KARNAVATI UNIVERSITY

Operating Systems

Course Code: **71203002004**

Advanced Memory Techniques

by -

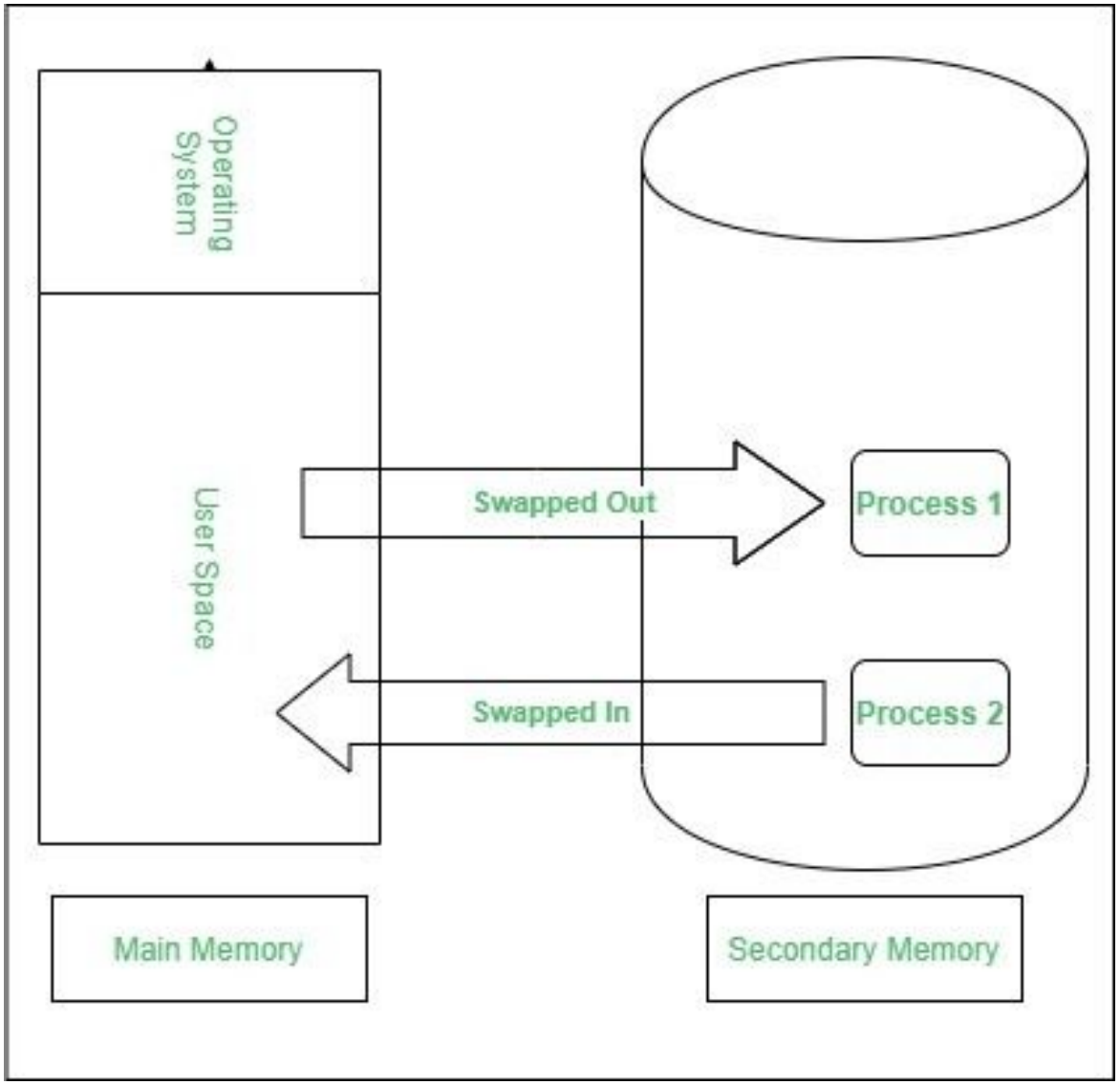
Minal Rajwar

Swapping

- Swapping is a memory management technique used in multiprogramming to increase CPU utilization.
- It involves moving processes between RAM (main memory) and secondary storage (hard disk/SSD).
- When RAM is full, inactive processes are swapped out to secondary storage, and active or high-priority processes are swapped in.
- Although swapping slows performance due to frequent disk access, it allows multiple and larger programs to run on systems with limited RAM.

Swapping has been subdivided into two concepts: swap-in and swap-out.

- Swap-out is a technique for moving a process from RAM to the hard disc.
- Swap-in is a method of transferring a program from a hard disc to main memory, or RAM.



Process of Swapping

1. When RAM is full, inactive data is moved to secondary storage.
2. Space is freed for new or high-priority programs.
3. Swapped-out processes are reloaded when needed.

Example: Like keeping essential books on a desk (RAM) and the rest in a cabinet (secondary storage), swapping lets you work with more items than your desk can hold.

Advantages	Disadvantages
Increases CPU utilization.	Risk of data loss if power fails.
Allows larger programs to run on limited memory.	Frequent swapping causes slowdowns (disk access is slower than RAM).
Prevents system overload by freeing space for active processes.	Too much swapping reduces CPU efficiency.

Swapping enables efficient use of memory by temporarily moving processes between RAM and secondary storage, supporting multitasking and better performance despite some trade-offs.

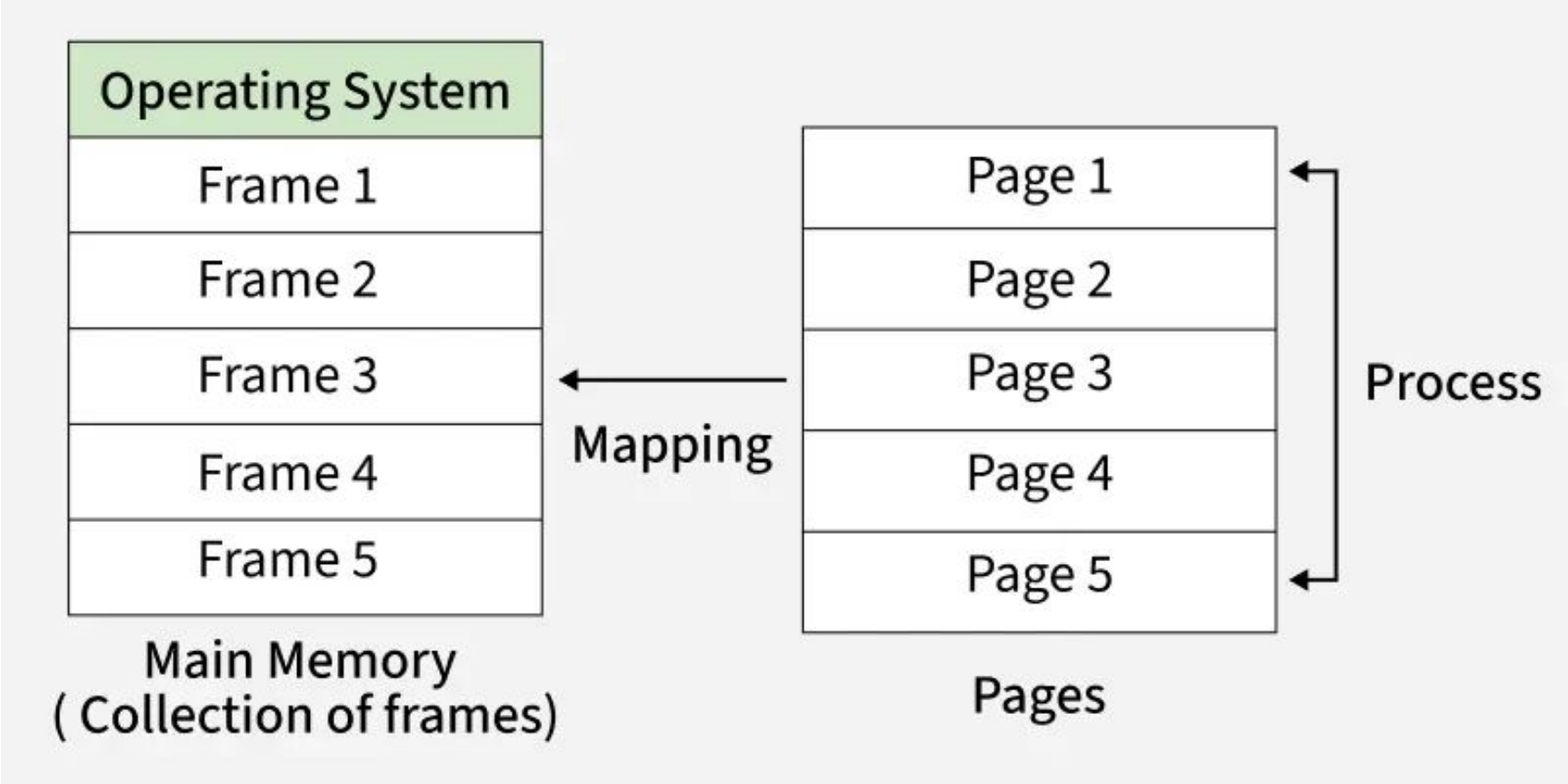
Paging

Definition:

Paging is a memory management technique where a process is divided into **fixed-size blocks called pages**. These pages are loaded into **frames** of physical memory (RAM).

Functions of a Page Table:

- Maintains mapping between **logical pages** and **physical frames**.
- Memory Management Unit (MMU) translates logical → physical addresses.
- To speed up lookup, **TLB (Translation Lookaside Buffer)** is used.


Operating System
Frame 1
Frame 2
Frame 3
Frame 4
Frame 5
Main Memory
(Collection of frames)
Mapping
Page 1
Page 2
Page 3
Page 4
Page 5
Pages
Process

Why Paging?

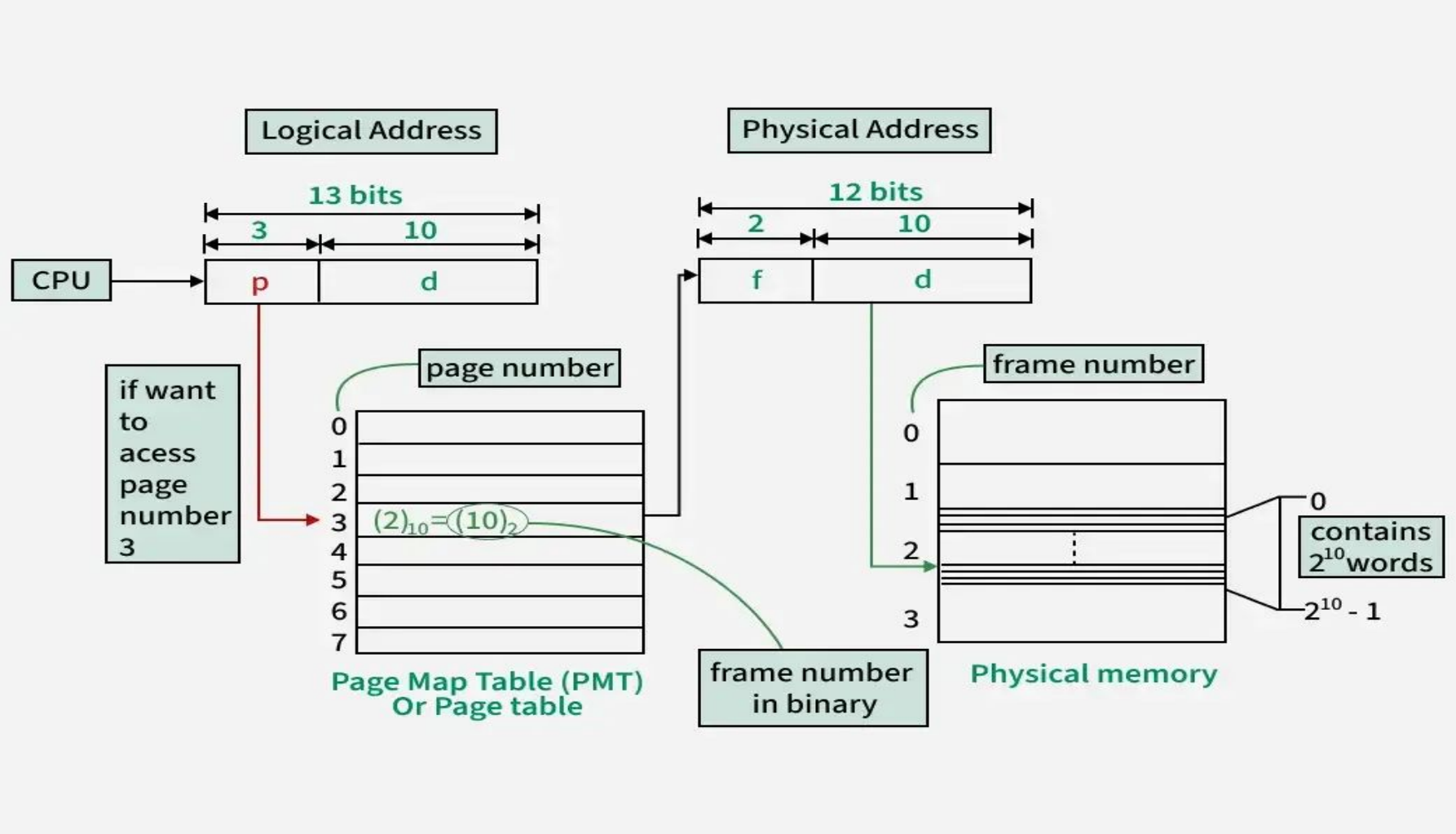
- Memory may not be available as one large block.
- Allows non-contiguous allocation.
- Programs can grow dynamically.

