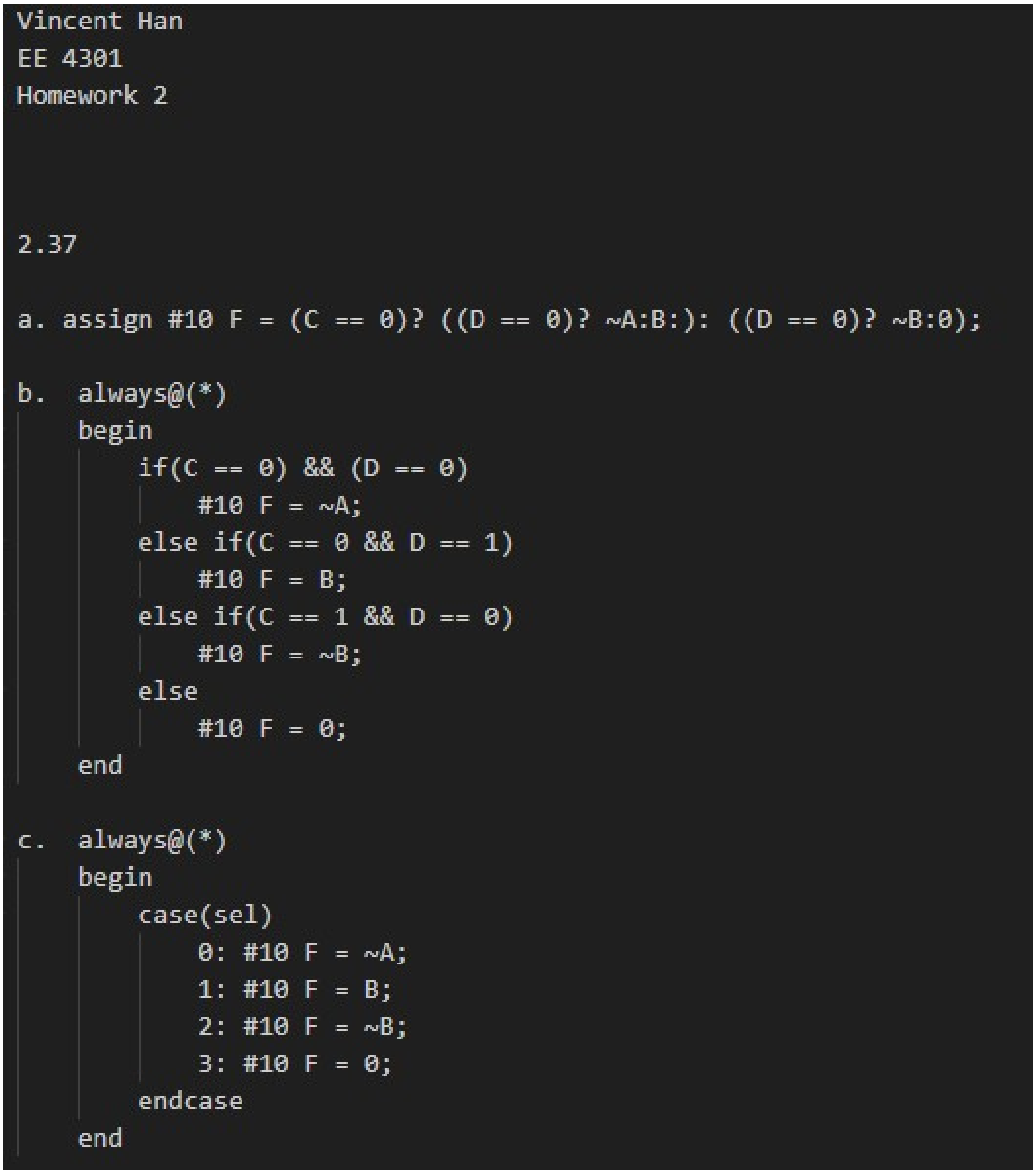
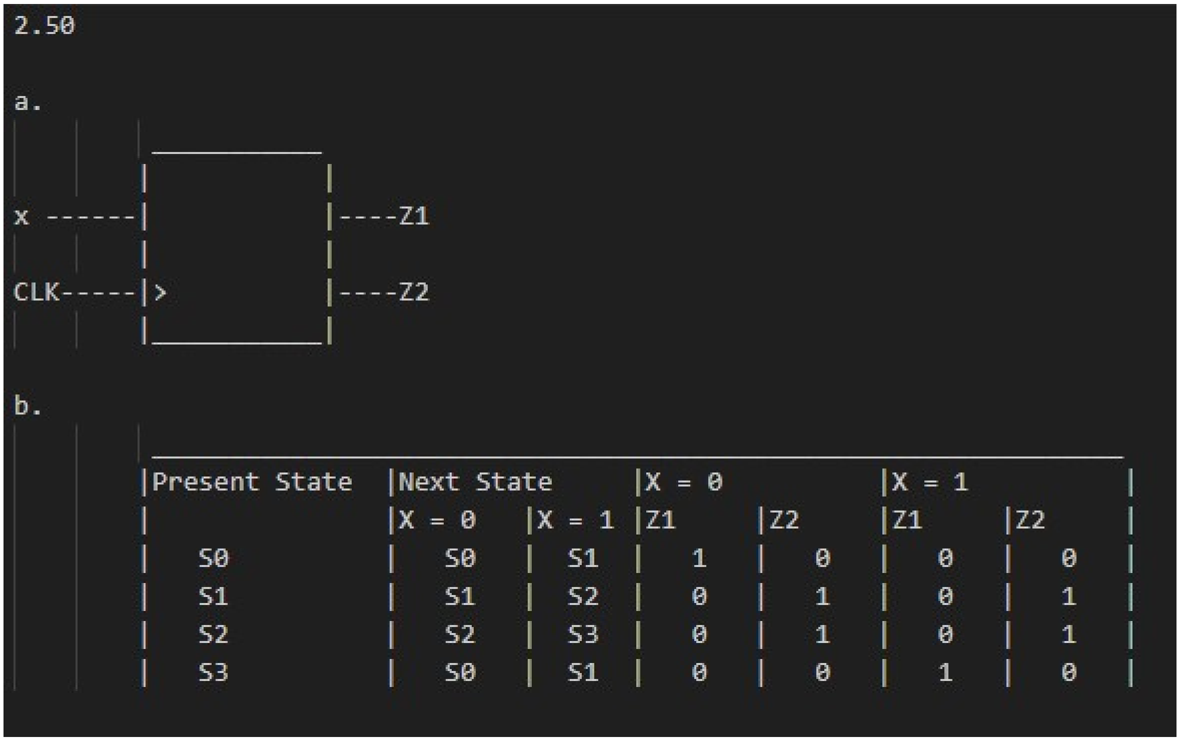


