CPE 315

Spring 2019
Seng
Laboratory #3
Complete by 11:59pm *5/8/19* (Wednesday night)

Objectives:

- To build a working MIPS emulator

Lab Description:

In the previous lab, you wrote an assembler which parses MIPS assembly. For this lab, you will write a MIPS emulator which will model the execution of instructions on a MIPS CPU. This program will work like SPIM in that it will emulate the state of the registers and memory.

Your program will emulate the MIPS registers, data memory (use an int array size 8,192), and program counter.

You can assume the input assembly files will be formatted to the specifications given in the previous lab and that script files will be formatted correctly (all lowercase with valid commands).

The emulator will run into either of 2 modes: interactive or script

Interactive mode

In interactive mode, the program should display a prompt and wait for input:

```
mips>
```

The program should accept the following commands:

- h = show help
- d = dump register state
- s = single step through the program (i.e. execute 1 instruction and stop)
- s *num* = step through *num* instructions of the program
- r = run until the program ends
- m *num1 num2* = display data memory from location *num1* to *num2*
- c = clear all registers, memory, and the program counter to 0
- q = exit the program

Example interactive session:

```
mips> c
        Simulator reset

mips> d

pc = 0
$0 = 0          $v0 = 0         $v1 = 0         $a0 = 0
$a1 = 0         $a2 = 0         $a3 = 0         $t0 = 0
$t1 = 0         $t2 = 0         $t3 = 0         $t4 = 0
$t5 = 0         $t6 = 0         $t7 = 0         $s0 = 0
$s1 = 0         $s2 = 0         $s3 = 0         $s4 = 0
$s5 = 0         $s6 = 0         $s7 = 0         $t8 = 0
$t9 = 0         $sp = 0         $ra = 0

mips> m 100 101

[100] = 0
[101] = 0

mips> s
        1 instruction(s) executed

mips> s 5
        5 instruction(s) executed
```

Script mode

Your program should take a second optional command line argument which is a script file. This script file will contain a list of commands and will automate testing of your emulator. For example, a script file could contain:

```
d
m 100 101
```

When your program runs with a script file, it should run as if those commands were typed into the emulator.

Your program should run from the command line with 1 optional argument:

java lab3 assembly_file.asm *script_file*

The first argument is an assembly file in a format similar to the previous lab. The second argument is an optional script file.

Test input/output

test1: input script output
test2: input script output
test3: input script output

Handin electronically:

- source files
- Makefile