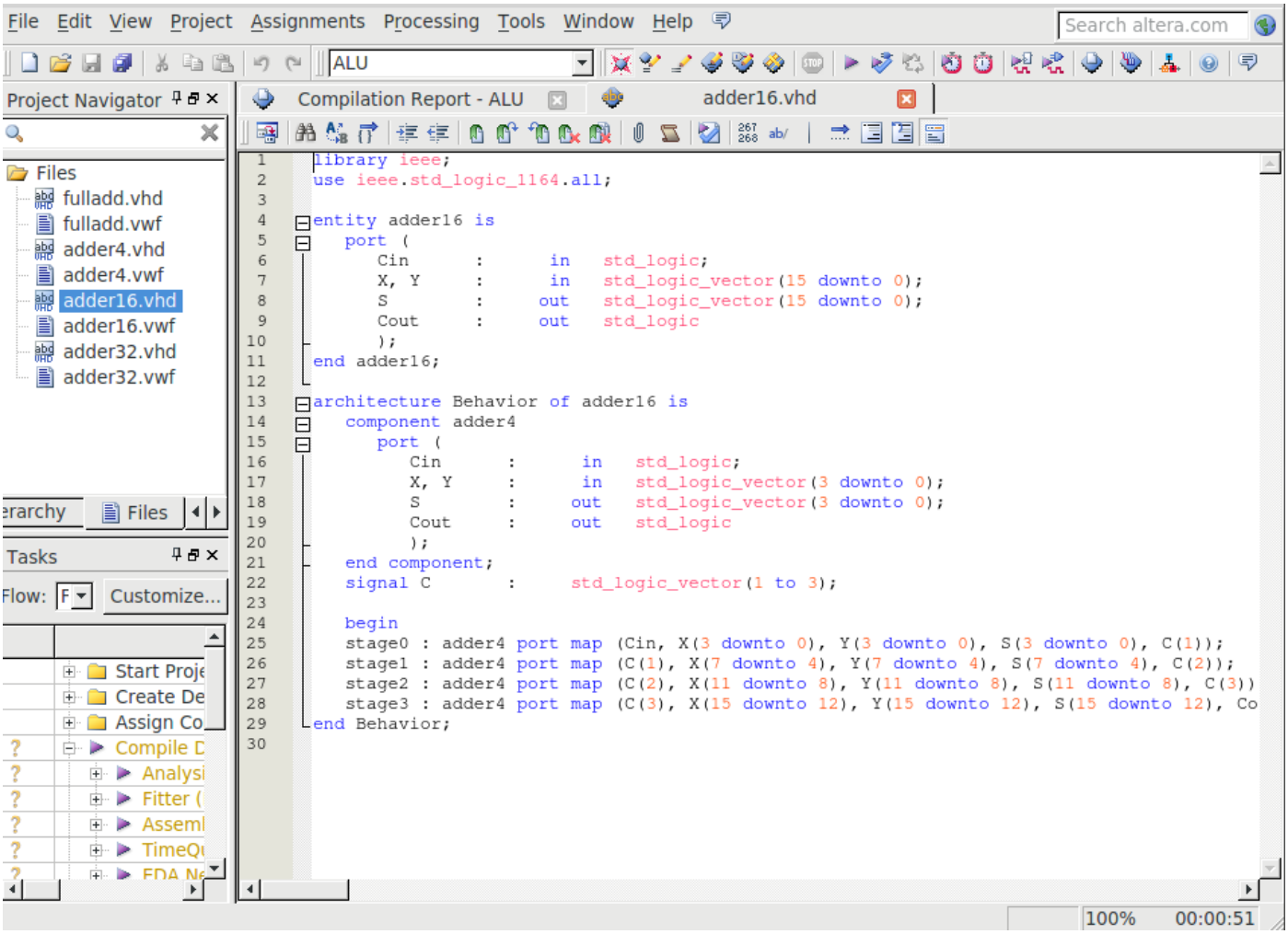
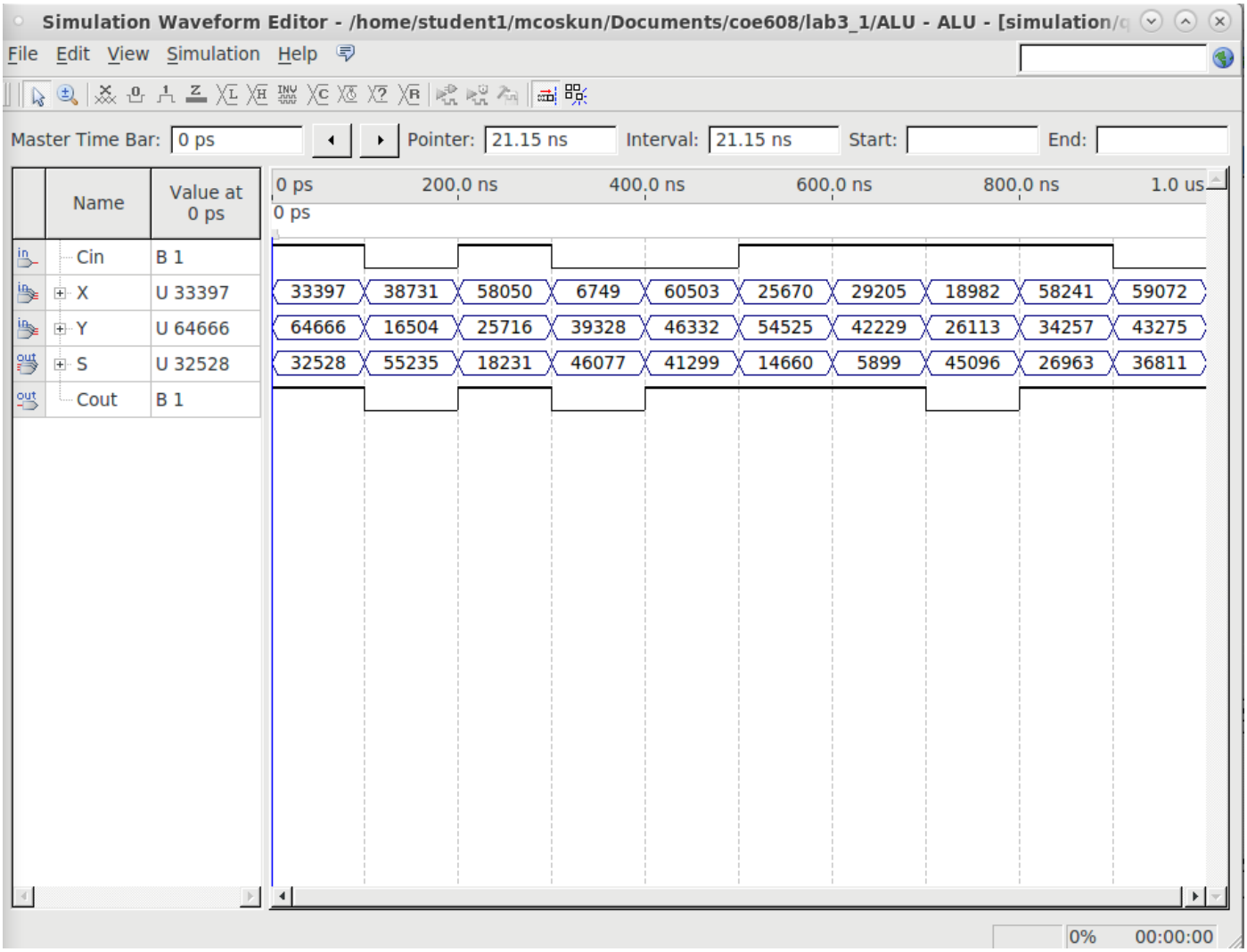
Below, you will see the implementation of 4-bit adder, 16-bit adder, 32-bit adder, fulladd, and ALU to create a 32-bit Arithmetic Logic Unit (ALU) capable of performing six operations using VHDL. The 32-bit ALU consists of two 32-bit inputs (a, b), 32-bit output (Result), 3-bit control signals (op), and two status flags (zero and Carry).

The VHDL code of the 16-bit adder can be seen below.

