**CPSC 457 Fall 2018 - Assignment 1**

1. Assume that a CPU cycle has three stages, as shown in class, and each stage is handled by a separate unit, namely, fetch unit, decoding unit, and execute unit. For every instruction, the fetch unit takes 8 nsec, the decoding unit takes 2 nsec, and the execute unit takes 1 nsec.

a) How many instructions per second can this CPU execute on average if the stages are not  
parallelized?

* **Fetch unit takes 8 nsec, decoding unit takes 2 nsec, executing unit takes 1 nsec so each instruction takes 11 nsec to execute. Thus, this CPU can execute 1 instructions/11 nsec = 0.0909090909 instruction/nsec = 90, 909, 909 instructions per second on average if the stages are not parallelized**

b) How many instructions per second can this CPU execute on average if all stages are  
operating in parallel?

* **Since all stages are operating in parallel, we take the slowest stage in this CPU cycle which is the fetch unit 8 nsec. Thus, this CPU can execute 1 instruction/8 nsec = 0.125 instruction/nsec = 125, 000, 000 instructions per second on average if all stages are operating in parallel**

2. Describe one benefit of using virtual machines for each of the following:

a) for a company;

* **It is great for OS research and development as normal system operation rarely needs to be disturbed from system development**

b) for a programmer;

* **The host system is protected and there is room to experiment because even if you were to run unsafe software, it will only damage your virtual OS**

c) for a regular user;

* **Virtual machines are isolated and safe from other; you can run old version of a software in one virtual OS and the updated version on another and receive the benefits of both**

d) for a system administrator.

* **System consolidation and saving money are some of the benefits. It saves a lot of money by buying one big server and administering the machines remotely**

3. Written questions.

a) Define Interrupts.

* **It is an external event delivered to the CPU to inform that an external device or application has completed its operation. Once the CPU has been informed, it can handle the interrupt appropriately**

b) Define Traps.

* **Trap is an internal event such as system calls, error conditions, etc. It is often a special instruction that switches from user mode to kernel mode and invoked a predefined function. Very similar to interrupt and often called a software interrupt**

c) Describe the differences between interrupts and traps.

* **Interrupts are external events delivered to the CPU while traps are internal events such as system calls and error conditions**
* **Interrupts are asynchronous with the current activity of the CPU while traps are synchronous.**
* **The time of interrupts are not predictable as the time of the event is unknown while traps are predictable as it occurs as a result of execution of a machine instruction**

d) Explain why interrupts and traps are handled in kernel mode instead of user mode.

* **Interrupts and traps are handled in kernel mode because the CPU is in unrestricted mode where you have full access to hardware and instruction set. Meanwhile in user mode, there is no access to hardware and there is limited instruction set. And interrupts and traps need to invoke kernel procedures, which can only be called from the kernel mode.**

4. Written questions.

a) What are the outputs of the time commands? Copy/paste this from the terminal output to

your report.

* **time ./countLines romeo-and-juliet.txt**

**4853 romeo-and-juliet.txt**

**real 0m0.187s  
user 0m0.059s  
sys 0m0.127s**

* **time wc -l romeo-and-juliet.txt   
  4853 romeo-and-juliet.txt**

**real 0m0.081s  
user 0m0.002s  
sys 0m0.000s**

b) How much time did the C++ program and ‘wc’ spend in the kernel mode and user mode,  
respectively?

* **C++ program took 0.127s in kernel mode and 0.059s in user mode while ‘wc’ took 0.000s in kernel mode and 0.002s in user mode.**

c) Why is the ‘wc’ program faster than the C++ program?

* **‘wc’ program is faster than C++ program because unlike the C++ program which reads every char (thus making around 179037 system calls according to the strace command), ‘wc’ only reads words (about 115 system calls).**

5. Programming Question

**time ./countLines romeo-and-juliet.txt  
4853 romeo-and-juliet.txt**

**real 0m0.187s  
user 0m0.059s  
sys 0m0.127s**

**time wc -l romeo-and-juliet.txt  
4853 romeo-and-juliet.txt**

**real 0m0.081s  
user 0m0.002s  
sys 0m0.000s**

**time ./myWc romeo-and-juliet.txt  
4853 romeo-and-juliet.txt**

**real 0m0.005s  
user 0m0.004s  
sys 0m0.000s**

**myWc.cpp program is much closer to the ‘wc’ than countLines.cpp is.**