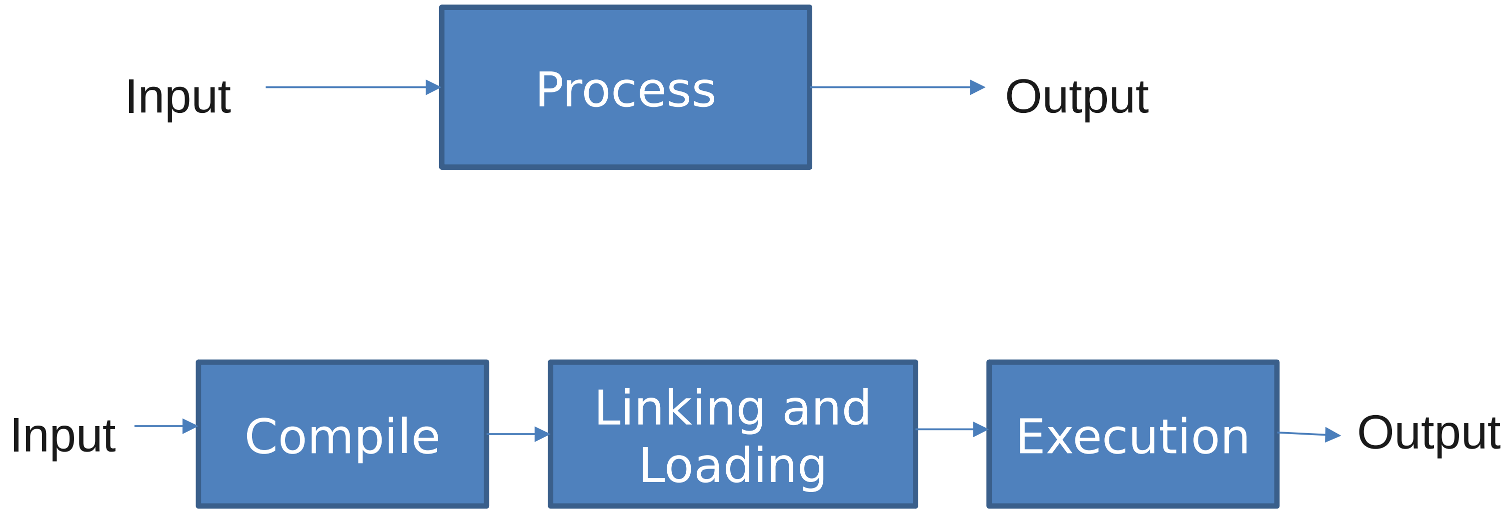
Unit- IV
Memory Management

Chhaya Gosavi

Memory Management

- Background
- Logical versus Physical Address Space
- Swapping
- Contiguous Allocation
- Paging
- Segmentation
- Segmentation with Paging

Background



Program must be brought into memory and for it to be executed.

Logical vs. Physical Address Space

Logical Address – Generated by the CPU; also referred to as virtual address.

Physical Address – Address seen by the memory unit.

Logical and physical addresses are the same in compile-time and load-time , differ in execution-time by address-binding scheme.