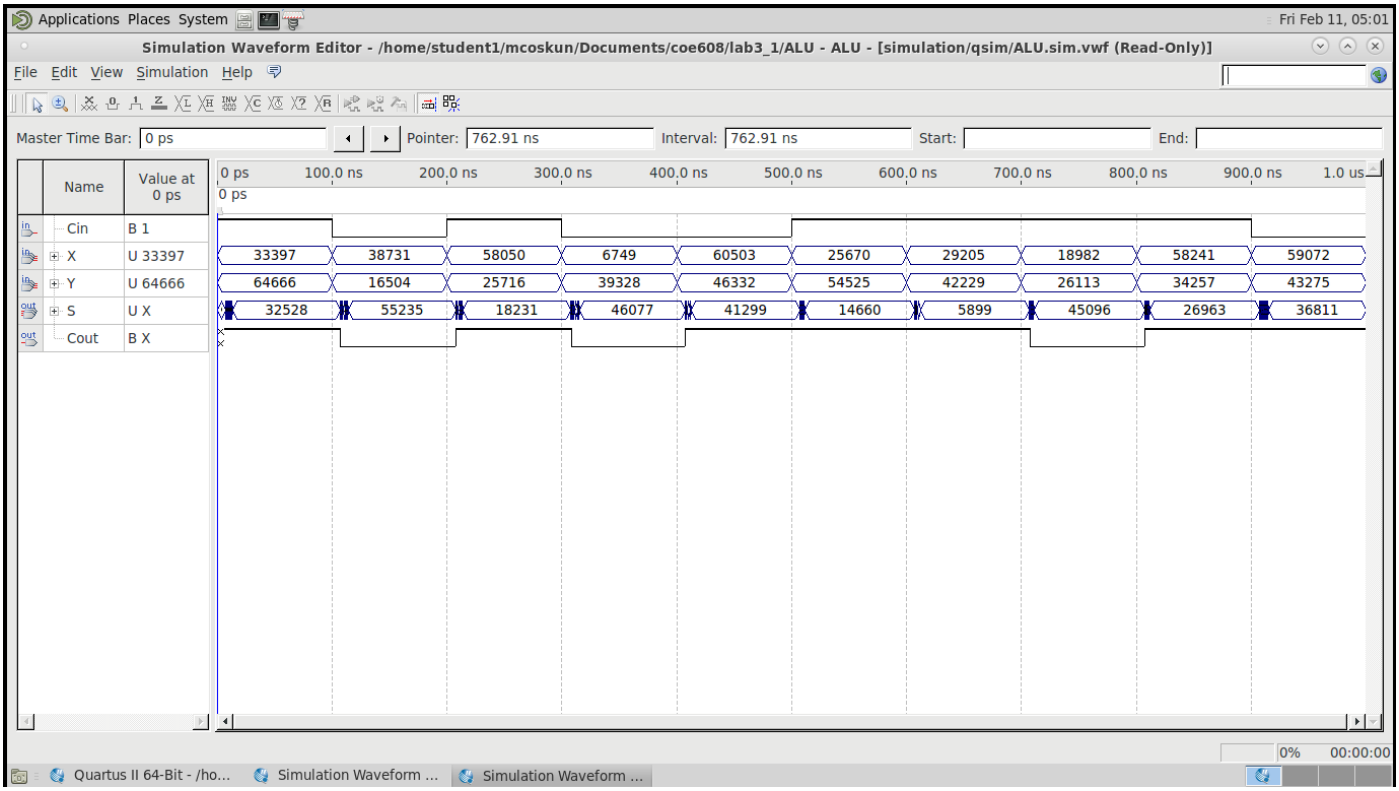



```
1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  entity adder16 is
5     port (
6        Cin      :      in   std_logic;
7        X, Y     :      in   std_logic_vector(15 downto 0);
8        S        :     out   std_logic_vector(15 downto 0);
9        Cout     :     out   std_logic
10       );
11 end adder16;
12
13 architecture Behavior of adder16 is
14    component adder4
15       port (
16          Cin      :      in   std_logic;
17          X, Y     :      in   std_logic_vector(3 downto 0);
18          S        :     out   std_logic_vector(3 downto 0);
19          Cout     :     out   std_logic
20          );
21    end component;
22    signal C       :     std_logic_vector(1 to 3);
23
24    begin
25    stage0 : adder4 port map (Cin, X(3 downto 0), Y(3 downto 0), S(3 downto 0), C(1));
26    stage1 : adder4 port map (C(1), X(7 downto 4), Y(7 downto 4), S(7 downto 4), C(2));
27    stage2 : adder4 port map (C(2), X(11 downto 8), Y(11 downto 8), S(11 downto 8), C(3))
28    stage3 : adder4 port map (C(3), X(15 downto 12), Y(15 downto 12), S(15 downto 12), Co
29 end Behavior;
30
```

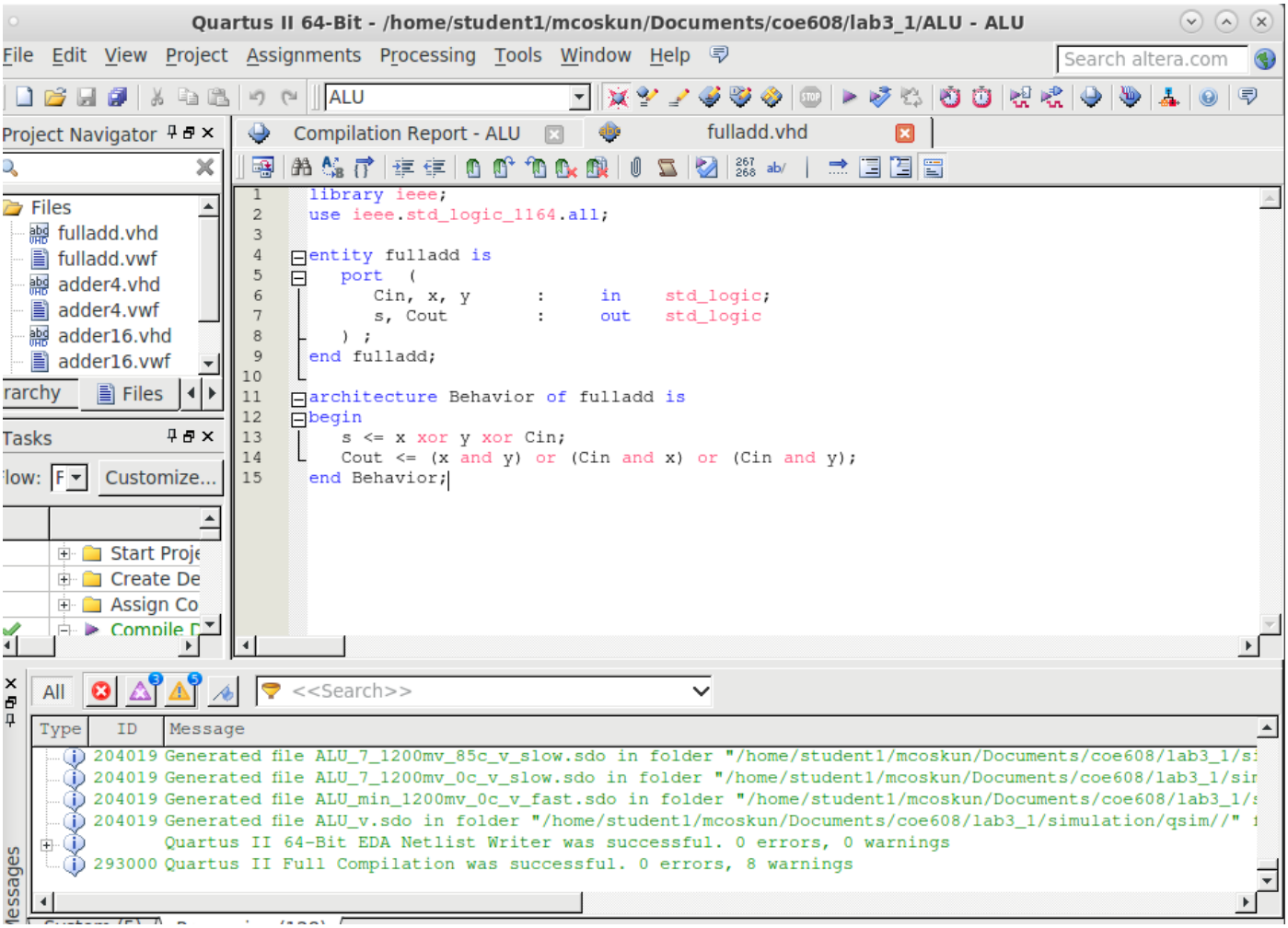
After compiling successfully, the waveform of 16-bit adder created following the lab manual is displayed below (Function waveform).



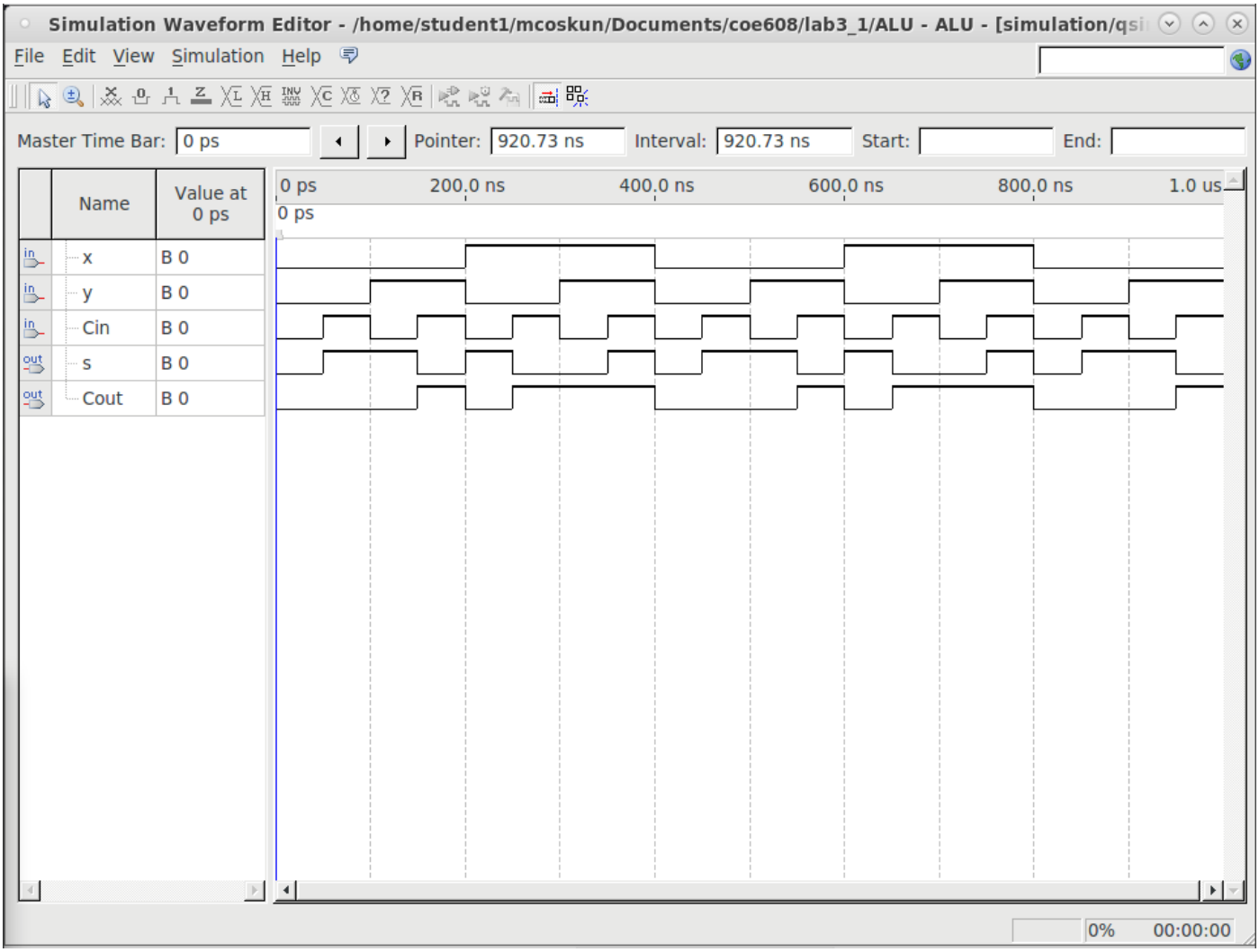
After compiling successfully, the waveform of 16-bit adder created following the lab manual is displayed below (Timing waveform).



