Assignment 1

Ameer Louly

July 2, 2025

Question 1

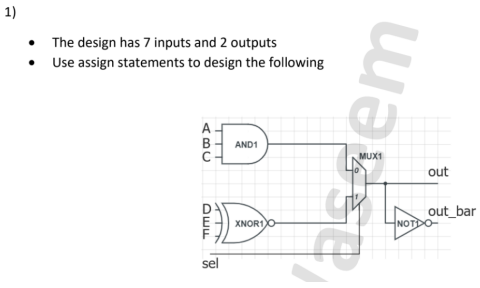

Figure 1: Question

Solution:

```
1  module Q1(A, B, C, D, E, F, sel, out, out_bar);
2
3      input A, B, C, D, E, F, sel;
4      output out, out_bar;
5      wire X1, X2;
6
7      and(X1, A, B, C);
8      xnor(X2, D, E, F);
9      assign out = (sel == 1)? X2 : X1;
10     assign out_bar = ~out;
11
12 endmodule
```

Figure 2: Q1 Code

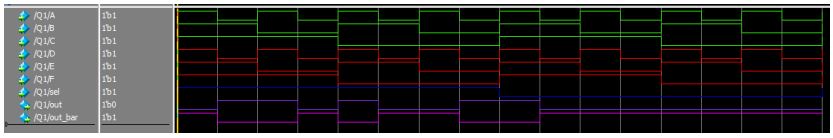

Figure 3: Q1 Waveform

Question 2

- The design has 5 inputs and 2 outputs
  - Inputs
    - i. D -> width = 3
    - ii. A, B, C, Sel -> width = 1
  - Outputs
    - i. Out, out_bar -> width = 1
- Use Behavioral coding style to implement the following

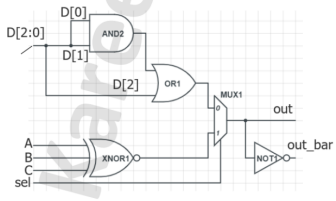

Figure 4: Question

Solution:

```
1  module Q2(D, A, B, C, sel, out, out_bar);
2
3      input [2:0] D;
4      input A, B, C, sel;
5      output out, out_bar;
6      wire AND2, OR1, XNOR1;
7
8      and(AND2, D[0], D[1]);
9      or(OR1, AND2, D[2]);
10     xnor(XNOR1, A, B, C);
11
12     assign out = (sel == 1) ? XNOR1 : OR1;
13     assign out_bar = ~out_bar;
14
15 endmodule
```

Figure 5: Q2 Code

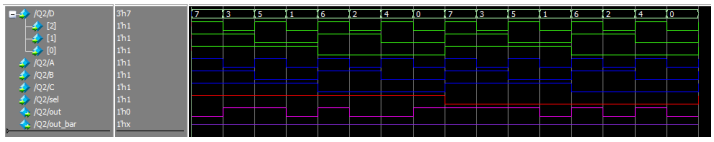

Figure 6: Q2 Waveform

Question 3

Question: Design a 4-bit adder

Solution:

```
1  module Q2(D, A, B, C, sel, out, out_bar);
2
3      input [2:0] D;
4      input A, B, C, sel;
5      output out, out_bar;
6      wire AND2, OR1, XNOR1;
7
8      and(AND2, D[0], D[1]);
9      or(OR1, AND2, D[2]);
10     xnor(XNOR1, A, B, C);
11
12     assign out = (sel == 1) ? XNOR1 : OR1;
13     assign out_bar = ~out_bar;
14
15 endmodule
```

Figure 7: Q3 Code

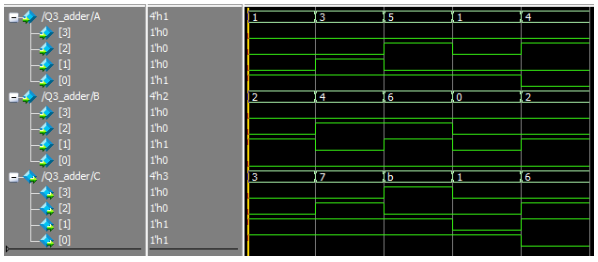

Figure 8: Q3 Waveform

Question 4

4) Implement 2-to-4 Decoder using conditional operator (A logic decoder has n input lines and 2^n output lines. Each output line corresponds to a unique combination of the input values.)

