
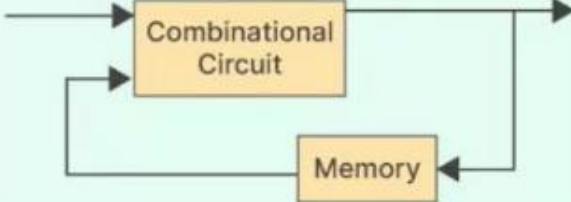


UNIT IV: Sequential Logic and Sequential Circuit

Que.: Give the difference between combinational circuit and sequential circuit.

Ans:-

Combinational Circuit	Sequential Circuit
Output only depends on the present input	Output depends on present input & past output
Memory element is absent	Memory element is present
No clock signal is applied	Clock signal is required
	
Example - Half Adder, Full Adder, Multiplexer	Example - Flipflop, Counters, Registers

Parameters	Combinational Circuit	Sequential Circuit
Meaning and Definition	It is a type of circuit that generates an output by relying on the input it receives at that instant, and it stays independent of time.	It is a type of circuit in which the output does not only rely on the current input. It also relies on the previous ones.
Feedback	A Combinational Circuit requires no feedback for generating the next output. It is because its output has no dependency on the time instance.	The output of a Sequential Circuit, on the other hand, relies on both- the previous feedback and the current input. So, the output generated from the previous inputs gets transferred in the form of feedback. The circuit uses it (along with inputs) for generating the next output.

Performance	We require the input of only the current state for a Combinational Circuit. Thus, it performs much faster and better in comparison with the Sequential Circuit.	In the case of a Sequential Circuit, the performance is very slow and also comparatively lower. Its dependency on the previous inputs makes the process much more complex.
Complexity	It is very less complex in comparison. It is because it basically lacks implementation of feedback.	This type of circuit is always more complex in its nature and functionality. It is because it implements the feedback, depends on previous inputs and also on clocks.
Elementary Blocks	Logic gates form the building/ elementary blocks of a Combinational Circuit.	Flip-flops form the building/ elementary blocks of a Sequential Circuit.
Operation	One can use these types of circuits for both- Boolean as well as Arithmetic operations.	You can mainly make use of these types of circuits for storing data.

Que :-What are different types triggering in flipflop.

Ans; Triggering Methods of Flip-Flop

Types of triggering

- Level Triggering
- Edge Triggering

Level Triggering:

- There are two levels, namely logic high and logic low in the clock signal
- Level triggering is of two types
 - Positive level triggering
 - Negative level triggering
- If the sequential circuit is operated with the clock signal when it is in logic high, then that type of triggering is known as a **positive level triggering**.



Fig:- Positive Level Triggering

- If the sequential circuit is operated with the clock signal when it is in logic low, then that type of triggering is known as **Negative level triggering**.



Fig:- Negative Level Triggering

Edge Triggering:

- Two types of transition occur in the clock signal. That means, the clock signal transitions either from logic low to logic high or logic high to logic low.
- Types of edge triggering based on the transition of the clock signal.
 - Positive edge triggering
 - Negative edge triggering
- If the sequential circuit is operated with the clock signal that is transitioning from logic low to logic high, then that type of triggering is known as **positive edge triggering**. It is also called as rising edge triggering.



Fig:- Positive Edge Triggering

- If the sequential circuit is operated with the clock signal that is transitioning from logic high to logic low, then that type of triggering is known as **Negative edge triggering**. It is also called as falling edge triggering.



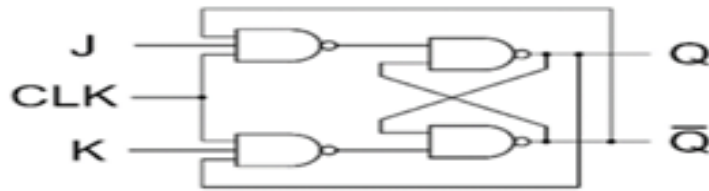
Fig:- Negative Edge Triggering

Que : What is race around condition? How it will be overcome, Explain in details.

Ans: Race around condition occurs in JK Flip Flop when both its inputs $J=K=1$ and also the clock =1 for a long period.

J-K Flip Flop

The truth table is shown below with its circuit diagram.

