Transistors

20 January 2021 12:22 PM

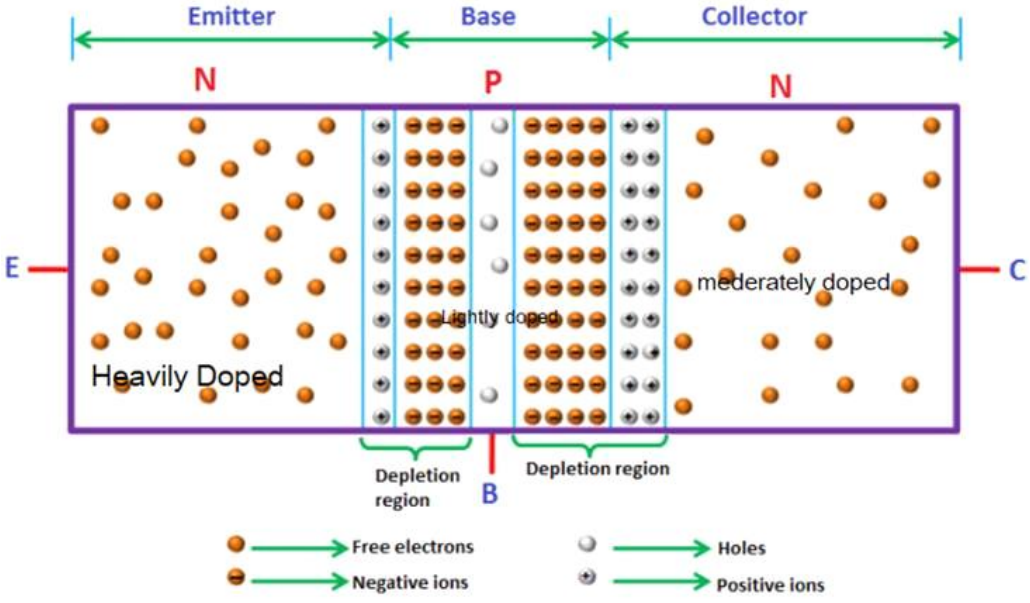
Semiconductors

P- Type- such as a Group 3 element like Boron (B) or Gallium (Ga)

n-type- atom with five valence electrons, such as the Group 5 atoms arsenic (As) or phosphorus (P)

Structure

1. In npn transistor, the left side n-region (emitter) is heavily doped. So the emitter has a large number of free electrons.
   We know that in p-type semiconductor, holes are the majority charge carriers and free electrons are the minority charge carriers.
2. The p-region (base) is lightly doped. So the base has a small number of holes.
3. The right side n-region (collector) is moderately doped. Its doping level lies between that of emitter and base.



α and β Relationship in a NPN Transistor

$$\frac{I_c}{I_e} = \alpha$$

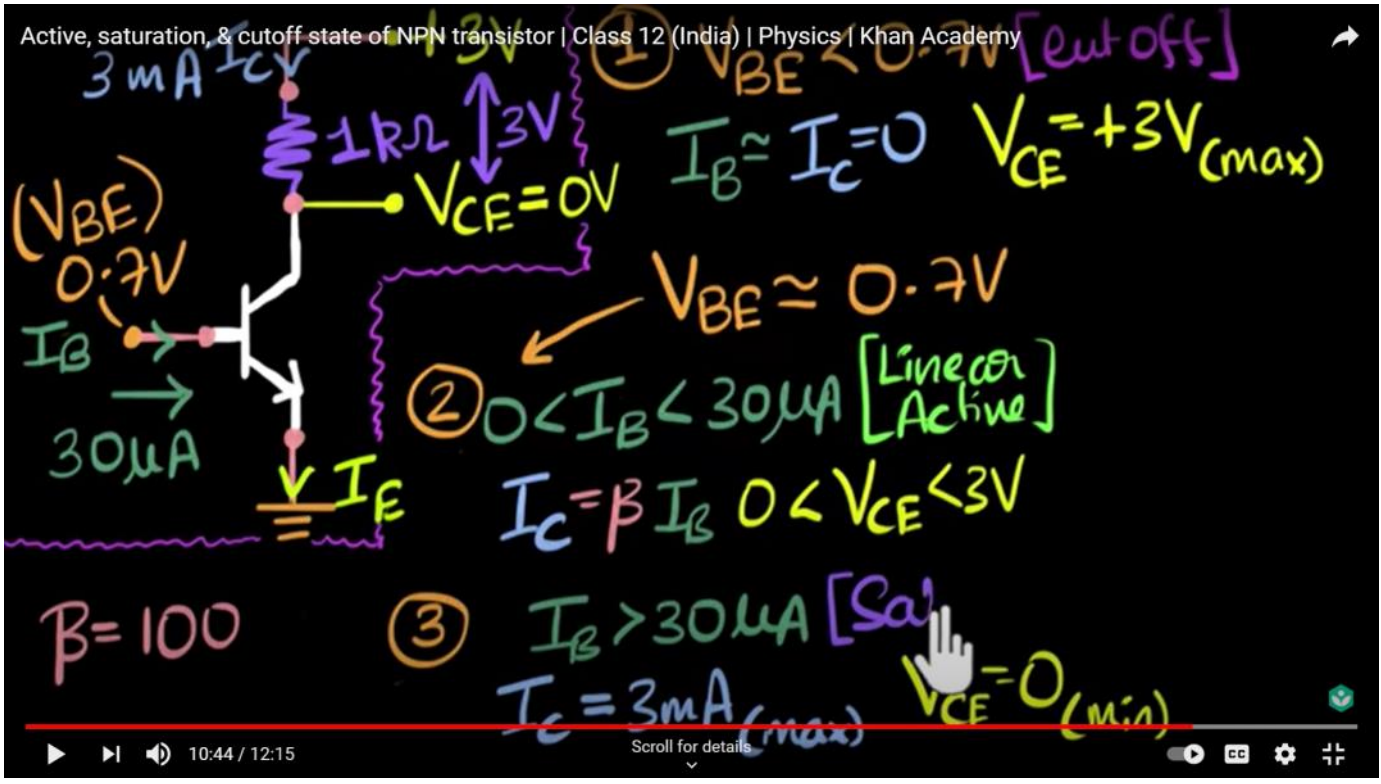
$$\frac{I_c}{I_b} = \beta$$

$$I_e = I_b + I_c$$

$$\alpha = \frac{\beta}{\beta + 1}$$

$$\beta = \frac{\alpha}{1 - \alpha}$$

States



State 1 - cut off if voltage at base is less than a particular minimum voltage the current at base is close to 0 and in tern their will be no current any where in the transistor.

$v_{BE} < \min voltage$
$v_{CE} = Max$
$I_B = I_C = 0$

State 2 - Active if voltage at base is equal or greater than a particular minimum voltage the current at base is less than some critical value and in tern their will be some current in collector as well as base, according to that current in collector I_C , $v_{CE} = V - IR$

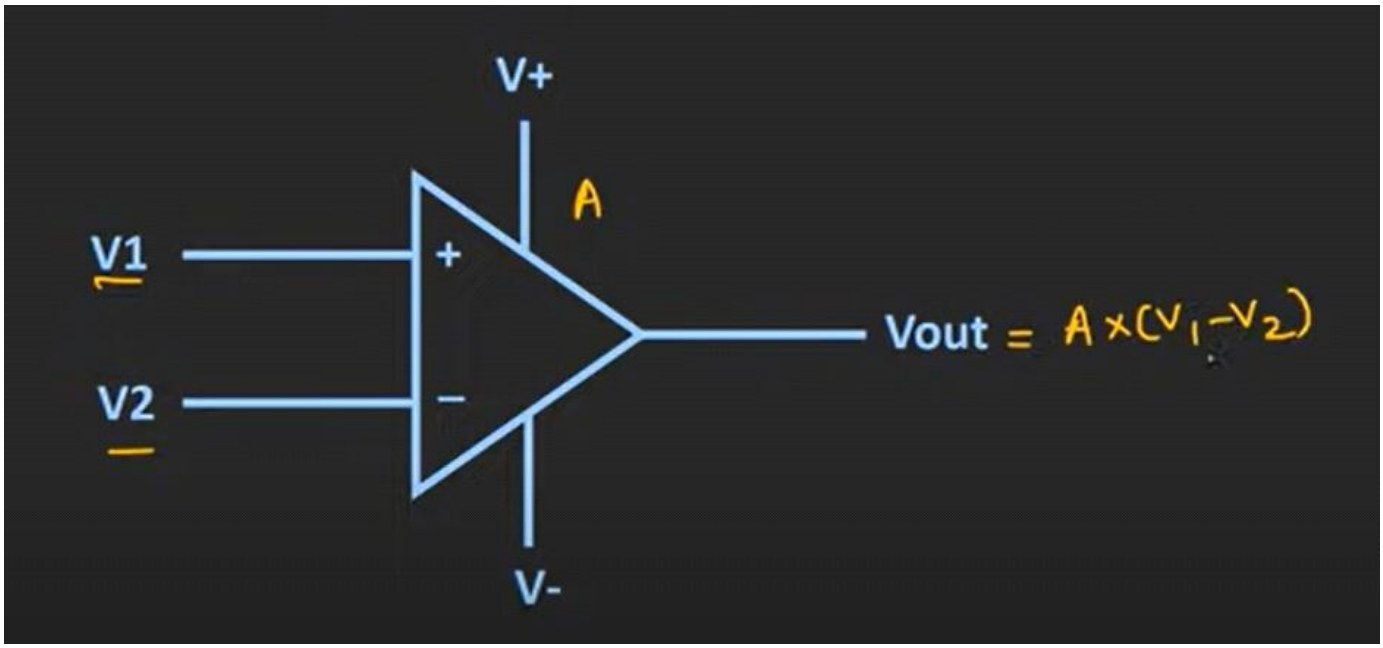
$v_{BE} \geq \min voltage \ \ (in\ practice\ it\ will\ always\ be\ close\ to \min voltage)$
$v_{CE} = V - IR$
$I_B = depends\ on\ transistor$
$I_C = \beta I_B$

State 3 - saturation if voltage at base is equal or greater than a particular minimum voltage the current at base is less than some critical value and in tern their will be some current in collector as well as base, according to that current in collector I_C , $v_{CE} = V - IR$

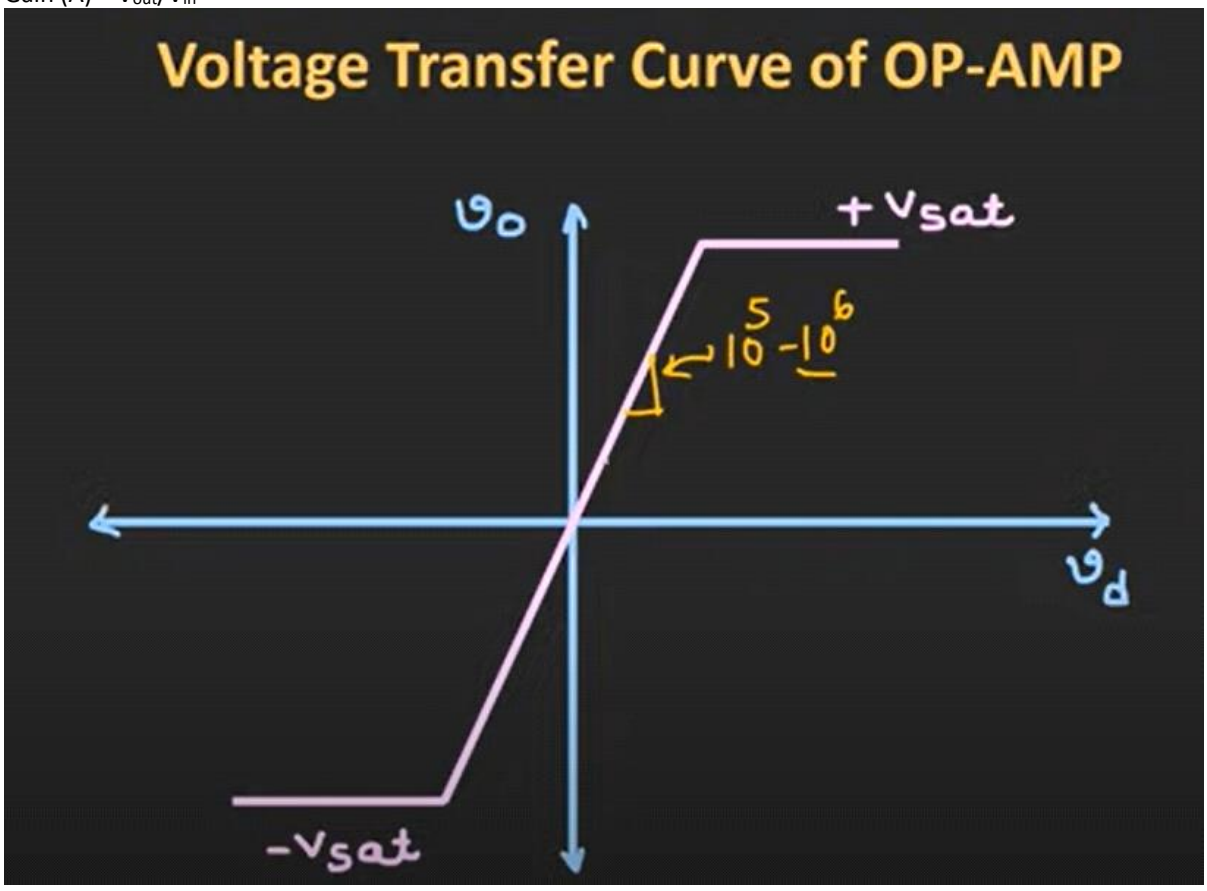
$v_{BE} \geq \min voltage \ \ (in\ practice\ it\ will\ always\ be\ close\ to \min voltage)$
$v_{CE} = 0$
$I_B = depends\ on\ transistor$
$I_C = max$

Op-Amplifiers

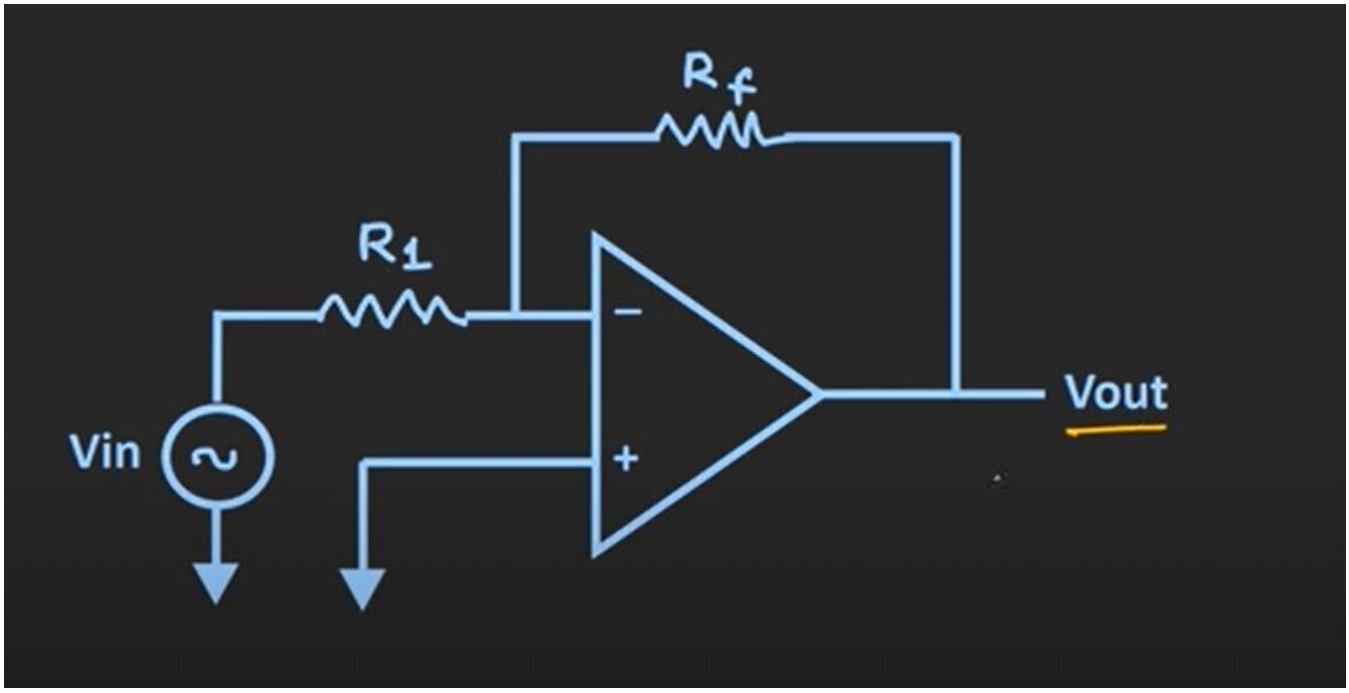
07 March 2021 08:36 AM



Gain (A) = V_{out}/V_{in}

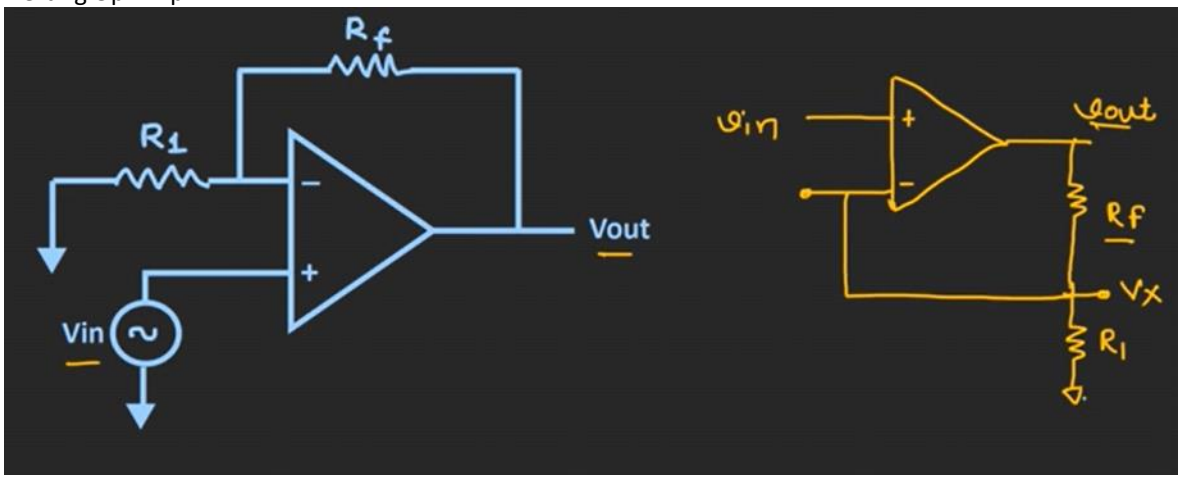


Inverting op-Amp



$V_{out}/V_{in} = - R_f / R_1 = A_{\text{closed loop gain}}$
-ve sign implies 180° out of phase

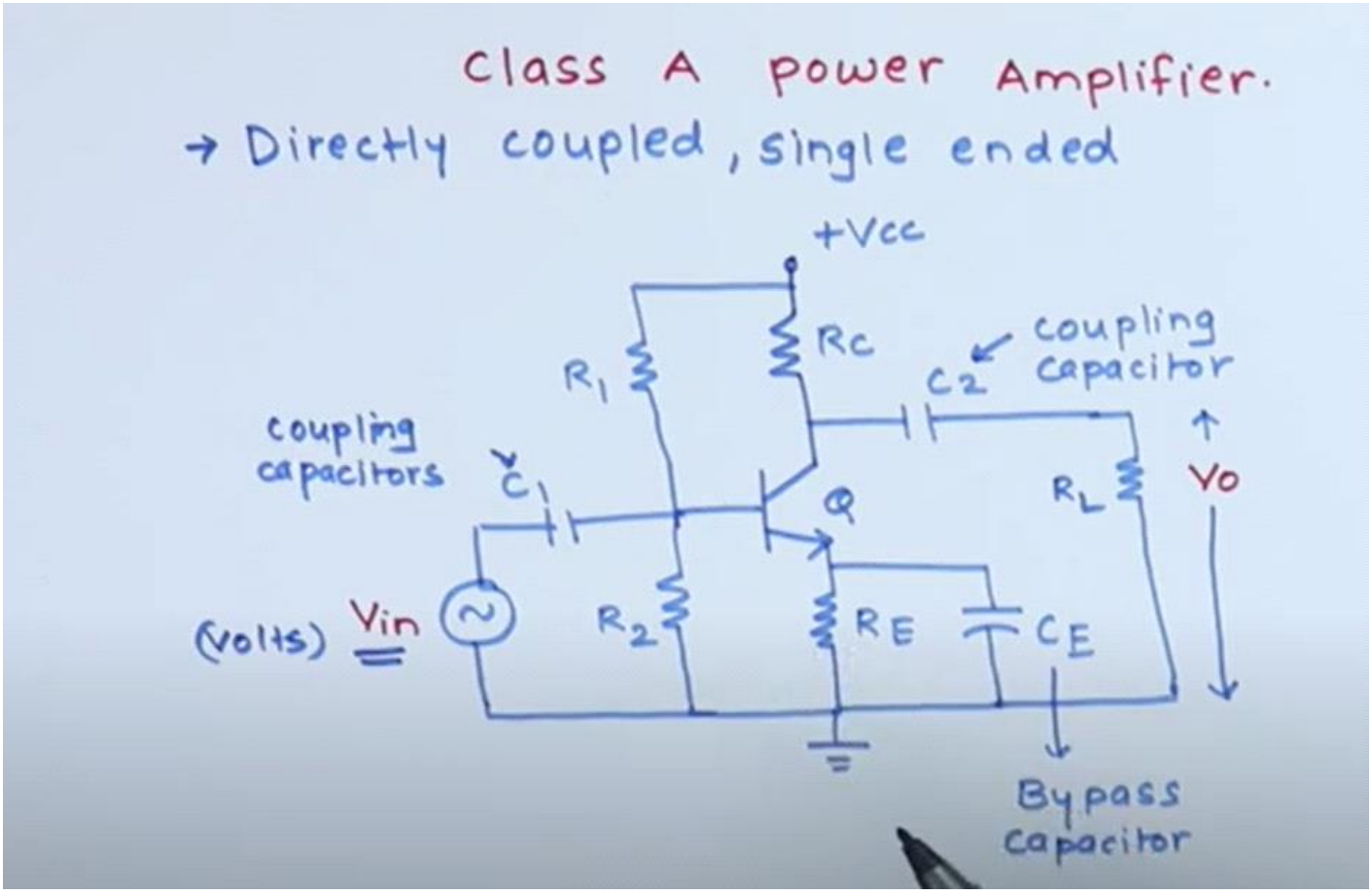
Non inverting Op-Amp



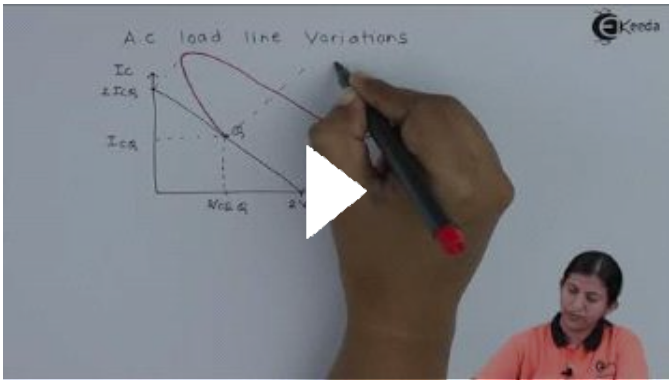
$V_x = R_1/R_1+R_f * V_{out}$
$V_{out}/V_{in} = (1+ R_f / R_1) = A_{\text{closed loop gain}}$

Class A

22 January 2021 07:13 AM



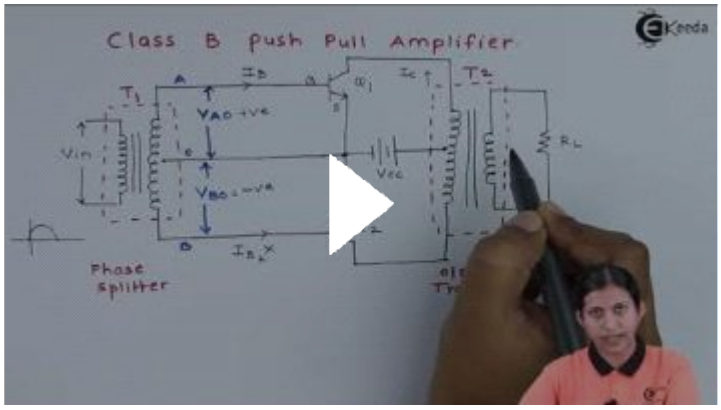
[Single Stage Class A Power Amplifier - High Power Amplifiers - Applied Electronics]



Class B

22 January 2021 09:29 AM

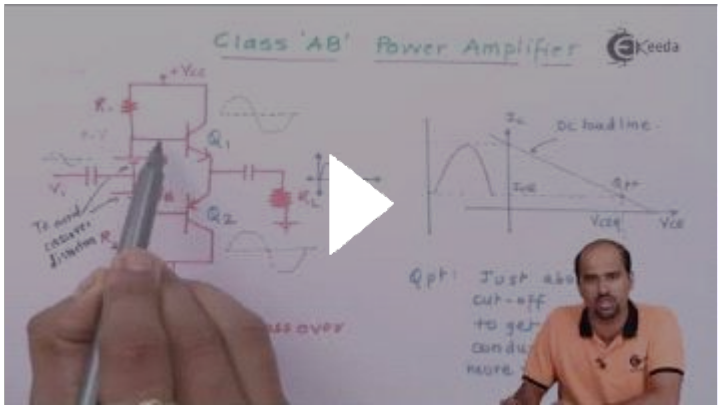
[Class B Power Amplifier - High Power Amplifiers - Applied Electronics]



Class AB

23 January 2021 11:56 AM

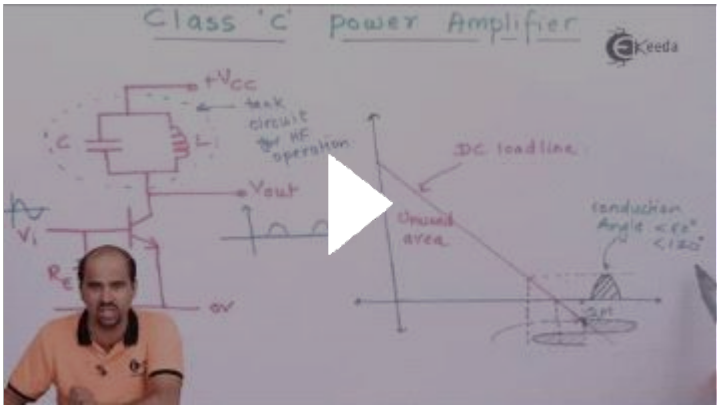
[Class AB Power Amplifier - Large Signal Amplifier - Electronic Devices and Circuits]



Class C

23 January 2021 11:56 AM

[Class C Power Amplifier - Large Signal Amplifier - Electronic Devices and Circuits]



Class C Amplifier

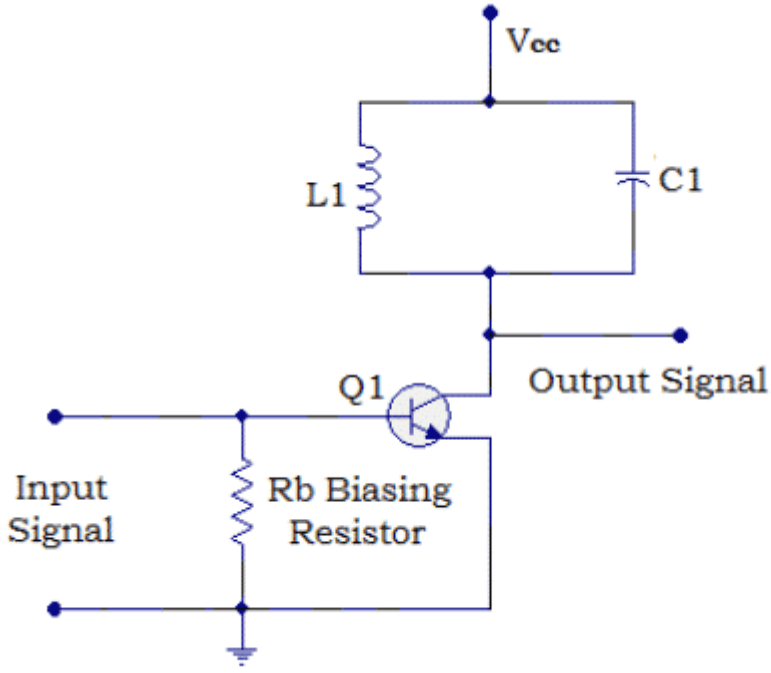


Fig. 4 – Circuit Diagram of Class C Power Amplifier

As shown in the above circuit diagram, Resistor Rb connects to the transistor Q1 base. Biasing resistor Rb pulls the base of Q1 further downwards and the Q-point will be set some way below the cut-off point(In cutoff the collector current is I_{CO} which will be of micro amperes order and hence can be assumed to be zero) in the DC load line. The dc load line is the locus of I_C and V_{CE} at which BJT remains in active region. As a result the transistor will start conducting only after the input signal

amplitude has risen above the base emitter voltage (Vbe~0.7V) plus the downward bias voltage caused by Rb. That is the reason why the major portion of the input signal is absent in the output signal.

As shown in Figure 4, inductor L1 and capacitor C1 forms a tuned circuit which is also called a tank circuit. LC circuits are used either for generating signals at a particular frequency, or picking out a signal at a particular frequency from a more complex signal which extract the required signal from the pulsed output of the transistor.
A series of current pulses is produced by the transistor (active element) according to the input which flow through the resonant circuit. The tank circuit oscillates in the frequency of the input signal by selecting the proper value of L and C. All other frequencies are attenuated by tank circuit and the tank circuit oscillates in one frequency.

We can set the two reactance formula equal to each other

Then the frequency at which this will happen is given as:

$$X_L = 2\pi f L \quad \text{and} \quad X_C = \frac{1}{2\pi f C}$$

$$\text{at resonance:} \quad X_L = X_C$$

$$\therefore \; 2\pi f L = \frac{1}{2\pi f C}$$

$$2\pi f^2 L = \frac{1}{2\pi C}$$

$$\therefore \; f^2 = \frac{1}{(2\pi)^2 LC}$$

$$f = \frac{\sqrt{1}}{\sqrt{(2\pi)^2 LC}}$$

Then by simplifying the above equation we get the final equation

for **Resonant Frequency**, f_r in a tuned LC circuit as:

Resonant Frequency of a LC Oscillator

$$f_r = \frac{1}{2\pi\sqrt{LC}}$$

- Where:
- L is the Inductance in Henries
- C is the Capacitance in Farads
- f_r is the Output Frequency in Hertz

The required frequency is obtained by using a suitably tuned load. The output signal noise can be eliminated by using additional filters. For transferring the power to the load from the tank circuit, a coupling transformer is used.

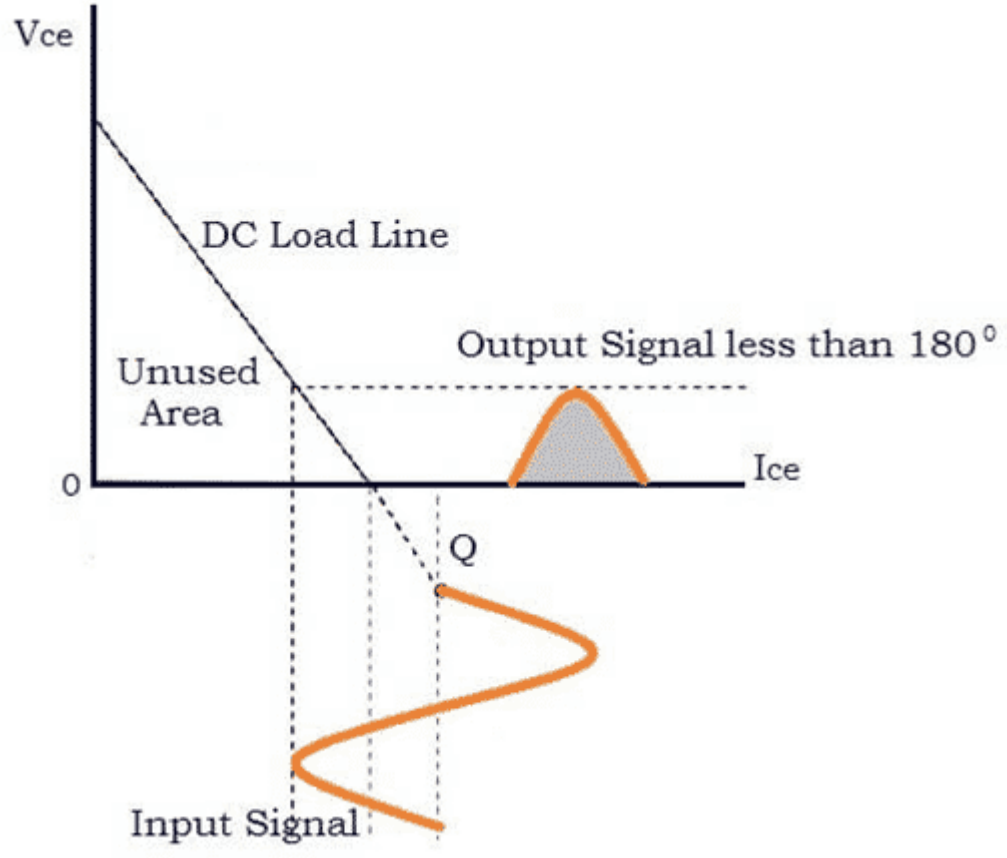


Fig. 5 – Characteristics of Class C Amplifier

As shown in Figure 5, it can be observed that the operating point is placed some way below the cut-off point in the DC load-line and so only a fraction

of the input waveform is available at the output.

