* Analysis of Clocked Sequential Circuits

The analysis of a sequential circuit consists of obtaining a state table and/or a state diagram for the time sequence of inputs, outputs and internal states.

The aim of the analysis is to determine the outputs and the next state of the circuit depending on the inputs and the present state.

Analysis Procedure:

1- Label the inputs and the outputs of each flip-flop with different symbols.

2- Obtain the input equation of each flip-flop in terms of the present state and input variables.

3- Substitute the equations found in step 2 into the corresponding flip-flop characteristic equation.

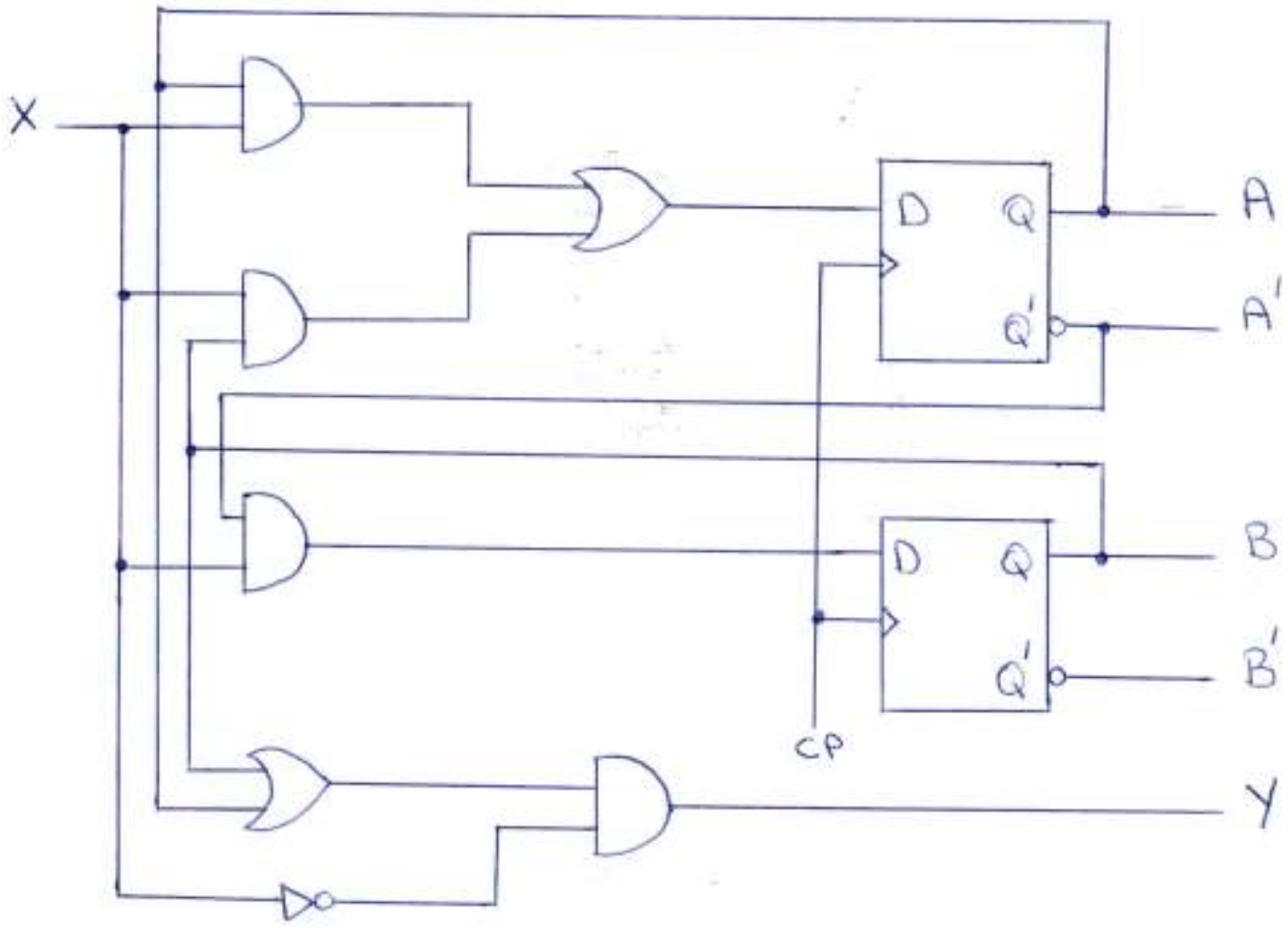
4- Find the output equations.

5- Draw the state table and the state diagram.

