UNIT III SYNCHRONOUS SEQUENTIAL LOGIC

DIVYA ARIVALAGAN
Assistant Professor
Electronics Engineering
MIT Campus
Anna University

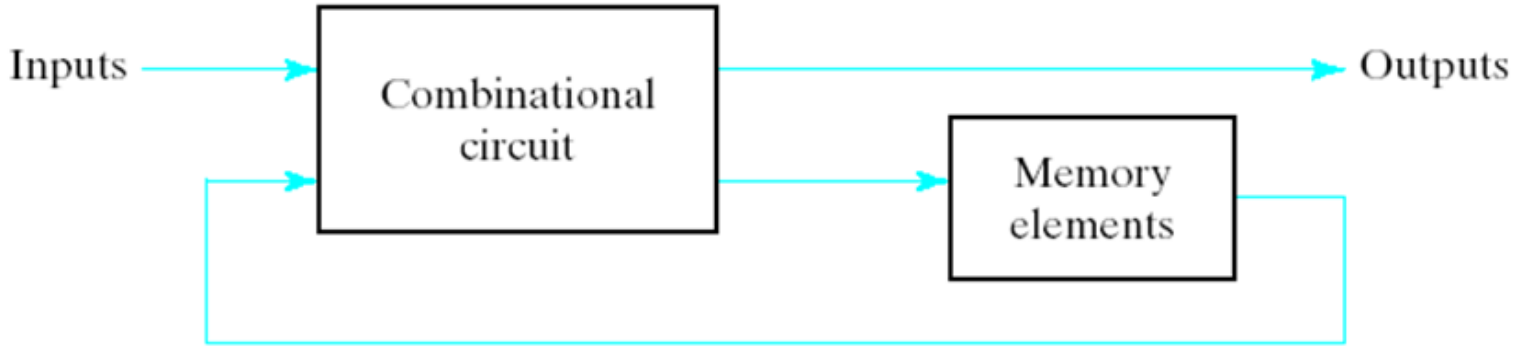
Donald D.Givone, M.Morris Mano,Digital Circuits and Design Pearson

Sequential Circuits

- Storage Elements: Latches , Flip-Flops
- Analysis of Clocked Sequential Circuits
- State Reduction and Assignment
- Design Procedure
- Registers
- Counters
- HDL Models of Sequential Circuits.

Sequential Circuits

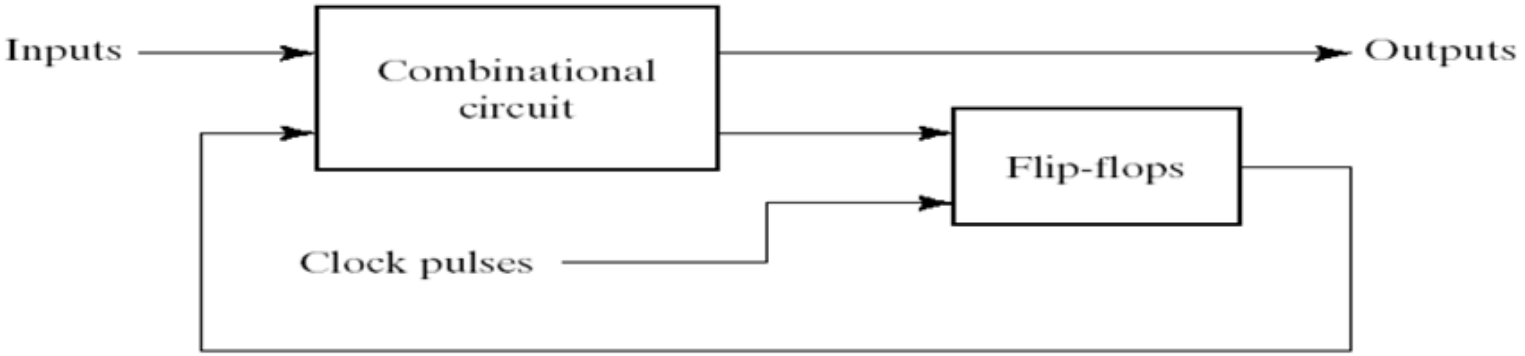
Every digital system is likely to have combinational circuits, most systems encountered in practice also include storage elements, which require that the system be described in term of sequential logic.



Block Diagram of Sequential Circuit

Synchronous Clocked Sequential Circuit

A sequential circuit may use many flip-flops to store as many bits as necessary. The outputs can come either from the combinational circuit or from the flip-flops or both.



(a) Block diagram

(b) Timing diagram of clock pulses

Synchronous Clocked Sequential Circuit

Latches

- Latches are basic storage elements
- Storage elements that operate with signal levels are referred to as latches.
- These are controlled by a clock transition are flip flops.

Flip Flop (Clocked Latches)

- A flip flop is a binary storage device capable of storing one bit information either 0 or 1
- 2 stable states HIGH and LOW
- It has the property to remain in one state indefinitely until it is directed by an input signal to switch over to the other state .It is also called as Bistable multivibrator.
- The basic formation of flip flop is storage of data
- The memory element used in the clocked seuential circuit are called Flip Flops.

Donald D.Givone, M.Morris Mano,Digital circuits and design,Pearson

Flip-Flops

The state of a latch or flip-flop is switched by a change in the control input.

This momentary change is called a trigger and the transition it cause is said to trigger the flip-flop.

The D latch with pulses in its control input is essentially a flip-flop that is triggered every time the pulse goes to the logic 1 level.

As long as the pulse input remains in the level, any changes in the data input will change the output and the state of the latch.

Applications of Flip Flop

- Registers
- Counters/timers
- As a delay element
- As a memory element

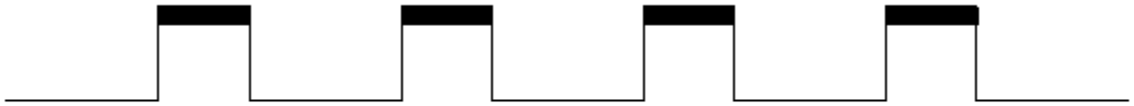
Types of Flip Flop

- SR Flip flop
- JK Flip flop
- D Flip flop
- T Flip flop

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

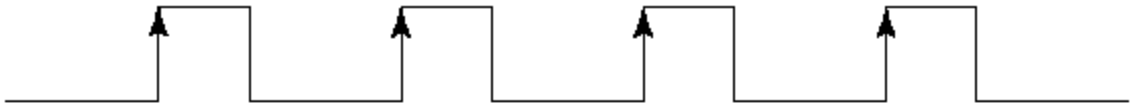
Clock Response in Latch

In Fig (a) a positive level response in the control input allows changes, in the output when the D input changes while the clock pulse stays at logic 1.



(a) Response to positive level

Clock Response in Flip-Flop

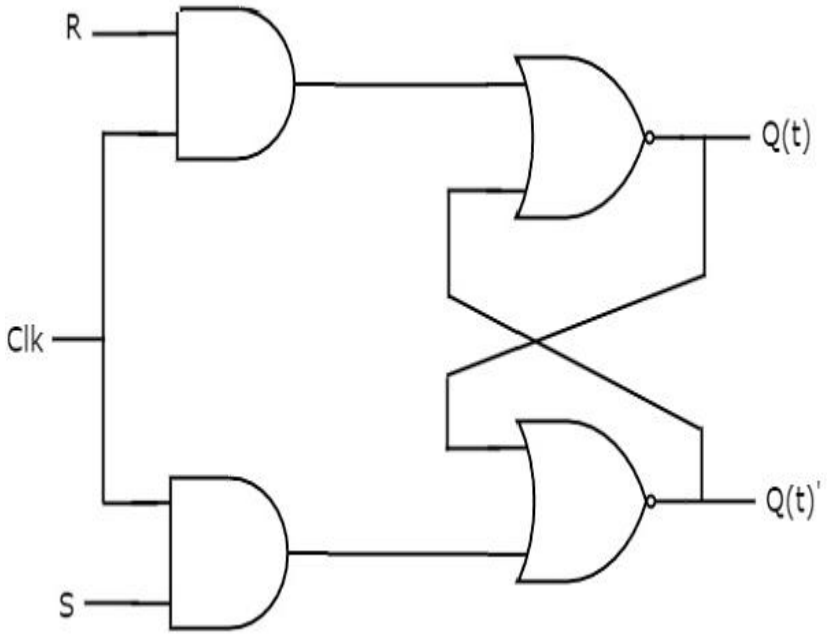


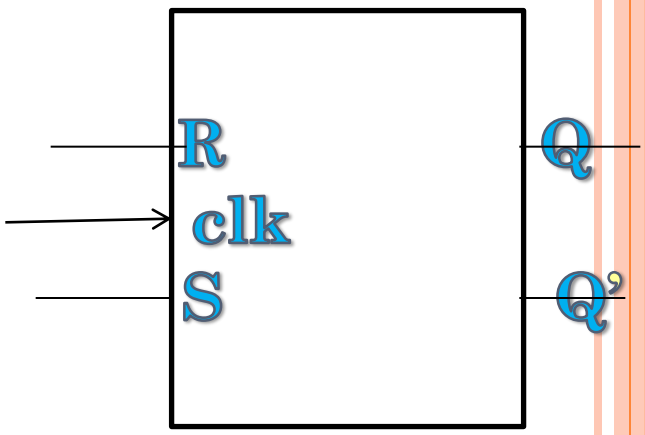
(b) Positive-edge response

(c) Negative-edge response

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

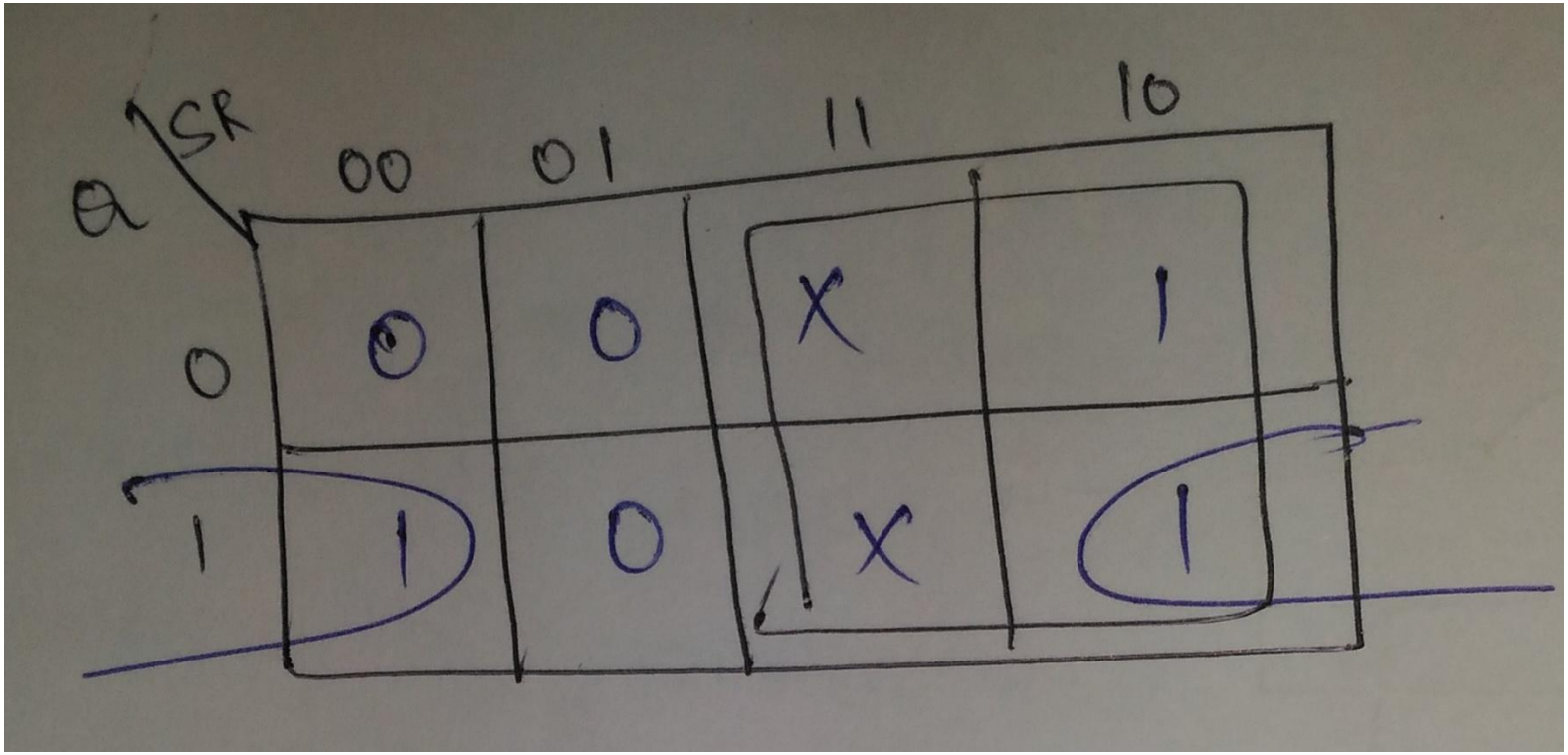
SR Flip Flop

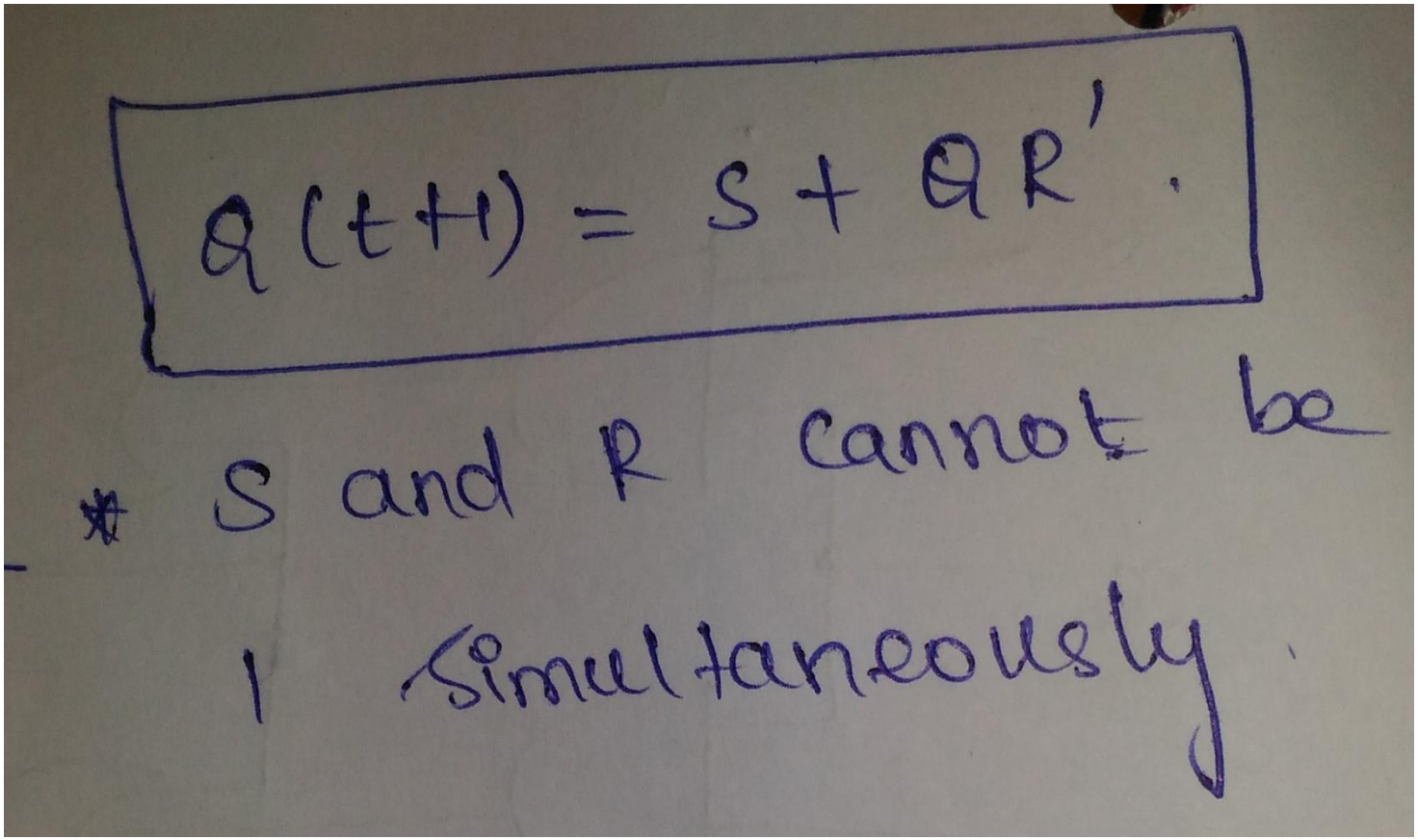




Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

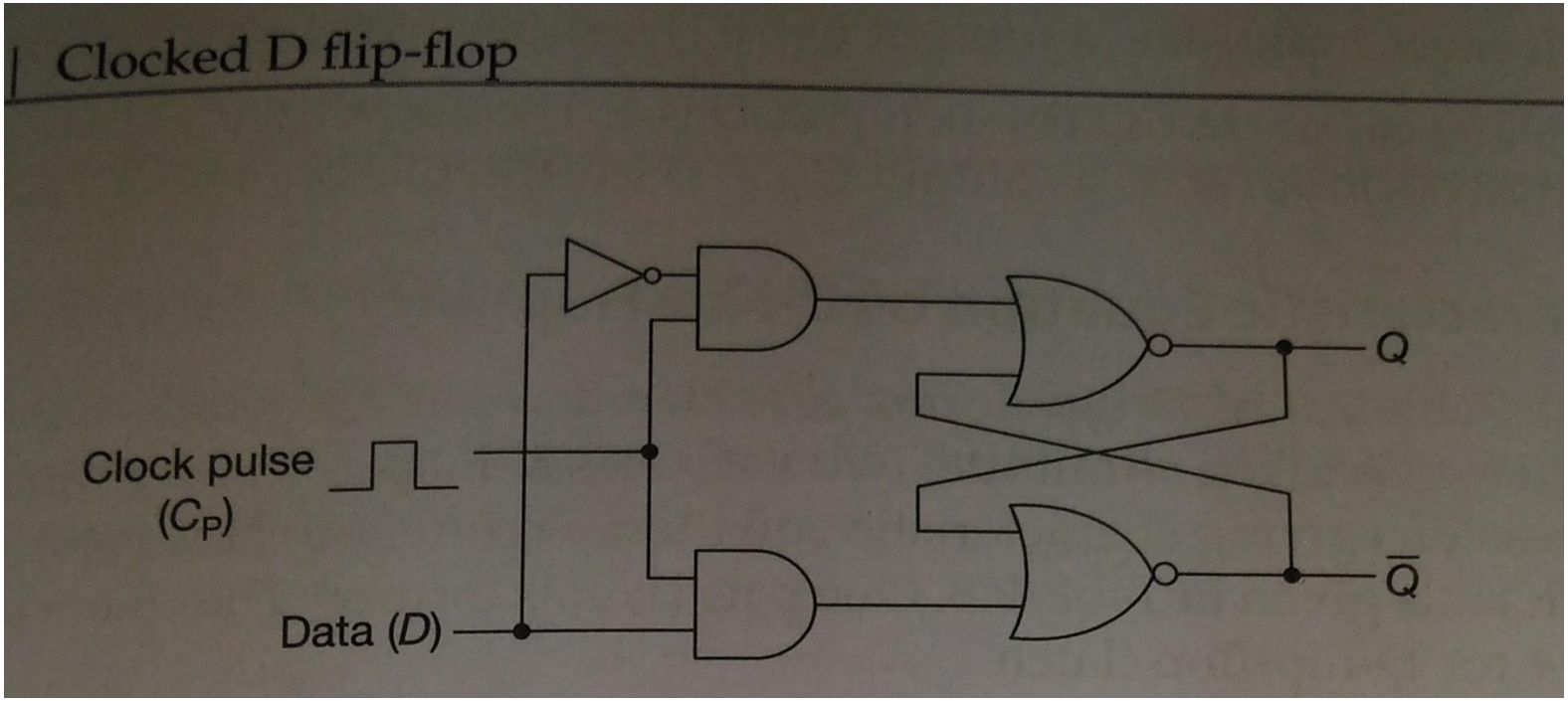
Q	S	R	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	X
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X


Q
SR
00
01
11
10
0
1
0
0
X
1
1
1
0
X
1

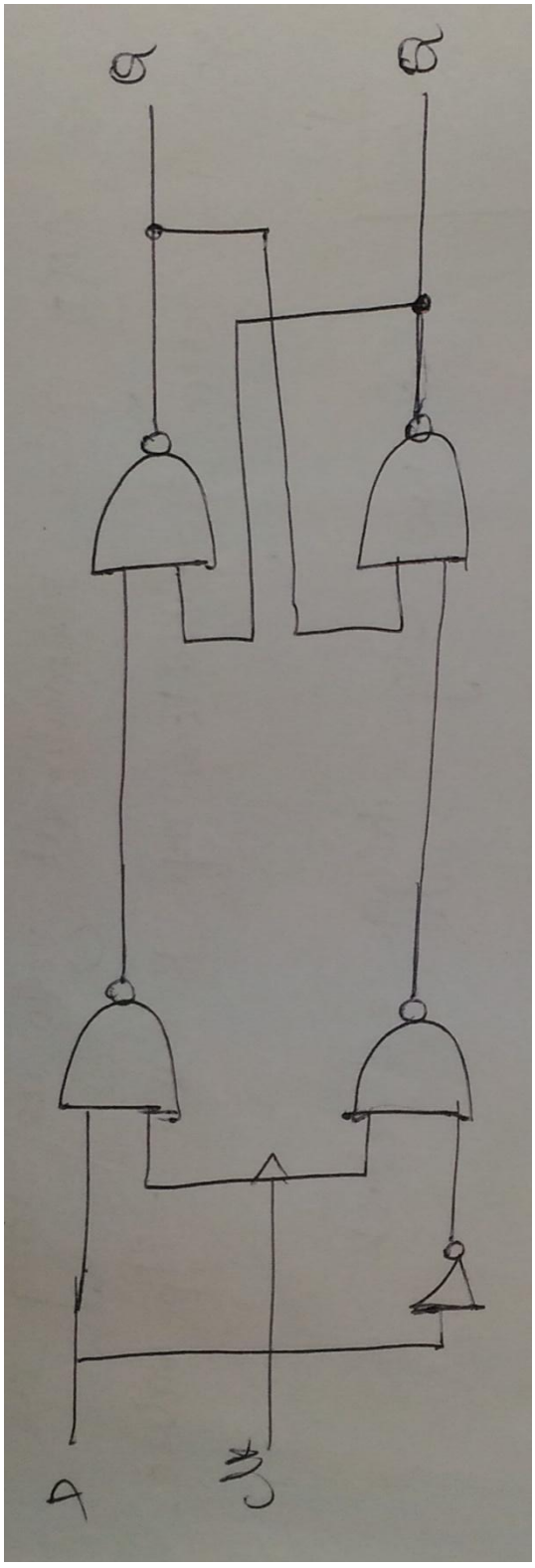


Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

D Flip Flop



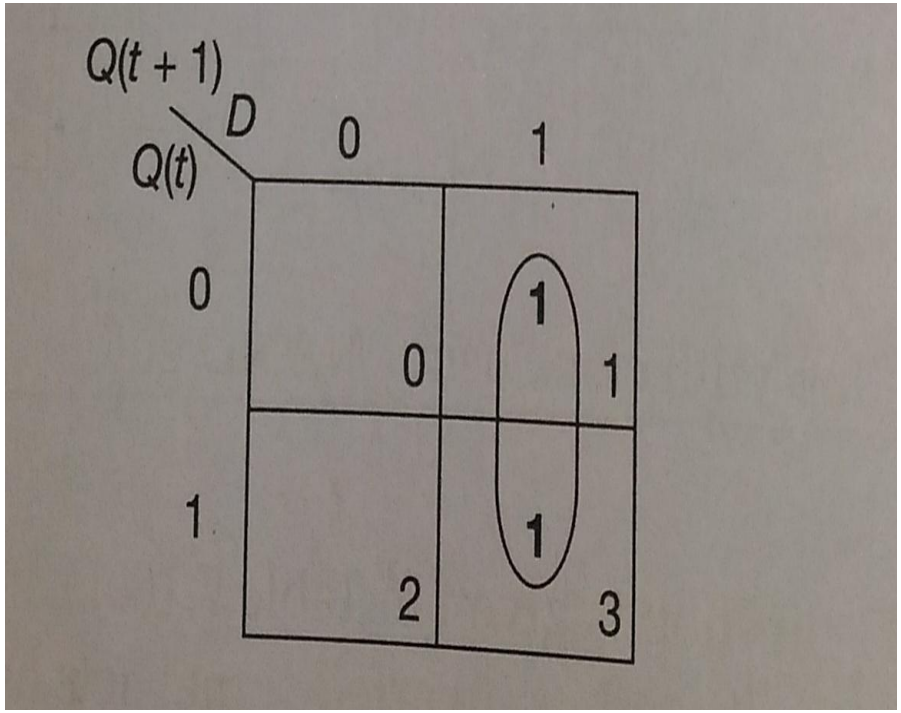
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson



Characteristics of D Flip flop

Inputs		Outputs	
$Q(t)$	D	$Q(t+1)$	$\bar{Q}(t+1)$
0	0	0	1
0	1	1	0
1	0	0	1
1	1	1	0

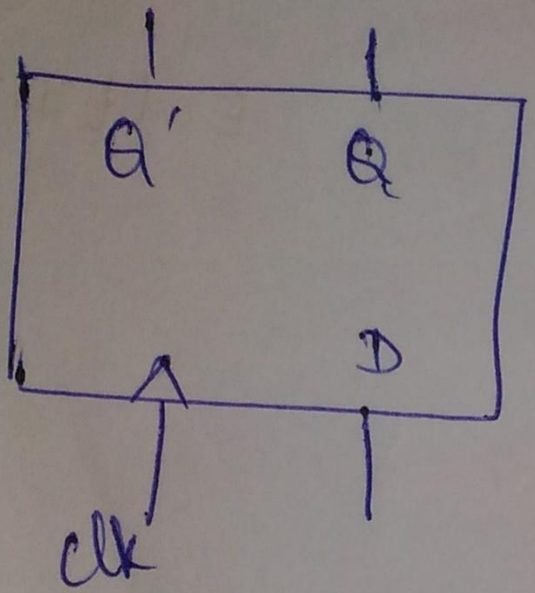
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson



$$Q(t+1) = D$$

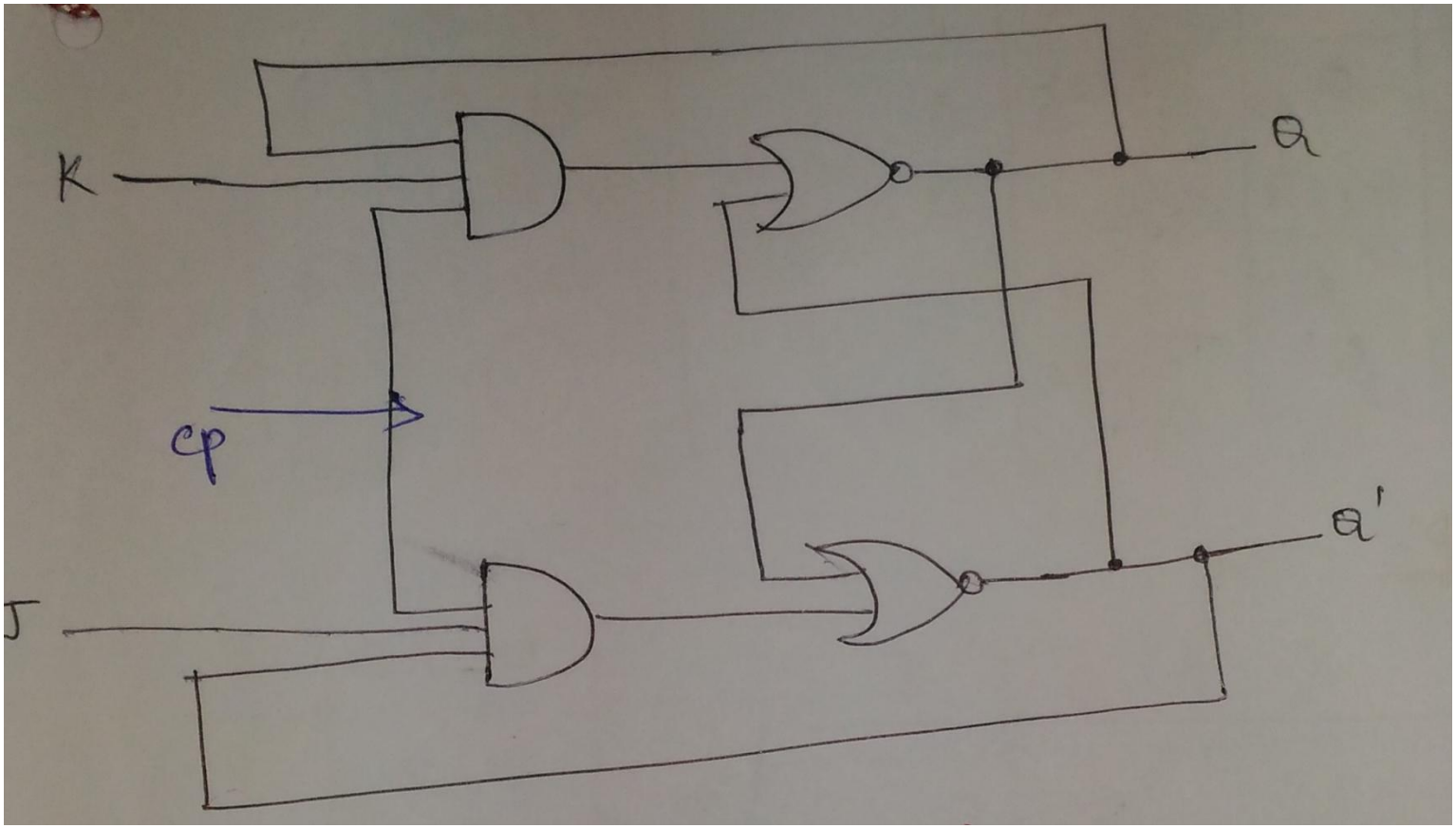
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

logic Representation of D - flip flop

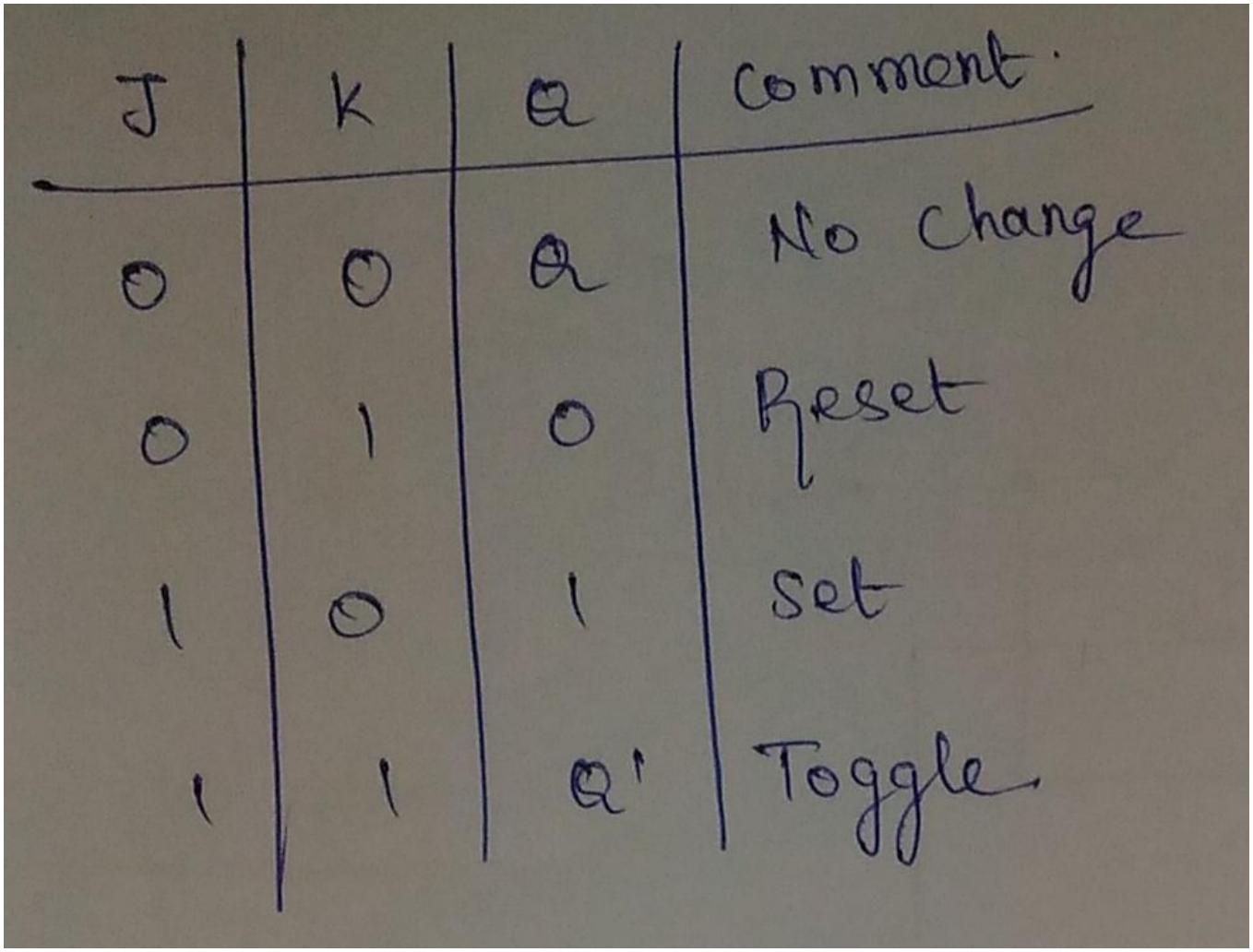


Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

JK Flip Flop



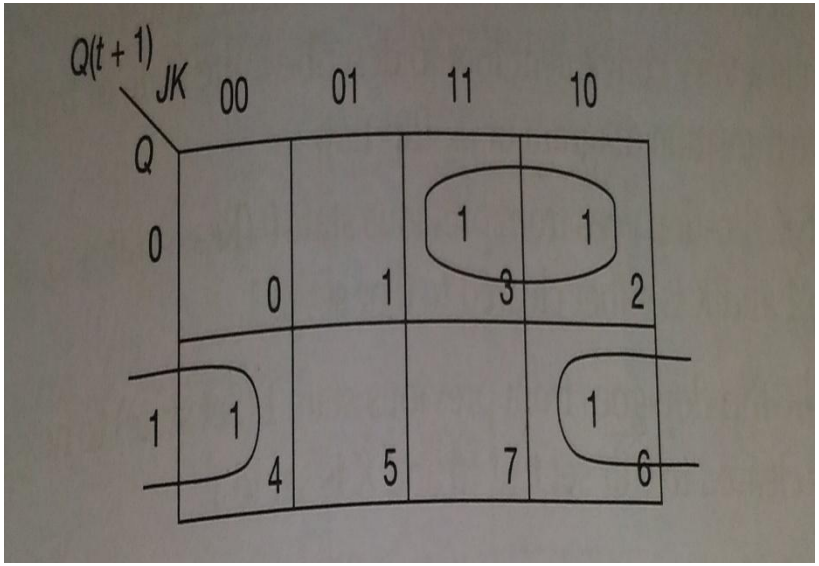
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson



J	K	Q	Comment
0	0	Q	No change
0	1	0	Reset
1	0	1	Set
1	1	Q'	Toggle

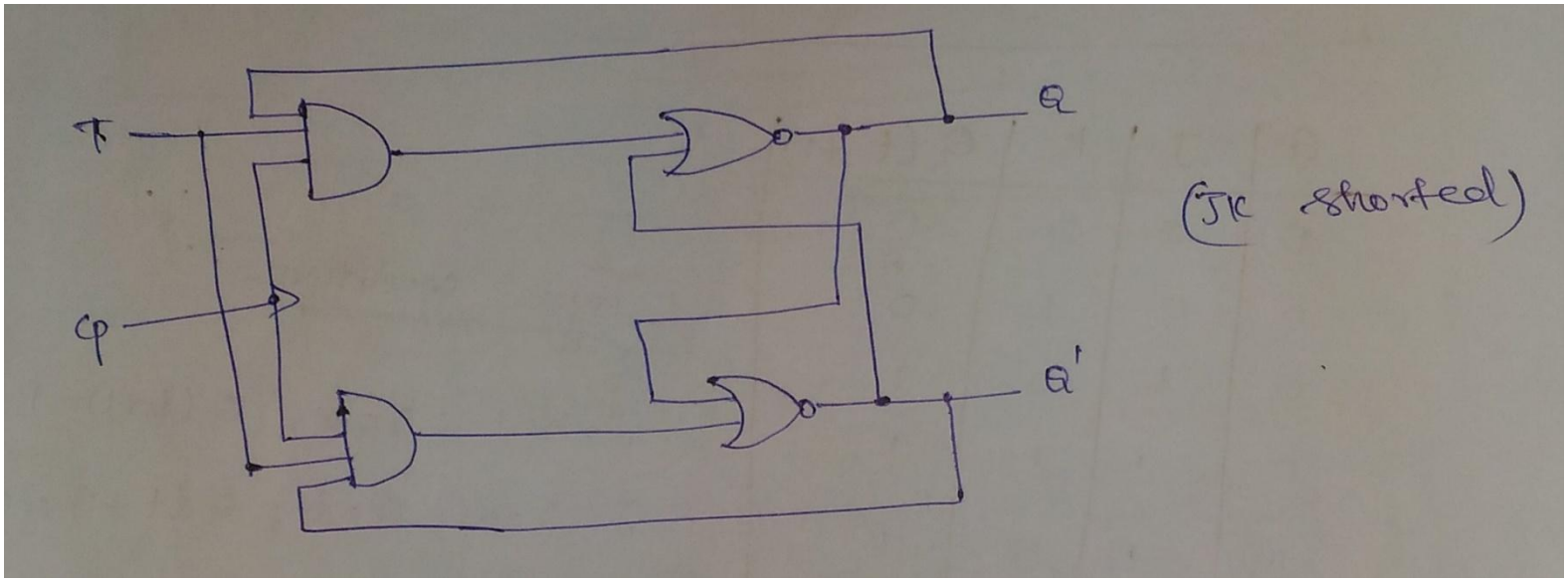
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Q	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0



$$Q(t+1) = J\bar{Q}(t) + \bar{K}Q(t)$$

T Flip Flop



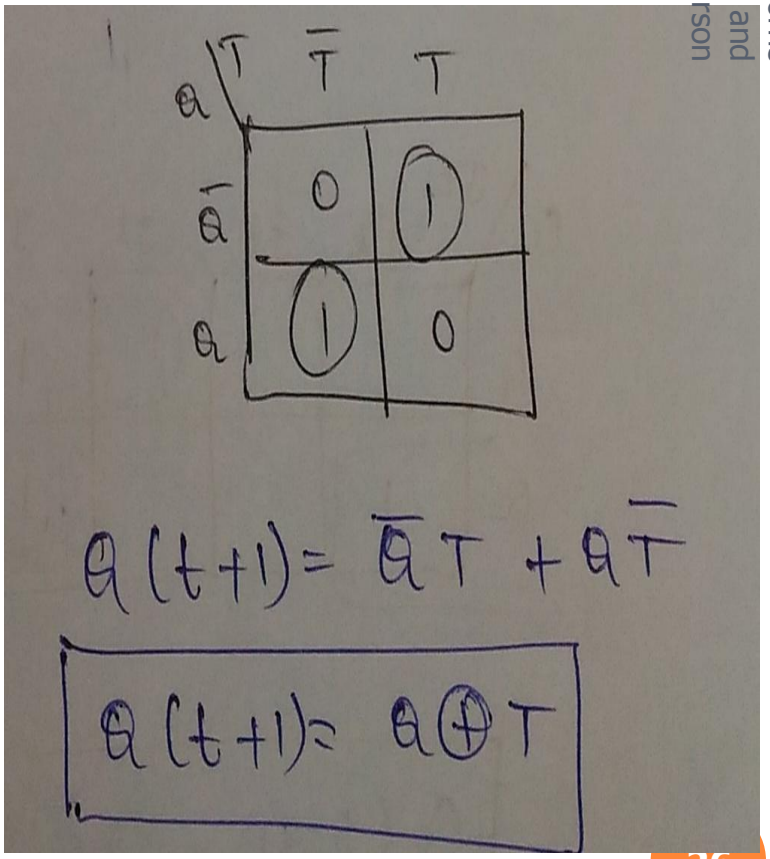
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

