Kokila.K.S
Asst.professor
Dept of ECE
BGSIT.

Module - 4

Simple flip-flop Applications

Introduction:

Flip-flop is nothing but a binary cell capable of storing one bit information, & can be connected to perform counting operations. Such a group of flip-flops is called Counters.

The collection of flip-flops can be used to store a word, is called Register.

A flip-flop can store 1-bit information. So an n-bit register has a group of n flip-flops & is capable of storing any binary information/number containing n-bits.

Register.

* Registers are used to store data in a digital system.
* A 4-bit register can store binary bits from 0000 - 1111. These are called Contents or States of a register. Thus a 4-bit register has 16 possible states.
* A cascade of 4 flip-flops configured as a register can store one nibble of data.
* Shift registers are capable of moving or shifting the data stored in their flip-flops in either directions.

Ex:- Consider a 4-bit shift register with data 0100 or decimal 4. A left shift results in 1000 or decimal 8 & a right shift results in 0010 or decimal 2.

* Each left shift has a "multiplication by 2" effect & Each right shift has a "division by 2" effect.
* Shift register which can shift data in both directions are called bi-directional shift registers.
* Those which can shift data in only one direction are called unidirectional shift registers.
* Hence these are classified based on whether data is input or output in - Serial or - parallel fashion.

* when information is transferred in parallel manner, all the bits in the information are handled simultaneously, as a single entity at a time.

No of i/p lines required are equal to no of bits in an information & requires one clock pulse.

* Information transfer in serial manner involves bit by bit availability at a time [One bit at a time].

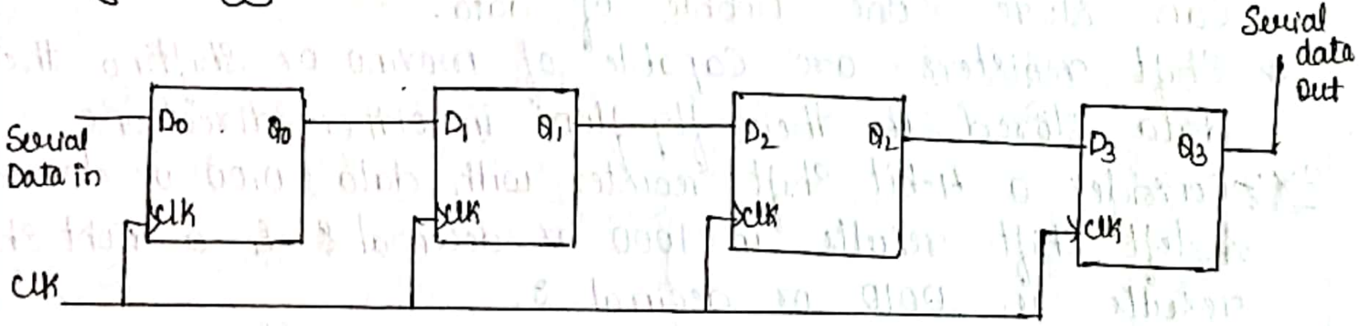
This type of transfer requires single i/p line & no of clock pulses = no of bits in an information.

* Thus, there are 4 possible ways of data Transfer. They are

1). Serial In Serial out. [SISO]
2). serial In parallel out [SIPO]
3). Parallel In parallel out [PIPO]
4). parallel In Serial out [PISO]

Serial In Serial out [SISO]:-

* A 4-bit SISO unidirectional shift register using positive edge Triggered D flip-flops is shown below.



* The D inputs of each flip-flop is connected to the 'Q' o/p of the previous stage to the left.
* The Control i/p's of all flip flops are connected together to a common synchronized clock. Thus, upon the occurance of the positive edge of clock signal, the content of each flip flop is shifted one position to the right.
* The data on the serial in line gets stored in flip-flop A & appears at Q_A.

* The Content of left most flip flop after clock signal depends on serial data input line & Content of right most flip-flop before a clock signal is lost.

Ex:- serial data: 1101

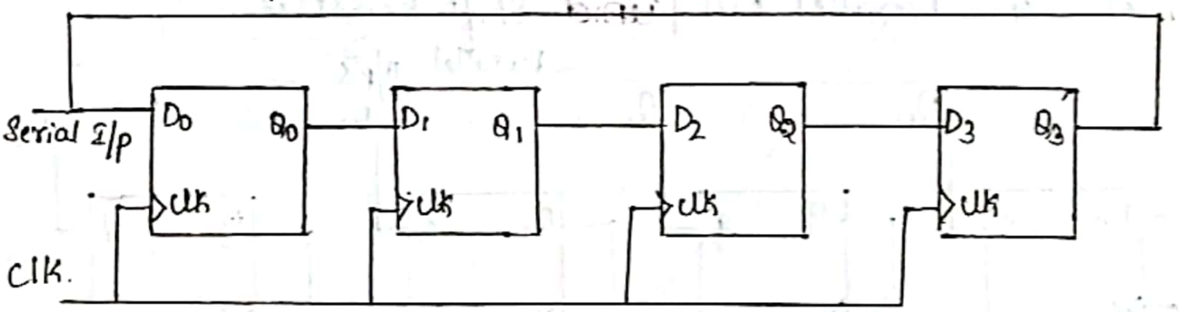
initial data 0000

clk i/p	Q_A	Q_B	Q_C	Q_D			serial o/p
↑	1→1	0	0	0	serial i/p	↑	x 1 1 0
↑	0→0	1	0	0		↑	x x 1 1
↑	1→1	0	1	0		↑	x x x 1
↑	1→1	1	0	1			

Circular shift Register:-

In some application, the information within register must be preserved, by avoiding loss of information at the o/p of last flip flop.

For this, serial data o/p line is connected to serial data in line as shown below. This type of register is called Circular shift register.

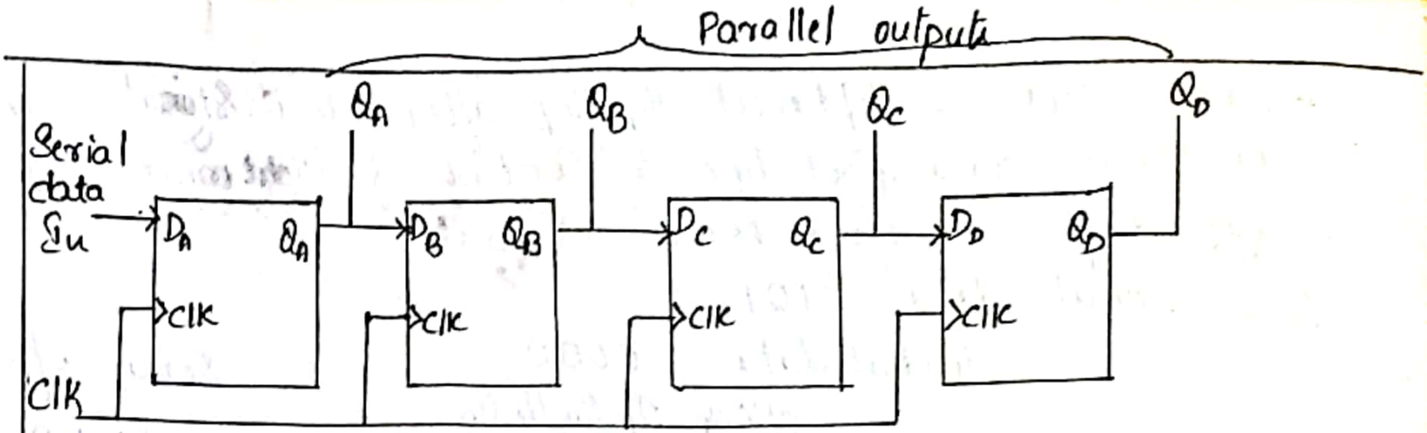


Ex:- If content is 1011, upon the occurrence of +ve edge of clock signal it becomes, 1101.

Serial In Parallel Out [SIPO] Shift Register:

* In this case, the data bits are entered into the register serially i.e., One after the other, The information is available as a single entity i.e., parallel out @ flip flop o/p terminals.

* This type of register provides serial to parallel conversion of Information.

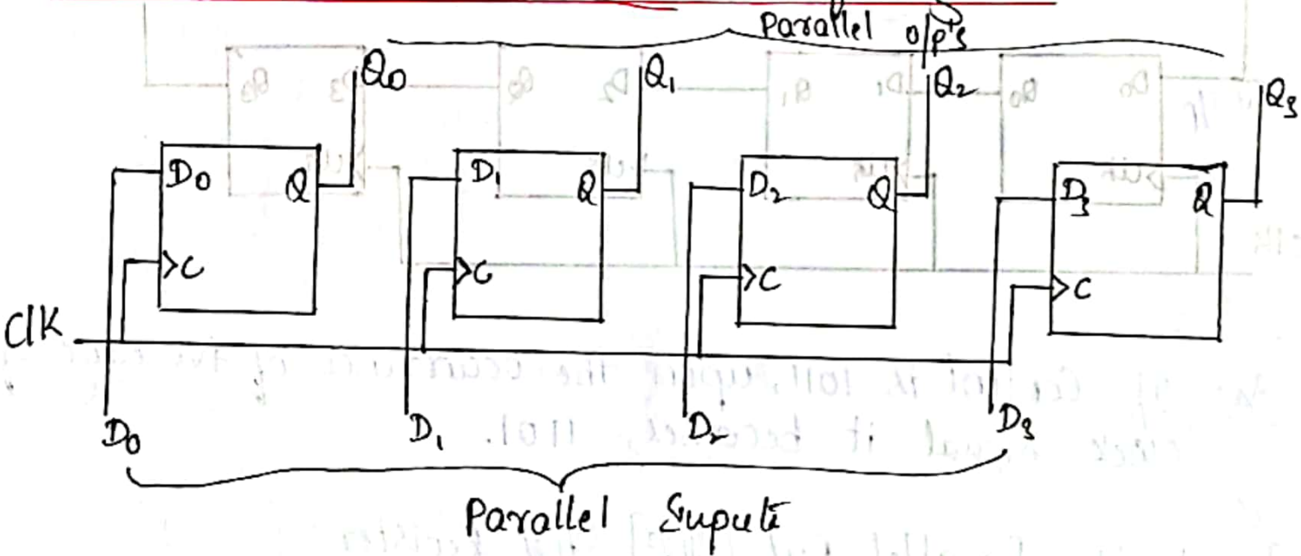


Clk	Serial data	Q_A	Q_B	Q_C	Q_D
		1	x	x	x
↑	1	1	x	x	x
↑	0	0	1	x	x
↑	1	1	0	1	x
↑	1	1	1	0	1

Parallel out

observe that once the 4 bit data is shifted in after 4 clock pulses, the data stored in each flip-flop is available at the respective 'Q' o/p's.

Parallel In Parallel out [PIPO] Shift Registers:



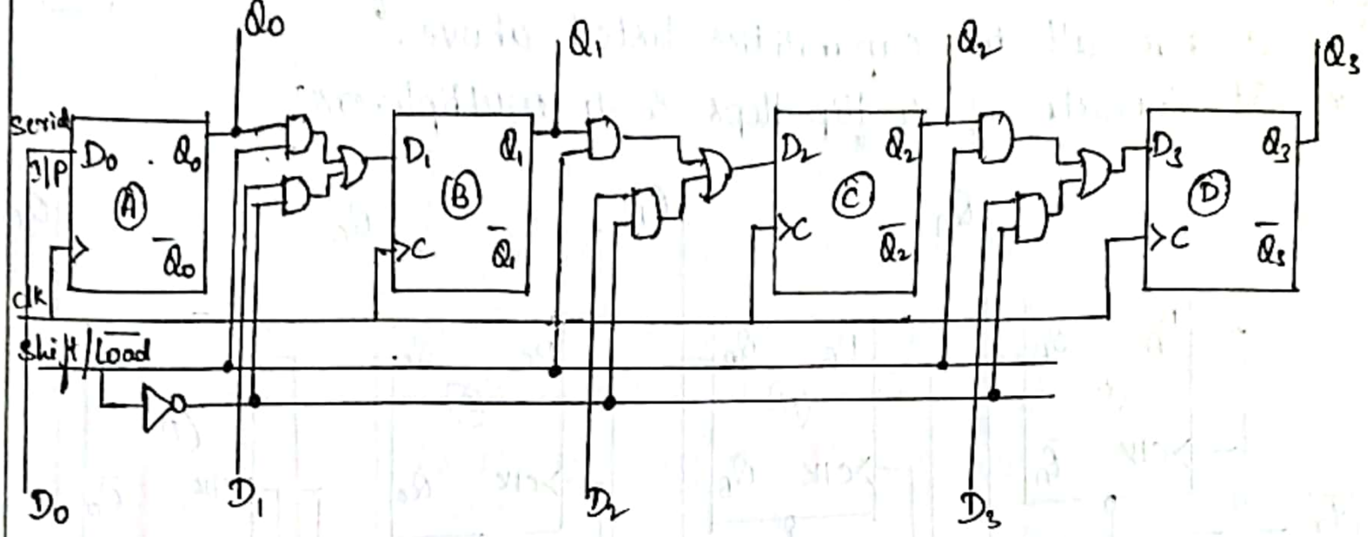
In this type, the bits are entered in parallel i.e, Simultaneously into their respective stages on parallel lines & the bits appear on parallel outputs simultaneously upon the occurrence of clock pulse.

Eg:

Clk	Parallel In				Parallel out			
	D_0	D_1	D_2	D_3	Q_0	Q_1	Q_2	Q_3
↑	1	0	0	1	1	0	0	1

Parallel in Serial out [PISO] Shift Register.

In this type, the bits are entered in parallel i.e, Simultaneously into their respective stages on parallel lines.



* Fig above shows a parallel in Serial out unidirectional shift register.

* There are 4 i/p lines D_0, D_1, D_2, D_3 for entering data in parallel into the register & parallel o/p's are Q_0, Q_1, Q_2, Q_3.

* Shift/$\overline{\text{Load}}$ is the control i/p which allows shift or loading data operation of the register.

* when shift/$\overline{\text{load}}$ is @ logic 1, all the upper AND gates are enabled, Serial i/p (D_0) appears at D i/p of flip flop A.

* The 'Q' o/p's of flip-flops A, B & C appear respectively at the 'D' i/p's of flip flops B, C, & D (next right flip flops).

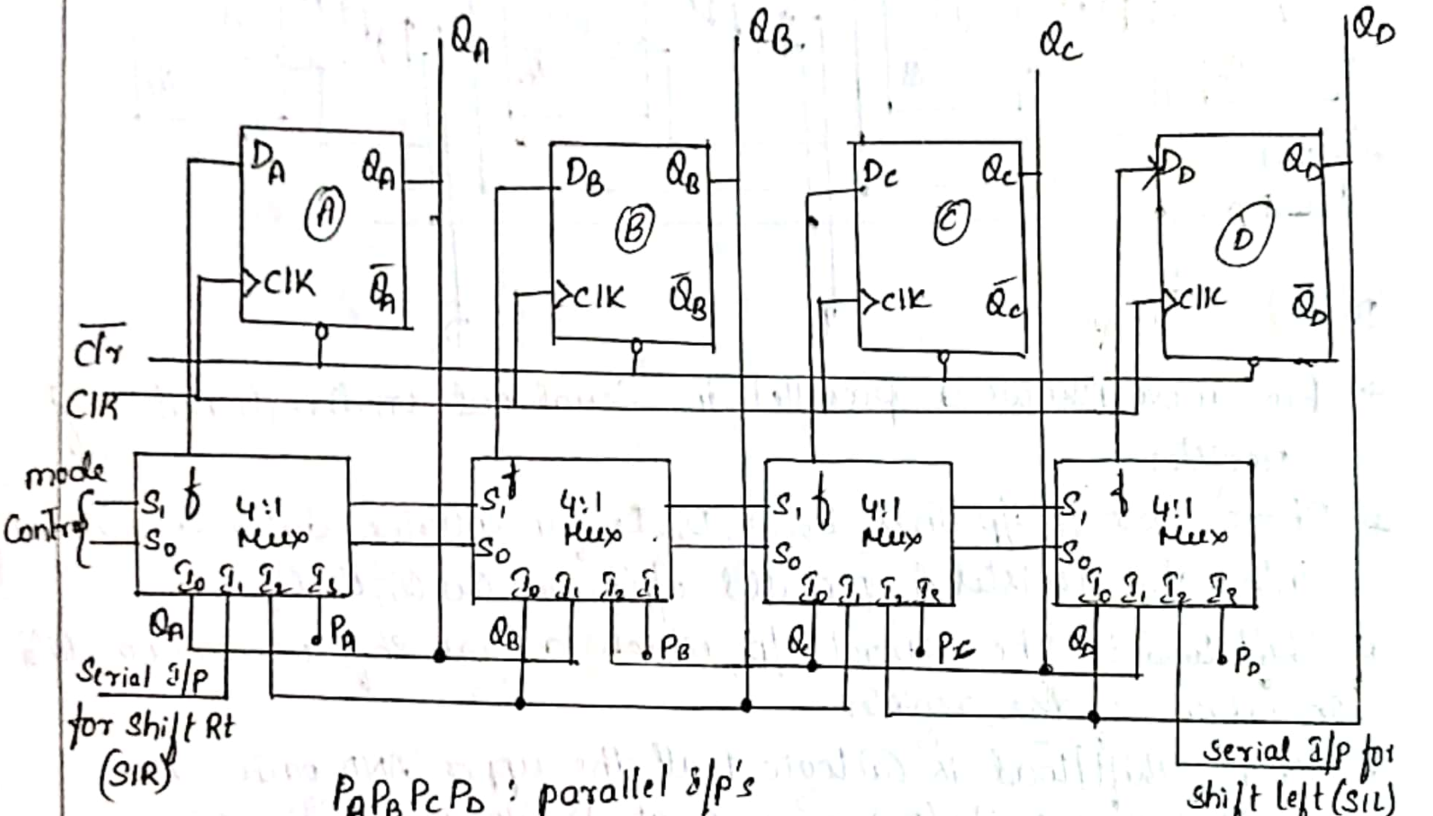
* The data gets Shifted right on the application of clock pulses.

* when Shift/$\overline{\text{Load}}$ is at logic '0', lower AND gates gets enabled & upper AND gates get disabled.

* The parallel i/p's now appear at the respective 'D' i/p's thereby facilitating parallel i/p. (i.e., all 4 bits are stored Simultaneously].

Universal Shift Register

- A universal shift register is one which is bi-directional & has capabalities to accept both serial & parallel inputs as well as capabilities of serial & parallel o/p.
- Fig below shows the 4-bit universal shift register. It has all the capabilities listed above.
- It consists of 4 flip-flops & 4 multiplexers.



PA PB PC PD : parallel i/p's
QA QB QC QD: Parallel o/p's

- The mode selection table & symbol are shown in fig below.

Select lines S1 S0	Data selected	operation.
0 0	I0	hold
0 1	I1	Shift right
1 0	I2	shift left
1 1	I3	parallel load

fig: Mode selection table.

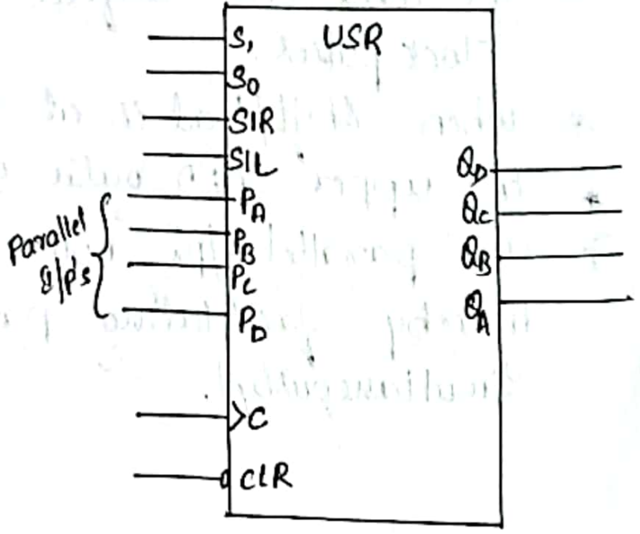


Fig: Symbol.

* Depending on select lines of multiplexers [mode Control lines] the register can retain its Current State, Shift left, Shift right. Each of these operations is the result of +ve edge of clock signal.

* logic-0 on the Asynchronous i/p $\overline{Clr}$, clears register Contents

* when $S_1S_0=00$, I_0 of the mux is selected. The Q o/p of flip flops are Connected to their respective 'D' i/p's & upon the occurrence of the Clock pulse 'D' i/p's appears at 'Q' o/p.
∴ There is no change in register Content.

* when $S_1S_0=0$, I_1 of mux is selected, Serial input for Shift right gets Connected to 'D' i/p of flip flop A, Q_A to D_B, Q_B to D_C, Q_C to D_D. Now, upon the occurrence of the Clock pulse, register shift the data to right by one bit position

* when $S_1S_0=10$, I_2 of mux is selected, Serial input for left shift get Connected to D i/p of flip flop D, Q_D to D_C, Q_C to D_B & Q_B to D_A, Now upon the occurrence of clock pulse, register shifts the data to left by one bit position. i.e, it performs left shift operation.

* when $S_1S_0=11$, I_3 is selected & I_A, I_B, I_C, I_D appears at D_A, D_B, D_C, D_D respectively Constituting parallel inputs.

Applications of Shift Register:

1. Serial In Serial out [SISO] shift register can be used introduce time delay in digital signals.
2. Serial In parallel out [SIPO] shift register Can be used to Convert data in serial form to the parallel form.
3. Parallel In Serial out [PISO] shift register can be used to Convert data in parallel form to Serial form.
4. Shift register can also be used as Counter, to identify time sequences for Specific instances.

Counters.

* A Sequential circuit that generates prescribed sequence of states upon application of clock pulses is known as Counter.

(or)

* Counter is a cascaded arrangement of flip-flops configured to o/p a specific sequence on application of a clock.

* Counters are used for counting number of clock pulses arriving at its clock input & are useful for generating timing sequences to control operations in a digital system.

* Each o/p of the sequence is dependent on the contents of the flip-flops & is called a state of a counter.

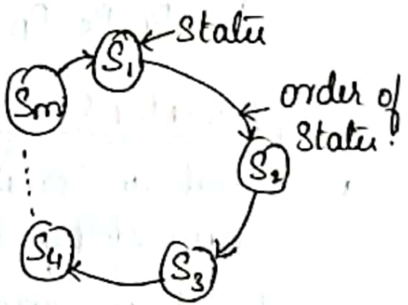
* The modulus of a counter is the total no of states of the counter.

*. If counter is cascade of n flip-flops, then no possible states are 2^n. The no of states may be $\le 2^n$. i.e., If counter has 'm' (2^n) distinct states, then it is called modulus-m (or) mod-m counter.

* The order in which states appears is called Counting Sequence

*. Graphical representation of counting sequence is called "State diagram".

* Each node indicates states of the counter & arrows in the graph denotes order in which states occur.



Synchronous Counter: When counter is clocked such that each flip-flop in the counter is triggered at the same time, that counter is said to be synchronous counter.

Asynchronous Counter/Ripple Counter: In this type of counter flip flops are not clocked simultaneously & flip flops are connected in such a way that output of first flipflop drives the clock for the next flipflop.

Binary Ripple Counter (or) Asynchronous Counter:

* Counters whose Counting sequence Corresponds to binary no are known as Binary Counters

* In this type of Counter flip flops are not clocked Simultaneously & flip flops are Connected in Such a way that o/p of first flip flop drives the clock for the next flip flop.
  ∴ This type of Counter is known as asynchronous Counter. (or) Ripple Counter.

* An 'n' bit binary consists of 'n' flip flops can count from '0' to $(2^n - 1)$.
  ∴ modulus of binary Counter is 2^n. where n → no of flip flops.

* For up counter, counting sequence is from '0' to '2^n-1'.

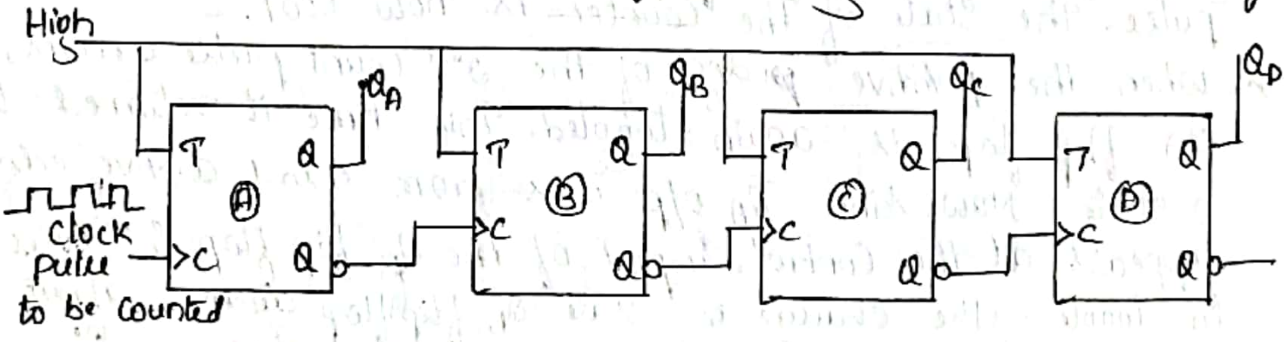
* For down counter, Counting sequence is from (2^n-1) to 0.
  After reaching its max (or) min Count Counting Sequence is repeated from its initial state.

* Ex:- A 3-bit binary up counter sequences from '000' (0) to '111' $(2^n = 2^3 - 1 = 8-1=7)$ & repeats the sequence on reaching '111'.

Asynchronous 4-bit binary up Counter (or) Ripple Counter [using +ve edge FF]

* In 4-bit Counter, a cascade of 4 flip-flops can be used to Configure a Counter upto modulus 16.

* Fig Shows a 4-bit binary up Counter Configured using Positive edge Triggered 'T' flip-flops along with its Count Sequence.



* With the Count enable or T inputs held at logic 1, the o/p of each flip flop toggles for every 0 to 1 transition of its clock i/p or at every positive edge of its clock input.

* The clock inputs of flip-flops B, C, & D are connected to $\bar{Q}$ o/p of the previous flip-flop.
* These flip-flops change state on 1 to 0 transition of $\bar{Q}$ o/p's which correspond to 0-1 transition of Q outputs.

Count	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1(⊓L)	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Repeat

fig: Counter table

* Since it is a 4-bit counter, no of states are $2^4 = 16$.
∴ It is a mod-16 counter counting sequence is from 0000 to 1111 [0-15]

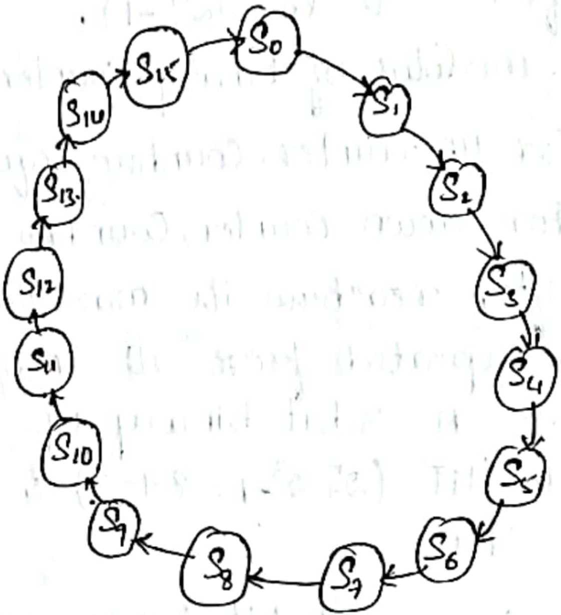


fig: State diagram

* Assume, initially the count is in state 0000 & count enable signal is logic-1. upon the occurrence of +ve edge of the first count pulse, Q_A flip flop changes to its 1-state. Since $\bar{Q}_0$ goes from 1 to 0, flip flop Q_B is not affected by the i/p pulse. The state of the counter is now 0001.
* when the positive edge of the 2nd count pulse arrives, the Q_A flip flop is again toggled. This time it returned to 0-state. Now, since $\bar{Q}_A$ o/p goes from 0 to 1, a +ve edge appears at the control input of the Q_B flip flop & causes it to toggle. The change in state Q_B flip flop doesnot affect Q_C flip flop since -ve edge occurs at its control i/p.
Hence at the end of the pulse, state of counter is 0010.

* The 3rd Count pulse causes only Q_A flip flop to change state & Count to become 0011.

* For the 4th Count pulse it toggles Q_A o/p to change state from 1 to 0, & $\bar{Q}_A$ becomes 1. This causes +ve edge to occur at $\bar{Q}_A$ terminal. Thus Q_B is toggled, returning it to 0-state. In addition, when Q_B flip flop changes its state, the Q_C flipflop is toggeled by 0 to 1 appearing at $\bar{Q}_B$ o/p terminals. Now the state of the Counter is 0100. like This Counting Sequence Continuous upto 1111.

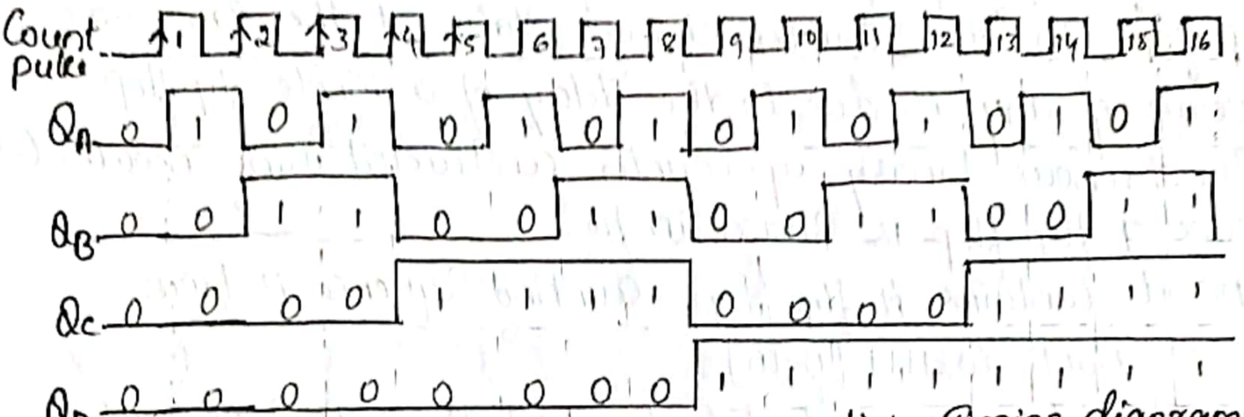
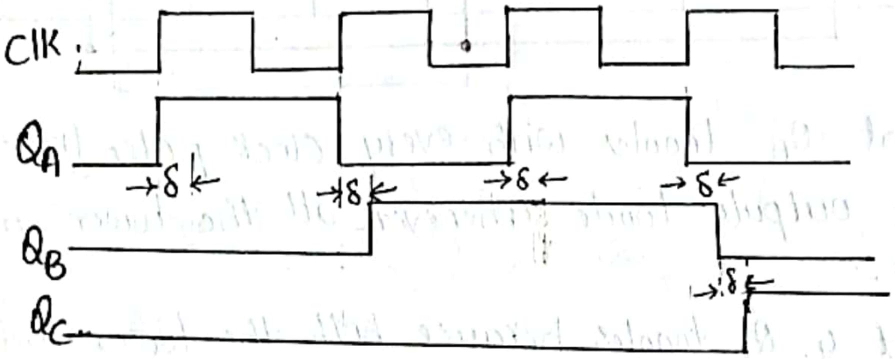

Count pulse
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Q_A 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
Q_B 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1
Q_C 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1
Q_D 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1

fig: Timing diagram

* Asynchronous (or) binary Counter is also known as ripple Counter since change in state of Q_i flip flop is used to toggle the Q_{i+1} flip flop. Thus, the effect of Count pulse must ripple through the Counter.

Draw back of Asynchronous Counter:



δ - tpd
propagation delay time.

fig: Timing dig with delay (δ)

As shown in the fig above there is a propagation delay b/n i/p & o/p of flip flop, this rippling behaviour affects the overall time delay b/n the occurrence of Count pulse & when stability Count appears at o/p terminals. This delay is more when all flip flop's need to toggle for final Count.

For n-stage binary ripple counter, the worst case settling time becomes n x tpd.

Synchronous Binary Counter :-

In this type of Counter, all the flip flops are clocked Simultaneously.

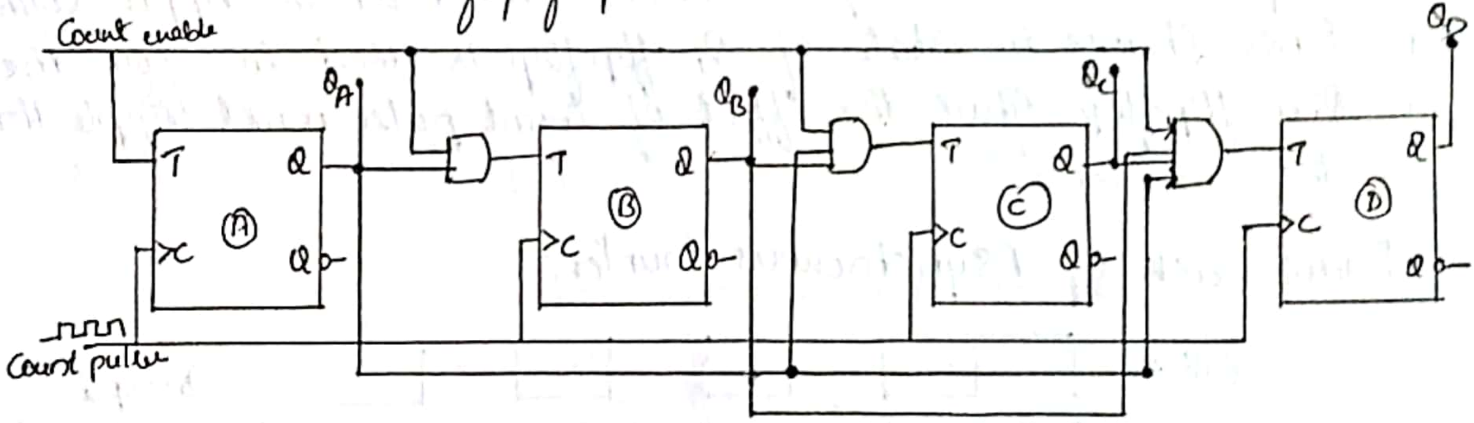
* when a counter is clocked such that each flipflop in the Counter is triggered at the same time, such counter is called as Synchronous Counter.

* The effect of propagation delay is not greater since all flip flops in the Counter change state at the same time.

* Delay if any is due to the delay of a single flip flop.

* Synchronous binary up counter constructed using negative (-ve) edge 'T' flip flop is shown in fig.
As it contains 4 flip flops counting sequence is from 0000 to 1111 [0-15].

* Count pulses are applied to the Control input C of each clocked flip flop.



* Observe that Q_A toggles with every Clock pulse [Refer Truth table of up counter]

* The other outputs toggle whenever all the lower order flip-flops are at 1.

Ex:- at Count 4, Q_2 toggles because both the lower order flip-flops Q_1 & Q_0 at 1 at Count 3. [0011].

* This can be applied to all Counts in the table.

* Thus the lowest significant flip-flop must toggle at every clock pulse & the rest must toggle whenever every lower

order flipflop is at 1.

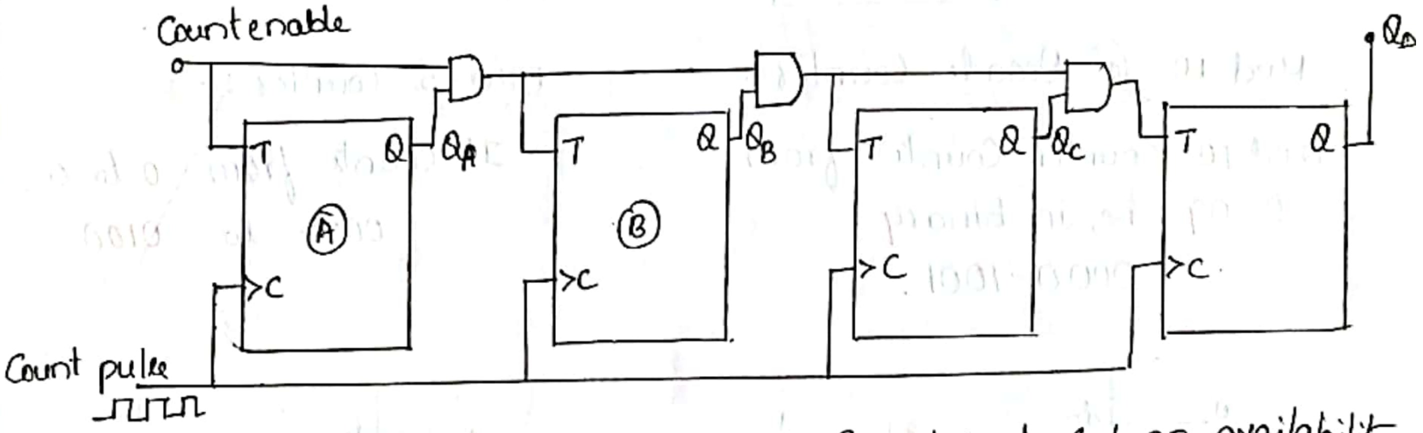
- All lower order flip-flop outputs can be ANDed to enable the toggle of a given flip-flop.
- When Count enable line is at logic 1, the AND gate outputs place a 1 at the T inputs when all the previous flip-flop outputs are at 1.

Note:- 1. write Truth table of upcounter & Timing diagram.

2. Circuit is same for down Counter, but Instead of Q, $\bar{Q}$ is given as I/p to AND gate.

Drawback:-

- NO of gates or stages increases the no of inputs to the AND gate.
- Making use of the fact that ANDed outputs of all previous flip-flops are available at the output of each AND gate, the gating can be modified to keep the no of inputs to the AND gates are constant as shown below.



Note:- In Synchronous Counters, Counter Speed is based on availability of next count at the o/p terminal. Hence, Synchronous Counters are faster than Asynchronous.

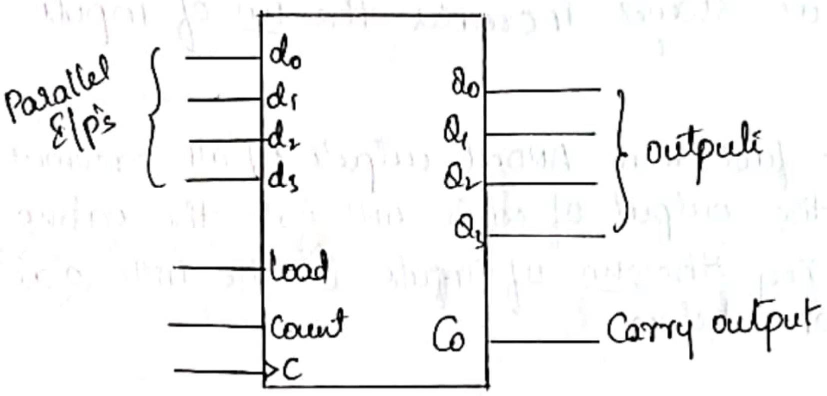
Synchronous counter w [4 bit] with parallel load.

- we know that a n-bit Counter can be used as a mod-2^n Counter.
- we can configure a mod-m Counter where $m < 2^n$, if we include a facility to parallel load an initial Count.

Ex:- a 0000 to 1111 counter can be made to Count from 0000 to 1000, which becomes a mod-9 Counter.

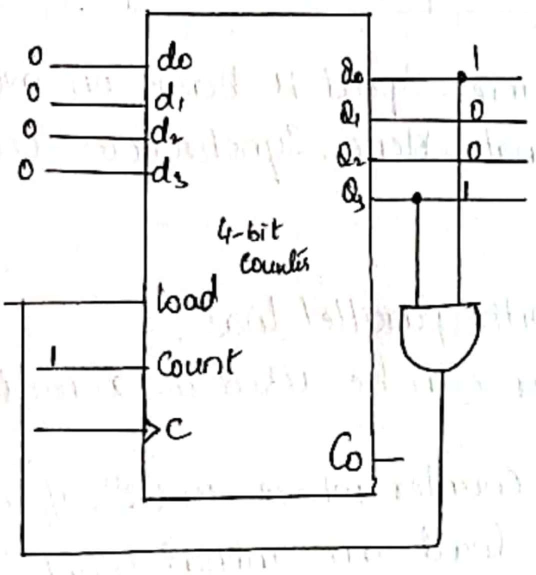
- Such 4-bit counters are available in a commercial package with load & Count i/p's.
- A logic 1 at the load input would load the Count at d_0, d_1, d_2, d_3 into the Counter.
- A logic 1 at the Count input would enable the up-Counting.
- CO → Carry output, is used to cascade 4 bit Counters to form higher bit counters.

  Co which goes high during the 1111 to 0000 transition.



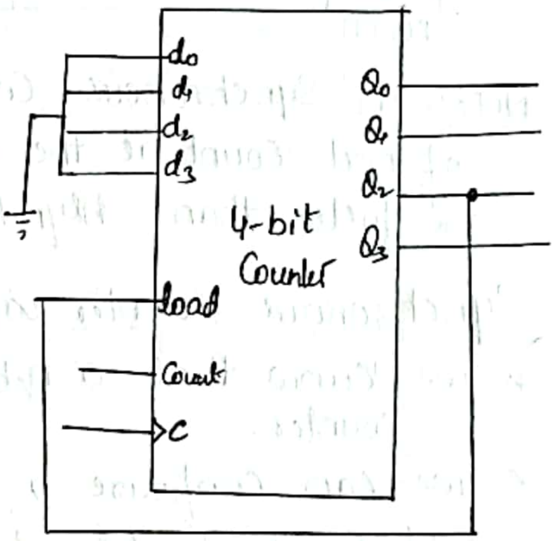
Mod-10 (or) Decade Counter :

mod-10 Counter Counts from 0-09 i.e, in binary 0000-1001.

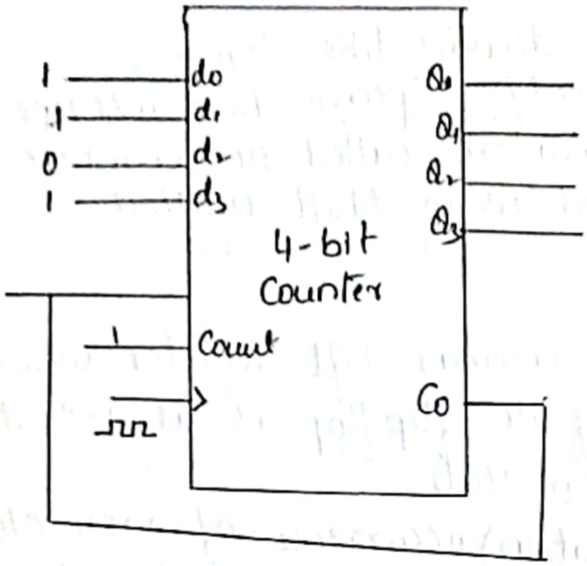


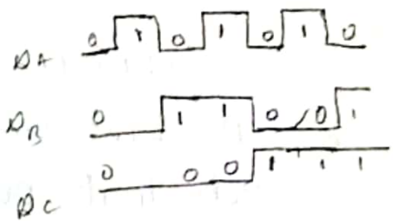
mod-5 Counter :-

It Counts from 0 to 4
0000 to 0100

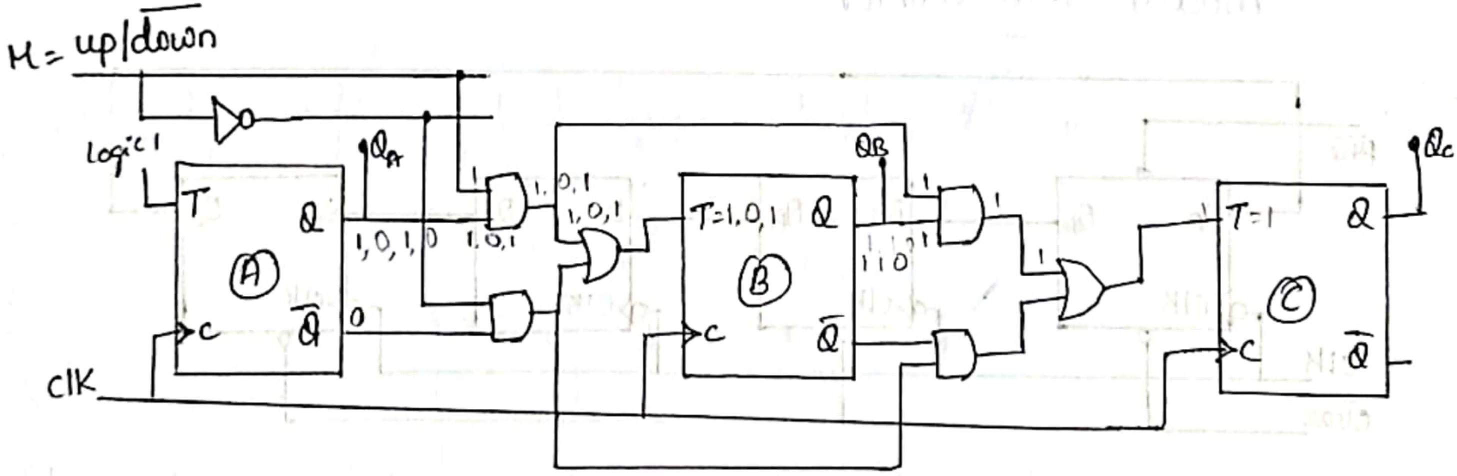


Design a Counter to o/p Sequence from 1011 to 1111 & Repeat from 1011





Synchronous up/down Counter using T-flip-flop.



- * 8 bit Synchronous up/$\overline{down}$ Counter is as shown in the fig.
- * The gating at each T input should be modified to accept all the previous ANDed Q outputs as well as the ANDed $\bar{Q}$ o/p's controlled by the up/down control input.
- * with the count enable at logic 1, up/$\overline{down}$ at logic 1 enables the upper AND gate for up counting & up/$\overline{down}$ at logic 0 enables the lower AND gate for down counting.
- * For 4-bit up/$\overline{down}$ Counter 4 Flip-flops has to be used.

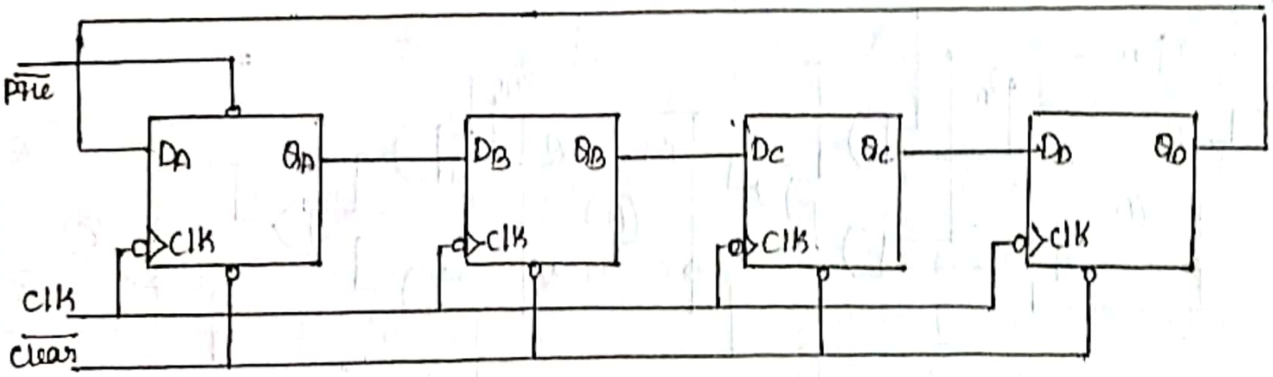
Counters Based on Shift Registers:-

- * There are Several applications in digital Systems where non-binary Counters are required.
- * These Counters output a decoder like Sequence.
- * These are used to identify Specific time instance.
- * 2 types of non-binary Counters called ring Counters & Switch-tail Counters can be designed using shift registers.

Ring Counter.

* A Ring Counter is a circular shift register which is initialized Such that only one of its flip flop is at 1-State, while all others are in 0-State.

Thus, upon the (application) occurrence of each clock pulse, the single 1 is shifted around the register. Fig below shows mod-4 ring Counter.



* In the above figure $\overline{CLR}$ followed by $\overline{Preset}$ makes the o/p of 1st Stage to logic '1' & remaining outputs are '0'. i.e., Q_A is One & Q_B, Q_C, Q_D are '0'.

