

A decorative vertical bar on the left side of the slide. It consists of a dark teal background with a white dotted vertical line running through its center. To the right of this bar, there are several orange circles of varying sizes, arranged in a cluster. The largest circle is at the top, with several smaller ones below and to its right. The entire slide is framed by thin orange vertical lines on the far left and far right.

PRINCIPLES OF OPERATING SYSTEMS

LECTURE 13:PAGING

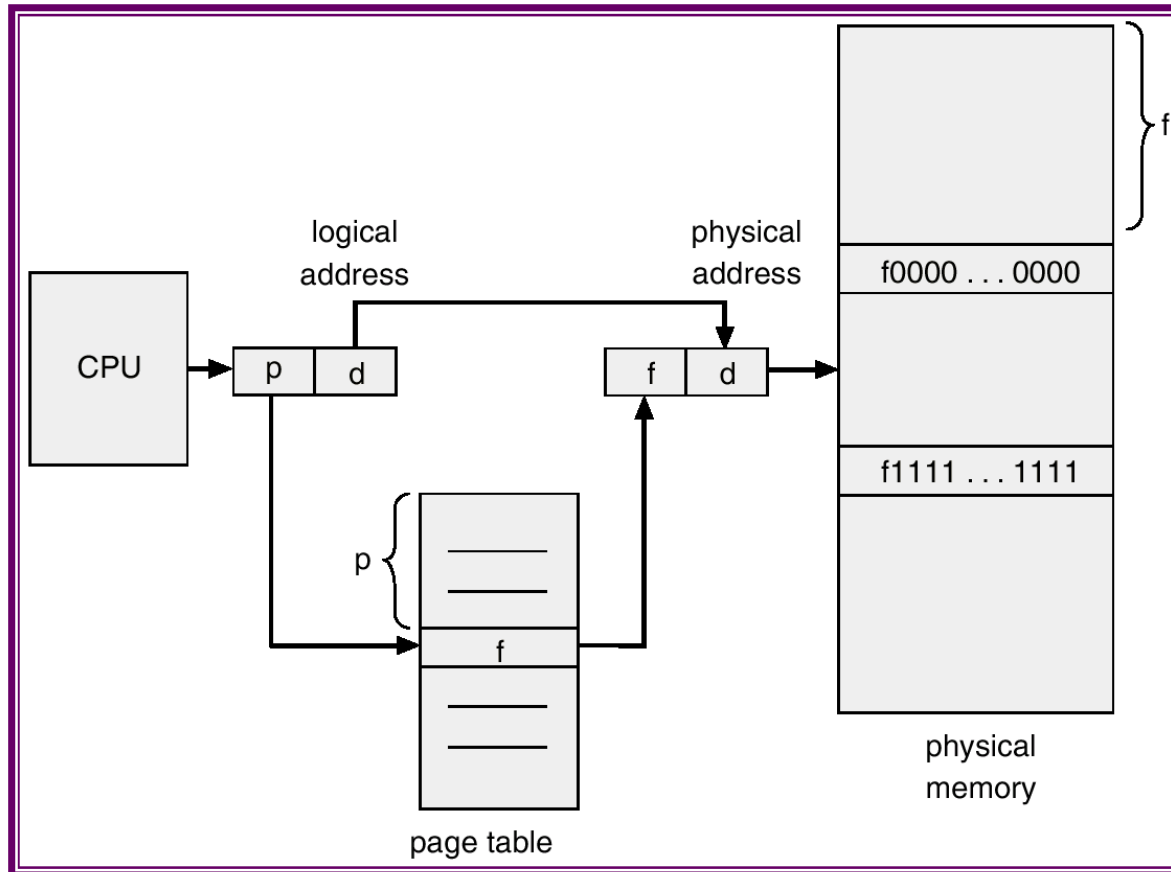
Paging

- 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.

