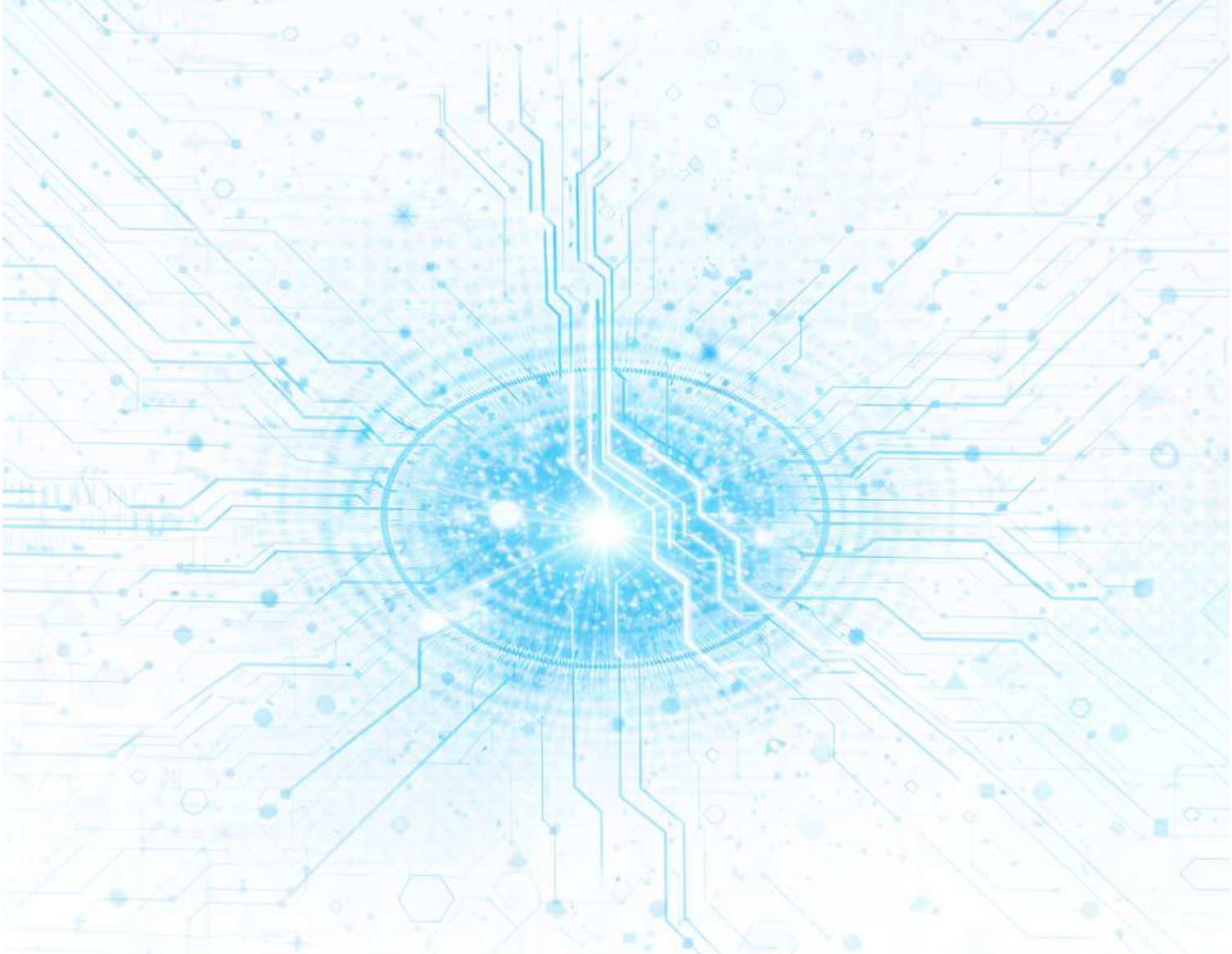
MASTERING
FLIP-FLOPS

SR · JK · D · T – Truth Tables,
Characteristic & Excitation Tables



SR JK D T

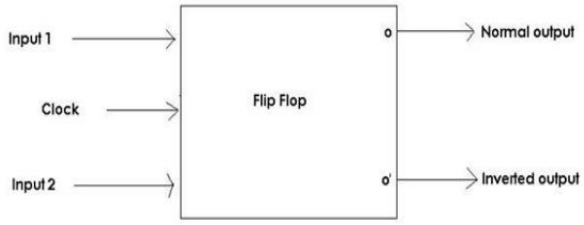
Flip Flop

A flip-flop is a clock-controlled bistable sequential circuit that can store one bit of information (0 or 1) and changes its output only on the active edge of the clock.

