CPSC 213: Assignment 10

Due: Friday, December 2, 2011 at 6pm.

No late assignments will be accepted. This rule is strictly applied and there are no exceptions.

Goal

In this assignment you extend the Simple Machine simulator to implement page-based virtual memory and then observe a test program running with two separate page tables.

Adding Virtual Memory to the SimpleMachine

You will use your implementations of **`MainMemory`** and **`CPU`** from previous assignments and a new version of the simulator provided with this assignment, by adding a new class called **`VirtualMemoryCPU`**. A skeleton of this class is provided for you.

You will implement the method **`translateAddress`** of this class, which translates a virtual address to a physical address. This method needs to read from memory using a physical address in order to read the page table. To read from using a physical address use **`physMem`** instead of the **`mem`** class used in **`CPU`** to read memory. Both **`mem`** and **`physMem`** implement the same set of methods; the only difference is that **`mem`** uses virtual addresses and **`physMem`** uses physical. The other thing you will need to do is to read the value of the page table base register, the base physical address of the current page table. This register, which is settable in the SimpleMachine GIU (bottom left), is read by the statement **`ptbr.get()`**.

To run the virtual-memory version of the simulator you use a different target (or different command-line option). The new target is the class **`SimpleMachine.Sm213Vm.`** If you select the machine using command-line arguments, change "**`-a sm213`**" to "**`-a sm213-vm`**".

Requirements

Here are the requirements for this week's assignment.

1. Implement the **`translateAddress`** method of **`VirtualMemoryCPU`** and test your implementation. Use 32-byte pages as indicated in this class.
2. Edit the program you wrote for assignment 4 as follows. First, create two separate data sections, that are copies of each other, but with different values. Second, create two additional data sections to store two pages tables. Both page tables should map the same copy of the code (there is just one) and one of the data sections that contains the program's input (each page table should map a different data section).
3. Run your version of max.s twice, once using each page table so that each execution uses a different set of inputs. Set the current page table by setting the **`ptbr`** to store the

appropriate base address for each page table, in turn. Look closely at what happens as the program runs. Record your observations.

Material Provided

A new version of the simulator in the file sm-student-213.zip. The template for **`VirtualMemoryCPU.java`** in the file code.zip.

What to Hand In

Use the `handin` program. The assignment directory is **a10**.

1. A single file called "README.txt" that includes your name, student number, four-digit cs-department undergraduate id (e.g., the one that's something like a0b1), and all written material required by the assignment as listed below.
2. Your implementation of VirtualMemoryCPU.java
3. You observations from step 3.