Flip Flops

A digital computer needs devices that can store information. A flip-flop is a binary storage device. It can store binary bits either 0 or 1. It has two stable states HIGH and LOW i.e. 1 and 0. It has the property to remain in one state indefinitely until it is directed by an input signal to switch over to the other state. It is also called a **bistable multivibrator**.

The basic function of the flip-flop is to store data. They can be used to keep a record of the value of a variable (input, output, or intermediate). They are also used to exercise control over the functionality of a digital circuit i.e. change the operation of a circuit depending on the state of one or more flip flops. These devices are mainly used in situations that require one or more of these three. Operations, storage, and sequencing.

Latch Flip Flop

The R-S (Reset Set) flip flop is the simplest of all and easiest to understand. It is basically a device that has two outputs one being the inverse or complement of the other, and two inputs. A pulse on one of the inputs takes on a particular logical state. The outputs will then remain in this state until a similar pulse is applied to the other input. The two inputs are called the Set and Reset inputs (sometimes called the preset and clear inputs).

Such flip flops can be made simply by cross coupling two inverting gates either NAND or NOR gate. Figure 1(a) shows an RS flip flop using the NAND gate and Figure 1(b) shows the same circuit using NOR gate.

Figure 1: Latch R-S Flip Flop Using NAND and NOR Gates

To describe the circuit of Figure 1(a), assume that initially both R and S are at the logic 1 state and that output is at the logic 0 state.

Now, if Q = 0 and R = 1, then these are the states of inputs of gate B, therefore the outputs of gate B is at 1 (making it the inverse of Q i.e. 0). The output of gate B is connected to an input of gate A so if S = 1, both inputs of gate A are at the logic 1 state. This means that the output of gate A must be 0 (as was originally specified). In other words, the 0 state at Q is continuously disabling gate B so that any change in R has no effect. Also, the 1 state at Q is continuously enabling gate A so that any change in S will be transmitted through to Q. The above conditions constitute one of the stable states of the device referred to as the Reset state since Q = 0.

Now suppose that the R-S flip flop in the Reset state, and the S input goes to 0. The output of gate A i.e. Q will go to 1 and with Q = 1 and R = 1, the output of gates B Q will go to 0 with Q now 0 gate A is disabled keeping Q at 1. Consequently, when S returns to the 1 state it has no effect on the flip flop whereas a change in R will cause a change in the output of gate B. The above conditions constitute the other stable state of the device, called the Set state since Q = 1. Note that the change of the state of S from 1 to 0 has caused the flip flop to change from the Reset state to the Set state.

There is another input condition that has not yet been considered. That is when both the R and S inputs are at the logic state 0. When this happens both Q and Q will be forced to 1 and will remain so far as long as R and S are kept at 0. However, when both inputs return to 1 there is no way of knowing whether the flip flop will latch in the Reset state or the Set state. The condition is said to be indeterminate because of this indeterminate state great care must be taken when using R-S flip flop to ensure that both inputs are not instructed simultaneously.

Table 1: The truth table for the NAND R-S flip flop

Initial Conditions	Inputs (Pulsed)		Final Output	
Q	S	R	Q	Q

Table 1: The truth table for the NAND R-S flip flop

Initial Conditions	Inputs (Pulsed)		Final Output	
1	0	0	indeterminate	
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0
0	0	0	indeterminate	
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1

or more simply shown in Table 2

Table 2: Simple NAND R-S Flip Flop Truth Table

S	R	Q
0	0	indeterminate
0	1	Set (1)
1	0	Reset(0)
1	1	No Change

When NOR gates are used the R and S inputs are transposed compared with the NAND version. Also, the stable state when R and S are both 0. A change of state is effected by pulsing the appropriate input to the 1 state. The indeterminate state is now when both R and S are simultaneously at logic 1. Table 3 shows this operation.

Table 3: NOR Gate R-S Flip Flop Truth Table

S	R	Q
0	0	No Change
0	1	Reset (0)
1	0	Set (1)
1	1	Indeterminate

Clocked RS Flip Flop

The RS latch flip flop required direct input but no clock. It is very use full to add a clock to control precisely the time at which the flip-flop changes the state of its output.

In the clocked R-S flip flop the appropriate levels applied to their inputs are blocked till the receipt of a pulse from another source called a clock. The flip-flop changes state only when the clock pulse is applied depending upon the inputs. The basic circuit is shown in Figure 2. This circuit is formed by adding two AND gates at inputs to the R-S flip flop. In addition to control inputs Set (S) and Reset (R), there is a clock input (C) also.

