Unit-4
Analysis of Sequential Circuits

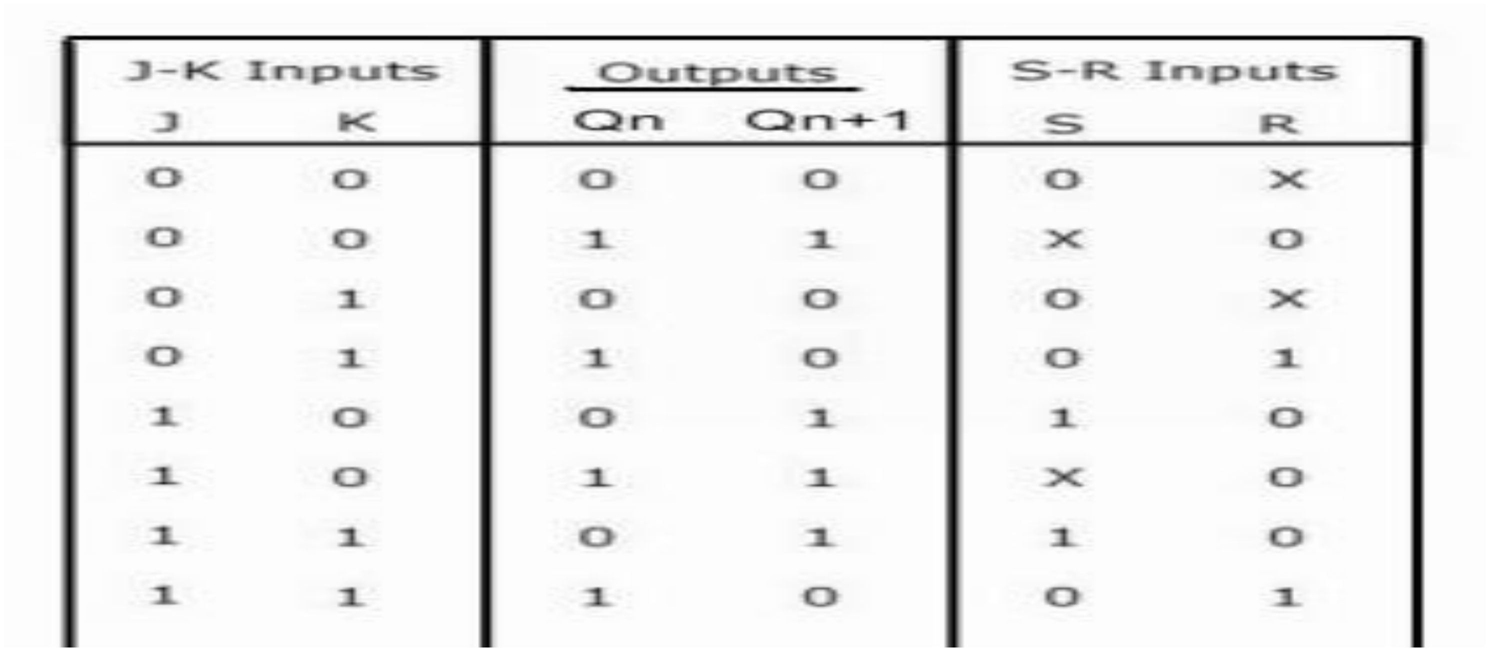
- **Analysis of Sequential Circuits: Conversion of flip flops: A synthesis example, Types of Shift Register, SISO, SIPO, PISO and PIPO, Applications of Shift Registers as Ring Counter, Johnson Counter, Serial Adder.**
- **Counters: Asynchronous counters (4 bit), Synchronous Counters (4 bit), Changing the counter Modulus, Decade counter (using IC 7490).**
- **Tutorial: Application of IC 7490, Design and implementation of MOD-N counter**

.

Conversion of Flip Flops

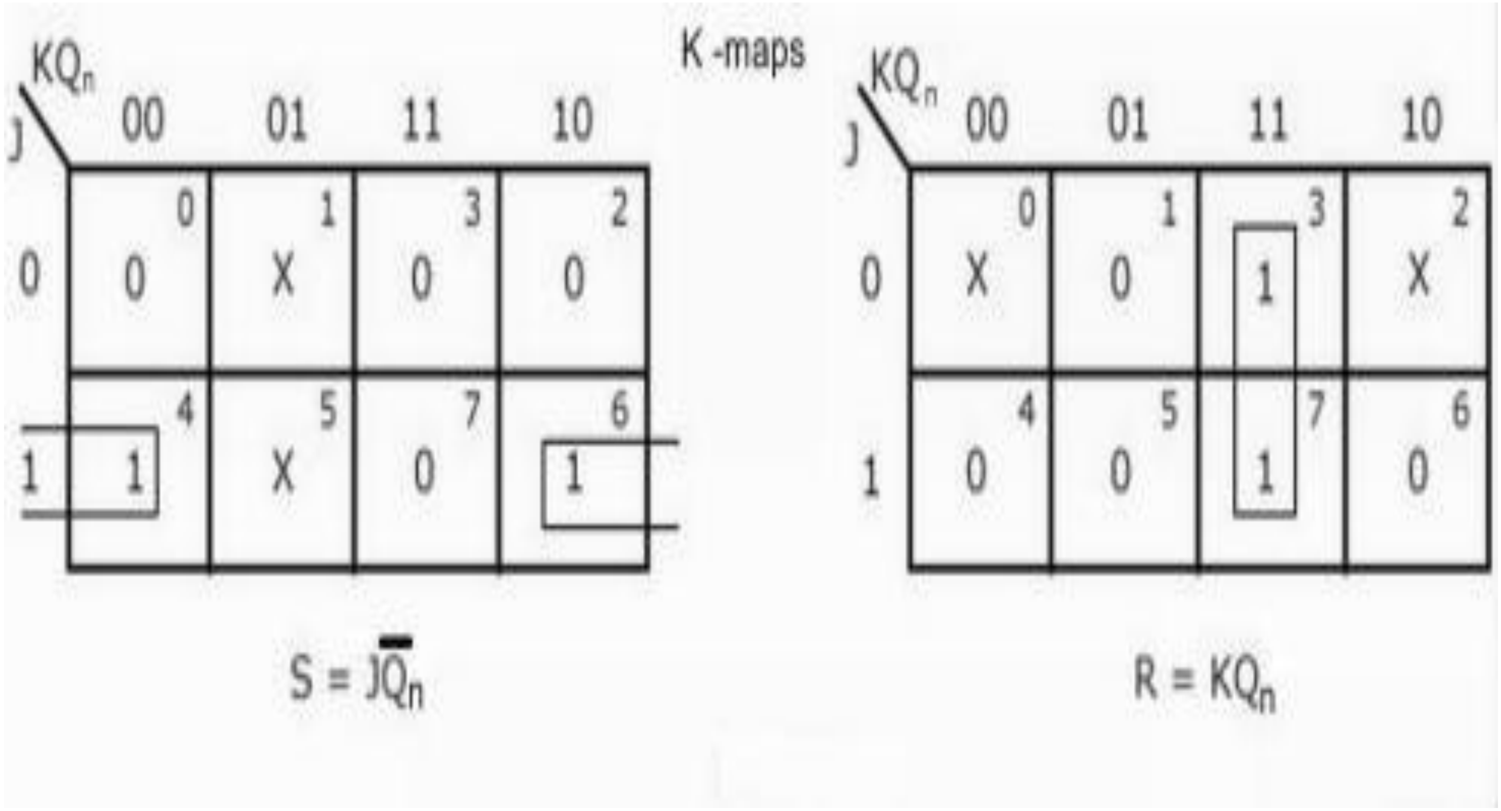
Conversion of SR Flip Flop to JK

- **For the flip flop to be constructed(JK), the next state is found with present inputs and present state of the flip flop. The excitation table of SR ff is written based on Qn and Qn+1.**



J-K Inputs		Outputs		S-R Inputs	
J	K	Qn	Qn+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

Expressions for S and R



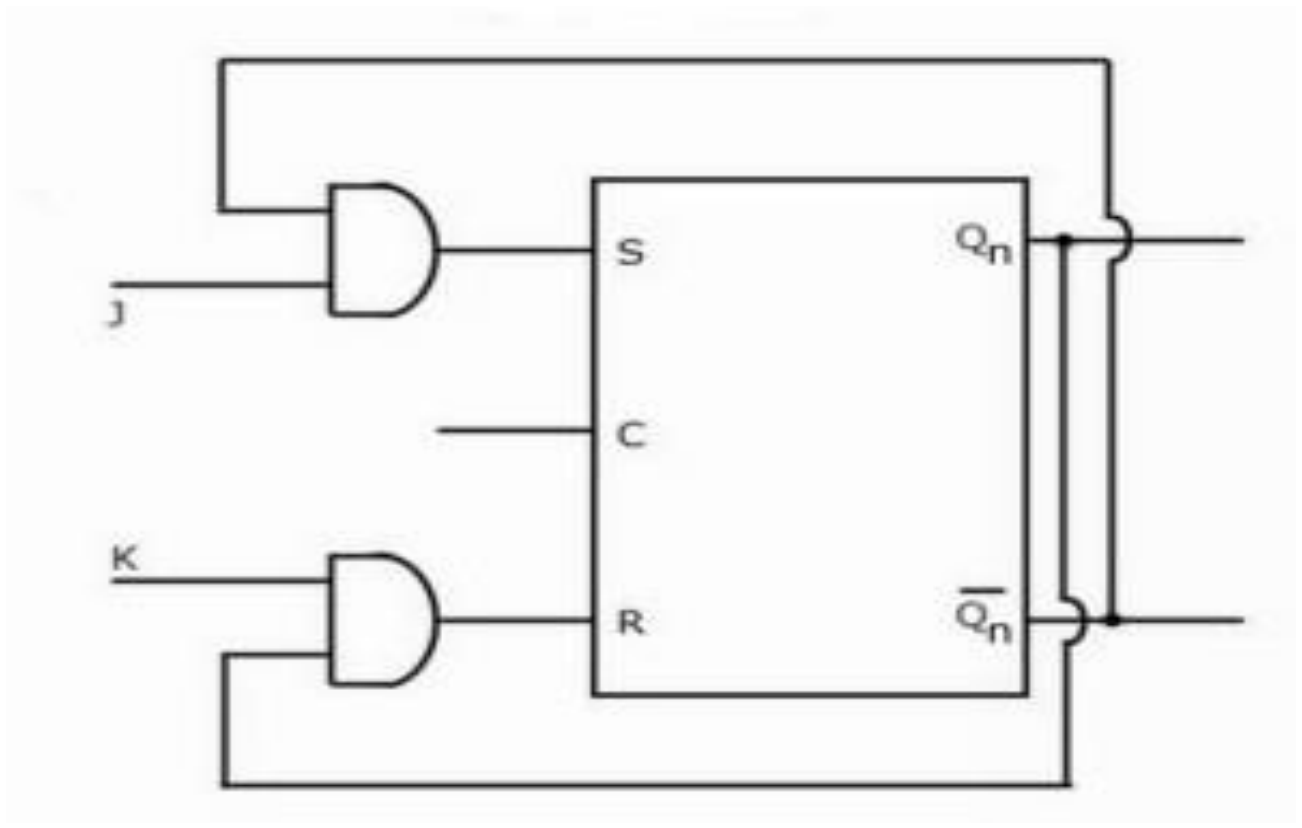
$J \backslash KQ_n$	00	01	11	10
0	0	X	0	0
1	1	X	0	1

$S = J\overline{Q}_n$

$J \backslash KQ_n$	00	01	11	10
0	X	0	1	X
1	0	0	1	0

$R = KQ_n$

Logic Diagram

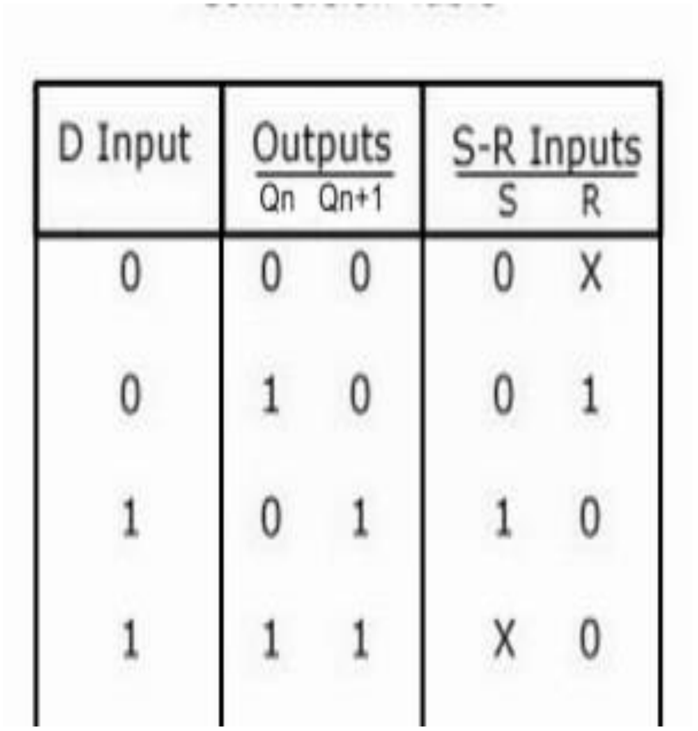


Continued…

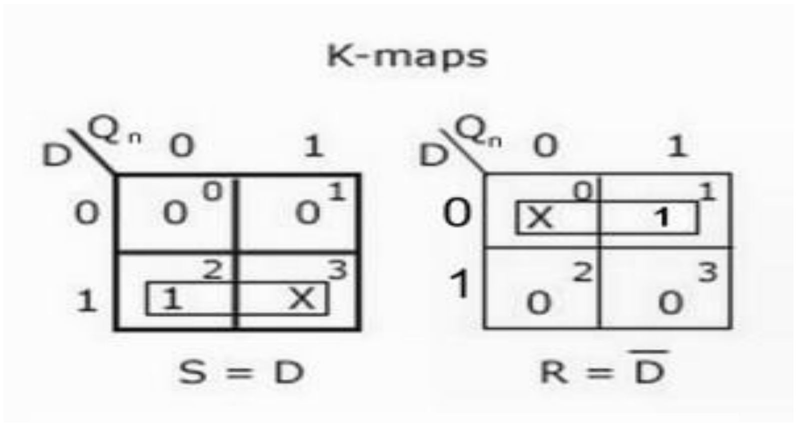
- **In this case we are required to convert SR flip flop into JK flip flop. Therefore we have to first design and connect the combinational circuit to the input of SR flip flop so that it will produce same outputs as that of JK flip flop. Here the external inputs are J and K. S and R will be the outputs of designed combinational circuit which are inputs of actual flip flop. We write a truth table with J, K, Q_n, Q_{n+1}, S and R. where Q_n is the present state of the flip flop and Q_{n+1} will be the next state obtained when the particular J and K inputs are applied.**
- **J, K and Q_n can have eight combinations. For each combination of J, K and Q_n find the corresponding Q_{n+1}, i.e. determine to which next state the JK flip flop will go from the present state Q_n if the present inputs J and K are applied. Now complete the table by writing the values of S and R required to get each Q_{n+1} from the corresponding Q_n. i.e. write what values of S and R are required to change the state of the flip flop from Q_n to Q_{n+1}.**