CLK	Q_A	Q_B	Q_C	Q_D
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	0	0	0

* No. of States = 4

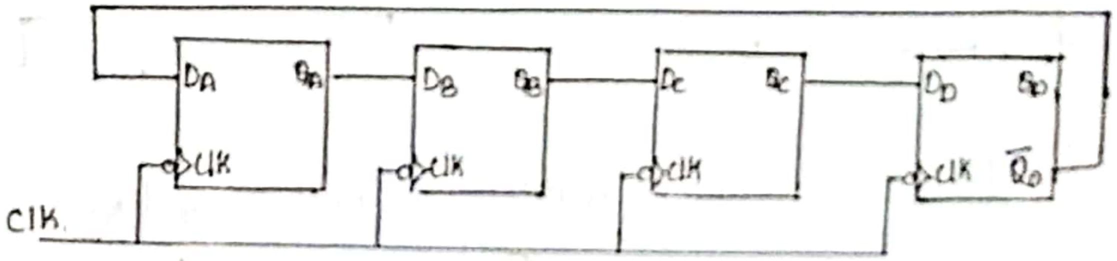
∴ It is a mod-4 ring Counter.

Counter Initialized to 1000, then Counting Sequence shows results as in T.T.

* A Mod-n ring Counter would require 'n' flip-flops.

Johnson Counter or Twisted Ring Counter or Switch tail Counter

* In a Johnson Counter, the Q output of each stage of flip-flop is Connected to the 'D' input of the next stage. But the Complement output of the last flip-flop is connected back to the D-input of the first flip-flop as shown in the fig below.



* Initially, the register (all flip-flops) are Cleared. So, all the o/p's Q_A, Q_B, Q_C, Q_D are zero.

* The o/p of last stage, Q_D is zero. ∴ Complement o/p of last stage, $\overline{Q_D}$ is One. This is Connected back to the D i/p of first flip/flop i.e. FF A. $[D_A]$. i.e., $D_A = 1$. Hence for the next clock pulse o/p becomes 1000.

* Sequence of states are summarized in table below.

* A n-stage Johnson Counter has 2n states – [modulus 2n]

* In 4-bit Johnson Counter, total no of states = 8. ∴ It is a mod-8 Counter.

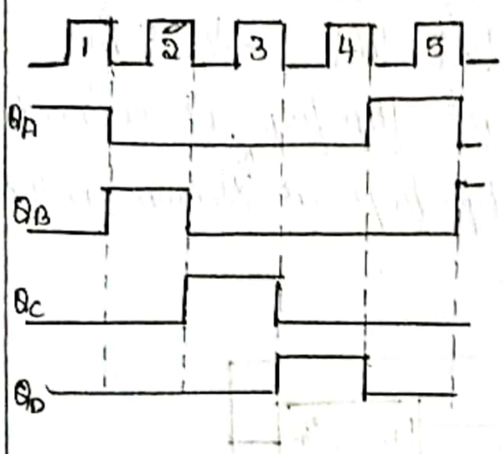
CLK	Q_A	Q_B	Q_C	Q_D
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

** Johnson Counter is a digital Counter in which the 'Q' o/p of one flip flop is directly connected to i/p of next flip flop. But $\overline{Q}$ of LSB is Connected back to the i/p of MSB.

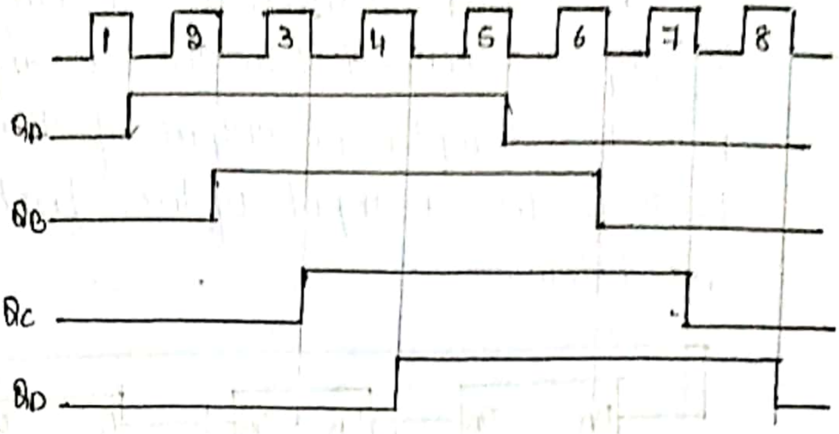
Note:

** Refer Class Notes for problems.

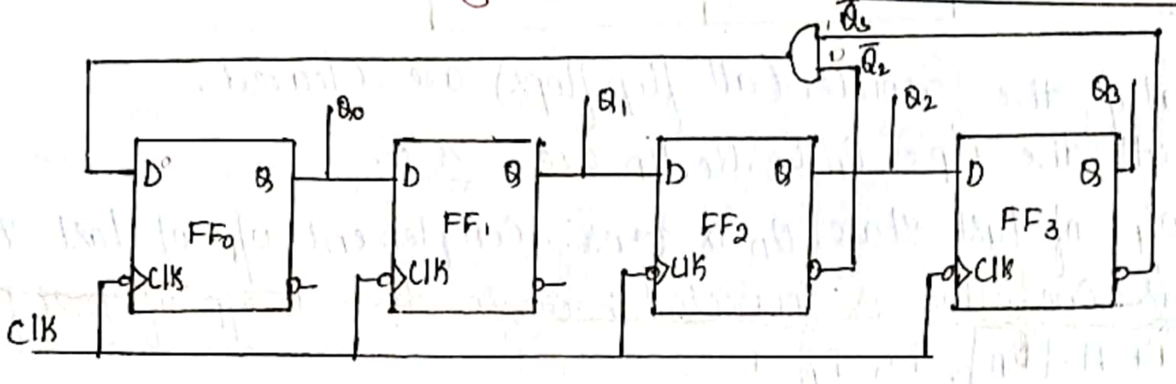
* Timing diagram of 4-bit Ring Counter



* Timing dig of 4-bit Johnson Counter



Mod-7 twisted Ring Counter [~~Xs~~ Mod-7 Johnson Counter].



0 0 0 0
1 0 0 0
1 1 0 0
1 1 1 0
0 1 1

* Fig above shows a mod-7 twisted ring counter with a odd no of states where the 1111 state gets bypassed.

* Here the compliment o/p ($\bar{Q}$) of LSB flip-flop (FF_3) & ~~the~~ ~~1st~~ FF_2 are ANDed & the result is given as i/p to the FF_1. i.e., MSB.

CLK	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	0	1	1	1
5	0	0	1	1
6	0	0	0	1

Kokila K.S
Asst. Professor.
B.G.S.I.T.
18

Module-3 ## Flip - Flops

Introduction:

* There are many applications in which digital o/p's are required to be generated in accordance with the Sequence in which the i/p Signals are received. This requirement cannot be Satisfied using a Combinational logic System.
* These applications require o/p to be generated that are not only dependent on the present i/p conditions but they also depend upon the past history of these i/p. The past history is provided by Feedback from the o/p back to the i/p.

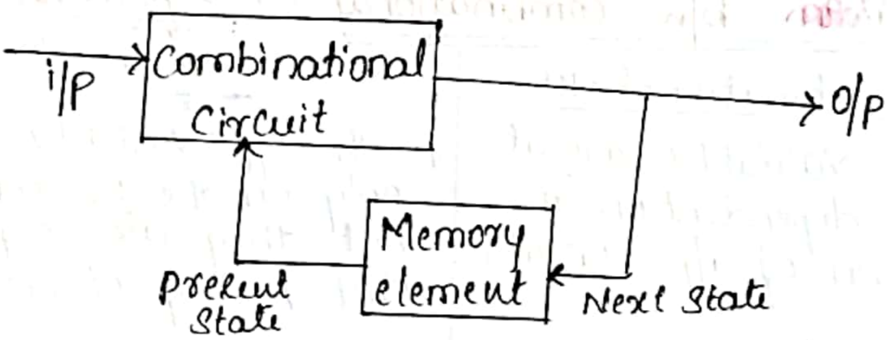


Fig: Block diagram of Sequential CKt/FSM.

Definition: A Sequential Network is defined as a two valued network in which the o/p's at any instant are dependent not only upon the inputs present at that instant but also on past sequence of inputs.

Sequential Circuits have memory to store past-sequence of i/p's.

The term used to represent information stored is called State (or) internal State (or) Secondary state.

Sequential Circuits requires feedback from o/p to i/p.

[Internal State:- It is a Collection of Signals at a set of points within the N/w].

There are 2 types of Sequential ckts based on timing of Signals.

1. Synchronous Sequential network
2. Asynchronous Sequential network.

A Synchronous Sequential network is one in which its behaviour is determined by the values of Signals at only discrete instant of time.
These n/w usually have master clock generator.

Asynchronous Sequential n/w is one in which its behaviour is immediately affected by the input signal changes.
They does not depend on clk signal.

Comparision b/n Combinational & Sequential Circuits.

Combinational ckt	Sequential ckt.
1. The o/p variables are at all times dependent on the Combination of i/p variables.	1. The o/p variables dependent not only on the present i/p variables but they also depend upon the Past history of these i/p variables.
2. Memory unit is not required.	2. Memory unit is required to store the past history of i/p variables.
3. These circuits are faster in speed because the delay b/n i/p & o/p is due to Propagation delay of gates.	3. Sequential Circuits are slower than the Combinational ckts.
4. Combinational ckts are easy to design.	4. Sequential ckts are Comparatively harder to design.
5. Ex:- parallel adder Encoder, decoder, Mux etc.	5. Ex:- Serial adder.

Flip-Flop: It is the basic memory element in Sequential ckt. Flip-Flop is a simple Sequential ckt able to Store One bit of information i.e., '0' or '1'.

* Flip-Flop has feedback & also 2 stable states. It Consists of basic bistable element inwhich appropriate logic is added in order to Control its state.

* Process of storing logic-1 into flip-flop is called Set (or) Preset. & the flip flop is said to be 1-state.

* Process of storing logic-0 into flip-flop is called Clear (or) Reset Condition. & the flip-flop is said to be 0-state.

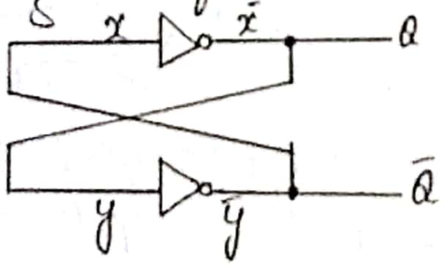
* The inputs to a flip-flop are of 2 types, Asynchronous & Synchronous.

* Asynchronous (or) Direct i/p is One in which signal change produces immediate change in state of flip-flop.

* Synchronous i/p does not immediately affect the State of input o/p changes when some Control input, called enable (or) clock i/p occurs.

Basic Bistable Element: Every Flip-flops Contains basic bistable element.

The basic bistable element is a ckt having two Stable States. Logic diagram of bistable element is shown below,



The ckt has 2 outputs Q & $\bar{Q}$. where $Q \rightarrow$ normal output

$\bar{Q} \rightarrow$ Complementary output

Working: Initially assume $x=0$, $\therefore \bar{x}=1$ & thus $Q=1$ & $\bar{x}$ is i/p to lower NOT gate 'y' becomes 1 i.e., $y=1$ $\therefore \bar{y}=0$ & thus $\bar{Q}=0$. The ckt Continues in this State until power off.

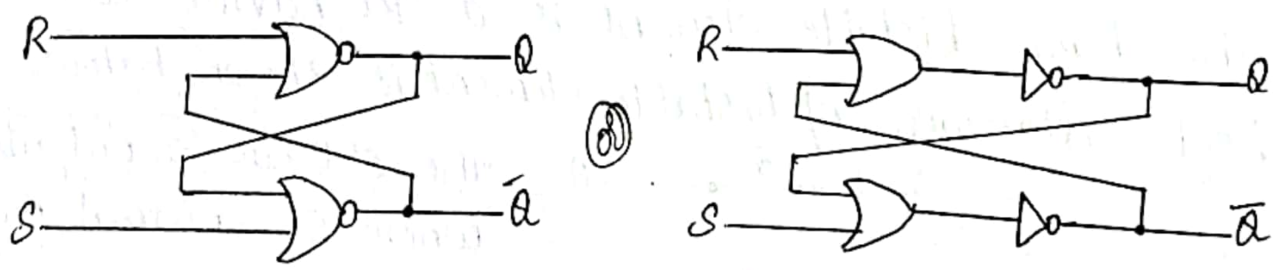
Thus the ckt is stable with $Q=\bar{x}=y=1$ & $\bar{Q}=x=\bar{y}=0$.
IIIly, now assume $x=1$, ∴ $\bar{x}=0$ & thus $Q=0$. This implies $y=0$ ∴ $\bar{y}=1$ & thus $\bar{Q}=1$. Thus, now ckt is stable with $Q=\bar{x}=y=0$ & $\bar{Q}=x=\bar{y}=1$.

The binary symbol [0 & 1] stored in basic bistable element is known as <u>content</u> or <u>state</u> of the element.

Along with 2 stable states bistable element has one more equilibrium condition. This occurs when the 2 o/p signals are halfway b/n logic-0 & 1. Thus o/p is not a valid logic signal. This is known as <u>metastable state</u>. Small change in internal signal due to ckt noise, quickly causes bistable element to leave its metastable state. The amount of time that element can stay in this state is unpredictable.

<u>Latches</u>: are types of flip-flops, in which timing of o/p changes is not controlled.
i.e., o/p responds to changes on the i/p lines immediately, although a special control signal, called enable (or) clock might also need to be present..
latches are not clocked.

S-R latch [Set-Reset latch]



SR latch is constructed using 2 cross coupled NOR gates.
It has 2 i/ps S[set] & R[Reset] & 2 outputs Q & $\bar{Q}$.
Q → Normal o/p, $\bar{Q}$ → Complementary o/p.

Working: when $S=R=0$, the logic diagram simplifies to bistable element. Thus latch is in one of its 2 stable states when there i/ps are applied. In this next state of the device is same as present state i.e., there is no change in latch o/p & present state is retained.

when $S=0$ & $R=1$, regardless of 2nd i/p, upper NOR gates o/p becomes 0 i.e., $Q=0$, since $R=1$, this signal which is fed back to the lower NOR gate along with 0 on 'S' i/p causes o/p of lower NOR gate '1', i.e $\bar{Q}$ to become 1.

Thus latch resets when $S=0$ & $R=1$ i.e., $Q^+=0$ & $\bar{Q}^+=1$.

III^{ly} when $R=0$ & $S=1$, the latch becomes set regardless of present state. ∴ $Q^+=1$ & $\bar{Q}^+=0$.

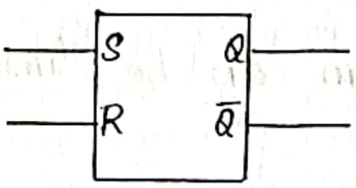
when $S=R=1$, this causes o/p's of both NOR gates to become '0' & they are not complementary. It is difficult to decide the final state if both returns to '0'. & the device may enter its meta-stable state & one i/p should return '0' before the other. Final o/p is determined by the order in which i/p's are changed. ∴ Final state is unpredictable based on construction differences & thermal noise. For this reason & o/p's are not complementary this i/p condition is reffered as forbidden i/p condition.

S	R	Q^+	$\bar{Q}^+$	
0	0	Q	$\bar{Q}$	→ No change
0	1	0	1	→ Reset
1	0	1	0	→ Set
1	1	0^*	0^*	→ Forbidden (or) Indeterminant (or) unpredictable

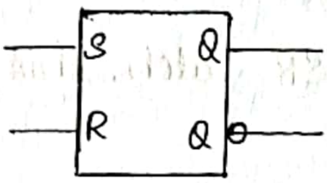
where Q → present state of latch at the time i/p signals are applied.

Q^+ → Next state of latch at Q & $\bar{Q}$ o/p terminals at a consequence of applying various i/ps.

Symbol:-



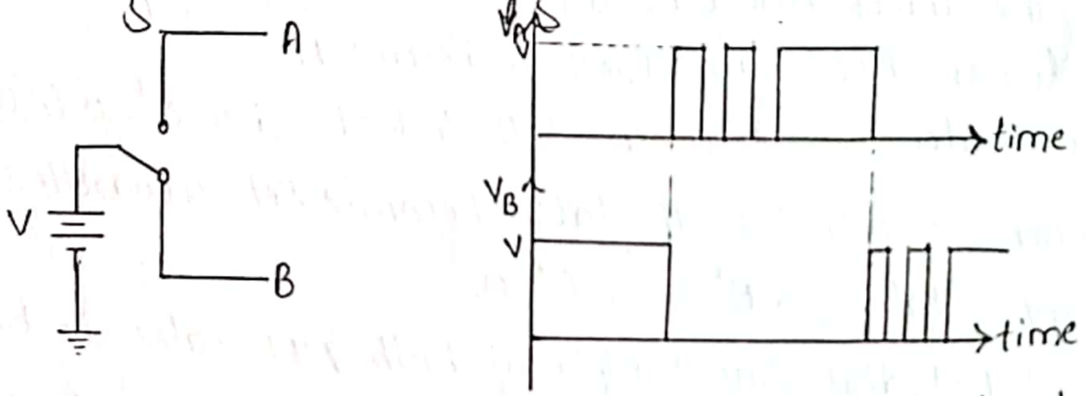
(or)



Application of SR Latch: Switch Debouncer:

* A Simple & important application of SR latch is to eliminate the effect of Contact bounce.

* For interfacing keys to the digital systems, usually push botton keys are used. These push button keys when pressed bounces a few times, closing & opening the contacts before providing a steady reading as shown in fig below.



* Reading taken during bouncing period may be faulty. This problem is known as key debounce.

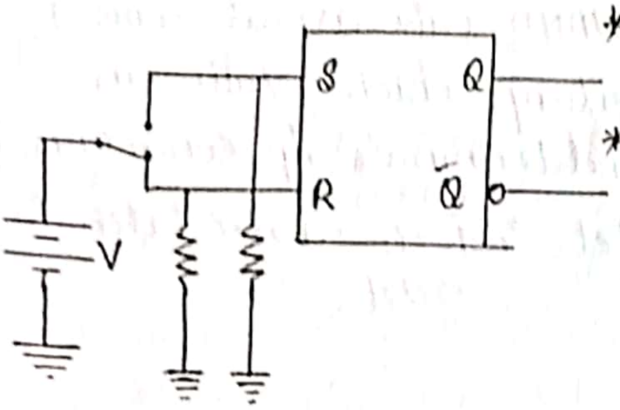
* Key debounce is undesirable & it must be avoided.

* As shown in W/F, when the Center contact of the Switch is in lower position, vtg at B is +V Volts & vtg at A is 'O'V. Now, if contact moved from lower to upper position, vtg at B becomes 'O'V & then the vtg at A is +V volts.

As a result of Contact bounce, the Center Contact of the Switch leaves terminal 'A' Causing Vtg to become 'O' & then return to 'A' again causing vtg to '+V' volts. This Opening & Closing effect due to Springiness of the Contacts may occur several times before the center contact of switch remains in its upper position. During contact bounce, the center bounce doesnot return to terminal B.

|||ly, contact bounce again occurs when the switch is moved from upper to lower position.

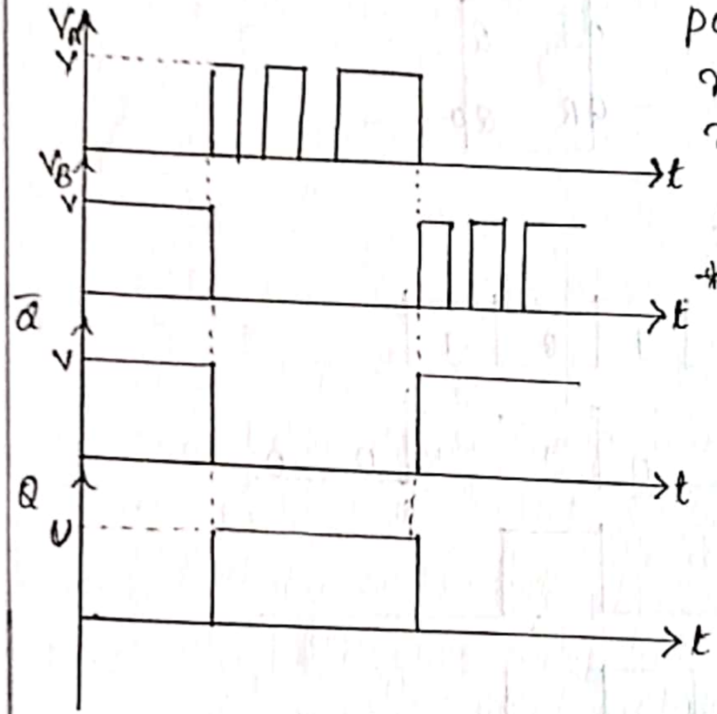
Using SR latch, this problem can be eliminated.



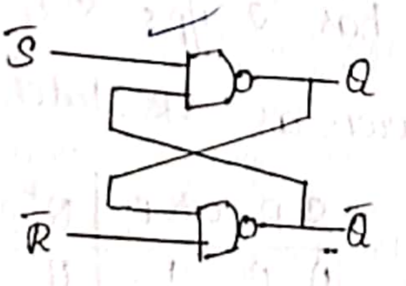
* Assume +ve logic, ∴ +V volts = logic-1 & Gnd = logic-0.

* By the use of 2 pull down resistors, logic-0 values are ensured at S & R terminals of the latch when switch is open. Thus when the center Contact moves from its lower to upper position, SR latch remains in its reset state until Center Contact reaches terminal 'A'. At this time Q o/p of SR latch becomes 1.

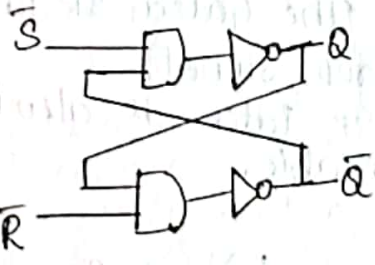
* If the switch now opens as a result of Contact bounce, then '0' input on S & R i/p's of latch causes Q & Q̄ outputs to remain unchanged.



$\bar{S}\bar{R}$-latch :- SR latch constructed using NAND gates is $\bar{S}\bar{R}$ latch.



(or)



$\bar{S}$	$\bar{R}$	Q^+	$\bar{Q}^+$
0	0	1*	1*
0	1	1	0
1	0	0	1
1	1	Q	$\bar{Q}$

when $\bar{S} = \bar{R} = 1$, logic diagram Simplifies basic bistable element. Thus device remains in one of its 2 stable states i.e., latch retains its present state Q & $\bar{Q}$.

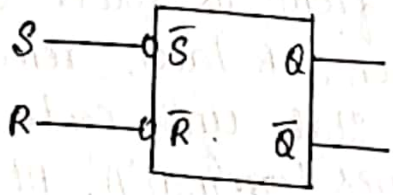
when $\bar{R} = 0$ & $\bar{S} = 1$, R becomes 1, thus resets latch to 0-state i.e., $Q = 0$ & $\bar{Q} = 1$

when $\bar{R} = 1$ & $\bar{S} = 0$, $[S = 1]$, sets the latch to 1-state i.e, $Q = 1$ & $\bar{Q} = 0$.

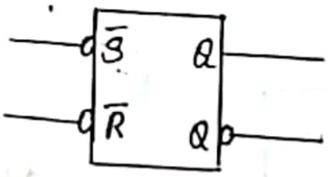
when $\bar{S}=\bar{R}=0$, o/p of both NAND gates are at logic-1. i.e., Q & $\bar{Q}$ o/p's are not Complementory, which results in unpredictable (or) forbidden (or) indeterminate i/p condition.

Thus $\bar{S}=0$ causes latch to set & $\bar{R}=0$ causes latch to reset.

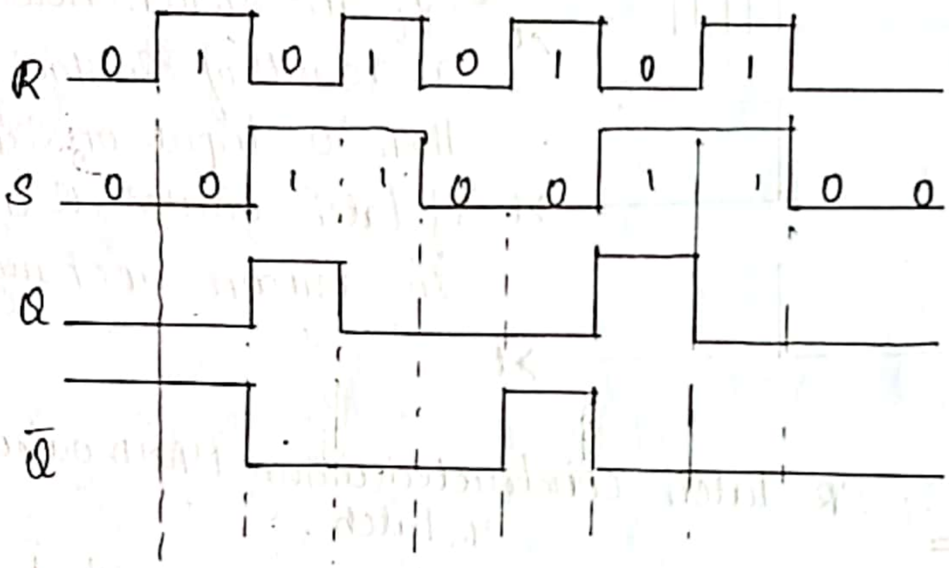
Symbol:



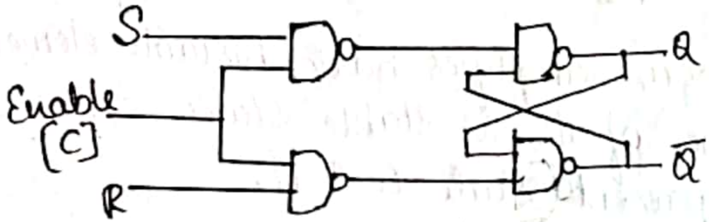
(or)



Timing Diagram of SR Latch:



Gated SR latch: The Gated SR latch has 2 i/p's S & R along with Control Signal. The gated SR latch is also known as SR latch with enable.



S	R	C & E	Q^+	$\bar{Q}^+$
0	0	1	Q	$\bar{Q}$
0	1	1	0	1
1	0	1	1	0
1	1	1	1*	1*
X	X	0	Q	$\bar{Q}$

* It consists of $\bar{S}\bar{R}$ latch along with 2 additional NAND gates & Control input [C]. The Control is also known as enable gate (or) clock input. The 'C' determines when S & R inputs becomes effective.

* when C=0, the o/p's of 1st two NAND gates are at 1, ∴ $\bar{S}=\bar{R}=1$, which keeps the latch in its current stable state. i.e., In this case the latch is said to be disabled.

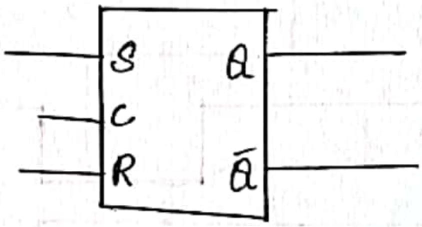
* when C=1, Gated latch is enabled, now the latch behaves like a regular SR latch.

* The 1st two NAND gates are used to invert S & R i/p lines when the latch is enabled.

Thus S=1 sets the latch, R=1 resets the latch & S=R=1, results in unpredictable Condition.

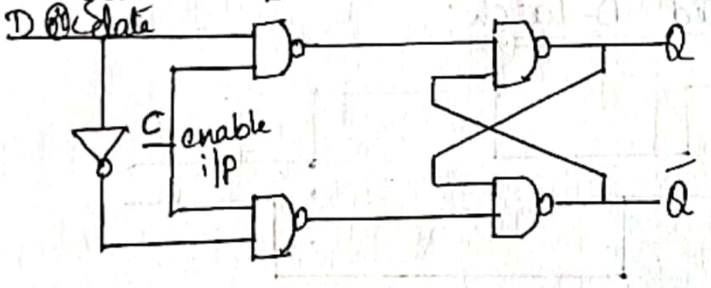
* Since the effects of S & R i/ps are dependent upon the presence of enable signal, these i/p's are known as Synchronous i/p's.

Symbol:-



Gated D-latch:

SR & $\overline{SR}$ latch has uncontrol (or) unpredictable i/p Condt [i.e., when S=R=1 & S=R=0]. This can be avoided using gated D latch [D→data].



D	C	Q^+	$\bar{Q}^+$	
X	0	Q	$\bar{Q}$	→ No change
0	1	0	1	} Q^+ follows D
1	1	1	0	

This ckt. Consists of single i/p D (data) & the control i/p C. Data 'D' determines its next state.

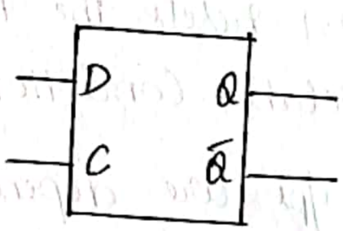
when C=0, there is no Change in latch o/p, It retains Previous State.

when C=1, latch is enabled & its o/p follows values applied to the 'D' i/p.

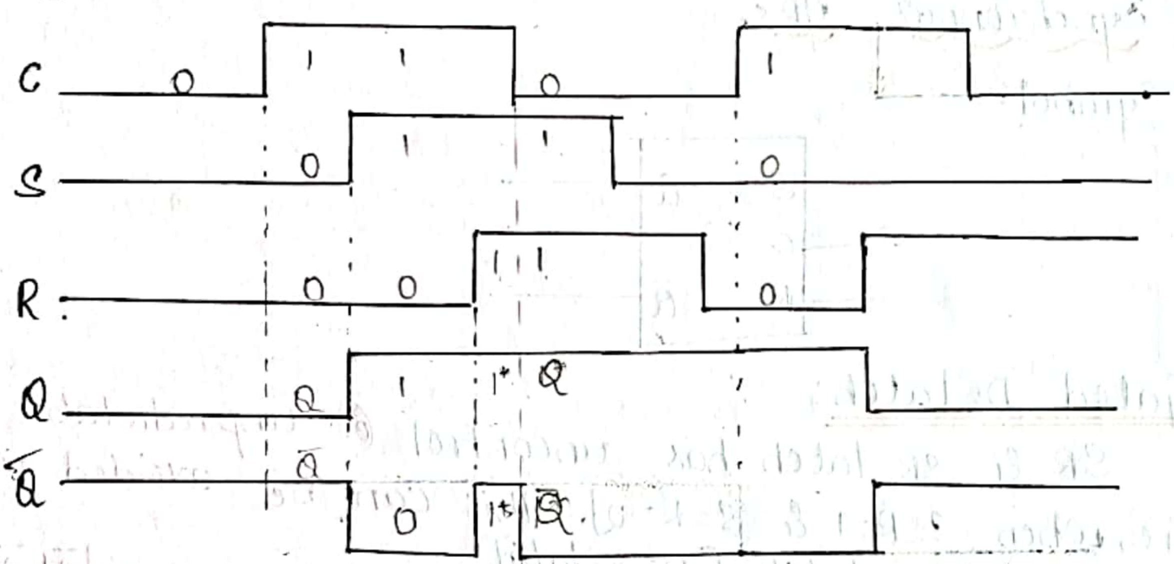
i.e., If D=0, then latch is in 0-State

If D=1, then latch is in 1-State.

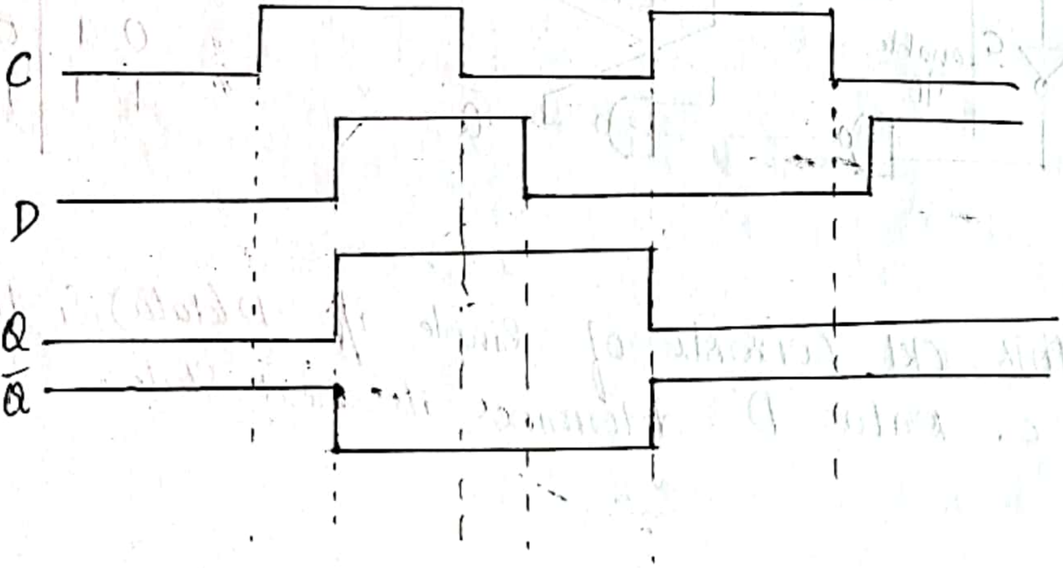
Symbol:-



Timing diagram for gated SR latch:-



Timing diagram for gated D-latch:



Clocked JK Flip-Flop:

The uncertainty in the state of an SR flip-flop when $S=R=1$ can be eliminated by converting it into a JK flip-flop.

The data inputs are J & K which are ANDed with Q & $\bar{Q}$, respectively, to obtain S & R inputs as shown in fig below.

Thus, $S=J.\bar{Q}$ & $R=KQ$.

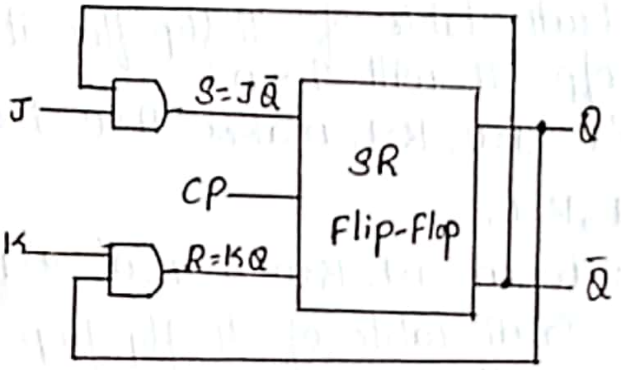


Fig: JK flip-flop using SR flip-flop.

* It consists of SR flip-flop along with 2 additional AND gates.

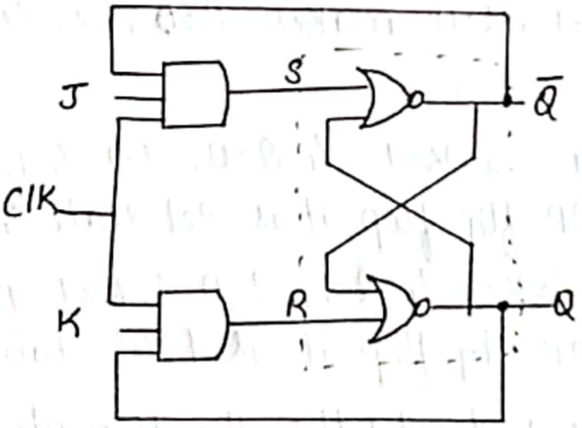


Fig: Clocked JK flip-flop using NOR gates.

* Fig above shows the circuit dig of clocked JK flip-flop using SR latch.

Here, the clock determines the o/p of JK flip-flop.

* when clk=0, the o/p's of two AND gates becomes '0'.
∴ S=R=0, which keeps the latch in its current stable state. i.e, In this case the latch is said to be disabled.

* when C=1, Gated latch is enabled, now the flip-flop generates the o/p with respect to J & K i/p's.

* when, C=1 & J=K=0, S=R=0 & according to the Truth table of SR flip-flop there is no change in the o/p.

* when J=0 & K=1,

a) $Q=0, \bar{Q}=1$: when J=0, K=1 & Q=0, S=0 & R=0.
Since SR=00, there is no change in o/p. ∴ $Q=0$ & $\bar{Q}=1$.

b) $Q=1, \bar{Q}=0$: when J=0, K=1 & Q=1, S=0 & R=1. [∵ R=KQ=1·1].
Acc to truth table of SR flip-flop it is reset state.
& o/p Q will be 0.
i.e, I/p's J=0, K=1, makes Q=0 i.e, reset state.

* when J=1, K=0,

a) $Q=0, \bar{Q}=1$: when J=1, K=0 & Q=0, S=1 $[S=J\bar{Q}]$, & R=0.
Acc to Truth table of SR flip-flop it is set state & o/p 'Q' will be 1.

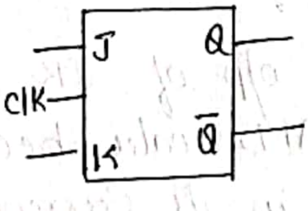
b) $Q=1, \bar{Q}=0$: when J=1, K=0 & Q=0, S=0 & R=0,
Since SR=00 there is no change in o/p. ∴ $Q=1$ & $\bar{Q}=0$.
i.e, I/p's J=1, K=0 makes Q=1, i.e., Set state.

* when J=K=1,

a) $Q=0, \bar{Q}=1$: when J=K=1 & Q=0, S=1 & R=0. According to Truth table of SR flip-flop it is set state & o/p 'Q' will be '1'.

b) $Q=1, \bar{Q}=0$: when J=K=1 & Q=1, S=0 & R=1, According to Truth table of SR flip-flop it is Reset state & o/p 'Q' will be '0'.

∴ The i/p J=K=1, toggles the flip-flop o/p.

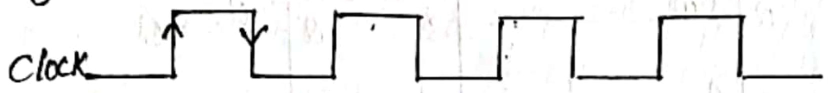


ⓐ Logic Symbol.

clK	J	K	Q	$\bar{Q}$	
0	X	X	Q	$\bar{Q}$	} NO change
⎍	0	0	Q	$\bar{Q}$	
⎍	0	1	0	1	
⎍	1	0	1	0	
⎍	1	1	$\bar{Q}$	Q	→ Toggle.

ⓑ Truth table.

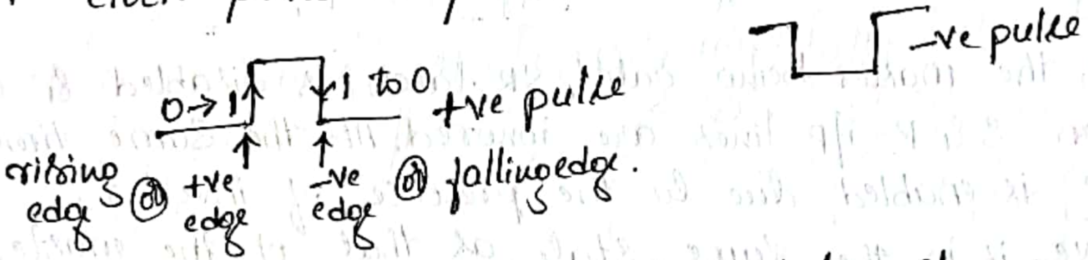
Clock:- Periodic, rectangular waveform used as basic timing signal.



* There are 2 signal transitions,
Transition from 0 to 1 is known as positive edge (or) Rising edge of the clk.
Transition from 1 to 0 is known as negative edge (or) falling edge of the clk.

* One complete pulse includes both transitions[i.e, 0 to 1 & 1 to 0]

* clock pulse may be +ve or -ve.



* The state of flip-flop is changed by change in clk i/p. This change is called trigger & transition it causes is known as triggering the flip-flop.

* Based on clock pulse consideration, there are 2 types of F.F
1. Master-Slave flip-flop (or) pulse triggered (or) level triggered flip flop
2. Edge-triggered flip-flop.

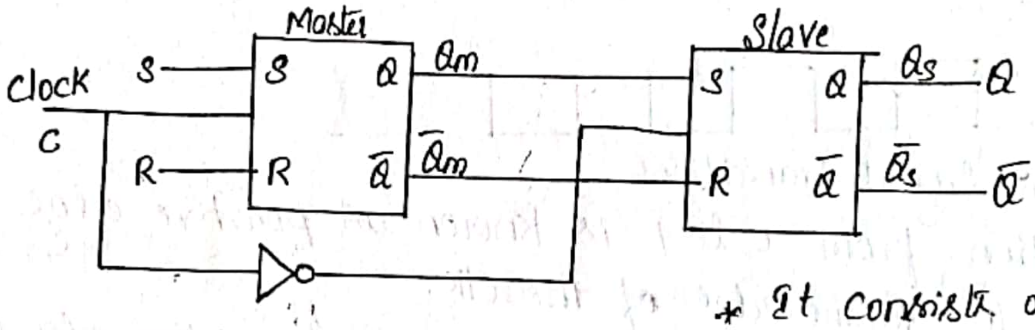
Master-Slave flip flops:

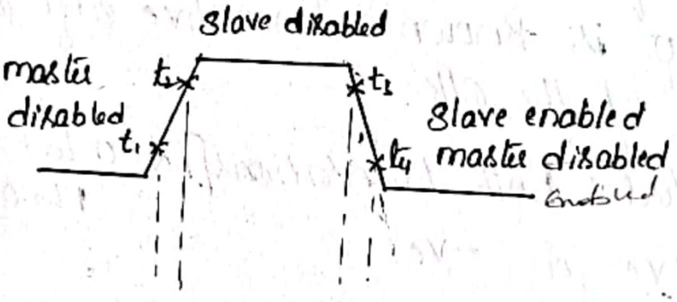
* Master Slave flip-flop consists of 2 cascaded sections each capable of storing a binary symbol.

* The 1st section is called master & the 2nd section is called slave.

* Information entered into the master on one edge of clock signal is transferred to the slave on next edge.

Master Slave SR flip flop:-





* It consists of 2 SR latches & an inverter.
* The i/p lines S & R are used to set & reset the flip-flop.
* A Clock Signal C is applied to control i/p line.

when C=0, the master being gated, SR latch is disabled & any changes on S & R i/p lines are ignored. At the Same time the Slave is enabled due to the presence of inverter. Hence, Slave is in the Same State as that of the master. Since Qm & $\bar{Q}m$ o/p's of Master are Connected to S & R Input of Slave respectively. As the control Signal starts to rise & it is at time 't2' that the master is enabled.

while C=1, the master responds to the i/p's on S & R lines. Since slave is disabled due to presence of inverter. Changes in master latch are not reflected to Slave.

when Control signal returns to low level (logic-0) at time t4, master is disabled & Slave is enabled. Now State of master is transferred to Slave.

Thus o/p change of the master-Slave flip-flop is Synchronized to the falling edge of the Control Signal.

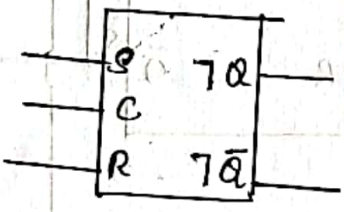
Truth table

S	R	C	Q^+	$\bar{Q}^+$ ← next state.
X	X	0	Q	$\bar{Q}$ → present state
0	0	⎍	Q	$\bar{Q}$
0	1	⎍	0	1
1	0	⎍	1	0
1	1	⎍	undefined.	

⎍ → indicates the master is enabled while the Control signal is high(1) & State of master is transferred to the Slave & Correspondingly to o/p of flip-flop at the end of pulse period.

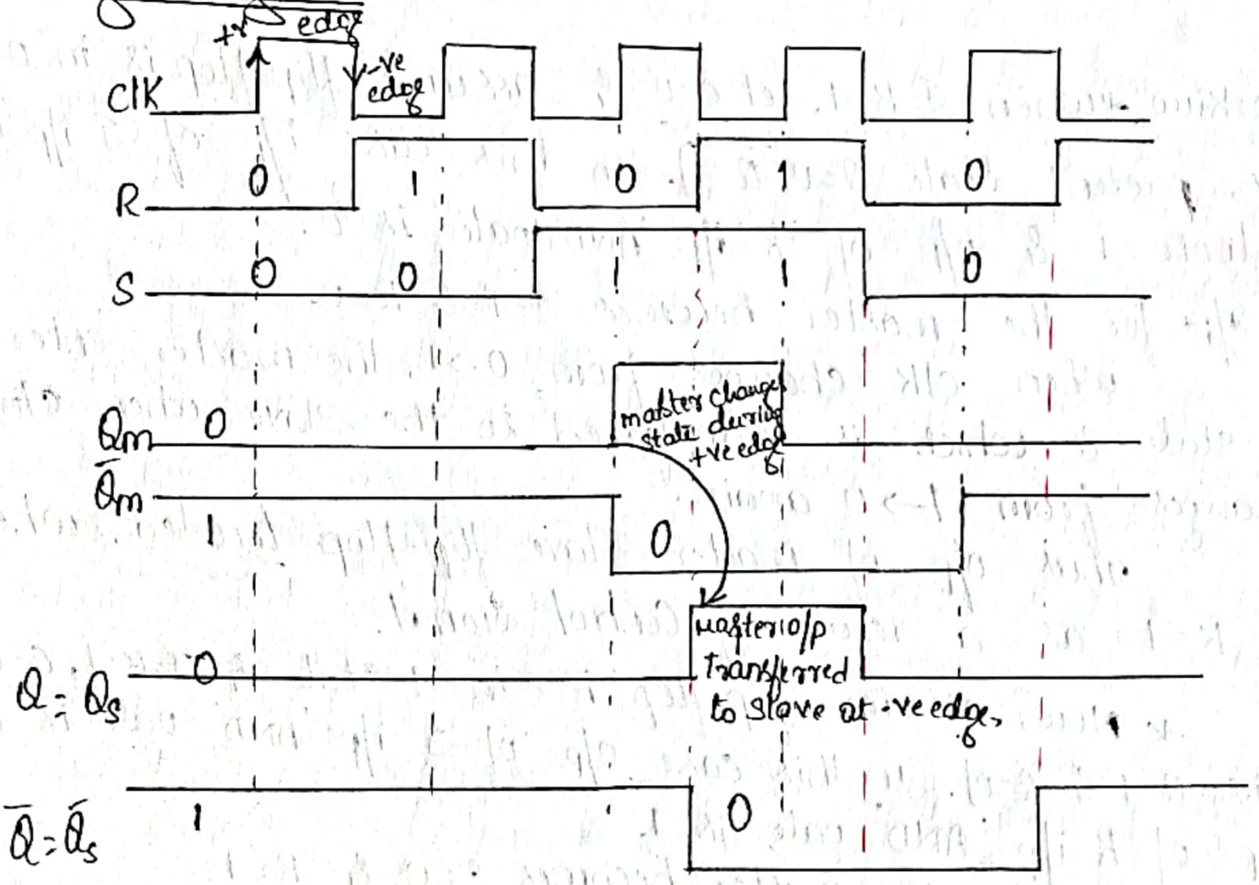
Since the behaviour of master Slave flip-flops Constructed from latches is dependent upon the rising & falling edges of the Control Signal as well as the period of time in which the control is high. They are also known as Pulse Triggered Flip-flop.

Logic Symbol:-



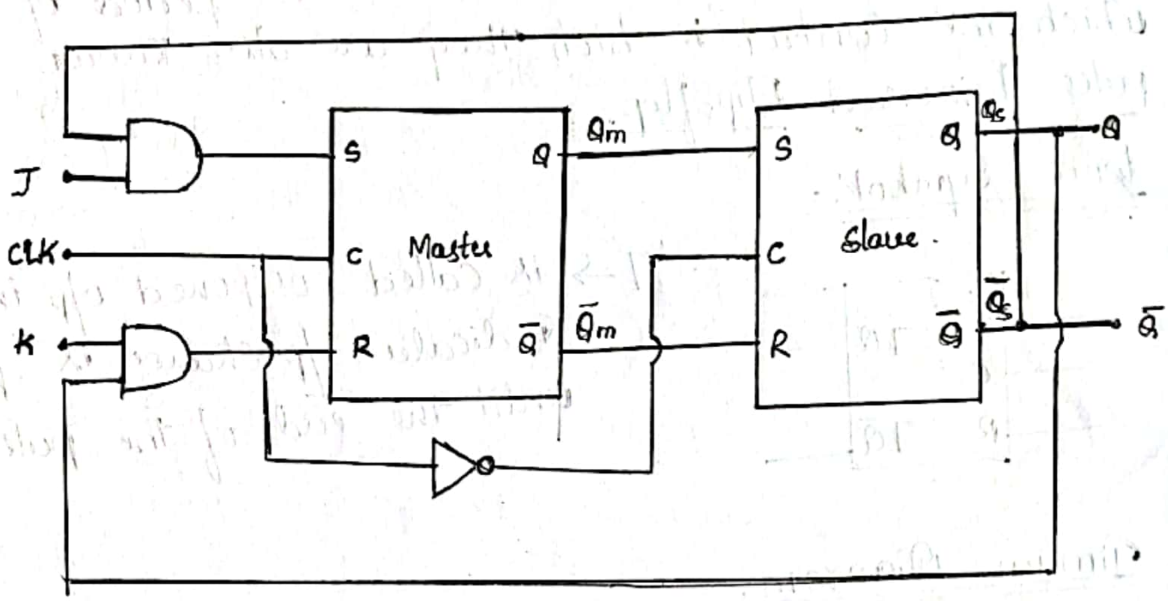
'7' → is called postponed o/p indicator. Indicates o/p change is postponed until the end of the pulse period.