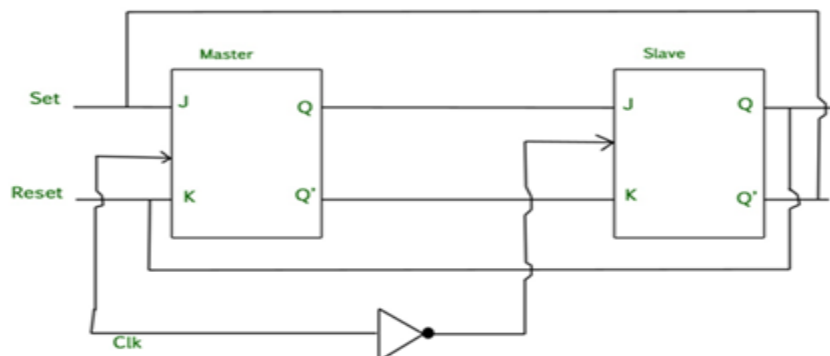


TRUTH TABLE

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

Race Around Condition in JK Flip-flop

- For J-K flip-flop, if $J=K=1$, and if $clk=1$ for a long period of time, then output Q will toggle as long as CLK remains high which makes the output unstable or uncertain.
- This is called a race around condition in J-K flip-flop.
- We can overcome this problem by making the clock =1 for very less duration.
- The circuit used to overcome race around conditions is called the Master **Slave JK** flip flop.
- Master Slave JK flip flop**
- Here two JK flip flops are connected in series.
- The first JK flip flop is called the “**master**” and the other is a “**slave**”.
- The output from the master is connected to the two inputs of the slave whose output is fed back to inputs of the master.
- The circuit also has an inverter other than the two flip flops.
- The Clock Pulse and inverter are connected because of which the flip flops get an inverted clock pulse.
- In other words, if $CP=0$ for a master flip-flop, then $CP=1$ for a slave flip-flop and vice versa.



Working of a Master Slave flip flop

- When the clock pulse goes high, the slave is isolated; J and K inputs can affect the state of the system. The slave flip-flop is isolated when the CP goes low.
- When the CP goes back to 0, information is transmitted from the master flip-flop to the slave flip-flop and output is obtained.
- As the master flip flop is positive triggered it responds first and the slave later (it is negative edge triggered).
- The master goes to the K input of the slave when both inputs $J=0$ and $K=1$, and also $Q' = 1$. In this case the slave copies the master as the clock forces the slave to reset.
- The master goes to the J input of the slave when both $J=1$ and $K=0$, $Q = 1$. The clock is set due to the negative transition of the clock.
- There is a state of toggle when both $J=1$ and $K=1$. On the negative transition of clock slave toggles and the master toggles on the positive transition of the clock.
- Both the flip flops are disabled when both $J=0$ and $K=0$ and Q is unchanged.

Counters

1. Compare Asynchronous counter and Synchronous counter

Ans: Let's see the difference between these two counters:

.NO	Synchronous Counter	Asynchronous Counter
1.	In synchronous counter we use a universal clock that is common to all flip flops throughout the circuit.	In asynchronous counter main clock is only applied to the first flip flop and then for rest of flip flops the output of previous flip flop is taken as a clock.
2.	Synchronous Counter is faster in operation as compared to Asynchronous Counter.	Asynchronous Counter is slower as compared to synchronous counter in operation.
3.	Synchronous Counter does not produce any decoding errors.	Asynchronous Counter produces decoding error.
4.	Synchronous Counter is also called Parallel Counter.	Asynchronous Counter is also called Serial Counter.
5.	Synchronous Counter designing as well implementation are	Asynchronous Counter designing as well as implementation is very easy.

	complex due to increasing the number of states.	
6.	Synchronous Counter will operate in any desired count sequence.	Asynchronous Counter will operate only in fixed count sequence (UP/DOWN).
7.	Synchronous Counter examples are: Ring counter, Johnson counter.	Asynchronous Counter examples are: Ripple UP counter, Ripple DOWN counter.
8.	In synchronous counter, propagation delay is less.	In asynchronous counter, there is high propagation delay.

2. Draw and Explain three-bit ripple Up counter with timing diagram.

Ans: Ripple counter is a cascaded arrangement of flip-flops where the output of one flip-flop drives the clock input of the following flip-flop.

