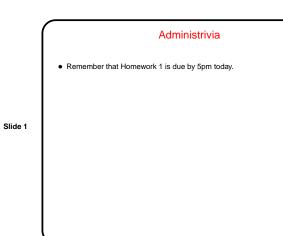
CSCI 4320

Slide 3



Minute Essay From Last Lecture

- Question: In a system with 8 CPUs and 100 processes, what's the maximum number of processes that can be running? ready? blocked?
- Answer:

Slide 2

- 8 processes can be running (assuming there are 8 that are runnable).
- 92 processes can be ready (if there are more, some will be running).
- 100 processes can be blocked.

Recap — Processes

- Process abstraction "program running on virtual CPU" (virtual program counter, virtual registers, etc.).
- Apparent concurrency (in almost all respects identical to real concurrency) provided by interleaving / context switches.
- Context switch switch between virtual CPUs, triggered by interrupts (I/O, error, system call, timer).
- Process can also be a way of grouping together other resources needed by a running program, e.g., "address space", list of open files.
 - These resources may form part of the "context" that must be saved / restored on a context switch.

Recap — Process States

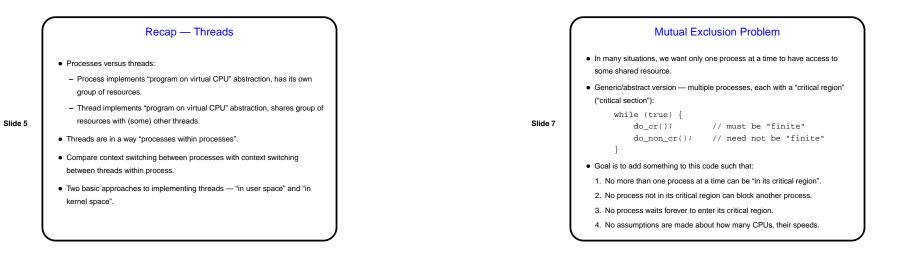
- Three basic states for processes running, ready, blocked.
- Some transitions are obvious, others require decision-making (ready to running); for now, assume existence of "scheduler" to make decisions.

Slide 4

CSCI 4320

Slide 8

September 16, 2003



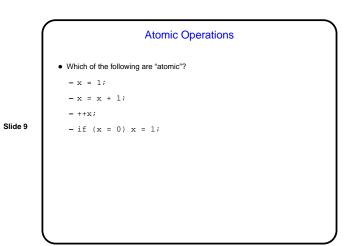


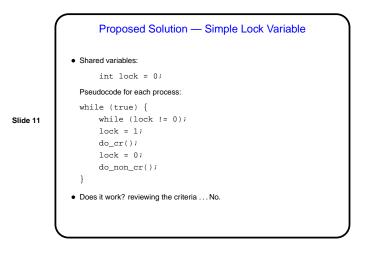
- Processes almost always need to interact with other processes:
 - "Ordering constraints" e.g., process B uses as input some data produced by process A.
 - Use of shared resources files, shared memory locations, etc.
- Slide 6
 - Use of shared resources can lead to "race conditions" output depends on details of interleaving.
 - Processes must communicate to avoid race conditions and otherwise synchronize.

	Mutual Exclusion Problem, Continued
• We'll l	pok at various solutions (some correct, some not):
	ng only hardware features always present (some notion of shared iable).
– Us	ng optional hardware features.
	ng "synchronization primitives" (abstractions that help solve this and er problems).
 Recall 	that a correct solution
– Mu	st work for more than 1 CPU.
– Mu	st work even in the face of unpredictable context switches — whatever
we	re doing, another process can pull the rug out from under us between
"ate	omic operations" (machine instructions).

CSCI 4320

September 16, 2003



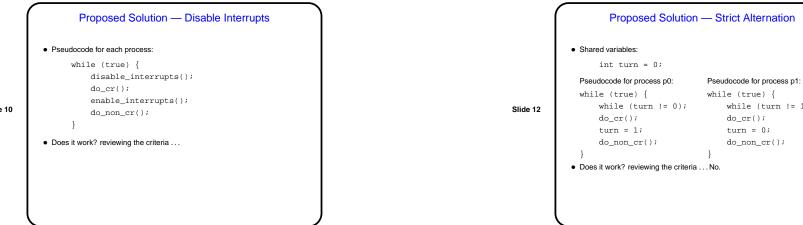


while (turn != 1);

do_cr();

turn = 0;

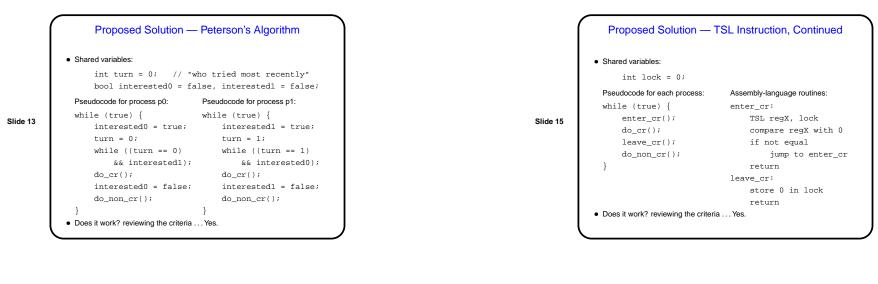
do_non_cr();

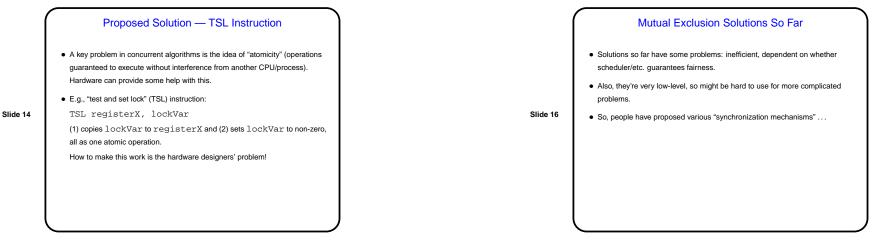


Slide 10

CSCI 4320

September 16, 2003





September 16, 2003

Minute Essay

- Do you see why the various solutions to the mutual exclusion problem so far work / don't work?
- Give an example (other than those discussed) of a situation in which you think a solution to this problem would be needed.

Slide 17