**COMPUTER ARCHITECTURE**

**ASSIGNMENT NO. 1, Fall 2019**

**Due Date: 28-10-2019**

**Q1:** What is the clock frequency of a processor that executes a program of 60,000 instructions in 4 microseconds? The average clock cycles per instruction of the processor is 1.5.

**Q2:** Calculate the CPI for a processor that executes a 30,000 instructions program in 18 microseconds, if the clock frequency is 6 GHz.

**Q3:** Calculate the CPI of a processor using the data available in the following table:

**Type of Instruction Frequency of Occurrence No. of clock cycles**

ALU 30% 2

Load/Store 25% 4

Control flow 18% 3

Others 27% 3

Calculate the CPI of a processor using the data available in the following (4)

**Type of Instruction Frequency of Occurrence No. of clock cycles**

ALU 40% 4

Load/Store 22% 3

Control flow 20% 3

Others 18% 4

**Q4:** A processor is enhanced by adding a graphics unit. A speedup of 15 is achieved for this enhanced mode. If the overall speedup is observed to be 2 for an application that runs on the enhanced processor, what is the percentage of graphics instructions in the example application?

**Q5:** Designers are given two alternatives to choose from while enhancing the performance of a processor: Enhance the entire ALU to get an average speedup of 5 in the ALU instructions or enhance ADD/SUB instructions to get a speedup of 10. ALU instructions constitute 40% of the total instruction and ADD/SUB instructions constitute 25% of the total. Evaluate both alternatives to select the better option.

**Q6:** A non-pipelined processor has 4 steps of execution, where the time taken is 30nsec, 25nsec, 35nsec and 40nsec for the respective steps. The processor is pipelined by adding pipeline latches between steps of execution. If the latch delay is 2nsec, what is the pipeline clock frequency? Calculate the speedup of the pipelined processor over its non- pipelined counterpart? Is it possible to achieve this speedup? Why or why not?

**Q7:** Identify all data hazards in the following code segment. Give solutions through internal forwarding in a RISC V processor. Show the space time diagram of the code after all hazards are removed/optimized.

**1. LD x1, 8(x2)**

**2. ADD x3, x1, x5**

**3. SUB x6, x1, x5**

**4. ADD x4, x3, x6**

**5. OR x5, x4, x7**

**6. SD x5, 8(x2)**

**Q8:** What is a delayed branch technique of handling control hazards? Give an example to illustrate using RISC V and the five stage integer pipeline.