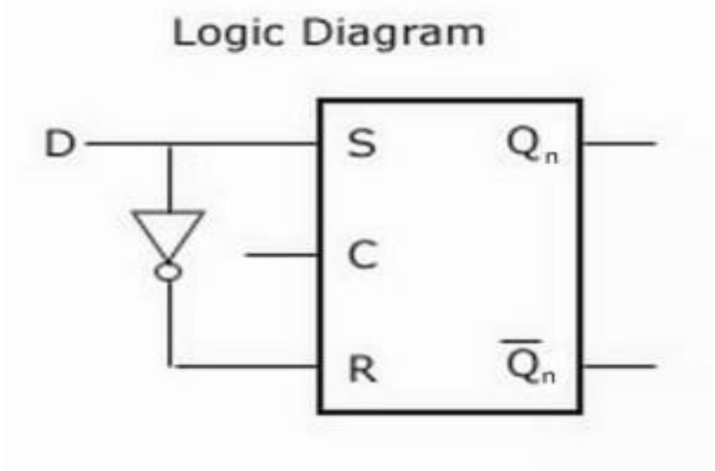
SR flip flop to D flip flop

- **Conversion Table:**



D Input	Outputs		S-R Inputs	
	Qn	Qn+1	S	R
0	0	0	0	X
0	1	0	0	1
1	0	1	1	0
1	1	1	X	0

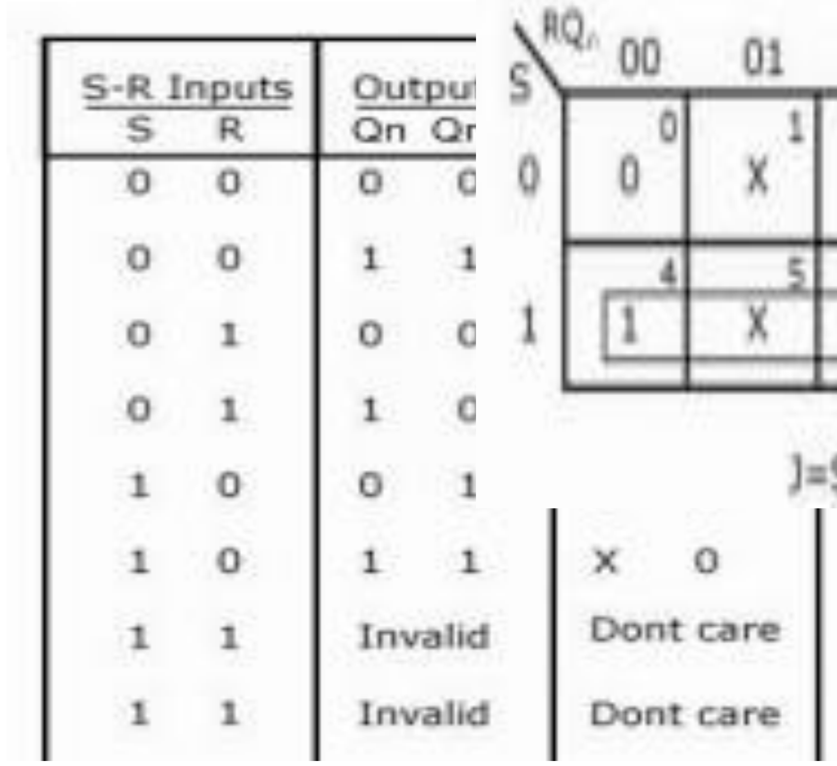




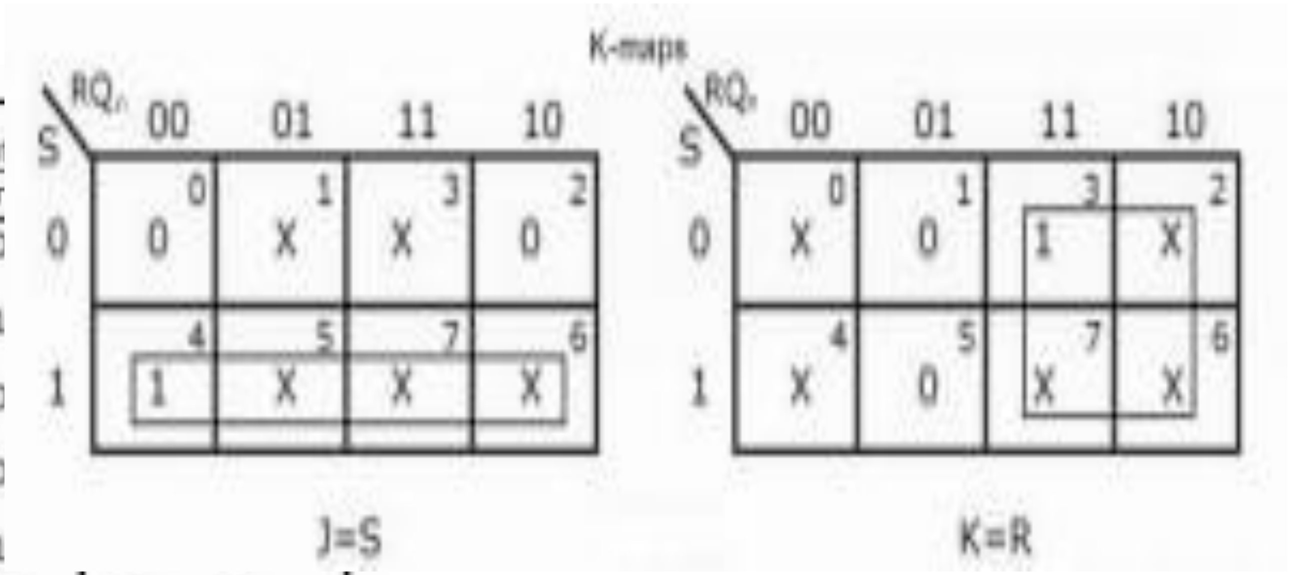
Conversion of JK flip flop to SR flip flop

- In case of converting JK flip flop into SR flip flop, external inputs (inputs of combinational circuit) are S and R, while J and K are the inputs of actual flip flop.
- So we have to get values of J and K in terms of S, R and Q_n. thus we prepare a conversion table S, R, Q_n, Q_{n+1}, J and K.
- The external inputs S and R and the output Q_n can make 8 combinations. For each combination find the corresponding Q_{n+1}.
- In the SR flip flop, the combination S=1 and R=1 is not permitted. So, the corresponding output is invalid and, therefore the corresponding J and K are don’t cares.

Conversion Table



S-R Inputs		Outpu			
S	R	Qn	Qr		
0	0	0	0		
0	0	1	1		
0	1	0	0		
0	1	1	0		
1	0	0	1		
1	0	1	1	X	0
1	1	Invalid		Dont care	
1	1	Invalid		Dont care	



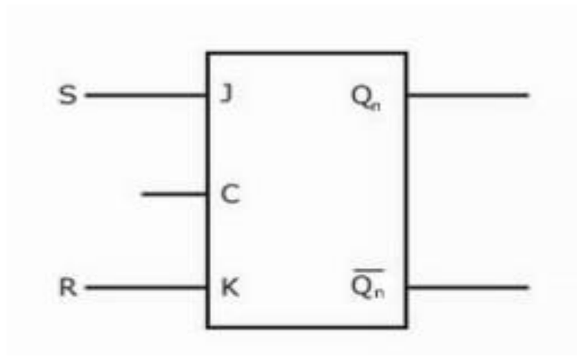
K-maps

S \ RQn	00	01	11	10
0	0	X	X	0
1	1	X	X	X

J=S

S \ RQn	00	01	11	10
0	X	0	1	X
1	X	0	X	X

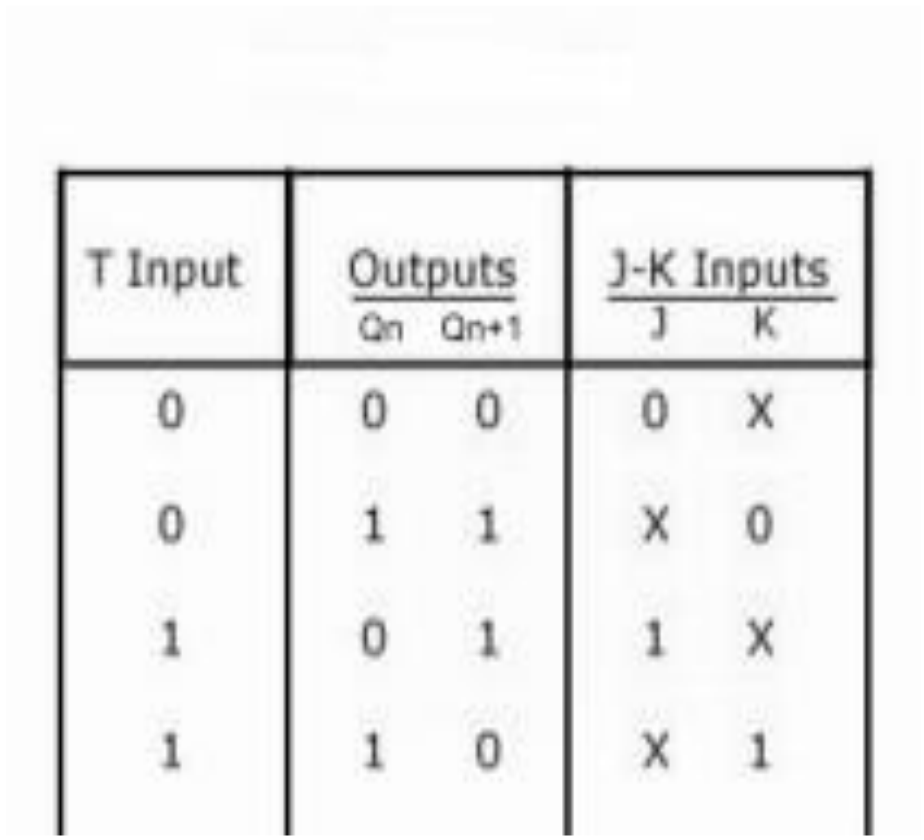
K=R



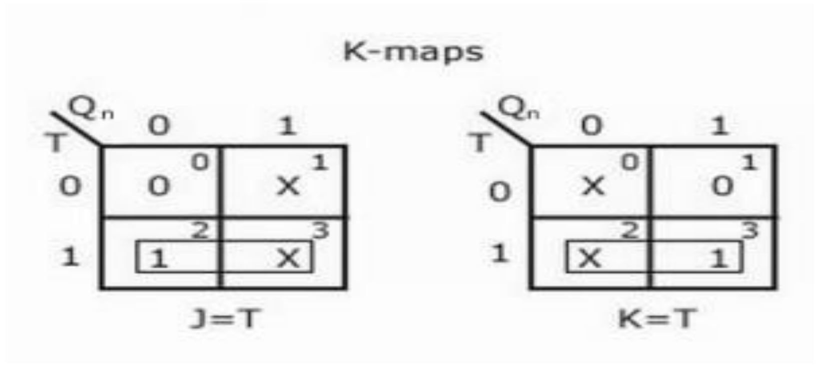
Conversion of JK flip flop to T flip flop

- For the conversion of JK flip flop to T type of flip flop, T will be the external input (input of combinational circuit) and the output of this combinational circuit is connected to the inputs of actual flip flop (J and K).
- Then we prepare conversion table and using this table express J and K in terms of T and Q_n.
- The conversion table, K-Maps and logic diagram for the conversion of JK flip flop to T type of flip flop is shown below:

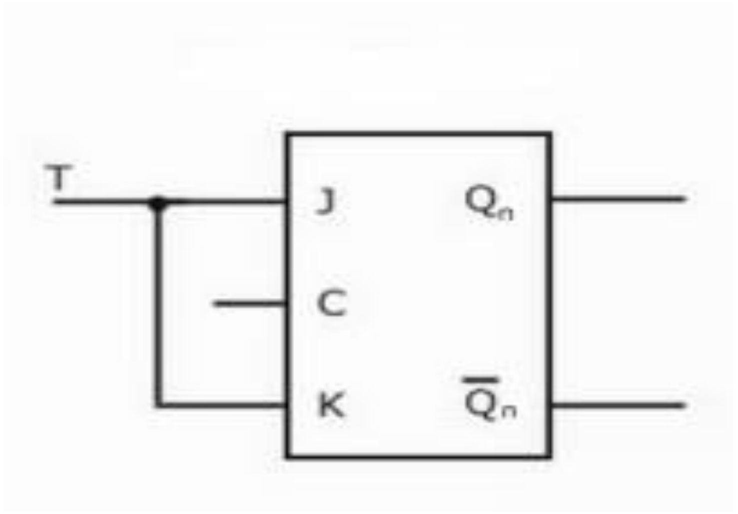
Conversion Table



T Input	Outputs		J-K Inputs	
	Qn	Qn+1	J	K
0	0	0	0	X
0	1	1	X	0
1	0	1	1	X
1	1	0	X	1

