📘 **Day 21 Topic: Verilog codes for D Flipflop Using Cadence (Xcelium)**

🔹 What is a Flip-Flop?

- A flip-flop is a small memory element in digital electronics.
- It can store 1 bit of data: either 0 (LOW) or 1 (HIGH).
- It is used in registers, counters, and memory circuits.

D Flip-Flop Description

A D flip-flop (Data or Delay flip-flop) is a sequential logic circuit that captures the value of the D (data) input at a particular clock edge (either rising or falling).
After the clock edge, the captured value is stored and appears at the output Q until the next active clock edge arrives.

It is widely used as a data storage element in digital systems such as registers, counters, and memory devices.

Inputs and Outputs

- Inputs:
  - D (Data input)
  - Clock (CLK) – edge-triggered (rising ↑ or falling ↓)
  - *(Sometimes asynchronous inputs like Preset and Clear are also provided in practical ICs)*
- Outputs:
  - Q (main output)
  - Q′ (complement output)

Working Principle

The D flip-flop simply transfers the input D to output Q on every active clock edge.

- When the clock edge occurs (↑ or ↓ depending on design):
  - If D = 1 → Q becomes 1 (Set)

  - If D = 0 → Q becomes 0 (Reset)

Between clock edges, Q remains constant, even if D changes.

Hence, the D flip-flop is also called a data latch or data buffer, because it holds the value of D until the next clock edge.

Truth Table (Edge-Triggered)

CLK Edge	D	Next State (Q^{n+1})
↑ / ↓	0	0 (reset)
↑ / ↓	1	1 (set)

(↑ means rising-edge triggered, ↓ means falling-edge triggered depending on the flip-flop design)

Characteristic Equation

$$Q_{next} = D$$

This means the next state of the flip-flop is simply the value of D input at the clock edge.

Implementation

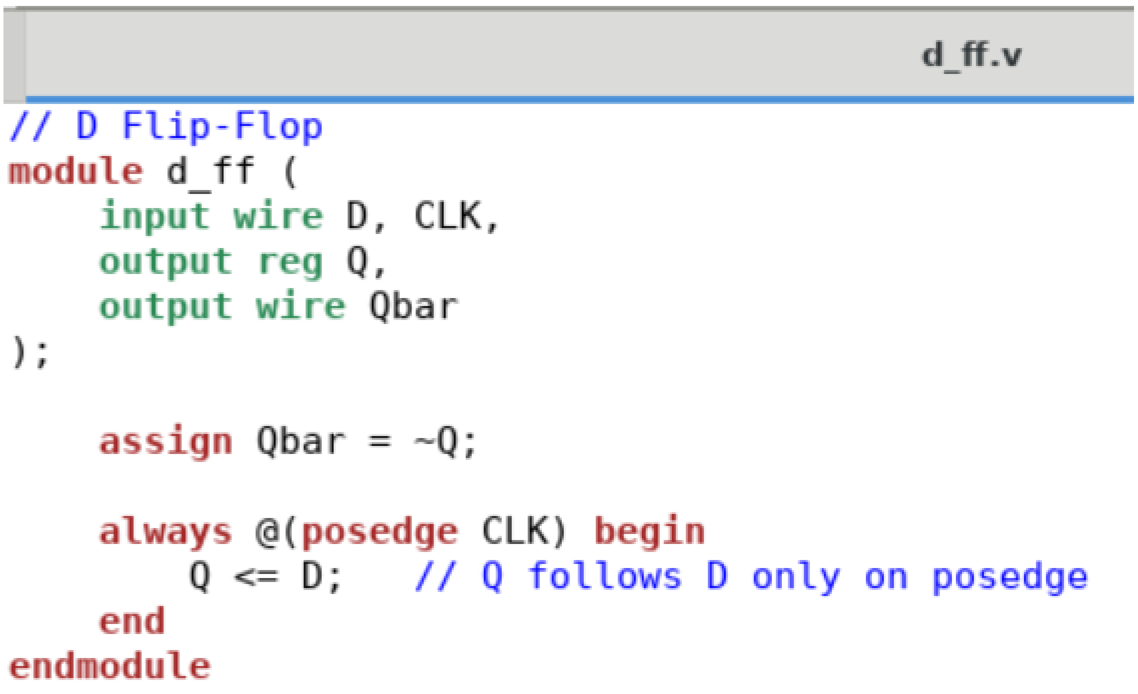
- The D flip-flop can be constructed from an SR flip-flop by connecting:

$$S = D, R = \bar{D}$$

This eliminates the invalid state (S = R = 1) problem of the SR flip-flop.

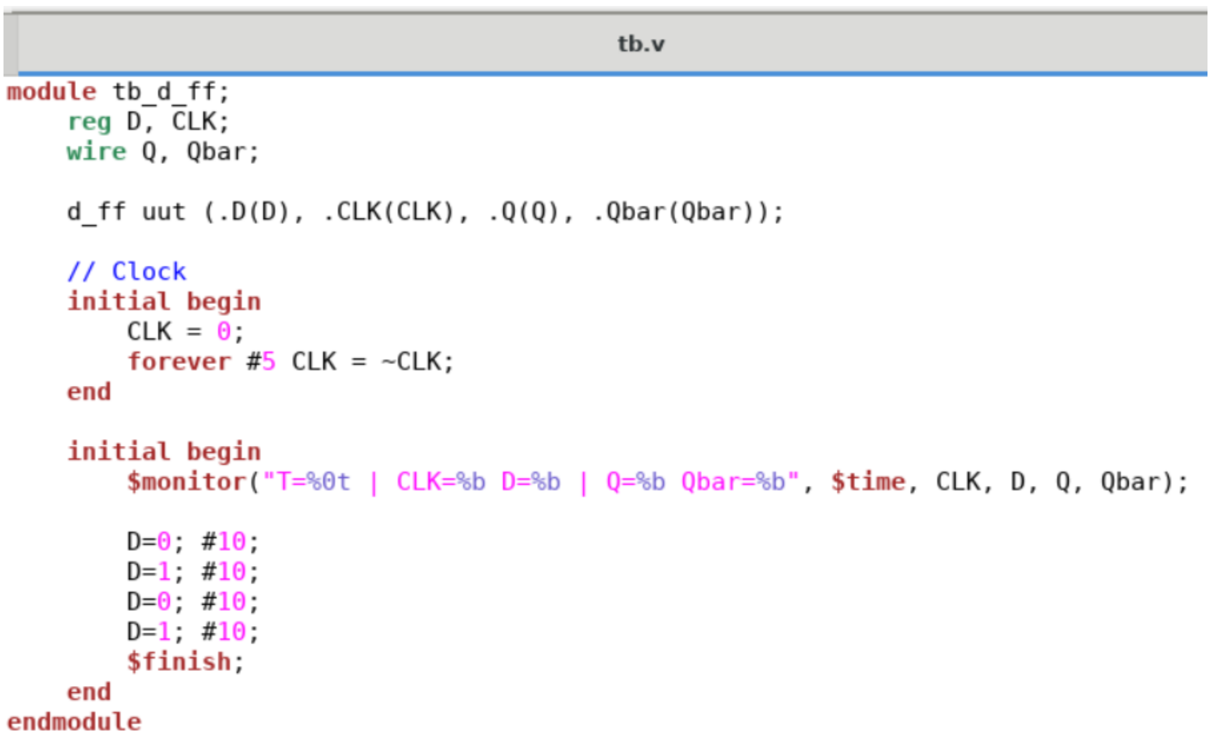
- Modern D flip-flops are edge-triggered and often built using master-slave latch configurations.

