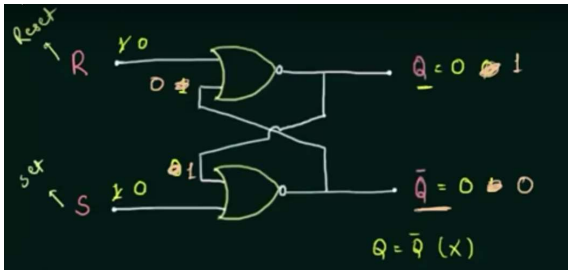
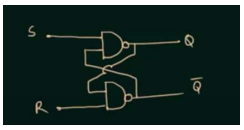
ESE LOGIC DESIGN

13 April 2024 01:32 AM

SR LATCH

Level sensitive



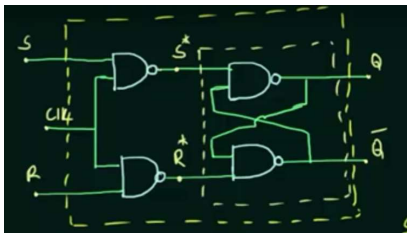


S	R	Q Q̄
0	0	Not used
0	1	1 0
1	0	0 1
1	1	Memory

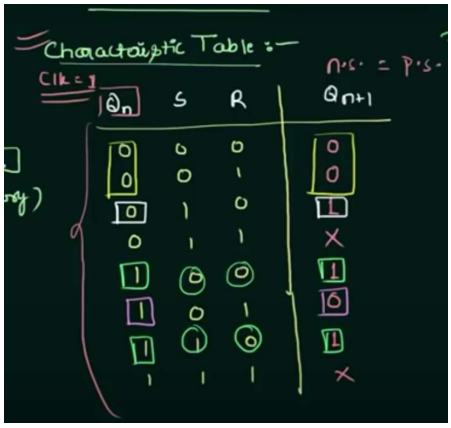
S	R	Q Q̄
0	0	Memory (as before)
0	1	0 1
1	0	1 0
1	1	Not used

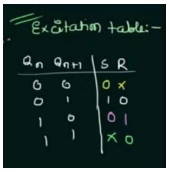
FLIP FLOP

Edge sensitive



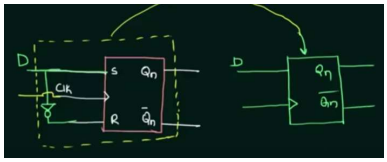
Clk	S	R	Q Q̄
0	X	X	Memory
1	0	0	Memory
1	0	1	0 1
1	1	0	1 0
1	1	1	Not used





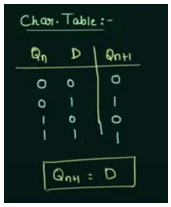
$Q_{n+1} = S + Q_n\bar{R}$

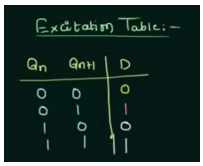
D FLIP FLOP



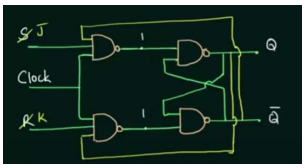
Clk	D	Qn+1
0	X	Qn
1	0	0
1	1	1

Only to store data we use this FF

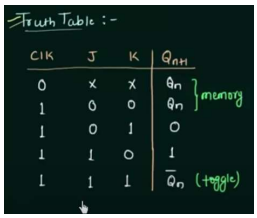


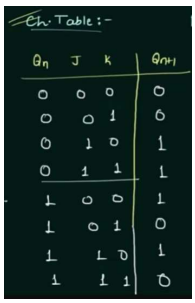


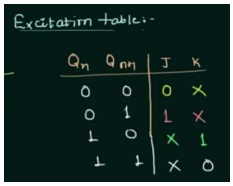
JK FLIP FLOP

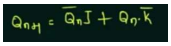


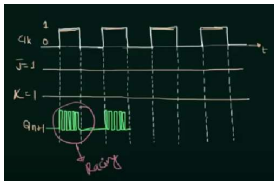
Start initially with Q=0







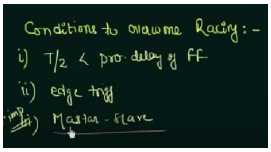




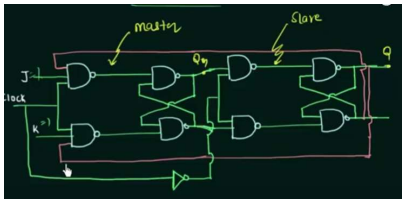
Unstable
diff than toggle

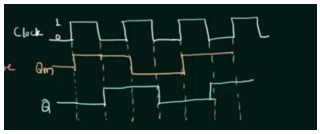
Delay > T/2

Toggling is controlled
But



Master slave is same as neg edge trig

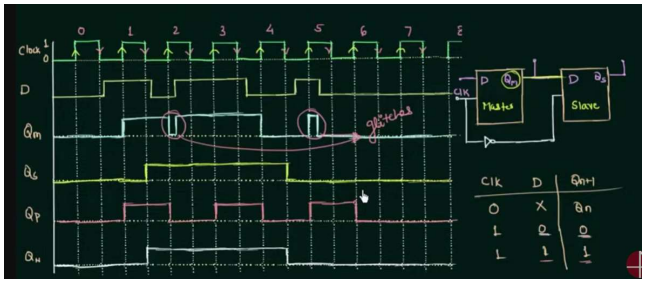




master

slave

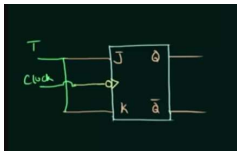
Glitches = sudden change in the output signal



There are no glitches in slave that why we use master slave FF
Master slave output is same as the neg edge output

Master slave work on level triggering
Which has the same effect as neg edge triggering

T Flip Flop - Toggle



T-T for T FF –

Clk	T	Qn+1
0	X	Qn (memory)
1	0	Qn (memory)
1	1	Q̄n (toggling)

Ch. Table:-

Qn	T	Qn+1
0	0	0
0	1	1
1	0	1
1	1	0

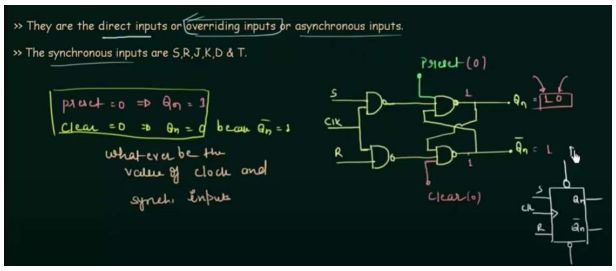
Excitation Table:-

Qn	Qn+1	T
0	0	0
0	1	1
1	0	1
1	1	0

T FF is neg edge triggered

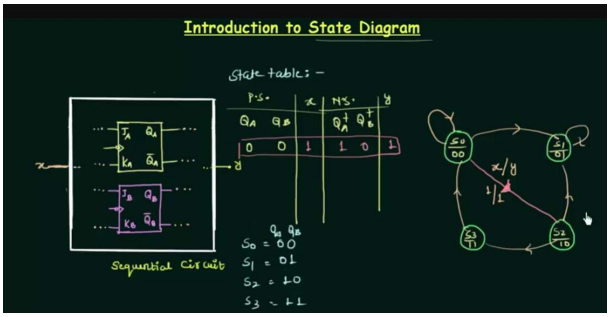
$Q_{n+1} = Q_n \oplus T$

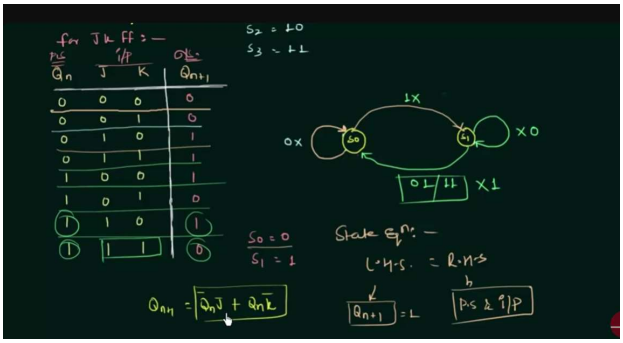
PRESET AND RESET



Preset	Clear	Qn
0	0	Not used
0	1	1
1	0	0
1	1	FF will perform normally

Asynchronous inputs = set rest
Synchronous inputs = s r j k d t





$Q_{n+1} = \bar{Q}_n J + Q_n \bar{K}$

Design Procedure for Clocked Sequential Circuits

Step 1: A State diagram or timing diagram is given, which describes the behaviour of the circuit that is to be designed.

Step 2: Obtain the state table.

Step 3: The number of states can be reduced by state reduction method.

Step 4: Do state assignment. (If required)

Step 5: Determine the number of flip-flops required and assign letter symbols.

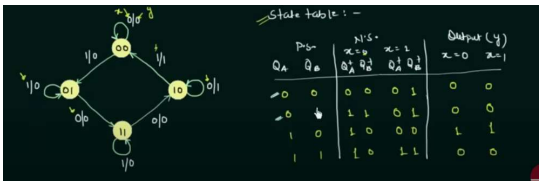
Step 6: Decide the type of flip-flop to be used.

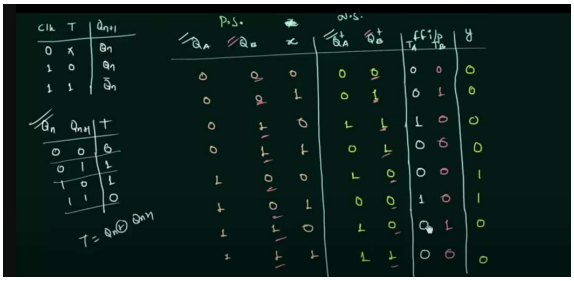
Step 7: Derive the circuit excitation table from state table.

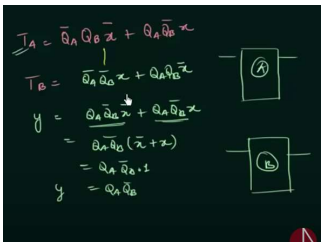
Step 8: Obtain the expression for circuit output and flip flop input.

