High Performance Computer Architecture (CS60003)

Tutorial 3 Solutions

Indian Institute of Technology Kharagpur

Probable Solutions

1.

Instructions	Issue	Execute	Write	Commit
DIV R2 R3 R4	1	2	6	7
MUL R1 R5 R6	7	8	10	11
ADD R3 R7 R8	8	9	10	12
MUL R1 R1 R2	11	12	14	15
SUB R4 R2 R5	12	13	15	16
ADD R1 R4 R3	16	17	18	19

2. Red ones denote flushed stages after error is thrown at 9^{th} cycle

Instructions	Issue	Execute	Write	Commit
ADD R2 R2 R1	1	2	3	4
SUB R1 R1 R3	2	3	4	5
ADD R3 R4 R5	3	4	5	6
DIV R3 R2 R3	4	6		
ADD R1 R4 R4	5	6	7	
ADD R3 R3 R2	6			

3. (a) i. The states of the reservation stations at each cycle are shown below:

Cycle 1

	ADD	
RS	**Source 1**	**Source 2**
RS0	10	11
RS1	–	–
RS2	–	–

	MUL	
RS	**Source 1**	**Source 2**
RS3	–	–
RS4	–	–
RS5	–	–

Cycle 2

	ADD	
RS	**Source 1**	**Source 2**
RS0	10	11
RS1	10	RS0
RS2	–	–

	MUL	
RS	**Source 1**	**Source 2**
RS3	–	–
RS4	–	–
RS5	–	–

Cycle 6: No change in Reservation Station(s).

Cycle 3

	ADD	
RS	**Source 1**	**Source 2**
RS0	10	11
RS1	10	RS0
RS2	–	–

	MUL	
RS	**Source 1**	**Source 2**
RS3	RS1	10
RS4	–	–
RS5	–	–

Cycle 4

	ADD	
RS	**Source 1**	**Source 2**
RS0	–	–
RS1	10	21
RS2	–	–

	MUL	
RS	**Source 1**	**Source 2**
RS3	RS1	10
RS4	10	10
RS5	–	–

Cycle 5

	ADD	
RS	**Source 1**	**Source 2**
RS0	RS3	RS4
RS1	10	21
RS2	–	–

	MUL	
RS	**Source 1**	**Source 2**
RS3	RS1	10
RS4	10	10
RS5	–	–

Cycle 7

	ADD	
RS	**Source 1**	**Source 2**
RS0	RS3	RS4
RS1	–	–
RS2	–	–

	MUL	
RS	**Source 1**	**Source 2**
RS3	31	10
RS4	10	10
RS5	–	–

Cycle 8 and Cycle 9: No change in Reservation Station(s).

Cycle 10

	ADD	
RS	**Source 1**	**Source 2**
RS0	RS3	100
RS1	–	–
RS2	–	–

	MUL	
RS	**Source 1**	**Source 2**
RS3	31	10
RS4	–	–
RS5	–	–

ii. Register Alias Table at the end of cycle 10 is shown below.

(b) i. Contents of the Register File and the Reorder Buffer at each cycle are shown below.

Register Alias Table (RAT)	
R0	RS3
R1	
R2	~~RS4~~ RS0
R3	~~RS1~~ 31
R4	
R5	
R6	
R7	~~RS0~~ 21

Cycle 1

Register File			
R0	R1	R2	R3
5	4	3	2

Reorder Buffer		
Result Register	Result Value	Done
R0	—	0
—	—	—

Cycle 2

Register File			
R0	R1	R2	R3
5	4	3	2

Reorder Buffer		
Result Register	Result Value	Done
R0	—	0
R1	—	0

Cycle 3

Register File			
R0	R1	R2	R3
5	4	3	2

Reorder Buffer		
Result Register	Result Value	Done
R0	—	0
R1	5	1

Cycle 4

Register File			
R0	R1	R2	R3
5	4	3	2

Reorder Buffer		
Result Register	Result Value	Done
R0	1	1
R1	5	1

Cycle 5

Register File			
R0	R1	R2	R3
5	4	3	2

Reorder Buffer		
Result Register	Result Value	Done
–	–	–
R1	5	1

ii. The contents will be the same as the initial state. This is bad because the programmer will not know if the `LD` executed and stored its value in architectural register `R0`, making programs harder to debug.

Cycle 6

Register File			
R0	R1	R2	R3
1	5	3	2

Reorder Buffer		
Result Register	Result Value	Done
–	–	–
–	–	–

4. (a) 8×2
   (b) $16 \times 8 \times 2 \times 8 + 8 \times 32$
   (c) $\log(16 \times 8) = 7$
   (d) 72×32 (64 bits for data, 7 bits for tag, 1 for valid bit)
   (e) $7 \times 32 + 7 \times 16 \times 8 \times 2$

5. (a) The 5 instructions in order :

```
ADD r0 <- r2, r1
MUL r3 <- r1, r0
ADD r3 <- r5, r2
MUL r4 <- r3, r1
ADD r3 <- r0, r5
```

```
       1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10| 11| 12| 13| 14
ADD    F | D | E | W | W
MUL        F | D | E | E | E | W | W
ADD            F | D | E | W | W
MUL                F | D | E | E | E | W | W
ADD                    F | D | E | W | W
```

   (b) Contents of the reorder buffer :

Reorder Buffer

Valid	Tag	Opcode	Rd	Dest. Value	Dest. Value Ready	Exception?
0	A	ADD	r0	5	1	0
1	X	MUL	r3	?	0	1
1	B	ADD	r3	12	1	0
1	Y	MUL	r4	?	0	?
1	C	ADD	r3	?	0	?
0						
0						