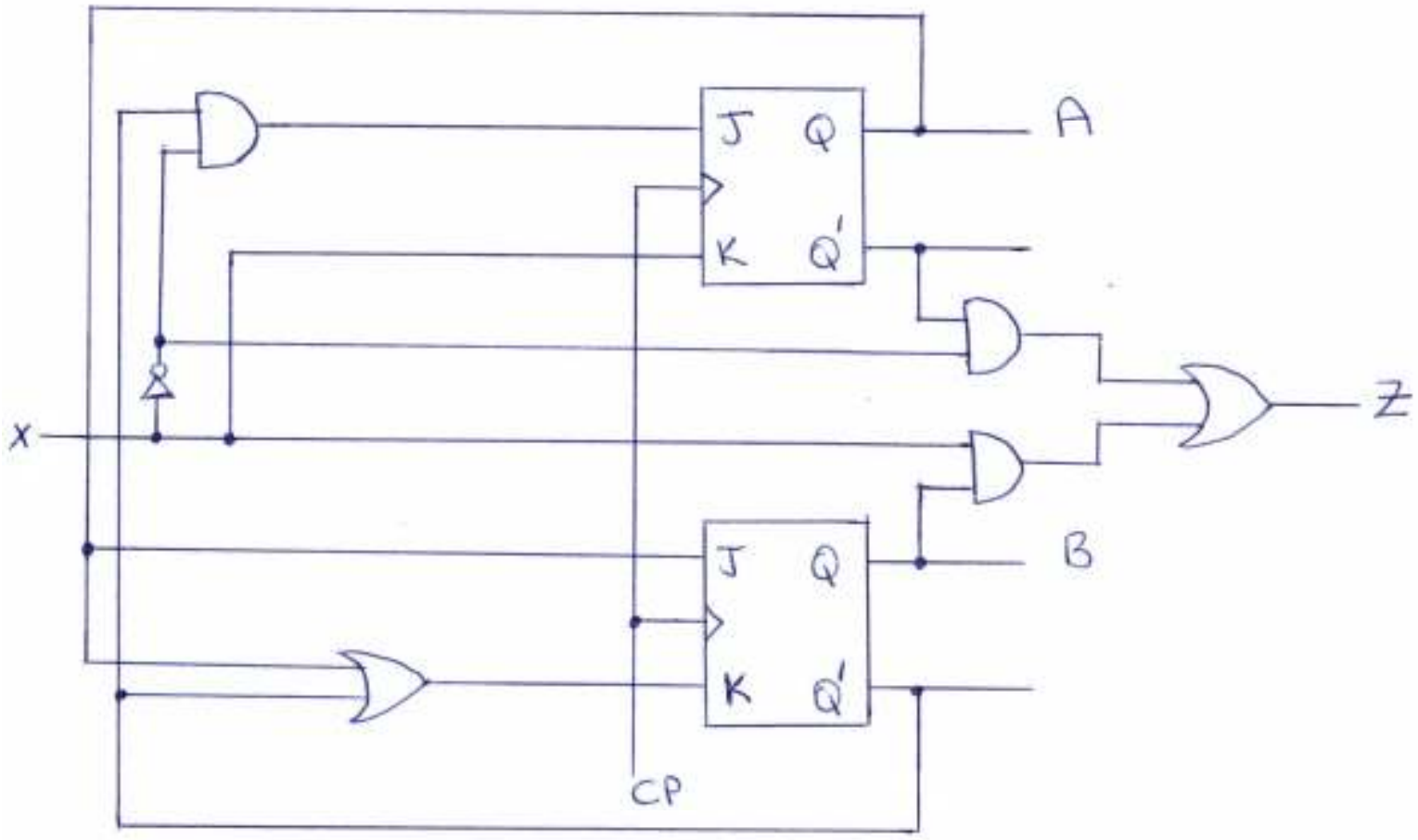
Remember; Flip-flop characteristic equations:

$$SR \Rightarrow Q(t+1) = S + R'Q$$
$$JK \Rightarrow Q(t+1) = JQ' + K'Q$$
$$T \Rightarrow Q(t+1) = TQ' + T'Q$$
$$D \Rightarrow Q(t+1) = D$$

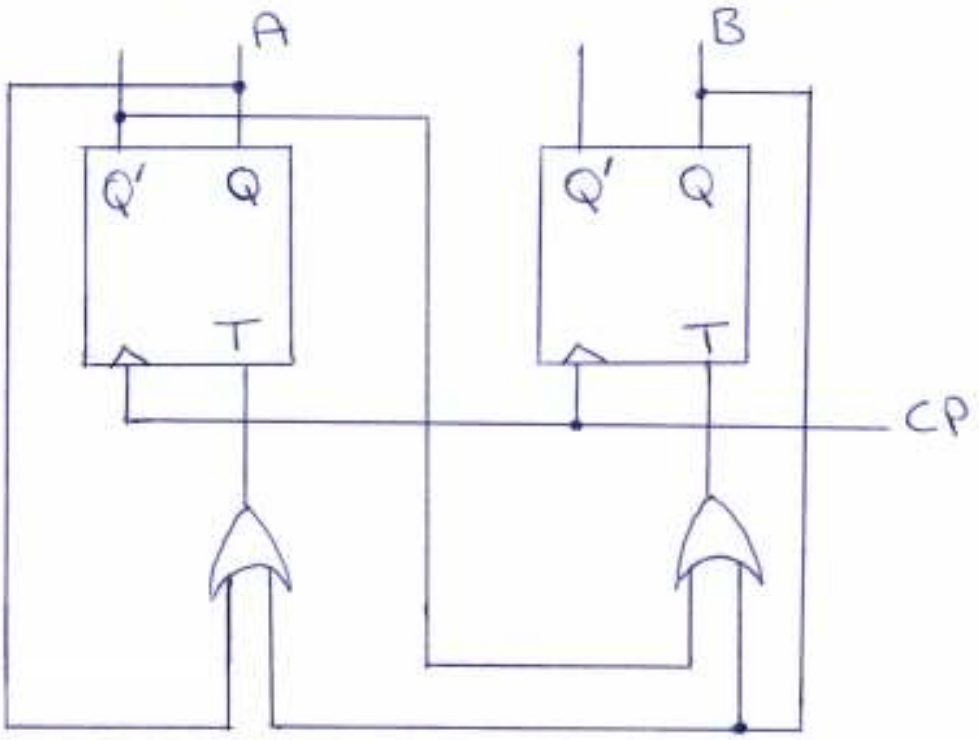
Example: Derive the state table and draw the state diagram of the following sequential circuit.



