PPTS ON

ANALOG AND DIGITAL ELECTRONICS (CSE)

II B.Tech III semester

Course code:AECB05

Ms. M.Lavanya, Assistant Professor

Institute of Aeronautical Engineering

SYLLABUS

Module-1:DIODE AND APPLICATIONS

Module-2:BIPOLAR JUNCTION TRANSISTOR (BJT)

Module-3:NUMBER SYSTEMS

Module-4: MINIMIZATION OF BOOLEAN FUNCTIONS

Module-5:SEQUENTIAL CIRCUITS FUNDAMENTALS

MODULE-I

DIODE AND APPLICATIONS

Diode - Static and Dynamic resistances, Equivalent circuit, Load line analysis, Diffusion and Transition Capacitances, Diode Applications: Switch-Switching times. Rectifier - Half Wave Rectifier, Full Wave Rectifier, Bridge Rectifier, Rectifiers with Capacitive Filter

CONTENTS

- Introduces the PN junction diodes.
- Analysis of diode characteristics.
- Equivalent circuits and load line.
- Diode behaviour as a switch.
- Diodes in rectifiers.
- Breakdown mechanisms- Avalanche, Zener.
- . Rectifiers with Capacitive Filter

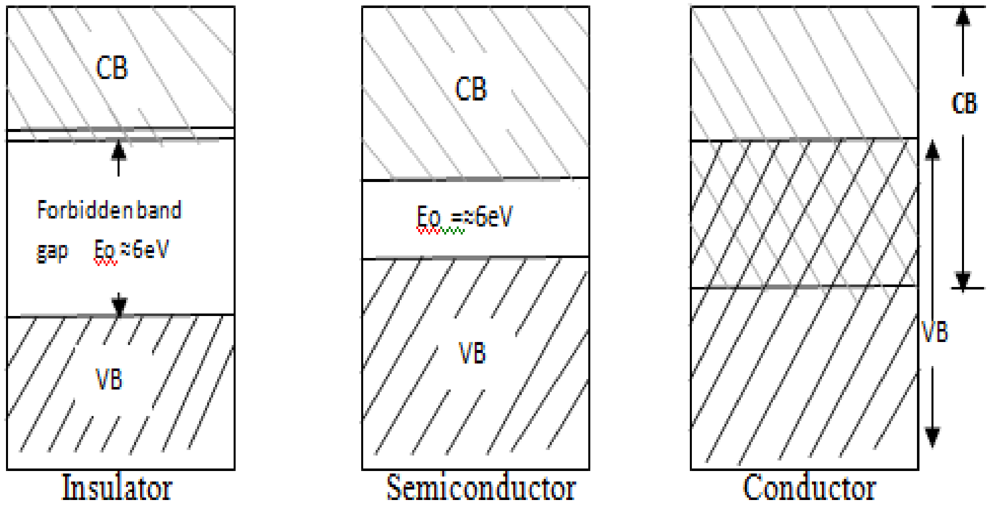
MATERIALS

Based on the electrical properties of the materials like conductivity, materials are divided into three types.

i) Conductors

ii) Semiconductors

iii) Insulators



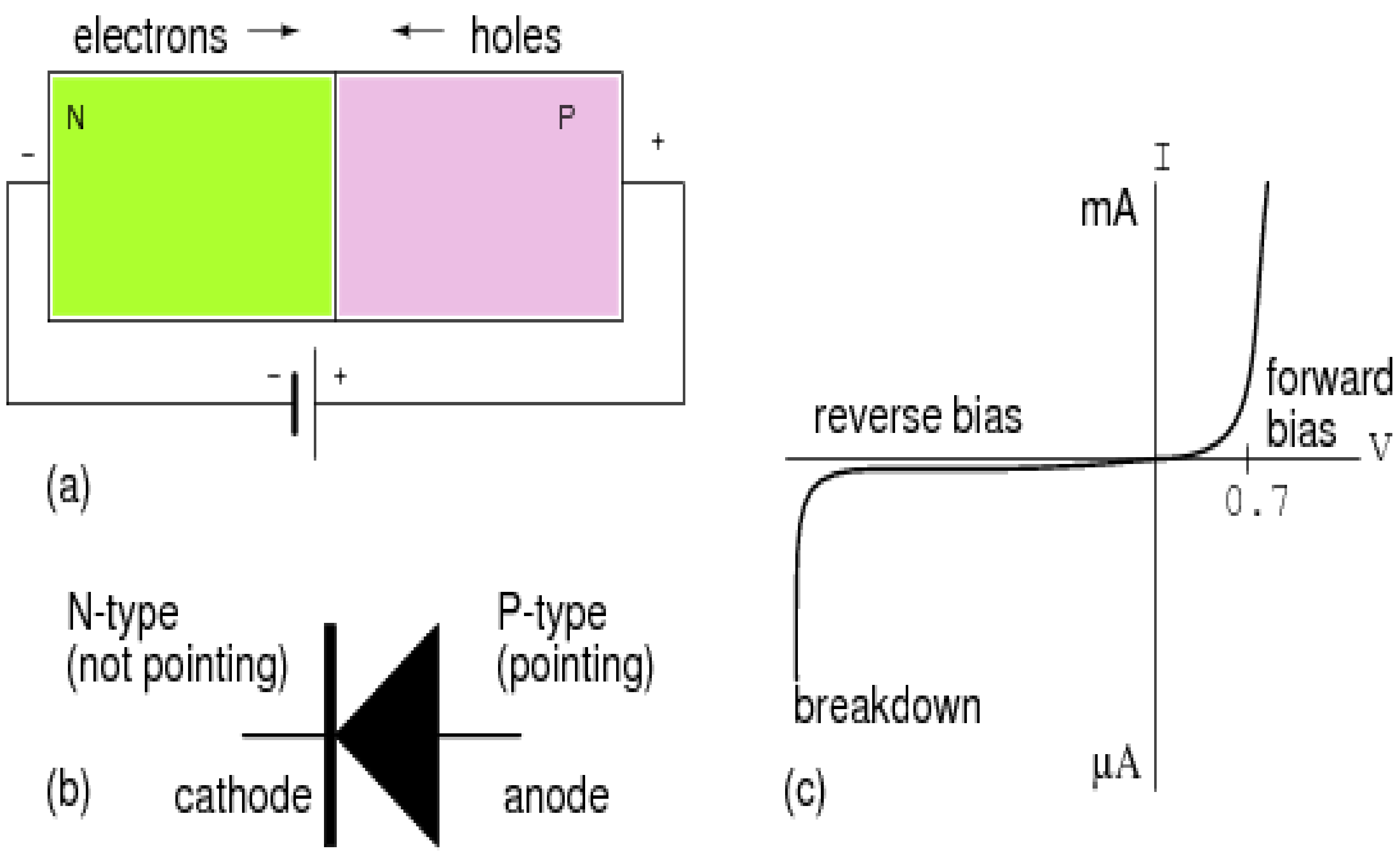
Energy band diagrams for insulator, semiconductor and conductor

PN JUNCTION DIODE

What do you mean by diode?

A PN junctionis a device formedby joining p-type with n-type semiconductors and separated by a thin junction is called PN Junction

PN JUNCTION DIODE



THEORY OF PN JUNCTION DIODE

- The depletion layer contains no free and mobile charge carriers but only

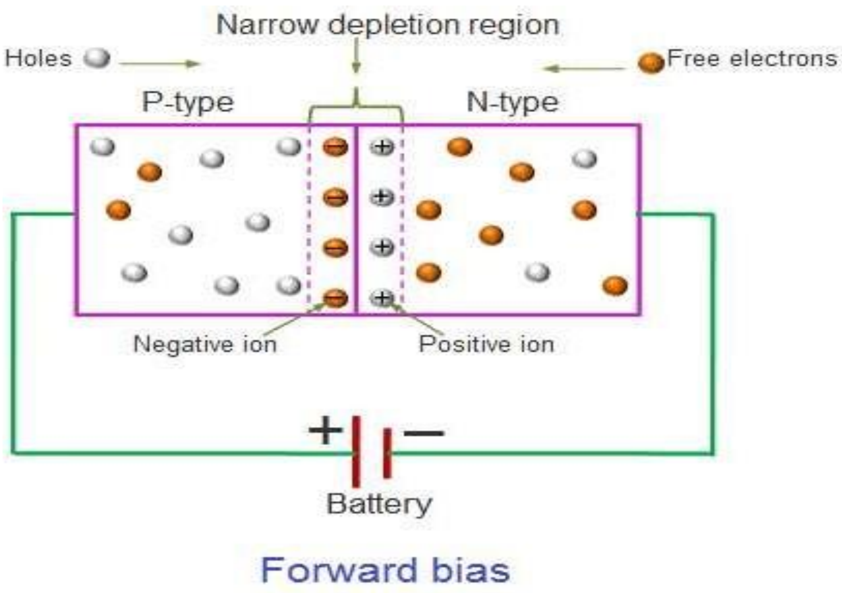
  fixed and immobile ions.

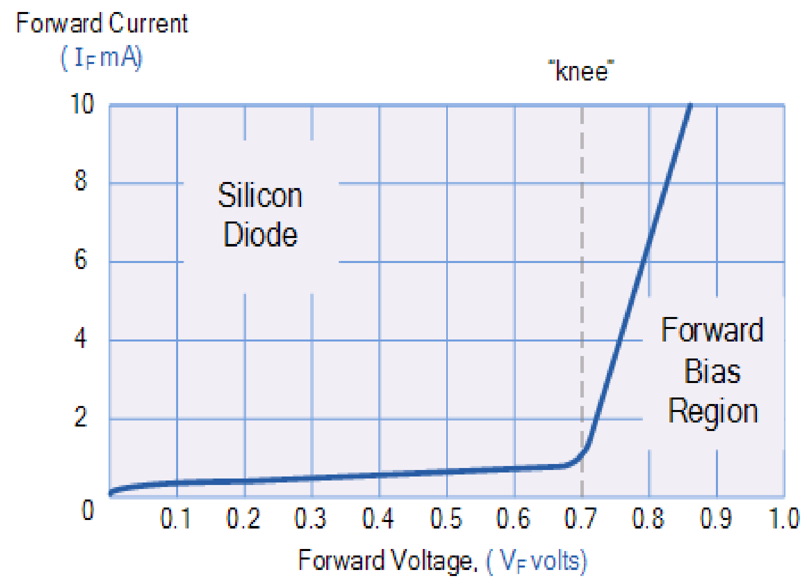
- Its width depends upon the doping level..
- Heavily doped……..thin depletion layer
- lightly doped……..thick depletion layer

Forward bias mode : positive terminal connected to p-region and negative terminal connected to n region.

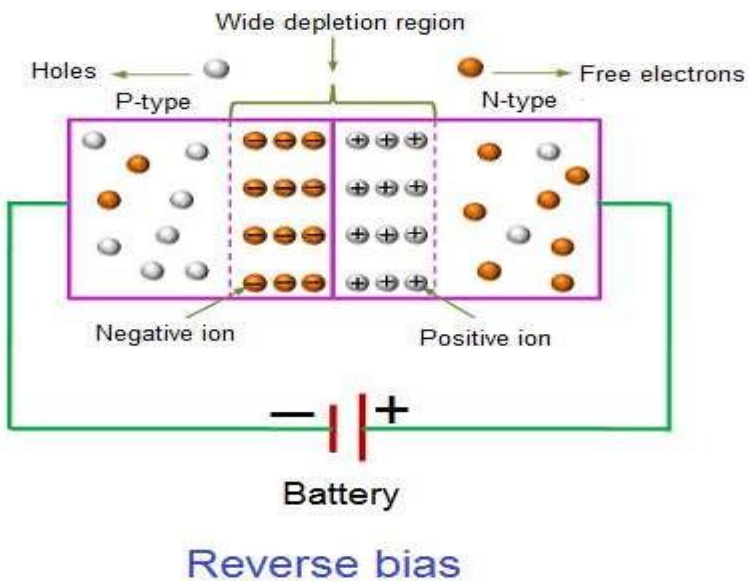
Reverse bias mode:negative terminal connected to p-region and positive terminal connected to n region

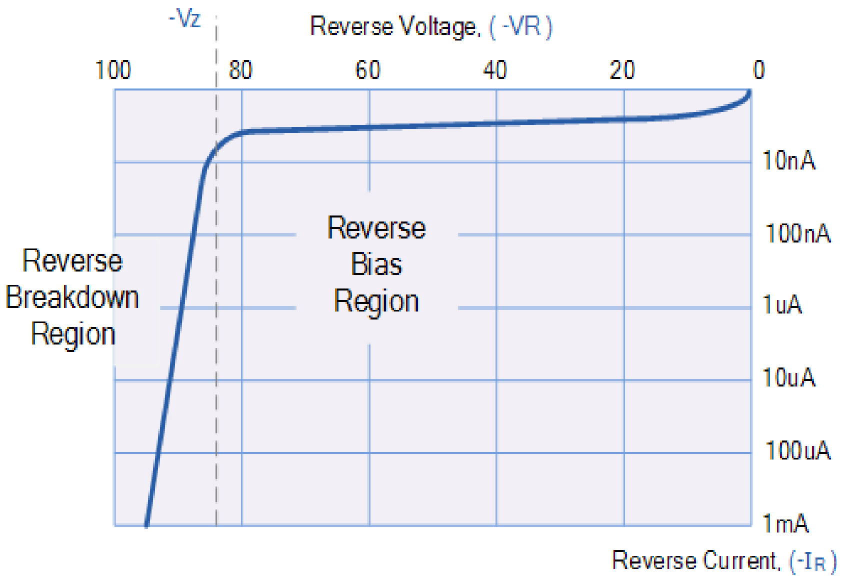
PN JUNCTION –FORWARD BIAS





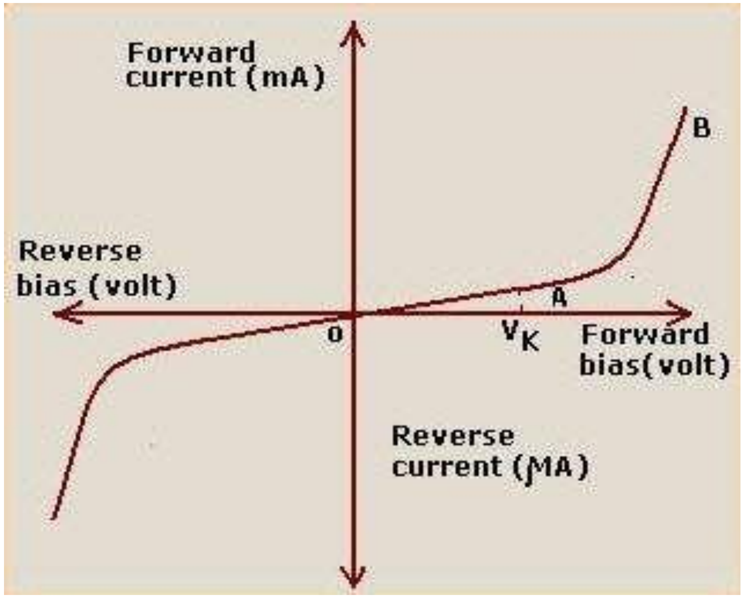
- It forcesthe majority charge carriers to move across the junction decreasing the width of the depletion layer.





- The free electrons and free holes are attracted towards the battery, hence depletion layer width increases.

V-I CHARACTERISTICS OF PN JUNCTION DIODE



V-I characteristics of PN junction diode

STATIC OR DC RESISTANCE

- The resistance of a diode at a particular operating point is called the dc or static resistance diode.
- The resistance of the diode at the operating point can be found simply by finding the corresponding levels of V_D and I_D.
- It can be determined using equation

$$R_D = V_D / I_D$$

- The lower current through a diode the higher the dc resistance level

STATIC OR DC RESISTANCE

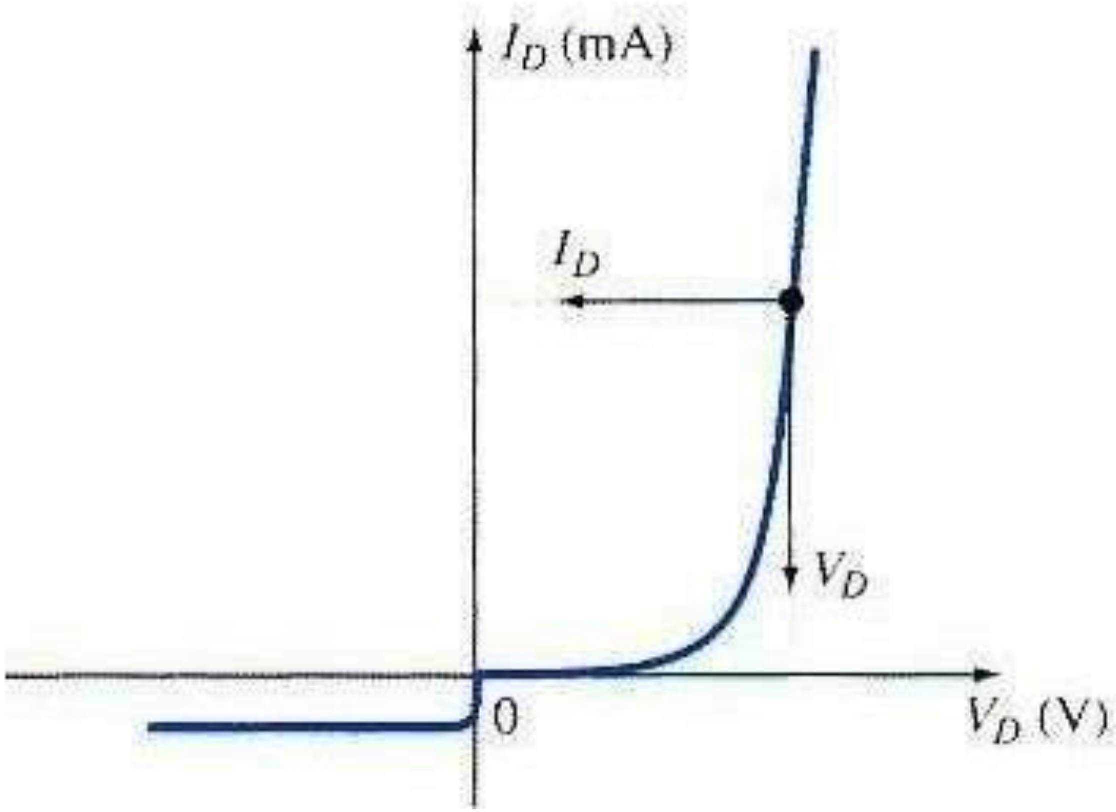


Fig: Static resistance curve

DYNAMIC OR AC RESISTANCE

- Static resistance is using dc input. If the input is sinusoidal the scenario will change.

- The ac resistance is determinedby straight line drawn betweenthe two intersections of the maximum and minimum values of input voltage.

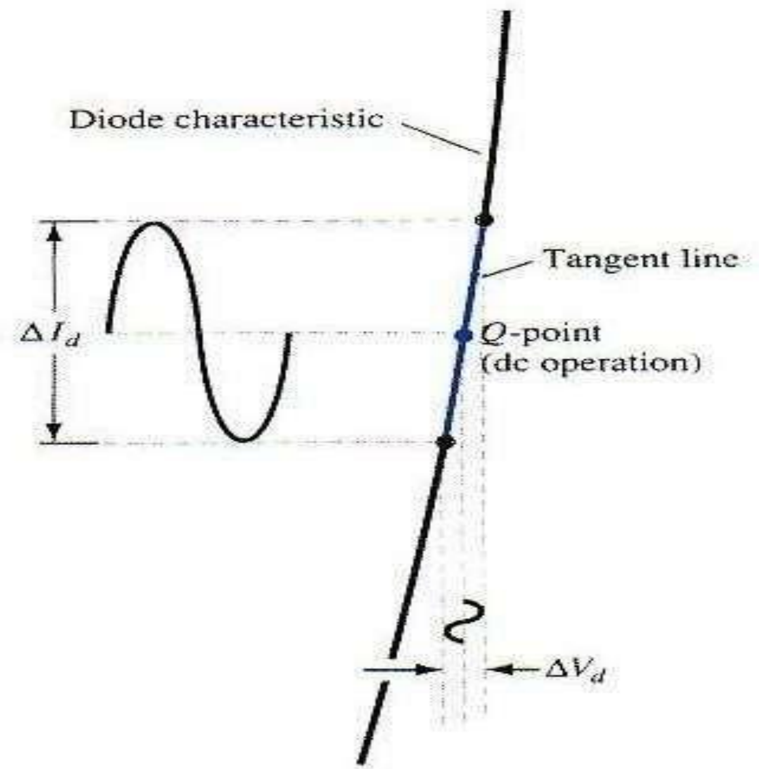
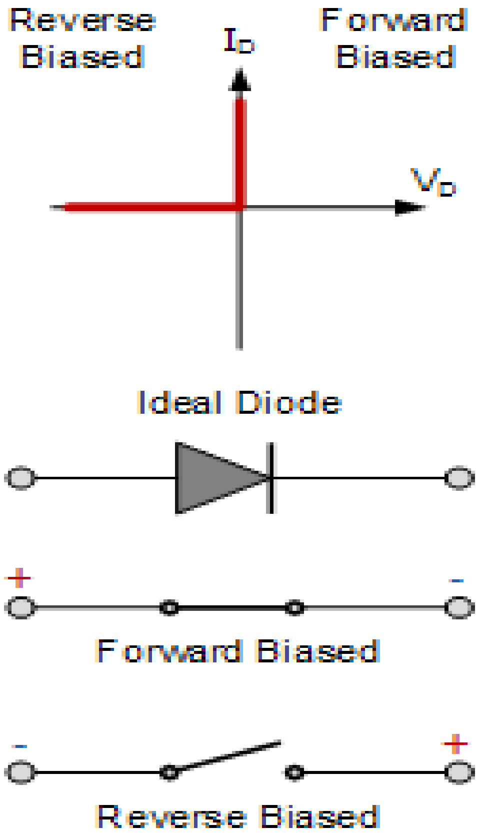


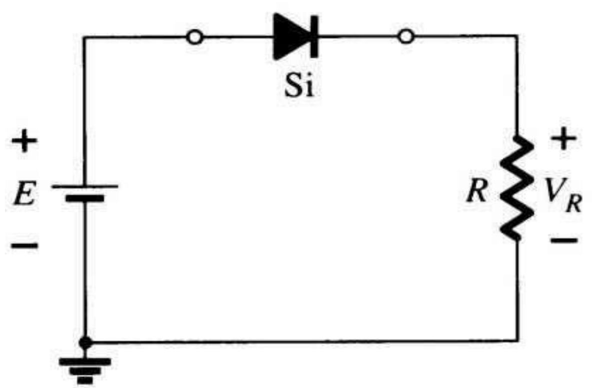
Fig: Dynamic resistance curve

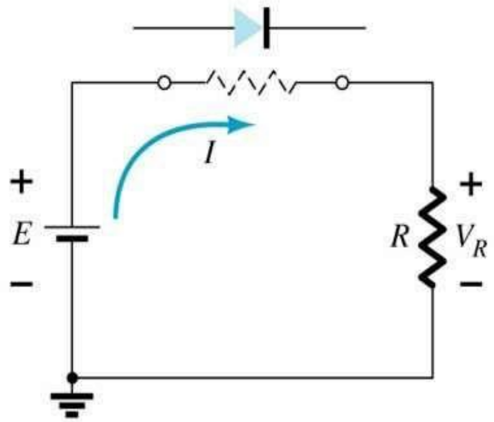
PN JUNCTION DIODE CHARACTERISTICS

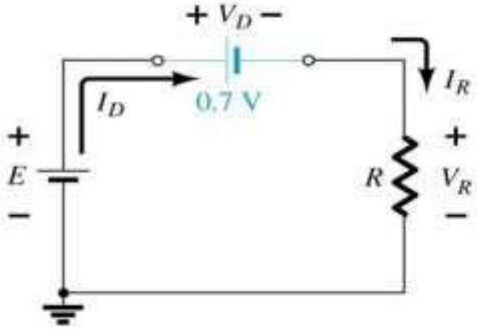


DIODE EQUIVALENT CIRCUIT

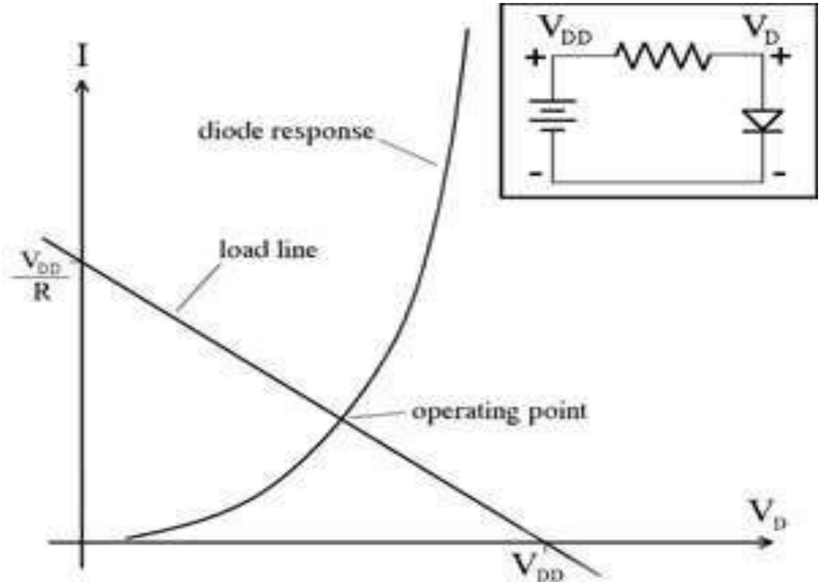
When a diode is F.B, we can use the approximate model for the on state







PN DIODE- LOAD LINE ANALYSIS



A load line is a line drawn on the characteristic curve, a graph of the current vs. voltage in a nonlinear device like a diode.

- The curve shows the diode response (I vs V_D) while the straight line shows

the behavior of the linear part of the circuit:

$$I=(V_{DD}-V_D)/R.$$

- The point of intersection gives the actual current and voltage.

JUNCTION CAPACITANCE

In a p-n junction diode, two types of capacitance take place. They are,

- Transition capacitance (C_T)
- Diffusion capacitance (C_D)

Transition Capacitance C_T

The amount of capacitance changed with increase in voltage is called transition capacitance. The transition capacitance is also known as depletion region capacitance, junction capacitance or barrier capacitance.

TRANSITION CAPACITANCE C_T

The change of capacitance at the depletion region can be defined as the

change in electric charge per change in voltage.

$$C_T = dQ / dV$$

Where,

C_T = Transition capacitance

dQ = Change in electric charge

dV = Change in voltage

DIFFUSION CAPACITANCE (C_D)

Diffusion capacitance occurs in a forward biased p-n junction diode. Diffusion capacitance is also sometimes referred as storage capacitance. It is denoted as C_D.

