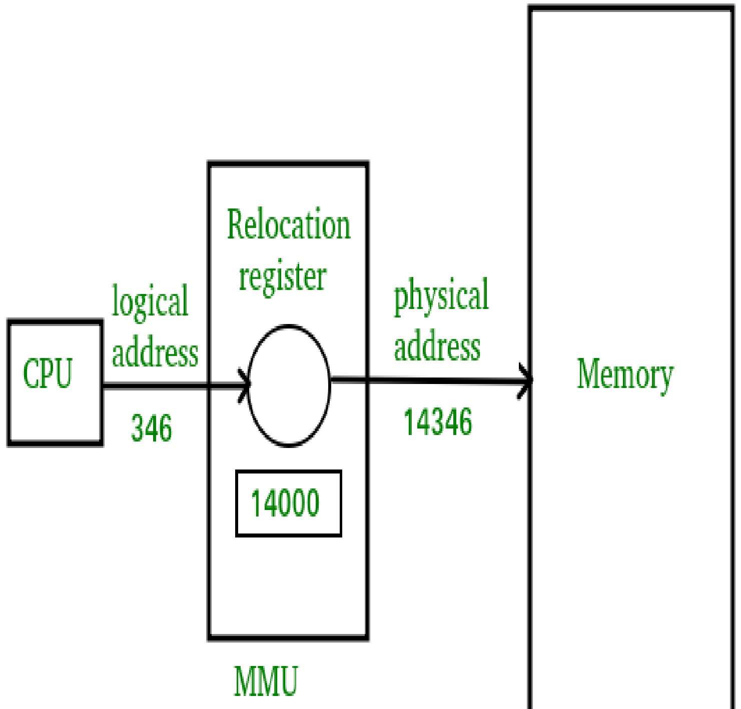
- The **physical** address is **not visible** to the computer user.

Logical Address Vs. Physical Address

- A program will have variables and … etc. When the program gets compiled, the compiler translates the addresses into relative addresses (Logical Address).

- This is important for the OS to then load the program in memory through a physical addresses

- The mapping from logical address to physical address is done by the memory management unit (MMU) which is a hardware device



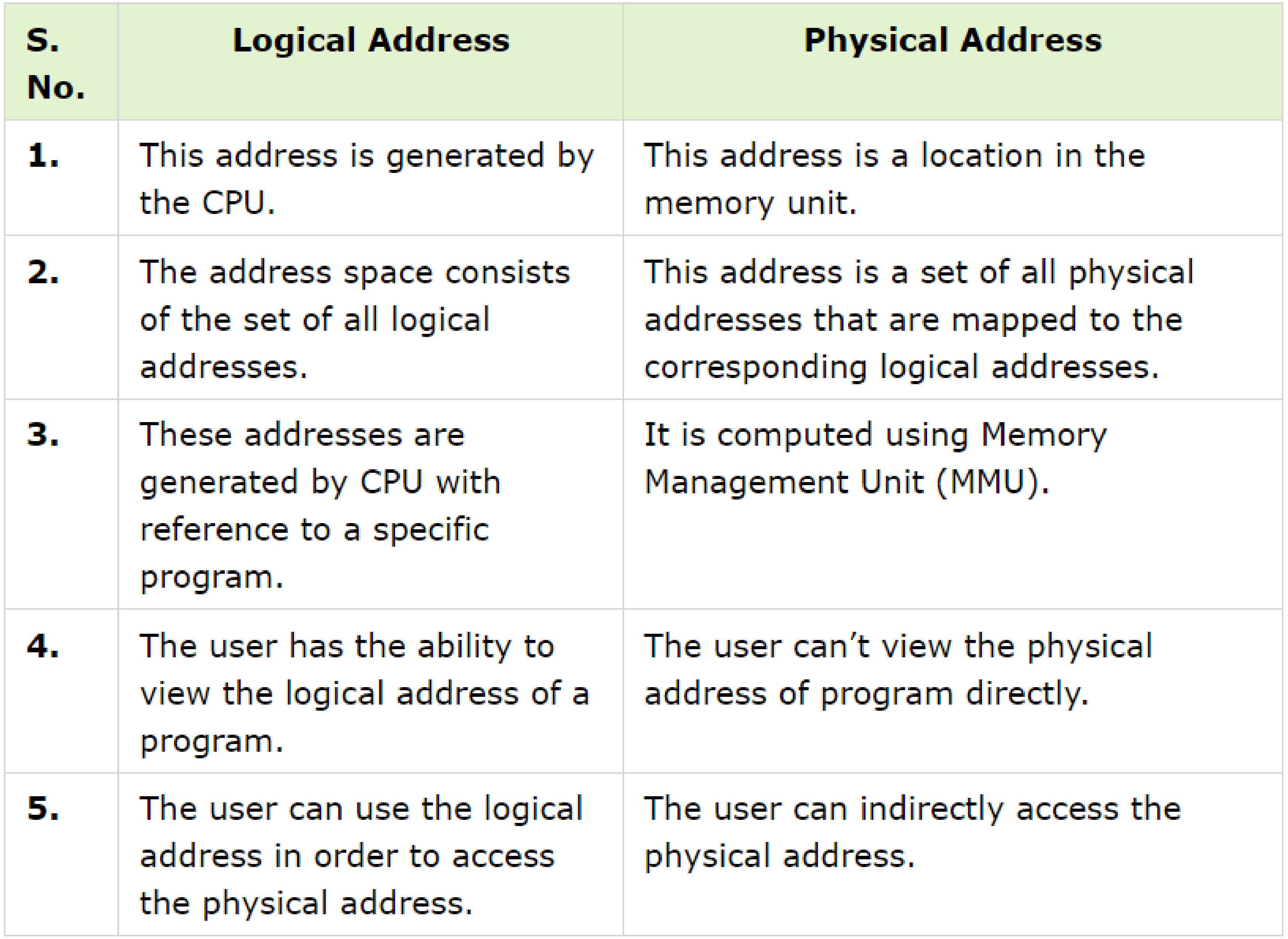
Ref: https://www.geeksforgeeks.org/memory-allocation-techniques-mapping-virtual-addresses-to-physical-addresses/

Logical Address Vs. Physical Address

- **the process of mapping from one address space to another address space** is called **Address binding**

- Logical address is an address generated by the CPU during execution, whereas Physical Address refers to the location in the memory unit(the one that is loaded into memory).

Logical Address Vs. Physical Address



S. No.	Logical Address	Physical Address
1.	This address is generated by the CPU.	This address is a location in the memory unit.
2.	The address space consists of the set of all logical addresses.	This address is a set of all physical addresses that are mapped to the corresponding logical addresses.
3.	These addresses are generated by CPU with reference to a specific program.	It is computed using Memory Management Unit (MMU).
4.	The user has the ability to view the logical address of a program.	The user can’t view the physical address of program directly.
5.	The user can use the logical address in order to access the physical address.	The user can indirectly access the physical address.

Thank You