VE370 RC Final (Part 1)

2020/12/04

Contents

- Control Hazards
- Quick Review on Data Hazards

Contents

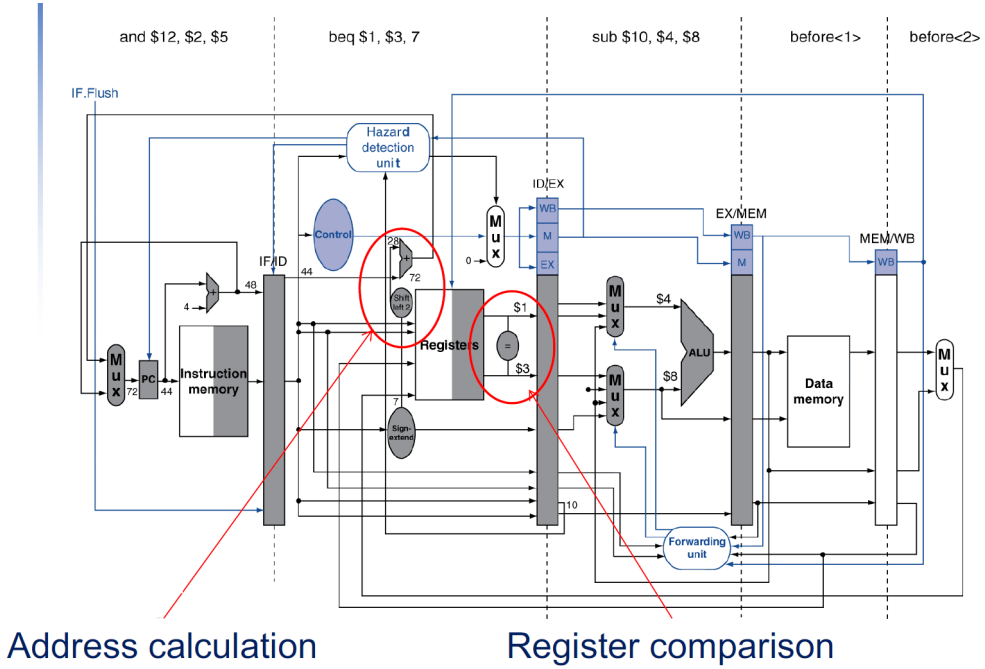
- Control Hazards
- Quick Review on Data Hazards

Control Hazards Overview

- Issue: Fetching next instruction depends on branch outcome, but pipeline can't always fetch correct instruction because
  - When branch instruction is still in the ID stage, target instruction is needed in the IF stage
- Solution:
  - Stall on branch
  - **Always assume branch not taken or taken**
  - **Branch prediction**
  - Delayed Branch
- Different from the data hazards for branches

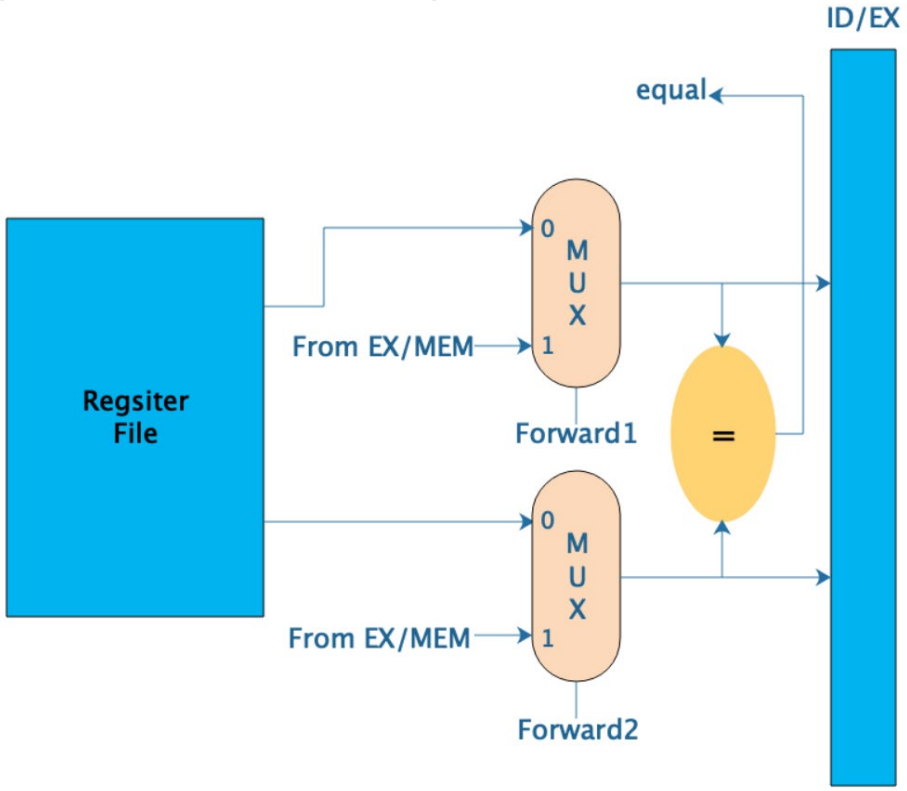
Assume Branch Not Taken

- For branch not taken, correct assumption, no penalty
- For taken branch, penalty will be to flush some (up to 3) instructions
  - Previously, branch outcomes are determined in MEM stage
- Penalty can be reduced
  - Moving hardware for determining PC to **ID** stage including
    - **Target address calculation**
    - **Register comparator**
  - Now only **1** instruction to be flushed
- Flush an instruction
  - Clear IF/ID pipeline register by **IF.Flush**
- Control hazard detection
  - if (Branch && Equal)
    IF.Flush = 1



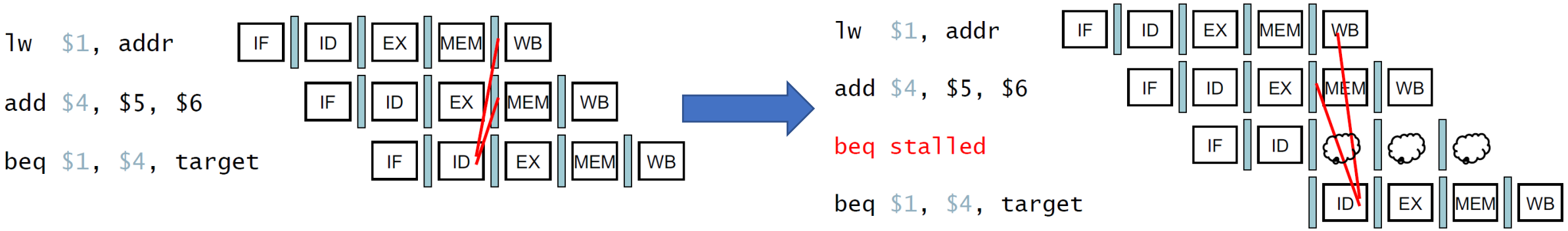
Forwarding paths

- Changing datapath structure causes more possible data hazards
  - Can resolve using new forwarding paths

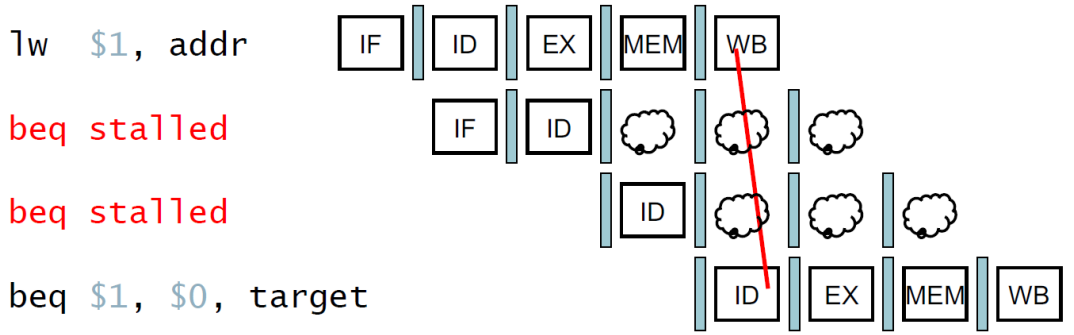


Data Hazards for Branches

- If a comparison register is a destination of immediately preceding ALU instruction or 2nd preceding load instruction
  - Need 1 stall cycle (then the ALU instruction needs forwarding)



- If a comparison register is a destination of immediately preceding load instruction
  - Need 2 stall cycles



Example: HW5.1 (2)

```
lw R2,0(R1)
Label1: beq R2,R0,Label2 # Not taken
once, then taken
lw R3,0(R2)
beq R3,R0,Label1 # Taken
add R1,R3,R1
Label2: sw R1,0(R2)
```

- For the given code, what is the speedup achieved by moving branch execution into the ID stage?

