Computer Organization and Software Systems

Session 14

Dr. Lucy J. Gudino
WILP & Department of CS & IS

BITS Pilani
Pilani Campus

innovate achieve lead

BITS Pilani

Pilani Campus

Memory Management

Recap - Memory Management Requirements

- Five requirements
  - Relocation
  - Protection
  - Sharing
  - Logical organization
  - Physical organization

Fragmentation

- **fragmentation** is a phenomenon in which storage space is used inefficiently, reducing capacity and often performance.
  - **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

- Paging permits the physical address space of a process to be non- contiguous.
- Divide physical memory into fixed-sized blocks called **frames** (size is power of 2).
- 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.
- Keep track of all free frames.
- Set up a page table to translate logical to physical addresses.

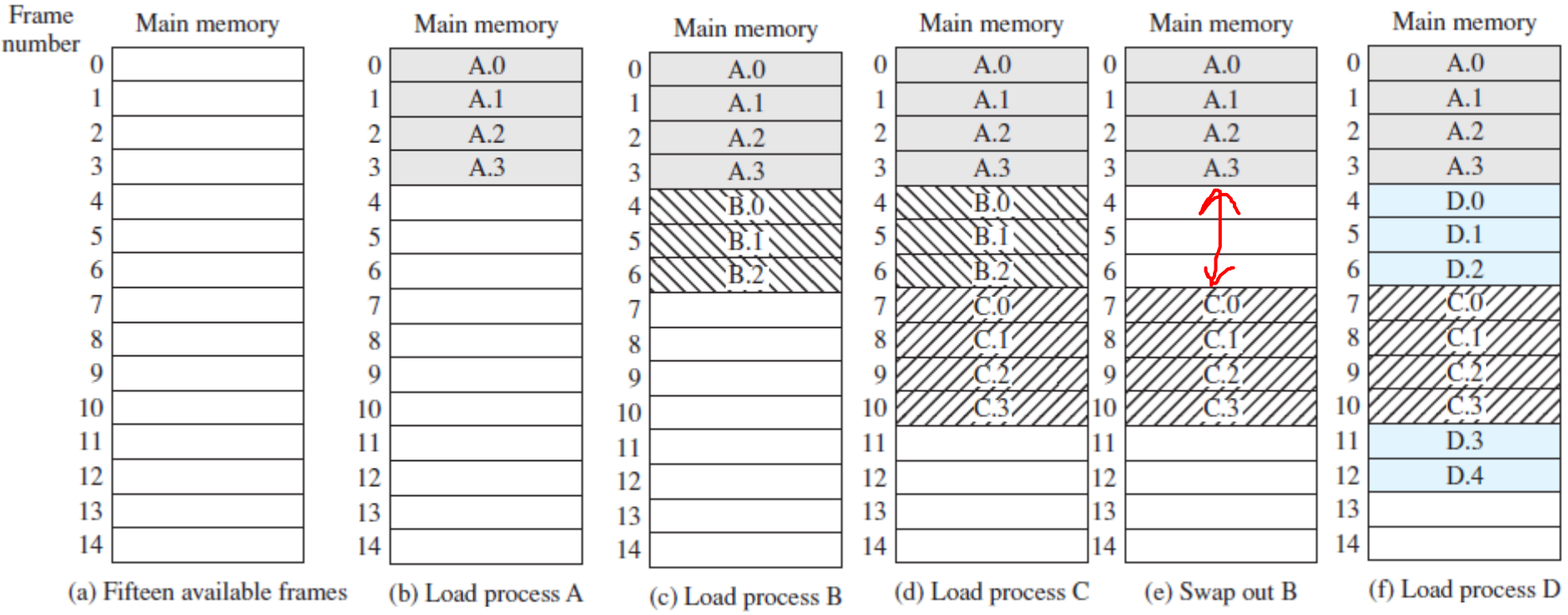
Example

Process A : 4 pages
Process B : 3 pages
Process C : 4 Pages
Process D : 5 Pages

LA

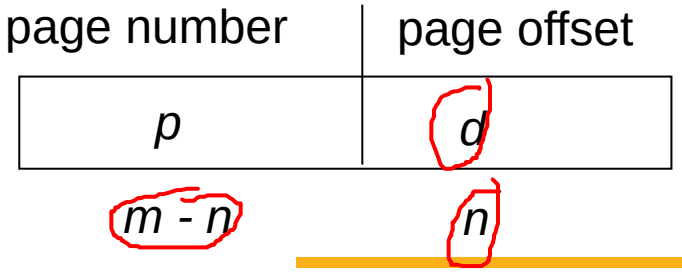
Main Memory : 15 frames

PA

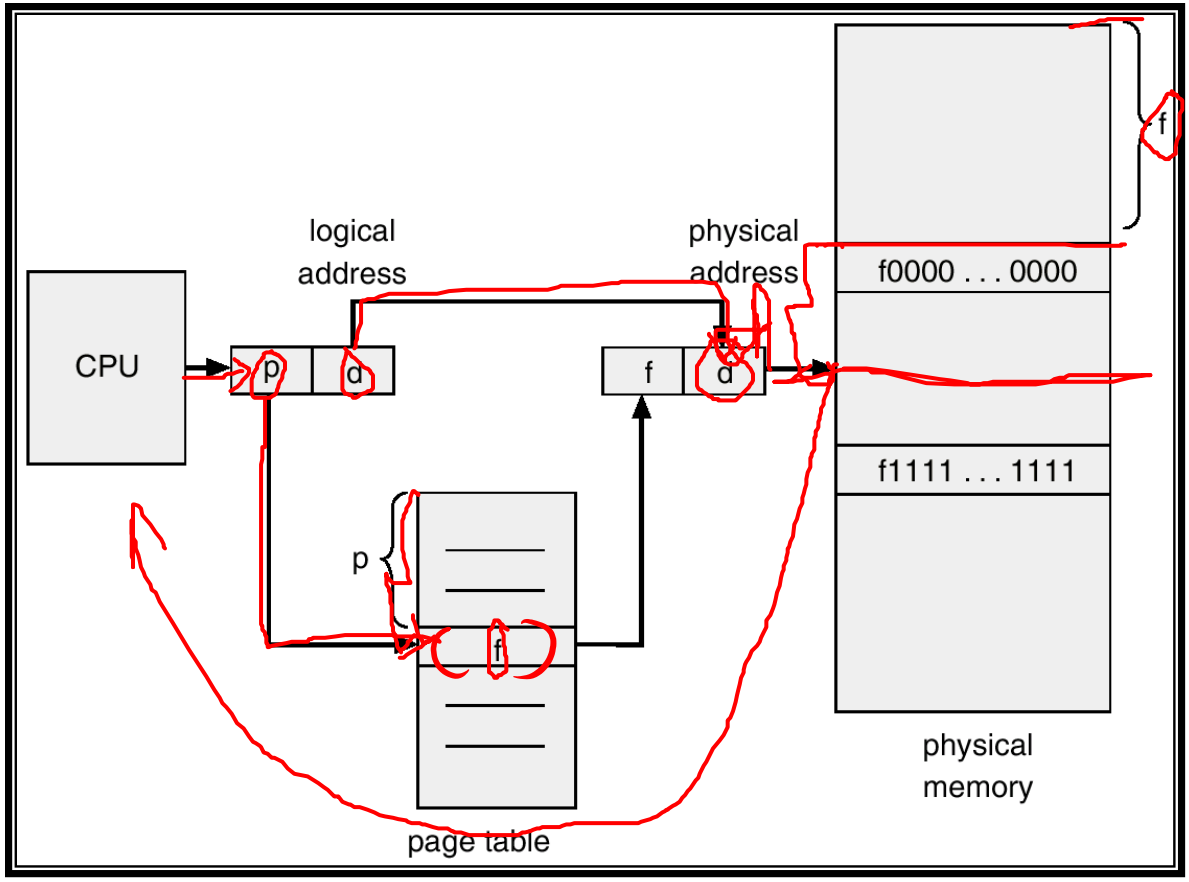


Address Translation Scheme

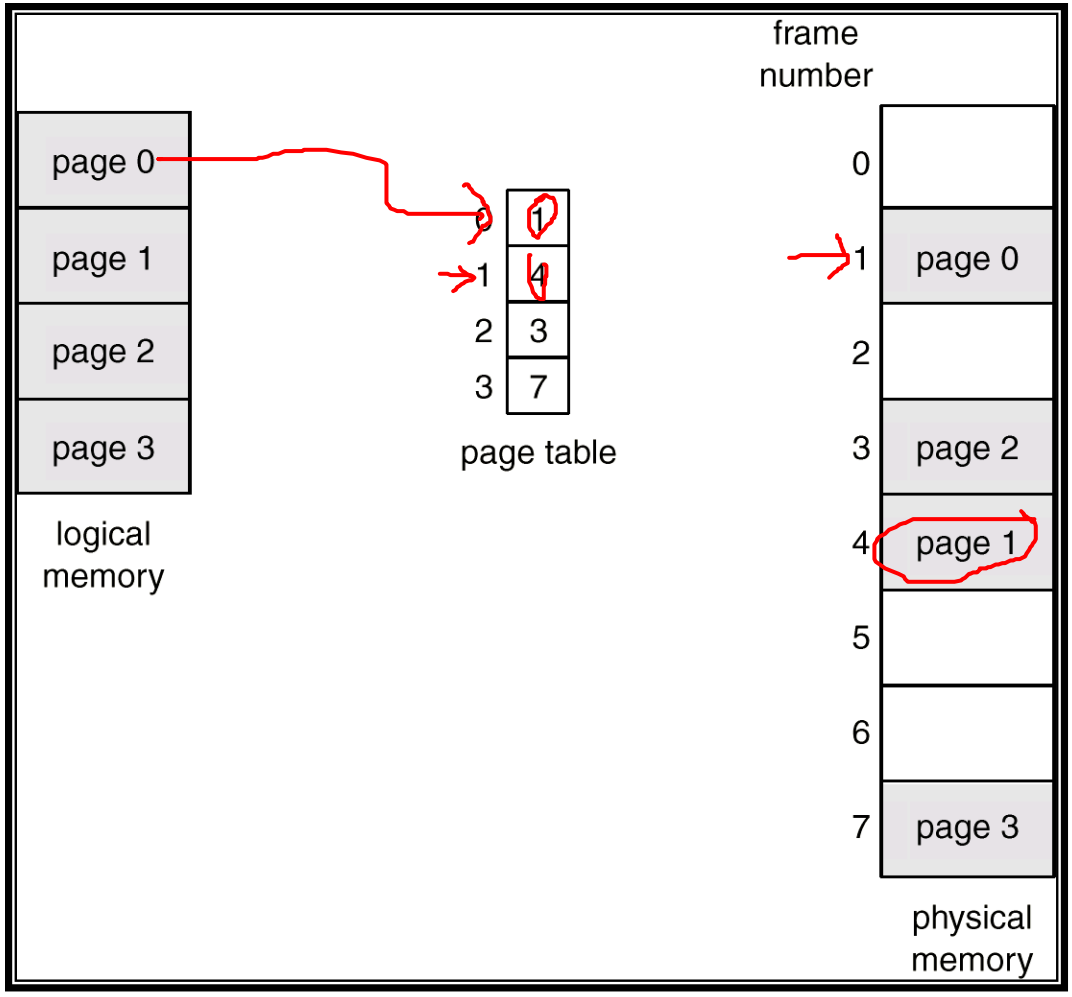
- Address generated by CPU is divided into:
  - *Page number (p)* – used as an index into a *page table*. Page table contains base address of each page in physical memory.
  - *Page offset (d)* – combined with base address to define the physical memory address that is sent to the memory unit.
  - For given logical address space 2^m and page size 2^n



