

SOLUTION

**Institute of Business Administration
MIS & CS Department
Operating Systems, Fall Semester 2003
BCS IV
Second Hourly Test
October 2, 2003**

Time Allowed: One Hour

Total Marks: 100

Instructions

- a. Attempt all questions.
 - b. Maximum/Total Marks are 100.
 - c. Time allowed is 1 hour.
 - d. Do NOT write any thing on the Question Paper except your name. Provide your answers on the answer sheet provided for this purpose.
-

Question 1: State whether True (T) or False(F) (40 Marks)

1. The UNIX SVR4 Zombie process state means that the process is awaiting an event and has been swapped to secondary storage. -- FALSE
2. UNIX is unsuitable for real-time applications because it has a monolithic kernel. -- FALSE
3. In multi-threading operating system kernels, the unit of dispatching is a thread. -- TRUE
4. A thread on a multi-threading operating system can continue to run once the process that created it has died. -- FALSE
5. Thread operation latencies in Kernel-Level Threads are higher than User Level Threads. -- TRUE
6. In a multiprocessor system where each processor has its own local cache, the *cache coherence* problem is typically addressed by the operating system rather than by the hardware. -- FALSE
7. The enforcement of mutual exclusion may create deadlock or starvation. -- TRUE
8. SMP is a type of SIMD parallel processing. -- FALSE
9. If a process is swapped out, all of its threads will also be swapped out. -- TRUE
10. Linux is a micro-kernel based implementation of a UNIX operating system. -- FALSE

Question 2: (15 Marks)

List (only list) the five activities related to process management that an operating system has to perform.

Answer:

1. Process creation and termination
2. Process scheduling and dispatching
3. Process switching
4. Process synchronization and support for interprocess communication
5. Management of process control blocks

SOLUTION

Question 3:

(15 Marks)

List and briefly explain 3 advantages and 2 disadvantages of User Level Threads.

Answer:

Three Advantages of User Level Threads (ULT):

1. Thread switching does not require kernel mode privileges because all of the thread management data structures are within the user address space of a single process. Therefore, the process does not switch to the kernel mode to do thread management. This saves the overhead of two mode switches (user to kernel; kernel back to user).
2. Switching between threads can be application specific. One application may benefit most from a simple round-robin scheduling algorithm, while another might benefit from a priority-based scheduling algorithm. The scheduling algorithm can be tailored to the application without disturbing the underlying OS scheduler.
3. ULTs can run on any operating system. No changes are required to the underlying kernel to support ULTs. The threads library is a set of application-level utilities shared by all applications

Two Disadvantages of User Level Thread compare with Kernel Level Threads (KLT):

1. In a typical operating system, most system calls are blocking. Thus when a thread executes a system call, not only is that thread blocked, but all threads within that process are blocked.
2. In a pure ULT strategy, a multithreaded application cannot take advantage of multiprocessing. A kernel assigns one process to only one processor at a time. Therefore, only a single thread within a process can execute at a time. In effect, we have application-level multiprogramming within a single process. While this multiprogramming can result in a significant speedup of the application, there are applications the would benefit from the ability to execute portions of code concurrently.

Question 4:

(15 Marks)

In no more than one page describe the final version of Dekker's Algorithm using a representation of your choice (pseudo code, text, picture, flowchart, etc.)

Answer:

- A process sets its flag to indicate its desire to enter its critical section
- Other processes are checked. If they have their flag set to false, the process may immediately enter its critical section. If they have their flag also set to true then the variable turn is examined - each process gets a turn at the critical section
- If the turn variable indicates the other process' turn then the process resets its flag and periodically re-examines turn.
- When turn indicates its turn the process sets its flag and enters its critical section. After it comes out of its critical section the process sets its flag to false and turn to other process' value

SOLUTION

Question 5:

(15 Marks)

For a simple five-state process model draw a Process State Transition Diagram and a Queuing Model with Multiple Blocked Queues.

