CS M151B Homework 5

Charles Zhang

February 6, 2022

Problem 4.16

In this exercise, we examine how pipelining affects the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies:

`IF`	`ID`	`EX`	`MEM`	`WB`
250ps	350ps	150ps	300ps	200ps

Also, assume that instructions executed by the processor are broken down as follows:

ALU/Logic	Jump/Branch	Load	Store
45%	20%	20%	15%

a) What is the clock cycle time in a pipelined and non-pipelined processor?

b) What is the total latency of an `lw` instruction in a pipelined and non-pipelined processor?

c) If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor?

d) Assuming there are no stalls or hazards, what is the utilization of the data memory?

e) Assuming there are no stalls or hazards, what is the utilization of the write-register port of the `Registers` unit?

Problem 4.27

Problems in this exercise refer to the following sequence of instructions, and assume that it is executed on a five-stage pipelined datapath:

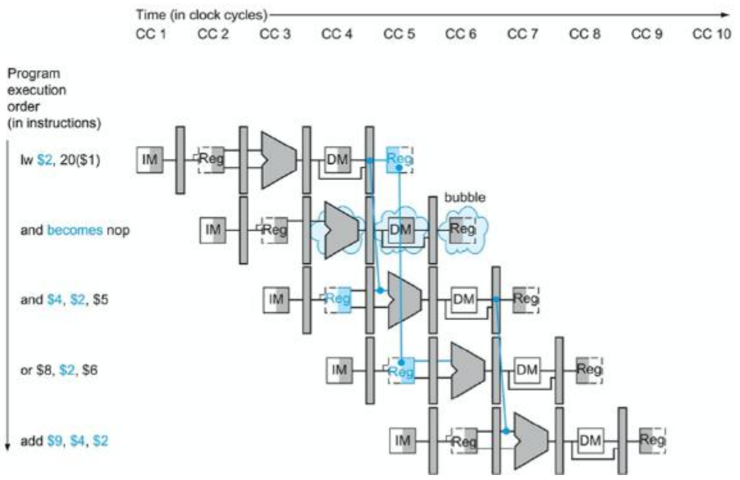
```
add $s3, $s1, $s0
lw $s2, 4($s3)
lw $s1, 0($s4)
or  $s2, $s3, $s2
sw $s2, 0($s3)
```

a) If there is no forwarding or hazard detection, insert `NOP`s to ensure correct execution.

b) Now, change and/or rearrange the code to minimize the number of `NOP`s needed. You can assume register `$t0` can be used to hold temporary values in your modified code.

c) If the processor has forwarding, but we forgot to implement the hazard detection unit, what happens when the original code executes?

d) If there is forwarding, for the first seven cycles during the execution of this code, specify which signals are asserted in each cycle by hazard detection and forwarding units in the following figure:



e) If there is no forwarding, what new input and output signals do we need for the hazard detection unit depicted above? Using this instruction sequence as an example, explain why each signal is needed.

f) For a new hazard detection unit using full forwarding, specify which output signals it asserts in each of the first five cycles during the execution of this code.

Problem 4.28

The importance of having a good branch predictor depends on how often conditional branches are executed. Together with branch predictor accuracy, this will determine how much time is spent stalling due to mispredicted branches. In this exercise, assume that the breakdown of dynamic instructions into various instruction categories is as follows:

R-type	`beqz/bnez`	`jal`	`lw`	`sw`
40%	25%	5%	25%	5%

Also, assume the following branch predictor accuracies:

Always-Taken	Always-Not-Taken	2-Bit
45%	55%	85%

a) Stall cycles due to mispredicted branches increase the CPI. What is the extra CPI due to mispredicted branches with the always-taken predictor? Assume that branch outcomes are determined in the `ID` stage and applied in the `EX` stage that there are no data hazards, and that no delay slots are used.

b) Repeat a) for the "always-not-taken" predictor.

c) Repeat a) for the 2-bit predictor.

d) With the 2-bit predictor, what speedup would be achieved if we could convert half of the branch instructions to some ALU instruction? Assume that correctly and incorrectly predicted instructions have the same chance of being replaced.

e) With the 2-bit predictor, what speedup would be achieved if we could convert half of the branch instructions in a way that replaced each branch instruction with two ALU instructions? Assume that correctly and incorrectly predicted instructions have the same chance of being replaced.

f) Some branch instructions are much more predictable than others. If we know that 80% of all executed branch instructions are easy-to-predict loop-back branches that are always predicted correctly, what is the accuracy of the 2-bit predictor on the remaining 20% of the branch instructions?