**Post Lab 8: Introduction to Sequential Logic**

Faizan Bangash

TA : Saurav Kumar Sahu

ECEN 248 – 302

July 19, 2018.

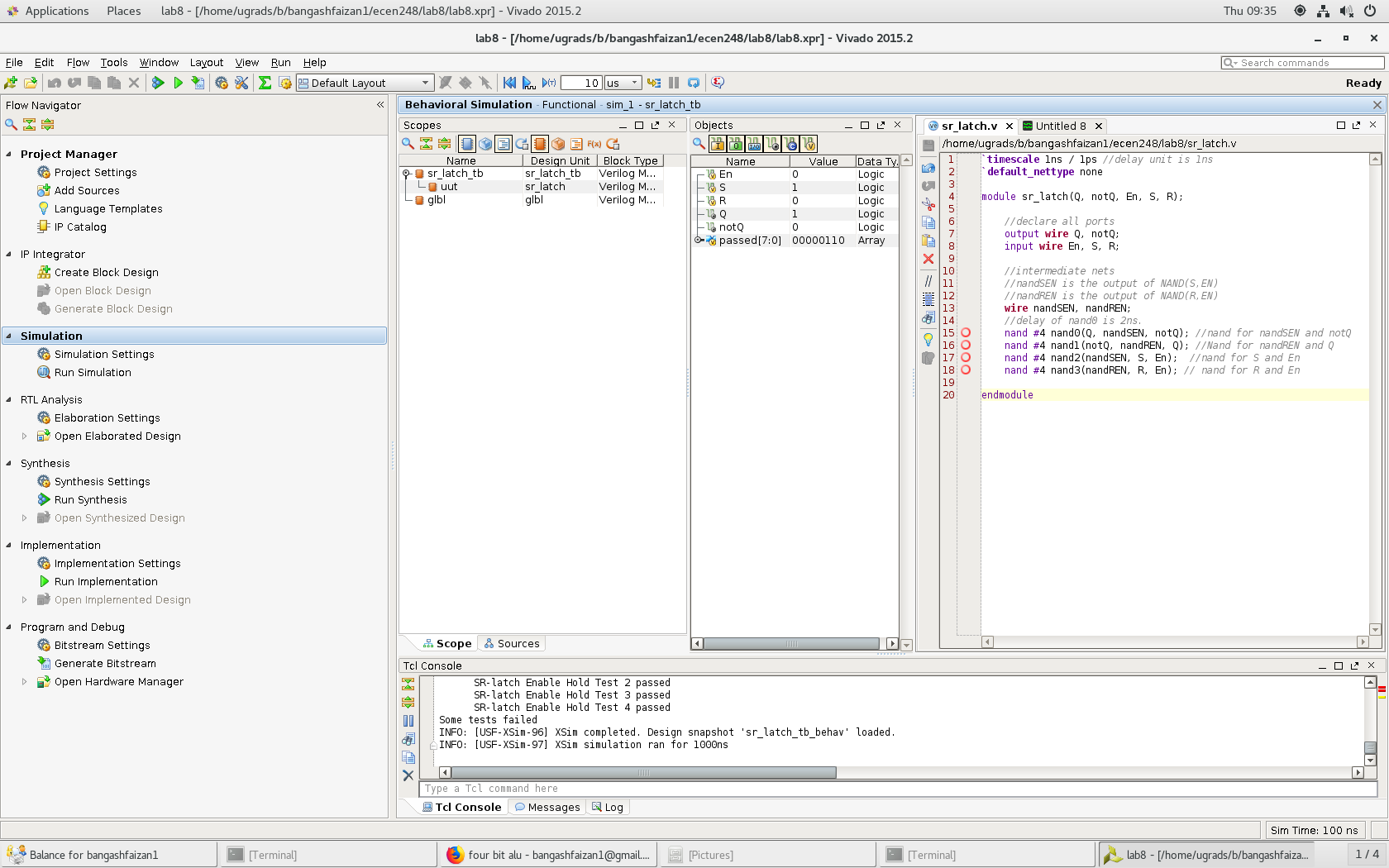
**Objectives** –

This lab introduces sequential flip-flop circuits and how to implement them. The main point of these circuits is to maintain memory. In sequential circuits, the output is dependent not only on the memory but the clock state. Additionally, delays are added to observe what would happen. This is all seen through D-Latch, SR-Latch, and Flip-Flops.

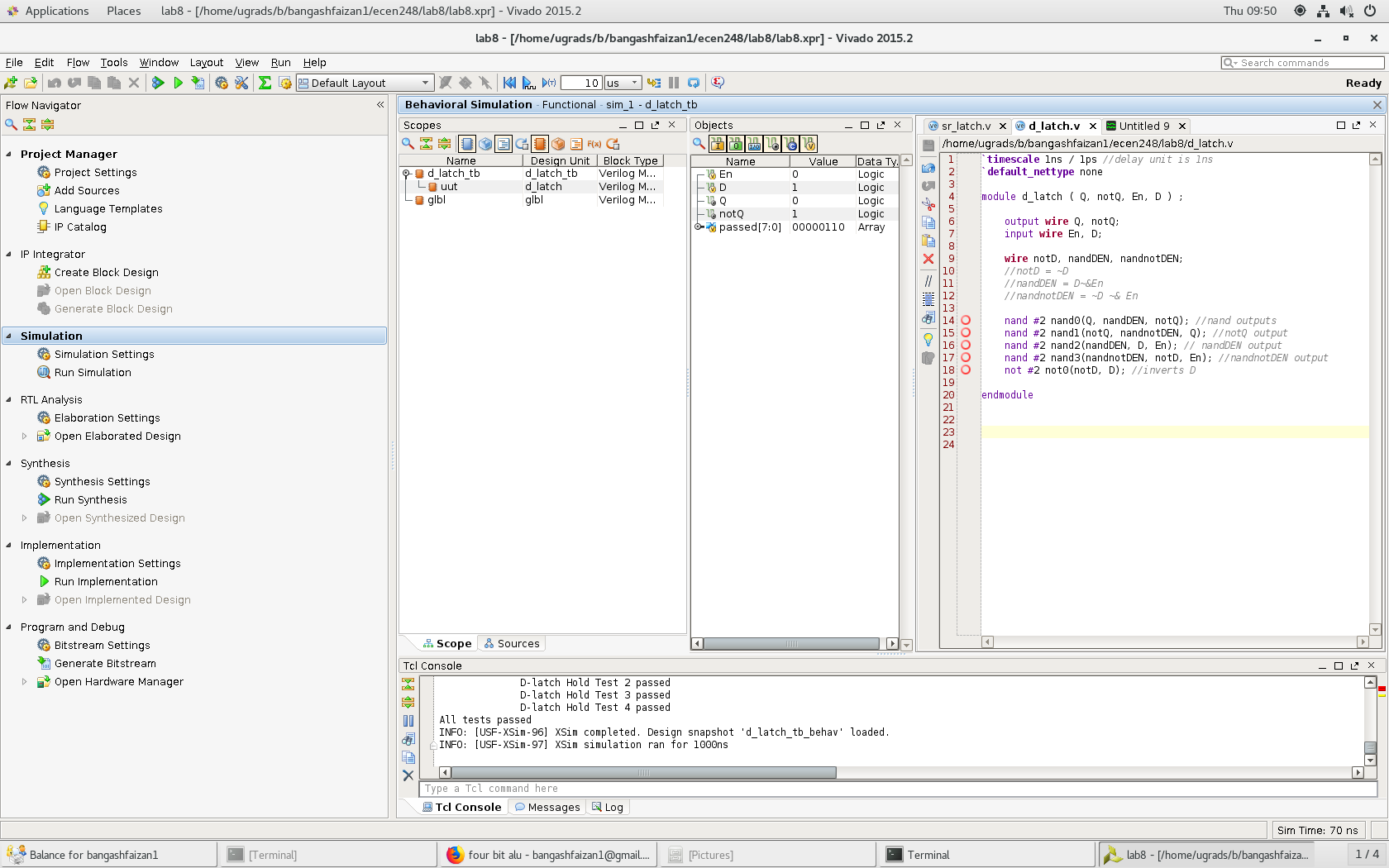
**Design** –

Part 1:

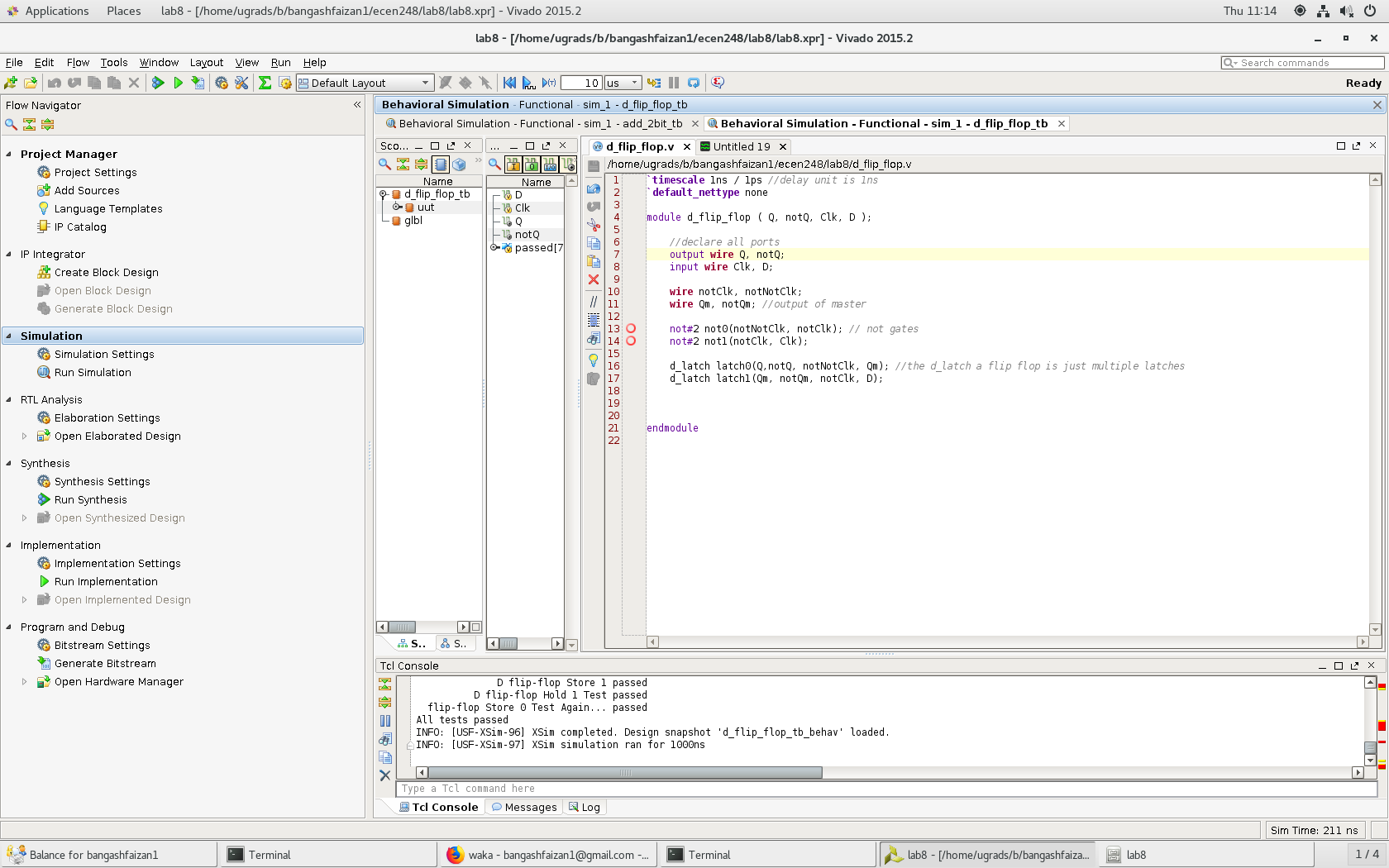
In Experiment 1, I used an SR Latch, a D latch, and a D Flip-Flop all coded in Verilog. The code is as follows :



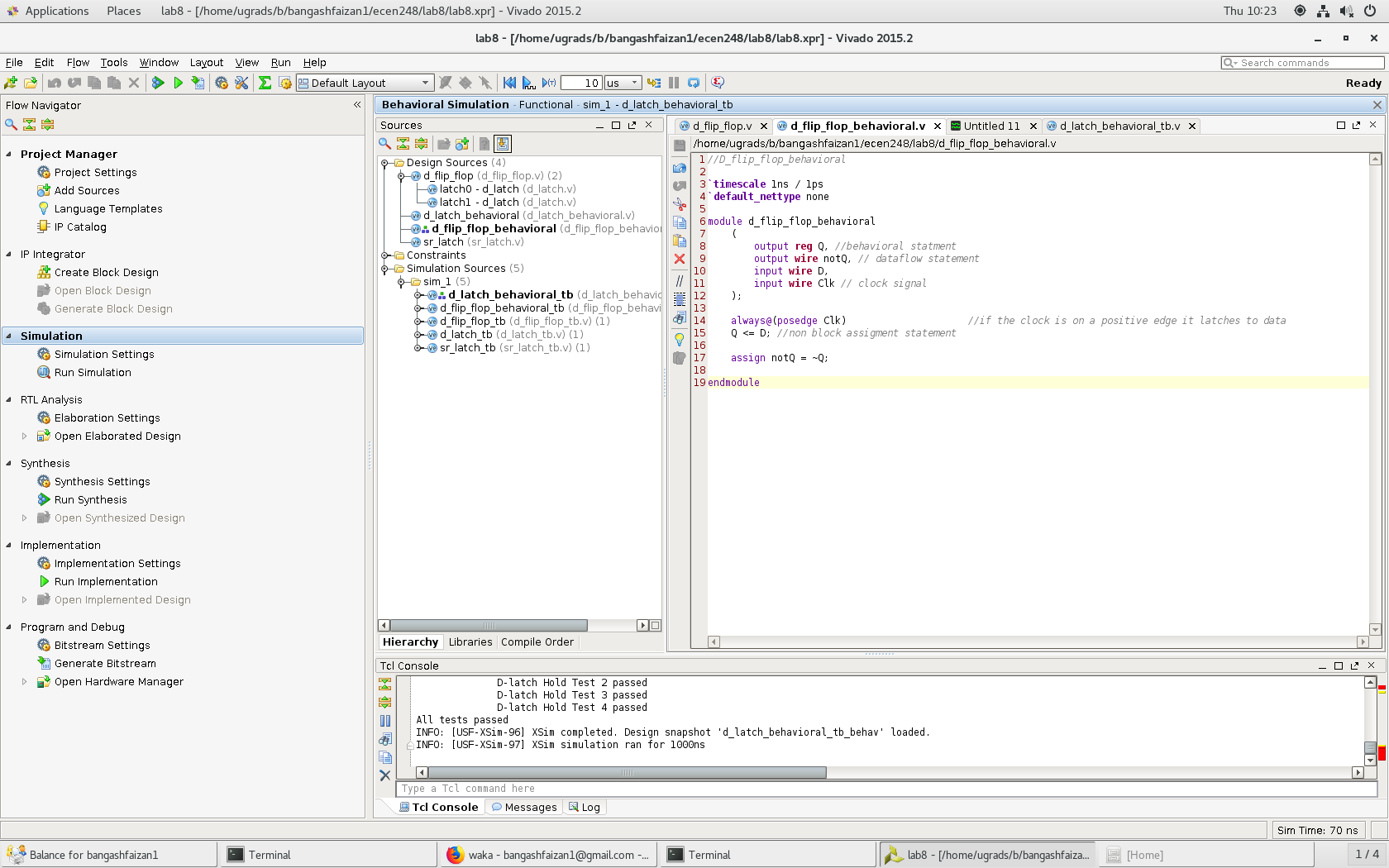
Code for SR-Latch



Code for D-Latch



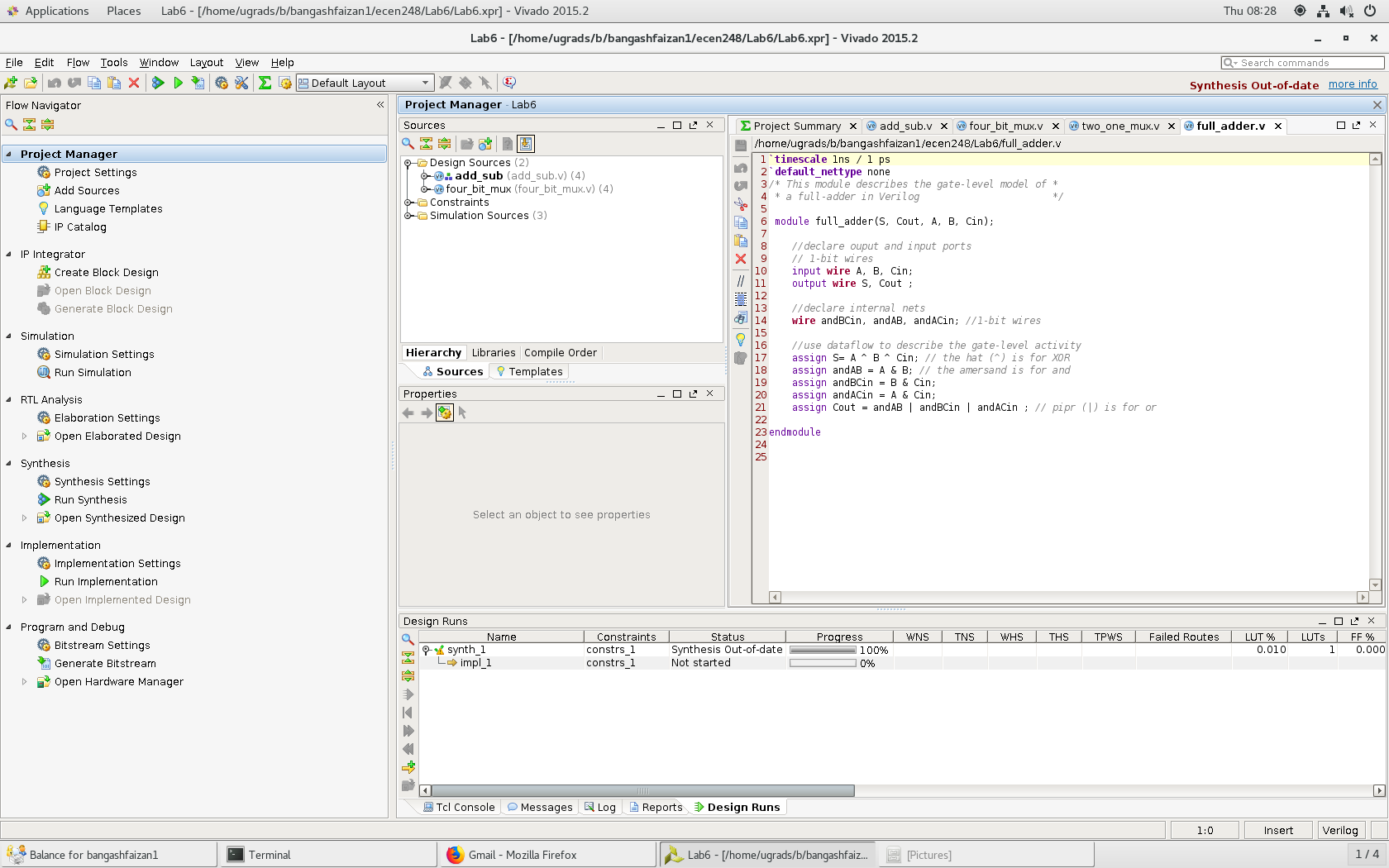
Code for D Flip Flop



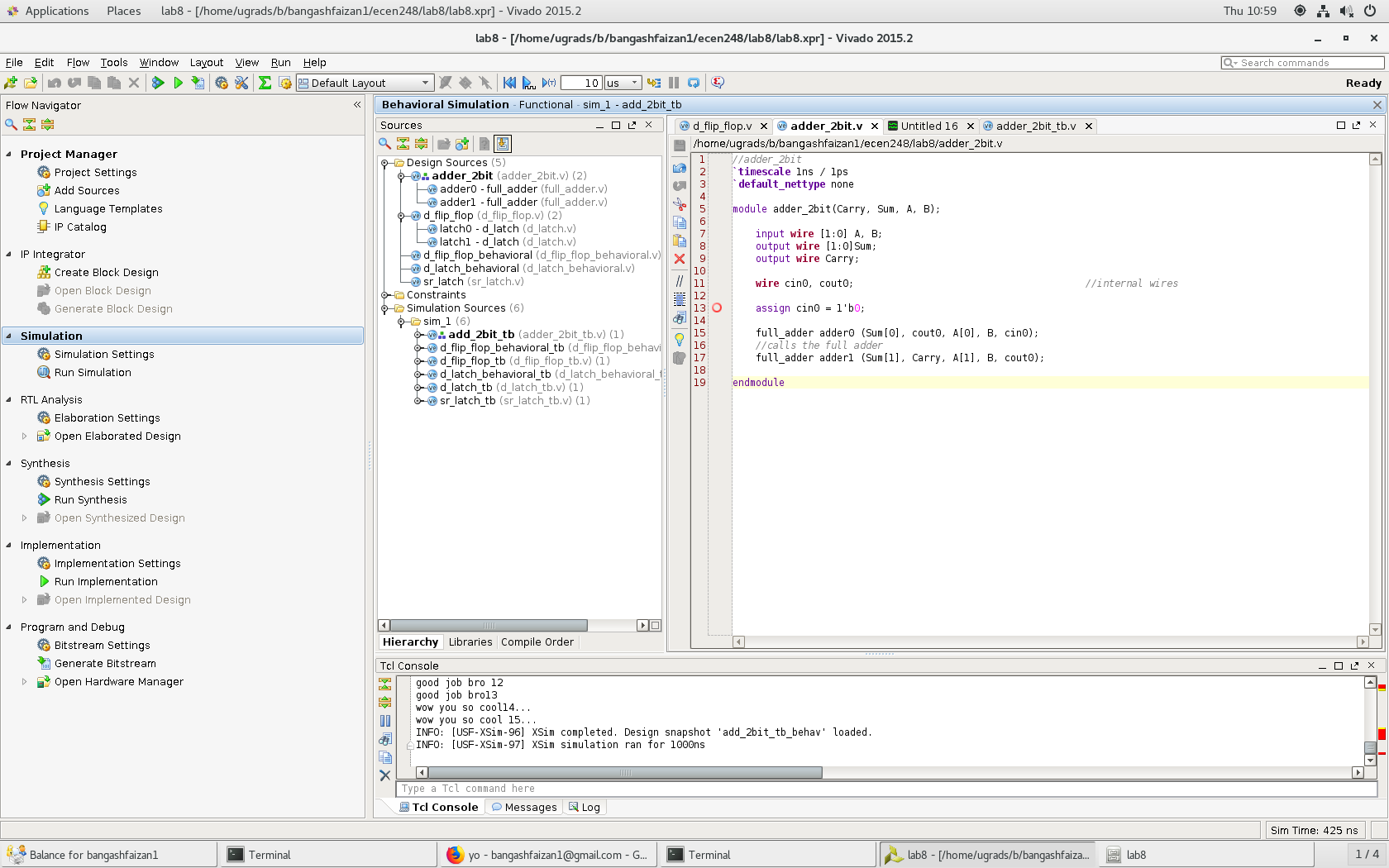
Code for D-Flip-Flop Behavioral

Part 2:

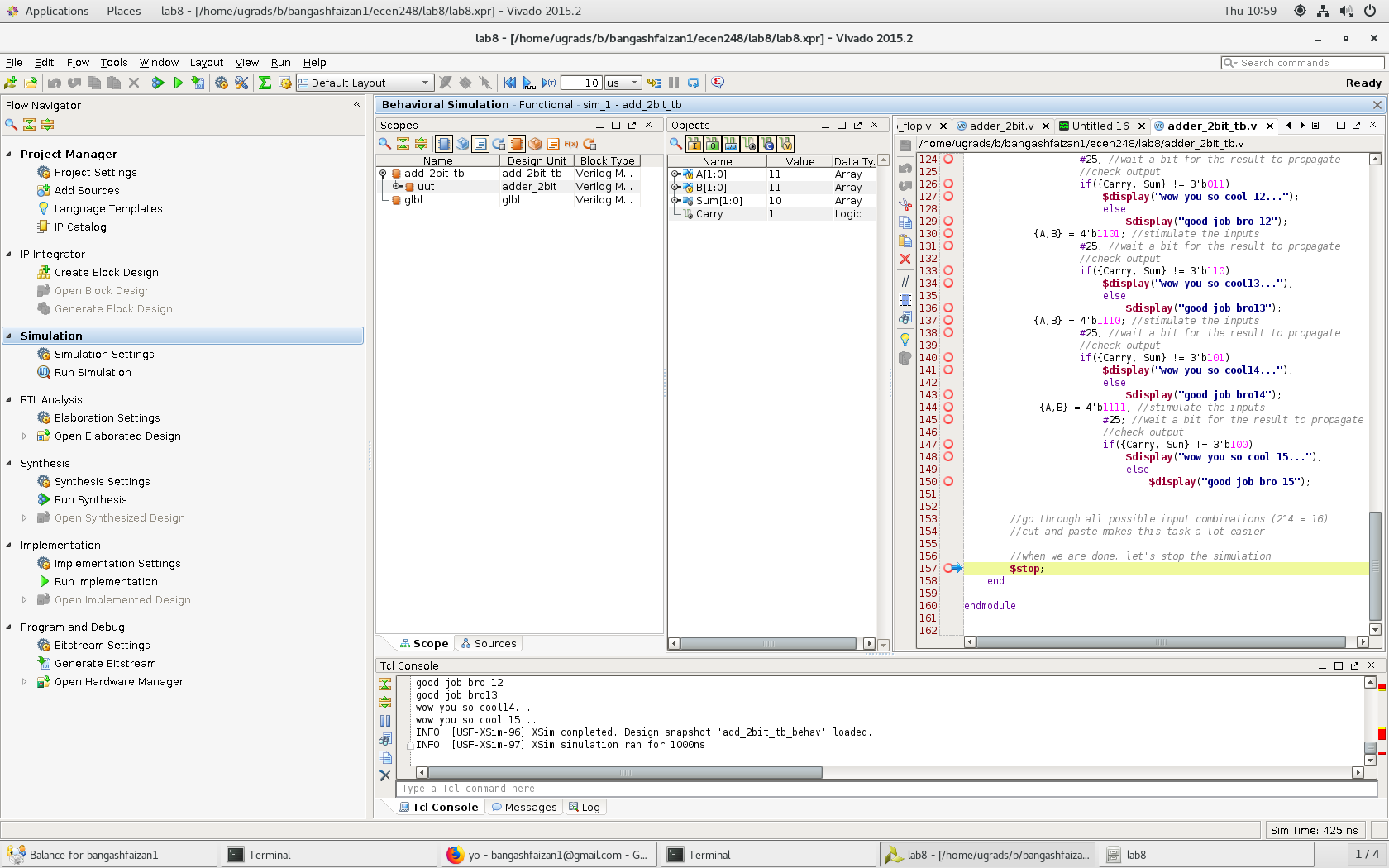
For part 2, we combine the old full adder code from previous labs as well as a new adder 2 bit and combine them to create a Synchronous adder. This adder is similar to a ripple adder however it acts as a Synchronous process so it depends on clock signals and delays. The code for the following files and the created test bench as well as their results can be seen in the following figures.



