CMPE 200

COMPUTER ARCHITECTURE

Lecture 4 – The Processor – Pipelining
Homework Assignment Review

Bo Yu
Computer Engineering Department
SJSU

Adapted from Computer Organization and Design, 5th Edition, 4th edition, Patterson and Hennessy, MK and Computer Architecture – A Quantitative Approach, 4th edition, Patterson and Hennessy, MK

Lecture 4 Key Concepts Review

4.10 In this exercise, we examine how resource hazards, control hazards, and Instruction Set Architecture (ISA) design can affect pipelined execution. Problems in this exercise refer to the following fragment of MIPS code:

```
sw r16,12(r6)
lw r16,8(r6)
beq r5,r4,Label          # Assume r5!=r4
add r5,r1,r4
slt r5,r15,r4
```

Assume that individual pipeline stages have the following latencies:

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

4.10.1 [10] <4.5> For this problem, assume that all branches are perfectly predicted (this eliminates all control hazards) and that no delay slots are used. If we only have one memory (for both instructions and data), there is a structural hazard every time we need to fetch an instruction in the same cycle in which another instruction accesses data. To guarantee forward progress, this hazard must always be resolved in favor of the instruction that accesses data. What is the total execution time of this instruction sequence in the 5-stage pipeline that only has one memory? We have seen that data hazards can be eliminated by adding nops to the code. Can you do the same with this structural hazard? Why?

Lecture 4 Key Concepts Review

4.10.1 Solution: When an instruction cannot be fetched because of structure hazard, i.e., load/store is using memory in the same cycle as instruction fetch, a Stall is inserted to that cycle.

Instruction	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8	CC9	CC10	CC11
sw r16,12(r6)	IF	ID	EX	MEM	WB						
lw r16,8(r6)		IF	ID	EX	MEM	WB					
beq r5,r4,Label			IF	ID	EX	MEM	WB				
add r5,r1,r4				S	S	IF	ID	EX	MEM	WB	
slt r5,r15,r4							IF	ID	EX	MEM	WB

a) Total execution time = 11 cycles x 200ps/cycle = 2200ps

b) We can not add NOPs to the code to eliminate this hazard – NOPs need to be fetched just like any other instructions, so this hazard must be addressed with a hardware hazard detection unit in the processor.

NOP – an instruction that does no operation to change state.

Terminology Explanation: Stall vs. NOP

- **Stall (Bubble):**
  - Stall is the cycle inserted into the pipeline stage by processor to solve the pipeline data hazard
  - The stall is only inserted at the pipeline stage when the wait is needed
  - The inserted cycle(s) will cause the delay of the next stage until the expected data is available. Once the data is available, it will resume the next stage at the cycle after the stall (delay)
  - Stall delays the entire pipeline, i.e., when stall is inserted at CC5 of the 3rd instruction, the same stall is inserted at CC% of the 4th and 5th instructions as well, to avoid the hazards
  - When an instruction is converted to NOP, it gets re-fetched next cycle
  - Hazard detection unit will determine where and when to insert a stall
- **NOP (No Operation):**
  - NOP is an instruction that does no operation to change state
  - NOP is an instruction that needs to be fetched and executed, it takes 5 stages of pipeline (doing nothing though)
  - In MIPS, NOP is equivalent to “sll $zero $zero 0”. It can help solve pipeline hazards, and also be used in certain type of interrupt/exception
  - the instruction after NOP needs to be re-fetched from the beginning
  - NOP should not use in place of the stall and vice versa

Lecture 4 Key Concepts Review

4.13 This exercise is intended to help you understand the relationship between forwarding, hazard detection, and ISA design. Problems in this exercise refer to the following sequence of instructions, and assume that it is executed on a 5-stage pipelined datapath:

```
add r5,r2,r1
lw r3,4(r5)
lw r2,0(r2)
or r3,r5,r3
sw r3,0(r5)
```

4.13.1 [5] <4.7> If there is no forwarding or hazard detection, insert nops to ensure correct execution.