Timing Diagram.


+ve edge
-ve edge
CLK
R
0
1
0
1
0
S
0
0
1
1
0
Qm
0
Q̄m
1
0
master changes state during +ve edge
master o/p transferred to slave at -ve edge.
Q = Qs
0
Q̄ = Q̄s
1
0

Since the o/p state of master Slave SR flip-flop is undefined when S=R=1, it is necessary to avoid this Condition. This is avoided in master slave JK flip-flop.

Master Slave JK Flip-flop!



working:* when $J=K=1$, let $C=0$ & assume flip-flop is in '0-state' [i.e, present State $Q=0$, $\bar{Q}=1$]. In this case o/p of J i/p 'AND' gate is logic-1 & o/p of 'K' i/p 'AND' gate is 0.

∴ I/ps for the master becomes $S=1$ & $R=0$.

when clk changes from 0→1, the master enters 1-state & which is transferred to the slave when clock changes from 1→0 again.

Thus o/p of master slave flip-flop toggeled when $J=K=1$ as a result of Control Signal.

* Now, assume flip-flop is in 1-State & $J=K=1$, $C=0$. [i.e., $Q=1$ & $\bar{Q}=0$]. In this case o/p of J i/p 'AND' gate is 0 & o/p of K i/p 'AND' gate is 1.

∴ I/p's for the master becomes $S=0$ & $R=1$.

when clk changes from 0→1, the master enters to 1-state & which is transferred to the slave when clk changes from 1→0; Thus o/p of flip-flop is toggeled.

* If $J=0$ & $K=1$, the master slave JK flip flop enters to 0-state, '1' on K i/p resets the Q o/p of the flip-flop.

* If J=1, K=0, the master slave JK flip flop entere to 1-state, 1 on J i/p sets the Q o/p of the flip-flop. after clk pulse has accured.

* when J=K=0, the master slave JK flip flop retains its current state during a clk pulse.

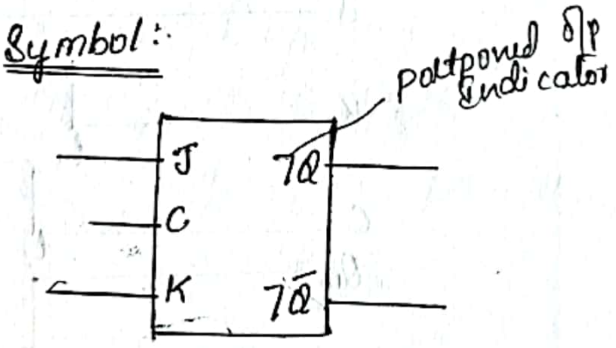
* & when C=0, state of the flip-flop does not change.

Truth table:

C	J	K	Q^+	$\bar{Q}^+$ ← Next state	
⎍	0	0	Q	$\bar{Q}$	→ no change
⎍	0	1	0	1	→ reset
⎍	1	0	1	0	→ set
⎍	1	1	$\bar{Q}$	Q	→ Toggle
□	x	x	Q	$\bar{Q}$	→ no change

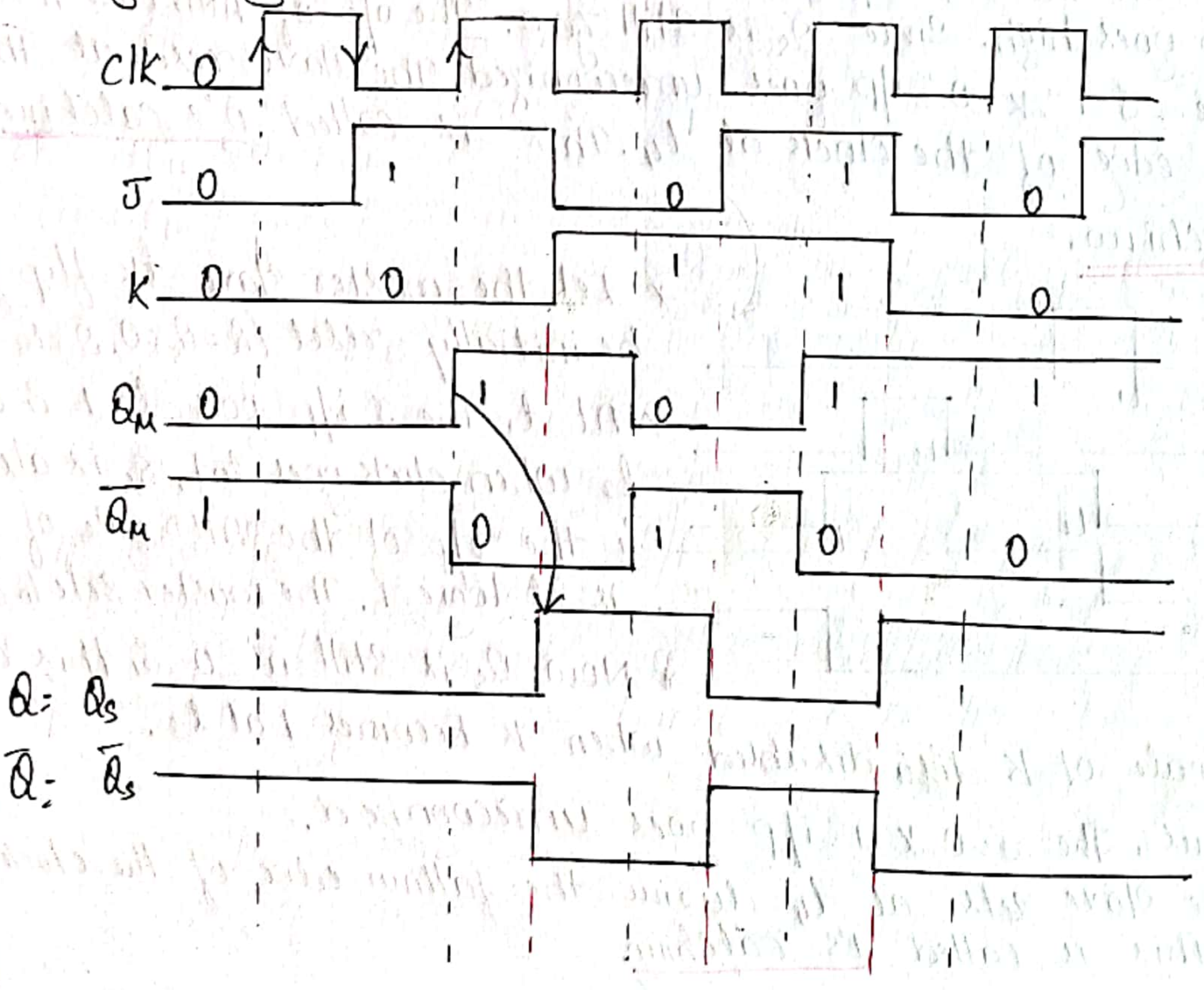
↑ present state

Symbol:



7→ o/p is postponed till the end of pulse period.

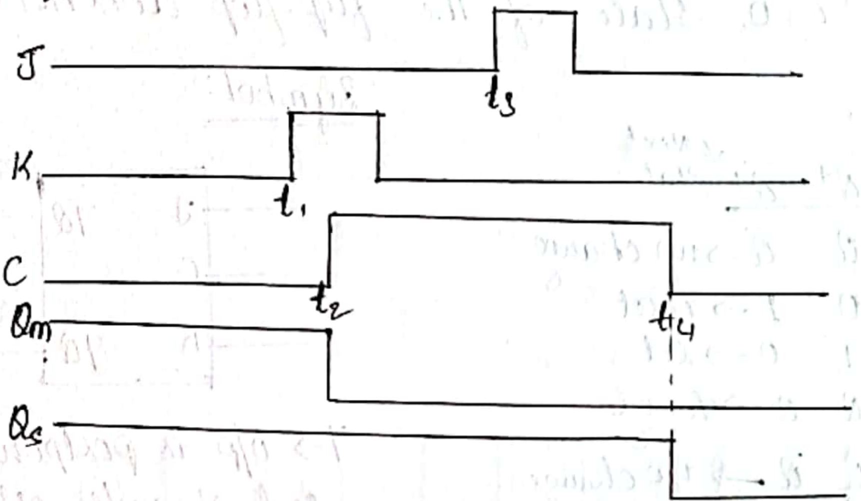
Timing Diagram:



0's catching & 1's catching: This problem occurs in master-slave JK flip-flop

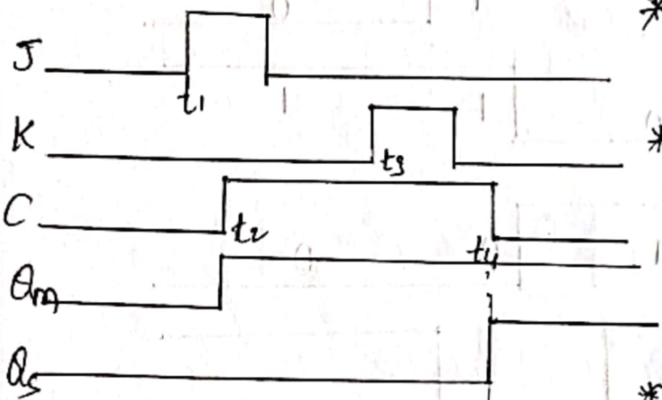
0's catching:

Consider the master-Slave JK flip-flop. Initially, let the flip-flop be set $(Q = Q_s = 1, \bar{Q} = \bar{Q}_s = 0)$. Fig shows the response of the flip-flop to a given i/p sequence.



* At t_1 the K i/p goes to 1. Since $Q_s = 1$, at t_2, when the clock goes high, the o/p of AND gate B goes to 1 & the master resets as shown in the Q_M W/F. Note that Q_s is still at logic 1. At t_3, the J i/p goes high. Since $\bar{Q}_s$ is still zero. The o/p of AND gate A is 0. & the J=1, K=0 i/p goes unrecognized. The slave resets at the falling edge of the clock at t_4. This is called 0's catching.

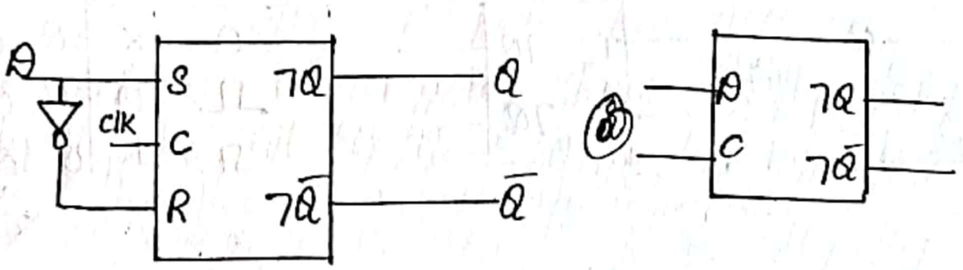
1's catching:



* Let the master slave JK flip-flop be initially reset $(Q = Q_s = 0, \bar{Q} = \bar{Q}_s = 1)$
* At t_1 the J i/p goes to 1 & at t_2 when clock goes to 1, $\bar{Q}_s$ is also 1 & the o/p of the AND gate of J i/p is at logic 1. The master sets to $Q_m = 1$.
* Now Q_s is still at 0 & this keeps AND gate of K i/p disabled when K becomes 1 at t_3.
* Thus, The J=0, K=1 i/p goes unrecognized.
  The slave sets at t_4 during the falling edge of the clock.
* This is Called 1's catching

Master Slave D(Data) flip-flop:-

Symbol:-



Truth table

clk	D	Q^+ $\overline{Q^+}$
0	X	Q $\bar{Q}$
⎍	0	0 1
⎍	1	1 0

* In D flip-flop Q o/p follows D i/p. If D=0, then S=0 & R=1, this resets the flip-flop i.e., Q is in 0-state.

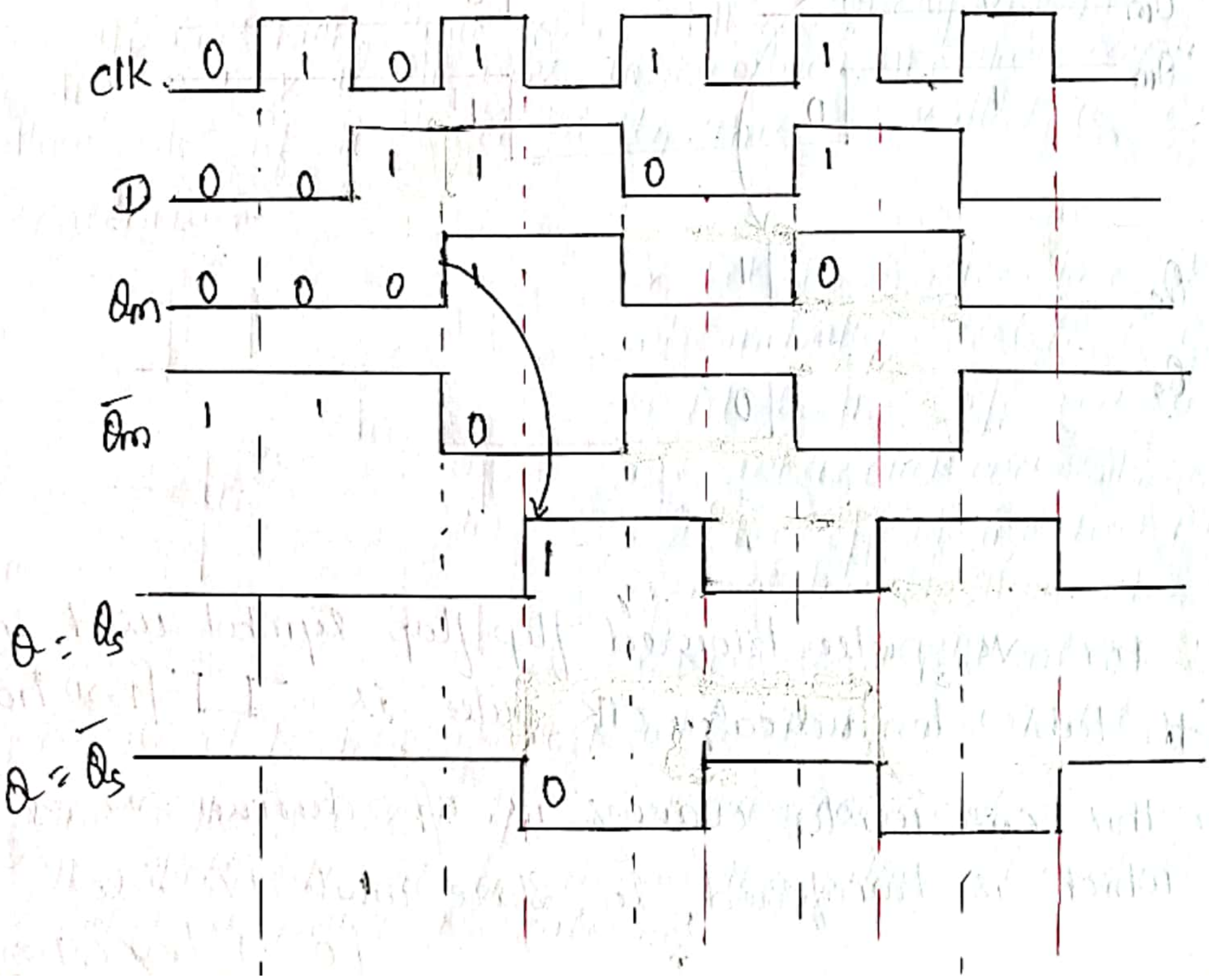
* If D=1, then S=1 & R=0, this sets the flip-flop to 1-state.

∴ If D=0 then Q=0 & if D=1, then Q=1.

Thus Q-follows D with clk pulse.

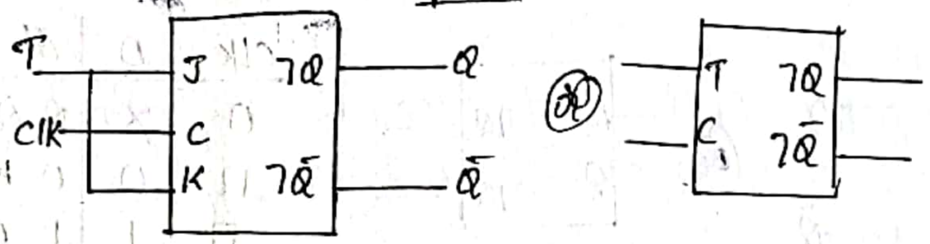
* when C=0, then there is no change in flip-flop o/p.

Timing diagram:



Master Slave T(toggle) Flip-flop:

Symbol

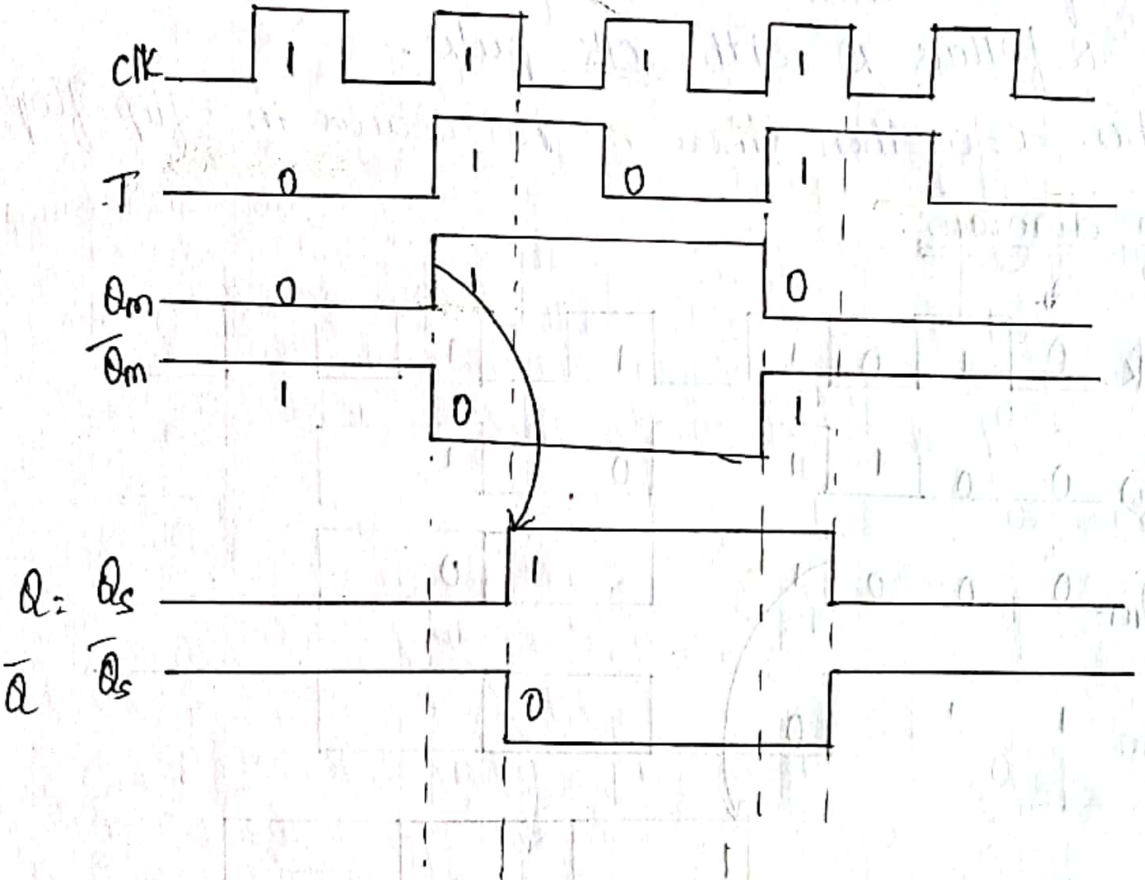


Truth table:

Clk	T	Q	$\bar{Q}$	
0	×	Q	$\bar{Q}$	No change
⊓	0	Q	$\bar{Q}$	
⊓	1	$\bar{Q}$	Q	Toggle.

* when C=0, then there is no change in state of flip-flop.
* If T=0, then J=K=0, thus flip-flop state doesnot change.
* If T=1, then J=K=1, Toggles the flip-flop o/p.

Timing diagram:



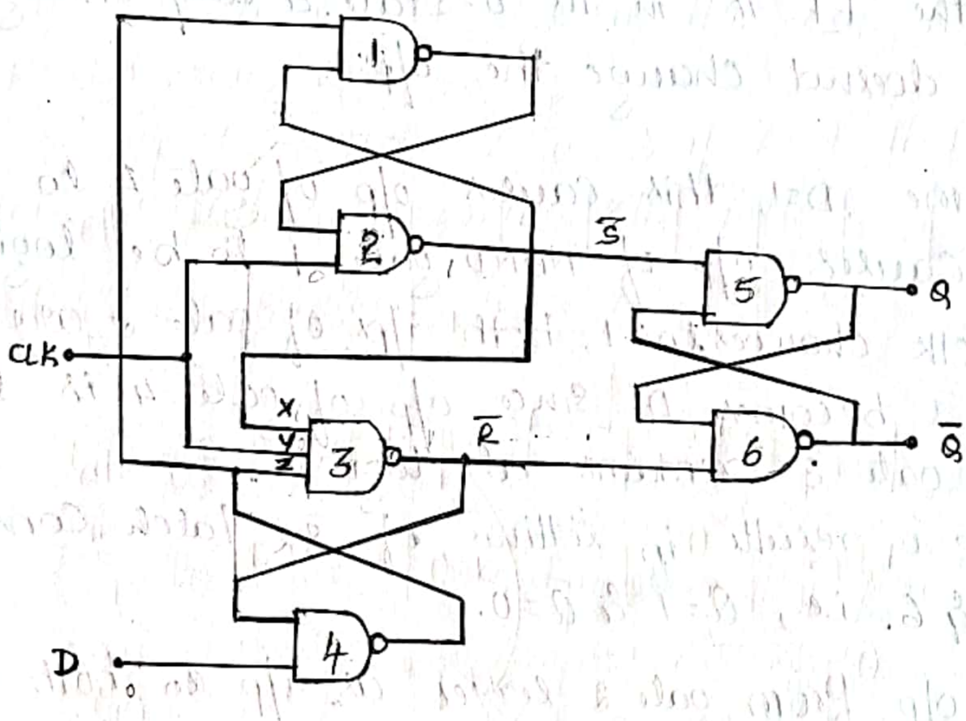
NOTE! For -ve pulse triggered flip-flop, symbol used in Truth table to indicate Clk pulse is ⊔⊓ [1→0 transition]

* In this case master changes its o/p during -ve edge & which is transferred to slave while +ve edge [0→1 transition].

Edge Triggered Flip-Flops:

Edge Triggered flip-flops uses just one edge of clk signal to change its state. This is reffered as Triggering edge. They use either +ve (or) -ve edge of the flip flop. Once Triggering edge occurs, the flip-flop remains unrespon-sive to information i/p changes until the next Triggering edge of control signal.

Positive Edge Triggered D-Flip Flop:



Positive edge Trigger meant that D i/p is transferred to Q o/p [i.e, Q follows 'D'] only upon the accurance of rising (or) +ve edge of the clk signal. This is indicated by (↑) symbol.

Working: Consider the logic diagram shown above, NAND gate 5 & 6 serves as $\bar{S}\bar{R}$ latch.

Initially, assume clock [C] = 0, regardless of D i/p, the o/p of NAND gates 2 & 3 are at 1. ∴ $\bar{S} = \bar{R} = 1$. These signals are applied to $\bar{S}\bar{R}$ latch, causing it to retain its present state.

Thus when c=0, latch retains its present state.

Now, assume D=0, ∴ o/p of NAND gate 4 is at logic-1. Thus i/p's for NAND gate 1 are at logic-1 & corresponding o/p of NAND gate 1 becomes '0'.

when the clock changes from 0 to 1 i.e., +ve edge of the clk, all the 3 i/p's to gate 3 becomes 1, causing o/p of gate to '0'. Thus making $\bar{R}=0$ & $\bar{S}=1$. The Q o/p resets or remains at 0. i.e., Q=0 & $\bar{Q}=1$.

Thus, After accurence of +ve edge of clock signal when D=0, the FF is in its 0-state. & any changes in the 'D' i/p doesnt change the o/p.

* Now assume D=1, this causes o/p of gate 4 to be '0' & this o/p causes o/p of NAND gate 1 to be logic-1. Now when clk changes to 1, both i/ps of gate 2 are '1' & its o/p $\bar{S}$ becomes '0'. Since o/p of gate 4 is logic-0, the o/p of gate 3 remains at logic-1. The $\bar{R}=1$ & $\bar{S}=0$ results in setting of $\bar{S}\bar{R}$ latch consisting of gates 5 & 6. i.e., Q=1 & $\bar{Q}=0$.

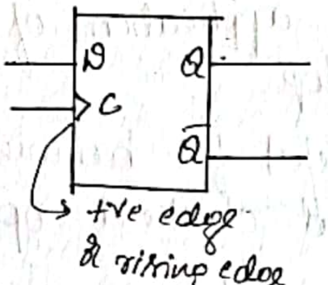
* The '0'-o/p from gate 2 serves as i/p to both gates 1 & 3 & confirms that their o/p's remain at 1. Thus if 'D' subsequently change 1→0 when C=1, causing o/p of gate 4 to change, then the o/p's of gates 1 & 3 do not change. ∴ Once +ve edge of clock has accured, changes in 'D' i/p with C=1 has no effect on state of flip-flop.

* In summary, only upon the accurence of +ve edge of clk signal does the flip-flop respond to the value of D i/p.

Truth table :-

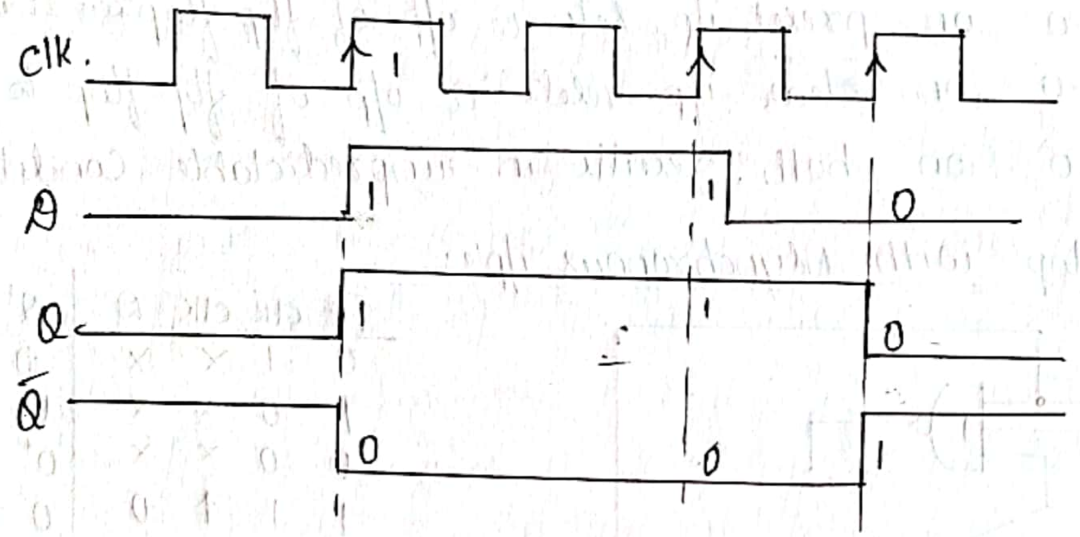
clk	D	Q^+	$\bar{Q}^+$
+ve→ edge ↑	0	0	1 → reset
edge ↑	1	1	0 → set
0	X	Q	$\bar{Q}$ } no change
1	X	Q	$\bar{Q}$

Symbol:



↓ Triangular symbol
→ :- This symbol is called dynamic o/p indicator.
It indicates o/p change occurs only upon transition of control signal.

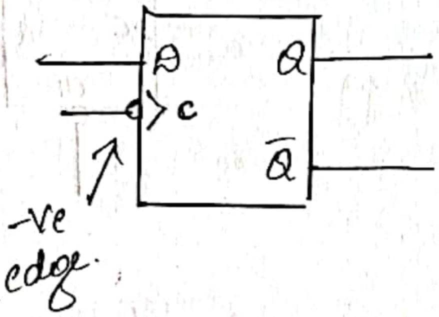
Timing Diagram :-



Negative Edge Triggered D-flip flop :- [1→0 transition]

In this type of flip-flop the negative edge (or) falling edge of clock signal is to change state of flip flop... working principle is same as +ve edge

Symbol :-



clk	D	Q^+	$\bar{Q}^+$
↓	0	0	1 → reset
↓	1	1	0 → set.
0	X	Q	$\bar{Q}$ } No change
1	X	$\bar{\bar{Q}}$	$\bar{Q}$

Asynchronous I/p's:- Asynchronous i/p's does not depends on clock signal. without application of clock i/p their i/ps set & reset the flip flop.

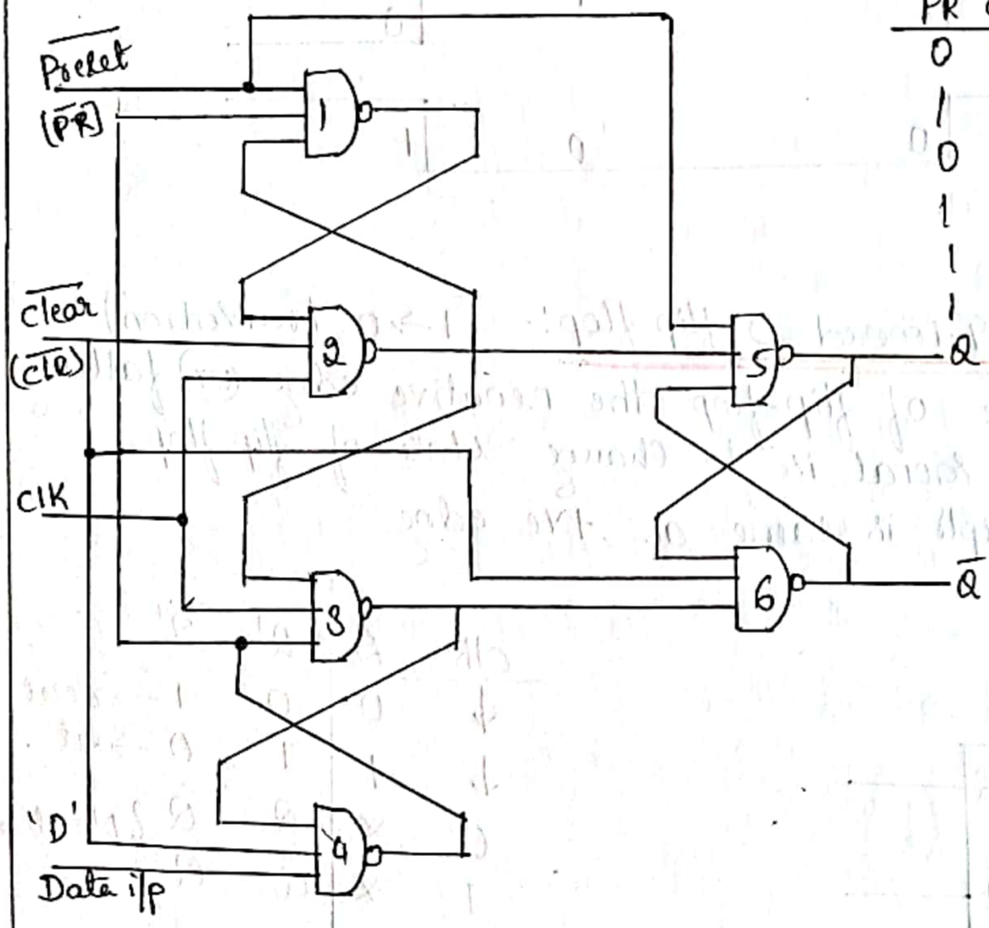
* These i/p's are useful for bringing flip-flop into desired initial state before normal clocked operation.

* The Asynchronous i/p's are preset($\overline{Pr}$) & clear[clr].

These 2 i/p's are active low.

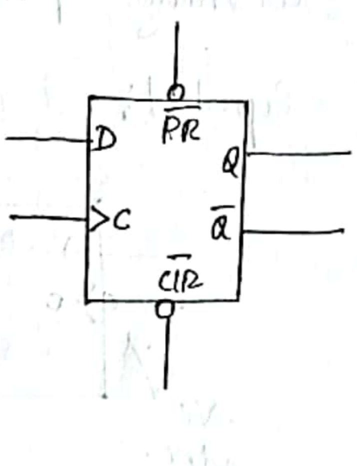
* Logic-0 on $\overline{\text{preset}}$ i/p, sets 'Q' o/p of flip-flop. to 1-State.
logic-0 on $\overline{\text{clear}}$ i/p resets 'Q' o/p of flip-flop to 0-State.
logic-0 on both results in unpredictable condition.

D-flip-flop with Asynchronous i/p's:-


$\overline{\text{Preset}}$
($\overline{PR}$)
$\overline{\text{Clear}}$
($\overline{\text{clr}}$)
clk
'D'
Data i/p
1
2
3
4
5
6
Q
$\overline{Q}$

$\overline{PR}$	$\overline{CLR}$	clk	D	Q^+	$\overline{Q}^+$
0	1	×	×	1	0
1	0	×	×	0	1
0	0	×	×	0^*	0^*
1	1	↑	0	0	1
1	1	↑	1	1	0
1	1	⊗	×	Q	$\overline{Q}$

Symbol:



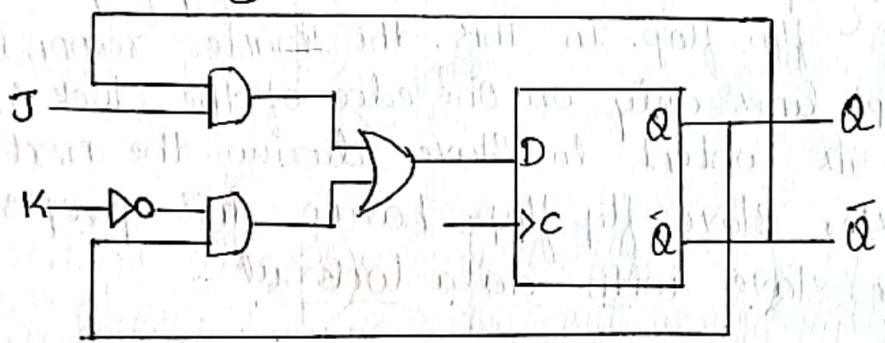
when $\overline{PR}=0$, that sets the flip flop to 1-state, $\overline{clr}=0$, resets the flip flop to 0-state. when $\overline{PR}=\overline{clr}=0$, which results in undefined i/p Condition. when $\overline{PR}=\overline{clr}=1$, it behaves like a D-flip-flop.

working:- when $\overline{PR}=0$ & $\overline{ClR}=1$, with clk[c]=0, then o/p of NAND gate 5 becomes '1' & the o/p of NAND gate 6 is '0'. This Corresponds to Set (or) 1-State.

|||ly, when $\overline{PR}=1$ & $\overline{ClR}=0$ is applied, then o/p of gate 5 becomes '0' & O/p of NAND gate 6 is '1'. This Corresponds to 0-State.

* The $\overline{PR}$ & $\overline{ClR}$ i/p's are also applied to NAND gates 1, 2 & 4. This is done to ensure the effect of an asynchronous i/p on flip flop o/p's when C=1.
i.e., if either $\overline{PR}$ (or) $\overline{ClR}$ becomes 0 with C=1, then flipflop responds immediately.

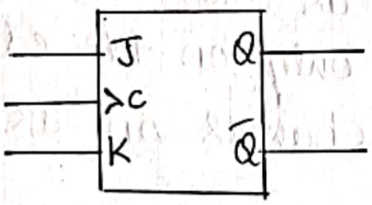
Positive Edge Triggered JK flip-flop:-



Truth table

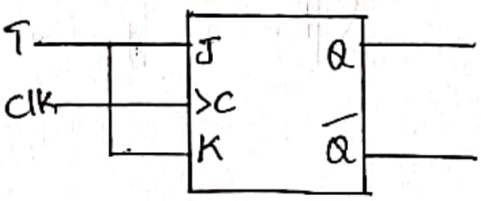
J	K	clk	Q	$\bar{Q}$	
0	0	↑	Q	$\bar{Q}$	
0	1	↑	0	1	
1	0	↑	1	0	
1	1	↑	$\bar{Q}$	Q	
X	X	0	Q	$\bar{Q}$	No change
X	X	1	Q	$\bar{Q}$	

Symbol:

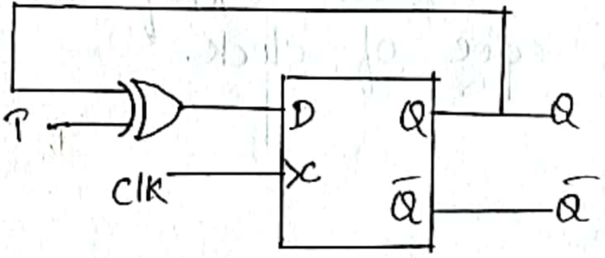


Edge Triggered JK flip flop are not subject to 0's & 1's catching since they respond to the values on information i/p lines only at the time of Triggering edge.

Edge Triggered T flipflop:-



(or)



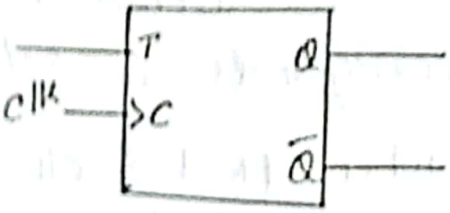
using positive edge Triggered JK flip flop.

Using +ve edge triggered D flip flop.

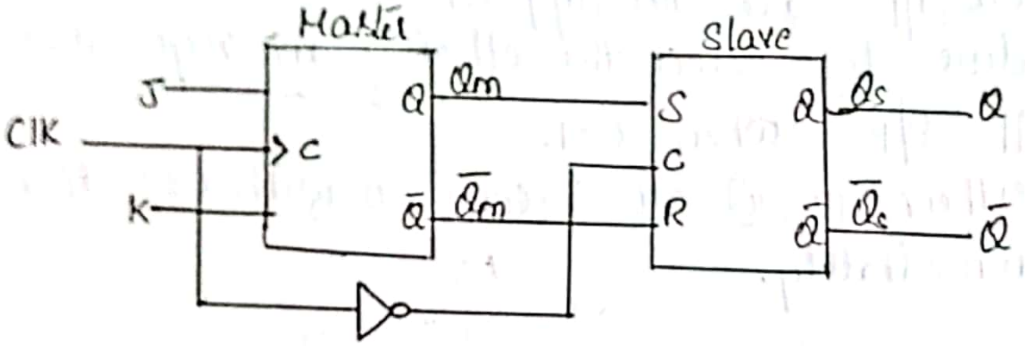
Truth table:

T	C	Q^+	$\bar{Q}^+$	
0	↑	Q	$\bar{Q}$	→ No change.
1	↑	$\bar{Q}$	Q	→ Toggle
x	0	Q	$\bar{Q}$	
x	1	Q	$\bar{Q}$	

Logic Symbol



Master Slave flip flop with data lockout:



O's & 1's Catching problem in master slave flip flop is avoided using this type of flip flop. In this, the master responds to the information(i/p) lines only on one edge of the clock signal & then transfers its content to Slave during the next edge of clock. The master slave flip flop having this property is called Master slave with data lockout.

In the above dig, Positive edge Triggered JK flip flop is used as master & gated SR latch as slave. The Information enters to master only on +ve edge of clock. Since master is edge triggered, any changes on JK i/p lines when C=1 are neglected.

The Content(o/p) of master Transferred to slave at negative edge of clock.

Kokila. K.S
Asst. Professor.
B.G.S.I.T.
18

Module-3

Flip-Flops

Introduction:

* There are many applications in which digital o/p's are required to be generated in accordance with the sequence in which the i/p signals are received. This requirement cannot be satisfied using a combinational logic system.

* These applications require o/p to be generated that are not only dependent on the present i/p conditions but they also depend upon the past history of these i/p. The past history is provided by feedback from the o/p back to the i/p.

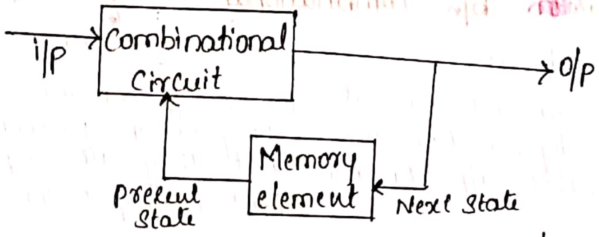


Fig: Block diagram of Sequential ckt/FSM.

Definition: A Sequential Network is defined as a two valued network in which the o/p's at any instant are dependent not only upon the inputs present at that instant but also on past sequence of inputs.

Sequential circuits have memory to store past-sequence of i/p's.

The term used to represent information stored is called State (or) internal State (or) Secondary state.

Sequential circuits requires feedback from o/p to i/p.

[Internal State:- It is a collection of signals at a set of points within the N/w].

There are 2 types of Sequential ckts based on timing of Signals.

1. Synchronous Sequential network
2. Asynchronous Sequential network.

A Synchronous sequential network is one in which its behaviour is determined by the values of signals at only discrete instant of time.
These n/w usually have master clock generator.

Asynchronous Sequential n/w is one in which its behaviour is immediately affected by the input signal changes.
They does not depend on clk signal.

Comparision b/n Combinational & Sequential Circuits.

Combinational ckt	Sequential ckt.
1. The o/p variables are at all times dependent on the combination of i/p variables	1. The o/p variables dependent not only on the present i/p variables but they also depend upon the past history of these i/p variables.
2. Memory unit is not required	2. Memory unit is required to store the past history of i/p variables.
3. These circuits are faster in speed because the delay b/n i/p & o/p is due to propagation delay of gates.	3. Sequential circuits are slower than the combinational ckts.
4. Combinational ckts are easy to design.	4. Sequential ckts are comparatively harder to design.
5. Ex: parallel adder, encoder, decoder, Mux etc.	5. Ex: Serial adder.

Flip-Flop: It is the basic memory element in sequential ckt. Flip-Flop is a simple sequential ckt able to store one bit of information i.e, '0' & '1'.

* Flip-Flop has feedback & also 2 stable states. It consists of basic bistable element inwhich appropriate logic is added in order to control its state.

* Process of storing logic-1 into flip-flop is called Set (or) Preset. & the flip flop is said to be 1-state.

* Process of storing logic-0 into flip-flop is called Clear (or) Reset Condition. & the flip-flop is said to be 0-state.

* The inputs to a flip-flop are of 2 types, Asynchronous & Synchronous.

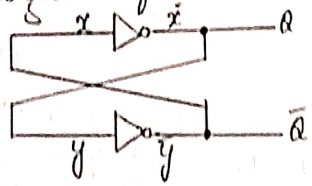
* Asynchronous (or) Direct i/p is one in which signal change produces immediate change in state of flip-flop.

* Synchronous i/p does not immediately affect the state of input, o/p changes when some control input, called enable (or) clock i/p occurs.

Basic Bistable Element: Every Flip-flops contains basic bistable element.

The basic bistable element is a ckt having two stable states.

Logic diagram of bistable element is shown below.



The ckt has 2 outputs Q & $\bar{Q}$.
where Q → normal output
$\bar{Q}$ → Complementary output

Working: Initially assume $x=0$, ∴ $\bar{x}=1$ & thus $Q=1$ & $\bar{x}$ is i/p to lower NOT gate 'y' becomes 1 i.e., $y=1$. ∴ $\bar{y}=0$ & thus $\bar{Q}=0$. The ckt continues in this state until power off.

Thus the ckt is stable with $Q = \bar{x} = y = 1$ & $\bar{Q} = x = \bar{y} = 0$.

|||^{ly}, now assume $x = 1$, $\therefore \bar{x} = 0$ & thus $Q = 0$. This implies $y = 0$ $\therefore \bar{y} = 1$ & thus $\bar{Q} = 1$. Thus, now ckt is stable with $Q = \bar{x} = y = 0$ & $\bar{Q} = x = \bar{y} = 1$.

The binary symbol [0 & 1] stored in basic bistable element is known as Content or State of the element.

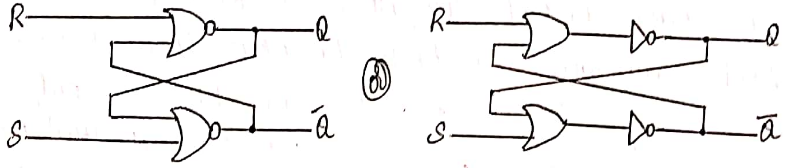
Along with 2 stable states bistable element has one more equilibrium condition. This occurs when the 2 o/p signals are halfway b/n logic-0 & 1. Thus o/p is not a valid logic signal. This is known as metastable state. Small change in internal signal due to ckt noise, quickly causes bistable element to leave its metastable state. The amount of time that element can stay in this state is unpredictable.

Latches: are types of flip-flops, in which timing of o/p changes is not controlled.

i.e., o/p responds to changes on the i/p lines immediately, although a special control signal, called enable or clock might also need to be present..

Latches are not clocked.

S-R latch [Set-Reset latch]



SR latch is constructed using 2 cross coupled NOR gates. It has 2 i/ps S[set] & R[Reset] & 2 outputs Q & $\bar{Q}$.

Q → Normal o/p, $\bar{Q}$ → Complementary o/p.

Working: when $S=R=0$, the logic diagram simplifies to bistable element. Thus latch is in one of its 2 stable states when there i/ps are applied. In this next state of the device is same as present state i.e., there is no change in latch o/p & present state is retained.

when $S=0$ & $R=1$, regardless of 2nd i/p, upper NOR gates o/p becomes 0 i.e., $Q=0$, since $R=1$, this signal which is fed back to the lower NOR gate along with 0 on 'S' i/p causes o/p of lower NOR gate '1', i.e. $\bar{Q}$ to become 1.

Thus latch resets when $S=0$ & $R=1$ i.e., $Q^+=0$ & $\bar{Q}^+=1$.

lll^ly when $R=0$ & $S=1$, the latch becomes set regardless of present state. $\therefore Q^+=1$ & $\bar{Q}^+=0$.

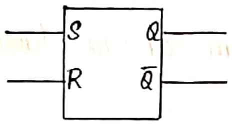
when $S=R=1$, this causes o/p's of both NOR gates to become '0' & they are not complementary. It is difficult to decide the final state if both returns to '0' & the device may enter its meta-stable state & one i/p should return '0' before the other. Final o/p is determined by the order in which i/p's are changed. $\therefore$ Final state is unpredictable based on construction differences & thermal noise. For this reason & o/p's are not complementary this i/p condition is reffered as forbidden i/p condition.

S	R	Q^+	$\bar{Q}^+$	
0	0	Q	$\bar{Q}$	→ No change
0	1	0	1	→ Reset
1	0	1	0	→ Set
1	1	0^*	0^*	→ Forbidden (or) Indeterminant (or) unpredictable

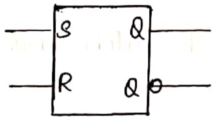
where $Q \rightarrow$ present state of latch at the time i/p signals are applied.

$Q^+ \rightarrow$ Next state of latch at Q & $\bar{Q}$ o/p terminals at a consequence of applying various i/ps.

