

**Assignment 1**

**Name: Muhammad Shaheer**

**Roll no: 20p-0480**

**Subject: Operating Systems**

**Question #1 - 10 Marks**

When all of the processes in main memory are in the Blocked state,

The OS can suspend one process by putting it in the Suspend state and

Transferring it to disk. The space that is freed in main memory can

Then be used to bring in another process.

When the OS has performed a swapping-out operation, it has two

Choices for selecting a process to bring into main memory: It can

Admit a newly created process or it can bring in a previously

Suspended process. It would appear that the preference should be to

Bring in a previously suspended process, to provide it with service

Rather than increasing the total load on the system.

But this line of reasoning presents a difficulty. All of the processes that

Have been suspended were in the Blocked state at the time of

Suspension. It clearly would not do any good to bring a blocked

Process back into main memory, because it is still not ready for

Execution. Recognize, however, that each process in the Suspend state

Was originally blocked on a particular event. When that event occurs,

The process is not blocked and is potentially available for execution.

Therefore, we need to rethink this aspect of the design. There are two

Independent concepts here: whether a process is waiting on an event

(Blocked or not) and whether a process has been swapped out of main

Memory (suspended or not). To accommodate this 2 \* 2 combination,

We need four states:

● Ready: The process is in main memory and available for

Execution

● Blocked: The process is in main memory and awaiting an event.

Event occurs

● Blocked/Suspend: The process is in secondary memory and

Awaiting an event.

● Ready/Suspend: The process is in secondary memory but is

Available for execution as soon as it is loaded into main memory.

The resulting figure contains seven states shown as follows:

In principle, one could draw a transition between any two states, for a

Total of 42 different transitions.

a) List all of the possible transitions and give an example of what

Could cause each transition.

b) List all of the impossible transitions and explain why.

**Solution:**

a) **List of possible transitions and their causes:**

1. **Ready to Blocked:**

This transition occurs when a process is running and makes a system call that requires waiting for an event let’s having example reading data from a file.

**Example:** Process A is in the Ready state and issues a read operation on a file that doesn't have data ready; it transitions to the Blocked state.

1. **Blocked to Ready:**

This transition occurs when an event a blocked process is waiting for becomes available for a scenario to data is ready to be read.

**Example:** Process B was in the Blocked state waiting for user input; the user provides input, and the process transitions back to the Ready state.

1. **Blocked to Blocked/Suspend:**

When a process is in the Blocked state and the Operating Systems decide to free up main memory, it may swap out this process to secondary memory, transitioning it to the Blocked/Suspend state.

**Example:**

Operating Systems performs swapping to free up memory, so Process C, which was blocked, is moved to secondary memory.

1. **Blocked/Suspend to Blocked:**

If an event a Blocked/Suspend process was waiting for becomes available while it's in secondary memory, it transitions back to the Blocked state.

**Example:**

Process D is in Blocked/Suspend waiting for data, and the data becomes available on disk; it transitions back to Blocked state.

1. **Blocked/Suspend to Ready/Suspend:**

If a previously blocked process in secondary memory is brought back into main memory and is still waiting for an event, it transitions to the Ready/Suspend state.

**Example:**

Process E was blocked and swapped out but still awaits data; it's loaded into main memory as Ready/Suspend.

1. **Ready/Suspend to Ready:**

When a process in the Ready/Suspend state is loaded into main memory, it transitions back to the Ready state.

**Example:**

Process F, which was in Ready/Suspend, is loaded from secondary memory into main memory; it transitions to Ready.

1. **Ready/Suspend to Blocked/Suspend:**

If a process in Ready/Suspend is swapped out due to a lack of main memory, it transitions to the Blocked/Suspend state.

**Example:** OS swaps out Process G from Ready/Suspend to free up memory.

b) **List of impossible transitions and explanations:**

1. **Blocked to Ready/Suspend:**

This transition is not possible because if a process in the Blocked state becomes ready, it should transition directly to the Ready state since it's in main memory.

1. **Blocked/Suspend to Ready:**

Similar to the above, a process in the Blocked/Suspend state doesn't become directly ready when an event occurs. It first needs to be brought back into main memory, transitioning to Ready/Suspend.

1. **Ready/Suspend to Blocked:**

When a process in Ready/Suspend is brought into main memory, it's still potentially available for execution. It doesn't transition directly to Block because it's not waiting for an event at that moment.

1. **Ready to Ready/Suspend:**

A process in the Ready state doesn't transition directly to Ready/Suspend when loaded into secondary memory. It remains in the Ready state until its loaded back into main memory.

1. **Blocked to Ready/Suspend and Ready/Suspend to Blocked/Suspend:**

These transitions are not possible because they involve moving processes between main memory and secondary memory based on their event-waiting status, not their readiness for execution. Processes can move between Ready and Blocked states, and between Ready/Suspend and Blocked/Suspend states, but they don't directly switch between these pairs.



**Question #2 - 10 Marks**

Assume that at time 5 no system resources are being used except for

The processor and memory. Now consider the following events:

a) At time 5: P1 executes a command to read from disk unit 3.

b) At time 15: P5’s time slice expires.

c) At time 18: P7 executes a command to write to disk unit 3.

d) At time 20: P3 executes a command to read from disk unit 2.

e) At time 24: P5 executes a command to write to disk unit 3.

f) At time 28: P5 is swapped out.

g) At time 33: An interrupt occurs from disk unit 2: P3’s read is

Complete.

h) At time 36: An interrupt occurs from disk unit 3: P1’s read is

Complete.

i) At time 38: P8 terminates.

j) At time 40: An interrupt occurs from disk unit 3: P5’s write is

Complete.

k) At time 44: P5 is swapped back in.

l) At time 48: An interrupt occurs from disk unit 3: P7’s write is

Complete.

For each time 22, 37, and 47, identify which state each process is in. If a

Process is blocked; further identify the event on which it is blocked.

Solution:

**At Time 22:**

* P1 executed a read command at time 5 and received an interrupt at time 36, indicating that its read operation is complete. Therefore, at time 22, P1 is in the **Ready State**.
* P3 executed a read command at time 20 and received an interrupt at time 33, indicating that its read operation is complete. Therefore, at time 22, P3 is in the **Ready State**.
* P5 was swapped out at time 28 and swapped back in at time 44. Therefore, at time 22, P5 is in the **Ready State**.

**AT Time 37:**

* P1's read operation was completed at time 36, and it is not involved in any other events. Therefore, at time 37, P1 is in the **Terminated State**.
* P3 executed a read command at time 20 and received an interrupt at time 33, indicating that its read operation is complete. Therefore, at time 37, P3 is in the **Ready State**.
* P5 executed a write command at time 24 and received an interrupt at time 40, indicating that its write operation is complete. Therefore, at time 37, P5 is in the **Ready State**.

**At Time 47:**

* P3 executed a read command at time 20 and received an interrupt at time 33, indicating that its read operation is complete. However, there is no indication of any other events involving P3 after that. Therefore, at time 47, P3 is in the **Terminated State**.
* P5 executed a write command at time 24 and received an interrupt at time 40, indicating that its write operation is complete. However, there is no indication of any other events involving P5 after that. Therefore, at time 47, P5 is in the **Terminated State**.

Note:

P7 and P8 are not mentioned in the events at times:

* 22
* 37
* 47

So we can assume that they are either already terminated or not involved in any ongoing events during these specific times.