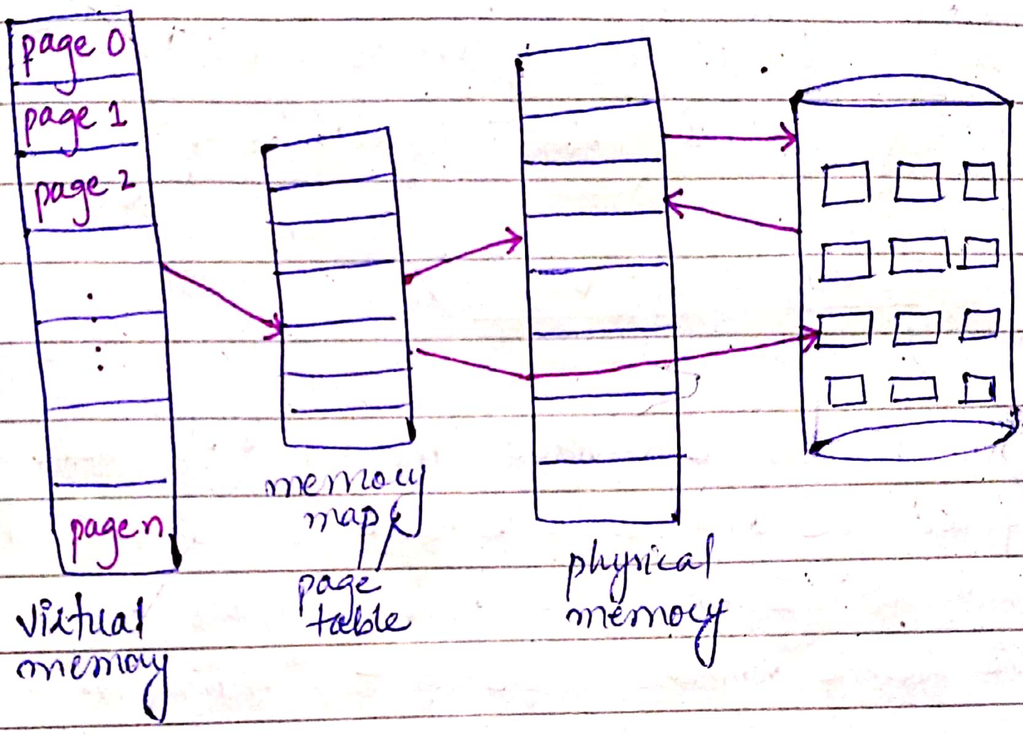
Virtual Memory

A space inside the hard disk which can be addressed as if it was a part of main memory. The programmers are provided a very large virtual memory so that any program is written without the needs of reducing its size.

There are some code which are seldom used such as error/exception handling code. Or sometimes, programmer allocates extra memory which is not actually needed. So, creating a scheme that would only load the part of program which is currently needed helps to increase multiprogramming by allowing many processes to be stored in the main memory.

→ Increase multiprogramming
→ Increase CPU utilization &, throughput
→ provide separation for logical and physical address space.
→ during swapping or loading. less I/O will be done so reduce the time.

→ a large virtual address space is provided which is much larger than the actual physical memory.



virtual memory ~~stores~~ shows the pages that a process is divided into and the logical addresses for those pages. so that when CPU generates logical address we know which page is present on that address.

→ allows the address spaces to be shared by several process. If there is a shared memory than in logical address space, each process will have their separate page for that memory but in actual physical memory, they will be mapped to the same frame.

Virtual Memory is implemented using Demand Paging.

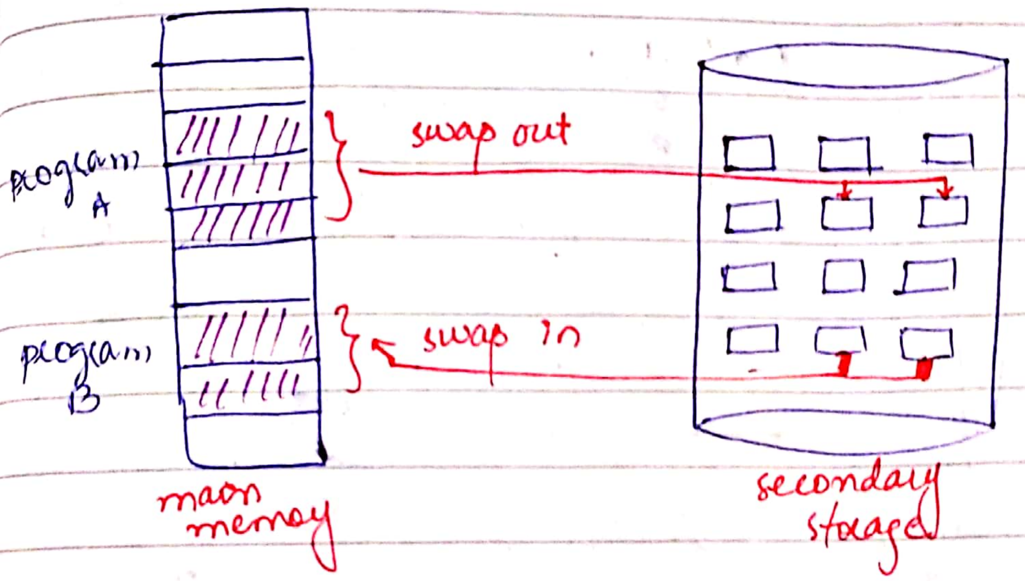
~~Jo paging main poora likha hai~~
~~.vo. with first (address mapping) diagram~~

~~invalid-valid bit diagram.~~

The process is divided into multiple pages of fixed size and a page table is maintained with all the entries of process's pages along with the frame number where in the main memory they are located and a valid-invalid bit.

Initially whole process is not loaded into the memory. Only those pages which are currently required will be loaded into the memory and in page table, the frame no. of those pages will be added. Also, the valid-invalid bit is set to valid.

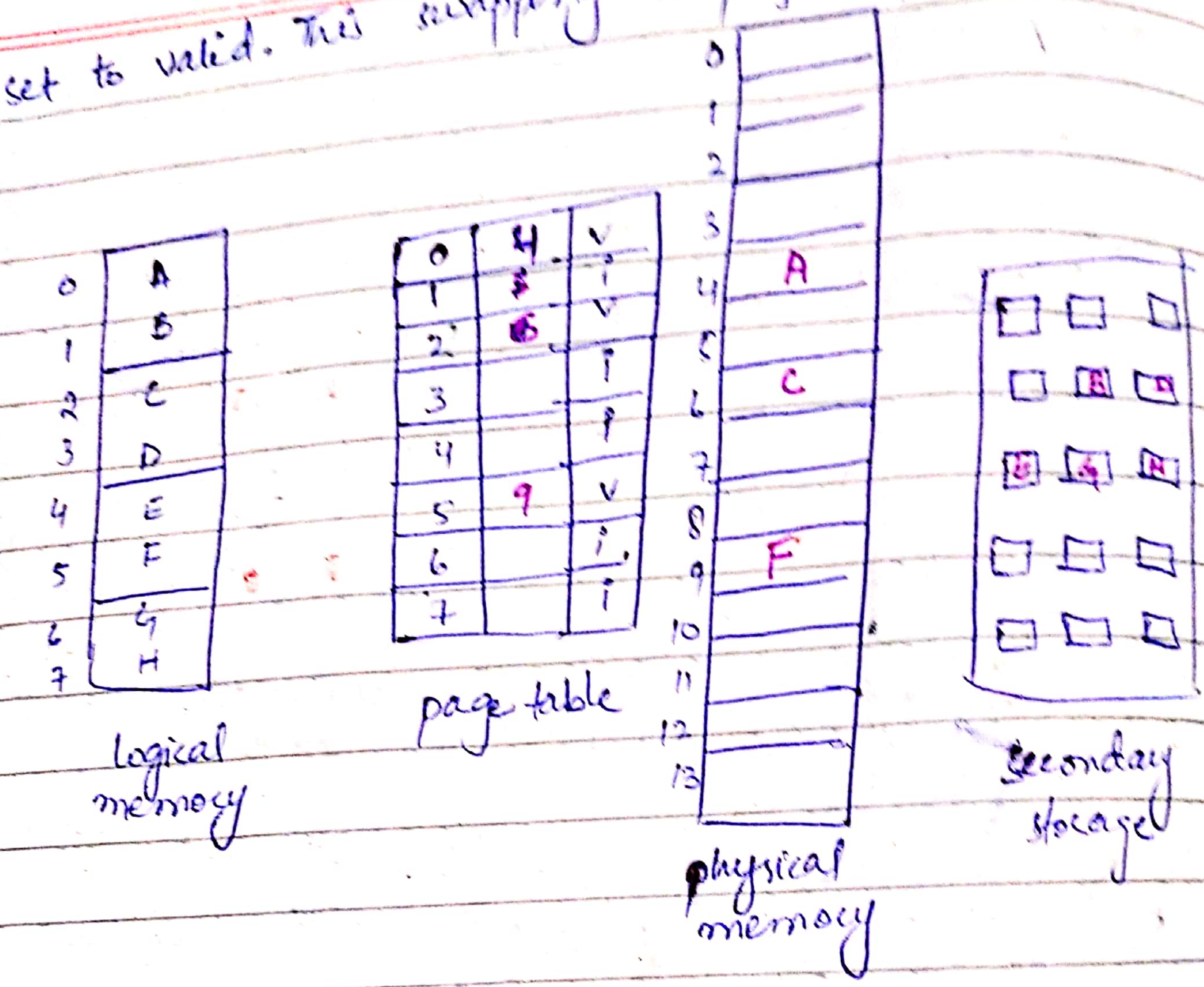
When some new process or new pages of current process has to be loaded and the frames are not available then some of the unused pages are swapped out to the hard disk and the required pages are swapped in.



