





 [23008859](#) / [SR-FLIPFLOP-USING-CASE](#) Public

forked from [naavaneetha/SR-FLIPFLOP-USING-CASE](#)

 GPL-3.0 license

 0 stars  69 forks  Branches  Tags  Activity

 Star



 Notifications

 **Code**  Pull requests  Actions  Projects  Security  Insights

main ▾

1 Branch

0 Tags

Go to file

Go to file

Code

⋮

This branch is **8 commits ahead of** [naavaneetha/SR-FLIPFLOP-USING-CASE:main](#) .

23008859 Update README.md now

db	EXP6	3 months ago
incremental_db	EXP6	3 months ago
output_files	EXP6	3 months ago
simulation	EXP6	3 months ago
DIGITAL EX NO 6.pdf	Add files via upload	1 hour ago
LICENSE	Initial commit	3 months ago
README.md	Update README.md	now
SRFLIPFLOPUSINGCASE....	EXP6	3 months ago
SRFLIPFLOPUSINGCASE....	EXP6	3 months ago
SRFLIPFLOPUSINGCASE....	EXP6	3 months ago
SRFLIPFLOPUSINGCASE.v	Update SRFLIPFLOPUSINGCASE.v	3 months ago
SRFLIPFLOPUSINGCASE.v...	EXP6	3 months ago
Waveform.vwf	EXP6	3 months ago

README License

SR-FLIPFLOP-USING-CASE

AIM:

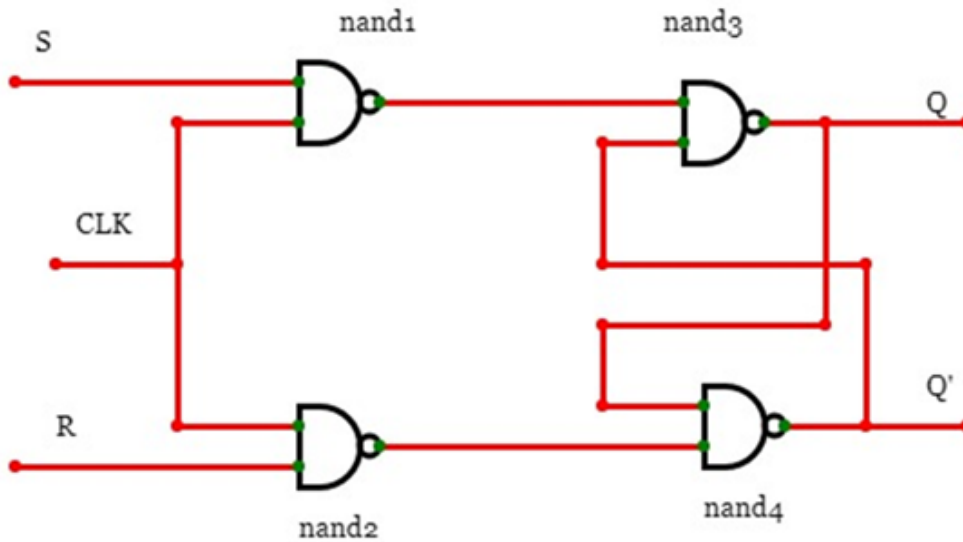
To implement SR flipflop using verilog and validating their functionality using their functional tables

SOFTWARE REQUIRED:

Quartus prime

THEORY

SR Flip-Flop SR flip-flop operates with only positive clock transitions or negative clock transitions. Whereas, SR latch operates with enable signal. The circuit diagram of SR flip-flop is shown in the following figure.



This circuit has two inputs S & R and two outputs Q_t & Q_t' . The operation of SR flipflop is similar to SR Latch. But, this flip-flop affects the outputs only when positive transition of the clock signal is applied instead of active enable. The following table shows the state table of SR flip-flop.

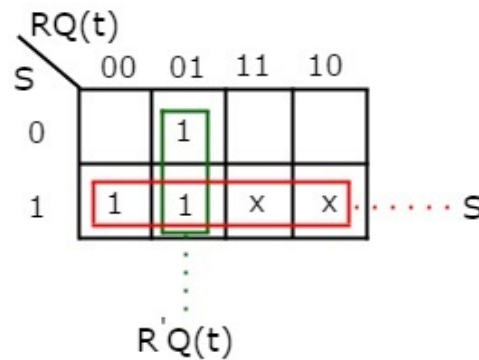
S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	-

Here, Q_t & Q_{t+1} are present state & next state respectively. So, SR flip-flop can be used for one of these three functions such as Hold, Reset & Set based on the input conditions, when positive transition of clock signal is applied. The following table shows the characteristic table of SR

flip-flop. Present Inputs Present State Next State

Present Inputs		Present State	Next State
S	R	$Q(t)$	$Q(t+1)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	x
1	1	1	x

By using three variable K-Map, we can get the simplified expression for next state, Q_{t+1} . The three variable K-Map for next state, Q_{t+1} is shown in the following figure.



The maximum possible groupings of adjacent ones are already shown in the figure. Therefore, the simplified expression for next state Q_{t+1} is $Q(t+1) = S + R'Q(t)Q(t+1) = S + R'Q(t)$

Procedure

1. Type the program in Quartus software.
2. Compile and run the program.
3. Generate the RTL schematic and save the logic diagram.
4. Create nodes for inputs and outputs to generate the timing diagram.
5. For different input combinations generate the timing diagram.

PROGRAM

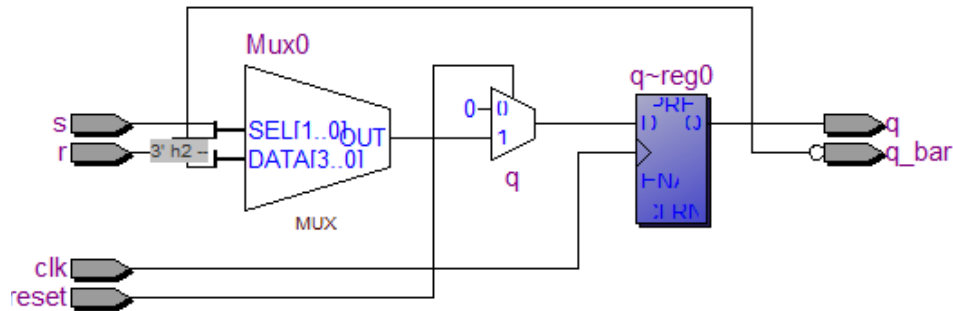
/* Program for flipflops and verify its truth table in quartus using Verilog programming. Developed by: ROSHINI S RegisterNumber: 212223230174 */

```
module ex6(q, q_bar, s,r, clk, reset);//SR Flip Flop Behavioral Level using 'case'
    input s,r,clk, reset;
    output reg q;
    output q_bar;

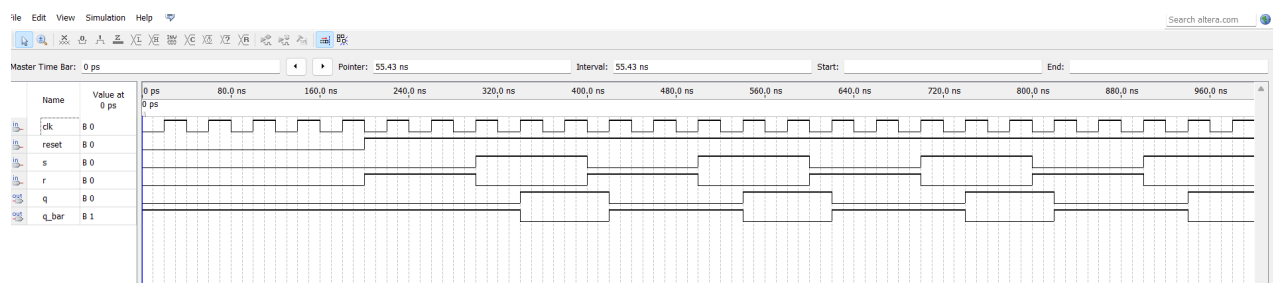
    always@(posedge clk) begin // for synchronous reset
        if(!reset)      q <= 0;
        else
            begin
                case({s,r})
                    2'b00: q <= q;
                    2'b01: q<=1'b0;
                    2'b10: q<=1'b1;
                    2'b11: q<=1'bx;
                endcase
            end
        end
        assign q_bar = ~q;
    endmodule
```



RTL LOGIC FOR FLIPFLOPS



TIMING DIGRAMS FOR FLIP FLOPS



RESULTS

To implement SR flipflop using verilog and validating their functionality using their functional tables was completed successfully.

Releases

No releases published

Packages

No packages published

Languages

● VHDL 45.2%
 ● Verilog 22.4%
 ● Stata 16.8%
 ● HTML 14.3%
 ● Standard ML 1.3%