Operating Systems

Quiz

Q. Critical Section Problem can be resolved by using

A. Binary Semaphore
B. Mutex Object
C. Classic Semaphore
D. Both A & B
E. None of the above

Q. Which of the following signal an OS send to a process for forcefull termination.

A. SIGTERM
B. SIGEND
C. SIGSTOP
D. SIGKILL

Quiz

Q Processes which shares data with another processes referred as

A. related processes
B. cooperative processes
C. independent processes
D. all of the above
E. none of the above

Q. Which of the following IPC mechanism is used for communication across the systems?

A. Pipe
B. message queue
C. chatting application
D. socket
E. shared memory model

Quiz

Q. What is ‘Aging’?

a) keeping track of cache contents
b) keeping track of what pages are currently residing in memory
c) keeping track of how many times a given page is referenced
d) increasing the priority of jobs to ensure termination in a finite time

Q. The segment of code in which the process may change common variables, update tables, write into files is known as ______________

a) program
b) critical section
c) non – critical section
d) synchronizing

Quiz

Q. If the semaphore value is negative ______________

a) its magnitude is the number of processes waiting on that semaphore
b) it is invalid
c) no operation can be further performed on it until the signal operation is performed on it
d) none of the mentioned

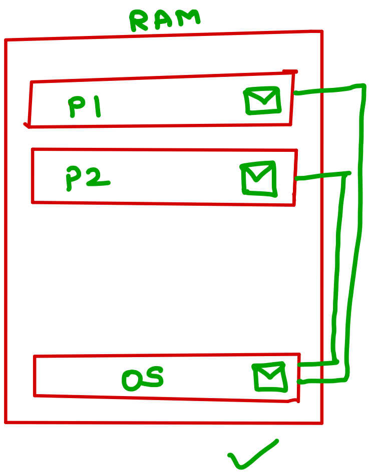
Q. Which of the following is not a CPU scheduling criteria?

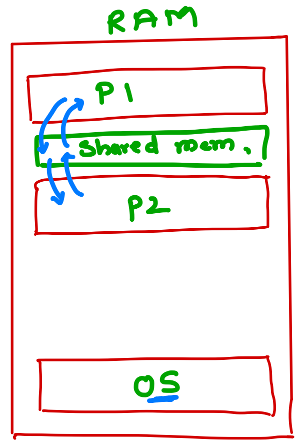
A. Waiting Time
B. Response Time
C. CPU Burst Time
D. Turn-Around-Time

IPC 2 Type :

① Message Passing

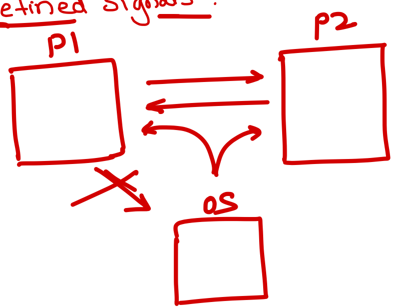
(2) Shared Memory





① Signals

- predefined signals.



64 signals..

② SIGINT : cntrl + C → process terminated.

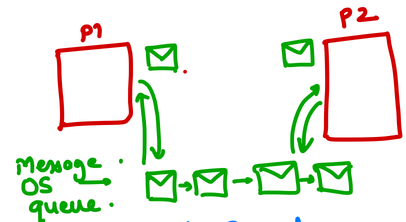
⑨ SIGKILL : Shutdown.

⑪ SIGSEGV : dangling pointer

Segment violation.

ie seg fault → abnormal terminate.

② Message queue :



→ Packet Based.
→ Bi-directional

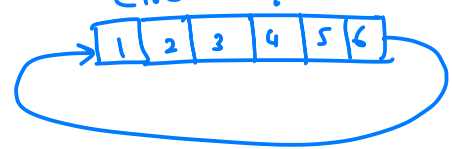
③ Pipe

. terminal> Command 1 | command 2

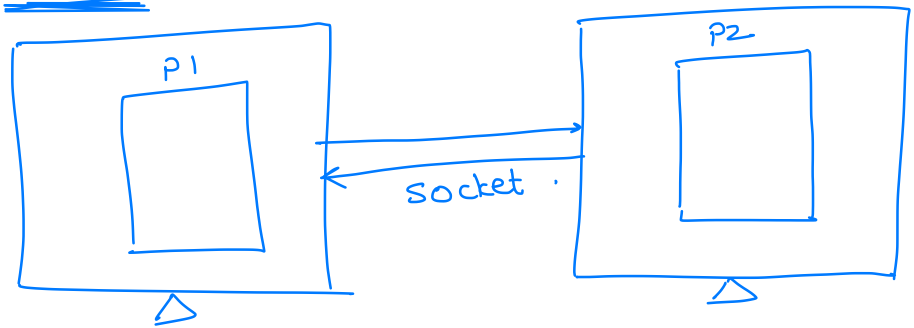
output
pipe.
input.

