Assignment 1 for CISC 324, Winter 2018

Due Thursday, Jan 18 at the 3:30 lecture

[Information about labs is given at the end of this document. Labs 1 and 2 are due Monday, Jan 22 at the 2:30 lecture.]

Handing in assignments and labs

Hand in hardcopy (paper) for your assignment and lab solutions. There is no on-line handing in of assignments or labs. Your assignment answers may be typed or handwritten (please write neatly enough so that the TA can easily read your answers).

I do not accept late papers. If you cannot make it to lecture when an assignment or lab is due, then hand it in early:

- During business hours, bring your assignment/lab to Goodwin 557. That's the School of Computing Office, located on the fifth floor of Goodwin Hall across from the elevators. Ask the receptionist, Erin Gunsinger, to put your assignment into my mailbox. Alternatively, you can slide the assignment/lab under my office door Goodwin 720.
- After business hours, slide the assignment/lab under the door of Goodwin 557. Clearly label it as "CISC324 for Prof. Blostein", and Erin will put it into my mailbox the next day.

Assignments and labs may be completed individually or with a partner

If you work with a partner, write both names on the assignment and lab papers. Make sure that both of you participate fully. Working with a partner can help both of you learn: discuss the concepts, ask each other questions, explain to each other.

Carefully working through the assignments and labs is essential for learning the course material. My exam questions are aimed at testing whether students thoroughly understand the assignments and labs.

Readings for Assignment 1

Readings in the CISC324 Course Reader, for questions (1) to (6) below. Most of this is review of concepts from previous courses.

Pages 5-8 Levels of Description of a Computer System

Pages 9-15 Overview of the Instruction Execution Cycle

In order to provide a clear review of instruction execution, the course reader provides details about the Pentium assembly language and machine language. You can easily answer question (3) to (6) by referring to these pages in the course reader. Rest assured that you are not expected to memorize these details. For CISC324 exams you need to understand instruction execution in general, but you are not expected to know any details specific to the Pentium assembly language and machine language.

Review of interrupts

- Page 13, step 4: interrupt handling is a step in the instruction execution cycle.
- Pages 75-76: *interrupt vectors and code for handling interrupts.* How the hardware saves the old PC value and finds the new PC value (the address of the first instruction in the interrupt handler).

Processes and *Process Control Block* and *context switches* are introduced in the first two weeks of lecture. The following readings provide explanation and examples.

- Pages 16-18 Processes versus Interrupt Handlers. (These are quite different, but when you first learn about them it is possible to get them confused. These pages explain the difference.)
- Pages 77-78 describe a context switch in detail. The interrupt handler – part of the operating system kernel – saves the PC and SP values for the process that was executing on the CPU. Next it restores the PC and SP values for some other process that now gets a turn executing on the CPU.

Readings in the textbook, *Operating System Concepts*, 9th edition (or 8th edition), for questions (7) to (9) below.

Chapter 1 Introduction

Chapter 2 Operating-System Structures

Chapters 1 and 2 contain a lot of information. Read/skim these chapters for a general introduction. The assignment questions draw your attention to the topics that are most important to this course.

Questions for Assignment 1

(1) Describe three reasons to use simulation in the design of a computer system.

(2) I have access to two computers, ComputerX and ComputerY, that run the same machine language and use the same operating system. I want to move a C program from ComputerX to ComputerY. Will it work for me to transfer the object file (that's the output produced by the C compiler executing on ComputerX) to ComputerY, or do I have to transfer the source code (the C code) and recompile on ComputerY? Briefly justify your answer.

Questions (3) to (6) review assembly language, registers, machine code, and instruction execution. You need to understand these concepts as background information for CISC324 material. Pentium assembly language is used to illustrate these concepts in the course reader. Read these examples carefully, but don't try to memorize any details about the Pentium assembly language; I will not ask exam questions about this assembly language.

(3) Translate the following C code into Pentium assembly language. (Modify the example given on pages 9-10 of the course reader.)

```
total = 12;
for (i = 2;  i <= 50;  i=i+1)  {
  total = i + total + i;
  }
```

(4) Show Pentium machine code for the assembly language statement `MOV BX, 23`. (Modify the example at the start of page 11.)

(5) (a) Briefly describe the roles played by PC (Program Counter) and IR (Instruction Register) during instruction execution.

(b) Which of the following operations takes place during execution of a JMP instruction? Briefly explain.

- update the value in IR using the value in PC
- update the value in PC using the value in IR
- update the value in SP using the value in PC
- update the value in PC using the value in SP

(6) (a) Why is it necessary for the hardware (or software) to save and restore register values during an interrupt?

(b) Which of the following registers must be saved? Indicate all that apply.

AX BX CX DX PC IR MAR MDR Flags

(7) Briefly describe the meaning of the following italicized terms.

(a) *kilobyte*, *megabyte*, *gigabyte*
(b) *process*
(c) An operating system responds to events that are signaled by an *interrupt*. (Section 1.2.1)
(d) *processor*, *multiprocessor* (Section 1.3)
(e) *multiprogramming* (Section 1.4) [Watch out: don't confuse the terms *multiprogramming* and *multiprocessor*.]
(f) *Kernel* of an operating system (Section 1.1.3 and 2.7.3)
(g) *Virtual Machine* (Section 16.1 in 9th edition; section 2.8 in 8th edition)

(8) Describe the steps involved in a DMA (Direct Memory Access) transfer. Discuss the following example. Process A is executing on the CPU, and requests the transfer of 100 bytes from disk to main memory. Describe how this data transfer is carried out. Include mention of how interrupts are used, and describe what happens to process A while the transfer is taking place.
[In case it helps you to visualize this scenario, here is a concrete example. Process A calls the unix function *fread* as follows `fread(ptr, 1, 100, myfile)`. This is how Process A requests the OS to transfer 100 bytes from file *myfile* to the block of main memory whose address is given by *ptr*.]

(9) Choose one of the following answers and justify your choice. When several processes are executing,

- the operating system can let all processes use one shared stack.
- the operating system must maintain a separate stack for each process. The operating system does this by saving and restoring the value of the SP register at each context switch.

Hint: the second answer is correct. To see why the first answer is wrong, consider what happens when several processes are executing code with procedure calls. For example, suppose process A calls a proc1. Then there is a context switch to process B, and process B calls proc2. Next there is a context switch back to process A, and process A executes a *return* from proc1. Process A should go back to its main program. Does this work correctly if the two processes are using the same stack?

Information about Labs

This course does not have scheduled lab times, so you can carry out the lab work whenever you wish. The first two labs are easy and straightforward. You should be able to complete each of them in less than an hour.

The files needed for labs are available at these locations:

- on caslab machines at /cas/course/cisc324
- on the CISC324 web page www.cs.queensu.ca/home/cisc324
- on the caslab ftp site

Lab instructions are provided in the course reader. The instructions tell you what you need to hand in to demonstrate that you have completed the lab work.

Pages 84-90	Instructions for Lab 1: Introduction to Concurrency in Unix
Pages 91-92	Instructions for Lab 2: Introduction to Concurrency in Java

Lab 1 provides an introduction to concurrency in Unix, using the & and | commands to launch processes. This lab must be performed on a computer that runs Inix. You can use the caslab machines as follows. Log onto any of the caslab PCs in Goodwin Hall 248 or Walter Light 310, and put them into Unix mode by selecting XWin32 from the *start* menu. This opens up a window that runs Unix. When you are finished with the lab, please type **ps** to get a list of your processes, and make sure that you got rid of the a.out process that you created in step 6 of the lab. Type **kill <process id number>** to kill a process. If you don't kill the a.out process, it will keep running even after you log out, and will waste a lot of CPU time.

Lab 2 provides an introduction to concurrency using threads in Java. Lab 2, and all the other CISC324 labs, can be performed using any Java platform. The lab instructions tell you how to capture your Java output if you are running Java under Unix. If you are using another Java platform, you must find a way to capture your output. If you have problems with any of the labs, please contact a TA.