**Lab 11: A Simple Digital Combination Lock**

Faizan Bangash

TA : Saurav Kumar Sahu

ECEN 248 – 302

July 31st, 2018.

**Objectives** –

In this lab, we looked at a practical application of a FSM as well as memory by implementing them into the combination lock. This lab utilized the FPGA board so we were able to test and debug our code in real time. In this lab, memory was necessary because the lock required multiple inputs and states, so the device had to know where it was at.

**Design** –

1. In Experiment 1 of the lab, we simulated the code made in the pre-lab to make sure it worked. In figure 1, you can see the waveform that proves it did work. The source code can also be seen below.

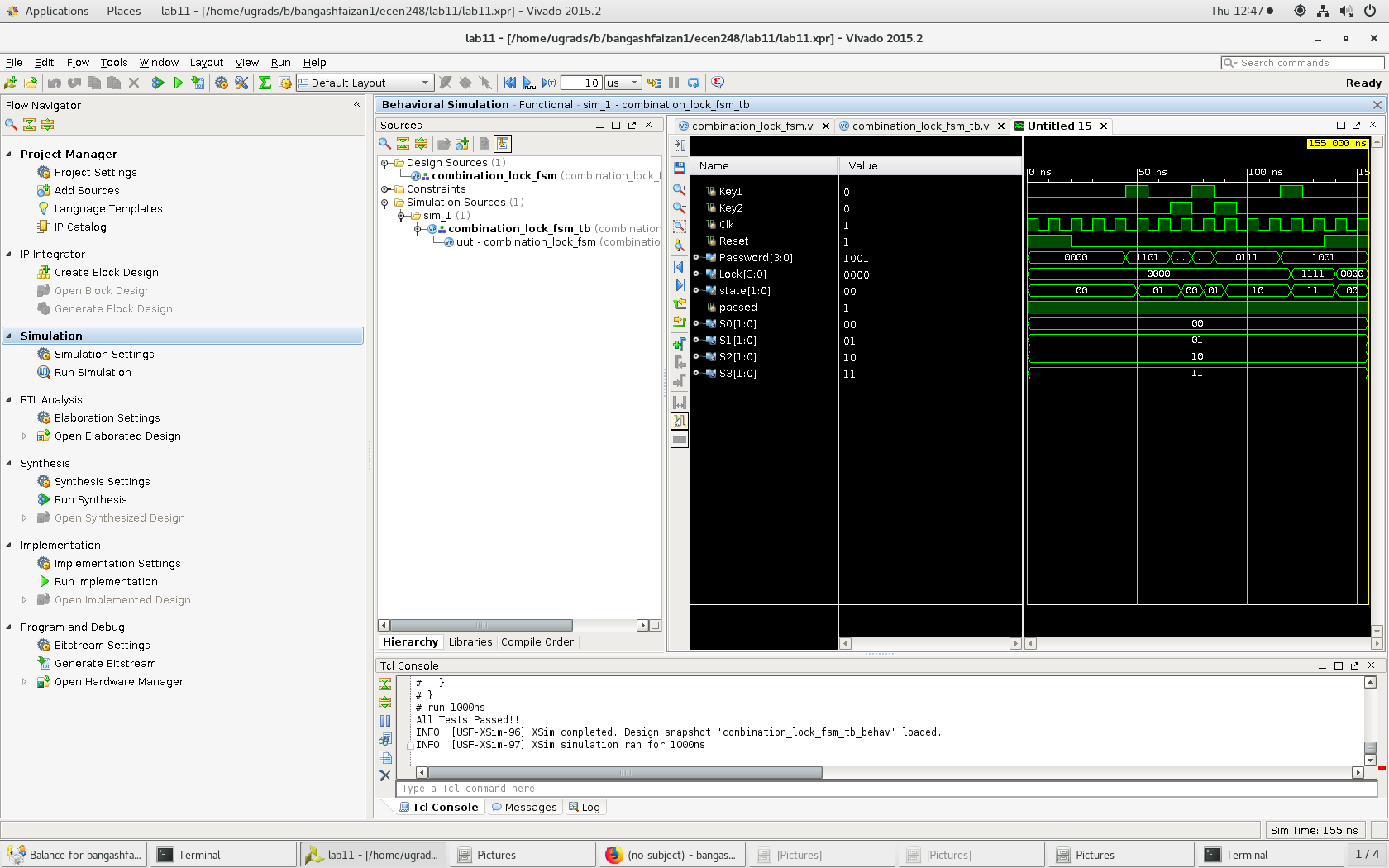
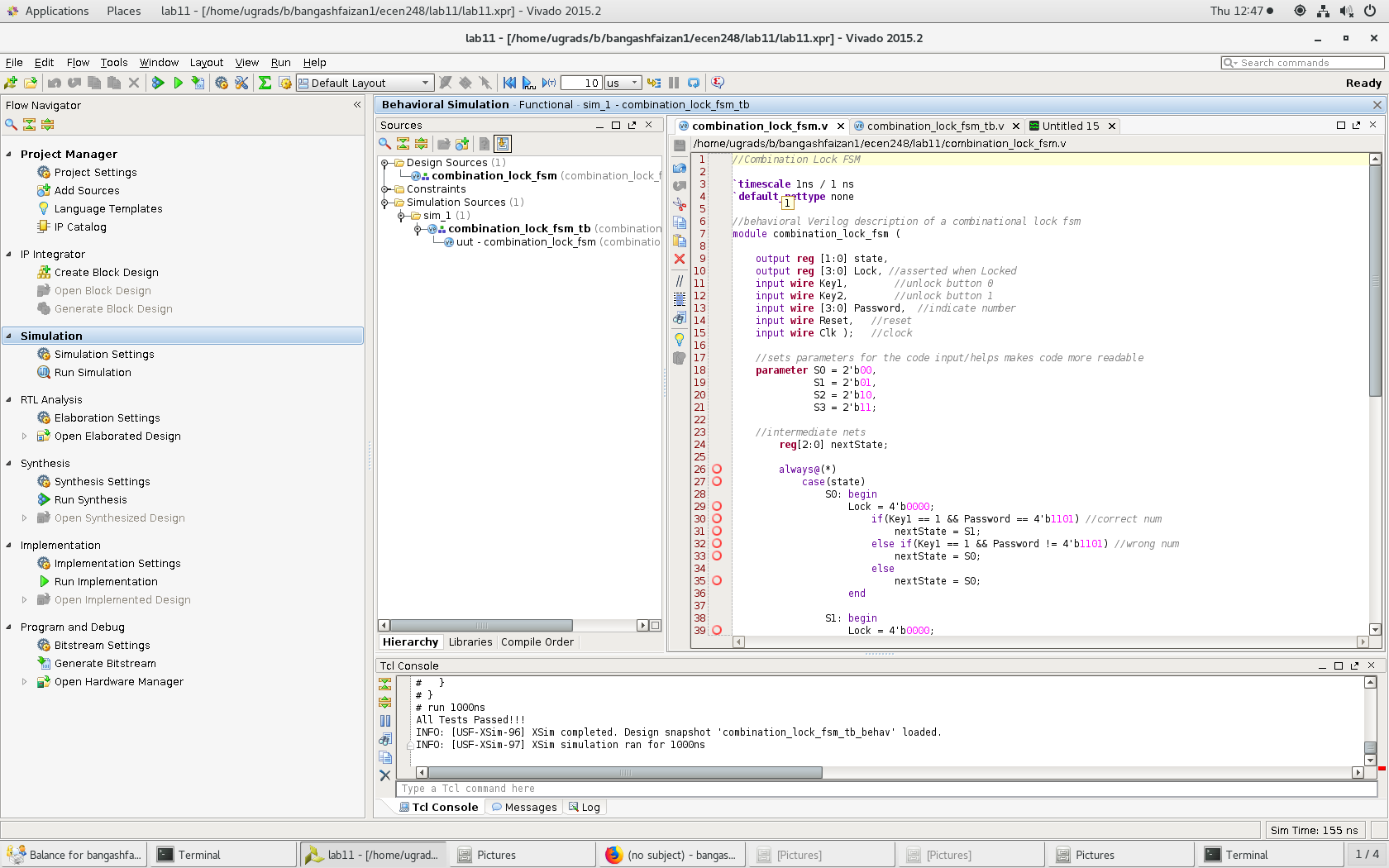
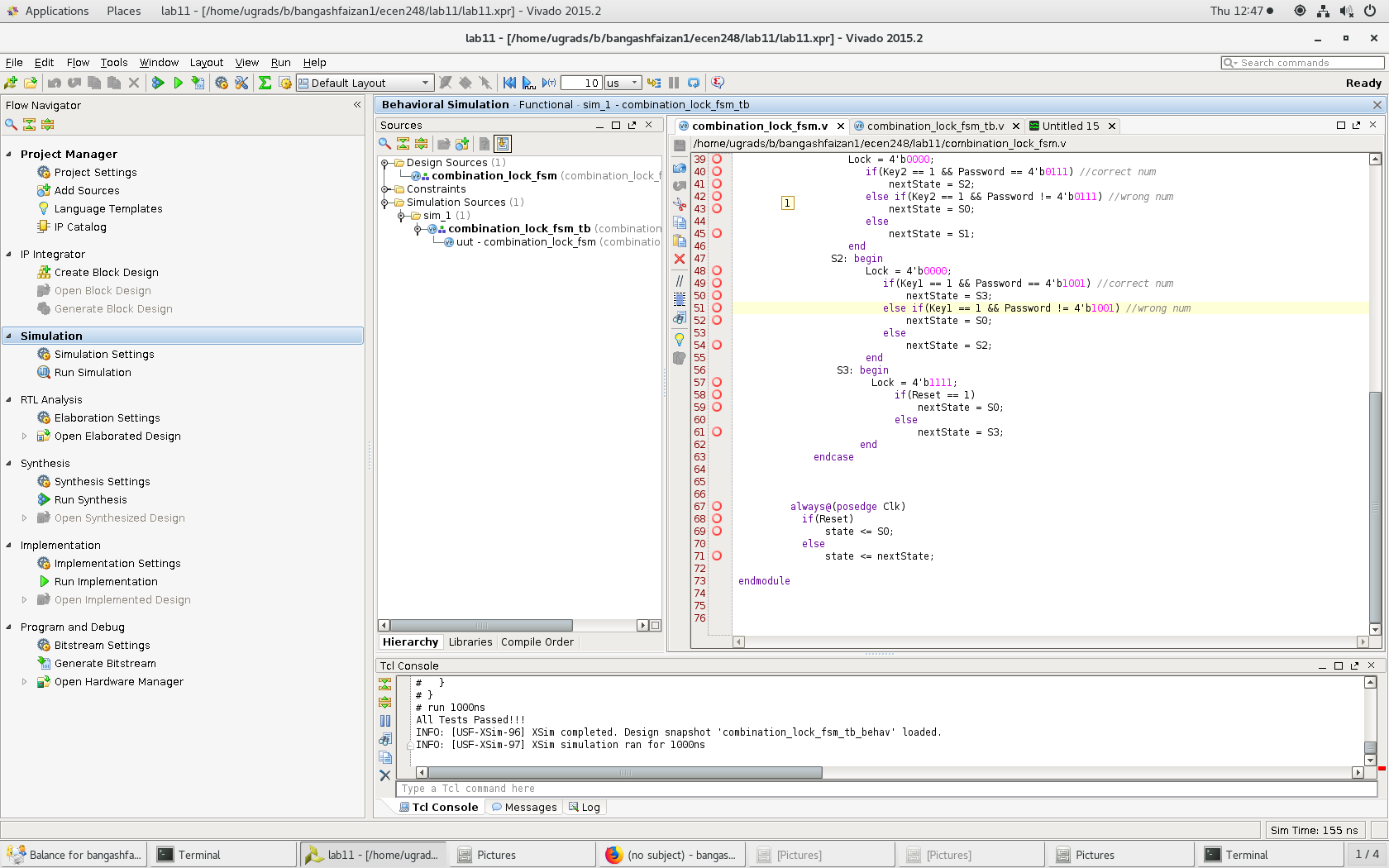
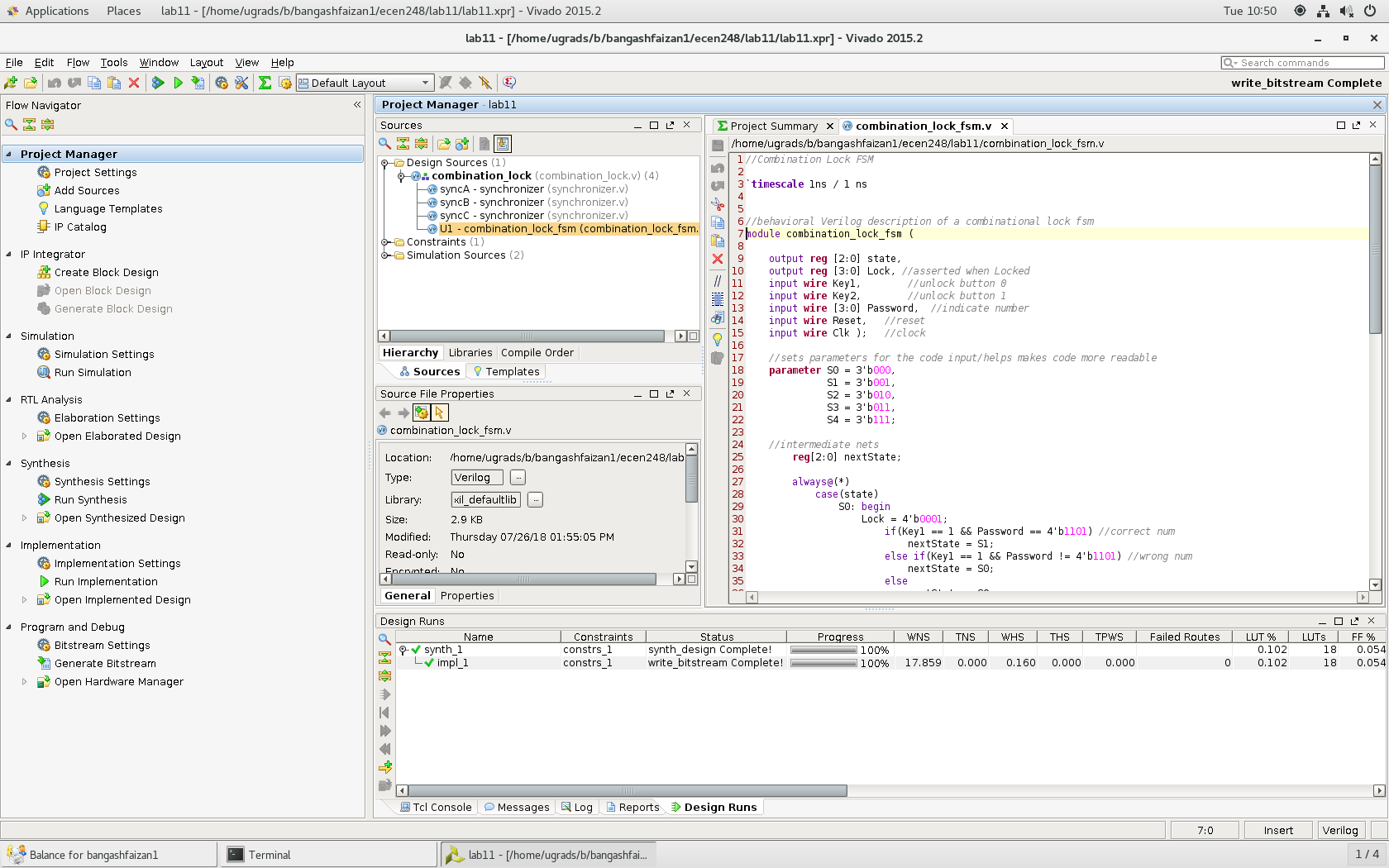


