- **General concepts**
- **Operation of RS, JK, D and T Flip-Flops**
- **Synchronous and Asynchronous Counters**
- **Ring Counters**
- **Shift Registers**

Introduction

As mentioned earlier, in previous chapters, the logic circuits are classified into two major groups

1. Combinational Logic Circuits.
2. Sequential Logic Circuits.

Logic gates are in the category of combinational logic circuits. Sequential circuits involve timing, counting and memory devices. The basic building block of sequential logic circuits is the **Flip-flop (FF)** like logic gate, the building block of combinational logic circuits. There are several types of Flip-Flop circuit and the control inputs vary with each type. FF are wired to form counters, shift registers and various memory devices. They are capable of storing binary information.

A flip-flop has two stable states 0 or 1. When it is said to one of these states, it remains in that state until the application of a control signal causes it to FLIP or 'FLOP'to the other state. It is a bistable multivibrator circuit. It is a digital circuit has two outputs Q and $\overline{Q}$ which are always in opposite state if Q is 1 then `Q is 0 the flip-flop is said to be SET, ON or PRESET. If Q is 0 then `Q is 1 and the flip-flop is said to be RESET, OFF or CLEARED. The logic levels on the flip-flop inputs will determine the state of the Q and `Q outputs according to the truth-table for the type of flip-flop.

Unlike the gates, the flip-flops can in some states maintain its output state (ON or OFF) after the input signals which produce the output. Thus the flip-flop can store a bit of information or one place of a larger binary number.

The basic types of Flip-Flops are

- J-K Flip-Flop
- R-S Flip-Flop (R-S stand for Reset-set)
- D-Flip-Flop (D stand for Delay)
- T-Flip-Flop (T stand for Toggle)

The block diagram of a sequential circuit is shown in Figure 5.1. The role of the combinational circuit is to accept digital signals from external inputs and from outputs of memory elements. Then it generates signals for external outputs and for inputs to memory elements. The output of a sequential circuit is a function of the time sequence of inputs and the internal states.

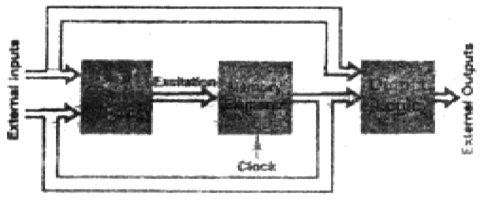


Figura 5.1 Block Diagram of a Sequential Circuit

Depending on timing of their signals, sequential circuits are classified in two main categories.

- Synchronous
- Asynchronous

A sequential circuit whose performance depends upon the sequence in which the input signals change is referred to as an asynchronous sequential circuits. The commonly used memory elements in these circuits are time delay devices. A sequential circuit whose operation can be defined from the knowledge of its signal at discrete instance of time is referred to as synchronous sequential circuit. In these systems, the memory elements are affected only at discrete instance of time. Synchronous circuits are also known as *Clocked-sequential Circuits.* The synchronisation is achieved by a timing device known as a S*ystem-clock,* which generates a periodic train of clock pulses as shown in Figure 5.2.

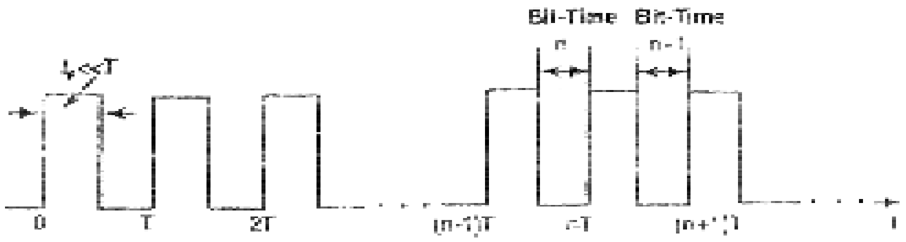


Figure 5.2 A Train of Pulses

- **Clock Flip-Flop**

To push the inputs to the outputs at any desired time and then to hold the outputs for the desired time we use clock and flip flop, now the flip-flop has three inputs e.g. S,R and Clock sometime it is also called **R-S-T Flip Flop.**

- **Clock**

A pulse generator within a digital system to which all operations are synchronized is called clock. The square wave in Figure 5.3 is a typical clock waveform used in a digital system and it is not necessary that the clock to be perfectly a symmetrical square wave as shown in Figure 5.3. It could simply be a series of positive (or negative) pulses but the main requirement is that the clock should be perfectly periodic.

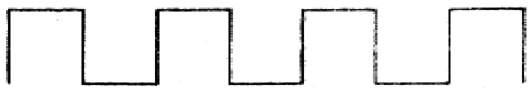


Figure 5.3 Ideal Clock Waveform

For a clock to be ideal it is necessary that

1. The clock level remains absolutely stable, when the clock is HIGH, its level must hold a steady value of +5V, and when it is LOW the level must remain at 0V.
2. The second necessity for an ideal clock is that transition time for the clock should be zero, i.e. the change occurring from 0V to +5V should take no time and vice versa but this is not possible in real practice, the waveform take sometime to make the change.

The practical square wave looks like as shown in Figure 5.4.

The time required for transition from LOW to HIGH is defined as **Rise Time (t_r)**. The time requires for transition from HIGH to LOW is defined as the **Fall Time (t_r)**.

- **Rise Time (t_r)**

  Time required for the waveform to travel from 10% or its initial value to 90% of the final value is known as rise time.

- **Fall Time (t_r)**

  Time required for the waveform to travel from 90%of its final value to 10% of the initial value is known as fall time.

Some other definitions that will come across in the following topics are:

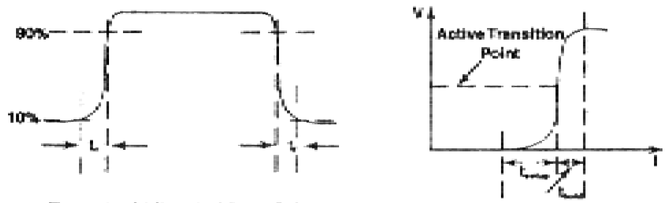

90%
10%
t_r
t_f
V
Active Transition Point
t

Figure 5.4 (a) Practical Clock Pulse (b)

Inputs can change upto this point
Inputs can change from this point

- **Propagation Delay Time**

  The amount of time taken by the gate or flip-flop to cause a change at the output after input has changed. It is around 10 nano seconds for LSTTL logic ICs.

- **Set UP Time**

  It is minimum time over which the data bit must be present before the clock edge hits

- **Hold Time**

  It is the minimum amount of time over which data bit must be present after the clock edge arrives.

OPERATION OF R S, JK, D AND T FLIP-FLOP

Clocked S R Flip-Flop

The addition of two AND gates at the S-R inputs are shown in Figure 5.5 (a). When the CLK input is low the AND gate outputs will be LOW and changes in neither S or R will have be transmitted to the outputs. When the ENABLE inputs goes LOW, the output will retain the information that was present on the input when HIGH to LOW transition takes place. The Truth Table 5.1 represents its detailed working principle.

CLK	*S*	*R*	*Q*
0	0	0	No change
0	0	1	No change
0	1	0	No change
0	1	1	No change
1	0	0	No change
1	0	1	0
1	1	0	1
1	1	1	? (Forbidden)

Table 5.1: Truth Table for Clocked SR Flip-Flop.

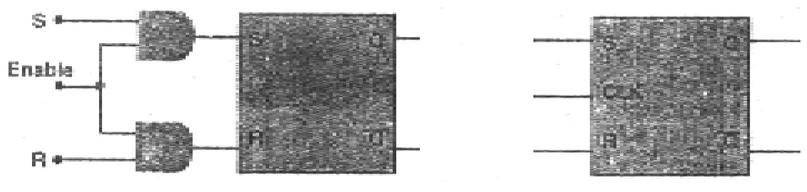


Figure 5.5 (a) Logic Diagram (b) Symbol

Clocked S-R Flip Flop using NAND Gate Only

Before discussing the condition let us first know the meaning of Sn, Rn and Qn+1:Sn and Rn means inputs applied during nth clock cycle and Qn means output during Nth clock cycle. Whereas, Qn+1 means output during (n+1)th clock cycle (as depicted in Figures 5.6 (a) and (b)).

