Digital Design Diploma

Assignment 2 [Extra]

Combinational Circuit Design

Name	كريم حسن عاطف علي
Group	G2

Submitted to: Eng. Kareem Waseem

1) Gray One-Hot Encoder:

```
Gray_One_Hot_Encoder.v > ...
1   module Gray_One_Hot_Encoder (input [2:0] A, output reg [6:0] B);
2   parameter USE_GRAY= 1;   // Set to 1 for Gray encoding, 0 for One-hot encoding
3
4   generate
5       if (USE_GRAY) begin  // Binary-Gray encoding logic
6           always @(*) begin
7               case (A)
8                   3'b000:  B = 7'b0000000; // 0 in Gray
9                   3'b001:  B = 7'b0000001; // 1 in Gray
10                  3'b010:  B = 7'b0000011; // 2 in Gray
11                  3'b011:  B = 7'b0000010; // 3 in Gray
12                  3'b100:  B = 7'b0000110; // 4 in Gray
13                  3'b101:  B = 7'b0000111; // 5 in Gray
14                  3'b110:  B = 7'b0000101; // 6 in Gray
15                  3'b111:  B = 7'b0000100; // 7 in Gray
16                  default: B = 7'b0000000; // Default case
17              endcase
18          end
19      end
20      else begin       // Binary one-hot encoding logic
21          always @(*) begin
22              case (A)
23                  3'b000:  B = 7'b0000000; // 0 in One-hot
24                  3'b001:  B = 7'b0000001; // 1 in One-hot
25                  3'b010:  B = 7'b0000010; // 2 in One-hot
26                  3'b011:  B = 7'b0000100; // 3 in One-hot
27                  3'b100:  B = 7'b0001000; // 4 in One-hot
28                  3'b101:  B = 7'b0010000; // 5 in One-hot
29                  3'b110:  B = 7'b0100000; // 6 in One-hot
30                  3'b111:  B = 7'b1000000; // 7 in One-hot
31                  default: B = 7'b0000000; // Default case
32              endcase
33          end
34      end
35  endgenerate
36  endmodule
```

Figure 1: Q1 Code

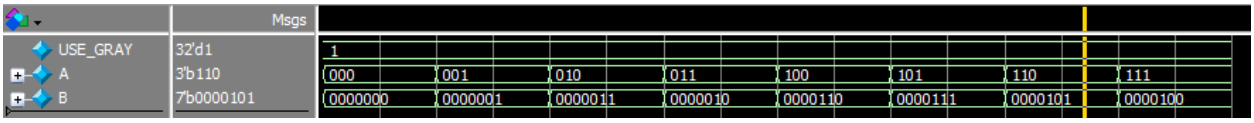

Figure 2: Q1 Wave (Gray)

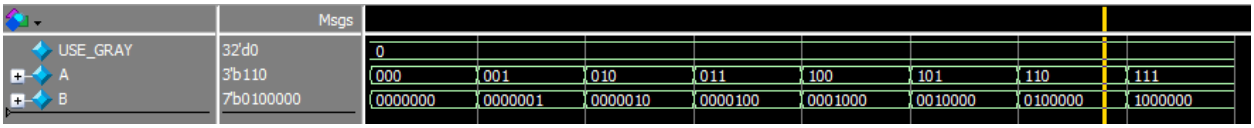

Figure 3: Q1 Wave (One-Hot)

Gray_One_Hot_Encoder_tb_1.v > ...

```v
module Gray_One_Hot_Encoder_tb_1 ();
reg [2:0] A;
wire [6:0] B;
parameter USE_GRAY= 1; // Set to 1 for Gray encoding, 0 for One-hot encoding

//module Gray_One_Hot_Encoder (input [2:0] A, output [6:0] B);
Gray_One_Hot_Encoder #(USE_GRAY) DUT(A,B);

integer i;
initial begin
    for (i = 0; i < 8; i = i + 1) begin
        A = i;
        #10;
    end
    $stop;
end
endmodule
```

Figure 4: Q1 Testbench (Gray)

Gray_One_Hot_Encoder_tb_2.v > ...

```v
module Gray_One_Hot_Encoder_tb_2 ();
reg [2:0] A;
wire [6:0] B;
parameter USE_GRAY= 0; // Set to 1 for Gray encoding, 0 for One-hot encoding

//module Gray_One_Hot_Encoder (input [2:0] A, output [6:0] B);
Gray_One_Hot_Encoder #(USE_GRAY) DUT(A,B);

integer i;
initial begin
    for (i = 0; i < 8; i = i + 1) begin
        A = i;
        #10;
    end
    $stop;
end
endmodule
```

Figure 5: Q1 Testbench (One-Hot)

2) DEMUX:

```v
DEMUX.v > ...
1   module DEMUX(input D, input [1:0] S, output reg [3:0] Y);
2   always @(*) begin
3       case (S)
4           2'b00:      Y = {3'b000, D};
5           2'b01:      Y = {2'b00, D, 1'b0};
6           2'b10:      Y = {1'b0, D, 2'b00};
7           2'b11:      Y = {D, 3'b000};
8           default:    Y = 4'b0000;
9       endcase
10  end
11  endmodule
```

Figure 6: Q2 Code

```v
DEMUX_tb.v > ...
1   module DEMUX_tb ();
2   reg D;
3   reg [1:0] S;
4   wire [3:0] Y;
5
6   //module DEMUX (input D, input [1:0] S, output reg [3:0] Y);
7   DEMUX DUT(D,S,Y);
8
9   integer i,j;
10  initial begin
11      for (i = 0; i < 2; i = i + 1) begin
12          D = i;
13          for (j = 0; j < 4; j = j + 1) begin
14              S = j;
15              #10;
16          end
17      end
18      $stop;
19  end
20  endmodule
```

Figure 7: Q2 Testbench

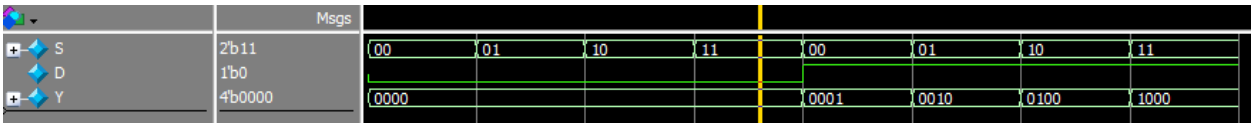

Figure 8: Q2 Wave