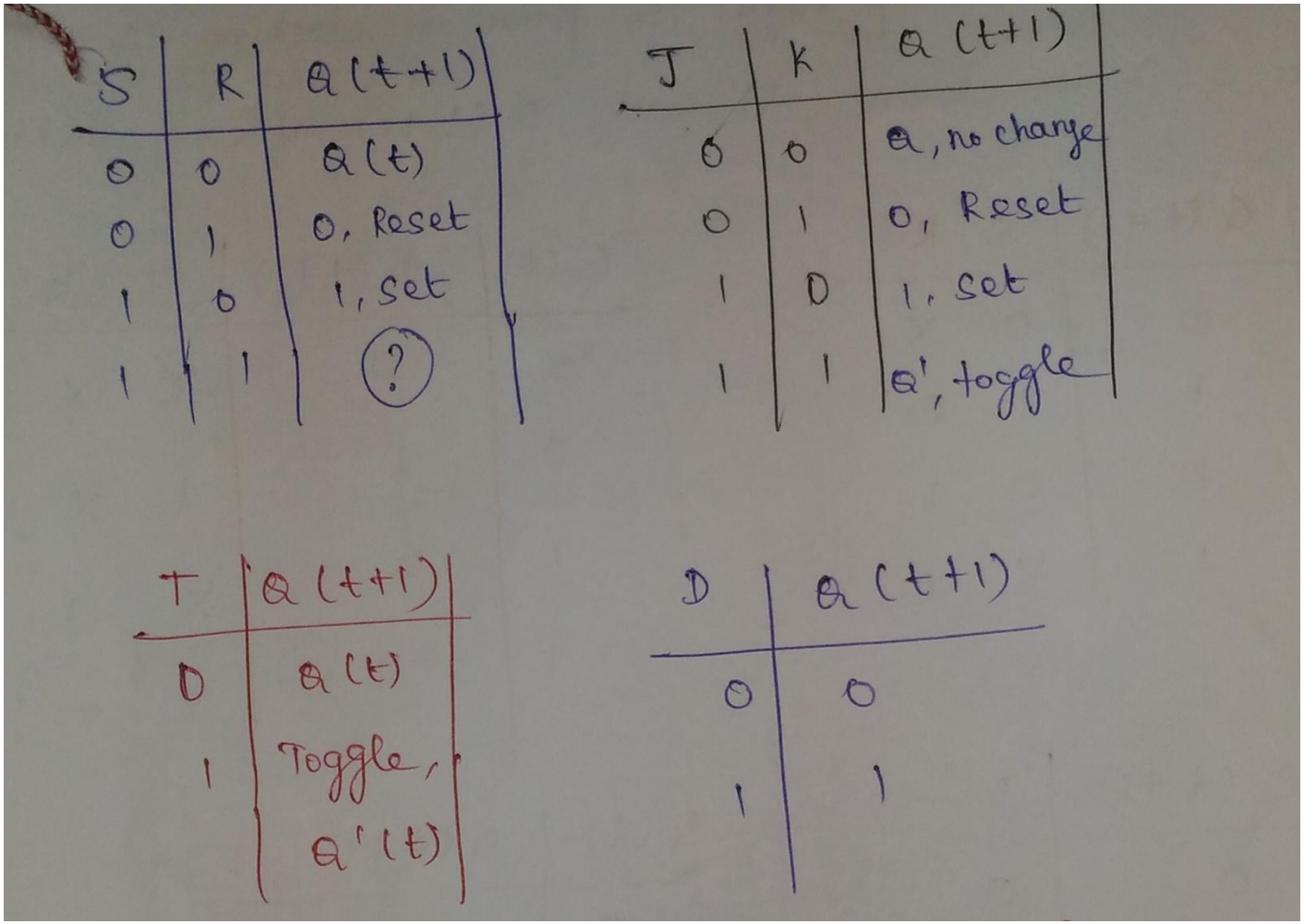
$T=1 \quad Q=0$

$T=0 \quad Q=0.$

Q	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0



Characteristic Table



S	R	Q(t+1)
0	0	Q(t)
0	1	0, Reset
1	0	1, set
1	1	?

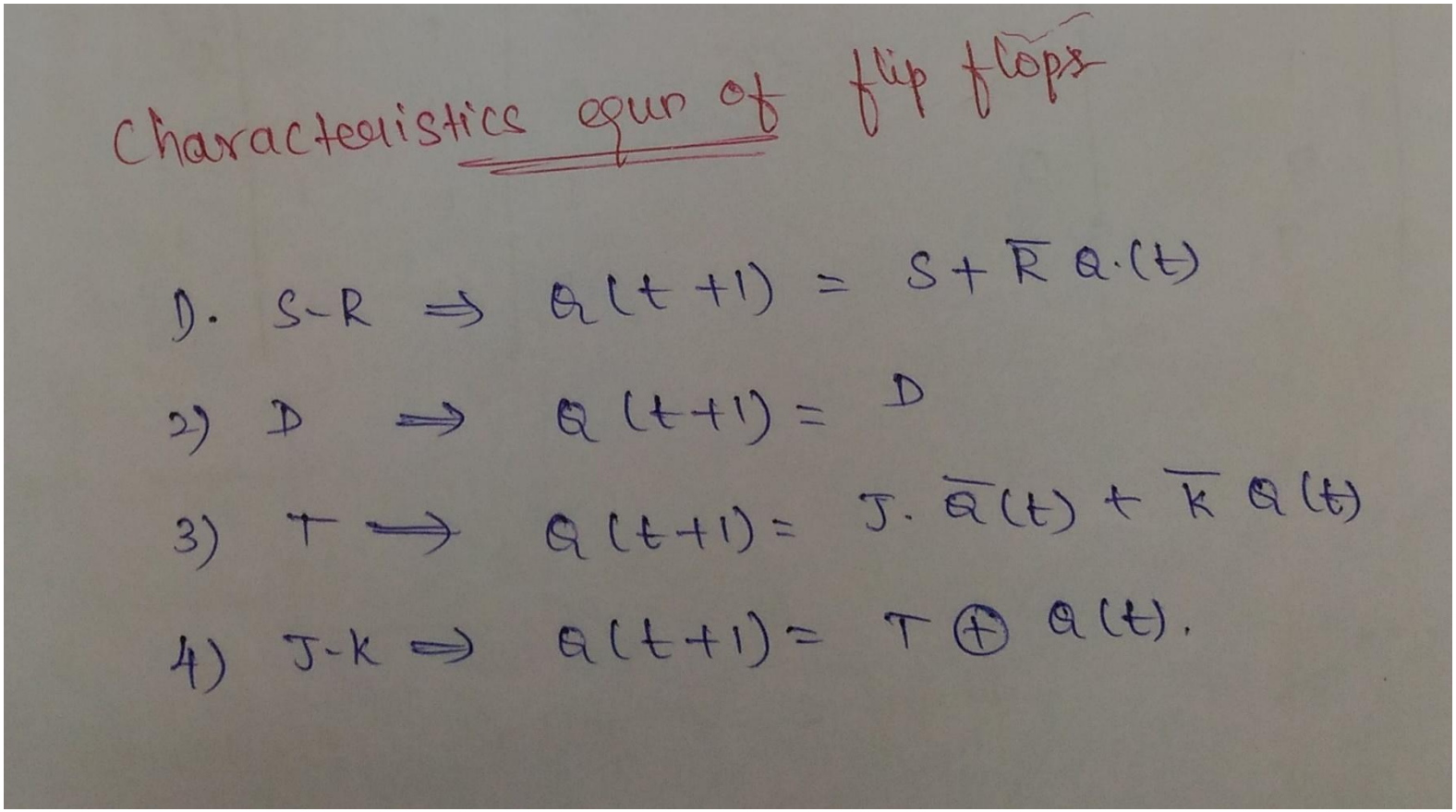
J	K	Q(t+1)
0	0	Q, no change
0	1	0, Reset
1	0	1, set
1	1	Q', toggle

T	Q(t+1)
0	Q(t)
1	Toggle, Q'(t)

D	Q(t+1)
0	0
1	1

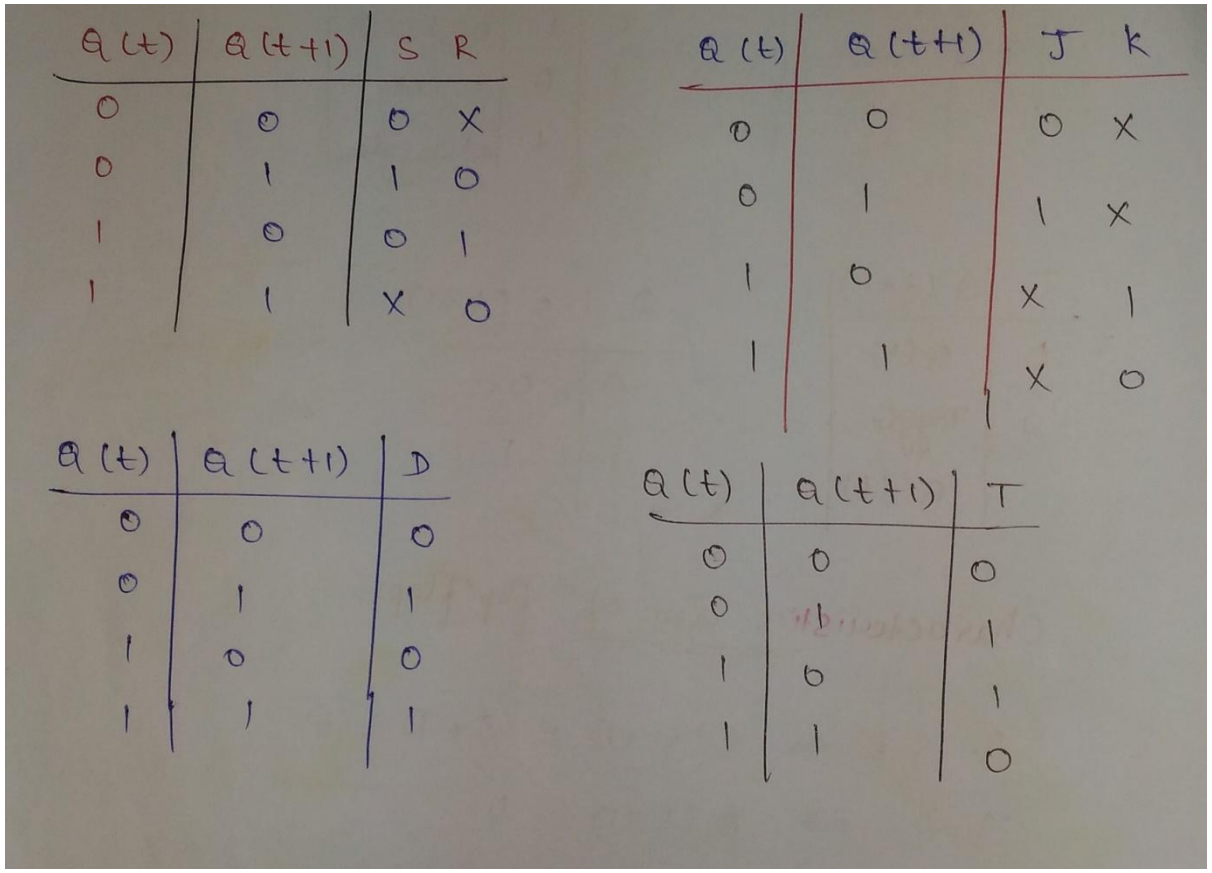
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Characteristic Equation



Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Excitation Table



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

Q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Q(t)	Q(t+1)	D
0	0	0
0	1	1
1	0	0
1	1	1

Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Conversion of Flip Flops

SR TO JK

Minterm	Input of J-K-flip-flop		Output of J-K-flip-flop		Input of SR-flip-flop	
	J	*K*	*Q*(*t*)	*Q*(*t* + 1)	*S*	*R*
m_0	0	0	0	0	0	X
m_1	0	0	1	1	X	0
m_2	0	1	0	0	0	X
m_3	0	1	1	0	0	1
m_4	1	0	0	1	1	0
m_5	1	0	1	1	X	0
m_6	1	1	0	1	1	0
m_7	1	1	1	0	0	1

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

$$S(J,K,Q(t)) = \sum m(4,6) + \sum d(1,5)$$

$$R(J,K,Q(t)) = \sum m(3,7) + \sum d(0,2)$$

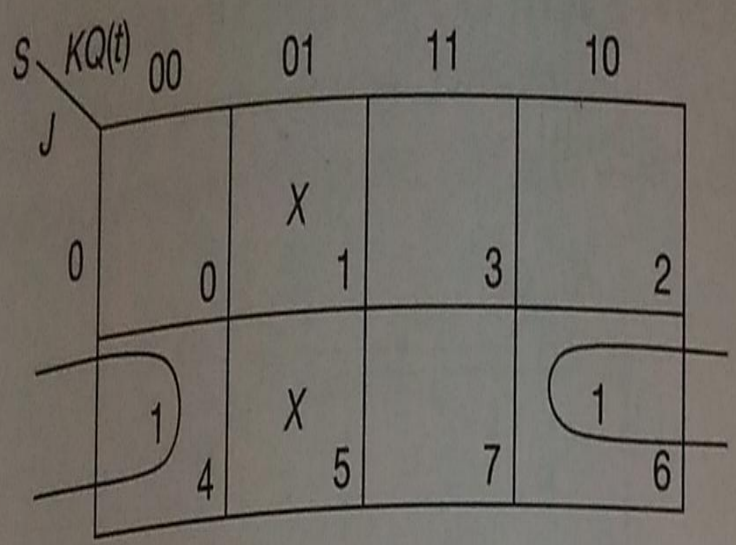


FIGURE 7.51 | K-map for S

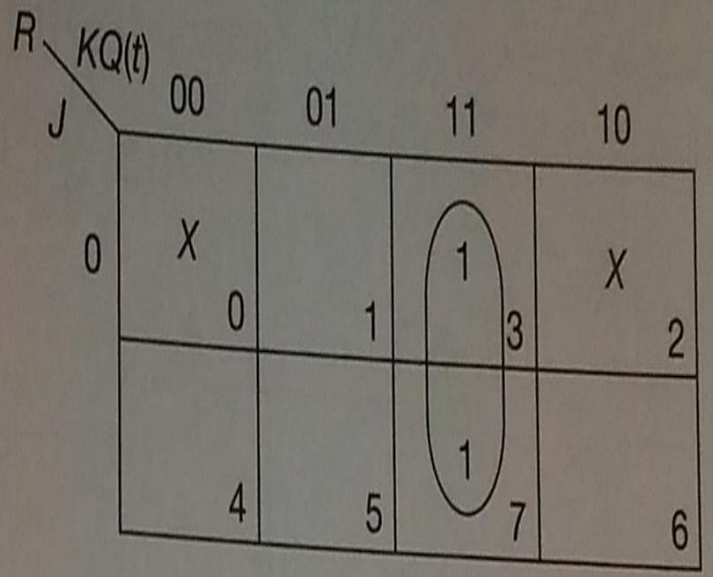
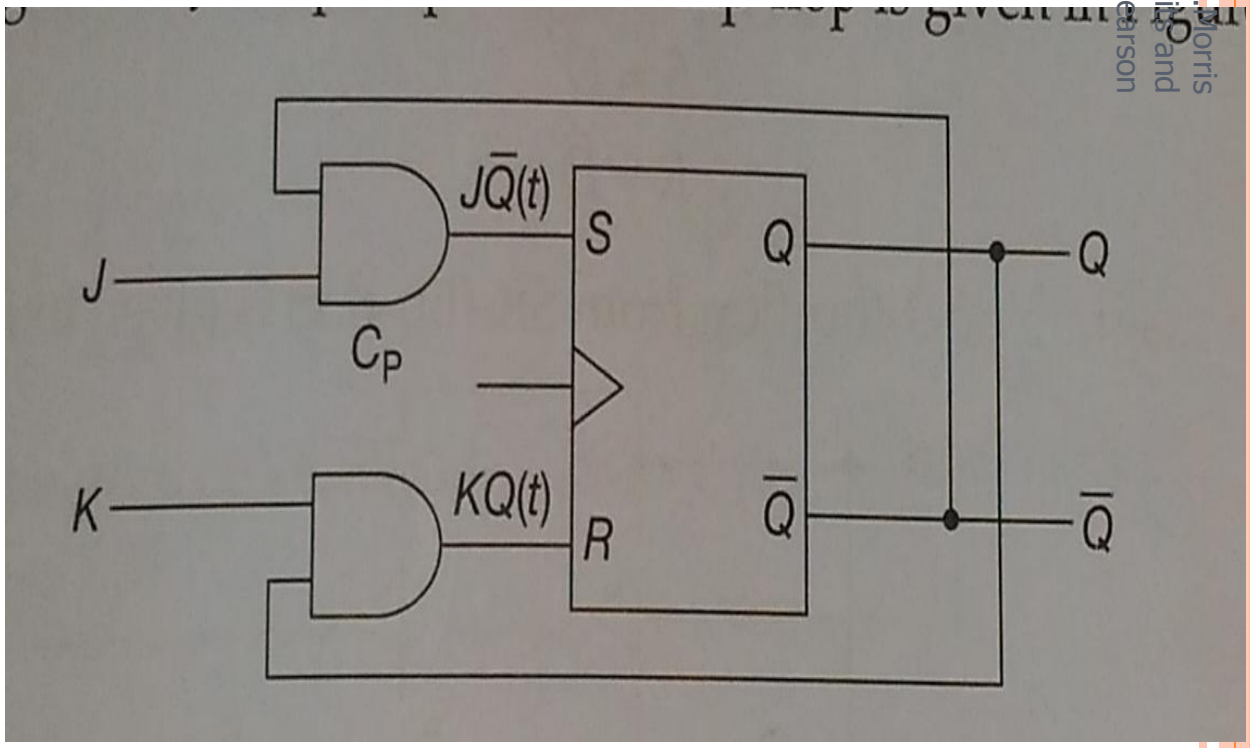


FIGURE 7.52 | K-map for R

$$S = J\bar{Q}(t)$$

$$R = KQ(t)$$



Donald D.Givone, M. Morris Mano,Digital circuits and design Pearson

SR TO D

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Minterm	Input of D-flip-flop	Output of SR-flip-flop		Input of SR-flip-flop	
	D	$Q(t)$	$Q(t+1)$	S	R
m_0	0	0	0	0	X
m_1	0	1	0	0	1
m_2	1	0	1	1	0
m_3	1	1	1	X	0

$$S(D,Q(t)) = \sum m(2) + \sum d(3)$$

$$R(D,Q(t)) = \sum m(1) + \sum d(0)$$

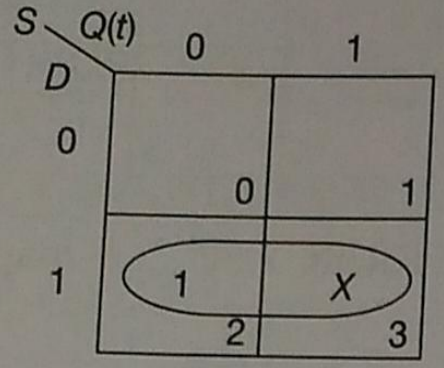


FIGURE 7.54 | K-map for S

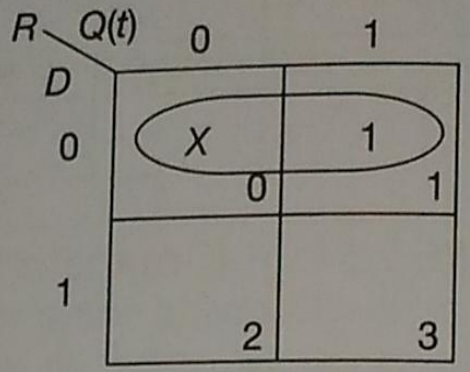


FIGURE 7.55 | K-map for R

$$S = D$$
$$R = \bar{D}$$

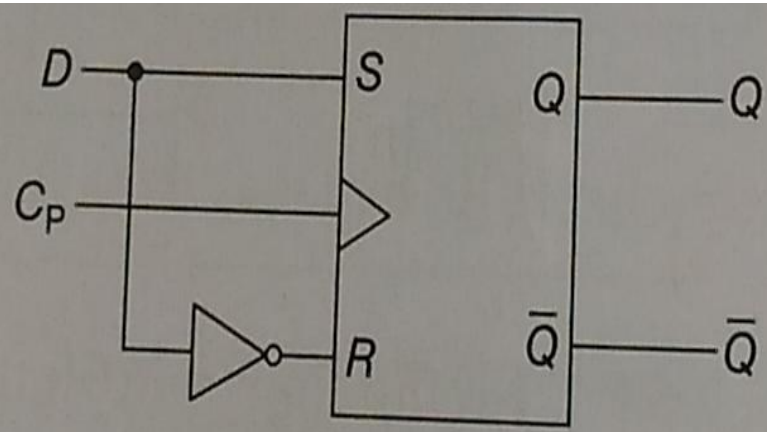


FIGURE 7.56 | D flip-flop from SR flip-flop

JK TO T

Conversion of J-K-flip-flop to T-flip-flop

Minterm	Input of T-flip-flop	Output of T-flip-flop		Input of J-K-flip-flop	
	T	$Q(t)$	$Q(t+1)$	J	K
m_0	0	0	0	0	X
m_1	0	1	1	X	0
m_2	1	0	1	1	X
m_3	1	1	0	X	1

$$J(T,Q) = \sum m(2) + \sum d(1,3)$$

$$K(T,Q) = \sum m(3) + \sum d(0,2)$$

f $J(T, Q)$ and $K($

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

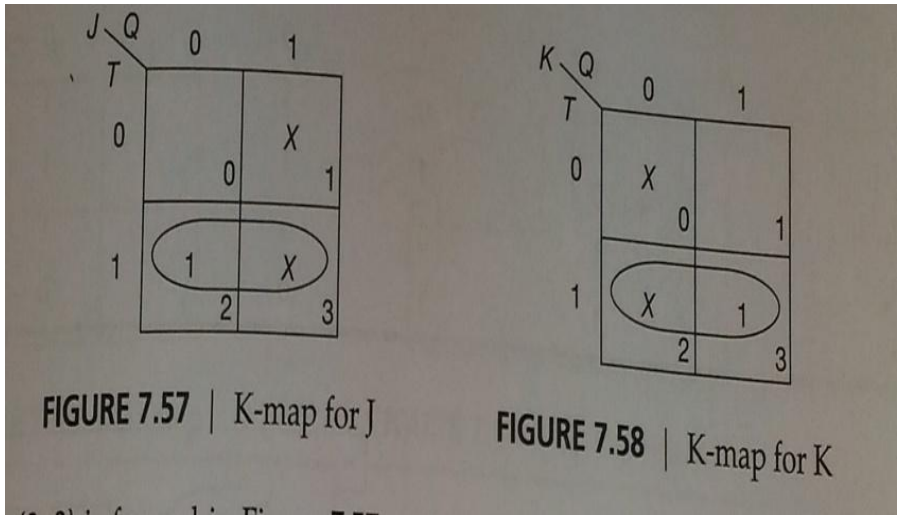
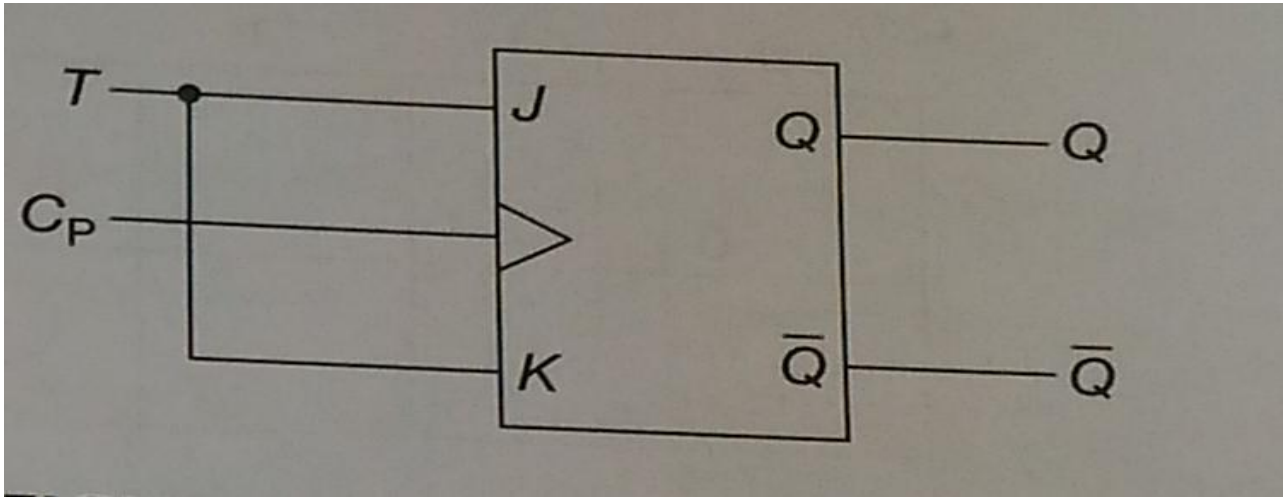


FIGURE 7.57 | K-map for J

FIGURE 7.58 | K-map for K

$$J = T$$

$$K = T$$



D TO JK

Minterm	Input of J-K-flip-flop		Output of J-K-flip-flop		Input of D-flip-flop
	J	K	$Q(t)$	$Q(t+1)$	D
m_0	0	0	0	0	0
m_1	0	0	1	1	1
m_2	0	1	0	0	0
m_3	0	1	1	0	0
m_4	1	0	0	1	1
m_5	1	0	1	1	1
m_6	1	1	0	1	1
m_7	1	1	1	0	0

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

$$D(J,K,Q(t)) = \sum m(1,4,5,6)$$

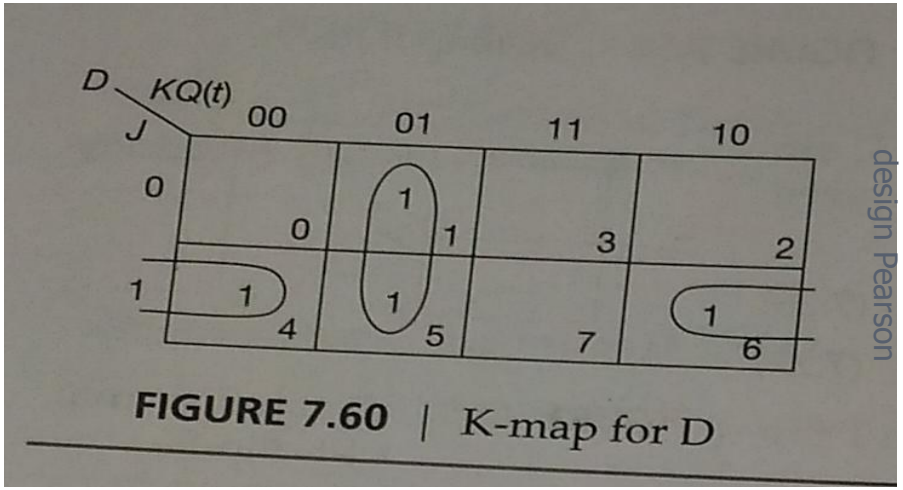


FIGURE 7.60 | K-map for D

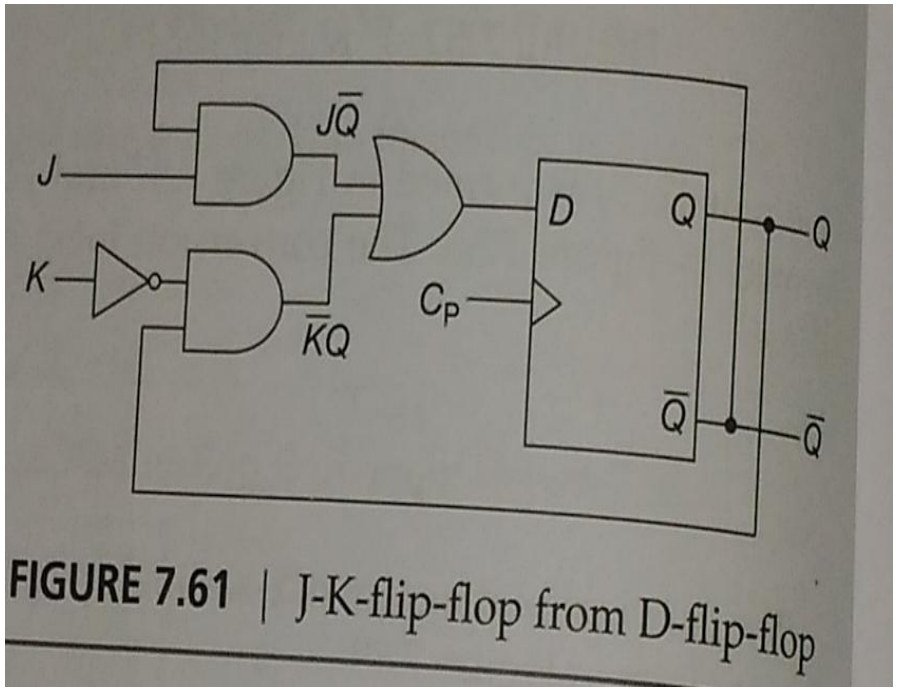


FIGURE 7.61 | J-K-flip-flop from D-flip-flop

$$D = J\overline{Q(t)} + \bar{K}Q(t)$$

T TO D

Conversion of D-flip-flop to T-flip-flop

Minterm	Input of D-flip-flop	Output of D-flip-flop		Input of T-flip-flop
	D	$Q(t)$	$Q(t+1)$	T
m_0	0	0	0	0
m_1	0	1	0	1
m_2	1	0	1	1
m_3	1	1	1	0

$$T(D,Q(t)) = \sum m(1,2)$$

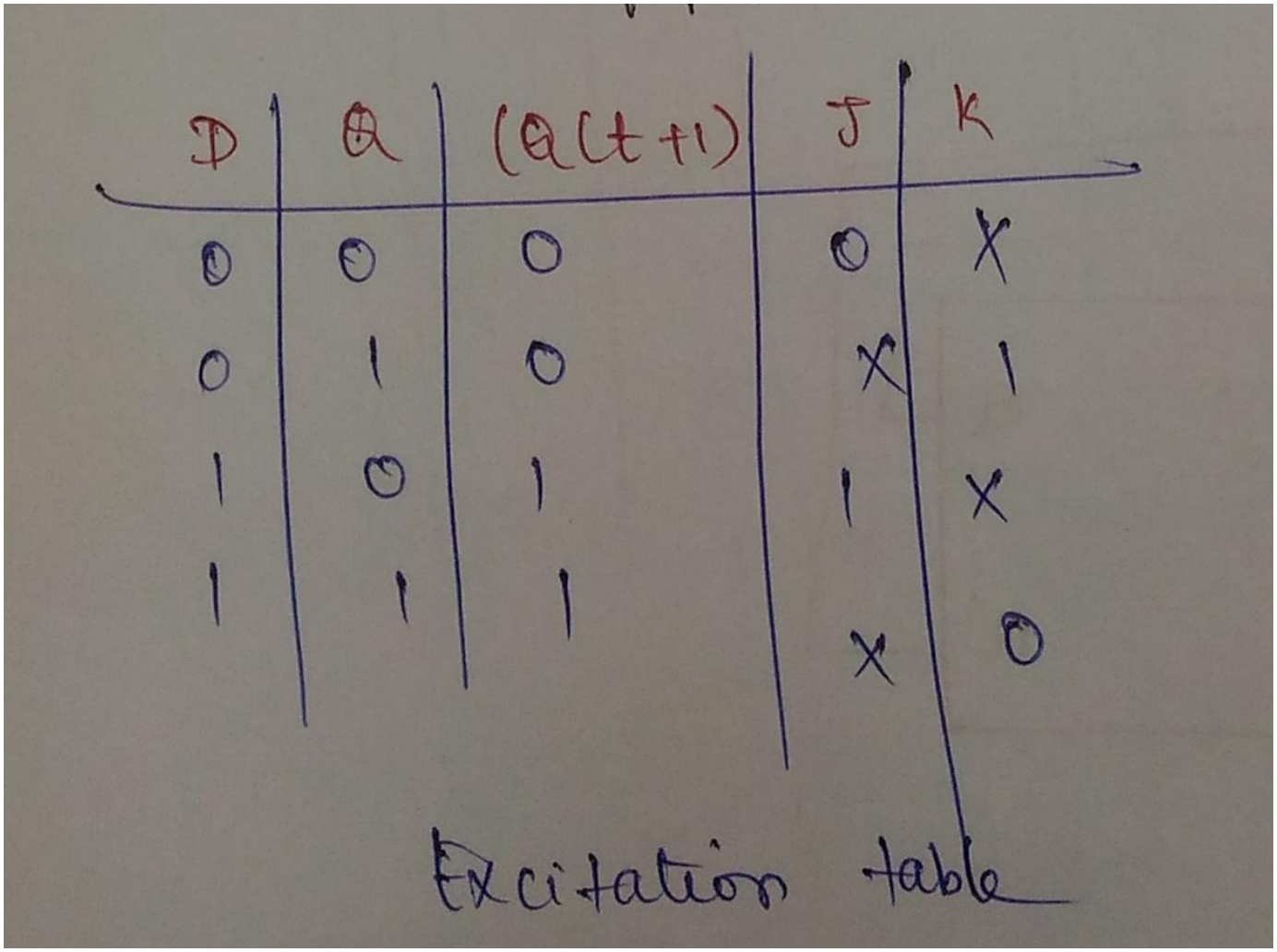
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

$$T = D\overline{Q(t)} + \bar{D}Q(t)$$

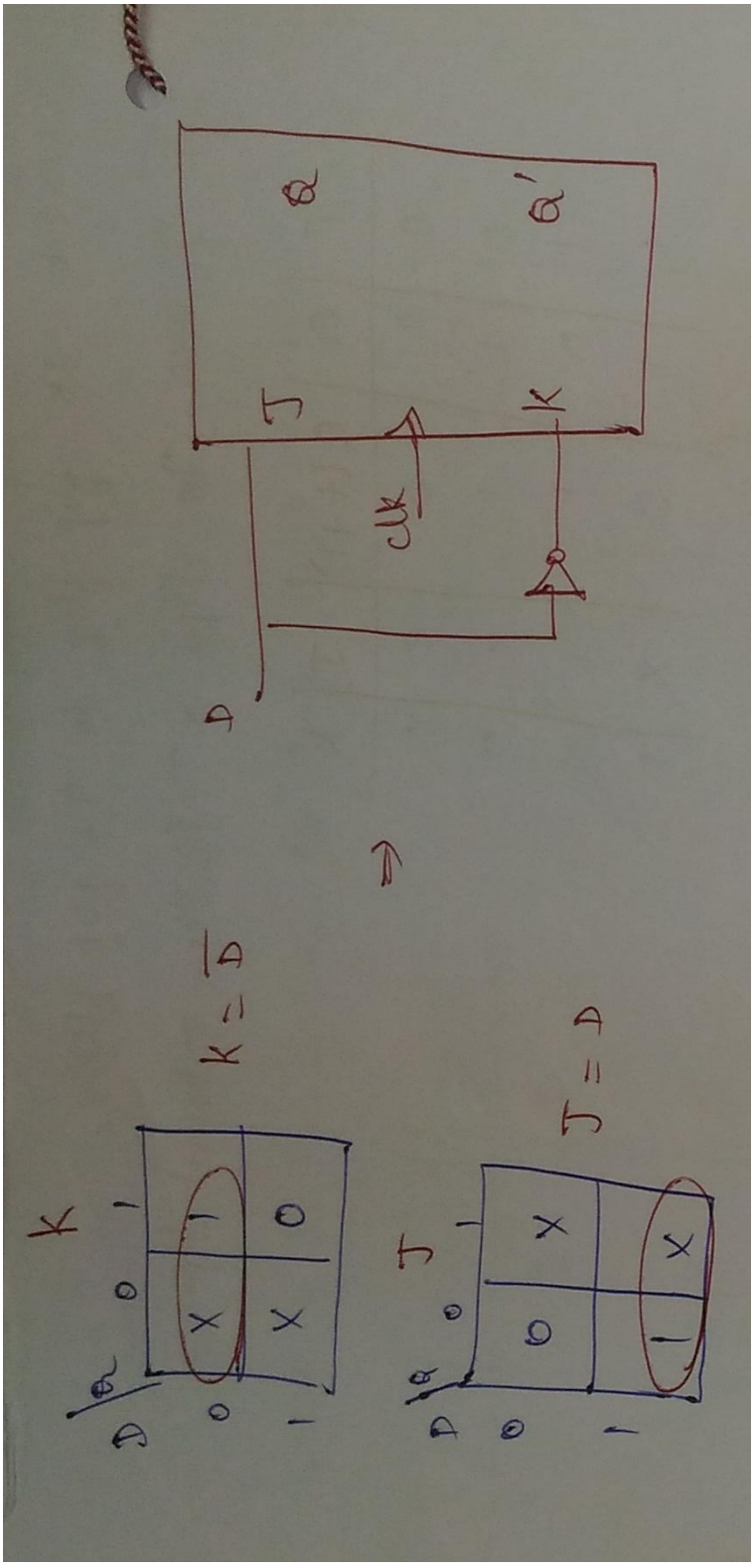
FIGURE 7.62

JK TO D

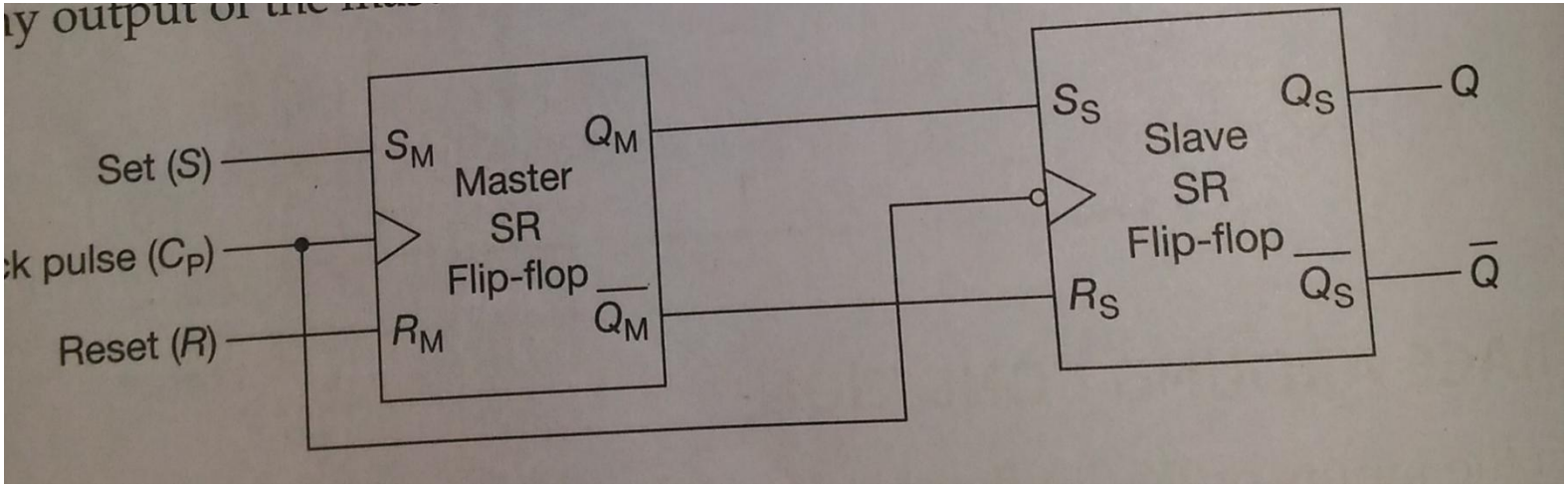


D	Q	Q(t+1)	J	K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	X
1	1	1	X	0

Excitation table



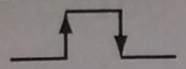
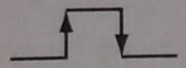
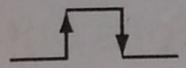
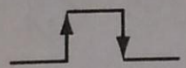
Master-Slave Flip Flop