Step 9: Implement the circuit.

Next stage as well as the output must be same for two states for them to be reduced

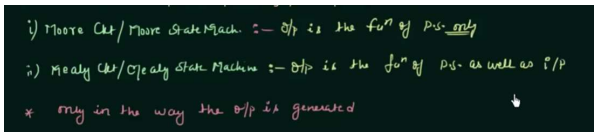


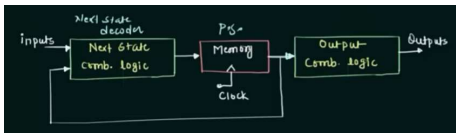




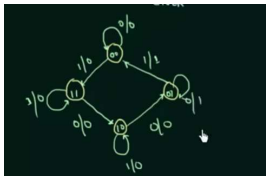
MEALY AND MOORE CIRCUITS

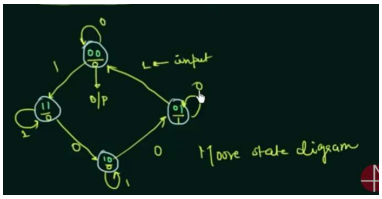
MOORE = Doesn't depend upon the input
Diff is the way the output is calc





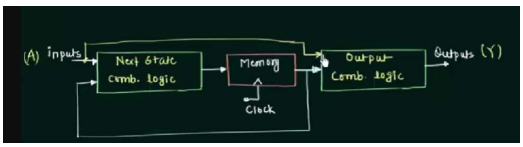
moore





Conversion to a state diagram which follows moore prop

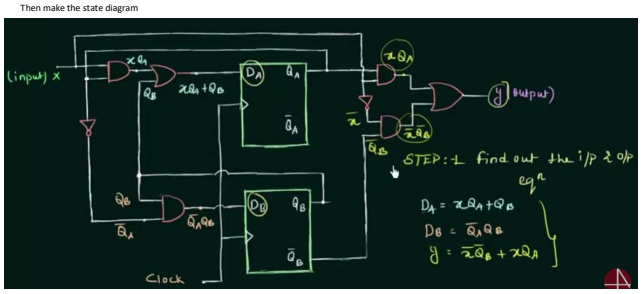
MEALY



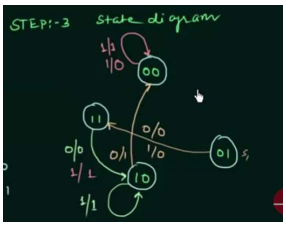
If same logic is applied

ANALYSIS OF CLOCKED SEQUENTIAL CIRCUITS

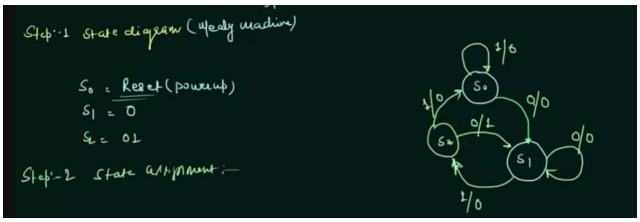
Write the equations for the inputs and outputs
Then write the state table
Then make the state diagram

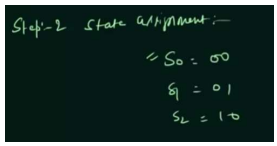


Q_A	Q_B	x	Q_A^+	Q_B^+	y
0	0	0	0	0	1
0	0	1	0	0	0
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	1
1	1	0	1	0	0
1	1	1	1	0	1

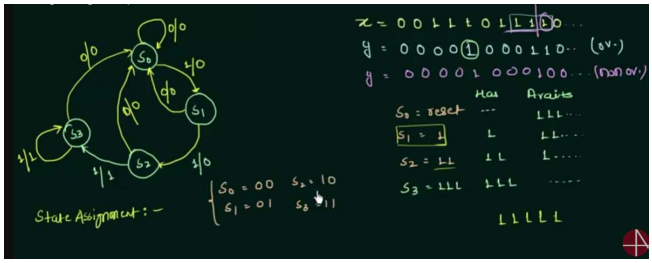


PATTERN DETECTOR

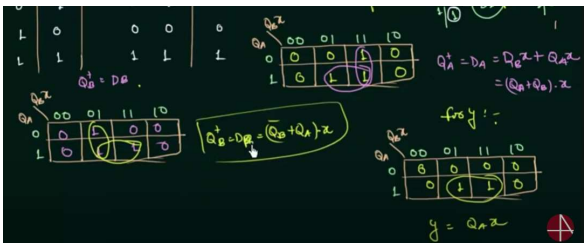


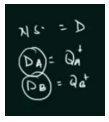


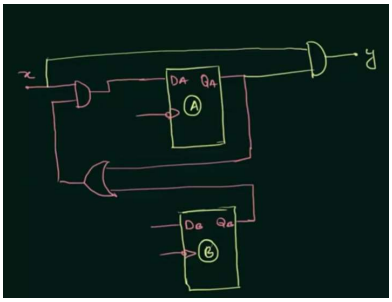
Considering OV



Q_A	Q_B	x	Q_A^+	Q_B^+	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	1	1	1

