EXPT.-04

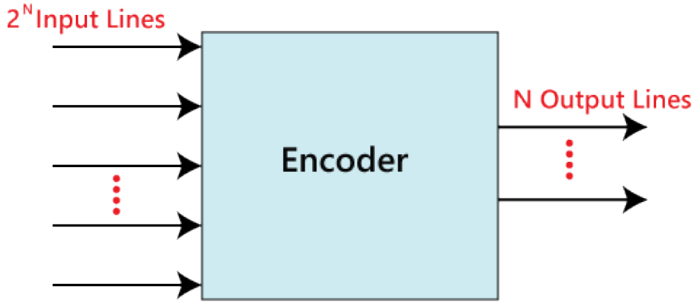
AIM: - Design and Simulate 4:2 Encoder/ 2:4 Decoder using Verilog HDL.

Software: - Xilinx Vivado 2019.1

Theory: -

Encoders

The combinational circuits that change the binary information into N output lines are known as **Encoders**. The binary information is passed in the form of 2^N input lines. The output lines define the N-bit code for the binary information. In simple words, the **Encoder** performs the reverse operation of the **Decoder**. At a time, only one input line is activated for simplicity. The produced N-bit output code is equivalent to the binary information.

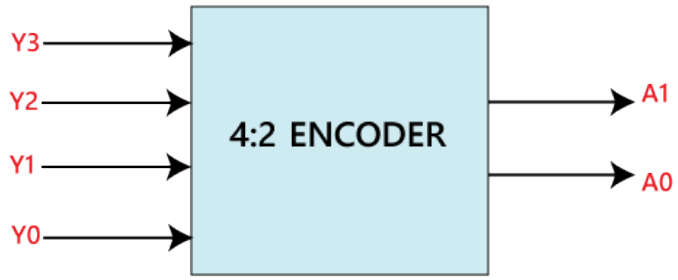


There are various types of encoders which are as follows:

4 to 2 line Encoder:

In 4 to 2 line encoder, there are total of four inputs, i.e., Y_0, Y_1, Y_2, and Y_3, and two outputs, i.e., A_0 and A_1. In 4-input lines, one input-line is set to true at a time to get the respective binary code in the output side. Below are the block diagram and the truth table of the 4 to 2 line encoder.

Block Diagram:



Truth Table:

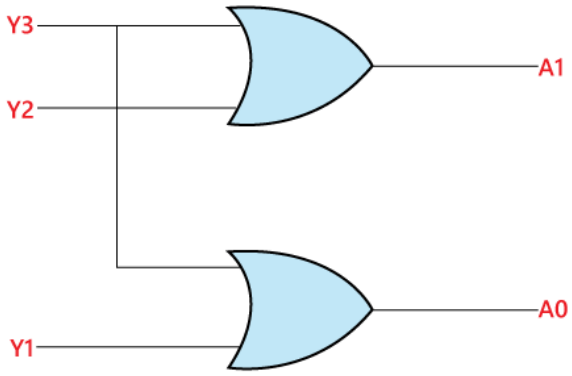
INPUTS				OUTPUTS	
Y_3	Y_2	Y_1	Y_0	A_1	A_0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

The logical expression of the term A0 and A1 is as follows:

$A_1=Y_3+Y_2$

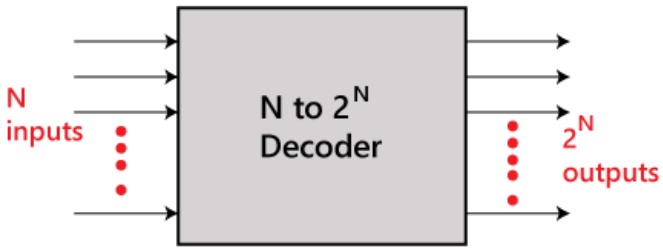
$A_0=Y_3+Y_1$

Logical circuit of the above expressions is given below:



Decoder

The combinational circuit that changes the binary information into 2^N output lines is known as **Decoders.** The binary information is passed in the form of N input lines. The output lines define the 2^N-bit code for the binary information. In simple words, the **Decoder** performs the reverse operation of the **Encoder**. At a time, only one input line is activated for simplicity. The produced 2^N-bit output code is equivalent to the binary information.

