ECE552, Fall 2014

Lab 1 Report

Timmy Rong Tian Tse (998182657)
Brian Aguirre (998528213)

Using the `/cad2/ece552f/benchmarks/gcc.eio` benchmark, our group received a CPI of 1.6669 for question 1 (an increase of 66.69 percent) and a CPI of 1.3910 for question 2 (an increase of 39.10 percent). In obtaining these numbers, we used the following mathematical derivations:

$$CPI = 1 + 1 \times \left(\frac{x_1}{N}\right) + 2 \times \left(\frac{x_2}{N}\right) + \cdots + n \times \left(\frac{x_n}{N}\right)$$

where n is the number of stalls, x_n is the number of instructions that contain a hazard requiring n number of stalls to resolve and finally, N is the total number of instructions. All of these parameters are relative to one single program.

In question 1 with the 5-stage pipeline with no forwarding, we have concluded that the only real hazards that we need to take into consideration is the RAW hazards. However, depending on when the dependent register is used, a RAW hazard can be resolved either with two stalls or one stall. The two stalls case occurs when the dependent register is immediately read from after it was written to. Since the dependent instruction needs to decode on the cycle that the previous instruction writes back, the dependent instruction needs to wait two cycles. On the other hand, the one stall case occurs when the dependent register is used two instructions later. Since the dependent instruction arrives two cycles later, it only needs to stall one more cycle to resolve the hazard. Because there is no forwarding and bypassing, we can ignore the first corner case that was mentioned in Lab 0. Using this methodology, we received 9955637 one stall instructions, 88182314 two stall instructions and finally, 279373007 total instructions. Using these statistics, we computed CPI as follows:

$$CPI = 1 + 1 \times \left(\frac{9955637}{279373007}\right) + 2 \times \left(\frac{88182314}{279373007}\right) \approx 1.6669$$

In question 2 with the 6-stage pipeline with two cycle executes and full bypassing, the real hazards remain to be RAW hazards. Due to full bypassing, a RAW hazard now requires only one stall to resolve with LTU being an exception. In a LTU, if the loaded value is immediately read from, then it requires two stalls to resolve otherwise if there is one non-dependent instruction in between, only one stall is required. We received 68978653 one stall instructions and 20126394 two stall instructions. CPI is as follows:

$$CPI = 1 + 1 \times \left(\frac{68978653}{279373007}\right) + 2 \times \left(\frac{20126394}{279373007}\right) \approx 1.3910$$

In our microbenchmark code, we inlined assembly code into our C program to directly test the validity of our code. We used the `-O0` flag in our compilation with the hopes that our logic will not be optimized away by the compiler. In assembly code, we have three loops each with 30000 iterations that directly generate a two stall RAW hazards. Since we looped this three times, we would expect the result to be 3*30000 = 90000 two stall hazards in our output and indeed, that is what out output reflected. Similarly, we have two loops in our assembly code that generate a one stall RAW hazards. Again, our output reflected the expected 2*30000 = 60000 one stall hazards. Note that we ensured the total number of one stall hazards were different from the total number of two stall hazards for the reason of differentiating between the two. In our verification, we also changed the loop counters for testing and again, our code output reflected this change.