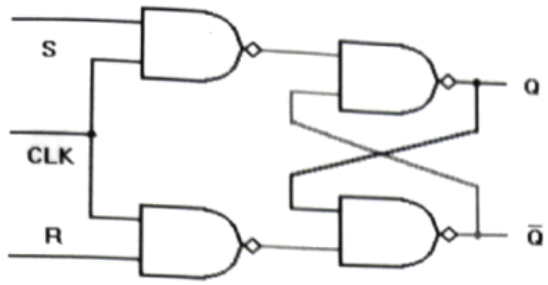


Figure 2: Clocked RS Flip Flop

Table 4: The truth table for the Clocked R-S flip flop

Initial Conditions	Inputs (Pulsed)		Final Output	Comment
Q	S	R	Q (t + 1)	No Change
0	0	0	0	No Change
0	0	1	0	Clear Q
0	1	0	1	Set Q
0	1	1	???	indeterminate
1	0	0	1	No Change
1	0	1	0	Clear Q
1	1	0	1	Set Q
1	1	1	???	indeterminate

The excitation table for R-S flip flop is very simply derived as given below

Table 5: Excitation table for R-S Flip Flop

S	R	Q
0	0	No Change
0	1	Reset (0)
1	0	Set (1)
1	1	Indeterminate

D Flip Flop

A D type (Data or delay flip flop) has a single data input in addition to the clock input as shown in Figure 3.

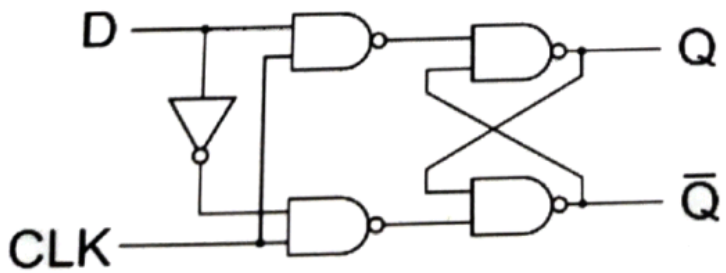


Figure 3: D Flip Flop

Basically, such type of flip flop is a modification of clocked RS flip flop gates from a basic Latch flip flop, and NOR gates modify it into a clock RS flip flop. The D input goes directly to the S input and its complement through the NOT gate, is applied to the R input.

This kind of flip-flop prevents the value of D from reaching the output until a clock pulse occurs. The action of the circuit is straightforward as follows.

When the clock is low, both AND gates are disabled, therefore D can change values without affecting the value of Q. On the other hand, when the clock is high, both AND gates are enabled. In this case, Q is forced equal to D when the clock again goes low, and Q retains or stores the last value of D. The truth table for such a flip-flop is as given below in Table 6.

Table 6: Truth table for D Flip Flop

S	R	Q(t + 1)
0	0	0
0	1	1
1	0	0
1	1	1

The excitation table for D flip flop is very simply derived given as under.

Table 7: Excitation table for D Flip Flop

S	Q
0	0
1	1

JK Flip Flop

One of the most useful and versatile flip flops is the JK flip flop the unique features of a JK flip flop are:

1. If the J and K input are both at 1 and the clock pulse is applied, then the output will change state, regardless of its previous condition.
2. If both J and K inputs are at 0 and the clock pulse is applied there will be no change in the output. There is no indeterminate condition, in the operation of the JK flip flop i.e. it has no ambiguous state. The circuit diagram for this is shown in Figure 4.

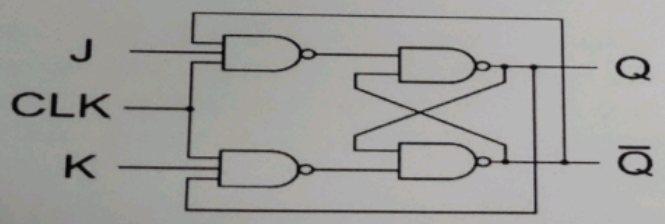


Figure 4: JK Flip Flop

When J = 0 and K = 0

These J and K inputs disable the NAND gates, therefore clock signals have no effect on the flip flop. In other words, Q returns its last value.

When J = 0 and K = 1,

The upper NAND gate is disabled the lower NAND gate is enabled if Q is 1 therefore, the flip flop will be reset (Q = 0 , Q =1)if not already in that state.

When J = 1 and K = 0

The lower NAND gate is disabled and the upper NAND gate is enabled if Q is at 1, As a result, we will be able to set the flip flop (Q = 1, Q= 0) if not already set

When J = 1 and K = 1

If Q = 0 the lower NAND gate is disabled the upper NAND gate is enabled. This will set the flip flop and hence Q will be 1. On the other hand, if Q = 1, the lower NAND gate is enabled and the flip flop will be reset hence Q will be 0. In other words, when J and K are both high, the clock pulses cause the JK flip-flop to toggle. The truth table for the JK flip-flop is shown in Table 8.

Table 8: The truth table for the JK flip flop

Initial Conditions	Inputs (Pulsed)		Final Output
Q	S	R	Q (t + 1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

The excitation table for JK or SR flip flop is very simply derived as given in table 9.

Table 9: Excitation table for JK Flip Flop

S	R	Q
0	0	No Change
0	1	0
1	0	1
1	1	Toggle

T Flip Flop

A method of avoiding the indeterminate state found in the working of RS flip flop is to provide only one input (the T input) such, flip flop acts as a toggle switch. Toggle means to change in the previous stage i.e. switch to opposite state. It can be constructed from clocked RS flip flop by incorporating feedback from output to input as shown in Figure 5.

