Information Technology Institute

Operating System Fundamentals

© Copyright Information Technology Institute - 2018

Chapter Seven

Memory Management

© Copyright Information Technology Institute - 2018

Table of Content

© Copyright Information Technology Institute - 2018

LOGICAL VS PHYSICAL ADDRESS SPACE

© Copyright Information Technology Institute - 2018

Logical vs. Physical Address Space

- The concept of a logical *address space* that is bound to a separate *physical address space* is central to proper memory management.
  - *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 address-binding schemes; logical (virtual) and physical addresses differ in execution-time address-binding scheme.

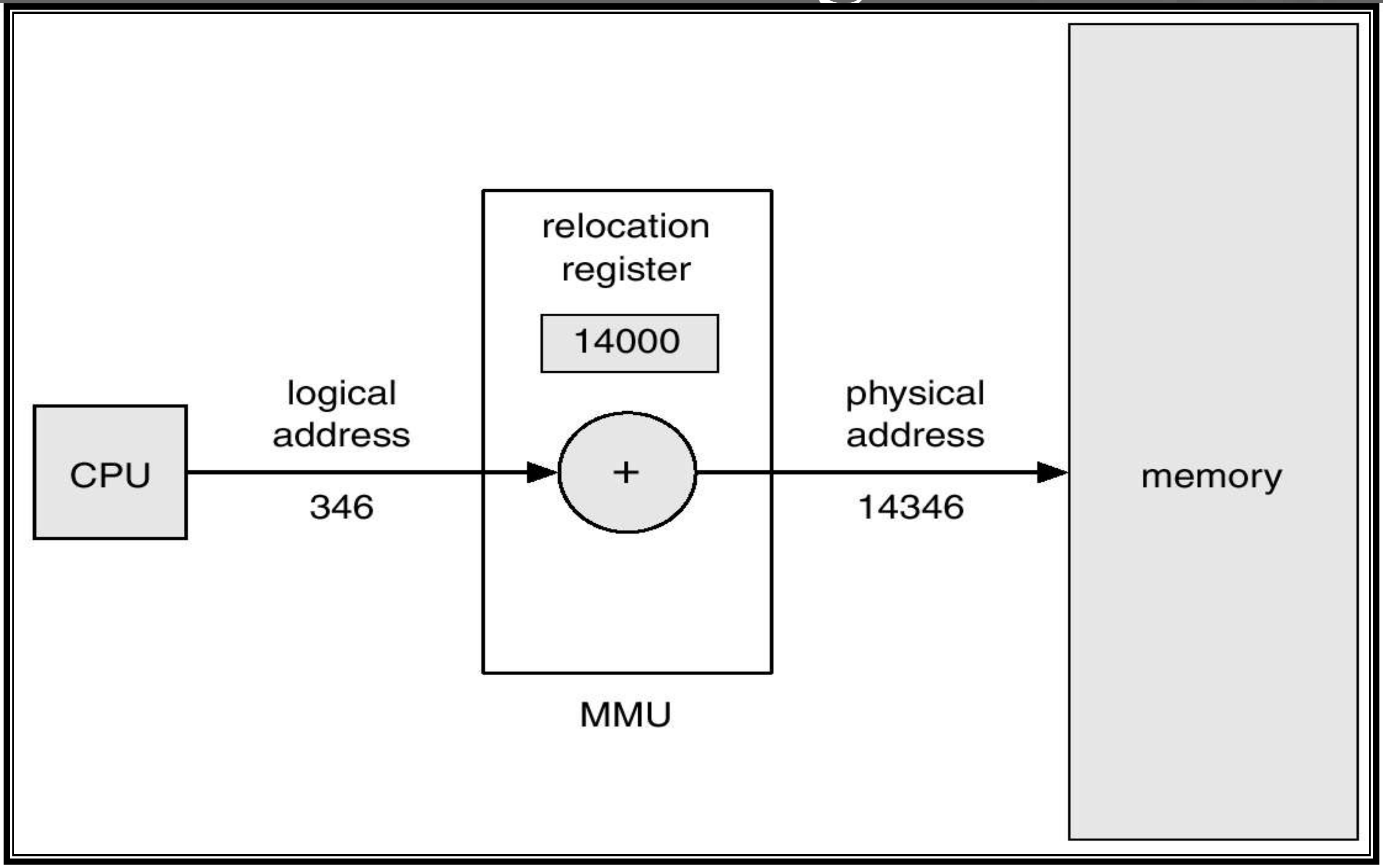
© Copyright Information Technology Institute - 2018

Memory-Management Unit (MMU)

- Hardware device that maps logical (virtual) to physical address.
- In MMU scheme, the value in the relocation register (base 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.

© Copyright Information Technology Institute - 2018

Dynamic relocation using a relocation register



© Copyright Information Technology Institute - 2018

SWAPPING

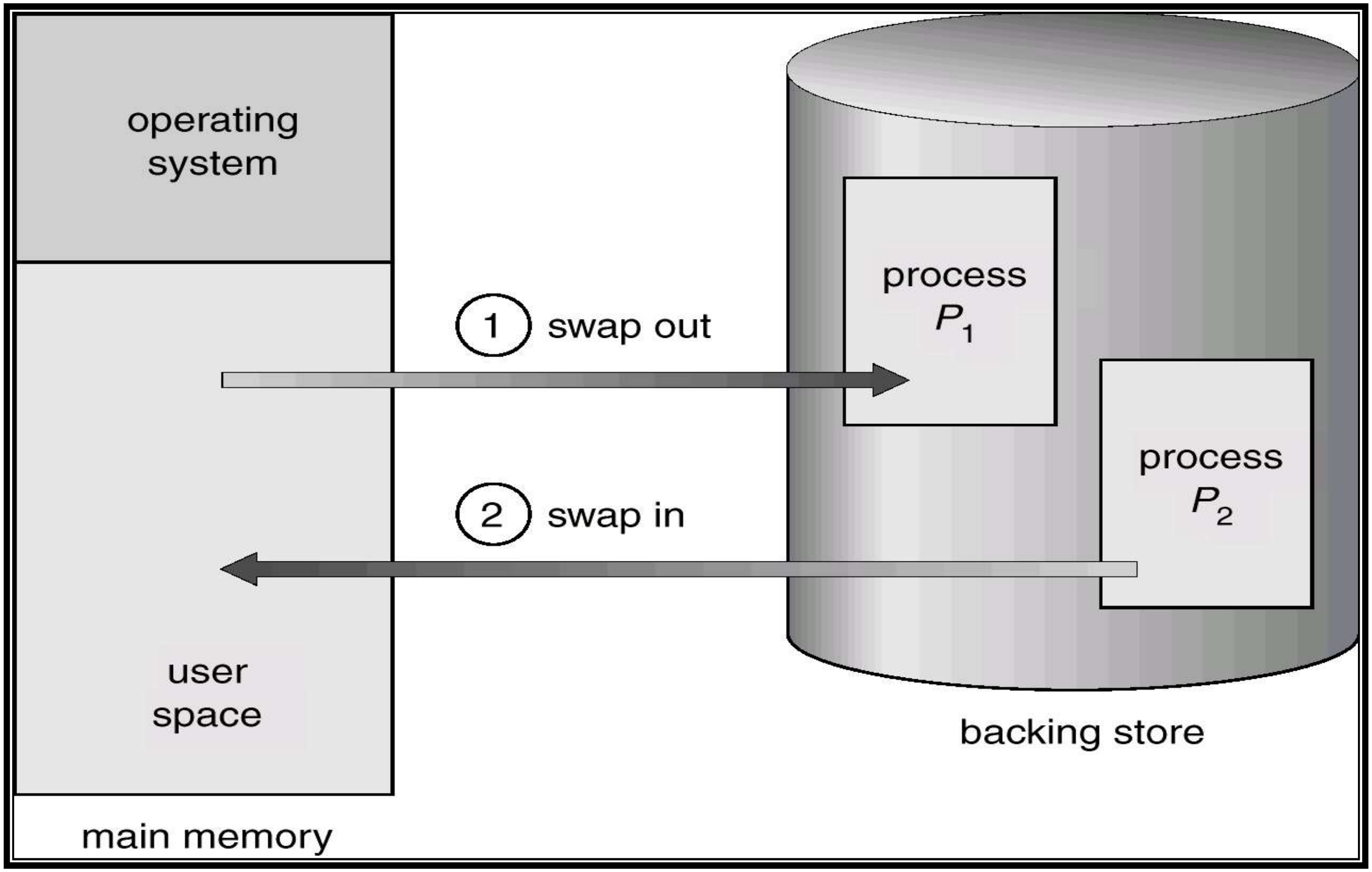
© Copyright Information Technology Institute - 2018

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, Linux, and Windows.

© Copyright Information Technology Institute - 2018

Schematic View of Swapping



© Copyright Information Technology Institute - 2018

CONTIGUOUS ALLOCATION

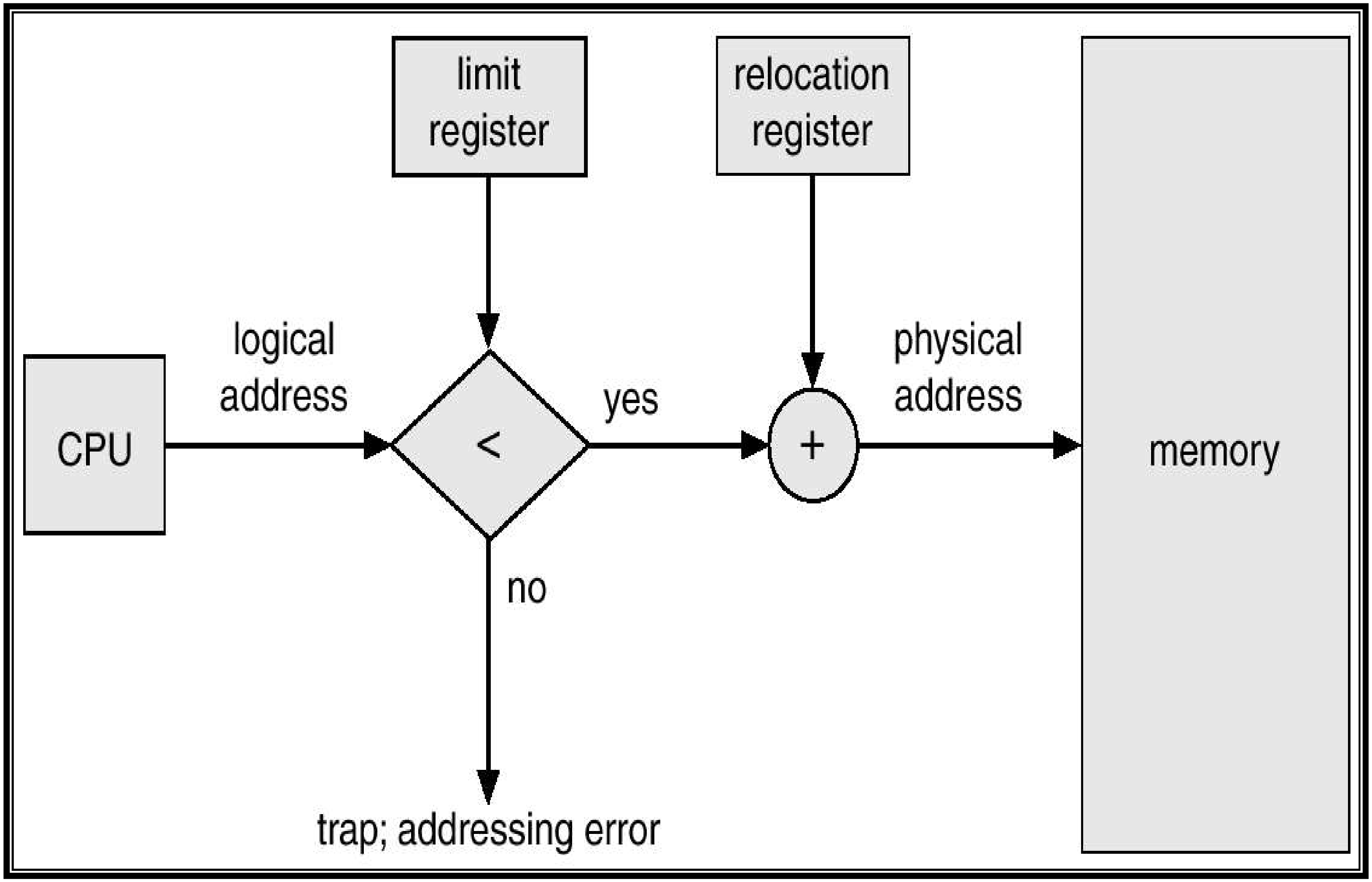
© Copyright Information Technology Institute - 2018

Contiguous Allocation

- Main memory usually 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.

© Copyright Information Technology Institute - 2018

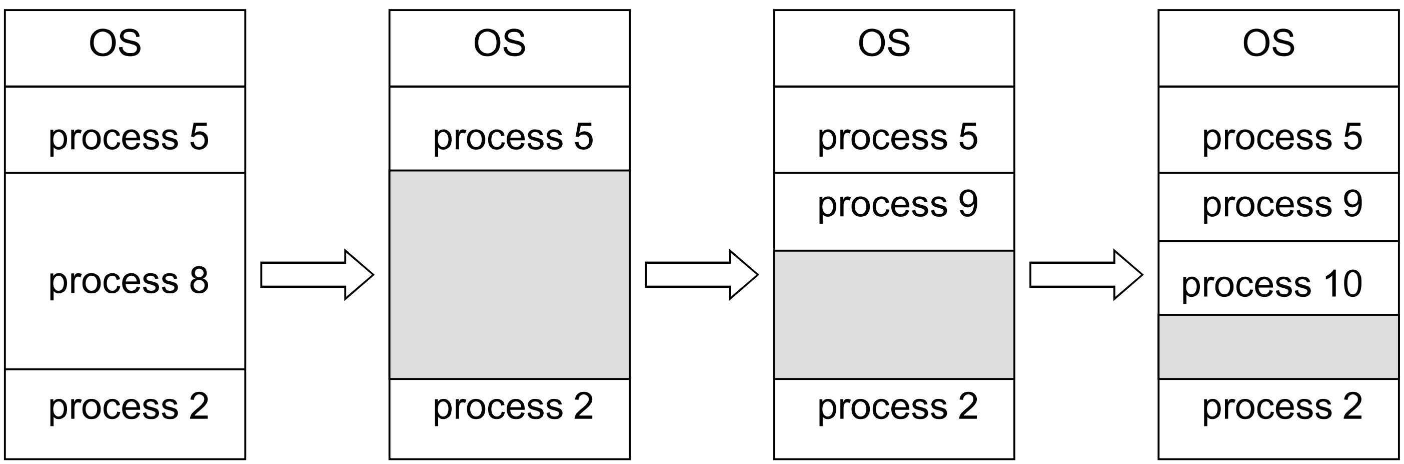
Hardware Support for Relocation and Limit Registers



© Copyright Information Technology Institute - 2018

Contiguous Allocation (Cont.)

- 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)



© Copyright Information Technology Institute - 2018

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.

© Copyright Information Technology Institute - 2018

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

© Copyright Information Technology Institute - 2018

PAGING

© Copyright Information Technology Institute - 2018

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 8192 bytes).
- 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.*

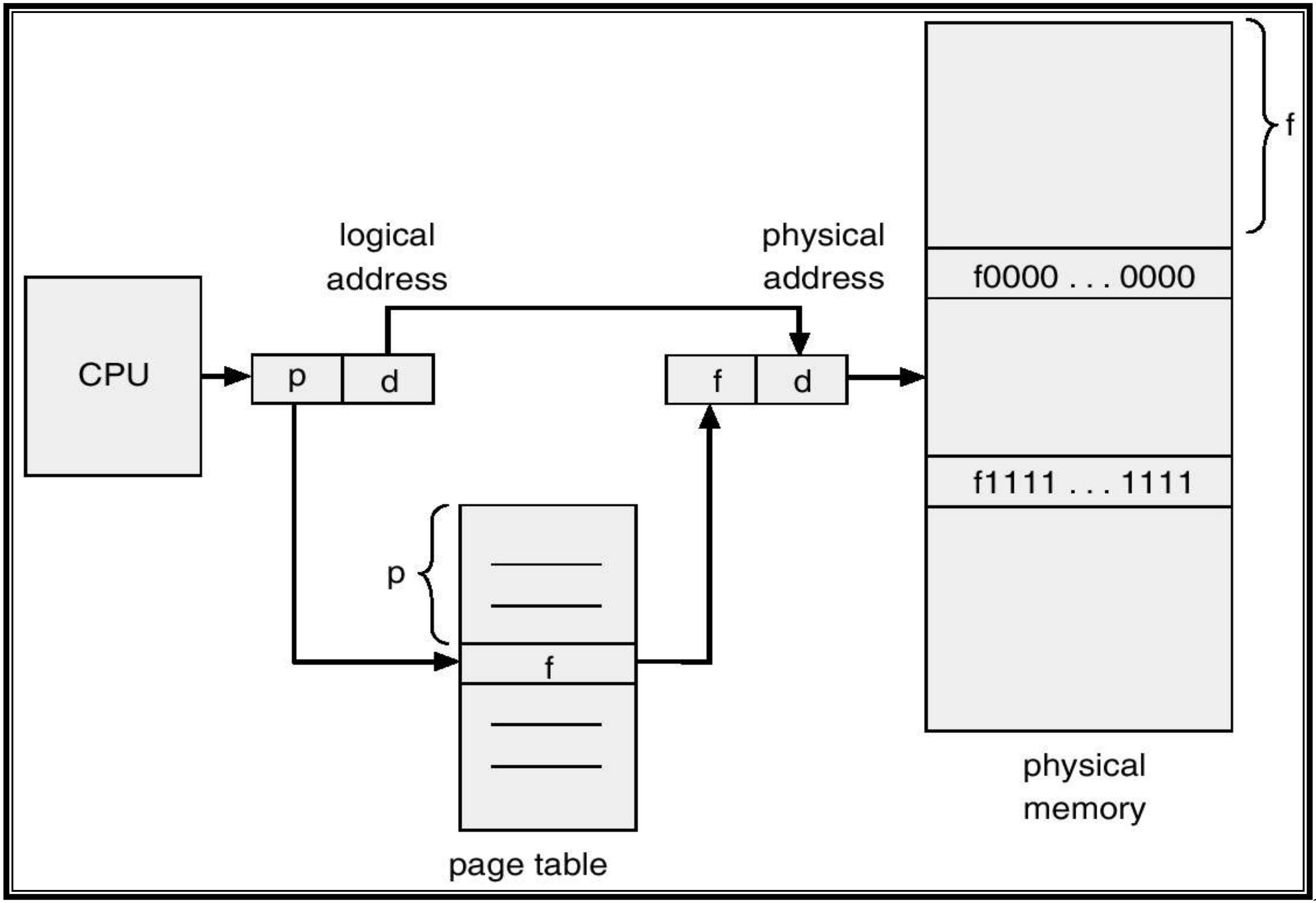
© Copyright Information Technology Institute - 2018

Address Translation Scheme

- 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.

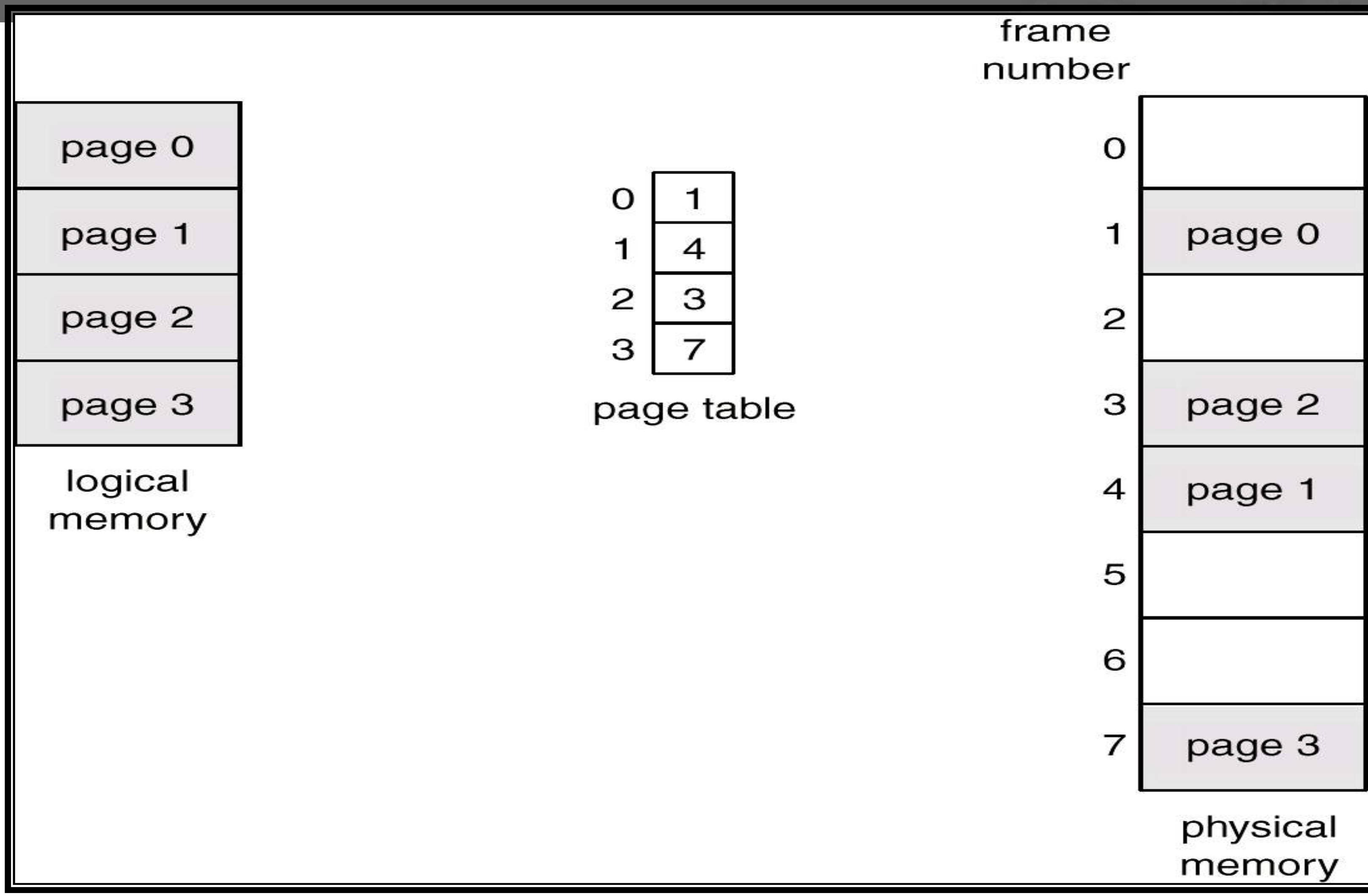
© Copyright Information Technology Institute - 2018

Address Translation Architecture



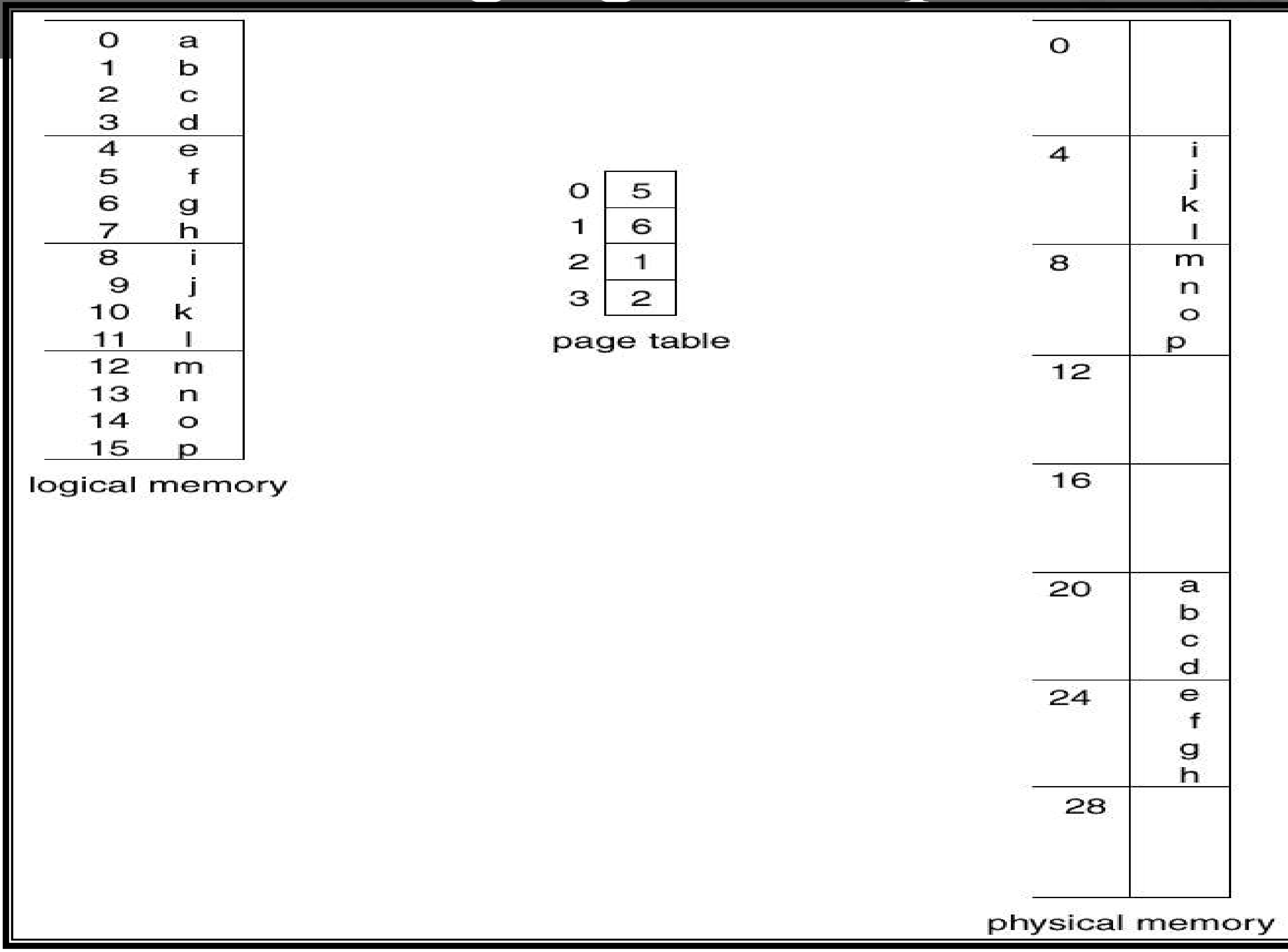
© Copyright Information Technology Institute - 2018

Paging Example



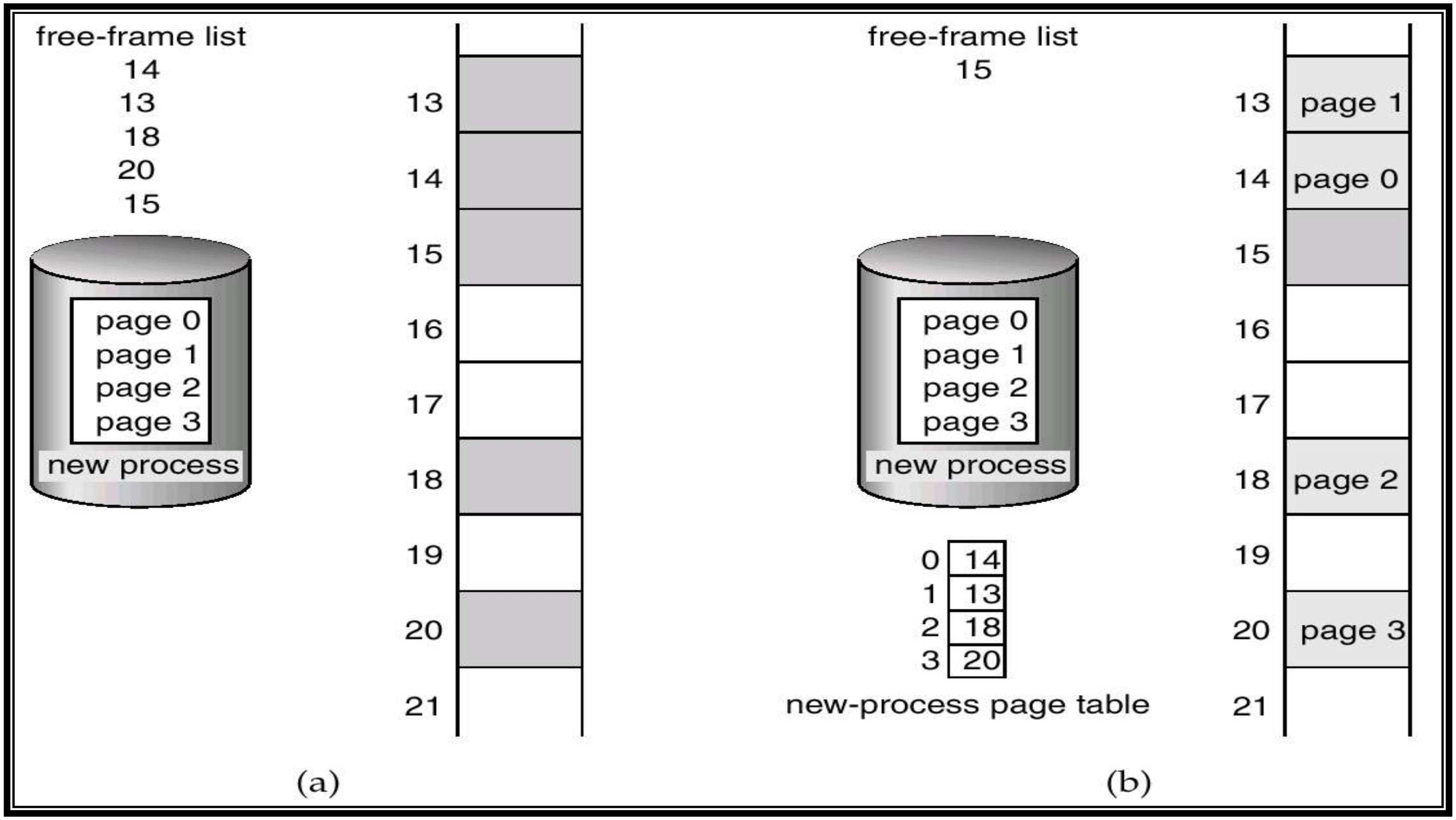
© Copyright Information Technology Institute - 2018

Paging Example



© Copyright Information Technology Institute - 2018

Free Frames



Before allocation

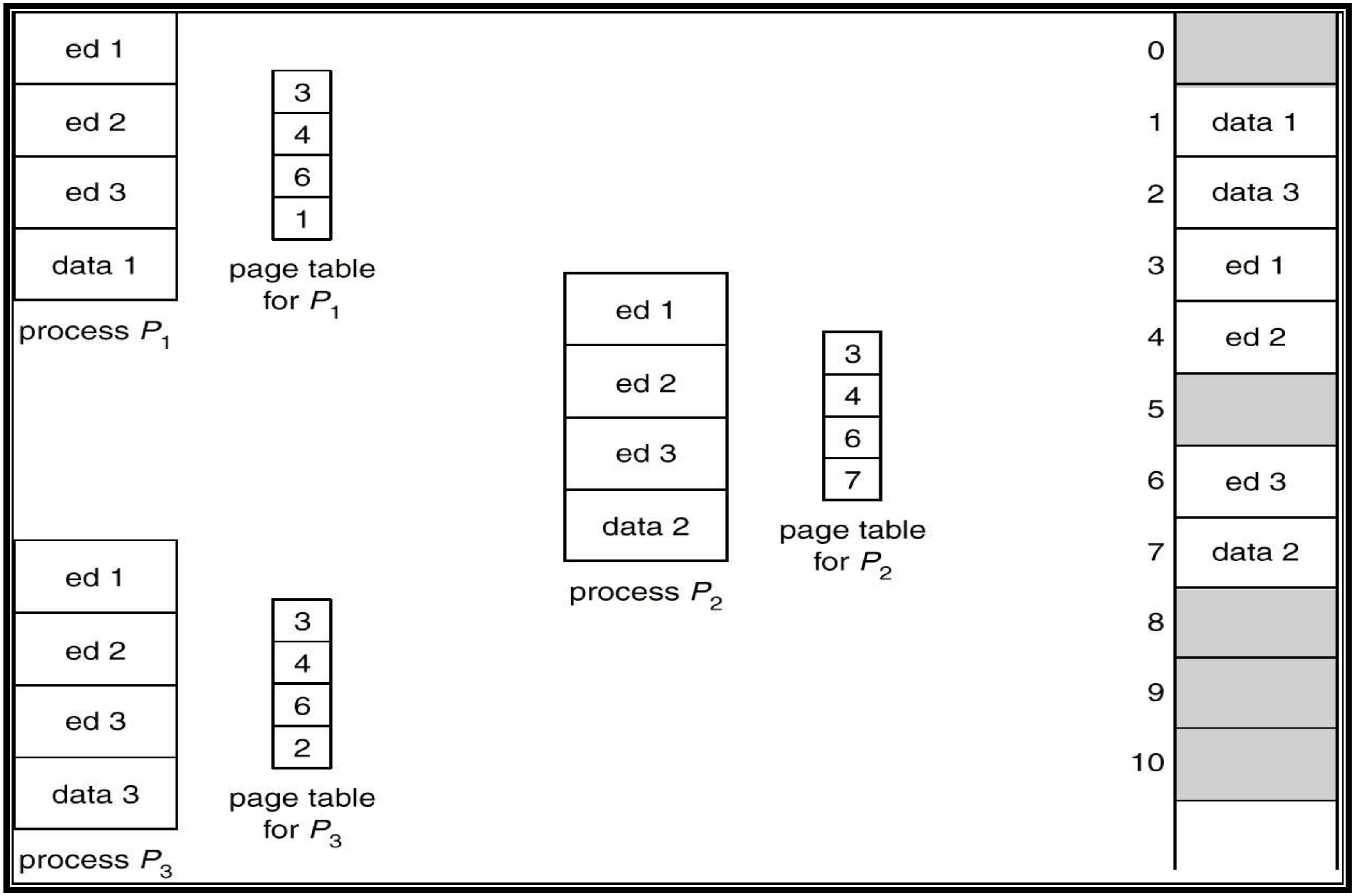
After allocation

Shared Pages

- Shared code
  - One copy of read-only (reentrant) code shared among processes (i.e., text editors, compilers, window systems).
  - Shared code must appear in same location in the logical address space of all processes.

© Copyright Information Technology Institute - 2018

Shared Pages Example



SEGMENTATION

© Copyright Information Technology Institute - 2018

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,

  method,

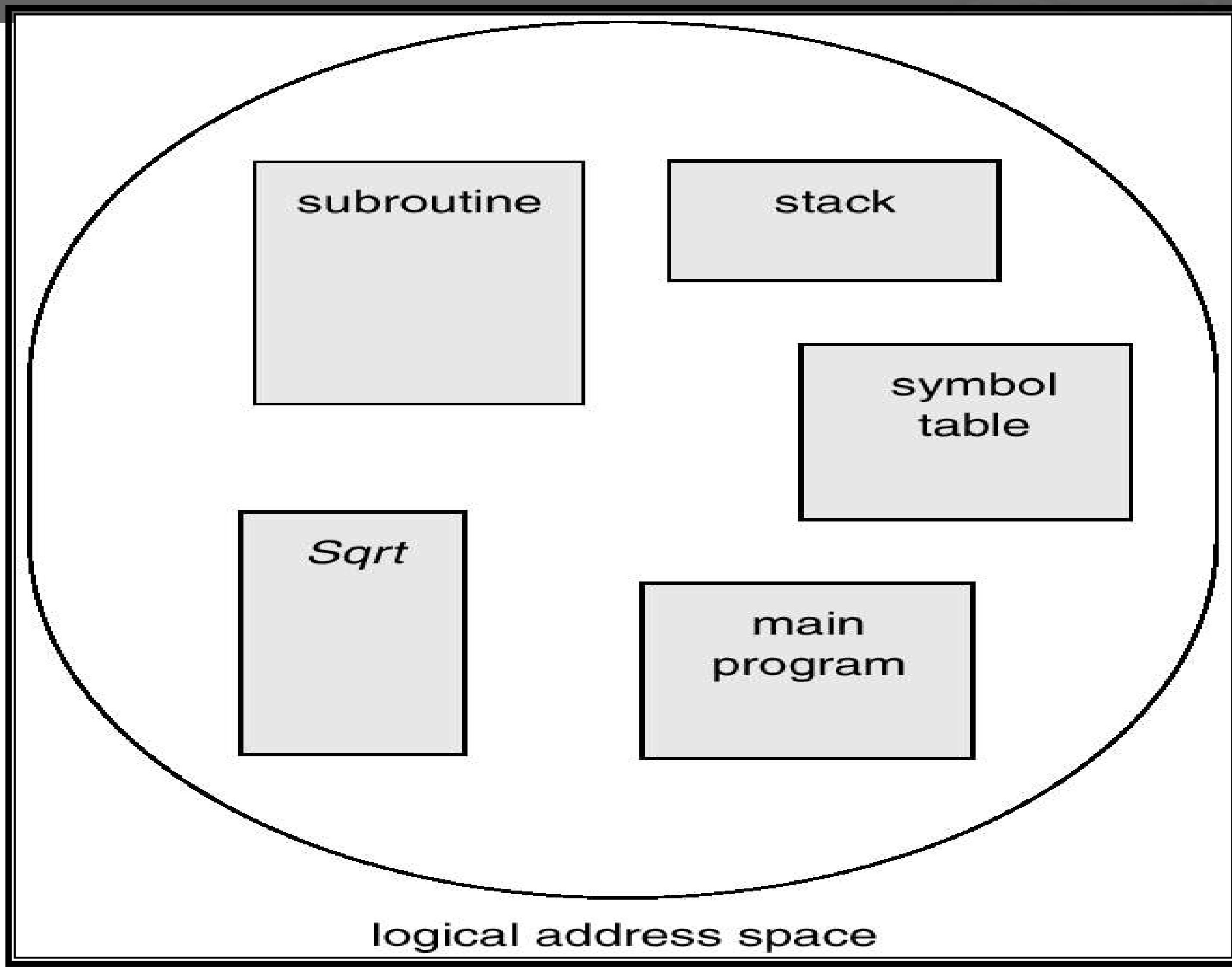
  common block,

  stack,

  symbol table, arrays

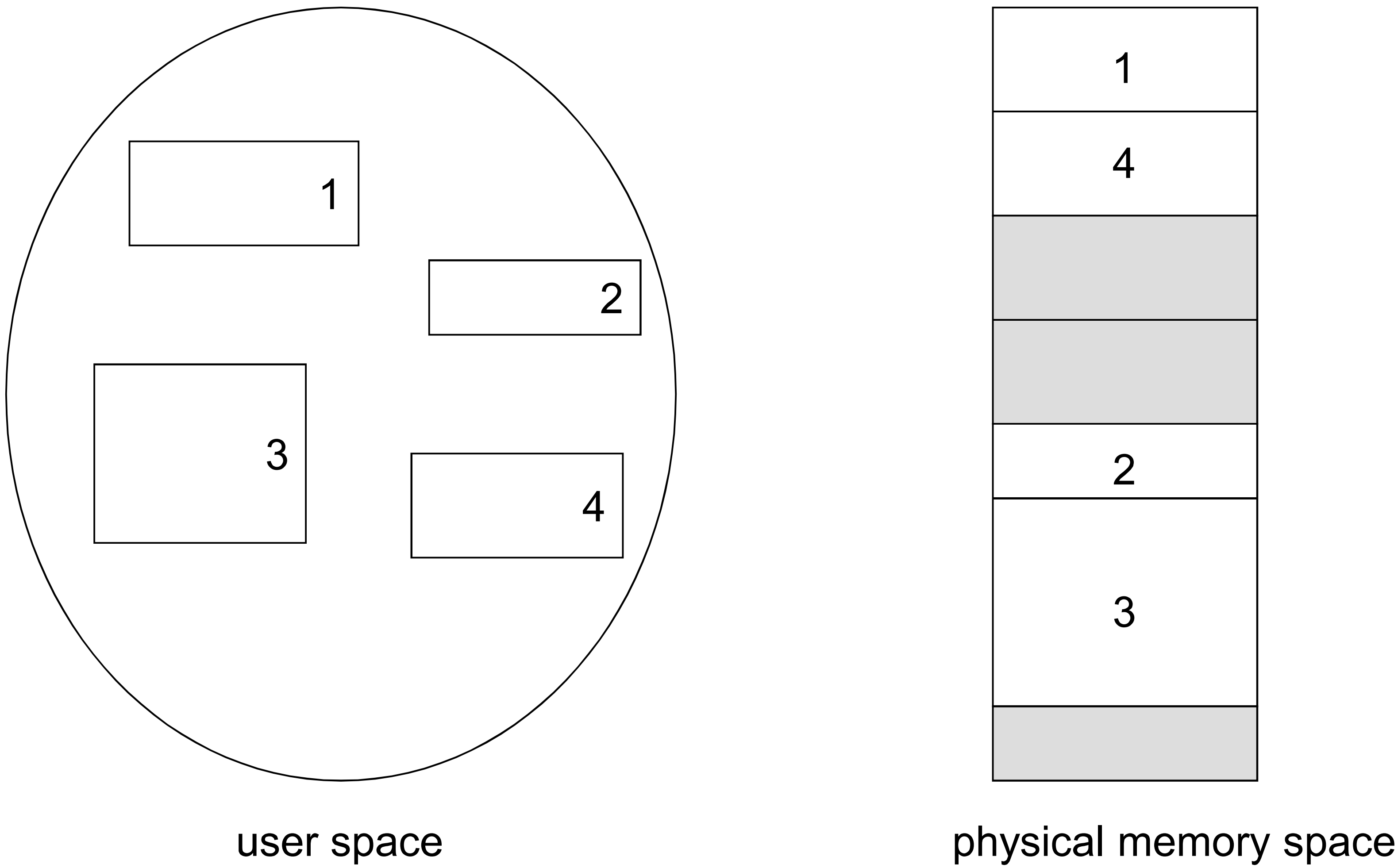
© Copyright Information Technology Institute - 2018

User's View of a Program



© Copyright Information Technology Institute - 2018

Logical View of Segmentation



© Copyright Information Technology Institute - 2018

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.

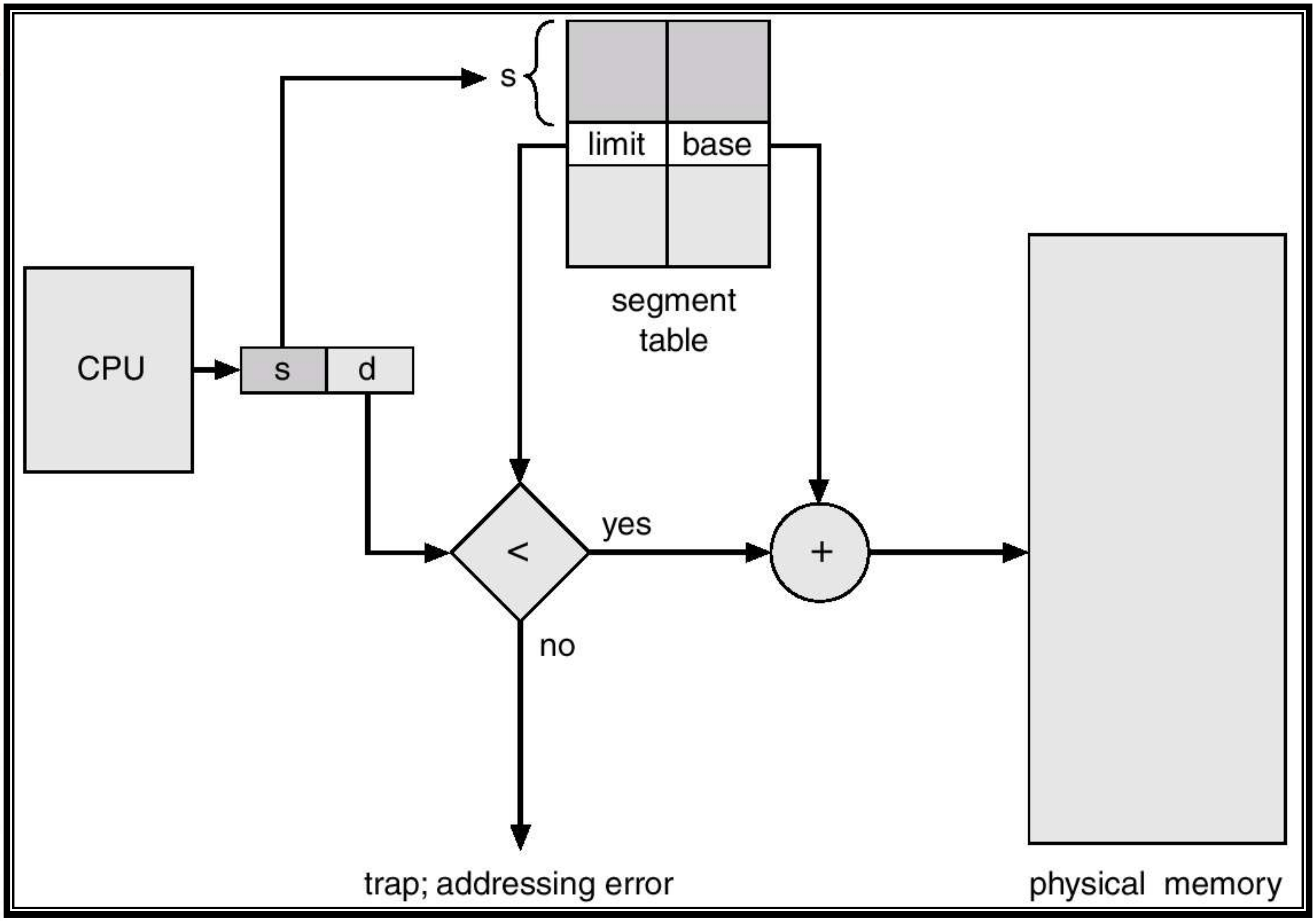
© Copyright Information Technology Institute - 2018

Segmentation Architecture (Cont.)

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

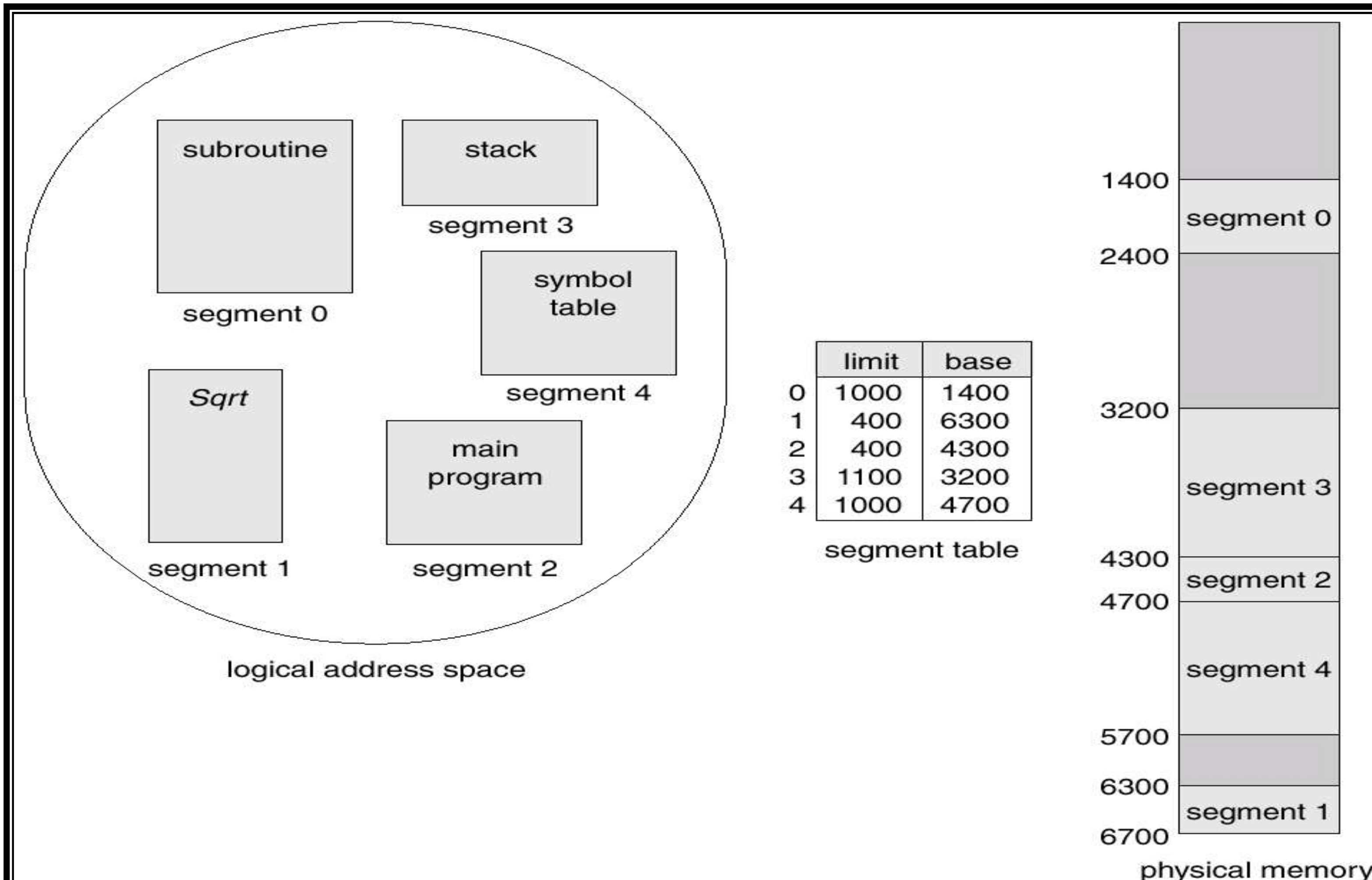
© Copyright Information Technology Institute - 2018

Segmentation Hardware



© Copyright Information Technology Institute - 2018

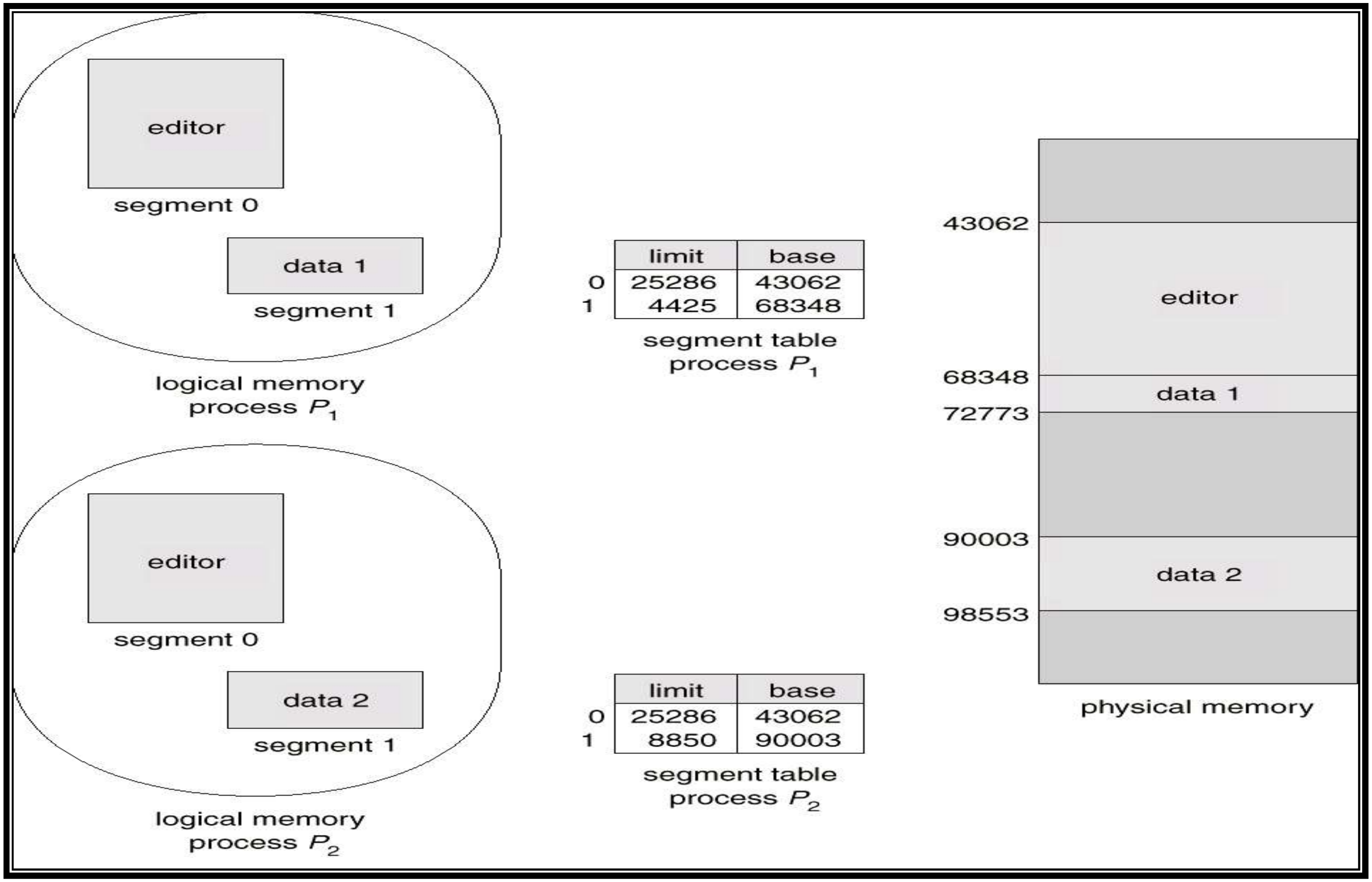
Example of Segmentation



	limit	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

© Copyright Information Technology Institute - 2018

Sharing of Segments



© Copyright Information Technology Institute - 2018

SEGMENTATION WITH PAGING

© Copyright Information Technology Institute - 2018

Segmentation with Paging – MULTICS

- The MULTICS system solved problems of external fragmentation and lengthy search times by paging the segments.

- Solution differs from pure segmentation in that the segment-table entry contains not the base address of the segment, but rather the base address of a *page table* for this segment.

© Copyright Information Technology Institute - 2018

© Copyright Information Technology Institute - 2018