# CS 343 - Operating Systems

## Module-4B

## Segmentation and Paging Techniques


**Dr. John Jose**

**Associate Professor**

**Department of Computer Science & Engineering**

**Indian Institute of Technology Guwahati**

# Overview of Memory Management

- Background
- Swapping
- Contiguous Memory Allocation
- Segmentation
- Paging
- Structure of the Page Table

# Dynamic Storage-Allocation Problem

- How to satisfy a request of size ***n*** from a list of free holes?
- **First-fit**
- **Best-fit**
- **Worst-fit**
- **Quick-fit**

# Fragmentation

- **External Fragmentation** – total memory space exists to satisfy a request, but it is not contiguous
- **Internal Fragmentation** – allocated memory may be slightly larger than requested memory; this size difference is memory internal to a partition, but not being used

# Fragmentation

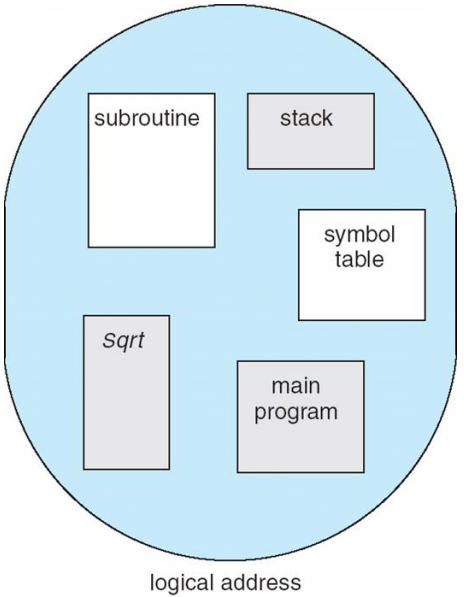
- Reduce external fragmentation by **compaction**
  - Shuffle memory contents to place all free memory together in one large block
  - Compaction is possible *only* if relocation is dynamic, and is done at execution time
  - Latch I/O job in memory while it is involved in I/O

# Principle of Locality

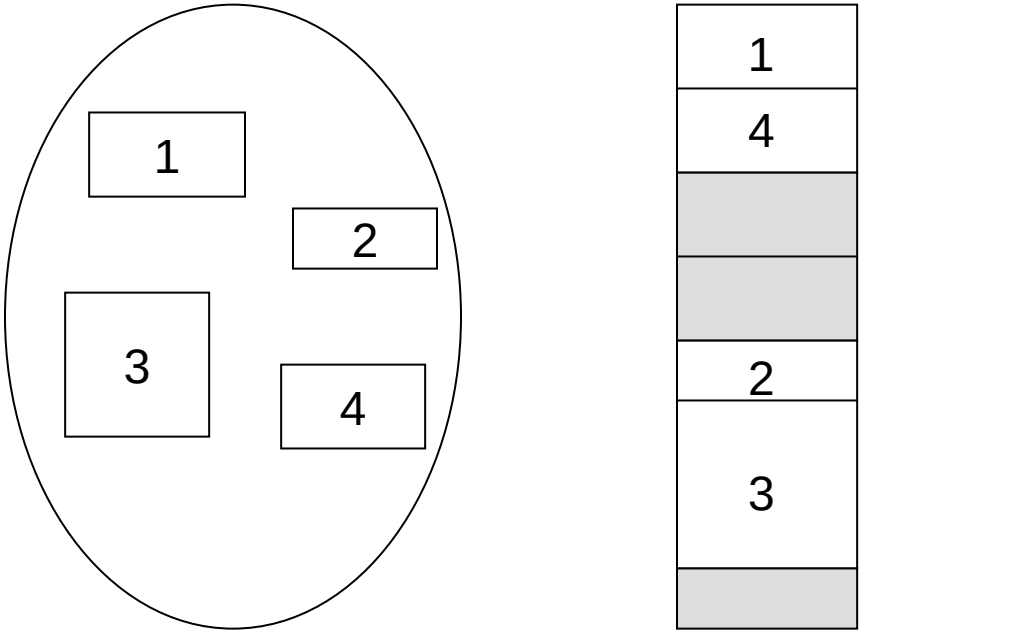
- Program and data references within a process tend to cluster around an address space. Only a few pieces of a process will be needed over a short period of time
- This help us to make intelligent guesses about which pieces will be needed in the future
- To implement virtual memory, hardware must support paging and segmentation
- OS must do the movement of pieces of process between main and secondary memory

