Samuel Cooper, Keeton Kowalski, Harrison Doll
ECE 552 Introduction to Computer Architecture
Demo3 final project report

Overview of Features

Our project implemented:

- MEM stage to EXE Stage forwarding for memory output -> ALU input
- MEM stage to EXE Stage forwarding for ALU output -> ALU input
- MEM stage to EXE Stage forwarding for const selection -> ALU input
- WB stage to EXE Stage forwarding for memory output -> ALU input
- WB stage to EXE Stage forwarding for ALU output -> ALU input
- WB stage to EXE Stage forwarding for const selection -> ALU input
- MEM stage to DE Stage forwarding for memory output -> Branch RS
- MEM stage to DE Stage forwarding for ALU output -> Branch RS
- MEM stage to DE Stage forwarding for const selection -> Branch RS
- WB stage to DE Stage forwarding for memory output -> Branch RS
- WB stage to DE Stage forwarding for ALU output -> Branch RS
- WB stage to DE Stage forwarding for const selection -> Branch RS

- Branches resolve in the decode stage
- Predict branches as not taken
- Adjust so that when branches are taken we are given branch_addr and branch_addr + 2

- I memory unaligned fully working
- I memory stalled fully working
- I memory cache fully working
- D memory unaligned fully working
- D memory stalled skipped*
- D memory cache partially working

*We skipped D memory stalled and went straight to implementing caching due to running out of time to do the memory stall module first

ICache Report

The tar file “demo3_icache” is the last fully working version of our code. This version implements everything listed on the first page of this report and also:

- Unaligned accesses to instruction memory
- Instruction memory stalling
- Instruction memory 2-way set-associative caching
- Unaligned accesses to data memory
- We DO NOT have data memory caching or stalling implemented in this version

We were unable to get data memory stalling/caching fully working. The next page of this report outlines how far we got while we were trying to implement data memory stalling/caching. We included this version of our code and test results to show how far we got while also passing the vast majority of tests. We pass all tests except those listed below with the version:

- Extra credit tests including exceptions and general
- rand_final/ldld_rand_large.asm
- rand_final/ldld_rand_small.asm
- rand_final/ldst_rand_large.asm
- rand_final/ldst_rand_small.asm
- most of the rand_ldst tests are also failing

These failures are likely due to not having data memory caching implemented. If this is the case, this version of our processor is passing 100% of expected tests. The following page outlines the steps and results we achieved by attempting to add data memory caching to our processor.

ICache + DCache (full) Report

The tar file “demo3_IDcache” contains the version of our code with the above changes and a broken data memory cache implementation. Our DCache works as long as there are not multiple stores/loads in a row. This issue results in the following summary of tests passing and failing.

Perf Test - PARTIAL 5/7
Complex_demoFinal - SUCCESS
Rand_final - NOTE
Rand_ldst - SUCCESS
Rand_idcache - NOTE
Rand_icache - NOTE
Rand_dcache - FAIL

Complex_demo1 - NOTE
Complex_demo2 - PARTIAL 4/8
Rand_complex - NOTE
Rand_ctrl - NOTE
Inst_tests - PARTIAL ~80%

NOTE - While running the ‘run-all’ bash script we get tests passing and failing like normal. But once our code misses an instruction and results in an infinite loop the bash script hangs.This causes the summary.log file with the tests that pass and fail to NOT be outputted. Thus, we are not able to tell which tests pass and fail for these folders as we are unable to get a report.

Without being able to use the ‘run-all’ bash script, we were forced to run the ‘all.list’ from each subfolder that the script runs. However, we found the same problem with doing this where some folders would result in infinite loops and no summary file output.

We do not have enough time to run all 1000 tests from these subfolders individually so we are just leaving this as a note to see if any partial credit can be received from the tests that pass in these noted subfolders.