**Tutorial 4**

**Computer-Architecture-II**

Q2

1) R1 = 2

R2 = 0

Clock cycles = 10

2) R1 = 2

R2 = 0

Clock cycles = 18

3) R1 = 2

R2 = 2

Clock cycles = 10

Explanation :

1) With an ALU forwarding enabled, an ALU instruction can move forward to the next instruction due to which there is no stall in the part the program. As a result the number of clock cycles is 10 and the answer in R1 is 2 and in R2 is 0 (which is the correct answer).

2) With ALU forwarding disabled but with CPU data dependency interlocks enabled, we get the same answers in R1 and R2. However the number of clock cycles is longer (n=18) due to stalls that were needed to have the right values for R1 and R2.

3) With both ALU forwarding and CPU data dependency interlocks disabled, we get a wrong answer in R1, with the number of clock cycles being equal to 10. This is because due to disabling the interlock, all the data dependencies were ignored and we got a bad answer in R1.

Q3

1) Clock cycles = 50

Instructions executed = 38

This is because there are three stalls of 4 clock cycles taking place. The first 4 are passed before the first instruction has finished executing. The second stall is due to branching and the third is due to shifting. (namely BEQZ and SLLi/SRLi instructions )

2) Clock cycles = 53

Instructions executed = 45

We need to put a NOP operation after we encounter a branch or a jump operation. On doing so we get the clock cycles and instructions executed given above.

This time the program takes longer to execute as due to branch prediction being disabled, the program will stall every time there is a branch.

3) Data dependency exists in the two shift instructions, namely SRLi and SLLi. On interchanging these instructions, the following clock cycles and instructions executed were obtained.

Clock cycles = 46

Instructions executed = 38

The clock cycles decrease as it is no longer needed to stall before shifting so we save up on time.