CSc 452: Principles of Operating Systems

Fall 24 (Lewis)

Test 2

Thu 7 Nov 2024

Name: ______________________________ NetID: ______________

Question	Points	Score
Page 1	20	
Page 2	20	
Page 3	20	
Deadlock Conditions	20	
All False	20	
Total:	100	

NetID: ____________________

1. (a) (7 points) In Phase 2, you used a Mailbox, from an interrupt handler, to send a message from the interrupt handler to a process that needs to be woken up. Why was it critical that you used `MboxCondSend()` instead of `MboxSend()`?

   (b) (8 points) When we discussed deadlock, I said that one classic strategy for preventing deadlock was to grab locks in a certain order. Which of the four deadlock conditions did this prevent? **Explain how it prevents the condition!**

   (c) (5 points) What is the difference between blocking a process, and performing a context switch?

NetID: ____________________

2. (a) (7 points) The Banker's Algorithm requires that you give the locking system a list of **all** of the locks you will ever need; it locks them all at once, atomically.

   Why does this make deadlock impossible?

   (b) (8 points) What is a synchronous interrupt? Also, give at least one example of an event that causes a synchronous interrupt.

   (c) (5 points) What is a "zombie" process?

NetID: ____________________

3. (a) (5 points) What is a race condition?

   (b) (5 points) What is the difference between a process and a program? Give an example of a common situation which helps illustrate the difference.

   (c) (5 points) Explain how you can use a Mailbox to implement sleep/wakeup. Be specific about what you call in each situation, and also how you initialize the system.

   (d) (5 points) Explain how you can use a Mailbox to implement a lock Be specific about what you call in each situation, and also how you initialize the system.

4. (20 points) We said that there were 4 conditions that must all be true, in order for us to have deadlock. Explain each of them, with a sentence or two.

   Circular Wait

   No Preemption

   Hold and Wait

   Mutual Exclusion

5. Each of the statements below is **False.** Explain why.

(a) (5 points) When virtual memory is turned on, the program must be careful to translate each memory access to use the proper physical address, before it actually touches memory.

(b) (5 points) Page faults are sent by the CPU when a process attempts an illegal access to virtual memory, such as following a wild pointer. The OS reports a page fault to the user as a "segfault."

(c) (5 points) When a user-mode process completes its `main()` function, it returns to kernel mode, and then eventually destroys the process.

(d) (5 points) Suppose that we have defined an order for our locks, in order to prevent deadlock. It is illegal for any process to even attempt locks out of order, since if it blocks it can participate in deadlock.