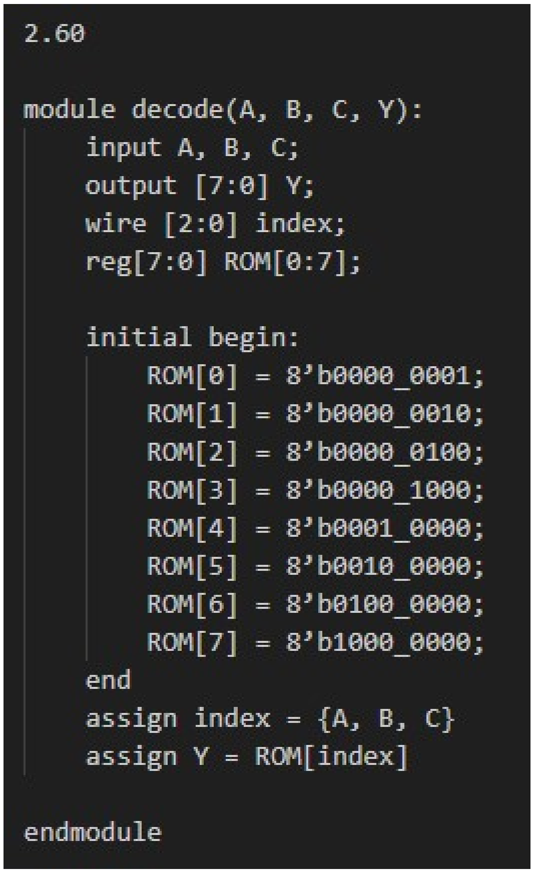
Vincent Han
EE 4301
Homework 2

2.37

```
a. assign #10 F = (C == 0)? ((D == 0)? ~A:B:): ((D == 0)? ~B:0);

b.  always@(*)
    begin
        if(C == 0) && (D == 0)
            #10 F = ~A;
        else if(C == 0 && D == 1)
            #10 F = B;
        else if(C == 1 && D == 0)
            #10 F = ~B;
        else
            #10 F = 0;
    end

c.  always@(*)
    begin
        case(sel)
            0: #10 F = ~A;
            1: #10 F = B;
            2: #10 F = ~B;
            3: #10 F = 0;
        endcase
    end
```



