DLD LAB WIN 2021

Experiment 5

Aim: To simulate different types of logic gates 16 to 4 Encoder and 4 to 16 Decoder

Tools used: vivado software

Truth tables:

4 to 16 Decoder:

	Inp	uts	20 0		an 1				SE 11		Out	puts	SE 11						
A	В	C	D	Yo	Yı	Y ₂	Y ₃	Y ₄	Y ₅	Y ₆	Y,	Y ₈	Y ₉	Y ₁₀	Y ₁₁	Y ₁₂	Y ₁₃	Y ₁₄	Y ₁₅
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
1	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

16 to 4 Encoder:

115	114	I13	112	111	110	19	18	17	16	15	14	13	12	11	10	D3	D2	D1	D0
1	X	Χ	X	X	X	X	X	X	X	X	X	X	X	X	X	1	1	1	1
0	1	Χ	X	X	X	X	X	X	X	X	X	X	X	X	X	1	1	1	0
0	0	1	X	X	X	X	X	X	X	X	X	X	X	X	X	1	1	0	1
0	0	0	1	X	X	X	X	X	X	X	X	X	X	Χ	X	1	1	0	0
0	0	0	0	1	X	X	X	X	X	X	X	X	X	X	X	1	0	1	1
0	0	0	0	0	1	X	X	X	X	X	X	X	X	X	X	1	0	1	0
0	0	0	0	0	0	1	X	X	X	X	X	X	X	X	X	1	0	0	1
0	0	0	0	0	0	0	1	X	X	Χ	X	X	X	Χ	X	1	0	0	0
0	0	0	0	0	0	0	0	1	X	X	X	X	X	Χ	X	0	1	1	1
0	0	0	0	0	0	0	0	0	1	X	X	X	X	Χ	X	0	1	1	0
0	0	0	0	0	0	0	0	0	0	1	X	X	X	Χ	X	0	1	0	1
0	0	0	0	0	0	0	0	0	0	0	1	X	X	Χ	X	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	X	Χ	X	0	0	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	X	X	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	X	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Codes:

Decoder 4to16:

```
22 module deco 4tol6(i0, i1, i2, i3, i4, i5, i6, i7, i8, i9, i10, i11, i12, i13, i14, i15, z0, z1, z2, z3);
23 input z0, z1, z2, z3;
24 output reg i0, i1, i2, i3, i4, i5, i6, i7, i8, i9, i10, i11, i12, i13, i14, i15;
25 always@(z0,z1,z2,z3)
26 begin
27 if (z0 == 1'b0 && z1 == 1'b0 && z2 == 1'b0 && z3 == 1'b0)
28 begin
    i0 = 1'b0;i1 = 1'b1;i2 = 1'b1;i3 = 1'b1;
     i4 = 1'bl;i5 = 1'bl;i6 = 1'bl;i7 = 1'bl;
    i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
      il2 = 1'bl;il3 = 1'bl;il4 = 1'bl;il5 = 1'bl;
33 end
34 else if (z0 == 1'b0 && z1 == 1'b0 && z2 == 1'b0 && z3 == 1'b1)
    i0 = 1'b1; i1 = 1'b0; i2 = 1'b1; i3 = 1'b1;
     i4 = 1'bl;i5 = 1'bl;i6 = 1'bl;i7 = 1'bl;
     i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
39
    il2 = 1'bl;il3 = 1'bl;il4 = 1'bl;il5 = 1'bl;
41 else if (z0 == 1'b0 && z1 == 1'b0 && z2 == 1'b1 && z3 == 1'b0)
42 begin
      i0 = 1'bl;i1 = 1'bl;i2 = 1'b0;i3 = 1'bl;
43
      i4 = 1'bl;i5 = 1'bl;i6 = 1'bl;i7 = 1'bl;
      i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
46
      i12 = 1'bl;i13 = 1'bl;i14 = 1'bl;i15 = 1'bl;
47 end
48 else if (z0 == 1'b0 && z1 == 1'b0 && z2 == 1'b1 && z3 == 1'b1)
49 begin
      i0 = 1'b1;i1 = 1'b1;i2 = 1'b1;i3 = 1'b0;
      i4 = 1'bl;i5 = 1'bl;i6 = 1'bl;i7 = 1'bl;
      i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
      il2 = 1'bl;il3 = 1'bl;il4 = 1'bl;il5 = 1'bl;
54 end
55 else if (z0 == 1'b0 && z1 == 1'b1 && z2 == 1'b0 && z3 == 1'b0)
      i0 = 1'b1;i1 = 1'b1;i2 = 1'b0;i3 = 1'b1;
      i4 = 1'b0; i5 = 1'b1; i6 = 1'b1; i7 = 1'b1;
      i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
60
      i12 = 1'bl;i13 = 1'bl;i14 = 1'bl;i15 = 1'bl;
62 else if (z0 == 1'b0 && z1 == 1'b1 && z2 == 1'b0 && z3 == 1'b1)
63 begin
```

```
64
       i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
 65
      i4 = 1'b1; i5 = 1'b0; i6 = 1'b1; i7 = 1'b1;
 66
       i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
       i12 = 1'bl;i13 = 1'bl;i14 = 1'bl;i15 = 1'bl;
 68 end
 69 else if(z0 == 1'b0 && z1 == 1'b1 && z2 == 1'b1 && z3 == 1'b0)
 70 begin
       i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
 72
       i4 = 1'b1; i5 = 1'b1; i6 = 1'b0; i7 = 1'b1;
 73
      i8 = 1'b1;i9 = 1'b1;i10 = 1'b1;i11 = 1'b1;
 74
       i12 = 1'bl;i13 = 1'bl;i14 = 1'bl;i15 = 1'bl;
 75 end
 76 else if (z0 == 1'b0 && z1 == 1'b1 && z2 == 1'b1 && z3 == 1'b1)
 77 begin
 78
     i0 = 1'bl;i1 = 1'bl;i2 = 1'bl;i3 = 1'bl;
 79
       i4 = 1'b1; i5 = 1'b1; i6 = 1'b1; i7 = 1'b0;
 80
      i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