The formula for diffusion capacitance is

$$C_D = dQ / dV$$

DIODE APPLICATIONS

common applications of diodes are

- Switches
- Rectifiers
- Clipper Circuits
- Clamping Circuits
- Reverse Current Protection Circuits
- In Logic Gates
- Voltage Multipliers

RECTIFIER

- A circuit that converts ac voltage of main supply into pulsating dc voltage

  using one or more PN junction diodes is called rectifier.
- Half Wave Rectifier
- Full Wave Rectifier
- Bridge Rectifier

Half Wave Rectifier

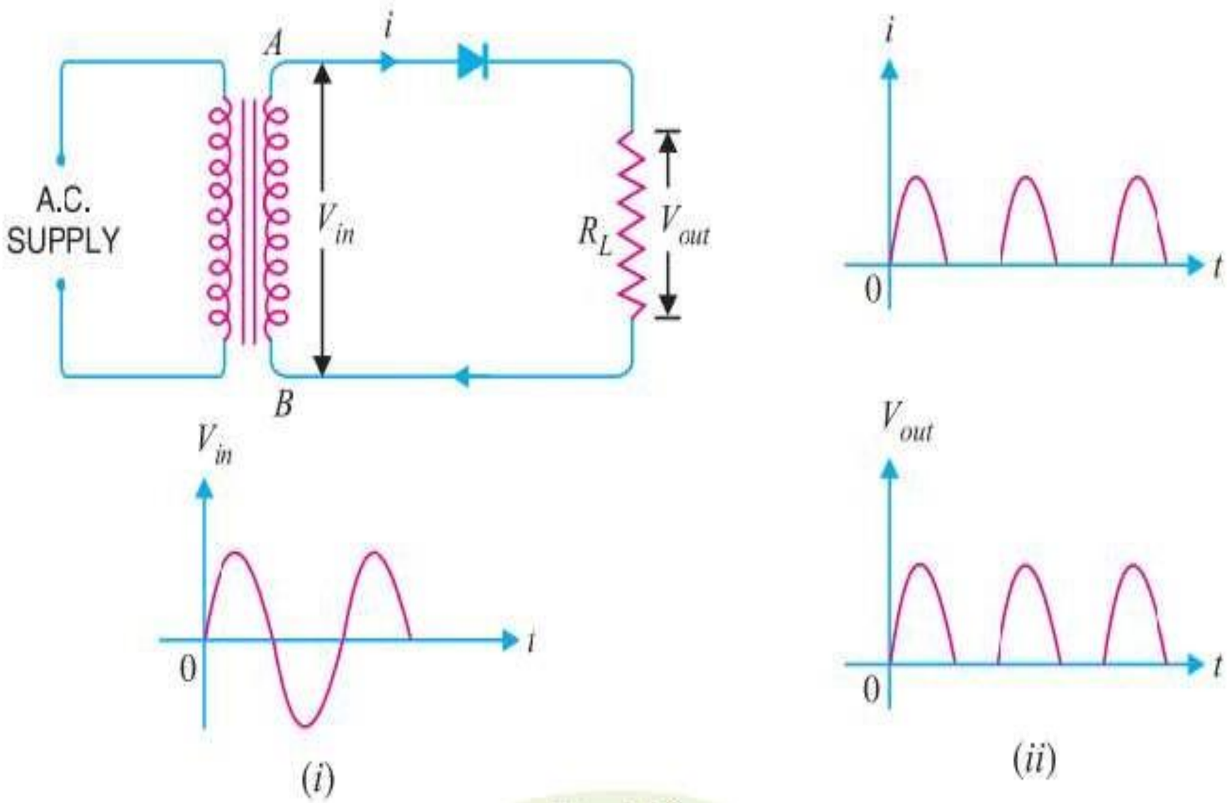
- The process of removing one-half the input signal to establish a dc level is called half-wave rectification.

- In Half wave rectification, the rectifier conducts current during positive

- half cycle of input ac signal only.

- Negative half cycle is suppressed.

Half Wave Rectifier



Average DC load Current (I_{DC}):

Mathematically, current waveform can be described as,

$$i_L = I_m \sin \omega t \qquad \text{for } 0 \le \omega t \le \pi$$

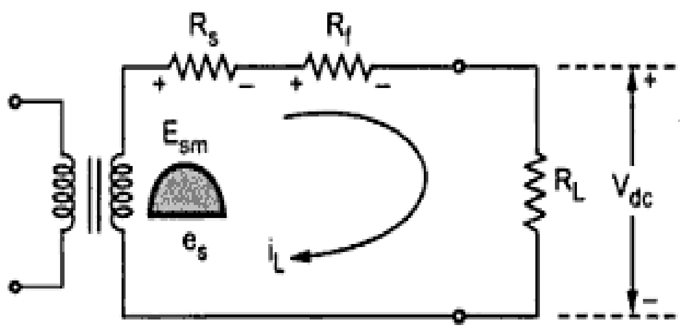
$$i_L = 0 \qquad \text{for } \pi \le \omega t \le 2\pi$$

I_m = peak value of load current

$$I_{DC} = \frac{1}{2\pi}\int_0^{2\pi} i_L \, d(\omega t) = \frac{1}{2\pi}\int_0^{2\pi} I_m \sin(\omega t)\, d(\omega t)$$

$$\boxed{I_{DC} = \frac{I_m}{\pi} = \text{average value}}$$

$$\boxed{I_m = \frac{E_{sm}}{R_f + R_L + R_s}}$$



(a) Equivalent circuit

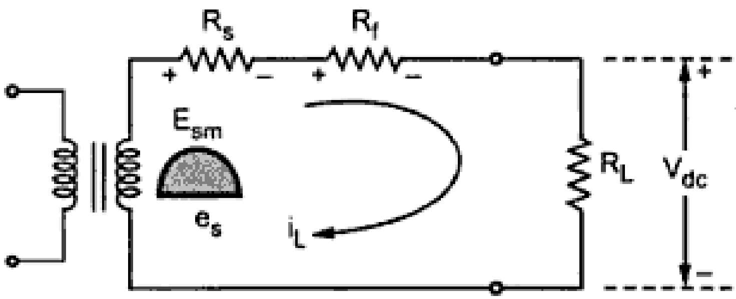
where R_s = resistance of secondary winding of transformer. If R_s is not given it should be neglected while calculating I_m.

Average DC voltage (Edc):

$$E_{DC} = I_{DC} R_L$$

$$E_{DC} = \frac{I_m}{\pi} R_L$$

$$= \frac{E_{sm}}{(R_f + R_L + R_s)\pi} R_L$$



(a) Equivalent circuit

But as R_f and R_s are small compared to R_L, $(R_f + R_s)/R_L$ is negligibly small compared to 1. So neglecting it we get,

$$\boxed{E_{DC} \approx \frac{E_{sm}}{\pi}}$$

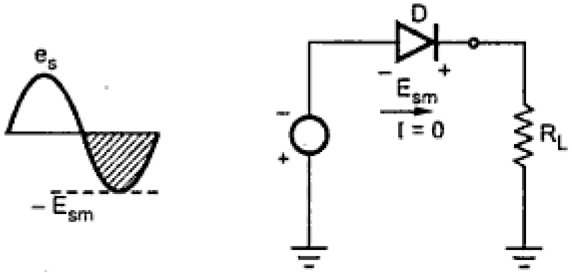
RMS Load Current (Irms):

$$I_{RMS} = \sqrt{\frac{1}{2\pi}\int_0^{\pi}(I_m \sin \omega t)^2\, d(\omega t)} \rightarrow \boxed{I_{RMS} = \frac{I_m}{2}}$$

RMS Load Voltage (Erms):

$$E_{L\ (RMS)} \approx \frac{E_{sm}}{2}$$

Peak Inverse Voltage (PIV):



PIV = Em

Diode must be selected based on the PIV rating and the circuit specification.

DC Power Delivered to the load:

$$\boxed{P_{DC} = E_{DC}\, I_{DC} = I_{DC}^2 R_L}$$

$$\text{D.C. Power output} = I_{DC}^2 R_L = \left[\frac{I_m}{\pi}\right]^2 R_L = \frac{I_m^{\,2}}{\pi^2} R_L$$

$$P_{DC} = \frac{I_m^{\,2}}{\pi^2} R_L$$

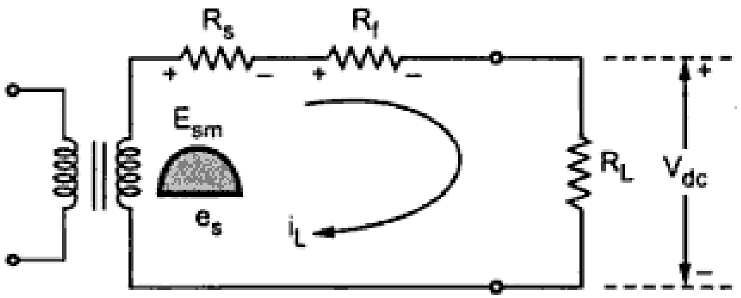
AC input power from transformer secondary:

The power input taken from the secondary of transformer is the power supplied to three resistances namely load resistance R_L , the diode resistance R_f and winding resistance R_s. The a.c. power is given by,

$$\boxed{P_{AC} = I^2_{RMS}[R_L + R_f + R_s]}$$

$$I_{RMS} = \frac{I_m}{2} \quad \text{for half wave,}$$

$$\boxed{P_{AC} = \frac{I^2_m}{4}[R_L + R_f + R_s]}$$



(a) Equivalent circuit

Rectifier Efficiency (η):

$$\eta = \frac{\text{D.C. output power}}{\text{A.C. input power}} = \frac{P_{DC}}{P_{AC}}$$

$$\eta = \frac{\frac{I_m^2}{\pi^2} R_L}{\frac{I_m^2}{4}[R_f + R_L + R_s]} = \frac{(4/\pi^2)R_L}{(R_f + R_L + R_s)}$$; η= 40.6 %

Under best conditions (no diode loss) only 40.6% of the ac input power is converted into dc power.

The rest remains as the ac power in the load

Ripple Factor:

$$\text{Ripple factor} \quad \gamma = \frac{\text{R.M.S. value of a.c. component of output}}{\text{Average or d.c. component of output}}$$

(or)

$$\text{Ripple factor} = \frac{I_{ac}}{I_{DC}}$$

$$\gamma = \sqrt{\left(\frac{I_{RMS}}{I_{DC}}\right)^2 - 1}$$

$$\gamma = \sqrt{\left[\frac{\left(\frac{I_m}{2}\right)}{\left(\frac{I_m}{\pi}\right)}\right]^2 - 1} = \sqrt{\frac{\pi^2}{4} - 1} = \sqrt{1.4674} \rightarrow \boxed{\gamma = 1.211}$$

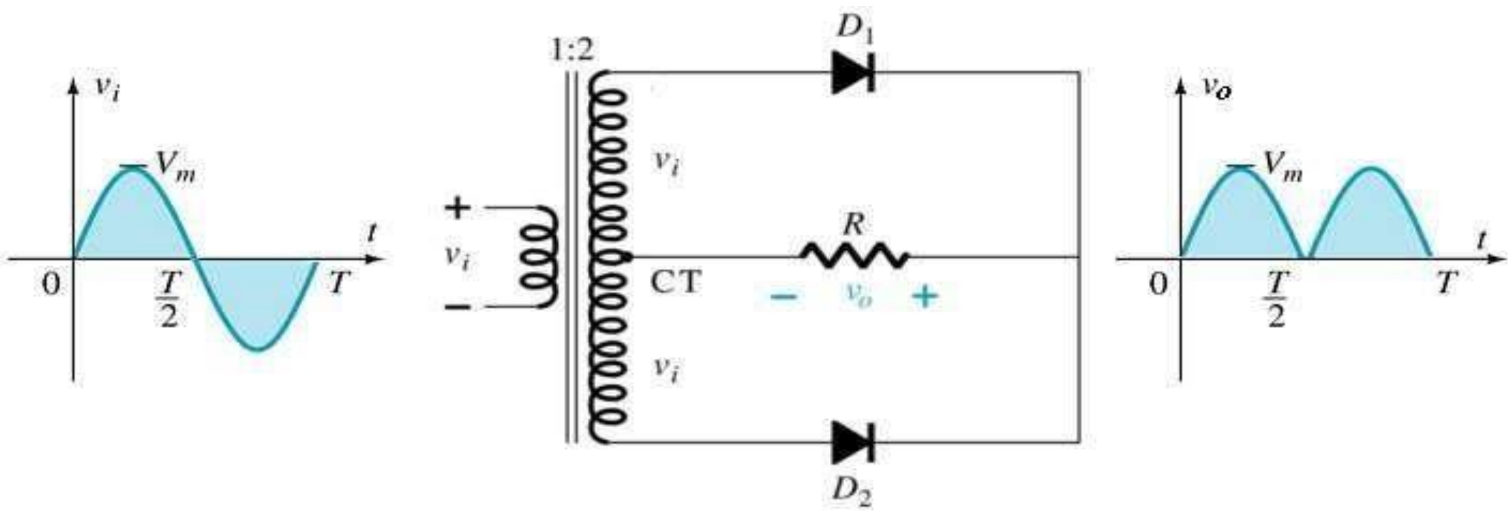
This indicates that the ripple content in the output are 1.211 times the dc component.

i.e. 121.1 % of dc component.

Disadvantage of HWR:

- The ripple factor of half wave rectifier is 1.21, which is quite high.
- The output contains lot of ripples
- The maximum theoretical efficiency is 40%.
- The practical value will be quite less than this.
- This indicates that HWR is quite inefficient.

FULL-WAVE RECTIFIER

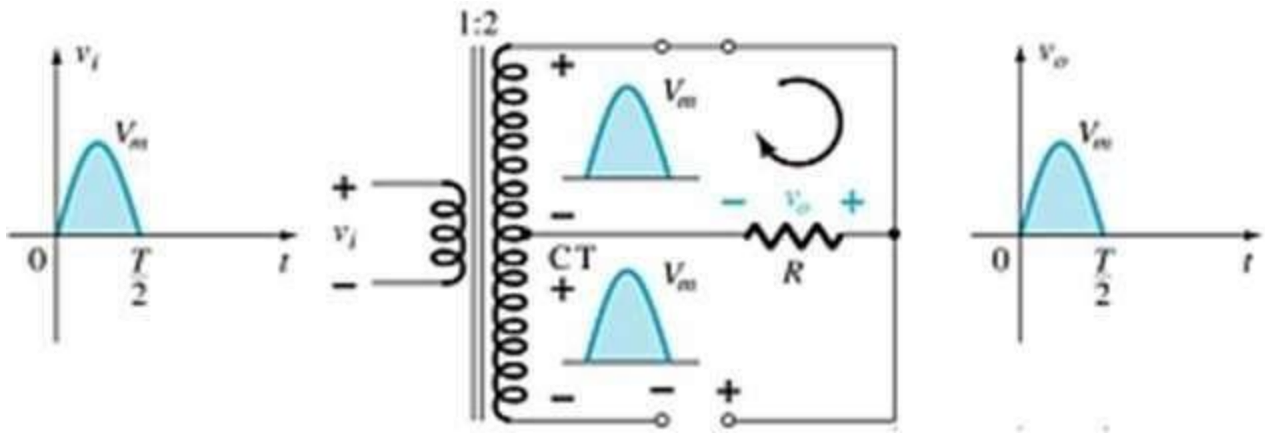


- The full wave rectifier circuit consists of two *power diodes* connected to a single load resistance (R_L) with each diode taking it in turn to supply current to the load.
- When point A of the transformer is positive with respect to point C, diode D_1 conducts in the forward direction as indicated by the arrows.

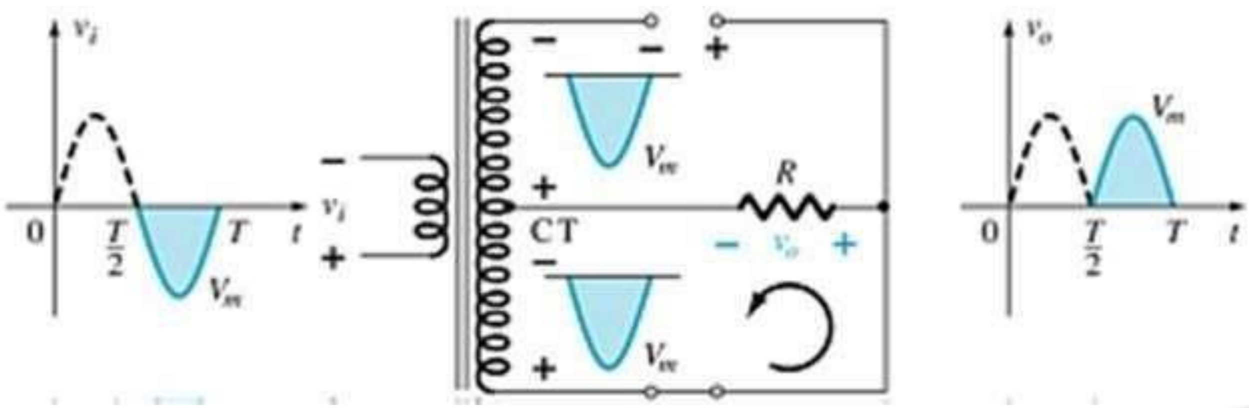
FULL-WAVE RECTIFIER

- When point B is positive (in the negative half of the cycle) with respect to point C, diode D_2 conducts in the forward direction and the current flowing through resistor R is in the same direction for both half-cycles.
- As the output voltage across the resistor R is the phasor sum of the two waveforms combined, this type of full wave rectifier circuit is also known as a “bi-phase” circuit.

FULL-WAVE RECTIFIER



Current Flow during the positive half of the input cycle



Current Flow during the negative half of the input cycle

FULL-WAVE RECTIFIER

Average DC current:

$$I_{av} = I_{DC} = \frac{1}{\pi}\int_0^{\pi} i_L \, d(\omega t) = \frac{1}{\pi}\int_0^{\pi} I_m \sin \omega t \, d\omega t$$

$$\boxed{I_{DC} = \frac{2I_m}{\pi}}$$ for full wave rectifier

Average (DC): Voltage $E_{DC} = I_{DC}R_L = \dfrac{2I_mR_L}{\pi}$

Substituting value of I_{av}

$$E_{DC} = \frac{2E_{sm}R_L}{\pi[R_f + R_s + R_L]} = \frac{2E_{sm}}{\pi\left[1 + \dfrac{R_f + R_s}{R_L}\right]}$$

But as R_f and $R_s << R_L$ hence $\dfrac{R_f + R_s}{R_L} << 1$

$$\boxed{E_{DC} = \frac{2E_{sm}}{\pi}}$$

FULL-WAVE RECTIFIER

RMS Load Current (Irms):

$$I_{RMS} = \sqrt{\frac{1}{\pi}\int_0^{\pi} i_L^2 d(\omega t)} \rightarrow I_{RMS} = \sqrt{\frac{1}{\pi}\int_0^{\pi} [I_m \sin \omega t]^2 d(\omega t)} \rightarrow \boxed{I_{RMS} = \frac{I_m}{\sqrt{2}}}$$

RMS Load Voltage:

$$\boxed{E_{L\,(RMS)} = I_{RMS} R_L = \frac{I_m}{\sqrt{2}} R_L}$$

DC Output Power:

$$\text{D.C. Power output} = E_{DC} I_{DC} = I_{DC}^2 R_L$$

$$P_{DC} = I_{DC}^2 R_L = \left(\frac{2I_m}{\pi}\right)^2 R_L$$

$$P_{DC} = \frac{4}{\pi^2} I_m^2 R_L$$

AC input power (Pac):

The a.c. power input is given by,

$$\therefore \quad \boxed{P_{AC} = I_{RMS}^2(R_f + R_s + R_L) = \left(\frac{I_m}{\sqrt{2}}\right)^2 (R_f + R_s + R_L)}$$

$$\therefore \quad P_{AC} = \frac{I_m^2(R_f + R_s + R_L)}{2}$$

Rectifier Efficiency (η):

$$\boxed{\eta = \frac{P_{DC}\ \text{output}}{P_{AC}\ \text{input}}} \Rightarrow \eta = \frac{\frac{4}{\pi^2} I_m^2 R_L}{\frac{I_m^2(R_f + R_s + R_L)}{2}} \Rightarrow \eta = \frac{8\ R_L}{\pi^2(R_f + R_s + R_L)}$$

But if $R_f + R_s << R_L$, neglecting it from denominator

$$\eta = \frac{8\ R_L}{\pi^2 (R_L)} = \frac{8}{\pi^2}$$

$$\therefore \quad \boxed{\%\ \eta_{max} = \frac{8}{\pi^2} \times 100 = 81.2\ \%}$$

Ripple Factor:

$$\text{Ripple factor} = \sqrt{\left[\frac{I_{RMS}}{I_{DC}}\right]^2 - 1}$$

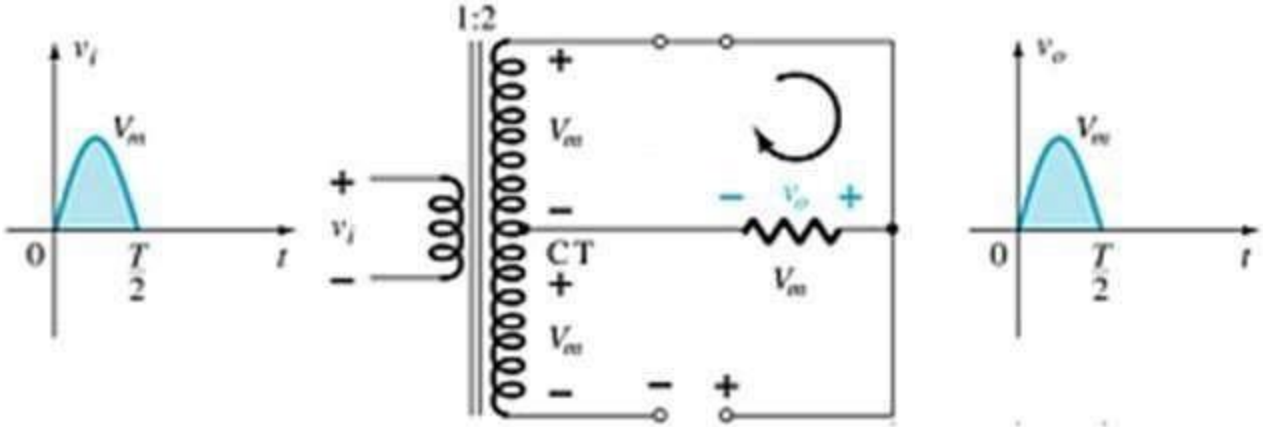
For full wave $I_{RMS} = I_m/\sqrt{2}$ and $I_{DC} = 2I_m/\pi$

$$\text{Ripple factor} = \sqrt{\left[\frac{I_m/\sqrt{2}}{2I_m/\pi}\right]^2 - 1} = \sqrt{\frac{\pi^2}{8} - 1}$$

$$\boxed{\text{Ripple factor} = \gamma = 0.48}$$

FULL-WAVE RECTIFIER

Peak Inverse Voltage:



PIV of diode = 2 E_{sm}

FULL-WAVE RECTIFIER

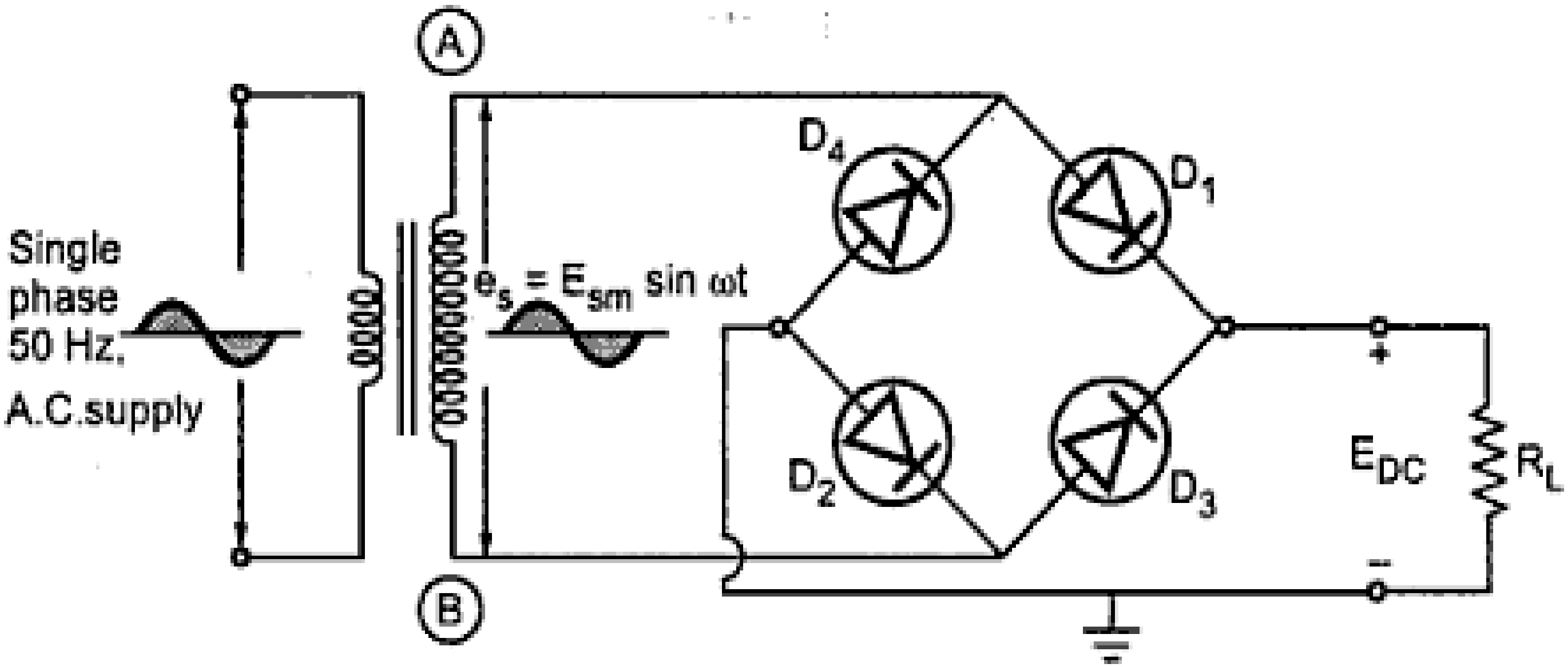
Advantages of Full Wave Rectifier:

- Efficiency is higher.
- The large dc power output
- The ripple factor is less

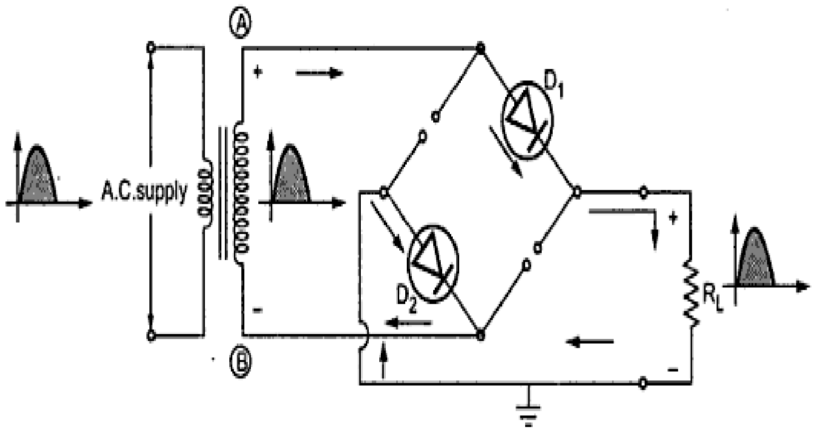
Disadvantages of Full Wave Rectifier:

- PIV rating of diode is higher.
- Higher PIV diodes are larger in size and costlier.
- The cost of center tap transformer is high.

