CS342/CS343
Instructor: Professor Izidor Gertner
Spring 2022
Comparators Lab, Azwad Shameem, 2/28/2022

Table of Contents

Objective

The objective of this assignment is to create VHDL code in Quartus and test the correctness of the VHDL code with waveforms in ModelSim. In order to accomplish this the assignment requires the design of 1 bit, 2 bit and 8 bit comparators using Quartus and the verification of correctness of each of comparators using the test bench file to create a waveform simulation in ModelSim. Furthermore, this assignment also intends for the practice of optimization of VHDL code in each of the 1 bit, 2 bit and 8 bit comparators by simplifying or reducing the lines of code of circuits that are already made and then testing that optimized code by verifying its correctness by utilizing the testbench's simulation waveform.

Code:

Task 0: Complete tutorial as listed below.

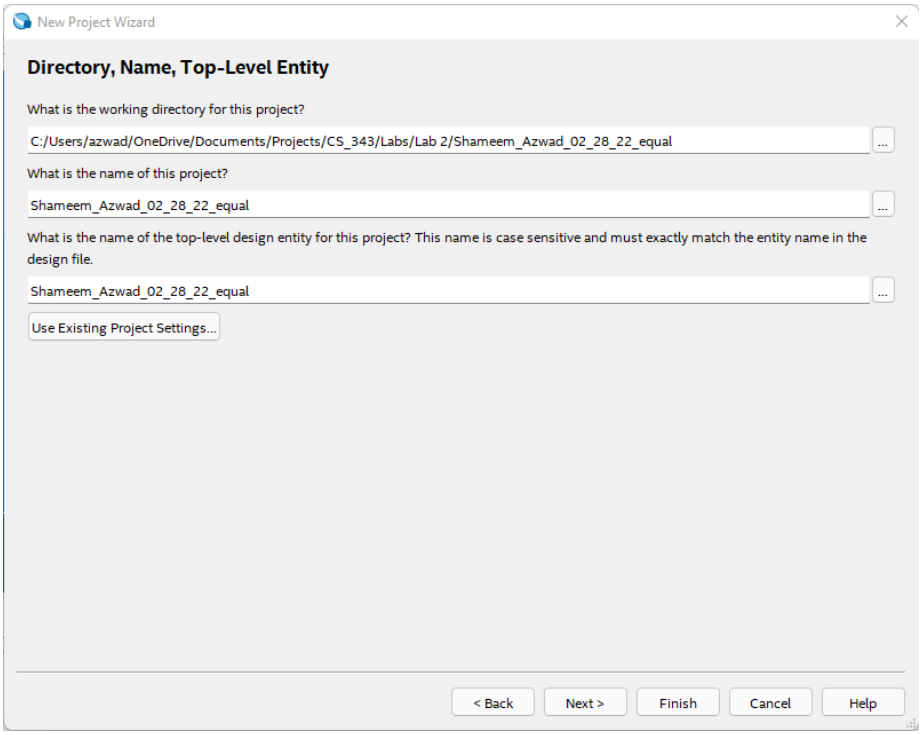

Figure 1: Creating project in Quartus

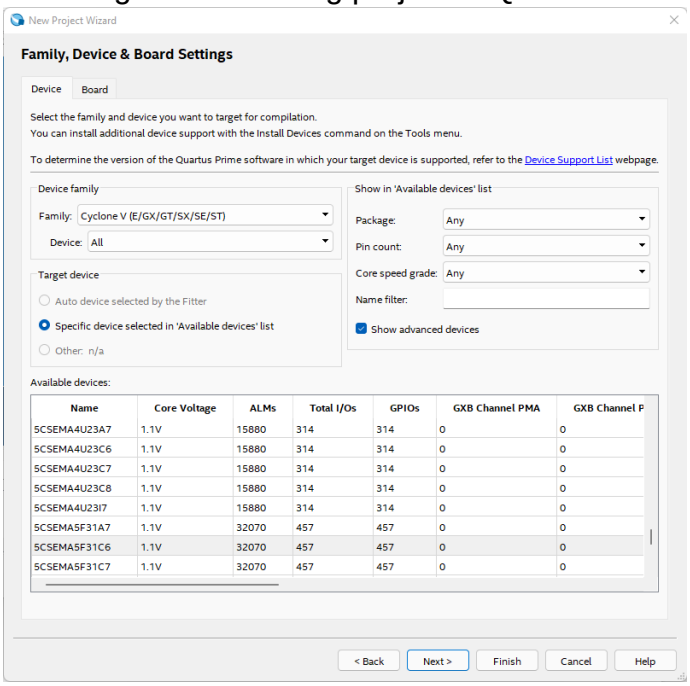

Figure 2: Device chosen

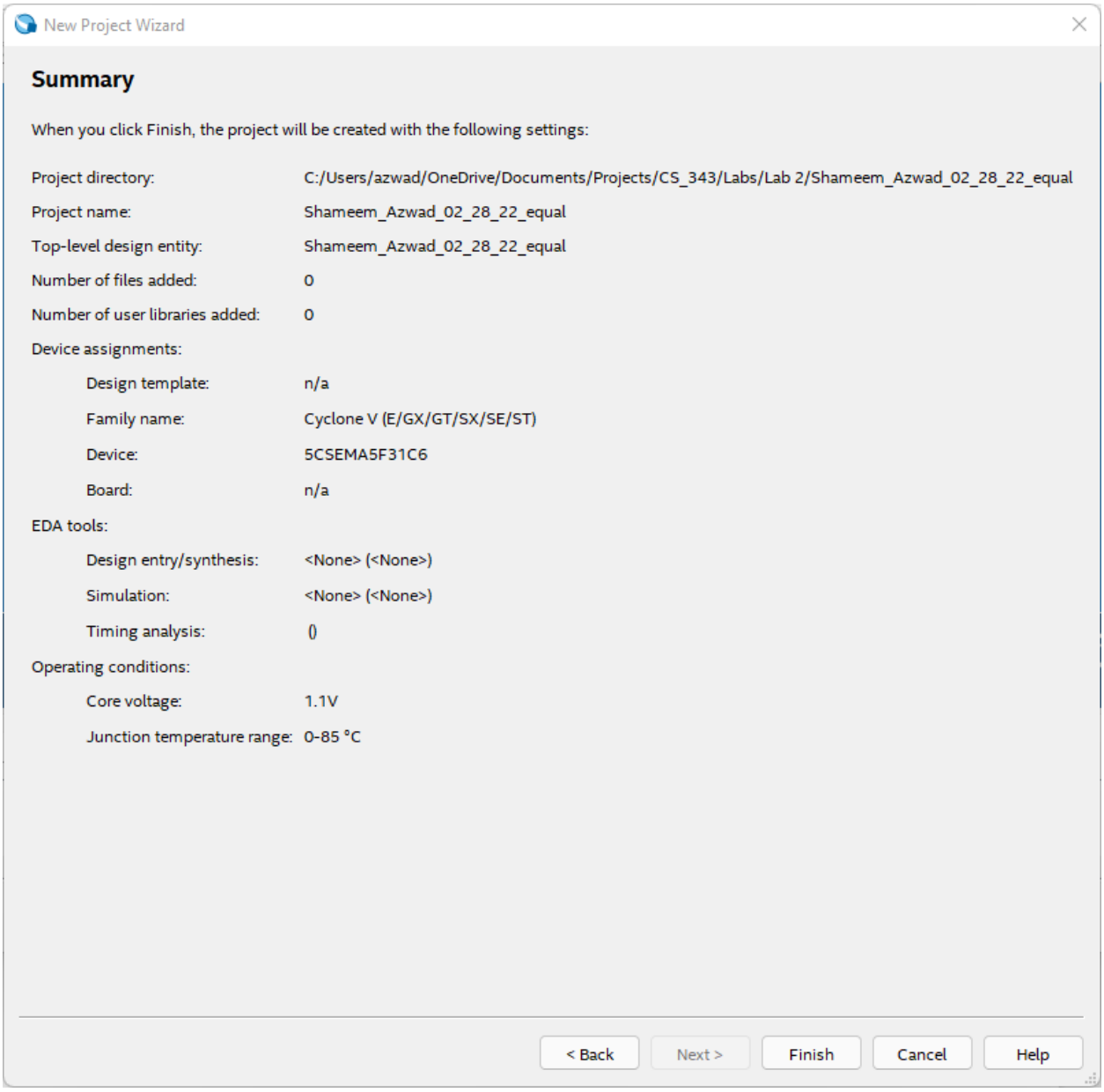

Figure 3: Project Summary

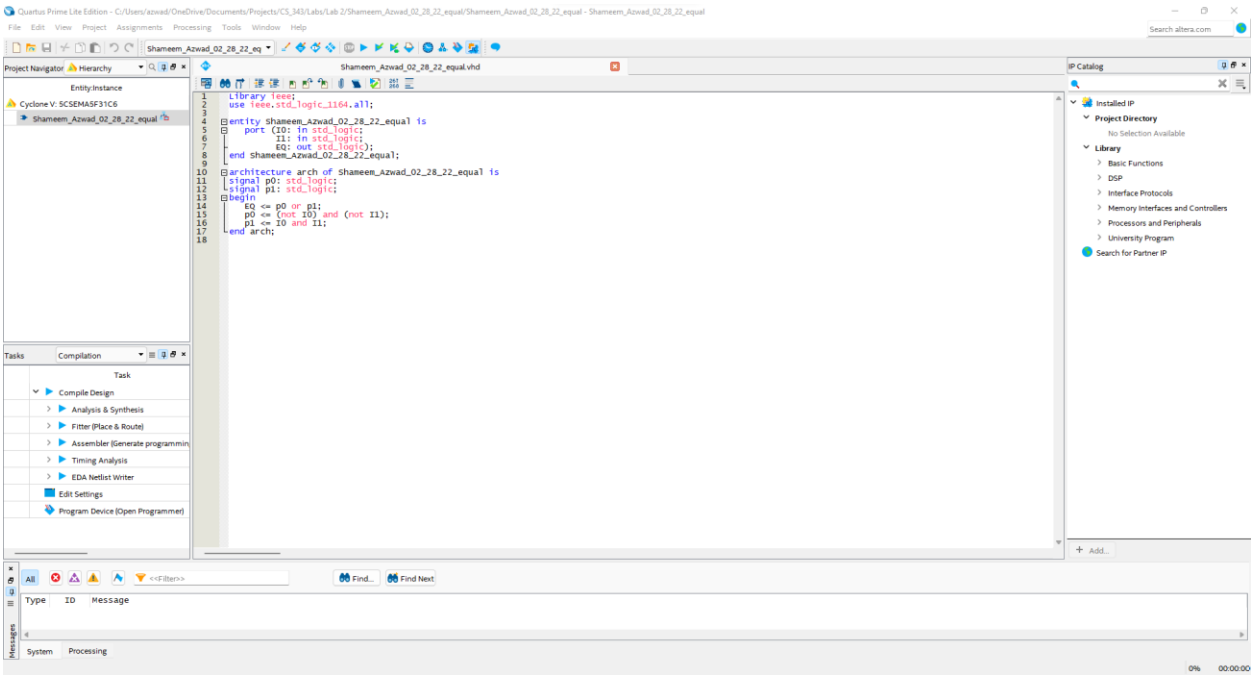

Figure 4: VHDL code for equal (one bit comparator)