It remembers the last value until the next clock

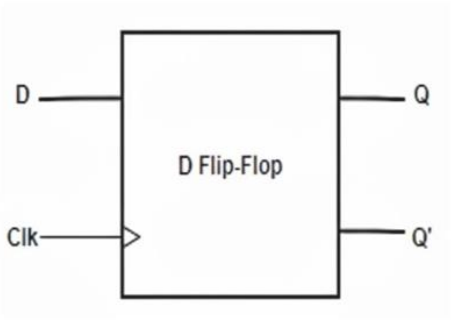
It is used to build:

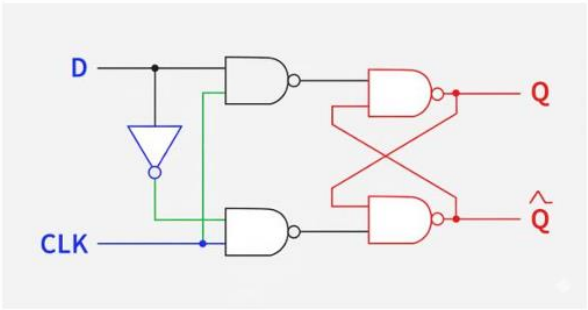
- Registers
- Counters
- Memory
- State machines (FSM)



D Flip Flop

In D flip flop D stands for Data or delay. It stores the value of D at the clock edge.





Truth table:

D	Qn+1
0	0
1	1

Characteristic table:

D	Qn	Qn+1
0	0	0
0	1	0
1	0	1
1	1	1

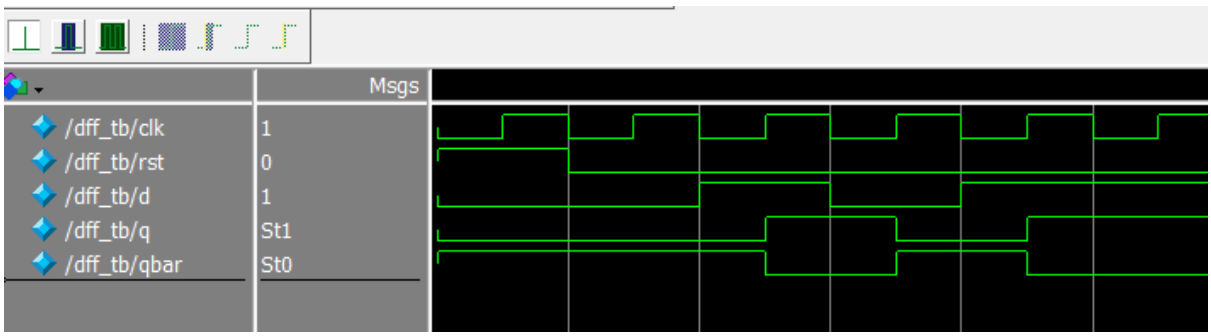
Excitation table:

Qn	Qn+1	D
0	0	0
0	1	1
1	0	0
1	1	1

Characteristic equation: Qn+1 = D

```
module d_flipflop(
input d,
input clk,rst,
output reg q,qbar
);
always @(posedge clk or posedge rst)
begin
if(rst) begin
  q<=1'b0;
  qbar<=1'b1;
end
else begin
  q<=d;
  qbar<=~d;
end
end
endmodule
```

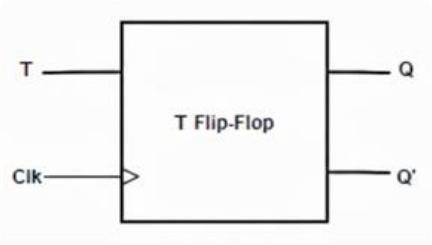
```
`timescale 1ns/100ps
module dff_tb;
reg d;
reg clk,rst;
wire q,qbar;

d_flipflop uut(d,clk,rst,q,qbar);
always #5 clk = ~clk;
initial begin
clk=0; d=0; rst=1;#10;

rst=0;
d=0;#10;
d=1;#10;
d=0;#10;
d=1;#10;

end
endmodule
```

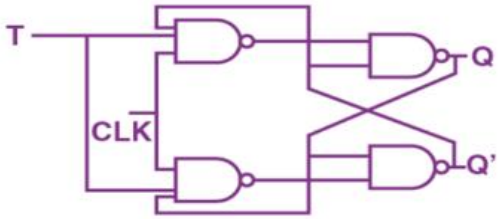


T Flip Flop

In T Flip flop T stands for toggle input

- If T = 0 → No change (Hold)
- If T = 1 → Toggle (Invert)





