Nome, Cognome, Matricola .....

Domanda 10 (8 punti)

Si consideri il seguente frammento di codice C:

```
for (i = 0; i < 100; i++) {
     v7[i] = (v1[i]*v2[i]);
     v8[i] = (v3[i]*v4[i]);
     v9[i] = (v5[i]*v6[i]);
}
```

dove i vettori `v1[]`… `v6[]` contengono numeri Floating Point (FP), sono lunghi 100 e sono stati salvati in precedenza nella memoria. Inoltre sono stati allocati in memoria i vettori vuoti `v7[]`, `v8[]` e `v9[]`.

Si eseguano le seguenti operazioni:

1) Con riferimento al programma riportato nel seguito, scritto per l'architettura del processore MIPS64 (descritta sotto), ed utilizzando gli spazi a ciò appositamente destinati, si calcoli il numero di colpi di clock richiesti per l'esecuzione dell'intero programma. L'architettura da considerare ha le seguenti caratteristiche:
   - L'unità di moltiplicazione FP è un'unità pipelined a 10 stadi
   - L'unità aritmetica FP è un'unità pipelined a 2 stadi
   - L'unità di divisione FP è un'unico blocco con una latenza pari a 10 colpi di clock
   - Il branch delay slot è pari ad 1 colpo di clock
   - Il delay slot non è abilitato (ossia, la pipeline viene svuotata se il salto viene preso)
   - Il data forwarding è abilitato.

   Si assuma che le diverse unità funzionali che compongono lo stadio di EX possano lavorare in parallelo su istruzioni diverse: l'architettura considerata può quindi implementare un meccanismo che permette il completamento out-of-order delle istruzioni.
2) Si ottimizzi il programma utilizzando la tecnica di *scheduling statico*, *register renaming* e abilitando il *branch delay slot* in maniera tale che il programma esegua lo stesso calcolo; infine, si esegua lo stesso calcolo dei colpi di clock del punto 1 con il nuovo programma evidenziando il miglioramento delle prestazioni.
3) Con riferimento all'architettura di un processore MIPS che implementa la strategia multiple-issue con speculazione (descritta sotto), si calcoli il numero di colpi di clock necessari all'esecuzione di 2 cicli del programma proposto. L'architettura da considerare ha le seguenti caratteristiche:
   - può eseguire l'issue di 2 istruzioni per colpo di clock
   - in presenza di un'istruzione di salto, viene eseguita una sola issue
   - può eseguire il commit di 2 istruzioni per colpo di clock
   - sono disponibili le seguenti unità funzionali indipendenti:
     i. 1 unità Memory address
     ii. 1 unità per operazioni intere (ALU)
     iii. 1 unità per il calcolo dei salti
     iv. 1 unità di moltiplicazione FP pipelined (latenza 10)
     v. 1 unità di divisione FP no pipelined (latenza 10)
     vi. 1 unità di somma e sottrazione FP pipelined (latenza 2)
   - la previsione sui salti è sempre corretta
   - le cache non produce mai situazioni di miss
   - sono disponibili due CDB (Common Data Bus).

Punto 1): calcolo durata programma originario

; ********** WinMIPS64 **********
;
;

	Commenti	Colpi di clock
.data		
V1: .double “100 valori”		
…		
V6: .double “100 valori”		
V7: .double “100 zeri”		
V8: .double “100 zeri”		
V9: .double “100 zeri”		
.text		
main: daddui r1,r0,0	r1 <= puntatore	5
daddui r2,r0,100	r2 <= 100	1
loop: l.d f1,v1(r1)	f1 <= v1[i]	1
l.d f2,v2(r1)	f2 <= v2[i]	1
mul.d f7,f1,f2	f7 <= v1[i]*v2[i]	11
s.d f7,v7(r1)	f7 => v7[i]	1
l.d f3,v3(r1)	f3 <= v3[i]	1
l.d f4,v4(r1)	f4 <= v4[i]	1
mul.d f7,f3,f4	f7 <= v3[i]*v4[i]	11
s.d f7,v8(r1)	f7 => v8[i]	1
l.d f5,v5(r1)	f5 <= v5[i]	1
l.d f6,v6(r1)	f6 <= v6[i]	1
mul.d f7,f5,f6	f7 <= v5[i]*v6[i]	11
s.d f7,v9(r1)	f7 => v9[i]	1
daddui r1,r1,8	r1 <= r1 + 8	1
daddi r2,r2,-1	r2 <= r2 - 1	1
bnez r2,loop		2
halt		1
total		

Punto 2): ottimizzazione del programma

Punto 3): calcolo durata su architettura superscalare

# iterazione		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 f7,f1,f2	2	6		16	17
1	s.d f7,v7(r1)	2	4			17
1	l.d f3,v3(r1)	3	5	6	7	18
1	l.d f4,v4(r1)	3	6	7	8	18
1	mul.d f7,f3,f4	4	9		19	20
1	s.d f7,v8(r1)	4	7			20
1	l.d f5,v5(r1)	5	8	9	10	21
1	l.d f6,v6(r1)	5	9	10	11	21
1	mul.d f7,f5,f6	6	12		22	23
1	s.d f7,v9(r1)	6	10			23
1	daddui r1,r1,8	7	8		9	24
1	daddi r2,r2,-1	7	9		10	24
1	bnez r2,loop	8	11			25
2	l.d f1,v1(r1)	9	11	12	13	25
2	l.d f2,v2(r1)	9	12	13	14	26
2	mul.d f7,f1,f2	10	15		25	26
2	s.d f7,v7(r1)	10	13			27
2	l.d f3,v3(r1)	11	14	15	16	27
2	l.d f4,v4(r1)	11	15	16	17	28
2	mul.d f7,f3,f4	12	18		28	28
2	s.d f7,v8(r1)	12	16			29
2	l.d f5,v5(r1)	13	17	18	19	29
2	l.d f6,v6(r1)	13	18	19	20	30
2	mul.d f7,f5,f6	14	21		31	32
2	s.d f7,v9(r1)	14	19			32
2	daddui r1,r1,8	15	16		17	33
2	daddi r2,r2,-1	15	17		18	33
2	bnez r2,loop	16	19			34

I primi 2 cicli sono eseguiti in _____ colpi di clock.