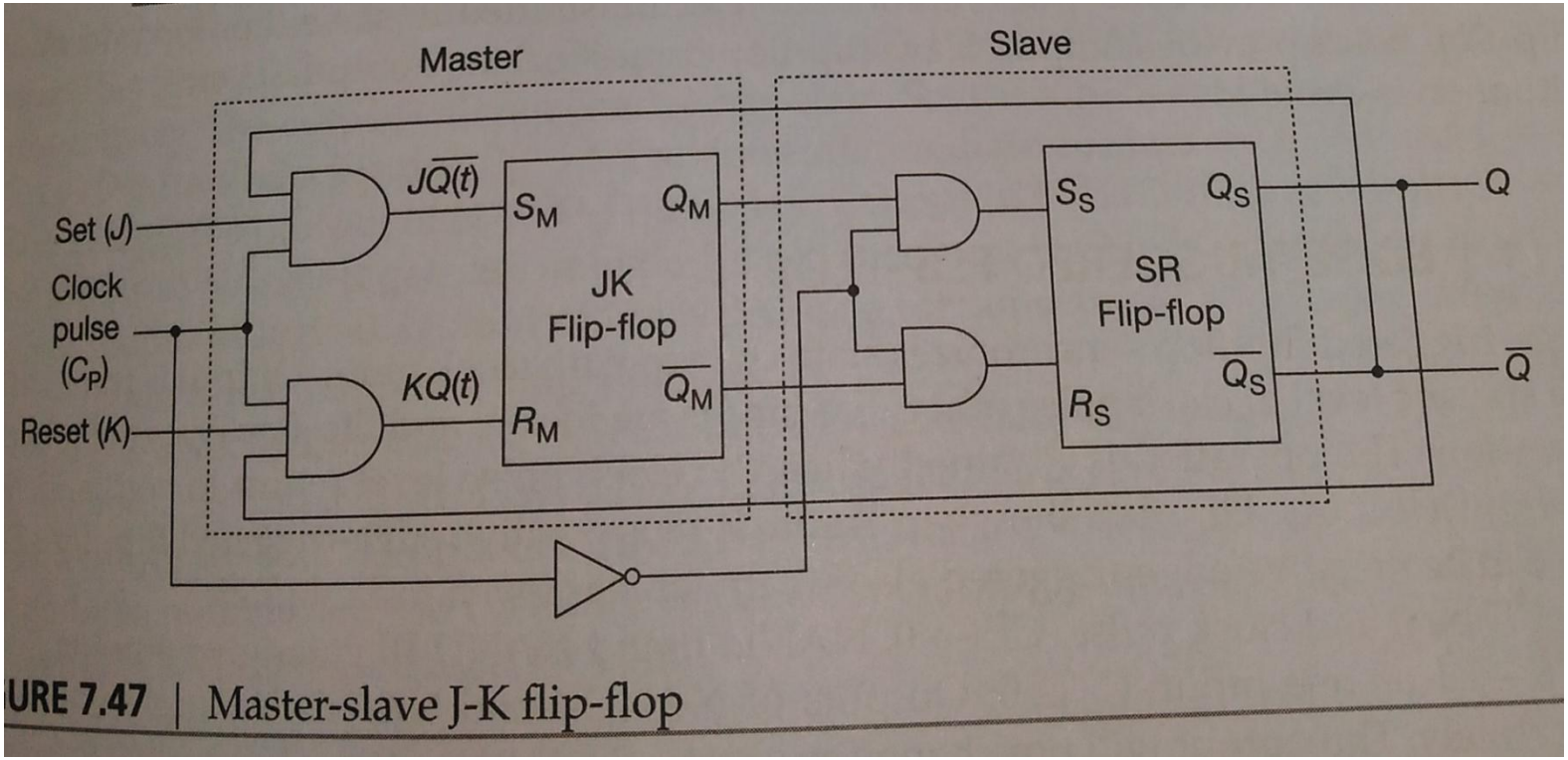


Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

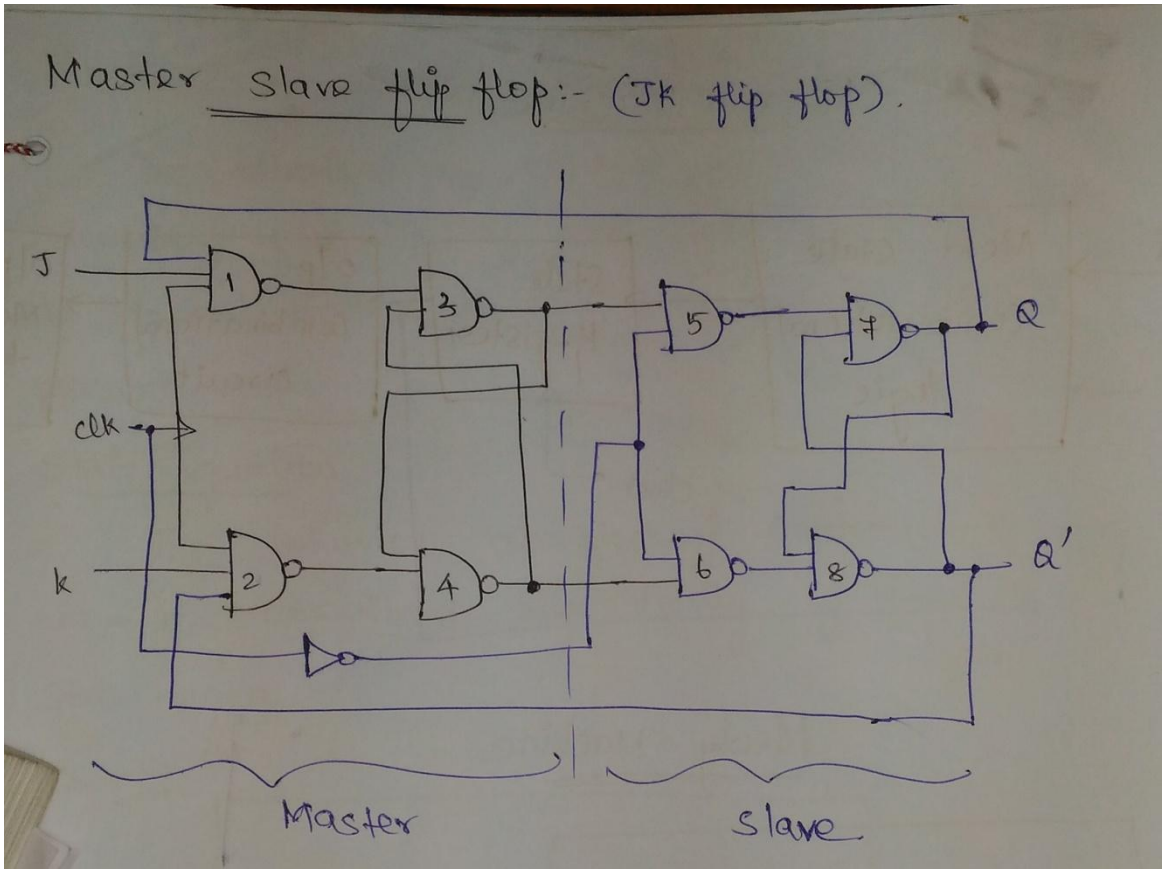
- Master flip flop performs the operation during positive level or edge triggering
- Slave flip flop performs the operation during negative level or edge triggering while sharing the same clock pulse.

TABLE 7.23 | Characteristics of master and slave SR-flip-flop

Inputs			Outputs	Remarks
S (Set)	*R* (Reset)	CP	*Q*(*t* + 1)	
0	0		*Q*(*t*)	Previous state
0	1		0	Set
1	0		1	Clear (Reset)
1	1		?	Indeterminate state



URE 7.47 | Master-slave J-K flip-flop



Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Mealy and Moore Models (1)

- The most general model of a sequential circuit has inputs, outputs, and internal states. It is customary to distinguish between two models of sequential circuits:

⇨ the Mealy model and the Moore model

- They differ in the way the output is generated.
  - In the Mealy model, the output is a function of both the present state and input.
  - In the Moore model, the output is a function of the present state only.

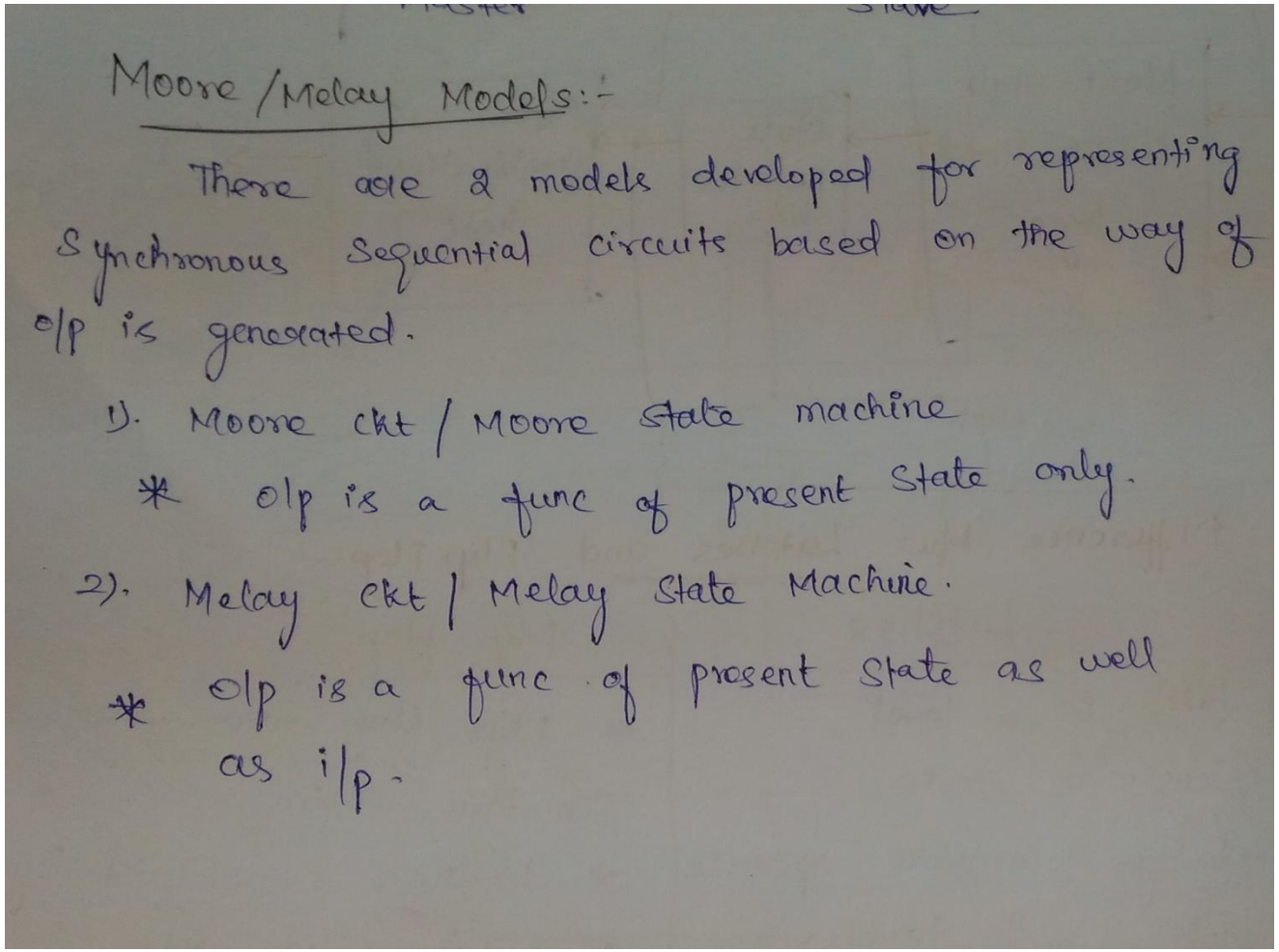
Mealy and Moore Models (2)

When dealing with the two models, some books and other technical sources refer to a sequential circuit as a finite state machine abbreviated FSM.

- The Mealy model of a sequential circuit is referred to as a Mealy FSM or Mealy machine.

- The Moore model is refereed to as a Moore FSM or Moore machine.

Moore/Melay Models



Moore / Melay Models :-

There are 2 models developed for representing Synchronous Sequential circuits based on the way of o/p is generated.

1). Moore ckt / Moore State machine

* o/p is a func of present state only.

2). Melay ckt / Melay State Machine.

* o/p is a func of present state as well as i/p.

Moore Model

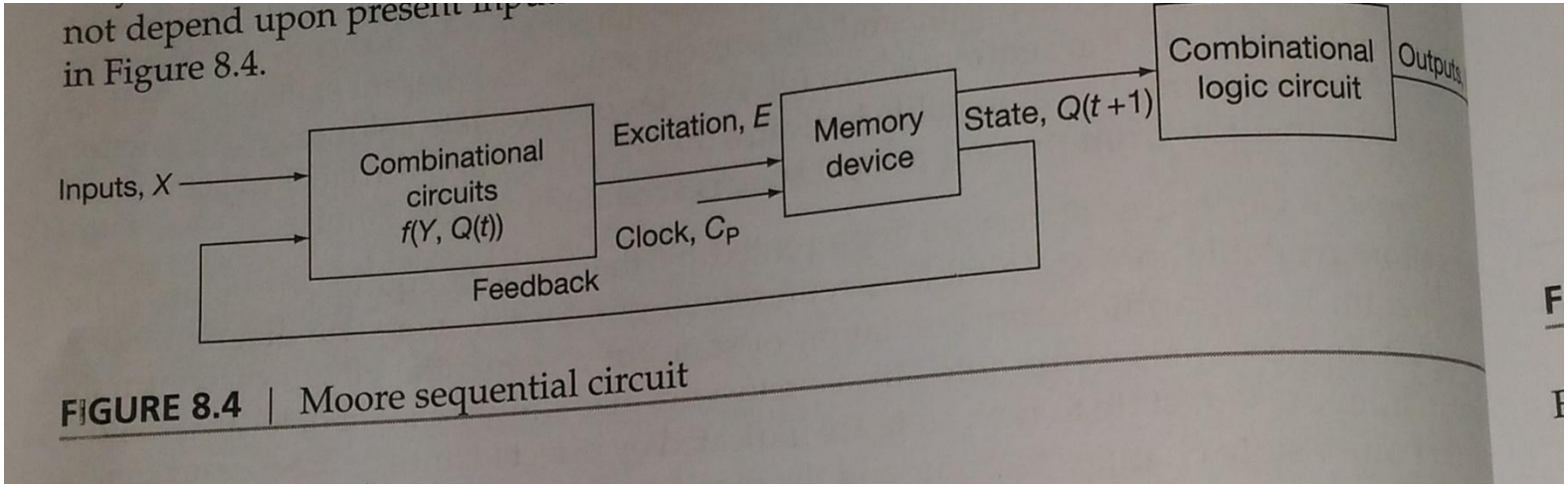


FIGURE 8.4 | Moore sequential circuit

Excitation is given by

$$E = f(Q(t), X)$$

Next state is determined by

$$Q(t+1) = f(E, Q(t))$$

Output is given by

$$Y = f(Q(t))$$

where X is the input variables
E is excitation for the memory device
$Q(t)$ is the present output state (at time t) of memory device.
Y is the output variables

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Melay Model

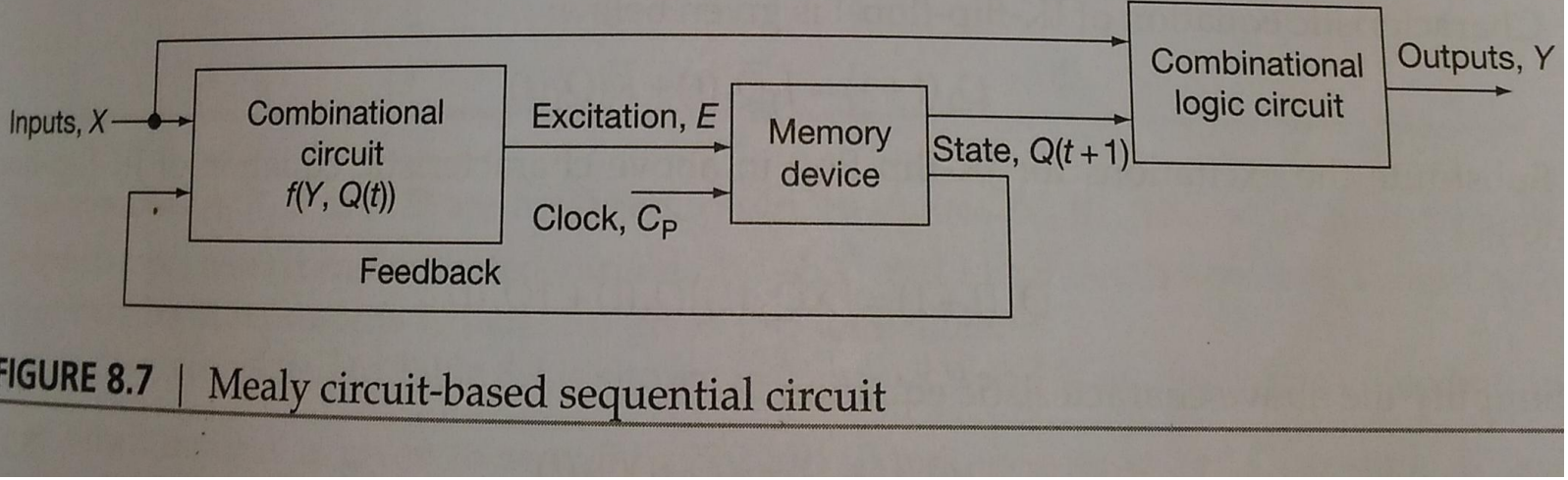


FIGURE 8.7 | Mealy circuit-based sequential circuit

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Excitation is given by

$$E = f(Q(t), X)$$

Next state is determined as

$$Q(t+1) = f(E, Q(t))$$

Output is given by

$$Y = f(X, Q(t))$$

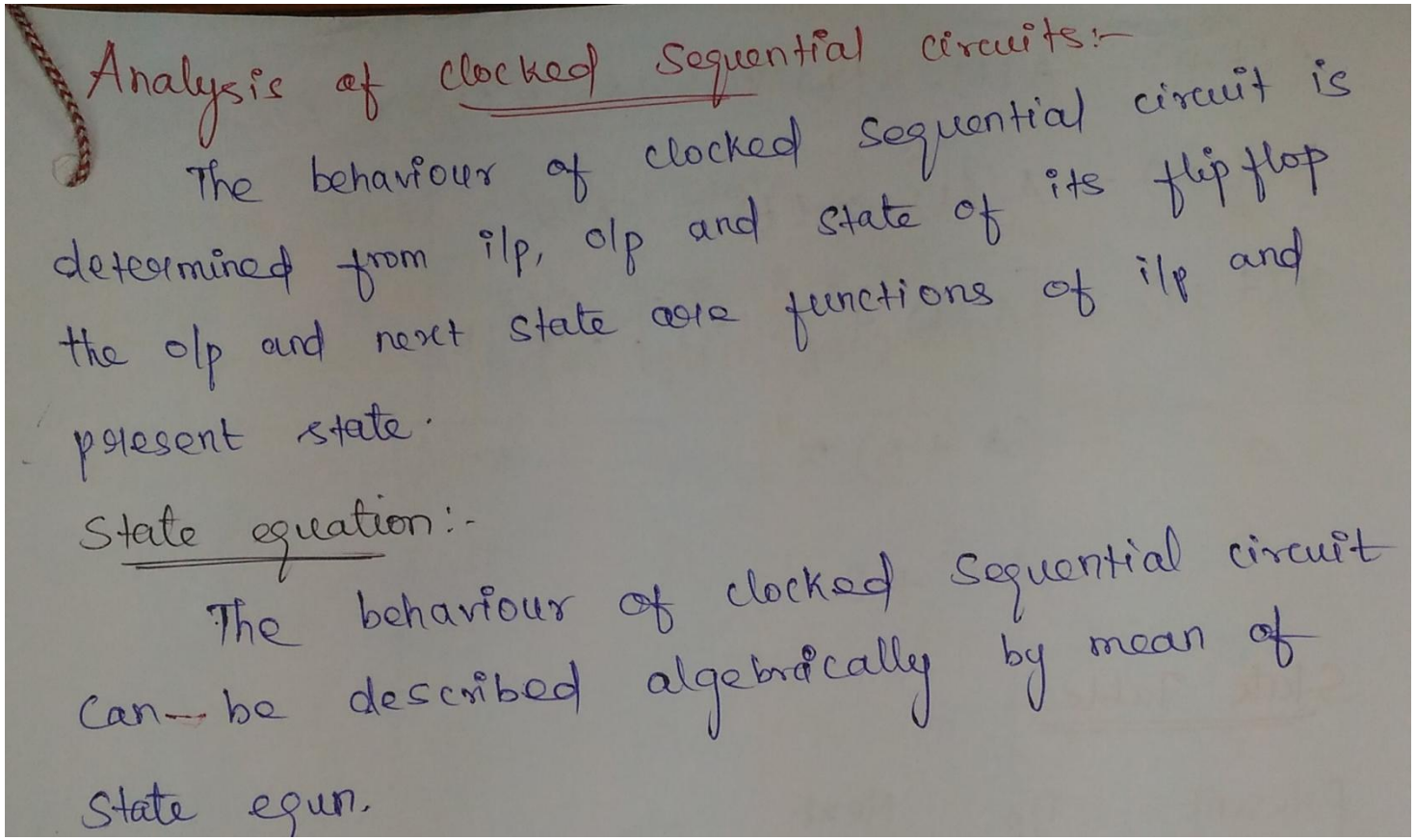
where X is the input variables
E is excitation for the memory device
$Q(t)$ is the present output state (at time t) of memory device.
Y is the output variables

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Analysis of clocked sequential circuits

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Analysis of clocked sequential circuits



Analysis of clocked Sequential circuits:-

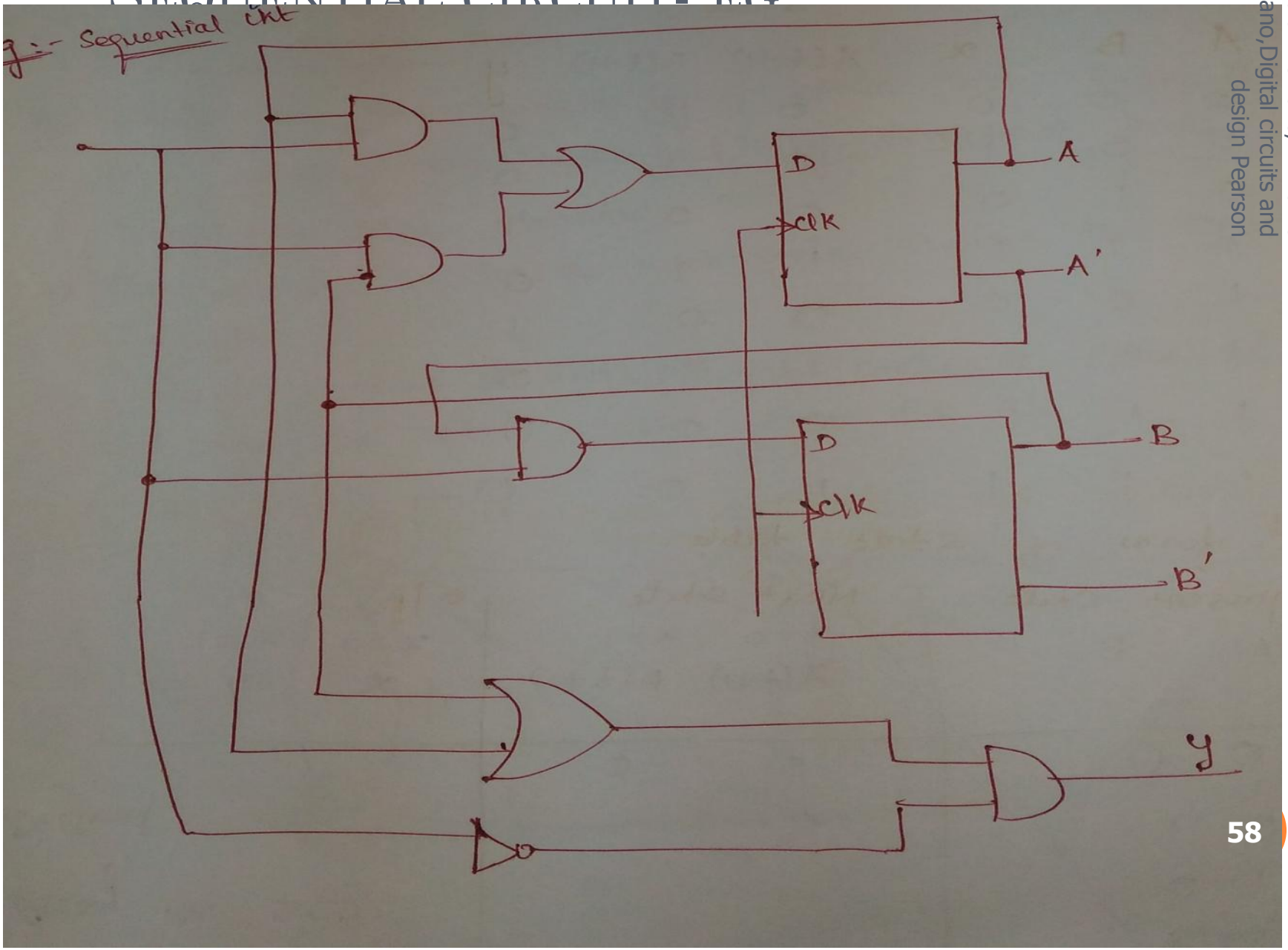
The behaviour of clocked sequential circuit is determined from i/p, o/p and state of its flip flop the o/p and next state are functions of i/p and present state.

State equation:-

The behaviour of clocked sequential circuit can be described algebraically by mean of state equn.

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Sequential circuit- eg



Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

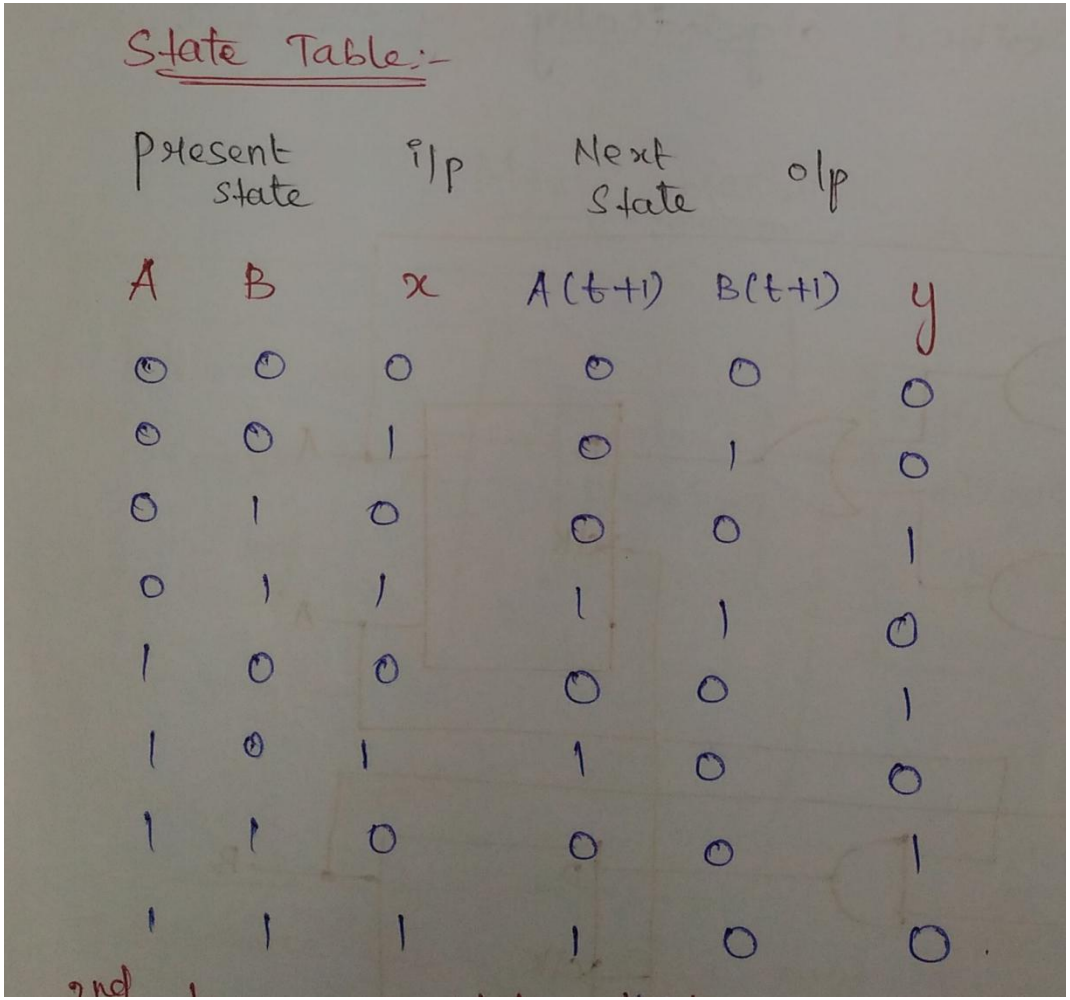
$$A(t+1) = A(t) \cdot x(t) + B(t) \cdot x(t) = Ax + Bx$$

$$B(t+1) = A'(t) \cdot x(t) = A'x$$

$$y(t+1) = [A(t) + B(t)] \cdot x'(t)$$

$$= (A + B)\, x'$$

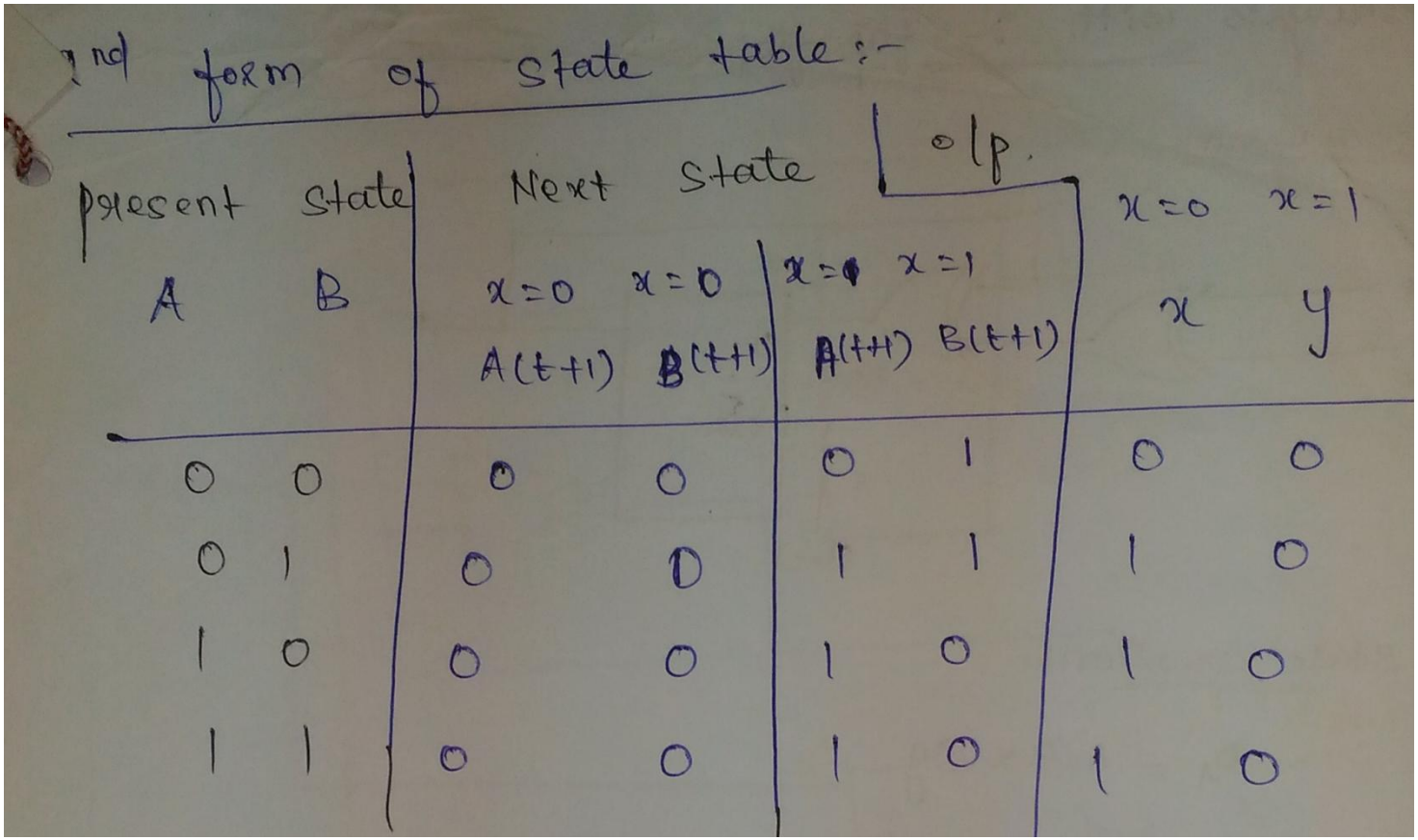
$$= Ax' + Bx'$$



State Table:-

Present state		i/p	Next State		o/p
A	B	x	A(t+1)	B(t+1)	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

2nd



2nd form of state table :-

Present State		Next State				o/p.	
		x=0	x=0	x=1	x=1	x=0	x=1
A	B	A(t+1)	B(t+1)	A(t+1)	B(t+1)	x	y
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

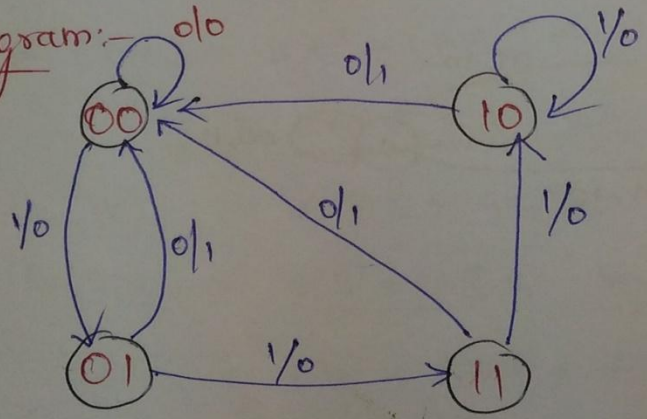
State Diagram Rules

1). State is represented by a circle and transitions b/w states are indicated by directed lines connecting the circuits

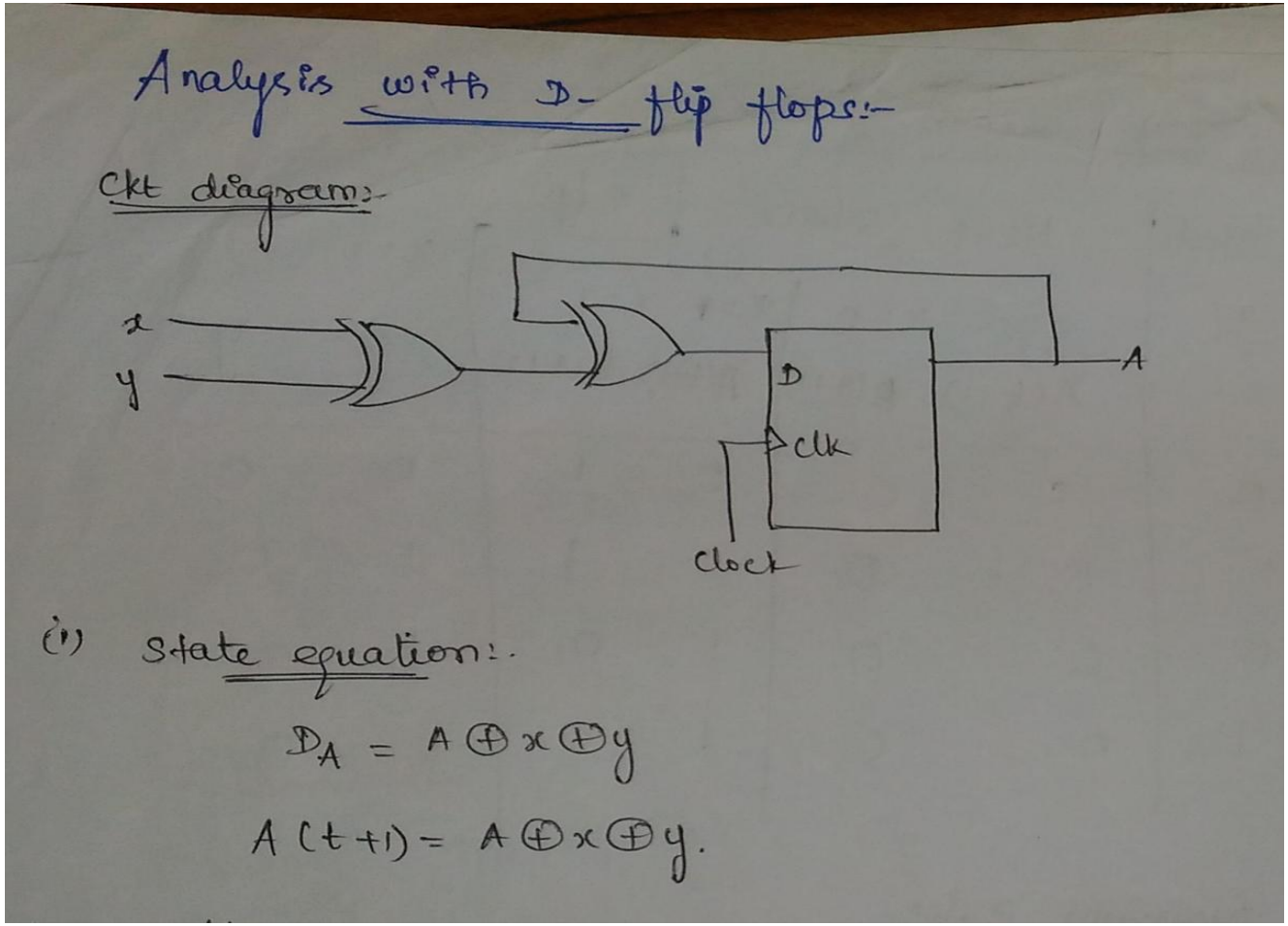
2) The binary no. inside each circle shows the state of flip flop.

