Analog
and
Digital Electronics

(Second Year BTech Computer)

(2019-20)

(Faculty: Mr. Y. K. Sharma)

Courtesy: R. P. Jain
(Modern Digital Electronics)

UNIT- 3

Sequential Logic

Courtesy: R. P. Jain & J. S. Katre

Sequential Logic

- **Flip-flop:** SR, JK, D, T; Preset & Clear, Master and Slave Flip Flops, Truth Tables and Excitation tables, Conversion from one type to another type of Flip Flop.

- **Registers:** Buffer register, shift register (SISO, SIPO, PISO & PIPO), Applications of shift registers.

Sequential Logic

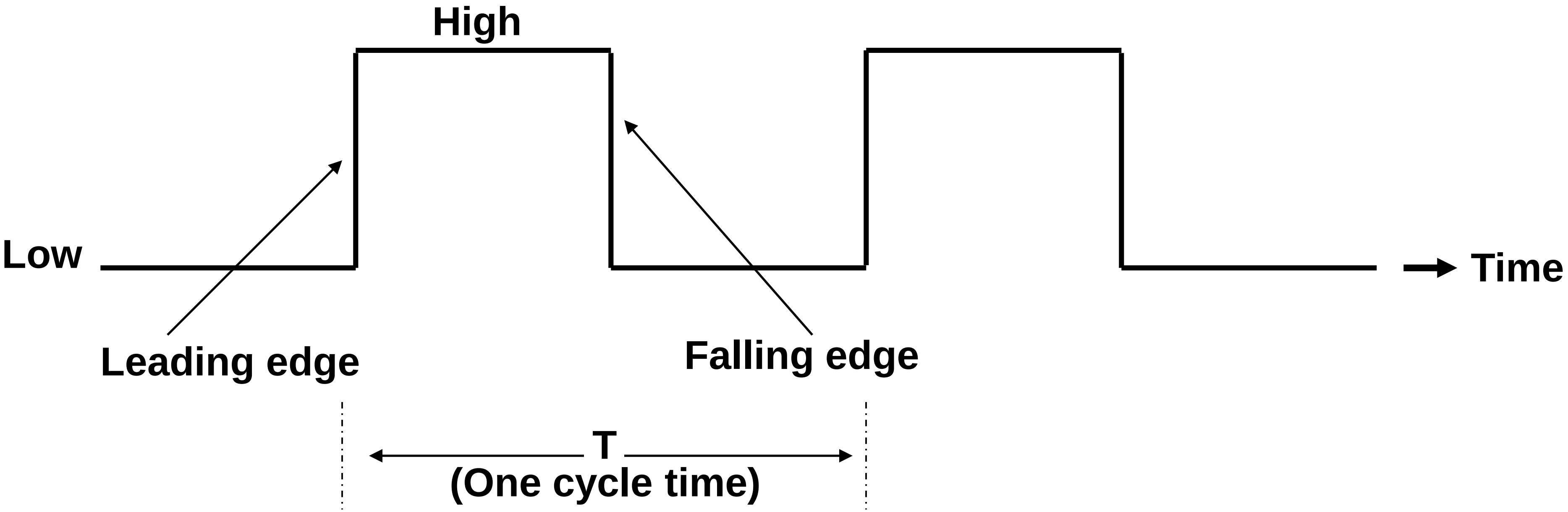
Introduction:

- Timing parameter comes into picture.
- Output depends on present time inputs, the previous output & the sequence in which the inputs are applied.
- To store a previous input or output, a memory element is required.
- The data stored by the memory element at any given instant of time is called as the present state of the sequential circuit.
- Combinational circuit operates on the external inputs and the present state to produce new outputs. Some of these new outputs are stored in the memory element and called as the next state of the sequential circuit.
- Memory element is the most important part of sequential circuits and is known as flip flop (FF). It is the basic memory element.

Sequential Logic

Clock Signal:

- **It is timing signal. Every sequential signal will have this timing signal applied.**
- **It is rectangular signal, with a duty cycle equal to 50%.**
- **Clock signal repeats itself after every T seconds. Hence the clock frequency if f = 1 / T.**



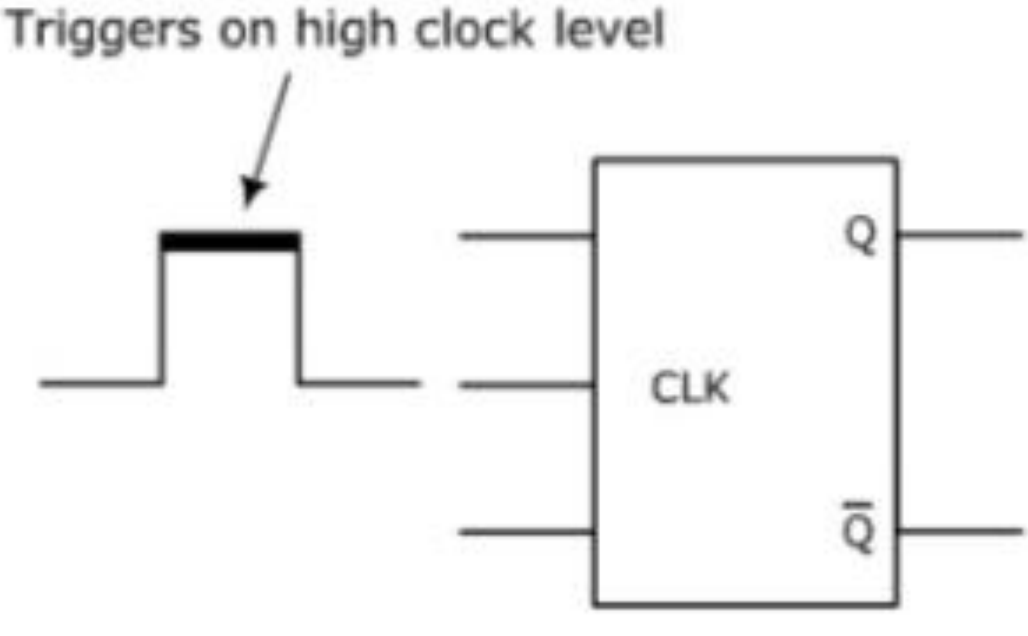
Triggering

- Sequential circuits are dependant on clock pulses applies to their inputs.
- The result of flip-flop responding to a clock input is called **clock pulse triggering**, of which there are four types. Each type responds to a clock pulse in one of four ways :-

1. High level triggering
2. Low level triggering
3. Positive edge triggering
4. Negative edge triggering

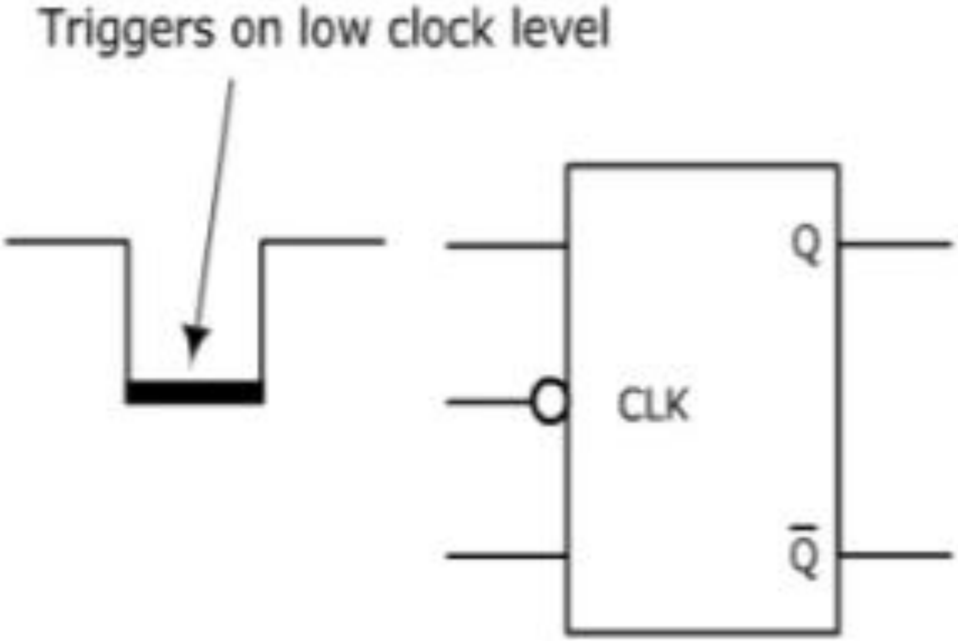
High Level Triggering

One type of flip-flop responds to a clock signal during the time at which it is in the logic High state. This type is identified by a straight lead at the clock input, as shown below.



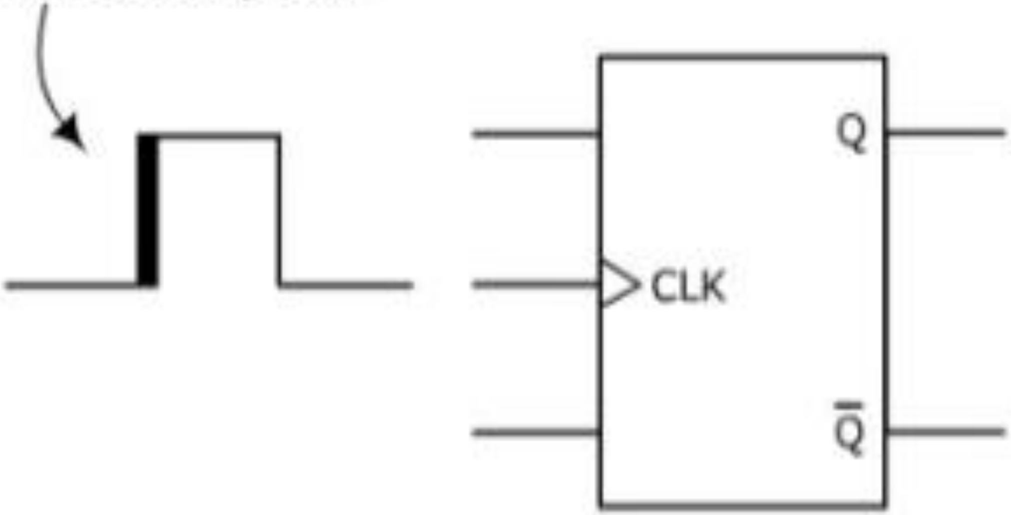
Low Level Triggering

Another type of flip-flop responds to a clock signal during the time at which it is in the logic Low state. This type is identified by a clock input lead with a low-state indicator bubble, as shown below.



Positive Edge Triggering

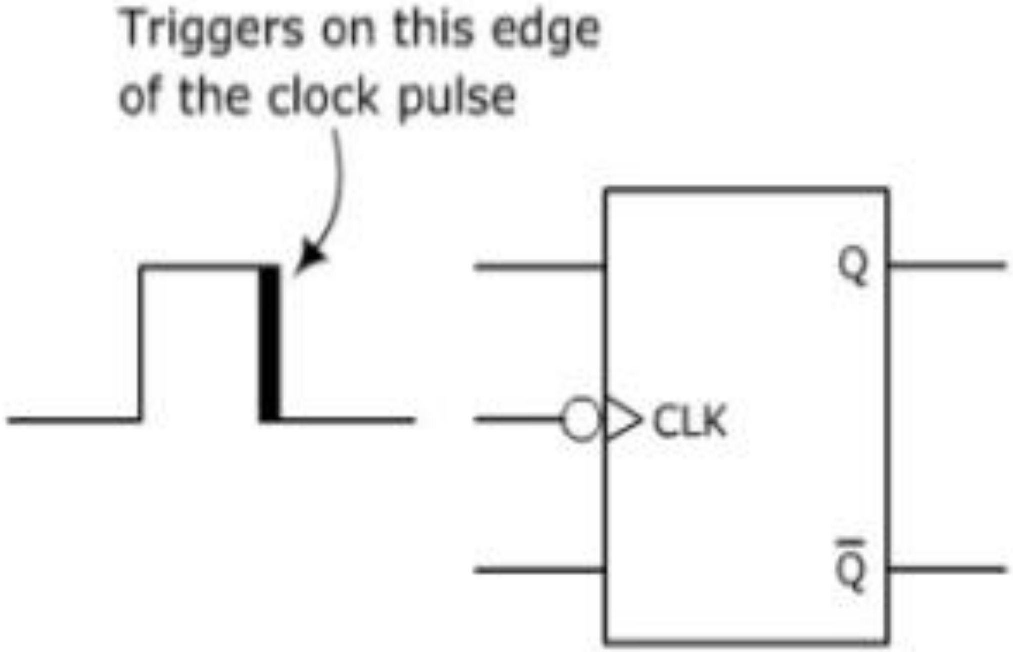
A third type of flip-flop responds to a clock signal during the low-to-high transition of a clock pulse. This type is identified by a clock input lead with a triangle, as shown below.



Triggers on this edge of the clock pulse

Negative Edge Triggering

The fourth type of flip-flop responds to a clock signal during the high-to-low transition of a clock pulse. This type is identified by a clock input lead with a low-state indicator and triangle, as shown below.



Triggers on this edge of the clock pulse

Sequential Logic

Clock Skew:

- **It is defined as the difference in time between the clock edges arriving at a pair of clock inputs.**
- **In a perfect system, all clock signals arrive at all the various clock input pins of the system at exactly the same time & the skew is zero.**
- **In real time systems, the edges do not arrive at exactly the same time & there is some skew.**
- **Maximum allowable clock skew for the system is the difference between the shortest and longest path delays.**
- **It is an important design parameter in high-speed clock systems.**
- **It appears due to different delays on different paths from the clock generator to various circuits. The major reasons for this are:**
  - **Different length of wires**
  - **Gates (buffers) on the paths.**
  - **Flip-flops that clock on different edges**
  - **Gating the clock to control loading of registers**

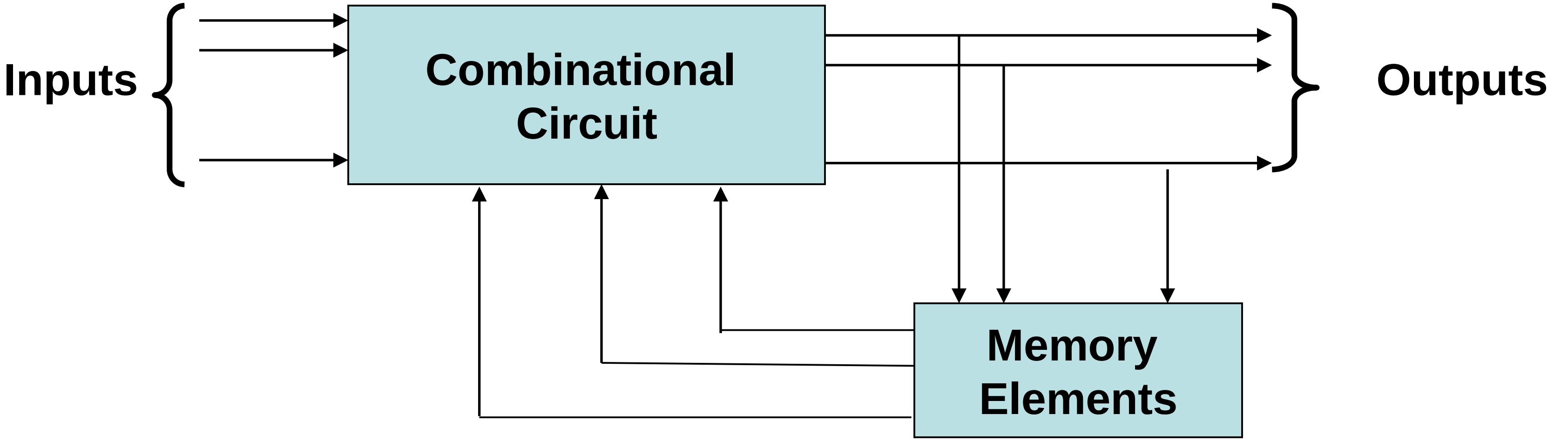
Sequential Logic

Comparison of Combinational & Sequential Circuits:

No.	Parameter	Combinational	Sequential
1.	Output depend on	Inputs present at that instant of time	Present inputs & past inputs/ outputs
2.	Memory	Not necessary	Necessary
3.	Clock Input	Not necessary	Necessary
4.	Examples	Adders, Subtractors, Code converters	Flip flops, shift registers, counters

Sequential Logic

- Sequential circuit consists of combinational circuit along with the memory element.
- Memory element will always appear in the feedback path.
- Output of sequential circuit depends on the present state as well as the past state.
- Present state is stored in the memory elements.
- Present states are applied to the combinational circuit along with the external inputs.
- Thus outputs & the next state of the sequential circuit are decided by the external inputs & the present state.



Sequential Logic

1-Bit Memory Cell:

- **Basic digital memory circuit is also known as Flip-flop.**
- **It has two stable states namely logic 1 state and logic 0 state. It can be designed either using NOR gates or NAND gates.**
- **A Flip-flop can be designed using NAND gates 1 and 2, where these two gates G1 & G2 act as inverters. Hence this circuit is called as a cross coupled inverter.**
- **Output of G1 is connected to the input of G2 & output of G2 is connected to input of G1.**
- **Assume Q=1 then input for G2 i.e. B2=1**
- **As B2=1, output of gate-2 ie. G2 i..e $\overline{Q}$=0. This makes B1=0.**
- **Hence Q continues to be equal to 1.**
- **Similarly, if we start with Q=0, then we end up obtaining Q=0 and $\overline{Q}$=1.**

Sequential Logic

Cross Coupled Inverters:

- Outputs Q & $\overline{Q}$ are always complimentary.
- The circuit has two stable states, in one stable state Q=1 & $\overline{Q}$=0 and it is called as 1 state or set state. Whereas the other Q=0 & $\overline{Q}$=1 is called as 0 state or reset state.
- If circuit is in 1 reset state, then it will continue to be in the reset state and if it is in the set state then it will continue to remain in the set state.
- This property of circuit shows that it can store 1 bit of digital information and hence known as 1 bit memory cell.

