CS342/CS343
Instructor: Professor Izidor Gertner
Spring 2022
Lab 1, Azwad Shameem, 2/13/2022

Table of Contents **Page**

Objective:

Part A. Create the project for the digital circuit based on the tutorial.
Part B. Create a 2:1 multiplexer where each signal is one bit.
Part C. Create a 2:1 multiplexer where each of the signals are 32 bits, beside the selector signal which is only 1 bit.

Code:

(Part A)

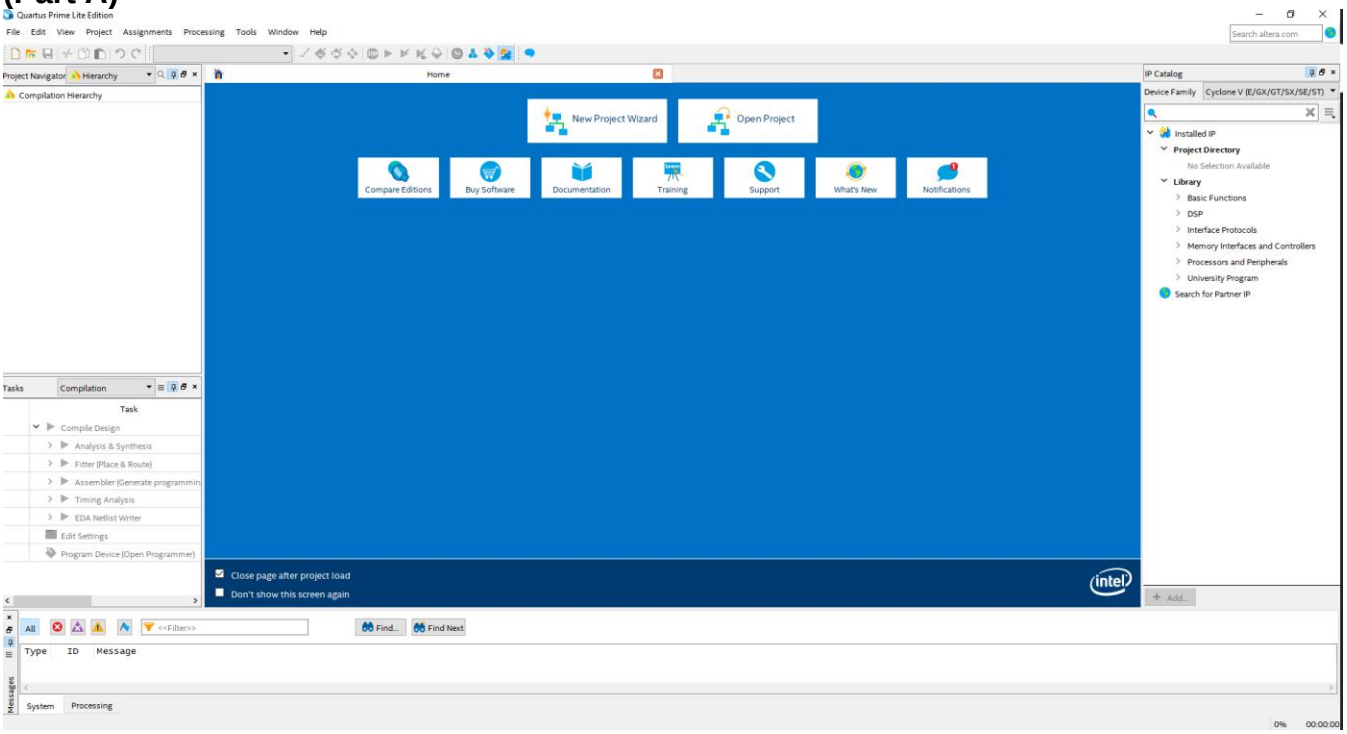

Figure 1: The Main Quartus Prime Display Screen

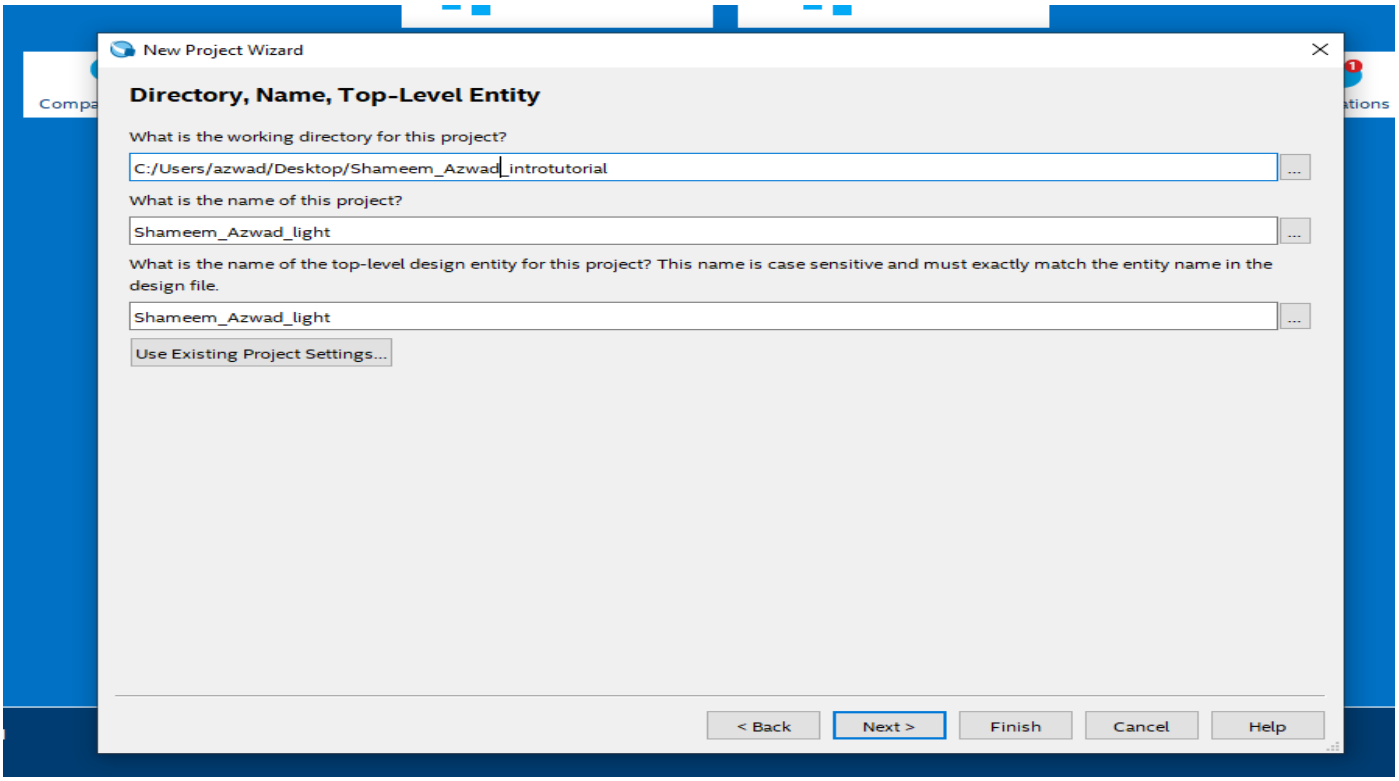

Figure 2: Creation of the project

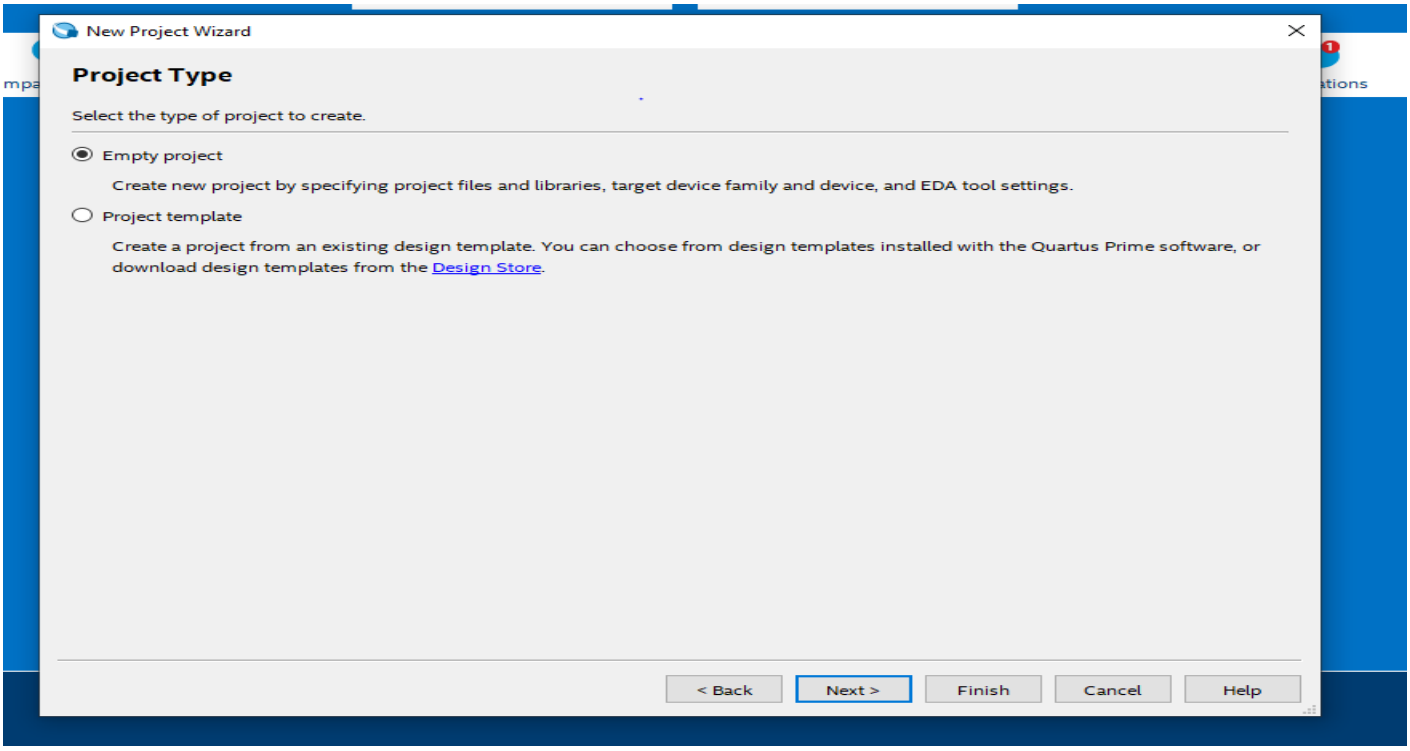

Figure 3: The Project created as an Empty project

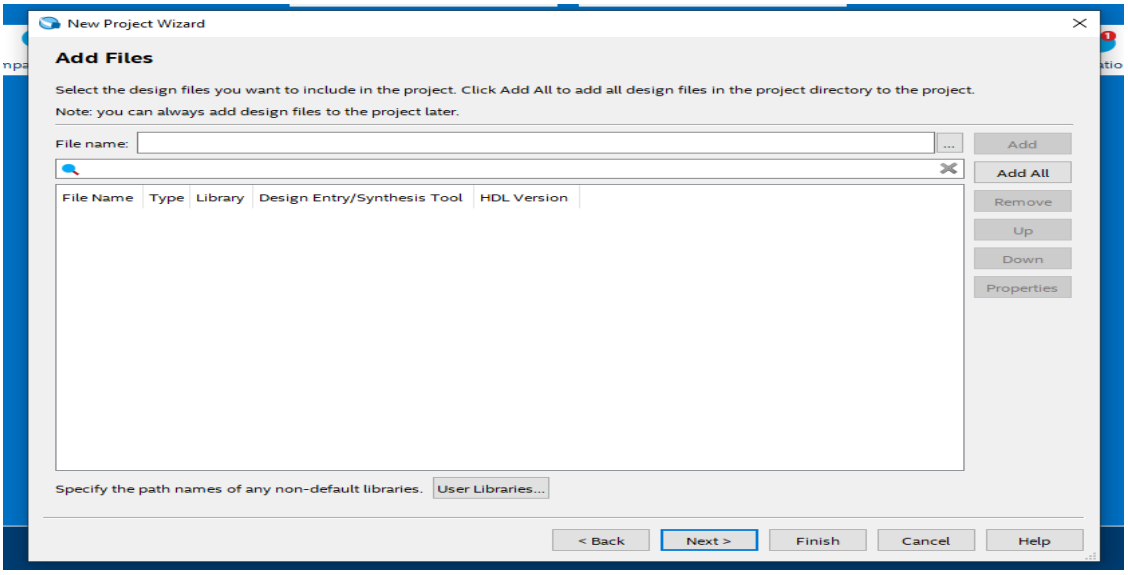

