National University of Computer and Emerging Sciences, Lahore Campus

Course:	Advance Computer Architecture	Course Code:	EE502
Program:	MS(Computer Science)	Semester:	Spring 2017
Duration	30 mins	Total Marks:	25
Date:	09-03-2017	Weight	3.75
Section:	A	Page(s):	2
Exam:	Quiz 2		

Question1: Suppose we want to execute the following program on a computer with two execution units that have the following latencies. (10+10+5) marks

Adder 2 cycles
Multiplier 3 cycles

```
Loop:  LD R1, 40(R6)        // Instruction#1
       MUL  R5, R5, R1      // Instruction#2
       ST R1, 20(R5)        // Instruction#3
       ADD R6, R6, 2        // Instruction#4
       SUB R5, R5, 4        // Instruction#5
       BEQ R5, R0, Loop     // Instruction#6
```

You have to show the execution of two iterations of this loop. You can move an instruction from one stage of the processor to the other as soon as a slot is available in the next stage. **Instruction wait in the decode stage in case there is any dependency**. **Forwarding is implemented**. **After branch we need a cycle to avoid wrong fetching of next instruction**.

a. In-order issue and out-of-order execution rules are applied. For each cycle, show which instructions in at which stage by filling the following table. You don't need to write complete instructions. Just write the instruction numbers from the code above. Remember you cannot re-arrange the instructions for this part of the question.

Cycle No.	IF	ID	EX	MEM	WB
1					
2					
.					
.					
.					
.					
.					
.					
.					
.					
.					
.					
.					
.					

b. Now unroll the loop for level-2 (2 iterations only) and reschedule (rearrange) the instructions to get best execution time with the specifications described above.

c. Show the execution of this sequence of instruction for In-order issue and out-of-order execution and completion. Don't write whole instructions instead use the Instruction number assigned in part a.

Cycle No.	IF	ID	EX	MEM	WB
1					
2					
.					
.					
.					
.					
.					
.					
.					
.					
.					
.					
.					
.					