Main Memory IV

Segmentation

- Memory-management scheme that supports user view of memory
- A process is a collection of segments
  - A segment is a logical unit such as:

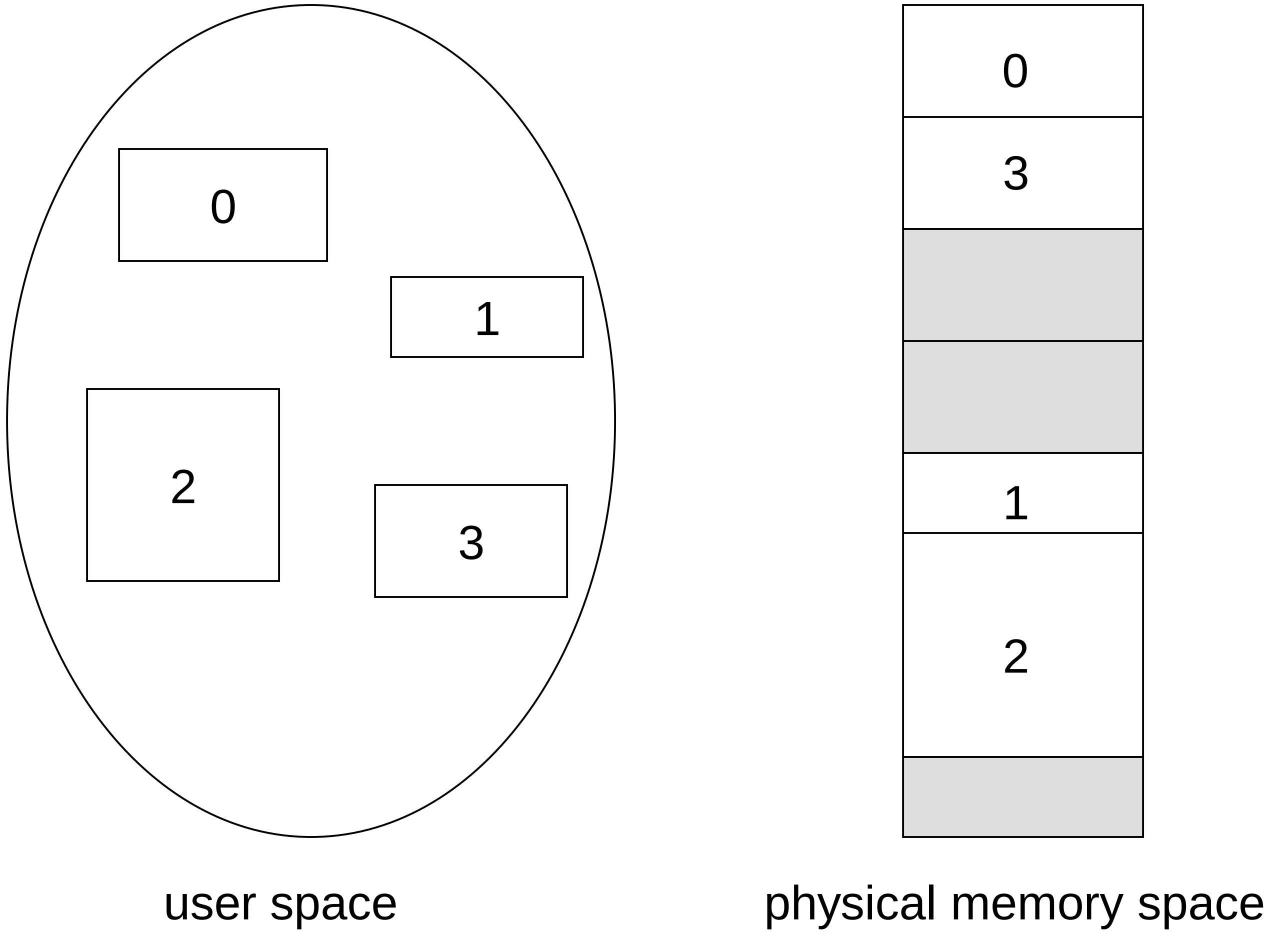
    code (Read-Only & Executable)

    data (global and static variables; R/W)

    stack (local variables; R/W)

    heap (dynamically allocated variables; R/W)

