

**UNIT 6**

**SEQUENTIAL LOGIC**

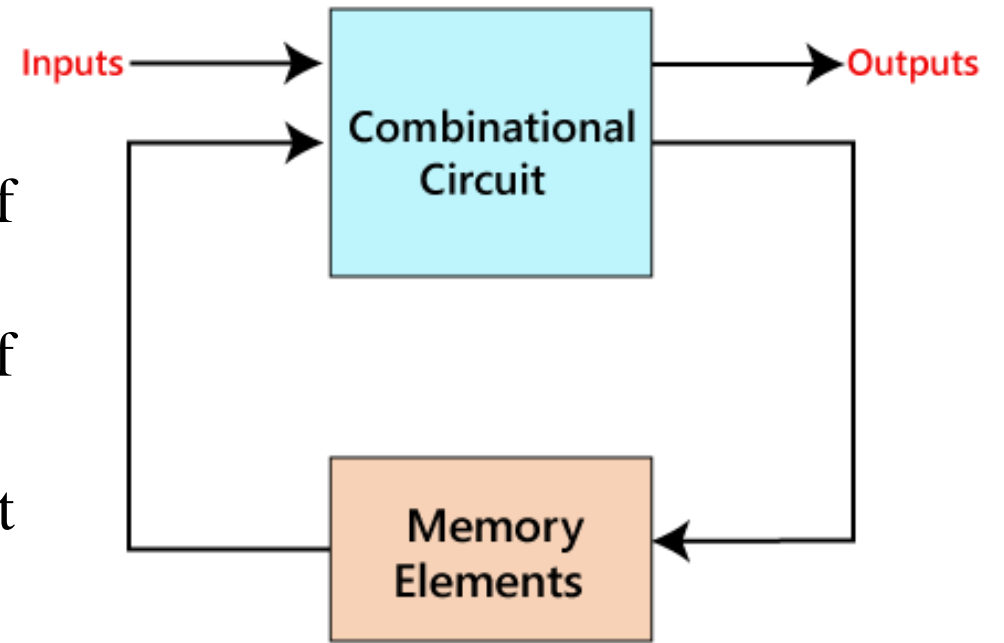
**SYNCHRONOUS AND ASYNCHRONOUS**

# Introduction

- Digital circuits considered thus far have been combinational.
- In combinational circuits, the output depends only and immediately on their inputs. They have no memory, i.e. dependence on past values.
- Sequential circuits, however, act as storage elements and have memory. They can store, retain, and then retrieve information when needed at a later time.
- This course distinguishes sequential logic from combinational logic.

# Sequential circuit

- A special type of circuit that has a series of inputs and outputs.
- The outputs depend on both the combination of present inputs and previous outputs.
- The previous output is treated as the present state.
- Contains the combinational circuit and its memory storage elements.
- Can contain only the memory element.

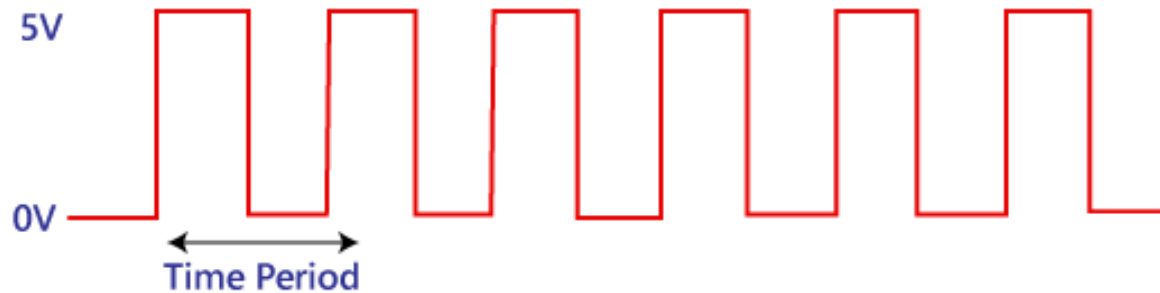


<b>S.N.</b>	<b>Combinational Circuits</b>	<b>Sequential Circuits</b>
<b>Output</b>	Depend only on the present inputs	Depend on both present inputs and present state(previous output)
<b>Feedback</b>	Not present	Present
<b>Memory element</b>	Not required/present	Memory elements play an important role and require
<b>Clock signal</b>	Not required	Required
<b>Design</b>	simple to design	Not simple/Complex

# Clock Signal and Triggering

- **Clock signal**

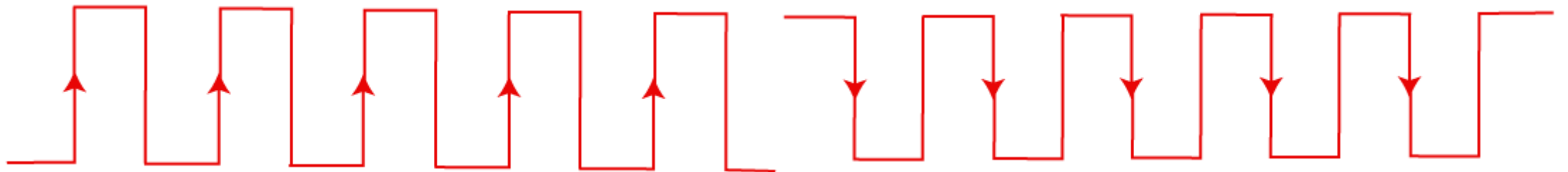
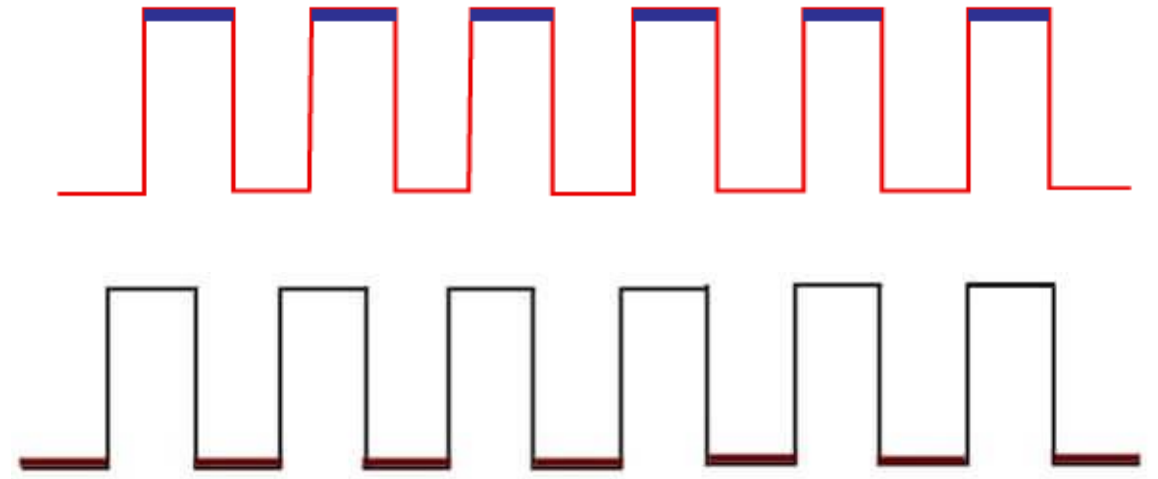
- A periodic signal in which ON time and OFF time need not be the same.
- When on time and off time of the clock signal are the same, a square wave is used to represent the clock signal.



- A clock signal is considered as the square wave.
- It repeats with a certain time period, which will be equal to twice the 'ON time' or 'OFF time'.

# Triggering of Flip-flop

- Defines when output state of Flip-flop changes.
- Two ways:
  - i. Level trigger
    - a. Positive/High level triggering
    - b. Negative/Low level triggering
  - ii. Edge Trigger
    - a. Positive edge triggering
    - b. Negative edge triggering



# Types of Sequential Circuits

## 1. Asynchronous sequential circuits

- The clock signals are not used
- Operated through the pulses. So, the changes in the input can change the state of the circuit.
- The internal state is changed when the input variable is changed.
- The un-clocked flip-flops or time-delayed are the memory elements.
- Similar to the combinational circuits with feedback.

## 2. Synchronous sequential circuits

- synchronization of the memory element's state is done by the clock signal.
- The output is stored in either flip-flops or latches(memory devices).
- The synchronization of the outputs is done with either only negative edges of the clock signal or only positive edges.

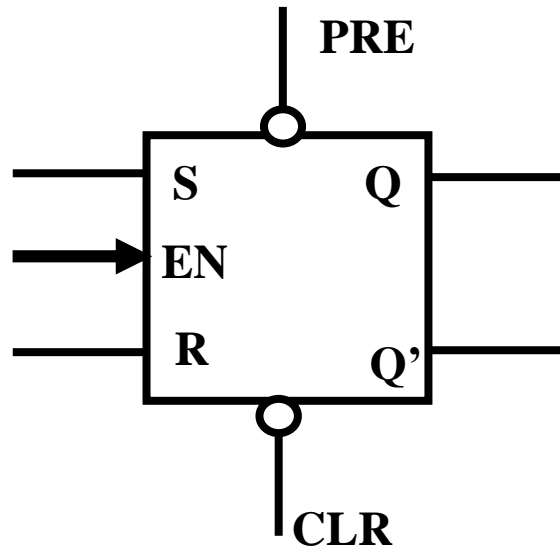
# Storage Elements: Latches/Flip-flops

- A storage element can maintain a binary state indefinitely (as long as power is delivered to the circuit), or until directed by an input signal to switch states.
- Storage elements that operate with signal levels (rather than signal transitions) are referred to as **latches**.
- Storage elements controlled by a clock transition are **flip-flops**.
- For this reason latches are said to be level sensitive devices, and flip-flops are said to be edge sensitive devices.
- Although latches can store information they are not practical for use as storage elements in synchronous sequential circuits.
- They must be studied however, because they are the basic building blocks of flip-flops.

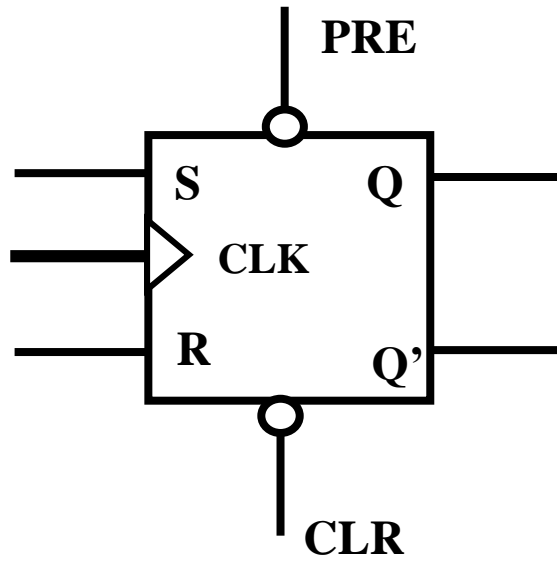


# Flip-flops

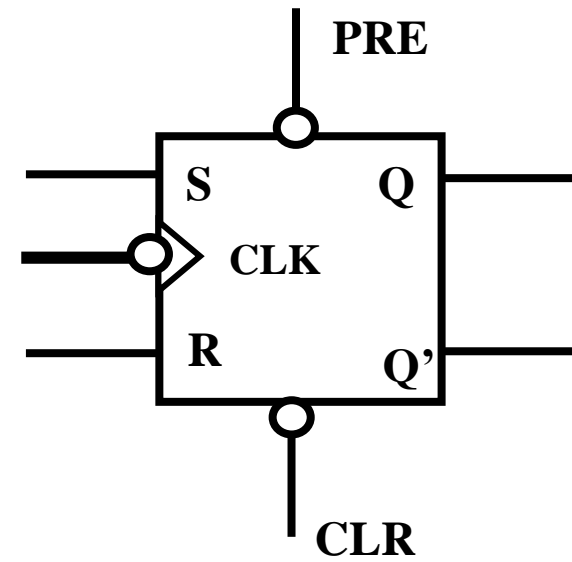
- Basic digital memory circuit.
- Stores 1-bit, therefore called 1-bit memory cell.
- It has two stable states: logic 1 and logic 0
- It can flip from one state to another and then flop back, so named Flip-flop.
- Also known as multivibrator.
- Two outputs  $Q$  and  $Q'$  (always complementary).
- If  $Q=1$  and  $Q'=0$ , it is in SET state.
- If  $Q=0$  and  $Q'=1$ , it is in RESET state.



**Positive Level trigg. FF**



**Positive Edge trigg. FF**



**Negative Edge trigg. FF**

# **Application of Flip-flops**

- As a memory element
- In registers
- In counters/timers
- As a delay element

## **Types of Flip-flops**

- SR Flip-flops**
- D Flip-flops**
- JK Flip-flops**
- T Flip-flops**
- Master-slave Flip-flops**

# SR Flip-flops

- The SR flip flop is a 1-bit memory bi-stable device having two inputs, i.e., SET and RESET.
- The SET input 'S' set the device or produce the output 1, and the RESET input 'R' reset the device or produce the output 0.
- The SET and RESET inputs are labelled as **S** and **R**, respectively.
- Can be implemented using NOR and NAND gates.
- Often called a ***latch*** as it holds or latch, in either stable state.

# NAND gate SR latch

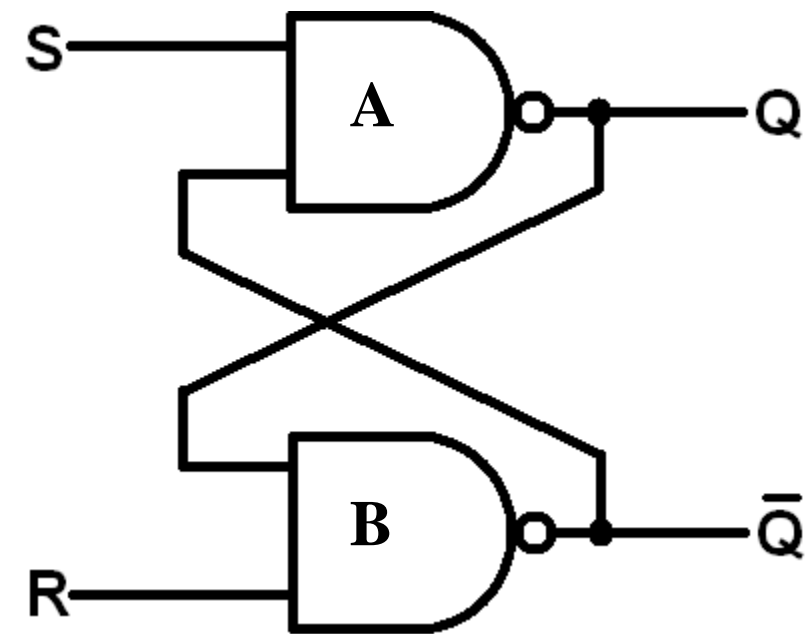


Fig: Logic Diagram

Truth Table

State	S	R	Q	Q'	Description
Set	1	0	0	1	Set Q'>>1
	1	1	0	1	No change
Reset	0	1	1	0	Reset Q'>>0
	1	1	1	0	No change
Invalid	0	0	1	1	Invalid Condition

**Case 1:  $R' = 0$  and  $S' = 1$  (set  $Q' \gg 1$ )**

- As S is HIGH, the output of NAND gate A i.e., Q becomes LOW. This causes both the inputs of NAND gate B to become LOW and hence, the output of NAND gate B i.e., Q' becomes HIGH.

**Case 2:  $R' = 1$  and  $S' = 0$**

- As R is HIGH, the output of NAND gate B i.e., Q' becomes LOW. This causes both the inputs of NAND gate A to become LOW and hence, the output of NAND gate A i.e., Q becomes HIGH.

**Case 3:  $R' = 1$  and  $S' = 1$**

- The output remains in previous state i.e., it holds the previous data.

**Case 4:  $R' = 0$  and  $S' = 0$**

The flip flop will be in undefined state. Because the low inputs of S and R, violates the rule of flip – flop that the outputs should complement to each other.

*So, the flip flop is in undefined state (or forbidden state).*

# NOR gate SR latch

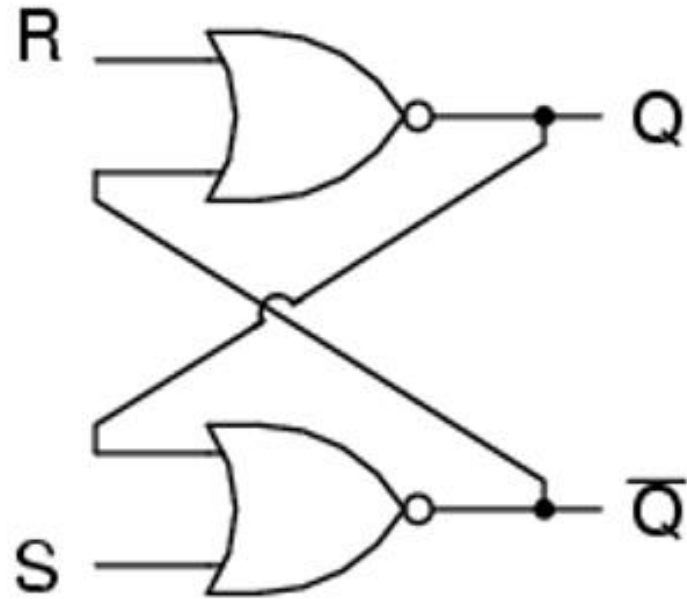


Fig: Logic Diagram

Truth Table

S	R	Q	Q'	REMARKS
1	0	1	0	SET
0	0	1	0	NO CHANGE
0	1	0	1	RESET
0	0	0	1	NO CHANGE
1	1	?	?	INVALID

# 1. Clocked/Gated S-R Flip-Flop

- The operation of a basic flip-flop can be modified by providing an additional control input that determines when the state of the circuit is to be changed.
- The limitation with a S-R flip-flop using NOR and NAND gate is the *invalid state*.
- This problem can be overcome by using a stable SR flip-flop that can change outputs when certain invalid states are met, regardless of the condition of either the Set or the Reset inputs.

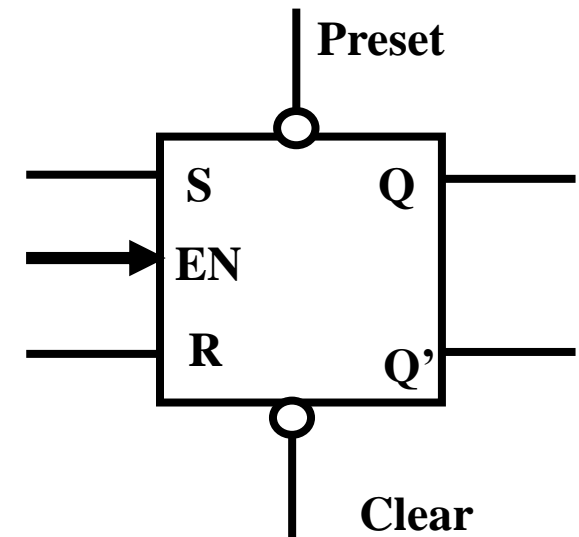


Fig: Logic Symbol



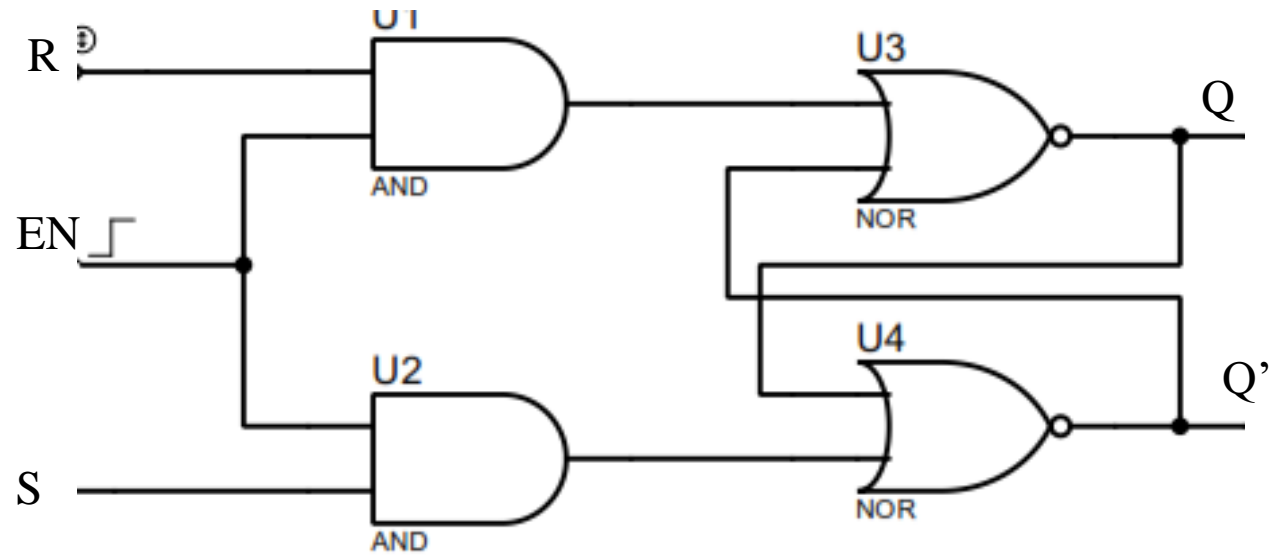
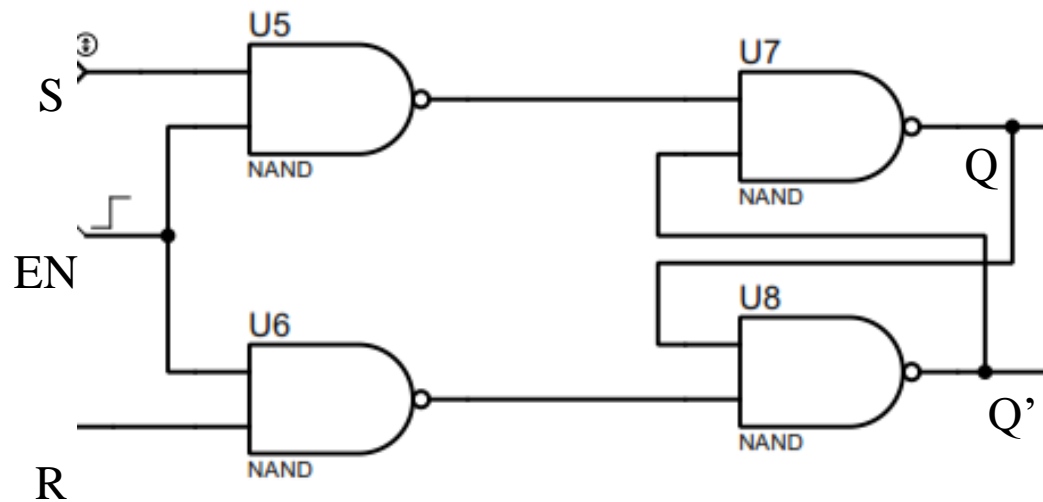