3). The i/p value during present state is labelled 1st and the no. after slash gives the o/p.

State diagram:-



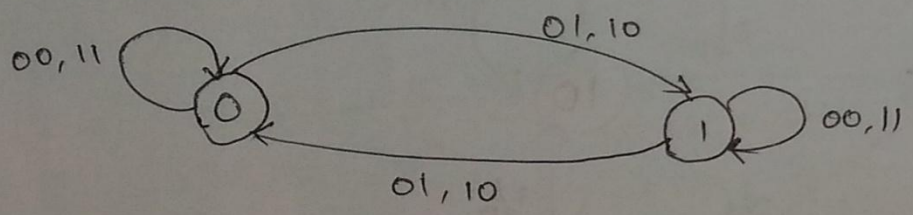
Analysis with D Flip Flop



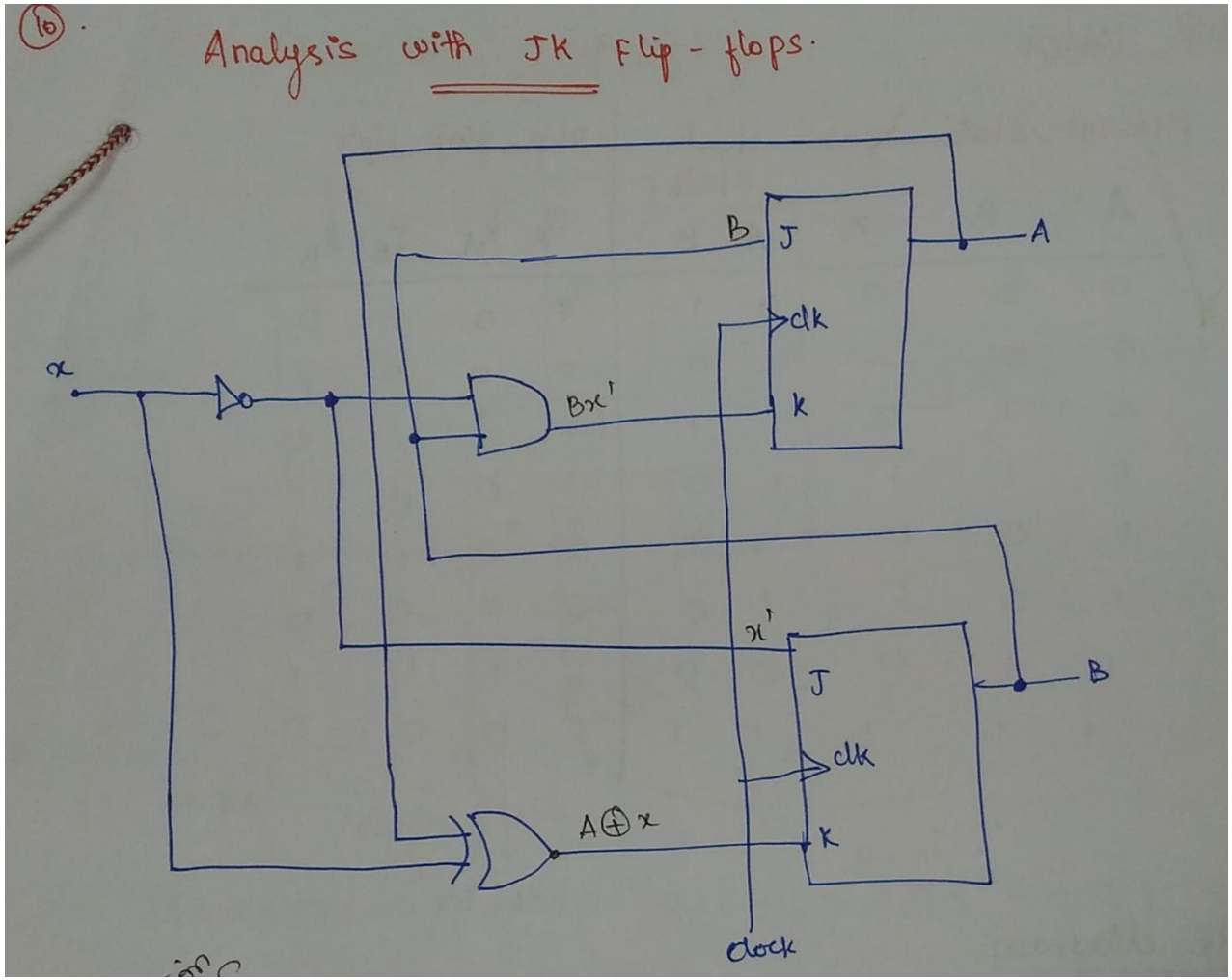
State table:-

Present state A	Inputs x	Inputs y	Next state A
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

State diagram:-



Analysis with JK Flip Flop



i/p equation

$J_A = B$

$J_B = x'$

Clock

$K_A = Bx'$

$K_B = A'x + Ax' = A \oplus x$

State equation:-

$A(t+1) = JA' + K'A$

$B(t+1) = JB' + K'B$

$A(t+1) = BA' + (Bx')'A = A'B + (B'+x)A$

$= BA' + B'A + xA.$

$= A \oplus B + Ax$

$B(t+1) = x'B' + (A \oplus x)'B$

$= B'x' + ABx + A'Bx'.$

State Table:-

present state A	present state B	input x	Next state A	Next state B	Flip-flop i/p's J_A	K_A	J_B	K_B
0	0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	1	1	1	1	1	0
0	1	1	1	0	1	0	0	1
1	0	0	1	1	0	0	1	1
1	0	1	1	0	0	0	0	0
1	1	0	0	0	1	1	1	1
1	1	1	1	1	1	0	0	0

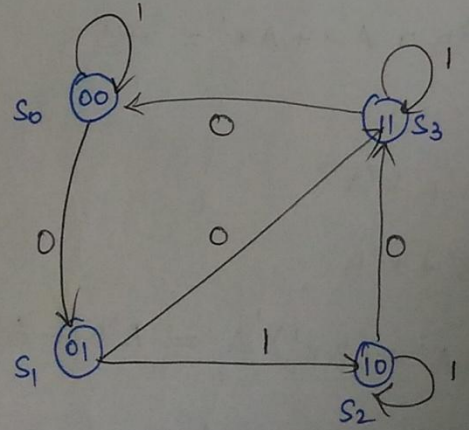
$\therefore J_A = B$

$K_A = Bx'$

x'

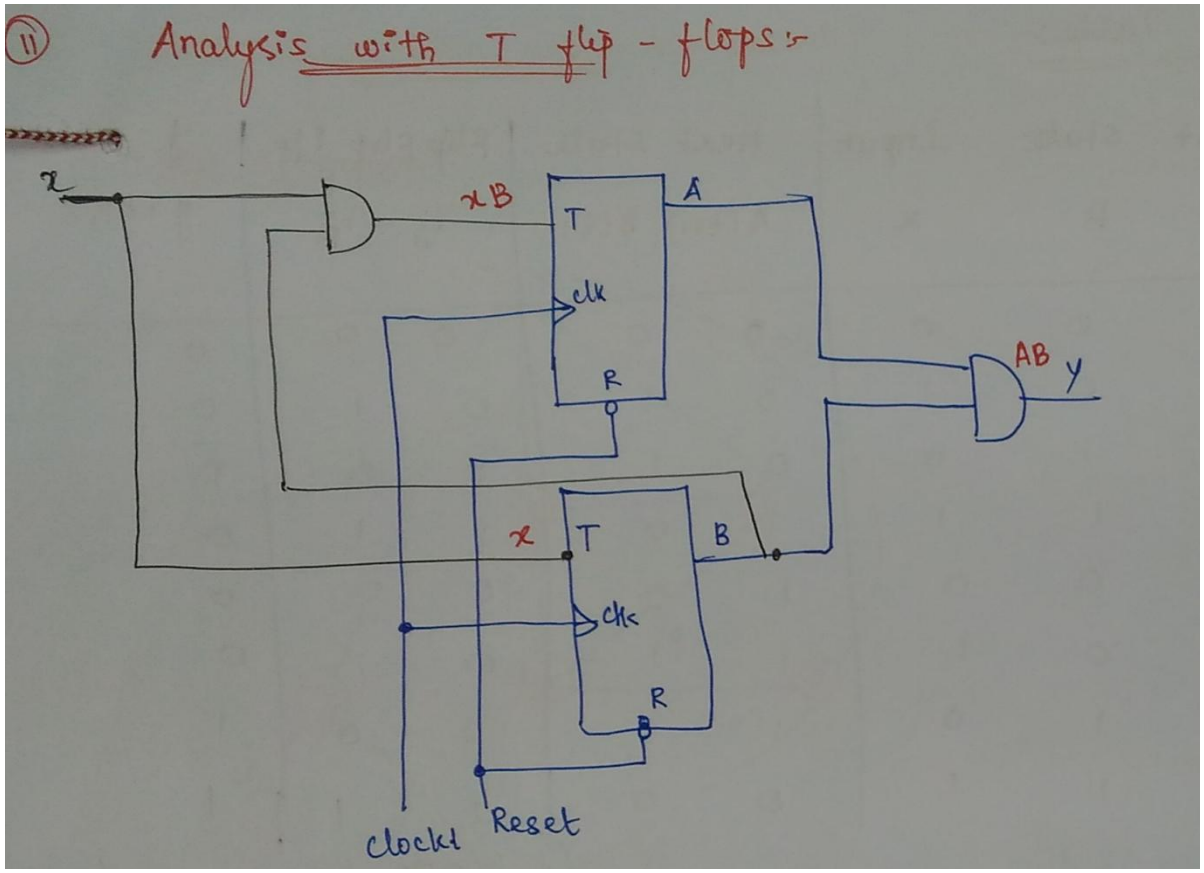
$A \oplus x$

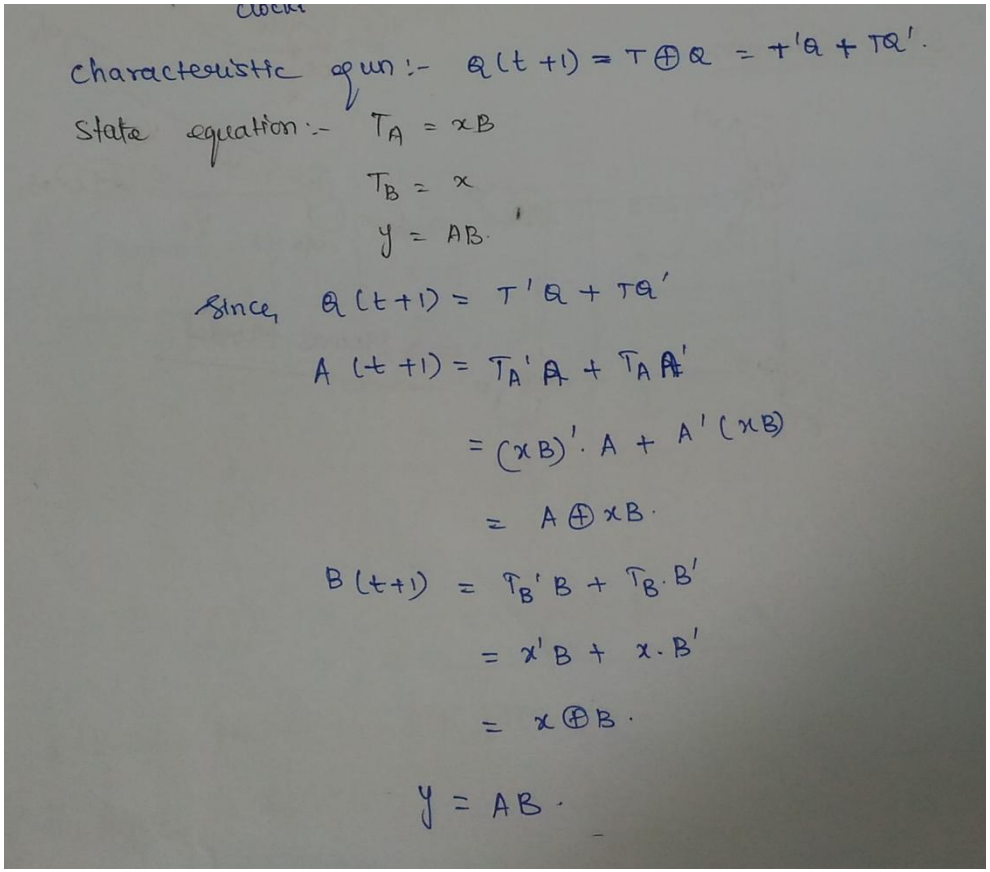
State diagram:-

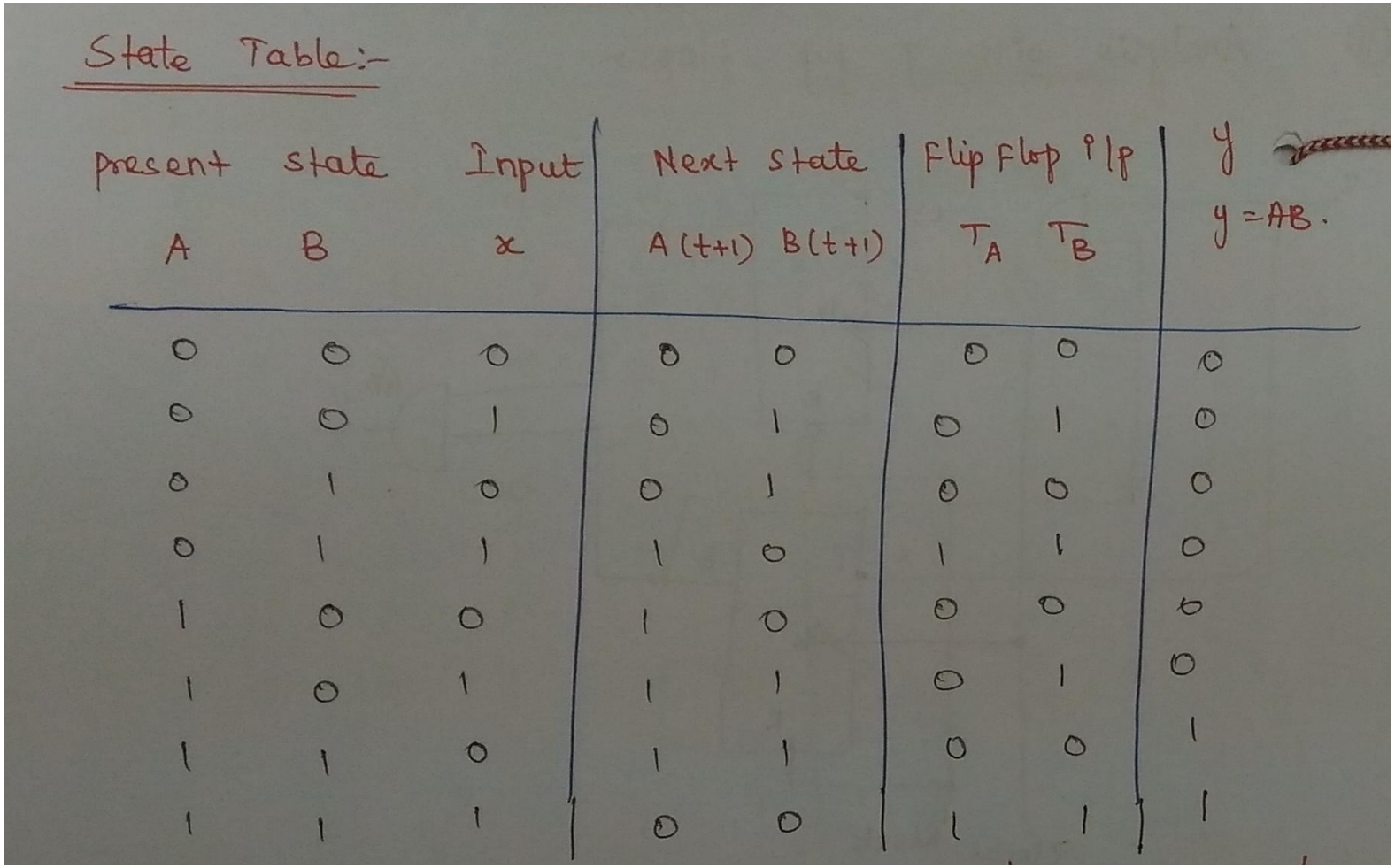


Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Analysis with T flip flop



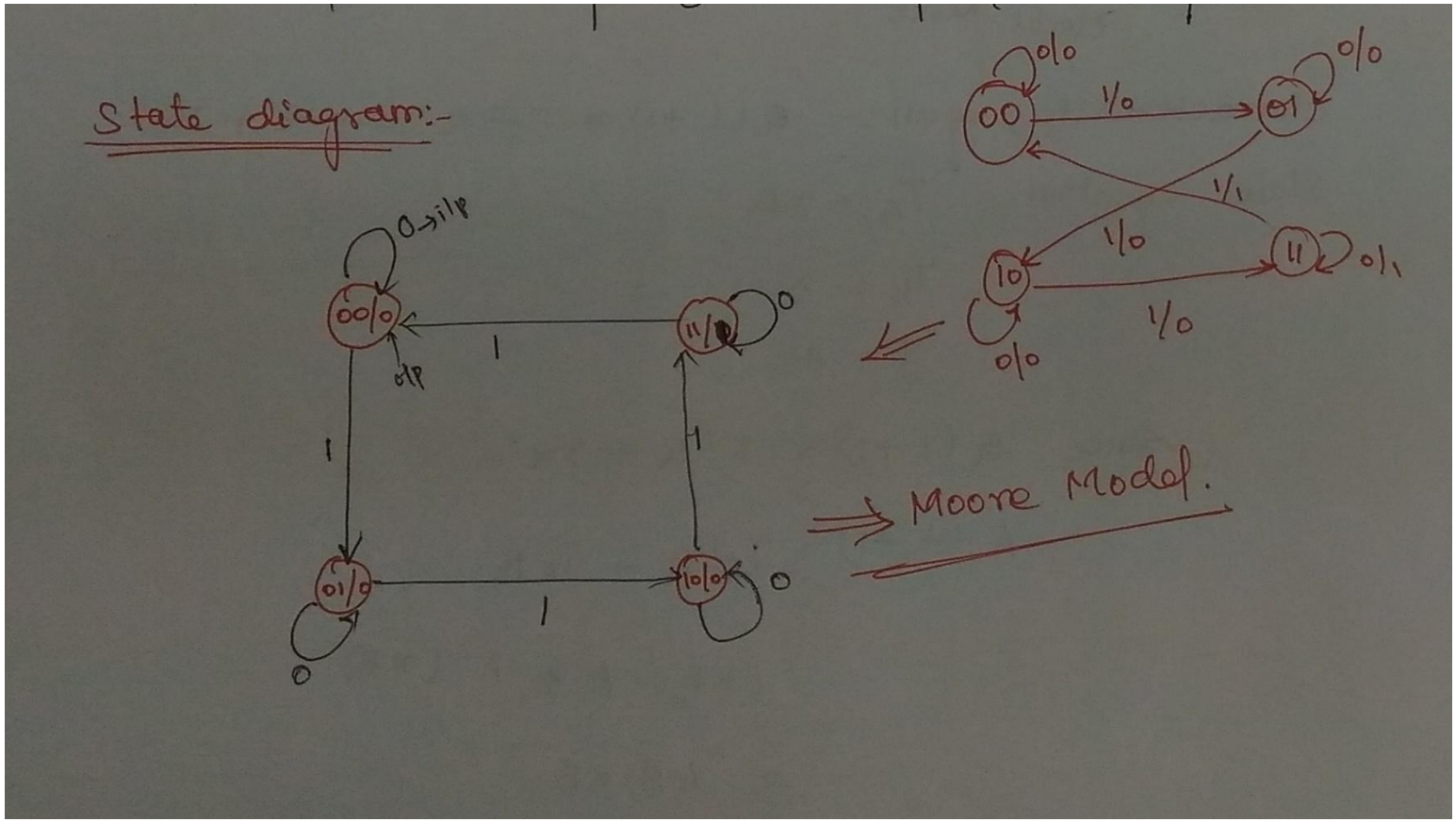

Characteristic eqn :- Q(t+1) = T ⊕ Q = T'Q + TQ'.
State equation :- T_A = xB
T_B = x
y = AB.
Since Q(t+1) = T'Q + TQ'
A(t+1) = T_A'A + T_A A'
= (xB)'. A + A'(xB)
= A ⊕ xB.
B(t+1) = T_B'B + T_B.B'
= x'B + x.B'
= x ⊕ B.
y = AB.



State Table:-

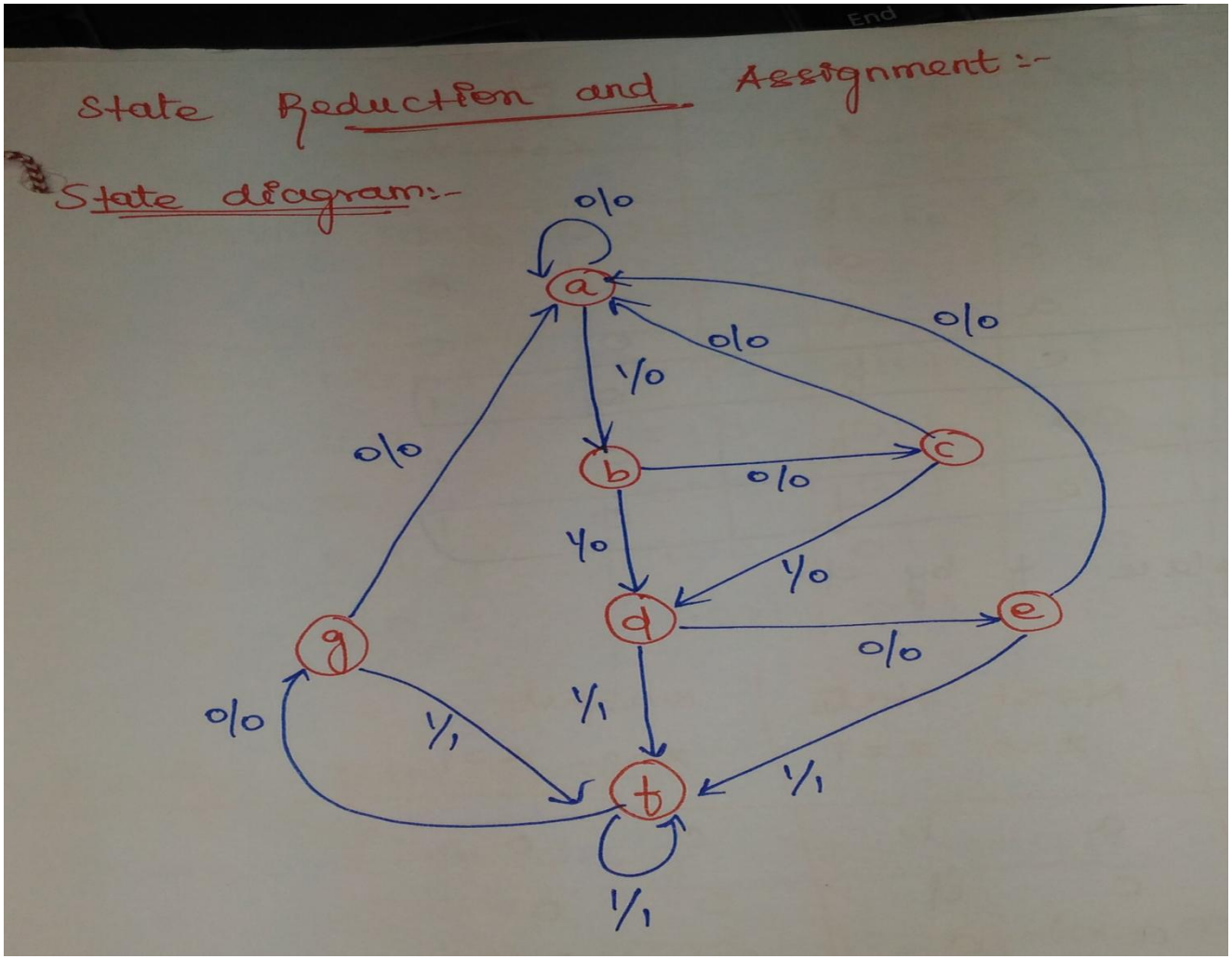
present state A	present state B	Input x	Next state A(t+1)	Next state B(t+1)	Flip Flop i/p T_A	Flip Flop i/p T_B	y $y = AB$
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	1	0	0	1	0	0	0
0	1	1	1	0	1	1	0
1	0	0	1	0	0	0	0
1	0	1	1	1	0	1	0
1	1	0	1	1	0	0	1
1	1	1	0	0	1	1	1

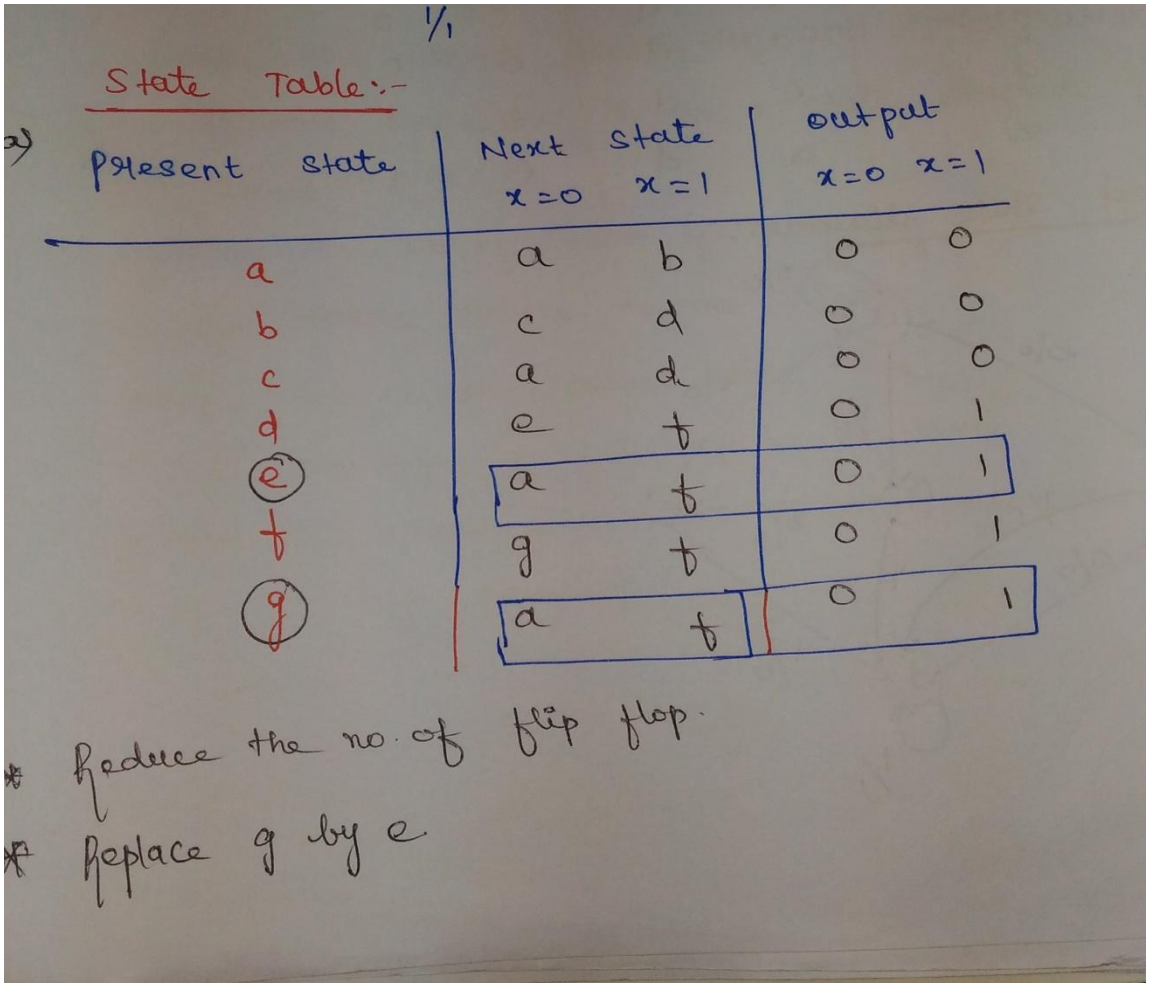
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson


State diagram:-
0→ip
00/0
11/0
01/0
10/0
ip
1
1
1
1
0
0
0
0/0
1/0
00
01
1/1
1/0
10
11
0/1
1/0
0/0
Moore Model.

State Reduction and Assignment

State Reduction and Assignment





State Table :-

Present state	Next state x=0	Next state x=1	output x=0	output x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

* Reduce the no. of flip flop.
* Replace g by e

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Reducing the state table

b)

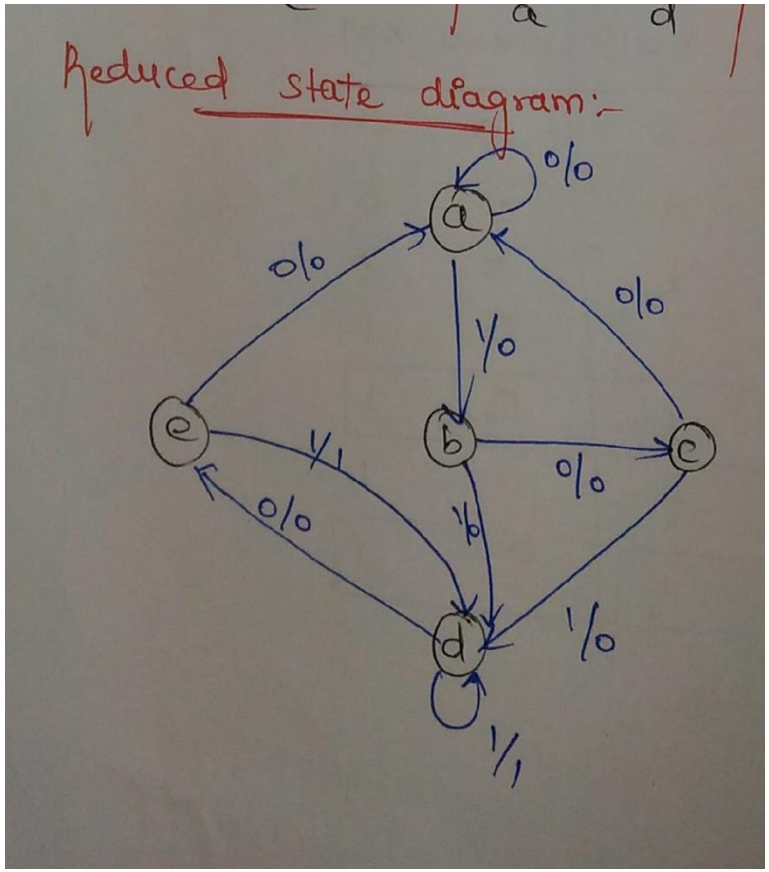
Present State	Next State x=0	Next State x=1	o/p x=0	o/p x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

c). Replace f by d.

c). Replace f by q.

Reduced state table

present state	Next state x=0	Next state x=1	output x=0	output x=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1


Reduced state diagram:-
0/0
a
0/0
0/0
1/0
e
b
c
1/1
0/0
0/0
1/0
d
1/1

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Reduced state table with binary assignment :-

Present state	Next state x=0	Next state x=1	output x=0	output x=1
a → 000	000	001	0	0
b → 001	010	011	0	0
c → 010	000	011	0	0
d → 011	100	011	0	1
e → 100	000	011	0	1

IMPLICATION CHART

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

EXAMPLE 8.8 Reduce the states given in the state Table 8.21 using implication chart.

