Unit 6

Counters and Registers

Introduction

Circuits that include flip-flops are usually classified by the function they perform. Two such circuits are counters and registers.

Counters

- A register that goes through a predetermined sequence of states upon the application of input pulses
- The gates in a counter are connected in such a way as to produce a predefined sequence of binary states in the register.

Registers

- ❖ A group of binary cells (FFs) suitable for holding binary data information

- In addition to the FFs, a register may have combinational gates to control when and how the new information is transferred into the register

Counters

- Counters are circuits that cycle through a specified number of states.

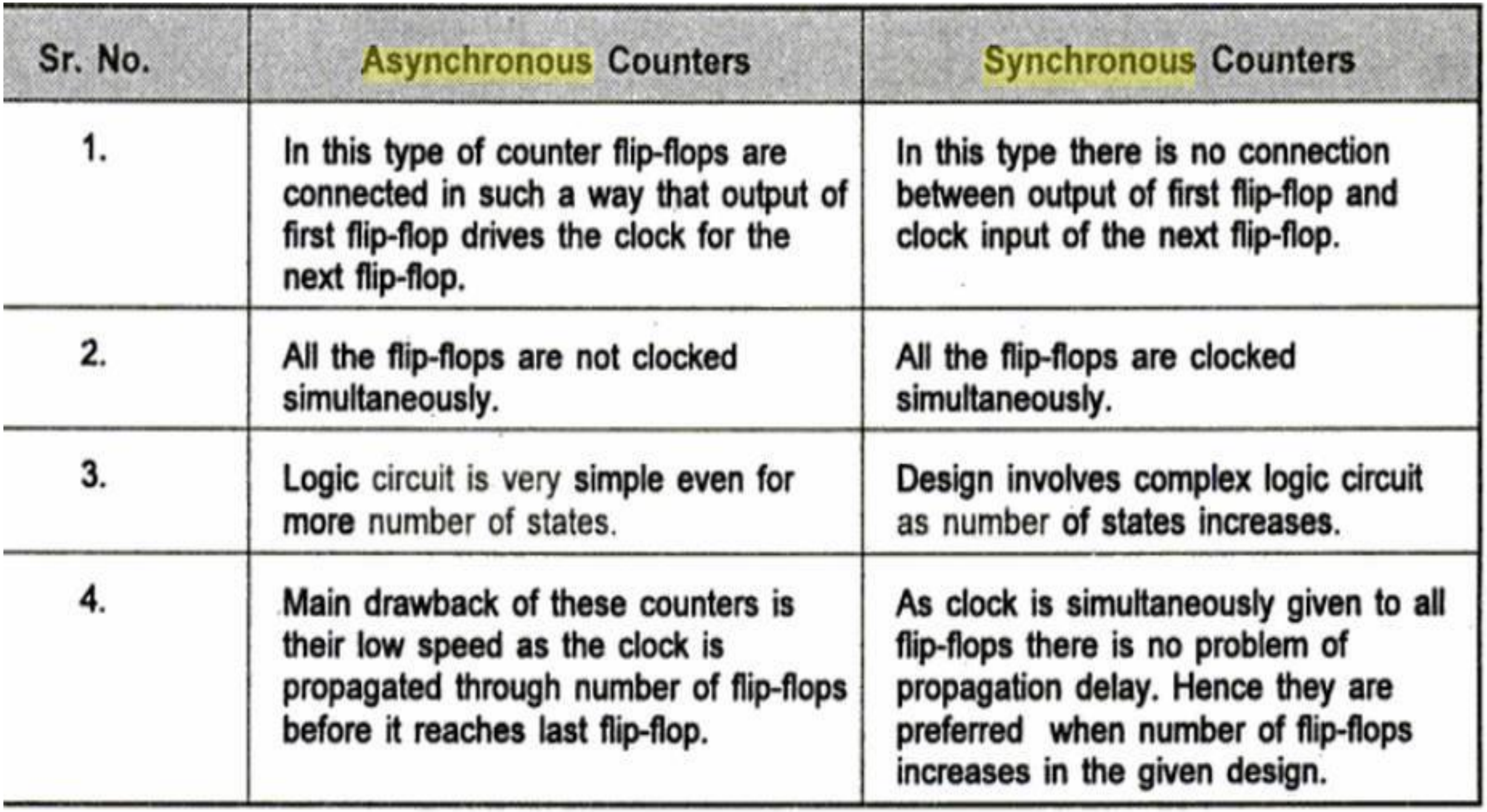
  Two types of counters:

  - Asynchronous (ripple) counters
  - Synchronous (parallel) counters

- Ripple counters allow some flip-flop outputs to be used as a source of clock for other flip-flops.

- Synchronous counters apply the same clock to all flip flops.

Asynchronous Vs synchronous Counters



Sr. No.	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 propagated 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.

Asynchronous (Ripple) Counters

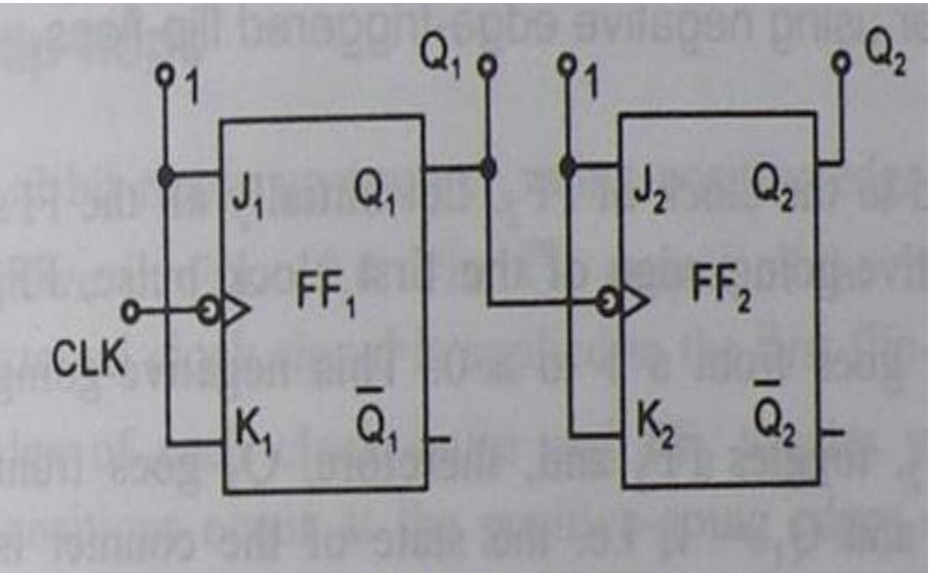
Asynchronous counters: The flip-flops do not change states at exactly the same time as they do not have a common clock pulse.

- Also known as ripple counters, as the input clock pulse “ripples” through the counter – cumulative delay is a drawback.
- *n* flip-flops: a MOD (modulus) 2^n counter. (Note: A MOD-*x* counter cycles through *x* states.)
- Output of the last flip-flop (MSB) divides the input clock frequency by the MOD number of the counter, hence a counter is also a *frequency divider*.

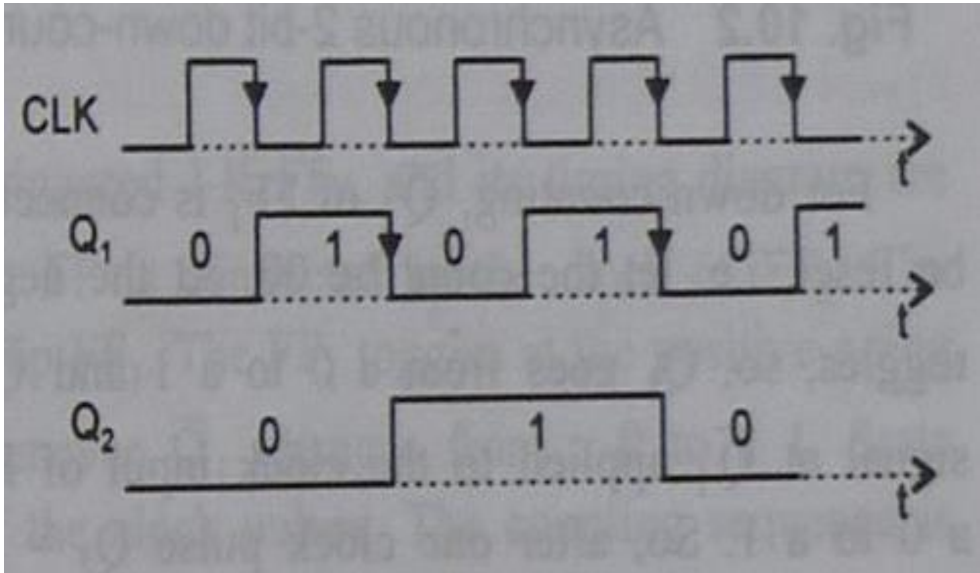
- Idea:
  - to connect the output of one flip-flop to the C input of the next high-order flip-flop
- We need “complementing” flip-flops – We can use T flip-flops to obtain complementing flip flops or
  - JK flip-flops with its inputs are tied together

2 bit Ripple binary up Counters

- Output of one flip-flop is connected to the clock input of the next more-significant flip-flop.



(a) Logic diagram



(b) Timing diagram

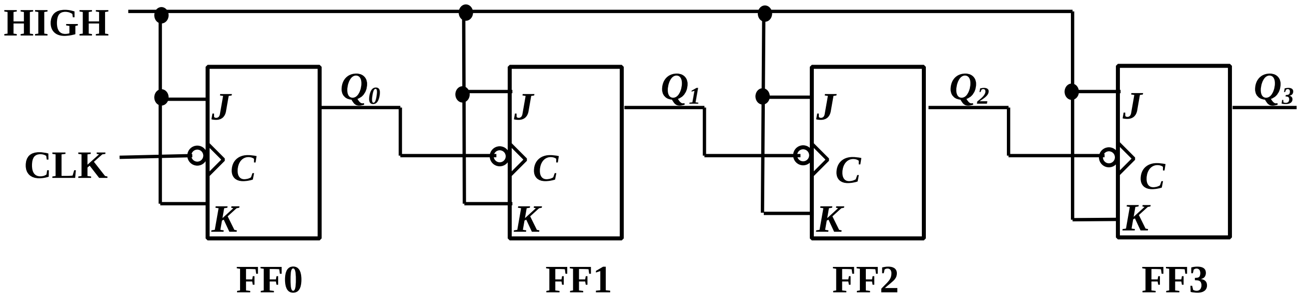
Timing diagram

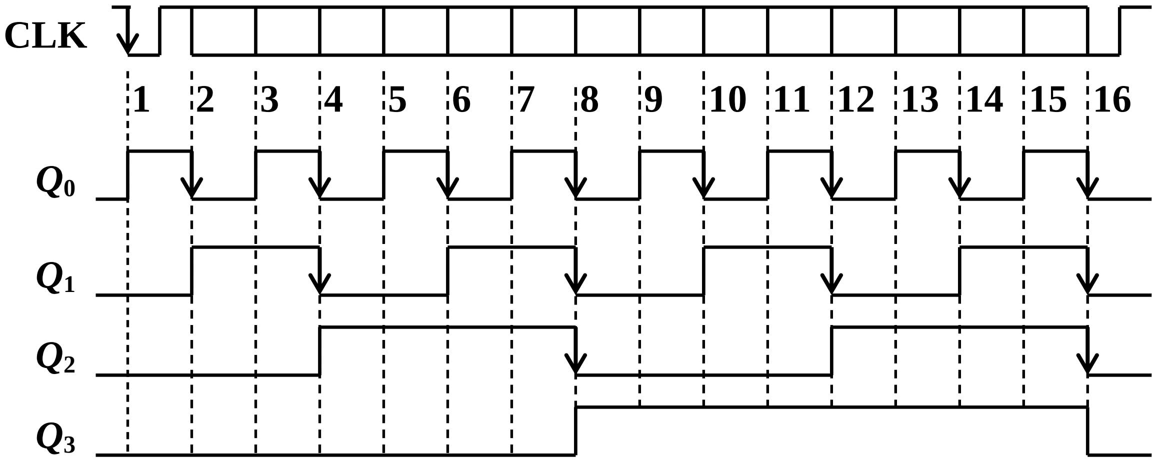
Count Sequence

Q2	Q1
0	0
0	1
1	0
1	1
0	0 ...

Asynchronous (Ripple) Counters

- Example: 4-bit ripple binary counter (negative edge triggered).




CLK
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
Q_0
Q_1
Q_2
Q_3

Binary Ripple Counter

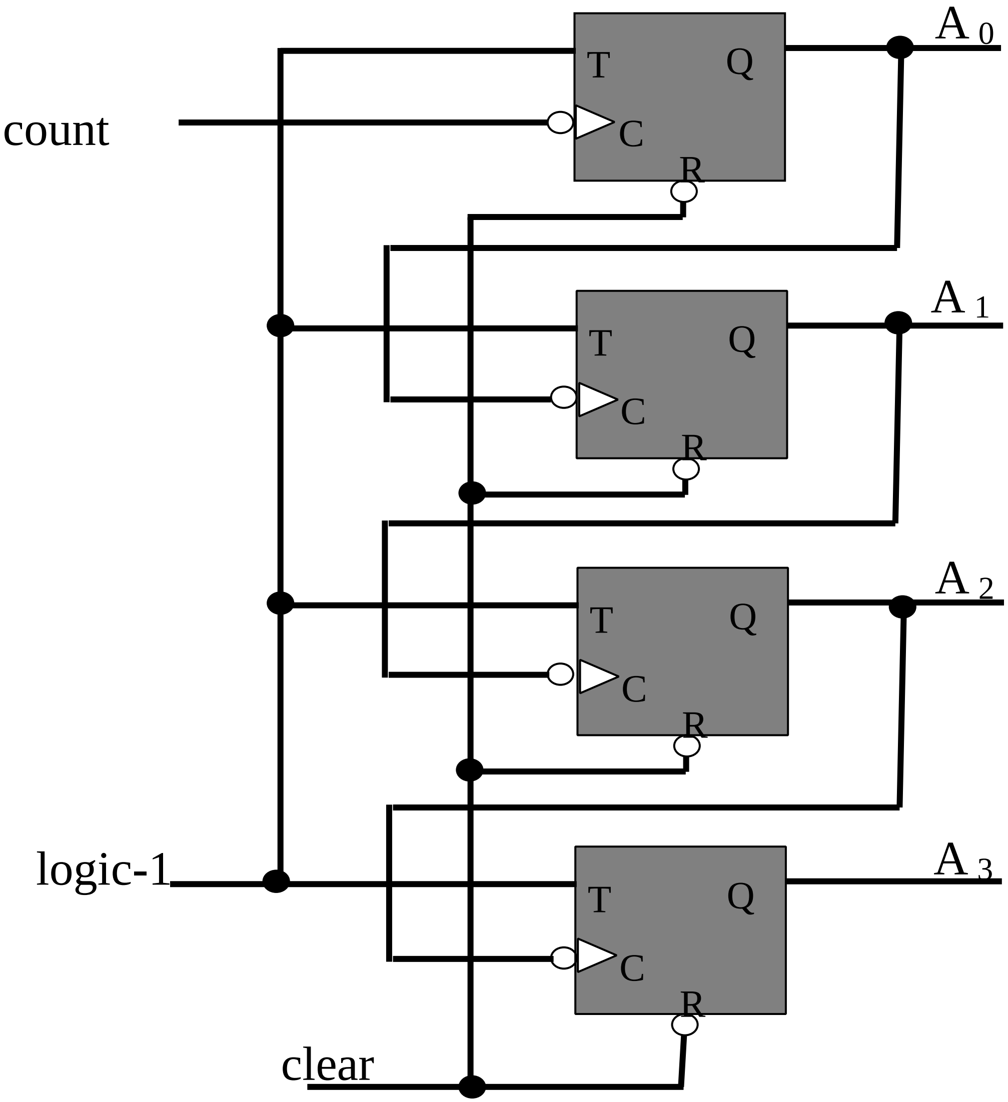
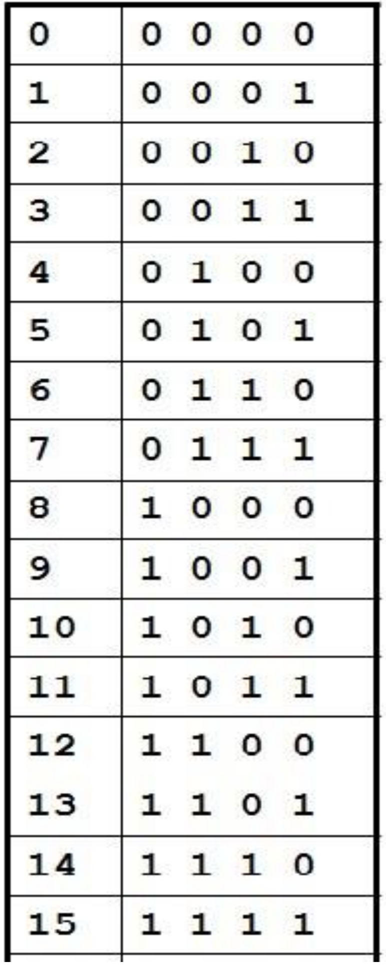


Fig: 4-bit Binary Ripple Counter



	A_3	A_2	A_1	A_0
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

Fig: Count Sequence

Operation

- It is obvious that the lowest-order bit A_0 must be complemented with each count pulse.
- Every time A_0 goes from 1 to 0, it complements A_1.
- Every time A_1 goes from 1 to 0, it complements A_2.
- Every time A_2 goes from 1 to 0, it complements A_3, and so on.
- The flip-flops change one at a time in rapid succession, and the signal propagates through the counter in a ripple fashion.

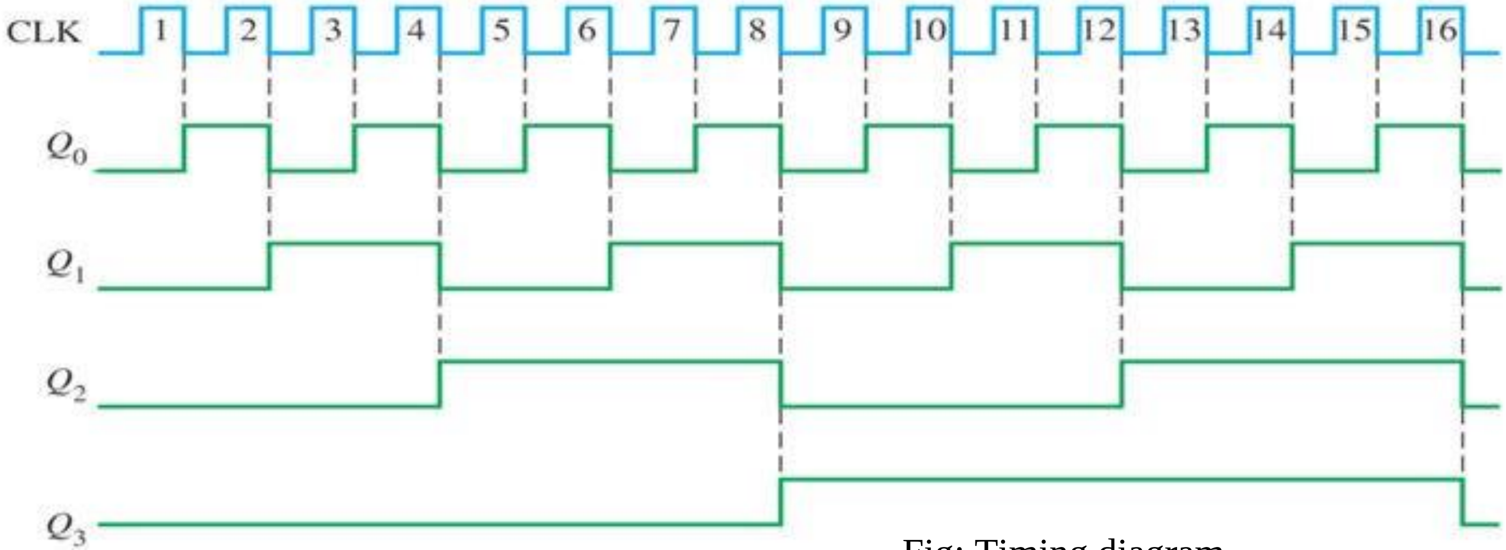


Fig: Timing diagram

BCD Ripple Counter (Decade Counter)

- These counters have 10 states in their sequence.
- A decade counter of states from 0000 to 1001 is called a **BCD decade counter.**
- Since the counter would have 16 states, it is forces to recycle before going through all of its possible state.
- The **BCD decade counter** is recycled to 0000 after the 1001 state.
- Such a counter must have at least four flip-flops to represent each decimal digit, since a decimal digit is represented by a binary code with at least four bits.
- The sequence of states in a decimal counter is dictated by the binary code used to represent a decimal digit.

