2 March 2012 -- Computer Architectures –- part 2/2

Surname, Name, Matricola

.....

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 4 stages
- FP divider unit: not pipelined unit that requires 10 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?

```
; ********************* MIPS64 ***********************
;     for (i = 0; i < 100; i++) {
;          v5[i] = v1[i]/v2[i];
;          v6[i] = v2[i]*v3[i];
;          v7[i] = v6[i]/v4[i] + v3[i];
;     }
```

		comments	Clock cycles
	.data		
V1:	.double “100 values”		
…			
V4:	.double “100 values”		
V5:	.double “100 zeros”		
…			
V7:	.double “100 zeros”		
	.text		
main:	daddui r1,r0,0	r1← pointer	
	daddui r2,r0,100	r2 <= 100	
loop:	l.d f1,v1(r1)	f1 <= v1[i]	
	l.d f2,v2(r1)	f2 <= v2[i]	
	div.d f5,f1,f2	f5 ← v1[i]/v2[i]	
	s.d f5,v5(r1)	v5[i] ← f5	
	l.d f3,v3(r1)	f3 <= v3[i]	
	mul.d f6,f2,f3	f6 ← v2[i]*v3[i]	
	l.d f4,v4(r1)	f4 <= v4[i]	
	div.d f7,f6,f4	f7 ← v6[i]/v4[i]	
	add.d f7,f7,f3	f7 ← v6[i]/v4[i]+v3[i]	
	s.d f6,v6(r1)	v6[i] ← f6	
	s.d f7,v7(r1)	v7[i] ← f7	
	daddi r2,r2,-1	r2 ← r2 - 1	
	daddui r1,r1,8	r1 ← r1 + 8	
	bnez r2,loop		
	Halt		
	Total		

2 March 2012 -- Computer Architectures –- part 2/2

Surname, Name, Matricola

.....

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: 10 clock cycles
  vi. 1 FP Arithmetic unit, which is pipelined: 4 stages
- Branch prediction is always correct
- There are no cache misses
- There are 2 CDB (Common Data Bus).

o Complete the table reported below showing the processor behavior for the 2 initial iterations.
o

# 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	div.d f5,f1,f2	2	6		16	17
1	s.d f5,v5(r1)	2	4			17
1	l.d f3,v3(r1)	3	5	6	7	18
1	mul.d f6,f2,f3	3	8		16	18
1	l.d f4,v4(r1)	4	6	7	8	19
1	div.d f7,f6,f4	4	26		36	37
1	add.d f7,f7,f3	5	37		41	42
1	s.d f6,v6(r1)	5	7			42
1	s.d f7,v7(r1)	6	8			43
1	daddi r2,r2,-1	6	7		8	43
1	daddui r1,r1,8	7	8		9	44
1	bnez r2,loop	8	9			44
2	l.d f1,v1(r1)	9	10	11	12	45
2	l.d f2,v2(r1)	9	11	12	13	45
2	div.d f5,f1,f2	10	16		26	46
2	s.d f5,v5(r1)	10	12			46
2	l.d f3,v3(r1)	11	13	14	15	47
2	mul.d f6,f2,f3	11	16		24	47
2	l.d f4,v4(r1)	12	14	15	16	48
2	div.d f7,f6,f4	12	36		46	48
2	add.d f7,f7,f3	13	47		51	52
2	s.d f6,v6(r1)	13	15			52
2	s.d f7,v7(r1)	14	16			53
2	daddi r2,r2,-1	14	15		17	53
2	daddui r1,r1,8	15	17		18	54
2	bnez r2,loop	16	18			54

4 5 6 8 9 12 13 15 16 17 26 31 36 46 51

8 15 16