→ Stream based.
→ Uni-directional.

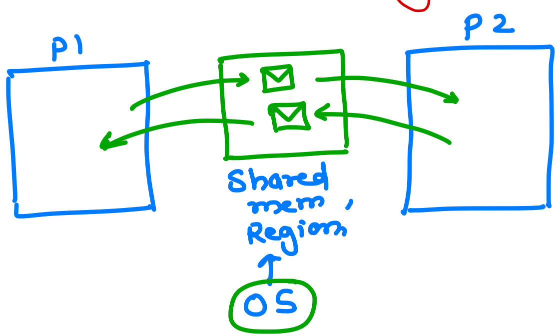
Circular queue.



④ Socket



⑤ Shared Memory



→ Shared. mem is fastest IPC mech.

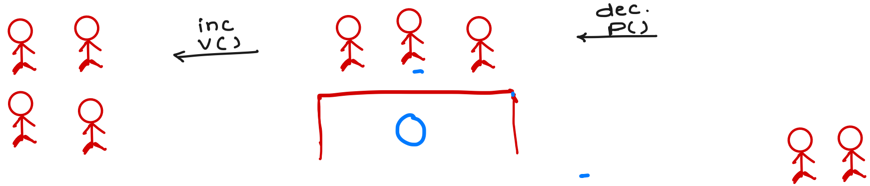
Semaphore

- Semaphore was suggested by Dijkstra scientist (dutch math)
- Semaphore is a counter
- On semaphore two operations are supported:
- wait operation: decrement op: P operation:
  - semaphore count is decremented by 1.
  - if cnt < 0, then calling process is blocked(block the current process).
  - typically wait operation is performed before accessing the resource.
- signal operation: increment op: V operation:
  - semaphore count is incremented by 1.
  - if one or more processes are blocked on the semaphore, then wake up one of the process.
  - typically signal operation is performed after releasing the resource.
- Q. If sema count = -n, how many processes are waiting on that semaphore?
  - Answer: "n" processes waiting

Semaphore:

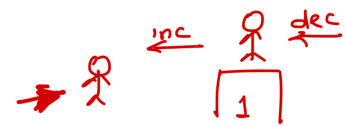
Semaphone is a counter

cnt< 0
↳ process - block.



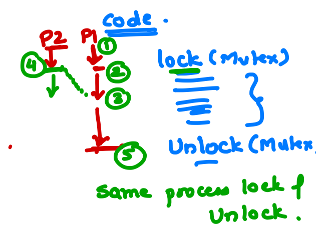
If semaphore count is -n, number of waiting processes are n.

Semaphore → Counting/Classic — when multiple processes can access a resource at a time.
Semaphore → Binary(0 or 1).
↳ when a single process can access a resource (at a time).



Mutex (Mutal Exclusion): when only one process access resource. Like Binary semaphore.

Mutex is not counter. It has two states, ie Unlocked & locked state.



Semaphore

- Counting Semaphore/classic
  - When multiple processes can access a resource.
  - Allow "n" number of processes to access resource at a time.
  - Or allow "n" resources to be allocated to the process.
- Binary Semaphore
  - When a single process can access a resource at a time.
  - Allows only 1 process to access resource at a time or used as a flag/condition

Mutex

Mutex(Mutual Exclusion)

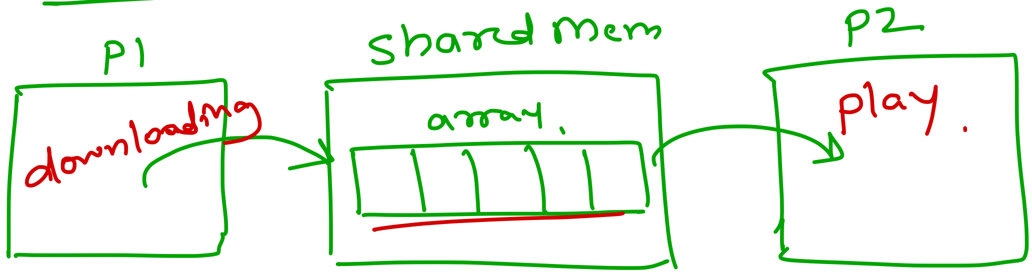
- When only one process access resource like binary semaphore
- Mutex has two states ie. Unlocked or locked state.
- Rule of mutex is that which process has lock the mutex can only unlock the mutex.

