CLC - Topic 7

Colt Uhlenhopp & Jack Utzerath

Grand Canyon University

CST - 307

Professor Citro

4/14/2023

CLC - Topic 7 Project

Problems in this exercise refer to the following sequence of instructions:

Instructions	Execution Cycles
FP_Add/Sub	2
FP_Multiply	3
FP_Divide	5
INT_Divide	4

Inst 0: INT_add R3, R1, R2

Inst 1: Br_Untaken offset, R3

Inst 2: fp_add F6, F1, F2

Inst 3: fp_div F6, F2, F3

Inst 4: fp_sub F2, F3, F6

Inst 5: fp_mult F6, F2, F3

Inst 6: fp_add F5, F2, F1

Inst 7: fp_sub F1, F5, F3

For this problem, assume:

IF	ID	EX	MEM	WB
200ps	120ps	150ps	190ps	100ps

William Stallings Link for Simulating the Programs: CPU Cycles Program Run Simulation

A. Indicate All Dependencies that can be found in this instruction set, and their Type (RAW, WAR, RAR, WAW, Structure, Control)

Dependency Type	Register in Conflict	Instr. Nos. Involved
RAW	R3	0, 1
WAW	F6	2, 3
RAW	F6	2, 4
RAW	F6	3, 4
WAR	F2	3, 4
WAW	F6	3, 5
RAW	F2	4, 5
RAW	F2	4, 6
RAW	F5	6, 7
Structural	Unit: + | - (f)	6, 7
Data Hazard	MEM and WB enter stage at same time	3, 6
Data Hazard	MEM and WB enter stage at same time	3, 5

Data Hazard	MEM and WB enter stage at same time	5, 6

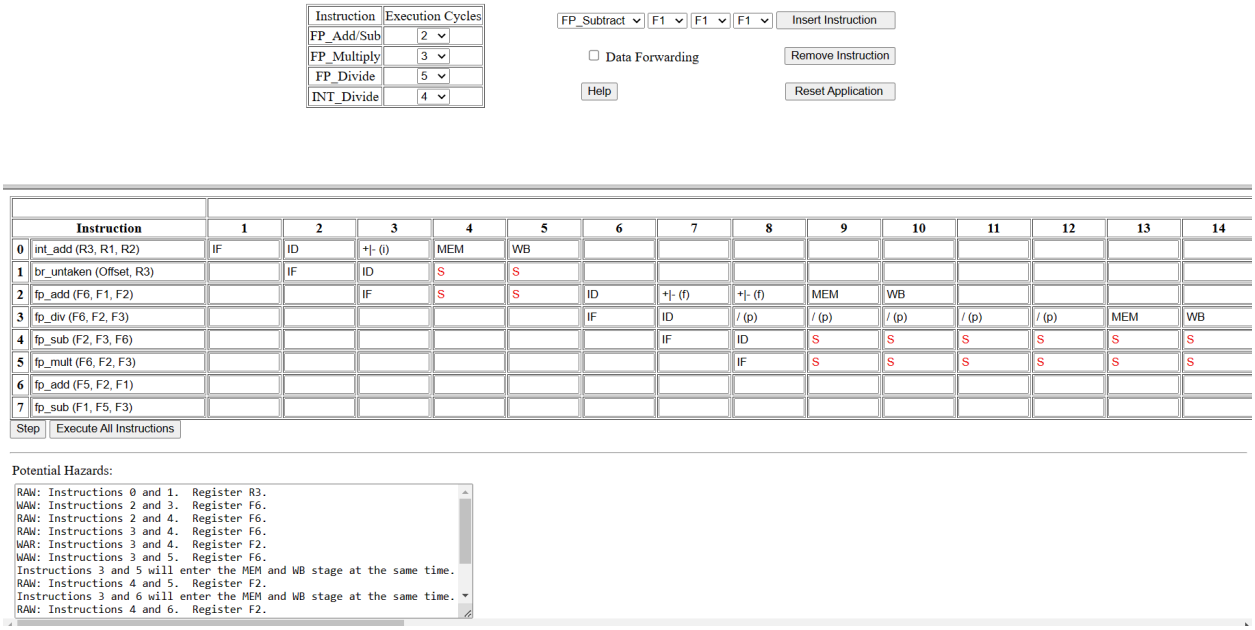
B. Assume "no forwarding" in this pipelined processor and apply (Inst. 1, 2, and 3), from above. Indicate hazards and add NOP (Stall) instructions to eliminate them.