       il2 = l'bl;il3 = l'bl;il4 = l'bl;il5 = l'bl;
 81
 82 end
 83 else if (z0 == 1'bl && z1 == 1'b0 && z2 == 1'b0 && z3 == 1'b0)
 84 begin
 85
       i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
 86
      i4 = 1'b1; i5 = 1'b1; i6 = 11'b1; i7 = 1'b1;
       i8 = 1'b0;i9 = 1'b1;i10 = 1'b1;i11 = 1'b1;
 88
       i12 = 1'bl;i13 = 1'bl;i14 = 1'bl;i15 = 1'bl;
 89 end
 90 else if (z0 == 1'bl && zl == 1'b0 && z2 == 1'b0 && z3 == 1'b1)
 91 begin
       i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
     i4 = 1'bl;i5 = 1'bl;i6 = 11'bl;i7 = 1'bl;
      i8 = 1'b1;i9 = 1'b0;i10 = 1'b1;i11 = 1'b1;
 94
 95
       i12 = 1'bl;i13 = 1'bl;i14 = 1'bl;i15 = 1'bl;
 97 else if(z0 == 1'bl && z1 == 1'b0 && z2 == 1'bl && z3 == 1'b0)
 98 begin
 99
       i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
100
       i4 = 1'bl;i5 = 1'bl;i6 = 11'bl;i7 = 1'bl;
101
      i8 = 1'b1;i9 = 1'b1;i10 = 1'b0;i11 = 1'b1;
102
       il2 = 1'bl;il3 = 1'bl;il4 = 1'bl;il5 = 1'bl;
103 end
104 else if (z0 == 1'bl && zl == 1'b0 && z2 == 1'bl && z3 == 1'bl)
105 begin
106 i0 = l'bl;i1 = l'bl;i2 = l'bl;i3 = l'bl;
107
     i4 = 1'b1;i5 = 1'b1;i6 = 11'b1;i7 = 1'b1;
```

```
108
      i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'b0;
109
       il2 = 1'bl;il3 = 1'bl;il4 = 1'bl;il5 = 1'bl;
110 end
111 else if(z0 == 1'bl && z1 == 1'bl && z2 == 1'b0 && z3 == 1'b0)
112 begin
113
       i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
114
       i4 = 1'b1; i5 = 1'b1; i6 = 11'b1; i7 = 1'b1;
115
      i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
116
      i12 = 1'b0;i13 = 1'b1;i14 = 1'b1;i15 = 1'b1;
117 end
118 else if (z0 == 1'b1 & & z1 == 1'b1 & & z2 == 1'b0 & & z3 == 1'b1)
119 begin
120
      i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
121
      i4 = 1'b1; i5 = 1'b1; i6 = 11'b1; i7 = 1'b1;
122
      i8 = 1'b1; i9 = 1'b1; i10 = 1'b1; i11 = 1'b1;
123
      i12 = 1'b1; i13 = 1'b0; i14 = 1'b1; i15 = 1'b1;
124 end
125 else if (z0 == 1'bl && z1 == 1'bl && z2 == 1'bl && z3 == 1'b0)
126 begin
127
      i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
128
      i4 = 1'b1; i5 = 1'b1; i6 = 11'b1; i7 = 1'b1;
129
      i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
130
       il2 = 1'bl;il3 = 1'bl;il4 = 1'b0;il5 = 1'bl;
131 end
132 else
133 begin
134
      i0 = 1'b1; i1 = 1'b1; i2 = 1'b1; i3 = 1'b1;
135
      i4 = 1'b1; i5 = 1'b1; i6 = 11'b1; i7 = 1'b1;
136
      i8 = 1'bl;i9 = 1'bl;i10 = 1'bl;i11 = 1'bl;
137
      i12 = 1'b1;i13 = 1'b1;i14 = 1'b1;i15 = 1'b0;
138 end
139 end
140 endmodule
```