4.13.4 [20] <4.7> If there is forwarding, for the first five cycles during the execution of this code, specify which signals are asserted in each cycle by hazard detection and forwarding units in Figure 4.60 (use Figure 4.57 for the correct forwarding lines) .

Lecture 4 Key Concepts Review

4.13.1 Solution (no forwarding or hazard detection): Stall cannot be used.

Instruction	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8	CC9	CC10	CC11	CC12	CC13	CC14
add r5,r2,r1	IF	ID	EX	MEM	WB									
lw r3,4(r5)				IF	ID	EX	MEM	WB						
lw r2,0(r2)					IF	ID	EX	MEM	WB					
or r3,r5,r3							IF	ID	EX	MEM	WB			
sw r3,0(r5)										IF	ID	EX	MEM	WB

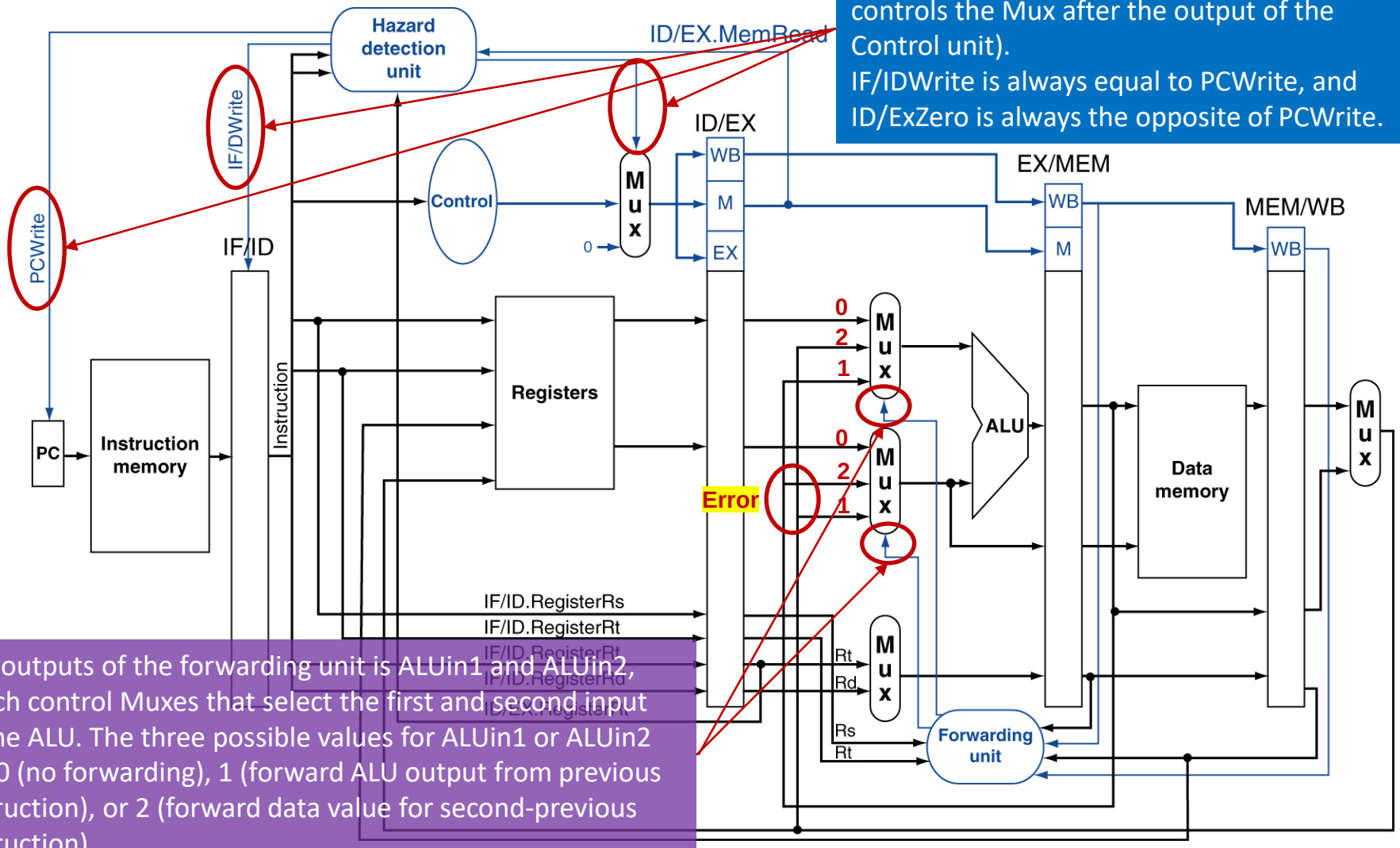
ADD R5,R2,R1
NOP
NOP
LW R3,4(R5)
LW R2,0(R2)
NOP
OR R3,R5,R3
NOP
NOP
SW R3,0(R5)

