📘 **Day 17 Topic: Verilog codes for JK Latch Using Cadence (Xcelium)**

🔹 What is a Latch?

A latch is the simplest form of a memory element in digital electronics.

- It can store 1 bit of data.
- It is level-sensitive → works as long as the Enable (or Clock) input is active.
- Unlike flip-flops (which are edge-triggered), latches are transparent when enabled.

JK Latch Description

A JK latch is a sequential logic circuit and an improved version of the SR latch.
The problem with an SR latch is that when S = 1 and R = 1, the output becomes invalid/undefined. The JK latch removes this problem by adding feedback from the output to the inputs.

Inputs and Outputs

- Inputs:
    - J (Set input)
    - K (Reset input)
    - Enable (CLK or Gate signal) – allows the latch to change state
- Outputs:
    - Q (main output)
    - Q′ (complement output)

Working

When Enable = 0, the latch is disabled → output remains unchanged (latched).
When Enable = 1, the output depends on J and K inputs:

1. J = 0, K = 0 → No change (latch holds the same state)
2. J = 0, K = 1 → Reset (Q = 0, Q′ = 1)

3. J = 1, K = 0 → Set (Q = 1, Q′ = 0)
4. J = 1, K = 1 → Toggle (Q switches to the opposite state)

Truth Table

Enable	J	K	Next State (Q^{n+1})
0	X	X	Q^n (no change)
1	0	0	Q^n (no change)
1	0	1	0 (reset)
1	1	0	1 (set)
1	1	1	$\bar{Q}^n$ (toggle)

Circuit

- Built using two cross-coupled NAND gates or NOR gates (like SR latch).
- Feedback from outputs ensures that when J = K = 1, the latch toggles instead of being invalid (as in SR latch).

🔹 **1.1) Design Code**

jk_latch.v

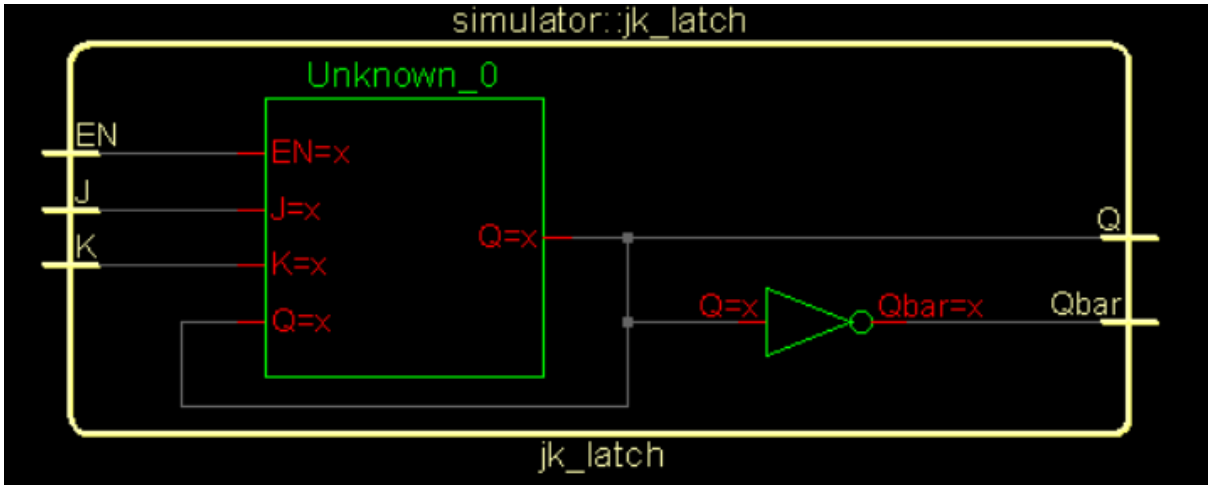
```
// JK Latch (Level Sensitive)
module jk_latch (
    input wire J, K, EN,
    output reg Q,
    output wire Qbar
);

    assign Qbar = ~Q;

    always @(*) begin
        if (EN) begin
            case ({J,K})
                2'b00: Q <= Q;      // Hold
                2'b01: Q <= 0;      // Reset
                2'b10: Q <= 1;      // Set
                2'b11: Q <= ~Q;     // Toggle
            endcase
        end
        else
            Q <= Q; // Hold when EN=0
    end
endmodule
```

- **1.2) Test Bench Code**

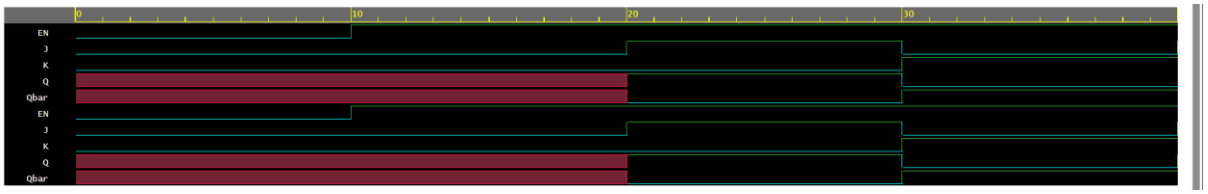
tb.v

```
module tb_jk_latch;
    reg J, K, EN;
    wire Q, Qbar;

    // Instantiate JK Latch
    jk_latch uut (.J(J), .K(K), .EN(EN), .Q(Q), .Qbar(Qbar));

    initial begin
        $monitor("T=%0t | EN=%b J=%b K=%b | Q=%b Qbar=%b", $time, EN, J, K, Q, Qbar);

        // Start with EN=0 (Latch disabled)
        EN=0; J=0; K=0; #10;

        // Enable latch and test inputs
        EN=1; J=0; K=0; #10; // Hold
        EN=1; J=1; K=0; #10; // Set
        EN=1; J=0; K=1; #10; // Reset
        EN=1; J=1; K=1; #10; // Toggle
        EN=0; J=1; K=0; #10; // Hold (because EN=0)

        $finish;
    end
endmodule
```

- **1.3) Schematic**



- **1.4) Wave Forms**



Applications

- Counters: JK latches are widely used in asynchronous and synchronous counters.
- Frequency Division: Works as a toggle circuit.
- Shift Registers & FSMs: Used in sequential logic design where toggle and memory are required.
- Memory Elements: Basic building block for flip-flops.