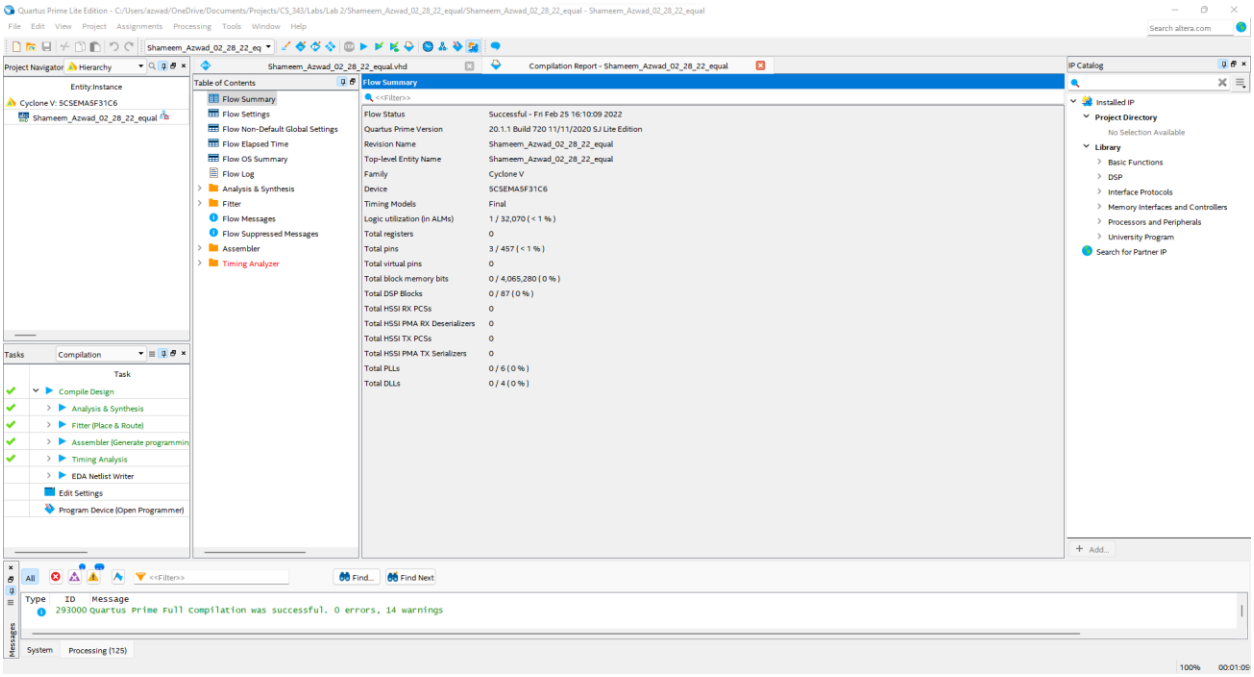

Figure 5: VHDL code compiled successfully on Quartus

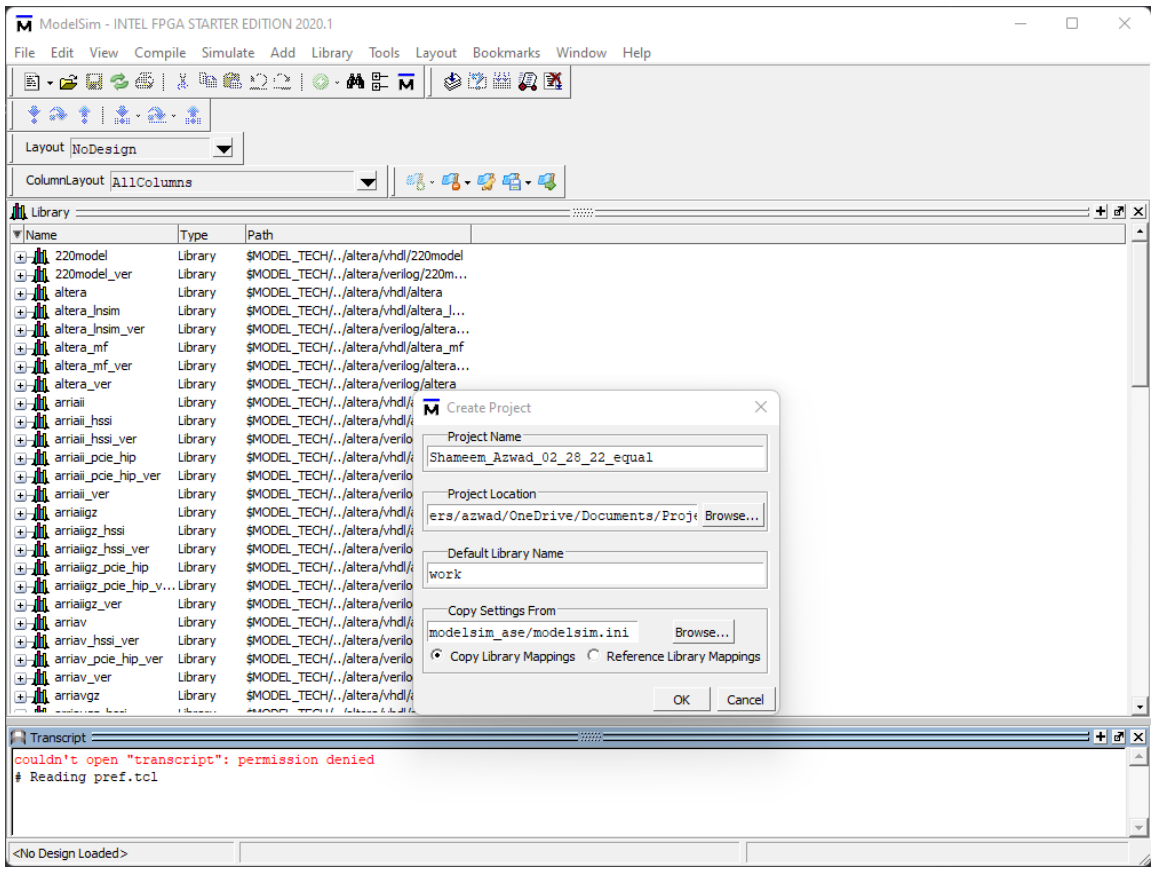

Figure 6: Project created in ModelSim

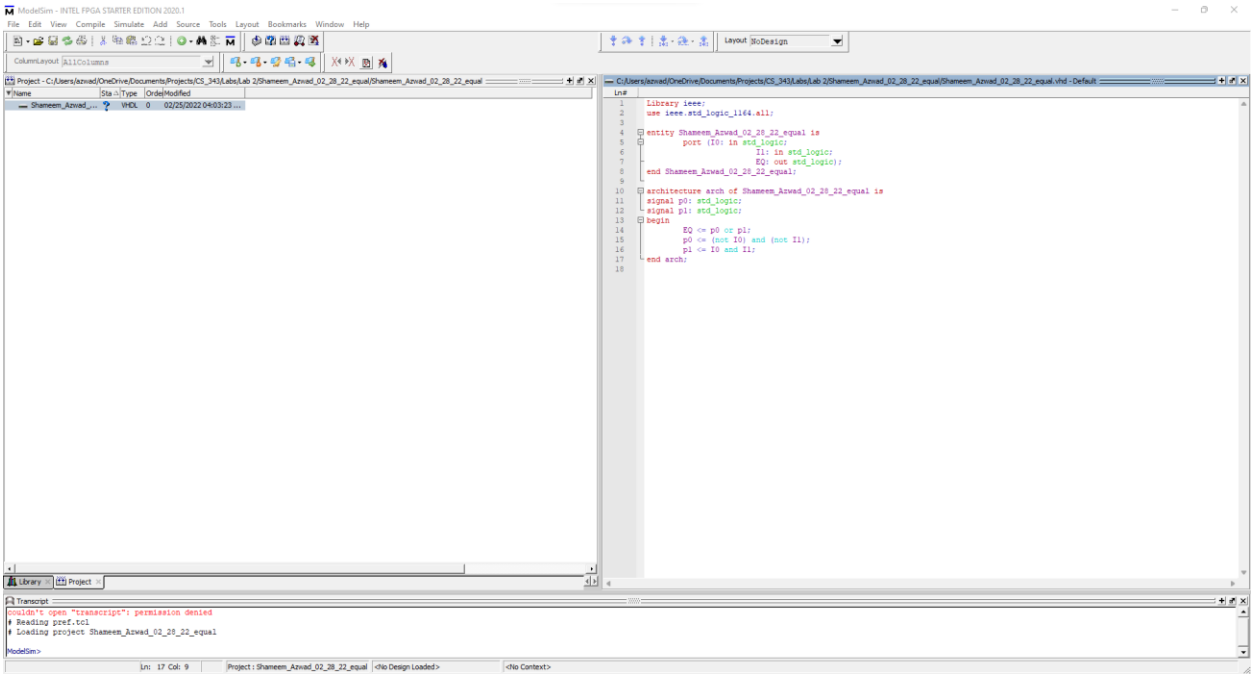

Figure 7: VHDL code in ModelSim

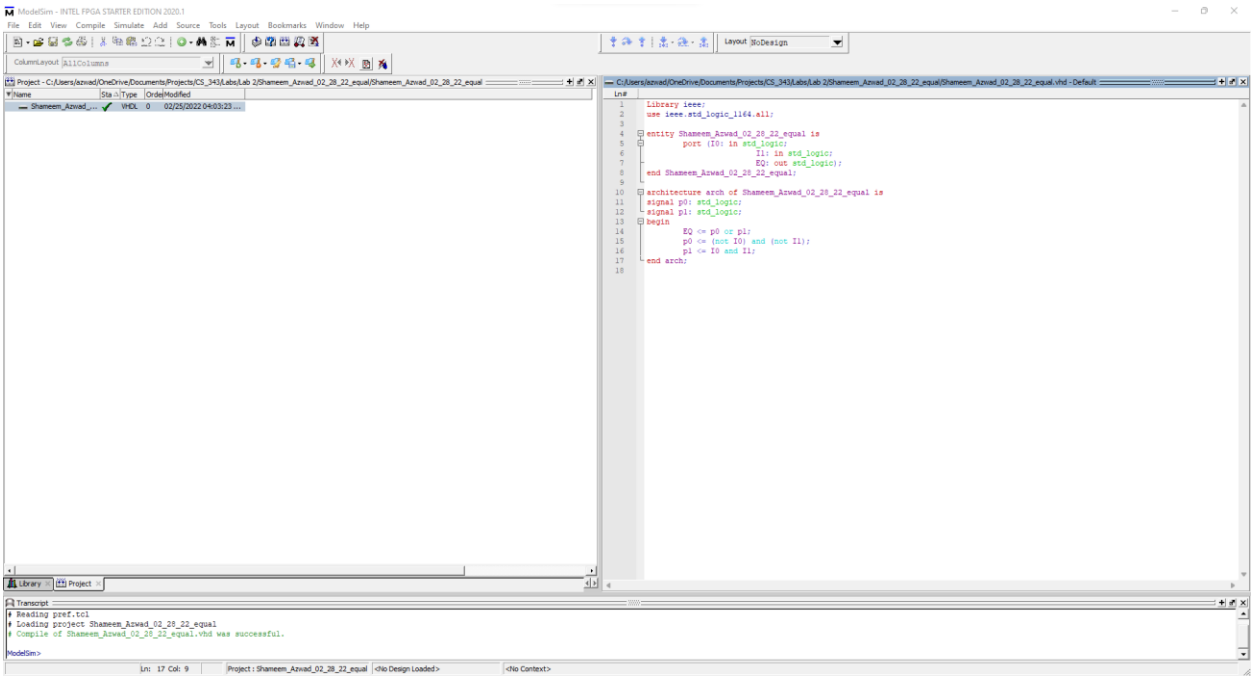

Figure 8: VHDL code compiled successfully in ModelSim.

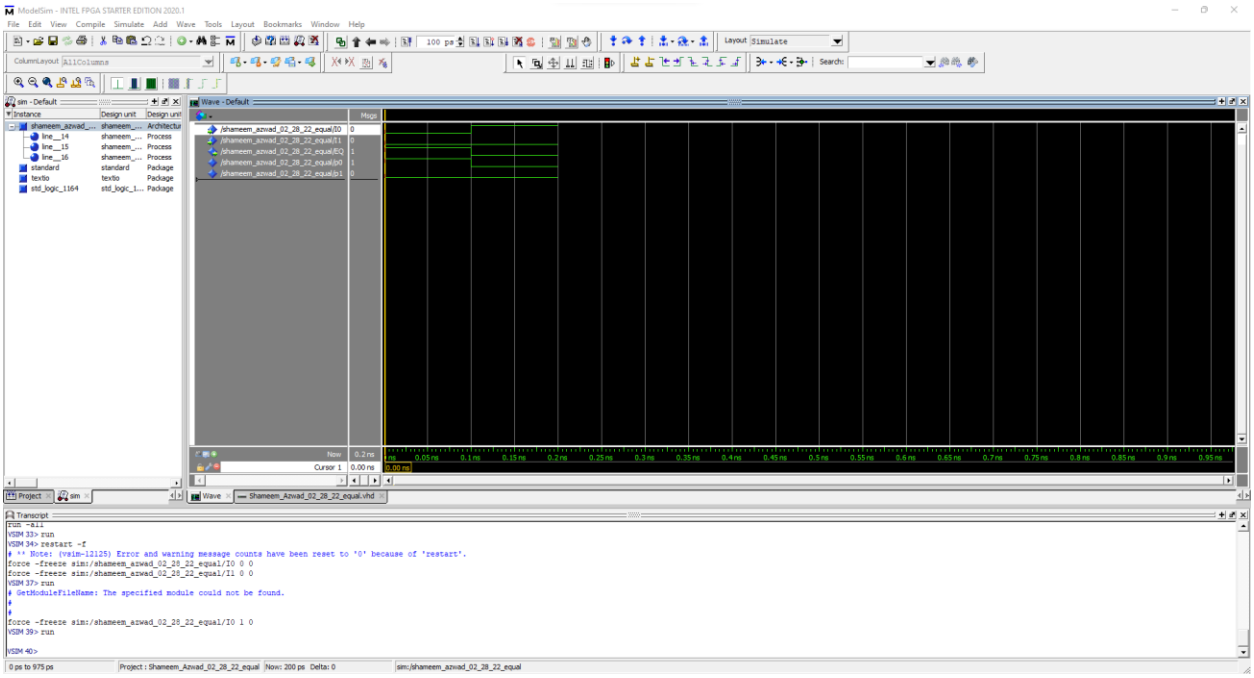

Figure 9: Simulation following tutorial of equal

The above image follows the tutorial in where we are using forced values for the inputs in order to test the circuit.

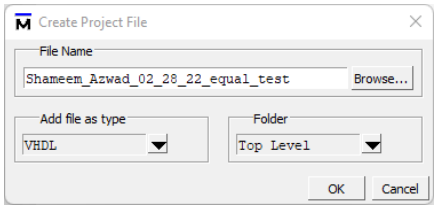

Figure 10: Creating test bench file

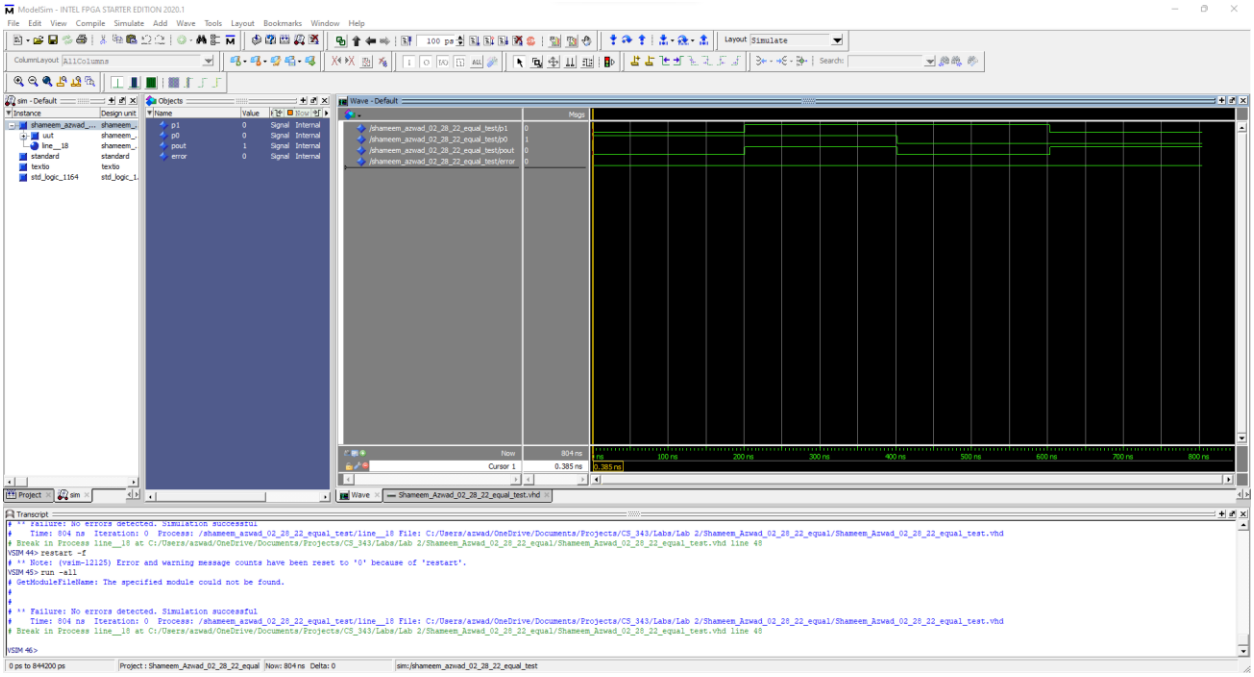

Figure 11: Simulation using the test bench

The above image utilizes the test bench in order to test the correctness of the circuit. In fact, the waveform follows the truth table of a one bit comparator which proves its correctness.