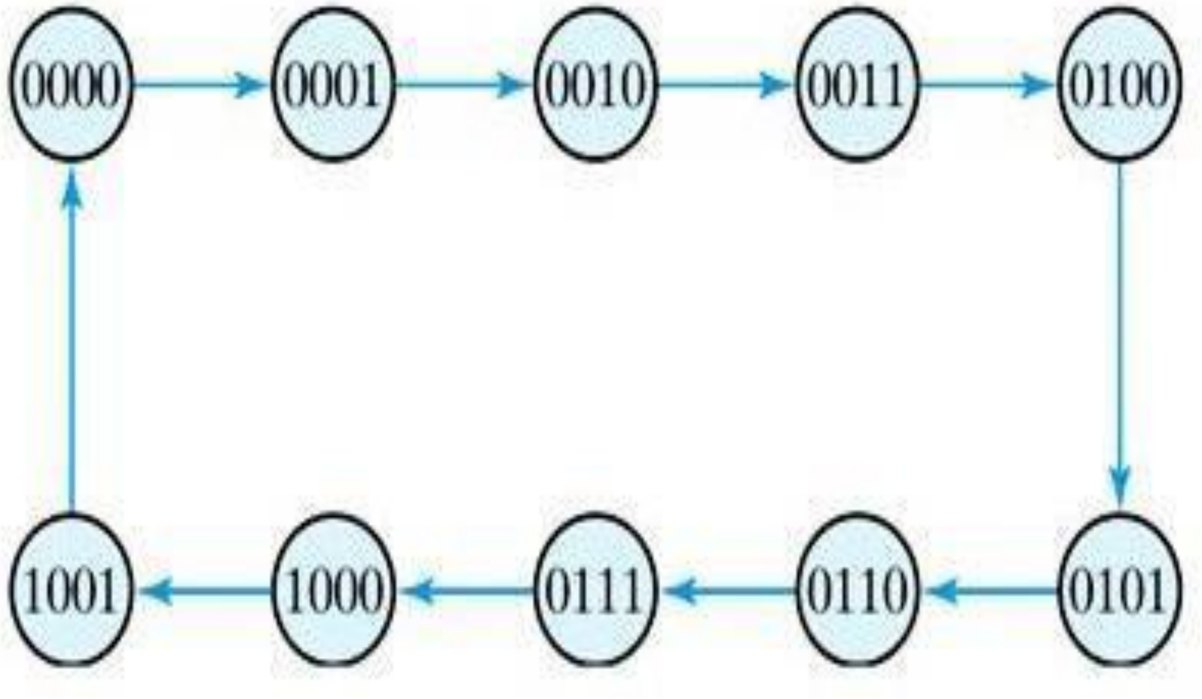
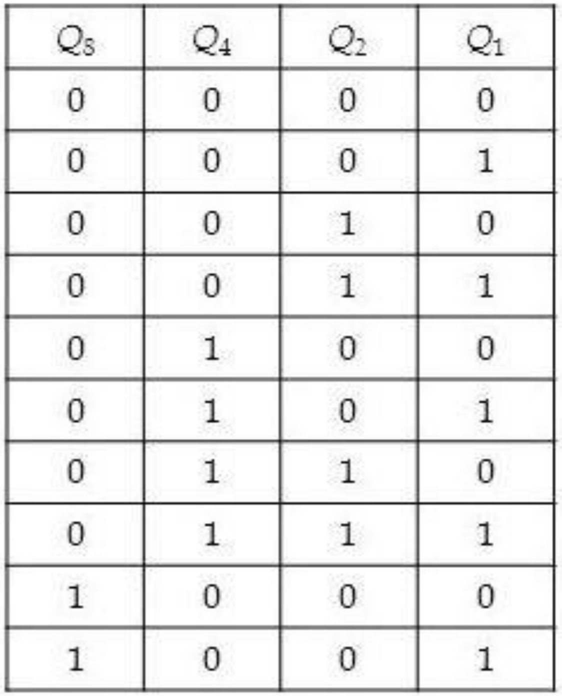


Fig: state diagram

BCD Ripple Counter



Q_8	Q_4	Q_2	Q_1
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1

Fig: count sequence

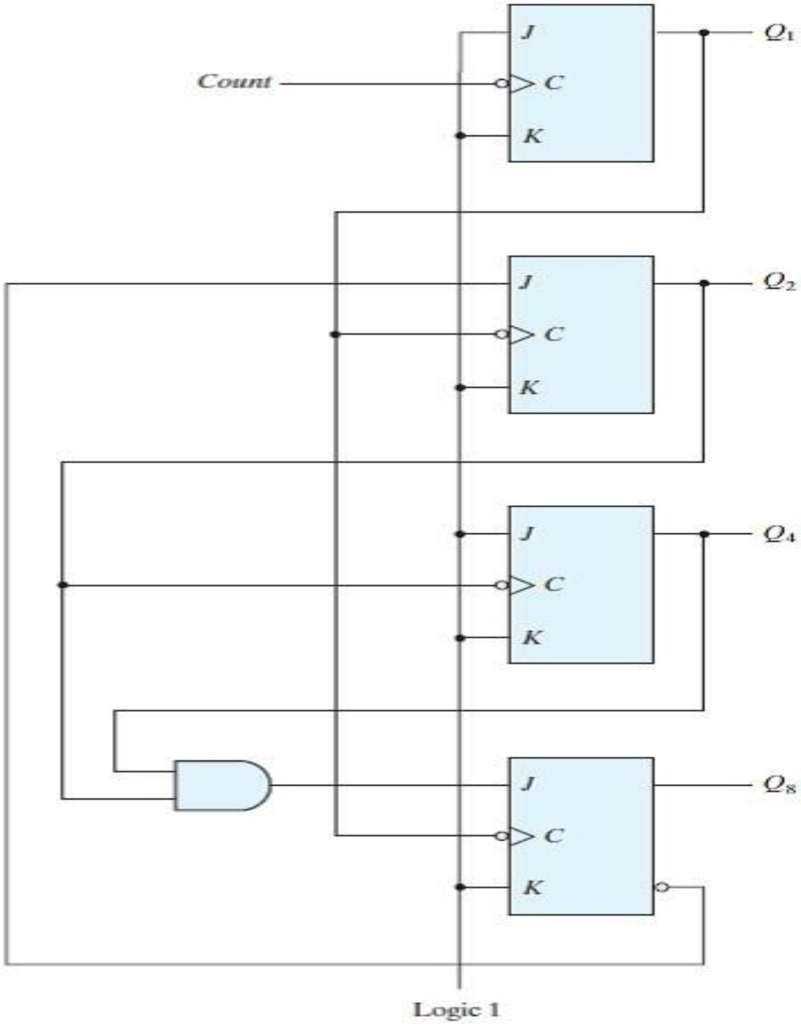


FIGURE
BCD ripple counter

Operation:

- When CP input goes from 1 to 0, the flip-flop is set if J = 1, is cleared if K = 1, is complemented if J = K = 1, and is left unchanged if J = K = 0. The following are the conditions for each flip-flop state transition:

1. Q_1 is complemented on the negative edge of every count pulse.
2. Q_2 is complemented if $Q_8 = 0$ and Q_1 goes from 1 to 0. Q_2 is cleared if $Q_8 = 1$ and Q_1 goes from 1 to 0.
3. Q_4 is complemented when Q_2 goes from 1 to 0.
4. Q_8 is complemented when $Q_4\ Q_2 = 11$ and Q_1 goes from 1 to 0. Q_8 is cleared if either Q_4 or $Q2$ is 0 and $Q1$ goes from 1 to 0.

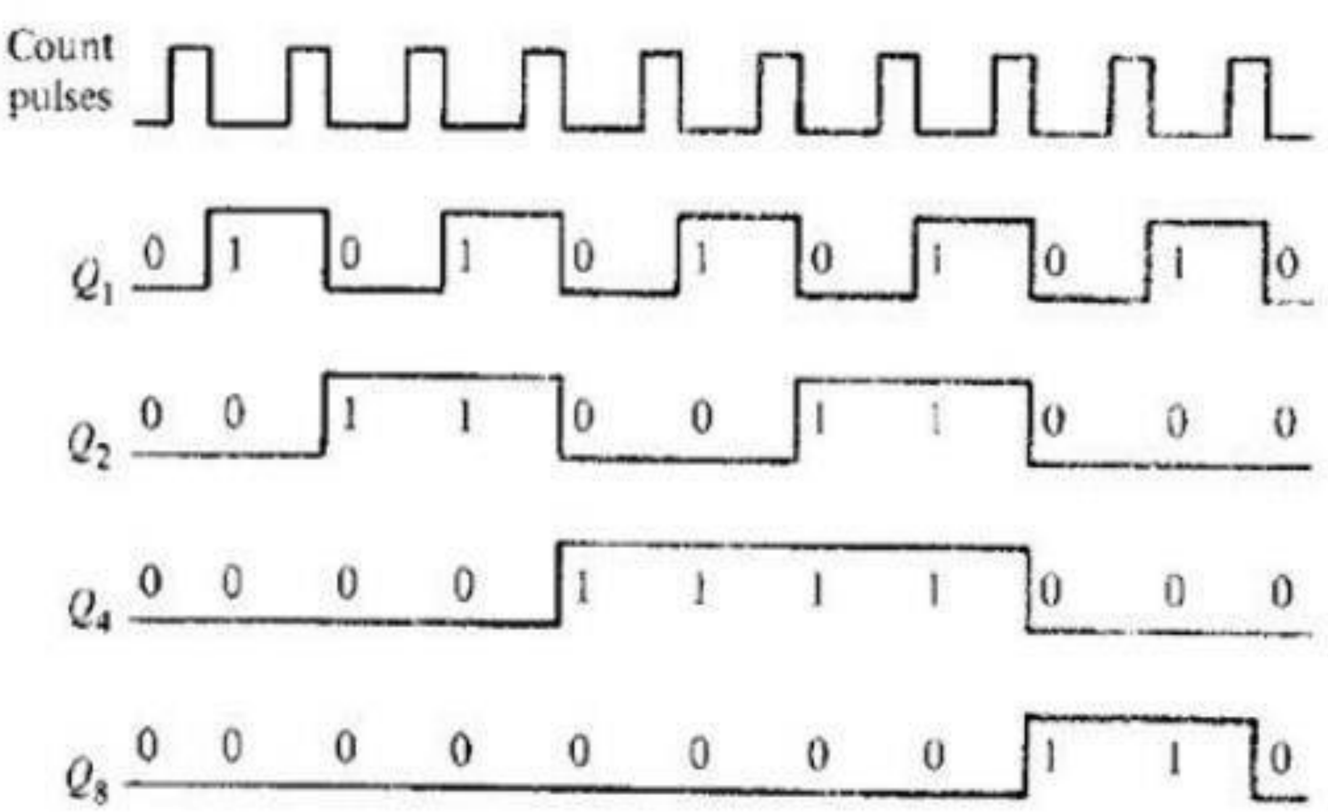
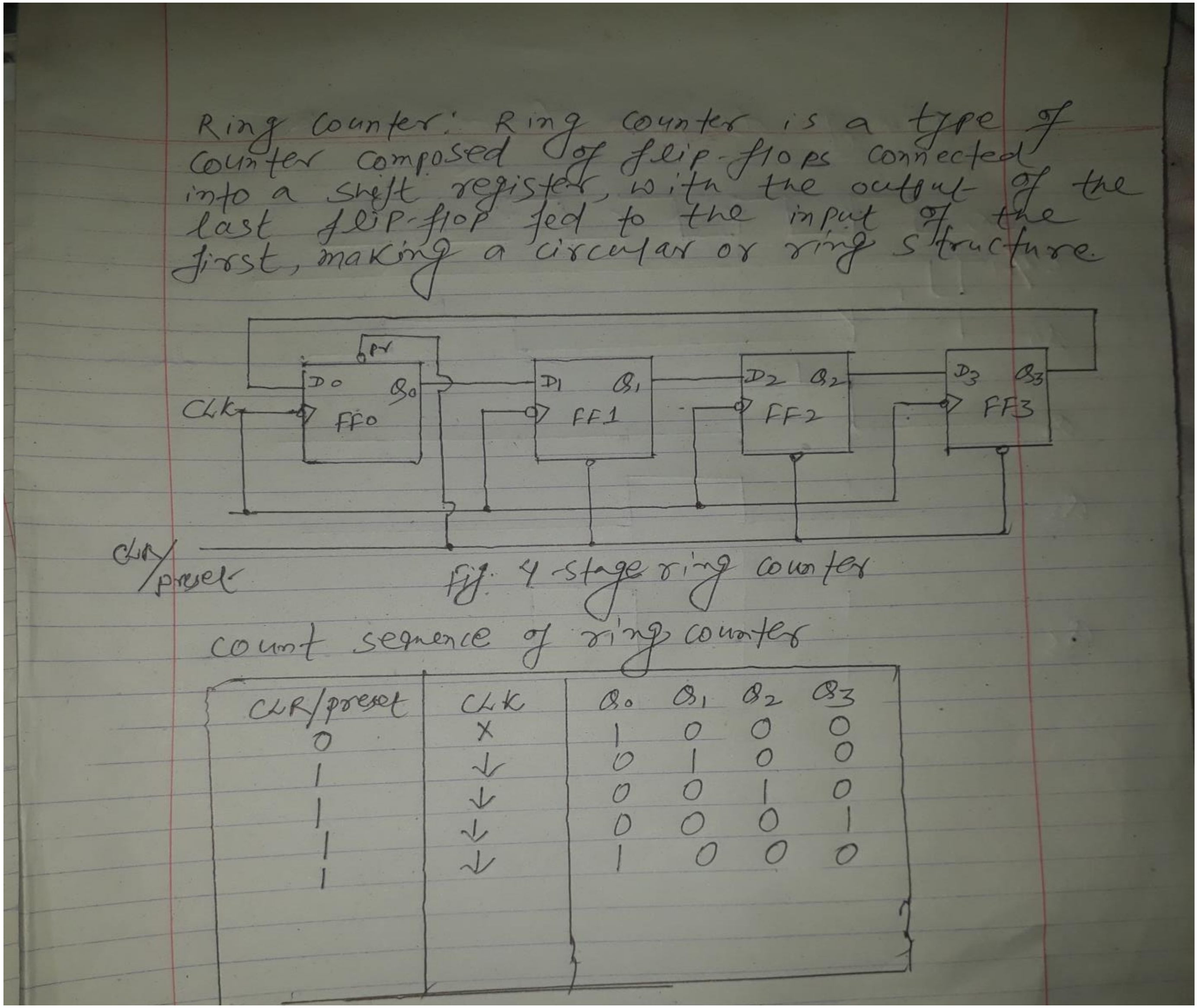


Fig: Timing diagram



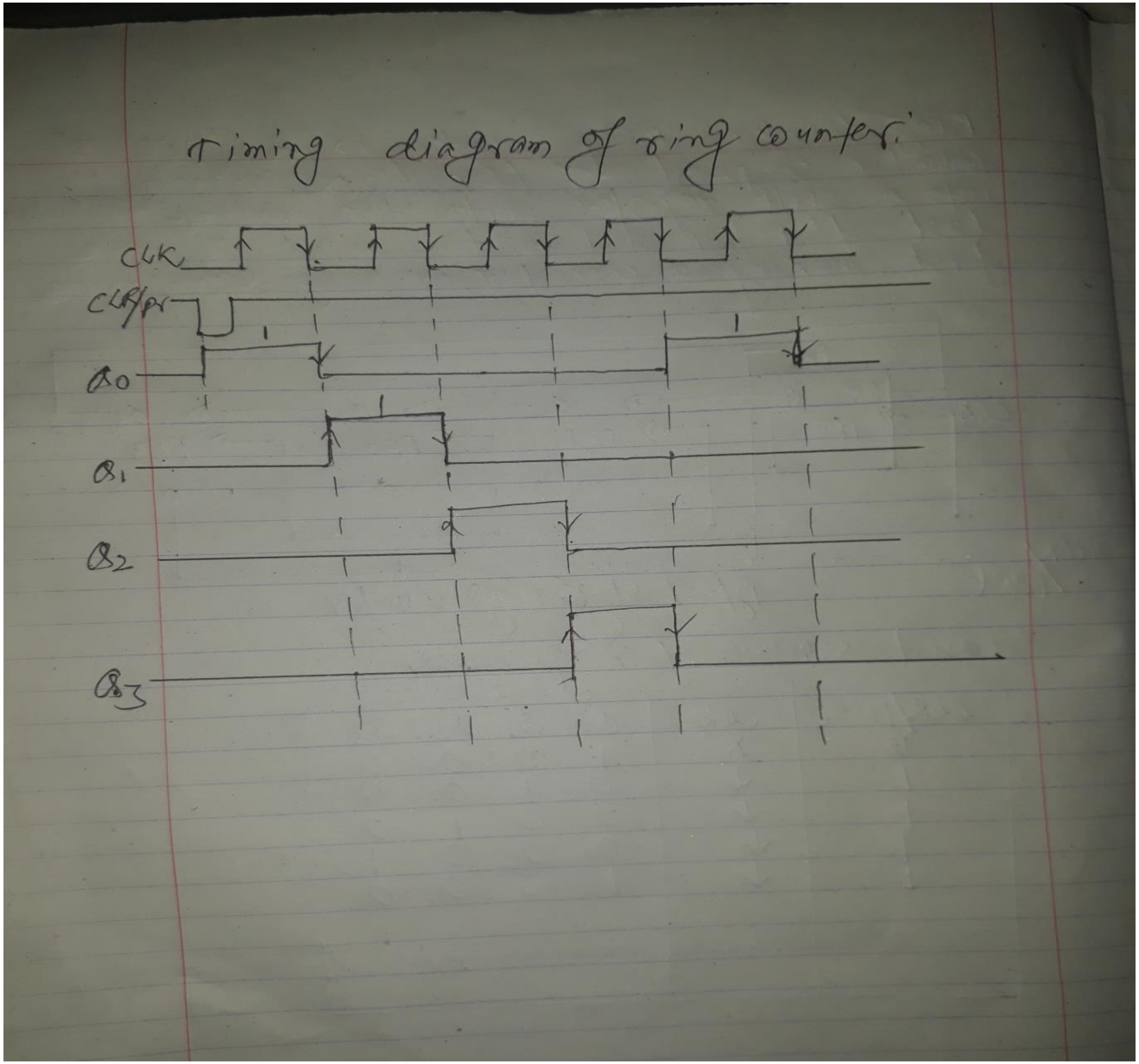
Ring counter: Ring counter is a type of counter composed of flip-flops connected into a shift register, with the output of the last flip-flop fed to the input of the first, making a circular or ring structure.