Example: Derive the state table and draw the state diagram of the following sequential circuit.

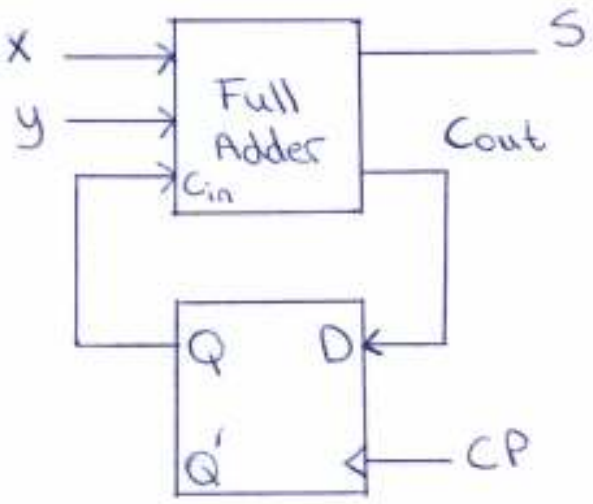


Example: Derive the state table and draw the state diagram of the following sequential circuit.



Example: Derive the state table and state diagram of the sequential circuit which has

- One flip-flop, Q
- Two inputs, x and y
- One output, S



* Design of Clocked Sequential Circuits

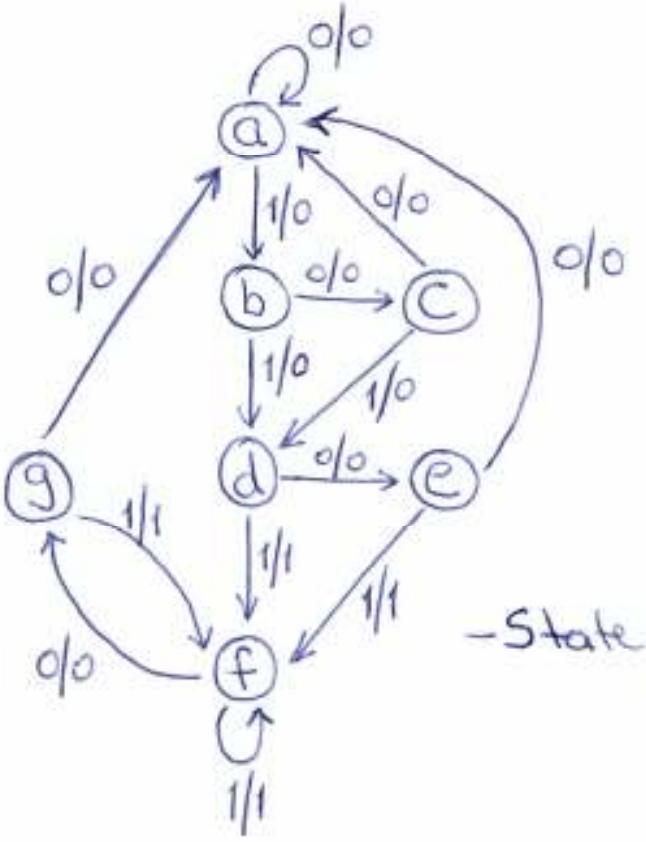
Two important concepts for the design of sequential circuits:

- State reduction
- Flip-flop excitation tables

State Reduction:

State reduction means minimizing the cost of the final circuit by minimizing the number of states in a state table without changing external input output requirements.

Example: Reduce the number of states of the sequential circuit specified in the following state diagram.



-State Diagram-

$$\#\text{ of states} = 2^{\#\text{ of flip-flops}}$$

$$7 \text{ states} \Rightarrow 2^{3 \text{ flip-flops}}$$

but $2^3 = 8$, so 7 states used, 1 state unused

- The state diagram of the sequential circuit has 7 states. So, the sequential circuit requires 3 flip-flops with other combinational logic for implementation.

- To reduce the number of states, the state table is needed:

Present State	Next State		Output	
	x=0	x=1	x=0	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 Table -

2 Algorithms → 1. Using Table
2 Algorithms → 2. Partitioning

Algorithm 1:

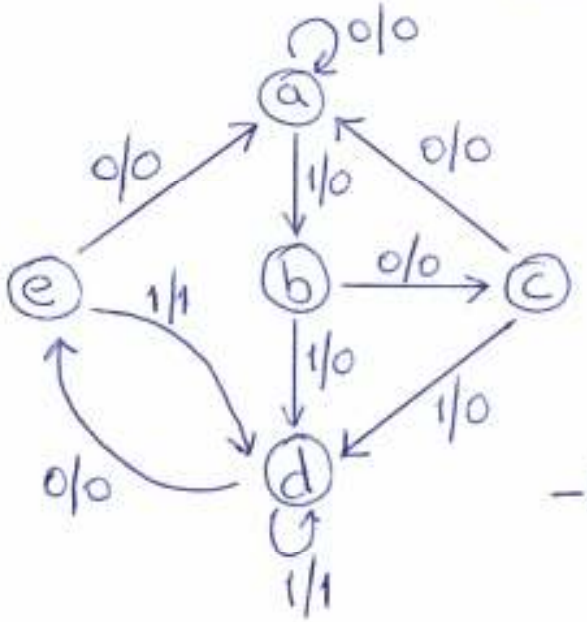
- Two states are said to be equivalent if, for each value of the input, they give exactly the same output and send the circuit either to the same state or to an equivalent state.
- When two states are equivalent, one of them can be removed.

