BBM 203: Logic Design Lab

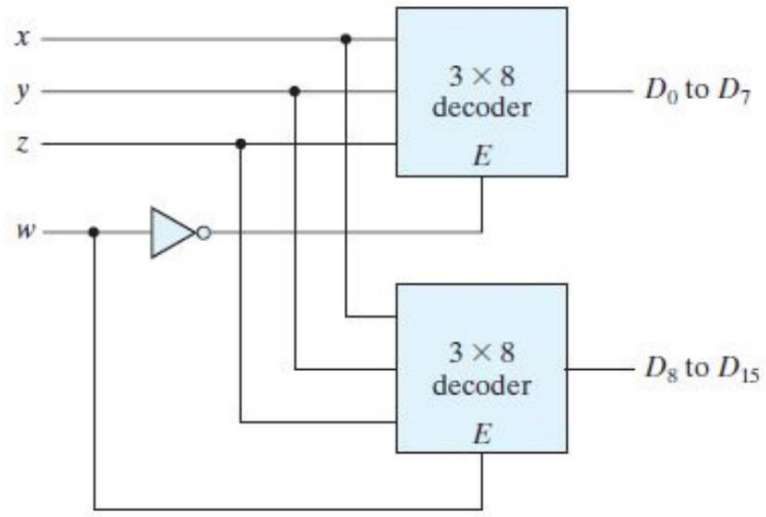
Fall 2020

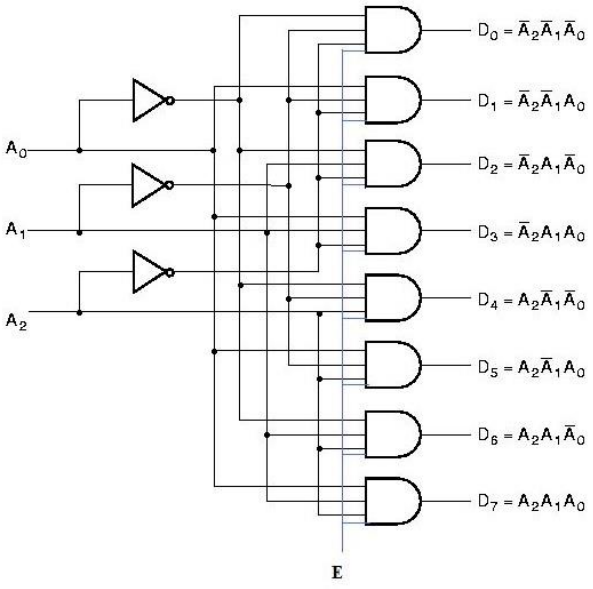
Lab Experiment 4 – Report

Emirhan Topcu 21827899

Problem

Designing and simulating combinational circuits in Verilog HDL.





Verilog Code Solutions

“decoder_3x8.v”

```
`timescale 1ns / 1ps

module decoder_3x8(A,B,C,EN,d0,d1,d2,d3,d4,d5,d6,d7);
input A,B,C,EN;
output d0,d1,d2,d3,d4,d5,d6,d7;
wire d0,d1,d2,d3,d4,d5,d6,d7;
assign d0 = ~(A) & ~(B) & ~(C) & EN;
assign d1 = ~(A) & ~(B) & (C) & EN;
assign d2 = ~(A) & (B) & ~(C) & EN;
assign d3 = ~(A) & (B) & (C) & EN;
assign d4 = (A) & ~(B) & ~(C) & EN;
assign d5 = (A) & ~(B) & (C) & EN;
assign d6 = (A) & (B) & ~(C) & EN;
assign d7 = (A) & (B) & (C) & EN;

endmodule
```

“decoder_3x8_tb.vt”

```
`timescale 1ns / 1ps
`include "decoder_3x8.v"

module decoder_3x8_tb;
	reg A,B,C,EN;
	wire d0,d1,d2,d3,d4,d5,d6,d7;
	decoder_3x8 UUT(A,B,C,EN,d0,d1,d2,d3,d4,d5,d6,d7);
	initial
		begin
		A = 1'b0; B = 1'b0; C = 1'b0; EN = 1'b0;
		#100 A = 1'b0; B = 1'b0; C = 1'b0; EN = 1'b1;
		#100 A = 1'b0; B = 1'b0; C = 1'b1; EN = 1'b1;
		#100 A = 1'b0; B = 1'b1; C = 1'b0; EN = 1'b1;
		#100 A = 1'b0; B = 1'b1; C = 1'b1; EN = 1'b1;
		#100 A = 1'b1; B = 1'b0; C = 1'b0; EN = 1'b1;
		#100 A = 1'b1; B = 1'b0; C = 1'b1; EN = 1'b1;
		#100 A = 1'b1; B = 1'b1; C = 1'b0; EN = 1'b1;
		#100 A = 1'b1; B = 1'b1; C = 1'b1; EN = 1'b1;
		#100 $finish;
		end

endmodule
```