Cycles Where Hazard Eliminated	Instruction No. Delayed	Pipeline Phase Delayed	Reason Delay Occurred
4 - 5	# 1	During Execution	RAW data hazard, NOOP (stall) was added to prevent data hazard
4 -5	# 2	ID	Same NOOP (stall) was added to prevent hazard
9 - 14	# 4	During Execution	RAW and WAR hazard, NOOP (stall) was added
9 - 14	# 5	ID	Same NOOP (stall) was added to prevent the hazards
16 - 18	# 5	During execution	RAW data hazard, NOOP (stall) added to prevent data hazard
16 - 18	# 6	ID	Same NOOP (stall) added
20	# 6	During Execution	RAW data hazard, NOOP (stall) added
20	# 7	ID	Same NOOP (stall) was added to prevent the hazard

22 - 24	**# 7**	**During Execution**	**RAW and Structural Data hazard, NOOP (stall) was added to prevent hazard**

C. What is the total execution time of this instruction sequence in the "no forwarding" pipelined processor?

Total Execution Time for "No Forwarding" Pipeline Processor
4630 ps



D. Now, assume a "forwarding" pipelined processor and apply (Inst. 1, 2, and 3), from above. Indicate hazards found, and add NOP (Stall) instructions to eliminate them.

Dependency Type	**Register in Conflict**	**Instr. Nos. Involved**
RAW	**R6**	**2, 3**
WAW	**F6**	**3, 4**

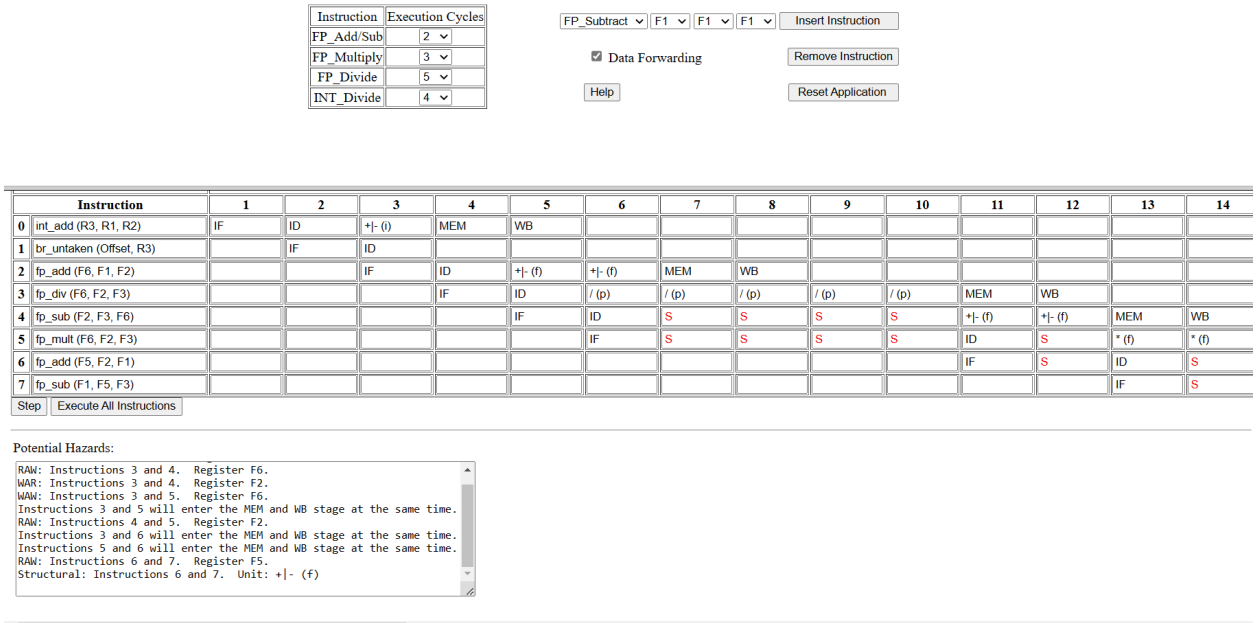
WAR	**F2**	**3, 4**
WAW	**F6**	**3, 5**
Data Hazard	**MEM and WB enter stage at same time**	**3, 5**
RAW	**F2**	**4, 5**
Data Hazard	**MEM and WB enter stage at same time**	**3, 6**
Data Hazard	**MEM and WB enter stage at same time**	**5, 6**
RAW	**F5**	**6, 7**
Structural	**Unit: + | - (f)**	**6, 7**