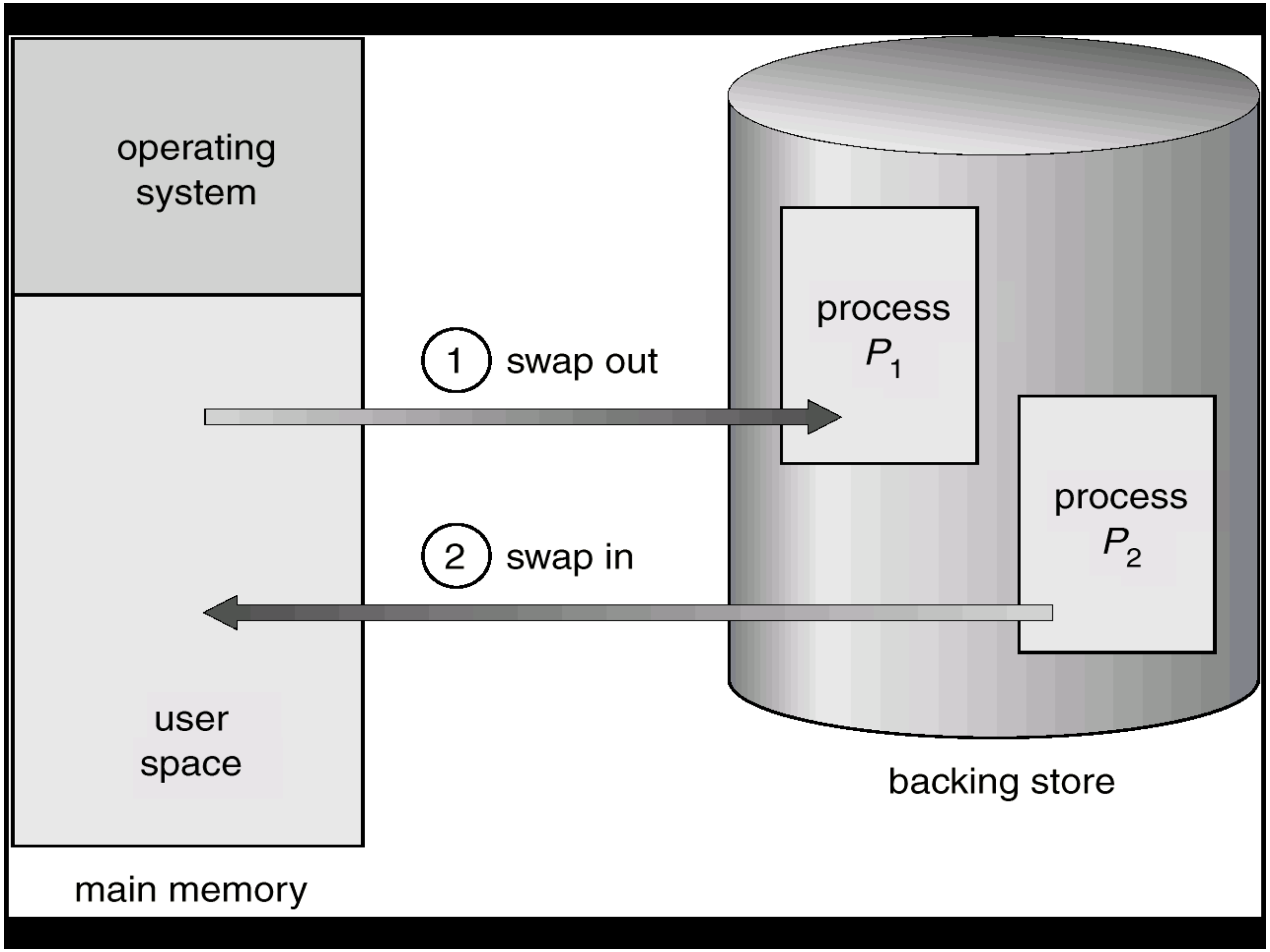
Memory Management Unit

- Hardware device that maps virtual address to physical address.
- In MMU scheme, the value in the relocation register is added to every address generated by a user process at the time it is sent to memory.
- The user program deals with *logical* addresses; it never sees the *real* physical addresses.

Swapping

- A process can be *swapped* temporarily out of memory to a *backing store*, and then brought back into memory for continued execution.
- Backing store – fast disk large enough to accommodate copies of all memory images for all users; must provide direct access to these memory images.
- *Roll out, roll in* – swapping variant used for priority-based scheduling algorithms; lower-priority process is swapped out so higher-priority process can be loaded and executed.
- Major part of swap time is transfer time; total transfer time is directly proportional to the *amount* of memory swapped.
- Modified versions of swapping are found on many systems, i.e., UNIX and Microsoft Windows.

Swapping


operating
system
user
space
main memory
1 swap out
2 swap in
process
P_1
process
P_2
backing store