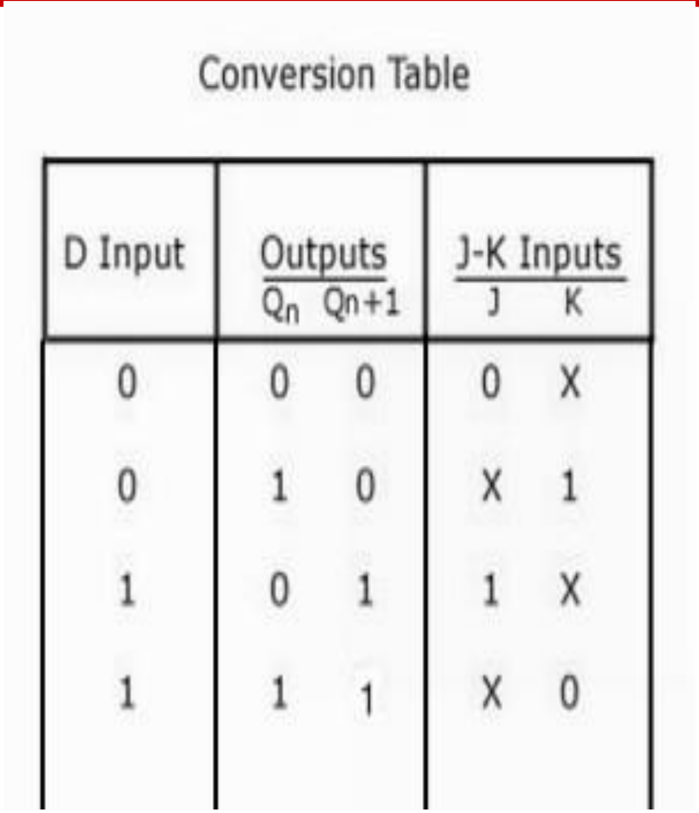


Logic Diagram



Conversion of JK flip flop to D flip flop:

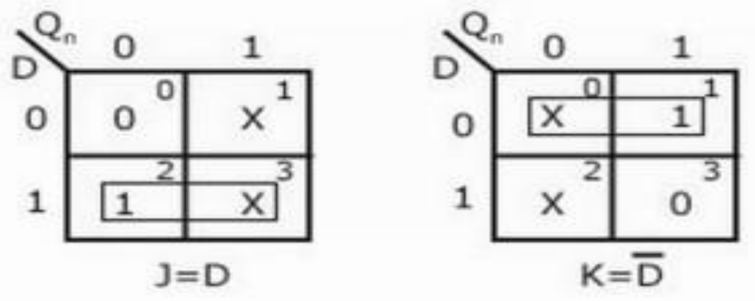
- **In case of converting JK flip flop into D flip flop, D is the external input of combinational circuit, whereas J and K are the inputs of actual flip flop.**
- **D and Q_n make four combinations. So, prepare a conversion table and using this table express J and K in terms of D and Q_n.**
- **The conversion table, K-map and logic diagram for the conversion of JK flip flop to D flip flop is shown below:**



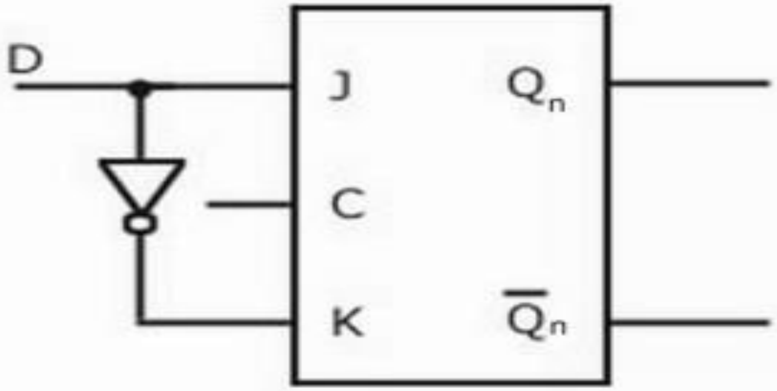
Conversion Table

D Input	Outputs		J-K Inputs	
	Q_n	Q_{n+1}	J	K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	X
1	1	1	X	0

K-maps



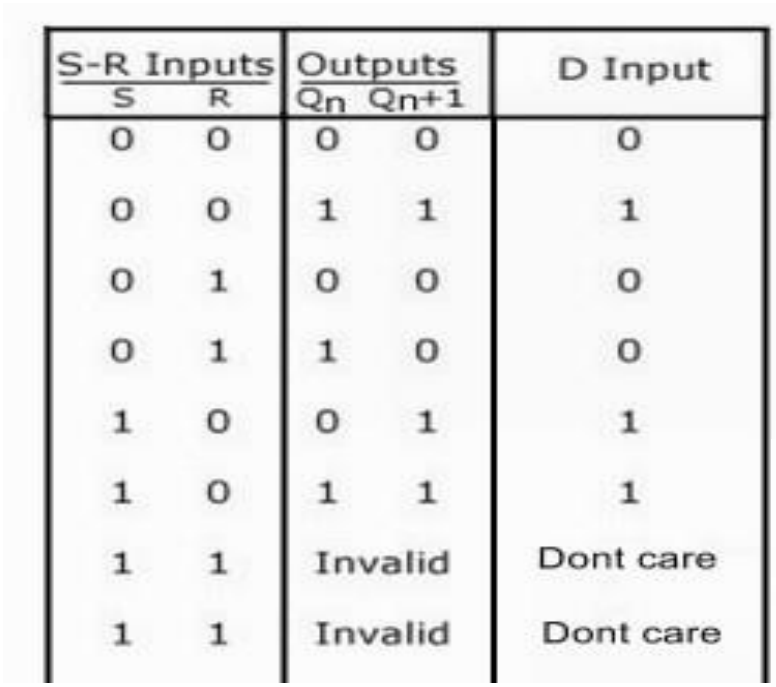
Logic Diagram



Conversion of D flip flop into SR flip flop

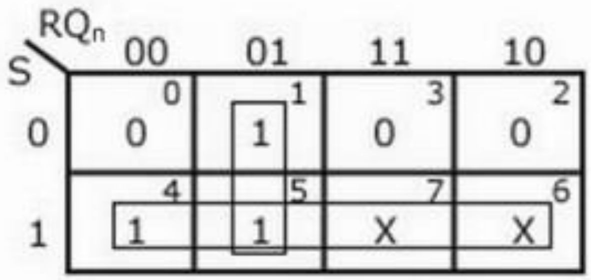
- **For converting D flip flop to SR flip flop, we use S and R as external inputs and D is the actual input to the flip flop. S, R and Qn makes eight possible combinations, but S=R=1 is an invalid combination. So, the corresponding entries for Qn+1 and D are don't cares. Then we have to express D in terms of S, R and Qn for the design of required flip flop.**
- **The conversion table, K-Maps and logic diagram for the conversion of D flip flop into SR flip flop is shown below:**

Conversion able



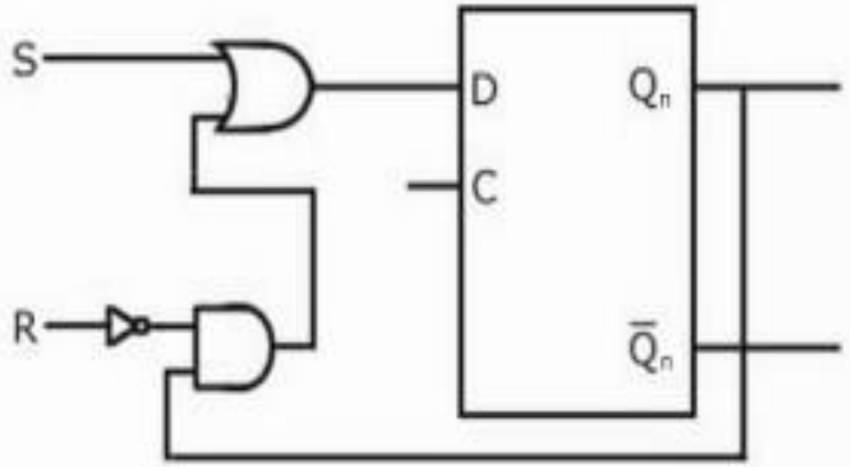
S-R Inputs		Outputs		D Input
S	R	Qn	Qn+1	
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	Invalid		Dont care
1	1	Invalid		Dont care

K-map



S \ RQn	00	01	11	10
0	0 (0)	1 (1)	0 (3)	0 (2)
1	1 (4)	1 (5)	X (7)	X (6)

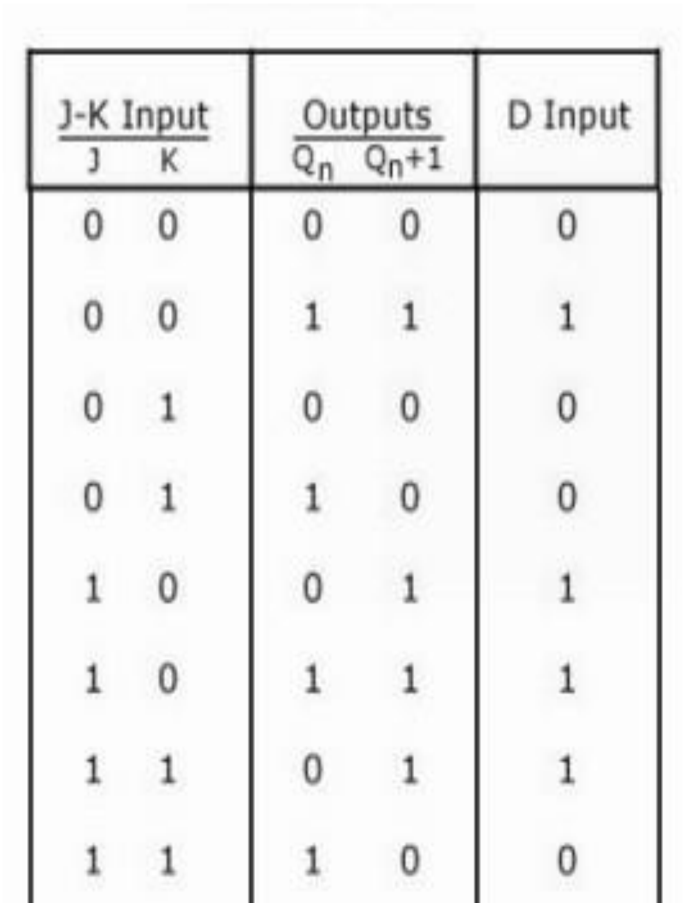
$D = S + \overline{R}Q_n$



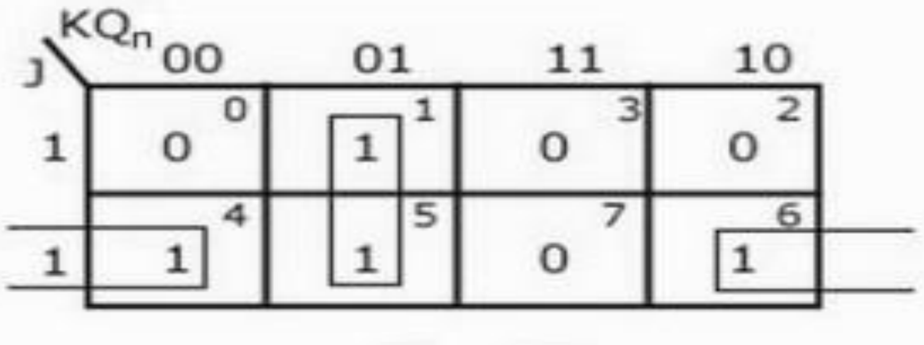
Conversion of D flip flop into JK flip flop:

- **In case of conversion of D flip flop to JK flip flop we have to use J and K as the external inputs and D as the input of actual flip flop. J,K and Qn makes eight possible combinations. Express D in terms of J, K and Qn.**
- **The conversion table, K-Maps and logic diagram for the conversion of D flip flop into JK flip flop is shown below**