Task 1

Task 1a) write VHDL code for 2 bit comparator compile it

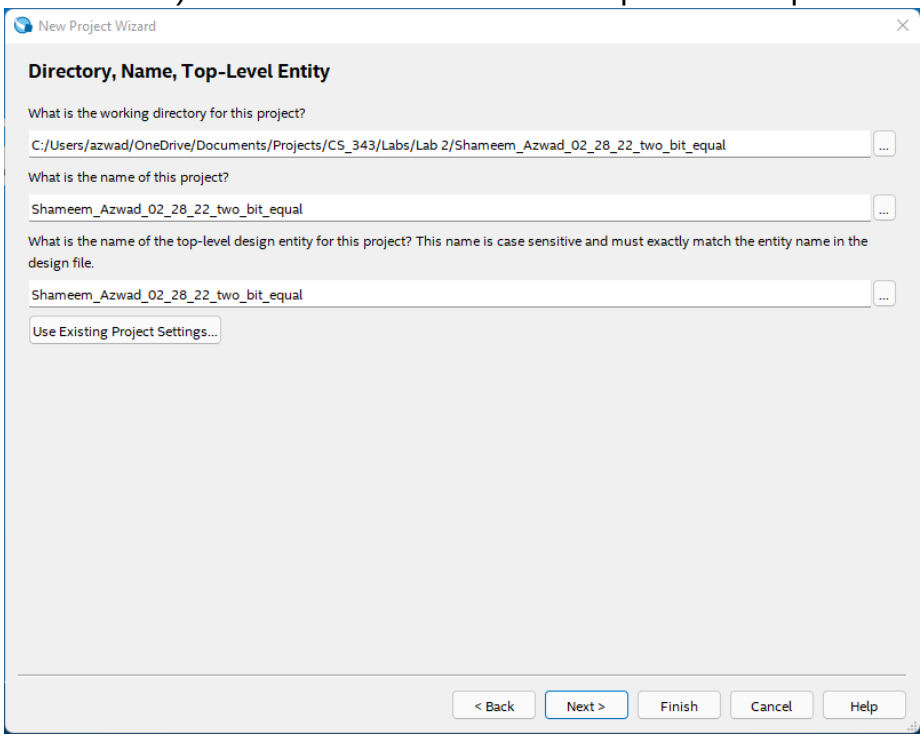

Figure 12: Two Bit Equal Project created

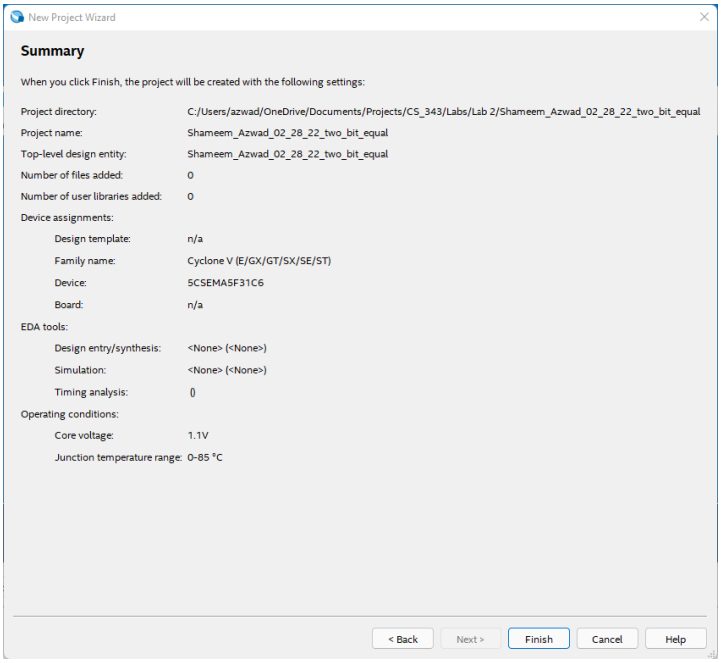

Figure 13: Project Summary for Two Bit Equal

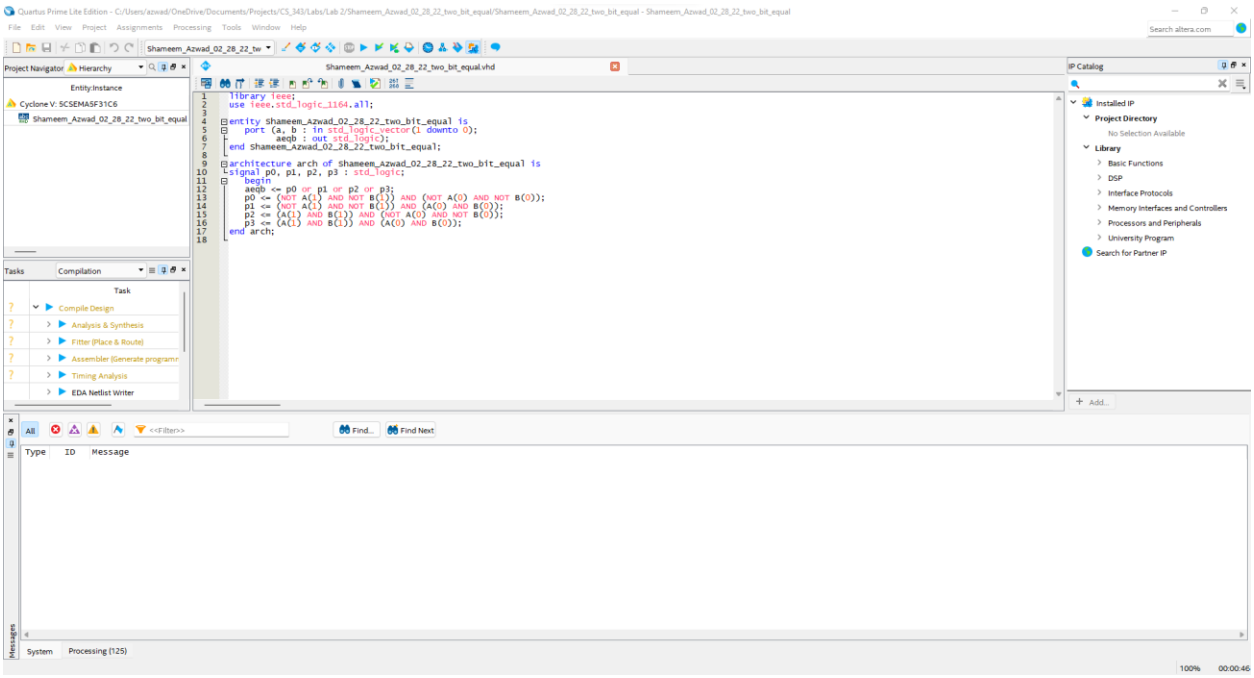

Figure 14: VHDL code for two bit equal

The equation used to deduce the logic for p0, p1, p2, p3 was

(a1'.b1').(a0'.b0') + (a1'.b1').(a0.b0) + (a1.b1).(a0'b0') + (a1.b1).(a0.b0).

This equation allows us to see that each plus sign is an OR so p0, p1, p2, p3 have to bee the values in between the plus signs. In addition, we know that the ‘ sign means NOT and the . sign must be an AND.

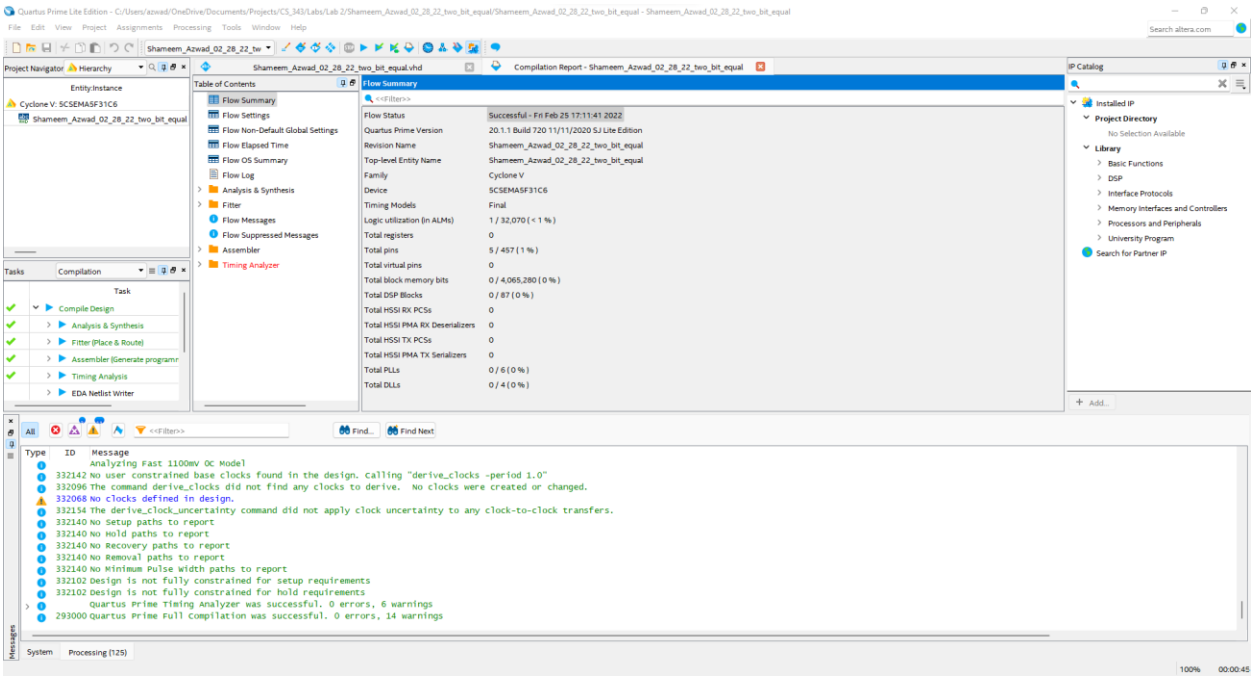

Figure 15: VHDL code compiled successfully in Quartus

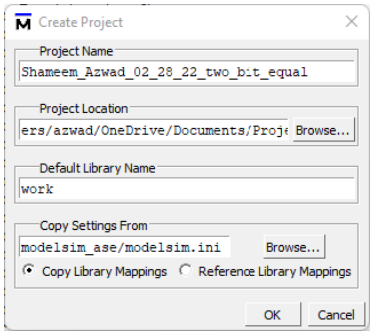

Figure 16: Two bit equal Project created on ModelSim

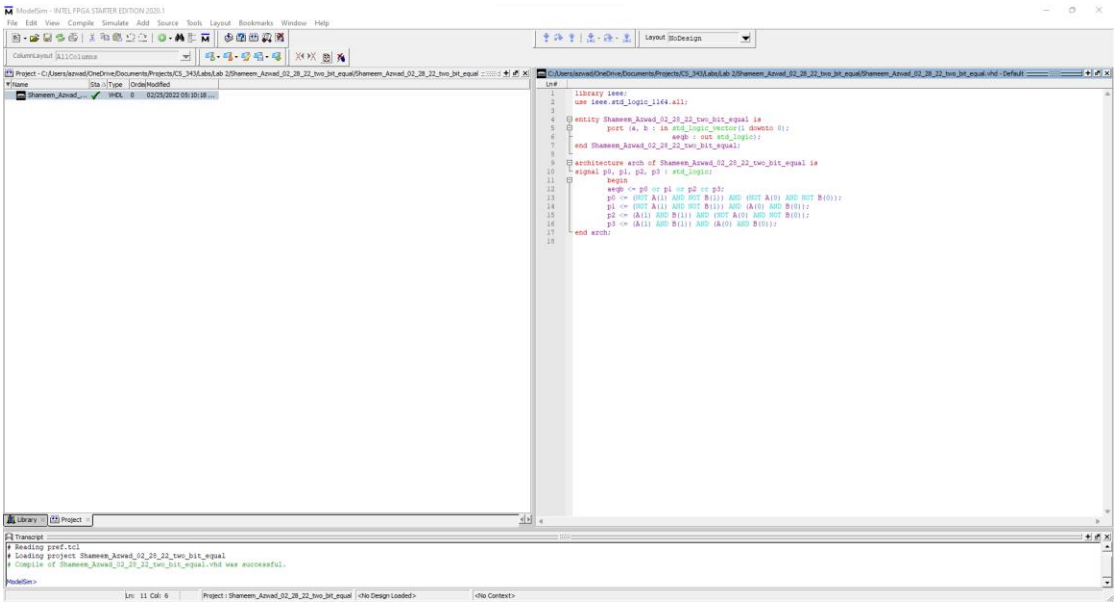

Figure 17: VHDL code compiled successfully on ModelSim

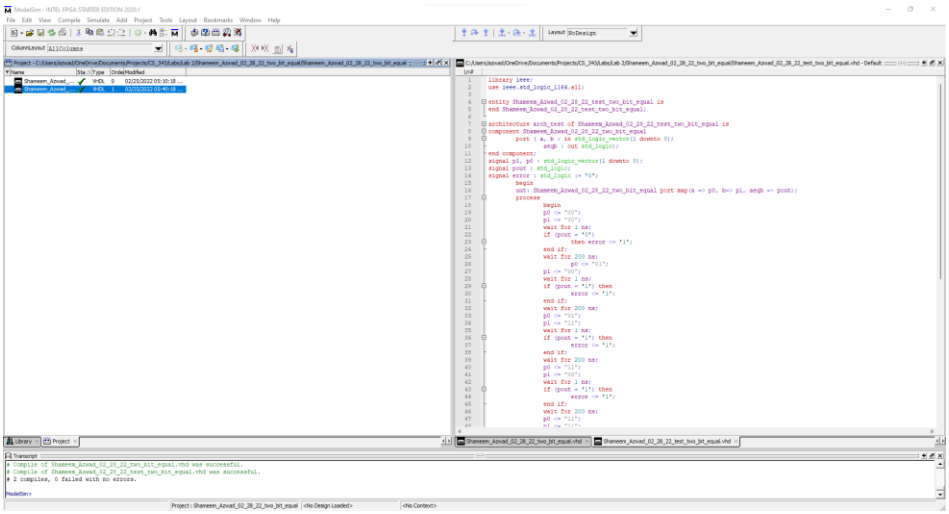

Figure 18: testbench code for two bit equal compiled

Task 1b) Verify correctness of 2 bit comparator using Model-SIM using tutorial.

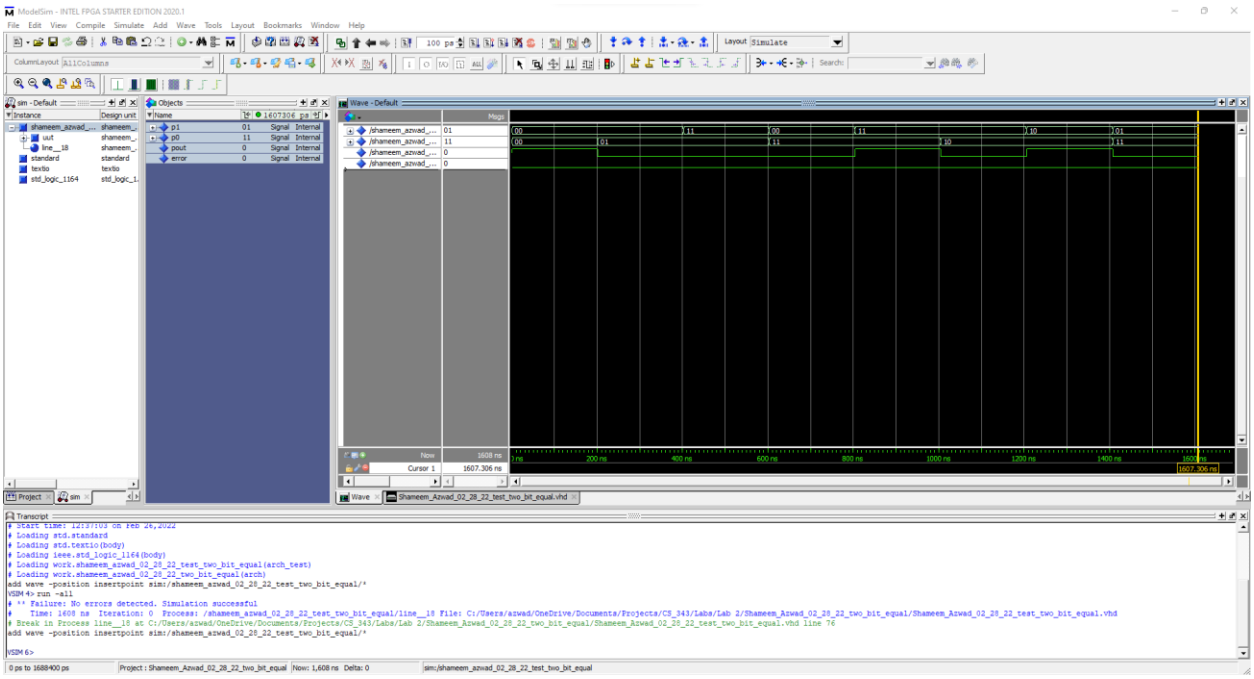

Figure 19: Simulation for two bit equal

This waveform shows the correctness of the two bit equal.