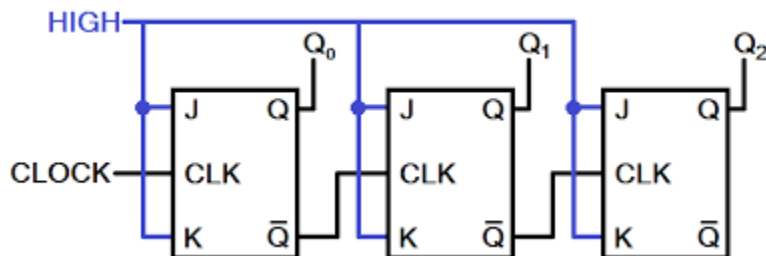
A n-bit ripple counter can count up to 2^n states. It is also known as MOD n counter. It is known as ripple counter because of the way the clock pulse ripples its way through the flip-flops. Some of the features of ripple counter are:

- It is an asynchronous counter.
- Different flip-flops are used with a different clock pulse.
- All the flip-flops are used in toggle mode.
- Only one flip-flop is applied with an external clock pulse and another flip-flop clock is obtained from the output of the previous flip-flop Q' (Complement of Q)
- The flip-flop applied with an external clock pulse act as LSB (Least Significant Bit) in the counting sequence.

The truth table of the count obtained on every positive edge is as shown.

Counter State	Q_2	Q_1	Q_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

A 3-bit Ripple counter using a JK flip-flop is as follows:



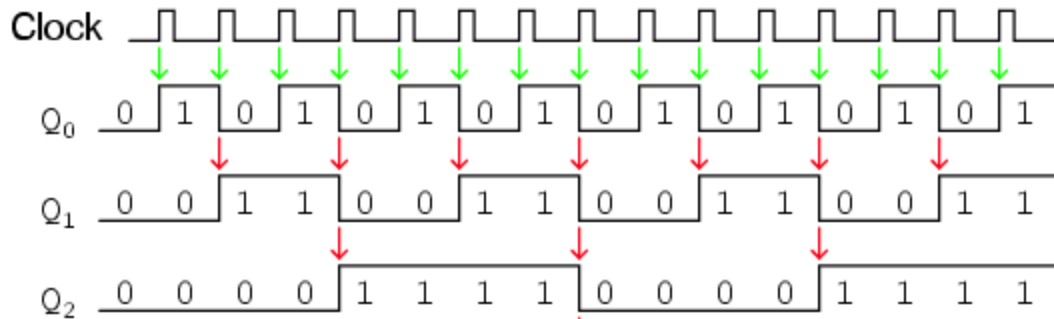
In the circuit shown in the above figure, Q_0 (LSB) will toggle for every clock pulse because JK flip-flop works in toggle mode when both J and K are applied 1, 1, or high input. The following counter will toggle when the previous one changes from 1 to 0.

The 3-bit ripple counter used in the circuit above has eight different states, each one of which represents a count value. Similarly, a counter having n flip-flops can have a maximum of 2 to the power n states. The number of states that a counter owns is known as its mod (modulo) number. Hence a 3-bit counter is a mod-8 counter.

Timing diagram

The timing diagram shows the transition of the outputs on every positive edge of the clock.

Let us assume that the clock is positive edge triggered so the above the counter will act as an up counter because the clock is positive edge triggered and output is taken from Q.



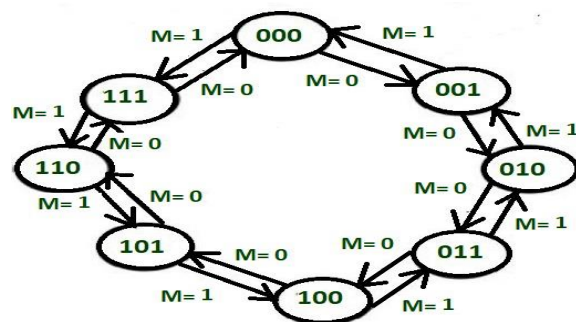
1. Design of a synchronous 3-bit up-down counter using JK Flip-Flops

Step1: determine the number of flip-flops required.

Here we are performing 3 bit or mod-8 **Up or Down counting**, so 3 Flip Flops are required, which can count up to $2^3-1 = 7$.

A 3-bit counter requires three FFs. It has 8 states (000,001,010,011,101,110,111) and all the states are valid. Hence no don't cares. For selecting up and down modes, a control or mode signal M is required. When the mode signal $M=1$ and counts down when $M=0$. The clock signal is applied to all the FFs simultaneously.

