

LAB 2: **ENCODER, PRIORITY ENCODER AND 8TO3** **ENCODER**

ARYA AGARWAL
Roll Number - 210070012

16 August 2022

PART A : 4TO2 ENCODER

1. Overview of the Experiment / Assignment:

- Encoder is a combination of gates that encodes a set of inputs into a given set of well defined output signals.
- 4 to 2 encoder means encoder with 4 inputs and 2 outputs.
- If Enable is 0, then the encoder will not work means all output 0.
- In this experiment our motive is to make circuit of encoder using AND and OR gates.

2. Experiment Setup or Approach to the Assignment:

- By observing the Truth table (given below) for 4to2 encoder, we try to formulate the possible circuit which is drawn below.

A	B	C	D	E = 1	Y1	Y0
0	0	0	1	1	0	0
0	0	1	0	1	0	1
0	1	0	0	1	0	0
1	0	0	0	1	1	1

- Therefore, the possible figure of encoder drawn is given in Fig 1

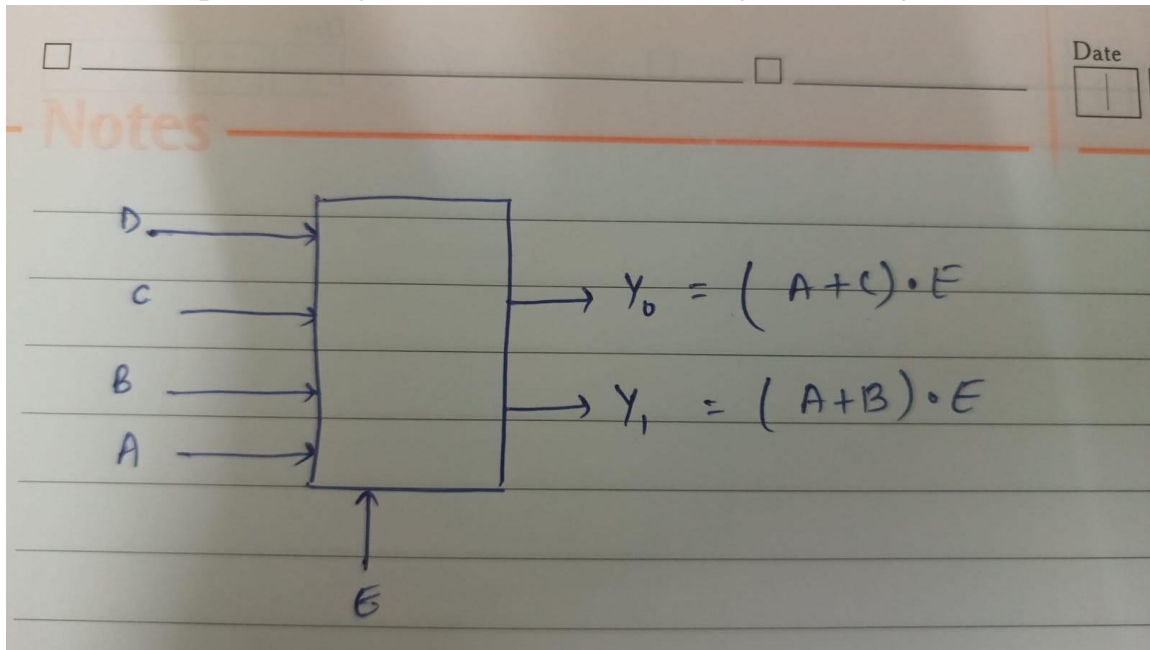


Fig 1

2.1 Design Code and Documentation:

Now once we have drawn the circuits on paper, its time to describe the circuit using code in vhdl language in Quartus. The dut, gates files will almost be same except some minor changes in both case.

The code for the main file is written below which is used to describe the circuit.