Truth table:

T	Qn+1
0	Qn
1	$\overline{Qn}$

Characteristic table:

T	Qn	Qn+1
0	0	0
0	1	1
1	0	1
1	1	0

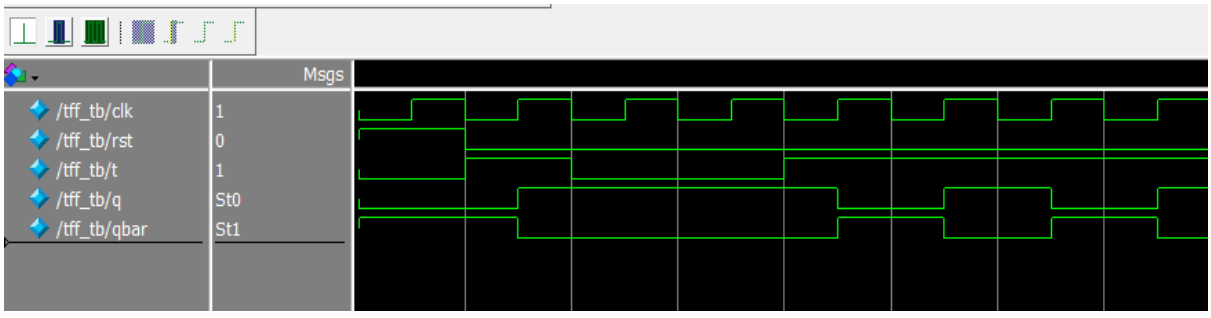
Excitation table:

Qn	Qn+1	T
0	0	0
0	1	1
1	0	1
1	1	0

Characteristic equation: $Qn+1 = T \oplus Qn$

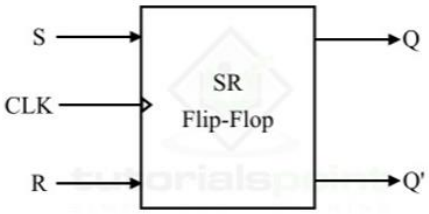
```
module t_flipflop(
input t,
input clk,rst,
output reg q,
output qbar
);

always @(posedge clk or posedge rst) begin
   if(rst)
     q <= 1'b0;
   else
     q <= q^t;
   end
assign qbar = ~q;
endmodule
```

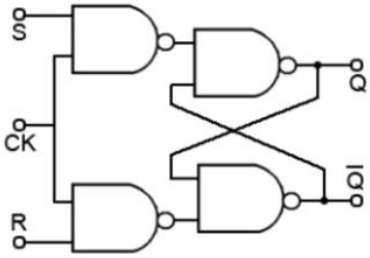
```
`timescale 1ns/100ps
module tff_tb;
reg t;
reg clk,rst;
wire q,qbar;

t_flipflop uut(t,clk,rst,q,qbar);
always #5 clk = ~clk;

initial begin
clk=0; t=0; rst=1; #10;
rst=0;

t=1; #10;
t=0; #10;
t=0; #10;
t=1; #10;

end
endmodule
```



SR Flip Flop

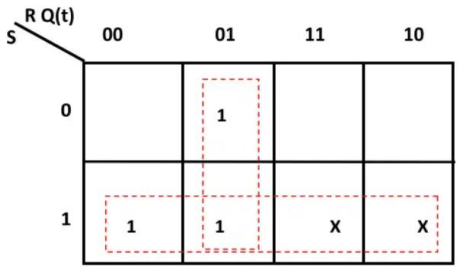
The SR Flip-Flop (Set–Reset Flip-Flop) is one of the basic memory elements in digital electronics. It is a clock-controlled bistable sequential circuit used to store 1 bit of data (0 or 1).





Truth table:

S	R	Qn+1
0	0	Qn
0	1	0
1	0	1
1	1	Invalid



$$Q_{n+1} = S + \overline{R}\,Q_n$$

Characteristic table:

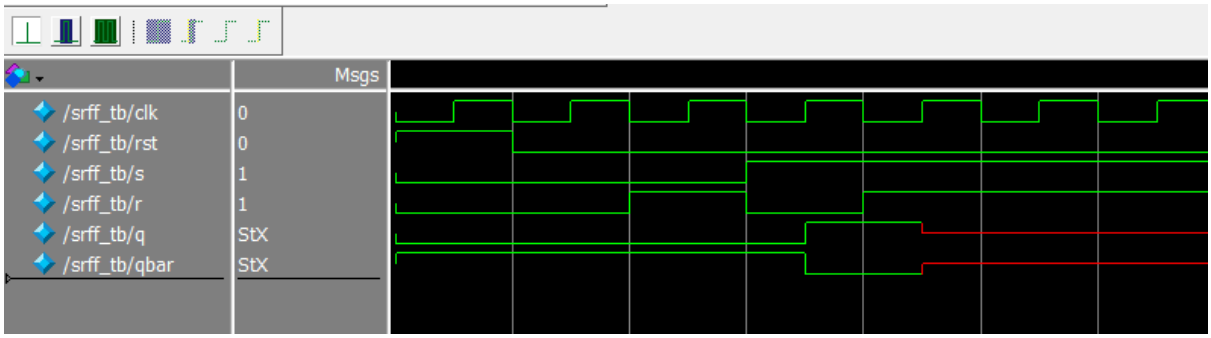
S	R	Qn	Qn+1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X

Excitation table:

Qn	Qn+1	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

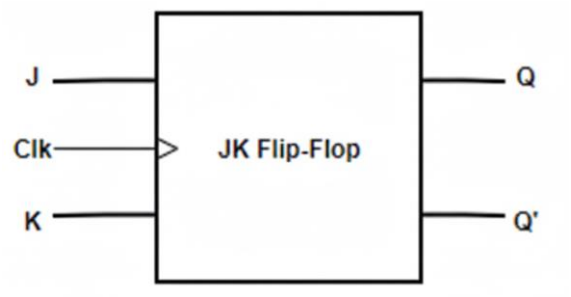
```
module sr_flipflop(
input s,r,
input clk,rst,
output reg q,
output qbar
);

always @(posedge clk or posedge rst) begin
   if(rst) begin
     q<=1'b0;
   end

  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 qbar = ~q;
endmodule
```

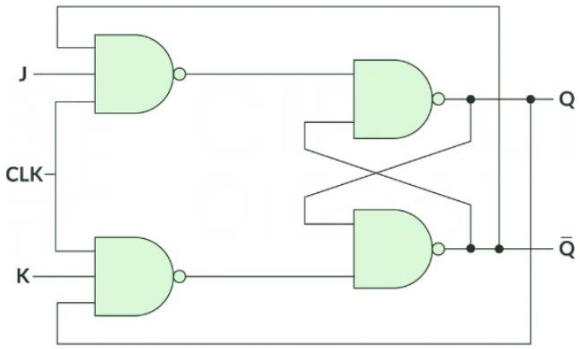
```
`timescale 1ns/100ps
module srff_tb;
reg s,r;
reg clk,rst;
wire q,qbar;

sr_flipflop uut(s,r,clk,rst,q,qbar);
always #5 clk = ~clk;
initial begin
clk=0; s=0; r=0; rst=1; #10;
rst=0;

s=0; r=0; #10; //memory
s=0; r=1; #10; //reset
s=1; r=0; #10; //set
s=1; r=1; #10; //Invalid in SR
end
endmodule
```



JK Flip Flop

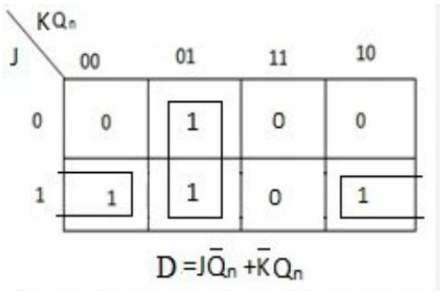
The JK Flip-Flop is an improved version of the SR Flip-Flop that removes the invalid condition (S=1, R=1). It is a clock-controlled bistable sequential circuit whose output changes only on the active clock edge.





Truth table:

J	K	Qn+1
0	0	Qn
0	1	0
1	0	1
1	1	~Qn



Characteristic table:

J	K	Qn	Qn+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

Excitation table:

Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

```
module jk_flipflop(
input j,k,
input clk,rst,
output reg q,
output qbar
);

always @(posedge clk or posedge rst) begin
  if(rst) begin
     q<=1'b0;
  end

  else begin
    case({j,k})
      2'b00: q<=q;
      2'b01: q<=1'b0;
      2'b10: q<=1'b1;
      2'b11: q<=~q;
    endcase
  end
end
assign qbar = ~q;
endmodule
```

```
`timescale 1ns/100ps
module jk_tb;
reg j,k;
reg clk,rst;
wire q,qbar;

jk_flipflop uut(j,k,clk,rst,q,qbar);
always #5 clk = ~clk;
initial begin
clk=0; j=0; k=0; rst=1; #10;
rst=0;

j=0; k=0; #10;
j=0; k=1; #10;
j=1; k=0; #10;
j=1; k=1; #10;

end
endmodule
```