Task 2

Task 2a) write VHDL code for 8 bit comparator compile it

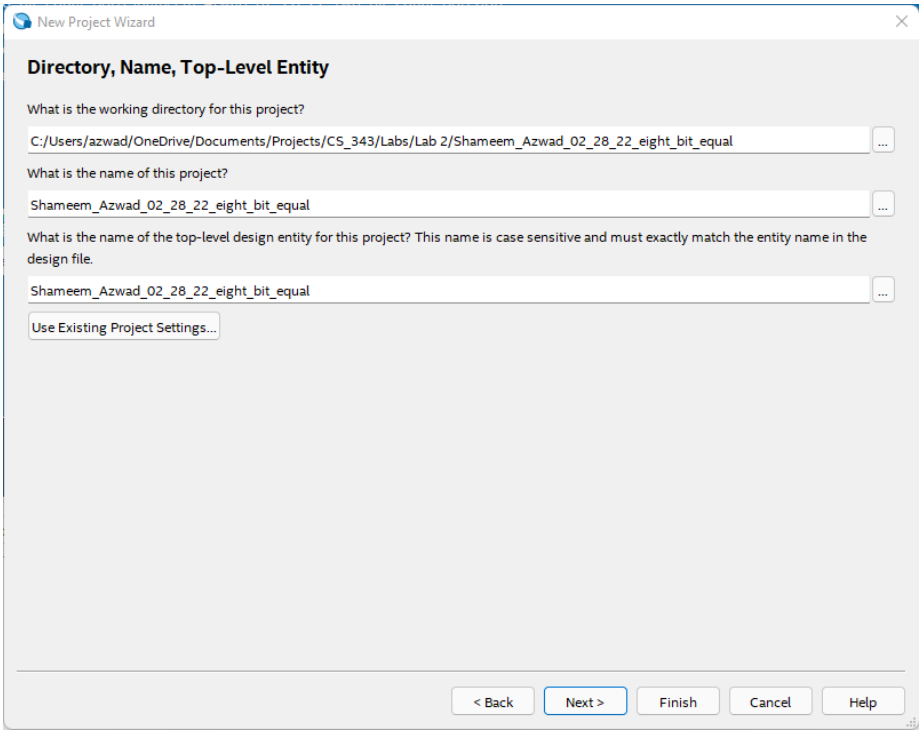

Figure 20: 8 bit comparator project creation

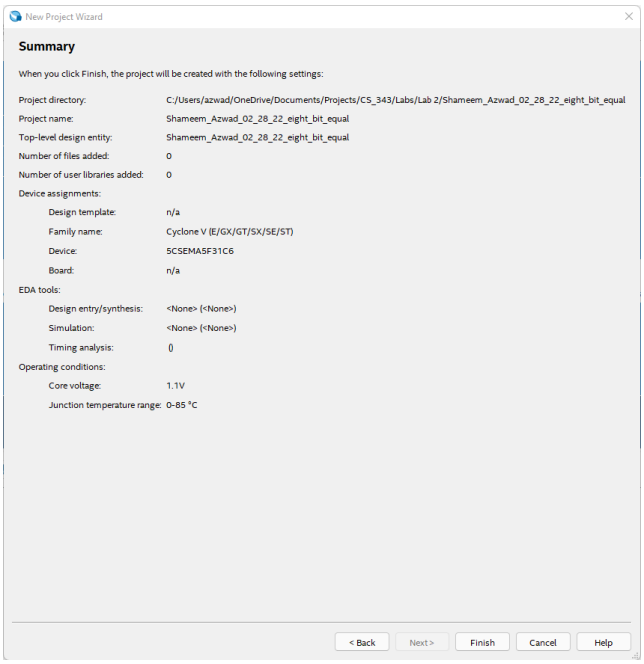

Figure 21: Project Summary

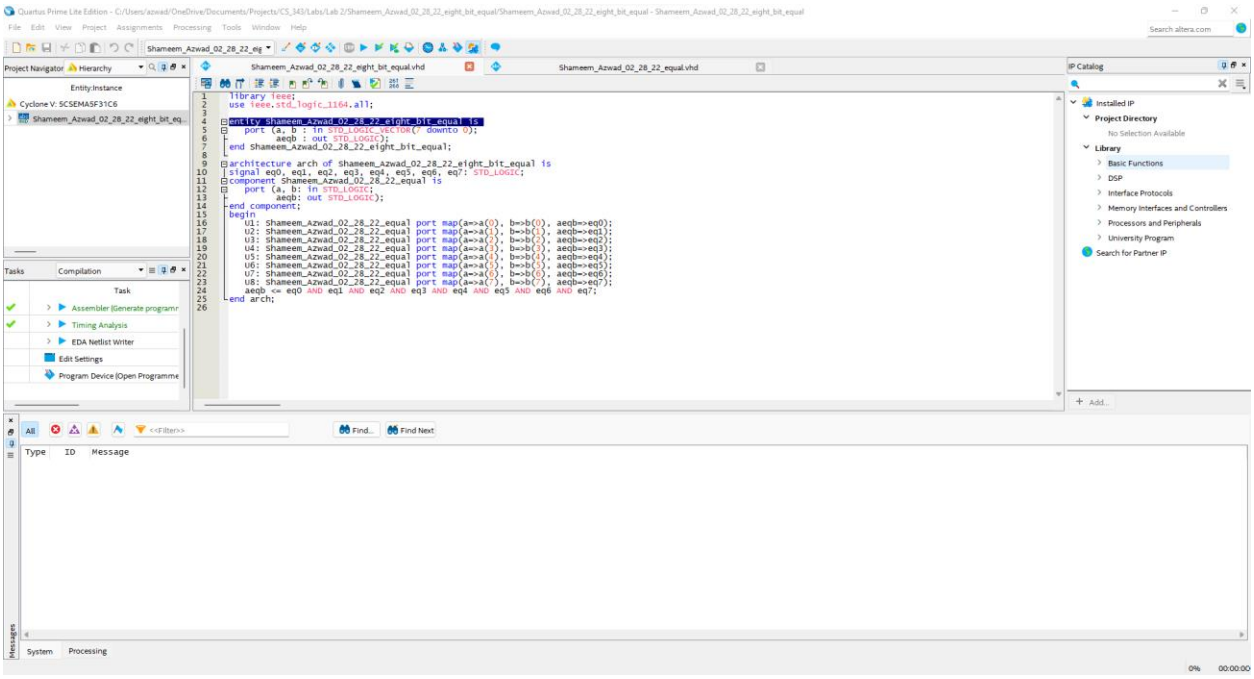

Figure 22: VHDL Code for the 8 bit comparator

The 8 bit comparator utilizes eight 1 bit comparators and ANDs them together.