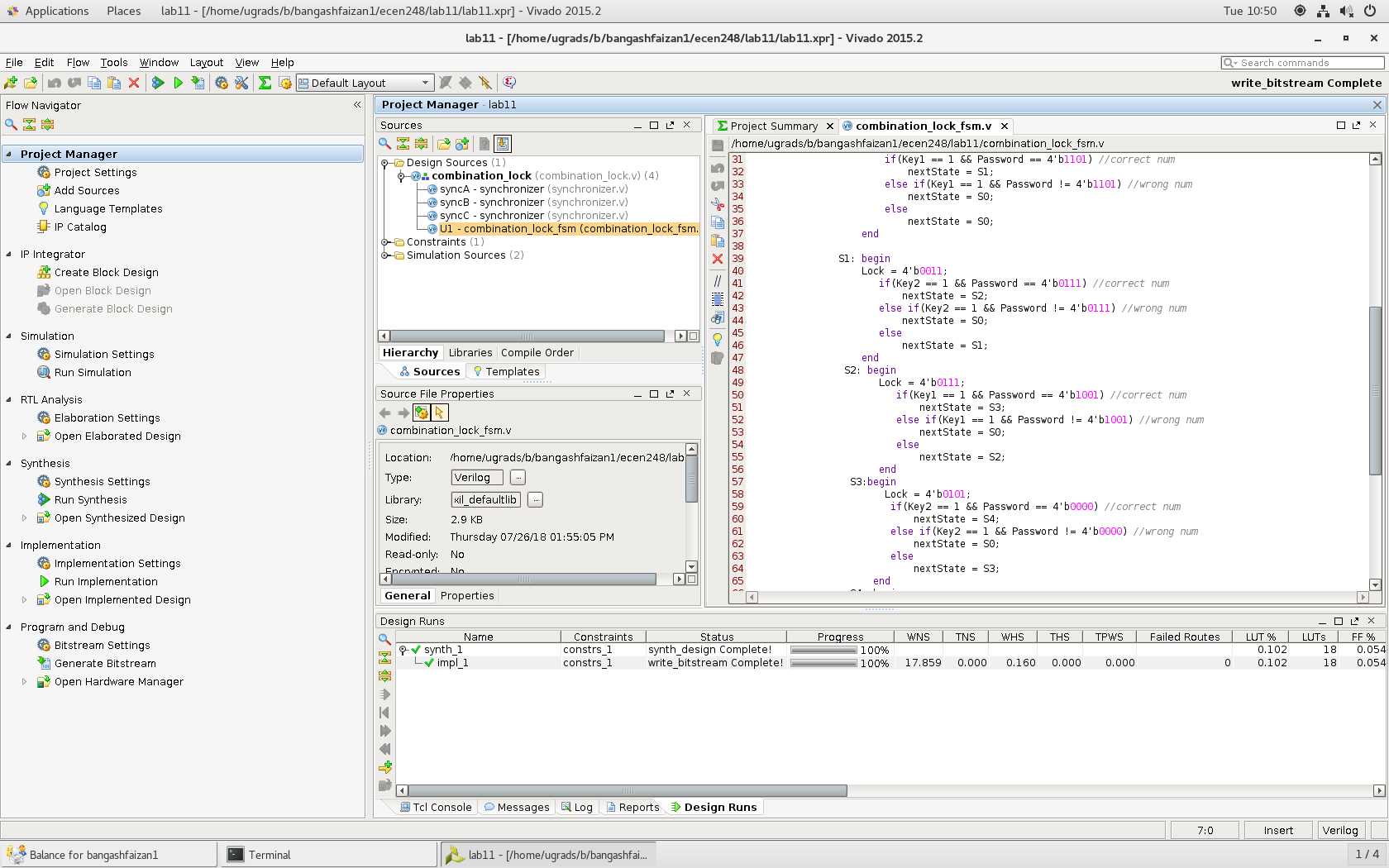
Figure 1- Waveform results of Experiment 1.