TABLE 8.21 | State table

Present State	Next State		Output, Z	
	X = 0	X = 1	X = 0	X = 1
A	A	B	0	0
B	C	D	0	0
C	E	F	0	0
D.	B	A	0	1
E	C,	D	0	0
F	B	A	0	1

SET UP IMPLICATION CHART

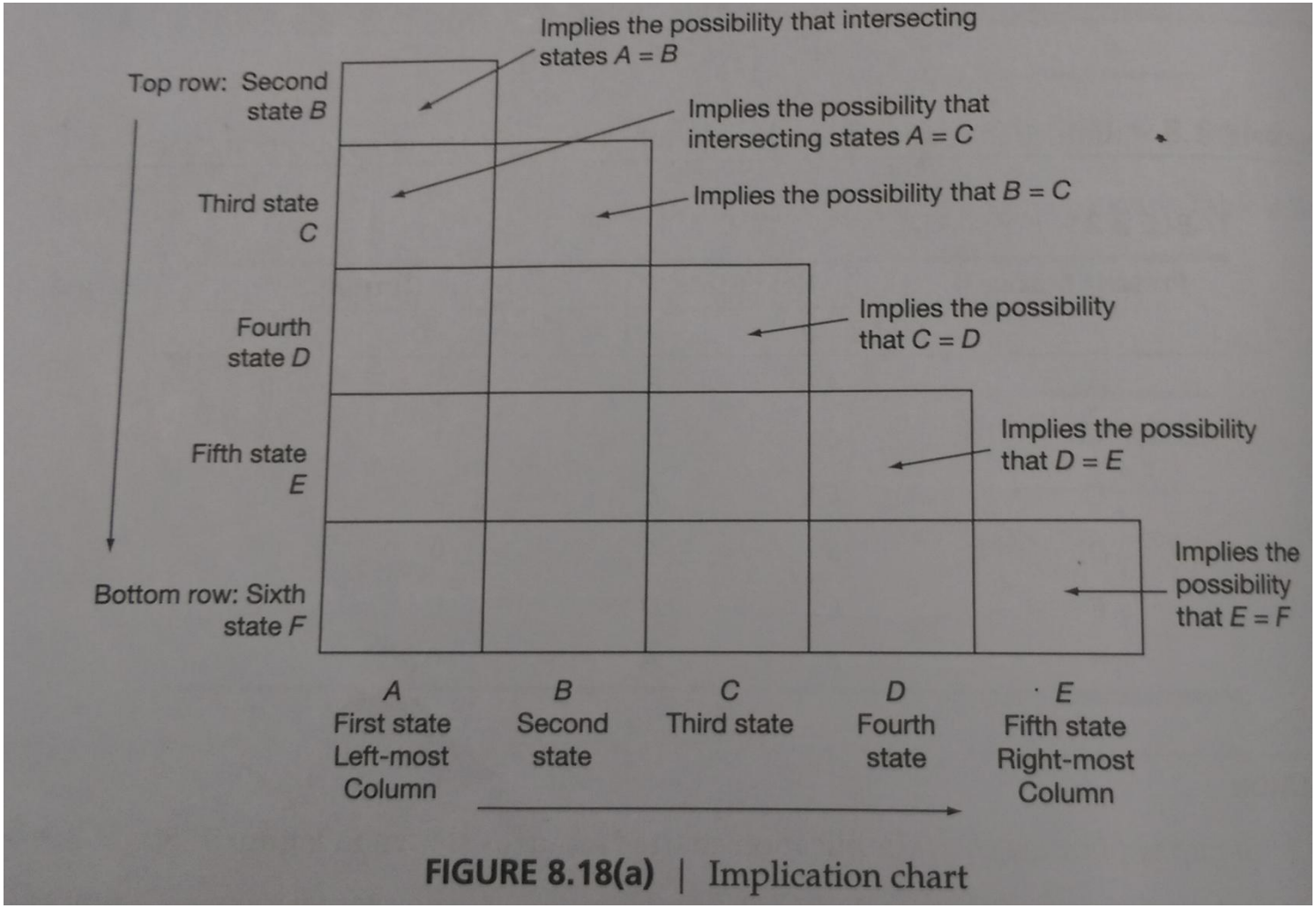


FIGURE 8.18(a) | Implication chart

Fill the square for state pairs which have different outputs

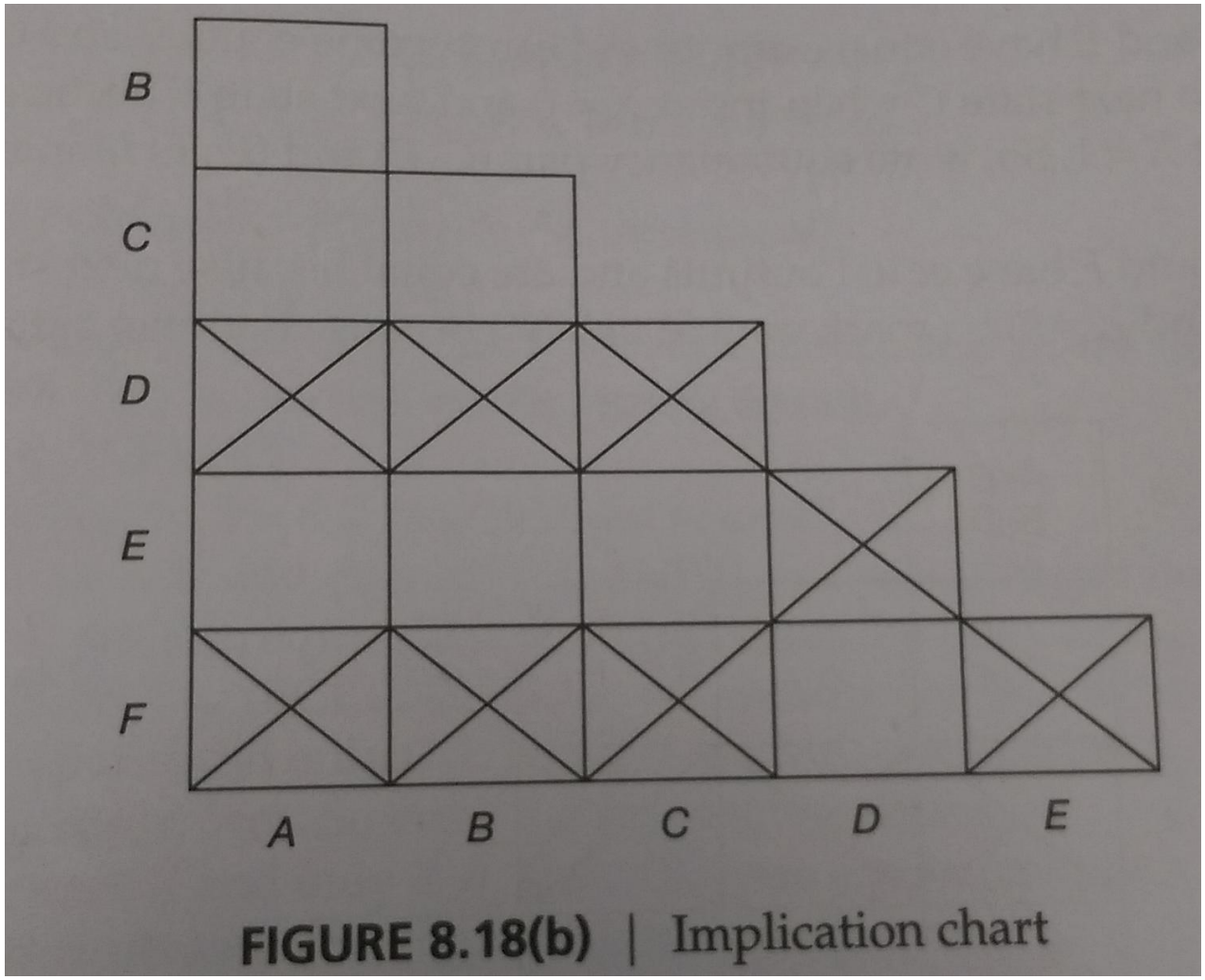


FIGURE 8.18(b) | Implication chart

FILL THE SQUARES FOR STATE PAIRS WHICH MAY HAVE SAME OUTPUTS

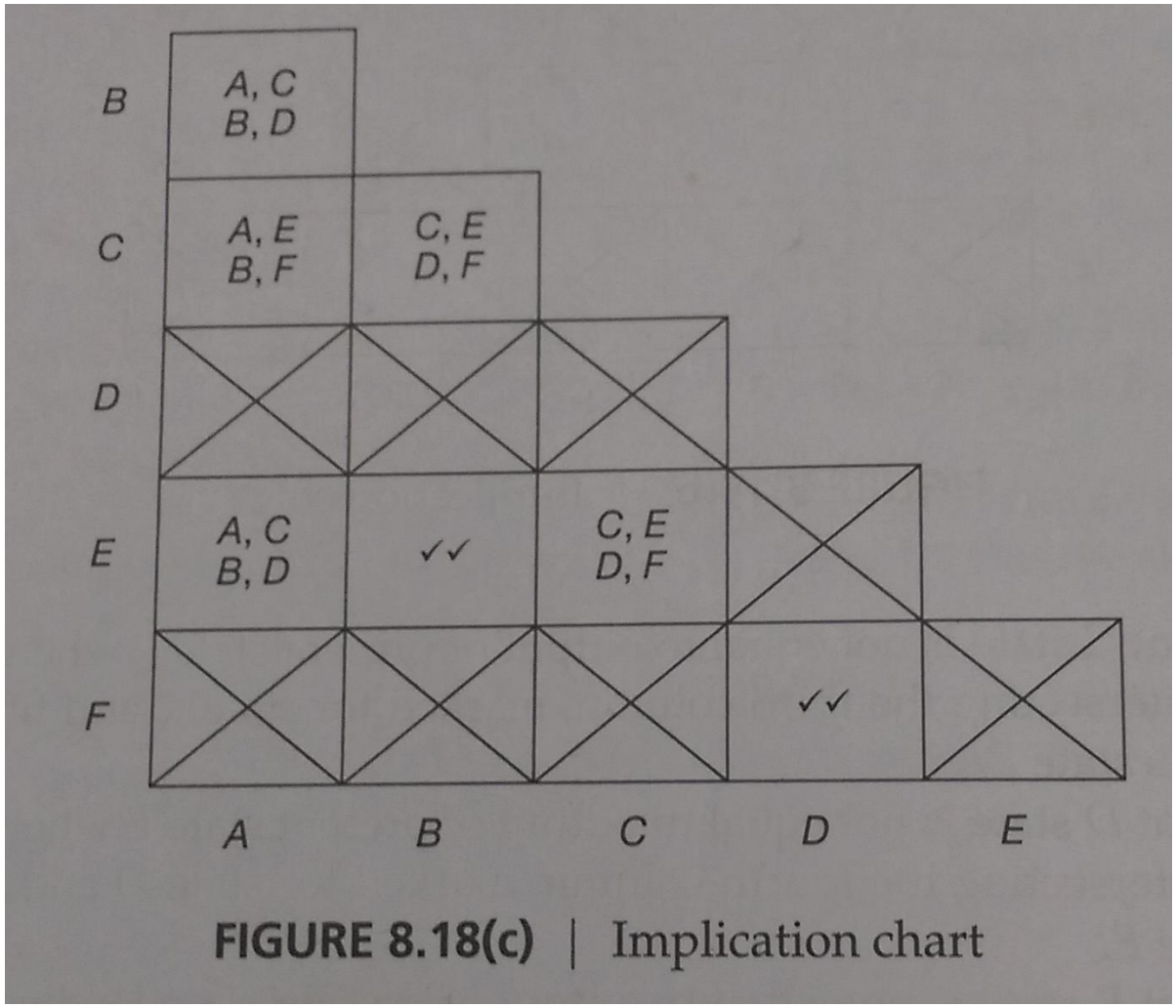


FIGURE 8.18(c) | Implication chart

APPLY EQUIVALENCY TEST OF EACH SQUARE

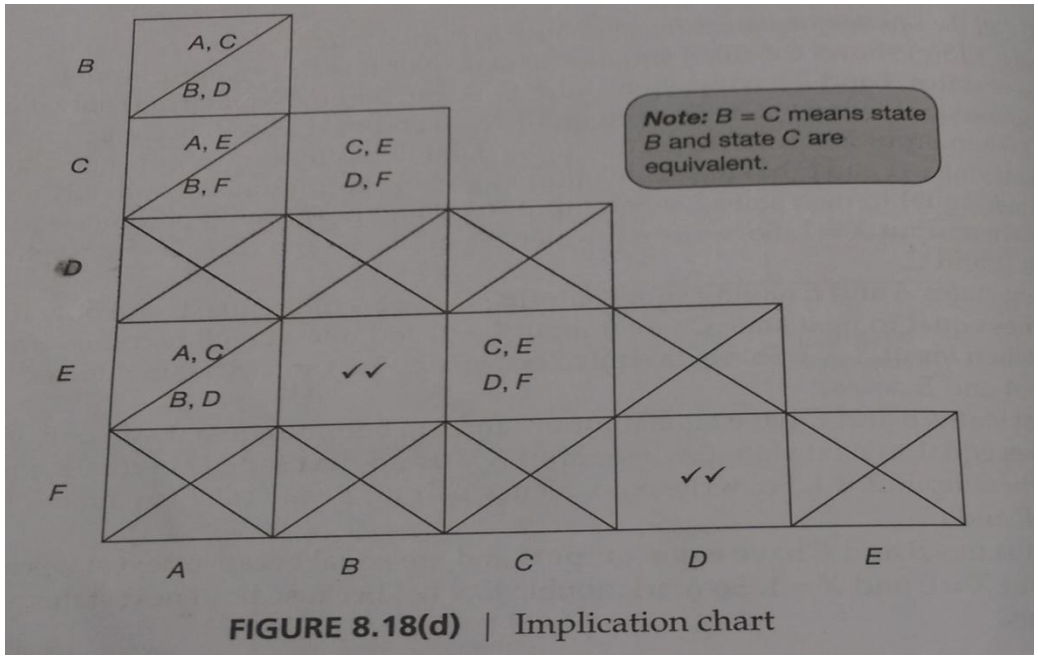


FIGURE 8.18(d) | Implication chart

B=C=E

C=E

D=F

TABLE 8.22 | Reduced state table

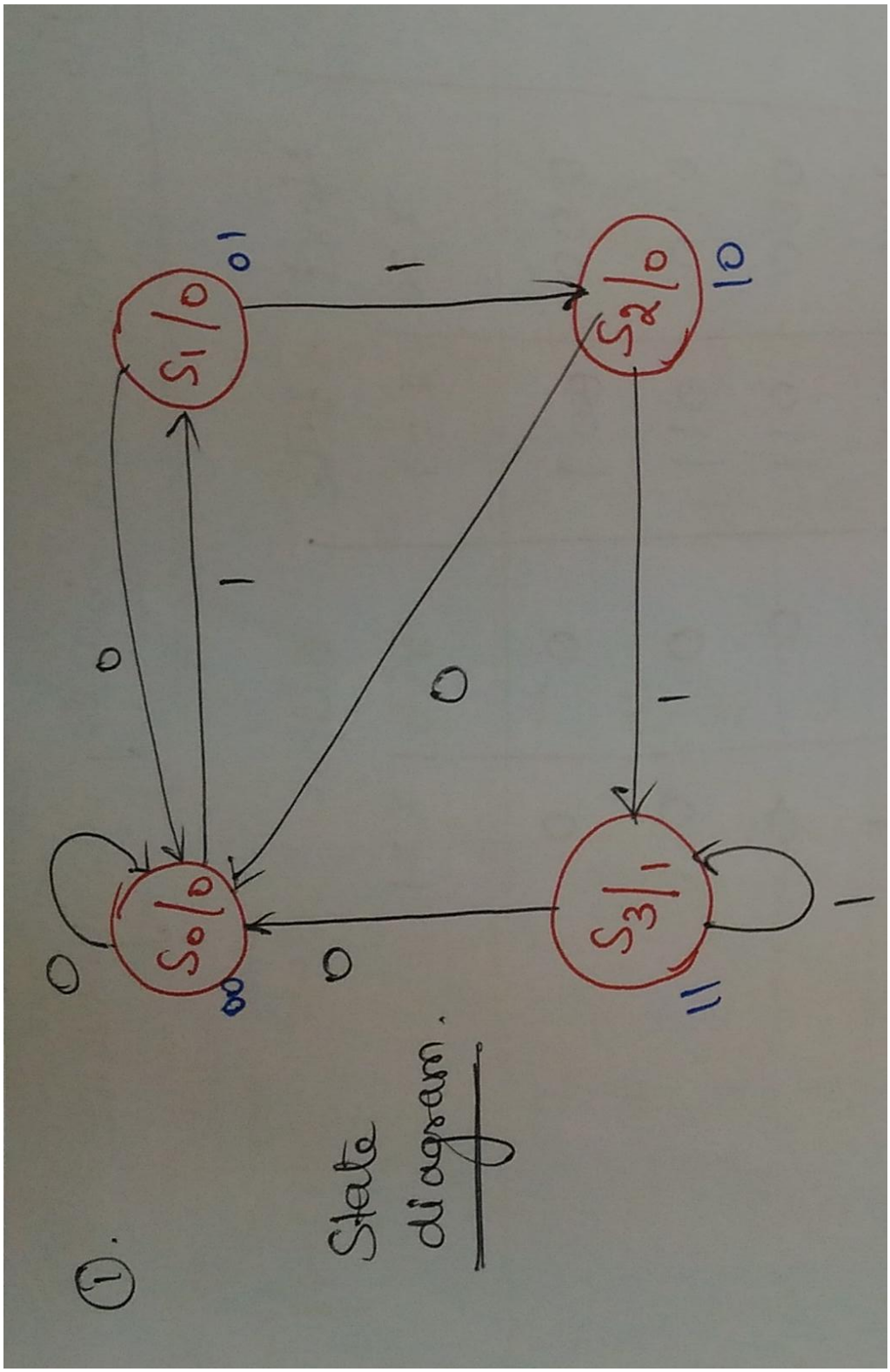
Present State	Next State		Output, Z	
	$X = 0$	$X = 1$	$X = 0$	$X = 1$
A	A	B	0	0
B	B	D	0	0
D	B	A	0	1

Design procedure

Procedure to design synchronous sequential circuit

1. From the word description and specifications of the desired operation, derive a state diagram for the circuit.
2. Reduce the number of states if necessary
3. Assign binary values to the state
4. Obtain the binary coded stable table
5. Choose the type of flip flop to be used
6. Derive the simplified flip flop input and output equations
7. Draw the logic diagram

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson


S1/0
01
S2/0
10
S0/0
00
S3/1
11
1
0
State diagram.

State table:-

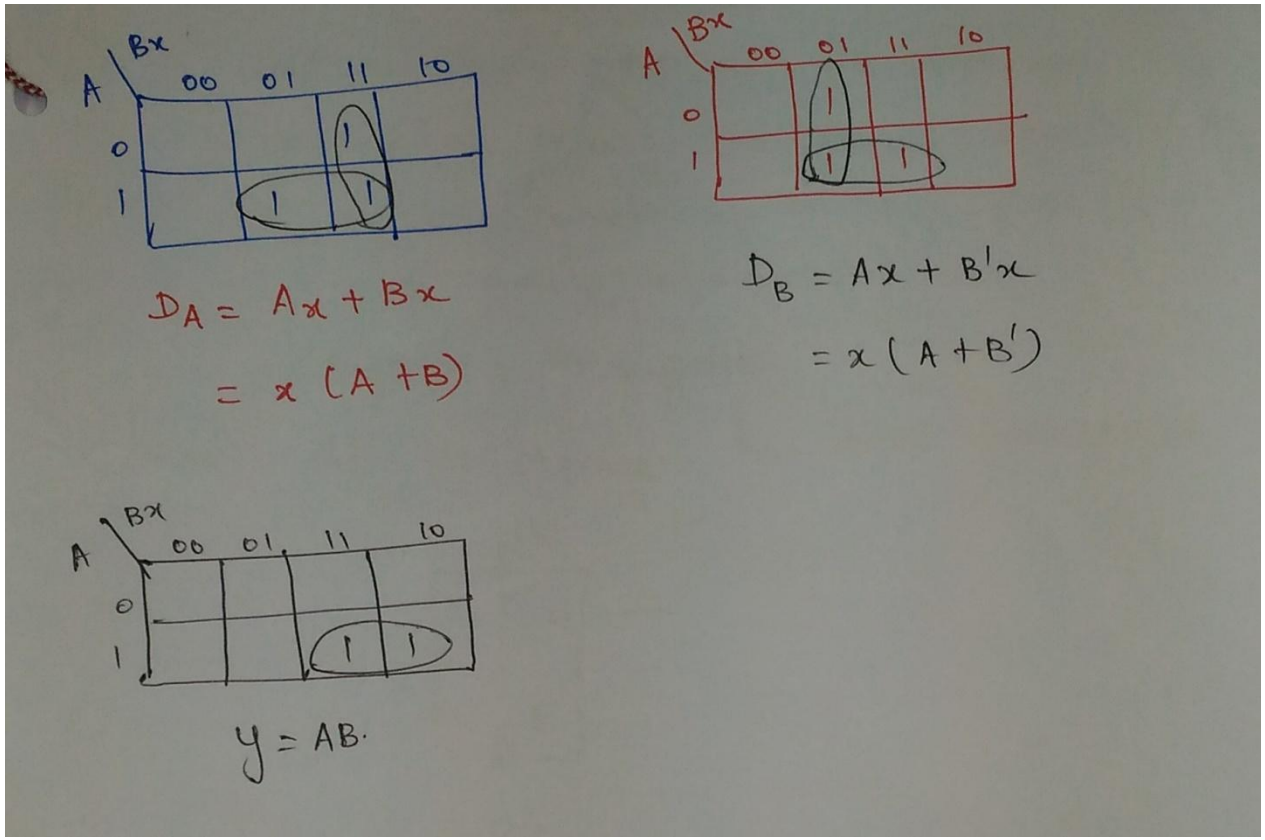
Present State A	Present State B	i/p x	Next State A (t+1)	Next State B (t+1)	o/p y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	1

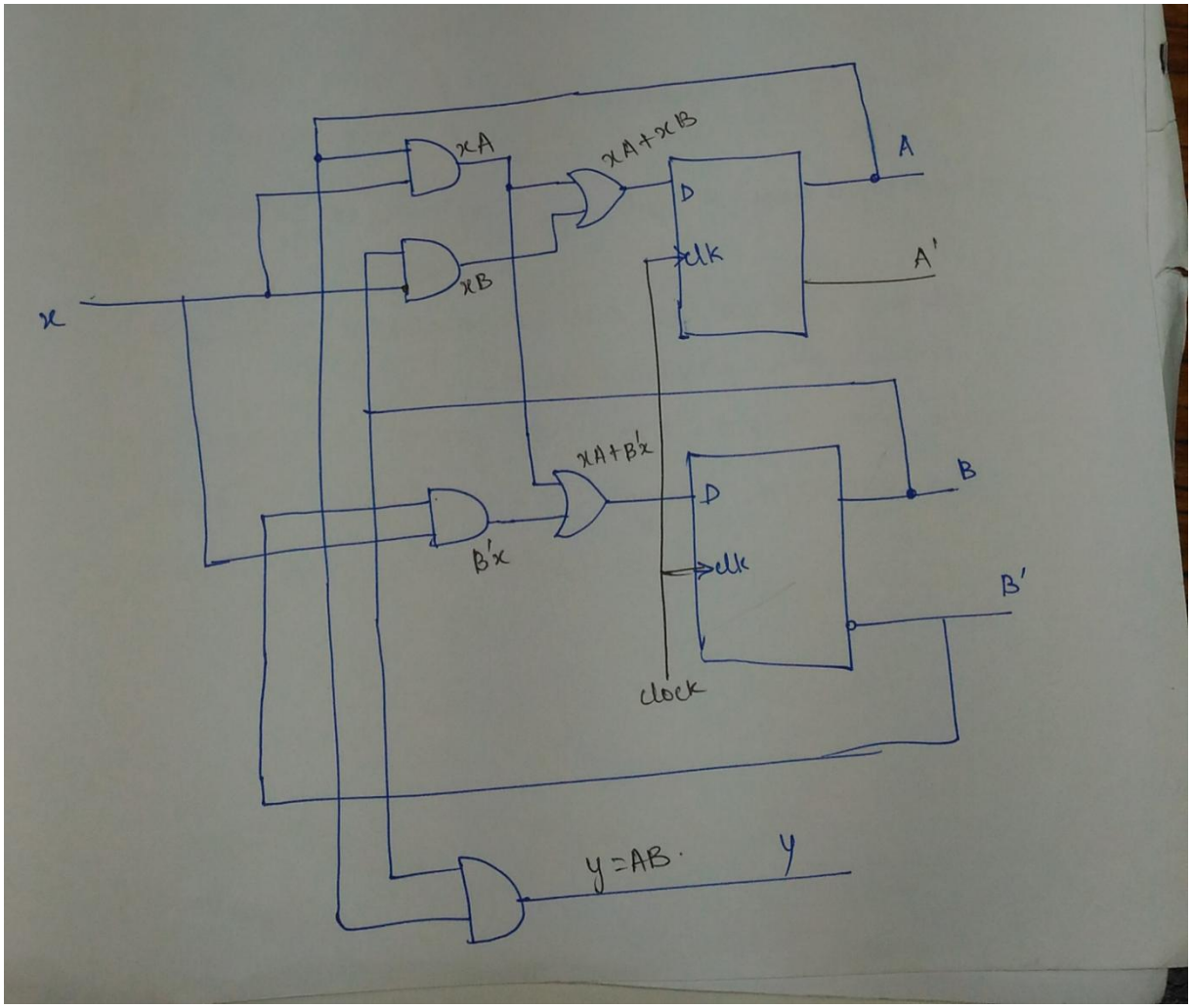
$A(t+1) = D_A(A, B, x) = \Sigma(3, 5, 7)$

$B(t+1) = D_B(A, B, x) = \Sigma(1, 5, 7)$

$y(A, B, x) = \Sigma(6, 7).$

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson


Bx
A
00 01 11 10
0
1
1 1 1
$D_A = Ax + Bx$
$= x(A + B)$
Bx
A
00 01 11 10
0
1
1 1 1
$D_B = Ax + B'x$
$= x(A + B')$
Bx
A
00 01 11 10
0
1
1 1
$y = AB$



Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

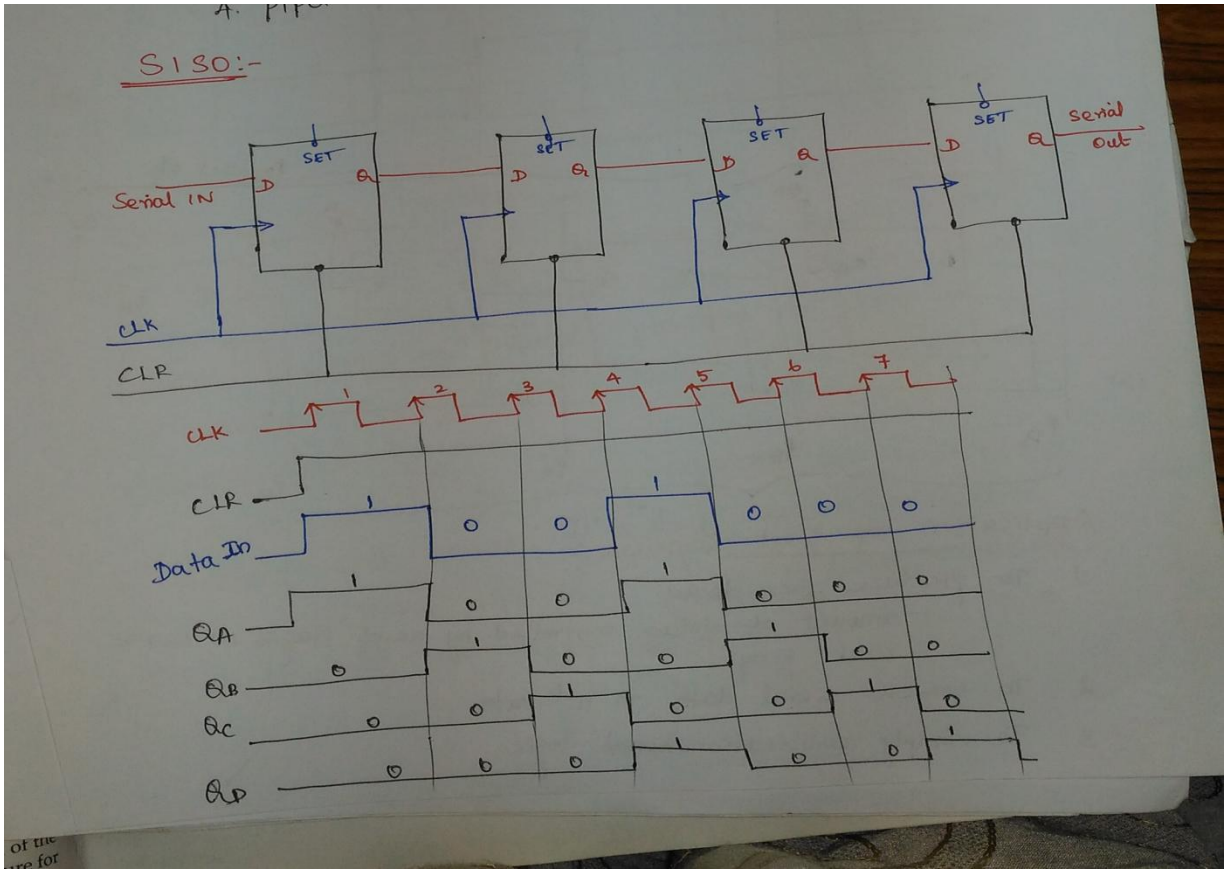
Shift Registers

Shift Registers

- One flip-flop can store one-bit of information. In order to store multiple bits of information, we require multiple flip-flops.
- The group of flip-flops, which are used to hold (store) the binary data is known as **register**.
- If the register is capable of shifting bits either towards right hand side or towards left hand side is known as **shift register**.
- An 'N' bit shift register contains 'N' flip-flops.
- Four types of shift registers based on applying inputs and accessing of outputs.
  - Serial In - Serial Out shift register
  - Serial In - Parallel Out shift register
  - Parallel In - Serial Out shift register
  - Parallel In - Parallel Out shift register

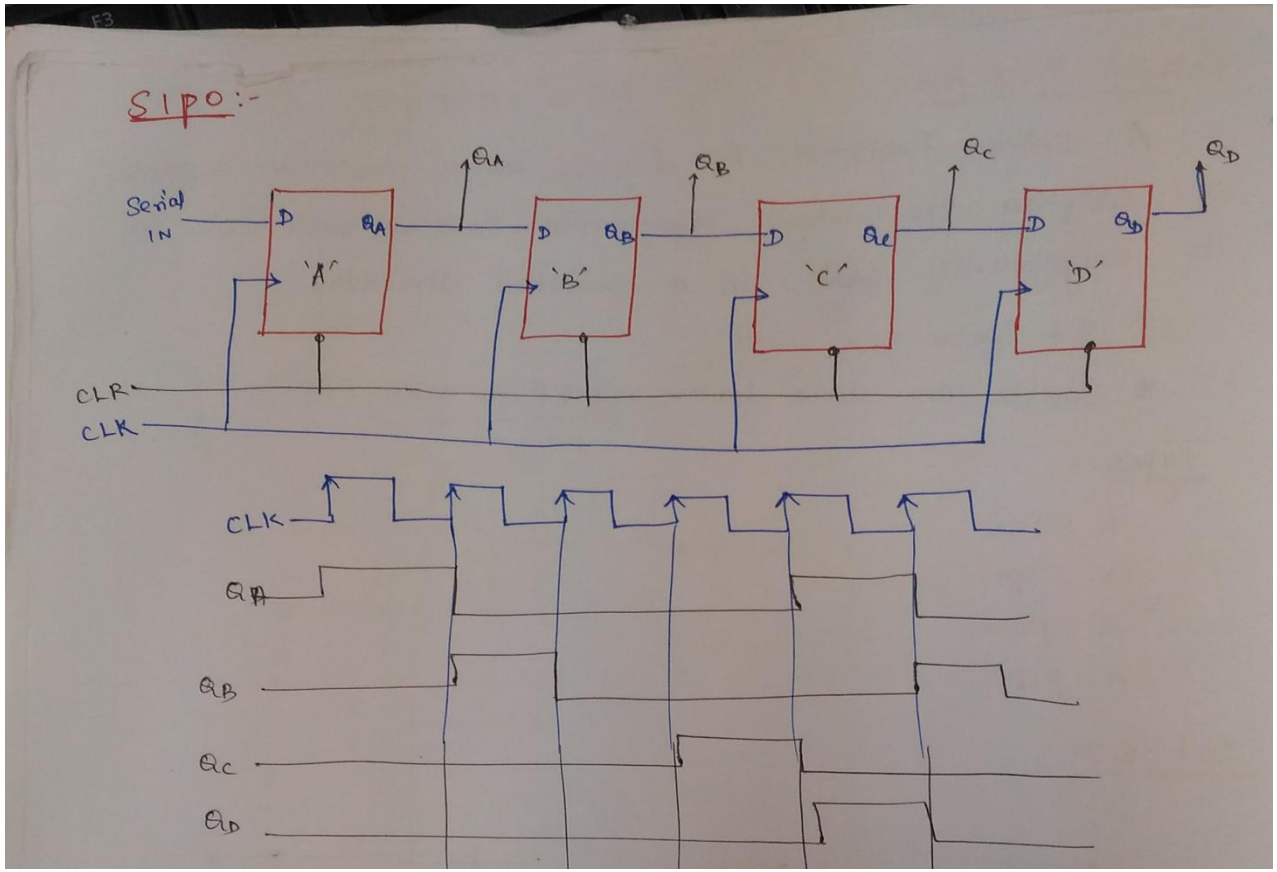
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

SISO



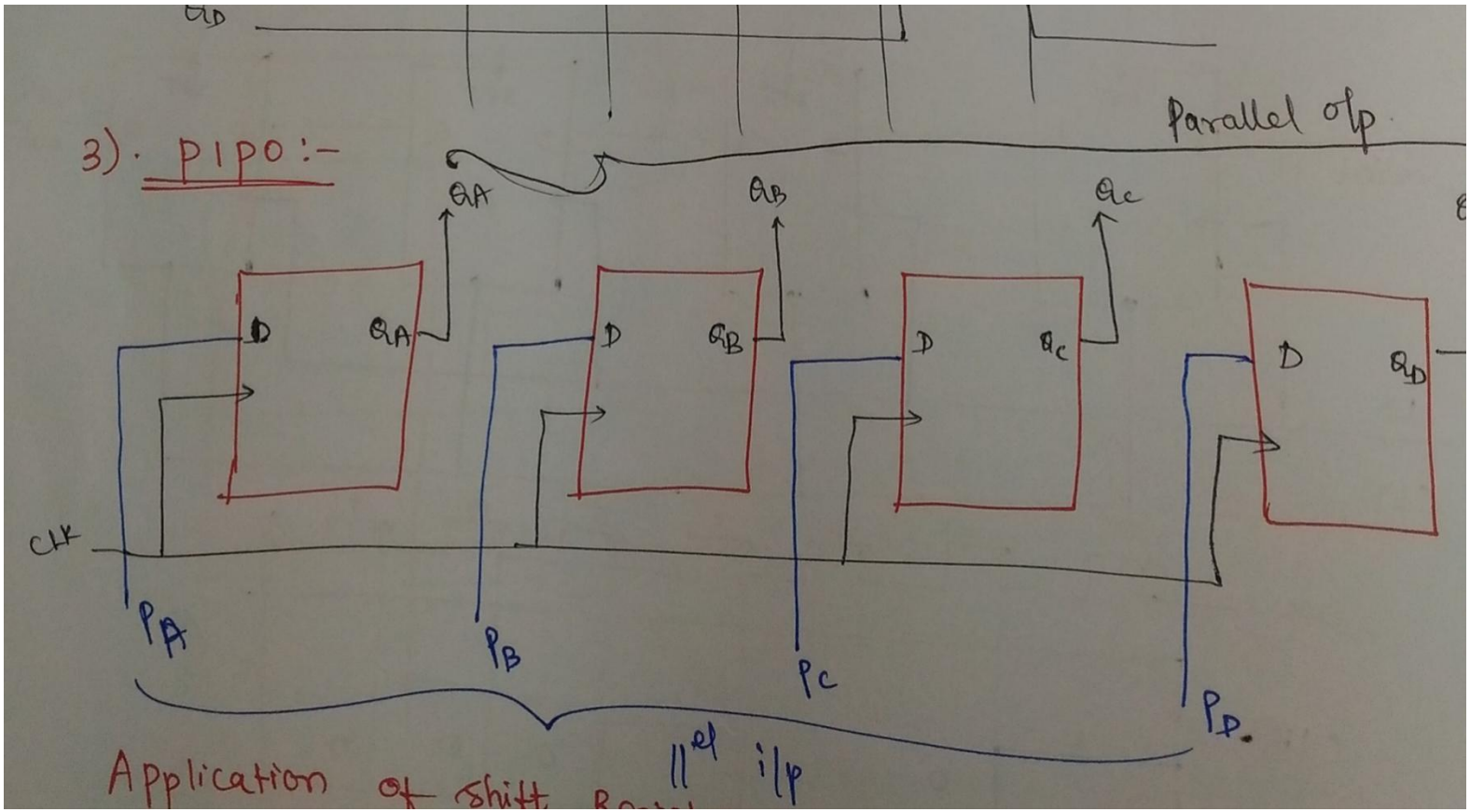
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

SIPO



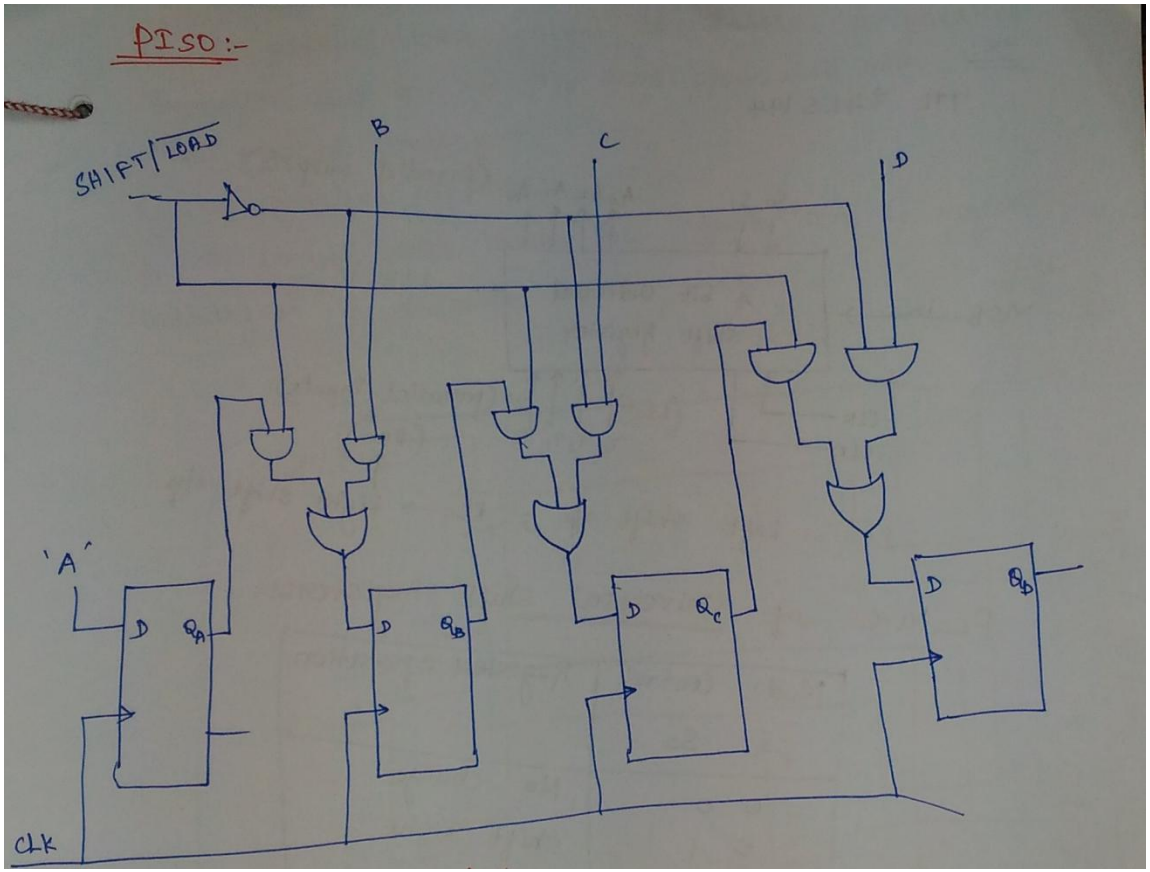
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

PIPO



Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

PISO



UNIVERSAL SHIFT REGISTER

- ***Universal Shift Register*** is a register which can be configured to load and/or retrieve the data in any mode (either serial or parallel) by shifting it either towards right or towards left.
- In other words, a combined design of unidirectional (either right- or left-shift of data bits as in case of SISO, SIPO, PISO, PIPO) and bidirectional shift register along with parallel load provision is referred to as **universal shift register**.
- Such a shift register capable of storing n input bits .

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

Features of Universal Shift Registers:-

Mode Control S_1	Mode Control S_0	Register operation
0	0	No change
0	1	shift Right
1	0	shift left
1	1	parallel load.

