GOVERNMENT OF TAMILNADU
DIRECTORATE OF TECHNICAL EDUCATION
CHENNAI – 600 025

STATE PROJECT COORDINATION UNIT

Diploma in Instrumentation and Control Engineering

Course Code: 34241

M – Scheme

e-TEXTBOOK
on
ANALOG AND DIGITAL ELECTRONICS
for
IV Semester DICE

Convener for ICE Discipline:

Dr.S.Rajakumari,
Head of Department/ECE,
Dr.Dharmambal Govt. Polytechnic College for Women,
Tharamani, Chennai - 600 113

Team Members for Analog And Digital Electronics:

Tmt.Dr.S.R.Ruckmani M.E.,Ph.D.,
Lecturer (Sr.Gr.) / ICE,
210, CIT Sandwich Polytechnic College,
Coimbatore – 641 014

Mr.K.Annamalai M.E.,
Lecturer (SG) / ICE,
221, Seshasayee Institute of Technology,
Tiruchirappalli – 620 010

Tmt.D.Sivakami B.E.,
Lecturer (SG) / ICE,
280, A.D.J.Dharmambal Polytechnic College,
Nagapattinam – 611 001

Validated By

Tmt.M.Muzhumathi M.E.,
Lecturer (SG) / ECE,
277, Padmabhushan Sri N Ramaswami Ayyar Memorial
Polytechnic College for Girls
Tiruchirappalli – 620 002

ANALOG AND DIGITAL ELECTRONICS
DETAILED SYLLABUS

UNIT – I

LINEAR ICS: OP-AMPS, TIMERS AND THEIR APPLICATIONS

1.1 The operational amplifier:

An operational amplifier is a direct coupled high gain amplifier consisting of one or more differential (OPAMP) amplifiers and followed by a level translator and an output stage. An operational amplifier is available as a simplex integrated circuit package.

An operational amplifier is designed to perform mathematical operations like addition, subtraction, differentiation, integration, multiplication, division etc in analog computers. Hence it is called as operational amplifiers.

Symbol of Operational Amplifier:

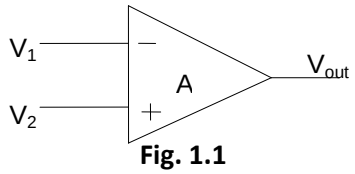


Fig. 1.1

1. In fig 1.1 an op.amp contains two inputs and a single output. The inputs are named as inverting input marked with a negative or "minus" sign, (-) and non-inverting input marked with a positive or "plus" sign (+).
2. If the non-inverting input is grounded and a signal is applied to the inverting input, the output signal will be 180 degrees out of phase with the input signal.
3. If the inverting input is grounded and a signal is applied to the non-inverting input, the output signal will be in phase with the input signal.

Block Diagram of Op-Amp

The block diagram of Op-Amp is shown in **Fig. 1.2**.

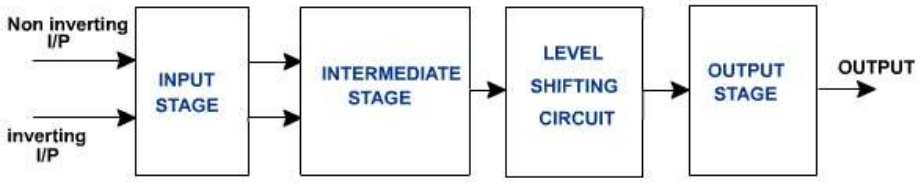


Fig. 1.2

The input stage is a dual input balanced output differential amplifier. This stage provides most of the voltage gain of the amplifier and also establishes the input resistance of the Op-Amp. The intermediate stage of Op-Amp is another differential amplifier which is driven by the output of the first stage. This is usually dual input unbalanced output.

Because of direct coupling, the dc voltage level at the output of intermediate stage is well above ground potential. Therefore level shifting circuit is used to shift the dc level at the output downwards to zero with respect to ground. The output stage is generally a push pull complementary amplifier. The output stage increases the output voltage swing and raises the current supplying capability of the Op-Amp. It also provides low output resistance.

AN IDEAL OP.AMP:

An ideal op.amp shown in **Fig. 1.3** is a differential input and a single ended output device.

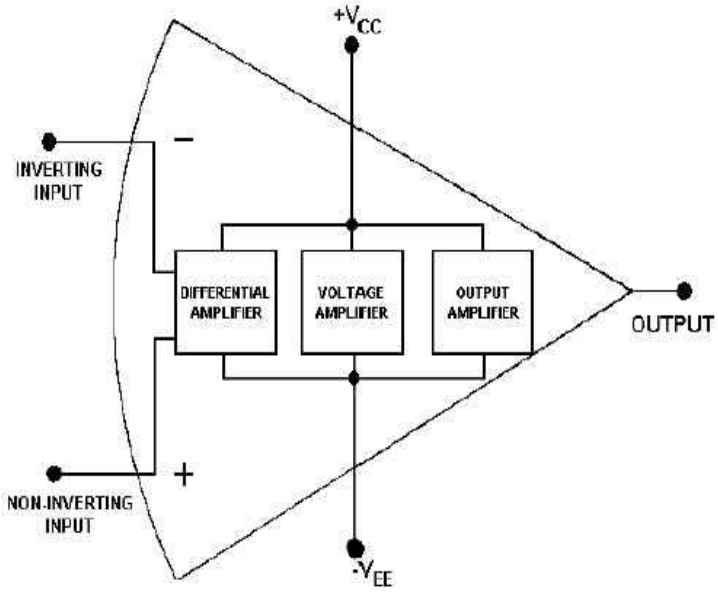


Fig. 1.3

CHARACTERISTICS OF AN IDEAL OP.AMP.

