

Assignment 2 Extra
By Amr Mohamed Ahmed Ebrahim

Question 1

Gray Encoder Code

```
1  module gray_encoder (  
2      input [2:0] A,  
3      output reg [6:0] B);  
4  
5      always @(*)  
6          case(A)  
7              0:B=7'b000;  
8              1:B=7'b001;  
9              2:B=7'b011;  
10             3:B=7'b010;  
11             4:B=7'b110;  
12             5:B=7'b111;  
13             6:B=7'b101;  
14             7:B=7'b100;  
15             default:B=7'bx;  
16         endcase  
17 endmodule
```

Hot Encoder

```
20 module hot_encoder (  
21     input [2:0] A,  
22     output reg [6:0] B);  
23  
24     always @(*)  
25         case(A)  
26             0:B=7'b0000000;  
27             1:B=7'b0000001;  
28             2:B=7'b0000010;  
29             3:B=7'b0000100;  
30             4:B=7'b0001000;  
31             5:B=7'b0010000;  
32             6:B=7'b0100000;  
33             7:B=7'b1000000;  
34             default:B=7'bx;  
35         endcase  
36 endmodule
```

Gray Hot Encoder

```
39 ▼ module gray_hot_encoder #(parameter USE_GRAY = 1)
40 ▼ (
41     input [2:0] A,
42     output [6:0] B
43 ▼ );
44
45 ▼ generate
46 ▼     if(USE_GRAY)
47         gray_encoder gray_encoder (A,B) ;
48 ▼     else
49         hot_encoder hot_encoder (A,B) ;
50     endgenerate
51 endmodule
```

Gray Encoder Testbench

```
53 `timescale 1ns/1ps
54 ▼ module gray_encoder_tb ();
55
56     reg [2:0] A;
57     wire [6:0] B_dut ;
58     reg [6:0] B_exp ;
59
60     gray_hot_encoder #(.USE_GRAY(1)) dut (A,B_dut);
61
62 ▼ initial begin
63     #0 A=0; B_exp=7'b000;
64     #10 A=1; B_exp=7'b001;
65     #10 A=2; B_exp=7'b011;
66     #10 A=3; B_exp=7'b010;
67     #10 A=4; B_exp=7'b110;
68     #10 A=5; B_exp=7'b111;
69     #10 A=6; B_exp=7'b101;
70     #10 A=7; B_exp=7'b100;
71     #10 $stop;
72 end
73
74 ▼ initial
75     $monitor ("A=%b, B_dut=%b, B_exp=%b", A, B_dut, B_exp) ;
76
77 endmodule
```

Msgs									
A	111	000	001	010	011	100	101	110	111
B_dut	0000100	0000000	0000001	0000011	0000010	0000110	0000111	0000101	0000100
B_exp	0000100	0000000	0000001	0000011	0000010	0000110	0000111	0000101	0000100

```

81 module hot_encoder_tb ();
82
83     reg [2:0] A;
84     wire [6:0] B_dut ;
85     reg [6:0] B_exp ;
86
87     gray_hot_encoder #(.USE_GRAY(0)) dut (A,B_dut);
88
89     initial begin
90         #0 A=0; B_exp=7'b0000000;
91         #10 A=1; B_exp=7'b0000001;
92         #10 A=2; B_exp=7'b0000010;
93         #10 A=3; B_exp=7'b0000100;
94         #10 A=4; B_exp=7'b0001000;
95         #10 A=5; B_exp=7'b0010000;
96         #10 A=6; B_exp=7'b0100000;
97         #10 A=7; B_exp=7'b1000000;
98         #10 $stop;
99     end
100
101     initial
102         $monitor ("A=%b, B_dut=%b, B_exp=%b", A, B_dut, B_exp) ;
103
104 endmodule

```

Msgs													
A	111	000	001	010	011	100	101	110	111				
B_dut	1000000	0000000	0000001	0000010	0000100	0001000	0010000	0100000	1000000				
B_exp	1000000	0000000	0000001	0000010	0000100	0001000	0010000	0100000	1000000				

Question 2

DUT Code

```
1 ▼ module demux_1x4_dut (  
2     input D,  
3     input [1:0] S,  
4 ▼   output reg [3:0] Y);  
5  
6     always @(*)  
7 ▼   begin  
8         Y=4'b0000;  
9 ▼     case(S)  
10        0: Y[0]=D;  
11        1: Y[1]=D;  
12        2: Y[2]=D;  
13        3: Y[3]=D;  
14    endcase  
15    end  
16  
17 endmodule
```

REF Code

```
20 ▼ module demux_1x4_ref (  
21     input D,  
22     input [1:0] S,  
23 ▼   output reg [3:0] Y);  
24  
25 ▼   always @(*)  
26 ▼       if(S==0)  
27           Y={3'b0,D} ;  
28 ▼       else if(S==1)  
29           Y={2'b0,D,1'b0} ;  
30 ▼       else if(S==2)  
31           Y={1'b0,D,2'b0} ;  
32 ▼       else  
33           Y={D,3'b0} ;  
34  
35 endmodule
```

TestBench Code

```
38 module demux_1x4_tb ();
39
40     reg D;
41     reg [1:0] S ;
42     wire [3:0] Y_dut, Y_ref ;
43
44     demux_1x4_dut dut (D, S, Y_dut) ;
45     demux_1x4_ref ref (D, S, Y_ref) ;
46
47     initial begin
48         repeat(100) begin
49             D = $random ;
50             S = $random ;
51             #10;
52             if(Y_dut != Y_ref) begin
53                 $display ("Fail") ;
54                 $stop ;
55             end
56         end
57         $display("Pass") ;
58         $stop ;
59     end
60
61     initial begin
62         $monitor("D=%b, S=%b, Y_dut=%b, Y_ref=%b",D,S,Y_dut,Y_ref) ;
63     end
64
65 endmodule
```

TestBench Wave

	Msgs	
D	1	
S	01	01 11 01 10 01 00 10 11 10 01
Y_dut	0010	0000 1000 0010 0100 0010 0000 0001 0100 1000 0000
Y_ref	0010	0000 1000 0010 0100 0010 0000 0001 0100 1000 0000