PR
D0
Q0
FF0
CLK
D1
Q1
FF1
D2
Q2
FF2
D3
Q3
FF3
CLR/preset

fig. 4-stage ring counter

count sequence of ring counter

CLR/preset	CLK	Q0	Q1	Q2	Q3
0	X	1	0	0	0
1	↓	0	1	0	0
1	↓	0	0	1	0
1	↓	0	0	0	1
1	↓	1	0	0	0
1					

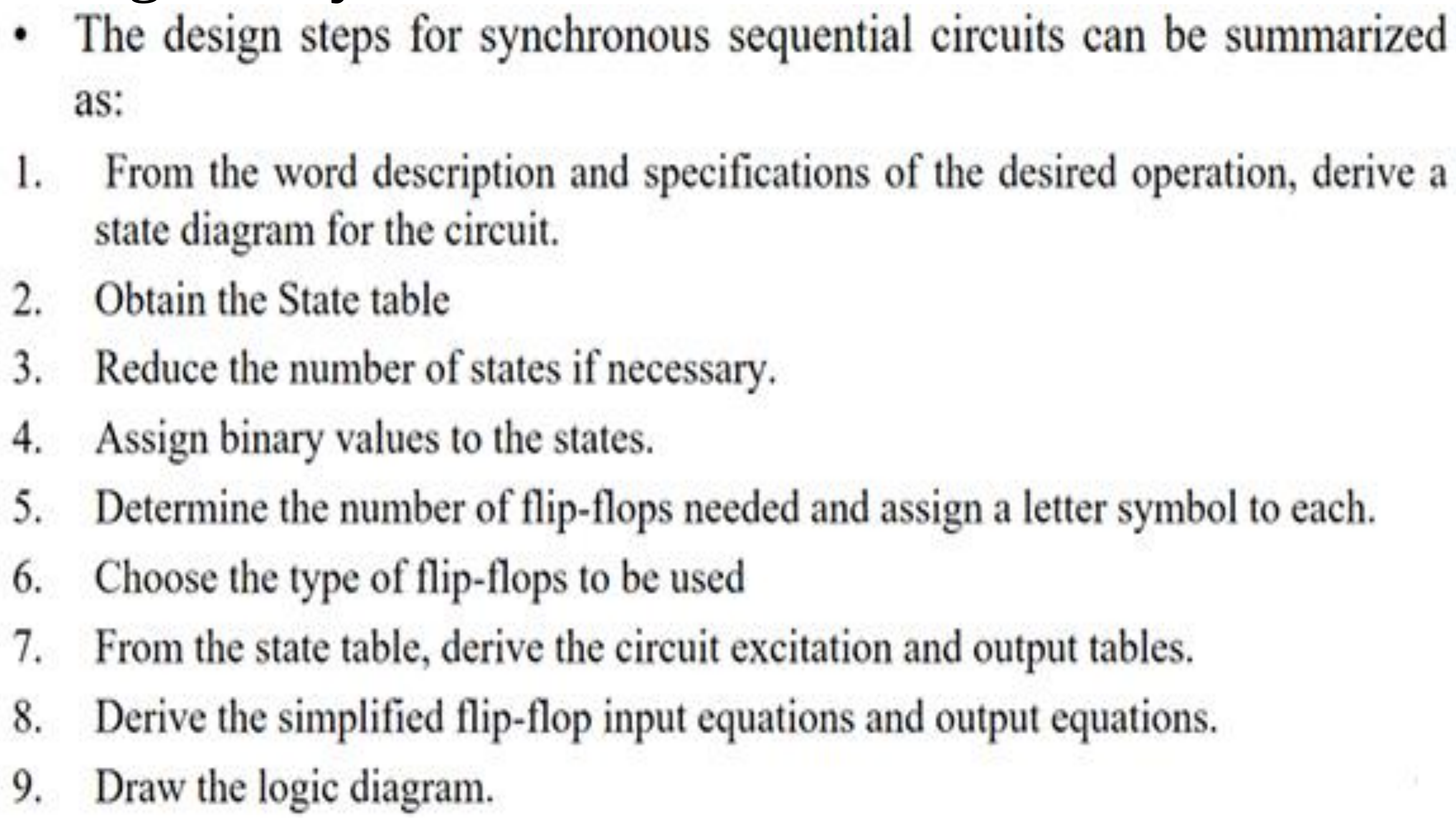


Timing diagram of ring counter.

Synchronous (Parallel) Counters

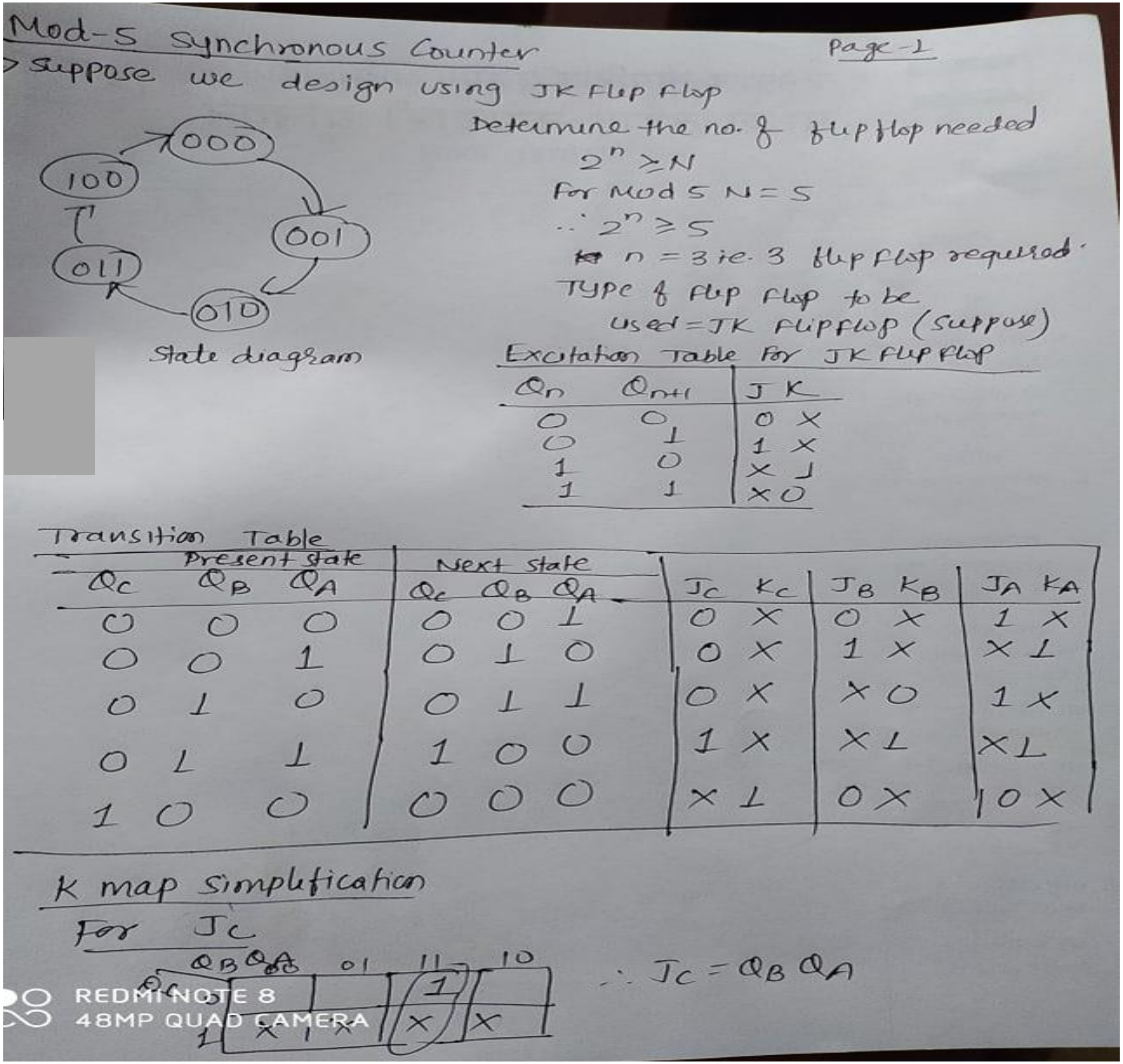
- Synchronous counters are distinguished from ripple counters in that clock pulses are applied to the *CP* inputs of *all* flip-flops.
- Synchronous counters are counters in which all the flip-flops are triggered simultaneously (in parallel) by the clock input pulses.
- All the FFs change state simultaneously in synchronization with the clock pulse.
- The decision whether a flip-flop is to be complemented or not is determined from the values of the J and K inputs at the time of the pulse.
  - If $J = K = 0$, the flip-flop remains unchanged. If $J = K = 1$, the flip flop complements.
- Synchronous counters have the advantage of high speed but the disadvantage of having more circuitry than that of asynchronous counters

Design of Synchronous Counter



- The design steps for synchronous sequential circuits can be summarized as:

1. From the word description and specifications of the desired operation, derive a state diagram for the circuit.
2. Obtain the State table
3. Reduce the number of states if necessary.
4. Assign binary values to the states.
5. Determine the number of flip-flops needed and assign a letter symbol to each.
6. Choose the type of flip-flops to be used
7. From the state table, derive the circuit excitation and output tables.
8. Derive the simplified flip-flop input equations and output equations.
9. Draw the logic diagram.


000
001
010
011
100

Mod-5 Synchronous Counter

Page-1

> Suppose we design using JK flip flop

State diagram

Determine the no. of flip flop needed

$2^n \geq N$

For Mod 5 N = 5

$\therefore 2^n \geq 5$

$n = 3$ i.e. 3 flip flop required.

Type of flip flop to be used = JK flipflop (suppose)

Excitation Table For JK flip flop

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

Transition Table

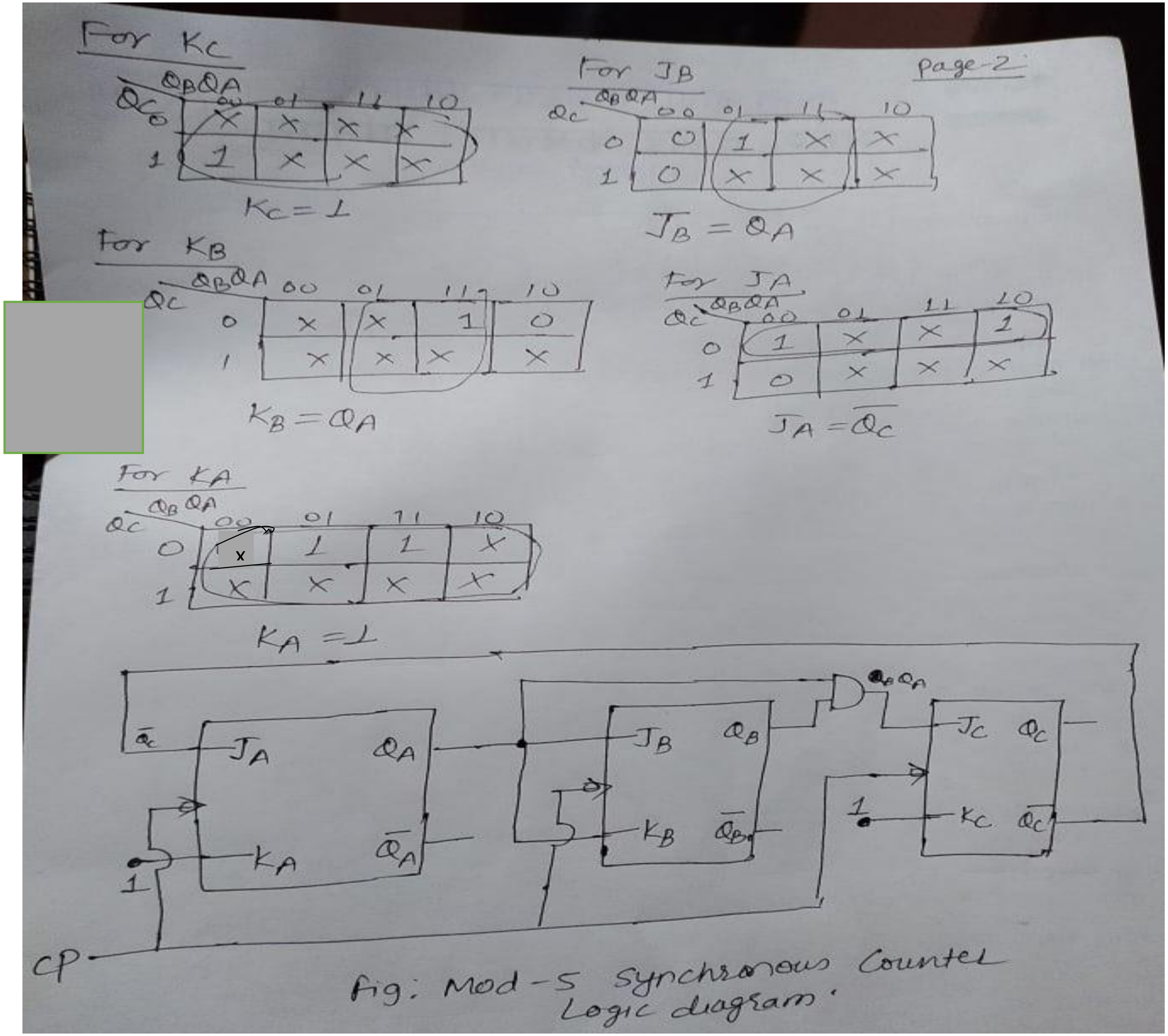
Present state			Next state								
Q_C	Q_B	Q_A	Q_C	Q_B	Q_A	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	0	0	0	X	1	0	X	0	X

K map simplification

For J_C

Q_C \ Q_BQ_A	00	01	11	10
0			1	
1	X	X	X	X

$\therefore J_C = Q_B Q_A$



Page-2

For K_C

Q_C \ Q_BQ_A	00	01	11	10
0	X	X	X	X
1	1	X	X	X

$K_C = 1$

For J_B

Q_C \ Q_BQ_A	00	01	11	10
0	0	1	X	X
1	0	X	X	X

$J_B = Q_A$

For K_B

Q_C \ Q_BQ_A	00	01	11	10
0	X	X	1	0
1	X	X	X	X

$K_B = Q_A$

For J_A

Q_C \ Q_BQ_A	00	01	11	10
0	1	X	X	1
1	0	X	X	X

$J_A = \overline{Q_C}$

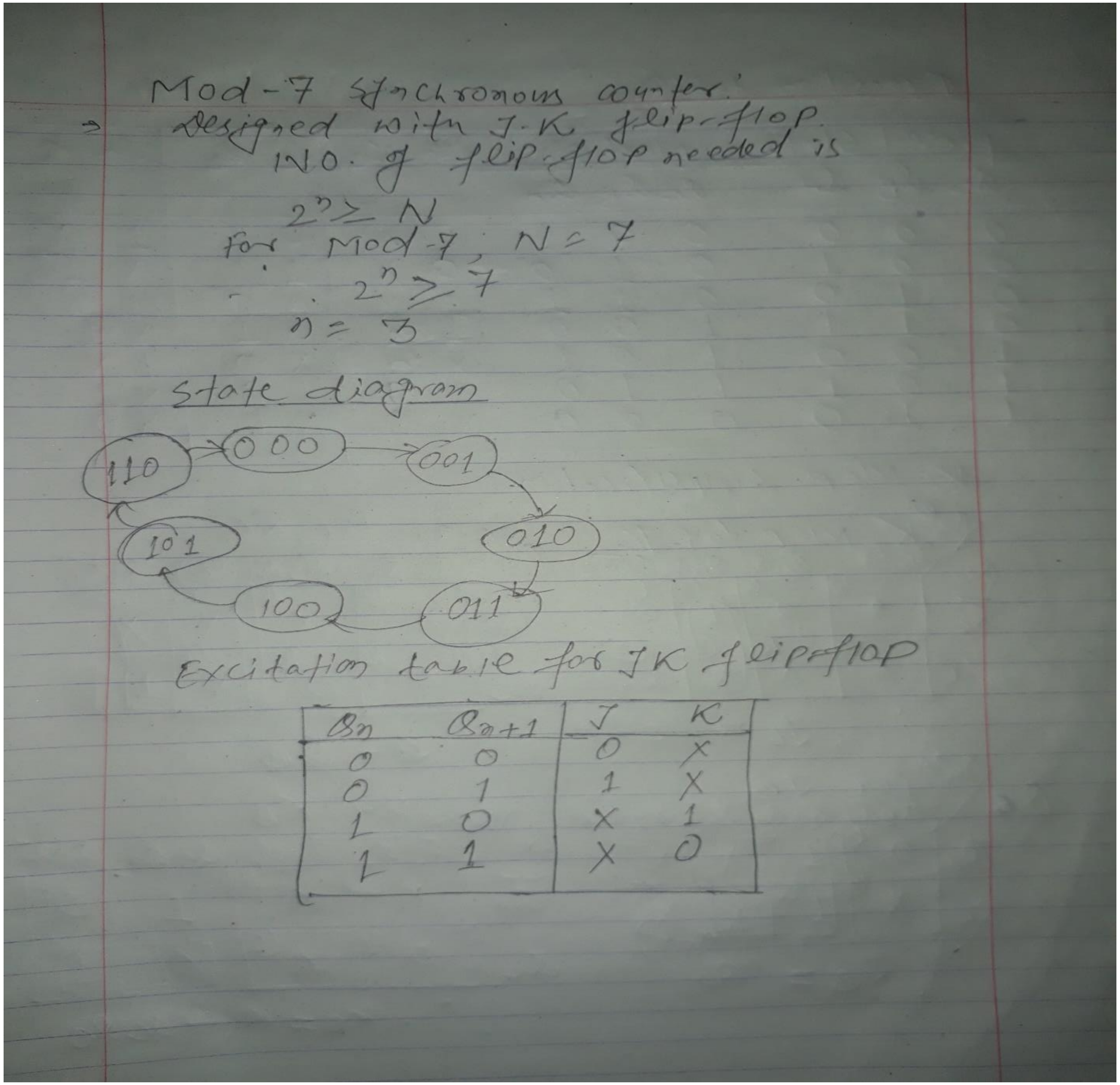
For K_A

Q_C \ Q_BQ_A	00	01	11	10
0	x	1	1	X
1	X	X	X	X

$K_A = 1$

$\bar{Q}_c$
J_A
Q_A
K_A
$\bar{Q}_A$
1
J_B
Q_B
K_B
$\bar{Q}_B$
Q_BQ_A
J_C
Q_C
K_C
$\bar{Q}_C$
1
CP

fig: Mod-5 Synchronous Counter Logic diagram.



Mod-7 Synchronous counter.

→ Designed with J-K flip-flop.

No. of flip-flop needed is

$2^n \geq N$

for Mod-7, N = 7

∴ $2^n \geq 7$

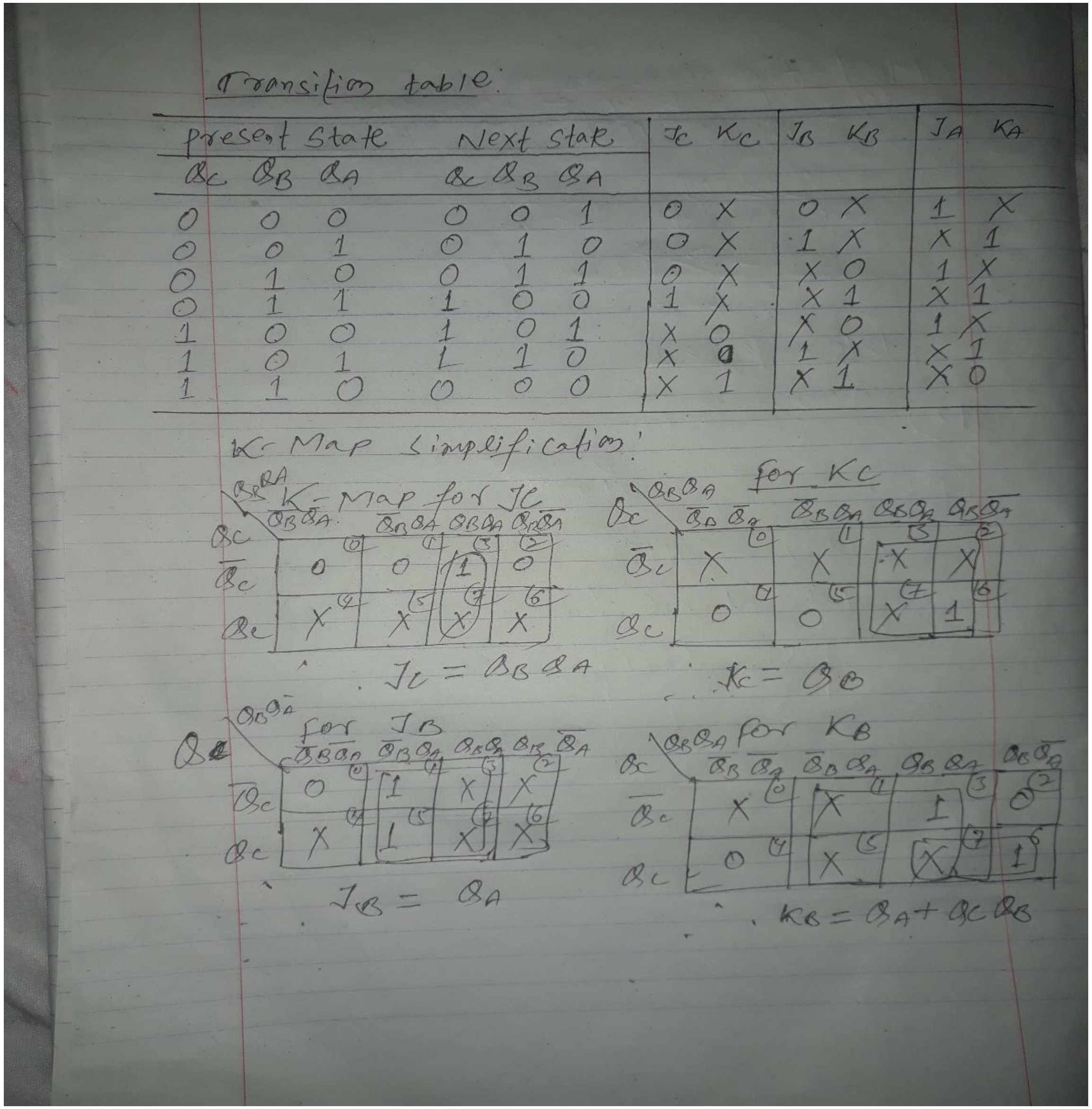
$n = 3$

State diagram

000
001
010
011
100
101
110

Excitation table for JK flip-flop

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



Transition table:

present State			Next State								
Q_C	Q_B	Q_A	Q_C	Q_B	Q_A	J_C	K_C	J_B	K_B	J_A	K_A
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	X	0	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	0	0	0	X	1	X	1	X	0