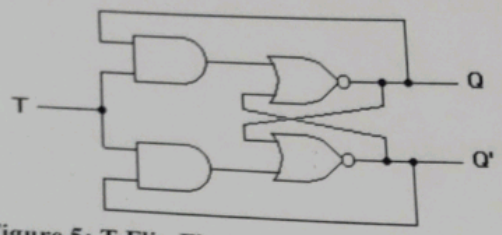


Figure 5: T Flip Flop

Such a flip-flop is also called a toggle flip-flop. In such a flip flop a train of extremely narrow triggers drives the T input each time one of these triggers, the output of the flip flop changes stage. For instance, Q equals 0 just before the trigger. Then the upper AND gate is enabled and the lower AND gate is disabled. When the trigger arrives, it results in a high S input.

This sets the Q output to 1. When the next trigger appears at point T, the lower AND gate is enabled and the trigger passes through to the R input this forces the flip flop to reset.

Since each incoming trigger is alternately changed into the set and reset inputs the flip-flop toggles. It takes two triggers to produce one cycle of the output waveform. This means the output has half the frequency of the input stated another way, a T flip flop divides the input frequency by two. Thus such a circuit is also called a divide-by-two circuit.

A disadvantage of the toggle flip flop is that the state of the flip flop after a trigger pulse has been applied is only known if the previous state is known. The truth table for a T flip flop is as given Table 7.

Table 7: Truth table for T Flip Flop

Q_n	T	$Q_n + 1$
0	0	0
0	1	1
1	0	1
1	1	0

The excitation table for T flip flop is very simply derived as shown in Table 8.

Table 8: Excitation table for T Flip Flop

T	Q

Table 8: Excitation table for T Flip Flop

T	Q
0	Q_n
1	Q_n

Generally, T flip-flop ICs are not available. It can be constructed using JK, RS, or D flip flop. Figure 6 shows the relation of the T flip flop using the JK flip flop.

Figure 6: T Flip Flop Using JK Flip Flop

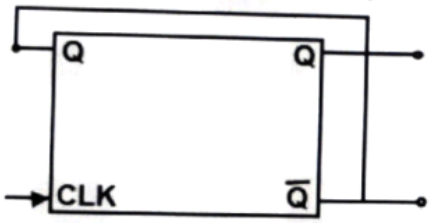


Figure 7: D-type Flip Flop connected as toggle stage

Figure 7: JK & D Flip Flop Connected as T Flip Flop

A D-type flip flop may be modified by external connection as a T-type stage as shown in Figure 7. Since the Q logic is used as D-input the opposite of the Q output is transferred into the stage of each clock pulse. Thus the stage having Q - 0 transistors Q= 1, Providing a toggle action, if the stage had Q = 1 the clock pulse would result in Q = 0 being transferred, again providing the toggle operation. The D-type flip flop connected as in Figure 6 will thus operate as a T-type stage, complementing each clock pulse.