The reduced state table will be as follows;

Present State	Next State		Output	
	x=0	x=1	x=0	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

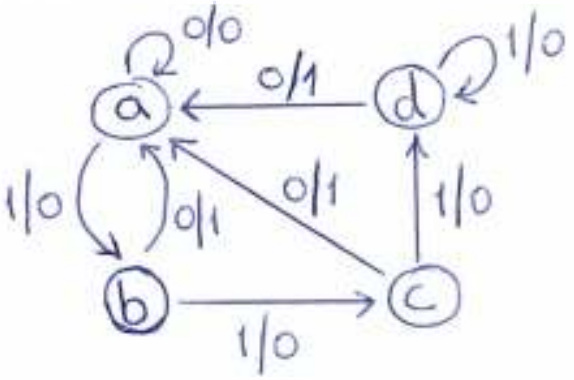
– Reduced State Table –

Now, the state diagram for the reduced state table consists of only 5 states.



– Reduced State Diagram –

Example: Reduce the number of states of the sequential circuit specified in the following state diagram.



- State Diagram -

Algorithm 2: Partitioning Algorithm

Consider the state table of the 1st Example on page 20:

Present State	Next State		Output	
	x=0	x=1	x=0	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 Table -

1- Partition the set of all states into subsets, such that all members of a set have identical rows in the state table. Under each state, record the number of set of which the next state is a member.

Note; According to the given state table, there are two kind of outputs: (0,0) and (0,1)

S_1 (output (0,0)) S_2 (output (0,1))

$P_1 = \{ \underset{1,1}{a}, \underset{1,2}{b}, \underset{1,2}{c} \}$ $\{ \underset{2,2}{d}, \underset{1,2}{e}, \underset{2,2}{f}, \underset{1,2}{g} \}$

2- Divide existing states so that all members of a new state set posses the same subscript.

$$P_2 = \{\underset{1,2}{a}\}^{S1}\ \{\underset{2,3}{b},\underset{1,3}{c}\}^{S2}\ \{\underset{4,3}{d},\underset{4,3}{f}\}^{S3}\ \{\underset{1,3}{e},\underset{1,3}{g}\}^{S4}$$

$$P_3 = \{\underset{1,2}{a}\}^{S1}\ \{\underset{3,4}{b}\}^{S2}\ \{\underset{1,4}{c}\}^{S3}\ \{\underset{5,4}{d},\underset{5,4}{f}\}^{S4}\ \{\underset{1,4}{e},\underset{1,4}{g}\}^{S5}$$

3- When no states can be formed, the algorithm terminates.

Present State	Next state		Output	
	x=0	x=1	x=0	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 Table -

State Assignment:

State assignment procedures are concerned with methods for assigning binary values to states.

Rules for state assignment:

1-(a) Check for rows of the state table which have identical next state entries in every column. Assign adjacent codes to such rows.

Example:

PS	NS		Output	
	x=0	x=1	x=0	x=1
A	D	F	1	0
B	D	F	0	1

⇒ Assign adjacent codes to A and B

(b) Check for rows of the state table which have the same next state entries but in different column order. Assign adjacent codes to such rows if the next state entries can be assigned adjacent codes.

Example:

PS	NS		Output	
	x=0	x=1	x=0	x=1
A	D	F	0	0
B	F	D	0	0

=> If D, F can be assigned adjacent codes, then assign adjacent codes to A, B.

(c) Rows with identical next state entries, but not in all columns, should be given adjacent codes. Rows having more identical columns should be given priority.

Example:

PS	NS			
	xy=00	xy=01	xy=11	xy=10
A	A	B	C	E
B	A	D	C	E
C	A	F	B	B

=> A and B have much more common next state entries. Assign adjacent codes to A and B.

2- Next state entries of a given row should be given adjacent codes.

3- Assignments should be made such that the output function is simplified.

Example:

PS	NS		Output	
	X=0	X=1	X=0	X=1
a	b	c	0	0
b	d	e	0	0
c	e	d	0	0
d	f	g	0	0
e	g	f	0	0
f	a	a	1	0
g	a	a	0	1

- Rule 1.(a): f, g are adjacent.
- Rule 1.(b): d, e are adjacent since f, g are adjacent.
- Rule 1.(b): b, c are adjacent since d, e are adjacent.

	00	01	11	10
0	a	b	d	f
1		c	e	g

a=000 e=111
b=001 f=010
c=101 g=110
d=011

PS	NS		Output	
	X=0	X=1	X=0	X=1
000	001	101	0	0
001	011	111	0	0
101	111	011	0	0
011	010	110	0	0
111	110	010	0	0
010	000	000	1	0
110	000	000	0	1

Flip-Flop Excitation Tables:

During the design process, we usually know the transition from present state to next state and wish to find the flip-flop input conditions that will cause the required transition. For this reason, we need the excitation table.

