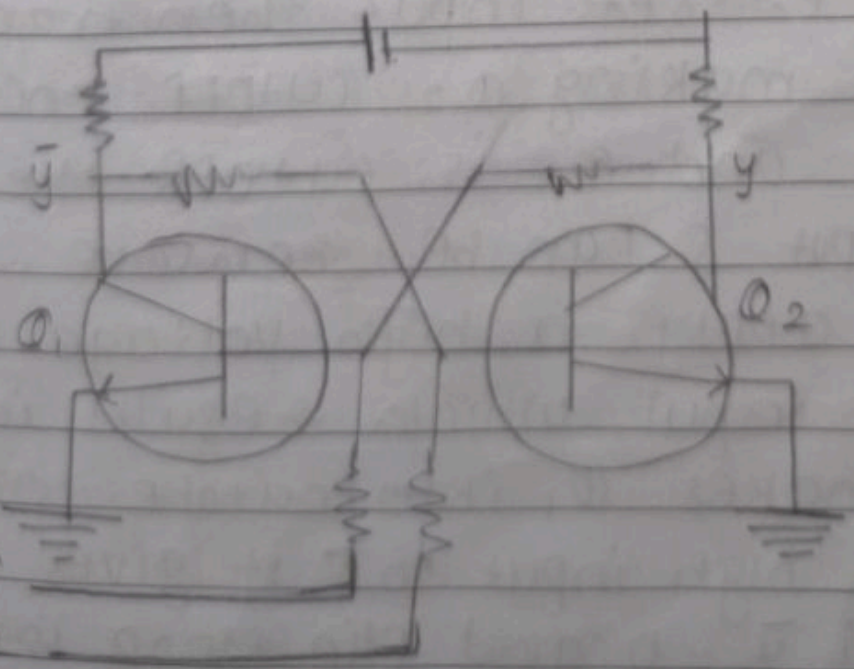
flip floop is a basic memory element used in designing of sequencial logic circuit. flip floops are also knocon as bistable multivibraters or latches or Multibinary and togle and are used in various devices.

flip floops can be fabricated by using logic gates like NAND gate, NOR Gate,

* RS flip floop

It is a one of the simplest storage device here. R Stands for Reset and S stands for set.



This RS flip floop can be form by using two NAND gates OR two NOR gates It is one of the simpliest sequential circuit element.

① The output of flip floop is low i.e. 0 or high i.e. 1 and it remains the same

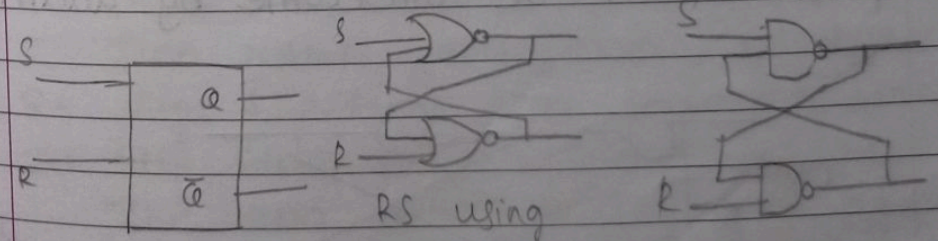
② If we wish to change the circuit have to be drive known as Triggering

③ until trigger arrives The output remains. low or high

The above figure shows. The transistor version of RS flip floop. From

From the above figure we observe Q_1 and if a high voltage is applied to S input i.e. set input then transister Q_1 Saturates making Q_2 cutoff once Q_1 is saturated and Q_2 is cutoff the trigger at the input S can be remove.. similarly we can apply a high voltage to R input i.e. reset input which results in saturating Q_2 and makes Q_1 into cutoff. when we apply the high input to S it gives at output $y = 1$ and $\bar{y} = 0$ and flip floop is said to be in set state. in the same way

when we apply high voltage to input R at output we get $y = 0$ and $\bar{y} = 1$ the 0nd the flip floop is set to be in reset state. when input S and R is 0 the both transister are in cutoff state and output y and $\bar{y}$ no change is there. and flip floop is said to be latched. In the same way when we ~~put~~ apply input $S = 1$ & $R = 1$ both transisters Q_1 & Q_2 try to saturates giving at output $y = 1$ & $\bar{y} = 1$ which is invalid and not allowed condition. following figure. (A) (B) (C) shows. the logical symbol, representation of R, S flip floop using NOR gate and using NAND gate. The working of this RS flip floop can be summarized by using following truth table.