Symbol:-



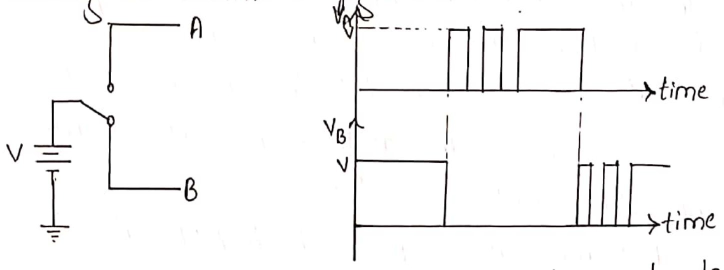
(or)



Application of SR Latch: Switch Debouncer:

* A Simple & important application of SR latch is to eliminate the effect of Contact bounce.

* For interfacing keys to the digital systems, usually push botton Keys are used. These push button keys when pressed bounces a few times, closing & opening the contacts before providing a steady reading as shown in fig below.



* Reading taken during bouncing period may be faulty. This problem is known as key debounce.

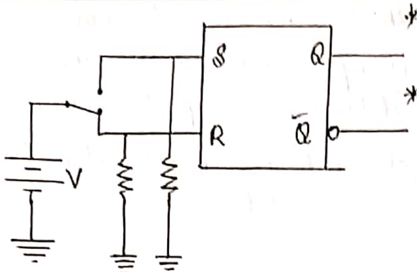
* Key debounce is undesirable & it must be avoided.

* As shown in w/f, when the Center contact of the Switch is in lower position, Vtg at B is +V Volts & Vtg at A is '0'V. Now, if contact moved from lower to upper position, Vtg at B becomes '0'V & then the Vtg at A is +V Volts.

As a result of Contact bounce, the Center Contact of the Switch leaves terminal 'A' Causing Vtg to become '0' & then return to 'A' again causing Vtg to +V volts. This opening & closing effect due to springiness of the contacts may occur several times before the center contact of switch remains in its upper position. During contact bounce, the center bounce doesnot return to terminal B.

|||ly, contact bounce again occurs when the switch is moved from upper to lower position.

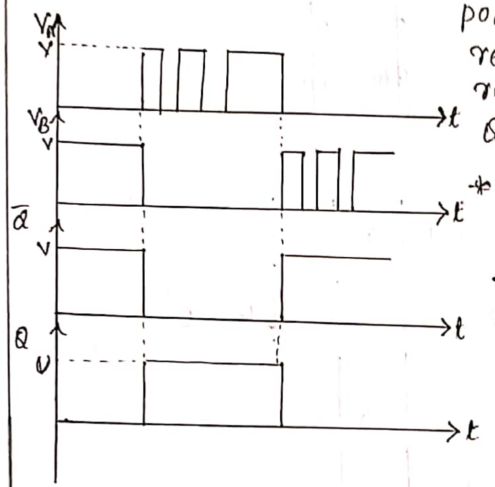
Using SR latch, this problem can be eliminated.



* Assume +ve logic. ∴ +V volts = logic-1 & Gnd = logic-0.

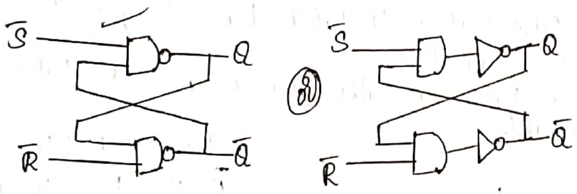
* By the use of 2 pull down resistors, logic-0 values are ensured at S & R terminals of the latch when switch is open. Thus when the center contact moves from its lower to upper position, SR latch remains in its reset state until center contact reaches terminal 'A'. At this time Q o/p of SR latch becomes 1.

* If the switch now opens as a result of contact bounce, then '0' input on S & R i/p's of latch causes Q & $\bar{Q}$ outputs to remain unchanged.



$\bar{S}\bar{R}$-latch:-

SR latch constructed using NAND gates is $\bar{S}\bar{R}$ latch.



$\bar{S}$	$\bar{R}$	Q^+	$\bar{Q}^+$
0	0	1*	1*
0	1	1	0
1	0	0	1
1	1	Q	$\bar{Q}$

when $\bar{S} = \bar{R} = 1$, logic diagram simplifies basic bistable element. Thus device remains in one of its 2 stable states i.e., latch retains its present state Q & $\bar{Q}$.

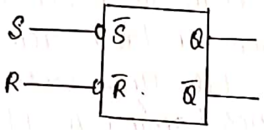
when $\bar{R} = 0$ & $\bar{S} = 1$, R becomes 1, thus resets latch to 0-state i.e., $Q = 0$ & $\bar{Q} = 1$

when $\bar{R} = 1$ & $\bar{S} = 0$, $[S = 1]$, sets the latch to 1-state i.e, $Q = 1$ & $\bar{Q} = 0$.

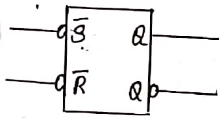
when $\bar{S}=\bar{R}=0$, o/p of both NAND gates are at logic-1. i.e., Q & $\bar{Q}$ o/p's are not complementory, which results in unpredictable (or) forbidden (or) indeterminate i/p condition.

Thus $\bar{S}=0$ Causes latch to set & $\bar{R}=0$ Causes latch to reset.

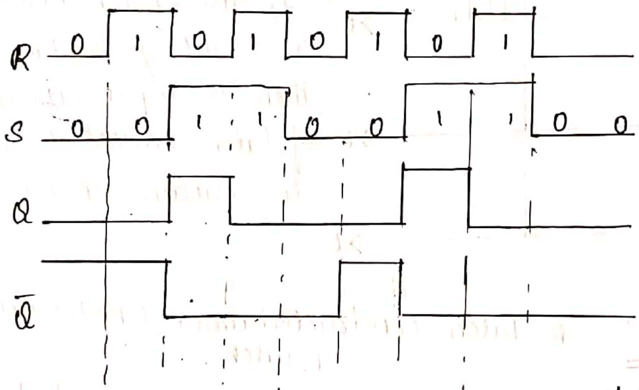
Symbol:



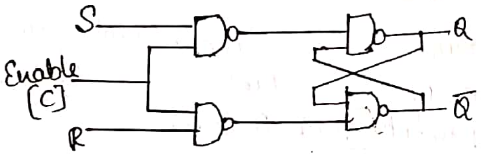
(or)



Timing Diagram of SR Latch:



Gated SR latch: The Gated SR latch has 2 i/p's S & R along with Control Signal. The gated SR latch is also known as SR latch with enable.



S	R	C & E	Q^+	$\bar{Q}^+$
0	0	1	Q	$\bar{Q}$
0	1	1	0	1
1	0	1	1	0
1	1	1	1*	1*
X	X	0	Q	$\bar{Q}$

* It Consists of $\bar{S}\bar{R}$ latch along with 2 additional NAND gates. & Control input[C]. The Control is also known as enable gate (or) clock input. The 'C' determines when S & R inputs becomes effective.

*when C=0, the o/p's of 1st two NAND gates are at 1, ∴ $\bar{S}=\bar{R}=1$, which keeps the latch in its current stable state i.e., In this case the latch is said to be disabled.

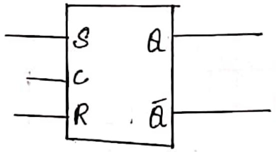
* when C=1, Gated latch is enabled, now the latch behaves like a regular SR latch.

* The 1st two NAND gates are used to invert S & R i/p lines when the latch is enabled.

Thus S=1 sets the latch, R=1 resets the latch & S=R=1, results in unpredictable condition.

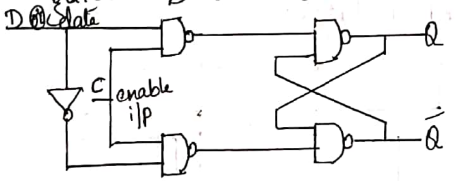
* Since the effects of S & R i/ps are dependent upon the presence of enable signal, these i/p's are known as Synchronous i/p's.

Symbol:-



Gated D-latch:

SR & $\bar{S}\bar{R}$ latch has uncontrol (or) unpredictable i/p Condt [i.e., when S=R=1 & S=R=0]. This can be avoided using gated D latch [D→data].



D	C	Q^+	$\bar{Q}^+$	
X	0	Q	$\bar{Q}$	→ No change
0	1	0	1	} Q^+ follows D
1	1	1	0	

This ckt. consists of single i/p D(data) & the control i/p c. Data 'D' determines its next state.

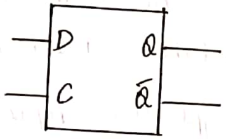
when C=0, there is no Change in latch o/p, It retains Previous State.

when C=1, latch is enabled & its o/p follows values applied to the 'D' i/p.

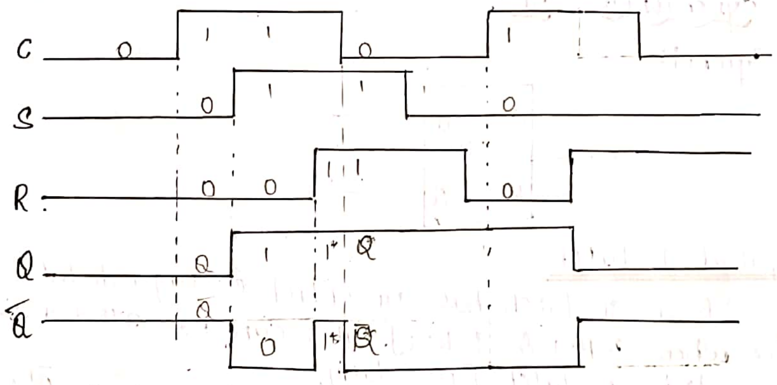
i.e., If D=0, then latch is in 0-State

If D=1, then latch is in 1-State.

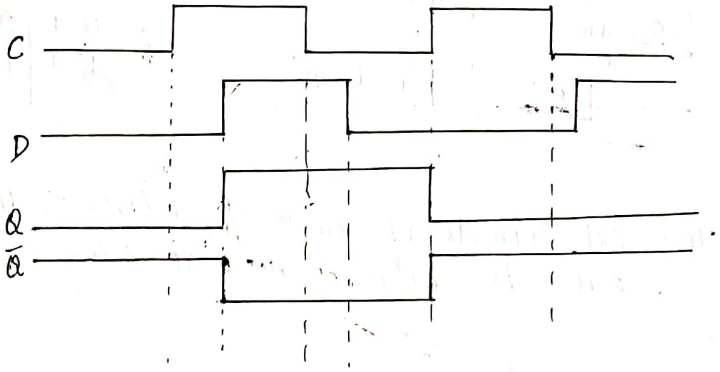
Symbol:-



Timing diagram for gated SR latch:-



Timing diagram for gated D-latch:



Clocked JK Flip-Flop:

The uncertainty in the State of an SR flip-flop when $S=R=1$ can be eliminated by Converting it into a JK flip-flop.

The data inputs are J & K which are ANDed with Q & $\bar{Q}$, respectively, to obtain S & R inputs as shown in fig. below.

Thus, $S=J.\bar{Q}$ & $R=KQ$.

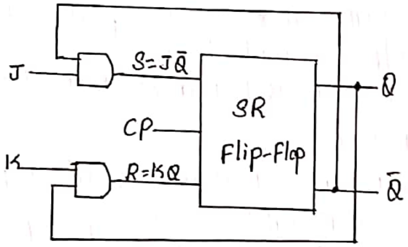


Fig: JK flip-flop using SR flip-flop.

* It Consists of SR flip-flop along with 2 additional AND gates.

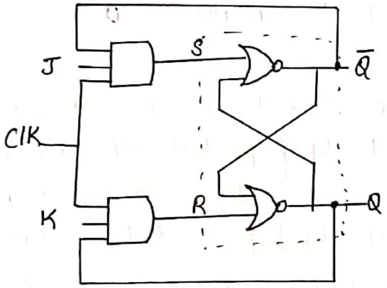


Fig: Clocked JK flip-flop using NOR gates.

* Fig above shows the Circuit dig of Clocked JK flip-flop using SR latch.

Here, the clock determines the o/p of JK flip-flop.

* when Clk=0, the o/p's of two AND gates becomes '0'.
∴ S=R=0, which keeps the latch in its Current stable state. i.e, In this case the latch is said to be disabled.

* when C=1, Gated latch is enabled, now the flip-flop Generate the o/p with respect to J & K i/p's.

* When, C=1 & J=K=0, S=R=0 & according to the Truth table of SR flip-flop there is no change in the o/p.

* when J=0 & K=1,

a) $Q=0, \bar{Q}=1$: when J=0, K=1 & Q=0, S=0 & R=0.
Since SR=00, there is no change in o/p. ∴ Q=0 & $\bar{Q}=1$.

b) $Q=1, \bar{Q}=0$: when J=0, K=1 & Q=1, S=0 & R=1. [∵ R=KQ=1·1].
Acc to truth table of SR flip-flop it is reset state.
& o/p Q will be 0.
i.e, I/p's J=0, K=1, makes Q=0 i.e, reset state.

* when J=1, K=0,

a) $Q=0, \bar{Q}=1$: when J=1, K=0 & Q=0, S=1 [$S=J\bar{Q}$], & R=0.
Acc to Truth table of SR flip-flop it is set state &
o/p 'Q' will be 1.

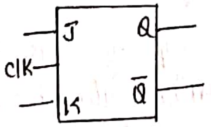
b) $Q=1, \bar{Q}=0$: when J=1, K=0 & Q=0, S=0 & R=0,
Since SR=00 there is no change in o/p. ∴ Q=1 & $\bar{Q}=0$.
i.e, I/p's J=1, K=0 makes Q=1, i.e., Set state.

* when J=K=1,

a) $Q=0, \bar{Q}=1$: when J=K=1 & Q=0, S=1 & R=0. According to Truth table of SR flip-flop it is set state & o/p 'Q' will be '1'.

b) $Q=1, \bar{Q}=0$: when J=K=1 & Q=1, S=0 & R=1, According to Truth table of SR flip-flop it is Reset state & o/p 'Q' will be '0'.

∴ The i/p J=K=1, toggles the flip-flop o/p.

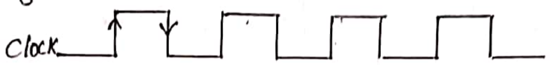


ⓐ Logic Symbol.

ClK	J	K	Q	$\bar{Q}$	
0	×	×	Q	$\bar{Q}$	} No change
⎍	0	0	Q	$\bar{Q}$	
⎍	0	1	0	1	
⎍	1	0	1	0	
⎍	1	1	$\bar{Q}$	Q	→ Toggle.

ⓑ Truth table.

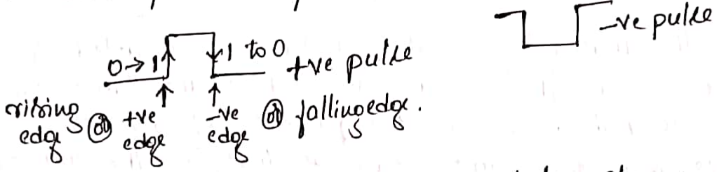
Clock: Periodic, rectangular waveform used as basic timing signal.



* There are 2 signal transitions,
  Transition from 0 to 1 is known as positive edge @ Rising edge of the clk.
  Transition from 1 to 0 is known as negative edge @ falling edge of the clk.

* One complete pulse includes both transitions[i.e, 0 to 1 & 1 to 0]

* clock pulse may be +ve or -ve.



* The state of flip-flop is changed by change in clk i/p. This change is called trigger & transition it causes is known as triggering the flip-flop.

* Based on clock pulse consideration, there are 2 types of F.F
  1. Master-slave flip-flop @ pulse triggered @ level triggered flip flop
  2. Edge-triggered flip-flop.

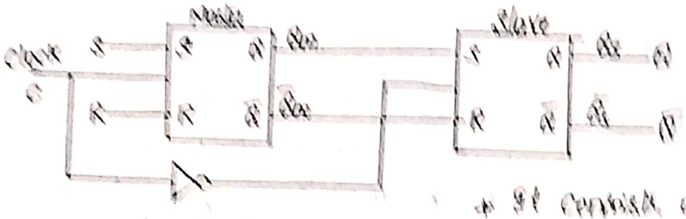
Master-slave flip flops:

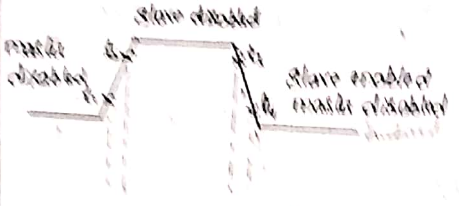
* Master slave flip-flop consists of 2 cascaded sections each capable of storing a binary symbol.

* The 1st section is called master & the 2nd section is called slave.

* Information entered into the master on one edge of clock signal is transferred to the slave on next edge.

Master slave SR flip flop:





- It consists of 2 SR latches & an inverter.
- The i/p lines S & R are used to set & reset the flip-flop.
- A clock signal C is applied to control i/p line.

When C=0, the master being gated SR latch is disabled & any changes on S & R i/p lines are ignored. At the same time the slave is enabled due to the presence of inverter. Hence, slave is in the same state as that of the master. Since Qm & Q̄m o/p's of Master are connected to S & R input of Slave respectively. As the control signal starts to rise & it is at time t1 that the master is enabled.

While C=1, the master responds to the i/p's on S & R lines. Since slave is disabled due to presence of inverter, changes in master latch are not reflected to slave.

When control signal returns to low level (logic-0) at time t4, master is disabled & slave is enabled. Now state of master is transferred to slave.

Thus o/p change of the master-slave flip-flop is synchronized to the falling edge of the control signal.

Truth table

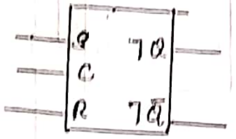
S	R	C	Q^+	$\bar{Q}^+$
X	X	0	Q	$\bar{Q}$
0	0	_⊓_	Q	$\bar{Q}$
0	1	_⊓_	0	1
1	0	_⊓_	1	0
1	1	_⊓_	undefined.	

← next state.
→ present state

_⊓_ → Indicates the master is enabled while the control signal is high(1) & state of master is transferred to the slave & correspondingly to o/p of flip-flop at the end of pulse period.

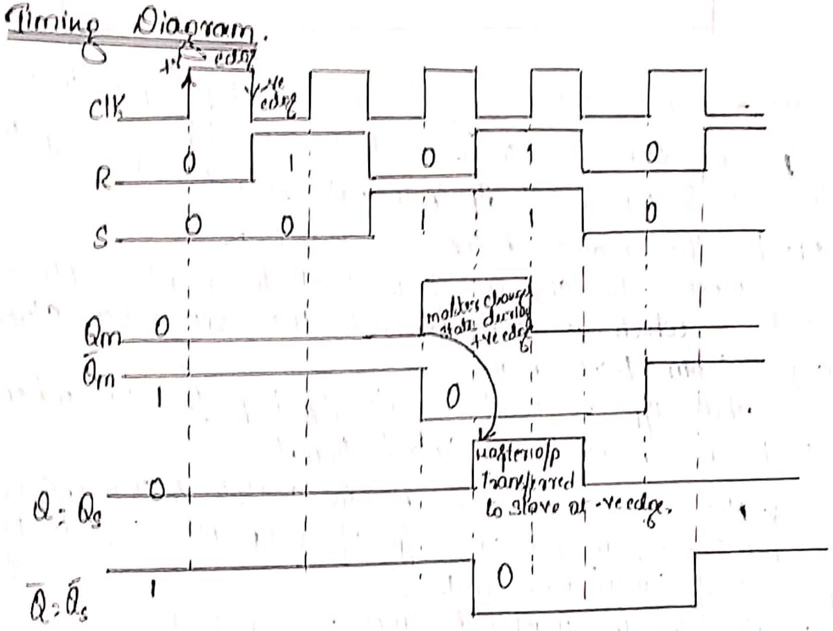
Since the behaviour of master slave flip-flops constructed from latches is dependent upon the rising & falling edge of the Control Signal as well as the period of time in which the control is high. They are also known as Pulse Triggered Flip-flop.

Logic Symbol:



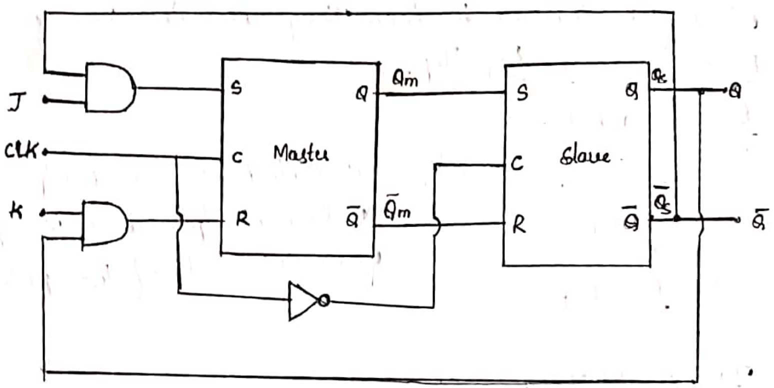
'┐' → is called postponed o/p indicator. Indicates o/p change is postponed until the end of the pulse period.

Timing Diagram.


+ve edge
-ve edge
Clk
R
S
0
1
0
1
0
0
0
1
1
0
Qm
Q̄m
0
1
0
master changes state during +ve edge of
master o/p transferred to slave at -ve edge.
Q = Qs
Q̄ = Q̄s
0
1
0

Since the o/p state of master slave SR flip-flop is undefined when S=R=1, it is necessary to avoid this condition. This is avoided in master slave JK flip-flop.

Master slave JK flip-flop!



working:* when J=K=1, let C=0 & assume flip-flop is in '0-state' [i.e, present state Q=0, $\bar{Q}$=1]. In this case o/p of J i/p 'AND' gate is logic-1 & o/p of 'K' i/p 'AND' gate is 0.

∴ I/Ps for the master becomes S=1 & R=0.

when clk changes from 0→1, the master enters 1-state & which is transferred to the slave when clock changes from 1→0 again.

Thus o/p of master slave flip-flop toggeled when J=K=1 as a result of control signal.

* Now, assume flip-flop is in 1-state & J=K=1, C=0. [i.e., Q=1 & $\bar{Q}$=0]. In this case o/p of J i/p 'AND' gate is 0 & o/p of K i/p 'AND' gate is 1.

∴ I/p's for the master becomes S=0 & R=1.

when clk changes from 0→1, the master enters to 1-state & which is transferred to the slave when clk changes from 1→0, Thus o/p of flip-flop is toggeled.

* If J=0 & K=1, the master slave JK flip flop enters to 0-state, '1' on K i/p resets the Q o/p of the flip-flop.

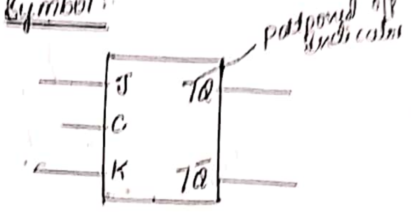
- If J=1, K=0, the master slave JK flip flop enters to 1-state, 1 on J i/p sets the Q o/p of the flip-flop. after clk pulse has occured.
- when J=K=0, the master slave JK flip flop retains its current state during a clk pulse.
- & when C=0, state of the flip-flop does not change.

Truth table:

C	J	K	Q^+	$\bar{Q}^+$
⊓	0	0	Q	$\bar{Q}$ → no change
⊓	0	1	0	1 → reset
⊓	1	0	1	0 → set
⊓	1	1	$\bar{Q}$	Q → Toggle
0	×	×	Q	$\bar{Q}$ → no change

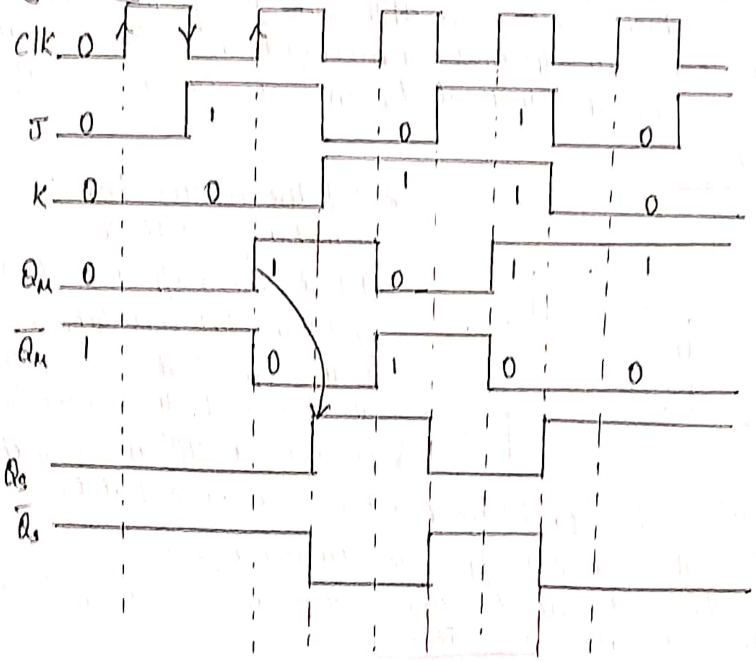
↑ Present state ↙ Next state

Symbol:



7→ o/p is postponed till the end of pulse period.

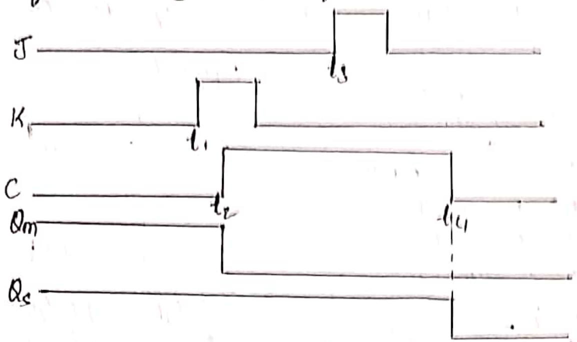
Timing Diagram:



0's catching & 1's catching: This problem occurs in master-slave JK flip-flop

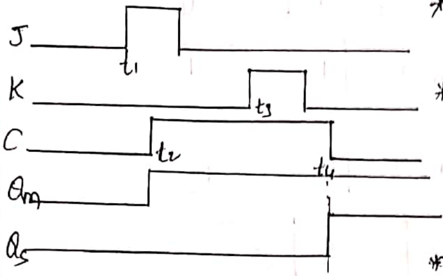
0's catching:

Consider the master-slave JK flip-flop. Initially, let the flip-flop be set ($Q = Q_S = 1$, $\bar{Q} = \bar{Q}_S = 0$). Fig shows the response of the flip-flop to a given i/p sequence.



* At t_1 the K i/p goes to 1. Since $Q_s = 1$, at t_2, when the clock goes high, the o/p of AND gate B goes to 1 & the master resets as Shown in the Q_M W/F. Note that Q_S is still at logic 1. At t_3, the J i/p goes high. Since $\bar{Q}_S$ is still zero. The o/p of AND gate A is 0. & the J=1, K=0 i/p goes unrecognized. The slave resets at the falling edge of the clock at t_4. This is called 0's catching.

1's catching:

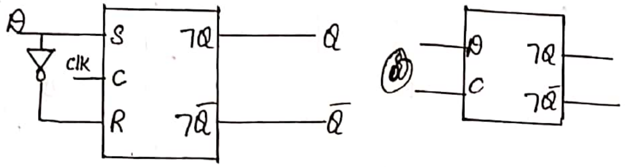


* Let the master slave JK flip-flop be initially reset ($Q = Q_s = 0$, $\bar{Q} = \bar{Q}_s = 1$)
* At t_1 the J i/p goes to 1 & at t_2 when clock goes to 1, $\bar{Q}_s$ is also 1 & the o/p of the AND gate of J i/p is at logic 1. The master sets to $Q_m = 1$.
* Now Q_s is still at 0 & this keeps AND gate of K i/p disabled when K becomes 1 at t_3.
* Thus, The J=0, K=1 i/p goes unrecognized.

The slave sets at t_4 during the falling edge of the clock.
* This is Called 1's Catching.

Master Slave D(Data) flip-flop:-

Symbol:-



Truth table

Clk	D	Q^+	$\bar{Q}^+$
0	X	Q	$\bar{Q}$
⎍	0	0	1
⎍	1	1	0

* In D flip-flop Q o/p follows D i/p. If D=0, then S=0 & R=1, this resets the flip-flop i.e., Q is in 0-state.

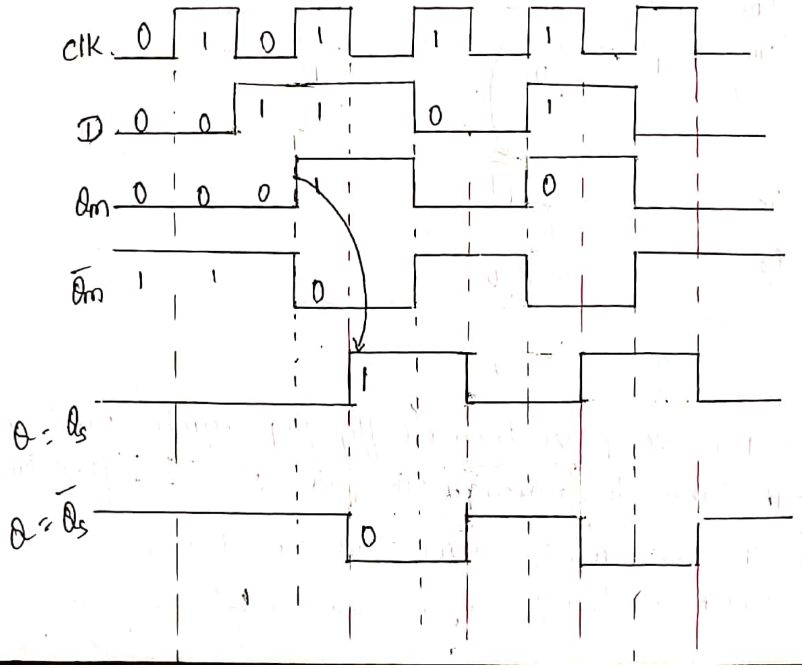
* If D=1, then S=1 & R=0, this sets the flip-flop to 1-state.

∴ If D=0 then Q=0 & if D=1, then Q=1.

Thus Q-follows D with clk pulse.

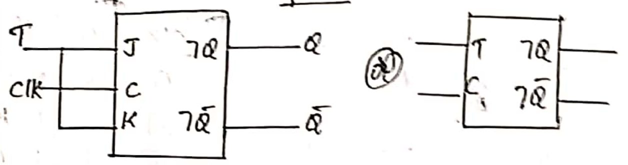
* When C=0, then there is no change in flip-flop o/p.

Timing diagram:-



Master Slave T(toggle) Flip-flop:

Symbol



Truth table:

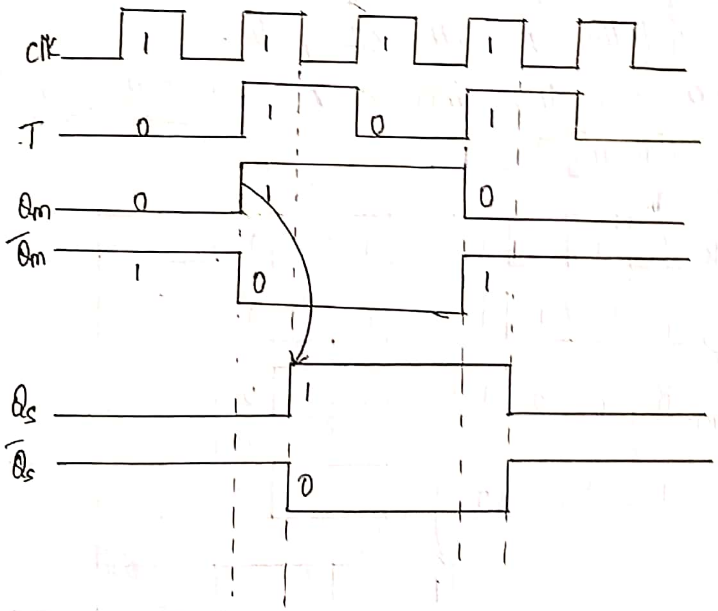
Clk	T	Q	Q̄	
0	x	Q	Q̄	} NO change
⊓	0	Q	Q̄	
⊓	1	Q̄	Q	Toggle.

* when C=0, then, there is no change in state of flip-flop.

* If T=0, then J=K=0, thus flip-flop state doesnot change.

* If T=1, then J=K=1, Toggles the flip-flop o/p.

Timing diagram:



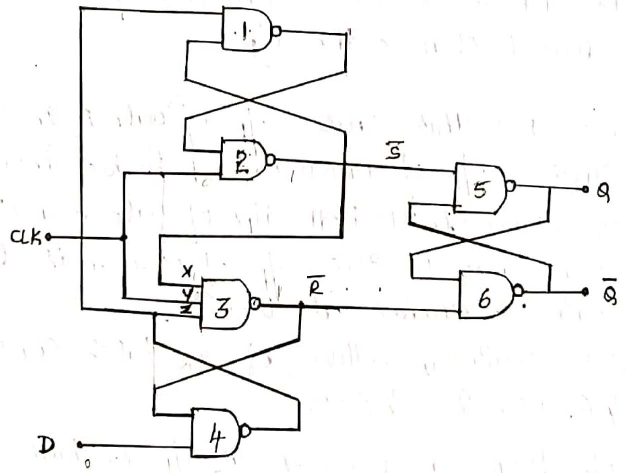
NOTE!. For -ve pulse triggered flip-flop, symbol used in Truth table to indicate Clk pulse is ‾|_|‾ [1→0 transition]

* In this case master changes its o/p during -ve edge & which is transferred to slave while +ve edge [0→1 transition].

Edge Triggered Flip-Flops:

Edge Triggered flip-flops uses just one edge of clk signal to change its state. This is reffered as Triggering edge. They use either +ve (or) -ve edge of the flip flop. Once Triggering edge occurs, the flip-flop remains unrespon-sive to information i/p changes until the next Triggering edge of control signal.

Positive Edge Triggered D-Flip Flop:



Positive edge Trigger meant that D I/p is transferred to Q o/p [i.e, Q follows D] only upon the accurence of rising (or) +ve edge of the clk signal. This is indicated by (↑) symbol.

WORKING: Consider the logic diagram shown above, NAND gate 5 & 6 serves as $\bar{S}\bar{R}$ latch.

Initially, assume clock[C]=0, regardless of D i/p, the o/p of NAND gates 2 & 3 are at 1. ∴ $\bar{S}=\bar{R}=1$. These signals are applied to $\bar{S}\bar{R}$ latch, causing it to retain its present state. Thus when C=0, latch retains its present state.

Now, assume D=0, ∴ o/p of NAND gate 4 is at logic-1. Thus i/p's for NAND gate 1 are at logic-1 & corresponding o/p of NAND gate 1 becomes '0'.

when the clock changes from 0 to 1 i.e., +ve edge of the clk, all the 3 i/p's to gate 3 becomes 1, causing o/p of gate to '0'. Thus making $\bar{R}=0$ & $\bar{S}=1$. The Q o/p resets or remains at 0. i.e., $Q=0$ & $\bar{Q}=1$.

Thus, After accurence of +ve edge of clock signal when D=0, the FF is in its 0-state. & any changes in the 'D' i/p doesnt change the o/p.

* Now assume D=1, this causes o/p of gate 4 to be '0' & this o/p causes o/p of NAND gate 1 to be logic-1. Now when clk changes to 1, both i/ps of gate 2 are '1' & its o/p $\bar{S}$ becomes '0'. Since o/p of gate 4 is logic-0, the o/p of gate 3 remains at logic-1. The $\bar{R}=1$ & $\bar{S}=0$ results in setting of $\bar{S}\bar{R}$ latch consisting of gates 5 & 6. i.e., $Q=1$ & $\bar{Q}=0$.

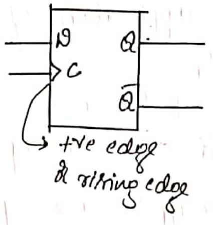
* The '0'-o/p from gate 2 serves as i/p to both gates 1 & 3 & confirms that their o/p's remain at 1. Thus if 'D' subsequently change 1→0 when C=1, causing o/p of gate 4 to change, then the o/p's of gates 1 & 3 do not change. ∴ Once +ve edge of clock has occured, changes in 'D' i/p with C=1 has no effect on state of flip-flop.

* In summary, only upon the accurence of +ve edge of clk signal does the flip-flop respond to the value of D i/p.

Truth table:-

clk	D	Q^+	$\bar{Q}^+$
+ve edge ↑	0	0	1 → reset
↑	1	1	0 → set
0	X	Q	$\bar{Q}$ } no change
1	X	Q	$\bar{Q}$

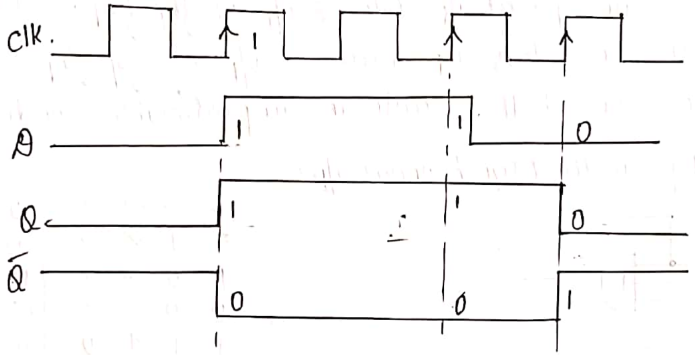
Symbol:



+ve edge & rising edge

→ Triangular symbol :- This symbol is called dynamic o/p indicator.
It indicates o/p change occurs only upon transition of control signal.

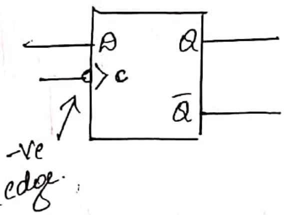
Timing Diagram:.



Negative Edge Triggered D-flip flop:- [1→0 transition]

In this type of flip-flop the negative edge (or) falling edge of clock signal is to change state of flip flop... working principle is same as +ve edge

Symbol:-



-ve edge.

clk	D	Q^+	$\bar{Q}^+$
↓	0	0	1 → reset
↓	1	1	0 → set.
0	X	Q	$\bar{Q}$ } No change
1	X	$\bar{Q}$	$\bar{Q}$

Asynchronous I/p's:- Asynchronous i/p's does not depends on clock signal. without application of clock i/p their i/ps set & reset the flip flop.

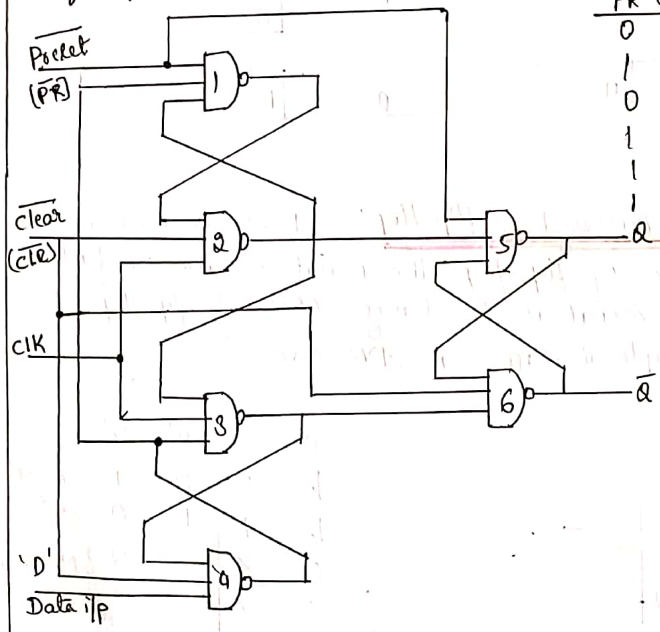
* These i/p's are useful for bringing flip-flop into desired initial state before normal clocked operation.

* The Asynchronous i/p's are preset(Pr) & clear[clr].

These 2 i/p's are active low.

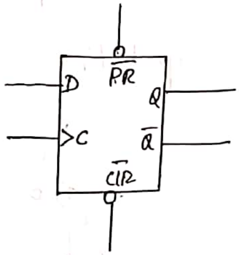
* Logic-0 on $\overline{\text{preset}}$ i/p, sets 'Q' o/p of flip-flop to 1-state.
logic-0 on $\overline{\text{clear}}$ i/p resets 'Q' o/p of flip-flop to 0-state.
logic-0 on both results in unpredictable condition.

D-flip-flop with Asynchronous i/p's:-



$\overline{PR}$	$\overline{CLR}$	clK	D	Q^+	$\overline{Q}^+$
0	1	X	X	1	0
1	0	X	X	0	1
0	0	X	X	0^*	0^*
1	1	↑	0	0	1
1	1	↑	1	1	0
1	1	⊗	X	Q	$\overline{Q}$

Symbol:



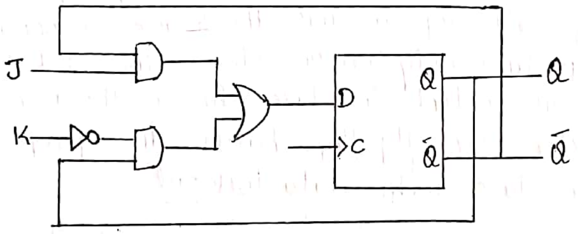
when $\overline{PR}=0$, that sets the flip flop to 1-state, $\overline{clr}=0$, resets the flip flop to 0-state. when $\overline{PR}=\overline{clr}=0$, which results in undefined i/p condition, when $\overline{PR}=\overline{clr}=1$, it behaves like a D-flip-flop.

Working:- when $\overline{PR}=0$ & $\overline{CLR}=1$, with clk[c] = 0, then o/p of NAND gate 5 becomes '1' & the o/p of NAND gate 6 is '0'. This Corresponds to Set or 1-State.

|||ly, when $\overline{PR}=1$ & $\overline{CLR}=0$ is applied, then o/p of gate 5 becomes '0' & O/p of NAND gate 6 is '1'. This Corresponds to 0-State.

*The $\overline{PR}$ & $\overline{CLR}$ i/p's are also applied to NAND gates 1, 2 & 4. This is done to ensure the effect of an asynchronous i/p on flip flop o/p's when C=1. i.e., if either $\overline{PR}$ or $\overline{CLR}$ becomes 0 with C=1, then flipflop responds immediately.

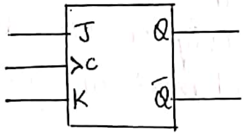
Positive Edge Triggered JK flip-flop:-



Truth table

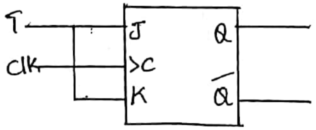
J	K	clk	Q	$\overline{Q}$
0	0	↑	Q	$\overline{Q}$
0	1	↑	0	1
1	0	↑	1	0
1	1	↑	$\overline{Q}$	Q
X	X	0	Q	$\overline{Q}$ } No chang
X	X	1	Q	$\overline{Q}$

Symbol:



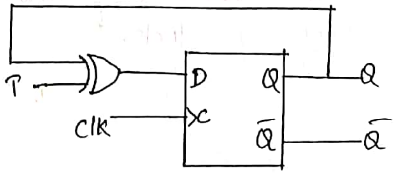
Edge Triggered JK flip flop are not subject to 0's & 1's catching since they respond to the values on information i/p lines only at the time of Triggering edge.

Edge Triggered T flip flop:-



Using positive edge Triggered JK flip flop.

or

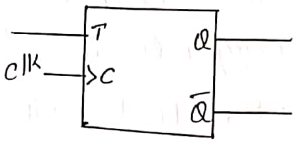


Using +ve edge Triggered D flip flop.

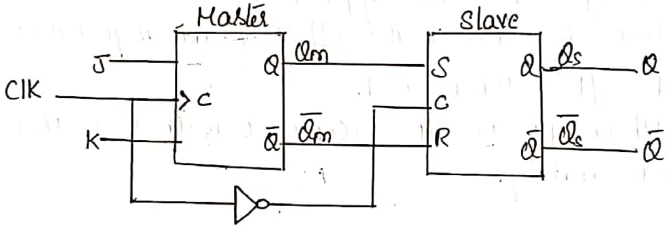
Truth table:

T	C	Q^+	$\bar{Q}^+$	
0	↑	Q	$\bar{Q}$	→ No change.
1	↑	$\bar{Q}$	Q	→ Toggle
x	0	Q	$\bar{Q}$	
x	1	Q	$\bar{Q}$	

Logic Symbol



Master slave flip flop with data lockout:



O's & 1's catching problem in master slave flip flop is avoided using this type of flip flop. In this, the master responds to the information (i/p) lines only on one edge of the clock signal & then transfers its content to slave during the next edge of clock. The master slave flip flop having this property is called Master slave with data lockout.

In the above dig, Positive edge Triggered JK flip flop is used as master & gated SR latch as slave. The Information enters to master only on +ve edge of clock. Since master is edge triggered, any changes on JK I/p lines when C=1 are neglected.

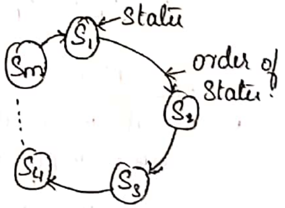
The Content(o/p) of master Transferred to slave at negative edge of clock.

Counters.

* A Sequential circuit that generates prescribed sequence of states upon application of clock pulses is known as Counter.

(or)

* Counter is a cascaded arrangement of flip-flops configured to o/p a specific sequence on application of a clock.
* Counters are used for counting number of clock pulses arriving at its clock input & are useful for generating timing sequences to control operations in a digital system.
* Each o/p of the sequence is dependent on the contents of the flip-flops & is called a state of a counter.
* The modulus of a counter is the total no of states of the counter.
* If counter is cascade of n flip-flops, then no possible states are 2^n. The no of states may be $\le 2^n$.
  i.e., If counter has 'm' (2^n) distinct states, then it is called modulus-m (or) mod-m counter.
* The order in which states appears is called Counting Sequence
* Graphical representation of counting sequence is called "State diagram".
* Each node S_i indicates states of the counter & arrows in the graph denotes order in which states occur.



Synchronous Counter : When counter is clocked such that each flip-flop in the counter is triggered at the same time, that counter is said to be synchronous counter.

Asynchronous Counter/Ripple Counter: In this type of counter flip flops are not clocked simultaneously & flip flops are connected in such a way that output of first flip flop drives the clock for the next flip flop.

* Depending on select lines of multiplexers [mode Control lines] the register can retain its Current State, Shift left, Shift right. Each of these operations is the result of +ve edge of clock signal.

* logic-0 on the Asynchronous I/p $\overline{clr}$, clears register Contents

* when $S_1S_0 = 00$, I_0 of the mux is selected. The Q o/p of flip flops are connected to their respective 'D' i/p's & upon the occurrence of the clock pulse 'D' i/p's appears at 'Q' o/p.
  ∴ There is no change in register Content.

* when $S_1S_0 = 01$, I_1 of mux is selected, Serial input for shift right gets connected to 'D' i/p of flip flop A, Q_A to D_B, Q_B to D_C, Q_C to D_D. Now, upon the occurrence of the clock pulse, register shift the data to right by one bit position.

* when $S_1S_0 = 10$, I_2 of mux is selected, Serial input for left shift get connected to D i/p of flip flop D, Q_D to D_C, Q_C to D_B & Q_B to D_A, Now upon the occurrence of clock pulse, register shifts the data to left by one bit position. i.e., it performs left shift operation.

* when $S_1S_0 = 11$, I_3 is selected & I_A, I_B, I_C, I_D appears at D_A, D_B, D_C, D_D respectively constituting parallel inputs.

Applications of Shift Register:

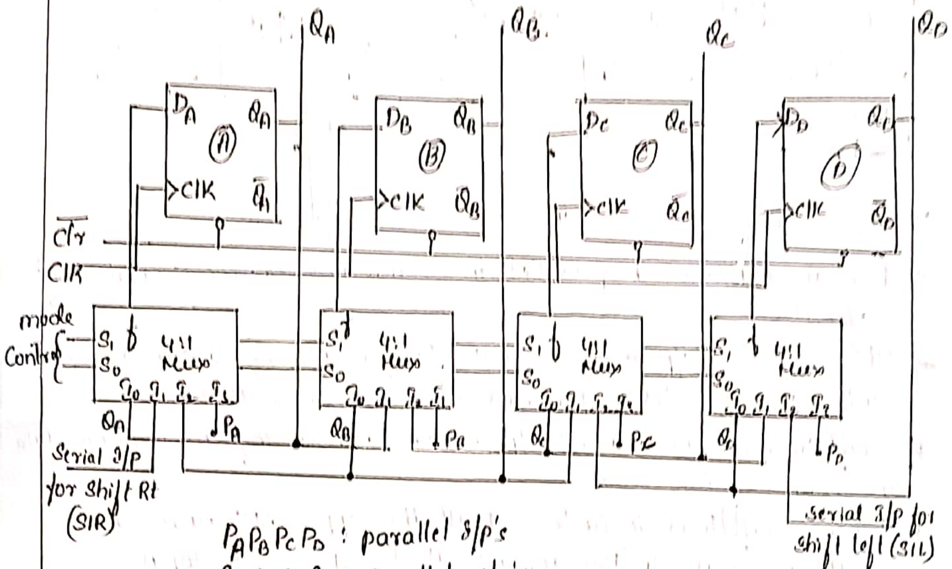
1. Serial In Serial out [SISO] shift register Can be used introduce time delay in digital Signals.
2. Serial In parallel out [SIPO] shift register Can be used to Convert data in serial form to the parallel form.
3. Parallel In Serial out [PISO] shift register can be used to Convert data in parallel form to Serial form.
4. Shift register can also be used as Counter, to identify time sequences for Specific instances.