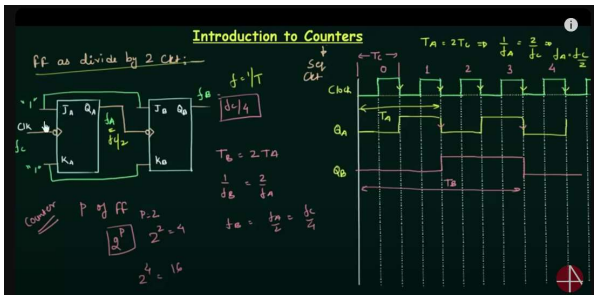


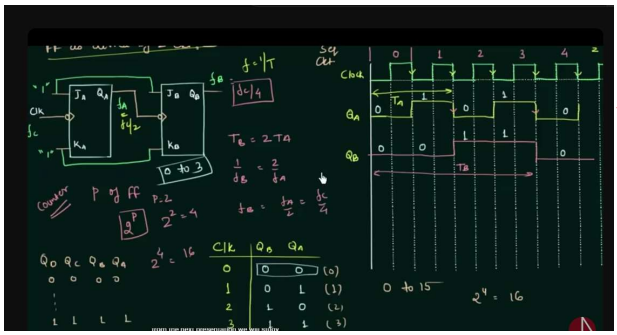




Synchronous Seq. Ckt.	Asynchronous Seq. Ckt.
1. These circuits are easy to design.	1. These circuits are difficult to design.
2. A clocked flip flop acts as memory element.	2. An unclocked flip flop or time delay element is used as memory element
3. they are Slower.	3. Faster as clock is not present
4. The status of memory element is affected only at the active edge of clock, if input is changed.	4. The status of memory element will change any time as soon as input is changed

