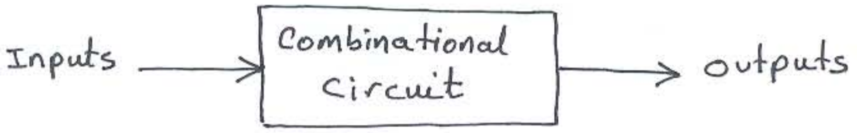
SYNCHRONOUS SEQUENTIAL LOGIC

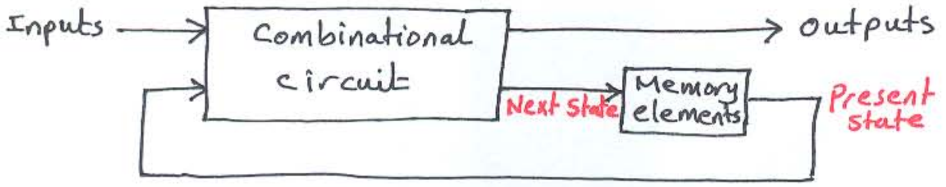
The digital circuits are classified into two types, "Combinational" and " sequential".

The Logic circuits considered in CMPE 223 were combinational.



In a Combinational circuit, the outputs depend only on the current inputs. i.e outputs = f (Inputs).

A sequential Logic circuit is one whose outputs depend not only on its current inputs, but also on the past sequence of inputs, which determine the circuit "internal states". A block diagram of a sequential circuit is shown below:



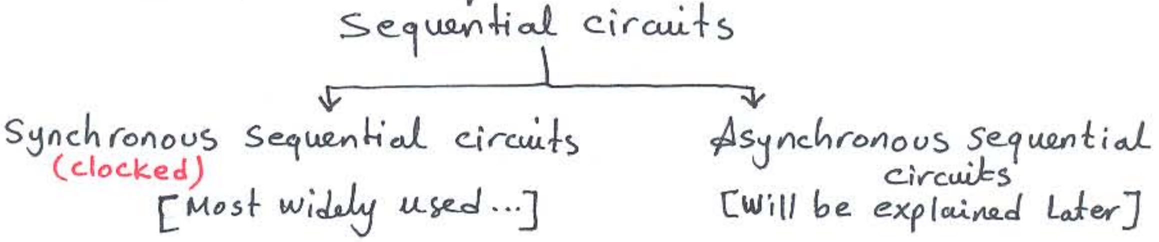
⇒ It consists of a combinational circuit and memory elements, connected in a feedback path. The memory elements are devices capable of storing binary information within them. The binary information stored in the memory elements at any given time defines the state of the sequential circuit.

∴ outputs = f (external inputs ; present state)

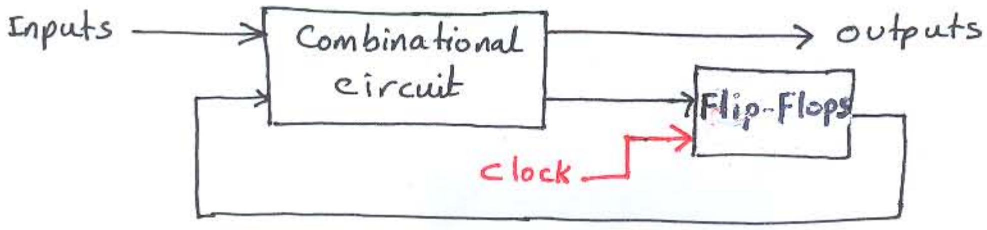
Next state = f (external inputs, Present state)

Thus, a sequential circuit is specified by a time sequence of inputs, outputs, and internal states.

There are two main types of sequential circuits, and their classification depends on the times at which their inputs are observed and their internal state changes:


Sequential circuits
Synchronous Sequential circuits
(clocked)
[Most widely used...]
Asynchronous Sequential circuits
[Will be explained Later]

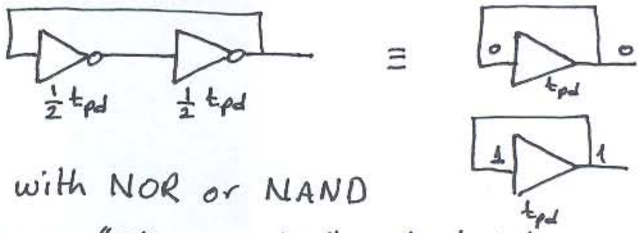
A synchronous circuit is one whose behavior can be defined from the knowledge of its signals at discrete instants of time. The memory elements can replace their present state with a new state in a controlled way. For this purpose clock pulses are used.



The behavior of an asynchronous sequential circuit can change at any instant of time.

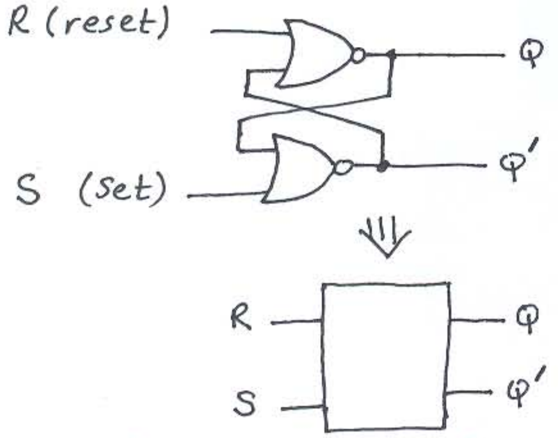
The most basic storage elements are Latches, from which flip-flops are usually constructed. Therefore a flip-flop is a device that stores either a "0" or a "1".

In fact, a more simple storage element is a buffer which permenantly stores a binary value. A simple buffer can be implemented with two inverters.



By replacing the inverters with NOR or NAND gates, we can store either a "0" or a "1" ⇒ Latch.

SR Latch with NOR gates: Remember: $(1+x)' \overset{\text{for sure}}{=} 0$; $(0+x)' \overset{\text{depend on } x!!}{=} x'$ Don't consider!



S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after S=1, R=0) i.e No change
0	1	0	1	
0	0	0	1	(after S=0, R=1) i.e No change
1	1	0	0	!! undefined.

Truth table

SR Latch with NAND gates: Remember: $(0\cdot x)' \overset{\text{for sure}}{=} 1$; $(1\cdot x)' = x'$

- Draw the circuit.
- Derive the Truth table.

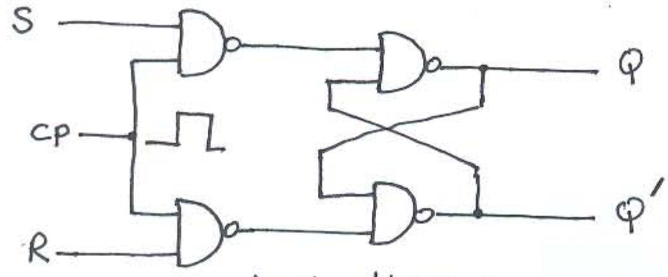
The operation of the basic NOR and NAND latches can be modified by providing an additional control input that determines when the state of the latch can be changed ⇒ flip-flops.

Flip-flops:

The memory elements used in clocked sequential circuits are called flip-flops. The state of a flip-flop is the value that it currently stores. The stored value can be only changed only at certain times determined by the "clock" input.

There are different types of flip-flops:

RS Flip-flop:

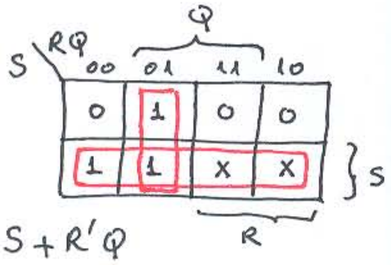


Logic diagram

Present state (PS) — Next state (NS)

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

(Characteristic) Truth table

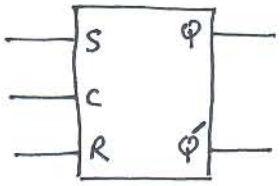


S \ RQ	00	01	11	10
0	0	1	0	0
1	1	1	X	X

(SR = 0)

$Q(t+1) = S + R'Q$

↑ Characteristic equation



Graphic symbol

An unpredictable state occurs when CP = 1 and both S and R are equal to 1.

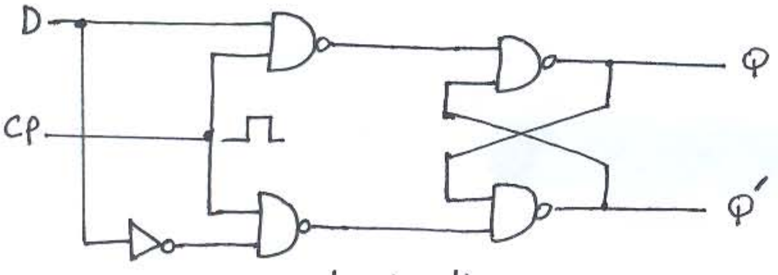
The truth table can be put in another form which is called the characteristic or functional table:

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

← Characteristic table of the RS flip-flop

D Flip-Flop:

One way to eliminate the undesirable condition of the undefined state of the RS flip-flop is to ensure that inputs S and R are never equal to 1 at the same time. This is possible by applying the complement of S to the R input, leading to the D flip-flop:

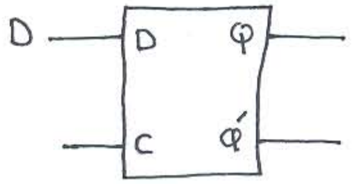


Logic diagram

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

Truth table

$Q(t+1) = D$ ↲ characteristic equation



Graphic symbol

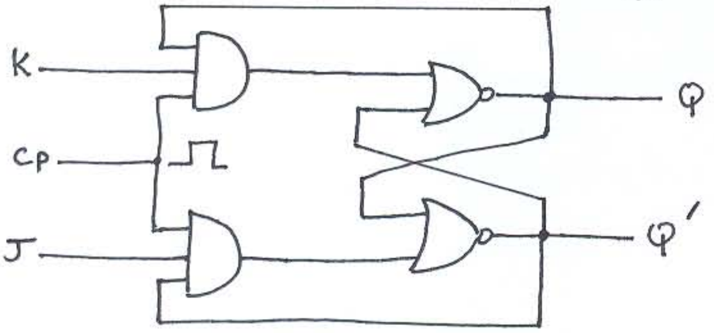
D	Q(t+1)
0	0
1	1

Characteristic table

As long as $CP=0$, the circuit cannot change state. when $CP=1$, the Q output follows the D input (i.e $Q=0$ if $D=0$; and $Q=1$ if $D=1$).

JK Flip-Flop:

A JK flip-flop is a refinement of the RS flip-flop to eliminate the undefined state of the RS type.



