Pipelining The Datapath

Part 3

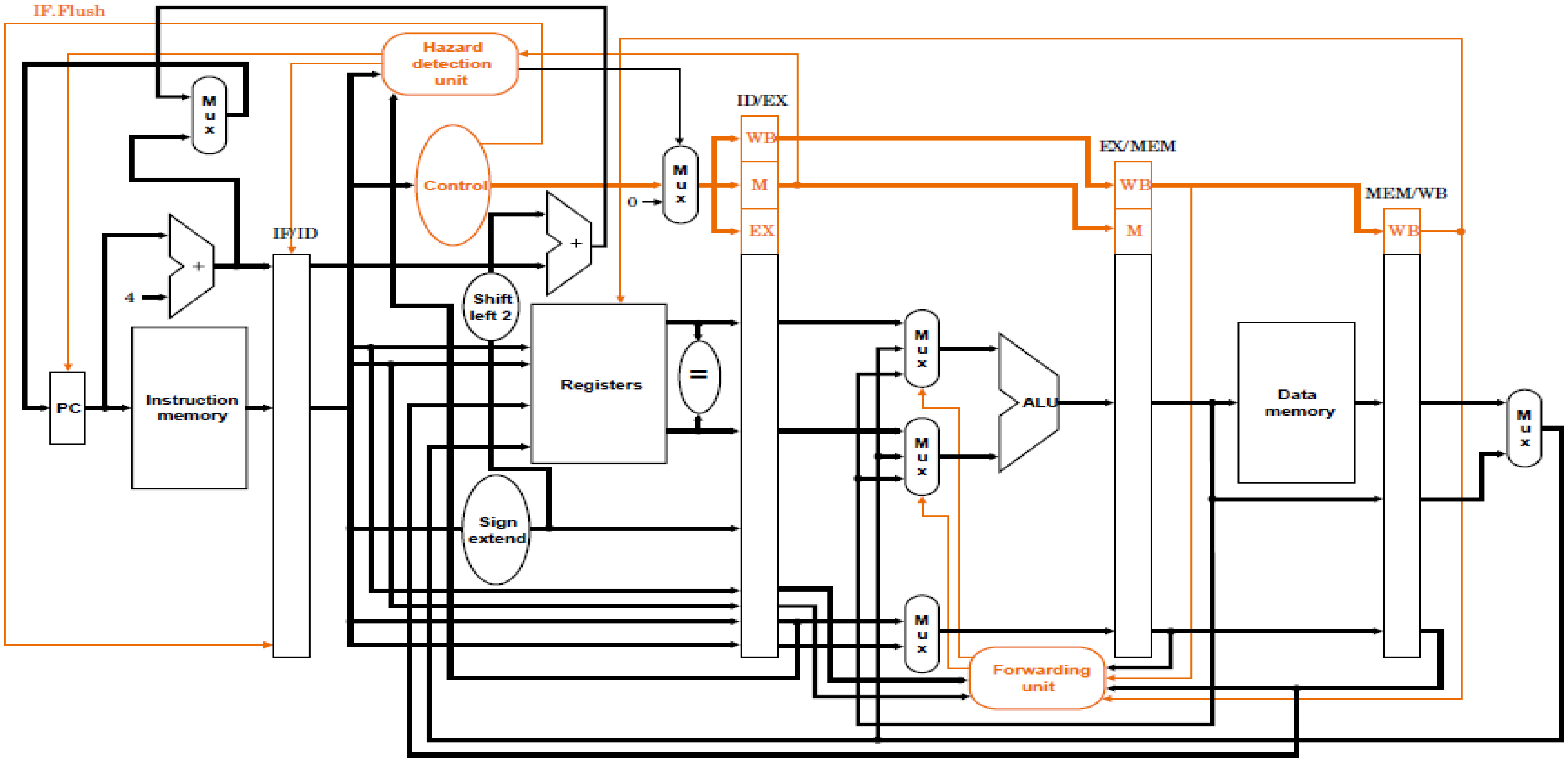
A5 Important Notes :

- Please submit your assignment: download your submission to make sure it is as you expected.
- **A5:** **-25%** penalty for any remark requests where it was the error on the part of the student.
- Submit EARLY ☺
- Avoid Line up at Scanners and Submission delay on Markus!
- Office Hours this Week: **No Office Hours On Thursday July 16**

Trouble in the Pipeline

- Structural Hazards:
  - MULTIPLE INSTRUCTIONS ARE TRYING TO USE THE SAME HARDWARE
  - Memory was a Structural hazard: Multi-cycle Datapath
    - Solution: Split up instruction memory and data memory
    - What Else was a possible structural Hazard?
- Data Hazards:
  - Some data computation needed for a subsequent instruction: **lw $2 100($1), add $1, $2, $8**
  - Or **add $1, $2, $3** followed by **sub $4, $1, $2**
- Control Hazards:
  - Flow of Control Change: PC updated in beq or jump instruction, when do we start next instruction?

Hardware for Branch Flushing



Branching : Examples

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 nop
```

Branching : Examples

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
```

Loop:
Lines
108 to
124

What is this code doing?

Branching : Examples

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
```

Loop:
Lines
108 to
124

What is this code doing?

Summation of an array of nums
Check if this sum is positive
Or negative at the end

Branching : Examples Loop

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

MAY ASSUME Datapath Does Branch Flushing for 3 Errant Instructions

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
```

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

How many Lines of Code execute?

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

How many Lines of Code execute?

2 before loop

5 in the loop plus 1 for lw stall

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

How many Lines of Code execute?

2 before loop

5 in the loop plus 1 for lw stall

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

How many Lines of Code execute?

2 before loop

5 in the loop plus 1 for lw stall

Plus 3 for Branch flush

How many times will this happen for the loop

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

How many Lines of Code execute?

*2 before loop

*5 in the loop plus 1 for lw stall: 6

*Plus 3 for Branch flush : **TOTAL 9 Loop instructions**

*How many times will this happen for the loop?

- * 20 times all these instructions will run
- * 20 x 9 =180

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

How many Lines of Code execute?

*2 before loop

*5 in the loop plus 1 for lw stall: 6

*Plus 3 for Branch flush

*How many times will this happen for the loop?

 * 20 times all these instructions will run

 * 20 x 9 =180

- 3 instructions following loop included in 180
- Total 2+ 180 + …

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

How many Lines of Code execute?

*2 before loop

*5 in the loop plus 1 for lw stall: 6

*Plus 3 for Branch flush

*How many times will this happen for the loop?

   * 20 times all these instructions will run

   * 20 x 9 =180

- 3 instructions following loop
- Total 2+ 180 + …4
- 4 getting pipeline running = 186

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

**How many instructions FLUSHED
Due to Branch Misprediction:**

If this is run on a pipeline that Implements Forwarding and Stalling but NO Early Branch Detection . Ie) Branch is known in $4th Cycle

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

How many instructions FLUSHED Due to Branch Misprediction:

A) 20 x 3

B) 3

C) 19 x 3

D) 21 x 3

E) 19 x 3 + 2

NOW: Forwarding and Stalling and Branching Known in ID stage….One Flushed Branch Instruction, if Branch Mispredicted

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

How many Lines of Code execute?

NOW: Forwarding and Stalling and Branching Known in ID stage….

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

Prior Computation (for your reference) :

*2 before loop

*5 in the loop plus 1 for lw stall: 6

*Plus 3 for Branch flush

*How many times will this happen for the loop?

* 20 times all these instructions will run

* 20 x 9 =180

- 3 instructions following loop
- Total 2+ 180 + …4
- 4 getting pipeline running = 186

NOW: Forwarding and Stalling and Branching Known in ID stage….

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

Prior Computation (for your reference) :

*2 before loop

*5 in the loop plus 1 for lw stall: 6

*Plus 3 for Branch flush

*How many times will this happen for the loop?

