CSEE W3827

Fundamentals of Computer Systems

Homework Assignment 6

Prof. Martha A. Kim

Columbia University

Due December 8, 2015 at **10:10**.

Write your name **and UNI** on your solutions

Show your work for each problem; we are more interested in how you get the answer than whether you get the right answer.

The remaining problems on this assignment require analysis of the two functions, adapted from the HW#4 solutions. Except for `sll` all of the instructions are supported by the pipelined processor from the lecture slides. The `sll` instruction will flow through the pipeline in the same manner as an `addi`.

The first is `manhattan_dist`:

```
i0:    slt $t0, $a2, $a0
i1:    beq $t0, $0, i4
i2:    sub $t0, $a0, $a2
i3:    beq $0, $0, i5
i4:    sub $t0, $a2, $a0
i5:    slt $t0, $a3, $a1
i6:    beq $t0, $0, i9
i7:    sub $t1, $a1, $a3
i8:    beq $0, $0, i10
i9:    sub $t1, $a3, $a1
i10:   add $v0, $t0, $t1
```

The second is `edge_count_iter`:

```
0_start:  addi $t0, $0, 0
i1:        addi $t1, $0, -1
i2:        addi $v0, $0, 0
i3_ntop:   beq $t0, $a1, i17_ndone
i4:        sll $t2, $t0, 2
i5:        add $t2, $a0, $t2
i6:        lw  $t2, 0($t2)
i7:        beq $t2, $t1, i15_edone
i8:        sll $t2, $t2, 2
i9:        add $t2, $a0, $t2
i10_etop:  lw  $t3, 0($t2)
i11:       beq $t3, $t1, i15_edone
i12:       addi $v0, $v0, 1
i13:       addi $t2, $t2, 4
i14:       beq $0, $0, i10_etop
i15_edone: addi $t0, $t0, 1
i16:       beq $0, $0, i3_ntop
i17_ndone:
```

1. (20 pts.) Give a cycle-by-cycle execution trace for the following implementation of `manhattan_dist`, when called on (0,0) and (5,5) (i.e., `$a0=0, $a1=0, $a2=5, $a3=5`) on a five-stage pipeline with *no forwarding*. The only option to resolve hazards is to stall (i.e., insert bubbles).

2. (20 pts.) Provide a second trace of `manhattan_dist` on the fully bypassed pipeline (i.e., branches resolved in D, forwarding from W-E, M-E, and M-D).

3. (20 pts.) Imagine how `edge_count_iter` would execute on the fully bypassed pipeline. List all pairs of instructions between which one or more bubbles would occur. If a bubble occurs between i3 and i4, then you should write i3 $\longrightarrow$ i4. (HINT: Think systematically through all scenarios that result in an empty slot in the pipeline.)

4. (20 pts.) Now list where in the program (still `edge_count_iter` on the fully bypassed 5-stage pipe) data operands would be forwarded, and which forwarding path would be used. If i3 forwards the future value of $a0 to i4 using the M-E forwarding path, write $a0, i3 $\longrightarrow$ i4, M-E, per the example below. (HINT: Think through the values consumed by each instruction systematically.)

5. (20 pts.) Assuming a very large graph with one million nodes and an average of 500 edges per node, what is the CPI of the `edge_count_iter` code? Assume that the `beq` in `i7` is taken 2% of the time.