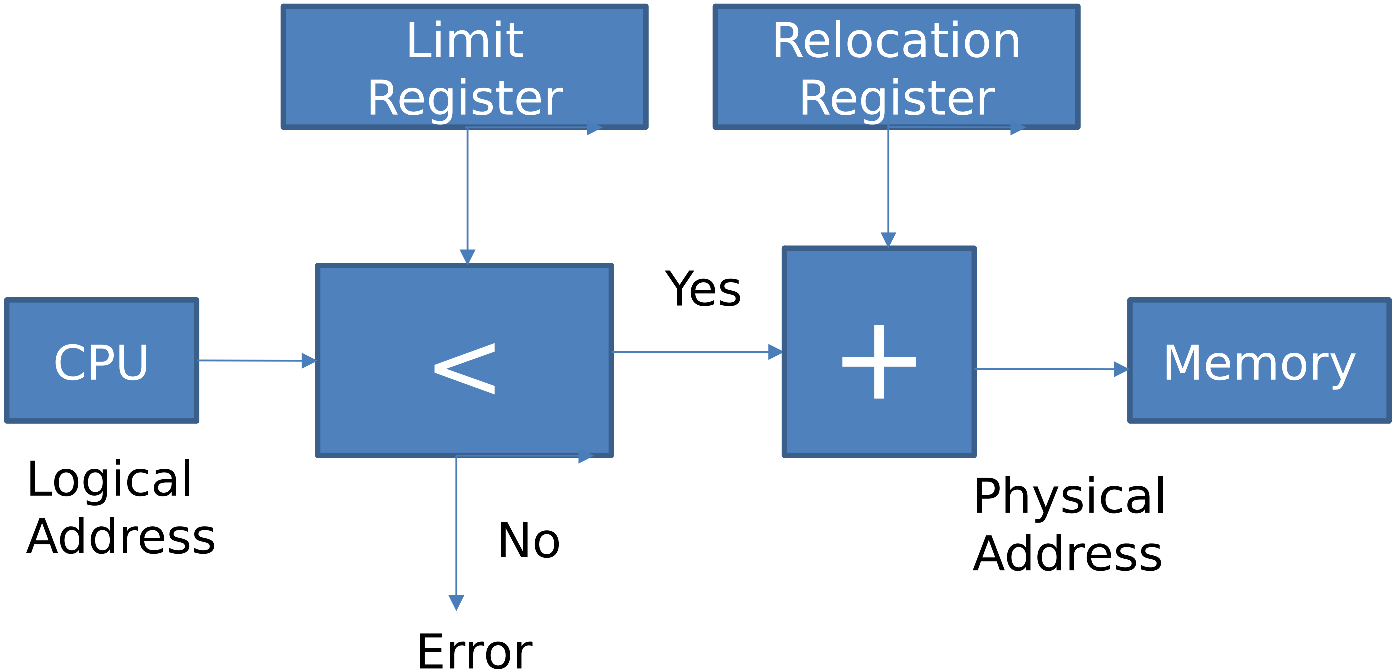
Contiguous Allocation

- Main memory usually divided into two partitions:
  - Resident operating system, usually held in low memory with interrupt vector.
  - User processes then held in high memory.

- Single-partition allocation
  - Relocation-register scheme used to protect user processes from each other, and from changing operating-system code and data.
  - Relocation register contains value of smallest physical address; limit register contains range of logical addresses – each logical address must be less than the limit register.

Contiguous Allocation

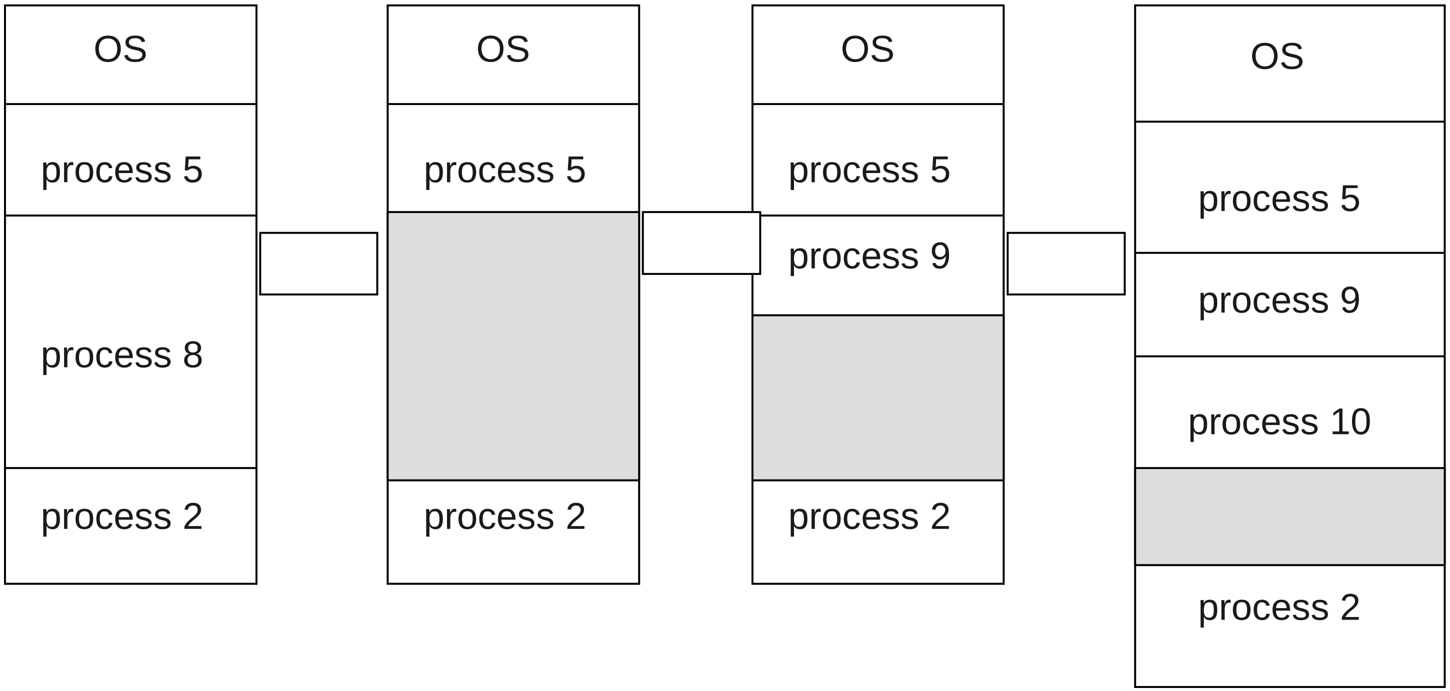
Memory Protection :- OS From User Process and One User Process from Other Process