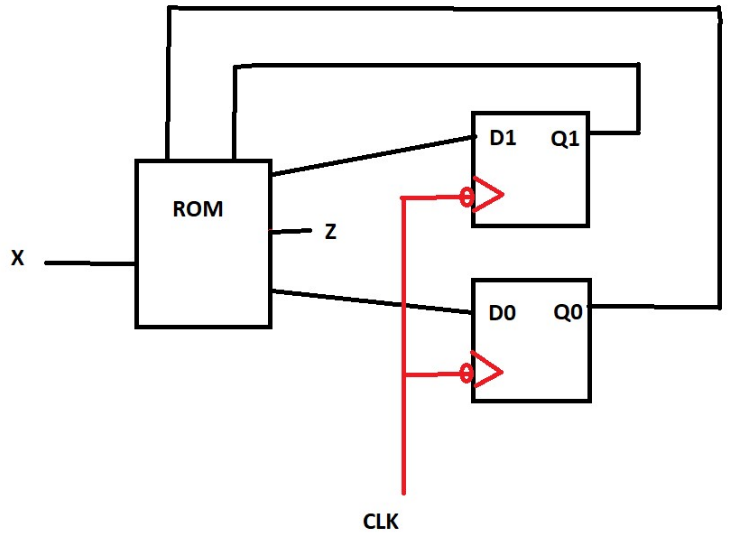
```
2.50

a.
         ___________
        |           |
x ------|           |----Z1
        |           |
CLK-----|>          |----Z2
        |___________|

b.
         _______________________________________________________________
        |Present State  |Next State     |X = 0          |X = 1          |
        |               |X = 0   |X = 1 |Z1      |Z2     |Z1      |Z2     |
        |   S0          |   S0   |  S1  |   1   |   0   |   0   |   0   |
        |   S1          |   S1   |  S2  |   0   |   1   |   0   |   1   |
        |   S2          |   S2   |  S3  |   0   |   1   |   0   |   1   |
        |   S3          |   S0   |  S1  |   0   |   0   |   1   |   0   |
```

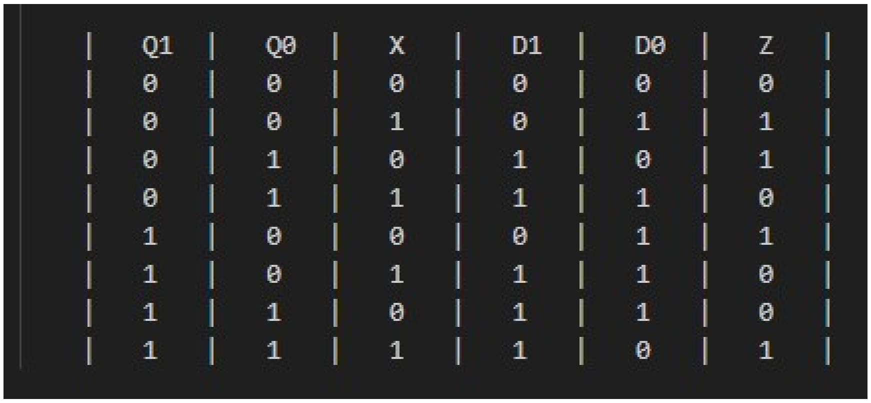


```
2.60

module decode(A, B, C, Y):
    input A, B, C;
    output [7:0] Y;
    wire [2:0] index;
    reg[7:0] ROM[0:7];

    initial begin:
        ROM[0] = 8’b0000_0001;
        ROM[1] = 8’b0000_0010;
        ROM[2] = 8’b0000_0100;
        ROM[3] = 8’b0000_1000;
        ROM[4] = 8’b0001_0000;
        ROM[5] = 8’b0010_0000;
        ROM[6] = 8’b0100_0000;
        ROM[7] = 8’b1000_0000;
    end
    assign index = {A, B, C}
    assign Y = ROM[index]

endmodule
```