- The design has input **A** (2 bits) and output **D** (4 bits)
- you can use the following format for the conditional operator.
- assign <output_signal> = <condition1> ? <value1> : <condition2> ? <value2> : <default_value>);

A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Figure 9: Question 4

Solution:

```
1  module Q4_decoder(A, D);
2
3      input [1:0] A;
4      output [3:0] D;
5
6      assign D = (A == 2'b00) ? 4'b0001 :
7                 (A == 2'b01) ? 4'b0010 :
8                 (A == 2'b10) ? 4'b0100 : 4'b1000;
9
10 endmodule
```

Figure 10: Q4 Code

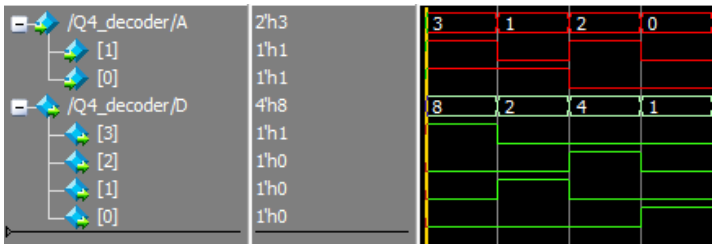

Figure 11: Q4 Waveform

Question 5

Question: Implement an even parity generator module using assign statement.

Solution: Model Generates Even Parity

```
1  module Q5(A, out_with_parity);
2
3      input [7:0] A;
4      output [8:0] out_with_parity;
5
6      assign out_with_parity[8:1] = A;
7      assign out_with_parity[0] = ~^A;
8
9  endmodule
```

Figure 12: Q5 Code

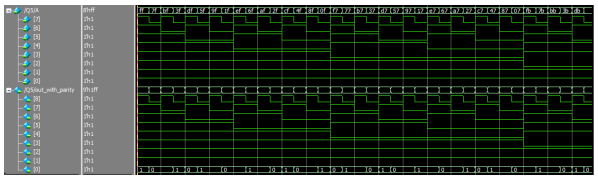

Figure 13: Q5 Waveform

Question 6

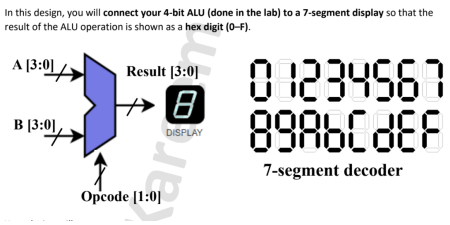

Figure 14: Question 6

Solution:

```
module SevenSeg_decoder(A, B, opcode, en, a, b, c, d, e, f, g);

    input [3:0] A, B;
    input [1:0] opcode;
    input en;
    output reg a, b, c, d, e, f, g;
    wire [3:0] ALU_out;

    ALUx #(4) ALU4bit(A, B, opcode, ALU_out);

    always @(*) begin

        if(en) begin
            case(ALU_out)
                4'd0:  {a, b, c, d, e, f, g} = 7'b1111110;
                4'd1:  {a, b, c, d, e, f, g} = 7'b0110000;
                4'd2:  {a, b, c, d, e, f, g} = 7'b1101101;
                4'd3:  {a, b, c, d, e, f, g} = 7'b1111001;
                4'd4:  {a, b, c, d, e, f, g} = 7'b0110011;
                4'd5:  {a, b, c, d, e, f, g} = 7'b1011011;
                4'd6:  {a, b, c, d, e, f, g} = 7'b1011111;
                4'd7:  {a, b, c, d, e, f, g} = 7'b1110000;
                4'd8:  {a, b, c, d, e, f, g} = 7'b1111111;
                4'd9:  {a, b, c, d, e, f, g} = 7'b1111011;
                4'd10: {a, b, c, d, e, f, g} = 7'b1110111;
                4'd11: {a, b, c, d, e, f, g} = 7'b0011111;
                4'd12: {a, b, c, d, e, f, g} = 7'b1001110;
                4'd13: {a, b, c, d, e, f, g} = 7'b0111101;
                4'd14: {a, b, c, d, e, f, g} = 7'b1001111;
                4'd15: {a, b, c, d, e, f, g} = 7'b1000111;
                default:
                    {a, b, c, d, e, f, g} = 7'b0000000;
            endcase
        end // End of If statement
        else
            {a, b, c, d, e, f, g} = 7'b0000000;
    end // End of Always block

endmodule
```

Figure 15: Q6 Code

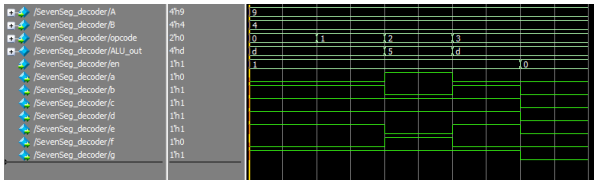

Figure 16: Q6 Waveform