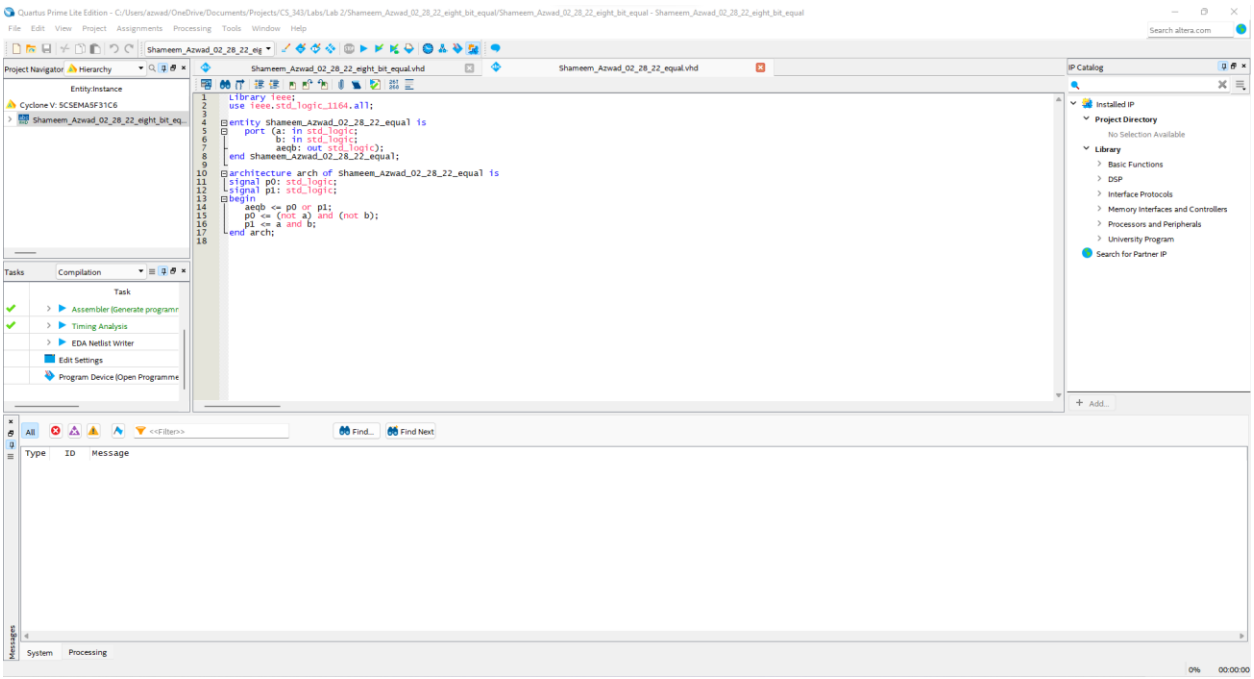

Figure 23: VHDL code for the 1 bit comparator that is used as a component

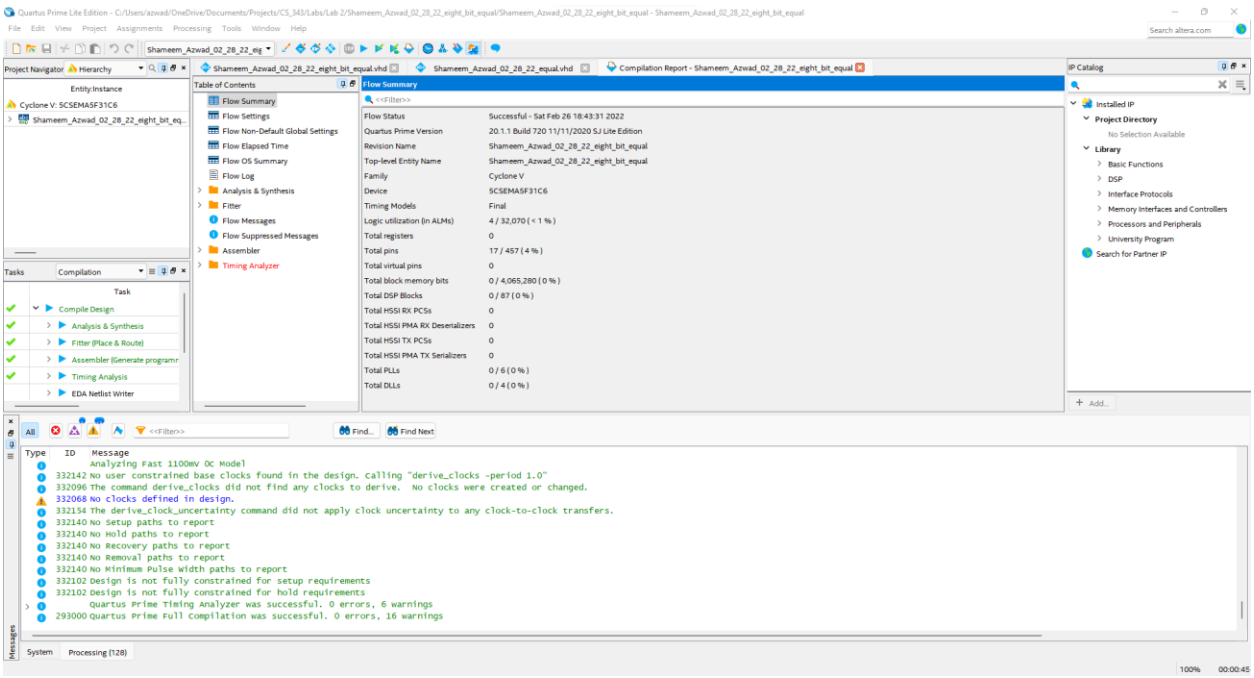

Figure 24: VHDL code successfully compiled on Quartus

Task 2b) Verify correctness of 8 bit comparator using Model-SIM

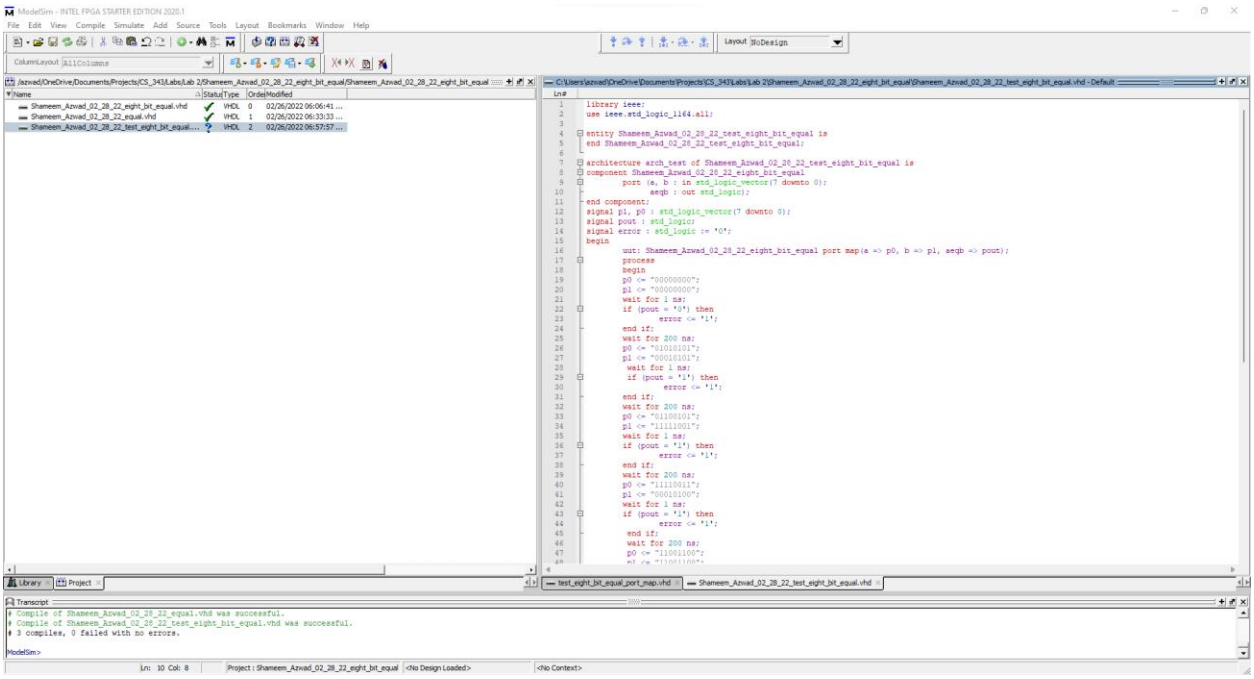

Figure 25: VHDL code successfully compiled on ModelSim

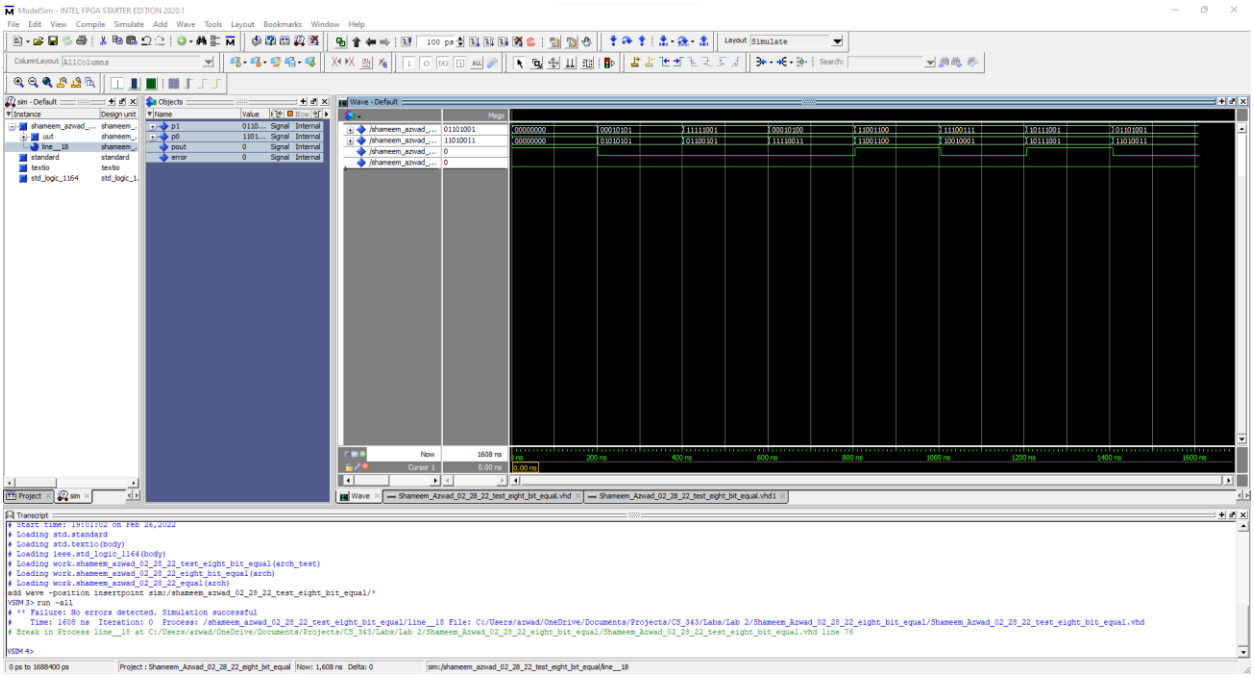

Figure 26: Eight bit comparator simulation waveform

Waveform correctly follows the outputs intended, which shows its correctness.

Task A1) Optimize the 1 bit comparator VHDL code shown on page 3, to replace lines 12,13,14 with ONE Boolean operation

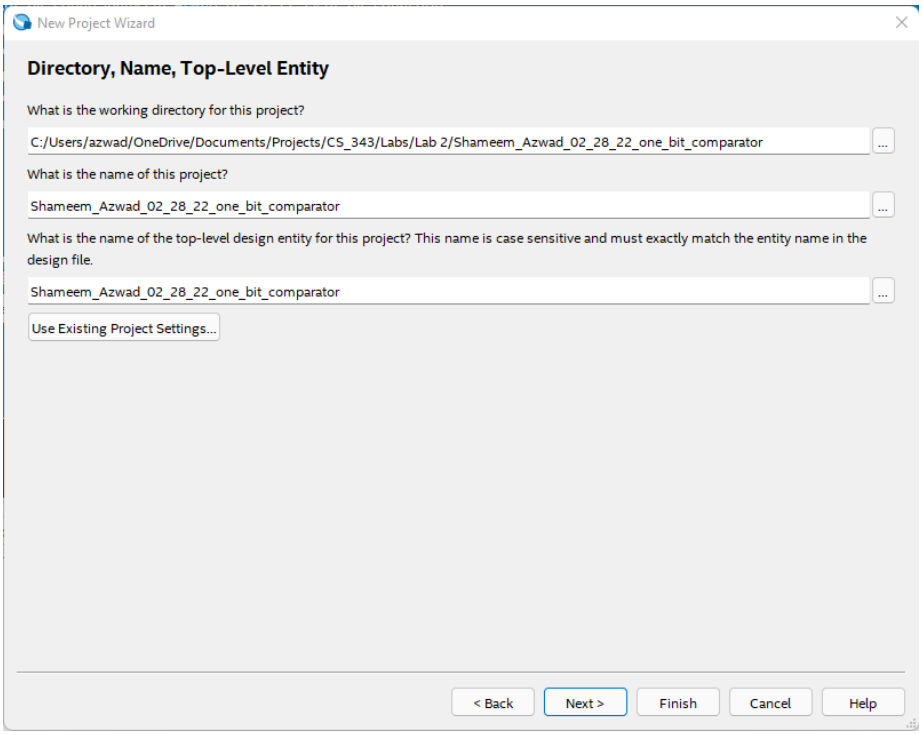

Figure 27: Creating new project for 1 bit comparator (this version is the optimized one)

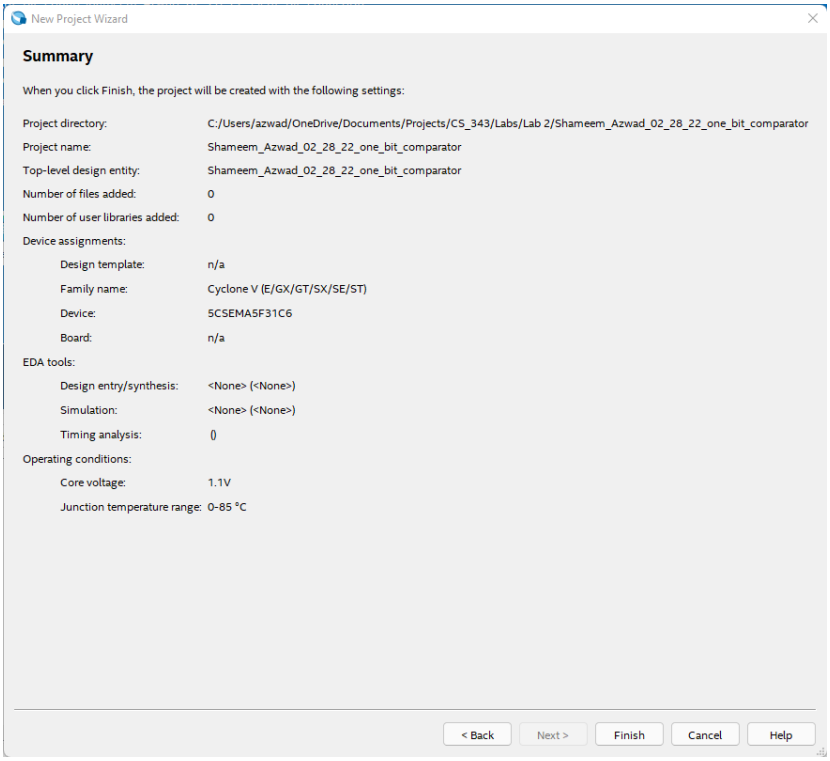

Figure 28: Project Summary

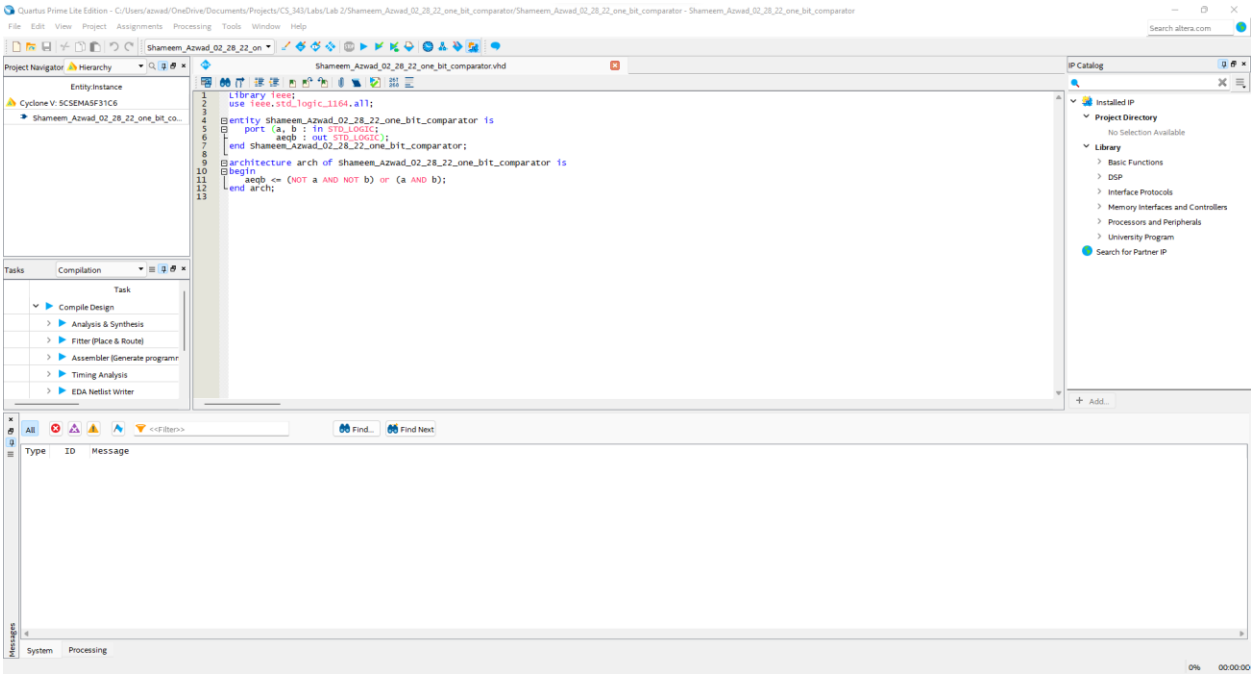

Figure 29: Optimized 1 bit comparator.

Replaced lines 12,13,14 with ONE Boolean operation.