Semaphore vs Mutex

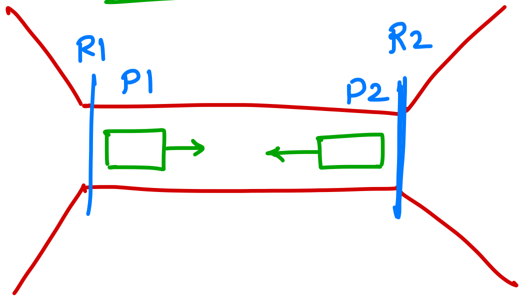
- S: Semaphore can be decremented by one process and incremented by same or another process. M: The process locking the mutex is owner of it. Only owner can unlock that mutex.
- S: Semaphore can be counting or binary.
  M: Mutex is like binary semaphore. Only two states: locked and unlocked.
- S: Semaphore can be used for counting, mutual exclusion or as a flag.

  M: Mutex can be used only for mutual exclusion.

Bounded Shared



Deadlock.



Process → Resource.
↳ OS

1. Resource allocation graph.
2. Banker's algoth

The process are blocked indefintely indeadlock.

4 characteristic. Deadlock

1. No pre-emption. →
2. No Mutual Exclusion
3. Hold & wait.
4. Circular queue.

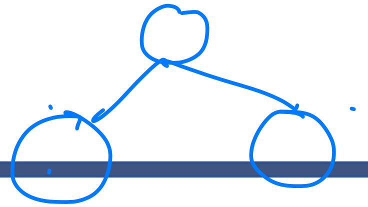
Deadlock

- The processes are blocked indefinitely in deadlock
- Deadlock occurs when four conditions/characteristics hold true at the same time.
  - No preemption: A resource should not be released until task is completed.
  - Mutual exclusion: Resources is not sharable.
  - Hold & Wait: Process holds a resource and wait for another resource.
  - Circular wait: Process P1 holds a resource needed for P2, P2 holds a resource needed for P3 and P3 holds a resource needed for P1.

- **Deadlock Prevention**
  - OS system are designed so that at least one deadlock condition is never true.
  - This will prevent deadlock

.

Deadlock



Deadlock Avoidance

- **The processes should inform system about the resources it needs later. OS will accept or reject resource request based on deadlock possibility.**
- **Algorithms used for this are:**
  - Resource allocation graph: OS maintains graph of resources and processes. A cycle in graph indicate circular wait will occur. In this case OS can deny a resource to a process.
  - Banker's algorithm: A bank always manage its cash so that they can satisfy all customers.

- **Deadlock Recovery**
  - Deadlock can be solved by resource preemption or terminating one of the process (involved in deadlock).

Resource pre-emption -> forcibly withdraw resources given to the processes.

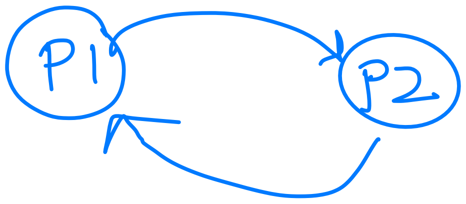
Process termination -> forcibly kill one of the process.

Victim Process.

Deadlock

Starvation vs Deadlock

- Starvation: The process not getting enough CPU time for its execution.
  - Process is in ready state/queue.
    Reason: Lower priority (CPU is busy in executing high priority process).
- Deadlock: The process not getting the resource for its execution.
  - Process is in waiting state/queue indefinitely.
    Reason: Resource is blocked by another process (and there is circular wait).



Memory.

Store data (digit) → Binary formate.

Bit → Binary digit.

Byte → KB, MB, TB, GB, HB, PB, ZB etc.

1 Byte → 8-bit

Mem

- Primary
  ↓
  directly accessible CPU
  eg:- CPU reg, cache, RAM
- Secondary → Indirect accessible.
  eg:- Harddisk.

