30 Days of RTL Coding

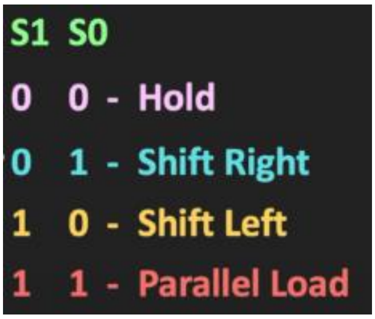
-By T Tushar Shenoy

Day 24

Problem Statement: Implementing Universal Shift register in Structural Style of Modelling.

Theory:

A Universal shift register is a register which has both the right shift and left shift with parallel load capabilities. Universal shift registers are used as memory elements in computers. A Unidirectional shift register is capable of shifting in only one direction. A bidirectional shift register is capable of shifting in both the directions. The Universal shift register is a combination design of **bidirectional** shift register and a **unidirectional** shift register with parallel load provision.



S1	S0	
0	0	- Hold
0	1	- Shift Right
1	0	- Shift Left
1	1	- Parallel Load

FIG: USR Operation for different Combination of S1 and S0

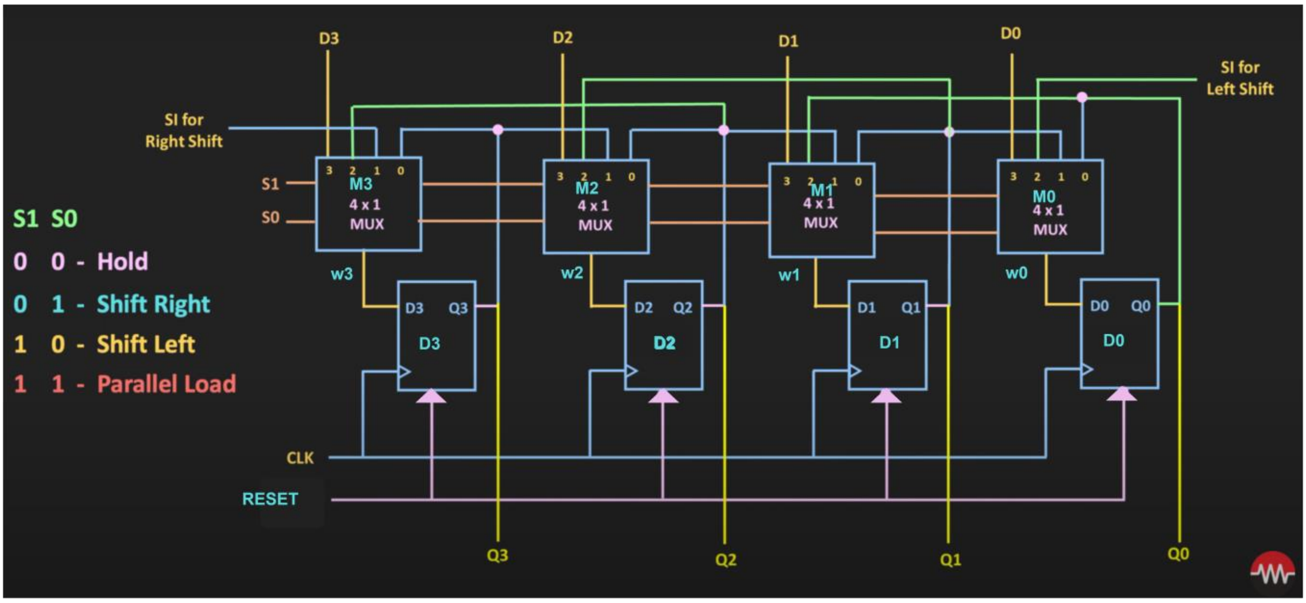


FIG: Block Diagram of USR

Verilog Code:

```
//Verilog Code for Mux
module mux_4x1(I0,I1,I2,I3,S,O);
input I0,I1,I2,I3;
input [1:0]S;
output reg O;

always@(*)
begin
case(S)
 2'b00 :O<=I0;
 2'b01 :O<=I1;
 2'b10 :O<=I2;
 2'b11 :O<=I3;
endcase
end
endmodule
```

```
// Verilog Code for D Flip Flop
module D_Flip_Flop(D, reset, clk, Q);
  input D, reset, clk;
  output reg Q;

  always @(posedge clk,posedge reset)
  begin
    if(reset)
       Q=0;
    else