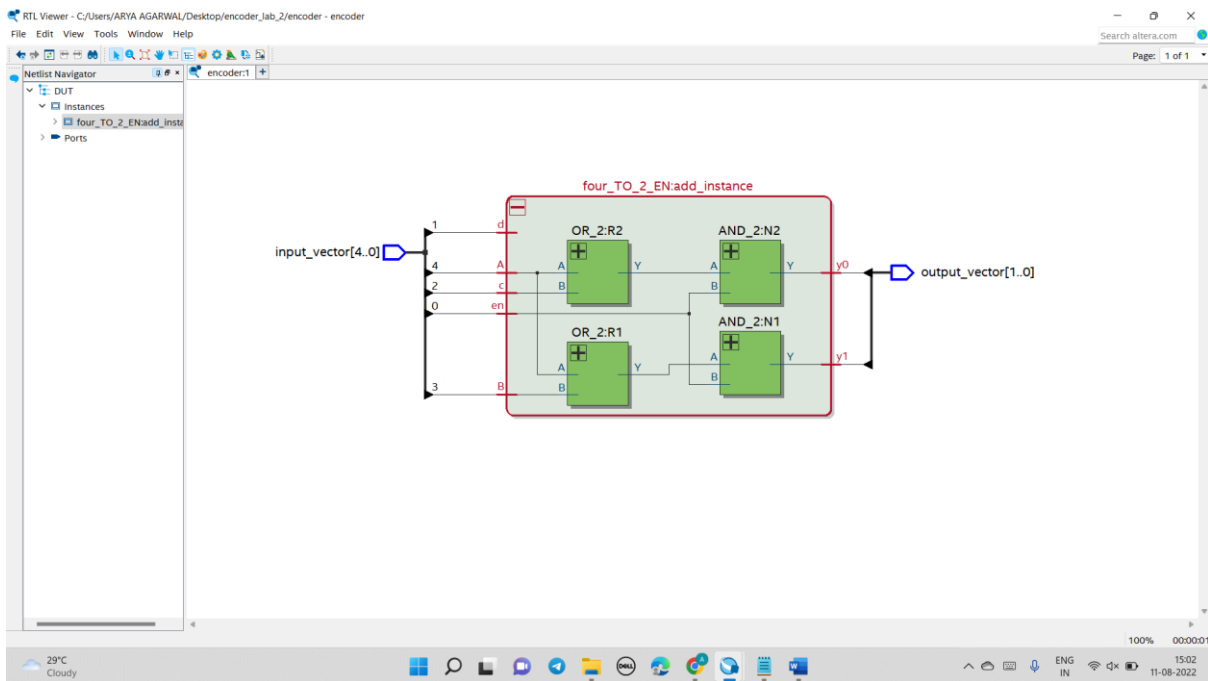
```

1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  library work;
5  use work.Gates.all;
6
7  entity four_TO_2_EN is
8  port (A,B,c,d,en: in std_logic;
9        y1,y0: out std_logic);
10 end entity four_TO_2_EN;
11
12 architecture struct of four_TO_2_EN is
13   signal s1,s2:std_logic;
14   begin
15     R1:OR_2
16     port map(A=>A, B=>B, Y=>s1);
17     R2:OR_2
18     port map(A=>A, B=>C, Y=>s2);
19     N1: AND_2
20     port map(A=>s1, B=>en, Y=>Y1);
21     N2: AND_2
22     port map(A=>s2, B=>en, Y=>Y0);
23   end struct;

```

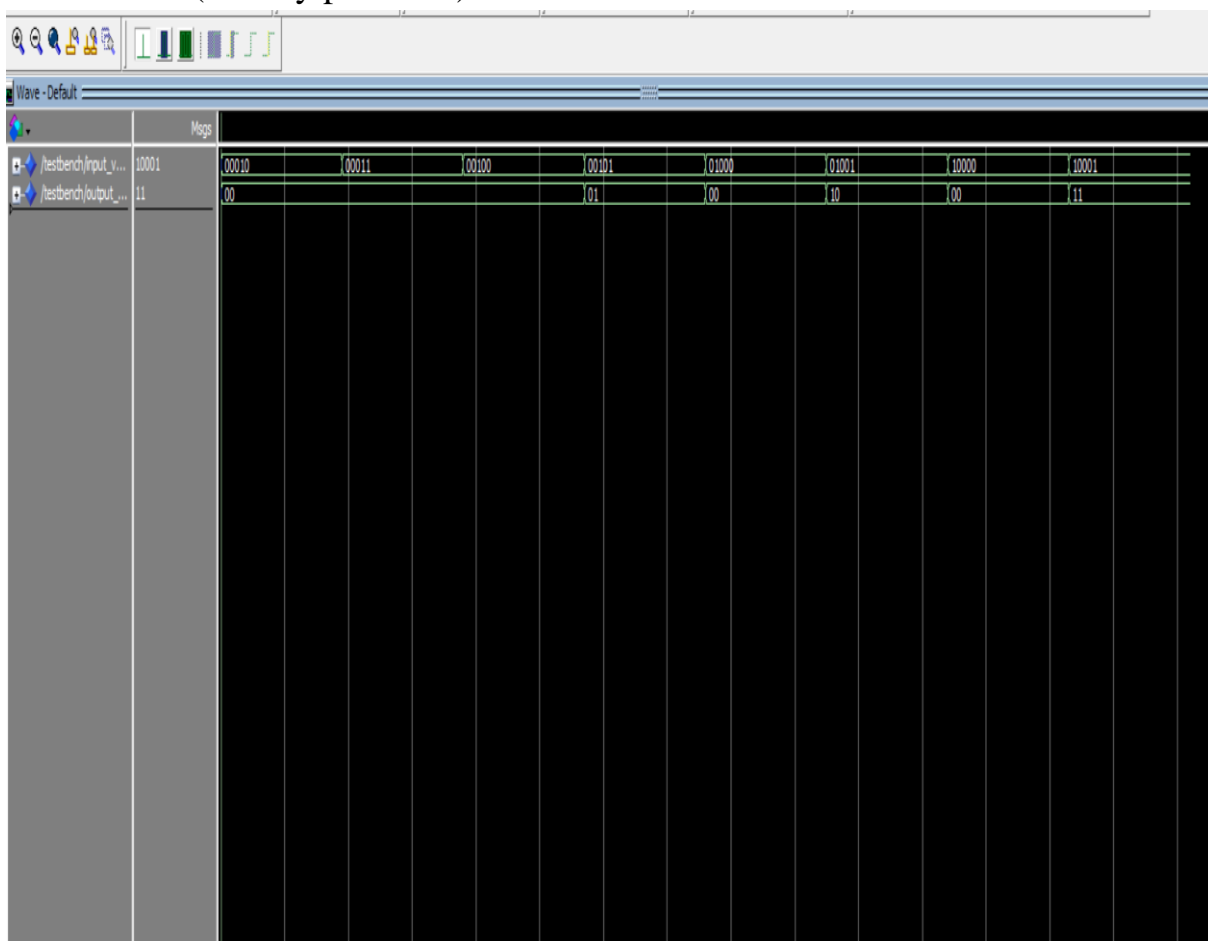
Digital System Lab

It is clear from the code that I have used 2 OR and 2 AND gates to fulfil my circuit. The RTL view of the code written above is shown below.



3. Observations:

After you run the analysis of code with no error. The next step is to run the simulation. For this we again use the Modelsim - Altera Software. We also need the Testbench and the tracefile (already provided) to run the simulation. The simulation is shown below.



TRANSCRIPT(all test cases passed successfully)

```

ModelSim - INTEL FPGA STARTER EDITION 2020.1
File Edit View Compile Simulate Add Transcript Tools Layout Bookmarks Window Help
ColumnLayout [AllColumns]
100 ps
Layout Simulate

Transcript
# -- Compiling entity four_to_2_en
# -- Compiling architecture struct of four_to_2_en
# End time: 14:58:16 on Aug 11, 2022, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
#
# vcom -93 -work work (C:/Users/ARYA AGARWAL/Desktop/encoder_lab_2/Testbench.vhdl)
# Model Technology ModelSim - Intel FPGA Edition vcom 2020.1 Compiler 2020.02 Feb 28 2020
# Start time: 14:58:16 on Aug 11, 2022
# vcom -reportprogress 300 -93 -work work C:/Users/ARYA AGARWAL/Desktop/encoder_lab_2/Testbench.vhdl
# -- Loading package STANDARD
# -- Loading package TEXTIO
# -- Loading package std_logic_1164
# -- Compiling entity Testbench
# -- Compiling architecture Behave of Testbench
# End time: 14:58:16 on Aug 11, 2022, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
#
# vsim -t lps -L altera -L lpm -L sgate -L altera_mf -L altera_insim -L fiftyfive -L rtl_work -L work -voptargs="+acc" Testbench
# vsim -t lps -L altera -L lpm -L sgate -L altera_mf -L altera_insim -L fiftyfive -L rtl_work -L work -voptargs="+acc" Testbench
# Start time: 14:58:17 on Aug 11, 2022
# Loading std.standard
# Loading std.textio(body)
# Loading ieee.std_logic_1164(body)
# Loading work.testbench(behave)
# Loading work.dut(dutwrap)
# Loading work.gates
# Loading work.four_to_2_en(struct)
# Loading work.or_2(equations)
# Loading work.and_2(equations)
#
# add wave *
# view structure
# .main-pane.structure.interior.cs.body.struct
# view signals
# .main-pane.objects.interior.cs.body.tree
# run -all
# ** Note: SUCCESS, all tests passed.
# Time: 152 ns Iteration: 0 Instance: /testbench

VSM2>

```

PART B: PRIORITY ENCODER

1. Overview of the Experiment / Assignment:

- The job of a priority encoder is to produce a binary output address for the input with the highest priority.
- It is also a 4to2 encoder with same number of inputs and outputs but with different characteristics.
- Here in this experiment we will draw the circuit for Priority Encoder using the truth table.

2. Experiment Setup or Approach to the Assignment:

- Below is the truth table for the priority encoder.

Note - X represents Don't Care meaning the output does not depend on the particular input.

A	B	C	D	V	Y1	Y0
0	0	0	0	0	0	0
0	0	0	1	1	0	0
0	0	1	X	1	0	1
0	1	X	X	1	1	0
1	X	X	X	1	1	1

2.1 Design Code and Documentation:

Now once we have drawn the circuits on paper, its time to describe the circuit using code in vhdl language in Quartus. The dut, gates files will almost be same except some minor changes in both case.