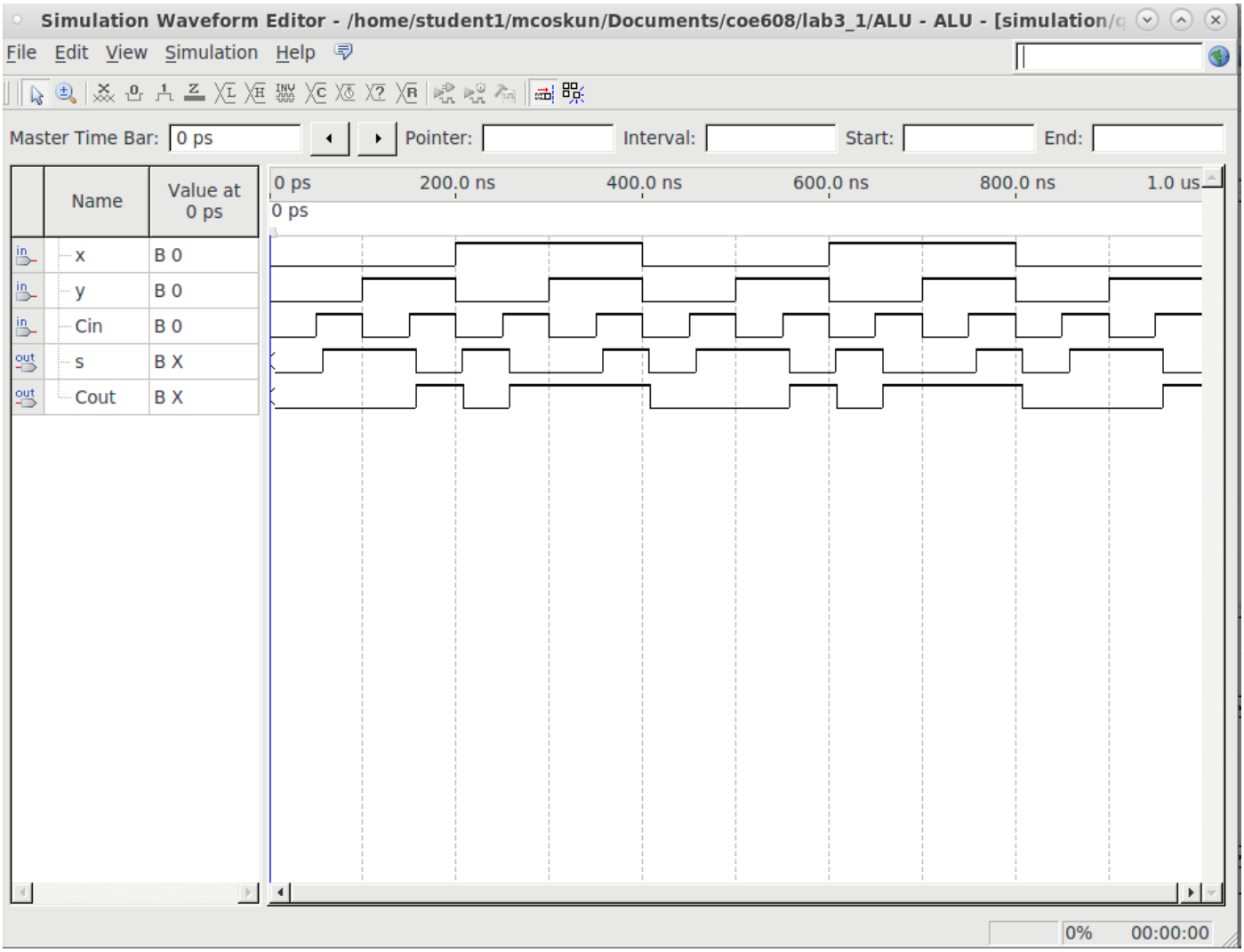
The VHDL code of the fulladd can be seen below.



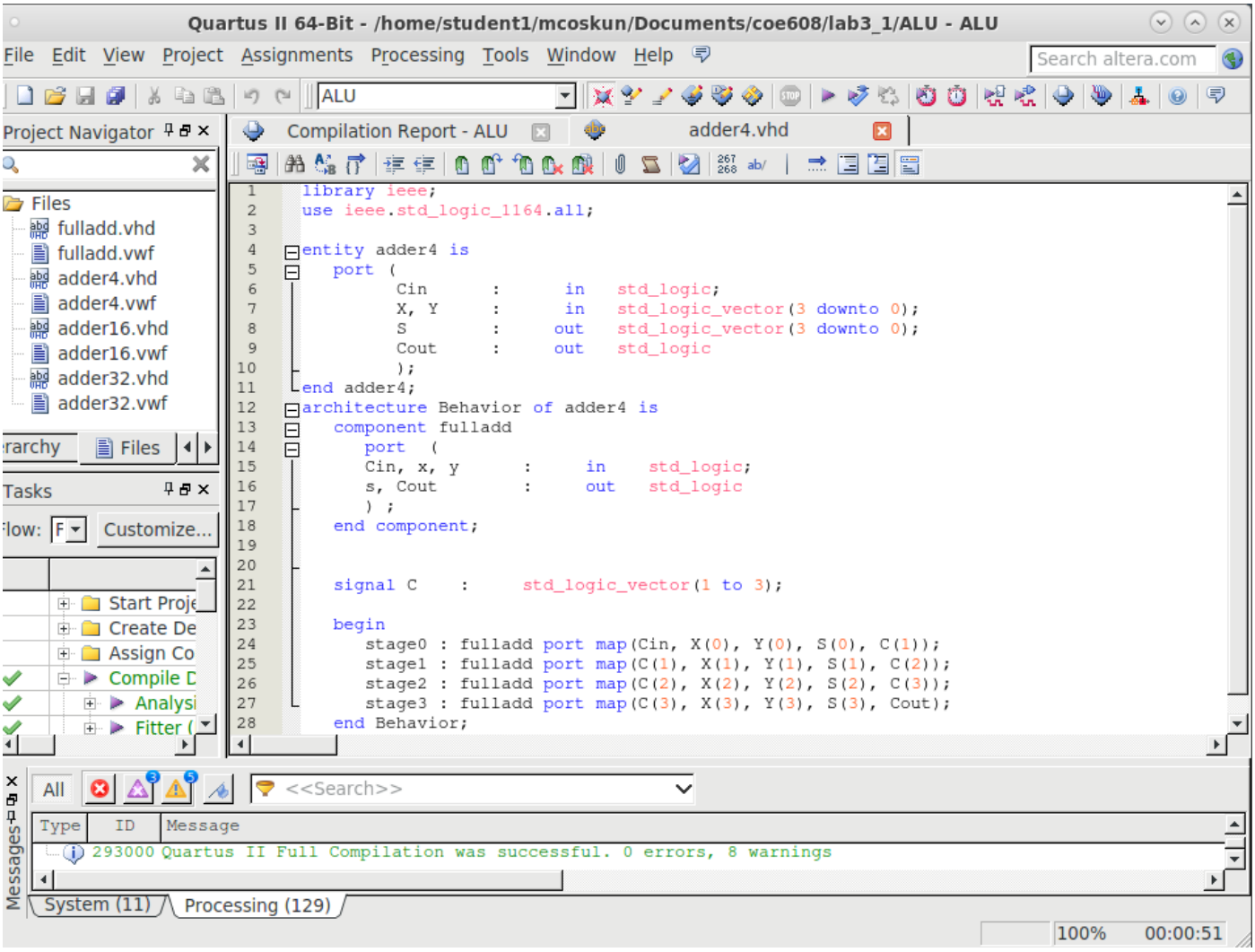
After compiling successfully, the waveform of fulladd created following the lab manual is displayed below (Function waveform).



