7

Registers

7.1 Introduction

we know that, a flip-flop is nothing but a binary cell capable of storing one bit information, and can be connected together to perform counting operations. Such a group of flip-flops is called **counter**. We have also seen that group of flip-flops can be used to store a word, which is called **register**. In next sections we are going to study various types of counters and registers.

A register is a group of flip-flops. A flip-flop can store 1-bit information. So an n-bit register has a group of n flip-flops and is capable of storing any binary information/number containing n-bits.

7.2. Buffer Register

Fig. 7.1 shows the simplest register constructed with four D flip-flops. This register is also called **buffer register**. Each D-flip-flop is triggered with a common negative edge clock pulse. The input X bits set up the flip-flops for loading. Therefore, when the first negative clock edge arrives, the stored binary information becomes,

$$Q_A Q_B Q_C Q_D = ABCD$$

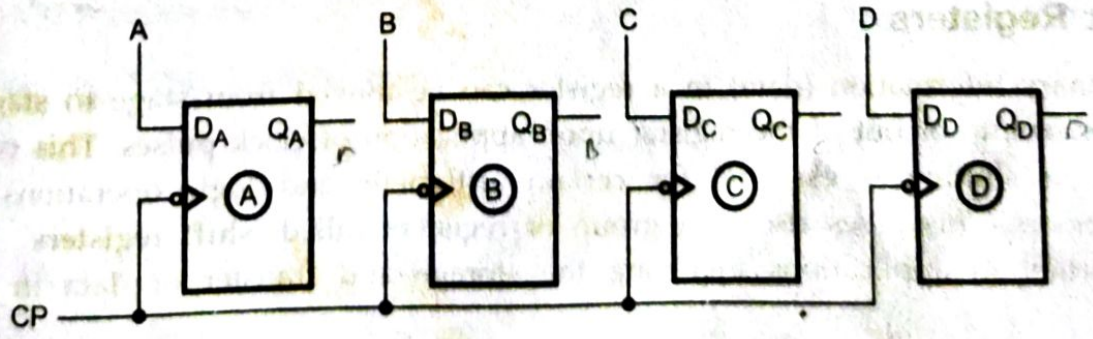


Fig. 7.1 Buffer register

In this register, four D flip-flops are used. So it can store 4-bit binary information. Thus the number of flip-flop stages in a register determines its total storage capacity.

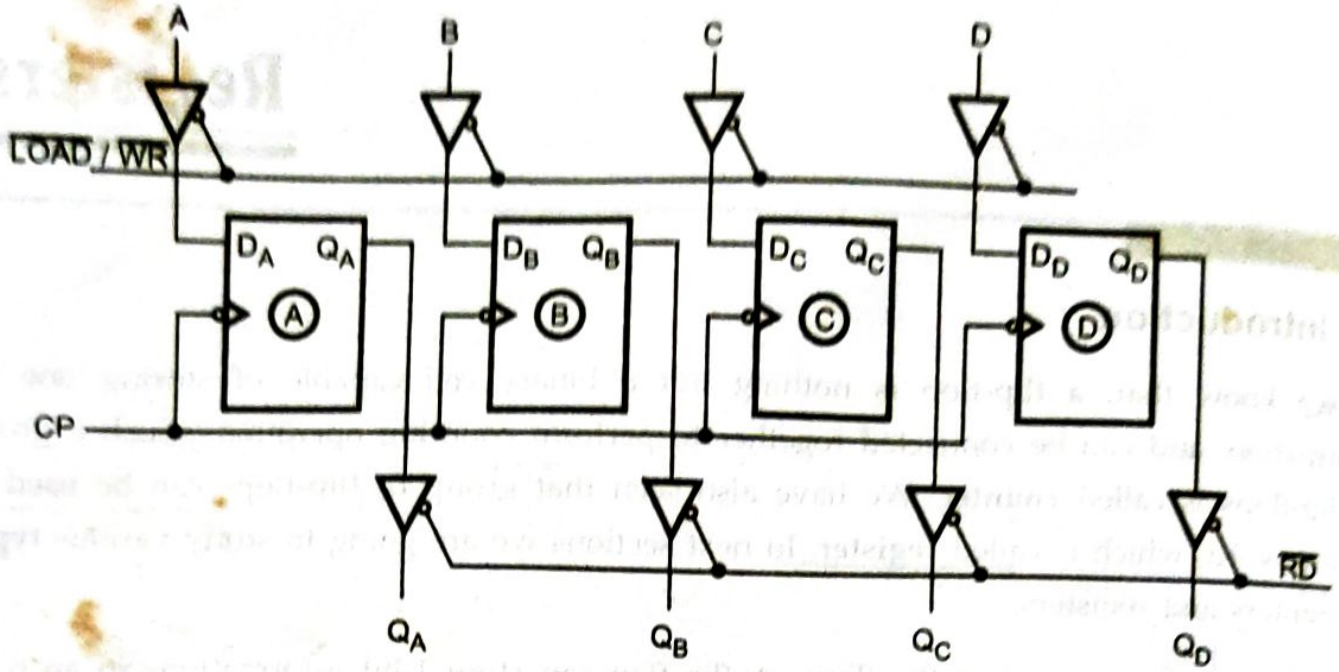


Fig. 7.2 Controlled buffer register

In this buffer register, there is no control over input as well as output bits. We can control input and output of the register by connecting tri-state devices at the input and output sides of register as shown in Fig. 7.2. So this register is called '**controlled buffer register**'.

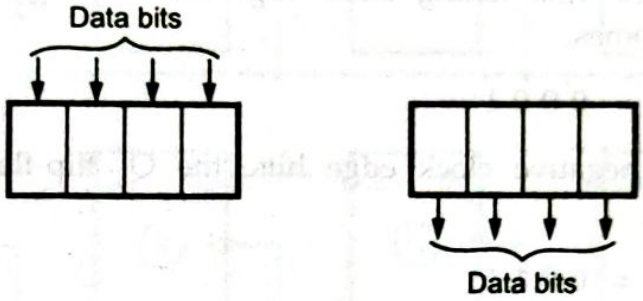
Here, tri-state switches are used to control the operation. When you want to store data in the register, you have to make $\overline{LOAD}$ or $\overline{WR}$ signal low to activate the tri-state buffers. When you want the data at the output, you have to make $\overline{RD}$ signal low to activate the buffers. Controlled buffer registers are commonly used for temporary storage of data within a digital system.

7.3 Shift Registers

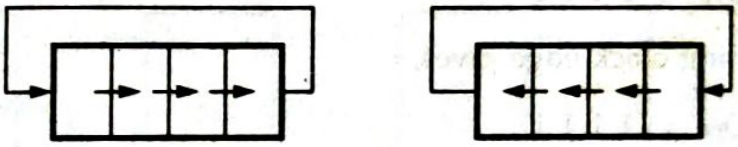
The binary information (data) in a register can be moved from stage to stage within the register or into or out of the register upon application of clock pulses. This type of bit movement or shifting is essential for certain arithmetic and logic operations used in microprocessors. This gives rise to a group of registers called '**shift registers**'. They are very important in applications involving the storage and transfer of data in a digital system.

Fig. 7.3 gives the symbolical representation of the different types of data movement in shift register operations.

(a) Serial shift right, then out (b) Serial shift left, then out



(c) Parallel shift in (d) Parallel shift out



(e) Rotate right (f) Rotate left

Fig. 7.3 Basic data movement in registers

According to the data movement in a register, let us see some of the types of shift registers.

Types of Shift Registers

1) Serial In Serial Out Shift Register

Fig. 7.4 shows serial in serial out shift left register.

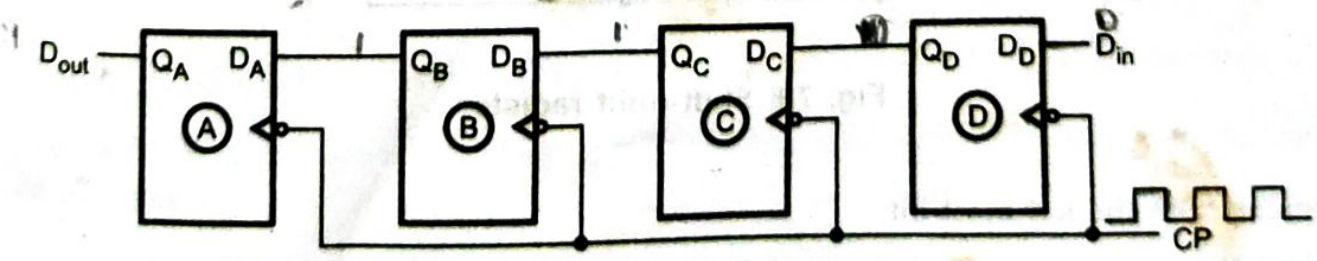


Fig. 7.4 Shift-left register

We will illustrate the entry of the four bit binary number 1111 into the register, beginning with the left-most bit.