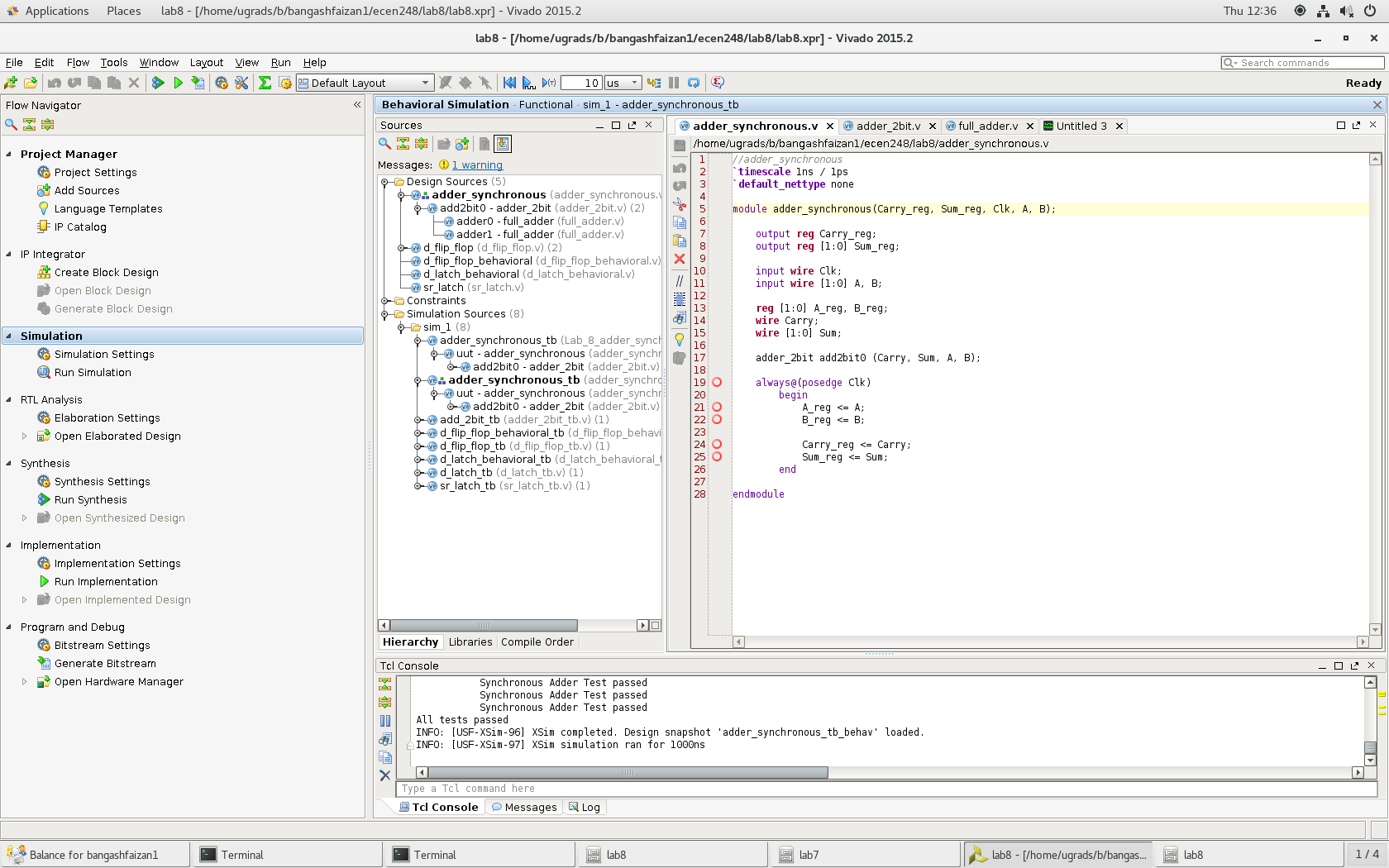
Code for Full Adder



Code for Adder 2 bit



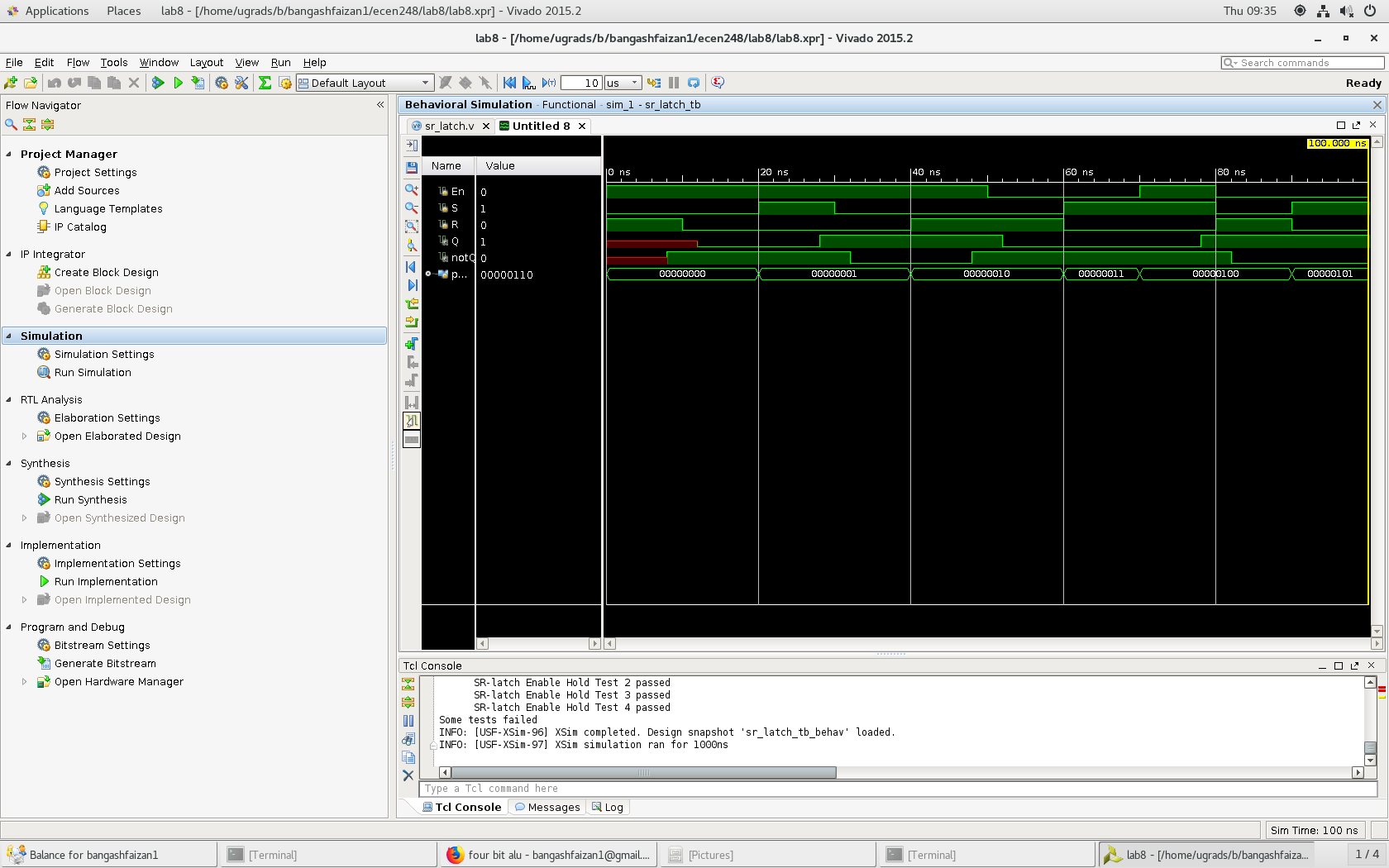
Code for add\_2bit\_tb



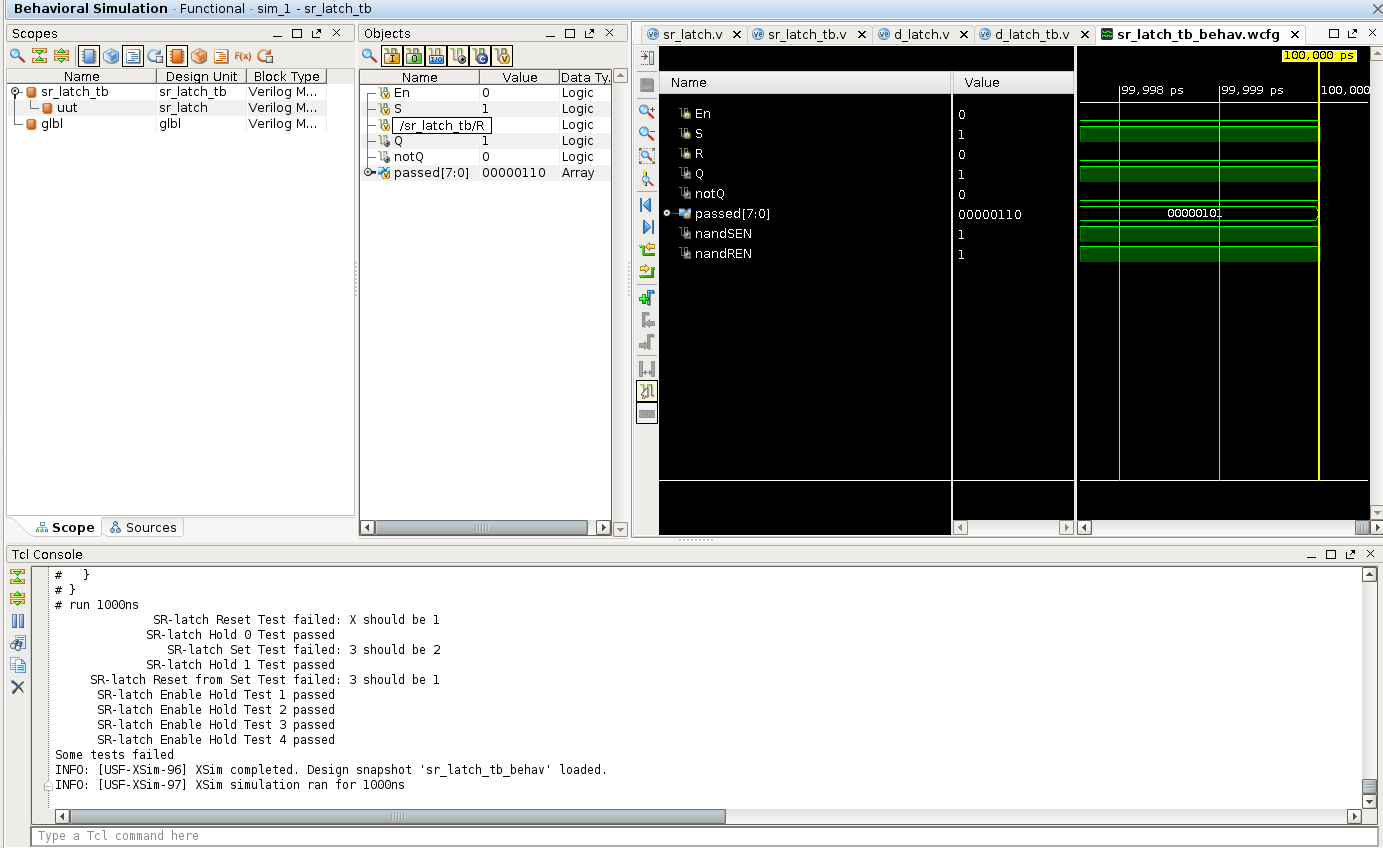
