

A decorative graphic consisting of five circles arranged in two rows. The top row has three circles: a white circle with a purple outline on the left, a solid purple circle in the middle, and a solid purple circle on the right. The bottom row has two solid purple circles on the left and a white circle with a purple outline on the right.

# CS 162 Midterm Review

*Prepared especially for you by your smilin' TAs:*

Steve Martin  
Pete Broadwell  
Ben Huang

Note: the illustrations in this presentation are by Silberschatz, Galvin and Gagne ©2002

A decorative graphic consisting of five circles arranged in a single row. From left to right: a solid purple circle, a white circle with a purple outline, a solid purple circle, a white circle with a purple outline, and a solid purple circle.

## What is an operating system?

- Batch systems
- Uniprogramming
- Multiprogramming

## Function of an operating system

- Coordinator
  - Concurrency
  - I/O
  - Memory
  - Files
  - Networks
- Extended Machine
  - Present nice interface to messy hardware

## Operating system structure

- What are the components of an OS?
- What kinds of services does the OS provide?
- What are the design goals for the OS?
- System structure
  - “Simple” structure
  - Layered approach
  - Microkernel

## Processes

- What are they?
- How are they made?
- How are they represented in an OS?
  - E.g. what state do they have?
- How many processes can run on a CPU at once?
- What are process states?

## Threads

- What are they?
- How are they made?
  - How are they joined?
- What state do they maintain?
- How many can run on a CPU at once?
- How do they differ from processes?
- Why separate threads from processes?

## Threads, Cont. . .

- What is different between a thread and a process context switch?
- What does the state diagram for a thread look like?
- How does a thread go from inactive to running?
  - What needs to happen?
- Can we use both threads and processes?
  - Why would we do this?

## Cooperating threads

- What is the difference between multiprocessing and multiprogramming?
- What is the definition of 'run concurrently'?
- What are independent threads?
- What are cooperating threads?



## Exceptions

- What is an interrupt?
  - What happens when one occurs?
- What is a trap/exception?
  - What happens when one occurs?



## Atomicity

- What is an atomic operation?
- Why are they important?
- How can you ensure an instruction is atomic?
  - How about with multiple CPUs?

## Synchronization

- What is synchronization?
- What is a critical section?
- What is mutual exclusion?
- What is busy-waiting? (or 'spinning')
- What is a lock?
  - What are its properties?

## Synchronization, cont. . .

- What is a semaphore?
  - What are its properties?
- What is a condition variable?
  - What are its properties?
  - How are these different from semaphores?
- What is a monitor?
  - What is a Mesa-style monitor?
  - What is a Hoare-style monitor?

## Synchronization, cont. . .

- What is a mutex?
- Can you build one kind of synchronization primitive out of another?
- What is important about test-and-set and swap?
- What language support exists for synchronization?

## Deadlock


- What is deadlock?
  - What are the four conditions of deadlock?
- What are examples of preemptible and non-preemptible resources?
- What are the two approaches to the deadlock problem?
- What are ways to prevent deadlock?
  - What is a 'safe state'?

## Deadlock, cont. . .

- Bankers Algorithm
- Graph Algorithm
  - What is a resource allocation graph?

## Scheduler

- What is a dispatcher/scheduler?
  - What are its responsibilities?
- When does it run?
- How does it maintain control of the CPU?
  - How does it put a process on the CPU?



## Scheduling

- What are scheduling goals?
  - What is flow time?
  - What is throughput?
  - What is response time?
  - Why is fairness important?
    - When is it not important?



## Scheduling, cont. . .

- How are processes scheduled on the CPU?
  - FIFO (First in first out)
  - RR (Round robin)
  - SJF (Shortest job first)
  - SRTF (Shortest remaining time first)
  - EQ, or MLFQ (Multi-level feedback queues)

## Scheduling, cont. . .

- How do job characteristics influence scheduling choice?
- What is the difference between a preemptive and non-preemptive scheduling policy?
- What is an adaptive scheduling policy?

## Protection

- Multiprogramming without protection
  - How do linkers and loaders work?
  - What are the components of a program in memory?
- What are hardware protection modes?
  - Why do we need these?
- How can a user program access OS services?
  - How does the protection mode switch?

