Lab 1

261226740 Ming Li Liu
261139430 Georges Honein
261135566 Guy Khairallah

In this assignment, we have learned how to use the Quartus Prime 18.1 and familiarize ourselves with VHDL from the examples. We followed the instructions to create a project and the files. Using the example or_gate and testbench, we used ModelSim to visualize the outputs. We then inspired ourselves from the example to write the xnor gate and did the same steps to see the outputs.

Unfortunately, the snip tool of the computer did not work, so we could not take screenshots and had to take pictures of the screen instead.

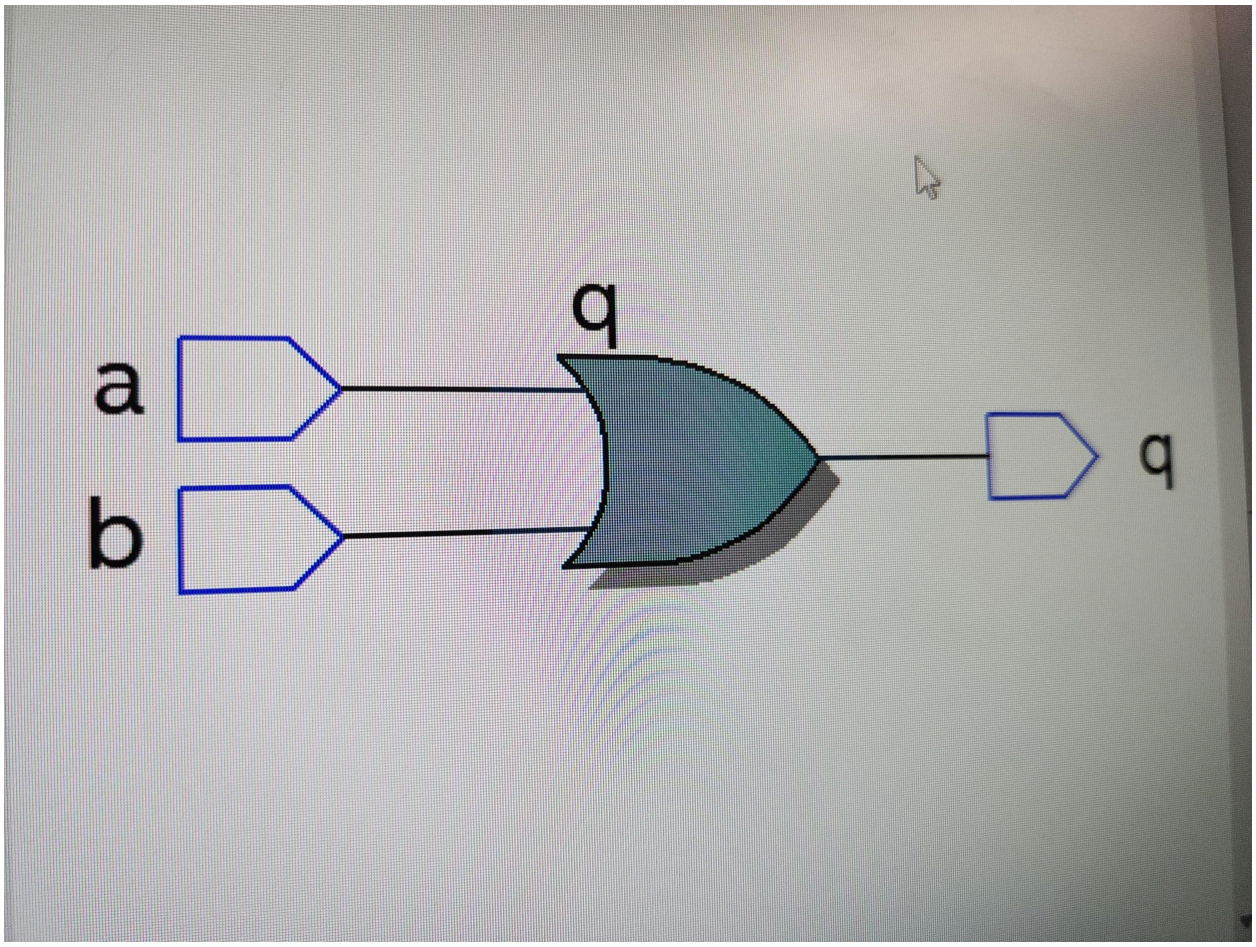

Fig 1: Or gate, Schematic representation of the or vhdl code

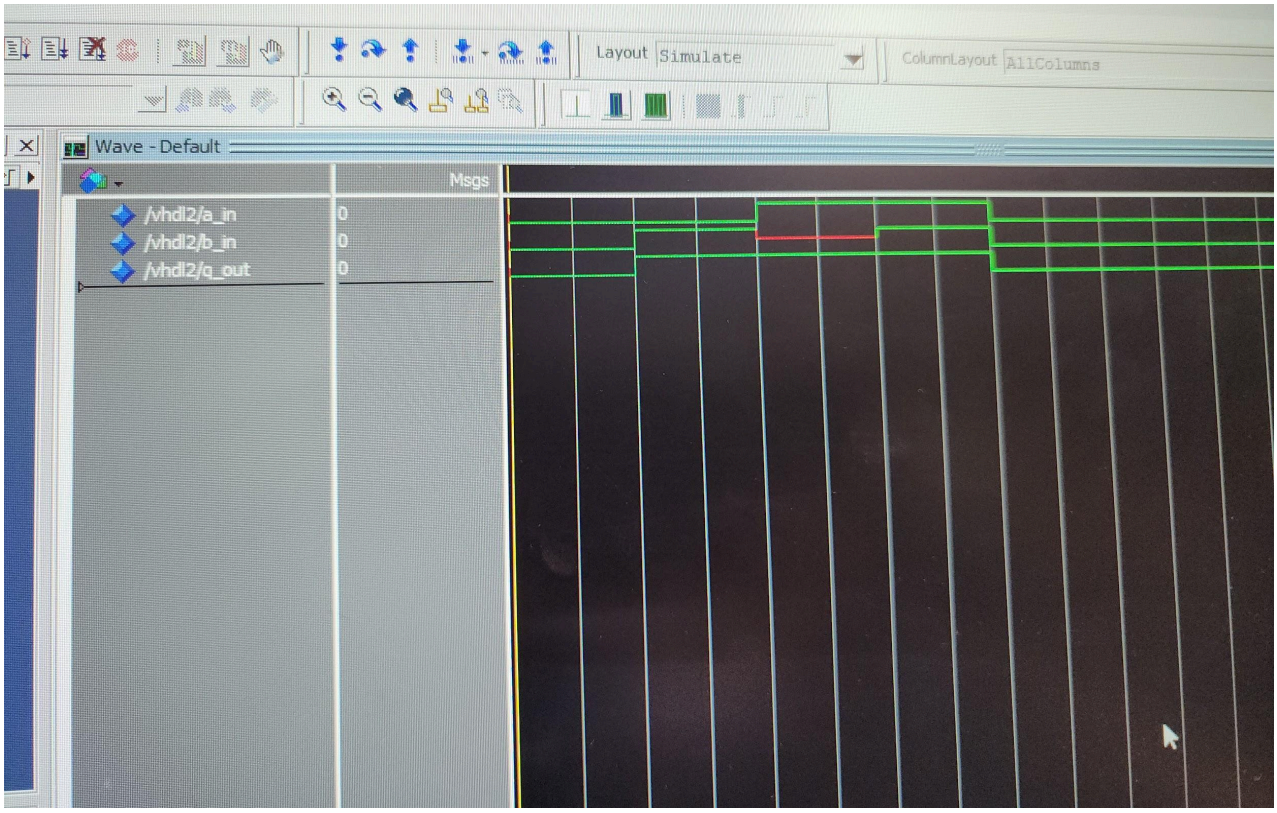