# Segmentation

- Memory-management scheme that supports user view of memory
- A program is a collection of segments
  - A segment is a logical unit such as:
- main program
- procedures
- stack
- symbol table
- local variables,
- global variables
- common block



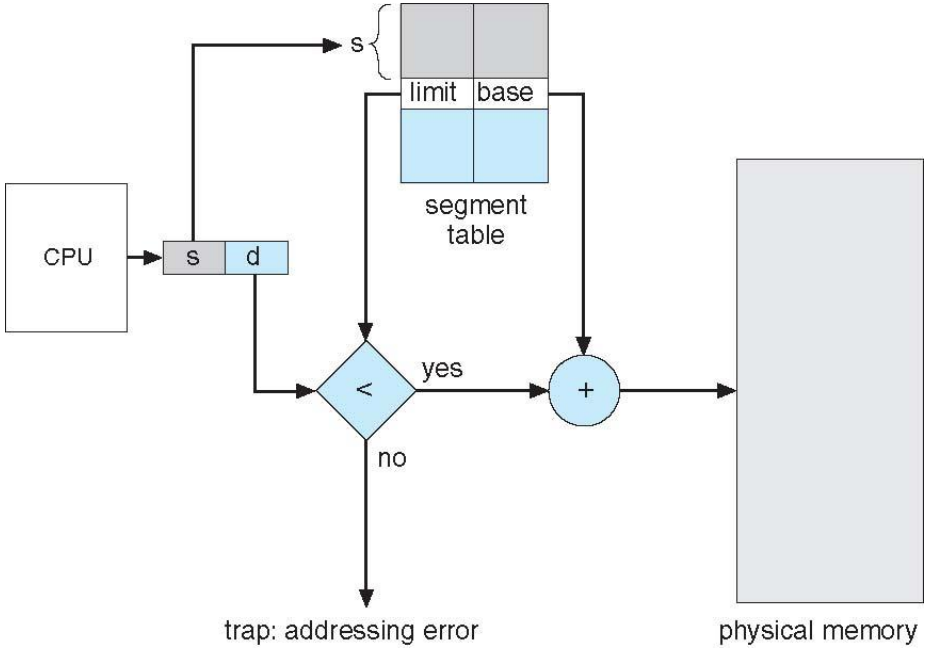
# Logical View of Segmentation



# Segmentation Architecture

- Logical address consists of a two tuple: <segment-number, offset>,
- **Segment table** – maps two-dimensional physical addresses; each table entry has:
  - **base** – contains the starting physical address where the segments reside in memory
  - **limit** – specifies the length of the segment
- **Segment-table base register (STBR)** points to the segment table's location in memory
- **Segment-table length register (STLR)** indicates number of segments used by a program;

# Segmentation Hardware

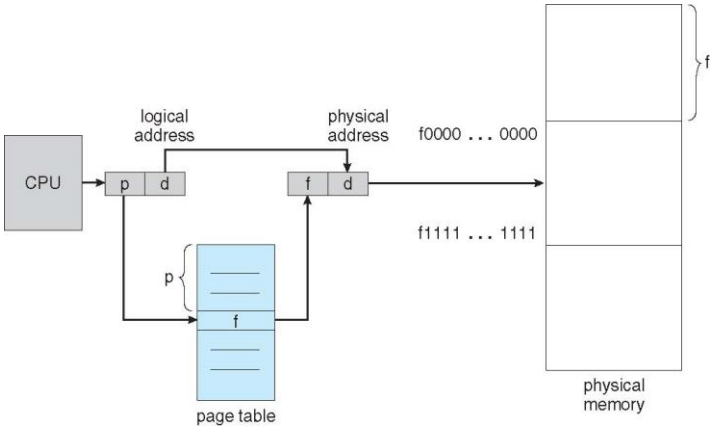


# Paging

- Physical address space of a process can be noncontiguous; process is allocated physical memory whenever the latter is available
  - Avoids external fragmentation
  - Avoids problem of varying sized memory chunks
- Divide physical memory into fixed-sized blocks called **frames**
  - Size is power of 2, between 512 bytes and 16 Mbytes
- Divide logical memory into blocks of same size called **pages**
- Keep track of all free frames
- To run a program of size ***N*** pages, need to find ***N*** free frames
- Set up a **page table** to translate logical to physical addresses