Initially, register is cleared. So

$Q_AQ_BQ_CQ_D = 0\ 0\ 0\ 0$

a) When data 1 1 1 1 is applied serially, i.e.

left-most 1 is applied as D_{in}

$D_{in} = 1, \quad Q_AQ_BQ_CQ_D = 0\ 0\ 0\ 0$

The arrival of the first falling clock edge sets the right-most flip-flop, and the stored word becomes,

$Q_AQ_BQ_CQ_D = 0\ 0\ 0\ 1$

b) When the next negative clock edge hits, the Q_1 flip-flop sets and the register contents become,

$Q_AQ_BQ_CQ_D = 0\ 0\ 1\ 1$

c) The third negative clock edge results in,

$Q_AQ_BQ_CQ_D = 0\ 1\ 1\ 1$

d) The fourth falling clock edge gives,

$Q_AQ_BQ_CQ_D = 1\ 1\ 1\ 1$

Fig. 7.5 illustrates the input and output condition for each flip-flop stage upon application of each clock pulse.

Fig. 7.6 shows serial in serial out shift right register.

We will illustrate the entry of the four bit binary number 1111 into the register,

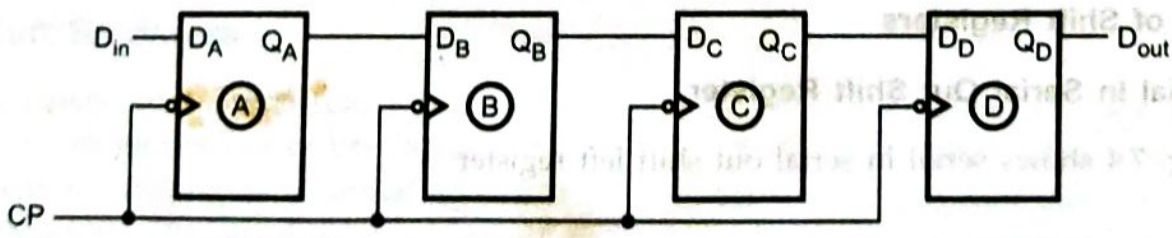


Fig. 7.6 Shift-right register

beginning with the left-most bit.

Initially, register is cleared. So $Q_AQ_BQ_CQ_D = 0\ 0\ 0\ 0$

a) When data 1 1 1 1 is applied serially, i.e. left-most 1 is applied as D_{in}.

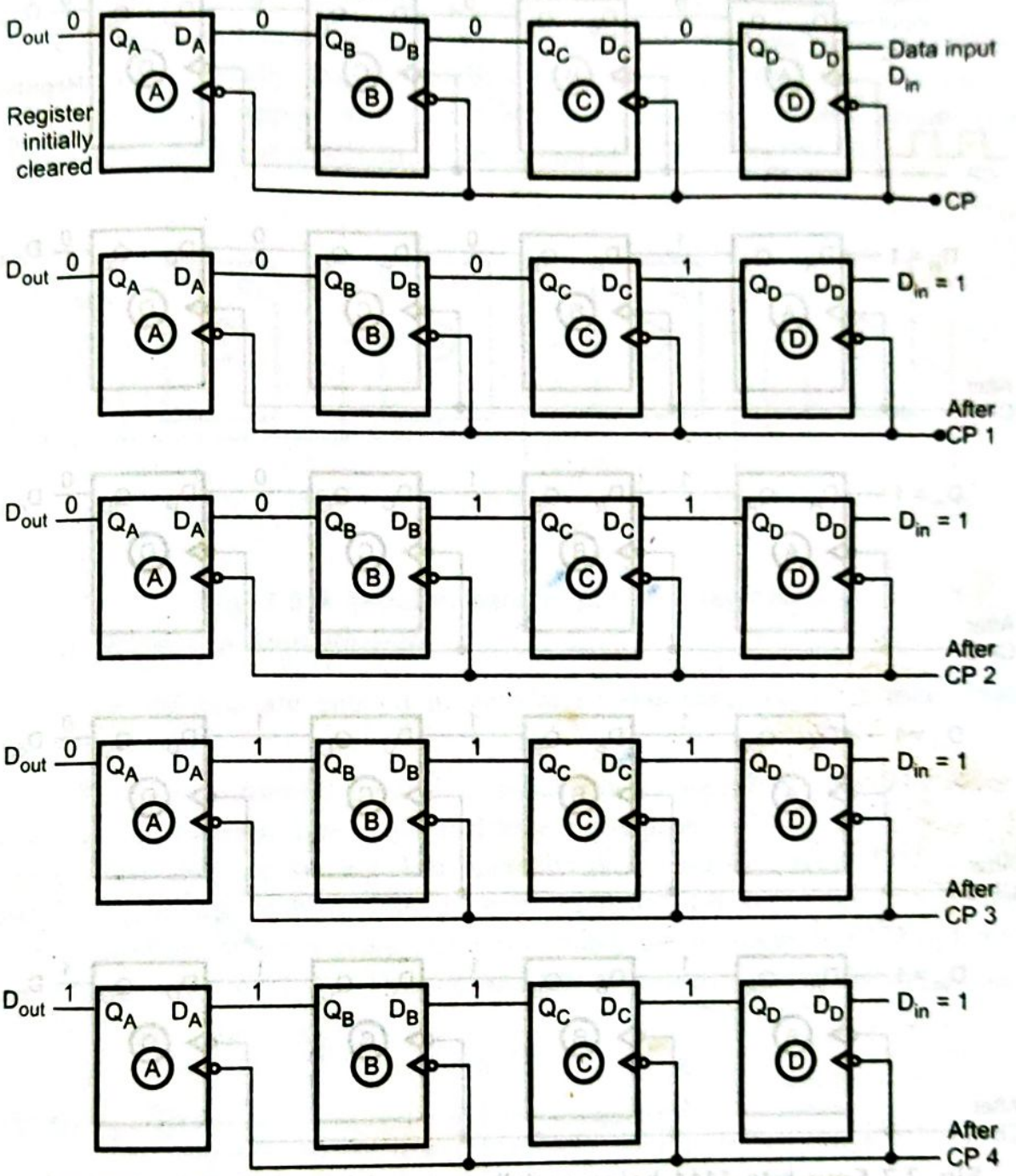


Fig. 7.5 Four bits 1111 being serially entered into shift-left register

$$D_{in} = 1$$

$$Q_A Q_B Q_C Q_D = 0\ 0\ 0\ 0$$

The arrival of the first falling clock edge sets the left-most flip-flop, and the stored word becomes,

$$Q_A Q_B Q_C Q_D = 1\ 0\ 0\ 0$$

b) When the next falling clock edge hits, the Q_1 flip-flop sets and the register contents become,