“decoder_4x16.v”

```
`timescale 1ns / 1ps
`include "decoder_3x8.v"

module decoder_4x16(W,X,Y,Z,D0,D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,
D12,D13,D14,D15);
input W,X,Y,Z;
output D0,D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12,D13,D14,D15;
wire D0,D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12,D13,D14,D15;
decoder_3x8 d1(.A(X),.B(Y),.C(Z),.EN(~W),.d0(D0),.d1(D1),.d2(D2),
.d3(D3),.d4(D4),.d5(D5),.d6(D6),.d7(D7));
decoder_3x8 d2(.A(X),.B(Y),.C(Z),.EN(W),.d0(D8),.d1(D9),.d2(D10),
.d3(D11),.d4(D12),.d5(D13),.d6(D14),.d7(D15));
endmodule
```

“decoder_4x16_tb.v”

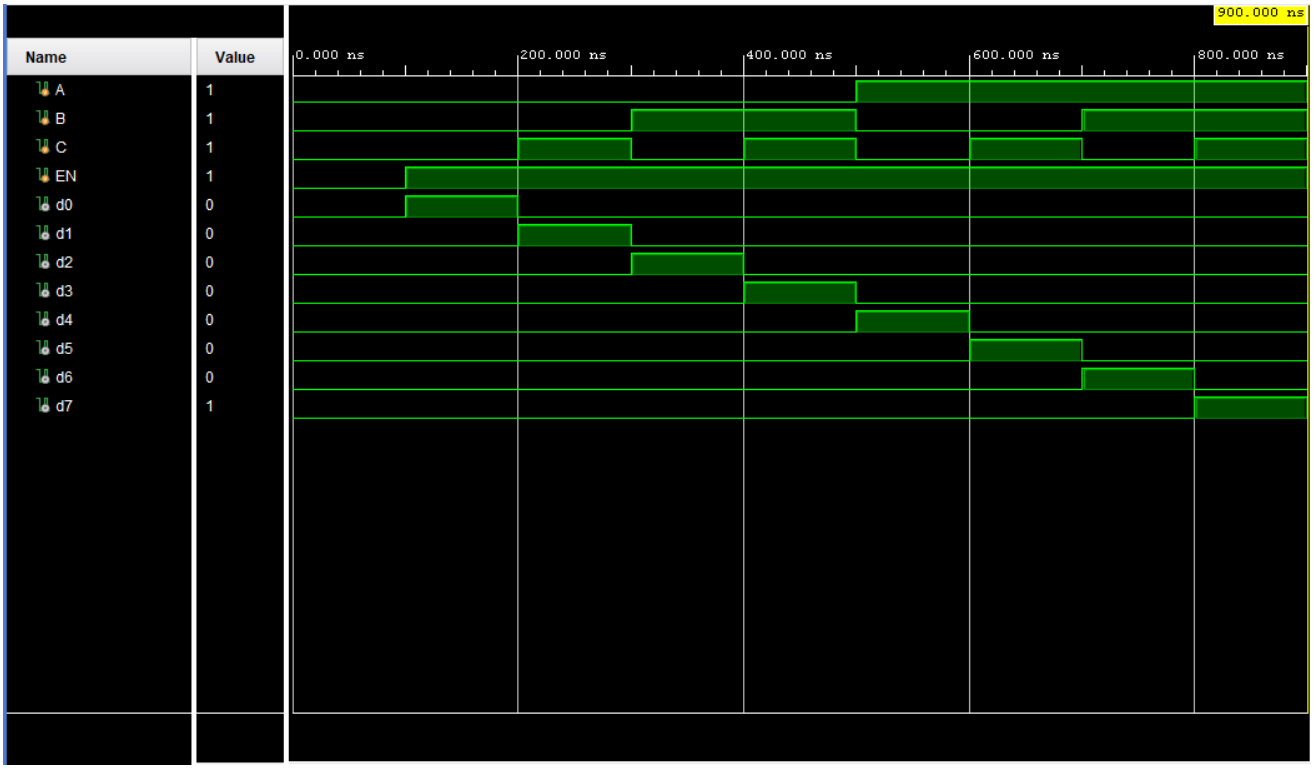
```
`timescale 1ns / 1ps
`include "decoder_4x16.v"

module decoder_4x16_tb;
	reg W,X,Y,Z;
	wire D0,D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12,D13,D14,D15;
	decoder_4x16 UUT2(W,X,Y,Z,D0,D1,D2,D3,D4,D5,D6,D7,D8,
D9,D10,D11,D12,D13,D14,D15);
	initial
		begin
		W = 1'b0; X = 1'b0; Y = 1'b0; Z = 1'b0;
		#100 W = 1'b0; X = 1'b0; Y = 1'b0; Z = 1'b1;
		#100 W = 1'b0; X = 1'b0; Y = 1'b1; Z = 1'b0;
		#100 W = 1'b0; X = 1'b0; Y = 1'b1; Z = 1'b1;
		#100 W = 1'b0; X = 1'b1; Y = 1'b0; Z = 1'b0;
		#100 W = 1'b0; X = 1'b1; Y = 1'b0; Z = 1'b1;
		#100 W = 1'b0; X = 1'b1; Y = 1'b1; Z = 1'b0;
		#100 W = 1'b0; X = 1'b1; Y = 1'b1; Z = 1'b1;
		#100 W = 1'b1; X = 1'b0; Y = 1'b0; Z = 1'b0;
		#100 W = 1'b1; X = 1'b0; Y = 1'b0; Z = 1'b1;
		#100 W = 1'b1; X = 1'b0; Y = 1'b1; Z = 1'b0;
		#100 W = 1'b1; X = 1'b0; Y = 1'b1; Z = 1'b1;
		#100 W = 1'b1; X = 1'b1; Y = 1'b0; Z = 1'b0;
		#100 W = 1'b1; X = 1'b1; Y = 1'b0; Z = 1'b1;
		#100 W = 1'b1; X = 1'b1; Y = 1'b1; Z = 1'b0;
		#100 W = 1'b1; X = 1'b1; Y = 1'b1; Z = 1'b1;
		#100 $finish;
		end

endmodule
```

