Lecture 12.2
Mutual Exclusion

EN 600.320/420

Instructor: Randal Burns

13 March 2017

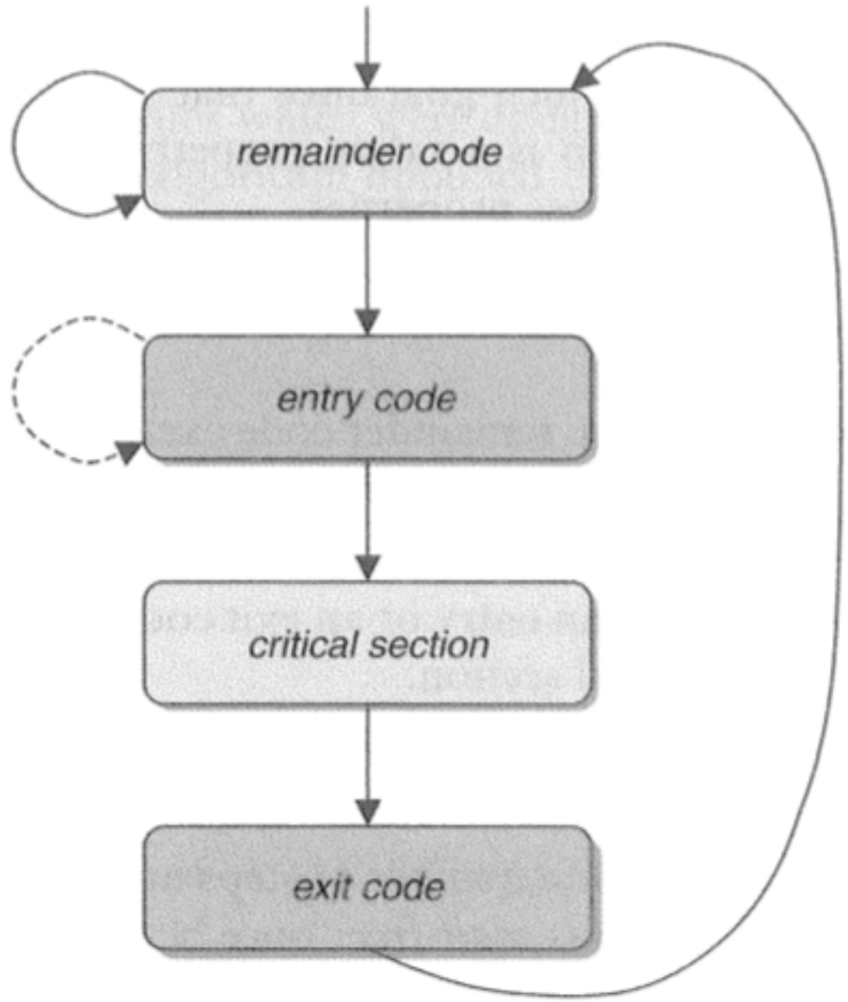
Department of Computer Science, *Johns Hopkins University*

The Point Again

- Going to take a somewhat more formal look at synchronization
  - Not just present the constructs
- Synchronization issues are the major bug in parallel programs
  - Deadlock
  - Incorrect results
- The constructs/algorithms underlying critical sections, locks, atomic variables are complex
  - Understanding them will help you use them well

Mutual Exclusion (2 processes)

- Guarantees
  - Exclusive access to a shared resource among competing processes
  - No deadlocks
  - Starvation resistance (must make progress eventually)
- Core problem of synchronization



Peterson's Algorithm

PROGRAM FOR PROCESS 0:

