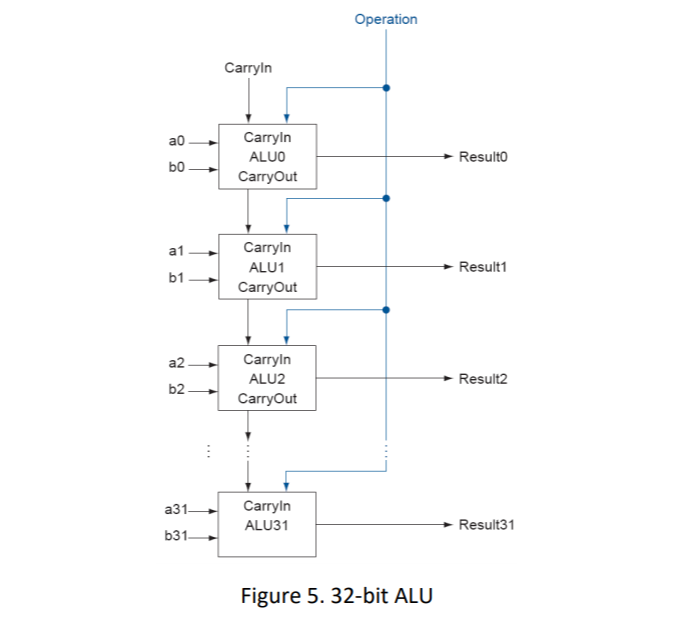
Project I: 32-Bit Adder Design



Acknowledgement: I acknowledge all works including figures, codes and writings belong to

me and/or persons who are referenced. I understand if any similarity in the code, comments, customized program behavior, report writings and/or figures are found, both the helper (original work) and the requestor (duplicated/modified work) will be called for academic disciplinary action.

Signature: 

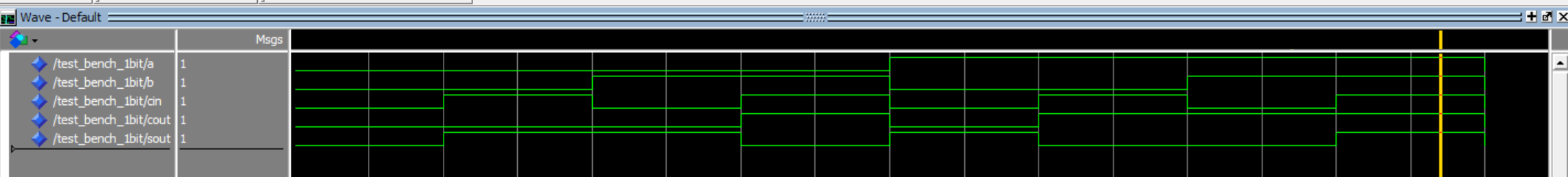
Abstract: The purpose of this project is to provide an introduction to VHDL programming and prepare for the design of a basic 32-bit RISC processor in the future.

Introduction: This goal of this project is to design three different adders using VHDL code: 1-bit full adder, 4-bit full adder, and a 32-bit full adder. The reason for this is to further prepare for project two in which we will design a basic 32-bit RISC processor. Since one of the main building blocks of a processor is an ALU, having a 32-bit adder already designed and tested will be helpful as a starting point in that project.

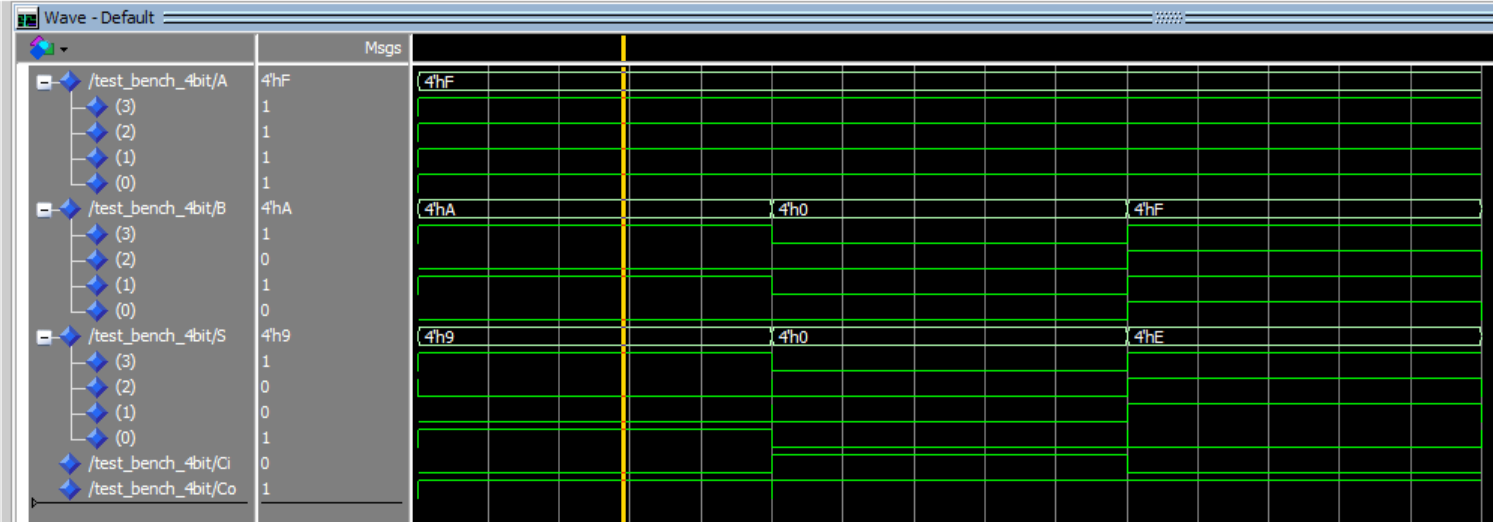
Background: The final 32-bit adder design in this project will be a 32-bit ripple carry adder. While this design has drawbacks in terms of propagation delay of the carry bits throughout the adder, the design is relatively simple and is good as an introduction to VHDL programming. One goal of this project is to learn about the various methods of creating designs in VHDL. This is achieved by first creating a 1-bit full adder using a behavioral model as a starting point for the rest of the design. That 1-bit adder will then be cascaded 4 times to create a 4-bit full adder. In turn, the 4-bit full adder will be cascaded 8 times to create a 32-bit full adder. This method of using previous VHDL components is described as a structural model. Using a structural model provides faster design times and less error prone designs as each component used can be tested individually and verified before use in a larger design. Simulation software often provides tools where these designs can be tested. First a testbench is created in VHDL that provides different signals to measure and components to test. Modelsim was used in this project and its simulation tools provided graphs of input signals and output signals that were used to verify correct functionality of all adder designs.

Results: The 1-bit adder was tested with all possible inputs and the 4/32-bit adders were tested with 3 different input combinations that can be seen in the result graphs.

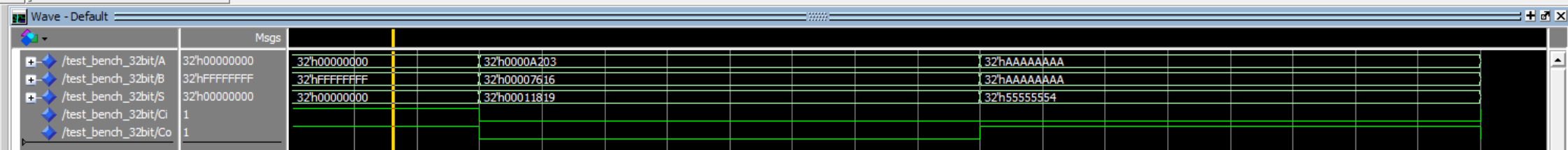
\*Note all graphs are attached in full resolution in the blackboard submission

1-bit Full Adder

This graph shows all possible inputs to a 1 bit adder with carry in. Each section of 20ns shows a change in input and a corresponding change in output.

4-bit Full Adder

This graph shows three different input combinations and their corresponding outputs for a 4-bit adder made by cascading four of the previous 1-bit adders.

32-bit Full Adder

This graph shows three different 32 bit input combinations in hexadecimal and their corresponding outputs for a 32-bit adder made by cascading 8 of the previous 4-bit adders. The second combination of 7616 and A203 was the input required by the project description of the first and last 4 digits of my student A number.

Challenges: Some of the challenges I encountered in this project was how to use modelsim and how to design components in VHDL. This was the first project I have done in VHDL so the syntax and design methodology was new to me. As such, it took time to read tutorials and sample code in order to create the designs.

Conclusion: This project is a good starting point in the creation of an ALU in the future since a 32-bit adder is an essential component of a 32-bit ALU. This project also demonstrated how important structural design is and how it can make the design of large projects much simpler. For example, the VHDL code for the 32-bit adder was similar in length as the 4-bit adder even though there would be much more complexity if done with a behavioral model. It also took much less time to create than using a behavioural model. Testing was also much easier, and locating mistakes was easier by using this method. By testing the 1-bit adder as soon as it was finished, it was possible to minimize errors in creating the 4-bit adder and the 32-bit adder.

List of References:

1. VHDL Tutorial by Peter J. Ashenden
   1. This tutorial was used when trying to figure out the syntax and design techniques for representing designs in VHDL. It was also helpful when trying to create loops in VHDL.
2. ECE 485 Lecture 2 Slides by Dr. Won-Jae Yi
   1. The slides were referenced in the creation of the 1-bit full adder and the 4-bit full adder. My final designs for the behavior model of the 1-bit adder and the structural model for the 4-bit full adder were heavily based on the designs presented in the slides and in class.
3. Tutorial - Using Modelsim for Simulation, for Beginners by nandland.com
   1. This tutorial was used while trying to understand how to use the simulation tools of modelsim and how to capture the changes in output and inputs while varying time.
   2. https://www.nandland.com/vhdl/tutorials/tutorial-modelsim-simulation-walkthrough.html

Appendix:

Source code attached as vhd files.