Discussion Session 8

CS/ECE 552
Ramkumar Ravi
19 Mar 2012

Mark your calendar

IMPORTANT DATES

- **03/23** – Project Demo I (also due in class are workplan and schedule for demo 2. If you have already included this in your project plan, then you need not submit again)
- Demo I sign up sheets is with professor Wood
- **04/09** – HW5 due (HW5 – 2 verilog memory design based problems and 3 simple cache design based questions)

GENERAL PROJECT STRUCTURE (Please follow hierarchy)

- Top Level module: **proc_hier.v** (Your processor with clk-rst module added. Similar to what you did for HW3)

- In **proc_hier.v**, you will instantiate **proc.v** which will contain your processor code

- You will use **proc_hier_bench.v** as the testbench for non-pipelined processor (WARNING: Do not EDIT this file)

- You will use the single cycle memory for Demo I. The source code along with the synthesizable version is available here

http://pages.cs.wisc.edu/~david/courses/cs552/S12/includes/single-cycle-mem.html

SUBMISSION INSTRCUTIONS

- We will use handin for Demo I submission. All your .v files need to be in one folder called **demo1** which needs to be submitted (A folder called **demo1** will be created in your respective handin directories)

- You will use handin command to submit your folder

handin cs552-1 demo1 <path_to_your_demo1_folder>

- There will be ONE electronic submission per project group

- Failure to follow naming conventions, top level directory structure could cause automatic grading to FAIL (POINTS WILL BE DEDUCTED)

- **Vcheck** for all *.v files (other than testbenches) is MANDATORY. You would be reusing most of the modules designed in HW 1, 2 and 3.

TESTS FOR DEMO1

- Tests for demo1 is available here

  /p/course/cs552-david/public/html/S12/project/tests/public/complex_demo1

- Your design must ideally pass all the tests in the link above. However, a small number of failures are accepted (provided you know the reason for the failures)

HW5 – Problems 1 and 2

- Implement a hierarchical memory system in Verilog
    - Level-1 WB cache with write-allocate policy
    - Problem 1: Direct mapped cache with a four-banked, four-cycle memory
    - Cache controller FSM design is the critical part of this problem (Please refer to Figure 5.34 (page 533 of COD 4e, 2011) as a starting point)
    - Verification will also be an important part of this exercise
    - Please go through the problem specification from the website in detail
- In the second part of the problem , you will implement a 2-way set associative cache (which is required for your project as well)

Problems 3, 4 and 5

- Keeping in mind the time required to implement problems 1 and 2, I am thinking about reducing the amount of work you will have to do for the remaining problems [Yay !! Thank you TA :)]
- You will still have to spend about an hour or so on them [Right !! Thanks for nothing Mr. TA :( ]
- Problem 3 → Given a direct mapped cache with some configuration and address bits, indicate which bits correspond to tag, index and offset. Also categorize hits/misses and so on
- Problem 4 → Repeat the same for a 2-way set associative cache
- Problem 5 → Given a cache with certain parameters, calculate cache capacity and total number of bits needed to implement it