Universal Shift Register

* A universal shift register is one which is bi-directional & has capabolities to accept both serial & parallel inputs as well as capabilities of serial & parallel o/p.

* Fig below shows the 4-bit universal shift register. It has all the capabilities listed above.

* It consists of 4 flip-flops & 4 multiplexers.


QA
QB
QC
QD
DA QA (A) >clk $\bar{Q}_A$
DB QB (B) >clk $\bar{Q}_B$
DC QC (C) >clk $\bar{Q}_C$
DD QD (D) >clk $\bar{Q}_D$
$\overline{Clr}$
Clk
mode Control
S1 S0 4:1 Mux I0 I1 I2 I3
QA PA
QB PB
QC PC
QD PD
Serial i/p for shift Rt (SIR)
serial i/p for shift left (SIL)

$P_A P_B P_C P_D$: parallel i/p's

$Q_A Q_B Q_C Q_D$: parallel o/p's

* The mode selection table & symbol are shown in fig below.

Select lines		Data selected	operation.
S_1	S_0		
0	0	I_0	hold
0	1	I_1	Shift right
1	0	I_2	shift left
1	1	I_3	parallel load

fig: Mode selection table.

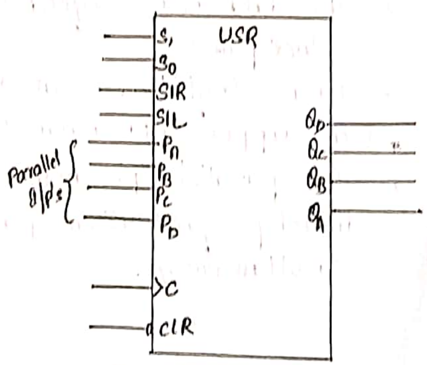
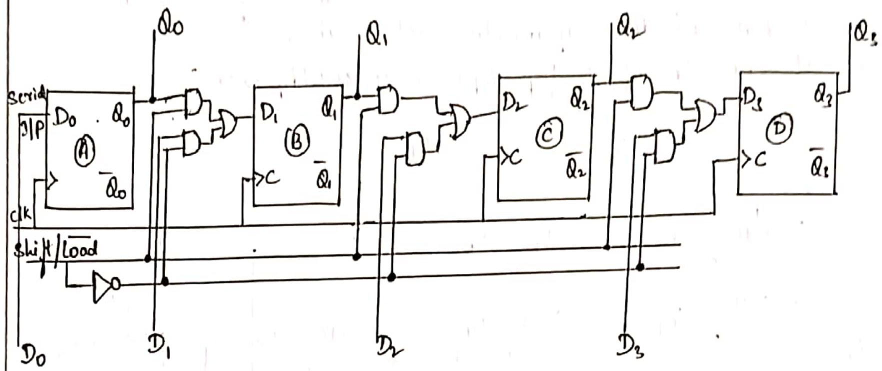


Fig: Symbol.

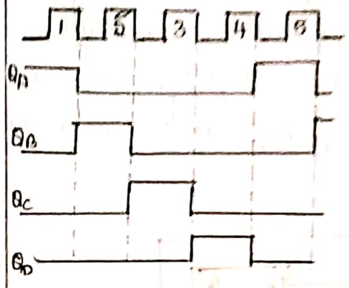
Parallel in Serial out [PISO] Shift Register.

In this type, the bits are entered in parallel i.e, simultaneously into their respective stages on parallel lines.

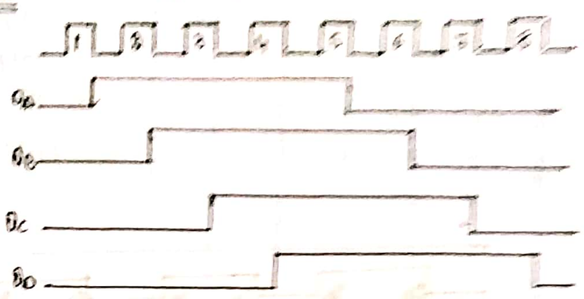


- * Fig above shows a parallel in Serial out unidirectional shift register.
- * There are 4 i/p lines D0, D1, D2, D3 for entering data in parallel into the register & parallel o/p's are Q0, Q1, Q2, Q3.
- * Shift/$\overline{Load}$ is the control i/p which allows shift or loading data operation of the register.
- * When shift/$\overline{load}$ is @ logic 1, all the upper AND gates are enabled, Serial i/p (D0) appears at D i/p of flip flop A.
- * The 'Q' o/p's of flip-flops A, B & C appear respectively at the 'D' i/p's of flip flops B, C, & D (next right flipflops).
- * The data gets shifted right on the application of clock pulses.
- * When shift/$\overline{load}$ is at logic '0', lower AND gates gets enabled & upper AND gates get disabled.
- * The parallel i/p's now appear at the respective 'D' i/p's thereby facilitating parallel i/p. (i.e, all 4 bits are stored simultaneously).

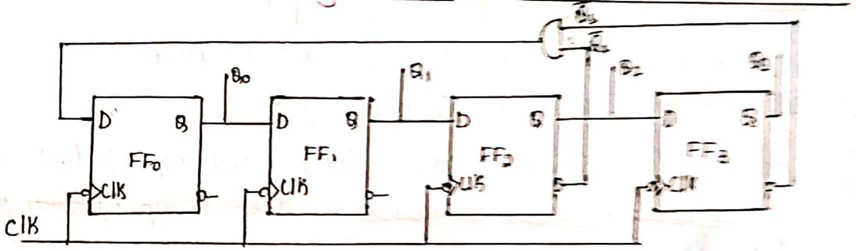
* Timing diagram of 4-bit Ring Counter



* Timing dig of 4-bit Johnson Counter



Mod-7 twisted Ring Counter [Ie mod-7 Johnson Counter].



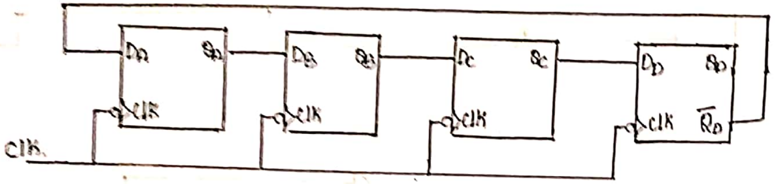
* Fig above shows a mod-7 twisted ring counter with a odd no of states where the 1111 state gets bypassed.

* Here the compliment o/p ($\bar{Q}$) of LSB flip-flop (FF3) & ~~the~~ 3rd FF2 are ANDed & the result is given as i/p to the FF1, i.e, MSB.

CLK	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	0	1	1	1
5	0	0	1	1
6	0	0	0	1

Johnson Counter or Twisted Ring Counter or Switch tail Counter

* In a Johnson Counter, the Q output of each stage of flip-flop is connected to the 'D' input of the next stage. But the Complement output of the last flip-flop is connected back to the D-input of the first flip-flop as shown in the fig below.



* Initially, the register (all flip-flops) are cleared. So, all the o/p's Q_A, Q_B, Q_C, Q_D are zero.

* The o/p of last stage, Q_D is zero. ∴ Complement o/p of last stage, $\bar{Q}_D$ is one. This is connected back to the D i/p of first flip/flop i.e. FF A. $[D_A]$. i.e, $D_A = 1$.

Hence for the next clock pulse o/p becomes 1000.

* Sequence of states are summarized in table below.

* A n-stage Johnson Counter has 2n states – [modulus 2n]

* In 4-bit Johnson Counter, total no of states = 8.

∴ It is a mod-8 Counter.

clK	Q_A	Q_B	Q_C	Q_D
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

** Johnson Counter is a digital Counter in which the 'Q' o/p of one flip flop is directly connected to i/p of next flip flop. But $\bar{Q}$ of LSB is connected back to the i/p of MSB.

Note:-

** Refer Class Notes for problems.

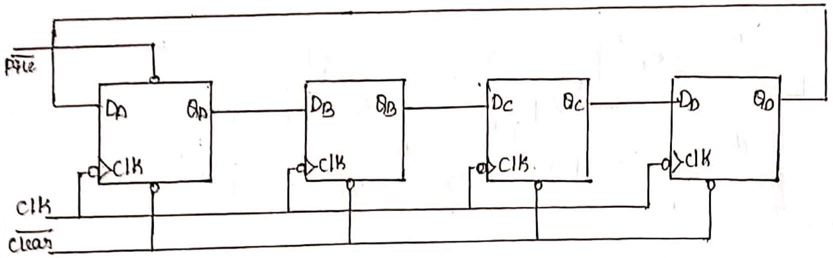
Counters Based on Shift Register:-

* There are Several applications in digital Systems where non-binary Counters are required.
* These Counters output a decoder like Sequence.
* These are used to identify Specific time instance.
* 2 types of non-binary Counters called ring Counters & Switch-tail Counters can be designed using shift registers.

Ring Counter.

* A Ring Counter is a circular shift register which is initialized Such that only one of its flip flop is at 1-State, while all others are in 0-State.

Thus, upon the (application) occurrence of each clock pulse, the Single 1 is shifted around the register. Fig below shows mod-4 ring Counter.



* In the above figure $\overline{CLR}$ followed by $\overline{Preset}$ makes the o/p of 1st Stage to logic '1' & remaining outputs are '0'. i.e., Q_A is One & Q_B, Q_C, Q_D are '0'

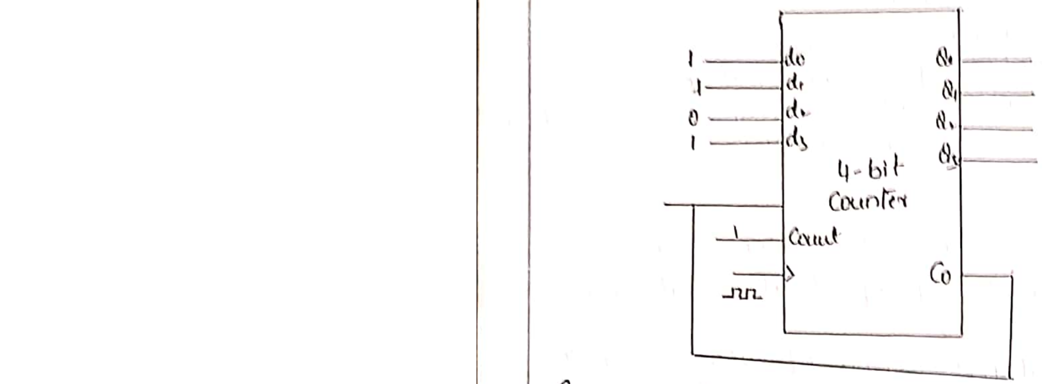
* No: of States = 4
∴ It is a mod-4 ring Counter. -

Counter Initialized to 1000, then Counting Sequence shows results as in T.T.

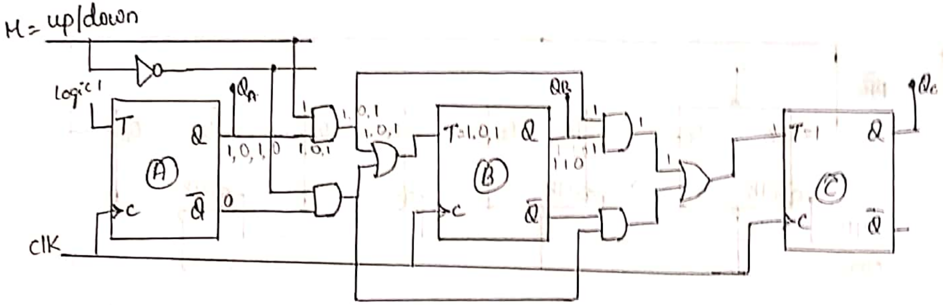
CLK	Q_A	Q_B	Q_C	Q_D
0	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	0	0	0

* A Mod-n ring Counter would require 'n' flip-flops.

Design a Counter to o/p Sequence from 1011 to 1111 & Repeat from 1011



Synchronous up/down Counter using T-flip-flop.



* 3 bit Synchronous up/down Counter is as shown in the fig.

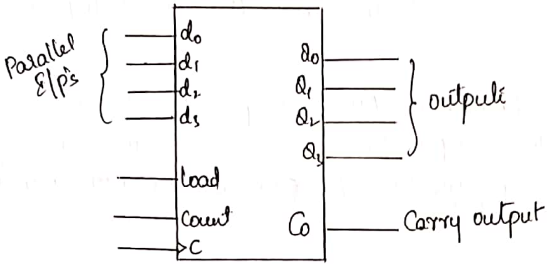
* The gating at each T input should be modified to accept all the previous ANDed Q outputs as well as the ANDed Q̄ o/p's controlled by the up/down control input.

* with the count enable at logic 1, up/down at logic 1 enables the upper AND gate for up Counting & up/down at logic 0 enables the lower AND gate for down Counting.

* For 4-bit up/down Counter 4 Flip-flops has to be used.

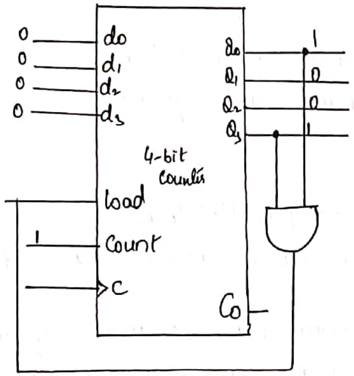
Ex:- a 0000 to 1111 counter can be made to count from 0000 to 1000, which becomes a mod-9 Counter.

- * Such 4-bit counters are available in a commercial package with load & count i/p's.
- * A logic 1 at the load input would load the count at d_0, d_1, d_2, d_3 into the counter.
- * A logic 1 at the count input would enable the up-counting.
- * CO → carry output, is used to cascade 4 bit counters to form higher bit counters.
  Co which goes high during the 1111 to 0000 transition.



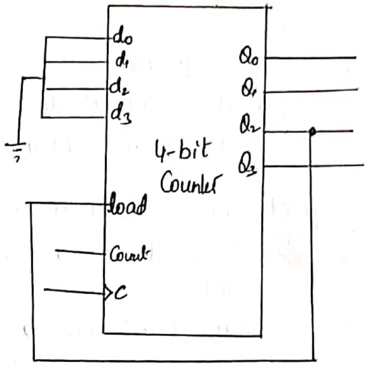
Mod-10 (or) Decade Counter :

mod-10 Counter Counts from 0-09 i.e, in binary 0000-1001.



mod-5 Counter :-

It Counts from 0 to 4
0000 to 0100



order flipflop is at 1.

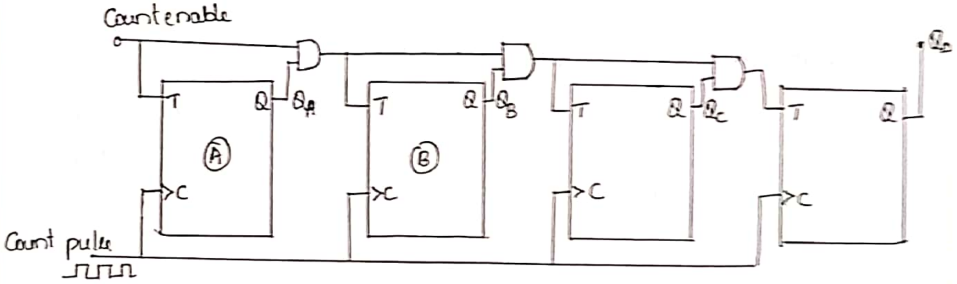
- * All lower order flip-flop outputs can be ANDed to enable the toggle of a given flip-flop.
- * When Count enable line is at logic 1, the AND gate outputs place a 1 at the T inputs when all the previous flip-flop outputs are at 1.

Note:- 1. write Truth table of upcounter & Timing diagram.

2. Circuit is same for down Counter, but Instead of Q, $\bar{Q}$ is given as I/p to AND gate.

Drawback:-

- * No of gates or stages increases the no of inputs to the AND gate.
- * Making use of the fact that ANDed outputs of all previous flip-flops are available at the output of each AND gate, the gating can be modified to keep the no of inputs to the AND gate are Constant as shown below.



Note:- In Synchronous Counters, Counter Speed is based on availability of next Count at the o/p terminal. Hence, Synchronous Counters are faster than Asynchronous

Synchronous counter (4 bit) with parallel load.

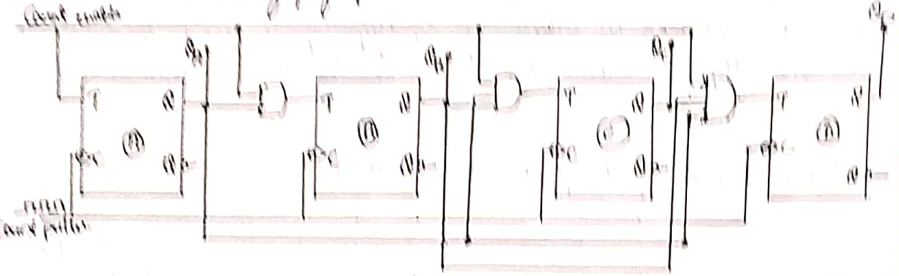
- * we know that a n-bit Counter can be used as a mod-2^n Counter.
- * we can configure a mod-m Counter where $m<2^n$, if we include a facility to parallel load an initial Count.

For n-stage binary ripple counter, the worst case settling time becomes "n × tpd"

Synchronous Binary Counter:

In this type of counter, all the flip flops are clocked simultaneously.

- * When a counter is clocked such that each flipflop in the counter is triggered at the same time, such counter is called as synchronous counter.
- * The effect of propagation delay is not greater since all flip flops in the counter change state at the same time
- * Delay of any is due to the delay of a single flip flop
- * Synchronous binary up counter constructed using negative (-ve) edge T flip flop is shown in fig.
  as it contains 4 flip flops counting sequence is from
  0000 to 1111 [0-15].
- * Count pulses are applied to the control input C of each clocked flip flop.



- * Observe that Q_0 toggles with every clock pulse (refer truth table of upcounter)
- * The other outputs toggle whenever all the lower order flip-flops are at 1.

Ex: at count 4, Q_2 toggles because both the lower order flip-flops Q_1 & Q_0 at 1 at count 3 (0011).

- * This can be applied to all counts in the table.
- * Thus the lowest significant flip-flop must toggle at every clock pulse & the rest must toggle whenever every lower

* The 3rd Count pulse causes only Q_A flip flop to change state & Count to become 0011.

* For the 4th Count pulse it toggles Q_A o/p to change state from 1 to 0, & $\bar{Q}_A$ becomes 1. This causes +ve edge to occurs at $\bar{Q}_A$ terminal. Thus Q_B is toggled, returning it to 0-state. In addition, when Q_B flip flop changes its state, the Q_C flip flop is toggeled by 0 to 1 appearing at $\bar{Q}_B$ o/p terminals. Now the state of the Counter is 0100. Like This Counting Sequence Continuous upto 1111.

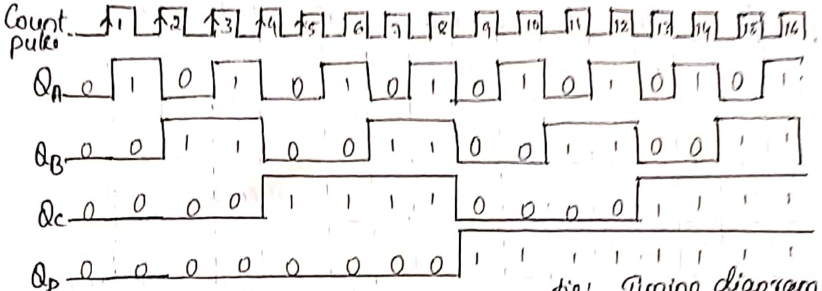
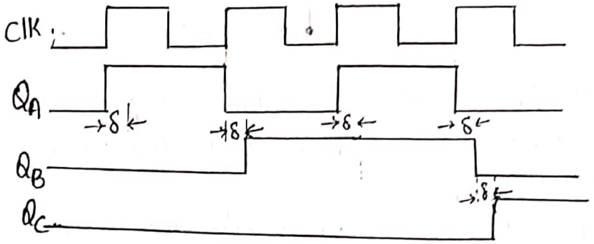


fig: Timing diagram

* Asynchronous (or) binary Counter is also known as ripple Counter since change in state of Q_i flip flop is used to toggle the Q_{i+1} flip flop. Thus, the effect of Count pulse must ripple through the Counter.

Draw back of Asynchronous Counter:



δ - tpd propagation delay time.

fig: Timing dig with delay (δ)

As shown in the fig above there is a propagation delay b/n i/p & o/p of flip flop, this rippling behaviour affects the overall time delay b/n the occurrence of Count pulse & when stability Count appears at o/p terminals. This delay is more when all flip flop's need to toggle for final Count.

* The clock inputs of flip flops B, C, D & are connected to Ā o/p of the previous flip flop.
* These flip flops change state on 1 to 0 transitions of Ā o/p which correspond to 0 to 1 transition of A output.

Count	Q_D	Q_C	Q_B	Q_A
0(0000)	0	0	0	0
1(1st)	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Repeat

fig: Counter table

* Since it is a 4-bit counter, no. of states are $2^4 = 16$
∴ It is a mod-16 counter counting sequence is from 0000 to 1111 (0 to 15)

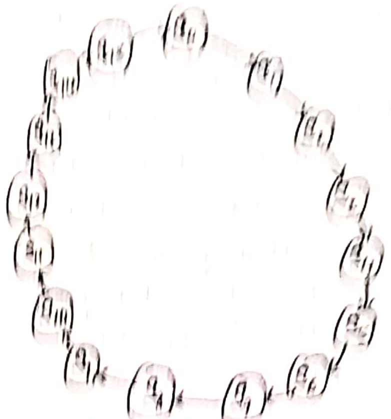


fig: State diagram

* Assume, initially the count is in state 0000 & count enable signal is logic-1 upon the occurrence of +ve edge of the first count pulse, Q_A flip flop changes to 01-state, and Q_A goes from 1 to 0, flip flop Q_B is not affected by the i/p pulse. The state of the counter is now 0001.
* When the positive edge of the 2nd count pulse arrives, the Q_A flip flop is again toggled. This time it returned to 0-state. Now, since Q_A o/p goes from 0 to 1, a +ve edge appears at the control input of the Q_B flip flop & causes it to toggle. The change in state Q_B flip flop cannot affect Q_C flip flop since -ve edge occurs at its control i/p.
Hence at the end of the pulse, state of counter is 0010.

Binary Ripple Counter (or) Asynchronous Counter:-

* Counters whose counting sequence corresponds to binary no are known as Binary Counters

* In this type of counter flip flops are not clocked simultaneously & flip flops are connected in such a way that o/p of first flip flop drives the clock for the next flip flop.
 ∴ This type of counter is known as asynchronous counter (or) Ripple counter.

* An 'n' bit binary consists of 'n' flip flops can count from '0' to (2^n-1).
 ∴ modulus of binary counter is 2^n. where n → no of flip flops.

* For up counter, counting sequence is from '0' to '2^n-1'.

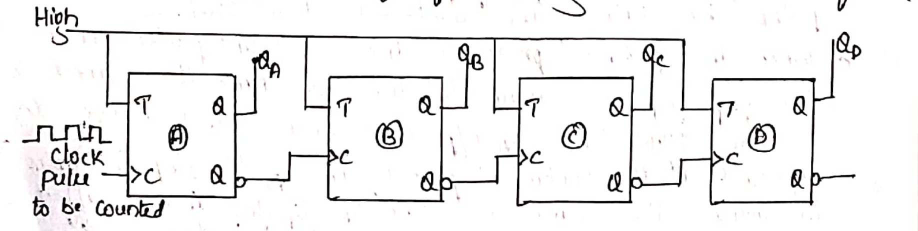
* For down counter, counting sequence is from (2^n-1) to 0. After reaching its max (or) min count counting sequence is repeated from its initial state.

* Ex:- A 3-bit binary up counter sequences from '000' (0) to '111' ($2^n = 2^3-1 = 8-1=7$) & repeats the sequence on reaching '111'.

Asynchronous 4-bit binary up counter (or) Ripple Counter [using +ve edge FF]

* In 4-bit counter, a cascade of 4 flip-flops can be used to configure a counter upto modulus 16.

* Fig shows a 4-bit binary up counter configured using positive edge Triggered 'T' flip-flops along with its count sequence.



* With the count enable or T inputs held at logic 1, the o/p of each flip flop toggles for every 0 to 1 transition of its clock i/p or at every positive edge of its clock input.

Kokila K.S
Asst. Professor
Dept of ECE
BGSIT.

Module-4

Simple flip-flop Applications

Introduction:

Flip-flop is nothing but a binary cell capable of storing one bit information, & can be connected to perform counting operations. Such a group of flip-flops is called Counters.

The collection of flip-flops can be used to store a word, is called Register.

A flip-flop can store 1-bit information. So an n-bit register has a group of n flip-flops & is capable of storing any binary information/number containing n-bits.

Register.

- * Registers are used to store data in a digital system.
- * A 4-bit register can store binary bits from 0000 - 1111. These are called Contents or States of a register. Thus a 4-bit register has 16 possible states.
- * A cascade of 4 flip-flops configured as a register can store one nibble of data.
- * Shift registers are capable of moving or shifting the data stored in their flip-flops in either directions.

Ex:- Consider a 4-bit shift register with data 0100 or decimal 4. A left shift results in 1000 or decimal 8 & a right shift results in 0010 or decimal 2.

- * Each left shift has a "multiplication by 2" effect & each right shift has a "division by 2" effect.
- * Shift register which can shift data in both directions are called bi-directional shift registers.
- * Those which can shift data in only one direction are called unidirectional shift registers.
- * Hence these are classified based on whether data is input or output in - serial or - parallel fashion.

* When information is transferred in parallel manner, all the bits in the information are handled simultaneously, as a single entity at a time.

No of i/p lines required are equal to no of bits in an information & requires One clock pulse.

* Information transfer in Serial manner involves bit by bit availability at a time [One bit at a time].

This type of transfer requires single i/p line & no of clock pulses = no of bits in an Information.

* Thus, there are 4 possible ways of data Transfer.

They are 1) Serial In Serial out. [SISO]

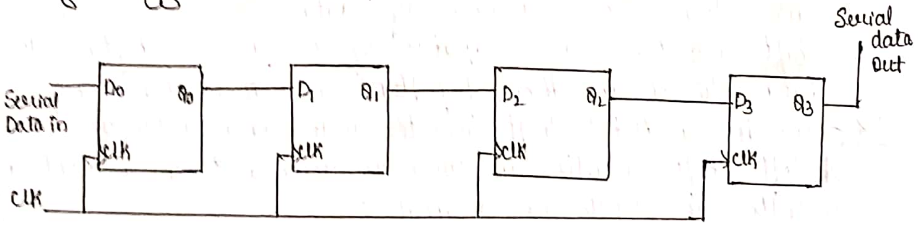
2) Serial In parallel out [SIPO]

3) Parallel In parallel out [PIPO]

4) parallel In Serial out [PISO]

Serial In Serial out [SISO]:-

* A 4-bit SISO unidirectional Shift register using positive edge Triggered D flip-flops is shown below.



* The D inputs of each flip-flop is connected to the 'Q' o/p of the previous stage to the left.

* The Control i/p's of all flip flops are Connected together to a Common synchronized clock. Thus, upon the occurance of the positive edge of clock signal, the Content of each flip flop is shifted to one position to the right.

* The data on the serial in line gets stored in flip-flop A & appears at Q_A.

* The content of left most flip flop after clock signal depends on serial data input line & content of right most flip-flop before a clock signal is lost.

Ex:- Serial data: 1101

initial data 0000

clk	i/p	Q_A	Q_B	Q_C	Q_D
↑	1 →	1	0	0	0
↑	0 →	0	1	0	0
↑	1 →	1	0	1	0
↑	1 →	1	1	0	<u>1</u>

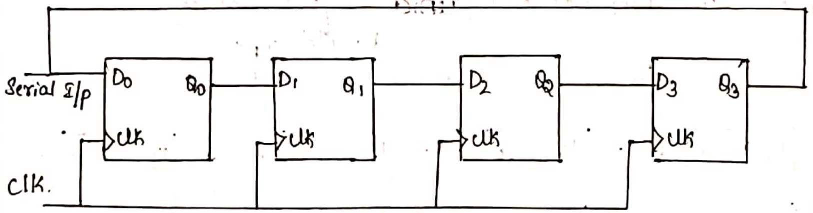
serial i/p

	Serial o/p
↑	× 1 1 <u>0</u>
↑	× × 1 <u>1</u>
↑	× × × <u>1</u>

Circular shift Register:-

In some application, the information within register must be preserved, by avoiding loss of information at the o/p of last flip flop.

For this, serial data o/p line is connected to serial data in line as shown below. This type of register is called circular shift register.

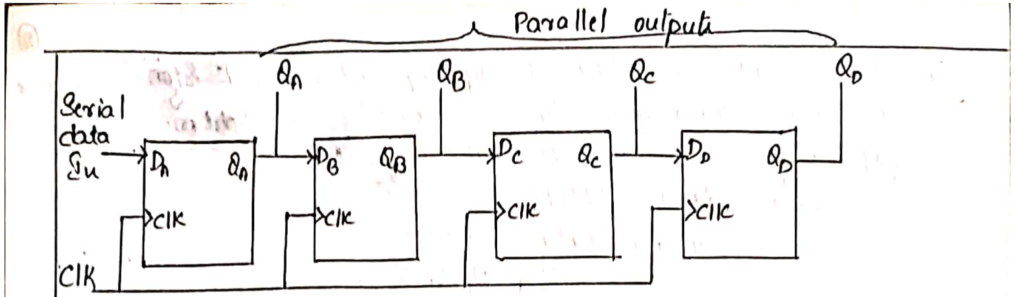


Ex:- If content is 1011, upon the occurrence of +ve edge of clock signal it becomes, 1101.

Serial In Parallel Out [SIPO] Shift Register:

* In this case, the data bits are entered into the register serially i.e., one after the other, The information is available as a single entity i.e., parallel out @ flip flop o/p terminals.

* This type of register provides serial to parallel conversion of information.

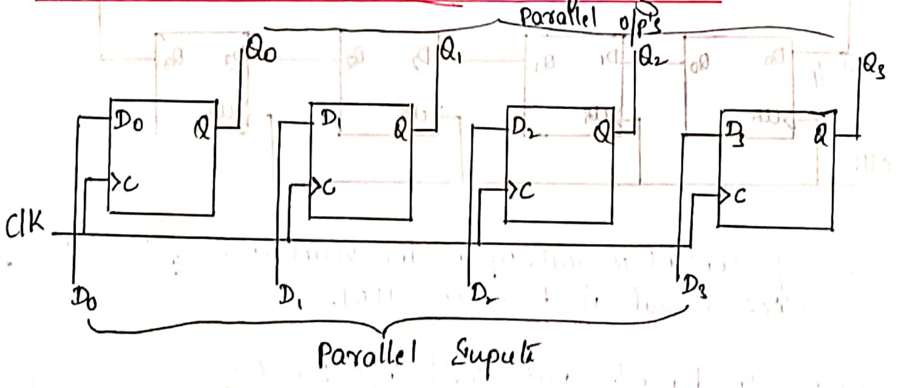


Clk	Serial data	Q_A	Q_B	Q_C	Q_D
↑	1	1	×	×	×
↑	0	0	1	×	×
↑	1	1	0	1	×
↑	1	1	1	0	1

Parallel out

observe that once the 4 bit data is shifted in after 4 clock pulses, the data stored in each flip-flop is available at the respective 'Q' o/p's.

Parallel In Parallel out [PIPO] Shift Registers:-



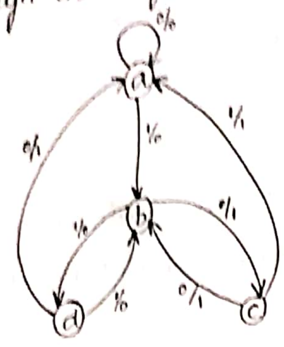
In this type, the bits are entered in parallel i.e, simultaneously into their respective stages on parallel lines & the bits appear on parallel outputs simultaneously upon the occurrence of clock pulse.

Ex:-

Clk	Parallel In D_0 D_1 D_2 D_3	parallel out Q_0 Q_1 Q_2 Q_3
↑	1 0 0 1	1 0 0 1

Construction of state diagram.

Q1) Obtain the transition table for the given state diagram and design the sequential Network using JK flip-flop.



i) State Table.

present state	Next state		output	
	x = 0	x = 1	x = 0	x = 1
F_1 F_2	F_1' F_2'	F_1' F_2'	Z	Z
a	a	b	0	0
b	c	d	1	0
c	b	a	1	1
d	a	b	1	0

ii) Transition Table

A B X	A' B' Z	J_A K_A	J_B K_B
0 0 0	0 0 0	0 x	0 x
0 0 1	0 1 0	0 x	1 x
0 1 0	1 0 1	1 x	x 1
0 1 1	1 1 0	1 x	x 0
1 0 0	0 1 1	x 1	1 x
1 0 1	0 0 1	x 1	0 x
1 1 0	0 0 1	x 1	x 1
1 1 0	0 1 0	x 1	x 0

iii) K-Map Simplification

A \ BX	00	01	11	10
0	0	0	1	1
1	x	x	x	x

$J_A = B$

A \ BX	00	01	11	10
0	x	x	x	x
1	1	1	1	1

$K_A = 1$

A \ BX	00	01	11	10
0	0	1	1	x
1	1	0	x	x

$J_B = \bar{A}X + A\bar{X}$

$= A \oplus X$

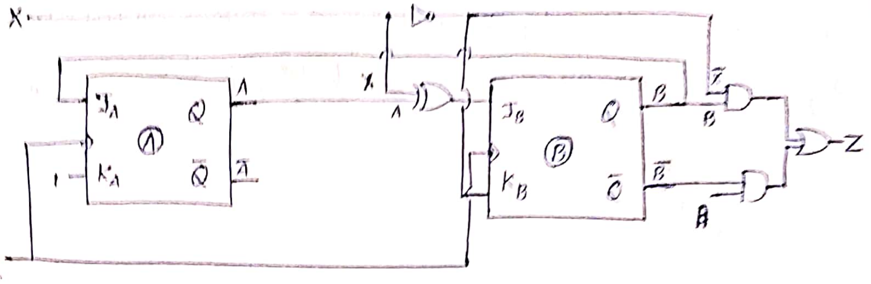
A \ BX	00	01	11	10
0	x	x	0	1
1	x	x	0	1

$K_B = \bar{X}$

A \ BX	00	01	11	10
0	0	0	0	1
1	1	1	0	1

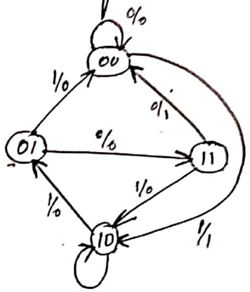
$Z = A\bar{B} + B\bar{X}$

iv) diagram.



05.) A Sequential has 1 input and 1 output. The state diagram is shown in figure. Design the Sequential circuit with

a) D-flip-flop
b) T-flip flop
c) SR-flip flop
d) JK-flip flop.



i) state table.

present state	Next state		output	
	x=0	x=1	x=0	x=1
A B	A^+ B^+	A^+ B^+	Y	Y
00	0 0	1 0	0	1
01	1 1	0 0	0	0
1 0	1 0	0 1	1	0
1 1	0 0	1 0	1	0

ii) Transition Table for D FF

A B X	A^+ B^+ Y	D_A^+ D_B^+
0 0 0	0 0 0	0 0
0 0 1	1 0 1	1 0
0 1 0	1 1 0	1 1
0 1 1	0 0 0	0 0
1 0 0	1 0 1	1 0
1 0 1	0 1 0	0 1
1 1 0	0 0 1	0 0
1 1 1	1 0 0	1 0

iii) K-Map

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	0	1	0	1
A	1	0	1	0

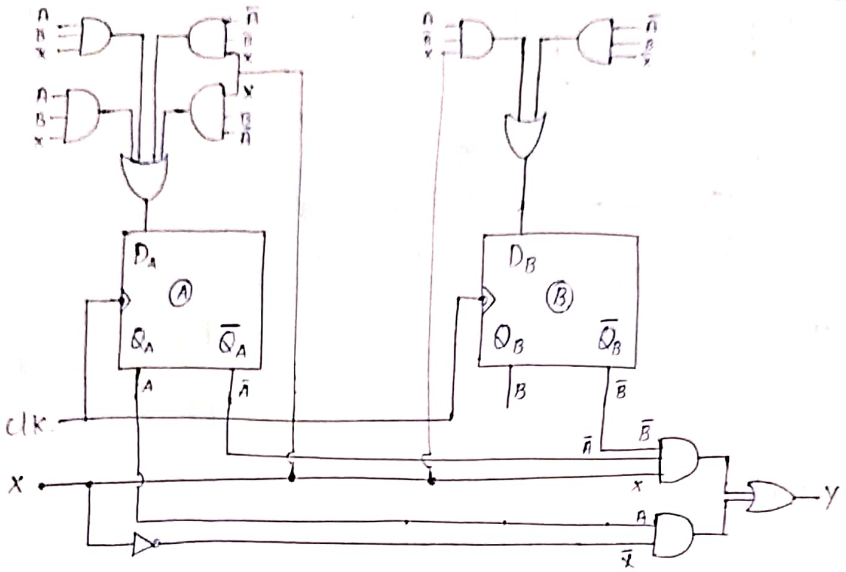
$D_A^+ = A\bar{B}\bar{X} + \bar{A}\bar{B}X + ABX + \bar{A}B\bar{X}$

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	0	0	0	1
A	0	1	0	0

$D_B^+ = A\bar{B}X + \bar{A}B\bar{X}$

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	0	1	0	0
A	1	0	0	1

$Y = \bar{A}\bar{B}X + A\bar{X}$



ii b) Transition Table for T flip flop.

A	B	X	A^+	B^+	Y	T_A^+	T_B^+
0	0	0	0	0	0	0	0
0	0	1	1	0	1	1	0
0	1	0	1	1	0	1	0
0	1	1	0	0	0	0	1
1	0	0	1	0	1	0	0
1	0	1	0	1	0	1	1
1	1	0	0	0	1	1	1
1	1	1	1	0	0	0	1

iii b) K-Map for T

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	0	1	0	1
A	0	1	0	1

$T_A = \bar{B}X + B\bar{X}$

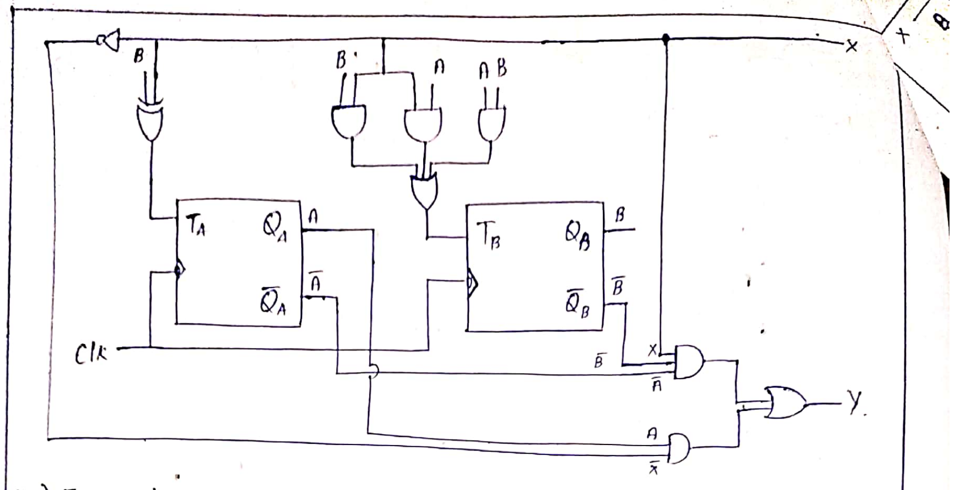
$= B \oplus X$

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	0	0	1	0
A	0	1	1	1

$T_B = BX + AX + AB$

and from output Y result will be same for all flip flop.

$\therefore Y = A\bar{B}X + A\bar{X}$



ii c) Transition Table.

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	1	1	0	0	X
0	1	0	1	1	0	1	0	X	0
0	1	1	0	0	0	0	X	0	1
1	0	0	1	0	1	X	0	0	X
1	0	1	0	1	0	0	1	1	0
1	1	0	0	0	1	0	1	0	1
1	1	1	1	0	0	X	0	0	1

iii) c) K-Map

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	0	1	0	1
A	X	0	X	0

$S_A = \bar{A}\bar{B}X + \bar{A}B\bar{X}$
$= \bar{A}(B \oplus X)$

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	X	0	X	0
A	0	1	0	1

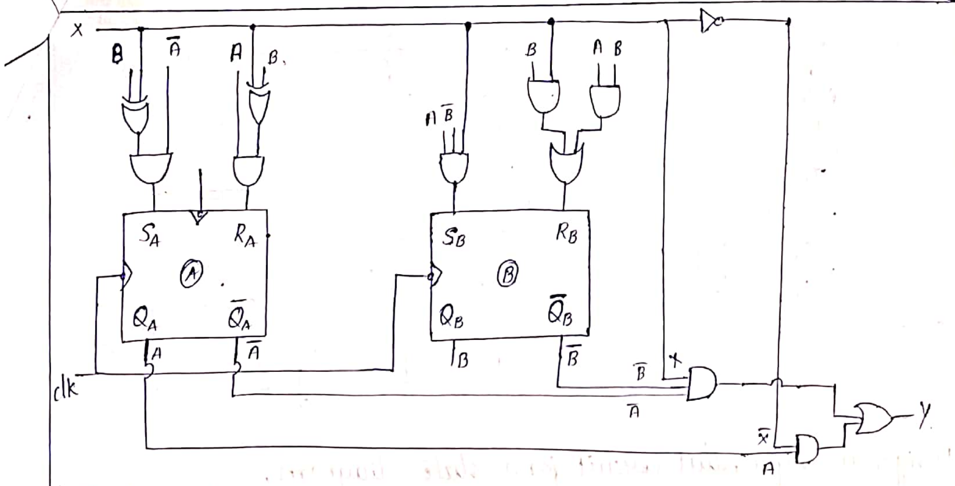
$R_A = AX\bar{B} + AB\bar{X}$
$= A(B \oplus X)$

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	0	0	0	X
A	0	1	0	0

$S_B = A\bar{B}X$

	$\bar{B}\bar{X}$	$\bar{B}X$	BX	$B\bar{X}$
$\bar{A}$	X	X	1	0
A	X	0	1	1

$R_B = BX + AB$



i)d. Transition Table for J and K flip flop.

A	B	X	A^+	B^+	Y	J_A	K_A	J_B	K_B
0	0	0	0	0	0	0	X	0	X
0	0	1	1	0	1	1	X	0	X
0	1	0	1	1	0	1	X	X	0
0	1	1	0	0	0	0	X	X	1
1	0	0	1	0	1	X	0	0	X
1	0	1	0	1	0	X	1	1	X
1	1	0	0	0	1	X	1	X	1
1	1	1	1	0	0	X	0	X	1

ii) d. K-Map for J_A

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

$J_A = \bar{B}X + B\bar{X}$

$= B \oplus X$

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

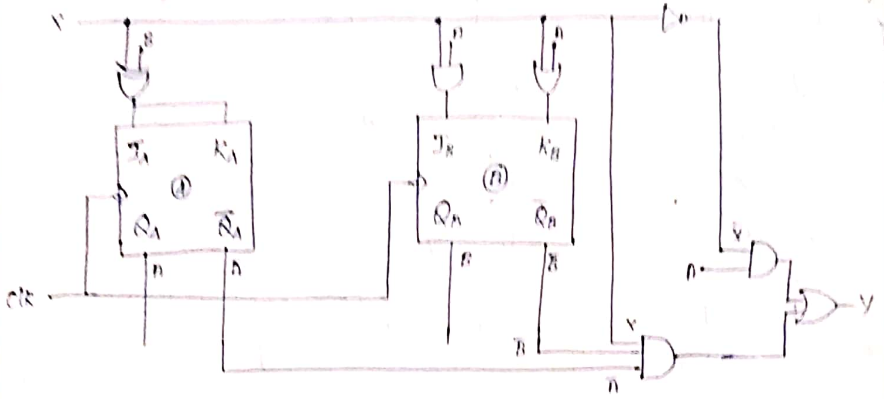
$K_A = B \oplus X$

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

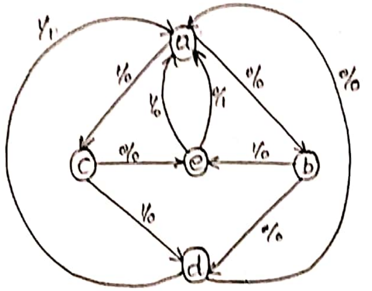
$J_B = AX$

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

$K_B = A + X$



03) Design a sequential circuit for a state diagram.



i) Transition Table.

present state	Next state		output.	
	0	1	0	1
0 0 0	0 0 1	0 1 0	0	0
0 0 1	0 1 1	1 0 0	0	0
0 1 0	1 0 0	0 1 1	0	0
0 1 1	0 0 0	0 0 0	0	1
1 0 0	0 0 0	0 0 0	1	0
1 0 1	x x x	x x x	x	x
1 1 0	x x x	x x x	x	x
1 1 1	x x x	x x x	x	x

ii) Excitation table.

A	B	C	X	A'	B'	C'	Y	D_A	D_B	D_C
0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	1	0	0	0	1	0
0	0	1	0	0	1	1	0	0	1	1
0	0	1	1	1	0	0	0	1	0	0
0	1	0	0	1	0	0	0	1	0	0
0	1	0	1	0	1	1	0	0	1	1
0	1	1	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	1	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	0	1	0	0	0	0	0	0	0
1	0	1	0	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x
1	1	1	1	x	x	x	x	x	x	x

iii) K-Map

AB \ CX	00	01	11	10
00	0	0	1	0
01	1	0	0	0
11	x	x	x	x
10	0	0	x	x

$D_A = B\bar{C}\bar{X} + \bar{B}CX$

AB \ CX	00	01	11	10
00	0	1	0	1
01	0	1	0	0
11	x	x	x	x
10	0	0	x	x

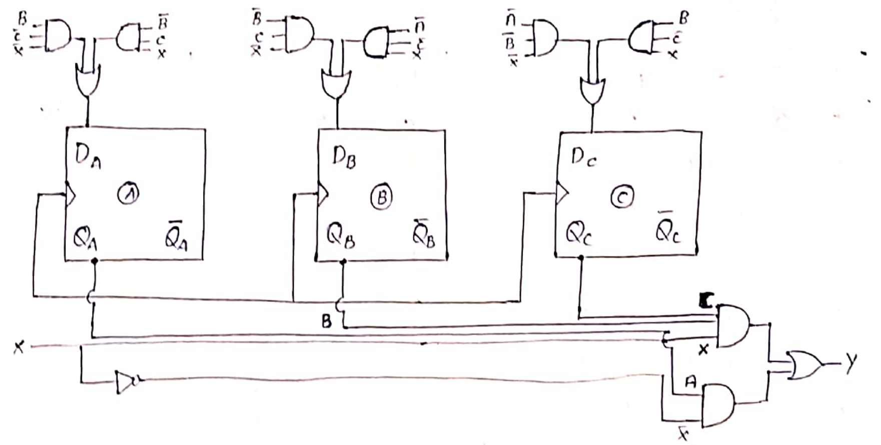
$D_B = \bar{A}\bar{C}X + \bar{B}C\bar{X}$

AB \ CX	00	01	11	10
00	1	0	0	1
01	0	1	0	0
11	x	x	x	x
10	0	0	x	x

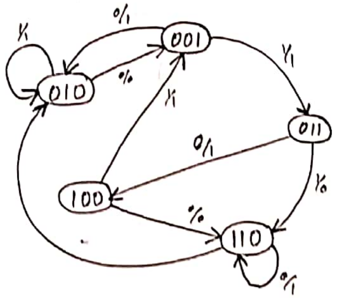
$D_C = \bar{A}\bar{B}\bar{X} + B\bar{C}X$

AB \ CX	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	x	x	x	x
10	1	0	x	x

$Y = A\bar{X} + BCX$



04.) Design a sequential circuit for diagram shown in figure using JK flip-flop.