Whenever a CPU generates ~~the~~ logical address, the address translation is done.

<< refer to the diagram in paging >>

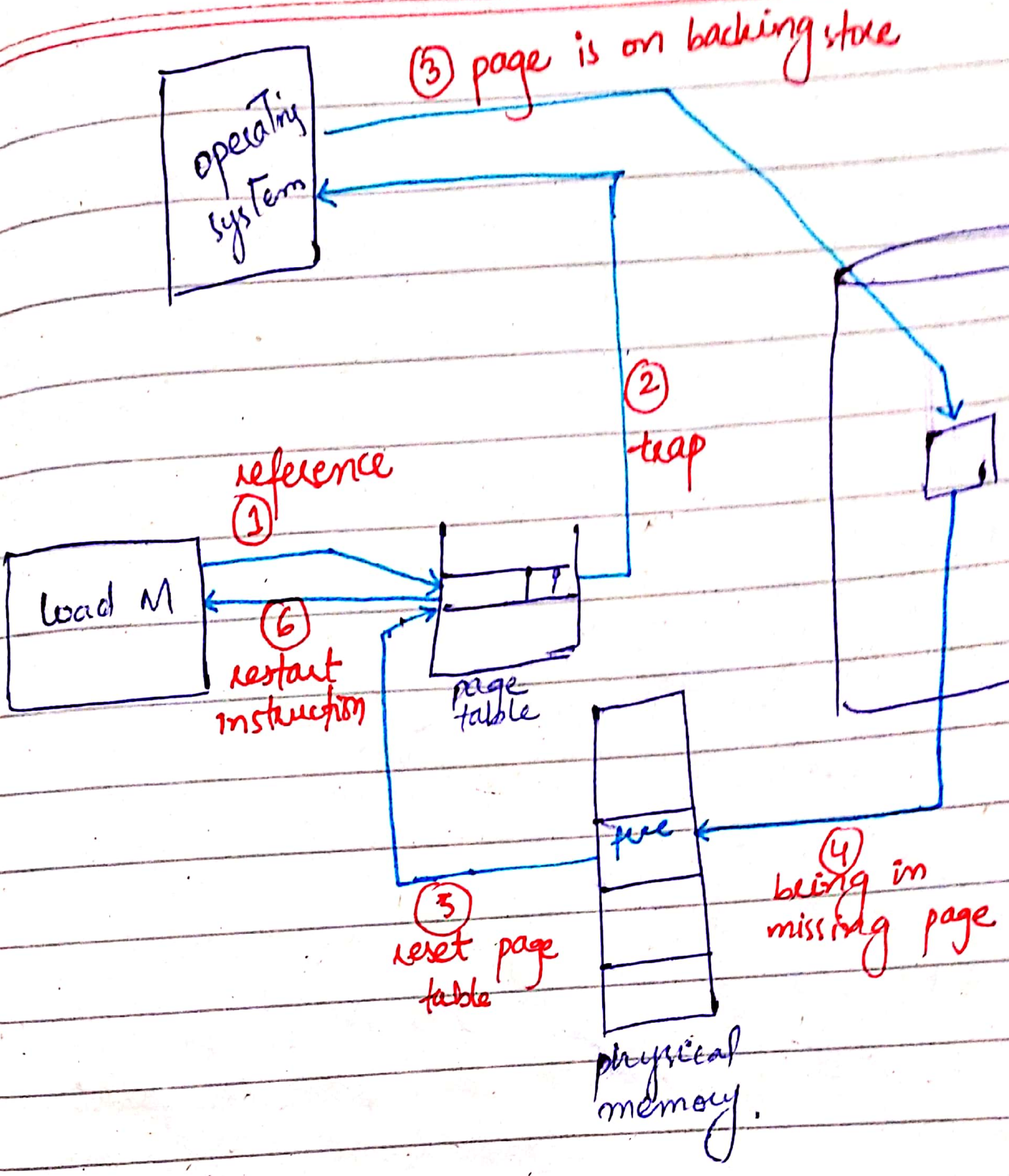
When in page table, the address of the page has to be ~~achieved~~ extracted/fetched then first it is checked whether the bit is ~~saved~~ set to valid or invalid. If the bit is valid, it means the page is currently loaded into the memory otherwise the page has to be fetched into the memory from the secondary storage. After that the frame no. associated with that page will be added in the page table and bit will be set to valid. This swapping is page replacement.



Page Fault

When some address is wrong/invalid, the process is terminated. If the bit is invalid because the ~~sale~~ page is not in memory then ~~memory hand~~ page fault handling will take place.

(explain the diagram)



Demand paging advantages:

↳ less I/O needed (bcz not each page will be fetched into memory).

↳ less memory needed

↳ faster response

↳ more users