23008859 / **JKFLIPFLOP-USING-IF-ELSE** Public

forked from naavaneetha/JKFLIPFLOP-USING-IF-ELSE

GPL-3.0 license

0 stars **64** 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 1 commit ahead of naavaneetha/JKFLIPFLOP-USING-IF-ELSE:main .

23008859 Update README.md 3 minutes ago

db	EXP7	3 months ago
incremental_db	EXP7	3 months ago
output_files	EXP7	3 months ago
simulation	EXP7	3 months ago
JKFLIPFLOPUSINGIFELSE....	EXP7	3 months ago
JKFLIPFLOPUSINGIFELSE....	EXP7	3 months ago
JKFLIPFLOPUSINGIFELSE....	EXP7	3 months ago
JKFLIPFLOPUSINGIFELSE.v	EXP7	3 months ago
JKFLIPFLOPUSINGIFELSE....	EXP7	3 months ago
LICENSE	Initial commit	3 months ago
README.md	Update README.md	3 minutes ago
Waveform1.vwf	EXP7	3 months ago

README License

JKFLIPFLOP-USING-IF-ELSE

AIM:

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

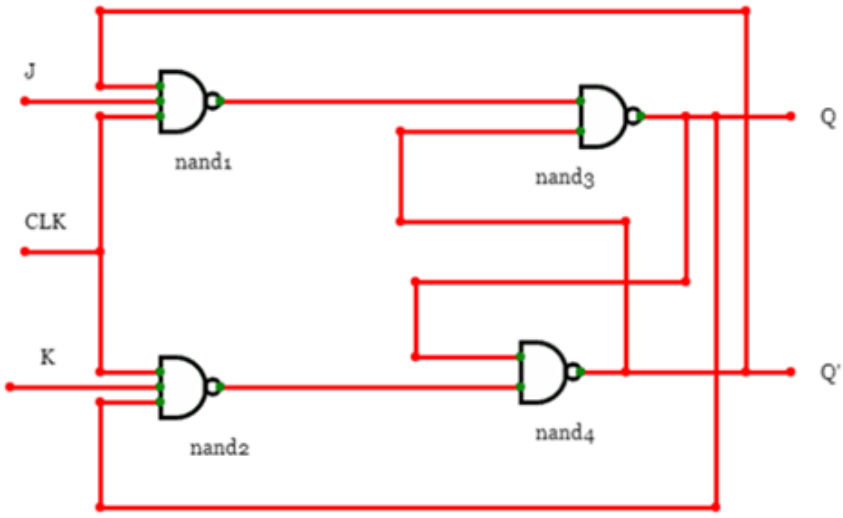
SOFTWARE REQUIRED:

Quartus prime

THEORY

JK Flip-Flop

JK flip-flop is the modified version of SR flip-flop. It operates with only positive clock transitions or negative clock transitions. The circuit diagram of JK flip-flop is shown in the following figure.



This circuit has two inputs J & K and two outputs Qtt & Qtt'. The operation of JK flip-flop is similar to SR flip-flop. Here, we considered the inputs of SR flip-flop as S = J Qtt' and R = KQtt in order to utilize the modified SR flip-flop for 4 combinations of inputs. The following table shows the state table of JK flip-flop.

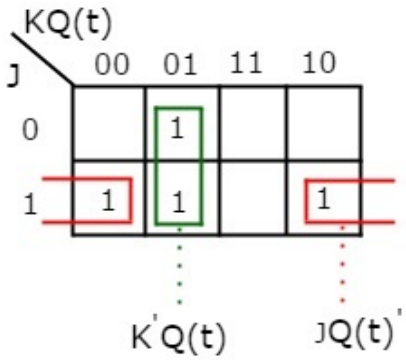
J	K	Q $t+1$
0	0	Q t
0	1	0
1	0	1
1	1	Q t '

Here, Qtt & Qt+1t+1 are present state & next state respectively. So, JK flip-flop can be used for one of these four functions such as Hold, Reset, Set & Complement of present state based on the input conditions, when positive transition of clock signal is applied. The following table shows the characteristic table of JK flip-flop. Present Inputs Present State Next State

Present Inputs		Present State	Next State
J	K		

		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	1
1	1	1	0

By using three variable K-Map, we can get the simplified expression for next state, Qt+1t+1. Three variable K-Map for next state, Qt+1t+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 Qt+1t+1 is Q(t+1)=JQ(t)′+K′Q(t)Q(t+1)=JQ(t)′+K′Q(t)

Procedure

1. Go to quartus software.
2. Set new environment.
3. Type the code to implement SR flipflop using verilog and validating their functionality using their functional tables.
4. Run the program.
5. Give inputs in the waveform table.
6. Run the program