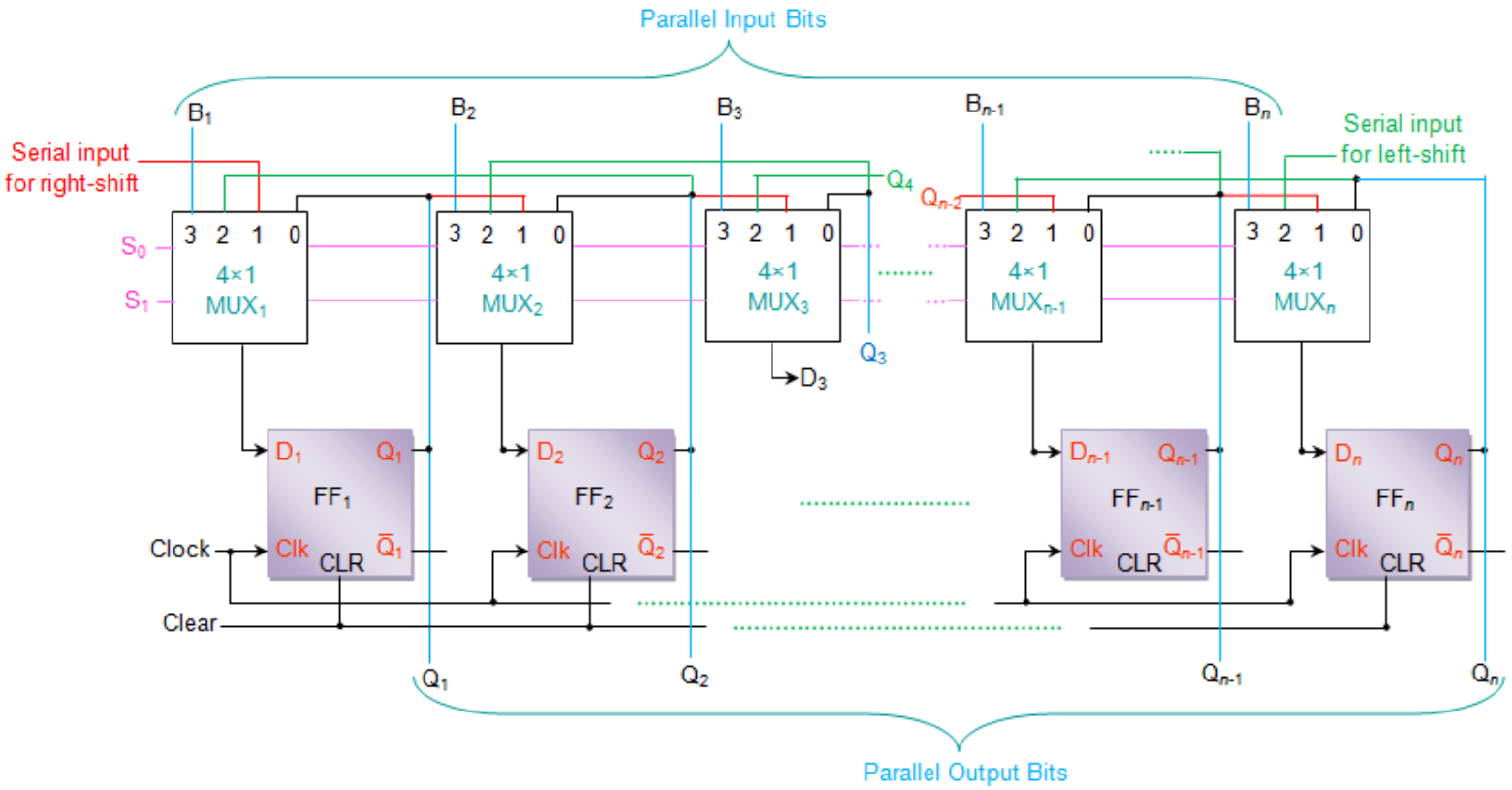


Figure 1 *n*-bit Universal Shift Register

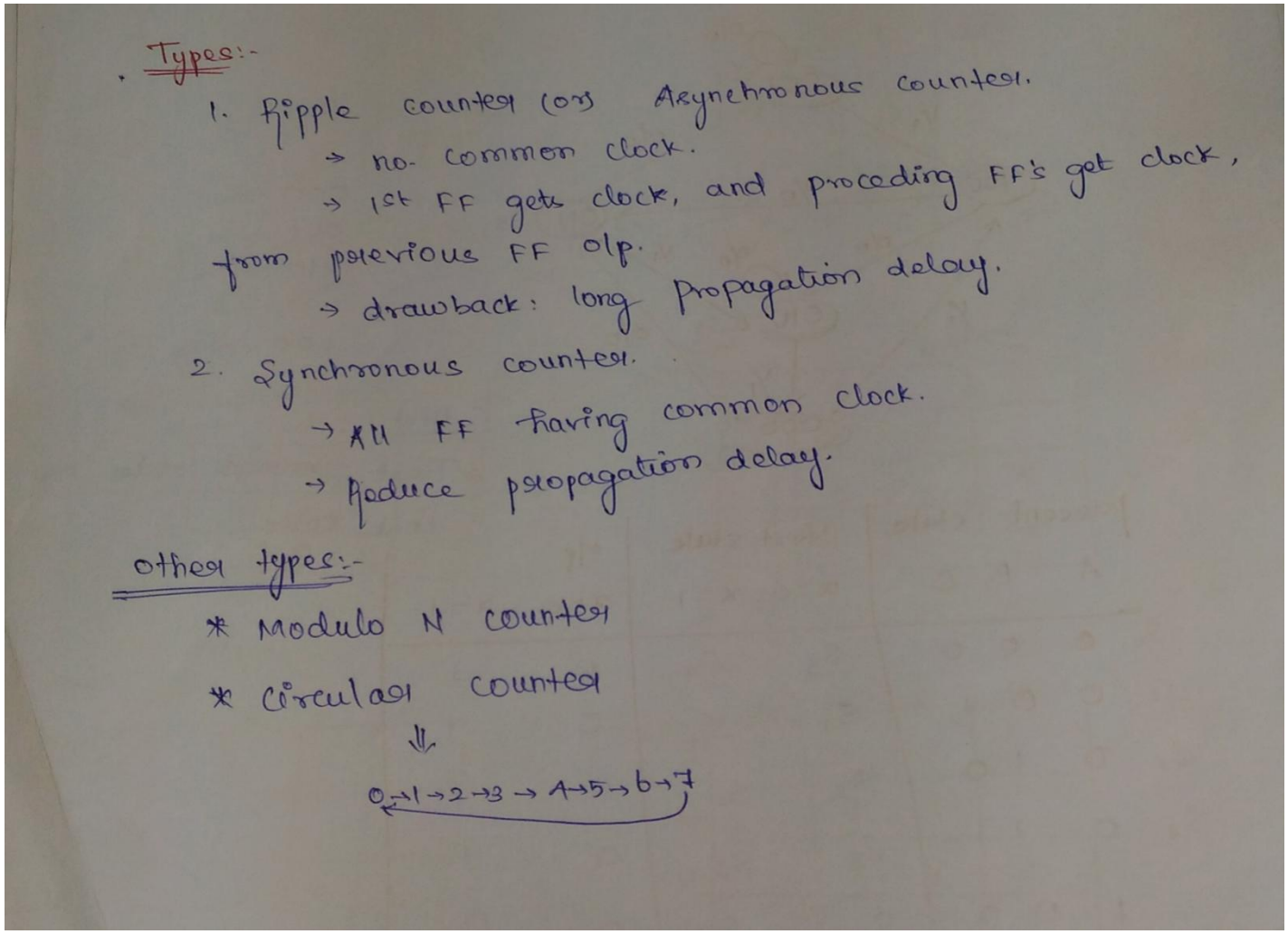
Applications of Shift Registers

- **Delay Lines of Shift Registers**
- Shift registers of Serial In Serial Out (SISO) kind can be used to delay the digital signals by a definite period time.
- The time delay introduced by the n-bit shift register is equal to n times the inverse of the clock frequency driving the shift register.
- Very long shift registers of this kind can be used as delay line memory in computer systems to store the temporary data.
- Further, multiple bidirectional shift registers connected in parallel can be used as a stack.
- To convert serial data into parallel data
- To simplify the combinational logic

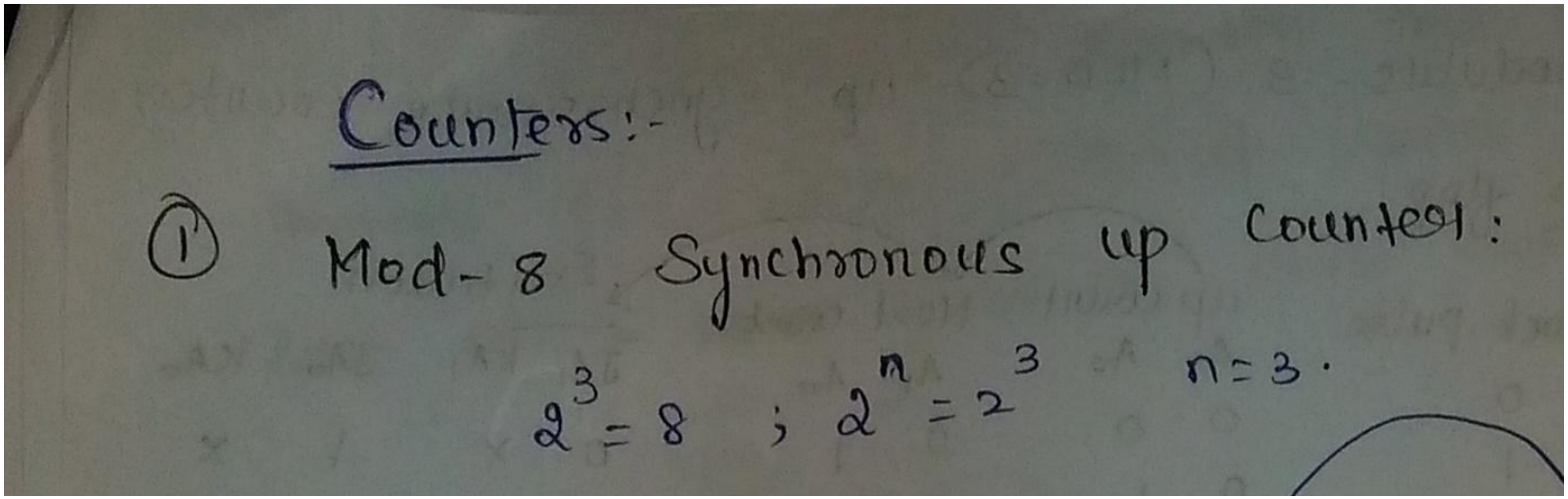
Counters

- Counter is an electronic circuit used to count the number of times an event occurs.
- In digital electronics counters are constructed using series of flip-flops.

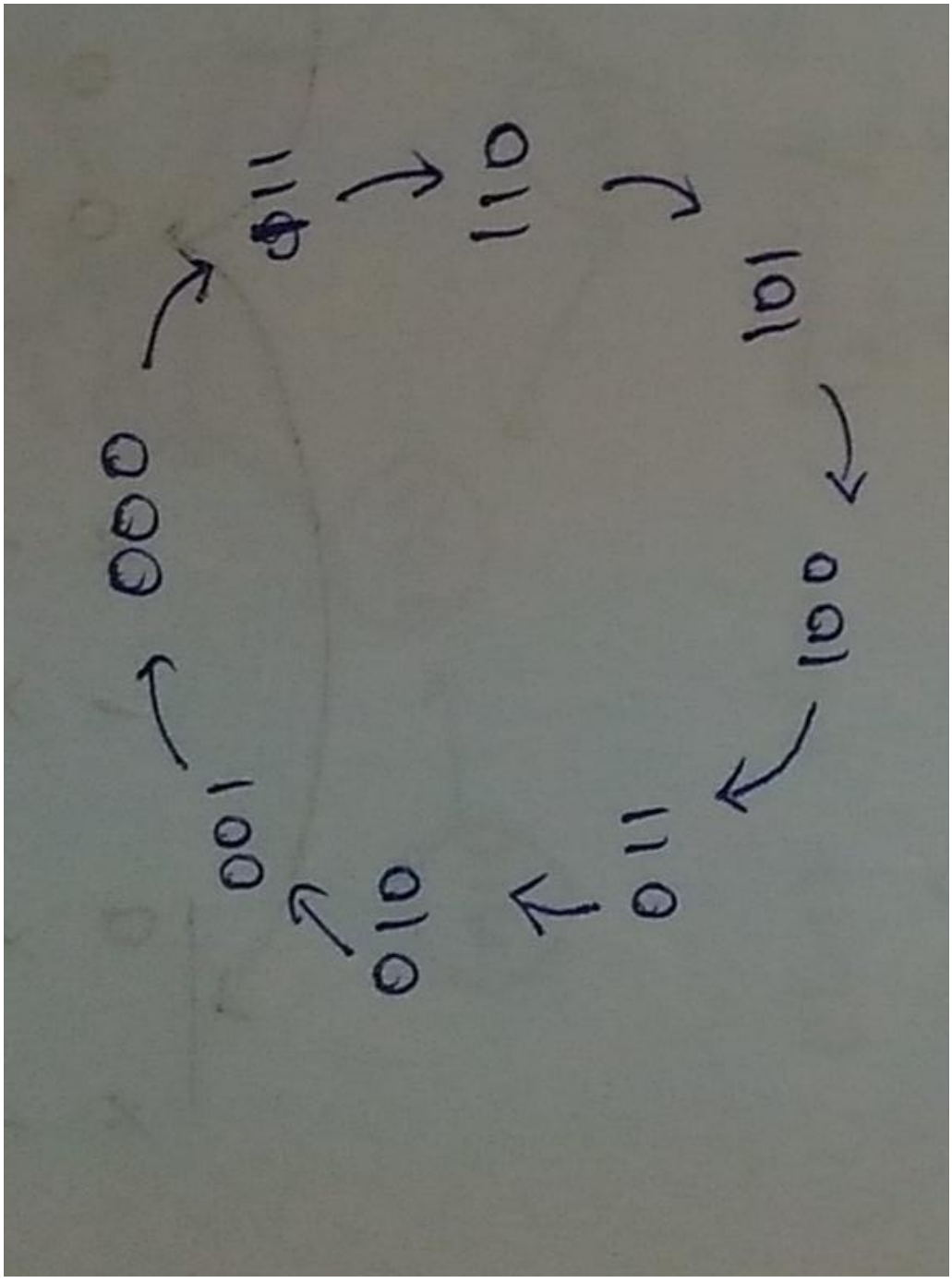
Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

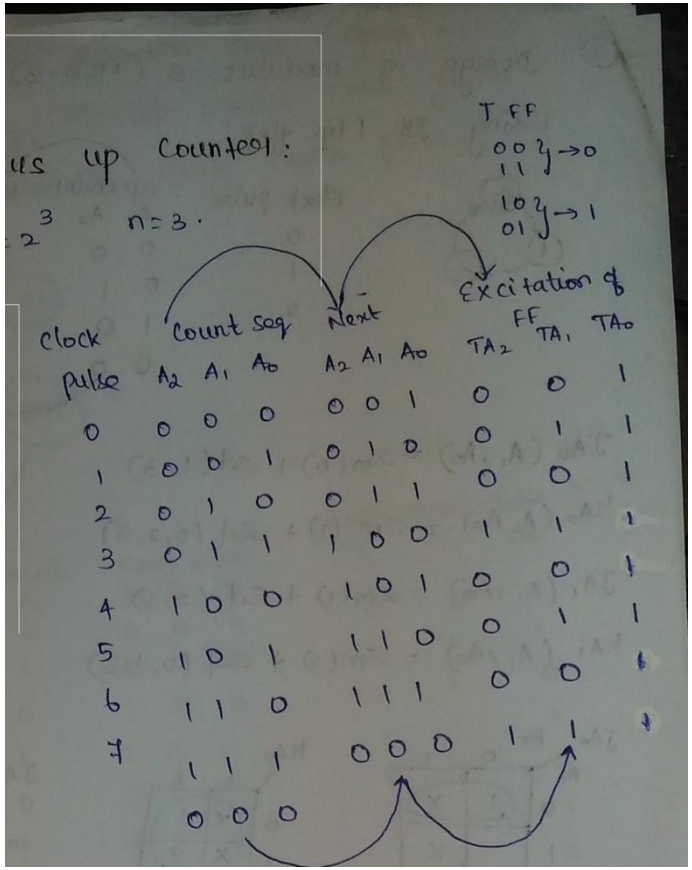

Types:-
1. Ripple counter (or) Asynchronous counter.
→ no common clock.
→ 1st FF gets clock, and proceding FF's get clock, from previous FF o/p.
→ drawback: long Propagation delay.
2. Synchronous counter.
→ All FF having common clock.
→ Reduce propagation delay.
other types:-
* Modulo N counter
* Circular counter
0→1→2→3→4→5→6→7

MOD-8 UP COUNTER



Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

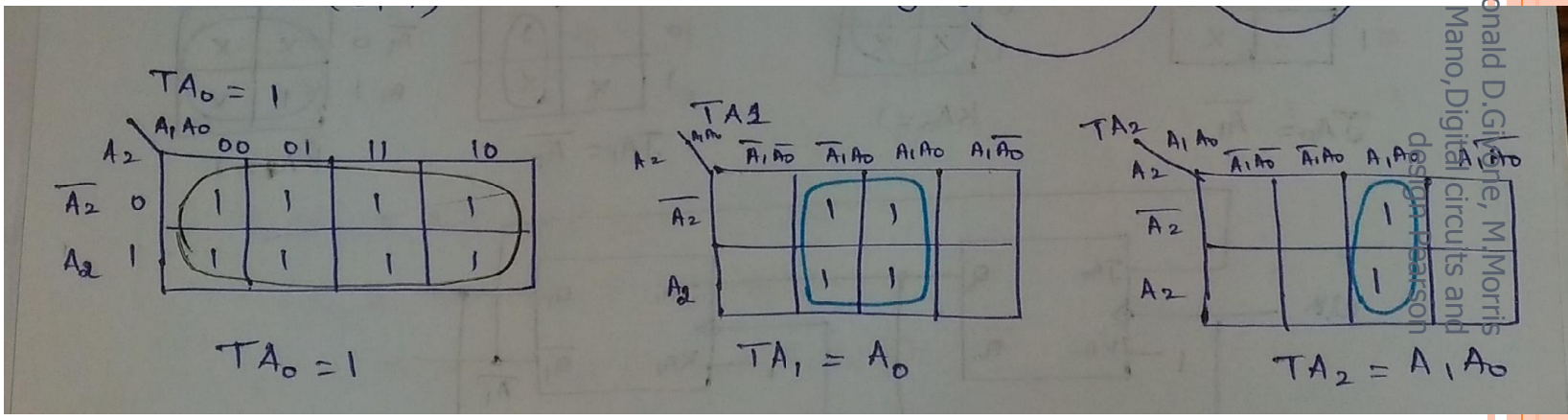

000
001
010
011
100
101
110
111

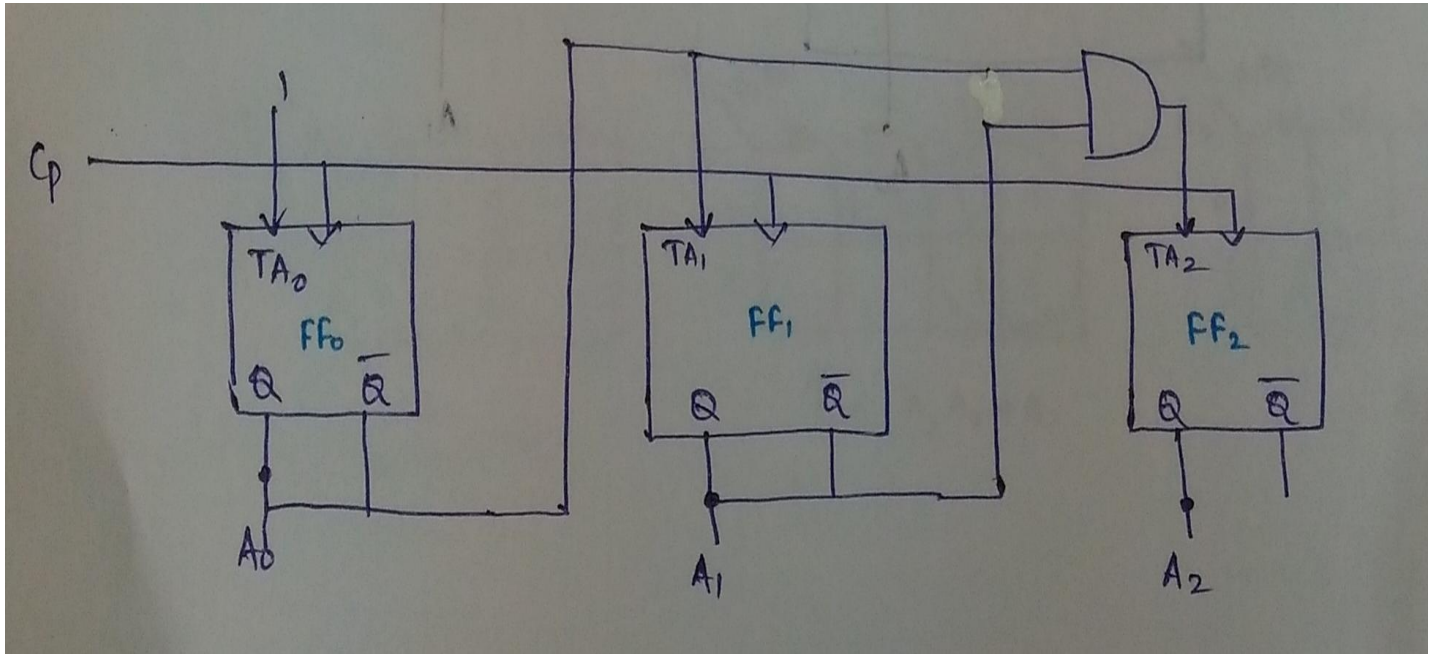


$$TA_0 = \Sigma m(0,1,2,3,4,5,6,7)$$

$$TA_1 = \Sigma m(1,3,5,7)$$

$$TA_2 = \Sigma m(3,7)$$





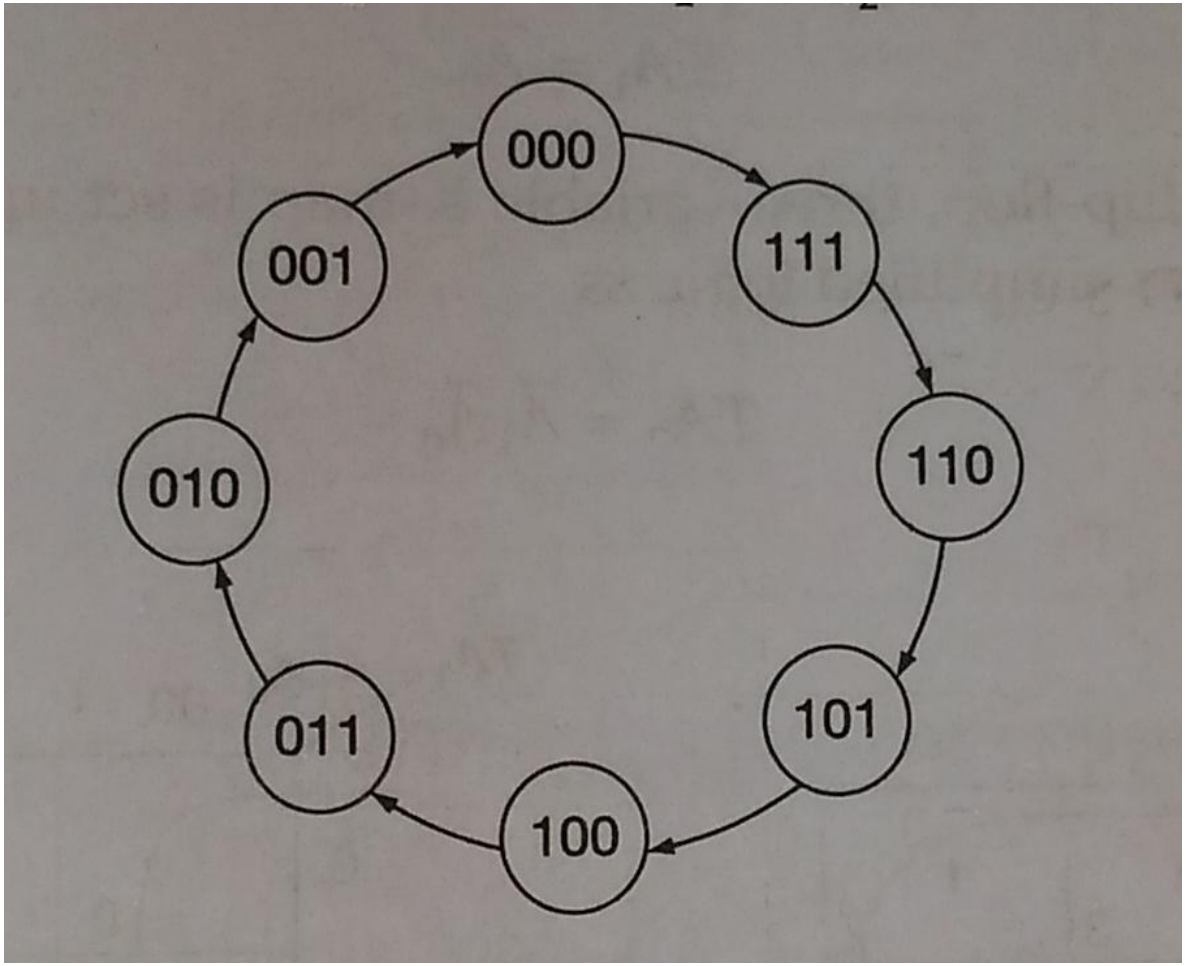
Donald D.Gillie, M.Morris Mano,Digital circuits and design,pearson

MOD-8 Down Counter

10.5.5 | Modulus-8 Synchronous Down Counter

Modulus 8 (MOD-8) synchronous up counters require three flip-flops.

$$2^3 = 8 \Rightarrow 2^n = 2^3, \text{ so } n = 3.$$



Donald D. Givone, M. Morris Mano, Digital circuits and design Pearson

TABLE 10.21 | Down synchronous MOD-8 counter

Clock Pulse Count	Countdown Sequence			Excitations of Flip-flop		
	A_2	A_1	A_0	TA_2	TA_1	TA_0
7	1	1	1	0	0	1
6	1	1	0	0	1	1
5	1	0	1	0	0	1
4	1	0	0	1	1	1
3	0	1	1	0	0	1
2	0	1	0	0	1	1
1	0	0	1	0	0	1
0	0	0	0	1	1	1
7	1	1	1			

$$TA_0(A_2, A_1, A_0) = \sum m(0,1,2,3,4,5,6,7)$$

$$TA_1(A_2, A_1, A_0) = \sum m(0,2,4,6)$$

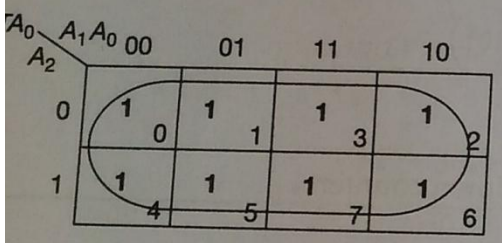
$$TA_2(A_2, A_1, A_0) = \sum m(0,2)$$

$$TA_0 = 1$$

$$TA_1 = \bar{A}_0$$

$$TA_2 = \bar{A}_1 \bar{A}_0$$

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson



GURE 10.60 | K-map for TA_0

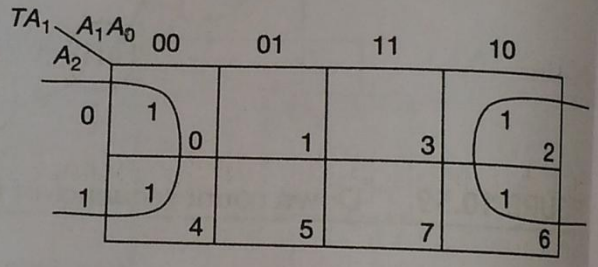
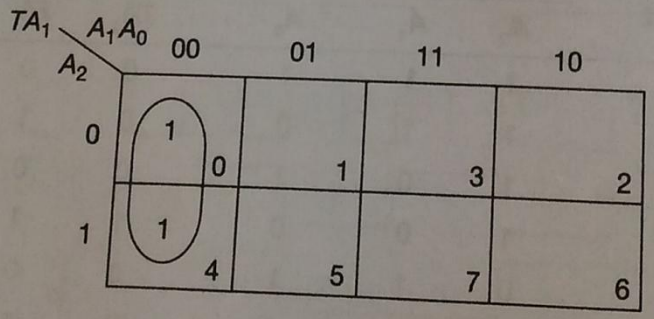
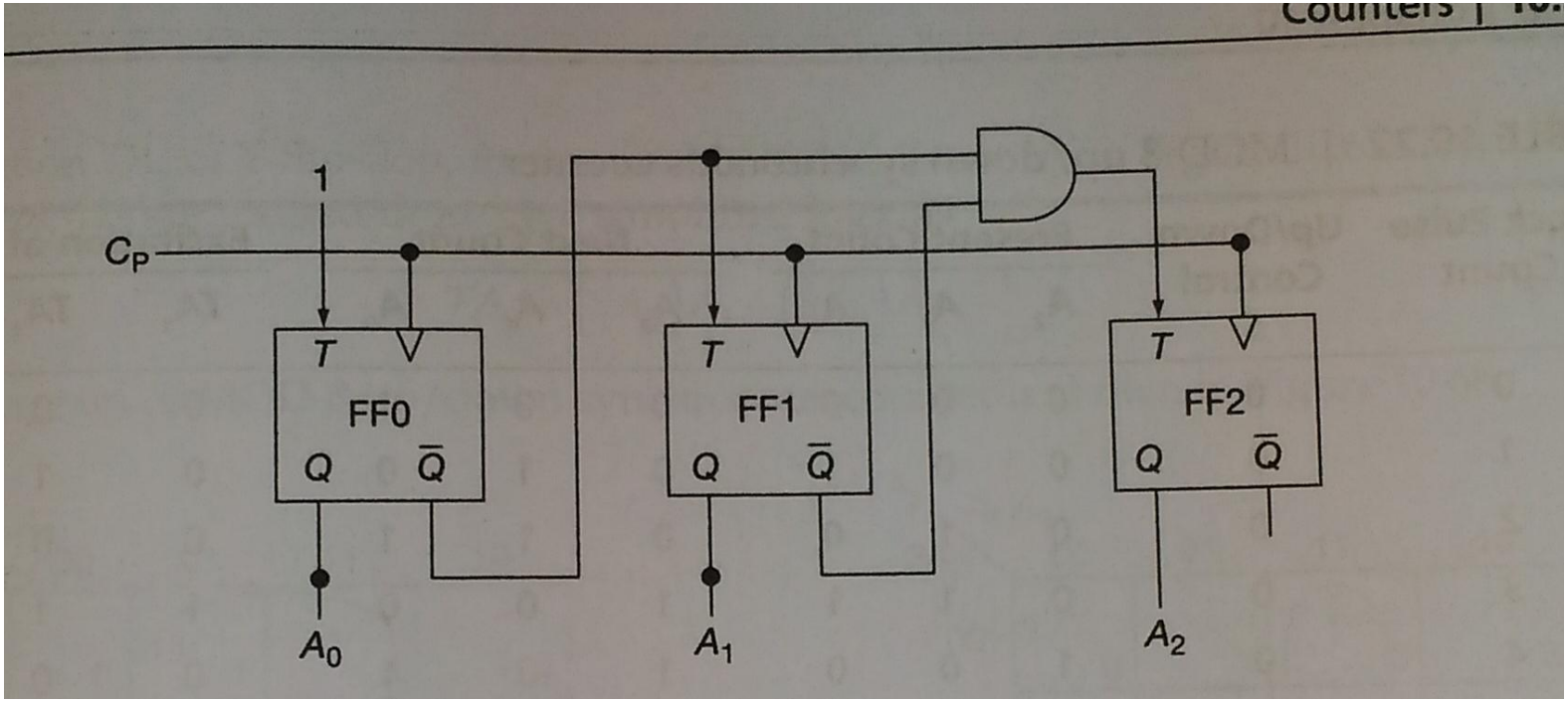
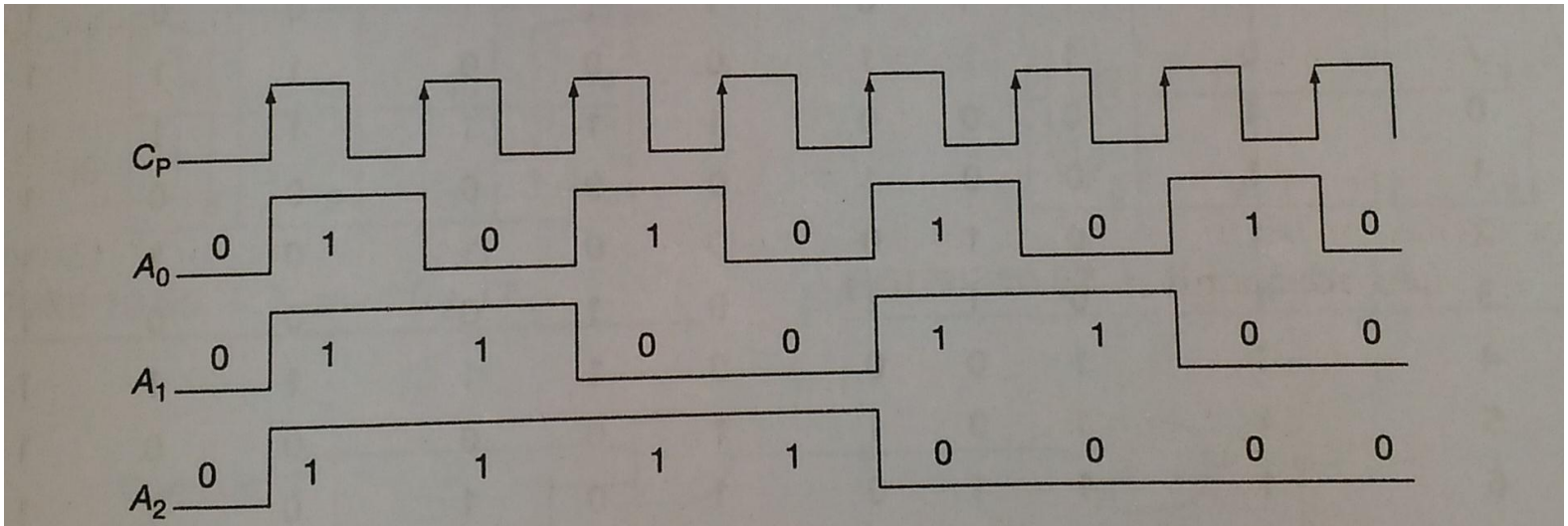


FIGURE 10.61 | K-map for TA_1



RE 10.62 | K-map for TA_2


1
C_P
T
FF0
Q
$\bar{Q}$
T
FF1
Q
$\bar{Q}$
T
FF2
Q
$\bar{Q}$
A_0
A_1
A_2



URE 10.64 | Timing diagram of MOD-8 down synchronous counter

Design a modulus-3 (MOD-3) up synchronous counter using JK-flip-flop.