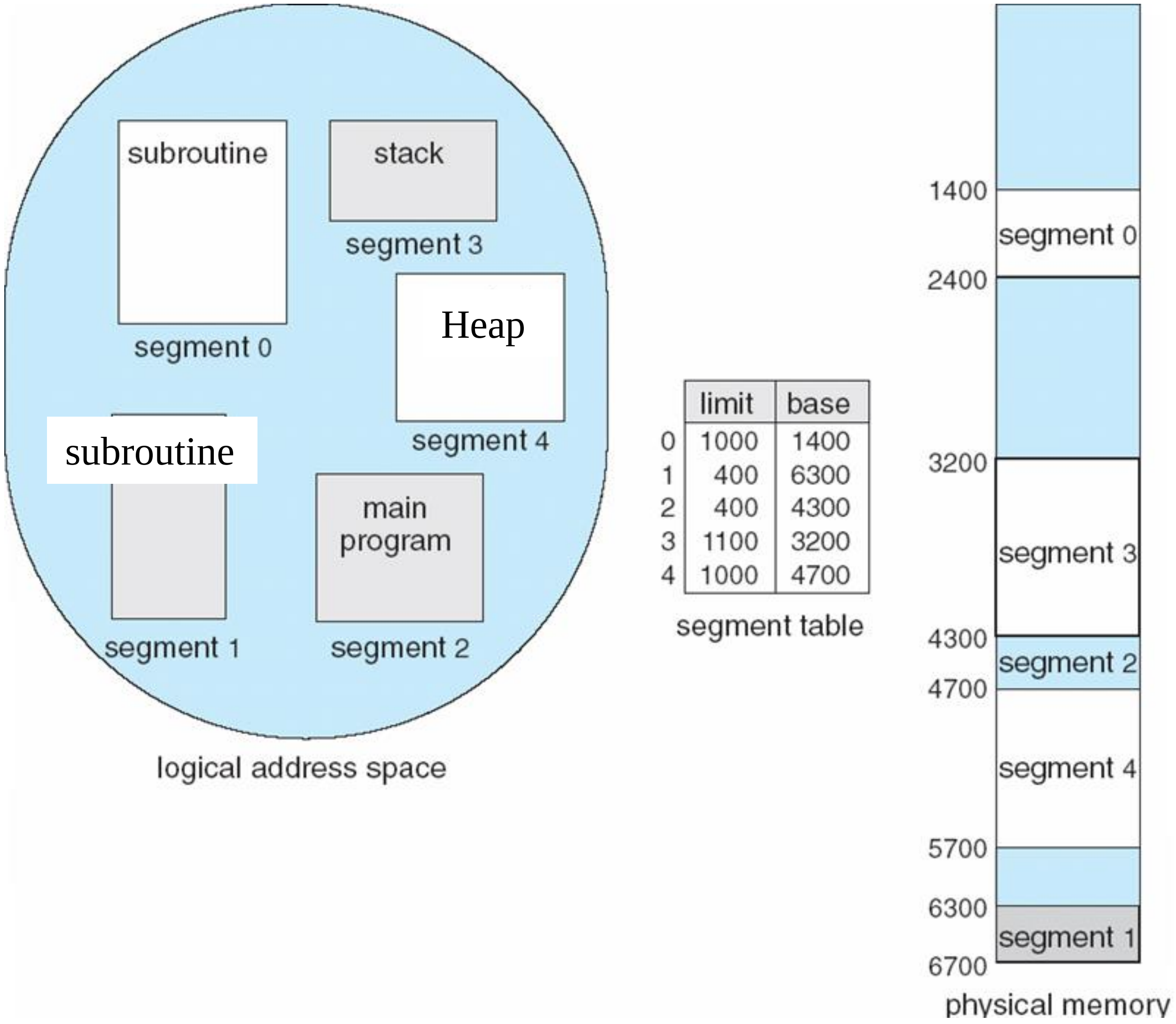
Logical View of Segmentation



Segmentation Architecture

- **Segment table** – maps segments to memory; each 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

Example of Segmentation



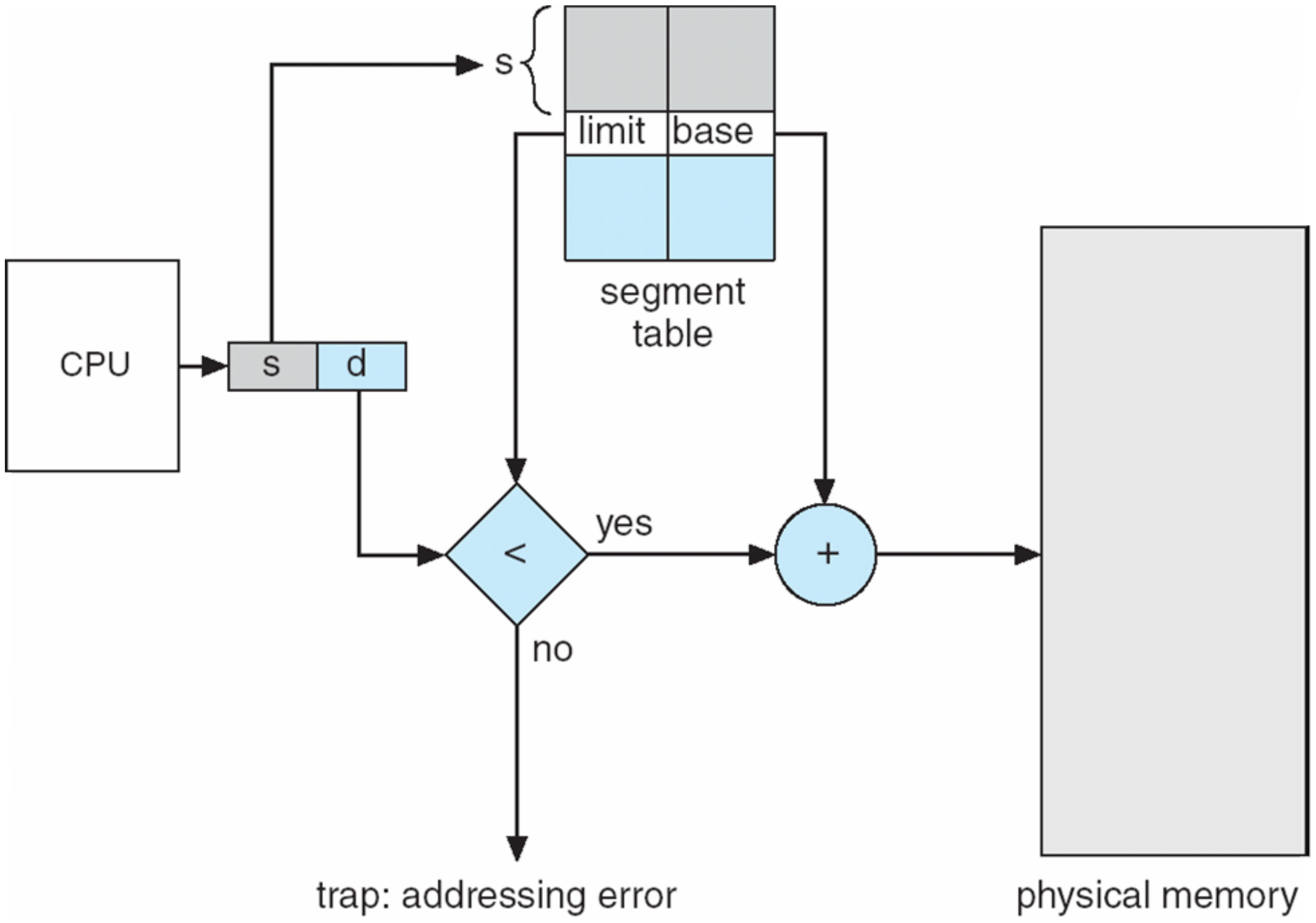
	limit	base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

segment table

Segmentation Architecture

- Logical address consists of a two tuple: <segment-number, offset>
- Segment number ***s*** is legal if ***s*** < **STLR**

