Mid-Term Exam

Instructions:
(1) This is an OPEN BOOK exam. You are permitted to bring notes and other material.
(2) Your work must be strictly INDIVIDUAL. Consulting others is not permitted.

Read and Sign the statement below:

I certify that I am the person registered for the CS211 course. That no substitute is appearing for this exam on my behalf. And that my failure to STRICTLY adhere to ANY of the instructions stated above will result in severe penalties even resulting in a FAIL grade for this course.

Name :
SID :
Section :
Email :

Signature :

Instructor’s Signature :

Photo_Id verified :

Short Questions
Virtual Memory

I) The concept of virtual memory was invented to give the programmer the illusion of bigger memory than there actually is. (physical memory). While we would like to have infinite virtual memory at our disposal, this cannot be the case. What elements of the computer system conspire to limit the size of virtual memory? (5 points)

Interrupts

II) Before servicing an interrupt request, the processor saves its state in memory. (This state is associated with some process that is executing - course). Rather than saving state information in memory, I would like my system to save state information in another set of registers to reduce this overhead. The usual set of registers could then be used for the interrupting process. However, my scheme has a small flaw. What could that be? Explain. (5 points)

Cache Organizations

III) Assume your grandma graduated from Rutgers University. When she took CS211 course, cache memory was not around. She has written the following code snippets and presented them to you on Halloween. You are working on machine CS211 has a cache with the following specifications: Line size = 10 words, #Lines = 2000.

(a) for (i = 0 ; i < 1000000 ; i++)
   for (j = 0 ; j < 3000 ; j++)
   x += a[i] * b[j] ;

(b) for (j = 0 ; j < 3000 ; j++)
   for (i = 0 ; i < 1000000 ; i++)
   x += a[i] * b[j] ;

Your job is to please grandma by examining both code snippets and determining which one executes faster. Of course, to convince grandma, an explanation must be attached. (5 points)

I/O

IV) CPU performances are 5-6 orders of magnitude greater than I/O speeds. This difference is called the "Performance Gap". Over the past 2 decades, performance gap has continued to widen. Computer CS211 is unique in the sense that it does not have the famous "performance gap". (i.e. I/O runs at the same speed as the CPU). Now consider the 3 I/O techniques we discussed in class: Programmed I/O, Interrupt I/O and DMA. Discuss how these techniques will fare on my hypothetical CS211 machine. Which technique would be fastest? Why? (5 points)

Paging & Segmentation

V) How is 2-level page table different from 1-level segmentation and paging? (5 points)
Design Questions

System Bus

VI) Draw and explain the Timing Diagram for a PCI write operation. Assume that the processor needs **ONE EXTRA** cycle **BEFORE** it can put the 3rd word on the bus and that the device needs **TWO EXTRA** cycles **BEFORE** it can read the 3rd word of data.
Note: You may use the same processor as in text fig 3.22.
Hint: You may want to look at the following signal Lines - IRDY# and TRDY#.

(5+5+5)(15 points)

Cache Organizations

VII) Consider a byte-addressable memory with a 32-bit address space. The size of the cache is 64K bytes. The size of a cache line is 32 bytes. The word length for that computer is same as the byte length. Initially, the cache is "fresh" (clean). The CPU now makes the following sequence of references (Column 1 in the table below)

<table>
<thead>
<tr>
<th>Reference</th>
<th>Line #</th>
<th>Status (Hit/Miss)</th>
<th>Replacement (Line #)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>298</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>91</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>141</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>581</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>319</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>360</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

a. Fill the above table for a direct mapped organization of the cache.
b. What is the **HIT RATIO**?
c. Given the cache hit time to be 20ns and a cache miss time of 90ns, calculate the average access time based on the hit ratio computed above.
d. Calculate the actual amount of memory needed for the cache i.e. inclusive of the tag memory.

(5+5+5+5) (20 points)

DMA

VIII) Consider a system with a 200MHz processor.
a. On this system, a bus cycle equals 10 CPU cycles. Each bus cycle enables to transfer one word of data (16 bits) to be read/written from/to an I/O device. If the processor utilizes the bus 10% of the time and a DMA is used in this configuration, then what would be the data transfer rate from an I/O device to memory that would be experienced.
b. If the processor uses programmed I/O for I/O transfer, and each one word I/O transfer requires the processor to execute 25 CPU cycles, estimate the maximum I/O data transfer rate in words/sec. (10 points)

**RAID Organization**

IX) Assume a RAID 4 organization with 3 disks and a parity disk. Suppose that read and write commands from the CPU have the following form:
   - Logical_Read (start_block, number_of_blocks)
   - Logical_Write (start_block, number_of_blocks)

*LOGICAL* commands from the CPU are translated internally by the controller to actual *PRIMITIVES*. The primitives are:
   - Physical_Read (block number) : Read a block from disk with the given number.
   - Physical_Write (block number, data) : Write “data” to a block to a given block number.
   - Compute_Parity (Block1, Block2) : Compute the parity of blocks Block1 and Block2.

Using the above primitives, give the call sequences for each of the following. You may use any temporary variables where necessary.
   (a) Logical_Read (1, 1)
   (b) Logical_Write (2, 2)
   (c) Logical_Read (3, 3)
   (d) Logical_Write (3, 3) (10 points)

**Operating Systems: Virtual Memory**

X) Consider a 2-level page table based virtual memory system, with the following characteristics.
   - A page size of 16 words (1 word = 1 byte).
   - The number of the entries in the first level page table (Master Directory) is 8
   - A 10-bit logical address (Virtual Memory Address)
   - A 9-bit physical address

Page table entries occupy only one word of memory.

The following is the snapshot of memory at a particular point in time. The master directory is located at Location 7. Only the relevant regions of memory have been illustrated. The second column in each entry of the page table denotes whether the entry is valid or not (valid = 1, invalid = 0). The number in the left of the figure denotes the address.
<table>
<thead>
<tr>
<th></th>
<th>07</th>
<th>320</th>
<th>064</th>
<th>512</th>
<th>128</th>
<th>160</th>
<th>224</th>
<th>544</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>256</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>064.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>064.</td>
<td>25</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>064.</td>
<td>29</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>160.</td>
<td>30</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>160.</td>
<td>31</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>160.</td>
<td>22</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>256.</td>
<td>25</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>256.</td>
<td>6</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320.</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320.</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320.</td>
<td>10</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320.</td>
<td>13</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320.</td>
<td>17</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320.</td>
<td>11</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320.</td>
<td>20</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
a. Find the logical address for the physical address 219.
b. Find the physical address of the logical address 91.

(5+5)(10 points)

Operating Systems: Scheduling

XI) An I/O bound program is one that, if run alone, would spend more time waiting for (or doing) I/O than using the processor. A processor bound program is the opposite. Suppose a short term scheduling algorithm favors these programs that have used little processor time in the recent past. Explain why this algorithm favors I/O bound programs and yet does not permanently deny processor time to processor-bound programs.

(10 points)