PART A

Problem Set 10

10.1 What do these numbers mentioned on each line of the program represent?

Ans. Virtual Addresses.

10.2 What is the meaning of the value 5 on the stack?

Ans. The return address for the next return instruction.

10.3 Which procedure is being executed by the processor?

Ans. PRINT_MSG.

10.4 PRINT_MSG writes a value to quote, which is stored at the address 71ff2hex, with the expectation that the value will end up on the terminal. What technique is used to make this work?

Ans. Memory-mapped I/O.

10.5 Thread 0 was running (i.e., current_thread = 0). Which instruction will the processor be running after thread 0 executes the return instruction in YIELD the next time?

Ans. 34 Continue

10.6 What address values can be on the stack of each thread?

Ans. Addresses to which called procedures return.

10.7 What expression should be evaluated in the while at address 42 to ensure correct operation of the thread package?

Ans. state[current_thread] = waiting.

10.8 Assume thread 0 is running and thread 1 is not running (i.e., it has called YIELD). What event or events need to happen before thread 1 will run?

Ans. Thread 0 calls yield and the interrupt function for input device 1 calls notify.

10.9 What values can be on the stack of each thread?

Ans. Addresses of any instruction except those in the device driver interrupt procedure.

10.10 Under which scenario can thread 0 deadlock?

Ans. When device 0 interrupts thread 0 just before the first instruction of YIELD and When device 0 interrupts thread 0 between instructions 35 and 36 in the READ_INPUT procedure on page ps-37.

Exercise 6.4

Ques. Mike R. Kernel is designing the OutTel P97 computer system, which currently has one-page table in hardware. The first tests with this design show excellent performance with one application, but with multiple applications, performance is awful. Suggest three design changes to Mike's system that would improve

performance with multiple applications, and explain your choices briefly. You cannot change processor speed, but any other aspect of the system is fair game.

Ans. The OutTel computer system performs awful, while working with multiple applications. The problem might be context switching overhead, because it requires to context switch between them if many applications are running concurrently. The context switching is not needed when there is just one application running. New page table is needed and registers that are restoring and many more things for context switching. The given system has just one-page table.

There is fair possibility of page faults. As the physical memory for several applications running together will need more physical memory than what is provided because it results in poor performance. The two causes can be the reason for poor performance.

This can be improved in several ways.

Enhancing the hardware can improve performance up.

- Increase the capacity of memory(RAM). This can reduce the number of page faults.
- Insert caches in the disk can save time by reusing the recently used blocks.
- The number of processors could be increased.
- The page fault handling time can be reduced if we add a fast disk in it.
- The hardware page tables can be added to avoid reusing a page table on every context switch.
- The hardware page table can be partitioned and allow each application to use one partition.
- To minimize the context switching time, the number of registers can be reduced.
- A page table pointer will help to switch between multiple hardware or physical memory page tables by having a page map register.

Enhancing the physical memory for applications by reducing the size of kernel.

Enhancing performance by virtual memory system.

- A TLB can be added. This will help in reducing the address translation time.
- Decreasing the size of pages helps to efficiently use the memory by using smaller parts.
- Multilevel page table can reduce the time when working set is very small.
- A more efficient page replacement algorithm and implementing working set models like prefetch pages etc.

Enhancing performance with the help of thread scheduler.

The scheduler can allocate more time slices to the processes & reducing the context switches.

Preemptive scheduling is another way, the processes could be stopped from controlling CPU.