Segmentation Hardware



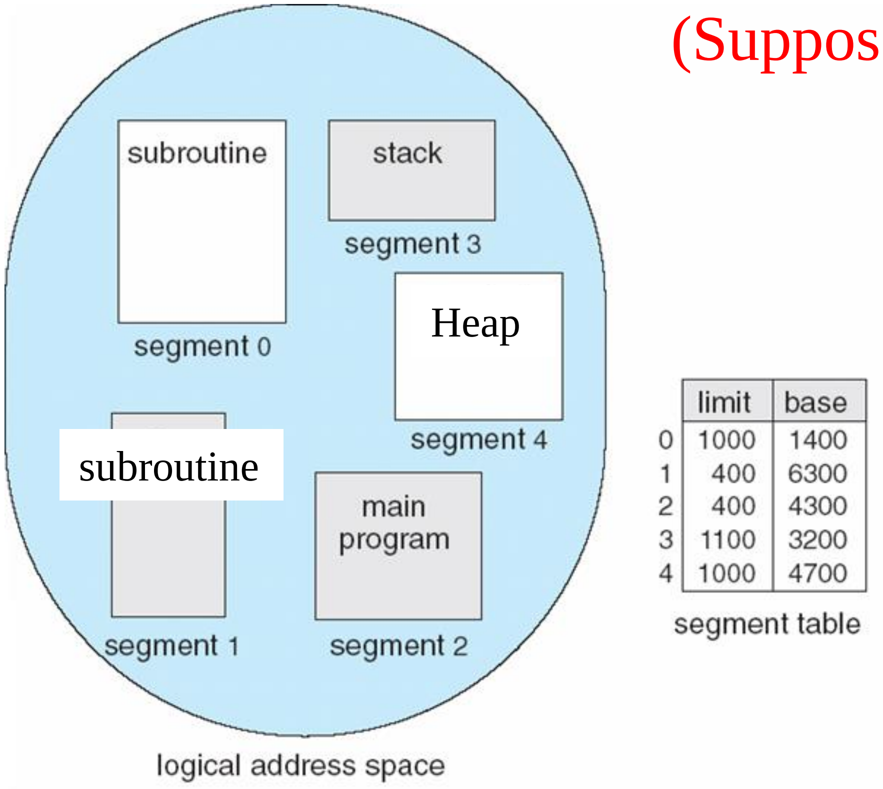
Segmentation Architecture

- Protection
  - With each entry in segment table associate:
    - read/write/execute privileges

- Since segments vary in length, memory allocation is a dynamic storage-allocation problem
- Could have external fragmentation problem: To address the problem, combination of segmentation and paging may be applied

Example of Segmentation/Paging

(Suppose Page size = 512 bytes)



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

segment table

Segment Table

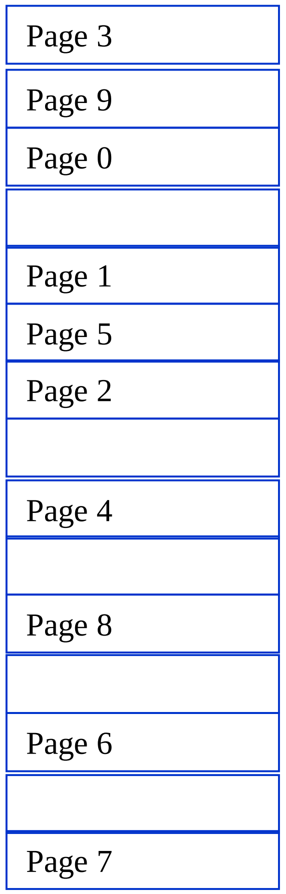
	limit	base
0	1024	0
1	512	1024
2	512	1536
3	1536	2048
4	1024	3584

The size of each segment of the process is expanded to be a multiple of page size; then, all the segments is mapped to a contiguous logical address space starting from 0.