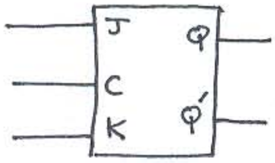
Logic diagram

		PS	NS	
J(t)	K(t)	Q(t)	Q(t+1)	
0	0	0	0	No change
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	1	Toggle
1	1	1	0	

Truth taple

J \ KQ	00	01	11	10	
		Q	Q		
0	0	1	0	0	
1	1	1	0	1	J
			K	K	

$Q(t+1) = JQ' + K'Q$ ↑ characteristic equation



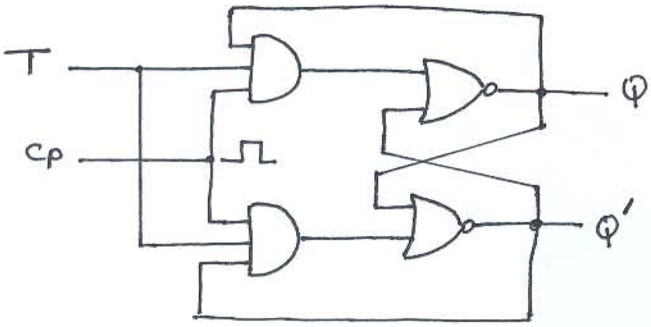
Graphic symbol

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

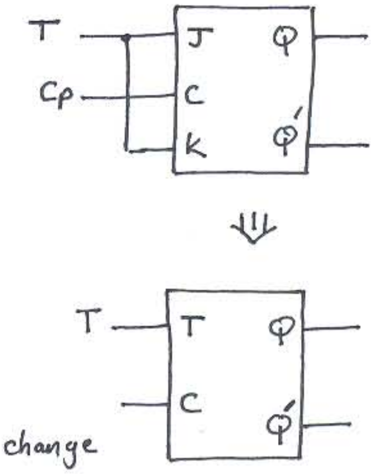
characteristic table

T Flip-Flop:

The T flip-flop is a single-input version of the JK flip-flop. It is obtained from the JK flip-flop when both inputs are tied together. Regardless of the present state, the T flip-flop complements its output when cp = 1.



Logic diagram



Graphic symbol

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

Truth table

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

characteristic table

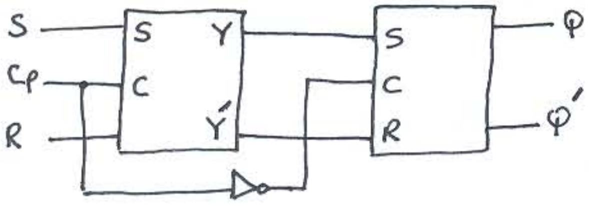
$\Rightarrow Q(t+1) = TQ' + T'Q$ ↲ characteristic equation.

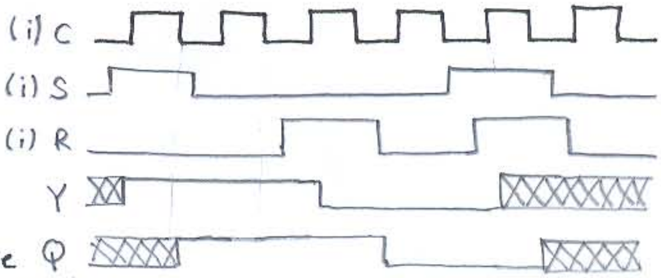
Note: you can design and define your own flip-flop.

Go To Page 5'

MASTER-SLAVE FLIP-FLOPS:

The master-Slave flip-flop consists of two latches and an inverter.
A master-slave SR flip-flop:

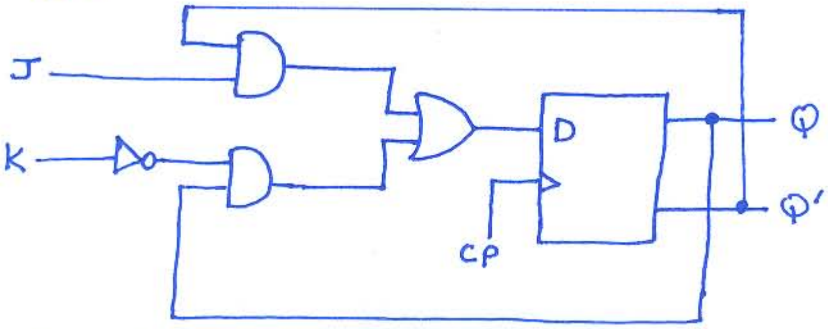




when the input clock c is 0, the slave latch is enabled, and its output Q is equal to the Master output Y. when input clock is 1, the values on S and R control the value stored in the master Latch Y.

Go To 5''

Note: a JK flip-flop can be implemented using a D FF and some logic.

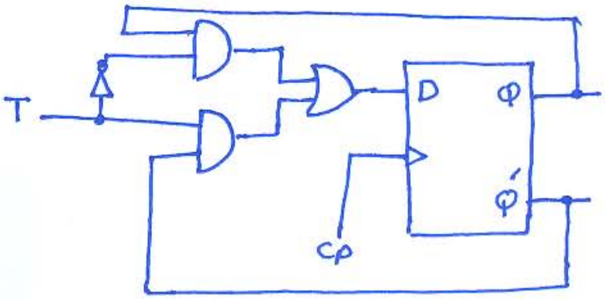


Remember:

D	Q(t+1)
0	0
1	1

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

Also, a T FF can be implemented using a D FF and some logic.

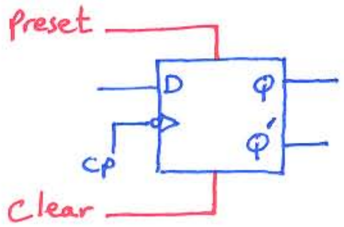


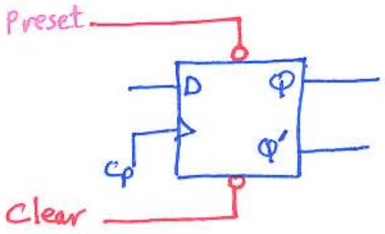
Configurable FF's:

For some circuits, one type of FF may lead to a more efficient implementation than another type. In general purpose chips like PLDs, the FFs are sometimes configurable. For example in MAX 7000 CPLDs, the FFs can be configured as D or T-type.

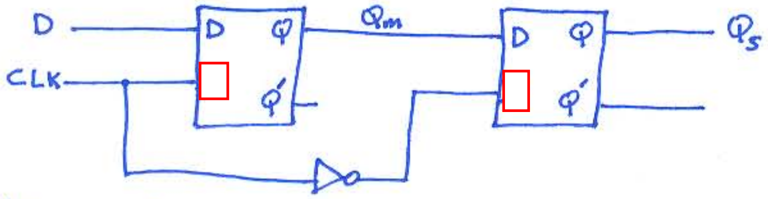
Flip-Flops with Preset and Clear inputs:

In some circuits, it is sometimes necessary to force the circuit into a known initial state. For this purpose FFs are provided with two extra inputs Preset and Clear, so that when activated the FF will go to state 1 and 0 respectively. These inputs are asynchronous and they may be active high or active low.

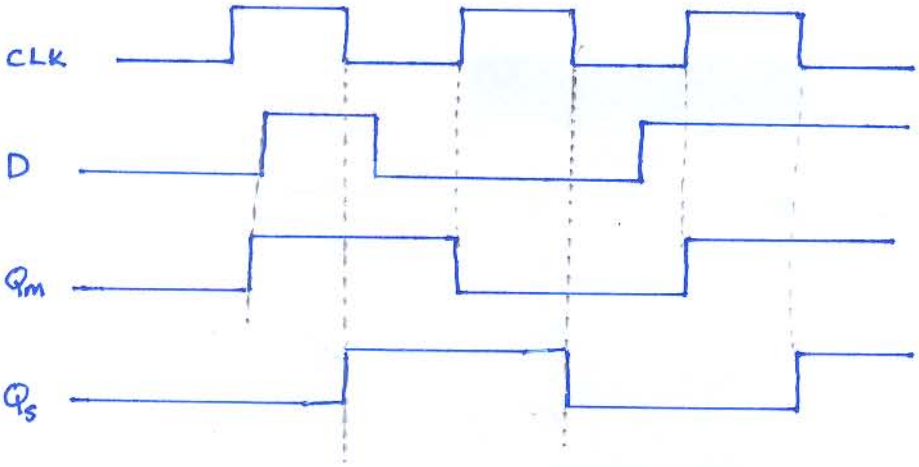




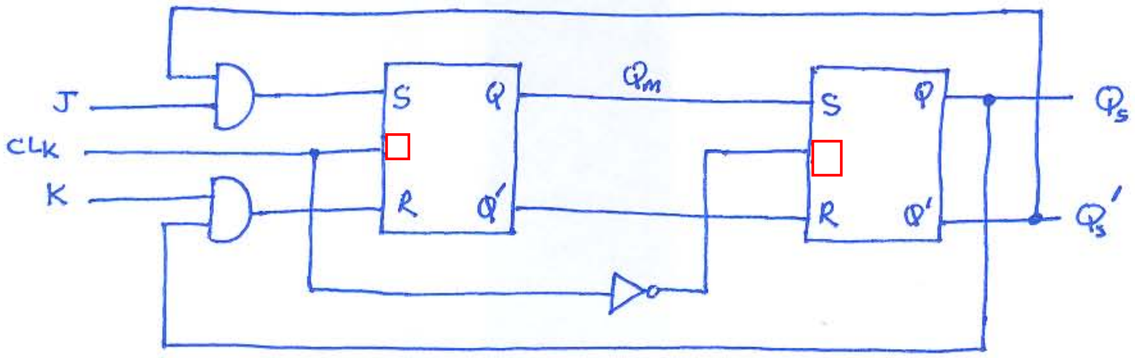
Example: Master-slave D FF:



The timing diagram is:



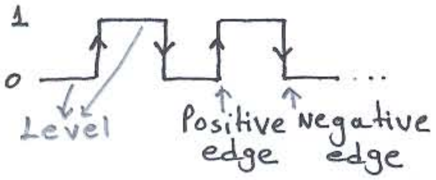
Example: Master-slave JK flip-flop



Draw the corresponding timing diagram.

TRIGGERING OF FLIP-FLOPS:

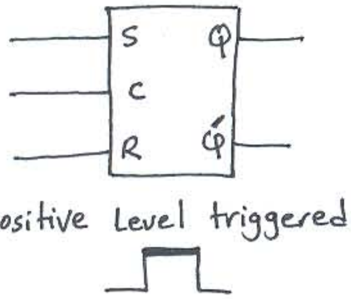
The state of a flip-flop is switched by a momentary change in the clock pulse input signal. This momentary change is called a "trigger". The clock pulse alternates between "0" and "1".