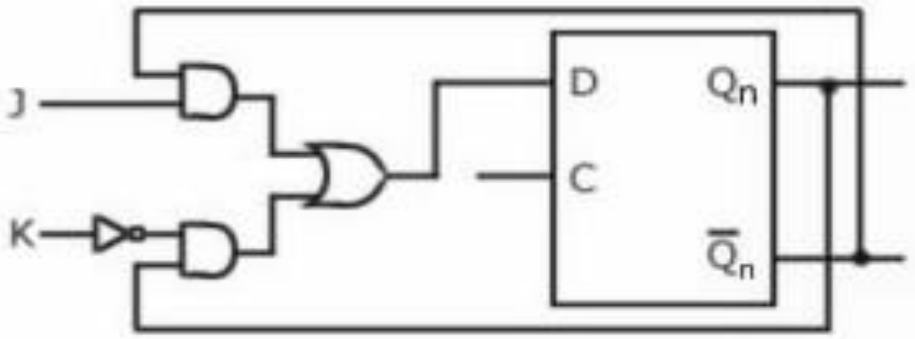
Conversion Table



J-K Input		Outputs		D Input
J	K	Q_n	Q_n+1	
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0



$$D = J\overline{Q}_n + \overline{K}Q_n$$



Conversion of D Flip Flop to T Flip Flop

1. Truth Table for T Flip Flop

Input	Outputs	
T	Q_n	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

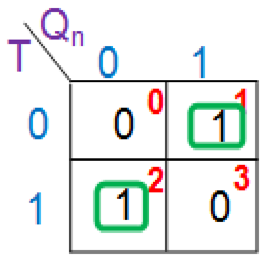
2. Excitation Table for D Flip Flop

Outputs		Input
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

3. Conversion Table

T	Q_n	Q_{n+1}	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

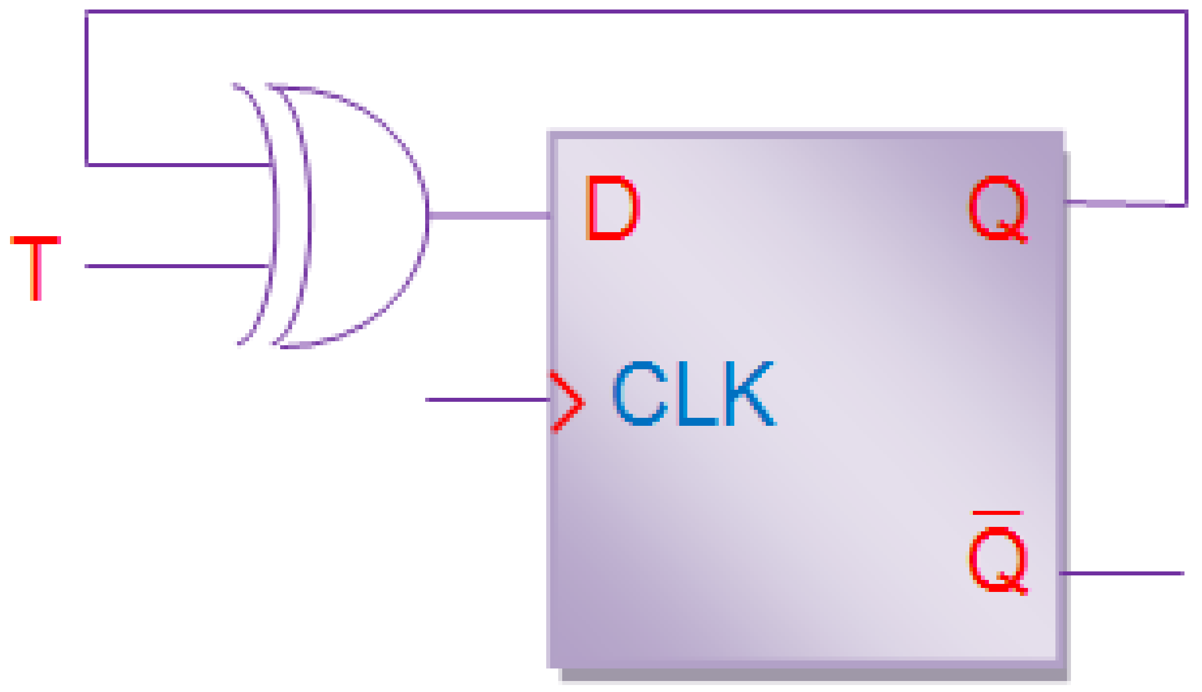
4. K-map Simplification



$$D = T\bar{Q}_n + \bar{T}Q_n$$

$$= T \oplus Q_n$$

5. Circuit Design



Conversion of T Flip Flop to SR Flip Flop

1. Truth Table for SR Flip Flop

S	R	Q_n	Q_{n+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	invalid	
1	1	invalid	

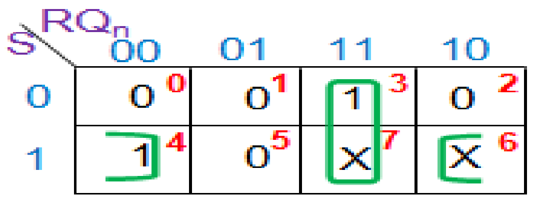
2. Excitation Table for T Flip Flop

Outputs		Input
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

3. Conversion Table

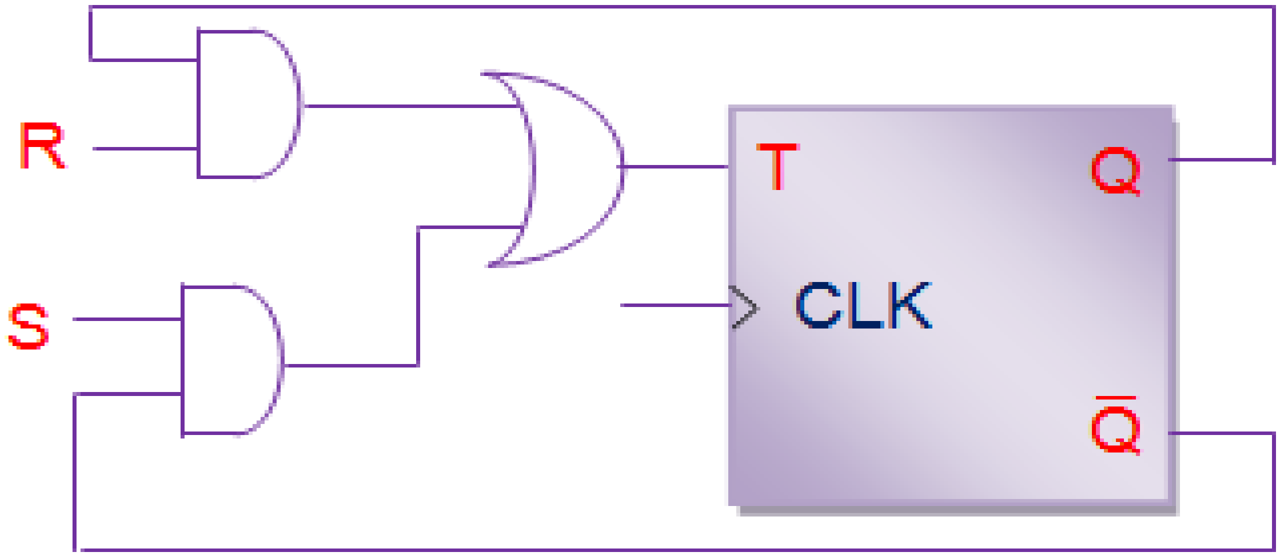
S	R	Q_n	Q_{n+1}	T
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	0
1	1	invalid		X
1	1	invalid		X

4. K-map Simplification



$$T = S\bar{Q}_n + RQ_n$$

5. Circuit Design



Conversion of T Flip Flop to JK Flip Flop

1. Truth Table for JK Flip Flop

Inputs		Outputs	
J	K	Q_n	Q_{n+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

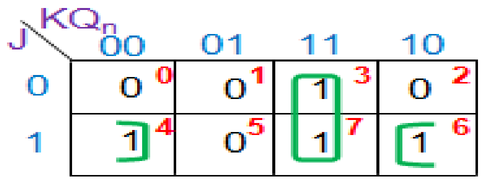
2. Excitation Table for T Flip Flop

Outputs		Input
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

3. Conversion Table

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

4. K-map Simplification



J \ KQ_n	00	01	11	10
0	0 [0]	0 [1]	1 [3]	0 [2]
1	1 [4]	0 [5]	1 [7]	1 [6]

$$T = J\overline{Q}_n + KQ_n$$

Conversion of T Flip Flop to D Flip Flop

3. Conversion Table

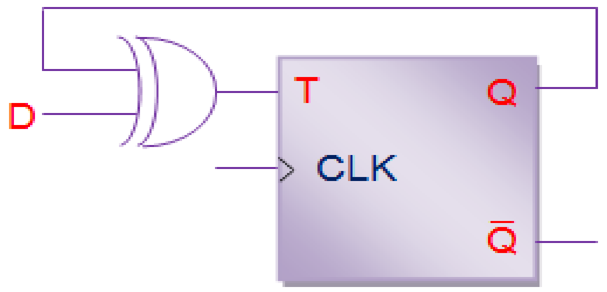
D	Q_n	Q_{n+1}	T
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

4. K-map Simplification

$$\begin{array}{c|cc} D \backslash Q_n & 0 & 1 \\ \hline 0 & 0^{0} & \boxed{1}^{1} \\ 1 & \boxed{1}^{2} & 0^{3} \end{array}$$

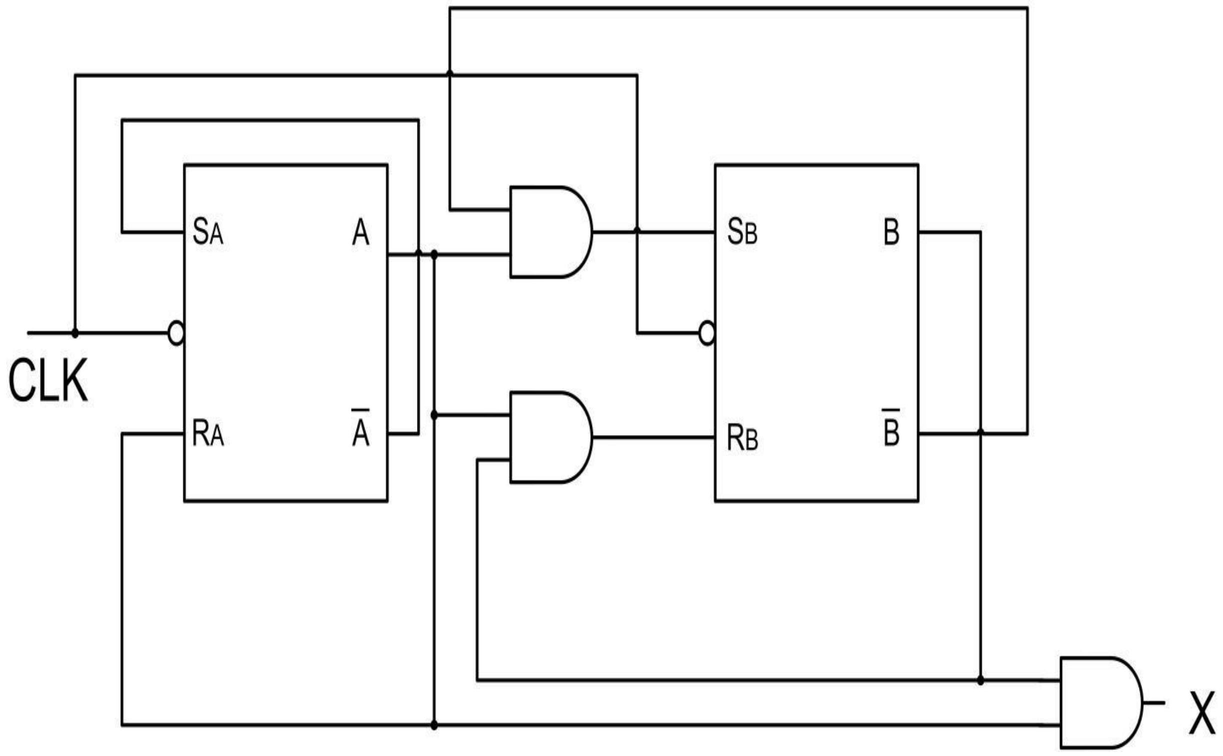
$$D = D\bar{Q}_n + \bar{D}Q_n$$
$$= D \oplus Q_n$$

5. Circuit Design



Analysis of Sequential Circuits

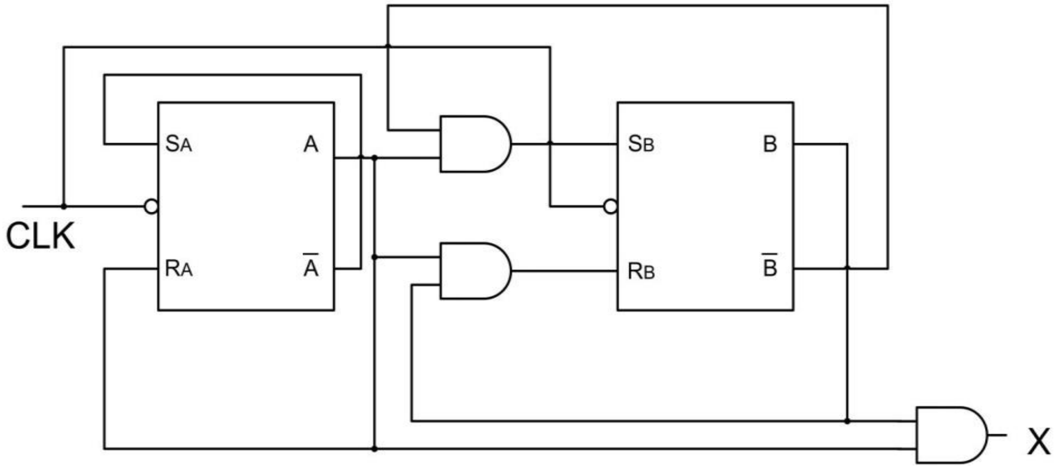
Analyze the given sequential circuit :



Continued…

- Solution: From the circuit diagram, we have the flip-flop input relations:
- Using characteristic equation of SR flip-flop: $Q_{n+1} = S + R'Q_n$, we can write,

  for flip flop A,
- $A_{n+1} = S_A + R_A'A_n$
- $A_n' + A_n' A_n$
- A_n'