State table.

present state	Next state		output	
	x=0	x=1	x=0	x=1
	a' b' c'	a' b' c'	z	z
0 0 1	0 1 0	0 1 1	1	1
0 1 0	0 0 1	0 1 0	0	1
0 1 1	1 0 0	1 1 0	1	0
1 0 0	1 1 0	0 0 1	0	1
1 1 0	1 1 0	0 1 0	1	0

ii) Transition Table

A	B	C	X	A^+	B^+	C^+	Y	J_A	K_A	J_B	K_B	J_C	K_C
0	0	1	0	0	1	0	1	0	X	1	X	X	1
0	0	1	1	0	1	1	1	0	X	1	X	X	0
0	1	0	0	0	0	1	0	0	X	X	1	1	X
0	1	0	1	0	1	0	1	0	X	X	0	0	X
0	1	1	0	1	0	0	1	1	X	X	1	X	1
0	1	1	1	1	1	0	0	1	X	X	0	X	1
1	0	0	0	1	1	0	0	X	0	1	X	0	X
1	0	0	1	0	0	1	1	X	1	0	X	1	X
1	1	0	0	1	1	0	1	X	0	X	0	0	X
1	1	0	1	0	1	0	0	X	1	X	0	0	X
0	0	0	0	X	X	X	X	X	X	X	X	X	X
0	0	0	1	X	X	X	X	X	X	X	X	X	X
1	0	1	0	X	X	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X	X	X

iii) K-Map

AB \ CX	00	01	11	10
00	X	X	0	0
01	0	0	1	1
11	X	X	X	X
10	X	X	X	X

$J_A = BC$

AB \ CX	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	0	1	X	X
10	0	1	X	X

$K_A = X$

AB \ CX	00	01	11	10
00	X	X	1	1
01	0	1	0	1
11	1	0	X	X
10	0	1	X	X

$Y = C\bar{X} + AB\bar{X} + \bar{A}\bar{C}X + \bar{B}X$

	00	01	11	10
00	X	X	1	1
01	X	X	X	X
11	X	X	X	X
10	1	0	X	X

$J_B = C + \bar{X}$

AB \ CX	00	01	11	10
00	X	X	X	X
01	1	0	0	1
11	0	0	X	X
10	X	X	X	X

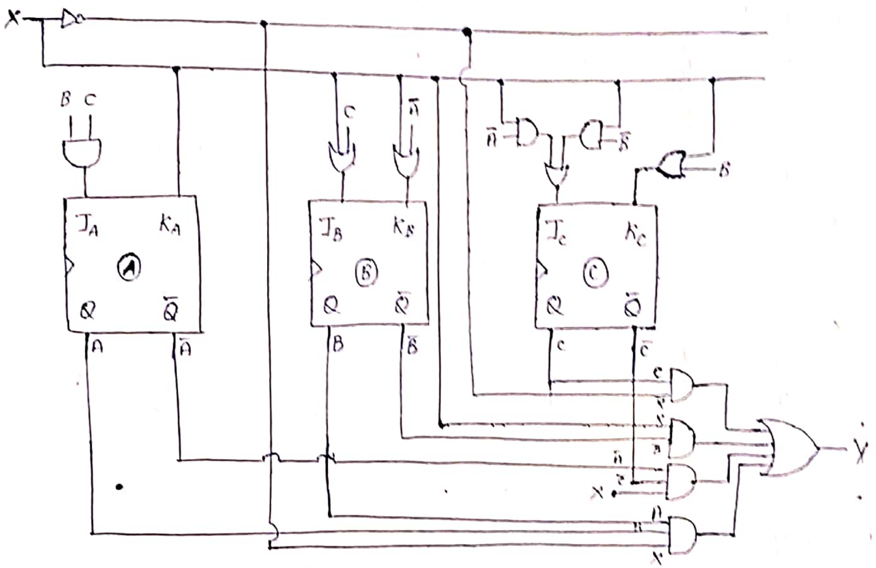
$K_B = \bar{A}\bar{X}$

AB \ CX	00	01	11	10
00	X	X	X	X
01	1	0	X	X
11	0	0	X	X
10	0	1	X	X

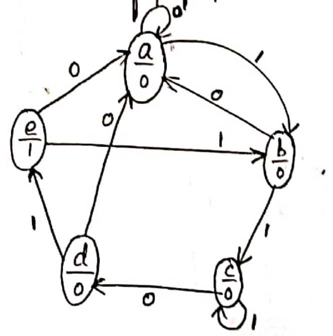
$J_C = \bar{A}\bar{X} + \bar{B}X$

AB \ CX	00	01	11	10
00	X	X	0	1
01	X	X	1	1
11	X	X	X	X
10	X	X	X	X

$K_C = \bar{X} + B$



05) Design a sequential circuit for diagram shown in figure using JK flip-flop.



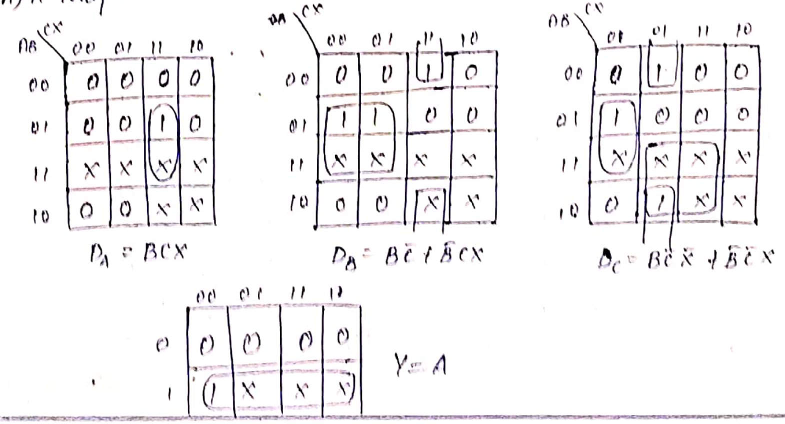
i) State Table

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

ii) Transition state Table.

present state	Next state		Output
A B C	X=0	X=1	Y
	$A^+ B^+ C^+$	$A^+ B^+ C^+$	
0 0 0	0 0 0	0 0 1	0
0 0 1	0 0 0	0 1 0	0
0 1 0	0 1 1	0 1 0	0
0 1 1	0 0 0	1 0 0	0
1 0 0	0 0 0	0 0 1	1

ii) K-Map.



AB \ CX	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	X	X	X	X
10	0	0	X	X

$D_A = BCX$

AB \ CX	00	01	11	10
00	0	0	1	0
01	1	1	0	0
11	X	X	X	X
10	0	0	X	X

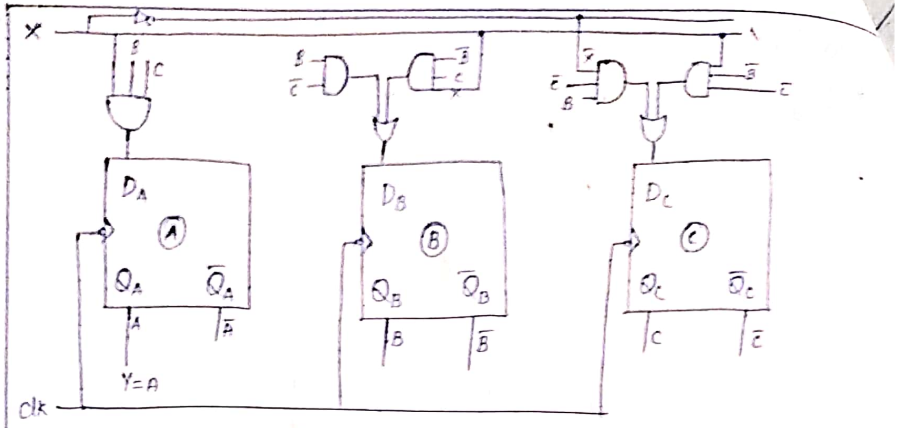
$D_B = B\bar{C} + \bar{B}CX$

AB \ CX	01	01	11	10
00	0	1	0	0
01	1	0	0	0
11	X	X	X	X
10	0	1	X	X

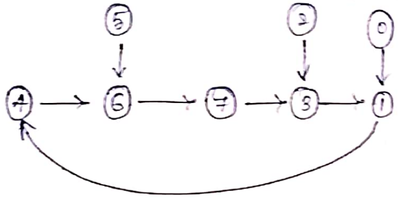
$D_C = B\bar{C}\bar{X} + \bar{B}\bar{C}X$

	00	01	11	10
0	0	0	0	0
1	1	X	X	X

$Y = A$



06) Design a synchronous counter for 4 → 6 → 7 → 3 → 1 → 4 avoid lockout condition use JK type design.



0, 2, 5 are lock out we should avoid using dont care

Here states 5, 2 and 0 are forced to go into 6, 3 & 1 states respectively to avoid lockout condition.

i) Transition state table

present state			Next state			J_A	K_A	J_B	K_B	J_C	K_C
A	B	C	A^+	B^+	C^+						
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	1	0	0	1	X	0	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	0	0	1	0	X	X	1	X	0
1	0	0	1	1	0	X	0	1	X	0	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	1	1	1	X	0	0	X	1	X
1	1	1	0	1	1	X	1	0	X	X	0

ii) K-Map

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

$J_A = \overline{B}C$

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

$K_A = BC$

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

$J_B = A$

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

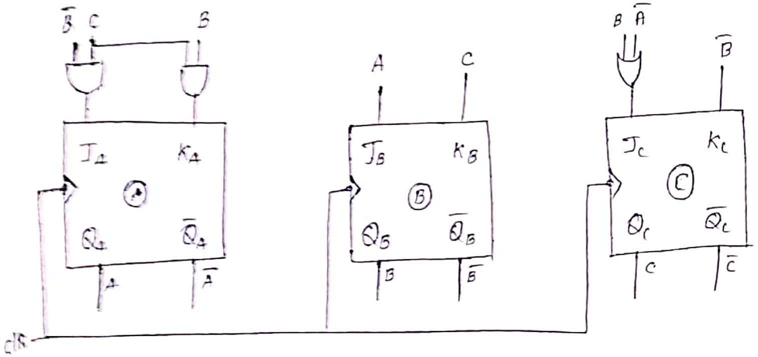
$K_B = C$

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

$J_c = B + \overline{A}$

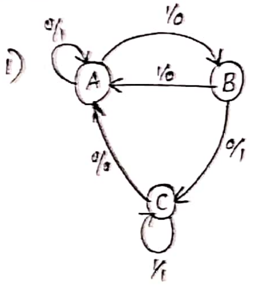
A \ BC	00	01	11	10
0	X	1	0	X
1	X	1	0	X

$K_c = \overline{B}$



Q7) construct the state table for state diagram.

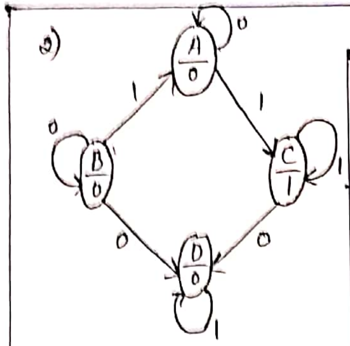
i)



Ans:

present state	Next state		output	
	x=0	x=1	x=0	x=1
Q_A Q_B	Q_A^+ Q_B^+	Q_A^+ Q_B^+	Z	Z
A	A	B	1	0
B	C	A	1	0
C	A	C	0	1

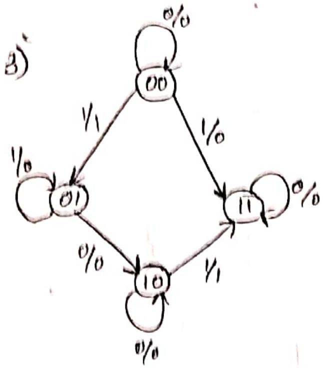
2)



ANS

present state	Next state		output
	x=0	x=1	Z
Q_A Q_B	Q_A^+ Q_B^+	Q_A^+ Q_B^+	
A	A	C	0
B	B	A	0
C	D	C	1
D	B	D	0

3)



present state	Next state		output	
	x=0	x=1	x=0	x=1
Q_A Q_B	$Q_A^{'}$ $Q_B^{'}$	Q_A^+ Q_B^+	Z	Z
A	A	B	0	1
B	C	B	0	0
C	C	D	0	1
D	D	A	0	0

Design of Synchronous counter:

01. Determine the number of flip flops needed. If n represents number of flip flops

$$2^n \geq \text{number of states in the Counter.}$$

02. choose the type of flip-flops to be used.
03. using excitation table for selected flip-flop determine the Excitation table for the Counter.
04. use K-Map or any other simplification method to derive the flip-flop input functions
05. Draw the logic diagram.

Information required for designing synchronous circuit:

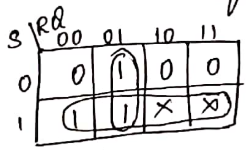
Truth table for SR.

S R	Q	Q_{n+1}
0 0	0	0
	1	1 (No change)
0 1	0	0
	1	0 (Reset)
1 0	0	1
	1	1 (Set)
1 1	0	1*
	1	1*

Excitation Table

Q	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Characteristic Eqⁿ for SR.



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

$$Q^+ = S + \bar{R}Q$$

Truth table for JK.

J	K	Q	Q_{n+1}
0	0	0	0
		1	1
0	1	0	0
		1	0 (Reset)
1	0	0	1
		1	1 (Set)
1	1	0	1 (Toggle state)
		1	0

Excitation Table

Q	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Charecteristic Eq:-

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

$\langle Q^+ = J\bar{Q} + \bar{K}Q \rangle$

Excitation Table for T

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

T \ Q	0	1
0	0	1
1	1	0

$\langle Q^+ = \bar{T}Q + T\bar{Q} \rangle$

Excitation Table for D

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

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

D \ Q	0	1
0	0	0
1	1	1

$\langle Q^+ = D \rangle$

01. Design of a Synchronous MOD-6 Counter using clocked JK Flip-flop.

(1) mod-6 ⟹ 000 to 101

we require 3 flip flop.

$2^n \geq m$

$2^3 = 8 \geq 6$

Hence condition statisfies

now we have To design.

Step 2 : JK flip-flop

Step 3 :

present state			Next state			flip-flop Inputs		
Q_A	Q_B	Q_c	Q_{A+1}	Q_{B+1}	Q_{C+1}	$J_A K_A$	$J_B K_B$	$J_c K_c$
0	0	0	0	0	1	0 X	0 X	1 X
0	0	1	0	1	0	0 X	1 X	X 1
0	1	0	0	1	1	0 X	X 0	1 X
0	1	1	1	0	0	1 X	X 1	X 1
1	0	0	1	0	1	X 0	0 X	1 X
1	0	1	0	0	0	X 1	0 X	X 1
1	1	0	X	X	X	X X	X X	X X
1	1	1	X	X	X	X X	X X	X X

Step 4 : K-map simplification for flip-flop inputs.

For J_A

Q_A \ Q_BQ_c	00	01	11	10
0	0	0	1	0
1	X	X	X	X

$(J_A = Q_BQ_C)$

For K_A

Q_A \ Q_BQ_c	00	01	11	10
0	X	X	X	X
1	0	1	X	X

$(K_A = Q_c)$

For J_B

Q_A \ Q_BQ_c	00	01	11	10
0	0	1	X	X
1	0	0	X	X

$(J_B = \bar{Q}_A Q_c)$

For K_B

Q_A \ Q_BQ_c	00	01	11	10
0	X	X	1	0
1	X	X	X	X

$(K_B = Q_c)$

For J_c

Q_A \ Q_BQ_c	00	01	11	10
0	1	X	X	1
1	1	X	X	X

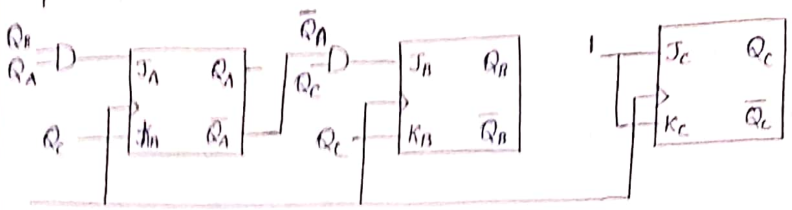
$(J_c = 1)$

For K_c

Q_A \ Q_BQ_c	00	01	11	10
0	X	1	1	X
1	X	1	X	X

$(K_c = 1)$

Step 06 :



02. Design a Synchronous mod-6 counter using SR flip-flop.

Step 01 : Mod 6 (0-5) 000-101

$2^n \geq N$

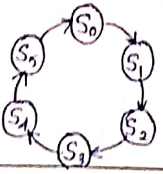
$2^3 \geq 6$

$8 \geq 6$

Step 02 : SR flip flop.

Step 03 : Excitation table.

Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	S_AR_A	S_BR_B	S_CR_C
0	0	0	0	0	1	0 X	0 X	1 0
0	0	1	0	1	0	0 X	1 0	0 1
0	1	0	0	1	1	0 X	X 0	1 0
0	1	1	1	0	0	1 0	0 1	0 1
1	0	0	1	0	1	X 0	0 X	1 0
1	0	1	0	0	0	0 1	0 X	0 1
1	1	0	X	X	X	X X	X X	X X
1	1	1	X	X	X	X X	X X	X X



For S_A

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	0	1	0
1	X	0	X	X

$S_A = Q_BQ_C$

For S_B

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	1	0	X
1	0	0	X	X

$S_B = \bar{Q}_A\bar{Q}_BQ_C$

For S_C

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	1	0	0	1
1	1	0	X	X

$S_C = \bar{Q}_C$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	X	X	0	X
1	0	1	X	X

$R_A = \bar{Q}_BQ_C$

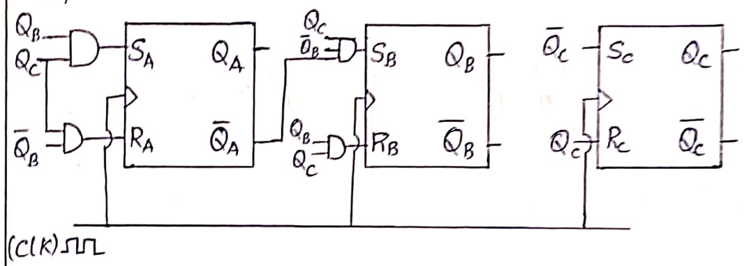
$Q_A \backslash Q_BQ_C$	00	01	11	10
0	X	0	1	0
1	X	X	X	X

$R_B = Q_BQ_C$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	1	1	0
1	0	1	X	X

$R_C = Q_C$

step 05:



03.) Design a synchronous mod-6 counter using T flip flop.

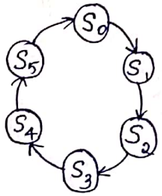
Step 01 :· (0−5)

$2^3 \geq N$

$8 \geq 6$

Step 02 : T-flip-flop

Step 03 : Excitation Table.



Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	T_A	T_B	T_C
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	0	0	0	1	0	1
1	1	0	x	x	x	x	x	x
1	1	1	x	x	x	x	x	x

Step 04 :

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	0	1	0
1	0	1	X	x

$T_A = Q_AQ_C + Q_BQ_C$

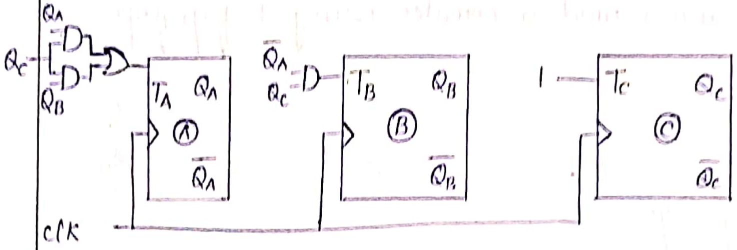
$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	1	1	0
1	0	0	X	X

$T_B = \bar{Q}_AQ_C$

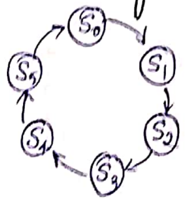
$Q_A \backslash Q_BQ_C$	00	01	11	10
0	1	1	1	1
1	1	1	X	X

$T_C = 1$

Step 05 :



04) Design a synchronous mod-6 counter using D flip-flop.



Step 01 : (0-5)

$2^3 \geq N$

$8 \geq 6.$

Step 02 : D flip flop

Step03:

present state			Nex state			D-flip-flop		
Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	D_A	D_B	D_C
0	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	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	0	0	0	0	0	0
1	1	0	X	X	X	X	X	X
1	1	1	X	X	X	X	X	X

Step04:

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	0	1	0
1	1	0	X	X

$D_A = Q_A\overline{Q_C}$

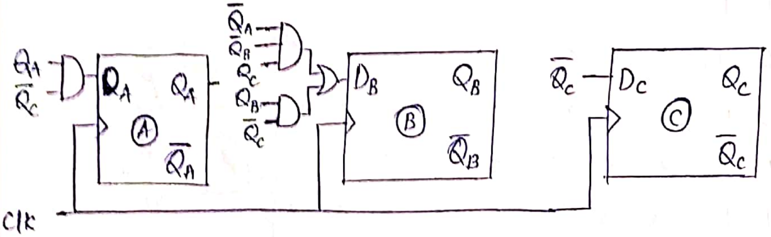
$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	1	0	1
1	0	0	X	X

$D_B = \overline{Q_A}\,\overline{Q_B}Q_C + Q_B\overline{Q_C}$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	1	0	0	1
1	1	0	X	X

$D_C = \overline{Q_C}$

Step05:



05. Design a synchronous decade counter using D-flipflop

- Mod 10 (∵ decade)

  ∴ 0 - 9

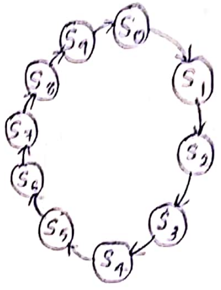
  $2^4 \geq N$

  $16 \geq 9$

- D flip-flop

- Excitation table.

Q_D	Q_C	Q_B	Q_A	Q_{D+1}	Q_{C+1}	Q_{B+1}	Q_{A+1}	D_D	D_C	D_B	D_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	0
0	0	1	0	0	0	1	1	0	0	1	1
0	0	1	1	0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	1	0	1	0	1
0	1	0	1	0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	1	0	1	1	1
0	1	1	1	1	0	0	0	1	0	0	0
1	0	0	0	1	0	0	1	1	0	0	1
1	0	0	1	X	X	X	X	X	X	X	X
1	0	1	0	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X



$Q_DQ_C \backslash Q_BQ_A$	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	X	X	X	X
10	1	0	X	X

$D_D = Q_D\overline{Q_A} + Q_AQ_BQ_C$

$Q_DQ_C \backslash Q_BQ_A$	00	01	11	10
00	0	0	1	0
01	1	1	0	1
11	X	X	X	X
10	0	0	X	X

$D_C = \overline{Q_B}Q_C + Q_C\overline{Q_A} + \overline{Q_C}Q_AQ_B$

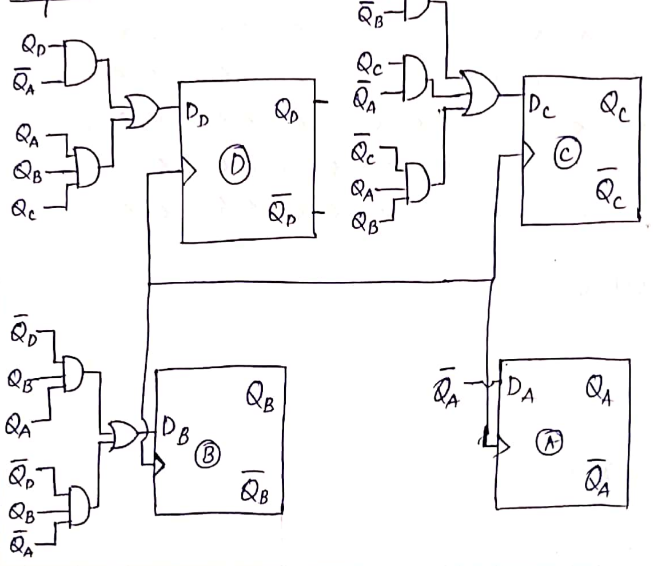
$Q_DQ_C \backslash Q_BQ_A$	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	X	X	X	X
10	0	0	X	X

$D_B = \overline{Q_D}Q_BQ_A + \overline{Q_D}Q_B\overline{Q_A}$

$Q_DQ_C \backslash Q_BQ_A$	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	X	X	X	X
10	1	0	X	X

$D_A = \overline{Q_A}$

Step05:



06.) Design a Synchronous-to count from 0000 – 1001 using JK flip-flop.

0000 – 1001

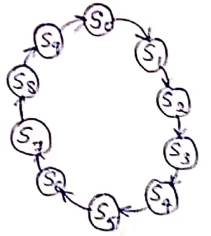
0 – 9

$2^4 \geq 9$

$16 \geq 9$

JK-flip-flop

Q_A Q_B Q_C Q_D	Q_A^* Q_B^* Q_C^* Q_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
1 0 1 0	X X X X	X X	X X	X X	X X
1 0 1 1	X X X X	X X	X X	X X	X X
1 1 0 0	X X X X	X X	X X	X X	X X
1 1 0 1	X X X X	X X	X X	X X	X X
1 1 1 0	X X X X	X X	X X	X X	X X
1 1 1 1	X X X X	X X	X X	X X	X X



	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	X	X	X	X
10	X	X	X	X

$J_A = Q_B Q_C Q_D$

	00	01	11	10
00	0	0	1	0
01	X	X	X	X
11	X	X	X	X
10	0	0	X	X

$J_B = Q_C Q_D$

	00	01	11	10
00	0	1	X	X
01	0	1	X	X
11	X	X	X	X
10	0	0	X	X

$J_C = \bar{Q}_A Q_D$

	00	01	11	10
00	1	X	X	1
01	1	X	X	1
11	X	X	X	X
10	1	X	X	X

$J_D = \bar{Q}_D$

	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	X	X	X	X
10	0	1	X	X

$K_A = Q_D$

	00	01	11	10
00	X	X	X	X
01	0	0	1	0
11	X	X	X	X
10	X	X	X	X

$K_B = Q_C Q_D$

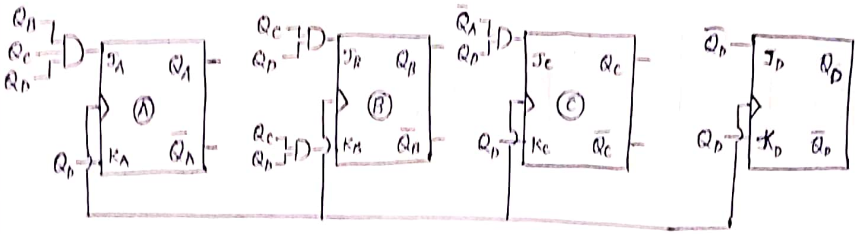
	00	01	11	10
00	X	X	1	0
01	X	X	1	0
11	X	X	X	X
10	X	X	X	X

$K_C = Q_D$

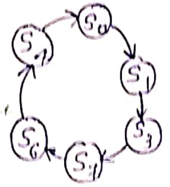
	00	01	11	10
00	X	1	1	X
01	X	1	1	X
11	X	X	X	X
10	X	1	X	X

$K_D = Q_D$

Step 05:



07) Design a counter with the Sequence 0,1,3,7,6,4,0



- n = 3.
- $2^n \geq N$
- $8 \geq 6$
- JK flip flop
- Excitation Table.

Q_A	Q_B	Q_C	Q_A^*	Q_B^*	Q_C^*	$J_A K_A$	$J_B K_B$	$J_C K_C$
0	0	0	0	0	1	0 X	0 X	1 X
0	0	1	0	1	1	0 X	1 X	X 0
0	1	0	X	X	X	X X	X X	X X
0	1	1	1	1	1	1 X	X 0	X 0
1	0	0	0	0	0	X 1	0 X	0 X
1	0	1	X	X	X	X X	X X	X X
1	1	0	1	0	0	X 0	X 1	0 X
1	1	1	1	1	0	X 0	X 0	X 1

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	0	1	X
1	X	X	X	X

$J_A = Q_B$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	0	1	X	X
1	0	X	X	X

$J_B = Q_C$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	1	X	X	X
1	0	X	X	0

$J_C = \overline{Q}_A$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	X	X	X	X
1	1	X	0	0

$K_A = \overline{Q}_B$

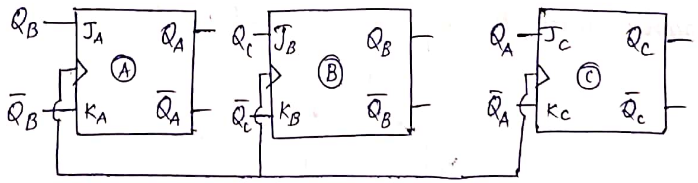
	00	01	11	10
0	X	X	0	X
1	X	X	0	1

$K_B = \overline{Q}_C$

	00	01	11	10
0	X	0	0	X
1	X	X	1	X

$K_C = Q_A$

Step-05:



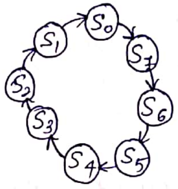
6. using positive Edge triggering SR flip-flop design a Counter which converts in the following Sequence

000, 111, 110, 101, 100, 011, 010, 001, 000

- $2^3 \geq N$

$8 \geq 8$.

- SR flip-flop
- Excitation table.



Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	$S_A R_A$	$S_B R_B$	$S_C R_C$
0	0	0	1	1	1	1 0	1 0	1 0
0	0	1	0	0	0	0 X	0 X	0 1
0	1	0	0	0	1	0 X	0 1	1 0
0	1	1	0	1	0	0 X	X 0	0 1
1	0	0	0	1	1	0 1	1 0	1 0
1	0	1	1	0	0	X 0	0 X	0 1
1	1	0	1	0	1	X 0	0 1	1 0
1	1	1	1	1	0	X 0	X 0	0 1

Q_A \ Q_BQ_C	00	01	11	10
0	1	0	0	0
1	0	X	X	X

$S_A = \bar{Q}_A \bar{Q}_B \bar{Q}_C$

Q_A \ Q_BQ_C	00	01	11	10
0	1	0	X	0
1	1	0	X	0

$S_B = \bar{Q}_B \bar{Q}_C$

	00	01	11	10
0	1	0	0	1
1	1	0	0	1

$S_C = \bar{Q}_C$

Q_A \ Q_BQ_C	00	01	11	10
0	0	X	X	X
1	1	0	0	0

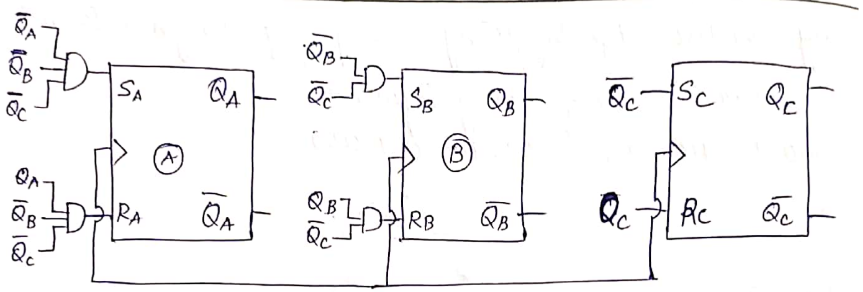
$R_A = Q_A \bar{Q}_B \bar{Q}_C$

Q_A \ Q_BQ_C	00	01	11	10
0	0	X	0	1
1	0	X	X	1

$R_B = Q_B \bar{Q}_C$

Q_A \ Q_BQ_C	00	01	11	10
0	0	1	1	0
1	0	1	1	0

$R_C = Q_C$



09.) Design a Synchronous counter using JK flip-flop to count the following Sequence. 7, 4, 3, 1, 6, 0, 7.

- $N = 6.$

  $2^n \geq 6.$

  $2^3 \geq 6$

  $8 \geq 6.$

- JK flip flop

- Excitation table

Q_A	Q_B	Q_C	Q_A^*	Q_B^*	Q_C^*	$J_A K_A$	$J_B K_B$	$J_C K_C$
0	0	0	1	1	1	1 X	1 X	1 X
0	0	1	1	1	0	1 X	1 X	X 1
0	1	0	X	X	X	X X	X X	X X
0	1	1	0	0	1	0 X	X 1	X 0
1	0	0	0	1	1	X 1	1 X	1 X
1	0	1	X	X	X	X X	X X	X X
1	1	0	0	0	0	X 1	X 1	0 X
1	1	1	1	0	0	X 0	X 1	X 1

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	1	1	0	X
1	X	X	X	X

$J_A = \bar{Q}_B$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	1	1	X	X
1	1	X	X	X

$J_B = 1$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	1	X	X	X
1	1	X	X	0

$J_C = \bar{Q}_B$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	X	X	X	X
1	1	X	0	1

$K_A = \bar{Q}_C$

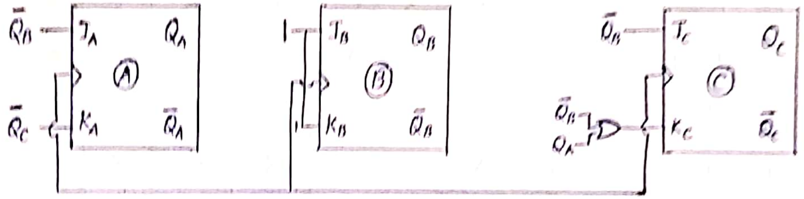
$Q_A \backslash Q_BQ_C$	00	01	11	10
0	X	X	1	X
1	X	X	1	1

$K_B = 1$

$Q_A \backslash Q_BQ_C$	00	01	11	10
0	X	1	0	X
1	X	X	1	X

$K_C = \bar{Q}_B + Q_A$

Design:



10.) Design and implement a synchronous decade counter using T flip flop. draw the timming diagram

Step:01 :- N=10

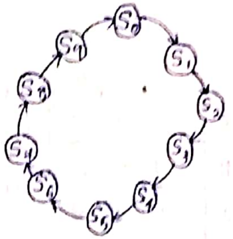
$2^n \geq 10$

$2^4 \geq 10$

$16 \geq 10$

Step:02: T flip flop.

Step:03: Excitation table.



Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	$J_A K_A$	$J_B K_B$	$J_C K_C$
0	0	0	0	0	1	0 X	0 X	1 X
0	0	1	0	1	0	0 X	1 X	X 1
0	1	0	0	1	1	0 X	X 0	1 X
0	1	1	1	0	0	1 X	X 1	X 1
1	0	0	1	0	1	X 0	0 X	1 X
1	0	1	1	1	0	X 0	1 X	X 1
1	1	0	1	1	1	X 0	X 0	1 X
1	1	1	0	0	0	X 1	X 1	X 1

Q_A \ Q_BQ_C	00	01	11	10
0	0	0	1	0
1	X	X	X	X

$J_A = Q_BQ_C$

Q_A \ Q_BQ_C	00	01	11	10
0	X	X	X	X
1	0	0	1	0

$K_A = Q_BQ_C$

Q_A \ Q_BQ_C	00	01	11	10
0	0	1	X	X
1	0	1	X	X

$J_B = Q_C$

Q_A \ Q_BQ_C	00	01	11	10
0	X	X	1	0
1	X	X	1	0

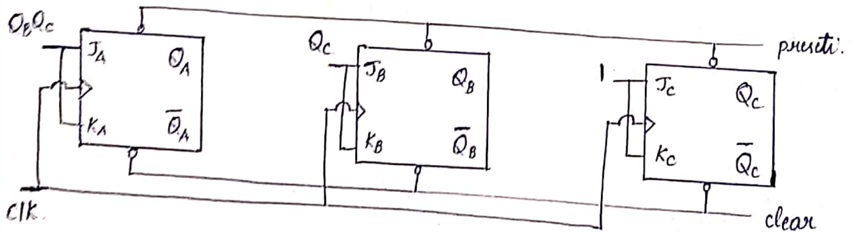
$K_B = Q_C$

Q_A \ Q_BQ_C	00	01	11	10
0	1	X	X	1
1	1	X	X	1

$J_c = 1$

Q_A \ Q_BQ_C	00	01	11	10
0	X	1	1	X
1	X	1	1	X

$K_c = 1$



) Design and implement a Synchronous 3 bit up/down counter using JK flip flop.

- $2^3 \geq 8$
- JK flip flop.
- Excitation table.

M	present state			Next state			J_AK_A	J_BK_B	J_CK_C
	A	B	C	A'	B'	C'			
0	0	0	0	0	0	1	0X	0X	1X
0	0	0	1	0	1	0	0X	1X	X1
0	0	1	0	0	1	1	0X	X0	1X
0	0	1	1	1	0	0	1X	X1	X1
0	1	0	0	1	0	1	X0	0X	1X
0	1	0	1	1	1	0	X0	1X	X1
0	1	1	0	1	1	1	X0	X0	1X
0	1	1	1	0	0	0	X1	X1	X1
1	0	0	0	1	1	1	1X	1X	1X
1	0	0	1	0	0	0	0X	0X	X1
1	0	1	0	0	0	1	0X	X1	1X
1	0	1	1	0	1	0	0X	X0	X1
1	1	0	0	0	1	1	X1	1X	1X
1	1	0	1	1	0	0	X0	0X	X1
1	1	1	0	1	0	1	X0	X1	1X
1	1	1	1	1	1	0	X0	X0	X1

Step: 04

$MQ_A \backslash Q_BQ_C$	00	01	11	10
00	0	0	1	0
01	X	X	X	X
11	X	X	X	X
10	1	0	0	0

$J_A = M\bar{Q}_B\bar{Q}_C + \bar{M}Q_BQ_C$

$MQ_A \backslash Q_BQ_C$	00	01	11	10
00	0	1	X	X
01	0	1	X	X
11	1	0	X	X
10	1	0	X	X

$J_B = M\bar{Q}_C + \bar{M}Q_C$

$MQ_A \backslash Q_BQ_C$	00	01	11	10
00	1	X	X	1
01	1	X	X	1
11	1	X	X	1
10	1	X	X	1

$J_C = 1$

$MQ_A \backslash Q_BQ_C$	00	01	11	10
00	X	X	X	X
01	0	0	1	0
11	1	0	0	0
10	X	X	X	X

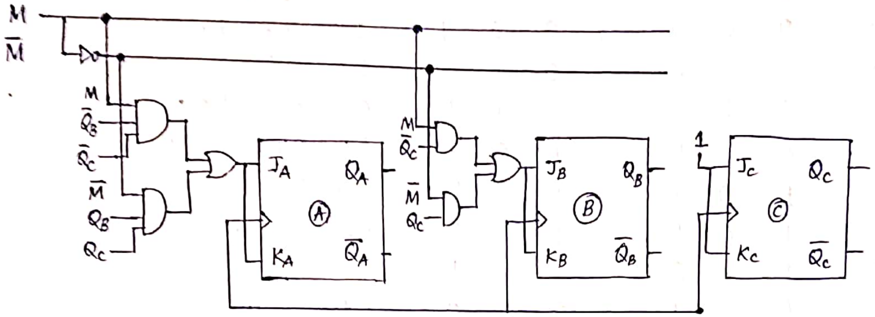
$J_A = M\bar{Q}_B\bar{Q}_C + \bar{M}Q_BQ_C$

$MQ_A \backslash Q_BQ_C$	00	01	11	10
00	X	X	1	0
01	X	X	1	0
11	X	X	0	1
10	X	X	0	1

$K_B = \bar{M}Q_C + M\bar{Q}_C$

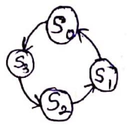
$MQ_A \backslash Q_BQ_C$	00	01	11	10
00	X	1	1	X
01	X	1	1	X
11	X	1	1	X
10	X	1	1	X

$K_C = 1$



13) Design a Mod 4 Synchronous down counter using SR flip-flop

- $2^2 \geq 4$
- SR flip-flop
- Excitation Table



present state		Next state		SR flip flop.	
Q_A	Q_B	Q_A^*	Q_B^*	S_AR_A	S_BR_B
1	1	1	0	X 0	0 1
1	0	0	1	0 1	1 0
0	0	0	0	0 X	0 1
1	1	1	1	1 0	1 0

K-Map:

$Q_A \backslash Q_B$	0	1
0	X	0
1	0	1

$S_A = Q_AQ_B$

$Q_A \backslash Q_B$	0	1
0	0	1
1	X	0

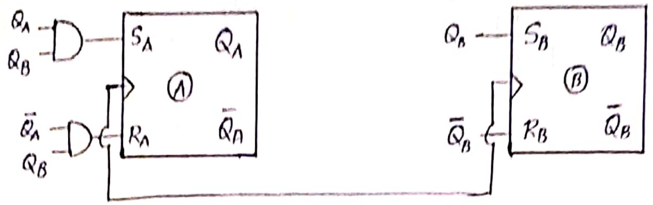
$R_A = \bar{Q}_AQ_B$

$Q_A \backslash Q_B$	0	1
0	0	1
1	0	1

$S_B = Q_B$

$Q_A \backslash Q_B$	0	1
0	1	0
1	1	0

$R_B = \bar{Q}_B$



14.) Design Synchronous Mod-6 counter using D-flipflop to generate the sequence 0, 2, 3, 6, 5, 1, 0...

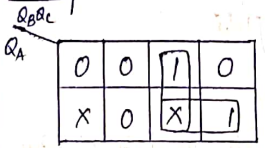
- $2^3 \geq 6$

$8 \geq 6$

- D flipflop
- Excitation Table

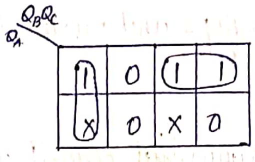
present state			Next state			D-flip-flop		
Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	D_A	D_B	D_C
0	0	0	0	1	0	0	1	0
0	0	1	0	0	0	0	0	0
0	1	0	0	1	1	0	1	1
0	1	1	1	1	0	1	1	0
1	0	0	x	x	x	x	x	x
1	0	1	0	0	1	0	0	1
1	1	0	1	0	1	1	0	1
1	1	1	x	x	x	x	x	x

K-Map



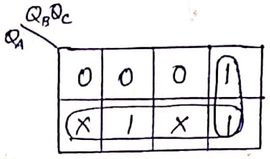
0	0	1	0
x	0	x	1

$D_A = Q_BQ_C + Q_BQ_A$



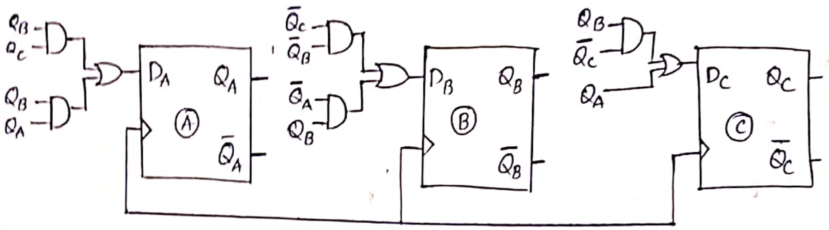
1	0	1	1
x	0	x	0

$D_B = \bar{Q}_B\bar{Q}_C + \bar{Q}_AQ_B$



0	0	0	1
x	1	x	1

$D_C = Q_B\bar{Q}_C + Q_A$



Mealy and Moore Model:

In Synchronous or clocked Sequential circuits, clocked flip flop's are called as a Memory Elements, which changes their individual states in Synchronism with the periodic clock signal. Therefore, the change in status of flip-flop's and change in state of the entire circuit Occurs at the transistion of the clock Signal. The states of the output of the flip-flop in the sequential circuit gives the state of the Sequential circuit.

Present state: The status of state Variables, at Some time, 't' before the next clock edge, represent Condition called present state

Next state: The Status of state Variables, at Some time, 't+1' represents a condition called Next state.

The Synchronous or clocked Sequential circuits are represented by two models.

1) **Moore Model**: The output depends only on the present state of the flip flops.

2) **Mealy Model**: The output depends on both the present state of the flip flops and on the inputs.

Moore Model:

* Figure (a) shows the Sequential circuit which consists of two JK flipflop's and AND Gate. The circuit has one input 'x' & one

output 'y'

* As shown in the figure (a), input is used to determine the output of the flipflop. It is not used to determine the output. The output is derived using only present states of the flip-flops
* In general form the Moore model can be represented with its block schematic as shown in figure (b) & (c)

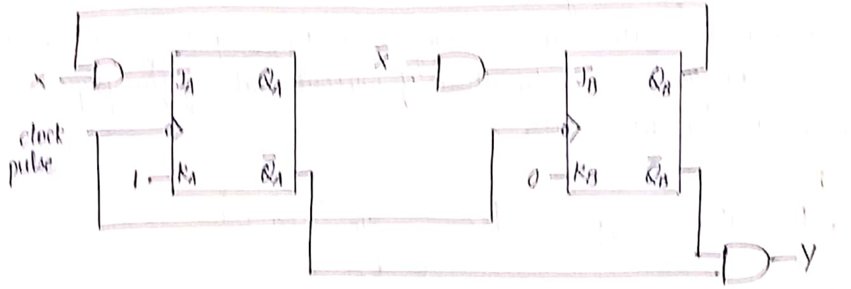


figure (a) Example of Moore Model.

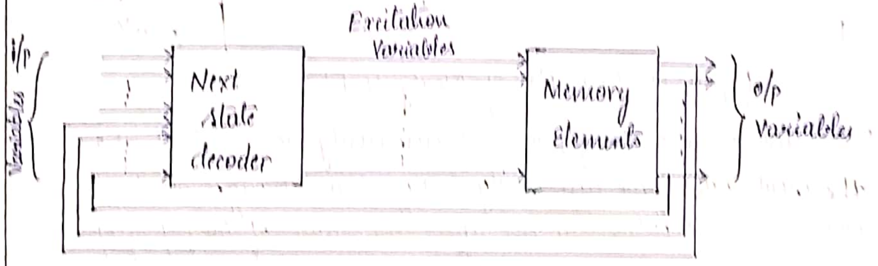


figure (b) Moore Model.

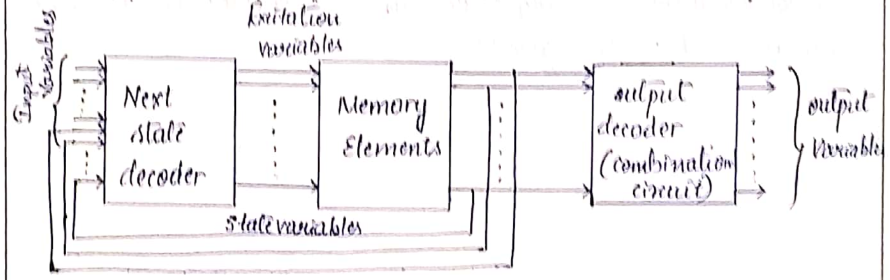


figure (c) Moore circuit model with an output decoder

* In the Moore Model, as output depends only on present state of flipflop's, it appears only after the clock pulse is applied, i.e., it varies in synchronism with the clock input.

Mealy Model:

* when the output of the sequential circuit depends on both the present state of flip flop's and on the inputs, the sequential ckt is referred to as Mealy model.
* figure (a) shows the sample Mealy Model, the output of the circuit is derived from the combination of present state of flip flops and inputs of the circuit.

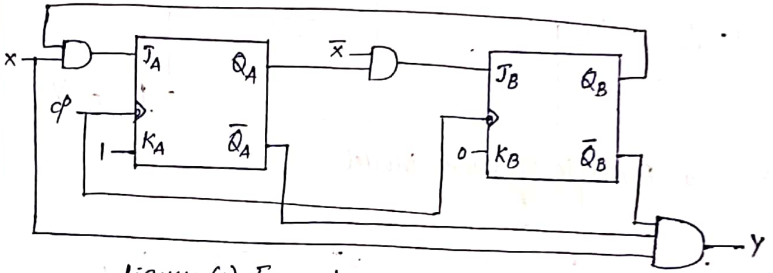


figure (a) Example of Mealy Model

* As shown the above figure, we can easily realize that change in the input within, we can easily realize that changes in the input within the clock pulses can not affect the state of the flip-flop. However, they can affect the output of the circuit.
* In general form the Mealy Model can be represented with its block schematic as shown in figure (b).