Test bench code:

```
module tb deco4tol6;
reg z0,z1,z2,z3;
i wire i0, i1, i2, i3, i4, i5, i6, i7, i8, i9, i10, i11, i12, i13, i14, i15;
ideco_4tol6 m3(i0, i1, i2, i3, i4, i5, i6, i7, i8, i9, i10, i11, i12, i13, i14, i15, z0, z1, z2, z3);
initial
begin
     z0 = 1'b0; z1 = 1'b0; z2 = 1'b0; z3 = 1'b0;
) #10 z0 = 1'b0; z1 = 1'b0; z2 = 1'b0; z3 = 1'b1;
#10 z0 = 1'b0; z1 = 1'b0; z2 = 1'b1; z3 = 1'b0;
#10 z0 = 1'b0;z1 = 1'b0;z2 = 1'b1;z3 = 1'b1;
1 #10 z0 = 1'b0; z1 = 1'b1; z2 = 1'b0; z3 = 1'b0;
#10 z0 = 1'b0;z1 = 1'b1;z2 = 1'b0;z3 = 1'b1;
i #10 z0 = 1'b0;z1 = 1'b1;z2 = 1'b1;z3 = 1'b0;
#10 z0 = 1'b0;z1 = 1'b1;z2 = 1'b1;z3 = 1'b0;
1 #10 z0 = 1'b0;z1 = 1'b1;z2 = 1'b1;z3 = 1'b1;
1 #10 z0 = 1'b1; z1 = 1'b0; z2 = 1'b0; z3 = 1'b0;
\#10 z0 = 1'b1; z1 = 1'b0; z2 = 1'b0; z3 = 1'b1;
) #10 z0 = 1'b1;z1 = 1'b0;z2 = 1'b1;z3 = 1'b0;
| #10 z0 = 1'b1;z1 = 1'b0;z2 = 1'b1;z3 = 1'b1;
#10 z0 = 1'b1;z1 = 1'b1;z2 = 1'b0;z3 = 1'b0;
#10 z0 = 1'b1;z1 = 1'b1;z2 = 1'b0;z3 = 1'b1;
#10 z0 = 1'bl;z1 = 1'bl;z2 = 1'bl;z3 = 1'b0;
i #10 z0 = 1'b1;z1 = 1'b1;z2 = 1'b1;z3 = 1'b1;
end
endmodule
```

16 to 4 Encoder:

Code:

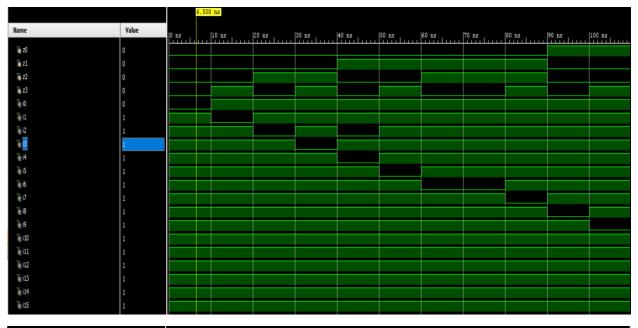
```
module Encoder16to4(i0,i1,i2,i3,i4,i5,i6,i7,i8,i9,i10,i11,i12,i13,i14,i15,z0,z1,z2,z3);
input i0,i1,i2,i3,i4,i5,i6,i7,i8,i9,i10,i11,i12,i13,i14,i15;
output z0,z1,z2,z3;
orgate al(i1,i3,i5,i7,i9,i11,i13,i15,z0);
orgate a2(i2,i3,i6,i7,i10,i11,i14,i15,z1);
orgate a3(i4,i5,i6,i7,i12,i13,i14,i15,z2);
orgate a4(i8,i9,i10,i11,i12,i13,i14,i15,z3);
endmodule
module orgate(a,b,c,d,e,f,g,h,z);
input a,b,c,d,e,f,g,h;
output z;
assign z = a | b | c | d | e | f | g | h;
endmodule
```

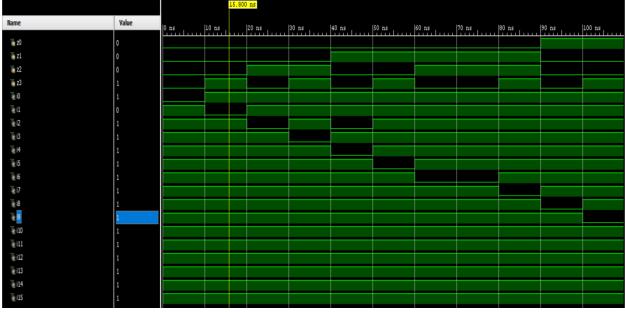
Test bench code:

```
module tb Encoder16to4;
reg i0,i1,i2,i3,i4,i5,i6,i7,i8,i9,i10,i11,i12,i13,i14,i15;
wire z0, z1, z2, z3;
Encoder16to4 m2(i0,i1,i2,i3,i4,i5,i6,i7,i8,i9,i10,i11,i12,i13,i14,i15,z0,z1,z2,z3);
initial
begin
          i0 = 1'bl;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 1'b0
#10 i0 = 1'b0;i1 = 1'b1;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b1;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b1;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b1;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b1;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b1;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b1;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b1;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 1
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b1;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b1;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b1;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 1
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b1;i13 = 1'b0;i14 = 1'b0;i15 = 1
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b1;i14 = 1'b0;i15 = 
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b1;i15= 1'b0;
#10 i0 = 1'b0;i1 = 1'b0;i2 = 1'b0;i3 = 1'b0;i4 = 1'b0;i5 = 1'b0;i6 = 1'b0;i6 = 1'b0;i7 = 1'b0;i8 = 1'b0;i9 = 1'b0;i10 = 1'b0;i11 = 1'b0;i12 = 1'b0;i13 = 1'b0;i14 = 1'b0;i15 = 1'b1;
end
  endmodule
```

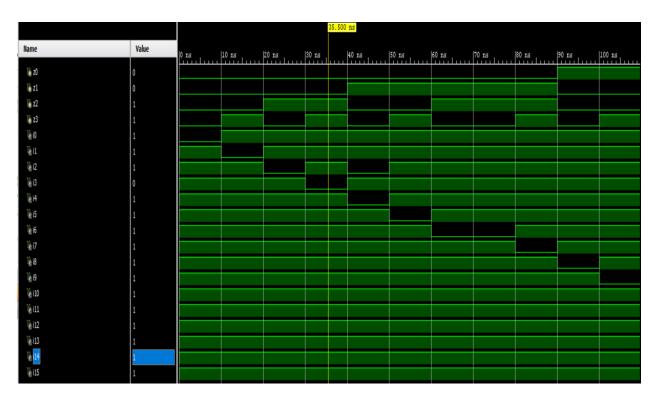
Results:

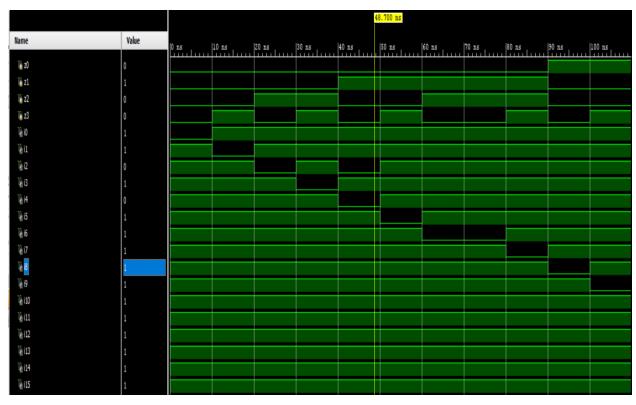
Decoder 4to16:

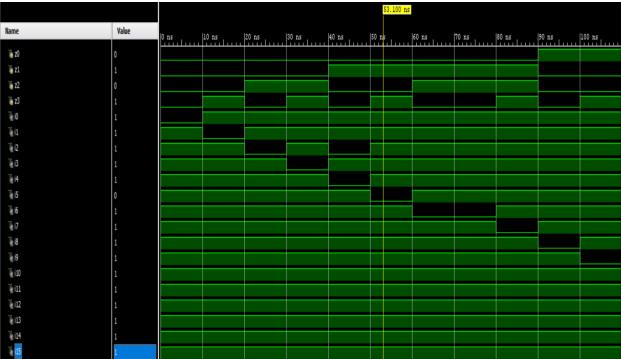




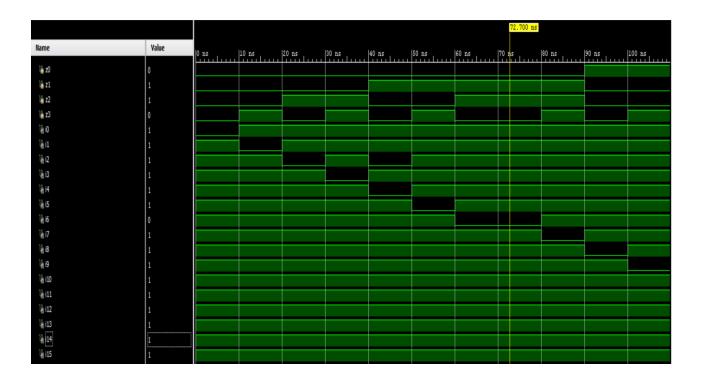


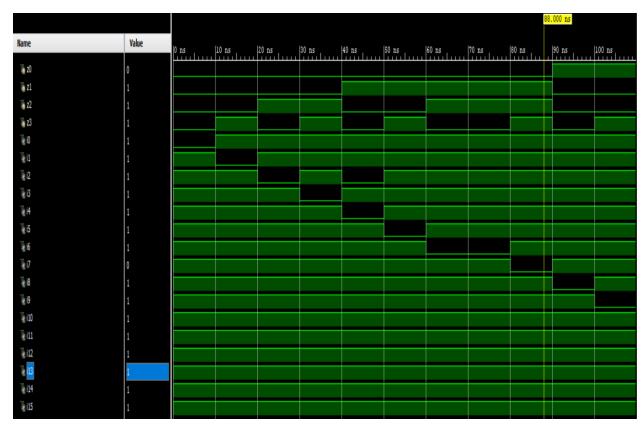


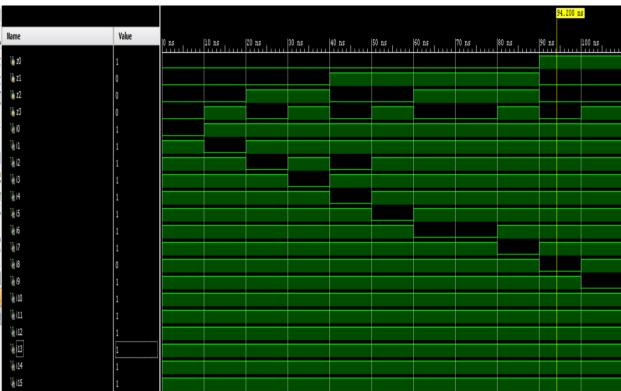




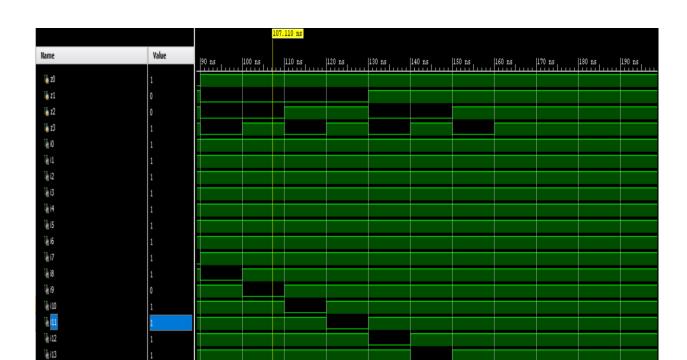
								61.800 ns				
Name	Value	0 ns	10 ns	20 ns	30 ns	40 ns	50 ns	60 ns	70 ns	80 ns	90 ns	100 ns
16 z0	0											
₩ z1	1											
™ z2	1											
™ z3	0											
™ _a io	1											
1 € 11	1											
1 € i2	1											
1 € i3	1											
1 € i4	1											
™ i5	1											
1	0											
16 17	1											
1 € 18	1											
T ₆ i9	1											
™ i10	1											
¼ i11	1											
⅓ i12	1											
¼ i13	1											
T ₀ 114	1											
1 € i15	1											