Logical symbol For RS Flip floop (A)

RS using NOR (B)

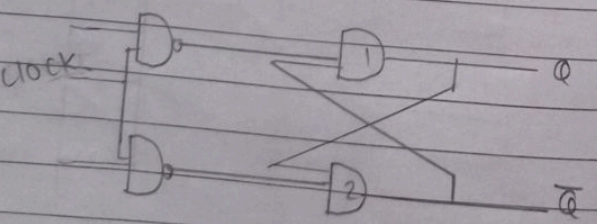
RS using NAND (C)

INPUT		output
S	R	
0	0	Latch
0	1	RSet
1	0	~~Reset~~ Set
1	1	Not allowed/ invalid

* Clock RS flip floop

As we have discussed the basic R-S flipfloop does not operate in step with clock or timing device.

When an input is applied the output changes accordingly. either it is set or Reset depending on input applied. i.e the circuit is going to work as a combination logic circuit. This problem can be overcome by adding



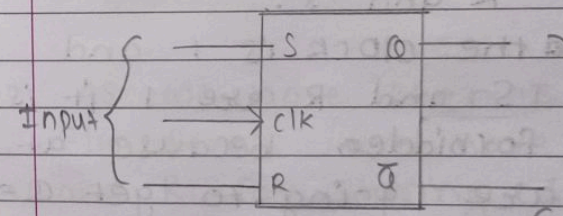
(A)

clock to Basic R.S flip floop. for adding clock to basic RS flip floop we have to use two more additional NAND Gate / AND gates. and providing a clock input.

The clock RS flip floop operates in step with clock or timing device

In otherwords we can say that the device operate singronsly.

The above. figure shows the logic symbol use for clock RS flip floop.



(Logical diagram)

The (A) figure shows realization of clock RS flip floop.

Two NAND gates are added to RS ~~floo~~ flip floop to form a clocked RS flip floop.

This two additional NAND gates add clock feature to the Basic RS flip floop

In this circuit if clockplus is present the function of circuit is exactly same as Basic RS flip floop i.e. when clock is 0 both additional NAND gate are disable and the output is Latch i.e. There is no change at output If we make clock input 1 both the NAND gates are Enable and R and S input can Reach to the RS flip floop. In this case flip floop can be set or Reset depending on the value of R and S.

Now if the clock is 1 and both input S and R are 1 it is invalid or forbidden because at output we are going to get $Q=1$ and $\bar{Q}=1$ which is Not allowed.

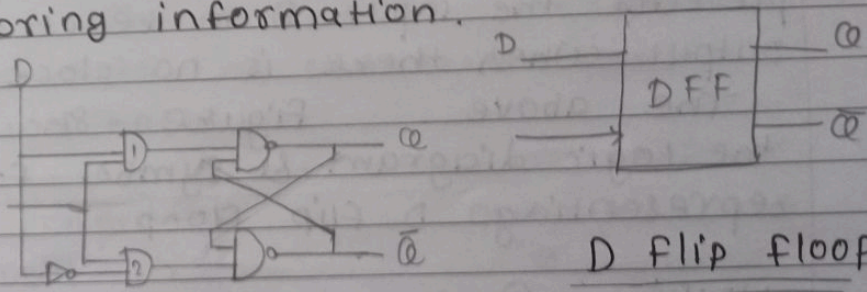
We can summaries the working of clock RS flip floop as shown in truth table below.

INPUT			output
CLK	S	R	
0	0	0	→ No change / Latch
1	1	0	→ Set
1	0	1	→ Reset
1	1	1	→ Not allowed

* D flip floop

D flip floop is Nothing but it is a modifide version of clock RS flip floop

The letter D stands for Delay and such a flip floops are used for storing information.



D flip floop

AS we have seen In case of RS flip floop. we have two inputs S and R for storing 1 or high we have to Set the flip floop and for storing 0 or low we have to Reset the flip floop.

The use of two signals or inputs is some what disadvantagous. In some application and also as we have observed there is a forbidden condition in RS flip floop. In D flip floop the care is taken not to

occur a forbidden condition or instead of two inputs only one input is used to drive the flip flop.

The D flip floop allows the value of D to reach the output only when the clockplus occurs and prevents the input D to reach to output when there is no clockplus

The above figure shows the logic diagram & symbol for representing D flip floop.

Working : ① When clock is low i.e. zero both and gates are disable and D can change the value without affecting the output Q

② During the positive transition of the clockpulse the D flip floop transfers to the output only the data which is present. on the input D.