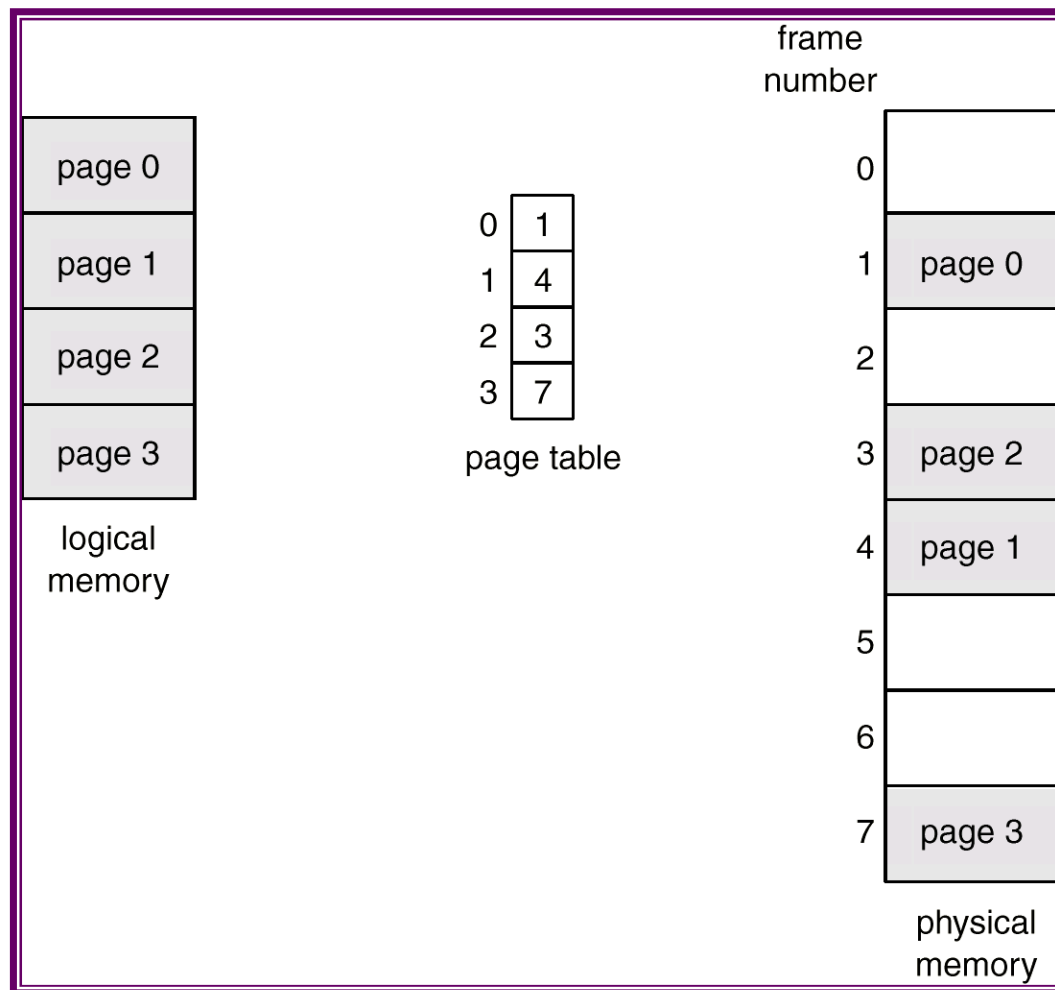
Address Translation Scheme

- 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.