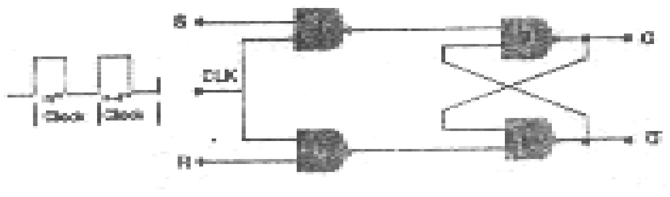


Figure 5.6 (a) Clocked NAND S-R Flip Flop

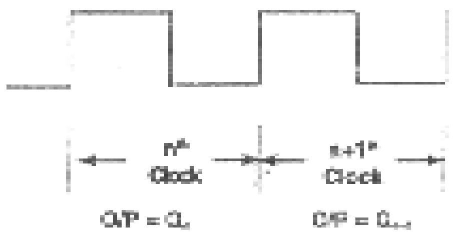
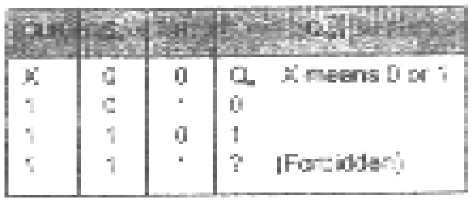


Figure 5.6 (b)



[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
X	0	0	Q_n	X means 0 or 1
1	0	1	0	
1	1	0	1	
1	1	1	?	(Forbidden)

Truth Table 5.2

Condition 1: $S_n=0$, $R_n=0$

Suppose when nth cycle was commencing at that time output was Qn=1 and when the next clock cycle i.e. (n+1)th cycle comes, the output remains the same as it was during the (n+1)th cycle is $Q_n+1=1$ (No change).

If the output during nth cycle was 0 i.e. $Q_n=0$ then it will remain 0 during (n+1)th cycle, so for $S_n = R_n=0$, output is not changed.

Condition 2: $S_n=0$, $R_n=1$

Suppose when nth cycle was commencing, at that time the output was say Q=1 and we apply the output $S_n=0$, $R_n=1$ and when (n+1)th cycle comes, the output becomes $Q_n+1=0$

Condition 3: S_n=1, R_n=0

Now when S_n=1 and R_n=0 is applied during the low state of nth cycle then whatever be the output during nth cycle the output become 1 during (n+1)th cycle i.e. Q_n+1=1

Condition 4: S_n=1, R_n=1

As discussed earlier, this condition is forbidden as both the outputs try to become 1, which violates the definition of flip-flop, that one output is the complement of the other. The Truth Table 5.2 explains its working.

Edge Triggered S-R Flip Flop using NAND Gates

The Figure 5.7(a) shows as R-S circuit at the input of a S-R flip flop, which is acting as a differentiator whose R-C time constant is kept very low, because of this the capacitor charged fully when the clock goes high, this exponential charging produces a narrow positive spike across the resistor, the trailing edge of the pulse result in a narrow negative spike. This narrow positive spike enables the NAND gate for an instant and the negative spike has no role. This type of edge triggering allows to make the transition at a fixed time interval.

If the triggering is occurring at the positive edge it is called positive edge triggered S R flip-flop and is represented by an upward arrow in the truth table (Table5.3) and by a notch ">" in the symbol (Figure 5.7 (b).

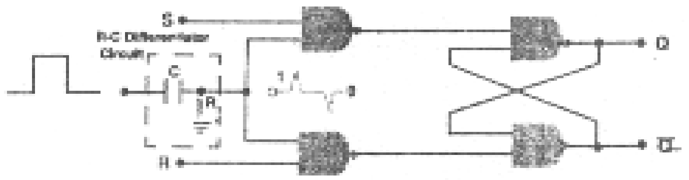
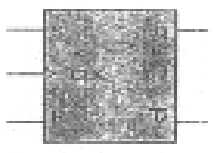
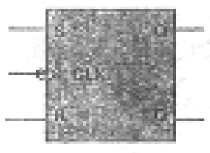


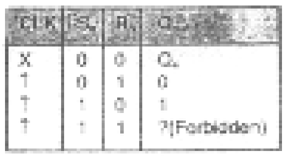
Figure 5.7 (a) Edge Triggered S R Flip Flop with RC Differentiator Circuit



(b) Positive Edge Triggered Flip Flop



(c) Negative Edge Triggered Flip Flop



CLK	S_n	R_n	Q_{n+1}
X	0	0	Q_n
↑	0	1	0
↑	1	0	1
↑	1	1	?(Forbidden)

Truth Table 5.3

If the triggering is occurring at the negative edge pulse, then it is called negative edge triggered S R flip-flop and is to be shown by a downward arrow in the truth table and by a bubble “O” at the clock input as shown in Figure 5.7 ©.

J K FLIP FLOP

J K Flip Flop from S R Flip Flop

J K Flip Flop is the most widely used flip flop and it eliminates the uncertainty in the fourth row of the truth table of the S R flip flop when S=R condition is applied. One way of constructing J K flip flop is by using S R flip flop and AND gates as shown in Figure 5.8(c). The upper AND gates has two inputs J and the other one connection to the ‘Q output.

When input are J=K=1 then output is complemented on the application of clock pulse i.e. if before application of clock pulse the output was 0 then after application of the clock the output becomes 1. The working principle is explained in the truth Table 5.4 and 5.5.

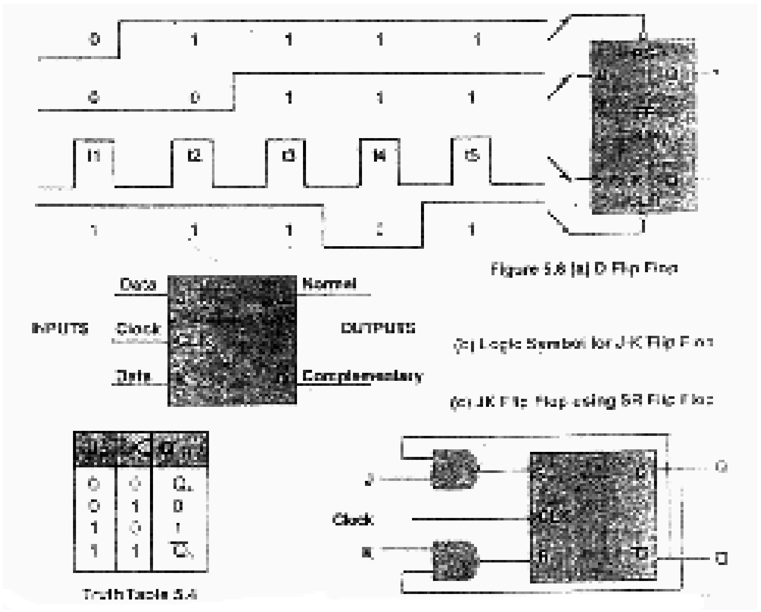


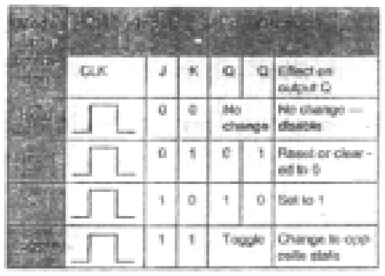
Figure 5.8 (a) D Flip Flop

(b) Logic Symbol for J-K Flip Flop

(c) JK Flip Flop using SR Flip Flop

J	K	Q
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

Truth Table 5.4



CLK	J	K	Q	Q	Effect on output Q
⎍	0	0	No change		No change — disable
⎍	0	1	0	1	Reset or clear - ed to 0
⎍	1	0	1	0	Set to 1
⎍	1	1	Toggle		Change to opp osite state

Truth Table 5.5 J-K Flip Flop

J K Flip Flop using NAND Gates

The same function can be realized using NAND gates also and is shown below in Figure 5.9 (a).

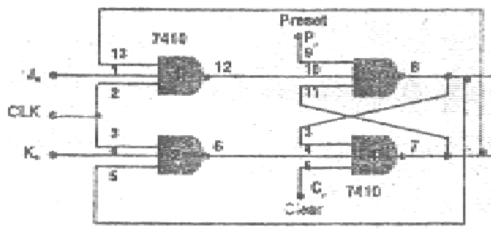


Figure 5.9 (a) J K Flip Flop using NAND Gates only

In the Figure, two new inputs are shown they are Preset (P_r) and Clear (C_r). These inputs are used for assigning initial state for the flip-flop. Suppose it is necessary to have clear outputs i.e. Q=0 when clock goes to zero, this can be done by giving C_r=0 and Pr=1.

Similarly, Q can be set to 1 by choosing P_r=0 and C_r=1 when clock is at zero state. It should be noted that these inputs should only be given when the clock is at zero and once the state of flip flop is established it is necessary to have P_r=C_r=1 before the next clock pulse arrives and at no time P_r=C_r=0 should be given as it leads to uncertainty in the state of flip flop.