Fig: Logic Diagram

**Truth Table**

Clk	S	R	$Q_{n+1}$
0	x	x	$Q_n$
1	0	0	$Q_n$
1	0	1	0
1	1	0	1
1	1	1	Invalid

A clock pulse is given to the inputs of the AND Gate. If the value of the clock pulse is '0', the outputs of both the AND Gates remain '0'.



*Fig: Gated RS Flip-flop (NAND)*

		SR			
		00	01	11	10
Q <sub>n</sub>	0	0	0	x	1
	1	1	0	x	1

*Characteristics Table*

Q <sub>n</sub>	S	R	Q <sub>n+1</sub>	Remarks
0	0	0	0	No Change
0	0	1	0	Reset
0	1	0	1	Set
0	1	1	x	Invalid
1	0	0	1	No Change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	x	Invalid

*Characteristics equation,  $Q_{n+1} = S + Q_n R'$*

## 2. Gated D (Delay or Data) Flip-flop

- Only one input called data (*D*) *input* and two *outputs* *Q* and *Q'*.
- Can be constructed from SR Flip-flop by inserting an inverter between S and R and assigning symbol D to input S.
- *The output Q is always same as D input.*

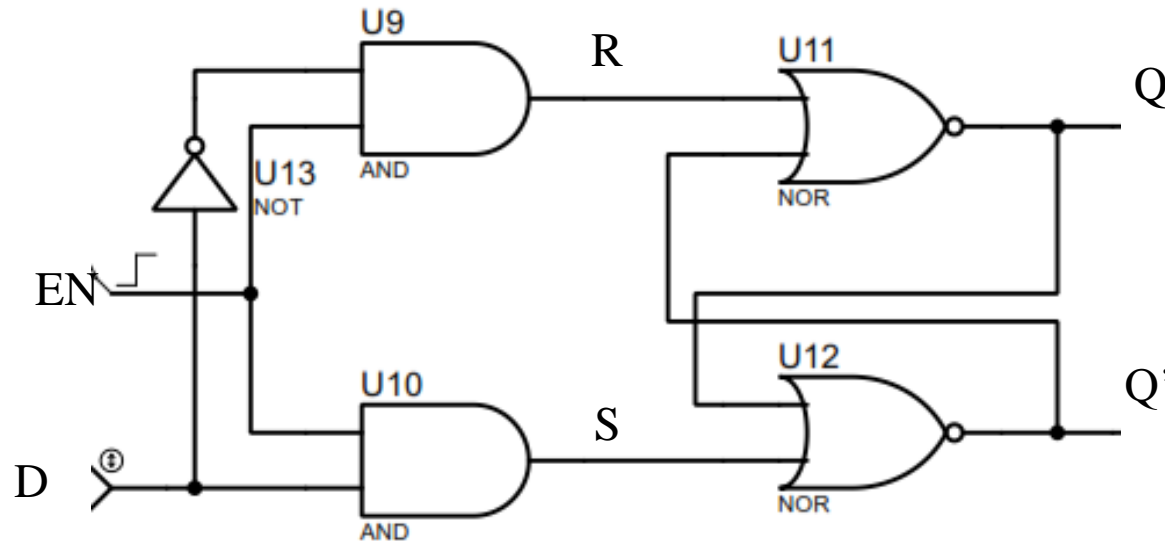


Fig: Logic Diagram D Flipflop

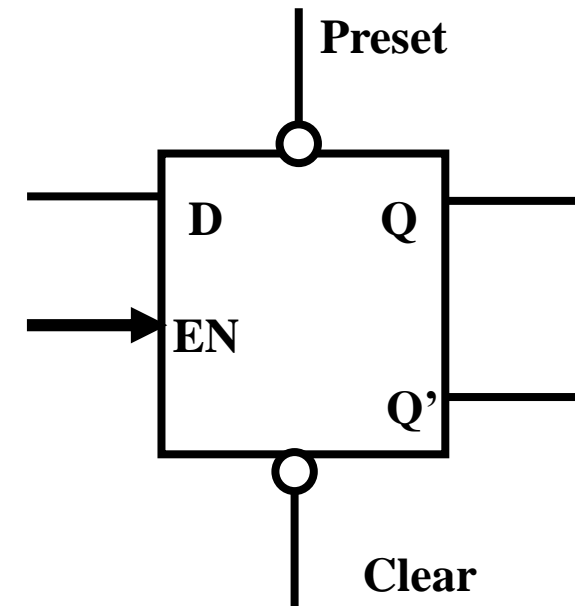
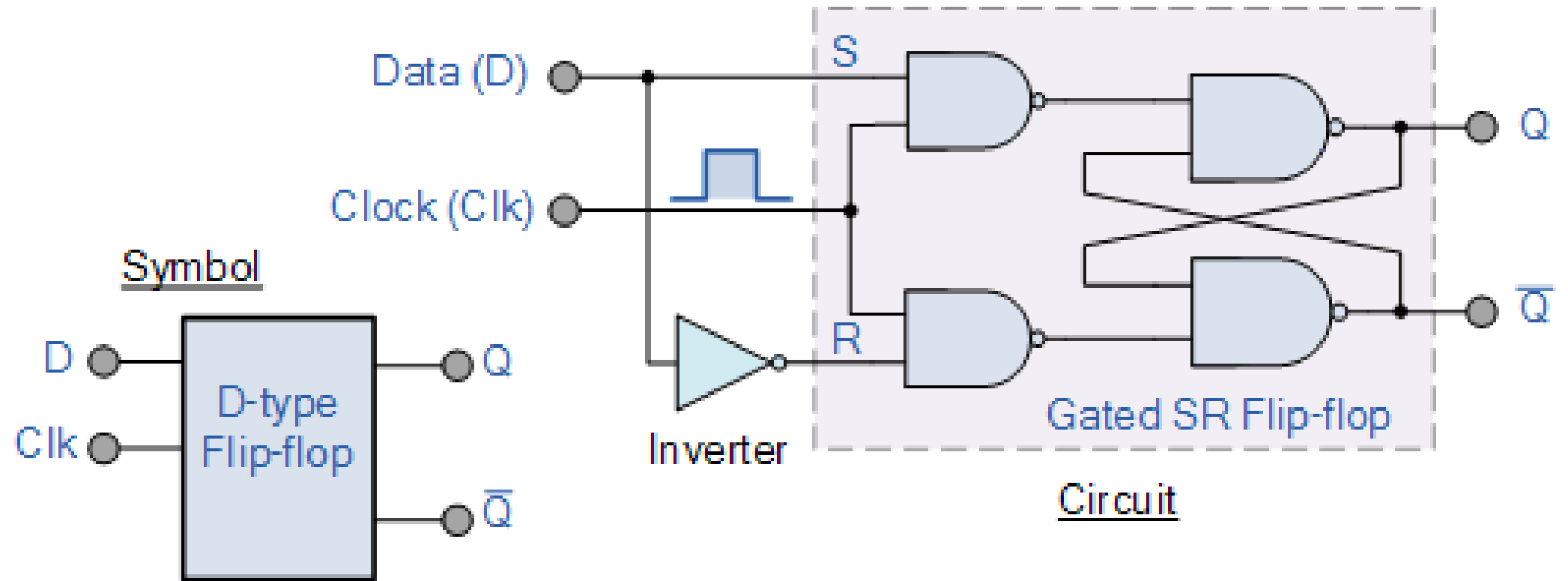


Fig: Logic Symbol



*Characteristics Table*

$Q_n$	<b>D</b>	$Q_{n+1}$	<b>Remarks</b>
0	0	0	Same as D
0	1	1	Same as D
1	0	0	Same as D
1	1	1	Same as D

	<b>D</b>	$\bar{D}$	<b>D</b>
<b>Q<sub>n</sub></b>			
0	0	1	
1	0	1	

$$Q_{n+1} = D$$

*Advantages:*

- *Drawback of RS flipflop (invalid state) is overcome.*
- *Single input to drive the flipflop.*

### 3. Gated JK Flip-flop

- Refined form of RS Flip-flop.
- Invalid states are defined as toggled state.
- Constructed using cross coupled NOR gates and two AND gates.

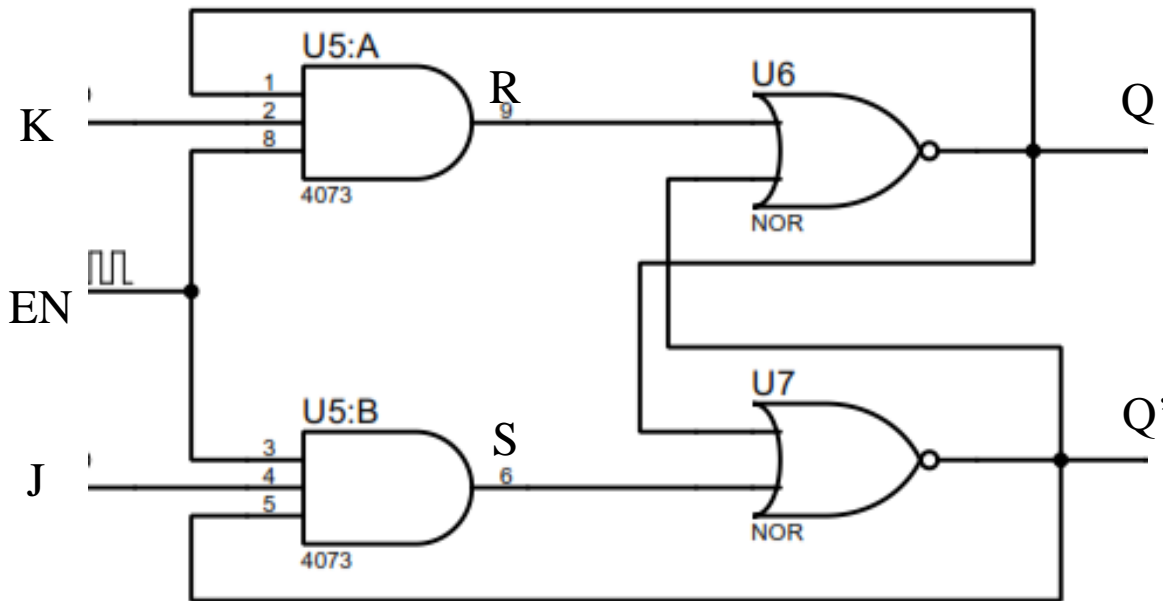


Fig: Logic Diagram JK Flipflop

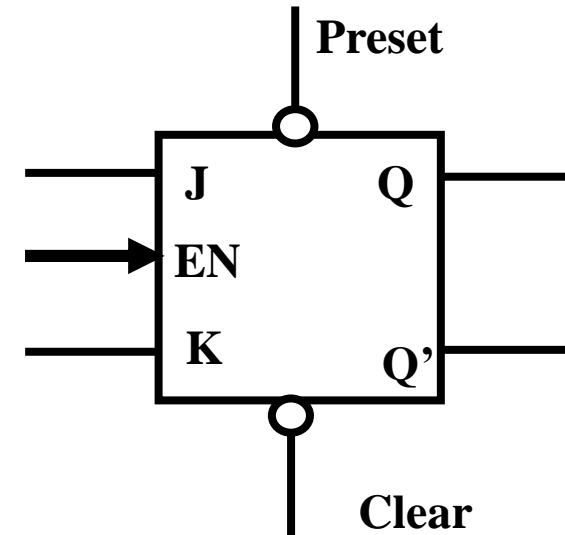
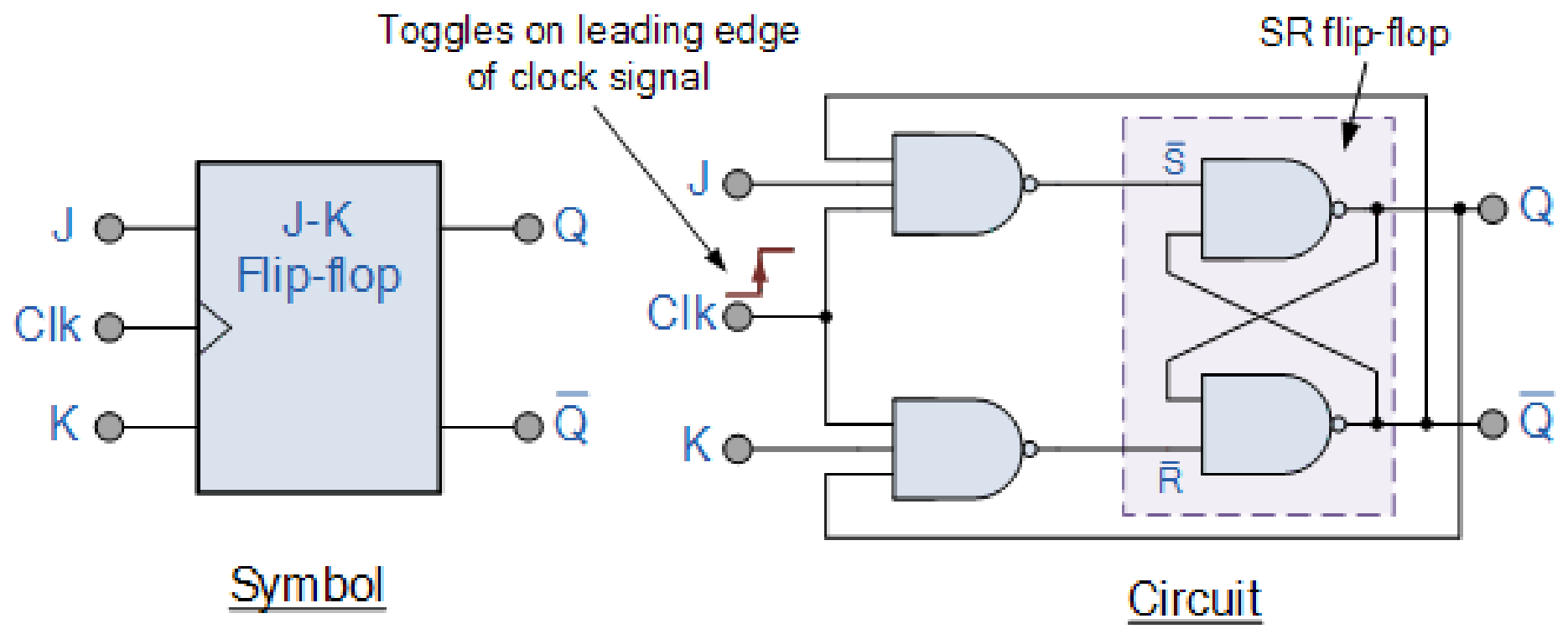