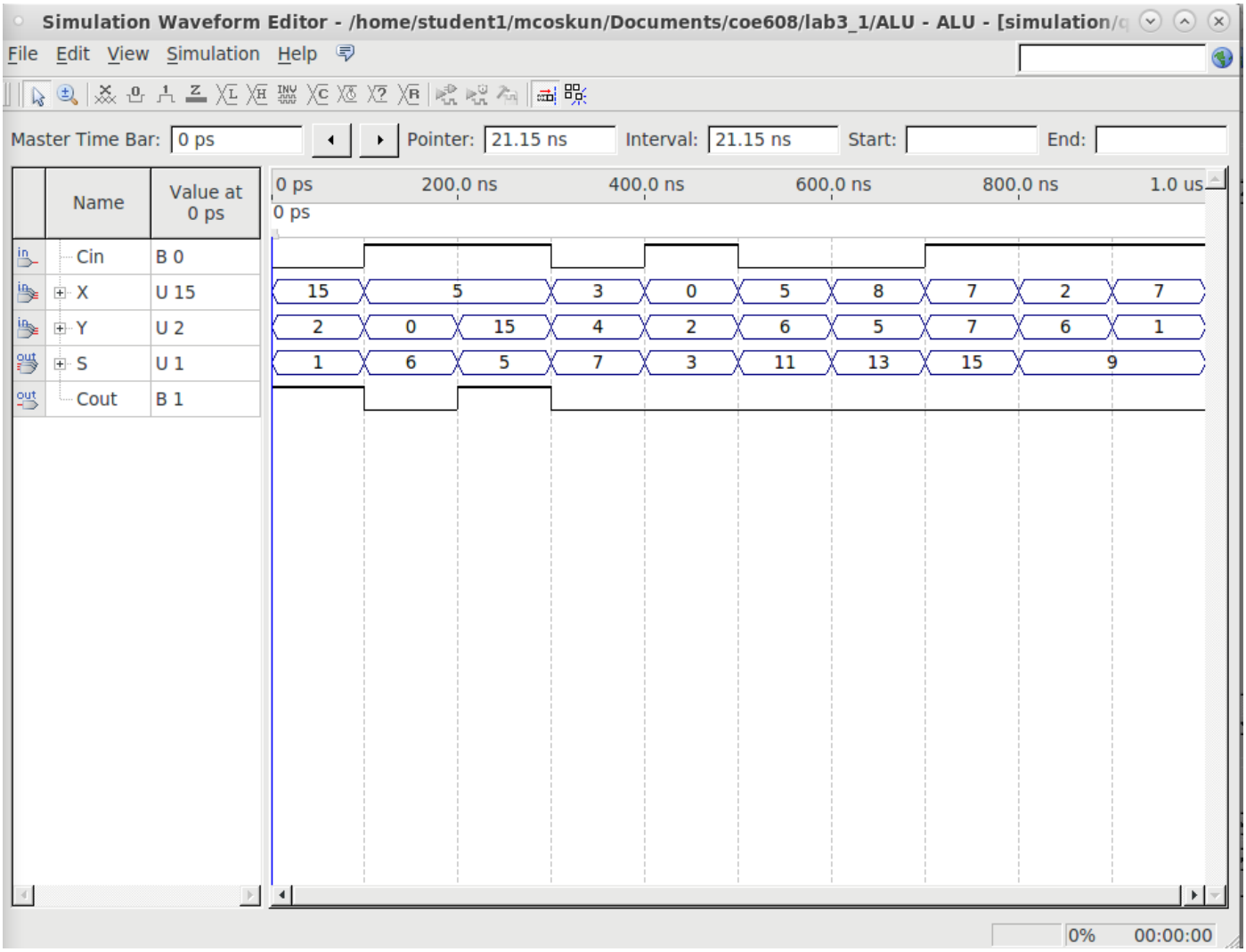
After compiling successfully, the waveform of fulladd created following the lab manual is displayed below (Timing waveform).



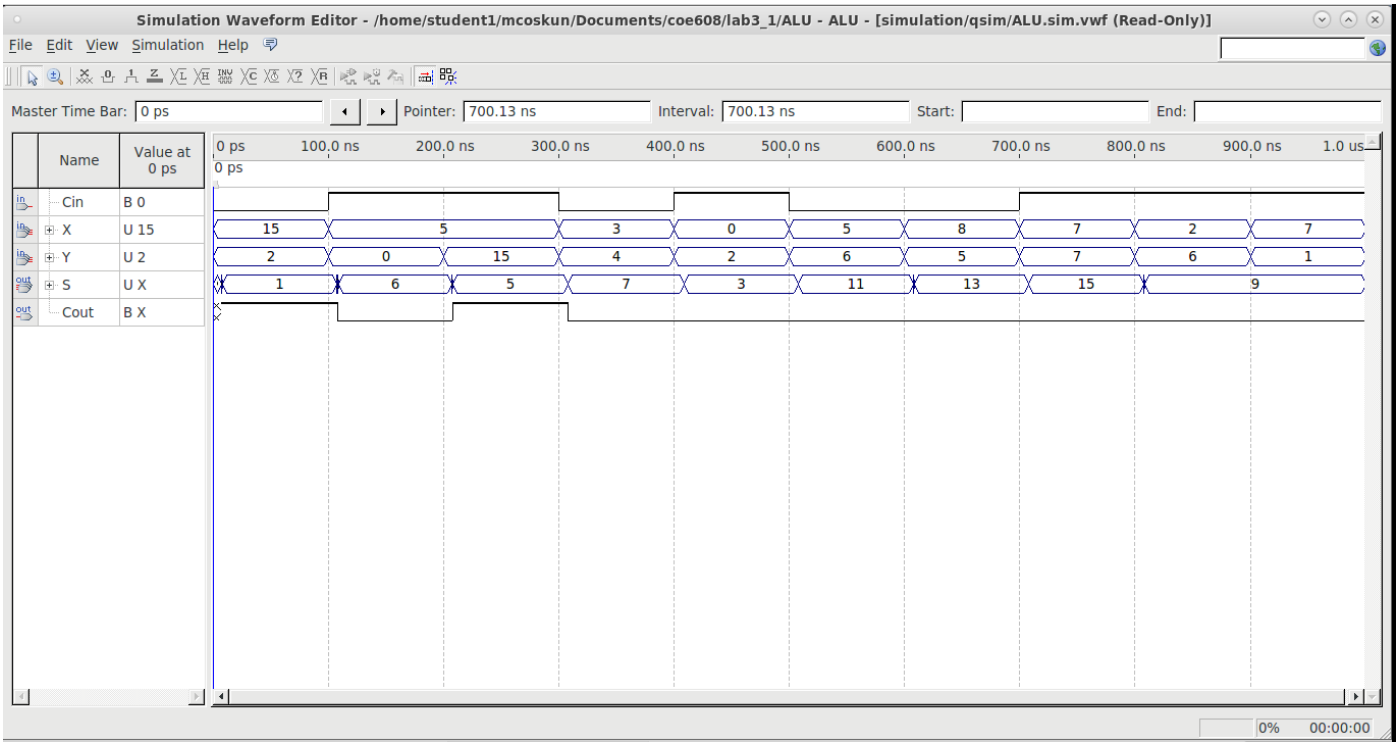
The VHDL code of the 4-bit adder can be seen below.



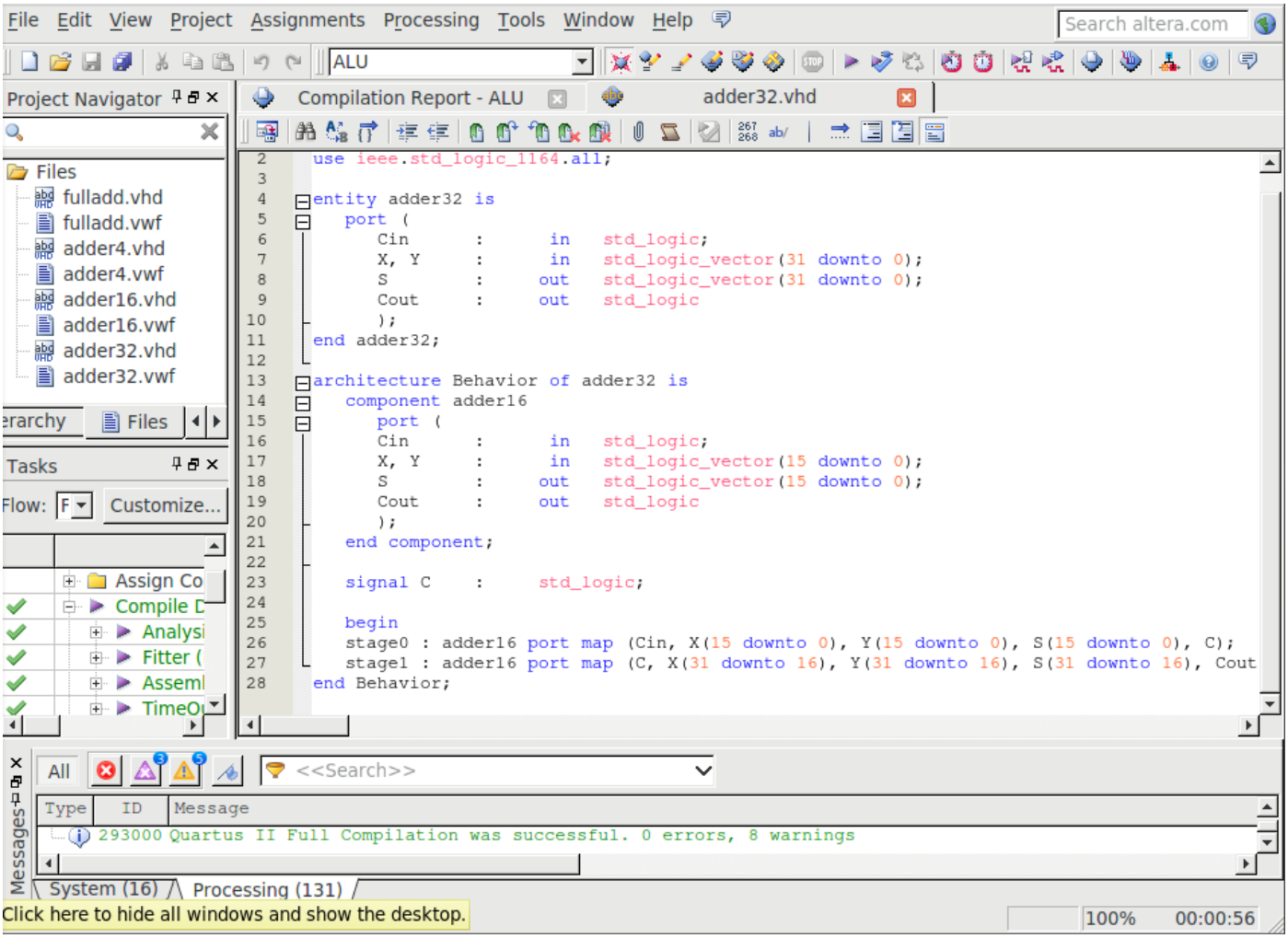
After compiling successfully, the waveform of 4-bit adder created following the lab manual is displayed below (Function waveform).