RS FF: $Q(t+1) = S + R'Q$

S	R	Q(t+1)	
0	0	Q(t)	No change
0	1	0	Reset
1	0	1	Set
1	1	?	Undefined

– Characteristic Table –

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

– Excitation Table –

JK FF: $Q(t+1) = JQ' + K'Q$

J	K	Q(t+1)	
0	0	Q(t)	No change
0	1	0	Reset
1	0	1	Set
1	1	Q'(t)	Toggle

– Characteristic Table

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

– Excitation Table –

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

D	Q(t+1)
0	0
1	1

– Characteristic Table

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

– Excitation Table –

T FF: $Q(t+1) = TQ' + T'Q$

T	Q(t+1)	
0	Q(t)	No change
1	Q'(t)	Toggle

– Characteristic Table

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

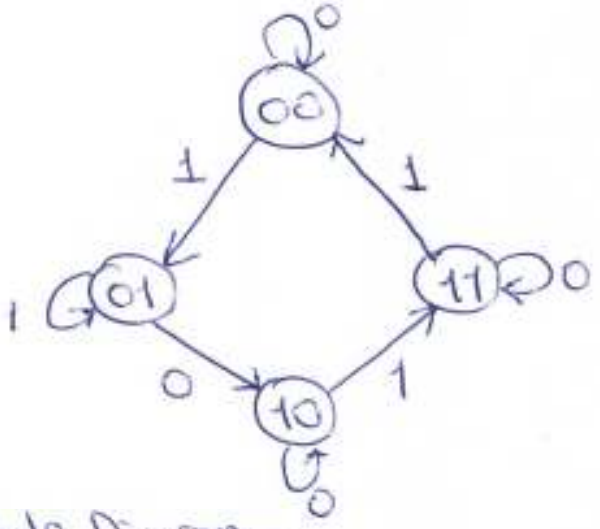
– Excitation Table –

Design Procedure:

The design of a sequential circuit starts from a set of specifications and culminates in a logic diagram or a list of Boolean functions from which the logic diagram can be obtained.

1- The word description of the circuit behaviour is stated. This may be accompanied by a state diagram or a timing diagram.
2- Obtain the state table from the given information.
3- Reduce the number of states (if possible).
4- Determine the number of flip-flops needed and assign binary values to each state.
5- Choose the type of flip-flop to be used.
6- From the state table, derive the excitation table which contains the flip-flop inputs and combinational circuit outputs.
7- Using the map or any other simplification method, derive the circuit output functions and the flip-flop input functions.
8- Draw the logic diagram.

Example: Design a clocked sequential circuit whose state diagram is given below using JK flip flops.



– State Diagram –

Present State		Next State	
		x=0	x=1
A	B	A B	A B
0	0	0 0	0 1
0	1	1 0	0 1
1	0	1 0	1 1
1	1	1 1	0 0

– State Table –
(already minimized)

* The state diagram consists of four states with binary values already assigned.
* Directed lines are marked with a single binary digit without a slash ⟹ – one input variable
  – no output variables
* 4 states ⟹ 2 FFs

$$\#\text{ of states} = 2^{\#\text{ of FFs (flip-flops)}}$$

$$4 = 2^{2} \rightarrow \#\text{ of FFs}$$

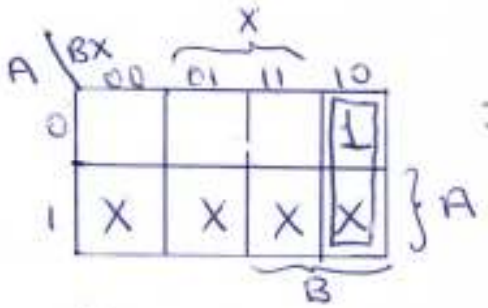
*– 2 FFs are designated as A and B.
– Input variable is designated as X.

– The excitation table of the circuit can be derived from the state table and JK excitation table.

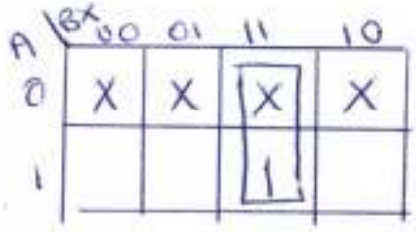
Present State		Input	Next State		Flip-Flop Inputs			
A	B	X	A	B	JA	KA	JB	KB
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	1	0	1	X	X	1
0	1	1	0	1	0	X	X	0
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

— Excitation Table —

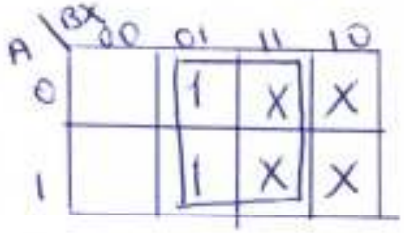
- The simplified Boolean functions can now be derived