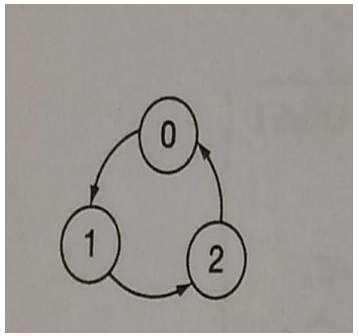


TABLE 10.25 | MOD-3 synchronous counter

Clock Pulse	Sequence up Count		Excitation			
	A_1	A_0	JA_1	KA_1	JA_0	KA_0
0	0	0	0	X	1	X
1	0	1	1	X	X	1
2	1	0	X	1	0	X
	0	0	–		–	

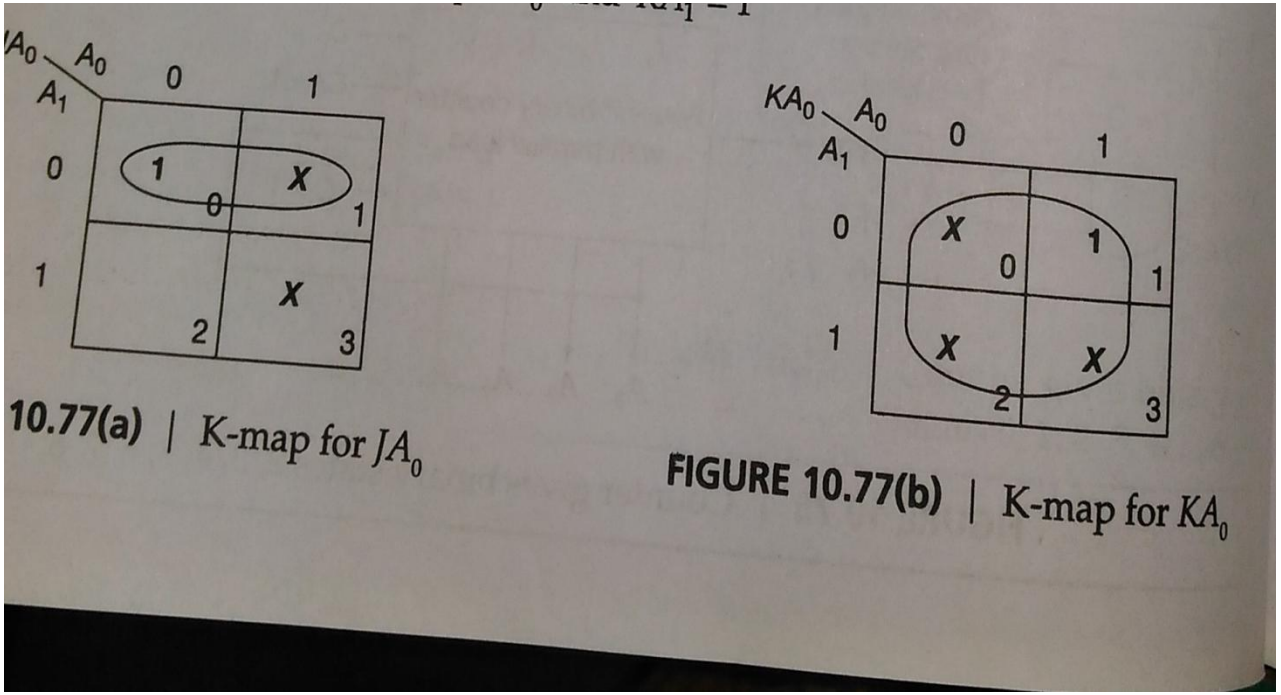
$$JA_0(A_1, A_0) = \sum m(0) + \sum m(1,3)$$

$$KA_0(A_1, A_0) = \sum m(1) + \sum m(0,2,3)$$

$$JA_1(A_1, A_0) = \sum m(1) + \sum d(2,3)$$

$$KA_1(A_1, A_0) = \sum m(2) + \sum d(0,1,3)$$

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson



10.77(a) | K-map for JA_0

FIGURE 10.77(b) | K-map for KA_0

$$JA_0 = \bar{A}_1 \text{ and } KA_0 = 1$$

$$JA_1 = A_0 \text{ and } KA_1 = 1$$

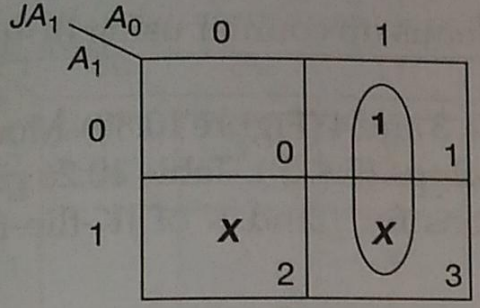


FIGURE 10.77(c) | K-map for JA_1

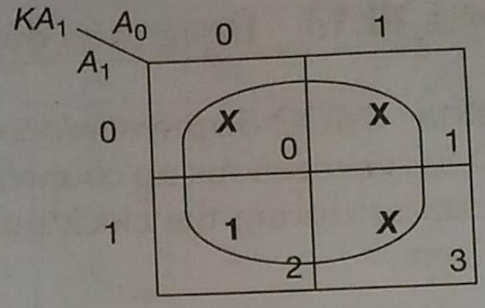


FIGURE 10.77(d) | K-map for KA_1

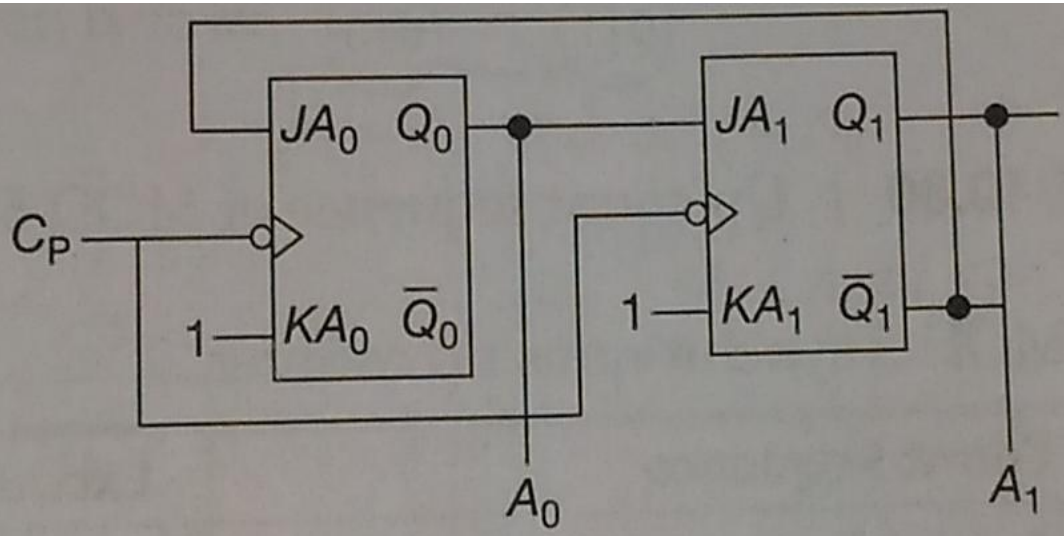


FIGURE 10.78 | MOD-3 synchronous up counter

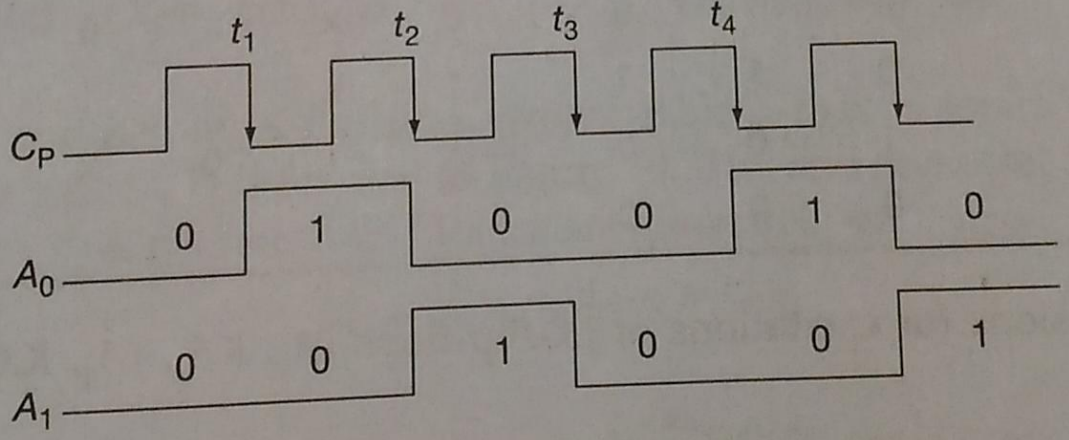


FIGURE 10.79 | Timing diagram of MOD-3 up synchronous counter

BCD to Gray code

TABLE 10.27 | Excitation table

Clock Pulse Count	Present Count				Next Count				Excitation of Flip-flop			
	A_2	A_2	A_1	A_0	g_3	g_2	g_1	g_0	T_3	T_2	T_1	T_0
0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	0	0	0	0
2	0	0	1	0	0	0	1	1	0	0	0	1
3	0	0	1	1	0	0	1	0	0	0	0	1
4	0	1	0	0	0	1	1	0	0	0	1	0
5	0	1	0	1	0	1	1	1	0	0	1	0
6	0	1	1	0	0	1	0	1	0	0	1	1
7	0	1	1	1	0	1	0	0	0	0	1	1
8	1	0	0	0	1	1	0	0	0	1	0	0
9	1	0	0	1	1	1	0	1	0	1	0	0

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

$$T_0(A_3, A_2, A_1, A_0) = \sum m(2,3,6,7) + \sum d(10,11,12,13,14,15)$$

$$T_1(A_3, A_2, A_1, A_0) = \sum (4,5,6,7) + \sum d(10,11,12,13,14,15)$$

$$T_2(A_3, A_2, A_1, A_0) = \sum (8,9) + \sum d(10,11,12,13,14,15)$$

$$T_3(A_3, A_2, A_1, A_0) = 0$$

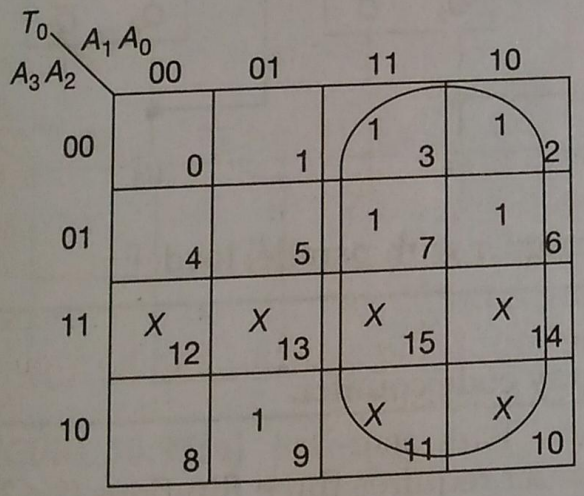


FIGURE 10.93(a) | K-map for T_0

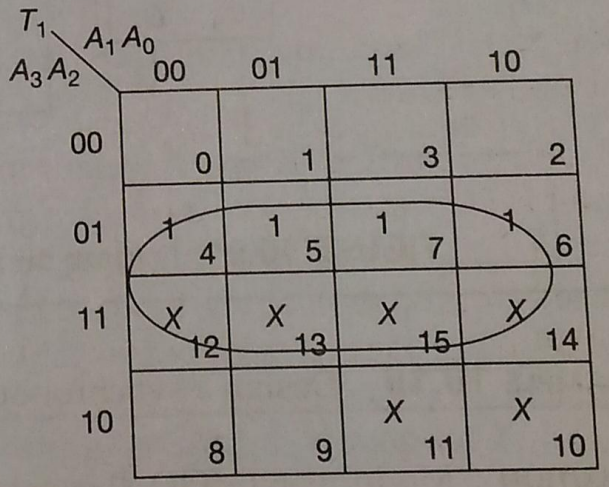


FIGURE 10.93(b) | K-map for T_1

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

$$T_0 = A_1$$

$$T_2 = A_3$$

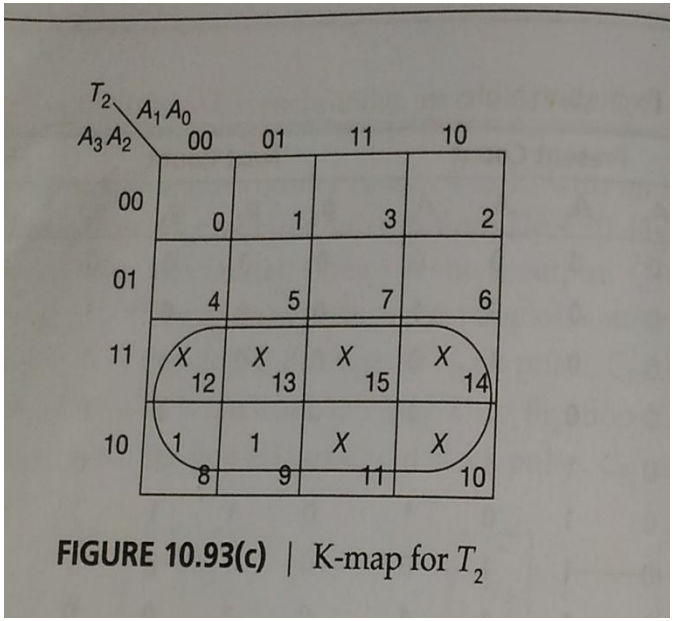
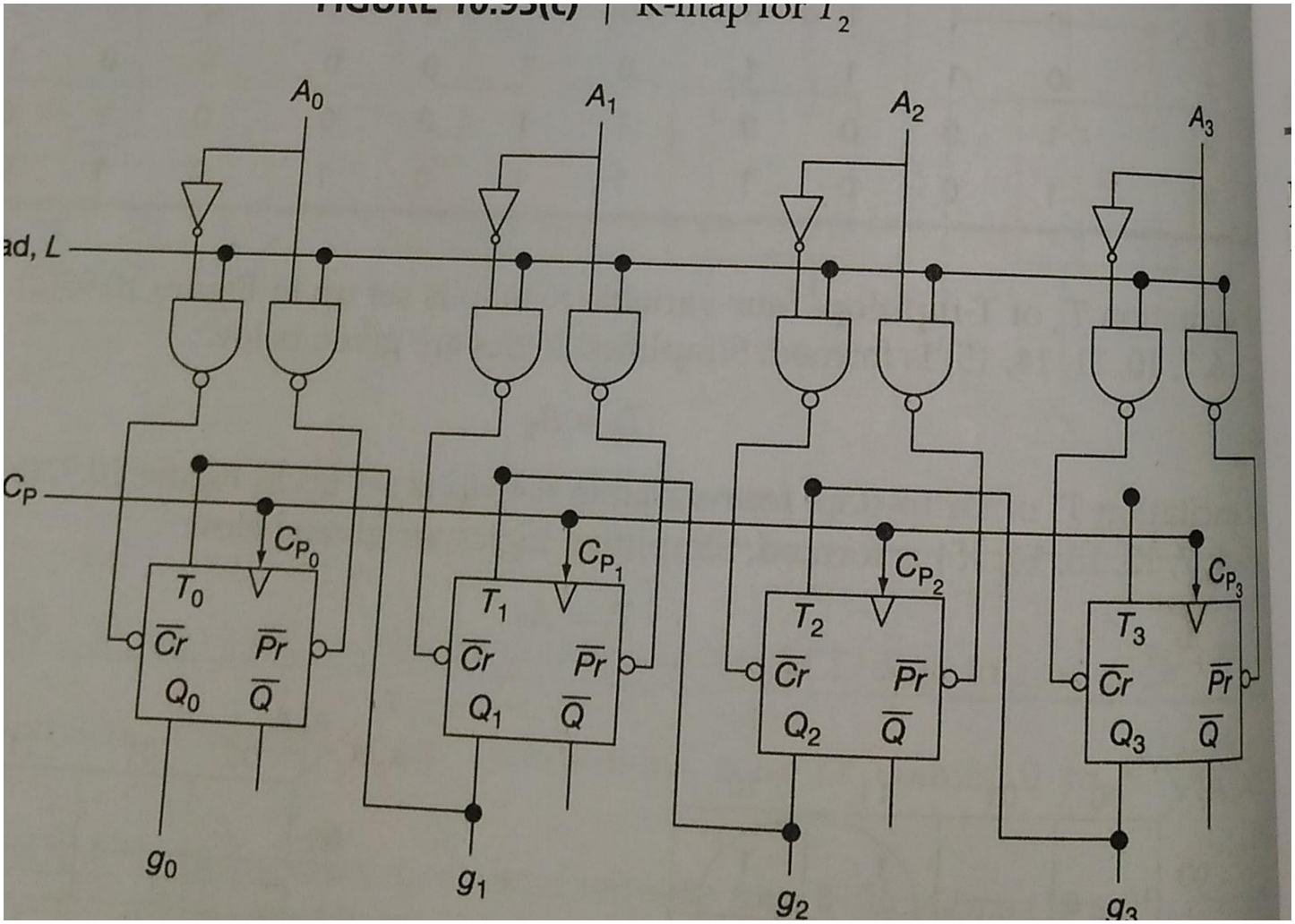


FIGURE 10.93(c) | K-map for T_2

$$T_1 = A_2$$


K-map for T_2
A_0
A_1
A_2
A_3
ad, L
C_P
C_{P_0}
C_{P_1}
C_{P_2}
C_{P_3}
T_0
T_1
T_2
T_3
$\overline{Cr}$
$\overline{Pr}$
Q_0
Q_1
Q_2
Q_3
$\overline{Q}$
g_0
g_1
g_2
g_3

EXAMPLE 10.17 Design a synchronous counter that goes through 0, 1, 2, 4, 0. Unused states must go to zero or to next state on next clock pulse.

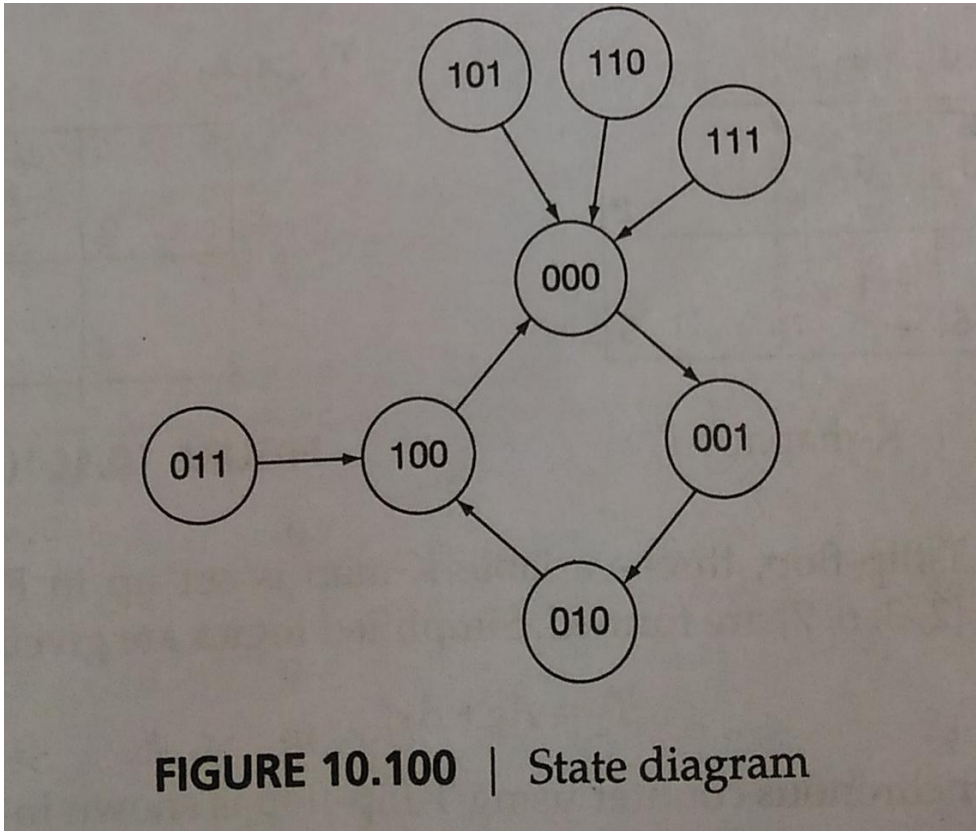


FIGURE 10.100 | State diagram

TABLE 10.29 | MOD-8 synchronous counter

Clock Pulse Count	Present Count			Next Count			Excitation of Flip-flop		
	A_2	A_1	A_0	A_2	A_1	A_0	T_2	T_1	T_0
0	0	0	0	0	0	1	0	0	1
1	0	0	1	0	1	0	0	1	1
2	0	1	0	1	0	0	1	1	0
3	0	1	1	1	0	0	1	1	1
4	1	0	0	0	0	0	1	0	0
5	1	0	1	0	0	0	1	0	1
6	1	1	0	0	0	0	1	1	0
7	1	1	1	0	0	0	1	1	1

$$T_0(A_2, A_1, A_0) = \sum m(0,1,3,5,7)$$

$$T_1(A_2, A_1, A_0) = \sum m(1,2,3,6,7)$$

$$T_2(A_2, A_1, A_0) = \sum m(2,3,4,5,6,7)$$

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

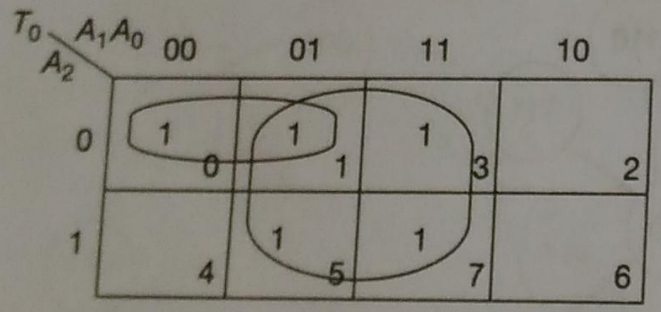


FIGURE 10.101(a) | K-map for T_0

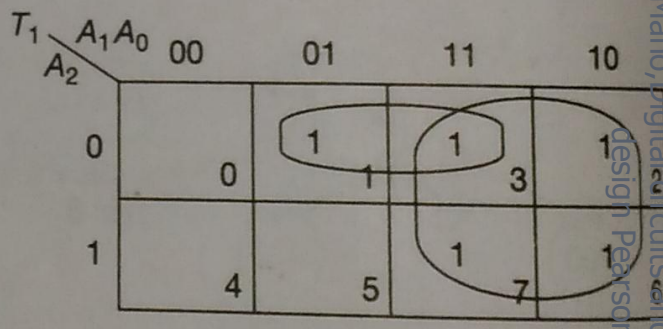


FIGURE 10.101(b) | K-map for T_1

$$T_0 = \bar{A}_2\bar{A}_1 + A_0$$

$$T_1 = \bar{A}_2 A_0 + A_1$$

$$T_2 = A_1 + A_2$$

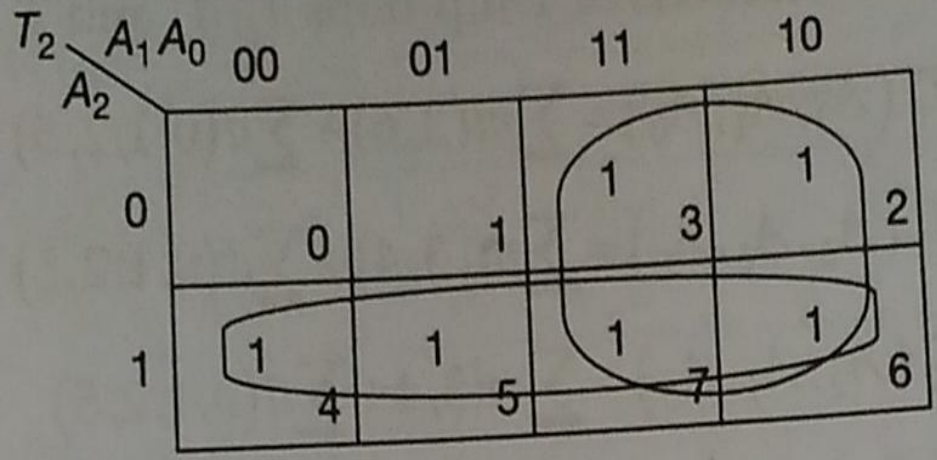


FIGURE 10.101(c) | K-map for T_2

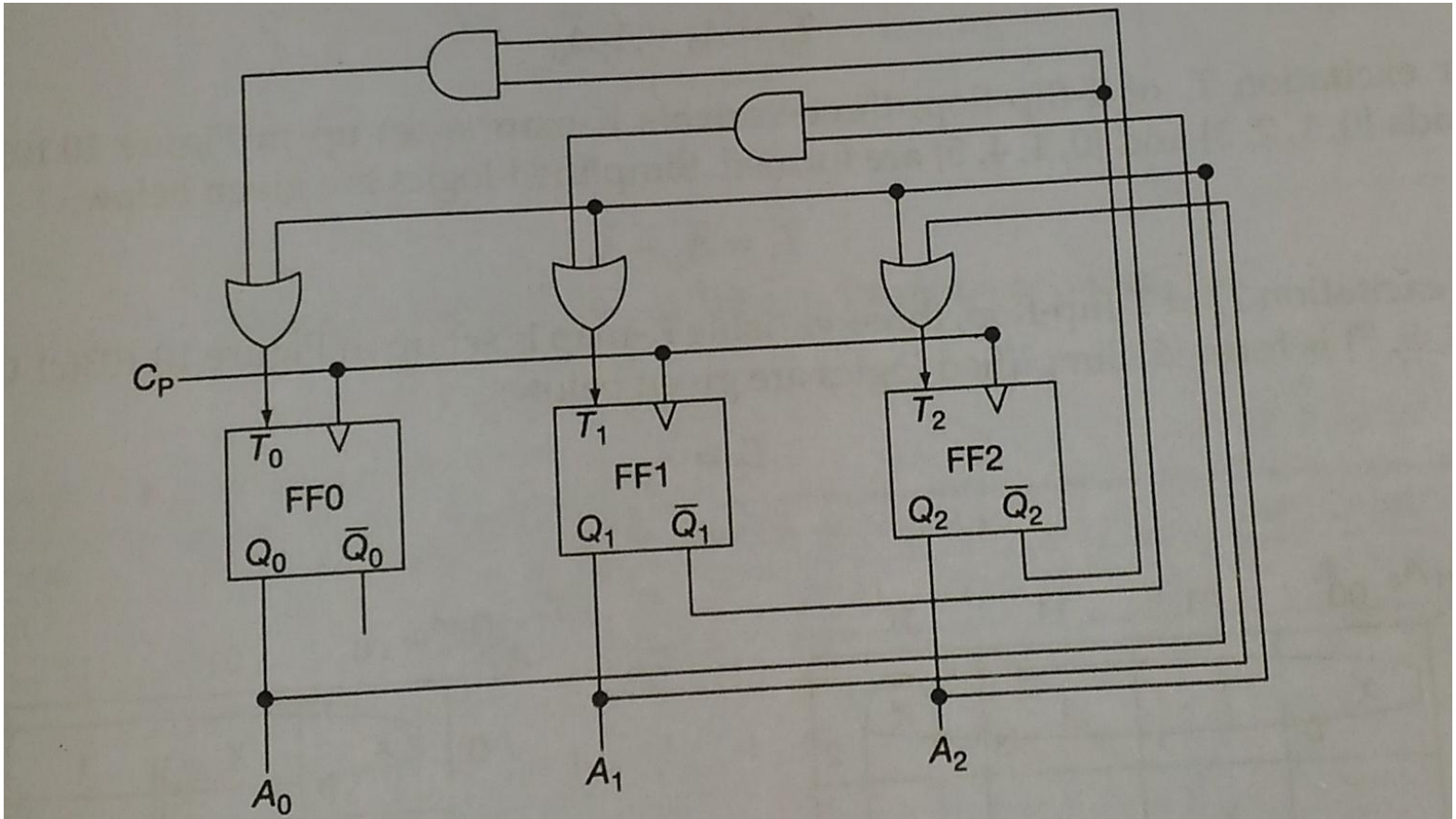


FIGURE 10.102 | Synchronous counter

Ring Counter

- Counter is nothing but a digital device meant to count. These are usually built using bi-stable devices called flip-flops.
- Generally either D or JK type flip-flops are used to design the counters, no matter which type they are of applies even for **ring counter**.
- In fact, the way of connection which exists between the flip-flops is the factor which determines the kind of counter designed. Now, let us assume that we have arranged all the flip-flops in series such that the output of the preceding flip-flop is fed as an input to the immediate next flip-flop. Further, let us connect the output of the last flip-flop as an input to the first flip-flop in the chain.

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

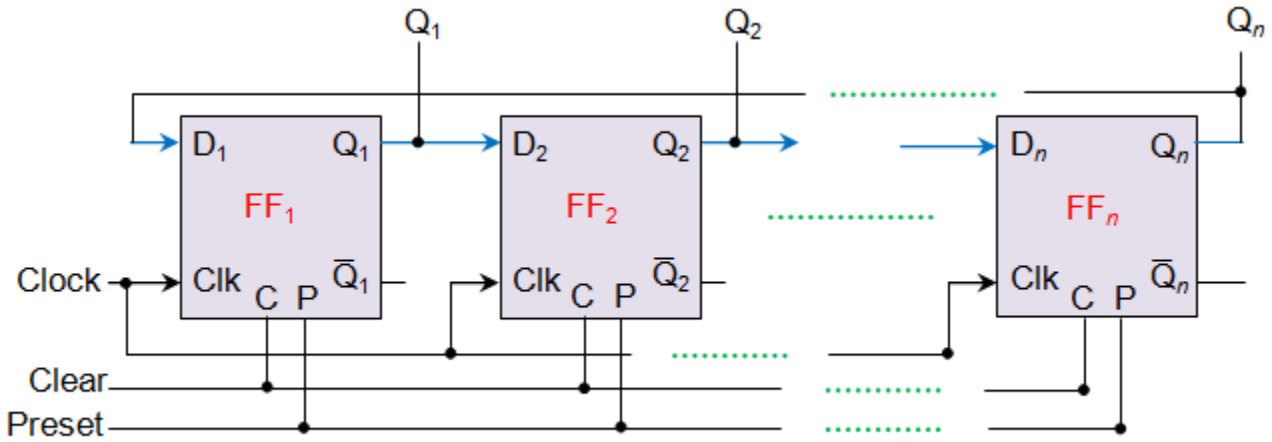


Figure 1 *n*-bit Ring Counter Designed Using D Flip-Flops

Clock Cycle	Q_1	Q_2	Q_3	Q_4
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

Bit-pattern repeats for every 4 clock cycles

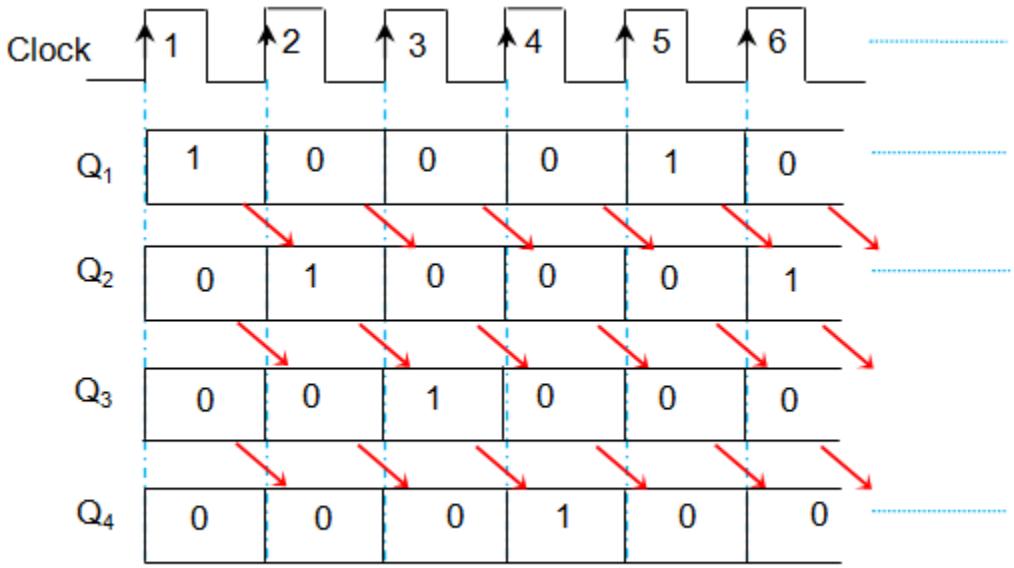


Figure 2 Output waveform of 4-bit ring counter

JOHNSON COUNTER

- A circuit which is used to count the number of times an event occurs is called a counter.
- In digital sense, these circuits comprise of bi-stable devices called flip-flops arranged in a particular fashion.
- Such a chain can be regarded to be a shift register, due to which counters can be considered as an application of shift registers.
- Either D or JK type flip-flops are preferred while designing the counter circuits. One such counter designed using D flip-flop chain is shown by Figure and is called **Johnson counter**.

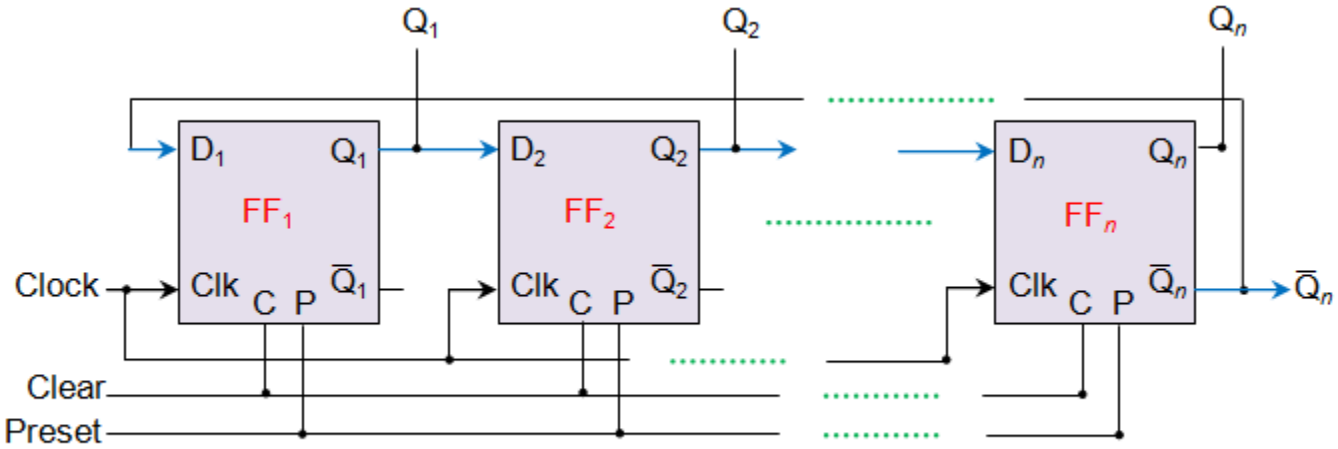


Figure 1 *n*-bit Johnson Counter Designed Using D Flip-Flops

Table I Truth Table of 3-bit Johnson Counter

Clock Cycle	Q_1	Q_2	Q_3	$\bar{Q}_3$
1	0	0	0	1
2	1	0	0	1
3	1	1	0	1
4	1	1	1	0
5	0	1	1	0
6	0	0	1	0
7	0	0	0	1
8	1	0	0	1
9	1	1	0	1
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

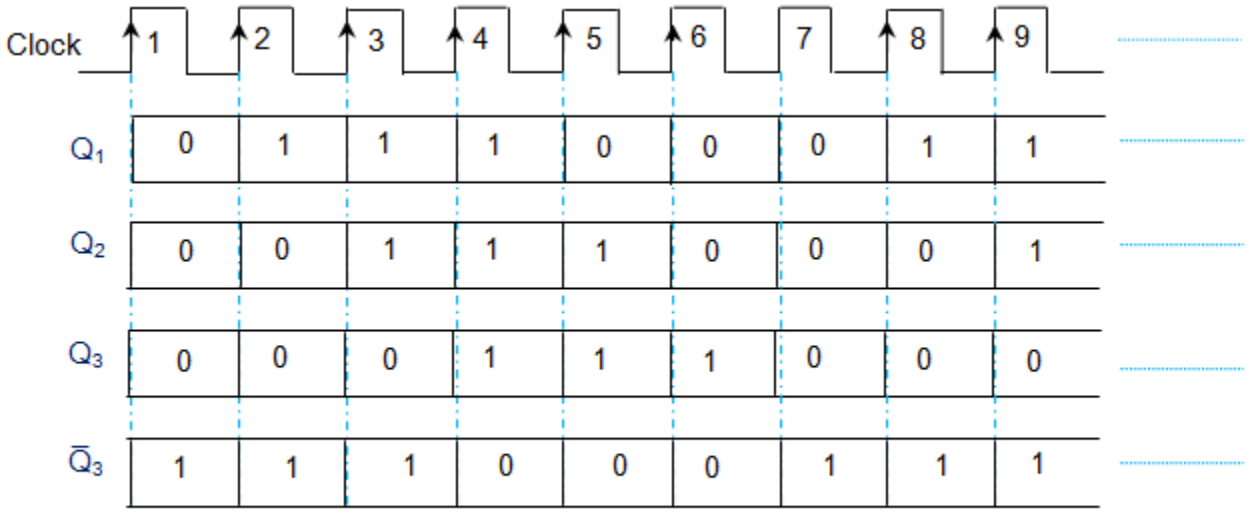
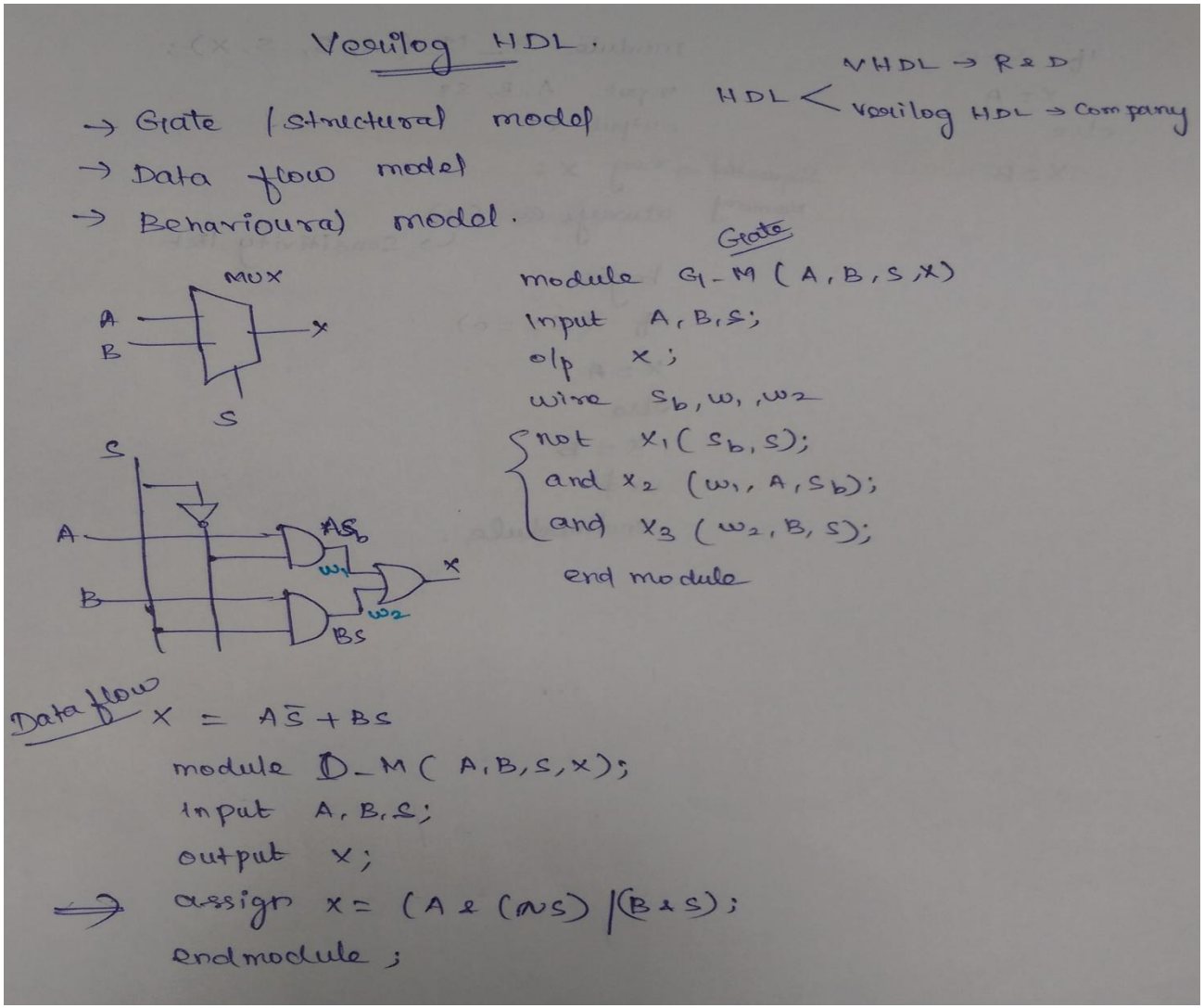


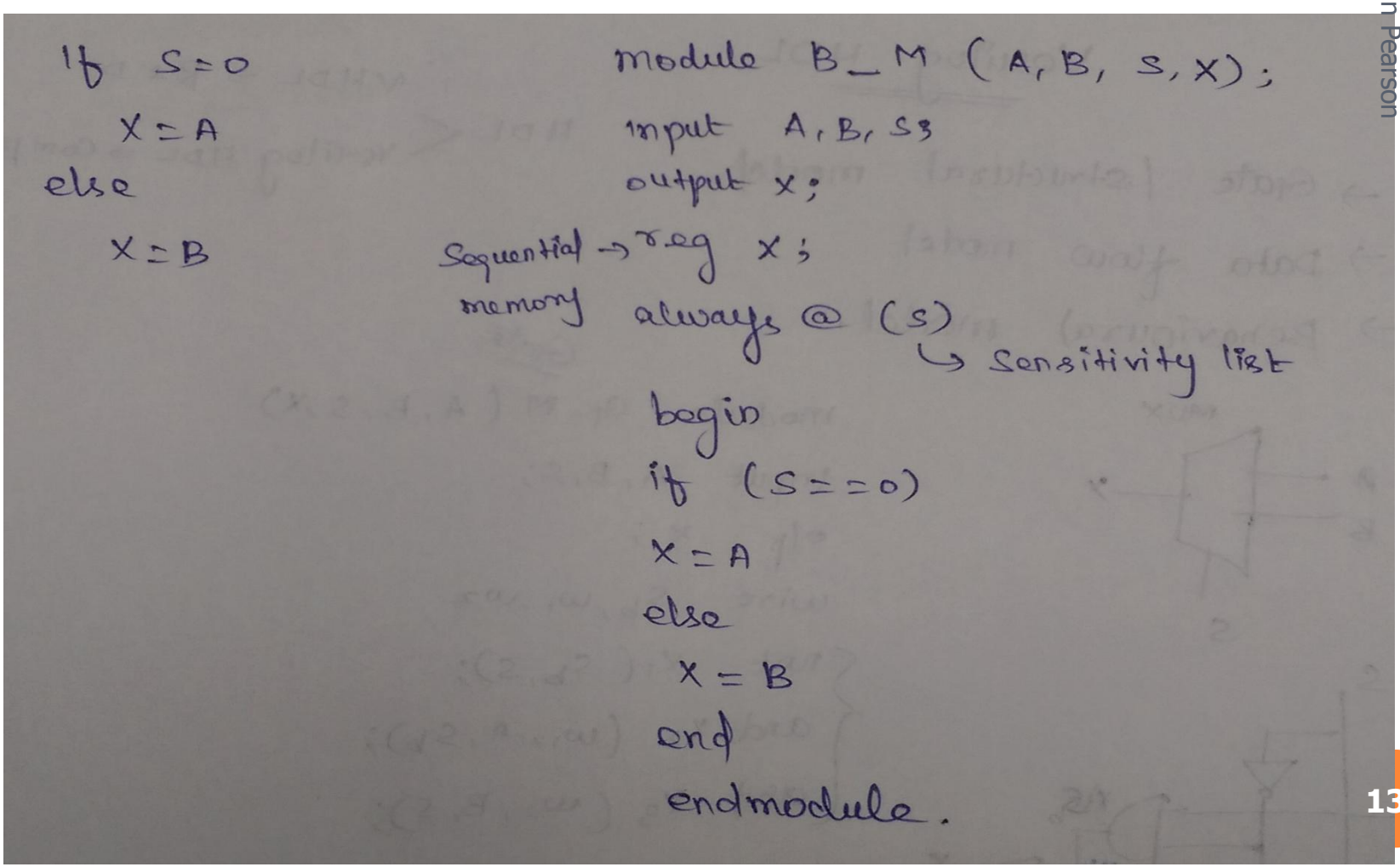
Figure 2 Output Waveform of 3-bit Johnson Counter

HDL

Donald D.Givone, M.Morris Mano,Digital circuits and design Pearson

VERILOG HDL


Verilog HDL.
→ Gate (structural) model
→ Data flow model
→ Behavioural model.
HDL < VHDL → R & D
Verilog HDL → Company
MUX
A
B
S
X
Gate
module G1-M (A, B, S, X)
input A, B, S;
o/p X;
wire Sb, w1, w2
not X1 (Sb, S);
and X2 (w1, A, Sb);
and X3 (w2, B, S);
end module
AS
BS
Data flow
X = AS̄ + BS
module D_M (A, B, S, X);
input A, B, S;
output X;
assign X = (A & (~S)) | (B & S);
endmodule;


If S=0
X=A
else
X=B
module B_M (A, B, S, X);
input A, B, S;
output X;
Sequential memory → reg X;
always @ (S)
↳ Sensitivity list
begin
if (S==0)
X=A
else
X=B
end
endmodule.