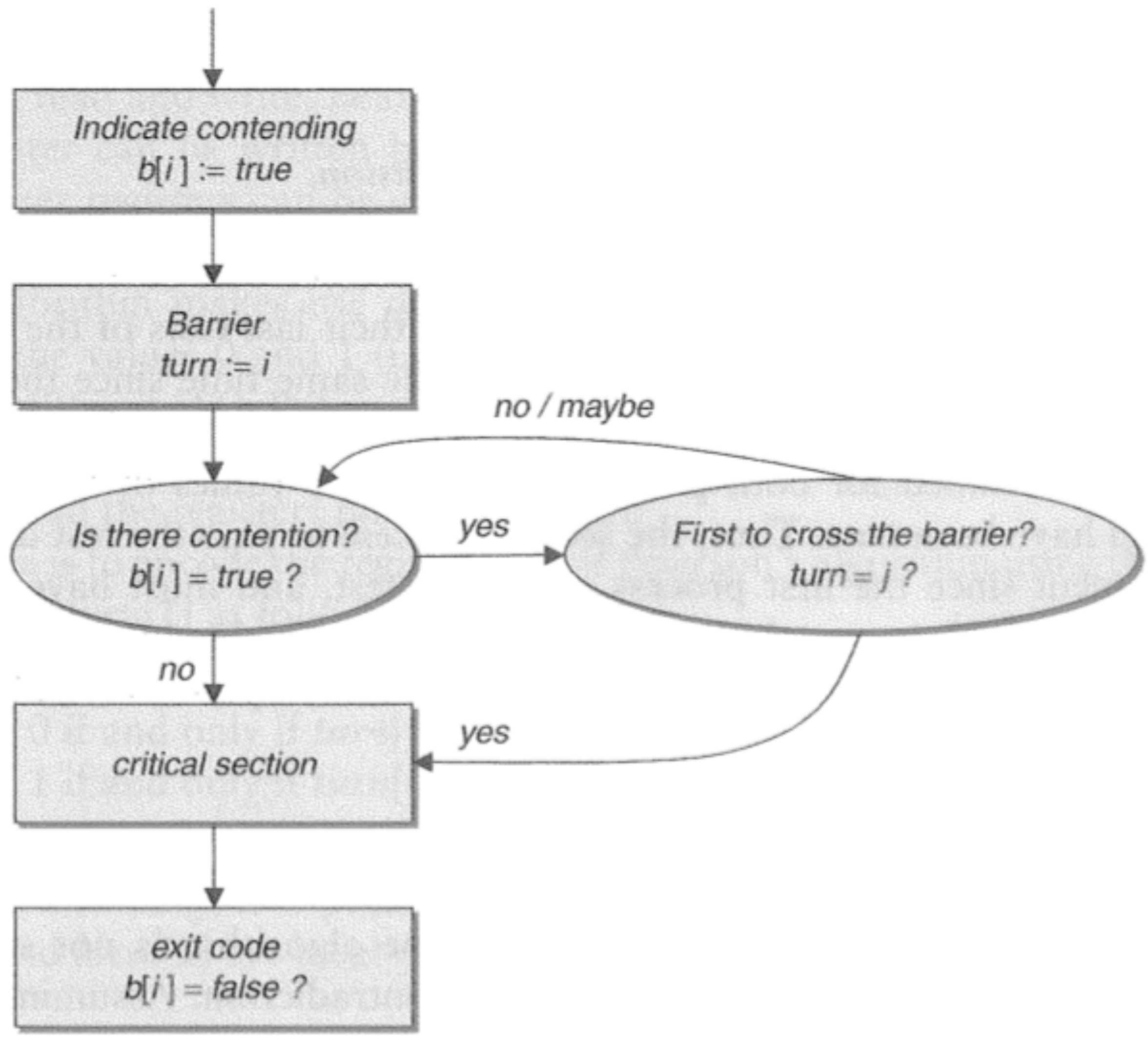
```
b[0] := true;
turn := 0;
await (b[1] = false or turn = 1);
critical section;
b[0] := false;
```

PROGRAM FOR PROCESS 1:

```
b[1] := true;
turn := 1;
await (b[0] = false or turn = 0);
critical section;
b[1] := false;
```

- b[x] indicates process b's desire for resource x
- Write to turn indicates who got there first
- Wait for other party to either
  - Give priority (through turn)
  - Not desire

Peterson's Algorithm



Properties of Petersons's Alg.

- Mutual exclusion
- Starvation resistant
- Contention free overhead = 4 accesses
- Arbitrary waits (non-preemptive)
- Uses three shared registers

- Requires *atomic* registers
  - volatile variables useful here
  - Doesn't work in message passing environments
  - Need a simple modification

On busy waiting

- The *await* construct in Peterson's algorithm *busy waiting* aka *spinning*
  - Use an active processor to poll the state of a memory location
- This is a good construct when:
  - There are many processors
  - There is no other useful work to do
  - Wait periods are very short
- The alternative is to sleep/restart
  - Typically implemented by hardware interrupts
  - More overhead to start/stop,
  - Frees hardware for processing of other tasks
- *Do power constraints change this?*