/

78/100

Midterm Examination
CS 111
Fall 2015

Name: Brandon Pon

Answer all questions. All questions are equally weighted. This is a closed book, closed notes test. You may not use electronic equipment to take the test.

1. One principle of achieving good robustness in a system is to be tolerant of inputs and strict about outputs. Why? Describe an example in the context of operating systems.

Robustness means that the system must be able to handle a wide variety of input, including invalid input, and be able to gracefully continue. This is to ensure that nothing crashes unexpectedly, and that this error cannot be exploited or abused by people who would do this to crash other people's systems on purpose. Strict output is necessary so that the system know exactly what to expect. This streamlines the whole system so that it may safely assume a certain type or range of data and act accordingly.

7

Example

2. What is the advantage of using the copy-on-write optimization when performing a fork in the Unix system?

child processes get a copy of their own stack, heap, and instructive code, and are separate from the parent process. This follows Volatile of processes, so that one cannot affect the other. An example of this is that if child process modifies data in an object, this will not unexpectedly modify the parent's object as well.

3

Not discuss copy-on-write

3. What is emulation? What is the main challenge in software emulation?

emulation is attempting to copy the functionalities of an OS within software. The main challenge of emulation is that it does not have hardware, so it must do hardware emulation. This can be cpu intensive, and so emulations are not as fast as their original OS counterpart. Hardware emulation is also very difficult, if not impossible, to do 100%. You cannot expect the same performance emulating 4 cores running the emulation on a single core machine. Parallel computing will not be the same, and there are no hardware registers, caches, or the like.

10

4. Round Robin, First Come First Serve, and Shortest Job First are three scheduling algorithms that can be used to schedule a CPU. Which one is likely to have the largest overhead? Why?

Round Robin is a preemptive scheduling algorithm as opposed to the other two, which makes it have the largest overhead. It uses context switching to give each process a time slice, and context switching is the source of high overhead. It must copy all of the instructions, data, program state/counter etc. out of a process, load a new program in, and continue. This takes lots of time.

Other algs. also need to do context switching. Just less.

9

5. What is the difference between a first and a second level trap handler? Describe one advantage of using this two-level approach to handle traps.

1st level handles basic transition to kernel, save SP, etc.

The first level trap handler is a table of possible traps. Once it finds the correct trap, it provides a pointer to the second level handler, which actually has the instructions of what to do with the trap, going into kernel mode, perhaps to request priveledged access of hardware or something else. The advantage to this is efficiency. With the first level handler, it can identify the trap very fast, and go to the correct code for the trap much faster than if it had to look through the entire second handler for the right place.

8

6. What is fate sharing? Three common interprocess communications mechanisms are messages, shared memory, and remote procedure calls. For which of these is fate sharing most likely? Why?

Fate sharing means that for interprocess communications, the resulting success or failure will be the same for both sending and receiving processes. This is most likely to occur for messages. If either the sender or receiver fail, more than likely the other process will fail as well. If one succeeds, the other will succeed and everything will carry on normally.

No, shared memory

5

7. What is a bus master? Why is a device other than a CPU likely to become a bus master, and what operations will it typically use this role to perform?

A bus master is something that controls the bus, and decides what goes through the bus at any time. A device other than the CPU is likely to be bus master because the CPU is focused mainly on running processes, and so the bus master must work and communicate with the CPU to oversee that the correct data is going to and from the right devices that the CPU needs at the moment.

6

8. What is the asynchronous completion problem? Is a spin lock a good solution for this problem? Why?

Asynchronous completion problem is caused by the fact that it is impossible to predict when a process will end. If two processes run asynchronously and another process relies on them both being completed, it must somehow ensure both process are finished somehow. A spin lock is not a good solution because it is wasteful of the CPU by constantly checking whether the processes are done or not.

10

9. In the context of locks, what is the single acquire protocol? Describe a case in which it can be safely relaxed.

The single acquire protocol means that only one process is allowed to acquire a certain resource at a time before it is locked, and other processes must wait for the lock to be released to be acquired again. A lock can be safely relaxed if only reading of the resource needs to be done. For example if many processes need to read input from a textFile, nothing is modified, so there does not need to be a lock, until someone needs to write.

10

10. Why can locks be correctly implemented using assembly language instructions like Compare and Swap or Test and Set?

These assembly language instructions are atomic, and making use of hardware, can be implemented in a way that cannot be interrupted in the middle. This is vital in preemptive scheduling and parallel programming, where timing is unpredictable.

10