Example of Segmentation/Paging

Page Table

Page #	Frame #
0	2
1	4
2	6
3	0
4	8
5	5
6	12
7	14
8	10



Segment Table

	limit	base
0	1024	0
1	512	1024
2	512	1536
3	1536	2048
4	1024	3584

Page size = 512 bytes

Each segment is paged and mapped to physical memory

Exercises: Given the segment/page tables,
(1) translate logical address (2,3) to physical address;
(2) translate physical address 2050 to logical address.

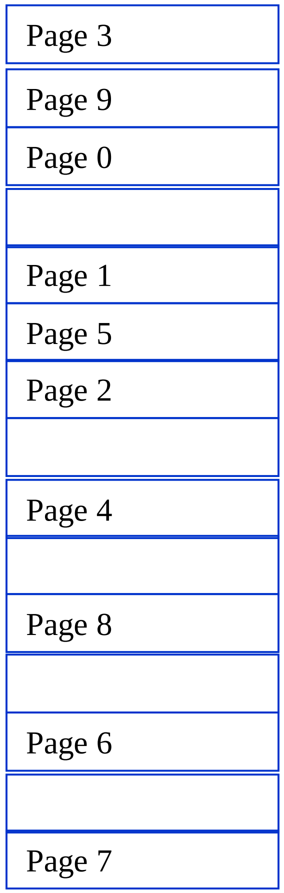
Segment Table

	limit	base
0	1024	0
1	512	1024
2	512	1536
3	1536	2048
4	1024	3584

Page size = 512 bytes

Page Table

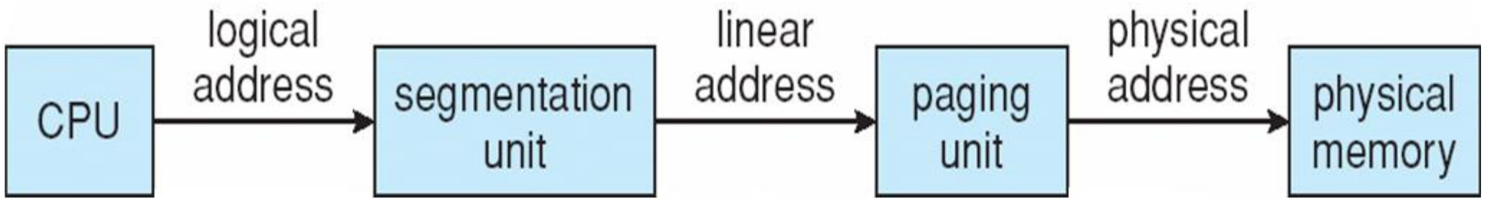
Page #	Frame #
0	2
1	4
2	6
3	0
4	8
5	5
6	12
7	14
8	10