Master Slave Flip Flop

Figure 8 shows the schematic diagram of the master-slave J-K flip flop

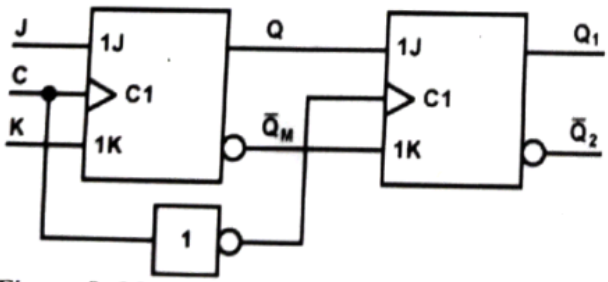


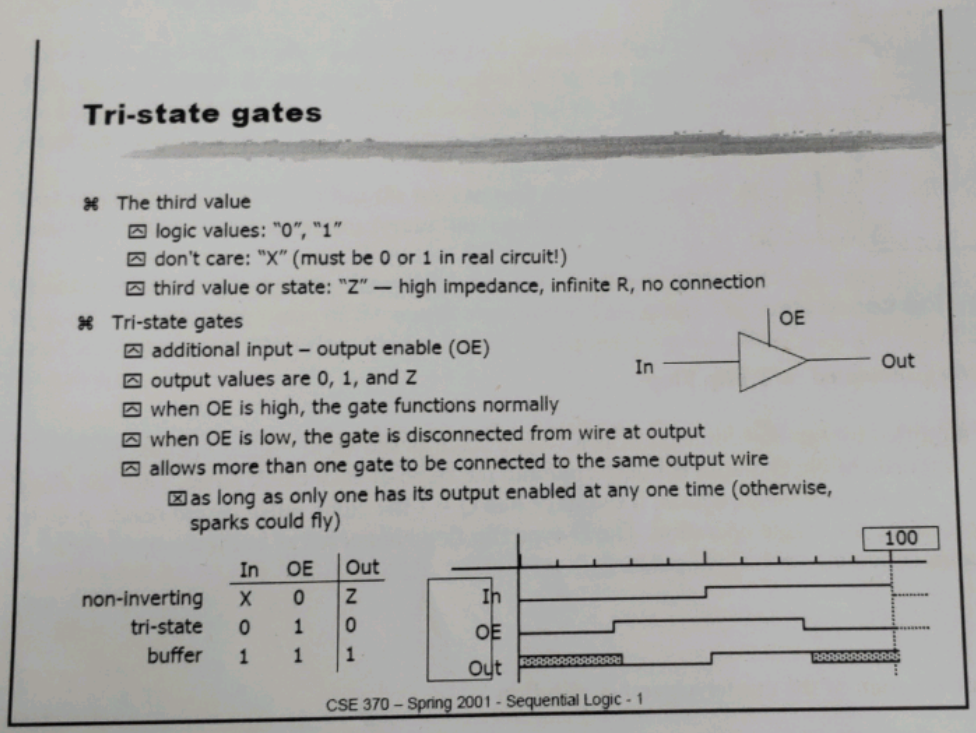
Figure 8: Master Slave JK Flip Flop

A master-slave flip-flop contains two clocked flip-flops. The first is called a master and the second is a slave. When the clock is high the master is active. The output of the master is set or reset according to the state of the

input. As the slave is in active during this period its output remains in the previous state. When the clock becomes low the output of the slave flip-flop changes because it becomes active during the low clock period. The final output of the master-slave flip-flop is the output of the slave flip-flop. So the output of master-slave flip flop is available at the end of a clock pulse.

Tri-state logic:

As you might guess, tri-state logic has three states: H, L, and disconnect (which is sometimes called *high-impedance* or *High Z* state). If an output is in the High-Z state, it will not draw current from any other outputs to force them into any state. It behaves just as if the gate has been disconnected from the bus. Any particular device that wants to write to the data lines must have its Input/Output Write (IOW) line asserted before its output leaves the High Z state. It is the core duty of the bus master to insure that only one device is *IOW enabled* at a time.



Counter:

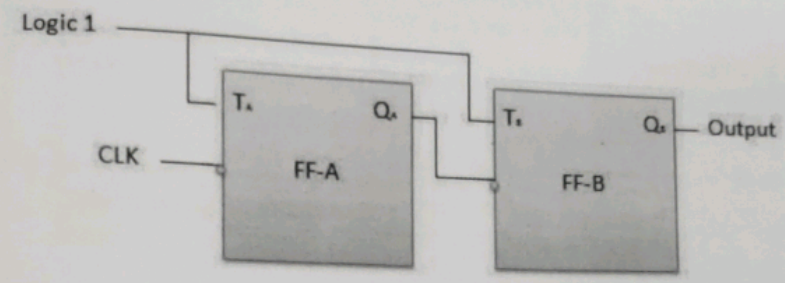
Counter is a sequential circuit. A digital circuit which is used for a counting pulses is known counter. Counter is the widest application of flip-flops. It is a group of flip-flops with a clock signal applied. Counters are of two types.

- Asynchronous or ripple counters.
- Synchronous counters.

Asynchronous or ripple counters

The logic diagram of a 2-bit ripple up counter is shown in figure. The toggle (T) flip-flop are being used. But we can use the JK flip-flop also with J and K connected permanently to logic 1. External clock is applied to the clock input of flip-flop A and Q_A output is applied to the clock input of the next flip-flop i.e. FF-B.

Logical Diagram



Operation

S.N.	Condition	Operation
1	**Initially let both the FFs be in the reset state**	Q_BQ_A = 00 initially
2	**After 1st negative clock edge**	As soon as the first negative clock edge is applied, FF-A will toggle and Q_A will be equal to 1. Q_A is connected to clock input of FF-B. Since Q_A has changed from 0 to 1, it is treated as the positive clock edge by FF-B. There is no change in Q_B because FF-B is a negative edge triggered FF. Q_BQ_A = 01 after the first clock pulse.
3	**After 2nd negative clock edge**	On the arrival of second negative clock edge, FF-A toggles again and Q_A = 0. The change in Q_A acts as a negative clock edge for FF-B. So it will also toggle, and Q_B will be 1. Q_BQ_A = 10 after the second clock pulse.
4	**After 3rd negative clock edge**	On the arrival of 3rd negative clock edge, FF-A toggles again and Q_A become 1 from 0. Since this is a positive going change, FF-B does not respond to it and remains inactive. So Q_B does not change and continues to be equal to 1. Q_BQ_A = 11 after the third clock pulse.
5	**After 4th negative clock edge**	On the arrival of 4th negative clock edge, FF-A toggles again and Q_A becomes 1 from 0. This negative change in Q_A acts as clock pulse for FF-B. Hence it toggles to change Q_B from 1 to 0.