The first three conditions in the truth table of J K flip flop is same as S R flip flop but the last condition when J=K=1 is the subject of interest.

Let Q_n=0 and J_n and K_n=1, when the clock is made HIGH, then the output changes to Q_{n+1}=$Q_{n=1}$ this change at the output takes place after a time period corresponding to the propagation delay through two NAND GATES, which is around 20-25 n sec for each gate for TTL logic, therefore, during the presence of clock the output again change to 0 and it go on changing from 0 to 1 and 1 to 0 until the clock is HIGH therefore, it is difficult to predict the output at the end of the clock as the 1 output state is dependent on the pulse width. This is known as **RACE AROUND CONDITION.** The Figure 5.9 (b) shows the output toggles four times for the given width of the clock pulse and the final output is 0 where as in the second pulse of Figure 5.9(b) for given clock pulse T_2 the output toggles five times and the final output is 1 this is an ambiguous situation where outputs are unpredictable, thereby, giving different outputs every time.

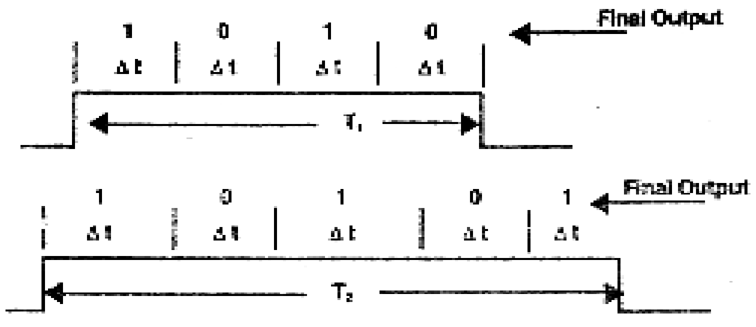


Figure 5.9 (b) Train of Pulses

One easy way to eliminate the race around condition is by making the time period of the clock less than the propagation delay time of the gate but practically this is not possible because the propagation delay is very small, the problem is solved by master slave J K flip flop. (Figure 5.10 (a) and (b)).

Master Slave Flip Flop provides a way to avoid race around problem which is caused because of the changing inputs while the clock is HIGH as the inputs are connected to the output, the racing problem can be checked some arrangement is made so that inputs are disabled while outputs are changing, this can be easily achieved by dividing the flip flop into two parts one is named **MASTER** and the other **SLAVE** as it follows the master (Figure 5.10 (b)).

The MASTER flip flop is positive edge triggered J K flip flop and the SLAVE is negative edge triggered flip flop, therefore, when clock is HIGH the MASTER flip flop is enabled and passes the inputs to its output, during this time the SLAVE flip flop is disabled and do not function. When the clock goes LOW during the time the MASTER is disabled and SLAVE flip flop is enabled and passes the outputs of the MASTER which were produced when the clock was HIGH to its output.

Let us study the last condition of M/S JK flip-flop which is the subject of interest for us when $J_n=K_n=1$, let us assume the output $Q_n=0$, the various conditions are shown n Figure 5.10 (b).

When the clock is HIGH then the output of MASTER flip flop toggles from 0 to 1, i.e. Q=1 now the inputs of the SLAVE flip flop becomes 1 and 0. When the clock goes LOW these inputs are on the output thus changing the output from 0 to 1 i.e. Q_{n+1}=1 which is complement of the previous output this known as toggling. When the SLAVE flip flop output is toggling at the time the MASTER flip flop is disabled and accepts no change at its input thus avoiding the problem of racing. The Figure 5.10(c) shows the M/S flip-flop using NAND gates only.

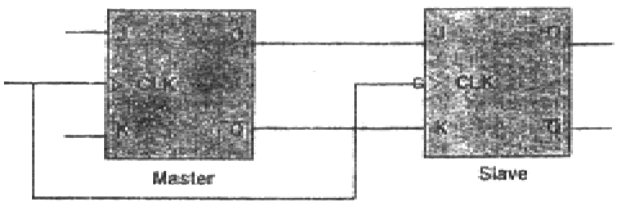


Figure 5.10 (a) Master and Slave Flip Flop

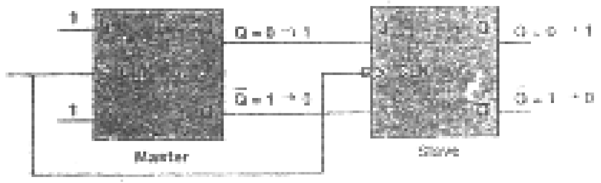


Figure 5.10 (b)

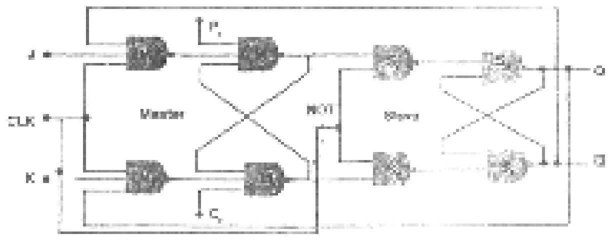


Figure 5.10 (c) Master Slave using NAND Gates

Edge Triggered M/S JK Flip Flop

Now the question comes in the mind that what is the advantage of edge triggering over level triggering. The reason for that is the output of level triggered flip flop can change at any instant when the clock is HIGH, one flip flop may change after DT1 time and the other at some DT_2 time so the exact time when the output appears is uncertain. Therefore, if one wants to cascade two or more flip flops together it will not work satisfactorily, for that reason edge triggered flip flop is preferred in which flip flop is made to trigger only at positive or negative edge of the pulse thus ascertaining the time of toggling of the flip flop. The Table 5.6 explains its working details.

CLK	J_n	K_n	Q_{n+1}
X	0	0	Q_n
↓	0	1	0
↓	1	0	1
↓	1	1	$\overline{Q}_n$ **(Toggle)**

Table 5.6: Truth Table of Edge Triggered M/S J K Flip-Flop

The Pin Out of IC 7476 is shown here, which is a negative edge triggered dual M/S JK flip flop (Figure 5.11).

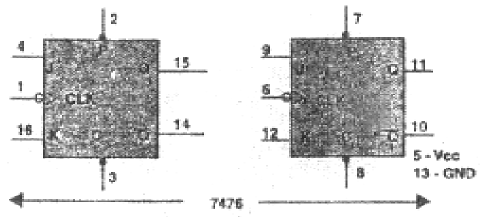


Figure 5.11 Pin Out of 7476 IC

D Flip Flop (DFF)

It is essentially a delay flip flop and it is so called because the bit at the input is transferred at the output of the flip flop when the next clock pulse comes. The D flip flop can be constructed using either SR flip flop of JK flip flop. If we look at the middle two conditions of these flip flops we find that the inputs are dissimilar if one is 0 then the other is 1 and when clock is applied the dissimilar inputs are passed on to the output. This can be done here by placing an inverter in between the two inputs of the AND gate as shown in Figure 5.12 (a).

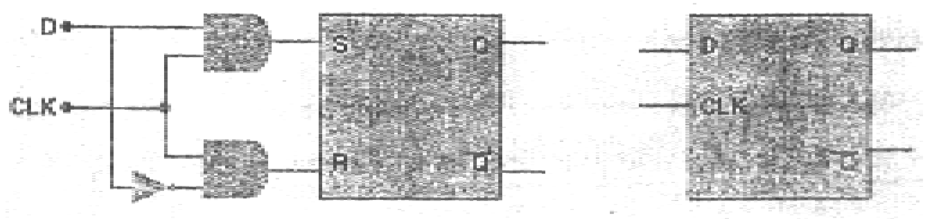


Figure 5.12 (a) D Flip Flop Circuit (b) Symbol

Let us analyses the working of the flip flop

Condition 1: D=0 (0 or 1) and CLK=0

When the clock is zero and the input is either 0 or 1, the input of the AND gates are disabled thus giving 0-0 to the input of say SR flip flop and for that condition the output is non-change condition.

Condition 2: D=0 and CLK=1

For this condition the upper AND gate is disabled and the lower one enabled thus giving say S=0 and R=1, for this condition of SR flip flop the output is 0.

Condition 3: D=1 and CLK=1

Here the upper AND gate is enabled and the lower one disabled thus producing S=1 and R=0, for this condition of SR flip flop the output is 1.

Thus we see that the output flows the input on the application of the clock pulse. All these conditions are tabulated in Table 5.7.