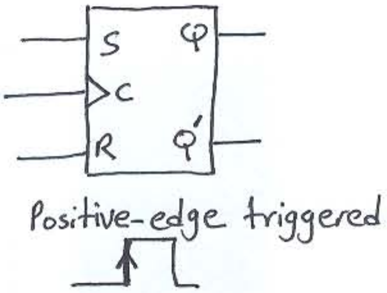
- It is not desirable that the flip-flops respond to the Level duration since it creates some problems.

- flip-flops must respond to either the positive edge or the negative edge only. i.e when the clock is 1 (or 0), the flip-flop is not responding to any change in inputs until the clock pulse returns to 0 (or 1).

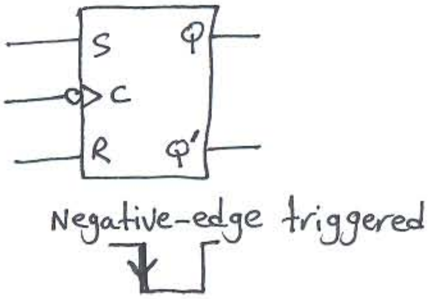
The graphic symbols are:



positive Level triggered



Positive-edge triggered



Negative-edge triggered

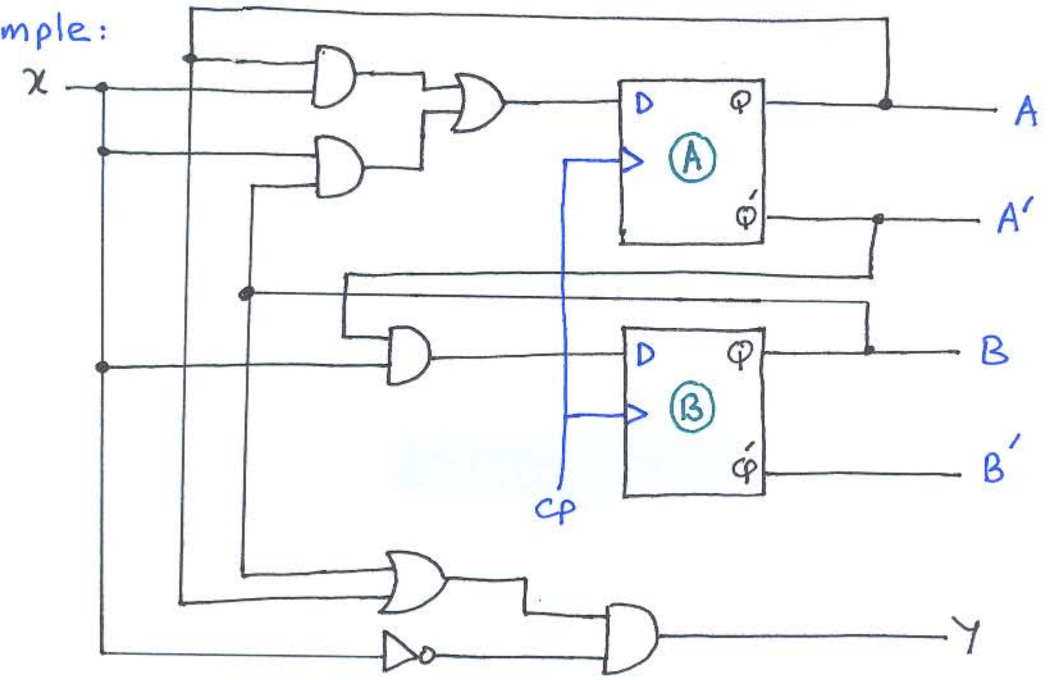
Remember, Q(t) refers to the present state prior to the application of a pulse and Q(t+1) is the next state one clock period later.

ANALYSIS OF CLOCKED SEQUENTIAL CIRCUITS:

The behavior of a sequential circuit is determined from the inputs, the outputs, and the state of its flip-flops. The analysis of a sequential circuit consists of obtaining a state table or a state diagram for the time sequence of inputs, outputs, and states.

The flip-flops may be of any type, and the logic diagram may or may not include combinational gates.

Example:



For the D flip-flop; $Q(t+1) = D$ ⇒ Next-state equations:

$A(t+1) = A(t)\,x(t) + B(t)\,x(t)$

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

For simplicity, omit the designation (t) ⇒

$$\left.\begin{aligned} A(t+1) &= Ax + Bx \\ B(t+1) &= A'x \end{aligned}\right\} \text{state equations.}$$

Similarly,

$y(t\ ) = [A(t) + B(t)]\,x'(t)$ or $y = (A+B)\,x'$

↑output equation

State table:

The time sequences of inputs, outputs, and flip-flop states can be listed in a state table as shown below. It has four sections: present state, input(s), next state, and possible output(s).

present state: of the flip-flops

input(s): values

next state: of the flip-flops obtained from the state equations

output(s): from output equation(s)

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	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

← State table

In general, a sequential circuit with m flip-flops and n-inputs needs 2^{m+n} rows in the state table! ⇒ sometimes t is more convenient to express the state table in another form shown below: (idea: # of rows = # of states)

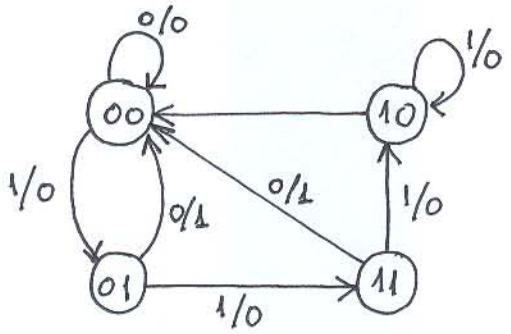
Present state		Next state				output	
		X = 0		X = 1		X = 0	X = 1
A	B	A	B	A	B	y	y
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

state table

State diagram:

The information available in a state table can be represented graphically in a state diagram. Each state is represented by a circle, and the transition between states is indicated by directed lines labeled with input(s)/output(s) binary values.

⇒ the state diagram of the previous example:

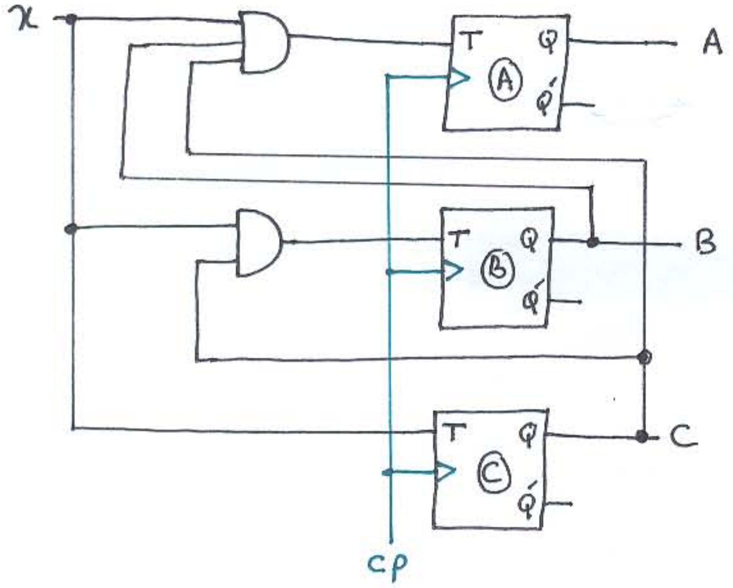


state diagram

General steps of the analysis:

1) obtain the input expression for each flip-flop in terms of the present state, and input variables. (i.e. flip-flop input functions)
2) Use the corresponding flip-flop characteristic table to determine the next state. (# of states = $2^{\text{\# of flip-flops}}$)
3) Find the output(s) from the output equation(s) [if any...].
4) Draw the resultant state diagram.

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



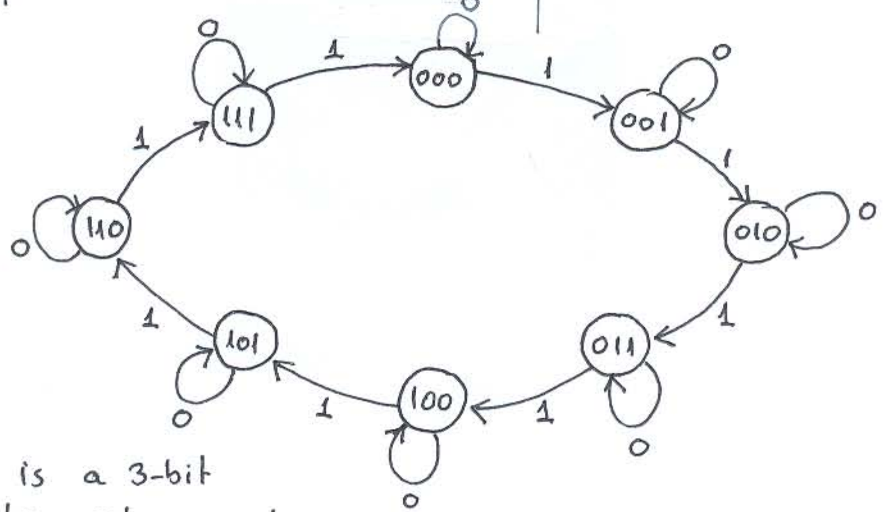
$$\left.\begin{aligned}T_A &= BCx\\ T_B &= Cx\\ T_C &= x\end{aligned}\right\}\text{flip-flop input equations}$$

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

T flip-flop characteristic table

Present state			Flip-flop inputs						Next state					
			x=0			x=1			x=0			x=1		
A	B	C	T_A	T_B	T_C	T_A	T_B	T_C	A	B	C	A	B	C
0	0	0	0	0	0	0	0	1	0	0	0	0	0	1
0	0	1	0	0	0	0	1	1	0	0	1	0	1	0
0	1	0	0	0	0	0	0	1	0	1	0	0	1	1
0	1	1	0	0	0	1	1	1	0	1	1	1	0	0
1	0	0	0	0	0	0	0	1	1	0	0	1	0	1
1	0	1	0	0	0	0	1	1	1	0	1	1	1	0
1	1	0	0	0	0	0	0	1	1	1	0	1	1	1
1	1	1	0	0	0	1	1	1	1	1	1	0	0	0

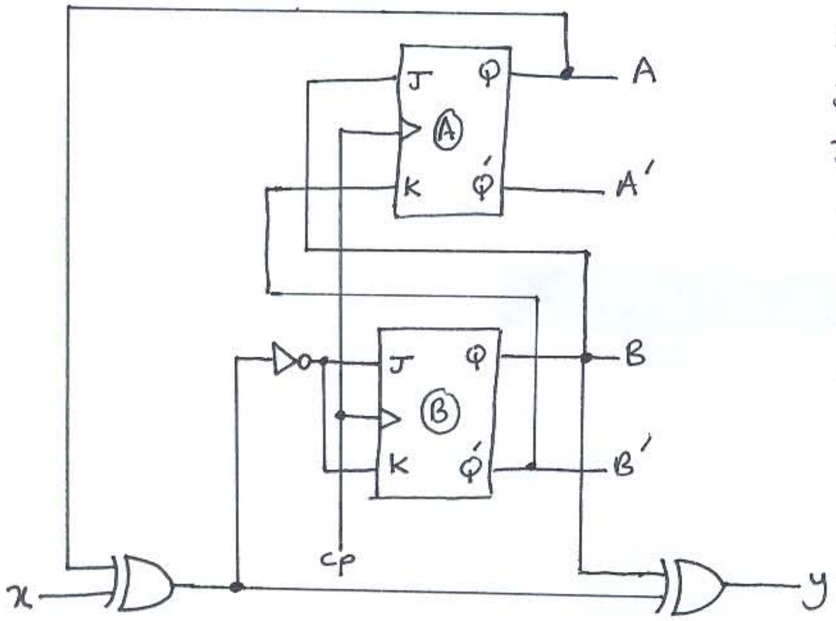
state table



state diagram

The circuit is a 3-bit binary counter. The counter Counts when the input x is 1 and the clock is applied.