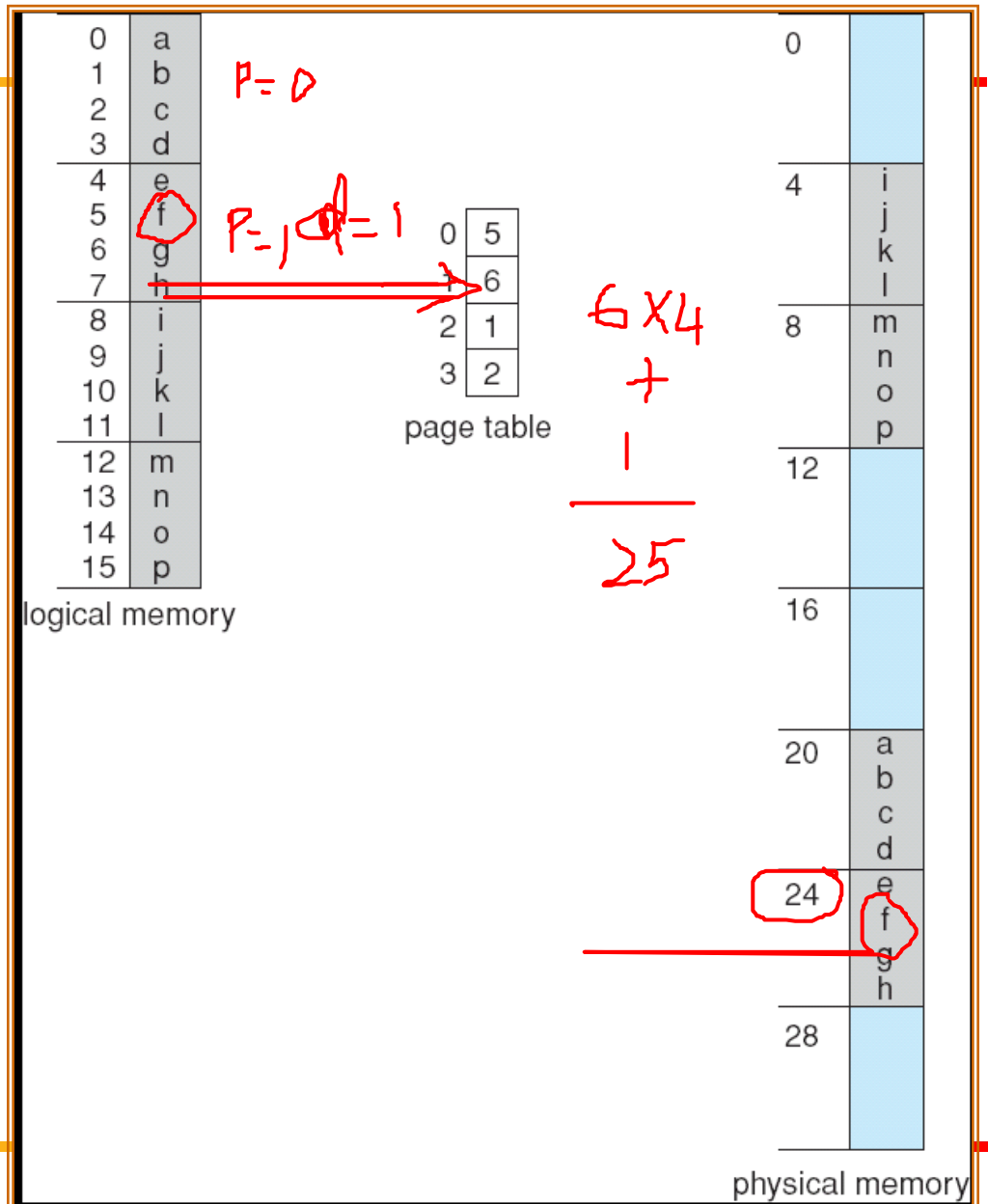
Address Translation Architecture



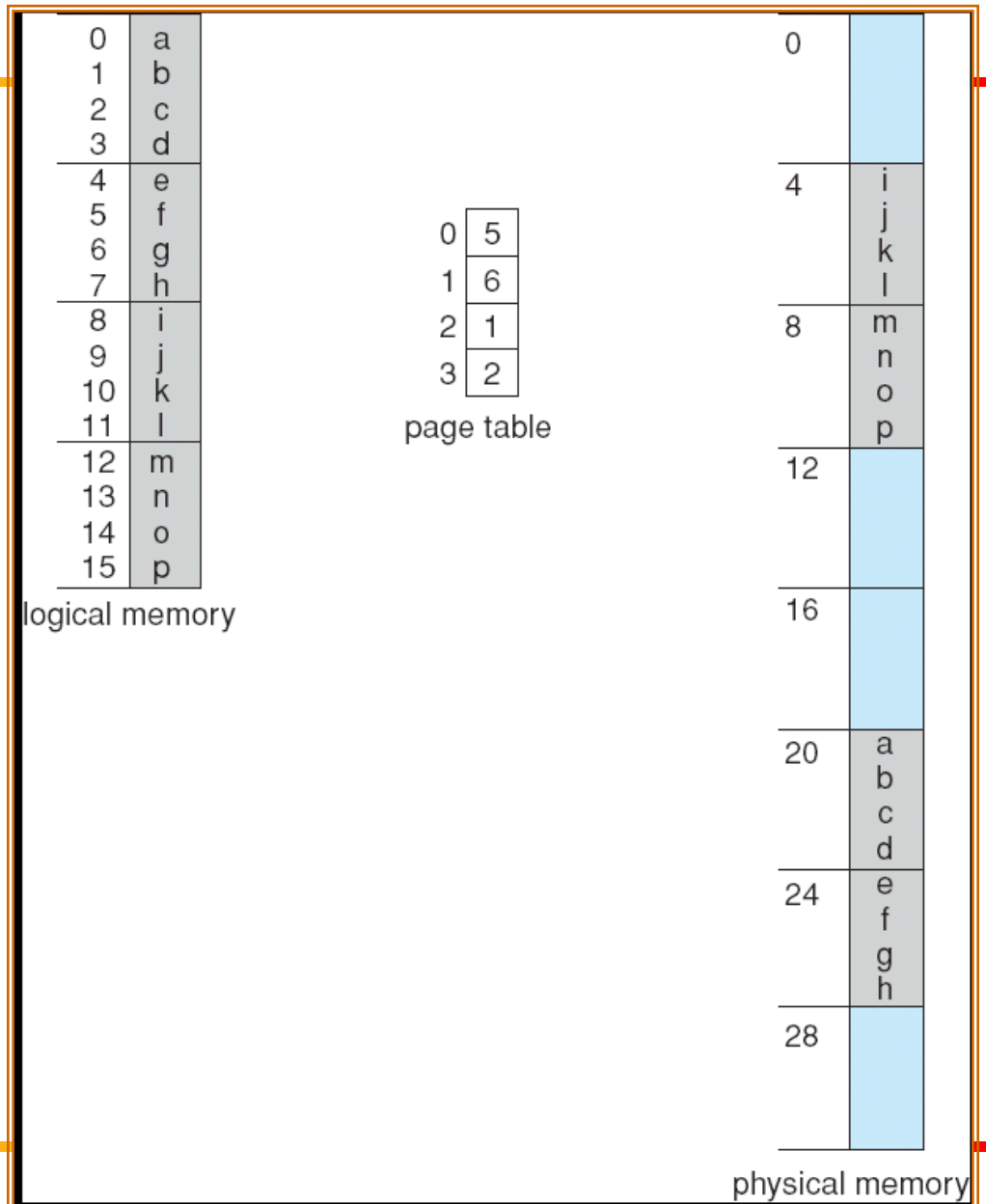
Paging Example



Paging Example



Paging Example



Important points....

- Paging is a form of dynamic relocation
- No external fragmentation but may have some internal fragmentation
- Small or large page size ?
- Page size determines
  - Memory wastage due to internal fragmentation
  - Size of the page table for a process
  - disk I/O data transfer
- small page : increases paging table overhead !!!!, internal fragmentation decreases
- large page : Decreases paging table overhead, internal fragmentation increases, disk I/O is more efficient when the number of data being transferred is larger

Contd…

- page size is usually 4 KB to 8 KB
- Needs to maintain the allocation details of main memory ➔ frame table

Problem

Consider a logical address space of 64 pages of 1,024 words each, mapped onto a physical memory of 32 frames.

a. How many bits are there in the logical address?

b. How many bits are there in the physical address?

$64 \times 1024 = 2^6 \times 2^{10} = 2^{16}$

~~$32 \times 1024 = 2^{15}$~~

innovate achieve lead

BITS Pilani

Pilani Campus

Virtual Memory

Introduction

- How to increase the degree of multiprogramming ?
- Motivation:
  - Programs often have code to handle unusual error conditions which is almost never executed.
  - Arrays, lists, and tables are often allocated more memory than they actually need.
  - Certain options and features of a program may be used rarely
  - Principle of locality
  - trashing is a condition where system spends more time in swapping than executing instructions
- Virtual memory is a technique that allows the execution of processes that are not completely in memory.