Step2: draw the state diagrams: the state diagram of the 3-bit up-down counter is drawn as



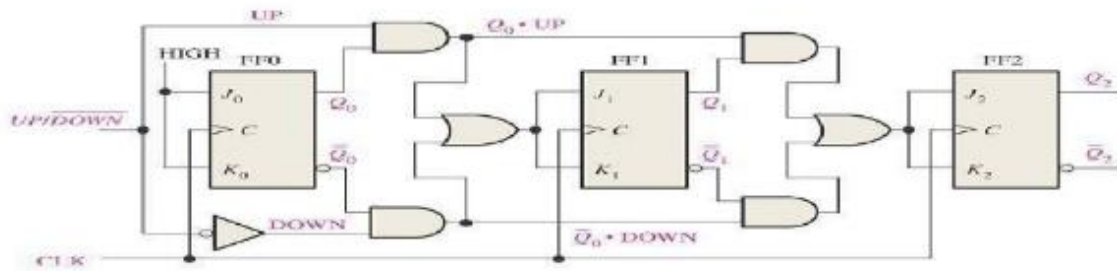
Step3: select the type of flip flop and draw the excitation table: JK flip-flops are selected and the excitation table of a 3-bit up-down counter using JK flip-flops is drawn as shown in fig.

PS			mode	NS			required excitations					
Q3	Q2	Q1	M	Q3	Q2	Q1	J3	K3	J2	K2	J1	K1
0	0	0	0	1	1	1	1	x	1	x	1	x
0	0	0	1	0	0	1	0	x	0	x	1	x
0	0	1	0	0	0	0	0	x	0	x	x	1
0	0	1	1	0	1	0	0	x	1	x	x	1
0	1	0	0	0	0	1	0	x	x	1	1	x
0	1	0	1	0	1	1	0	x	x	0	1	x
0	1	1	0	0	1	0	0	x	x	0	x	1
0	1	1	1	1	0	0	1	x	x	1	x	1
1	0	0	0	0	1	1	x	1	1	x	1	x
1	0	0	1	1	0	1	x	0	0	x	1	x
1	0	1	0	1	0	0	x	0	0	x	x	1
1	0	1	1	1	1	0	x	0	1	x	x	1
1	1	0	0	1	0	1	x	0	x	1	1	x
1	1	0	1	1	1	1	x	0	x	0	1	x
1	1	1	0	1	1	0	x	0	x	0	x	1
1	1	1	1	0	0	0	x	1	x	1	x	1

Step4: obtain the minimal expressions: From the excitation table we can conclude that $J1=1$ and $K1=1$, because all the entries for $J1$ and $K1$ are either X or 1. The K-maps for $J3$, $K3$, $J2$ and $K2$ based on the excitation table and the minimal expression obtained from them are shown in fig

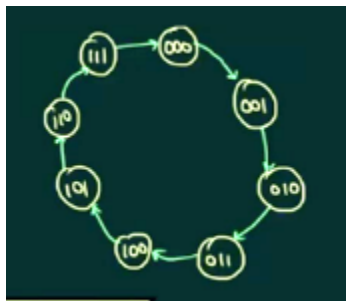
		Q1Q0			
		00	01	11	10
Q3Q2	00	1			
	01			1	
Q3Q2	10	X	X	X	X
	11	X	X	X	X

Step5: draw the logic diagram: a logic diagram using those minimal expressions can be drawn as shown in fig.



Que.4. Implement synchronous 3 bit up counter using JK flip flop.

State diagram:



Characteristic Table for 3 bit UP counter

Q	Q _{t+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Truth Table for 3 bit UP Counter

Present State			Next State			A		B		C	
Q _A	Q _B	Q _C	Q _{A+1}	Q _{B+1}	Q _{C+1}	J _A	K _A	J _B	K _B	J _C	K _C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	1	0	1	0	X	X	0	1	X

0 1 1	1 0 0	1 X	X 1	X 1
1 0 0	1 0 1	X 0	0 X	1 X
1 0 1	1 1 0	X 0	1 X	X 1
1 1 0	1 1 1	X 0	X 0	1 X
1 1 1	0 0 0	X 1	X 1	X 1

K Map

TA

$Q_B Q_C$	00	01	11	10
Q_A 0	0	0	1	0
Q_A 1	X	X	X	X

$TA = Q_B Q_C$

KA

$Q_B Q_C$	00	01	11	10
Q_A 0	X	X	X	X
Q_A 1	0	0	1	0

$KA = Q_B Q_C$

TB

$Q_B Q_C$	01	11	10
Q_A 0	0	X	X
Q_A 1	0	X	X

$\therefore TB = Q_C$

KB

$Q_B Q_C$	00	01	11	10
Q_A 0	X	X	1	0
Q_A 1	X	X	1	0

$\therefore KB = Q_C$

Handwritten Karnaugh maps for a 3-bit UP counter.