③ As shown in above fig. D input is transfer to output on occurence of clockpluse

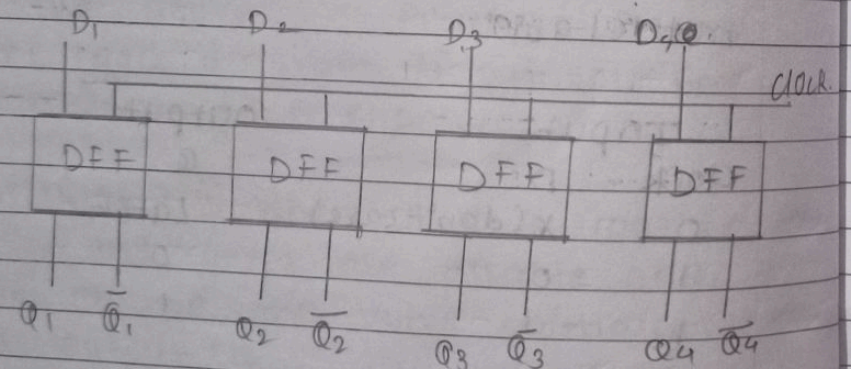
A D flip floop having clock high or 1 will transfer the data from D to output i.e. In a D flip floop the output takes the input as long as clock is high and this type of D flip floop is called as D latch

The action of D Latch can be summaries as shown in following truth table.

Input		output
clock	D	Q
0	X (don't care)	Latch
1	0	0
1	1	1

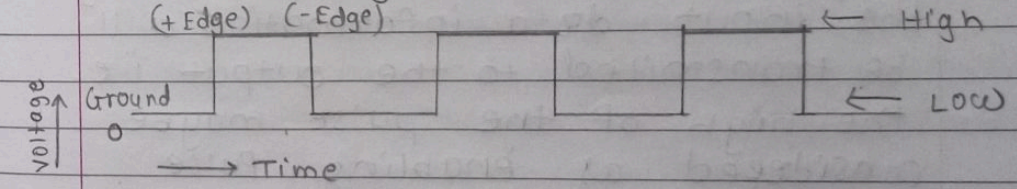
From the above truth table we can say that when the value of clock is zero then the value of D is in inmaterial so that it is marked as don't care and the forbidden state of RS flip floop is not going to occur as we have only one input signal. The operation is such that Q will be the same as D while

clock is high and Q is latch when clock is zero. that's why storing a word or data or information of n bit we require n D flip flop ex: if we want to store a 4 bit data or information we requires 4 D flipfloops. as shown in fig. below.



* Triggering of Flip floops.

Most of the complicated degital circuit operates as a sinkronus sequencial circuits. i.e. a master clockpuls is sent to all parts of the circuits to co-ordinate operations of the system. A typical clockpulse is as shown in figure below.

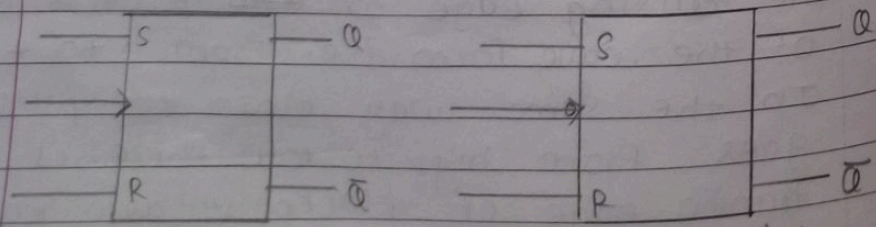


As shown in fig along x axis we have time and ~~alog~~ along y axis it is voltage. these clockpulse have mini. voltage 0(zero) and maxi. voltage +5 At the left of the wave form voltage at 1st is low and when it goes from ~~high~~ low to ~~low~~ high. the pulse is having the rising edge or ~~and~~ positive edge of the wave form i.e. from 0 to +5 In the same way when the pulse goes from high to low these is a traning edge of clockpulse also knowns

as negative edge of clockpulse.

While discussing clock RS flip flop It has been shown that, the clock input triggers the flip flop i.e. enable the flip flop when the clock goes high and flip flop is said to be level triggered device any time the clockpulse is high the input data information will be transmited to the output i.e. the high of the pulse maybe considered as enabling pulse.

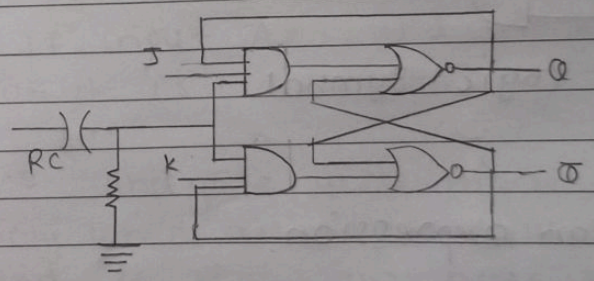
In some cases the input data is transpered only at the occurance of the change which could be from 0 to 5 i.e. at positive edge or leading edge when the device changes its state on occurance of following edge or traning edge it is also known as negative edge trigger



positive logic diagram negative

* J-K flip floop.

As we have seen in case of RS flip floop when both input R and S = 1 we could not get the proper output. as it is invalid and forbidden condition J-K flip floop is an improved version of RS flip floop in this flip floop when both inputs J-K = 1 on occurence of clockpulse at output we get compliement of last state. following fig. shows the +ve edge trigged JK flip floop



J-K flip floop

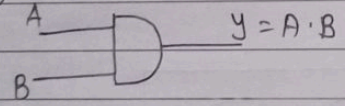
* Aim : Study of logic gates.

* Requirements.

AND gate IC 7408	power supply
OR gate IC 7432	
NOT gate IC 7404	
NOR gate IC 7402	connecting wires etc.
XOR gate IC 7486	
NAND gate IC 7400	

* Theory :

AND

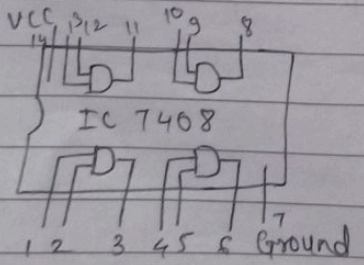


logic Symbol

$Y = A \cdot B$

Boolean expression.

IC :



Truth table :

A	B	$Y = A \cdot B$
0	0	0
0	1	0
1	0	0
0	0	1

* Working : ① If input A = 0 and input B = 0 then output is also 0

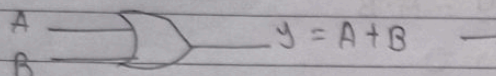
② If input A = 0 and input B = 1 then output is 0

③ If input A = 1 and input B = 0 then output is 0

④ If input A = 1 and input B = 1 then output is 1

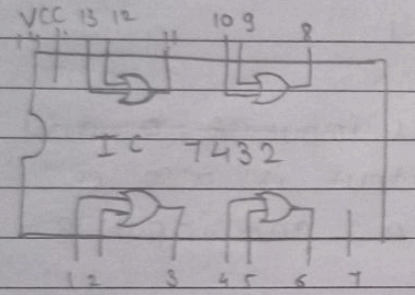
i.e. And gate gives high ~~input~~ output only when all of the inputs are high and it give low output only when all of the input or any ~~or~~ one of the input is low.

② OR gate.



→ logic symbol

Y = A + B → Boolean expression



→ IC diagram.

* Truth table

A	B	Y = A+B
0	0	0
0	1	1
1	0	1
1	1	1

* ~~Theory~~ Working

① If input A = 0 and input B = 0 then output Y = 0

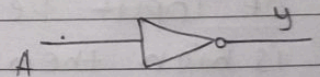
② If input A = 0 and input B = 1 then output Y = 1

③ If input A = 1 and input B = 0 then output Y = 1

④ If input A = 1 and input B = 1 then output Y = 1

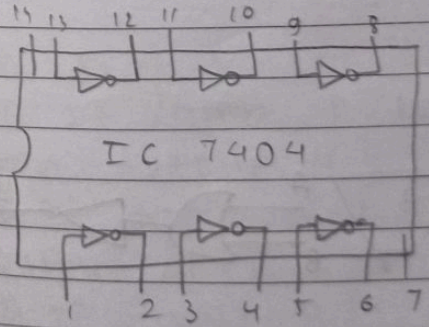
i.e. OR gate gives low output only when all inputs are low and it give high output only when all of the inputs are high or any one input is high.

③ NOT gate.



→ logic symbol

$Y = \bar{A}$ → Boolean expression



→ IC diagram

* Truth table.

A	$y = \bar{A}$
0	1
1	0

* working :

① ~~If A = 0 the~~

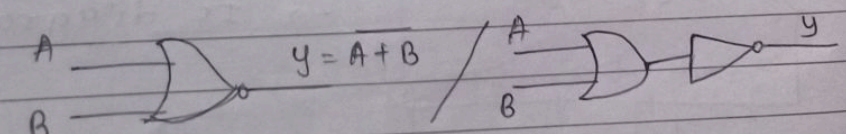
① If input A = 0 then output y = 1

② If input A = 1 then output y = 0

i.e. ② If input is low then output is the compliment of input i.e. high and If ~~out~~ input is high then output is low.