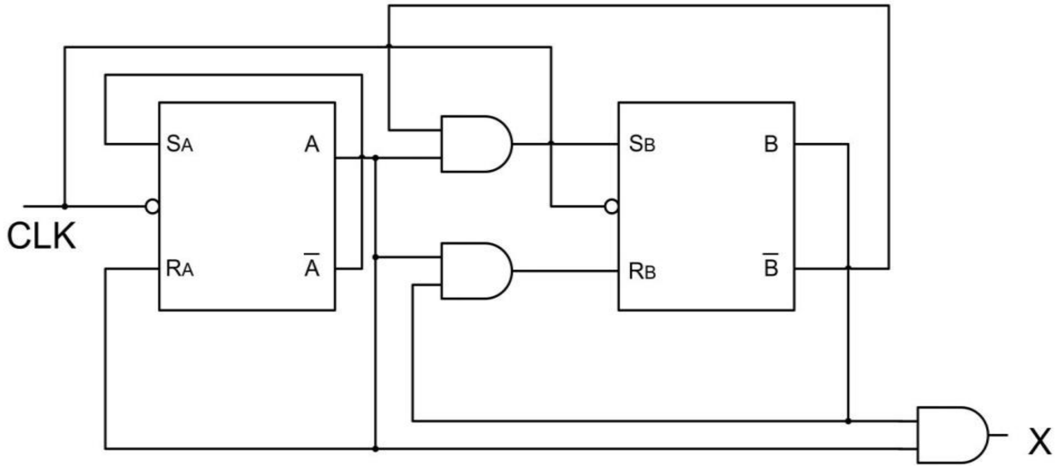
Continued…

- Solution: From the circuit diagram, we have the flip-flop input relations:
- Using characteristic equation of SR flip-flop: $Q_{n+1} = S + R'Q_n$, we can write,

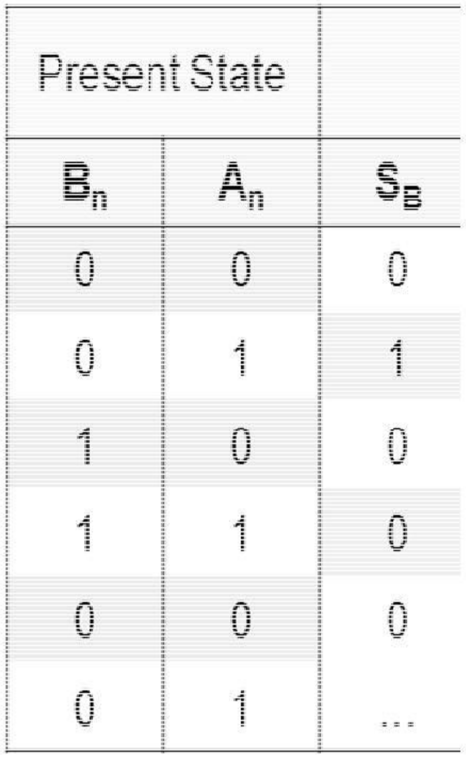
  for flip flop A,
- $A_{n+1} = S_A + R_A'A_n$

- $A_n' + A_n' A_n$

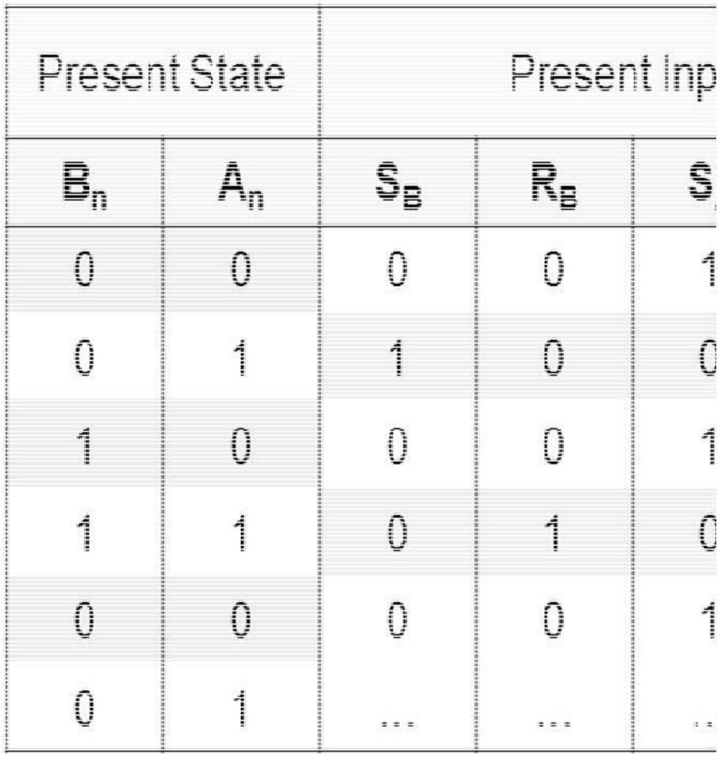
- A_n'



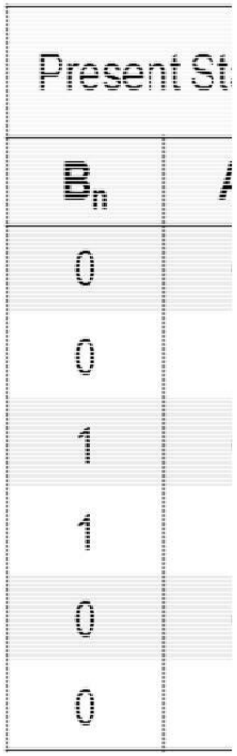
State Analysis Table



Present State		
B_n	A_n	S_B
0	0	0
0	1	1
1	0	0
1	1	0
0	0	0
0	1	...

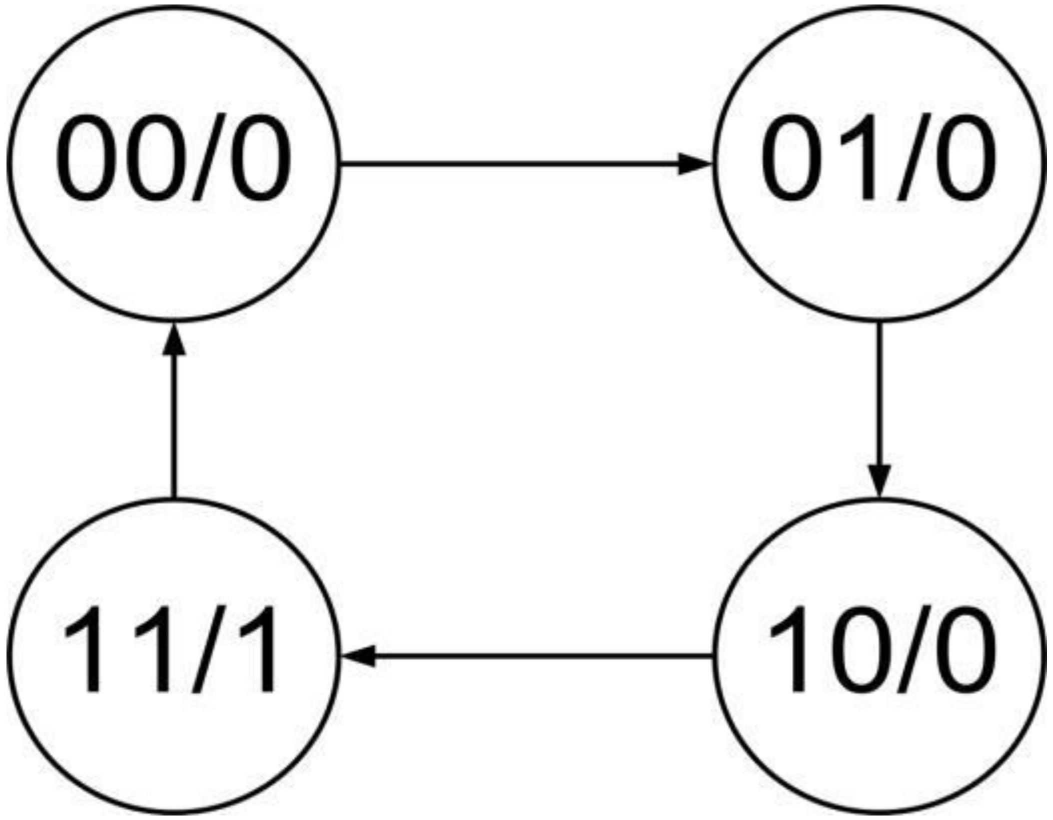


Present State		Present Inp		
B_n	A_n	S_B	R_B	S
0	0	0	0	1
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0
0	0	0	0	1
0	1	...	...	..



Present St
B_n
0
0
1
1
0
0

State Transition Diagram


$$\begin{tikzcd}
00/0 \arrow[r] & 01/0 \arrow[d] \\
11/1 \arrow[u] & 10/0 \arrow[l]
\end{tikzcd}$$

In order to convert a given T flip-flop into SR-type, we need to combine the information presented in the SR flip-flop's truth table and the information in the T flip-flop's excitation table into a common table. This can be referred to as a T-to-SR conversion table and is as shown in Figure 1.

- Notice the don't care (X) entries in the last two rows of the conversion table's "T input" column. These indicate that when both inputs (S and R) are driven high, the output of the SR flip-flop is unpredictable (owing to the "race around condition").
- Next, we should express the input of the given flip-flop in terms of the present-state, $Q_{n,}$ and the input(s) of the desired flip-flop. This can be done by using a suitable simplification technique, such as the K-map

The Shift Register

- The Shift Register is another type of sequential logic circuit that can be used for the storage or the transfer of data in the form of binary numbers.
- This sequential device loads the data present on its inputs and then moves or “shifts” it to its output once every clock cycle, hence the name Shift Register.
- A shift register basically consists of several single bit “D-Type Data Latches”, one for each data bit, either a logic “0” or a “1”, connected together in a serial type daisy-chain arrangement so that the output from one data latch becomes the input of the next latch and so on.

Continued…

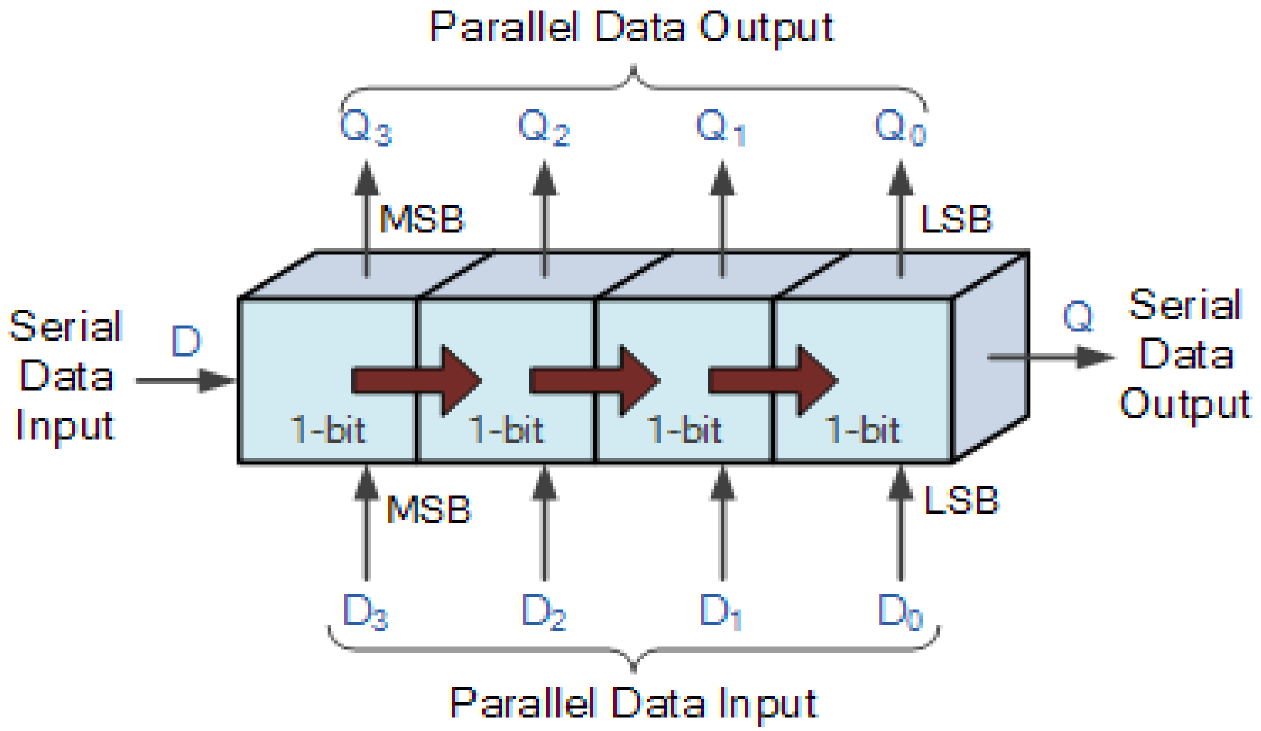
- **Data bits may be fed in or out of a shift register serially, that is one after the other from either the left or the right direction, or all together at the same time in a parallel configuration.**
- **The number of individual data latches required to make up a single Shift Register device is usually determined by the number of bits to be stored with the most common being 8-bits (one byte) wide constructed from eight individual data latches.**
- ***Shift Registers* are used for data storage or for the movement of data and are therefore commonly used inside calculators or computers to store data such as two binary numbers before they are added together, or to convert the data from either a serial to parallel or parallel to serial format. The individual data latches that make up a single shift register are all driven by a common clock (Clk) signal making them synchronous devices.**
- **Shift register IC's are generally provided with a *clear* or *reset* connection so that they can be "SET" or "RESET" as required.**

Types of Shift Registers

- **Generally, shift registers operate in one of four different modes with the basic movement of data through a shift register being:**
- **Serial-in to Parallel-out (SIPO) - the register is loaded with serial data, one bit at a time, with the stored data being available at the output in parallel form.**
- **Serial-in to Serial-out (SISO) - the data is shifted serially “IN” and “OUT” of the register, one bit at a time in either a left or right direction under clock control.**
- **Parallel-in to Serial-out (PISO) - the parallel data is loaded into the register simultaneously and is shifted out of the register serially one bit at a time under clock control.**
- **Parallel-in to Parallel-out (PIPO) - the parallel data is loaded simultaneously into the register, and transferred together to their respective outputs by the same clock pulse.**

