CSE 140 HW #2b

Solve the following problems and place your answers in a text document for submission.

1. In this exercise, we examine how pipelining affects the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies:

IF	ID	EX	MEM	WB
400ps	300ps	200ps	350ps	250ps

Also, assume that instructions executed by the processor are distributed as follows:

ALU	BEQ	LW	SW
40%	30%	20%	10%

a. What is the clock cycle time in a pipelined and non-pipelined processor?

b. What is the total latency of an LW instruction in a pipelined and non-pipelined processor?

c. If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what are the new clock cycle time and latency of an LW in the pipelined processor?

d. Assuming there are no stalls or hazards, what is the utilization of the data memory?

e. Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?

f. Instead of a single-cycle organization, we can use a multi-cycle organization where each instruction takes multiple cycles but one instruction finishes before another is fetched. In this organization, an instruction only goes through stages it actually needs (e.g., SW only takes 4 cycles because it does not need the WB stage). What are the average cycle times of each type of processors (single-cycle, multi-cycle, and pipelined)?

2. In this exercise, we examine how data dependences affect execution in the basic 5-stage pipeline described in Chapter 4.5 (An overview of pipelining). Problems in this exercise refer to the following sequence of instructions:

```
add  r3, r1, r2
add  r2, r1, r3
add  r2, r2, r3
```

Also, assume registers can be read and written in the same cycle and the following cycle times for each of the options related to forwarding:

Without Forwarding	With Full Forwarding	With ALU-ALU Forwarding Only
200ps	350ps	300ps

a. Indicate ALL dependences and their type (RAW/WAR/WAW).

b. Assume there is **no forwarding** in this pipelined processor. Draw a pipeline execution diagram. Insert **nop** instructions to eliminate any hazards.

c. Assume there is **full forwarding**. Draw a pipeline execution diagram. Insert **nop** instructions to eliminate any hazards.

d. What is the total execution time of this instruction sequence **without forwarding** and **with full forwarding**? What is the speedup achieved by adding full forwarding to a pipeline that had no forwarding?

e. Assume there is **ALU-ALU forwarding** only (no forwarding from the MEM to the EX stage). Draw a pipeline execution diagram. Insert **nop** instructions to eliminate any hazards.

f. What is the total execution time of this instruction sequence **with only ALU-ALU** forwarding? What is the speedup over a no-forwarding pipeline?

3. In this exercise, we examine how resource hazards, control hazards, and Instruction Set Architecture (ISA) design can affect pipelined execution. Problems in this exercise refer to the following fragment of MIPS code:

```
sw   r16, 12(r6)
lw   r16, 8(r6)
beq  r5,  r4,  Label        # Assume r5 != r4
add  r5,  r1,  r4
slt  r5,  r15, r4
```

Unless stated, all problems are independent scenarios.
Also, assume registers can be read and written in the same cycle and the following cycle times for each of the options related to forwarding:

IF	ID	EX	MEM	WB
200ps	100ps	150ps	250ps	100ps

a. For this problem, assume that all branches are **perfectly predicted** (this eliminates all control hazards) and that **no delay slots are used**. Draw a pipeline execution diagram. What is the total execution time of the instruction sequence?

b. For this problem, assume that all branches are **perfectly predicted** (this eliminates all control hazards) and that **no delay slots are used**. If we change load/store instructions to use a register (without an offset) as the address, these instructions no longer need to use the ALU. As a result, every instruction will only utilize either MEM or EX stage, and the pipeline has only 4 stages. Draw a pipeline execution diagram to reflect this change. Assuming this change does not affect clock cycle time, what is the total execution time of the instruction sequence? What is the speedup of this change compared to part a?

c. For this problem, assume that all branches are **perfectly predicted** (this eliminates all control hazards) and that **no delay slots are used**. If we only have one memory (for both instructions and data), there is a structural hazard every time we need to fetch an instruction in the same cycle in which another instruction accesses data. To guarantee forward progress, this hazard must always be resolved in favor of the instruction that accesses data. Draw a pipeline execution diagram. What is the total execution time of this instruction sequence in the 5-stage pipeline that only has one memory?

d. We have seen that data hazards can be eliminated by adding **nops** to the code. Can you do the same with the structural hazard described in part c? Why?

e. Assuming **stall-on-branch and no delay slots** and branch outcomes are determined in the ID stage instead of EXE stage, draw a pipeline execution diagram. Insert **nops** if necessary. What is the total execution time of the instruction sequence?