1. High input impedance $R_i = \infty$
2. Low output impedance $R_O = 0$
3. Infinite open loop gain $A_v = \infty$
4. Infinite bandwidth $BW = \infty$
5. Perfect balance $V_0 = O$ when $V_1 = V_2$

Characteristics do not change with temperature.

PIN DIAGRAM OF AN OP.AMP

Operational amplifiers are available in IC packages of either single, dual or quad op-amps within one single chip. The most commonly available operational amplifiers in basic electronic kits and projects is the industry standard **µA-741 IC s**how in **Fig. 1.4**

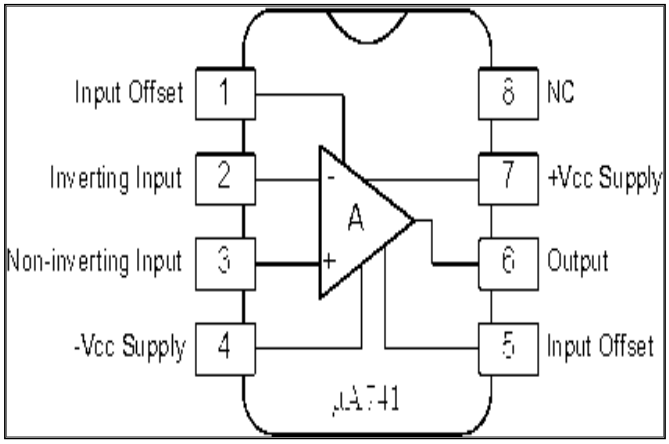


Fig. 1.4

Parameters of OP AMP (Specifications):

Sl. No.	Parameters
1	Input offset Voltage
2	Input offset current
3	Input Bias Current
4	Large Signal Voltage gain
5	Output Voltage Swing
6	Differential input resistance Ri
7	Input Capacitance Ci
8	Common Mode Rejection Ratio(CMRR)
9	Supply voltage Rejection Ratio
10	Slew Rate
11	Gain Bandwidth Product
12	Maximum differential input voltage
13	Maximum common mode input voltage

1)Input offset Voltage

Input offset voltage is the voltage that must be applied between the two input terminals of an op-amp to null the output.

2) Input offset current

The algebraic difference between the current in the inverting and non inverting terminal is known as the input offset current

3) Input Bias Current

IB is the average current flows in the inverting and non-inverting terminal of an op-amp.

IB = (IB1 + IB2)/2

4) Large Signal Voltage gain

It is the ratio of the output voltage and the differential input voltage

A = Outputvoltage/Differential input voltage = Vo/Vid

5) Output Voltage Swing

This parameter indicates the values of positive and negative saturation voltage of the op-amp. For 741 IC, it is +13 and -13V.

6) Differential input resistance Ri

Differential input resistance Ri is the equivalent resistance that can be measured at either the inverting or non-inverting input terminals with the other terminal connected to ground.

Typical value for 741 IC is 2 mega ohm.

7) Input Capacitance Ci

Input capacitance is the equivalent capacitance that can be measured at either the inverting or non-inverting input terminal with the other terminal connected to ground.

Typical value for a 741 IC is 1.4 pF.

8) Common Mode Rejection Ratio(CMRR)

When the same voltage is applied to both the input terminals the voltage is called a common mode voltage(Vcm) and the op-amp is said to be operating in the common mode configuration,CMRR is defined as the ratio of the differential mode voltage gain ($A_{d)}$ and common mode voltage gain.

CMRR = Ad/Acm

9) Supply voltage Rejection Ratio

The change in an op-amps input offset voltage Vio caused by variations in the supply voltage is called the SVRR.It is expressed in microvolts per volt or in decibels.

SR= ΔVio/ΔV

10) Slew Rate

Slew rate is defined as the maximum rate of change of output voltage per unit of time and is expressed as volts per micro second.

11) Gain Bandwidth Product

The gain bandwidth product (GB) is the bandwidth of the op-amp when the voltage gain is 1. Typical value for 741 IC is 1MHz.

12) Maximum differential input voltage

It is the maximum value of differential input voltage that can be applied without damaging the op-amp.

13) Maximum common mode input voltage

It is the maximum voltage to which that the two inputs can be raised above ground potential before the op-amp becomes non-linear

VIRTUAL GROUND

In op-amp, the differential input voltage is ideally zero, that is the voltage at the inverting terminal V_1 is approximately equal to that at the non inverting terminal V_2.In other words, the inverting terminal voltage V_1 is approximately at ground potential. The inverting terminal is not directly connected to ground. But it acts like a ground terminal. Therefore the inverting terminal is said to be at virtual ground

ADVANTAGES OF OP-AMP

i) Inexpensive
ii) Versatile
iii) Easy to use

APPLICATIONS OF OP-AMP

i) Used in negative amplifiers.
ii) Used in wave shaping circuits
iii) Used in filter circuit
iv) Used in many mathematical operations.
v) Used as inverting amplifier
vi) Used as an non-inverting amplifier
vii) Other applications includes
   (a) summing amplifier
   (b) multiplier or scale changer
   (c) integrator
   (d) differentiator
   (e) Voltage follower.
   (f) Comparator
   (g) Zero crossing detector,etc

1.2 INVERTING AMPLIFIER:

If an input is applied to the inverting terminal, the output is 180 out of phase with the input signal. Hence it is called as INVERTING AMPLIFIER.