Fig: Logic Symbol



### Case I: J=0, K=0

- If J=0, K=0, the output of two AND gates i.e., R and S are equal to 0.
- We know that if R=S=0, the output Q and Q' i.e.,  $Q_{n+1}=Q_n$  and  $Q'_{n+1}=Q'_n$ .

### Case II: J=0, K=1

#### i. Provided that Q=1 and Q'=0

$$S=EN.J.Q'$$

$$=1.0.0 = 0$$

$$\text{i.e., } Q_{n+1}=0$$

$$R=EN.K.Q$$

$$=1.1.1 = 1$$

#### ii. Provided that Q=0 and Q'=1

$$S=EN.J.Q'$$

$$=1.0.1 = 0$$

$$\text{i.e., remain in rest state, } Q_{n+1}=0$$

$$R=EN.K.Q$$

$$=1.1.0 = 0$$



### Case III: J=1, K=0

- i. Provided that  $Q=1$  and  $Q'=0$

$$S = EN.J.Q'$$

$$= 1.1.0 = 0$$

$$\text{i.e., } Q_{n+1}=1$$

$$R = EN.K.Q$$

$$= 1.0.1 = 0$$

- ii. Provided that  $Q=0$  and  $Q'=1$

$$S = EN.J.Q'$$

$$= 1.1.1 = 1$$

$$\text{i.e., } Q_{n+1}=1$$

$$R = EN.K.Q$$

$$= 1.0.0 = 0$$

### Case IV: J=1, K=1

- i. Provided that  $Q=1$  and  $Q'=0$

$$S = EN.J.Q'$$

$$= 1.1.0 = 0$$

$$\text{i.e., } Q_{n+1}=0$$

$$R = EN.K.Q$$

$$= 1.1.1 = 1$$

- ii. Provided that  $Q=0$  and  $Q'=1$

$$S = EN.J.Q'$$

$$= 1.1.1 = 1$$

$$\text{i.e., } Q_{n+1}=1$$

$$R = EN.K.Q$$

$$= 1.1.0 = 0$$

- The outputs are inverted i.e., toggled from reset to set [ $Q=0$  to  $Q=1$ ].*

# Characteristics table

$Q_n$	J	K	$Q_{n+1}$	Remarks
0	0	0	0	No Change
0	0	1	0	Reset
0	1	0	1	Set
0	1	1	1	Toggle
1	0	0	1	No Change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	0	Toggle

		JK			
		00	01	11	10
$Q_n$	0	0	0	1	1
	1	1	0	0	1

*Characteristics equation,  $Q_{n+1} = Q_n'J + Q_nK$*

## 4. Gated Toggle Flip-flop (T Flip-flop)

- Made by shorting J and K input of JK Flip-flop.
- Output Toggles each time for  $T=1$ .