◆ **1.1) Design Code**



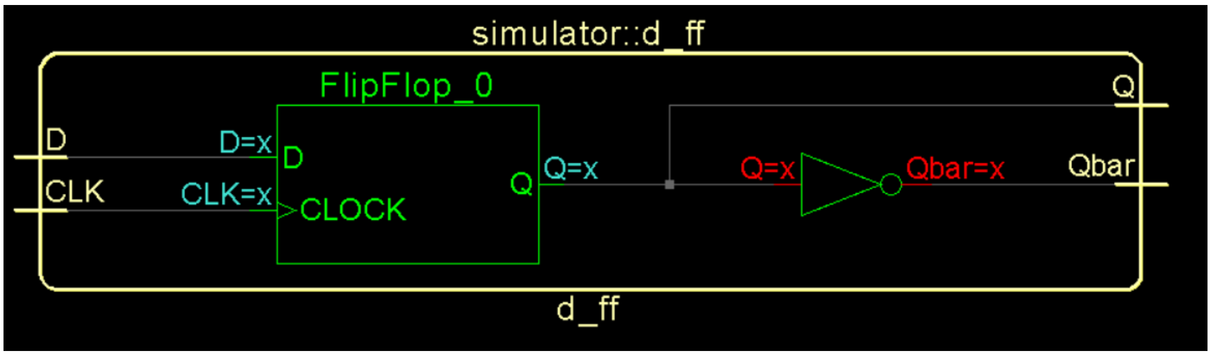
```verilog
// D Flip-Flop
module d_ff (
    input wire D, CLK,
    output reg Q,
    output wire Qbar
);

    assign Qbar = ~Q;

    always @(posedge CLK) begin
        Q <= D;   // Q follows D only on posedge
    end
endmodule
```

- **1.2) Test Bench Code**

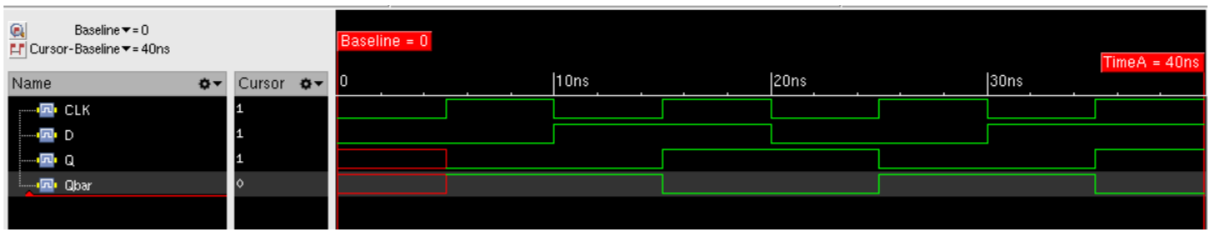


```verilog
module tb_d_ff;
    reg D, CLK;
    wire Q, Qbar;

    d_ff uut (.D(D), .CLK(CLK), .Q(Q), .Qbar(Qbar));

    // Clock
    initial begin
        CLK = 0;
        forever #5 CLK = ~CLK;
    end

    initial begin
        $monitor("T=%0t | CLK=%b D=%b | Q=%b Qbar=%b", $time, CLK, D, Q, Qbar);

        D=0; #10;
        D=1; #10;
        D=0; #10;
        D=1; #10;
        $finish;
    end
endmodule
```

◆ **1.3) Schematic**



◆ **1.4) Wave Forms**



Applications

1. Data Storage:
   - Stores one bit of data per flip-flop — fundamental memory element.
2. Registers:
   - Multiple D flip-flops are combined to form shift registers and data registers.

3. Counters:
    - Used in synchronous and asynchronous counters as intermediate storage.
4. Frequency Division:
    - Can act as a simple frequency divider (divides input clock by 2 when connected with feedback).
5. Synchronization:
    - Used to synchronize asynchronous signals with the system clock in digital systems (avoids metastability).
6. Finite State Machines (FSM):
    - Used to store and update the present state of FSMs.