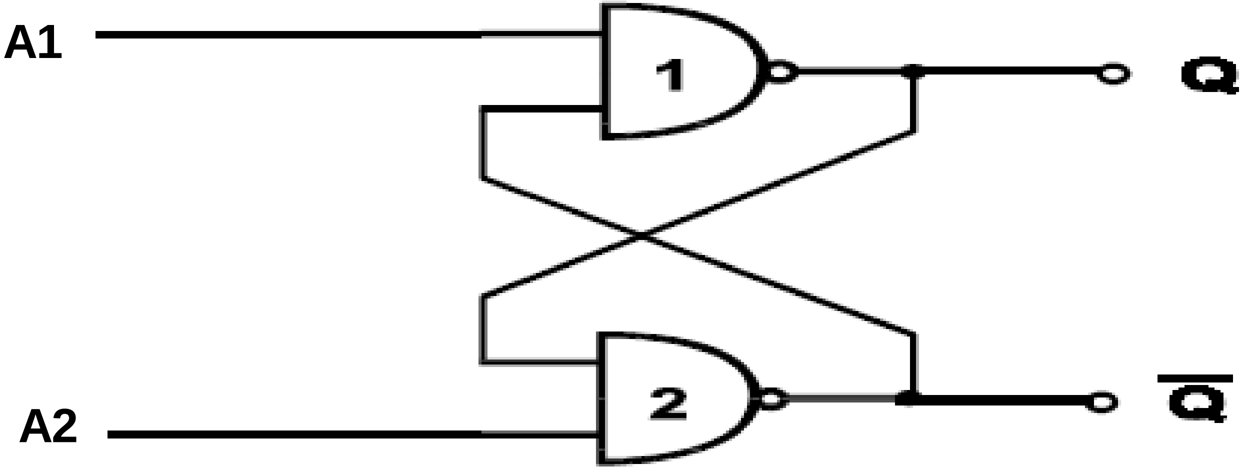


Fig. Cross Coupled inverters as a memory element

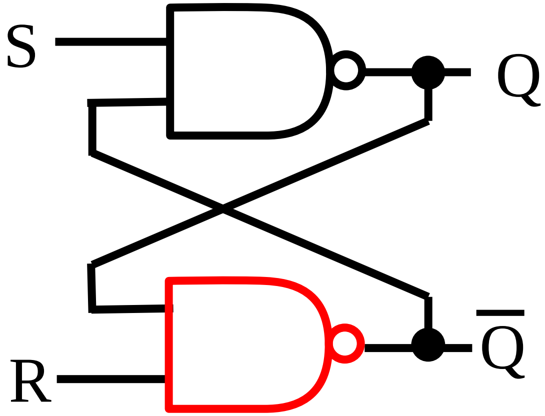
Sequential Logic

Latch:

- **The cross coupled inverter is capable of locking or latching the information, hence is known as a latch.**
- **Disadvantage of this circuit is that we cannot enter the desired digital data into it.**
- **This disadvantage can be overcome by modifying the circuit, which allows us to enter the desired digital data into the circuit.**
- **In latch there is no way of entering the desired digital information to be stored in it. Since when the circuit is switched ON, the circuit switches to one of the stable states. Thus additional gates are added to enter the desired digital information.**
- **The two input terminals are designated as set (S) and reset (R) because S=1 brings the circuit in set state and R=1 brings it to reset or clear state.**

Bistable- Flip Flops made with NANDs

- While analyzing this circuit, as soon as one input is a Low or zero (either a direct input or a feedback input) a NAND has High (1) as an output.
- On the other hand for NORs if an input is High (1) then the output must be a Low (0).



State Table

S	R	Q	Q'
0	1	1	0
1	0	0	1
0	0	1	1
1	1	?	?

S 'sets' Q HI when it is LO

R 'resets' Q LO when it is HI

S = R = LO is a 'disallowed' state
what happens when S = R = HI?

The last state depends on the previous state, since there are two possible output states - thus sequential logic. What happens if you go from 00 to 11?

Sequential Logic

Operation:

Case I: S = R = 0:

- **Outputs of gates 1 & 2 will become 1.**
- **Let Q=0 & $\overline{Q}$=1 initially, hence both the inputs to gate 3 are 1 & the inputs to gate 4 are (01).**
- **Hence gate-3 output i.e. Q=0 & gate-4 output $\overline{Q}$=1.**
- **Thus with S=R=0, there is no change in the state of outputs.**

Sequential Logic

Case II : S = 1, R = 0:

- **Since S=1 & R=0, one of the inputs to gate-3 will be 0. This will force Q output to 1.**
- **Hence both the inputs to gate-4 will be 1. This forces $\overline{Q}$ to 0.**
- **Hence for S=1, R=0 the outputs are Q=1 & $\overline{Q}$=0. This is set state.**

Case III: S = 0, R = 1:

- **If S=0, R=1 then one of the inputs to gate-4 will be 0. This will force the $\overline{Q}$ output to 1.**
- **Hence both the inputs to gate-3 will be 1. This forces Q to 0.**
- **Thus for S=0, R=1, the outputs are Q=0, $\overline{Q}$=1. This is the reset state of clear state.**

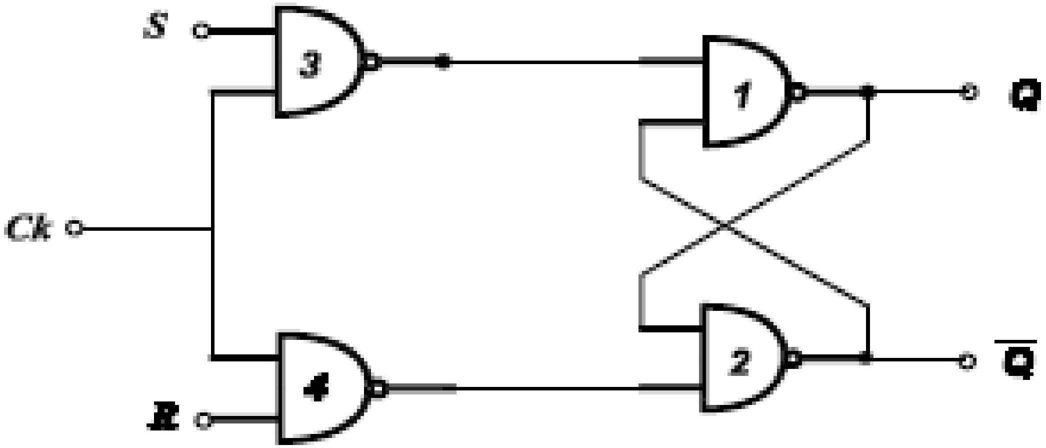
Case IV: S = R = 1:

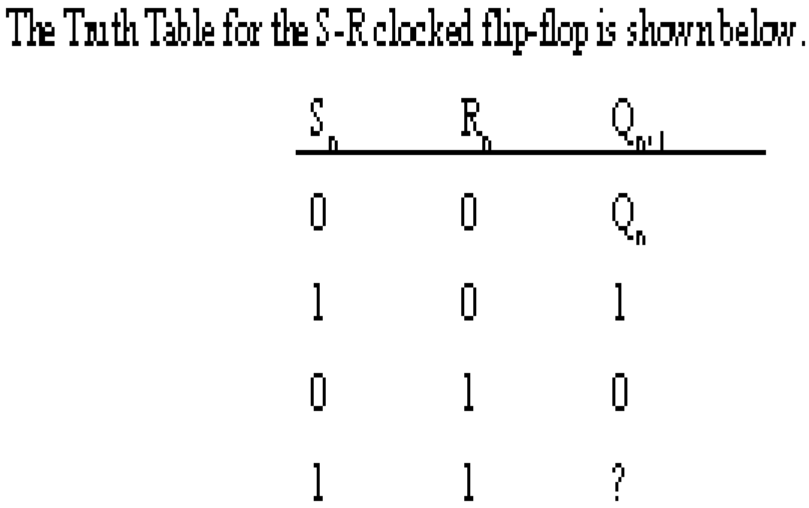
- **If S=R=1 then outputs of gates-1 and 2 will be zero.**
- **Hence one of the inputs to gate-3 and 4 will be 0.**
- **So outputs Q & $\overline{Q}$ both will try to become 1. It is not allowed as Q and $\overline{Q}$ should be complementary. Hence S=R=1 condition is prohibited.**

Sequential Logic

Clocked S-R Flip-Flop:

- ➢ **It is required to set or reset the memory cell in synchronous with a train of pulses known as clock (known as CK), & such circuit is referred to as a clocked set-reset (S-R) Flip-Flop.**
- ➢ **If CK=1, then it performs the operation exactly the same as that of S-R Flip-Flop, and if CK=0 the gates 3 & 4 are inhibited. i.e. their outputs are 1 irrespective of the values of S or R.**
- ➢ **The circuit responds to the inputs S & R only when the clock is present.**





The Truth Table for the S-R clocked flip-flop is shown below.

S_n	R_n	Q_{n+1}
0	0	Q_n
1	0	1
0	1	0
1	1	?

Sequential Logic

Preset & Clear:

- **In clocked S-R Flip-Flop, when the power is switched on, the state of the circuit is uncertain. It may come to set (Q=1) of reset (Q=0) state.**
- **In many cases the FF is to be set or reset i.e. the initial state of the FF is to be assigned. This is achieved by asynchronous inputs, referred to as preset (Pr) and Clear (Cr) inputs.**
- **These inputs may be applied at any time between clock pulses and are not in synchronism with the clock.**
- **If Pr=Cr=1 the Ckt operates in accordance with the T.T of S-R FF.**
- **If Pr=0 & Cr=1, the output of G1(Q) will be 1. Thus all the 3 inputs to G2 will be 1 which make $\overline{Q}$=0, hence Pr=0 sets the FF.**
- **If Pr=1 & Cr=0, the FF is reset, Once the state of the FF is established asynchronously, the asynchronous inputs Pr & Cr must be connected to logic 1 before the next clock is applied.**
- **The condition Pr=Cr=0 must not be used, since this leads to an uncertain state.**
- **Thus in logic symbol, bubbles are used for Pr & Cr inputs, which means these are active low, i.e. the intended function is performed when the signal applied to Pr & Cr is LOW.**

Sequential Logic

Summary of Operation of S-R Flip-Flop:

	Inputs		Output	Operation Performed
CK	Cr	Pr	Q	
1	1	1	Qn+1	Normal FF Operation
0	0	1	0	Clear
0	1	0	1	Preset

Note: The circuit can be set or reset even in the presence of the clock pulse.

J-K Flip-Flop

J-K FLIP-FLOP:

- **The uncertainty in the state of an S-R FF, when Sn=Rn=1 can be eliminated by converting it into a J-K FF.**
- **The data inputs are J & K which are ANDed with $\overline{Q}$ & Q respectively, to obtain S & R inputs. i.e. S=J.$\overline{Q}$ & R = K. Q**
- **Truth Table for J-K Flip-Flop:**

Inputs			Output
Jn	**Kn**		**Qn+1**
0	0		Qn
1	0		1
0	1		0
1	1		$\overline{Q}$n

J-K Flip-Flop

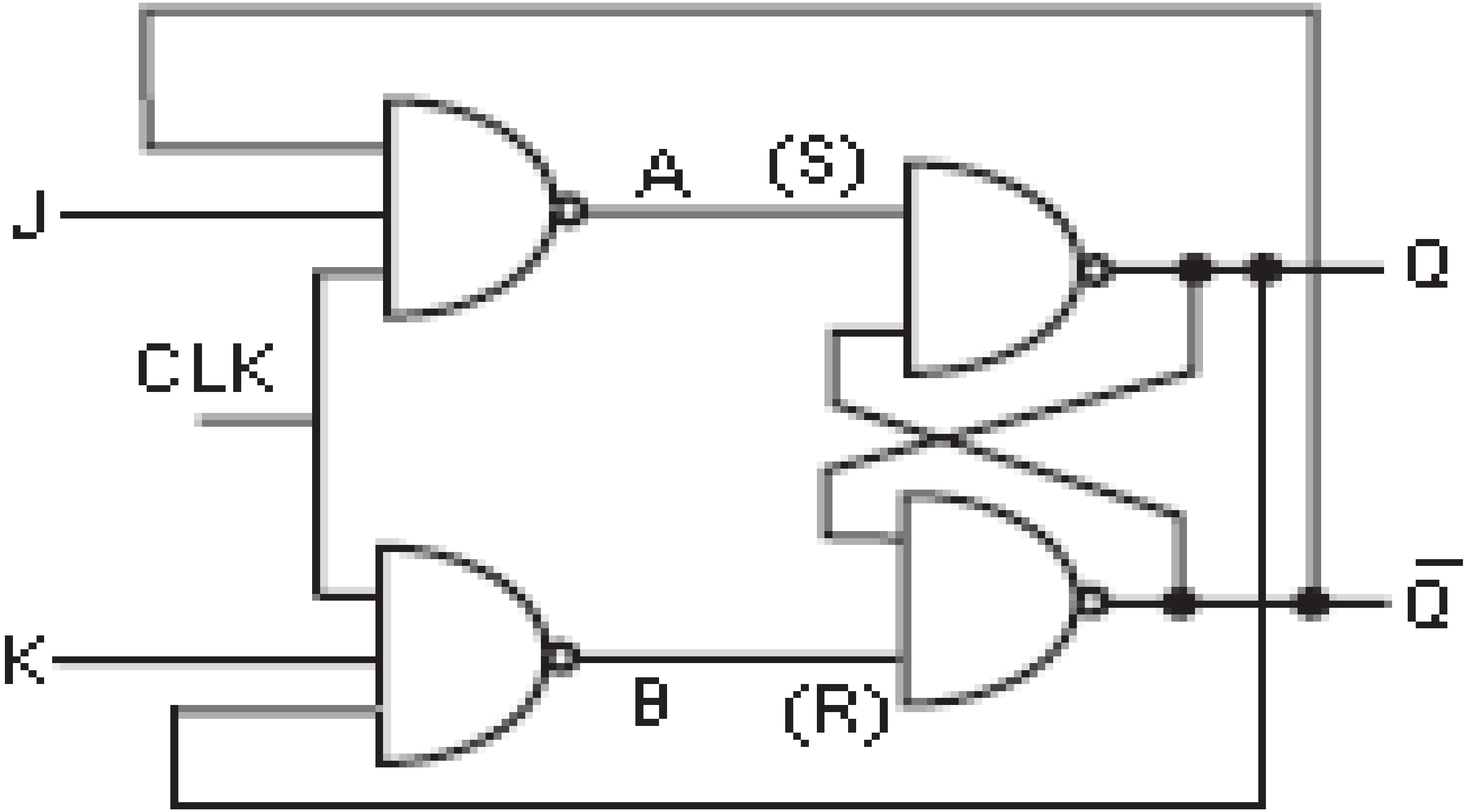
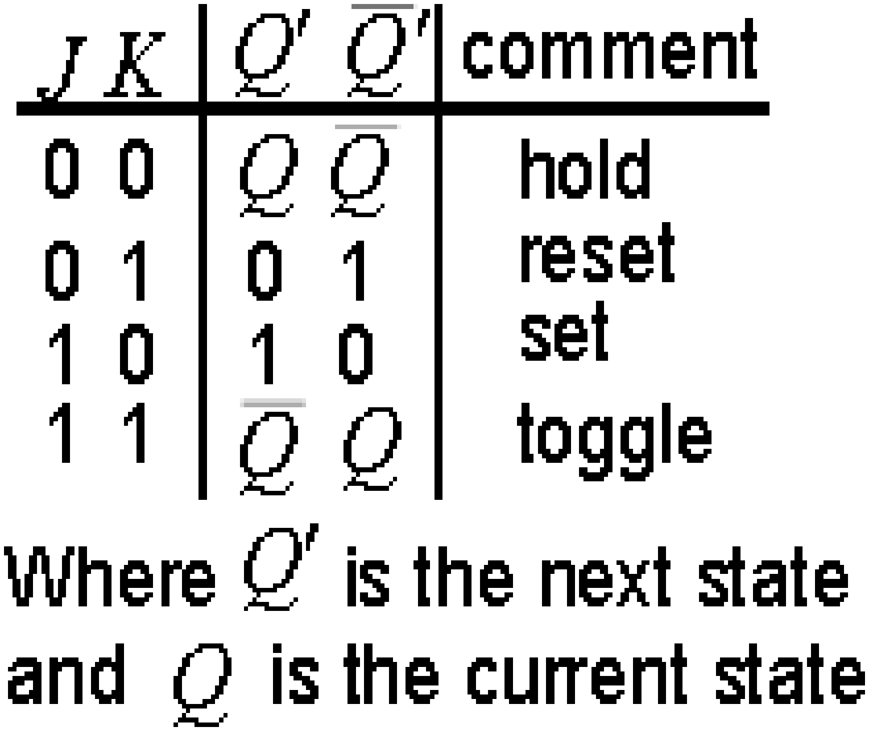


Fig. S-R Flip-Flop Converted into J-K Flip-Flop

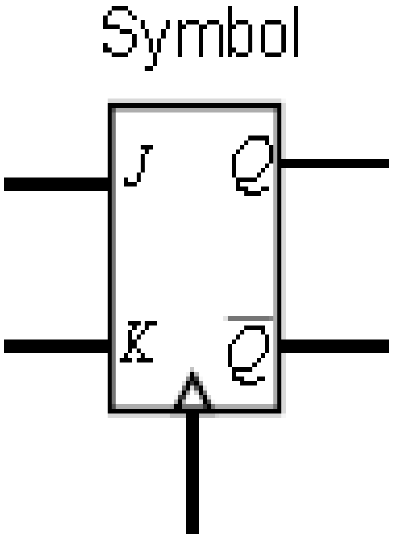
J-K Flip-Flop

- The J-K FF is similar in function to a clocked RS FF, but with the illegal state replaced with a new ‘toggle’ state



J K	Q'	$\overline{Q}'$	comment
0 0	Q	$\overline{Q}$	hold
0 1	0	1	reset
1 0	1	0	set
1 1	$\overline{Q}$	Q	toggle