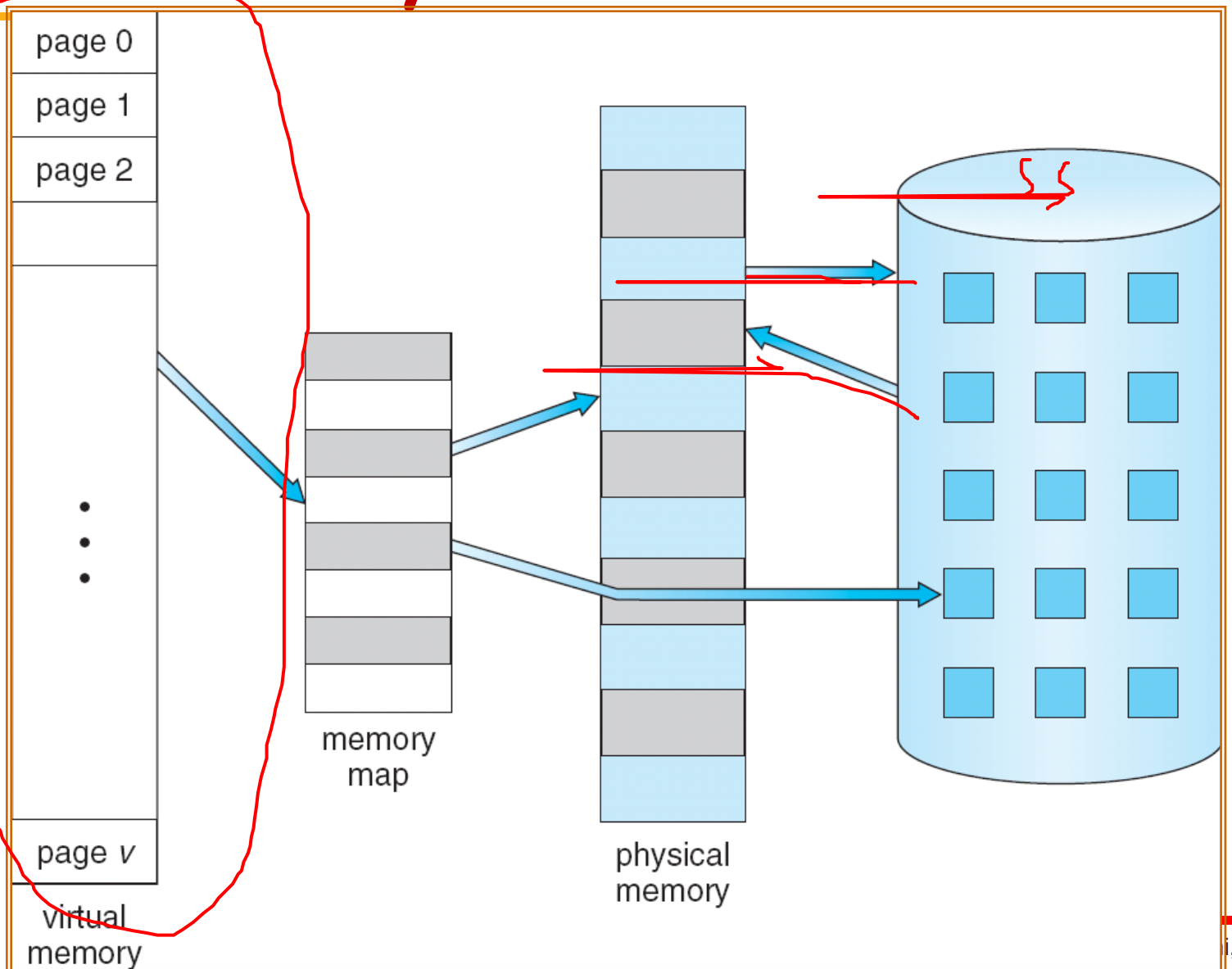
Contd…

- Advantages:
  - A program would no longer be constrained by the amount of physical memory that is available
  - Because each user program could take less physical memory, more programs could be run at the same time
  - increases the CPU utilization and throughput
  - Less I/O would be needed to load or swap each user program into memory, so each user program would run faster