Contiguous Allocation

- Multiple-partition allocation
  - *Hole* – block of available memory; holes of various size are scattered throughout memory.
  - When a process arrives, it is allocated memory from a hole large enough to accommodate it.
  - Operating system maintains information about: a) allocated partitions   b) free partitions (hole)

Contiguous Allocation



Dynamic Storage-Allocation Problem

How to satisfy a request of size *n* from a list of free holes ?.

First-fit: Allocate the *first* hole that is big enough.
Best-fit: Allocate the *smallest* hole that is big enough; must search entire list, unless ordered by size. Produces the smallest leftover hole.
Worst-fit: Allocate the *largest* hole; must also search entire list. Produces the largest leftover hole.

First-fit and best-fit better than worst-fit in terms of speed and storage utilization.

Four Process

P1(100K), P2(250K), P3(450K), P4(380K)

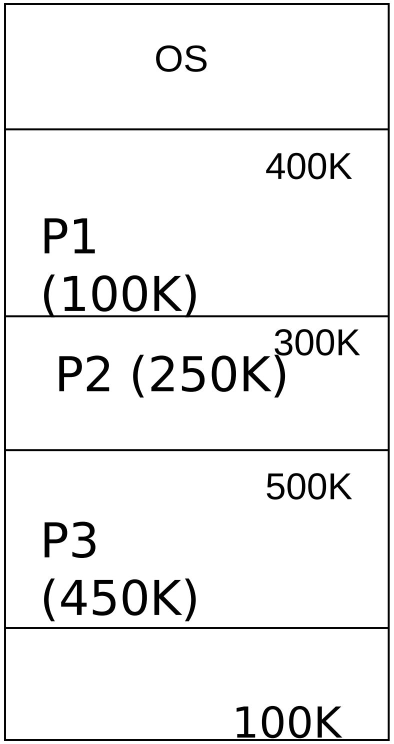
Four Holes

Four Process

P1(100K), P2(250K), P3(450K), P4(380K)

Four Holes

First - Fit

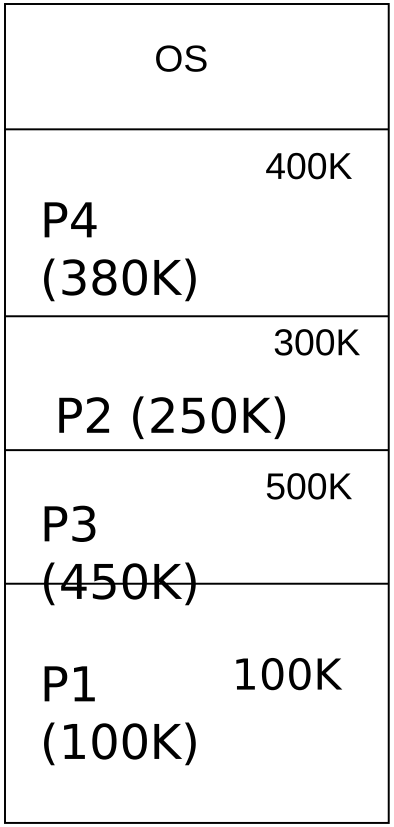


Four Process

P1(100K), P2(250K), P3(450K), P4(380K)