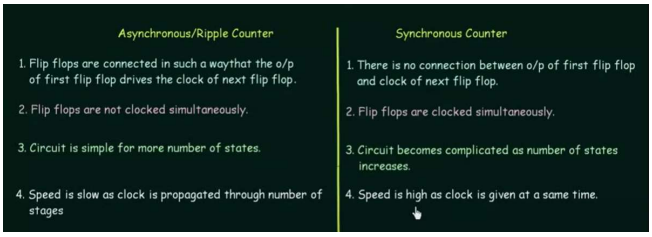
COUNTERS

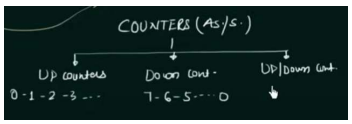




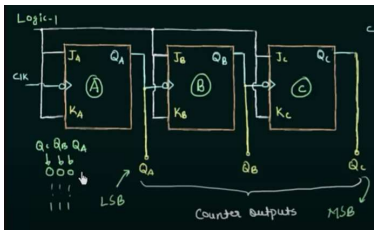
LSB

MSB



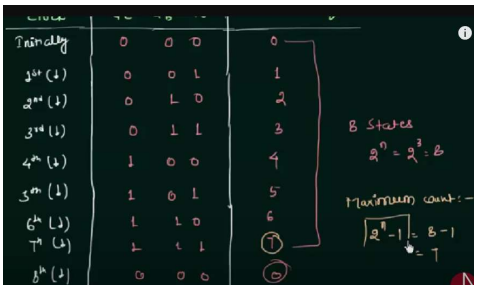


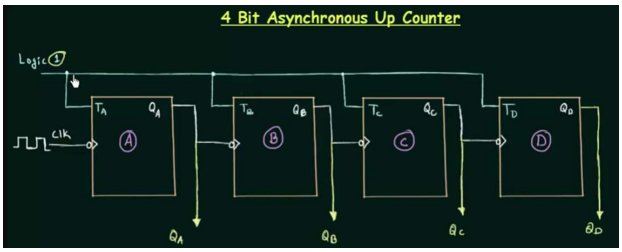
3 BIT ASYNC COUNTER



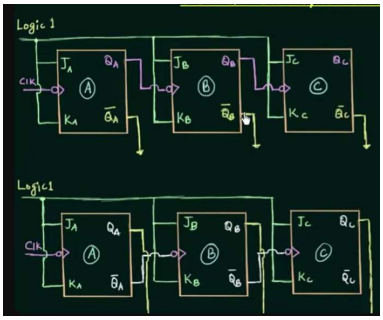
N bit = n FF

No of states = 2^n

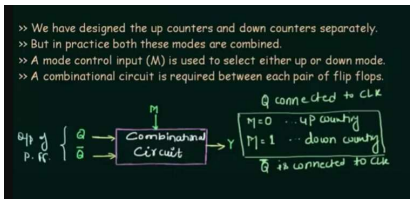


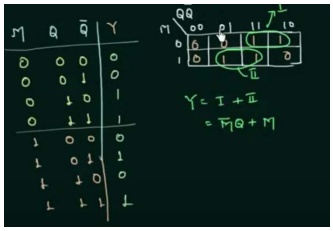


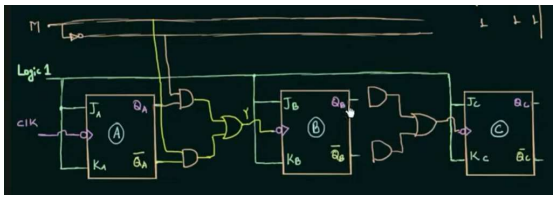
3 BIT DOWN COUNTER



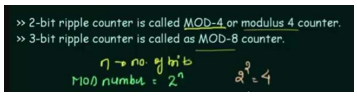
UPDOWN COUNTER



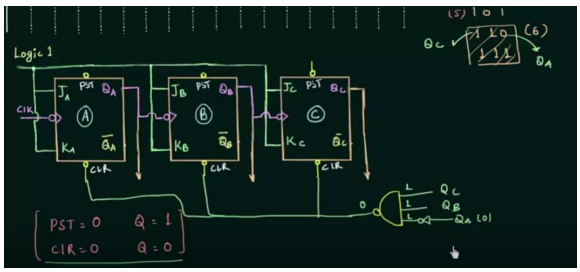


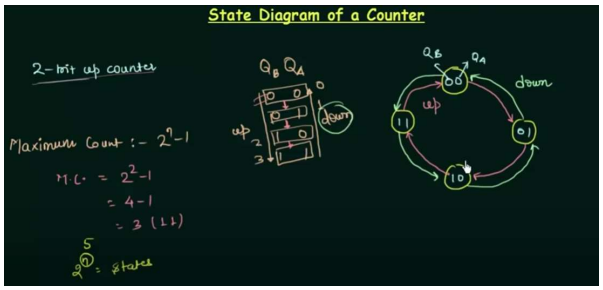


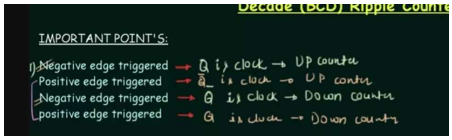
MODULUS COUNTER

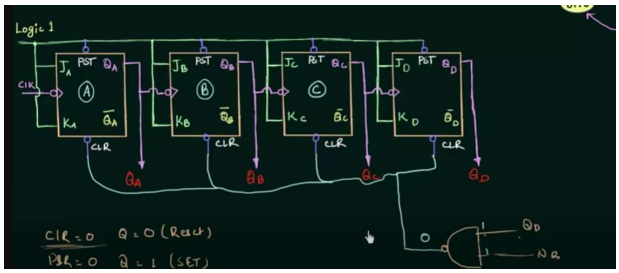


Modulus of counter = number of states

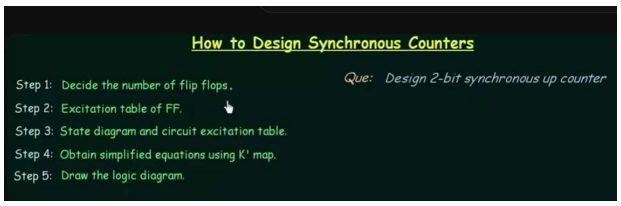


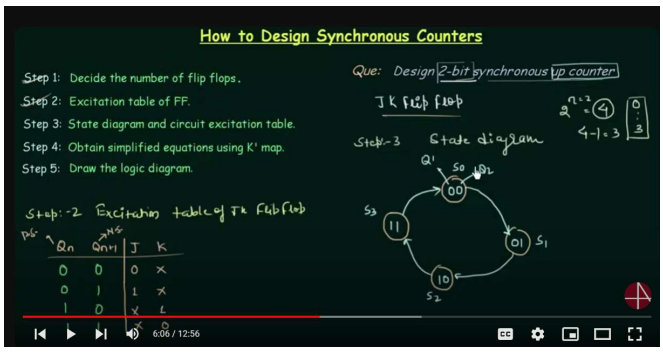


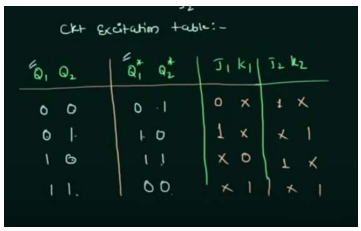


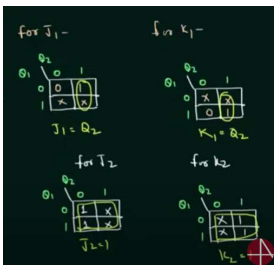


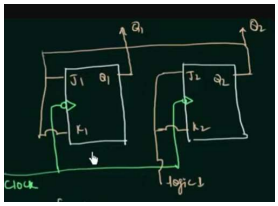
Mod 10 counter / BCD coutner

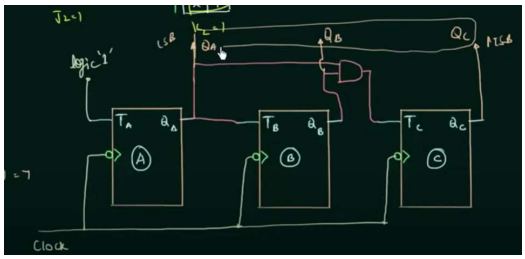




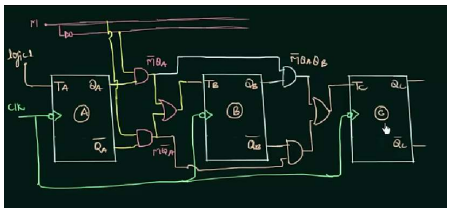




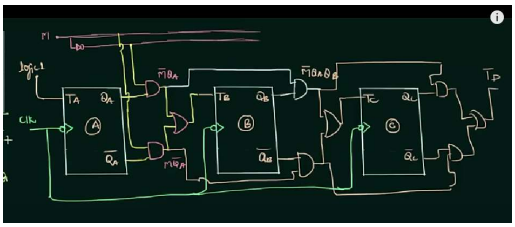




3 bit sync counter



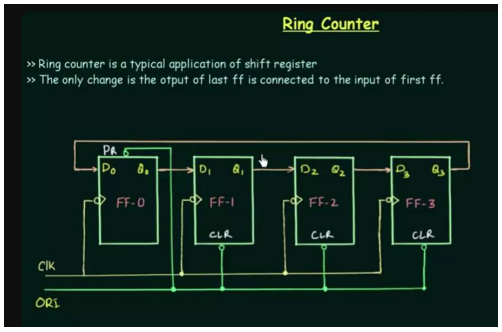
3 bit updown sync counter

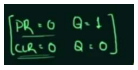