Truth Table

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

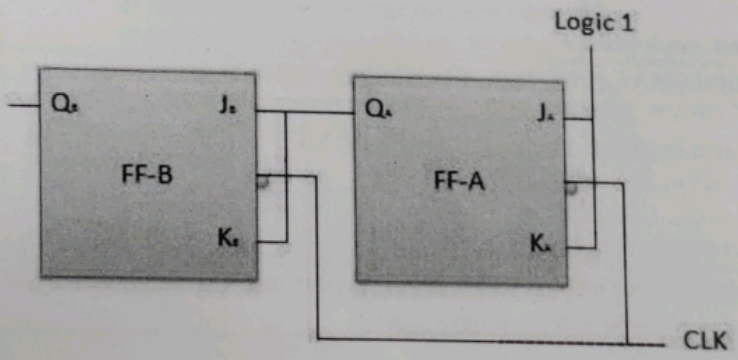
Synchronous counters

If the "clock" pulses are applied to all the flip-flops in a counter simultaneously, then such a counter is called as synchronous counter.

2-bit Synchronous up counter

The J_A and K_A inputs of FF-A are tied to logic 1. So FF-A will work as a toggle flip-flop. The J_B and K_B inputs are connected to Q_A.

Logical Diagram



Operation

S.N.	Condition	Operation
1	**Initially let both the FFs be in the reset state**	Q_BQ_A = 00 initially.
2	**After 1st negative clock edge**	As soon as the first negative clock edge is applied, FF-A will toggle and Q_A will change from 0 to 1. But at the instant of application of negative clock edge, Q_A , $J_B = K_B = 0$. Hence

		FF-B will not change its state. So Q_B will remain 0. Q_BQ_A = 01 after the first clock pulse.
3	**After 2nd negative clock edge**	On the arrival of second negative clock edge, FF-A toggles again and Q_A changes from 1 to 0. But at this instant Q_A was 1. So $J_B = K_B = 1$ and FF-B will toggle. Hence Q_B changes from 0 to 1. Q_BQ_A = 10 after the second clock pulse.
4	**After 3rd negative clock edge**	On application of the third falling clock edge, FF-A will toggle from 0 to 1 but there is no change of state for FF-B. Q_BQ_A = 11 after the third clock pulse.
5	**After 4th negative clock edge**	On application of the next clock pulse, Q_A will change from 1 to 0 as Q_B will also change from 1 to 0. Q_BQ_A = 00 after the fourth clock pulse.

Classification of counters

Depending on the way in which the counting progresses, the synchronous or asynchronous counters are classified as follows –

- Up counters
- Down counters
- Up/Down counters

UP/DOWN Counter

Up counter and down counter is combined together to obtain an UP/DOWN counter. A mode control (M) input is also provided to select either up or down mode. A combinational circuit is required to be designed and used between each pair of flip-flop in order to achieve the up/down operation.

- Type of up/down counters
- UP/DOWN ripple counters
- UP/DOWN synchronous counter

UP/DOWN Ripple Counters

In the UP/DOWN ripple counter all the FFs operate in the toggle mode. So either T flip-flops or JK flip-flops are to be used. The LSB flip-flop receives clock directly. But the clock to every other FF is obtained from (Q = Q bar) output of the previous FF.

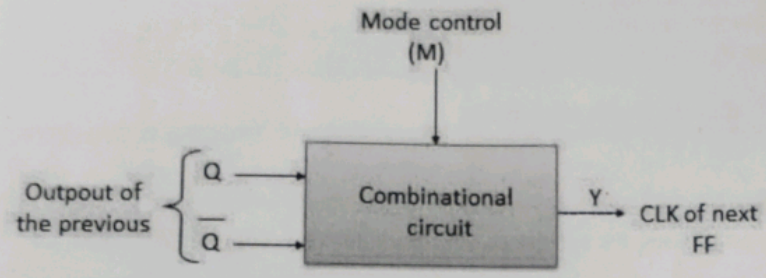
- **UP counting mode (M=0)** – The Q output of the preceding FF is connected to the clock of the next stage if up counting is to be achieved. For this mode, the mode select input M is at logic 0 (M=0).
- **DOWN counting mode (M=1)** – If M = 1, then the Q bar output of the preceding FF is connected to the next FF. This will operate the counter in the counting mode.

Example

3-bit binary up/down ripple counter.

- 3-bit – hence three FFs are required.
- UP/DOWN – So a mode control input is essential.
- For a ripple up counter, the Q output of preceding FF is connected to the clock input of the next one.
- For a ripple up counter, the Q output of preceding FF is connected to the clock input of the next one.
- For a ripple down counter, the Q bar output of preceding FF is connected to the clock input of the next one.
- Let the selection of Q and Q bar output of the preceding FF be controlled by the mode control input M such that, If M = 0, UP counting. So connect Q to CLK. If M = 1, DOWN counting. So connect Q bar to CLK.

Block Diagram



Truth Table

Inputs			Outputs	
M	Q	$\overline{Q}$	Y	
0	0	0	0	Y = Q for up counter
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	0	$Y = \overline{Q}$ for up counter
1	0	1	1	
1	1	0	0	
1	1	1	1	

Operation

S.N.	Condition	Operation
1	**Case 1 – With M = 0 (Up counting mode)**	If M = 0 and M bar = 1, then the AND gates 1 and 3 in fig. will be enabled whereas the AND gates 2 and 4 will be disabled. Hence Q_A gets connected to the clock input of FF-B and Q_B gets connected to the clock input of FF-C. These connections are same as those for the normal up counter. Thus with M = 0 the circuit work as an up counter.