Computer Fundamental

1 Reg = 8-bit
16 bit
32-bit

Memory

- Memory holds (digital) data or information.
  - Bit = Binary Digit (0 or 1) --> Internally it is an electronic circuit i.e. FlipFlop.
    1 Byte = 8 Bits
    B, KB (2^10), MB (2^20), GB (2^30), TB (2^40), PB (2^50), XB (2^60), ZB (2^70)

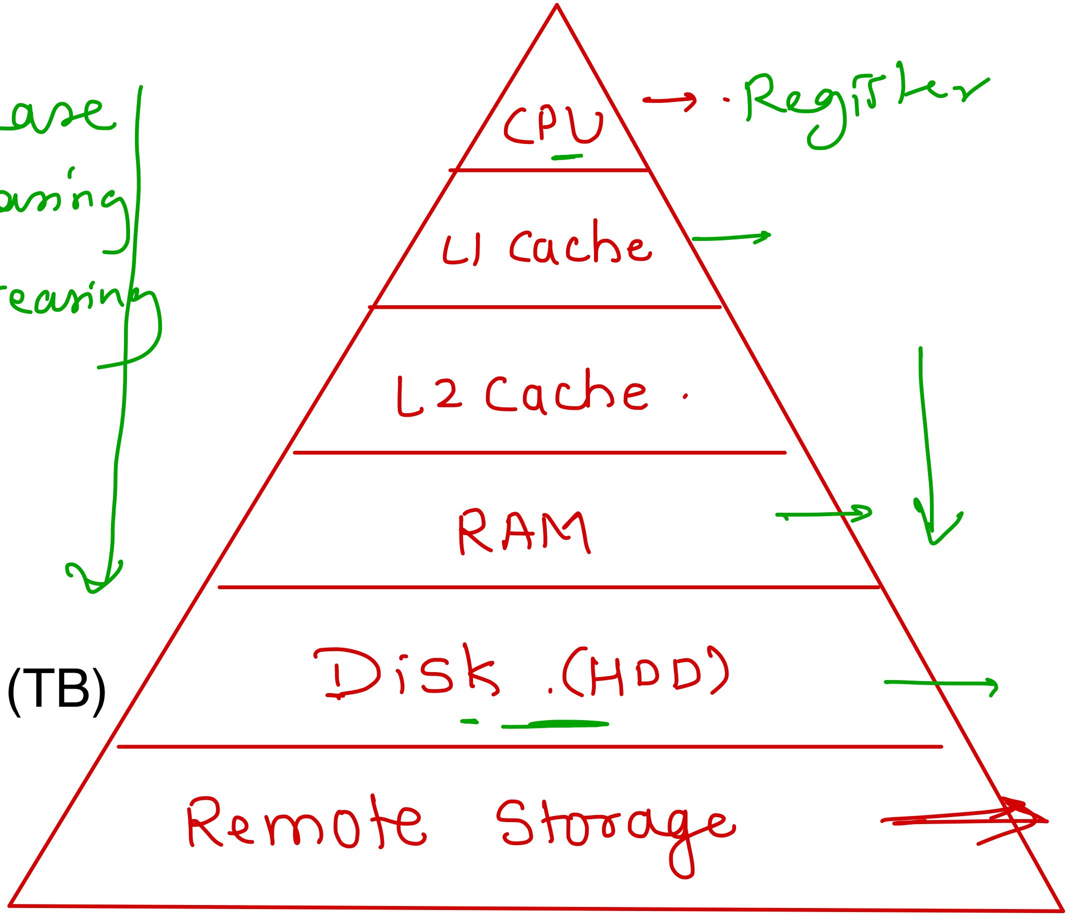
$1024 = 2^{10}$

- Primary memory – Memory
  - Directly accessible by the CPU.
    E.g. CPU registers, Cache, RAM.
- Secondary memory -- Storage
  Memory accessible via Primary memory. E.g. Disk, CD/DVD, Tape, ROM.
- Volatile vs Non-volatile memory
  - Volatile memory: The contents of memory are lost when power is OFF.
    Non-volatile memory: The contents of memory are retained even after power is OFF.