Applications of Class C Amplifier

Class C Amplifier is used in: –

RF oscillators.
RF amplifier.
FM transmitters.
Booster amplifiers.
High frequency repeaters.
Tuned amplifiers etc.

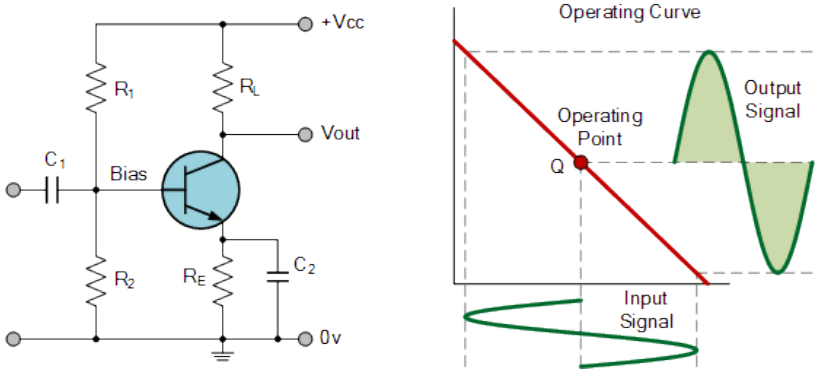
Summary

23 January 2021 11:56 AM

Classification of Power Amplifier.

Parameter	Class A	Class B	class AB	class C
1) Angle of conduction	360°	180°	more than 180° But less than 360°	less than 180°
2) Efficiency	25%. can extend 50%.	78.5%.	78.5 %.	95%.
3) Position of operating point (Q-point)	Exactly at the centre of load line 	on x-axis 	just above x-axis 	Below x-axis 
4) Distortion	No distorion	more than A & AB less than C	less distorition than B, C, A more distortion	Maxm distortion
5) Application	outdoor musical sym	Audio Ampr	RF Ampr	Audio power Amplr.

Class A Amplifier

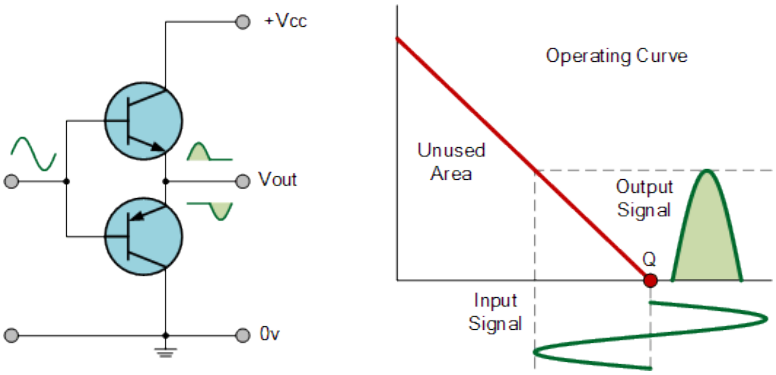


The efficiency of this type of circuit is very low (less than 30%) and delivers small power outputs for a large drain on the DC power supply. A Class A amplifier stage passes the same load current even when no input signal is applied so large heatsinks are needed for the output transistors.

Class B Amplifier

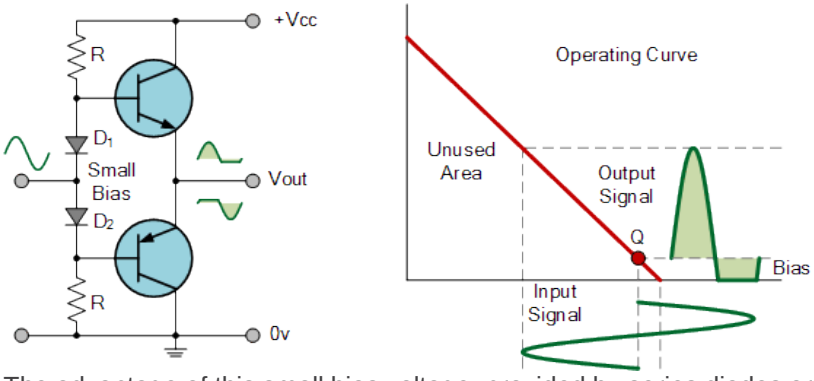
Class B amplifier is a type of power amplifier where the active device (transistor) conducts only for one half cycle of the input signal. That means the conduction angle is 180° for a Class B amplifier.

Theoretical maximum efficiency of Class B power amplifier is 78.5%.



When the input signal goes positive, the positive biased transistor conducts while the negative transistor is switched “OFF”. Likewise, when the input signal goes negative, the positive transistor switches “OFF” while the negative biased transistor turns “ON” and conducts the negative portion of the signal. Thus the transistor conducts only half of the time, either on positive or negative half cycle of the input signal.

Class AB Amplifier

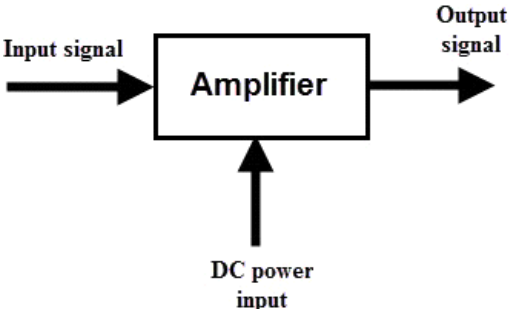


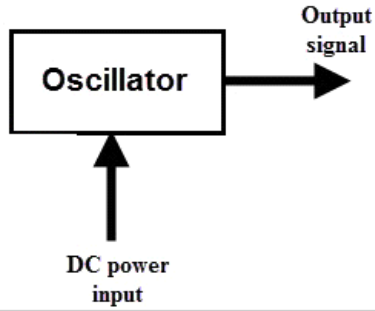
The advantage of this small bias voltage, provided by series diodes or resistors, is that the crossover distortion created by the class B amplifier characteristics is overcome, without the inefficiencies of the class A amplifier design. So the class AB amplifier is a good compromise between class A and class B in terms of efficiency and linearity, with conversion efficiencies reaching about 50% to 60%.

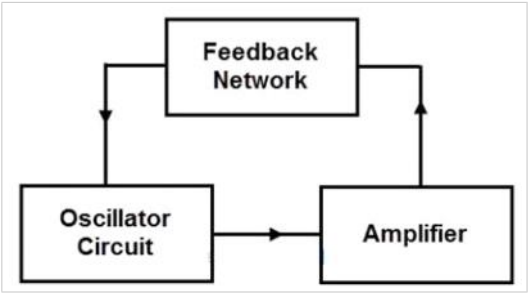
Feedback and Oscillation

An electronic circuit used to generate the output signal with constant amplitude and constant desired frequency is called as an oscillator. It is also called as a waveform generator which incorporates both active and passive elements.

The primary function of an oscillator is to convert DC power into a periodic signal or AC signal at a very high frequency. An oscillator does not require any external input signal to produce sinusoidal or other repetitive waveforms of desired magnitude and frequency at the output and even without use of any mechanical moving parts.







In case of amplifiers, the energy conversion starts as long as the input signal is present at the input, i.e., amplifier produces an output signal whose frequency or waveform is similar to the input signal but magnitude or power level is generally high. The output signal will be absent if there is no input signal at the input.

In contrast, to start or maintain the conversion process an oscillator does not require any input signal as shown figure. As long as the DC power is connected to the oscillator circuit, it keeps on producing an output signal with frequency decided by components in it.

The above figure shows the block diagram of an oscillator. An oscillator circuit uses a vacuum tube or a transistor to generate an AC output.

The output oscillations are produced by the tank circuit components either as R and C or L and C. For continuously generating output without the requirement of any input from preceding stage, a feedback circuit is used.

From the above block diagram, oscillator circuit produces oscillations that are further amplified by the amplifier. A feedback network gets a portion of the amplifier output and feeds it the oscillator circuit in correct phase and magnitude.

Therefore, undamped electrical oscillations are produced , by continuously supplying losses that occur in the tank circuit.

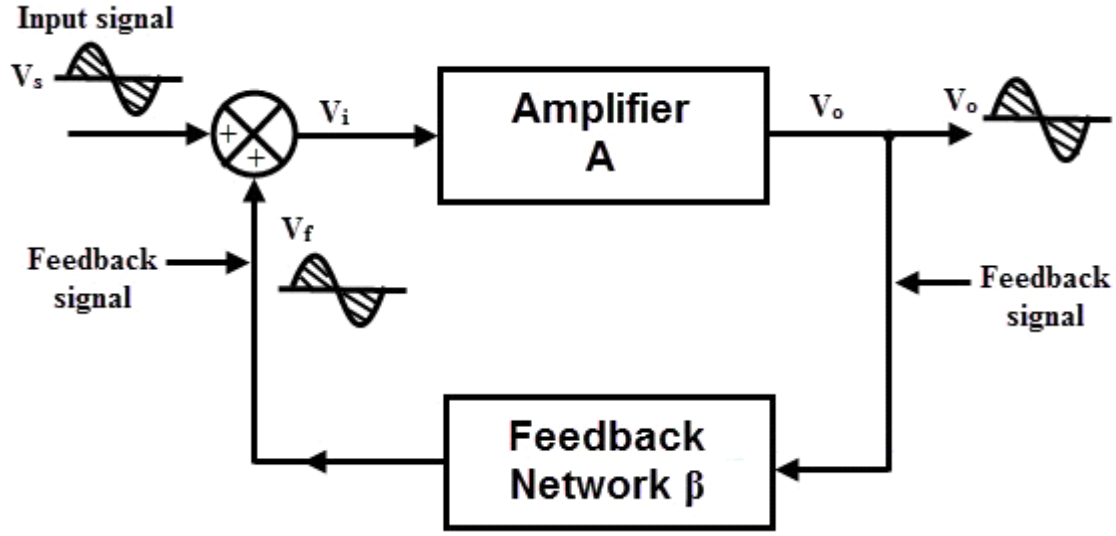
Oscillators

23 January 2021 01:38 PM

Oscillators Theory

The main statement of the oscillator is that the oscillation is achieved through positive feedback which generates the output signal without input signal. Also, the voltage gain of the amplifier increases with the increase in the amount of positive feedback.

In order to understand this concept, let us consider a non-inverting amplifier with a voltage gain 'A' and a positive feedback network with feedback gain of β as shown in figure.



Let us assume that a sinusoidal input signal Vs is applied at the input. Since the amplifier is non-inverting, the output signal Vo is in phase with Vs. A feedback network feeds the part of Vo to the input and the amount Vo fed back depends on the feedback network gain β.

No phase shift is introduced by this feedback network and hence the feedback voltage or signal Vf is in phase with Vs. A feedback is said to be positive when the phase of the feedback signal is same as that of the input signal.

The open loop gain 'A' of the amplifier is the ratio of output voltage to the input voltage, i.e.,

$A = Vo/Vi$

By considering the effect of feedback, the ratio of net output voltage Vo and input supply Vs called as a closed loop gain Af (gain with feedback).

$A_f = V_o/V_s$

Since the feedback is positive, the input to the amplifier is generated by adding Vf to the Vs,

$V_i = V_s + V_f$

Depends on the feedback gain β, the value of the feedback voltage is varied, i.e.,

$V_f = \beta V_o$

Substituting in the above equation,

$V_i = V_s + \beta V_o$

$V_s = V_i - \beta V_o$

Then the gain becomes

$A_f = V_o / (V_i - \beta V_o)$

By dividing both numerator and denominator by Vi, we get

$A_f = (V_o / V_i) / (1 - \beta)(V_o / V_i)$

$A_f = A / (1 - A\beta)$ since $A = V_o/V_i$

Where Aβ is the loop gain and if $A\beta = 1$, then Af becomes infinity. From the above expression, it is clear that even without external input ($V_s = 0$), the circuit can generate the output just by feeding a part of the output as its own input.

And also closed loop gain increases with increase in amount of positive feedback gain. The oscillation rate or frequency depends on amplifier or feedback network or both.

Barkhausen Criterion or Conditions for Oscillation

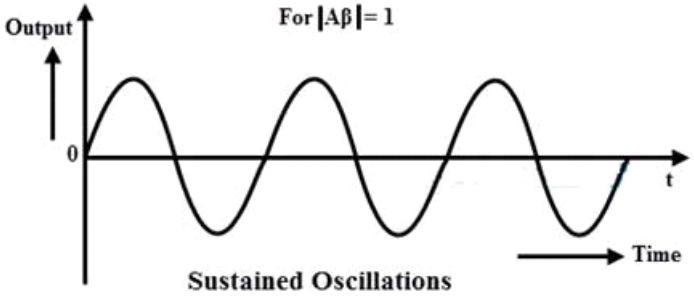
23 January 2021 01:39 PM

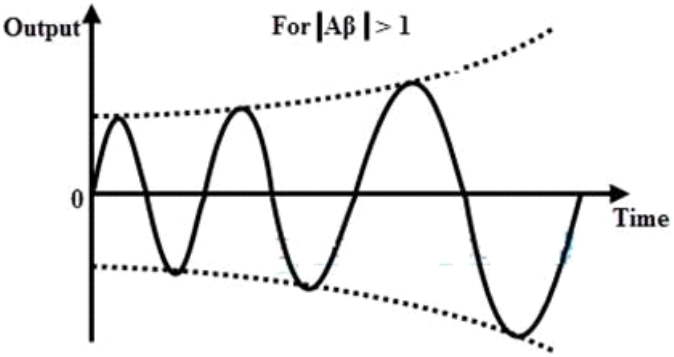
The circuit will oscillate when two conditions, called as Barkhausen's criteria are met. These two conditions are

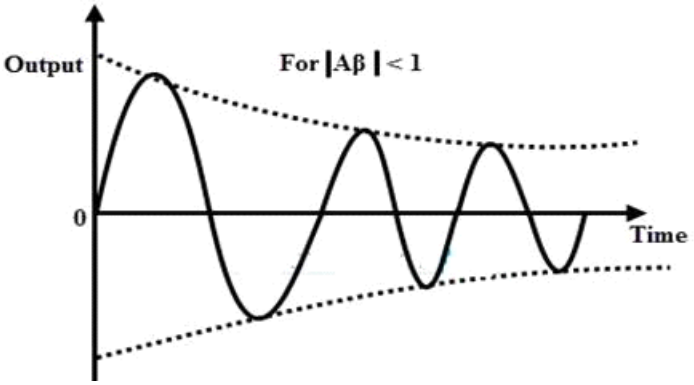
1. The loop gain must be unity or greater

2. The feedback signal feeding back at the input must be phase shifted by 360 degrees (which is same as zero degrees). In most of the circuits, an inverting amplifier is used to produce 180 degrees phase shift and additional 180 degrees phase shift is provided by the feedback network.

At only one particular frequency, a tuned inductor-capacitor (LC circuit) circuit provides this 180 degrees phase shift.







Intro

23 January 2021 01:54 PM

Wien Bridge Oscillator

A Wien-Bridge **Oscillator** is a type of phase-shift oscillator which is based upon a Wien-Bridge network (Figure 1a) comprising of four arms connected in a bridge fashion. Here two arms are purely resistive while the other two arms are a combination of resistors and capacitors. In particular, one arm has resistor and capacitor connected in series (R_1 and C_1) while the other has them in parallel (R_2 and C_2). This indicates that these two arms of the network behave identical to that of high pass filter or low pass filter, mimicking the behavior of the circuit shown by Figure 1b.

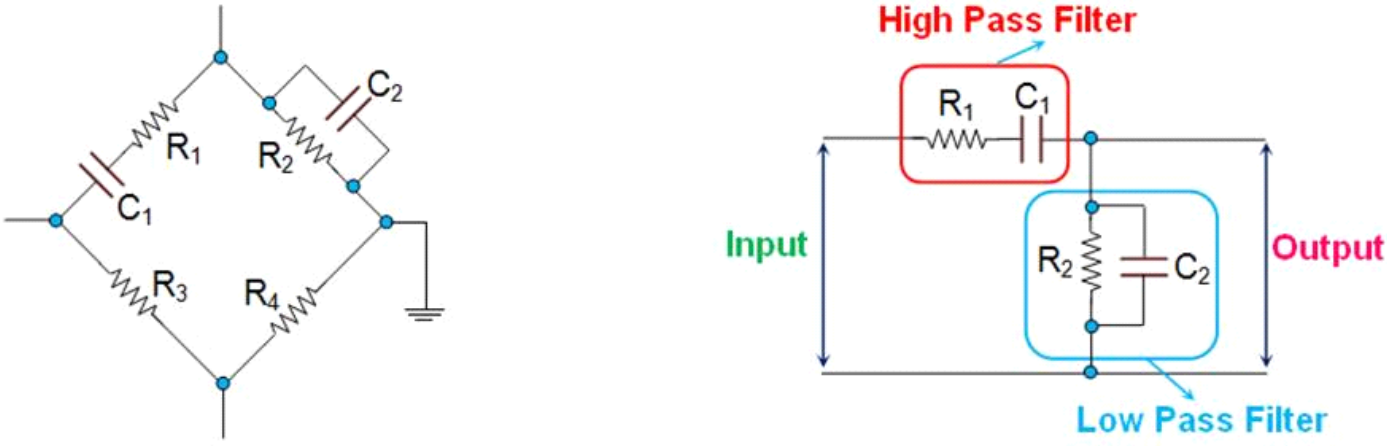


Figure 1 (a) Wien-Bridge Network (b) Two arms of the Wien-Bridge Network

In this circuit, at high frequencies, the reactance of the capacitors C_1 and C_2 will be much less due to which the voltage V_0 will become zero as R_2 will be shorted. Next, at low frequencies, the reactance of the capacitors C_1 and C_2 will become very high.
However even in this case, the output voltage V_0 will remain at zero only, as the capacitor C_1 would be acting as an open circuit. This kind of behavior exhibited by the Wien-Bridge network makes it a lead-lag circuit in the case of low and high frequencies, respectively.

Frequency

23 January 2021 01:56 PM

Wien Bridge Oscillator Frequency Calculation

Nevertheless, amidst these two high and low frequencies, there exists a particular frequency at which the values of the resistance and the capacitive reactance will become equal to each other, producing the maximum output voltage. This frequency is referred to as resonant frequency. The resonant frequency for a Wein Bridge Oscillator is calculated using the following formula:

Further, at this frequency, the phase-shift between the input and the output will become zero and the magnitude of the output voltage will become equal to one-third of the input value. In addition, it is seen that the Wien-Bridge will be balanced only at this particular frequency.

$$f_r = \frac{1}{2\pi\sqrt{R_1 C_1 R_2 C_2}}$$

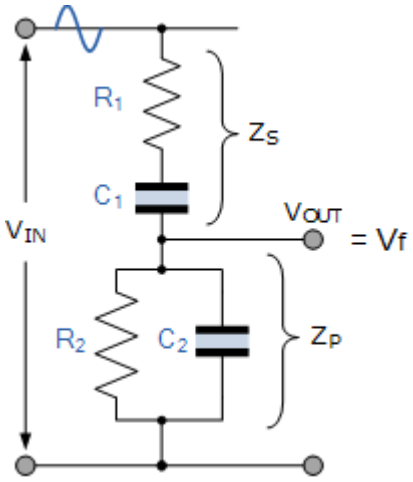
$$if\ R_1 = R_2 = R\ and\ C_1 = C_2 = C$$

$$then\ f_r = \frac{1}{2\pi RC}$$

Frequency calculation

23 January 2021 02:00 PM

The feedback network of wein bridge oscillator is as shown in the figure:



•In order to obtain the expression for the feedback factor and oscillation frequency we have to consider above feedback network. Vin is the actually the output voltage Vo which comes from the amplifier output and voltage across R2 and C2 acts as feedback voltage Vf.

•The feedback factor or the gain of the feedback network β is defined as:

$$\beta = Vf/Vin \quad \text{.....1}$$

• But Vf is the voltage across the parallel combination of R2 and C2 .Let the impedance of the arm which has R1 and C1 in series be Zs and the arm which has R2 and C2 in parallel be Zp. Therefore

$$Vf = (Zp)/((Zp+Zs))*Vin \quad \text{.....2}$$

And $\beta = Zp/((Zp+Zs))$ 3

Lets obtained equations for Zp and Zs

$Zs = R1 + 1/jwC1 = (1 + jwR1C1)/jwC1$ 4

And

$Zp = R2||Xc2 = R2||1/(jwC2) = (R2*(1/jwC2))/(R2+(1/jwC2)) = R2/((1+jwR2C2))$

.....5

Substituting equation 4 and 5 in equation 3 we get

$\beta = ([R2/(1+jwR2C2)])/([1+jwR1C1/jwC1]+[R2/(1+jwR2C2)])$

Substituting jw=s in above equation and solving for β we get

β= ([R2/(1+sR2C2)])/([1+sR1C1/sC1]+[R2/(1+sR2C2)])

β=sC1R2/([(1+sR1C1)(1+sR2C2)]+sR2C1)

=sC1R2/(1+s(R1C1+R2C2)+S^2 R1R2C1C2+SR2C1)

β= sC1R2/(1+s(R1C1+R2C2+R2C1)+s^2 R1C1R2C2)
.....6

Re substitute s = jw and s^2= j^2.w^2= -w^2 into equation 6

β= jwC1R2/ (1 +jw(R1C1+R2C2+R2C1)+s-w^2 R1C1R2C2)

β = jwC1R2/((1-w^2 R1R2C1C2)+jw(R1C1+R2C2+R2C1))
.....7

Rationalize the equation 7 to get

β = (jwC1R2[(1-w^2 R1C1R2C2)-jw(R1C1+R2C2+R2C1)])/((1-w^2 R1C1R2C2)^2+w^2 (R1C1+R2C2+R2C1)^2)
.....8

As mentioned earlier, the phase shift introduced by wein bridge circuit at the desired output frequency should be 0°.For that the imaginary part of equation 8 should have a zero value.

wC1R2[(1-w^2 R1C1R2C2) =0

Therefore w^2 R1C1R2C2 =1

or w^2=1/R1C1R2C2

w = 1/√(R1C1 R2C2)

And frequency f = 1/(2π√(R1C1 R2C2))
.....9

This is the expression for the oscillator frequency.

If we substitute R1 = R2 =R and C1 =C2= C in the expression for the oscillator frequency, then equation 9 gets modified as follows.

Oscillator frequency f = 1/2πRC.....10

Similerly if we substitute R1 = R2 =R and C1 =C2= C into expression of β in equation 8 then we get

Feedback factor β = (w^2 CR(3RC)+jwCR(1-w^2 R^2 c^2))/((1-w^2 R^2 c^2)^2+w^2 (3RC)^2)
.....11

This is required expression for the feedback factor β.

Now substitute w =1/RC in above equation we get,

$\beta = 3/(0+[1/(C^2 R^2)(3RC)]^2)=3/9$

$\beta = 1/3$.....12

Thus at the oscillator frequency "f" the value of feedback factor β is "1/3".

Gain of amplifier for sustained oscillations:

According to the barkhausen criteria the loop gain should be greater than 1 and phase shift around the loop should be zero. The positive sign of β in equation 12 shows that the phase shift introduced at frequency "f" is zero. To satisfy the other condition ie **$|A\beta| \geq 1$**

Substituting value of β we get

$A \geq 3$.....13

Thus the amplifier gain should be at least equal to 3 to ensure sustained oscillations.

Wein bridge with bjt

23 January 2021 02:00 PM

In the case of **Wien-Bridge oscillator**, the Wien-Bridge network of Figure 1 will be used in the feedback path as shown in Figure 2. The circuit diagram for a Wein Oscillator using a BJT (Bipolar Junction Transistor) is shown below:

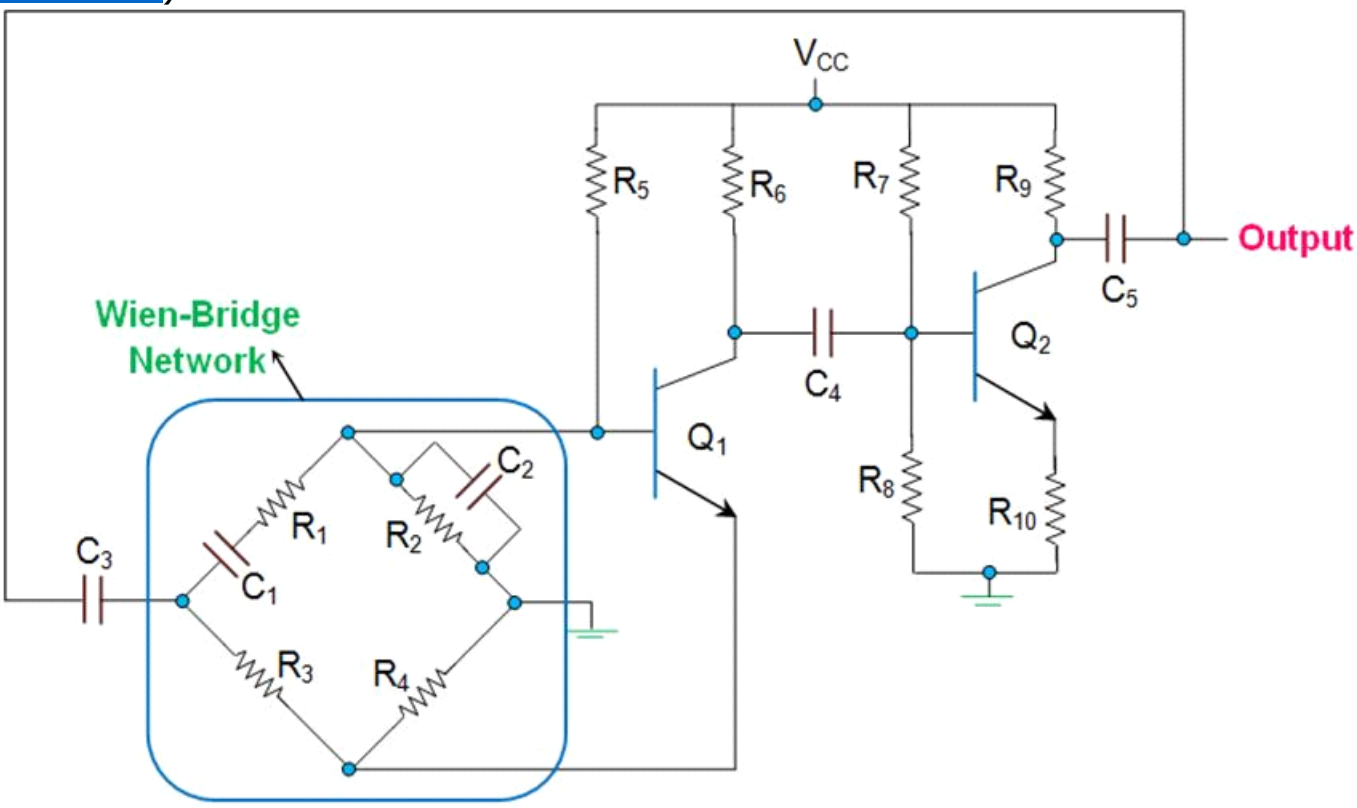


Figure 2 Wien-Bridge Oscillator Using BJT

In these oscillators, the amplifier section will comprise of two-stage amplifier formed by the transistors, Q_1 and Q_2, wherein the output of Q_2 is back-fed as an input to Q_1 via Wien-Bridge network (shown within the blue enclosure in the figure). Here, the noise inherent in the circuit will cause a change in the base current of Q_1 which will appear at its collector point after being amplified with a phase-shift of 180^o.
This is fed as an input to Q_2 via C_4 and gets further amplified and appears with an additional phase-shift of 180^o. This makes the net phase-difference of the signal fed back to the Wien-Bridge network to be 360^o, satisfying phase-shift criterion to obtain sustained oscillations. However, this condition will be satisfied only in the case of resonant frequency, due to which the Wien-Bridge oscillators will be highly selective in terms of frequency, leading to a frequency-stabilized design.

Intro

23 January 2021 02:05 PM

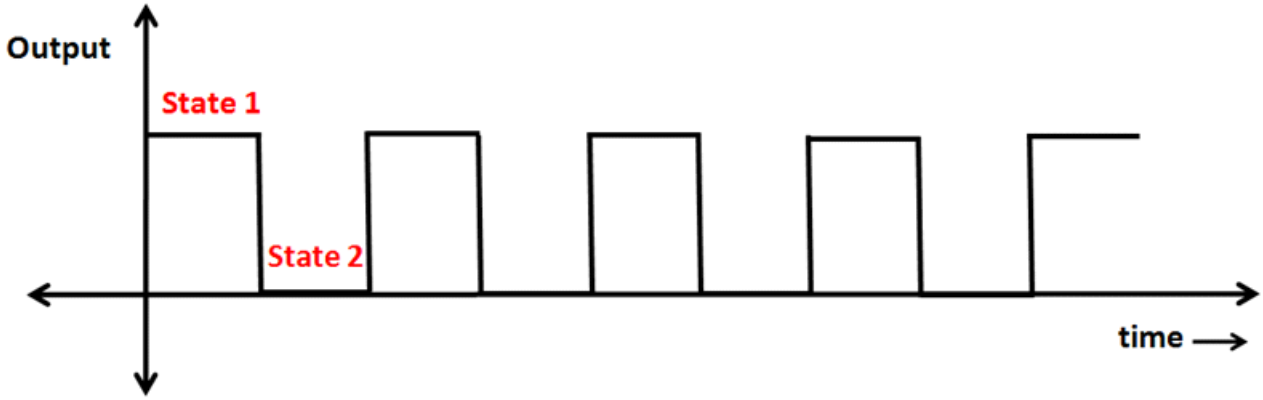
The Multivibrator circuits are widely used in electronics. It is the electronics circuit which is used to implement the two state devices like Relaxation Oscillator, Timer and Flip-flops.
Here the two states refers to the two voltage levels of the output. (e.g 0V, and 5V). Many times the two voltage levels are also represented as either logic high (e.g 5V) and logic low. (e.g 0V).

The multivibrators can be classified into three catagories.

1. Astable Multivibrator
2. Monostable Multivibrator
3. Bistable Multivibrator

Astable Multivibrator

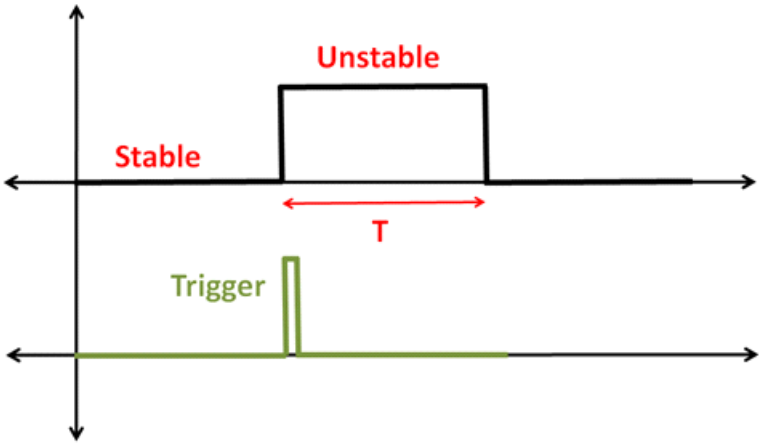
The astable multivibrator does not remain stable in any of the two states. And the output of the multivibrator continuously changes between the two states.



The relaxation oscillators are the example of the astable multivibrator.

Monostbale Multivibrator

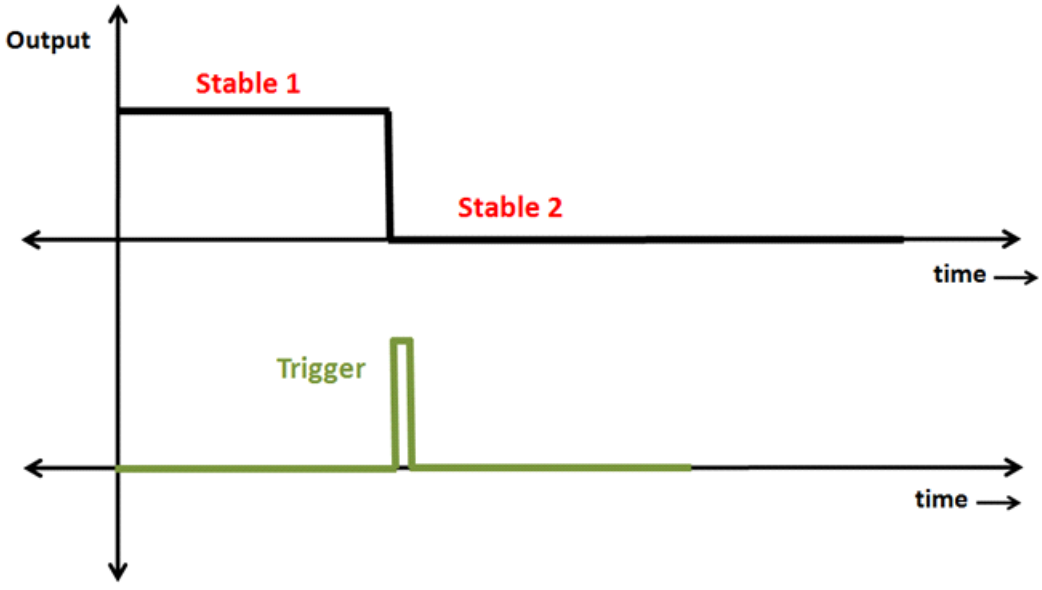
In monostable multivibrator, the output remains in the one stable state but whenever an external trigger signal is applied, the output momentarily goes into the unstable state. And after some time it comes back into the stable state. The time required to come back into the stable state depends on the passive components like R and C.