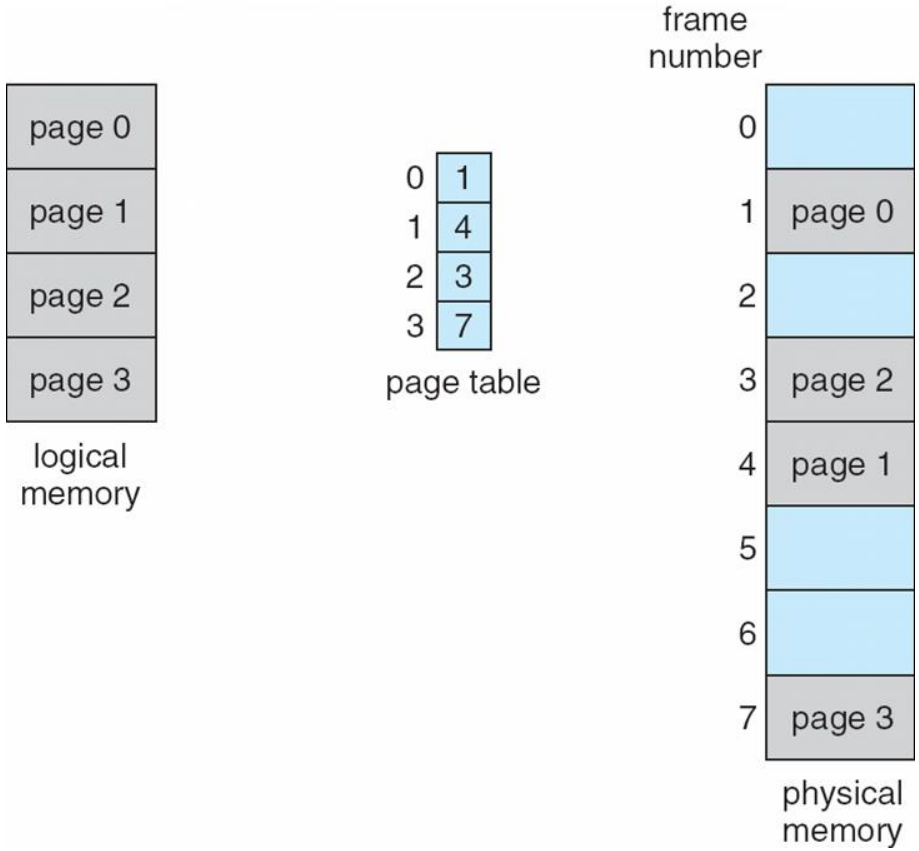
# Address Translation Scheme

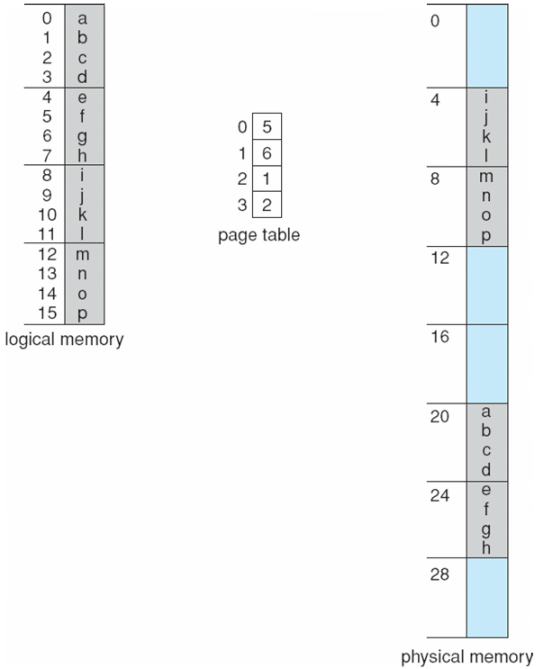
- Address generated by CPU is divided into:
  - **Page number** (***p***) – used as an index into a page table
  - **Page offset** (***d***) – displacement within a page




| page number | page offset |
|---|---|
| p | d |
| m -n | n |

# Paging Model and Page Tables

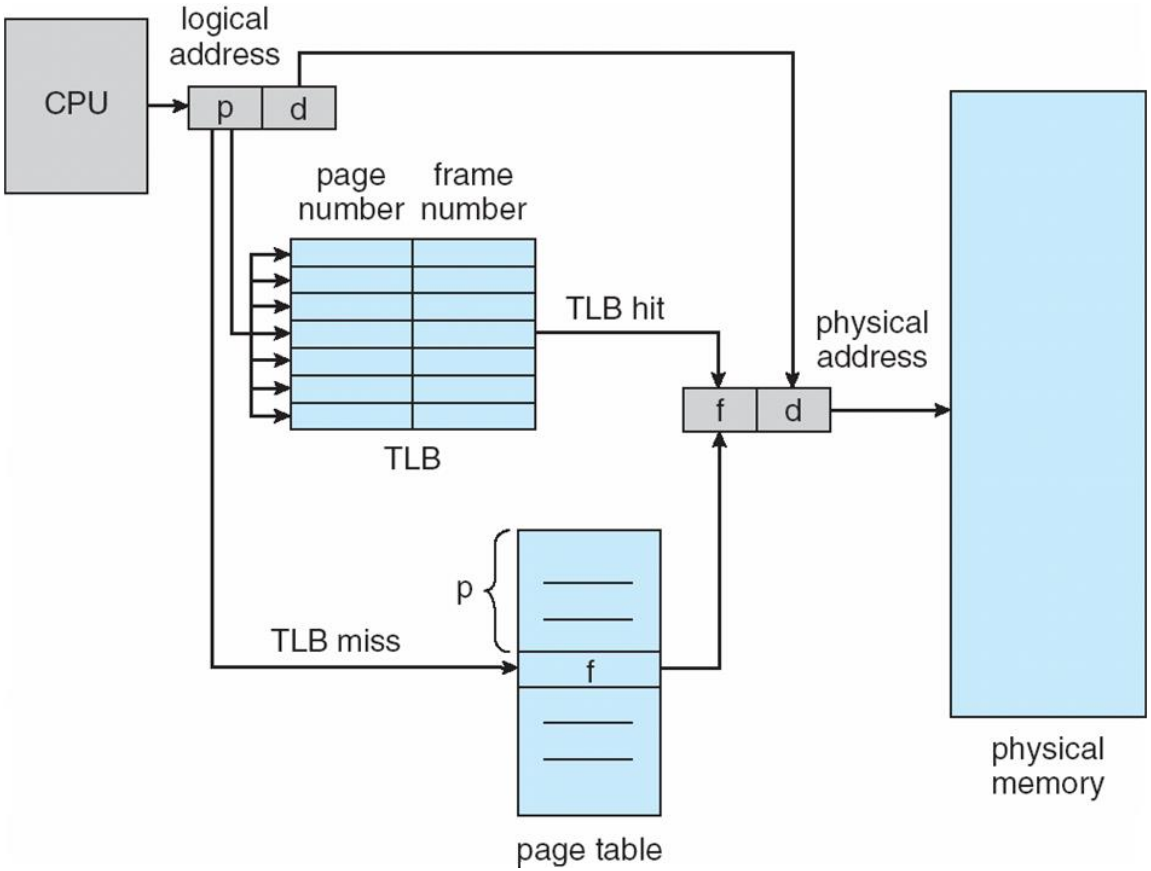


# Implementation of Page Table

- Page table is kept in main memory
- **Page-table base register** (**PTBR**) points to the page table
- **Page-table length register** (**PTLR**) indicates size of the page table
- In this scheme every data/instruction access requires two memory accesses : one for the page table and one for the data / instruction
- The two memory access problem can be solved by the use of a special fast-lookup hardware cache called **associative memory** or **translation look-aside buffers** (**TLBs**)

# Paging Hardware With TLB




**johnjose@iitg.ac.in**

http://www.iitg.ac.in/johnjose/