4 bit updown counter

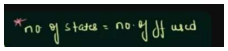
RING COUNTER

Difference from register = output of the last FF is connected to the input of the first FF



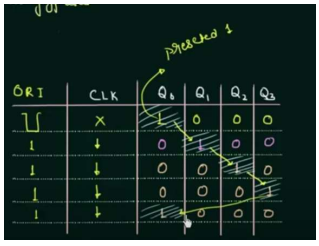


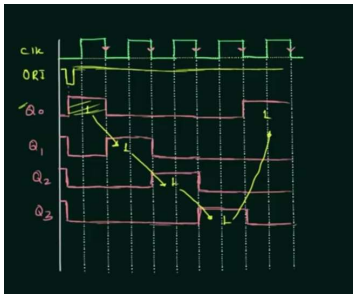
ORI	CLK	Q_0	Q_1	Q_2	Q_3
1	↓	1	0	0	0
1	↓	0	1	0	0
1	↓	0	0	1	0
1	↓	0	0	0	1
1	↓	1	0	0	0



ORI = overidding inputs

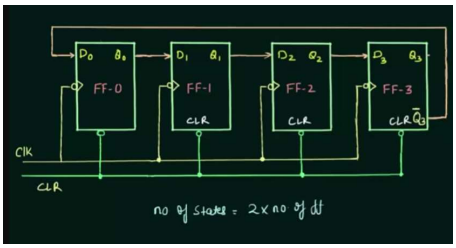
Number of states = n

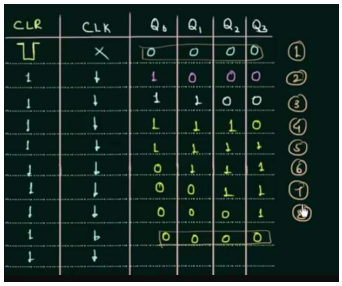




Johnson Counter

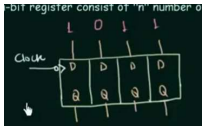
Better than the ring counter due to more number of states
Number of states = 2n



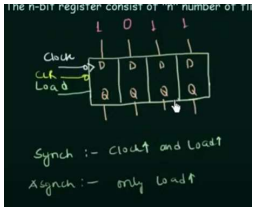


REGISTER

>> Flip Flop is 1-bit memory cell.
>> To increase the storage capacity, we have to use group of flip-flop. This group of ff is known as REGISTER.
>> The n-bit register consist of "n" number of flip-flops and is capable of storing "n-bit" word.



Clock is applied to all directly
This can be a problem as the data stored will get changed as clock pulse changes

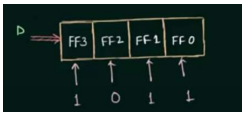


Hence load are applied

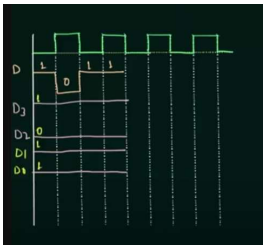
Load = independent control

>> Data can be entered in serial or in parallel form.
one bit at a time
all bits at a time