Continued…

- **The effect of data movement from left to right through a shift register can be presented graphically as:**

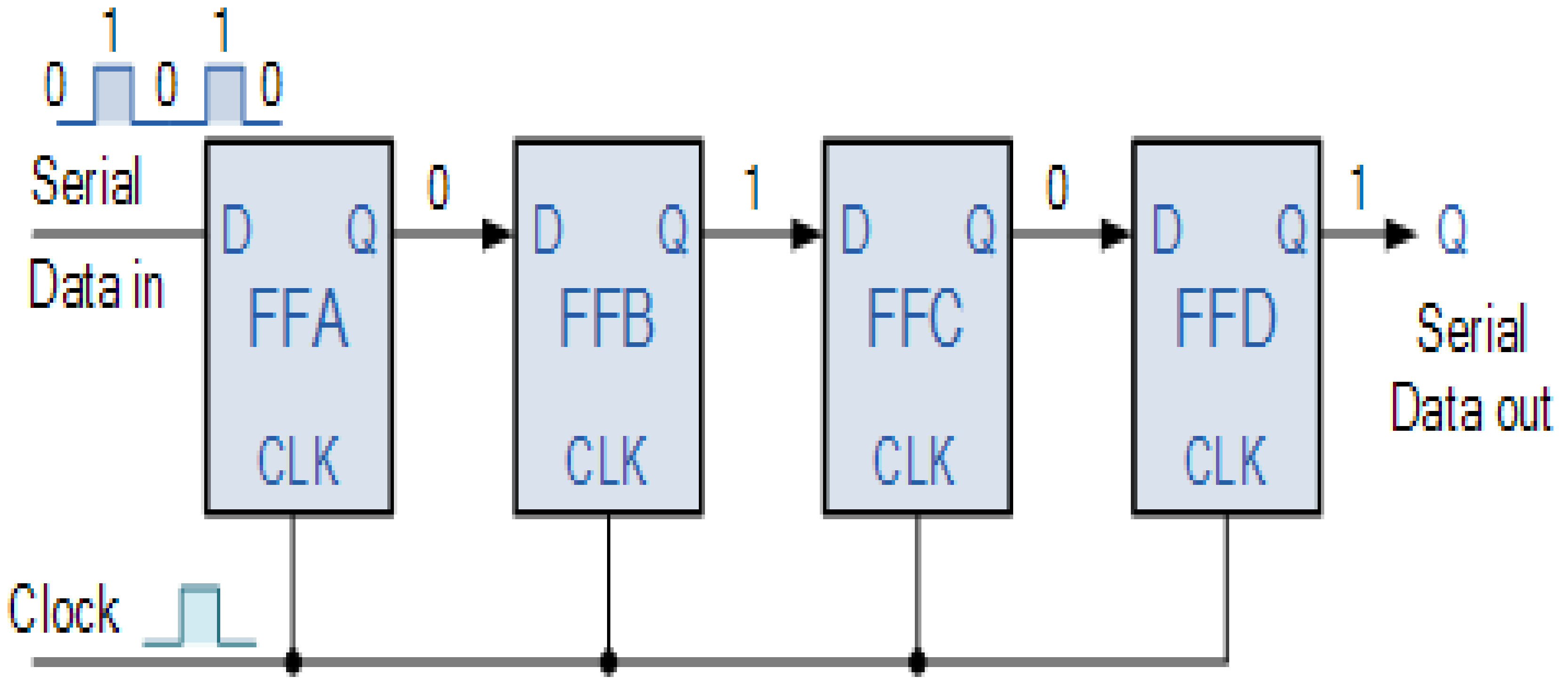


Serial-in to Serial-out (SISO) Shift Register

- **The SISO shift register is one of the simplest of the four configurations as it has only three connections, the serial input (SI) which determines what enters the left hand flip-flop, the serial output (SO) which is taken from the output of the right hand flip-flop and the sequencing clock signal (Clk). The logic circuit diagram below shows a generalized serial-in serial-out shift register.**

- **In this shift register, the data is allowed to flow straight through the register and out of the other end. Since there is only one output, the DATA leaves the shift register one bit at a time in a serial pattern, hence the name Serial-in to Serial-Out Shift Register or SISO.**

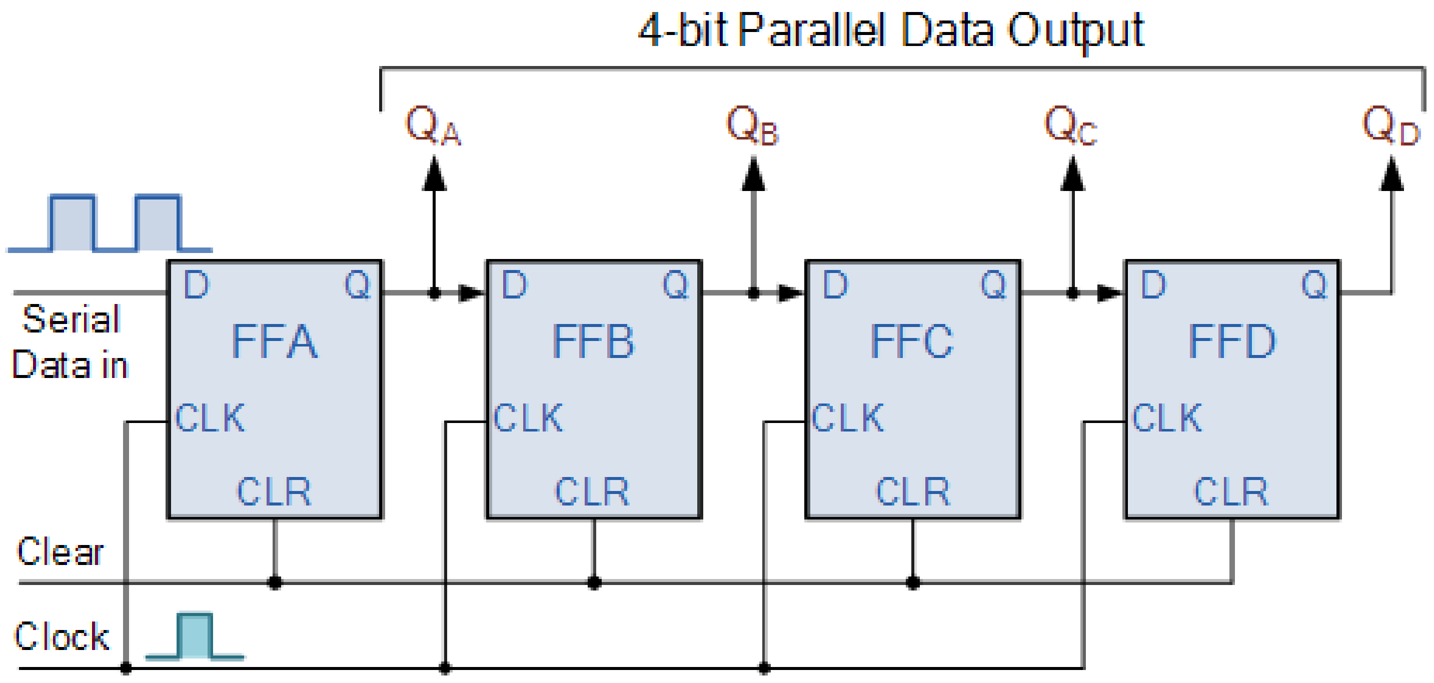
4-bit Serial-in to Serial-out Shift Register



- **You may think what’s the point of a SISO shift register if the output data is exactly the same as the input data. Well this type of Shift Register also acts as a temporary storage device or it can act as a time delay device for the data, with the amount of time delay being controlled by the number of stages in the register, 4, 8, 16 etc or by varying the application of the clock pulses. Commonly available IC’s include the 74HC595 8-bit Serial-in to Serial-out Shift Register all with 3-state outputs.**

Serial-in to Parallel-out (SIPO) Shift Register

- **4-bit Serial-in to Parallel-out Shift Register**

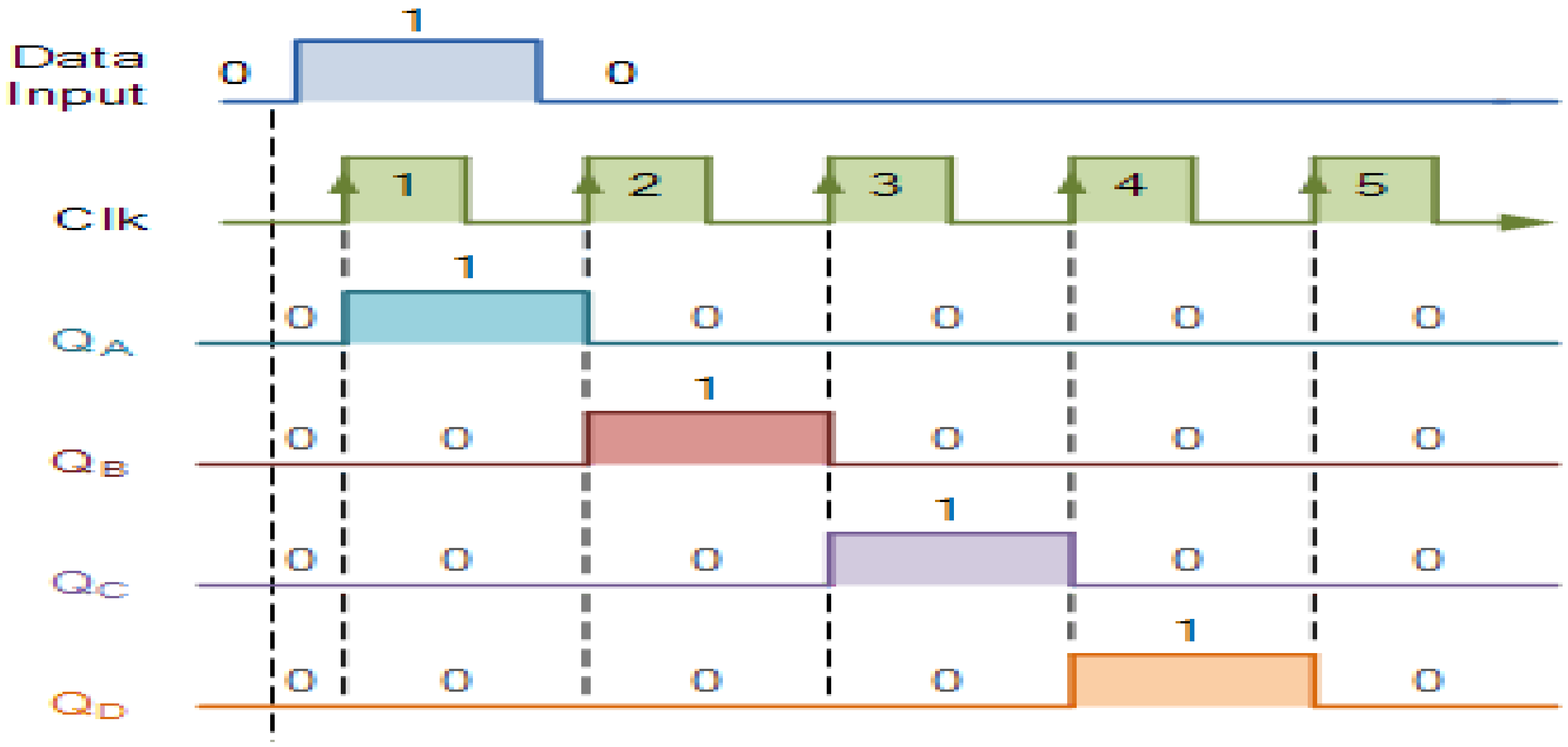


The operation

- Lets assume that all the flip-flops (FFA to FFD) have just been RESET (CLEAR input) and that all the outputs Q_A to Q_D are at logic level “0” ie, no parallel data output.
- If a logic “1” is connected to the DATA input pin of FFA then on the first clock pulse the output of FFA and therefore the resulting Q_A will be set HIGH to logic “1” with all the other outputs still remaining LOW at logic “0”. Assume now that the DATA input pin of FFA has returned LOW again to logic “0” giving us one data pulse or 0-1-0.
- The second clock pulse will change the output of FFA to logic “0” and the output of FFB and Q_B HIGH to logic “1” as its input D has the logic “1” level on it from Q_A. The logic “1” has now moved or been “shifted” one place along the register to the right as it is now at Q_A.

