Computer Science and Engineering 431
Computer Architecture
Fall 2015

Homework #4

Distributed: October 13, 2015
Due: October 26, 2015 via Angel by midnight
Points: 22

1. (2.5 pts) For the following MIPS instruction sequence:

```
or   $t1, $t2, $t3       #$t1 <- $t2 | $t3
or   $t2, $t1, $t4
or   $t1, $t1, $t2
```

   a. Indicate all the data dependencies between instructions and their type (RAW, WAR, WAW)

   b. How many of these are true data dependencies?

   c. Assuming you have the registers `$t5`, `$t6`, … available for "renaming" modify the code using as *few* of these rename registers as possible to remove as *many* data dependencies as possible.

2. (3 pts) Consider the following MIPS instruction sequence for the loop given below to be run on a 5-stage pipelined VLIW MIPS datapath with 2 instructions per bundle (the first must be an ALU or branch operation, the second must be a load or store operation).

```
loop:   lw   $2, 40($6)
        lw   $3, 60($6)
        add  $4, $3, $2
        sll  $2, $2, 2
        sw   $4, 80($7)
        sw   $2, 40($6)
        addi $6, $6, 4
        addi $7, $7, 4
        addi $1, $1, -1
        bnez $1, loop
```

   a. Show the schedule for the code in the minimum number of cycles without using loop unrolling. What is the CPI of your (not unrolled) scheduled code?

b. Now show the schedule for the code in the minimum number of cycles assuming the loop has been unrolled by the compiler *one* time. What is the CPI of the once-unrolled scheduled code?

3. (3.5 pts) In class we talked about the instruction bundle dependency checking that has to be done in a static superscalar datapath. The example we considered in class was a 2-way instruction bundle. How many load-use cross-checks have to be done for a 4-way datapath? How many for a 6-way datapath? How many RAW intra-bundle dependency checks have to be done for a 4-way datapath? How many for a 6-way datapath? How many WAW intra-bundle dependency checks have to be done for a 4-way datapath? How many for a 6-way datapath? How many read and write ports does the RF have to have for a 4-way datapath? How many for a 6-way datapath?

4. (3 pts) How many cycles will it take to execute the code from the two threads shown below, X and Y, on a single-issue, fine-grain multi-threaded (FGMT) 5-stage pipelined MIPS datapath? Use round-robin scheduling which can skip stalled threads. How many cycles does it take (total) to complete both threads? How many issue slots are wasted due to hazards? How many cycles would it take on a static superscalar datapath that completed one thread at a time?

Thread X	Thread Y
A1 – no dependencies	B1 – a load that takes 3 cycles (total) to complete
A2 – no dependencies	B2 – has a load-use hazard with B1
A3 – no dependencies, FU takes 2 cycles (total) to complete	B3 – has a FU structural hazard with A3
A4 – RAW dependency with A3	B4 – no dependencies

5. (10 pts) In this round of SimpleScalar experiments you are going to explore the performance of different datapath designs. As usual, for all of your simulations fast forward past 500,000 instructions before starting to collect simulation data and simulate 2,500,000 instructions.

Baseline Datapath: For the baseline simulations configure a single-issue (so in-order) datapath with the settings you used in HW#3 (for the first set of experiments). Hopefully you have already collected the simulation data for IPC and total number of instructions executed for your six benchmarks. Otherwise, you will have to rerun these simulations to determine the baseline data.

Static SS Datapath: Now configure your datapath as a 2-way, static (in-order) superscalar datapath with the following settings

```
–fetch:ifqsize 2 –fetch:speed 2 –fetch:mplat 3 –decode:width 2
–issue:width 2 –issue:inorder true –issue:wrongpath false –ruu:size 8 –
lsq:size 4 –res:ialu 2 –res:imult 1 –res:memport 2 –res:fpalu 2       –
res:fpmult 1 –mem:width 16
```

Keep the branch predictor (not taken), cache sizes/latencies and memory/TLB sizes/latencies as in the Baseline datapath. Collect the simulation data for your six benchmarks focusing on IPC and the total number of instructions executed.

Dynamic SS Datapath#1: Now configure your datapath as a 2-way, dynamic (out-of-order) superscalar datapath with the following settings

```
–fetch:ifqsize 2 –fetch:speed 2 –fetch:mplat 3 –decode:width 2
–issue:width 2 –issue:inorder false –issue:wrongpath true –ruu:size 8 –
lsq:size 4 –res:ialu 2 –res:imult 1 –res:memport 2 –res:fpalu 2       –
res:fpmult 1 –mem:width 16
```

Dynamic SS Datapath#2: Now configure your datapath as a 4-way, dynamic (out-of-order) superscalar datapath with the following settings

```
–fetch:ifqsize 4 –fetch:speed 2 –fetch:mplat 3 –decode:width 4
–issue:width 4 –issue:inorder false –issue:wrongpath true –ruu:size 16
–lsq:size 8 –res:ialu 4 –res:imult 1 –res:memport 2 –res:fpalu 4
–res:fpmult 1 –mem:width 16
```

For the both dynamic SS datapath experiments, keep the branch predictor, cache sizes/latencies and memory/TLB sizes/latencies as in the Baseline datapath. Collect the simulation data for your six benchmarks focusing on IPC and the total number of instructions executed. (The number of instructions *committed* for all four datapaths should be the same – 2,500,000 instructions.)

For each set of experiments, report in tabulated form the number of instructions *executed* and sim_IPC. Note that we are now using IPC (Instrs Per Cycle) rather than CPI (Cycles per Instr)! For the dynamic SS datapaths also report (in tabulated form) the average ifq occupancy rate (ifq_occupancy), ruu occupancy rate (ruu_occupancy) and the lsq occupancy rate (lsq_occupancy). Discuss your simulation results. Also provide an explanation for the changes in performance (IPC) for the different datapaths.