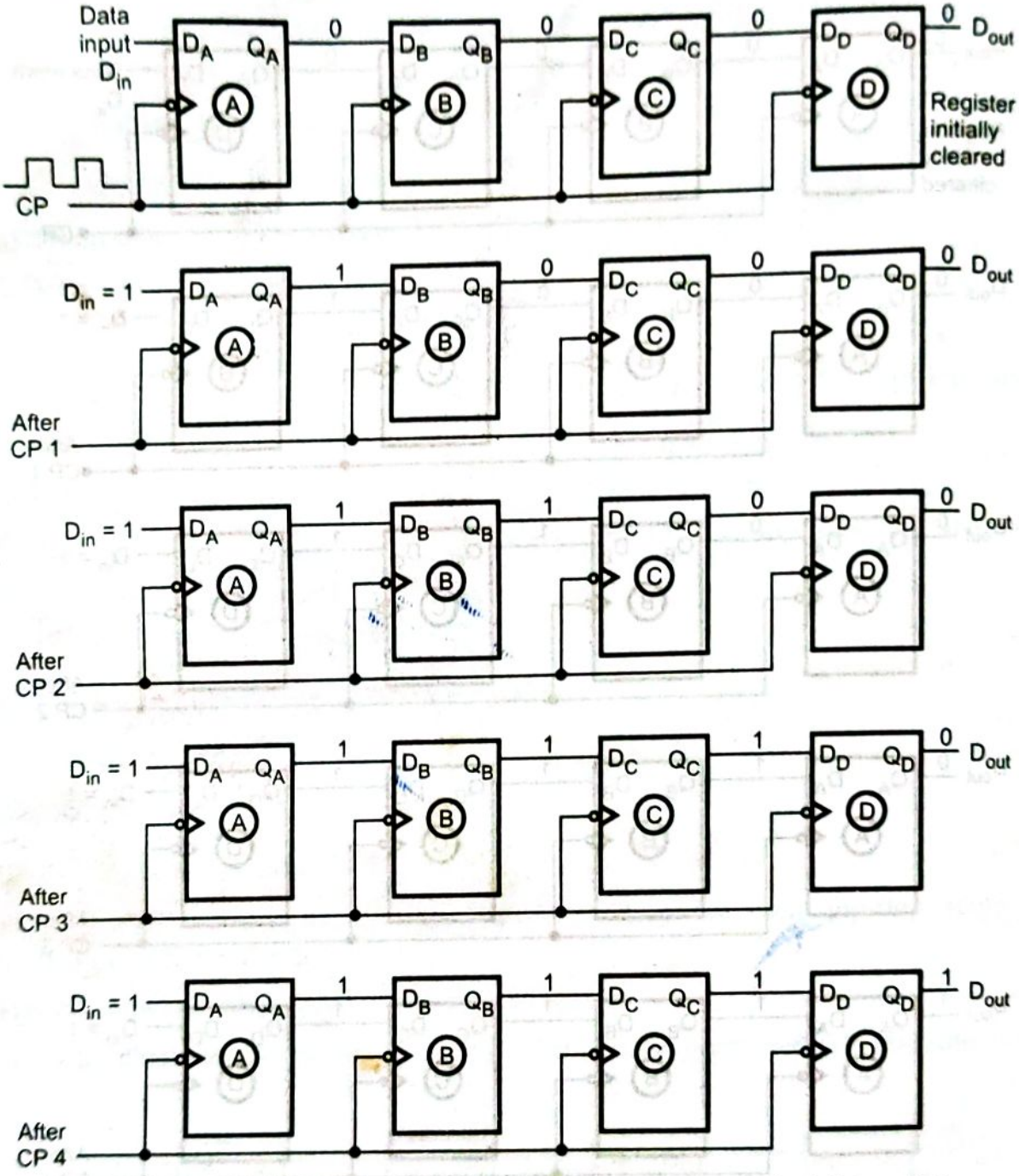


Fig. 7.7 Four bits 1111 being serially entered into shift-right register

$Q_A Q_B Q_C Q_D$ = 1100

c) The third falling clock edge results in,

$Q_A Q_B Q_C Q_D$ = 1110

d) the forth falling clock edge gives,

$Q_A Q_B Q_C Q_D$ = 1111

Fig. 7.7 illustrates the input and output condition of each flip-flop stage upon application of each clock pulse.

2) Serial In Parallel Out Shift Register

In this case, the data bits are entered into the register in the same manner as discussed in the last section, i.e. serially. But the output is taken in parallel. Once the data are stored, each bit appears on its respective output line and all bits are available simultaneously, instead of on a bit-by-bit basis as with the serial output. (Shown in Fig. 7.8).

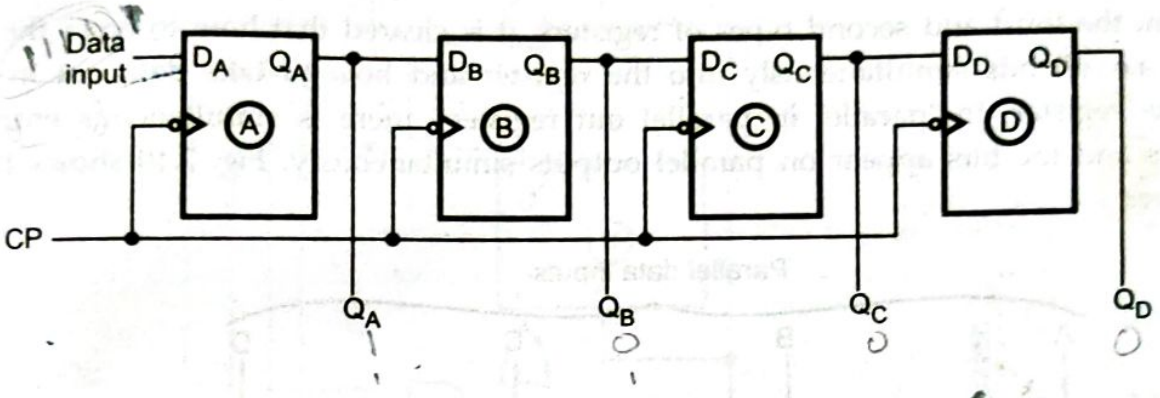


Fig. 7.8 A serial in parallel out shift register

3) Parallel In Serial Out Shift Register

In this type, the bits are entered in parallel i.e simultaneously into their respective stages on parallel lines.

Fig. 7.9 illustrates a four-bit parallel in serial out register. There are four input lines X_A, X_B, X_C, X_D for entering data in parallel into the register. SHIFT/$\overline{\text{LOAD}}$ is the control input which allows shift or loading data operation of the register. When SHIFT/$\overline{\text{LOAD}}$ is low, gates G_1, G_2, G_3 are enabled, allowing each input data bit to be applied to D input of its respective flip-flop. When a clock pulse is applied, the flip-flops with D = 1 will SET and those with D = 0 will RESET. Thus all four bits are stored simultaneously.

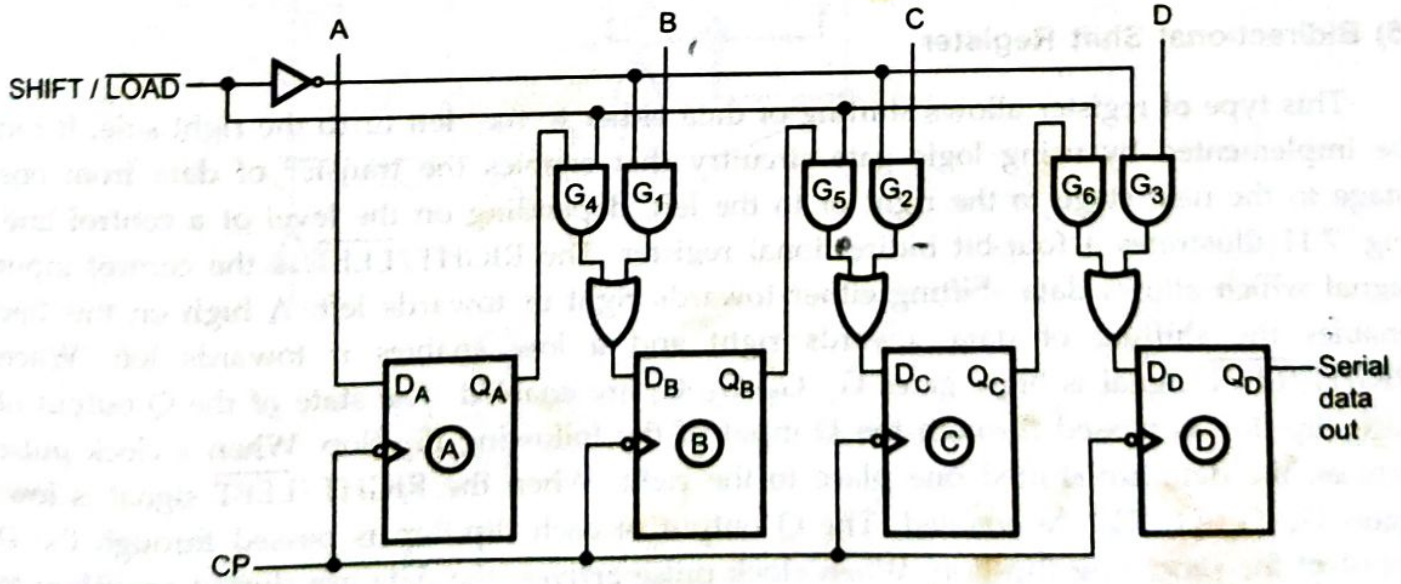


Fig. 7.9 Parallel in serial out shift register

When SHIFT/$\overline{\text{LOAD}}$ is high, gates G_1, G_2, G_3 are disabled and gates G_4, G_5, G_6 are enabled. This allows the data bits to shift left from one stage to the next. The OR gates at the D-inputs of the flip-flops allow either the parallel data entry operation or shift operation, depending on which AND gates are enabled by the level on the SHIFT/$\overline{\text{LOAD}}$ input.

4) Parallel In parallel Out Register

From the third and second types of registers, it is cleared that how to enter the data in parallel i.e. all bits simultaneously into the register and how to take data out in parallel from the register. In 'parallel in parallel out register', there is simultaneous entry of all data bits and the bits appear on parallel outputs simultaneously. Fig. 7.10 shows this type of register.

