CISC 324 Winter 2015
Assignment 1 and the First Two Labs
Due Friday, Jan 16 at the 9:30AM lecture

Handing in assignments; complete assignments individually or with a partner
I do not accept late assignment papers. If you cannot make it to lecture when an assignment is due, then hand the assignment in early:

• During business hours, bring your assignment to the School of Computing Office (fifth floor of Goodwin Hall across from the elevators) and ask Anna to put it my mailbox.
• After business hours, slide the assignment under the door of the School of Computing office (fifth floor of Goodwin Hall across from the elevators): clearly label it as “CISC324 for Prof. Blostein”, and Anna will put it into my mailbox the next day.

Your answers may be handwritten or typed. Please make sure your handwriting is easy to read.

Assignments and labs may be completed individually or with a partner (write both of your names on the assignment and lab papers you hand in). Working as a pair can benefit both of you because discussing concepts with your partner deepens your understanding. However, make sure that both of you participate fully. Working through the assignments and labs is essential for learning the course material.

Assignment 1 Questions
For Questions (1) to (6): Read pages 6-19 of the CISC324 Course Reader, titled “Levels of Description of a Computer System”, “Overview of the Instruction Execution Cycle”, and “Processes versus Interrupt Handlers”. Much of this material should be familiar to you from previous courses you have taken. For the midterm and final exam you need to understand instruction execution in general, but you are not expected to know any details specific to the Pentium assembly language or machine language. Some of the assignment questions below refer to Pentium details: you can solve the assignment problems easily by referring to the course reader, but rest assured that you are not expected to memorize this.

For Questions (7) to (9): Read the following parts of the textbook Operating System Concepts, 9th edition (or 8th edition):
Chapter 1 Introduction
Chapter 2 Operating System Structures
Chapters 1 and 2 contain a lot of information. Read over this for a general introduction. The assignment questions draw your attention to the topics that are most important to this course.

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

(2) I have access to two computers, ComputerX and ComputerY, which 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 running 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) provide review of assembly language, registers, machine code, and instruction execution. You need to understand these concepts as background information for CISC324 material. The course reader illustrates these concepts using specific examples expressed in 80x86 assembly language. Read these examples carefully, but don’t try to memorize any details about the 80x86 assembly language; I will not ask exam questions about this assembly language.

(3) Translate the following C code into 80x86 assembly language. (Modify the example given on pages 10-11 of the course reader.)

```c
int total = 12;
for (i = 2; i <= 50; i=i+1) {
    total = i + total + i;
}
```
(4) Give 80x86 machine code for the assembly language statement MOV BX, 23. (Modify the example at the start of page 12.)

(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 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 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.
    [As a concrete example, process A could call the unix function fread with the parameters “fread(ptr, 1, 100, myfile)”. This 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. After a context switch back to process A, process A tries to return from proc1 back to its main program. What happens?

Lab Work
This course does not have scheduled lab times, so you can carry out the lab work whenever you wish. Lab instructions are given in the CISC 324 Course Reader. Find the computer files that you need for the labs here:
- on caslab machines at /cas/course/cisc324
- on the CISC324 web page www.cs.queensu.ca/home/cisc324
- on the Caslab ftp site

The lab instructions in the CISC324 Course Reader tell you what material to hand in, to show that you have completed the lab.

Lab 1 provides an introduction to concurrency in unix, using the & and | commands. The lab should take less than an hour. This lab must be performed on a computer that runs unix. You can use the caslab machines. 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 (from 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 in Java, using threads. This lab also should take less than an hour. Lab 2, and all subsequent 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.