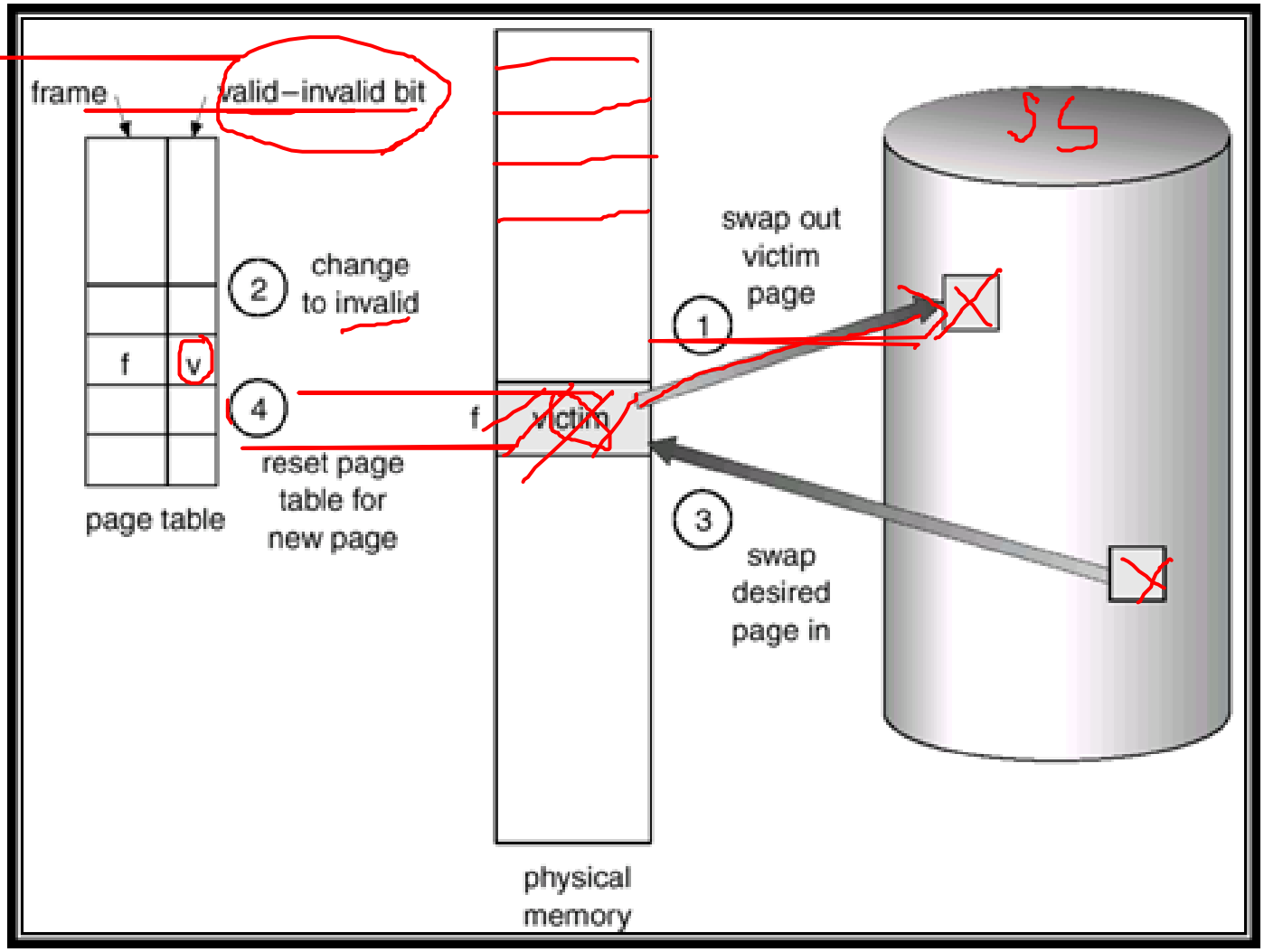
Background

- **Virtual memory** – separation of user logical memory from physical memory.
  - Only part of the program needs to be in memory for execution
  - Logical address space can therefore be much larger than physical address space
  - Allows physical address spaces to be shared by several processes
  - Allows for more efficient process creation

Virtual Memory That is Larger Than Physical Memory



What happens if there is no free frame?



Contd…

- If no frames are free, *two page transfers (one out and one in) are* required.
- Use **modify (dirty) bit** to reduce overhead of page transfers – only modified pages are written to disk
- Demand paging requires two algorithms:
  - frame – allocation algorithm
  - page – replacement algorithm
- Frame allocation algorithm : handles frame allocation to each process.
- page – replacement algorithm : deals with the frames that are to be replaced

Procedure

1. Find the location of the desired page on the disk.
2. Find a free frame:
   a) If there is a free frame, use it.
   b) If there is no free frame, use a page-replacement algorithm to select a victim frame.
   c) Write the victim frame to the disk; change the page and frame tables accordingly.
3. Read the desired page into the newly freed frame; change the page and frame tables.
4. Restart the user process.

Problem

Consider a logical address space of 64 pages of 1,024 words each, mapped onto a physical memory of 32 frames.

a. How many bits are there in the logical address?

b. How many bits are there in the physical address?

Problem

Assuming a 1-KB page size, what are the page numbers and offsets for the following address references

a. 3085(d)

b. 42095(H)

c. 215201(O)

d. 650000(H)

e. 2000001 (H)

Problem

Consider a computer system with a 32-bit logical address and 4-KB page size. The system supports up to 512 MB of physical memory. Find out #pages and #frames.

LM = 32 bits

P.S. = 12 b

20 bits

1M pages

PM. = 29 b

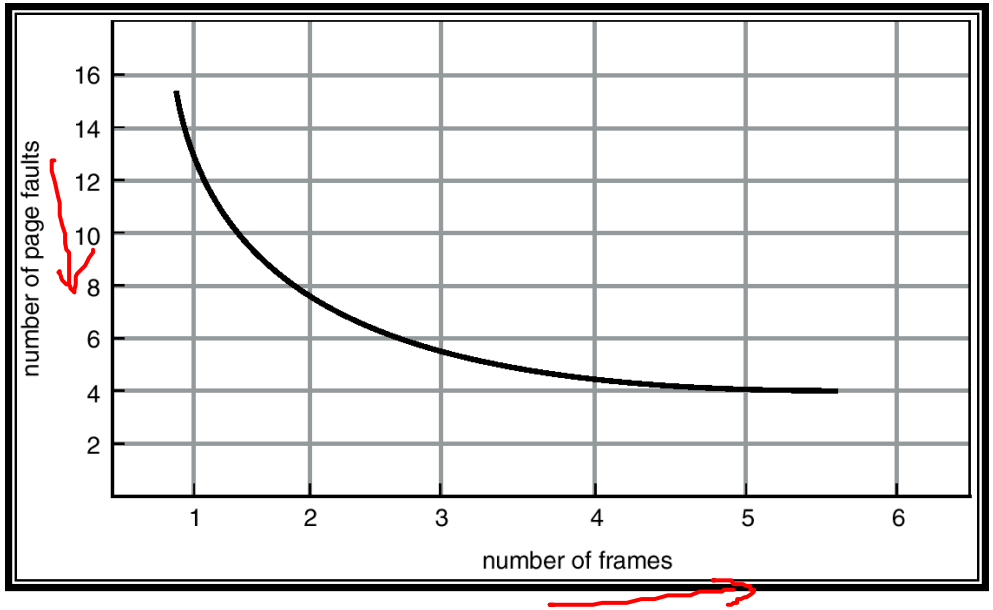
12

17 bits

2^{17} = 128K fr

Page Replacement Algorithms

- Want lowest page-fault rate.

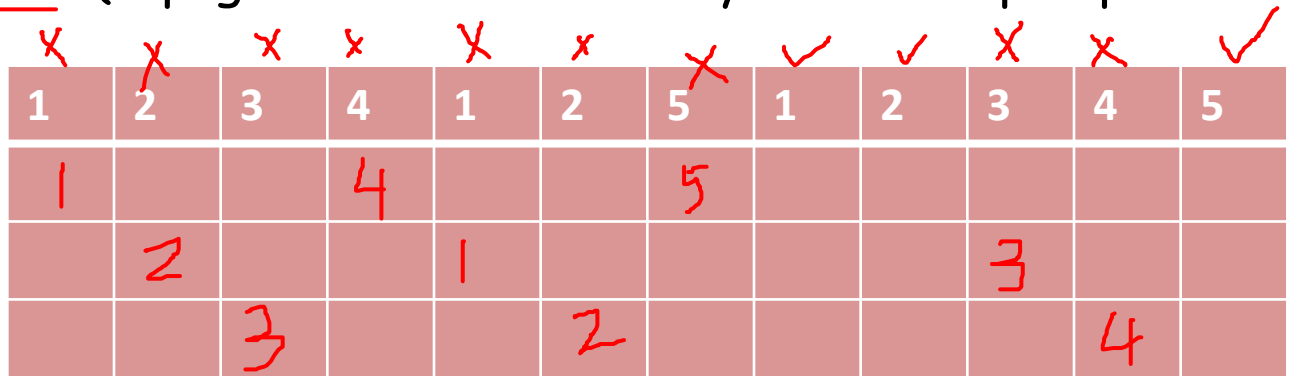


- Evaluate algorithm by running it on a particular string of memory references (reference string) and computing the number of page faults on that string.