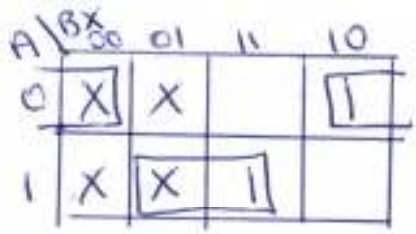
$JA = BX'$



$KA = BX$



$JB = X$

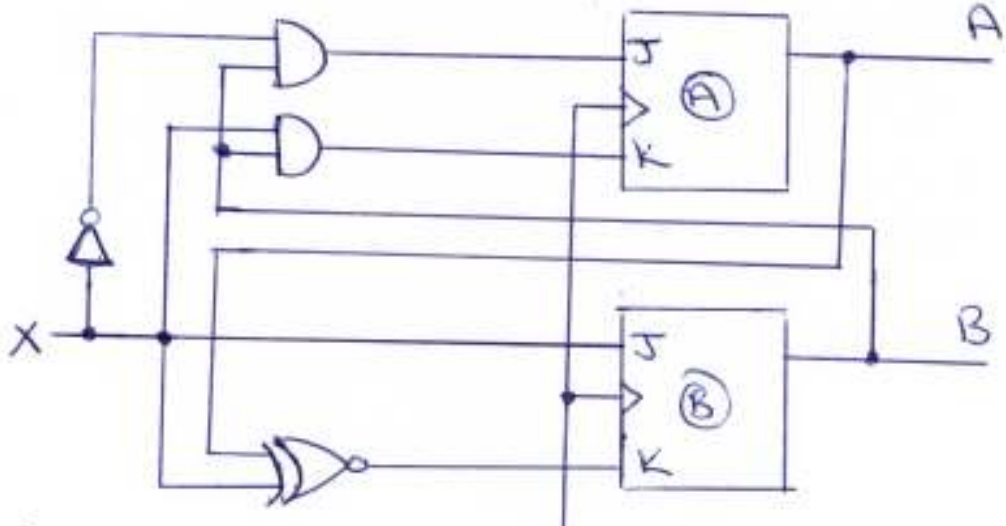


$KB = AX + A'X'$

$= (A \oplus X)'$

$= A \odot X$

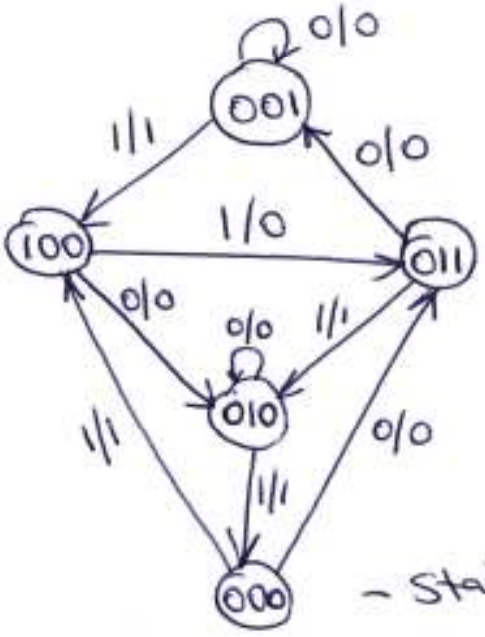
— Maps —



— Logic Diagram —

Example (Design with Unused States): A sequential circuit has three flip-flops, A, B, C; one input, x; and one output, y. The state diagram is shown below. Design the circuit: a) using D flip-flops
b) using JK flip-flops

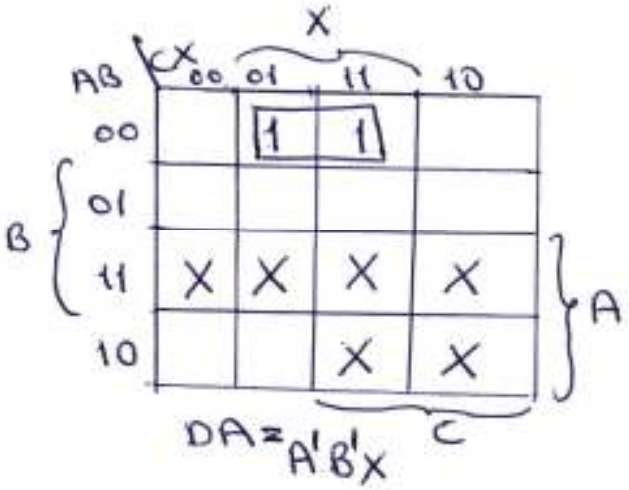
a)

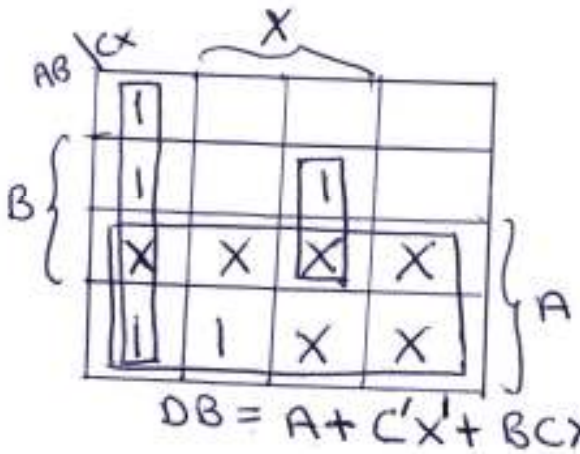


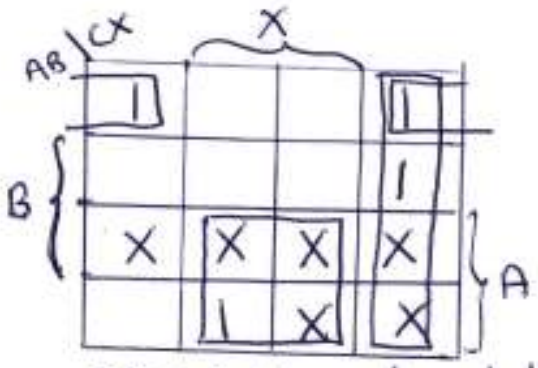
– State Diagram –

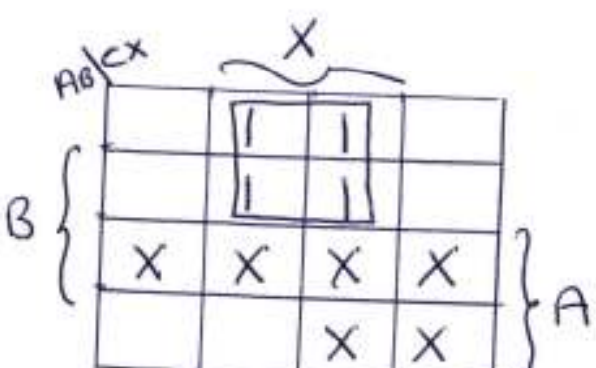
(Present State) PS		Input	(Next State) NS		Output
A	B C	x	A	B C	y
0	0 0	0	0	1 1	0
0	0 0	1	1	0 0	1
0	0 1	0	0	0 1	0
0	0 1	1	1	0 0	1
0	1 0	0	0	1 0	0
0	1 0	1	0	0 0	1
0	1 1	0	0	0 1	0
0	1 1	1	0	1 0	1
1	0 0	0	0	1 0	0
1	0 0	1	0	1 1	0

– State Table ≡ Excitation Table – for D FF

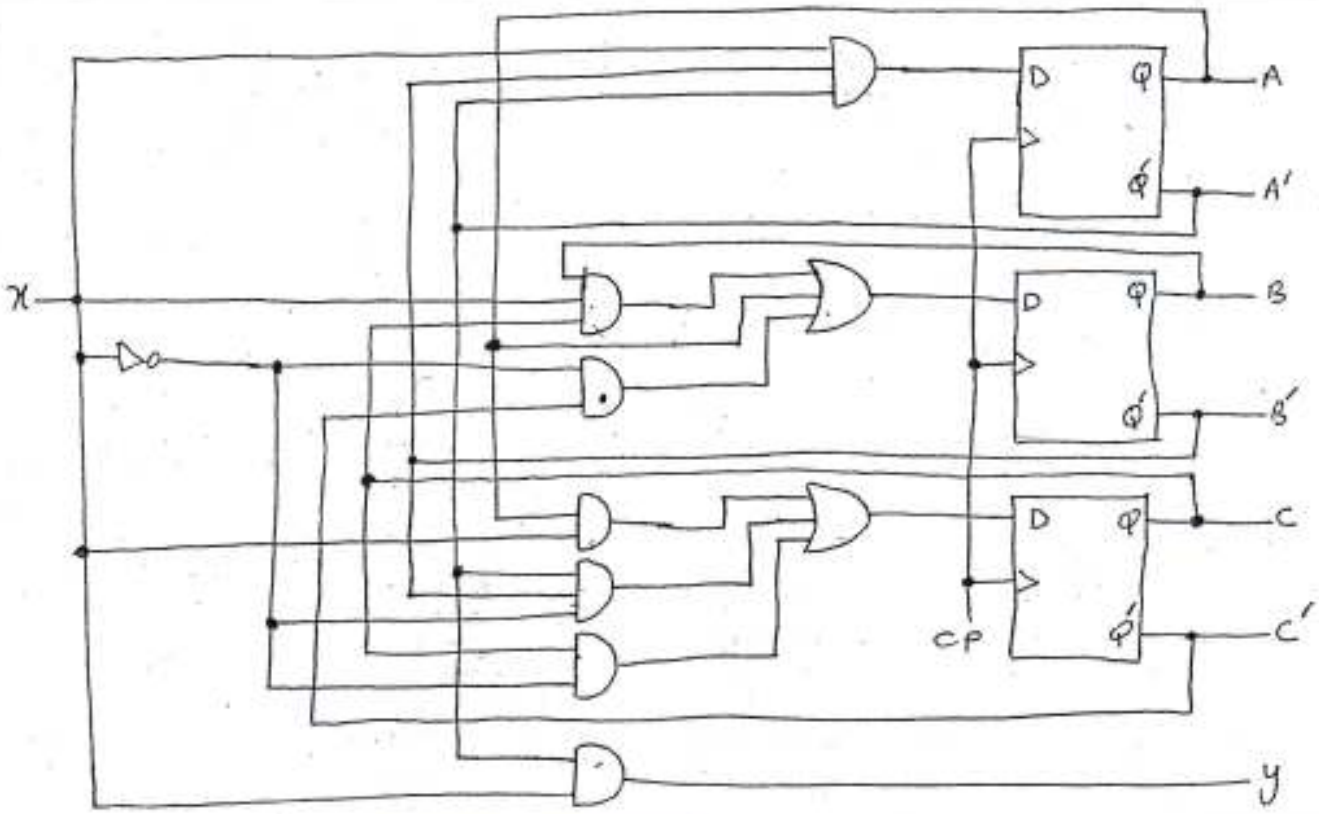








– Maps



—Logic Diagram—

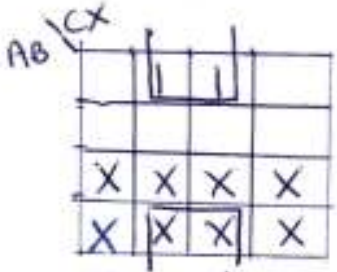
* Checking the design with unused states:

- When power is first turned on in a digital system, we don't know in which state the flip-flop will settle (start). What happens if the circuit start with an unused state or during proper operation, because of some noise, the circuit may go into one of its unused states.
- To ensure that a return to normal operation is possible without resetting the entire system, the next-state behaviour for the unused states may be specified. The next state of the unused states should be selected such that one of the valid states is reached within a few clock cycles, regardless of input values.