BRIDGE RECTIFIER

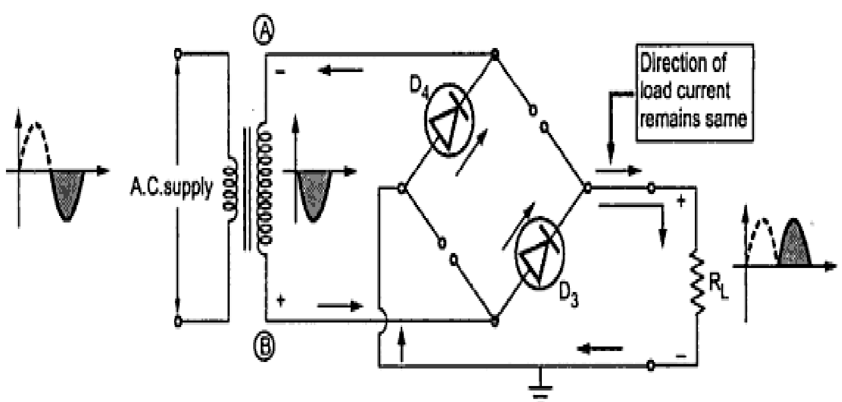


WORKING OF BRIDGE RECTIFIER



Current flow during positive half cycle

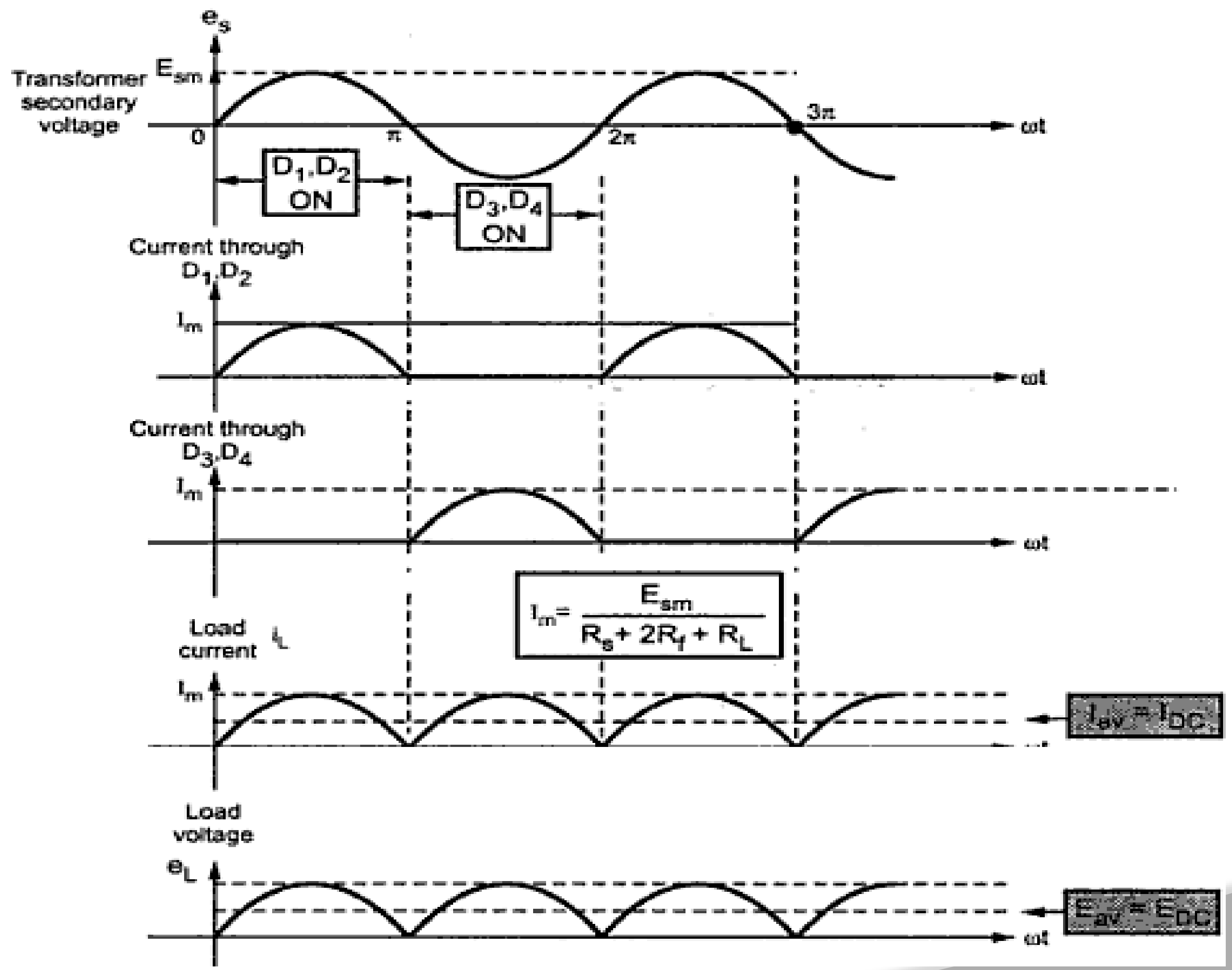
- During the positive half cycle of secondary voltage, the diodes D1 and D2 are forward-biased, but diodes D3 and D4 do no conduct. The current is through D1, R, D2 and secondary winding.



Current flow during negative half cycle

- During the negative half cycle, the diodes D3 and D4 are forward-biased, but diodes D1 and D2 do not conduct. The current is through D3, secondary winding, D4 and R.

BRIDGE RECTIFIER WAVEFORMS



BRIDGE RECTIFIER PARAMETERS

$$I_{DC} = \frac{2I_m}{\pi} \quad \text{and} \quad I_{RMS} = \frac{I_m}{\sqrt{2}}$$

$$E_{DC} = I_{DC} R_L = \frac{2E_{sm}}{\pi}$$

$$P_{DC} = I_{DC}^2 R_L = \frac{4}{\pi^2} I_m^2 R_L$$

$$P_{AC} = I_{RMS}^2 (R_s + 2R_f + R_L) = \frac{I_m^2 (2R_f + R_s + R_L)}{2}$$

$$\eta = \frac{8R_L}{\pi^2 (R_s + 2R_f + R_L)}, \quad \% \eta_{max} = 81.2\%$$

$$\gamma = 0.48$$

RECTIFIERS WITH CAPACITIVE FILTER

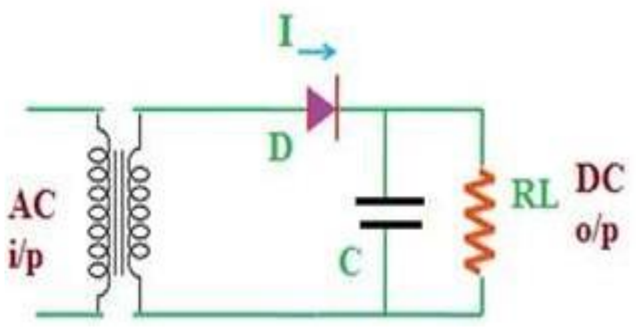


Fig: H/W rectifier with filter

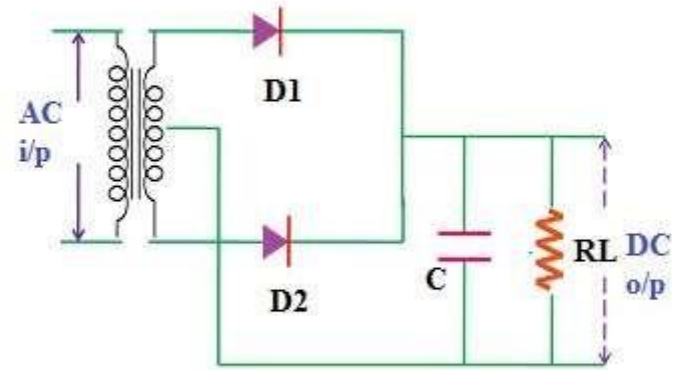
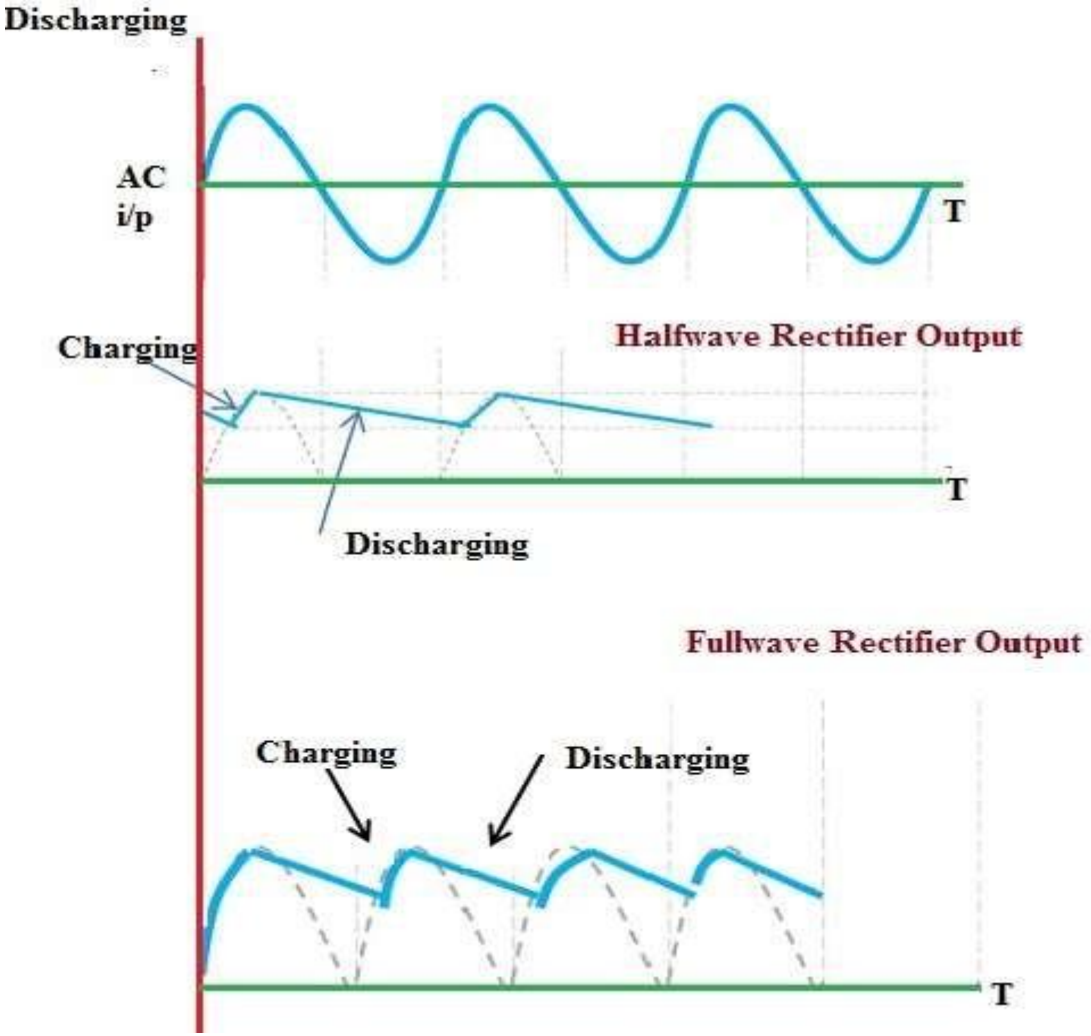


Fig: F/W rectifier with filter

- In full wave rectifier circuit using a capacitor filter, the capacitor C is located across the RL load resistor. The working of this rectifier is almost the same as a half wave rectifier.

RECTIFIERS WITH CAPACITIVE FILTER WAVEFORMS



MODULE-II

BIPOLAR JUNCTION TRANSISTOR

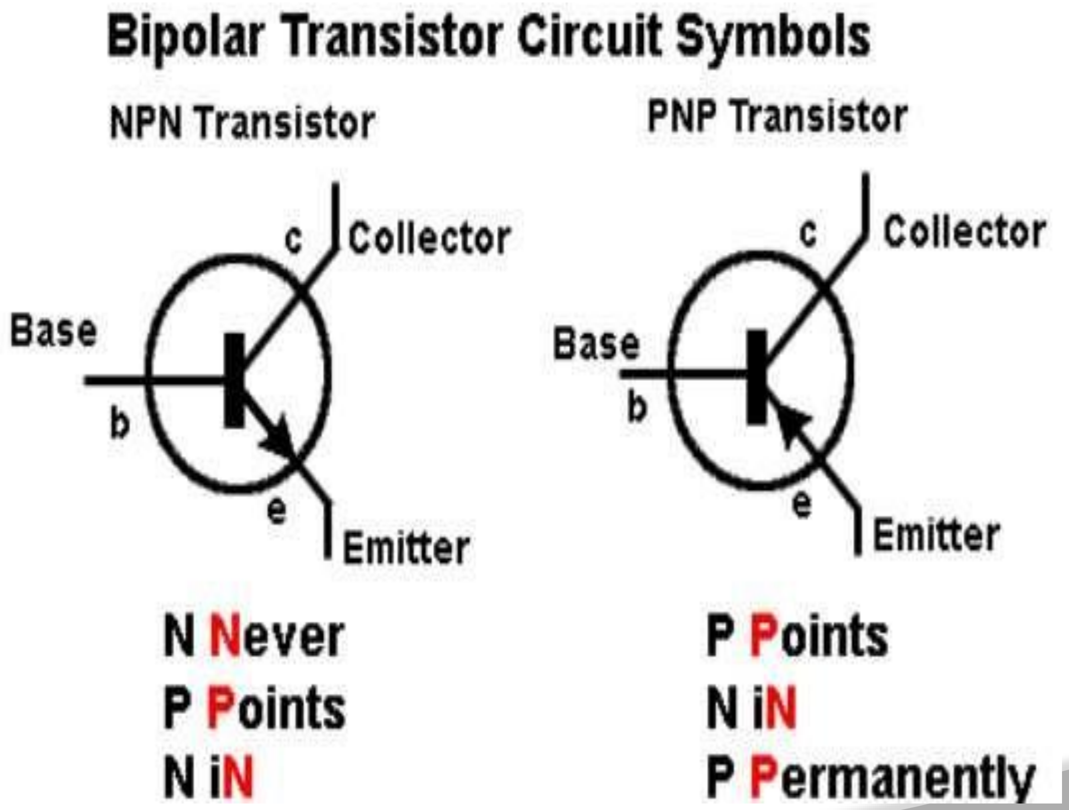
Principle of Operation and characteristics - Common Emitter, Common Base, Common Collector Configurations, Operating point, DC & AC load lines, Transistor Hybrid parameter model, Determination of h- parameters from transistor characteristics, Conversion of h-parameters.

CONTENTS

- Introduction
- Common Emitter Configuration
- Common Base Configuration
- Common Collector Configuration
- Operating point, DC & AC load lines
- Transistor Hybrid parameter model
- Conversion of h-parameters.

INTRODUCTION

Transistor is a device that can be used as either an amplifier or a switch. Transistor is current controlling device.



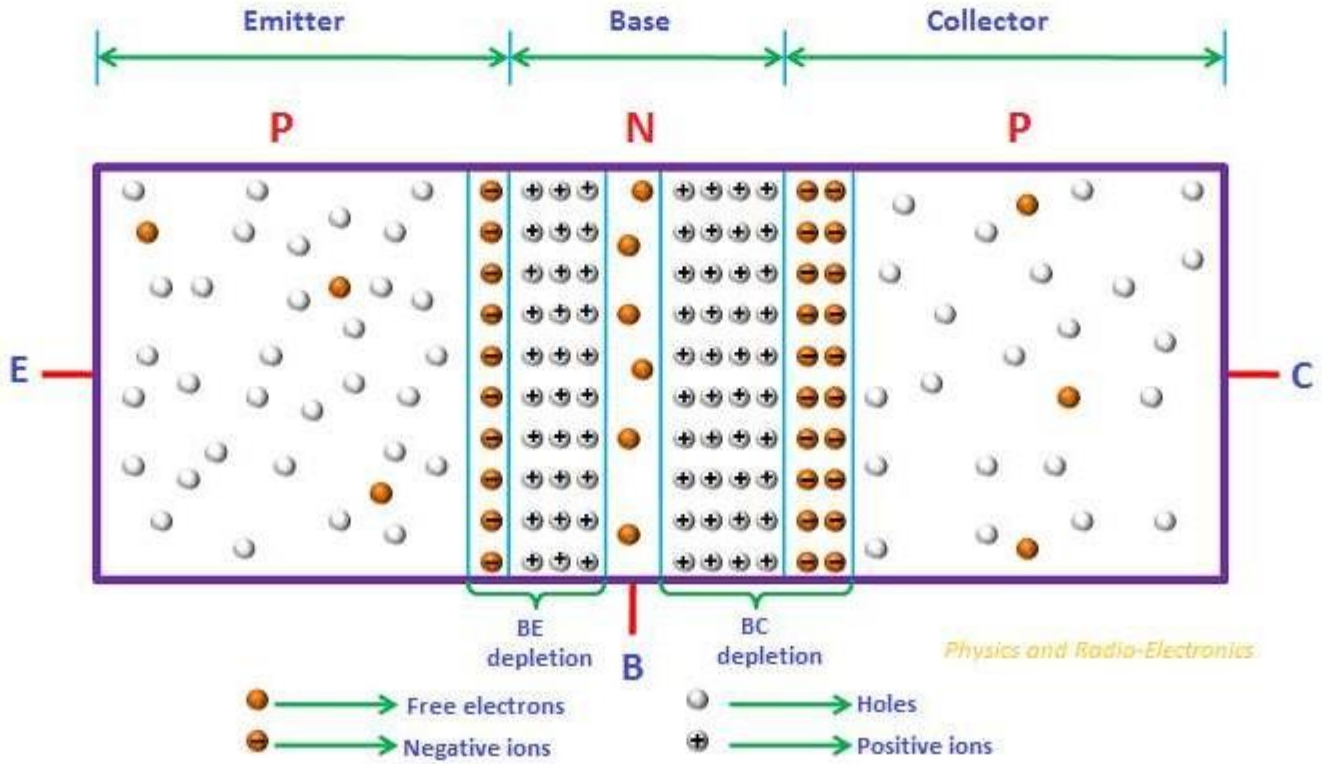
INTRODUCTION

- The three layers of BJT are called Emitter, Base and Collector
- Base is very thin compared to the other two layers
- Base is lightly doped. Emitter is heavily doped. Collector is moderately doped
- NPN – Emitter and Collector are made of N-type semiconductors; Base is P-type
- PNP – Emitter and Collector are P-type, Base is N-type
- Both types (NPN and PNP) are extensively used, either separately or in the same circuit
- BJT has two junctions – Emitter-Base (EB) Junction
- and Collector-Base (CB) Junction
- The device is called "**bipolar junction transistor**" because current is due to motion of two types of charge carriers – free electrons & holes
- Transistor Analogous to two diodes connected back-to-back: EB diode and CB diode

TRANSISTOR OPERATION

- Operation of **NPN** transistor is discussed here
- For normal operation (amplifier application)
- – EB junction should be forward biased
- – CB junction should be reverse biased
- Depletion width at EB junction is narrow (forward biased)
- Depletion width at CB junction is wide (reverse biased)
- When EB junction is forward biased, free electrons from emitter region drift towards base region
- Some free electrons combine with holes in the base to form small base current

TRANSISTOR OPERATION



Common Base (CB) Configuration of Transistor

- In CB Configuration, the base terminal of the transistor will be common between the input and the output terminals as shown by Fig.4. This configuration offers low input impedance, high output impedance, high resistance gain and high voltage gain.

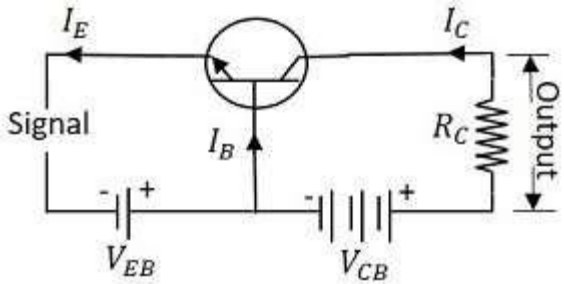


Fig.4: NPN Transistor in CB Configuration

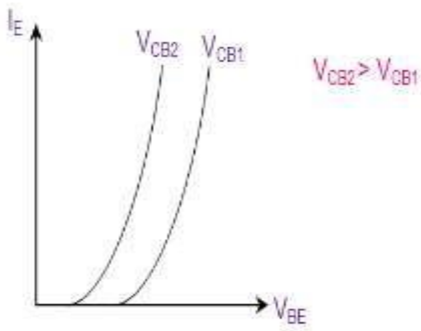


Fig.5: CB Configuration I/P Characteristics

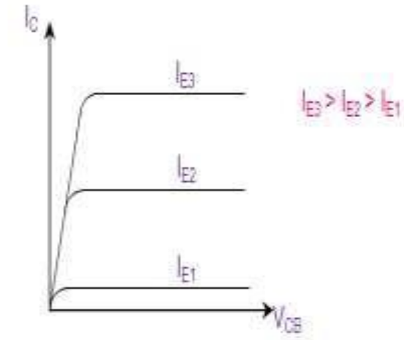


Fig.6: CB Configuration O/P Characteristics

- This leads to the expression for the input resistance as

$$R_{in} = \frac{\Delta V_{BE}}{\Delta I_E}\Big|_{V_{CB}=constant}$$

- output resistance can be obtained as

$$R_{out} = \frac{\Delta V_{CB}}{\Delta I_C}\Big|_{I_E=constant}$$

- The current gain has a value less than 1 and can expressed as

$$\alpha = \frac{\Delta I_C}{\Delta I_E}\Big|_{V_{CB}=constant}$$

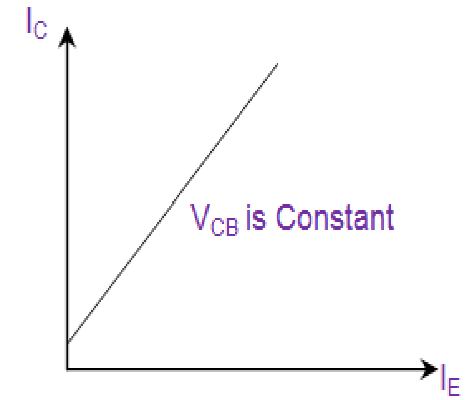


Fig.7: CB Configuration Current Transfer Characteristics

Common Collector (CC) Configuration of Transistor

- This offers high input impedance, low output impedance, voltage gain less than one and a large current gain.

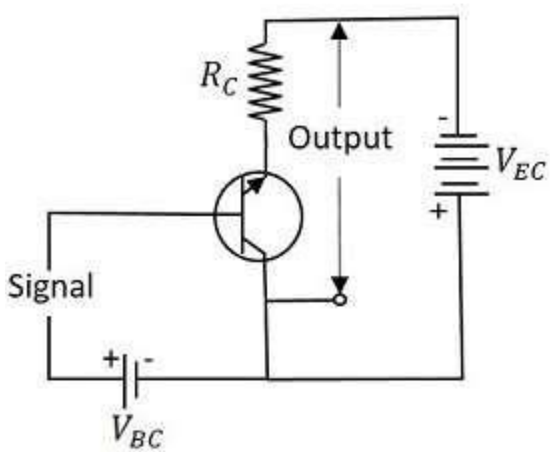


Fig.8: NPN Transistor in CC Configuration

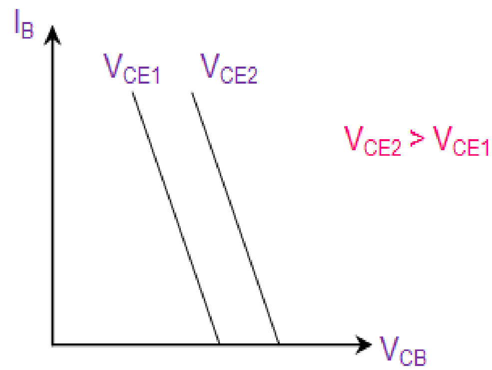


Fig.9: CC Configuration I/P Characteristics

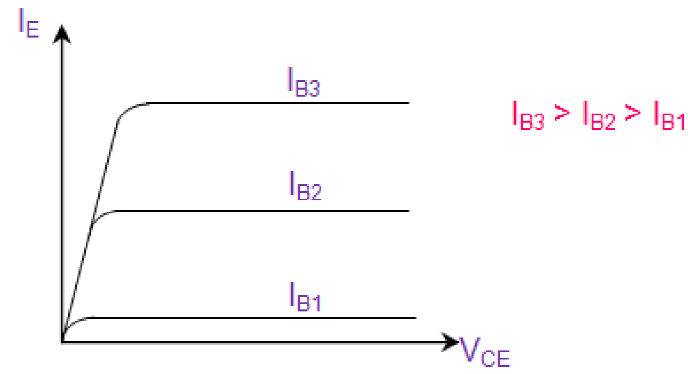


Fig.10: CC Configuration O/P Characteristics

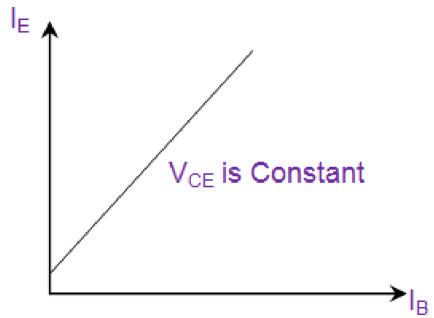


Fig.11: CC Configuration Current Transfer Characteristics

Common Emitter (CE) Configuration of Transistor

- In this configuration, the emitter terminal is common between the input and t he output terminals as shown by Fig.12. This configuration offers medium input impedance, medium output impedance, medium current gain and

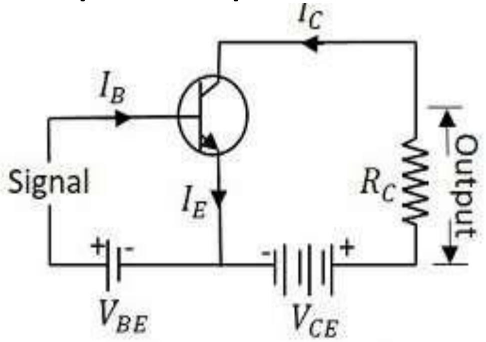


Fig.12 NPN Transistor in CE Configuration

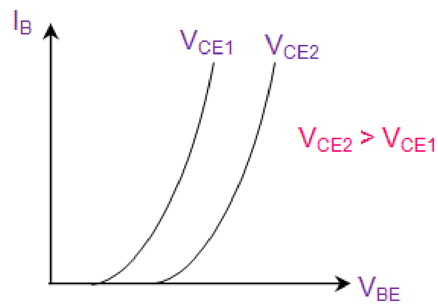


Fig.13: CE Configuration I/P Characteristics

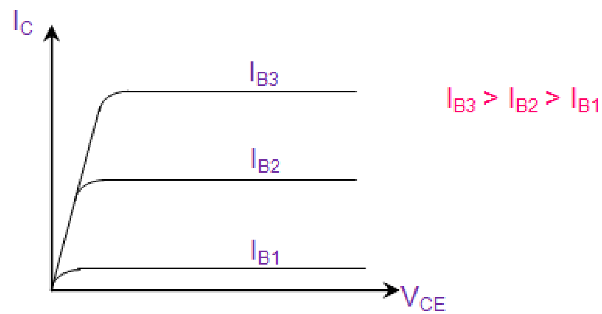


Fig.14: CE Configuration O/P Characteristics

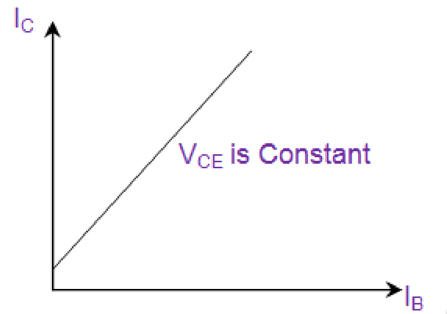


Fig.15: CE Configuration Current Transfer Characteristics

COMPARISON OF TRANSISTOR CONFIGURATIONS

S. No.	Characteristic	Common base	Common emitter	Common collector
1.	Input resistance	Low (about 100 Ω)	Low (about 750 Ω)	Very high (about 750 kΩ)
2.	Output resistance	Very high (about 450 kΩ)	High (about 45 kΩ)	Low (about 50 Ω)
3.	Voltage gain	about 150	about 500	less than 1
4.	Applications	For high frequency applications	For audio frequency applications	For impedance matching
5.	Current gain	No (less than 1)	High (β)	Appreciable

DC Load Line

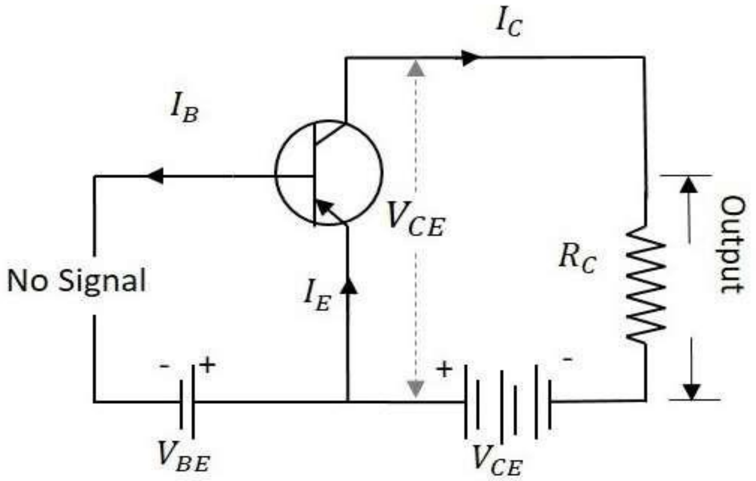


Fig.17: CE Amplifier circuit with no I/P signal

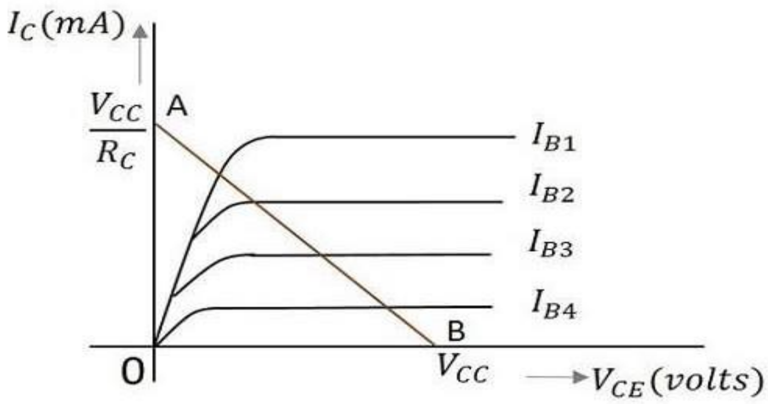


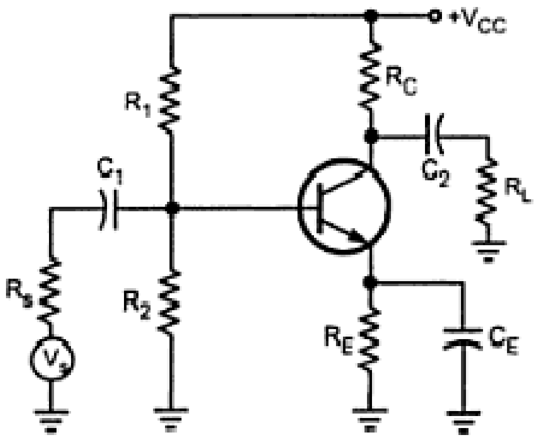
Fig.18: CE O/P characteristics with DC load line

- The value of collector emitter voltage at any given time will be

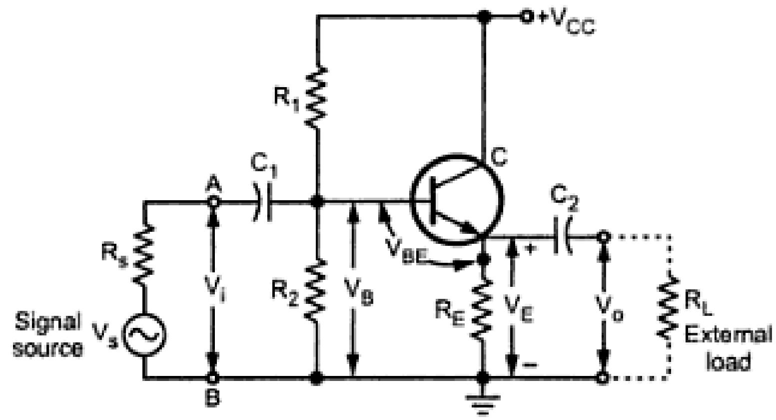
$$V_{CE}=V_{CC}-I_C R_C$$

- The fig.18 shows the DC load line.
- To obtain the load line, two points be A and B of the straight line are to be determined.

