ECE552 LAB1 Report

Zexi Pan 1000138246
Xiaoyang Guo 999578513

CPI Performance:

	Question 1	Question 2
CPI	1.645	1.388
CPI performace drop against CPI=1	64.5%	38.8%

Calculations:

The CPI is calculated with the following formula:

$$\text{CPI} = \frac{SimNumInsn + numDoubleCycleStall * 2 + numSingleCycleStall}{SimNumInsn}$$

CPI is equal to the number of cycles per instruction. Since both the top and bottom have the same number of instructions, we only need to consider the number of cycles. The number of cycles is equal to the number of instructions + number of stalls + pipline stages (4 in q1, although this is negligible because it is so small).

There is also overhead from other sources such as initialization in the microbenchmark and trace, however, it can be reduced with enough iterations and instructions.

Microbenchmark:

Our microbenchmark consists of one for loop that gives a large amount of instructions with a controlled number of data hazards. Here are the hazard scenarios we took into consideration:

Read After Write(double cycle)

	c1	c2	c3	c4	c5	c6	c7	c8
add **r1** r2 r3	F	D	Ex	Mem	W			
add r4 **r1** r2		F	d*	d*	D	Ex	Mem	W

Read After Write(Single cycle)

	c1	c2	c3	c4	c5	c6	c7	c8
add **r1** r2 r3	F	D	Ex	Mem	W			
add r4 r5 r2		F	D	Ex	Mem	W		
add r6 **r1** r7			F	d*	D	Ex	Mem	W

Inside the loop we placed two double cycle stalls and one single cycle stall. The loop currently runs for 30000 iterations. The results we got when running sim-safe were:

SingleCycle_q1 11330 # single cycle stalls (q1)
DoubleCycle_q1 34006 # double cycle stalls (q1)

When checking the assembly code we noticed that the for loop itself causes another double cycle stall for the incrementing and comparison of the index. This would mean there are three doucle cycle stalls and one single cycle stall in our microbenchmark and that aligns with the results we are seeing. The results could be improved by running more iterations as there seems to be a fair amount of overhead we are seeing (4000 for double cycle and 1000 for single cycle).