      Q=D;
  end
endmodule
```

```
//Verilog Code for USR
module USR(D,S,clk,reset,SR,SL,Q);

input [3:0]D;
input [1:0]S;
input clk,reset,SR,SL;

output [3:0]Q;

wire [3:0]w;

mux_4x1 M3(.I0(Q[3]),.I1(SR),.I2(Q[2]),.I3(D[3]),.S(S),.O(w[3]));
mux_4x1 M2(.I0(Q[2]),.I1(Q[3]),.I2(Q[1]),.I3(D[2]),.S(S),.O(w[2]));
mux_4x1 M1(.I0(Q[1]),.I1(Q[2]),.I2(Q[0]),.I3(D[1]),.S(S),.O(w[1]));
mux_4x1 M0(.I0(Q[0]),.I1(Q[1]),.I2(SL),.I3(D[0]),.S(S),.O(w[0]));

D_Flip_Flop D3(.D(w[3]),.reset(reset),.clk(clk),.Q(Q[3]));
D_Flip_Flop D2(.D(w[2]),.reset(reset),.clk(clk),.Q(Q[2]));
D_Flip_Flop D1(.D(w[1]),.reset(reset),.clk(clk),.Q(Q[1]));
D_Flip_Flop D0(.D(w[0]),.reset(reset),.clk(clk),.Q(Q[0]));

endmodule
```

Testbench Code:

```
//Testbench code for USR
module USR_tb();

reg [3:0]D;
reg [1:0]S;
reg clk,reset,SR,SL;

wire [3:0]Q;

USR dut(.D(D),.S(S),.clk(clk),.reset(reset),.SR(SR),.SL(SL),.Q(Q));

initial begin
clk=1'b0;
reset=1'b1;
SR=1'b0;
SL=1'b0;
S=2'b00;        //Hold State
D=4'b0000;
#8 reset=1'b0;
#16 S=2'b01;    //Shifting Value Right
    SR=1'b1;
#8  SR=1'b1;
#8  SR=1'b1;
#8  SR=1'b1;

#16 S=2'b10;    //Shifting Value Left
    SL=1'b0;
```

```
#8  SL=1'b0;

#8  SL=1'b0;

#8  SL=1'b0;



#16 S=2'b11;     //Hold

    D=4'b1010;


//ADD More Test Cases Here

#32 $finish;

end


always #5 clk=~clk;


endmodule
```

Schematic:

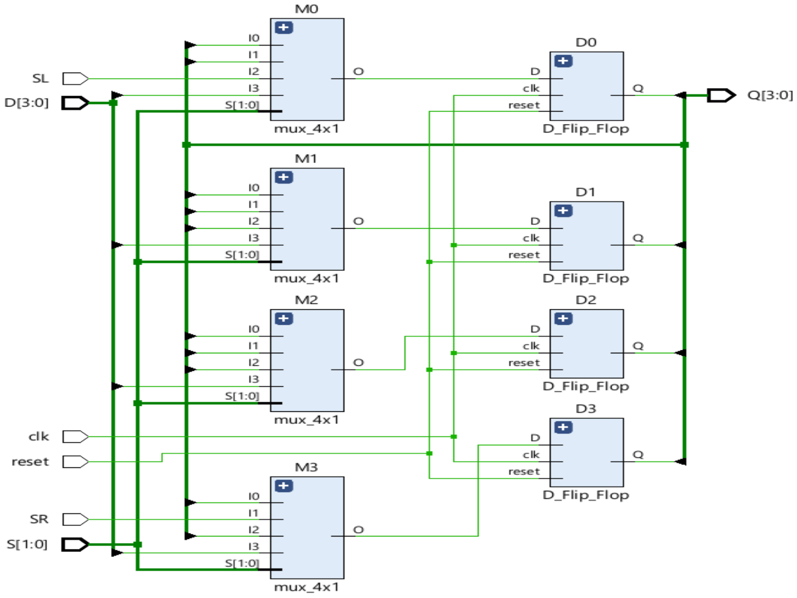


Fig: Schematic of USR

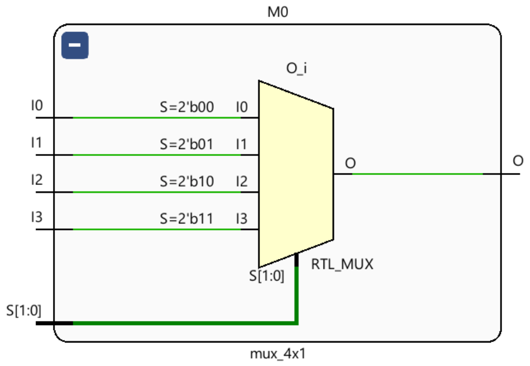


Fig: Schematic of 4x1 Mux

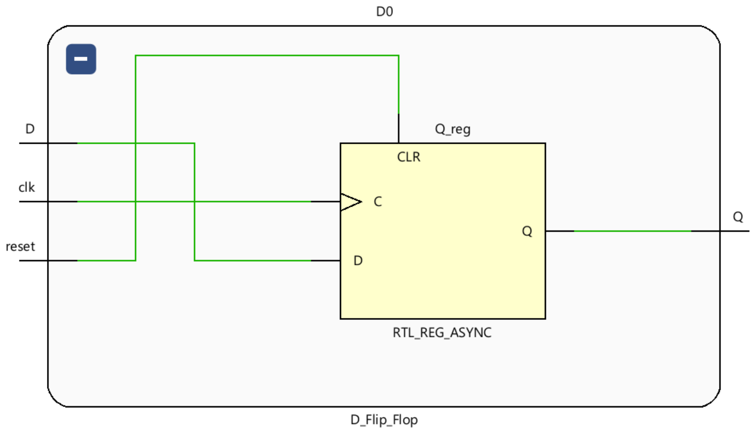
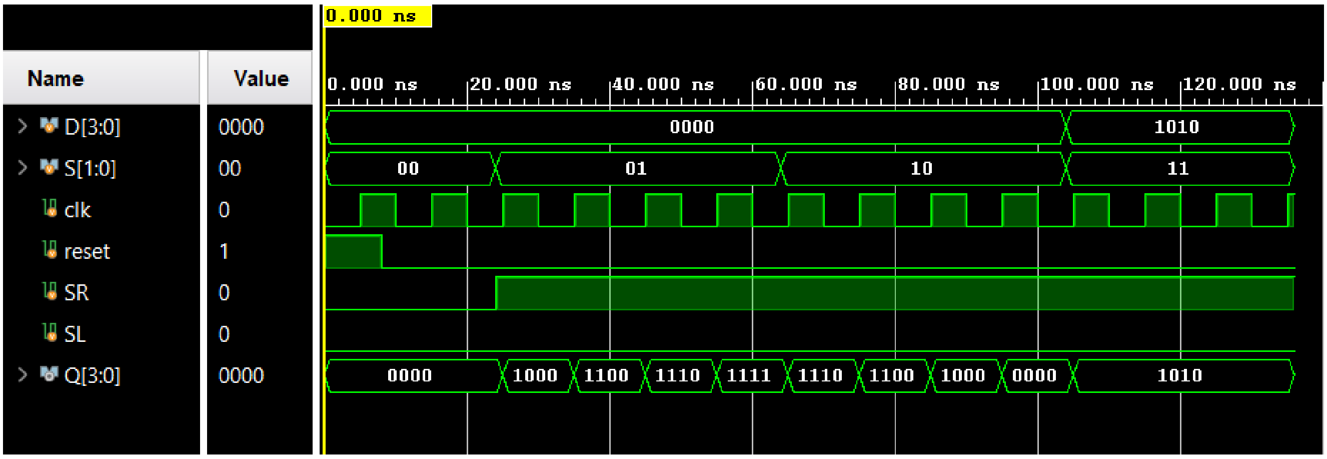


Fig: Schematic of D Flip Flop

Simulation Output:



GitHub Repository URL: https://github.com/tusharshenoy/RTL-Day-24-USR