University of Toronto

October 2011 Test 1
Course: CSC369H1F
Instructor: Karen Reid
Duration: 45 minutes
Aids allowed: One 8.5x11 sheet (double-sided) of notes

Last Name: ____________________________________________
Given Name: __________________________________________
Student Number: _______________________________________

_____________________________________________________

Marking Guide

This test consists of 4 questions on 5 pages (including this one). Please write legibly and be as specific as possible. Precise answers will be given higher marks than vague ones, and marks will be deducted for any incorrect statements in an answer.

# 1: ______/11
# 2: ______/ 3
# 3: ______/ 6
# 4: ______/ 6

TOTAL: ______/26
Question 1. Processes and System Calls

1 mark Part (a)
Name two system calls that might cause a process to change from the running state to the blocked state.
read, write, wait (not fork, exit)

1 mark Part (b)
Briefly describe one event that would cause a process to move from the running state to any other state involuntarily. (In other words, the event is not generated by the process itself.)
Interrupt, suspend, kill

1 mark Part (c)
Describe the characteristics of a process in the ready state.
A process that could begin executing instructions if it had the CPU.

2 marks Part (d)
Explain how the trap frame is used in OS161.

It is used to store registers, pass parameters between user and system space.
Part (e)  
1 mark
In the OS161 implementation of \texttt{thread\_exit}, a thread detaches its children before exiting. Explain why.

\textit{If the parent is not interested in the exit value of the children, no one else will be. Detaching the children is the way of noting that the exit value of the children will not be needed.}

Part (f)  
1 mark
Define the term “zombie” in the context of operating system processes.

\textit{A zombie process is a process that has exited, but whose pid and exit status need to be retained until the parent terminates or collects the exit status.}

Part (g)  
4 marks
In one line, define the term “system call”. Then, in list form, describe what steps must be taken to invoke, serve, and return from a system call. You may refer to OS/161 functions if that helps in your description.
Question 2. Virtual Memory

Consider a virtual memory system that uses a 3-level hierarchical page table.

Part (a) In the worst case, how many memory accesses are required to read one byte of memory?
4 (3 for page table and one for the real one)

Part (b) How many memory accesses are required to read one byte of memory in the optimal case? Briefly outline the mechanisms used to reduce the number of memory accesses from the worst case.
0 or 1 - TLB can cache address translation, and if the memory happens to be in the cache, it could be 0 accesses.

Question 3. Dynamically partitioned physical memory

Part (a) Explain how address translation works when physical memory is dynamically partitioned.
A logical address is translated to a physical address by adding the value of the base register to the logical address.

Part (b) Describe two major drawbacks of dynamically partitioned memory.
External Fragmentation, Needed to know memory size at process load time.

Part (c) Explain how such a system could satisfy requests to increase the amount of physical memory for a process.
Relocate process, might need to compact, might need to reject if insufficient memory.
6 marks Question 4. Page Tables and Address Translation

The following questions test your understanding of paging and address translation. The memory system in question has **24 bit virtual (logical) addresses** with **2^{12} byte pages**.

Here is a fragment of a page table. “——” indicates that a page is not resident in physical memory.

<table>
<thead>
<tr>
<th>Logical Page Number</th>
<th>Physical Frame Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0x010</td>
<td>0x2C</td>
</tr>
<tr>
<td>0x011</td>
<td>0x8B</td>
</tr>
<tr>
<td>0x012</td>
<td>——</td>
</tr>
<tr>
<td>0x013</td>
<td>——</td>
</tr>
<tr>
<td>0x014</td>
<td>0x00</td>
</tr>
<tr>
<td>0x015</td>
<td>——</td>
</tr>
<tr>
<td>0x016</td>
<td>0x03</td>
</tr>
<tr>
<td>0x017</td>
<td>0x40</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

For reference:

<table>
<thead>
<tr>
<th>Hexadecimal</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x1</td>
<td>0001</td>
</tr>
<tr>
<td>0x4</td>
<td>0100</td>
</tr>
<tr>
<td>0x8</td>
<td>1000</td>
</tr>
<tr>
<td>0xE</td>
<td>1111</td>
</tr>
</tbody>
</table>

| 2^4 = 16   |
| 2^8 = 256  |
| 2^{10} = 1024 |
| 2^{12} = 4096 |
| 2^{14} = 16384(16K) |
| 2^{16} = 65536(64K) |
| 2^{20} = 1048576(1M) |
| 2^{24} = 16777216(16M) |

1 mark Part (a) At maximum, how many entries are in a process’s page table?

\[2^{12}\]

1 mark Part (b) If a page table entry takes 4 bytes, how many entries can be stored in one page?

\[\frac{2^{12}}{2} = 2^{10}\]

1 mark Part (c) If there are \(2^8\) physical page frames, how large is physical memory, in bytes?

\[2^8 \times 2^{12} = 2^{20}\]

1 mark Part (d) Using the table above, what physical address does the virtual address 0x01008B correspond to?

0x2C08B

1 mark Part (e) Using the table above, what virtual address does the physical address 0x4008B correspond to?

0x01608B

1 mark Part (f) If the virtual address 0x01502C is accessed, what will occur?

page fault