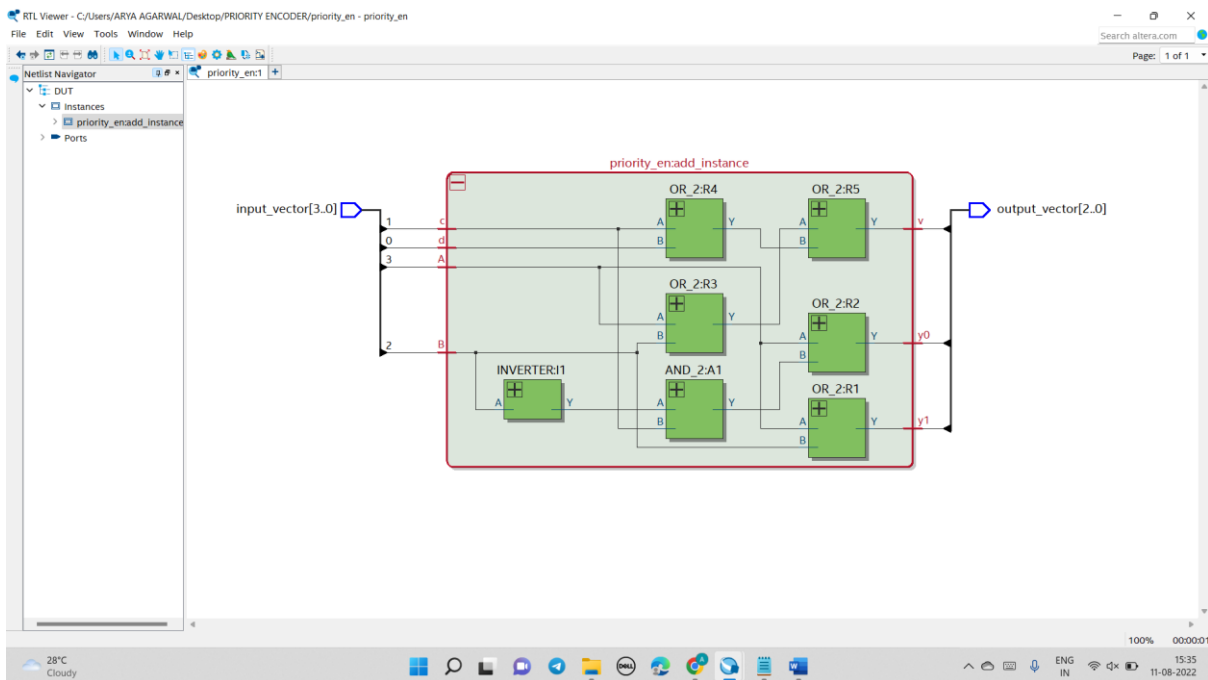
Now its time to write the code for the circuit drawn above in order to run its simulation. We have already made another folder. Now we will make a file in which I would write the code to describe the circuit. The same code is also written below.

```

1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  library work;
5  use work.Gates.all;
6
7  entity priority_en is
8  port (A,B,c,d: in std_logic;
9        y1,y0,v: out std_logic);
10 end entity priority_en;
11
12 architecture struct of priority_en is
13   signal s1,s2,s3,s4:std_logic;
14   begin
15     R1:OR_2
16     port map(A=>A, B=>B, Y=>y1);
17     I1: INVERTER
18     port map(A=>b, y=>s1);
19
20     A1: AND_2
21     port map(A=>s1, B=>c, Y=>s2);
22     R2:OR_2
23     port map(A=>A, B=>s2, Y=>y0);
24     R3:OR_2
25     port map(A=>A, B=>B, Y=>s3);
26     R4:OR_2
27     port map(A=>c, B=>d, Y=>s4);
28     R5:OR_2
29     port map(A=>s3, B=>s4, Y=>v);
30
31   end struct;