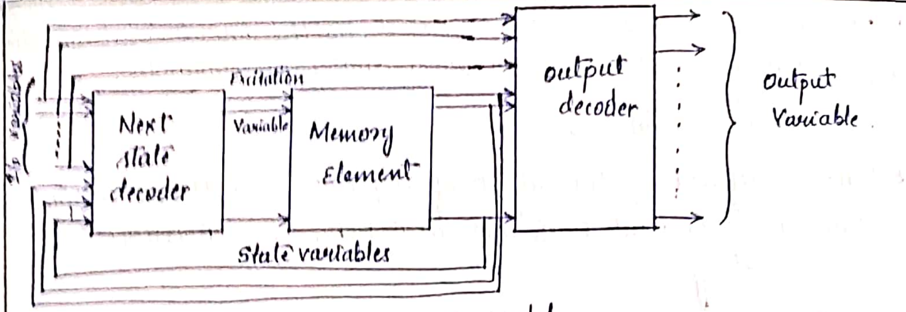


figure (b) Mealy circuit Model.

Moore v/s Mealy Model:

Moore Model	Mealy Model
1) It's output is a function of present state only	1) It's output is a functionof present state as well as present input
2) Input changes does not affect the output.	2.) Input changes may affect the output of the circuit
3.) Moore Model requires more number of states for implementing some functions	3) It requires less number of states for implementing same function.

State Machine Notations:

* In the state Machine, the Boolean Variables have different names according to other generation plane.

Input Variables: All Variables that Originate outside the Sequential machine are called input variables.

output Variables: All variables that Exit the Sequential machine are called output variables.

State Variables: The output of flip flop's defines the state of a. Sequential Machine. Therefore, the state variables are flipflop o/p's

Excitation Variables: Excitation Variables are the inputs to the flip flops Excitation Variables are Generated by the input Combinati logic operating on the state variables and input Variables.

State and State Variables :

* State is defined by the output of flip-flops. In the state Machin. State Variables and states are related by the expression

$$2^x = y$$

where,

x = number of state variables

y = Maximum number of possible states

The 4 state Variables can represent a Maximum of 16 states.

present state and Next state :

* In state Machines, it is necessary to distinguish state Variables before and after the clock pulse. state Variable 'A' Canbe represented as A before the arrival of a clock pulse and as A^+ after the arrival of a Synchronizing clock pulse. The idea of present state and next state illustrated in figure (a).

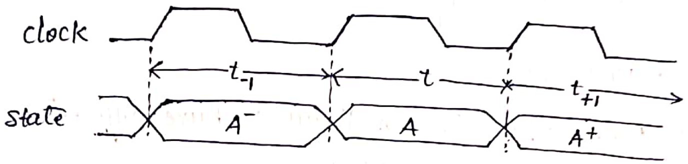


figure (a) illustrating present state and Next state.

present state : The status of all state Variables, at some time t, before the next clock Edge, represents condition called present state.

Next state : The status of all state Variable, at some time, t+1, represents a condition called Next state.

state diagram : state diagram is a pictorial representation of a behaviour of a Sequential circuit

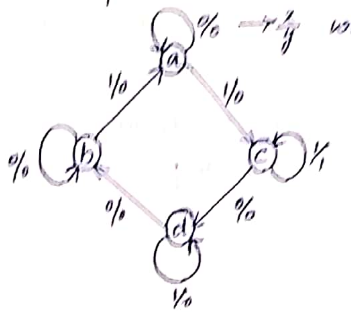
Mealy Model: * figure (a) shows a state diagram; the state is represented by the circle (a,b,c,d) states

* Transition between states is indicated by directed lines connecting the circles

* A directed line connecting the circle with itself indicates that next state is same as present state.

* Binary number inside the each circle identifies the state represents by the circle

$$\frac{0}{0} = \frac{\text{Input Value}}{\text{output value.}}$$

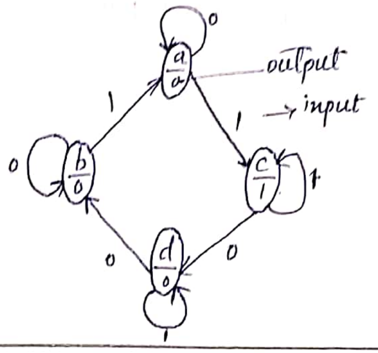


→ x/y where x = input
y = output.

Moore Model :

* In this directed lines are labelled with only one binary number representing the state of the input, that causes the transition.

* The output state is indicated within the circle, below the present state because output state depends only on the present state



State table:

* Although the state diagram provides a description of the behaviour of a Sequential circuit that is easy to understand to proceed with the implementation of the circuit, it is convinent to translate the information contained in the state diagram into a tabular form called state Synthesis table or simply state table.

Mealy Model State table:

Present state	Next state		output	
AB	X=0 AB	X=1 AB	X=0 Y	X=1 Y
a	a	c	0	0
b	b	a	0	0
c	d	c	0	1
d	b	d	0	0

* This table shows the state table for the static diagram of Mealy. It represents relationship between input, output and flip flop states

* It consists of three Sections labeled present state & output.

Moore Model State table:

present state	Next state		output
AB	X=0 AB	X=1 AB	Y
a	a	c	0
b	b	a	0
c	d	c	1
d	b	d	0

* In case of Moore circuit the output Section has only one column since output does not depend on input.

Transition table :

* A transition table takes the state table one step further. The state diagram and state table represent state using symbols or names. In the transition table specific state variable values are assigned to each state

* Assignment of values to state variables is called state assignment.

Transistion table of Mealy Model.:

present state	Next state		output	
	X=0	X=1	X=0	X=1
AB	AB	AB	Y	Y
0 0	0 0	1 0	0	0
0 1	0 1	0 0	0	0
1 0	1 1	1 0	0	1
1 1	1 0	1 1	0	0

a = 00
b = 01
c = 10
d = 11

Transistion table of Moore Model:

present state	Next state		output
	X=0	X=1	
AB	AB	AB	Y
0 0	0 0	1 0	0
0 1	0 1	0 0	0
1 0	1 1	1 0	1
1 1	0 1	1 1	0

Synchronous Sequential Circuit Analysis:

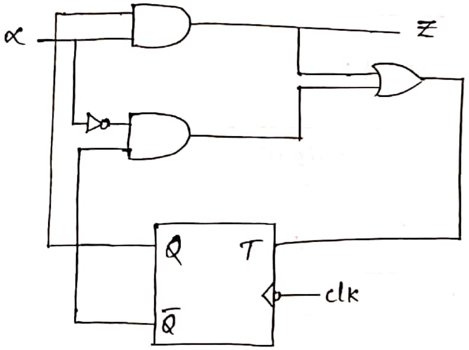
* The behaviour of a sequential network is determined from the inputs the outputs and the states of its flip flop. Both the inputs and the next state are function of the inputs and the present state

* The analysis of Sequential circuit consists of obtaining a table or a diagram for the time sequence of inputs outputs and internal states.

* The Success of analysis or design of sequential network depends largely on the aids and systematic techniques such as transition table, state tables, state diagram & static Equation used in these process

problems:

1) Construct the transition table, state table and state diagram for the Mealy sequential circuit Given below.



Solution: Consider the Sequential circuit to be analysed as shown above figure

* Let us see the steps to analyze the Given Synchronous Sequential circuit

Step 01: Determine the flip flop input Equation and the output Equations from the Sequential circuit.

$Z = \alpha Q$

$T = \alpha Q + \bar{\alpha}\bar{Q}$

Step 02: Derive the transition equations using characteristic equation

the transistion Equation for T flip-flop is.

$Q^{+} = T \oplus Q$

$$\boxed{Q^{+} = (\alpha Q + \bar{\alpha}\bar{Q}) \oplus Q}$$

∵ according to circuit

$T = \alpha Q + \bar{\alpha}\bar{Q}$.

Step 03: plot the transition table

present state	Next state		output	
	$\alpha=0$	$\alpha=1$	$\alpha=0$	$\alpha=1$
Q	Q^{+}	Q^{+}	Z	Z
a= 0	1	0	0	0
b= 1	1	0	0	1

Here only 1 flip flop in circuit So $2^1 = 2$ ∴ 2 inputs

Let $Q=0$ & $\alpha=0$

Then $Q^{+} = (\alpha Q + \bar{\alpha}\bar{Q}) \oplus Q$

$= (00 + 11) \oplus 0$

$= 1 \oplus 0$

$= 1$

$Q=1$ $\alpha=1$

Then $Q^{+} = (\alpha Q + \bar{\alpha}\bar{Q}) \oplus Q$

$= (11 + 00) \oplus 1$

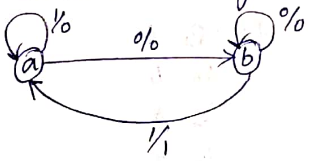
$= 1 \oplus 1$

$= 0$.

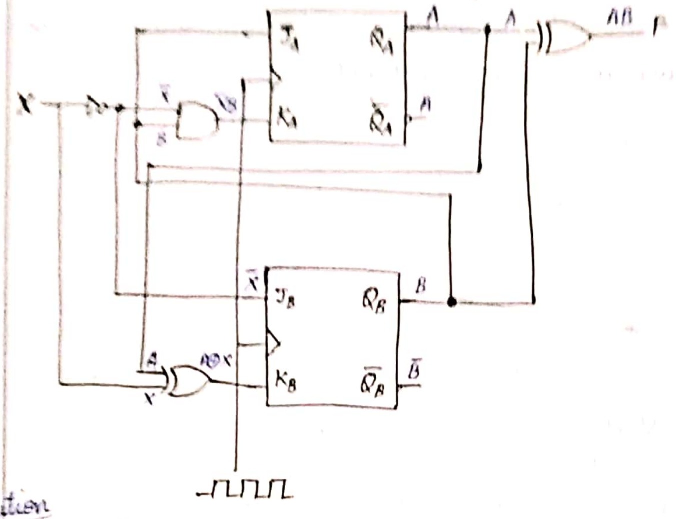
Step 04: Draw the state table

present state	Next state		output	
	$\alpha=0$	$\alpha=1$	$\alpha=0$	$\alpha=1$
(0) a	b	a	0	0
(1) b	b	a	0	1

Step: 05: Draw state diagram.



2) Construct the transition table, state table and state diagram for the Moore Sequential circuit given below.



Solution

Step 01: Determine the flip-flop input Equations and the output Equations from the Sequential circuit.

$$F = A \oplus B$$

$J_A = B$, $K_A = \bar{X}B$, $JB = \bar{X}$, $KB = X \oplus A$

Step 02: Derive the transition equation

The transistion Equations for JK flip flop's can be derived from the characteristic equation of JK flip flop as follows

$$Q^+ = J\bar{Q} + \bar{K}Q$$

$$A = Q_A^+ = J_A\bar{Q}_A + \bar{K}_A Q_A$$

$$= B\bar{Q}_A + \overline{\bar{X}B}\,Q_A$$

$$= B\bar{Q}_A + (\bar{\bar{X}} + \bar{B})Q_A$$

$$= B\bar{Q}_A + (X + \bar{B})Q_A$$

$$B = Q_B^+ = JB\bar{Q}_B + \bar{K}_B Q_B$$

$$= \bar{X}\bar{Q}_B + \overline{X \oplus A}\,Q_B$$

$$= \bar{X}\bar{B} + \overline{X \oplus A} \cdot B$$

where $Q_A = A$

$Q_B = B$

$\bar{Q}_A = \bar{A}$

$\bar{Q}_B = \bar{B}$

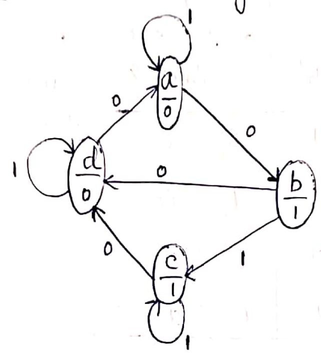
Step:03: Transition table.

present state	Next state		output
	X = 0	X = 1	
A B	A' B'	A' B'	F = A⊕B
0 0	0 1	0 0	0
0 1	1 1	1 0	1
1 0	1 1	1 0	1
1 1	0 0	1 1	0

Step:04: Draw the state table

present state	Next state		output
	X=0	X=1	
A B	A' B'	A' B'	F
a	b	a	0
b	d	c	1
c	d	c	1
d	a	d	0

Step:05: Draw state diagram



Q5 A sequential circuit with 2 D flip-flops A and B and input x and output Y is specified by the following next state and output equations.

$A(t+1) = Ax + Bx$

$B(t+1) = \bar{A}x$

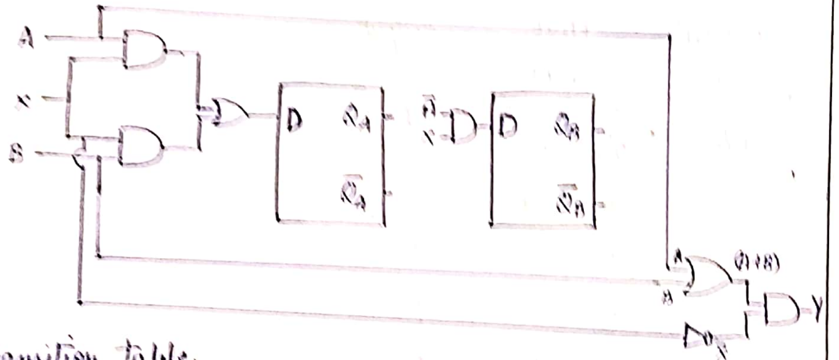
$Y = (A+B)\bar{x}$

(i) Draw the logic diagram of the circuit

(ii) Derive the state table

(iii) Derive the state diagram

(i) logic diagram of the circuit.



(ii) Transition table.

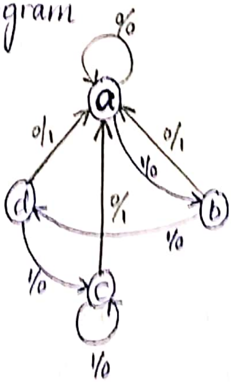
present state	Next state				output	
	x = 0		x = 1		x = 0	x = 1
A B	A'	B'	A'	B'	$Y = (A+B)\bar{x}$	
0 0	0	0	0	1	0	0
0 1	0	0	1	1	1	0
1 0	0	0	1	0	1	0
1 1	0	0	1	0	1	0

Output depends on input Hence it is a Mealy circuit

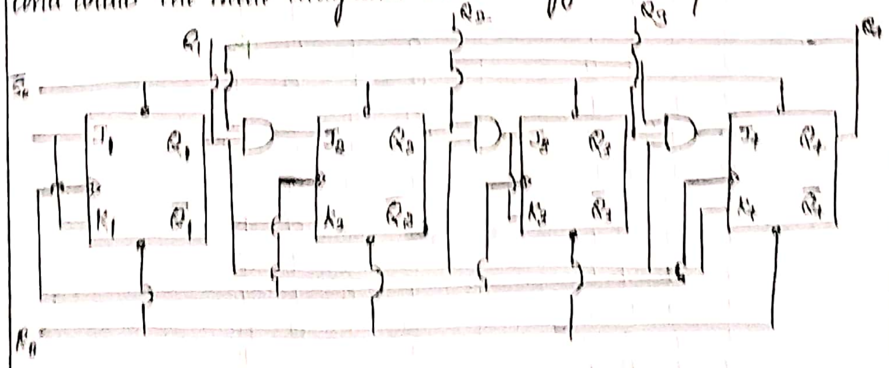
(iii) State table

present state	Next state		output y	
	X=0	X=1	X=0	X=1
	A^+B^+	A^+B^+		
a	a	b	0	0
b	a	d	1	0
c	a	c	1	0
d	a	c	1	0

(iv) State diagram



04) For the circuit shown in figure, write down the state table and draw the state diagram and analyze the Operation.



Solution (i) Inputs for each flipflop's .

$J_1 = 1$ $K_1 = 1$

$J_2 = Q_1 Q_4$ $K_2 = Q_1$

$J_3 = Q_2 Q_1$ $K_3 = Q_2 Q_1$

$J_4 = Q_1 Q_2 Q_3$ $K_4 = Q_1$

(ii) transition Equation for each flip-flop.

$$JK = J\bar{Q} + \bar{K}Q$$

$$Q_1^+ = J_1\bar{Q}_1 + \bar{K}_1 Q_1$$

$$= \bar{Q}_1$$

$$Q_2^+ = Q_1 Q_4 \bar{Q}_2 + \bar{Q}_1 Q_2$$

$$Q_3^+ = Q_1 Q_2 \bar{Q}_3 + \overline{Q_1 Q_2} Q_3$$

$$= Q_1 Q_2 \bar{Q}_3 + (\bar{Q}_1 + \bar{Q}_2) Q_3$$

$$Q_4^+ = Q_1 Q_2 Q_3 \bar{Q}_4 + \bar{Q}_1 Q_4$$

(iii) Transition table.

prusent state				Next state .			
Q_4	Q_3	Q_2	Q_1	Q_4^+	Q_3^+	Q_2^+	Q_1^+
0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	0	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	0	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	0	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

iv) State Assignment = To Assign the state to particular Sequence

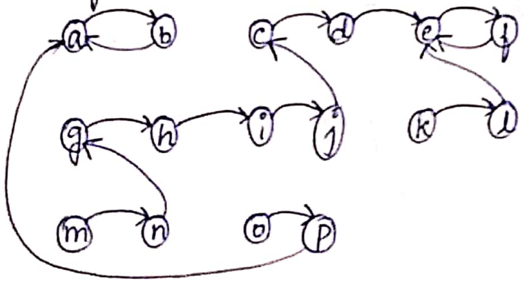
a=0000, b=0001, c=0010, d=0011, e=0100, f=0101, g=0110, h=0111

i=1000, j=1001, k=1010, L=1011, m=1100, n=1101, o=1110, p=1111.

v) State table :=

present state	Next state.
a	b
b	a
c	d
d	e
e.	f
f	e
g	h
h	i
i	j
j	c
k	l
L	e
m	n
n	g
o	p
p	a

vi) State diagram.

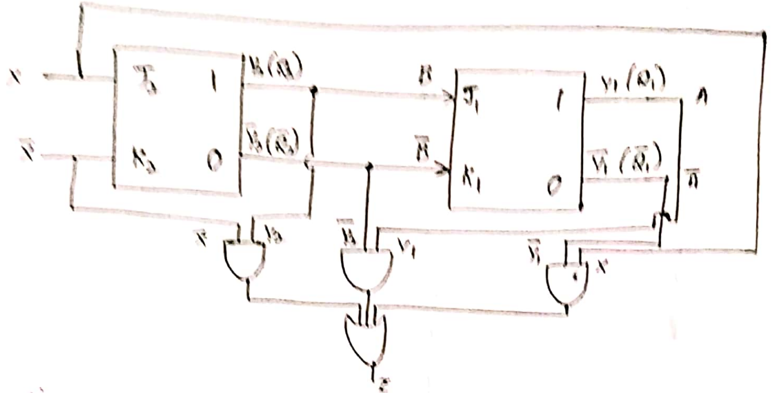


Q8) Analyze the synchronous ckt of the figure (clock not shown but is implicit)

i) write down the excitation and o/p functions

ii) form the Excitation and state table

iii) Give a word description of ckt operation.



Solution: $J_2 = x$, $J_1 = y_2$ and $Z = \bar{x}y_2 + \bar{y}_2 y_1 + \bar{y}_1 x$

$K_2 = \bar{x}$, $K_1 = \overline{y_2}$

For JK flip flop $Q^+ = J\bar{Q} + \bar{K}Q$.

$$J_2^+ = Q_2^+ = J_2\bar{Q}_2 + \bar{K}_2 Q_2$$

$Q_2 = Y_2$

$Q_1 = Y_1$

$$= x\bar{Q}_2 + xQ_2$$

$$= x\bar{y}_2 + xy_2 = x(\bar{y}_2 + y_2) = x$$

$$J_1^+ = Q_1^+ = J_1\bar{Q}_1 + \bar{K}_2 Q_2$$

$$= Y_2(\bar{Q}_1 + Q_1)$$

$$= y_2(\bar{y}_1 + y_1)$$

$$= y_2$$

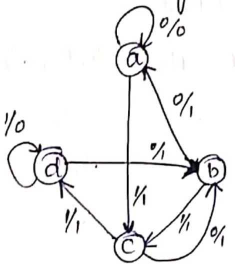
iii) Transition state:

Present state	Next state		output	
	x=0	x=1	x=0	x=1
y_2 y_1	y_2^+ y_1^+	y_2^+ y_1^+		
0 0	0 0	1 0	0	1
0 1	0 0	1 0	1	1
1 0	0 1	1 1	1	1
1 1	0 1	1 1	1	0

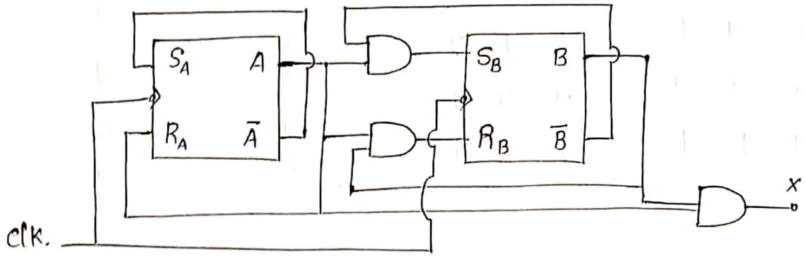
iv) State Table.

present state	Next state		output	
	X=0	X=1	X=0	X=1
y_2 y_1	y_2^+ y_1^+	y_2^+ y_1^+	Z.	
a	a	c	0	1
b	a	c	1	1
c	b	d	1	1
d	b	d	1	0

v) State diagram.



6) Give output function, transistion table and state diagram by analyzing the sequential circuit shown in figure.



Solution: $S_A = \bar{A}$ $S_B = A\bar{B}$ $X = AB$

$R_A = A$ $R_B = AB$

ii) characteristics Equation

$$Q_A^+ = S_A + \bar{R}_A Q_A$$
$$= \bar{A} + \overset{0}{\cancel{\bar{A}A}}$$
$$= \bar{A}$$

$$Q_B^+ = S_B + \bar{R}_B Q_B$$
$$= A\bar{B} + \overline{AB}(B)$$
$$= A\bar{B} + (\bar{A}+\bar{B})B$$
$$= A\bar{B} + \bar{A}B + \overset{0}{\cancel{B\bar{B}}}$$
$$= A \oplus B$$

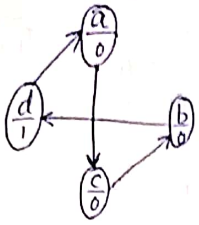
iii) Transition Table

present state		Next state		output
Q_A	Q_B	Q_A^+	Q_B^+	x
0	0	1	0	0
0	1	1	1	0
1	0	0	1	0
1	1	0	0	1

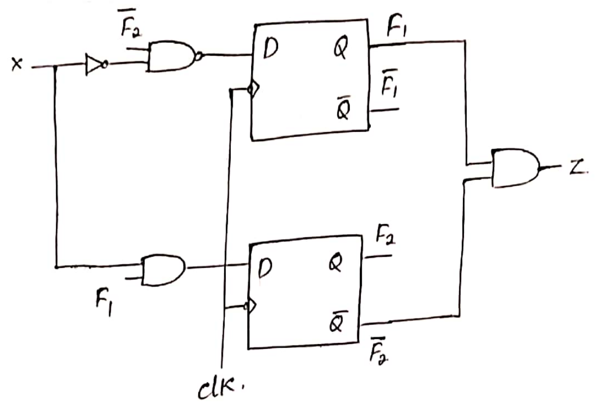
iv) State Table.

present State	Next State	output.
Q_A Q_B	Q_A^+ Q_B^+	x
a	c	0
b	d	0
c	b	0
d	a	1

v) State diagram.



07) For the logic diagram given in the figure. Derive the Excitation and output Equation ii) write the Next State Equation iii) Construct a transition Table iv) Draw the State diagram.



i) Determine the equation.

$D_1 = \overline{\overline{F_2}\,\overline{X}} = F_2 + X$

$D_2 = XF_1$

$Z = F_1\overline{F_2}$

ii) Transition Equation.

$Q_1^+ = F_1^+ = D_1 = F_2 + X$

$Q_2^+ = F_2^+ = D_2 = XF_1.$

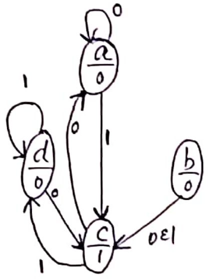
iii) Transition Table

present state	Next state		output
	X=0	X=1	
F_1 F_2	$F_1^+ F_2^+$	$F_1^+ F_2^+$	Z.
0 0	0 0	1 0	0
0 1	1 0	1 0	0
1 0	0 0	1 1	1
1 1	1 0	1 1	0

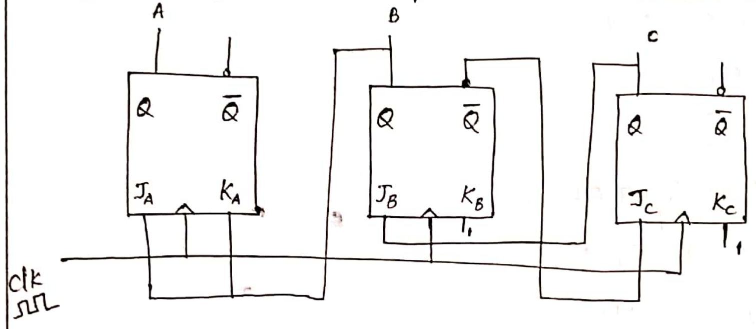
iv) state Table.:

present state	Next state		output
	X=0	X=1	
F_1 F_2	$F_1^+ F_2^+$	$F_1^+ F_2^+$	Z
a	a	c	0
b	c	c	0
c	a	d	1
d	c	d	0

v) state diagram.



8.) Analyze the Synchronous Sequential circuit shown in figure.



i) Determine the input flipflop Equation.

$J_A = B$, $K_A = B$

$J_B = C$ $\quad$ $J_C = \bar{B}$

$K_B = 1$ $\quad$ $K_C = 1$

ii) Transition Equation

$$J_A K_A = Q_A^+ = J_A \bar{Q}_A + \bar{K}_A Q_A$$
$$= B\bar{A} + \bar{B}A$$
$$= A \oplus B$$

$$J_B K_B = Q_B^+ = J_B \bar{Q}_B + \bar{K}_B Q_B$$
$$= C\bar{B} + \bar{1}\cancel{B}^0$$
$$= C\bar{B}$$

$$J_C K_C = Q_C^+ = J_C \bar{Q}_C + \bar{K}_C Q_C$$
$$= \bar{B}\bar{C} + \bar{1}\cancel{C}^0$$
$$= \bar{B}\bar{C}$$

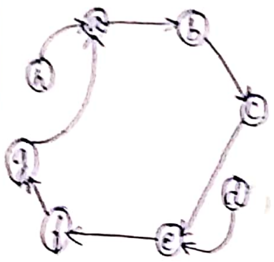
iii) Transition Equation table

present state			Next state.		
Q_A	Q_B	Q_C	Q_A^+	Q_B^+	Q_C^+
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	0	0	0
1	1	1	0	0	0

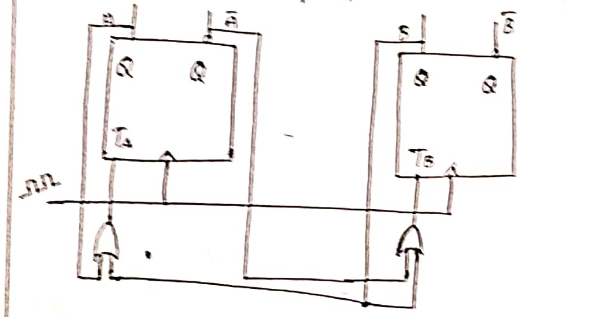
iv) state table:

present state	Next state.
Q_A Q_B Q_C	Q_A^+ Q_B^+ Q_C^+
a	b
b	c
c	e
d	e
e	f
f	g
g	a
h	a

state diagram.



(8) Analyze the following synchronous circuit



i) $T_A = A+B$, $T_B = \bar{A}+B$

ii) Transition Equation.

$$T^+ = T \oplus Q$$

$$T_A^+ = (A+B) \oplus A$$

$$T_B^+ = (\bar{A}+B) \oplus B$$

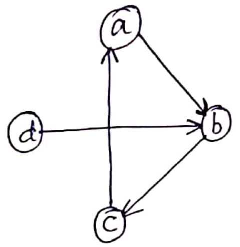
iii) Transition Table

present state		Next state	
A	B	A^+	B^+
0	0	0	1
0	1	1	0
1	0	0	0
1	1	0	0

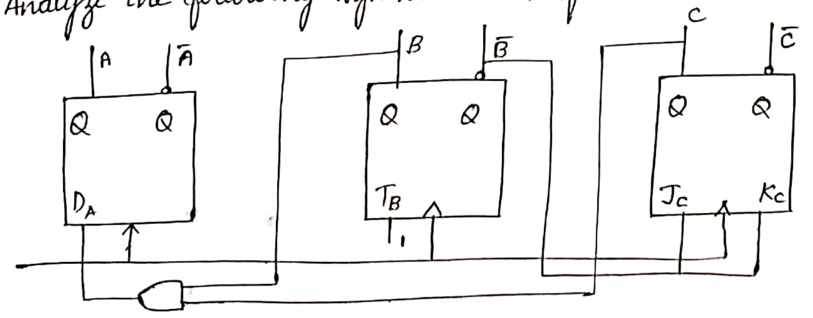
iv) state table

present state	Next state
A B	A^+ B^+
a	b
b	c
c	d
d	a

v) State diagram



10.) Analyze the following synchronous sequential circuit.



Solution/

$D_A = BC \quad , T_B = 1 \; , J_C = \overline{B} \; , K_C = \overline{B}$

Transition Equation :

For D $Q^{+} = D$

$Q_A^{1} = D_A = B.C$

for TB $- Q_B^{1} = T \oplus Q$

$= T_B \oplus Q_B$

$= 1 \oplus B$

for JK $- Q_C^{1} = J_C \overline{Q}_C + \overline{K}_C Q_C$

$= \overline{B}\, \overline{Q}_C + \overline{\overline{B}}\, Q_C$

$= \overline{B}\, \overline{C} + BC$

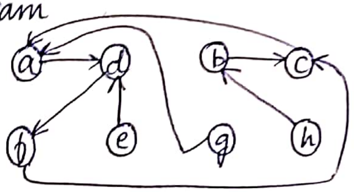
Transition Table :

present state			Next state		
Q_A	Q_B	Q_C	Q_A^{+}	Q_B^{1}	Q_C^{+}
0	0	0	0	1	1
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	1
1	0	0	0	1	1
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	1	0	1

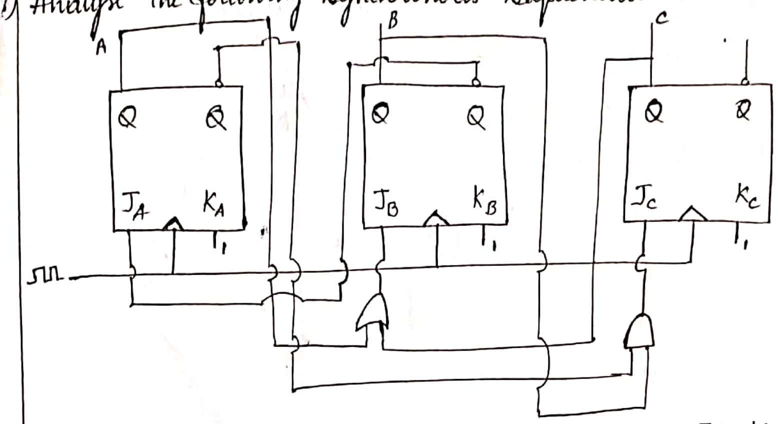
State Table

present State Q_A Q_B Q_C	Next State Q_A^{+} Q_B^{+} Q_C^{+}
a	d
b	c
c	a
d	f
e	d
f	c
g	a
h	b

state diagram



) Analyse the following Synchronous Sequential circuit.



Solution i) $J_A = \bar{B}$, $K_A = 1$, $J_B = A+B$, $K_B = 1$, $J_C = \bar{A}B$, $K_C = 1$

ii) Transition Eq^n

$$Q_A^+ = J_A K_A = J_A \overline{Q_A} + \bar{K_A} Q_A$$
$$= \bar{B}\bar{A} + \bar{1}A$$
$$= \bar{B}\bar{A} + 0$$
$$= \bar{B}\bar{A}$$

$$Q_B^+ = J_B K_B = J_B \overline{Q_B} + \overline{K_B} Q_B$$
$$= (A+B)\bar{B} + \bar{1}\, Q_B$$
$$= (A+B)\,\bar{B}$$

$$Q_C^+ = J_C K_C = J_C \overline{Q_C} + \overline{K_C} Q_C$$
$$= \bar{A}B\,\bar{c} + 0C$$
$$= \bar{A}B\bar{C}$$

iii) Transition Table

Present state			Next state		
Q_A	Q_B	Q_C	Q_A^+	Q_B^+	Q_C^+
0	0	0	1	0	0
0	0	1	1	0	0
0	1	0	0	0	1
0	1	1	0	0	0
1	0	0	0	1	0
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	0	0

iv) State Assignment

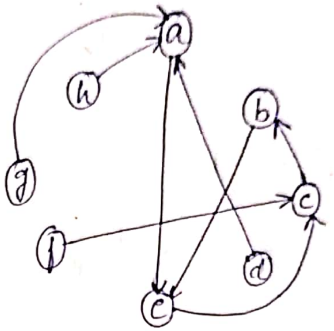
a = 000 , b = 001 , c = 010 , d = 011 , e = 100

f = 101 , g = 110 , h = 111.

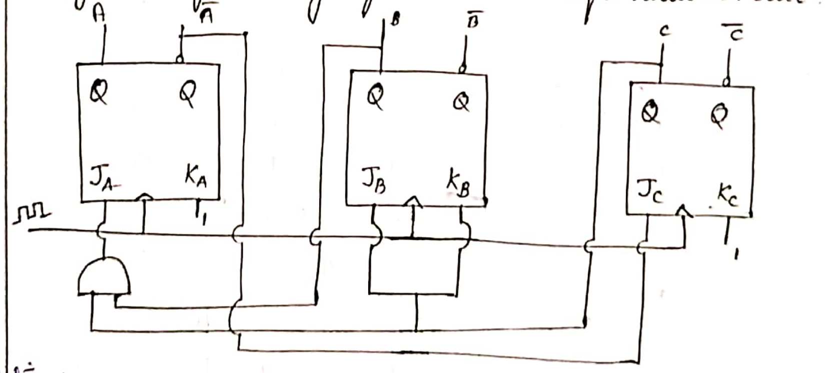
state table

Present state	Next state
Q_A Q_B Q_C	Q_A^+ Q_B^+ Q_C^+
a	e
b	e
c	b
d	a
e	c
f	c
g	a
h	a

state diagram



12) Analyse the following synchronous sequential circuit.



Solution: $J_A = BC$, $K_A = 1$, $J_B = C$, $K_B = C$, $J_C = \bar{A}$, $K_C = 1$

ii) Transition Equation.

$$J_A K_A = Q_A^+ = J_A \bar{Q}_A + \bar{K}_A Q_A$$
$$= BC\,\bar{A} + \bar{1}\,A$$
$$= BC\bar{A} + 0$$
$$= BC\bar{A}$$

$$J_B K_B = Q_B^+ = J_B \bar{Q}_B + \bar{K}_B Q_B$$
$$= C\,\bar{B} + \bar{C}\,B$$
$$= C\bar{B} + \bar{C}B$$
$$= C \oplus B$$

$$J_cK_c = Q_c^{+} = J_c\bar{Q}_c + \bar{K}_cQ_c$$
$$= \bar{A}\bar{C} + \bar{1}C$$
$$= \bar{A}\bar{C}$$

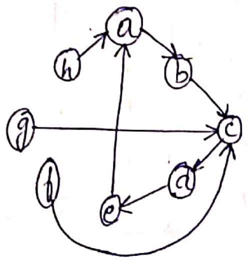
iii) Transition Table.

present state			Next state		
Q_A	Q_B	Q_C	Q_A^+	Q_B^+	Q_C^+
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	0	0	0

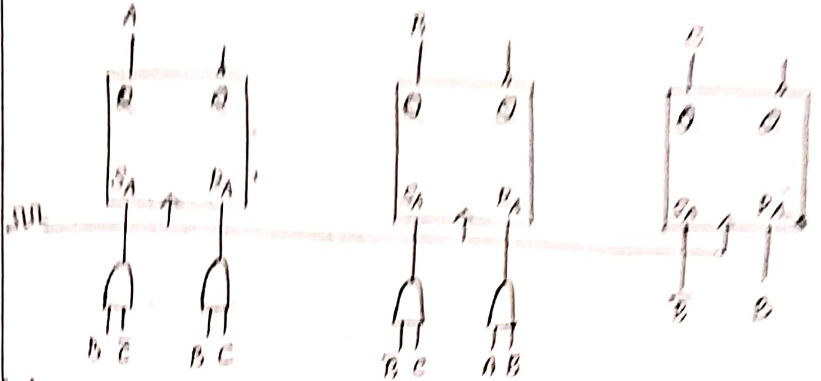
iv) State Table

present state	Next state
Q_A Q_B Q_C	Q_A' Q_B' Q_C
a	b
b	c
c	d
d	e
e	a
f	c
g	c
h	a

v) state diagram.



13.) Analyse the following Synchronous Sequential circuit.



Solution:

$S_A = B\bar{C}$, $R_A = BC$, $S_B = \bar{B}C$, $R_B = AB$, $S_C = \bar{B}$, $R_C = B$

Transition Equation

$Q^+ = S + \bar{R}Q$

$Q_A^+ = B\bar{C} + \overline{BC}\,Q_A$

$= B\bar{C} + (\bar{B} + \bar{C})A$

$Q_B^+ = S_B + \bar{R}_B Q_B$

$= \bar{B}C + (\overline{AB})B$

$= \bar{B}C + (\bar{A} + \bar{B})B$

$Q_C^+ = S_C + \bar{R}_C Q_C$

$= \bar{B} + \bar{B}C$

$= \bar{B}(1+c)$

$= \bar{B}$

Transition Table

present state			Next state		
Q_A	Q_B	Q_C	Q_A^+	Q_B^+	Q_C^+
0	0	0	0	0	1
0	0	1	0	1	1
0	1	0	1	1	0
0	1	1	0	1	0
1	0	0	1	0	1
1	0	1	1	1	1
1	1	0	1	0	0
1	1	1	1	0	0

iv) State Assignment.

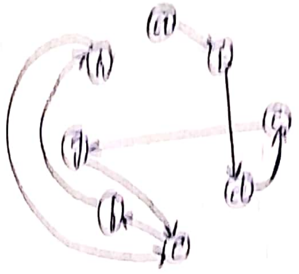
a=000 , b=001 , c=010 , d=011 , e=100

f=101 , g=110 , h=111.

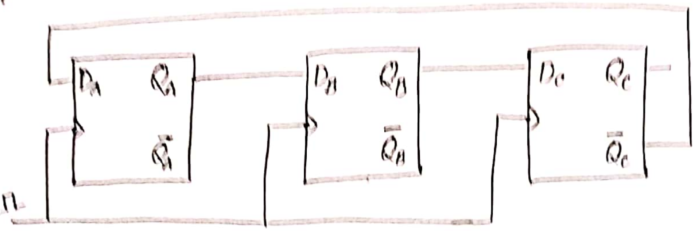
v) state table

present state			Next state		
Q_A	Q_B	Q_C	Q_A^+	Q_B'	Q_C'
	a			b	
	b			d	
	c			g	
	d			c	
	e			f	
	f			h	
	g			e	
	h			e	

vi) state diagram



ii) Give the Excitable for D flip-flop.



Solution $D_A = \bar{Q}_C$, $D_B = Q_A$, $D_C = Q_B$

Transition Equation.

$Q_B^+ = D_A^+ = \bar{Q}_A = \bar{C}$

$Q_A^+ = D_B = Q_B = A$

$Q_C^+ = D_C = Q_C = B$

Excitation Table for D

Q	Q^+	D
0	0	0
0	1	1
1	0	0
1	1	1

Transition table

present state			Next state		
Q_A	Q_B	Q_C	Q_A^+	Q_B^+	Q_C^+
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	0	1
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	1	1	1
1	1	1	0	1	1

Excitation table.

present state			Next state			flip flop		
Q_A	Q_B	Q_C	Q_A^+	Q_B^+	Q_C^+	D_A	D_B	D_C
0	0	0	1	0	0	1	0	0
0	0	1	0	0	0	0	0	0
0	1	0	1	0	1	1	0	1
0	1	1	0	0	1	0	0	1
1	0	0	1	1	0	1	1	0
1	0	1	0	1	0	0	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	1	1	0	1	1

Introduction To Digital Principles

Classification of Electronic Circuits.

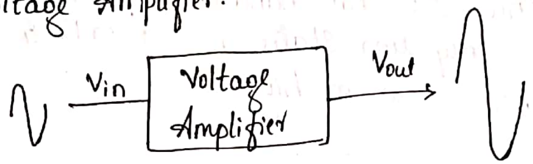
Based on type of signals they process

1) Analog Electronic Circuits.
2) Digital Electronic Circuits.

Analog Electronic Circuits.

These Circuits are designed for use with (Small signals) Analog signals. It exhibits linear operation.

Ex:- Voltage Amplifier.



The o/p is the faithful Amplified Version of the i/p voltage signal.

Analog Components : Resistors, capacitors etc

Digital Electronics Circuits:-

* These circuits are used with large signals. It Exhibits non-linear operation.

  Ex:- Remote Control Circuit for automatic Switching of a light.

* The signal is the Current from a light sensing circuit. The o/p signal is 'ON' or 'OFF' & it is not an amplified version of the i/p signal.

ⓐ Analog signal: It is a Continuously varying signal & all possible values are represented.

* The o/p is linearly proportional to the i/p.
* All naturally occurring physical phenomena are analog signals. Ex: Audio signals, sine wave, ~~etc~~ Temp, pressure

velocity, are signals that take all possible values b/n given limits.

(b) Digital Signal :-

The signal that can have only two discrete values is called digital signal.

Ex:- Square wave.

* An Electronic ckt that is designed for two State operation is Called digital Circuit. (or)

* An Electronic ckt that handle only digital Signal, it operates on only two States, i.e., 0 or 1, & on or off, True or false, Low or high.

Features of Digital Circuits.

→ In digital circuits and subsystems both i/p & o/p are digital signals.

* Advantages are:-

→ DC elements usually operate in one of the 2 States - ON/OFF States which results in simple Circuit.

→ DC are highly reliable, Extremely small in size & cost very less.

→ DC needs knowledge of Boolean Algebra i.e, simple & easy to understand.

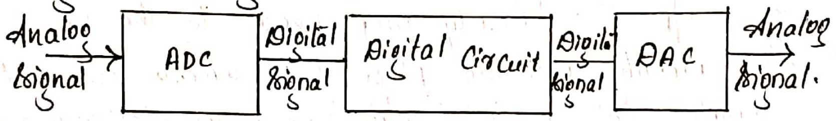
Digital circuits analysed practically.

→ Signals around us are analog which are difficult to process.

→ So we convert analog signal to digital using ADC circuit.

→ The digital signal is processed in a digital Circuit.

→ The digital o/p signal is converted back to analog signal using DAC Circuit.



Binary System: Digital circuits involves 2 possible states.

→ A System having only 2 states is said to be binary [bi=2]

→ Binary no. System has exactly 2 values 0 & 1.

→ There are only 2 Vtg levels - usually 0V & +5V indicated as H & L.

Logic: used to deal with process of determining truth (or) falsity of a statement.

Logic design: It is the application of a set of rules & Techniques for developing digital ckt to create a Solⁿ for some problems.

Binary variable: It is a variable or value changeable quantity but it can have only two values '0' or '1'.

Truth Table: It is a graphical representation of showing the Truth relationship b/n i/p & o/p variables.

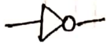
Logic Gate: It is a digital ckt with one or more i/p signal with only one o/p signal.

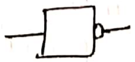
Logic diagram: A drawing madeup of logic Symbols Showing i/p & o/p Connections b/n various logical functions.

Logic Equation (or) Boolean Expression: A Boolean Expression is a Expression that result in a boolean value i.e, in a value of either True (or) false, (a) '0' or '1'

Logic gates | **Symbol** | **IEEE Symbol** | **Truth table.**

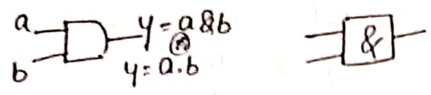
1) NOT





a	y
0	1
1	0

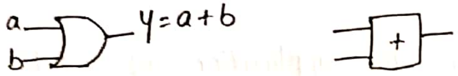
2) AND



$y = a \& b$ (or) $y = a.b$

a	b	y
0	0	0
0	1	0
1	0	0
1	1	1

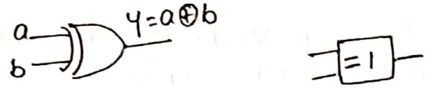
3) OR



$y = a+b$

a	b	y
0	0	0
0	1	1
1	0	1
1	1	1

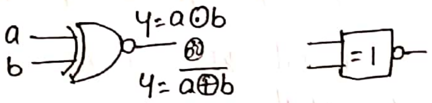
4) Ex-OR



$y = a \oplus b$

a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

5) Ex-NOR

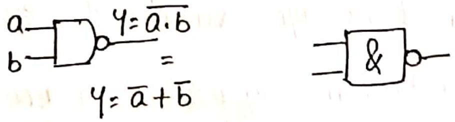


$y = a \odot b$ (or) $y = \overline{a \oplus b}$

a	b	y
0	0	1
0	1	0
1	0	0
1	1	1

universal gates:

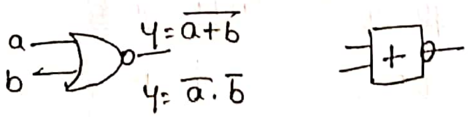
6) NAND



$y = \overline{a.b}$ = $y = \bar{a} + \bar{b}$

a	b	y
0	0	1
0	1	1
1	0	1
1	1	0

7) NOR



$y = \overline{a+b}$

$y = \bar{a} . \bar{b}$

a	b	y
0	0	1
0	1	0
1	0	0
1	1	0

The following are the most Common types of logic gates.

→ Resistor Transistor logic [RTL]:

Common in the middle of the 1960 cheap but only of Historical interest. A Typical Circuit is RTL923.

→ Diode Transistor logic (DTL):

Slow Compare to TTL & more Expensive than RTL.

→ Emitter Coupled logic (ECL):

Very fast but Expensive. High power consumption. A typical Circuit is 1004.

→ Transistor Transistor logic (TTL).

Moderately fast, cheap & Moderatly immune to Noise.

→ Complementary Metal oxide Semiconductor [CMOS].

Very low power, good Noise immunity, easy to Match (interface) to other digital (or) analog ckt, potentially very cheap. Typically ckt is 4001.

Boolean Algebra (or) Switching Algebra.

It is a different kind of algebra. Invented by George Boole (1854).

This is one of the method to Simplify the design of logic Circuits.

Boolean Algebra or Switching algebra is a system of Mathematical logic to perform different mathematical operations in binary system.

Theorems of Boolean algebra are used to Simplify logic function or boolean Equation or Switching Equation.

① Equivalence: If two binary variables a & b have Same values then they are said to be Equal.

i.e., $a=b$, $b=a$.

(2) Identity law:- [I_e]

Identity element doesnot change the value of Boolean Expression.

$a+I_e = a \longrightarrow I_e$ for OR operation is '0'.

$a.I_e = a \longrightarrow I_e$ for AND operation is '1'.