Instruction	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8	CC9	CC10	CC11	CC12	CC13	CC14
add r5,r2,r1	IF	ID	EX	M	WB									
NOP														
NOP														
lw r3,4(r5)				IF	ID	EX	M	WB						
lw r2,0(r2)					IF	ID	EX	M	WB					
NOP														
or r3,r5,r3							IF	ID	EX	M	WB			
NOP														
NOP														
sw r3,0(r5)										IF	ID	EX	M	WB

Lecture 4: Key Concepts Review

Datapath with Hazard Detection

The outputs of the hazard detection unit are: PCWrite, IF/IDWrite, and ID/EXZero (which controls the Mux after the output of the Control unit).
IF/IDWrite is always equal to PCWrite, and ID/ExZero is always the opposite of PCWrite.



The outputs of the forwarding unit is ALUin1 and ALUin2, which control Muxes that select the first and second input of the ALU. The three possible values for ALUin1 or ALUin2 are 0 (no forwarding), 1 (forward ALU output from previous instruction), or 2 (forward data value for second-previous instruction).

Lecture 4: Key Concepts Review

4.13.4 Solution (first 5 cycles):

Instruction	CC1	CC2	CC3	CC4	CC5	Signals from Hazard Detection Unit And Forwarding Unit
add r5,r2,r1	IF	ID	EX	MEM	WB	1. PCWrite=IF/IDWrite=1,ID/EXZero=0, ALUin1=X, ALUin2=X 2. PCWrite=IF/IDWrite=1,ID/EXZero=0, ALUin1=X, ALUin2=X 3. PCWrite=IF/IDWrite=1,ID/EXZero=0, ALUin1=0, ALUin2=0 4. PCWrite=IF/IDWrite=1,ID/EXZero=0, ALUin1=1, ALUin2=0 5. PCWrite=IF/IDWrite=1,ID/EXZero=0, ALUin1=0, ALUin2=0
lw r3,4(r5)		IF	ID	EX	MEM	
lw r2,0(r2)			IF	ID	EX	
or r3,r5,r3				IF	ID	
sw r3,0(r5)					IF	

Lecture 4: Key Concepts Review

4.13.4 Solution (all cycles):

Instruction	CC1	CC2	CC3	CC4	CC5	CC6	CC7	CC8	CC9	
add r5,r2,r1	IF	ID	EX	MEM	WB					
lw r3,4(r5)		IF	ID	EX	MEM	WB				
lw r2,0(r2)			IF	ID	EX	MEM	WB			
or r3,r5,r3				IF	ID	EX	MEM	WB		
sw r3,0(r5)					IF	ID	EX	MEM	WB	
PCWrite	1	1	1	1	1	1	1	1	1	**Signals from Hazard Detection Unit**
IF/IDWrite	1	1	1	1	1	1	1	1	1	
ID/EXZero	0	0	0	0	0	0	0	0	0	
ALUin1 (Rs)	X	X	0	1	0	0	0	0	0	**Signals from Forwarding Unit**
ALUin2 (Rt)	X	X	0	0	0	2	1	0	0	