The Fig. 1.5 shows an inverting amplifier using op amp. Input is applied to the inverting terminal through R_{in}. Non-inverting terminal is grounded. A suitable resistor R_F is connected across the amplifier from the output terminal back to the inverting input terminal This negative feedback is used to **reduce and control the overall gain of the amplifier**. The junction of the input and feedback signal (X) is at the same potential as the

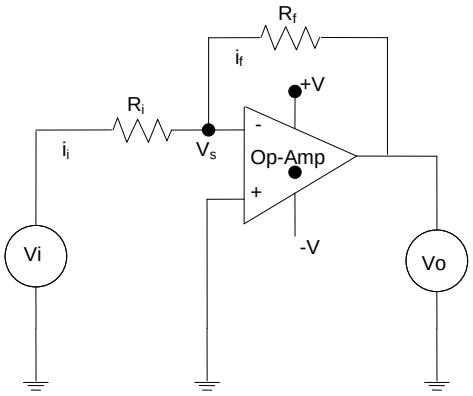


Fig. 1.5

positive (+) input which is at zero volts or ground then, the junction is said to be in **"Virtual ground."**

Due to its virtual ground, No Current Flows into the Input Terminals and the Differential Input Voltage is Zero as V1 = V2 = 0.Hence, the current flowing through the input resister R_{in} is equal to the current flowing through the feedback resistor R_F

Applying Kirchoff's voltage law,

$$i_n = i_f$$

$$\frac{V_i - V_s}{R_i} = \frac{V_s - V_0}{R_f}$$

Substitute $V_s = 0$ due to virtual ground

$$\frac{V_i - 0}{R_i} = \frac{0 - V_0}{R_f}$$

$$\frac{V_i}{R_i} = \frac{-V_0}{R_f}$$

$$\frac{R_f}{R} v_i = -v_o$$

$$\frac{R_f}{R} v_{i=-Vo}$$

Then, the **Closed-Loop Voltage Gain** of an Inverting Amplifier is given as.

$$\text{Gain (Av)} = \frac{V_{out}}{V_{in}} = -\frac{R_f}{R_{in}}$$

and therefore Vout is

$$Vout = -\frac{Rf}{Rin} \times Vin$$

From the above equation, the output voltage of an inverting amplifier is equal to the input voltage multiplied with gain. The minus sign indicates input is given to inverting terminal.

If $RF = R_{IN} = R$, V out = -Vin

1.3 The Summing Amplifier:

As its name suggests, the output of the "summing amplifier" is the sum of input voltages in reverse direction.It is also called as *summing inverter*" or a "*voltage adder*" .

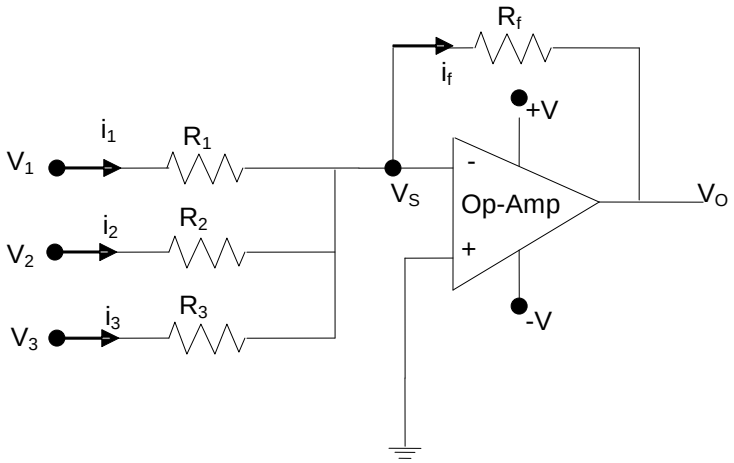


Fig. 1.6

The above circuit shows in fig 1.6 the summing amplifier using op amp. In this circuit,the output voltage, (Vout) proportional to the sum of the input voltages, V1, V2, V3etc.

According to Kirchoff's current law, the sum of the current flowing through input resistors is equal to the current flowing through feedback resister.

$$i_n = i_f$$

$$i_n = i_1 + i_2 + i_3 = i_f$$

$$\frac{V_1 - V_s}{R_1} + \frac{V_2 - V_s}{R_2} + \frac{V_3 - V_s}{R_3} = \frac{V_s - V_0}{R_f}$$

Substitute $V_s = 0$ due to virtual ground

$$\frac{V_1 - 0}{R_1} + \frac{V_2 - 0}{R_2} + \frac{V_3 - 0}{R_3} = \frac{0 - V_0}{R_f}$$

$$\frac{V_1}{R_1} + \frac{V_2}{R_2} + \frac{V_3}{R_3} = -\frac{V_0}{R_f}$$

If $R_1 = R_2 = R_3 = R$,

$$\frac{V_1}{R} + \frac{V_2}{R} + \frac{V_3}{R} = -\frac{V_0}{R_f}$$

$$\frac{1}{R}\,(V_1 + V_2 + V_3) = -\frac{V_0}{R_f}$$

$$-\frac{R_f}{R}\,(V_1 + V_2 + V_3) = V_0$$

If R_f = R,

$$V_0 = -\,(V_1 + V_2 + V_3) \times \frac{R}{R}$$

$$\boldsymbol{V_0 = -\,(V_1 + V_2 + V_3)}$$

From the above equation, the output voltage is equal to the sum of the input voltages. The minus sign indicates input is given to inverting terminal.

