

DSD LAB SESSION 7

Design an 8-bit PIPO-SISO Shift Register in Verilog HDL using Behavioral Description

Verilog HDL Code:

```
module pipo_siso_reg(
    input clk, rst, enable, shift_loadbar, serial_in,
    input [7:0] parallel_in,
    output reg serial_out,
    output reg [7:0] parallel_out );
always@(negedge rst or posedge clk) begin
    if (!rst) begin
        parallel_out      <= 'b0;
        serial_out        <= 'b0;
    end
    else begin
        if (enable) begin
            if (!shift_loadbar)
                parallel_out <= parallel_in;
            else begin
                serial_out    <= serial_in;
                parallel_out   <= {parallel_out[6:0], serial_in};
            end
        end
    end
end
endmodule
```

Testbench:

```
`timescale 1ns/1ns
module pipo_siso_tb;
reg clk, rst, enable, shift_loadbar, serial_in;
reg [7:0] parallel_in;
wire serial_out;
wire [7:0] parallel_out;
// Instantiate DUT
pipo_siso_reg DUT(
    .clk(clk),
    .rst(rst),
    .enable(enable),
    .shift_loadbar(shift_loadbar),
    .serial_in(serial_in),
    .parallel_in(parallel_in),
    .serial_out(serial_out),
    .parallel_out(parallel_out));
// Clock generation: 10 ns period
initial clk = 0;
always #5 clk = ~clk;
initial begin
    // Initialize inputs
    rst = 0; enable = 0; shift_loadbar = 0; serial_in = 0; parallel_in = 8'd0;
    // Release reset after 1 clock cycle
    @(posedge clk); //10 ns
    rst = 1;
    // Wait a clock cycle after reset; 20ns
    // Test 1: Load parallel data (shift_loadbar=0)
```

```

enable = 1;
shift_loadbar = 0;
parallel_in = 8'b00001111; // Load this pattern
@(posedge clk); // 30 ns // parallel_out should be 00001111
// Test 2: Shift operation (shift_loadbar=1), serial_in=1
shift_loadbar = 1;
serial_in = 1;
@(posedge clk); //40 ns //parallel_out should shift left, LSB=1
// Test 3: Shift operation again, serial_in=0
serial_in = 0;
@(posedge clk); //50 ns
// Test 4: Disable enable (outputs should hold)
enable = 0;
@(posedge clk);
$stop;
end
endmodule

```