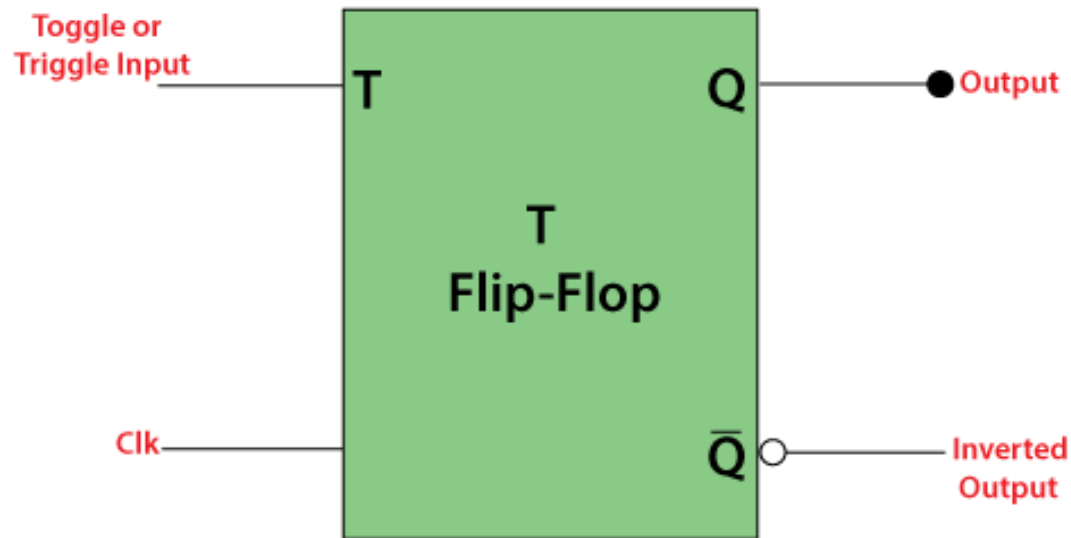


Fig: Logic Symbol

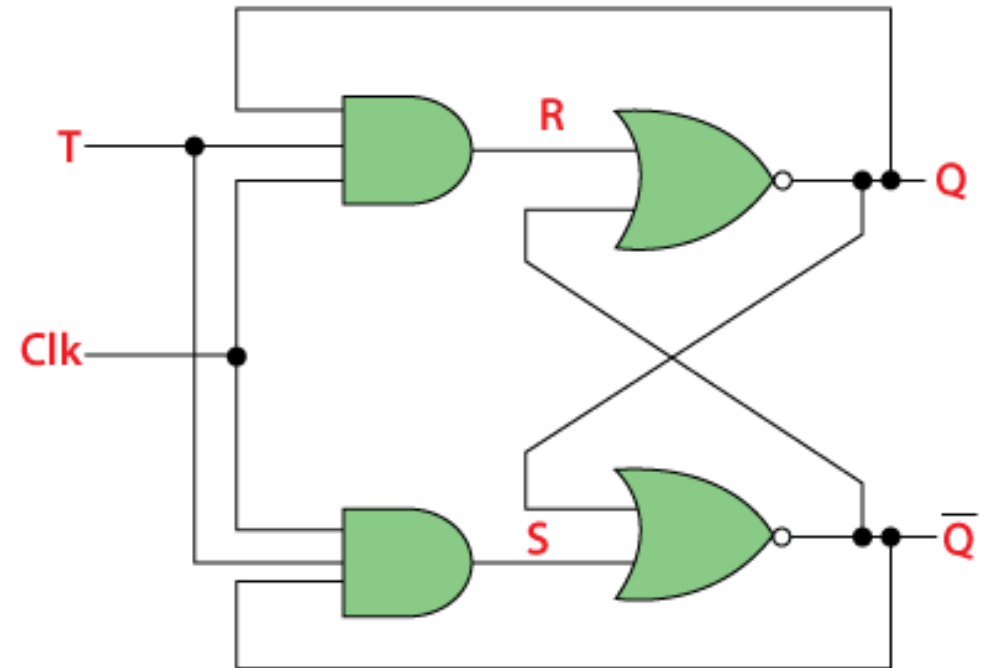


Fig: Logic Diagram T Flipflop

### Characteristics Table:

$Q_n$	$T$	$Q_{n+1}$	Remarks
0	0	0	No change
0	1	1	Toggle
1	0	1	No change
1	1	0	Toggle

	$T$	$\bar{T}$	$T$
$Q_n$			
0	0	1	
1	1	0	

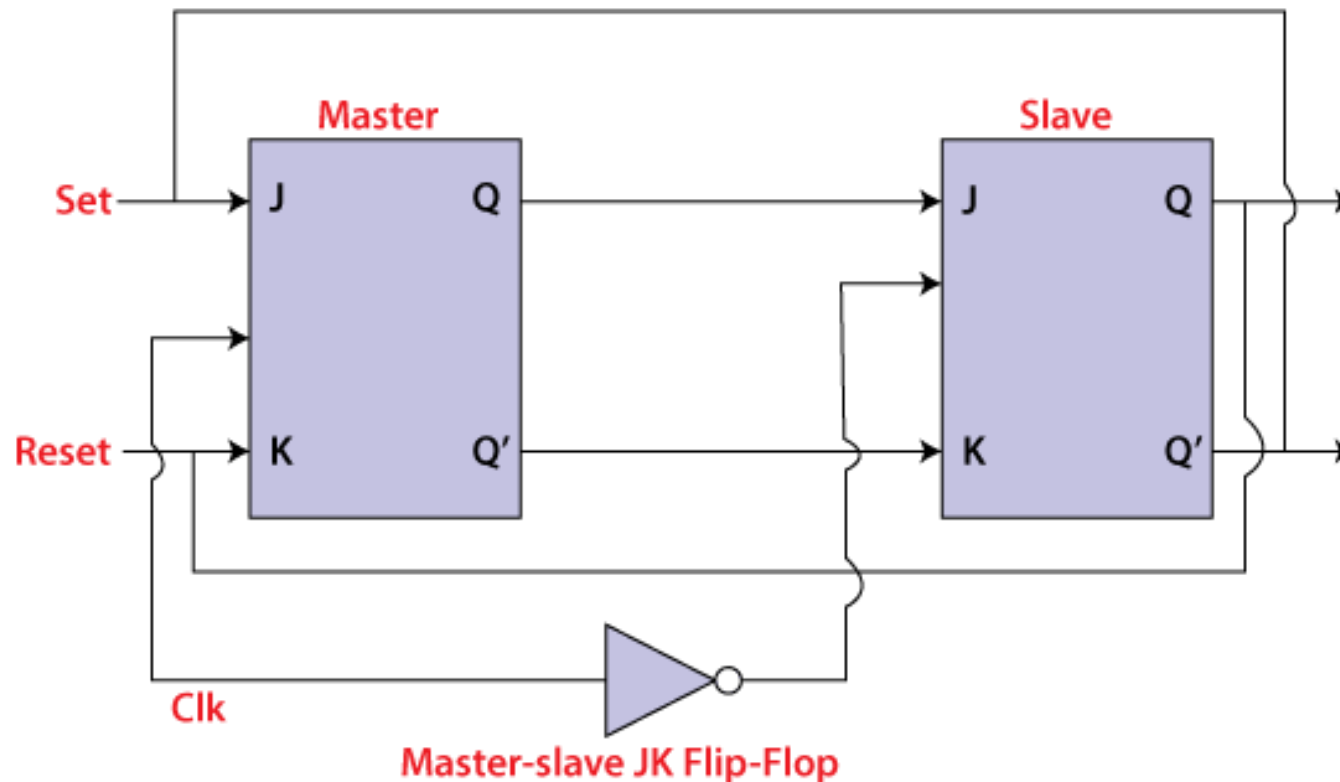
Characteristics equation,  $Q_{n+1} = Q_n' T + Q_n T'$   
 $= (Q_n \oplus T)$