Where Q' is the next state
and Q is the current state



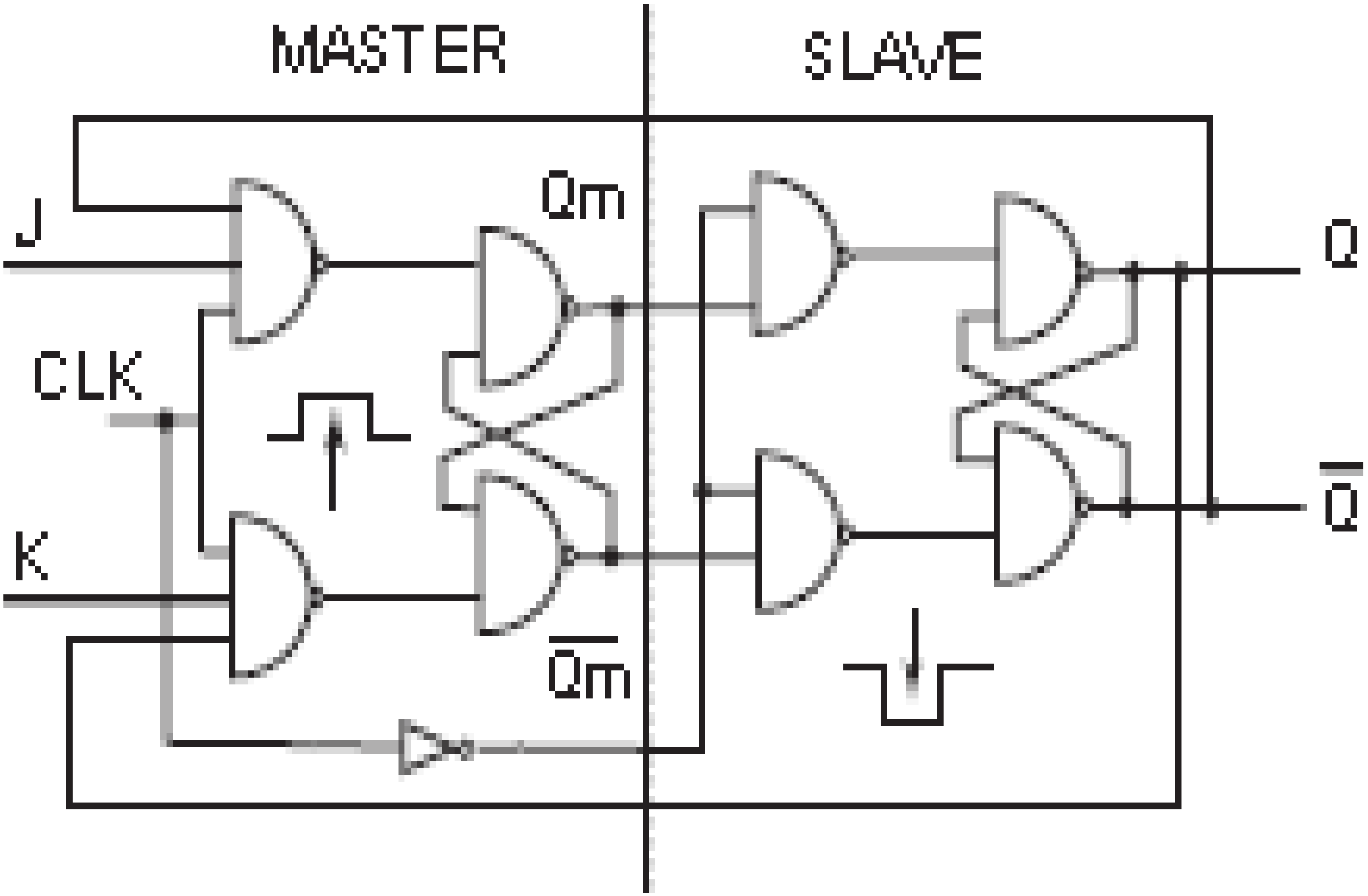
Logic Symbols of J-K FF

Sequential Logic

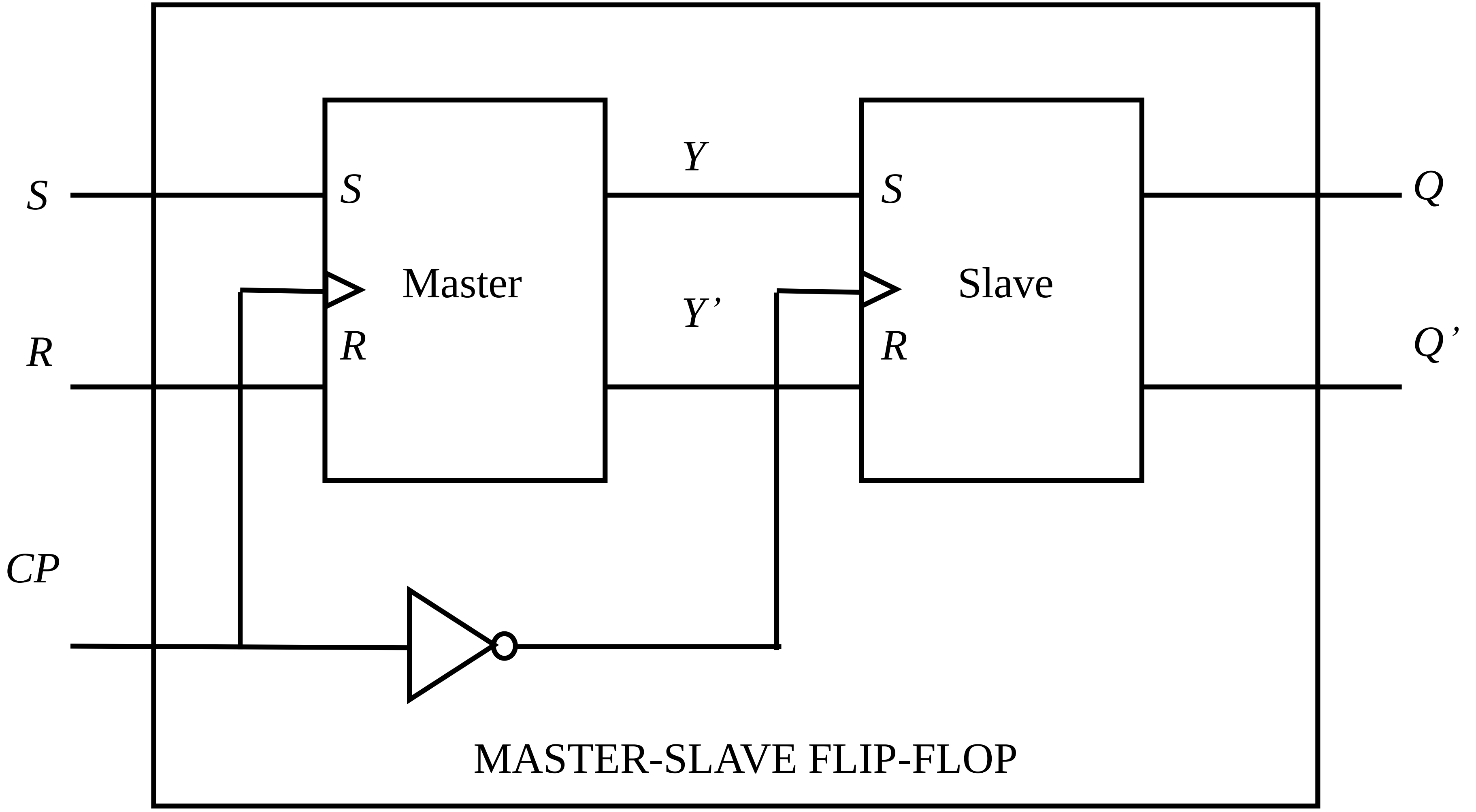
Race- Around Condition:

- **In J-K Flip-Flop, condition for both the inputs as 1 i.e. S=R=1 is taken into consideration by using the feedback connection from outputs to the inputs of the gates.**
- **In J-K FF, the assumption that the inputs do not change during the clock pulse (CK=1), which is not true because of the feedback connections.**
- **For e.g. the inputs J=K=1 & Q=0 and a pulse is applied at the clock input.**
- **After a time interval Δt equal to the propagation delay through two NAND gates in series, the output will change to Q=1.**
- **Now J=K=1 & Q=1 and after another time interval of Δt the output will change back to Q=0. Thus for the duration tp of clock pulse, the output will oscillate back and forth between 0 & 1, & at the end of the clock pulse, the value of Q is uncertain. This situation is called as race around condition.**
- **It can be avoided if tp< Δt <T. The best way is to use a master-slave J-K Flip-Flop.**

Master-Slave J-K Flip-Flop

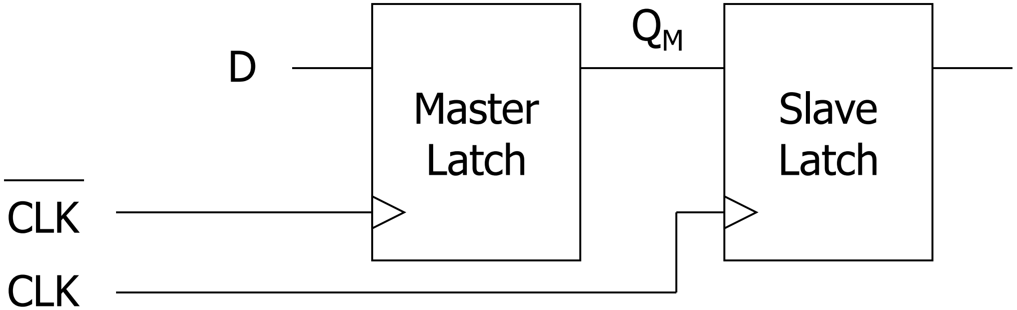


Master-Slave Flip-Flop

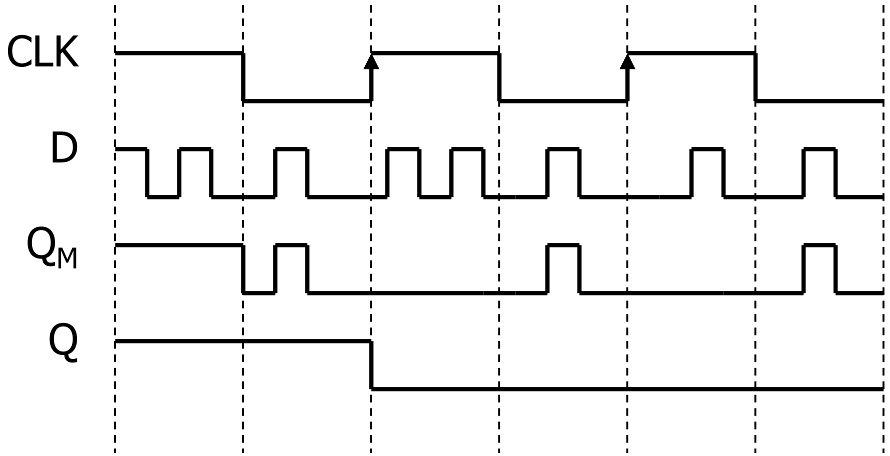


Master-Slave Edge-Triggered Flip-Flop

- **Can connect two level-sensitive latches in <u>Master-Slave</u> configuration to form <u>edge-triggered flip-flop</u>**
- **<u>Master</u> latch "catches" value of "D" at "Q_M" when CLK is low**
- **<u>Slave</u> latch causes "Q" to change only at rising edge of CLK**



2 x 8 = 16 Transistors



D-Type Flip-Flop

- It has only one input referred to as D-input or data input.
- The output Qn+1 at the end of the clock pulse equals the input Dn before the clock pulse.
- Input data appears at the output at the end of the clock pulse. Thus transfer of data from the input to the output is delayed and hence the name delay (D) FF.
- It is either used as a delay device or as a latch to store 1 bit of binary information.

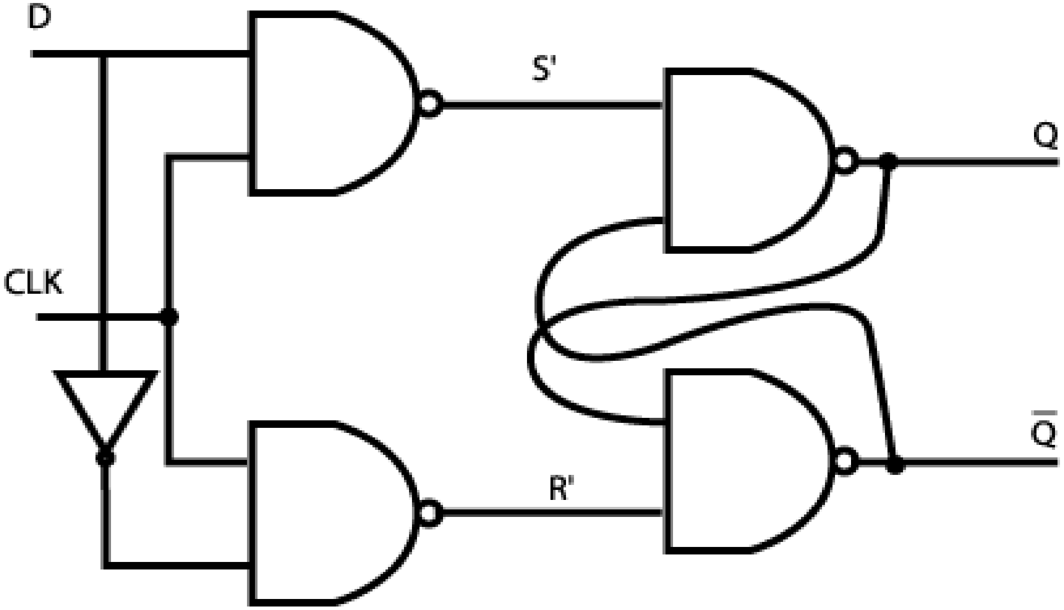
Truth Table:

Input	Output
Dn	Qn+1
0	0
1	1

Sequential Logic

D Flip-Flop:-

Preset	Clear	D	Clock	Qn+1	$\overline{Qn+1}$
1	1	0	⎍	0	1
1	1	1	⎍	1	0



Sequential Logic

T- Type Flip-Flop:

- In J-K FF, if J=K, the resulting FF is referred to as a T-type FF.
- It has only one input, referred to as T-input.
- From the truth table of T-type FF, if T=1 it acts as a toggle switch. For every clock pulse, the output Q changes.
- An S-R FF cannot be converted into a T-type FF since S=R=1 is not allowed. But its logic symbol can be used as a toggle switch. i.e. the output of Q changes with every clock pulse.

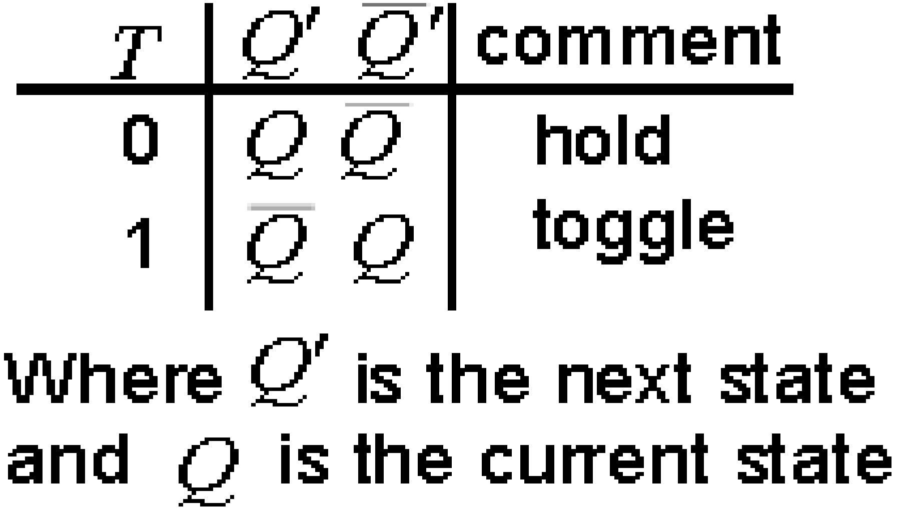
Truth Table:

Input	Output
Tn	Qn+1
0	Qn
1	$\overline{Qn}$

Sequential Logic

T Flip-Flop

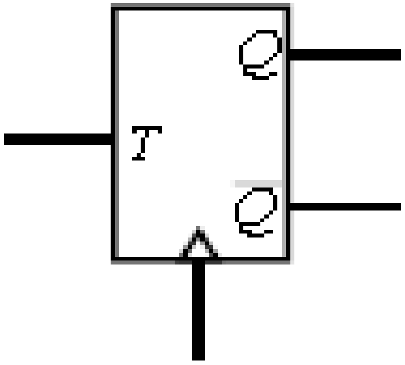
- This is essentially a J-K FF with its J and K inputs connected together and renamed as the T input