K_C Map:

\bar{Q}_A	00	01	11	10
0	1	X	1	X
1	X	1	1	X

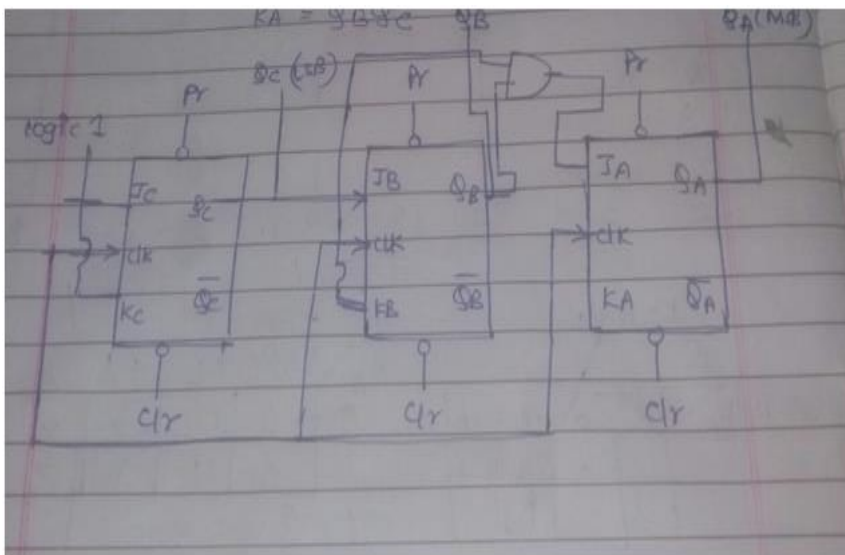
$\therefore K_C = 1$

K_B Map:

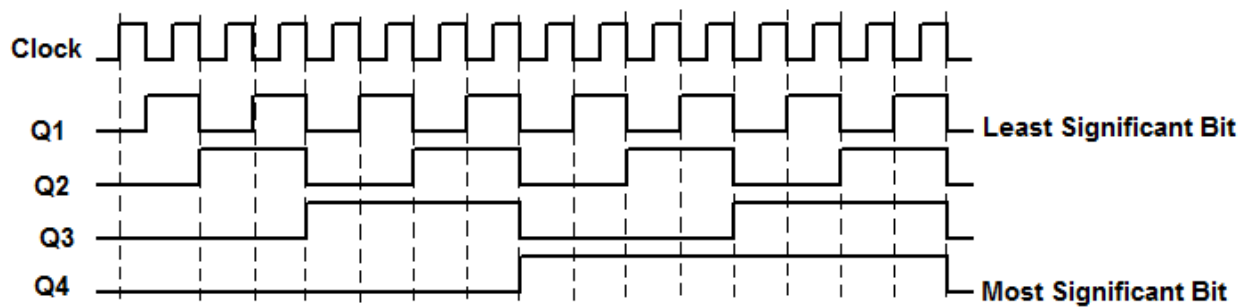
\bar{Q}_A	00	01	11	10
0	1	X	X	1
1	1	X	X	1

$\therefore K_B = 1$

Logic Diagram for 3 bit UP counter



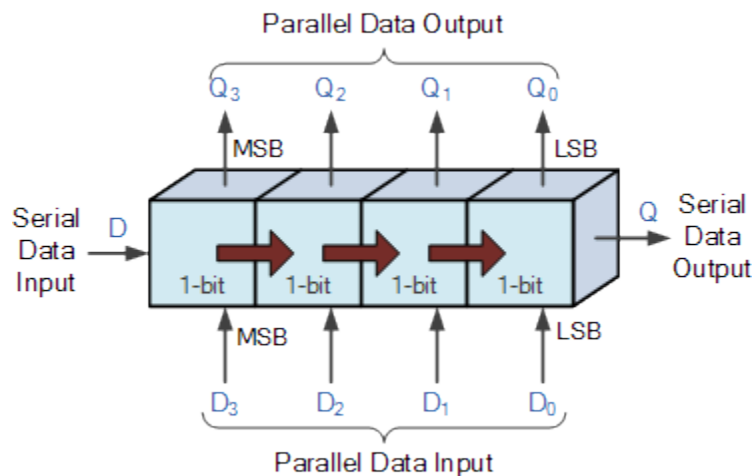
Timing Diagram for 3 bit UP counter



Registers

The Shift Register

The Shift Register is another type of sequential logic circuit that can be used for the storage or the transfer of binary data



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

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.

