**Module: SV for Verification**

**Section:** Coverage **Task:** Assertions

**Task 1 -** [**EDA Playground**](https://www.edaplayground.com/x/fU8V)

Assertions

* **Source Code:**
* **Design Code:**

module SignalDetector(input logic clk, signal\_in);

// Property for checking signal stability

property check\_stability\_3\_cycles;

@(posedge clk)

(signal\_in == 1'b1) ##1 $stable(signal\_in) ##1 $stable(signal\_in);// Check if signal\_in is 1 for 3 consecutive cycles

endproperty

// Detecting rising edge

assert property(@(posedge clk) $rose(signal\_in)) // Check if the signal has gone from 0 to 1

else $error("ERROR: Signal Didn't rise @%0t", $time);

// Detecting falling edge

assert property(@(posedge clk) $fell(signal\_in)) // Check if the signal has gone from 1 to 0

else $error("ERROR: Signal Didn't Fall @%0t", $time);

// Stability Check

assert property (check\_stability\_3\_cycles)

else $error("ERROR: signal\_in was not stable for 3 consecutive clock cycles @%0t", $time);

endmodule

* + **Testbench Code:**

module tb;

reg clk, signal\_in;

// Module Instantiation

SignalDetector dut(.clk(clk), .signal\_in(signal\_in));

// Clock Generator

initial begin

clk = 0;

forever #5 clk = ~clk;

end

// Stimulus

initial begin

signal\_in = 0;

$display("@%0t Driven Value:: Signal\_in = %0d", $time, signal\_in);

#20ns;

signal\_in = 1;

$display("@%0t Driven Value:: Signal\_in = %0d", $time, signal\_in);

#30ns;

signal\_in = 0;

$display("@%0t Driven Value:: Signal\_in = %0d", $time, signal\_in);

#10ns;

signal\_in = 1;

$display("@%0t Driven Value:: Signal\_in = %0d", $time, signal\_in);

#20ns;

signal\_in = 0;

$display("@%0t Driven Value:: Signal\_in = %0d", $time, signal\_in);

end

// Dump file

initial begin

$dumpvars;

$dumpfile("dump.vcd");

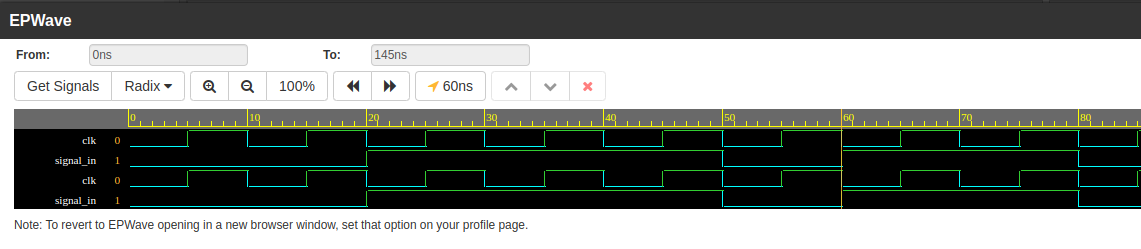
#150;

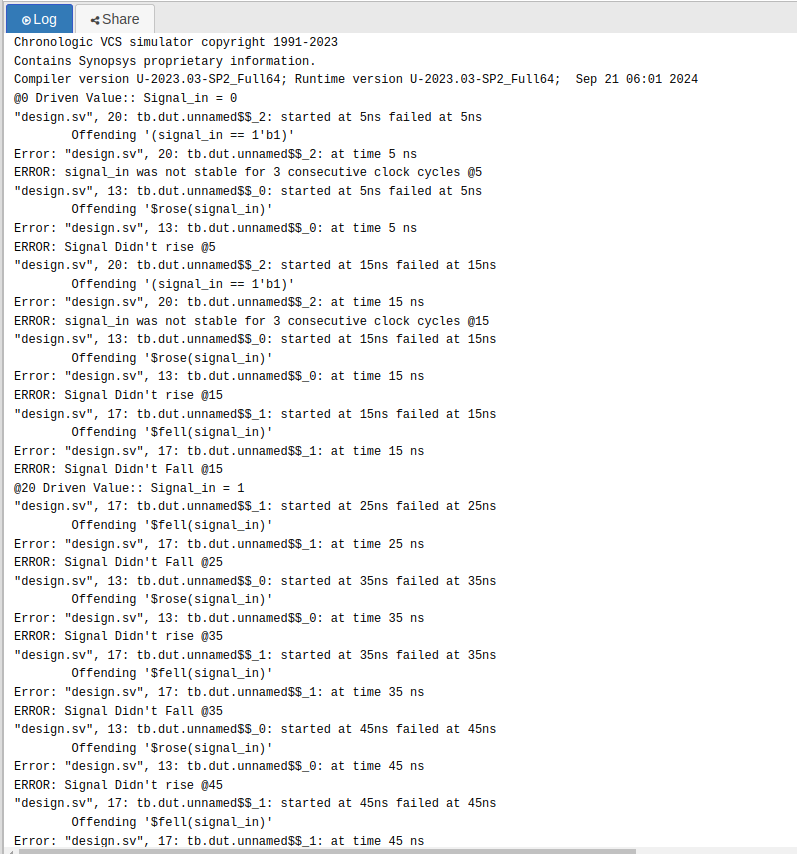
$finish;

end

endmodule

* **Simulation Output:**

****

****