EXPT.-06

AIM: - Design and Simulate D-Flip Flop and JK-Flip Flop Verilog HDL.

Software: - Xilinx Vivado 2019.1

Theory: -

D Flip Flop

In **SR NAND Gate Bistable** circuit, the undefined input condition of SET = "0" and RESET = "0" is forbidden. It is the drawback of the SR flip flop. This state:

1. Override the feedback latching action.
2. Force both outputs to be 1.
3. Lose the control by the input, which first goes to 1, and the other input remains "0" by which the resulting state of the latch is controlled.

We need an **inverter** to prevent this from happening. We connect the inverter between the Set and Reset inputs for producing another type of flip flop circuit called **D flip flop**

, Delay flip flop, D-type Bistable, D-type flip flop.

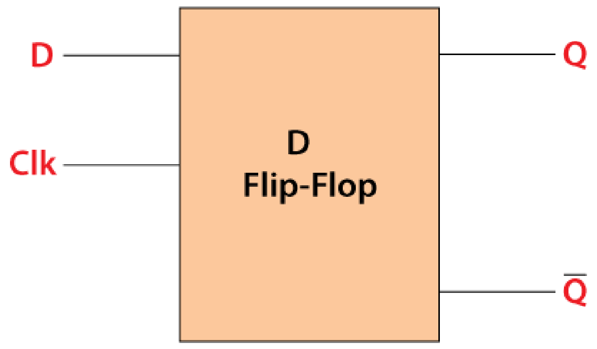
The D flip flop is the most important flip flop from other clocked types. It ensures that at the same time, both the inputs, i.e., S and R, are never equal to 1. The Delay flip-flop is designed using a gated SR flip-flop

with an inverter connected between the inputs allowing for a single input D(Data).

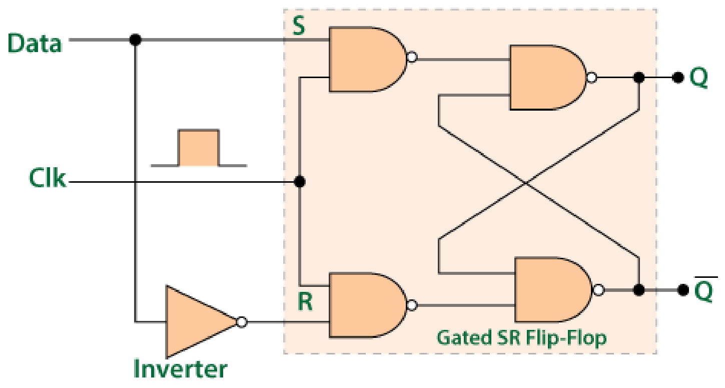
This single data input, which is labeled as "D" used in place of the "Set" input and for the complementary "Reset" input, the inverter is used. Thus, the level-sensitive D-type or D flip flop is constructed from a level-sensitive SR flip flop.

So, here S=D and R= ~D(complement of D)

Block Diagram



Circuit Diagram



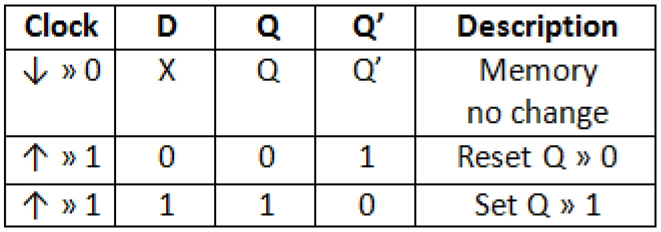
We know that the SR flip-flop requires two inputs, i.e., one to "SET" the output and another to "RESET" the output. By using an inverter, we can set and reset the outputs with only one input as now the two input signals complement each other. In SR flip flop, when both the inputs are 0, that state is no longer possible. It is an ambiguity that is removed by the complement in D-flip flop.

In D flip flop, the single input "D" is referred to as the "Data" input. When the data input is set to 1, the flip flop would be set, and when it is set to 0, the flip flop would change and become reset. However, this would be pointless since the output of the flip flop would always change on every pulse applied to this data input.

The "CLOCK" or "ENABLE" input is used to avoid this for isolating the data input from the flip flop's latching circuitry. When the clock input is set to true, the D input condition is only copied to the output Q. This forms the basis of another sequential device referred to as **D Flip Flop**.

When the clock input is set to 1, the "set" and "reset" inputs of the flip-flop are both set to 1. So it will not change the state and store the data present on its output before the clock transition occurred. In simple words, the output is "latched" at either 0 or 1.

Truth Table for the D-type Flip Flop



Clock	D	Q	Q'	Description
↓ » 0	X	Q	Q'	Memory no change
↑ » 1	0	0	1	Reset Q » 0
↑ » 1	1	1	0	Set Q » 1

Symbols ↓ and ↑ indicates the direction of the clock pulse. D-type flip flop assumed these symbols as edge-triggers.

JK Flip Flop

The SR Flip Flop or Set-Reset flip flop has lots of advantages. But, it has the following switching problems:

- When Set 'S' and Reset 'R' inputs are set to 0, this condition is always avoided.
- When the Set or Reset input changes their state while the enable input is 1, the incorrect latching action occurs.

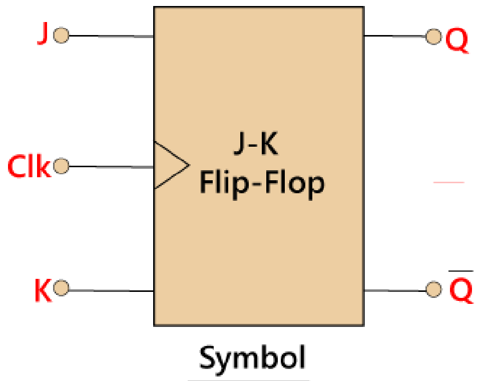
The JK Flip Flop removes these two drawbacks of SR Flip Flop.

The JK flip flop is one of the most used flip flops in digital circuits. The JK flip flop is a universal flip flop having two inputs 'J' and 'K'. In SR flip flop, the 'S' and 'R' are the shortened abbreviated letters for Set and Reset, but J and K are not. The J and K are themselves autonomous letters which are chosen to distinguish the flip flop design from other types.

The JK flip flop work in the same way as the SR flip flop work. The JK flip flop has 'J' and 'K' flip flop instead of 'S' and 'R'. The only difference between JK flip flop and SR flip flop is that when both inputs of SR flip flop is set to 1, the circuit produces the invalid states as outputs, but in case of JK flip flop, there are no invalid states even if both 'J' and 'K' flip flops are set to 1.

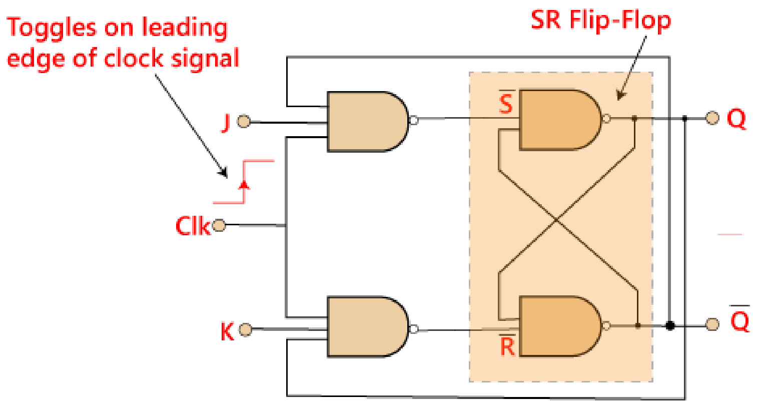
The JK Flip Flop is a gated SR flip-flop having the addition of a clock input circuitry. The invalid or illegal output condition occurs when both of the inputs are set to 1 and are prevented by the addition of a clock input circuit. So, the JK flip-flop has four possible input combinations, i.e., 1, 0, "no change" and "toggle". The symbol of JK flip flop is the same as **SR Bistable Latch** except for the addition of a clock input.

Block Diagram:



Symbol

Circuit Diagram:



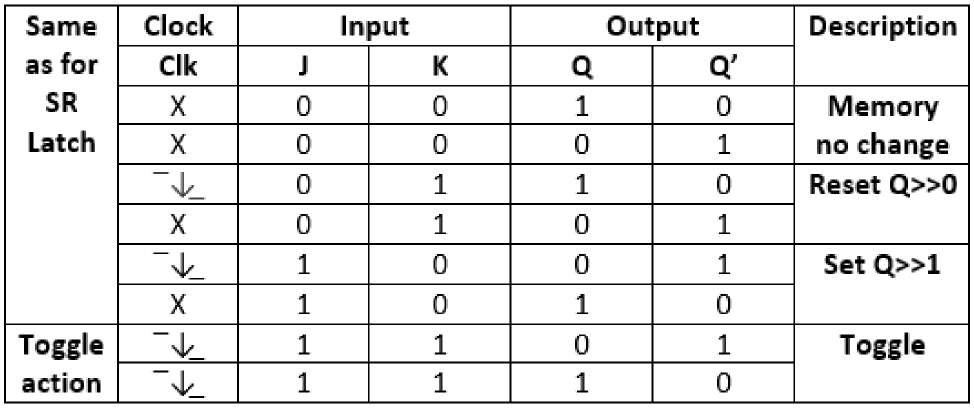
Circuit

In SR flip flop, both the inputs 'S' and 'R' are replaced by two inputs J and K. It means the J and K input equates to S and R, respectively.

The two 2-input AND gates are replaced by two 3-input NAND gates. The third input of each gate is connected to the outputs at Q and Q'. The cross-coupling of the SR flip-flop permits the previous invalid condition of (S = "1", R = "1") to be used to produce the "toggle action" as the two inputs are now interlocked.

If the circuit is "set", the J input is interrupted from the "0" position of Q' through the lower NAND gate. If the circuit is "RESET", K input is interrupted from 0 positions of Q through the upper NAND gate. Since Q and Q' are always different, we can use them to control the input. When both inputs 'J' and 'K' are set to 1, the JK toggles the flip flop as per the given truth table.

Truth Table:



Same as for SR Latch	Clock	Input		Output		Description
	Clk	J	K	Q	Q’	
	X	0	0	1	0	Memory no change
	X	0	0	0	1	
	¯↓_	0	1	1	0	Reset Q>>0
	X	0	1	0	1	
	¯↓_	1	0	0	1	Set Q>>1
	X	1	0	1	0	
Toggle action	¯↓_	1	1	0	1	Toggle
	¯↓_	1	1	1	0	

When both of the inputs of JK flip flop are set to 1 and clock input is also pulse "High" then from the SET state to a RESET state, the circuit will be toggled. The JK flip flop work as a T-type toggle flip flop when both of its inputs are set to 1.

The JK flip flop is an improved clocked SR flip flop. But it still suffers from the **"race"** problem. This problem occurs when the state of the output Q is changed before the clock input's timing pulse has time to go **"Off"**. We have to keep short timing plus period (T) for avoiding this period.

Code: -

```
//DFF
`timescale 1ns / 1ps

module D_FF(q,d,clk);
input d,clk;
output reg q;

always @(posedge clk)
q<=d;

endmodule
```

```
//tb DFF
`timescale 1ns / 1ps

module tb_D_FF();
reg d,clk;
wire q;
integer i;

D_FF DF1(.q(q),.d(d),.clk(clk));

initial
begin
clk=1'b0;
forever #3 clk=~clk;
end

initial
begin
d=1'b0;
#10;
for(i=0;i<5;i=i+1)
begin
d=~d;
#10;
end
end

initial
begin
$monitor($time,"q=%b,d=%b,clk=%b",q,d,clk);
#60 $finish;
end
endmodule
```

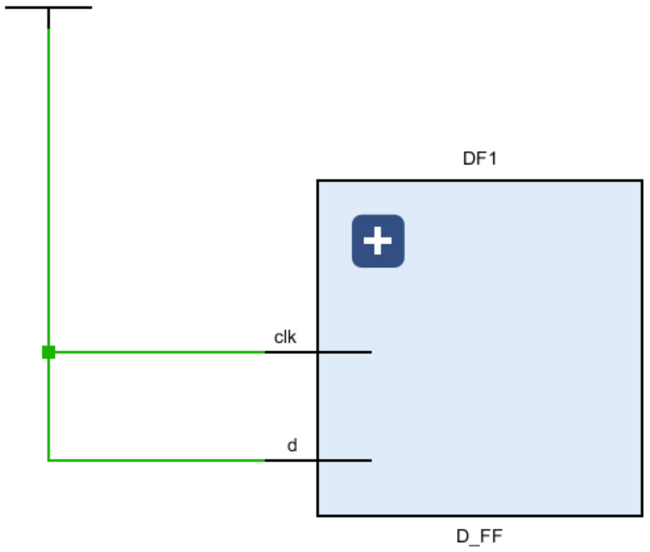
```
//JKFF
`timescale 1ns / 1ps

module JKFF(q,qb,j,k,clk);
input j,k,clk;
output reg q,qb;
always @(posedge clk)
case({j,k})
2'b01: q<=0;
2'b10: q<=1;
2'b11:q<=~q;
endcase
endmodule
```

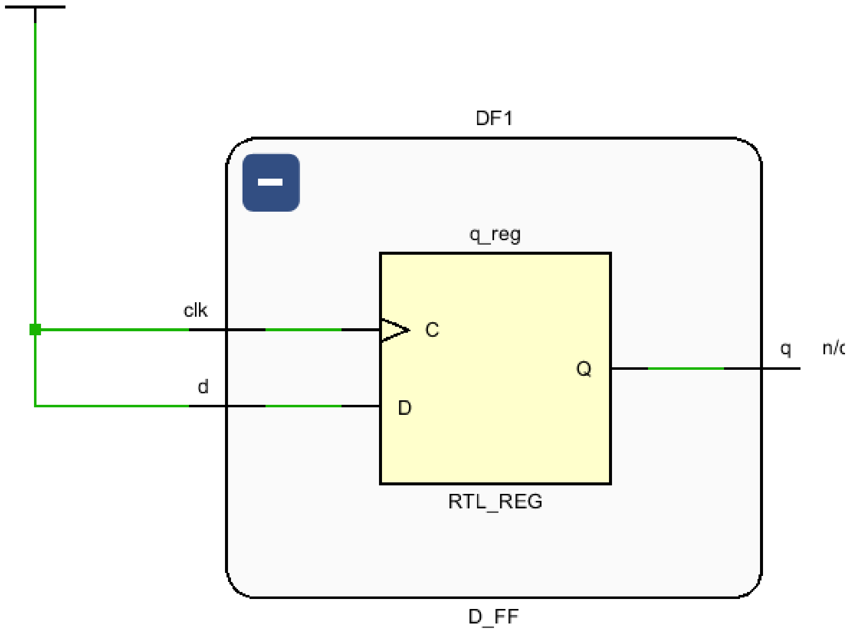
```
//tb JKFF
`timescale 1ns / 1ps

module tb_JKFF();
reg j,k,clk;
wire q;

JKFF FF1(.q(q),.j(j),.k(k),.clk(clk));

integer i=4;
initial
begin
{j,k}=2'b00;
repeat(2)
begin
for(i=0;i<4;i=i+1)
begin
{j,k}=i;
#10;
end
end
end

initial
$monitor($time,"q=%b,j=%b,k=%b,clk=%b",q,j,k,clk);

initial
begin
clk=1'b0;
forever #3 clk<=~clk;
end

initial
#50 $finish;
endmodule
```

Schematics and Simulated Waveforms: -

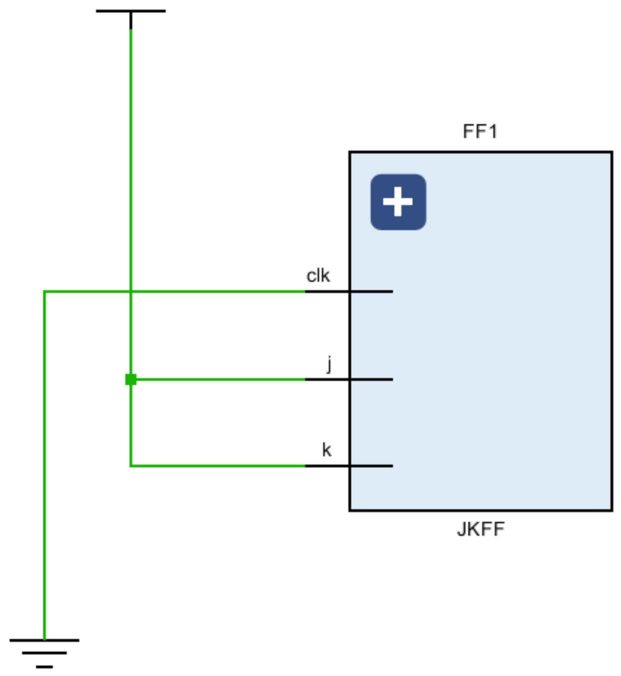
Schematic:

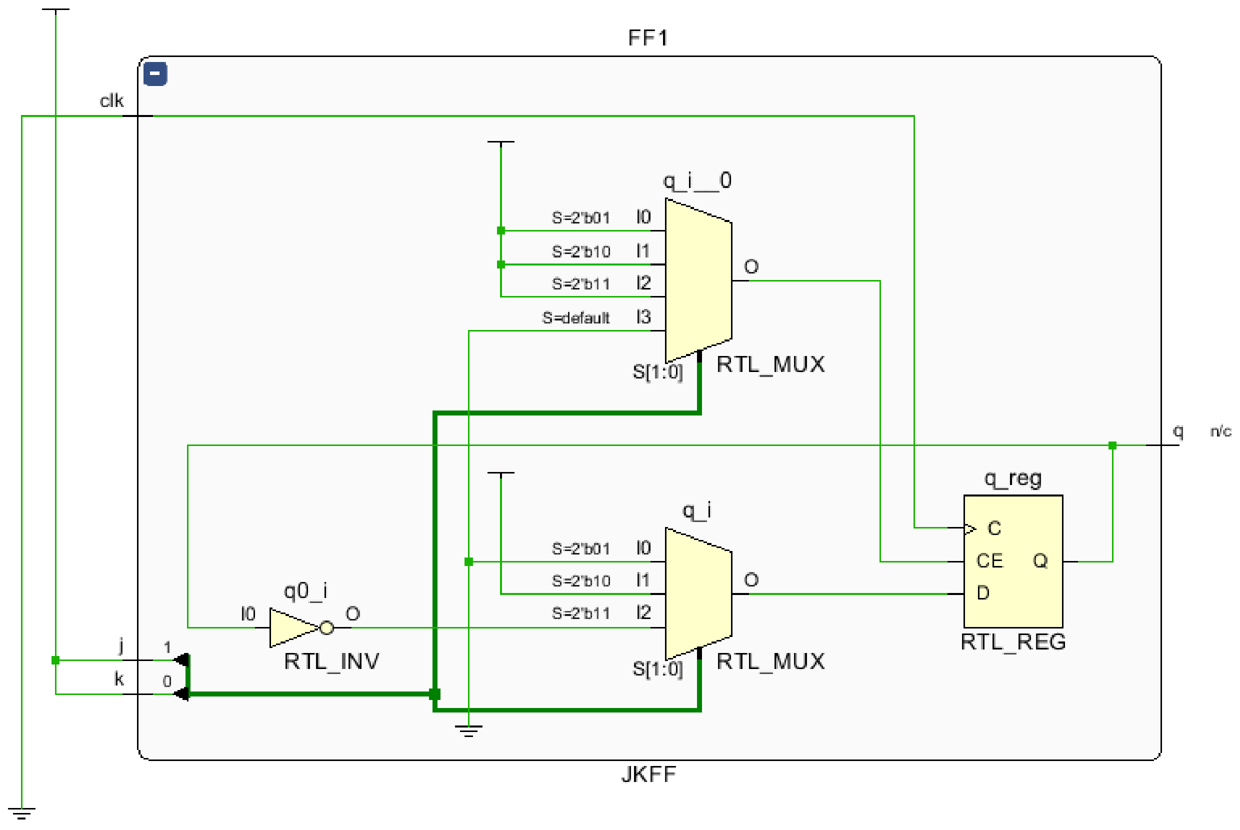
DFF





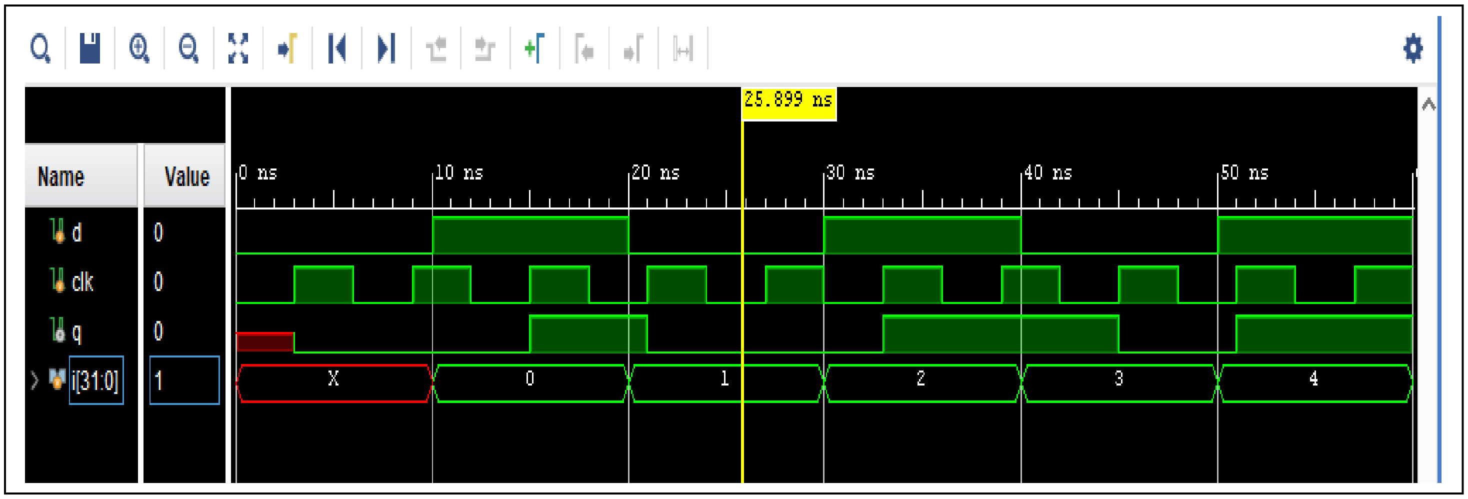
JKFF


FF1
clk
j
k
JKFF

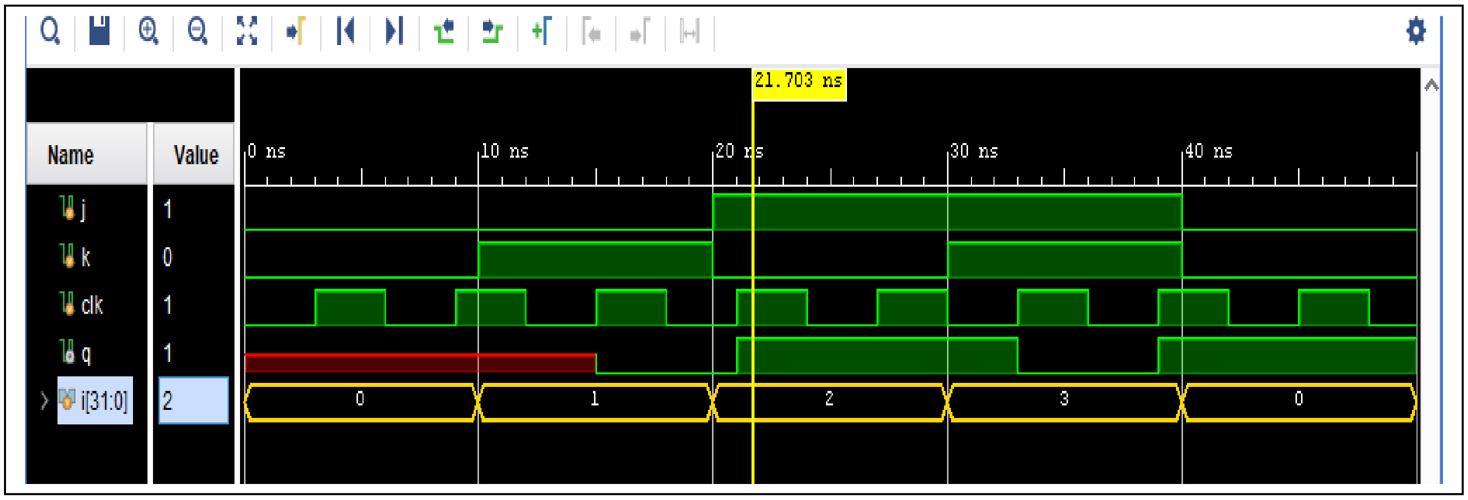

FF1
clk
q_i__0
S=2'b01 I0
S=2'b10 I1
S=2'b11 I2
S=default I3
O
S[1:0]
RTL_MUX
q
n/c
q_reg
C
CE
Q
D
RTL_REG
q_i
S=2'b01 I0
S=2'b10 I1
S=2'b11 I2
O
S[1:0]
RTL_MUX
q0_i
I0
O
RTL_INV
j
k
1
0
JKFF

Waveform:

DFF



JKFF



Conclusion: -

Hence , we have successfully designed and simulated D-Flip Flop and JK-Flip Flop using behavioral modeling style in Verilog HDL. Also verified simulated waveforms with actual truth tables using Xilinx Vivado 2019.1