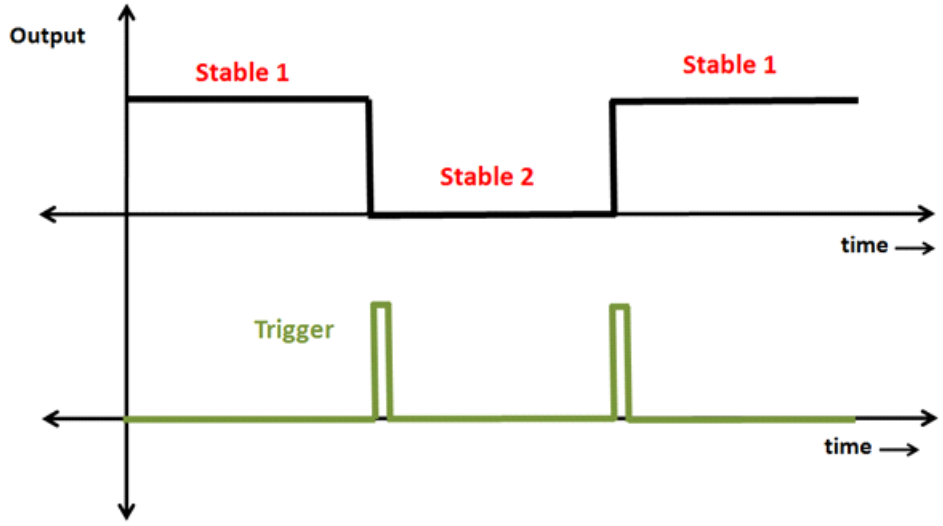
This type of multivibrator is used as a timer in many applications.

Bistable Multivibrator

This type of multivibrator has two stable states. The output used to be in any one of the two stable states. And whenever an external trigger signal is applied, the output goes from one stable state to another stable state. If no triggering action is applied thereafter, then it remains in the new stable state.



The bistable multivibrator is one kind of flip-flop circuit.



All the three types of multivibrator can be designed using either op-amp, transistor pairs or 555 timer IC.

Astable

23 January 2021 03:11 PM

Astable Multi-vibrator

Fig. Astable Multi-vibrator Circuit

Initially, when the supply is given, the capacitor is uncharged and a logic low signal is fed to the input of the NOT gate. This causes the output to be at logic high level. As this logic high signal is fed back to the AND gate, its output is at logic 1. The capacitor starts charging and the input level of the NOT gate increases until it reaches the logic high threshold, and the output is at logic low.

Again, the AND gate output is at logic low (logic low input is being fed back), and the capacitor starts discharging until its potential at input of the NOT gate reaches logic low threshold, and the output is again switched back to the logic high.

This is actually a type of relaxation oscillator circuit.

Monostable

24 January 2021 09:25 AM

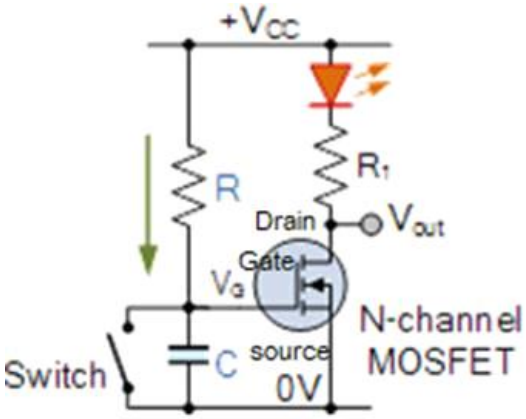


Fig: MOSFET Monostable

Monostable Multivibrators have only **ONE** stable state (hence their name: "Mono"), and produce a single output pulse when it is triggered externally. Monostable Multivibrators only return back to their first original and stable state after a period of time determined by the time constant of the RC coupled circuit.

Consider the MOSFET circuit on the left. The resistor R and capacitor C form an RC timing circuit. The N-channel enhancement mode MOSFET is switched "ON" due to the voltage across the capacitor with the drain connected LED also "ON".

When the switch is closed the capacitor is short circuited and therefore discharges while at the same time the gate of the MOSFET is shorted to ground. The MOSFET and therefore the LED are both switched "OFF". While the switch is closed the circuit will always be "OFF" and in its "unstable state".

When the switch is opened, the fully discharged capacitor starts to charge up through the resistor, R at a rate determined by the RC time constant of the resistor-capacitor network. Once the capacitors charging voltage reaches the lower threshold voltage level of the MOSFETs gate, the MOSFET switches "ON" and illuminates the LED returning the circuit back to its stable state.

Then the application of the switch causes the circuit to enter its unstable state, while the time constant of the RC network returns it back to its stable state after a preset timing period thereby producing a very simple "one-shot" or **Monostable Multivibrator** MOSFET circuit.

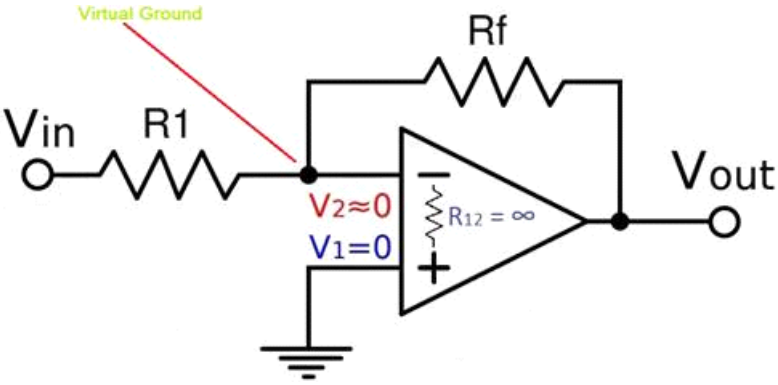
Intro

24 January 2021 09:32 AM

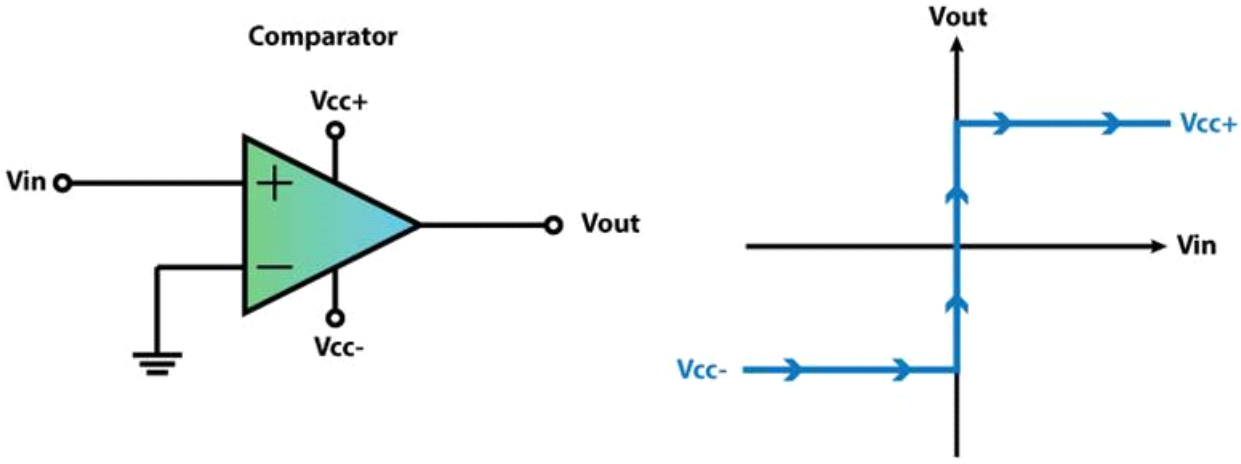
Operational Amplifier based Schmitt Trigger

Virtual ground

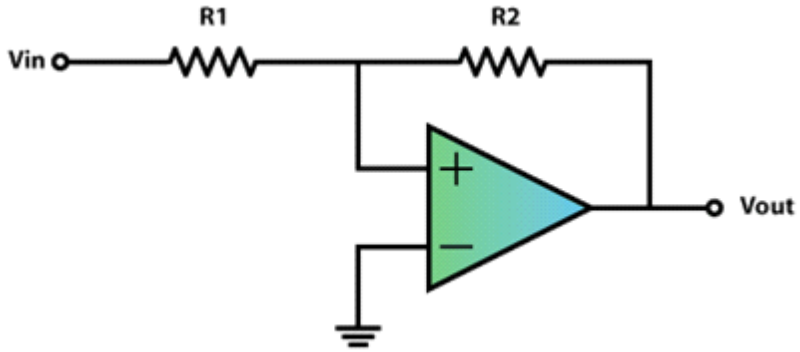
In electronics, a **virtual ground** (or **virtual earth**) is a node of a circuit that is maintained at a steady reference potential, without being connected directly to the reference potential.



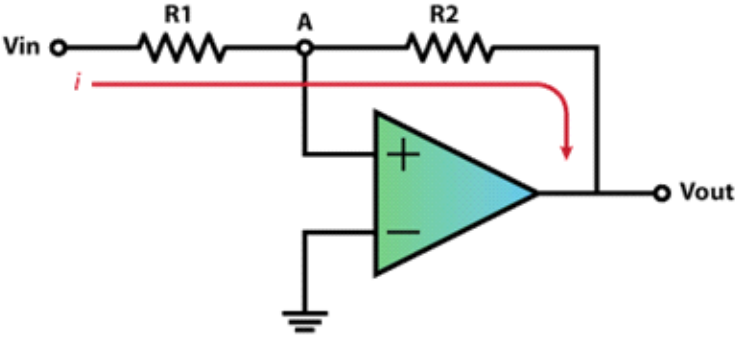
Here we have an op-amp which inverting input is connected to the ground or zero volts and the non-inverting input is connected to a voltage input, V_{IN}. So this is actually a comparator and compares the non-inverting input to the inverting input or in this case the input voltage V_{IN} to 0 V. So when the V_{IN} value is below 0 volts the output of the comparator will be the negative V_{CC} and if the input voltage is above 0 volts the output will be positive V_{CC}.



Now if we add a positive feedback by connecting the output voltage to the non-inverting input with a resistor between them and another resistor between the V_{IN} and the non-inverting input we will get the Schmitt Trigger. Now the output will switch from V_{CC}– to V_{CC}+ when the voltage at the A node will cross 0 volts.

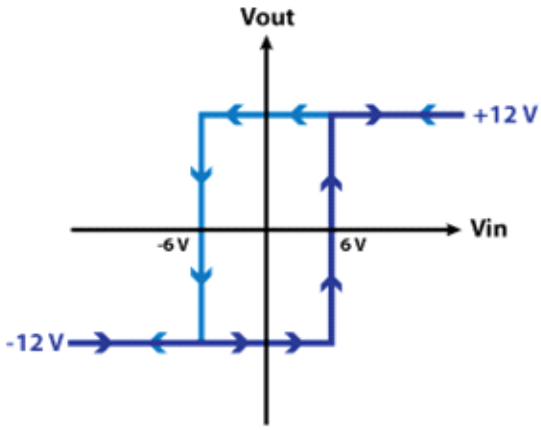


That means that now by adjusting the values of the resistors we can set at what value of the V_{IN} input the switch will occur using the following equations. We get these equations with the following relationships. The current "i" through this line equals $V_{IN} - V_A$ divided by R_1 as well as $V_A - V_{OUT}$ divided by R_2. So if we replace the V_A with zero, as we need that value for the switch to occur, we will get that final equation. For example if the output is -12 volts and the V_{IN} input is negative and rises, the switch from -12 V to +12 V will occur at 6 volts according to the equation and the values of the resistors and vice versa when the V_{IN} input is high and declines the switch from +12 V to – 12V will occur at -6 volts.



Example:
R1 = 1K ; R2 = 2K ; Vout = +/- 12 V

$$Vin = -\frac{1}{2}(+/-12) = +/-6\ V$$



$$i = \frac{Vin - Va}{R1} = \frac{Va - Vout}{R2}$$

$$Va = 0$$

$$\frac{Vin}{R1} = -\frac{Vout}{R2}$$

$$Vin = -\frac{R1}{R2}\,Vout$$

$$\frac{6 - V_a}{1K} = \frac{V_a - (-12)}{2K}$$

$$12\text{-}2V_a = V_a + 12$$

$$3V_a = 0$$

$$V_a = 0$$

At input voltage $V_{in} => from - 6\ V\ to\ 6V, output$ transits from -12V to + 12 Volt. Similarly when input voltage goes from +6 V to -6V, the output transits from +12V to -12 Volt.

intro

25 January 2021 03:35 PM

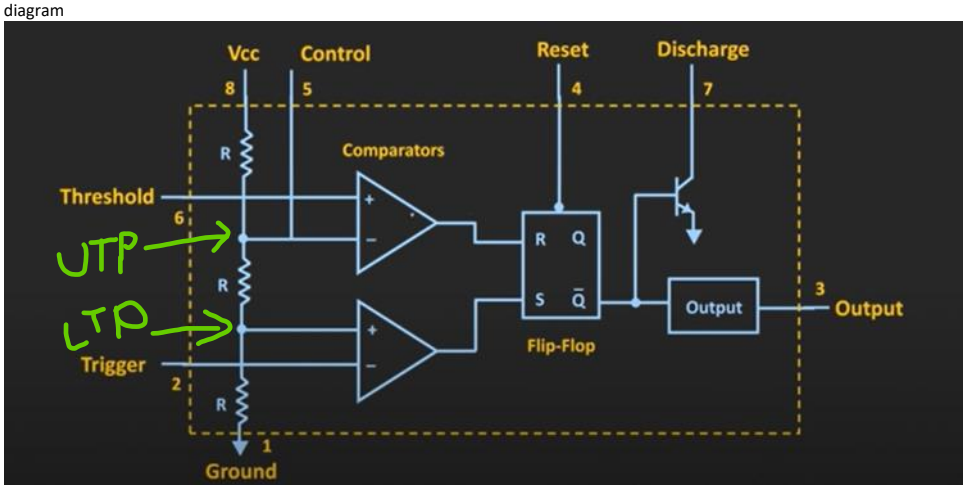
[Introduction to 555 Timer: The Internal Block Diagram and the Pin Diagram Explained]

Flip flop truth table

Truth Table of a RS flipflop

S	R	Q	STATE
0	0	PREVIOUS STATE	NO CHANGE
0	1	0	RESET
1	0	1	SET
1	1	?	FORBIDDEN

Block diagram



UTP = 2/3 Vcc
LTP = 1/3Vcc
Output circuit is just an inverter
State of 555 timer

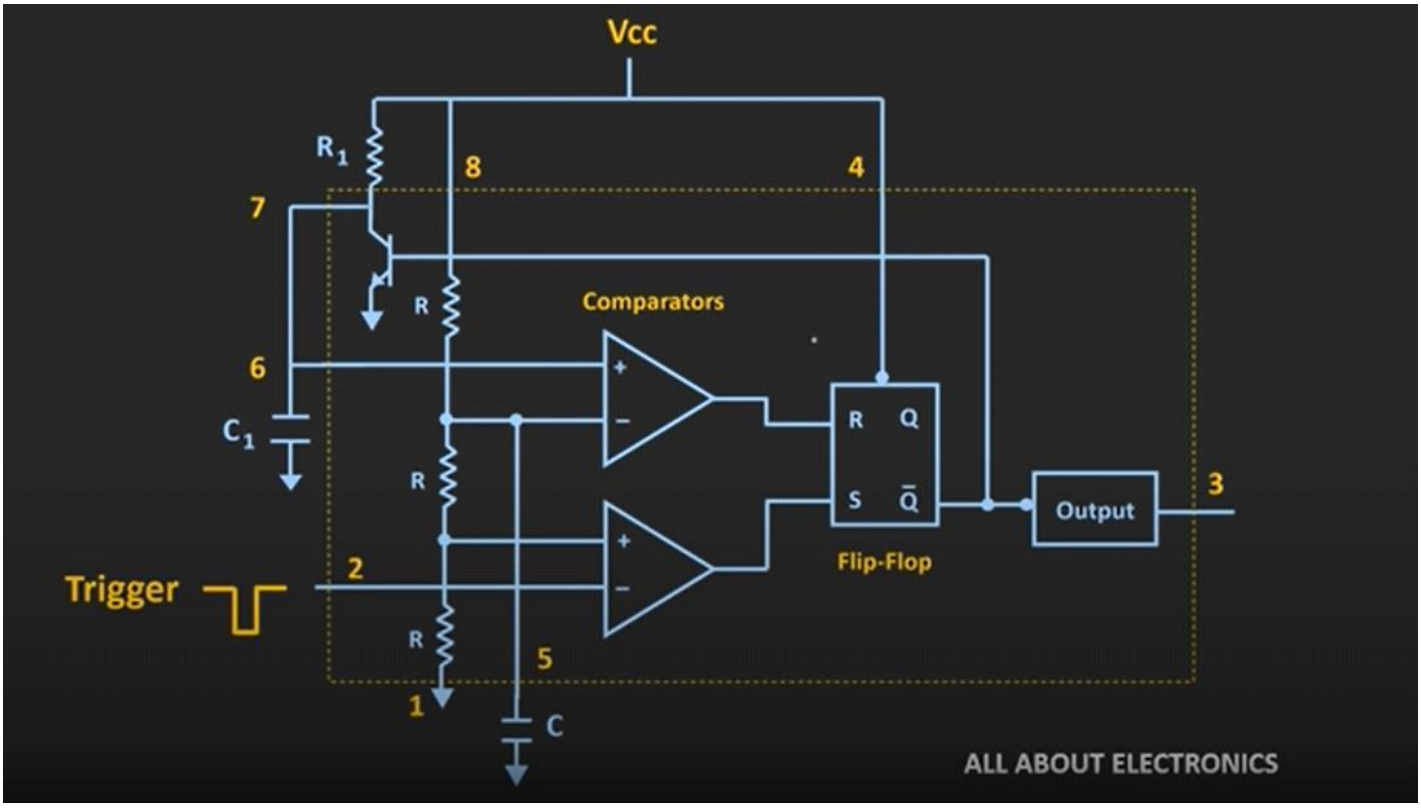
Threshold pin	Trigger pin	Upper R comparator	Lower S comparator	Output pin
Less than UTP	More than LTP	0	0	No change
More than UTP	More Than LTP	1	0	0
Less than UTP	Less than LTP	0	1	1

Monostable Multivibrator

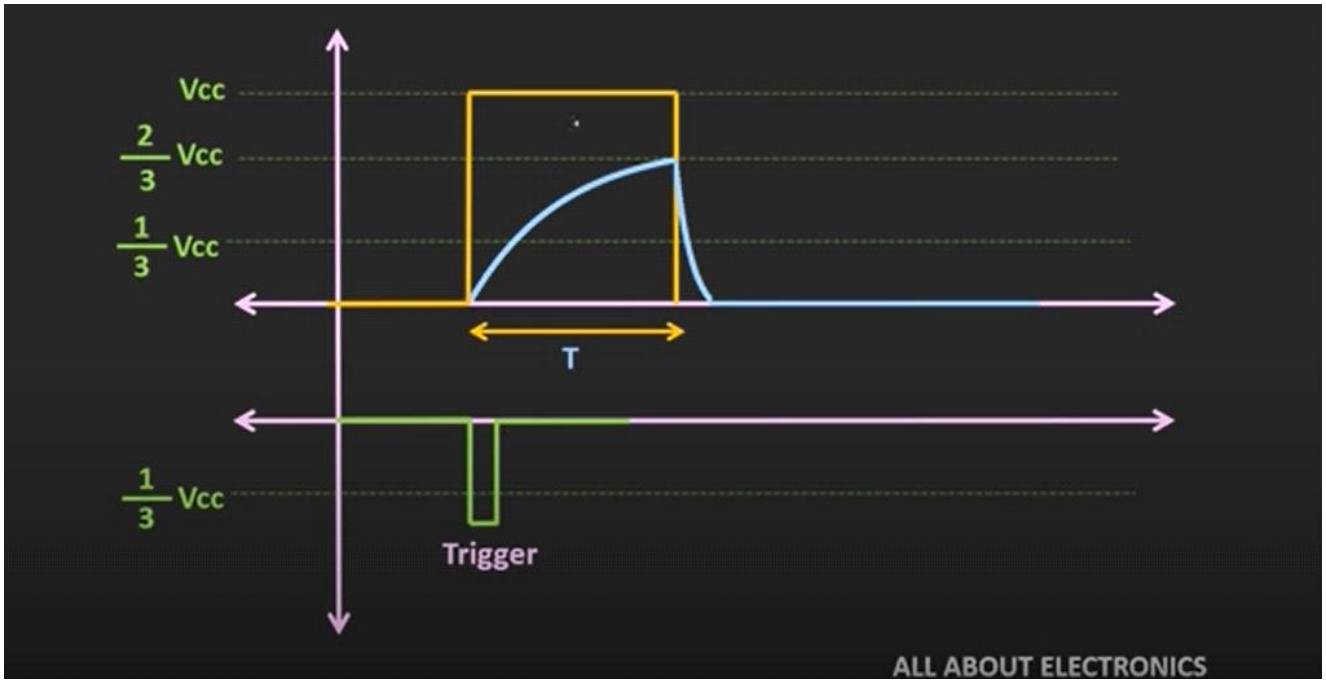
25 January 2021 04:01 PM

[555 Timer as Astable Multivibrator (Working, Design and Derivations)]

Block diagram



- Assume initially output is 0 (stable state).
- Initially upper comparator outputs 0 and lower comparator out puts 0 hence out put is previous state
  - Because q` line runs and is connected with transistor so since output is 0 transistor is fed with1 and pin 7 and 6 is grounded , Thus upper comperator out puts 0
  - Since trigger is not applied lower comperator outputs 0
- Now trigger is applied
- Lower comperator output is momentarily 1 making the s terminal of flip flop 1.
- So out put is 1.
- And the line connecting the transistor is 0 so it is ungrounded, un grounding the capacitor.
- The capacitor starts to charge up and as soon as the charge builds up just more then UTP upper comparator is returned back to normal state.
- The output is again 0 making the whole circuit return to its original state.



T = 1.1RC

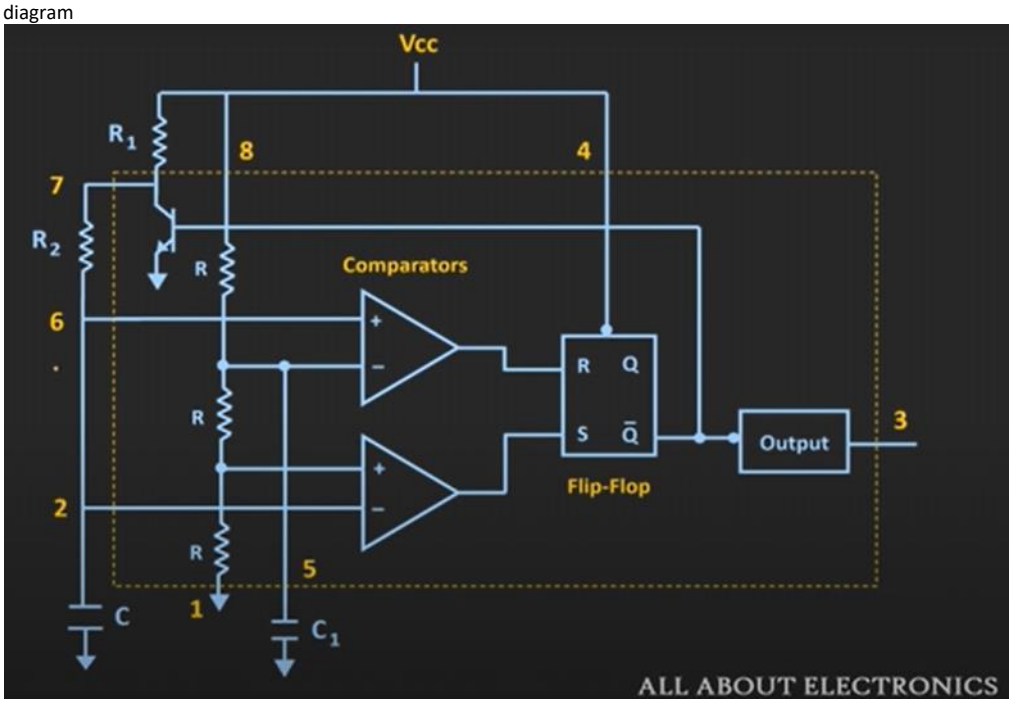
Astable Multivibrator

Aka oscillator

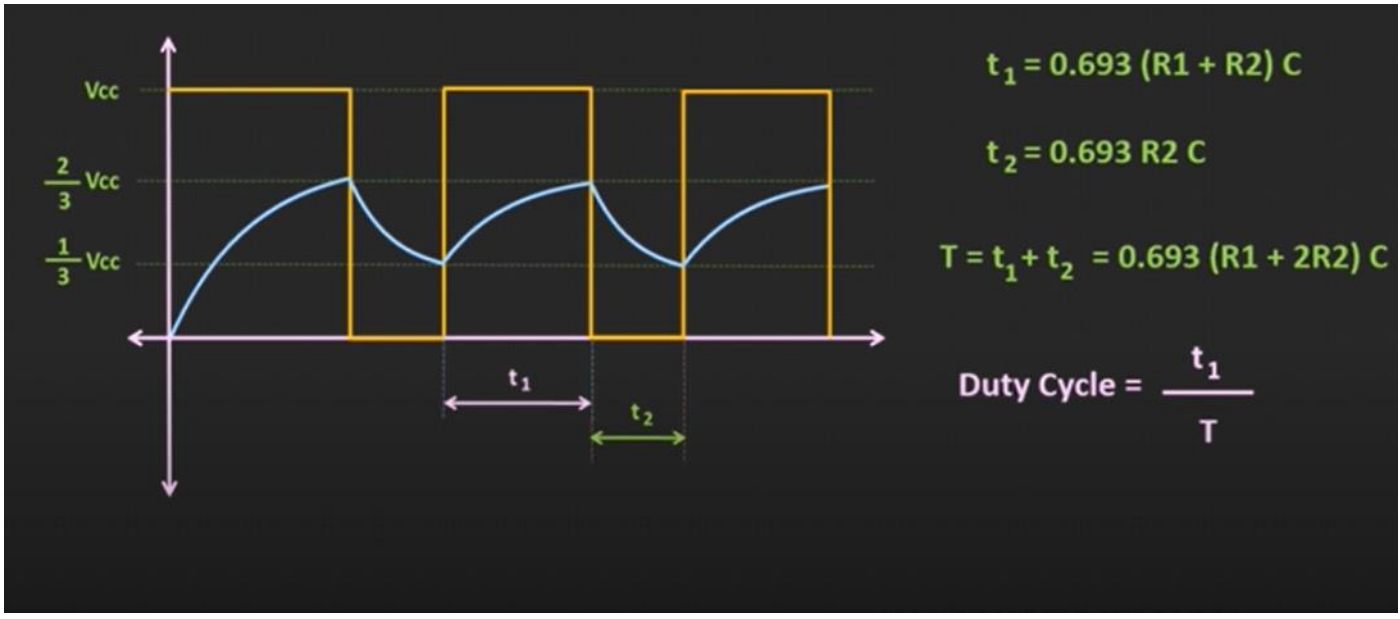
25 January 2021 05:19 PM

[555 Timer as Astable Multivibrator (Working, Design and Derivations)]

Block diagram



- The circuit is switched on capacitor C starts to charge
- Initially potential at 6 and 2 is 0 and is increasing hence initially upper comparator outputs 0 lower comparator outputs 1 the output of timer is 1.
- When voltage of C just surpasses LTP -> upper Comparator 0 , lower comparator 0 , timer output 1
- When voltage of C just surpasses UTP-> upper Comparator 1 , lower comparator 0 , timer output 0
- now the capacitor is grounded and starts to discharge through r1 and r2.
- While discharging --
- When voltage of C just goes below UTP -> upper Comparator 0 , lower comparator 0 , timer output 0 as before
- When voltage of C just goes below LTP -> upper Comparator 0 , lower comparator 1 , timer output 1
- The circuit is restored and process is repeated



In this configuration duty cycle > 50% is possible
If we short a diode along r2 it is possible to achieve any duty cycle

26 January 2021 11:02 AM

BCD or Binary Coded Decimal

Binary Coded Decimal, or **BCD**, is another process for converting decimal numbers into their binary equivalents.

- It is a form of binary encoding where each digit in a decimal number is represented in the form of bits.
- This encoding can be done in either 4-bit or 8-bit (usually 4-bit is preferred).
- **Truth Table for Binary Coded Decimal**

DECIMAL NUMBER	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

In the **BCD numbering system**, the given decimal number is segregated into chunks of four bits for each decimal digit within the number. Each decimal digit is converted into its direct binary form (usually represented in 4-bits).

29= 0010 1001

For example:

1. Convert $(123)_{10}$ in BCD

From the truth table above,
1 -> 0001
2 -> 0010
3 -> 0011
thus, BCD becomes -> 0001 0010 0011

2. Convert $(324)_{10}$ in BCD

$(324)_{10}$ -> 0011 0010 0100 (BCD)

Again from the truth table above,
3 -> 0011
2 -> 0010
4 -> 0100
thus, BCD becomes -> 0001 0010 0011

Many decimal values, have an infinite place-value representation in binary but have a finite place-value in binary-coded decimal.

For example, **0.2 in binary is .001100**… and in **BCD is 0.0010**.

Gray codes

26 January 2021 11:20 AM

Aka

- Reflected binary code
- Unit distance code
- Minimum error code

Properties

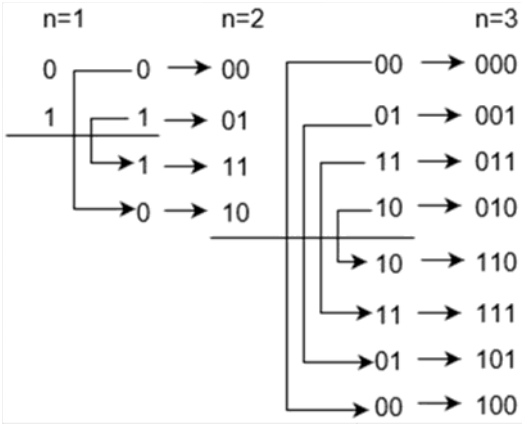
- Two successive values differ by one bit
- It Is preferred over binary to reduce switching operations

Why Gray codes are called reflected binary code?
Because of the way they are formed

Take previous code in sequence: 0 and 1.
Add reversed codes in the following list: 0, 1, 1 and 0.

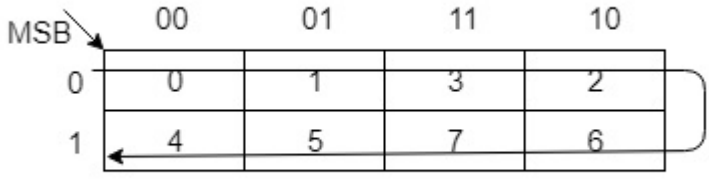
- Now add prefix 0 for original previous code and prefix 1 for new generated code: 00, 01, 11, and 10.

Therefore, Gray code 0 and 1 are for Binary number 0 and 1 respectively. Gray codes: 00. 01, 11, and 10 are for Binary numbers: 00, 01, 10, and 11 respectively. Similarly you can construct Gray code for 3 bit binary numbers:



It can also be generated using k-map as follows

For example, 3 bit Gray codes can be contracted using K-map which is given as following below:



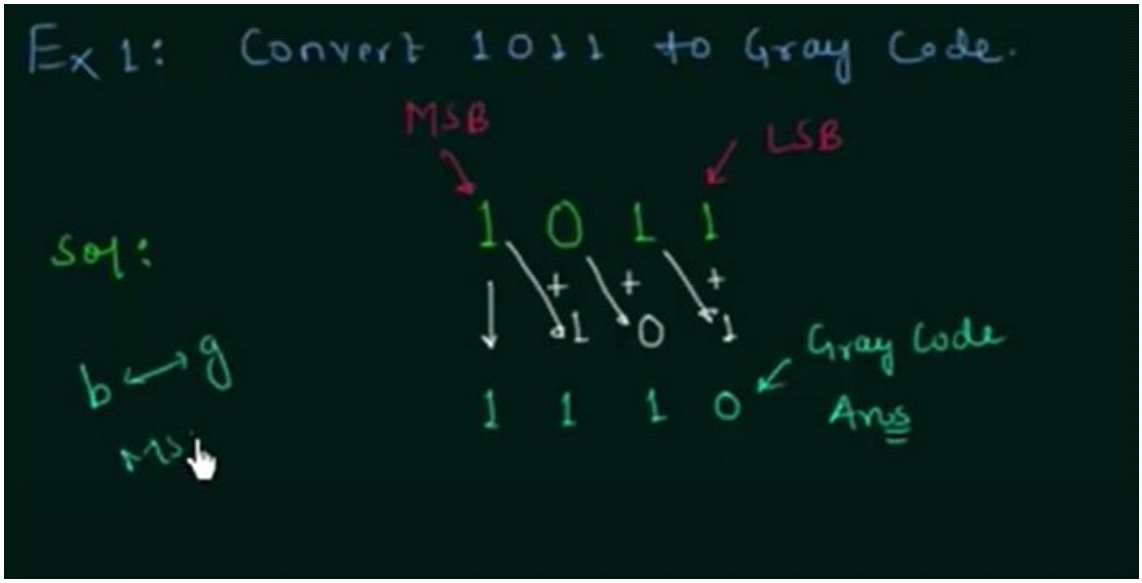
Binary to gray conversion

- Place the MSB of binary as it is

Repeat until 1 binary bits is left:

xor the next two bits of binary and append the result as gray bit

Delete the MSB



* + refers to xor in pic.

gray to binary conversion

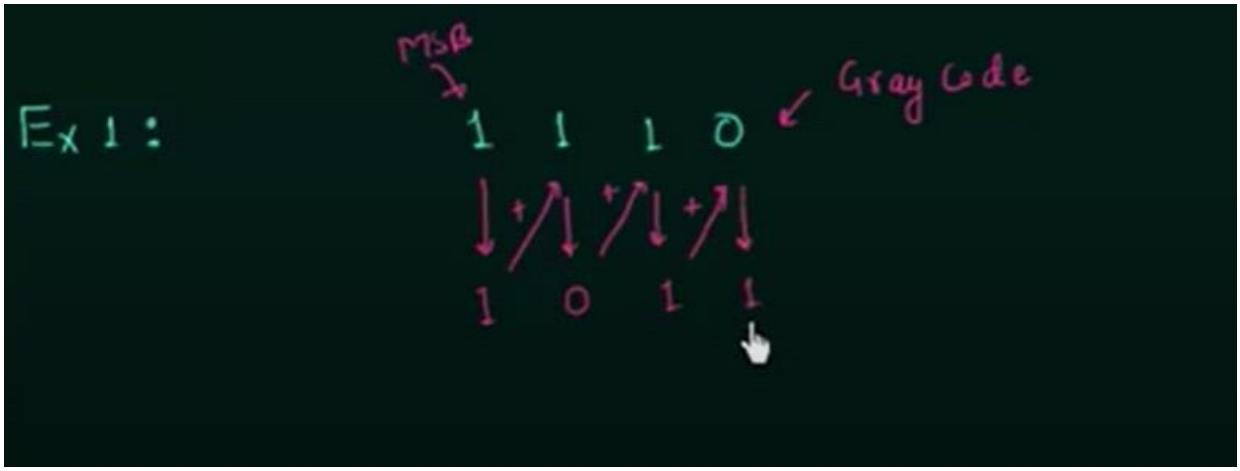
Place the MSB of gray as it is

Delete the MSB of gray

Repeat until no gray bits are left:

xor the last bit of binary with MSB of gray and append the result as gray bit

Delete the MSB of gray



Examples

26 January 2021 06:22 PM

DECIMAL NUMBER	BINARY CODE	GRAY CODE
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Addition

26 January 2021 06:27 PM

Binary Addition

It is a key for binary subtraction, multiplication, division. There are four rules of binary addition.