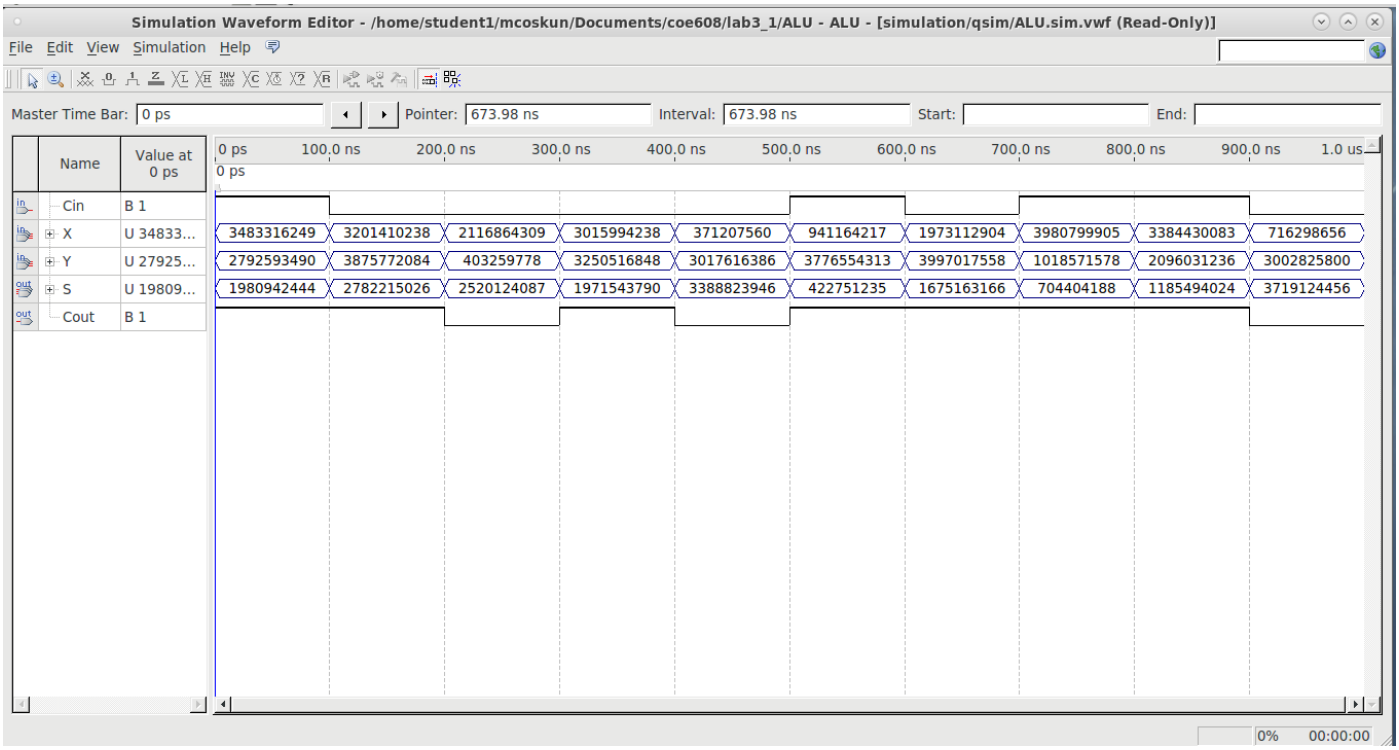
After compiling successfully, the waveform of 4-bit adder created following the lab manual is displayed below (Timing waveform).



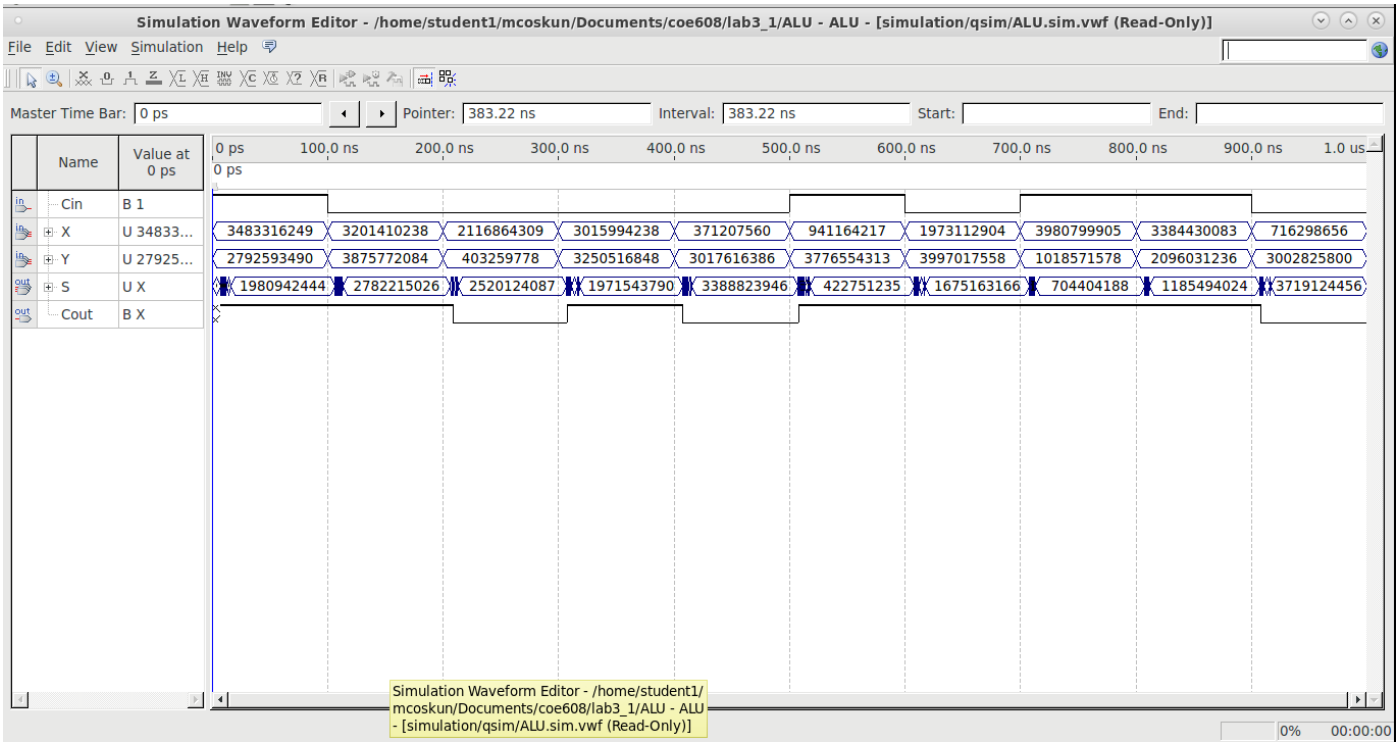
The VHDL code of the 32-bit adder can be seen below.



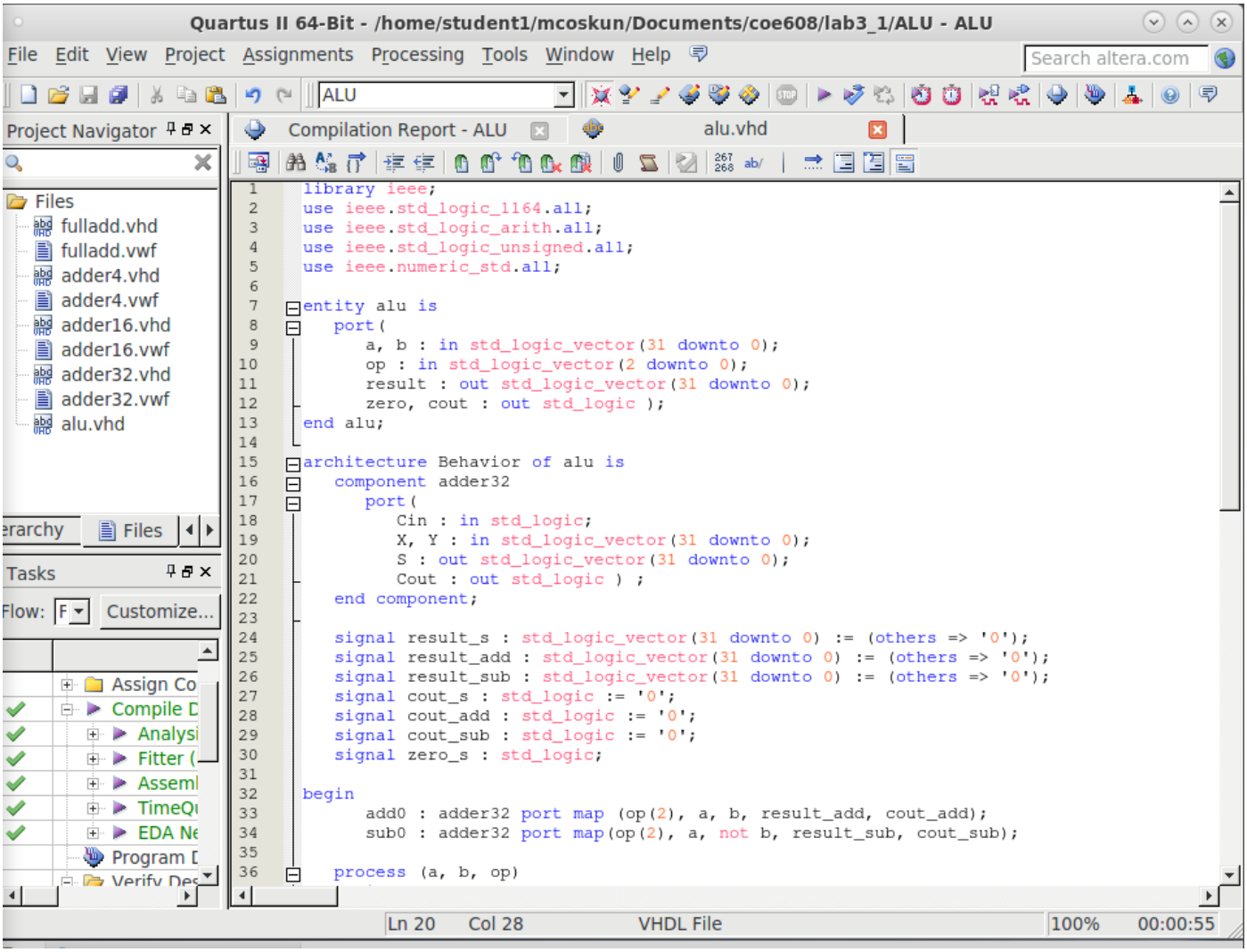
After compiling successfully, the waveform of 32-bit adder created following the lab manual is displayed below (Function waveform).