Four Holes

Best - Fit

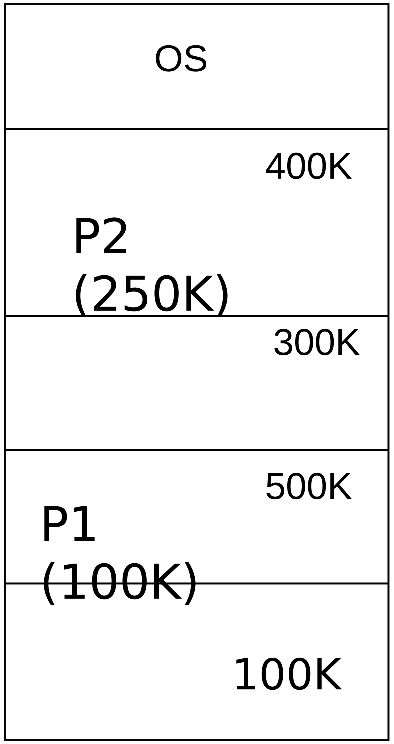


Four Process

P1(100K), P2(250K), P3(450K), P4(380K)

Four Holes

Worst - Fit



Fragmentation

External fragmentation – Total memory space exists to satisfy a request, but it is not contiguous.

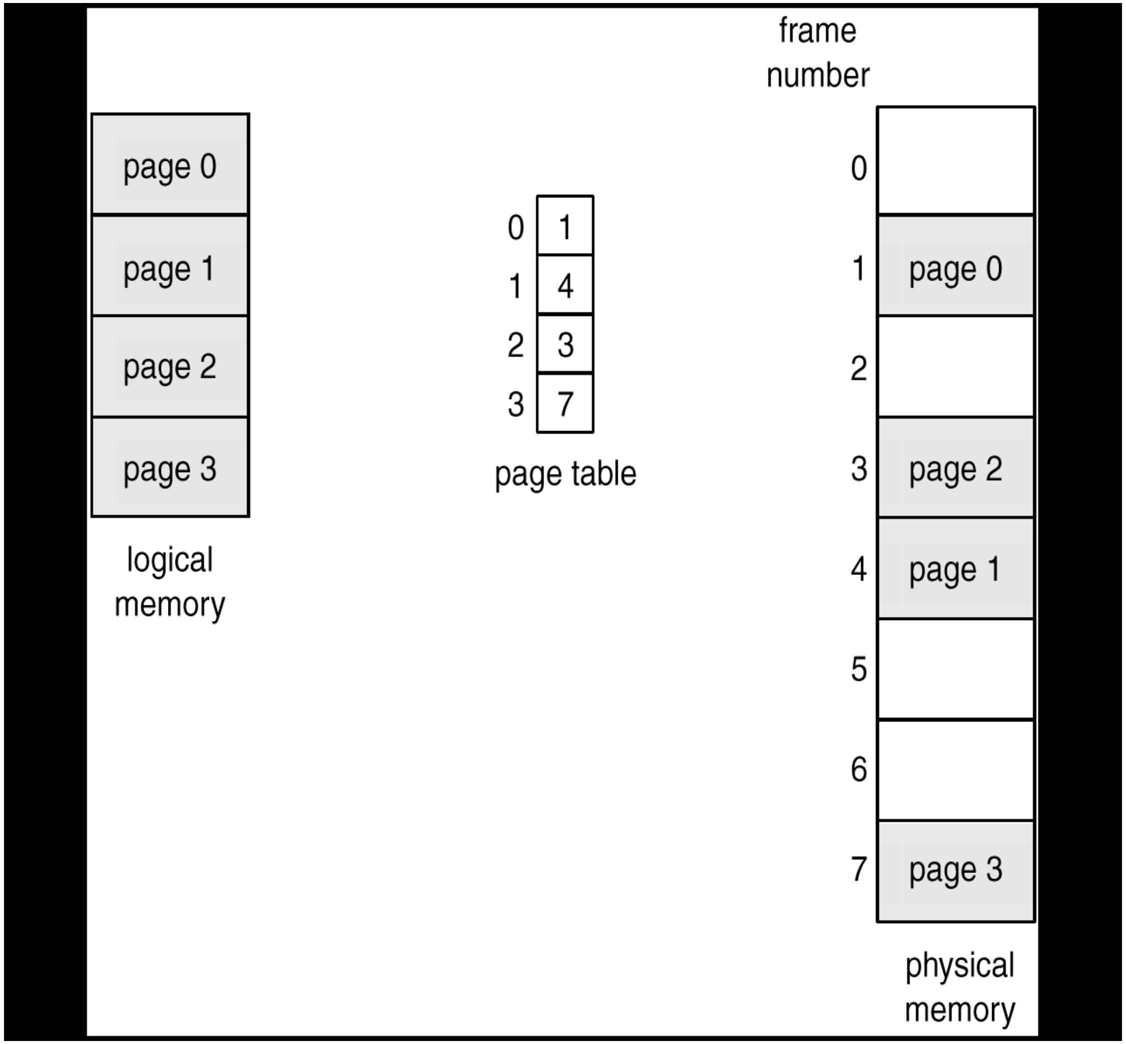
Internal fragmentation – Allocated memory may be slightly larger than requested memory; this size difference is memory internal to a partition, but not being used.