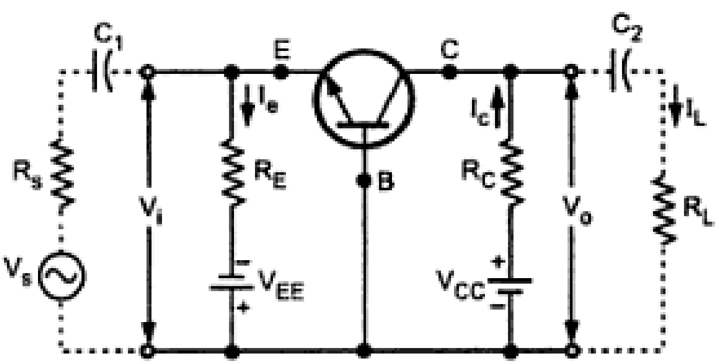
CE, CC, & CB Amplifiers



Practical common emitter amplifier circuit

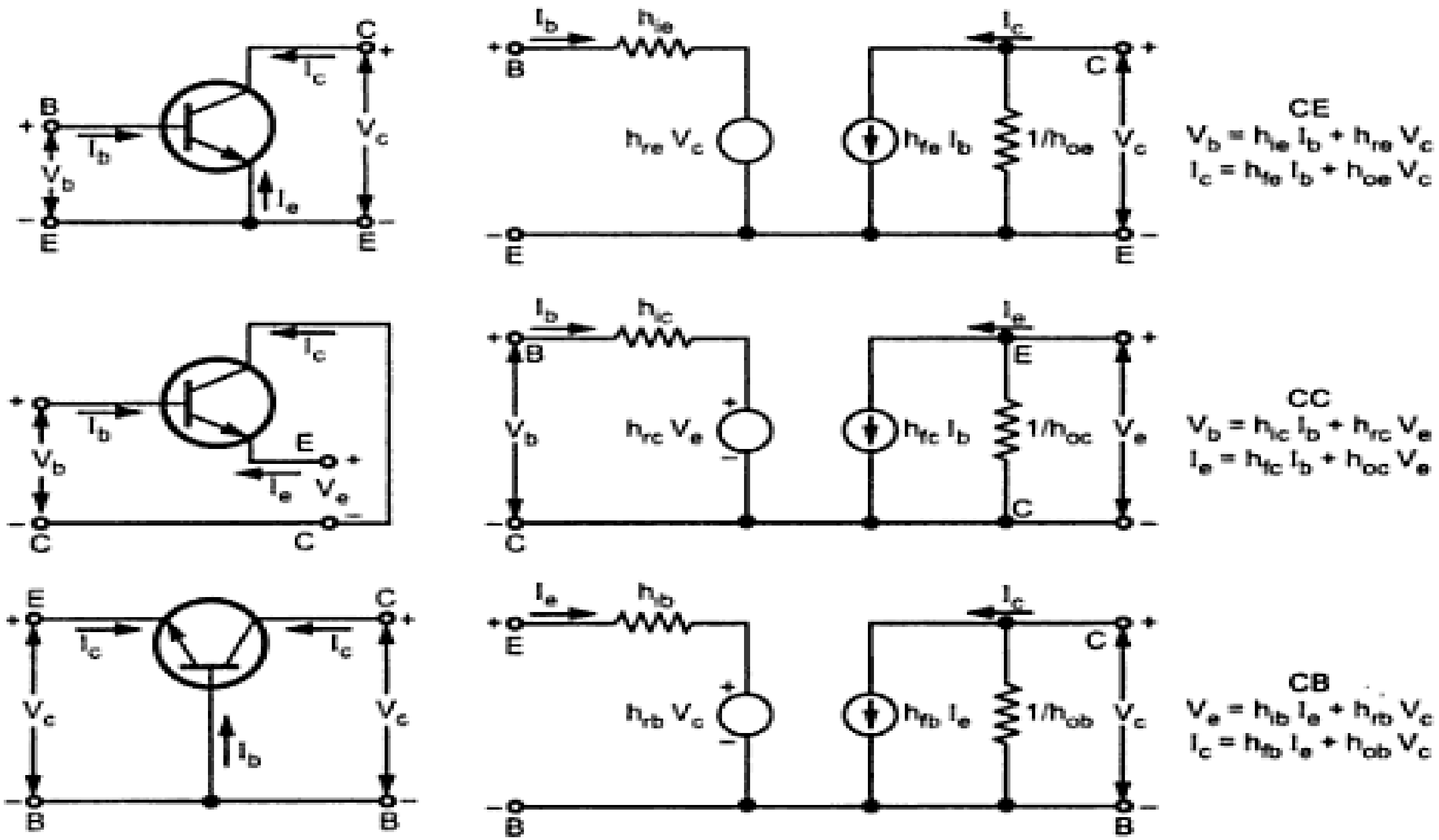


Common collector circuit



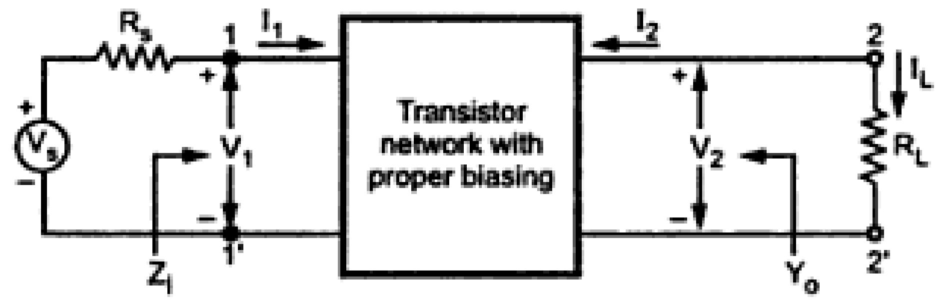
Common base circuit

Hybrid model of CE, CC, & CB Amplifiers

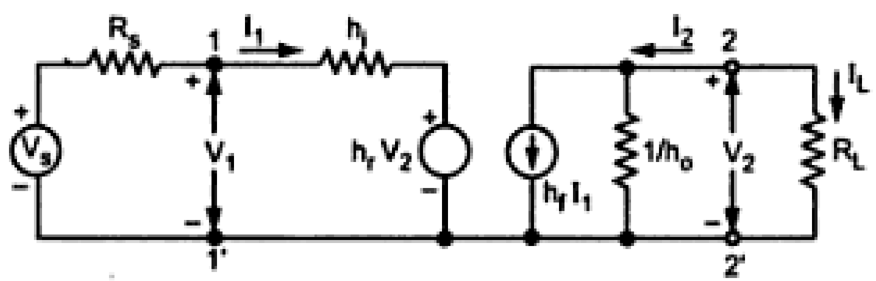


Transistor configurations and their hybrid models

Small Signal Analysis Of A Junction Transistor



Basic transistor amplifier



Transistor amplifier in its h-parameter model

Small signal analysis of transistor amplifier

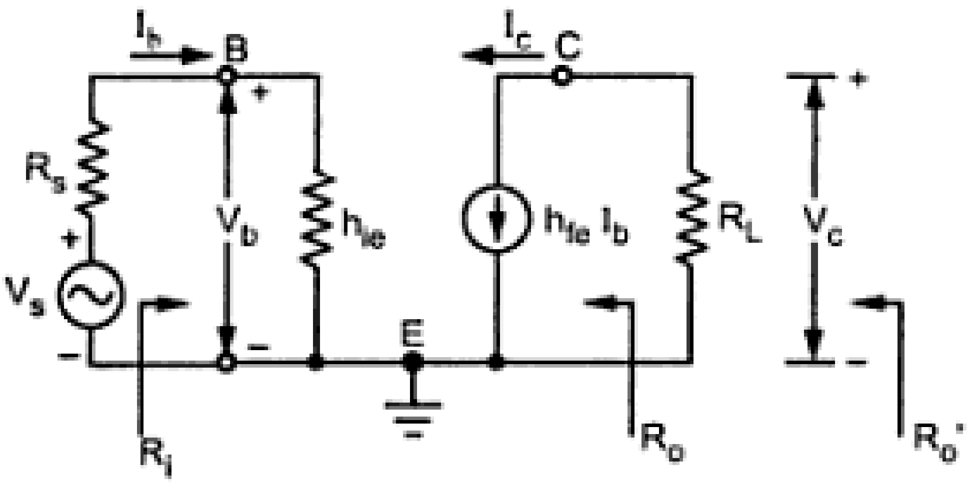
$A_i = -\dfrac{h_f}{1 + h_o R_L}$
$A_{is} = \dfrac{A_i R_s}{Z_i + R_s}$
$Z_i = h_i + h_r A_i R_L = h_i - \dfrac{h_f h_r}{h_o + Y_L}$
$A_v = \dfrac{A_i R_L}{Z_i}$
$A_{vs} = \dfrac{A_v R_i}{Z_i + R_s} = \dfrac{A_i R_L}{Z_i + R_s} = \dfrac{A_{is} R_L}{R_s}$
$Y_o = h_o - \dfrac{h_f h_r}{h_i + R_s} = \dfrac{1}{Z_o}$
$A_P = A_V A_i = A_i^2 \dfrac{R_L}{Z_i}$

TRANSISTOR HYBRID PARAMETER MODEL

Steps for ac analysis of a transistor circuit

- Draw the actual circuit diagram
- Replace Coupling Capacitors & emitter bypass capacitor by short circuit
- Replace dc source by a short circuit. In other words, short v_{cc} and ground lines
- Mark the points B(base), C(collector), E(emitter) on the circuit diagram and locate these points as the start of the equivalent circuit.
- Replace the transistor by its h-parameter model.

Approximate H-Model For CE Amplifier



Approximate CE model

Current Gain $A_i \approx -h_{fe}$

Input Impedance $R_i \approx h_{ie}$

Voltage Gain : $A_v = \frac{A_i R_L}{R_i} = \frac{A_i R_L}{h_{ie}}$

Output Impedance $Y_o = 0$

$R_o = \frac{1}{Y_o} = \infty$

$R'_o = R_o \| R_L = \infty \| R_L = R_L$

MODULE–III
NUMBERSYSTEMS

Number systems: Complements of Numbers, Codes- Weighted and Non-weighted codes and its Properties, Parity check code and Hamming code. **Boolean Algebra:** Basic Theorems and Properties, Switching Functions- Canonical and Standard Form, Algebraic Simplification, Digital Logic Gates, EX-OR gates, Universal Gates, Multilevel NAND/NOR realizations

CONTENTS

- Complements of Numbers
- Codes- Weighted and Non-weighted codes and its Properties
- Parity check code and Hamming code
- Basic Theorems and Properties
- Switching Functions- Canonical and Standard Form
- Algebraic Simplification
- Digital Logic Gates, EX-OR gates
- Universal Gates
- Multilevel NAND/NOR realizations

NUMBER SYSTEMS

- **Binary number system.**

  A method of representing **numbers** that has 2 as its base and uses only the digits 0 and 1.

  Ex:10100010

- **Decimal number system**

  A number system that uses a notation in which each number is expressed in base 10 by using one of the first nine integers or 0 in each place and letting each place value be a power of 10

  Numbers:0,1,2,3,4,5,6,7,8,9

- **Octal number system**

  The octal numbering system uses the numerals 0-1-2-3-4-5-6-7.

- **Hexa decimal number system**

  The hexadecimal numeral system, often shortened to "hex", is a numeral system made up of 16 symbols (base 16) they are 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E.

Binary to Decimal Conversion:

It is by the positional weights method . In this method,each binary digit of the no. is multiplied by its position weight . The product terms are added to obtain the decimal no

Example:

101011_2 =>

$$
\begin{aligned}
1 \times 2^0 &= 1\\
1 \times 2^1 &= 2\\
0 \times 2^2 &= 0\\
1 \times 2^3 &= 8\\
0 \times 2^4 &= 0\\
1 \times 2^5 &= 32
\end{aligned}
$$

43_{10}

Binary to Octal conversion:

Starting from the binary pt. make groups of 3 bits each, on either side of the binary pt, & replace each 3 bit binary group by the equivalent octaldigit.

Example:

$$1011010111_2 = ?_8$$

$$\begin{array}{cccc} 1 & 011 & 010 & 111 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 1 & 3 & 2 & 7 \end{array}$$

$$1011010111_2 = 1327_8$$

NUMBER BASE CONVERSION

Binary to Hexadecimal conversion:

For this make groups of 4 bits each , on either side of the binary pt & replace each 4 bit group by the equivalent hexadecimal digit.

Example:

$1010111011_2 = ?_{16}$

$$\underbrace{10}_{2}\ \underbrace{1011}_{B}\ \underbrace{1011}_{B}$$

$1010111011_2 = 2BB_{16}$

Decimal to Binary conversion:

Technique

- Divide by two, keep track of the remainder
- First remainder is bit 0 (LSB, least-significant bit)
- Second remainder is bit 1 etc

$125_{10} = ?_2$

2	125	
2	62	1
2	31	0
2	15	1
2	7	1
2	3	1
2	1	1
	0	1

$125_{10} = 1111101_2$

Decimal to Octal Conversion:

To convert a mixed decimal no. To a mixed octal no. convert the integer and fraction parts separately.

To convert decimal integer no. to octal, successively divide the given no by 8 till the quotient is 0. The last remainder is the MSD .The remainder read upwards give the equivalent octal integer no.

Toconvert the given decimal fraction to octal, successively multiply the decimal fraction & the subsequent decimal fractions by 8 till the product is 0 or till the required accuracy is the MSD. The integers to the left of the octal pt read downwards give the octal fraction.

NUMBER BASE CONVERSION

Example:

$$1234_{10} = ?_8$$

$$\begin{array}{r|r r}
8 & 1234 & \\ \hline
8 & 154 & 2 \\ \hline
8 & 19 & 2 \\ \hline
8 & 2 & 3 \\ \hline
 & 0 & 2
\end{array}$$

$$1234_{10} = 2322_8$$

Decimal to Hexadecimal conversion:

It is successively divide the given decimal no. by 16 till the quotient is zero. The last remainder is the MSB. The remainder read from bottom to top gives the equivalent hexadecimal integer.

Toconvert a decimal fraction to hexadecimal successively multiply the given decimal fraction & subsequent decimal fractions by 16, till the product is zero. Or till the required accuracy is obtained and collect all the integers to the left of decimal pt. The first integer is MSB & the integer read from top to bottom give the hexadecimal fraction known as **the hexadabble method.**

NUMBER BASE CONVERSION

Example:

$$1234_{10} = ?_{16}$$

$$\begin{array}{r|rl}
16 & 1234 & \\ \hline
16 & 77 & 2 \\ \hline
16 & 4 & 13 = D \\ \hline
 & 0 & 4
\end{array}$$

$$1234_{10} = 4D2_{16}$$

Octal to binary Conversion:

Convert each octal digit to a 3-bit equivalent binary representation

$705_8 = ?_2$

$$\begin{array}{ccc} 7 & 0 & 5 \\ \downarrow & \downarrow & \downarrow \\ 111 & 000 & 101 \end{array}$$

$705_8 = 111000101_2$

Octal to decimal Conversion:

Multiply each digit in the octal no by the weight of its position & add all the product terms Decimal value of the octal no.

$$
\begin{array}{lrcl}
724_8 \Rightarrow & 4 \times 8^0 & = & 4 \\
 & 2 \times 8^1 & = & 16 \\
 & 7 \times 8^2 & = & 448 \\
\hline
 & & & 468_{10}
\end{array}
$$

NUMBER BASE CONVERSION

Octal to hexadecimal conversion:

The simplest way is to first convert the given octal no. to binary & then the binary no. to hexadecimal.

$1076_8 = ?_{16}$

1 0 7 6

001 0|00 11|1 110

2 3 E

$1076_8 = 23E_{16}$

NUMBER BASE CONVERSION

Octal to hexadecimal conversion:

The simplest way is to first convert the given octal no. to binary & then the binary no. to hexadecimal.

$1076_8 = ?_{16}$

$$\begin{array}{cccc} 1 & 0 & 7 & 6 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 001 & 0|00 & 11|1 & 110 \end{array}$$

$$\begin{array}{ccc} 2 & 3 & E \end{array}$$

$1076_8 = 23E_{16}$

Hexa decimal to binary Conversion:

Convert each hexadecimal digit to a 4-bit equivalent binary representation

$10AF_{16} = ?_2$

1	0	A	F
↓	↓	↓	↓
0001	0000	1010	1111

$10AF_{16} = 0001000010101111_2$

Hexa decimal to decimal Conversion:

Convert each hexadecimal digit to a 4-bit equivalent binary representation

$$
\begin{array}{llllllll}
ABC_{16} \Rightarrow & C \times 16^0 & = & 12 & \times & 1 & = & 12 \\
 & B \times 16^1 & = & 11 & \times & 16 & = & 176 \\
 & A \times 16^2 & = & 10 & \times & 256 & = & \underline{2560} \\
 & & & & & & & 2748_{10}
\end{array}
$$

Binary Addition:

Rules:

0+0=0

0+1=1

1+0=1

1+1=10

i.e, 0 with a carry of 1.

Binary Subtraction:

Rules:

0-0=0

1-1=0

1-0=1

0-1=1

with a borrow of 1

COMPLEMENTS

9's & 10's Complements:

It is the Subtraction of decimal number can be accomplished by the 9's & 10's compliment methods similar to the 1's & 2's compliment methods of binary . The 9's compliment of a decimal number is obtained by subtracting each digit of that decimal number from 9. The 10's compliment of a decimal number is obtained by adding a 1 to its 9'scompliment.

BCD Addition:

It is individually adding the corresponding digits of the decimal numbers expressed in 4 bit binary groups starting from the LSD .

If there is no carry & the sum term is not an illegal code , no correction is needed.

If there is a carry out of one group to the next group or if the sum term is an illegal code then 6_{10} (0110) is added to the sum term of that group & the resulting carry is added to the next group.

BCD Subtraction:

Performed by subtracting the digits of each 4 bit group of the subtrahend the digits from the corresponding 4- bit group of the minuend in binary starting from the LSD . if there is no borrow from the next group , then 6_{10} (0110) is subtracted from the difference term of this group.

ERROR – DETECTING CODES

Some Common Error Detecting and Correcting Codes

- Parity Code
- Hamming Code

Parity Code:

- A parity bit is an extra bit added to a string of data bits in order to detect any error that might have crept into it while it was being stored or processed and moved from one place to another in a digital system.
- This simple parity code suffers from two limitations. Firstly, it cannot detect the error if the number of bits having undergone a change is even.

ERROR – DETECTING CODES

The parity bit can be set to 0 and 1 depending on the type of the parity required.

- For even parity, this bit is set to 1 or 0 such that the no. of "1 bits" in the entire word is even. Shown in fig. (a).
- For odd parity, this bit is set to 1 or 0 such that the no. of "1 bits" In the entire word is odd. Shown in fig. (b).

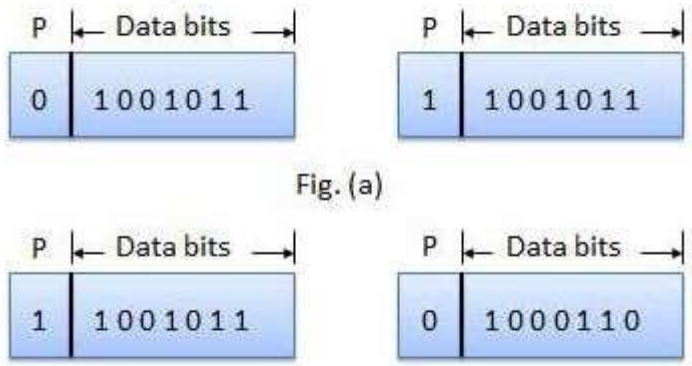


fig. (a)

fig. (b)

Hamming Code:

- An increase in the number of redundant bits added to message bits can enhance the capability of the code to detect and correct errors.
- If sufficient number of redundant bits arranged such that different error bits produce different error results, then it should be possible not only to detect the error bit but also to identify its location.
- In fact, the addition of redundant bits alters the ‘distance’ code parameter, which has come to be known as the Hamming distance.

- The code word sequence for this code is written as $P_1P_2D_1P_3D_2D_3D_4$, with P_1, P_2 and P_3 being the parity bits and D_1, D_2, D_3 and D_4 being the data bits.

- **Generation of Hamming Code:**

	P_1	P_2	D_1	P_3	D_2	D_3	D_4
Data bits (without parity)			0		1	1	0
Data bits with parity bit P_1	1		0		1		0
Data bits with parity bit P_2		1	0			1	0
Data bits with parity bit P_3				0	1	1	0
Data bits with parity	1	1	0	0	1	1	0

BOOLEAN ALGEBRA

- Identity Elements
  - a: $X+0=X$
  - b: $X \bullet 1=X$
- Commutativity
  - a: $X+Y=Y+X$
  - b: $X \bullet Y=Y \bullet X$
- Complements
  - a: $X+X'=1$
  - b: $X \bullet X'=0$

OR operation

X	Y	X+0	X+Y	Y+X	X’	X+X’
0	0	0	0	0	1	1
0	1	0	1	1	1	1
1	0	1	1	1	0	1
1	1	1	1	1	0	1

AND operation

X	Y	X•1	X•Y	Y•X	X’	X•X’
0	0	0	0	0	1	0
0	1	0	0	0	1	0
1	0	1	0	0	0	0
1	1	1	1	1	0	0

- Associativity
  - a: (X+Y)+Z=X+(Y+Z)
  - b: (X•Y)•Z=X•(Y•Z)

X	Y	Z	X+Y	(X+Y)+Z	Y+Z	X+(Y+Z)	X•Y	(X•Y)•Z	Y•Z	X•(Y•Z)
0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	1	1	0	0	0	0
0	1	0	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	0	0	1	0
1	0	0	1	1	0	1	0	0	0	0
1	0	1	1	1	1	1	0	0	0	0
1	1	0	1	1	1	1	1	0	0	0
1	1	1	1	1	1	1	1	1	1	1

- Distributivity
  - a: X+(Y•Z) = (X+Y)•(X+Z)
  - b: X•(Y+Z) = (X•Y)+(X•Z)

X	Y	Z	X+Y	X+Z	(X+Y)•(X+Z)	Y•Z	X+(Y•Z)	X•Y	X•Z	X•Y+X•Z	Y+Z	X•(Y+Z)
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0	1	0
0	1	0	1	0	0	0	0	0	0	0	1	0
0	1	1	1	1	1	1	1	0	0	0	1	0
1	0	0	1	1	1	0	1	0	0	0	0	0
1	0	1	1	1	1	0	1	0	1	1	1	1
1	1	0	1	1	1	0	1	1	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1

BOOLEAN ALGEBRA

- Idempotency

  a: $X+X=X$

  b: $X \bullet X=X$

- Null elements

  a: $X+1=1$

  b: $X \bullet 0=0$

- Involution

  a: $(X')'=X$

X	Y	X+Y (OR)	X•Y (AND)	X+X	X•X	X+1	X•0	X'	X''
0	0	0	0	0	0	1	0	1	0
0	1	1	0	0	0	1	0	1	0
1	0	1	0	1	1	1	0	0	1
1	1	1	1	1	1	1	0	0	1

- Absorption
  - a: (X•Y)+(X•Y’•Z)=(X•Y)+(X•Z)
  - b: (X+Y)•(X+Y’+Z) = (X+Y)•(X+Z)

X Y Z	Y’	XY	XY’Z	(XY)+ (XY’Z)	XZ	(XY)+ (XZ)	X+Y	X+Y’ +Z	(X+Y)• (X+Y’+Z)	X+Z	(X+Y)• (X+Z)
0 0 0	1	0	0	0	0	0	0	1	0	0	0
0 0 1	1	0	0	0	0	0	0	1	0	1	0
0 1 0	0	0	0	0	0	0	1	0	0	0	0
0 1 1	0	0	0	0	0	0	1	1	1	1	1
1 0 0	1	0	0	0	0	0	1	1	1	1	1
1 0 1	1	0	1	1	1	1	1	1	1	1	1
1 1 0	0	1	0	1	0	1	1	1	1	1	1
1 1 1	0	1	0	1	1	1	1	1	1	1	1

BOOLEAN ALGEBRA

- DeMorgan's theorem (very important!)

  a: (X+Y)'= X'•Y'

  $\overline{X+Y} = \overline{X}\bullet\overline{Y}$ break (or connect) the bar & change the sign

  b: (X.Y)'=X'+Y'

  $\overline{X\bullet Y} = \overline{X}+\overline{Y}$ break (or connect) the bar & change the sign

Generalized DeMorgan's theorem:

- GT8a: $(X_1+X_2+\ldots+X_{n-1}+X_n)' = X_1'\bullet X_2'\bullet\ldots\bullet X_{n-1}'\bullet X_n'$
- GT8b: $(X_1\bullet X_2\bullet\ldots\bullet X_{n-1}\bullet X_n)' = X_1'+X_2'+\ldots+X_{n-1}'+X_n'$

OR AND

X	Y	X+Y	X•Y	X'	Y'	(X+Y)'	X'•Y'	(X•Y)'	X'+Y'
0	0	0	0	1	1	1	1	1	1
0	1	1	0	1	0	0	0	1	1
1	0	1	0	0	1	0	0	1	1
1	1	1	1	0	0	0	0	0	0

BOOLEAN ALGEBRA

- Consensus Theorem
  - a: (X•Y)+(X’•Z)+(Y•Z) = (X•Y)+(X’•Z)
  - b: (X+Y)•(X’+Z)•(Y+Z) = (X+Y)•(X’+Z)

X	Y	Z	X’	XY	X’Z	YZ	(XY)+(X’Z)+(YZ)	(XY)+(X’Z)	X+Y	X’+Z	Y+Z	(X+Y)•(X’+Z)•(Y+Z)	(X+Y)•(X’+Z)
0	0	0	1	0	0	0	0	0	0	1	0	0	0
0	0	1	1	0	1	0	1	1	0	1	1	0	0
0	1	0	1	0	0	0	0	0	1	1	1	1	1
0	1	1	1	0	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0	0	1	1	1	1	1
1	1	0	0	1	0	0	1	1	1	0	1	0	0
1	1	1	0	1	0	1	1	1	1	1	1	1	1

SWITCHING FUNCTIONs

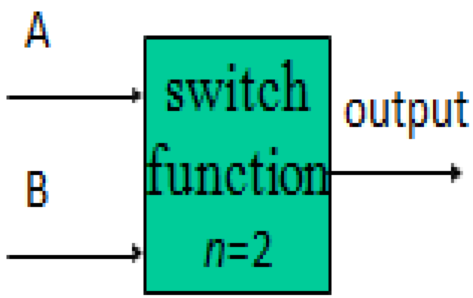
- For n variables, there are 2^n possible combinations of Values from all 0s to all 1s
- There are 2 possible values for the output of a function of a combination of values of n variables i.e. 0 and 1
- There are 2^{2^n} different switching functions for n variables
- n=0 (no inputs) $2^{2^n} = 2^{2^0} = 2^1 = 2$

  Output can be either 0 or 1
- n=1 (1 input, A) $2^{2^n} = 2^{2^1} = 2^2 = 4$

  Output can be 0, 1, A, or A'

SWITCHING FUNCTIONS EXAMPLE

- n=2 (2 inputs, A and B) $\Rightarrow$ $2^{2^n} = 2^{2^2} = 2^4 = 16$



AB	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	0 0
0 0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	
0 1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	
1 0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	
1 1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	

$f_0 = 0$

$f_1 = A'B' = (A+B)'$

$f_2 = A'B$

$f_3 = A'B' + A'B = A'(B'+B) = A'$

CANONICAL AND STANDERED FORMS

Logical functions are generally expressed in terms of different combinations of logical variables with their true forms as well as the complement forms. Binary logic values obtained by the logical functions and logic variables are in binary form. An arbitrary logic function can be expressed in the following forms.