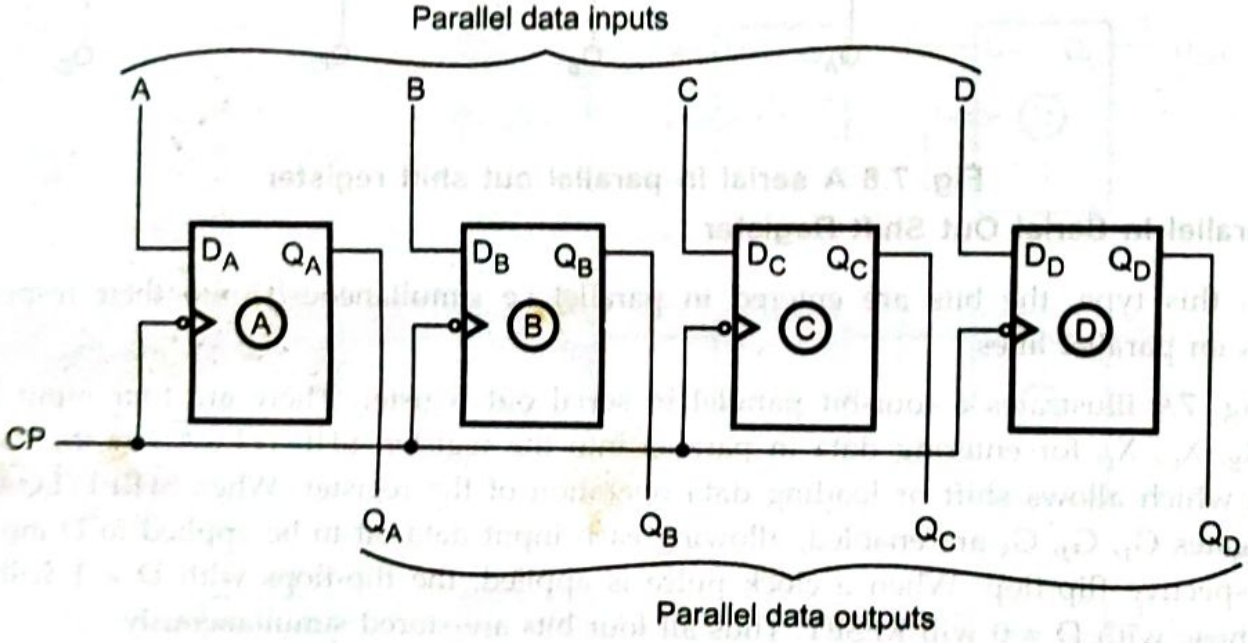


Fig. 7.10 Parallel in parallel out shift register

5) Bidirectional Shift Register

This type of register allows shifting of data either to the left or to the right side. It can be implemented by using logic gate circuitry that enables the transfer of data from one stage to the next stage to the right or to the left, depending on the level of a control line. Fig. 7.11 illustrates a four-bit bidirectional register. The RIGHT/$\overline{\text{LEFT}}$ is the control input signal which allows data shifting either towards right or towards left. A high on this line enables the shifting of data towards right and a low enables it towards left. When RIGHT/$\overline{\text{LEFT}}$ signal is high, gates G_1, G_2, G_3, G_4 are enabled. The state of the Q output of each flip-flop is passed through the D input of the following flip-flop. When a clock pulse arrives, the data are shifted one place to the right. When the RIGHT/$\overline{\text{LEFT}}$ signal is low, gates G_5, G_6, G_7, G_8 are enabled. The Q output of each flip-flop is passed through the D input of the preceding flip-flop. When clock pulse arrives, the data are shifted one place to the left.

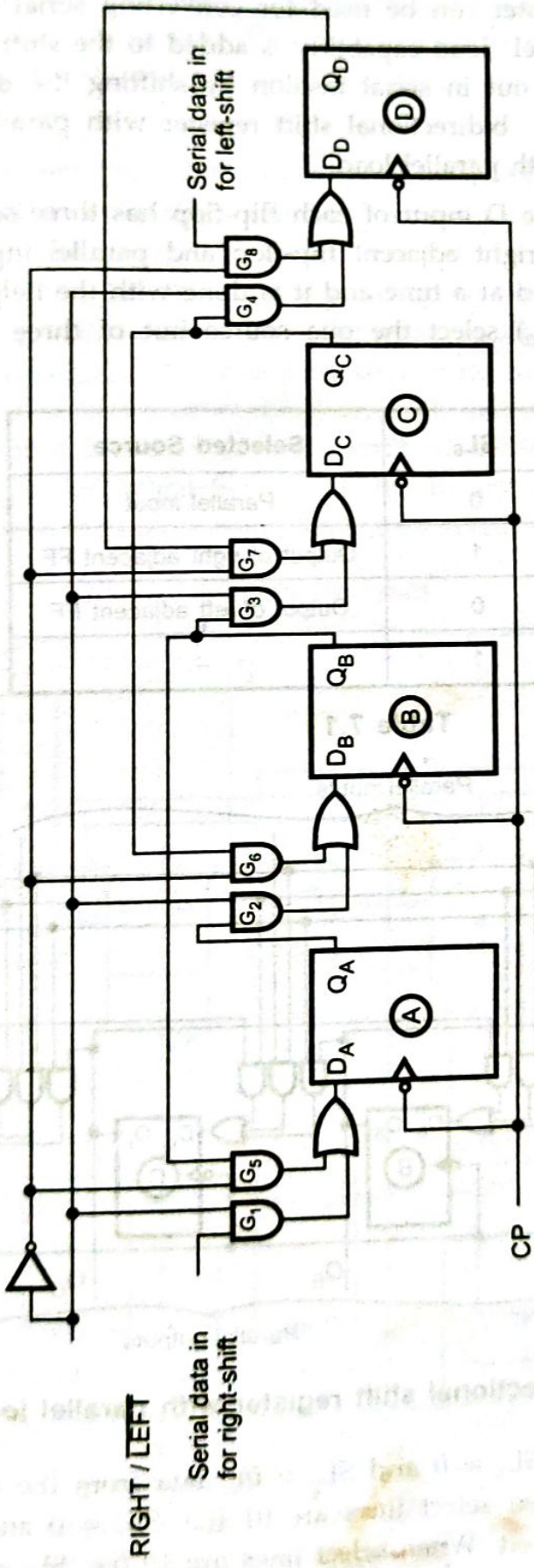


Fig. 7.11 4-bit bidirectional shift register

7.4 Universal Shift Register

A register capable of shifting in one direction only is a undirectional shift register. A register capable of shifting in both directions is a bidirectional shift register. If the register has both shifts (right shift and left shift) and parallel load capabilities, it is referred to as **Universal shift register.**

The Fig. 7.13 shows the 4 bit universal shift register. It has all the capabilities listed above. It consists of four flip-flops and four multiplexers. The four multiplexers have two common selection inputs S_1 and S_0, and they select appropriate input for D flip-flop. The Table 7.2 shows the register operation depending on the selection inputs of multiplexers. When $S_1S_0 = 00$, input 0 is selected and the present value of the register is applied to the D inputs of the flip-flops. This results no change in the register value. When $S_1S_0 = 01$, input 1 is selected and circuit connections are such that it operates as a right shift register. When $S_1S_0 = 10$, input 2 is selected and circuit connections are such that it operates as a left shift register. Finally, when $S_1S_0 = 11$, the binary information on the parallel input lines is transfered into the register simultaneously and it is a parallel load operation.

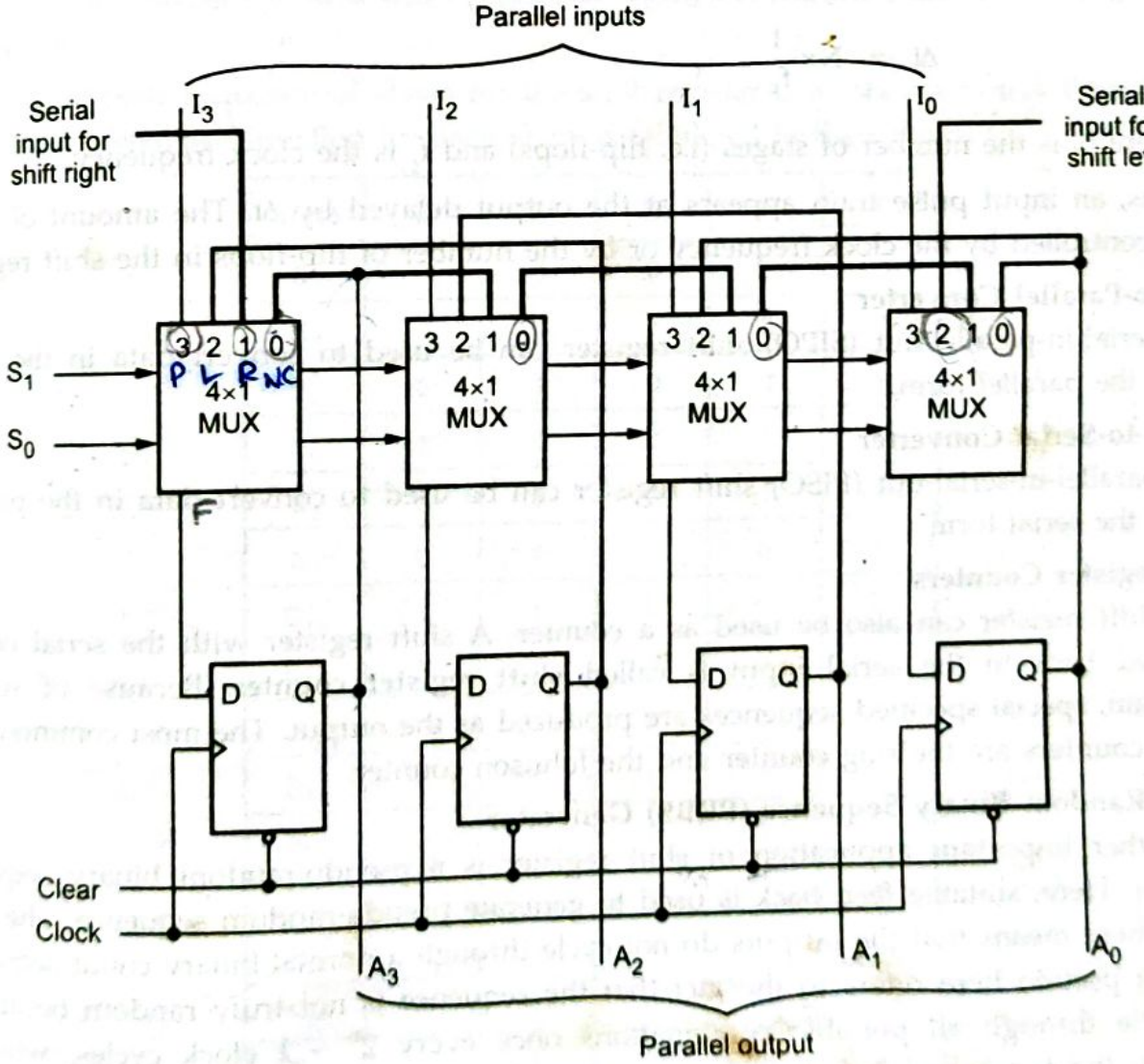


Fig. 7.13 4-Bit universal shift register

Mode Control		Register operation
S_1	S_0	
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

Table 7.2 Mode control and register operation

7.5 Applications of Shift Registers

We have seen that primary use of shift register is temporary data storage and b manipulations. Some of the common applications of shift registers are as discussed below

Delay Line

A serial-in-serial-out (SISO) shift register can be used to introduce time delay Δt digital signals. The time delay can be given as

$$\Delta t = N \times \frac{1}{f_c}$$

where N is the number of stages (i.e. flip-flops) and f_c is the clock frequency.