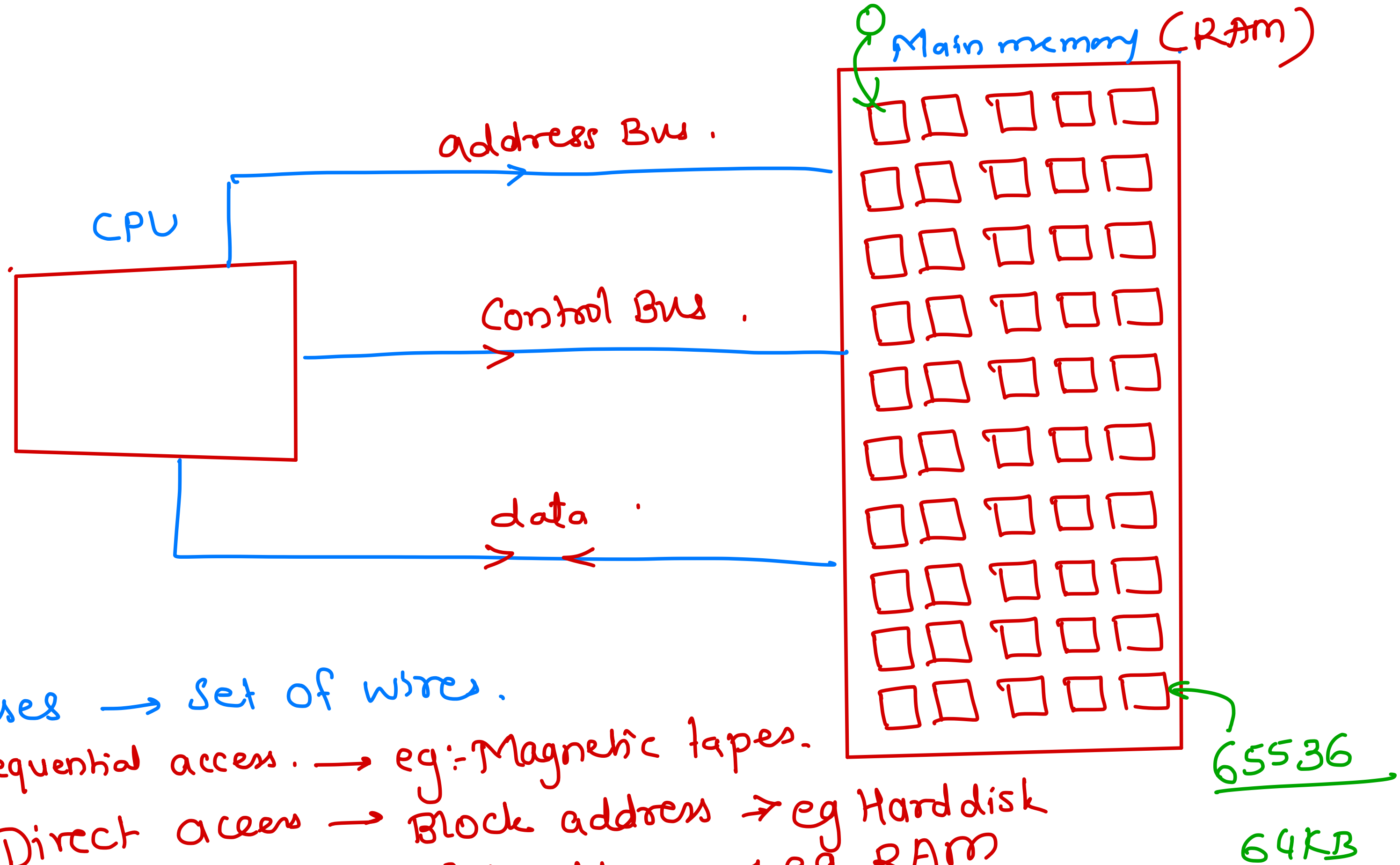
Memory Hierarchy

- Level 0: CPU Registers Bit
- Level 1: L1 Cache KB, MB
- Level 2: L2 Cache
- Level 3: RAM → GB
- Level 4: Disk (Local) → TB
- Level 5: Remote storage (NFS)
- Comparison
  - Capacity: Level 0 is smallest (B) to Level 5 is largest (TB)
  - Speed: Level 0 is fastest and Level 5 is slowest
  - Cost: Level 0 is costlier and Level 5 is cheaper

Size increase
Cost decreasing
Speed decreasing



pymaid



Buses → Set of wires.