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.

Paging

- Logical address space of a process can be noncontiguous; process is allocated physical memory whenever the latter is available.
- Divide **physical memory** into fixed-sized blocks called **frames** (size is power of 2, between 512 bytes and 32K).
- 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 and load program.
- Set up a **page table** to translate logical to physical addresses.
- **Internal fragmentation**.

Paging

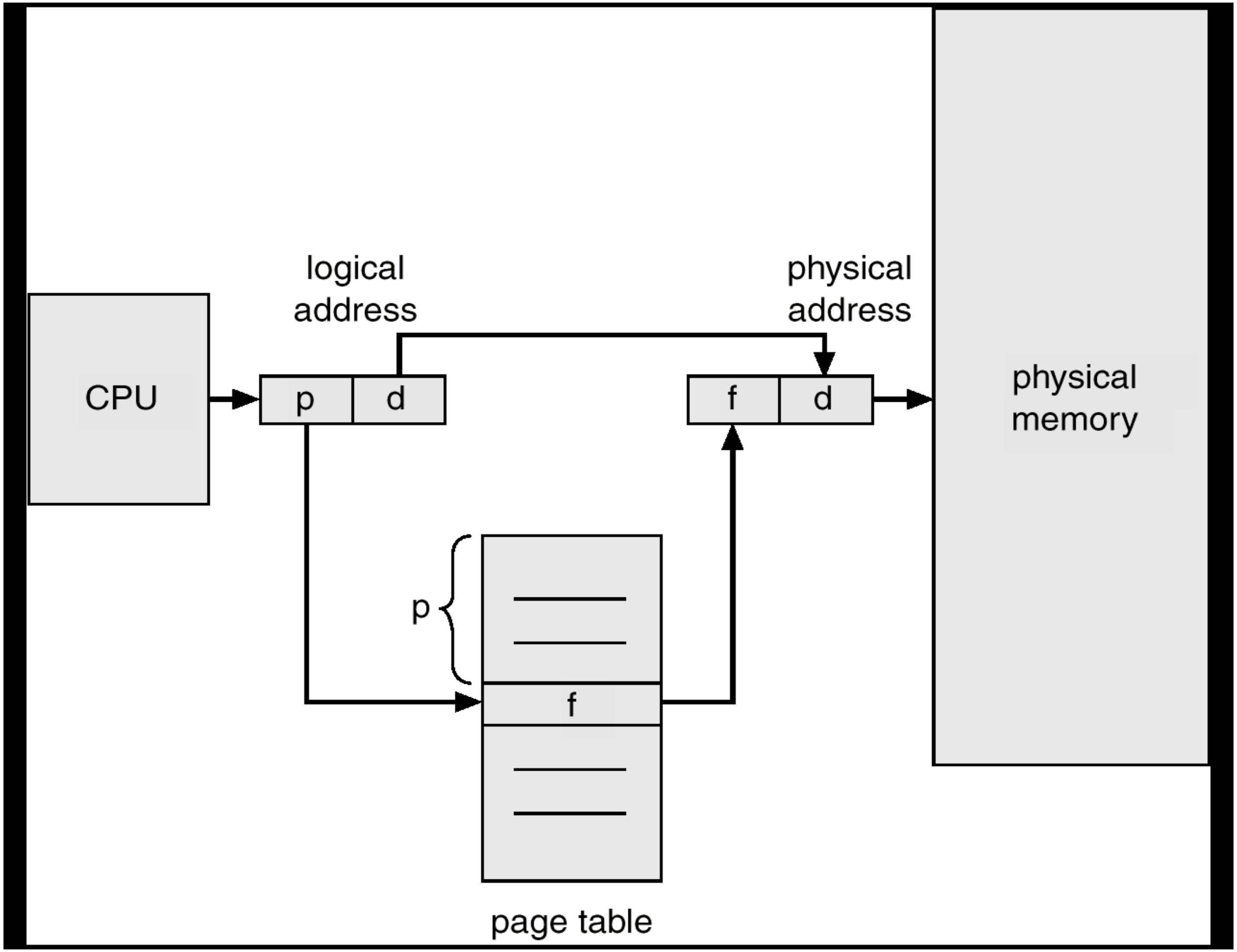

frame number
page 0
page 1
page 2
page 3
logical memory
0 1
1 4
2 3
3 7
page table
0
1 page 0
2
3 page 2
4 page 1
5
6
7 page 3
physical memory

Address generated by CPU is divided into:

Page number (p) – used as an index into a *page table* which contains base address of each page in physical memory.