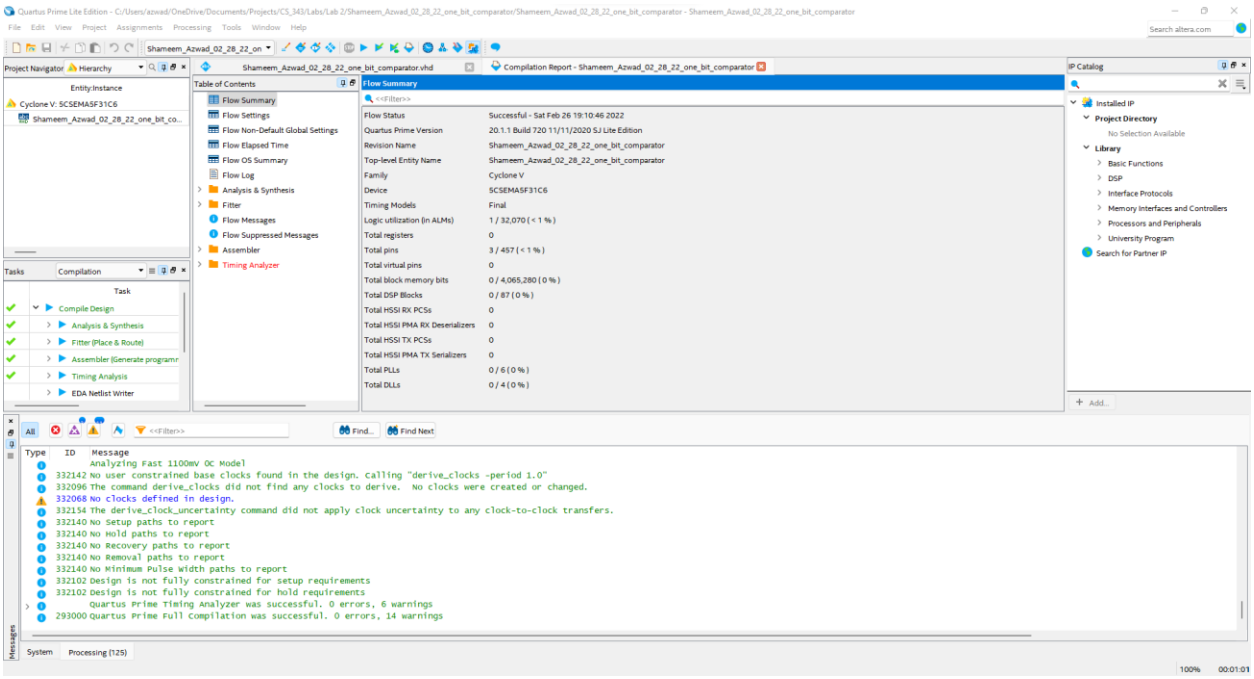

Figure 30: VHDL code compiled successfully on Quartus

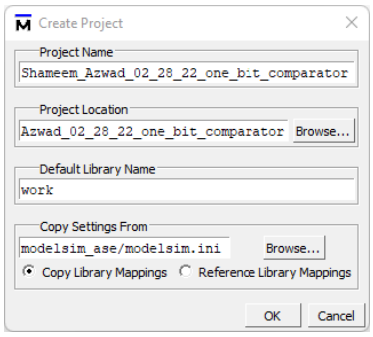

Figure 31: Project created on ModelSim

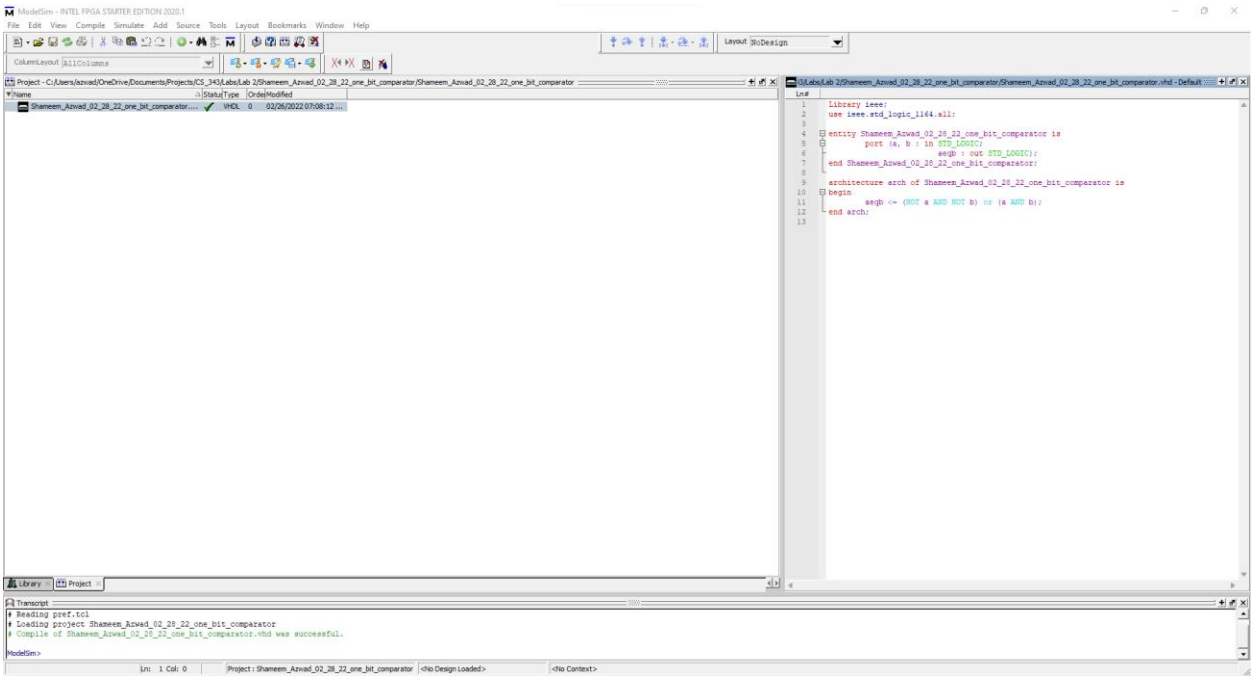

Figure 32: Project compiled successfully on ModelSim

Task A2) Optimize other comparators accordingly.

2-bit Comparator

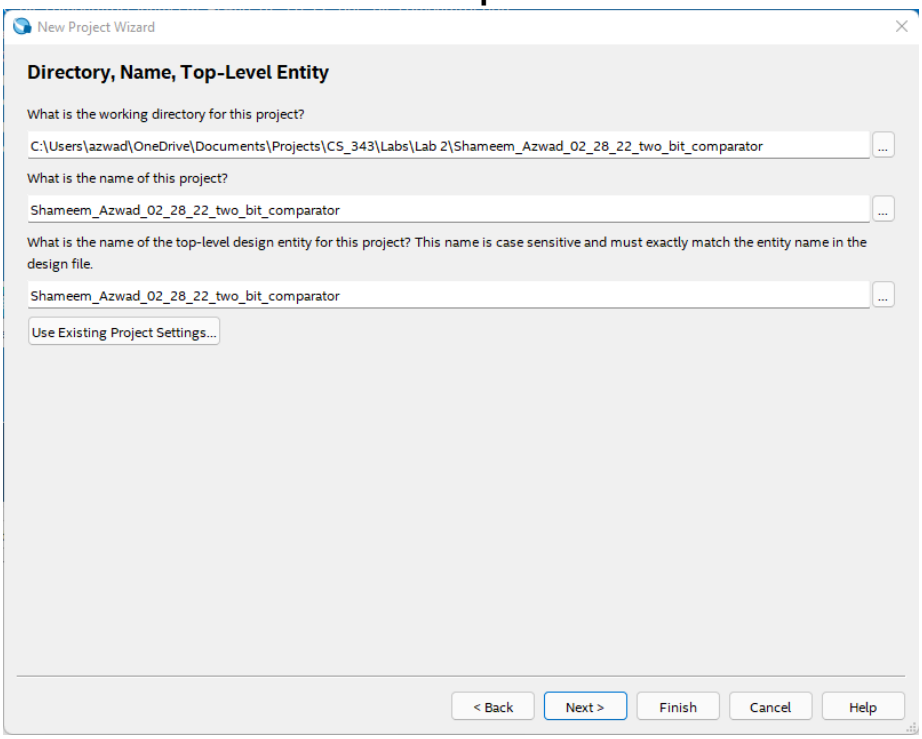

Figure 33: Optimized two-bit comparator created

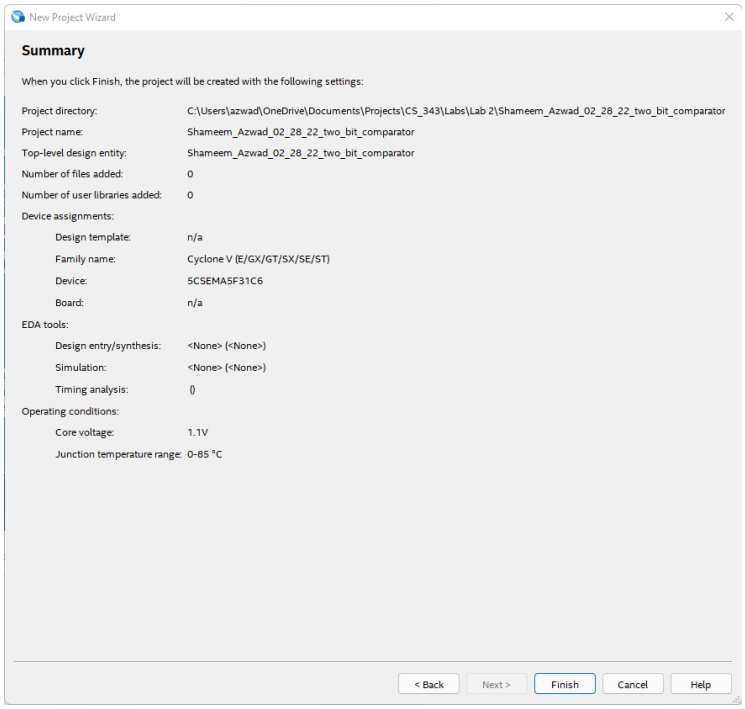

Figure 34: Project Summary

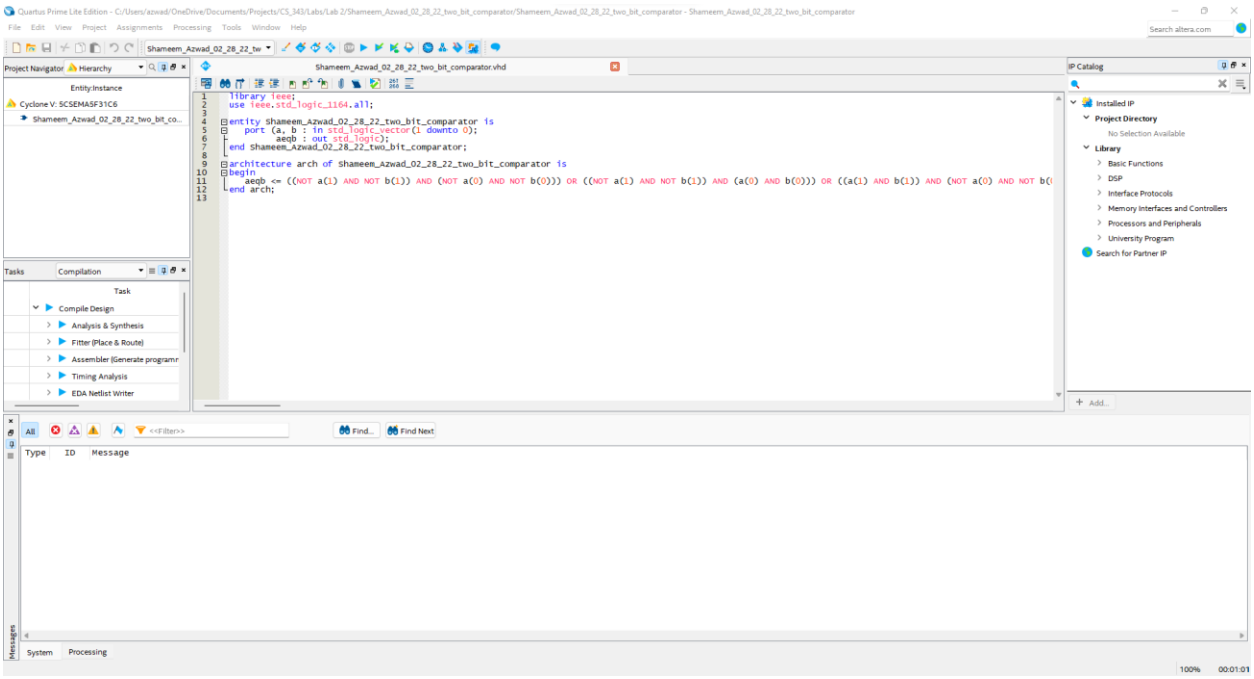

Figure 35: Optimized 2 bit comparator code.

Optimization is done by replacing multiple lines of Boolean operations with a one line Boolean expression.

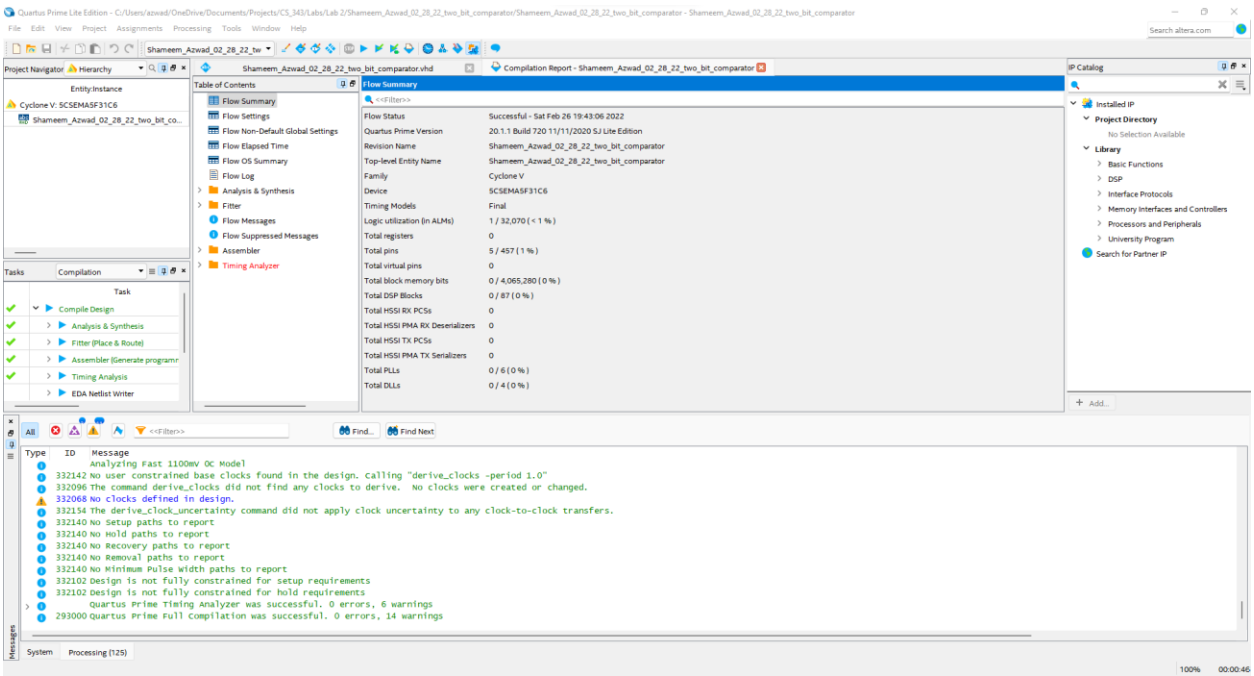

Figure 36: Optimized VHDL code compiled successfully on Quartus.

