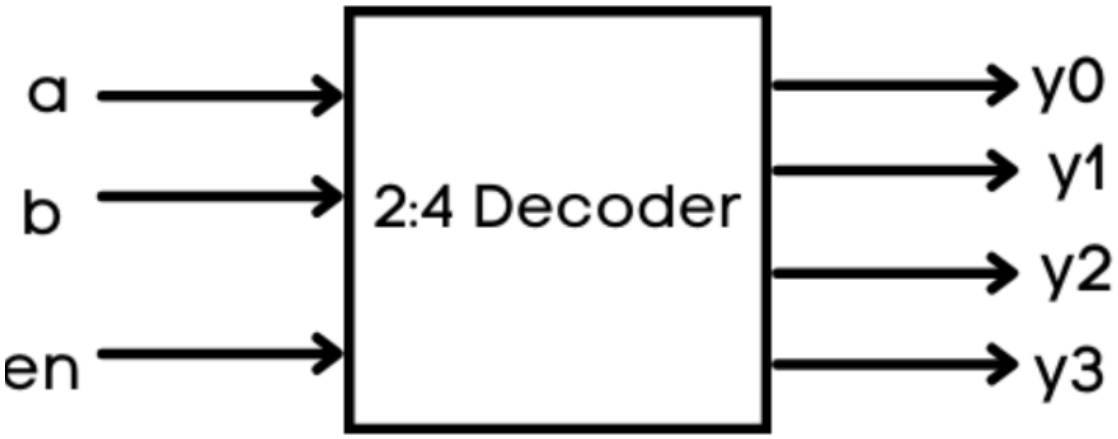
2_to_4_line decoder:

we will implement the 2:4 Decoder using gate level design in Verilog HDL with a step-by-step procedure. Before proceeding to code we shall look into the truth table and logic symbol of the 2:4 Decoder.

A decoder is a combinational logic circuit that has 'n' input signal lines and 2^n output lines. In the 2:4 decoder, we have 2 input lines and 4 output lines. In addition, we provide '*enable*' to the input to ensure the decoder is functioning whenever enable is 1 and it is turned off when enable is 0. The truth table and logic symbol are given below:

Logic Symbol:



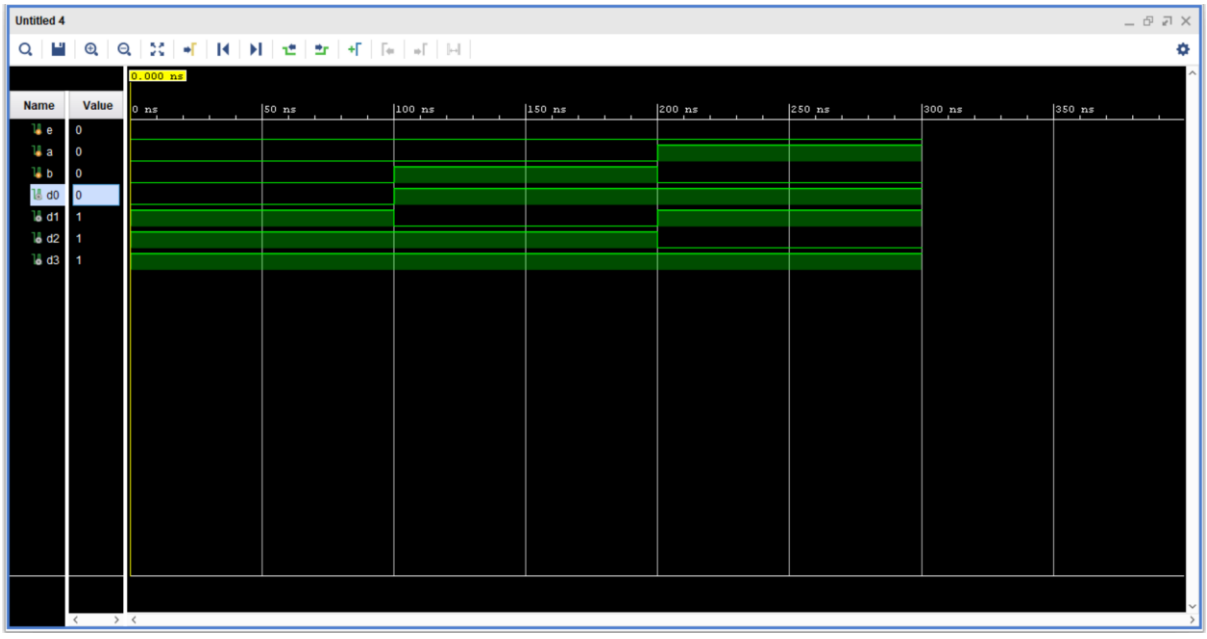
Truth Table:

E	a	b	D0	D1	D2	D3
1	x	x	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

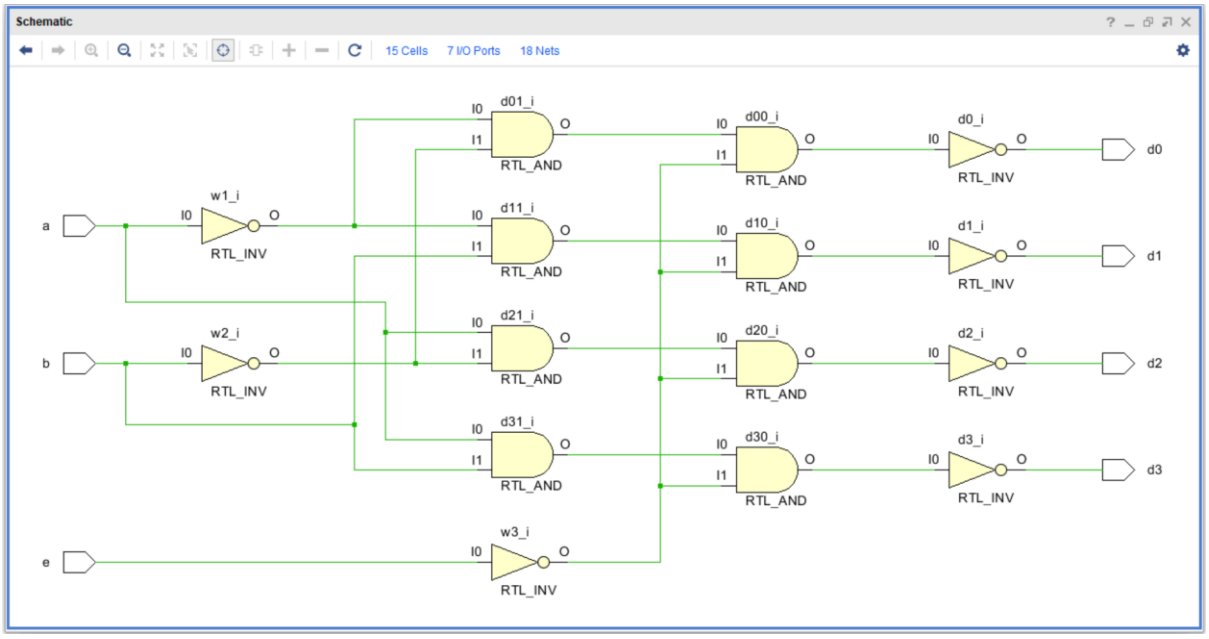
Verilog Code:	**Testbench:**

// the code is for enable=0
module
decoder_2_4(e,a,b,d0,d1,d2,d3);
input e,a,b;
output d0,d1,d2,d3;
wire w1,w2,w3;
not(w1,a);
not(w2,b);
not(w3,e);
nand(d0,w1,w2,w3);
nand(d1,w1,b,w3);
nand(d2,a,w2,w3);
nand(d3,a,b,w3);
endmodule
	// the code is for enable=0
	module decoder_2_4_tb();
	reg e,a,b;
	wire d0,d1,d2,d3;
	decoder_2_4 dec1(e,a,b,d0,d1,d2,d3);
	initial
	begin
	e=0;a=0;b=0;
	#100
	e=0;a=0;b=1;
	#100
	e=0;a=1;b=0;
	#100
	e=0;a=1;b=1;
	$stop;
	end
	initial
	begin
	$display("time e a b d0 d1 d2 d3");
	$monitor($time,"\t",e,"\t",a,"\t",b,"\t",d0,"\t",d1,"\t",d2,"\t",d3);
	end
	endmodule

Simulation:



RTL design:



Tcl Console: