Lab 0

Brendan Bovenschen A20323473, Landon Fox A20319340

Section 1: Introduction

The purpose of this lab is to gain an understanding of HDL, SystemVerilog, and Modelsim by creating a full adder and testing it on the board. This lab is laying the groundwork for the labs to come by introducing us to the software and hardware that will be used for all future labs. A full adder performs Boolean addition and provides outputs for each combination of inputs. This would generally be used in a digital counter as an operator of a single bit. Its function is to add two bits together and send this information to its output (usually another full adder). We completed the baseline design by implementing a full adder module and specifying which combinations to test in the test bench. We tried a few different variations of the test bench to make sure the paths were displayed properly by dividing up the inputs and outputs on the waveform. Since the lab instructions provided the necessary boolean algebra, we only used a single design which we ran through our testbench to ensure it functioned as intended.

Section 2: Baseline Design

Given an equation of the Boolean logic for a full adder, we were tasked with translating this into a Hardware Descriptive Language (specifically SystemVerilog). Although we didn't create a physical build of this design, this logic could be simulated and executed in SystemVerilog. To ensure that the design we created in SystemVerilog was functioning properly, we used a waveform graph that mapped each connection within the full adder to a respective digital wave. As seen in Figure 1, the design worked as intended with all possible inputs.

Section 3: Detailed Design

To implement the Boolean logic provided in the lab documentation, we created a module (which we named full adder) with the inputs of A, B, C-In (carry in) and outputs S (sum) and C-Out (carry out). Each output consists of separate logic functions (Figure 2). To create a full adder, we add a third input (C-In) to a half adder (with inputs of A and B and outputs of S and C-Out). S outputs a high signal if only one of its inputs is high, or all three are high and C-Out will output a high signal when two or more inputs are high. These truths can be seen in the truth table described in Figure 3. Within our SystemVerilog module, we assigned our outputs with this logic. We also created the waveforms within our testbench to display each input and output which was used to prove our design's functionality.

Section 4: Testing Strategy

For our testing, we observed a waveform graph that outputs the same as our programmed module that matches a full adder. We tested this by writing the full adder and giving the testbench each combination for the three inputs and simulating in Modelsim. Although we did not develop any other designs, we could have possibly tested a build with unsimplified logic. However, this would take much more time to run as it would include many more gates than the final, fully simplified version we created. We also tested on the board in the lab running through each set of combinations but this time using physical switches on the board that corresponded to the same combinations simulated in ModelSim. Each of the physical tests matched the corresponding test in the simulation/waveform graph.

Section 5: Evaluation

From this lab, we were able to further our sense of the language of SystemVerilog and the process of translating boolean algebra to a format that a computer can compile. The processes and practices used in our work will be implemented in all lab assignments. Although this is a small integrated circuit that would be a small component of a complex design, this basic component is a necessary starting point for all further projects in this course.

Figure 1

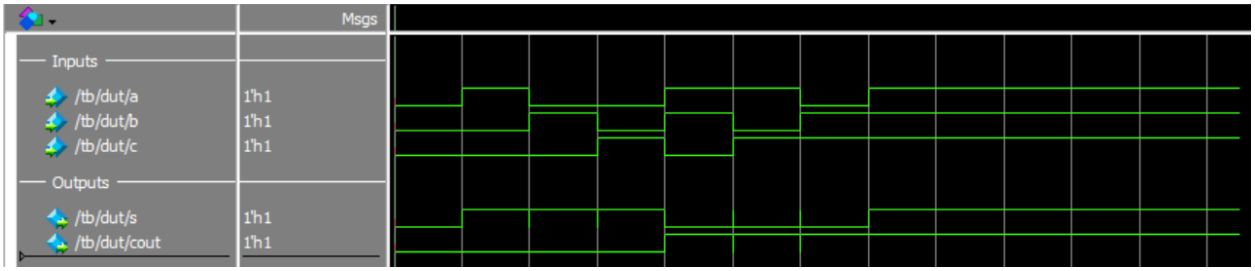

Figure 2

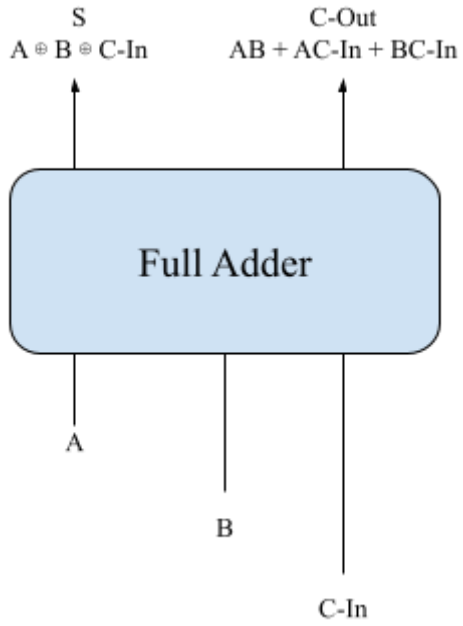

Figure 3

A	B	C-In		S	C-Out
0	0	0		0	0
0	0	1		1	0
0	1	0		1	0
0	1	1		0	1
1	0	0		1	0
1	0	1		0	1
1	1	0		0	1
1	1	1		1	1