Name: _____

Pipeline Implementation/Operation

Many of these exercises come from the Patterson and Hennessey text.

1. Assume that `t1` is initialized to 11 and `t2` is initialized to 22. Suppose you executed the code below on a version of the textbook's pipeline that does not handle data hazards (i.e., the programmer is responsible for addressing data hazards by inserting `nop` instructions where necessary). What would the final values of registers `t3` and `t4` be?

```
addi t1, t1, 5
add t3, t1, t2
addi t4, t1, 15
```

2. Assume that `t1` is initialized to 11 and `t2` is initialized to 22. Suppose you executed the code below on a version of the textbook's pipeline that does not handle data hazards (i.e., the programmer is responsible for addressing data hazards by inserting `nop` instructions where necessary). What would the final values of register `t5` be?

```
addi t1, t1, 5
add t3, t1, t2
addi t4, t1, 15
add t5, t1, t1
```

3. Add `nop` instructions to the code below so that it will run correctly on a pipeline that does not handle data hazards.

```
addi t1, t1, 5
add t3, t1, t2
addi t4, t1, 15
add t5, t3, t2
```

4. Consider the assembly code fragment below:

```
sw $16, 12($6)
lw $16, 8($6)
beq $5, $4 LABEL
add $5, $1, $4
sub $5, $15, $4
```

Name: _____

Suppose we modify the pipeline so that it has only one memory (that handles both instructions and data). In this case, there will be a structural hazard every time a program needs to fetch an instruction during the same cycle in which another instruction accesses data.

Draw a pipeline diagram to show were the code above will stall.

5. Which of the two pipeline diagrams below better describes the operation of the pipeline's hazard detection unit? Why?

   Choice 1:

```
lw $1, $2(0)   IF ID EX ME WB
add $3, $1, $4    IF ID EX .. ME WB
or $5, $6, $7        IF ID .. EX ME WB
```

   Choice 2:

```
lw $1, $2(0)   IF ID EX ME WB
add $3, $1, $3    IF ID .. EX ME WB
or $5, $6, $7        IF .. ID EX ME WB
```

6. **Not due for credit** This exercise is intended to help you understand the cost/complexity/performance tradeoffs of forwarding in a pipelined processor. Problems in this exercise refer to pipelined datapaths from Figure 7.53. These problems assume that, of all the instructions executed in a processor, the following fraction of these instructions have a particular type of RAW data dependence. The type of RAW data dependence is identified by the stage that produces the result (EX or MEM) and the next instruction that consumes the result (1st instruction that follows the one that produces the result, 2nd instruction that follows, or both). We assume that the register write is done in the first half of the clock cycle and that register reads are done in the second half of the cycle, so "EX to 3rd" and "MEM to 3rd" dependences are not counted because they cannot result in data hazards. We also assume that branches are resolved in the EX stage (as opposed to the ID stage), and that the CPI of the processor is 1 if there are no data hazards.

EX to 1st Only	MEM to 1st Only	EX to 2nd Only	MEM to 2nd Only	EX to 1st and EX to 2nd
5%	20%	5%	10%	10%

Assume the following latencies for individual pipeline stages. For the EX stage, latencies are given separately for a processor without forwarding and for a processor with different kinds of forwarding.

IF	ID	EX (no fwd.)	EX (full fwd.)	EX (fwd. from EX/MEM only)	EX (fwd. from MEM/WB only)	MEM	WB
120 ps	100 ps	110 ps	130 ps	120 ps	120 ps	120 ps	100 ps

Name: _____

(a) For each RAW dependency listed above, give a sequence of at least 3 assembly statements that exhibits that dependency.

(b) For each RAW dependency above, how many `nop`s would need to be inserted to allow your code from 6a to run correctly on a pipeline with no forwarding or hazard detection? Show where the `nop`s could be inserted.

(c) Analyzing each instruction independently will over-count the number of `nop`s needed to run a program on a pipeline with no forwarding or hazard detection. Write a sequence of three assembly instructions so that, when you consider each instruction in the sequence independently, the sum of the stalls is larger than the number of stalls the sequence actually needs to avoid data hazards.

(d) Assuming no other hazards, what is the CPI for the program described by the table above when run on a pipeline with no forwarding? What percent of cycles are stalls? (For simplicity, assume that all necessary cases are listed above and can be treated independently.)

(e) What is the CPI if we use full forwarding (forward all results that can be forwarded)? What percent of cycles are stalls?

(f) Let us assume that we cannot afford to have three-input multiplexors that are needed for full forwarding. We have to decide if it is better to forward only from the EX/MEM pipeline register (next-cycle forwarding) or only from the MEM/WB pipeline register (two-cycle forwarding). What is the CPI for each option?

(g) For the given hazard probabilities and pipeline stage latencies, what is the speedup achieved by adding each type of forwarding (EX/MEM, MEM/WB, or full) to a pipeline that has no forwarding?

(h) What would be the additional speedup (relative to the fastest processor from part 6g) if we added time-travel forwarding that eliminates all data hazards? Assume that the yet-to-be-invented time-travel circuitry adds 100 ps to the latency of the full-forwarding EX stage.

(i) The table of hazard types has separate entries for "EX to 1st" and "EX to 1st and EX to 2nd". Why is there no entry for "MEM to 1st and MEM to 2nd"?