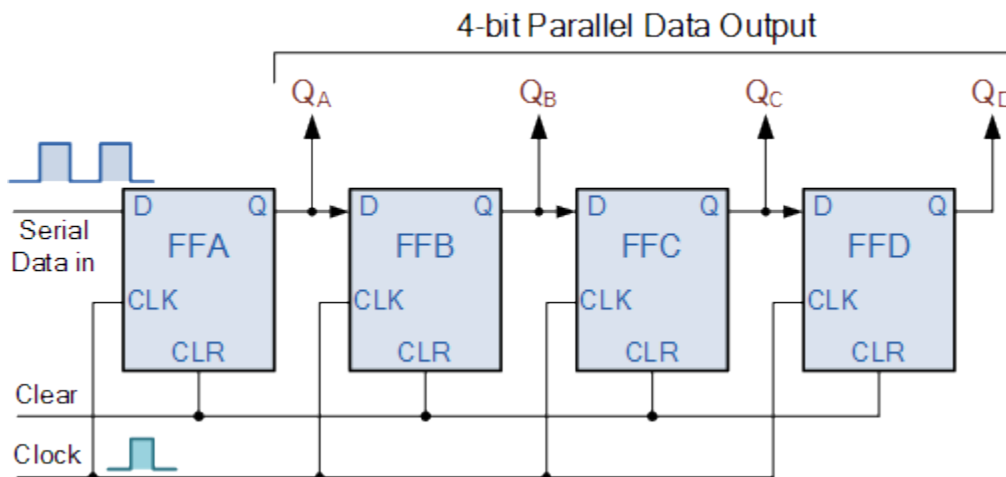
Generally, shift registers operate in one of four different modes with the basic movement of data through a shift register being:

- **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.

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

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



The operation is as follows. Let's 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

Clock Pulse No	Q_A	Q_B	Q_C	Q_D
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

5	0	0	0	0
---	---	---	---	---



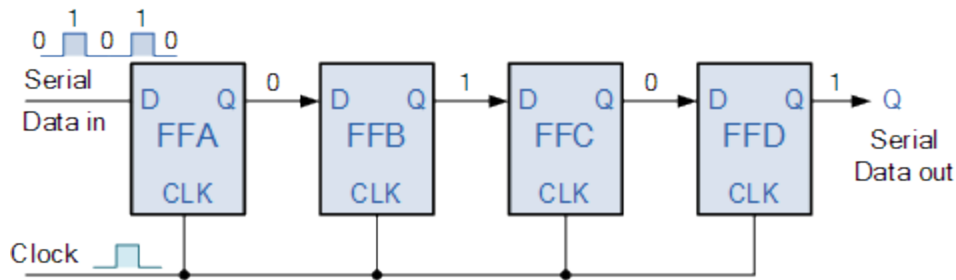
Note that after the fourth clock pulse has ended the 4-bits of data (0-0-0-1) are stored in the register and will remain there provided clocking of the register has stopped. In practice the input data to the register may consist of various combinations of logic “1” and “0”. Commonly available SIPO IC’s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

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

This **shift register** is very similar to the SIPO above, except were before the data was read directly in a parallel form from the outputs Q_A to Q_D , this time 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**.

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.

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



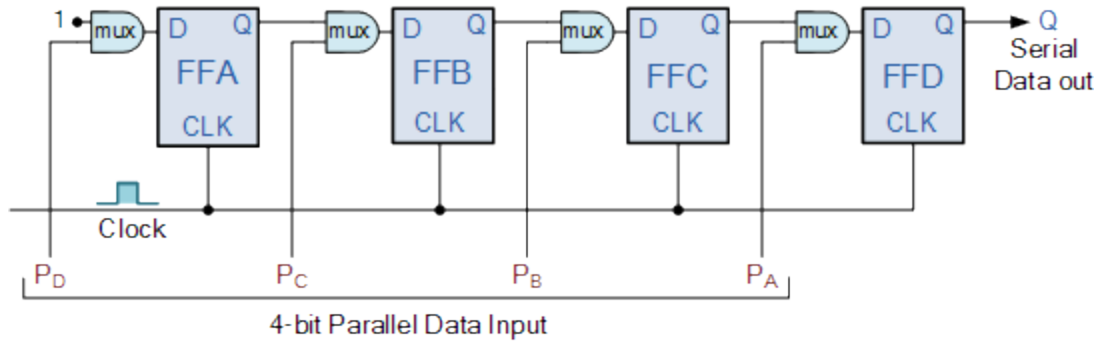
You may think what's the point of a SISO shift register if the output data is exactly the same as the input data. Well this type of **Shift Register** also acts as a temporary storage device or it can act as a time delay device for the data, with the amount of time delay being controlled by the number 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.

