# OPERATING SYSTEMS

## Memory Management

**Suresh Jamadagni**
Department of Computer Science

# OPERATING SYSTEMS

## Slides Credits for all the PPTs of this course

- The slides/diagrams in this course are an **adaptation**, **combination**, and **enhancement** of material from the following resources and persons:

1. Slides of Operating System Concepts, Abraham Silberschatz, Peter Baer Galvin, Greg Gagne - 9th edition 2013 and some slides from 10th edition 2018
2. Some conceptual text and diagram from Operating Systems - Internals and Design Principles, William Stallings, 9th edition 2018
3. Some presentation transcripts from A. Frank – P. Weisberg
4. Some conceptual text from Operating Systems: Three Easy Pieces, Remzi Arpaci-Dusseau, Andrea Arpaci Dusseau

# OPERATING SYSTEMS

## Paging

**Suresh Jamadagni**
Department of Computer Science

## Paging

- Segmentation permits Physical address space of a process can be **noncontiguous**
- Paging is another memory-management scheme that offers this advantage.
  - **Avoids external fragmentation** and need for compaction
  - Solves the problem of fitting memory chunks of varying sizes onto the backing store.
- The backing store also has the fragmentation problems

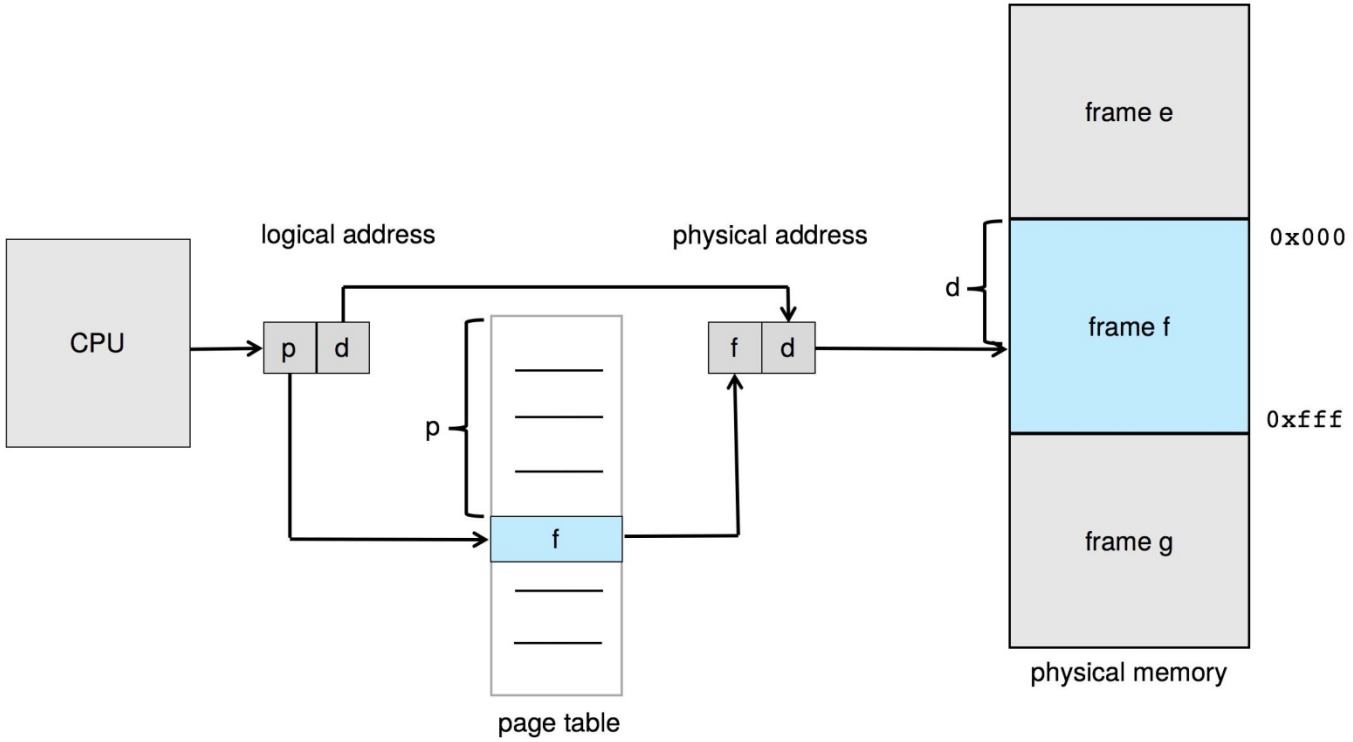
## Paging: Basic Method


- Divide physical memory into fixed-sized blocks called **frames**
  - Size is power of 2, between 512 bytes and 16 Mbytes
- Divide logical memory into blocks of same size called **pages**
- **Page size and frame size are defined by the hardware**
- 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
- Backing store likewise split into pages
- Still have Internal fragmentation
- Following command displays the page size supported in the system,