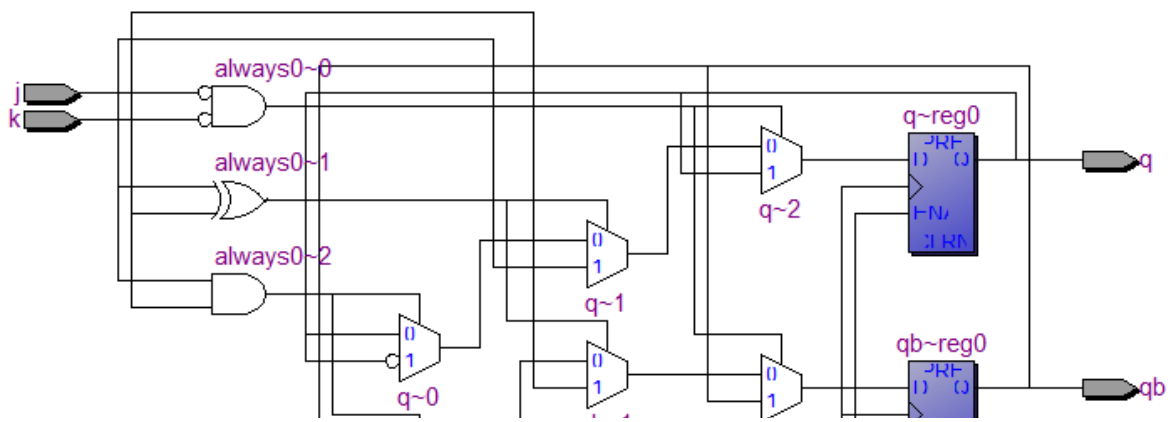
PROGRAM

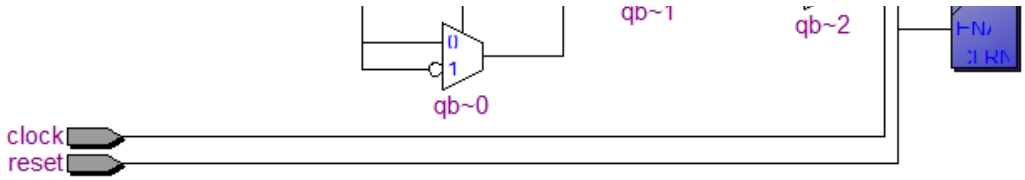
```
module exp7(q, qb,j,k,clock,reset);
```

```
    input j,k,clock,reset;
    output reg q, qb;

always @ (posedge (clock))

    begin
        if (!reset)
            begin
               q <= q;
               qb <=qb;
            end

else
        begin
                if(j==0&&k==0)
                        begin
                        q<=q;
                        qb<=qb;
                        end
                else if(j!=k)
                        begin
                        q<=j;
                        qb<=k;
                        end
                else if(j==1&&k==1)
                        begin
                        q<=~q;
                        qb<=~qb;
                        end
        end
end

endmodule
```

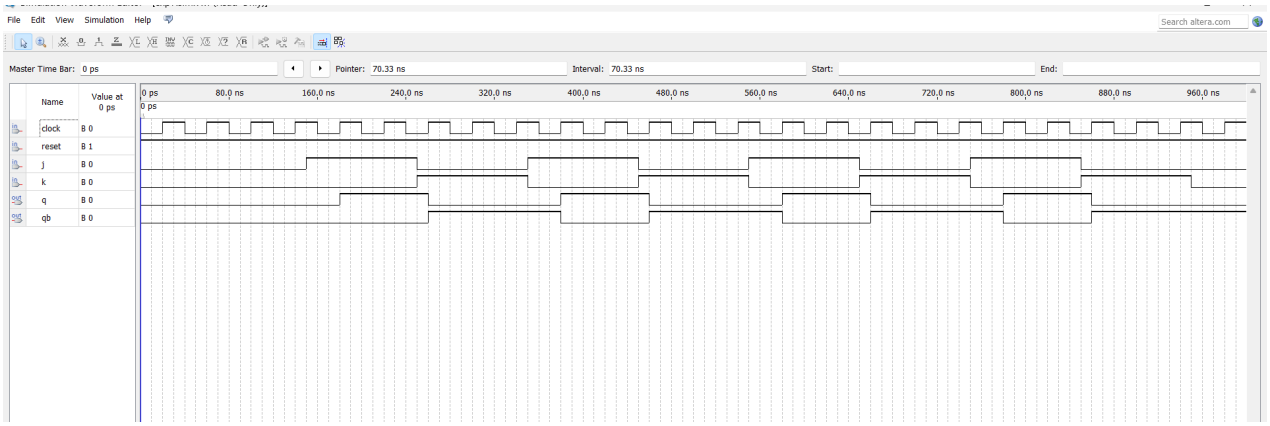
/* Program for flipflops and verify its truth table in quartus using Verilog programming. Developed by: ROSHINI S RegisterNumber: 212223230174 */

RTL LOGIC FOR FLIPFLOPS





TIMING DIGRAMS FOR FLIP FLOPS



RESULTS

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

Releases

No releases published

Packages

No packages published

Languages

VHDL 49.2% Stata 18.5% Verilog 15.6% HTML 15.3% Standard ML 1.4%