First-In-First-Out (FIFO) Algorithm

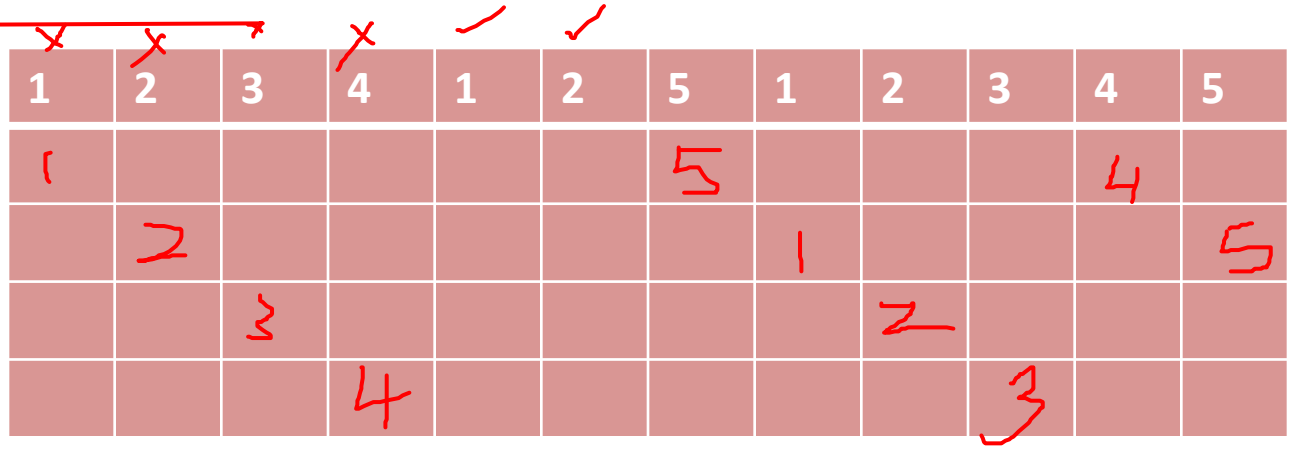
Reference string: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

3 frames (3 pages can be in memory at a time per process)



9/12

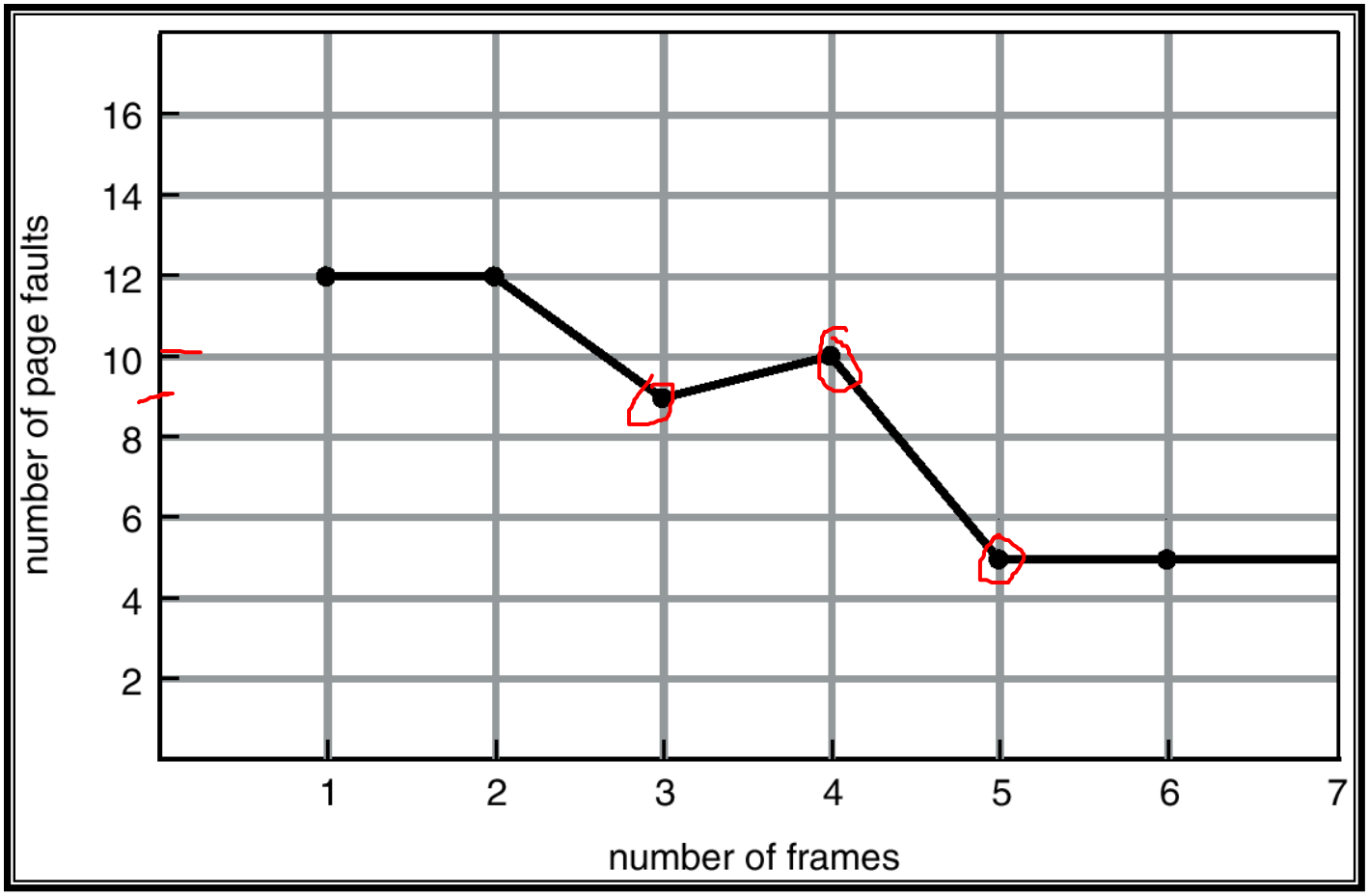
4 frames



10/12

FIFO Replacement - Belady's Anomaly ➔ more frames ⇒ more page faults

FIFO Illustrating Belady's Anamoly



Example 2

Consider the following sequence of address

0100, 0432, 0101,0612, 0102, 0103, 0104, 0101, 0611, 0102, 0103, 0104,0101,0610, 0102, 0103, 0104, 0101, 0609, 0102, 0105

Assume 100 byte page.

Find out the reference string.

1, 4, 1, 6, 1, 1

Replacement Algorithm

Given page reference string: 1,2,3,4,2,1,5,6,2,1,2,3,7,6,3

Compare the number of page faults for FIFO, Optimal and LRU page replacement algorithm

Solution : FIFO

11/15

	x	x	x	x	✓	✓	x	x	x	x	✓	x	x	x	✓
	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3
f_0	1						5					3			
f_1		2						6					7		
f_2			3						2					6	
f_3				4						1					

Problem3: Replacement Algorithm

Given page reference string: 1,2,3,4,2,1,5,6,2,1,2,3,7,6,3

Compare the number of page faults for FIFO, Optimal and LRU page replacement algorithm

7/15

Solution : Optimal ➔ **Replace page that will not be used for longest period of time**

	x	x	x	x	✓	✓	x	X	✓	✓	✓	✓	x	✓	✓
	1	2	3	4	2	1	5	6	2	1	2	3	7	6	3
	1												7		
		2													
			3												
				4	1		5	6							

Problem3: Replacement Algorithm

Given page reference string: 1,2,3,4,2,1,5,6,2,1,2,3,7,6,3

Compare the number of page faults for FIFO, Optimal and LRU page replacement algorithm

Solution : LRU

9/15

x	x	x	x	✓	✓	x	x	✓	✓	✓	x	x	x	✓
1	2	3	4	2	1	5	6	2	1	2	3	7	6	3
1													6	
	2													
		3				5	1				3			
			4				6					7		

Problem 1

Consider the following reference string: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

For Optimal and LRU replacement algorithm with number of frames = 2, 3, 4, check whether it exhibits Belady's Anomaly or not.

Belady's anomaly : more frames $\Rightarrow$ more page faults

Optimal

1	2	3	4	1	2	5	1	2	3	4	5

1	2	3	4	1	2	5	1	2	3	4	5

1	2	3	4	1	2	5	1	2	3	4	5

LRU

1	2	3	4	1	2	5	1	2	3	4	5

1	2	3	4	1	2	5	1	2	3	4	5

1	2	3	4	1	2	5	1	2	3	4	5

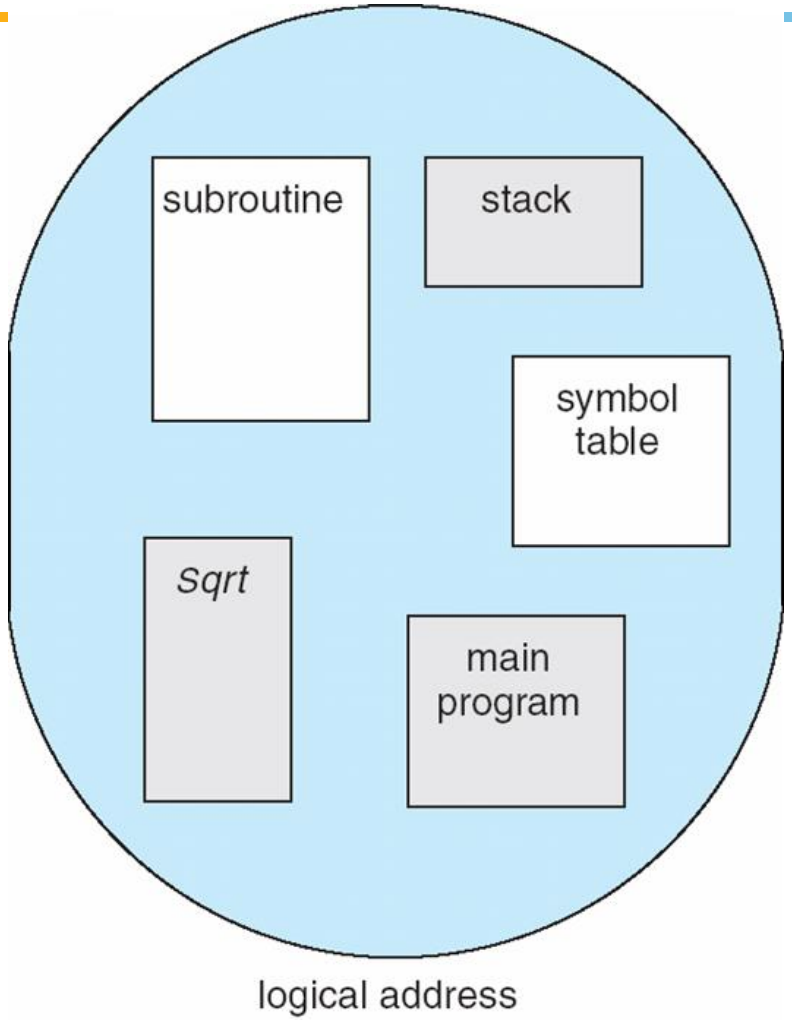
Segmentation

- Memory-management scheme that supports user view of memory

- A program is a collection of modules / segments
  - A segment is a logical unit such as:
    main program, procedure / function / method,
    object, local variables, global variables
    common block, stack
    symbol table, arrays

- Two Types : Simple Segmentation and Virtual memory segmentation

User's View of a Program



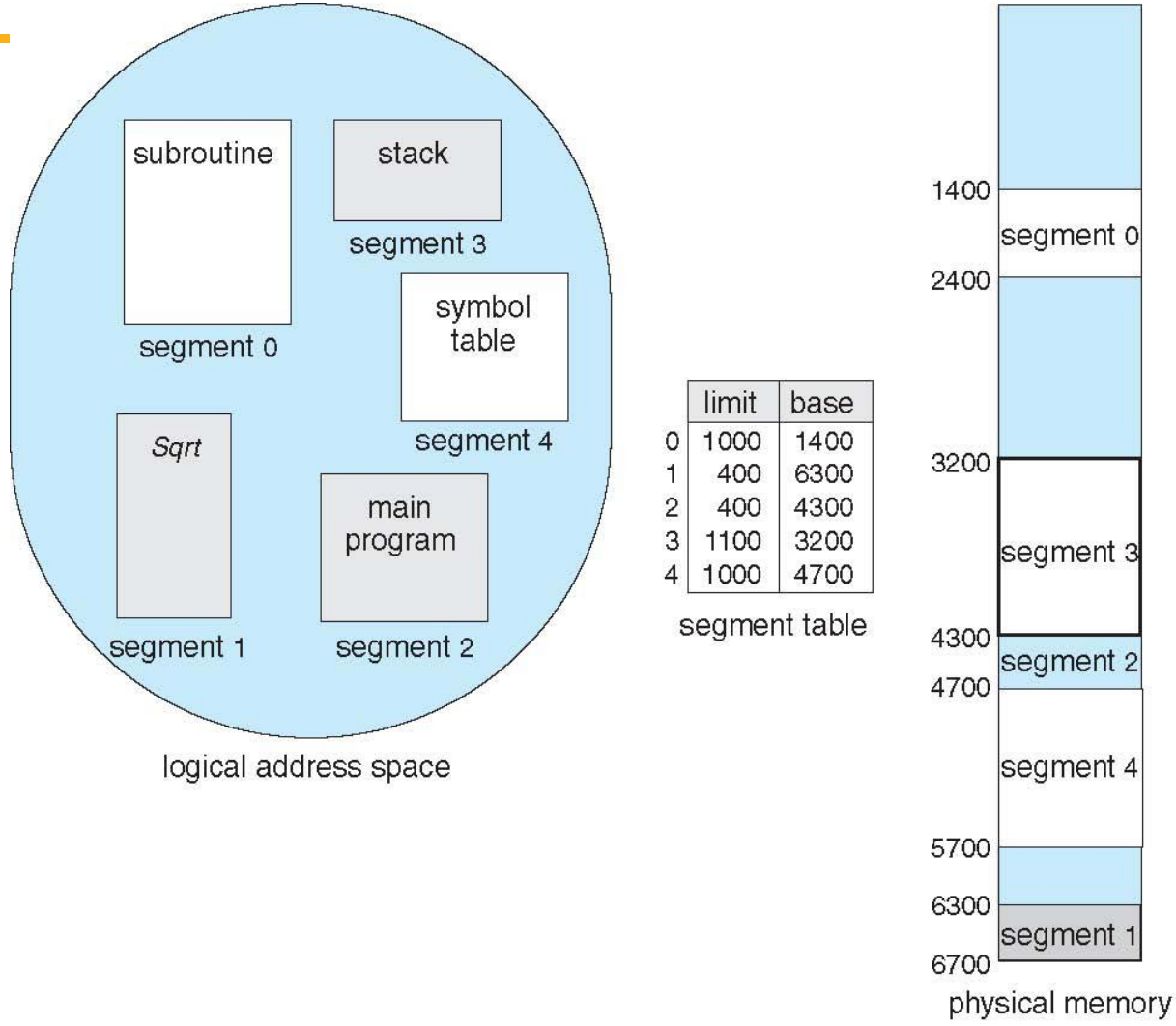
Logical address contains the tuple <segment#, offset>