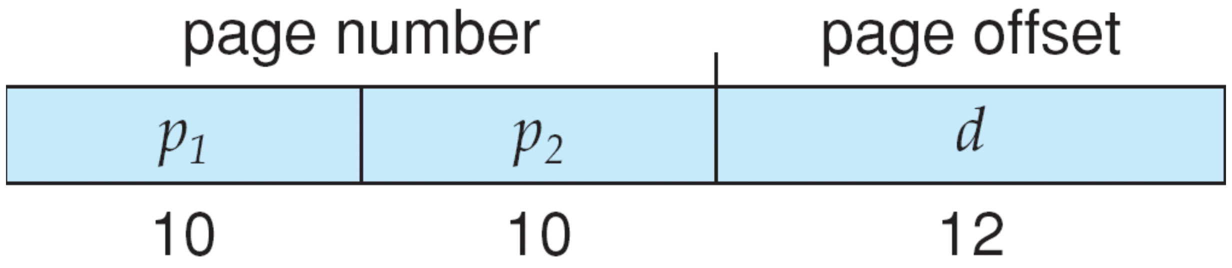


Example: The Intel Pentium

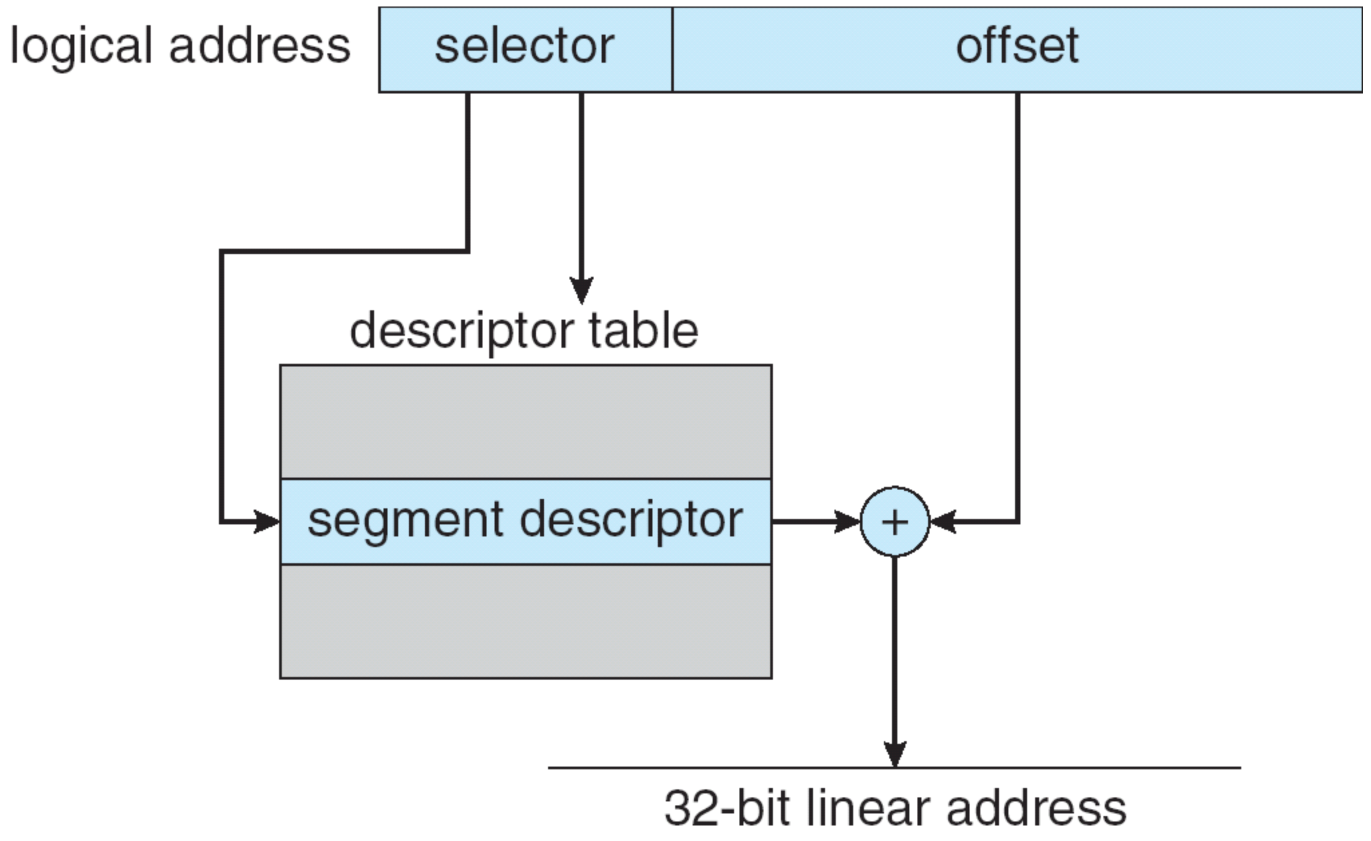
- Supports both segmentation and segmentation with paging
- CPU generates logical address
  - It is given to segmentation unit, which produces linear addresses
  - The linear address given to paging unit, which generates physical address in main memory