Terminologies & Important Features:

- **Logical (Virtual) Address Space:** All possible logical addresses a process can generate.
- **Physical Address Space:** Actual RAM addresses.
- **Page Number (p):** Identifies a page.
- **Page Offset (d):** Word within the page.
- **Frame Number (f):** Identifies a frame in RAM.
- **Frame Offset (d):** Word within the frame.

Working of Paging

- When a process requests memory, the OS allocates page frames and maps its logical pages to physical frames.
- Each process has a **page table** that tracks page locations.
- When the program accesses data, the OS uses the page table to translate **logical addresses** → **physical addresses** in RAM.


Logical Address
13 bits
3
10
CPU
p
d
if want to acess page number 3
page number
0
1
2
3
4
5
6
7
$(2)_{10}=(10)_2$
Page Map Table (PMT) Or Page table
Physical Address
12 bits
2
10
f
d
frame number in binary
frame number
0
1
2
3
0
contains 2^{10} words
2^{10} - 1
Physical memory

Steps in Paging

- Divide logical memory → pages, physical memory → frames.
- Load process pages into available frames.
- Use Page Table to map pages → frames.
- Translate logical → physical address.
- Handle page faults if required page isn’t in RAM.

Example (Paging Calculation):

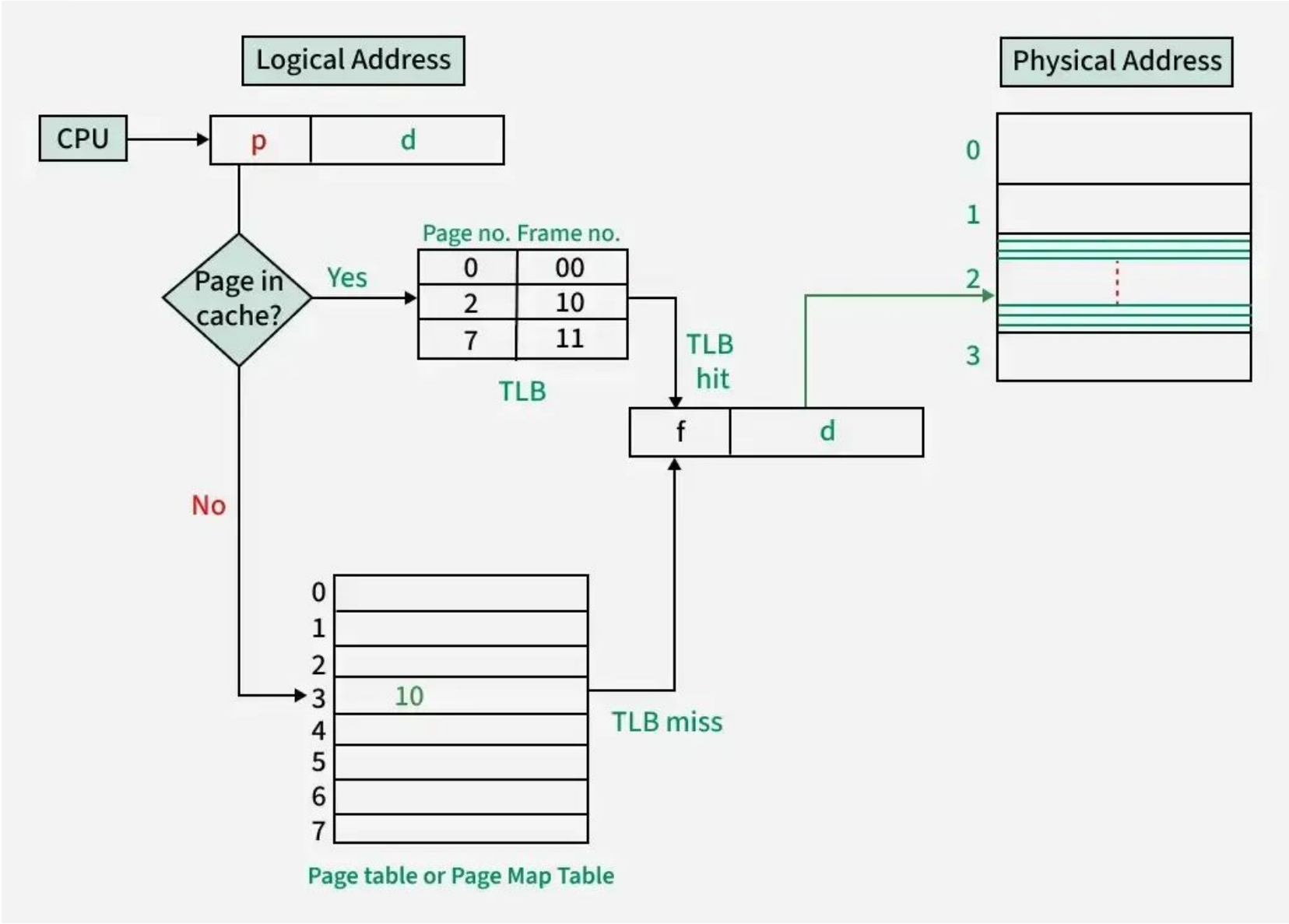
- Physical Address = **12 bits** → Physical Address Space = **2^{12} = 4K words**
- Logical Address = **13 bits** → Logical Address Space = **2^{13} = 8K words**
- Page size = Frame size = **1K words**

Calculations:

- Number of Frames = 4K / 1K = **4 = 2^2**
- Number of Pages = 8K / 1K = **8 = 2^3**