- **When the third clock pulse arrives this logic "1" value moves to the output of FFC (Q_C) and so on until the arrival of the fifth clock pulse which sets all the outputs Q_A to Q_D back again to logic level "0" because the input to FFA has remained constant at logic level "0".**
- **The effect of each clock pulse is to shift the data contents of each stage one place to the right, and this is shown in the following table until the complete data value of 0-0-0-1 is stored in the register. This data value can now be read directly from the outputs of Q_A to Q_D.**
- **Then the data has been converted from a serial data input signal to a parallel data output. The truth table and following waveforms show the propagation of the logic "1" through the register from left to right as follows.**

Basic Data Movement Through A Shift Register



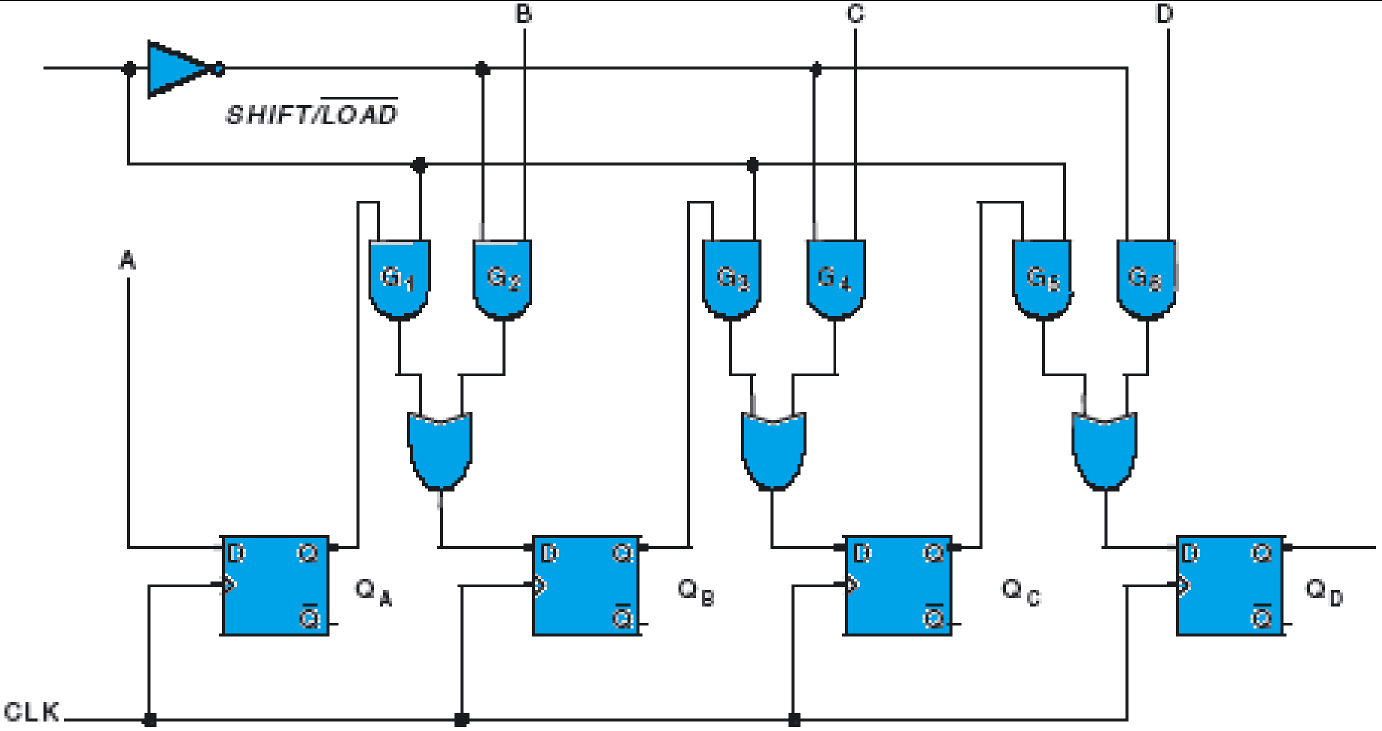
Note that after the fourth clock pulse has ended the 4-bits of data (0-0-0-1) are stored in the register and will remain there provided clocking of the register has stopped. In practice the input data to the register may consist of various combinations of logic “1” and “0”. Commonly available SIPO IC’s include the standard 8-bit 74LS164 or the 74LS594.

Parallel-in to Serial-out (PISO) Shift Register

- **The Parallel-in to Serial-out shift register acts in the opposite way to the serial-in to parallel-out one above. The data is loaded into the register in a parallel format in which all the data bits enter their inputs simultaneously, to the parallel input pins P_A to P_D of the register. The data is then read out sequentially in the normal shift-right mode from the register at Q representing the data present at P_A to P_D.**
- **This data is outputted one bit at a time on each clock cycle in a serial format. It is important to note that with this type of data register a clock pulse is not required to parallel load the register as it is already present, but four clock pulses are required to unload the data.**

4-bit Parallel-in to Serial-out Shift Register

- **As name suggests the input data will enter in parallel that means at a time to all flip flop. And output will get serially.**
- **Here the data bits are entered into the flip flops simultaneously, rather than a bit-by-bit basis.**
- **Let take an example suppose we have to save a 4-bit number (1011). Then all input bits are fed to the inputs of different 4 number of flip flops. With single clock pulse all data enter to all 4 flip flops. See the block diagram of 4 bit of parallel in serial out shift register.**



A 4-bit parallel-in–serial-out shift register.

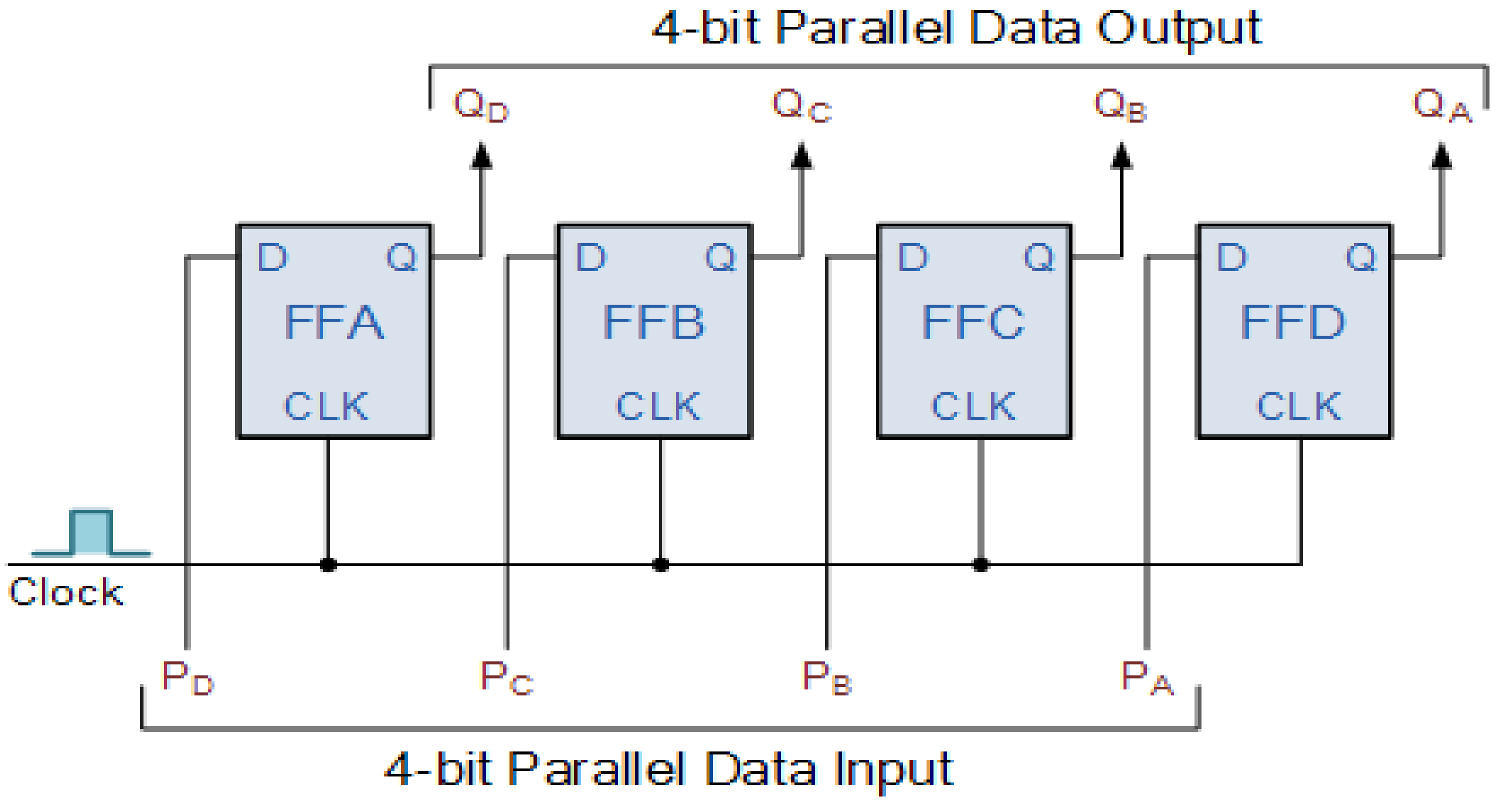
- **Now from above 4 bit parallel in serial out shift register we can see, A, B, C, and D are the four parallel data input lines and**
- ***SHIFT* / *LOAD* (*SH* / *LD*) is a control input that allows the four bits of data at A, B, C, and D inputs to enter into the register in parallel or shift the data in serial.**
- **When *SHIFT* / *LOAD* is HIGH, AND gates G1, G3, and G5 are enabled, allowing the data bits to shift right from one stage to the next.**
- **When *SHIFT* / *LOAD* is LOW, AND gates G2, G4, and G6 are enabled, allowing the data bits at the parallel inputs.**

- **When a clock pulse is applied, the flip-flops with D = 1 will be set and the flip-flops wit D = 0 will be reset, thereby storing all the four bits simultaneously.**
- **The OR gates allow either the normal shifting operation or the parallel data-entry operation, depending on which of the AND gates are enabled by the level on the *SHIFT* / *LOAD* input.**
- **As this type of shift register converts parallel data, such as an 8-bit data word into serial format, it can be used to multiplex many different input lines into a single serial DATA stream which can be sent directly to a computer or transmitted over a communications line. Commonly available IC's include the 74HC166 8-bit Parallel-in/Serial-out Shift Registers.**

Parallel-in to Parallel-out (PIPO) Shift Register

- Operation : The data is presented in a parallel format to the parallel input pins P_A to P_D and then transferred together directly to their respective output pins Q_A to Q_A by the same clock pulse. Then one clock pulse loads and unloads the register. This arrangement for parallel loading and unloading is shown below.
- This type of shift register also acts as a temporary storage device or as a time delay device similar to the SISO configuration above.

Parallel-in to Parallel-out (PIPO) Shift Register



What is a Counter?

- **A counter is a device which can count any particular event on the basis of how many times the particular event(s) is occurred. In a digital logic system or computers, this counter can count and store the number of time any particular event or process have occurred, depending on a clock signal. Most common type of counter is sequential digital logic circuit with a single clock input and multiple outputs. The outputs represent binary or binary coded decimal numbers. Each clock pulse either increase the number or decrease the number.**
- **There are 2 types of counters.**

1. **Asynchronous (Ripple).**
2. **Synchronous.**

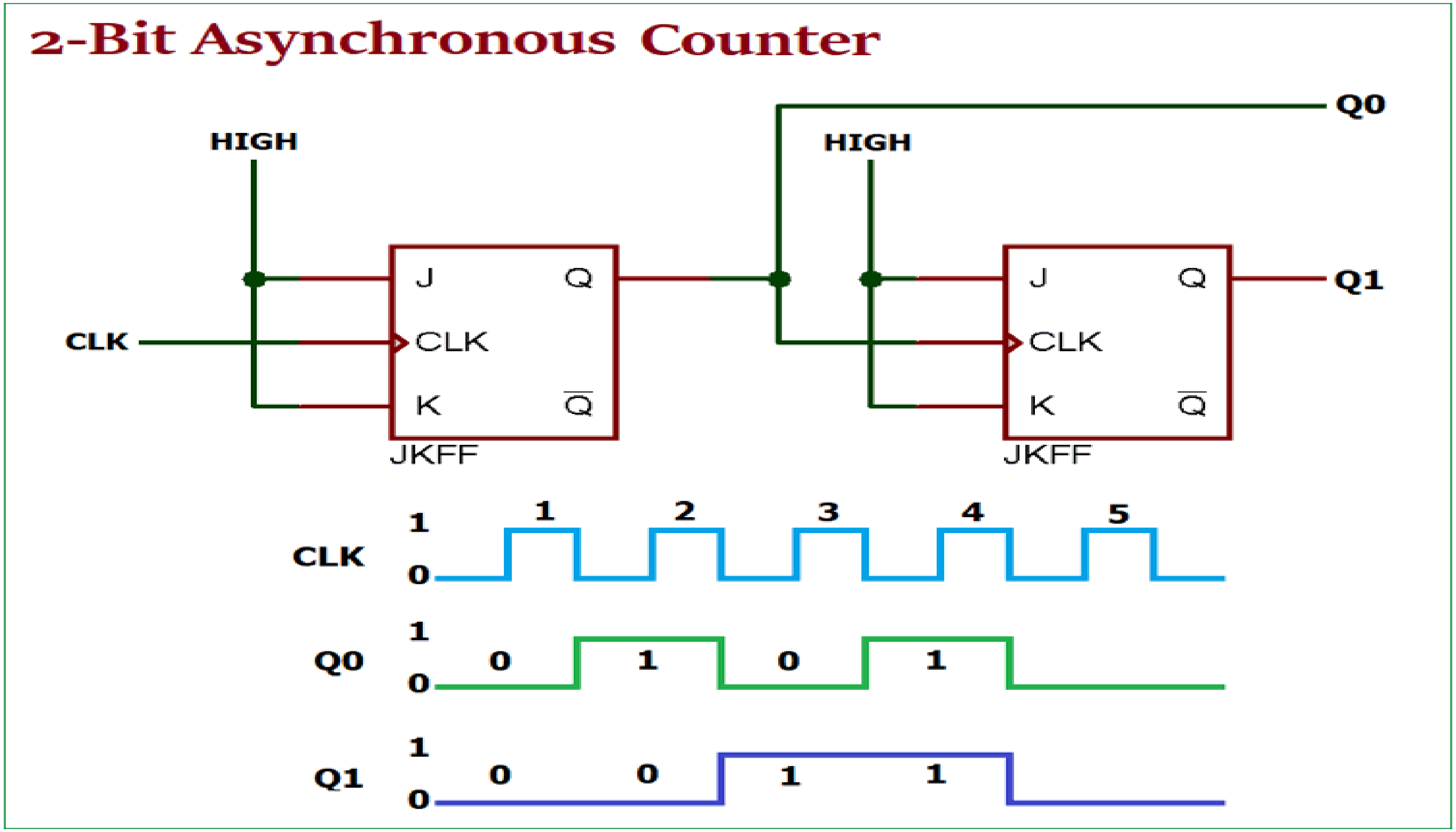
What is Asynchronous?