1.4 The Non-inverting Operational Amplifier

If an input is applied to the non-inverting terminal ,the output in phase with the input signal.Hence it is called as NON- INVERTING AMPLIFIER.So, **the output gain of the amplifier becomes "Positive" in value in contrast to the "Inverting Amplifier" circuit The above figure shows a non- inverting amplifier using op amp. Input is directly**

applied to the non- inverting terminal.Inverting terminal is grounded through the resistor R_2..A suitable resistor

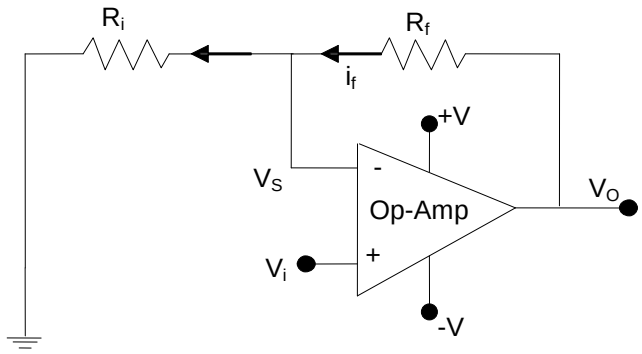


Fig. 1.7

R_F is connected across the amplifier from the output terminal back to the inverting input terminal .This negative feedback is used to **reduce and control the overall gain of the amplifier**

The above fig 1.7 shows a non- invertin amplifier using op amp. Input is directly applied to the non- inverting terminal.Inverting terminal is grounded through the resistor R_i..A suitable resistor R_F is connected across the amplifier from the output terminal back to the inverting input terminal .This negative feedback is used to **reduce and control the overall gain of the amplifier**.

Applying kirchoff's current law at the inverting terminal,

$$i_i = i_f$$

$$\frac{V_s}{R_i} = \frac{V_0 - V_s}{R_f}$$

Substitute $V_s = V_i$ in the the above equation,

$$\frac{V_i}{R_i} = \frac{V_0 - V_i}{R_f}$$

$$\frac{V_i}{R_i} = \frac{V_0}{R_f} - \frac{V_i}{R_f}$$

$$\frac{V_0}{R_f} - \frac{V_i}{R_f} = \frac{V_i}{R_i}$$

$$\frac{V_0}{R_f} = \frac{V_i}{R_i} + \frac{V_i}{R_f}$$

$$\frac{V_0}{R_f} = V_i \left(\frac{1}{R_i} + \frac{1}{R_f}\right)$$

$$V_0 = V_i R_f \left(\frac{1}{R_i} + \frac{1}{R_f}\right)$$

$$V_0 = V_i \left(\frac{R_f}{R_i} + \frac{R_f}{R_f}\right)$$

$$Vo = V_i \left(\frac{R_f}{R_i} + 1\right)$$

Then the closed loop voltage gain of a **Non-inverting Operational Amplifier** will be given as

$$\frac{V_o}{V_i} = \left(1 + \frac{R_f}{R_i}\right)$$

$$A_V = \frac{V_o}{V_i} = \left(1 + \frac{R_f}{R_i}\right)$$

From the above equation, the overall closed-loop gain of a non-inverting amplifier will always be greater but never less than one (unity), it is positive in nature and is determined by the ratio of the values of Rf and Ri.

1.5 Voltage Follower:

Voltage follower is an example of non-inverting amplifier.The output of the voltage follower follows the input voltage.Hence , it is called as unity gain amplifier.

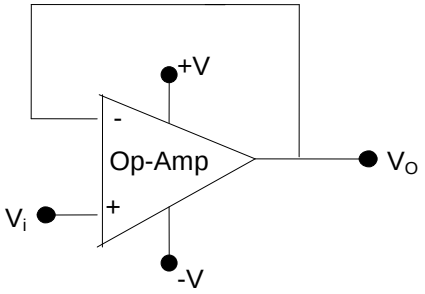


Fig. 1.8

In fig 1.8 Input is applied to the non-inverting terminal. Output is directly fed back to the inverting terminal without any feedback component.

The output of a non-inverting amplifier is given as

$$V_0 = \left(1 + \frac{R_f}{R_i}\right) V_i$$

If $R_f = 0$ & $R_i = \infty$,

the output voltage V_0 becomes

$$V_0 = \left(1 + \frac{0}{\infty}\right) V_i$$

$$= (1 + 0)\, V_i$$

$$\boldsymbol{V_0 = V_i}$$

From the above equation, the output voltage V_0 follows the input voltage V_i. Hence it is called as voltage follower

$$\text{Gain } A_V = \frac{V_0}{V_i} = 1$$

Hence, The gain of the voltage follower is unity. It is otherwise called as unity gain amplifier.

Applications:

1. used for impedance matching.
2. used as buffer.
3. used as unity gain amplifier.

1.6 COMPARATOR

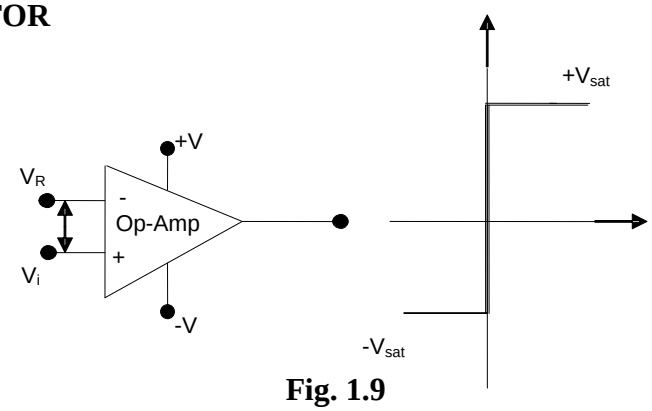


Fig. 1.9

Comparator compares two voltages of fixed reference voltage V_R and an analog input voltage V_i.

Input voltage V_i is applied to the non-inverting terminal and reference voltage V_R is applied to the inverting terminal shown in fig 1.9 If input voltage V_i is greater than V_{Ref}, the output of the op amp goes to positive saturation level.

If input voltage V_i is less than V_{Ref}, the output of the op amp goes to negative saturation level.

If the input voltage is equal to V_{Ref}, the output is equal to zero.

It is given as

i) if $V_i > V_{Ref}$, $V_0 = +V_{sat}$

ii) if $V_i > V_{Ref}$, $V_0 = -V_{sat}$

iii) if $V_i = V_{Ref}$, $V_0 = 0$

Application:

Used as sine wave to square wave generators

1.7 SINE WAVE TO SQUARE WAVE GENERATOR:

It is one of the applications of comparator. In fig1.10 Input is applied to the non-inverting input of op.amp. The inverting terminal of op.amp is grounded. So, the voltage at the inverting input is zero.

During the positive half cycle, the voltage at the non-inverting terminal is greater than the voltage at the inverting terminal. So, the output goes to positive saturation$+V_{sat}$.

During the negative half cycle, the voltage at the non-inverting terminal is less than the voltage at the inverting terminal. So, the output goes to negative saturation $-V_{sat}$.

Thus, a sine wave at the input of op amp is converted into square wave without changing the frequency.

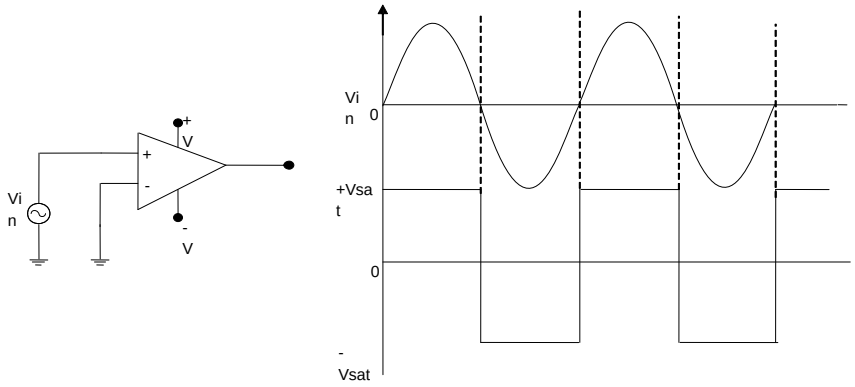


Fig. 1.10

1.8 ZERO CROSSING DETECTOR:

Zero crossing detectors are a circuit whose output changes from one state to another state when the input signal passes through zero.

Pn fig 1.11 It consists of comparator, differentiator and half wave rectifier. The comparator produces rectangular signal with respect to the input signal. Then the differentiator produces positive and negative spike according to the differentiator input. The diode D rectifies and produces positive pulse only. (i.e.) when the input signal crosses from negative to positive voltage through zero, positive pulses are produced at its output. It is an another application of comparator. If the inverting terminal of comparator is grounded, it act as zero crossing detector.

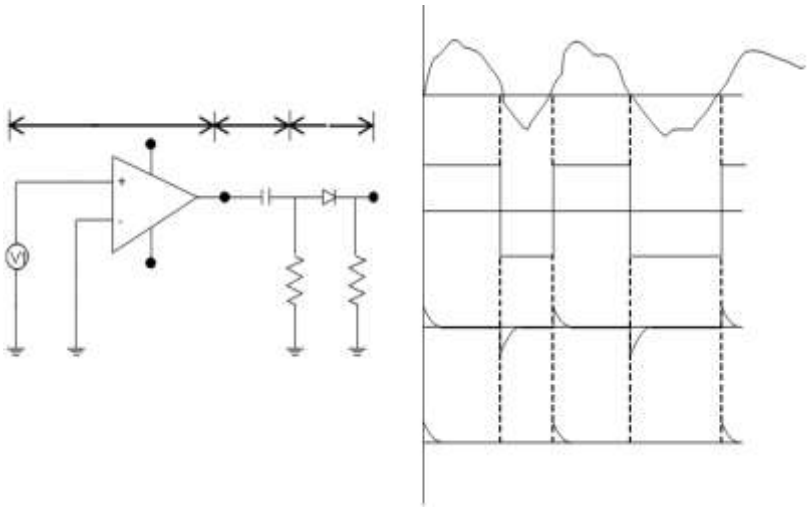


Fig. 1.11

1.9 INTEGRATOR

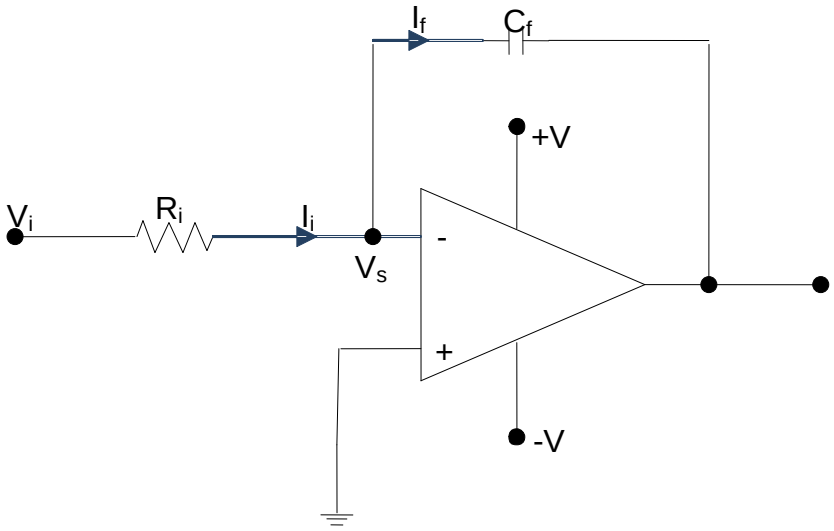


Fig. 1.12

An integrator is a circuit shown in fig 1.12 whose output voltage is proportional to the integral of the input voltage. Input V_i is applied to the inverting terminal through R_i and feedback is applied through capacitor C_f. The non-inverting terminal is grounded. When input voltage is applied, the charge in the capacitor C_f is given as

$$Q = CV$$

We know that current through the capacitor

$$I_C = \frac{dQ}{dt}$$

where Q =CV

$$\therefore I_C = \frac{dcv}{dt}$$

$$I_C = C\left(\frac{dv}{dt}\right)$$

where c is constant

Applying KCL,

$$i_i = i_f$$

$$\frac{V_i - V_s}{R_i} = C_f \frac{dv}{dt}$$

where V = $V_s - V_0$

$$\frac{V_i - V_s}{R_i} = C_f \frac{d(V_S - V_0)}{dt}$$

Due to virtual ground $V_S = 0$

$$\frac{V_i - 0}{R_i} = C_f \frac{d(0 - V_0)}{dt}$$

$$\frac{V_i}{R_i} = -C_f \frac{dV_0}{dt}$$

$$C_f \frac{dV_0}{dt} = -\frac{V_i}{R_i}$$

$$\frac{dV_0}{dt} = -\frac{V_i}{R_i C_f}$$

Integrate on both sides, we get

$$\int \frac{dV_0}{dt} = -\int \frac{V_i}{R_i C_f}$$

$$\boldsymbol{V_0 = -\frac{1}{R_i C_f} \int V_i dt}$$

From the above equation, the output voltage is equal to the integral of input voltage in reverse direction.

If a square wave input is applied to the integrator, it will produce triangular output show in fig 1.13

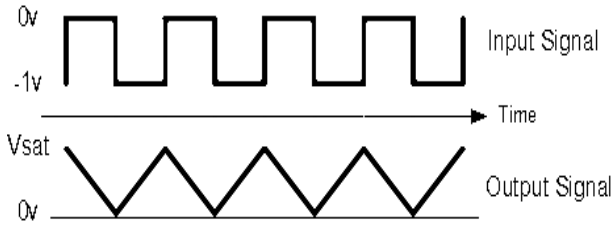


Fig 1.13

1.10 DIFFERENTIATOR:

Differentiator is a circuit whose output is directly proportional to the derivative of the input in reverse direction. Shown fig 1.14 Input is applied to the inverting terminal through C_i and feedback is applied through R_f. The non-inverting terminal is grounded. When an input voltage V_i is applied, the charge on the capacitor is

Q = CV

We know that the current through the capacitor is

$$I_C = \frac{dQ}{dt}$$

$= \frac{d(CV)}{dt}$ where Q = CV

$I_C = C\frac{dV}{dt}$ (C is constant)

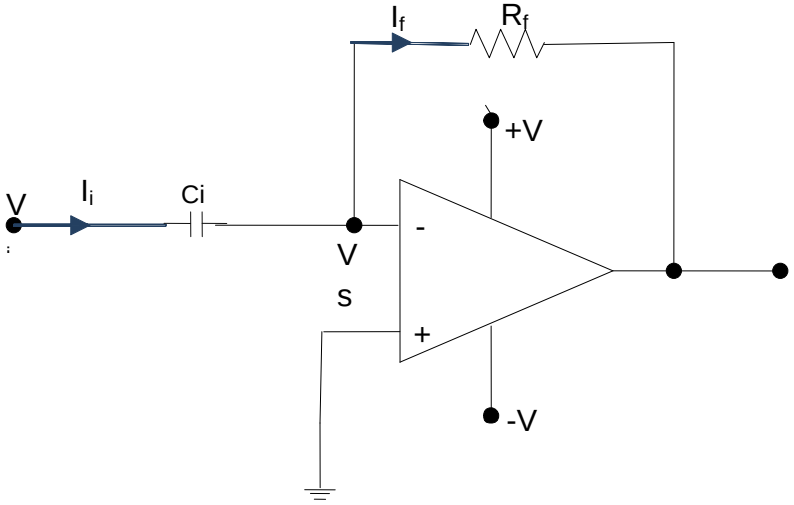


Fig. 1.14

Apply KCL,

$$i_i = i_f$$

$$C\frac{d(Vin)}{dt} = \frac{V_S - V_0}{R_f} \quad (Vin = V_i - V_s)$$

$$C\frac{d(V_i - V_s)}{dt} = \frac{V_S - V_0}{R_f}$$

Due to virtual ground $V_s = 0$

$$C_i\frac{dV_i}{dt} = -\frac{V_0}{R_f}$$

$$\frac{V_0}{R_f} = -C_i\frac{dV_i}{dt}$$

$$\boldsymbol{V_0 = -C_iR_f\frac{dV_i}{dt}}$$

From the above equation the output voltage is equal to the derivative of the input voltage in reverse direction (with respect to time). If a square wave input is applied to the differentiator, it will produce a spike output shown fig 1.15

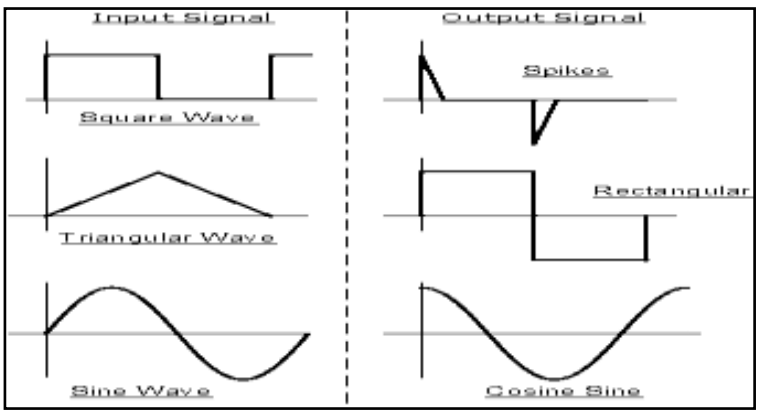


Fig 1.15

1.11 Timer IC.

Introduction 555 timer IC

The 555 timer IC was introduced in the year 1970 by Signetic Corporation and gave the name **SE/NE 555 timer**. It is basically a monolithic timing circuit that produces accurate and highly stable time delays or oscillation. When compared to the applications of an op-amp in the same areas, **the 555IC is also equally reliable and is cheap in cost**.

Its applications are

monostable multivibrator **and** astable multivibrator**,**

dc-dc converters

digital logic probes**,**

waveform generators**,**

analog frequency meters

and tachometers, temperature measurement

and control devices, voltage regulators etc.

The timer IC is setup to work in either of the two modes – one-shot or monostable or as a free-running or astable multivibrator.The **SE 555** can be used for temperature ranges between –55°C to 125°. The **NE 555** can be used for a temperature range between 0°to 70°C.

The important features of the 555 timer are:

1. It operates from a wide range of **power supplies** ranging from +5 Volts to +18 Volts supply voltage.
2. Sinking or sourcing 200 mA of load current.
3. The output of a 555 timer can drive transistor-transistor logic (TTL) due to its high current output.
4. It has a temperature stability of 50 parts per million (ppm) per degree Celsius change in temperature, or equivalently 0.005 % / °C.
5. The duty cycle of the timer is adjustable

IC Pin Configuration

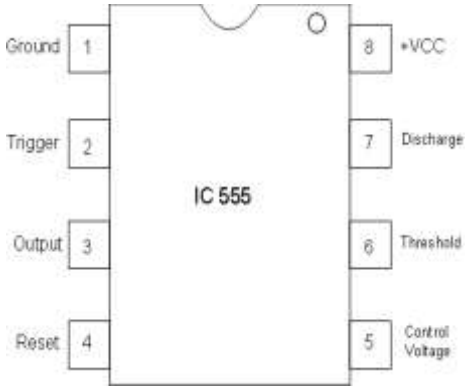


Fig. 1.16

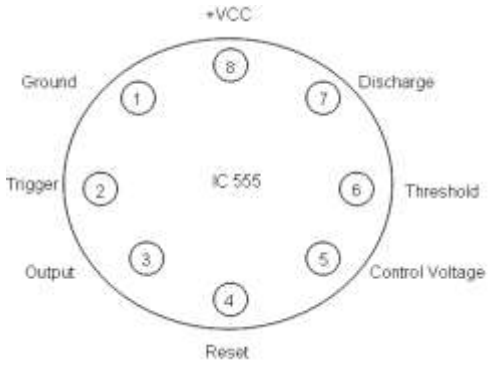


Fig. 1.17

The 555 Timer IC is available as an 8-pin metal can, an 8-pin mini DIP (dual-in-package) or a 14-pin DIP. The pin configuration is shown in the figures. 1. 16 and 1.17