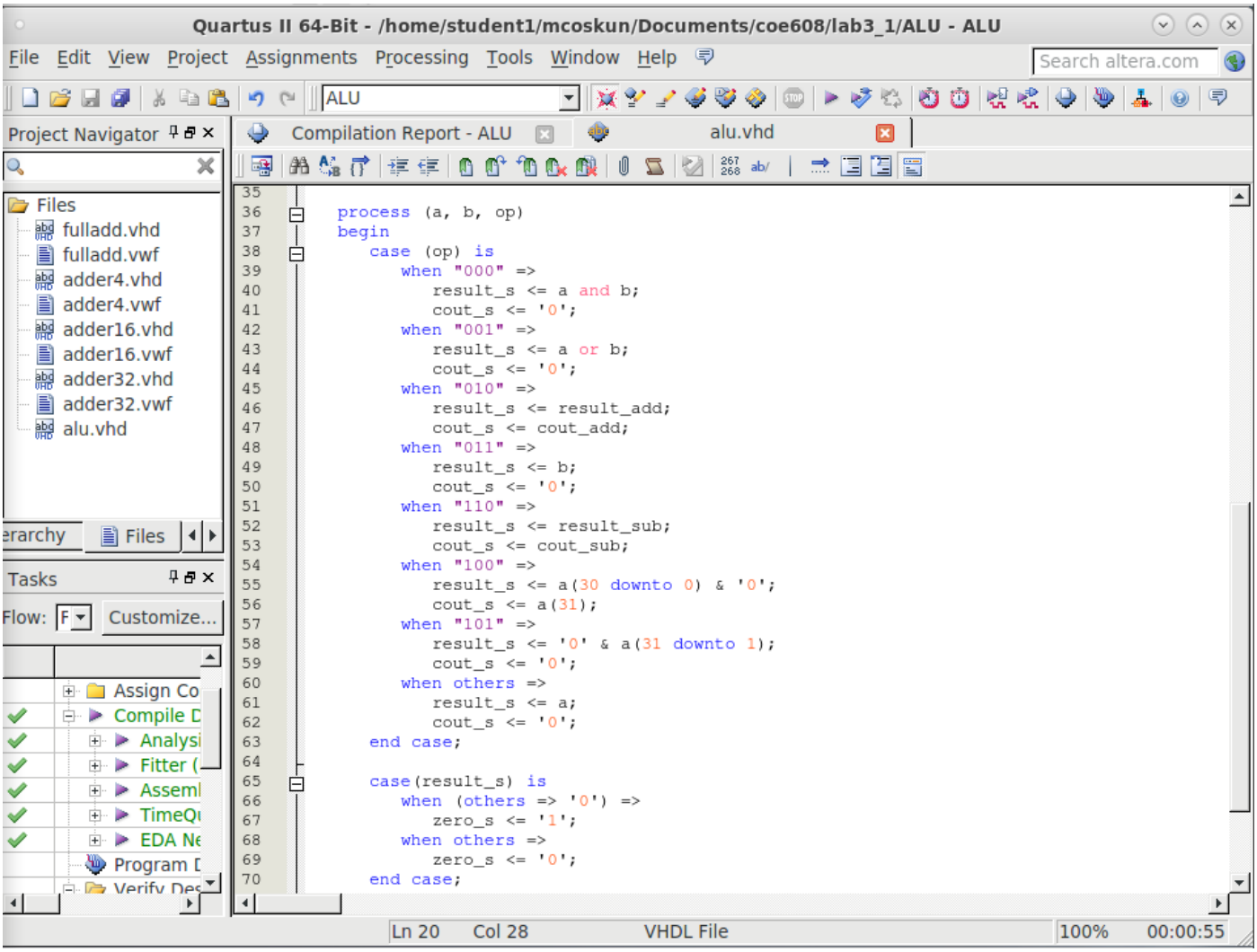


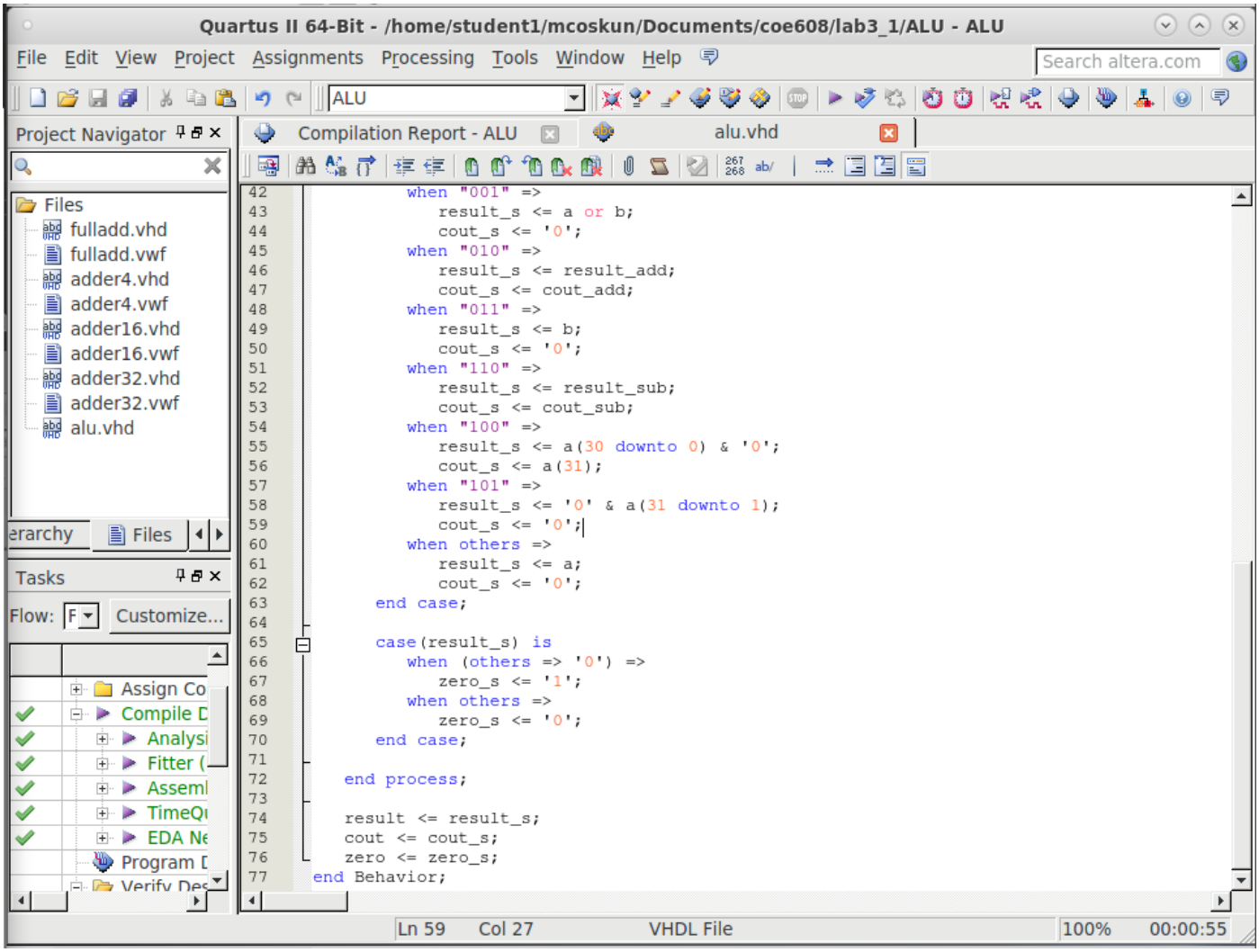
After compiling successfully, the waveform of 32-bit adder created following the lab manual is displayed below (Timing waveform).



