Part 2 – Executive Summary

Throughout this lab, we were able to get acquainted with the circuit design software Quartus, and the various simulation tools that accompany the intel Quartus program. We essentially performed the equivalent of 'Hello, World' in most programming languages; but since we are specifically dealing with VHDL, a hardware description language, this was performed by implementing a simple OR gate. Furthermore, this lab allowed for the familiarization of the Quartus configurations that are necessary for testing the various logic circuits. It also serves as an introduction to testbench files, and their structure. Finally, we were able to run a simulation of the given OR gate design using the ModelSim tool.

Part 3 & 4 – Implementation and Results

As per the instructions provided, there was no direct implementation that occurred for this lab. The implementation of the OR gate was supplied as part of the instructions. The implementation used (Figure 1), can be considered a structural or behavioral implementation. This is because while it uses the begin process statement to implement the functionality, the functionality is described using an OR gate itself, meaning that essentially it is a structural implementation within a behavioral structure. We can observe that the product of the OR operation of the signals a and b is recorded within the output signal q.

```
 5  entity or_gate is
 6  port(
 7     a: in std_logic;
 8     b: in std_logic;
 9     q: out std_logic);
10  end or_gate;
11
12  architecture rtl of or_gate is
13  begin
14     process(a, b) is
15     begin
16        q <= a or b;
17     end process;
18  end rtl;
19
```

Figure 1 Implementation of OR Gate

Furthermore, we were instructed to implement the testbench for the given design. The testbench script was also provided. Essentially, the testbench script supplied the different possible inputs to the design and using the assert() method, checked to see if the output is that which we expected. The testbench script also considers X, an unknown signal in order to exhibit the fact that the OR gate does not really take into account one of the inputs as long as one input is '1', the output will always be 1. This mirrors our understanding of an OR gate as represented within the truth table in figure 2.2.

```
26    process
27    begin
28      a_in <= '0';
29      b_in <= '0';
30      wait for 1 ns;
31      assert(q_out='0') report "Fail 0/0" severity error;
32
33      a_in <= '0';
34      b_in <= '1';
35      wait for 1 ns;
36      assert(q_out='1') report "Fail 0/1" severity error;
37
38      a_in <= '1';
39      b_in <= 'X';
40      wait for 1 ns;
41      assert(q_out='1') report "Fail 1/X" severity error;
42
43      a_in <= '1';
44      b_in <= '1';
45      wait for 1 ns;
46      assert(q_out='1') report "Fail 1/1" severity error;
47
48      -- Clear inputs
49      a_in <= '0';
50      b_in <= '0';
51
52      assert false report "Test done." severity note;
53      wait;
54    end process;
55  end tb;
56
```

Figure 2.1 Testbench Script for OR Gate

Truth Table		
B	A	Q
0	0	0
0	1	1
1	0	1
1	1	1

Figure 2.2 Truth Table for OR Gate

By running the testbench script by way of Quartus, we were able to generate a simulation plot of the OR gate, shown in figure 3.1. This simulation plot is essentially handled by the ModelSim program but initiated by Quartus. By observing the simulation plot we can find that the results match our logic and truth table of an OR gate. Where, when the inputs are '1' and 'X', we receive an output of 1. Moreover, we were able to generate a schematic of the design using Quartus' Netlist viewer. This can be observed in figure 3.2 and is simply a schematic of the OR gate we both described and used in our implementation of or_gate.

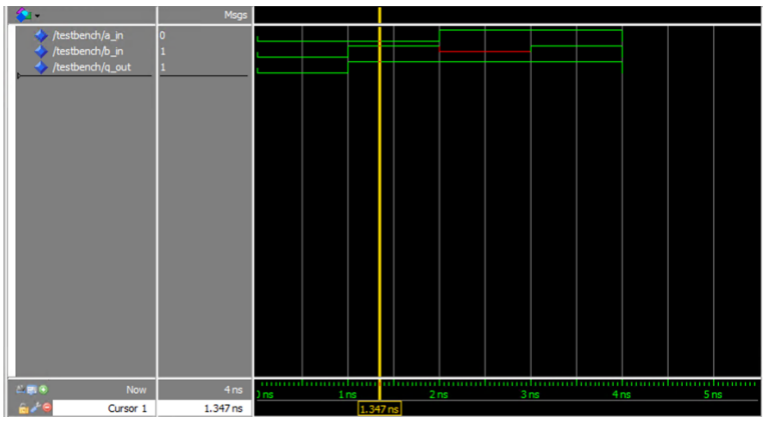

Figure 3.1 Simulation Plot for OR Gate

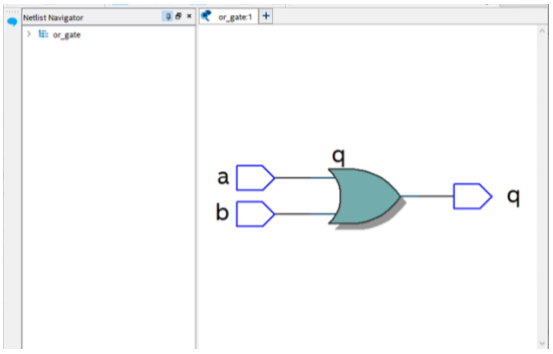

Figure 3.2 Schematic of or_gate

Part 5 – Conclusions

Throughout this lab/demonstration, we were able to familiarize ourselves with the tools and processes within the Quartus program. We have been able to incorporate a design and use the external tool of ModelSim to run our simulation of the design at hand. The use of the X input also gave us a more concrete understanding of how to use this type of signal within our designs/ simulation. We observed the behavior of an OR gate throughout this investigation, through a structural and behavioral lens. This serves as a building block to further investigations and labs.