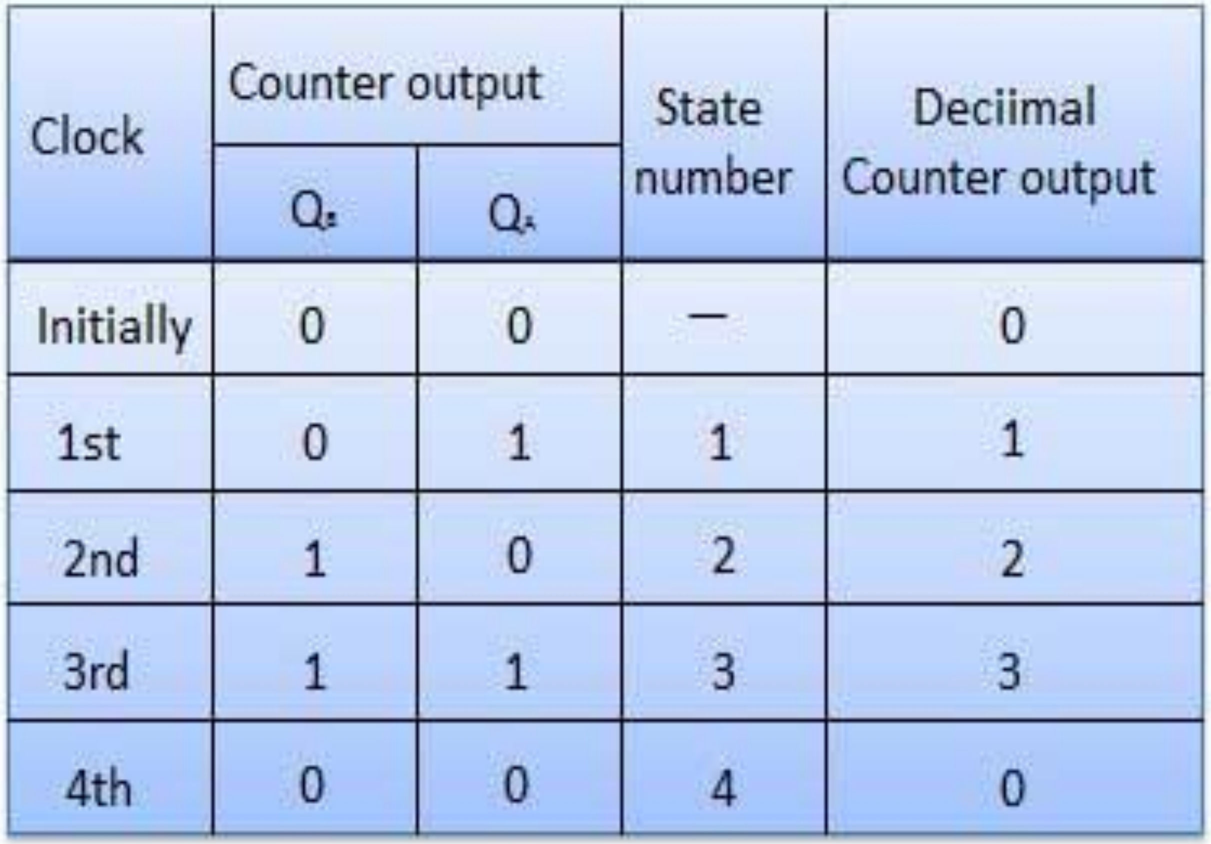
- **Asynchronous stands for the absence of synchronization. Something that is not existing or occurring at the same time. In computing or telecommunication stream, Asynchronous stands for controlling the operation timing by sending a pulse only when the previous operation is completed rather than sending it in regular intervals.**
- **Asynchronous Counter ((Ripple)**
- **A ripple counter is an asynchronous counter where only the first flip-flop is clocked by an external clock. All subsequent flip-flops are clocked by the output of the preceding flip-flop. Asynchronous counters are also called ripple-counters because of the way the clock pulse ripples it way through the flip-flops.. An Asynchronous counter can count 2^n - 1 possible counting states.**

- **The MOD of the ripple counter or asynchronous counter is 2^n if n flip-flops are used. For a 4-bit counter, the range of the count is 0000 to 1111 (2^4-1). A counter may count up or count down or count up and down depending on the input control. The count sequence usually repeats itself. When counting up, the count sequence goes from 0000, 0001, 0010, ... 1110 , 1111 , 0000, 0001, ... etc. When counting down the count sequence goes in the opposite manner: 1111, 1110, ... 0010, 0001, 0000, 1111, 1110, ... etc.**

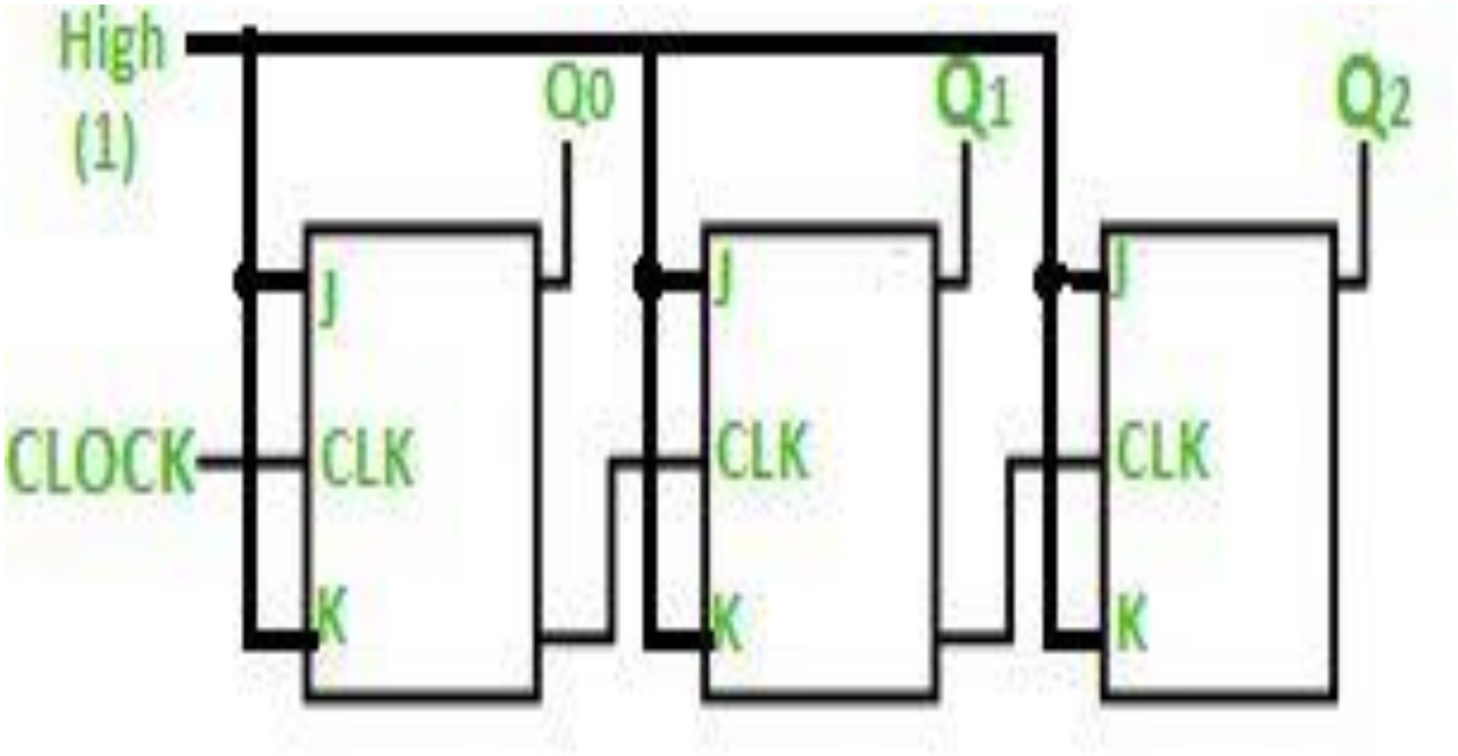
- Asynchronous counters are slower than synchronous counters because of the delay in the transmission of the pulses from flip-flop to flip-flop. With a synchronous circuit, all the bits in the count change synchronously with the assertion of the clock. Examples of synchronous counters are the Ring and Johnson counter.
- It can be implemented using D-type flip-flops or JK-type flip-flops.
- The circuit below uses 2 JK flip-flops to implement a divide-by-4 ripple counter ($2^n = 2^2 = 4$).

2-bit Binary Ripple Counter using JK Flip Flop





Clock	Counter output		State number	Deciimal Counter output
	Q_B	Q_A		
Initially	0	0	—	0
1st	0	1	1	1
2nd	1	0	2	2
3rd	1	1	3	3
4th	0	0	4	0


High
(1)
Q0
Q1
Q2
J
CLK
K
J
CLK
K
J
CLK
K
CLOCK

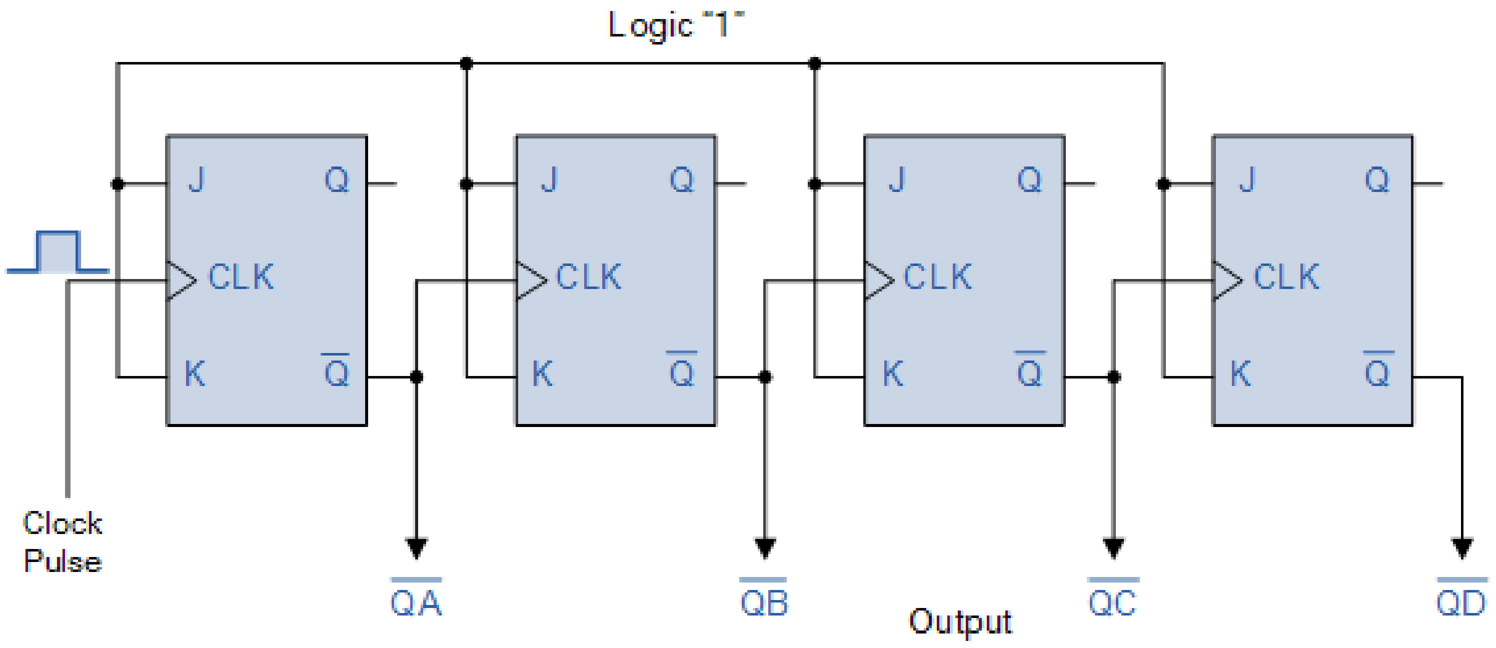
Down Counter

Clock Input	Output		
Count	Q_C	Q_B	Q_A
7	1	1	1
6	1	1	0
5	1	0	1
4	1	0	0
3	0	1	1
2	0	1	0
1	0	0	1
0	0	0	0

Advantages and Disadvantages of Asynchronous Counter

- **Advantages :**
  - **Are easy to build.**
  - **Offer more flexibility on larger counting range related applications.**
  - **Truncated counter can produce any modulus number count.**
- **Disadvantages :**
  - **Additional re-synchronizing output flip-flops required for resynchronizing the flip flops.**
  - **When counting a large number, due to the chain system, propagation delay by successive stages became too large which is very difficult to get rid off.**
- **In such a situation, Synchronous counters are faster and reliable !**

4-bit Count Down Counter



- In the 4-bit counter above the output of each flip-flop changes state on the falling edge (1-to-0 transition) of the CLK input which is triggered by the Q output of the previous flip-flop, rather than by the Q output as in the up counter configuration. As a result, each flip-flop will change state when the previous one changes from 0 to 1 at its output, instead of changing from 1 to 0.

DESIGN OF SYNCHRONOUS COUNTERS

- **COUNTER DESIGN PROCEDURE:**
- **1.Describe a general sequential circuit in terms of its basic parts and its input and outputs.**
- **2.Develop a state diagram for a given sequence.**
- **3.Develop a next-state table for a specific counter sequence.**
- **4.Create a FF transition table.**