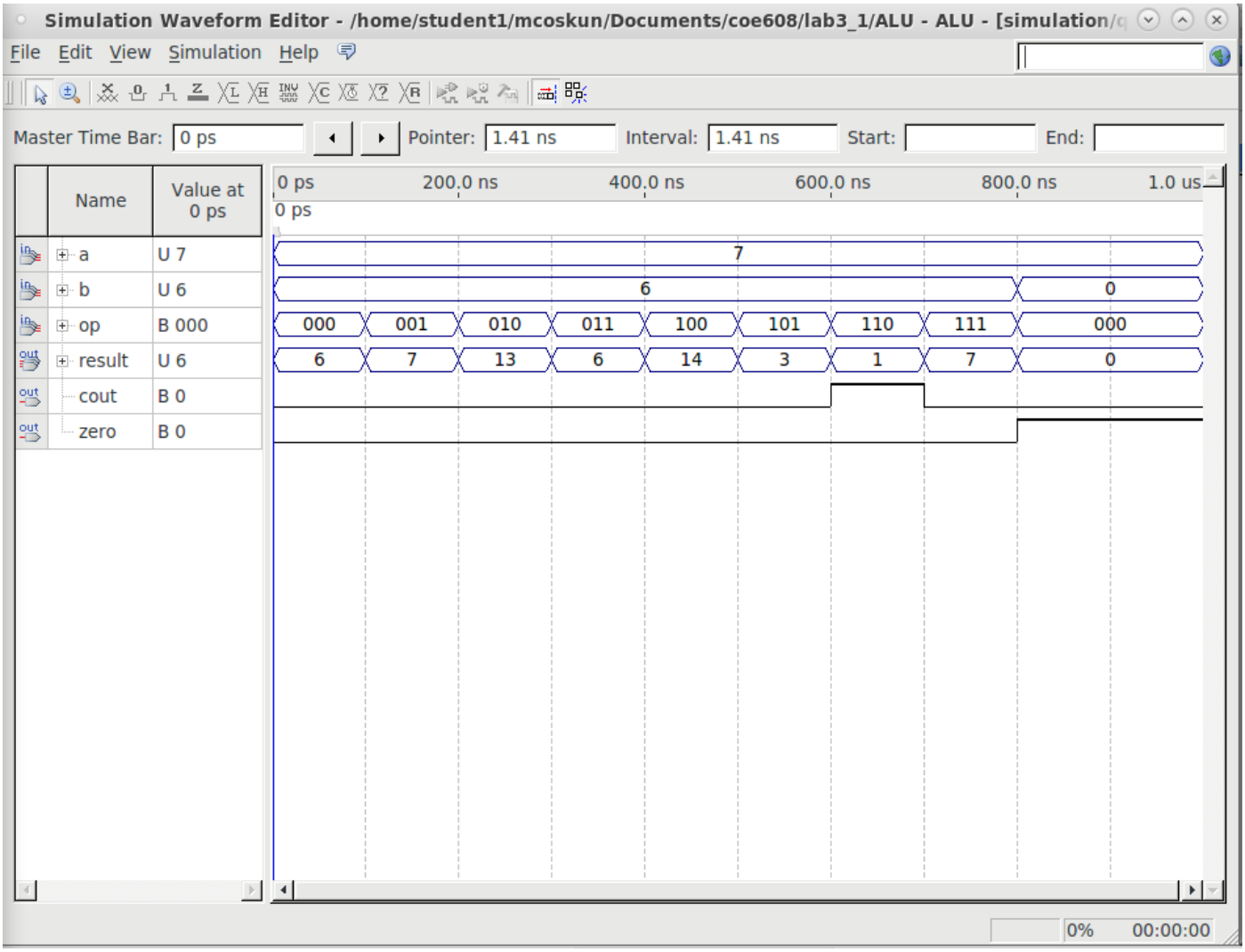
The VHDL code of the ALU can be seen below.



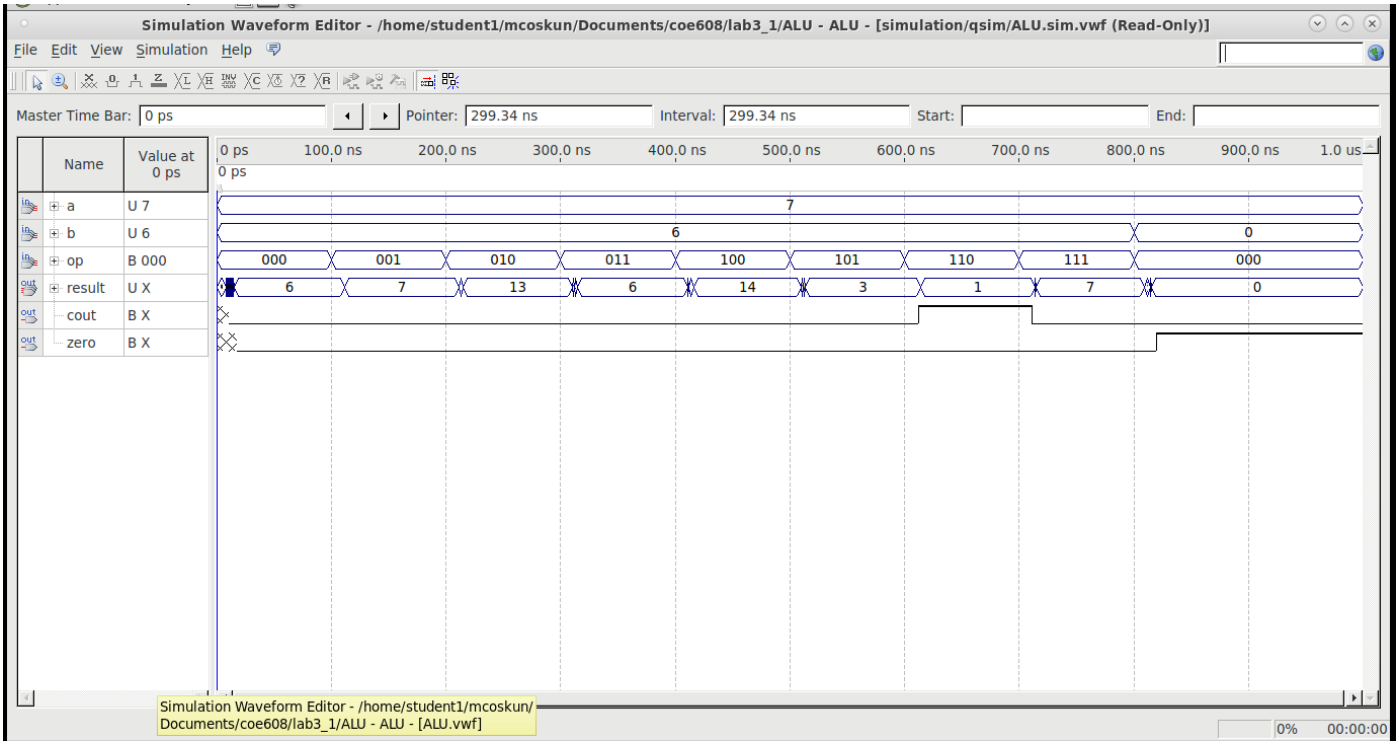

Quartus II 64-Bit - /home/student1/mcoskun/Documents/coe608/lab3_1/ALU - ALU
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
ALU
Project Navigator
Files
fulladd.vhd
fulladd.vwf
adder4.vhd
adder4.vwf
adder16.vhd
adder16.vwf
adder32.vhd
adder32.vwf
alu.vhd
Files
Tasks
Customize...
Compilation Report - ALU
alu.vhd
process (a, b, op)
begin
case (op) is
when "000" =>
result_s <= a and b;
cout_s <= '0';
when "001" =>
result_s <= a or b;
cout_s <= '0';
when "010" =>
result_s <= result_add;
cout_s <= cout_add;
when "011" =>
result_s <= b;
cout_s <= '0';
when "110" =>
result_s <= result_sub;
cout_s <= cout_sub;
when "100" =>
result_s <= a(30 downto 0) & '0';
cout_s <= a(31);
when "101" =>
result_s <= '0' & a(31 downto 1);
cout_s <= '0';
when others =>
result_s <= a;
cout_s <= '0';
end case;
case(result_s) is
when (others => '0') =>
zero_s <= '1';
when others =>
zero_s <= '0';
end case;
Ln 20 Col 28 VHDL File 100% 00:00:55


Quartus II 64-Bit - /home/student1/mcoskun/Documents/coe608/lab3_1/ALU - ALU
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
ALU
Project Navigator
Files
fulladd.vhd
fulladd.vwf
adder4.vhd
adder4.vwf
adder16.vhd
adder16.vwf
adder32.vhd
adder32.vwf
alu.vhd
Hierarchy
Files
Tasks
Flow:
Customize...
Assign Co
Compile D
Analysi
Fitter (
Assembl
TimeQu
EDA Ne
Program D
Verify Des
Compilation Report - ALU
alu.vhd
when "001" =>
result_s <= a or b;
cout_s <= '0';
when "010" =>
result_s <= result_add;
cout_s <= cout_add;
when "011" =>
result_s <= b;
cout_s <= '0';
when "110" =>
result_s <= result_sub;
cout_s <= cout_sub;
when "100" =>
result_s <= a(30 downto 0) & '0';
cout_s <= a(31);
when "101" =>
result_s <= '0' & a(31 downto 1);
cout_s <= '0';
when others =>
result_s <= a;
cout_s <= '0';
end case;
case(result_s) is
when (others => '0') =>
zero_s <= '1';
when others =>
zero_s <= '0';
end case;
end process;
result <= result_s;
cout <= cout_s;
zero <= zero_s;
end Behavior;
Ln 59 Col 27 VHDL File 100% 00:00:55

