CSE3666

Homework 7 Solutions

1

When executing LW, the processor uses ALU to compute the memory address. ALU result is the address, which is the sum of register rs1 and the immediate.

2

	Instructions	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Example	ADD x1, x2, x3	IF	ID	EX	MEM	WB										
	SUB x1, x1, x2		IF	ID	-	-	EX	MEM	WB							
a)	ADD x1, x2, x3	IF	ID	EX	MEM	WB										
	SUB x2, x1, x4		IF	ID	-	-	EX	MEM	WB							
	SW x1, 0(x2)			IF	-	-	ID	-	-	EX	MEM	WB				
b)	ADD x1, x2, x3	IF	ID	EX	MEM	WB										
	LW x2, 0(x1)		IF	ID	-	-	EX	MEM	WB							
	LW x3, 0(x1)			IF	-	-	ID	EX	MEM	WB						
	SUB x4, x2, x3						IF	ID	-	-	EX	MEM	WB			
	SW x4, 0(x3)							IF	-	-	ID	-	-	EX	MEM	WB
c)	LW x1, 0(x10)	IF	ID	EX	MEM	WB										
	SW x1, 8(x10)		IF	ID	-	-	EX	MEM	WB							
	LW x2, 0(x11)			IF	-	-	ID	EX	MEM	WB						
	ADD x3, x1, x2						IF	ID	-	-	EX	MEM	WB			
	SUB x4, x2, x2							IF	-	-	ID	EX	MEM	WB		

3

	Instructions	Forward rs1	Forward rs2	1	2	3	4	5	6	7	8	9	10
Example	ADD x1, x2, x3			IF	ID	EX	MEM	WB					
	SUB x1, x1, x2	EX/MEM->EX			IF	ID	EX	MEM	WB				
a)	ADD x1, x2, x3			IF	ID	EX	MEM	WB					
	SUB x2, x1, x4	EX/MEM->EX			IF	ID	EX	MEM	WB				
	SW x1, 0(x2)	EX/MEM->EX	MEM/WB->EX			IF	ID	EX	MEM	WB			
b)	ADD x1, x2, x3			IF	ID	EX	MEM	WB					
	LW x2, 0(x1)	EX/MEM->EX			IF	ID	EX	MEM	WB				
	LW x3, 0(x1)	MEM/WB->EX				IF	ID	EX	MEM	WB			
	SUB x4, x2, x3		MEM/WB->EX				IF	ID	-	EX	MEM	WB	
	SW x4, 0(x3)		EX/MEM->EX					IF	-	ID	EX	MEM	WB
c)	LW x1, 0(x10)			IF	ID	EX	MEM	WB					
	SW x1, 8(x10)		MEM/WB->MEM		IF	ID	EX	MEM	WB				
	LW x2, 0(x11)					IF	ID	EX	MEM	WB			
	ADD x3, x1, x2		MEM/WB->EX				IF	ID	-	EX	MEM	WB	
	SUB x4, x2, x2							IF	-	ID	EX	MEM	WB

4

a)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
I1: lw t0, 0(s0)	IF	ID	EX	MEM	WB													
I2: lw t1, 0(s1)		IF	ID	EX	MEM	WB												
I3: add t0, t0, t1			IF	ID	-	-	EX	MEM	WB									
I4: sw t0, 0(s0)				IF	-	-	ID	-	-	EX	MEM	WB						
I5: addi s0, s0, 4							IF	-	-	ID	EX	MEM	WB					
I6: addi s1, s1, 4										IF	ID	EX	MEM	WB				
I7: addi s2, s2, 1											IF	ID	EX	MEM	WB			
I8: bne s2, s3, I1												IF	ID	-	-	EX	MEM	WB
I9:													IF	-	-	ID	EX	X
I1: lw t0, 0(s0)																		IF
Number of cycles/iteration	17																	
Stall cycles due to data hazards	6																	
Stall cycles due to control hazards	3																	

b)

17 cycles per iteration. 8 instructions. 6 stall cycles for data hazards. 3 cycles for control hazards.

c)

There are many ways.
One way is to move I6 and I7 after I2 because I3 is stalled for 2 cycles. Doing so, we also remove the data hazards on I8. I4 still stalls for 2 cycles.

Each iteration now takes 13 cycles. 8 instructions. 2 stall cycles for data hazards. 3 cycles for control hazards.

d)

The speedup from scheduling is 17 / 13 = 1.31.

The longest dependency chain is I1, I2, I3, I4, I5, I8. There are 6 instructions, plus 4 cycles for data hazards and 3 cycles for control hazards.

If we are allowed to change the offset in I4, I5 can be removed from the chain. We can then save one more cycle.

5

a)

	1	2	3	4	5	6	7	8	9	10	11	12	13
I1: lw t0, 0(s0)	IF	ID	EX	MEM	WB								
I2: lw t1, 0(s1)		IF	ID	EX	MEM	WB							
I3: add t0, t0, t1			IF	ID	-	EX	MEM	WB					
I4: sw t0, 0(s0)				IF	-	ID	EX	MEM	WB				
I5: addi s0, s0, 4						IF	ID	EX	MEM	WB			
I6: addi s1, s1, 4							IF	ID	EX	MEM	WB		
I7: addi s2, s2, 1								IF	ID	EX	MEM	WB	
I8: bne s2, s3, I1									IF	ID	EX	MEM	WB
I9:										IF	ID	EX	X
I1: lw t0, 0(s0)													IF
Number of cycles/iteration	12												
Stall cycles due to data hazards	1												
Stall cycles due to control hazards	3												

b)

12 cycles per iteration. 8 instructions. One stall cycle for data hazards. Three stall cycles for control hazards.

c)

The key is to separate I2 and I3. One way is to move I7 between I2 and I3. Each iteration now takes 11 cycles, for 8 instructions and three install cycles due to control hazard.

d)

The speedup from scheduling is 12 / 11 = 1.09.

6

The CPI overhead from data hazard is 0.3 * 0.1 * 1= 0.03.
The CPI overhead from control hazard is 0.25 * 0.6 * 3 = 0.45.
The overall CPI of this application is 1 + 0.03 + 0.45 = 1.48.

7

If x10 is 0, the branch is taken. The number of cycles is 1 + 3 = 4. 1 cycle for branch and 3 stall cycles.

If x10 is not 0, the branch is not taken. There is no control hazard. The number of cycles is 2, one for each instruction.

The average number of cycles required is 0.8 * 4 + 0.2 * 2 = 3.6.

How would you improve the performance of the operation?