Case	A + B	Sum	Carry
1	0 + 0	0	0
2	0 + 1	1	0
3	1 + 0	1	0
4	1 + 1	0	1

In fourth case, a binary addition is creating a sum of (1 + 1 = 10) i.e. 0 is written in the given column and a carry of 1 over to the next column.

Example – Addition

0011010 + 001100 = 00100110

```
   1 1          carry
 0 0 1 1 0 1 0  = 26₁₀
+0 0 0 1 1 0 0  = 12₁₀
---------------
 0 1 0 0 1 1 0  = 38₁₀
```

Subtraction

26 January 2021 06:27 PM

Binary Subtraction

Subtraction and Borrow, these two words will be used very frequently for the binary subtraction. There are four rules of binary subtraction.

Case	A - B	Subtract	Borrow
1	0 - 0	0	0
2	1 - 0	1	0
3	1 - 1	0	0
4	0 - 1	0	1

Example – Subtraction

0011010 - 001100 = 00001110

```
          1 1        borrow
  0 0 1 1 0 1 0  = 26(10)
- 0 0 0 1 1 0 0  = 12(10)
-----------------
  0 0 0 1 1 1 0  = 14(10)
```

Multiplication

26 January 2021 06:29 PM

Binary Multiplication

Binary multiplication is similar to decimal multiplication. It is simpler than decimal multiplication because only 0s and 1s are involved. There are four rules of binary multiplication.

Case	A x B	Multiplication
1	0 x 0	0
2	0 x 1	0
3	1 x 0	0
4	1 x 1	1

Example – Multiplication

Example:

0011010 x 001100 = 100111000

$$
\begin{array}{rl}
 & 0\,0\,1\,1\,0\,1\,0 = 26_{10} \\
\times & 0\,0\,0\,1\,1\,0\,0 = 12_{10} \\
\hline
 & 0\,0\,0\,0\,0\,0\,0 \\
 & 0\,0\,0\,0\,0\,0\,0 \\
 & 0\,0\,1\,1\,0\,1\,0 \\
 & 0\,0\,1\,1\,0\,1\,0 \\
\hline
 & 0\,1\,0\,0\,1\,1\,1\,0\,0\,0 = 312_{10}
\end{array}
$$

Division

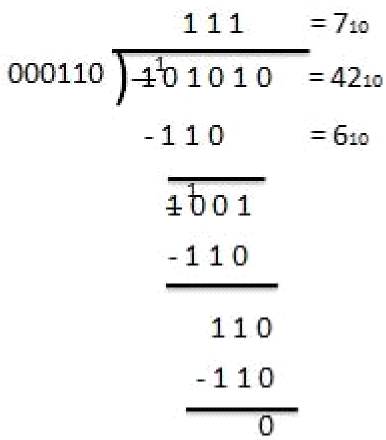
26 January 2021 06:30 PM

Binary Division

Binary division is similar to decimal division. It is called as the long division procedure.

Example − Division

101010 / 000110 = 000111



Basix-extended

26 January 2021 06:33 PM

Intersection of Sets

The intersection of two sets A and B is the set of elements which are in both sets A and B. The intersection of the two sets is written as A∩B.

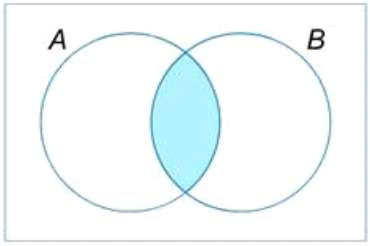


Figure 1.

Two sets are called disjoint if they have no elements in common.
Examples:

1. A={a,b,c}, B={k,ℓ,m}. These two sets are disjoint as they have no common elements. Their intersection is the empty set.
   A∩B={a,b,c}∩{k,l,m}=∅.
2. C={1,2,3,4}, D={2,4,6,7}. The intersection of these sets is
   C∩D={1,2,3,4}∩{2,4,6,7}={2,4}.

Union of Sets

The union of two sets A and B is defined as the set of elements which are either in set A or set B or in both A and B. This operation is denoted by the ∪ symbol.

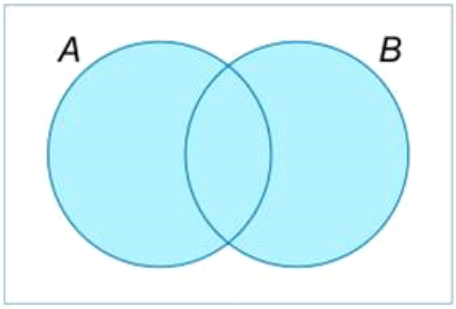


Figure 2.

Examples:

1. A={a,b,c}, B={k,ℓ,m}. The union of the two sets is given by
   A∪B={a,b,c}∪{k,l,m}={a,b,c,k,l,m}.
2. C={1,2,3,4}, D={2,4,6,7}. The union of the sets is given by
   C∪D={1,2,3,4}∪{2,4,6,7}={1,2,3,4,6,7}.

Principle of Inclusion-Exclusion

The cardinality of a finite set A, denoted by |A|, is equal to the number of

elements in it. The cardinality of the union of two finite sets A and B is given by the following relationship:
$|A \cup B| = |A| + |B| - |A \cap B|$,
where $|A \cap B|$ is the cardinality of the intersection of A and B.
The similar formula exists for the union of 3 finite sets:
$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$.

Difference of Two Sets

The difference of two sets A and B is the set that contains exactly all elements in A but not in B. The difference of two sets A and B is denoted by $A \setminus B$ or $A - B$.

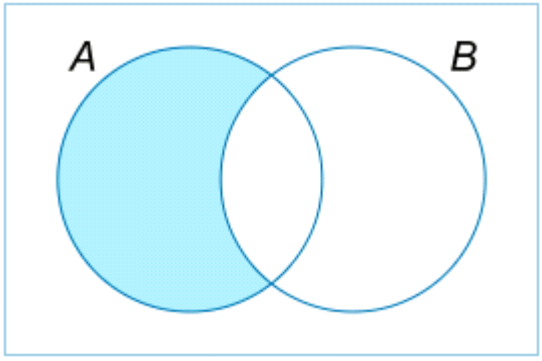


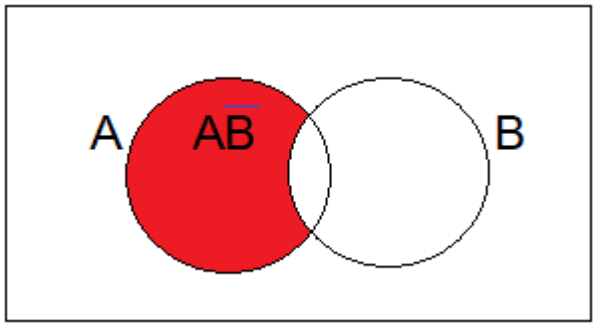
Figure 3.

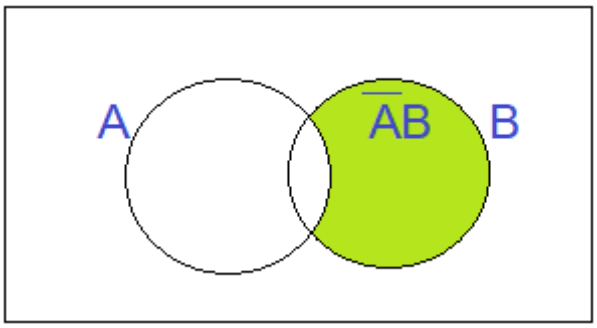
Examples:

1. $A=\{a,b,c\}$, $B=\{k,\ell,m\}$. The difference between two disjoint sets is equal to the initial set. So, we have
   $A \setminus B = A \setminus (A \cap B) = A \setminus \emptyset = A = \{a,b,c\}$.
2. $C=\{1,2,3,4\}$, $D=\{2,4,6,7\}$. The difference of two sets C and D is given by
   $C \setminus D = \{1,2,3,4\} - \{2,4,6,7\} = \{1,3\}$.

Symmetric Difference

The symmetric difference of two sets A and B is the set of all elements which belong to exactly one of the two original sets. This operation is written as $A \triangle B$ or $A \oplus B$.





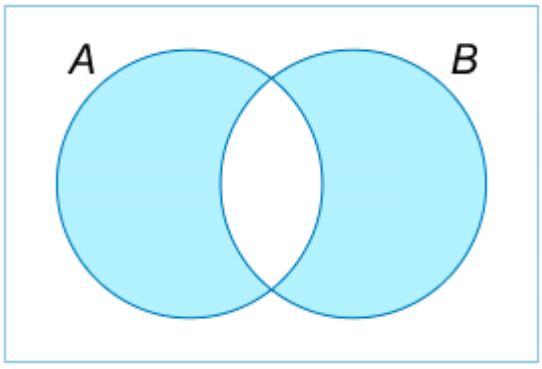


Figure 4. $A \oplus B = A'B + AB'$

In terms of unions and intersections, the symmetric difference of two sets A and B can be expressed as
$A \triangle B = (A \cup B) \setminus (A \cap B)$.
Examples:

1. $A=\{a,b,c\}$, $B=\{k,\ell,m\}$. The symmetric difference of two disjoint sets is equal to their union:
   $A \triangle B = (A \cup B) \setminus (A \cap B) = (A \cup B) \setminus \emptyset = A \cup B = \{a,b,c,k,\ell,m\}$.
2. $C=\{1,2,3,4\}$, $D=\{2,4,6,7\}$. The symmetric difference of the sets C and D is given by
   $C \triangle D = (C \cup D) \setminus (C \cap D) = \{1,2,3,4,6,7\} - \{2,4\} = \{1,3,6,7\}$.

Complement of a Set

The complement of a set A is the set of elements in the given universal set U that are not elements of A. The complement of A is denoted by A^c or A', or sometimes $\overline{A}$.

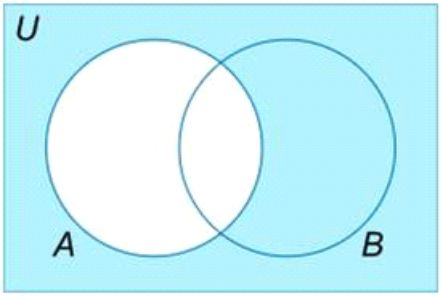


Figure 5.

So by definition, we have

$A^c = U \setminus A$.

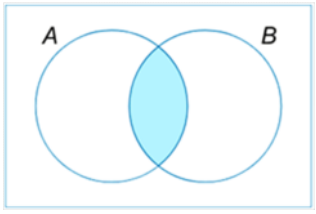
Examples:

1. Let the universal set be $U=\{a,b,c,d,e,f\}$. If $A=\{a,b,d,f\}$, then the complement of A is given by
   $A_C=U\setminus A=\{c,e\}$.
2. Suppose the universal set is $U=\{x\in Z|x^2<20\}$ and the set A is given by $A=\{x\in Z|-3\leq x<3\}$. Find the complement of A.
   $U=\{-4,-3,-2,-1,0,1,2,3,4\},A=\{-3,-2,-1,0,1,2\},\Rightarrow A_C=\{-4,3,4\}$.

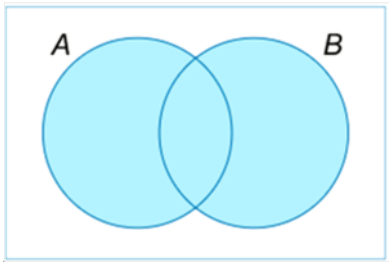
Basix-short

26 January 2021 06:35 PM

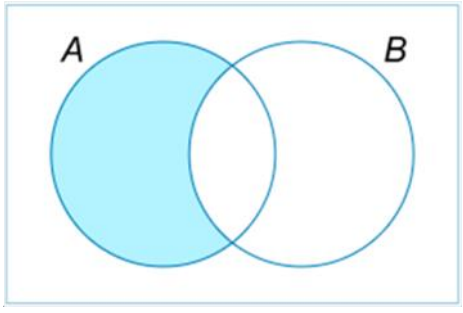
Intersection of Sets -> . (sign)



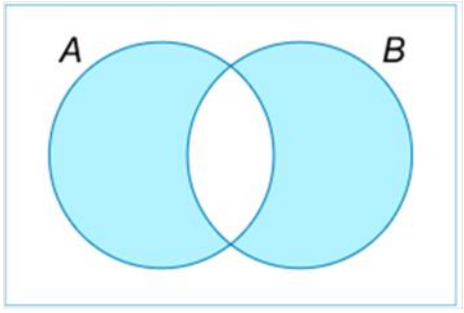
Union of Sets -> +



Difference of Two Sets -> - or/ (sign)



Symmetric Difference -> ⊕ (sign)



Statement

26 January 2021 06:45 PM

- Annulment Law
  - A . 0 = 0
  - A + 1 = 1
- Identity Law
  - A + 0 = A
  - A . 1 = A
- Idempotent Law
  - A + A = A
  - A . A = A
- Complement Law
  - A . A' = 0
  - A + A' = 1
- Commutative Law
  - A . B = B . A
  - A + B = B + A
- Double Negation Law
  - $\overline{\overline{A}}$=A
- De Morgan's Theorem
  - $\overline{A+B}=\overline{A}.\overline{B}$
  - $\overline{A.B}=\overline{A}+\overline{B}$
- Distributive Law
  - A(B + C) = A.B + A.C
  - A + (B.C) = (A + B).(A + C)
- Absorptive Law
  - A + (A.B) = (A.1) + (A.B) = A(1 + B) = A
  - A(A + B) = (A + 0).(A + B) = A + (0.B) = A
- Associative Law
  - A + (B + C) = (A + B) + C = A + B + C
  - A(B.C) = (A.B)C = A . B . C

My laws

- A+$\overline{A}.B=A+B$
- A xor B=A.B`+A`.B=(A+B).(A`+B`)
- A xnor B=A.B+A`.B`=(A+B`).(A`+B)

Postulates

26 January 2021 06:59 PM

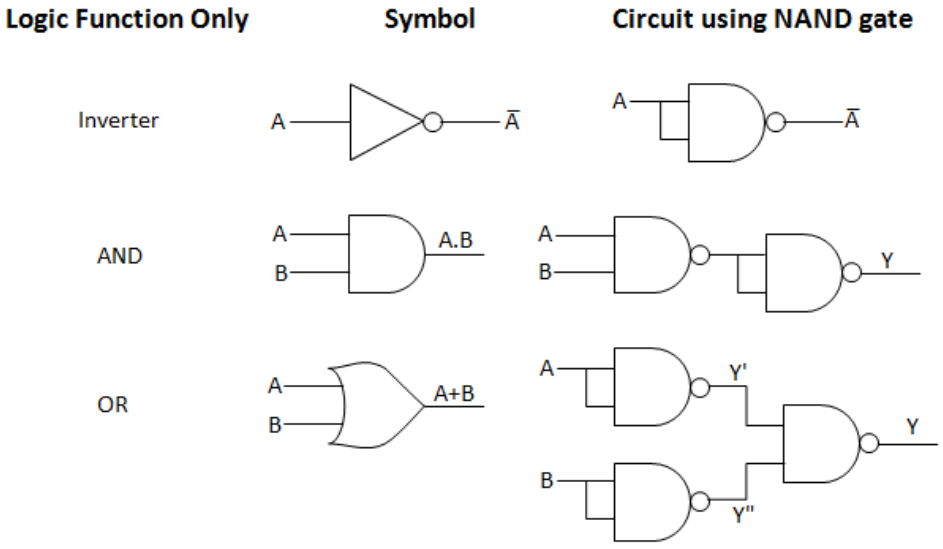
- 0 . 0 = 0
- 1 . 1 = 1
- 1 . 0 = 0
- 0 + 0 = 0
- 1 + 1 = 1
- 1 + 0 = 1
- 1’ = 0
- 0’ = 1

Universal gates

26 January 2021 07:06 PM

A universal gate is a gate which can implement any Boolean function without need to use any other gate type.

The NAND and NOR gates are universal gates.



Intro

26 January 2021 07:16 PM

The problem here is to find the function whose truth table is given

Two ways

Sop

Pos

Signs

+-> or

. -> and

Min term and max terms

When variables are all added that sum is called max term

Eg : A+B+C

When variables are all multiplied that product is called min term

Eg : A.B.C

Three variable min term m_4 represents the fifth column of a truth table 1 0 0 same goes for max term

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Relation between min and max term

$M_3 = \bar{m}_3$

$M_3 = (\bar{A}BC)'$

$M_3 = A + \bar{B} + \bar{C}$

SOP

26 January 2021 07:35 PM

If a function is given as a sum of min terms it is sop form

$$F = \sum(m_3, m_5, m_6, m_7)$$

Step 1:

Construct a truth table placing 1 at given min term position and 0 at other

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Note that bit string ABC represents decimal values from 0 to 7

Step 2:

Write the slandered canonical SOP form (sum of min terms where the function F is high)

F = A'BC + AB'C + ABC' + ABC

Note the value 0 is taken as complement of variable as we need high for those terms

1, the variable as it is (0 -> A` 1 - > A)

Note we don't actually require to write truth table with expertise we can directly produce the min term say m_3 just convert 3 to binary 0 1 1 => A`BC

Step 3:

Minimize the canonical form using laws

F = AB + BC + AC

This is minimized SOP form

POS

26 January 2021 09:46 PM

If a function is given as a product of max terms it is sop form

$$F = \prod (M_0, M_1, M_2, M_4)$$

Step 1:

Construct a truth table placing 1 at given min term position and 0 at other

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Note that bit string ABC represents decimal values from 0 to 7

Step 2:

Write the slandered canonical POS form (product of max terms where the function F is low)

F = (A + B + C) (A + B + C ‘) (A + B’ + C) (A’ + B + C)

Note the value 1 is taken as complement of variable as we need low for those terms

0, the variable as it is (1 -> A` 0 - > A)

Note we don't actually require to write truth table with expertise we can directly produce the min term say M_2 just convert 3 to binary 0 1 0 => A + B’ + C

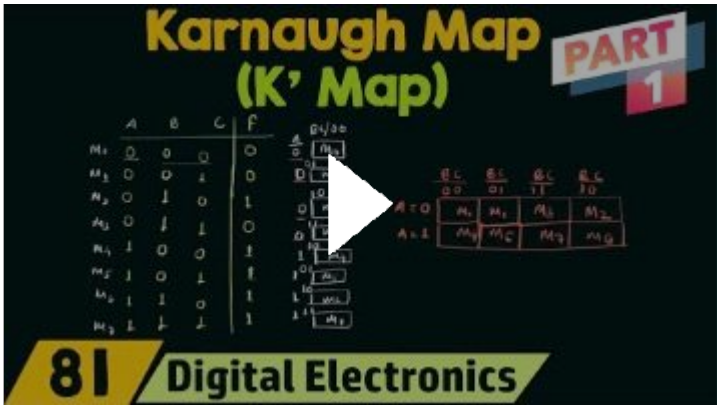
Step 3:

Minimize the canonical form using laws

Karnaugh Map

26 January 2021 10:26 PM

[Karnaugh Map (K' Map) - Part 1]
Very good resource



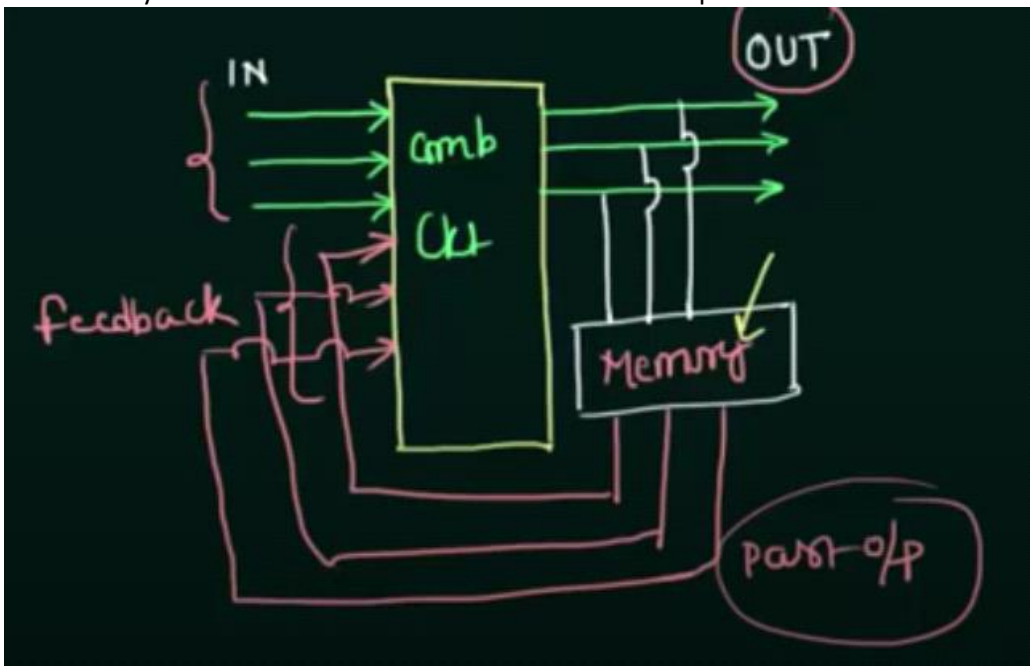
Basix

04 February 2021 03:52 PM

Definition : in sequential circuits the present output depends upon the present input as well as the previous output or outputs.
example flip flop, latch, registers.

In contrast a combinational circuit is a circuit in which the present output depends upon only the present input.
Example adder, subtractor.

Since a sequential circuit requires the previous output or outputs it has to store the current output in a memory element which can later be fed back to the input.

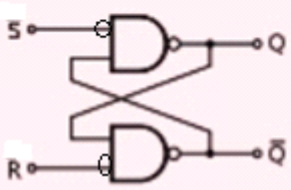


Flip flop/latch

04 February 2021 06:09 PM

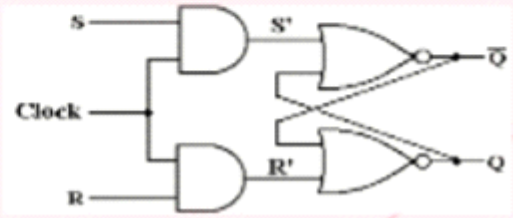
Latches..

- Both are same but there is a little difference between both.
- Latches are the building blocks of sequential circuits.
- latches can be built from gates.
- latch does not have a *clock signal.*



Flip Flop..

- flip-flops are also the building blocks of sequential circuits.
- Flip-flops can be built from latches.
- A flip-flop always has a *clock Signal*



What are flip flops.

In electronics, a **flip-flop** or **latch** is a circuit that has two stable states and can be used to store state information. A flip-flop is a bistable multivibrator. The circuit can be made to change state by signals applied to one or more control inputs and will have one or two outputs. It is the basic storage element in sequential logic.

Flip-flops maintain their state indefinitely until an input pulse called a trigger is received. When a trigger is received, the flip-flop outputs change state according to defined rules and remain in those states until another trigger is received.

Uses of flip flops.

- Flip flop and latches are the circuits that can store and remember information. They're the kind of circuits that are used in computers to store program information like RAM memory and Registers.
- Flip-flops can be used to store one bit, or binary digit, of data. The data may represent the state of a sequencer, the value of a counter, an ASCII character in a computer's memory or any other piece of information.

LATCH	FLIP – FLOP
Latches do not require clock signal.	Flip – flops have clock signals
A latch is an asynchronous device.	A flip – flop is a synchronous device.
Latches are transparent devices i.e. when they are enabled, the output changes immediately if the input changes.	A transition from low to high or high to low of the clock signal will cause the flip – flop to either change its output or retain it depending on the input signal.
A latch is a Level Sensitive device (Level Triggering is involved).	A flip – flop is an edge sensitive device (Edge Triggering is involved).
Latches are simpler to design as there is no clock signal (no careful routing of clock signal is required).	When compare to latches, flip – flops are more complex to design as they have clock signal and it has to be carefully routed. This is because all the flip – flops in a design should have a clock signal and the delay in the clock reaching each flip – flop must be minimum or negligible.
The operation of a latch is faster as they do not have to wait for any clock signal.	Flip - flops are comparatively slower than latches due to clock signal.
The power requirement of a latch is less.	Power requirement of a flip – flop is more.
A latch works based on the enable signal.	A flip – flop works based on the clock signal.

1. RS flip flop.

- **DEFINITION:**

"A circuit containing cross coupled connection, which is used to remain a memory state stable by using asynchronous sequential circuits, is called direct-coupled or RS flip flop."

- **EXPLANATION:**

Each flip flop has two inputs and two outputs. The inputs are denoted by R and S and outputs are denoted by Q and Q'. These circuits can be implemented through "NOR" and "NAND" gate

How the above circuit works:

Memory State:

In above circuit when two inputs are given as S=0 and R=0 then the memory remains stable.

Re-set state:

When inputs are given as S=0 and R=1, then the output comes Q=0 AND Q'=1.

("WHEN Q' comes 1 this state is called RE-SET state")

Set state:

When inputs are given as S=1 and R=0, then the output comes Q=1 AND Q'=0.

("WHEN Q comes 1 this state is called SET state")

Not Allowed state:

When the outputs come against the inputs then this state is called not allowed.

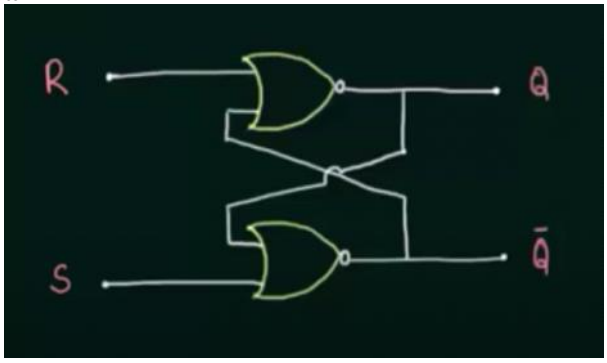
SR latch

04 February 2021 04:59 PM

There are 2 types of SR latch

- Nor latch
- Nand latch

Nor latch



S stands for set
R stands for reset
Truth table for nor gate

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

Case I :

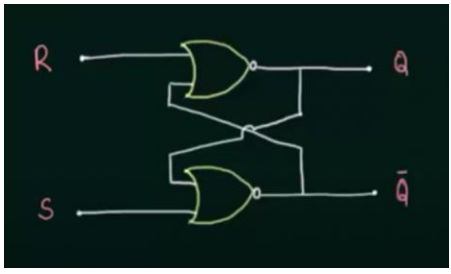
S=0 , R=1

- When R is high Q will be low irrespective what the second output is (Q`)
- A low Q makes the first input of lower nor gate low and s is already low so Q` is high
- The circuit is sustained and we get Q = 0; Q` = 1
- Reset high => Q low

Now when we remove inputs

S=0, R=0

- Input of upper nor gate is 0,1 so Q remains low
- Input of lower nor gate is 0,0 so Q` remains high
- Hence even when the input is removed it sustains its state (memory condition)



Case II :

S=1 , R=0

- When S is high Q` will be low irrespective what the second output is (Q)
- A low Q` makes the second input of upper nor gate low and R is already low so Q is high
- The circuit is sustained and we get Q = 1; Q` = 0
- Set high => Q high

Now when we remove inputs

S=0, R=0

- Input of upper nor gate is 0,0 so Q remains high
- Input of lower nor gate is 0,1 so Q` remains low
- Hence even when the input is removed it sustains its state (memory condition)

Case III :

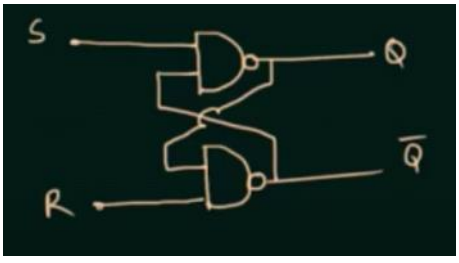
S=1 , R=1

proceeding the same way as above first with lower nor then with upper nor gives conflicting result
Also this state does not attain memory condition.
So it is undefined

Truth table of nor SR latch

S	R	Q	Q`
0	1	0	1
1	0	1	0
0	0	Prev. state	Prev. State
1	1	undefined	undefined

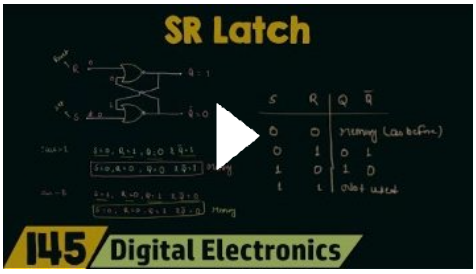
Nand SR latch



Every thing is same except the gates and S R point reversal
Truth table of nand SR latch

S	R	Q	Q`
0	1	1	0
1	0	0	1
0	0	undefined	undefined
1	1	Prev. state	Prev. state

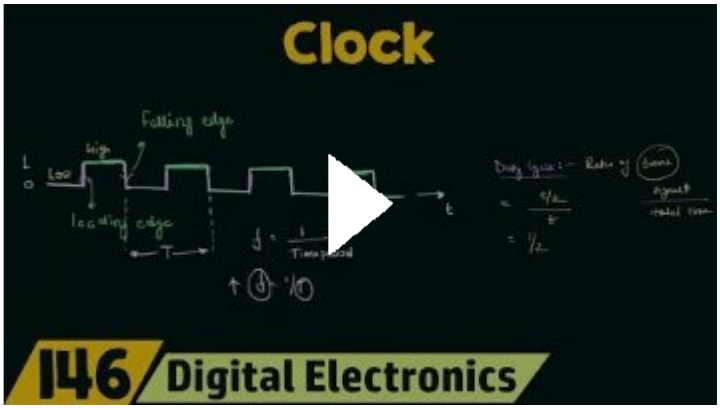
Resources:
SR Latch | NOR and NAND SR Latch



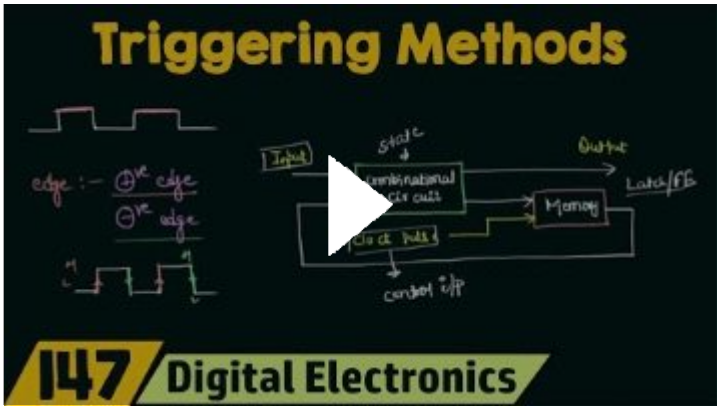
Clock

04 February 2021 06:08 PM

[What is a Clock?]



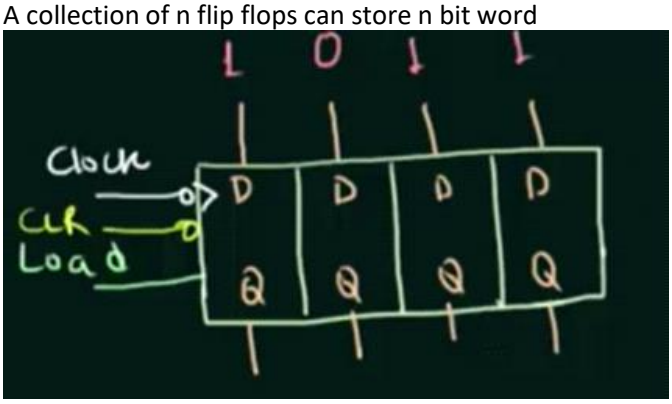
[Triggering Methods in Flip Flops]



Registers

05 February 2021 11:28 AM

Registers are nothing but multiple flip flops for multi bit storage
A collection of n flip flops can store n bit word



In the given block diagram this register is made up of D flip flop the Clock lines are internally connected
the individual D lines store one bit of data each and the Clr line clears all the data
since typically the frequency of the Clock is very high and as soon as the Clock goes high again the previous data will be overwritten with the current data on the bus which may be unwanted
hence we have a load line when the load line is high and the Clock is high the register writes the data else it does not

there are 2 types of registers in this way

- synchronous : when load and Clock both are used as control lines for writing data
- asynchronous : when only load line is used as control line for writing data .

Data formats

There are 2 kinds of data formats

- serial format /temporal code - in this format a single data line is used to insert and bit data depending on the Clock signals,
  for example if the data is 1011 then for the first Clock cycle the detail line is high then and for the next Clock cycle the deadline is low then it's high for the next 2 Clock cycles.