T	Q'	$\overline{Q}'$	comment
0	Q	$\overline{Q}$	hold
1	$\overline{Q}$	Q	toggle

Where Q' is the next state
and Q is the current state

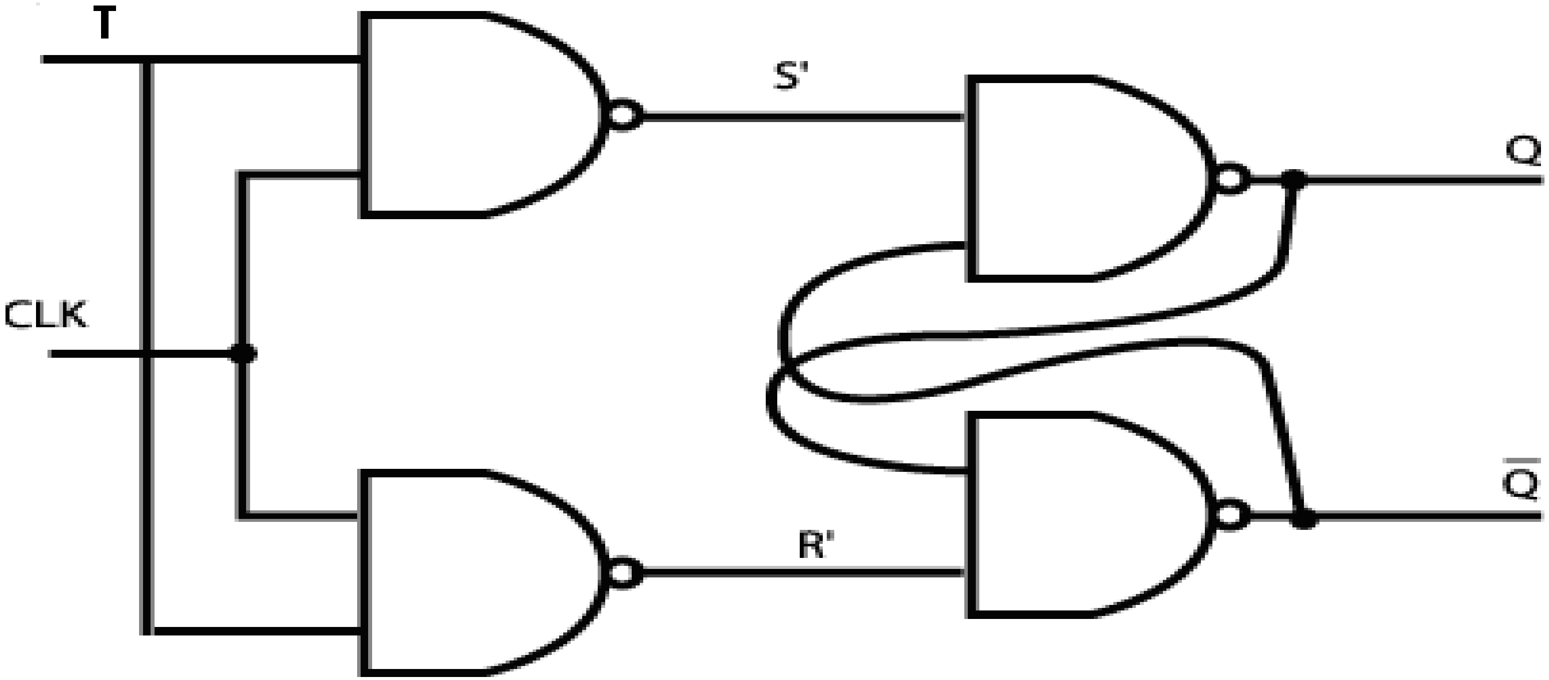
Symbol



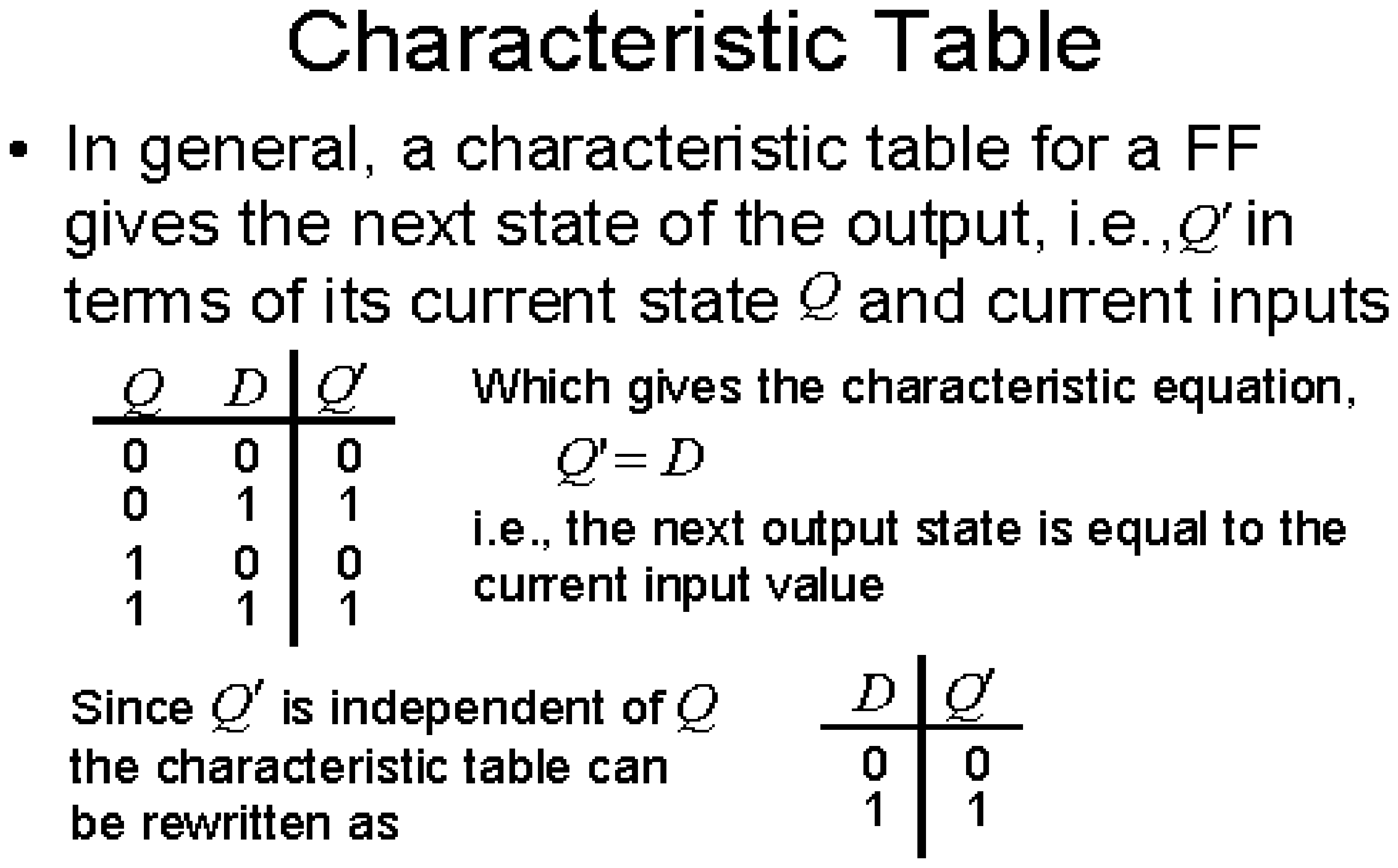
Sequential Logic

T Flip-Flop:-

Preset	Clear	T	Clock	Qn+1	$\overline{Qn+1}$
1	1	0	⎍	Qn	$\overline{Qn}$
1	1	1	⎍	$\overline{Qn}$	Qn



Sequential Logic



Characteristic Table

- In general, a characteristic table for a FF gives the next state of the output, i.e., Q' in terms of its current state Q and current inputs

Q	D	Q'
0	0	0
0	1	1
1	0	0
1	1	1

Which gives the characteristic equation,

$$Q' = D$$

i.e., the next output state is equal to the current input value

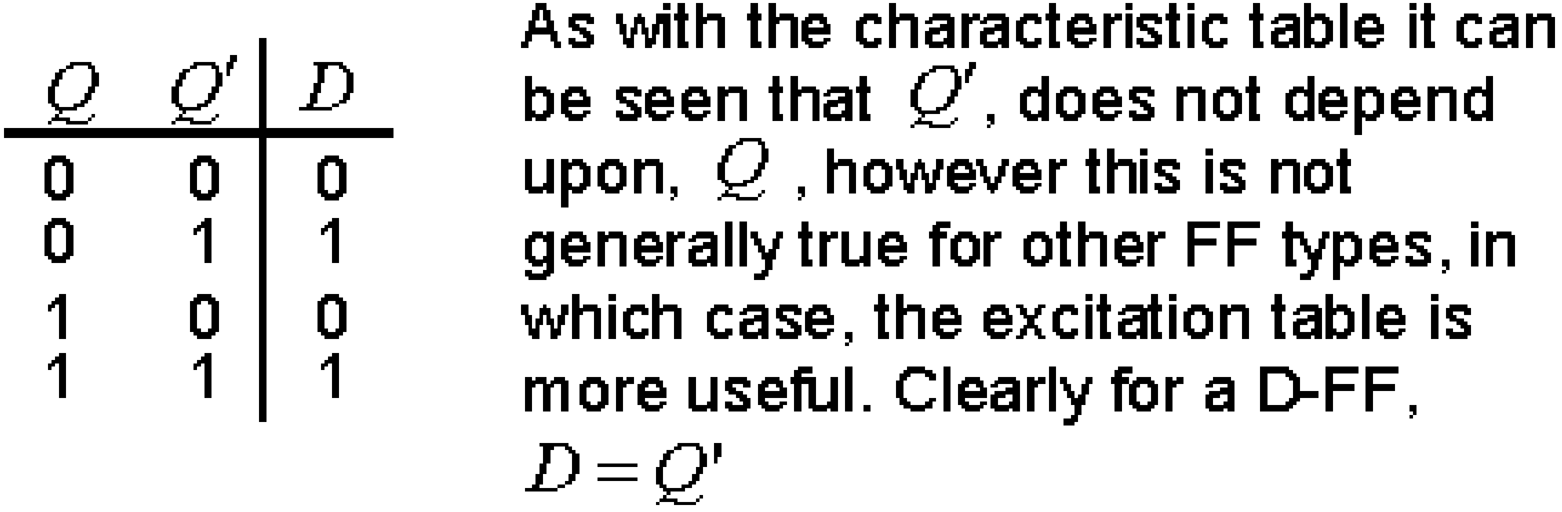
Since Q' is independent of Q the characteristic table can be rewritten as

D	Q'
0	0
1	1

Sequential Logic

Excitation Table

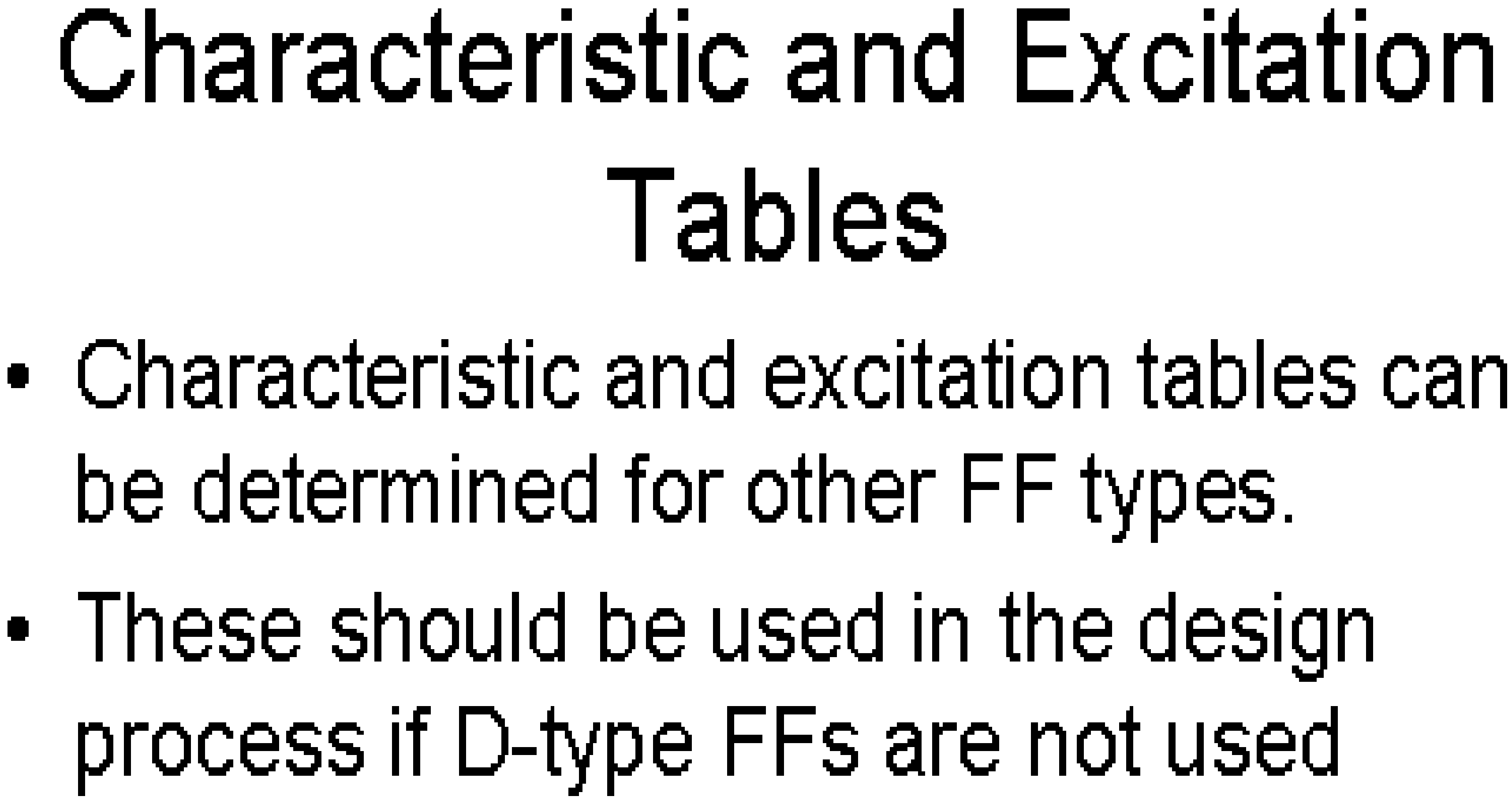
- The characteristic table can be modified to give the excitation table. This table tells us the required FF input value required to achieve a particular next state from a given current state



Q	Q'	D
0	0	0
0	1	1
1	0	0
1	1	1

As with the characteristic table it can be seen that Q', does not depend upon, Q , however this is not generally true for other FF types, in which case, the excitation table is more useful. Clearly for a D-FF, $D = Q'$

Sequential Logic



Characteristic and Excitation Tables

- Characteristic and excitation tables can be determined for other FF types.
- These should be used in the design process if D-type FFs are not used

Sequential Logic

Excitation Table of Flip-Flop:

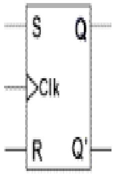
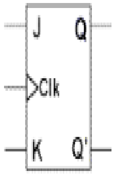
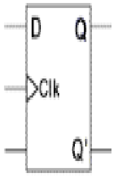
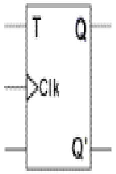
- **The truth table of a FF is referred as the characteristics table & specifies the operational characteristics of the FF.**
- **To design sequential circuits, it is needed that the input conditions corresponding to present state & next state is specified.**
- **Thus we have to find the input conditions that must prevail to cause the desired transition of the state.**
- **Present state & Next state means circuit prior to & after the clock pulse applied.**

Sequential Logic

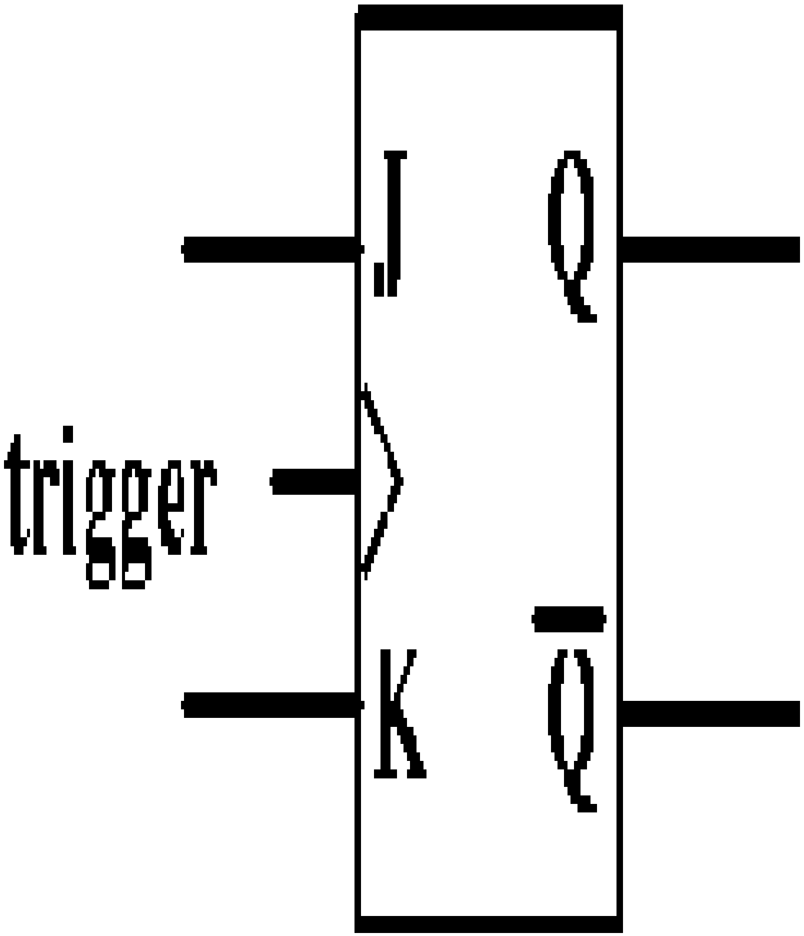
Excitation Table of Flip-Flops:

Present	Next	S-R	FF	J-K	FF	T-FF	D-FF
State	State	Sn	Rn	Jn	Kn	Tn	Dn
0	0	0	x	0	x	0	0
0	1	1	0	1	x	1	1
1	0	0	1	x	1	1	0
1	1	x	0	x	0	0	1

Summary of the Types of Flip-flop Behavior

<table>
<tr><th>FLIP-FLOP NAME</th><th>FLIP-FLOP SYMBOL</th><th>CHARACTERISTIC TABLE</th><th>CHARACTERISTIC EQUATION</th><th>EXCITATION TABLE</th></tr>
<tr><td>SR</td><td> </td><td><table><tr><th>S</th><th>R</th><th>$Q_{(next)}$</th></tr><tr><td>0</td><td>0</td><td>Q</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>?</td></tr></table></td><td>$Q_{(next)} = S + R'Q$
$SR = 0$</td><td><table><tr><th>Q</th><th>$Q_{(next)}$</th><th>S</th><th>R</th></tr><tr><td>0</td><td>0</td><td>0</td><td>X</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>X</td><td>0</td></tr></table></td></tr>
<tr><td>JK</td><td> </td><td><table><tr><th>J</th><th>K</th><th>$Q_{(next)}$</th></tr><tr><td>0</td><td>0</td><td>Q</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>Q'</td></tr></table></td><td>$Q_{(next)} = JQ' + K'Q$</td><td><table><tr><th>Q</th><th>$Q_{(next)}$</th><th>J</th><th>K</th></tr><tr><td>0</td><td>0</td><td>0</td><td>X</td></tr><tr><td>0</td><td>1</td><td>1</td><td>X</td></tr><tr><td>1</td><td>0</td><td>X</td><td>1</td></tr><tr><td>1</td><td>1</td><td>X</td><td>0</td></tr></table></td></tr>
<tr><td>D</td><td> </td><td><table><tr><th>D</th><th>$Q_{(next)}$</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table></td><td>$Q_{(next)} = D$</td><td><table><tr><th>Q</th><th>$Q_{(next)}$</th><th>D</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td></tr>
<tr><td>T</td><td> </td><td><table><tr><th>T</th><th>$Q_{(next)}$</th></tr><tr><td>0</td><td>Q</td></tr><tr><td>1</td><td>Q'</td></tr></table></td><td>$Q_{(next)} = TQ' + T'Q$</td><td><table><tr><th>Q</th><th>$Q_{(next)}$</th><th>T</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table></td></tr>
</table>

Sequential Logic



J	K	Q (t)	Q (t+1)	
0	0	0	0	no change
0	0	1	1	
0	1	0	0	clears
0	1	1	0	
1	0	0	1	sets
1	0	1	1	
1	1	0	1	toggles
1	1	1	0	

Sequential Logic

- **In general model for conversion from one type of FF to another type, we design the combinational logic decoder (conversion logic) for converting new input definitions into input codes which will cause the given FF to perform as desired.**
- **To design the conversion logic we need to combine the excitation tables for both FF's & make a truth table with data inputs & Q as the inputs and the inputs of the given FF as the outputs.**