CLK	**K_n**	**Q_{n+1}**
0	0	Q_n
0	1	0
1	0	1

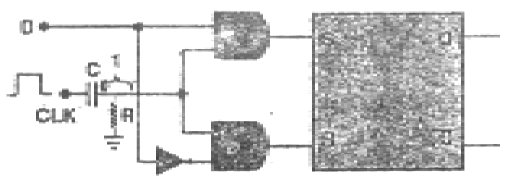
Table 5.7: Truth table for DFF

Edge Triggered D Flip Flop

The circuit symbol and the truth table of edge triggered D flip flop is shown in Figure 5.13 and Table 5.8. D flip flop finds its application in making registers.

CLK	**D**	**Q_{n+1}**
0	X	Q_n
↑	1	1

Table 5.8: Truth Table for Edge Triggered D Flip-Flop



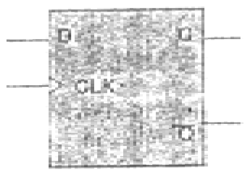


Figure 5.13 (a) Edge Triggered Flip Flop Circuit (b) Symbol

5.1.1 T Flip Flop (TFF)

Toggle flip flop is an extension of JK flip flop. When both J and K inputs of the flip flop is tied or shorted, it become T flip flop. The flip flop to work as toggle flip flop its input is at 1 and when the clock pulses are applied, the output toggles. The Figure 5.14 shows the symbol and truth table of positive edge triggered T flip flop. T flip flop is used in making counters.

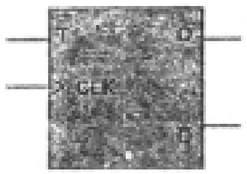


Figure 5.14 Symbol of Positive Edge Triggered T Flip Flop

CLK	**K_n**	**Q_{n+1}**
0	0	Q_n
0	1	0
1	0	1

Table 5.9: Truth Table for Edge Triggered T Flip-Flop

SYNCHRONOUS AND ASYNCHRONOUS COUNTERS

Synchronous Counter

A synchronous parallel or clocked counter is one in which all stages are triggered simultaneously. The resulting action of each stage depends on the getting inputs (synchronous-inputs) of each respective stage. This type of counter is faster than ripple or asynchronous counter since higher order stages don't have to wait for lower order changes to occur.

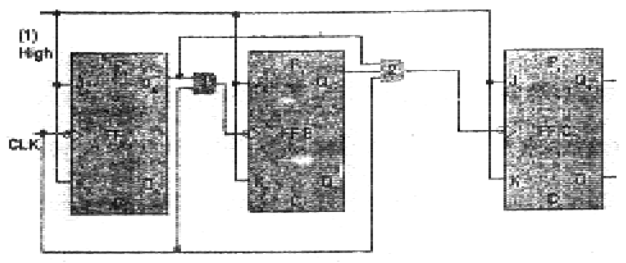


Figure 5.15 Synchronous Up Counter

The circuit of a three bit synchronous Up-counter is shown in the Figure 5.15. in the circuit, JK flip flops are made to work as T flip flop. It can be seen that all the flip flops are getting the clock at the same instant, the FFA gets direct clock and, FFB and FFC through AND gates. The AND gate at the CLK input of FFB passes every second clock to FFB and every fourth clock to FFC and so on. The circuit of synchronous counter can be constructed using T FF.

Working Principle (Table 5.10):

1. Before applying the CLK pulses all FF's are RESET.
2. When first clock pulse comes at the negative edge of the negative edge of the clock FF A toggles from 0 to 1 and the state of FF B and FF C remains unchanged as disabled AND gates do not pass the clock.

   $Q_C\, Q_B\, Q_A$=001
3. After first clock pulse the 1st AND gate is enabled, therefore, when second clock arrives then
4. Simultaneously FFA and FF B toggles. The outputs at the end of second pulse is:

   $Q_C\, Q_B\, Q_A$=010
5. Because Q_A =0, the upper pin of 1st AND disabled and due to Q_B one of the pin of 2nd AND gate is high but, its one pin is also connected to Q_A output which disables the 2nd AND gate. When third clock pulse hits only FF A responds and toggles from to 0 to 1. Thus the outputs at the end of third clock pulse is:

   $Q_C\, Q_B\, Q_A$=011

6. After third clock passes, 1st and 2nd AND gates are enabled and when fourth clock pulse arrives it passes to all FF's which results the output:

$Q_C\ Q_B\ Q_A$=100

7. Like this the count advances, at the end of the 7th clock pulse the count sequence is $Q_C\ Q_B\ Q_A$=111 and 8th clock all the FF's resets to 000 and the cycle repeats.

CLK	Q_C	Q_B	Q_A
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
8	0	0	0

Table 5.10: Truth Table for Synchronous Up Counter

Down Counter

Here minor changes are made in the UP-COUNTER circuit to make the DOWN COUNTER. The first AND gate is connected to 1_A and the second AND gate is connected to 1_B and outputs are considered from $Q_A\ Q_B\ Q_{C.}$ Initially, FF's are SET to QC, QB, QA=111 and then clock pulses are applied to it and with each clock the counting is reduced by one count and on the seventh clock the count sequence is QC, QB, QA=111 and when eight clock arrives they all are again set to QC, QB, QA=111. The count sequence is shown in the Truth Table 5.11

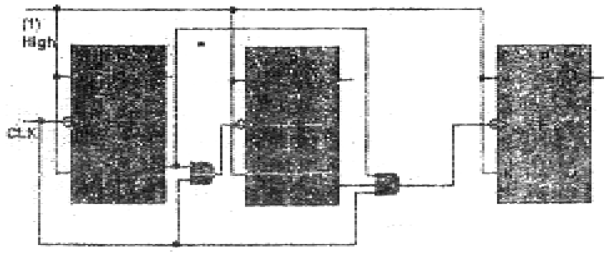


Figure 5.16 Synchronous Down Counter

CLK	Q_C	Q_B	Q_A
0	1	1	1
1	1	1	0
2	1	0	1
3	1	0	0
4	0	1	1
5	0	1	0
6	0	0	1
7	0	0	0
8	1	1	1

Table 5.11: Truth Table for Synchronous Down Counter

Up-Down Counter

This circuit (figure 5.17) has both the features of Up-and Down counter when clock is applied to UP input and Down input is kept zero then all the A-AND gates are enabled and all B-AND gates are disabled the circuit works as Up counter as discussed earlier.

When clock is applied at DOWN input and Up input is kept at zero then all B-

AND gates are enabled and A-AND gates are disabled now the circuit works as normal down counters as discussed above.

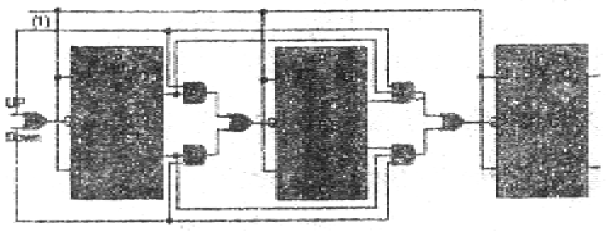


Figure 5.17 Synchronous Up Down Counter

ASYNCHRONOUS COUNTERS

Ripple Counter or Up Counter or Mod-16 Counter

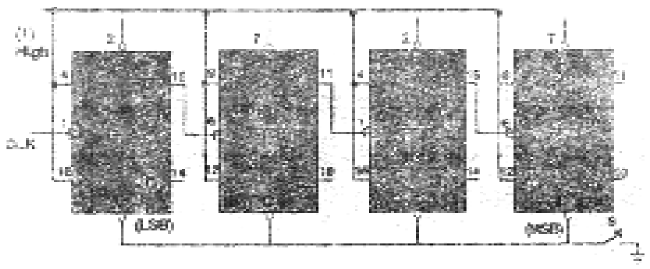


Figure 5.18 Asynchronous Ripple Counter

The binary ripple counter using IC 7476 is shown in Figure 5.18. The four M/S J-K FFs (or TFF) are connected in cascade where FFA represents the least significant bit (LS) and FF D the most significant bit (MSB). The flip flops in the Figure are negative edge triggered which is shown by notch and a bubble at the input of clock in input. All the J and K inputs are tied to +Vcc (HIGH) which means that each flip flop toggles with a negative transition at it clock input. The output of each flip flop is used as the clock input for the next flip flop. Since the clock applied to flip flop's is not in synchronism with the CLK applied to FF A it is called asynchronous counter.

