Verilog: Single Processor Components

Instructions: The purpose of this lab is to learn how a single piece of a general purpose processor operates. Additionally, you will learn (via presentations) how each component operates in a Verilog simulation.

Please note, all code must be in the 'lab03' directory. This directory is case sensitive!

Part 0:

If you have not already, pull down the latest from the course repository:

1. Open Bash for Ubuntu in your repository (make sure you are in your repository!)
2. If you haven't already, add upstream: 'git remote add upstream https://github.com/csu-cs/csci-330-spring-2018.git'.
3. To get the latest from the course do: 'git pull upstream master'

Part 1:

First pick a component to focus on. Here is the list:

1. Register File
2. Memory (Instruction & Data)
3. ALU and adder
4. Mux (2x1, 4x1, 8x1)
5. Decoder/ROM and Sign Extender

Email me your top 3 selections ASAP. I will be ensuring every component is covered by at least one person. **You may not get your first choice.**

Part 2:
Implement your component.

1. Depending on your component is which Verilog model to use:
   - Mux, Decode, and Sign extender should use structural.
   - ALU/Adder may use structural.
   - Register File or Memory may use any model (Structural, Data flow, or Behavioral.)
2. Be sure to include your gate delays. Assume for 2-bit inputs, they are:
   - Not: 1

- Nand: 2
- And, Or: 3
- Xor, Nor: 4
- Note: If you have structures with more than 2 inputs, calculate the delay based on the above.
- Second Note: These numbers are estimates of real-life.

3. Now implement your component and be sure to include **thorough** testing.
4. Note: We may have not gone through Behavioral or Data Flow in class. Those who pick Register File or Memory **shall** research those models.

Part 3:
Create a presentation that goes over the following:

1. Overview of functionality and design of implementation.
2. Go over the implementation and test cases.
3. Go over some issues you may have found.
4. If you went with a behavioral model, explain why did you not select structural or data flow.
5. If you went with structural/data flow, calculate and show the gates delay.

Part 4:
If you are online please submit a video presentation of your component. The easiest way I've found is to upload your video to YouTube and send me the link. In class students, be prepared to give a presentation in class (tentatively.)

Grading
Grade: 70% components, 30% presentation.

How to turn in:
Code shall be turned in via GitHub. The video link shall be emailed to me or a link shall exist in your GitHub repository. Ensure the file(s) are in your lab03 directory and push via the command line:

- $ git add <files>
- $ git commit
- $ git push

Due Date: February 19, 2018 2359
Presentation is due during class.

Teamwork: No teamwork. You may use any resource you can find, but please cite your resources.