```

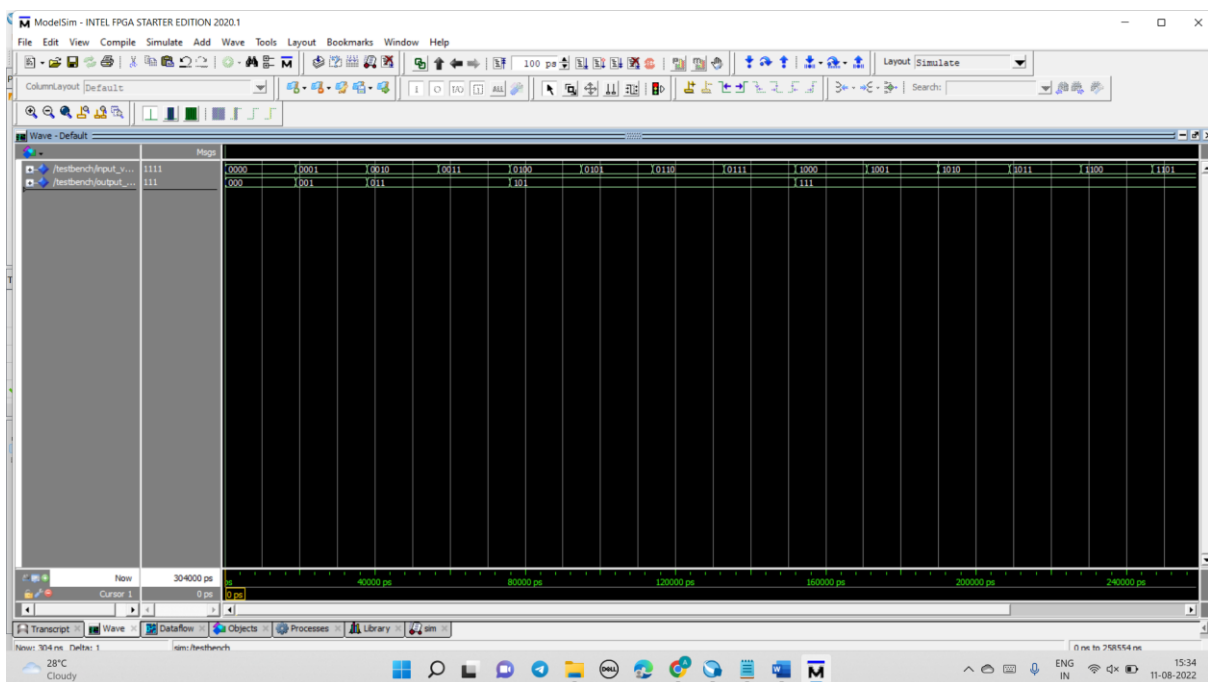
RTL VIEW



3. Observations:

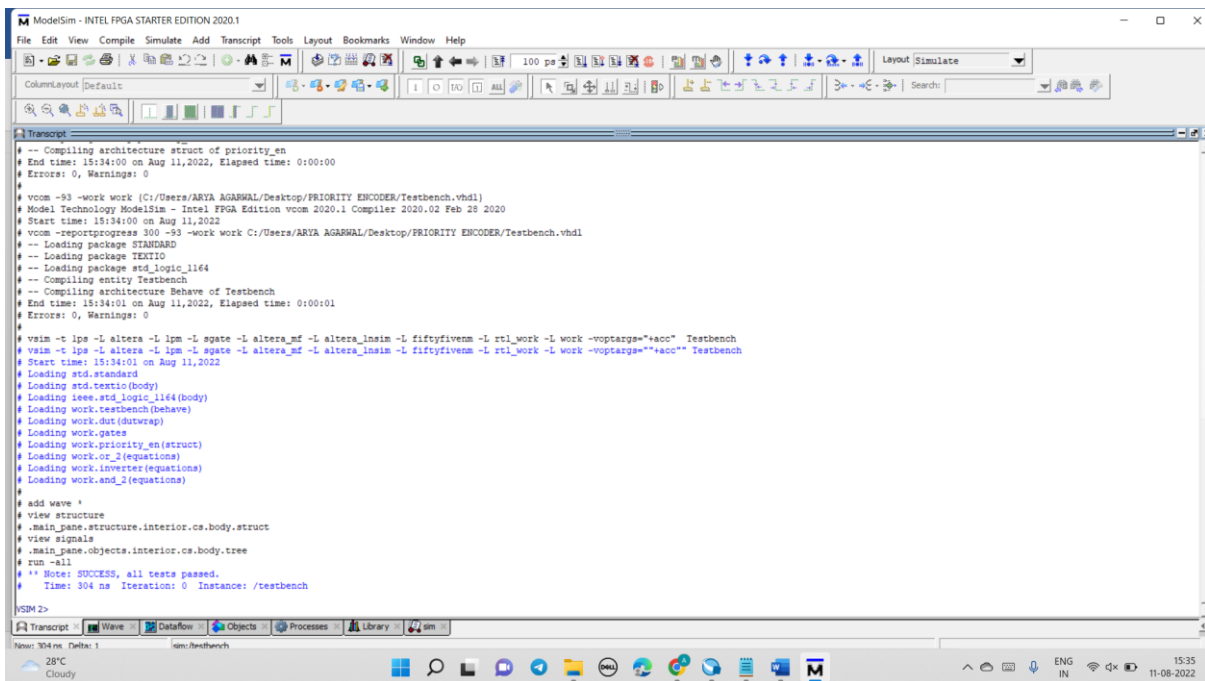
After you run the analysis of code with no error. The next step is to run the simulation. For this we again use the Modelsim - Altera Software. We also need the Testbench and the tracefile (already provided) to run the simulation.

The simulation thus obtained for the above code/circuit is shown below.



It can be easily observed that the above simulation correctly matches with the trace file as well as the Table which we wrote in Experiment Setup section. Thus we can conclude that our Experiment is successful.

TRANSCRIPT(all test cases passed successfully)



```
ModelSim - INTEL FPGA STARTER EDITION 2020.1
File Edit View Compile Simulate Add Transcript Tools Layout Bookmarks Window Help

ColumnLayout Default

Transcript
# -- Compiling architecture struct of priority_en
# End time: 15:34:00 on Aug 11,2022, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
#
# vcom -93 -work work [C:/Users/ARYA AGARWAL/Desktop/PRIORITY ENCODER/Testbench.vhdl]
# Model Technology ModelSim - Intel FPGA Edition vcom 2020.1 Compiler 2020.02 Feb 28 2020
# Start time: 15:34:00 on Aug 11,2022
# vcom -reportprogress 500 -93 -work work C:/Users/ARYA AGARWAL/Desktop/PRIORITY ENCODER/Testbench.vhdl
# -- Loading package STANDARD
# -- Loading package TEXTIO
# -- Loading package std_logic_1164
# -- Compiling entity Testbench
# -- Compiling architecture Behave of Testbench
# End time: 15:34:01 on Aug 11,2022, Elapsed time: 0:00:01
# Errors: 0, Warnings: 0
#
# vsim -t lps -L altera -L lpm -L sgate -L altera_mf -L altera_insim -L fiftyfivenm -L rtl_work -L work -voptargs="+acc" Testbench
# vsim -t lps -L altera -L lpm -L sgate -L altera_mf -L altera_insim -L fiftyfivenm -L rtl_work -L work -voptargs="+acc" Testbench
# Start time: 15:34:01 on Aug 11,2022
# Loading std.standard
# Loading std.textio(body)
# Loading ieee.std_logic_1164(body)
# Loading work.testbench(behave)
# Loading work.dut(dutwrap)
# Loading work.gates
# Loading work.priority_en(struct)
# Loading work.or_2(equations)
# Loading work.inverter(equations)
# Loading work.and_2(equations)
#
# add wave *
# view structure
# .main_pane.structure.interior.cs.body.struct
# view signals
# .main_pane.objects.interior.cs.body.tree
# run -all
# ** Note: SUCCESS, all tests passed.
# Time: 304 ns Iteration: 0 Instance: /testbench

VSIOM 2>
News: 304 ns Delta: 1
sim:/testbench
```