Cycle No. Where Hazard Eliminated	Instruction No. & Pipeline Name Delayed	Reason for Change
7 - 10	**# 4, During Execution**	**WAW and WAR hazards prevented with NOOP (stall)**
7 - 10	**# 5, ID**	**Same NOOP (stall) added to prevent hazards**
12	**# 5, During Execution**	**RAW data hazard eliminated with NOOP (stall) added**
12	**# 6, ID**	**Same NOOP (stall) added to prevent hazards**

14	**# 6, during execution**	**RAW data Hazard eliminated with NOOP (stall)**
14	**# 7, ID**	**Same NOOP (stall) added to prevent hazards**
16	**# 7, during Execution**	**Structural Data Hazard eliminated with NOOP (stall)**

E. What is the Total execution of this instruction sequence in the "forwarding" pipelined processor?

Total Execution Time for "Forwarding" Pipeline Processor
3510 ps



Instruction	Execution Cycles
FP_Add/Sub	2
FP_Multiply	3
FP_Divide	5
INT_Divide	4

FP_Subtract F1 F1 F1 Insert Instruction

Data Forwarding Remove Instruction

Help Reset Application

	Instruction	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	int_add (R3, R1, R2)	IF	ID	+|- (i)	MEM	WB									
1	br_untaken (Offset, R3)		IF	ID											
2	fp_add (F6, F1, F2)			IF	ID	+|- (f)	+|- (f)	MEM	WB						
3	fp_div (F6, F2, F3)				IF	ID	/ (p)	/ (p)	/ (p)	/ (p)	/ (p)	MEM	WB		
4	fp_sub (F2, F3, F6)					IF	ID	S	S	S	S	+|- (f)	+|- (f)	MEM	WB
5	fp_mult (F6, F2, F3)						IF	S	S	S	S	ID	S	* (f)	* (f)
6	fp_add (F5, F2, F1)											IF	S	ID	S
7	fp_sub (F1, F5, F3)													IF	S

Step Execute All Instructions

Potential Hazards:

```
RAW: Instructions 3 and 4.  Register F6.
WAR: Instructions 3 and 4.  Register F2.
WAW: Instructions 3 and 5.  Register F6.
Instructions 3 and 5 will enter the MEM and WB stage at the same time.
RAW: Instructions 4 and 5.  Register F2.
Instructions 3 and 6 will enter the MEM and WB stage at the same time.
Instructions 5 and 6 will enter the MEM and WB stage at the same time.
RAW: Instructions 6 and 7.  Register F5.
Structural: Instructions 6 and 7.  Unit: +|- (f)
```

F. What is the Speedup due to Using a Forwarding Pipelined Processor?

Processing Speedup due to Forwarding
4630 ps (no forwarding) / 3510 ps (forwarding) = 1.32