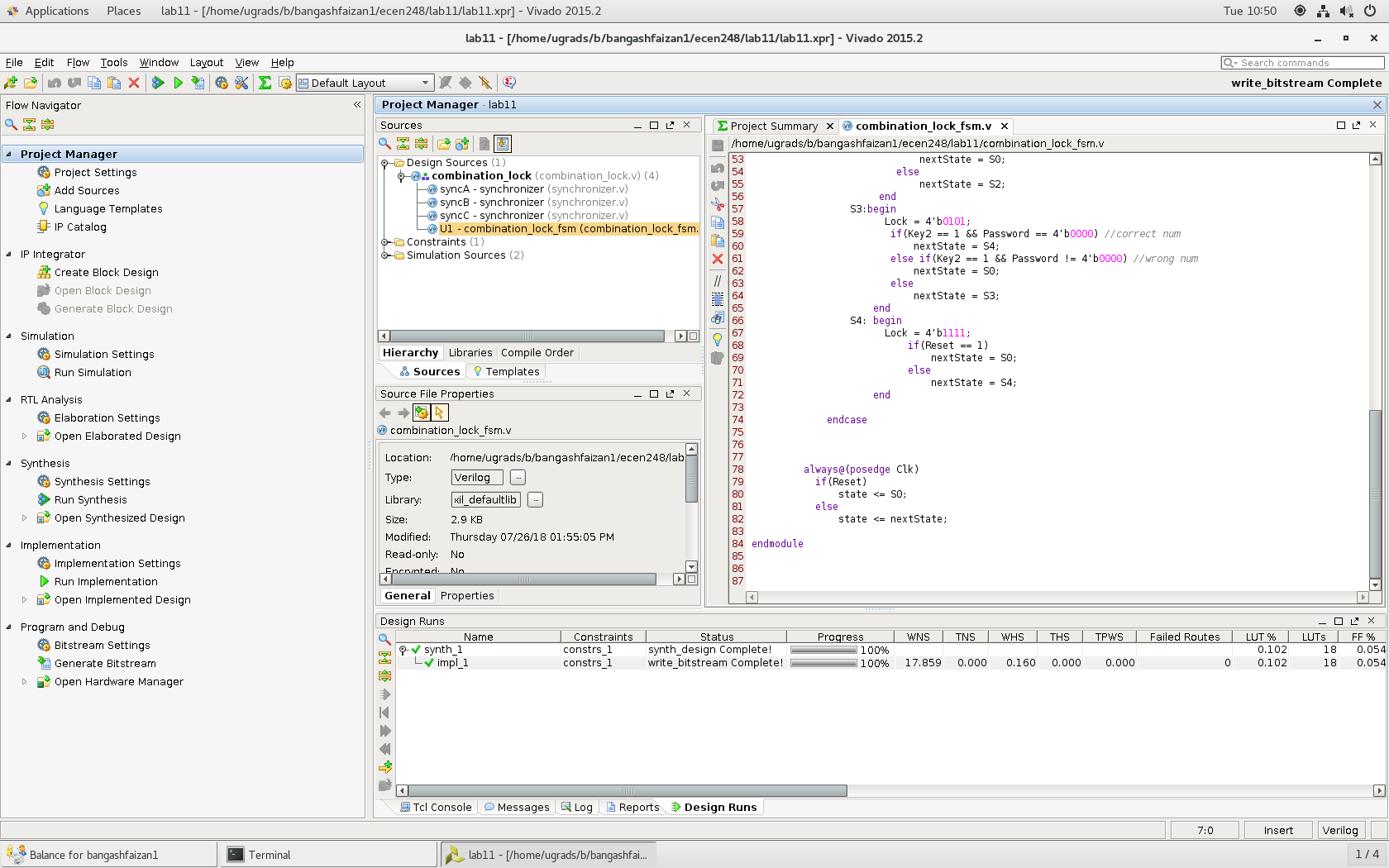




1. In experiment 2 of the lab, we integrated the design onto the Zybo board and tested to make it worked on the board instead of just the waveform. We implemented the above code to get a fully working Zybo board. In experiment 3, we added an additional number to the lock combination. The number we added was 0 and the code can be seen below.







**Results** –

Overall, in this lab I was able to get the final working product. However, the path to getting the lock to work was not the best. The main issue I had was when I checked to see if the switches were in the right state I was not checking to see if the button was also pressed. This was causing the lock to always loop back to the beginning. However, after I discovered this bug, the program worked as expected. Adding the extra number was no issue as well I just had to expand the bit size for the state variable.

**Conclusion –**

This lab was good as it allowed me to become more familiar with state machines as well as grasp how to manage memory in structural verilog. This lab was good as it allowed me to reenforce my knowledge of the Zybo board as well as it gave me a chance to make something that had a practical application. Lastly, this lab helped me learn how to debug more by using the Zybo board as well as the waveform when I was having issues with my program.

**Questions:**

1. Include the source code with comments for all modules you simulated and/or implemented in lab. You do not have to include test bench code that was provided! Code without comments will not be accepted!
   1. Included in the design section
2. Include screenshots of all waveforms captured during simulation in additional to the test bench console output for each test bench simulation.
   1. Included in the design section
3. Answer all questions throughout the lab manual
   1. Experiment 1, Question E- Take a look at the simulation waveform and take note of the tests that the test bench performs. Is this an exhaustive test? Why or why not?
      1. Yes the test bench evaluates if each state is correct at every given time and clock cycle.
4. A possible attack on your combination-lock is a brute-force attack in which every possible input combination is tried. Given the original design with a combination of three numbers between 0 and 15, how many possible input combinations exist? How about for the modified design with a combination of four numbers?
   1. For 3 numbers, there are 16^3 combinations which turns out to be 4096 total combinations. For 4 numbers, there are 16^4 combinations which turns out to be 65536 combinations.