CS 570 HW 1 Zhuoling Chen 933341319

1.

a) When $P = 1$, no communication is required

Therefore $T_1 = T_c * N$

one time iterate

$$T_p = \frac{N}{P} \cdot (T_c + P \cdot T_b)$$

$$\text{Speedup} = \frac{T_1}{T_p} = \frac{N \cdot T_c}{\frac{N}{P} \cdot T_c + N \cdot T_b} = \frac{P \cdot R}{P + R}$$

b) $\text{Speedup} = \frac{R}{1 + \frac{R}{P}}$ When $P \to \infty$, Speedup $= R$ which is the max limitation of speedup.

c) $\frac{P \cdot R}{P + R} > 1$

$PR > P + R$

$1 + \frac{R}{R-1} > P$

d) $1024 T_c = \frac{N}{P} (T_c + P T_b)$

$$N = \frac{1024 T_c \cdot P}{(T_c + P T_b)}$$

$$= \frac{1024 R T_b \cdot P}{R T_b + P T_b}$$

$$N = \frac{\cancel{T_b}\, 1024 R P}{\cancel{T_b}(R + P)} = \frac{1024 R P}{R + P}$$

2.

a)

$$S_{1\,Base+FPunit} = \frac{1s+10ms+10s}{1s+2ms+6s} = \frac{11.01}{7.02} = 1.57$$

$$S_{1\,Base+cache} = \frac{1s+10ms+10s}{0.7s+9ms+8s} = \frac{11.01}{8.709} = 1.26$$

$$\overline{S}_{2\,Base+FPunit} = \frac{1}{3} \times (1+5+5/3) = 2.56$$

$$\overline{S}_{2\,Base+cache} = \frac{1}{3} \times (10/7 + 10/9 + 5/4) = 1.26$$

$$\overline{S}_{3\,Base+FPunit} = 3/(1+2/10+3/5) = 1.67$$

$$\overline{S}_{3\,Base+cache} = 3/(7/10+9/10+4/5) = 1.25$$

$$\overline{S}_{4\,Base+FPunit} = \sqrt[3]{1 \times 5 \times 5/3} = 2.03$$

$$\overline{S}_{4\,Base+cache} = \sqrt[3]{10/7 \cdot 10/9 \cdot 5/4} = 1.26$$

Compare all of them, the Base+FP is the better one

b)

$$S_{1\,Base+FPunit} = \frac{1+1+1}{1+0.2+0.6} = 1.67$$

$$S_{1\,Base+cache} = \frac{1+1+1}{0.7+0.9+0.8} = 1.26$$

$$\overline{S}_{2\,Base+FPunit} = \frac{1}{3} \times (1+5+5/3) = 2.56$$

$\overline{S}_{2\,Base+cache} = \frac{1}{3} \times (10/7 + 10/9 + 5/4) = 1.26$

$\overline{S}_{3\,Base+FPUnit} = 1.67$

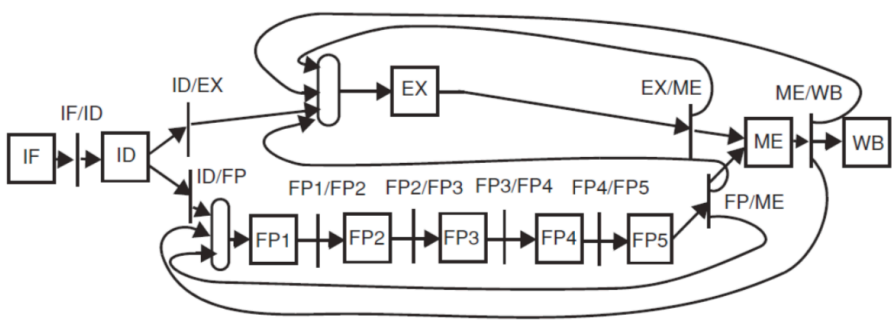
$\overline{S}_{3\,Base+cache} = 1.25$

$\overline{S}_{4\,Base+FPUnit} = 2.03$

$\overline{S}_{4\,Base+cache} = 1.26$

Compare all of them, the Base+FP is the better one

3.



Only in instruction "load" & "arithmetic" need to use write reg. Therefore. should check the instruction which is before the load & arithmetic.

RAW ① If integer / arithmetic / logic / store instructions in ID check ID/EX

② All load instruction in ID, check ID/EX

③ if FP store instruction in ID, check ID/EX ID/FP, FP_1/FP_2 FP_2/FP_3, FP_3/FP_4

④ if FP arithmetic instruction in ID, check ID/EX ID/FP, FP_1/FP_2 FP_2/FP_3, FP_3/FP_4

example:

Add	IF	ID	FP_1	FP_2	FP_3	FP_4	FP_5	M	W	
Add		IF	O	O	O	O	ID	FP_1		

⑤ if FP load instruction in ID, check ID/EX.