Working Principle (Truth Table5.12):

a) Clear all Q outputs of each FF by momentarily closing switch S and then releasing which give zero at preset inputs that clears all outputs

$Q_D\ Q_C\ Q_B\ Q_A = 0000$

b) When the first clock pulse comes then at its negative edge or trailing edge FF A toggles from 0 to 1. Since 0 to 1 is positive trigger pulse for FF B it does not respond because it is negative edge triggered and responds only when FF A's output goes from 1 to 0. Therefore the output at the end of first clock cycle is

$Q_D\ Q_C\ Q_B\ Q_A = 0001$

c) When second clock cycle comes then at it negative edge again FF A toggles (as FF A toggles every time clock pulse comes) from 1 to 0. Since this is negative trigger pulse for FF and now it responds by toggling from 0 to 1. The change at the output of FF B is from 0 to 1 which is positive going this change does not triggers FF C as it responds only when output of FF B goes from 1 to 0 therefore the output at the end of second clock cycle is

$Q_D\ Q_C\ Q_B\ Q_A=0010$

d) When third cycle comes FF A toggles from 0 to 1, this positive transition makes no change in FF B and the output of FF B remains 1. Therefore, at the end of third clock cycle the output is

$Q_D\ Q_C\ Q_B\ Q_A=0011$

e) When the fourth clock cycle comes FF A toggles 1 to 0 this negative change triggers FF B and its output also toggles from 1 to 0, this is turn toggle FF C from 0 to 1 as each flip flop toggles when the previous FF's output goes from 1 to 0. Now at the end of the fourth clock cycle the output is

$Q_D\ Q_C\ Q_B\ Q_A=0100$

Like this, the counting advance upwards up to $Q_D\ Q_C\ Q_B\ Q_A=1$ and when 16th clock pulse is applied all FF's toggles and the O/P output becomes 0000. The truth table shows the count sequence of the counter from 0000 to 1111.

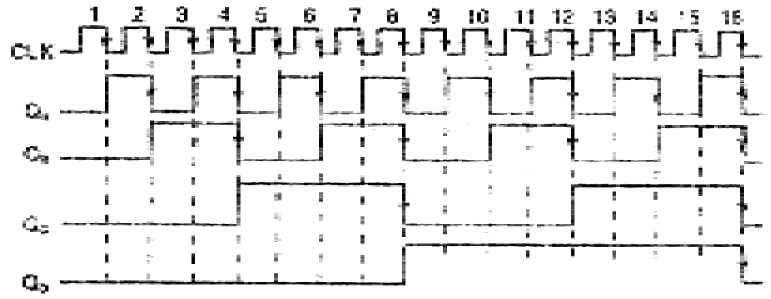


Figure 5.19 Waveform of Ripple Up Counter

CLK	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Table 5.12: Truth Table for Asynchronous Ripple Counter

The wave from shows the action of the counter as the clock advance. Every time there is negative transition of clock FF A (LSB Counter) toggles and the other toggles when the previous FF's output goes from HIGH to LOW. Thus the trigger moves through the flip flop like a ripple in water.

It can be seen from the waveform that the input clock is divided by each flip flop by a value of 2. Suppose clock applied to FF A has the frequency of 16 kHz. The FF A will divide this clock by 2 and it output will generate a clock frequency of 8kHz now the input of FFB is a clock of 8kHz, it will divide it by 2 and thus the output of FF B will produces a clock of 4 kHz. FF C will further divide thus clock and its output will generate a clock cycle of 2 kHz. Lastly, FF D will further divide this 2kHz by 2 and it output will generate a clock frequency of 1kHz. This division of frequency can easily be seen from the waveform. Figure 5.19 shows the waveform.

Down Counter

Here, little modification has been made from the previous counter, in stead of applying the clock from the Q output here we apply from the 1 output and the output is taken from Q_C Q_B and Q_B Q_{A}. The Figure 5.20 shows the circuit arrangement.

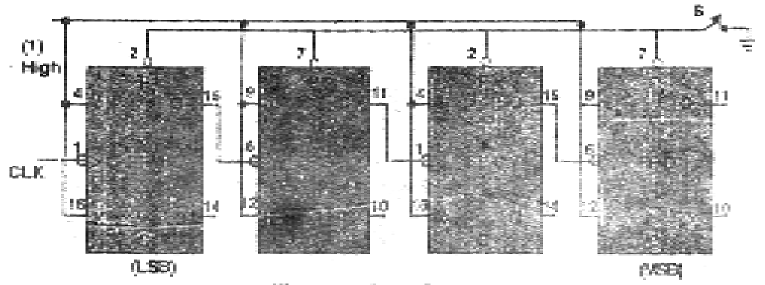


Figure 5.20 Down Counter

Working Principle (Table 5.13):

a) The switch S is momentarily pressed for few seconds which gives 0's at the preset input, which presets the output to

Q_D Q_C Q_B Q_A=1111

b) When first clock pulse is applied then its negative edge FF A toggles Q_A =0 and Q_A and 'Q_A =1. Since 'Q_A has changed from 0 to 1 which is positive clock for FF B it does not toggles and the output of counter at the end of first clock cycle is

Q_D Q_C Q_B Q_A=1110

c) When second clock pulse is applied than FF B toggles and its output becomes Q_B =0, the reason for that is its clock input receives negative transition of the clock as produced by the output of FF A (FFA toggles every time when clock pulse arrives) and the rest of the flip flops do not respond. Therefore, the output at the end of the second clock pulse is

Q_D Q_C Q_B Q_A=1101

d) When third clock pulse arrives only FFA toggles and the rest of lip flops remain unchanged. Therefore, the output at the end of third clock pulse is

$Q_D\ Q_C\ Q_B\ Q_A$=1100

e) When fourth clock pulse arrives at its negative edge FF A toggles and its output becomes Q_A =1 and 'Q_A changes from 1 to 0 which is negative going pulse for the FF B. Due to which the state of FF B is changed from 0 to 1 and that of 'Q_B 1 to 0 is connected to the clock input of FF C it toggles on the negative transition from 1 to 0.

$Q_D\ Q_C\ Q_B\ Q_A$=1011

Like this we go on applying the clock pulses the output go on decreasing by one count and at the end of 15th clock pulse the output becomes $Q_D\ Q_C\ Q_B\ Q_A$=0000. And when 16th clock pulse is applied the output becomes $Q_D\ Q_C\ Q_B\ Q_A$=1111. The count sequence of a down counter is shown in the truth Table 5.13.

CLK	*Q_D*	*Q_C*	*Q_B*	*Q_A*
0	1	1	1	1
1	1	1	1	0
2	1	1	0	1
3	1	1	0	0
4	1	0	1	1
5	1	0	1	0
6	1	0	0	1
7	1	0	0	0
8	0	1	1	1
9	0	1	1	0
10	0	1	0	1
11	0	1	0	0
12	0	0	1	1
13	0	0	1	0
14	0	0	0	1
15	0	0	0	0
16	1	1	1	1

Table 5.13 : Truth Table for Down Counter

Up Down Counter

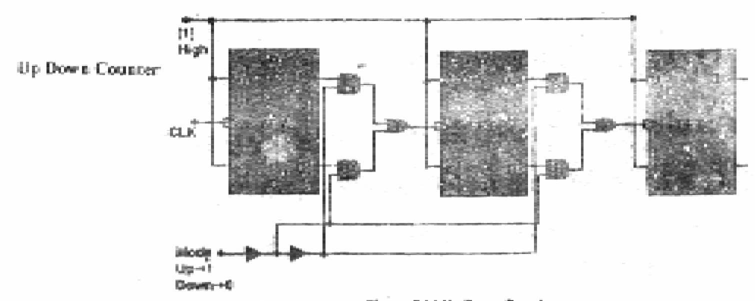


Figure 5.21 Up Down Counter

Here (Figure 5.21) additional hardware is connected between the outputs of FF A and clock input of FF B and the output of FF B and input of FF C in order to make the counter an up-down counter.

Up Counter

To make the circuit work as Up-Counter MODE input is kept at 1, this enables all A AND gates and disables all B AND gate. So any change (negative transition0 occuring at the Q outputs is transmitted to the next FF's clock input and counter works as normal UP-Counter as explained in asynchronous counters.

Down counter

To make the circuit work as Down counter 0 is given at the MODE input which enables all B AND gate and disables all A AND gate. So any change at the 1 outputs is transmitted to the next FF's clock input and the counter works as normal down counter as explained in case of asynchronous counters.

Mod 10 Counter or Decade Counter or BCD 8421 Counter

A counter that recycles in 10 pulses is called a MOD-10 or DECADE or BCD COUNTER. There are many ways to design this counter but we will discuss the method, which resets all FFs after a desired count is reached. The circuit shown in Figure 5.22 is a binary ripple counter that could count up to 16 but some modification are made which allows the circuit to count up to 9 and on the application of 10th pulse the counter is reset to 0000.