K-Map simplification:

K-Map for J_C

$Q_C \backslash Q_B Q_A$	$\bar{Q}_B \bar{Q}_A$	$\bar{Q}_B Q_A$	$Q_B Q_A$	$Q_B \bar{Q}_A$
$\bar{Q}_C$	0 ⓪	0 ①	1 ③	0 ②
Q_C	X ④	X ⑤	X ⑦	X ⑥

$\therefore J_C = Q_B Q_A$

for K_C

$Q_C \backslash Q_B Q_A$	$\bar{Q}_B \bar{Q}_A$	$\bar{Q}_B Q_A$	$Q_B Q_A$	$Q_B \bar{Q}_A$
$\bar{Q}_C$	X ⓪	X ①	X ③	X ②
Q_C	0 ④	0 ⑤	X ⑦	1 ⑥

$\therefore K_C = Q_B$

for J_B

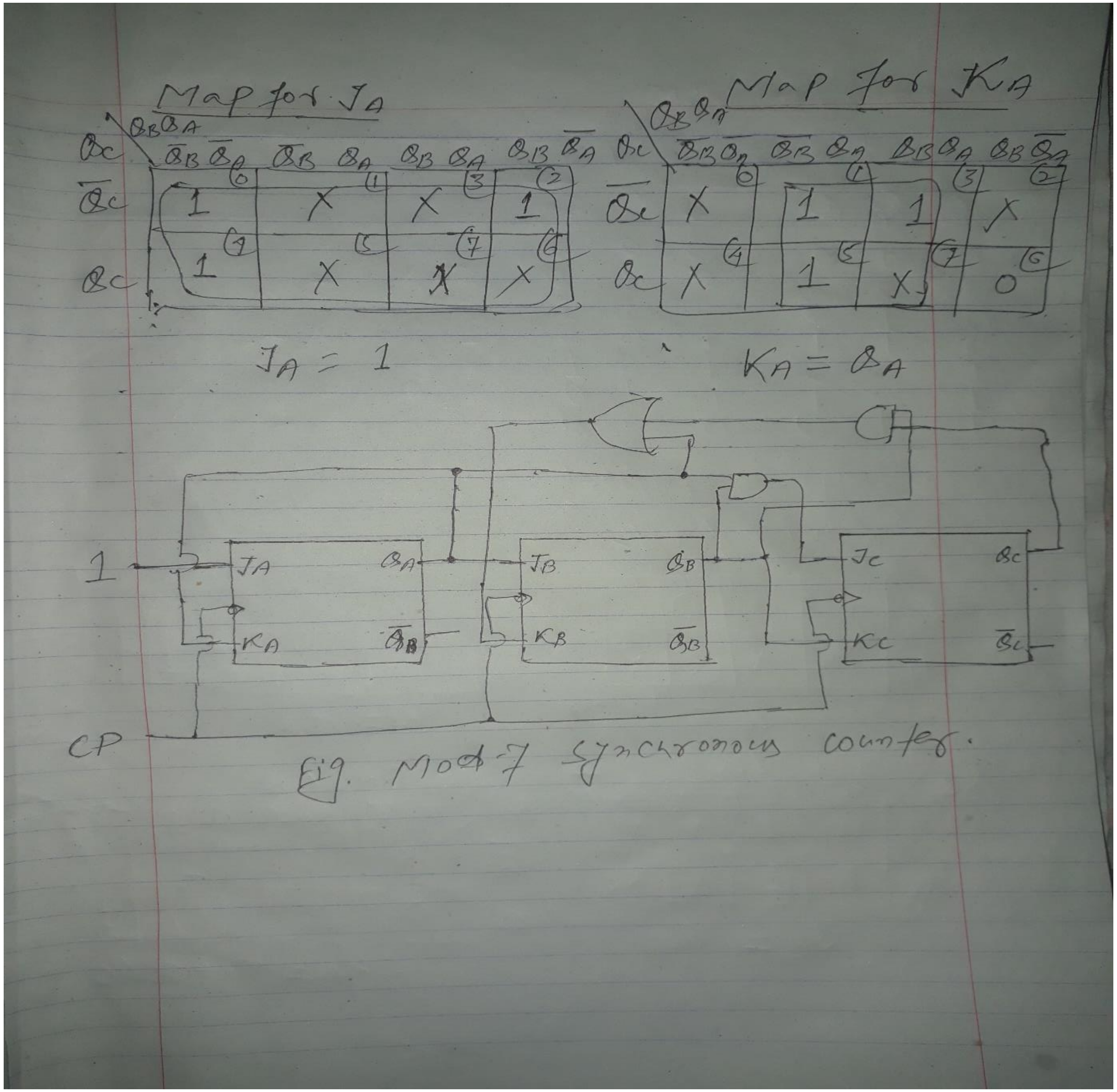
$Q_C \backslash Q_B Q_A$	$\bar{Q}_B \bar{Q}_A$	$\bar{Q}_B Q_A$	$Q_B Q_A$	$Q_B \bar{Q}_A$
$\bar{Q}_C$	0 ⓪	1 ①	X ③	X ②
Q_C	X ④	1 ⑤	X ⑦	X ⑥

$\therefore J_B = Q_A$

for K_B

$Q_C \backslash Q_B Q_A$	$\bar{Q}_B \bar{Q}_A$	$\bar{Q}_B Q_A$	$Q_B Q_A$	$Q_B \bar{Q}_A$
$\bar{Q}_C$	X ⓪	X ①	1 ③	0 ②
Q_C	0 ④	X ⑤	X ⑦	1 ⑥

$\therefore K_B = Q_A + Q_C Q_B$



Map for J_A

$Q_C \backslash Q_BQ_A$	$\bar{Q}_B\bar{Q}_A$	$\bar{Q}_BQ_A$	Q_BQ_A	$Q_B\bar{Q}_A$
$\bar{Q}_C$	(0) 1	(1) X	(3) X	(2) 1
Q_C	(4) 1	(5) X	(7) X	(6) X

Map for K_A

$Q_C \backslash Q_BQ_A$	$\bar{Q}_B\bar{Q}_A$	$\bar{Q}_BQ_A$	Q_BQ_A	$Q_B\bar{Q}_A$
$\bar{Q}_C$	(0) X	(1) 1	(3) 1	(2) X
Q_C	(4) X	(5) 1	(7) X	(6) 0

$J_A = 1$

$K_A = Q_A$

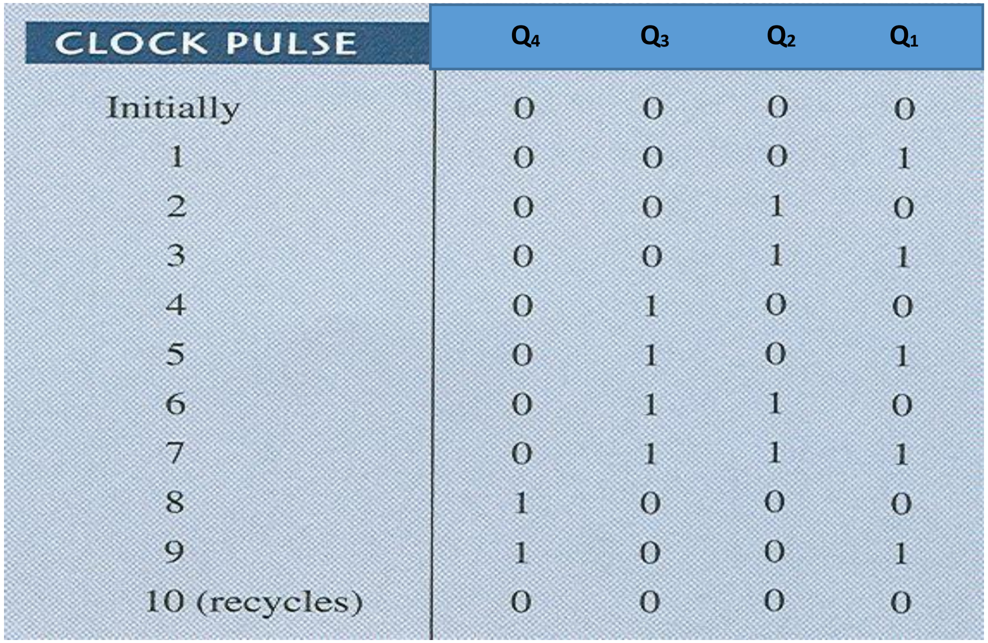
Fig. Mod-7 synchronous counter.

Synchronous BCD Counter

- A BCD Counter is nothing but a mod 10 counter.
- It requires 4 FFs
- A BCD counter counts in binary-coded decimal from 0000 to 1001 and back to 0000 where states 0000 through 1001 are stable.
- States 1010 through 1111 are invalid.
- After the tenth clock pulse, the counter resets.
- Because of the return to 0 after a count of 9, a BCD counter does not have a regular pattern as in a straight binary count.
- To derive the circuit of a BCD synchronous counter, it is necessary to go through a design procedure discussed earlier.

- The flip-flop input functions from the excitation table can be simplified by means of maps.
  - The unused states for minterms 10 to 15 are taken as don't-care terms.

Synchronous BCD Counter



CLOCK PULSE	Q_4	Q_3	Q_2	Q_1
Initially	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 (recycles)	0	0	0	0

Fig: count sequence

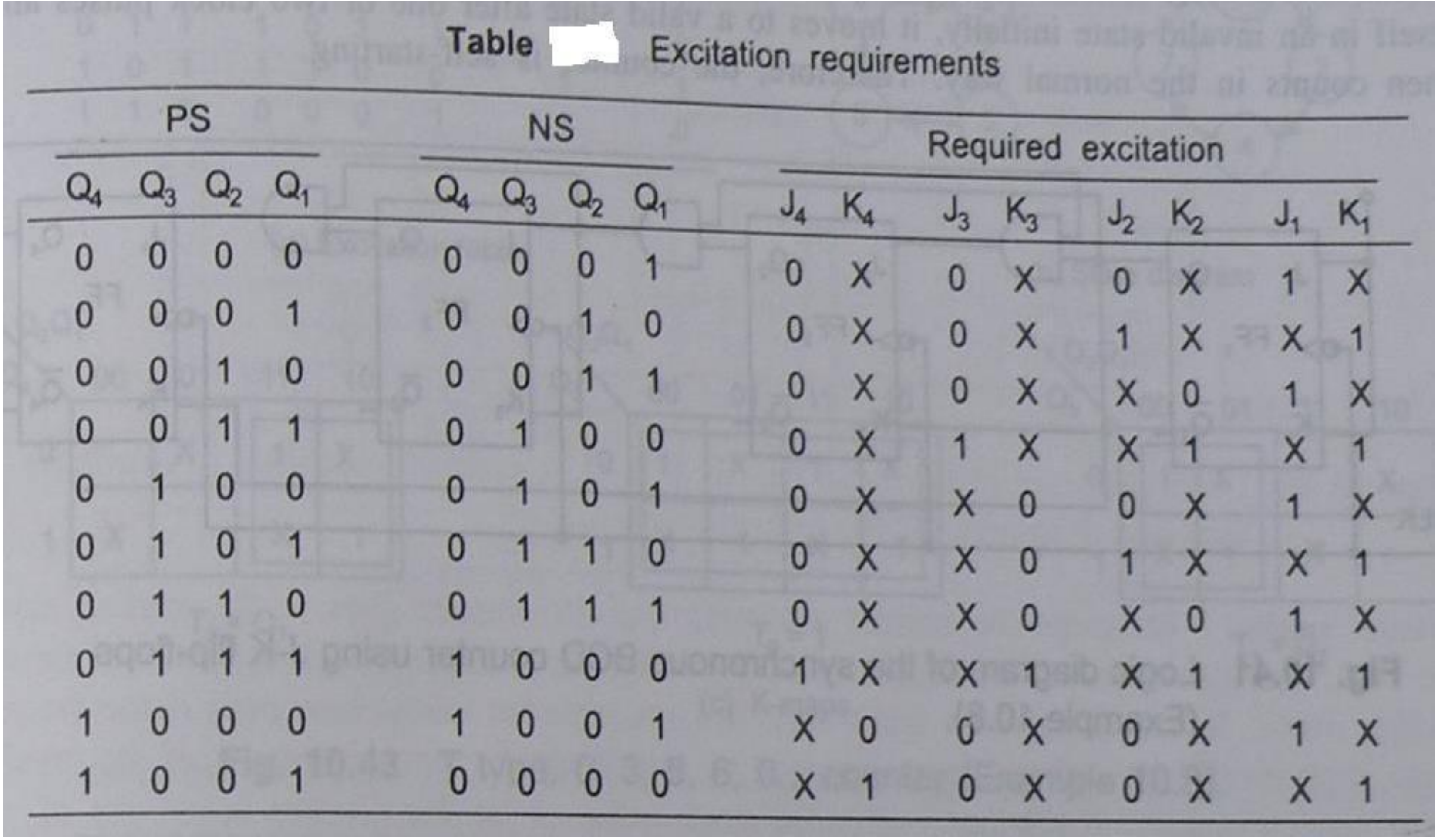
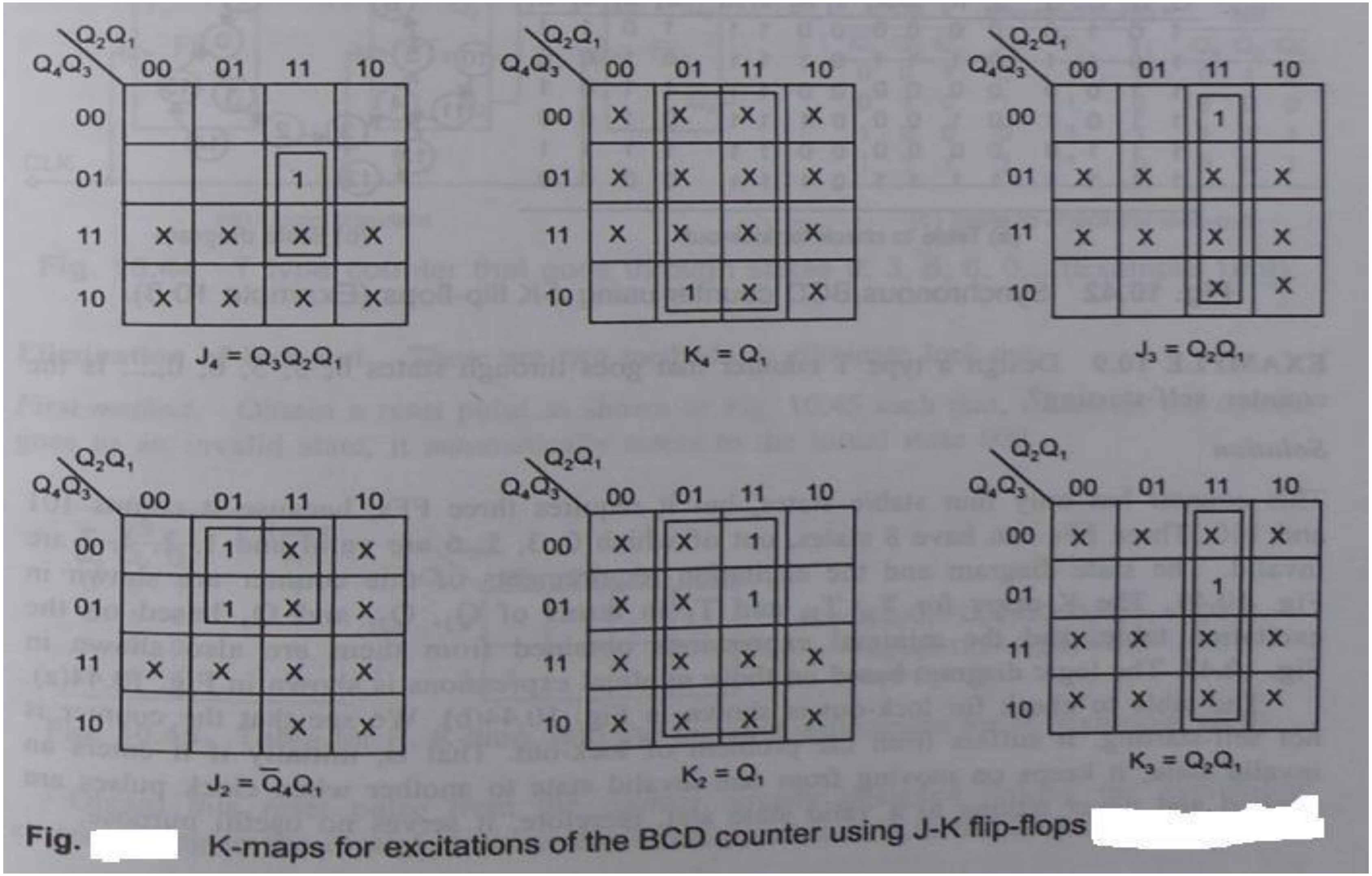


Table [redacted] **Excitation requirements**

PS				NS				Required excitation							
Q_4	Q_3	Q_2	Q_1	Q_4	Q_3	Q_2	Q_1	J_4	K_4	J_3	K_3	J_2	K_2	J_1	K_1
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	1	X
1	0	0	1	0	0	0	0	X	1	0	X	0	X	X	1



Q_4Q_3 \ Q_2Q_1	00	01	11	10
00				
01			1	
11	X	X	X	X
10	X	X	X	X

$J_4 = Q_3Q_2Q_1$

Q_4Q_3 \ Q_2Q_1	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	X	X	X	X
10		1	X	X