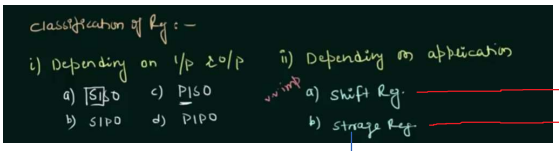
serial



parallel



Serail = temporal code
Parallel = special code



Shifts the data

Stores the data

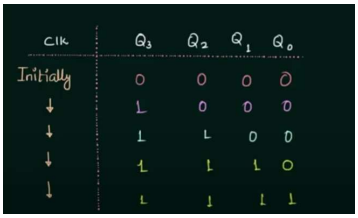
PIPO

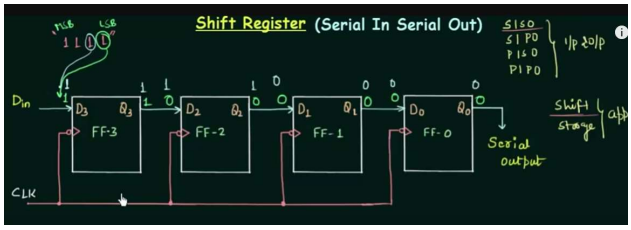
Serial IN and Serial OUT

On input = 1111
The clock stores on rising edge

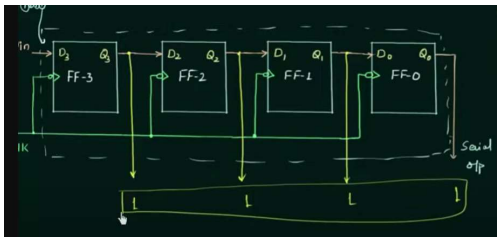
Beneficial for long distance transmission
Only one input line is required

Number of clock pulses required to store data = 4
But disadv is that we need 4 pulses to output the data as well



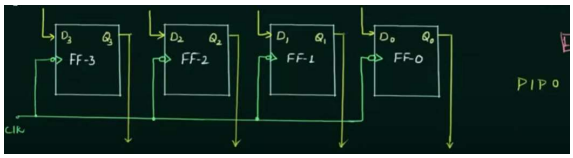


Serial IN and Parallel OUT



4 clock pulses to store the data
1 clock pulse to output the data

Parallel IN and Parallel OUT

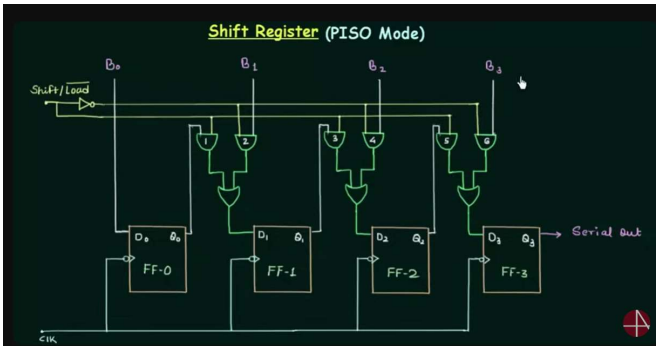


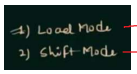
Input the number before clocks falling edge and then make clock =0
This stores the data
Output is given parallely

1 clock pulse to store the data

Storage register / buffer regsiter

Buffer
Input in = output out

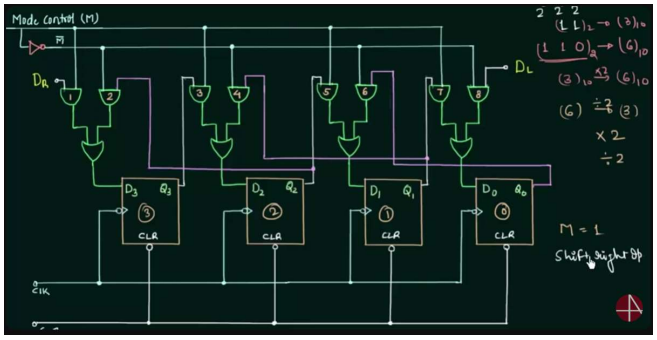


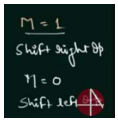


For input. Works when load =0

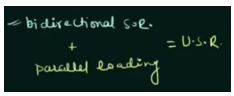
Shifts the data towards the right. Works when load =1

BIDIRECTIONAL SHIFT REGISTER

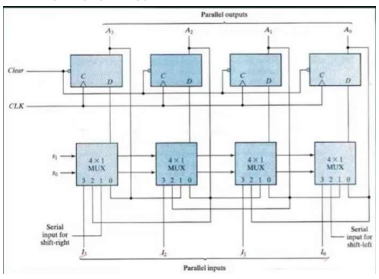


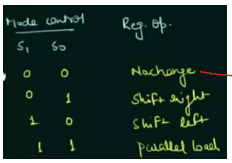


UNIVERSAL SHIFT REGISTER



All SISO SIPO PIPO PISO are possible in this , hence universal





Save data in flip flops

FSM = fininte state machine

Flip flop are edge sensitive while lathes are level sensitive

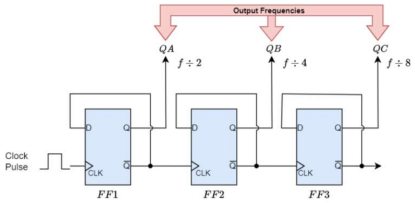
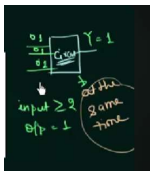
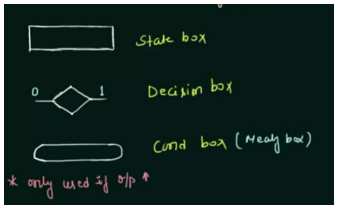