Parallel-in to Serial-out (PISO)

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

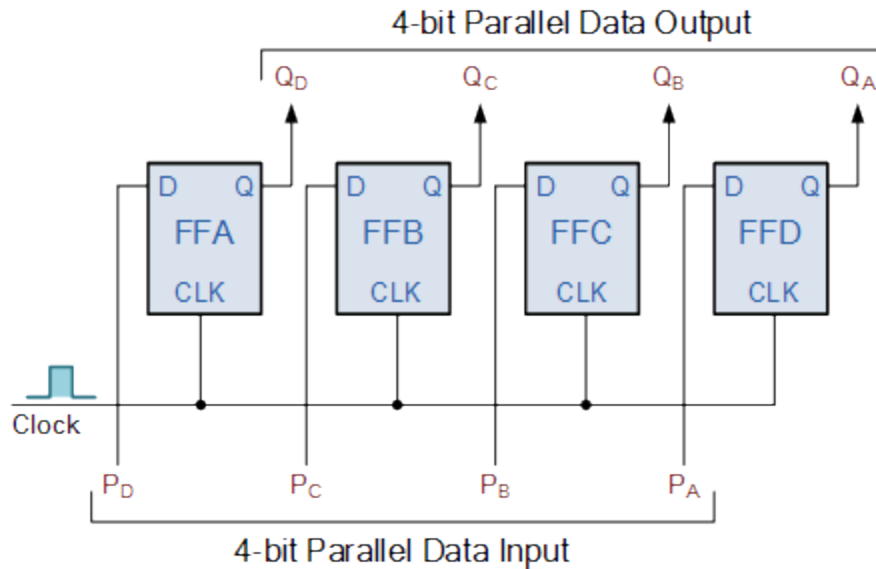


As this type of shift register converts parallel data, such as an 8-bit data word into serial format, it can be used to multiplex many different input lines into a single serial DATA stream which can be sent directly to a computer or transmitted over a communications line. Commonly available IC's include the 74HC166 8-bit Parallel-in/Serial-out Shift Registers.

Parallel-in to Parallel-out (PIPO)

The final mode of operation is the Parallel-in to Parallel-out Shift Register. This type of shift register also acts as a temporary storage device or as a time delay device similar to the SISO configuration above. 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_D by the same clock pulse. Then one clock pulse loads and unloads the register. This arrangement for parallel loading and unloading is shown below.

4-bit Parallel-in to Parallel-out



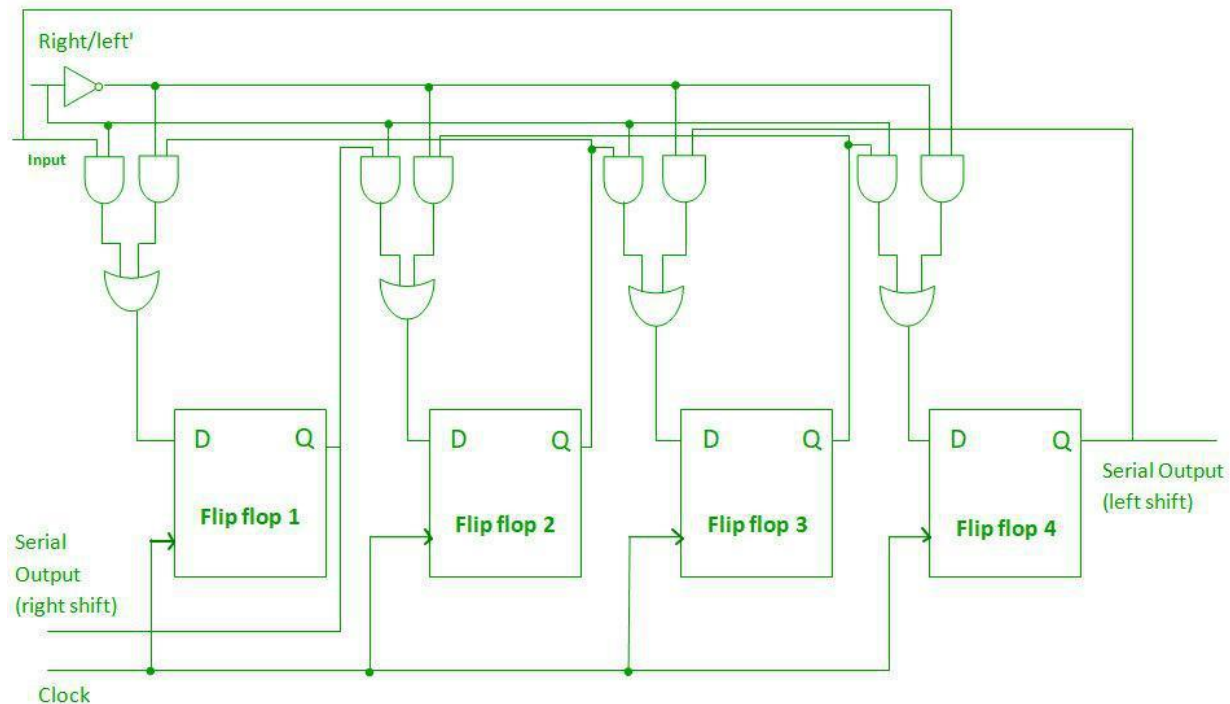
The PIPO shift register is the simplest of the four configurations as it has only three connections, the parallel input (PI) which determines what enters the flip-flop, the parallel output (PO) and the sequencing clock signal (Clk).

