Worksheet of Experiment

Date of Experiment	24 sep, 2024
Name	Akshada Jotiram Ghorpade.
Registration Number	12302956.

Title of Experiment: Develop RTL designs in verilog to implement JK flipflop and D flipflop verify their functionality using cadence NCSIM.

Name of Language: Verilog.

Name of Software: Cadence NCSIM, simvision.

Theory of Experiment:

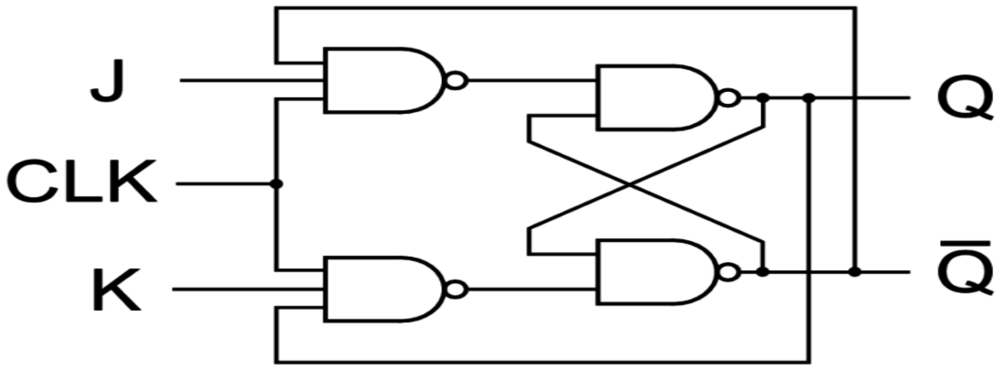
Flipflop:

- A flip-flop in digital electronics is a circuit with two stable states that can be used to store binary data.
- The stored data can be changed by applying varying inputs.
- Flip-flops and latches are fundamental building blocks of digital electronics systems used in computers, communications, and many other types of systems.
- Both are used as data storage elements.
- There are 4 types of flip-flops in digital electronics:

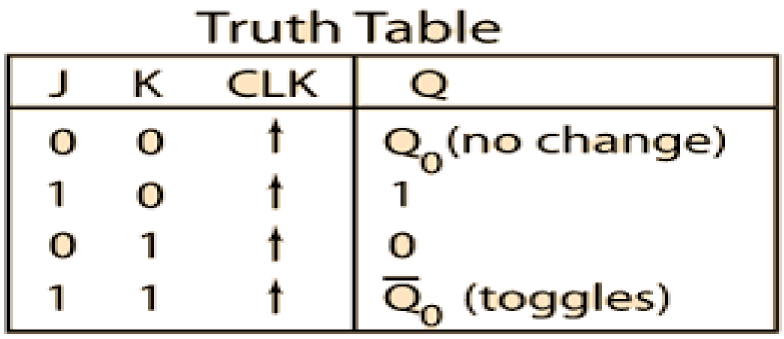
1. SR Flip-Flop
2. JK Flip-Flop
3. D Flip-Flop
4. T Flip-Flop

- **JK Flip-Flop**
  - Due to the undefined state in the SR flip-flops, another flip-flop is required in electronics. The JK flip-flop is an improvement on the SR flip-flop where S=R=1 is not a problem.

Circuit Diagram



Truth table:



Truth Table

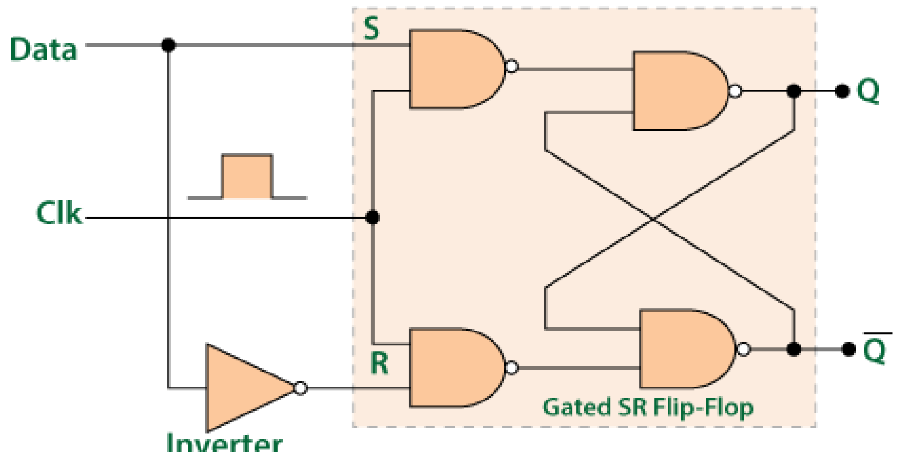
J	K	CLK	Q
0	0	↑	Q_0 (no change)
1	0	↑	1
0	1	↑	0
1	1	↑	$\overline{Q}_0$ (toggles)

The JK Flip Flop contains 4 states:

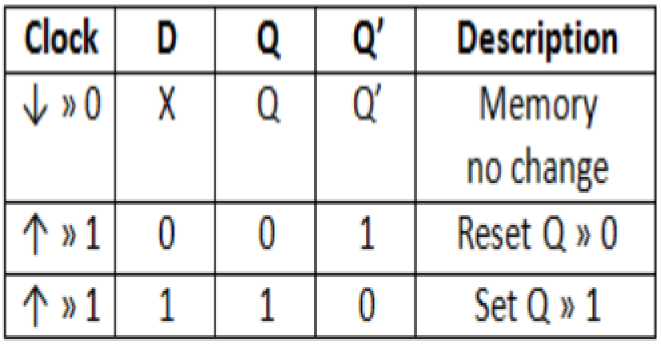
- set (Q = 1)
- reset (Q = 0)
- toggle (Q changes based on the input)
- no change (Q maintains its current value)

- **D Flipflop:**
  - 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.

Circuit Diagram



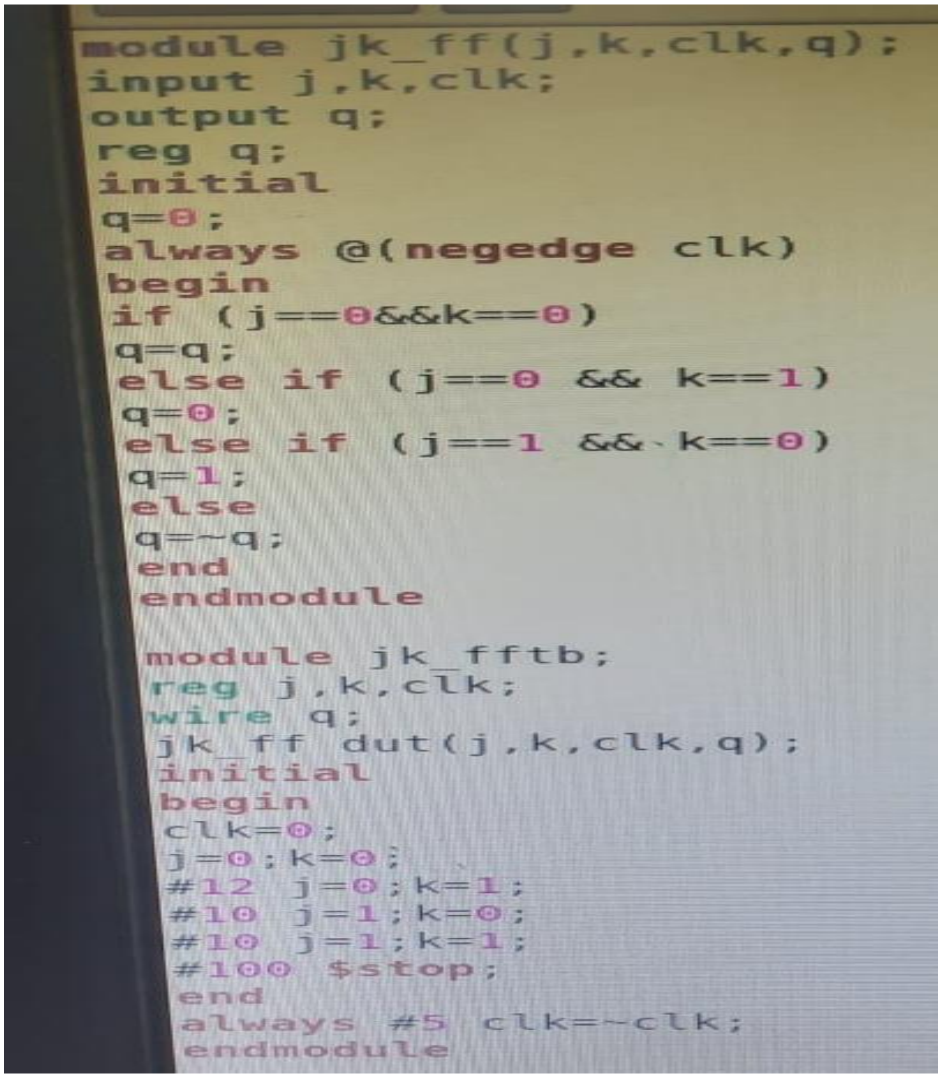
Truth table



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

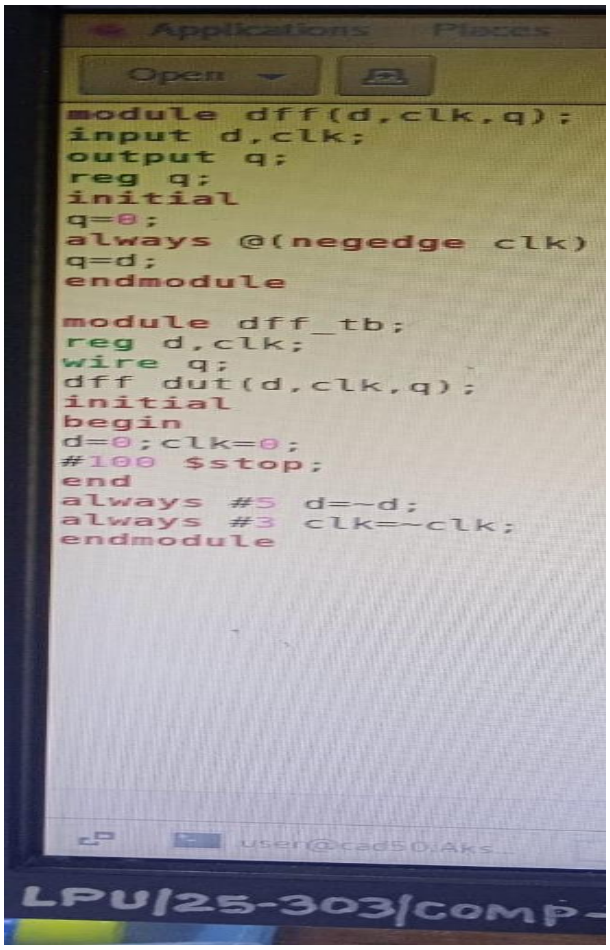
Program and Stimulus (Testbench) program:

PROGRAM (JK Flipflop):



```
module jk_ff(j,k,clk,q);
input j,k,clk;
output q;
reg q;
initial
q=0;
always @(negedge clk)
begin
if (j==0&&k==0)
q=q;
else if (j==0 && k==1)
q=0;
else if (j==1 && k==0)
q=1;
else
q=~q;
end
endmodule

module jk_fftb;
reg j,k,clk;
wire q;
jk_ff dut(j,k,clk,q);
initial
begin
clk=0;
j=0;k=0;
#12 j=0;k=1;
#10 j=1;k=0;
#10 j=1;k=1;
#100 $stop;
end
always #5 clk=~clk;
endmodule
```

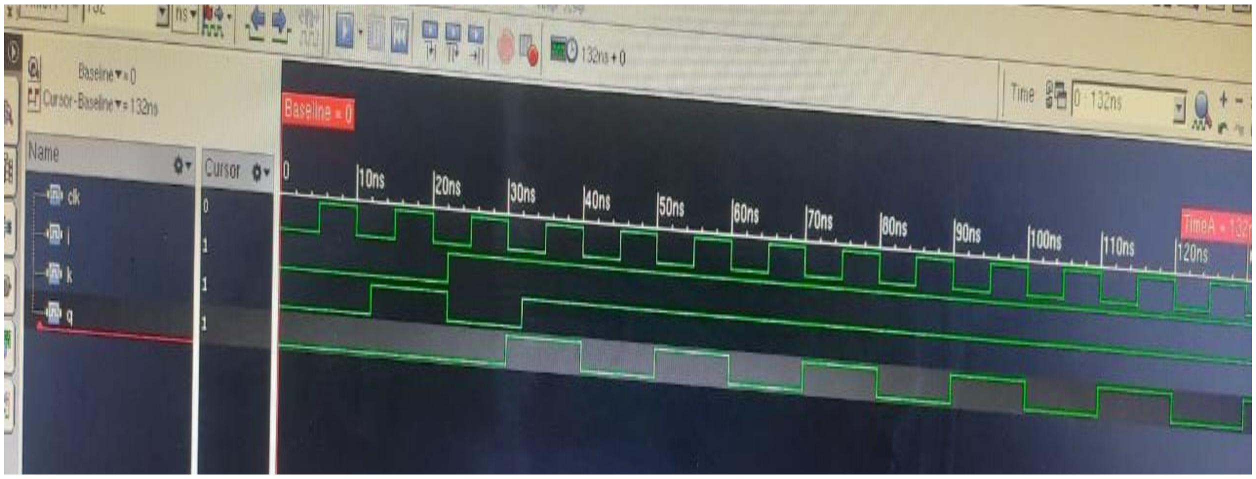
PROGRAM (D Flip flop):



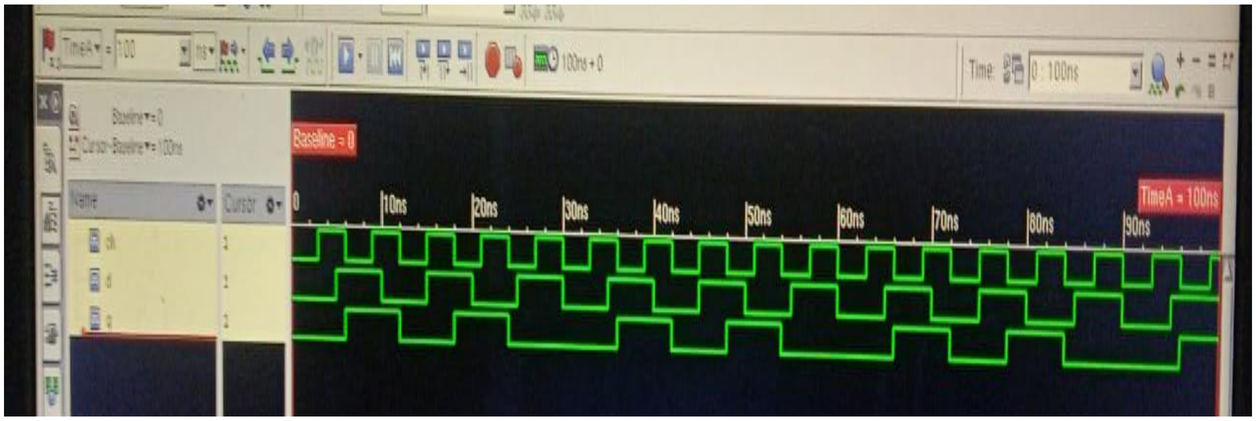
```
module dff(d,clk,q);
input d,clk;
output q;
reg q;
initial
q=0;
always @(negedge clk)
q=d;
endmodule

module dff_tb;
reg d,clk;
wire q;
dff dut(d,clk,q);
initial
begin
d=0;clk=0;
#100 $stop;
end
always #5 d=~d;
always #3 clk=~clk;
endmodule
```

Simulation waveform:

JK Flipflop

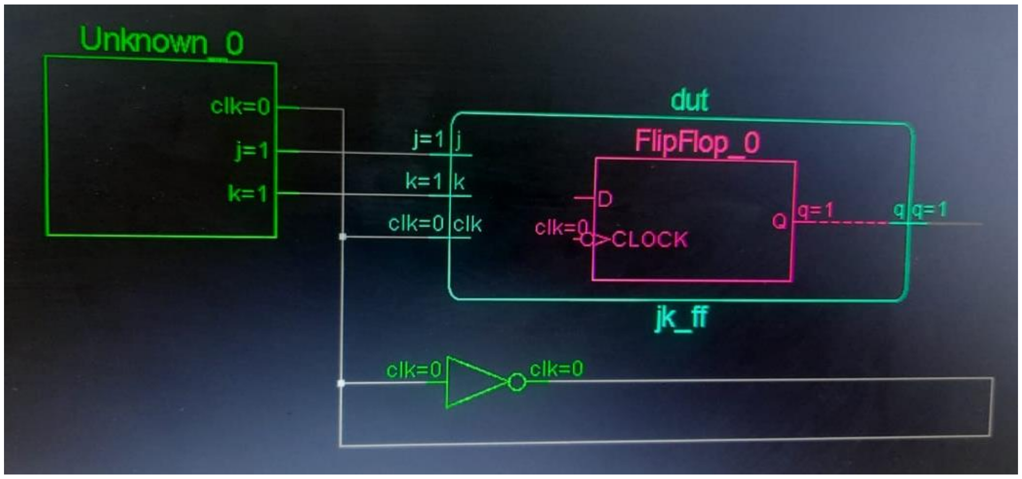


D Flip flop

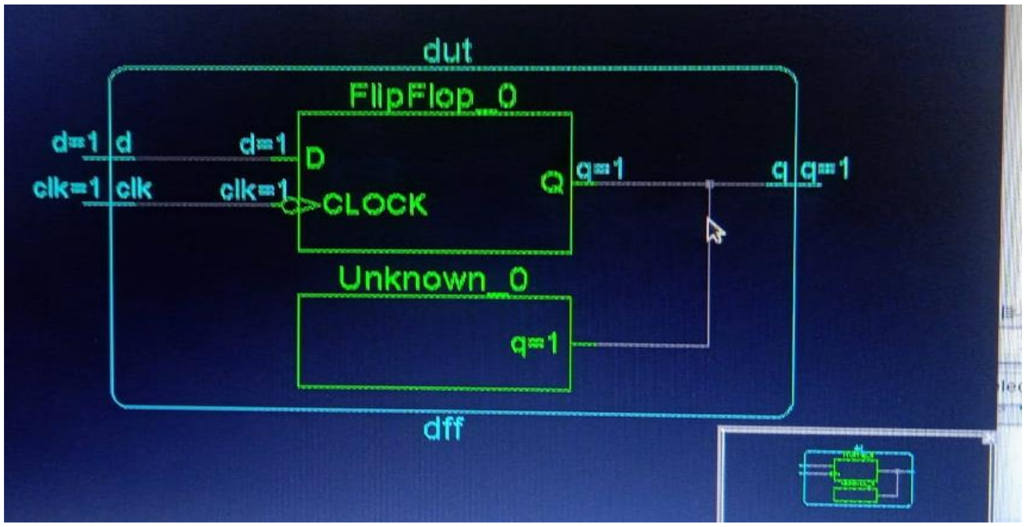


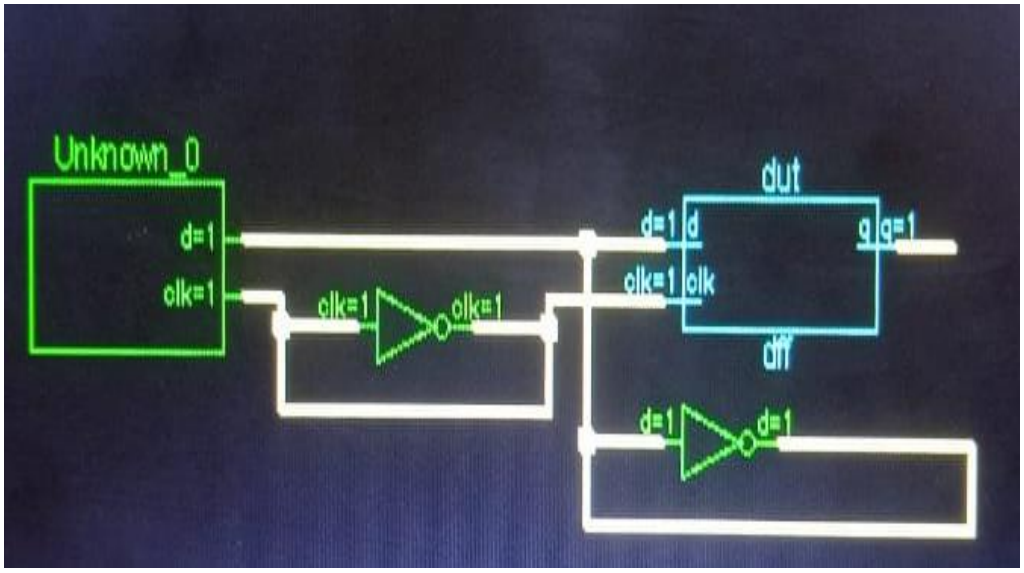
RTL Schematic:

JK Flipflop



D Flip flop





Learning Outcome:

1. Learn to write syntactically correct Verilog code for JK Flipflop and D Flip flop.

2. Create testbenches to simulate and verify the functionality of JK Flipflop and D Flip flop.

3. Apply techniques to optimize logic circuits for speed, area, and power efficiency.

4. Perform timing analysis to ensure circuits meet required timing constraints.

5. Analyze the performance of JK Flipflop and D Flip flop and circuits in terms of timing, power, and area using simulation results.