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



flip-flops input equations:

$J_A = B \qquad K_A = B'$

$J_B = K_B = (A \oplus x)' = Ax + A'x'$

output equation:

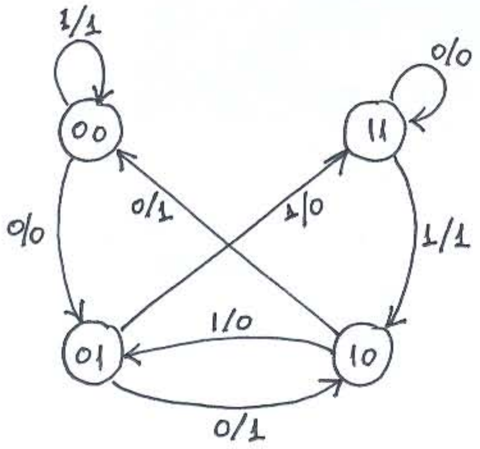
$y = (A \oplus x) \oplus B$

characteristic taple:

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

Present state		Input	Flip-flop inputs				Next state		output
A	B	x	J_A	K_A	J_B	K_B	A	B	y
0	0	0	0	1	1	1	0	1	0
0	0	1	0	1	0	0	0	0	1
0	1	0	1	0	1	1	1	0	1
0	1	1	1	0	0	0	1	1	0
1	0	0	0	1	0	0	0	0	1
1	0	1	0	1	1	1	0	1	0
1	1	0	1	0	0	0	1	1	0
1	1	1	1	0	1	1	1	0	1

State table



state diagram

DESIGN OF CLOCKED SEQUENTIAL CIRCUITS:

Before giving the full procedure for the design of synchronous sequential circuits, the designers (we) should know the following two important concepts:

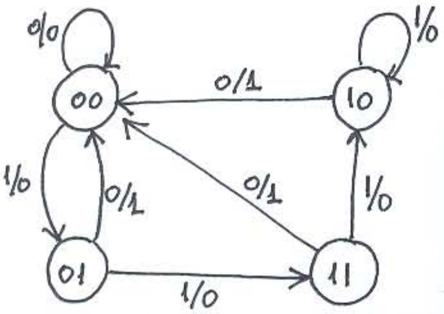
- State Reduction
- Flip-flops excitation tables.

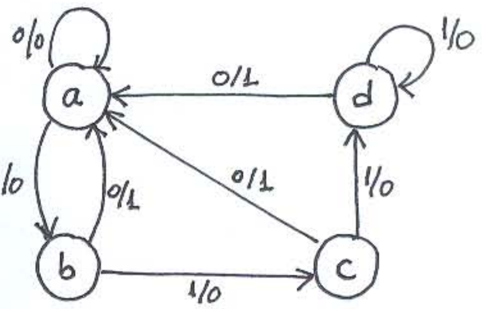
STATE REDUCTION:

state reduction means minimizing the cost of the final circuit, which can be achaived by minimizing the number of flip-flops.

Example:

Consider the following state diagram of a given sequential circuit.





For simplicity, the states may be denoted by Letter symbols ↑

state 00 ≡ a ; state 01 ≡ b ; state 11 ≡ c ; state 10 ≡ d

Since the circuit has 4 states ⇒ we need 2 FFs with other Combinational Logic for implementation.

For the above state diagram:

state:	a	a	b	a	b	a	b	c	a	b	a	a	b	c	d	d
input(x):	0	1	0	1	0	1	1	0	1	0	0	1	1	1	1	
output:	0	0	1	0	1	0	0	1	0	1	0	0	0	0	0	

Present state	Next state		Output		
	X=0	X=1	X=0	X=1	
a	a	b	0	0	
b	a	~~c~~ b	1	0	
~~c~~	a	~~d~~ c	1	0	b and c are equivalent
~~d~~	a	d	1	0	c and d are equivalent

The algorithm:

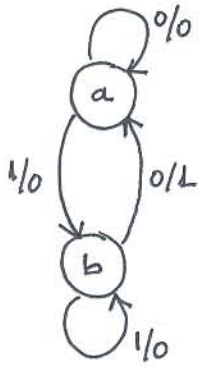
Two states are said to be equivalent if for each value of the input, they give exactly the same output and send the circuit either to the same state or to an equivalent state. When two states are equivalent, one of them can be removed.

⇒ for the above example, the reduced state table is:

Present state	Next state X=0	Next state X=1	Output X=0	Output X=1
a	a	b	0	0
b	a	b	1	0

⇒ Now, we have two states!

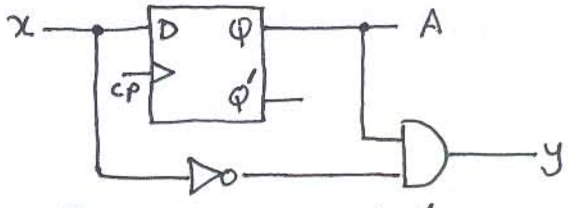
The state diagram:



state a ≡ 0

state b ≡ 1

⇒



$D_A = x$ $y = Ax'$

[will be explained Later...]

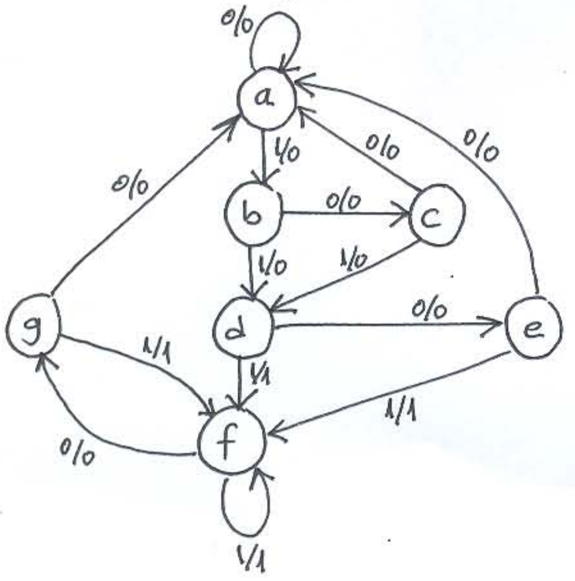
Check:

state :	a	a	b	a	b	a	b	b	a	b	a	a	b	b	b	b
input :	0	1	0	1	0	1	1	0	1	0	0	1	1	1	1	
output :	0	0	1	0	1	0	0	1	0	1	0	0	0	0	0	

Note: Sometimes, reduction of states does not Lead to a saving in the number of flip-flops or the number of gates!

Example:

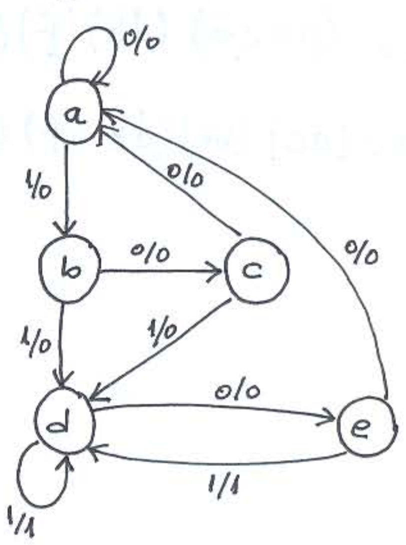
Reduce the number of states of the sequential circuit specified by the following state diagram.



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~~d	0	1
e	a	~~f~~d	0	1
f	~~g~~e	~~f~~d	0	1
g	a	f	0	1

The reduced state table and state diagram will be as:

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



Note that we couldn't reduce the number of FFs.

Example:

Reduce the number of states in the following state table.

Present state	Next state		Output	
	X=0	X=1	x=0	x=1
a	f	b	0	0
b	d	~~c~~a	0	0
~~c~~	f	~~e~~b	0	0
d	g	a	1	0
~~e~~	d	c	0	0
f	f	b	1	1
g	g	~~h~~d	0	1
~~h~~	g	a	1	0

⇒

Present state	Next state		Output	
	x=0	x=1	x=0	x=1
a	f	b	0	0
b	d	a	0	0
d	g	a	1	0
f	f	b	1	1
g	g	d	0	1

Reduced state table

GO TO page → (13')

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. The excitation table is used for that.

RS FF:

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

Characteristic table

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

Excitation table

STATE ASSIGNMENT:

State assignment procedures are concerned with methods for assigning binary values to states in such a way as to reduce the cost of the combinational circuit that drives the FFs.

Some common rules for state assignment are:

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

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

⇒ Assign adjacent codes to A and B.

b) Check for rows of the state table with 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.

Ps	NS $x=0$	NS $x=1$
A	D	F
B	F	D

If D, F can be assigned adjacent codes, then assigne adjacent codes to A, B.

2. Next state entries of a give row may be given adjacent codes.

Example:

PS	NS $x=0$	NS $x=1$	output $x=0$	output $x=1$
A	C	C	0	0
B	A	A	0	0
C	D	E	0	0
D	B	F	0	0
E	F	F	0	0
F	A	A	0	1

B, F must be adjacent (1.a)

A, B " " " (1.a)

Since B, F are adjacent ⇒ D, E must be adjacent (1.b)

⇓

	00	01	11	10
0	B	D	C	A
1	F	E	—	—

A, C can be adjacent because A, B are adjacent (Reverse of 1.b)

B, F adjacent; D, E adjacent ⇒ C, D can be adjacent (Reverse of 1.b).

∴ A ≡ 010 B ≡ 000

C ≡ 011 D ≡ 001

E ≡ 101 F ≡ 100

JK FF:

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:

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:

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

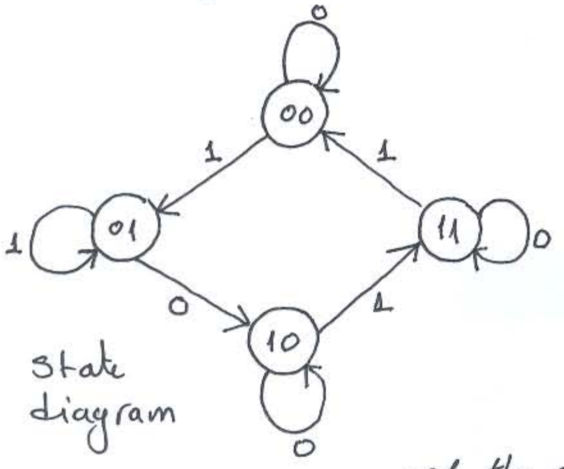
Excitation table

Design procedure:

1) The word description of the circuit behavior is stated, This may be accompanied by a state diagram or a timing diagram.
2) Obtain the state table.
3) Reduce the number of states (if possible)
4) Determine the number of flip-flops needed and assign a letter symbol to each flip-flop.
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 any simplification method, derive the flip-flop input functions and circuit output functions.
8) Draw the logic diagram.

Example:

Design a clocked sequential circuit whose state diagram is given bleow 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 excitation table of the circuit can be derived from the state table and JK excitation table.

Inputs of Comb. circuit					Outputs of Comb. circuit			
Present state		Input	Next state		Flip-flop inputs			
A	B	x	A	B	J_A	K_A	J_B	K_B
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

A \ Bx	00	01	11	10
0	0	0	0	1
1	X	X	X	X

$\Rightarrow J_A = Bx'$

A \ Bx	00	01	11	10
0	X	X	X	X
1			1	

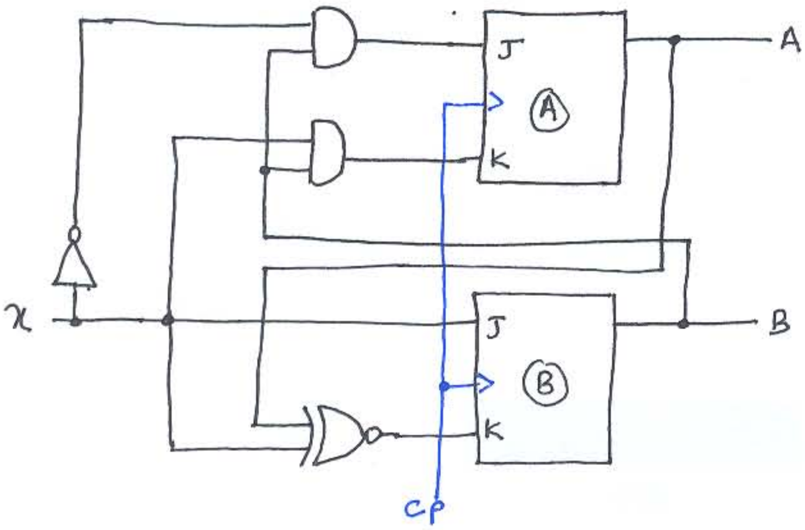
$\Rightarrow K_A = Bx$

A \ Bx	00	01	11	10
0		1	X	X
1		1	X	X

$\Rightarrow J_B = x$

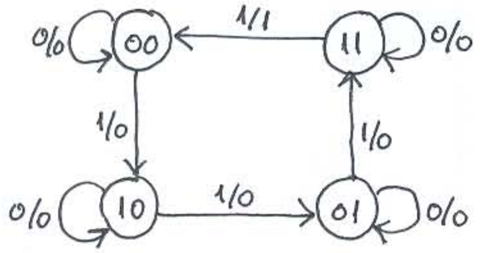
A \ Bx	00	01	11	10
0	X	X		1
1	X	X	1	

$\Rightarrow K_B = Ax + A'x' = (A \oplus x)' = A \odot x$



Example:

Using RS flip-flops, design the clocked sequential circuit whose state diagram is given below.



⇒

Present state		Next state				Output	
		$x=0$		$x=1$		$x=0$	$x=1$
A	B	A	B	A	B	y	y
0	0	0	0	1	0	0	0
0	1	0	1	1	1	0	0
1	0	1	0	0	1	0	0
1	1	1	1	0	0	0	1

State table

Present state		Input	Next state		Output	flip-flop inputs			
A	B	x	A	B	y	S_A	R_A	S_B	R_B
0	0	0	0	0	0	0	X	0	X
0	0	1	1	0	0	1	0	0	X
0	1	0	0	1	0	0	X	X	0
0	1	1	1	1	0	1	0	X	0
1	0	0	1	0	0	X	0	0	X
1	0	1	0	1	0	0	1	1	0
1	1	0	1	1	0	X	0	X	0
1	1	1	0	0	1	0	1	0	1

Excitation table.

Remember:

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

$y = ABx$

A \ Bx	00	01	11	10
0	0	1	1	0
1	X	0	0	X

$\Rightarrow S_A = A'x$

A \ Bx	00	01	11	10
0	X	0	0	X
1	0	1	1	0

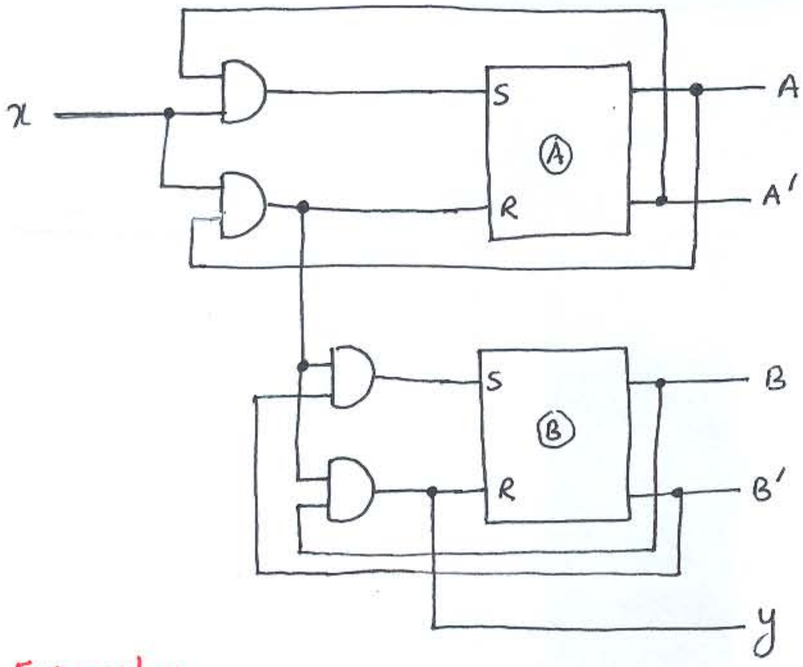
$\Rightarrow R_A = Ax$

A \ Bx	00	01	11	10
0	0	0	X	X
1	0	1	0	X

$\Rightarrow S_B = AxB'$

A \ Bx	00	01	11	10
0	X	X	0	0
1	X	0	1	0

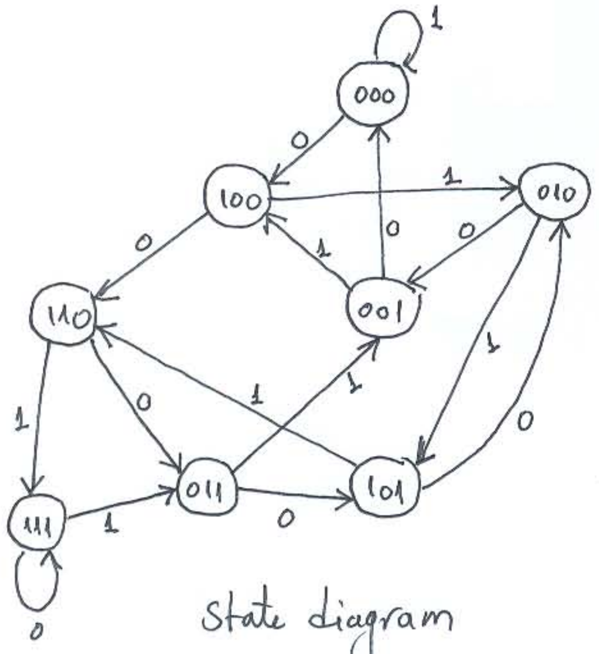
$\Rightarrow R_B = ABx$



Exercise: Repeat this example using JK FFs.

Example:

Design the sequential circuit whose state diagram is given below using D flip-flops.



State diagram

Present state	Next state	
	$x=0$	$x=1$
A B C	A B C	A B C
0 0 0	1 0 0	0 0 0
0 0 1	0 0 0	1 0 0
0 1 0	0 0 1	1 0 1
0 1 1	1 0 1	0 0 1
1 0 0	1 1 0	0 1 0
1 0 1	0 1 0	1 1 0
1 1 0	0 1 1	1 1 1
1 1 1	1 1 1	0 1 1

State table

Present state			Input	Next state ≡ flip-flop inputs		
A	B	C	x	A	B	C
0	0	0	0	1	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	1	0	1
0	1	1	0	1	0	1
0	1	1	1	0	0	1
1	0	0	0	1	1	0
1	0	0	1	0	1	0
1	0	1	0	0	1	0
1	0	1	1	1	1	0
1	1	0	0	0	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	0	1	1

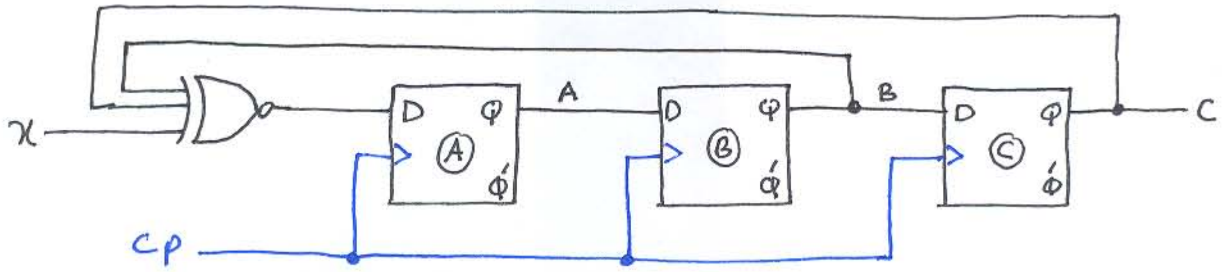
Note1: No need to put flip-flop inputs in the excitation table because for D FF next state is the same as its D input.