- Sum of the Products (SOP)
- Product of the Sums (POS)

CANONICAL AND STANDERED FORMS

- **Product Term:** In Boolean algebra, the logical product of several variables on which a function depends is considered to be a product term. In other words, the AND function is referred to as a product term or standard product.

- **Sum Term:** An OR function is referred to as a sum term
- **Sum of Products (SOP):** The logical sum of two or more logical productterms is referred to as a sum of products expression

$$Y = AB + BC + AC$$

- **Product of Sums (POS):** Similarly, the logical product of two or morelogical sum terms is called a product of sums expression

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

CANONICAL AND STANDERED FORMS

- **Standard form:** The standard form of the Boolean function is when it is expressed in sum of the products or product of the sums fashion

$$Y = AB + BC + AC$$

- **Nonstandard Form:** Boolean functions are also sometimes expressed in nonstandard forms like $F = (AB + CD)(\bar{A}\bar{B} + \bar{C}\bar{D})$, which is neither a sum of products form nor a product of sums form.

- **Minterm**: A product term containing all n variables of the function in either true or complemented form is called the minterm. Each minterm is obtained by an AND operation of the variables in their true form or complemented form.

- **Maxterm:** A sum term containing all n variables of the function in either true or complemented form is called the Maxterm. Each Maxterm is obtained by an OR operation of the variables in their true form or complemented form.
- The canonical sum of products form of a logic function can be obtained by using the following procedure:
- Check each term in the given logic function. Retain if it is a minterm, continue to examine the next term in the same manner.
- Examine for the variables that are missing in each product which is not a minterm. If the missing variable in the minterm is X, multiply that minterm with (X+X').
- Multiply all the products and discard the redundant terms.

- **Standard form:** The standard form of the Boolean function is when it is expressed in sum of the products or product of the sums fashion

$$Y = AB + BC + AC$$

- **Nonstandard Form:** Boolean functions are also sometimes expressed in nonstandard forms like $F = (AB + CD)(\bar{A}\bar{B} + \bar{C}\bar{D})$, which is neither a sum of products form nor a product of sums form.

- **Minterm**: A product term containing all n variables of the function in either true or complemented form is called the minterm. Each minterm is obtained by an AND operation of the variables in their true form or complemented form.

CANONICAL AND STANDERED FORMS

- **Example:** Obtain the canonical sum of product form of the following function $F\,(A,B,C) = A + BC$

- Solution: $F\,(A,B,C) = A + BC$

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

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

$$= ABC + A\bar{B}C + ABC̄ + A\bar{B}\bar{C} + ABC + \bar{A}BC$$

$$= ABC + A\bar{B}C + AB\bar{C} + A\bar{B}\bar{C} + \bar{A}BC\ (as\ ABC + ABC = ABC)$$

- Hence the canonical sum of the product expression of the given function is

$$F\,(A,B,C) = ABC + A\bar{B}C + AB\bar{C} + A\bar{B}\bar{C} + \bar{A}BC$$

CANONICAL AND STANDERED FORMS

- The product of sums form is a method (or form) of simplifying the Boolean expressions of logic gates. In this POS form, all the variables are ORed, i.e. written as sums to form sum terms. All these sum terms are ANDed (multiplied) together to get the product-of-sum form. This form is exactly opposite to the SOP form. So this can also be said as ―Dualof SOPform‖.

(A+B) * (A + B + C) * (C +D)

POS form can be obtained by

- Writing an OR term for each input combination, which produces LOW output.
- Writing the input variables if the value is 0, and write the complement of the variable if its value is AND the OR terms to obtain the outputfunction.

ALGEBRAIC SIMPLIFICATION

Minimize the following Boolean expression using Boolean identities –

F(A,B,C)=(A+B)(A+C)F(A,B,C)=(A+B)(A+C)

Solution

Given, F(A,B,C)=(A+B)(A+C)

F (A,B,C)=A.A+A.C+B.A+B.C [Applying distributive Rule]

F(A,B,C)=A+A.C+B.A+B.C [Applying Idempotent Law]

F(A,B,C)=A(1+C)+B.A+B.C [Applying distributive Law]

F(A,B,C)=A+B.A+B.C [Applying dominance

F(A,B,C)=(A+1).A+B.C [Applying distributive Law]

F(A,B,C)=1.A+B.C [Applying dominance Law]

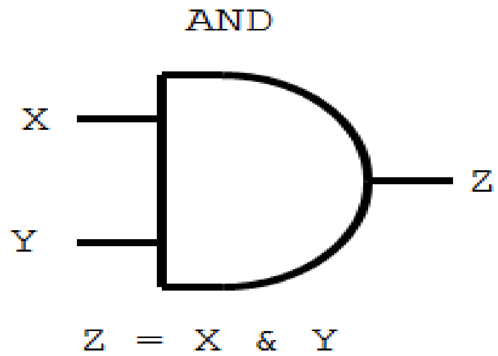
F(A,B,C)=A+B.C [Applying dominance Law]

So, F(A,B,C)=A+BCF(A,B,C)=A+BC is the minimized form.

DIGITAL LOGIC GATES

AND GATE:

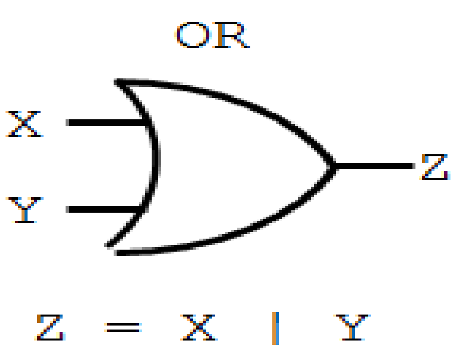
Z=A.B



X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

OR GATE:

Z=A+B

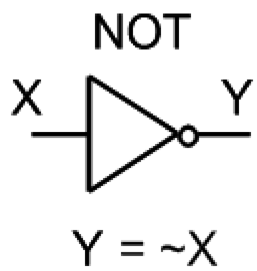


X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

DIGITAL LOGIC GATES

NOT GATE:

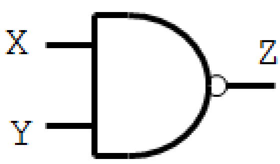
Z=A’



X	Y
0	1
1	0

NAND GATE:

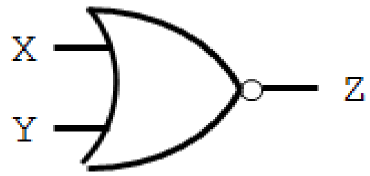
$Z=\overline{A.B}$



X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

NOR GATE:

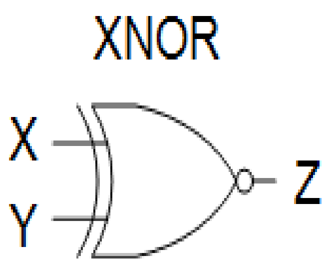
$Z=\overline{A+B}$



X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

Ex-OR GATE:

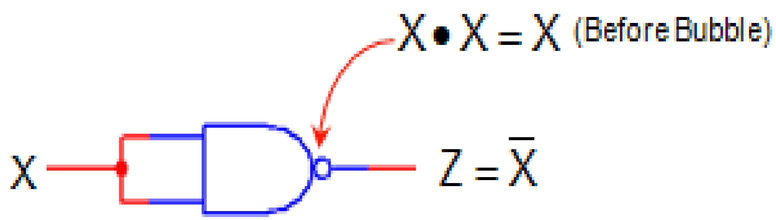
$Z=A \oplus B$



X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1

MULTILEVEL NAND-NOR REALIZATION

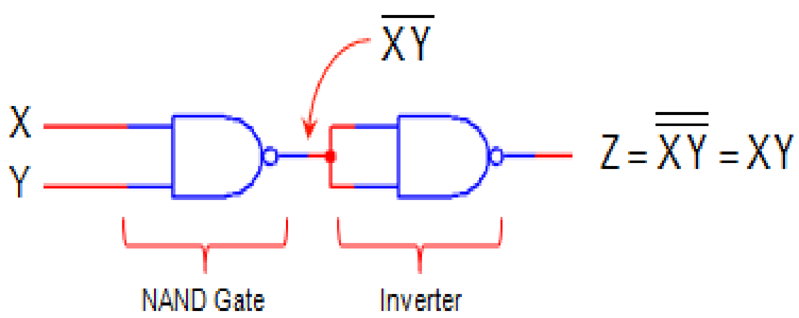
NAND Gate as an Inverter Gate



X	Z
0	1
1	0

Equivalent to Inverter

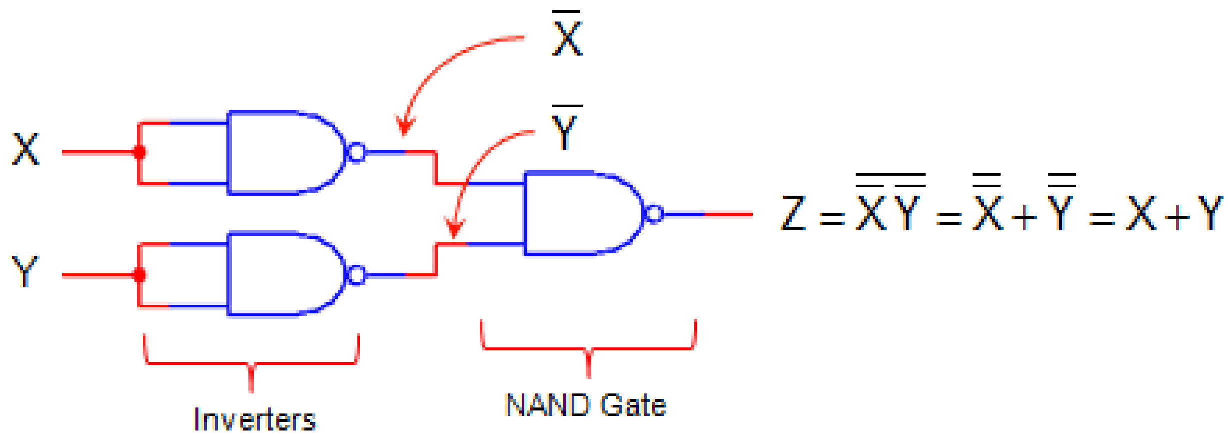
NAND Gate as an AND Gate



X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

Equivalent to AND Gate

NAND Gate as an OR Gate

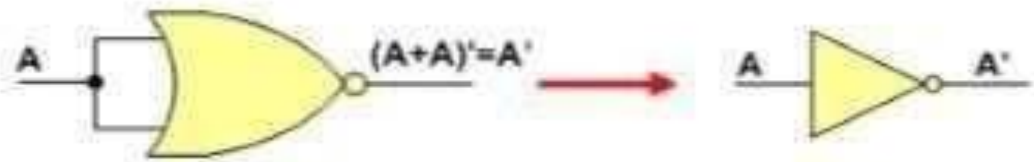


X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

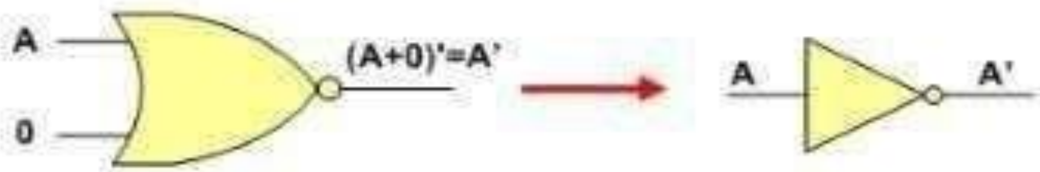
Equivalent to OR Gate

MULTILEVEL NAND-NOR REALIZATION

1. All NOR input pins connect to the input signal **A** gives an output **A'**.

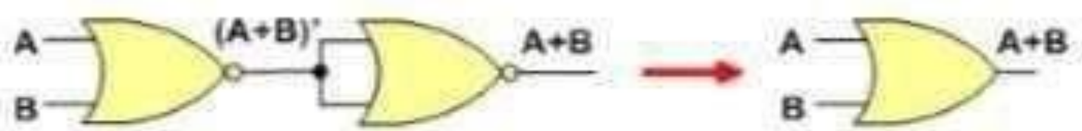


2. One NOR input pin is connected to the input signal **A** while all other input pins are connected to logic **0**. The output will be **A'**.



Implementing OR Using only NOR Gates

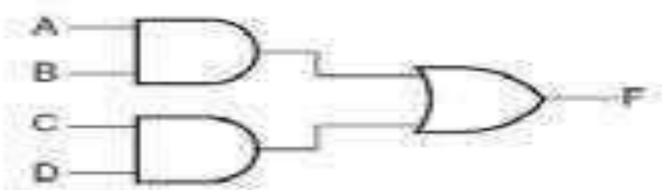
An OR gate can be replaced by NOR gates as shown in the figure (The OR is replaced by a NOR gate with its output complemented by a NOR gate inverter)



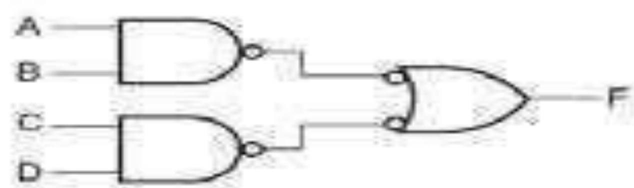
Example1: implement the following function *F = AB +CD*

- The implementation of Boolean functions with NAND gatesrequires that the functions be in sum of products (SOP)form.

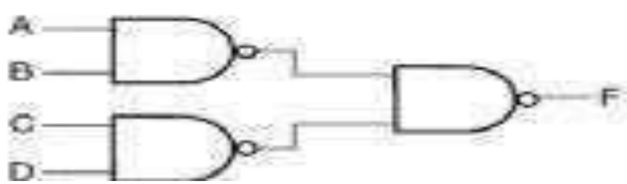
- This function can be implemented by three steps.



(a)

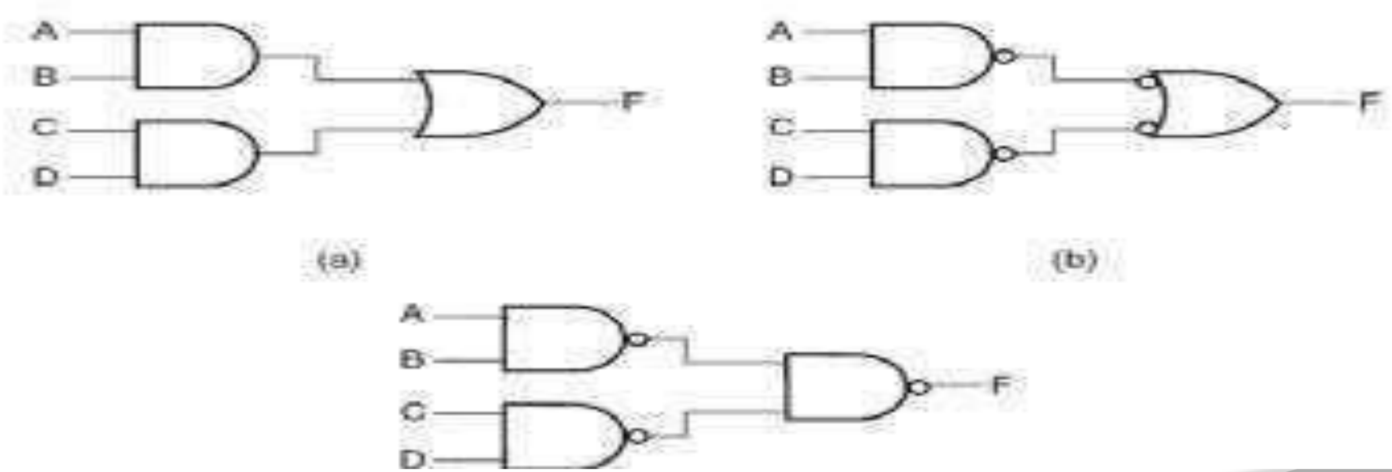


(b)



Example1: implement the following function $F = AB + CD$

- The implementation of Boolean functions with NAND gatesrequires that the functions be in sum of products (SOP)form.
- This function can be implemented by three steps.



MODULE-IV

MINIMIZATION OF BOOLEAN FUNCTIONS

Karnaugh Map Method - Up to five Variables, Don't Care Map Entries, Tabular Method,
Combinational Logic Circuits: Adders, Subtractors, comparators, Multiplexers, Demultiplexers, Encoders, Decoders and Code converters, Hazards and Hazard Free Relations.

CONTENTS

- Up to five Variables,
- Don't Care Map Entries,
- Tabular Method,
- Adders,
- Subtractors,
- comparators, Multiplexers, Demultiplexers,
- Encoders, Decoders and Code converters,
- Hazards and Hazard Free Relations.

- A two-variable function has four possible minterms. We can re- arrange these minterms into a Karnaughmap

x	y	minterm
0	0	x'y'
0	1	x'y
1	0	xy'
1	1	xy

		y: 0	y: 1
x	0	x'y'	x'y
	1	xy'	xy

- Now we can easily see which minterms containcommonliterals
  - Minterms on the left and right sides contain y’ and y respectively
  - Minterms in the top and bottom rows contain x’ and x respectively

		y: 0	y: 1
x	0	x'y'	x'y
	1	xy'	xy

	y'	y
x'	x'y'	x'y
x	xy'	xy

KARANAUGH MAP

- Make as few rectangles as possible, to minimize the number of products in the final expression.
- Make each rectangle as large as possible, to minimize the number of literals in each term.
- Rectangles can be overlapped, if that makes them larger
- The most difficult step is grouping together all the 1s in the K-map
  - Make rectangles around groups of one, two, four or eight 1s
  - All of the 1s in the map should be included in at least one rectangle. Do not include any of the 0s
  - Each group corresponds to one product term

			y	
	0	1	0	0
X	0	1	1	1
		z		

- Let's consider simplifying $f(x,y,z) = xy + y'z + xz$
- You should convert the expression into a sum of minterms form,
  - The easiest way to do this is to make a truth table for the function, and then read off the minterms
  - You can either write out the literals or use the minterm shorthand
  - Here is the truth table and sum of minterms for our example:

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$$\begin{aligned} f(x,y,z) &= x'y'z + xy'z + xyz' + xyz \\ &= m_1 + m_5 + m_6 + m_7 \end{aligned}$$

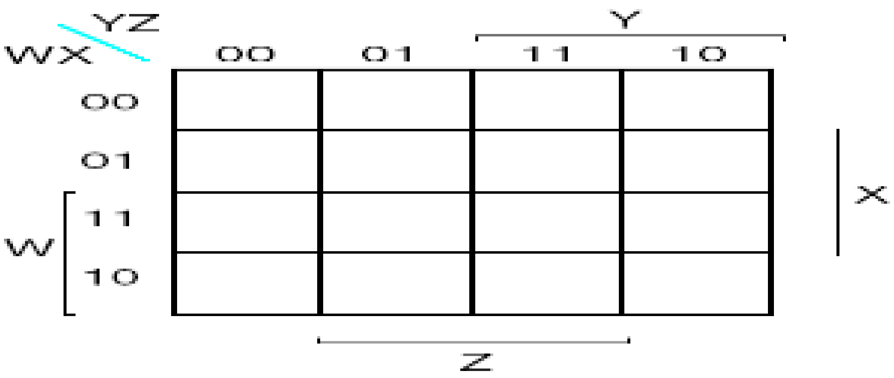
3 VARIABLE K-MAP

- Maxterms are grouped to find minimal PoS expression

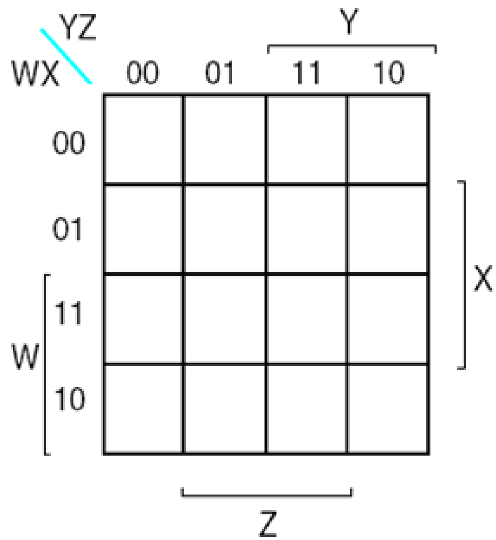
x \ yz	00	01	11	10
0	x +y+z	x+y+z'	x+y'+z'	x+y'+z
1	x' +y+z	x'+y+z'	x'+y'+z'	x'+y'+z

4-VARIABLE K-MAP

- We can do four-variable expressions too!
  - The minterms in the third and fourth columns, *and* in the third and fourth rows, are switched around.
  - Again, this ensures that adjacent squares have common literals
- Grouping minterms is similar to the three-variable case, but:
  - You can have rectangular groups of 1, 2, 4, 8 or 16 minterms
  - You can wrap around all four sides



4-VARIABLE K-MAP



y

w'x'y'z'	w'x'y'z	w'x'yz	w'x'yz'
w'xy'z'	w'xy'z	w'xyz	w'xyz'
wxy'z'	wxy'z	wxyz	wxyz'
wx'y'z'	wx'y'z	wx'yz	wx'yz'

X W Z

y

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

X W Z

DON'T CARE CONDITION

- You don't always need all 2^n input combinations in an n-variable function
  - If you can guarantee that certain input combinations never occur
  - If some outputs aren't used in the rest of the circuit

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	X
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	X
1	1	1	1

- We mark don't-care outputs in truth tables and K-maps with Xs.

DON’T CARE CONDITION

- Find a MSP for

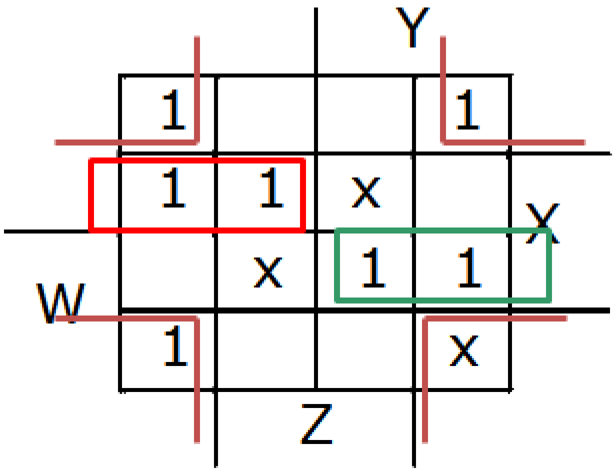
$$f(w,x,y,z) = \Sigma m(0,2,4,5,8,14,15),\ d(w,x,y,z) = \Sigma m(7,10,13)$$

- This notation means that input combinations wxyz = 0111, 1010 and 1101(corresponding to minterms m_7, m_{10} and m_{13}) are unused.

			y		
	1	0	0	1	
	1	1	x	0	x
w	0	x	1	1	
	1	0	0	x	
		z			

- Find a MSP for:

$$f(w,x,y,z) = \Sigma m(0,2,4,5,8,14,15),\ d(w,x,y,z) = \Sigma m(7,10,13)$$



$$f(w,x,y,z) = x'z' + w'xy' + wxy$$

COMBINATIONAL CIRCUITS

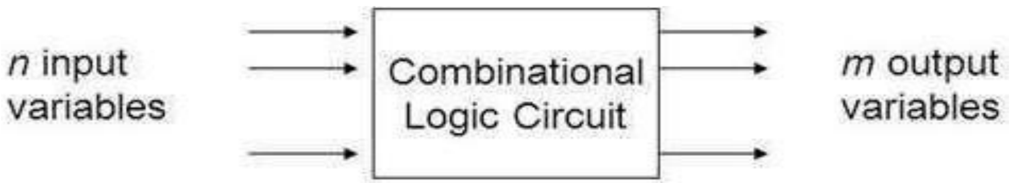
- Combinational circuit is a circuit in which we combine the different gates in the circuit, for example encoder, decoder, multiplexer and demultiplexer.

  Some of the characteristics of combinational circuits arefollowing:

- The output of combinational circuit at any instant of time, depends only on the levels present at inputterminals.
- The combinational circuit do not use any memory. The previousstate of input does not have any effect on the present state of thecircuit.
- A combinational circuit can have an n number of inputs and mnumber of outputs.

- **Block diagram:**

  possible combinations of inputvalues.



- Specific functions : of combinational circuits
- Adders, subtractors , multiplexers, comparators , encoder, Decoder. MSI Circuits and standardcells

Analysis procedure

Toobtain the output Boolean functions from a logic diagram,proceed as follows:

- Label all gate outputs that are a function of input variables with arbitrary symbols. Determine the Boolean functions foreachgate output.
- Label the gates that are a function of input variables and previously labeled gates with other arbitrary symbols. Find the Boolean functions for these gates.
- Repeat the process outlined in step 2 until the outputs of the circuit are obtained.

Design Procedure

- The problem is stated
- The number of available input variables andrequiredoutput variables is determined.
- The input and output variables are assigned letter symbols.
- The truth table that defines the required relationshipbetween inputs and outputs is derived.
- The simplified Boolean function for each output is obtained.
- The logic diagram is drawn.

ADDERS

Half Adder

A Half Adder is a combinational circuit with two binary inputs (augends and addend bits and two binary outputs (sum and carry bits.) It adds the two inputs (A and B) and produces the sum (S) and the carry (C) bits.

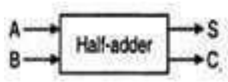


Fig 1:Blockdiagram

Inputs		Outputs	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Fig 2:Truthtable

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

Carry=AB

BINARYSUBTRACTORS

Full subtractor

- The full subtractor perform subtraction of three input bits: the minuend , subtrahend , and borrow in and generates two output bits difference and borrow out.

