Question 1

Considering the MIPS64 architecture presented in the following:

- Integer ALU: 1 clock cycle
- Data memory: 1 clock cycle
- FP multiplier unit: pipelined 8 stages
- FP arithmetic unit: pipelined 2 stages
- FP divider unit: not pipelined unit that requires 8 clock cycles
- branch delay slot: 1 clock cycle
- data forwarding is enabled
- it is possible to complete instruction EXE stage in an out-of-order fashion.

and using the following code fragment

```
; ******************** MIPS64 **********************
;for (i = 1; i <= 100; i++){
;        v5[i] = v1[i]*v2[i];
;        v6[i] = v3[i]/v4[i];
}
```

1. Show the timing of the presented loop-based program and compute how many cycles does this program take to execute?
2. Using all optimization techniques, re-write the developed code in order to eliminate the most data hazards.
   a. Compute once again the number of clock cycles needed to execute the new program

	Clock cycles - Initial	Optimized code	Clock cycles - Optimized
V1- V2-V3-V4-V5-V6: .double "100 values"	—	V1- V2-V3-V4-V5-V6: .double "100 values"	—
main: daddui r1,r0,0	5	main: daddui r1,r0,0	5
daddui r2,r0,100	1	daddui r2,r0,50	1
loop: l.d f1,v1(r1)	1 → 2	loop: l.d f1,v1(r1)	1
l.d f2,v2(r1)	1	l.d f2,v2(r1)	1
mul.d f5,f1,f2	9	l.d f3,v3(r1)	1
s.d f5,v5(r1)	1	l.d f4,v4(r1)	1
l.d f3,v3(r1)	1	mul.d f5,f1,f2	8
l.d f4,v4(r1)	1	div.d f6,f3,f4	1
div.d f6,f3,f4	9	l.d f7,v1+8(r1)	0
s.d f6,v6(r1)	1	l.d f8,v2+8(r1)	0
daddui r1,r1,8	1	l.d f9,v3+8(r1)	0
daddi r2,r2,-1	1	mul.d f11,f7,f8	4
bnez r2,loop	2	l.d f10,v4+8(r1)	0
halt	0 → 1	daddi r2,r2,-1	0
—	—	s.d f5,v5(r1)	0
—	—	div.d f12,f9,f10	4
—	—	s.d f6,v6(r1)	0

—	—	s.d f11,v5+8(r1)	0
—	—	daddui r1,r1,16	0
—	—	bnez r2,loop	0
—	—	s.d f12,v6-8(r1)	1
—	—	halt	0
Total	**2906**		**1106**

Question 2

Considering the initial loop-based program, and assuming the following processor architecture for a superscalar MIPS64 processor implemented with multiple-issue and speculation:

- issue 2 instructions per clock cycle
- jump instructions require 1 issue
- handle 2 instructions commit per clock cycle
- timing facts for the following separate functional units:
  i. 1 Memory address 1 clock cycle
  ii. 1 Integer ALU 1 clock cycle
  iii. 1 Jump unit 1 clock cycle
  iv. 1 FP multiplier unit, which is pipelined: 8 stages
  v. 1 FP divider unit, which is not pipelined: 8 clock cycles
  vi. 1 FP Arithmetic unit, pipelined: 2 stages
- Branch prediction is always correct
- There are no cache misses
- There are 2 CDB (Common Data Bus).

- **Complete the table reported below showing the processor behavior for the 2 initial iterations of the loop.**

# iteration		Issue	EXE	MEM	CDB x2	COMMIT x2
1	l.d f1,v1(r1)	1	2	3	4	5
1	l.d f2,v2(r1)	1	3	4	5	6
1	mul.d f5,f1,f2	2	6	—	14	15
1	s.d f5,v5(r1)	2	4	—	—	15
1	l.d f3,v3(r1)	3	5	6	7	16
1	l.d f4,v4(r1)	3	6	7	8	16
1	div.d f6,f3,f4	4	9	—	17	18
1	s.d f6,v6(r1)	4	7	—	—	18
1	daddui r1,r1,8	5	6	—	7	19
1	daddi r2,r2,-1	5	7	—	8	19
1	bnez r2,loop	6	9	—	—	20
2	l.d f1,v1(r1)	7	8	9	10	20
2	l.d f2,v2(r1)	7	9	10	11	21
2	mul.d f5,f1,f2	8	12	—	20	21
2	s.d f5,v5(r1)	8	10	—	—	22
2	l.d f3,v3(r1)	9	11	12	13	22
2	l.d f4,v4(r1)	9	12	13	14	23
2	div.d f6,f3,f4	10	17	—	25	26
2	s.d f6,v6(r1)	10	13	—	—	26
2	daddui r1,r1,8	11	12	—	13	27
2	daddi r2,r2,-1	11	13	—	15	27
2	bnez r2,loop	12	16	—	—	28