WAW

① if integer arithmetic/logic/store/load instruction in ID, check nothing.
It will not have this problem in five stage

② if FP store instruction in ID, check nothing.
Cuz store do not have write stage.

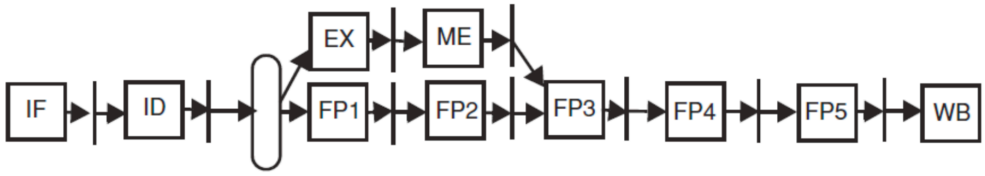
③ if FP arithmetic instruction in ID, check nothing. Same length.

④ if FP load instruction in ID, check the condition. if { load / Add } it will not cause hazards, if { Add / load } check ID/EX

example:

	IF	ID	FP_1	FP_2	FP_3	FP_4	FP_5	M	W		
	IF	O	O	O	O	ID	EX	M	W		ID/FP
											FP_1/FP_2, FP_2/FP_3, FP_3/FP_4

4.



a)

EX/ME → EX FP5/WB → EX

ME/FP3 → EX ME/FP3 → FP1

FP3/FP4 → FP1 FP3/FP4 → EX

FP4/FP5 → FP1 FP4/FP5 → EX

FP5/WB → FP1

b)

RAW ① If integer/arithmetic/logic/store instructions in ID check ID/EX

② if FP store instruction in ID, check ID/EX ID/FP, FP1/FP2 FP2/FP3, FP3/FP4

③ if FP arithmetic instruction in ID, check ID/EX ID/FP, FP1/FP2 FP2/FP3, FP3/FP4

c) When the pipeline executed, the data passing through the MEM, which means they are already in the memory. Once you want to rolling back, it is difficult.

5.

a)

Table 3 Tomasulo algorithm – no speculation

		Dispatch	Issue	Exec Start	Exec complete	Cache	CDB	Comments
I1	L.D F0,0(R1)	1	2	(3)	3	(4)	(5)	
I2	L.D F2,0(R2)	2	3	(4)	4	(5)	(6)	
I3	L.D F4,0(R3)	3	4	(5)	5	(6)	(7)	
I4	MUL.D F6,F2,F0	4	7	(8)	12	/	(13)	wait F2
I5	ADD.D F8,F6,F4	5	14	(15)	19	/	(20)	wait F6
I6	ADDI R1,R1,#8	6	7	(8)	8	/	(9)	
I7	ADDI R2,R2,#8	7	8	(9)	9	/	(10)	
I8	ADDI R3,R3,#8	8	9	(10)	10	/	(11)	
I9	S.D – A F8, -8(R2)	9	11	(12)	12	/	/	wait R2
I10	S.D – D F8, -8(R2)	10	21	(22)	22	(23)	/	wait F8
I11	BNE R4,R2,LOOP/	11	12	(13)	13	/	(15)	
I12	L.D F0,0(R1)	15	16	(17)	17	(18)	(19)	wait BNE

b)

Table 4 Speculative Tomasulo algorithm

		Dispatch	Isuue	Exec Start	Exec complete	Cache	CDB	Retire	Comments
I1	L.D F0,0(R1)	1	2	(3)	3	(4)	(5)	6	
I2	L.D F2,0(R2)	2	3	(4)	4	(5)	(6)	7	
I3	L.D F4,0(R3)	3	4	(5)	5	(6)	(7)	8	
I4	MUL.D F6,F2,F0	4	7	(8)	12	/	(13)	14	wait F2
I5	ADD.D F8,F6,F4	5	14	(15)	19	/	(20)	21	wait F6
I6	ADDI R1,R1,#8	6	7	(8)	8	/	(9)	22	
I7	ADDI R2,R2,#8	7	8	(9)	9	/	(10)	23	
I8	ADDI R3,R3,#8	8	9	(10)	10	/	(11)	24	
I9	S.D – A F8, -8(R2)	9	11	(12)	12	/	/	/	wait R2
I10	S.D – D F8, -8(R2)	10	21	(22)	22	(23)	/	25	wait F8
I11	BNE R4,R2,LOOP/	11	12	(13)	13	/	(14)	26	wait R2
I12	L.D F0,0(R1)	12	13	(14)	14	(15)	(16)	27	wait BNE