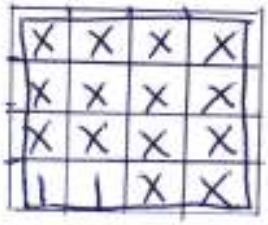
b)

PS	Input	NS	Output	Flip-Flop Inputs					
A B C	X	A B C	y	JA	KA	JB	KB	JC	KC
0 0 0	0	0 1 1	0	0	X	1	X	1	X
0 0 0	1	1 0 0	1	1	X	0	X	0	X
0 0 1	0	0 0 1	0	0	X	0	X	X	0
0 0 1	1	1 0 0	1	1	X	0	X	X	1
0 1 0	0	0 1 0	0	0	X	X	0	0	X
0 1 0	1	0 0 0	1	0	X	X	1	0	X
0 1 1	0	0 0 1	0	0	X	X	1	X	0
0 1 1	1	0 1 0	1	0	X	X	0	X	1
1 0 0	0	0 1 0	0	X	1	1	X	0	X
1 0 0	1	0 1 1	0	X	1	1	X	1	X

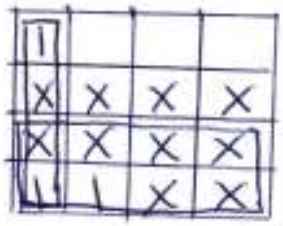
– Excitation Table –



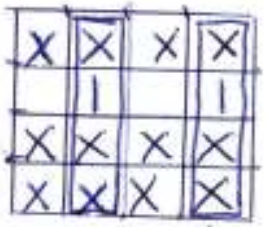
$JA = B'X$



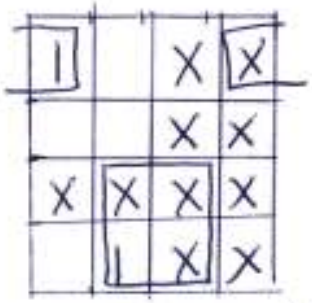
$KA = 1$



$JB = A + C'X'$

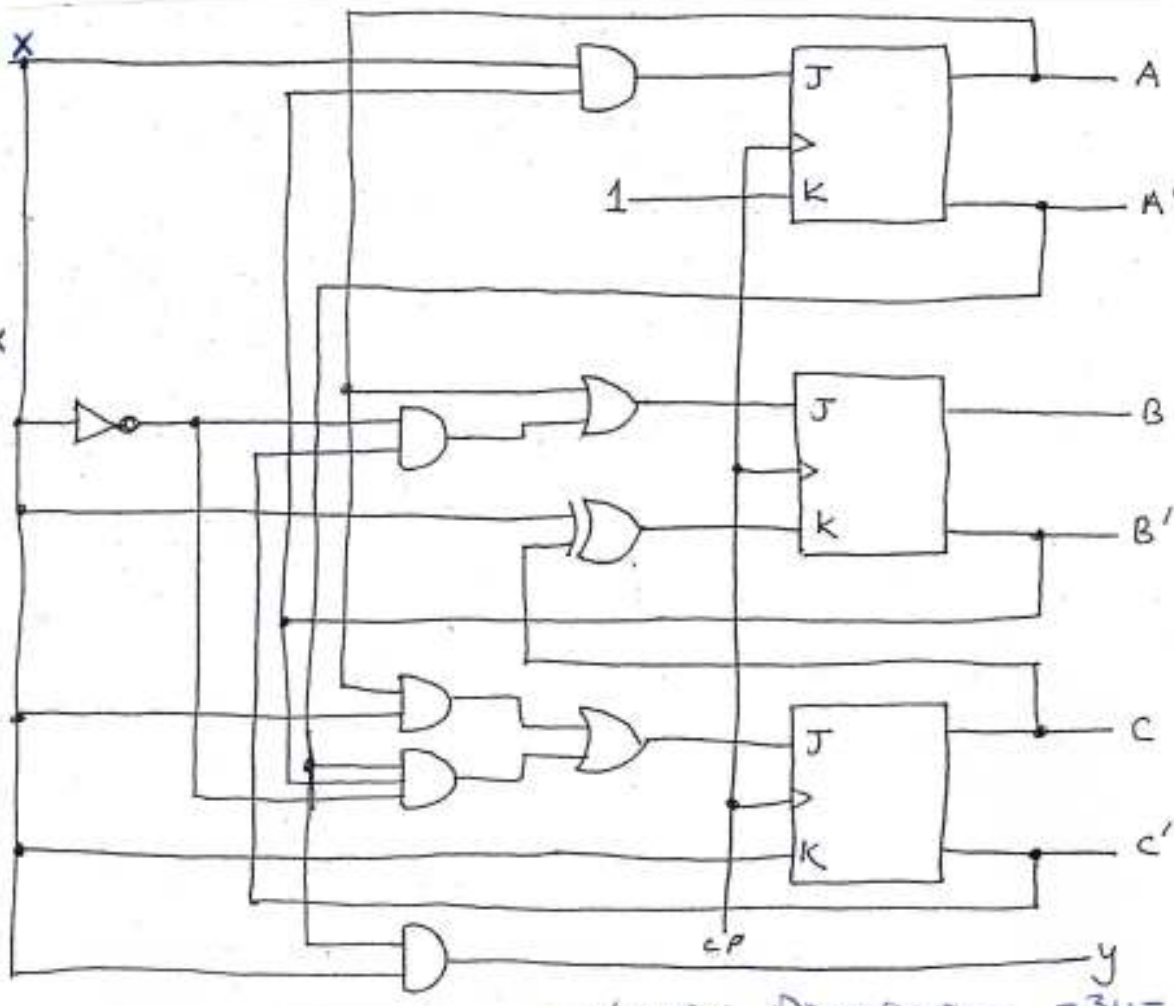


$KB = CX' + C'X = C \oplus X$



$JC = AX + A'B'x$

$KC = X$



– Logic Diagram –