CS 423/523 Assignment 2

Published: Mar. 2, 2017
Due: Mar. 8, 2017 (9:00am)

Total: 30 points

Please upload your solutions to classes*v2. To do so, please enter classes*v2, then click the “Assignment” button on your left-hand toolbar, and finally click “Assignment3” to upload your assignment.

If you know you are going to submit your assignment late, please let us know in advance (send an email to cs423ta@cs.yale.edu). Solutions will be posted 10 days after the deadline.

Any and all resources may be used as long as you cite them, with the exception of collaborating with other people. Please do not copypasta your definitions from Wiki. We do not really care if you do this, but you are not really learning anything.

If you have ANY questions, please do not hesitate to let us know (email, office hours, etc.)
Part 1: OS definitions (1-3 line answers, 5 x 2 points each = 10 points)

1. Critical section

A code segment that should be executed atomically by only one process (or thread) at any given time; or, among n processes that have the same code / code segment. Any reasonable description is acceptable.

2. Semaphore

A synchronization (software-level) mechanism that provides more sophisticated ways (than Mutex locks) for processes to synchronize their activities, e.g., using wait and signal. Any reasonable description of signal and wait should also be fine.

3. Deadlock

Two or more processes are waiting indefinitely for an event that can be caused by only one of the waiting processes.

4. Starvation

A process may never be removed from the semaphore queue in which it is suspended

5. Busy waiting

Any reasonable answer is acceptable. My definition is busy waiting is a software-level technique that repeatedly checks (most cases, using loop to check) whether some condition is true.

Part 2: Multiple choice (6 x 2 points each = 12 points)

1. Which of the following is correct about a binary semaphore:
   a. It is a hardware-based synchronization mechanism
   b. It requires busy waiting
   c. Both of the above
   d. None of the above

2. Test and set:
   a. Returns a pointer type
   b. Is interruptible
   c. Both of the above
   d. None of the above

3. Which of the following is not a property of Peterson’s algorithm:
   a. It can handle multiple processes case
   b. Instructions, e.g., load and store, should not be interrupted
   c. It is a reasonable software solution for critical-section handling
d. None of the above

4. Which of the following is not correct about the semaphore with no busy waiting:
   a. Each semaphore has an associated waiting queue
   b. Block operation places the process on the waiting queue
   c. Wakeup operation directly puts the process in the ready queue
   d. None of the above

5. Which of the following is correct about mutex locks?
   a. The value is either 1 or 0
   b. Requires busy waiting
   c. None of the above
   d. Both of the above

6. Which of the following is not correct about semaphores?
   a. We can implement a counting semaphore as a binary semaphore
   b. Semaphores offer more sophisticated ways to achieve synchronization than mutex locks
   c. They can be used to implement bounded-buffer problem
   d. They have the same purpose as spinlocks

Part 3: A longer question (4 x 2 points each = 8 points)

```c
1 do {
2    wait(rw_mutex);
3    ...
4    /* writing is performed */
5    ...
6    signal(rw_mutex);
7 } while (true);
    
    Writer algorithm

1 do {
2    wait(mutex);
3    read_count = read_count + 1;
4    if (read_count == 1)
5        wait(rw_mutex);
6        signal(mutex);
7    ...
8    /* reading is performed */
9    ...
10    wait(mutex);
11    read_count = read_count - 1;
12    if (read_count == 0)
13        signal(rw_mutex);
14        signal(mutex);
15 } while (true);
    Reader algorithm
```
The above code is extracted from the pages 42 and 43 of cha5.ppt, i.e., the first version of writer and reader algorithm design for readers-writers problem. Please answer the following questions.

1. What is the key purpose of line 2 and line 6 in the reader algorithm (just explain one purpose)?

Line 2 and 6 mainly aim to ensure that at most one reader can execute the “entry” section, i.e., line 2-6.

2. Why do we need lines 4-5 in the reader algorithm? In other words, without line 4-5, what may happen?

We need to check if you are the first reader trying to enter the critical section, i.e., line 8. If you are the first reader, lock the resource from writers, thus keeping the resource from changing by writers.

3. Why do we need lines 12-13 in the reader algorithm? In other words, without lines 12-13, what may happen?

We need to check if you are the last reader who is reading the resource. If you are the last reader, then you can unlock the resource, thus making the resource available to writers.

4. Does this readers-writers design have the starvation problem? Why?

It has starvation problem, since writers may not have any chance to enter the critical section.