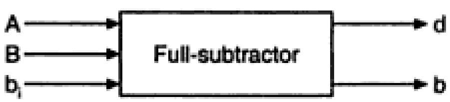


Fig 7:Blockdiagram

Inputs			Difference	Borrow
A	B	b_i	d	b
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

Fig 8: Truthtable

$$d = \bar{A}\bar{B}b_i + \bar{A}B\,\bar{b}_i + A\bar{B}\,\bar{b}_i + ABb_i = A \oplus B \oplus b_i$$

$$b = \bar{A}\bar{B}b_i + \bar{A}B\,\bar{b}_i + \bar{A}Bb_i + ABb_i = \bar{A}B + (\overline{A \oplus B})b_i$$

PARALLEL ADDER AND SUBTRACTOR

- A binary parallel adder is a digital circuit that adds two binary numbers in parallel form and produces the arithmetic sum of those numbers in parallel form

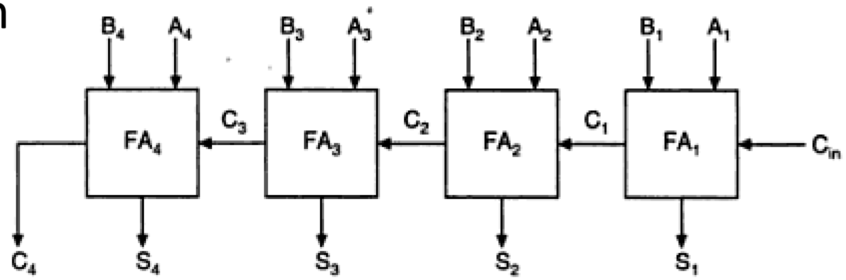


Fig 9:paralleladder

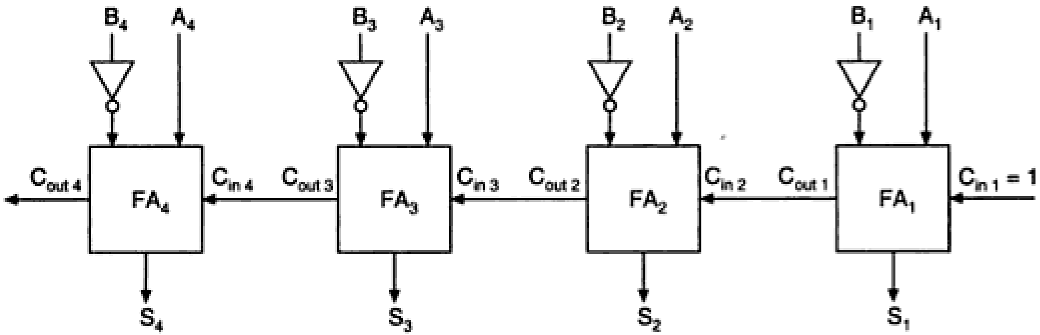


Fig 10:parallelsubtractor

CARRY LOOK-A- HEAD ADDER

- In parallel-adder , the speed with which an addition can be performed is governed by the time required for the carries to propagate or ripple through all of the stages of theadder.
- The look-ahead carry adder speeds up the process byeliminating this ripple carry delay.

$$S_n = P_n \oplus C_n \text{ where } P_n = A_n \oplus B_n$$
$$C_{on} = C_{n+1} = G_n + P_n C_n \text{ where } G_n = A_n \cdot B_n$$

CARRY LOOK-A- HEAD ADDER

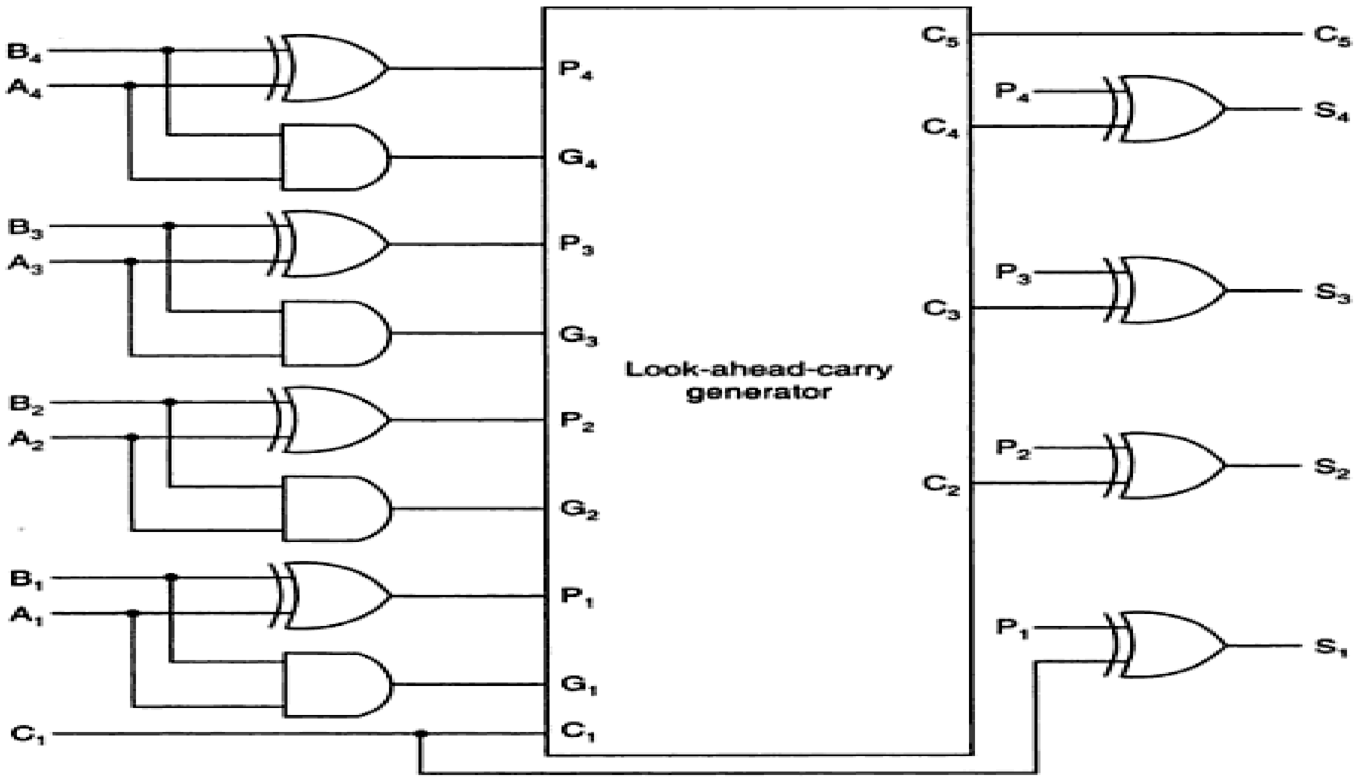


Fig:1 blockdiagram

➢A binary multiplier is an electronic circuit used in digital electronics, such as a computer, to multiply two binary numbers. It is built using binary adders.

Example: (101 x 011)

Partial products are: 101 ›

$$
\begin{array}{cccccc}
 & & & 1 & 0 & 1 \\
 & & \times & 0 & 1 & 1 \\
\hline
 & & & 1 & 0 & 1 \\
 & & 1 & 0 & 1 & \\
 & 0 & 0 & 0 & & \\
\hline
0 & 0 & 1 & 1 & 1 & 1
\end{array}
$$

BINARY MULTIPLIER

- We can also make an $n \times m$ “block” multiplier and use that to form partial products.
- Example: 2 × 2 – The logic equations for each partial-product binary digit are shownbelow
- We need to "add" the columns to get the product bits P0, P1, P2, and P3

		b_1	b_0
		a_1	a_0
		$(a_0 \cdot b_1)$	$(a_0 \cdot b_0)$
+	$(a_1 \cdot b_1)$	$(a_1 \cdot b_0)$	
P_3	P_2	P_1	P_0

BINARY MULTIPLIER

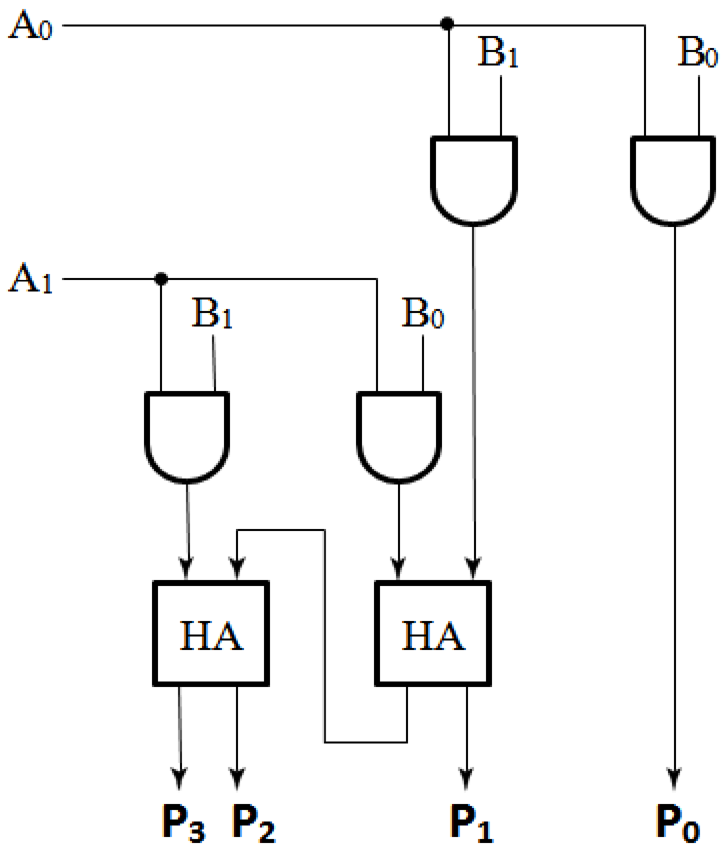


Fig 1: 2 x 2 multiplier array

MAGNITUDE COMPARATOR

- Magnitude comparator takes two numbers as input in binary form and determines whether one number is greater than, less than or equal to the other number.

1-Bit Magnitude Comparator

- A comparator used to compare two bits is called a single bit comparator.

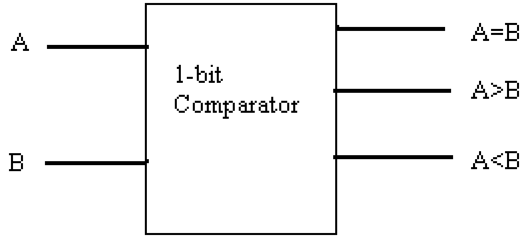


Fig :1 Block diagram

MAGNITUDE COMPARATOR

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

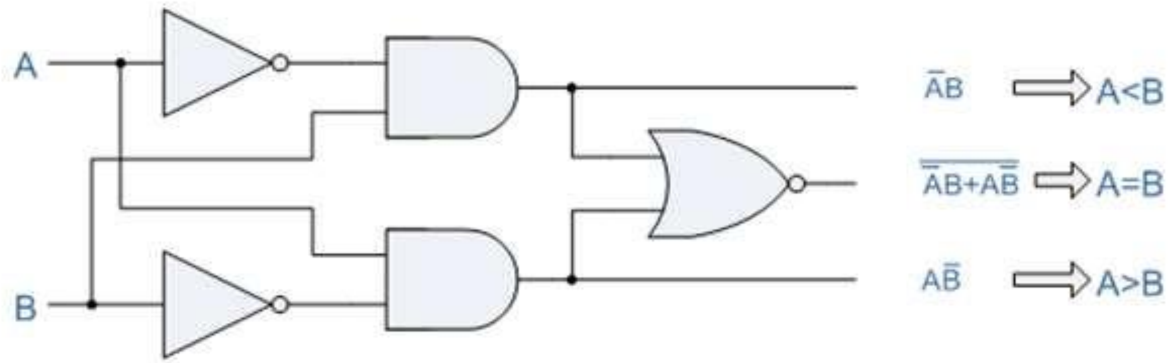


Fig 2:Logic diagram of 1-bit comparator

MAGNITUDE COMPARATOR

- 2 **Bit magnitude comparator**

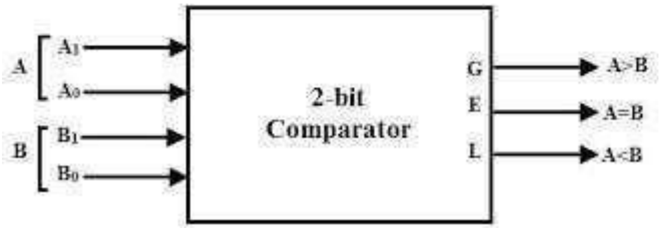


Fig :3 Blockdiagram

Inputs				Outputs		
A_1	A_0	B_1	B_0	A>B	A=B	A<B
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

Fig :4 Truthtable

MAGNITUDE COMPARATOR

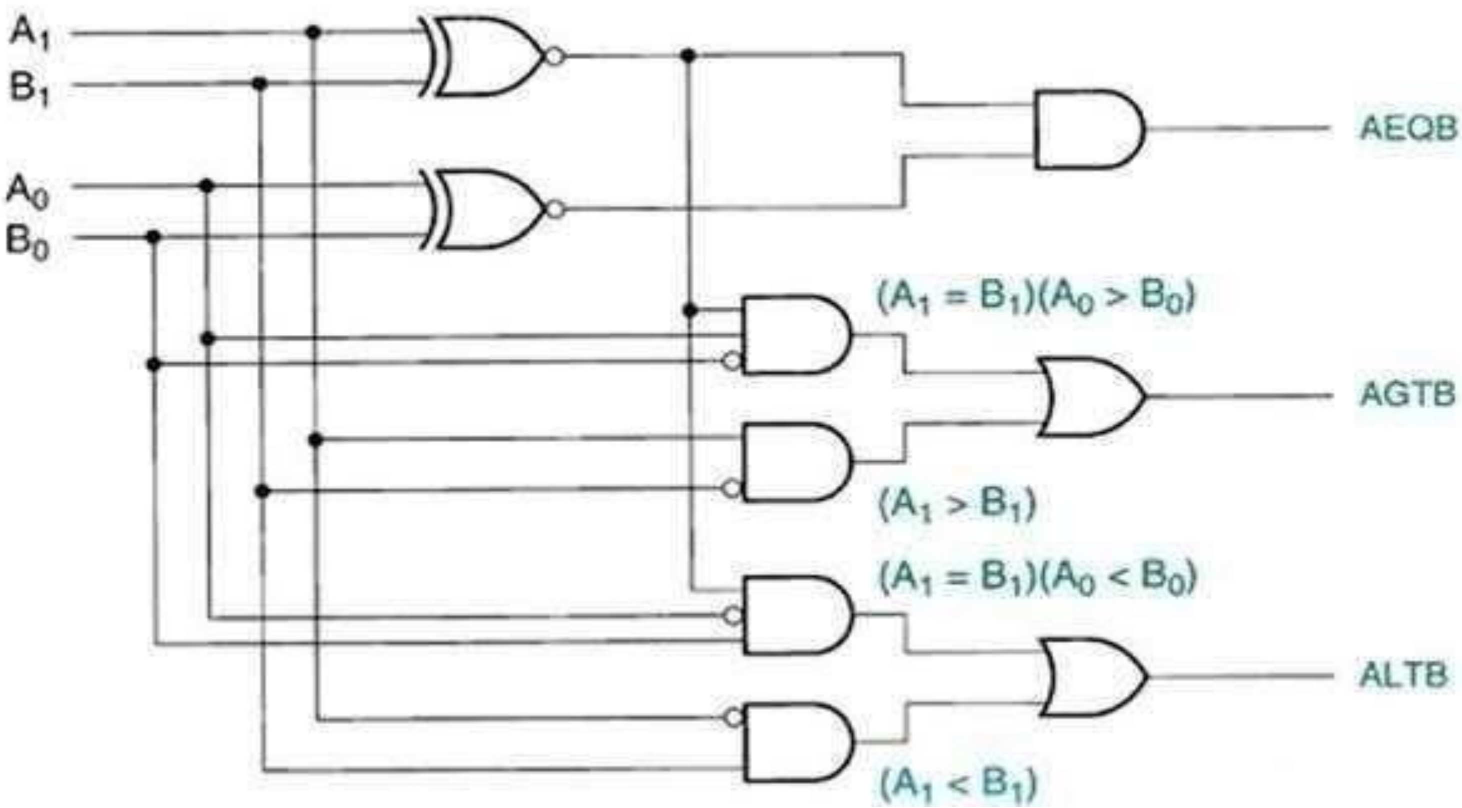
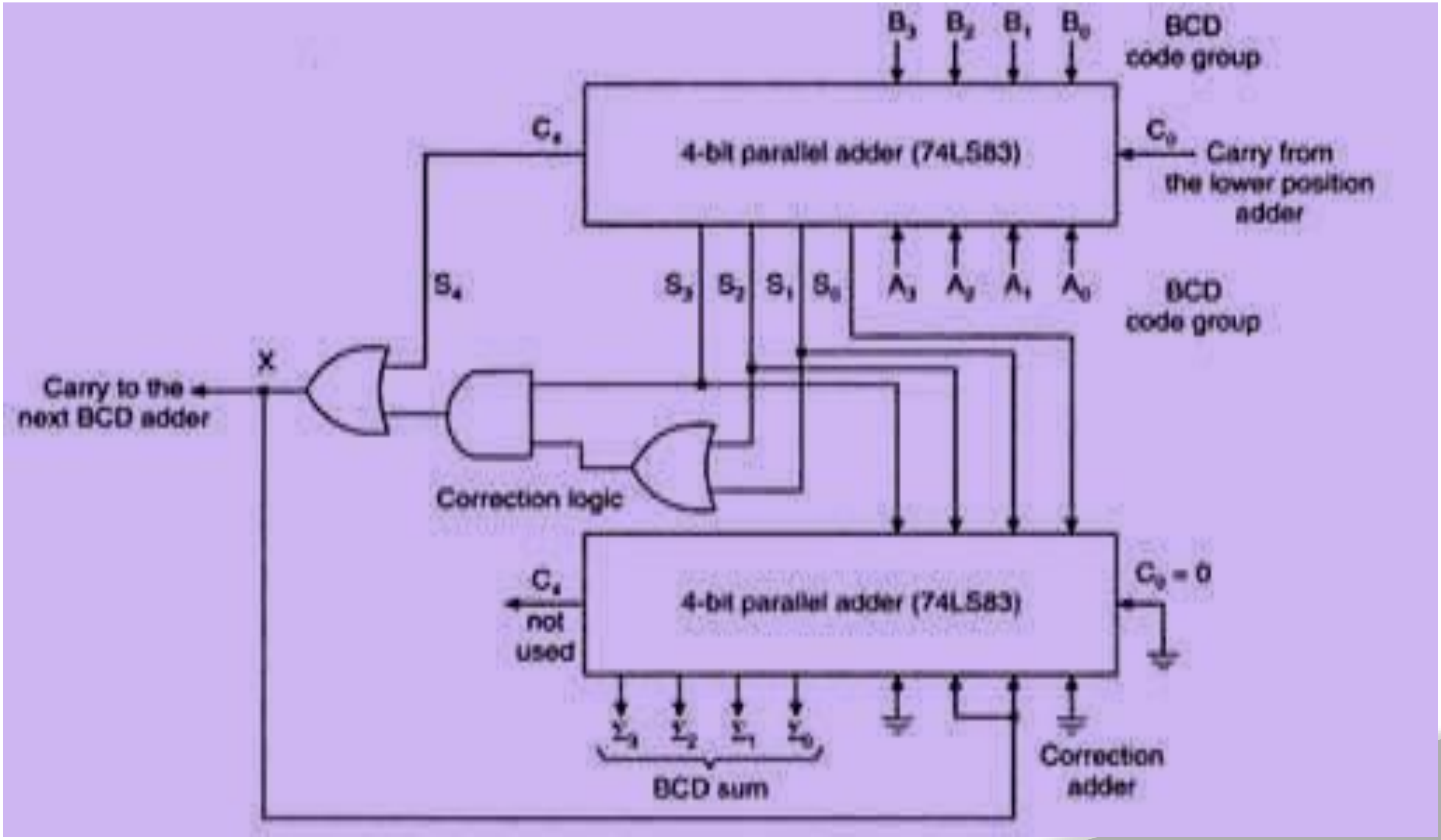


Fig 5:Logic diagram of 2-bitcomparator

BCD Adder

- Perform the addition of two decimal digits in BCD, together with an input carry from a previous stage.
- When the sum is 9 or less, the sum is in proper BCD form and no correction is needed.
- When the sum of two digits is greater than 9, a correction of 0110 should be added to that sum, to produce the proper BCD result. This will produce a carry to be added to the next decimal position.

BCD ADDER



B3 B2 B1 B0
BCD code group
C4
4-bit parallel adder (74LS83)
C0
Carry from the lower position adder
S4
S3 S2 S1 S0
A3 A2 A1 A0
BCD code group
X
Carry to the next BCD adder
Correction logic
C4
not used
4-bit parallel adder (74LS83)
C0 = 0
Σ3 Σ2 Σ1 Σ0
BCD sum
Correction adder

DECODER

- A binary decoder is a combinational logic circuit that converts binary information from the **n** coded inputs to a maximum of 2^n unique outputs.

- We have following types of decoders 2x4,3x8,4x16....

2x4 decoder

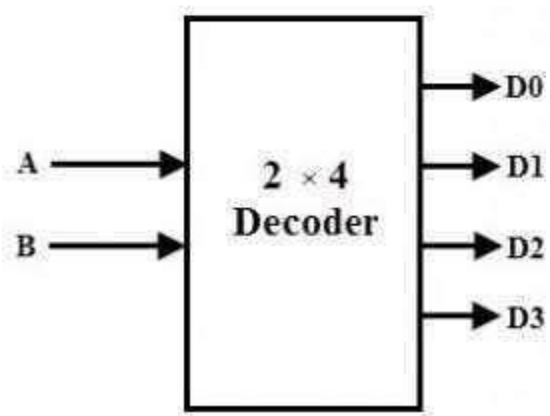


Fig 1: Blockdiagram

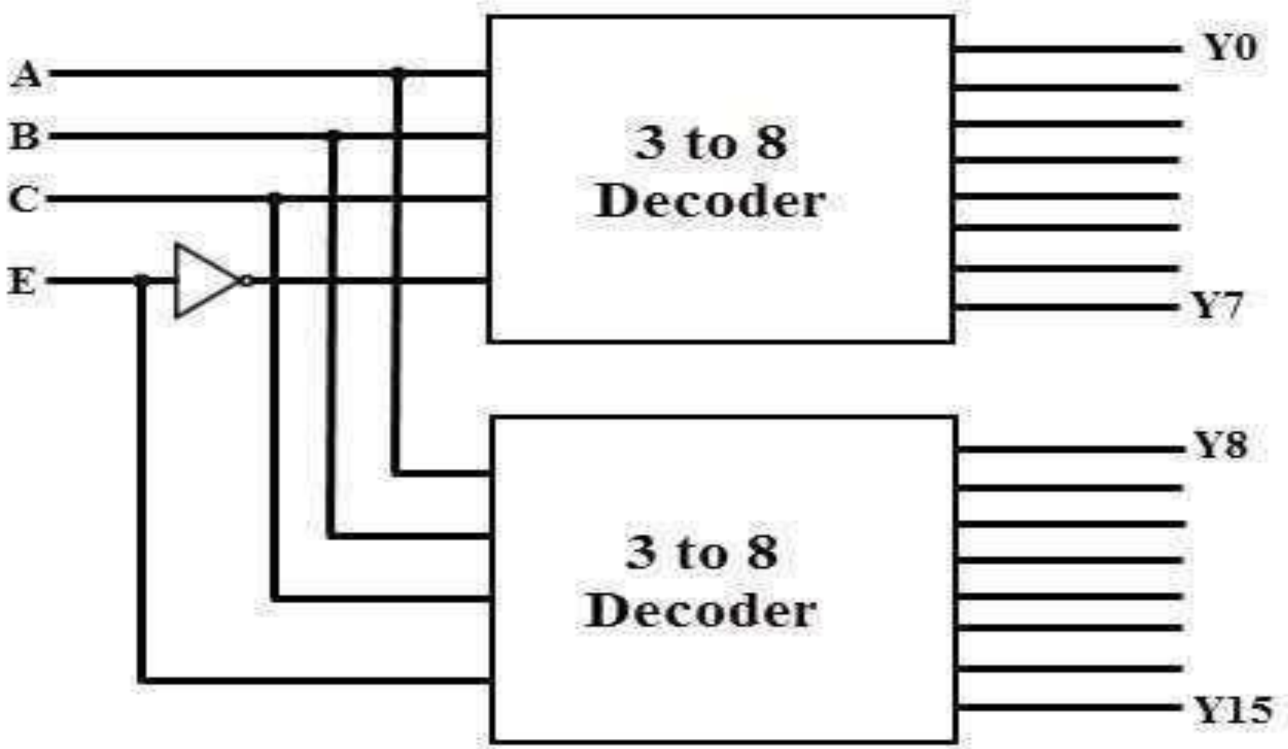
Inputs		Output			
A	B	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
0	1	0	0	1	0
1	1	0	0	0	1

Fig 2:Truthtable

DECODER

Higher order decoder implementation using lower order.

Ex:4x16 decoder using 3x8decoders



ENCODERS

- An Encoder is a combinational circuit that performs the reverse operation of Decoder. It has maximum of 2^n input lines and 'n' output lines.
- It will produce a binary code equivalent to the input, which is active High.

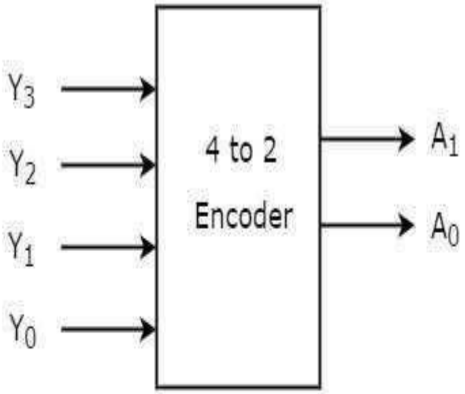


Fig 1:block diagram of 4x2encoder

Octal to binaryencoder

Octal digits		Binary		
		A_2	A_1	A_0
D_0	0	0	0	0
D_1	1	0	0	1
D_2	2	0	1	0
D_3	3	0	1	1
D_4	4	1	0	0
D_5	5	1	0	1
D_6	6	1	1	0
D_7	7	1	1	1

Fig 2:Truth table

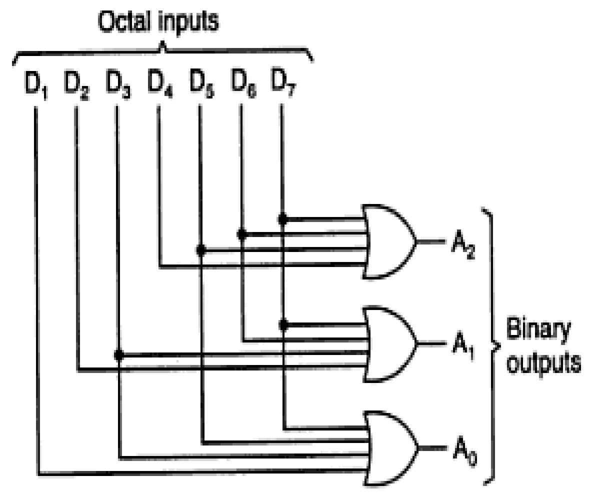


Fig 3: Logic diagram

Priority encoder

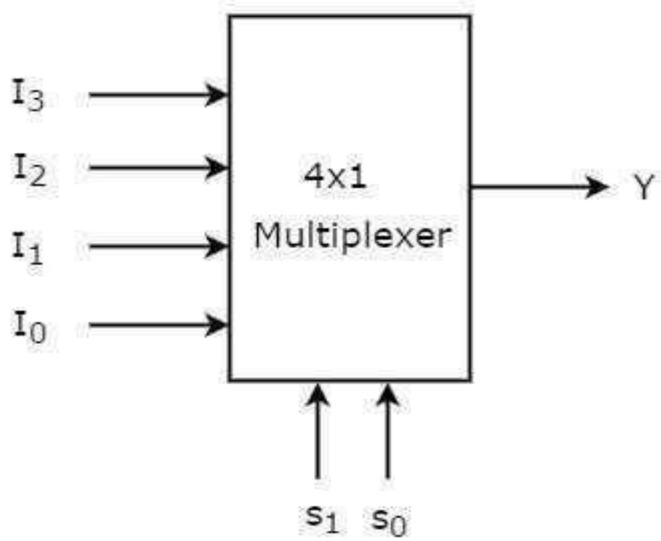
A 4 to 2 priority encoder has four inputs Y_3, Y_2, Y_1 & Y_0 and two outputs A_1 & A_0. Here, the input, Y_3 has the highest priority, whereas the input, Y_0 has the lowest priority.