2	**Case 2: With M = 1 (Down counting mode)**	If M = 1, then AND gates 2 and 4 in fig. are enabled whereas the AND gates 1 and 3 are disabled. Hence Q_A bar gets connected to the clock input of FF-B and Q_B bar gets connected to the clock input of FF-C. These connections will produce a down counter. Thus with M = 1 the circuit works as a down counter.

Modulus Counter (MOD-N Counter)

The 2-bit ripple counter is called as MOD-4 counter and 3-bit ripple counter is called as MOD-8 counter. So in general, an n-bit ripple counter is called as modulo-N counter. Where, MOD number = 2^n.

Type of modulus

- 2-bit up or down (MOD-4)
- 3-bit up or down (MOD-8)
- 4-bit up or down (MOD-16)

Application of counters

- Frequency counters
- Digital clock
- Time measurement
- A to D converter
- Frequency divider circuits
- Digital triangular wave generator.

Shift Registers

We know that one flip-flop can store one-bit of information. In order to store multiple bits of information, we require multiple flip-flops. The group of flip-flops, which are used to hold *store* the binary data is known as **register**.

If the register is capable of shifting bits either towards right hand side or towards left hand side is known as **shift register**. An 'N' bit shift register contains 'N' flip-flops. Following are the four types of shift registers based on applying inputs and accessing of outputs.

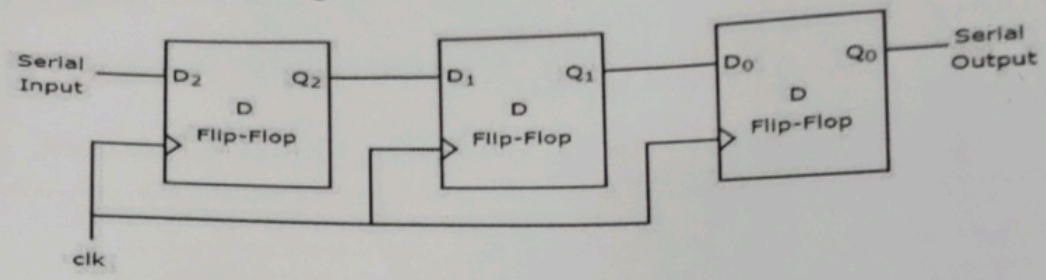
- Serial In – Serial Out shift register
- Serial In – Parallel Out shift register
- Parallel In – Serial Out shift register
- Parallel In – Parallel Out shift register

Serial In – Serial Out *SISO*

Shift Register

The shift register, which allows serial input and produces serial output is known as Serial In – Serial Out ***SISO***

shift register. The **block diagram** of 3-bit SISO shift register is shown in the following figure.



This block diagram consists of three D flip-flops, which are **cascaded**. That means, output of one D flip-flop is connected as the input of next D flip-flop. All these flip-flops are synchronous with each other since, the same clock signal is applied to each one.

In this shift register, we can send the bits serially from the input of left most D flip-flop. Hence, this input is also called as **serial input**. For every positive edge triggering of clock signal, the data shifts from one stage to the next. So, we can receive the bits serially from the output of right most D flip-flop. Hence, this output is also called as **serial output**.

Example

Let us see the working of 3-bit SISO shift register by sending the binary information **"011"** from LSB to MSB serially at the input.

Assume, initial status of the D flip-flops from leftmost to rightmost is $Q2Q1Q0=000$

. We can understand the **working of 3-bit SISO shift register** from the following table.

No of positive edge of Clock	Serial Input	Q_2	Q_1	Q_0
0	-	0	0	0
1	1*LSB*			

1 0 0

2 1 1 1 0

3 0*MSB*

0 1 1*LSB*

4 - - 0 1

5 - - - 0*MSB*

The initial status of the D flip-flops in the absence of clock signal is $Q2Q1Q0=000$

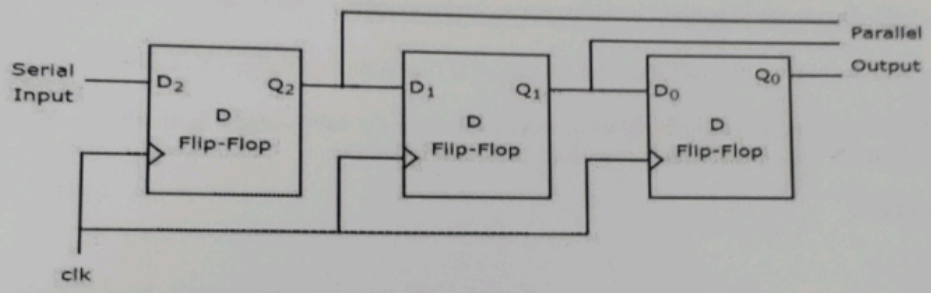
. Here, the serial output is coming from $Q0$. So, the LSB 1 is received at 3rd positive edge of clock and the MSB 0 is received at 5th positive edge of clock.