PART C: 8TO3 ENCODER

1. Overview of the Experiment.

- 8 to 3 encoder is an extended version of 4 to 2 encoder that we have already made.
- We know that number of output n is given as \log_2 to the base 2 and argument number of inputs. Thus for 8 input we have 3 output.
- The purpose of this experiment is to draw the circuit of 8to3 encoder using only 4to2 encoder and OR gates.
- Then as usual we are required to code the circuit made and run its simulation based on the trace file given.

2. Experiment Setup or Approach to the Assignment:

- Firstly we need to make the truth table based on the characteristics of the 8to3 encoder.
- The truth table for 8to3 encoder is drawn below. Do not forget the Enable (E) input that we used in the first experiment.

Note - X represents Don't Care meaning the output does not depend on the particular input.

I7	I6	I5	I4	I3	I2	I1	I0	E	A2	A1	A0
X	X	X	X	X	X	X	X	0	0	0	0
0	0	0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	0	1	0	1	0	0	1
0	0	0	0	0	1	0	0	1	0	1	0
0	0	0	0	1	0	0	0	1	0	1	1
0	0	0	1	0	0	0	0	1	1	0	0
0	0	1	0	0	0	0	0	1	1	0	1
0	1	0	0	0	0	0	0	1	1	1	0
1	0	0	0	0	0	0	0	1	1	1	1

Based on the truth table drawn above it is easy to observe the following equations of the output of the encoder. $A0 = E$ and $(I1 + I3 + I5 + I7)$

$A1 = E$ and $(I2 + I3 + I6 + I7)$

$A2 = E$ and $(I4 + I5 + I6 + I7)$

Now as we are directed to only use 4to2 encoder whose output we have already understood from the first experiment.

2.1 Design Code and Documentation:

Now once we have drawn the circuits on paper, its time to describe the circuit using code in vhdl language in Quartus. The dut, gates files will almost be same except some minor changes in both case.

Now its time to write the code for the circuit drawn above in order to run its simulation. We have already made another folder. Now we will make a file in which I would write the code to describe the circuit. The same code is also written below.

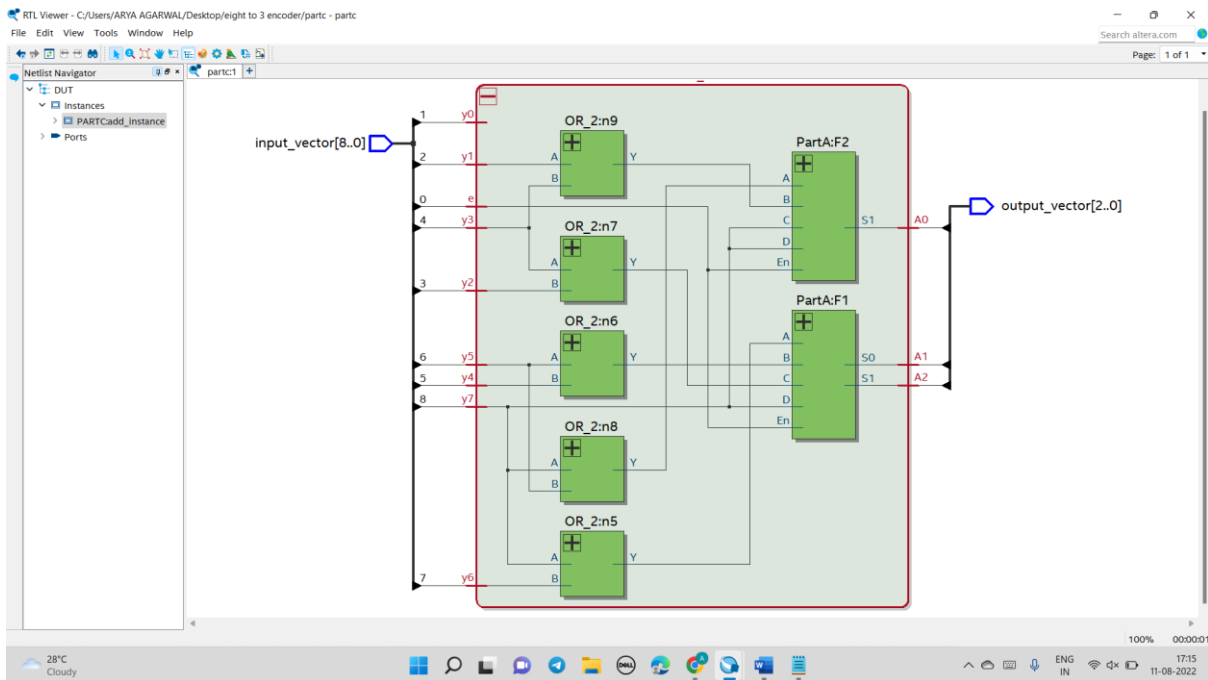
We have declared an entity named PARTA in order to reduce complexity.

```

1  library ieee;
2  use ieee.std_logic_1164.all;
3  library work;
4  use work.Gates.all;
5  entity PartA is
6  | port (A, B, C, D, En: in std_logic; S0, S1: out std_logic);
7  | end entity PartA;
8  architecture Struct of PartA is
9  | signal U,V: Std_logic;
10 | begin
11 |     -- component instances
12 |     n1 : OR_2    port map (A => A, B => C, Y => U); -- complete this port map
13 |     n2 : OR_2    port map (A => A, B => B, Y => V);
14 |     n3 : AND_2   port map (A => U, B => En, Y => S0);
15 |     n4 : AND_2   port map (A => V, B => En, Y => S1);
16 |
17 |
18 |
19 | end Struct;
20 |
21 | -----PARTC-----
22 | library ieee;
23 | use ieee.std_logic_1164.all;
24 | library work;
25 | use work.Gates.all;
26 |
27 | entity PARTC is
28 | | port (y0, y1, y2, y3, y4, y5, y6, y7, e:in std_logic; A2, A1, A0: out std_logic);
29 | | end entity PARTC;
30 | architecture struct of PARTC is
31 | | component PARTA is
32 | | | port (A, B, C, D, En: in std_logic; S0, S1: out std_logic);
33 | | | end component;
34 | | | signal s1, s2, s3, s4, s5, s6: Std_logic;
35 | | | begin
36 | | | n5 : OR_2    port map (A => y7, B => y6, Y => s1);
37 | | | n6 : OR_2    port map (A => y5, B => y4, Y => s2);
38 | | | n7 : OR_2    port map (A => y3, B => y2, Y => s3);
39 | | | n8 : OR_2    port map (A => y7, B => y5, Y => s4);
40 | | | n9 : OR_2    port map (A => y1, B => y3, Y => s5);
41 | | | F1 : PARTA   port map (A => s1, B => s2, C => s3, D => y7, En => e, S0 => A1, S1 => A2);
42 | | | F2 : PARTA   port map (A => s4, B => s5, C => y7, D => y7, En => e, S0 => s6, S1 => A0);
43 | | |
44 | | |
45 | | | end Struct;