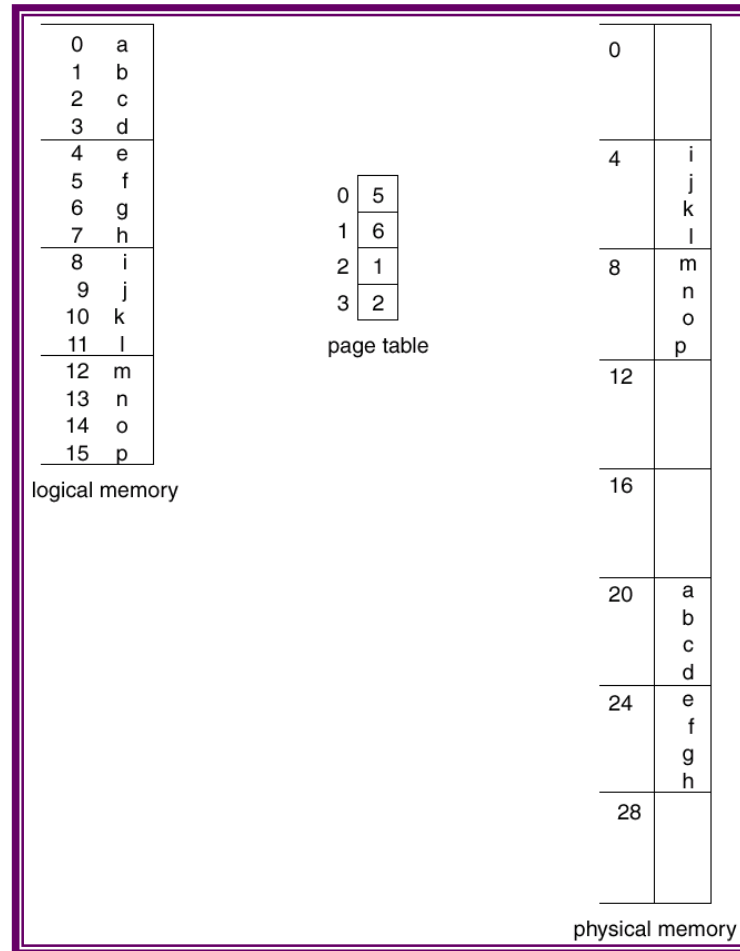
Address Translation Architecture



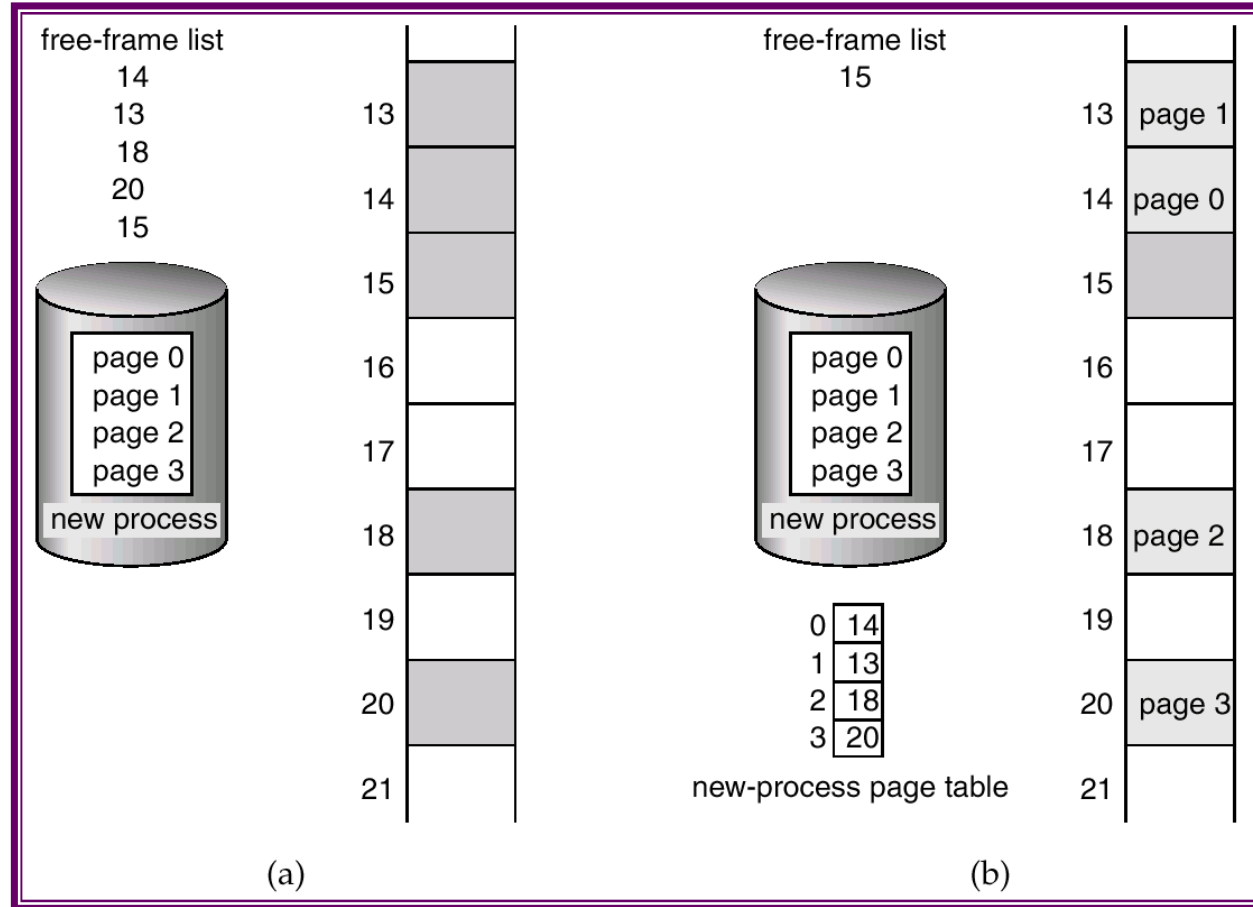
Paging Example



Paging Example



Free Frames



Before allocation

After allocation

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 memory* or *translation look-aside buffers (TLBs)*

