ECE 552 Lab 1 Report

Francis Guo 1005931397 Rongbo Zhang 1005273393

1) The ideal performance drop from the CPI will be:

sim_num_stall_one_cyc: the number of 1-cycle stalls due to data dependence.
sim_num_stall_two_cyc: the number of 2-cycle stalls due to data dependence.
sim_num_insn: total instruction count for the testbench
CPI = 1 + (sim_num_stall_one_cyc / sim_num_insn)*1 + (sim_num_stall_two_cyc / sim_num_insn) *2

Q1)

slowdown = (CPI from RAW hazard - idea CPI) / idea CPI = (1.6642 - 1) / 1 = 66.42%

Q2)

slowdown = (CPI from RAW hazard - idea CPI) / idea CPI = (1.3903 - 1) /1 = 39.03 %

2) For the compile flag, we used the command:
/cad2/ece552f/compiler/bin/ssbig-na-sstrix-gcc mbq1.c -O2 -o mbq1

For our microtestbench, we make a loop to cycle 10^5 for the test code to avoid the interference of the RAW hazard from the header file. And from the .S file, we could pull out the assembly code. Here we have hand analysis the number of hazard counts to compare with the result of simulation.

From the result of simulation we have:

```
sim_num_stall_one_cyc_q1      100080 # total number of one cycle stall (q1)
sim_num_stall_two_cyc_q1      500851 # total number of two cycle stall (q1)
sim_num_stall_one_cyc_q2      200768 # total number of one cycle stall (q2)
sim_num_stall_two_cyc_q2      100091 # total number of two cycle stall (q2)
```

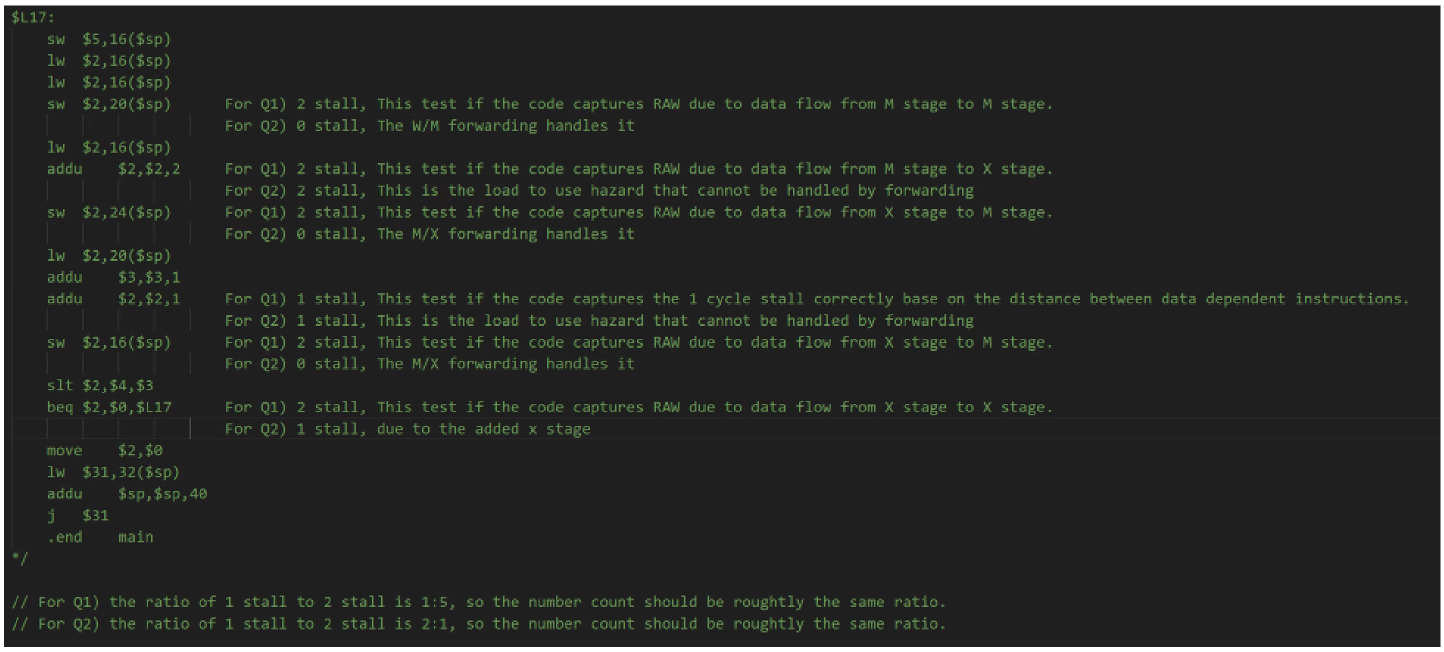
This means we have for 5-stage simple processor per single loop:

1 cycle stall count: 1
2 cycle stall count: 2

for 6-stage fully forwarded processor per single loop:

1 cycle stall count: 2
2 cycle stall count: 1

And this match the analysis from the assembly code (ignore the hazard created by the headers):



```
$L17:
    sw  $5,16($sp)
    lw  $2,16($sp)
    lw  $2,16($sp)
    sw  $2,20($sp)      For Q1) 2 stall, This test if the code captures RAW due to data flow from M stage to M stage.
                        For Q2) 0 stall, The W/M forwarding handles it
    lw  $2,16($sp)
    addu    $2,$2,2     For Q1) 2 stall, This test if the code captures RAW due to data flow from M stage to X stage.
                        For Q2) 2 stall, This is the load to use hazard that cannot be handled by forwarding
    sw  $2,24($sp)      For Q1) 2 stall, This test if the code captures RAW due to data flow from X stage to M stage.
                        For Q2) 0 stall, The M/X forwarding handles it
    lw  $2,20($sp)
    addu    $3,$3,1
    addu    $2,$2,1     For Q1) 1 stall, This test if the code captures the 1 cycle stall correctly base on the distance between data dependent instructions.
                        For Q2) 1 stall, This is the load to use hazard that cannot be handled by forwarding
    sw  $2,16($sp)      For Q1) 2 stall, This test if the code captures RAW due to data flow from X stage to M stage.
                        For Q2) 0 stall, The M/X forwarding handles it
    slt $2,$4,$3
    beq $2,$0,$L17      For Q1) 2 stall, This test if the code captures RAW due to data flow from X stage to X stage.
                        For Q2) 1 stall, due to the added x stage
    move    $2,$0
    lw  $31,32($sp)
    addu    $sp,$sp,40
    j   $31
    .end    main
*/

// For Q1) the ratio of 1 stall to 2 stall is 1:5, so the number count should be roughly the same ratio.
// For Q2) the ratio of 1 stall to 2 stall is 2:1, so the number count should be roughly the same ratio.
```