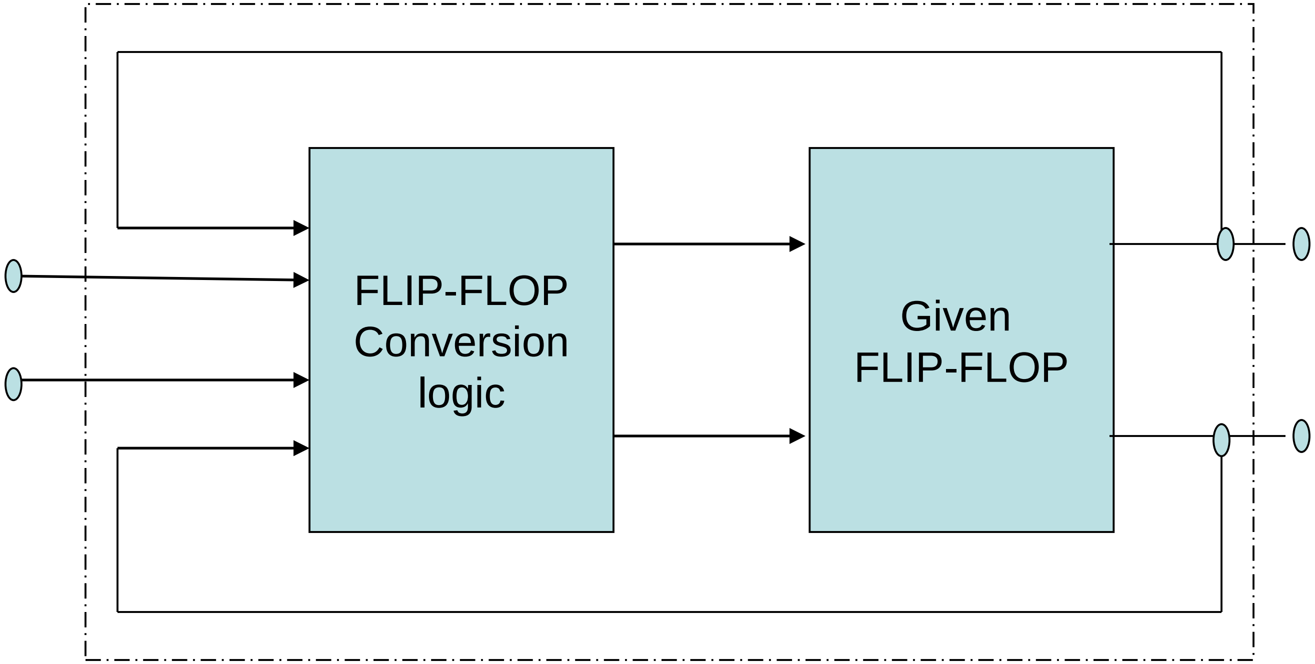


Fig. General Model used to convert one type of Flip-Flop to another type

Sequential Logic

Converting One Type of FF to another:

- **S-R FF to J-K FF:**

Steps for Conversion:

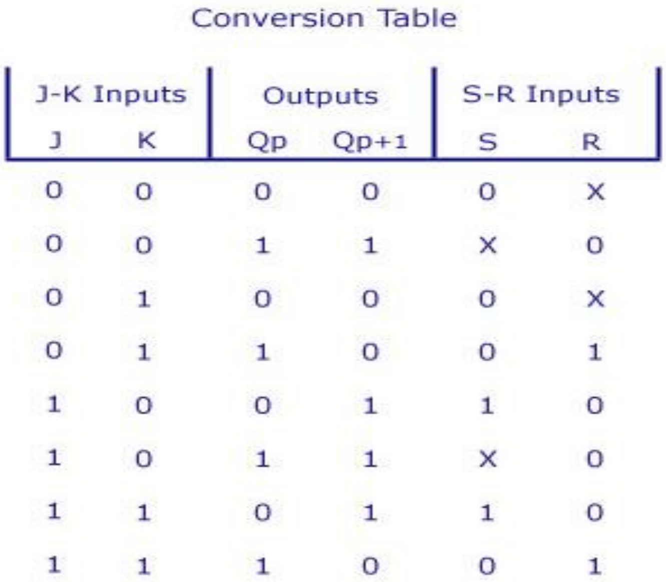
- **Prepare the excitation table**
- **Prepare K-Maps & derive minimized expression.**
- **Draw the logic diagram**

Data Inputs		Outputs		Inputs to S-R FF		Output	
Jn	Kn	Qn	$\overline{Q}$n	Sn	Rn		
0	0	0	1	0	0	0	Qn
0	0	1	0	0	0	1	
1	0	0	1	1	0	1	1
1	0	1	0	0	0	1	
0	1	0	1	0	0	0	0
0	1	1	0	0	1	0	
1	1	0	1	1	0	1	$\overline{Q}$n
1	1	1	0	0	1	0	

Flip-Flop Conversion

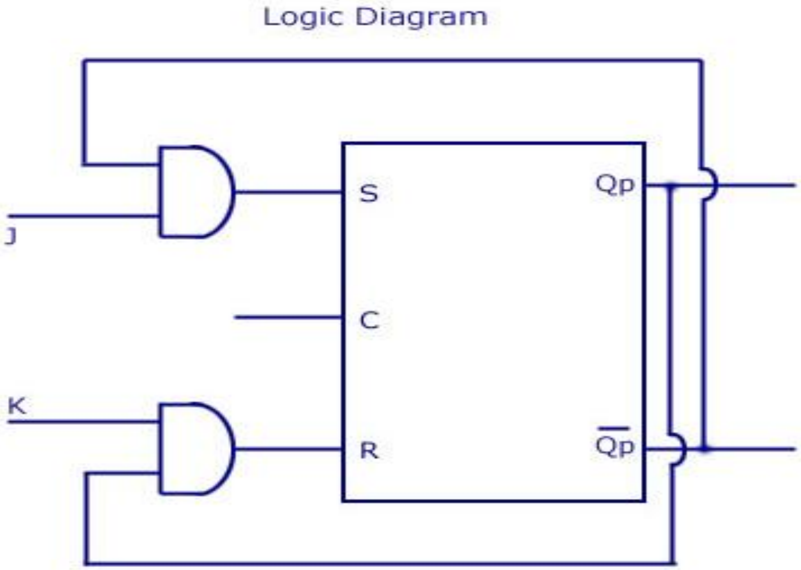
- S-R To J-K Flip-Flop

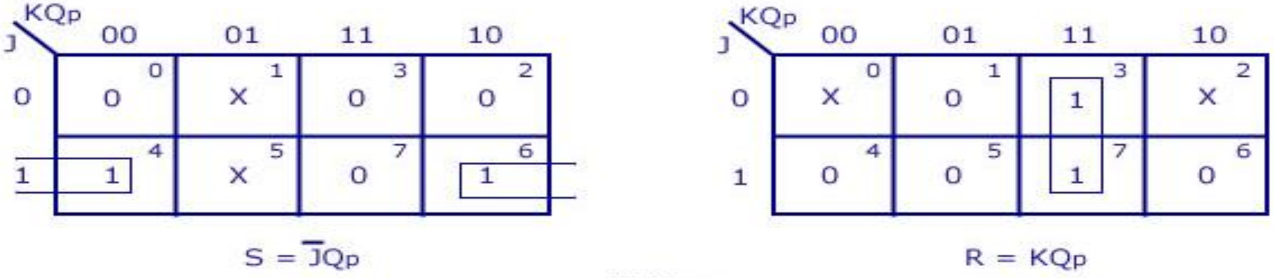
Conversion Table



J-K Inputs		Outputs		S-R Inputs	
J	K	Qp	Qp+1	S	R
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	X	0
1	1	0	1	1	0
1	1	1	0	0	1

Logic Diagram





J \ KQp	00	01	11	10
0	0	X	0	0
1	1	X	0	1

$S = \overline{J}Qp$

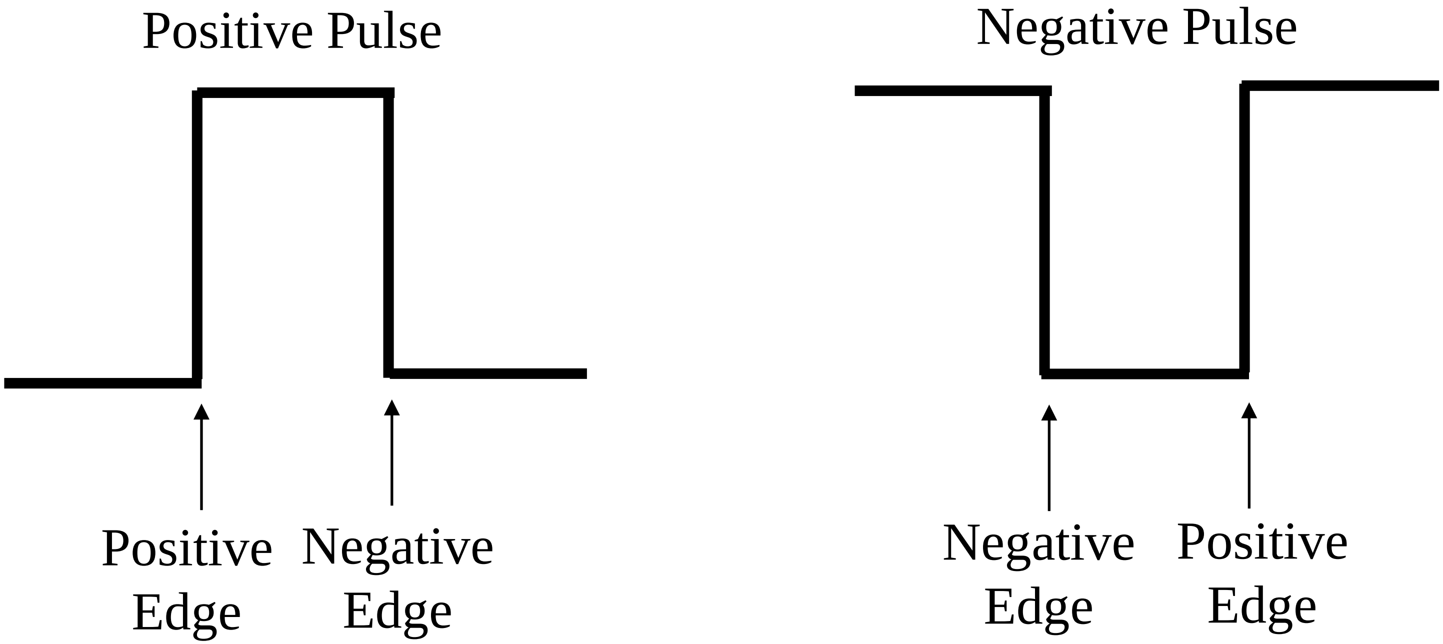
J \ KQp	00	01	11	10
0	X	0	1	X
1	0	0	1	0

$R = KQp$

K-Map

www.CircuitsToday.com

Clock Pulse Definition



Edges can also be referred to as leading and trailing.

Sequential Logic

Flip-Flops

- ➢ **Flip-flops are the fundamental element of sequential circuits**
  - **– bistable**
  - **– (gates are the fundamental element for combinational circuits)**
- ➢ **Flip-flops are essentially 1-bit storage devices**
  - **– outputs can be set to store either 0 or 1 depending on the inputs**
  - **– even when the inputs are de-asserted, the outputs retain their prescribed value**
- ➢ **Flip-flops have (normally) 2 complimentary outputs Q & $\overline{Q}$.**
- ➢ **Three main types of flip-flop**
  - **– R-S J-K D-type**

Flip-flop

- **Flip-flop is also known as the basic digital memory circuit.**
- **It has two stable states, memory logic 1 state & logic 0 state.**
- **We can design it, by using either NOR or NAND gates.**
- **A flip-flop can be designed by using the fundamental circuit which is also called as cross coupled inverter circuit, as the NAND gates in this circuit basically act as inverter.**
- **The cross coupled inverter is capable of clocking or batching the information, & hence this circuit is also called as latch.**
- **The disadvantage of the circuit is that we can’t enter the disturbed digital signal as data into it.**
- **Latch is a distable element with two stable states.**
- **It has two outputs, Q & $\overline{Q}$ which are complement of each other.**
- **Latch is a sequential logic circuit which checks all its inputs continuously & will change its output as soon as the input changes without waiting for the clock signal.**
- **Generally an enable signal is provided for a latch when the enable signal is active the output will change as soon as there is a change in input.**
- **S-R (Set-Reset) latch is the simplest type of latch.**

Flip-flop

- Difference between a latch & a Flip-flop.
- Latch & Flip-flop both are basically the bistable elements.
- Latch is a dual triggered flip-flop but flip-flop is a sequential circuit which generally samples inputs & changes its outputs only at particular instants of times & not continuously.
- Five types of flip-flops are as follows: S-R Flip-flop, J-K Flip-flop, Master Slave J-K Flip-flop, T- Flip-flop, D- Flip- flop
- In S-R flip-flop output is uncertain and is not according to assumption when both S & R are equal to 1.
- When initial conditions of flip-flops is to be set then we use the preset.
- Disadvantages of output being indeterminate for S-R flip-flop is overcome using J-K flip-flop.
- For △ t time period which is the propagation delay time of flip-flop, the output fluctuates between 0 and 1 & therefore is not certain. Fluctuation of output reduces life time of IC.
- In computer memory master slave principle is used to drive is slave while the other is master.
- D- Flip-flop can be used as delay or latch element in the circuit to delay the output.
- T- Flip-flop is used for toggling input. In this case output is always the complement of input.

Sequential Logic

Application of Flip-Flop:

- **Bounce Elimination Switch**
- **Latch**
- **Registers**
- **Counters**
- **Memory Elements**

Bounce Elimination Switch: (Fig 7.23 on Pg.256 R. P. Jain)

- **In sequential circuit, if a 1 is to be entered through switch, then the switch is thrown to the corresponding position.**
- **But when the switch is thrown to position 1, the output oscillates between 0 & 1 for some time due to make & break (bouncing) of the switch at the point of contact before coming to rest.**
- **This changes the output of the circuit and creates difficulties in the operation of the system. This problem is eliminated by using bounce-elimination switch.**

Sequential Logic

Registers:

- It is composed of a group of FF’s to store a group of bits (word).
- FF can store 1-bit of digital information & also called as 1-bit register.
- The number of FF required is equal to the number of bits in binary word.
- E.g. 8085 have seven 8-bit registers called as general purpose register & five 1-bit register called as flags.
- The data can be entered in serial (one bit at a time) or in parallel form (all the bits simultaneously) and can be retrieved in the serial or parallel form.
- Data in serial form is also called as temporal code & in parallel form as spacial code.
- For serial input/output it requires only one line for data input & one line for data ouput.
- Whereas for parallel input/output it requires lines equal to the number of bits.

Sequential Logic

Registers:

- **Registers are classified depending upon the way in which data are entered & retrieved.**
- **There are four possible modes of operation:**

1. **Serial-In, Serial-Out (SISO)**
2. **Serial-In, Parallel-Out (SIPO)**
3. **Parallel-In, Serial-Out (PISO)**
4. **Parallel-In, Parallel-Out (PIPO)**

Counters:

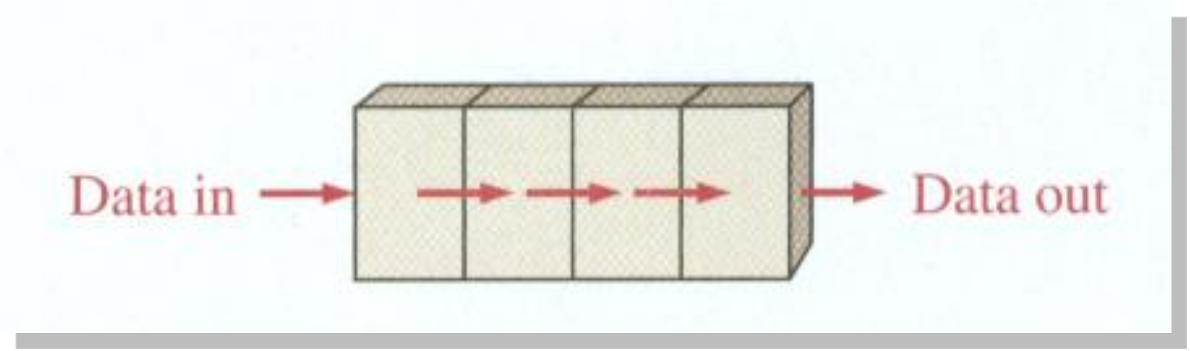
- **A circuit used for counting the pulses is known as a counter.**
- **Most widely used sequential circuits are registers & counters.**
- **Counters are composed of Flip-Flops.**
- **E.g. A circuit with n FF's has 2^n possible states. Therefore a 3-bit counter can count from decimal 0 to 7.**
- **Flip-Flop's are cleared by applying logic 0 at the clear input terminal momentarily.**
- **For normal counting operation, it is maintained at logic 1.**

Basic shift register functions

- A register is a digital circuit with basic functions:
    - Data storage and Data movement
- The storage capacity of a register is the total number of bits it can retain.
- Shift registers consists of an arrangement of flips-flops.
- Each stage (flip-flop) in a shift register represents one bit of storage capacity.
- The shifting capacity permits the movement of data from stage to stage within the register or into or out of the register upon application of clock pulses.
- The basic difference between a register and a counter is that a register has no specified sequence of states, except in certain very specialized applications.
- A register is used solely for storing and shifting data.