Figure 4: No extra files were added just like in the tutorial

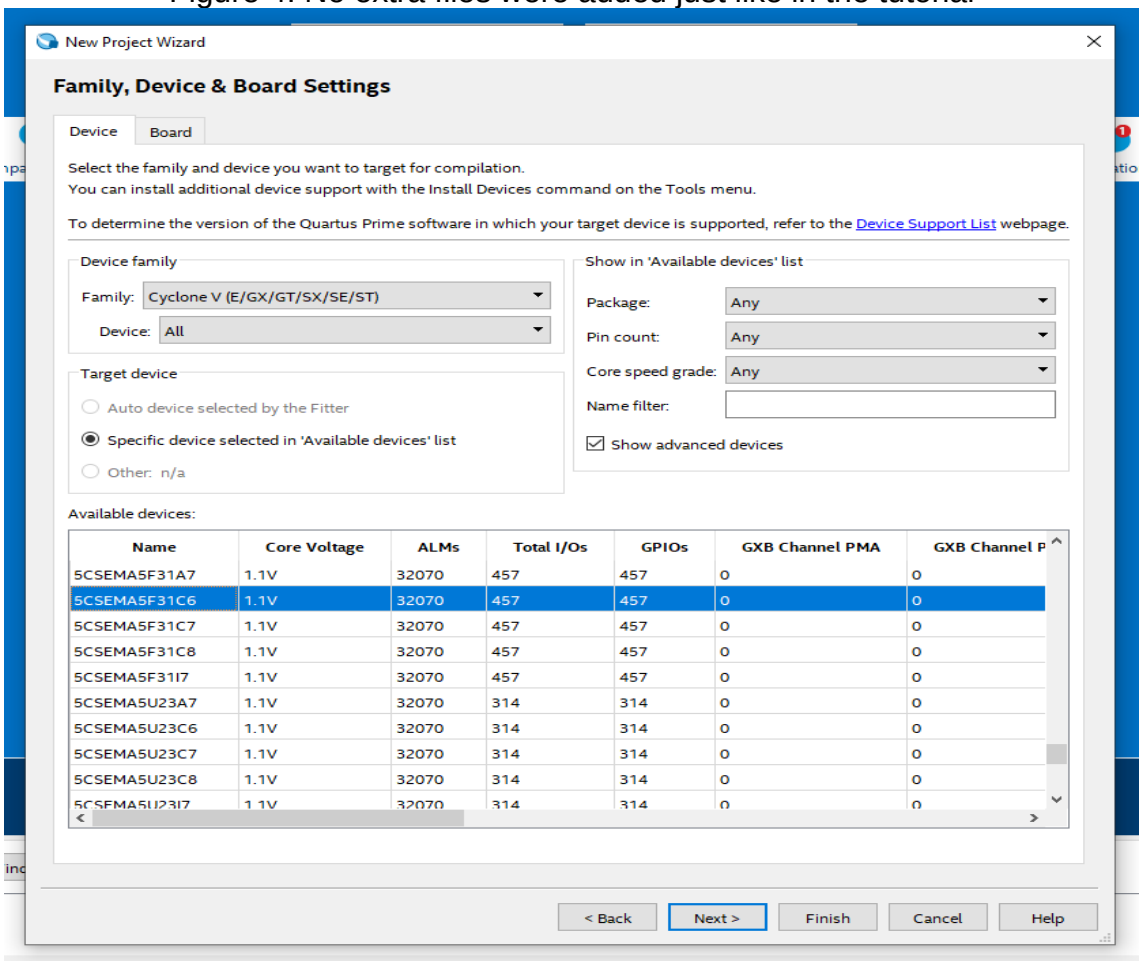

Figure 5: The same device used as in the tutorial

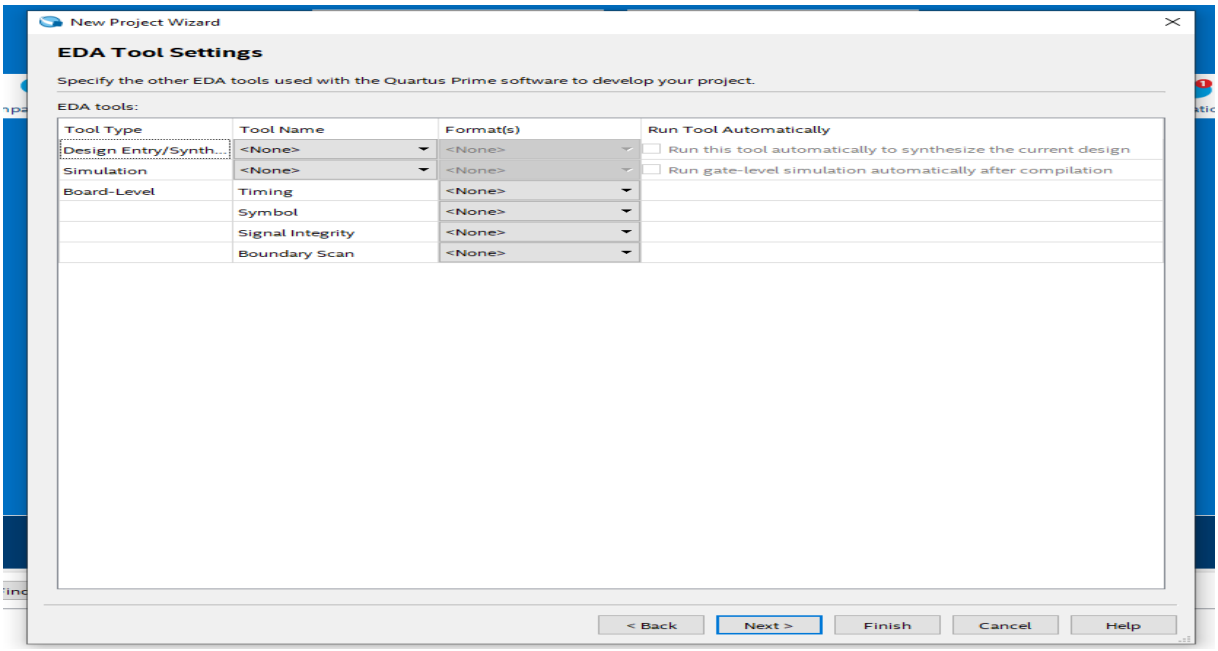

Figure 6: EDA Tools Settings

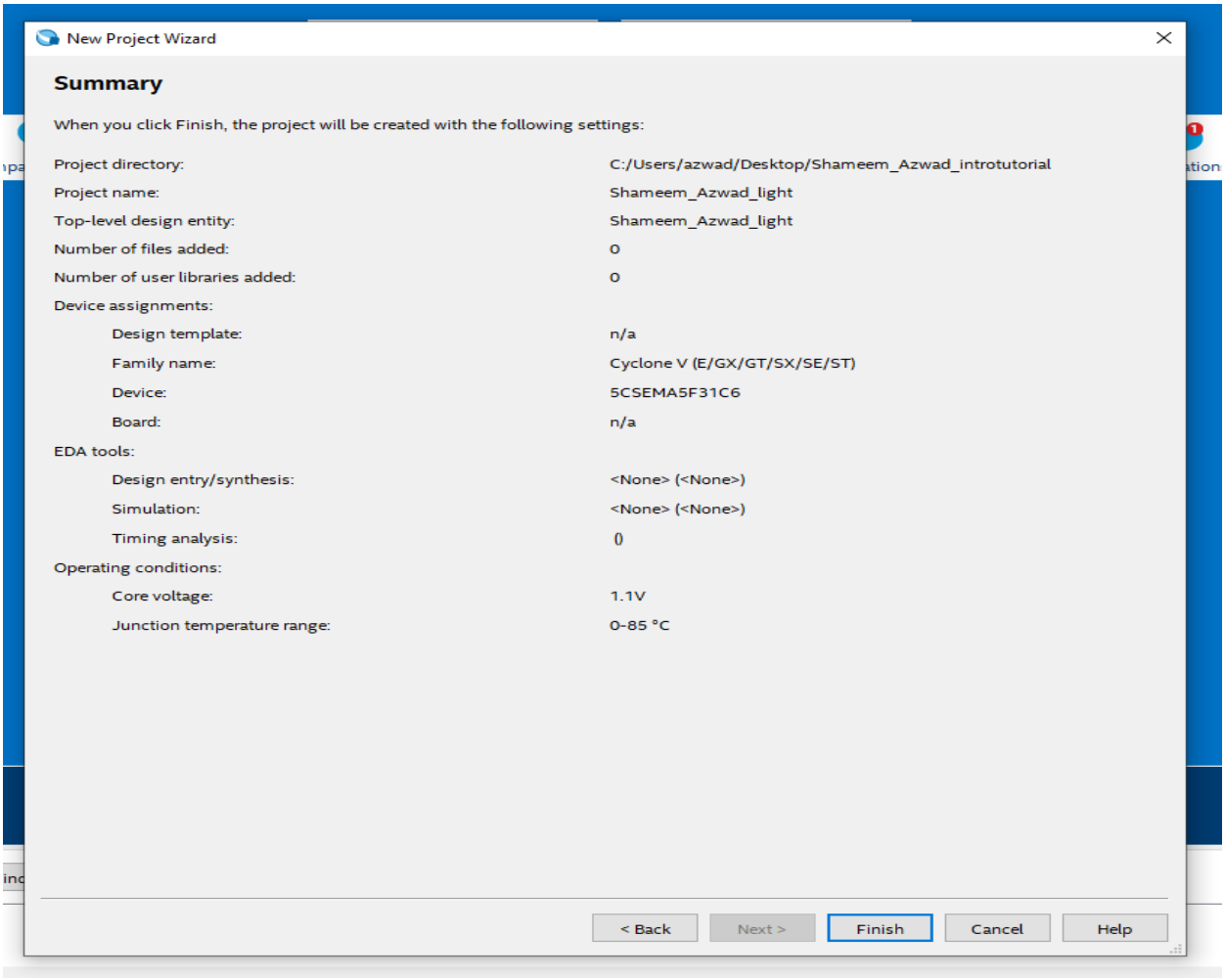

Figure 7: Project Summary

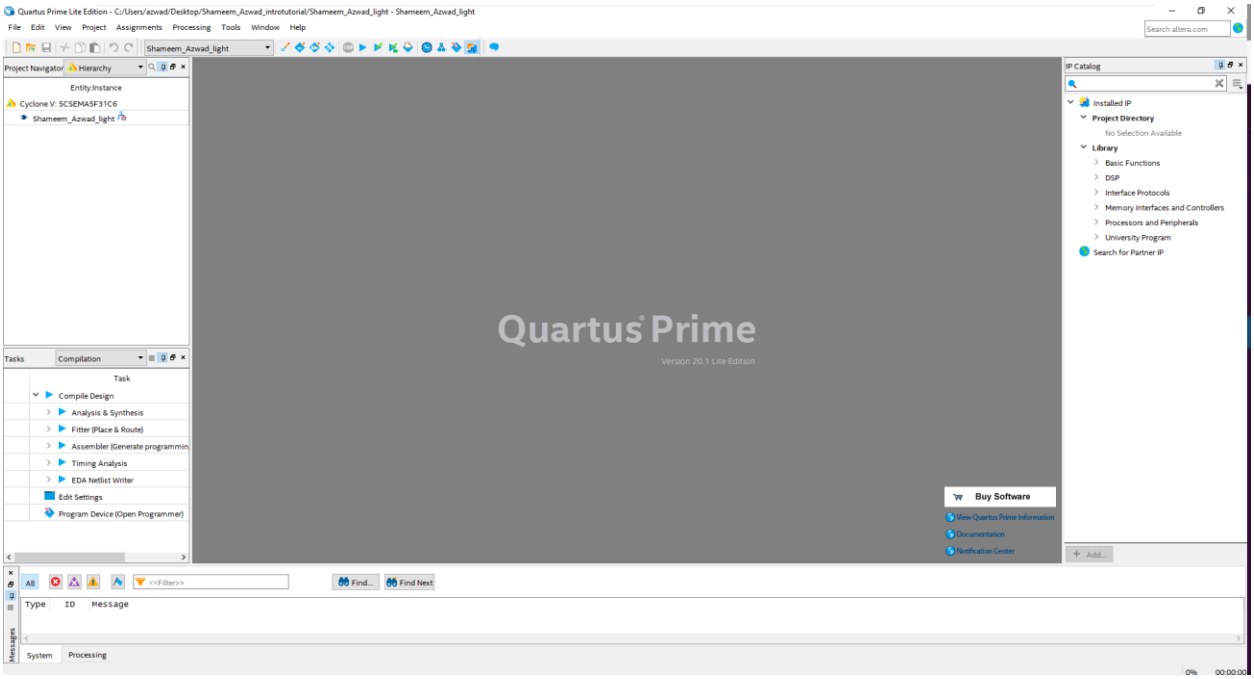