Note2: By intution, it is easy to see that

$D_B = A \quad ; \quad D_C = B$

AB \ Cx	00	01	11	10
00	1	0	1	0
01	0	1	0	1
11	0	1	0	1
10	1	0	1	0

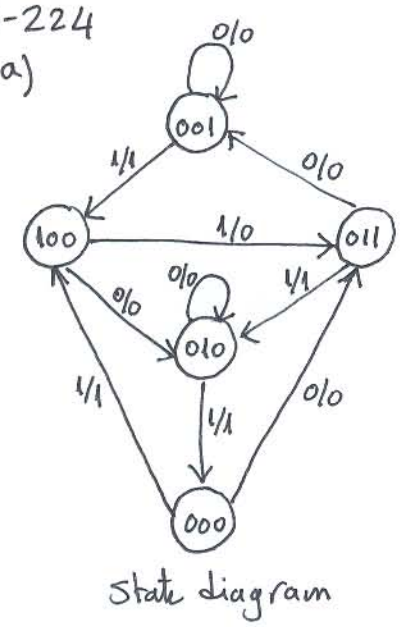
$$\Rightarrow D_A = (BC' + B'C)x + (BC + B'C')x'$$
$$= (B \oplus C)x + (B \oplus C)'x'$$
$$= (B \oplus C \oplus x)'$$



Example: Problem 6-22:

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

a)



State diagram

⇒

present state			input	Next state			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.

AB \ Cx			
0	1	1	0
0	0	0	0
X	X	X	X
0	0	X	X

⇒ $D_A = A'B'x$

AB \ Cx			
1	0	0	0
1	0	1	0
X	X	X	X
1	1	X	X

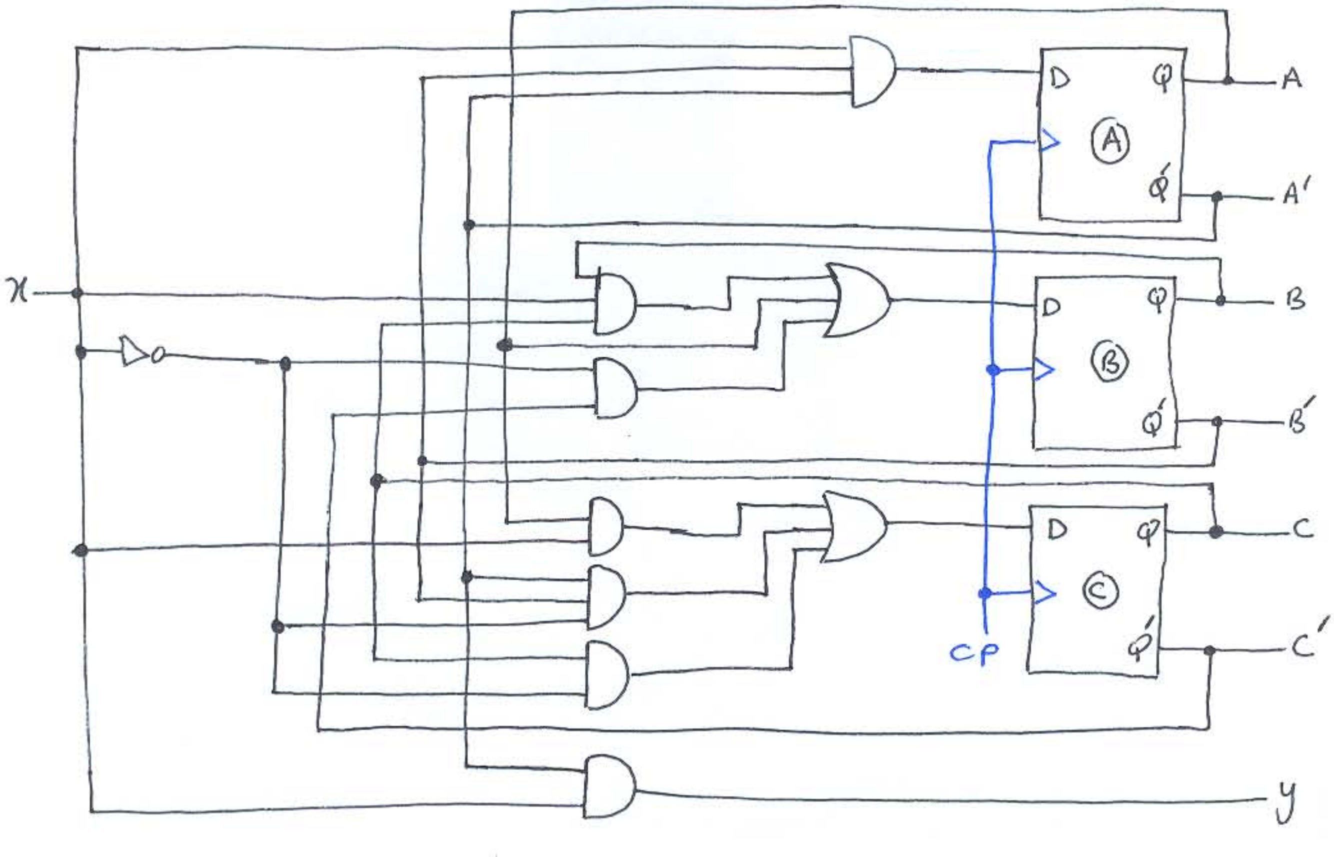
$D_B = A + C'x' + BCx$

AB \ Cx			
1	0	0	1
0	0	0	1
X	X	X	X
0	1	X	X

$D_C = Ax + Cx' + A'B'x'$

AB \ Cx			
0	1	1	0
0	1	1	0
X	X	X	X
0	0	X	X

$y = A'x$



b)

Present state			Input	Next state			output	Flip-flop inputs					
A	B	C	x	A	B	C	y	J_A	K_A	J_B	K_B	J_C	K_C
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

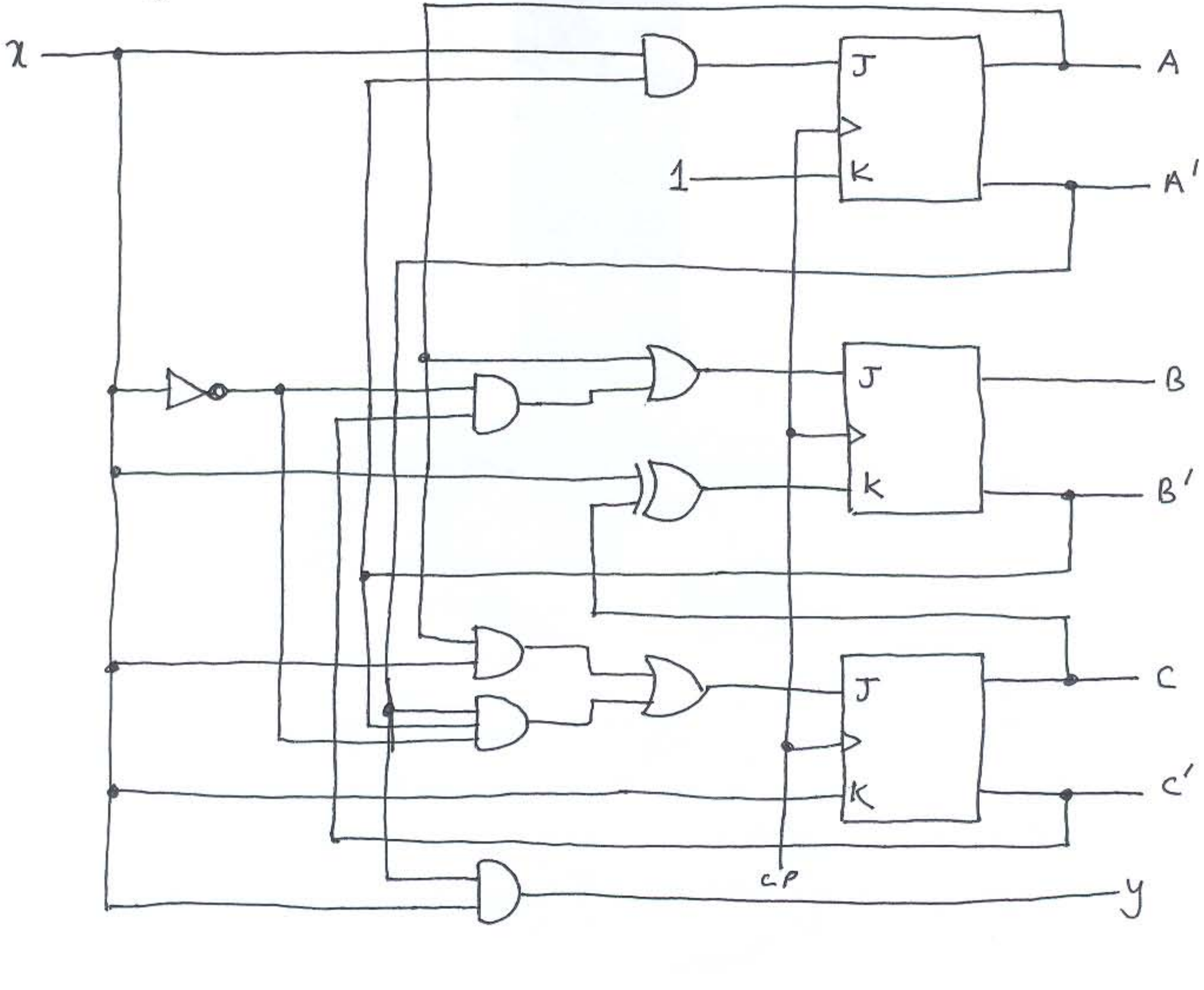
Excitation table.