5-5 HDL For Sequential Circuits

The Verilog hardware description language (HDL) is introduced in Section 3-9. The description of combinational circuits and an introduction to behavioral modeling is presented in Section 4-11.

- In this section, we continue the discussion of the behavioral modeling and present description examples of flip-flops and sequential circuits.

Behavioral Modeling

There are two kinds of behavioral statements in Verilog HDL: **initial** and **always**.

```
initial
   begin
        clock = 1`b0;
        repeat  (30)
        #10 clock = ~clock;
   end
```

```
initial
   begin
        clock = 1`b0;
        #300  $finish;
   end
always
        #10 clock = ~clock;
```

Behavioral Modeling

The **always** statement can be controlled by delays that wait for a certain time or by certain conditions to become true or by events to occur.

always @(event control expression)
procedural assignment statements.

```
always @(A or B or Reset)
```

```
always @(posedge clock or negedge reset)
```

Flip-Flops and Latches

HDL Example 5-1

```
//Description of D latch (See Fig. 5-6)
module D_latch (Q, D, control);
      output Q;
      input D, control;
      reg Q;
      always @(control or D)
      if (control) Q = D;        //Same as: if (control ==1)
endmodule
```

Flip-Flops and Latches

HDL Example 5-2

```
//D flip-flop
module D_FF (Q, D, CLK);
      output Q;
      input D, CLK;
      reg Q;
      always @(posedge CLK)
          Q = D;
endmodule
```

Flip-Flops and Latches

HDL Example 5-3

```
//T flip-flop from D flip-flop and gates
module TFF (Q, T, CLK, RST);
      output Q;
      input T, CLK, RST;
      reg DT;
      assign DT = Q ^ T;
//Instantiate the D flip-flop
   DFF TF1 (Q, DT, CLK, RST);
Endmodule
```

```
// JK flip-flop from D flip-flop and gates
module JKFF (Q, J, K, CLK, RST);
         output Q;
         input J, K, CLK, RST;
         wire JK;
         assign JK = (J & ~Q) | (~K & Q);
// Instantiate D flipflop
    DFF JK1 (Q, JK, CLK, RST);
endmodule
```

Flip-Flops and Latches

HDL Example 5-4

```
// Functional description of JK flip-flop
module   JK_FF (J, K, CLK, Q, Qnot);
      output Q, Qnot;
      input   J, K, CLK;
      reg     Q;
      assign Qnot = ~Q;
      always @ (posedge CLK)
            case  ({J, K})
               2`b00: Q = Q;
               2`b01: Q = 1`b0;
               2`b10: Q = 1`b1;
               2`b11: Q = ~Q;
             endcase
endmodule
```

State Diagram

HDL Example 5-5

```
//Mealy state diagram (Fig. 5-16)
module  Mealy_mdl (x, y, CLK, RST);
      input x, CLK, RST;
      output y;
      reg y;
      reg [1:0] Prstate, Nxtstate;
      parameter S0 = 2`b00, S1 = 2`b01, S2 = 2`b10, S3 =
                     2`b11;
        always @ (posedge CLK or negedge RST)
            if (~RST) Prstate = S0;    //Initialize to state S0
            else Prstate = Nxtstate;  //Clock operations

        always @ (Prstate or x)
                case (Prstate)
```

State Diagram

```
            S0: if (x) Nxtstate = S1;
                  else Nxtstate = S0;
            S1: if (x) Nxtstate = S3;
                  else Nxtstate = S0;
            S2: if (x) Nxtstate = S0;
                  else Nxtstate = S2;
            S3: if (x) Nxtstate = S2;
                  else Nxtstate = S0;
            endcase
always @ (Prstate or x)       //Evaluate output
        case (Prestate)
            S0: y = 0;
            S1: if (x) y = 1`b0; else y = 1`b1;
            S2: if (x) y = 1`b0; else y = 1`b1;
            S3: if (x) y = 1`b0; else y = 1`b1;
        endcase
endmodule
```

State Diagram

HDL Example 5-6

```
//Moore state diagram (Fig. 5-19)
module  Moore_md1 (x, AB, CLK, RST);
      input x, CLK, RST;
     output [1:0] AB;
     reg [1:0] state;
     parameter S0 = 2`b00, S1 = 2`b01, S2 = 2`b10, S3 = 2`b11;
        always @ (posedge CLK or negedge RST)
          if (~RST) state = S0;   //Initialize to state S0
          else
          case (state)
             S0: if (~x) state = S1; else state = S0;
             S1: if (x)   state = S2; else state = S3;
             S2: if (~x) state = S3; else state = S2;
             S3: if (~x) state = S0; else state = S3;
          endcase
     assign AB = state;              //Output of flip-flops
endmodule
```

Structural Description

HDL Example 5-7

```
//Structural description of sequential circuit
//See Fig. 5-20 (a)
module Tcircuit (x, y, A, B, CLK, RST);
    input x, CLK, RST;
    output y, A, B;
    wire TA, TB;
//Flip-flop input equations
    assign TB = x,
           TA = x & B;
//Output equation
    assign y = A & B;
//Instantiate T flip-flops
    T_FF BF (B, TB, CLK, RST);
    T_FF AF (A, TA, CLK, RST);
endmodule
```

Structural Description

```
//T flip-flop
module T_FF (Q, T, CLK, RST);
      output Q;
      input T, CLK, RST;
      reg Q;
           always @ (posedge CLK or negedge RST)
                if (~RST) Q = 1`b0;
                else Q = Q ^ T;
endmodule
```

Structural Description

```
//Stimulus for testing sequential circuit
module testTcircuit;
    reg x, CLK, RST;    //inputs for circuit
    wire y, A, B;         //output from circuit
Tcircuit TC (x, y, A, B, CLK, RST); //instantiate circuit
 initial
    begin
          RST = 0;
          CLK = 0;
    #5   RST = 1;
          repeat (16)
    #5   CLK = ~CLK;
     end
 initial
    begin
           x = 0;
    #15   x = 1;
          repeat (8)
    #10   x = ~x;
    end
endmodule
```

Structural Description

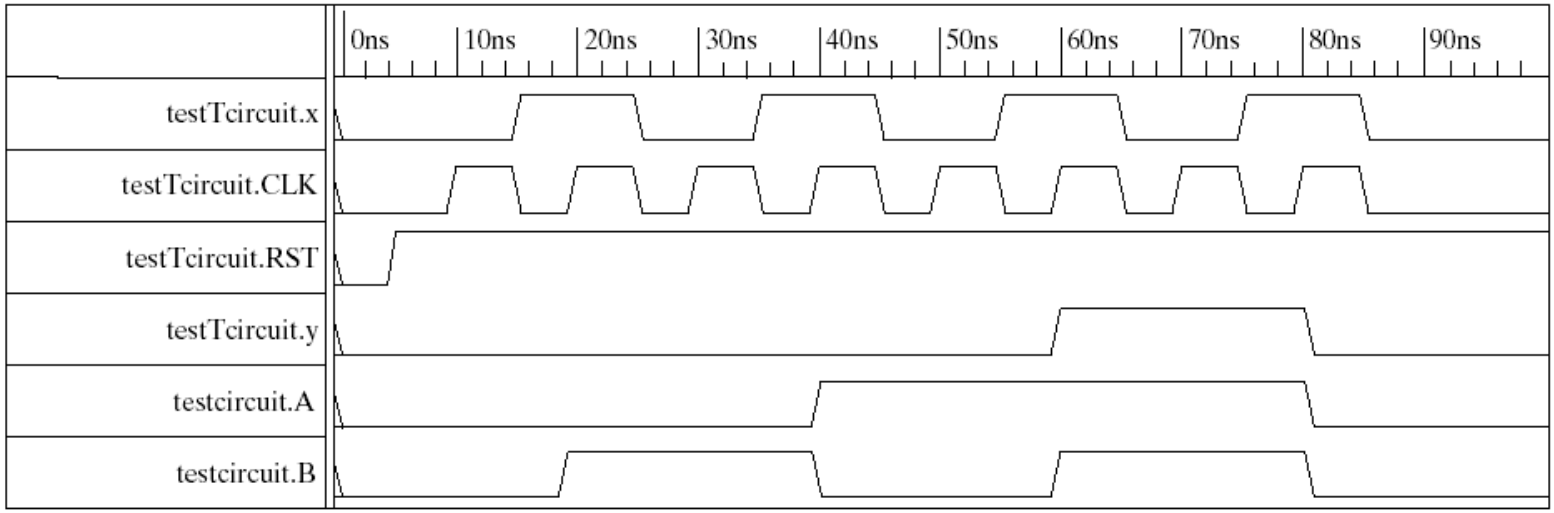


Fig. 5-21 Simulation Output of HDL Example 5-7

5-6 State Reduction and Assignment

- The analysis of sequential circuits starts from a circuit diagram and culminates in a state table or diagram.

- The design of a sequential circuit starts from a set of specifications and culminates discusses certain properties of sequential circuits that may be used to reduce the number of gates and flip-flops during the design.

State Reduction

- The reduction of the number of flip-flops in a sequential circuit is referred to as the state-reduction problem. State-reduction algorithms are concerned with procedures for reducing the number of states in a state table, while keeping the external input-output requirements unchanged.

- Since m flip-flops produce 2^m states, a reduction in the number of states may result in a reduction in the number of flip-flops. An unpredictable effect in reducing the number of flip-flops is that sometimes the equivalent circuit may require more combinational gates.

State Reduction

Example :

state	a	a	b	c	d	e	f	f	g	f	g	a
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

Initial point

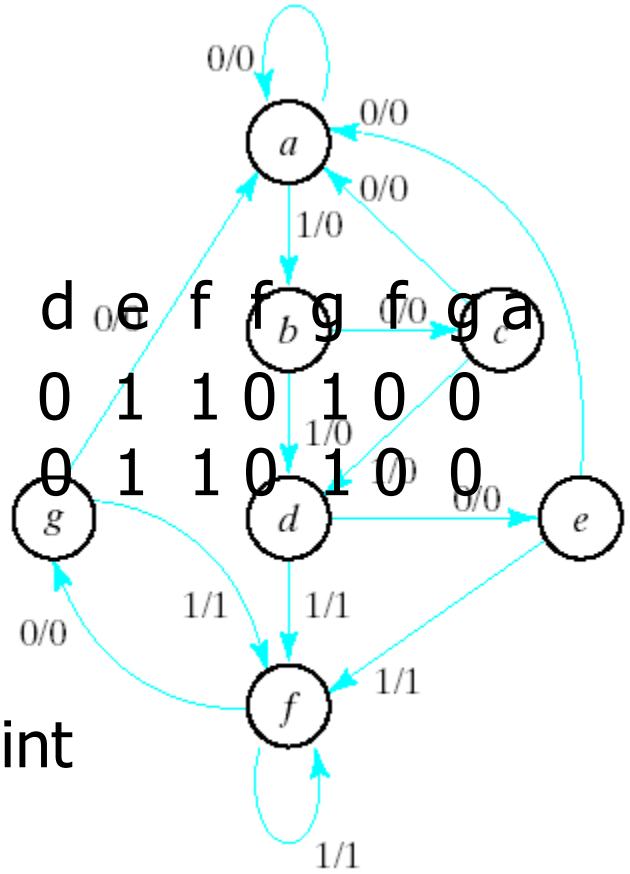


Fig. 5-22 State Diagram

State Reduction

We now proceed to reduce the number of states for this example. First, we need the state table; it is more convenient to apply procedures for state reduction using a table rather than a diagram. The state table of the circuit is listed in Table 5-6 and is obtained directly from the state diagram.

Table 5-6
State Table

Present State	Next State $x = 0$	Next State $x = 1$	Output $x = 0$	Output $x = 1$
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

State Reduction

States g and e are two such states: they both go to states a and f and have outputs of 0 and 1 for x=0 and x=1, respectively. Therefore, states g and e are equivalent and one of these states can be removed. The procedure of removing a state and replacing it by its equivalent is demonstrated in Table 5-7. The row with present g is removed and state g is replaced by state e each time it occurs in the next-state columns.

Table 5-7
Reducing the State Table

Present State	Next State $x=0$	Next State $x=1$	Output $x=0$	Output $x=1$
a	b		0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

State Reduction

Present state f now has next states e and f and outputs 0 and 1 for x=0 and x=1, respectively. The same next states and outputs appear in the row with present state d. Therefore, states f and d are equivalent and state f can be removed and replaced by d. The final reduced table is shown in Table 5-8. The state diagram for the reduced table consists of only five states and is shown in Fig. 5-23.

Table 5-8
Reduced State Table

Present State	Next State $x=0$	Next State $x=1$	Output $x=0$	Output $x=1$
a	b	d	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1

State Reduction

state	*a*	*a*	*b*	*c*	*d*	*e*	*d*	*c*
input	0	1	0	1	0	1	1	0
output	0	0	0	0	0	1	1	0

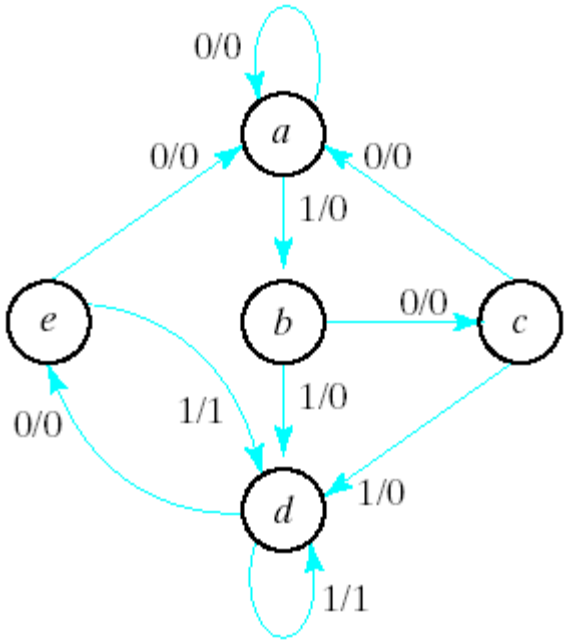


Fig. 5-23 Reduced State Diagram

State Assignment

Table 5-9
Three Possible Binary State Assignments

State	Assignment 1 Binary	Assignment 2 Gray code	Assignment 3 One-hot
a	000	000	00001
b	0		
c	0		
d	0		
e	1		

Table 5-10
Reduced State Table with Binary Assignment 1

Present State	Next State $x = 0$	Next State $x = 1$	Output $x = 0$	Output $x = 1$
000	000	001	0	0
001	010	011	0	0
010	000	011	0	0
011	100	011	0	1
100	000	011	0	1

5-7 Design Procedure

The procedure for designing synchronous sequential circuits can be summarized by a list of recommended steps.

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

Design Procedure

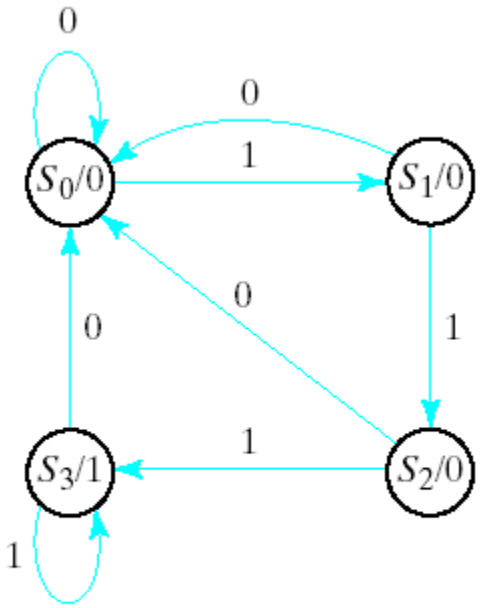


Fig. 5-24 State Diagram for Sequence Detector

SYNTHESIS USING D FLIP-FLOPS

$$A(t+1) = D_A(A, B, x) = \Sigma\,(3, 5, 7)$$
$$B(t+1) = D_B(A, B, x) = \Sigma\,(1, 5, 7)$$
$$y(A, B, x) = \Sigma\,(6, 7)$$

Table 5-11
State Table for Sequence Detector

Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	1

Synthesis Using D Flip-Flops

$D_A = Ax + Bx$

$D_B = Ax + B`x$

$y = AB$

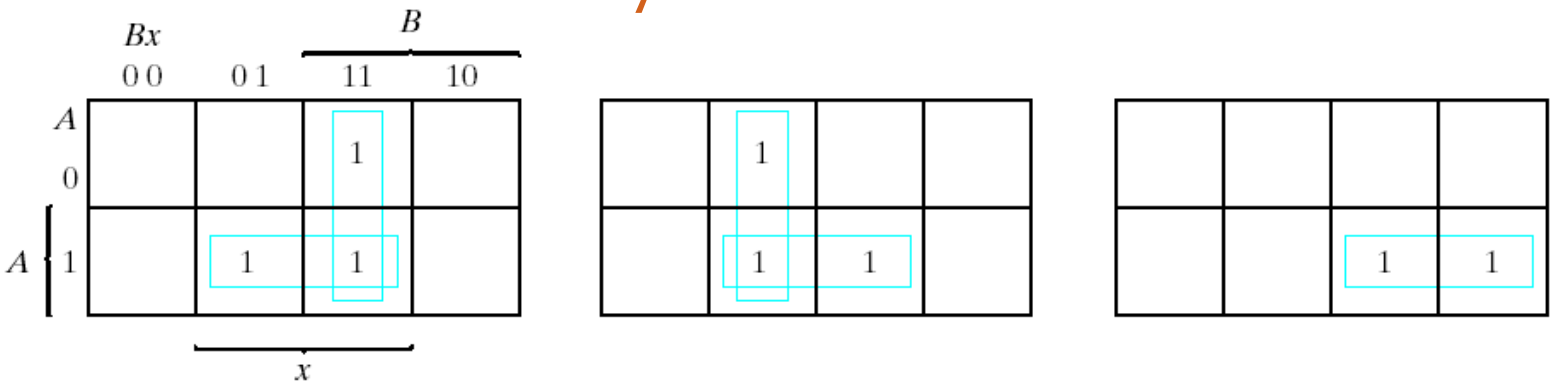


Fig. 5-25 Maps for Sequence Detector

SYNTHESIS USING D FLIP-FLOPS

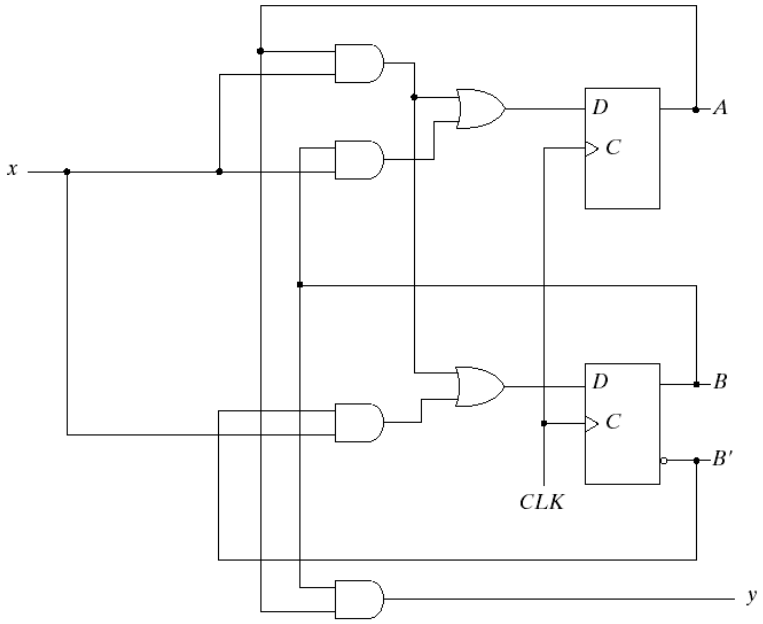


Fig. 5-26 Logic Diagram of Sequence Detector

Synthesis Using JK Flip-Flops

Different from Table 5-11 !!

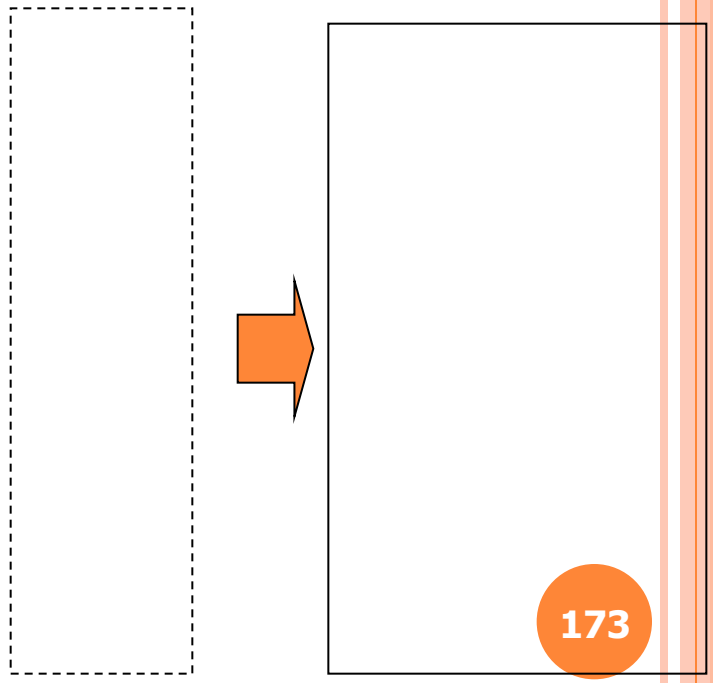
Table 5-12
Flip-Flop Excitation Tables

$Q(t)$	$Q(t+1)$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

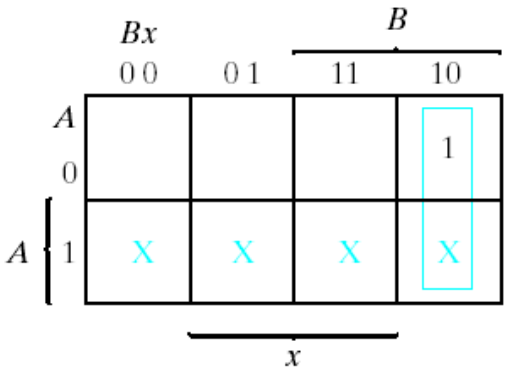
(a) JK

Flop Inputs		
	J_B	K_B
‹	0	X
‹	1	X
‹	X	1
‹	X	0
›	0	X
›	1	X
›	X	0
1	X	1

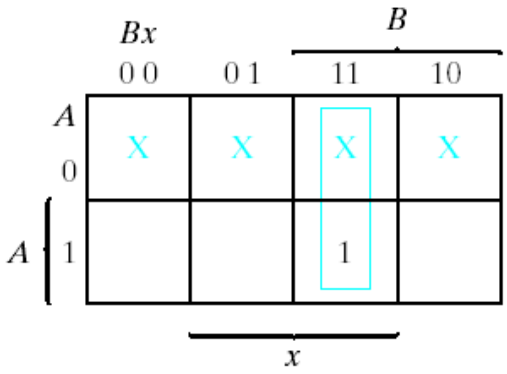
Ref. Table 5-1



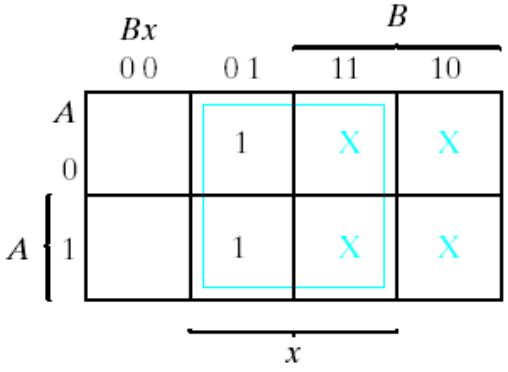
SYNTHESIS USING JK FLIP-FLOPS



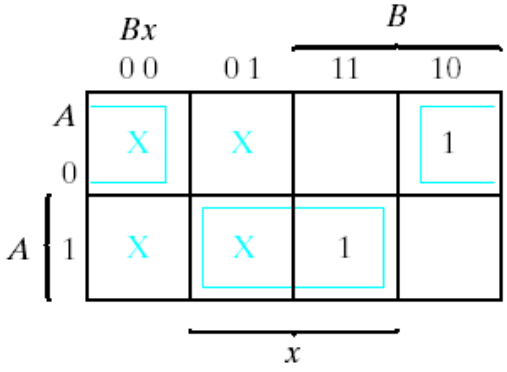
$J_A = Bx'$



$K_A = Bx$



$J_B = x$



$K_B = (A \oplus x)'$

Fig. 5-27 Maps for J and K Input Equations

SYNTHESIS USING JK FLIP-FLOPS

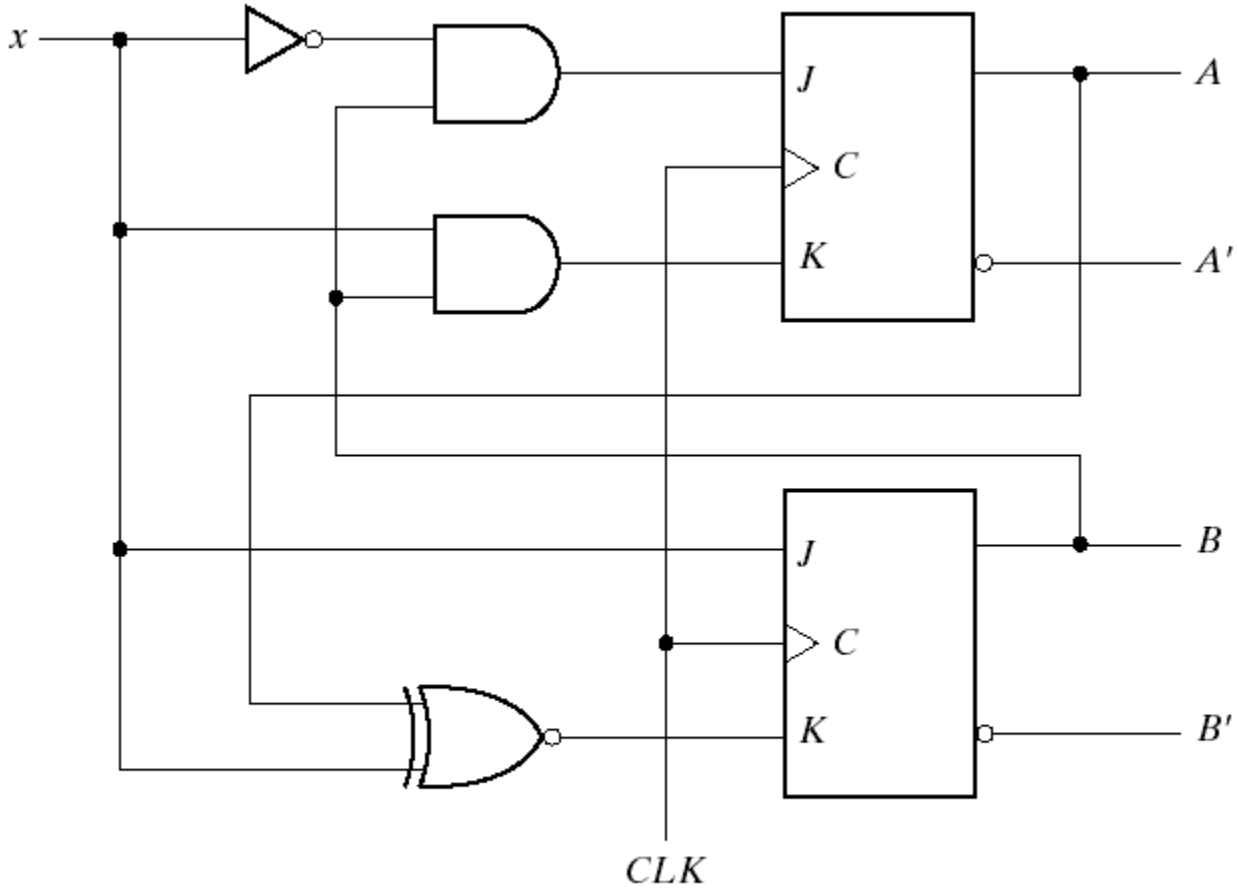
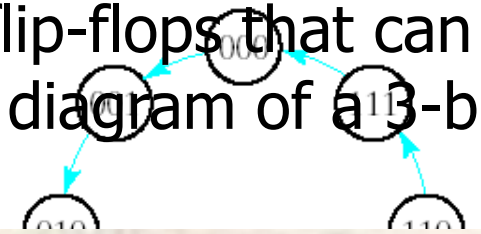


Fig. 5-28 Logic Diagram for Sequential Circuit with *JK* Flip-Flops

Synthesis Using T Flip-Flops

The synthesis using T flip-flops will be demonstrated by designing a binary counter. An n-bit binary counter consists of n flip-flops that can count in binary from 0 to 2^n-1. The state diagram of a 3-bit counter is shown in Fig. 5-29.



Q(t)	Q(t + 1)	T
0	0	0
0	1	1
1	0	1
1	1	0

(b) T

Ref. Table 5-1

SYNTHESIS USING T FLIP-FLOPS

Table 5-14
State Table for 3-Bit Counter

Present State			Next State			Flip-Flop Inputs		
A_2	A_1	A_0	A_2	A	A_0	T_{A2}	T_{A1}	T_{A0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	1	1
1	1	1	0	0	0	1	1	1

Synthesis Using T Flip-Flops

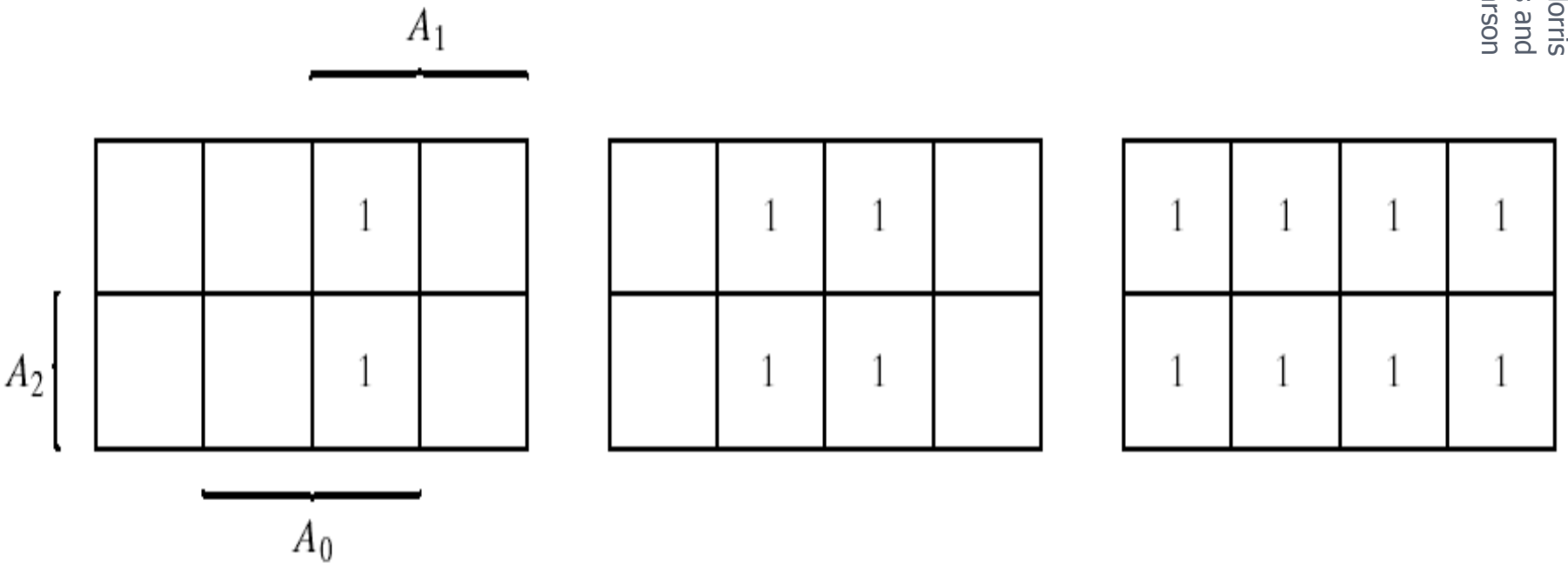


Fig. 5-30 Maps for 3-Bit Binary Counter

SYNTHESIS USING T FLIP-FLOPS

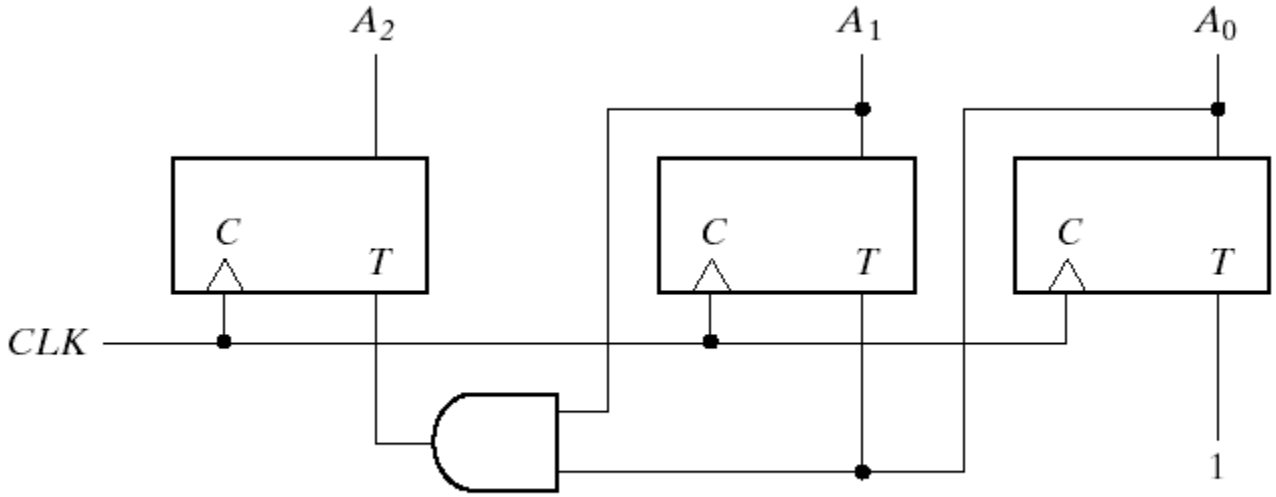


Fig. 5-31 Logic Diagram of 3-Bit Binary Counter