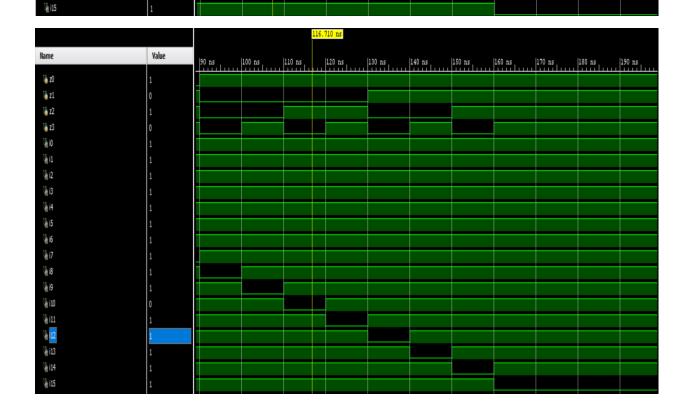


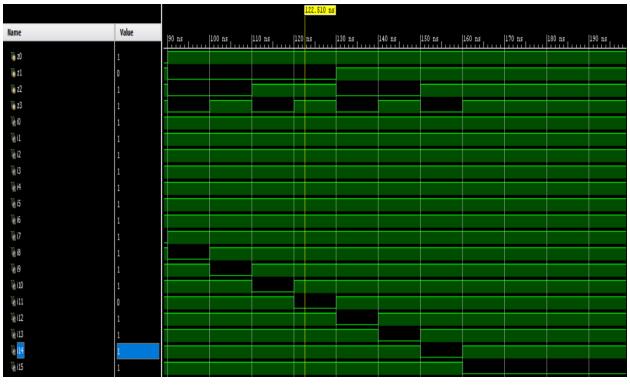


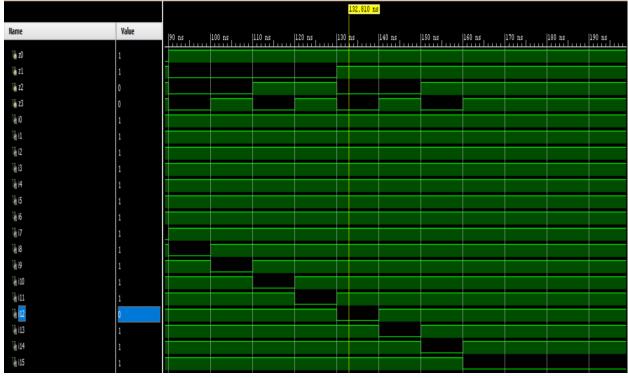


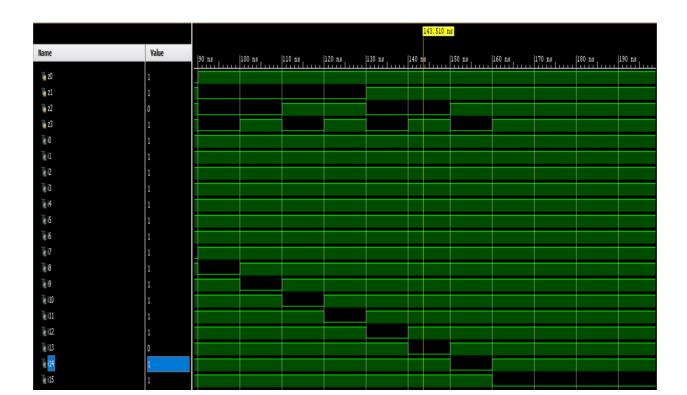
₩ i14

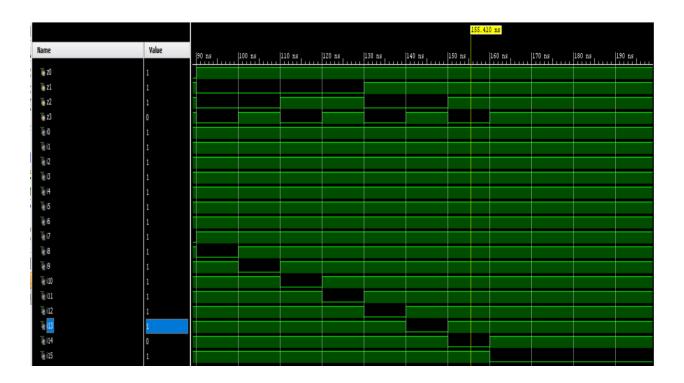


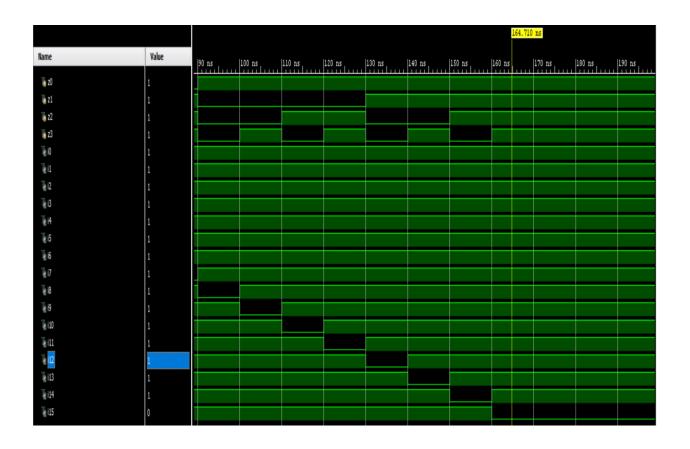












16 to 4 Encoder:



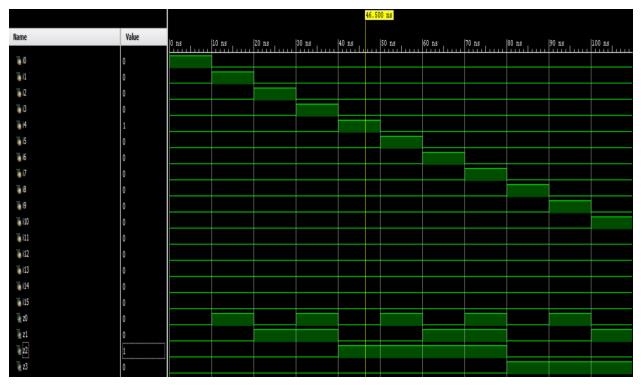












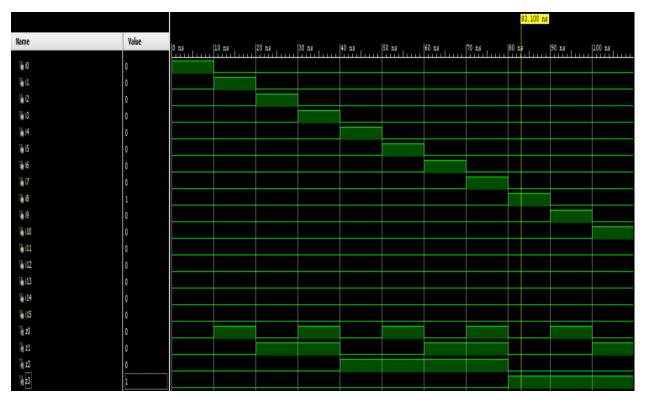


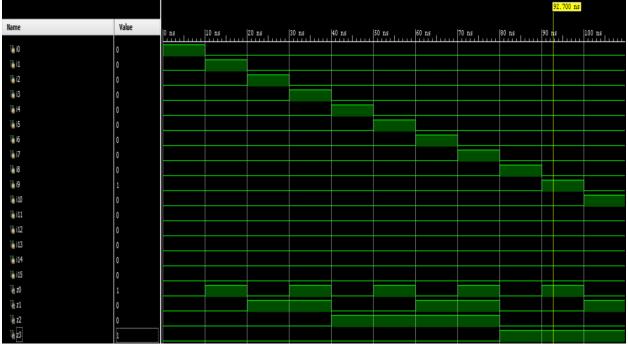




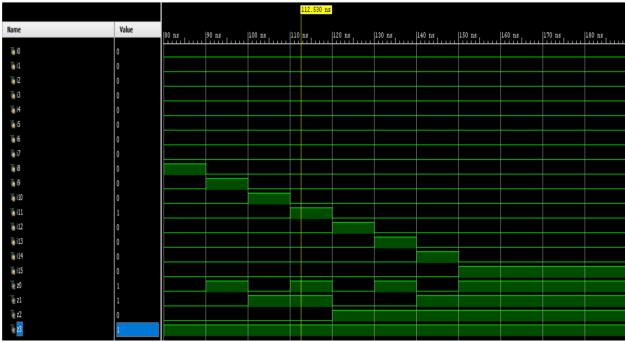




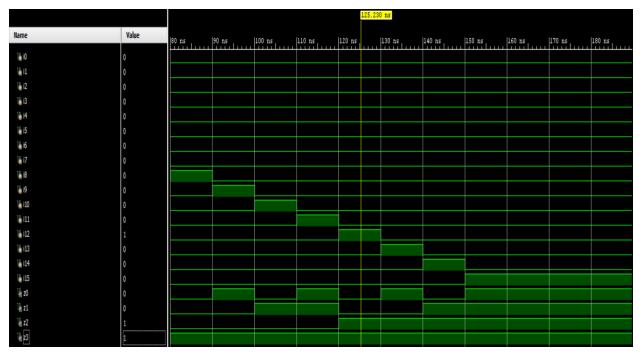






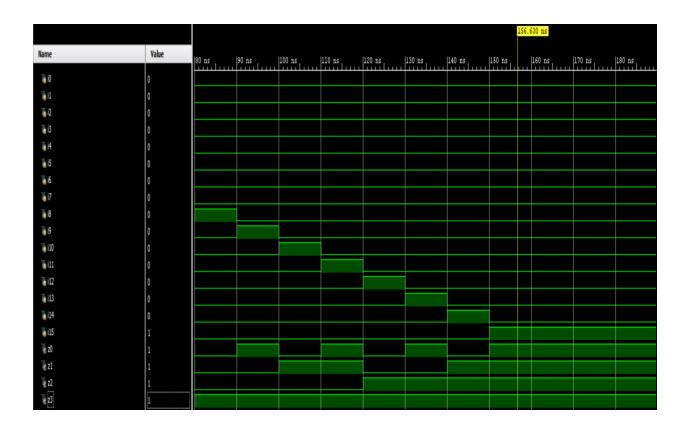








			143.130 ms											
Name	Value	80 ns	90 ns	100 ns	110 ns	120 ns	130 ns	140 ns		150 ns	160 ns	170 ns	180 ns	
1 <u>€</u> i0	0								11111					
1 <u>6</u> i1	0													
1 <u>6</u> i2	0													
1 <u>6</u> i3	0													
16 i4	0													
1 € i5	0													
1 € i6	0													
T <u>a</u> 17	0													
1 € i8	0							Ш						
1 € i9	0							Ш						
1 € i10	0							Ш						
1 <u>6</u> i11	0							Ш						
₩ i12	0							Ш						
1 i13	0							Щ						
16 i14	1													
₩ i15	0							Ш						
¼ z0	0							Ц						
₩ z1	1													
₩ z2	1													
1 22 1 23	1													



Conclusion:

Now I can simulate 16 to 4 Encoder and 4 to 16 Decoder codes