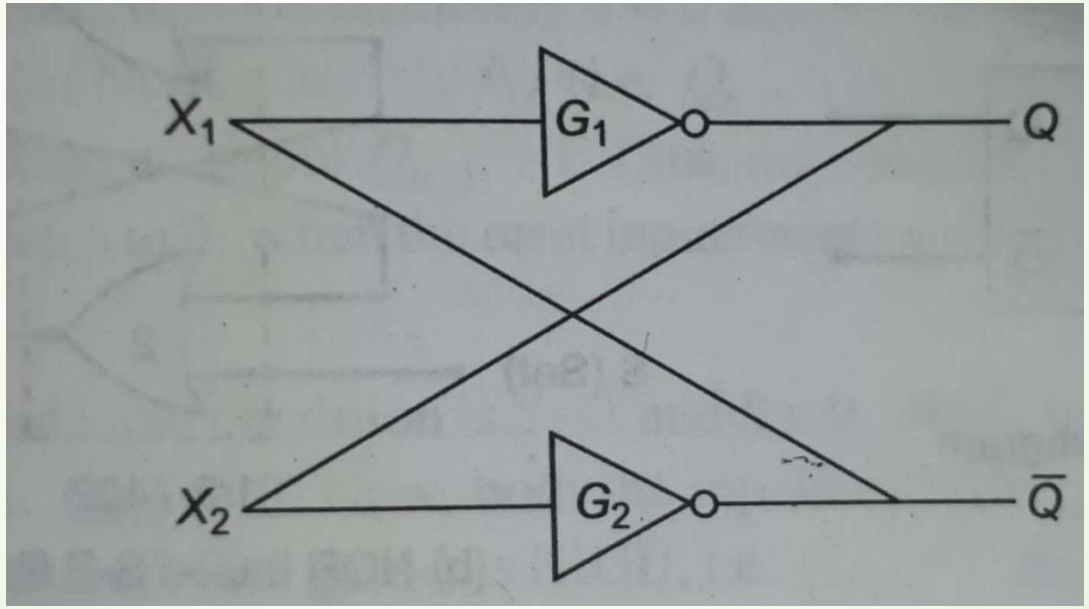
Sequential Circuits

- The logic circuits whose output at any instant of time depend not only on the present inputs but also on the past outputs are called sequential circuits.
- In sequential circuits the output signals are fed back to the input side.
- The output signal is a function of the present input signals and a sequence of the past input signals ie. the past output signals.
- Combinational circuits are often faster than sequential circuits since the combinational circuits do not require memory elements whereas the sequential circuits need memory devices to perform their operations in sequence.
- Sequential circuits are of two types:
    - Synchronous or clocked
    - Asynchronous or unclocked

By : Ms. Kanika Dhingra

Latches

- Simplest kind of a sequential circuit has only two states.
- It is a memory cell which is capable of storing one bit of information ie. Logic 1 or 0.
- This sequential circuit is called a latch, since one bit of information can be locked or latched.
- Basic latch consists of two inverters.



S-R Latch (NOR – based S R Latch)

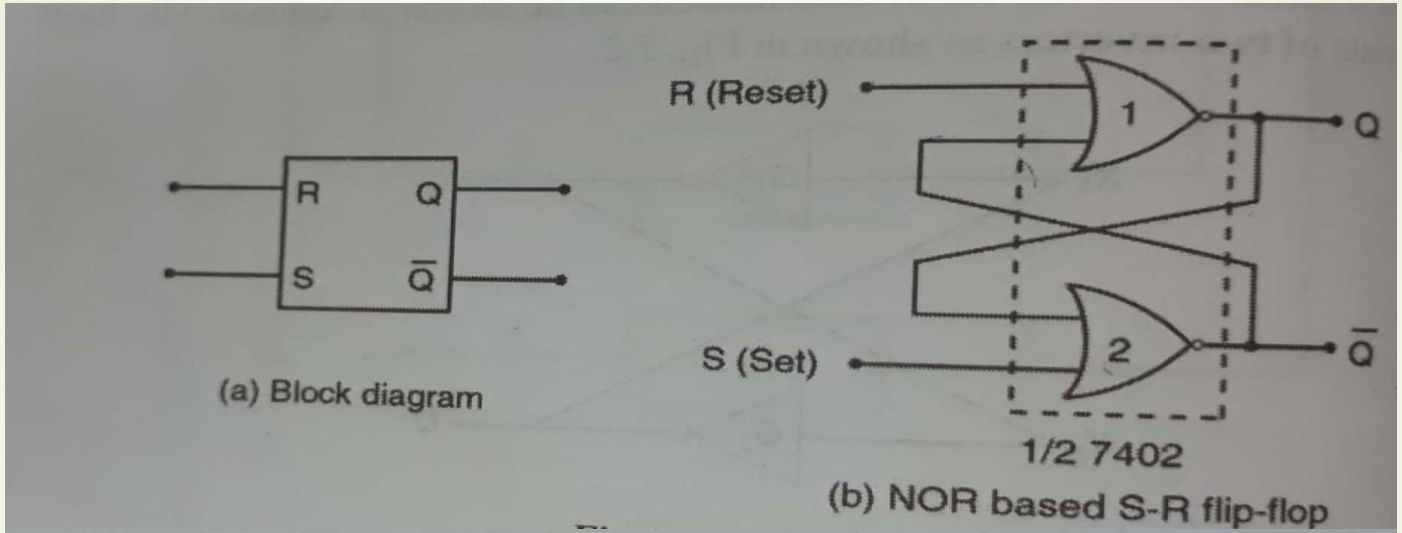


Table 7.1 Truth table of NOR-based S-R latch

Inputs		Outputs		Action
S	R	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	Q_n	$\overline{Q}_n$	No change
0	1	0	1	Reset
1	0	1	0	Set
1	1	?	?	Forbidden

By : Ms. Kanika Dhingra

NAND – based $\overline{S}$ – $\overline{R}$ Latch

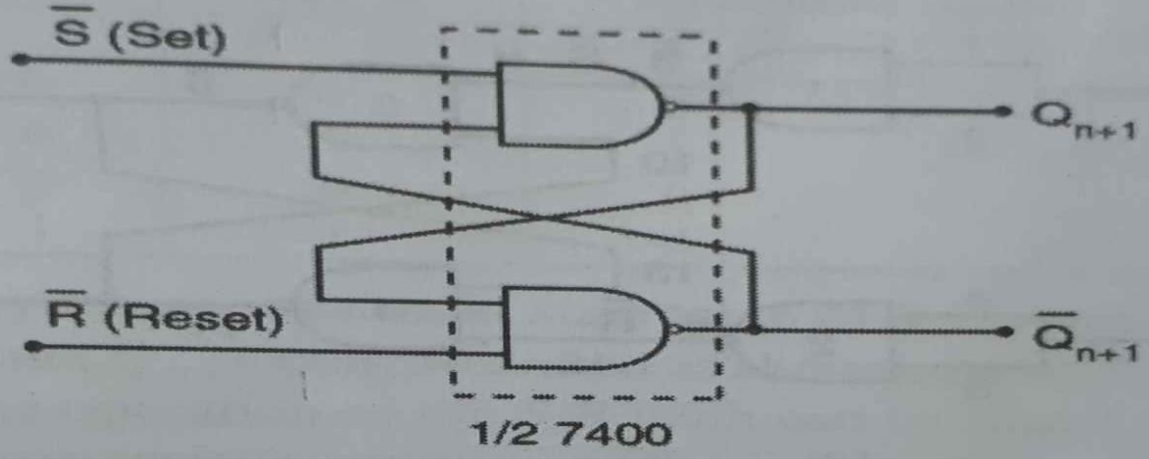


Fig. 7.5 NAND-based $\overline{S} - \overline{R}$ latch

***Table 7.2** Truth table of NAND-based $\overline{S} - \overline{R}$ latch*

Inputs		Outputs		Action
$\overline{S}$	$\overline{R}$	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	?	?	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q_n	$\overline{Q}_n$	No change

By : Ms. Kanika Dhingra

Flip Flops

- Flip-flop is a basic digital memory circuit, which stores one bit of information.
- Flip flops are the fundamental blocks of most sequential circuits. It is also known as a bistable multivibrator or a binary or one-bit memory.
- Flip-flops are used as memory elements in sequential circuit.
- The state of flip-flop changes at active state of clock pulses and remains unaffected when the clock pulse is not active.
- In particular, clocked flip flops serve as memory elements in synchronous sequential Circuits and unclocked flip-flops (i.e., latches) serve as memory elements in asynchronous sequential circuits.

By : Ms. Kanika Dhingra

SR Flip Flop

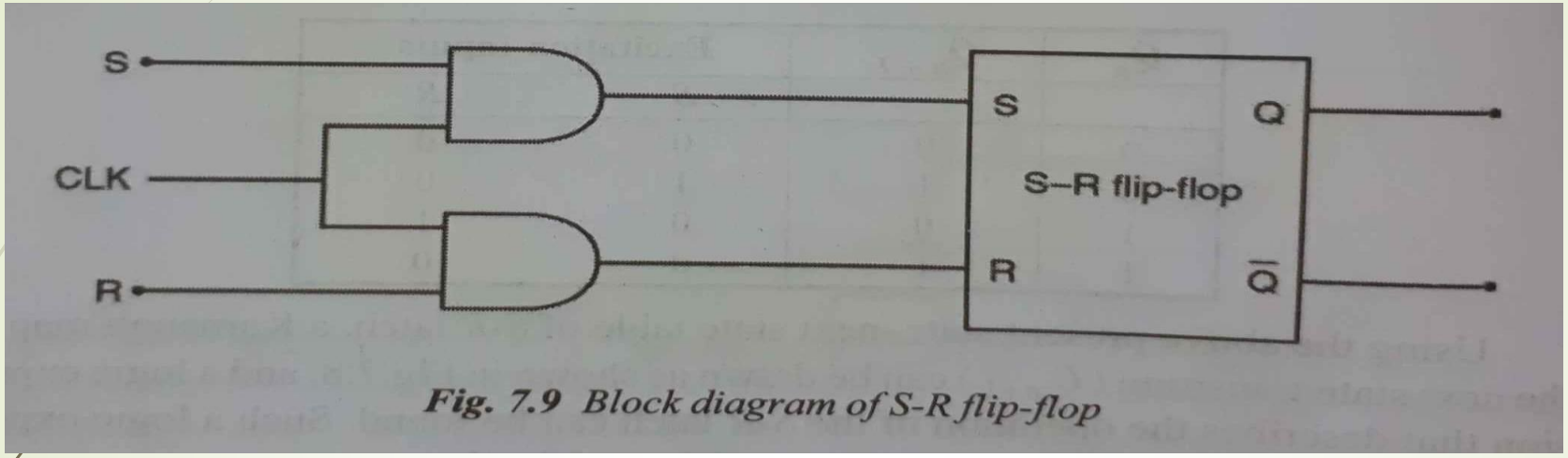


Fig. 7.9 Block diagram of S-R flip-flop

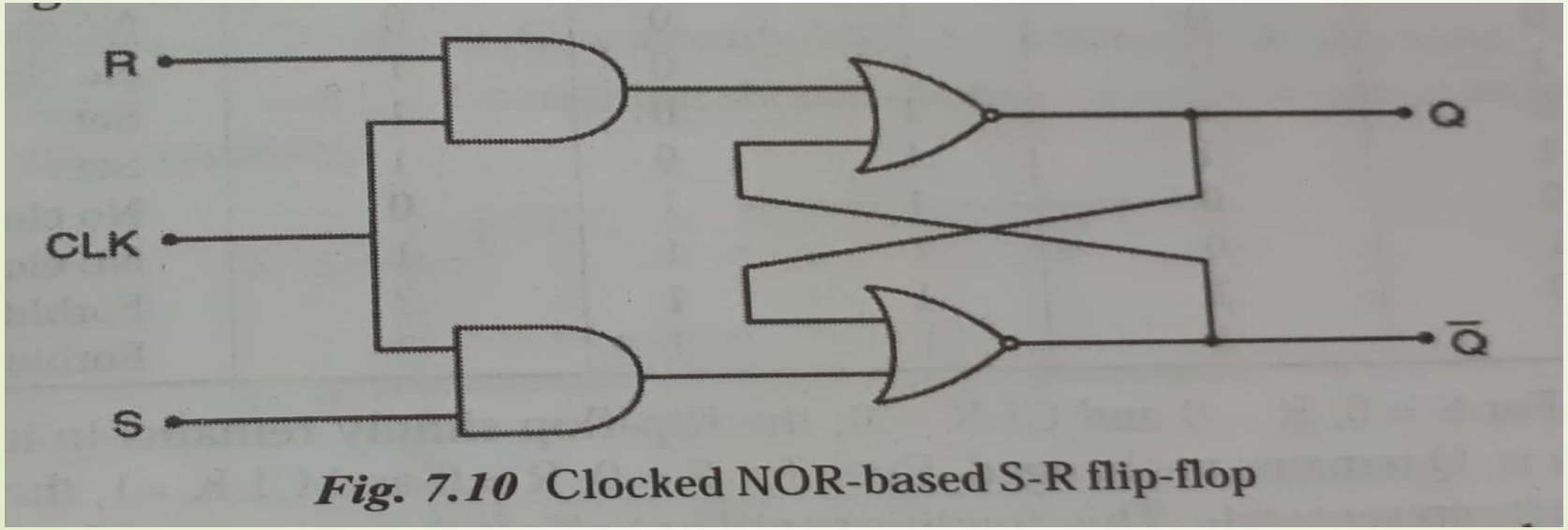
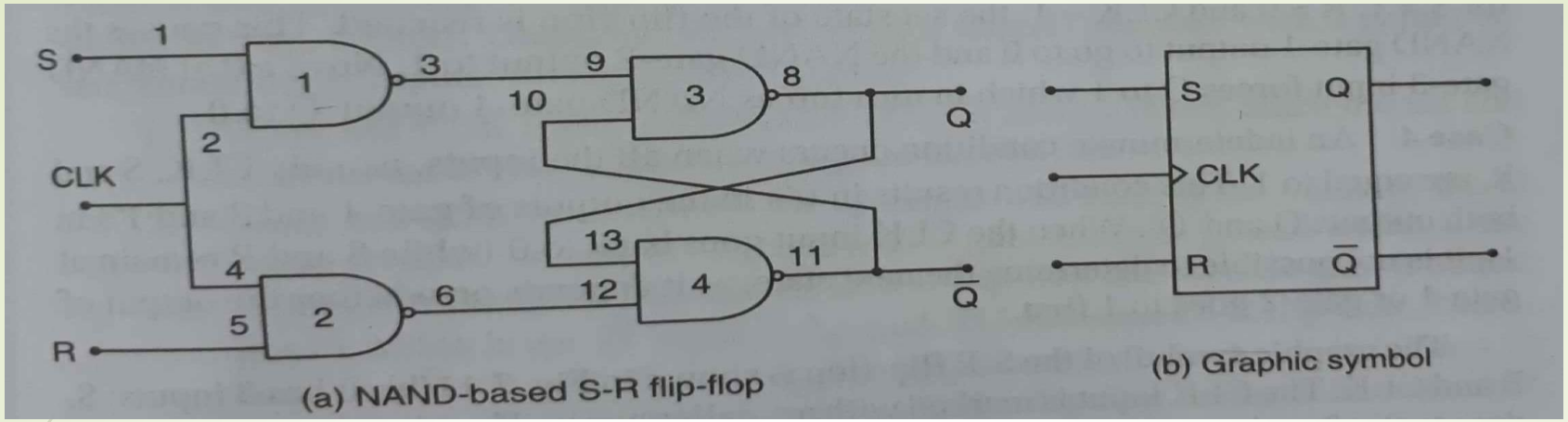