Similar to the Serial-in to Serial-out shift register, this type of register also acts as a temporary storage device or as a time delay device, with the amount of time delay being varied by the frequency of the clock pulses. Also, in this type of register there are no interconnections between the individual flip-flops since no serial shifting of the data is required.

Draw and Explain Parallel in, Serial out Shift register.

Draw and explain bidirectional shift register.

A bidirectional shift register is a sequential circuit that can be used for shifting data either to the right or left based on an input signal. It can be implemented using D flip-flops and logic gates, enabling the transfer of data from one stage to the next stage in the desired direction determined by a mode control signal. This flexible functionality makes bidirectional shift registers an essential component in various digital systems.



Working Principle of Bidirectional Shift Register

The bidirectional shift register operates based on clock pulses and control signals. It has two modes of operation: shift right and shift left. Let's explore these modes in detail:

1. **Shift Right:** In the shift right mode, the data is shifted from the leftmost bit to the rightmost bit. Each clock pulse triggers the transfer of data to the adjacent flip-flop on the right. The rightmost bit receives the input data, while the leftmost bit is discarded.
2. **Shift Left:** In the shift left mode, the data is shifted from the rightmost bit to the leftmost bit. Each clock pulse triggers the transfer of data to the adjacent flip-flop on the left. The leftmost bit receives the input data, while the rightmost bit is discarded.

Bidirectional shift registers often have additional control inputs for shifting operations, such as parallel load, asynchronous clear, and synchronous clear. These inputs enable various functionalities, such as loading data into the register, clearing the register, and synchronizing the register with external events.

1. Difference between Moore machine and Mealy machine.

Parameters	Mealy Machine	Moore Machine
Definition	A Mealy Machine changes its output on the basis of its present state and current input.	A Moore Machine's output depends only on the current state. It does not depend on the current input.
Output	Mealy Machine places its output on the transition.	Moore machine also places its output on the transition.
States	It has comparatively fewer or the same states as that of the Moore machine.	It has comparatively fewer or the same states as that of the Mealy machine.
Value of the Output Function	When the input logic is done in the present state, then the value of the	Whenever a change occurs in the state, the output function's value becomes the function

	output function becomes a function of transitions and changes.	of its current state along with the changes at the edges of the clock.
Reaction to the Inputs	A Mealy machine reacts comparatively faster to inputs than the Moore machine. Generally, it reacts in the very same clock cycle.	In a Moore Machine, one requires more logic for decoding the output. As a result, it leads to more delays in the circuit. Generally, these react after one clock cycle.
State and Output	The asynchronous generation of output through its state alters to synchronous on the present clock.	The state and output- both change the synchronous to its clock edge.
Requirement of States	A Mealy Machine generally requires only a very few states for the process of synthesis.	The states for synthesis required for this machine are also more.
Requirement of Hardware	It requires very little hardware for designing a Mealy Machine.	One requires more hardware to design a Moore Machine.
Counter	You cannot refer to the counter as a Mealy Machine.	You can refer to the counter as a Moore Machine.
Design	The designing process doesn't need to be very easy.	It is very easy to design.