Ex: $a+0=a$ $\quad$ $a.1=a$

$1+0=1$ $\quad$ $0.1=0$

(3) Associative Law:

$$a+(b+c) = (a+b)+c$$
$$a(bc) = (ab)c.$$

(4) Commutative law:

$$a+b = b+a$$
$$ab = ba$$

(5) Distributive law:

$$a(b+c) = ab+ac.$$
$$ab+c = (a+c)(b+c)$$
$$a+bc = (a+b)(a+c)$$

(6) Inversion or Complementary:

$$a.\bar{a} = 0$$
$$a+\bar{a} = 1$$

(7) Dual property:

$$A+1 = 0$$
$$A.0 = 0$$

(8) Idempotency law:

$A+A = A$ $\qquad$ $A.A = A$

(9) Involution law:

$$\bar{\bar{a}} = a.$$

⑩ **Absorption law:**

$$a+ab=a$$

⑪ **De-Morgan's law:**

$$\overline{A+B} = \bar{A}\cdot\bar{B}$$

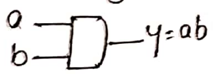
$$\overline{A\cdot B} = \bar{A}+\bar{B}$$

Simplify the following Boolean expression & implement them using basic & universal gates.

① **AND**

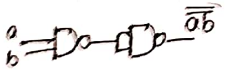
$y = ab$

Basic gate



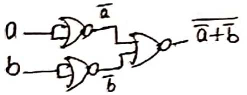
NAND

$y=\overline{\overline{ab}}$



NOR

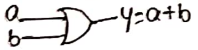
$y=\overline{\overline{ab}} = \overline{\bar{a}+\bar{b}}$



② **OR**

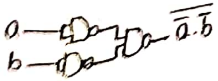
$y=a+b$

Basic gate

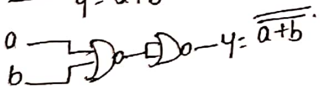


NAND

$y=\overline{\overline{a+b}} = \overline{\bar{a}.\bar{b}}$



NOR $y=\overline{\overline{a+b}}$



③ $y = \bar{a}b\bar{c} + \bar{a}bc + a\bar{b}\bar{c} + a\bar{b}c$

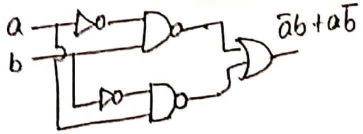
$= \bar{a}b\overset{1}{(\bar{c}+c)} + a\bar{b}\overset{1}{(\bar{c}+c)}$

$= \bar{a}b + a\bar{b}$

$= \underline{a \oplus b}$

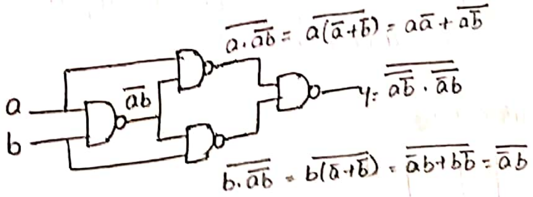
$y = \bar{a}b + a\bar{b}$

Basic gatir.



NAND gate

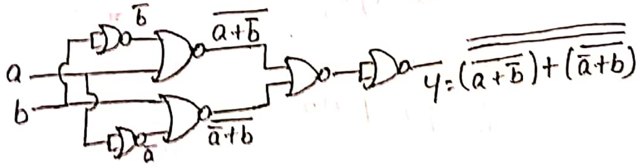
$$y = \overline{\overline{\bar{a}b + a\bar{b}}} = \overline{\overline{\bar{a}b} \cdot \overline{a\bar{b}}}$$



$\overline{a \cdot \overline{ab}} = \overline{a(\bar{a}+\bar{b})} = \overline{a\bar{a} + a\bar{b}}$
a
b
$\overline{ab}$
$y = \overline{\overline{a\bar{b}} \cdot \overline{\bar{a}b}}$
$\overline{b \cdot \overline{ab}} = \overline{b(\bar{a}+\bar{b})} = \overline{\bar{a}b + b\bar{b}} = \overline{\bar{a}b}$

NOR gate :

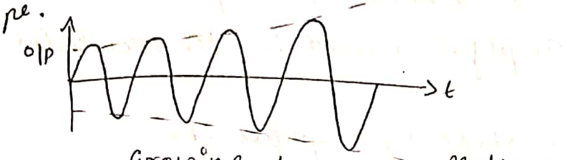
$$y = \overline{\overline{\bar{a}b + a\bar{b}}} = \overline{\overline{\bar{a}b} \cdot \overline{a\bar{b}}} = \overline{(\bar{\bar{a}} + \bar{b}) \cdot (\bar{a} + \bar{\bar{b}})} = \overline{\overline{\overline{(a+\bar{b})} + \overline{(\bar{a}+b)}}}$$
$$= \overline{\overline{\overline{(a+\bar{b})} + \overline{(\bar{a}+b)}}}$$



$\bar{b}$
$\overline{a+\bar{b}}$
a
b
$y = \overline{\overline{\overline{(a+\bar{b})} + \overline{(\bar{a}+b)}}}$
$\bar{a}$
$\overline{\bar{a}+b}$

1) $|A\beta|>1$

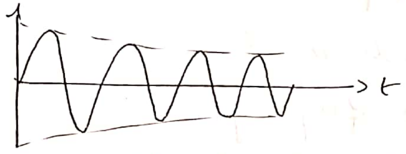
When the total phase shift around a loop is 0° or 360° then the o/p oscillates, but the oscillators are of growing type.



growing type of oscillations

2) $|A\beta|<1$,

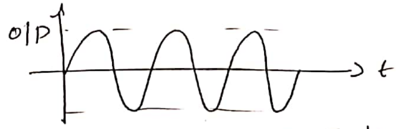
when the total phase shift around a loop is 0° or 360°, then the oscillations are of decaying type i.e. amplitude decreases exponentially & the oscillators finally cease.



Exponentially decaying oscillations.

3) $|A\beta|=1$,

when a total phase shift around a loop is 0° or 360°. then the ckt produces oscillations with constant frequency and amplitude called sustained oscillations



Sustained oscillations

Note:- For the feedback, the feedback signal should be in phase with i/p signal (v_s).

If Inverting amplifier is used in the oscillator ckt, then

→ 180° phase shift by amplifier ckt

→ 180° phase shift by feedback N/W.

So total phase shift around a loop is 360°.

If Non-Inverting amplifier is used in the oscillator ckt, then.

→ 0° phase shift by amplifier ckt

→ 0° phase shift by feedback n/w

So total phase shift around a loop is 0°.

Classification of oscillators:-

oscillators are classified based on the

* o/p waveform
* components used
* range of frequency

1) Based

"Jai Sri Gurudev"

[illegible]
Asst - Prof
Dept of ECE

Subject :- Digital Electronics

Module 05 :- Application of Digital Circuits

Programmable logic devices :-

A programmable logic device [PLD] is a general name for a digital integrated circuit capable of being programmed to provide a variety of different logic functions. In this section, we will discuss several types of combinational PLD's. Simple Combinational PLD's are capable of realizing from 2 to 10 functions of 4 to 16 variables with a single integrated circuit more complex PLD's may contain thousands of gates and flip flop. Thus, a single PLD can replace a large number integrated circuits and this leads to lower cost design. when a digital system is designed using a PLD, changes ~~to~~ the design can easily be made by changing the programming of the PLD without having to change the wiring in the system.

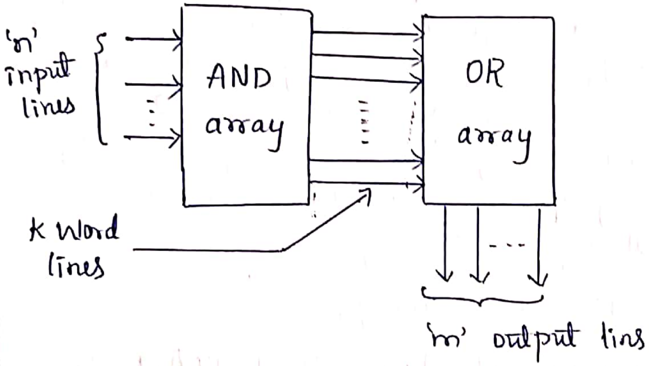
1. Programmable logic arrays :-

A programmable logic array (PLA) performs the same basic function as a ROM. A PLA with a 'n' input variables and 'm' output can realize m functions of n variables. The internal organisation of the PLA is different from that of the ROM. The Decoder is replaced with an AND array which realizes selected product terms of the input variable. The OR arrays ORs together the product terms needed to form the output functions, so a PLA implements a sum-of-products expression, while a ROM directly implements a truth-table.

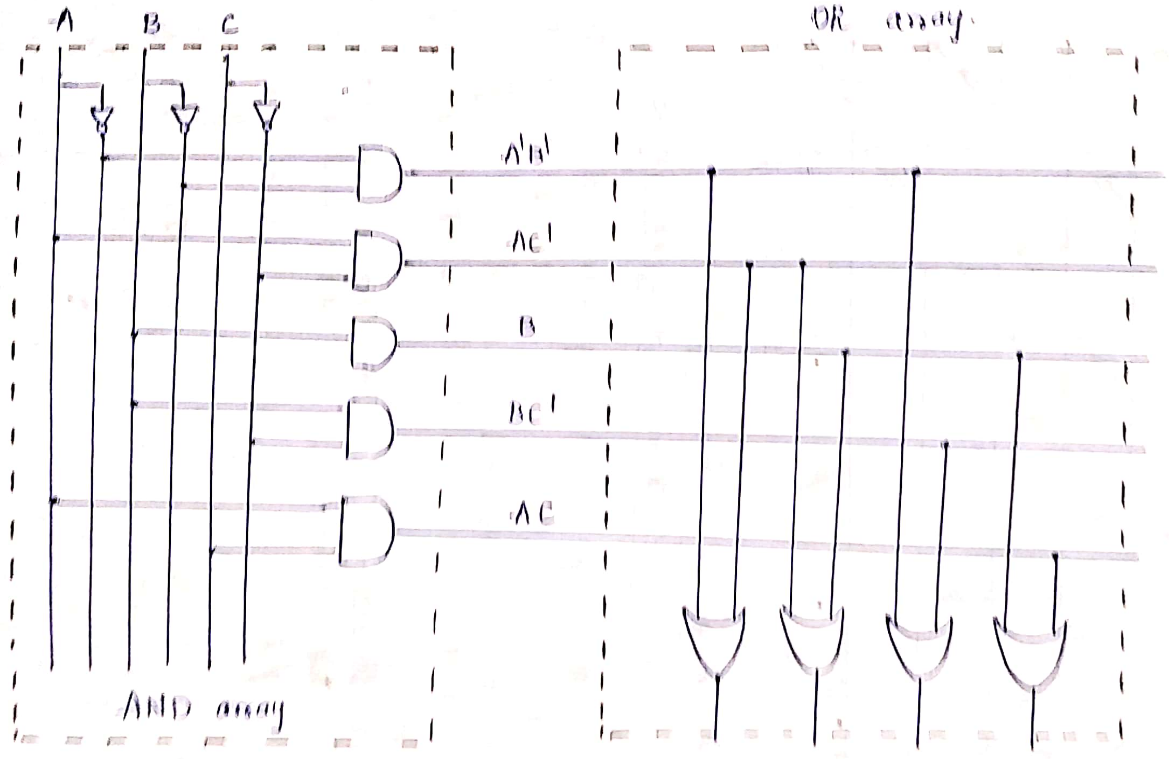
Figure shows a PLA which realizes the function of ROM. Product terms are formed in the AND array by connecting switching elements at appropriate points in the array. For example to form A'B' switching elements are used to connect the first word line with the A' and B' lines. Switching elements are connected in the OR array to select the product terms needed for the output function.

For example,

Fig (a)



Programmable logic array structure



fig(b) :- AND - OR array equivalent to fig (a).

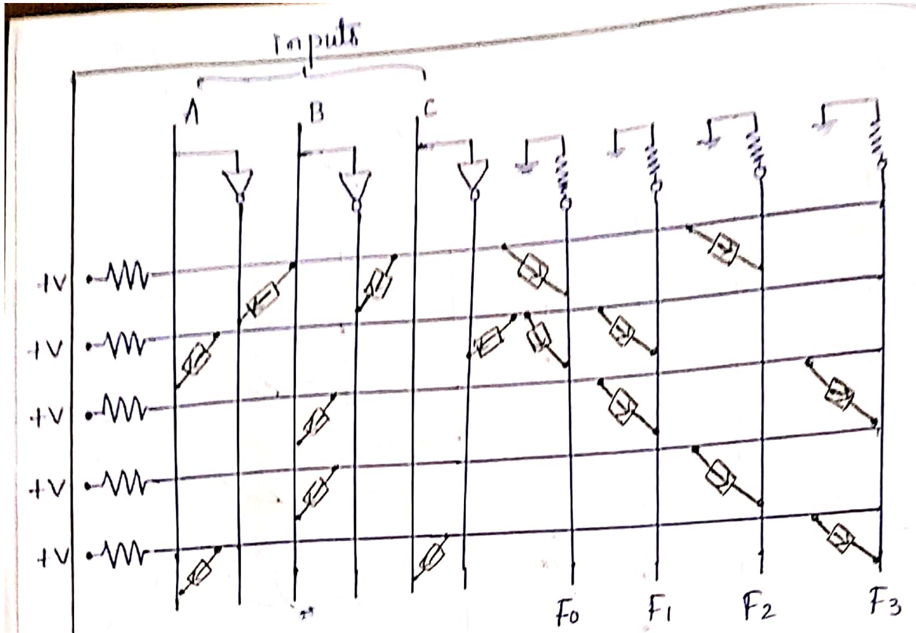


fig ⓒ :- PLA with three input, five product terms and four outputs

$F_0 = A'B' + AC'$, switching elements are used to Convert the $A'B'$ and AC' lines to the F_0 line. The Connections in the AND and OR arrays of this PLA make it equivalent to the AND-OR array fig ⓒ.

The Contents of a PLA Can be specified by a PLA table. Table 1 specifies the PLA in fig ⓑ. The input side of the table specifies the product terms. The symbol 0, 1 and so on indicate wheather a Variable is Complemented, not Complemented ⓞ not present in the Corresponding Product term. The output side of the table specifies which product terms appear in each output functions. A 1 ⓞ 0 indicates whether a given product term is present / not present in the Corresponding output functions. Thus, the first row of tabel-1 indicates that the term $A'B'$ is present in output functions F_0 and F_2, & also the 2^{nd} Row indicate that AC' is present in F_0 and F_1

Tabel 1:- PLB table for figure ⓒ.

Product term	A	B	C	F_0	F_1	F_2	F_3
	inputs			outputs			
$A'B'$	0	0	-	1	0	1	0
AC'	1	-	0	1	1	0	0
B	-	1	-	0	1	0	1
BC'	-	1	0	0	0	1	0
AC	1	-	1	0	0	0	1

2. Programmable array logic

The PAL [Programmable array logic] is a spEcial case of the programmable logic array in which the AND array is programmable and the OR array is fixed. The basic structure of the PAL is the same as the PAL. Here only the AND array is programmable, the PAL is less expensive than the more general PLA and the PAL is easier to program. For this reason, logic designers frequently use PAL's to replace individual logic gates among 2/ more OR gates; therefore each function to be realized can be simplified among two / more OR gates; ∴ each function to be realized can be simplified by itself without regard to common terms. For a given type of PAL, the number of AND terms that feed each output OR gate is fixed and limited of the number of And terms in a simplified function is too large. we may be forced to chose a PAL, with more gate inputs & fewer outputs.

Fig (d) PAL Segment

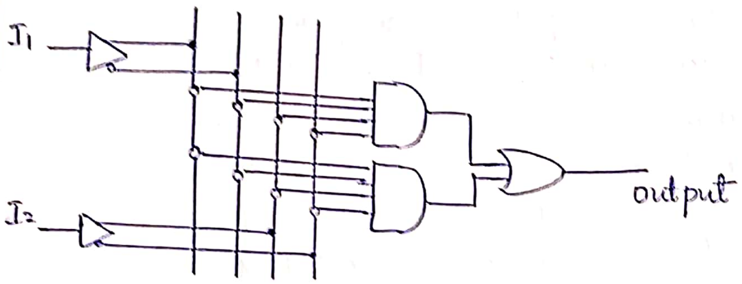


fig (d) (i) unprogrammed.

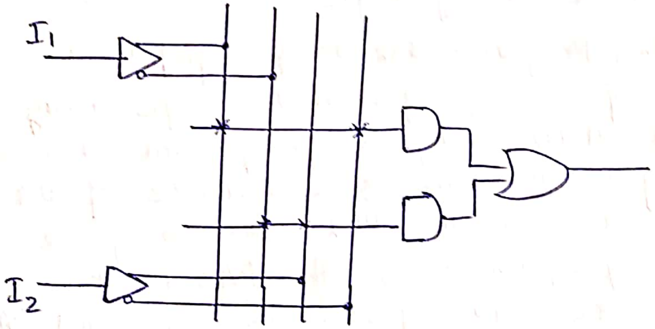
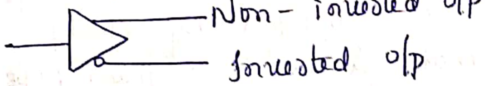


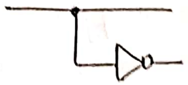
fig (d) (ii) programmed

when several logic functions must be realized.

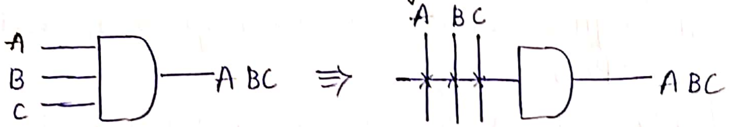
Fig (d) represents a segment of an unprogrammed PAL. the symbol represent an input



buffer which is logically equivalent to



A buffer is used because each PAL input must drive many AND gate inputs. when the PAL is programmed. Some of the interconnection points are programmed to make the desired connections to the AND gate inputs. Connections to the AND gate inputs in a PAL are represented by X's as shown.



As an examples we will use the PAL segment of figure @:- to realize the function $I_1 I_2' + I_1' I_2$. The X's in figure @ ii) indicate that I_1 and I_2' lines are connected to the first AND gate and the I_1' & I_2 lines are connected to the other gate.

When designing with PAL's. unlike the more general PLA, the AND terms can not be shared among two/ more OR gates; ∴ each function to be realized can be simplified by itself without regard to common terms. For a given type of PAL, the no. of And terms that feed each o/p OR gate is fixed & limited. we may be forced to choose a PAL with more gate inputs and fewer outputs.

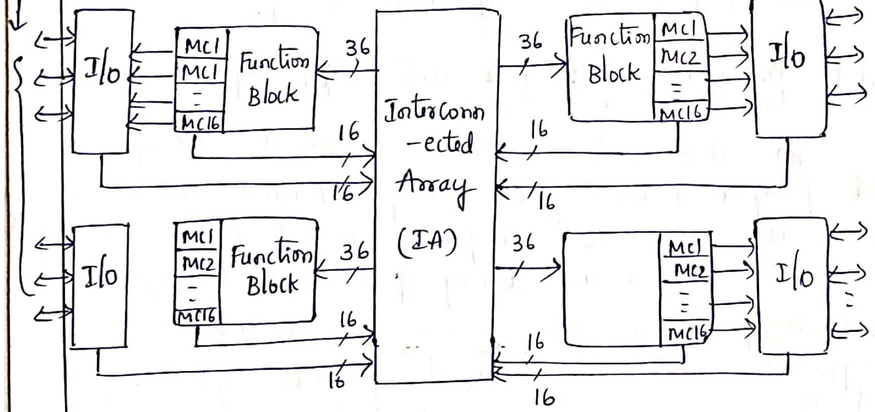
Complex programmable logic device

As integrated Circuit technology continues to improve more & more gates can be placed on a single chip. This has allowed the development of Complex programmable logic devices (CPLD's). Instead of a single PAL / PLA on a chip, many PAL's / PLA's can be placed on a single CPLD chip & interconnected. when storage elements such as flip-flops are also included on the same IC, a small digital system can be implemented with a single CPLD.

fig ⓐ shows the basic architecture of a xilinx XCR3064XL CPLD. This CPLD has 4 function block & each block has 16 associated macrocells (MC 1, MC 2, - - -) each function blocks is a programable AND-OR array that is configured as a PLA. Each macrocell contains a flip-flop and multiplexer that route signals from the functions block to the input-output (I/O) block / to the interconnect array (IA). The IA selects signals from the macrocell output / I/O blocks and connects them back to function block input. Thus a signal generated in one function block can be used as an i/p to any other function block. The I/o blocks provide an interface b/w the bi-directional I/o pins on the IC and the interior of the CPLD.

fig ⓑ ⇒ how a signal generated in the PLA is routed to an I/o pin through a macrocell. Any of the 36 inputs from the IA [or their Components] Can be Connected to any input of an 48 AND gates. Each OR gates can accept up to 48 product term inputs from the AND array. The macrocell logic in this diagram is a simplified version of the actual logic. The first Mux(1) Can be programmed to select the OR-gate output or its Complement. Details of the flip-flop operation. The Mux(2) at the output of the macro cell can be programmed to select either the combinational output (G) or the flip-flop output (Q). This output goes to the interconnect array and to the output cell. The output cell include three-state buffer(3) to drive the I/o pin. the buffer enable input

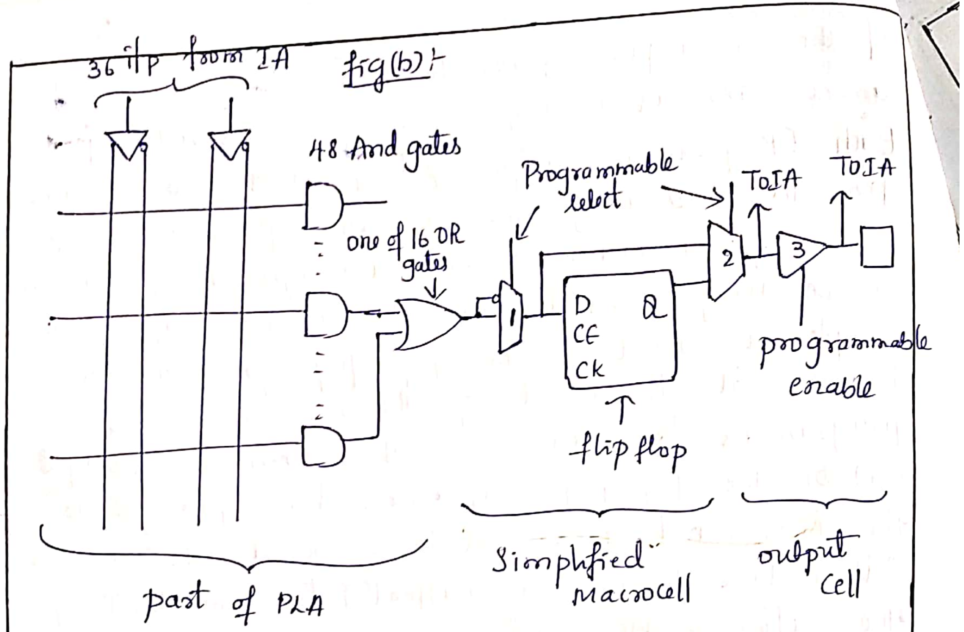
I/o pins fig(a)



Architecture of xiclinx XCR3064XL CPLD

Buffer enable input can be programmed from several sources when the I/o pin is used as an input, the buffer must be disable.

fig(b):-



CPLD function block and Macro cell.
[Simplified version of XCR3064XL].

Field programmable gate array [FPGA]

In this section, we introduce the use of field-programmable gate array (FPGA's) in combinational logic design. An FPGA is an IC that contains an array of identical logic cell with programmable inter connections. The user can program the functions realized by each logic cell and the connections b/w the cells. Figure ⓐ shows the layout of part of a typical FPGA. The interior of the FPGA consists of an array of logic cells also called configurable logic blocks (CLB's). The array of CLB is surrounded by a ring of I/p-o/p interface blocks. These I/O blocks connect the CLB signals to IC pins. The space b/w the CLB's is used to make connections b/w the CLB output and inputs.

Fig (b) shows a simplified version of a CLB. This CLB contains two function generators, a flip flop & various multiplexers for routing signals within the CLB. Each function generator has 4 i/p & can implement any function of up to four variables. The function generators are implemented as lookup table (LUT's). A 4 i/p LUT is essentially a reprogrammable ROM with 16 1 bit words. This ROM stores the truth table for the function being generated. The H multiplexer selects either F or G

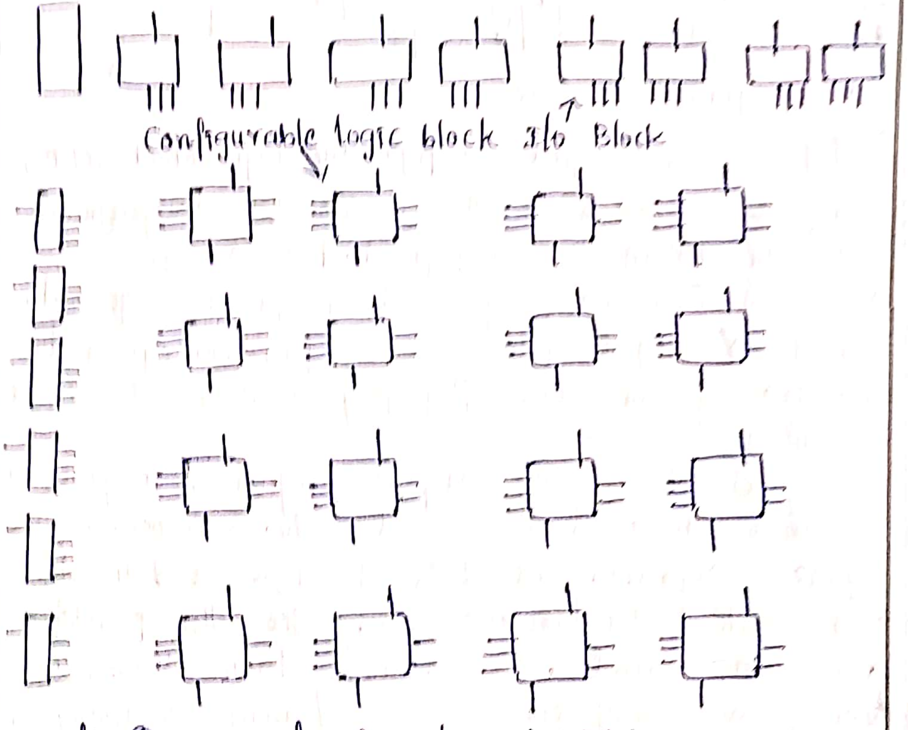
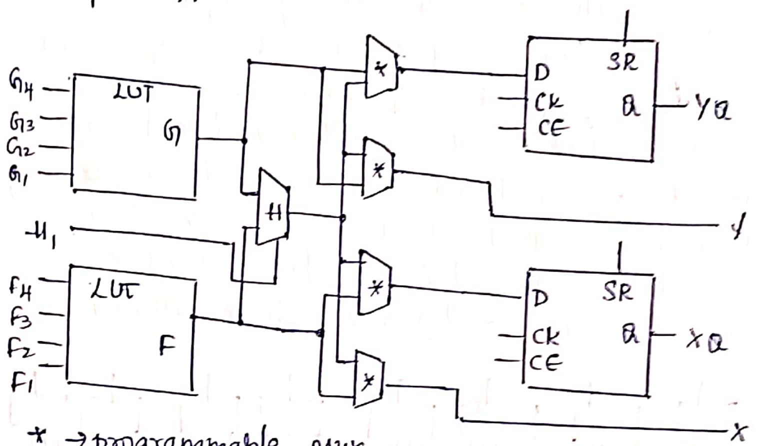


fig (a) Layout of a typical FPGA

depending on the value of H1, The CLB has two Combinational outputs (X and Y) and two flip-flop output (XQ and YQ). The X and Y outputs and the flip flop inputs are selected by programmable multiplexers.



* →programmable Mux.

fig ⓑ → Simplified Configurable logic block (CLB)

The select inputs to the Muxes are programmed when the FPGA is Configured. For example the X output Can Come from the F function generator, and the Y output from the H function generator. Operation of the CLB flip-flop will be described in unit II.

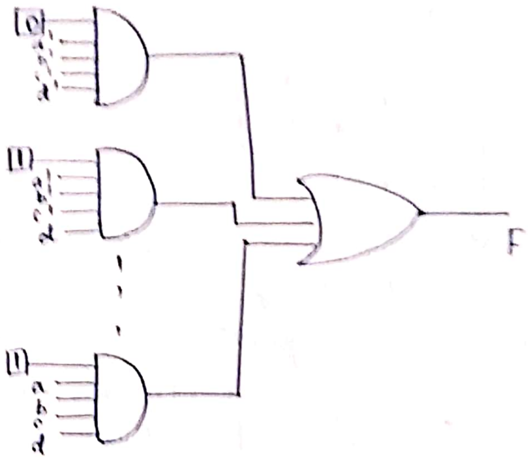
fig ⓒ shows one way to implement a function generator with input a, b, c, d. The numbers in the squares represent the bits stored in the LUT. These bits enable particular minterms. Bcoz the functions being implemented is stored as a truth table, a function with only one minterm (-with as many as 15 minterms requires a single function generator.

The functions F = abc and

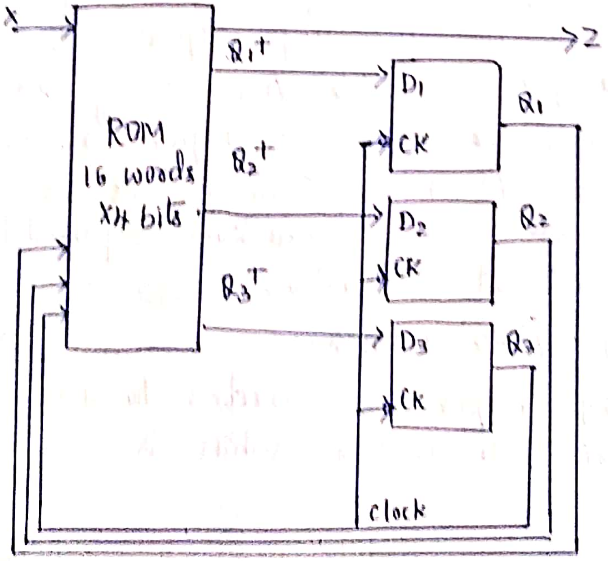
$F = a'b'c'd' + a'b'cd + a'bc'd + a'bcd' + ab'c'd + ab'cd' + abc'd' + abcd.$

each require a single function generator.

a	b	c	d	F
0	0	0	0	0
0	0	0	1	1
	⋮			⋮
1	1	1	1	1



Design of Sequential Circuits using ROM & PLA's.



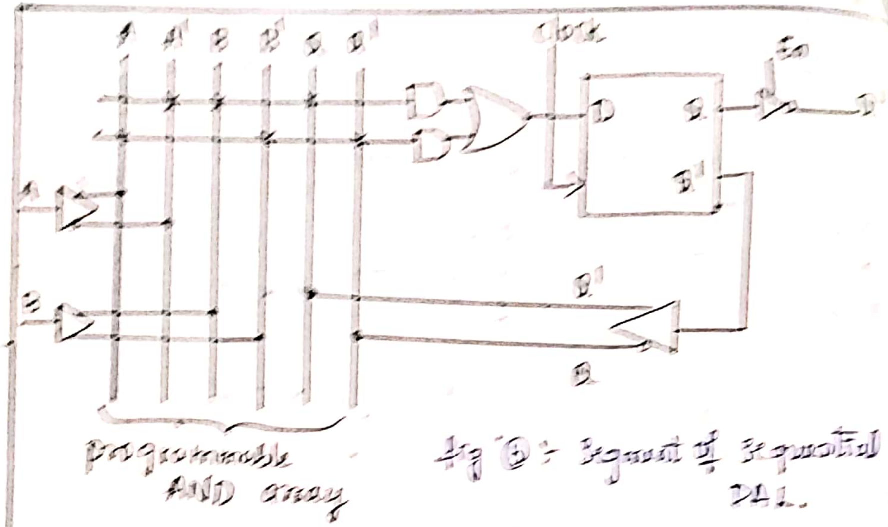


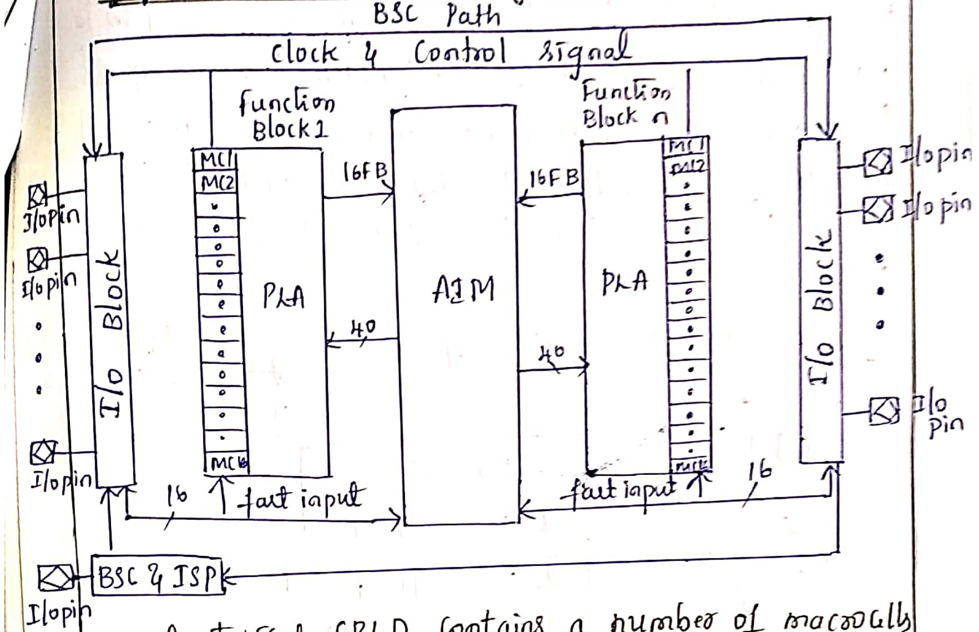
Fig ⑥ : Segment of Sequential PAL.

PAL's also provide a Convenient way of realizing sequential Circuits. PAL's are available which Contain D flip-flops that have their inputs driven from programmable array logic. Fig ⑥ shows a segment of a sequential PAL. The D flip-flop is driven from an OR gate which is fed by two AND gates. The flip-flop output is fed back to the programmable AND array through a buffer. Thus the AND gate input can be Connected to A, A', B, B', Q or Q'. The x's on the diagram show the Connections required to realize the next-state equation.

$$Q^+ = D = A'BQ' + AB'Q$$

The flip-flop output is Connected to an inverting tri-state buffer, which is enabled when En = 1.

Sequential Circuit design using CPLD's



A typical CPLD contains a number of macrocells that are grouped into function blocks. Connections b/w the function block are made through an interconnection array. each macrocell contains a flip-flops and an OR gate, which has its inputs connected to an AND gate array. some CPLD's are based on PAL's in which case each OR gate has a fixed set of AND gates associated with it other CPLD's are based on PLA's in which case any AND gate output within a function block can be connected to any OR gate input in that block.

Figure shows the structure of a Xilinx Coolrunner II CPLD. which uses a PLA in each function block. This CPLD family is available in sizes from two to 32 function block [32 to 512 macrocells]. each function block has 16 inputs from the AIM [advanced interconnection matrix] and up to 40 outputs to the AIM each function block PLA Contains the equivalent of 56 AND gates.

CPLD Implementation of a Mealy Machine

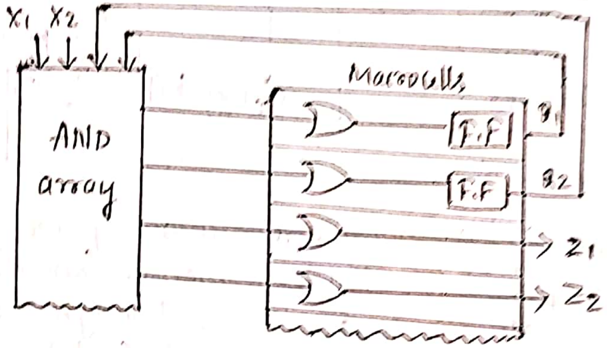


Fig shows how a Mealy sequental machine with two inputs, two outputs and two flip-flop can be implemented by a CPLD. Four macrocells are required, two to genrate the D inputs to the flip-flop and two to genrate the Z outputs. The flip flop output are fed back to the AND array input via interconnection matrix (not shown). The number of product terms required depends on the complexity of the equation for the D's and the Z's.

(2) CPLD implementation of a Shift register.

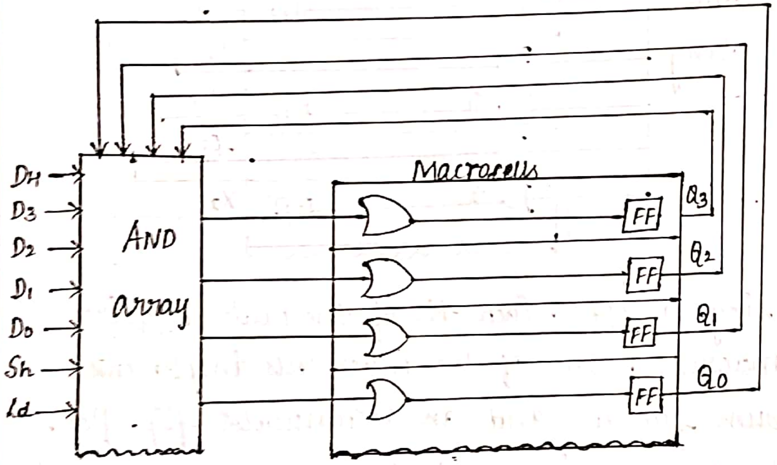
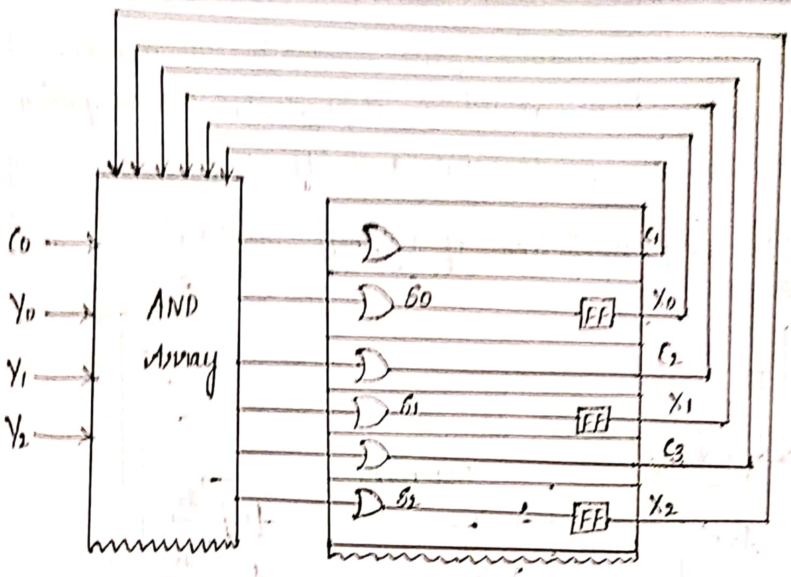


Figure shows how the 4-bit loadable right-shift register of can be implemented using four macro-cells of a CPLD. The four OR-gate outputs implement the D inputs. A total of 12 product terms are required. The Q output are fed back to the AND array via the interconnection matrix.

(3) CPLD implementation of a parallel adder with accumulator

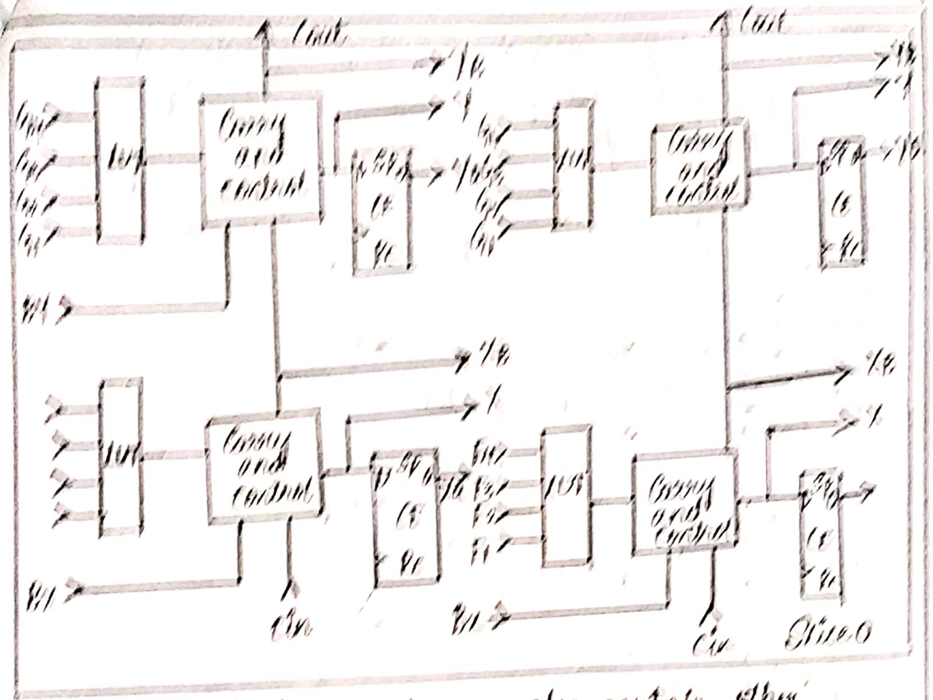
Figure shows how three bits of the parallel adder with accumulator can be implemented.



Using a CPLD. Each bit of the adder requires two macrocell. One of the macrocells implements the sum function and an accumulator flip-flop. The other macrocell implements the carry. which is fed back into the AND array. The Ad signal can be connected to the CE input of each flip-flop via an AND gate (not shown). Each bit of adder requires eight product terms (four for the sum, three for the carry, and one for CE). If the flip-flops are programmed as T flip-flops then the logic for the sum can be simplified. For each accumulator flip-flop.

Sequential circuit design using FPGA's.

An FPGA usually consists of an array of configurable logic blocks (CLBs) surrounded by a ring.



of I/O blocks. The FPGA's may also contain other components such as memory blocks, clock generators, tri-state buffers, etc. A typical CLB contains two or more function generators, often referred to as look-up table or LUTs programmable multiplexers, and D-CE flip-flop. The I/O blocks usually contain additional flip-flop for storing inputs or outputs and tri-state buffers for driving the I/O pins.

Figure shows a simplified block diagram for a Xilinx Virtex or Spartan II CLB. This CLB is divided into two nearly identical slices. Each slice contains two 4-variable function generators (LUTs), two D-CE flip flop and additional logic for carry and control. The additional logic includes MUXes for selecting the flip-flop inputs and for multiplexing the LUT output to form functions of five or more variables.

FPGA implementation of a Mealy Machine

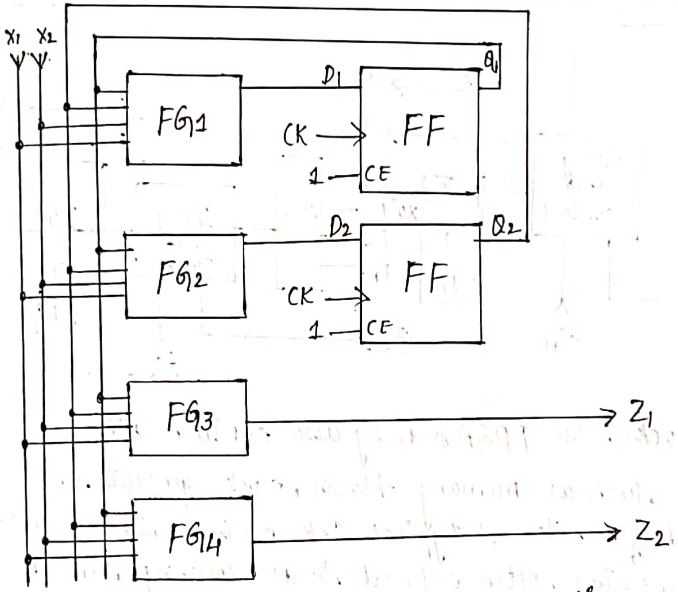


Figure shows how a Mealy sequential machine with two inputs, two outputs and two flip-flop can be implemented by a FPGA. Four LUT's (FGs or function generators) are required, two to generate the D inputs to the flip-flop and two to generate the Z outputs. The flip-flop outputs are fed back to the CLB inputs via interconnections external to the CLB. The entire circuit fits into one virtex CLB. This implementation works because each D and Z is a function of only four variable (X_1, X_2, Q_1 and Q_2). If more flip-flop or input are needed, the 6 & 7 function may have to be decomposed to use additional function generator.

FPGA Implementation of a shift register

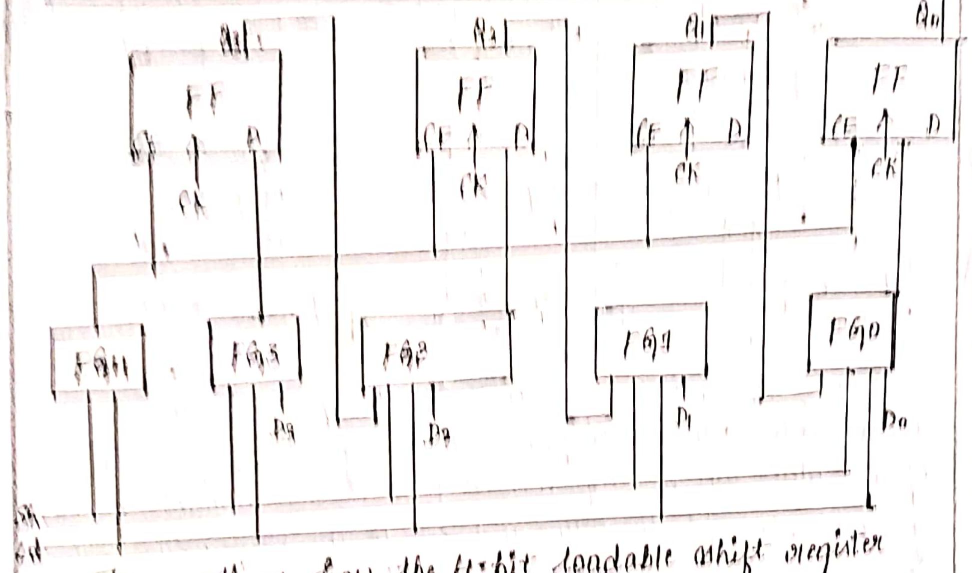


Figure shows how the 4-bit loadable shift register can be implemented using an FPGA. We would need to implement four 5- variable functions. This would require eight LUTs because each 5- variable function requires two - 4 - variable function generators. However, if we set CE = Ld + Sh, then CE = 0 when Ld = Sh = 0 and the flip - flop hold their current values.

FPGA implementation of a parallel adder with accumulator

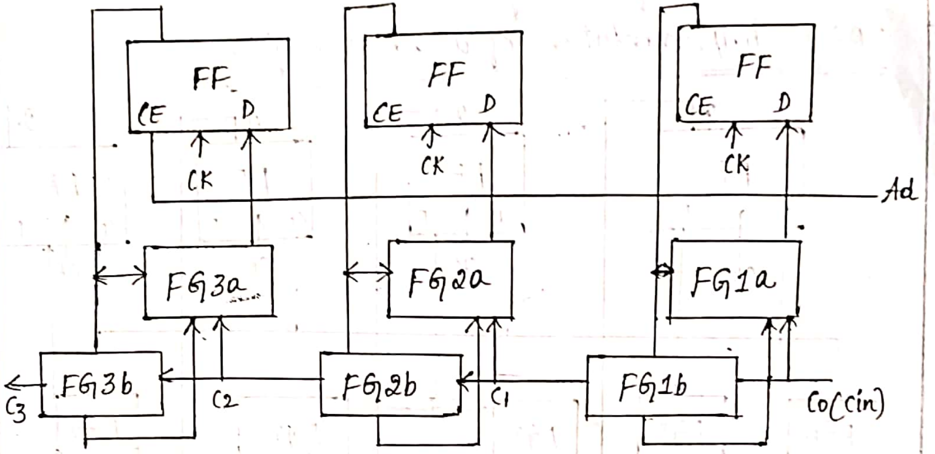


Figure shows how three bits of the parallel adder with accumulator can be implemented using an FPGA. Each bit of the adder can be implemented with two 3-variables function generators, one for the sum and one for the sum. The Ad signal can be connected to the CE input of each flip-flop so that the sum is loaded by the rising clock edge when Ad = 1. The arrangement for generating the carries, show in figure is rather show because the carry signal must propagate through a function generator and its external interconnection for each bit. Because adders are frequently used in FPGAs. most FPGAs, have bit-in fast carry logic in addition to the function generators. If the fast carry logic is used, the bottom row of function generator in figure is not needed, and a parallel adder with an accumulator can be implemented using only one function generator for each bit.