Computer Architectures -- example

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, and the branch delay slot is not enable
- forwarding is enabled
- it is possible to complete instruction EXE stage in an out-of-order fashion.

o and using the following code fragment, show the timing of the presented loop-based program and compute how many cycles does this program take to execute?

$ \begin{array}{llllllllllllllllllllllllllllllllllll$	con	npute how many cycles ********	does this program take to execut ** WinMIPS64 ******	e'? :*******
$ \begin{array}{c} \text{Commenti} \\ \text{Colpi di clock} \\ \text{V1: .double "100 valori"} \\ \text{V2: .double "100 valori"} \\ \text{V7: .double "100 valori"} \\ \text{V7: .double "100 zeri"} \\ \text{.text} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\$; for (i	= 0; i < 100; i++)		
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$		v7[i] = (v1)	[i]+v2[i])/v3[i] + (v4[i]*v5[i])/v6[i];
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$;}			
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	•		Commenti	Colpi di clock
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$.data		
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	V1: .de	ouble "100 valori"		
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	V2: .de	ouble "100 valori"		
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$				
main: daddui r1,r0,0 daddui r2,r0,100 loop: l.d f1,v1(r1) l.d f2,v2(r1) add.d f7,f1,f2 l.d f3,v3(r1) div.d f8,f7,f3 l.d f4,v4(r1) l.d f5,v5(r1) mul.d f9,f4,f5 l.d f6,v6(r1) div.d f11,f8,f10 s.d f11,v7(r1) daddui r1,r1,8 daddi r2,r2,-1 bnez r2,loop Halt	V6: .de	ouble "100 valori"		
$\begin{array}{llllllllllllllllllllllllllllllllllll$	V7: .de	ouble "100 zeri"		
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$.text		
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$				
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	main:	daddui r1.r0.0	r1 <= puntatore	5
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$			-	1
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	loop:	, ,	f1 <= v1[i]	1
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	1	, , ,		1
$\begin{array}{cccccccccccccccccccccccccccccccccccc$			E 3	2
$\begin{array}{lllll} & \text{div.d } f8, f7, f3 \\ & \text{l.d } f4, v4(r1) \\ & \text{l.d } f5, v5(r1) \\ & \text{mul.d } f9, f4, f5 \\ & \text{l.d } f6, v6(r1) \\ & \text{div.d } f10, f9, f6 \\ & \text{add.d } f11, f8, f10 \\ & \text{s.d } f11, v7(r1) \\ & \text{daddui } r1, r1, 8 \\ & \text{daddi } r2, r2, -1 \\ & \text{bnez } r2, loop \\ & \text{Halt} \\ \end{array}$		1.d f3,v3(r1)		1
$\begin{array}{cccccccccccccccccccccccccccccccccccc$		/	$f8 \le (v1[i]+v2[i])/v3[i]$	9
$\begin{array}{cccccccccccccccccccccccccccccccccccc$		1.d f4,v4(r1)	`	0
$\begin{array}{cccccccccccccccccccccccccccccccccccc$		1.d f5,v5(r1)	f5 <= v5[i]	0
$\begin{array}{cccccccccccccccccccccccccccccccccccc$			f9 <= v4[i]*v5[i]	4
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$		1.d f6,v6(r1)	f6 <= v6[i]	0
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$		div.d f10,f9,f6	$f10 \le (v4[i]*v5[i])/v6[i]$	8
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$		add.d f11,f8,f10	f11 <= f8+f10	1
daddi r2,r2,-1 $r2 \le r2 - 1$ 1 bnez r2,loop Halt		s.d f11,v7(r1)		1
bnez r2,loop Halt		daddui r1,r1,8	$r1 \le r1 + 8$	1
Halt		daddi r2,r2,-1	r2 <= r2 - 1	1
		bnez r2,loop		2 + 1
Total 3306		Halt		
		Total		3306

Computer Architectures -- example

Question 2

Considering the same 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, which is 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.

# iteration		Issue	EXE	MEM	CDB x2	COMMIT x2
1	l.d f1,v1(r1)	1	2m	3	4	5
1	l.d f2,v2(r1)	1	3m	4	5	6
1	add.d f7,f1,f2	2	6f		8	9
1	l.d f3,v3(r1)	2	4m	5	6	9
1	div.d f8,f7,f3	3	9d		17	18
1	l.d f4,v4(r1)	3	5m	6	7	18
1	l.d f5,v5(r1)	4	6m	7	8	19
1	mul.d f9,f4,f5	4	9x		17	19
1	l.d f6,v6(r1)	5	7m	8	9	20
1	div.d f10,f9,f6	5	25d		33	34
1	add.d f11,f8,f10	6	34		36	37
1	s.d fl1,v7(r1)	6	8m			37
1	daddui r1,r1,8	7	8i		9	38
1	daddi r2,r2,-1	7	9i		10	38
1	bnez r2,loop	8	11j			39
2	l.d f1,v1(r1)	9	10m	11	12	39
2	l.d f2,v2(r1)	9	11m	12	13	40
2	add.d f7,f1,f2	10	14f		16	40
2	l.d f3,v3(r1)	10	12m	13	14	41
2	div.d f8,f7,f3	11	17d		25	41
2	l.d f4,v4(r1)	11	13m	14	15	42
2	l.d f5,v5(r1)	12	14m	16	17	42
2	mul.d f9,f4,f5	12	17M		25	43
2	l.d f6,v6(r1)	13	15m	16	18	43
2	div.d f10,f9,f6	13	33d		41	44
2	add.d f11,f8,f10	14	42f		44	45
2	s.d f11,v7(r1)	14	16m			45
2	daddui r1,r1,8	15	16i		17	46
2	daddi r2,r2,-1	15	17i		18	46
2	bnez r2,loop	16	19j			47