Adder Synchronous code

**Results –**

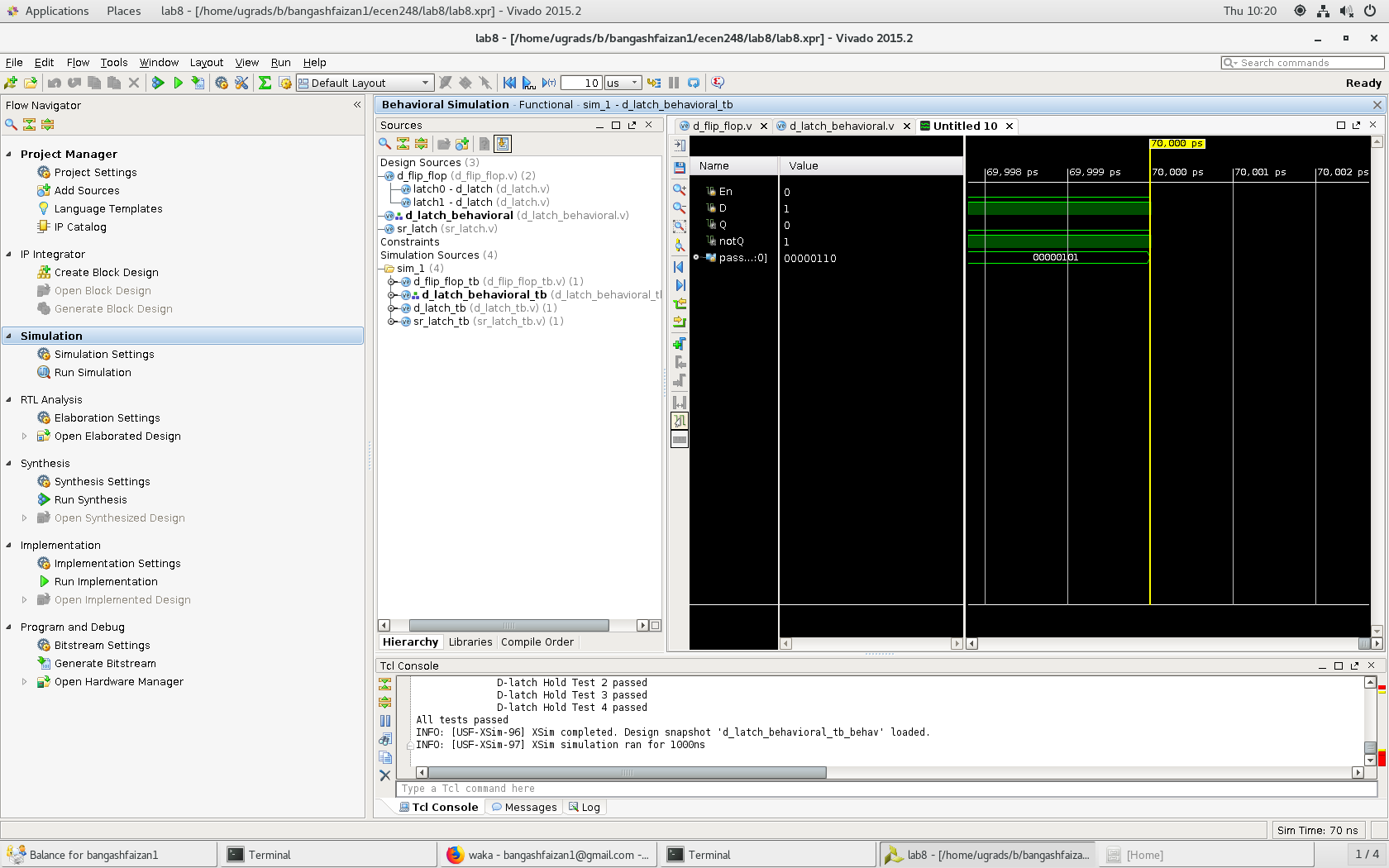
Results of experiment are as follows –



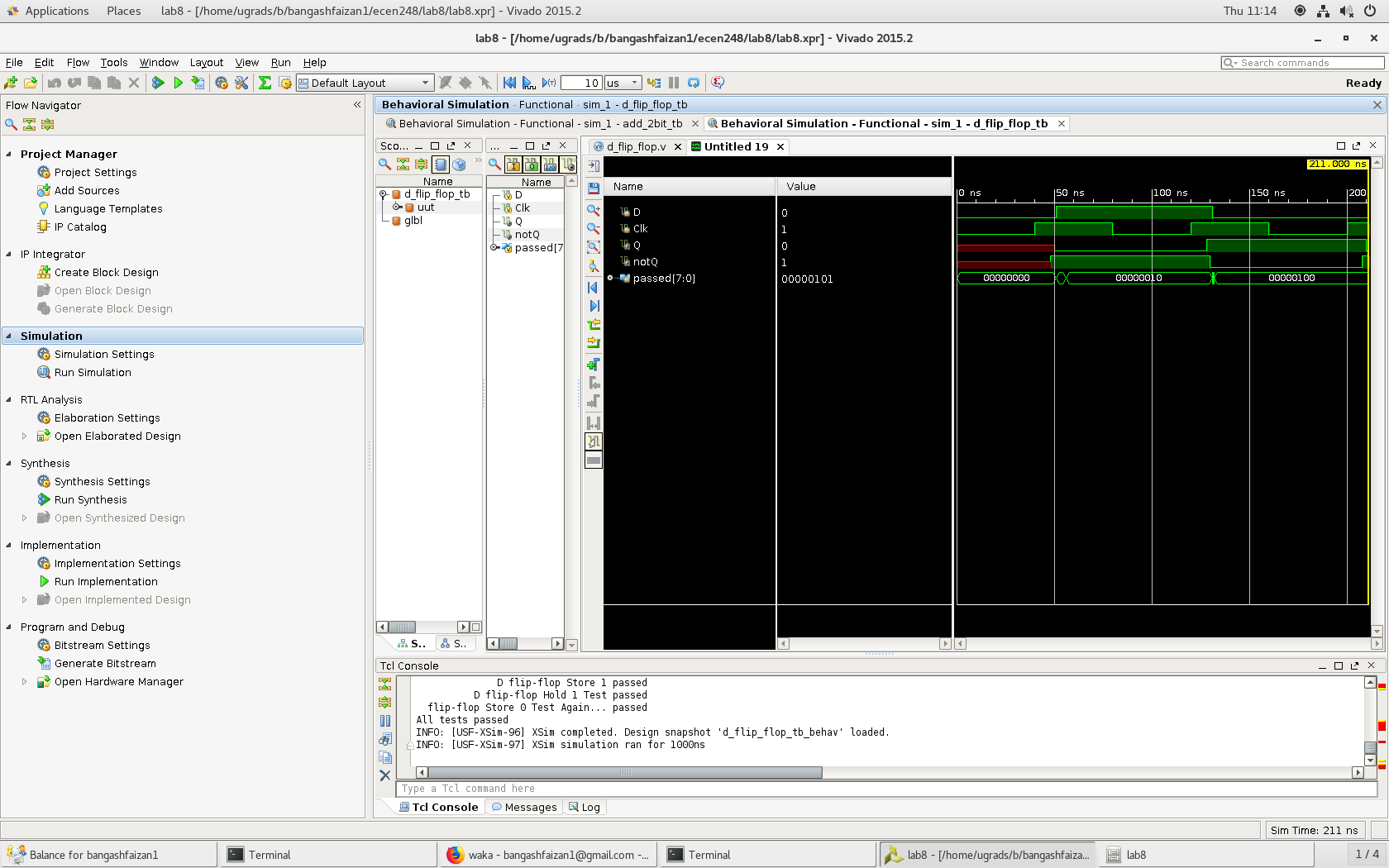
Results for SR-Latch on regular 2 ns delay



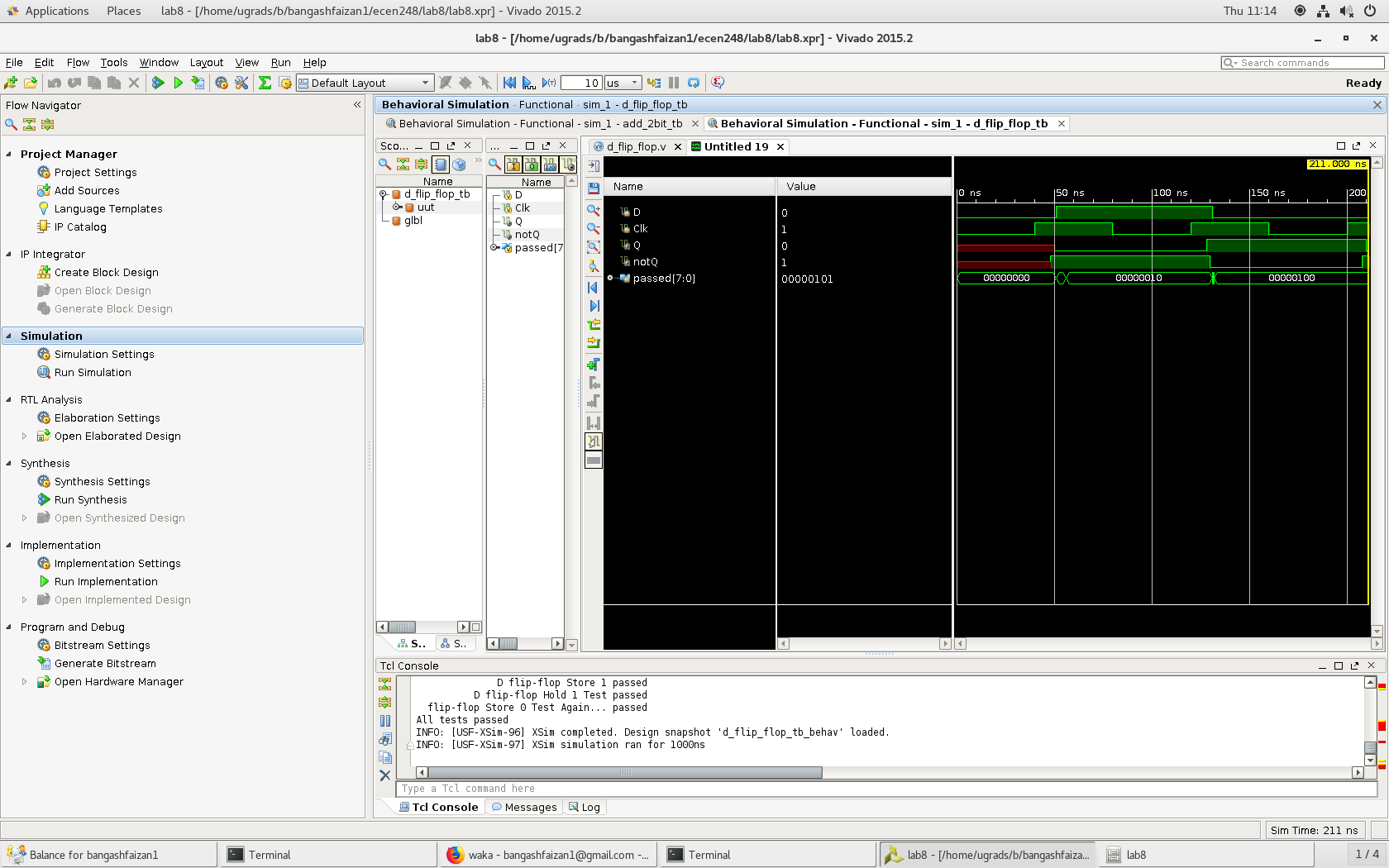
Results for SR-Latch on updated 4 ns delay