Answer:

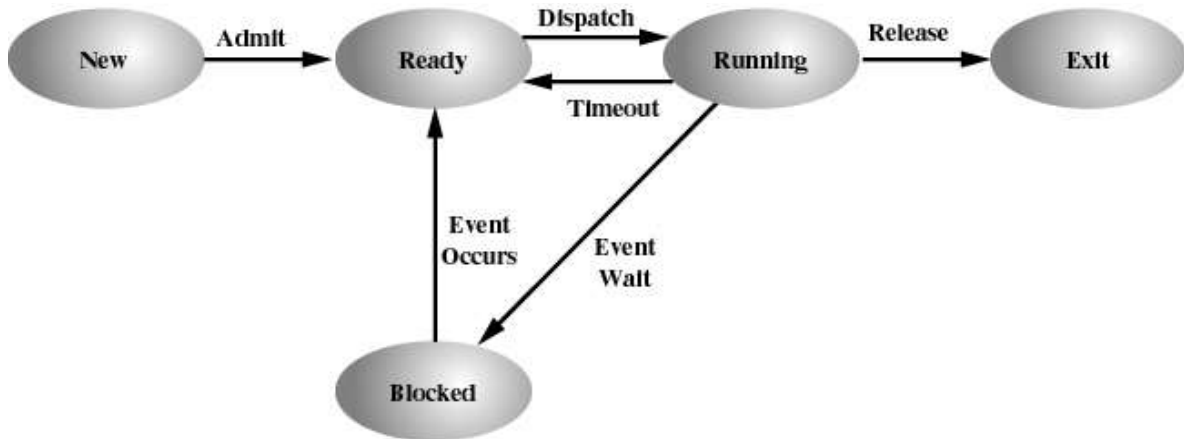
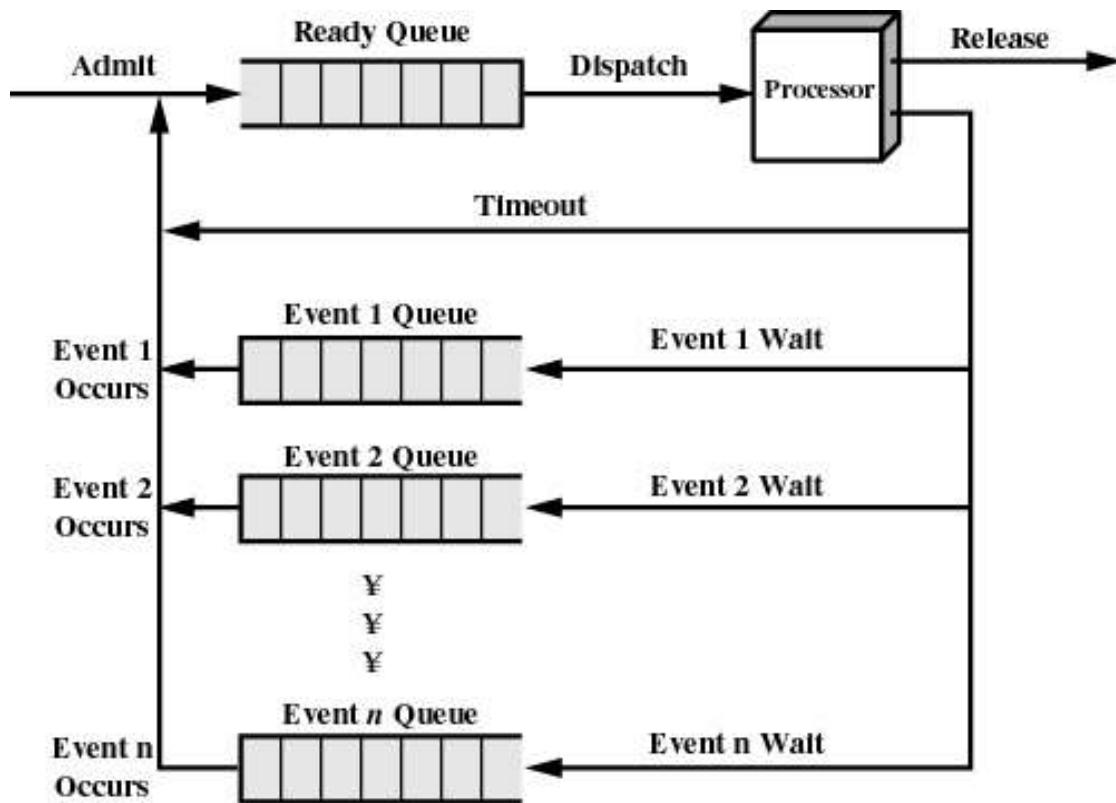


Figure 3.5 Five-State Process Model



(b) Multiple blocked queues

Queuing Model