Thus, an input pulse train appears at the output delayed by Δt. The amount of dela can be controlled by the clock frequency or by the number of flip-flops in the shift registe

Serial-to-Parallel Converter

A serial-in-parallel-out (SIPO) shift register can be used to convert data in the seria form to the parallel form.

Parallel-to-Serial Converter

A parallel-in-serial-out (PISO) shift register can be used to convert data in the paralle form to the serial form.

Shift Register Counters

A shift register can also be used as a counter. A shift register with the serial outpu connected back to the serial input is called **shift register counter**. Because of such connection, special specified sequences are produced as the output. The most common shi register counters are the ring counter and the Johnson counter.

Pseudo-Random Binary Sequence (PRBS) Generator

Another important application of shift register is a pseudo-random binary sequenc generator. Here, suitable feed back is used to generate pseudo-random sequence. The ter random here means that the outputs do not cycle through a normal binary count sequence The term pseudo here refers to the fact that the sequence is not truly random because i does cycle through all possible combinations once every $2^n - 1$ clock cycles, where represents the number of shift register stages (number of flip-flops).

8

Counters

8.1 Introduction

A register is used solely for storing and shifting data which is in the form of 1s and/or 0s, entered from an external source. It has no specific sequence of states except in certain very specialized applications. A counter is a register capable of counting the number of clock pulses arriving at its clock input. Count represents the number of clock pulses arrived. A specified sequence of states appears as the counter output. This is the main difference between a register and a counter. A specified sequence of states is different for different types of counters.

There are two types of counters, synchronous and asynchronous. In synchronous counter, the common clock input is connected to all of the flip-flops and thus they are clocked simultaneously. In asynchronous counter, commonly called, ripple counters, the first flip-flop is clocked by the external clock pulse and then each successive flip-flop is clocked by the Q or $\overline{Q}$ output of the previous flip-flop. Therefore in an asynchronous counter, the flip-flops are not clocked simultaneously. Let us start with asynchronous counters.

8.2 Binary Asynchronous / Ripple Counters

Fig. 8.1 shows 2-bit asynchronous counter using JK flip-flops. As shown in Fig. 8.1, the clock signal is connected to the clock input of only first stage flip-flop. The clock input of the second stage flip-flop is triggered by the $\overline{Q}_A$ output of the first stage. Because of the inherent propagation delay time through a flip-flop, a transition of the input clock pulse and a transition of the $\overline{Q}_A$ output of first stage can never occur at exactly the same time. Therefore, the two flip-flops are never simultaneously triggered, which reults in asynchronous counter operation.

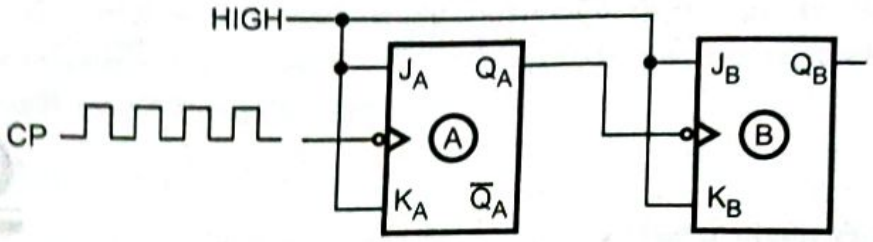


Fig. 8.1 A two-bit asynchronous binary counter

Fig. 8.2 shows the timing diagram for two-bit asynchronous counter. It illustrates the changes in the state of the flip-flop outputs in response to the clock.

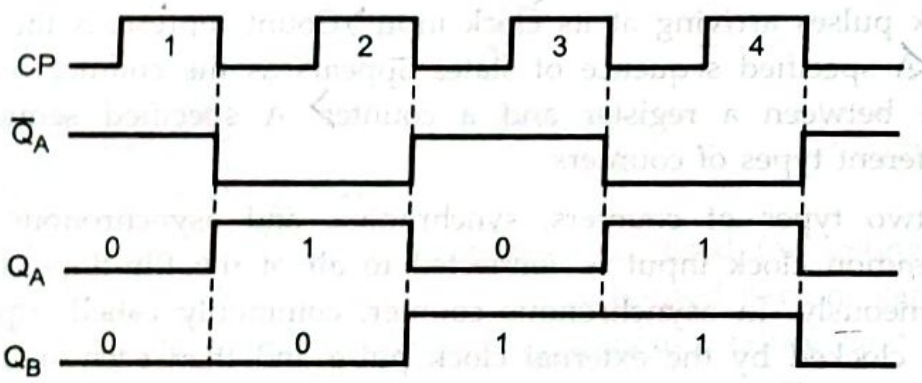


Fig. 8.2 Timing diagram for the counter of Fig. 8.2

Example 8.1 : *Extend the counter shown in Fig. 8.1 for 3-stages, and draw output waveforms.*

Solution :

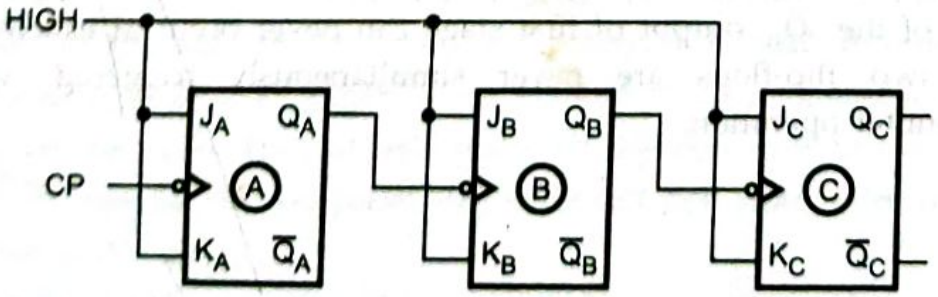


Fig. 8.3 (a) Logic Diagram

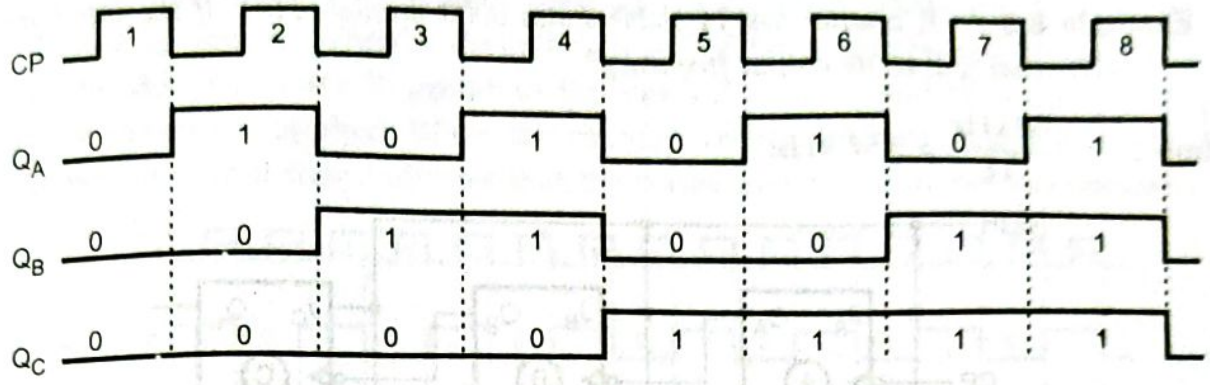


Fig. 8.3 (b) Output waveforms for 3-bit asynchronous counter

In Fig. 8.3 (b), timing diagram for 3-bit asynchronous counter we have not considered the propagation delays of flip-flops, for simplicity. If we consider the propagation delays of flip-flops we get timing diagram as shown in Fig. 8.4.