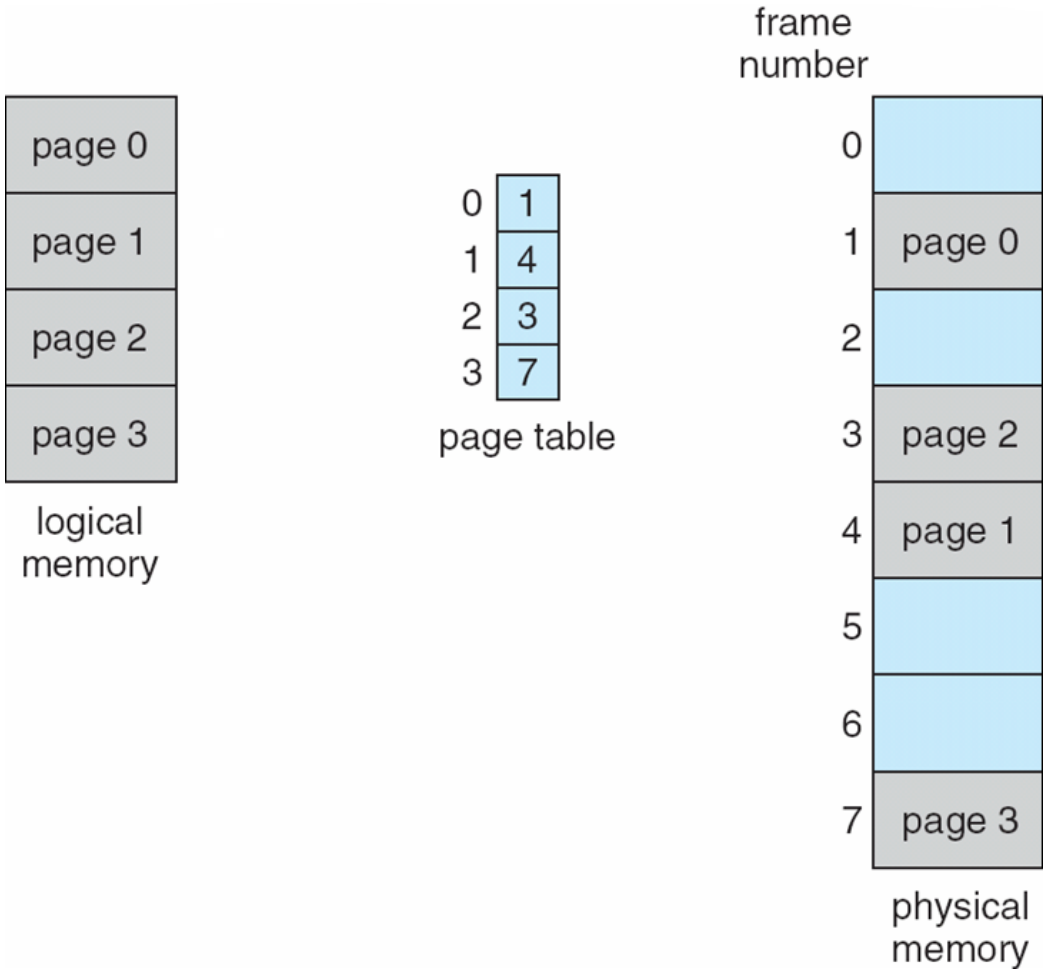
```
getconf   PAGESIZE
```

# OPERATING SYSTEMS

## Paging Hardware



# Paging Model of Logical and Physical Memory



# OPERATING SYSTEMS

## Address Translation Scheme

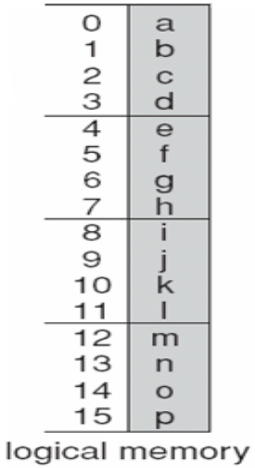
- Address generated by CPU is divided into:
  - **Page number** (***p***) – used as an index into a **page table** which contains base address of each page in physical memory
  - **Page offset** (***d***) – combined with base address to define the physical memory address that is sent to the memory unit

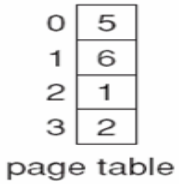

  - Page size defined by hardware.
  - Page size varies between 512 bytes and 1 GB per page
  - The logical address space is $2^m$ and page size $2^n$
  - The high-order m − n bits of a logical address designate the page number, and the n low-order bits designate the page offset

