Computer Organization and Structure

1. If the time for an ALU operation can be shortened by 25% (compared to the following table);

Instruction class	Instruction fetch	Register read	ALU operation	Data access	Register write	Total time
Load word (lw)	200 ps	100 ps	200 ps	200 ps	100 ps	800 ps
Store word (sw)	200 ps	100 ps	200 ps	200 ps		700 ps
R-format (add, sub, and, or, slt)	200 ps	100 ps	200 ps		100 ps	600 ps
Branch (beq)	200 ps	100 ps	200 ps			500 ps

    a. Will it affect the speedup obtained from pipelining? If yes, by how much? Otherwise, why?

    b. What if the ALU operation now takes 25% more time?

2. We have a program core consisting of five conditional branches. The program core will be executed thousands of times. Below are the outcomes of each branch for one execution of the program core (T for taken, N for not taken).

```
Branch 1: T-T-T
Branch 2: N-N-N-N
Branch 3: T-N-T-N-T-N
Branch 4: T-T-T-N-T
Branch 5: T-T-N-T-T-N-T
```

Assume the behavior of each branch remains the same for each program core execution. For dynamic schemes, assume each branch has its own prediction buffer and each buffer initialized to the same state before each execution. List the predictions for the following branch prediction schemes:

    a. Always taken

    b. Always not taken

    c. 1-bit predictor, initialized to predict taken

    d. 2-bit predictor, initialized to weakly predict taken

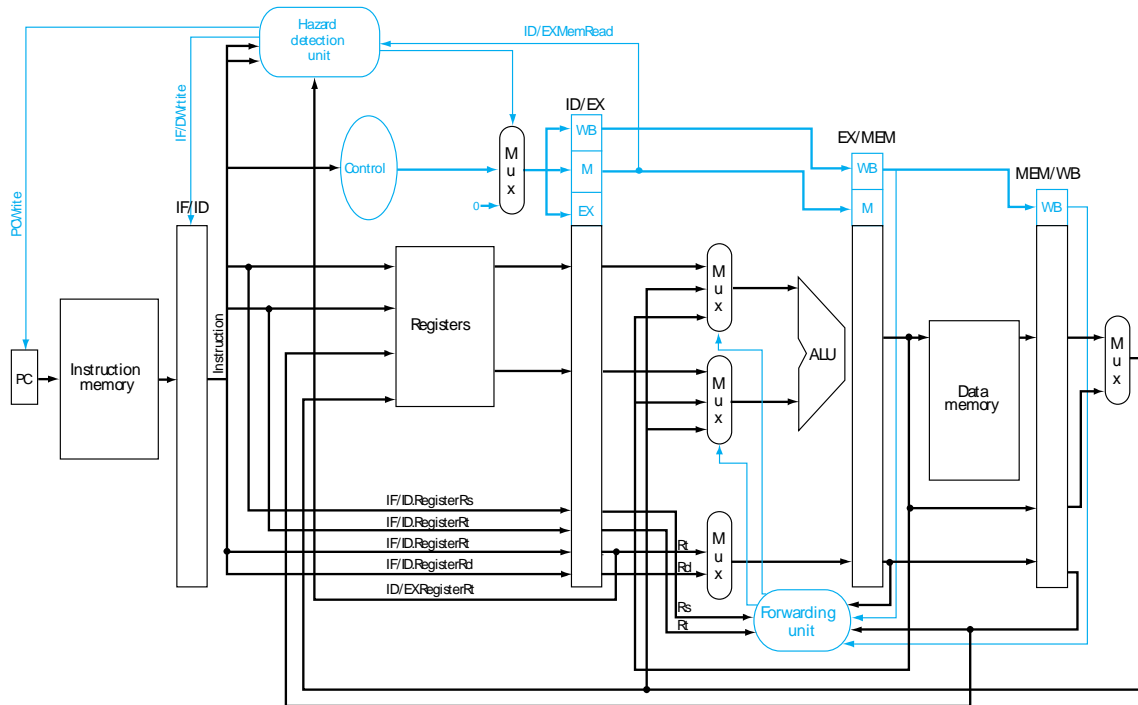What are the prediction accuracies?

3. Consider executing the following code on the pipelined datapath shown as the following figure:

```
add   $2, $3, $1
sub   $4, $3, $5
```

```
add     $5, $3, $7
add     $7, $6, $1
add     $8, $2, $6
```


a. At the end of the fifth cycle of execution, which registers are being read and which register will be written?

b. Explain what the forwarding unit is doing during the fifth cycle of execution. If any comparisons are being made, mention them.

c. Explain what the hazard detection unit is doing during the fifth cycle of execution. If any comparisons are being made, mention them.

4. Identify all of the data dependencies in the following code. Which dependencies are data hazards that will be resolved via forwarding? Which dependencies are data hazards are will cause a stall?

```
add     $3, $4, $2
add     $5, $3, $1
lw      $6, 200($3)
add     $7, $3, $6
```

5. The following code contains a "read after write" data hazard that is resolved by forwarding:

```
add     $2, $3, $4
add     $5, $2, $6
```

Consider the similar situation in which a memory read occurs after a memory write:

```
sw      $7, 100($2)
lw      $8, 100($2)
```

Write a paragraph describing how this situation differs from the one involving registers, and describe how the potential "read after write" problem is resolved.