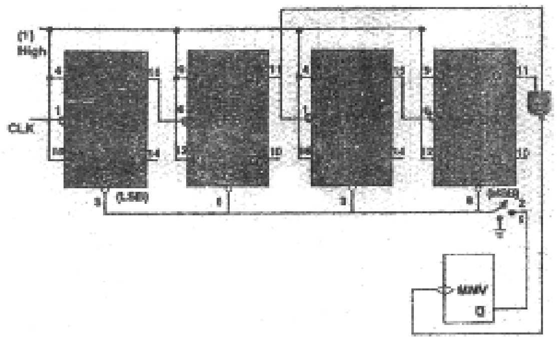


Figure 5.22 Decade Counter

Working Principle (Table 5.14):

a) Switch S is momentarily connected to point 1 and then kept permanently at 2. Which RESET the counter to

$Q_D\ Q_C\ Q_B\ Q_A$=0000

b) Input pulses advances counter as binary up counter up to 9 (1001) as a normal binary ripple counter as shown in the truth table of decade counter.

c) The next count pulse advance the count to 10 (Count=1010) but since the Q_D and Q_B outputs are given to the NAND gate which provides 0 at its output providing a 1 to 0 logic change to trigger the monostable multivibrator (MMV). This provides a short pulse to reset all counter's. The 'Q output signal is sued since it is normally high and goes low during the one shot timing period, the flip flop in this circuit being reset to $Q_D\ Q_C\ Q_B\ Q_A$=0000 by low signal level. The output of the NAND gate could have been directly connected to the CLEAR input to reset the Counter, but this is not reliable because propagation delay from CLEAR input to Q output of the FF varies from one flip flop to other. For example if Q_D out put takes longer time to reset than $Q_{A,}$ then the output of the NAND gate again goes to 1 when Q_A returns to 0, this make CLEAR =1 with the result that FF D will not reset. This problem makes importance when the counter outputs are loaded unevenly. So the use of one shot (Mono Stable Multivibrator) memorises the output of the NAND gate at the 10th pulse and eliminates the problem.

CLK	*Q_D*	*Q_C*	*Q_B*	*Q_A*
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
	0	0	0	0

Table 5.14: Truth Table BCD Decade Counter

These type of circuits has a disadvantage that the circuit uses a single shot unit and special timing adjustment on pulse duration. To overcome this difficulty direct reset counters are used.

Direct Reset Decade Counters

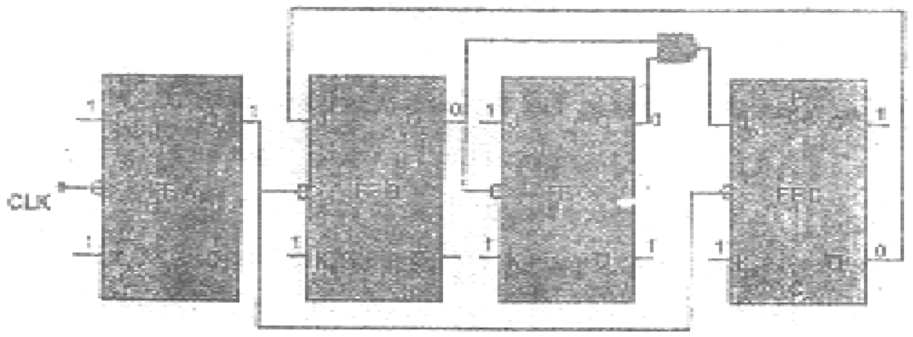


Figure 5.23 Direct Reset Decade Counter

Working Principle:

a) FFM A toggles at each input pulse.

b) FF B toggles when FF A output change from 1 to 0 except when Q_D=1and ‘Q_D=0 because during this condition FF B output is Q_B =0 and its input J_A and K_A are 0 and respectively.

c) FF C toggles when FF B output goes from 1 to 0.

d) FF D is reset each time FFA goes from 1 to 0 except when FF B and FF C are both are logical 1.

Let us see the operation of the circuit during 9th & 10th clock pulses. After 9th clock pulses hits the output are

$$Q_D\ Q_C\ Q_B\ Q_A=0000$$

These conditions are shown in the Figure 5.23 for better understanding of the condition. Now the ‘Q_D has become 0 which is given to the J_B input as the output of FF B i.e. J_B =0 K_B =1 of FF B and FFC are 0. When 10th clock pulse arrives then FF A toggles from 1 to 0 this shifts the 0 input of J_B to the output i.e. no change in the output of Q_B due to which FF C do not gets any clock at its clock input and its output remains 0 and the input of FF D i.e. J_D =0 and K_D =0 are passed to the output as FF D gets the clock pulse when FF A output changes from 1 to 0. This makes all the flip flops to RESET to Q_D Q_C Q_B Q_A=0000. Like MOD-10 counter MOD-9 counter can be easily realised.

Mod 5 Counter

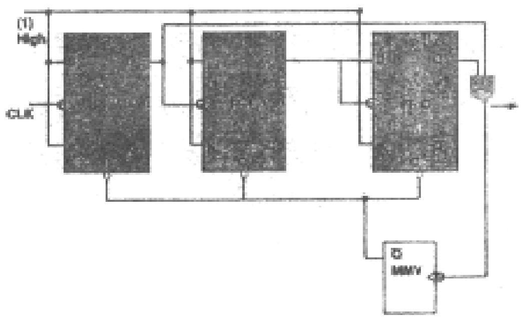


Figure 5.24 MOD 5 Counter

For MOD-5 counter we require 3-FF's. It works as normal ripple counter up to 4th clock pulses and on the 5th clock pulse all FF's are RESET to ZERO the working of the counter is similar to that of MOD –10 counter, this is shown in the truth table given below (Table 5.15).

CLK	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
	0	0	0	0

Table 5.15: Truth Table for MOD 5 Counter

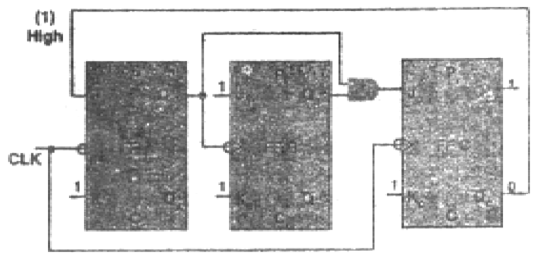


Figure 5.25 Direct Reset MOD-5 Counter

When 4th clock hits flip flops the output condition is Q_C Q_B Q_A=000. Q_C is connected to J_A input. So input of FF A is J_A=0, K_A =1, FF B acts as normal TFF and the input to FF C is J_C and K_C=1. When 5th clock pulse comes 0 at the input at FF A is passed to output.

Since there is no change at the of FF A, FF B do not respond and its output remains at 0 FF C input J_c=0, K_c =1 are passed to the output and thus MOD 5 counters output becomes - Q_C Q_B Q_A=000.

Ring Counters

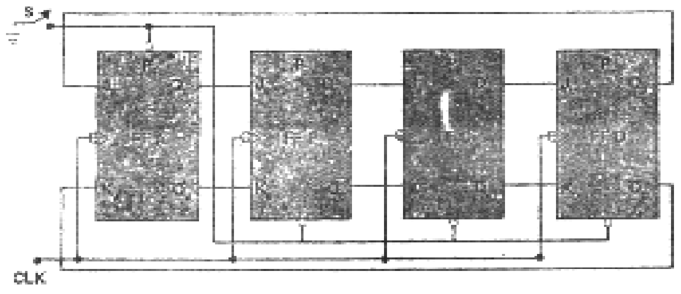


Figure 5.26 Ring Counter

Ring Counter is the simple serial shift register in which a single bit shifts from one flip-flop to the another in the form of a ring as shown in Figure 5.26.

Working Principle (Table 5.16):

1. The switch S is pressed which clear the FF B, FF C and FF D and sets FF A.

   $Q_D\,Q_C\,Q_B\,Q_A$=0001

2. Before the 1st clock hits input to FF B is 1 to rest FF's having 0 inputs, at the end of clock pulse the state of the FF's is

   $Q_D\,Q_C\,Q_B\,Q_A$=0010

3. The FF C has 1 input and the rest of the flip-flop have 0 inputs and when the succeeding clock arrives, the input data's are stored in the respective flip-flops.