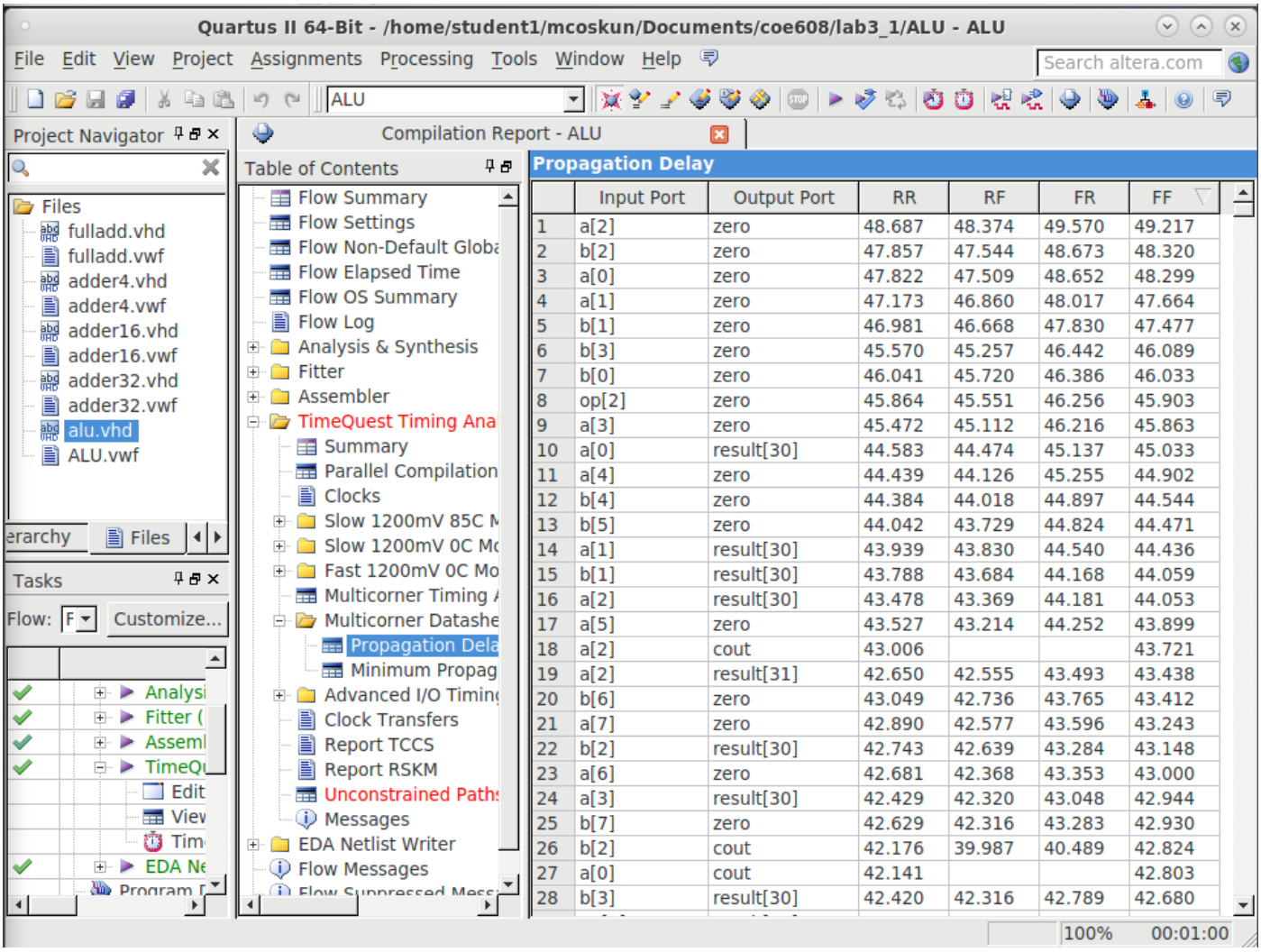
After compiling successfully, the waveform of ALU created following the lab manual is displayed below (Function waveform).



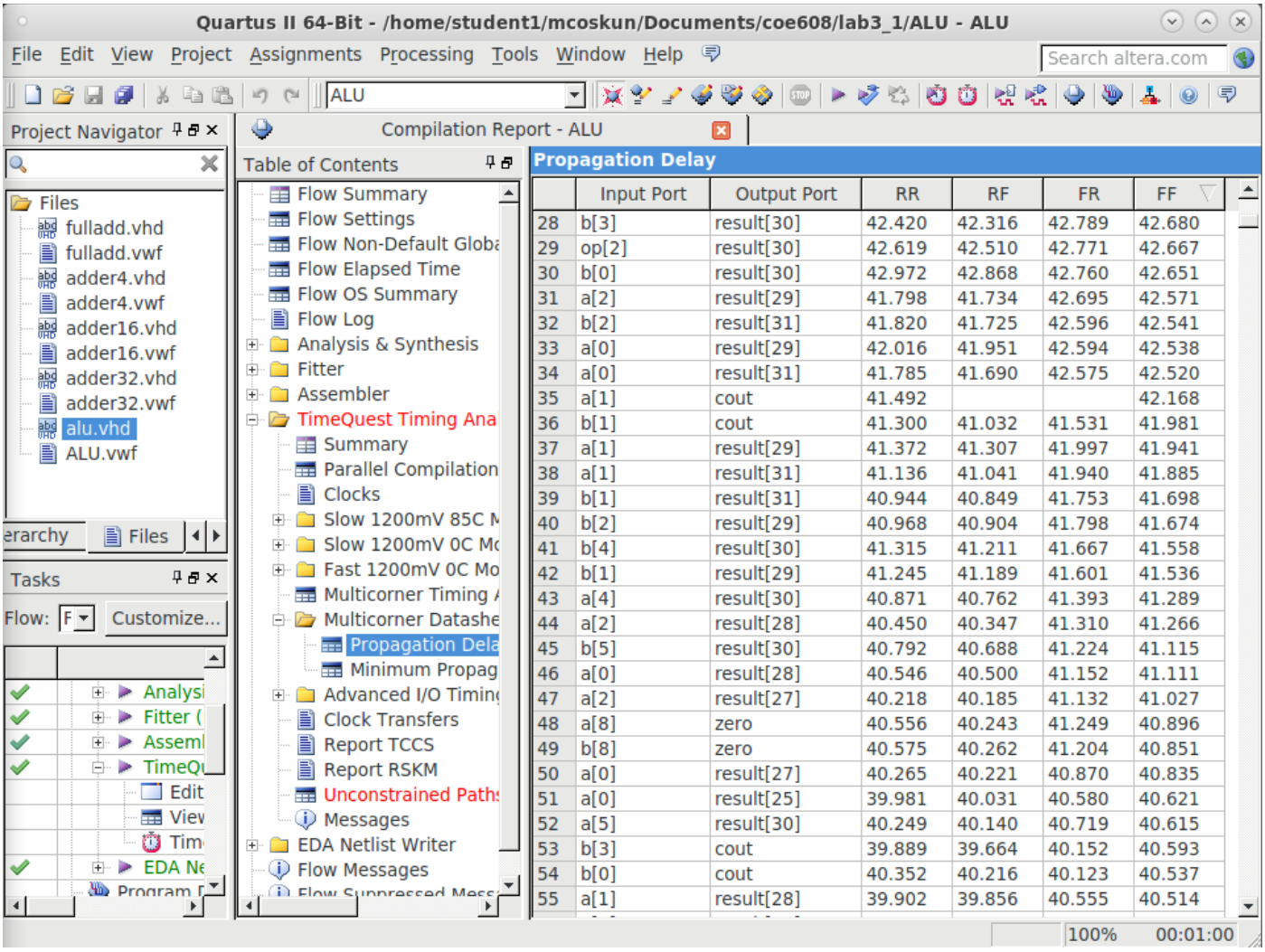
After compiling successfully, the waveform of ALU created following the lab manual is displayed below (Timing waveform).



Worst Case delays for various inputs and operations



	Input Port	Output Port	RR	RF	FR	FF
1	a[2]	zero	48.687	48.374	49.570	49.217
2	b[2]	zero	47.857	47.544	48.673	48.320
3	a[0]	zero	47.822	47.509	48.652	48.299
4	a[1]	zero	47.173	46.860	48.017	47.664
5	b[1]	zero	46.981	46.668	47.830	47.477
6	b[3]	zero	45.570	45.257	46.442	46.089
7	b[0]	zero	46.041	45.720	46.386	46.033
8	op[2]	zero	45.864	45.551	46.256	45.903
9	a[3]	zero	45.472	45.112	46.216	45.863
10	a[0]	result[30]	44.583	44.474	45.137	45.033
11	a[4]	zero	44.439	44.126	45.255	44.902
12	b[4]	zero	44.384	44.018	44.897	44.544
13	b[5]	zero	44.042	43.729	44.824	44.471
14	a[1]	result[30]	43.939	43.830	44.540	44.436
15	b[1]	result[30]	43.788	43.684	44.168	44.059
16	a[2]	result[30]	43.478	43.369	44.181	44.053
17	a[5]	zero	43.527	43.214	44.252	43.899
18	a[2]	cout	43.006			43.721
19	a[2]	result[31]	42.650	42.555	43.493	43.438
20	b[6]	zero	43.049	42.736	43.765	43.412
21	a[7]	zero	42.890	42.577	43.596	43.243
22	b[2]	result[30]	42.743	42.639	43.284	43.148
23	a[6]	zero	42.681	42.368	43.353	43.000
24	a[3]	result[30]	42.429	42.320	43.048	42.944
25	b[7]	zero	42.629	42.316	43.283	42.930
26	b[2]	cout	42.176	39.987	40.489	42.824
27	a[0]	cout	42.141			42.803
28	b[3]	result[30]	42.420	42.316	42.789	42.680



	Input Port	Output Port	RR	RF	FR	FF
28	b[3]	result[30]	42.420	42.316	42.789	42.680
29	op[2]	result[30]	42.619	42.510	42.771	42.667
30	b[0]	result[30]	42.972	42.868	42.760	42.651
31	a[2]	result[29]	41.798	41.734	42.695	42.571
32	b[2]	result[31]	41.820	41.725	42.596	42.541
33	a[0]	result[29]	42.016	41.951	42.594	42.538
34	a[0]	result[31]	41.785	41.690	42.575	42.520
35	a[1]	cout	41.492			42.168
36	b[1]	cout	41.300	41.032	41.531	41.981
37	a[1]	result[29]	41.372	41.307	41.997	41.941
38	a[1]	result[31]	41.136	41.041	41.940	41.885
39	b[1]	result[31]	40.944	40.849	41.753	41.698
40	b[2]	result[29]	40.968	40.904	41.798	41.674
41	b[4]	result[30]	41.315	41.211	41.667	41.558
42	b[1]	result[29]	41.245	41.189	41.601	41.536
43	a[4]	result[30]	40.871	40.762	41.393	41.289
44	a[2]	result[28]	40.450	40.347	41.310	41.266
45	b[5]	result[30]	40.792	40.688	41.224	41.115
46	a[0]	result[28]	40.546	40.500	41.152	41.111
47	a[2]	result[27]	40.218	40.185	41.132	41.027
48	a[8]	zero	40.556	40.243	41.249	40.896
49	b[8]	zero	40.575	40.262	41.204	40.851
50	a[0]	result[27]	40.265	40.221	40.870	40.835
51	a[0]	result[25]	39.981	40.031	40.580	40.621
52	a[5]	result[30]	40.249	40.140	40.719	40.615
53	b[3]	cout	39.889	39.664	40.152	40.593
54	b[0]	cout	40.352	40.216	40.123	40.537
55	a[1]	result[28]	39.902	39.856	40.555	40.514