Valerie Kristofic, Elena DeJaco, Darren Mascioli

vak34, ekd17, dam253

Dr. Rami Melhem

COE 1541

2 October 2018

Test Implementation Methods Project One

For testing this milestone, we made small trace files using the given trace_generator.c file. The purpose of each trace was to act like a unit test - focus on one, very specific functionality. All possible combinations were tested of branch taken and not taken, and for mode 1 prediction correct or incorrect, as well as testing data hazards for each mode. The results of running our modified pipeline on the traces files is shown on page 2, with the corresponding file name in a comment above the output.

All prediction mode 0 traces consist of 2 instructions. File branch_not_take_mode_0.tr shows that when the PC of the instruction following a branch instruction is not the target address of the branch instruction, no no-op is inserted. The file branch_taken_mode_0.tr shows that when the PC of the instruction following a branch instruction ***is*** the target address of the branch instruction, a no-op is inserted.

All prediction mode 1 traces consist of 4 instructions, and have the combinations of branch taken/not taken, and prediction correct/not correct. File branch_not_taken_mode_1_correct_prediction.tr sets up the prediction hash table to have the branch not taken, and then runs the PC sequence again and no no-op is inserted. File branch_not_taken_mode_1_incorrect_prediction.tr sets up the prediction hash table to have the

branch not taken, and then runs so that the branch is taken and a no-op is inserted. File branch_taken_mode_1_correct_prediction.tr sets up the prediction hash table to take the branch, a no-op is inserted, and then the branch is taken again with no no-op. File branch_taken_mode_1_incorrect_prediction.tr sets up the prediction hash table to take the branch, a no-op is inserted and then the branch is not taken and an additional no-op is inserted.

For the data hazard traces, the outcome is the same for each mode. The files d_hazard.tr in mode 1 and in mode 0 have a set of instructions that will not trigger a data hazard, and therefore no no-op is inserted, and a set of instructions that trigger a data hazard and the instructions are inserted.

// branch_not_taken_mode_0.tr
[cycle 5] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 6] RTYPE: (PC: 8)(sReg_a: 4)(sReg_b: 5)(dReg: 6)

//branch_taken_mode_0.tr
[cycle 5] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 6] NOP:
[cycle 7] RTYPE: (PC: 123456789)(sReg_a: 4)(sReg_b: 5)(dReg: 6)

//branch_not_taken_mode_1_correct_prediction.tr
[cycle 5] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 6] RTYPE: (PC: 8)(sReg_a: 4)(sReg_b: 5)(dReg: 6)
[cycle 7] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 8] RTYPE: (PC: 8)(sReg_a: 4)(sReg_b: 5)(dReg: 6)

//branch_not_taken_mode_1_incorrect_prediction.tr
[cycle 5] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 132456789)
[cycle 6] RTYPE: (PC: 8)(sReg_a: 4)(sReg_b: 5)(dReg: 6)
[cycle 7] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 8] NOP:
[cycle 9] RTYPE: (PC: 123456789)(sReg_a: 4)(sReg_b: 5)(dReg: 6)

//branch_taken_mode_1_correct_prediction.tr
[cycle 5] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 6] NOP:
[cycle 7] RTYPE: (PC: 123456789)(sReg_a: 4)(sReg_b: 5)(dReg: 6)
[cycle 8] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 9] RTYPE: (PC: 123456789)(sReg_a: 4)(sReg_b: 5)(dReg: 6)

//branch_taken_mode_1_incorrect_prediction.tr
[cycle 5] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 6] NOP:
[cycle 7] RTYPE: (PC: 123456789)(sReg_a: 4)(sReg_b: 5)(dReg: 6)
[cycle 8] BRANCH: (PC: 4)(sReg_a: 1)(sReg_b: 2)(addr: 123456789)
[cycle 9] NOP:
[cycle 10] RTYPE: (PC: 8)(sReg_a: 4)(sReg_b: 5)(dReg: 6)

//d_hazard.tr in mode 1
[cycle 5] RTYPE: (PC: 0)(sReg_a: 1)(sReg_b: 2)(dReg: 3)
[cycle 6] LOAD: (PC: 4)(sReg_a: 3)(dReg: 1)(addr: 0)
[cycle 7] LOAD: (PC: 4)(sReg_a: 1)(dReg: 3)(addr: 123456789)
[cycle 8] NOP:
[cycle 9] RTYPE: (PC: 8)(sReg_a: 3)(sReg_b: 2)(dReg: 1)

```
//d_hazard.tr in mode 0
[cycle 5] RTYPE: (PC: 0)(sReg_a: 1)(sReg_b: 2)(dReg: 3)
[cycle 6] LOAD: (PC: 4)(sReg_a: 3)(dReg: 1)(addr: 0)
[cycle 7] LOAD: (PC: 4)(sReg_a: 1)(dReg: 3)(addr: 123456789)
[cycle 8] NOP:
[cycle 9] RTYPE: (PC: 8)(sReg_a: 3)(sReg_b: 2)(dReg: 1)
```