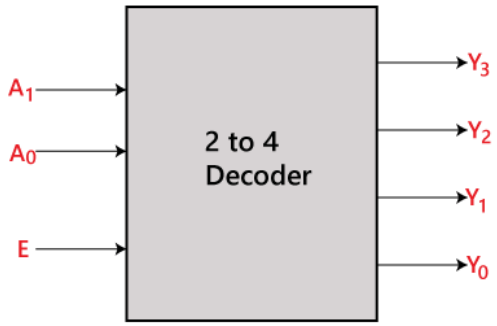


There are various types of decoders which are as follows:

2 to 4 line decoder:

In the 2 to 4 line decoder, there is a total of three inputs, i.e., A_0, and A_1 and E and four outputs, i.e., Y_0, Y_1, Y_2, and Y_3. For each combination of inputs, when the enable 'E' is set to 1, one of these four outputs will be 1. The block diagram and the truth table of the 2 to 4 line decoder are given below.

Block Diagram:



Truth Table:

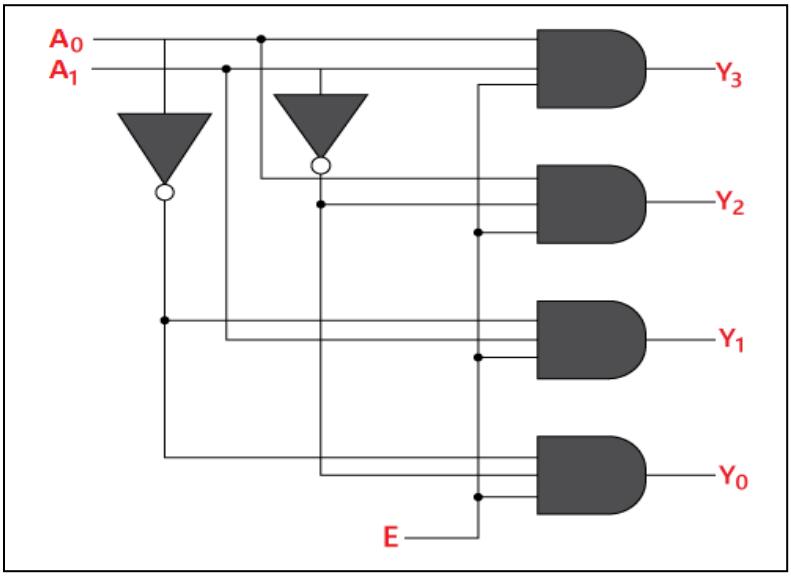
Enable	INPUTS		OUTPUTS			
E	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

The logical expression of the term Y0, Y0, Y2, and Y3 is as follows:

$Y_3=E.A_1.A_0$
$Y_2=E.A_1.A_0'$
$Y_1=E.A_1'.A_0$
$Y0=E.A_1'.A_0'$

Logical circuit of the

above expressions is given below:



Code: -

```
//encoder
module encoder_4to2(in,out);
input [3:0] in;
output reg [1:0] out;

always @(*)
begin

case(in)
4'b0001: out=2'b00;
4'b0010: out=2'b01;
4'b0100: out=2'b10;
4'b1000: out=2'b11;
default:out=2'bxx;
endcase

end
endmodule
```

```
//tb encoder
`timescale 1ns / 1ps

module enc_4to2_tb();
reg [3:0] in;
wire [1:0] out;

encoder_4to2 enc1(.in(in),.out(out));

initial
begin
in=4'b1000;
#10 in=4'b0100;
#10 in=4'b0010;
#10 in=4'b0001;
#10 in=4'b0000;
end

initial
$monitor($time,"in=%b,out=%b",in,out);

initial
#60 $finish;

endmodule
```