Fig. 7.10 Clocked NOR-based S-R flip-flop

By : Ms. Kanika Dhingra



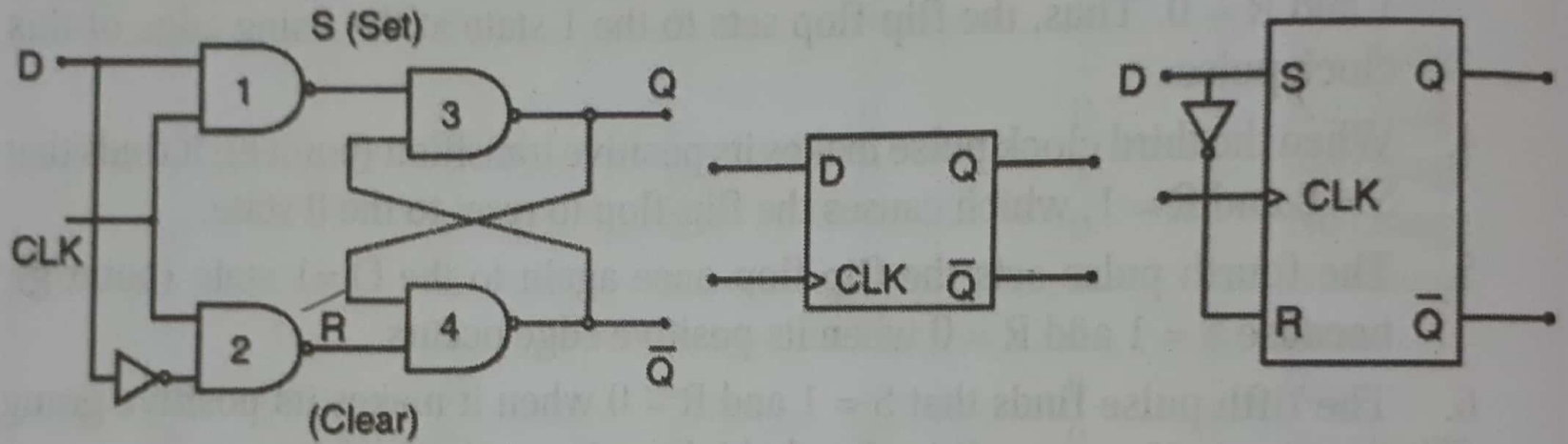
(a) NAND-based S-R flip-flop

(b) Graphic symbol

Table 7.5 Characteristic table of S-R flip-flop

Present state Q_n	Clock pulse CLK	Data inputs S	R	Next state Q_{n+1}	Action
0	0	0	0	0	No change
1	0	0	0	1	No change
0	1	0	0	0	No change
1	1	0	0	1	No change
0	0	0	1	0	No change
1	0	0	1	1	No change
0	1	0	1	0	Reset
1	1	0	1	0	Reset
0	0	1	0	0	No change
1	0	1	0	1	No change
0	1	1	0	1	Set
1	1	1	0	1	Set
0	0	1	1	0	No change
1	0	1	1	1	No change
0	1	1	1	?	Forbidden
1	1	1	1	?	Forbidden

D Flip Flop



(a) Using NAND gates (b) Logic symbol (c) Using S-R flip-flop

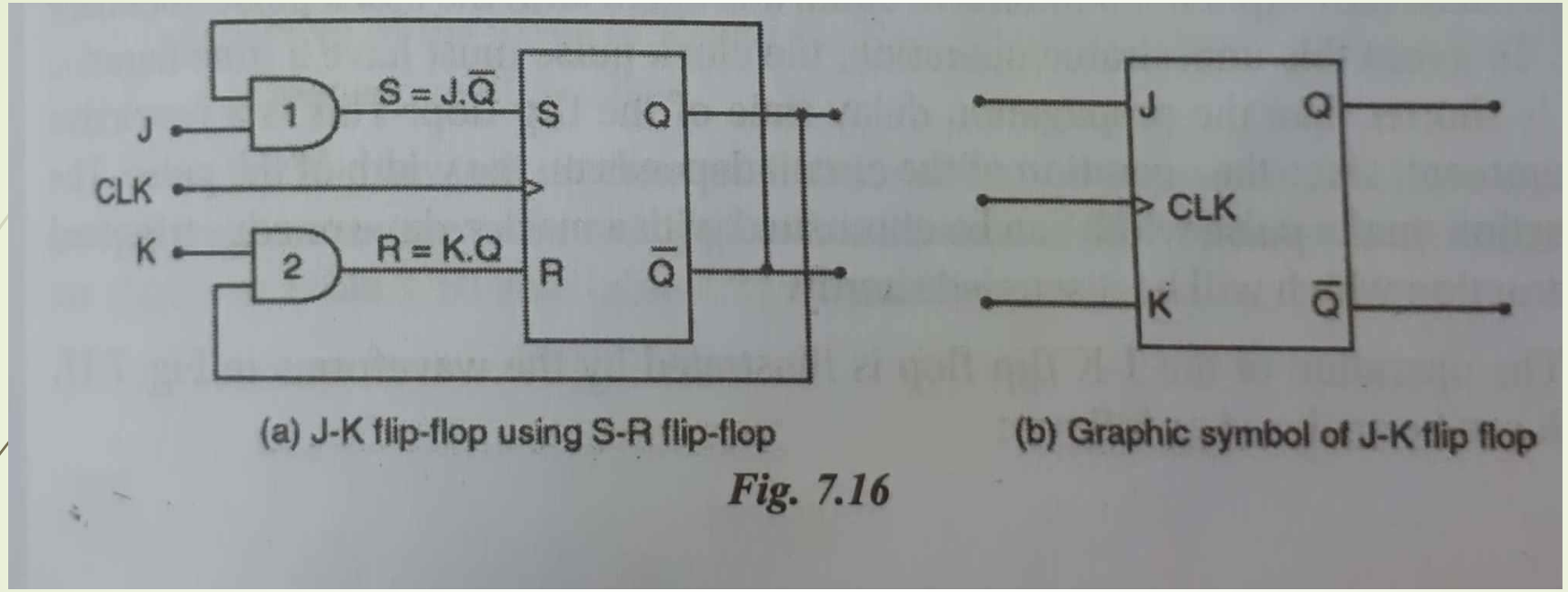
Fig. 7.13 D flip-flop

Table 7.6 Truth table of D flip-flop

CLK	Input *D*	Output Q_{n+1}
1	0	0
1	1	1
0	X	No change

By : Ms. Kanika Dhingra

JK Flip Flop



(a) J-K flip-flop using S-R flip-flop

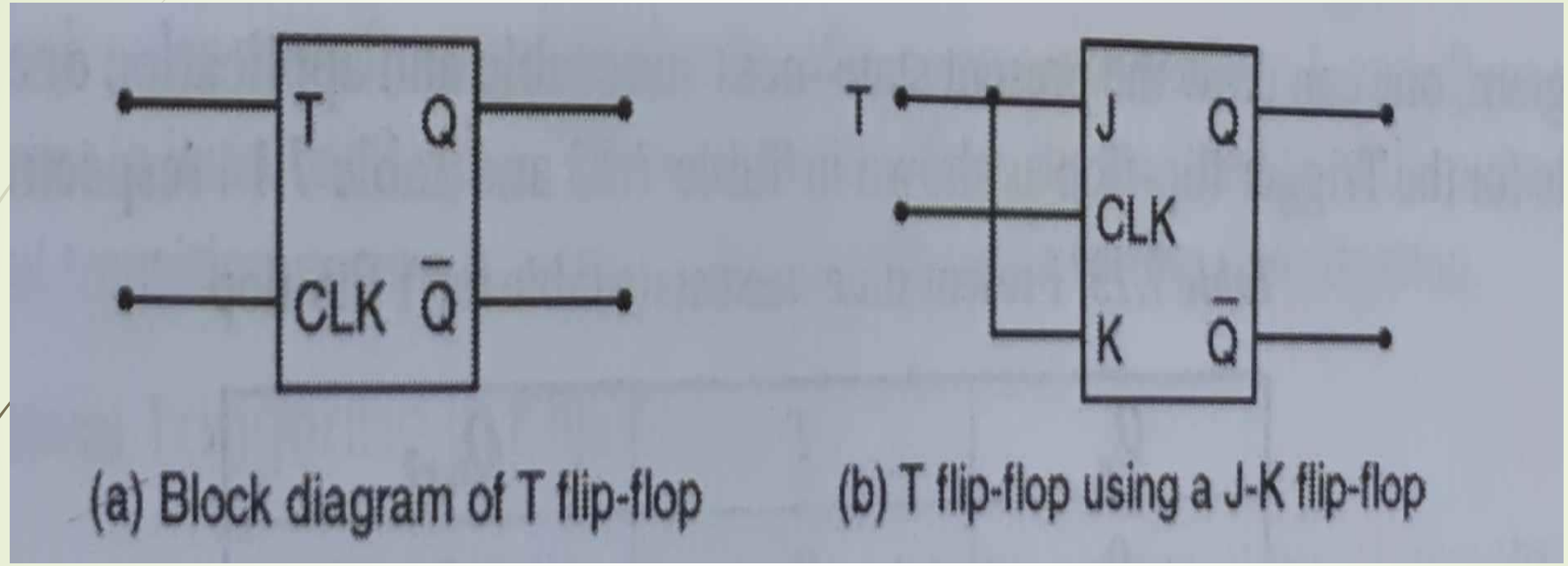
(b) Graphic symbol of J-K flip flop

Fig. 7.16

Table 7.9 Truth table of J-K flip-flop

CLK	Inputs		Output	
	J	*K*	Q_{n+1}	Action
X	0	0	Q_n	No change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	$\overline{Q}_n$	Toggle

T Flip Flop



(a) Block diagram of T flip-flop

(b) T flip-flop using a J-K flip-flop

Table 7.12 Truth table of T flip-flop

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

Master Slave JK Flip Flop

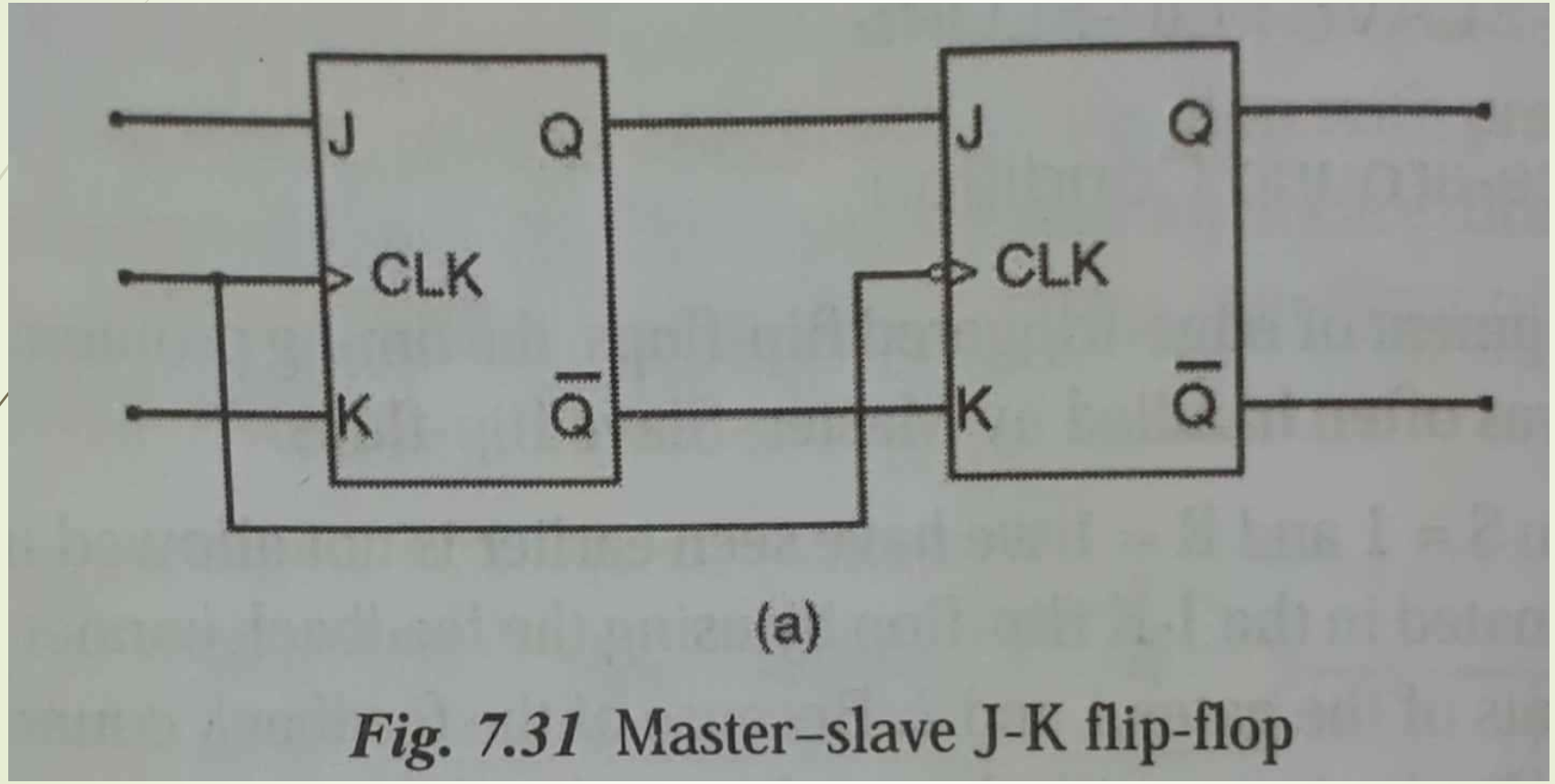


Fig. 7.31 Master–slave J-K flip-flop

REALISATION OF ONE FLIP-FLOP USING OTHER FLIP-FLOPS

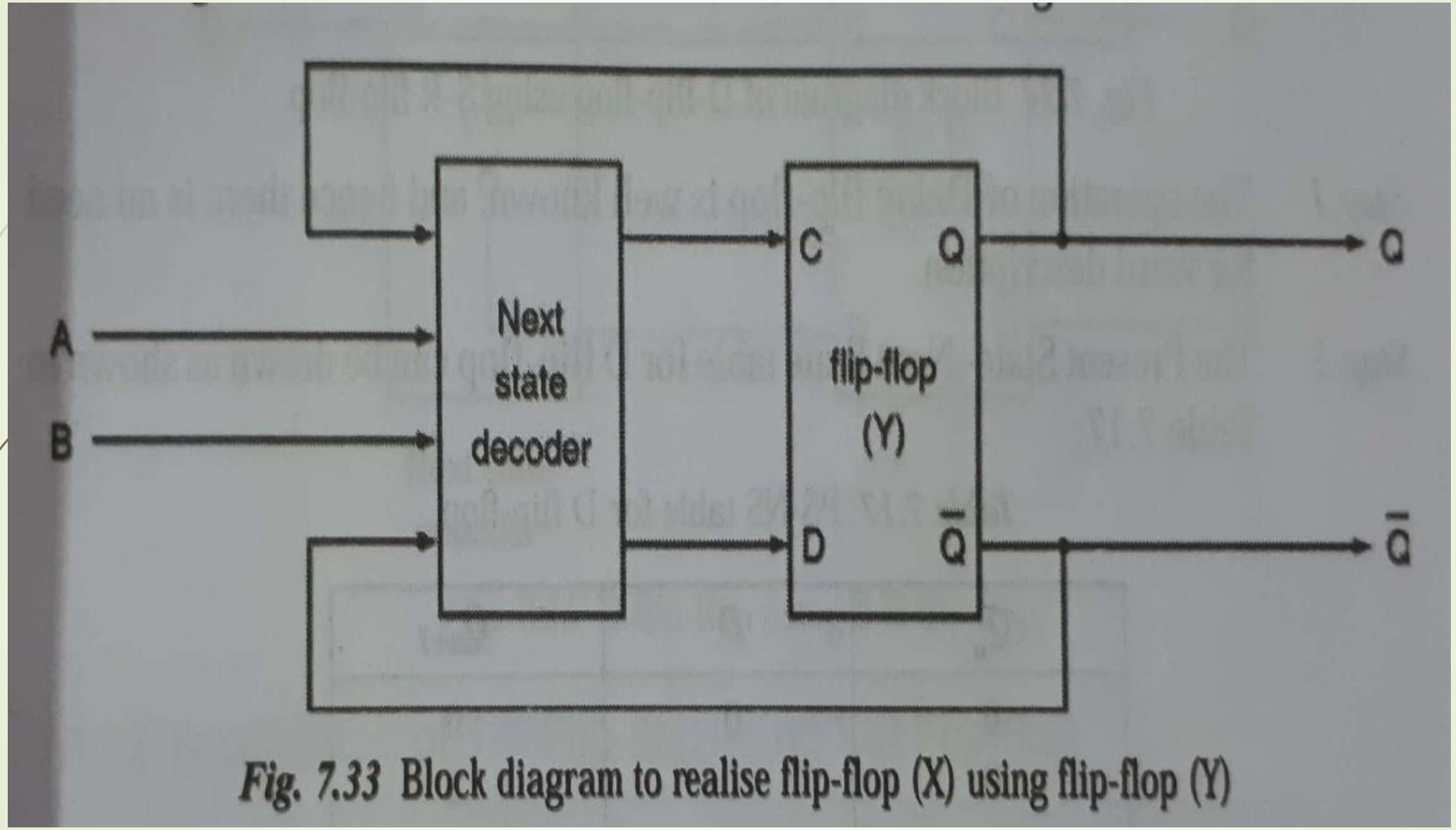


Fig. 7.33 Block diagram to realise flip-flop (X) using flip-flop (Y)

By : Ms. Kanika Dhingra

STEPS TO BE FOLLOWED:

(i) Obtain a clear word description of the desired flip-flop (X).

(ii) Obtain a Present State–Next State (PS-NS) table for the desired flip-flop(X).

(iii) Using the excitation table or application table of the chosen flip-flop(Y), append the next state code or the excitation input values to the above present state–next state table.

(iv) Using K-maps, simplify the logic expressions for excitation inputs of flip-flop(Y) and design the next state decoder logic.

(v) Draw a circuit for the desired flip-flop(X) using next state decoder logic and the chosen flip-flop(Y) as shown in the block diagram.

By : Ms. Kanika Dhingra

7.11.1 Realisation of Delay Flip-flop using S-R Flip-flop

Consider the realisation of Delay flip-flop using S-R flip-flop. Here, the desired flip-flop (X) is Delay flip-flop and the chosen flip-flop(Y) is S-R flip-flop. The basic block diagram is shown in Fig. 7.34.

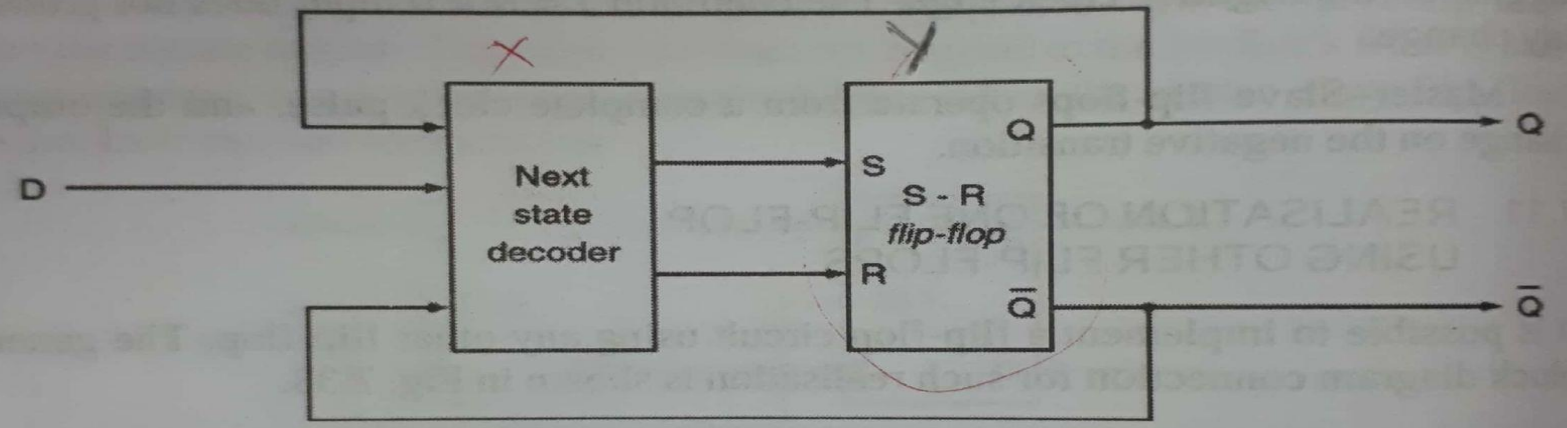


Fig. 7.34 Block diagram of D-flip-flop using S-R flip-flop

Step 1 The operation of Delay flip-flop is well known, and hence there is no need for word description.

Step 2 The Present State–Next State table for D flip-flop can be drawn as shown in Table 7.17.

Table 7.17 PS-NS table for D flip-flop

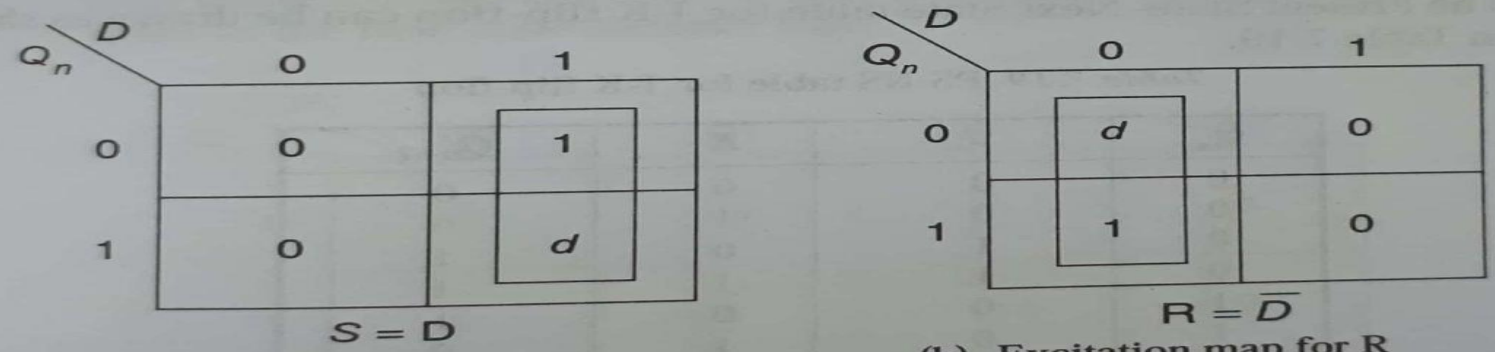
Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

Step 3 Using the application table of S-R flip-flop given in Table 7.4, the next state codes, i.e. S & R values, can be augmented in the above PS-NS table as shown in Table 7.18.

Table 7.18 Excitation table for D flip-flop realisation using S-R flip-flop

Q_n	D	Q_{n+1}	Excitation inputs	
			S	R
0	0	0	0	d
0	1	1	1	0
1	0	0	0	1
1	1	1	d	0

Step 4 In this step, one can design the next state decoder, i.e. the simplified expressions for S and R, from the excitation maps as shown in Fig. 7.35(a) and (b).



(a) Excitation map for S

(b) Excitation map for R

Fig. 7.35

Step 5 From the above step, $S = D$ and $R = \overline{D}$. Now, the circuit for Delay flip-flop using S-R flip-flop can be drawn as shown in Fig. 7.36, with a single NOT gate in the next state decoder logic. Note that the designed circuit confirms with the earlier construction of the delay flip-flop using S-R flip-flop.

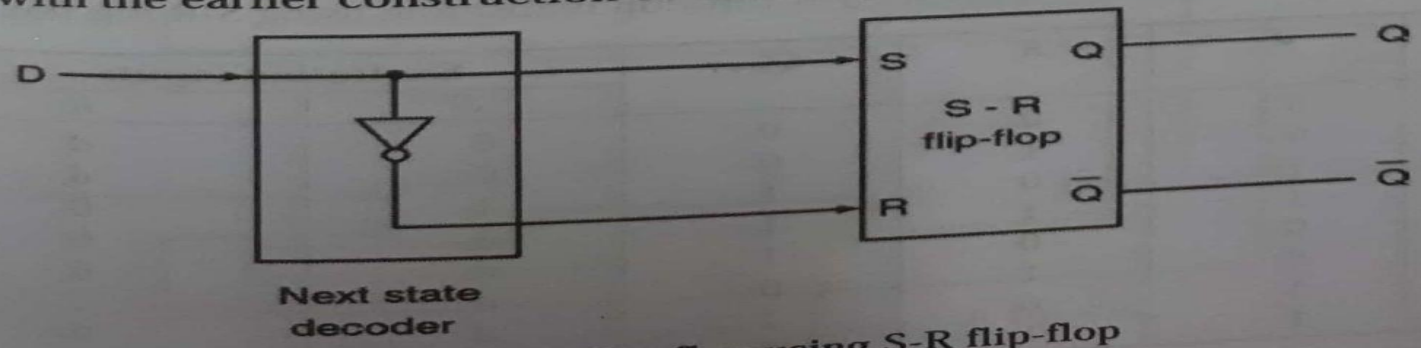


Fig. 7.36 D flip-flop using S-R flip-flop

Realization of JK flip flop using SR Flip flop

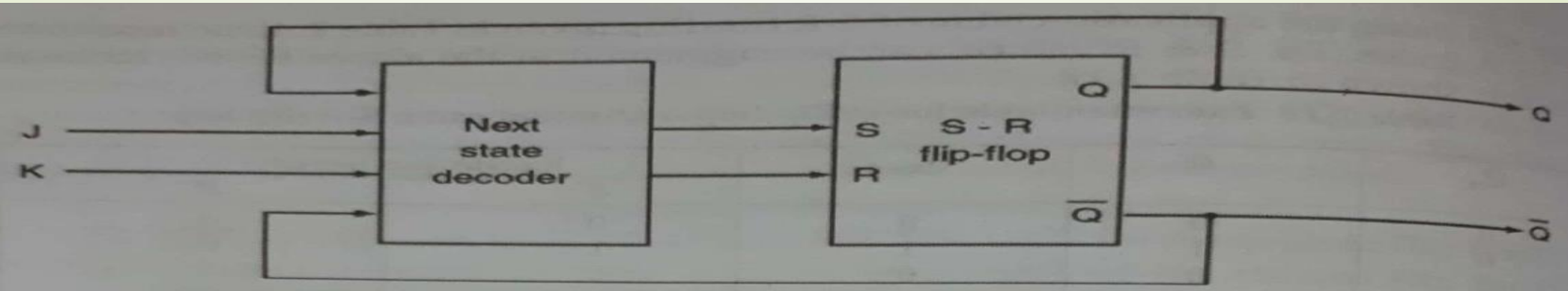


Fig. 7.37 Block diagram of J-K flip-flop using S-R flip-flop

Step 1 The operation of a J-K flip-flop is well known; hence, description is not given.

Step 2 The Present State–Next State table for J-K flip-flop can be drawn as shown in Table 7.19.

Table 7.19 PS-NS table for J-K flip-flop

Q_n	J	K	Q_{n+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

Step 3 Using the application table of S-R flip-flop given in Table 7.4, the next state codes, i.e. S & R values, can be augmented in the above PS-NS table as shown in Table 7.20.

Table 7.20 Excitation table for J-K flip-flop realisation using S-R flip-flop

Q_n	J	K	Q_{n+1}	Excitation inputs	
				S	R
0	0	0	0	0	d
0	0	1	0	0	d
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	1	d	0
1	0	1	0	0	1
1	1	0	1	d	0
1	1	1	0	0	1

Step 4 In this step, one can design the next state decoder, i.e. the simplified expressions for S and R, from the excitation maps as shown in Fig. 7.38(a) and (b).

Q_n \ JK	00	01	11	10
0	0	0	1	1
1	d	0	0	d

$$S = J\overline{Q}_n$$

Fig. 7.38(a) Excitation map for S

Step 5 From the above step, $S = J\overline{Q}_n$ and $R = KQ_n$. Now, the circuit for J-K flip-flop using S-R flip-flop can be drawn as shown in Fig. 7.39 with two AND gates in the next state decoder logic.

Q_n \ JK	00	01	11	10
0	d	d	0	0
1	0	1	1	0

$$R = KQ_n$$

Fig. 7.38(b) Excitation map for R

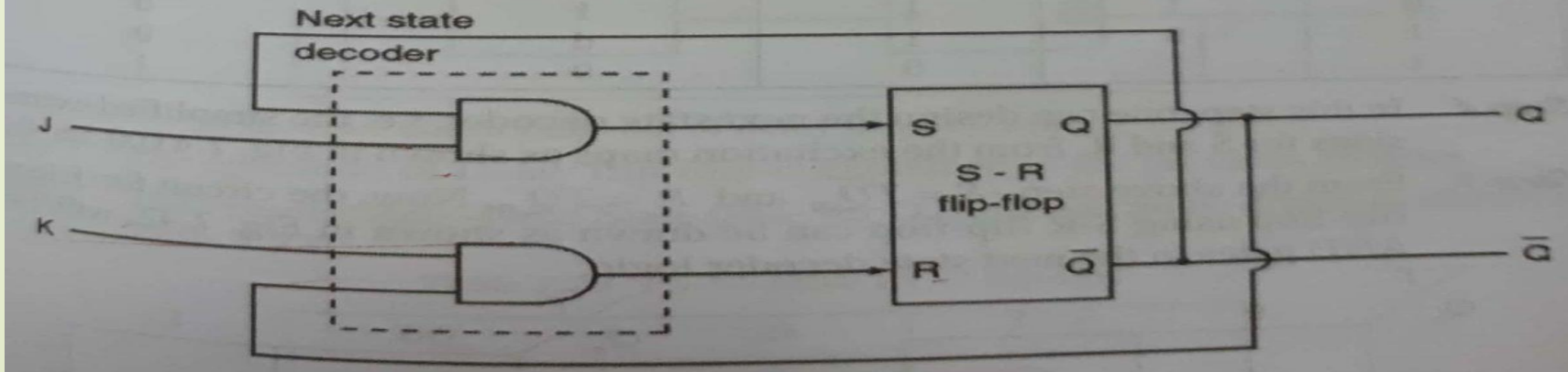


Fig. 7.39 J-K flip-flop using S-R flip-flop

7.11.3

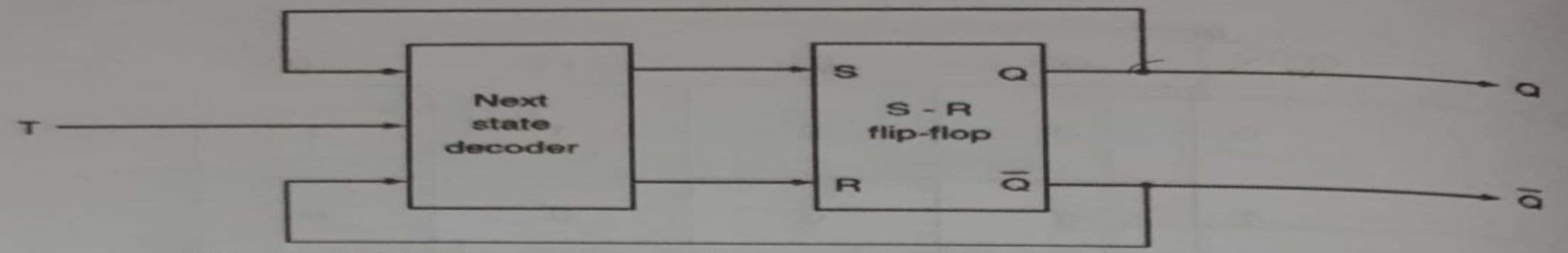


Fig. 7.40 Block diagram of Trigger flip-flop using S-R flip-flop

Step 1 The operation of Trigger flip-flop was explained previously.

Step 2 The Present State–Next State table for Trigger flip-flop can be drawn as shown in Table 7.21.

Table 7.21 PS-NS table for T flip-flop

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

Step 3 Using the application table of S-R flip-flop given in Table 7.4, the next state codes, i.e. S and R values, can be augmented in the above PS-NS table as shown in Table 7.22.

Table 7.22 Excitation table for realisation of T flip-flop using S-R flip-flop

Q_n	T	Q_{n+1}	Excitation inputs	
			S	R
0	0	0	0	d
0	1	1	1	0
1	0	1	d	0
1	1	0	0	1

Step 4 In this step, one can design the next state decoder, i.e. the simplified expressions for S and R, from the excitation maps as shown in Fig. 7.41 (a) and (b).

Step 5 From the above step, $S = T\overline{Q_n}$ and $R = TQ_n$. Now, the circuit for Trigger flip-flop using S-R flip-flop can be drawn as shown in Fig. 7.42, with two AND gates in the next state decoder logic.

Q_n \ T	0	1
0	0	(1)
1	d	0

$S = T\overline{Q_n}$

(a) Excitation map for S

Q_n \ T	0	1
0	d	0
1	0	(1)

$S = TQ_n$

(b) Excitation map for R

Fig. 7.41

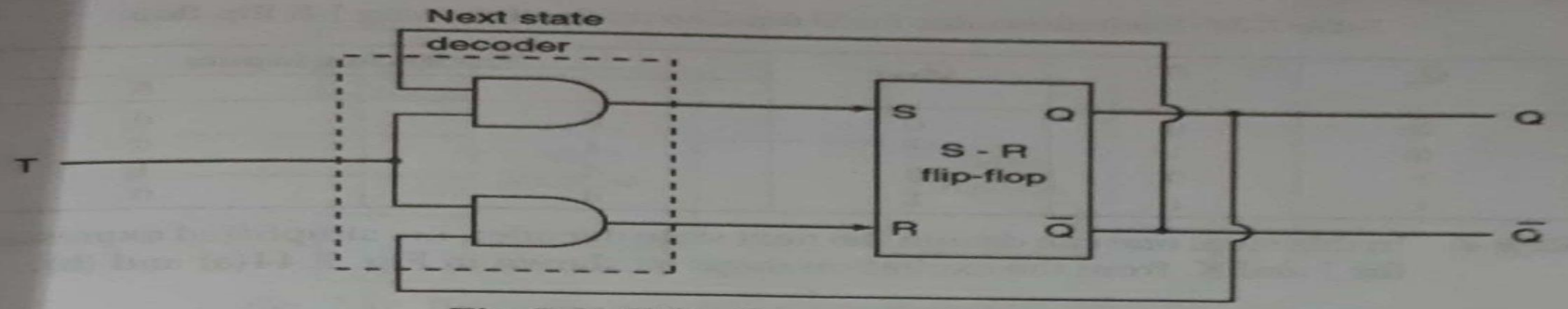


Fig. 7.42 T flip-flop using S-R flip-flop

From the above realisation of Delay, J-K and Trigger flip-flop using S-R flip-flop, one can understand that the resultant circuit is the same as the one discussed in the earlier sections.

Similarly, one can realise any flip-flop using another flip-flop. To get a better insight, the following two realisations are explained.

7.11.4 Realisation of Delay Flip-flop using J-K Flip-flop

Here, the desired flip-flop (X) is a Delay flip-flop and the chosen one (Y) is a J-K flip-flop. The simple block diagram of such a realisation is shown in Fig. 7.43.

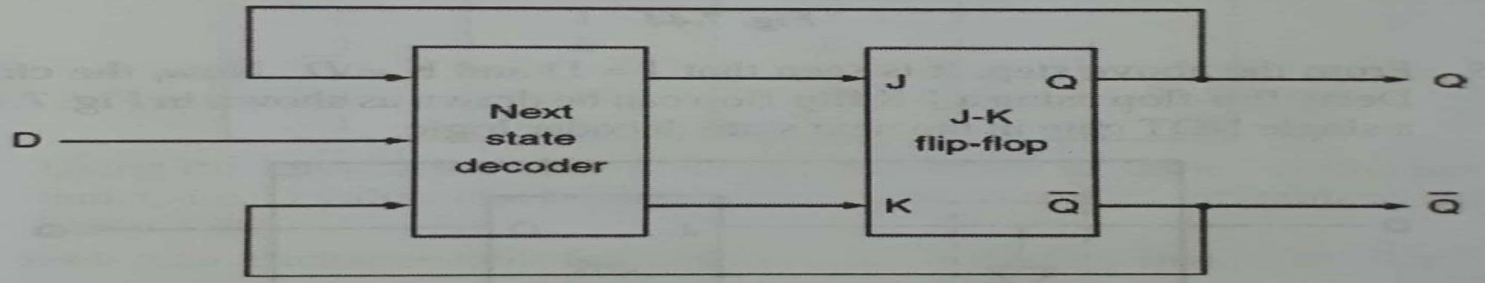


Fig. 7.43 Block diagram of delay flip-flop using J-K flip-flop

Step 1 The operation of Delay flip-flop is explained in the previous sections

Step 2 The Present State–Next State table for D flip-flop can be drawn as shown in Table 7.23.

Table 7.23 PS-NS table for D flip-flop

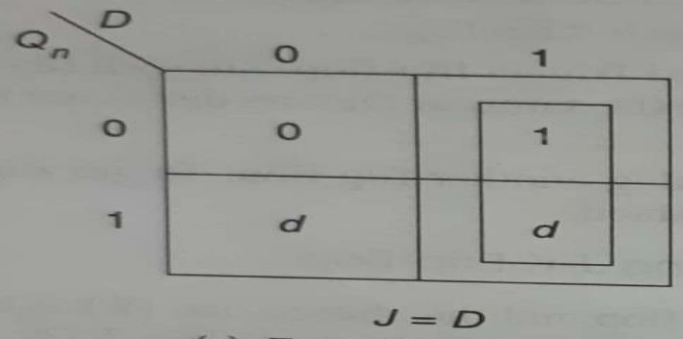
Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

Step 3 Using the application table of J-K flip-flop given in Table 7.11, the next state codes, i.e. J & K values, can be augmented in the above PS-NS table as shown in Table 7.24.

Table 7.24 Excitation table for D flip-flop realisation using J-K flip-flop

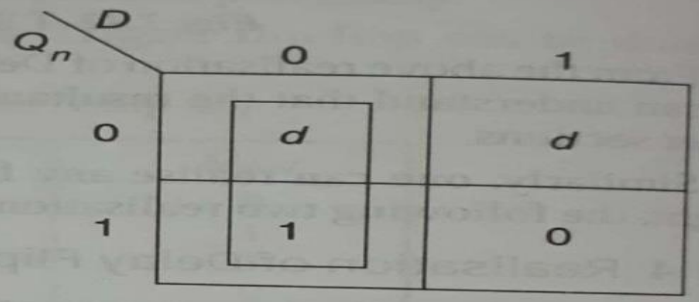
Q_n	D	Q_{n+1}	Excitation inputs	
			J	K
0	0	0	0	d
0	1	1	1	d
1	0	0	d	1
1	1	1	d	0

Step 4 In this step, one can design the next state decoder, i.e. simplified expressions for J and K, from the excitation maps as shown in Fig. 7.44(a) and (b).



$J = D$

(a) Excitation map for J



$K = \overline{D}$

(b) Excitation map for K

Fig. 7.44

Step 5 From the above step, it is seen that J = D and K = $\overline{D}$. Now, the circuit for Delay flip-flop using a J-K flip-flop can be drawn as shown in Fig. 7.45, with a single NOT gate in the next state decoder logic.

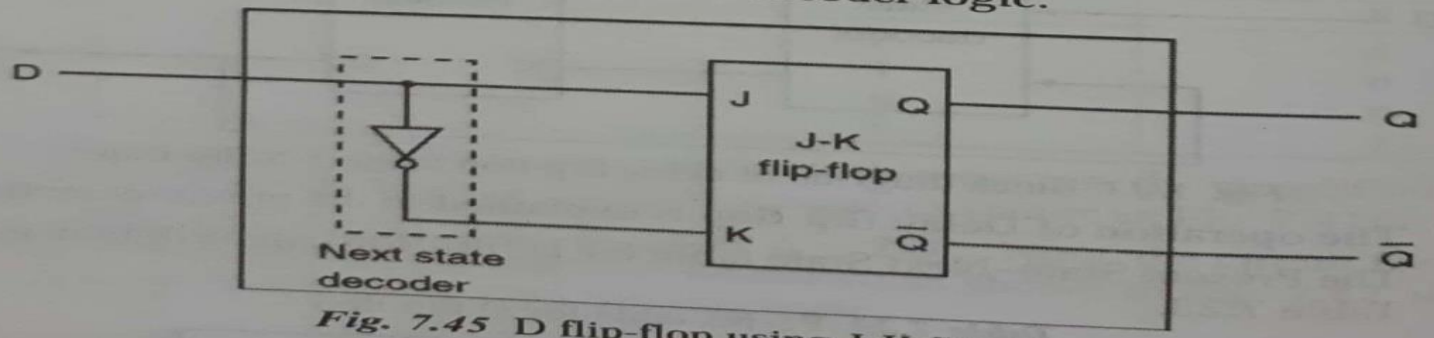


Fig. 7.45 D flip-flop using J-K flip-flop

7.11.5 Realisation of J-K Flip-flop using D Flip-flop

Under this section, the realisation of J-K flip-flop using D flip-flop is taken into consideration. Here, the desired flip-flop (X) is a J-K flip-flop and the chosen one (Y) is a D flip-flop. The block diagram of this realisation is shown in Fig. 7.46.

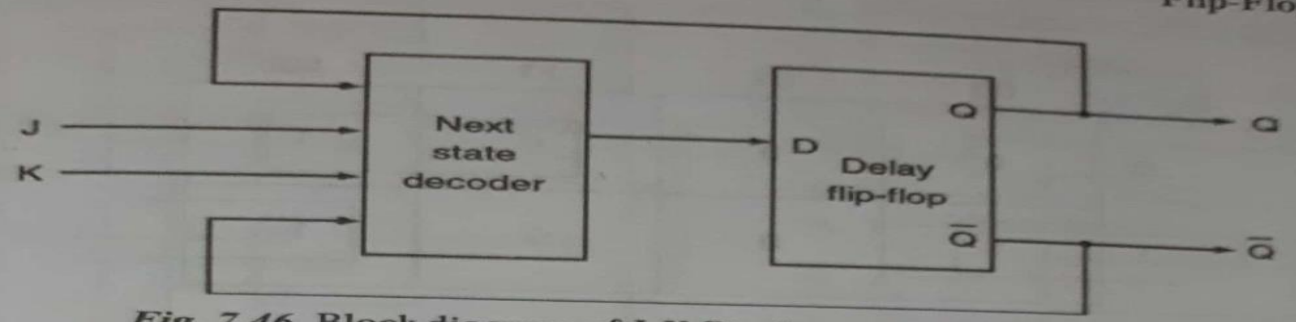


Fig. 7.46 Block diagram of J-K flip-flop using delay flip-flop

Step 1 The operation of J-K flip-flop is well known. Hence, there is no need for word description.

Step 2 The Present State–Next State table for J-K flip-flop can be drawn as shown in Table 7.25.

Table 7.25 PS-NS table for J-K flip-flop

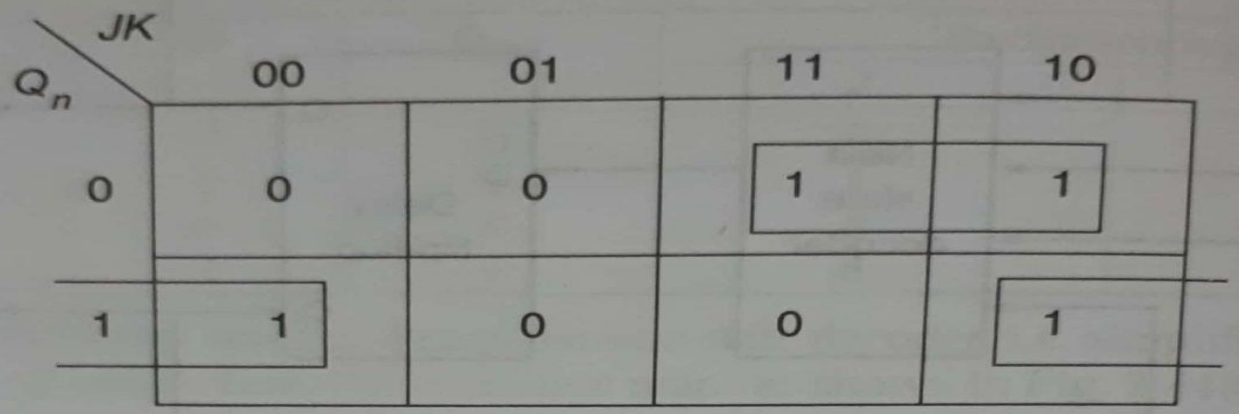
Q_n	J	K	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Step 3 Using the application table of D flip-flop given in Table 7.8, the next state codes, i.e. D value, can be augmented in the above PS-NS table as shown in Table 7.26.

Table 7.26 Excitation table for realisation of J-K flip-flop using D flip-flop

Q_n	J	K	Q_{n+1}	Excitation input D
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	1	1
1	0	0	1	1
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0

Step 4 In this step, one can design the next state decoder, i.e. the simplified expression for D, from the excitation map as shown in Fig. 7.47.



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

Fig. 7.47 Excitation map for D

Step 5 From the above step, it can be seen that $D = J\overline{Q}_n + \overline{K}Q_n$. Now, the circuit for J-K flip-flop using D flip-flop is as shown in Fig. 7.48 with two AND gates, one OR gate and one NOT gate in the next state decoder logic.

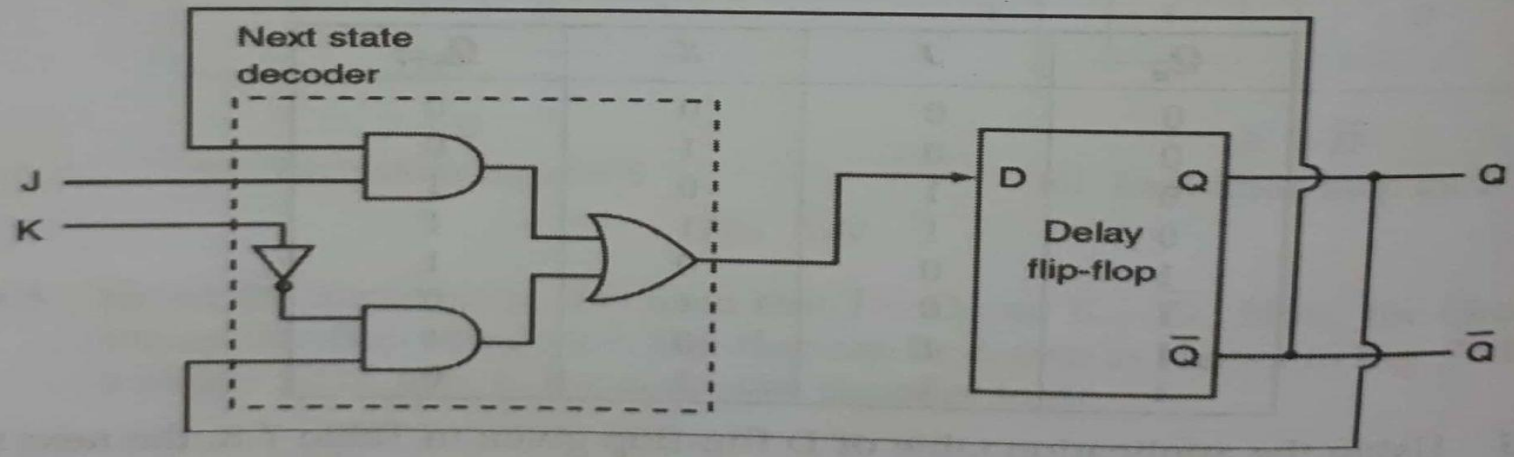


Fig. 7.48 J-K flip-flop using D flip-flop

Registers

A register is a group of flip-flops suitable for storing binary information. Each flip flop is a binary cell capable of storing one bit of information. An n-bit register has a group of n flip flops and is capable of storing any binary information containing n bits. The register is mainly used for storing and shifting binary data entered into it from an external source.

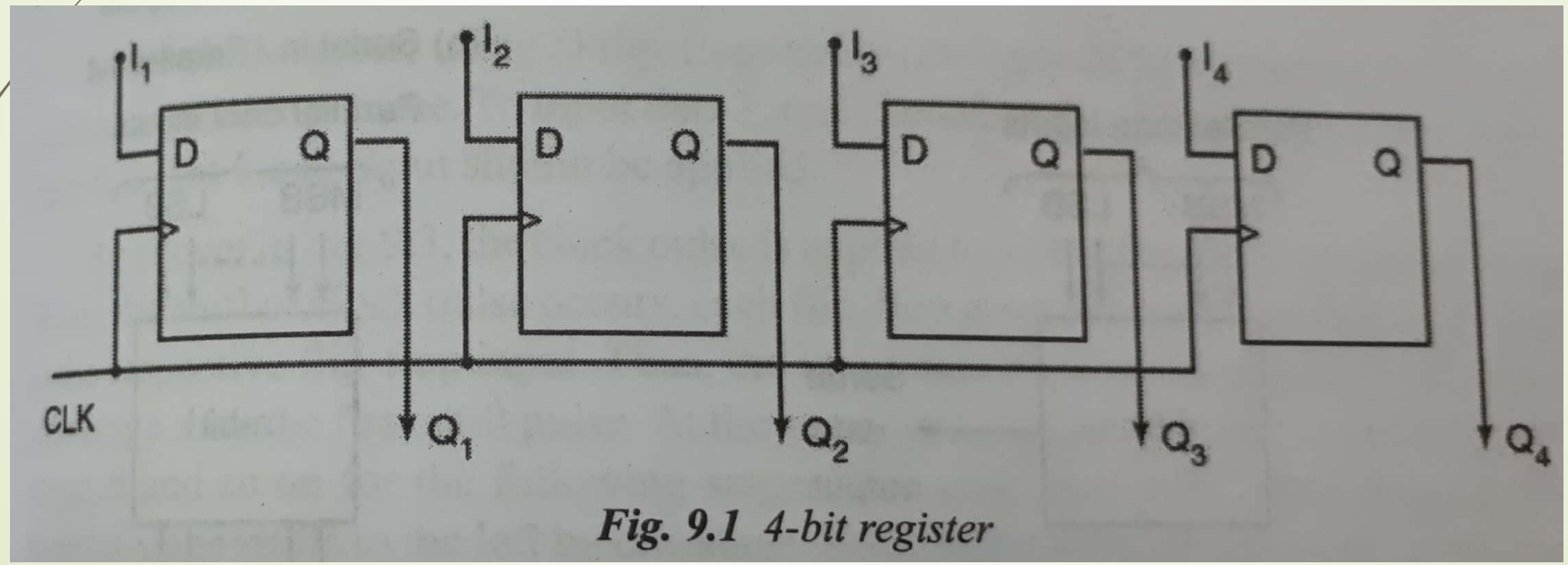


Fig. 9.1 4-bit register

Shift Registers

A register that is used to store binary information is known as a **memory register**. A register capable of shifting binary information either to the right or to the left is called a **shift register**.

There are two methods of shifting data:

i) Serial Shifting
ii) Parallel Shifting

Serial shifting method shifts one bit at a time for each clock pulse in a serial fashion, beginning with either MSB or LSB.

In Parallel shifting operation, all the data get shifted simultaneously during a single clock pulse.

Parallel shifting is faster than the serial shifting method.

By : Ms. Kanika Dhingra

Shift registers are classified into four types:

1. Serial-in Serial-out (SISO)
2. Serial-in Parallel-out (SIPO)
3. Parallel-in Serial-out (PISO)
4. Parallel-in Parallel-out (PIPO)

Serial-in Serial-out (SISO)

Shift-left register

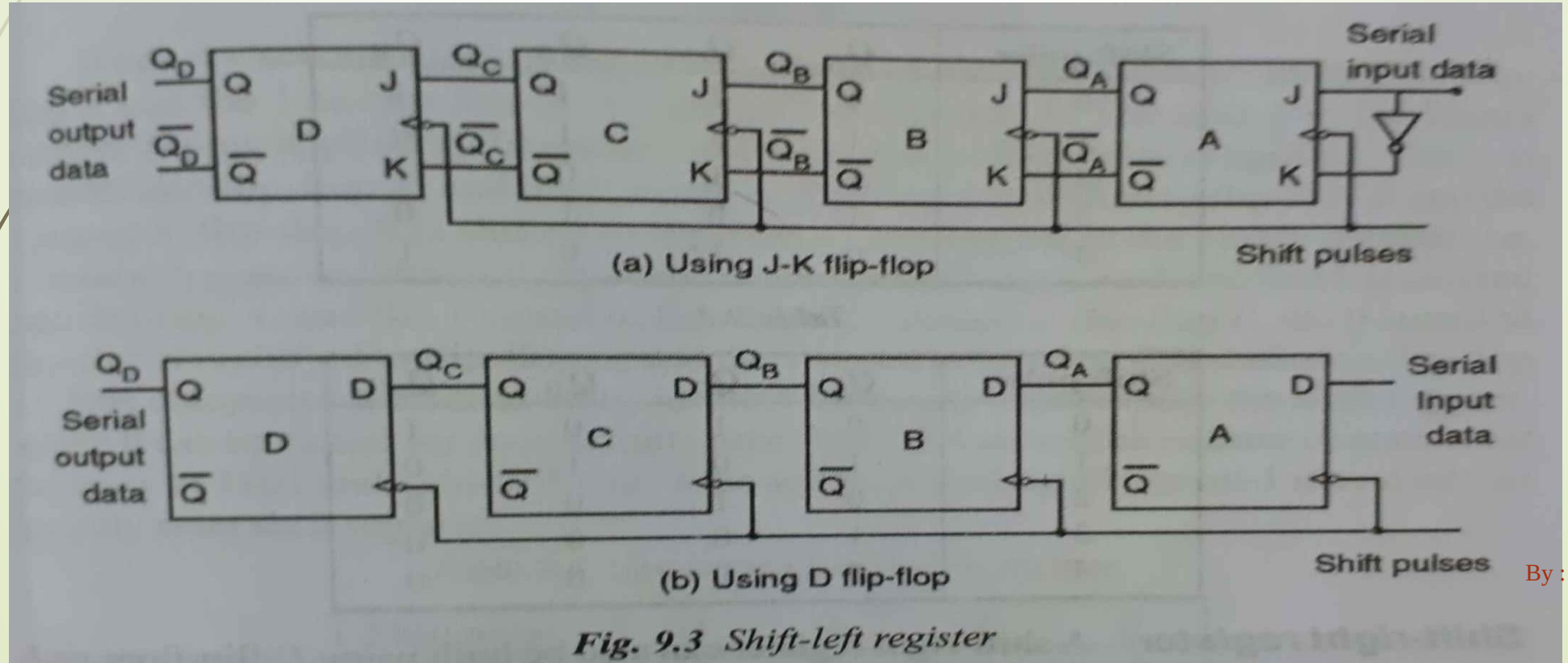


Fig. 9.3 Shift-left register

Table 9.1 Operation of shift-left register

Shift pulse	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	1	1	1
4	1	1	1	1

Table 9.2

Shift pulse	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	0	1	0
4	0	1	0	1

Table 9.3

Shift pulse	Q_D	Q_C	Q_B	Q_A
0	0	1	0	1
1	1	0	1	0
2	0	1	0	0
3	1	0	0	0
4	0	0	0	0

Shift right register

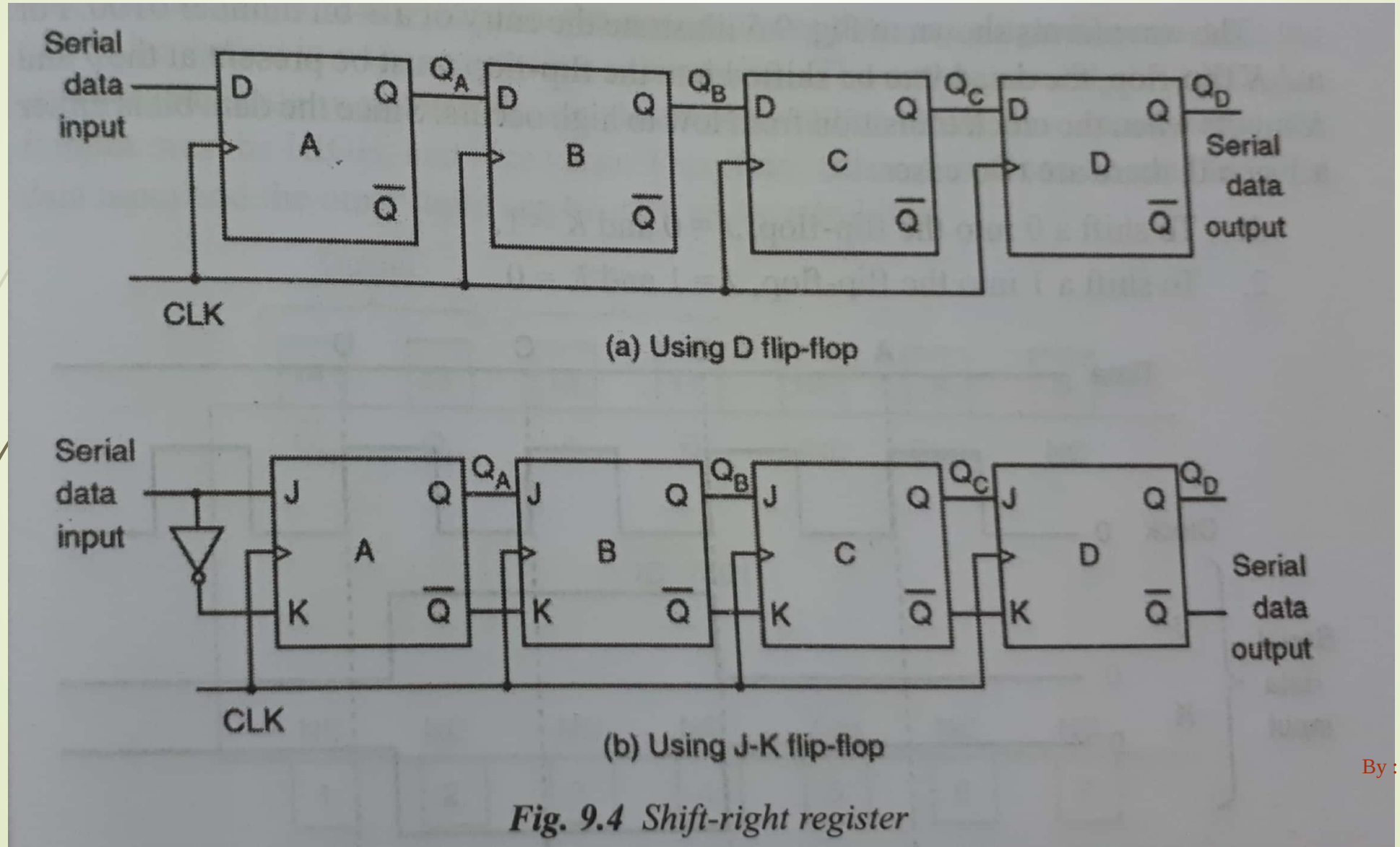


Fig. 9.4 Shift-right register

Table 9.4 Operation of shift-right register

Shift pulse	Q_A	Q_B	Q_C	Q_D
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	0	1	0
4	1	1	0	1

Table 9.5

Shift 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

4-bit serial-in-parallel-out shift register

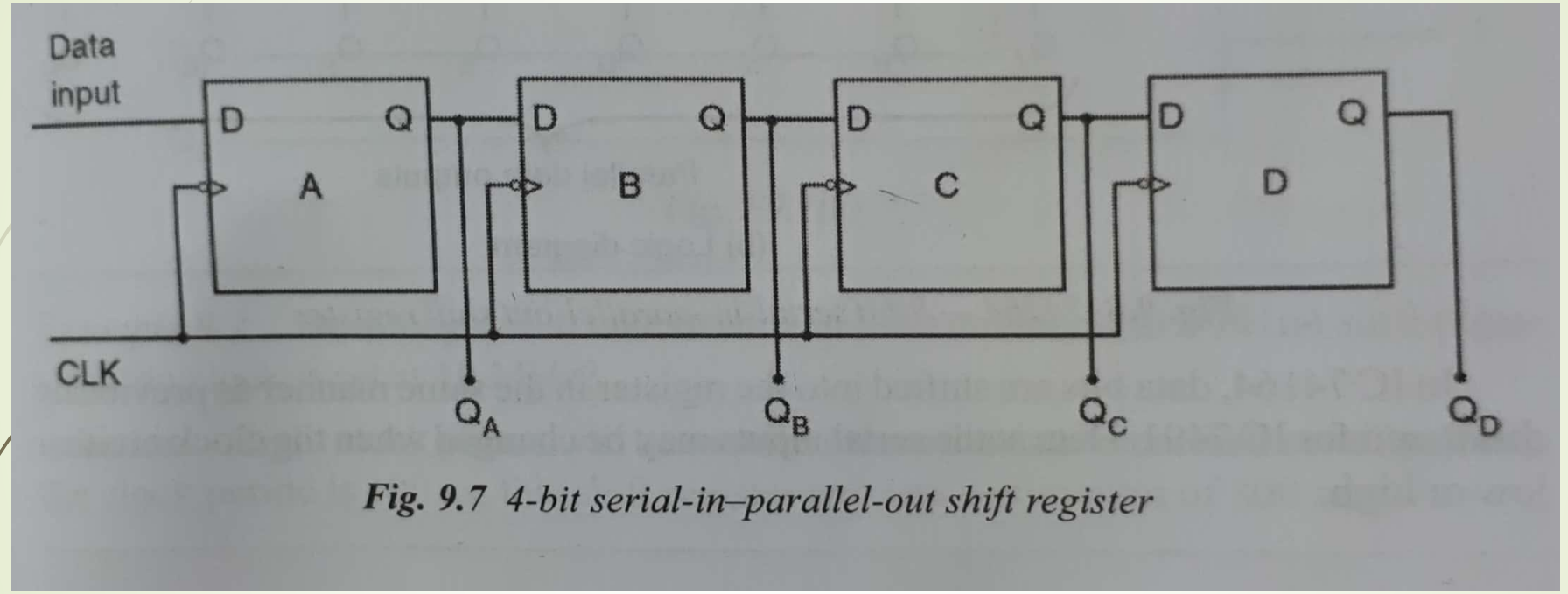


Fig. 9.7 4-bit serial-in–parallel-out shift register

By : Ms. Kanika Dhingra

4-bit parallel-in-serial-out shift register

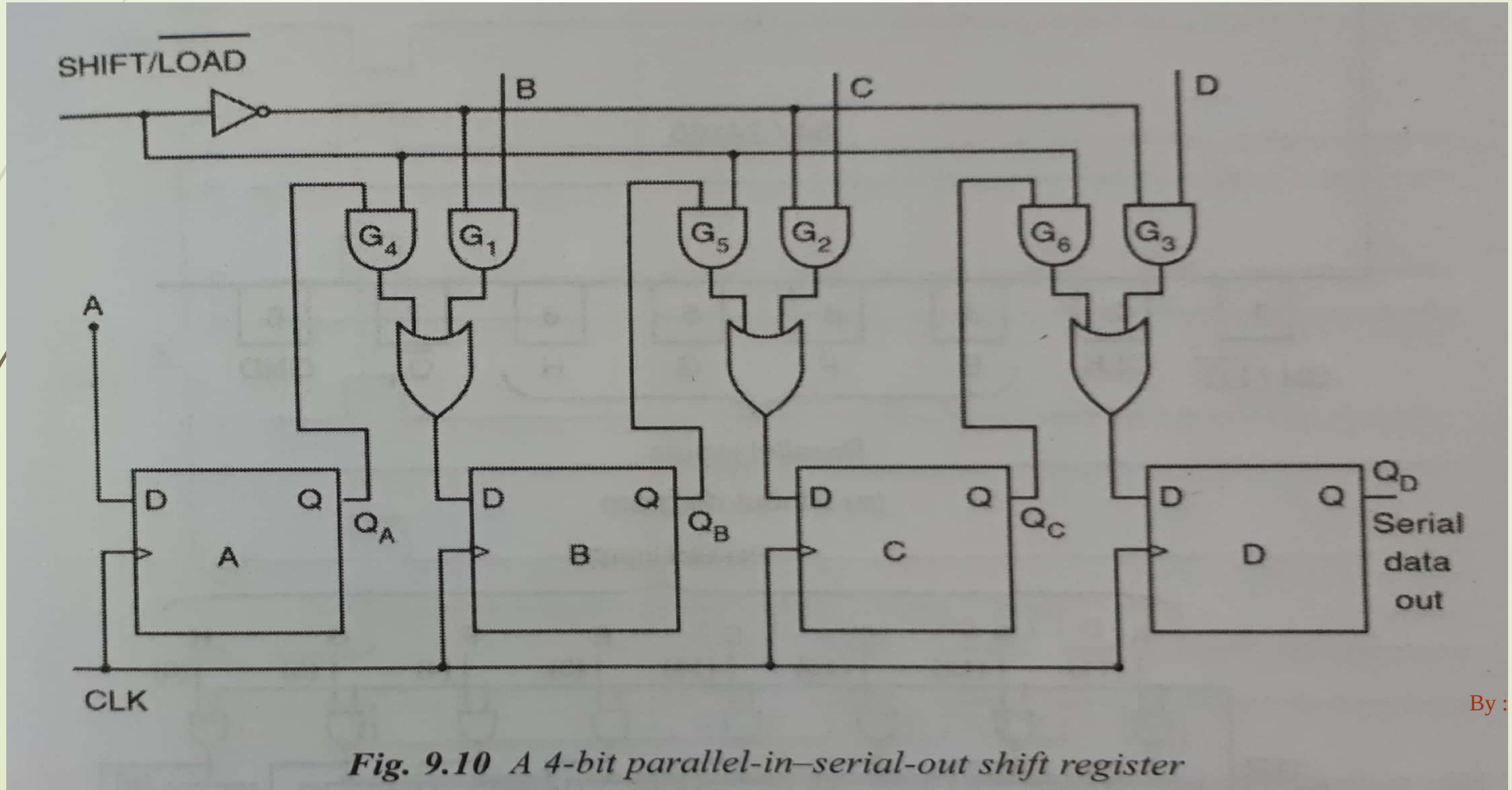


Fig. 9.10 A 4-bit parallel-in–serial-out shift register

4-bit parallel-in-parallel-out shift register

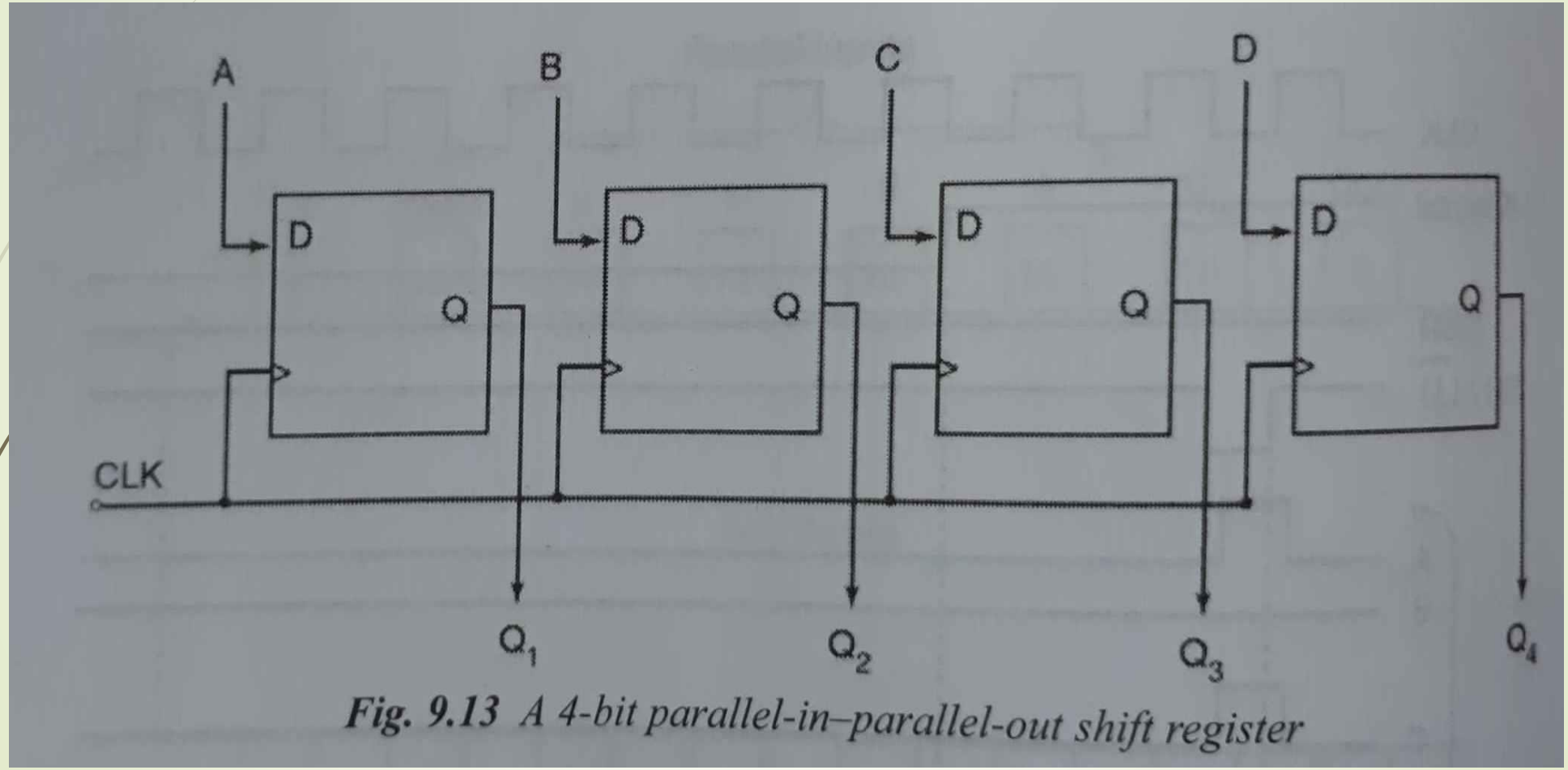


Fig. 9.13 A 4-bit parallel-in–parallel-out shift register

4-bit Universal Shift Register

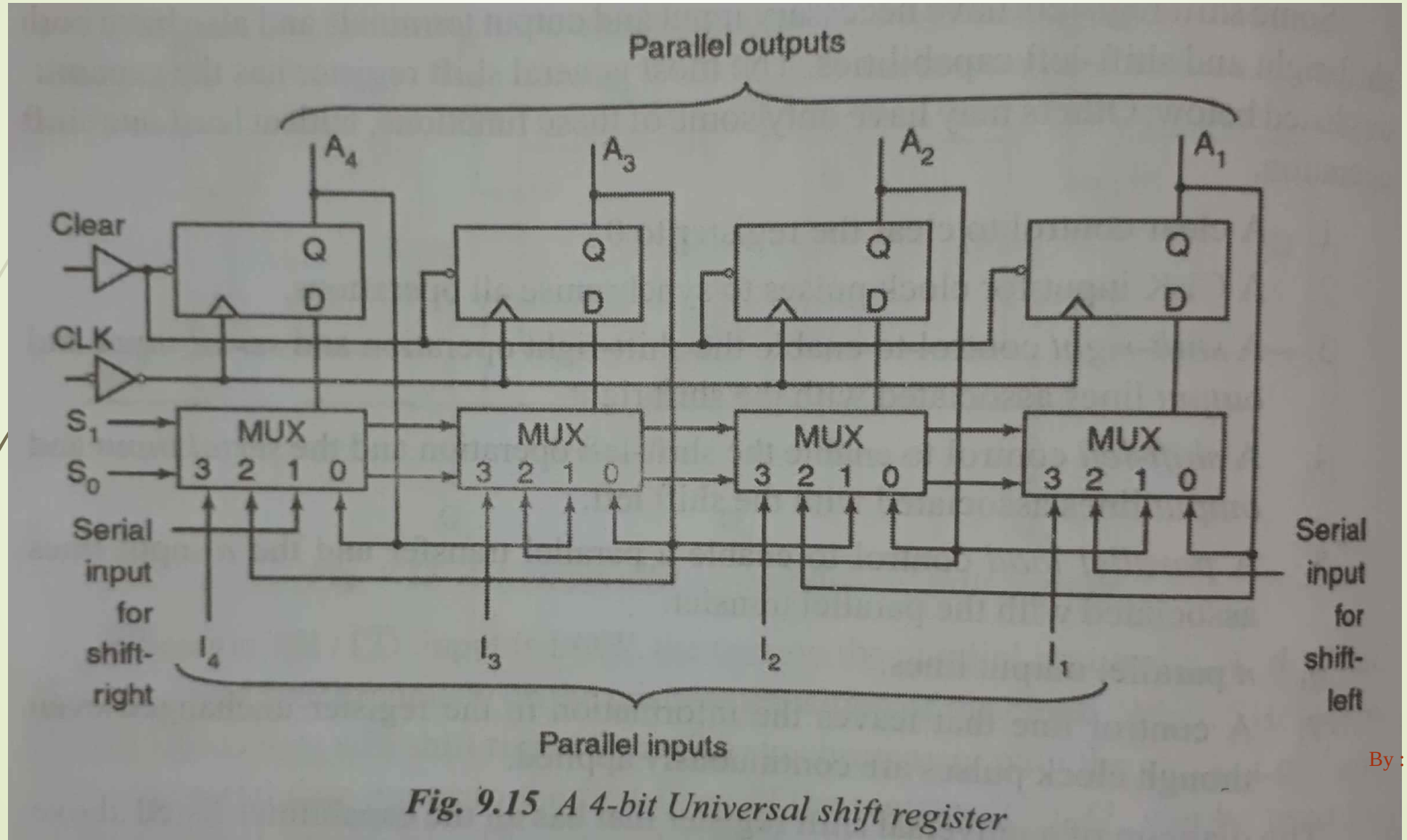


Fig. 9.15 A 4-bit Universal shift register

Mode control		Register operation
S_1	S_0	
0	0	No change
0	1	Shift-right
1	0	Shift-left
1	1	Parallel Load

4-bit bidirectional shift register

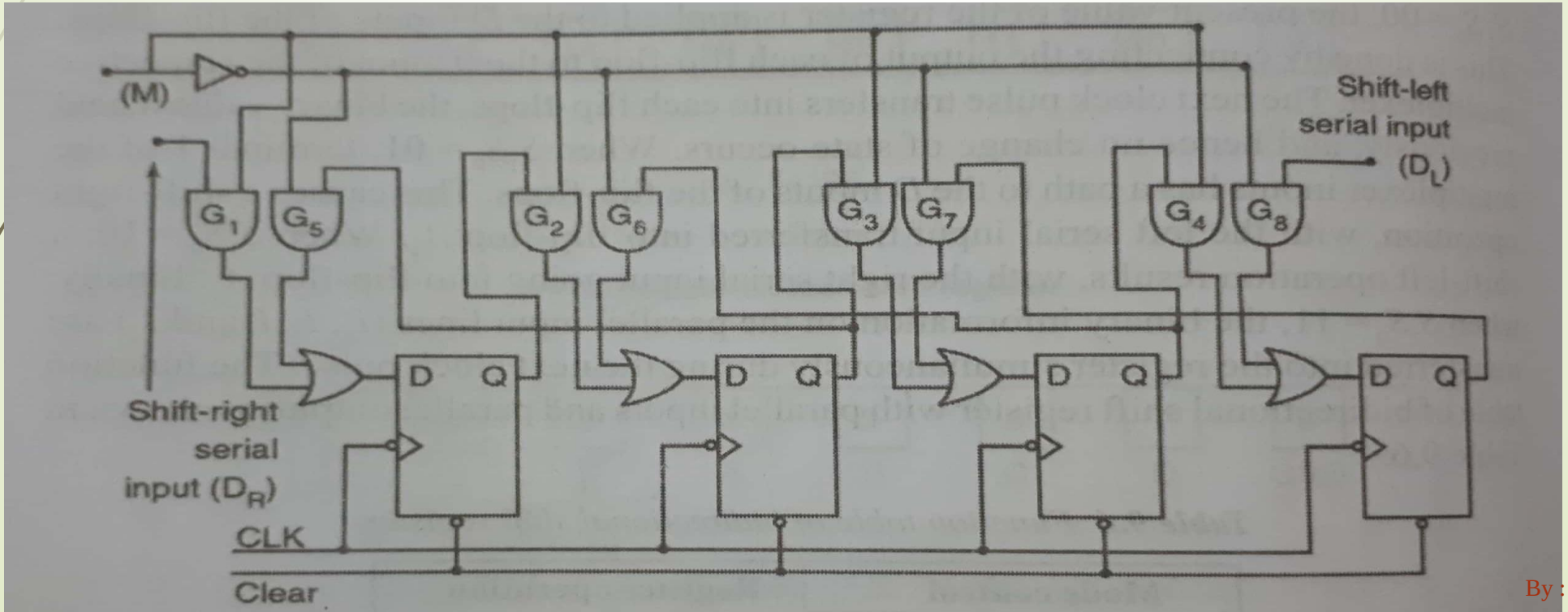


Fig. 9.16 A 4-bit bidirectional shift register

Asynchronous Counter

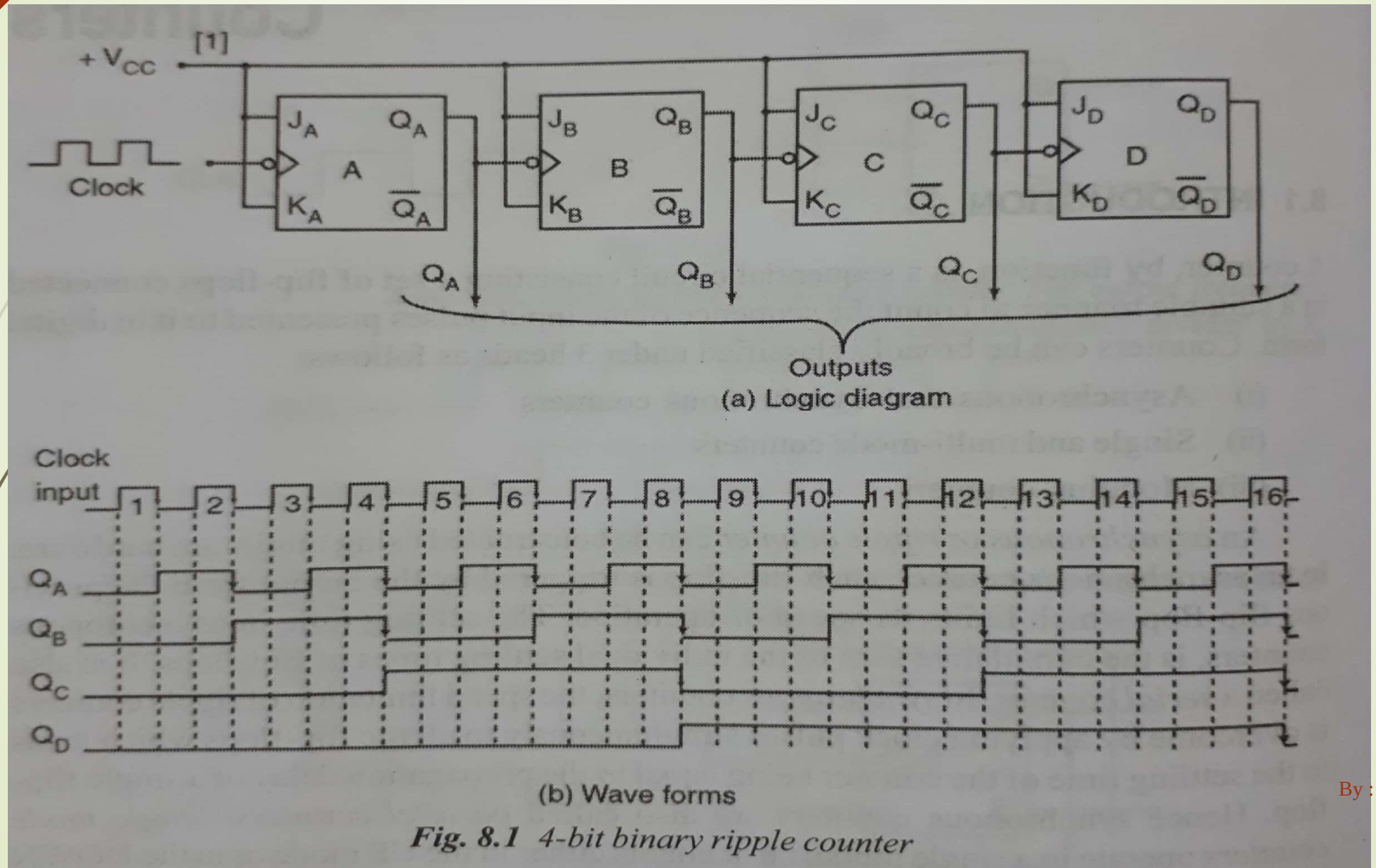


Fig. 8.1 *4-bit binary ripple counter*

Table 8.1 *Truth table of 4-bit binary ripple counter*

State	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
0	0	0	0	0

Mod-number or modulus The counter in Fig. 8.1 has 16 different states. Thus, it is a MOD-16 ripple counter. The MOD-number (or the Modulus) of a counter is the total number of states it sequences through in each complete cycle.

$$\text{MOD-number} = 2^n$$

where n = Number of flip-flops.

The maximum binary number counted by the counter is $2^n - 1$. Thus, a 4-flip-flop counter can count as high as $(1111)_2 = 2^4 - 1 = 15_{10}$.

Asynchronous Down Counter

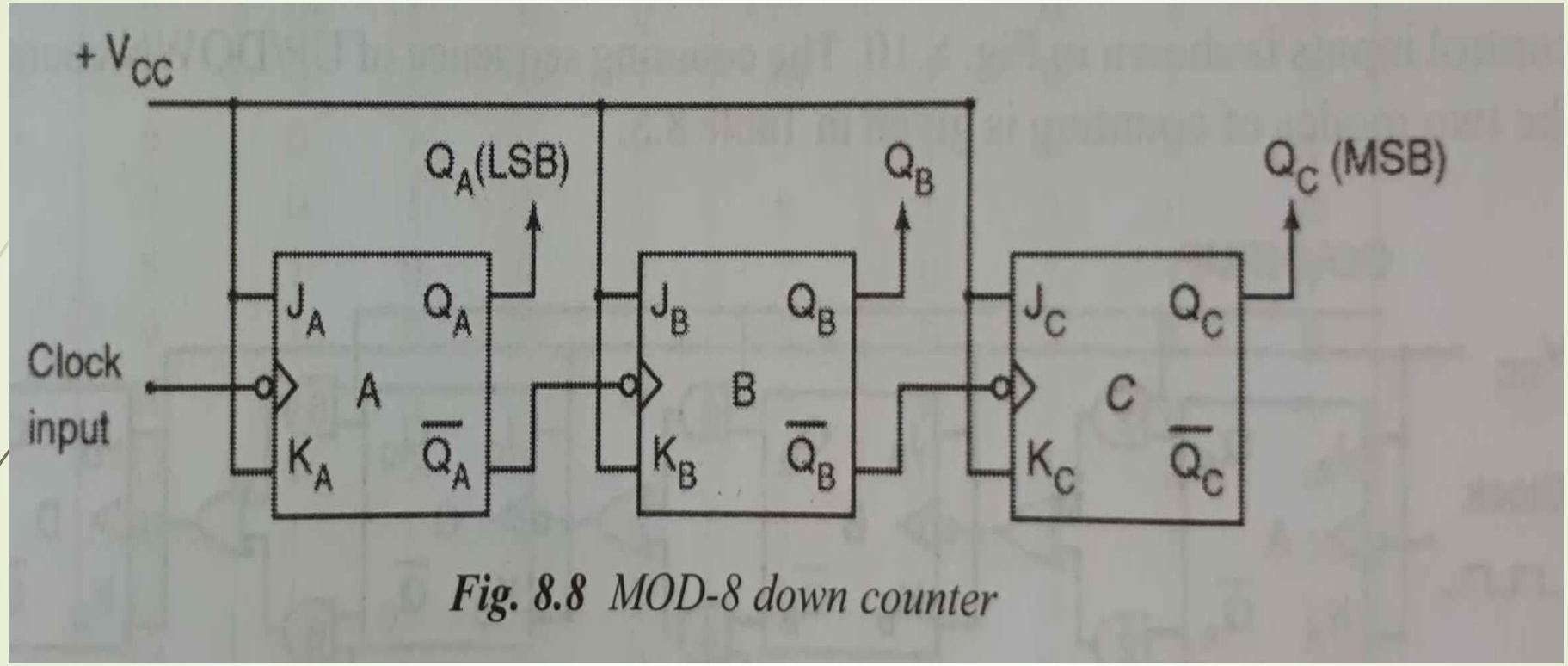


***Fig. 8.8** MOD-8 down counter*

State	Q_C	Q_B	Q_A
7	1	1	1
6	1	1	0
5	1	0	1
4	1	0	0
3	0	1	1
2	0	1	0
1	0	0	1
0	0	0	0
7	1	1	1