Figure 8: The Quartus Prime window for a created project

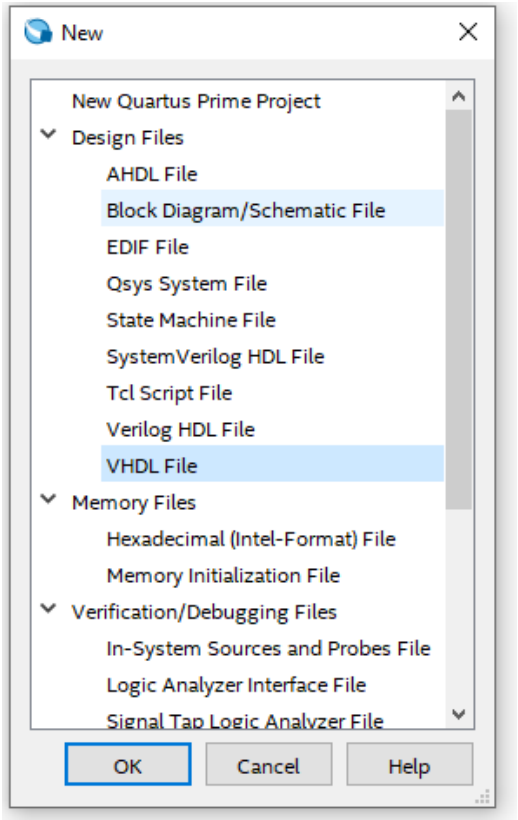

Figure 9: Create a VHDL File

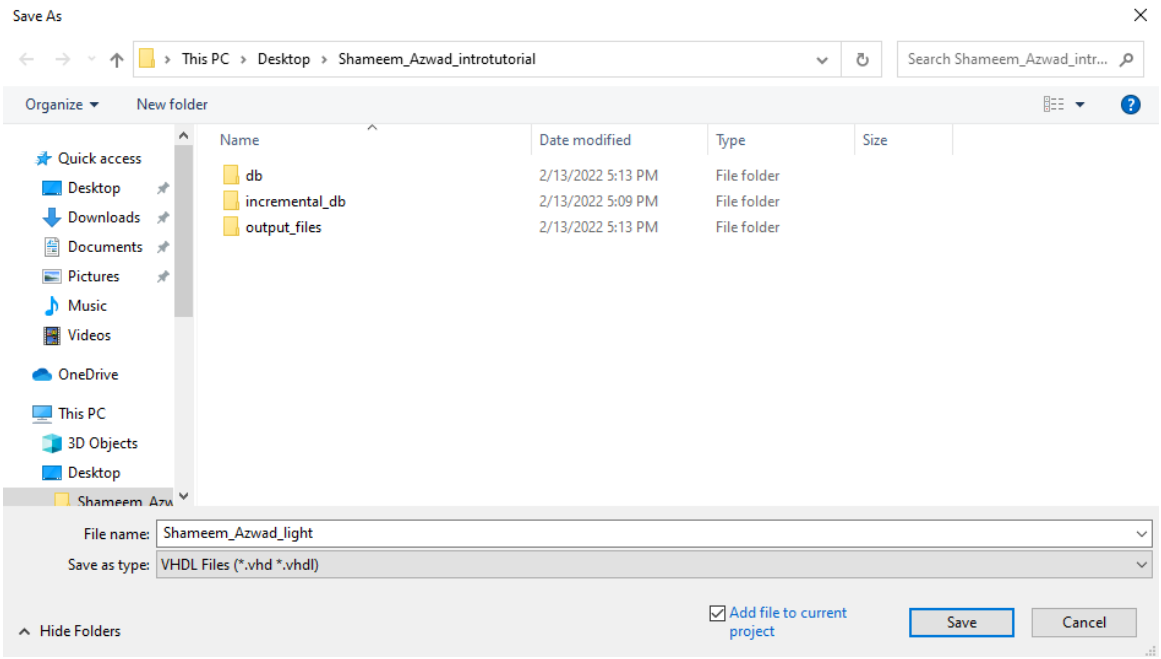

Figure 10: Name the File

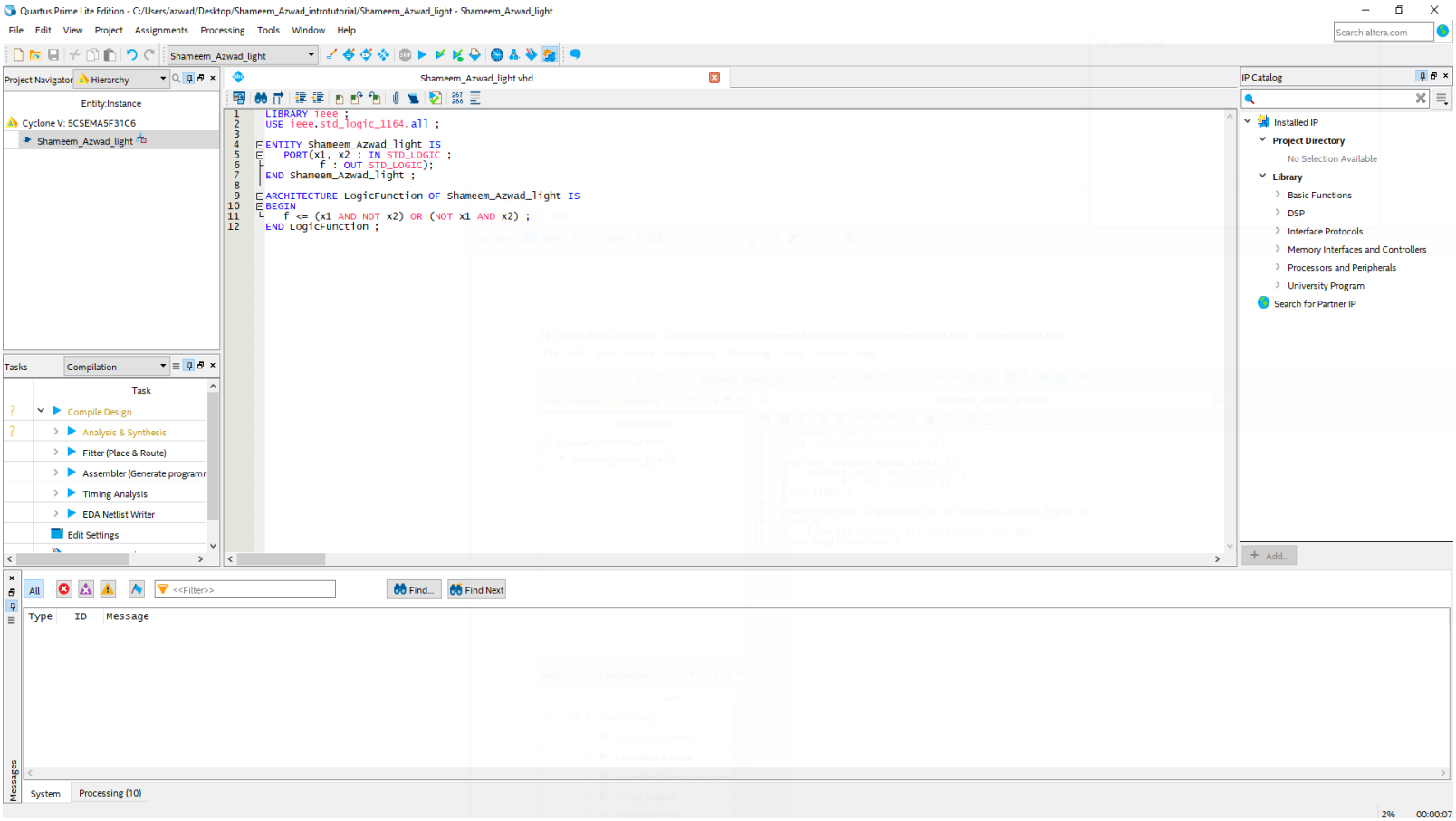

Figure 11: Text Editor Window

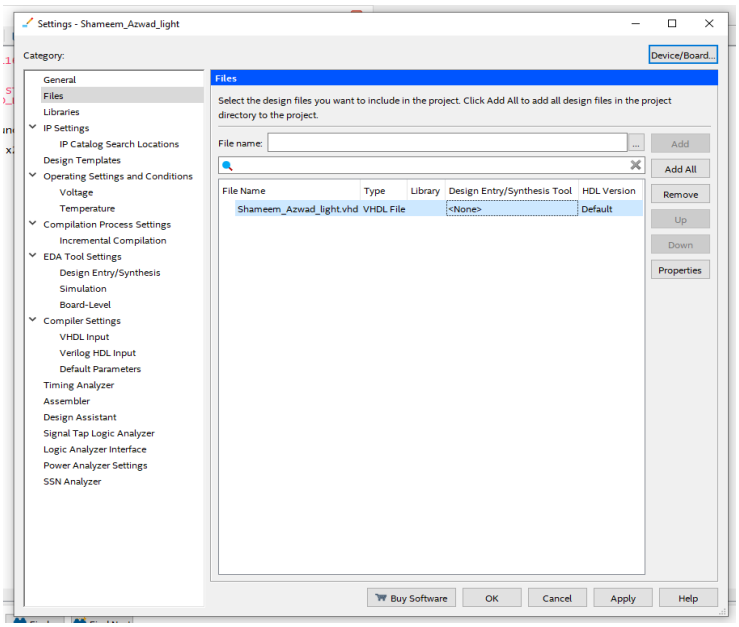

Figure 12: Adds the file to the project
(Image on the left)

