📘 **Day 9 Topic: Verilog codes for 8-to-3 Encoder Using Cadence (Xcelium)**

🔹 What is an Encoder?

- An encoder is the inverse of a decoder.
- It takes 2^n inputs and produces n outputs.
- At any given time, only one input should be active (1).
- The encoder generates the binary code of the active input line.

👉 Example: An 8-to-3 encoder has 8 inputs (D0–D7) and 3 outputs (A,B,C).

🔹 8-to-3 Encoder

- Inputs: D0 to D7 (8 lines, one active at a time)
- Outputs: A,B,C (3-bit binary code)

The decoder activates **exactly one output** corresponding to the binary value of (A,B,C).

🔹 Truth Table (8 inputs → 3 outputs)

Inputs (One Active = 1)	Output (ABC)
D0 = 1	000
D1 = 1	001
D2 = 1	010
D3 = 1	011
D4 = 1	100
D5 = 1	101
D6 = 1	110
D7 = 1	111

👉 If D5 = 1, the output will be 101 (binary for 5).

🔹 **1.1) Design Code**

encoder_8_3.v

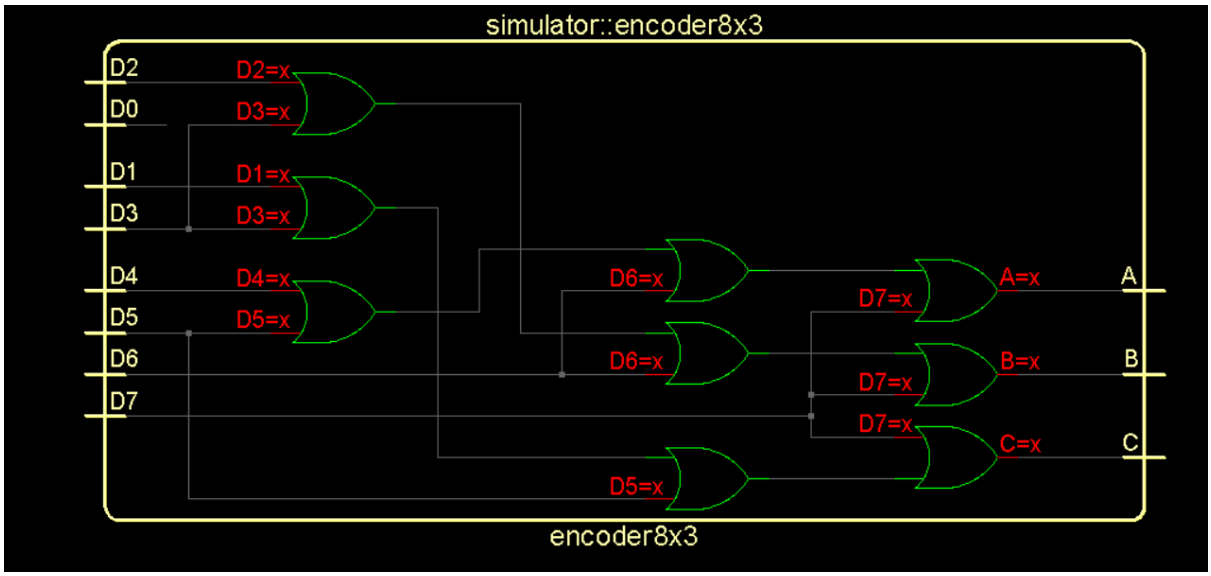
```
// 8-to-3 Encoder (no priority)
module encoder8x3(
    input  D0, D1, D2, D3, D4, D5, D6, D7,
    output A, B, C
);
    assign A = D4 | D5 | D6 | D7;    // MSB
    assign B = D2 | D3 | D6 | D7;
    assign C = D1 | D3 | D5 | D7;    // LSB
endmodule
```

- **1.2) Test Bench Code**

tb.v

```
module tb_encoder8x3;
    reg D0, D1, D2, D3, D4, D5, D6, D7;
    wire A, B, C;

    // Instantiate DUT
    encoder8x3 uut (.D0(D0), .D1(D1), .D2(D2), .D3(D3),
                    .D4(D4), .D5(D5), .D6(D6), .D7(D7),
                    .A(A), .B(B), .C(C));

    initial begin
        $display("D7..D0 | A B C");
        $display("----------------");

        {D7,D6,D5,D4,D3,D2,D1,D0} = 8'b00000001; #10;  // expect 000
        {D7,D6,D5,D4,D3,D2,D1,D0} = 8'b00000010; #10;  // expect 001
        {D7,D6,D5,D4,D3,D2,D1,D0} = 8'b00000100; #10;  // expect 010
        {D7,D6,D5,D4,D3,D2,D1,D0} = 8'b00001000; #10;  // expect 011
        {D7,D6,D5,D4,D3,D2,D1,D0} = 8'b00010000; #10;  // expect 100
        {D7,D6,D5,D4,D3,D2,D1,D0} = 8'b00100000; #10;  // expect 101
        {D7,D6,D5,D4,D3,D2,D1,D0} = 8'b01000000; #10;  // expect 110
        {D7,D6,D5,D4,D3,D2,D1,D0} = 8'b10000000; #10;  // expect 111

        $stop;
    end
endmodule
```

- **1.3) Schematic**



- **1.4) Wave Forms**