$K_4 = Q_1$

Q_4Q_3 \ Q_2Q_1	00	01	11	10
00			1	
01	X	X	X	X
11	X	X	X	X
10			X	X

$J_3 = Q_2Q_1$

Q_4Q_3 \ Q_2Q_1	00	01	11	10
00		1	X	X
01		1	X	X
11	X	X	X	X
10			X	X

$J_2 = \overline{Q}_4Q_1$

Q_4Q_3 \ Q_2Q_1	00	01	11	10
00	X	X	1	
01	X	X	1	
11	X	X	X	X
10	X	X	X	X

$K_2 = Q_1$

Q_4Q_3 \ Q_2Q_1	00	01	11	10
00	X	X	X	X
01			1	
11	X	X	X	X
10	X	X	X	X

$K_3 = Q_2Q_1$

Fig. K-maps for excitations of the BCD counter using J-K flip-flops

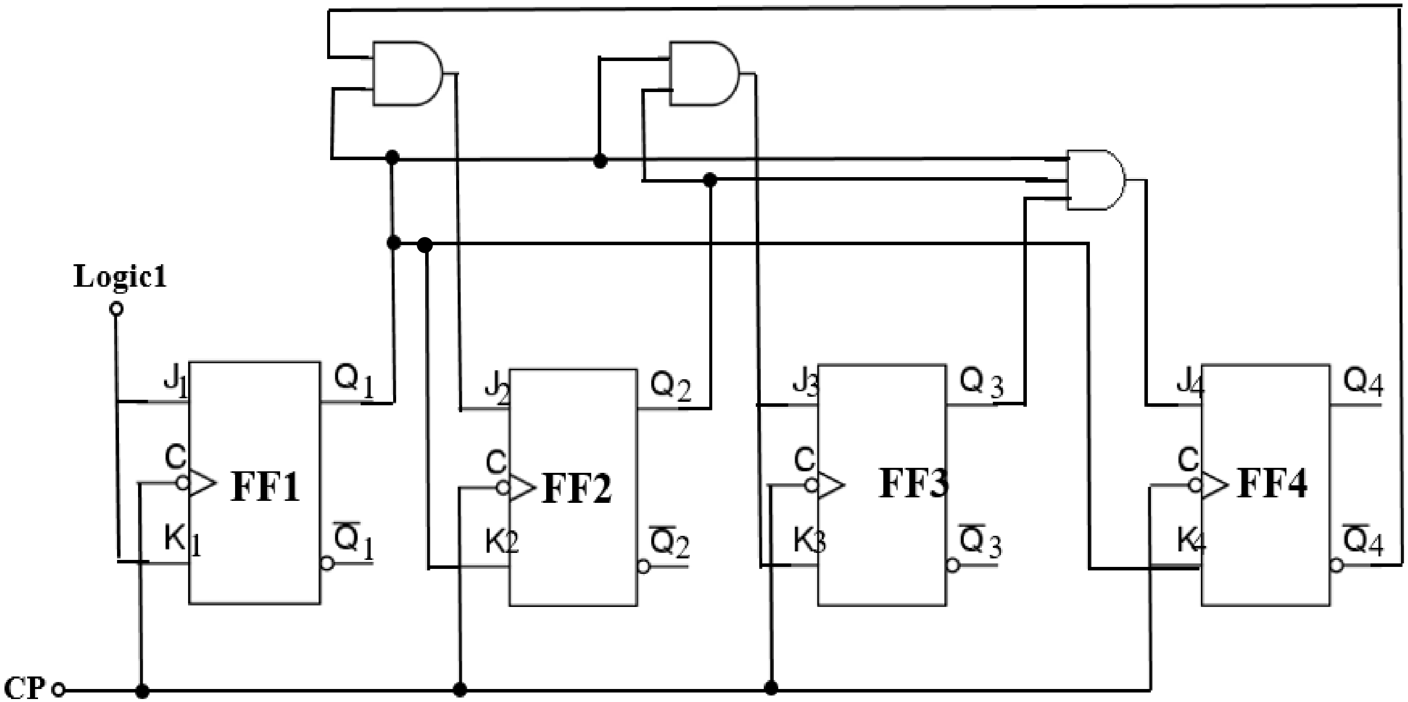
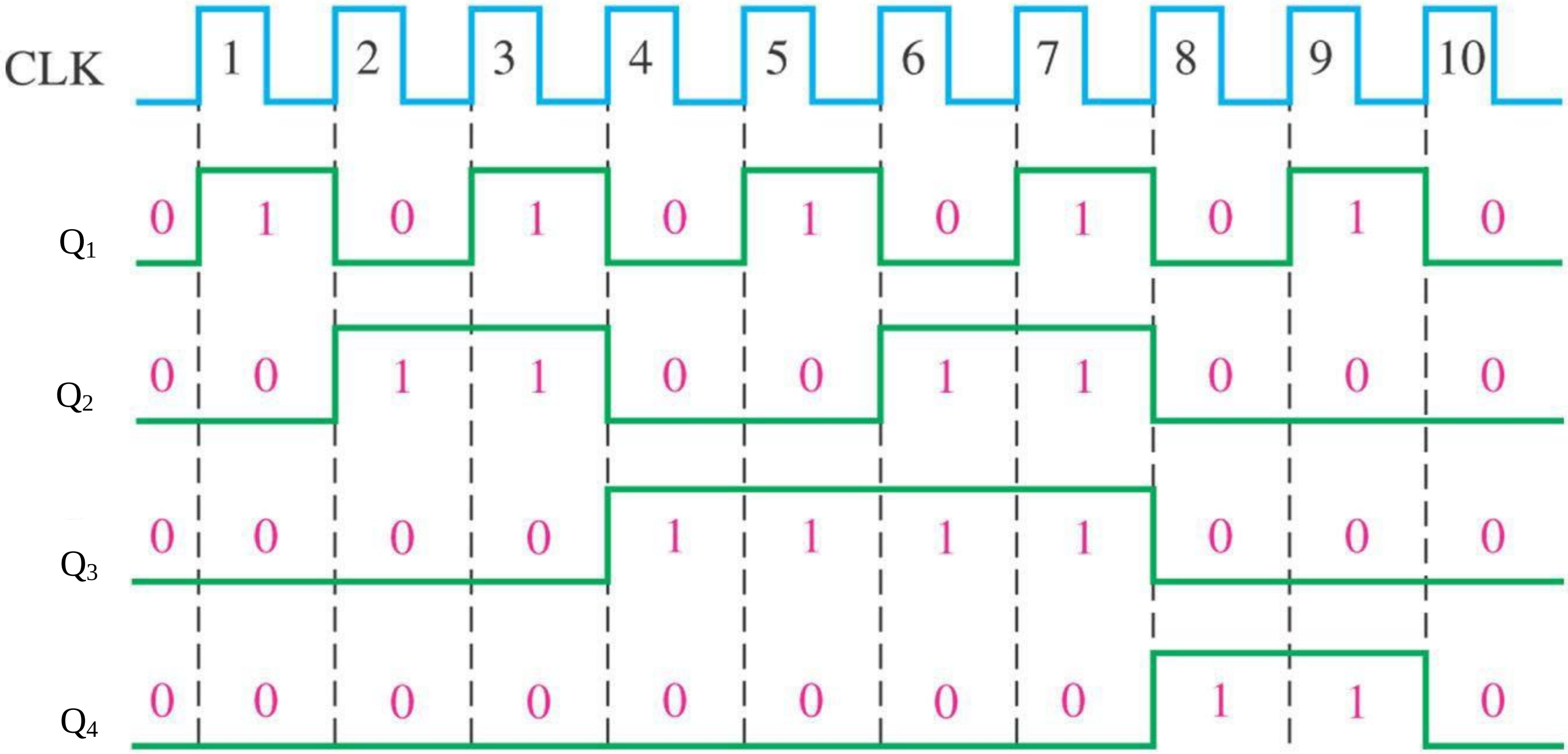
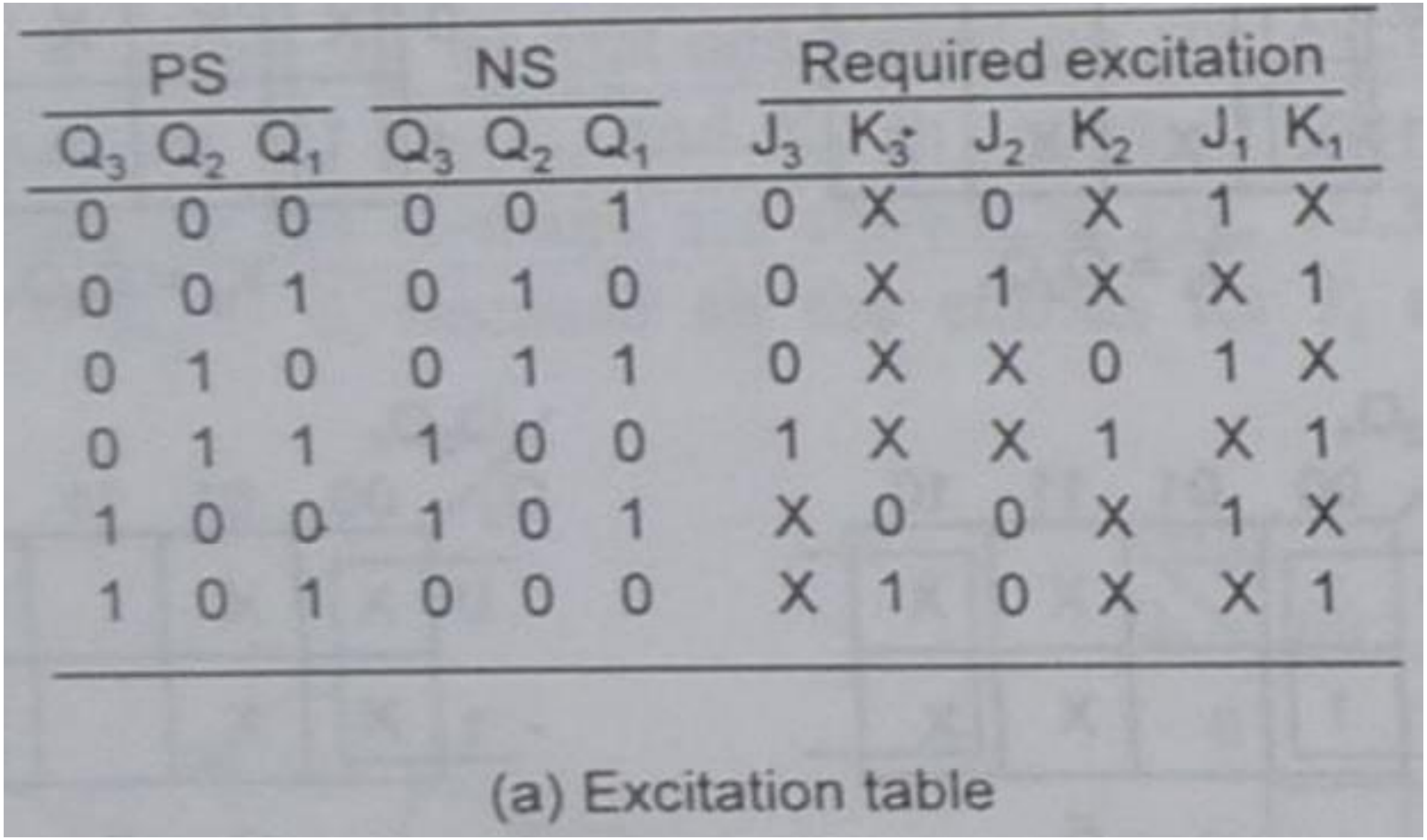


Fig: Logic Diagram



Timing diagram for the BCD decade counter (Q_0 is the LSB).

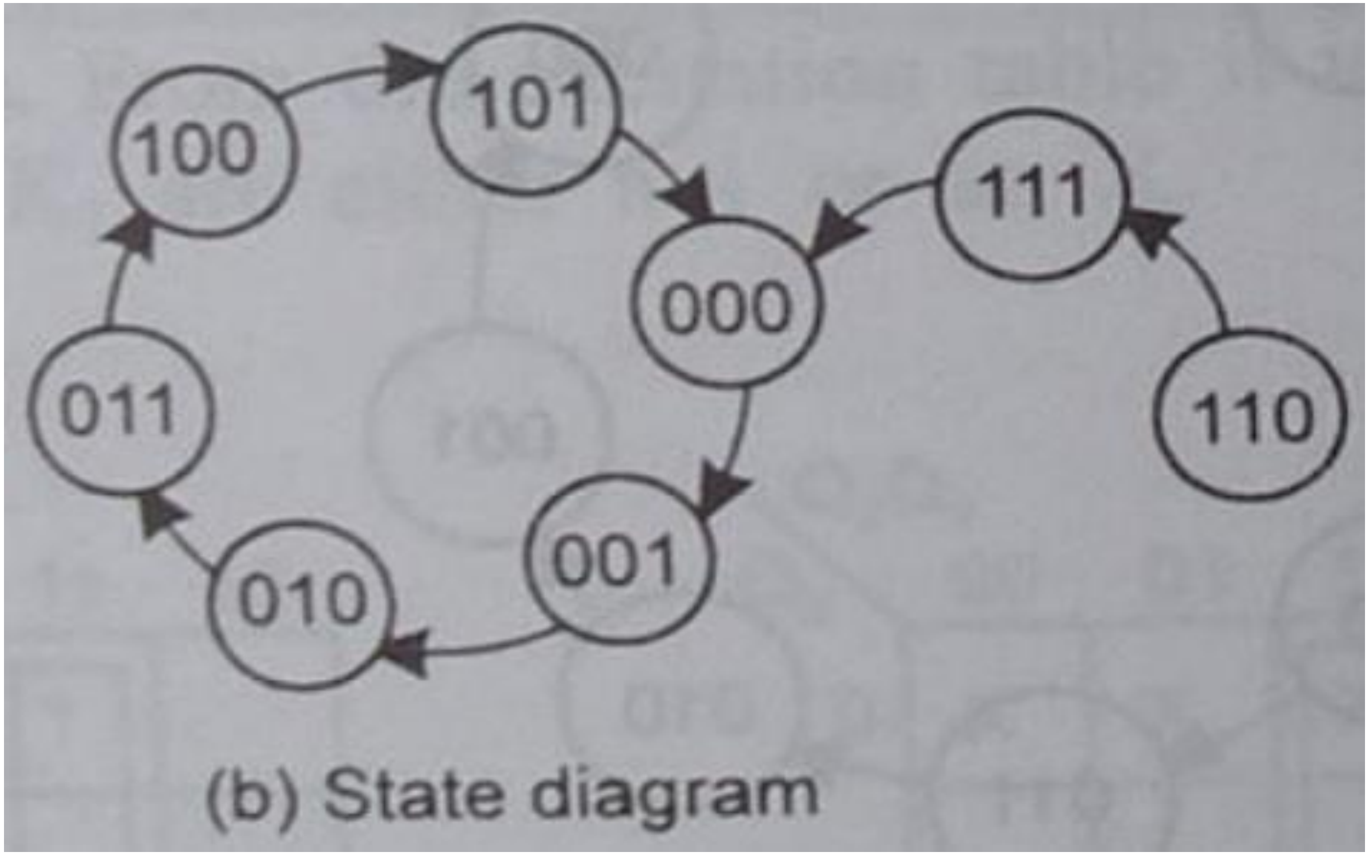
Mod 6 Counter



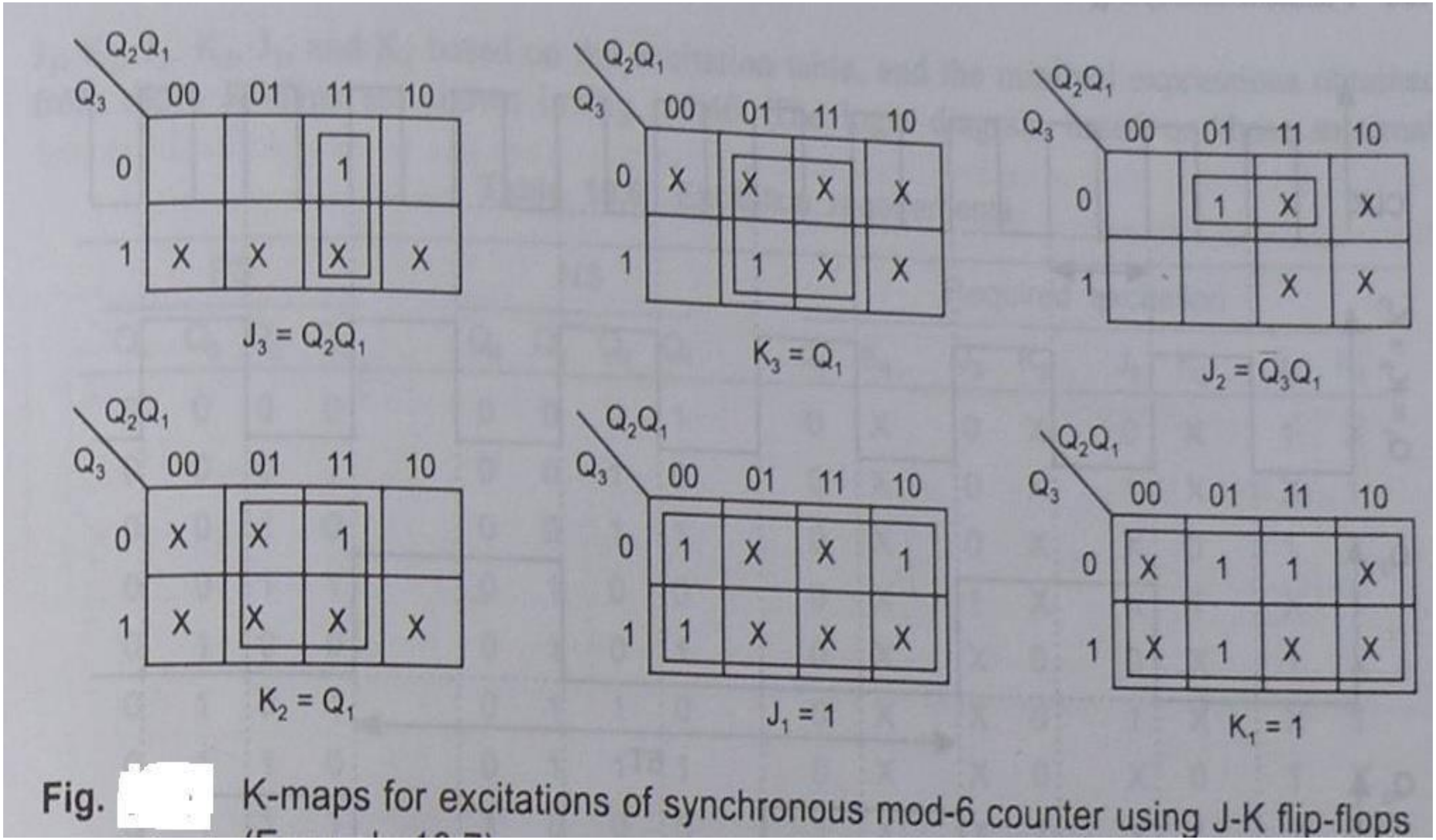
PS			NS			Required excitation					
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	J_3	K_3	J_2	K_2	J_1	K_1
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	0	0	0	X	1	0	X	X	1

(a) Excitation table

Mod 6 Counter



(b) State diagram



Q_3 \ Q_2Q_1	00	01	11	10
0			1	
1	X	X	X	X

$J_3 = Q_2Q_1$

Q_3 \ Q_2Q_1	00	01	11	10
0	X	X	X	X
1		1	X	X

$K_3 = Q_1$

Q_3 \ Q_2Q_1	00	01	11	10
0		1	X	X
1			X	X

$J_2 = \bar{Q}_3Q_1$

Q_3 \ Q_2Q_1	00	01	11	10
0	X	X	1	
1	X	X	X	X

$K_2 = Q_1$

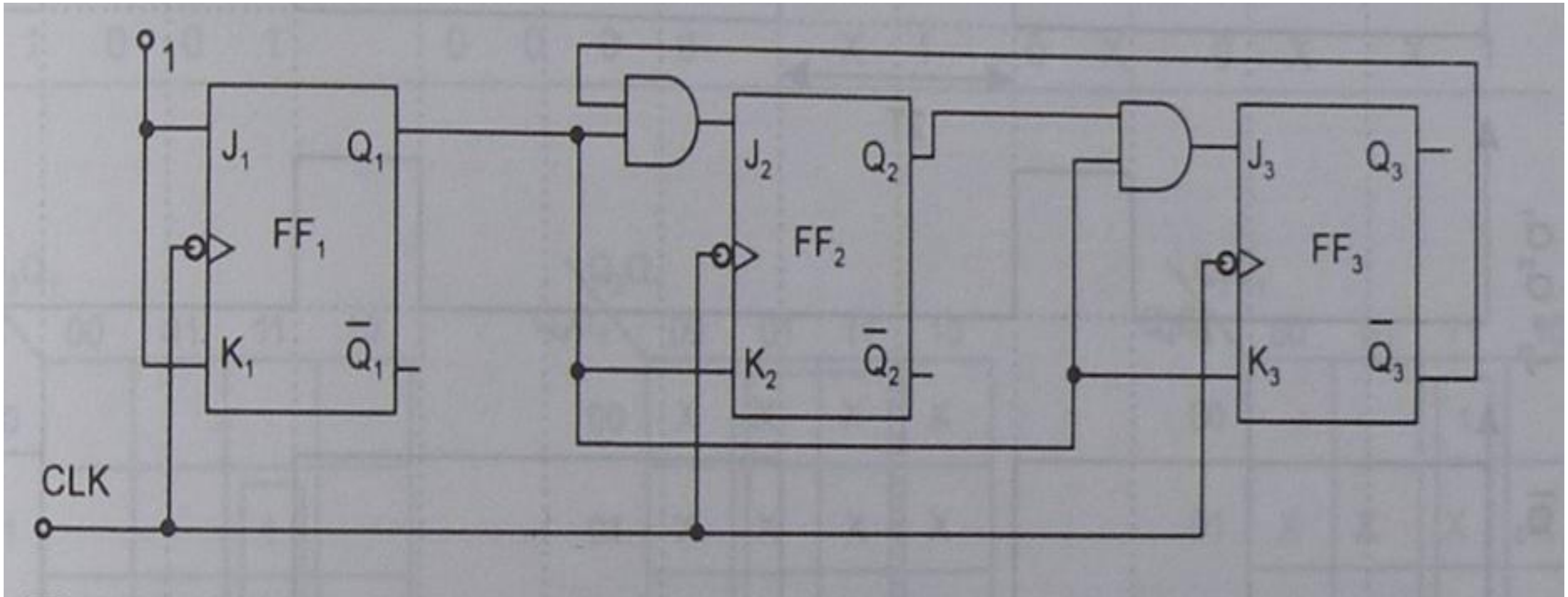
Q_3 \ Q_2Q_1	00	01	11	10
0	1	X	X	1
1	1	X	X	X

$J_1 = 1$

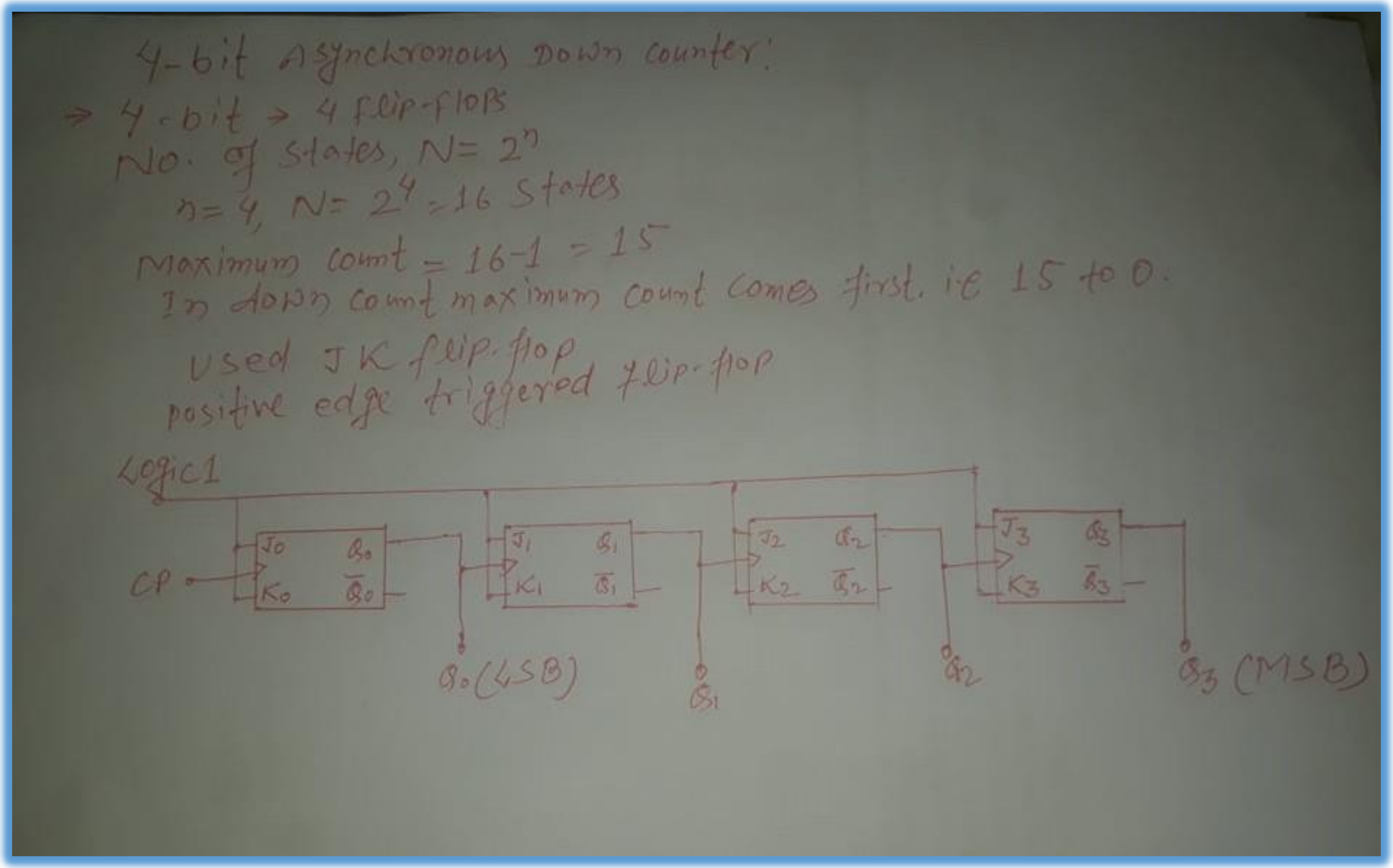
Q_3 \ Q_2Q_1	00	01	11	10
0	X	1	1	X
1	X	1	X	X

$K_1 = 1$

Fig. K-maps for excitations of synchronous mod-6 counter using J-K flip-flops



Logic diagram of the synchronous mod-6 counter using J-K flip-flops



4-bit Asynchronous Down counter:

→ 4-bit → 4 flip-flops

No. of states, $N = 2^n$

$n = 4$, $N = 2^4 = 16$ states

Maximum count = 16-1 = 15

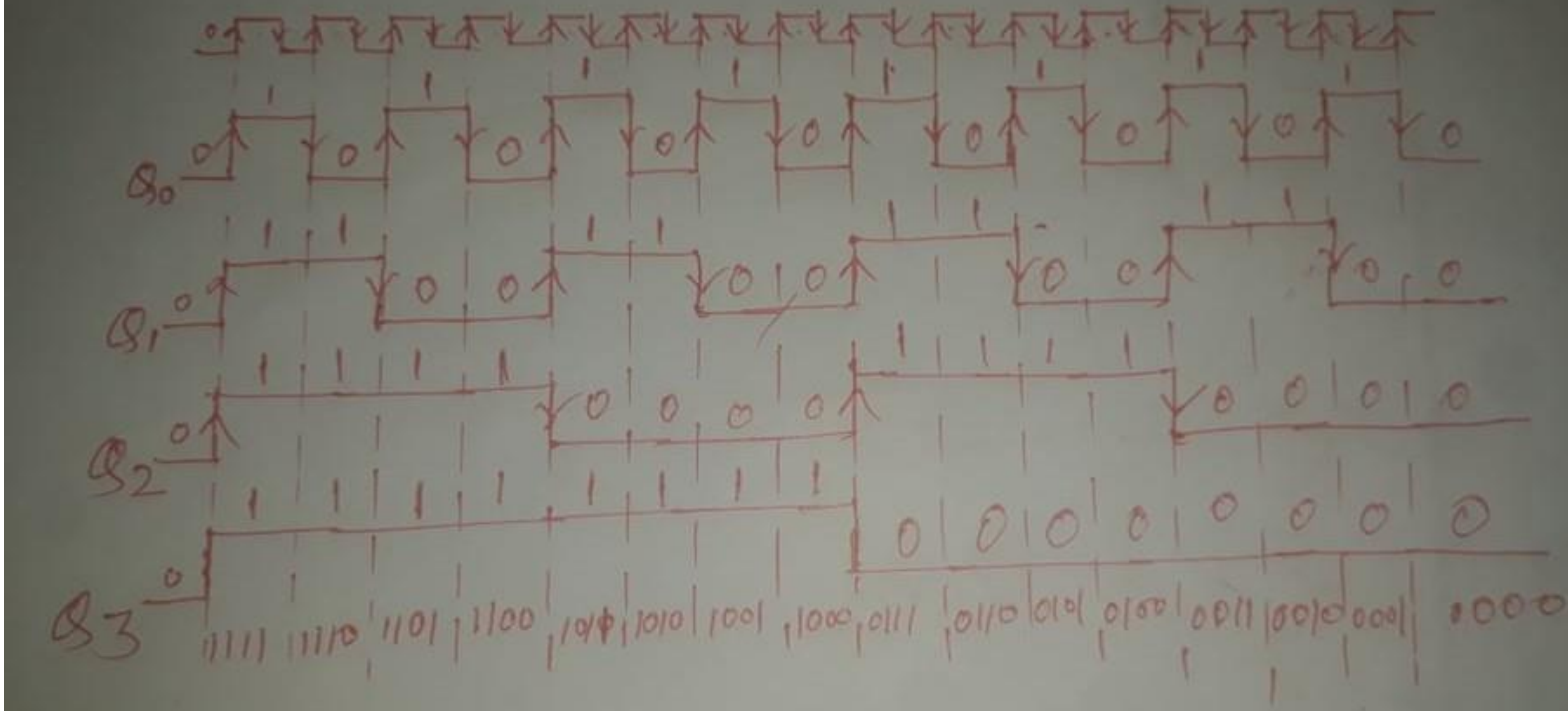
In down count maximum count comes first. i.e 15 to 0.

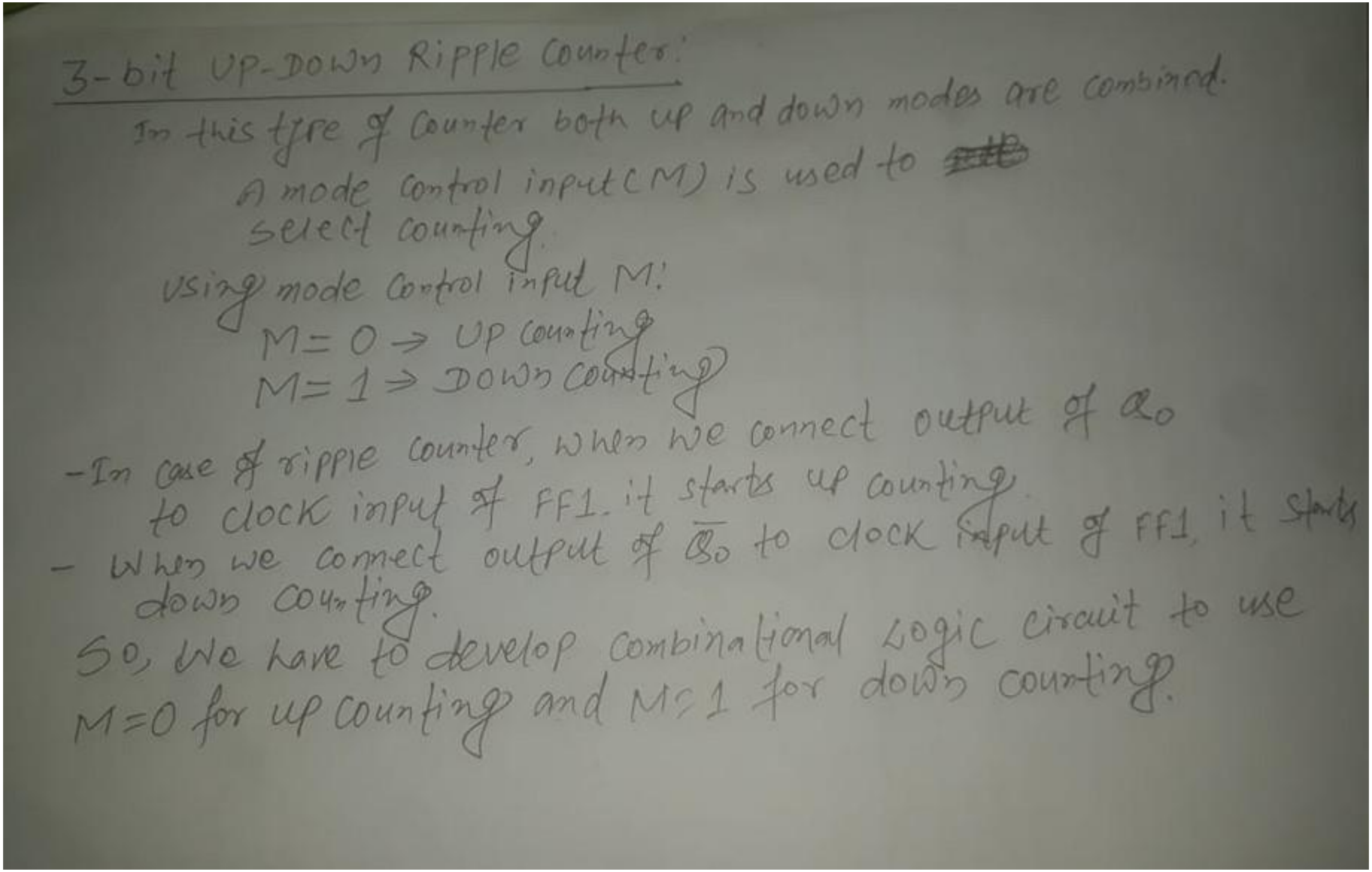
used JK flip-flop

positive edge triggered flip-flop

Logic1
CP
J_0
K_0
Q_0
$\overline{Q_0}$
J_1
K_1
Q_1
$\overline{Q_1}$
J_2
K_2
Q_2
$\overline{Q_2}$
J_3
K_3
Q_3
$\overline{Q_3}$
Q_0 (LSB)
Q_1
Q_2
Q_3 (MSB)

Timing Diagram:





3-bit UP-DOWN Ripple Counter:

In this type of Counter both up and down modes are combined.

A mode control input (M) is used to select counting.

using mode Control input M:

$M = 0 \Rightarrow$ Up counting

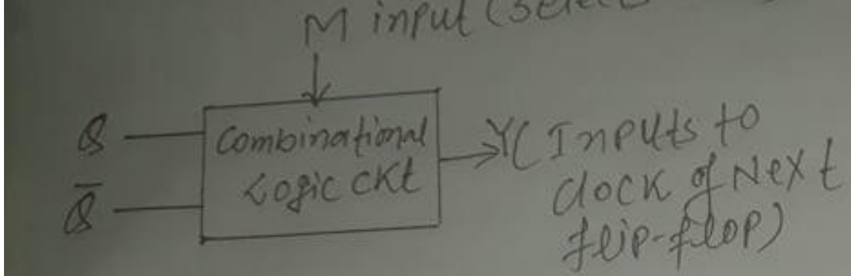
$M = 1 \Rightarrow$ DOWN counting

- In case of ripple counter, when we connect output of Q_0 to clock input of FF1. it starts up counting.
- When we connect output of $\overline{Q_0}$ to clock input of FF1, it starts down counting.

So, We have to develop combinational Logic circuit to use $M=0$ for up counting and $M=1$ for down counting.

3-bit up-down Ripple counter

Block diagram



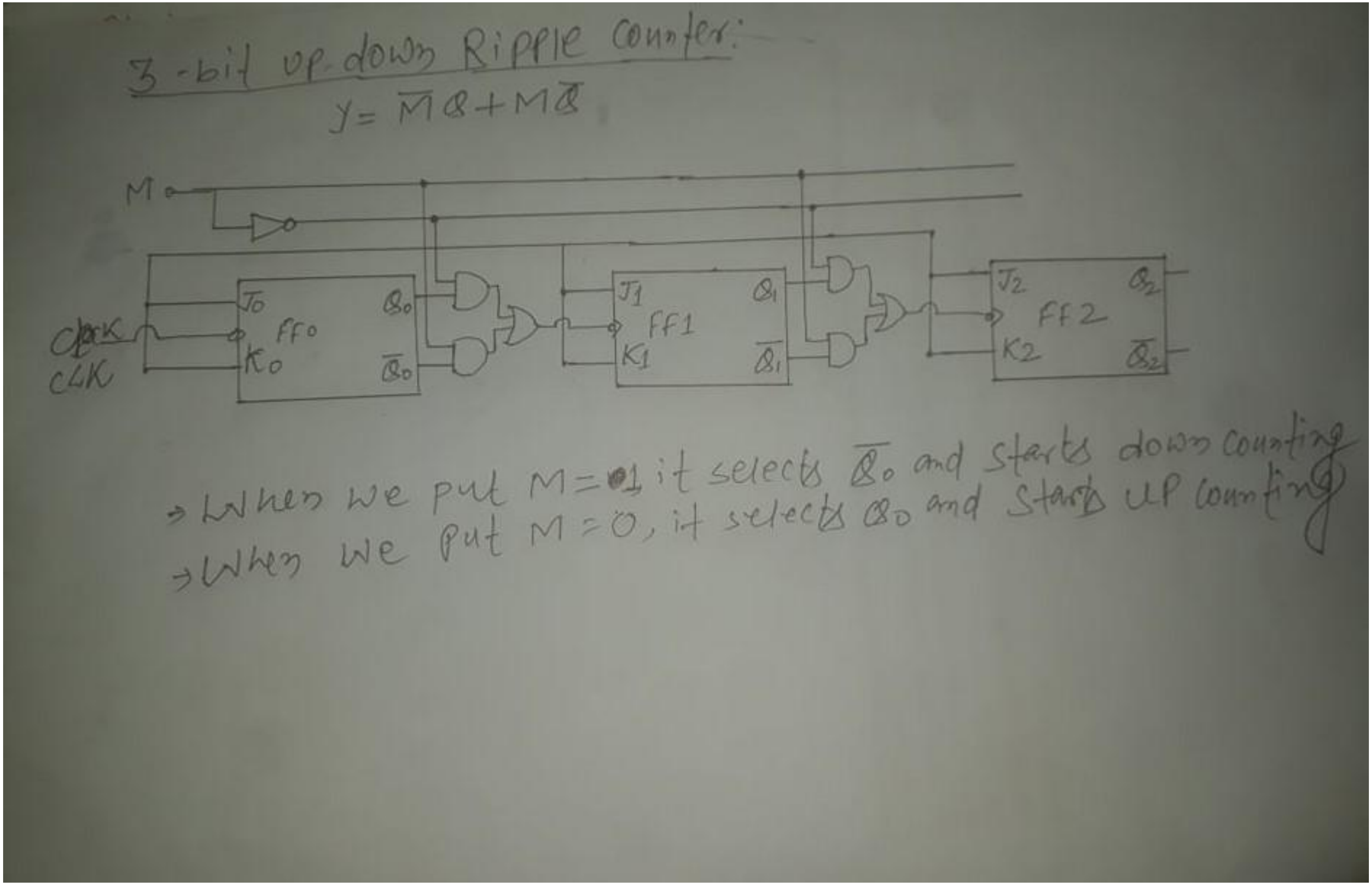
K-map

M \ $Q\bar{Q}$	00	01	10	11
0	0	0	1	1
1	0	1	1	0

$$\therefore Y = \bar{M}Q + M\bar{Q}$$

Truth table for above ckt

	Inputs			output	
	M	Q	$\bar{Q}$	Y	
UP counting	0	0	0	0	UP counting depends on Q.
	0	0	1	0	
	0	1	0	1	
	0	1	1	1	
Down counting	1	0	0	0	Down counting depends on $\bar{Q}$.
	1	0	1	1	
	1	1	0	0	
	1	1	1	1	



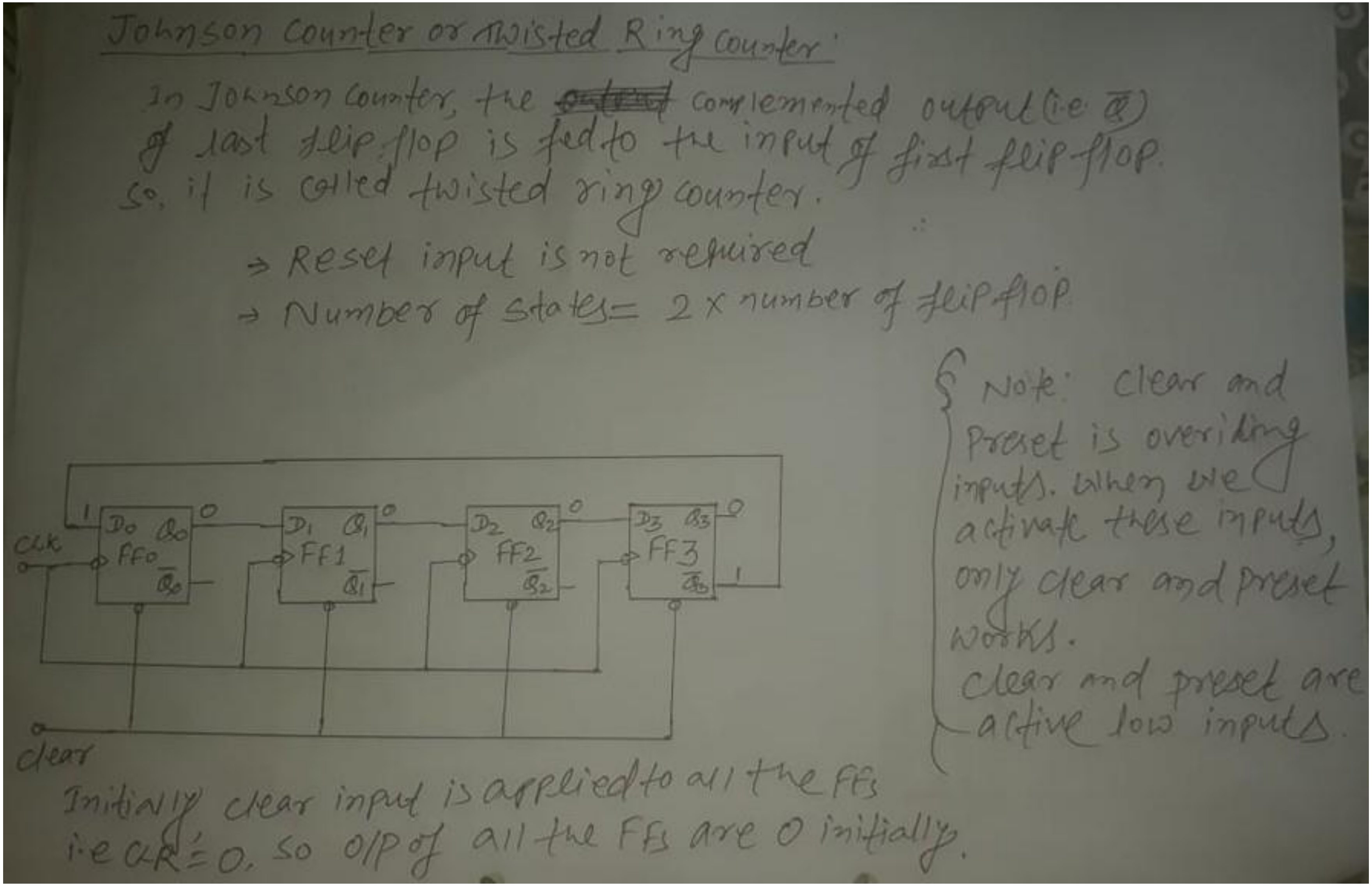
3-bit up-down Ripple counter:

$y = \overline{M}Q + M\overline{Q}$

→ When we put M = 1 it selects $\overline{Q}_0$ and starts down counting

→ When we put M = 0, it selects Q_0 and starts up counting

→ When we



Johnson Counter or Twisted Ring Counter:

In Johnson counter, the ~~output~~ complemented output (i.e. $\overline{Q}$) of last flip flop is fed to the input of first flip flop. So, it is called twisted ring counter.

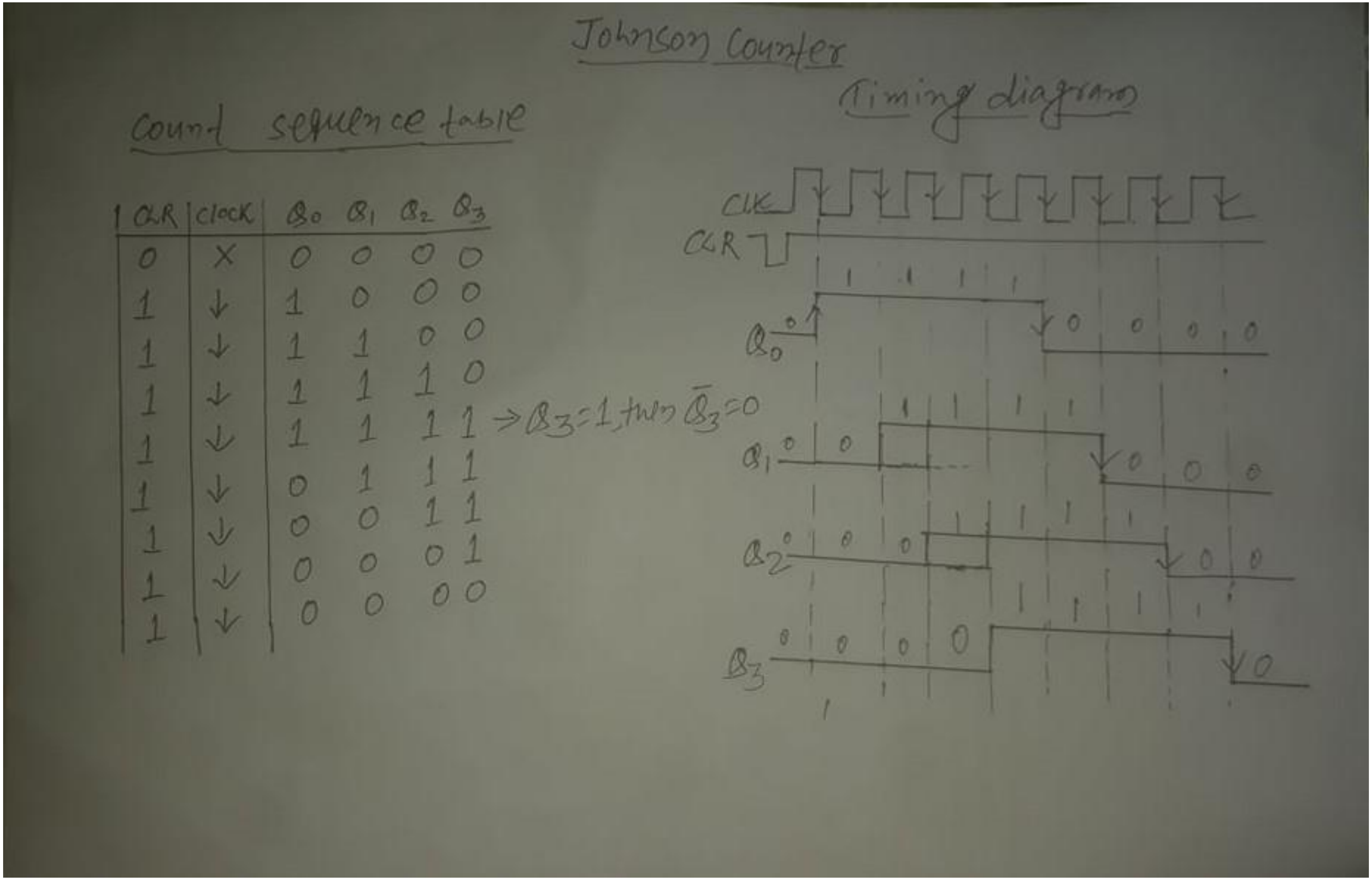
∴

→ Reset input is not required

→ Number of states = 2 × number of flip flop

Note: Clear and Preset is overiding inputs. When we activate these inputs, only clear and preset works.
clear and preset are active low inputs.

Initially clear input is applied to all the FFs i.e $\overline{CLR} = 0$. So O/P of all the FFs are 0 initially.



Johnson Counter

Count sequence table

CLR	Clock	Q_0	Q_1	Q_2	Q_3
0	X	0	0	0	0
1	↓	1	0	0	0
1	↓	1	1	0	0
1	↓	1	1	1	0
1	↓	1	1	1	1
1	↓	0	1	1	1
1	↓	0	0	1	1
1	↓	0	0	0	1
1	↓	0	0	0	0

→ $Q_3=1$, then $\bar{Q}_3=0$

Timing diagram

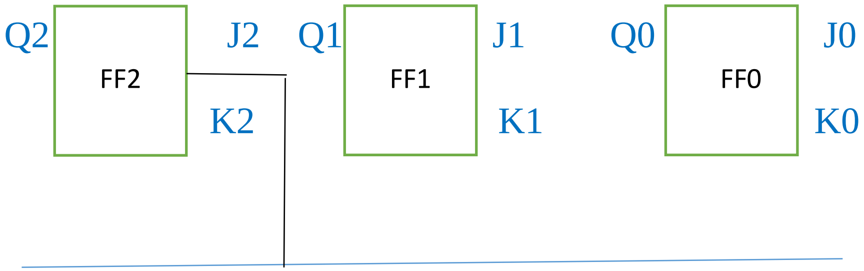
3-bit synchronous down counter

$2^3 = 8 = 7$ to 0

We use three negative edge triggered FFs.

Q2	Q1	Q0	J2	K2	J1	K1	J0	K0
1	1	1	x	0	x	0	x	1
1	1	0	x	0	x	1	1	x
1	0	1	x	0	0	x	x	1
1	0	0	x	1	1	x	1	x
0	1	1	0	x	x	0	x	1
0	1	0	x	x	x	1	1	x
0	0	1	0	x	0	x	x	1
0	0	0	1	x	1	x	1	x

K-map for J2, K2, J1,K1, J0,K0



3-bit synchronous Up-down counter:

Number of flip-flops used= 3

Total number of Counts = 2^3 =8

For up-counting 0-7 and for down counting 7-0

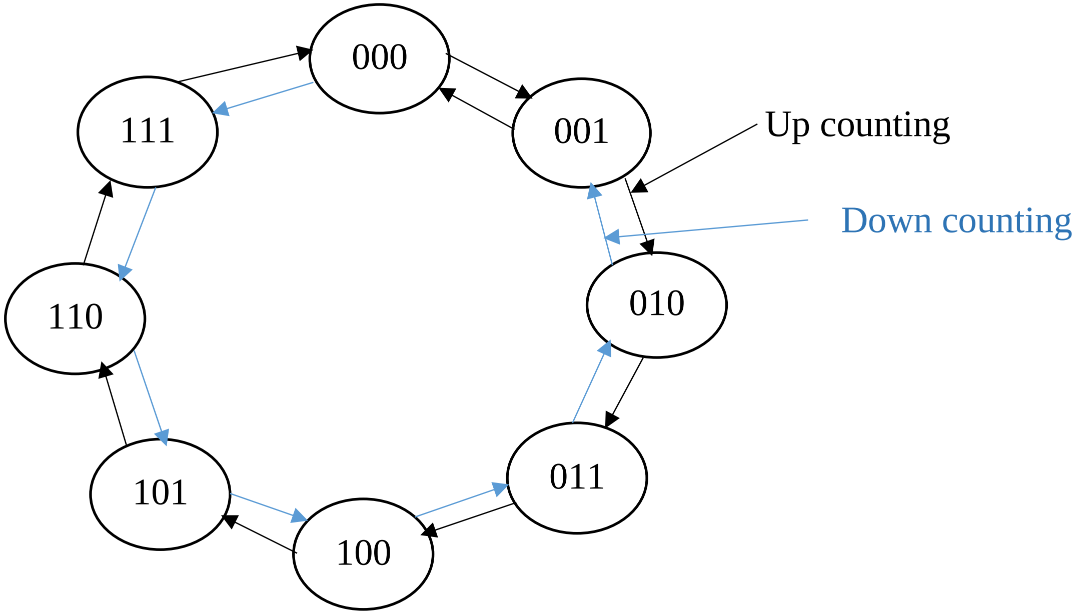
Clock pulse is simultaneously given to all the flip-flops.

A mode control input (M) is used to select counting.

M = 0 up counting

M= 1 down counting

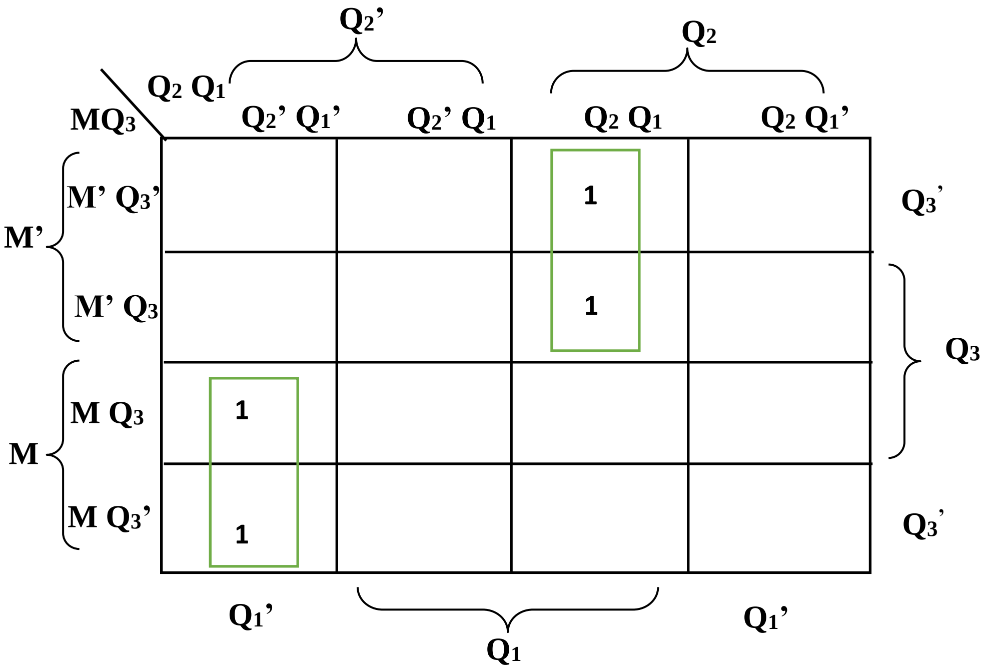
State diagram:



Truth table 3-bit Up down Counter:

Control Input	Present State			Next State			FFs Inputs		
M	Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	T_3	T_2	T_1
0	0	0	0	0	0	1	0	0	1
0	0	0	1	0	1	0	0	1	1
0	0	1	0	0	1	1	0	0	1
0	0	1	1	1	0	0	1	1	1
0	1	0	0	1	0	1	0	0	1
0	1	0	1	1	1	0	0	1	1
0	1	1	0	1	1	1	0	0	1
0	1	1	1	0	0	0	1	1	1
1	0	0	0	1	1	1	1	1	1
1	0	0	1	0	0	0	0	0	1
1	0	1	0	0	0	1	0	1	1
1	0	1	1	0	1	0	0	0	1
1	1	0	0	0	1	1	1	1	1
1	1	0	1	1	0	0	0	0	1
1	1	1	0	1	0	1	0	1	1
1	1	1	1	1	1	0	0	0	1

K-Map for T_3



MQ3 \ Q2 Q1	Q2’ Q1’	Q2’ Q1	Q2 Q1	Q2 Q1’
M’ Q3’			1	
M’ Q3			1	
M Q3	1			
M Q3’	1			

T3 = MQ2’ Q1’ + M’ Q2 Q1

Introduction to Registers

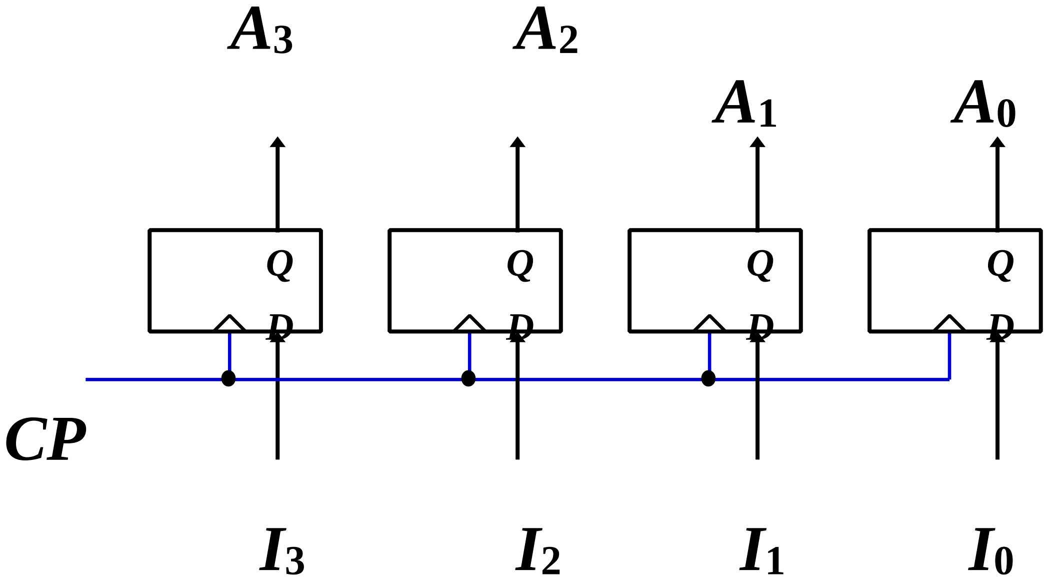
- An *n*-bit register has a group of *n* flip-flops and some logic gates and is capable of storing *n* bits of information.
  - The flip-flops store the information while the gates control when and how new information is transferred into the register.

Some functions of register:

- ❖retrieve data from register
- ❖store/load new data into register (serial or parallel)
- ❖shift the data within register (left or right)

Introduction to Registers

- No external gates.
- Example: A 4-bit register. A new 4-bit data is loaded every
- clock cycle.



Registers with Parallel Load

- Instead of loading the register at every clock pulse, we may want to control when to load.

- *Loading* a register: transfer new information into the register. Requires a *load* control input.

- *Parallel loading*: all bits are loaded simultaneously.

Shift Registers

- Another function of a register, besides storage, is to provide for *data movements*.

- Each *stage* (flip-flop) in a shift register represents one bit of storage, and the shifting capability of a register permits the movement of data from stage to stage within the register, or into or out of the register upon application of clock pulses.

- A register capable of shifting its binary information either to the right or to the left is called a shift register.

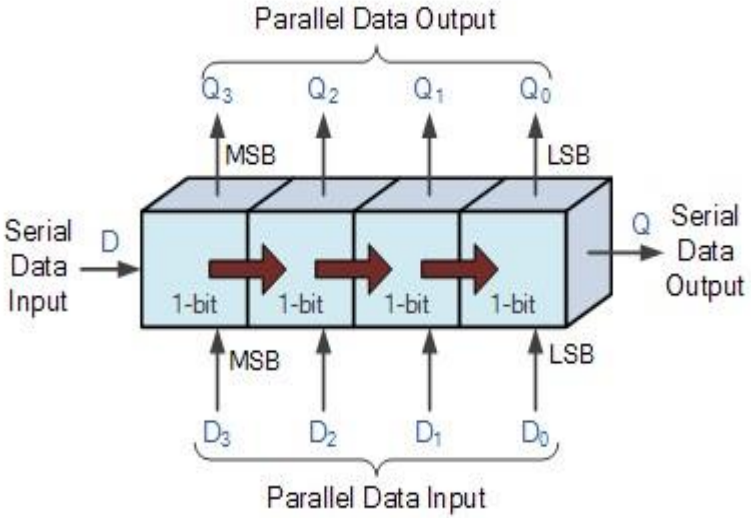
- The logical configuration of a shift register consists of a chain of flip-flops connected in cascade, with the output of one flip-flop connected to the input of the next flip-flop.
- All flip-flops receive a common clock pulse that causes the shift from one stage to the next.

Shift Registers

- Generally, shift registers operate in one of four different modes with the basic movement of data through a shift register being:
- Serial-in to Parallel-out (SIPO) - the register is loaded with serial data, one bit at a time, with the stored data being available in parallel form.
- Serial-in to Serial-out (SISO) - the data is shifted serially "IN" and "OUT" of the register, one bit at a time in either a left or right direction under clock control.

- Parallel-in to Serial-out (PISO) - the parallel data is loaded into the register simultaneously and is shifted out of the register serially one bit at a time under clock control.
- Parallel-in to parallel-out (PIPO)

  – the parallel data is loaded simultaneously into the register, and transferred together to their respective outputs by the same clock pulse.



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

- Accepts data serially.
- Outputs of all stages are available simultaneously.