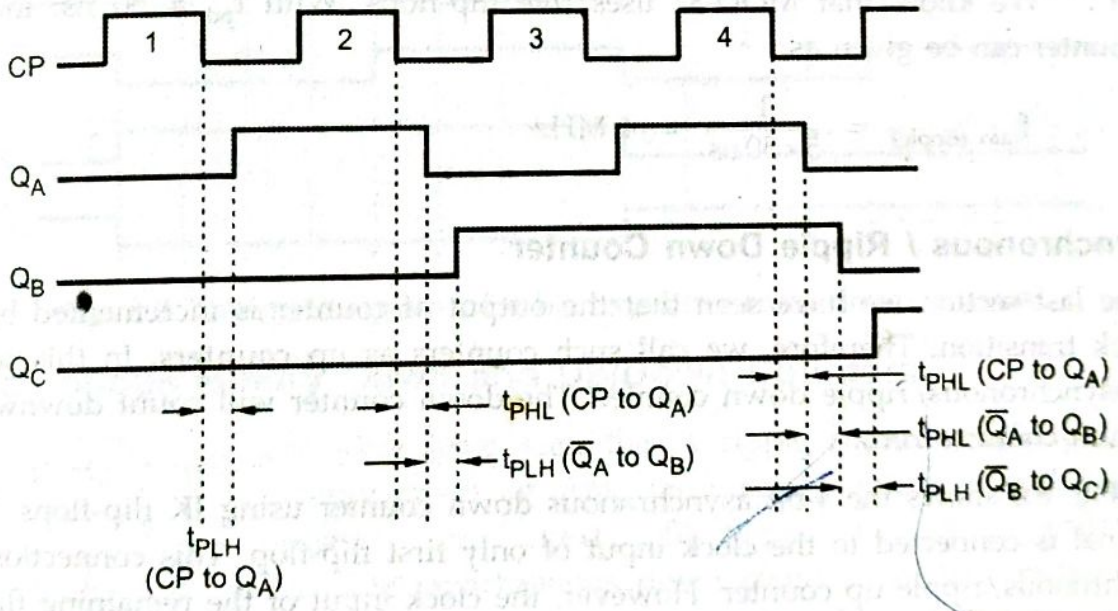


Fig. 8.4 Propagation delays in a ripple clocked binary counter

The timing diagram shows propagation delays. We can see that propagation delay of the first stage is added in the propagation delay of second stage to decide the transition time for third stage. This cumulative delay of an asynchronous counter is a major disadvantage in many applications because it limits the rate at which the counter can be clocked and creates decoding problems.

Example 8.2 : *Draw the logic diagram for 3-stage asynchronous counter with negative edge triggered flip-flops.*

Solution : When flip-flops are negatively edge triggered. The Q output of previous stage is connected to the clock input of the next stage. Fig. 8.5 shows 3-stage asynchronous counter with negative edge triggered flip-flops.

Example 8.3 : *A counter has 14 stable states 0000 through 1101. If the input frequency is 50 kHz what will be its output frequency?*

Solution : $\frac{50\,\text{kHz}}{14} = 3.57\ \text{kHz}$

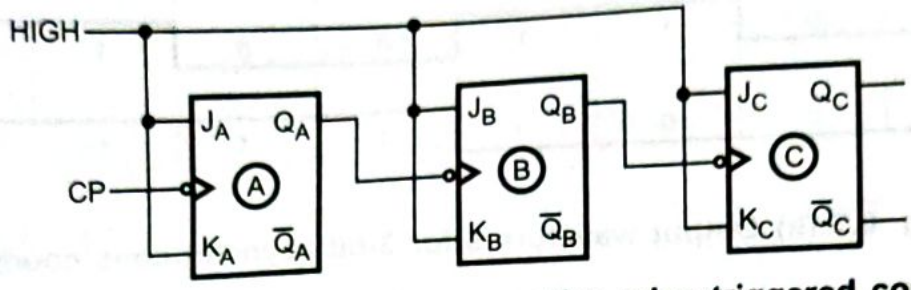


Fig. 8.5 Logic diagram of 3-stage negative edge triggered counter

Example 8.4 : *The t_{pd} for each flip-flop is 50 ns , determine the maximum operating frequency for MOD-32 ripple counter.*

Solution : We know that MOD-32 uses five flip-flops. With t_{pd} = 50 ns, the f_{max} for ripple counter can be given as,

$$f_{max\ (ripple)} = \frac{1}{5 \times 50\,\text{ns}} = 4\ \text{MHz}$$

8.3 Asynchronous / Ripple Down Counter

In the last section we have seen that the output of counter is incremented by one for each clock transition. Therefore, we call such counters as up **counters.** In this section we see the asynchronous/ripple down counter. The down counter will count downward from a maximum count to zero.

The Fig. 8.6 shows the 4-bit asynchronous down counter using JK flip-flops. Here, the clock signal is connected to the clock input of only first flip-flop. This connection is same as asynchronous/ripple up counter. However, the clock input of the remaining flip-flops is triggered by the $\overline{Q}_A$ output of the previous stage instead of Q_A output of the previous stage.

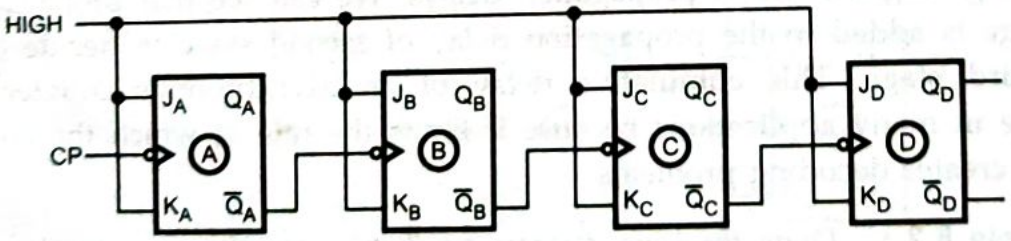


Fig. 8.6 4-bit asynchronous down counter

The Fig. 8.7 shows the timing diagram for 4-bit asynchronous down counter. It illustrates the changes in the state of the flip-flop outputs in response to the clock. Again the J and K inputs of JK flip-flops are tied to logic HIGH hence output will toggle for each negative edge of the clock input.

Down counters are not as widely used as up counters. They are used in situation where it must be known when a desired number of input pulses has occurred. In these situations the down counter is preset to the desired number and then allowed to count down as the pulses are applied. When the counter reaches the zero state it is detected by a logic gate whose output then indicates that the preset number of pulses has occurred.

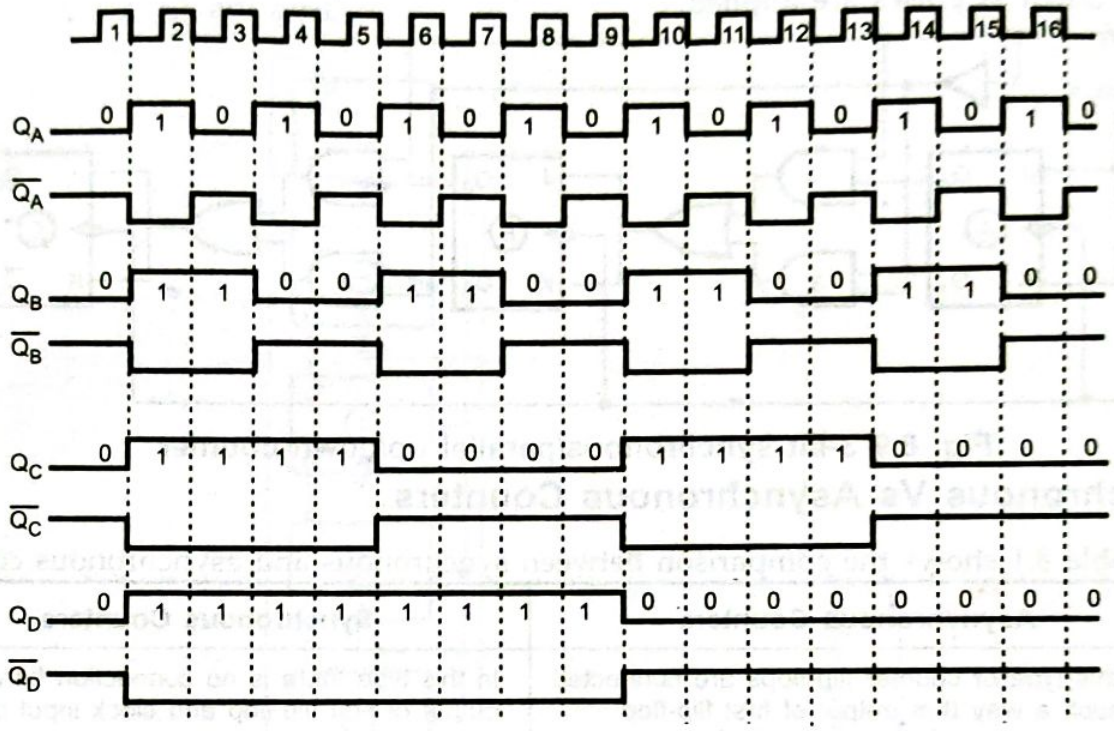


Fig. 8.7 Timing diagram of 4-bit asynchronous down counter

8.4 Synchronous Binary Down and Up/Down Counters

1	1	1	1
1	1	1	0
1	1	0	1
1	1	0	0
.	.	.	.
.	.	.	.
.	.	.	.
0	0	1	1
0	0	1	0
0	0	0	1
0	0	0	0

Fig. 8.8

We have seen that a ripple counter could be made to count down by using the inverted output of each flip-flop to drive the next flip-flops in the counter. A parallel/synchronous down counter can be constructed in a similar manner-that is, by using the inverted FF outputs to drive the following JK inputs. For example, the parallel up counter of Fig. 8.8 can be converted to a down counter by connecting the $\overline{Q}_A$, $\overline{Q}_B$, $\overline{Q}_C$ and $\overline{Q}_D$ outputs in place of Q_A, Q_B, Q_C and Q_D respectively. The counter will then proceed through the following sequence as input pulses are applied :

To form a parallel up/down counter the control input (UP/$\overline{\text{DOWN}}$) is used to control whether the normal flip-flop outputs or the inverted flip-flop outputs are fed to the J and K inputs of the following flip-flops. The Fig. 8.8 shows 3-bit up/down counter that will count from 000 up to 111 when the up/$\overline{\text{Down}}$ control input is 1 and from 111 down to 000 when the up/$\overline{\text{Down}}$ control input is 0.