- Parallel format /special code - in this format the n bit data is carried via n data lines

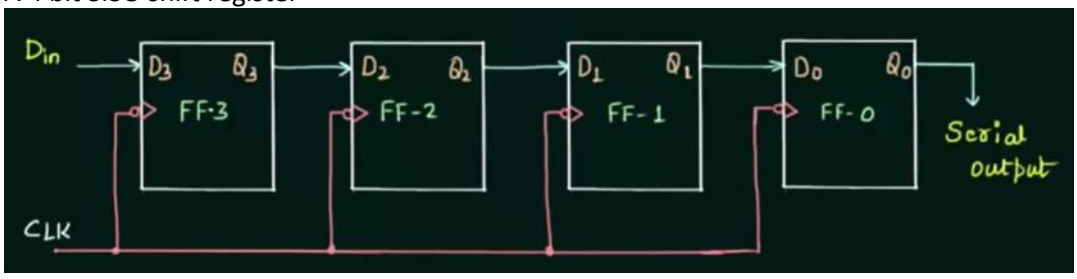
classification of Registers

- based on input and output
  - SISO (serial input serial output)
  - SIPO (serial input parallel output)
  - PIPO (parallel input parallel output)
  - PISO (Parallel input serial output)
- hey based on application
  - shift register - when data is shifted from one flip flop to another until it outputs .
  - storage register - when data is directly stored and retrieved without being transferred from one flip flop to another .

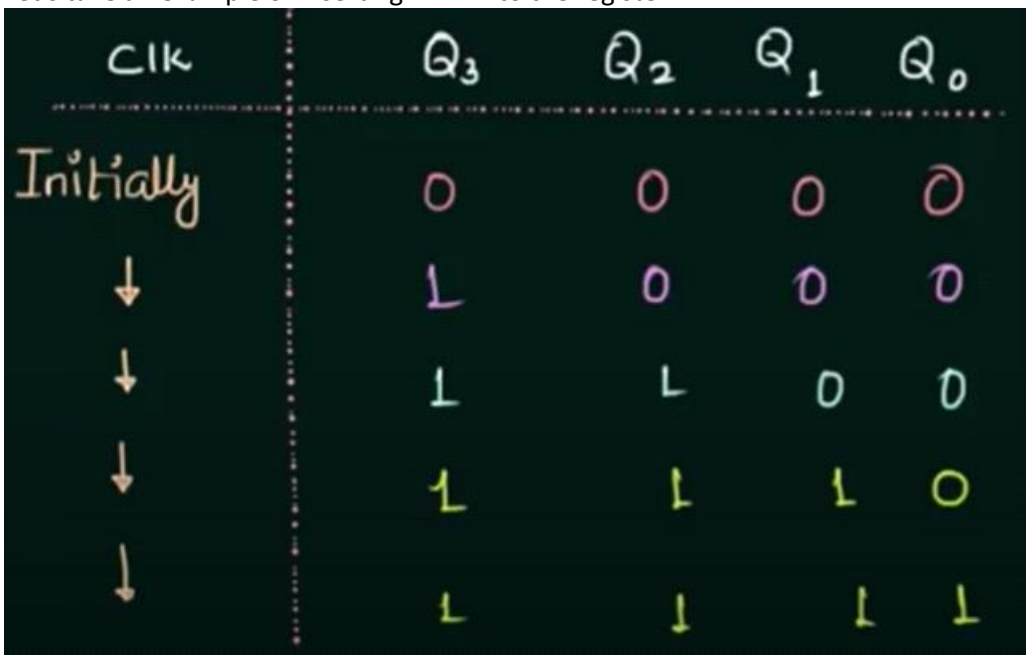
SISO shift register

05 February 2021 04:31 PM

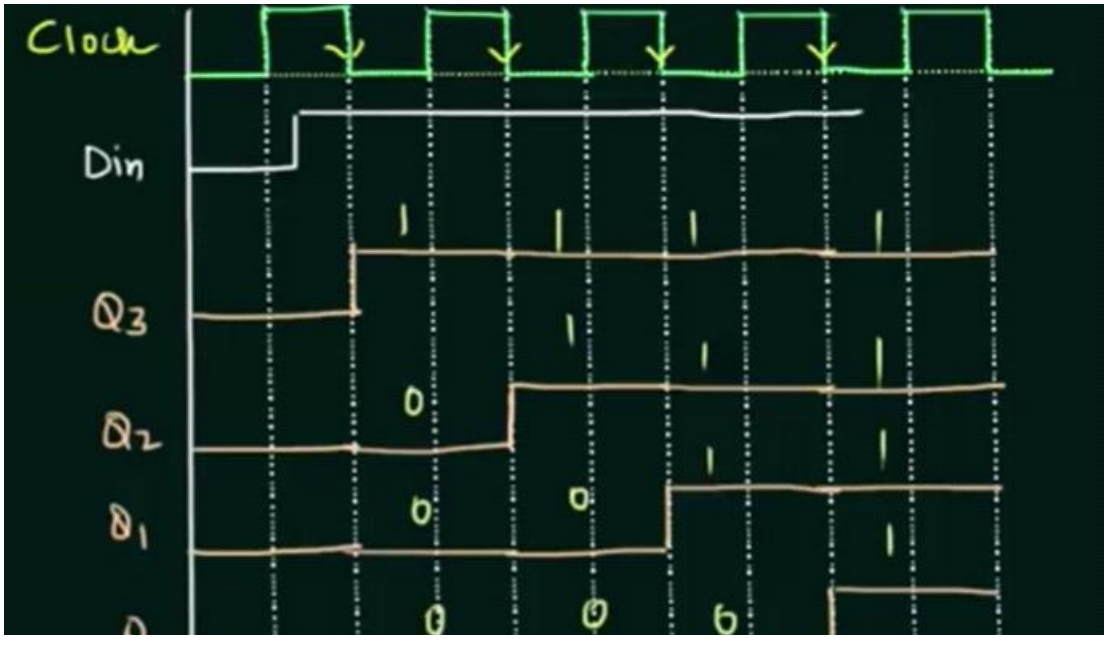
A 4 bit SISO shift register



This is an example of edge triggered Clock where the registers become active with negative edge triggering
Data is inserted from LSB to MSB
N clock pulses are required to store n bit data
Let's take an example of inserting 1111 into the register



The figure above shows the outputs of each flip flop at the end of successive negative edge triggers



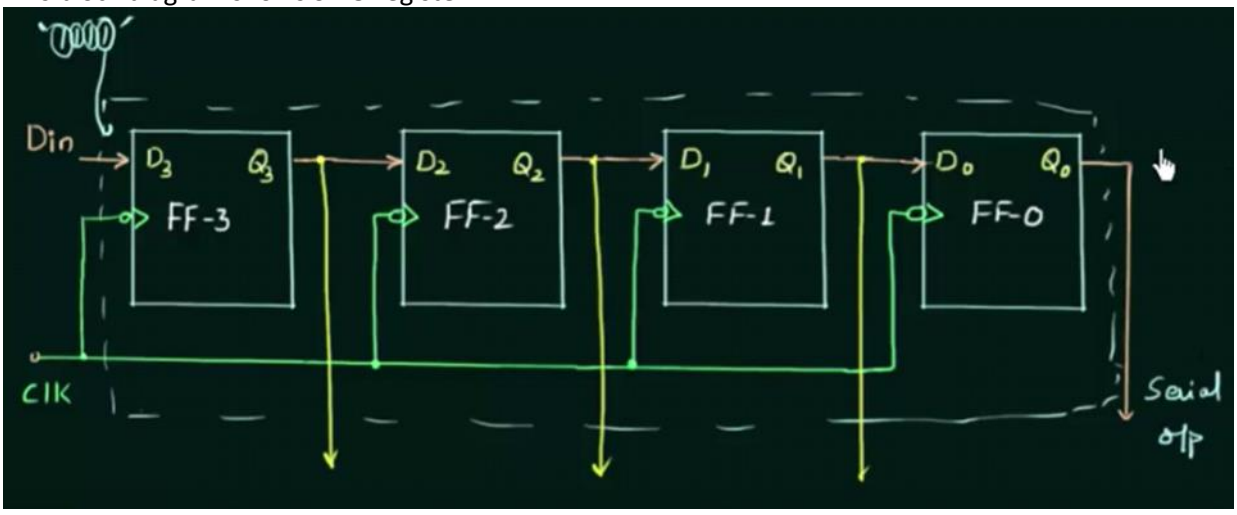
The figure above shows the signals with respect to time
at each successive negative edge trigger a bit shift take place (towards right)

[Shift Register (SISO Mode)]

SIPO and PIPO

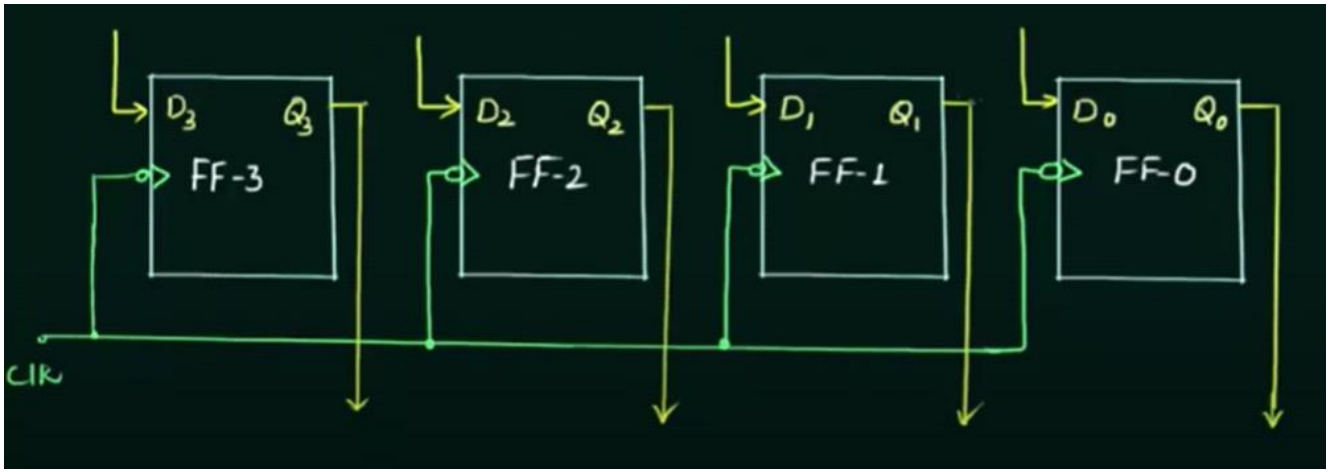
05 February 2021 04:44 PM

This block diagram shows SIPO register.



the input is serial as in SISO but the output is parallel
this means that at the end of 4 Clock pulses the data is ready to be output through the 4 output lines
However in the SISO mode hey it requires a total of 8 Clock pulses for the output

PIPO



Aka

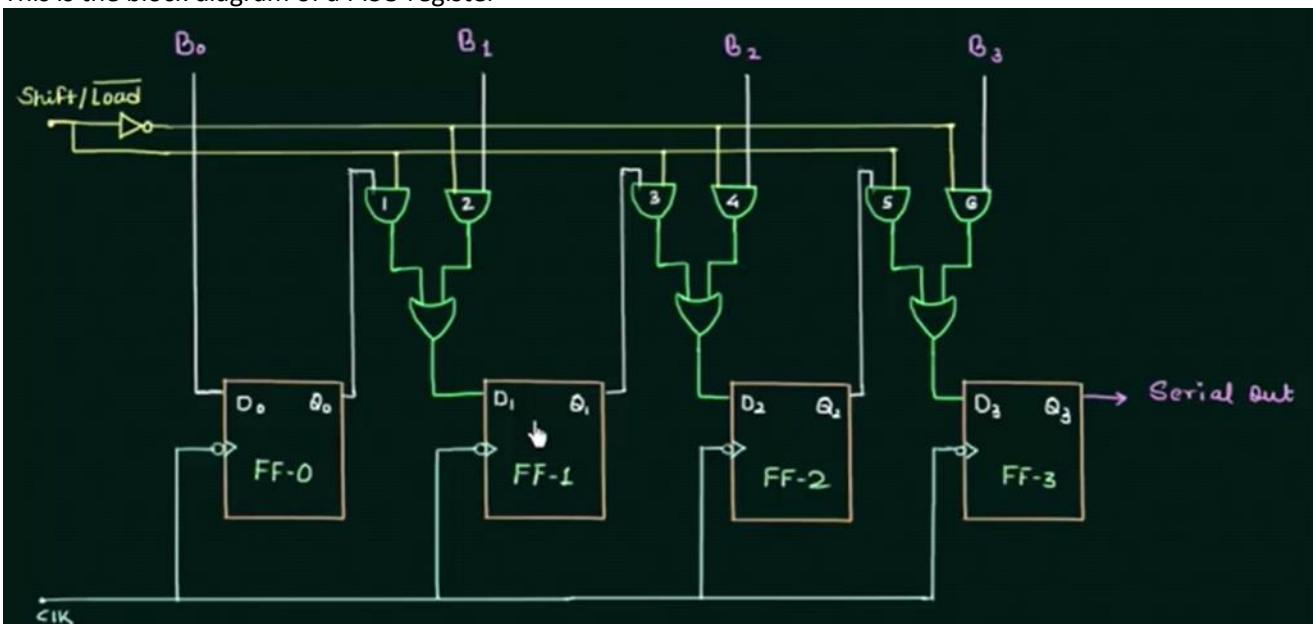
- Storage register
- Buffer register

Every thing is self explanatory here
We require a single clock pulse to store n bit data

PISO

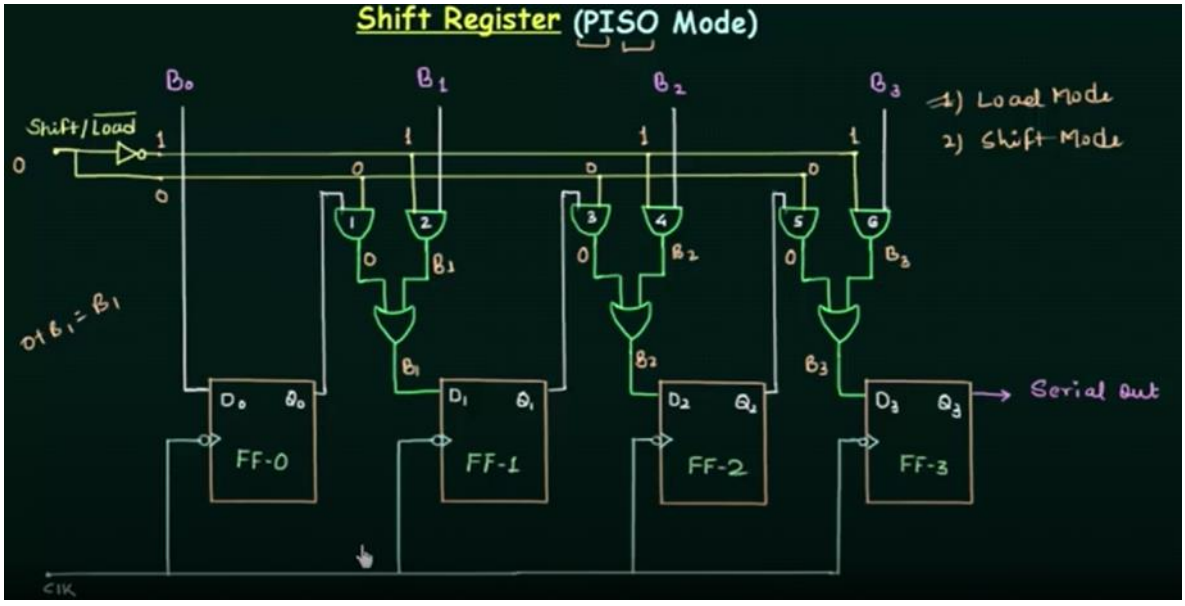
05 February 2021 04:55 PM

This is the block diagram of a PISO register



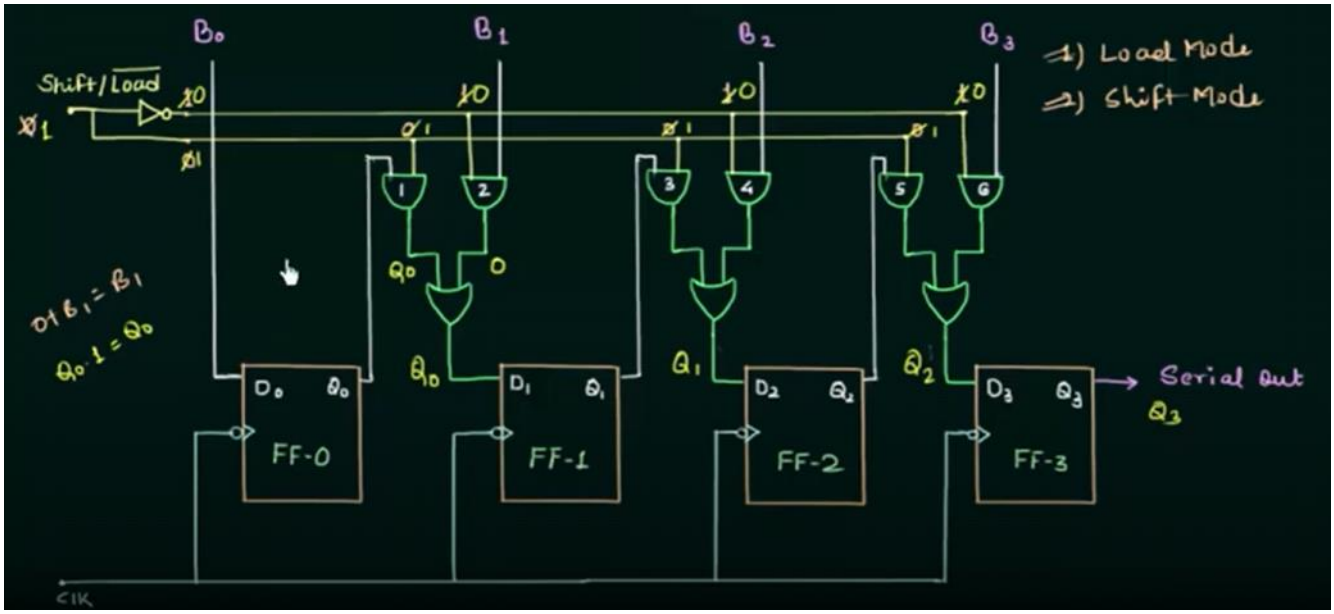
It works in two mode

- load mode



the load line is active when it is zero or low
the data is stored parallelly as shown as in PIPO

- Shift mode

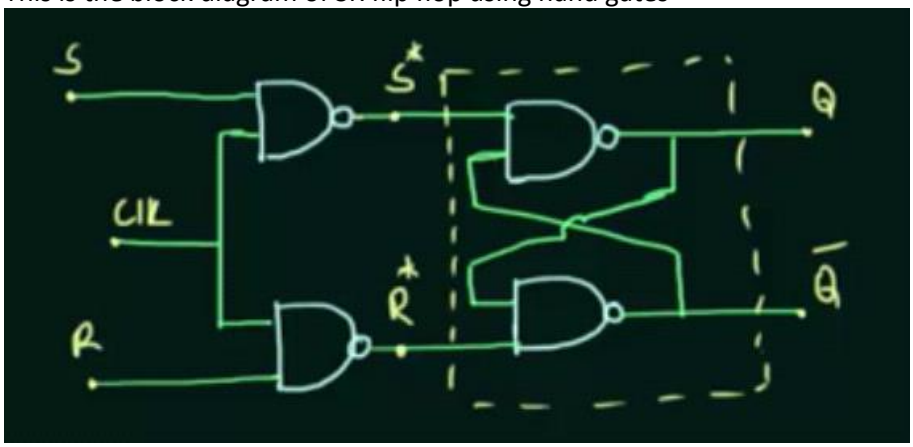


The shift mode is active when the load line is high
the working as shown in the figure as in SISO

SR Flip flop

04 February 2021 06:10 PM

This is the block diagram of SR flip flop using nand gates



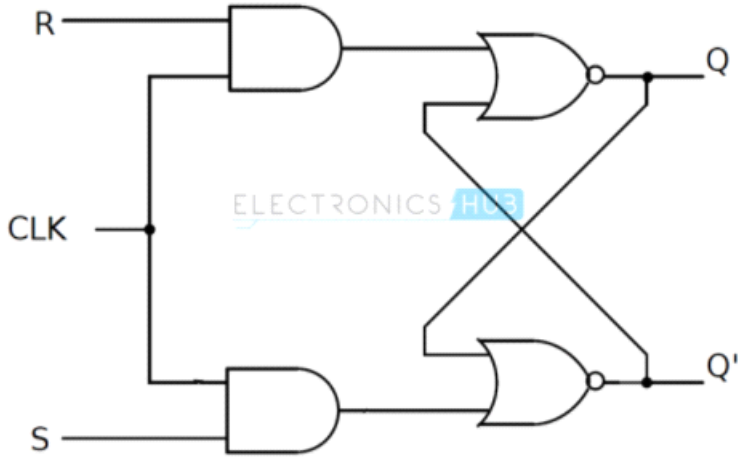
The part inside the dotted block is the SR nand latch

S* = $\overline{\text{clk. S}}$
R* = $\overline{clk.R}$
Truth table

Clk.	S	R	Q	Q`
0	whatever	whatever	Prev. state	Prev. state
1	0	0	Prev. state	Prev. state
1	0	1	0	1
1	1	0	1	0
1	1	1	Undefined	Undefined

so we see the T.T of SR flip flop using nand gate is almost same as SR latch using nor gate
Some times also called as clocked flip flop

This is the block diagram of SR flip flop using nor gates



S* = Clk . S
R* = Clk . R .... Using previous analogy

Truth table

Clk.	S	R	Q	Q`
0	whatever	whatever	Prev. state	Prev. state
1	0	0	Prev. state	Prev. state
1	0	1	0	1
1	1	0	1	0
1	1	1	Undefined	Undefined

so we see the T.T of SR flip flop using nor gate also is almost same as SR latch using nor gate
Some times also called as clocked flip flop

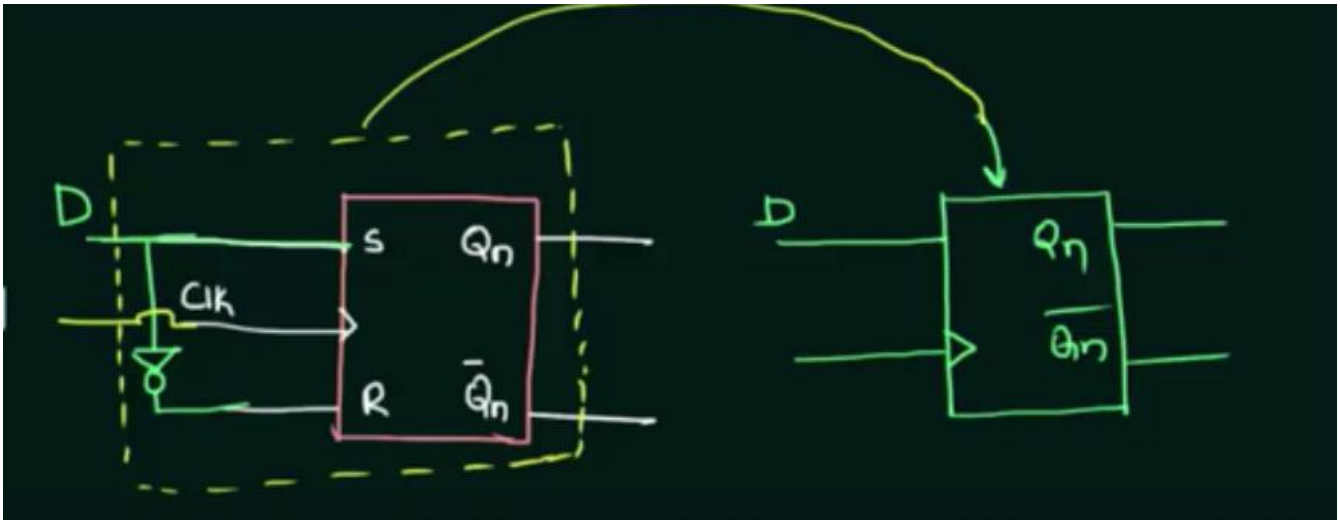
Other types of flip flop

05 February 2021 09:45 AM

D - flip flop

D stands for data

In this type of flip flops whatever data we want to store zero or one we just give it in the data line while the Clock is active and then deactivate the Clock the data will be stored .



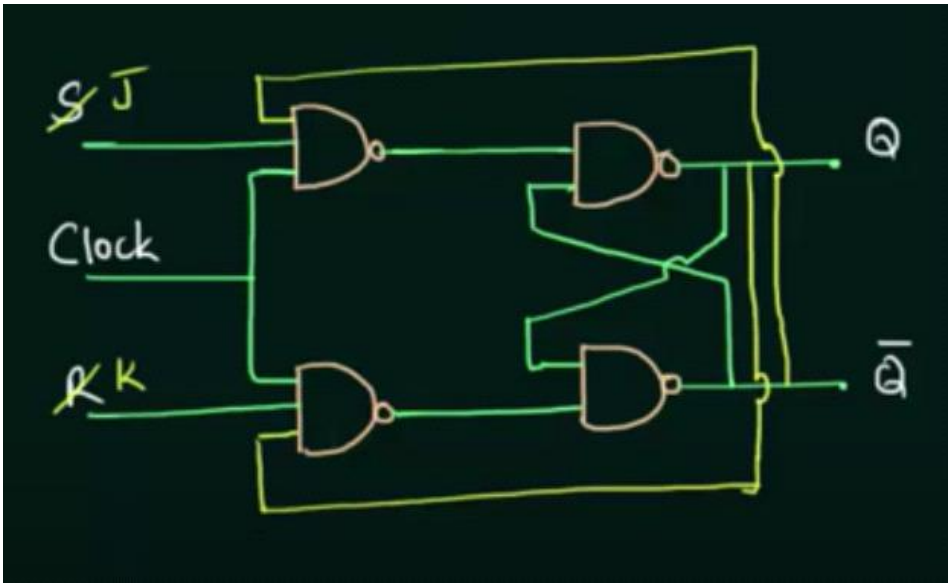
This figure shows how we can make a D flip flop with an Sr flip flop.

T.T of d flip flop

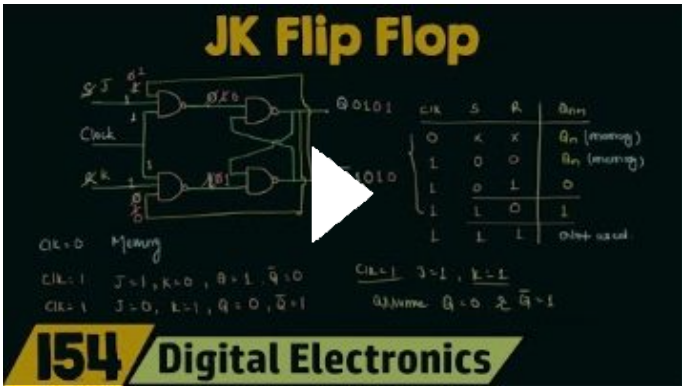
Clk	D	Qn+1
0	x	Qn
1	0	0
1	1	1

We can not have s=1 R=1 in this flip flop

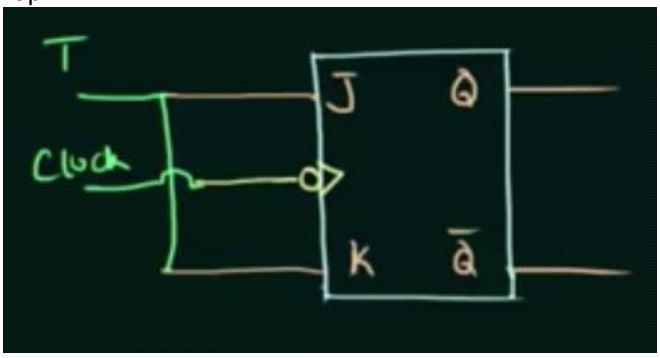
JK flip flop



The working is same as SR flip flop
But when J=K=1 the output Q is 101010 racing or toggles
Introduction to JK flip flop



T flip flop

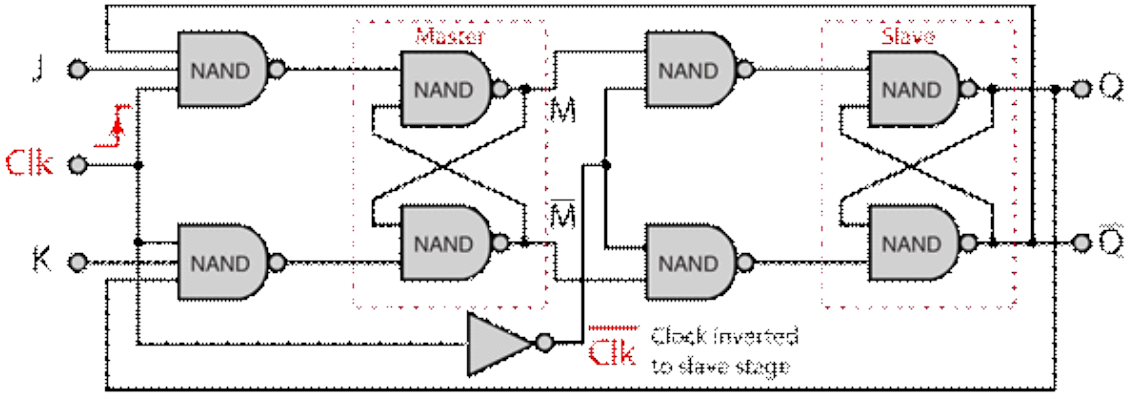


Here t stands for toggle
T.T for T flip flop

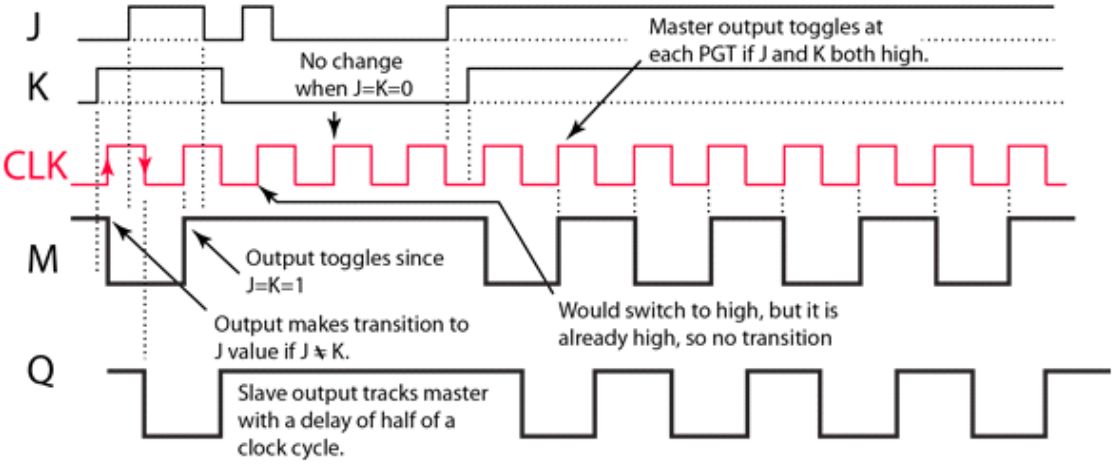
Clk	T	Q_{n+1}	
0	X	Q_n	(memory)
1	0	Q_n	(memory)
1	1	$\overline{Q}_n$	(toggling)

JK master slave flip flop

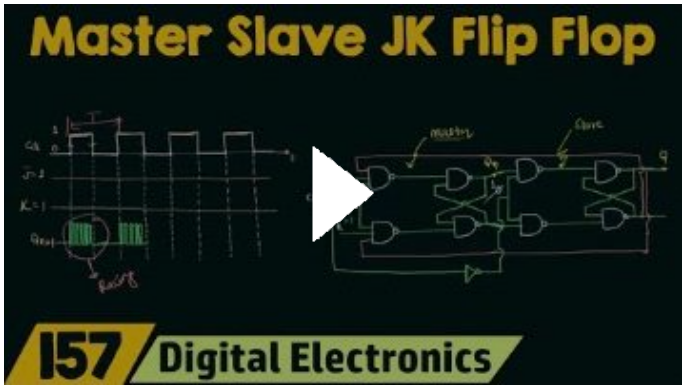
- The Master-Slave JK Flip Flop has two gated SR flip flops used as latches in a way that suppresses the "racing" or "race around" behavior. Another way to look at this circuit is as two J-K flip-flops tied together with the second driven by an inverted clock signal.



- When the clock makes a positive transition the master section is triggered but the slave section is not because its clock is inverted. At a half cycle of the clock, on the downward transition, the inverted clock has a positive transition and triggers the slave section. The final output Q then tracks the output of the master section M after a half cycle of the clock.