```

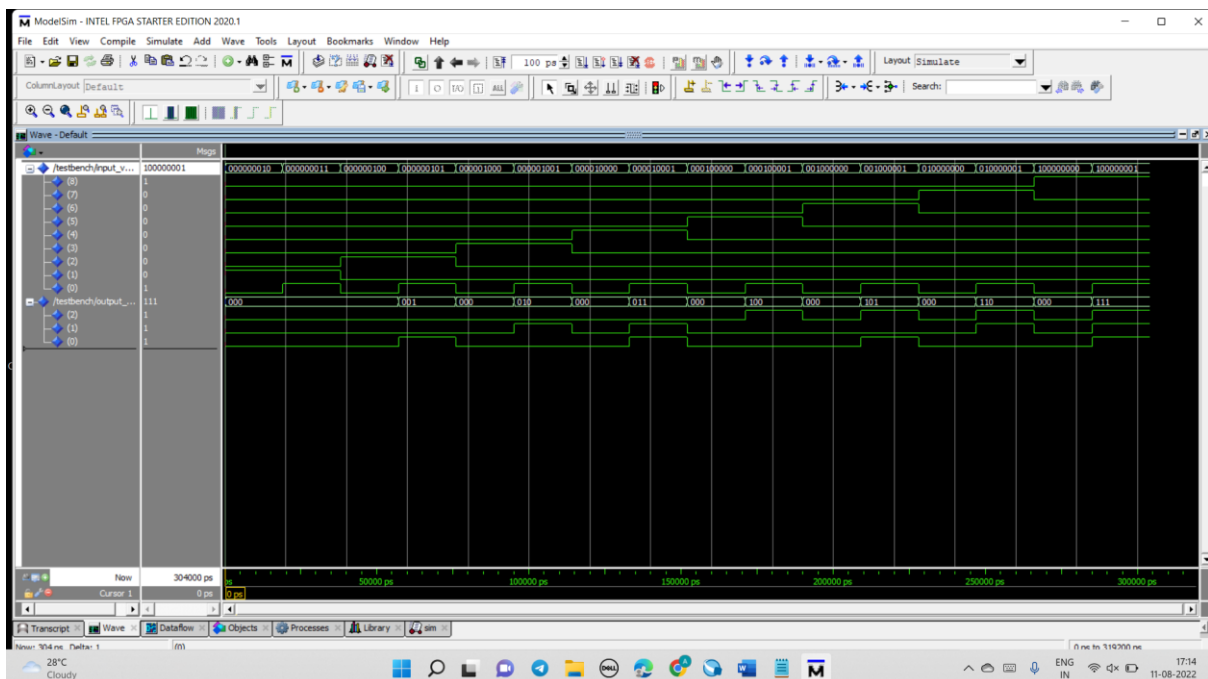
RTL VIEW



3. Observations:

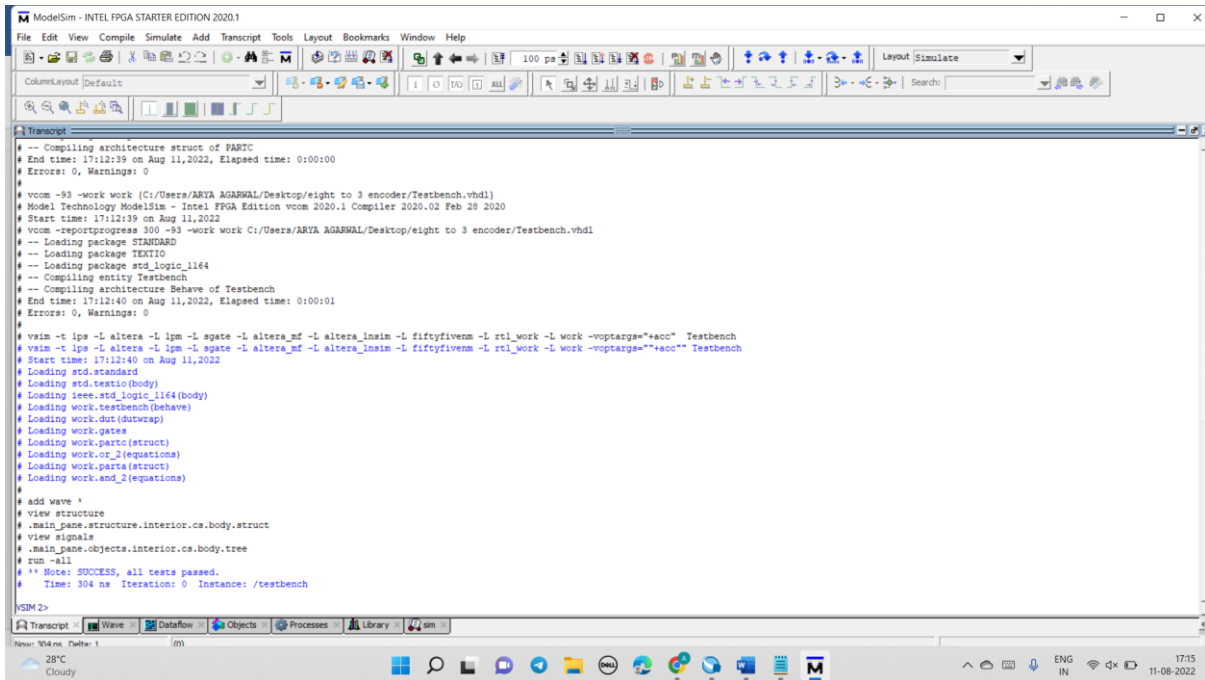
After you run the analysis of code with no error. The next step is to run the simulation. For this we again use the Modelsim - Altera Software. We also need the Testbench and the tracefile (already provided) to run the simulation.

The simulation thus obtained for the above code/circuit is shown below.



It can be easily observed that the above simulation correctly matches with the trace file as well as the Table which we wrote in Experiment Setup section. Thus we can conclude that our Experiment is successful.

TRANSCRIPT(all test cases passed successfully)



```
ModelSim - INTEL FPGA STARTER EDITION 2020.1
File Edit View Compile Simulate Add Transcript Tools Layout Bookmarks Window Help

ColumnLayout: Default

Transcript
# -- Compiling architecture struct of PARTIC
# End time: 17:12:39 on Aug 11,2022, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
#
# vcom -93 -work work [C:/Users/ARYA AGARWAL/Desktop/eight to 3 encoder/Testbench.vhdl]
# Model Technology ModelSim - Intel FPGA Edition vcom 2020.1 Compiler 2020.02 Feb 28 2020
# Start time: 17:12:39 on Aug 11,2022
# vcom -reportprogress 300 -93 -work work C:/Users/ARYA AGARWAL/Desktop/eight to 3 encoder/Testbench.vhdl