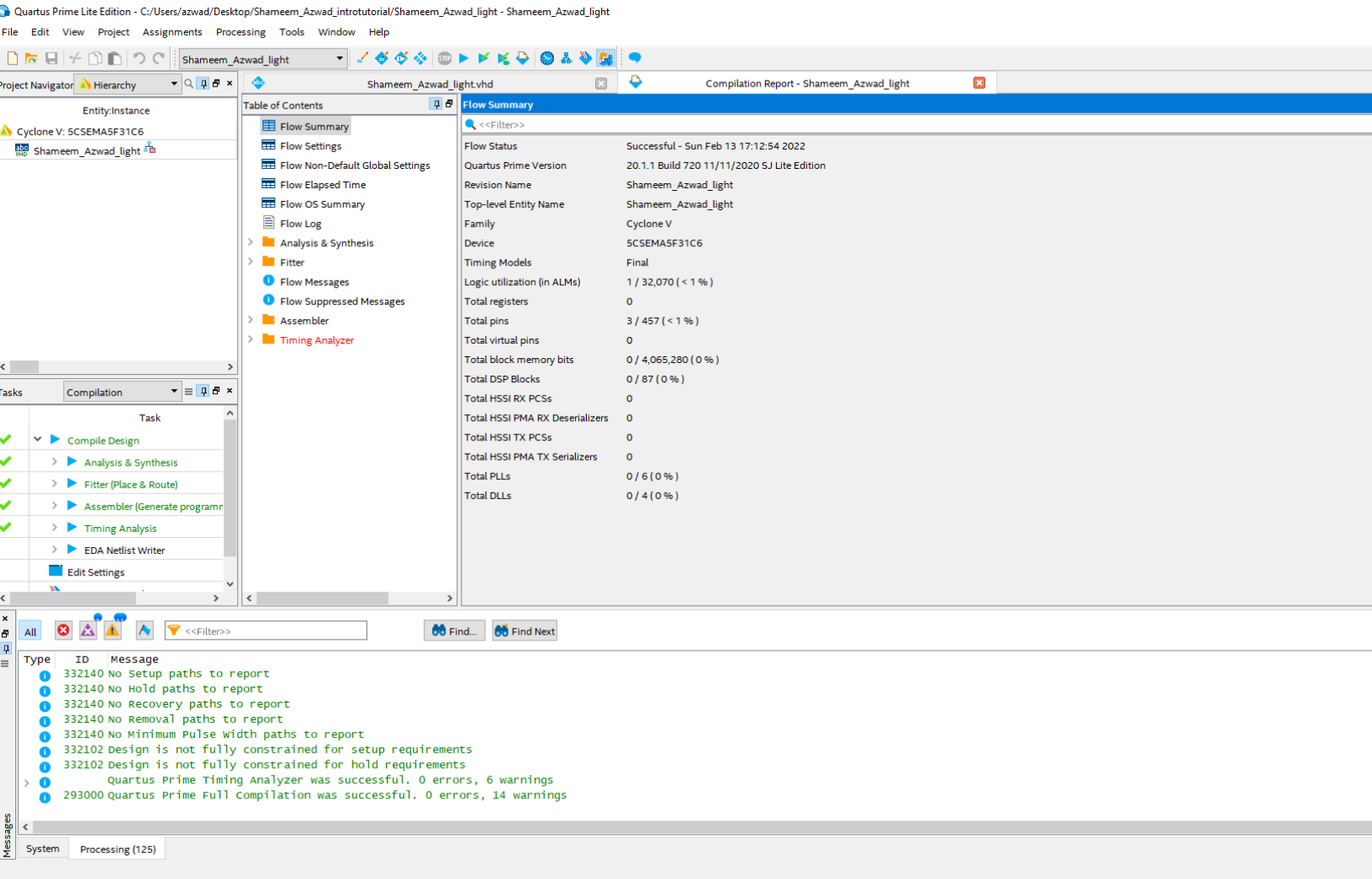

Figure 13: The project displayed after a successful compilation.

(Part B)

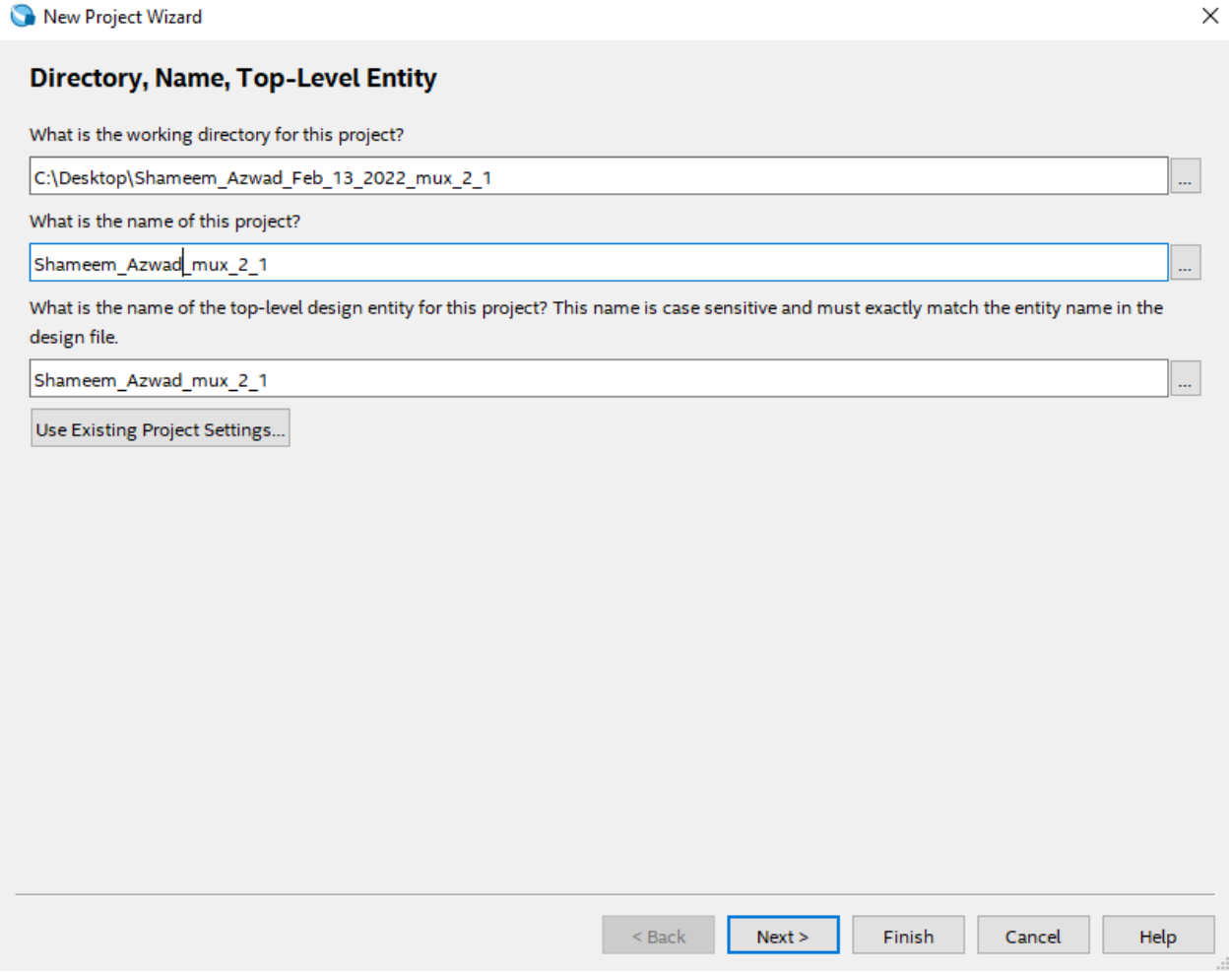

Figure 14: The directory name and the name of the project plus the top-level design entity name.

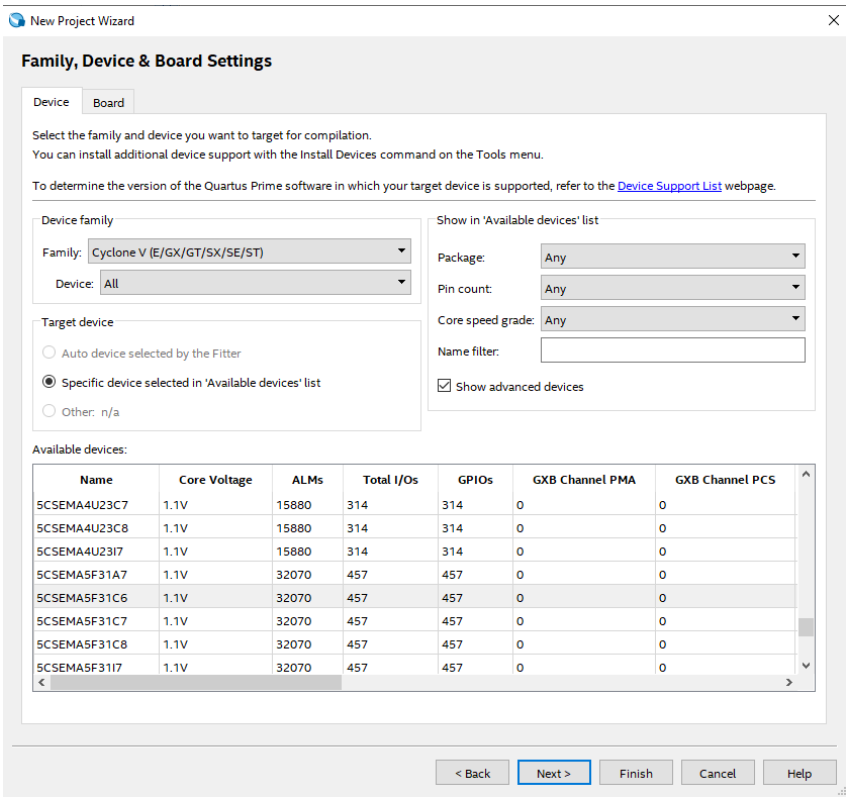

Figure 15: The device picked for this project (Image on the left)

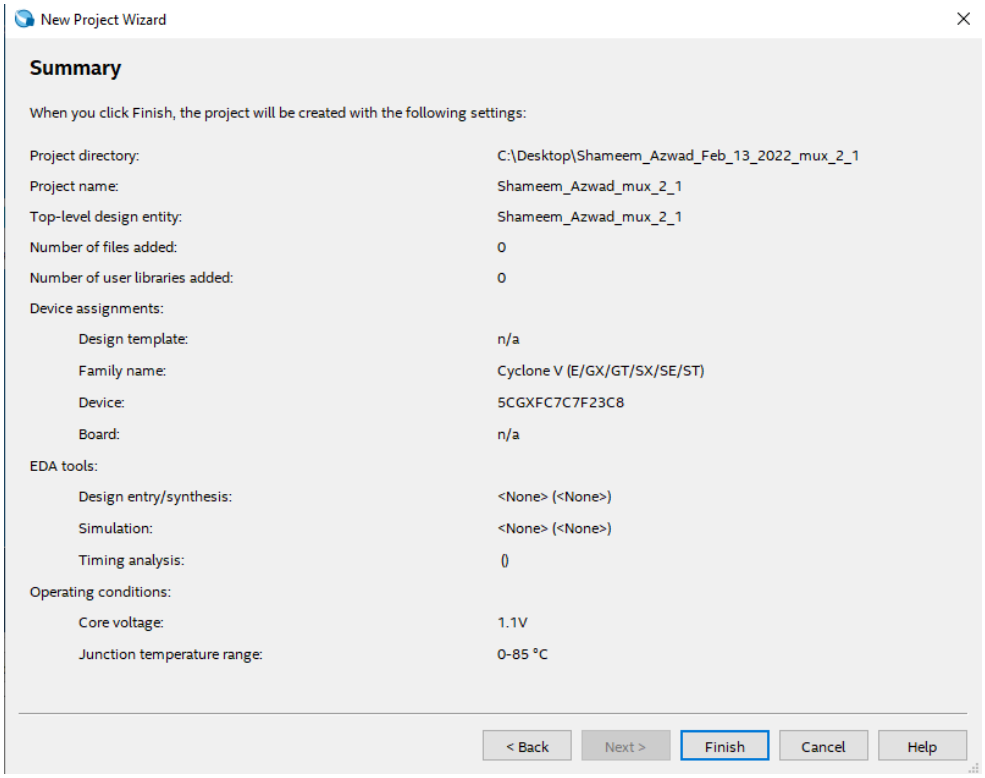

Figure 16: The Project summary
(Image on the left)

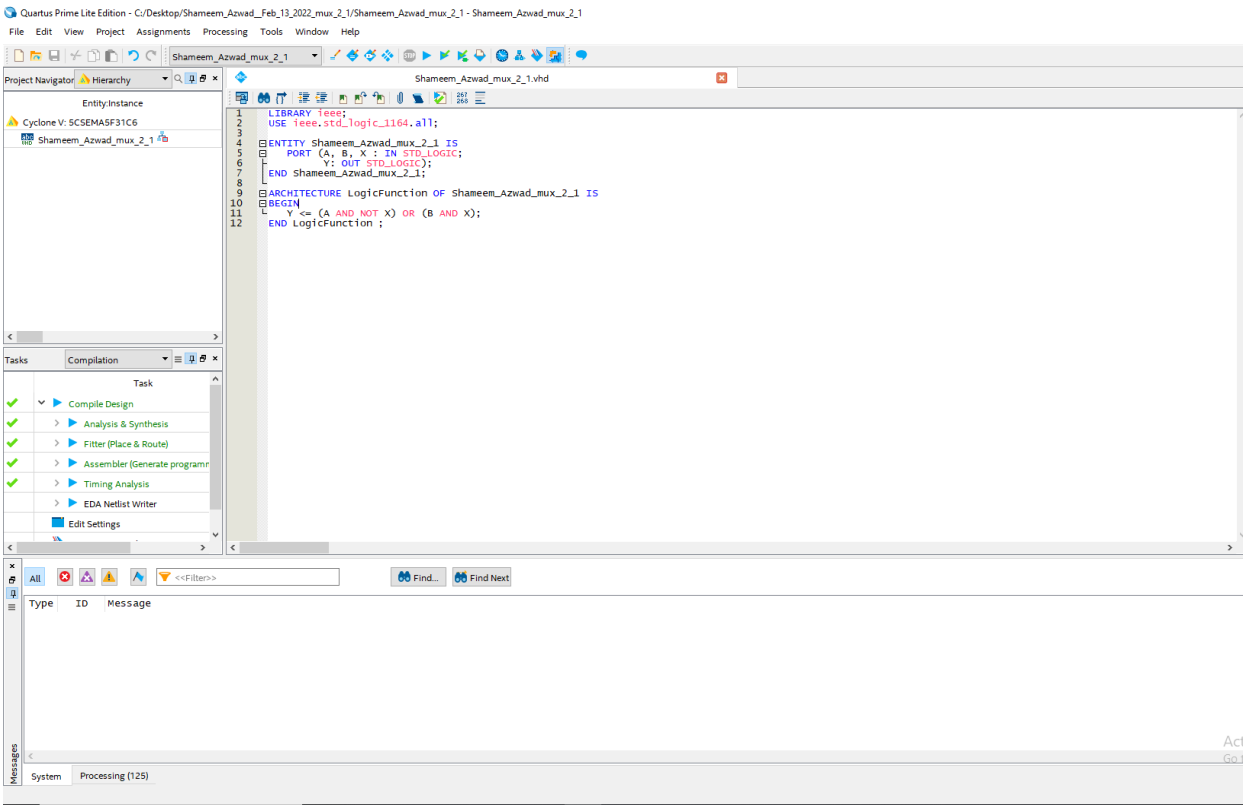

Figure 17: VHDL code for the mux 2:1

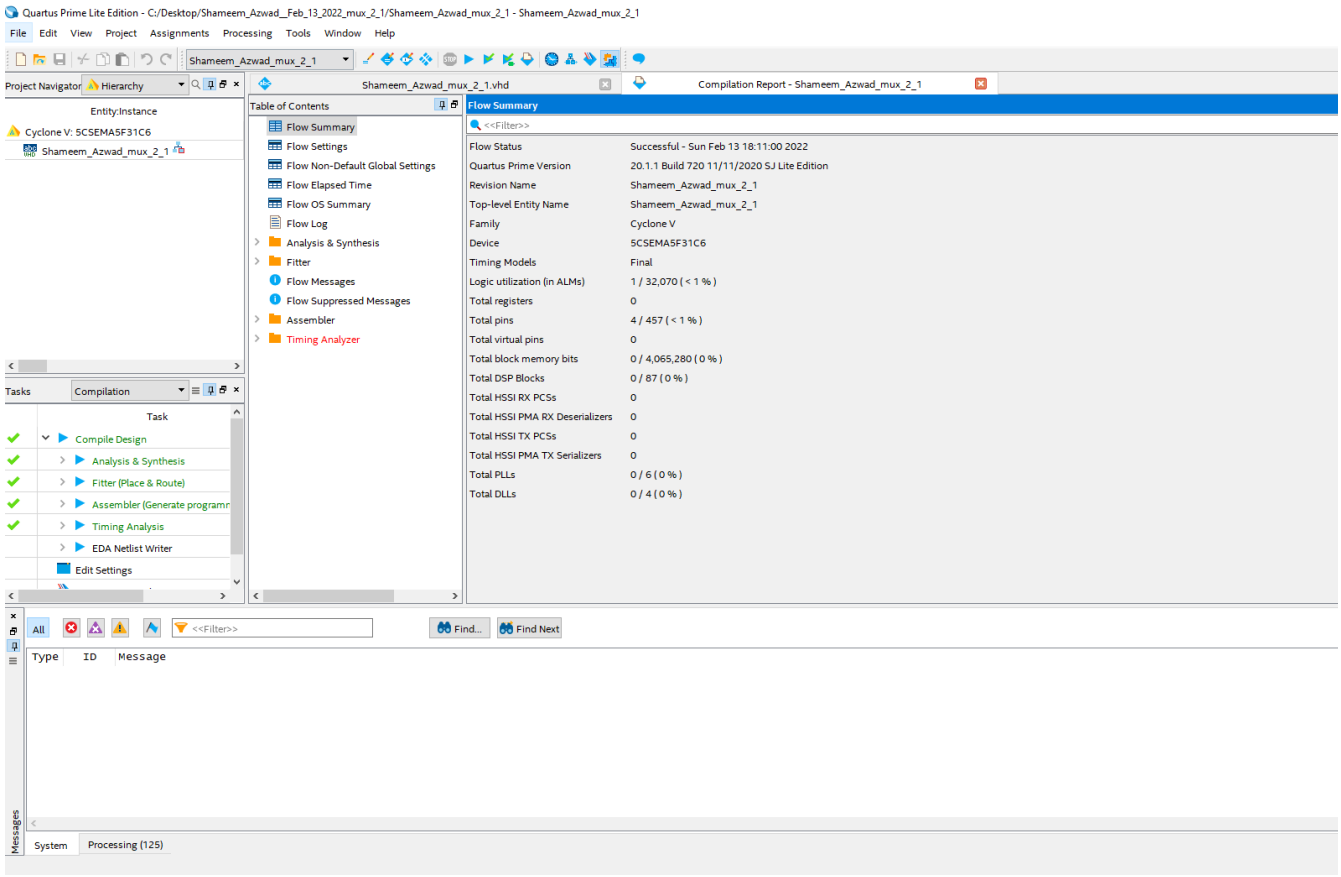

Figure 18: The project compiled successfully.

(Part C)

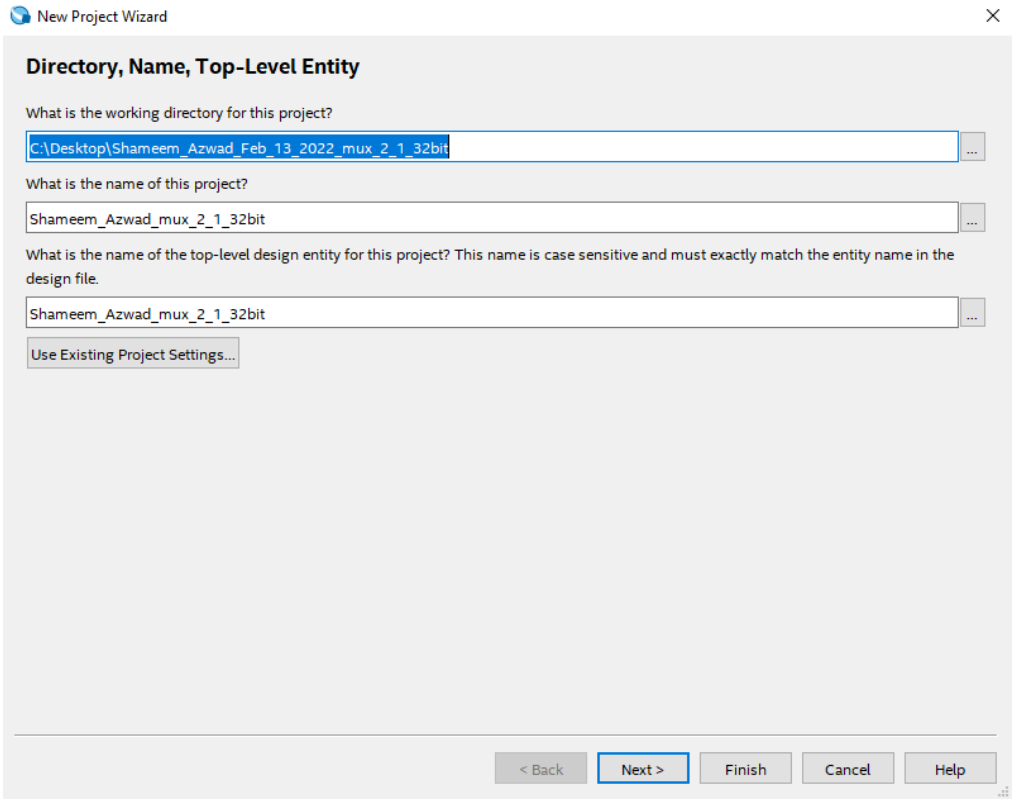

Figure 19:
The directory name and the project name plus the top-level design entity name is shown.

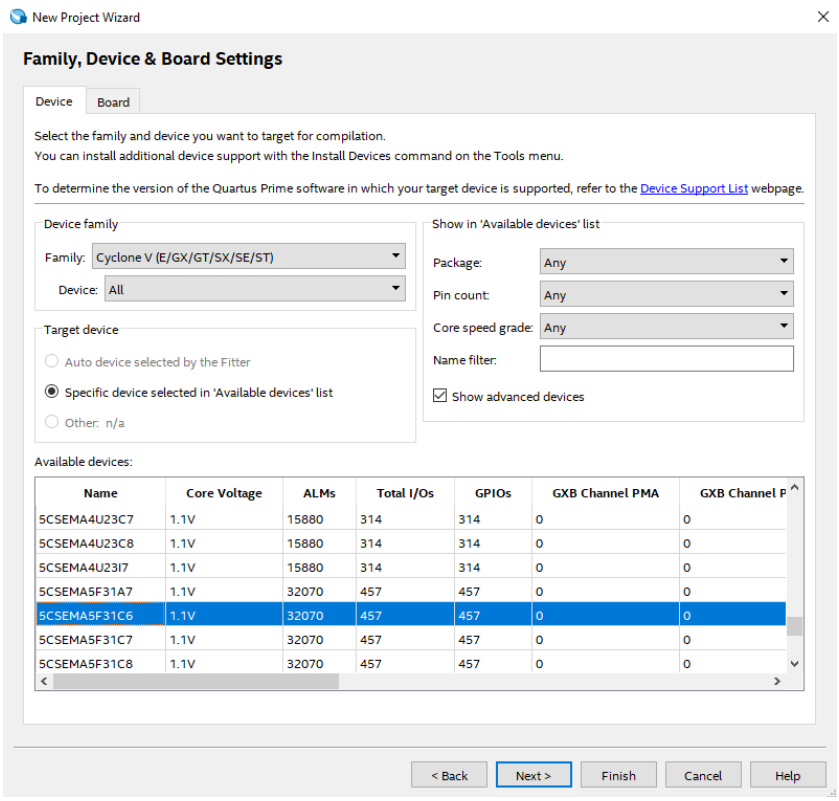

Figure 20:
Family, Device & Board settings displayed.

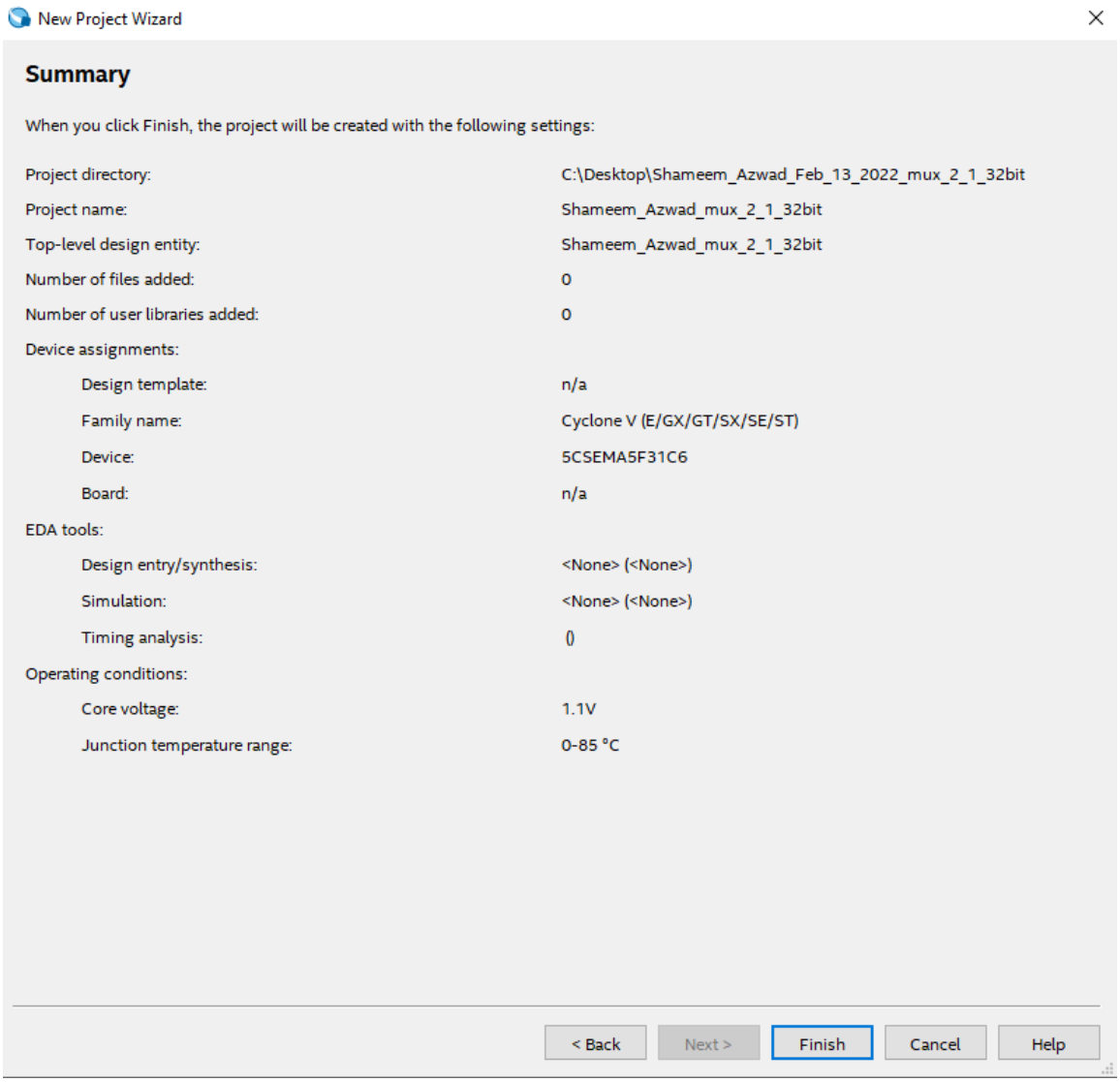

Figure 21: Project Summary displayed

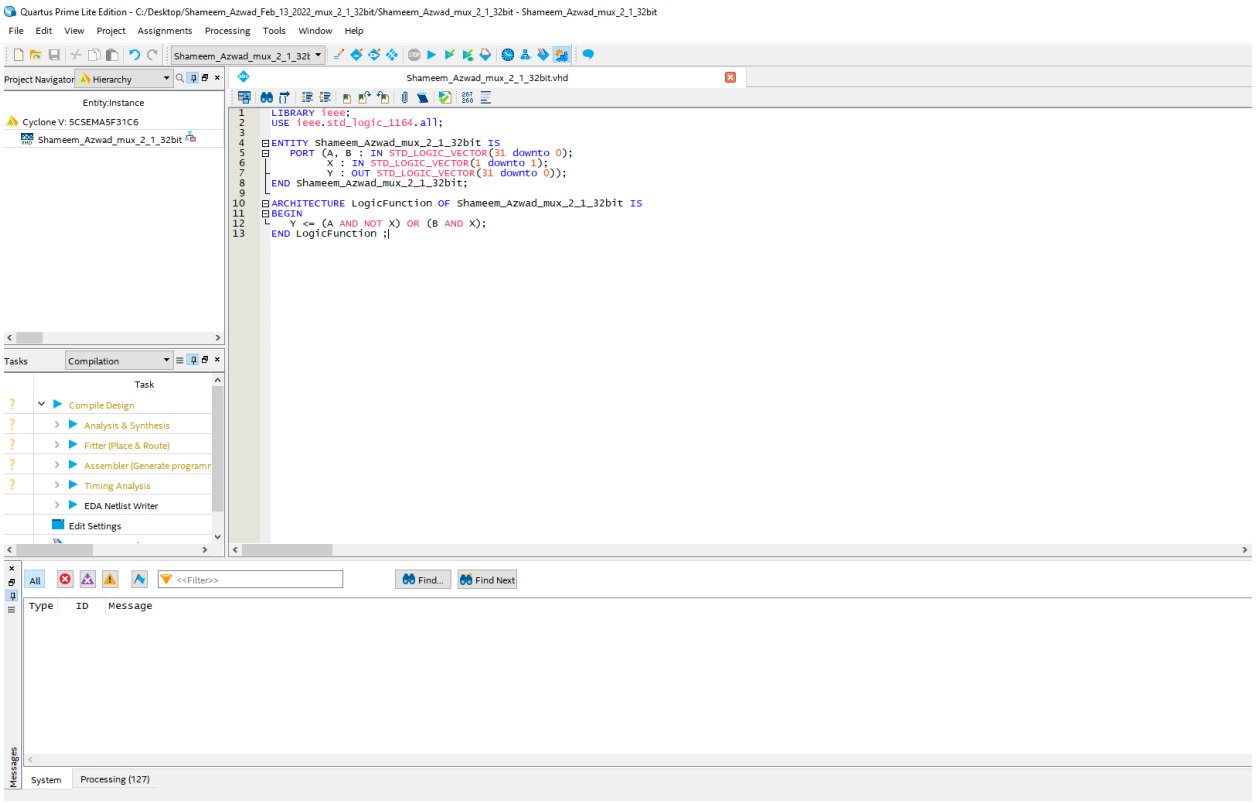

Figure 22: VHDL code used for mux 2:1 32-bit with a 1-bit signal selector

X is listed as a Vector because the circuit runs into errors with inputs as a bit. Furthermore, we know X is the signal selector with a single bit, which means X needs to be a single bit. Therefore, in order to solve the issue we used a std_logic_vector(1 downto 1) which is basically a vector with only one bit.

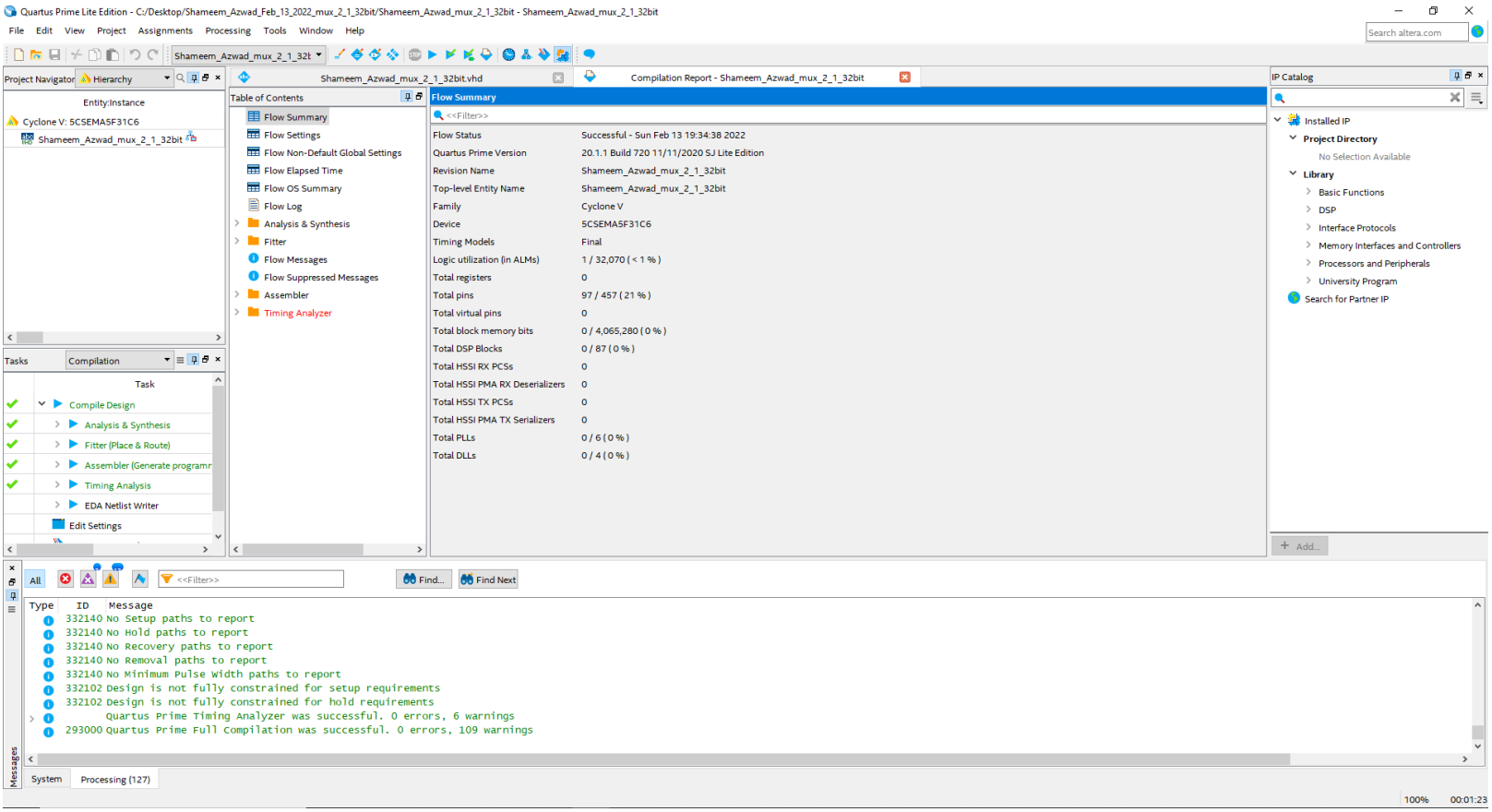

Figure 23: Project compiled successfully

Explanation:

(Part A)

The code and the pictures displaying the project follows the tutorial's instructions and shows the screenshots of the introtutorial.

(Part B)

The images show the creation of the project following the same route shown in introtutorial. The code contains A, B, X inputs as a single bit and Y as a single bit output as stated for this mux 2:1 circuit. In addition, the VHDL code uses the equation for the mux 2:1 circuit which is $Y = (A * \overline{x}) + (B * X)$, which follows in code as → (A AND NOT X) OR (B AND X). This equation makes sure the results are right because it uses the equation which follows the truth table for the right results.

(Part C)

The images show the creation of the project following the same route shown in introtutorial. The code contains A, B inputs as a 32-bit vector with X as a single bit and Y also as a 32-bit vector for this mux 2:1 circuit. In addition, the VHDL code uses the equation for the mux 2:1 circuit which is $Y = (A * \overline{x}) + (B * X)$, which follows in code as → (A AND NOT X) OR (B AND X). This equation makes sure the results are right because it uses the equation which follows the truth table for the right results.

Conclusion:

This lab was very useful to begin learning the usage of Quartus because it allows us to learn the creation and programming of VHDL code for a circuit by using Quartus. In fact, this lab also allowed us to learn the usage of inputs and outputs which are not only single bits but also several its. Lastly, this lab set up a starting understanding of the nuances of how Quartus Lite works and how to use it to code in VHDL.