Fig 2: Or gate simulation plot over time. The red line is when b_in is unknown, but since a_in is 1, no matter the value of b_in the q_output is 1. Every vertical line are 0.5 ns apart.

```
or_gate.vhd
1   -- Simple OR gate design
 1 library IEEE;
 2 use IEEE.std_logic_1164.all;
 3
 4 entity or_gate is
 5 port(
 6   a: in std_logic;
 7   b: in std_logic;
 8   q: out std_logic);
 9 end or_gate;
10
11 architecture rtl of or_gate is
12 begin
13   process(a, b) is
14   begin
15     q <= a or b;
16   end process;
17 end rtl;
```

Fig 3: Or gate vhdl code.

test_or.vhd

```
1   -- Testbench for OR gate
1 library IEEE;
2 use IEEE.std_logic_1164.all;
3
4 entity test_or is
5 -- empty
6 end test_or;
7
8 architecture tb of test_or is
9
10 -- DUT component
11 component or_gate is
12 port(
13   a: in std_logic;
14   b: in std_logic;
15   q: out std_logic);
16 end component;
17
18 signal a_in, b_in, q_out: std_logic;
19
20 begin
21
22   -- Connect DUT
23   DUT: or_gate port map(a_in, b_in, q_out);
24
25   process
26   begin
27     a_in <= '0';
28     b_in <= '0';
29     wait for 1 ns;
30     assert(q_out='0') report "Fail 0/0" severity error;
31
32     a_in <= '0';
33     b_in <= '1';
34     wait for 1 ns;
35     assert(q_out='1') report "Fail 0/1" severity error;
36
37     a_in <= '1';
38     b_in <= 'X';
39     wait for 1 ns;
40     assert(q_out='1') report "Fail 1/X" severity error;
41
42     a_in <= '1';
43     b_in <= '1';
44     wait for 1 ns;
45     assert(q_out='1') report "Fail 1/1" severity error;
46
47     -- Clear inputs
48     a_in <= '0';
49     b_in <= '0';
50
51     assert false report "Test done." severity note;
52     wait;
53   end process;
54 end tb;
```

Fig 4: Testbench vhdl code for the or gate.