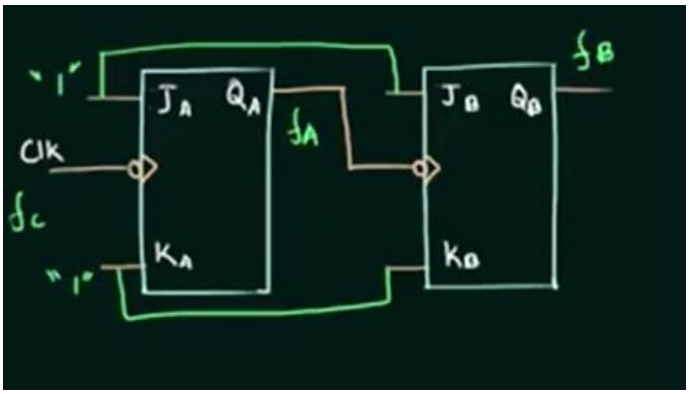
Master Slave JK Flip Flop


Master Slave JK Flip Flop
157
Digital Electronics

Counters

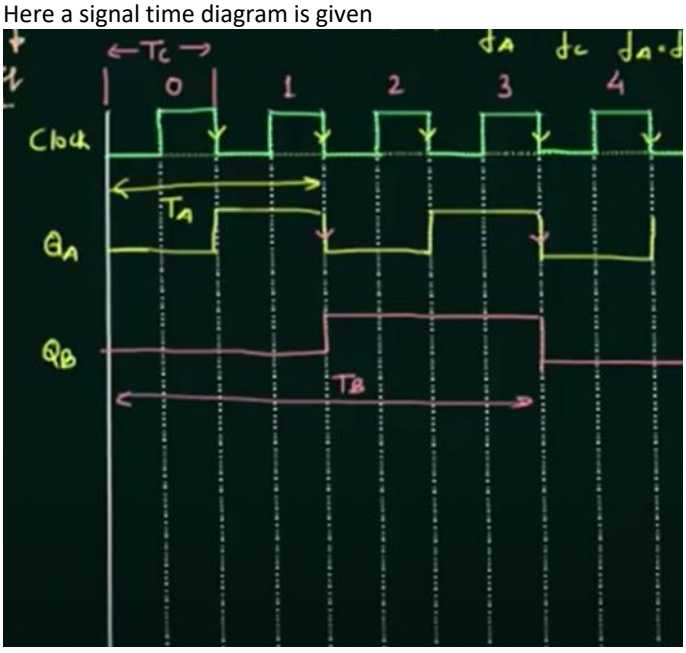
05 February 2021 05:25 PM

This is basic counter



This is a negative edge triggered flip flops

Here a signal time diagram is given



We have the following formula

Final frequency = initial frequency/2^n n is the nimber of flip flops

Working as a counter

count	Qb	Qa
0	0	0
1	0	1
2	1	0
3	1	1
4	0	0

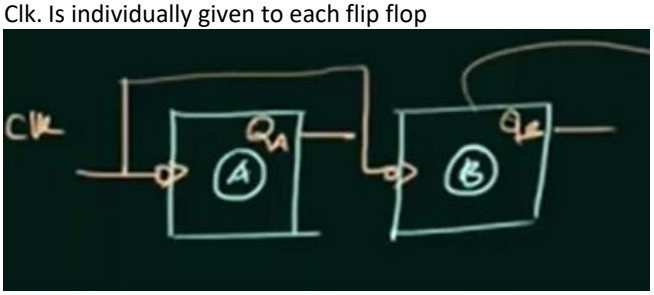
So this counter counts up to 3

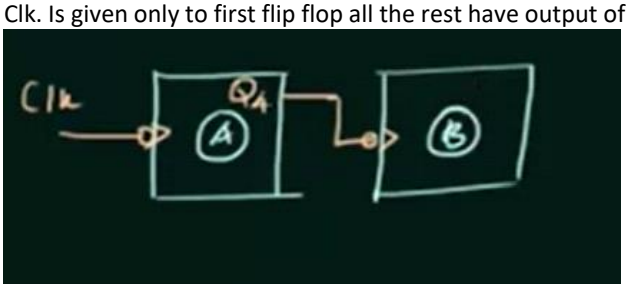
Increments at each pulse

Classification of counters

Two kinds of counters

- Synchronous counters

- Clk. Is individually given to each flip flop
- 

- Asynchronous counters
  - Clk. Is given only to first flip flop all the rest have output of previous flip flop as clock
  - 

Comparison

Asynchronous/Ripple Counter	Synchronous Counter
1. Flip flops are connected in such a waythat the o/p of first flip flop drives the clock of next flip flop.	1. There is no connection between o/p of first flip flop and clock of next flip flop.
2. Flip flops are not clocked simultaneously.	2. Flip flops are clocked simultaneously.
3. Circuit is simple for more number of states.	3. Circuit becomes complicated as number of states increases.
4. Speed is slow as clock is propagated through number of stages	4. Speed is high as clock is given at a same time.

Each of the two counters is further classified into 3 other types

- Up counter count straight 0123456
- Down counter count reverse 6543210
- Up/down counter a combination of both

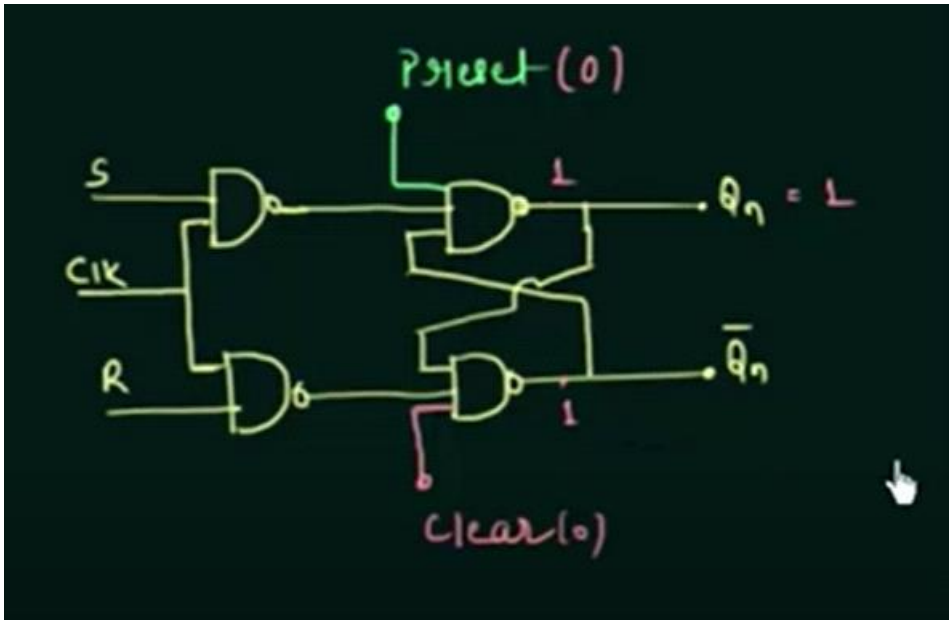
Overriding inputs

05 February 2021 11:36 PM

These are the input lines to flip flop which decides its state irrespective of normal inputs or normal inputs are deactivated.
There are two overriding line preset and clear.
Aka

- Asynchronous i/ps
- Direct inputs

S,R,J,K,D,T are called synchronous i/ps



preset	Clear	Q
0	0	Not used
0	1	1
1	0	0
1	1	Normal flip flop

Hence we can say preset and clear does their job when low.

Ring counter

05 February 2021 11:43 PM

Ring Counter

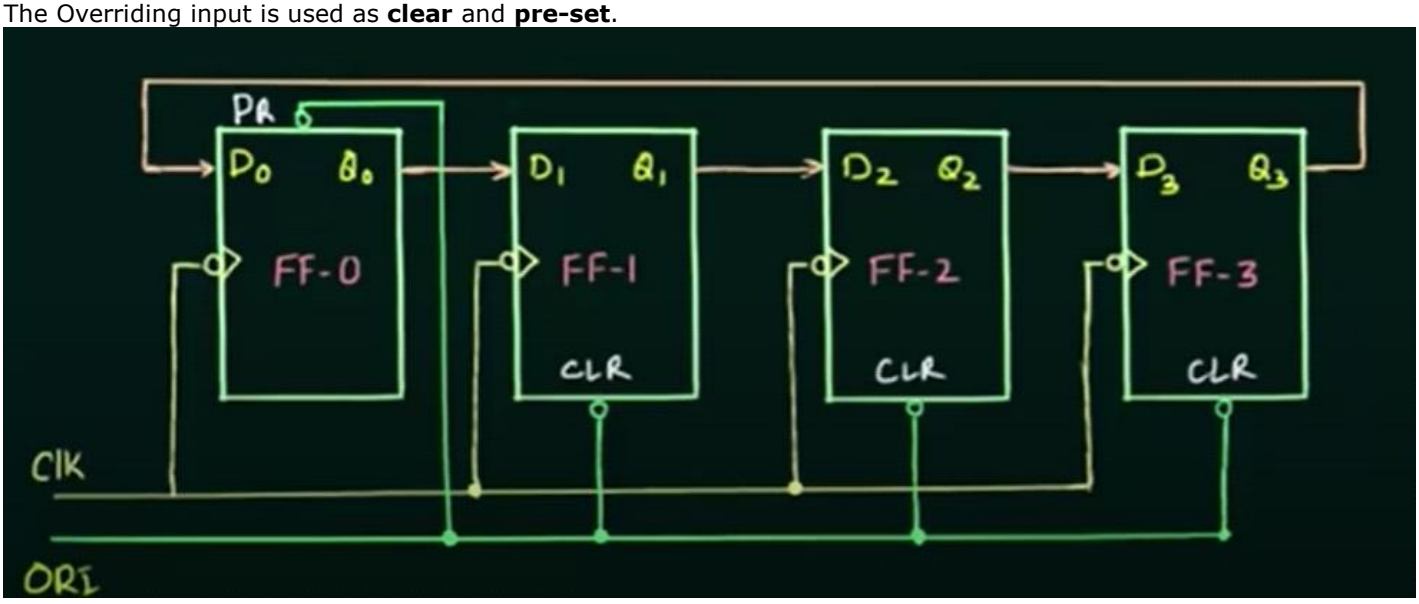
A ring counter is a special type of application of the **Serial IN Serial OUT** Shift register. The only difference between the shift register and the ring counter is that the last flip flop outcome is taken as the output in the shift register. But in the ring counter, this outcome is passed to the first flip flop as an input. All of the remaining things in the ring counter are the same as the shift register.

In **the Ring counter**

No. of states in Ring counter = No. of flip-flop used

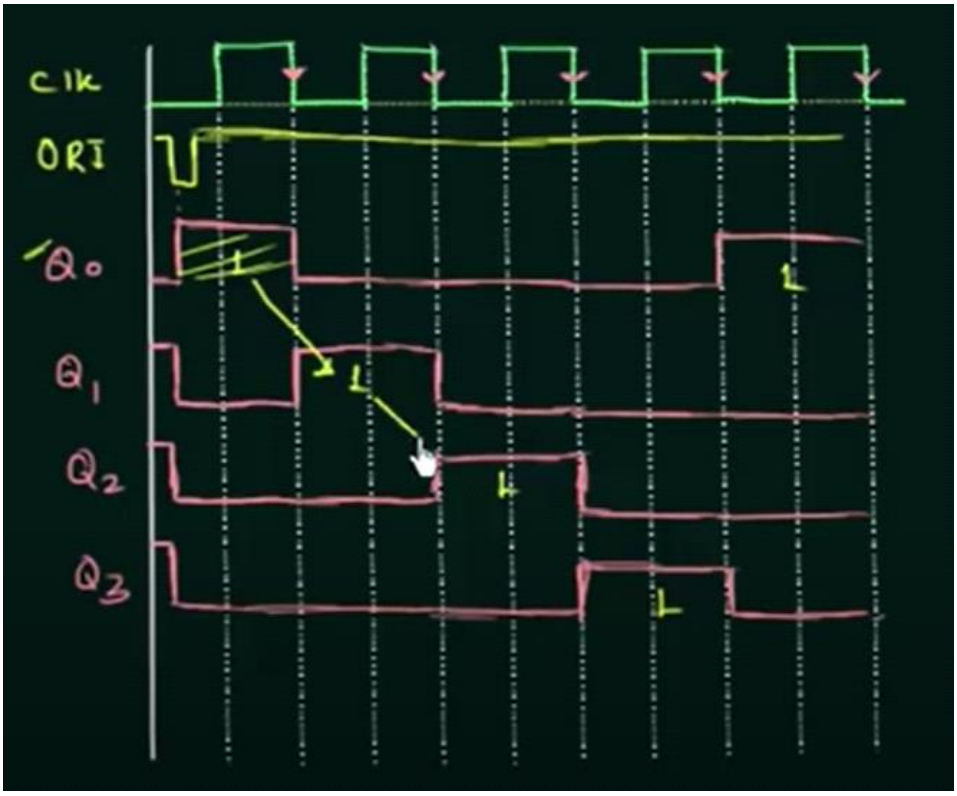
Below is the block diagram of the 4-bit ring counter. Here, we use 4 **D flip flops**. The same clock pulse is passed to the clock input of all the flip flops as a synchronous counter. The **Overriding input (ORI)** is used to design this circuit.

The Overriding input is used as **clear** and **pre-set**.



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

Signal diagram


CIK
ORI
Q0
Q1
Q2
Q3

Johnson's counter

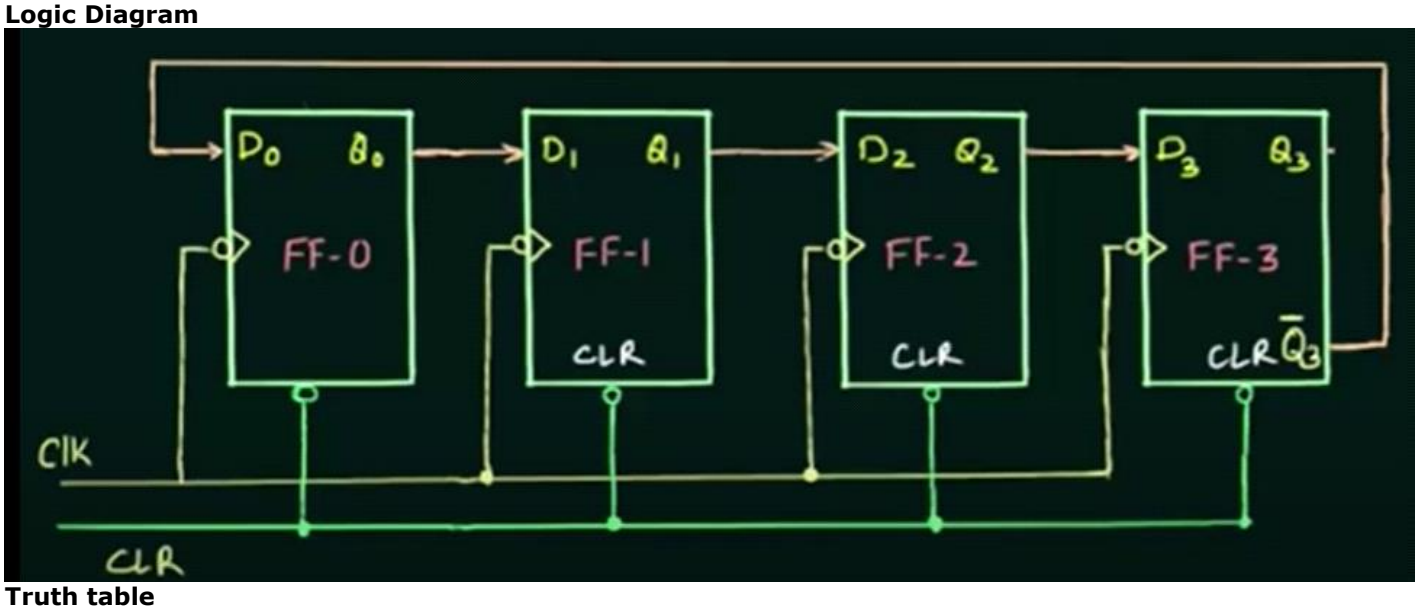
05 February 2021 11:51 PM

Twisted Ring Counter

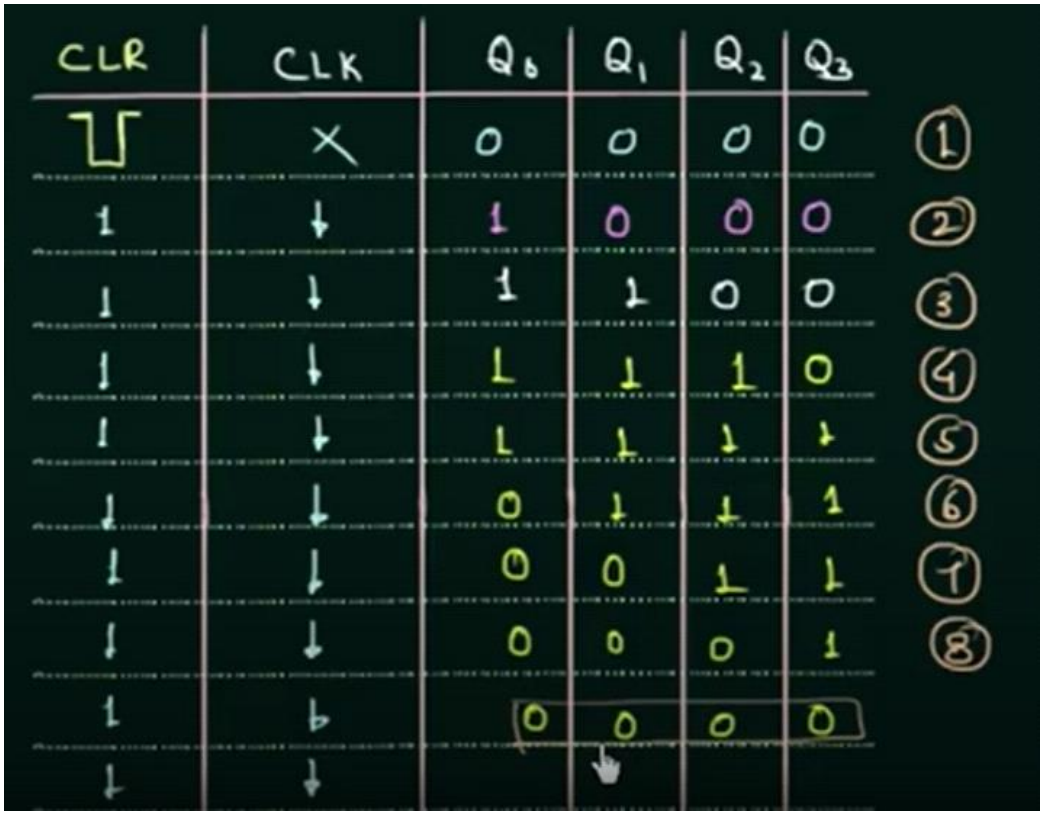
Another name of Johnson counter are: creeping counter, twisted ring counter, walking counter, mobile counter and switch tail counter.

The **Twisted Ring Counter** refers to as a **switch-tail ring Counter**. Like the **straight ring counter**, the outcome Q_3' of the last flip-flop is passed to the first flip-flop as an input. In the twisted ring counter, the ORI input is passed to all the flip flops as **clear** input.

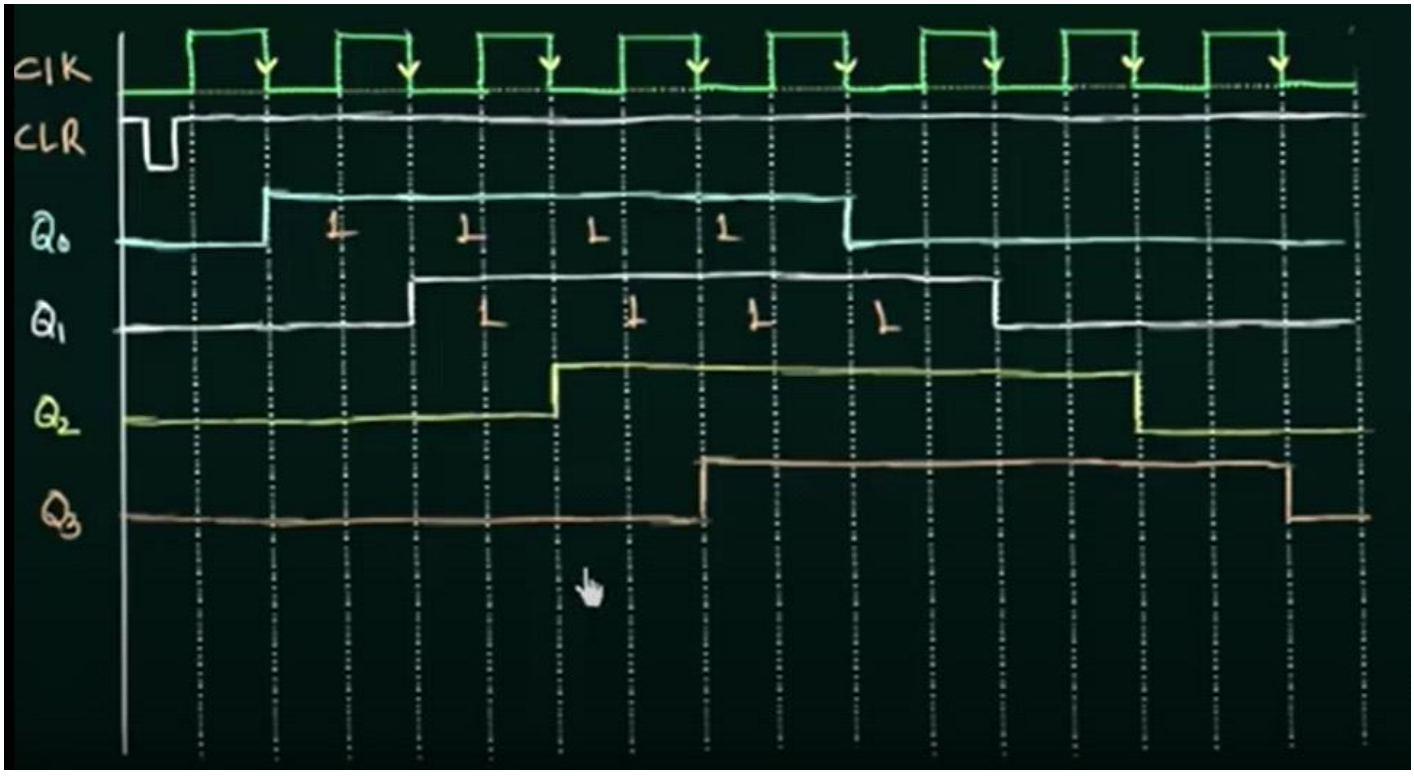
Logic Diagram



Truth table



Signal diagram



Difference between Straight and Twisted Ring Counter:

STRAIGHT RING COUNTER	TWISTED RING COUNTER
It connects the output of the last shift register to the input of first shift register.	It connects the complement of output of the last shift register to the input of the first register.
It is known as One hot counter.	It is known as Walking ring counter.
It circulates a single bit (0 or 1) around the ring.	It circulates stream of 1 followed by stream of 0.
PRESET is used in first shift register.	PRESET is not used in twisted ring counter.
CLEAR is used for last (n-1) flip-flops.	CLEAR is used for all flip-flips in it.
It is used in successive approximation and stepper motor control.	It is used in phase shift or function generator.

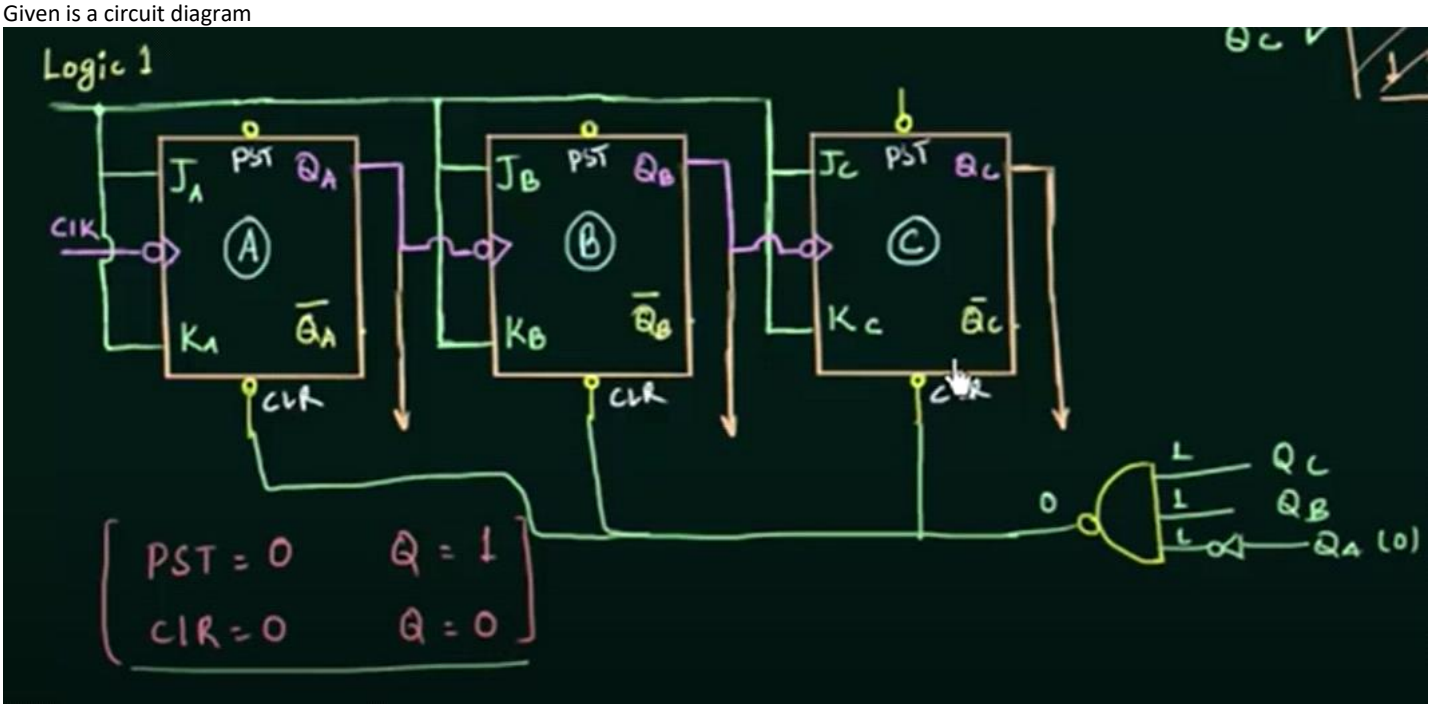
MOD n counter

06 February 2021 12:03 AM

A mod 2,4,8,16…. Is easy to design its just the 1,2,3,4…. Bit ripple counters respectively

What of a mod 6 counter

Given is a circuit diagram



We use a 3 bit ripple counter but as soon as it outputs the 6th state 110 we use a nand gate to reset all flip flops using overriding clear input

Adder

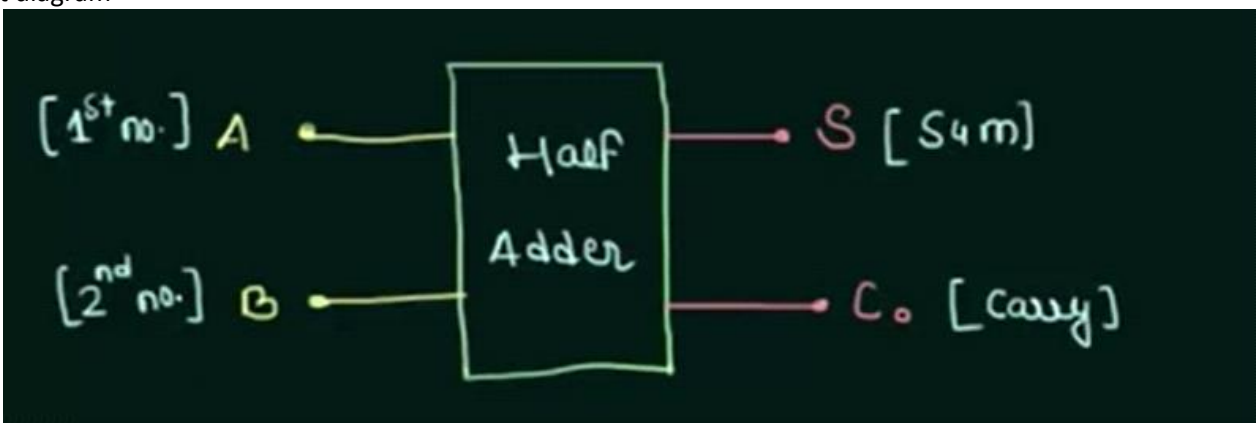
27 January 2021 07:17 AM

Half Adder

Takes two single bit input

Does not support carry input from previous sum

Circuit diagram



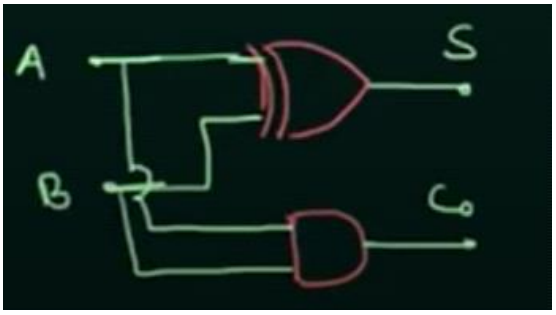
Truth table

A	B	S	C
0	0	0	0
1	0	1	0
0	1	1	0
1	1	0	1

Internal gates

S=A$\oplus B$

C=A.B

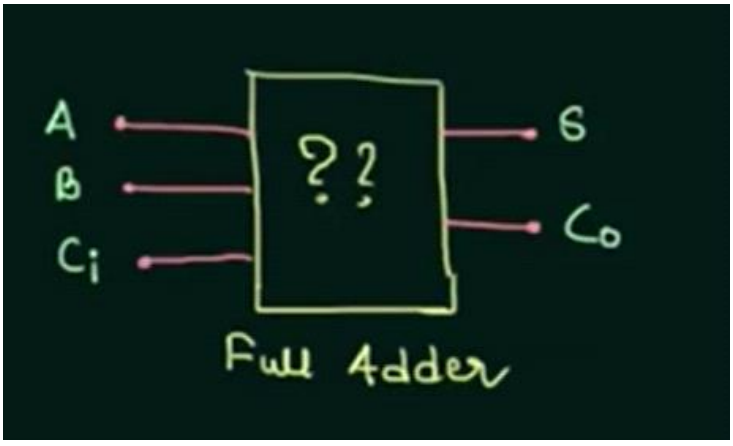


Full Adder

Takes three single bit input

Does support carry input from previous sum

Circuit diagram



Truth table

A	B	Ci	S	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Internal gates

S=A$\oplus B \oplus Ci$

C=A.B + Ci(A$\oplus B$)=ACi+BA+BCi

Subtractor

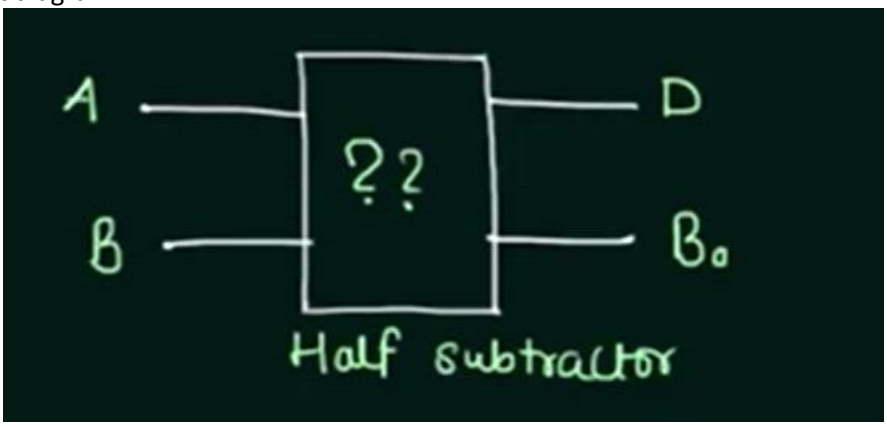
27 January 2021 07:44 AM

Half Subtractor

Takes two single bit input

Does not support borrow input from previous difference

Circuit diagram



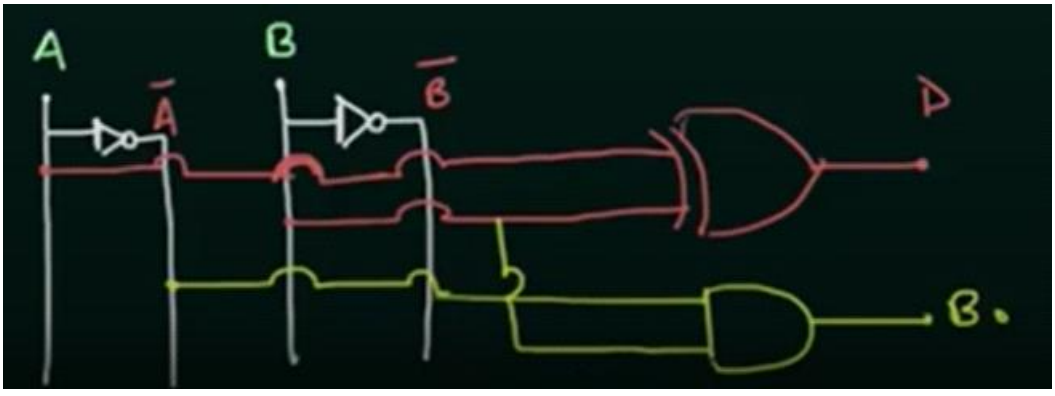
Truth table

A	B	D	Bo
0	0	0	0
1	0	1	1
0	1	1	0
1	1	0	0

Internal gates

D=A$\oplus B$

Bo=$\bar{A}$.B

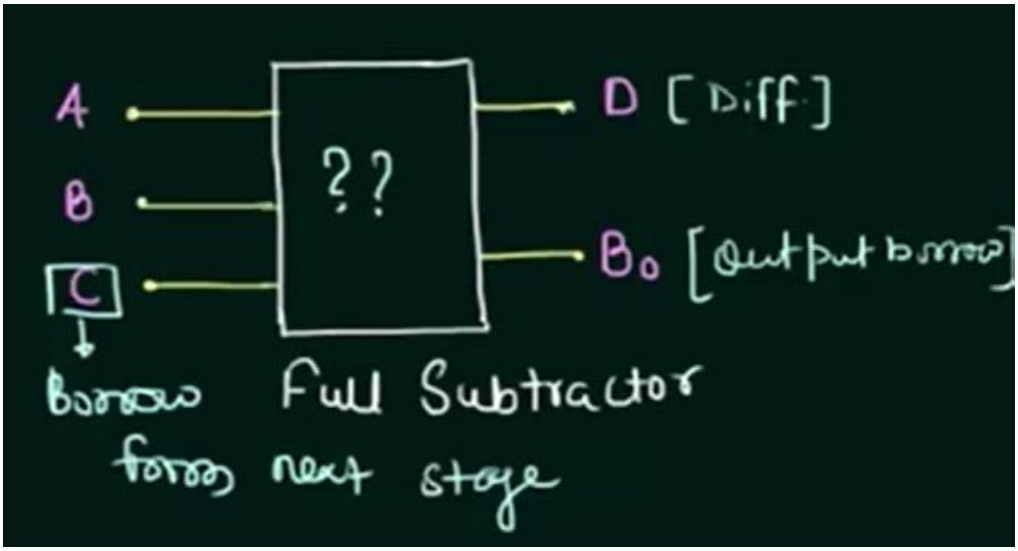


Full Subtractor

Takes three single bit input

Does support borrow input from previous difference

Circuit diagram



Truth table

A	B	C	D	Bo
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Internal gates

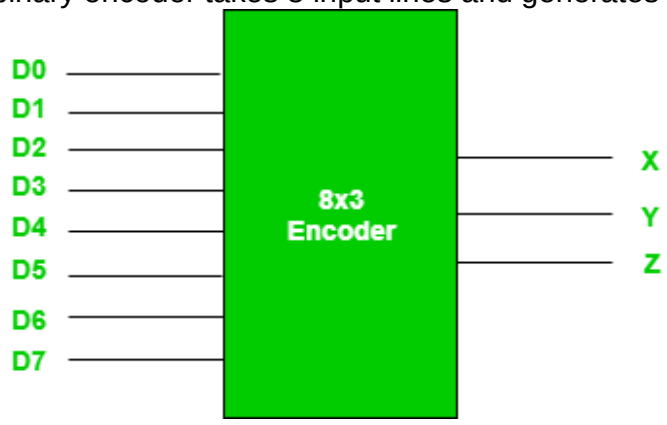
D=A$\oplus B \oplus C$

Bo=BC+$\bar{A}$C+$\bar{A}$B

Encoder

27 January 2021 08:16 AM

Encoders –
An encoder is a combinational circuit that converts binary information in the form of a 2^N input lines into N output lines, which represent N bit code for the input. For simple encoders, it is assumed that only one input line is active at a time.
As an example, let's consider **Octal to Binary** encoder. As shown in the following figure, an octal-to-binary encoder takes 8 input lines and generates 3 output lines.



$$N<=2^n$$

Truth Table –

D7	D6	D5	D4	D3	D2	D1	D0	X	Y	Z
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

As seen from the truth table, the output is 000 when D0 is active; 001 when D1 is active; 010 when D2 is active and so on.

$$X = D7'\,D6'\,D5'\,D4\;\;D3'\,D2'\,D1'\,D0'$$
$$+\; D7'\,D6'\,D5\;\;D4'\,D3'\,D2'\,D1'\,D0'$$
$$+\; D7'\,D6\;\;D5'\,D4'\,D3'\,D2'\,D1'\,D0'$$
$$+\; D7\;\;D6'\,D5'\,D4'\,D3'\,D2'\,D1'\,D0'$$
$$= D7 + D6 + D5 + D4$$

$$X = D4 + D5 + D6 + D7$$

Y = D2 + D3 + D6 + D7

Z = D1 + D3 + D5 + D7

Hence, the encoder can be realised with OR gates as follows:

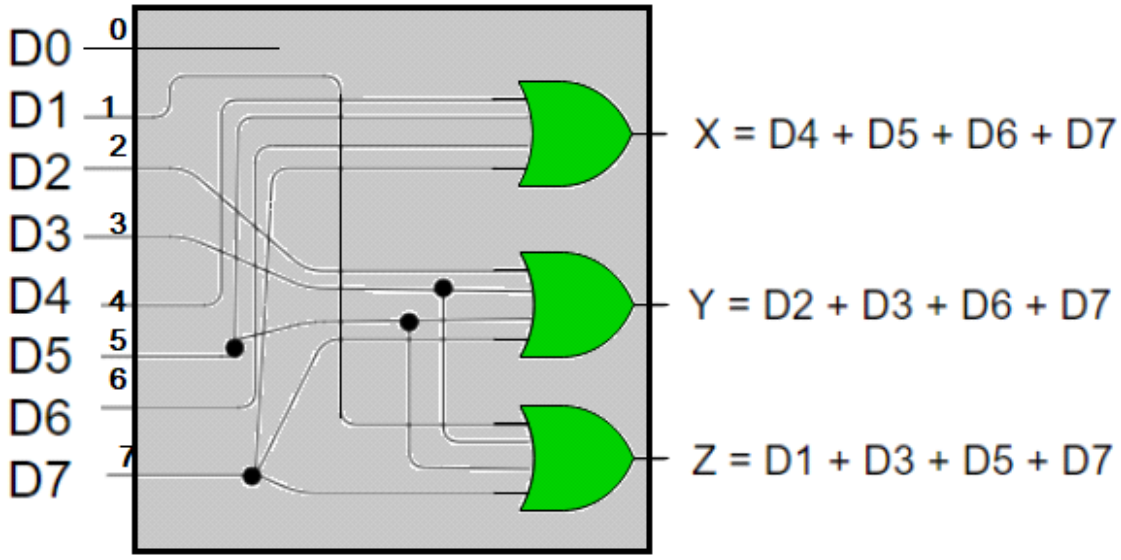
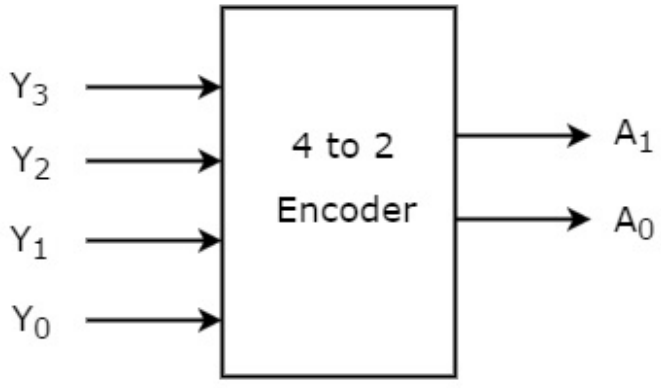


Fig: ENCODER

4 to 2 Encoder

Let 4 to 2 Encoder has four inputs Y_3, Y_2, Y_1 & Y_0 and two outputs A_1 & A_0. The **block diagram** of 4 to 2 Encoder is shown in the following figure.



At any time, only one of these 4 inputs can be '1' in order to get the respective binary code at the output. The **Truth table** of 4 to 2 encoder is shown below.

Inputs				**Outputs**	
Y_3	**Y_2**	**Y_1**	**Y_0**	**A_1**	**A_0**
0	**0**	**0**	**1**	**0**	**0**

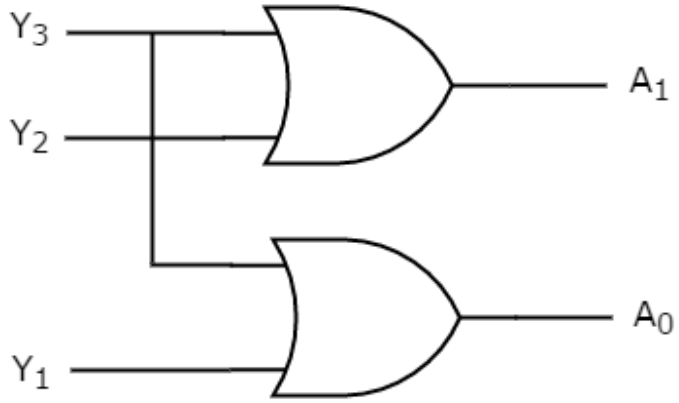
0	**0**	**1**	**0**	**0**	**1**
0	**1**	**0**	**0**	**1**	**0**
1	**0**	**0**	**0**	**1**	**1**

From Truth table, we can write the Boolean functions for each output as

A1=Y3+Y2

A0=Y3+Y1

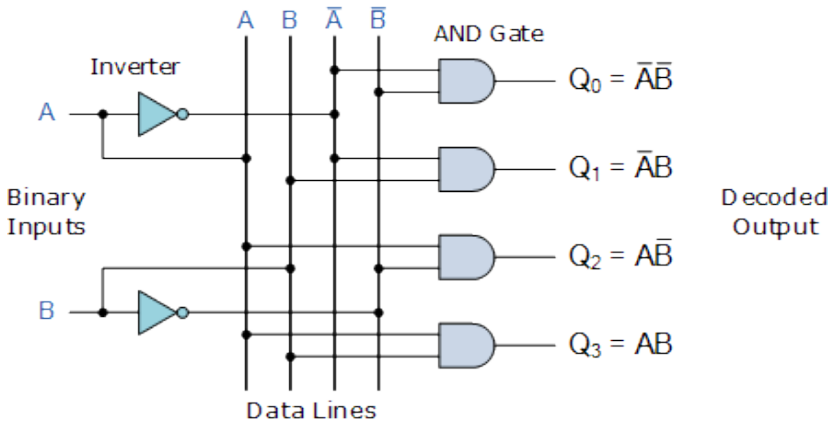
We can implement the above two Boolean functions by using two input OR gates. The **circuit diagram** of 4 to 2 encoder is shown in the following figure.

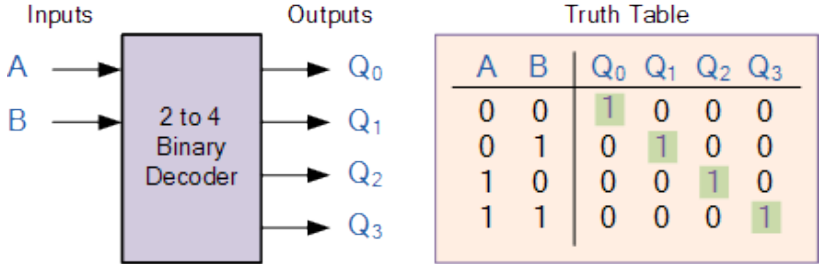


Decoder

27 January 2021 08:20 AM

A 2-to-4 Binary Decoders





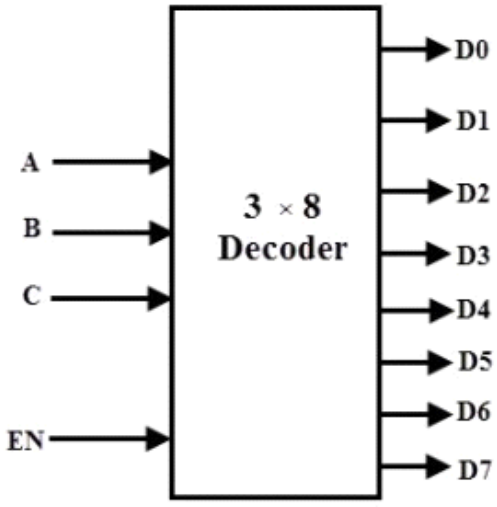
Truth Table

A	B	Q_0	Q_1	Q_2	Q_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Inputs			Outputs			
EN	A	B	Y_3	Y_2	Y_1	Y_0
0	×	×	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

3-to-8 Decoder

In a 3-to-8 decoder, three inputs are decoded into eight outputs. It has three inputs as A, B, and C and eight output from Y0 through Y7. Based on the combinations of the three inputs, only one of the eight outputs is selected.



The figure below shows the truth table of a 3-to-8 decoder. Enable input is provided to activate the decoded output depends on the input combinations A, B and C. Suppose if A = B=1 and C= 0, then the output Y6 is 1 and all other outputs are zero. So from the truth table, minterms represents the each output equation and are given as

$Y0 = \overline{A}\,\overline{B}\,\overline{C}$

$Y1 = \overline{A}\,\overline{B}\,C$

$Y2 = \overline{A}\,B\,\overline{C}$

$Y3 = \overline{A}\,B\,C$

$Y4 = A\,\overline{B}\,\overline{C}$

$Y5 = A\,\overline{B}\,C$

$Y6 = A\,B\,\overline{C}$

$Y7 = A\,B\,C$

Inputs				Outputs							
EN	A	B	C	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	×	×	×	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0

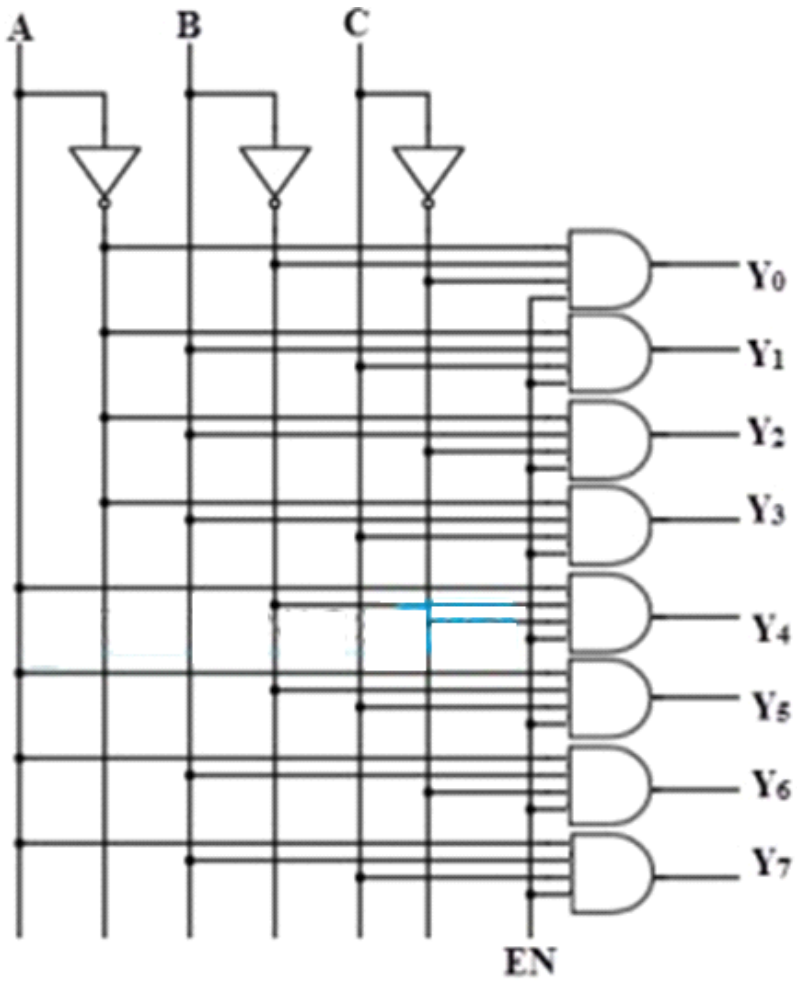
Inputs				Outputs							
EN	A	B	C	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	×	×	×	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

Using the above min term expressions for each output, the circuit of 3-to-8 decoder is can be implemented by using three NOT gates and eight AND gates. Each NOT gate provides the complement of the input and AND gates generates one of the minterms.

Also enable input activate the decoded output depends on the input data. The logic diagram of this decoder is shown below.

Only one of eight outputs is high at a given time for a particular input combination, that why this decoder is also called as 1-of-8 decoder. Suppose, when ABC = 011, then only AND gate 4 has all inputs high, thus Y3 is high.

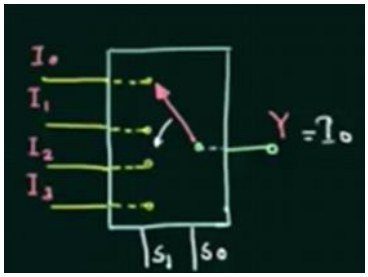
Also, 3-bit binary numbers at the input is converted to eight digits at the output (which is equivalent to octal number system), that's how; it is also called as a binary-to-octal decoder


A
B
C
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
EN

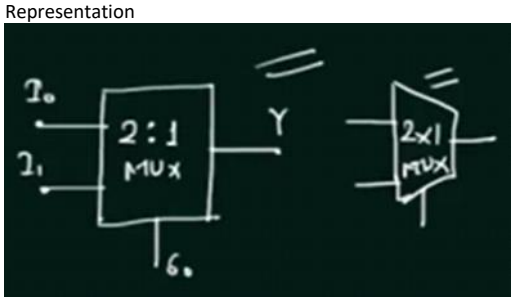
Multiplexer

27 January 2021 08:27 AM

Aka MUX, selector
It is circuit that selects one of the data line from many input lines and directs it to the output lines
Guided by selector lines

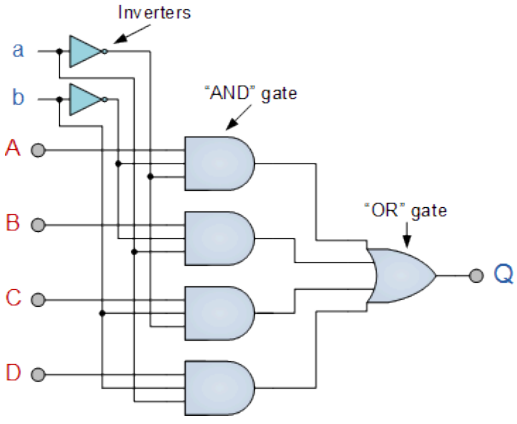


Representation



No of select lines = log(no of input) with base 2

4- Channel Multiplexer using Logic Gates

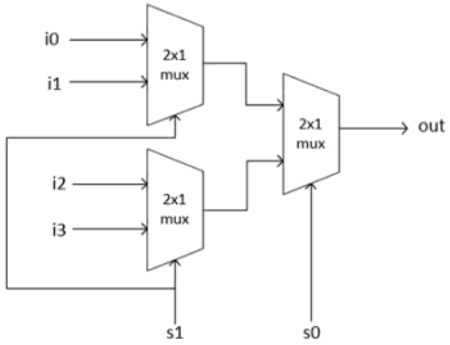


Q=Aa'b' + Bab' + Ca'b + Dab

a	b	Q
0	0	A
0	1	B
1	0	C
1	1	D

4×2 Mux using 2×1 muxs

we have constructed our 2×1 mux we can easily construct 4×2 mux using three of these 2×1 muxes as shown in the block diagram given below:



When S1 is set to HIGH it will select i1 and i3 now if s0 is LOW output will have i1 otherwise i3 and similar for i0 and i2. This combination is shown below:

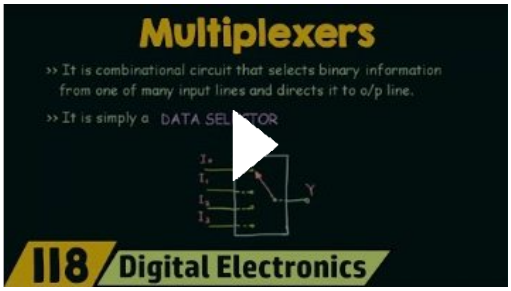
S0	S1	Out
0	0	i0
0	1	i1
1	0	i2
1	1	i3

Focus on the diagram of 2×1 mux and you will get it how this 4×2 mux

works. Now using hierarchical designing it is very easy to write Verilog code of 4×2 mux by just instantiating three 2×1 muxes.

Other resources

[Introduction to Multiplexers | MUX Basic]



Large MUX

27 January 2021 08:49 AM

8x1 Multiplexer

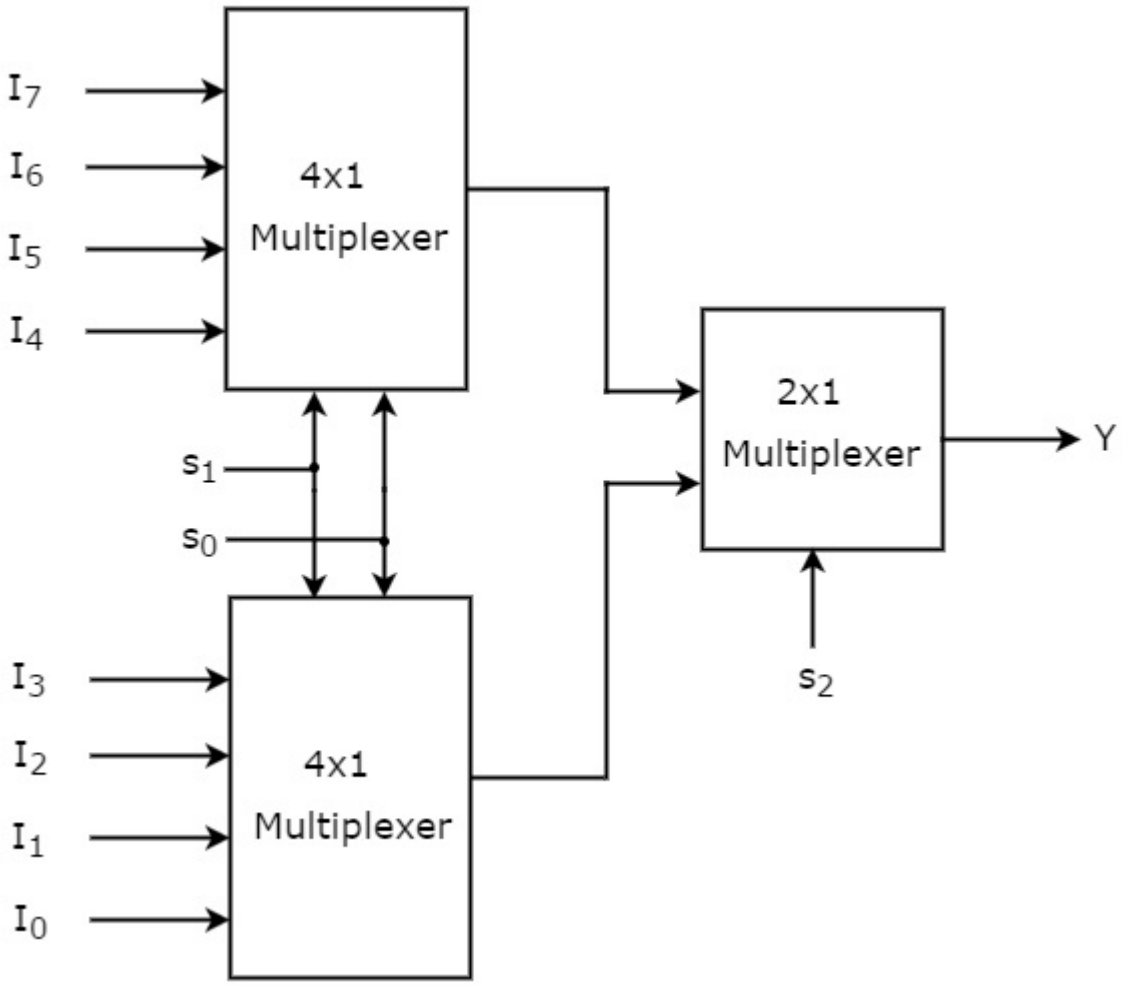
In this section, let us implement 8x1 Multiplexer using 4x1 Multiplexers and 2x1 Multiplexer. We know that 4x1 Multiplexer has 4 data inputs, 2 selection lines and one output. Whereas, 8x1 Multiplexer has 8 data inputs, 3 selection lines and one output.

So, we require two **4x1 Multiplexers** in first stage in order to get the 8 data inputs. Since, each 4x1 Multiplexer produces one output, we require a **2x1 Multiplexer** in second stage by considering the outputs of first stage as inputs and to produce the final output.

Let the 8x1 Multiplexer has eight data inputs I_7 to I_0, three selection lines s_2, s_1 & s0 and one output Y. The **Truth table** of 8x1 Multiplexer is shown below.

Selection Inputs			**Output**
S_2	S_1	S_0	Y
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

We can implement 8x1 Multiplexer using lower order Multiplexers easily by considering the above Truth table. The **block diagram** of 8x1 Multiplexer is shown in the following figure.



16x1 Multiplexer

In this section, let us implement 16x1 Multiplexer using 8x1 Multiplexers and 2x1 Multiplexer. We know that 8x1 Multiplexer has 8 data inputs, 3 selection lines and one output. Whereas, 16x1 Multiplexer has 16 data inputs, 4 selection lines and one output.

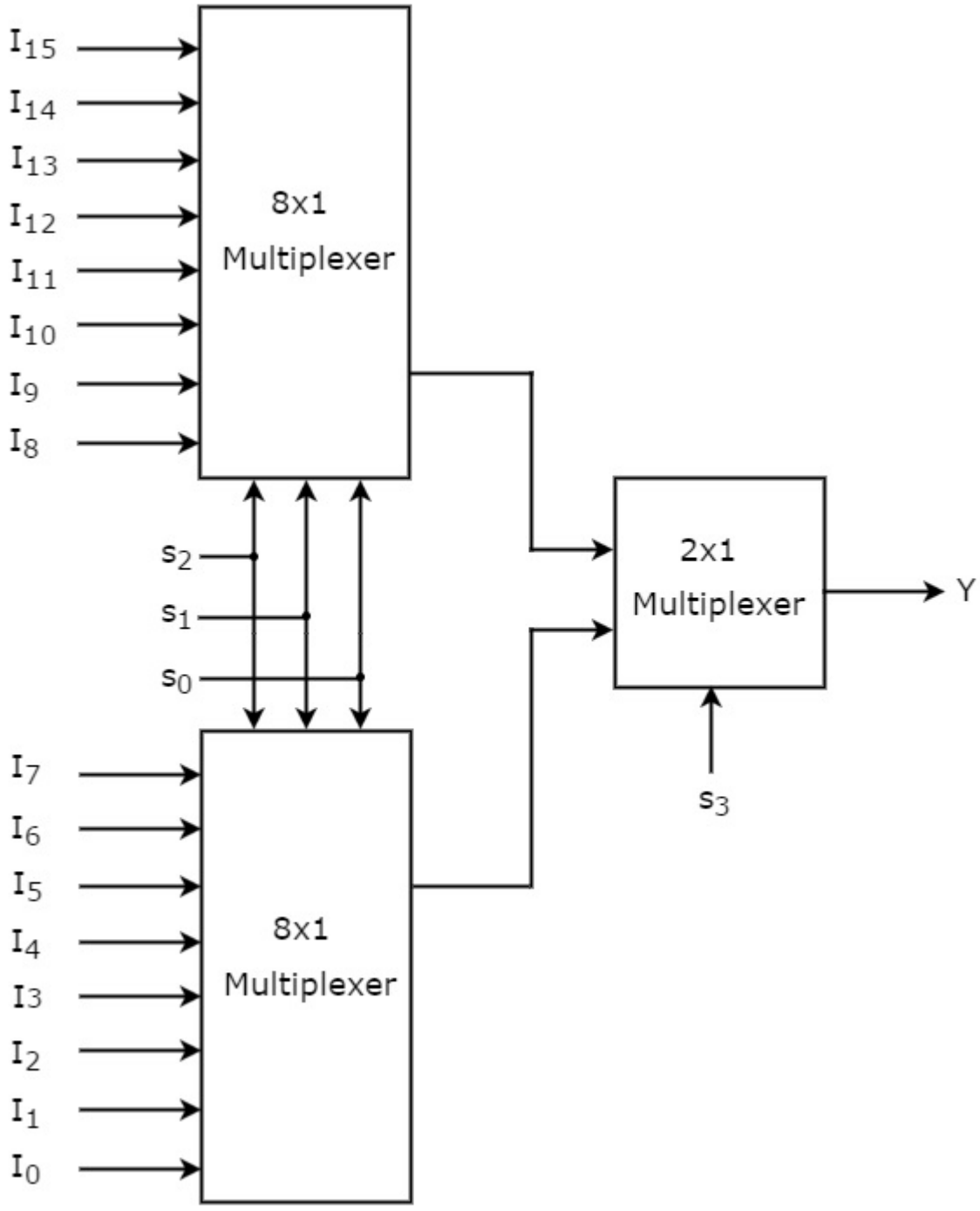
So, we require two **8x1 Multiplexers** in first stage in order to get the 16 data inputs. Since, each 8x1 Multiplexer produces one output, we require a 2x1 Multiplexer in second stage by considering the outputs of first stage as inputs and to produce the final output.

Let the 16x1 Multiplexer has sixteen data inputs I_{15} to I_0, four selection lines s_3 to s_0 and one output Y. The **Truth table** of 16x1 Multiplexer is shown below.

Selection Inputs				**Output**
S_3	**S_2**	**S_1**	**S_0**	**Y**
0	0	0	0	I_0
0	0	0	1	I_1
0	0	1	0	I_2
0	0	1	1	I_3
0	1	0	0	I_4
0	1	0	1	I_5
0	1	1	0	I_6
0	1	1	1	I_7
1	0	0	0	I_8

1	0	0	1	I_9
1	0	1	0	I_{10}
1	0	1	1	I_{11}
1	1	0	0	I_{12}
1	1	0	1	I_{13}
1	1	1	0	I_{14}
1	1	1	1	I_{15}

We can implement 16x1 Multiplexer using lower order Multiplexers easily by considering the above Truth table. The **block diagram** of 16x1 Multiplexer is shown in the following figure.



Demultiplexer

27 January 2021 08:50 AM

Demultiplexer

A **demultiplexer** (or demux) is a device that takes a single input line and routes it to one of several digital output lines. A **demultiplexer** of 2^n outputs has n select lines, which are used to select which output line to send the input. A **demultiplexer** is also called a data distributor.

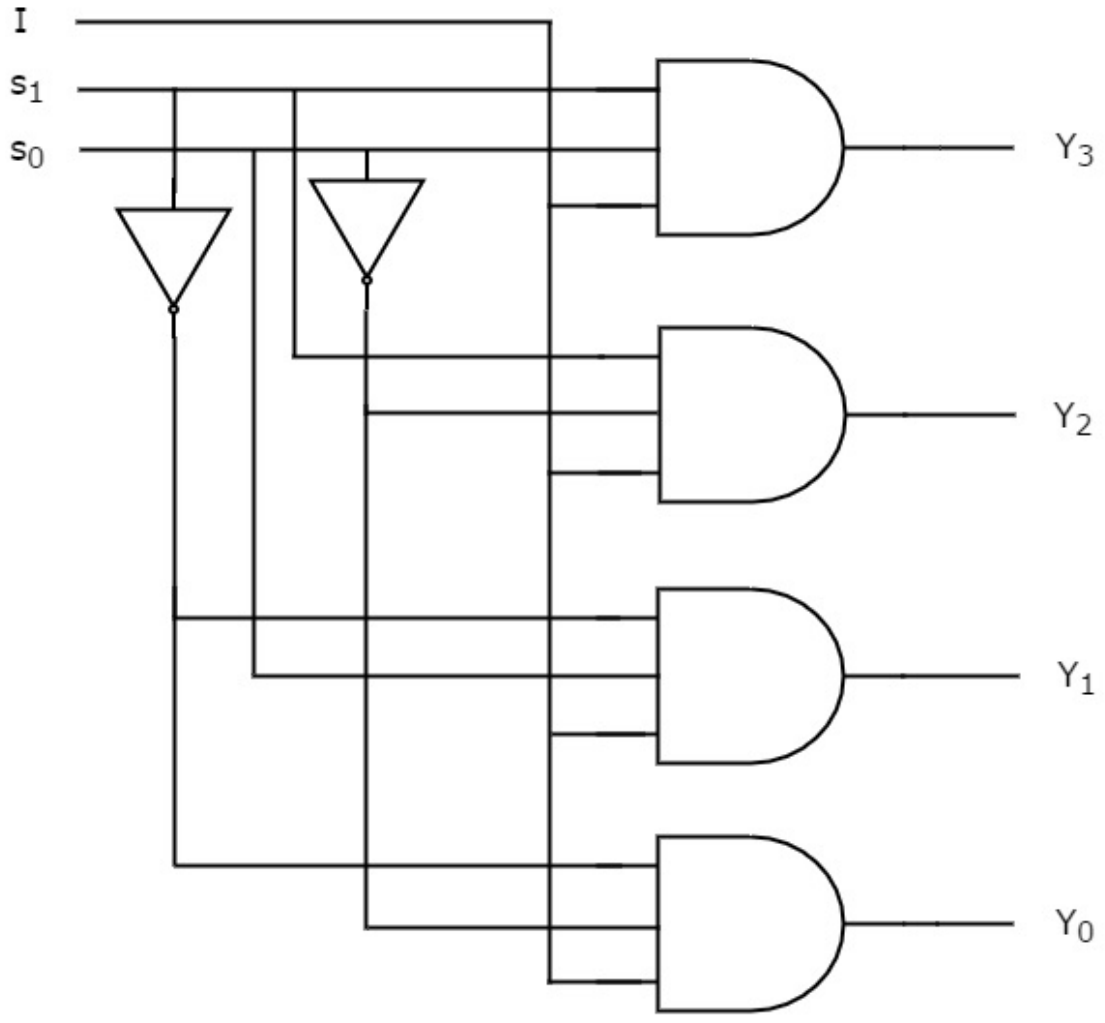
1 to 4 demultiplexer

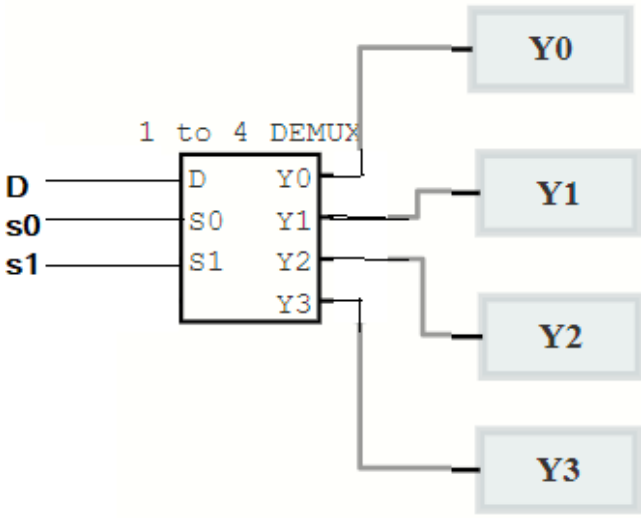
A 1 to 4 multiplexer uses 2 select lines (S0, S1) to determine which one of the 4 outputs (Y0 - Y3) is routed from the input (D). Its characteristics can be described in the following simplified truth table.

