Sujithmano2706 / JKFLIPFLOP-USING-IF-ELSE

Code Pull requests Actions Projects Wiki Security Insights Settings

GPL-3.0 license

0 stars 2.3k forks 0 watching Branches Activity
Tags

Public repository · Forked from naavaneetha/JKFLIPFLOP-USING-IF-ELSE

Local Codespaces

Clone

HTTPS SSH GitHub CLI

https://github.com/Sujithmano2706/JKFLIPFLOP-US

Clone using the web URL.

Open with GitHub Desktop

Download ZIP

... 1 Branch 0 Tags Go to file

This branch is 2 commits ahead of naavaneetha/JKFLIPFLOP-USING-IF-ELSE:main .

Sujithmano2706 Add files via upload

db	EXP7	
incremental_db	EXP7	
output_files	EXP7	last year
simulation	EXP7	last year
JKFLIPFLOPUSINGIFELSE.qpf	EXP7	last year
JKFLIPFLOPUSINGIFELSE.qsf	EXP7	last year
JKFLIPFLOPUSINGIFELSE.qws	EXP7	last year
JKFLIPFLOPUSINGIFELSE.v	EXP7	last year
JKFLIPFLOPUSINGIFELSE.v.bak	EXP7	last year
LICENSE	Initial commit	last year
README.md	Update README.md	2 weeks ago
Waveform1.vwf	EXP7	last year
rtl.png	Add files via upload	5 days ago
waveform.png	Add files via upload	5 days ago

README License

JKFLIPFLOP-USING-IF-ELSE

AIM:

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

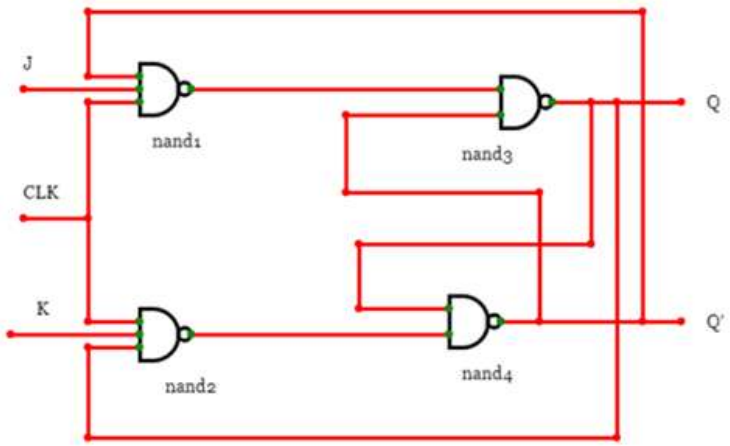
SOFTWARE REQUIRED:

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.



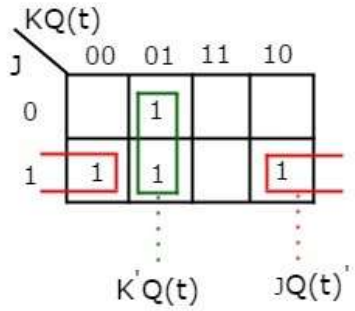
This circuit has two inputs J & K and two outputs Qtt & Qtt'. The operation of JK flip-flop is similar to SR flip-flop. Here, we considered the inputs of SR flip-flop as S = J Qtt' and R = KQtt in order to utilize the modified SR flip-flop for 4 combinations of inputs. The following table shows the state 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. Define Inputs/Outputs: Inputs: J (Set), K (Reset), c1k (clock); Outputs: q, qbar (~q).
2. Initialization: Set q = 0 and qbar = 1 at the start of the simulation.
3. JK Flip-Flop Logic: On posedge c1k, compute q
4. Complementary Output: Update qbar = ~q to maintain complementarity.
5. Testbench: Simulate with combinations of J, K, and c1k to verify JK Flip-Flop functionality.
```

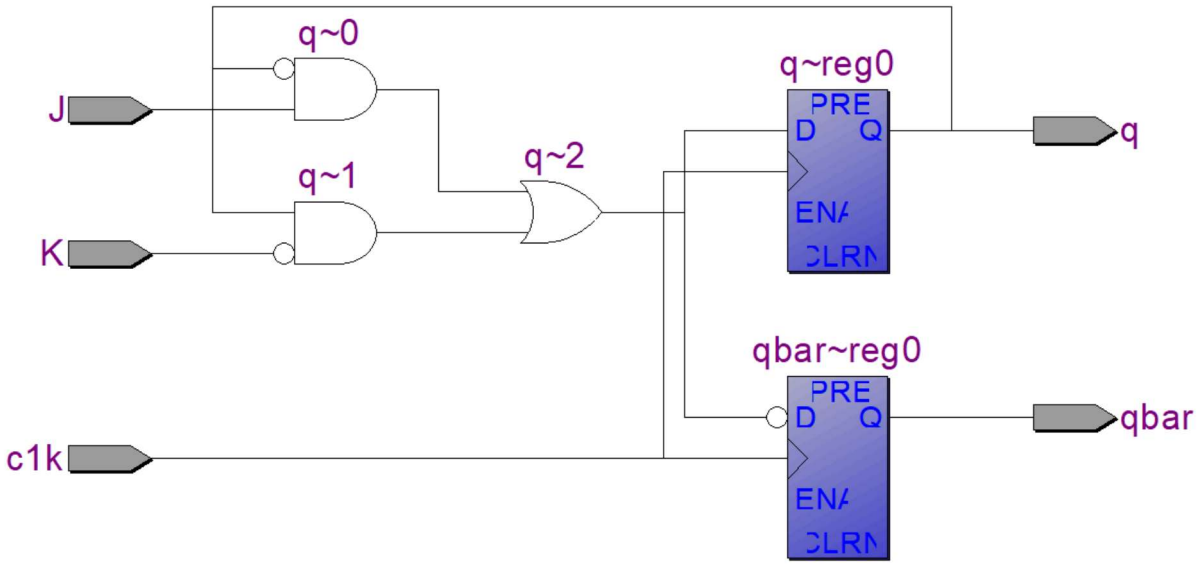
PROGRAM

```
/* Program for flipflops and verify its truth table in quartus using Verilog programming.
Developed by:SUJITH MANO M
RegisterNumber: 25018328
*/
```

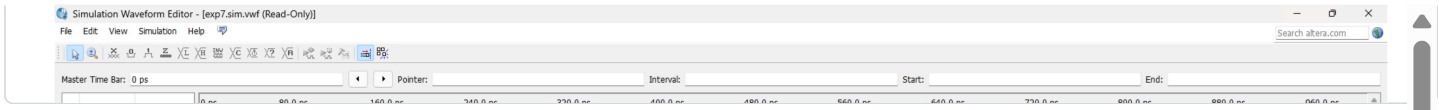
```
module exp7(J,K,c1k,q,qbar);
input J,K,c1k;
output reg q;
output reg qbar;
```

```
initial q=0;
initial qbar=1;
always @(posedge c1k)
begin
q=((J&(~q)))|((~K)&q);
qbar=~q;
end
endmodule
```

RTL LOGIC FOR FLIPFLOPS



TIMING DIGRAMS FOR FLIP FLOPS



Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

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

Suggested workflows

Based on your tech stack

Jekyll using Docker image Configure

Package a Jekyll site using the jekyll/builder Docker image.

SLSA Generic generator Configure

Generate SLSA3 provenance for your existing release workflows

More workflows Dismiss suggestions