ECE153A Final
Autumn Quarter
December 14, 1999
UC Santa Barbara
Total 120 pts

I. (28 pts) **True or False questions**. Indicate whether each of the following statement is *TRUE* or *FALSE*. If you choose *FALSE*, you must explain the reason(s) or NO credit is given.

1: Suppose a priority-based CPU scheduler always schedules the process that has the highest priority, and the scheduler employs a preemptive scheduling policy with a 100-millisecond quantum. Suppose a newly arrived process always enjoys the highest scheduling priority. The worst-case response time of a newly arrived process is 100 milliseconds.

2: A process context switch happens when a process is moved from the *Blocked* state to the *Ready* state.

3: Once a non-preemptive scheduler gives the processor to a program that is in an "while (TRUE);" loop, the non-preemptive scheduler will never regain control of the CPU.

4: Suppose a system employs a time-sliced (quantum) and round-robin scheduling policy. Having a large time-slice (quantum) improves the average turnaround time of the processes in he system.

5: The hardware and the DMA controllers know nothing about virtual memory– all they can see is physical memory and its addresses. Instead of using a page here and another there and gluing them together in virtual memory, we must allocate continuous blocks of physical memory, and we may not swap them out.

6: Suppose $P_{off,on} = P_{on}$. To conserve power, the system should place a device in the sleep mode (i.e., the soft-off mode) when the device is idle.

7: Memory mapped IO conveniently involves the use of CPU instructions in moving data to and from an I/O device. Memory mapped IO is thus an efficient mechanism to be employed to move large amount of data to/from a hard disk.

II. (35 pts) For each question, choose the letter (a), (b), (c), (d), or (e) as your answer. Score = (5$\times$ number right) - number wrong, so random guessing nets you nothing on the average. If you write down the steps that arrive at your solution to a problem, you may receive partial credit.

1: In a Linux device driver, which of the following calls must invoke init–module?
(a) open (b) read (c) write (d) close (e) ioctl.

2: Suppose that your power manager predicts that the next device idle duration will be longer than $T_{Be} > 0$. But the actual idle time is less than T_{Be}. What is the power consumption status of the device if the device is placed in the soft-off state at the start of the next idle period?
(a) The device saves power (b) The device neither saves nor wastes power (c) The device wastes power (d) indecisive.

3: Let X denote the page size and Y denote the size of a page table entry. Let V denote and virtual memory size and W the physical memory size. (All units are in Bytes.) What is the size of the virtual to physical page table?
(a) $\frac{V \times Y}{W}$ (b) $\frac{V \times Y}{X}$ (c) $2^{V - log^X} \times Y$ (d) $\frac{W \times Y}{X}$ (e) $2^{W - log^X} \times Y$.

4: You are given the following three periodic tasks. Identify a feasible real-time schedule that maximizes the total importance factor.

Task i	E_i (ms)	T_i (ms)	I_i
1	20	40	2
2	50	200	4
3	80	400	5

(a) Tasks 1, 2 and 3 (b) Task 1 and 3 (c) Tasks 1 and 2 (d) Task 1 (e) Tasks 2 and 3.

Suppose a disk drive has the following performance parameters. Answer the following three questions.

Parameter Name	*Value*
Disk Capacity	*20 GBytes*
Number of cylinders, CYL	*10,000*
Min. Transfer Rate TR	*100 Mbps*
Max. Transfer Rate TR	*200 Mbps*
RPM	*10,000*
Min. Seek Time	*3 milliseconds*
Max. Seek Time	*15 milliseconds*

5: The disk arm starts at a random location. An IO request accesses a 300 KBytes data block on track 5,000. Suppose the seek time is linear with respect to the number of tracks that the disk arm travels. What is the expected seek time of this IO?
(a) 9 ms (b) 4 ms (c) 5 ms (d) 6 ms (e) 8 ms.

6: What is the transfer time to transfer this data block on track 5,000 (excluding the seek time and rotational delay)?
(a) 2 ms (b) 8 ms (c) 12 ms (d) 16 ms (e) 24 ms.

7: Which of the following statements are true about the timeout power management policy? (Let T_{TO} denote the timeout value.)
I. A large T_{TO} is safe but could be useless.
II. A smaller T_{TO} can conserve more power.
III. The power during T_{TO} (from the time when the device is idle to the time when the device is transitioned to the off state) is wasted.
(a) I (b) I and II (c) II and III (d) I and III (e)I, II and III.

III. (24 pts) Short Answers.

1 (Device Drivers):

Suppose the following code fragments are involved in displaying strings to a terminal. What is its problem? (Hint: There is NO syntax error in the code fragment.)

```
#define MAXBUFFERLEN 80
char buffer[MAXBUFFERLEN]
int number_of_chars = 0;
int next = 0;

printString(char *string)
{
  int i;
  while (*string) {
    i = (next + number_of_chars) % MAXBUFFERLEN;
    number_of_chars++;
    buffer[i] = *string++;
  }
}

/* This interrupt routine is invoked after a character
   has been displayed and at every timer interrupt */

terminal_ISR ()
{
  while(number_of_chars) {
    print_on_screen(buffer[next]); /* print one character at a time */
    next = (next + 1) % MAXBUFFERLEN;
    number_of_chars--;
  }
  return_to_interrupt();
}
```

2 (Device Drivers): On method of reducing the number of writes to a block device (e.g., a disk) is to perform group-write. (a) Describe when group-write can be useless. (b) Describe a problem that group-write can run into?

3 (Instruction Sets): What is a register-deferred mode instruction? Suppose a register mode instruction takes 1 ns to complete its execution on a CPU. Can a register-deferred mode instruction take more than $10,000,000$ ns to complete? Why?

4 (Memory Management): Describe what is an *Inverted Page Table* and what is its primary purpose and advantages over the traditional virtual to physical page table.

IV. (17 pts). Consider four periodic tasks given in the table. Answer the following two questions:
1) According to the RMS test, can all these four tasks meet their deadlines (show all work)? (5 pts)
2) According to the exact schedulability test, will these four tasks always meet their deadlines (show all work)? If not, try using period transformation to make all tasks schedulable (context switch overhead is zero). (12 pts)

Task i	E_i (ms)	T_i (ms)
1	5	50
2	10	70
3	40	120
4	80	200

V. (16 pts) Suppose a system employs a Multi-Level Feedback Queue process scheduling policy with the following three scheduling rules:

1. There are three service queues in the system, each uses a FIFO scheduling policy. The first queue (the highest priority queue) has a quantum of 20 ms. The second queue has a quantum of 40 ms. The third queue (the lowest priority queue) has a quantum of 100 ms. A higher priority process always preempts a lower priority process. A preempted process is always given its full quantum back. A context switch takes 5 ms.

2. A newly arrived process is placed in the second queue.

3. A process' priority is demoted after a quantum-end and is promoted after an IO completion.

Answer the following two questions:

(a) How do you change the scheduling rules to minimize the response time of a newly arrived process? (6 pts)

(b) Suppose you have a CPU-bound job. How can you write a trick program to fool the scheduler to always gain the highest scheduling priority? (4 pts)

(c) As a system designer, how can you fix the scheduling rules to avoid the subterfuge attack described in part (b)? (6 pts)