A logic 1 on the Up/$\overline{\text{Down}}$ enables AND gates 1 and 2 and disables AND gates 3 and 4. This allows the Q_A and Q_B outputs through to the J and K inputs of the next flip-flops so that the counter will count up as pulses are applied. When Up/$\overline{\text{Down}}$ line is logic 0, AND gates 1 and 2 are disables and AND gates 3 and 4 are enabled. This allows the $\overline{Q_A}$ and $\overline{Q_B}$ outputs through to the J and K inputs of the next flip-flops so that the counter will count down as pulses are applied.

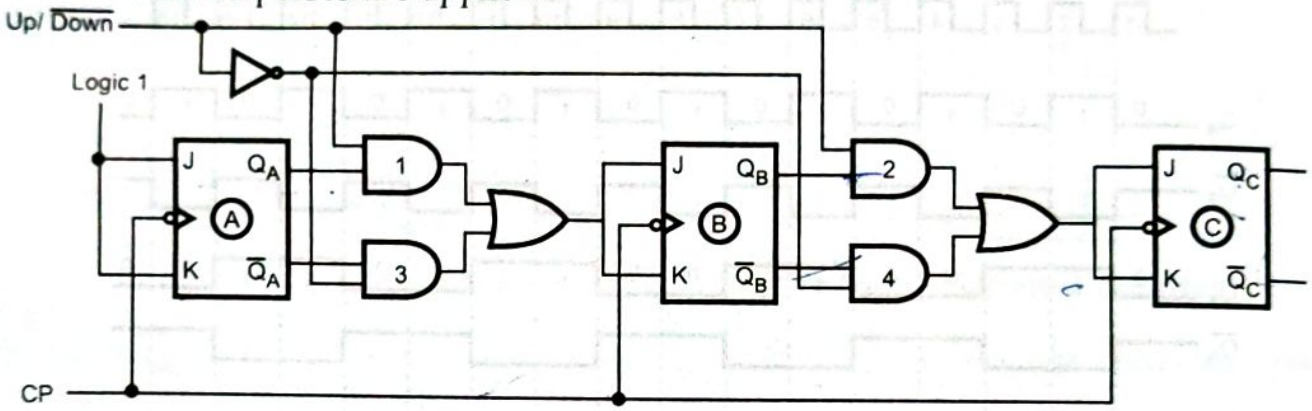


Fig. 8.9 3-bit synchronous/parallel up/down counter

8.5 Synchronous Vs Asynchronous Counters

The Table 8.1 shows the comparison between synchronous and asynchronous counters.

	Asynchronous Counters	Synchronous Counters
1)	In this type of counter flip-flops are connected in such a way that output of first flip-flop drives the clock for the next flip-flop.	In this type there is no connection between output of first flip-flop and clock input of the next flip-flop.
2)	All the flip-flops are not clocked simultaneously.	All the flip-flops are clocked simultaneously.
3)	Logic circuit is very simple even for more number of states.	Design involves complex logic circuit as number of states increases.
4)	Main drawback of these counters is their low speed as the clock is propogated through number of flip-flops before it reaches last flip-flop.	As clock is simultaneously given to all flip-flops there is no problem of propagation delay. Hence they are preferred when number of flip-flops increases in the given design.

Table 8.1 Comparision between synchronous and asynchronous counters

Before discussing the design procedures of synchronous counters of arbitrary modulo and of UP/$\overline{\text{DOWN}}$ counters we study the systematic technique of analysis and design of sequential circuits. The most commonly used techniques of analysis and design of sequential circuit are : state tables and flip-flop excitation tables.

8.6 Other Counters

8.6.1 Counters Based on Shift Registers

A shift register can also be used as a counter. A shift register with the serial output connected back to the serial input is called **shift register counter**. Because of such a connection, special specified sequences are produced as the output. The most common shift register counters are the ring counter and the Johnson counter.

8.6.1.1 Ring Counter

Fig. 8.10 shows the logic diagram for ten-bit ring counter. As shown in the Fig. 8.10, the Q output of each stage is connected to the D input of the next stage and the output of last stage is fed back to the input of first stage. The $\overline{CLR}$ followed by $\overline{PRE}$ makes the output of first stage to '1' and remaining outputs are zero, i.e. Q_A is one and Q_B, Q_C, Q_D, Q_E, Q_F, Q_G, Q_H, Q_I, Q_J are zero.

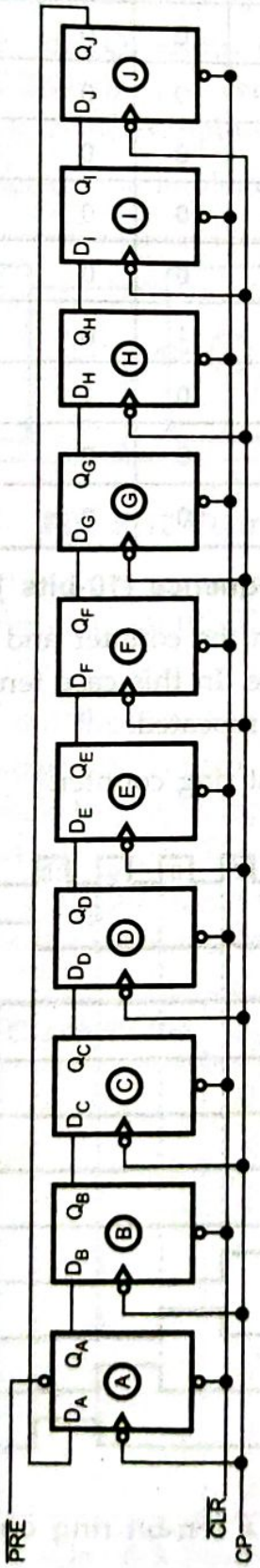


Fig. 8.10 A ten bit ring counter

The first clock pulse produces $Q_B = 1$ and remaining outputs are zero. According to the clock pulses applied at the clock input CP, a sequence of ten states is produced. These states are summarized in Table 8.1.

Clock Pulse	Q_A	Q_B	Q_C	Q_D	Q_E	Q_F	Q_G	Q_H	Q_I	Q_J
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	0	1

Table 8.2 Ring-counter sequence (10-bits)

As shown in Table 8.2, 1 is always retained in the counter and simply shifted 'around the ring', advancing one stage for each clock pulse. In this case ten stages of flip-flops are used. So a sequence of ten states is produced and repeated.

Fig. 8.11 gives the timing sequence for a ten-bit ring counter.

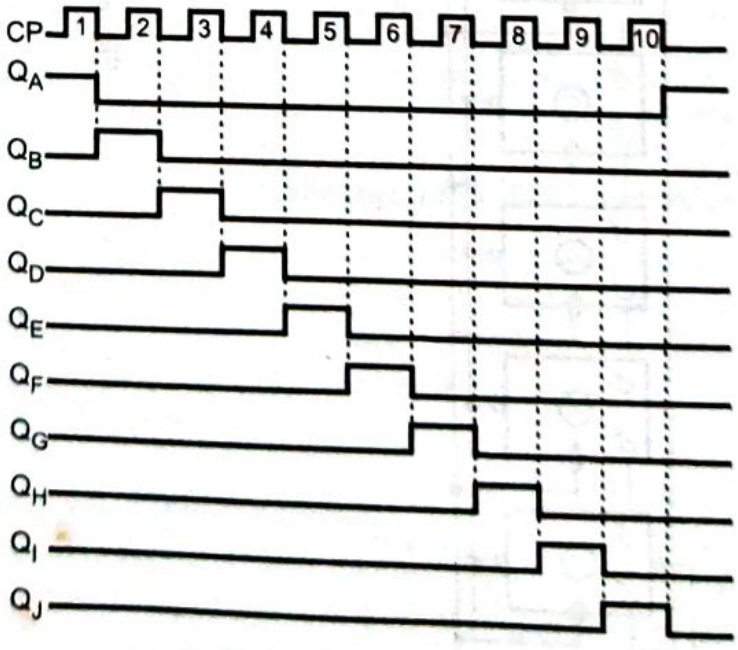


Fig. 8.11 Timing sequence for a ten-bit ring counter

The ring counter can be used for counting the number of pulses. The number of pulses counted is read by noting which flip-flop is in state 1. No decoding circuitry is required. Since there is one pulse at the output for each of the Nclock pulses, this circuit is also referred to as a divide-by-N-counter or an N : 1 scalar. Ring counters can be instructed for any desired MOD number, that is MOD N ring counter requires N flip-flops.