   $Q_D\,Q_C\,Q_B\,Q_A$=0100

4. Now the FF D has 1 input and the rest of the flip-flops have 0 input and when 3rd clock pulse comes this 1 is shifted into the FF D.

   $Q_D\,Q_C\,Q_B\,Q_A$=1000

5. Since the Q_D output is directly connected to the input of FF A. When 4th clock pulse comes this 1 is shifted into the FF A and the process repeats till the clock pulses are applied.

CLK	Q_D	Q_C	Q_B	Q_A
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0
4	0	0	0	1
5	0	0	1	0
6	0	1	0	0
7	1	0	0	0

Table 5.16: Truth table for Ring Counter

Self- Correcting Ring Counter

Sometimes due to false triggering the count sequence may be disrupted and we might not get the actual sequence of the ring counter. To avoid such conditions, self-correcting ring counter is used which is modified form of Ring counter.

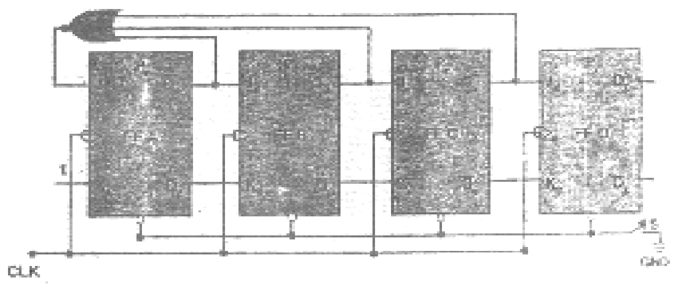


Figure 5.27 Self Correcting Ring Counter

Working principle:

1. Press the switch S to GND and then keeping it high initially resets all the FF;s.

   $Q_D\, Q_C\, Q_B\, Q_A$=0000

2. The Q_A Q_B and Q_C output are connected to the input of NOR gate whose output become 1, then the input of FF A is now 1 and reset all other FF's have 0 inputs.

3. When the 1st clock is applied the 1 the input of FF A is transferred to the output.

   $Q_D\, Q_C\, Q_B\, Q_A$=0001

4. Now the input of NOR is 1 and 0 and 0 which give 0 at the output (if any input of NOR gate is high the output is always low) which is the input to FF A, when 2nd clock pulse comes the 1 at the input of FFB, i.e. 1 is transferred to its output.

   $Q_D\, Q_C\, Q_B\, Q_A$=1000

   and in this way the process goes on like normal ring counter.

Advantages

No FF's is set before the first clock comes.

If by change due to false triggering or other reasons two FF's are set to 1 then in simple Ring counter, this extra one will keep on rotating in the loop and there is no option to correct it while counting is going on but on self- correcting ring counter this problem is eliminated after 2 or 3 cycles.

Johnson or Twisted Ring Counter

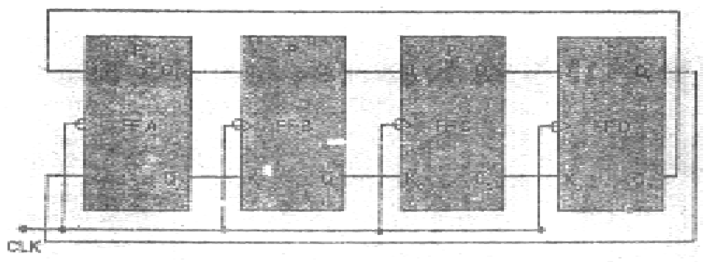


Figure 5.28 Johnson Ring Counter

In this circuit the output of last FF in cross connected to FF A, this technique is called inverse feedback due to each the sequence obtained is very different from the other count sequence which looks like as twisted one (Figure 5.28).

Working principle (Table 5.17):

1. Initially all FF's are RESET.
2. Now the input of FF a has 1 input and the rest flip-flops have 0 input on the

application of 1st clock pulse these inputs are stored in the respective FF's and can be seen at the outputs.

$Q_D\, Q_C\, Q_B\, Q_A$=0001

3. Again FF A is 1 and FF B is 1 rest have 0 input, when 2nd clock pulse comes these inputs are stored by the respective FF's and the count sequence is:

$Q_D\, Q_C\, Q_B\, Q_A$=0011

4. Now the input to FF A and FFB is 1 rest have 0 input, when 3rd clock pulse comes then input are stored in the respective FF's and the count sequence is

$Q_D\, Q_C\, Q_B\, Q_A$=0111

5. All flip-flops now have 1 input and when next clock pulse comes these inputs are stored in the respective FF's

$Q_D\, Q_C\, Q_B\, Q_A$=1111

6. Now the input to FF A has 0 input and the rest other has 1 input thus on the arrival of the 5th clock pulse the outputs are stored in the input of the respective flip-flop

$Q_D\, Q_C\, Q_B\, Q_A$=1110

The Truth Table 5.17 shows the count sequence of the Johnson Counter.

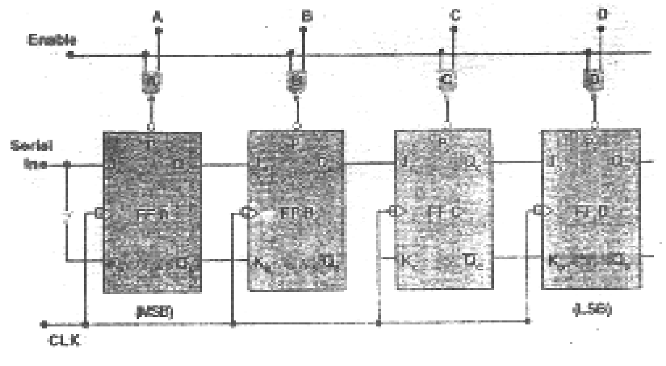


Figure 5.29 Register

CLK	Q_D	Q_C	Q_B	Q_A	*Decimal Equivalent*
0	0	0	0	1	0
1	0	0	0	1	1
2	0	0	1	1	3
3	0	1	1	1	7
4	1	1	1	1	15
5	1	1	1	0	14
6	1	1	0	0	12
7	1	0	0	0	8
8	0	0	0	0	0

Table 5.17: Truth Table for Johnson Ring Counter

5.4 Registers

A group of flip flops used for temporary data storage is known as registers. The features of registers are listed below:

- DFF-is used for storing a single bit data.
- To store n-bit data n-registers are required.
- Data can be entered in the register serial or parallel form and can be taken out serially or in parallel form.
- Shift registers are extensively used in arithmetic operations like multiplication (left shift) and division (right shift) of binary numbers.

  The four basic type of registers are

- Serial In Serial Out (SISO)
- Serial In Parallel Out (SIPO)
- Parallel In Serial Out(PISO)
- Parallel In Parallel Out (PIPO)

Serial In Serial Out

Working principle:

1. Initially all FF's are reset by apply a clear pulse at clear input.
2. LSB of data 1011 is applied to the serial input terminal.
3. When the first clock pulse comes at the negative edge of the clock this LSB bit i.e. 1 is stored into the FF A and the register outputs look like

   $Q_D\, Q_C\, Q_B\, Q_A$=1000
4. Now input to FF B is 1 because it is directly connected to the output of FF A and when second clock pulse comes input of FF A is stored in it and also the previous output of FF A which was 1 is stored in FF B.

   $Q_D\, Q_C\, Q_B\, Q_A$=1100
5. Input to FF C is 1 input to FF B is 1, now third data i.e. 0 is applied to FFA's input and when third clock arrive, these data are stored in two respective FF's as shown below

   $Q_D\, Q_C\, Q_B\, Q_A$=0110
6. The last input 1 is applied to the FF A and input to FFC is 1, input to FF B is 0 now when fourth clock comes these input data are stored in the respective FF's.

   $Q_D\, Q_C\, Q_B\, Q_A$=1011

This method of storing the data is called **Serial Data Storage.** Further if we apply fourth clock pulses. The data is shifted out of the right end of the register and lost after four clock times.

Serial In Parallel Out

In this register the data is shifter serially into the register as explained in case of SISO but the data is shifted out in parallel form. In order to get the data out in parallel form it is necessary to have all data bits available as the output at same time. This is done by connecting four parallel wires to get from $Q_D\, Q_C\, Q_B\, Q_{A.}$

Parallel In Serial Out

To enter the data (1011) in parallel form, inputs are given at A,B,C,D terminal. Initially, all registers are cleared, therefore, when we have to shift the data into the register we apply to ENABLE pulse at ENABLE input, now the output of

A-NADN gate is $\overline{1.1} = 0$

B-NAND gate is $\overline{0.1} = 0$

C-NAND gate is $\overline{1.1} = 0$

D-NAND gate is $\overline{1.1} = 0$

To clear input of FF A is 0so the output of FF A will be 1, the clear input of FF B is 1 which keeps the output of FF B at 0 the clear input of FF C is 0 so the output of FF C becomes 1 and the clear inputs of FF D which sets the output of FF D to 1. The register contents are

$$Q_A\ Q_B\ Q_C\ Q_D=1011$$

By applying a single pulse we have stored all four bit sat same time this is called parallel storing data. To retrieve the serially we apply four clock pulses at the CLK input and data is out of the register.