Hardware Implementation of Paging

- **Small Page Table:** Stored in dedicated registers.
- **Large Page Table:** Uses **TLB (Translation Lookaside Buffer)**, a fast associative cache.
- **TLB Working:** Each entry = tag + value; tags compared in parallel, matched value returned.
- **Access Time:** If page table in memory → Effective access time = m (table) + m (page).


Logical Address
Physical Address
CPU
p
d
Page in cache?
Yes
Page no. Frame no.
0 00
2 10
7 11
TLB
TLB hit
f
d
No
0
1
2
3
4
5
6
7
10
TLB miss
Page table or Page Map Table

Advantages	Disadvantages
Removes external fragmentation.	Internal fragmentation (last page waste).
Efficient use of scattered free memory.	Overhead of maintaining page table.
Supports virtual memory.	Slower access due to address translation (reduced by TLB).
Easier and faster swapping at page level.	Page faults cause I/O delays.
Provides security & isolation (processes can’t access each other’s memory).	Requires complex hardware (MMU, TLB, replacement algorithms).

DISCUSSION & REVISION

1. What are the fixed-size blocks of logical memory called in paging?
2. Which hardware unit converts logical addresses to physical addresses?
3. In swapping, processes are moved between RAM and which type of storage?
4. What is the name of the high-speed cache used to speed up page table lookup?
5. Which type of fragmentation is eliminated by paging?

REFERENCES

- https://www.geeksforgeeks.org/operating-systems/swapping-in-operating-system/
- https://www.geeksforgeeks.org/operating-systems/paging-in-operating-system/