Fig-1: Freq Division Counter

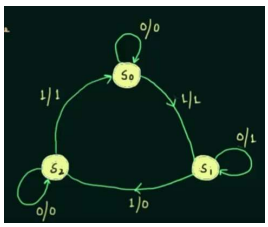


Coincidence logic

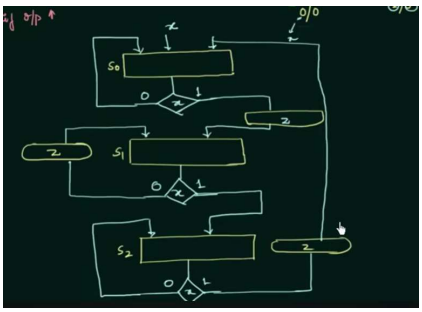
ASM CHART = Algorithmic State Machine



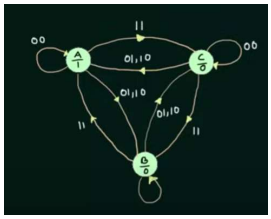
Condition box used only for mealy state machines



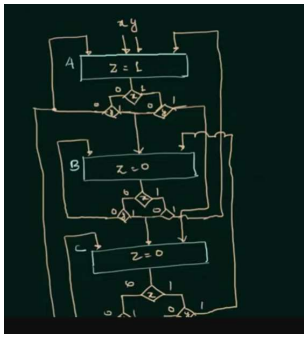
Mealy state machine



Z means that the output is 1
If output is 0 then don't do anything



Moore State Machine



PLA = Programmable Logic Array

PLD = Programmable Logic Device

PLD

It is a type of fixed architecture logic device with programmable AND gates followed by programable OR gates

A	B	C	Y_1	Y_2
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

Step:-1

$Y_1 = A\bar{B}\bar{C} + A\bar{B}C + ABC$

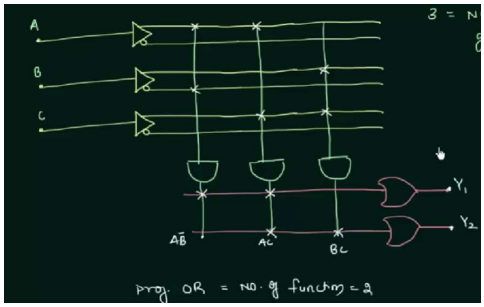
$Y_1 = A\bar{B} + AC$

$Y_2 = BC + AC$

Step:-2

Step:-1 No. of i/p buffer = number of variables

3 = No of prog. AND gate = No. of minterm (product term)



PAL = Programmable Array Logic

>> It is most commonly used type of PLD.
>> Has programmable AND array and fixed OR array.

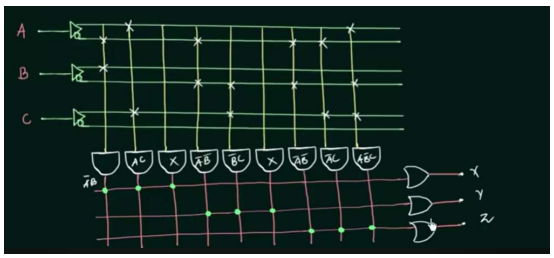
$X(ABC) = \Sigma m(2,3,5,7) = \bar{A}B + AC$

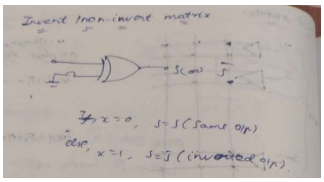
$Y(ABC) = \Sigma m(0,1,5) = \bar{A}\bar{B} + \bar{B}C$

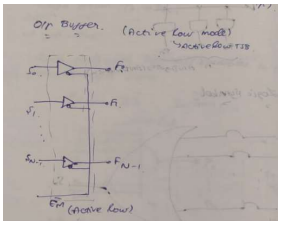
$Z(ABC) = \Sigma m(0,2,3,5) = \bar{A}B + \bar{A}\bar{C} + A\bar{B}C$

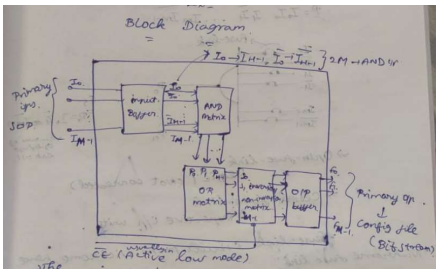
Number of AND gates per function = maximum number of minterms
Number of OR gates = number of outputs

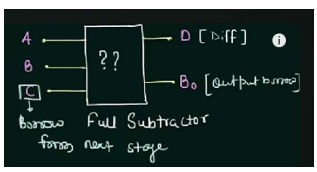
When we don't need an AND gate , simply fuse it











Full subtractor : A - B - - B - C

STEPS: -

1. Identify available and required flip flop.
2. Make characteristic table for required flip flop.
3. Make excitation table for available flip flop.
4. Write boolean expression for available ff.
5. Draw the circuit.

Steps for conversion of FFs

Ex:- JK to D Flip flop conversion.

av. ff = JK

req ff = D

Q_n	D	Q_{n+1}	J	K
0	0	0	0	x
0	1	1	1	x
1	0	0	x	1
1	1	1	x	0