④ NOR gate.

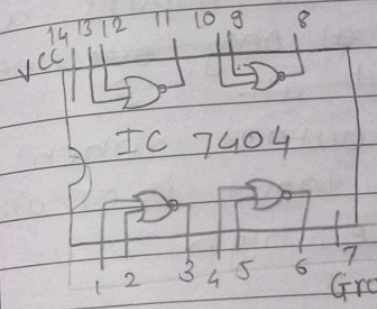
* Logic ~~di~~ Symbol



* Boolean expression

$$y = \overline{A + B}$$

* IC diagram.



* Truth table :

A	B	$y = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

* working :

① If input A = 0 and input B = 0 then output y = 1

② If input A = 0 and B = 1 then output y = 0

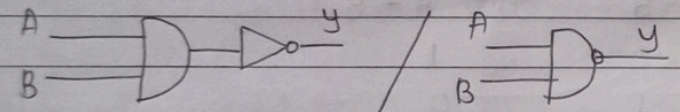
③ If input A = 1 and B = 0 then output y = 0

④ If input A = 1 and B = 1 then output y = 0

i.e. NOR gate gives high output only when ~~both~~ all of the inputs are low. ~~and If so el only out~~ one ~~input is low and all~~ and it gives low output when any one of the input ~~is~~ or all of the inputs are high.

⑤ NAND gate

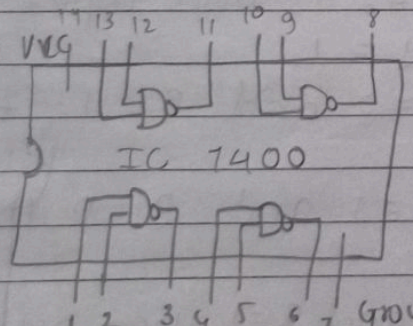
* logic Symbol



* Boolean expression

$y = \overline{A \cdot B}$

* IC diagram.



* truth table

A	B	$y = A \cdot B$
0	0	1
0	1	1
1	0	1
1	1	0

* Working:

① If input A = 0 and input B = 0 then output y = 1

② If input A = 0 and input B = 1 then output y = 1

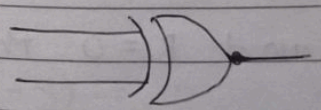
③ If input A = 1 and input B = 0 then output y = 1

④ If input A = 1 and input B = 1 then output y = 0

i.e. NAND gate gives ~~high~~ low output only when all of the inputs are high and it give high output when all of the inputs ~~are to~~ or only one input is low.

⑥ XOR gate

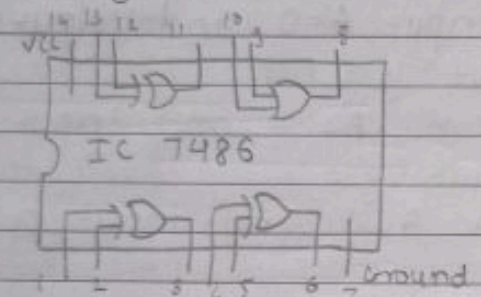
* logic Symbol



* Boolean expression

$y = A \oplus B$

* IC diagram



* Truth table

A	B	y = A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

* Working :

① If input A = 0 and B = 0 then output y = 0

② If A = 0 and B = 1 then output y = 1

③ If input A = 1 and B = 0 then output y = 1

④ If input A = 1 and B = 1 then output y = 0

i.e. XOR gate gives high output when one input is high and one input is low and it gives low output when both of the inputs are same i.e. high or low.

* Aim - implimentation of boolean expression

① $(x+y)+z = x+(y+z)$

→ Req : OR gate IC
logic Gate Kit
Power Supply
Connecting wire etc.

Theory :

	L.H.S. $(x+y)+z$							
x	0	0	0	0	1	1	1	1
y	0	0	1	1	0	0	1	1
z	0	1	0	1	0	1	0	1
$(x+y)$	0	0	1	1	1	1	1	1
$(x+y)+z$	0	1	1	1	1	1	1	1 → L.H.S
$(y+z)$	0	1	1	1	0	1	1	1
$x+(y+z)$	0	1	1	1	0	1	1	1 → R.H.S.

* ii logic diagram.