Results for D-latch

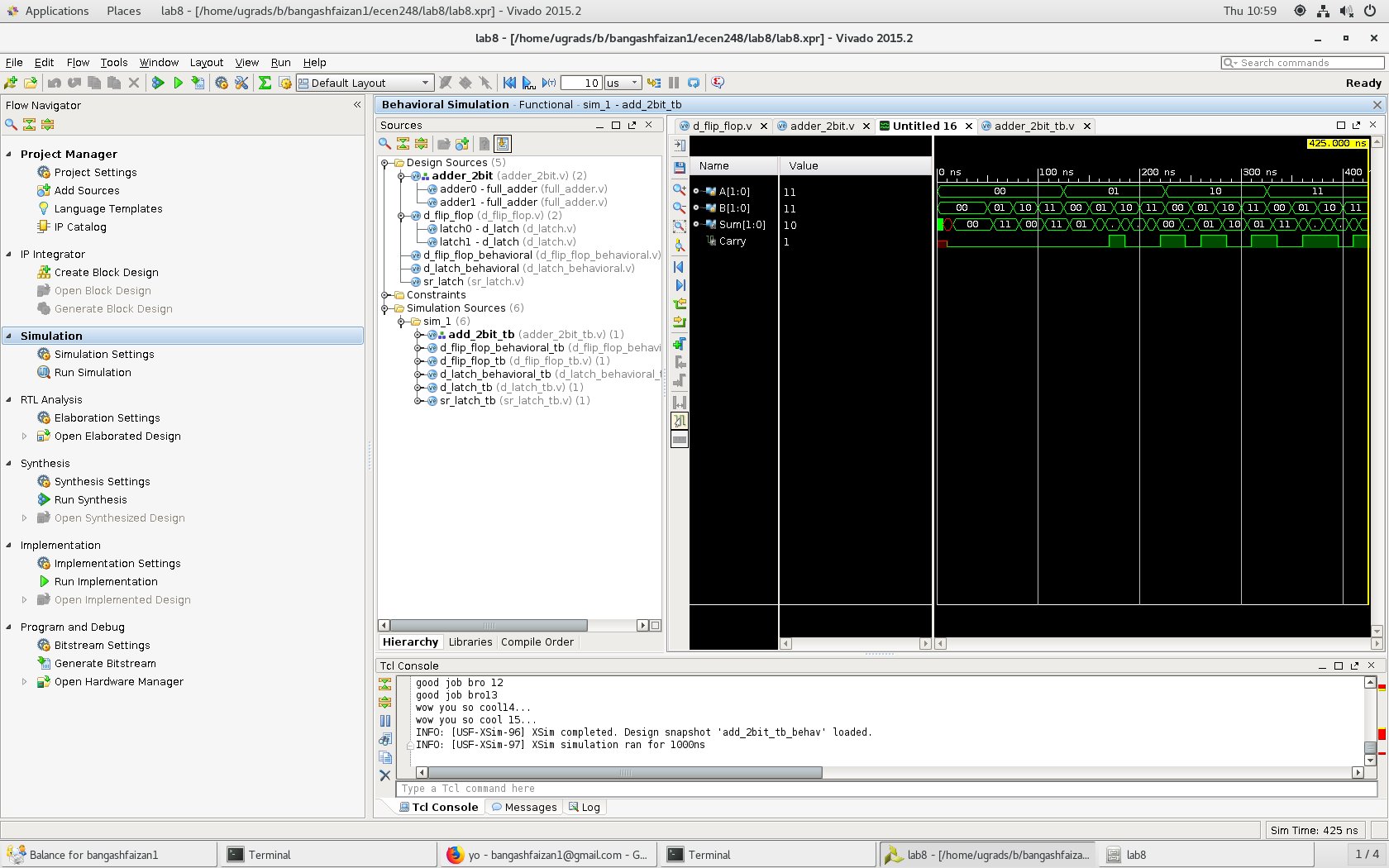


Results for D-Flip-Flop Behavioral

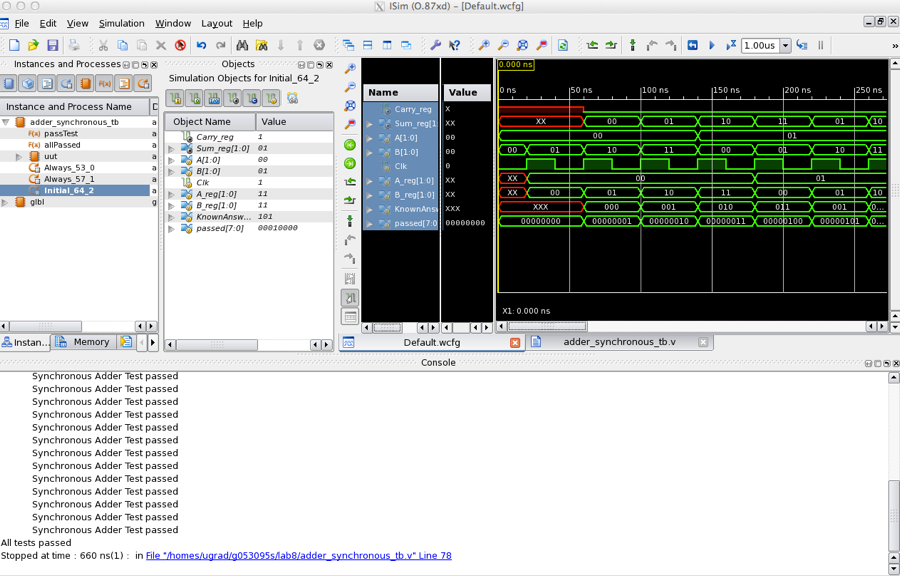


Results for D-Flip-Flop

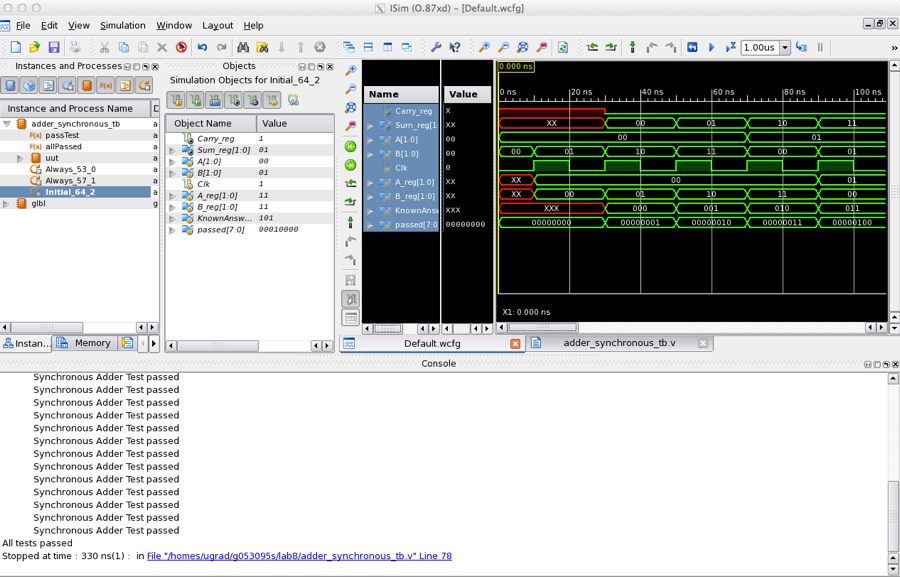
Experiment 2 :



Results for add\_2bit



Adder Sync results with 40 ns delay



Adder sync results with 20 ns delay

In this lab, I was able to witness memory storage in action. Additionally, I was able to see how time delays affect selected circuits in order to understand error propagation. Next, we designed a new version of the ripple adder using previous adder programs as well as new programs and an update test bench. Lastly, the adder files were all put together to create a synchronous adder. Overall, there were minimal issues except for the synchronous adder. When running the test bench, I was getting letters for my answers instead of numbers, however, there was no noticeable issues in my code.

**Conclusion –**

The focus of this lab was sequential circuits and how to implement them. Overall, they do bring new ideas to verilog however it is very similar to what we have before in verilog. Additionally, we also got to witness memory and how it works and how to adjust it for error. The skills learned in this lab will be beneficial for future labs as I will be growing my skills in verilog that will be needed for when the problems get more advanced.

**Questions:**

1. Explain the results of the simulation in Experiment 1.e.
   1. In 1.e, the 2 ns delay worked, however, the 4 ns delay failed. The 4 ns failed because it occurs after the input phase of the program, where R, S and En were set at a 2 ns delay. Because they were different, this causes a clash, leading to failure.
2. For 1.3, check the waveform with internal signals. Do the latches behave as expected? Why or why not?
   1. The latches do behave as expected. This is because it follows the gate logic despite the delays.
3. For experiment 1.4 in part 1, compare the waveforms you captured from the behavioral Verilog to those from the structural Verilog. Are they different? If so, how?
   1. The only difference is there was a time difference between the two. Behavioral verilog was faster than structural, however, the results were the same other than that. This was caused by the fact that the D-latch behavioral code had no time delays while the structural did.
4. Determine the worst case propagation delay through the ripple carry adder in experiment 1.e in part 2.
   1. The worse case propagation delay was 10 ns.
5. Based on the clock period you measured for your synchronous adder, What would be the theoretical maximum clock rate? What would be the effect of increasing the width of the adder on the clock rate? How might you improve the clock rate of the design?
   1. The lowest clock rate period was 20 ns, so the maximum clock rate would be 50 MHz. Increasing the width of the adder would cause the clock rate to drop because more inputs means higher delays. To improve the clock rate, you would want to optimize the circuit to have less gates as this would reduce the delay.