Nome, MATRICOLA .....

Considerando il processore MIPS64 e l'architettura descritta in seguito:

- Integer ALU: 1 clock cycle
- FP arithmetic unit: pipelined 3 stages
- Data memory: 1 clock cycle
- FP divider unit: not pipelined unit that requires 7 clock cycles
- FP multiplier unit: pipelined 7 stages branch delay slot: 1 clock cycle, and the branch delay slot disabled
- forwarding enabled
- it is possible to complete instruction EXE stage in an out-of-order fashion.

Usando il frammento di codice riportato, si calcoli il tempo di esecuzione dell'intero programma in colpi di clock e si completi la seguente tabella.

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

	<i>,</i> )																																									lock
	.data																																								C)	ycles
V1:	.double "100 values"																		Ш										$\perp$													
V2:	.double "100 values"																																									
V3:	.double "100 values"																																									
V4:	.double "100 values"																																									
V5:	.double "100 values"																																									
V6:	.double "100 values"																																									
	.text																																									
main:	daddui r1,r0,0	F				W																																				5
	daddui r2,r0,100		F	D	E	М	W																																			1
loop:	l.d f1,v1(r1)			F				W																																		1
	I.d f2,v2(r1)				F	D	Е	М	W																																	1
	div.d f5,f1,f2					F	D	S	/	/	/	/	/	/	/	М	W																									8
	s.d f5,v5(r1)						F	S	- 1					S	- 1	- 1	- 1																									1
	I.d f3,v3(r1)								F	S	S	S	S	S	S																											1
	I.d f4,v4(r1)															F	D	Е	М	W																						1
	mul.d f6,f1,f2																F	D		*			*		*	М	W															7
	div.d f7,f6,f3																	F						S			/ /	/ /	/	/	1 1	М	- 1									7
	add.d f7,f7,f4																		F	S	S	S	S	S	S						S			М								3
	s.d f7,v6(r1)																									F	S	S	S	S	S				- 1	- 1						1
	daddi r2,r2,-1																															F	S									1
	daddui r1,r1,8																																	F	D	Е	М	W				1
	bnez r2,loop																																		F	D			W			1
	Halt																																			F	N	N	N	N		1
	Total																						(	3+(3	5)*	100															;	3506

Nome, MATRICOLA .....

Considerando il processore MIPS64 e l'architettura descritta in seguito:

- Integer ALU: 1 clock cycle
- FP arithmetic unit: pipelined 3 stages
- Data memory: 1 clock cycle
- FP divider unit: not pipelined unit that requires 7 clock cycles
   branch delay slot: 1 clock cycle, and the branch delay slot disable
- FP multiplier unit: pipelined 7 stages branch delay slot: 1 clock cycle, and the branch delay slot disabled
- forwarding enabled
- it is possible to complete instruction EXE stage in an out-of-order fashion.

Usando il frammento di codice riportato, si calcoli il tempo di esecuzione dell'intero programma in colpi di clock e si completi la seguente tabella.

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

, ,																																				Clock
.data																																				cycles
V1: .double "100 values	,	$\top$		т	$\top$																$\neg$										Т	П		$\top$		
V2: .double "100 values																																				
V3: .double "100 values	,			T														П														П			П	
V4: .double "100 values	,																																			
V5: .double "100 values"	"			T			Т			Т																					П	П				
V6: .double "100 values"	,																																			
.text																																				
main: daddui r1,r0,0					М																															5
daddui r2,r0,100			F																																	1
loop: I.d f1,v1(r1)			F			E																														1
I.d f2,v2(r1)					F			M V																												1 (1)
l.d f3,v3(r1)						F	D	ΕN	Λ	/																										1 (1)
div.d f5,f1,f2								D /	/	/	/	/	/	/	М	W																				7 (8)
mul.d f6,f1,f2								FC		*			*	*	*	М	W																			1 (7)
I.d f4,v4(r1)								F	D	E		W	l 1																							0 (1)
daddi r2,r2,-1									F	D			W																							0 (1)
daddui r1,r1,8										F	1	1	М	- 1																						0 (1)
div.d f7,f6,f3											F	-	S			/			/	/	/	М	W													<mark>6 (7)</mark>
s.d f5,v5(r1)												F	S	S				1 1																		0 (1)
add.d f7,f7,f4															F				S					+		W										3 (3)
s.d f7,v6(r1)																F	S	S	S	S		- 1	S				W									1 (1)
bnez r2,loop																						F	S	S	D											1 (1)
Halt																									F	N	N	N	N							1 (1)
То	tal																					(	3 <b>+</b> (2	23)*	100	)										2306

Nome, MATRICOLA .....

SPEEDUPenhanced = (8+7+1+1+1+7+1)/(7+1+6) = 1.86

FRACTIONenhanced =  $100 \cdot (8+7+1+1+1+7+1) / 3506 = 0.74$ 

SPEEDUPcomputation = 1 / ( (1 – FRACTIONenhanced) + FRACTIONenhanced /SPEEDUPenhanced ) = 1.52

SPEEDUPobservation =  $3506 / 2306 \approx 1.52 \implies \checkmark$ 

Nome, MATRICOLA .....

#### Domanda 2

Si consideri una BHT di 1K elementi con contatori di saturazione a 2 bit. Usando il frammento di codice proposto, si calcoli lo stato finale della BHT al completamento dell'esecuzione del programma proposto. Si calcoli inoltre, la *misprediction rate* complessiva. Lo stato iniziale della BPU è indicato nella tabella.

#### Assunzioni generali:

- R10 è il registro di controllo del loop ed è inizializzato a 100
- R3 e R7 sono registro di riferimento con valore 10
- R2 e R6 sono i registri in ingresso
  - o R2 riceve un valore sempre maggiore a 10
  - o R6 riceve un valore sempre minore a 10
- Gli altri registri sono inizializzati a 0.

Address	Inst	ruction	BHT (2-bit)	Prediction	misP. counter
0x0000	L0:				
	;	Reading input values			
0x0010		SLT R1, R2, R3	2	Т	
0x0014		BEQZ R1, L1	3	Т	0
0x0018		DADDI R11, R0, 10	2	Т	
0x001C	L1:	SLT R4, R6, R7	2	Т	
0x0020		BEQZ R4, L2	2-1-0	Т	1
0x0024		DADDI R12, R0, 10	2	Т	
0x0028	L2:	SLT R5, R2, R6	2	Т	
0x002C		BEQZ R5, L3	3	Т	0
0x0030		DADDI R13, R0, 10	2	Т	
0x0038	L3:	DADD R14, R13, R12	2	Т	
0x003C		DADD R14, R14, R11	2	Т	
0x0040		SLT R5, R6, R14	2	Т	
0x0044		BEQZ R5, L4	2-1-0	Т	1
0x0048		DADDI R15, R0, 10	2	Т	
0x004C	L4:		2	Т	
0x0050		DADDI R10, R10, #-1	2	Т	
0x0054		BNEZ R10, L0	2	Т	1
0x0058					

Note:

SLT R1,R2,R3 ; IF (R2 < R3) R1 
$$\leftarrow$$
 1 ; ELSE R1  $\leftarrow$  0