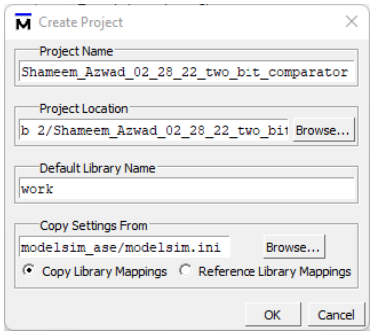

Figure 37: Project on ModelSim

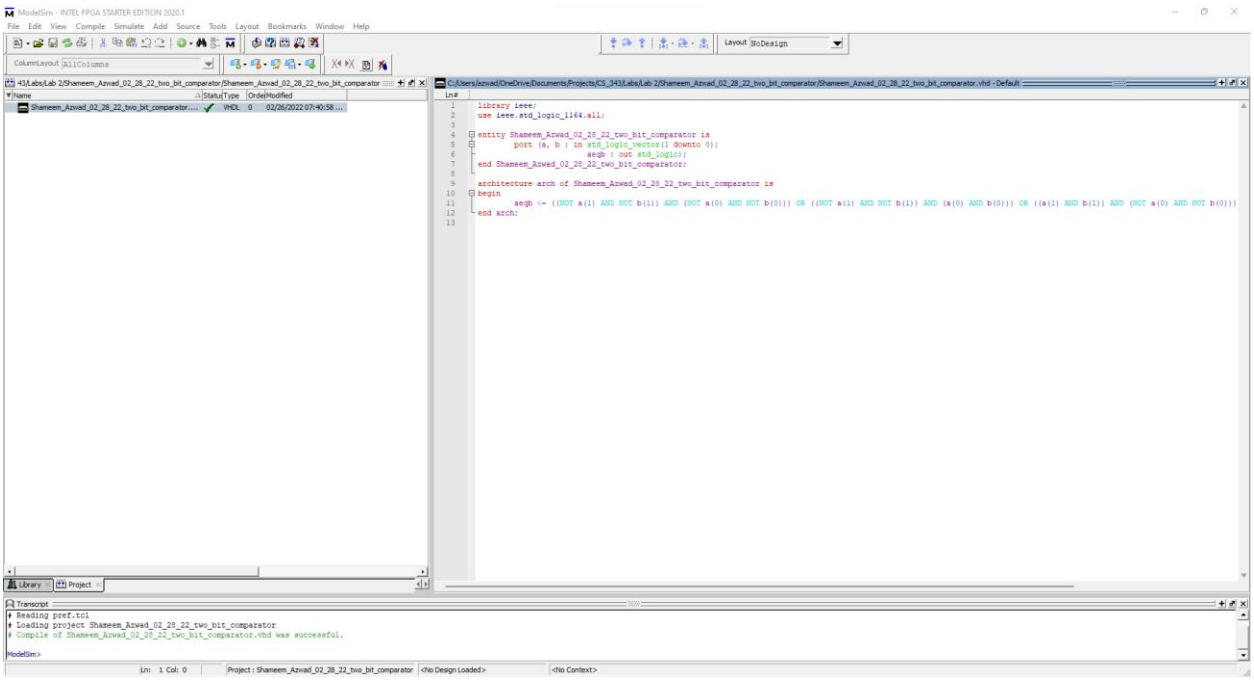

Figure 38: VHDL code compiled successfully on ModelSim

8-bit Comparator

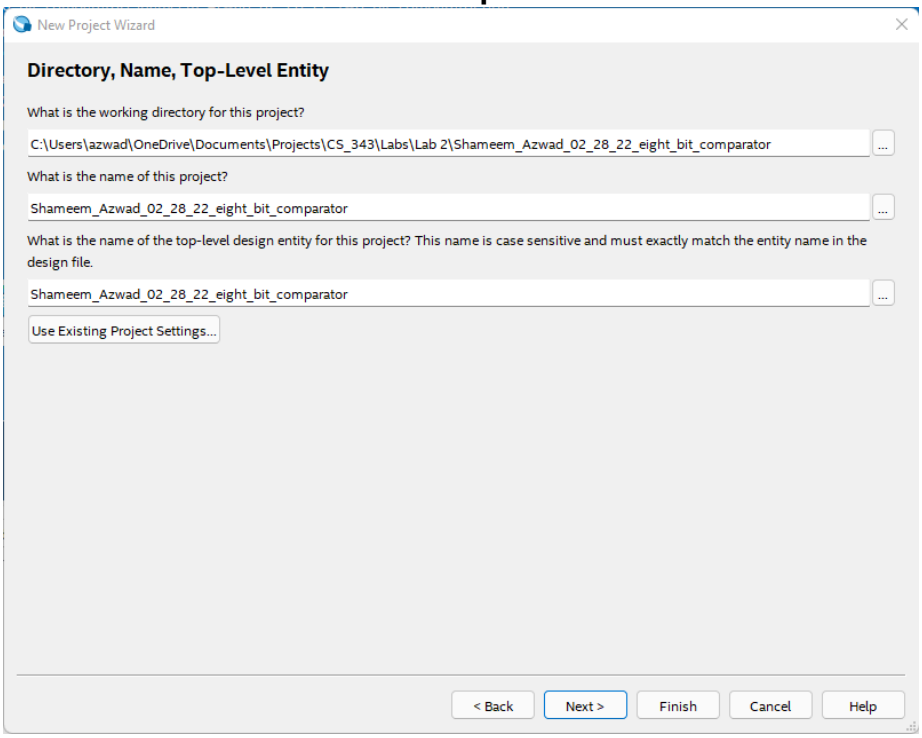

Figure 39: Project Created for Optimized eight bit comparator

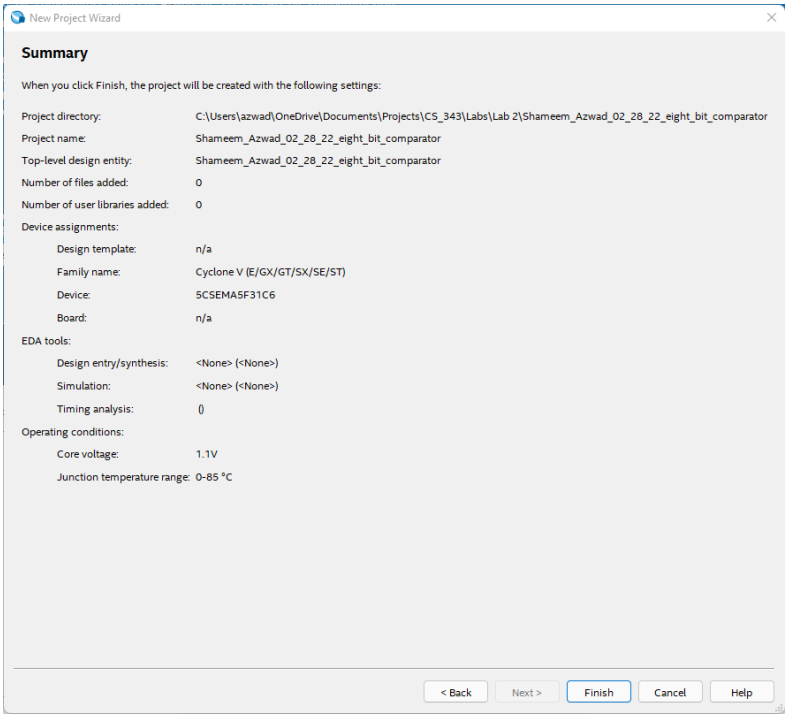

Figure 40: Project Summary

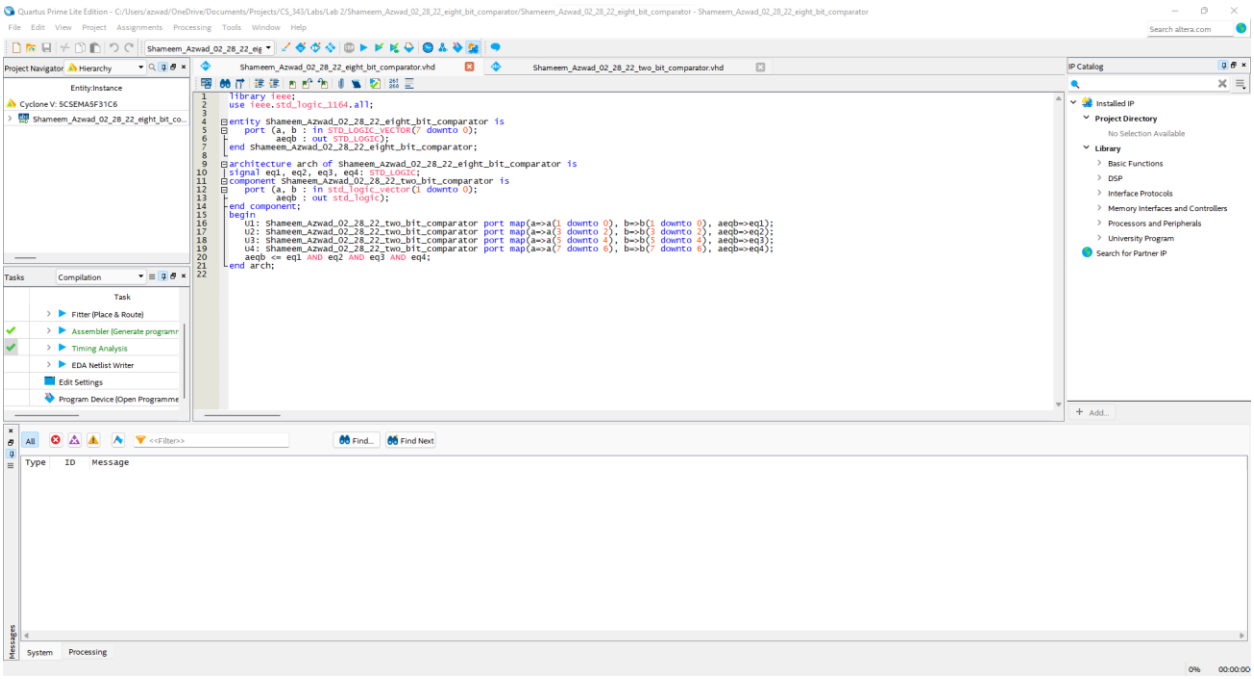

Figure 41: Optimized Eight bit Comparator.

Optimization is done by using the optimized two bit comparator instead of one bit which cuts down the code by 4 lines because you need 4 two bit comparators instead of 8 one bit comparators.

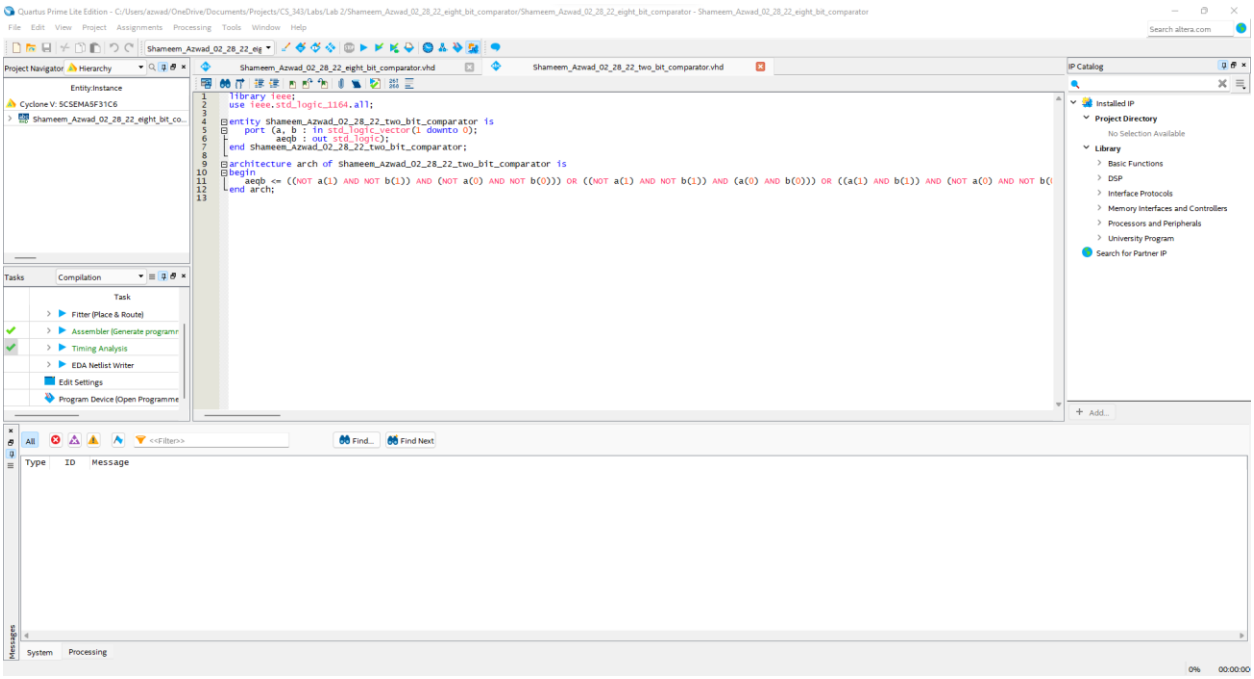

Figure 42: Utilizing Optimized two bit comparator as a component in the optimized eight bit comparator

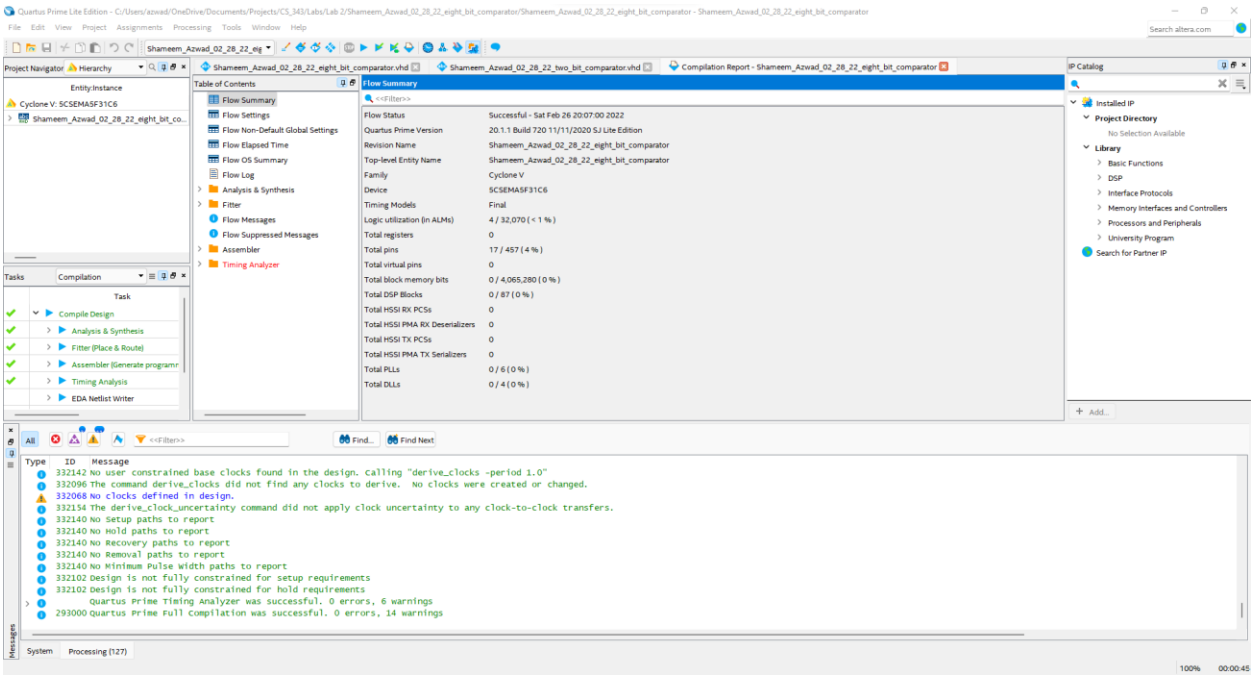

Figure 43: Optimized VHDL code successfully compiled on Quartus

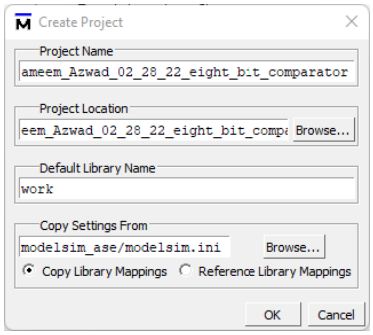

Figure 44: Project created on ModelSim

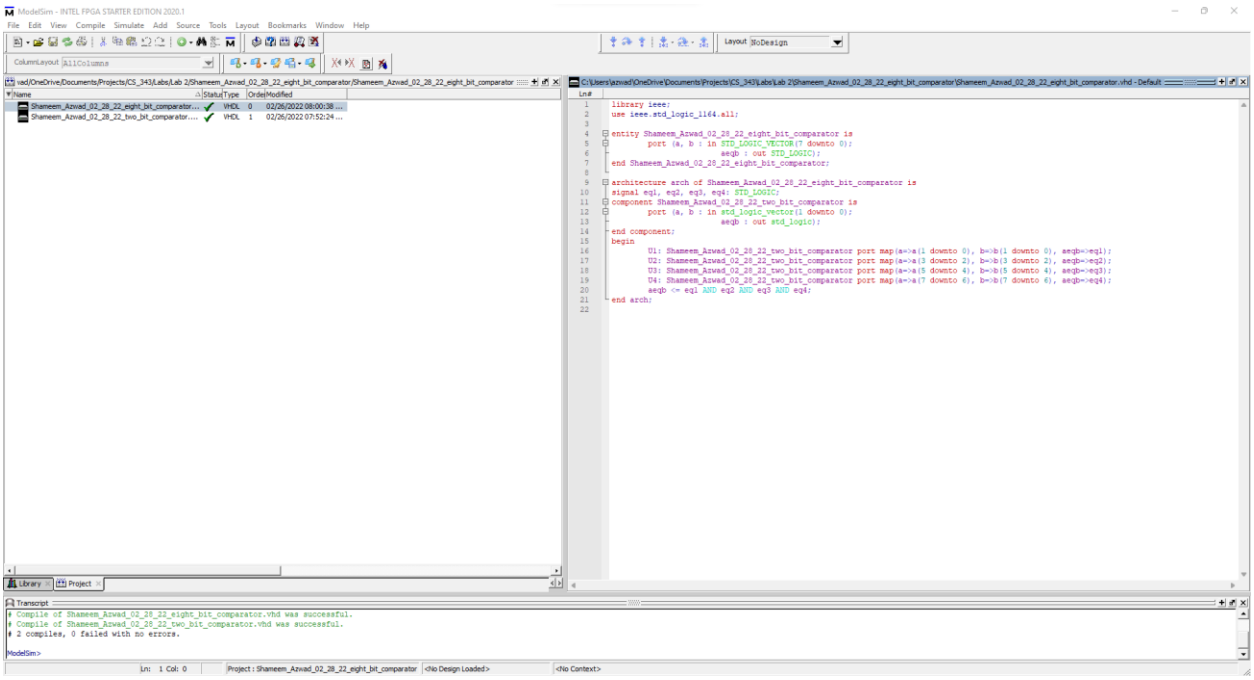

Figure 45: VHDL code compiled successfully on ModelSim

Task A3) Design and test using ModelSim optimized 2 and 8 bit comparators in VHDL.

