8. 4-bit shift register

CODE:

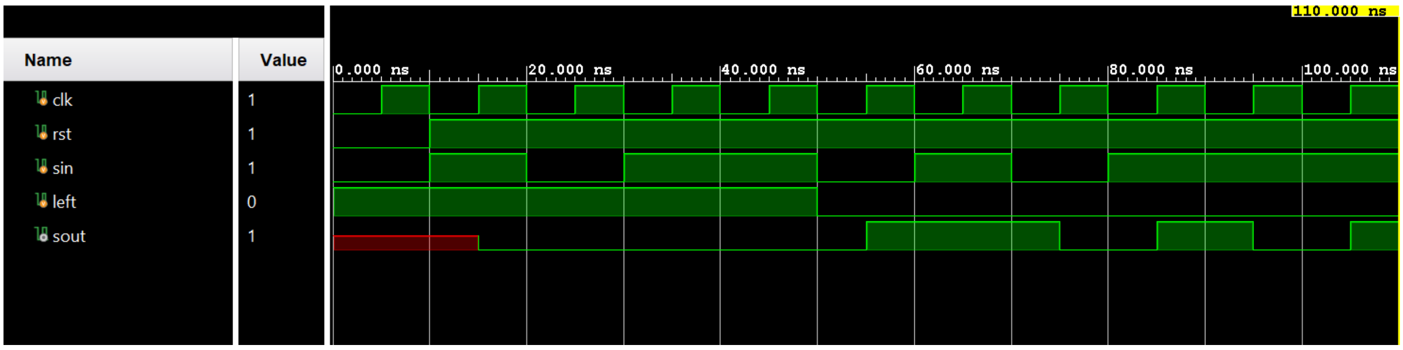
```
`timescale 1ns / 1ps

module shiftreg(
   input clk,rst,sin,left,
   output reg sout
   );
reg [3:0] register;
always@(posedge clk or negedge rst)begin
   if(!rst)
      register <= 4'b0000;
   else begin
      if(left == 1) begin
         sout <= register[3];
         register <= {register[2:0],sin};
      end
      else begin
         sout <= register[0];
         register <= {sin, register[3:1]};
      end
   end
end
endmodule
```

//testbench

```
`timescale 1ns / 1ps

module tb_shiftreg;
   reg clk, rst, sin, left;
   wire sout;

   // Instantiate shift register
   shiftreg uut (
      .clk(clk),
      .rst(rst),
      .sin(sin),
      .left(left),
      .sout(sout)
   );
```

```
    // Clock generation
    initial begin
        clk = 0;
        forever #5 clk = ~clk; // 10ns clock period
    end

    // Stimulus
    initial begin
        // Initialize
        rst = 0; sin = 0; left = 1; // Reset active
        #10 rst = 1;                // Release reset

        // Shift left
        sin = 1; #10; // insert 1
        sin = 0; #10; // insert 0
        sin = 1; #10; // insert 1
        sin = 1; #10; // insert 1

        // Change direction to right
        left = 0;
        sin = 0; #10;
        sin = 1; #10;
        sin = 0; #10;
        sin = 1; #10;

        // End simulation
        #20 $stop;
    end
endmodule
```

Simulation Result:



110.000 ns
Name
Value
clk 1
rst 1
sin 1
left 0
sout 1
0.000 ns
20.000 ns
40.000 ns
60.000 ns
80.000 ns
100.000 ns