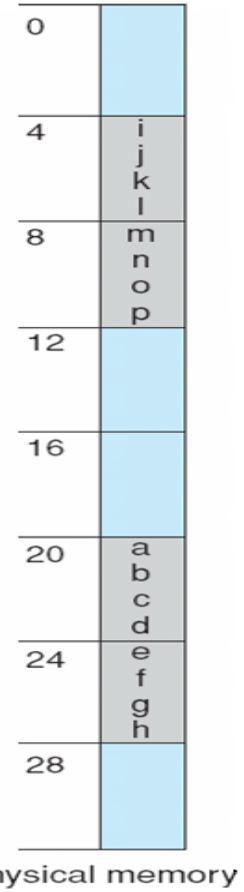
# OPERATING SYSTEMS

## Paging Example

- Logical address: n = 2 and m = 4. Using a page size of 4 bytes and a physical memory of 32 bytes (8 pages)
- Logical address space = 2^m
- Page size = 2^n bytes




- No external fragmentation, we may have internal fragmentation

# OPERATING SYSTEMS

## Paging (Cont.)

- Calculating internal fragmentation
  - Page size = 2,048 bytes
  - Process size = 72,766 bytes
  - 35 pages + 1,086 bytes
  - **Internal fragmentation of 2,048 - 1,086 = 962 bytes**
  - Worst case fragmentation = **1 frame – 1 byte**
  - On average fragmentation = 1 / 2 frame size
  - So small frame sizes desirable?
  - But each page table entry takes memory to track
  - Page sizes growing over time
- Programmer's  view and physical memory now very different
- By implementation the user process can only access its own memory

## Paging (Cont.)

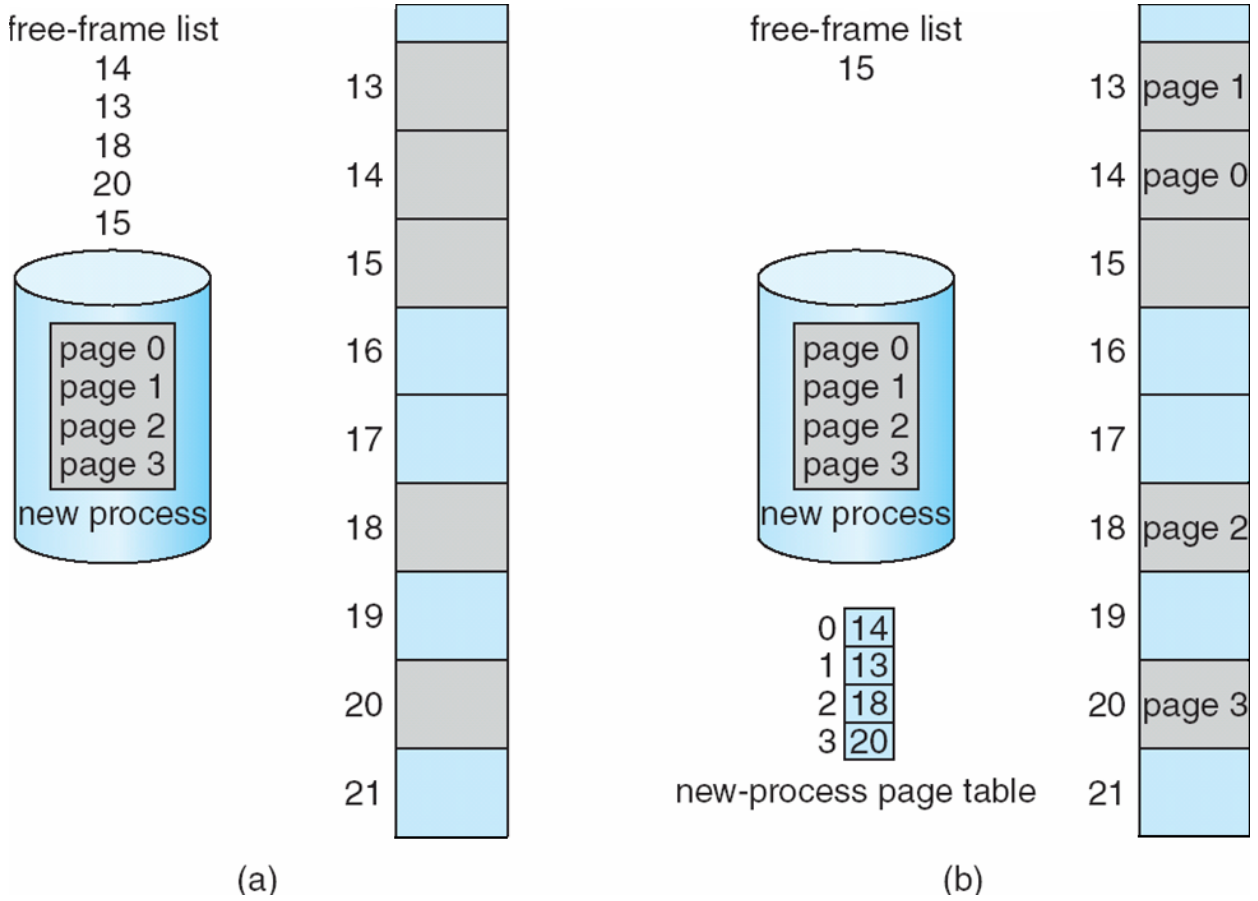
- Worst case, a process would need *n* pages plus 1 byte.
  - needs to be allocated *n* + 1 frames, resulting in internal fragmentation of almost an entire frame.
- If the page size is smaller, an overhead is involved with each page table entry
- The overhead can be reduced by increasing the page size
- Page size are 4KB, 8KB
- Some CPU and kernel support multiple page sizes.
- Researchers are now developing support for variable on-the-fly page size.

## Paging (Cont.)

- On a 32-bit CPU, each page-table entry is 4 bytes long, but that size can vary as well.
- A 32-bit entry can point to one of $2^{32}$ physical page frames.
- If frame size is 4 KB ($2^{12}$), then a system with 4-byte entries can address $2^{44}$ bytes (or 16 TB) of physical memory.
- If other information is kept in the page-table entries, it reduces number of  bits available to address page frames.
- A system with 32-bit page-table entries may address less physical memory than the possible maximum.

# OPERATING SYSTEMS

## Free Frames




Before allocation

After allocation

## Paging (Cont.)

- OS manages physical memory, keeps track of which frames are free or allocated using **frame table**.
- OS must be aware that user processes operate in user space, and all logical addresses must be mapped to produce physical addresses.
- **The operating system maintains a copy of the page table for each process**.
- It is used to translate logical addresses to physical addresses
- It is also used by the CPU dispatcher to define the hardware page table when a process is to be allocated the CPU.
- Paging therefore increases the context-switch time.
- How to reduce the context-switch time?
  - Using TLB


# THANK YOU

**Suresh Jamadagni**
Department of Computer Science Engineering

**sureshjamadagni@pes.edu**