# -- Loading package STANDARD
# -- Loading package TEXTIO
# -- Loading package std_logic_1164
# -- Compiling entity Testbench
# -- Compiling architecture Behave of Testbench
# End time: 17:12:40 on Aug 11,2022, Elapsed time: 0:00:01
# Errors: 0, Warnings: 0
#
# vsim -t lps -L altera -L lpm -L sgate -L altera_mf -L altera_insim -L fiftyfivevnm -L rtl_work -L work -voptargs="+acc" Testbench
# vsim -t lps -L altera -L lpm -L sgate -L altera_mf -L altera_insim -L fiftyfivevnm -L rtl_work -L work -voptargs="+acc" Testbench
# Start time: 17:12:40 on Aug 11,2022
# Loading std.standard
# Loading std.textio(body)
# Loading ieee.std_logic_1164(body)
# Loading work.testbench(behavior)
# Loading work.dut(dutwrap)
# Loading work.gates
# Loading work.parc2(struct)
# Loading work.or_2(equations)
# Loading work.parc2a(struct)
# Loading work.and_2(equations)
#
# add wave *
# view structure
# .main_pane.structure.interior.cs.body.struct
# view signals
# .main_pane.objects.interior.cs.body.tree
# run -all
# ** Note: SUCCESS, all tests passed.
# Time: 304 ns Iteration: 0 Instance: /testbench

VSM 2>
```

4. Reference for all above experiment.

All the images have been taken from the experiments done in lab on Quartus software. Special mention to the Modelsim software which is used to run the simulation.