CHAPTER 7

Synchronization Examples

Exercises

7.12 Describe two kernel data structures in which race conditions are possible. Be sure to include a description of how a race condition can occur.

7.13 The Linux kernel has a policy that a process cannot hold a spinlock while attempting to acquire a semaphore. Explain why this policy is in place.

7.14 Design an algorithm for a bounded-buffer monitor in which the buffers (portions) are embedded within the monitor itself.

7.15 The strict mutual exclusion within a monitor makes the bounded-buffer monitor of Exercise 7.14 mainly suitable for small portions.

a. Explain why this is true.

b. Design a new scheme that is suitable for larger portions.

7.16 Discuss the tradeoff between fairness and throughput of operations in the readers–writers problem. Propose a method for solving the readers–writers problem without causing starvation.

7.17 Explain why the call to the `lock()` method in a Java `ReentrantLock` is not placed in the `try` clause for exception handling, yet the call to the `unlock()` method is placed in a `finally` clause.

7.18 Explain the difference between software and hardware transactional memory.