📘 **Day 16 Topic: Verilog codes for SR 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.

SR Latch (Set-Reset Latch)

An SR latch is the simplest type of sequential logic circuit. It is used for storing one bit of information (0 or 1). The name SR comes from its two inputs:

- S → Set
- R → Reset

It is also called a bistable multivibrator, because it has two stable states.

Inputs and Outputs

- Inputs: S (Set), R (Reset)
- Outputs: Q (main output), $\bar{Q}$ (complement of Q)

Working Principle

1. Set Condition (S=1, R=0):
    - Output Q becomes 1
    - $\bar{Q}$ becomes 0
    - This means the latch is "set" (stores logic 1).
2. Reset Condition (S=0, R=1):
    - Output Q becomes 0
    - $\bar{Q}$ becomes 1

- This means the latch is "reset" (stores logic 0).

3. No Change Condition (S=0, R=0):
    - The latch retains its previous state.
    - It remembers the last stored value.
4. Invalid Condition (S=1, R=1):
    - Both Q and $\bar{Q}$ become 0, which is not valid because outputs should be complements.
    - This state is forbidden/undefined.

Truth Table

S	R	Q (next state)	$\bar{Q}$
0	0	Previous Q	Previous $\bar{Q}$
0	1	0	1
1	0	1	0
1	1	Invalid	Invalid

Implementation

- NAND-based SR latch: Inputs are active low (Set and Reset work when 0).
- NOR-based SR latch: Inputs are active high (Set and Reset work when 1).

◆ **1.1) Design Code**

sr_latch.v

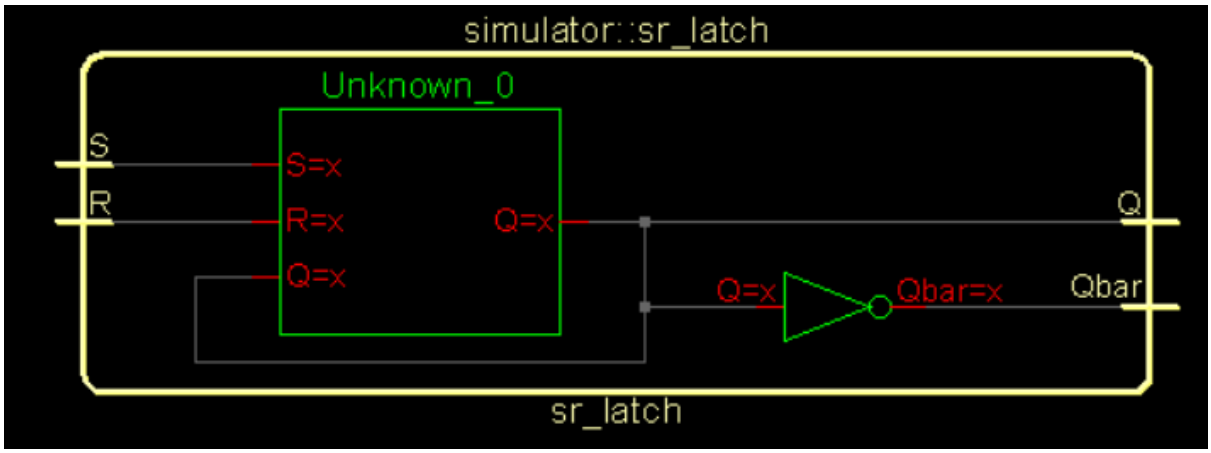
```
module sr_latch (
    input wire S,    // Set input
    input wire R,    // Reset input
    output reg Q,    // Output
    output wire Qbar // Complement of Q
);

    // Continuous assignment for complement
    assign Qbar = ~Q;

    // Always block models latch behavior
    always @(*) begin
        if (S && !R)       // Set condition
            Q <= 1;
        else if (!S && R)  // Reset condition
            Q <= 0;
        else if (!S && !R) // Hold condition
            Q <= Q; // no change
        else               // S=1, R=1 (invalid)
            Q <= 1'bx;     // unknown
    end

endmodule
```

- **1.2) Test Bench Code**

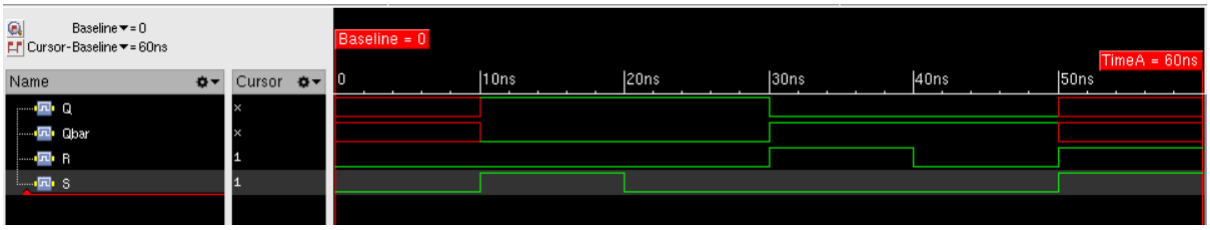
tb.v

```
module tb_sr_latch;
    reg S, R;        // inputs
    wire Q, Qbar;    // outputs

    // Instantiate the SR latch
    sr_latch uut (.S(S), .R(R), .Q(Q), .Qbar(Qbar));

    initial begin
        // Monitor changes
        $monitor("Time=%0t | S=%b R=%b | Q=%b Qbar=%b", $time, S, R, Q, Qbar);

        // Test cases
        S = 0; R = 0; #10; // Hold (initial)
        S = 1; R = 0; #10; // Set
        S = 0; R = 0; #10; // Hold
        S = 0; R = 1; #10; // Reset
        S = 0; R = 0; #10; // Hold
        S = 1; R = 1; #10; // Invalid state
        $finish;
    end
endmodule
```

- **1.3) Schematic**



- **1.4) Wave Forms**



Applications

- Used in basic memory elements.
- Used in flip-flop design (JK, D, T flip-flops are built from SR latch).
- Useful in control circuits where one-bit storage is needed.