Truth Table

S1	S0	Y3	Y2	Y1	Y0
0	0	0	0	0	D
0	1	0	0	D	0
1	0	0	D	0	0
1	1	D	0	0	0

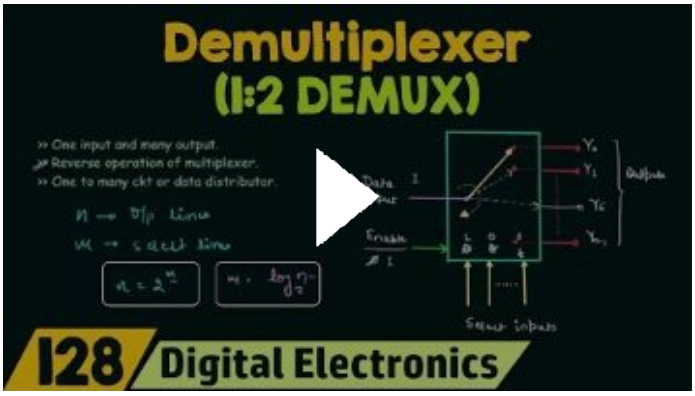
Y3=s1.s0.D
Y2=s1.s0`.D
Y1=s1`.s2.D
Y0=s1`.s2`.D





Other resources

[Introduction to Demultiplexer | 1:2 DEMUX]



Large DEMUX

27 January 2021 09:04 AM

1x8 De-Multiplexer

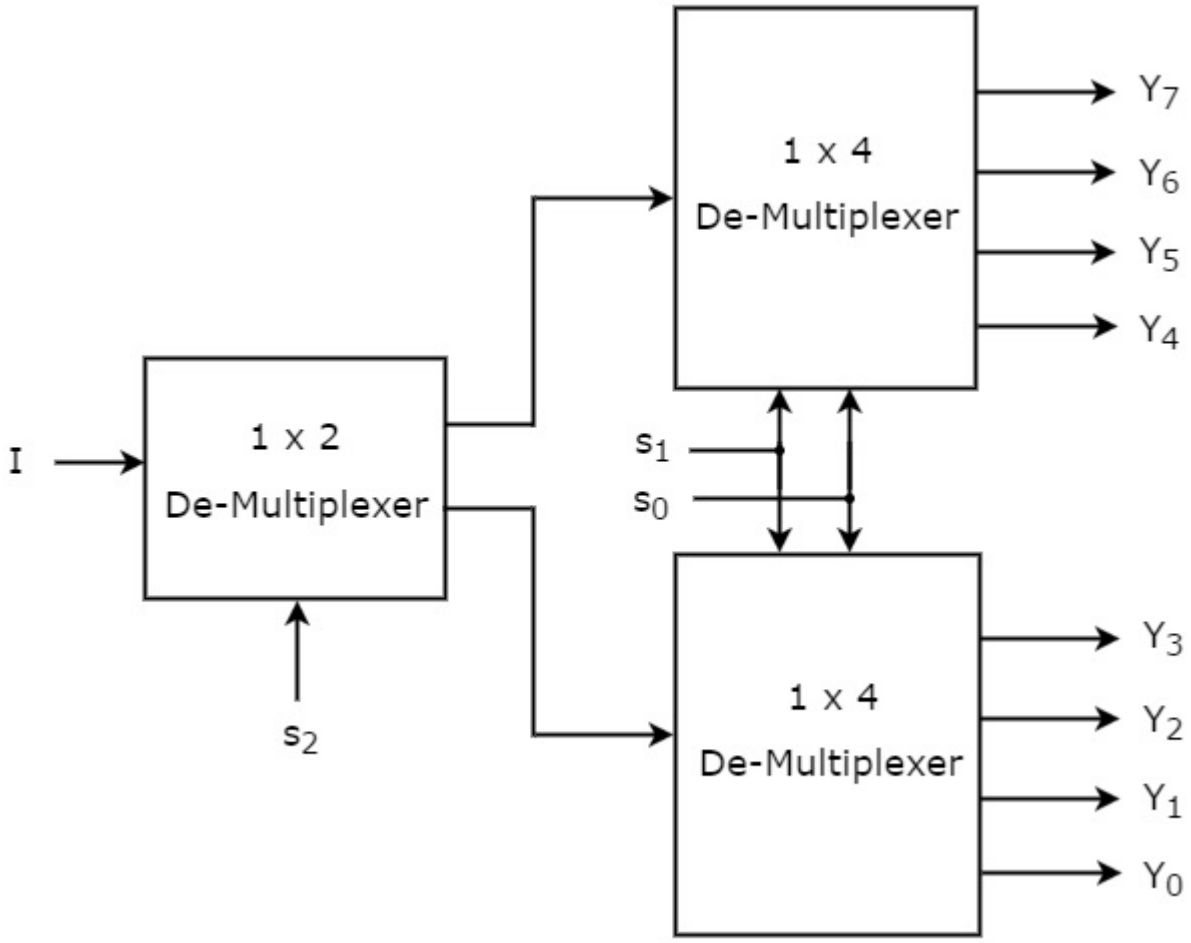
In this section, let us implement 1x8 De-Multiplexer using 1x4 De-Multiplexers and 1x2 De-Multiplexer. We know that 1x4 De-Multiplexer has single input, two selection lines and four outputs. Whereas, 1x8 De-Multiplexer has single input, three selection lines and eight outputs.

So, we require two **1x4 De-Multiplexers** in second stage in order to get the final eight outputs. Since, the number of inputs in second stage is two, we require **1x2 DeMultiplexer** in first stage so that the outputs of first stage will be the inputs of second stage. Input of this 1x2 De-Multiplexer will be the overall input of 1x8 De-Multiplexer.

Let the 1x8 De-Multiplexer has one input I, three selection lines s_2, s_1 & s_0 and outputs Y_7 to Y_0. The **Truth table** of 1x8 De-Multiplexer is shown below.

Selection Inputs			**Outputs**							
s_2	**s_1**	**s_0**	**Y_7**	**Y_6**	**Y_5**	**Y_4**	**Y_3**	**Y_2**	**Y_1**	**Y_0**
0	0	0	0	0	0	0	0	0	0	**I**
0	0	1	0	0	0	0	0	0	**I**	0
0	1	0	0	0	0	0	0	**I**	0	0
0	1	1	0	0	0	0	**I**	0	0	0
1	0	0	0	0	0	**I**	0	0	0	0
1	0	1	0	0	**I**	0	0	0	0	0
1	1	0	0	**I**	0	0	0	0	0	0
1	1	1	**I**	0	0	0	0	0	0	0

We can implement 1x8 De-Multiplexer using lower order Multiplexers easily by considering the above Truth table. The **block diagram** of 1x8 De-Multiplexer is shown in the following figure.



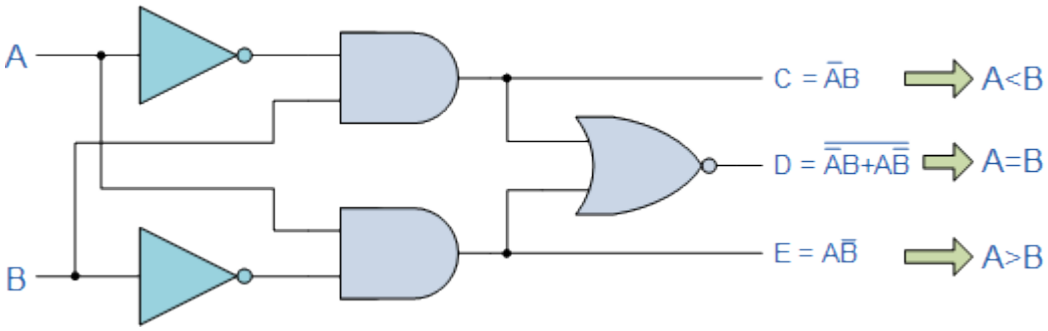
The common **selection lines, s_1 & s_0** are applied to both 1x4 De-Multiplexers. The outputs of upper 1x4 De-Multiplexer are Y_7 to Y_4 and the outputs of lower 1x4 De-Multiplexer are Y_3 to Y_0.

The other **selection line, s_2** is applied to 1x2 De-Multiplexer. If s_2 is zero, then one of the four outputs of lower 1x4 De-Multiplexer will be equal to input, I based on the values of selection lines s_1 & s_0. Similarly, if s_2 is one, then one of the four outputs of upper 1x4 DeMultiplexer will be equal to input, I based on the values of selection lines s_1 & s_0.

Comparator

27 January 2021 09:14 AM

1-bit Digital Comparator Circuit



Then the operation of a 1-bit digital comparator is given in the following Truth Table.

Digital Comparator Truth Table

Inputs		Outputs		
B	A	A > B	A = B	A < B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

You may notice two distinct features about the comparator from the above truth table.

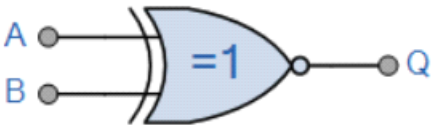
Firstly, the circuit does not distinguish between either two “0” or two “1”‘s as an output A = B is produced when they are both equal, either A = B = “0” or A = B = “1”.

Secondly, the output condition for A = B resembles that of a commonly available logic gate, the Exclusive-NOR or Ex-NOR function (equivalence) on each of the n-bits giving: Q = A ⊕ B

Digital comparators actually use Exclusive-NOR gates within their design for comparing their respective pairs of bits. When we are comparing two binary or BCD values or variables against each other, we are comparing the

“magnitude” of these values, a logic “0” against a logic “1” which is where the term **Magnitude Comparator** comes from.

The ex-Nor gate

<table>
<tr><th>Symbol</th><th colspan="3">Truth Table</th></tr>
<tr><td rowspan="5"> 

2-input Ex-NOR Gate</td><td>B</td><td>A</td><td>Q</td></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
<tr><td>Boolean Expression $Q = \overline{A \oplus B}$</td><td colspan="3">Read if A AND B the SAME gives Q</td></tr>
</table>

2-Bit Magnitude Comparator

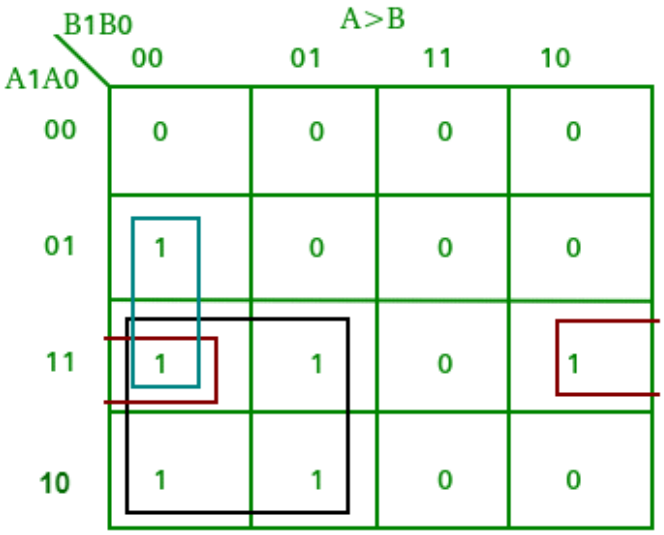
27 January 2021 09:24 AM

A comparator used to compare two binary numbers each of two bits is called a 2-bit Magnitude comparator. It consists of four inputs and three outputs to generate less than, equal to and greater than between two binary numbers.

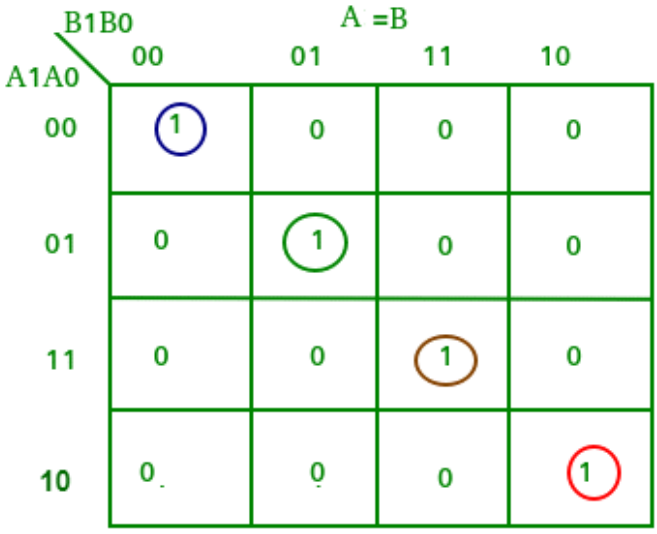
The truth table for a 2-bit comparator is given below:

INPUT				OUTPUT		
A1	A0	B1	B0	A<B	A=B	A>B
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

From the above truth table K-map for each output can be drawn as follows:



A>B: A1B1′ + A0B1′B0′ + A1A0B0′

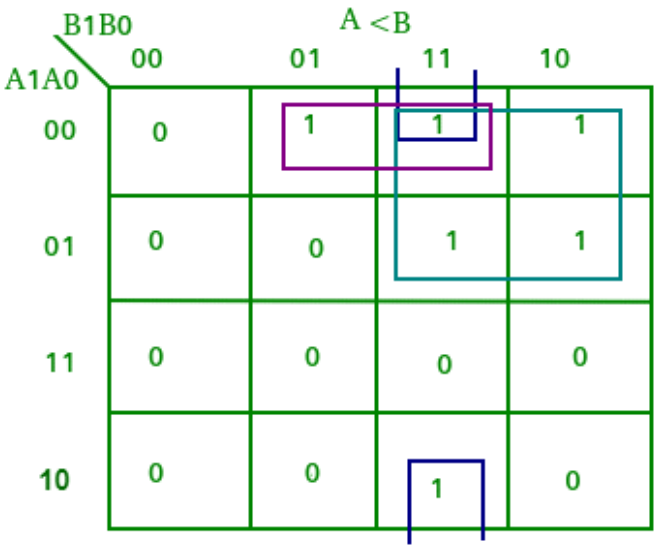


A=B: A1′A0′B1′B0′ + A1′A0B1′B0 + A1A0B1B0 + A1A0′B1B0′

: A1′B1′ (A0′B0′ + A0B0) + A1B1 (A0B0 + A0′B0′)

: (A0B0 + A0′B0′) (A1B1 + A1′B1′)

: (A0 Ex-Nor B0) (A1 Ex-Nor B1) =$(A_0 \oplus B_0)'(A_1 \oplus B_1)'$



A<B: A1′B1 + A0′B1B0 + A1′A0′B0

From the above K-maps logical expressions for each output can be expressed as follows:

A>B: A1B1′ + A0B1′B0′ + A1A0B0′

A=B: A1′A0′B1′B0′ + A1′A0B1′B0 + A1A0B1B0 + A1A0′B1B0′

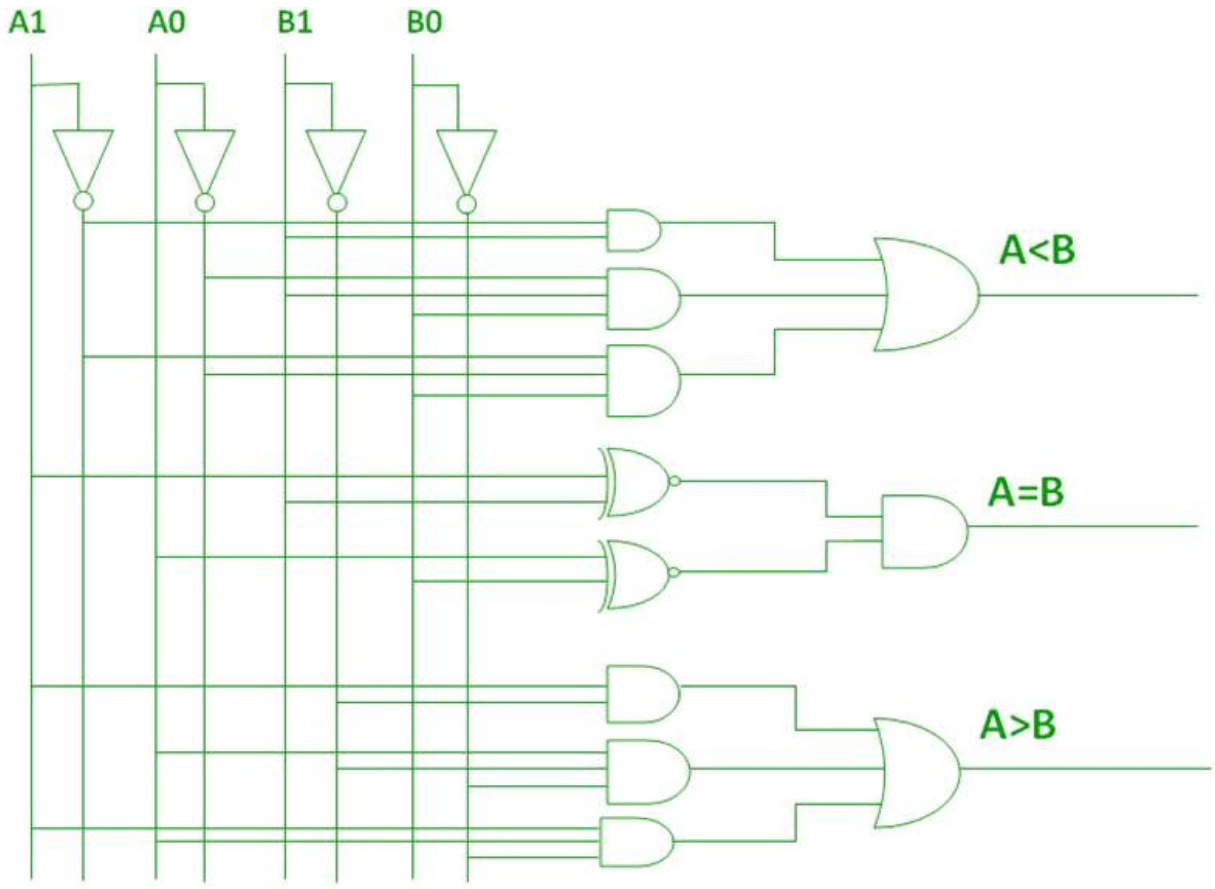
: A1′B1′ (A0′B0′ + A0B0) + A1B1 (A0B0 + A0′B0′)

: (A0B0 + A0′B0′) (A1B1 + A1′B1′)

: (A0 Ex-Nor B0) (A1 Ex-Nor B1) $=(A_0 \oplus B_0)'(A_1 \oplus B_1)'$

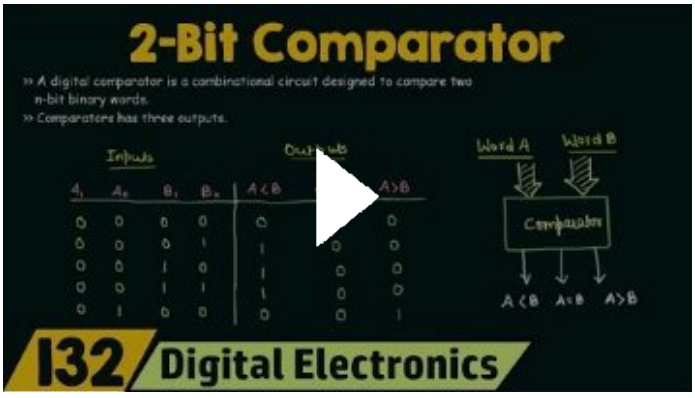
A<B: A1′B1 + A0′B1B0 + A1′A0′B0

By using these Boolean expressions, we can implement a logic circuit for this comparator as given below:



Other resources

[2-Bit Comparator]



Parity generator

27 January 2021 09:27 AM

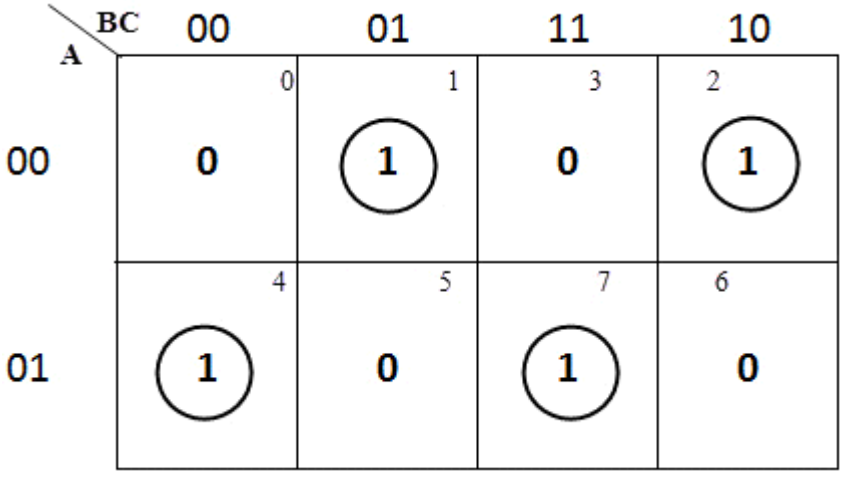
Even Parity Generator

Let us assume that a 3-bit message is to be transmitted with an even parity bit. Let the three inputs A, B and C are applied to the circuits and output bit is the parity bit P. The total number of 1s must be even, to generate the even parity bit P.

The figure below shows the truth table of even parity generator in which 1 is placed as parity bit in order to make all 1s as even when the number of 1s in the truth table is odd.

3-bit message			Even parity bit generator (P)
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

The K-map simplification for 3-bit message even parity generator is



From the above truth table, the simplified expression of the parity bit can be written as

$$P = \bar{A}\,\bar{B}\,C + \bar{A}\,B\,\bar{C} + A\,\bar{B}\,\bar{C} + A\,B\,C$$

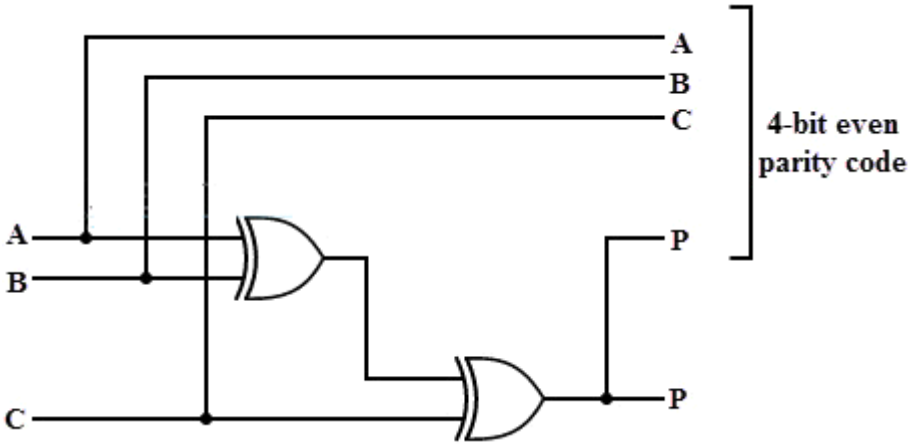
$$= \bar{A}\,(\bar{B}\,C + B\,\bar{C}) + A\,(\bar{B}\,\bar{C} + B\,C)$$

$$= \bar{A}\,(B \oplus C) + A\,(\overline{B \oplus C})$$

$$P = A \oplus B \oplus C$$

The above expression can be implemented by using two Ex-OR gates. The logic diagram of even parity generator with two Ex – OR gates is shown below. The three bit message along with the parity generated by this circuit which is transmitted to the receiving end where parity checker circuit checks whether any error is present or not.

To generate the even parity bit for a 4-bit data, three Ex-OR gates are required to add the 4-bits and their sum will be the parity bit.



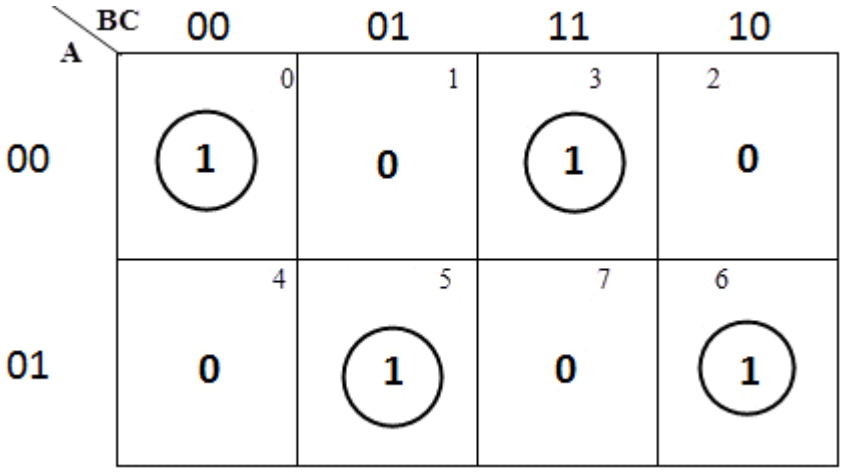
Odd Parity Generator

Let us consider that the 3-bit data is to be transmitted with an odd parity bit. The three inputs are A, B and C and P is the output parity bit. The total number of bits must be odd in order to generate the odd parity bit.

In the given truth table below, 1 is placed in the parity bit in order to make the total number of bits odd when the total number of 1s in the truth table is even.

3-bit message			Odd parity bit generator (P)
A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

The truth table of the odd parity generator can be simplified by using K-map as

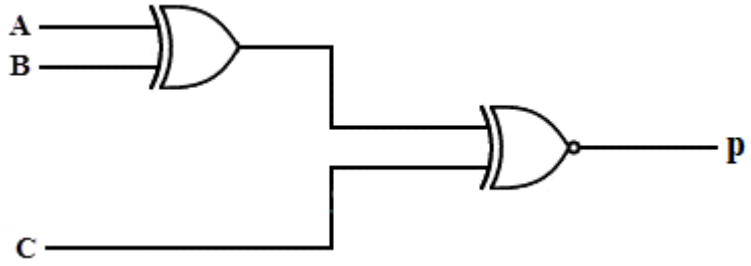


The output parity bit expression for this generator circuit is obtained as

P = A $\oplus$ B Ex-NOR C

The above Boolean expression can be implemented by using one Ex-OR gate and one Ex-NOR gate in order to design a 3-bit odd parity generator.

The logic circuit of this generator is shown in below figure , in which . two inputs are applied at one Ex-OR gate, and this Ex-OR output and third input is applied to the Ex-NOR gate , to produce the odd parity bit. It is also possible to design this circuit by using two Ex-OR gates and one NOT gate.


A
B
C
p

Parity checker

27 January 2021 09:36 AM

Parity Check

It is a logic circuit that checks for possible errors in the transmission. This circuit can be an even parity checker or odd parity checker depending on the type of parity generated at the transmission end. When this circuit is used as even parity checker, the number of input bits must always be even.

When a parity error occurs, the 'sum even' output goes low and 'sum odd' output goes high. If this logic circuit is used as an odd parity checker, the number of input bits should be odd, but if an error occurs the 'sum odd' output goes low and 'sum even' output goes high.

Even Parity Checker

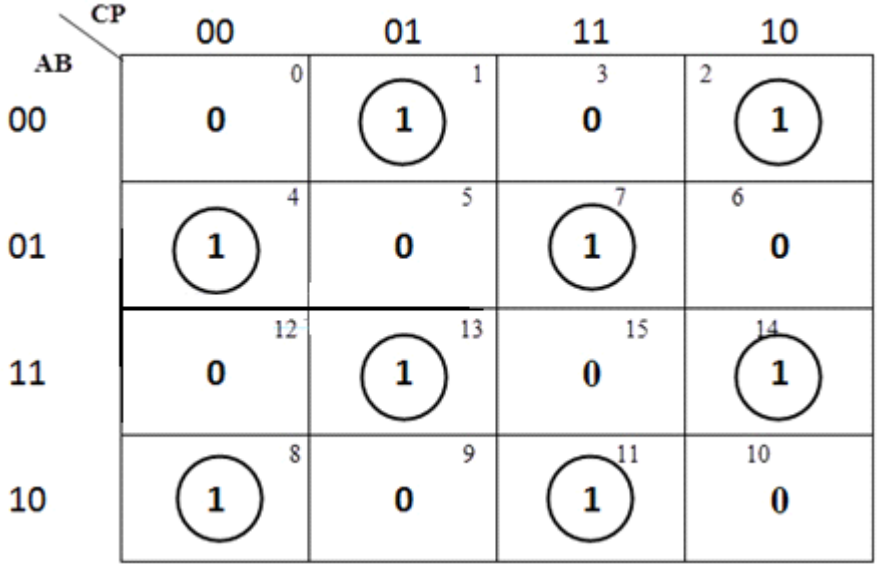
Consider that three input message along with even parity bit is generated at the transmitting end. These 4 bits are applied as input to the parity checker circuit which checks the possibility of error on the data. Since the data is transmitted with even parity, four bits received at circuit must have an even number of 1s.

If any error occurs, the received message consists of odd number of 1s. The output of the parity checker is denoted by PEC (parity error check).

The below table shows the truth table for the even parity checker in which PEC = 1 if the error occurs, i.e., the four bits received have odd number of 1s and PEC = 0 if no error occurs, i.e., if the 4-bit message has even number of 1s.

4-bit received message				Parity error check C_p
A	B	C	P	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

The above truth table can be simplified using K-map as shown below.



$$PEC = \bar{A}\,\bar{B}\,(\bar{C}\,D + C\,\bar{D}) + \bar{A}\,B\,(\bar{C}\,\bar{D} + C\,D) + A\,B\,(\bar{C}\,D + C\,\bar{D}) + A\,\bar{B}\,(\bar{C}\,\bar{D} + C\,D)$$

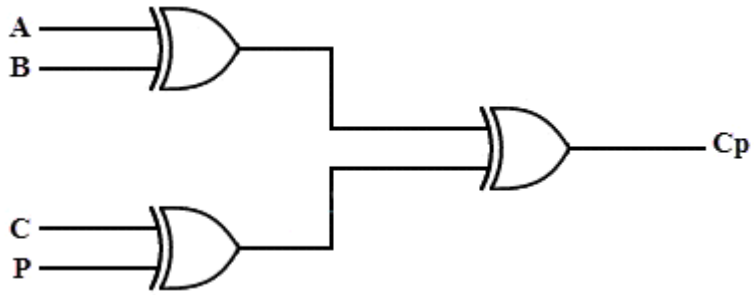
$$= \bar{A}\,\bar{B}\,(C \oplus D) + \bar{A}\,B\,(\overline{C \oplus D}) + A\,B\,(C \oplus D) + A\,\bar{B}\,(\overline{C \oplus D})$$

$$= (\bar{A}\,\bar{B} + A\,B)\,(C \oplus D) + (\bar{A}\,B + A\,\bar{B})\,(\overline{C \oplus D})$$

$= (A \oplus B)'\,(C \oplus D) + (A \oplus B)\,(C \oplus D)'$ [X'Y +XY= (X⊕Y)]

$= (A \oplus B) \oplus (C \oplus D)$

The above logic expression for the even parity checker can be implemented by using three Ex-OR gates as shown in figure. If the received message consists of five bits, then one more Ex-OR gate is required for the even parity checking.



Odd Parity Checker

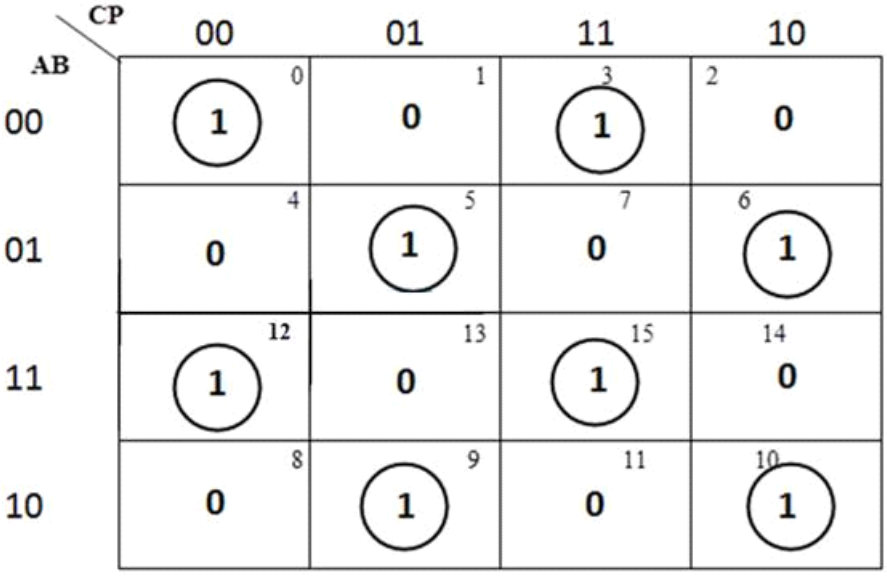
Consider that a three bit message along with odd parity bit is transmitted at the transmitting end. Odd parity checker circuit receives these 4 bits and checks whether any error are present in the data.

If the total number of 1s in the data is odd, then it indicates no error, whereas if the total number of 1s is even then it indicates the error since the data is transmitted with odd parity at transmitting end.

The below figure shows the truth table for odd parity generator where **PEC =1** if the 4-bit message received consists of **even number of 1s** (hence the error occurred) and **PEC= 0** if the message contains **odd number of 1s** (that means no error).

4-bit received message				Parity error check C_p
A	B	C	P	
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

The expression for the PEC in the above truth table can be simplified by K-map as shown below.

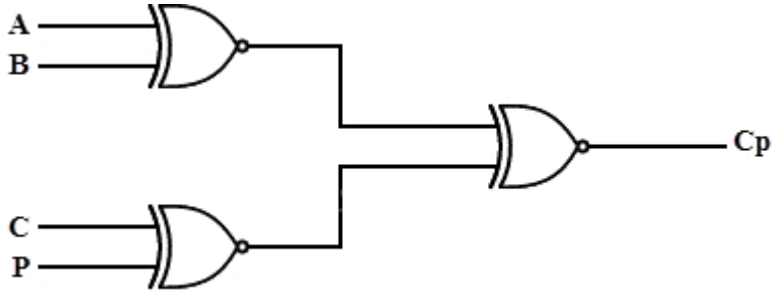


After simplification, the final expression for the PEC is obtained as

PEC = (A Ex-NOR B) Ex-NOR (C Ex-NOR D)

= $(A \oplus B)' \oplus (C \oplus D)'$

The expression for the odd parity checker can be designed by using three Ex-NOR gates as shown below.


A
B
C
P
Cp