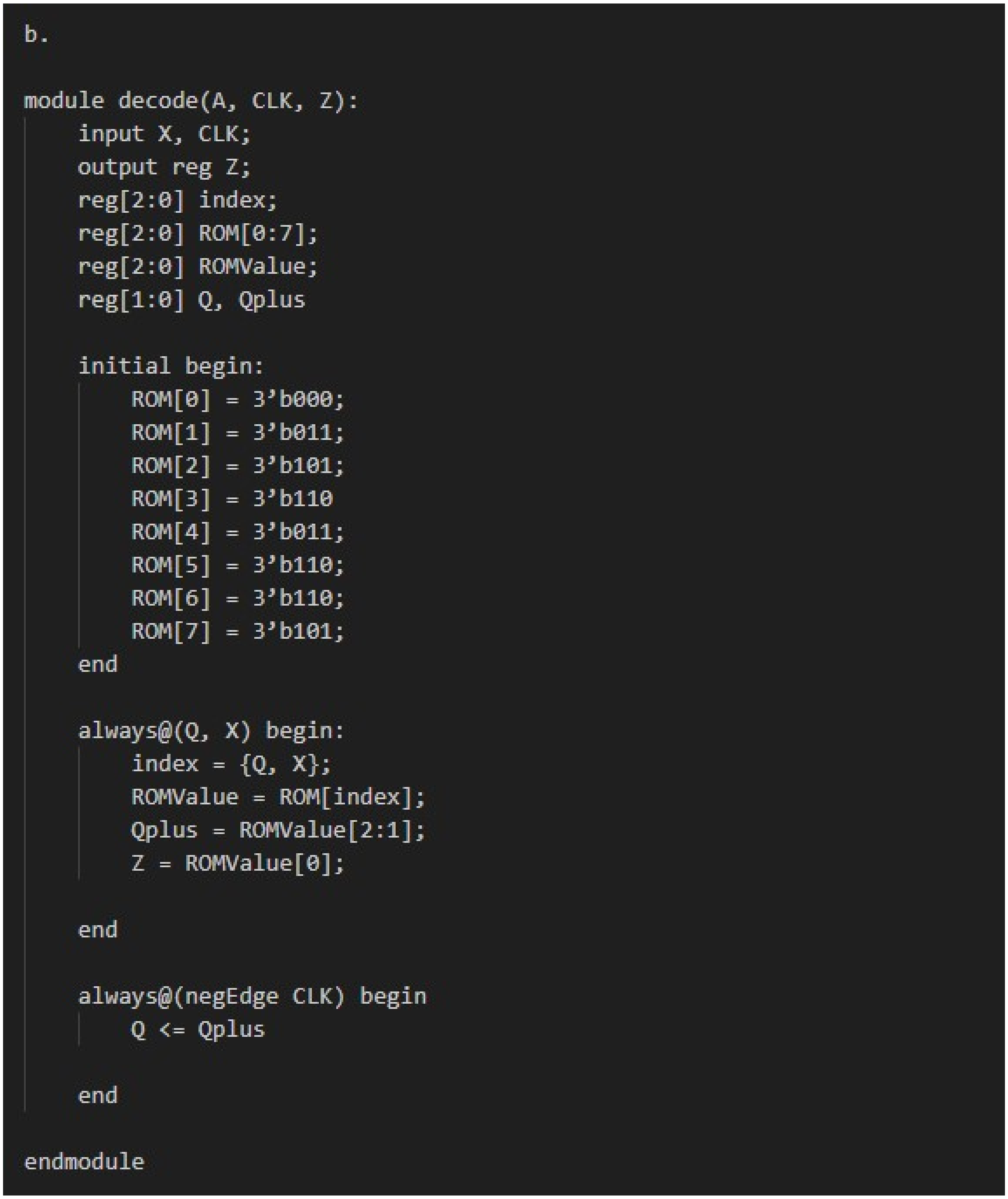
3.3

a.





Q1	Q0	X	D1	D0	Z
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	1	0	1
0	1	1	1	1	0
1	0	0	0	1	1
1	0	1	1	1	0
1	1	0	1	1	0
1	1	1	1	0	1



```
b.

module decode(A, CLK, Z):
    input X, CLK;
    output reg Z;
    reg[2:0] index;
    reg[2:0] ROM[0:7];
    reg[2:0] ROMValue;
    reg[1:0] Q, Qplus

    initial begin:
        ROM[0] = 3’b000;
        ROM[1] = 3’b011;
        ROM[2] = 3’b101;
        ROM[3] = 3’b110
        ROM[4] = 3’b011;
        ROM[5] = 3’b110;
        ROM[6] = 3’b110;
        ROM[7] = 3’b101;
    end

    always@(Q, X) begin:
        index = {Q, X};
        ROMValue = ROM[index];
        Qplus = ROMValue[2:1];
        Z = ROMValue[0];

    end

    always@(negEdge CLK) begin
        Q <= Qplus

    end

endmodule
```