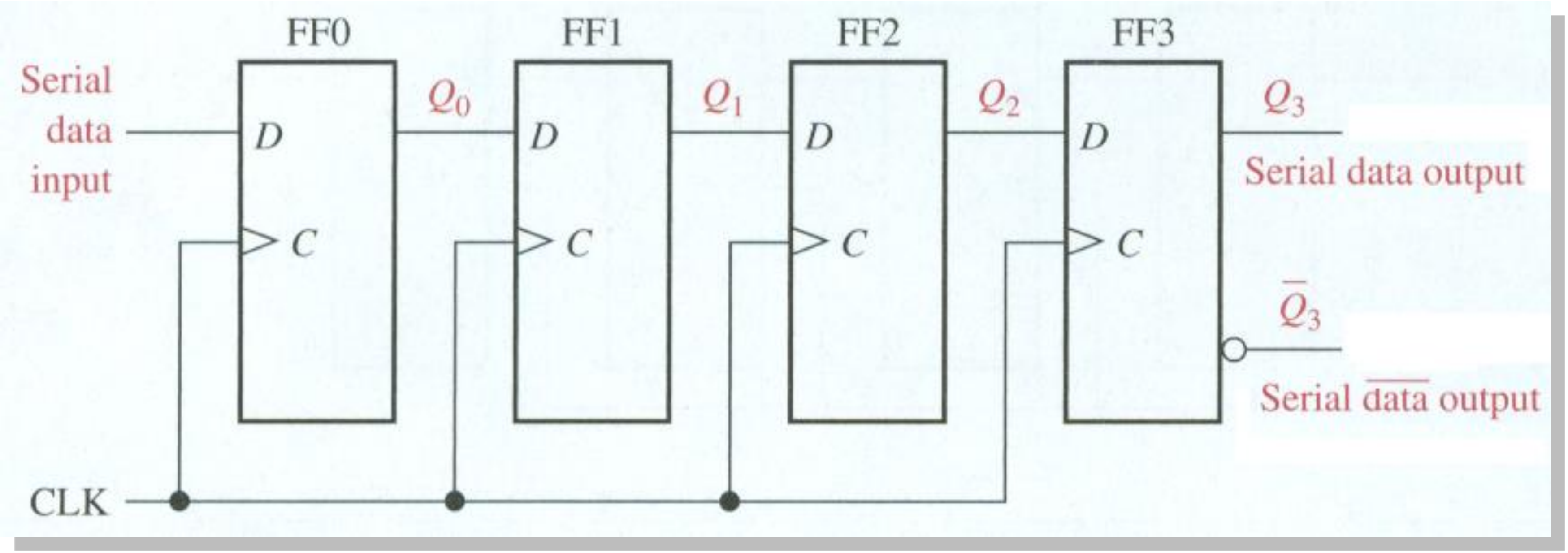
Serial in/serial out shift registers

- It accepts data serially, one bit at a time on a single line, and produces the sorted information on its output also in a serial form

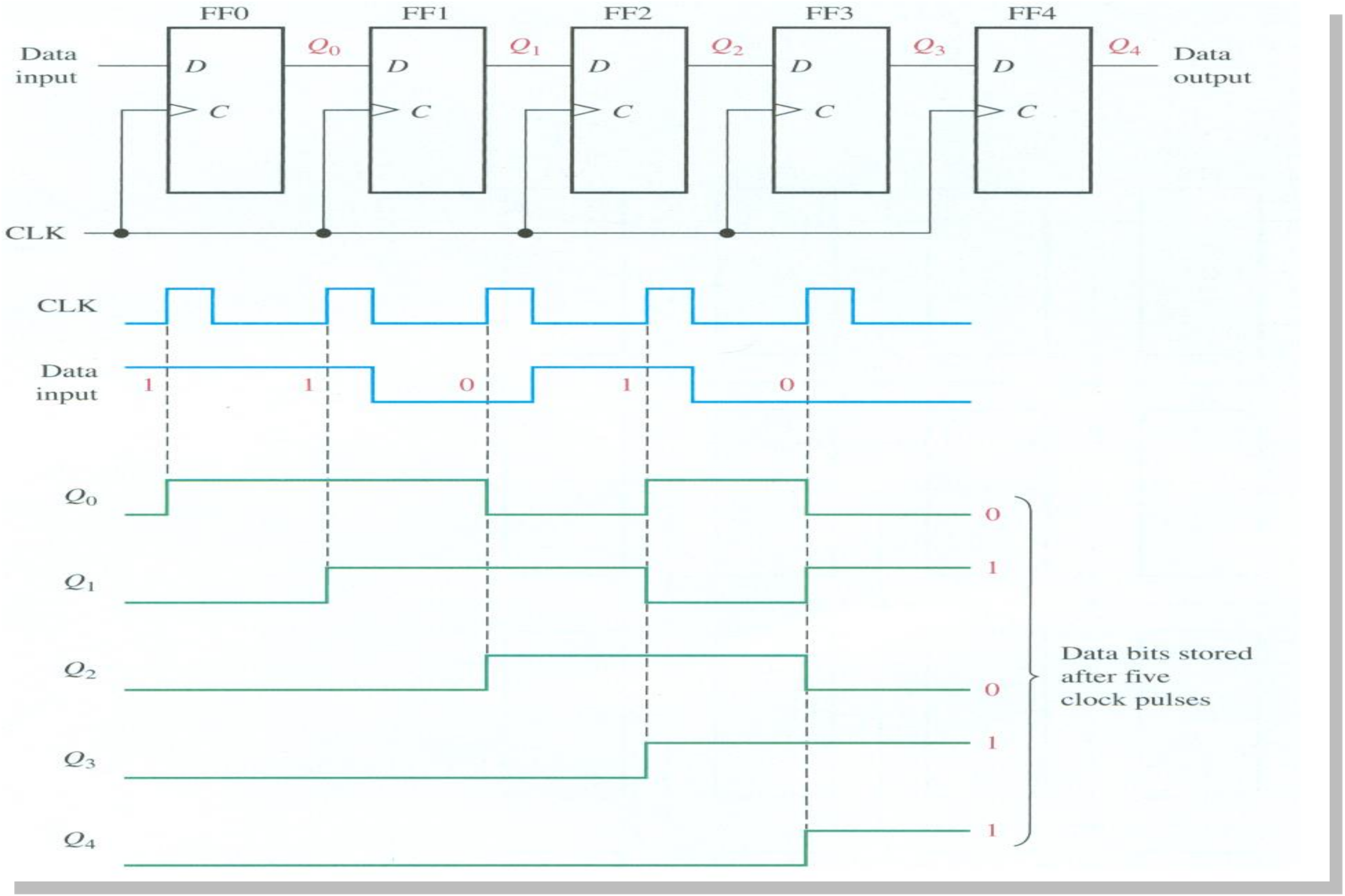


Serial in/serial out shift registers

- 4 bit register
- It needs 4 clock pulses to store 4 bits
- Example:
  - Illustrate entry of the 4 bits 1010 into the register.
  - Illustrate serially shifting the 4 bits out of the register, i.e. clearing the register.

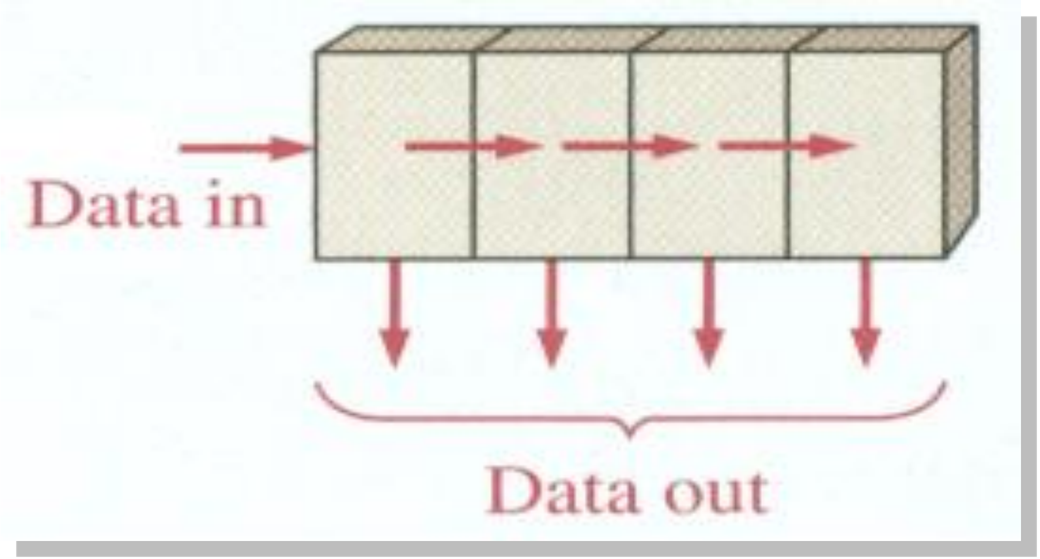


Example: Show the states of the 5-bit shift register for the specified data input and clock waveforms. The registered is initially cleared.



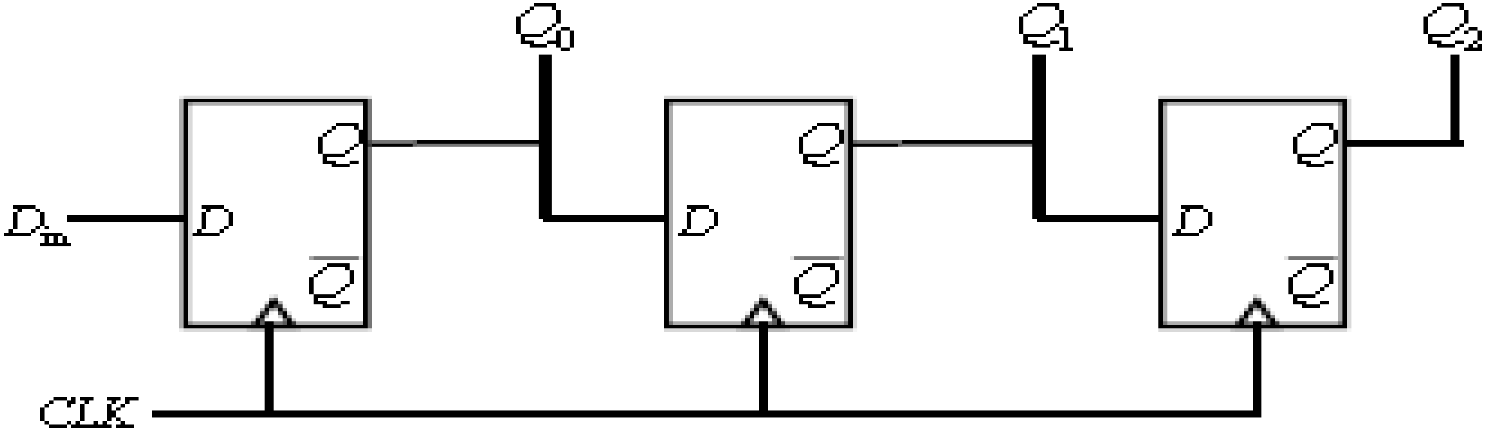
Serial in/parallel out shift registers

- Data bits are entered serially as illustrated before
- Once the data are stored, the output of each stage is available on its output line.



Shift Register

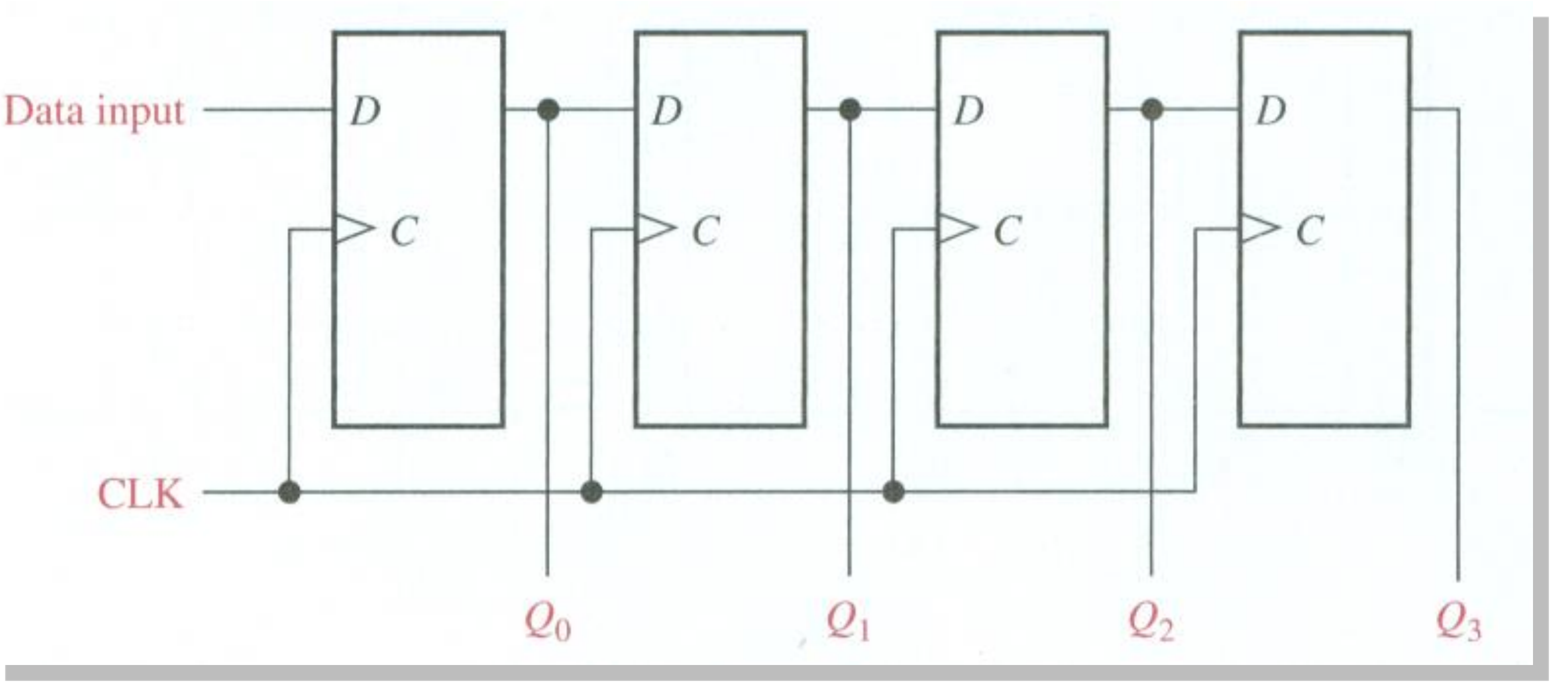
- A shift register can be implemented using a chain of D-type FFs



- Has a serial input, D_{in} and parallel output Q_0, Q_1 and Q_2.
- See data moves one position to the right on application of clock edge

Serial in/parallel out shift registers

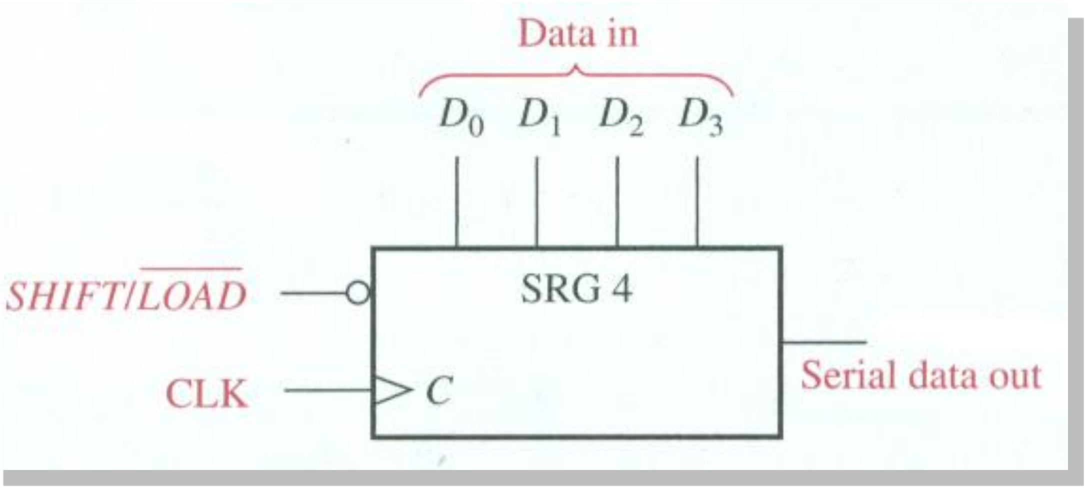
- 4-bit register



Parallel In/Serial Out Shift Registers

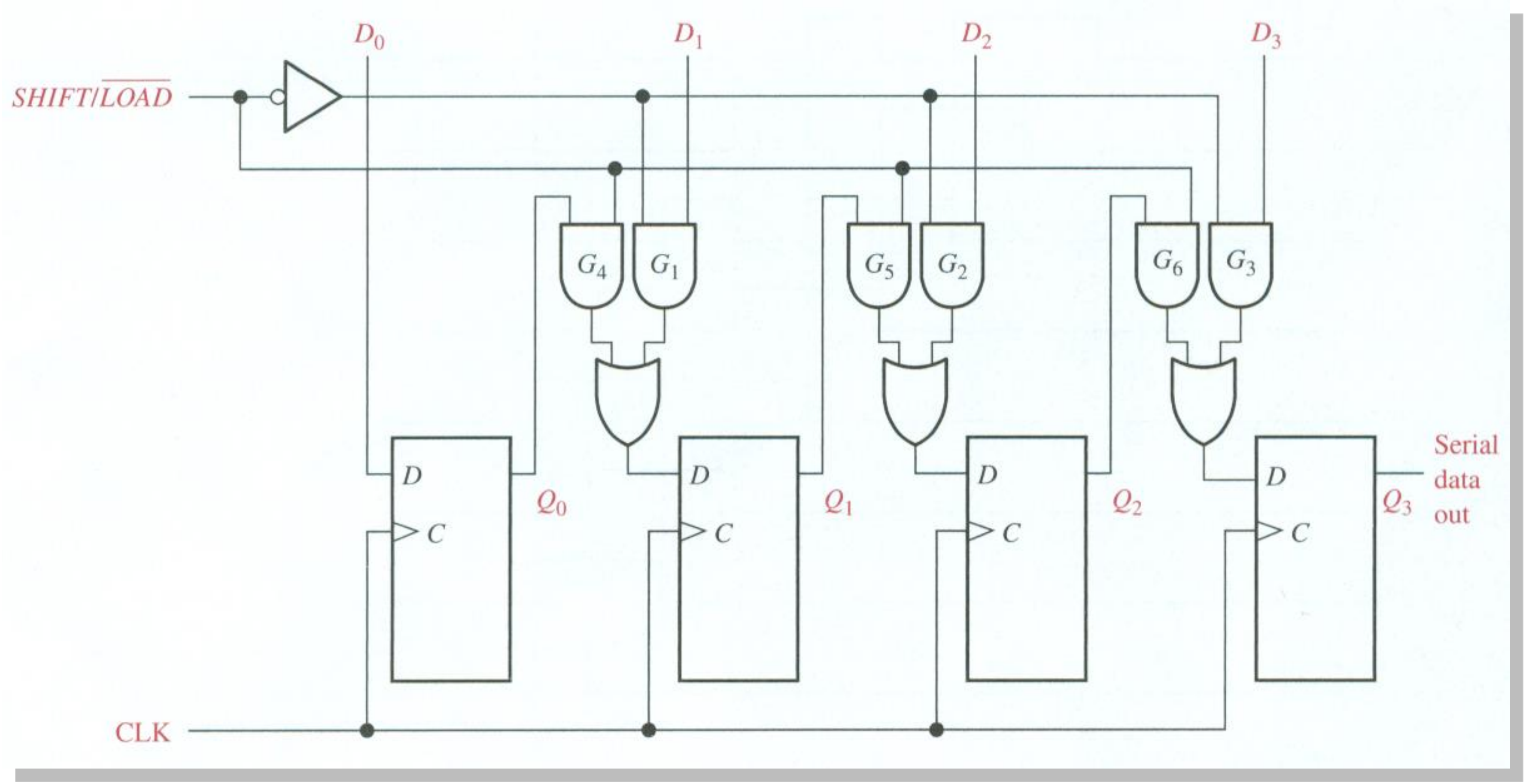
- The bits are entered simultaneously into their respective stages.
- The serial output appears bit by bit per clock pulse.
- To store 4 bits, we need 1 clock pulse
- To shift them out them, we need another 3 clock pulses.