Therefore, the 3-bit SISO shift register requires five clock pulses in order to produce the valid output. Similarly, the **N-bit SISO shift register** requires **2N-1** clock pulses in order to shift 'N' bit information.

Serial In - Parallel Out *SIPO*

Shift Register

The shift register, which allows serial input and produces parallel output is known as Serial In – Parallel Out ***SIPO*** shift register. The **block diagram** of 3-bit SIPO shift register is shown in the following figure.



This circuit consists of three D flip-flops, which are cascaded. That means, output of one D flip-flop is connected as the input of next D flip-flop. All these flip-flops are synchronous with each other since, the same clock signal is applied to each one.

In this shift register, we can send the bits serially from the input of left most D flip-flop. Hence, this input is also called as **serial input**. For every positive edge triggering of clock signal, the data shifts from one stage to the next. In this case, we can access the outputs of each D flip-flop in parallel. So, we will get **parallel outputs** from this shift register.

Example

Let us see the working of 3-bit SIPO shift register by sending the binary information **"011"** from LSB to MSB serially at the input.

Assume, initial status of the D flip-flops from leftmost to rightmost is $Q2Q1Q0$=000

. Here, $Q2$ & $Q0$

are MSB & LSB respectively. We can understand the **working of 3-bit SIPO shift register** from the following table.

No of positive edge of Clock	Serial Input	Q_2*MSB*

Q_1 Q_0*LSB*

0 - 000

1 1*LSB*

1 00

2 1 110

3 0*MSB*

011

The initial status of the D flip-flops in the absence of clock signal is $Q2Q1Q0=000$

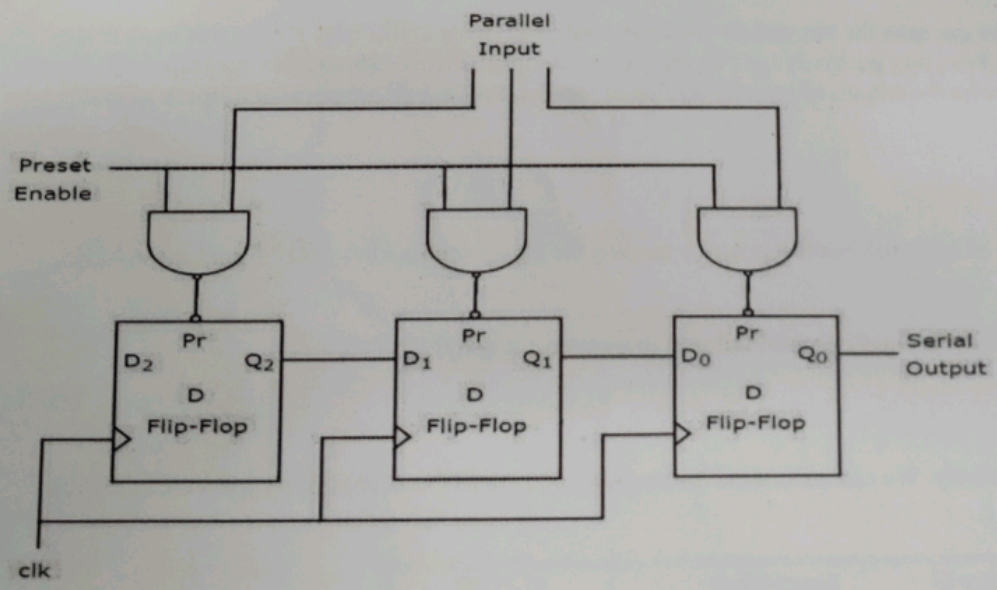
. The binary information **"011"** is obtained in parallel at the outputs of D flip-flops for third positive edge of clock.

So, the 3-bit SIPO shift register requires three clock pulses in order to produce the valid output. Similarly, the **N-bit SIPO shift register** requires **N** clock pulses in order to shift 'N' bit information.

Parallel In – Serial Out *PISO*

Shift Register

The shift register, which allows parallel input and produces serial output is known as Parallel In – Serial Out ***PISO*** shift register. The **block diagram** of 3-bit PISO shift register is shown in the following figure.



This circuit consists of three D flip-flops, which are cascaded. That means, output of one D flip-flop is connected as the input of next D flip-flop. All these flip-flops are synchronous with each other since, the same clock signal is applied to each one.

In this shift register, we can apply the **parallel inputs** to each D flip-flop by making Preset Enable to 1. For every positive edge triggering of clock signal, the data shifts from one stage to the next. So, we will get the **serial output** from the right most D flip-flop.

Example

Let us see the working of 3-bit PISO shift register by applying the binary information **"011"** in parallel through preset inputs.

Since the preset inputs are applied before positive edge of Clock, the initial status of the D flip-flops from leftmost to rightmost will be $Q2Q1Q0=011$

. We can understand the **working of 3-bit PISO shift register** from the following table.

No of positive edge of Clock	Q_2	Q_1	Q_0
0	0	1	1*LSB*
1 - 0	1		
2 - -	0*LSB*		

Here, the serial output is coming from $Q0$

. So, the LSB 1 is received before applying positive edge of clock and the MSB 0

is received at 2nd positive edge of clock.