Associative Memory

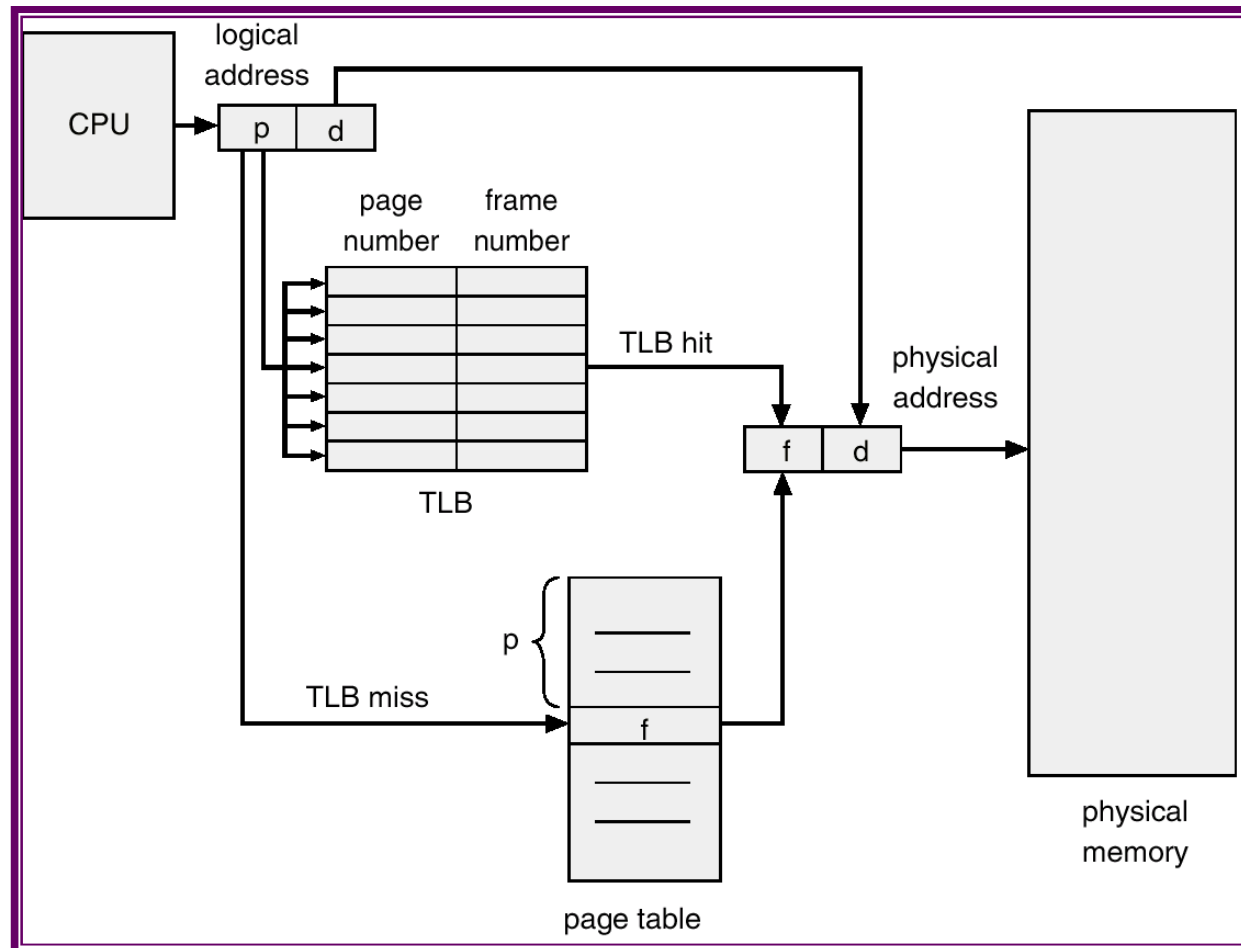
- Associative memory – parallel search

Page #	Frame #

Address translation (A' , A'')

- ☞ If A' is in associative register, get frame # out.
- ☞ Otherwise get frame # from page table in memory

Paging Hardware With TLB



Effective Access Time

- Associative Lookup = ε time unit
- Assume memory cycle time is 1 microsecond
- Hit ratio – percentage of times that a page number is found in the associative registers; rasion related to number of associative registers.
- Hit ratio = α
- Effective Access Time (EAT)

$$\begin{aligned} \text{EAT} &= (1 + \varepsilon) \alpha + (2 + \varepsilon)(1 - \alpha) \\ &= 2 + \varepsilon - \alpha \end{aligned}$$

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.

Valid (v) or Invalid (i) Bit In A Page Table