Block Diagram

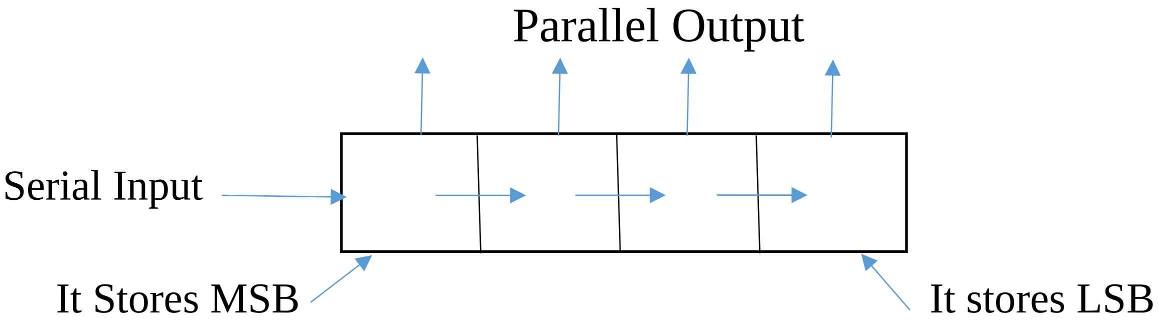


Fig: Block diagram of 4-bit SIPO Shift register

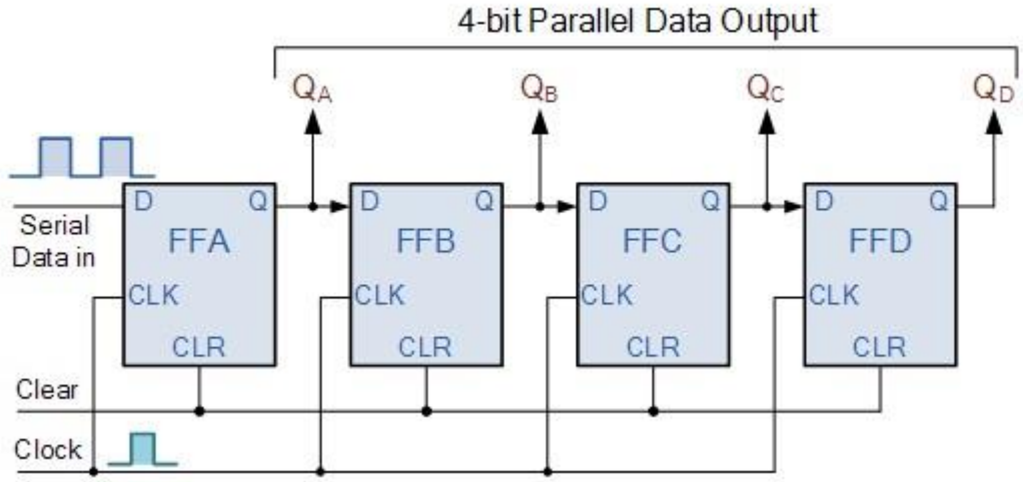
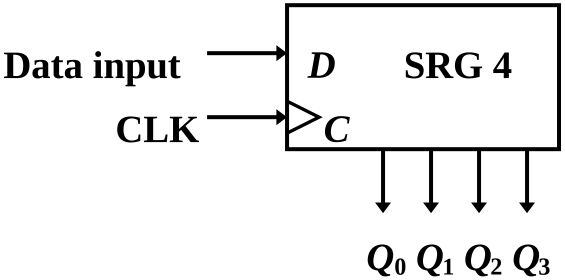


Fig. 4 bit serial in parallel out shift register



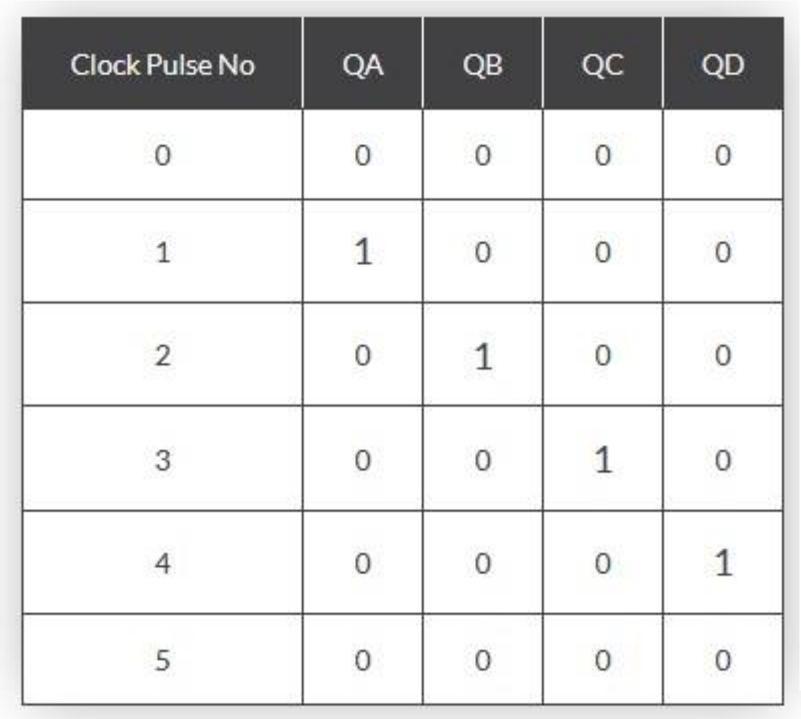
Logic symbol

- Let us assume that all the flip-flops (FFA to FFD) have just been RESET (CLEAR input) and that all the outputs Q_A to Q_D are at logic level “0” i.e., no parallel data output.
- If a logic “1” is connected to the DATA input pin of FFA then on the first clock pulse the output of FFA and therefore the resulting Q_A will be set

HIGH to logic “1” with all the other outputs still remaining LOW at logic “0”.

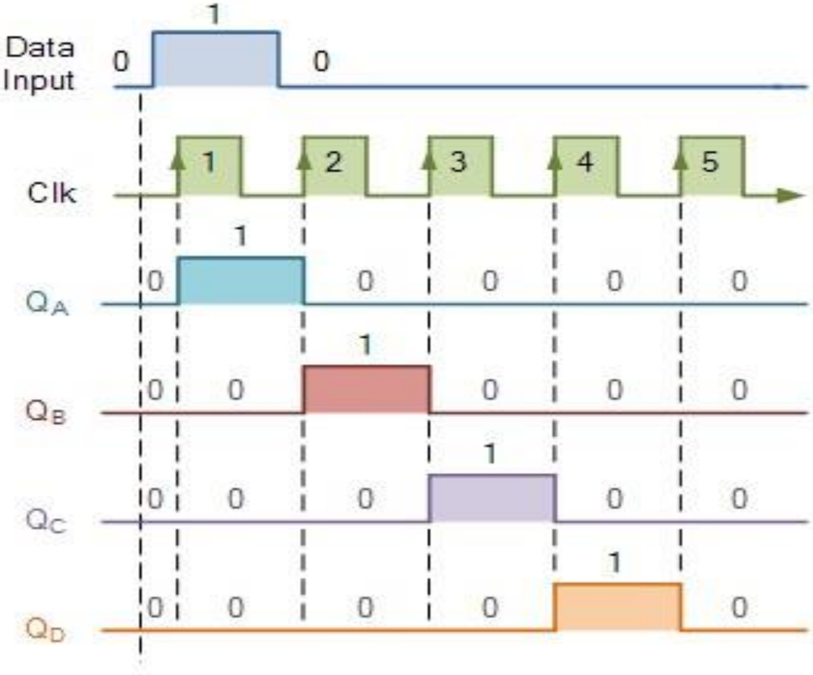
- Assume now that the DATA input pin of FFA has returned LOW again to logic “0” giving us one data pulse or 0-1-0.
- The second clock pulse will change the output of FFA to logic “0” and the output of FFB and Q_B HIGH to logic “1” as its input D has the logic “1” level on it from Q_A.
- When the third clock pulse arrives this logic “1” value moves to the output of FFC (Q_C) and so on until the arrival of the fifth clock pulse which sets all the outputs Q_A to Q_D back again to logic level “0” because the input to FFA has remained constant at logic level “0”.

- The effect of each clock pulse is to shift the data contents of each stage one place to the right, and this is shown in the following table until the data value of 0-0-0-1 is



Clock Pulse No	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	0	0	0	0

Shifting data 10000 serially



Timing diagram

Serial In/Serial Out Shift Registers

- Accepts data serially – one bit at a time – and also produces output serially.

Shifting data 1011

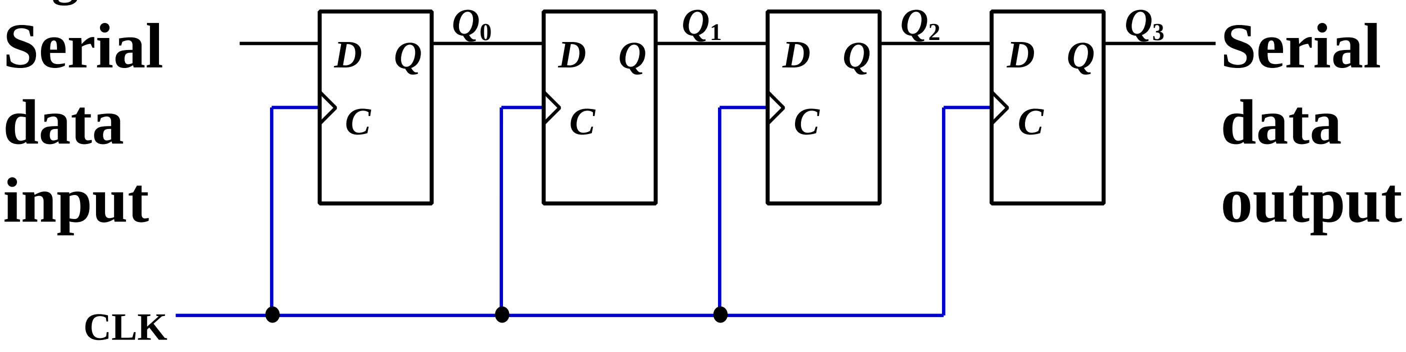
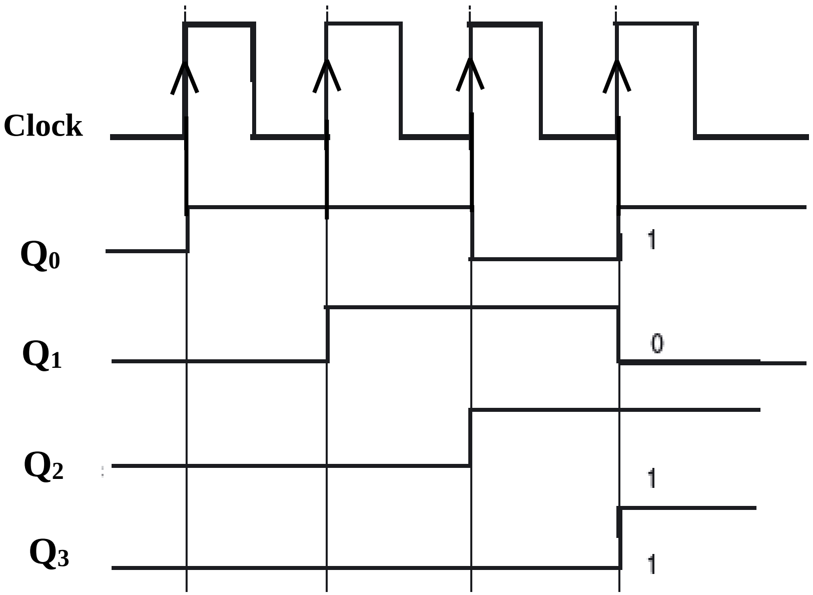


Fig: Serial in serial out shift register logic diagram

- This **shift register** is very similar to the SIPO except the data was read directly in a parallel form from the outputs Q_A to Q_D in SIPO, but the data is allowed to flow straight through the register and data out in the other end.
- Since there is only one output, the DATA leaves the shift register one bit at a time in a serial pattern, hence the name **Serial-in to Serial-Out Shift Register** or **SISO**.

Clock Pulse Number	Q_0	Q_1	Q_2	Q_3
Initially	0	0	0	0
↑	1	0	0	0
↑	1	1	0	0
↑	0	1	1	0
↑	1	0	1	1

Timing diagram of SISO Shift Register



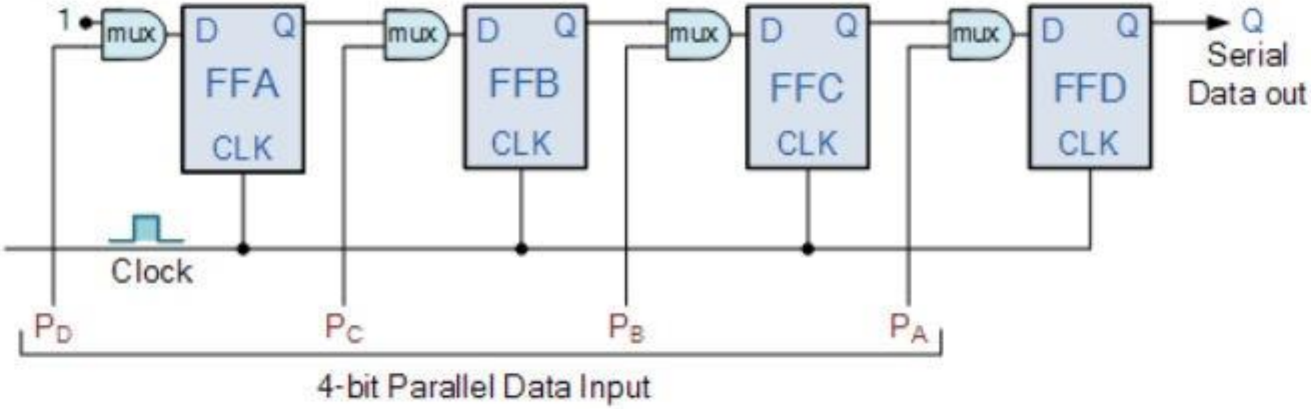
Parallel In/Serial Out Shift Registers

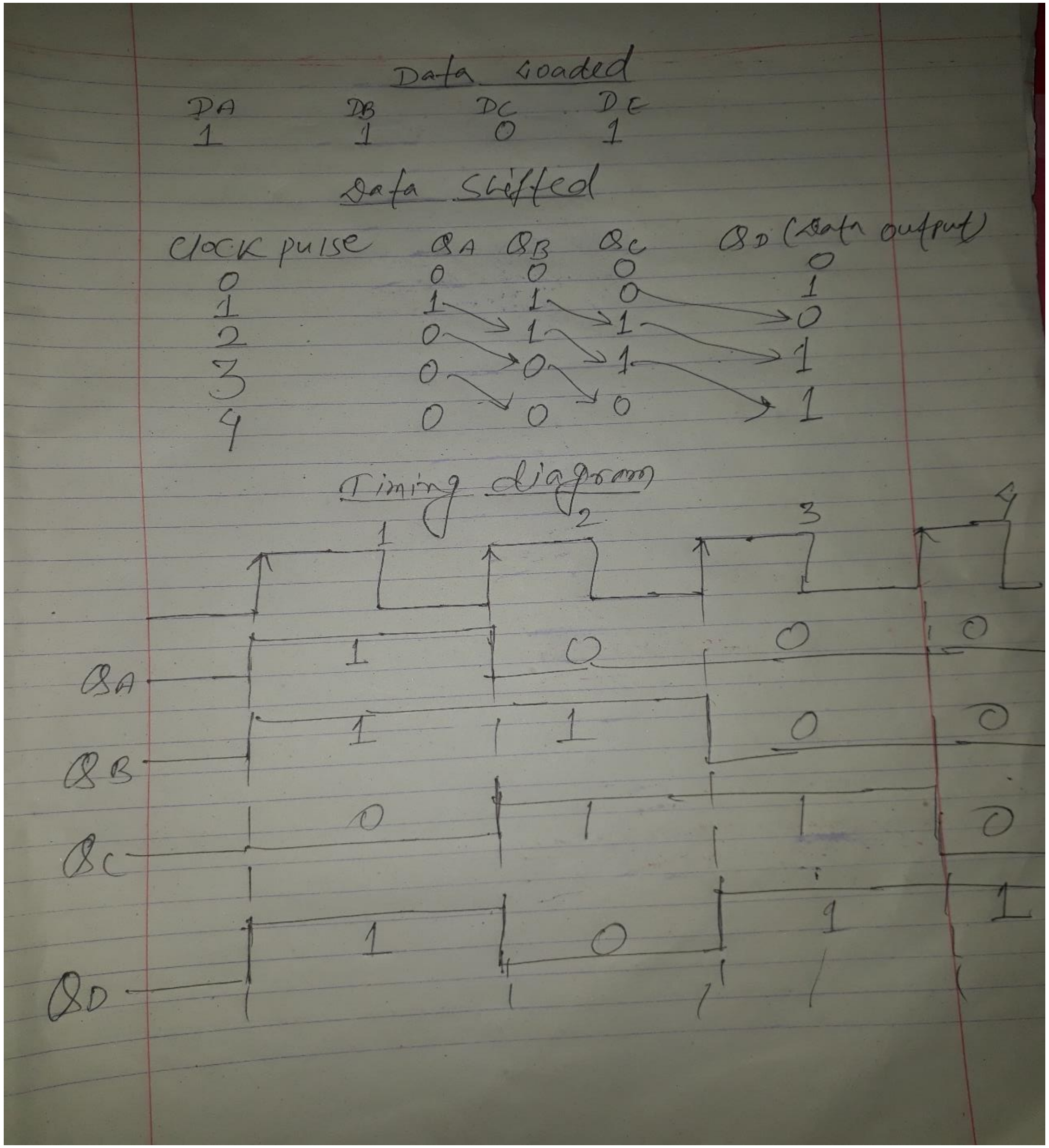
- Bits are entered simultaneously, but output is serial.

- The data is loaded into the register in a parallel in which all the data bits enter their inputs simultaneously, to the parallel input pins P_A to P_D of the register.

- The data is read out sequentially in the normal shift-right mode from the register at Q representing the data present at P_A to P_D.

- It is important to note that in this type of shift register a clock pulse is not required to parallel load the register as it is already present, but four clock pulses are required to unload the data.


1
mux
D
Q
FFA
CLK
FFB
FFC
FFD
Q
Serial
Data out
Clock
P_D
P_C
P_B
P_A
4-bit Parallel Data Input



Data Loaded

DA	DB	DC	DE
1	1	0	1

Data Shifted

Clock pulse	QA	QB	QC	QD (Data output)
0	0	0	0	0
1	1	1	0	1
2	0	1	1	0
3	0	0	1	1
4	0	0	0	1

Timing diagram

1
2
3
4
QA
1
0
0
0
QB
1
1
0
0
QC
0
1
1
0
QD
1
0
1
1

Parallel In/Parallel Out Shift Registers

- Simultaneous input and output of all data bits.
- The data is presented in a parallel format to the parallel input pins P_A to P_D and then transferred together directly to their respective output pins Q_A to Q_D by the same clock pulse.

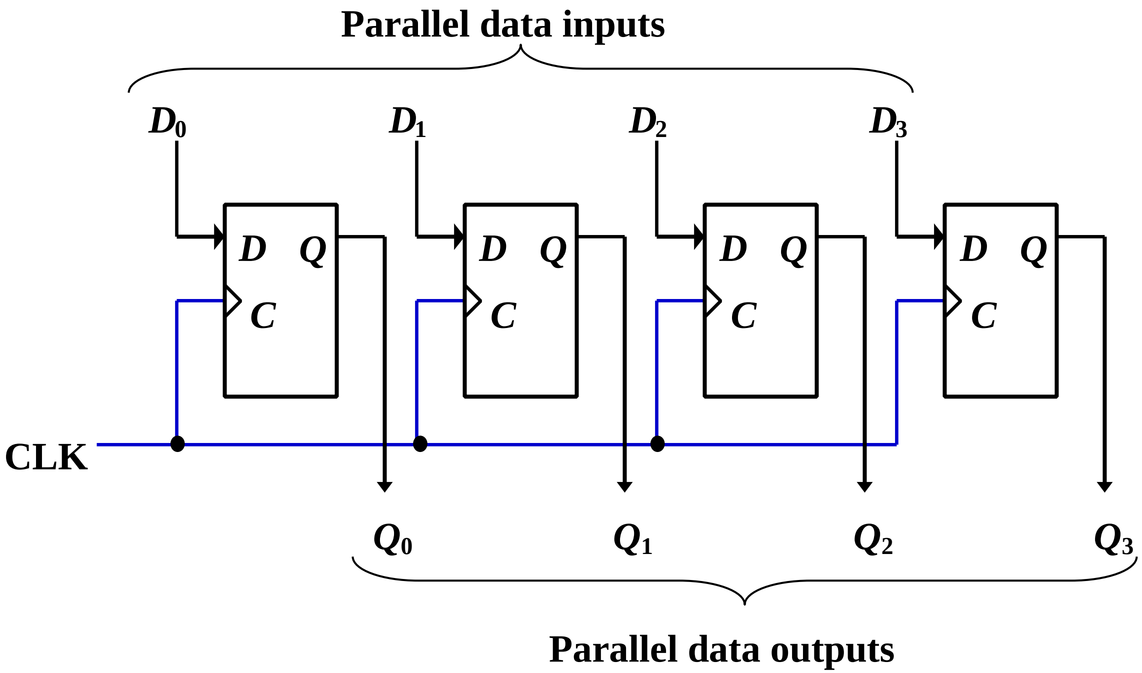
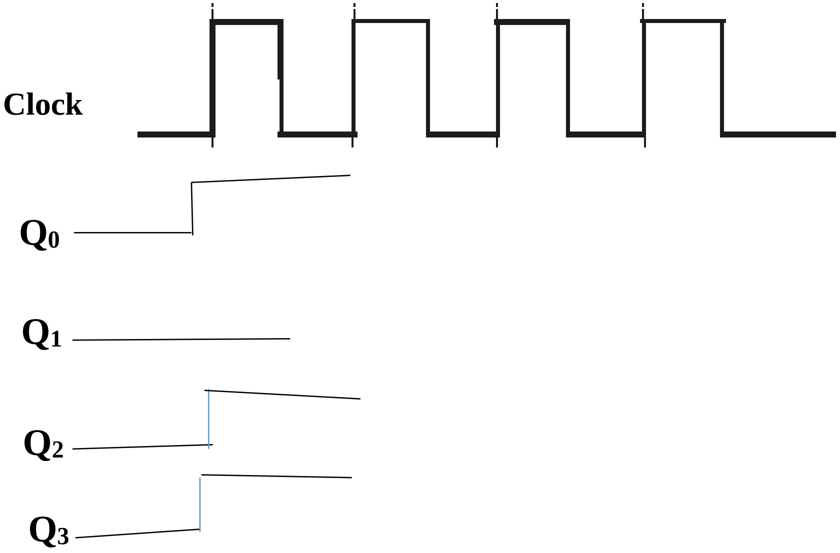


Fig: parallel in parallel out shift register logic diagram

1011


Clock
Q_0
Q_1
Q_2
Q_3