- Solution:

Not moved:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
lw R2,0(R1)	IF	ID	EX	ME	WB													
beq R2,R0,Label2		IF	*	ID	EX	ME	WB											
lw R3,0(R2)				IF	ID	EX	ME	WB										
beq R3,R0,Label1					IF	*	ID	EX	ME	WB								
beq R2,R0,Label2										IF	ID	EX	ME	WB				
sw R1,0(R2)														IF	ID	EX	ME	WB

Moved:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
lw R2,0(R1)	IF	ID	EX	ME	WB											
beq R2,R0,Label2		IF	*	*	ID	EX	ME	WB								
lw R3,0(R2)					IF	ID	EX	ME	WB							
beq R3,R0,Label1						IF	*	*	ID	EX	ME	WB				
beq R2,R0,Label2										IF	ID	EX	ME	WB		
sw R1,0(R2)												IF	ID	EX	ME	WB

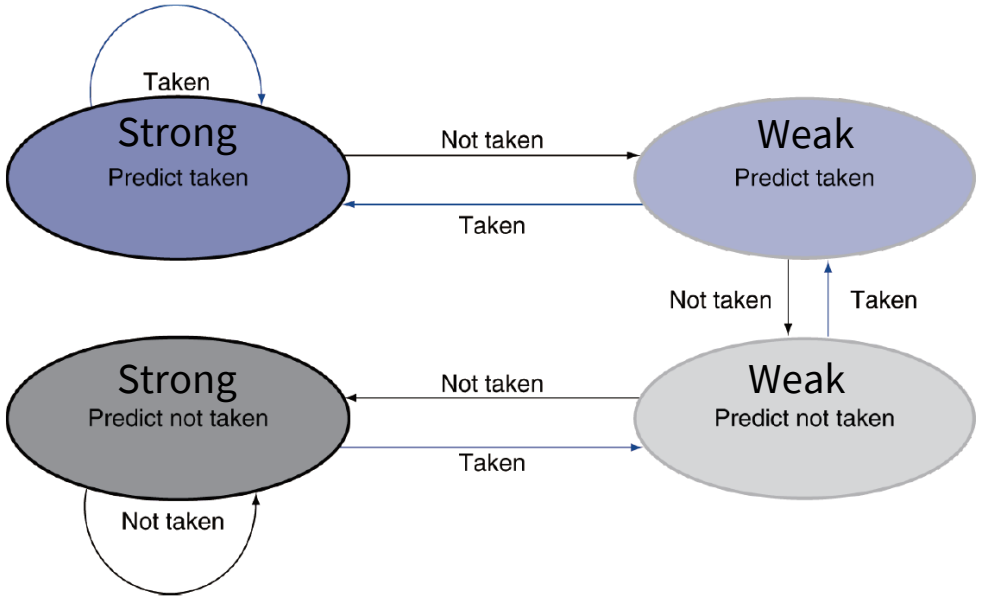
Not moved: 18 clock cylces

Moved: 16 clock cycles

Speedup: 18/16 = 1.125

Dynamic Branch Prediction

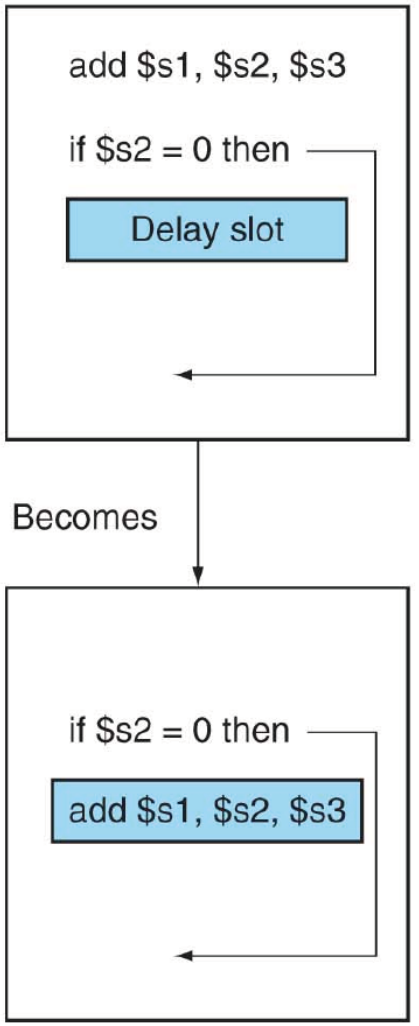
- 1-Bit Dynamic Predictor
  - Example (Starting from Assume branch taken): Pattern: NT Prediction: T->NT
- 2-Bit Dynamic Predictor
  - Only change prediction on two successive mispredictions



  - Example: HW5.3

Delayed Branch

- Always execute the instruction immediately following branch for 5 stage pipeline
  - Called Branch delay slot
  - Maybe more delays for deeper pipeline
  - Will remove the 1 clock cycle penalty
  - Will work only if instructions can be found to fill the delay slot



Contents

- Control Hazards
- Quick Review on Data Hazards

Forwarding Path

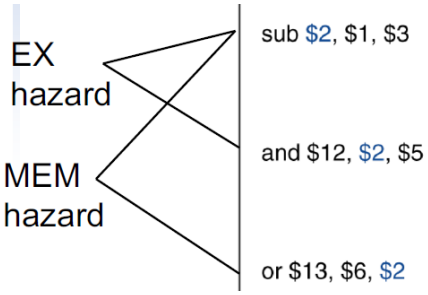
- Forwarding paths are created between stage pipeline registers and ALU inputs

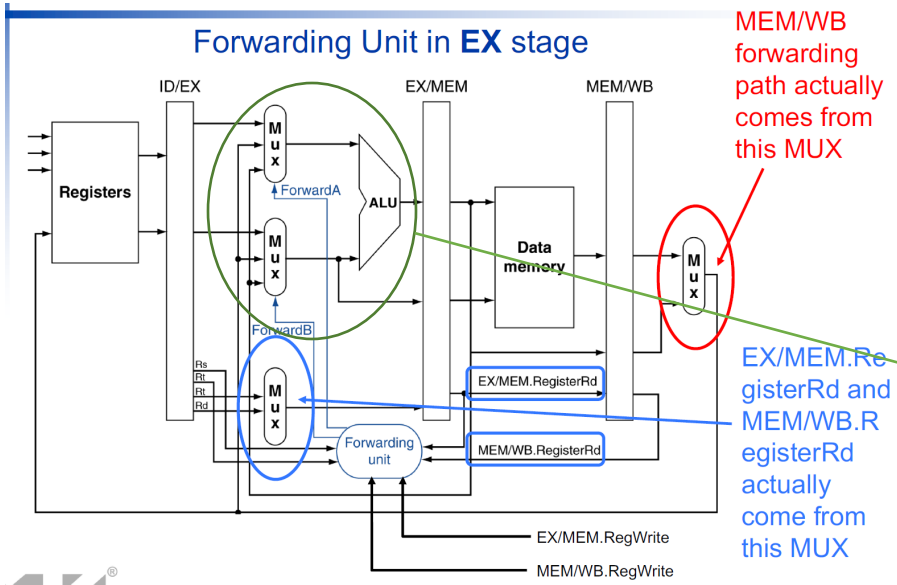
EX hazard

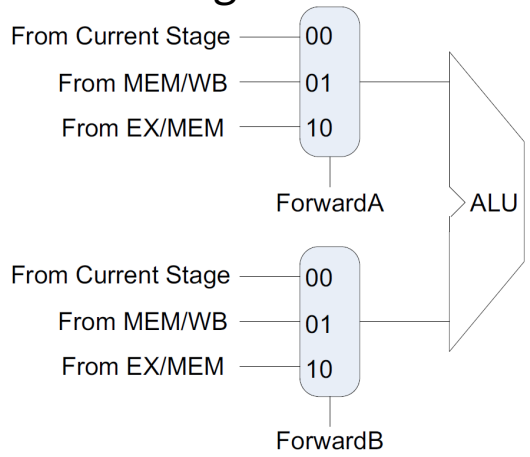
- if (EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0)
  and (EX/MEM.RegisterRd == ID/EX.RegisterRs))
  MUX select signal ForwardA = 10
- if (EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0)
  and (EX/MEM.RegisterRd == ID/EX.RegisterRt))
  MUX select signal ForwardB = 10

MEM hazard (not EX hazard)

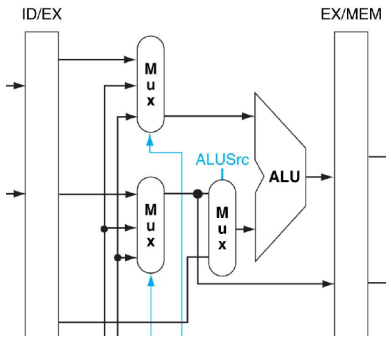
- if (MEM/WB.RegWrite and (MEM/WB.RegisterRd ≠ 0)
  and (MEM/WB.RegisterRd = ID/EX.RegisterRs)
  and not (EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0)
  and (EX/MEM.RegisterRd = ID/EX.RegisterRs)))
  ForwardA = 01
- if (MEM/WB.RegWrite and (MEM/WB.RegisterRd ≠ 0)
  and (MEM/WB.RegisterRd = ID/EX.RegisterRt)
  and not (EX/MEM.RegWrite and (EX/MEM.RegisterRd ≠ 0)
  and (EX/MEM.RegisterRd = ID/EX.RegisterRt)))
  ForwardB = 01







Forwarding Paths with ALUSrc



Load-use Hazard

- if ID/EX.MemRead and ((ID/EX.RegisterRt == IF/ID.RegisterRs) or (ID/EX.RegisterRt == IF/ID.RegisterRt))
- Hazard Detection in **ID** stage
- Resolve by inserting 1 stall cycle
  - Force control signals in ID/EX register to 0's
  - Prevent updates of PC and IF/ID registers
- Datapath with Hazard Detection