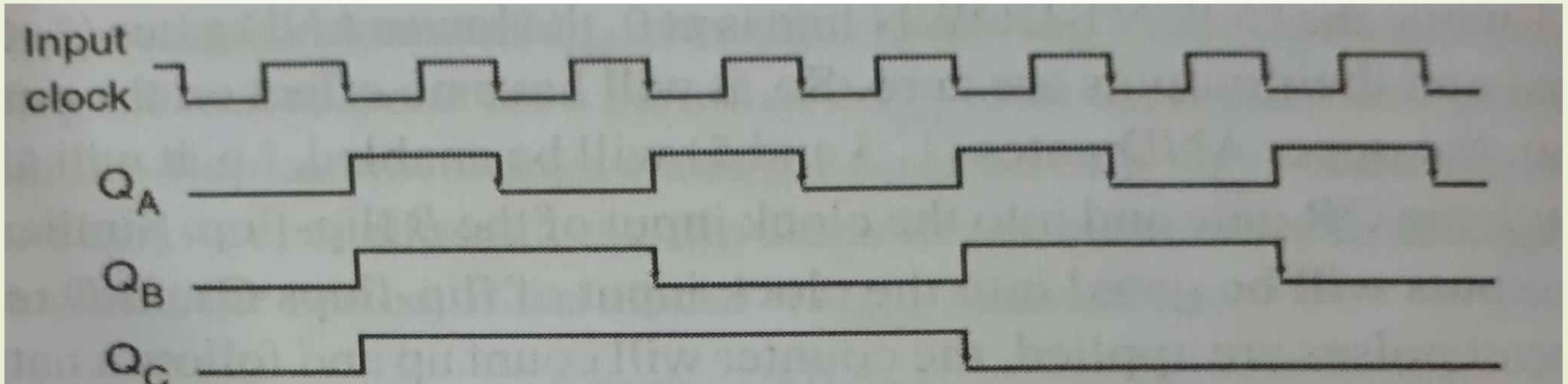


***Fig. 8.9** Waveforms of MOD-8 down counter*

Asynchronous UP- DOWN Counter

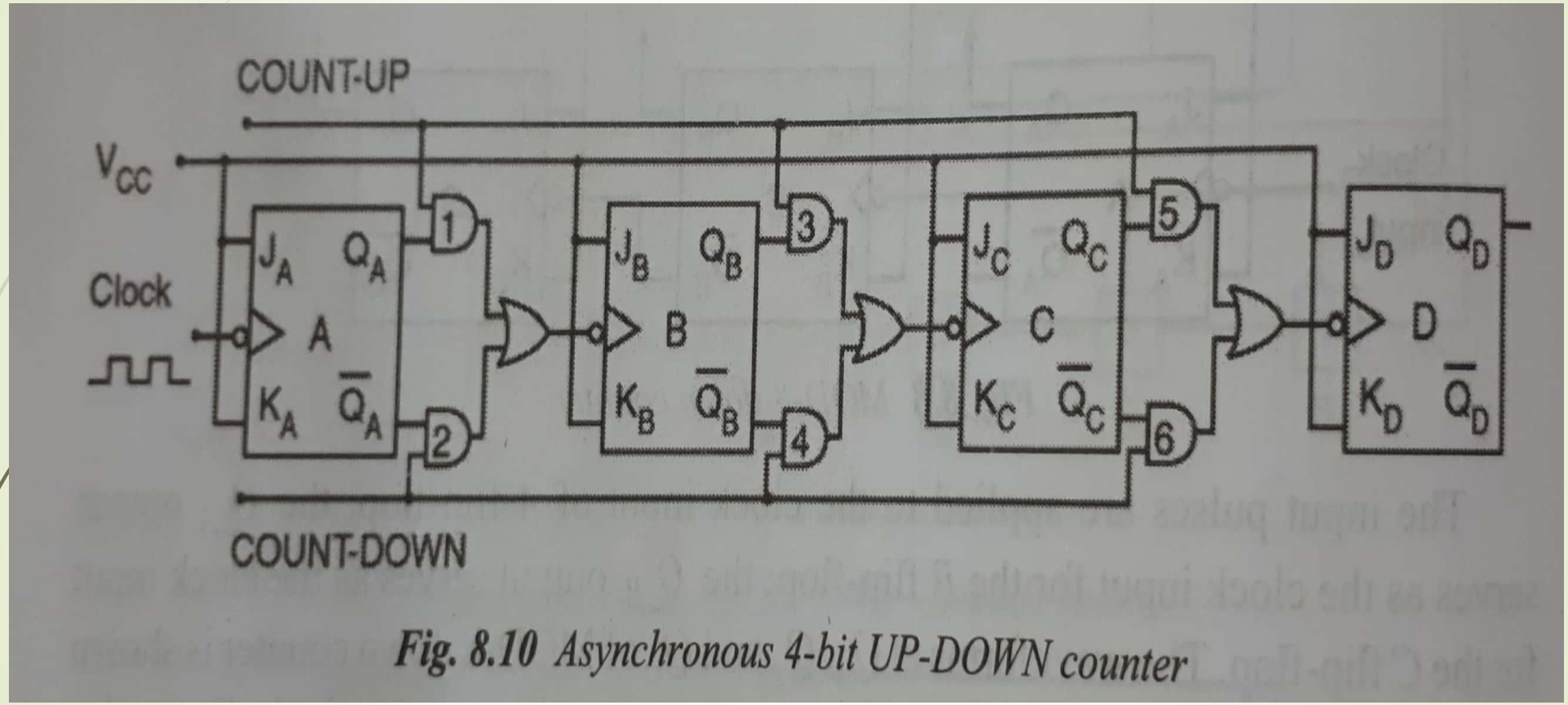


Fig. 8.10 Asynchronous 4-bit UP-DOWN counter

By : Ms. Kanika Dhingra

MOD-6 ripple counter

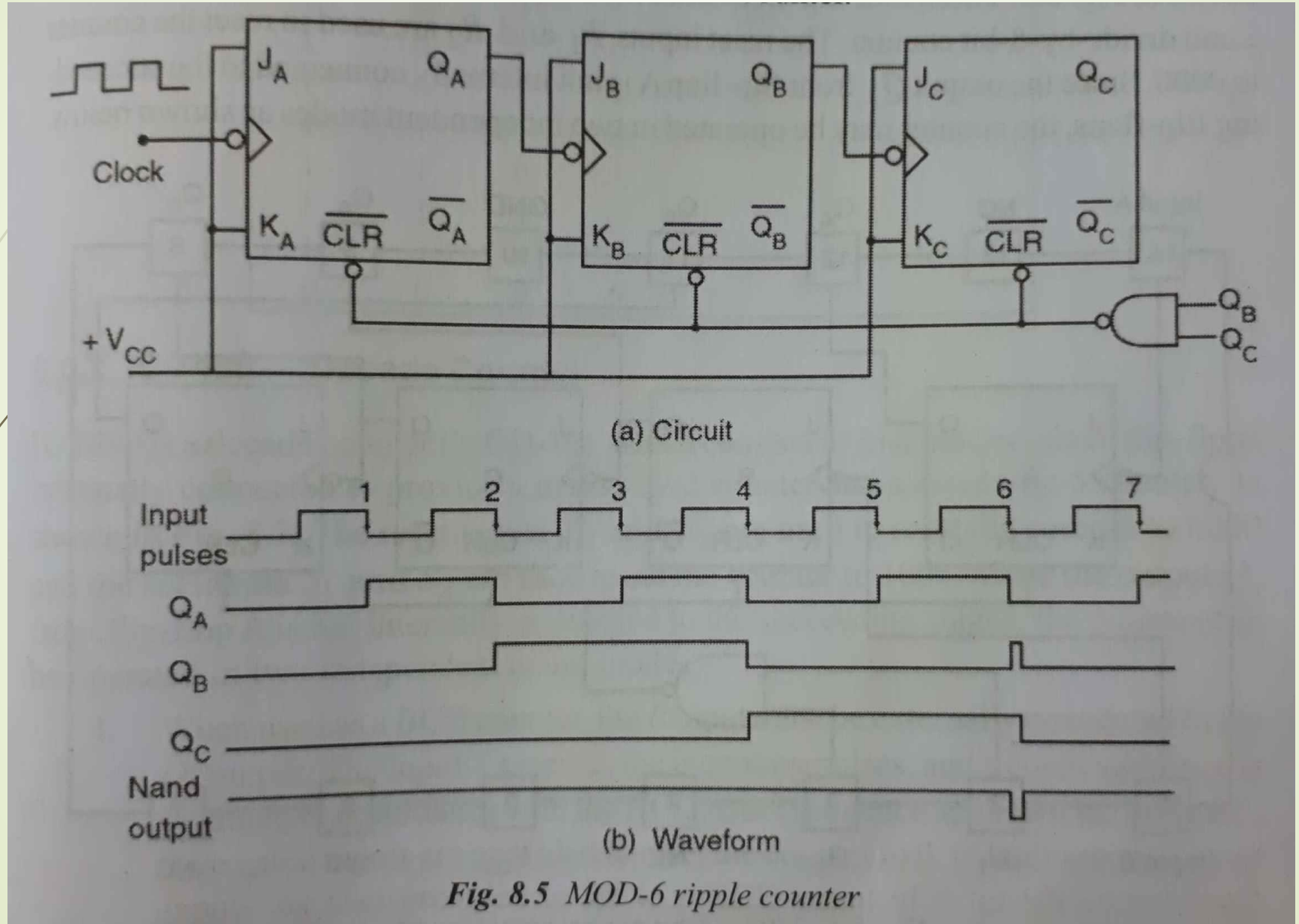


Fig. 8.5 MOD-6 ripple counter

By : Ms. Kanika Dhingra

4-bit Synchronous counter

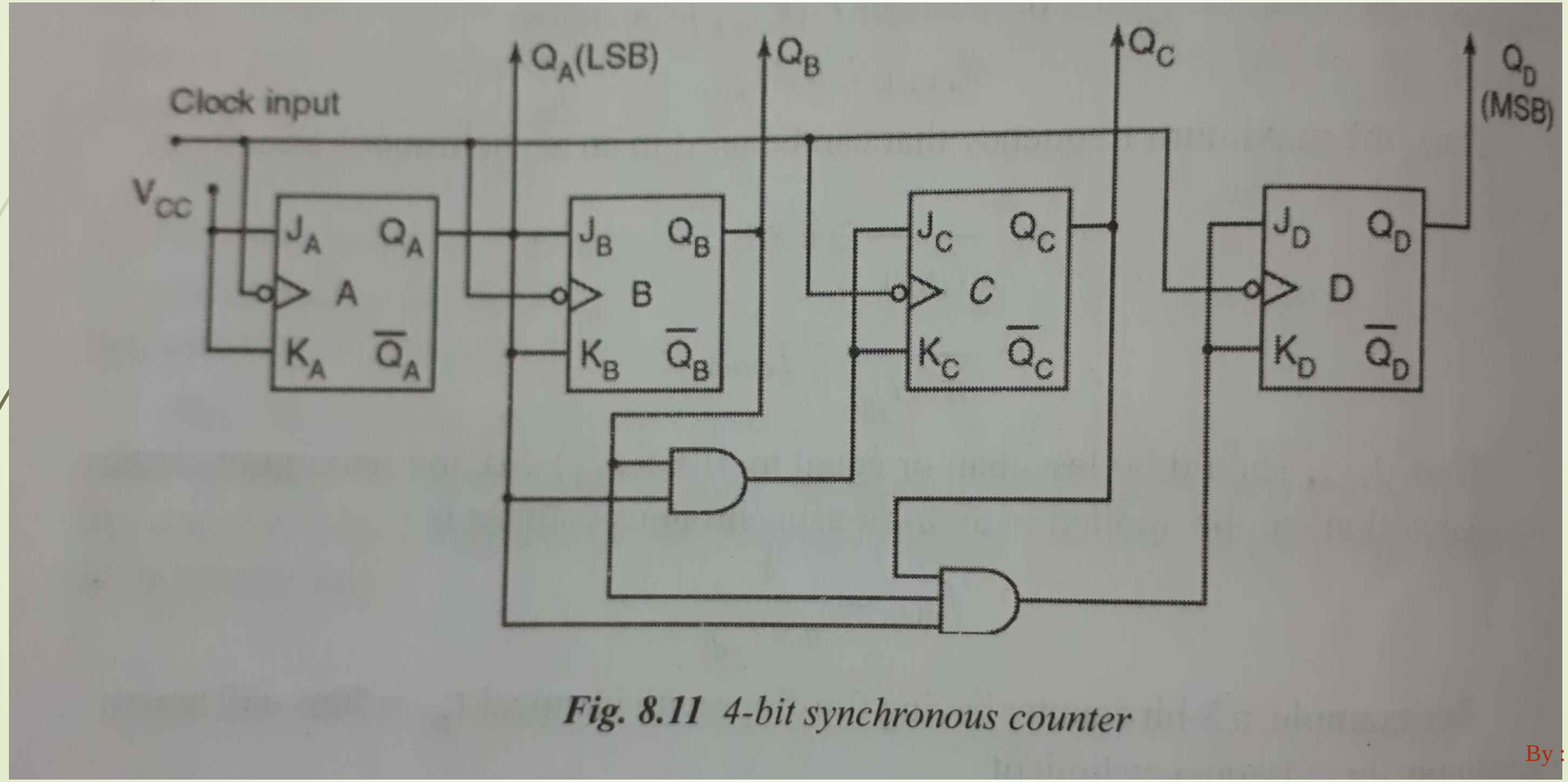


Fig. 8.11 4-bit synchronous counter

Table 8.6 Truth table of 4-bit binary synchronous counter

State	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
0	0	0	0	0

4-bit Synchronous Down Counter

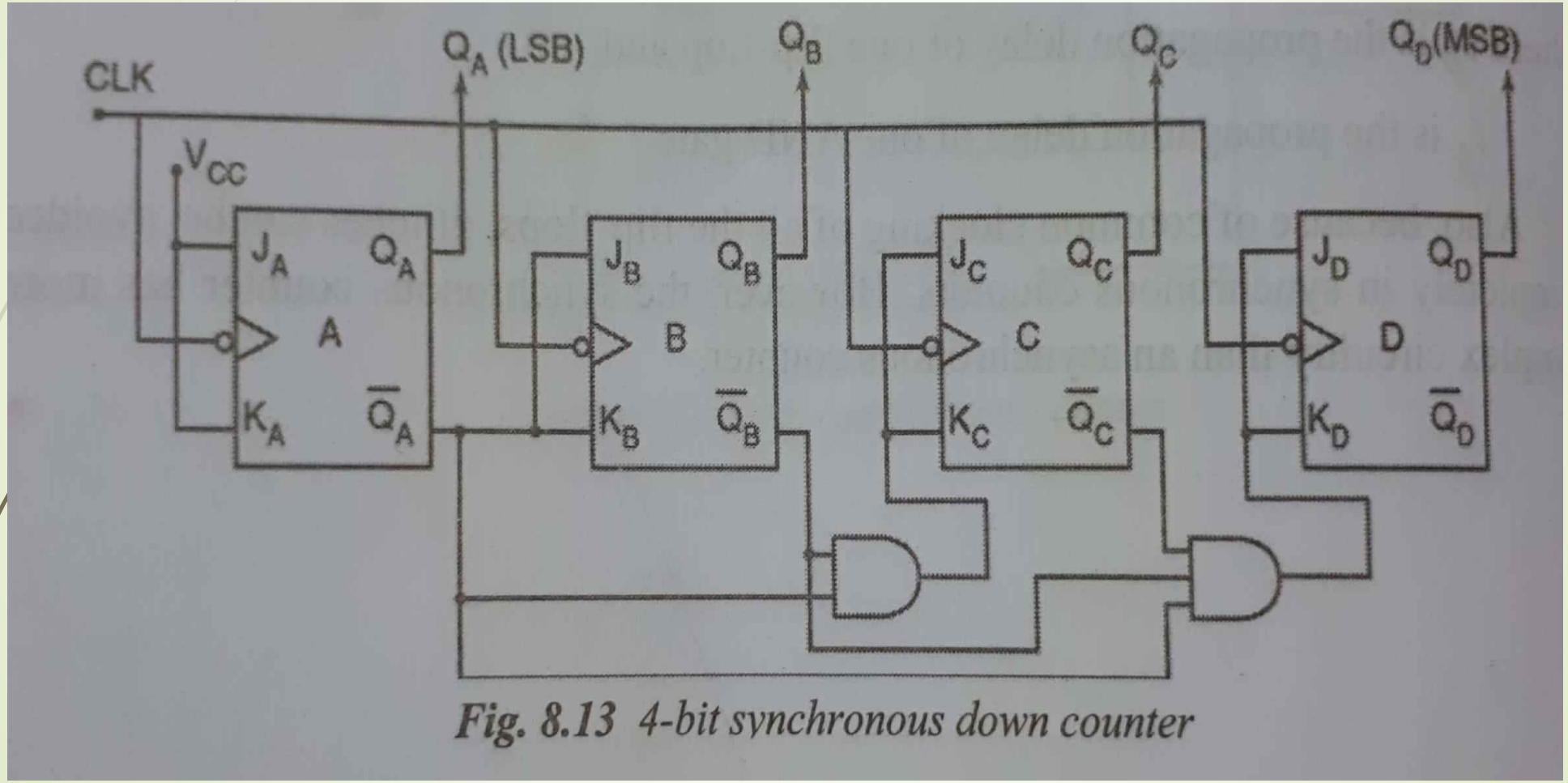


Fig. 8.13 4-bit synchronous down counter

By : Ms. Kanika Dhingra

Table 8.7 Truth table of 4-bit synchronous down counter

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

Synchronous UP/DOWN Counter

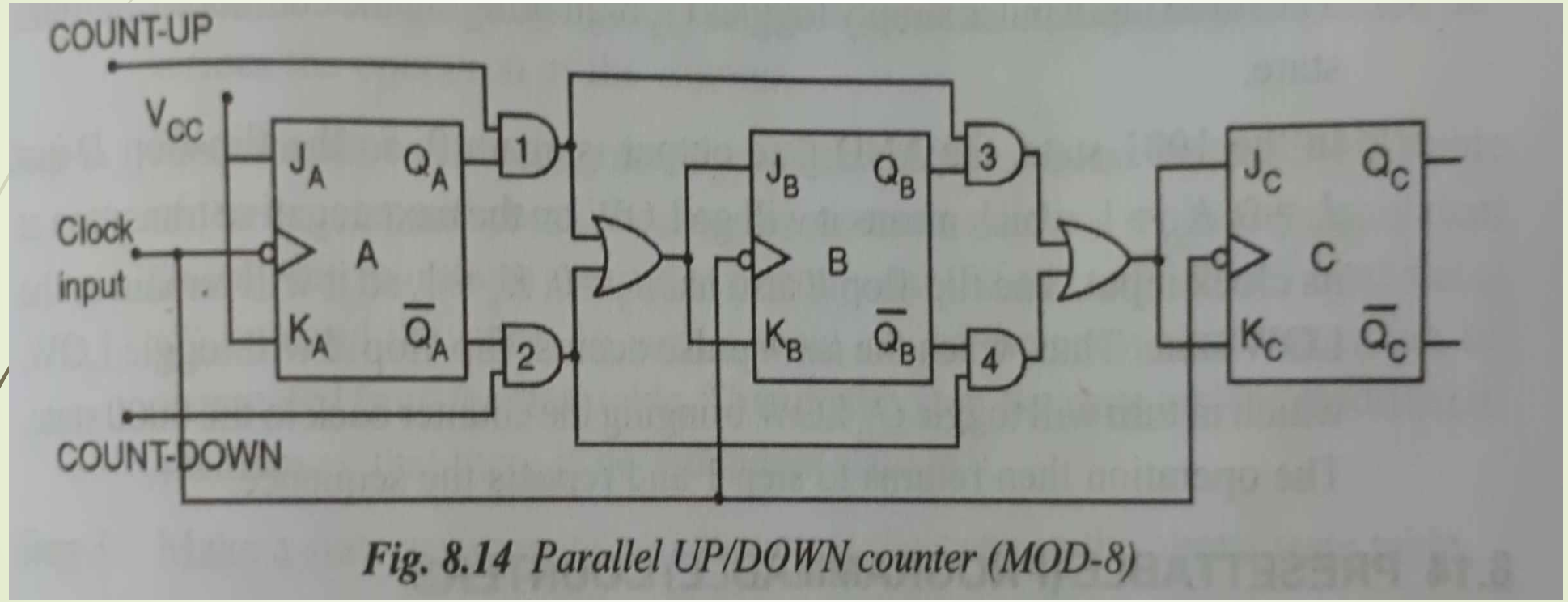


Fig. 8.14 Parallel UP/DOWN counter (MOD-8)

Ring Counter

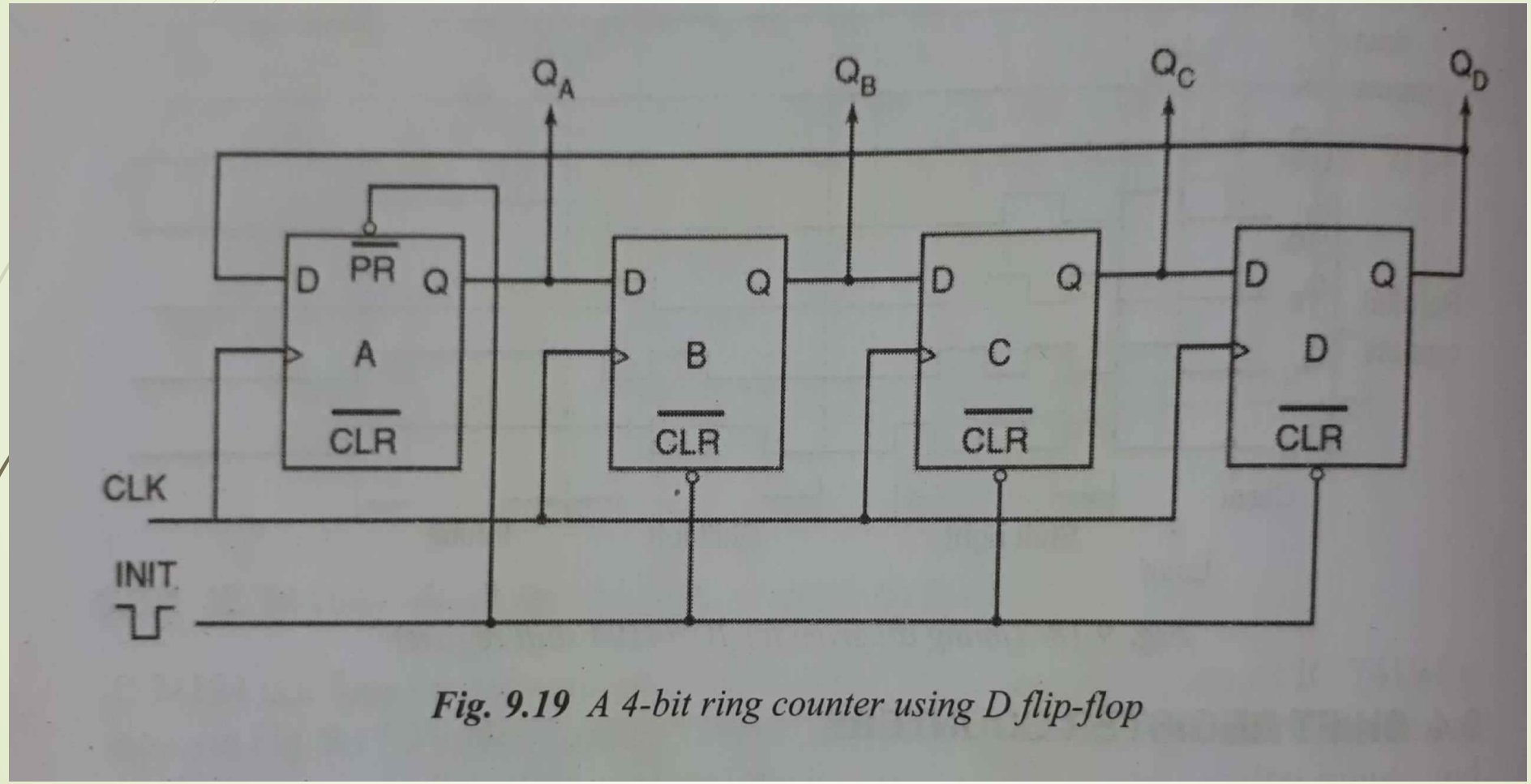


Fig. 9.19 A 4-bit ring counter using D flip-flop

By : Ms. Kanika Dhingra

Table 9.7 Truth table for 4-bit ring counter

INIT	CLK	Q_A	Q_B	Q_C	Q_D
L	X	1	0	0	0
H	↑	0	1	0	0
H	↑	0	0	1	0
H	↑	0	0	0	1
H	↑	1	0	0	0

Table 9.10 Truth table of 4-bit shift counter

CLK	Q_A	Q_B	Q_C	Q_D
0	0	0	0	0
↑	1	0	0	0
↑	1	1	0	0
↑	1	1	1	0
↑	1	1	1	1
↑	0	1	1	1
↑	0	0	1	1
↑	0	0	0	1
↑	0	0	0	0

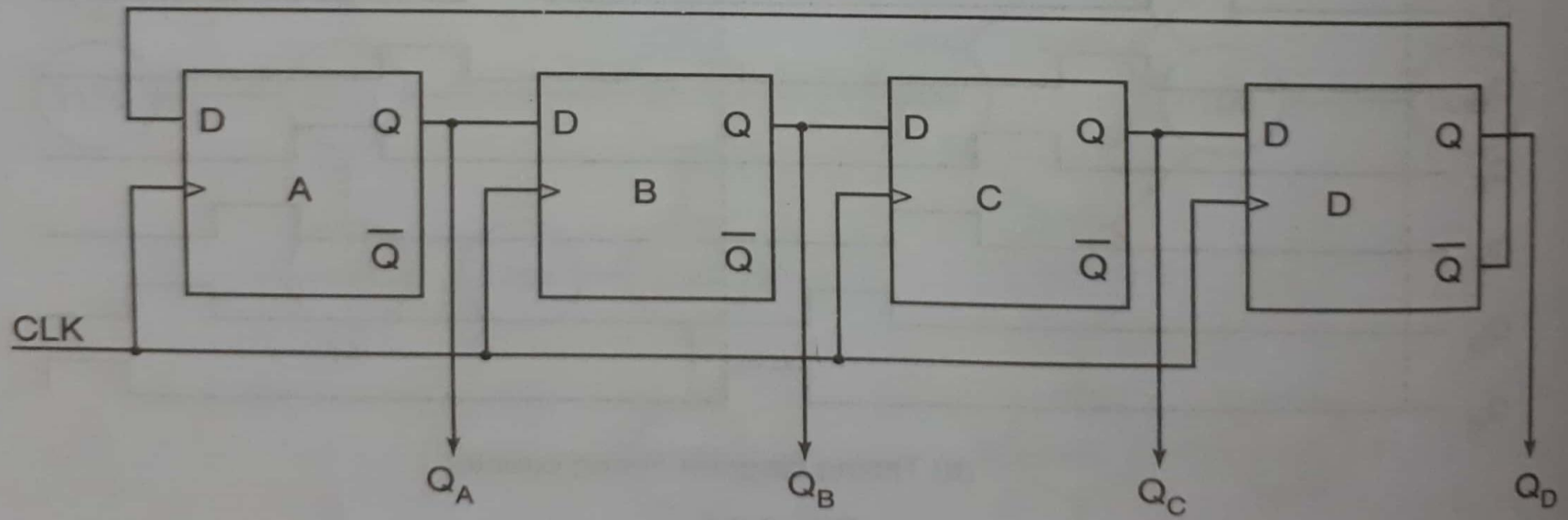


Fig. 9.25 4-bit shift counter

8.15 DESIGN OF SYNCHRONOUS COUNTERS

In this section, the general method to design the different types of synchronous counters using various types of flip-flops is explained in detail.

The steps involved in the design of synchronous counter are listed below:

Step 1 From the word description of the problem, draw a state diagram which describes the operation of the counter.

Step 2 From the above state diagram, obtain Present State – Next State (PS-NS) table of the counter and check the same to ascertain whether it has any equivalent states. Any two states are said to be equivalent, if and only if their next states are one and the same. In such a case, one of the equivalent states can be eliminated from the state table. Thus, in this step, the state table is modified in such a way that there is no redundant state in it.

Step 3 Make a state assignment and document the same in the above state table.

Step 4 Decide the type of memory element to be used in the counter design and then obtain the excitation table from PS–NS table using the application table of the flip-flop.

Step 5 Draw the excitation maps for various excitation inputs of flip-flops and simplify the excitation functions.

Step 6 Draw the schematic diagram of the counter.

To understand the above design procedures, the following counter design problems can be considered, in which q_i represents the present state and Q_i represents the next state of the flip-flop where $i = 0$ to $n-1$, and n is the number of flip-flops used.