1. Sequential access. → eg:- Magnetic tapes.
2. Direct access → Block address → eg Hard disk
3. Random access → Byte address → eg RAM
4. Associative access → Key-value pair storage → eg Cache

Memory Access

- CPU <----> Memory
- Address bus
  - Unidirectional from CPU to the memory
  - Address represent location of the memory to read/write
  - Number of lines = number of locations
- Data bus
  - Bi-directional from/to CPU to/from the memory
  - Carries the data
  - Number of lines = width of data unit
- Control bus
  - Read/Write operation
- CPU <---> Cache <---> RAM <---> Disk
- Sequential access: Read/write sequentially from start to end. e.g. Magnetic tapes
- Direct access: Read/write to the block address e.g. Hard disk
- Random access: Read/write to the memory (byte) address to e.g. RAM
- Associative access: Read/write to the scan/tag lines(Key –value storage)e.g. Cache

RAM (Random Access Memory)

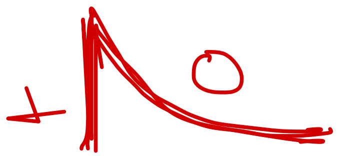
- RAM is packaged as a chip, Basic storage unit is a cell (one bit per cell)
- Internal memory of the CPU for storing data, program, and program result
- Used for Read/ Write
- Volatile (Temporary Storage)

Static RAM (SRAM)

- Retains its contents as long as power is being supplied.
- Made up of transistors.
- SRAM is more often used for system cache.
- SRAM is faster than DRAM.

Dynamic RAM (DRAM)

- Must be constantly refreshed or it will lose its contents.
- This is done by placing the memory on a refresh circuit that rewrites the data several hundred times per second.
- Made up of memory cells composed of capacitors and one transistor.
- DRAM is typically used as the main memory in computers.



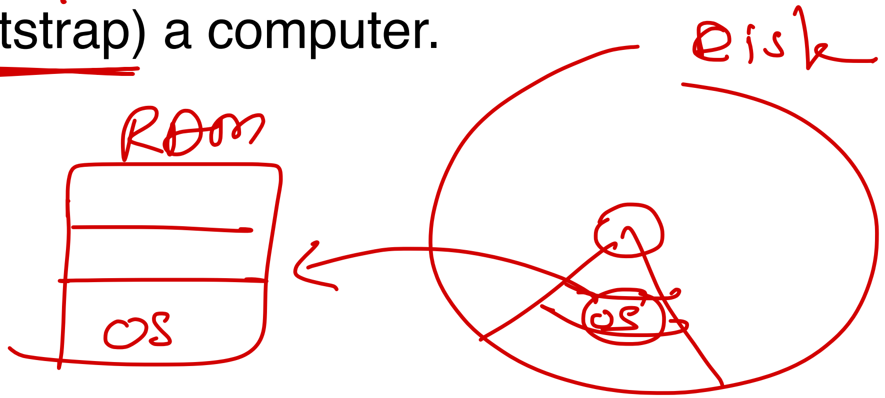
Computer Fundamental

ROM (Read Only Memory)

- Read-only memory (Not writable).
- This type of memory is non-volatile.
- The information is stored permanently.
- A ROM stores such instructions that are required to start (bootstrap) a computer.

Prog → ROM

- BIOS – Basic Input Output System
  - Set of programs stored in PC Base ROM (on Motherboard).
  - 1 POST(Power On Self Test) – To test the peripherals.
  - 2 Bootstrap Loader – To find OS in disk/usb/cd.com.
  - 3 Basic/minimal device drivers for basic device functionality.
  - 4 BIOS setup utility (F1 or ESC).
- Programs (executable instructions) stored in ROM are called -- Firmware. e.g. BIOS, Bootstrap loader, POST, ...