Segmentation Architecture

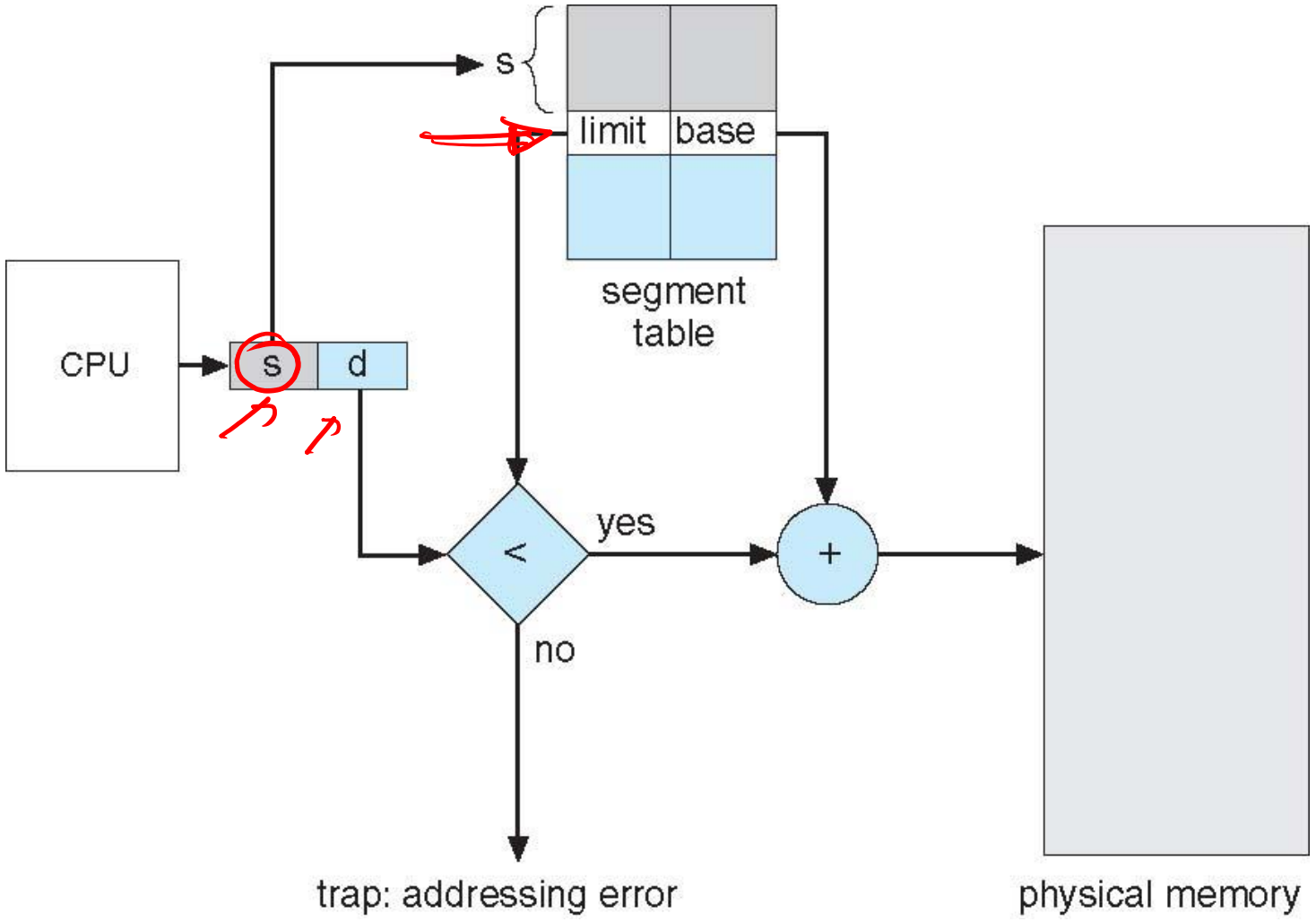
- **Segment table** – maps two-dimensional logical address to physical address;
- Each table entry has:
  - **base** – contains the starting physical address where the segments reside in memory
  - **limit** – specifies the length of the segment
- **Segment-table base register (STBR)** points to the 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**

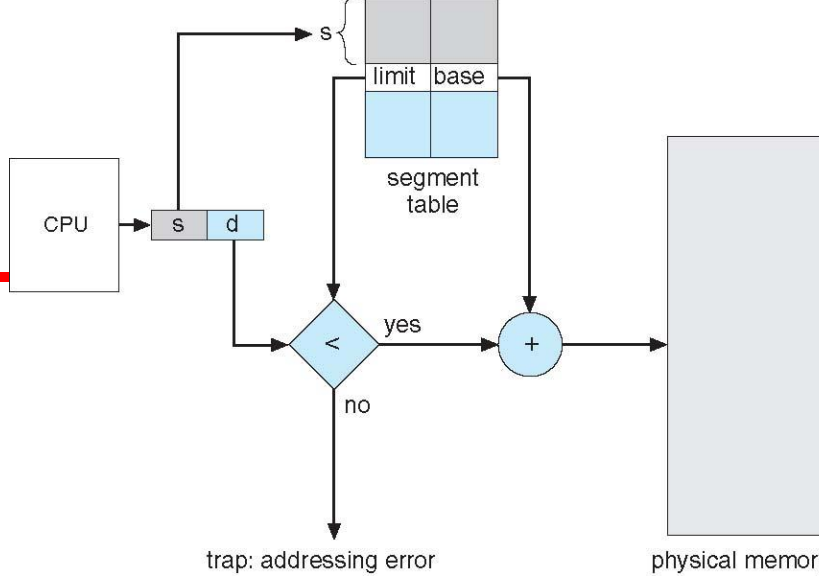
Example of Segmentation



Segmentation Hardware



Problem: Segmentation



Consider the following segment table:

Segment	Base	Length(Limit)
0	128	512
1	8192	2048
2	1024	4096
3	16384	8192
4	32768	1024
5	65536	16384

What are the physical addresses for the following logical addresses (s, d)?

a) 0, 430

b) 1, 2056

c) 2, 5024

d) 3, 7024