Example Questions

Question 1

Consider the following instruction set characteristics

Instruction Type	CPI	Percentage
A	5	20%
B	3	30%
C	6	5%
D	4	20%
E	3	25%

a) What is the average CPI?

b) In order to improve the performance, you can change one type of instructions to have a CPI of 1, which one do you choose to maximize the improvement, and what is the new CPI?

c) What if changing the CPI of one type to 1 will increase the maximum CPI of the remaining instructions by 1, which one do you choose to set to 1?

Question 2

a) A programmer writes these 2 pieces of code for MIPS, what if any is wrong with the code?

Code A

```
addi  $t0, $0, 4

sw    $t0, 2($t0)

add   $t0, $t0, $sp

lw    $t0, 4($sp)
```

Code B

```
Loop:       sll   $t1, $s3, 2

            add   $t1, $t1, $s6

            lw    $t0, 0($t1)

            bne   $t0, $s5, Exit

            addi  $s3, $s3, 1

            j     Loop

Exit:
```

b) MIPS does not include a "NOP" instruction, which one of these instructions can act as a NOP (NOP is an instructions that does no operation at all, it just occupy a space in the pipeline.

```
a) addi  $0, $0, $0

b) mul   $t0, $0, $0

c) sw    $t0, 0($t0)

d) lw    $t0, 0($sp)
```

Question 3

Consider the following 16-bit floating-point number system in which we use 1 bit for the sign, 5 bits for the exponent, and 10 bits for the significand (mantissa). The exponent bias is equal to 16, and the largest and smallest biased exponents are reserved (similar to IEEE 754 Standard). The significand uses a hidden (implicit) 1. The value of a floating-point number can be calculated using the following expression:

What is the value of these numbers?
```
X = 0111000001100000

Y = 0101101000100001
```

What if there is no hidden one?

Question 4

Assume that one implementation of the original MIPS ISA can run at 1 GHz. We want to modify the MIPS ISA and architecture design. For the same given code and data, our new design increases the instruction count by 10% but reduces the average CPI by 20%. How fast (i.e., at what clock frequency) must our new design run in order to achieve a speedup of 4 compared with the original implementation? Show your work. (Hint: You don't need to solve the previous problems in order to solve this one.)

Question 5

Consider a program consisting of 100 lw instructions in which each instruction is dependent on the instruction immediately preceding it, e.g.,
lw $2,0($1)
lw $3,0($2)
lw $4,0($3)
...
What would the actual CPI be in the pipelined processor?

Question 6

Fill the table with F, D, X, M, or W or space if delayed. Assume a pipelened MIPS machine with **no forwarding**. Assume also that we can read and write any register in the same cycle by performing write in the first half, and read in the second half.

F   Fetch
D   Decode and operand fetch
X   Execute
M   memory
W   Write back

```
Loop:  lw     $t0,   0($s1)
       addu   $t0,   $t0,    $s2
       sw     $t0,   0($s1)
       addi   $s1,   $s1,    -4
       sw     $s1, 4($sp)
```

lw $t0,0($s1)													
addu $t0,$t0,$s2													
sw $t0, 0($s1)													
addi $s1, $t1, -4													
sw $t0, 4($sp)													

Problem 7

Write a complete Verilog module to do the following.
The inputs are
- 2 4-bit numbers A and B
- 2-bit function

The outputs are
- 1 4-bit number
- A carry bit if the output has a carry (needs more than 4 bits to represent it)
- A zero bit if the output is a zero