8.6.1.2 The Johnson Counter or Twisted Ring Counter

In a Johnson counter, the Q output of each stage of flip-flop is connected to the D input of the next stage. The single exception is that the complement output of the last flip-flop is connected back to the D-input of the first flip-flop as shown in Fig. 8.12.

Note : Johnson counter can be implemented with SR or JK flip-flops as well.

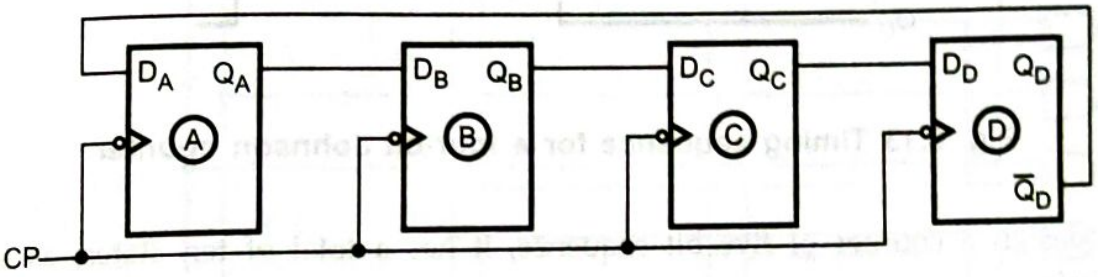


Fig. 8.12 Four-bit Johnson counter

As shown in Fig. 8.12, there is a feedback from the rightmost flip-flop complement output to the leftmost flip-flop input. This arrangement produces a unique sequence of states.

Initially, the register (all flip-flops) is cleared. So all the outputs, Q_A, Q_B, Q_C, Q_D are zero. The output of last stage, Q_D is zero. Therefore complement output of last stage, $\overline{Q}_D$ is one. This is connected back to the D input of first stage. So D_A is one. The first falling clock edge produces Q_A=1 and Q_B = 0,Q_C = 0, Q_D = 0 since D_B, D_C, D_D are zero. The next clock pulse produces Q_A=1 ,Q_B =1, Q_C = 0, Q_D = 0. The sequence of states is summarized in Table 8.3. After 8 states the same sequence is repeated.

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

Table 8.3 Four-bit Johnson sequence

In this case, four bit register is used. So the four bit sequence has a total of eig states. Fig. 8.13 gives the timing sequence for a four-bit Johnson counter.

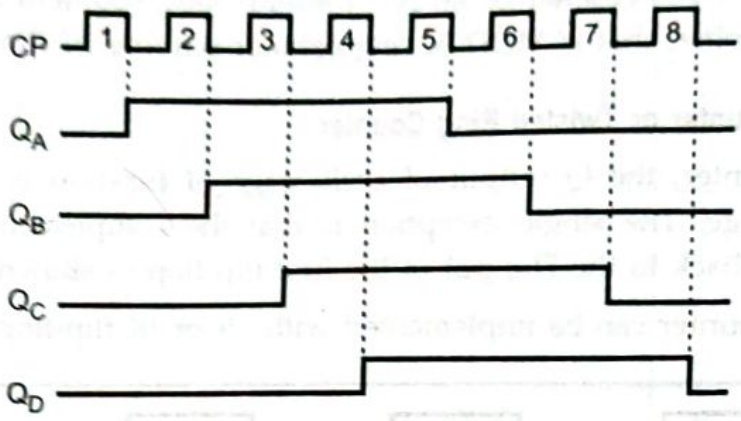


Fig. 8.13 Timing sequence for a four-bit Johnson counter

If we design a counter of five bit sequence, it has a total of ten states, as shown in Table 8.4 below.

Clock Pulse	Q_A	Q_B	Q_C	Q_D	Q_E
0	0	0	0	0	0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	1	0
5	1	1	1	1	1
6	0	1	1	1	1
7	0	0	1	1	1
8	0	0	0	1	1
9	0	0	0	0	1

Table 8.4 Five-bit Johnson sequence

So in general we can say that, an n-stage Johnson counter will produce a modulus of $2 \times n$, where n is the number of stages (i.e. flip-flops) in the counter. As shown in tables, the counter will 'fill up' with 1s from left to right and then it will 'fill up' with 0s again.One advantage of this type of sequence is that it is readily decoded with two input AND gates. Table 8.5 gives the count sequence and required decoding.

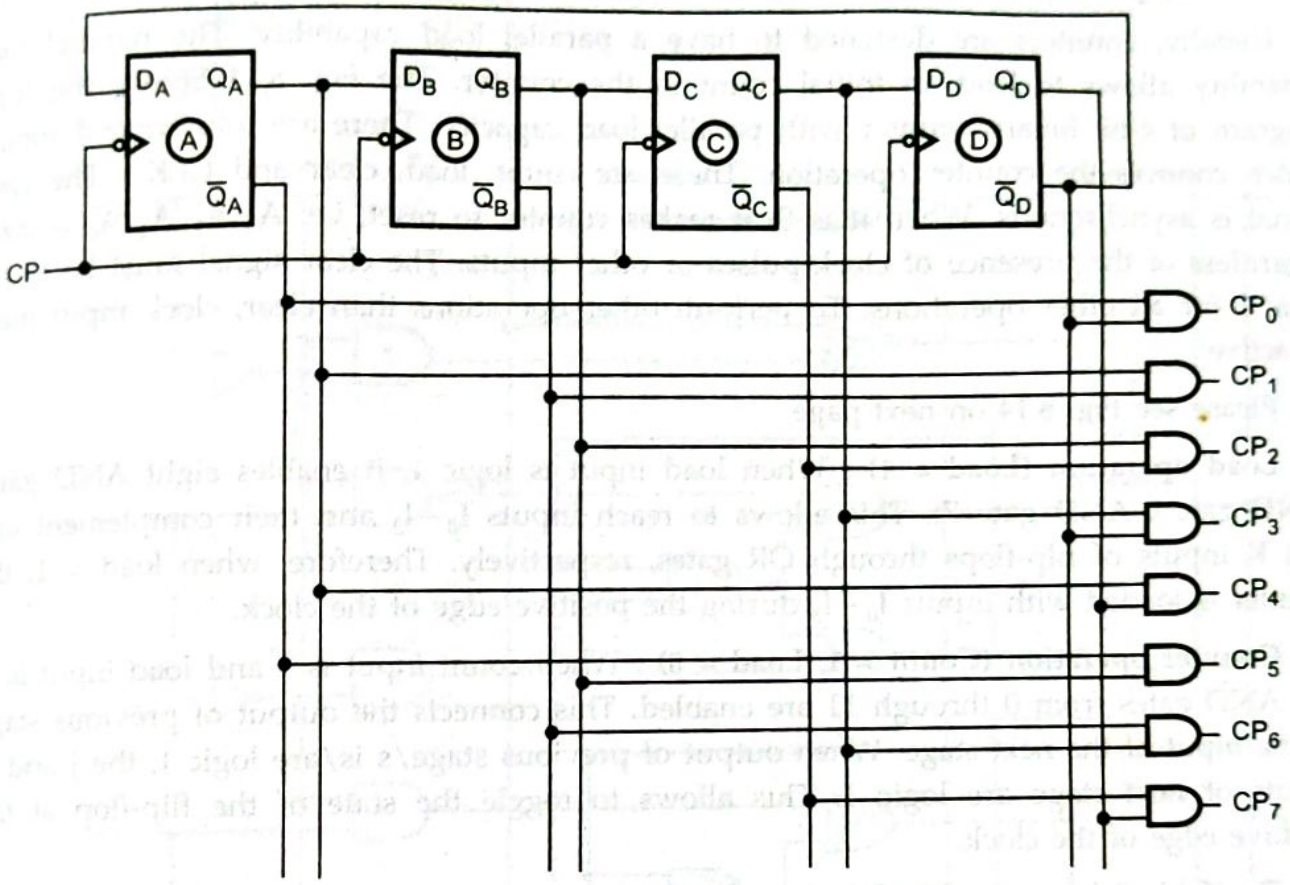


Fig. 8.13 (a) Johnson counter with decoder

Clock Pulse	Q_A	Q_B	Q_C	Q_D	AND Gate required for output
0	0	0	0	0	$\overline{Q}_A \overline{Q}_D$
1	1	0	0	0	$Q_A \overline{Q}_B$
2	1	1	0	0	$Q_B \overline{Q}_C$
3	1	1	1	0	$Q_C \overline{Q}_D$
4	1	1	1	1	$Q_A Q_D$
5	0	1	1	1	$\overline{Q}_A Q_B$
6	0	0	1	1	$\overline{Q}_B Q_C$
7	0	0	0	1	$\overline{Q}_C Q_D$

Table 8.5 Count sequence and required decoding