Testing optimized 2 bit comparator

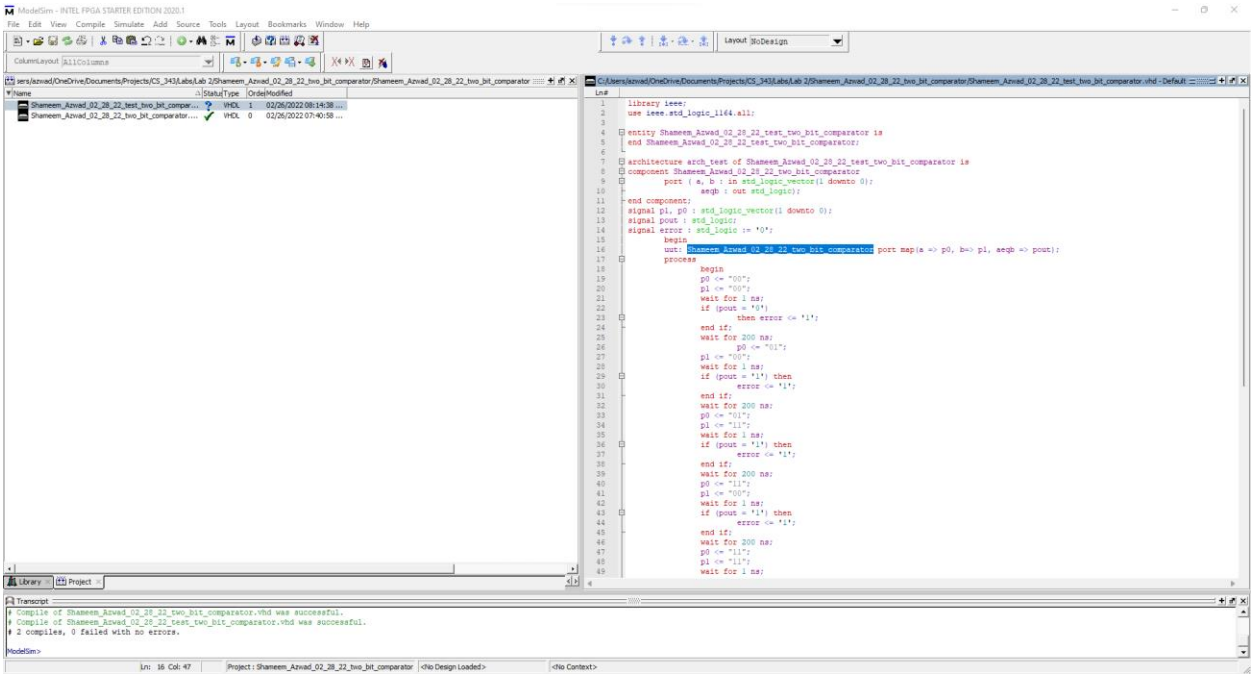

Figure 46: Testbench for optimized 2 bit comparator

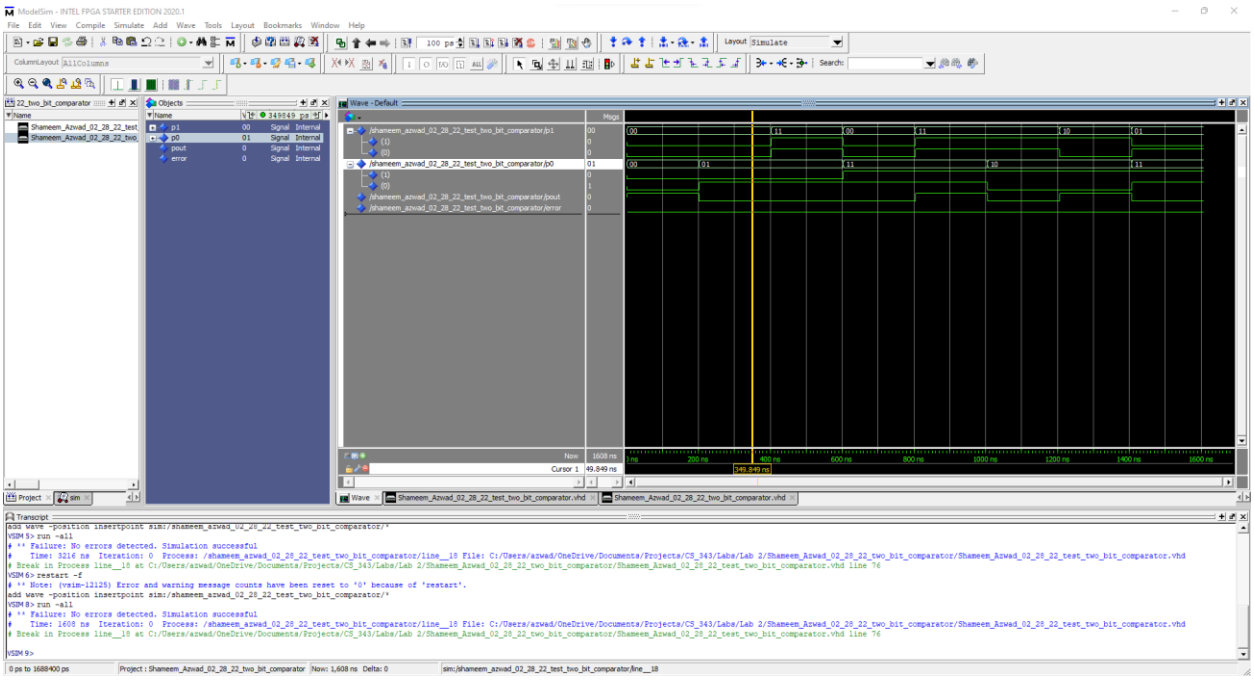

Figure 47: Simulation for optimized 2 bit comparator

8 bit Comparator

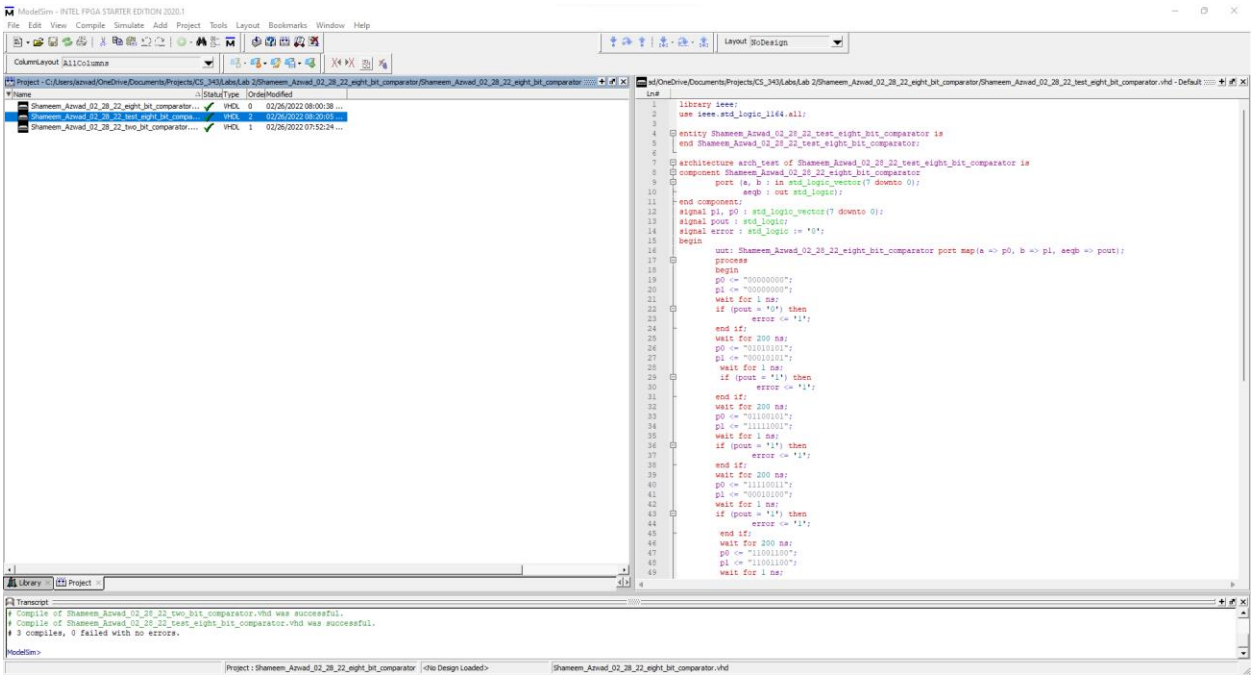

Figure 48: Testbench for optimized 8 bit comparator

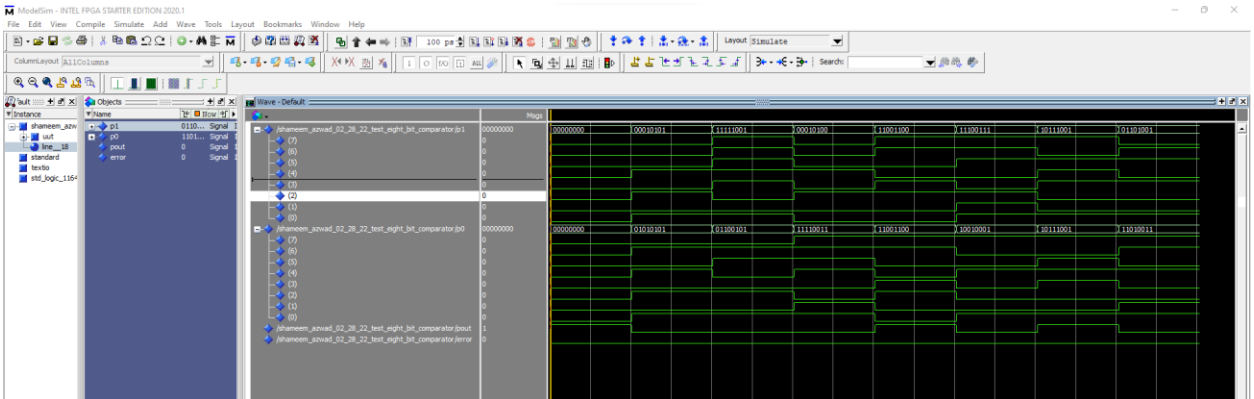

Figure 49: Simulation wavelength for optimized 8 bit comparator

Explanation

Task 0) The tutorial for the equal circuit has been followed. The code and images show that the VHDL code successfully compiles in Quartus and ModelSim for both the circuit and the testbench. The testbench simulation is also correct because it follows truth table accordingly.

Task 1) The VHDL code for the 2 bit comparator and it's testbench compiles successfully in both Quartus and ModelSim. The correctness of the 2 bit comparator was tested using the ModelSim and the testbench from the tutorial. The waveform obtained from running the testbench simulation verifies the correctness of the 2 bit comparator.

Task 2) The VHDL code for the 8 bit comparator and it's testbench compiles successfully in both Quartus and ModelSim. The 8 bit comparator utilizes the 1 bit comparator as components. Furthermore, the 1 bit comparator is correct because it followed the truth table for the comparator. In addition, the correctness of the 8 bit comparator was tested using the ModelSim and the testbench from the tutorial. The waveform obtained from running the testbench simulation verifies the correctness of the 8 bit comparator.

Task A1) The 1 bit comparator used to have three lines of Boolean operations, which was optimized by merging lines 12, 13, 14 into one line. The optimization was completed merging variables p0's and p1's logic. Essentially, the Boolean operation for p0 and p1 was put in place of the variables in line 12.

Task A2) The 2 bit comparator and the 8 bit comparator were also optimized accordingly. In the case of the optimized *2 bit comparator*, the variables p0, p1, p2, p3 which all had different Boolean operations which took up a line of space each. In order to optimize this the same process as the 1 bit comparator was taken. The Boolean operations was merged lines 12, 13, 14, 15, 16 into a one line Boolean operation which reduced the code by 4 lines of code. Therefore, the optimized 2 bit comparator was optimized because it gained a reduction of 4 lines of code. In the case of the optimized *8 bit comparator* the comparator utilized components of the optimized 2 bit comparators instead of 1 bit comparators, which made the circuit use four optimized 2 bit comparators instead of eight 1 bit comparators. Doing this allowed the optimized 8 bit comparator to reduce 4 lines of code, which optimized the 8 bit comparator by reducing the number of lines of code.

Task A3) The optimized 2 bit comparator and the optimized 8 bit comparator was designed and tested using the ModelSim testbench and the waveforms still match the correct outputs just like the non-optimized versions. The waveforms for both comparators still match the correct simulation results and have no error. This shows that the comparators have been designed and tested for correctness.

Conclusion

The lab was very beneficial to gain a better understanding of optimization VHDL coding because it made the designer first create a component using the given equation and then made them think about how to optimize by reducing lines of code. Furthermore, the lab had a great effect on understanding the benefits of using components with either port maps or logic vectors and how to optimize. Essentially, the lab was a great way to gain more practice with VHDL in general and a way to learn not only about designing circuits but also optimizing circuits and even after they are made.