- 4-bit parallel in/serial out



Parallel In/Serial Out Shift Registers

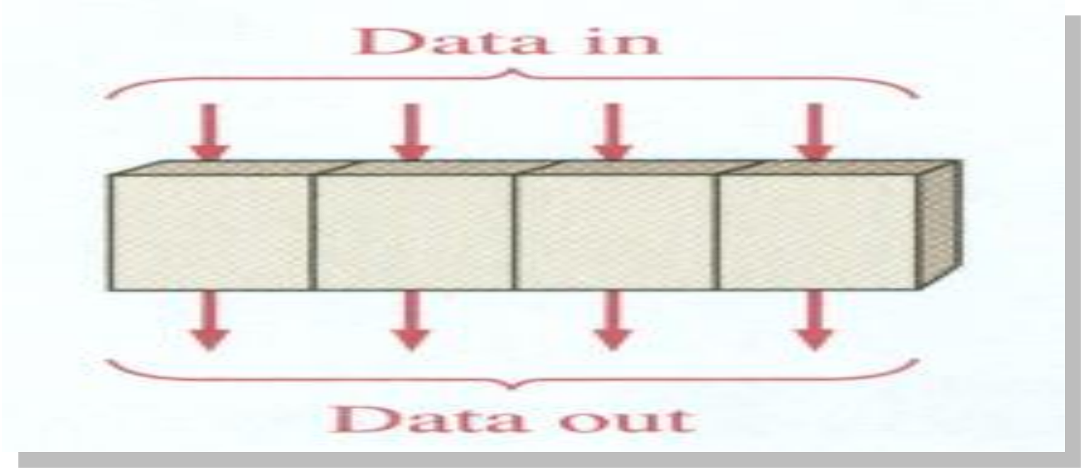
4-bit parallel in/serial out



Parallel In/Parallel Out Shift Registers

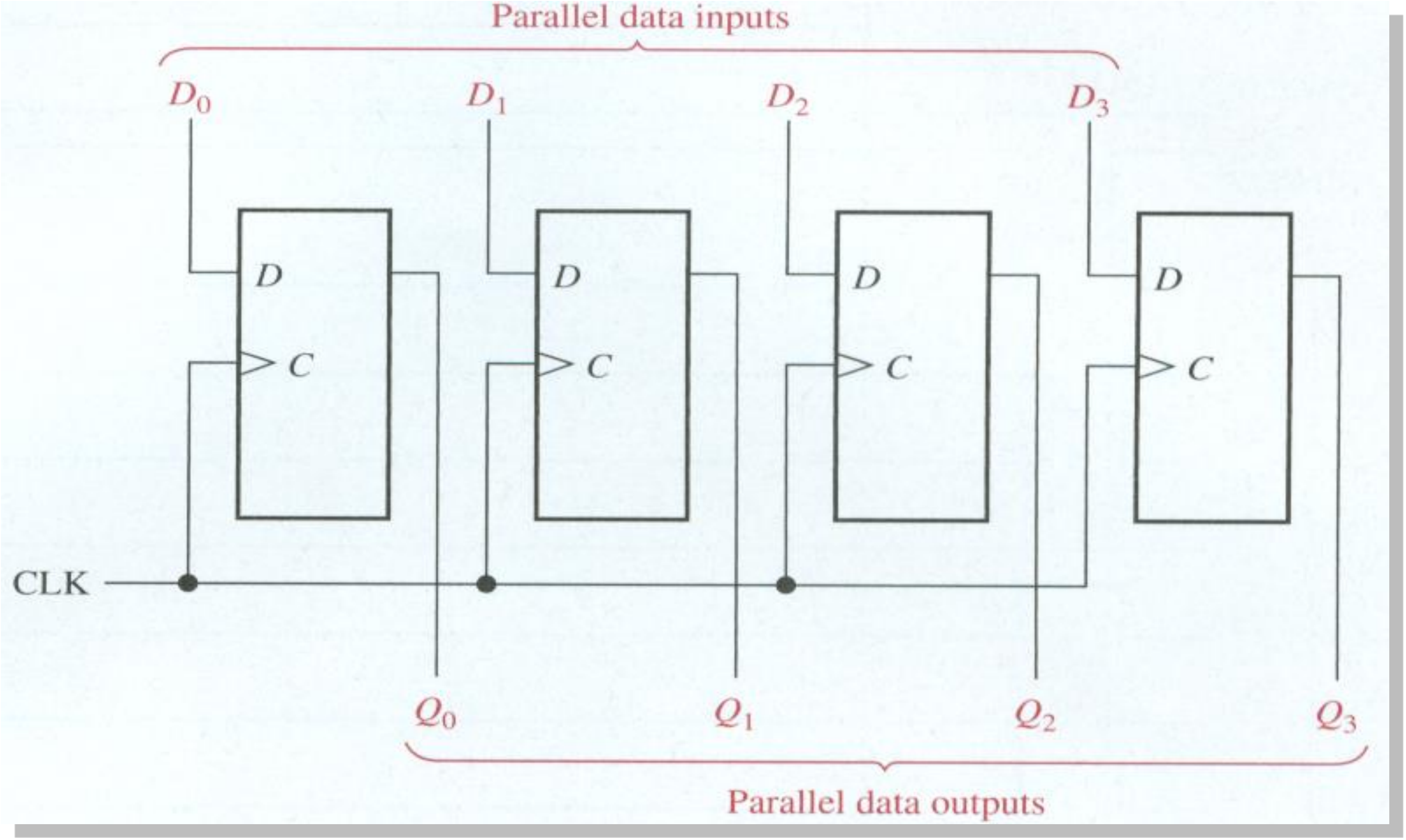
- The bits are entered simultaneously into their respective stages.
- Immediately, the bits appear on the parallel outputs.

4-bit version



Parallel In/Parallel Out Shift Registers

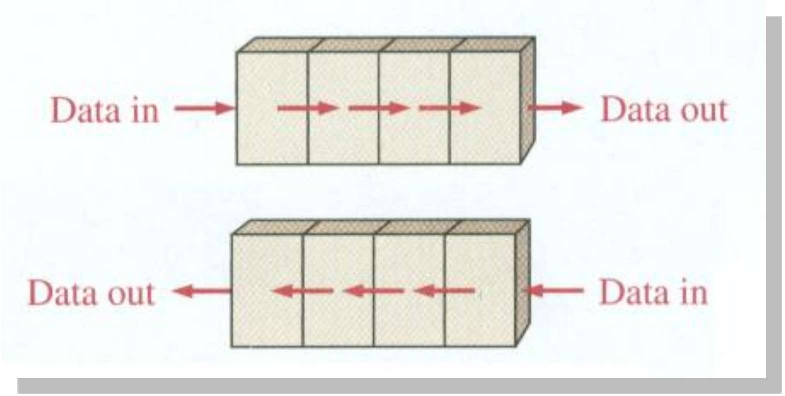
- 4-bit version



Bidirectional Shift Register

- A bidirectional shift register is one in which the data can be shifted either left or right.

4-bit version



Sequential Logic

Applications of Shift Registers:

1. Time Delays:

- **The time delay can be adjusted by controlling the number of stages in the register.**
- **By using serial-in & parallel-out registers and by taking the serial output at any one of the intermediate stages, we can delay output by any number of clock pulses.**

2. Serial/Parallel Data Conversion:

- **Data can be available in serial/parallel form.**
- **The transfer of data in parallel is much faster than serial form.**
- **Shift registers are used for converting serial data to parallel form, so that a serial input can be processed by a parallel system.**
- **A serial-in, serial-out, shift register can be used to perform serial to parallel conversion & a parallel in serial out, shift register can perform parallel-to-serial conversion.**

Sequential Logic

3. **Ring Counters:**

- **Ring counters are constructed by modifying the serial-in, serial-out, shift registers. There are two types of ring counters- basic ring counter & johnson counter.**
- **The basic ring counter can be obtained from a serial-in, serial-out, shift register by connecting Q output of the last FF to D input of first FF.**
- **The ring counter is the decimal counter. It is divide-by-N counter, where N is the number of stages. The keyboard encoder is an application of a ring counter.**

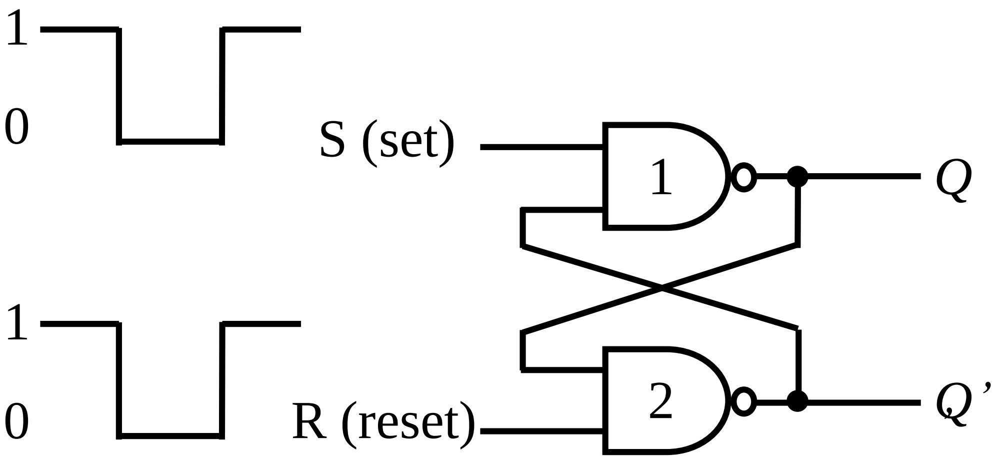
4. **UART (Universal Asynchronous Receiver Transmitter):**

- **Computers & microprocessor-based systems often send & receive data in a parallel format.**
- **They frequently communicate with external devices that send and/or receive serial data.**
- **An interfacing device used to accomplish these conversions is the UART.**

A Review

Sequential Logic

Basic RS Flip-Flop (NAND)



(a) Logic diagram

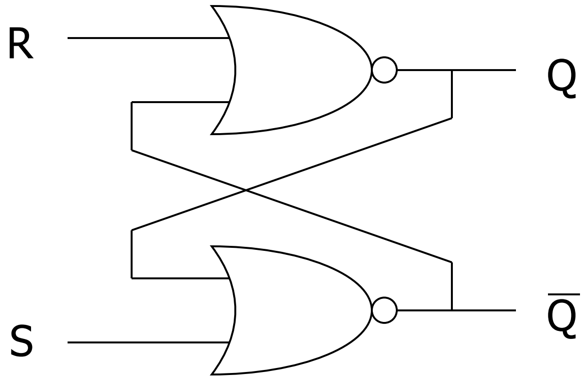
S	*R*	*Q*	*Q'*	
1	0	0	1	
1	1	0	1	(after S = 1, R = 0)
0	1	1	0	
1	1	1	0	(after S = 0, R = 1)
0	0	1	1	

(b) Truth table

A flip-flop holds 1 "bit".
"Bit" ::= "binary digit."

RS-Latch as Cross-Coupled NOR Gates

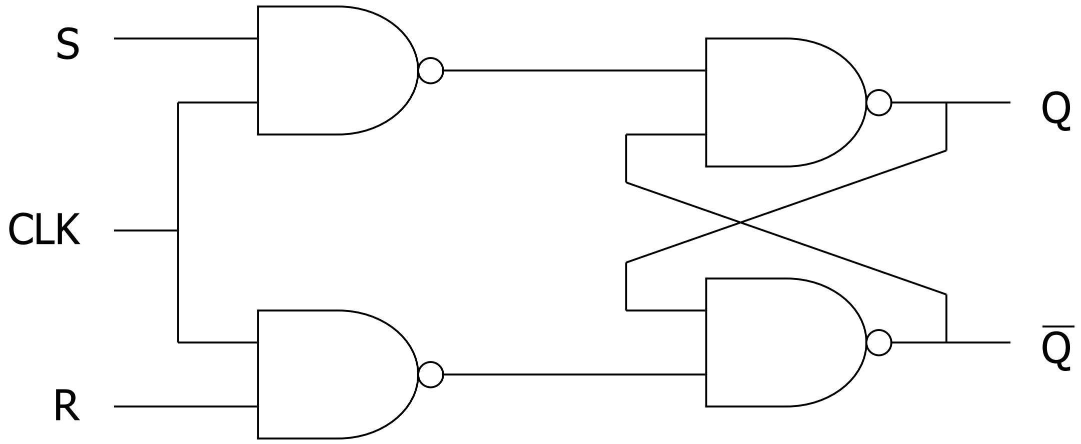
- If R = 1, Q resets to 0
- If S = 1, Q sets to 1
- If RS = 00, no change
- RS = 11 is not allowed because leads to oscillation



S R	Q
0 0	No change
0 1	0
1 0	1
1 1	Undefined

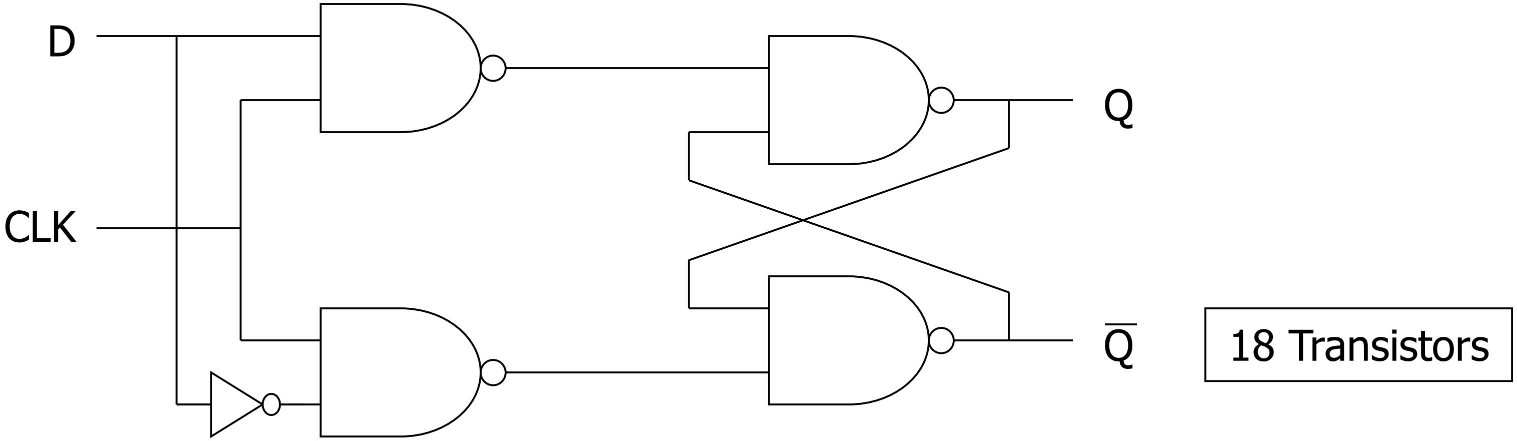
Level-Sensitive RS-Latch

- **“Q” only changes when CLK is high (i.e. level-sensitive)**
- **When CLK is high, behavior same as RS latch**



CLK	S R	Q
0	X X	No change
1	0 0	No change
1	0 1	0
1	1 0	1
1	1 1	Undefined

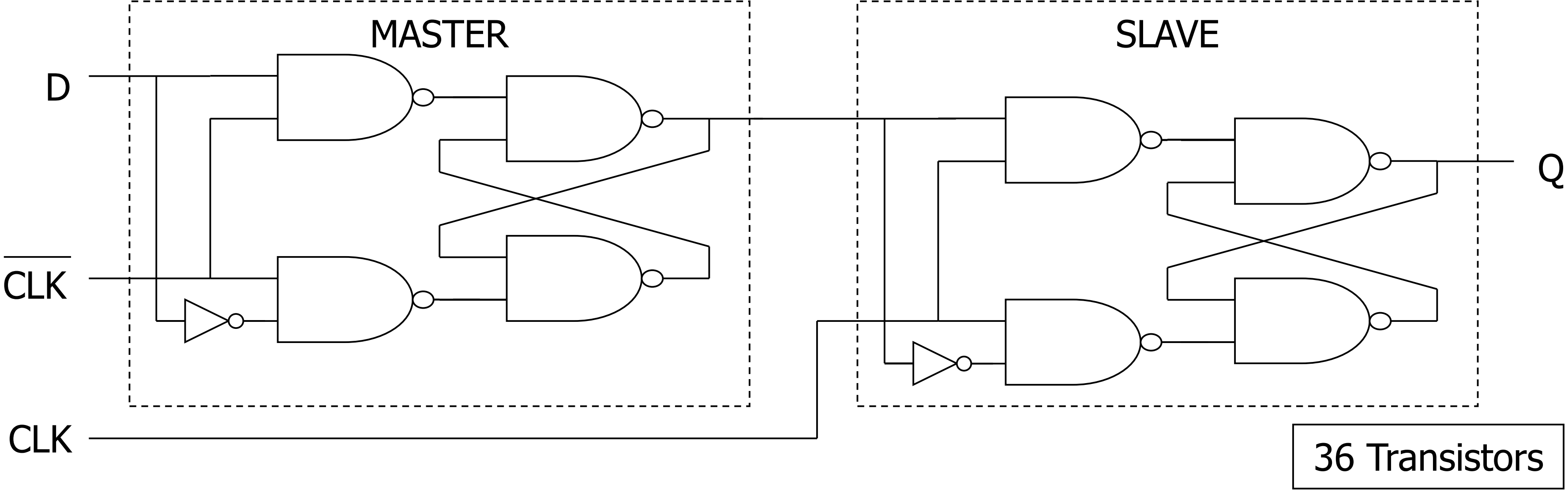
Level-Sensitive D-Latch



- Make level-sensitive D-latch from level-sensitive RS-latch by connecting S = D and R = not D

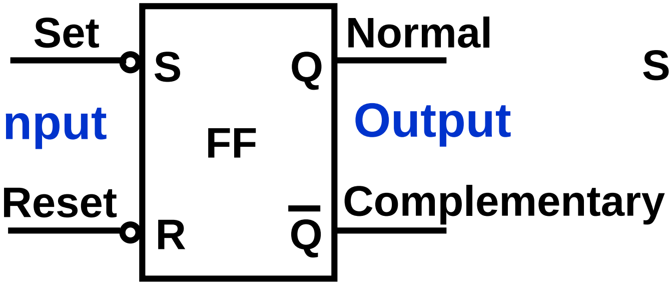
Master-Slave Edge-Triggered Flip-Flop

- Master-Slave configuration



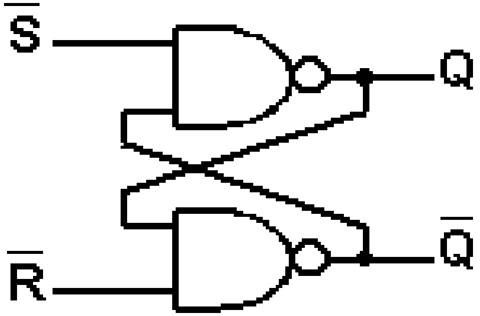
R-S Flip Flop

- Logic Symbol



Also called:
R-S Latch
Set-Reset FF

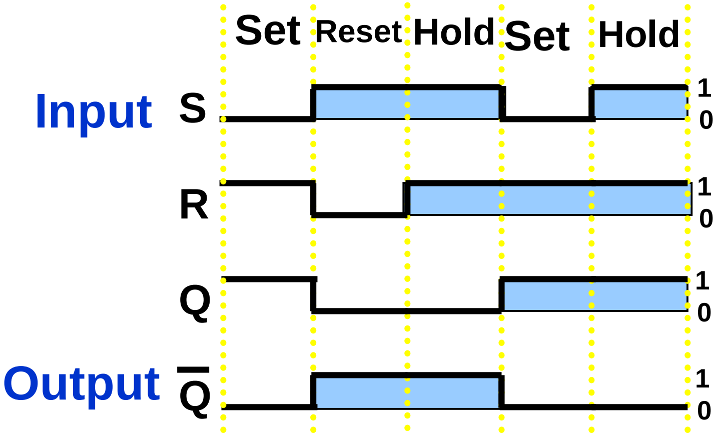
Wiring Diagram



- Truth Table

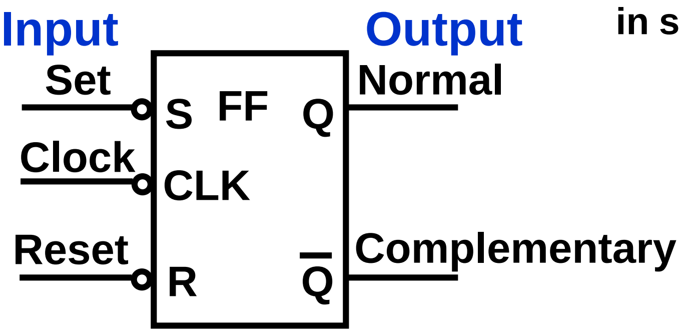
Mode of Operation	Input S	Input R	Output Q	Output Q	Effect
Prohibited	0	0	1	1	Prohibited Do not use
Set	0	1	1	0	For setting Q to 1
Reset	1	0	0	$\overline{1}$	Resetting Q to 0
Hold	1	1	Q	Q	Depends Previous State

Waveform Diagram



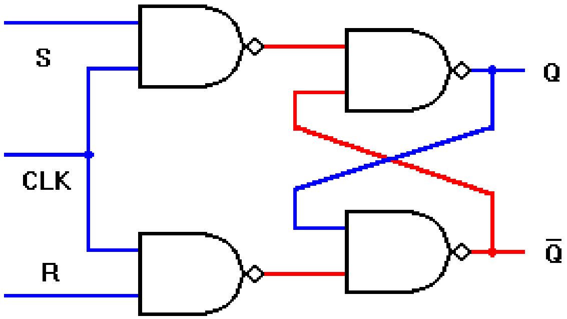
Clocked R-S Flip Flop

- Logic Symbol



Output FF operates Synchronously in step with clock.

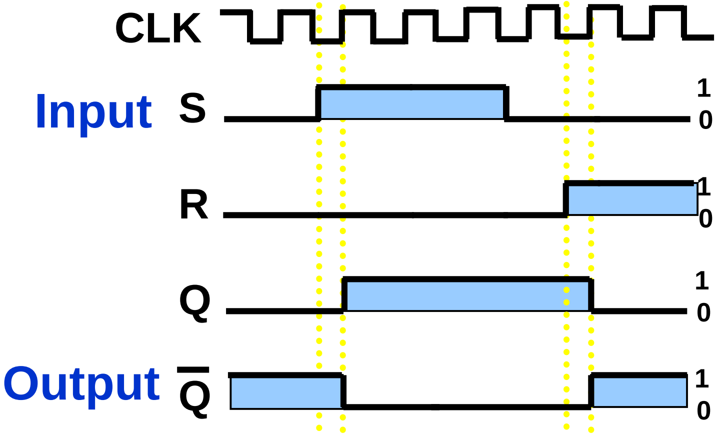
Wiring Diagram



- Truth Table

Mode of Operation	Input CLK	Input S	Input R	Output Q	Output Q	Effect
Hold	⎍	0	0	1	1	No Change
Reset	⎍	0	1	0	1	Reset or cleared to 0
Set	⎍	1	0	1	0	Set to 1
Prohibited	⎍	1	1	1	1	Do not use

Waveform Diagram



Questions

Q. What type of waveform is used in flip flops?

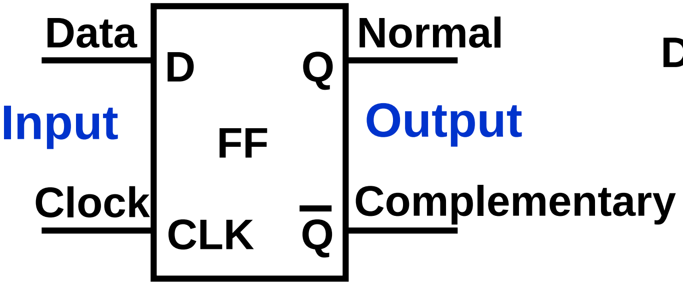
A. Square Waves.

Q. What does the RS stand for in the RS Flip Flop?

A. Reset Set.

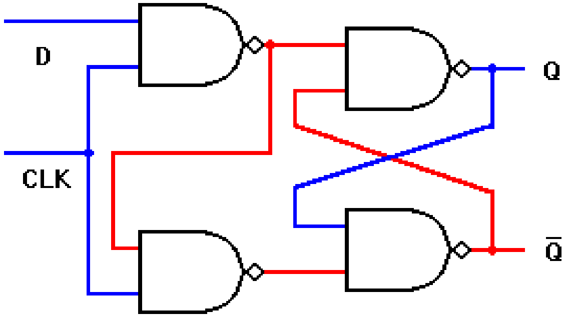
D Flip Flop

- Logic Symbol



Also called:
Delay FF
Data FF
D-type Latches
'Delayed 1 Clock Pulse'

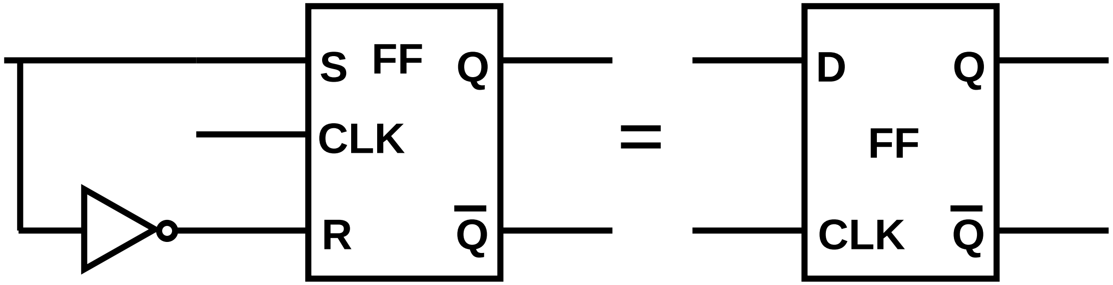
Wiring Diagram



- Truth Table

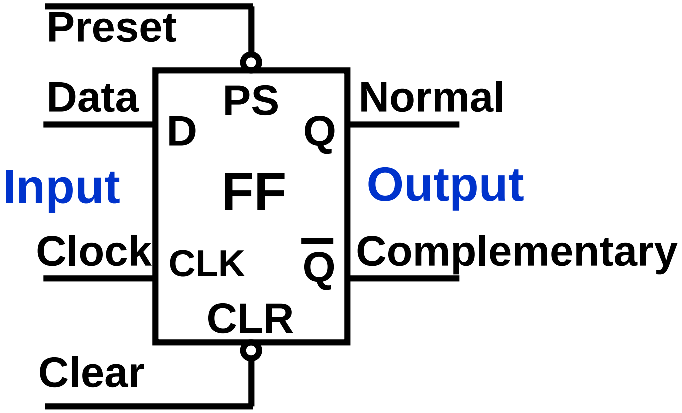
Input CLK	Input D	Output Q^{n+1}	Output $\overline{Q}$
‾|_|‾	0	0	1
‾|_|‾	1	1	0

Similar Wiring



7474 D Flip Flop

- Logic Symbol



Note: The asynchronous inputs (PS & CLR) Override the synchronous inputs (D & CLK) .

- Truth Table

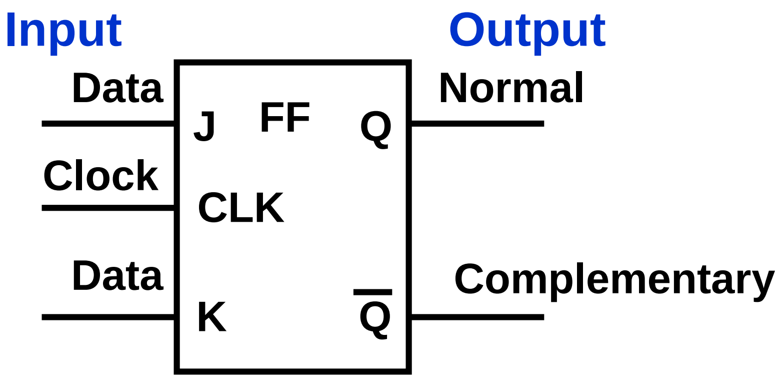
Mode of Operation	INPUTS				OUTPUTS	
	Asynchronous		Synchronous			
	PS	CLR	CLK	D	Q	$\overline{Q}$
Asynchronous Set	0	1	X	X	1	0
Asynchronous Reset	1	0	X	X	0	1
Prohibited	0	0	X	X	1	1
Set	1	1	↑ L to H	1	1	0
Reset	1	1	↑ L to H	0	0	1

D (Delay) Flip Flop Uses

- Sequential logic devices used in temporary memory devices.
- Wired together to form shift registers and storage registers.
- Delays data from reaching output Q one clock pulse.

J-K Flip Flop

- Logic Symbol

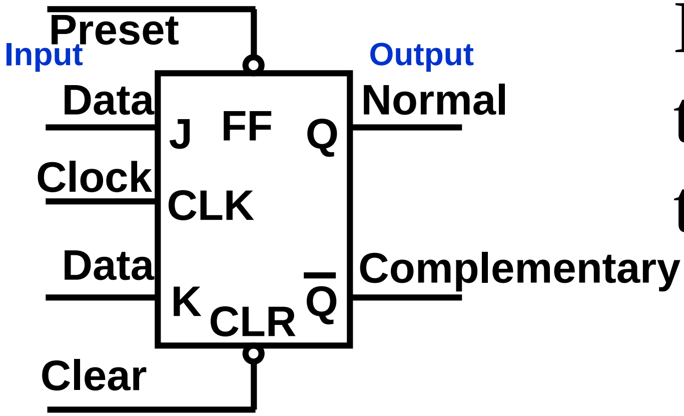


- Truth Table

Mode of Operation	INPUTS			OUTPUTS		Effect
	Input CLK	Input J	Input K	Output Q	Output Q	
Hold	⎍	0	0	No Change	No Change	No Change
Reset	⎍	0	1	0	1	Reset or cleared to 0
Set	⎍	1	0	1	0	Set to 1
Toggle	⎍	1	1	Toggle	Toggle	Changed to Opposite State

7476 J-K Flip Flop

- Logic Symbol



Note: 7476 uses the entire pulse to transfer data from J & K data inputs to Q & $\overline{Q}$ outputs.

- Truth Table

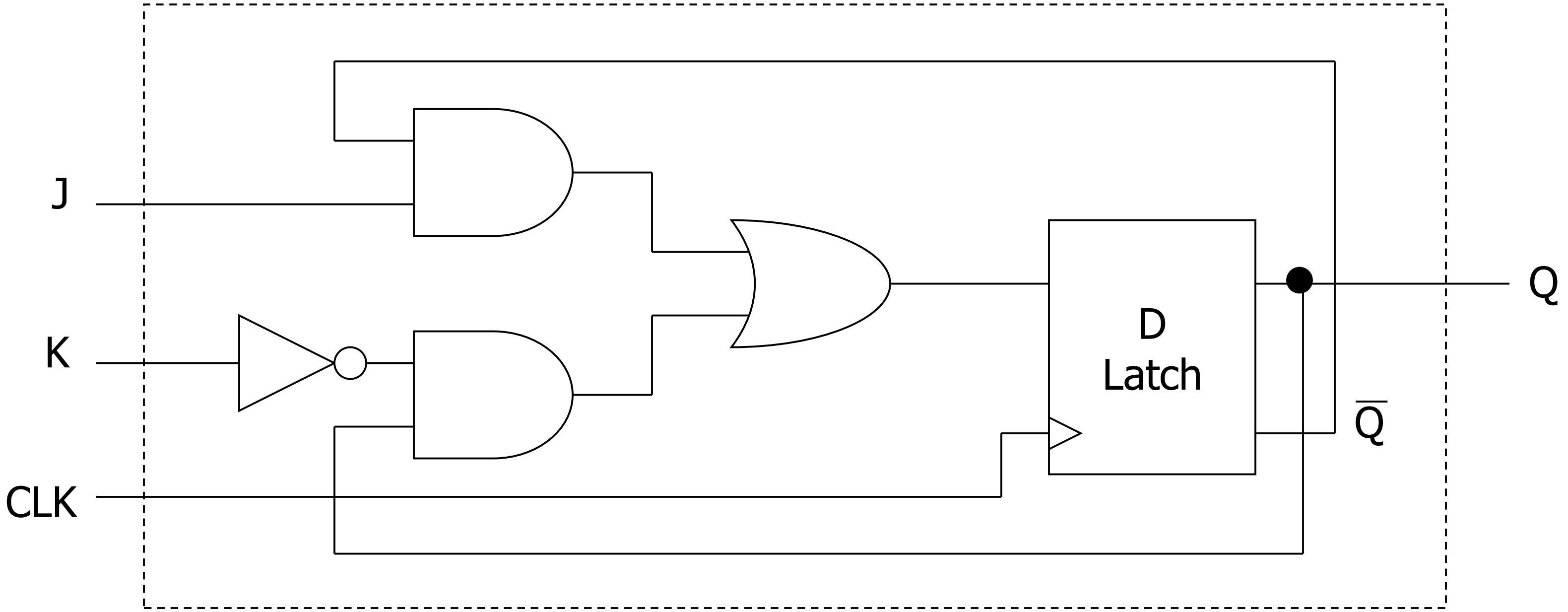
Mode of Operation	INPUTS					OUTPUTS	
	Asynchronous		Synchronous				
	PS	CLR	CLK	J	K	Q	$\overline{Q}$
Asynchronous Set	0	1	X	X	X	1	0
Asynchronous Reset	1	0	X	X	X	0	1
Prohibited	0	0	X	X	X	1	1
Hold	1	1	⎍	0	0	No Change	No Change
Reset	1	1	⎍	0	1	0	1
Set	1	1	⎍	1	0	1	0
Toggle	1	1	⎍	1	1	Opposite State	

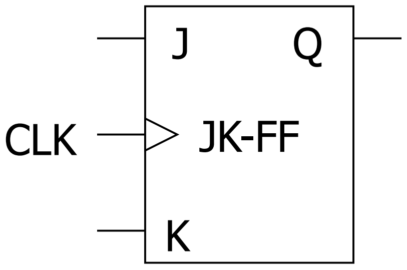
J-K Flip Flop Uses

- Universal Flip Flop. Has all features of other FF.
- When being used only in the toggle mode, commonly called a T Flip Flop.
- Most commonly used as counters.

JK Flip-Flop from D-Latch

- Same as RS-Latch except “toggle” on 11

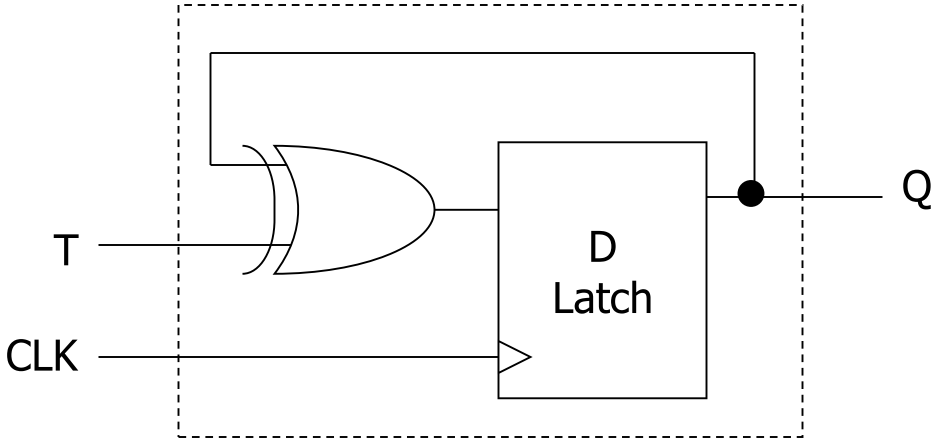


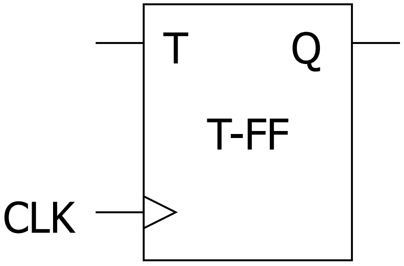


CLK	J K	Q
0	X X	No change
1	0 0	No change
1	0 1	0
1	1 0	1
1	1 1	**Toggle**

Toggle Flip-Flop from D-Latch

- Toggles stored value if T = 1 when CLK is high





CLK	T	Q
0	X	No change
1	0	No change
1	1	**Toggle**

End
of
Unit-3