Inputs				Outputs		
Y_3	Y_2	Y_1	Y_0	A_1	A_0	V
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

Fig 4:Truthtable

MULTIPLEXERS

- Multiplexer is a combinational circuit that has maximum of 2^n data inputs, ‘n’ selection lines and single output line. One of these data inputs will be connected to the output based on the values of selection lines.
- We have different types of multiplexers 2x1,4x1,8x1,16x1,32x1……



MULTIPLEXERS

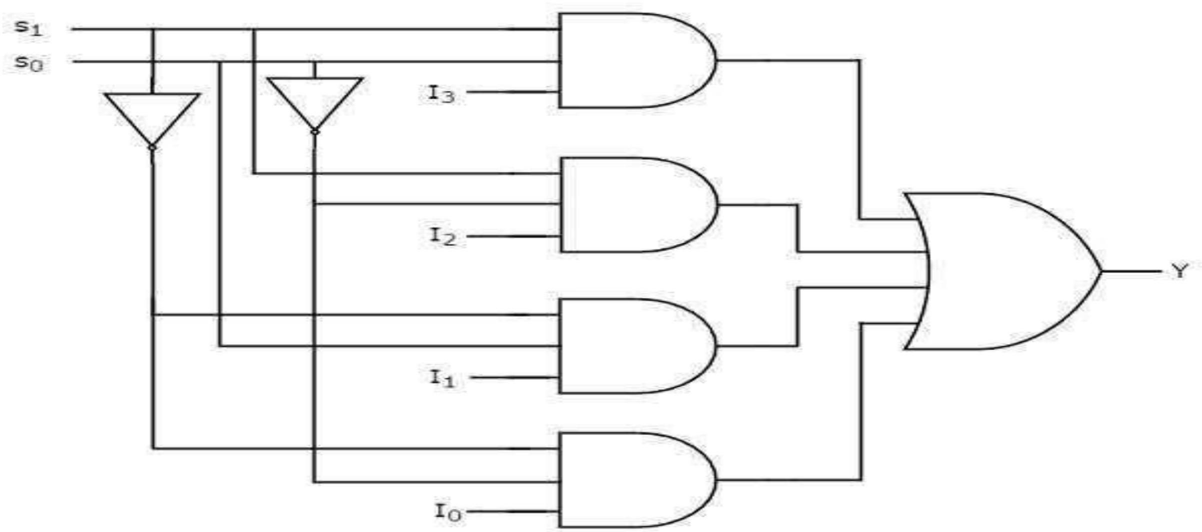


Fig 3: Logic diagram

- Now let us implement the higher-order Multiplexer using lower-order Multiplexers.

- Ex: 8x1 Multiplexer

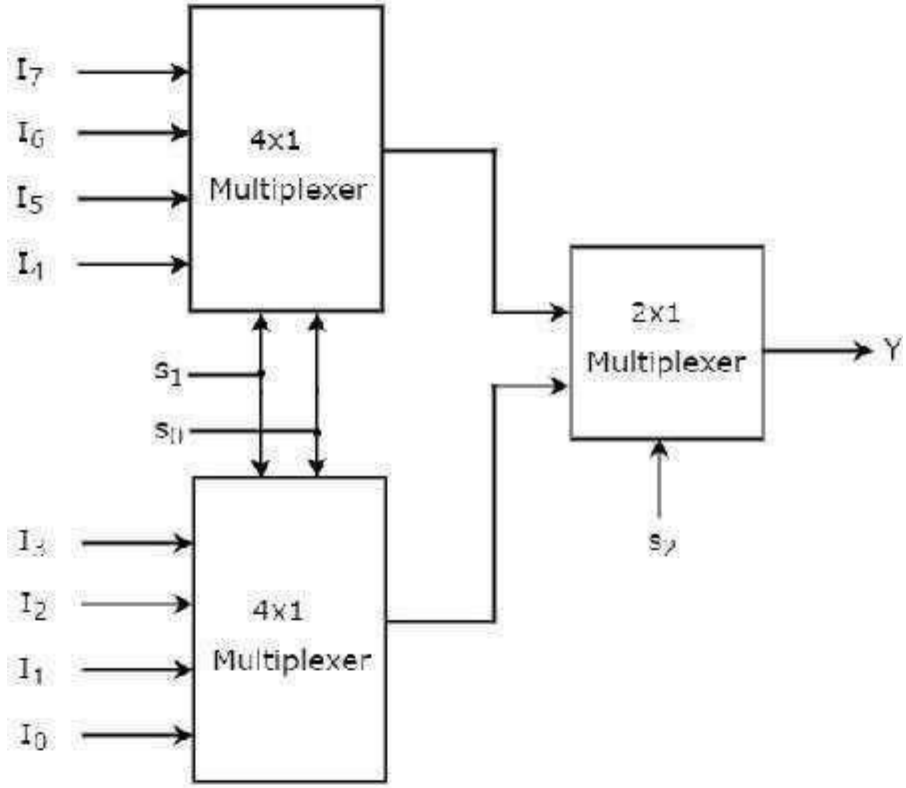
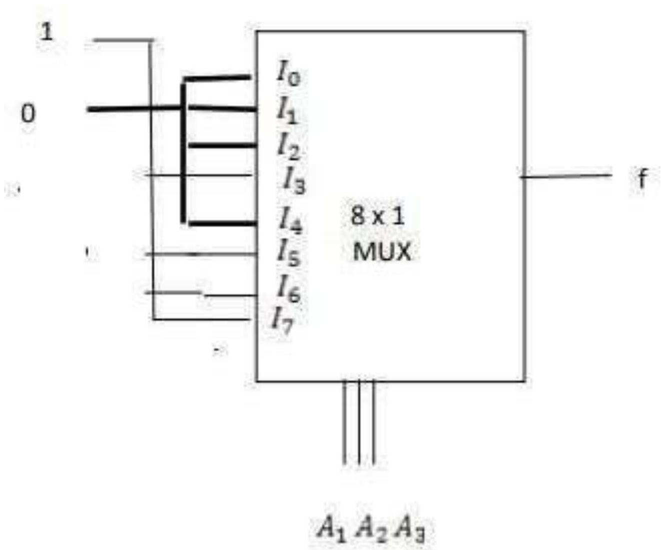


Fig 3: 8x1 Multiplexer diagram

MULTIPLEXERS

- Implementation of Boolean function usingmultiplexer
- f(A1 , A2 , A3) =Σ(3,5,6,7) implementation using 8x1mux



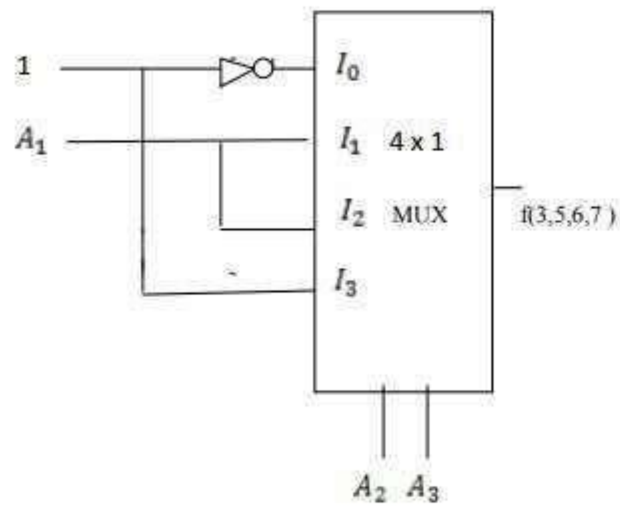
MULTIPLEXERS

f(A1 , A2 , A3) =Σ(3,5,6,7) implementation using 4x1mux

Method:1

Minterms	A_1	A_2	A_3	f
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

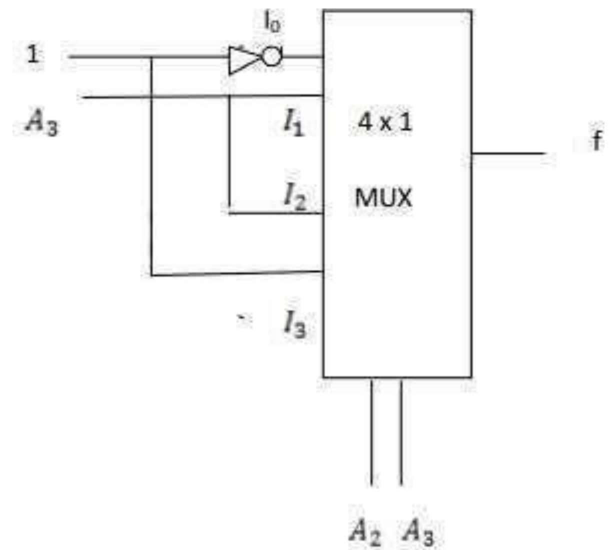
	I_0	I_1	I_2	I_3
$\overline{A_1}$	0	1	2	3
A_1	4	5	6	7
	0	A_1	A_1	1



MULTIPLEXERS

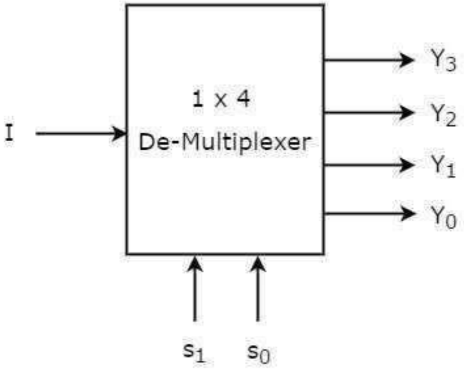
Method:2

Minterm	A_1	A_2	A_3	f		
0	0	0	0	0		
1	0	0	1	0	f= 0	I_0
2	0	1	0	0		
3	0	1	1	1	f= A_3	I_1
4	1	0	0	0		
5	1	0	1	1	f= A_3	I_2
6	1	1	0	1		
7	1	1	1	1	f= 1	I_3



DEMULTIPLEXER

- A demultiplexer 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.
- We have 1x2,1x4,8x1….Demultiplexers.



Boolean functions for each outputas

$$Y_3 = s_1 s_0 I$$

$$Y_2 = s_1 s_0' I$$

$$Y_1 = s_1' s_0 I$$

$$Y_0 = s_1' s_0' I$$

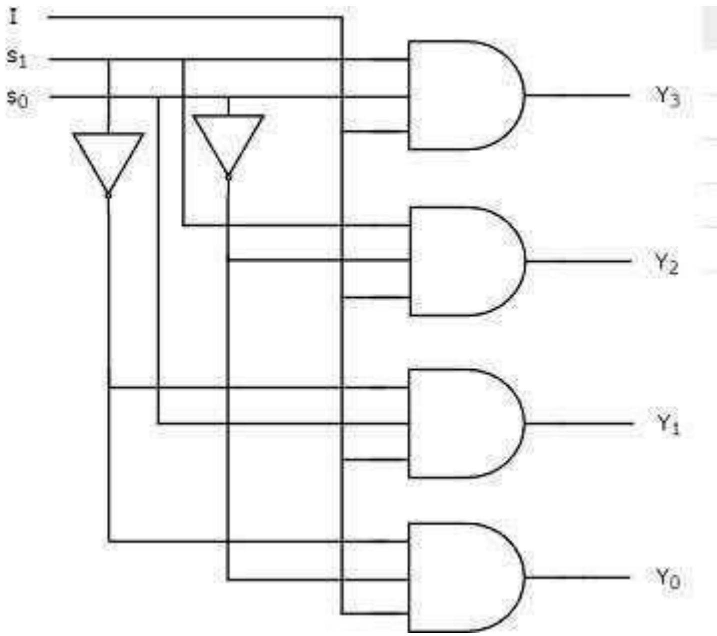


Fig:3 Logicdiagram

CODE CONVERTERS

➢A code converter is a logic circuit whose inputs are bit patterns representing numbers (or character) in one code and whose outputs are the corresponding representation in a different code.

Design of a 4-bit binary to gray codeconverter

4-bit binary				4-bit Gray			
B_4	B_3	B_2	B_1	G_4	G_3	G_2	G_1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Fig :1 Truthtable

CODE CONVERTERS

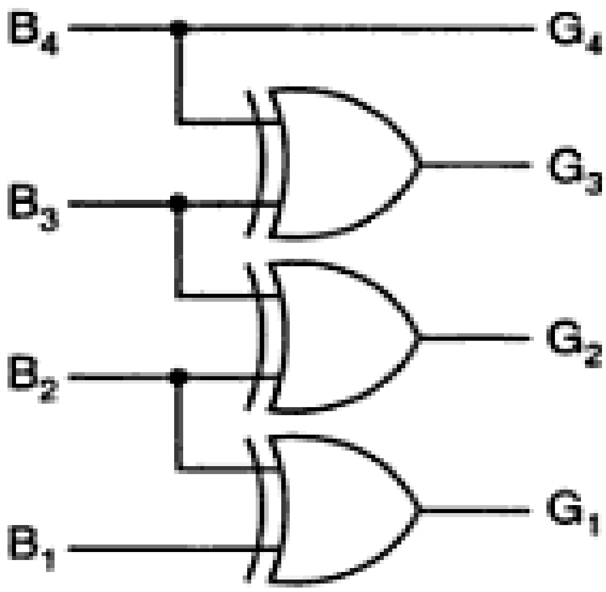


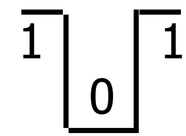
Fig: 2 Logicdiagram

HAZARDS AND GLITCHES

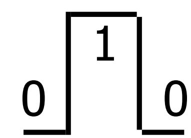
- **glitch**: unwanted output
- A circuit with the potential for a glitch has a **hazard**.
- Glitches occur when different pathways have different delays
  - Causes circuit noise
  - Dangerous if logic makes a decision while output is unstable
- Solutions
  - Design hazard-free circuit
  - Difficult when logic is multilevel
  - Wait until signals are stable

TYPES OF HAZARDS

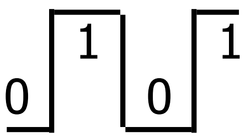
- Static 1-hazard
  - Output should stay logic 1
  - Gate delays cause brief glitch to logic 0

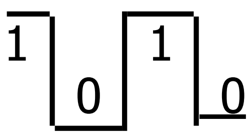


- Static 0-hazard
  - Output should stay logic 0
  - Gate delays cause brief glitch to logic 1



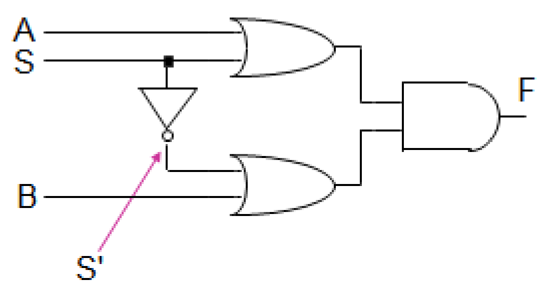
- Dynamic hazards
  - Output should toggle cleanly
  - Gate delays cause multiple transitions

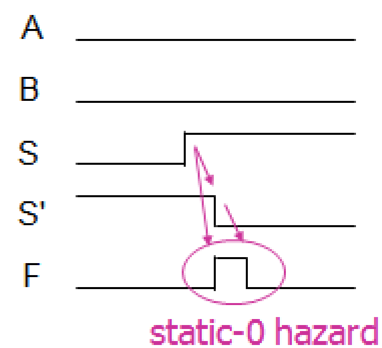




- Often occurs when a literal and its complement momentarily assume the same value
    - Through different paths with different delays
    - Causes an (ideally) static output to glitch

A multiplexer

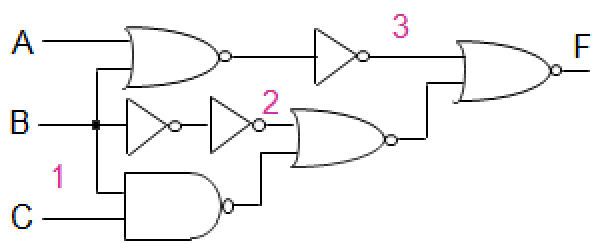


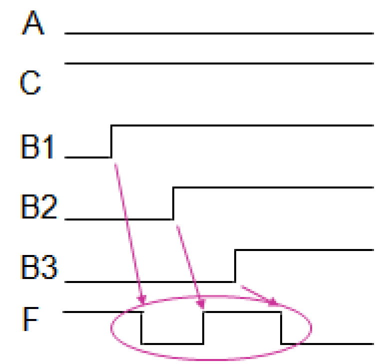


DYNAMIC HAZARDS

- Often occurs when a literal assumes multiple values
  - Through different paths with different delays
  - Causes an output to toggle multiple times

Dynamic hazards

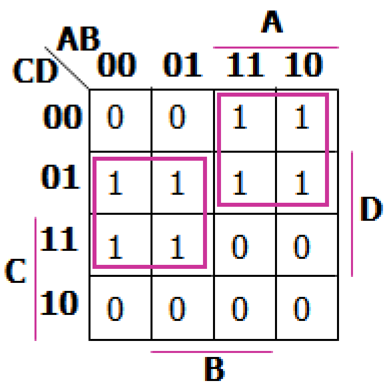




Dynamic hazard

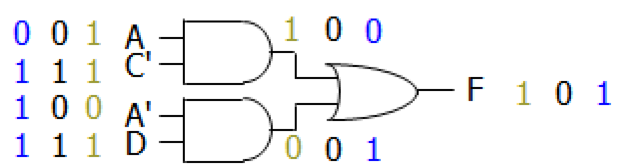
ELIMINATING STATIC HAZARDS

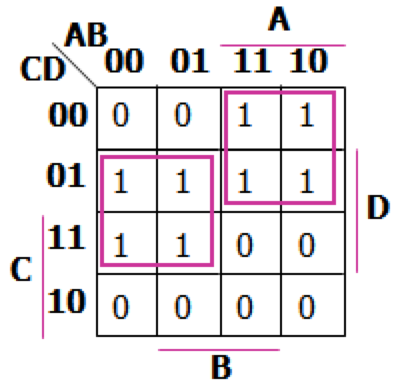
- Key idea: Glitches happen when a changing input spans separate K-map encirclements
  - Example: 1101 to 0101 change can cause a static-1 glitch



- ABCD: 1101 → 0101

F = AC' + A'D

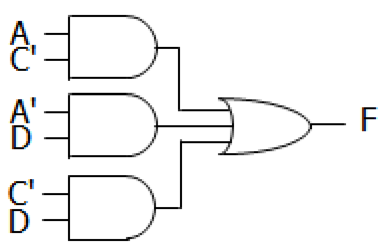




ELIMINATING STATIC HAZARDS

- Solution: Add redundant K-map encirclements
  - Ensure that all single-bit changes are covered by same block
  - First eliminate static-1 hazards: Use SOP form
  - If need to eliminate static-0 hazards, use POS form
- Technique only works for 2-level logic

F = AC' + A'D + C'D



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

ELIMINATING STATIC HAZARDS

- We can eliminate static hazards in 2-level logic for single-bit changes
  - Eliminating static hazards also eliminates dynamic hazards
- Hazards are a difficult problem
  - Multiple-bit changes in 2-level logic are hard
  - Static hazards in multilevel logic are harder
  - Dynamic hazards in multilevel logic are harder yet

MODULE-V
SEQUENTIAL CIRCUITS FUNDAMENTALS

Basic Architectural Distinctions between Combinational and Sequential circuits, SR Latch, Flip Flops: SR, JK, JK Master Slave, D and T Type Flip Flops, Excitation Table of all Flip Flops, Timing and Triggering Consideration, Conversion from one type of Flip-Flop to another.
Registers and Counters: Shift Registers – Left, Right and Bidirectional Shift Registers, Applications of Shift Registers - Design and Operation of Ring and Twisted Ring Counter, Operation of Asynchronous and
Synchronous Counters

CONTENTS

- Combinational and Sequential circuits,
- SR Latch, Flip Flops: SR, JK, JK Master Slave,
- D and T Type Flip Flops, Excitation Table of all Flip Flops,
- Timing and Triggering Consideration,
- Shift Registers – Left, Right and Bidirectional Shift Registers, Operation of Asynchronous and
- Synchronous Counters.

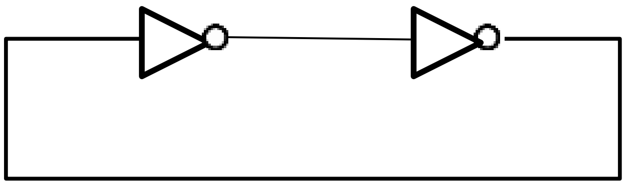
- Gated latch is a basic latch that includesinput gating and a control signal.

- The latch retains its existing state when the control input is equal to 0.

- Its state may be changed when the control signal is equal to 1.In our discussion we referred to the control input as the clock.

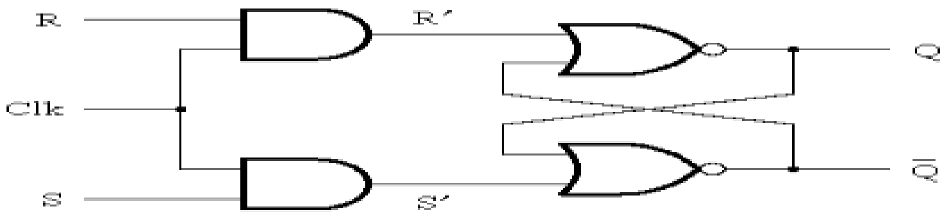
- We consider two types of gated latches:
    - Gated SR latch uses the *S* and *R* inputs to set the latch to 1
    - Gated D latch uses the *D* input to force the latch into a state that has the same logic value as the *D* input.

- Basic latch is a feedback connection oftwo NOR gates or two NAND gates.
- It can store one bit ofinformation.
- It can be set to 1 using the S input andreset to 0 using the *R*input.



- A feedback loop with even numberof inverters
- If A = 0, B = 1 or when A = 1, B =0
- This circuit is not useful due to the lack ofa mechanism for changing its state

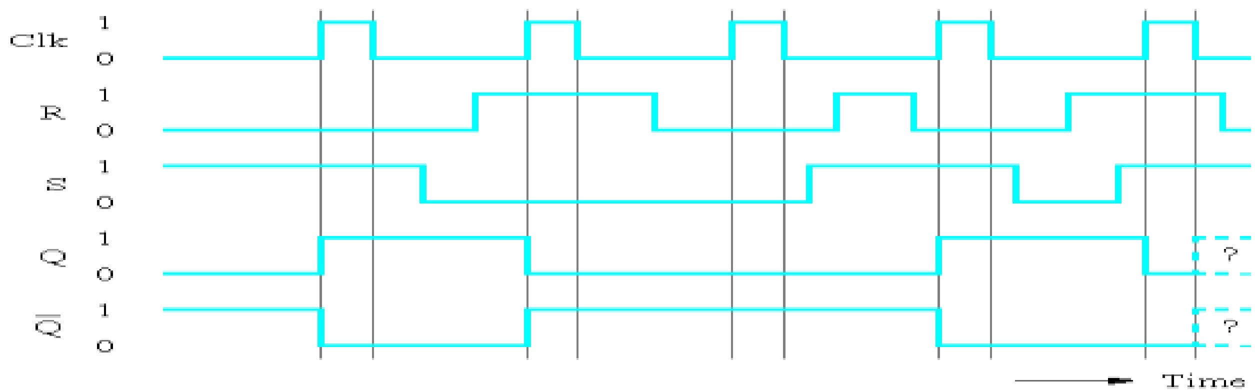
RS LATCH:



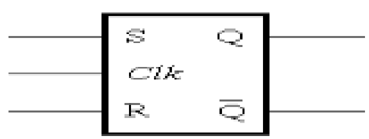
(a) Circuit

Clk	S	R	Q(t + 1)
0	x	x	Q(t) (no change)
1	0	0	Q(t) (no change)
1	0	1	0
1	1	0	1
1	1	1	x

(b) Characteristic table

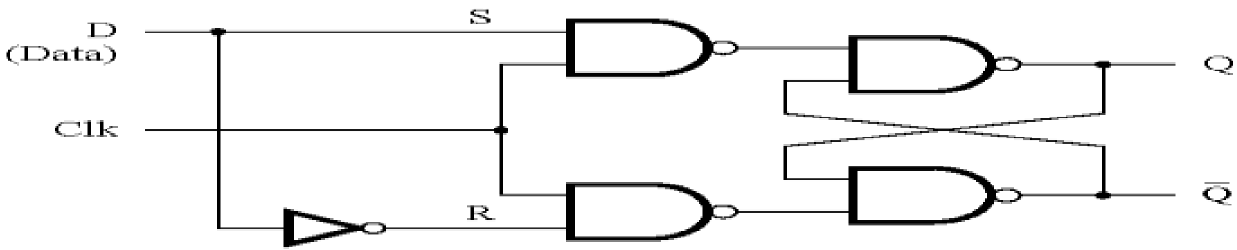


(c) Timing diagram



(d) Graphical symbol

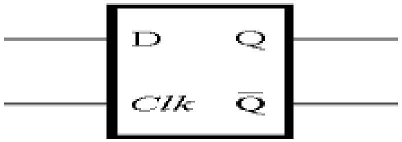
D LATCH:



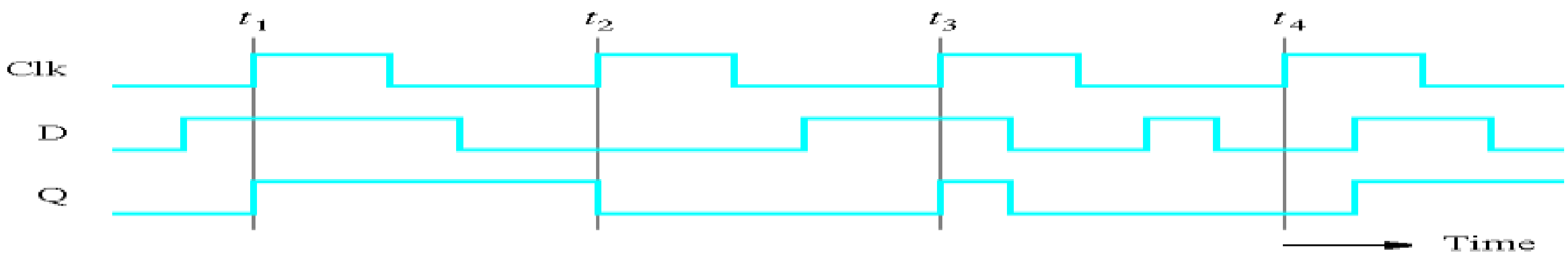
(a) Circuit

Clk	D	$Q(t+1)$
0	x	$Q(t)$
1	0	0
1	1	1

(b) Characteristic table



(c) Graphical symbol



(d) Timing diagram

FLIP FLOPS:

- A flip-flop is a storage element based onthe gated latch principle.
- It can have its output state changed onlyon the edge of the controlling clock signal.