Logical to Physical Address Translation in Pentium

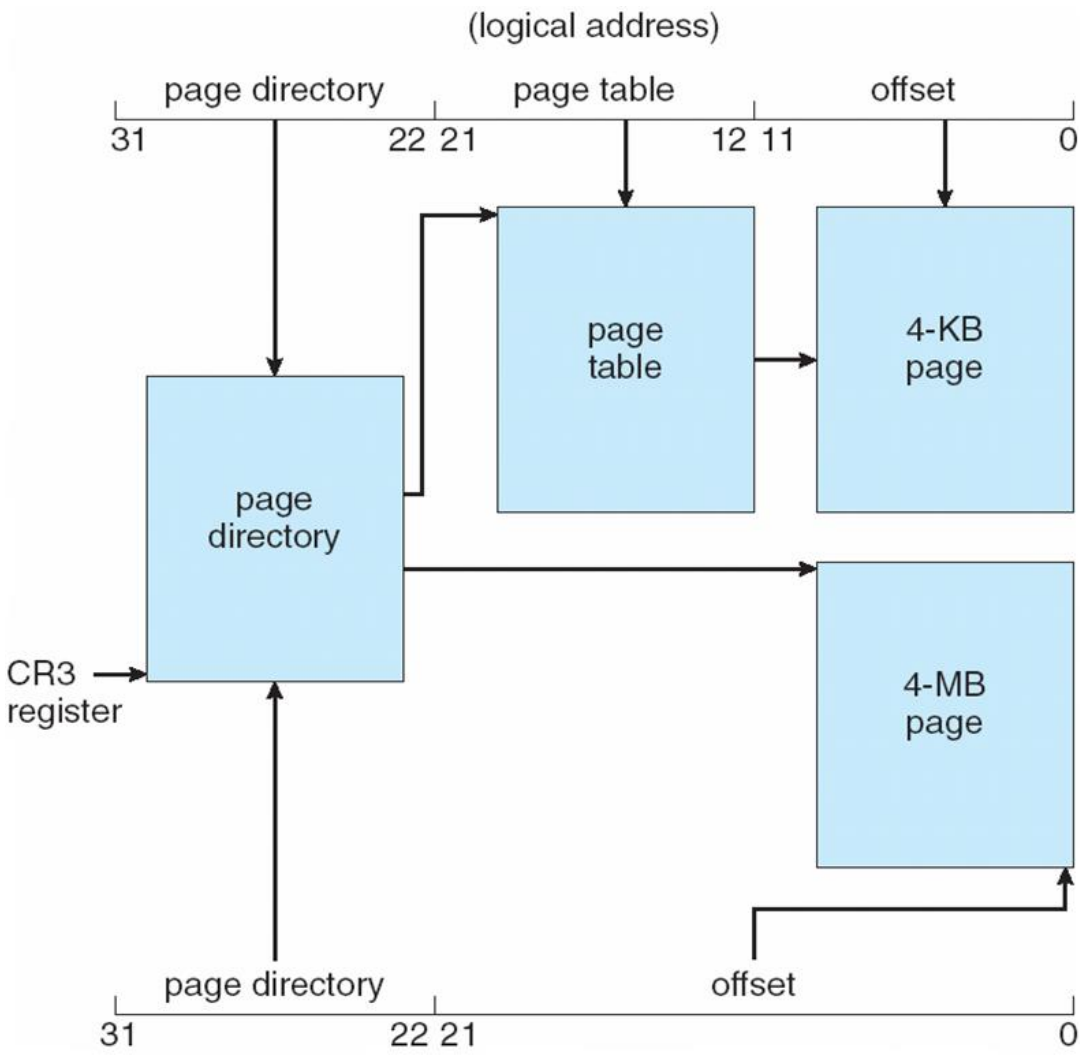




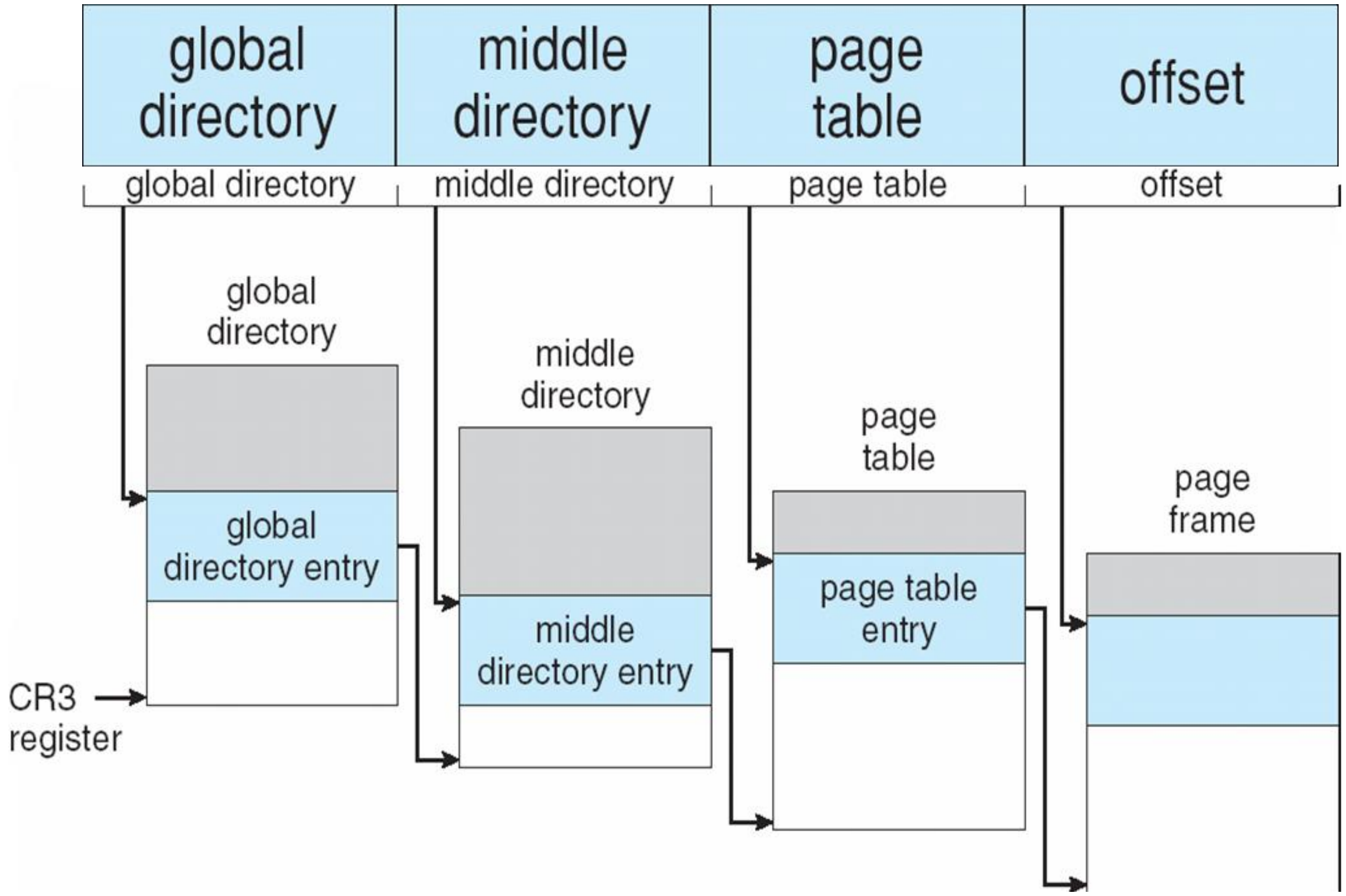
Intel Pentium Segmentation



Pentium Paging Architecture



Three-level Paging in Linux


global directory
middle directory
page table
offset
global directory
middle directory
page table
offset
global directory
global directory entry
middle directory
middle directory entry
page table
page table entry
page frame
CR3 register