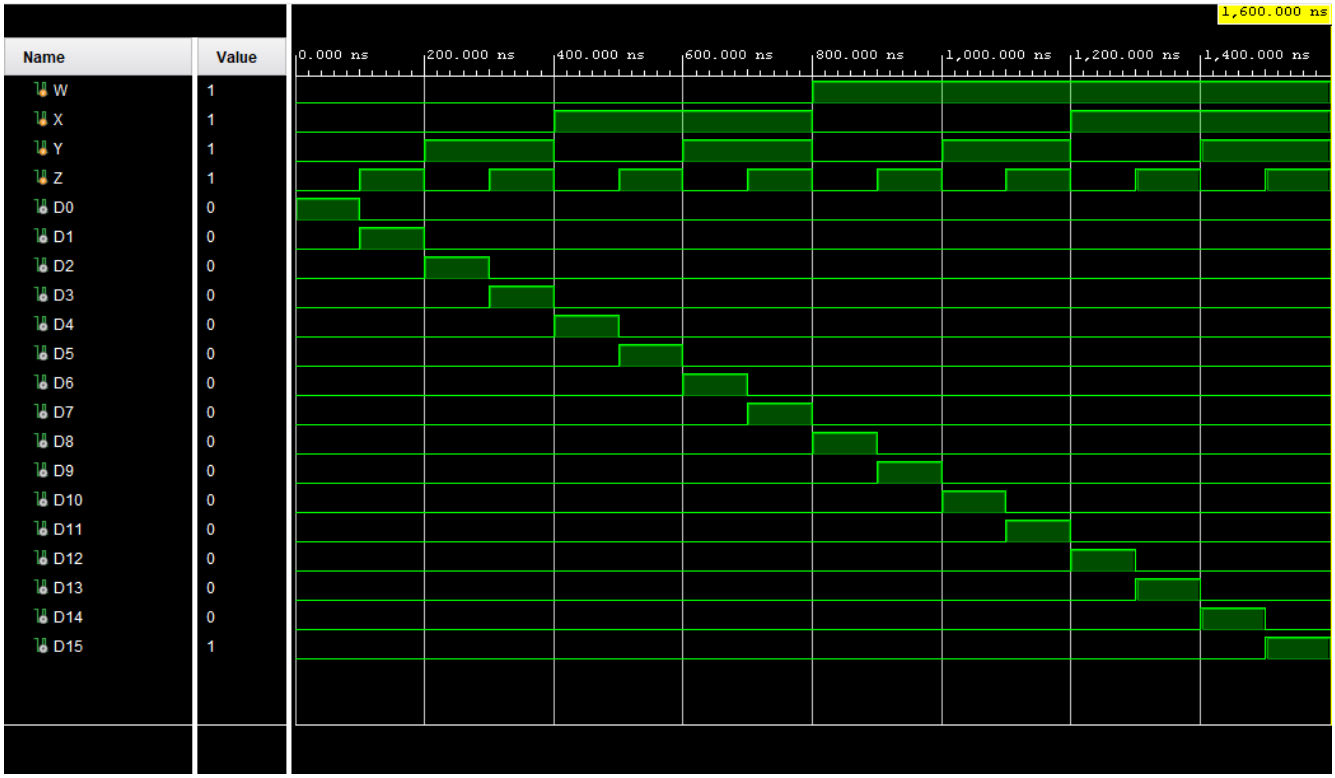
Waveforms

3 to 8 decoder:



This decoder performs in the exact way we want it to. It gives the true outputs for the given inputs. We can see it if we look at the inputs' values on the waveform. When A, B and C are low, output d0 is high. The rest are the same for every binary number until 7.

4 to 16 decoder:



As you can see this decoder works well too. It uses two 3x8 encoders and uses the enable input as the 4th binary number.