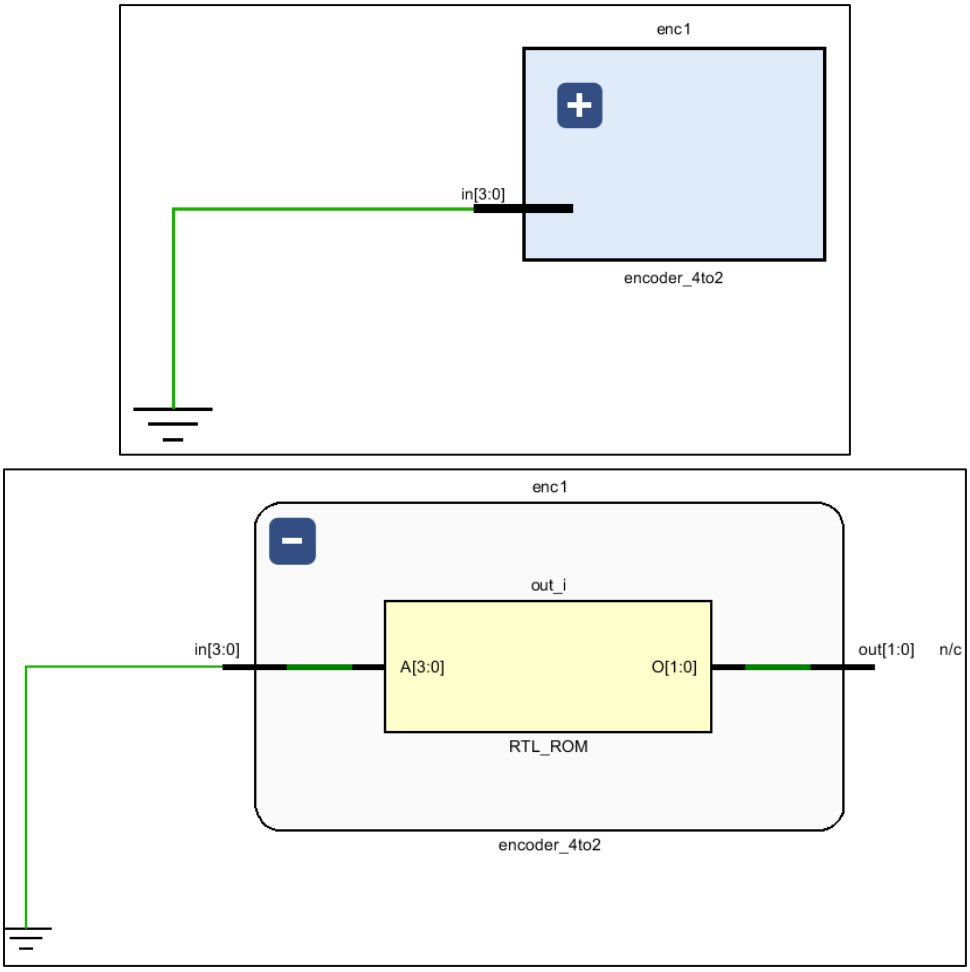
```
//decoder
`timescale 1ns / 1ps

module Decoder_2to4(in,out,en);
input [1:0] in;
input en;
output reg [3:0] out;

always @(*)
begin

if(en)

case(in)
2'b00: out<=4'b1000;
2'b01: out<=4'b0100;
2'b10: out<=4'b0010;
2'b11: out<=4'b0001;
default:out<=4'bxxxx;
endcase

else
out<=4'bxxxx;

end
endmodule
```

```
//tb decoder
module tb_Dec_2to4();
reg [1:0] in;
reg en;
wire [3:0] out;

Decoder_2to4 D1(.in(in),.out(out),.en(en));

initial
$monitor($time,"in=%b,out=%b",in,out);

integer i;
initial
begin
in=2'b00;

repeat(2)
for (i=0;i<4;i=i+1)
begin
in=i;
#10;
end

end

initial
begin
en=1'b1;
#40 en=~en;
end

initial
#80 $finish;

endmodule
```

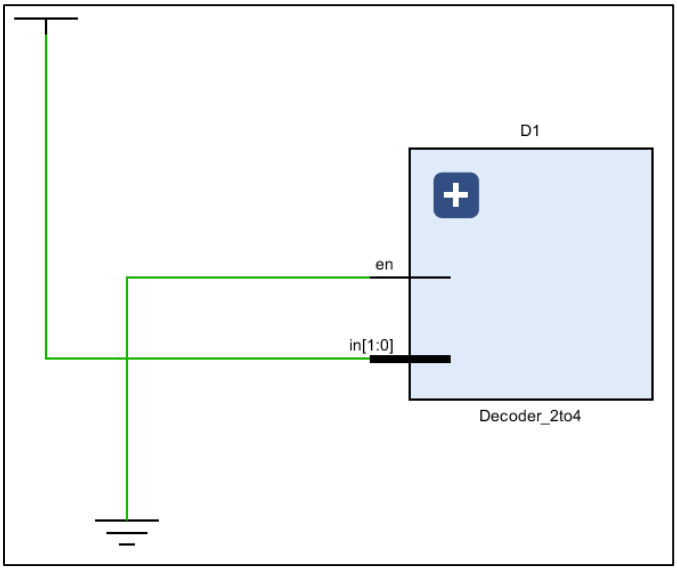
Schematics and Simulated Waveforms: -

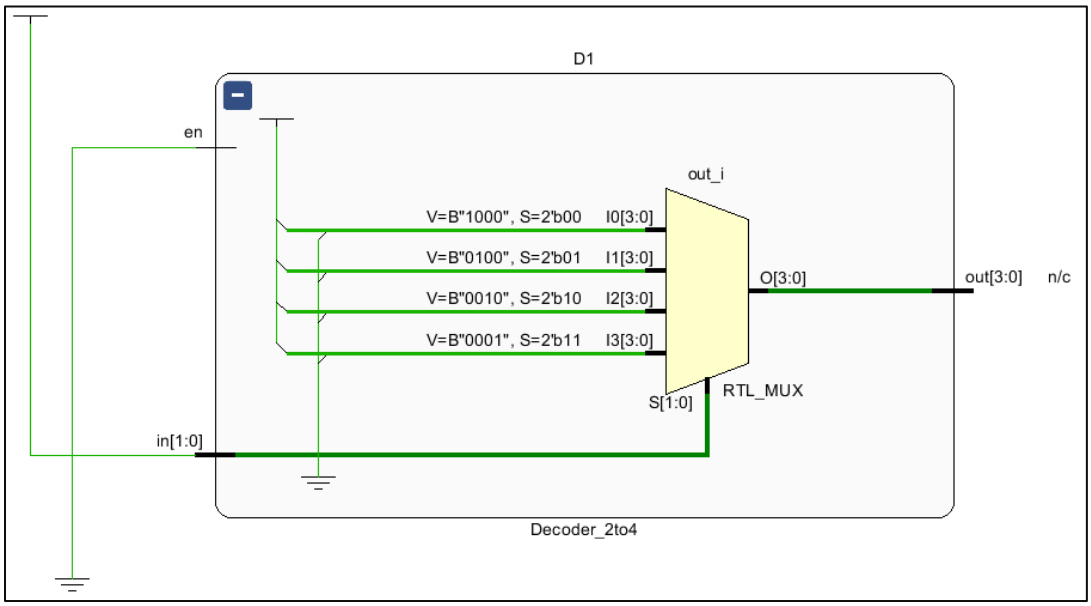
Schematic:

4:2 Encoder



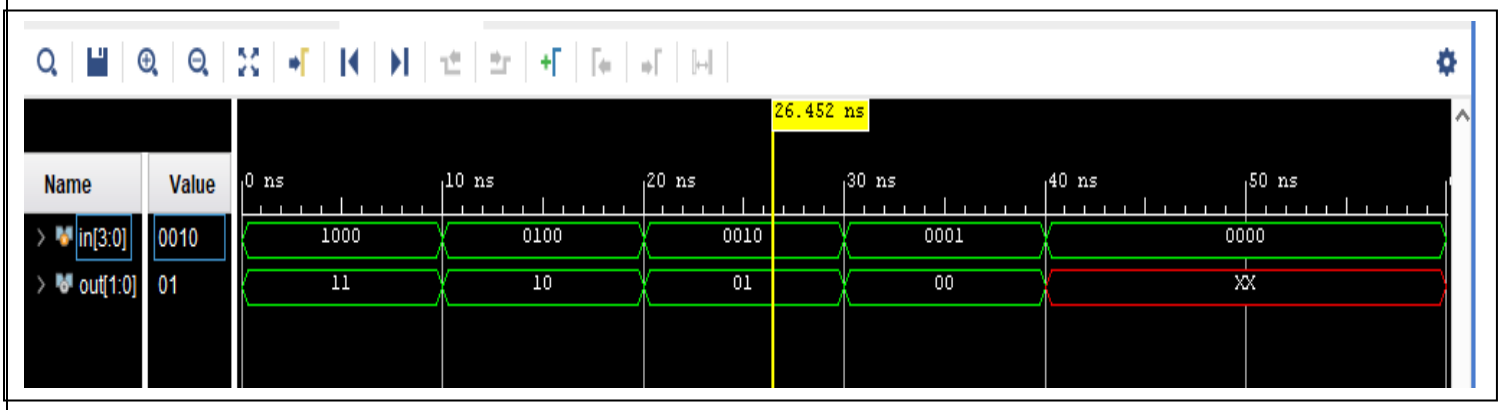
2:4 Decoder





Waveform:

4:2 Encoder



2:4 Decoder

Conclusion: -

Hence , we have successfully designed and simulated 4:2 Encoder/ 2:4 Decoder using behavioral modeling style in Verilog HDL. Also verified simulated waveforms with actual truth tables using Xilinx Vivado 2019.1