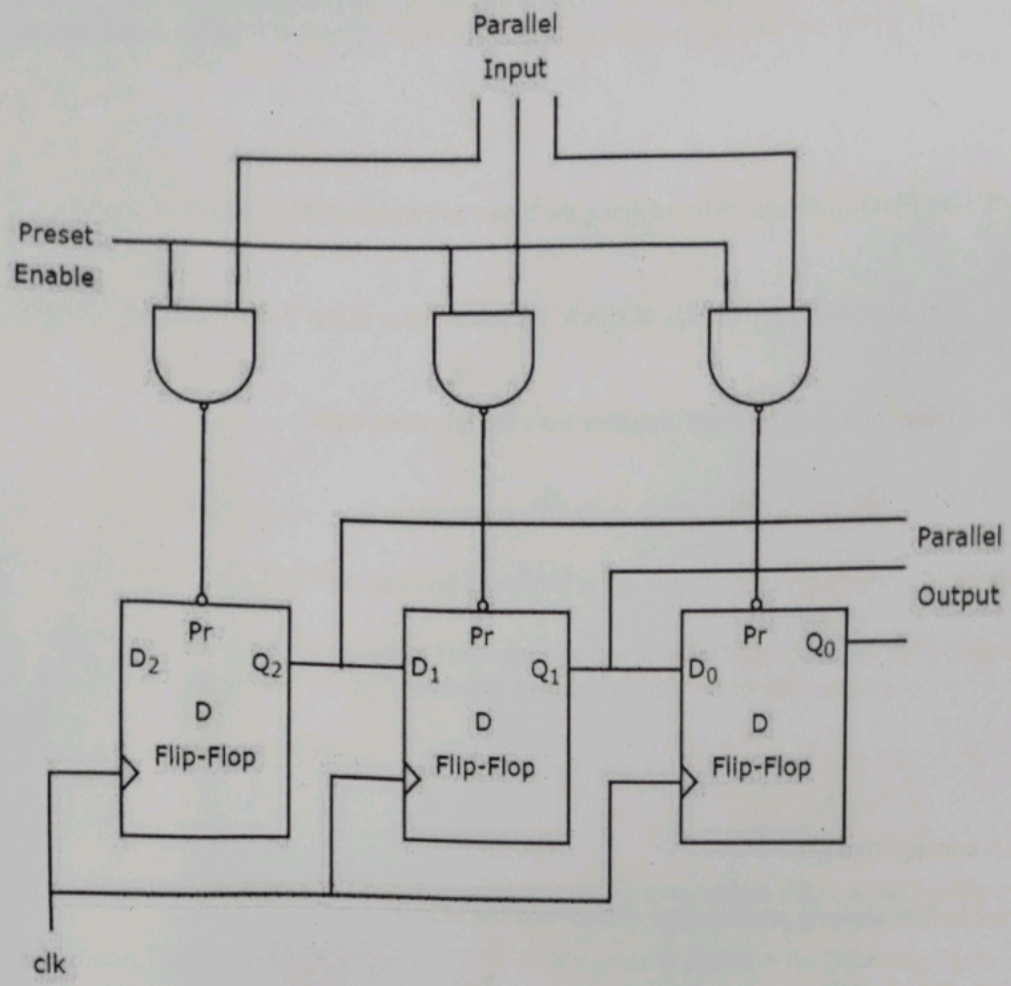
Therefore, the 3-bit PISO shift register requires two clock pulses in order to produce the valid output. Similarly, the **N-bit PISO shift register** requires **N-1** clock pulses in order to shift 'N' bit information.

Parallel In - Parallel Out *PIPO*

Shift Register

The shift register, which allows parallel input and produces parallel output is known as Parallel In – Parallel Out ***PIPO***

shift register. The **block diagram** of 3-bit PIPO shift register is shown in the following figure.



This circuit consists of three D flip-flops, which are cascaded. That means, output of one D flip-flop is connected as the input of next D flip-flop. All these flip-flops are synchronous with each other since, the same clock signal is applied to each one.

In this shift register, we can apply the **parallel inputs** to each D flip-flop by making Preset Enable to 1. We can apply the parallel inputs through preset or clear. These two are asynchronous inputs. That means, the flip-flops produce the corresponding outputs, based on the values of asynchronous inputs. In this case, the effect of outputs is independent of clock transition. So, we will get the **parallel outputs** from each D flip-flop.

Example

Let us see the working of 3-bit PIPO shift register by applying the binary information **"011"** in parallel through preset inputs.

Since the preset inputs are applied before positive edge of Clock, the initial status of the D flip-flops from leftmost to rightmost will be $Q2Q1Q0$=011

. So, the binary information **"011"** is obtained in parallel at the outputs of D flip-flops before applying positive edge of clock.

Therefore, the 3-bit PIPO shift register requires zero clock pulses in order to produce the valid output. Similarly, the **N-bit PIPO shift register** doesn't require any clock pulse in order to shift 'N' bit information.