Using the K-Map method (Exercise!), we have;

$J_A = B'x$ $\quad$ $J_B = A + C'x'$ $\quad$ $J_C = Ax + A'B'x'$

$K_A = 1$ $\quad$ $K_B = Cx' + C'x = C \oplus x$ $\quad$ $K_C = x$

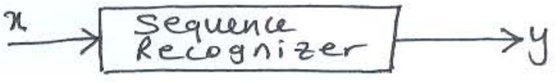
and $y = A'x$

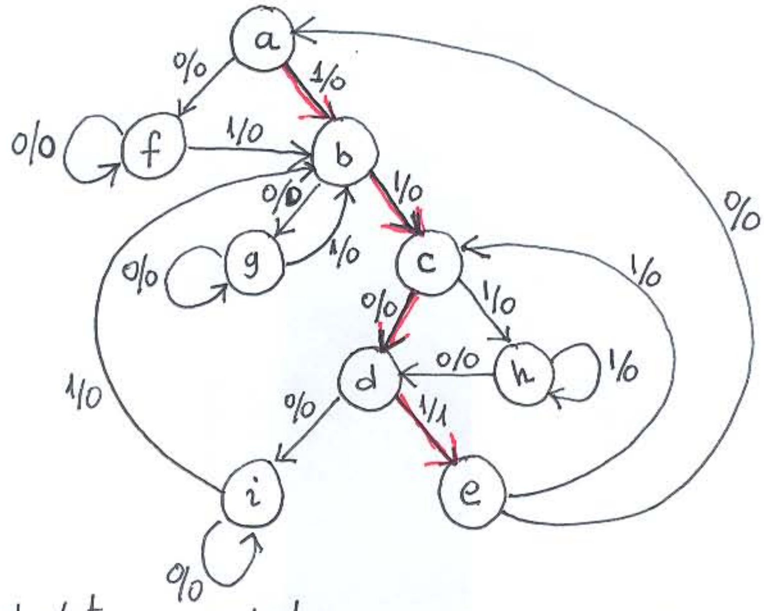


Example: (Sequence Recognizer)

Design Using FFs a synch. sequential circuit that will recognize the occurrence of the sequence of bits 1101 on an input x by making the output $y=1$ when the previous three inputs to the circuit were 110 and current input is a 1. Otherwise, $z=0$.

(Note that in the sequence 1101 : the occurrence order is 1 then 1, then 0, then 1. for example in $\underline{1101101}$ we have two occurrences of 1101)





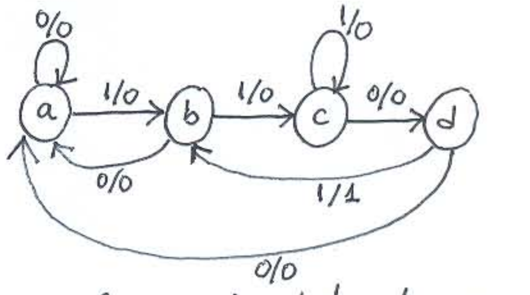
Present state	Next state $x=0$	Next state $x=1$	output $x=0$	output $x=1$
a	f	b	0	0
b	g	c	0	0
c	d	h	0	0
d	i	e	0	1
e	a	c	0	0
f	f	b	0	0
g	g	b	0	0
h	d	h	0	0
i	i	b	0	0

⇒

PS	NS $x=0$	NS $x=1$	output $x=0$	output $x=1$
a	a	b	0	0
b	a	c	0	0
c	d	c	0	0
d	a	b	0	1

Reduced table

⇓



Reduced state diagram

$P_1 = (abcefghi)(d)$

$P_2 = (abefgi)(ch)(d)$

$P_3 = (afgi)(be)(ch)(d)$

$P_4 = (afgi)(be)(ch)(d)$

a, d may be given adjacent codes (1-a)

a, c " " " " " (2)

⇒ a ≡ 00 ; b ≡ 11 ; c ≡ 10 ; d ≡ 01

	0	1
0	a	d
1	c	b

	PS		Input	NS		output	Flip-flop inputs			
	A	B	x	A	B	y	J_A	K_A	J_B	K_B
a	0	0	0	0	0	0	0	X	0	X
	0	0	1	1	1	0	1	X	1	X
d	0	1	0	0	0	0	0	X	X	1
	0	1	1	1	1	1	1	X	X	0
c	1	0	0	0	1	0	X	1	1	X
	1	0	1	1	0	0	X	0	0	X
b	1	1	0	0	0	0	X	1	X	1
	1	1	1	1	0	0	X	0	X	1

Remember:

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

A \ Bx	00	01	11	10
0	0	1	1	0
1		X	X	X

$J_A = x$

X	X	X	X
	0	0	1

$K_A = x'$

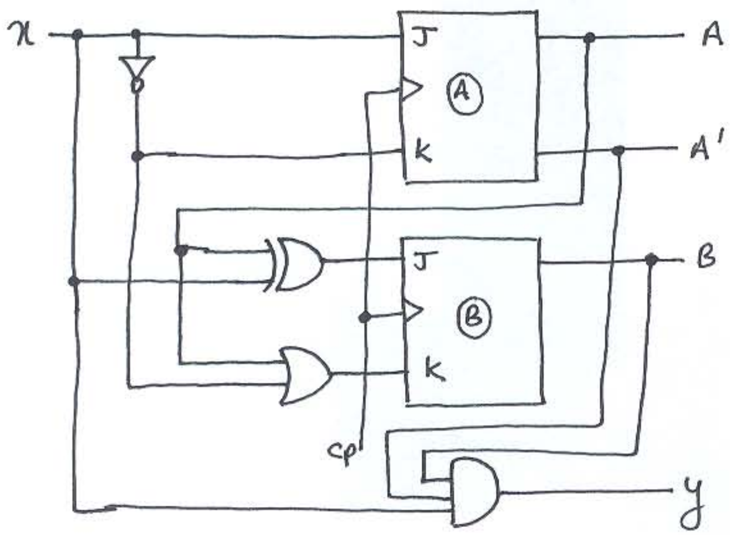
0	1	X	X
1	0	X	X

$J_B = Ax' + A'x$

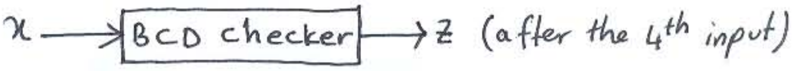
X	X	0	1
X	X	1	1

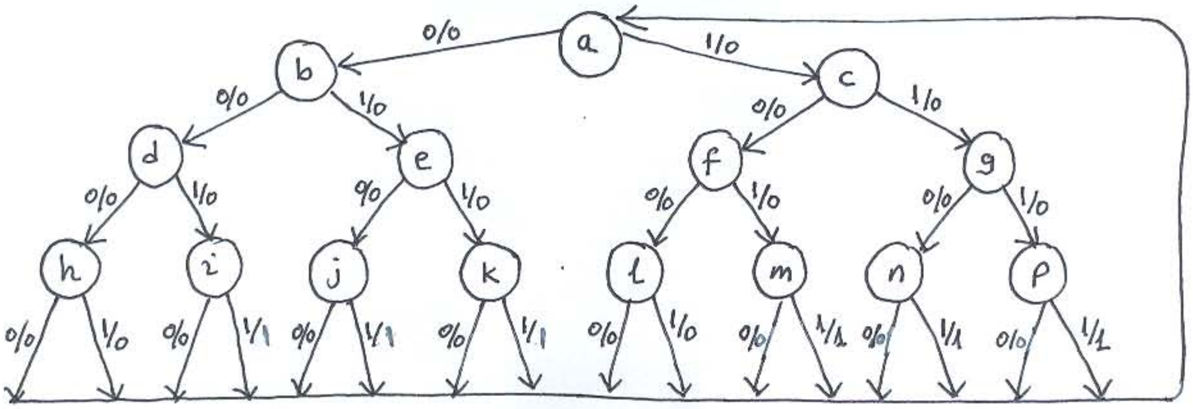
$K_B = A + x'$

$y = A'Bx$



Example: A single-input single-output clocked sequential circuit is to be designed. The circuit will determine whether a string of four consecutive input bits constitute a valid BCD code or not. The output z=1 if the four consecutive bits DO NOT constitute a valid BCD representation, and z=0 otherwise. Least significant bit of the BCD code is applied to the checker input first.





PS	NS x=0	NS x=1	output x=0	output x=1
a	b	c	0	0
b	d	e	0	0
c	f	g	0	0
d	h	i	0	0
e	j	k	0	0
f	l	m	0	0
g	n	p	0	0
h	a	a	0	0
i	a	a	0	1
j	a	a	0	1
k	a	a	0	1
l	a	a	0	0
m	a	a	0	1
n	a	a	0	1
p	a	a	0	1

$P_1 = (abcdefghl)(ijkmnp)$

$P_2 = (abchl)(df)(eg)(ijkmnp)$

$P_3 = (ahl)(bc)(df)(eg)(ijkmnp)$

$P_4 = (a)(hl)(bc)(df)(eg)(ijkmnp)$

⇒

PS	NS x=0	NS x=1	output x=0	output x=1
a	b	b	0	0
b	d	e	0	0
d	h	i	0	0
e	i	i	0	0
h	a	a	0	0
i	a	a	0	1

Reduced state table.

⇒ h, i must be adjacent
d, e may be adjacent ⇒

	00	01	11	10
0	h	d	b	a
1	i	e	–	–

⇒ h ≡ 000 ; d ≡ 001 ; a ≡ 010 ; b ≡ 011
i ≡ 100 ; e ≡ 101 ;

	PS			NS						output	
				$x=0$			$x=1$				
	A	B	C	A	B	C	A	B	C	$x=0$	$x=1$
h →	0	0	0	0	1	0	0	1	0	0	0
d →	0	0	1	0	0	0	1	0	0	0	0
a →	0	1	0	0	1	1	0	1	1	0	0
b →	0	1	1	0	0	1	1	0	1	0	0
i →	1	0	0	0	1	0	0	1	0	0	1
e →	1	0	1	1	0	0	1	0	0	0	0

Let us use JK FF for A
RS FF for B
and T FF for C.

