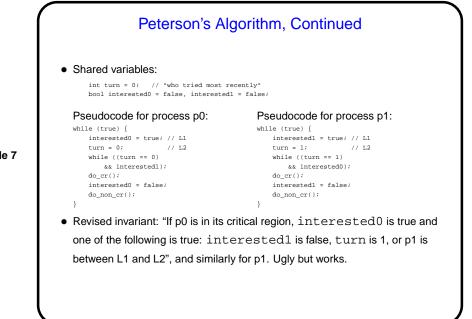


Peterson's Algorithm, Continued
Intuitive idea — p0 can only start do_cr() if either p1 isn't interested, or p1 is interested but it's p0's turn; turn "breaks ties".
Semi-formal proof using invariants is a bit tricky. Proposed invariant: "If p0 is in its critical region, interested0 is true and either interested1 is false or turn is 1"; similarly for p1.
If we can show this is an invariant, first requirement is met. Others are too.
But a fiddly detail — the invariant can be false if p0 is in its critical region when p1 executes the lines interested1 = true; turn = 1;. See next slide for revision.



Peterson's Algorithm, Continued
Requires essentially no hardware support (aside from "no two simultaneous writes to memory location X" — fairly safe assumption as long as X is a single "word"). Can be extended to more than two processes.
But complicated and not very efficient.

Slide 7