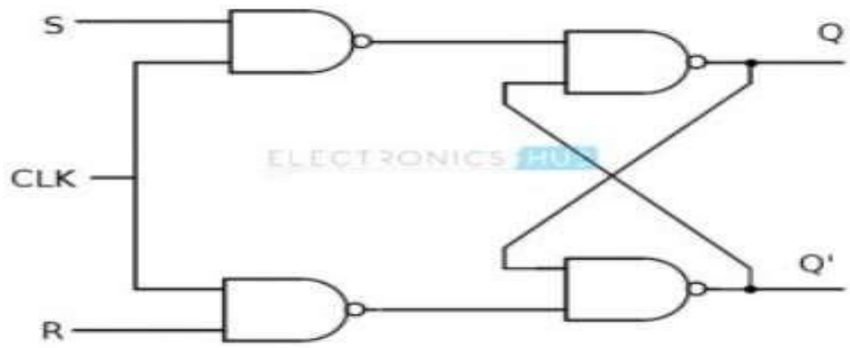
Types Of Flip-flops:

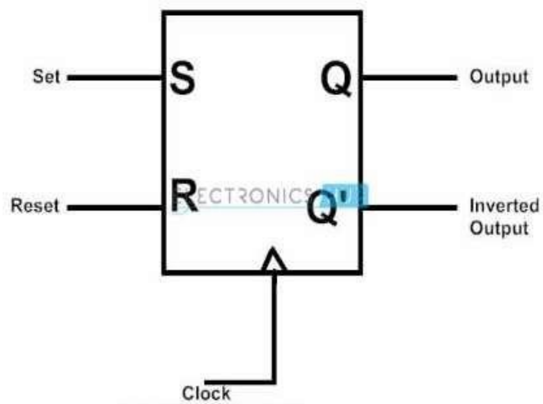
- SR flip-flop (Set, Reset)
- T flip-flop (Toggle)
- D flip-flop (Delay)
- JK flip-flop

FLIP FLOPS:

- Edge-triggered flip-flop is affected only by the input values present when the active edge ofthe clock occurs
- Master-slave flip-flop is built with twogated latches
- The master stage is active during half of the clock cycle, and the slave stage is active during the other half.
- The output value of the flip-flop changes on the edge of the clock that activates the transfer into the slave stage

SR FLIP FLOP



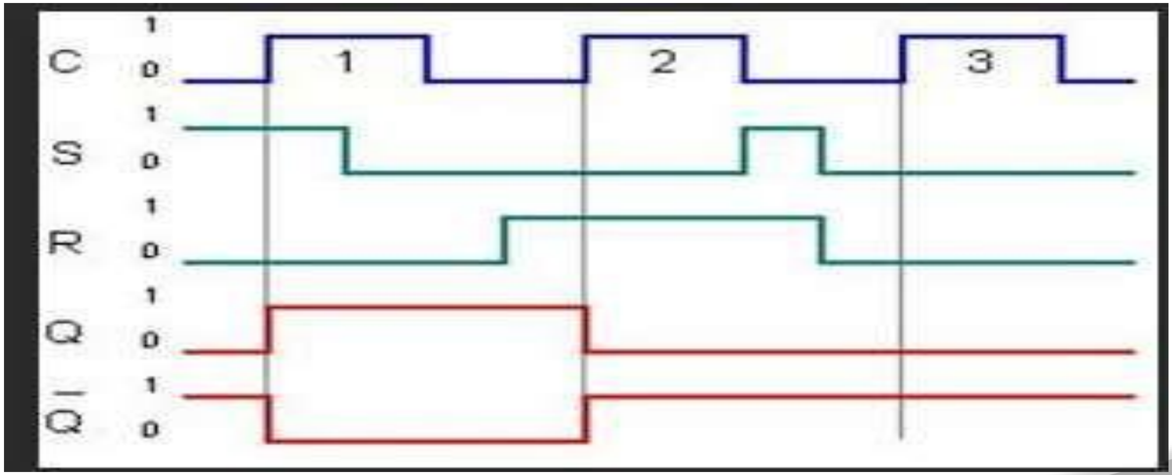


INPUTS			OUTPU T	STATE
CLK	S	R	Q	
X	0	0	No Change	Previous
↑	0	1	0	Reset
↑	1	0	1	Set
↑	1	1	-	Forbidde n

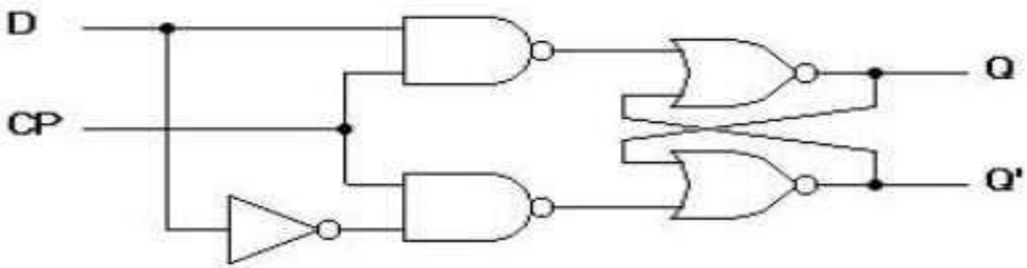
SR FLIPFLOP EXCITATION TABLE &TIMING DIAGRAM

SR FLIP-FLOP			
Q(t)	Q(t+1)	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

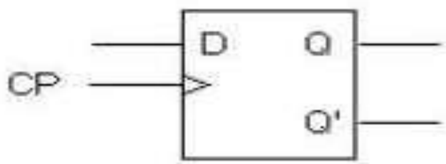
EXCITATION TABLE OF SR FLIP-FLOP



D FLIPFLOP:



(a) Logic diagram with NAND gates



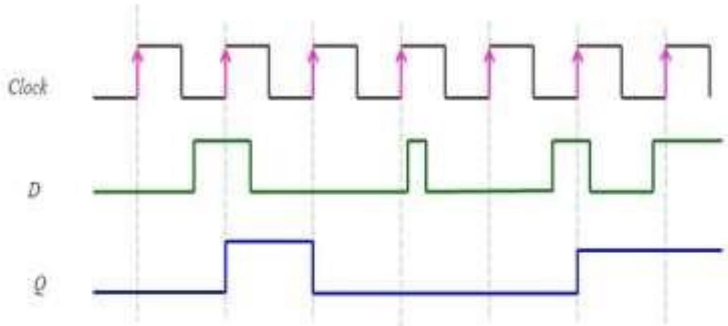
(b) Graphical symbol

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

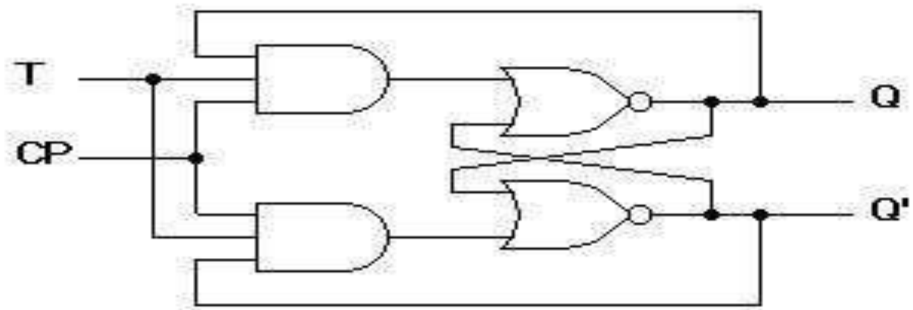
(c) Transition table

D FLIPFLOP EXCITATION TABLE &TIMING DIAGRAM:

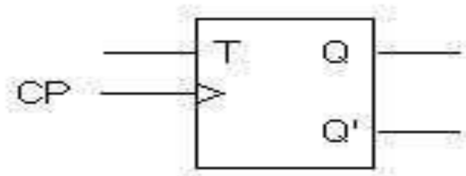
Present state (Q_n)	Next state (Q_{n+1})	D
0	0	0
0	1	1
1	0	0
1	1	1



T FLIPFLOP :



(a) Logic diagram

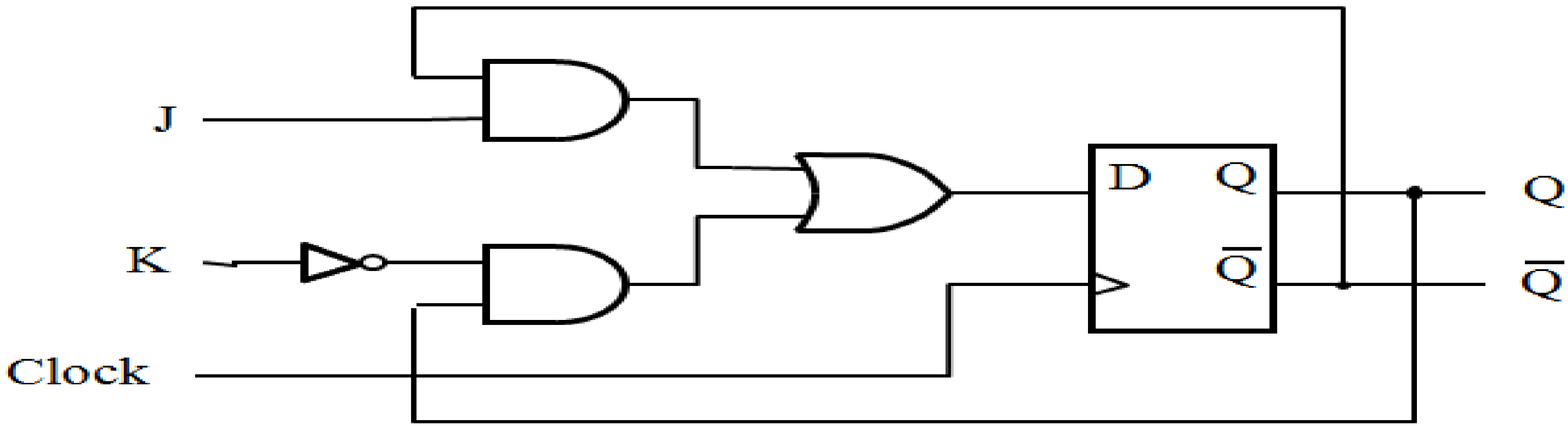


(b) Graphical symbol

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

(c) Transition table

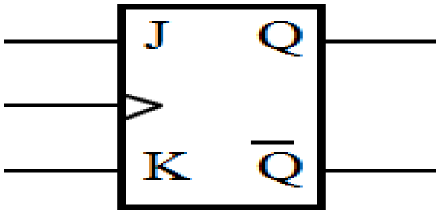
JK FLIPFLOP



(a) Circuit

J	K	Q (t+1)
0	0	Q (t)
0	1	0
1	0	1
1	1	$\overline{Q}$ (t)

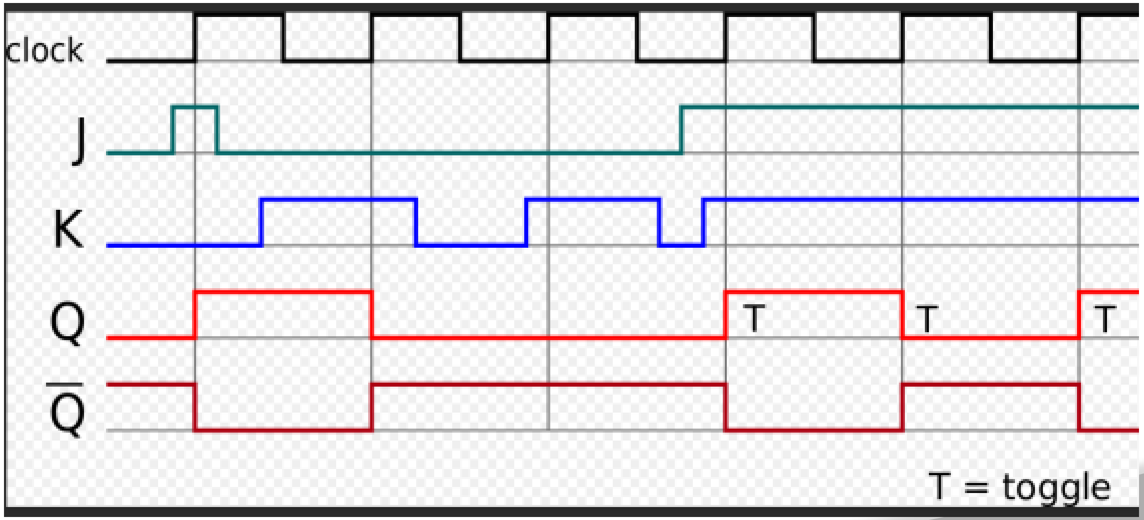
(b) Characteristic table



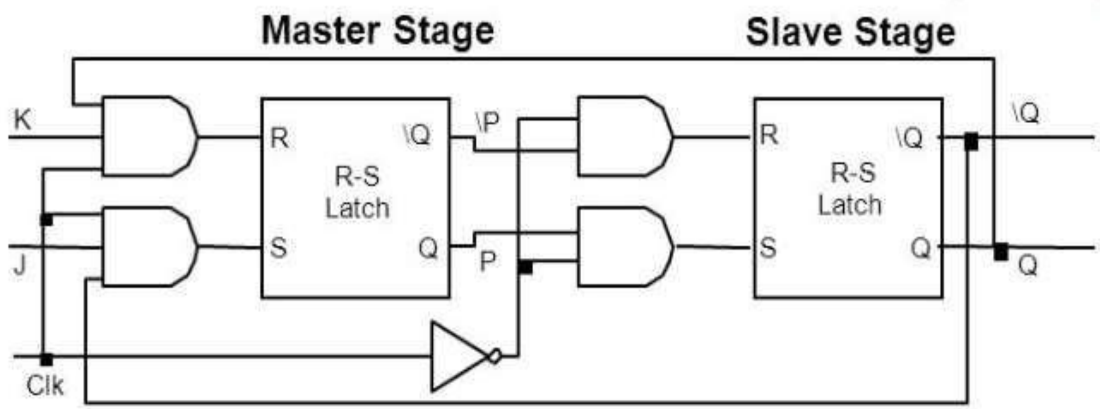
(c) Graphical symbol

JK FLIPFLOP

Q Output		Inputs	
Present State	Next State	J_n	K_n
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0



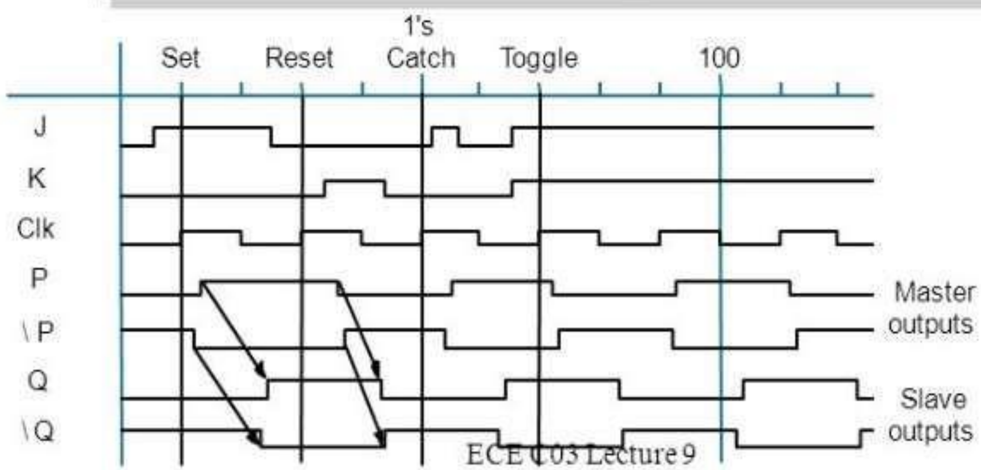
MASTER SLAVE JK FLIPFLOP



Sample inputs while clock high

Sample inputs while clock low

Uses time to break feedback path from outputs to inputs!



Correct Toggle Operation

CONVERSION OF FLIPFLOP:

1. Draw the block diagram of the target flip flop from the given problem.
2. Write truth table for the target flip-flop.
3. Write excitation table for the available flip-flop.
4. Draw k-map for target flip-flop.
5. Draw the block diagram.

JK TO SR FLIPFLOP:

- **Characteristic Table**

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

J	K	Q(t+1)	Operation
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$\bar{Q}(t)$	Complement (Toggle)

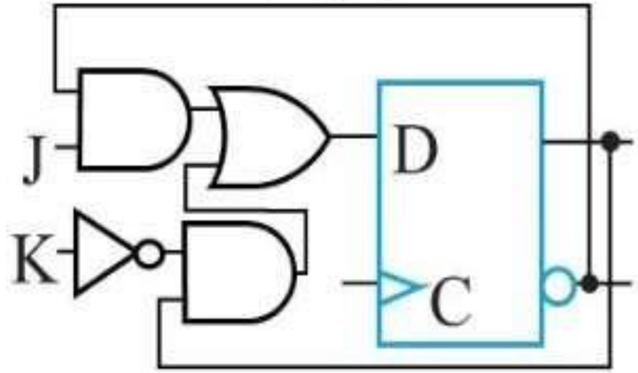
- **Characteristic Equation**

$$Q(t+1) = J\,\overline{Q} + \overline{K}\,Q$$

- **Excitation Table**

Q(t)	Q(t+1)	J	K	Operation
0	0	0	X	No change
0	1	1	X	Set
1	0	X	1	Reset
1	1	X	0	No Change

Change



CHARACTERISTIC EQUATION:

Flip-flop	Characteristic Equation
D	$Q(t+1) = D$
T	$Q(t+1) = T \oplus Q(t)$
SR	$Q(t+1) = S + R' Q(t)$
JK	$Q(t+1) = J Q(t)' + K' Q(t)$

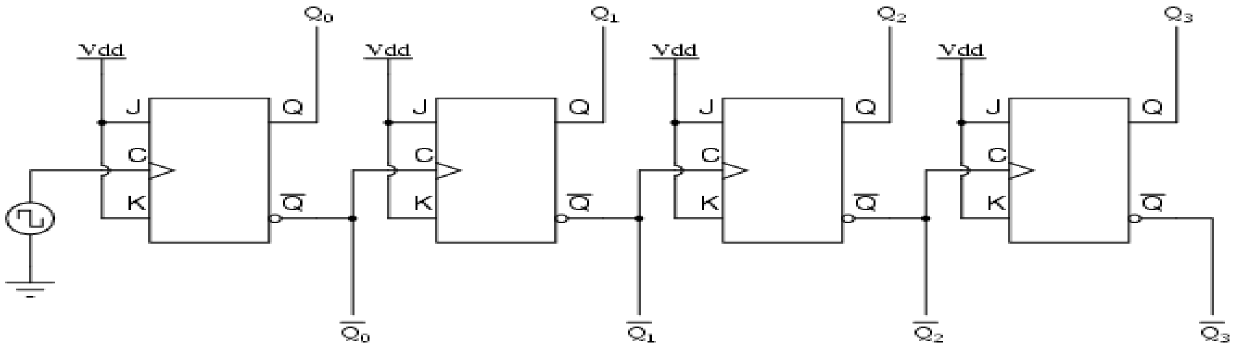
COUNTERS:

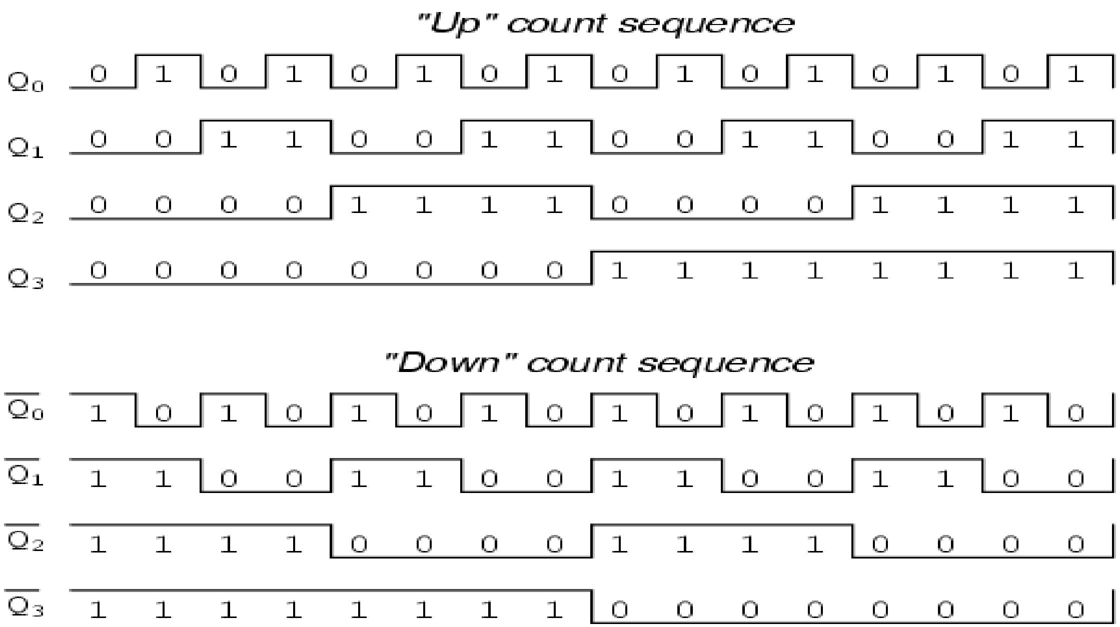
- Counters are a specific type of sequential circuit.
- Like registers, the state, or the flip-flop values themselves, serves as the "output."
- The output value increases by one on each clock cycle.
- After the largest value, theoutput "wraps around" back to 0.
- Counters can act as simple clocks to keep track of "time."

ASYNCHRONOUS COUNTERS:

- Asynchronous counter created from two JK flip-flops An asynchronous (ripple) counter is a single d-type flip-flop, with its J (data) input fed from its own inverted output. This circuit can store one bit, and hence can count from zero to one before it overflows (starts over from 0).

ASYNCHRONOUS UP/DOWN COUNTERS:





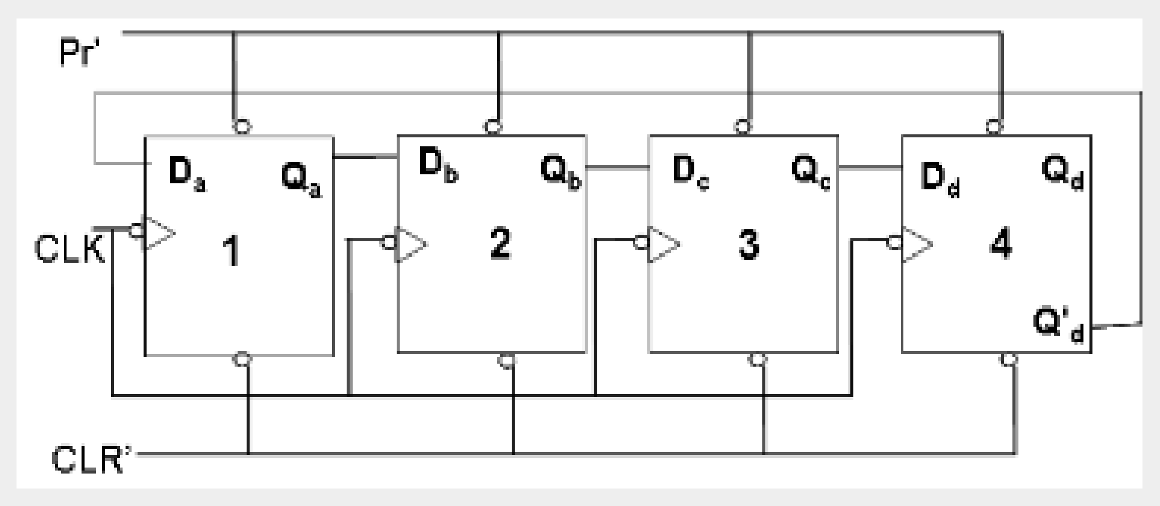
SYNCHRONOUS COUNTERS:

- The counters which use clock signal to change their transition are called "synchronous counters". this means the synchronous counters depends on their clock input to change state values. in synchronous counters, all flip flops are connected to the same clock signal and all flip flops will trigger at the same time.

Types of Counters:

- 4 bit Binary synchronous UP& DOWN counter
- 4 bit Binary synchronous UP / DOWN counter
- BCD counters
- Ring counters
- Johnson counters etc.

JOHNSON COUNTERS:



Johnson counter				
State	**Q0**	**Q1**	**Q2**	**Q3**
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1
0	0	0	0	0

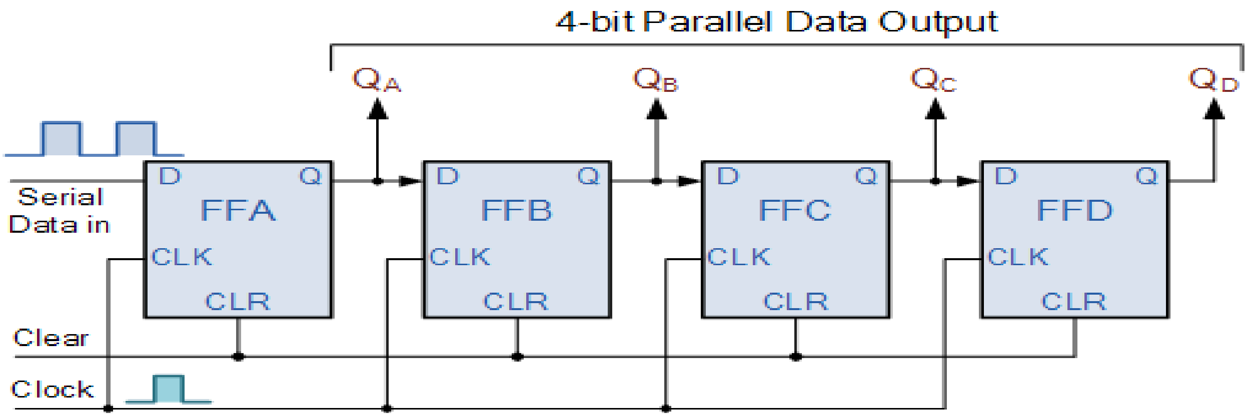
SHIFT REGISTER:

- Shift registers, like counters, are a form of sequential logic. Sequential logic, unlike Combinational Logic is not only affected by the present inputs, but also, by the prior history. In other words, sequential logic remembers past events. Basic shift registers are classified by structure according to the

Types of Shift Registers:

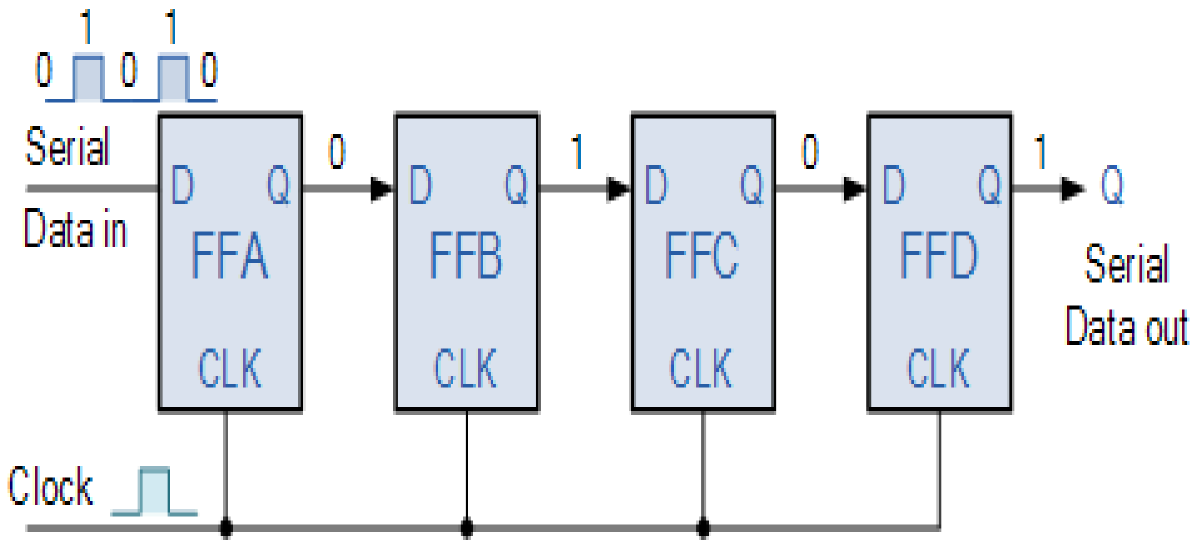
- Serial-in/serial-out
- Parallel-in/serial-out
- Serial-in/parallel-out
- Universal parallel-in/parallel-out

SERIAL IN TO PARALLEL OUTPUT:

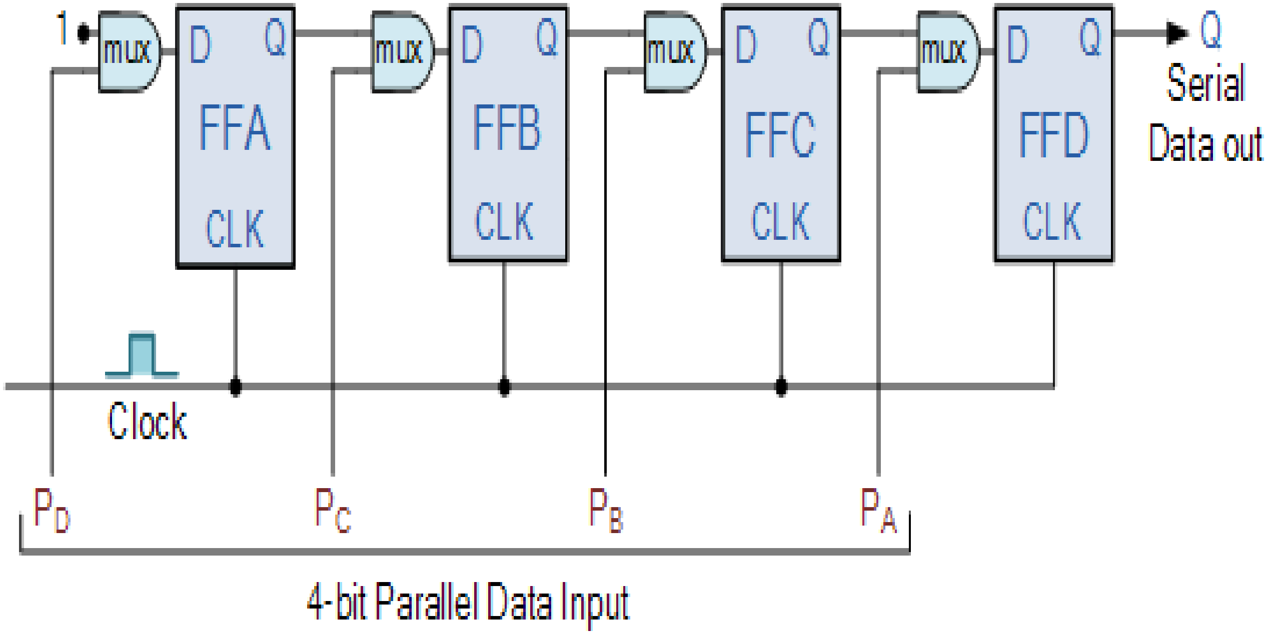


Clock Pulse No	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	0	0	0	0

SERIAL IN TO SERIAL OUTPUT:



PARALLEL IN TO SERIAL OUTPUT:



PARALLEL IN TO PARALLEL OUTPUT: