IC220: HW 8

Due: 15 Apr 2019

Full Name: _____ **Alpha:** _____

Circle Your Section: Aviv/1001 Aviv/2001 Aviv/4001 Choi/5001 Missler/5002

Total Points: 50

Preliminary: Carefully do the assigned reading for Chapter 4 (4.5-4.8)

---

1. **[5 points]** Draw a pipeline stage diagram for the following sequence of instruction. You dont need fancy pictures — just text for each stage: ID, MM, etc. Label the cycle number for each column, starting with cycle #1.

```
lw $v0, 0($a0)
lw $v1, 0($v0)
add $a0, $a0, $v1
sub $t0, $t0, $a0
```

2. Assume the previous sequence of instructions is repeated 100 times (so the processor does a load, another load, an add, a sub, then back to load, another load, an add, ...). There are no branches, just these 4 instructions repeated 100 times. **Do NOT reorder the code to improve efficiency**. *HINT: this question takes a bit more thought and care then you may realize at first.*

   (a) **[5 points]** What is the total number of cycles needed to execute those 400 instructions?

   (b) **[2 points]** What is the CPI?

   (c) **[2 points]** What is the CPI if we rant this sequence on a **single cycle** CPU?

   (d) **[4 points]** Using the two PI calculations, argue why the CPI alone may or may not provide a good estimate for the performance gained from the different CPU designs? For example, why would a single-cycle CPU not actually be faster than a pipeline CPU even if its CPI is lower?

3. Draw a pipeline stage diagram for the same sequence of instructions as before, **but this time there is no FORWARDING available**. Again, you must show the number of cycles

```
lw $v0, 0($a0)
lw $v1, 0($v0)
add $a0, $a0, $v1
sub $t0, $t0, $a0
```

4. Again, assume the previous sequence of instructions is repeated 100 times with no forwarding! There are no branches and you cannot reorder.
   (a) **[5 points]** What is the total number of cycles needed to execute those 400 instructions?

   (b) **[2 points]** What is the CPI?

5. For the following code sequence below stalls.

```
add $a0, $s0, $s1
sub $a2, $s2, $a0
add $t1, $t2, $s3
add $a3, $a0, $a2
```

   (a) **[5 points]** Show the pipeline (with any forwards!) That is, draw lines to show which stages forward to another stage.

   (b) **[5 points]** Does this pipeline requiring any stalls, yes or no? Explain why either yes or no using your pipeline diagram above.

6. For the following code sequence below stalls.

```
lw $a0, 0($t1)
lw $v0, 0($a0)
sub $t1, $s2, $t3
sw $v0, 4($s1)
```

   (a) **[5 points]** Show the pipeline (with any forwards!) with the stalling. **AND, explain why the stall must occur and forwarding doesn't help.**

   (b) **[5 points]** Rewrite the code to avoid stalls

   (c) **[5 points]** Show the pipeline (with any forwards!) that avoids the stall.