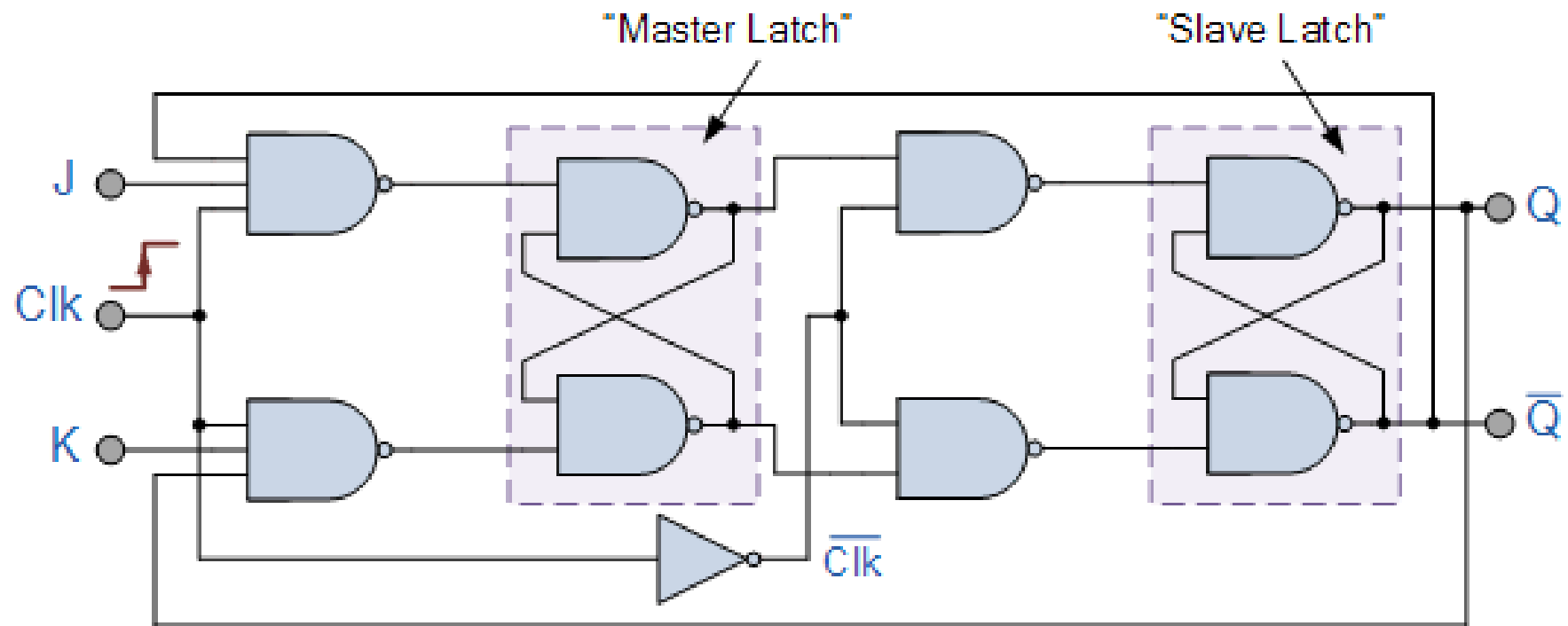
# Master slave Flip-flop

- **Race around condition in JK FF:**

- In "JK Flip Flop", when both the inputs and CLK set to 1 for a long time, then Q output toggle until the next CLK transition. Thus, the uncertain or unreliable output produces. This problem is referred to as a *race-round condition* in JK flip-flop and avoided by ensuring that the CLK set to 1 only for a very short time.
- Used to *eliminate the race around condition*.
- Constructed by combining two *JK flip flops*.
- Flip flops are connected in a *series configuration*.
- In these two flip flops, the *1st flip flop* work as "*master*", called the master flip flop, and the *2nd* work as a "*slave*", called slave flip flop.
- The master-slave flip flop is designed in such a way that the output of the "master" flip flop is passed to both the inputs of the "slave" flip flop.
- The output of the "slave" flip flop is passed to inputs of the master flip flop.
- Apart from these two flip flops, an *inverter or not gate* is also used for passing the inverted clock pulse to the "slave" flip flop, the inverter is connected to the clock's pulse.

- The master-slave flip flop is designed in such a way that the output of the "master" flip flop is passed to both the inputs of the "slave" flip flop.
- The output of the "slave" flip flop is passed to inputs of the master flip flop.
- Apart from these two flip flops, an *inverter or not gate* is also used for passing the inverted clock pulse to the "slave" flip flop, the inverter is connected to the clock's pulse.





## Working:

- When the clock pulse =1, the slave flip flop will be in the isolated state, and the system's state may be affected by the J and K inputs. The "slave" remains isolated until the CP is 1. When the CP set to 0, the master flip-flop passes the information to the slave flip flop to obtain the output.
- The master flip flop responds first from the slave because the master flip flop is the positive level trigger, and the slave flip flop is the negative level trigger.
- The output  $Q'=1$  of the master flip flop is passed to the slave flip flop as an input K when the input J set to 0 and K set to 1. The clock forces the slave flip flop to work as reset, and then the slave copies the master flip flop.
- When  $J=1$ , and  $K=0$ , the output  $Q=1$  is passed to the J input of the slave. The clock's negative transition sets the slave and copies the master.
- The master flip flop toggles on the clock's positive transition when the inputs J and K set to 1. At that time, the slave flip flop toggles on the clock's negative transition.
- The flip flop will be disabled, and Q remains unchanged when both the inputs of the JK flip flop set to 0

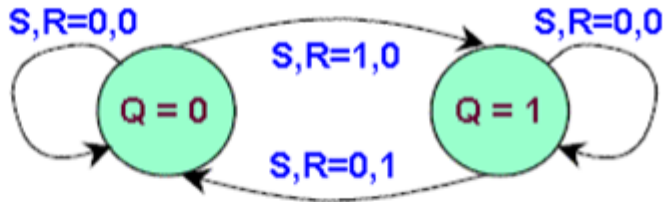
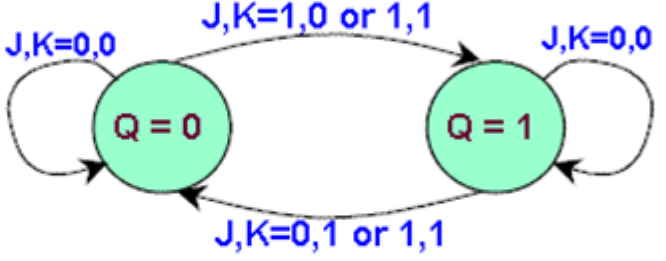
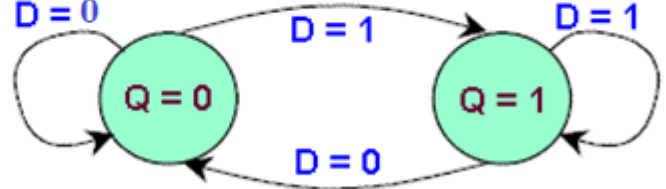
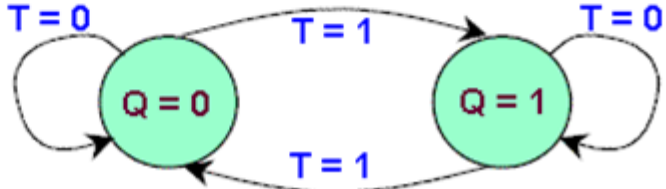


# Flip-flip excitation table

Transition		RS FF		JK FF		D FF	T FF
$Q_n \rightarrow Q_{n+1}$		S	R	J	K	D	T
0	0	0	x	0	x	0	0
0	1	1	0	1	x	1	1
1	0	0	1	x	1	0	1
1	1	x	0	x	0	1	0

# FF as a State Machine:

Table: State diagrams of the four types of flip-flops.

NAME	STATE DIAGRAM
SR	 <pre> graph LR     Q0((Q = 0))     Q1((Q = 1))     Q0 -- "S,R=0,0" --&gt; Q0     Q0 -- "S,R=1,0" --&gt; Q1     Q1 -- "S,R=0,1" --&gt; Q0     Q1 -- "S,R=0,0" --&gt; Q1         </pre>
JK	 <pre> graph LR     Q0((Q = 0))     Q1((Q = 1))     Q0 -- "J,K=0,0" --&gt; Q0     Q0 -- "J,K=1,0 or 1,1" --&gt; Q1     Q1 -- "J,K=0,1 or 1,1" --&gt; Q0     Q1 -- "J,K=0,0" --&gt; Q1         </pre>
D	 <pre> graph LR     Q0((Q = 0))     Q1((Q = 1))     Q0 -- "D=0" --&gt; Q0     Q0 -- "D=1" --&gt; Q1     Q1 -- "D=0" --&gt; Q0     Q1 -- "D=1" --&gt; Q1         </pre>
T	 <pre> graph LR     Q0((Q = 0))     Q1((Q = 1))     Q0 -- "T=0" --&gt; Q0     Q0 -- "T=1" --&gt; Q1     Q1 -- "T=1" --&gt; Q0     Q1 -- "T=0" --&gt; Q1         </pre>