Parallel in Parallel Out

The data is stored in parallel form as described in case of PISO and retrieved at the output in parallel form through four wires connected at each output.

Right – Left Shift Register

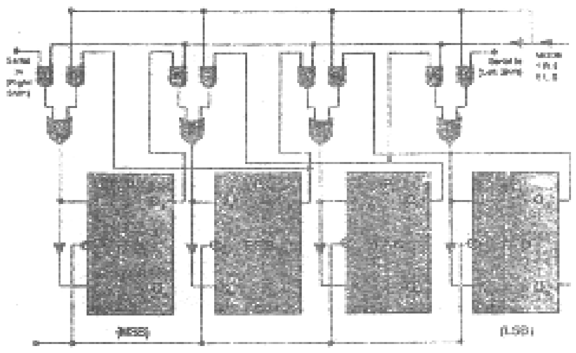


Figure 5.30 Right - Left Shift Register

Right-Left Shift Register find its extensive use in arithmetic unit for multiplication and division purpose. When the number has to be divided then with the shift of data toward right divides the number by two with each shift and for multiplication the data is left shifted, with each shift of the data towards left the number is multiplied by two.

Right Shift

To shift the data towards right MODE CONTROL line is kept HIGH which enables all A AND gates and disables all B AND gates. Let data 1011 is to be shifted towards right.

1. We give 1st bit of data i.e. 1 at the SERIAL IN terminal, since all A AND gates are enabled this data reaches to the input of the FF A, when negative edge of th first clock comes this 1 is stored in FF A and is received at QA output. The state of the output is

   $Q_D\,Q_C\,Q_B\,Q_{A.}= 1000$

2. The QA output passed by A2 AND and respective OR gate, the input of FF B becomes JA=1 and KA=0. Now we give the next data i.e. 1 at the SERIAL IN terminal which reaches to the input of FF A, when negative edge of 2nd clock hits the input data FF A and FF B are stored in the respective FF's

   $Q_D\,Q_C\,Q_B\,Q_{A.}= 1100$

3. The output of FF A and FF B reaches to the input of FF B and FF C respectively after passing through their respective AND and OR gates, now the input to FF B is J_B=1 K_B=0, FF C is J_C=1, K_C=0. Now the third data bit i.e. 0is applied at the SERIAL IN Terminal which ultimately reaches to the input of FA A therefore, when negative edge of 3rd clock comes all the inputs are stored in the respective FF's. The output state of the flip flop are

   $Q_D\,Q_C\,Q_B\,Q_{A.}$=0110

4. The output of FF A reaches to the input of FF B through A2 AND and OR gate, the output of FF B reaches to the input of FF C through A3 AND gate and respective OR gate and finally the output of FF C i.e. 1 reaches to the input of FF D through A4 AND gate and respective OR gate.

   When last data bit 1 is applied to the SERIAL IN reaches at the input of FF A. When negative edge of 4th clock arrives all the inputs at each FF is stored in the respective flip flop.

   $Q_D\,Q_C\,Q_B\,Q_{A.}$=1011

This is right shifting of data, the data can be taken out in parallel form through four parallel wires or serially from Q_D by applying four clock pulses.

Left Shift

To shift the data towards left the MODE CONTROL line is placed at ZERO which enables all B AND gates and disables A AND gates. The data is applied from one of the pin of B_4 AND gate shown in Figure 5.30.

For example the data 1001 has to be shifted towards left then following operations follows:

1. Giving 1 at the SERIAL IN (Left Shift) terminal, which reaches to the input of FF D as B4 AND gate OR gates are enable therefore, when the negative edge of the 1st clock comes the 1 at the input is stored in the FF D, the output of FF D is connected to B_3 AND gate which passes this output to the input of FF C.

2. The second data i.e. 0 at the SERIAL IN (Left Shift) terminal reaches to the input of FF D and when the 2nd clock pulse arrives then these data's at the input of the FF D and FF C are stored in the respective FF's, the register contents are:

   $Q_A\, Q_B\, Q_C\, Q_{D.}$=0010

3. The output of the flip flops reaches to the next flip flop, third data i.e. 0 is given at the input terminal which reaches to the input of FF D and when the third clock pulse hits the FF C are stored in the respective FF's, the register contents are:

   $Q_A\, Q_B\, Q_C\, Q_{D.}$=0100

4. The output Q_B is connected to one of the pin of B_1 AND gate so the 1 at the output of Q_B is passed to the input of FF A, like wise all other data are passed to their next FF's the last data bit which is 1 is given to the input terminal which reaches at the input of FF D and when the 4th clock pulse comes at the negative edge of it the data present at the input to each FF are stored in them

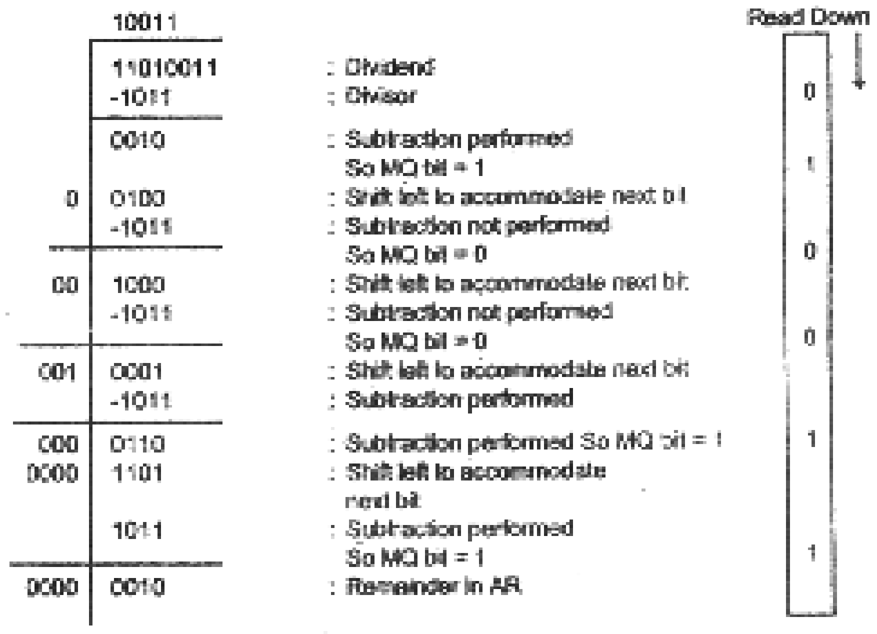
   $Q_A\, Q_B\, Q_C\, Q_{D.}$=1011

   The data can be collected in parallel by four parallel wires or serially from Q_A by applying four clock pulses. This is left shifting of data.

Assignment

Show the machine division of $(11010011)_2$ by $(1011)_2$

Solution : In this example we tried to show how a binary division can be performed in a manner similar to multiplication using Register. In binary division repeated subtraction and shift left is used.



```
         10011                                                Read Down
       | 11010011   : Dividend
       | -1011      : Divisor                                     0
       | 0010       : Subtraction performed
       |              So MQ bit = 1                               1
     0 | 0100       : Shift left to accommodate next bit
       | -1011      : Subtraction not performed
       |              So MQ bit = 0                               0
    00 | 1000       : Shift left to accommodate next bit
       | -1011      : Subtraction not performed
       |              So MQ bit = 0                               0
   001 | 0001       : Shift left to accommodate next bit
       | -1011      : Subtraction performed
   000 | 0110       : Subtraction performed So MQ bit = 1         1
  0000 | 1101       : Shift left to accommodate
       |              next bit
       | 1011       : Subtraction performed
       |              So MQ bit = 1                               1
  0000 | 0010       : Remainder in AR
```

The answer is in quotient: 10011 with remainder 0010 in AR.

The basic cycle can be summarized as follows:

1. Shift the data into subtractor unit by applying shift pulses (SP).
2. If subtraction is possible MQ=1, if not MQ=0.
3. Shift left the subtracted value left by one place to accommodate next bit.
4. If all dividend data is accessed, if YES output MQ data, else Go To number 1.