

Digital Design Assignment 1

Name: Ahmed Nour

ID in sheet: 16

Q1)

The screenshot displays a digital design software interface. The top window, titled 'q1_1.v', contains a Verilog module named 'muxed'. The code is as follows:

```
1 module muxed (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
6 assign out = (sel == 0)? (a & b & c) : (d ^ e ^ f);
7 assign out_bar = ~out ;
8
9 endmodule
```

The bottom window shows a timing diagram for the 'muxed' module. It lists the signals: 'muxed_a' (1b1), 'muxed_b' (1b0), 'muxed_c' (1b1), 'muxed_d' (1b1), 'muxed_e' (1b1), 'muxed_f' (1b1), 'muxed_out' (1b1), and 'muxed_out_bar' (1b0). The diagram shows the digital levels of these signals over time, with a vertical yellow line indicating a specific point in the simulation.

Q2)

```
q2_1.v x
Generate Simulate
Q2> q2_1.v muxed2
1 module muxed2 (d, a, b, c, sel, out, out_bar);
2
3 input [2:0] d;
4 input a, b, c, sel;
5 output reg out, out_bar;
6
7 always @(*) begin
8 casex (sel)
9 1'b0 : out = d[2] | (d[1] & d[0]);
10 1'b1 : out = a ^ (b ^ c);
11 endcase
12 out_bar = ~out;
13 end
14
15 endmodule
```

[illegible]

Q3)

The screenshot shows a Verilog code editor with the following code:

```
1 module adder4b (a, b, c);  
2  
3 input [3:0] a, b;  
4 output [3:0] c;  
5  
6 assign c = a + b;  
7  
8 endmodule
```

Below the code editor is a logic simulator window. It displays three test cases for the adder module. The inputs are labeled as /adder4b/a, /adder4b/b, and /adder4b/c. The outputs are shown in a table with columns for each input and the output c.

| | /adder4b/a | /adder4b/b | /adder4b/c |
|------|------------|------------|------------|
| 0011 | 0101 | 0000 | 0110 |
| 0011 | 0110 | | |
| 0110 | 1011 | 0110 | 1100 |
| | | | 0000 |

Note: the last test case has 1 in ignored carrier.

Q4)

q4_1.v

Generate Simulate

Ass1 > Q4 > q4_1.v > ...

```
1 module decoderin2out4 (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) :
9           (a == 2'b11) ? (4'b1000) :
10          (4'b0000);
11
12 endmodule
```

/decoderin2out4/a

/decoderin2out4/d

Msgs

-No Data-

-No Data-

| | | | | |
|---|---|---|---|--|
| 0 | 1 | 2 | 3 | |
| 1 | 2 | 4 | 8 | |

Q5)

```
q5_1.v x
Ass1 > Q5 > q5_1.v > ...
1 module out_with_parity (a, d);
2
3 input [7:0] a;
4 output [8:0] d;
5
6 assign d = {(^a) , a};
7
8 endmodule
9
```

| | Msgs | | | | |
|----------------------|--------------|-----------|-----------|--|--|
| + /out_with_parity/a | 8'b10011110 | 10011010 | 10011110 | | |
| + /out_with_parity/d | 9'b110011110 | 010011010 | 110011110 | | |
| | | | | | |

Q6)

```
Ass1 > Q6 > q6_1_aluv > Nbit ALU
1 module Nbit_ALU (in0, in1, opcode, out);
2     parameter N1 = 4;
3
4     input [N1-1:0] in0, in1;
5     input [1:0] opcode;
6
7     output reg [N1-1:0] out;
8
9     always @(*) begin
10         case (opcode)
11             2'b00: out = in0 + in1;
12             2'b01: out = in0 | in1;
13             2'b10: out = in0 - in1;
14             2'b11: out = in0 ^ in1;
15         endcase
16     end
17 endmodule
```

```

1 module sev_seg_dec (in0, enable, a, b, c, d, e, f, g);
2   parameter N2 = 4;
3   input [N2 - 1 : 0] in0;
4   input enable;
5   output reg a, b, c, d, e, f, g;
6
7   wire in1;
8   always @(*) begin
9     if (enable) begin
10      case (in0)
11        4'h0: {a,b,c,d,e,f,g} = 7'b1111110;
12        4'h1: {a,b,c,d,e,f,g} = 7'b0110000;
13        4'h2: {a,b,c,d,e,f,g} = 7'b1101101;
14        4'h3: {a,b,c,d,e,f,g} = 7'b1111001;
15        4'h4: {a,b,c,d,e,f,g} = 7'b0110011;
16        4'h5: {a,b,c,d,e,f,g} = 7'b1011011;
17        4'h6: {a,b,c,d,e,f,g} = 7'b1011111;
18        4'h7: {a,b,c,d,e,f,g} = 7'b1110000;
19        4'h8: {a,b,c,d,e,f,g} = 7'b1111111;
20        4'h9: {a,b,c,d,e,f,g} = 7'b1111011;
21        4'hA: {a,b,c,d,e,f,g} = 7'b1110111;
22        4'hB: {a,b,c,d,e,f,g} = 7'b0011111;
23        4'hC: {a,b,c,d,e,f,g} = 7'b1001110;
24        4'hD: {a,b,c,d,e,f,g} = 7'b0111101;
25        4'hE: {a,b,c,d,e,f,g} = 7'b1001111;
26        4'hF: {a,b,c,d,e,f,g} = 7'b1000111;
27        default: {a,b,c,d,e,f,g} = 7'b0000000;
28      endcase
29    end else begin {a,b,c,d,e,f,g} = 7'b0000000; end
30  end
31 endmodule

```

```

Ass1>Q6> @ q6_3_execv > e executor > [ ] ALUout
1  module executor (Ain, Bin, opcode, enable, ALUout, a, b, c, d, e, f, g);
2  parameter N = 4;
3
4  input [N-1 :0] Ain, Bin;
5  input [1:0] opcode;
6  input enable;
7
8  output [N-1 :0] ALUout;
9  output a, b, c, d, e, f, g;
10
11  Nbit_ALU alu1 (Ain, Bin, opcode, ALUout);
12  sev_seg_dec sdec1 (ALUout, enable, a, b, c, d, e, f, g);
13
14  endmodule

```

[illegible]