Page offset (d) – combined with base address to define the physical memory address that is sent to the memory unit.

Address Translation Scheme



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* (PRLR) 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 registers* or *translation look-aside buffers (TLBs)*

Memory Protection

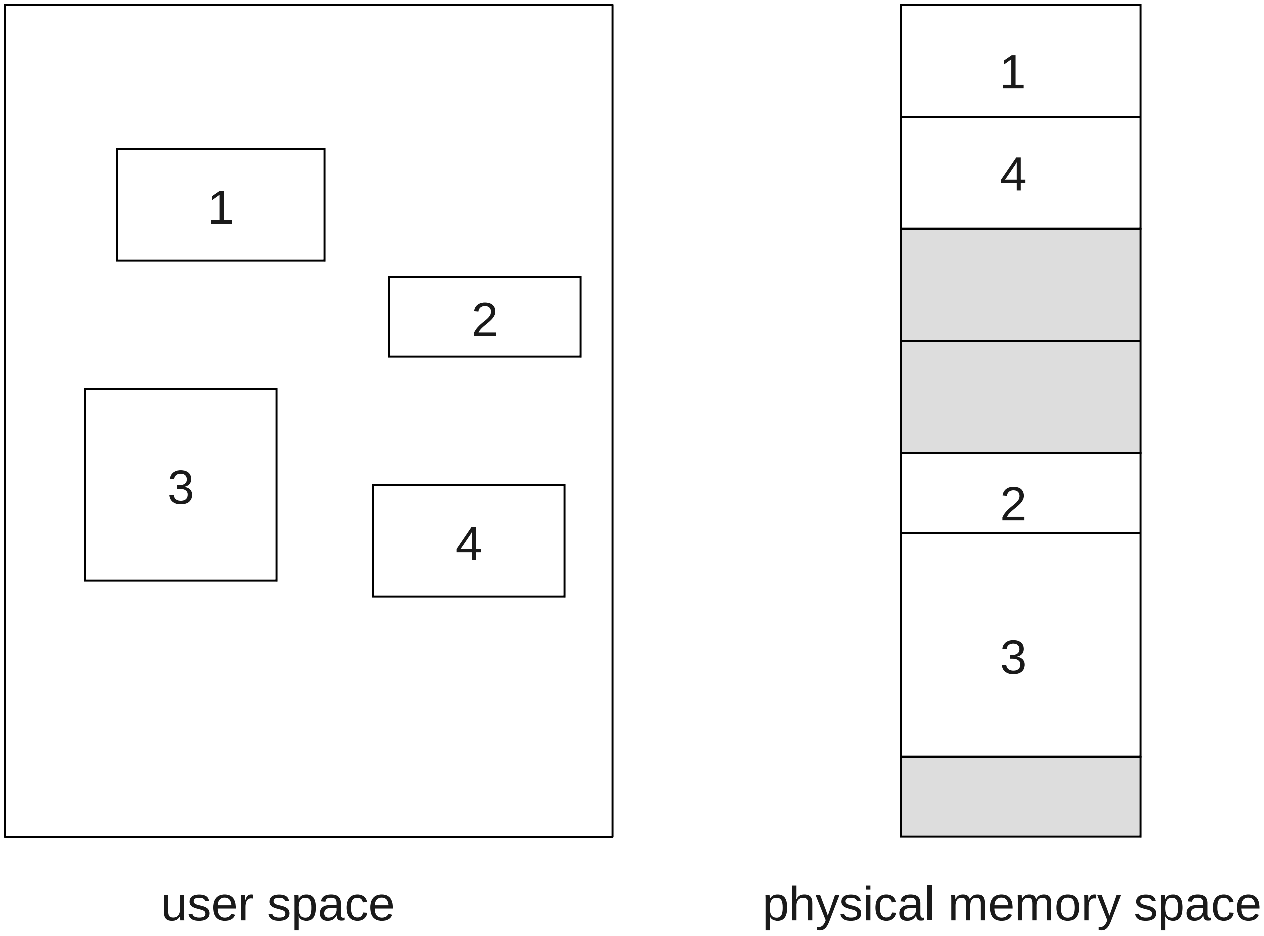
- Memory protection implemented by associating protection bit with each frame.
- *Valid-invalid* bit attached to each entry in the page table
- “valid” indicates that the associated page is in the process’ logical address space, and is thus a legal page.
- “invalid” indicates that the page is not in the process’ logical address space.

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,
    procedure,
    function,
    local variables, global variables,
    common block,
    stack,
    symbol table, arrays

Segmentation



Segmentation

- Logical address consists of a two tuple:

  $\langle$segment-number, offset$\rangle$
- *Segment table* – maps two-dimensional physical addresses.
- Each table entry has:

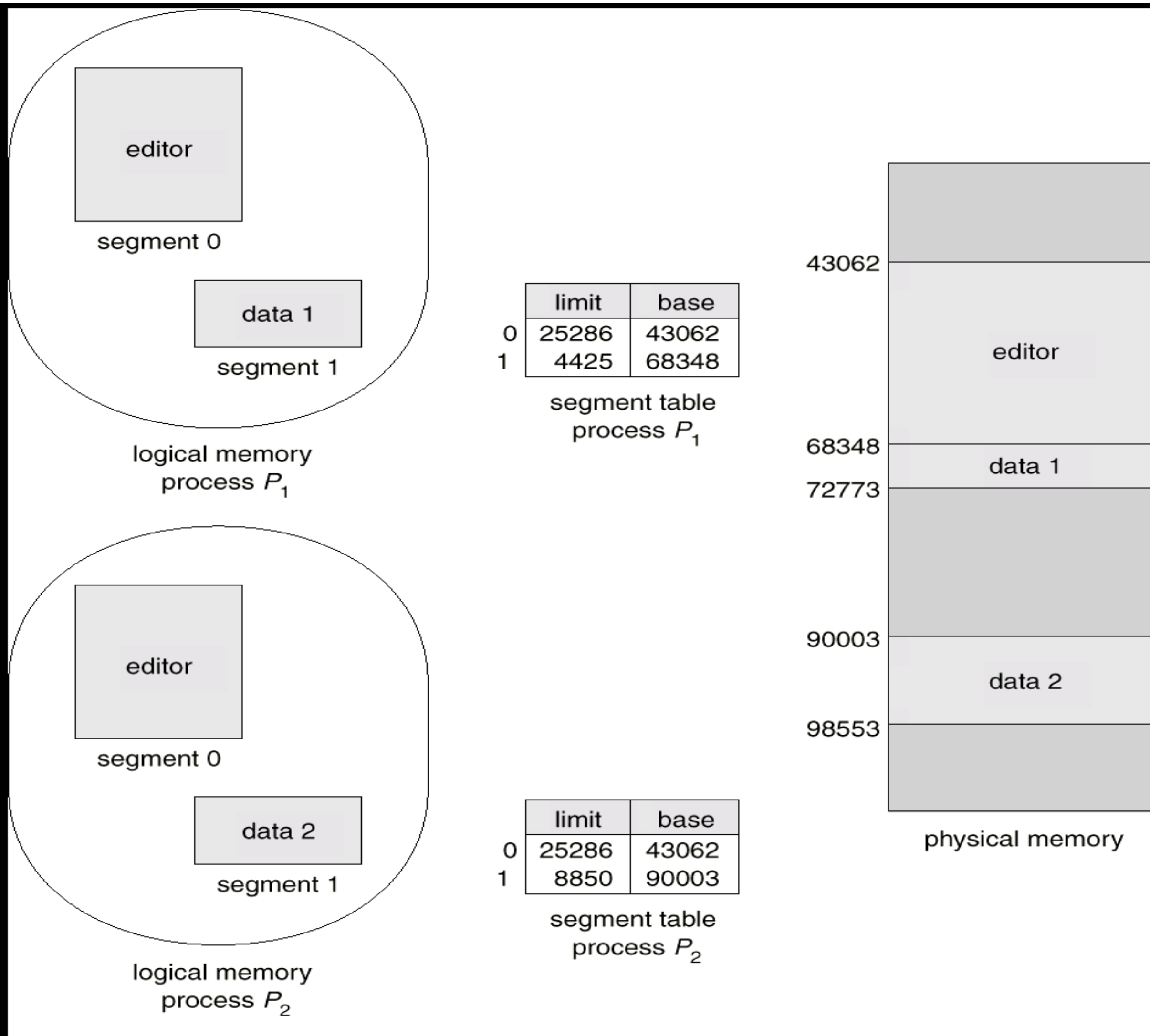
base – contains the starting physical address where the segments reside in memory.

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;

  segment number *s* is legal if s < STLR.

Segmentation



Segmentation Architecture

- Relocation – dynamic, by segment table
- Sharing - shared segments, same segment number
- Allocation - first fit/best fit, external fragmentation

Protection - With each entry in segment table associate: validation bit = 0 $\Rightarrow$ illegal segment read/write/execute privileges

Comparing Memory-Management Strategies

- Hardware support
- Performance
- Fragmentation
- Relocation
- Swapping
- Sharing
- Protection