CMPU-224 Lab10 Quiz Solutions

Spring 2026

Name: _____

This is a closed book, closed notes quiz. No electronic devices are allowed. You have until 3:30pm to complete the quiz. There are a total of 4 questions and 10 points.

There should be enough space on the quiz for your answers. If you need more space to work out a problem, blank paper will be available, just ask.

If you finish with time remaining, raise you hand and I will come and collect your quiz. You may then work on the lab assignment.

Good Luck!

1. (2 points) In the RISC-V 5-stage pipeline, in which stage or stages are used or can be used to compute the address of the next instruction?

   _____

   Solution: **Fetch** stage: computes **PC + 4** (next sequential instruction address)

   Execute stage: computes **PC + Offset** (branch/jump target address)

2. (2 points) Consider the following three-stage pipeline. The propagation delay of each stage is shown in the diagram below. Each pipeline register takes 20 ps to latch its value.

Stage A 100 ps	Reg 20 ps	Stage B 50 ps	Reg 20 ps	Stage C 75 ps	Reg 20 ps

   What is the shortest possible clock cycle period (in ps) for this pipeline? Be sure to count the propagation delay and the time to latch the data into the register.

   Minimum clock period: _____ ps

   What is the total latency of this pipeline assuming the shortest possible clock cycle? I.e., what is the total time for a single instruction to pass through all three stages and be latched into the final pipeline register?

   Total latency: _____ ps

Solution: In a pipeline, every stage is clocked at the same rate, determined by the **slowest** stage. The slowest stage is Stage A at 100 ps, so each stage requires at least 100 ps per clock cycle plus 20 ps to latch the result. This gives us a minimum period of 120 ps. With 3 stages at 100 ps each and 3 pipeline registers at 20 ps each:

$$\text{Latency} = 3 \times 100 + 3 \times 20 = 300 + 60 = \textbf{360 ps}$$

3. For each of the following RISC-V code fragments, determine the **total number of clock cycles** the fragment takes to complete on the 5-stage pipeline with forwarding and predict-not-taken branching. When there are no more instructions left to execute, assume the processor stops after the last instruction finishes the writeback stage.

   (a) (1 point)

```
1    addi  t0, x0, 5
2    add   t1, t0, t0
```

   Total cycles: _____

   Solution: RAW hazard on `t0`, resolved by forwarding with **0 stall cycles**.
   $2 + 4 + 0 =$ **6** cycles.

   (b) (1 point)

```
1    lw    t0, 0(t1)
2    add   t2, t0, t3
```

   Total cycles: _____

   Solution: Load/use hazard on `t0`. The `lw` does not produce its result until the Memory stage, so **1 stall cycle** is inserted.
   $2 + 4 + 1 =$ **7** cycles.

   (c) (1 point)

```
1    beq     x0, x0, target
2    addi    t0, x0, 1
3  target:
4    addi    t1, x0, 2
```

   Total cycles: _____

Solution: The branch `beq x0, x0, target` is **always taken** (`x0` = `x0`). Under predict-not-taken this is a misprediction, costing **2 bubble cycles**. The `addi t0` is squashed and does not retire. Instructions retired: 2 (`beq` and the target `addi`).
$2 + 4 + 2 = \mathbf{8}$ cycles.

(d) (1 point)

```
1     bne     x0, x0, target
2     addi    t0, x0, 1
3  target:
4     addi    t1, x0, 2
```

Total cycles: _____

Solution: The branch `bne x0, x0, target` is **never taken** (`x0` = `x0`, so the not-equal test fails). Under predict-not-taken this is **correctly predicted** with **0 penalty cycles**. Instructions retired: 3.
$3 + 4 + 0 = \mathbf{7}$ cycles.

4. (2 points) When the hazard unit detects a **load/use hazard**, which of the following correctly describes the control actions applied to the pipeline stages on the next clock cycle? **Circle one.**

   A. IF: normal, ID: normal, EX: bubble, MEM: normal, WB: normal

   B. IF: stall, ID: stall, EX: bubble, MEM: normal, WB: normal

   C. IF: normal, ID: bubble, EX: bubble, MEM: normal, WB: normal

   D. IF: stall, ID: bubble, EX: stall, MEM: normal, WB: normal

Solution: (B) is correct. A load/use hazard requires the Fetch and Decode stages to **stall** (hold their current instructions) while a **bubble** (NOP) is injected into the Execute stage. Memory and Writeback continue normally. This gives the load one extra cycle to complete its memory read so the value can be forwarded.

Choice (C) describes the **branch misprediction** response, not load/use.