PS			input	NS			output					
A	B	C	x	A	B	C	z	J_A	K_A	S_B	R_B	T_C
0	0	0	0	0	1	0	0	0	X	1	0	0
0	0	0	1	0	1	0	0	0	X	1	0	0
0	0	1	0	0	0	0	0	0	X	0	X	1
0	0	1	1	1	0	0	0	1	X	0	X	1
0	1	0	0	0	1	1	0	0	X	X	0	1
0	1	0	1	0	1	1	0	0	X	X	0	1
0	1	1	0	0	0	1	0	0	X	0	1	0
0	1	1	1	1	0	1	0	1	X	0	1	0
1	0	0	0	0	1	0	0	X	1	1	0	0
1	0	0	1	0	1	0	1	X	1	1	0	0
1	0	1	0	1	0	0	0	X	0	0	X	1
1	0	1	1	1	0	0	0	X	0	0	X	1
1	1	0	0	X	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X	X

Remember:

PS	NS	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

PS	NS	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

PS	NS	T
0	0	0
0	1	1
1	0	1
1	1	0

Using the K-map simplification;

$\underline{\underline{J_A:}}$

AB\Cx		x		
	0	0	1	0
B	0	0	1	0
A B	x	x	x	x
A	x	x	x	x
			C	

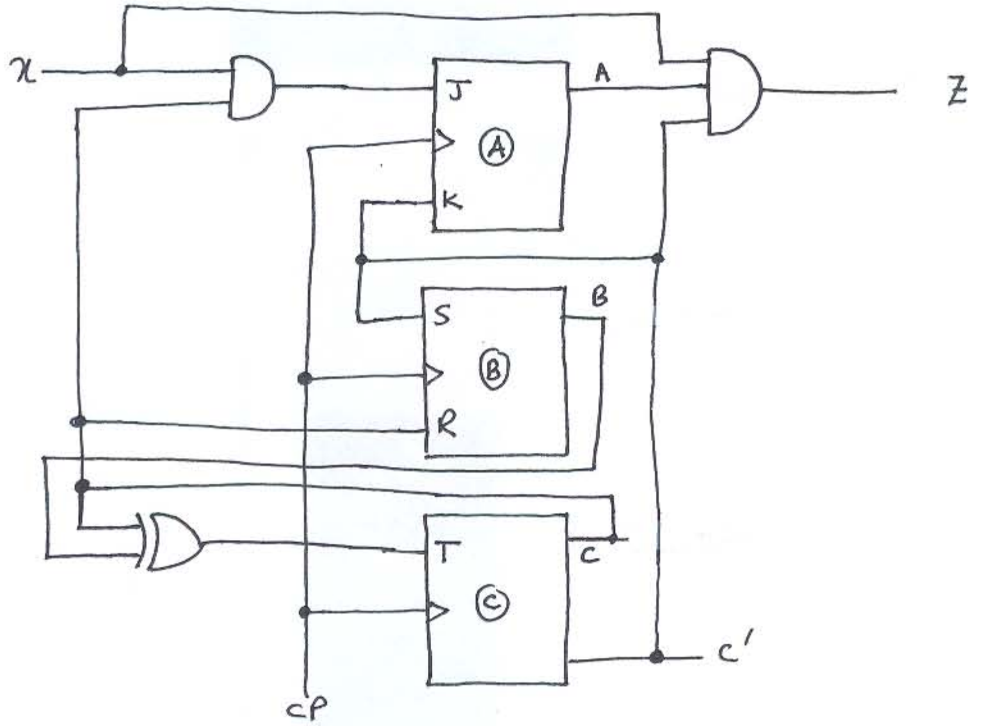
Do not use in Exams!!

$\Rightarrow J_A = Cx$

Similarly, it is easy to find out:

$K_A = C'$ $S_B = C'$ $R_B = C$ $T_C = CB' + BC' = B \oplus C$ $Z = AC'x$

(Note that in Exams, you have to show the K-map for every case)



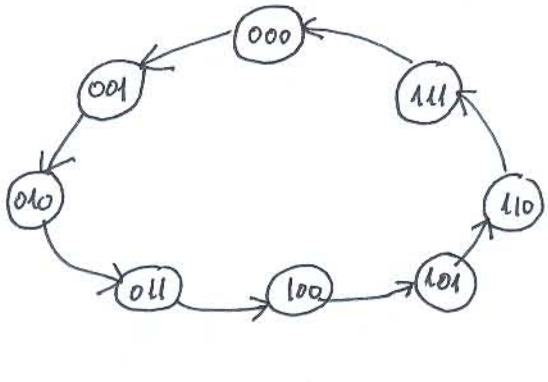
DESIGN OF COUNTERS:

A sequential circuit that goes through a prescribed sequence of states upon the application of input pulses is called a "counter". The input pulses, called count pulses, may be clock pulses or they may originate from an external source.

An n-bit binary counter consists of n FFs and can count from 0 to $2^n - 1$.

Example: Design a 3-bit binary up-counter using T flip-flops.

.Transitions occur every clock pulse



Remember:

PS	NS	T
0	0	0
0	1	1
1	0	1
1	1	0

PS			NS			FF inputs		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A_2}	T_{A_1}	T_{A_0}
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	0	1
1	1	1	0	0	0	1	1	1

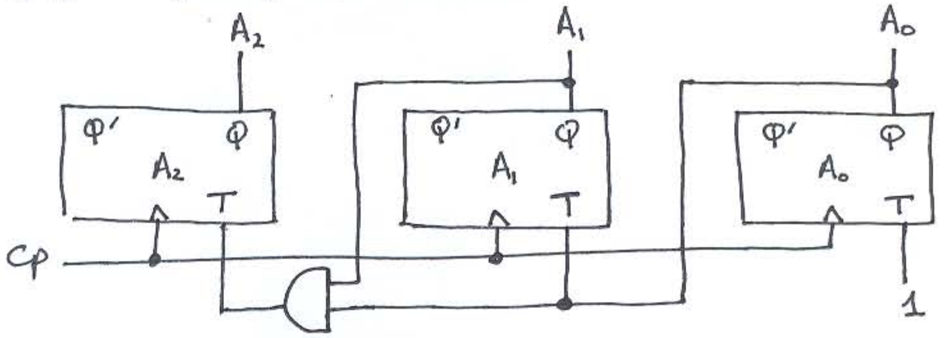
Using the k-map simplification;

⇒

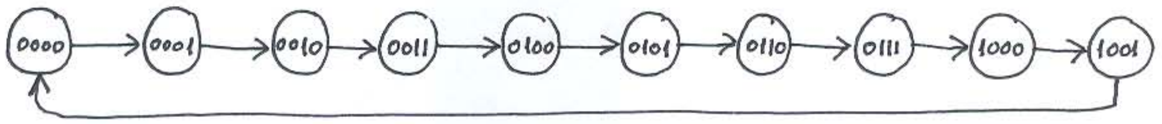
$T_{A_2} = A_1 A_0$

$T_{A_1} = A_0$

$T_{A_0} = 1$



Example: Using JK FFs, design a BCD counter.

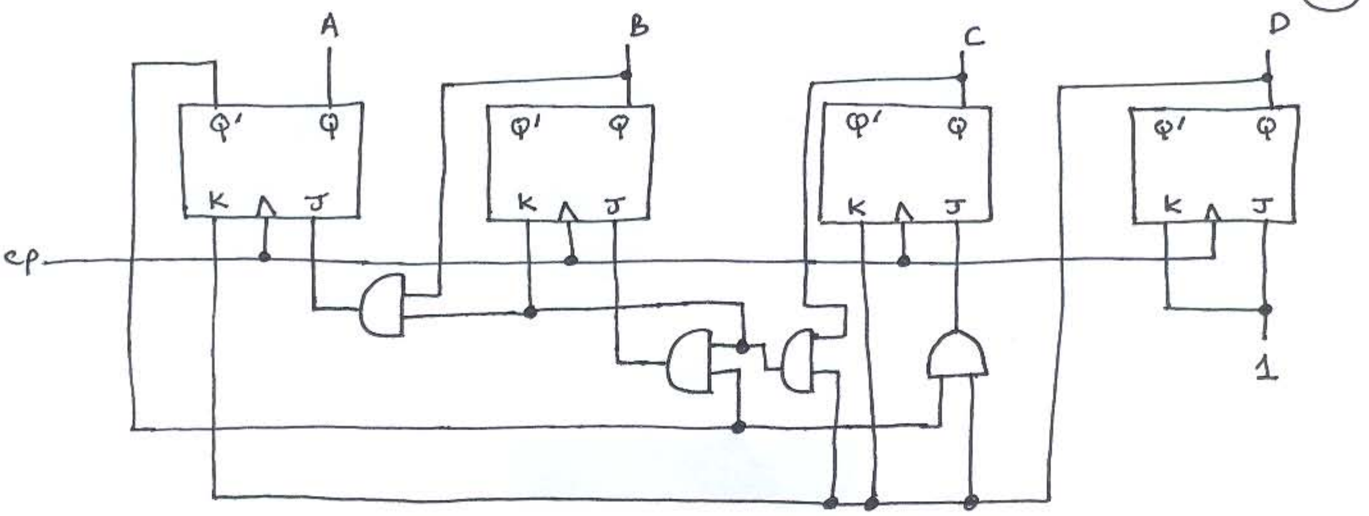


A	B	C	D	A	B	C	D	J_A	K_A	J_B	K_B	J_C	K_C	J_D	K_D
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	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

Using k-map simplification, we have:

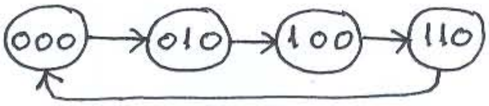
$J_A = BCD$ $\quad$ $J_B = A'CD$ $\quad$ $J_C = A'D$ $\quad$ $J_D = 1$

$K_A = D$ $\quad$ $K_B = CD$ $\quad$ $K_C = D$ $\quad$ $K_D = 1$



Example:

Using D FFs, design a counter with the following repeated sequence: 0, 2, 4, 6.



PS			NS			FF inputs		
A	B	C	A	B	C	D_A	D_B	D_C
0	0	0	0	1	0	0	1	0
0	1	0	1	0	0	1	0	0
1	0	0	1	1	0	1	1	0
1	1	0	0	0	0	0	0	0

A\BC				
	0	X	X	1
	1	X	X	0

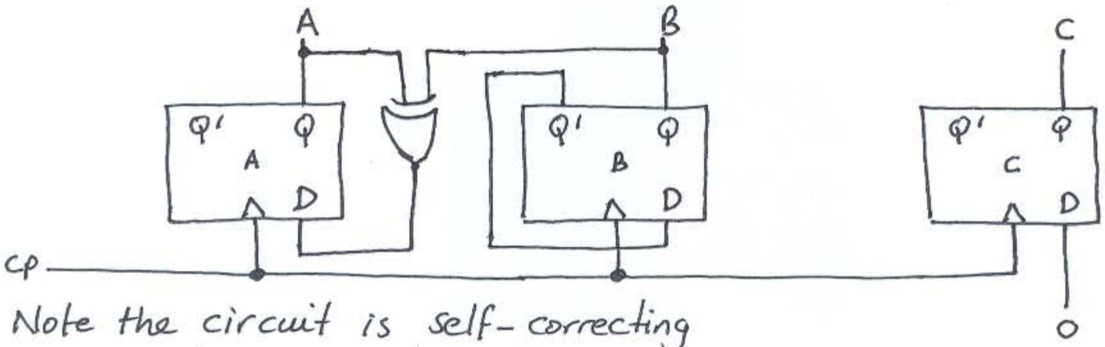
$D_A = AB' + A'B = A \oplus B$

A\BC			C	C	
	1	X	X	0	
	1	X	X	0	}A
			B	B	

$D_B = B'$

0	X	X	0
0	X	X	0

$D_C = 0$



Note the circuit is self-correcting

i.e if it happens that the circuit is in one of the unused states (because of an error signal) ⇒ it goes to one of the used states. Namely, if the circuit is in 001, it goes to 010 (i.e 1→2) and if 3→4, if 5→6 and if 7→0. (check by forming the circuit excitation table).

HW # 1) 6.6, 6.8, 6.10, 6.14, 6.18, 6.20, 6.25 (c).