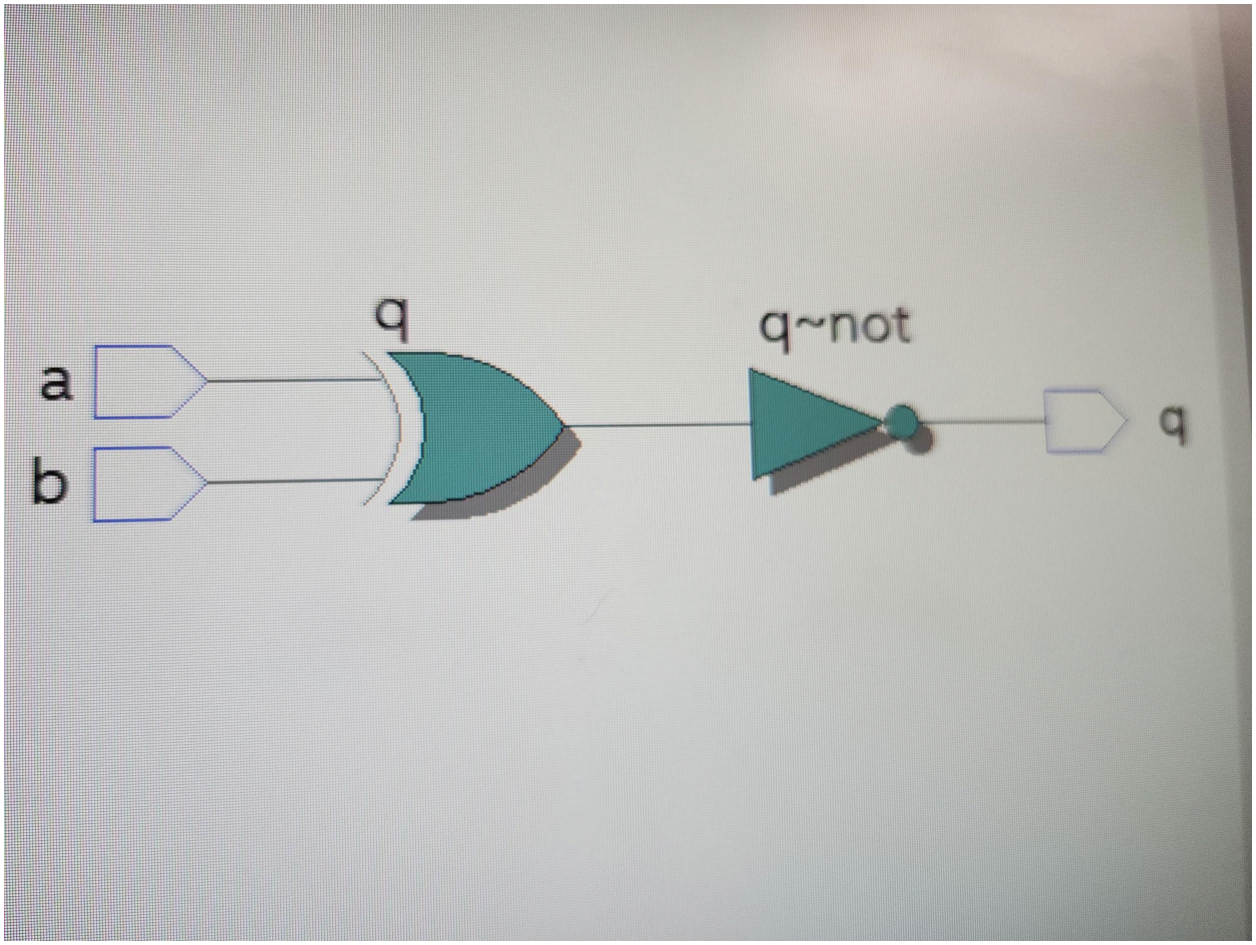

Fig 5: Xnor gate, Schematic representation of the xnor vhdl code

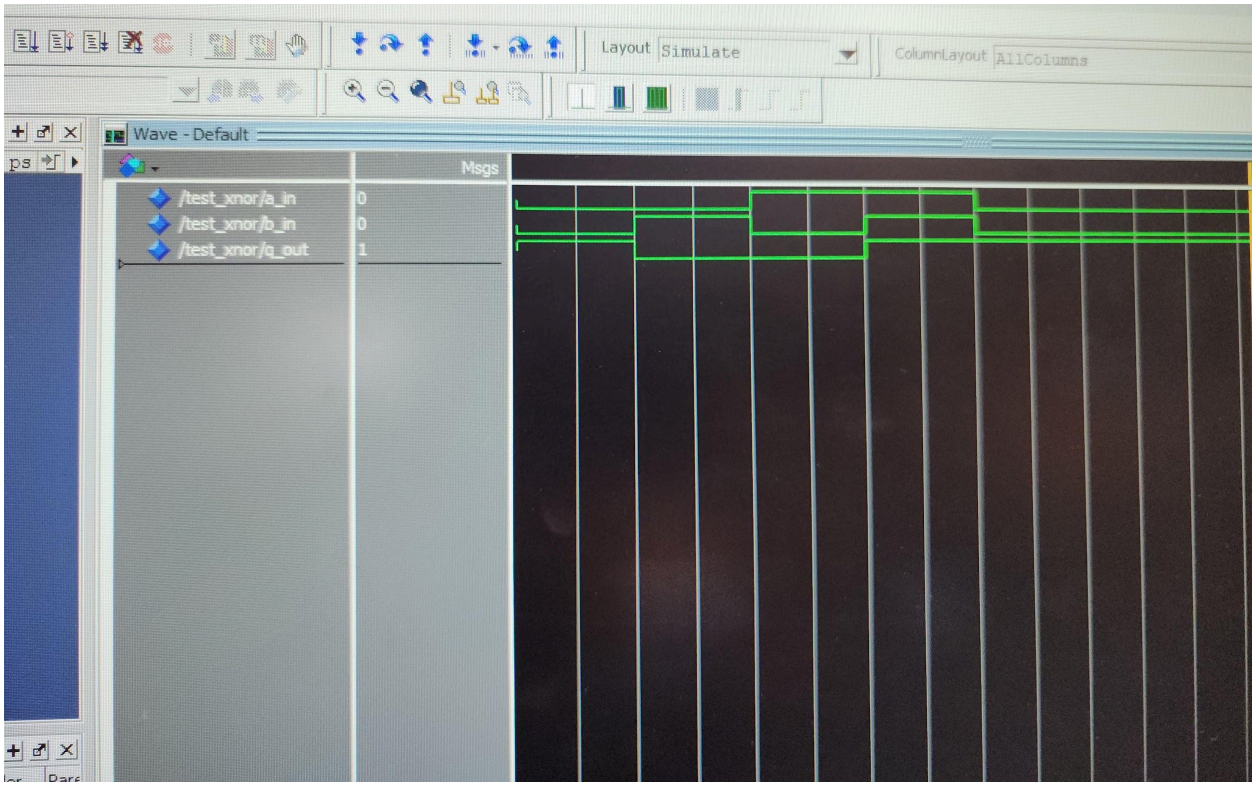

Fig 6: Xnor gate simulation plot over time. Every vertical line are 0.5 ns apart.

xnor_gate.vhd

```
1   -- Simple XNOR gate design
  1 library IEEE;
  2 use IEEE.std_logic_1164.all;
  3
  4 entity xnor_gate is
  5 port(
  6   a: in std_logic;
  7   b: in std_logic;
  8   q: out std_logic);
  9 end xnor_gate;
 10
 11 architecture rtl of xnor_gate is
 12 begin
 13   process(a, b) is
 14   begin
 15     q <= a xnor b;
 16   end process;
 17 end rtl;
```

Fig 7: xnor gate vhdl code.

test_xnor.vhd

```
1   -- Testbench for XNOR gate
 1 library IEEE;
 2 use IEEE.std_logic_1164.all;
 3
 4 entity test_xnor is
 5 -- empty
 6 end test_xnor;
 7
 8 architecture tb of test_xnor is
 9
10 -- DUT component
11 component xnor_gate is
12 port(
13   a: in std_logic;
14   b: in std_logic;
15   q: out std_logic);
16 end component;
17
18 signal a_in, b_in, q_out: std_logic;
19
20 begin
21
22   -- Connect DUT
23   DUT: xnor_gate port map(a_in, b_in, q_out);
24
25   process
26   begin
27     a_in <= '0';
28     b_in <= '0';
29     wait for 1 ns;
30     assert(q_out='1') report "Fail 0/0" severity error;
31
32     a_in <= '0';
33     b_in <= '1';
34     wait for 1 ns;
35     assert(q_out='0') report "Fail 0/1" severity error;
36
37     a_in <= '1';
38     b_in <= '0';
39     wait for 1 ns;
40     assert(q_out='0') report "Fail 1/0" severity error;
41
42     a_in <= '1';
43     b_in <= '1';
44     wait for 1 ns;
45     assert(q_out='1') report "Fail 1/1" severity error;
46
47     -- Clear inputs
48     a_in <= '0';
49     b_in <= '0';
50
51     assert false report "Test done." severity note;
52     wait;
53   end process;
54 end tb;
```

Fig 8: Testbench vhdl code for xnor gate.

In conclusion, the simulation of the or and xnor gates using the Quartus Prime Lite simulator showed that the vhdl code written accurately represented their behavior. This lab served as an introduction to vhdl simulation and the Quartus Prime Lite tool.