HW6.6. Pipelining hazards

Feel free to check out the guide that we have prepared to help you in this problem.

For this question, assume that we are working with the a 5-stage pipelined CPU, composed of Instruction Fetch, Instruction Decode/Register Read, ALU/Execution, Memory/Branch, and Writeback.

Note that in this question, **we assume that branches are resolved in the memory stage.** In other words, the branch target address becomes available at the input of the PC register during the memory stage.

We run the following code:

```
1. lw s1 0(s0)
2. add s2 s1 s1
3. slli s3 s1 1
4. bne s2 s3 exit #Note that this branch can never be taken
5. addi t1 t0 x0
6. exit:
```

For each pair of instructions, determine:

a: How many stalls that are needed in a **completely unoptimized** pipeline: Reg file does not "write-then-read" on the same cycle (i.e. no 'double pumping'), no forwarding, no branch prediction or pipeline flushing, etc.

b: The type of hazard that causes the stall

c: The pipeline optimization needed to best reduce the number of stall cycles required.

d: The number of stall cycles needed after all optimizations in part c are implemented.

Instructions	Unoptimized Stalls Needed	Hazard Type	Optimization Needed	Optimized Stalls Needed
1->2	3 ✓ 100%	Structural/Data Hazard ✓ 100%	Data Forwarding ✓ 100%	1 ✓ 100%
2->3	0 ✓ 100%	No Hazard ✓ 100%	No Hazard ✓ 100%	0 ✓ 100%
3->4	3 ✓ 100%	Structural/Data Hazard ✓ 100%	Data Forwarding ✓ 100%	0 ✓ 100%
4->5	3 ✓ 100%	Control Hazard ✓ 100%	Branch Prediction ✓ 100%	0 ✓ 100%

Now, assume that the register file supports "write-then-read" functionality on the same clock cycle and that branch target addresses become available at the input of the PC register during the execution stage. Which of the following statements will be true?

- [ ] (a) All the answers from the previous question remain the same.
- [ ] (b) Required number of optimized stalls for structural/data hazard decreases by 1 cycle.
- [ ] (c) Required number of optimized stalls for control hazard decreases by 1 cycle.
- [ ] (d) Some of the hazards from the previous question will disappear.
- [x] (e) Required number of unoptimized stalls for control hazard decreases by 1 cycle. ✓
- [x] (f) Required number of unoptimized stalls for structural/data hazard decreases by 1 cycle. ✓

Select all possible options that apply.

✓ 100%

Try a new variant

Homework 6

Assessment overview

Total points: 100/100

Score: 100%

Question

Value: 15

History: 15

Awarded points: 15/15

Report an error in this question

Previous question

Next question

Attached files

No attached files

Attach a file

Attach text

Correct answer

Instructions	Unoptimized Stalls Needed	Hazard Type	Optimization Needed	Optimized Stalls Needed
1->2	3	`Structural/Data Hazard`	`Data Forwarding`	1
2->3	0	`No Hazard`	`No Hazard`	0
3->4	3	`Structural/Data Hazard`	`Data Forwarding`	0
4->5	3	`Control Hazard`	`Branch Prediction`	0

1-2: There's a data hazard with s1; we need to move the ID stage of line 2 to after the WB stage of line 1, for 3 stalls. With forwarding, we can get the results at MEM and send it to Line 2's EX directly, so that can be done with one stall.

2-3: There's no hazard here, since we don't use the written register from line 2. Note that we don't have to wait for line 1, because the stalls from line 2 forced us to wait here long enough; the stalls from line 2 moved all future instructions one cycle back, and we don't count those as stalls caused by this line.

3-4: A similar data hazard as with 1-2, but this time, we get the result at the end of EX, so we need 0 stalls with forwarding.

4-5: With absolutely no optimizations, we have no choice but to stall IF of line 5 until after we run the MEM stage and determine that we don't need to take a branch. With branch prediction, we can infer that we won't take the branch (it is impossible for us to take the branch, given the previous instructions), so we need no stalls.

Now, assume that the register file supports "write-then-read" functionality on the same clock cycle and that branch target addresses become available at the input of the PC register during the execution stage. Which of the following statements will be true?

(e) Required number of unoptimized stalls for control hazard decreases by 1 cycle.
(f) Required number of unoptimized stalls for structural/data hazard decreases by 1 cycle.
Required number of unoptimized stalls for structural/data hazard decreases by 1 cycle. This is true. If the register file supports "write-then-read" on the same cycle, then we can align the WB stage of the first instruction to the ID stage of the second (data dependent) instruction. Without the "write-then-read" functionality, we have to wait until after the WB stage of the first instruction to start the ID stage of the second (data dependent) instruction.

Required number of unoptimized stalls for control hazard decreases by 1 cycle. This is true. If the branch target address becomes available at the input of the PC register durign the execution stage, then the IF stage of the next instruction can start right after the EX stage of the branch instruction. Compare this to the previous setup where the address becomes available during the MEM stage, which is one more cycle after EX.

Required number of optimized stalls for structural/data hazard decreases by 1 cycle. This is false. Data forwarding already makes the required number of stalls to 0, except for load instruction data dependency. We cannot remove the 1 cycle stall for this hazard since the result only becomes available at the MEM stage, which is 1 cycle after the EX stage (where the result of most instructions become available).

Required number of optimized stalls for control hazard decreases by 1 cycle. This is false. With perfect branch prediction, we don't incur stalls anymore. Thus, we cannot further decrease that.

Some of the hazards from the previous question will disappear. This is false. The hazards will still be there since the data dependency is still present. Without data forwarding or branch prediction, we will still need to stall.

All the answers from the previous question remain the same. This is false. Statements 1 and 2 are true, so there will be a change in the answers from the previous question.

Submitted answer **correct: 100%**
Submitted at 2022-10-23 08:38:29 (PDT)

hide

Instructions	Unoptimized Stalls Needed	Hazard Type	Optimization Needed	Optimized Stalls Needed
1->2	3 ✓ 100%	`Structural/Data Hazard` ✓ 100%	`Data Forwarding` ✓ 100%	1 ✓ 100%
2->3	0 ✓ 100%	`No Hazard` ✓ 100%	`No Hazard` ✓ 100%	0 ✓ 100%
3->4	3 ✓ 100%	`Structural/Data Hazard` ✓ 100%	`Data Forwarding` ✓ 100%	0 ✓ 100%
4->5	3 ✓ 100%	`Control Hazard` ✓ 100%	`Branch Prediction` ✓ 100%	0 ✓ 100%

Now, assume that the register file supports "write-then-read" functionality on the same clock cycle and that branch target addresses become available at the input of the PC register during the execution stage. Which of the following statements will be true?

(e) Required number of unoptimized stalls for control hazard decreases by 1 cycle. ✓
(f) Required number of unoptimized stalls for structural/data hazard decreases by 1 cycle. ✓
✓ 100%