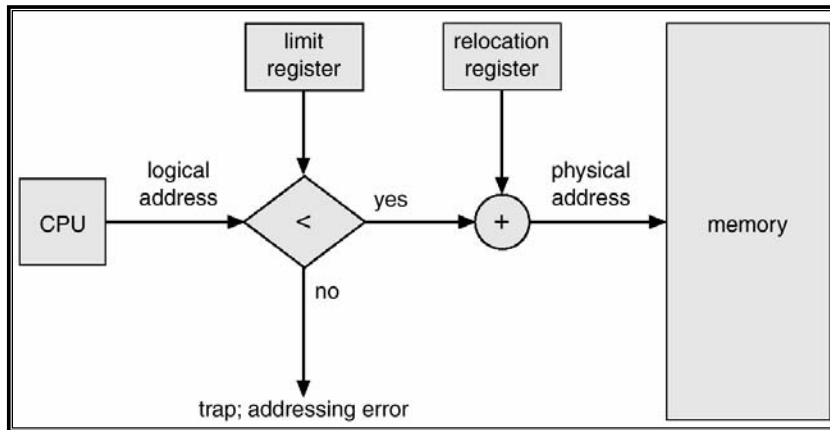
## Address Translation

- Why do we need this?
- What is a Virtual Address? What is a Physical Address?
- What is internal fragmentation? What is external fragmentation?

## Address Translation, cont. . .

- What is base and bound relocation?
  - What are the advantages
  - What are the disadvantages?
  - What does base-and-bounds relocation add to context switches?
  - What does it add to Memory lookups?

## Address Translation, cont. . .

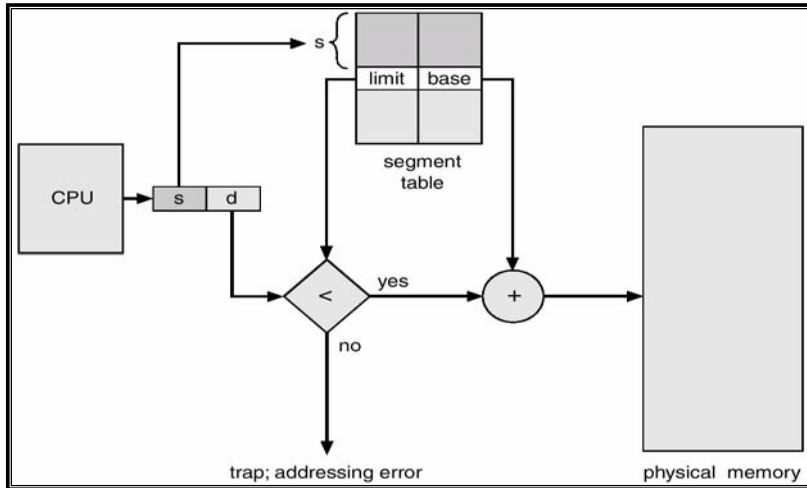


## Address Translation, cont. . .

### ● Segmentation

- What is a segment table?
- What does segmenting fix that is broken with base and bounds?
- What is a segment table entry? What does it look like?
  - What is each field for?
- What is a segment fault?
- What exactly happens when a program tries to read from memory?
- What does segmentation add to context switches?

## Address Translation, cont. . .

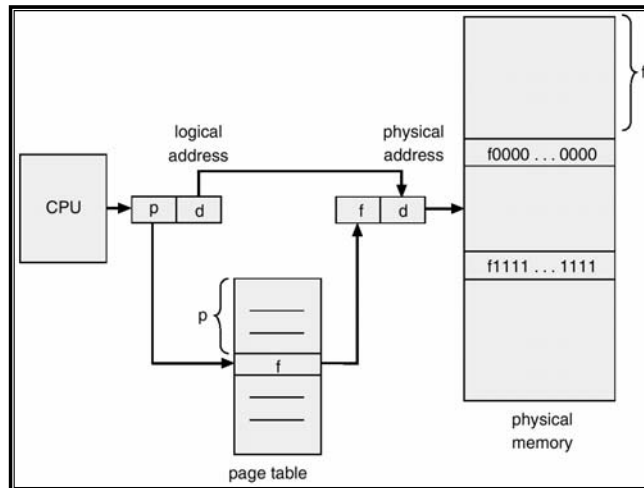


## Address Translation, cont. . .

### ● Paging

- What is a page table?
- What does paging fix that is broken in segmenting?
- What does a page table entry look like?
  - What is each field for?
- What is a page fault?
- What exactly happens when a program tries to read from memory?

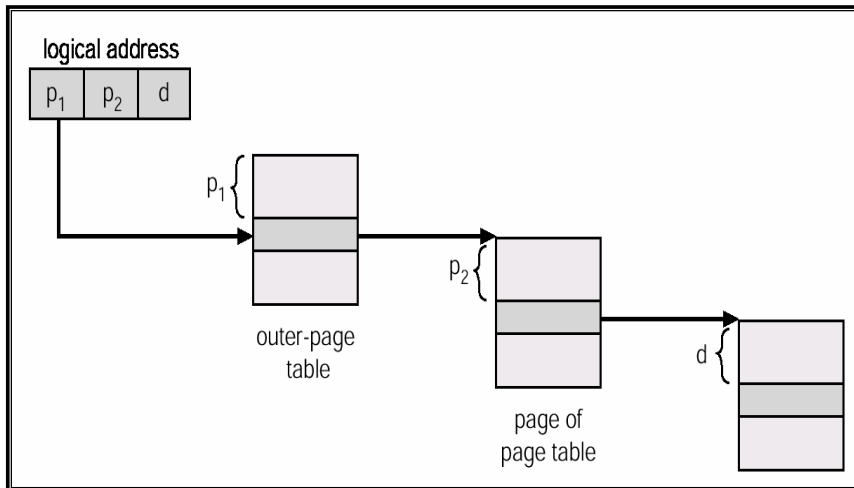
## Address Translation, cont. . .



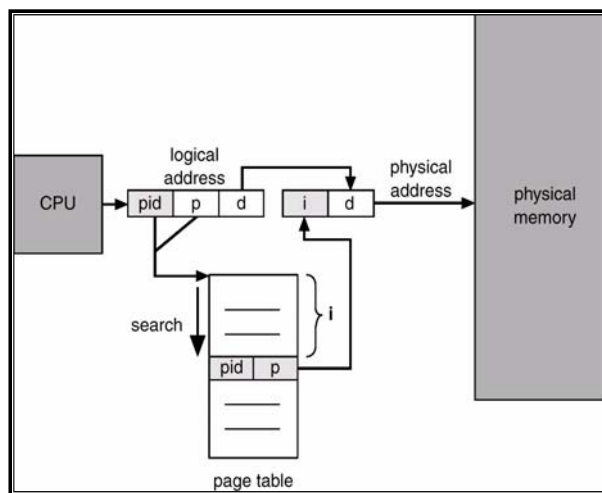
## Address Translation, cont. . .

- What does paging add to context switches?
- Can you page out anything in memory?
- How big is a page table?
- What is 2-level paging?
  - How does this work?
- What is an inverted page table?
  - How does this work?
  - Why do you need one?

## Address Translation, cont. . .



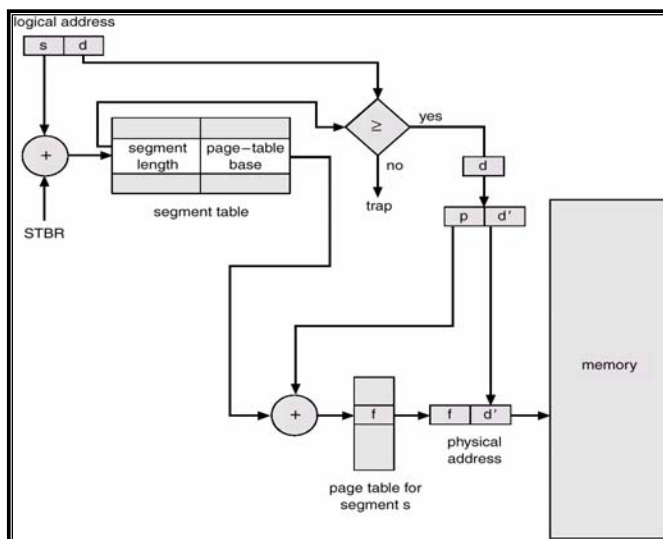
## Address Translation, cont. . .



## Address Translation, cont. . .

- How do these methods provide protection among processes?
- How would you combine segmentation with paging?
  - Why would you want to?

## Address Translation, cont. . .



## Caches and TLBs

- What is the primary motivation behind using a cache?
- What are the different methods for searching a cache?
  - What is direct mapped?
  - What is set associative?
  - What is fully associative?
  - What are the advantages and disadvantages of each?

## Caches and TLBs, cont. . .

- What is a TLB?
  - Why do we use them?
- What happens to the TLB on a context switch?
- What does using a TLB add to handling page faults?
- What is the effective access time of a TLB?
  - E.g. how is it calculated?

