An Overview of Pipelining

DR. RAJIB RANJAN MAITI

CSIS, BITS-PILANI, HYDERABAD

Designing Instruction Sets for Pipelining

- **First**, all instructions are of same length.
  - Easier to fetch instructions in the first pipeline stage
  - Easier to decode in the second stage.
  - **In x86**, **instructions** vary from **1 byte to 15 bytes**,
    - pipelining is considerably more challenging

- **Second**, only a few instruction formats,
  - the source register fields is in the same place
  - the second stage can begin reading the register file at the same time that the hardware is determining what type of instruction was fetched
  - **If MIPS instruction formats were not symmetric**, then we would need to split stage 2, increasing the number of stages

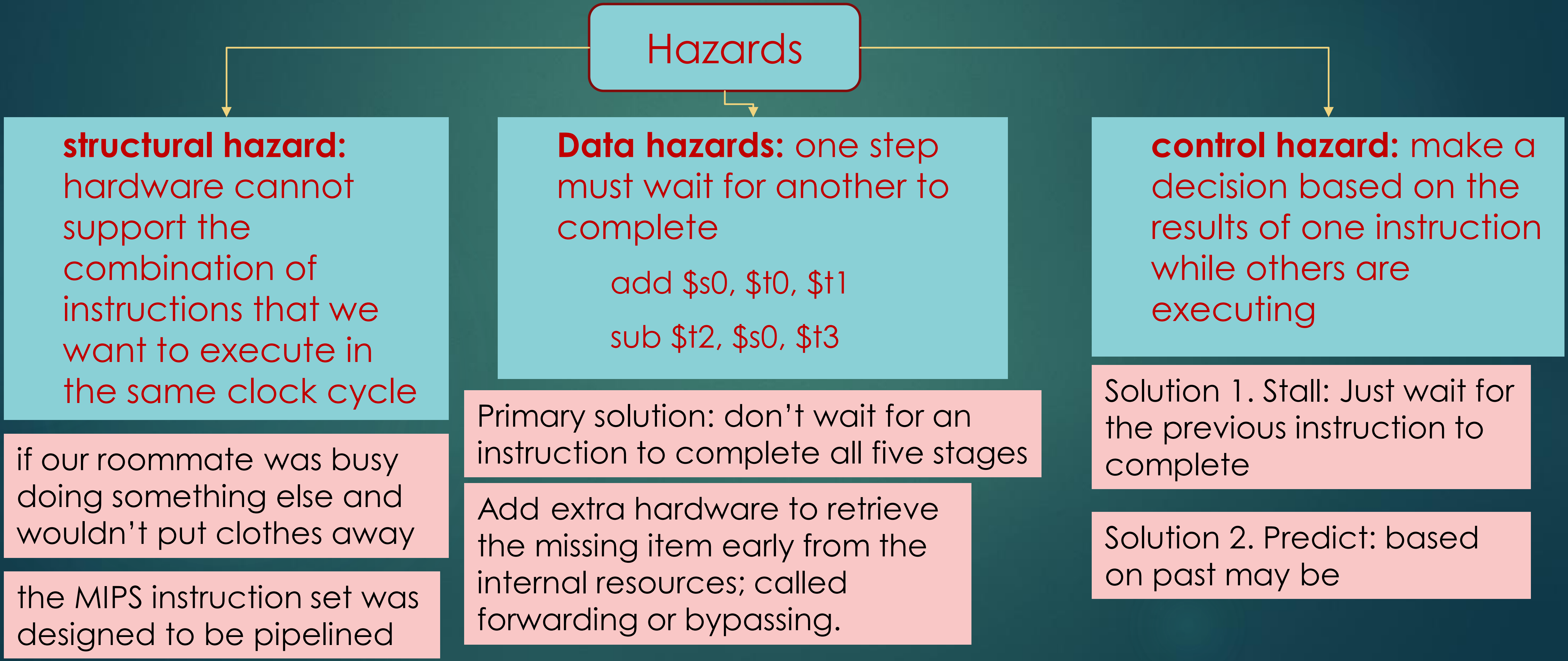
- **Third**, **memory operands** only in **loads or stores**
  - **As a result the execute stage can calculate the memory address** and then access data memory in next stage
  - If we could **operate on the operands in memory**, as in the **x86**, stages 3 and 4 in MIPS would expand to an address stage, memory stage, and then execute stage.

- **Fourth**, operands must be aligned in memory.
  - So that the requested data can be transferred between processor and memory in a single pipeline stage.

Pipeline Hazards

Pipeline Hazards

- A situation where the next instruction cannot be executed in the following clock cycle



Hazards

structural hazard: hardware cannot support the combination of instructions that we want to execute in the same clock cycle

if our roommate was busy doing something else and wouldn't put clothes away

the MIPS instruction set was designed to be pipelined

Data hazards: one step must wait for another to complete

add $s0, $t0, $t1

sub $t2, $s0, $t3

Primary solution: don't wait for an instruction to complete all five stages

Add extra hardware to retrieve the missing item early from the internal resources; called forwarding or bypassing.

control hazard: make a decision based on the results of one instruction while others are executing

Solution 1. Stall: Just wait for the previous instruction to complete

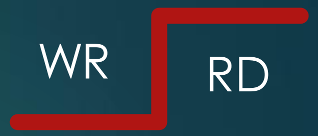
Solution 2. Predict: based on past may be

structural hazard:

- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

two instructions in the pipeline require the same resource

Solution is at hardware level: splitting the RF access to two clock phases, write on first phase and read on second phase



T1	T2	T3	T4	T5	T6	T7	T8	T9
IF	ID ($9)	Ex ($9)	Mem	WB ($8)				
	IF	ID ($11, $12)	Ex ($11, $12)	Mem	WB ($10)			
		IF	ID ($11, $12)	Ex ($8, $12)	Mem	WB ($10)		
			IF	ID ($8, $12)	Ex ($8, $12)	Mem	WB ($10)	

structural hazard:

- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

T1
IF

structural hazard:

- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

T1	T2
IF	ID ($9)
	IF

structural hazard:

- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

T1	T2	T3
IF	ID ($9)	Ex ($9)
	IF	ID ($11, $12)
		IF

Suppose, however, that we had a single memory instead of two memories

structural hazard:

- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

T1	T2	T3	T4
IF	ID ($9)	Ex ($9)	Mem
	IF	ID ($11, $12)	Ex ($11, $12)
		IF	ID ($11, $12)
			IF

Suppose, however, that we had a single memory instead of two memories

structural hazard:

- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

T1	T2	T3	T4	T5
IF	ID ($9)	Ex ($9)	Mem	WB ($8)
	IF	ID ($11, $12)	Ex ($11, $12)	Mem
		IF	ID ($11, $12)	Ex ($8, $12)
			IF	ID ($8, $12)

Suppose, however, that we had a single memory instead of two memories

structural hazard:

- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

T1	T2	T3	T4	T5	T6
IF	ID ($9)	Ex ($9)	Mem	WB ($8)	
	IF	ID ($11, $12)	Ex ($11, $12)	Mem	WB ($10)
		IF	ID ($11, $12)	Ex ($8, $12)	Mem
			IF	ID ($8, $12)	Ex ($8, $12)

Suppose, however, that we had a single memory instead of two memories

structural hazard:

- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

T1	T2	T3	T4	T5	T6	T7
IF	ID ($9)	Ex ($9)	Mem	WB ($8)		
	IF	ID ($11, $12)	Ex ($11, $12)	Mem	WB ($10)	
		IF	ID ($11, $12)	Ex ($8, $12)	Mem	WB ($10)
			IF	ID ($8, $12)	Ex ($8, $12)	Mem

Suppose, however, that we had a single memory instead of two memories

structural hazard:

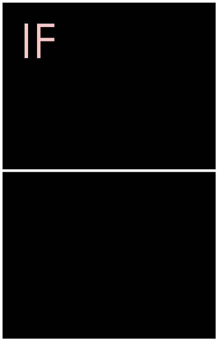
- lw $8, 100($9)
- add $10, $11, $12
- add $10, $11, $12
- add $10, $8, $12

T1	T2	T3	T4	T5	T6	T7	T8	T9
IF	ID ($9)	Ex ($9)	Mem	WB ($8)				
	IF	ID ($11, $12)	Ex ($11, $12)	Mem	WB ($10)			
		IF	ID ($11, $12)	Ex ($8, $12)	Mem	WB ($10)		
			IF	ID ($8, $12)	Ex ($8, $12)	Mem	WB ($10)	

Suppose, however, that we had a single memory instead of two memories

Data Hazards

An instruction depends on the result of prior instruction still in the pipeline



Add $s0, $t0, $t1

Sub $t2, $s0, $t3

Data Hazards

An instruction depends on the result of prior instruction still in the pipeline

Add $s0, $t0, $t1	IF	ID ($t0, $t1)
Sub $t2, $s0, $t3		IF

Data Hazards

An instruction depends on the result of prior instruction still in the pipeline

Add $s0, $t0, $t1	IF	ID ($t0, $t1)	EX ($t0, $t1)
Sub $t2, $s0, $t3		IF	ID ($s0, $t3)

Data Hazards

An instruction depends on the result of prior instruction still in the pipeline

Add $s0, $t0, $t1	IF	ID ($t0, $t1)	EX ($t0, $t1)	Mem
Sub $t2, $s0, $t3		IF	ID ($s0, $t3)	EX ----

Data Hazards

An instruction depends on the result of prior instruction still in the pipeline

Add $s0, $t0, $t1	IF	ID ($t0, $t1)	EX ($t0, $t1)	Mem	WB ($s0)
Sub $t2, $s0, $t3		IF	ID ($s0, $t3)	EX ----	EX ($s0, $t3)

Data Hazards

An instruction depends on the result of prior instruction still in the pipeline

Add $s0, $t0, $t1	IF	ID ($t0, $t1)	EX ($t0, $t1)	Mem	WB ($s0)	
Sub $t2, $s0, $t3		IF	ID ($s0, $t3)	EX ----	EX ($s0, $t3)	Mem

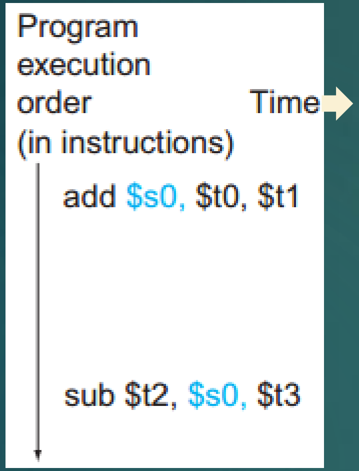
Data Hazards

An instruction depends on the result of prior instruction still in the pipeline

Add $s0, $t0, $t1	IF	ID ($t0, $t1)	EX ($t0, $t1)	Mem	WB ($s0)		
Sub $t2, $s0, $t3		IF	ID ($s0, $t3)	EX ----	EX ($s0, $t3)	Mem	WB ($t2)

Data Hazards: Solution

Solution: forwarding

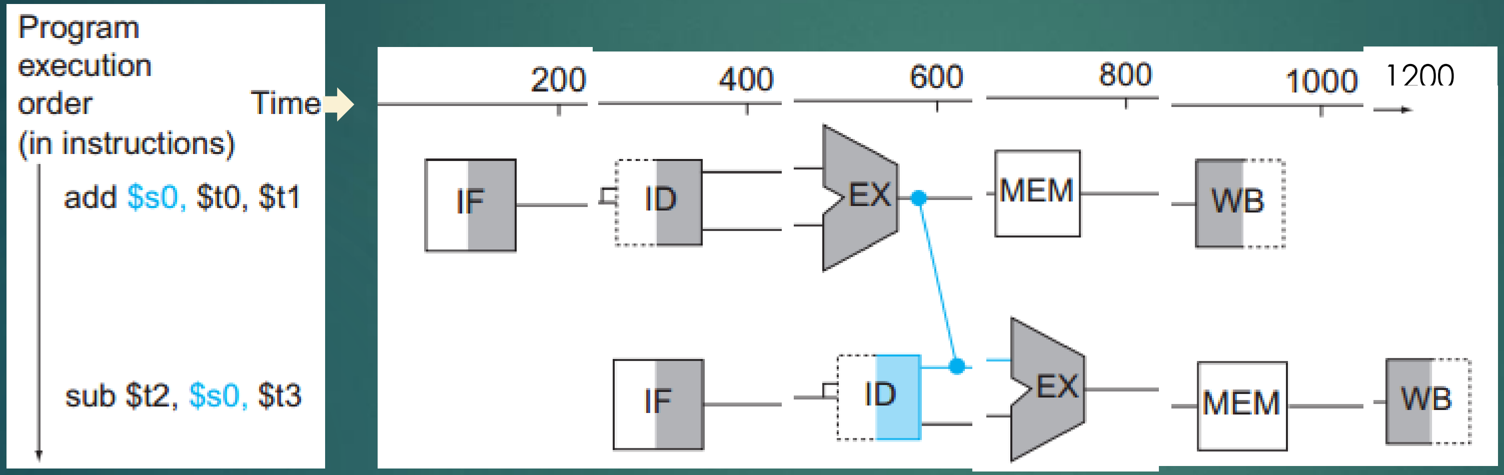


add $s0, $t0, $t1

sub $t2, $s0, $t3

Data Hazards: Solution

Solution: forwarding



Thank You