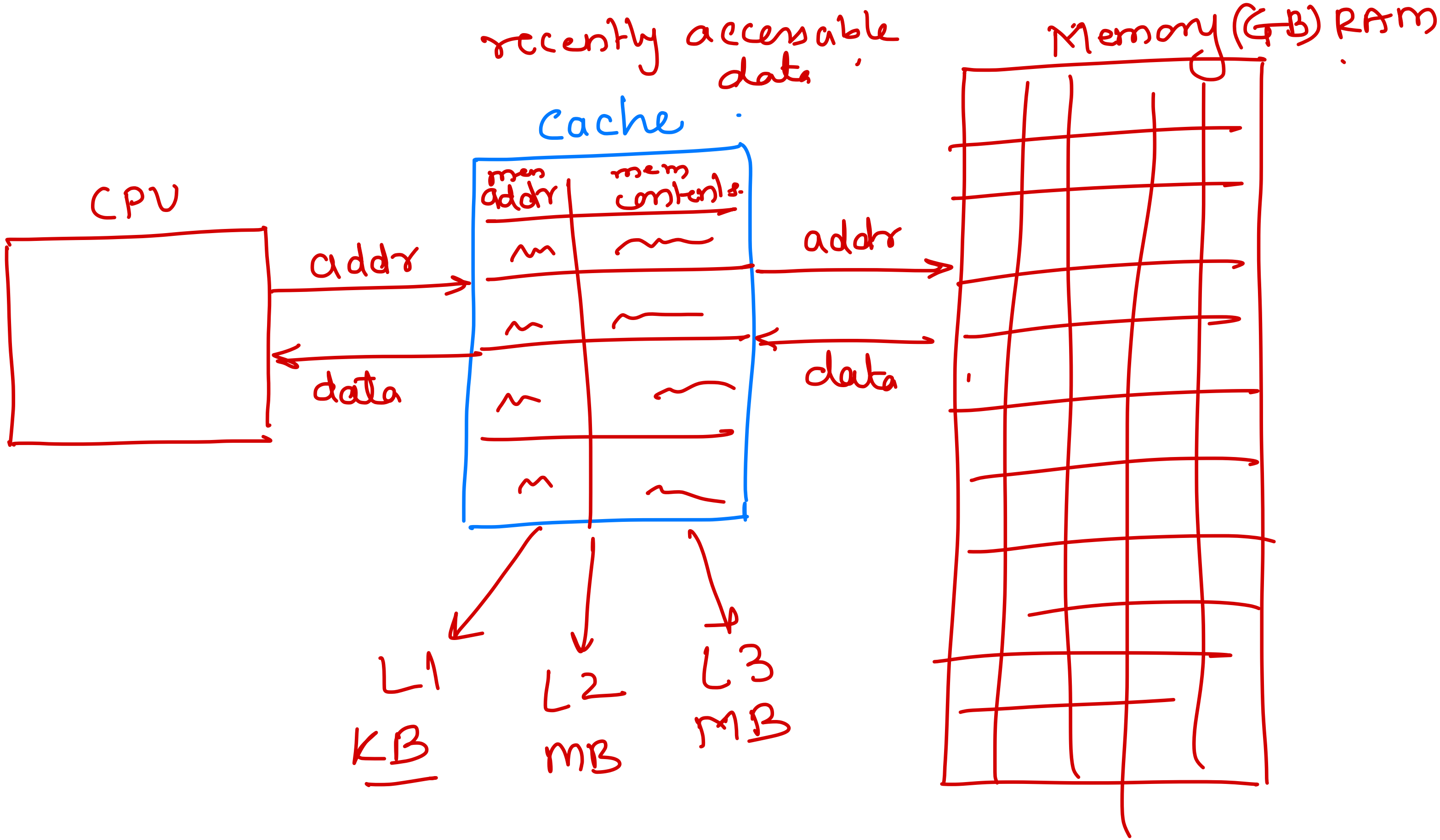


Types of ROM

- Masked ROM (MROM) -- contents are fixed while manufacturing
- Programmable ROM (PROM) -- one time writable
- Erasable Programmable ROM (EPROM) -- written multiple times with special circuit
  - Ultra-Violet EPROM (UV-EPROM) -- all contents erased using UV rays
  - Electrical EPROM (E-EPROM) -- erase selected bytes using high electric current
- Flash (like E-EPROM) -- erase selected blocks - high speed

Cache memory

- Associative memory. Key = memory address, Value = memory contents.
- Made up of cache lines tagged by tag index.
- In CPU chip or outside CPU chip.
- If requested data is found in cache (cache hit), contents are sent from cache itself to CPU (faster access).
- If requested data is not found in cache (cache miss), contents are accessed from main memory, copied in cache and then sent to CPU (slower access).
- Cache memory size is limited (KB or MB).
- When cache is full, oldest data is overwritten (Least Recently Used).
- Cache always stores recently accessed data.


recently accessable data
Memory (GB) RAM
cache
mem addr
mem contents
CPU
addr
data
addr
data
L1
KB
L2
MB
L3
MB