* 20 times all these instructions will run

* 20 x 9 =180

- 3 instructions following loop
- Total 2+ 180 + …4
- 4 getting pipeline running = 186

NOW with ID Stage Branch: A) 156 B) 146 C) 148 D) 186 E) 188

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

NOW: Forwarding and Stalling and Branching Known in ID stage….

Prior Computation (for your reference) :

*2 before loop

*5 in the loop plus 1 for lw stall: 6

*Plus 3 for Branch flush

*How many times will this happen for the loop?

* 20 times all these instructions will run

* 20 x 9 =180

- 3 instructions following loop
- Total 2+ 180 + …4
- 4 getting pipeline running = 186

NOW with ID Stage Branch: A) 156 B) 146 C) 148 D) 186 E) 188

REMEMBER TO THINK ABOUT

The ENTIRE Instruction Sequence

CODE REARRANGEMENT

Assume No Flushing Hardware :

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

Can we arrange the code such

That the instruction following the
LW, and BNE,, MUST Execute anyways

Therefore avoiding stalling or an errant instruction…

CODE REARRANGEMENT :

Can we arrange the code such

That the instruction following the LW, and BNE,, MUST Execute anyways

Therefore avoiding stalling or an errant instruction…

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

A) Not Possible

B) IS Possible

C) Only can avoid LW stall not Branch Flush

D) I don't know ☺

CODE REARRANGEMENT : Branch Loops

Can we arrange the code such
That the instruction following the
LW, and BNE,, MUST Execute anyways

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

What instructions in this loop need
To execute ANYWAYS

What instructions in this loop need To execute ANYWAYS

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

What instructions in this loop need
To execute ANYWAYS

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 addi $4, $4, 4
116 add $2, $2, $3
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

Not the Best Choice

What instructions in this loop need To execute ANYWAYS

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 addi $1, $1, -1
116 add $2, $2, $3
120 bne $1, $0, -5
124 addi $4, $4, 4
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

Does this WORK?

What instructions in this loop need
To execute ANYWAYS

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 addi $1, $1, -1
116 add $2, $2, $3
120 bne $1, $0, -5
124 addi $4, $4, 4
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

Does this WORK?

What instructions in this loop need To execute ANYWAYS

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 addi $1, $1, -1
116 add $2, $2, $3
120 bne $1, $0, -4
124 addi $4, $4, 4
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

Does this WORK?

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 addi $1, $1, -1
116 add $2, $2, $3
120 bne $1, $0, -4
124 addi $4, $4, 4
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

TOTAL EXECUTION TIME:

2 + 20x5 +3 + 4

=109 ☺

Good Work!

Exception Handling : FYI

- Slides 6-39 : 6-43 SKIM

Hardware for stall: load use hazard

- **In the diagram for a Hardware Stall: The Control Unit has added control lines going into PC and into IF/ID register**
- **What might these be used for?**

A)Allow no new instruction to be fetched, immediately after a LW

B)Following one instruction after the lw, if there is a data dependency- these control lines

Ensure no new instruction is written to IF/ID and PC is not updated

C)Forcing all the control lines to be zero- thereby implementing a NOP

D)Ensure that the next instruction after LW does not use the $rt register

E)All of the above

FROM EARLIER EXAMPLE

```
100 addi $1, $0, 20
104 addi $2, $0, 0
108 lw $3, 0($4)
112 add $2, $2, $3
116 addi $4, $4, 4
120 addi $1, $1, -1
124 bne $1, $0, -5
128 slt $6, $2, $0
132 add $2, $3,$4
136 sw $2, 100($4)
```

NOW: Forwarding and Stalling and Branching Known in ID stage….

NOW

*2 before loop

*5 in the loop plus 1 for lw stall:

*Plus one for Branch: 7

*How many times will this happen for the loop?

* 20 times all these instructions will run
* 20 x 7 =140

- 2 instructions following loop added
- Total 2+ 140 + 2 +4
- = 148