This IC consists of 23 transistors, 2 diodes and 16 resistors. The use of each pin in the IC is explained below. The pin numbers used below refers to the 8-pin DIP and 8-pin metal

can packages. These pins are explained in detail, and you will get a better idea after going through the entire post

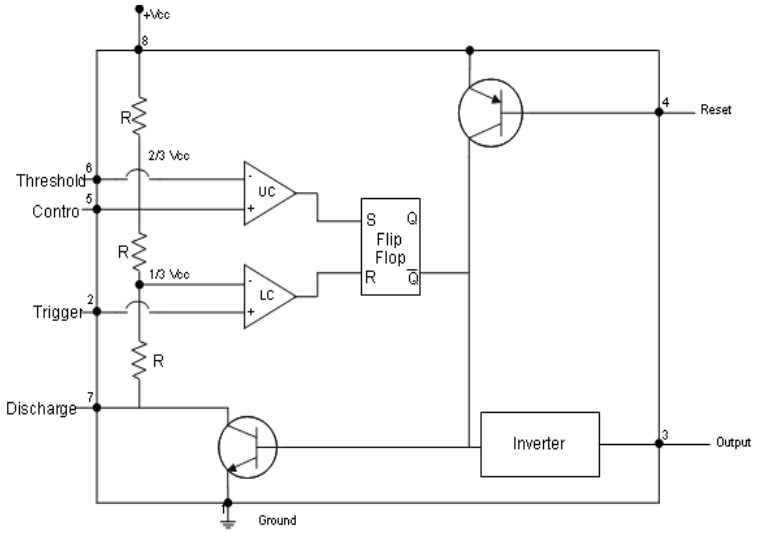


Fig. 1.18

Block Diagram

In fig 1.18 A 555 timer has two comparators, an R-S flip-flop, two transistors and a resistive network.

- Resistive network consists of three equal resistors and acts as a voltage divider.
- Comparator 1 compares threshold voltage with a reference voltage + 2/3 V_{CC} volts.
- Comparator 2 compares the trigger voltage with a reference voltage + 1/3 V_{CC} volts.

Output of both the comparators is supplied to the flip-flop. Flip-flop assumes its state according to the output of the two comparators. One of the two transistors is a discharge transistor of which collector is connected to **pin** 7. This transistor saturates or cuts-off according to the output state of the flip-flop. The saturated transistor provides a discharge path to a capacitor connected externally. Base of another transistor is connected to a reset terminal. A pulse applied to this terminal resets the whole timer irrespective of any input

Working Principle

1. The internal resistors act as a voltage divider network, providing (2/3)Vcc at the non-inverting terminal of the upper comparator and (1/3)Vcc at the inverting terminal of the lower comparator.
2. Upper comparator has a threshold input (pin 6) and a control input (pin 5).
3. Output of the upper comparator is applied to set (S) input of the flip-flop.
4. Whenever the threshold voltage exceeds the control voltage, the upper comparator will set the flip-flop and its output is high.
5. *A* high output from the flip-flop when given to the base of the discharge transistor saturates it and thus discharges the transistor that is connected externally to the discharge pin 7.
6. The complementary signal out of the flip-flop goes to pin 3, the output. The output available at pin 3 is low. These conditions will prevail until lower comparator triggers the flip-flop.

7. Even if the voltage at the threshold input falls below (2/3) V_{CC}, that is upper comparator cannot cause the flip-flop to change again. It means that the upper comparator can only force the flip-flop's output high.
8. To change the output of flip-flop to low, the voltage at the trigger input must fall below + (1/3) Vcc. When this occurs, lower comparator triggers the flip-flop, forcing its output low.

Different Modes of Operation

Generally, the 555 timer can be operated in three modes: Astable, Monostable (or one-shot).

Astable Mode

In this mode, the 555 work as a **free running mode**. The output of astable multivibrator will continuously **toggle between low and high**, there by generating a train of pulse,

Monostable Mode

In the monostable mode, as the name suggests, it stays in its stable state until and unless an external trigger is applied. In this mode, the 555 functions as a **"one-shot" pulse generato**r.

Features

Some of the important features of the 555 timer are

- The 555 timer can be **operated at a wide range** of power supplies ranging from **5V to 18V**.
- It is available in 3 different packages: 8-pin Metal Can package, 8-pin DIP and 14-pin DIP.
- Timing can be anywhere from microseconds to hours.
- It can operate in both astable and monostable modes.
- High output current.
- It has an adjustable duty cycle.
- It is TTL compatible due to its high output current.
- The output can source or sink a current 0mA to the load.

It has a temperature stability of 0.005% per 0C

1.11.1 Astable Mode

Circuit and Operation

Astable multivibrator is also called as Free Running Multivibrator. It has **no stable states** and continuously switches between the two states without application of any external trigger.

The IC 555 can be made to work as an astable multivibrator **with the addition of three external components: two resistors (R1 and R2) and a capacitor (C).** The schematic of the IC 555 as an astable multivibrator along with the three external components is shown fig 1.19.

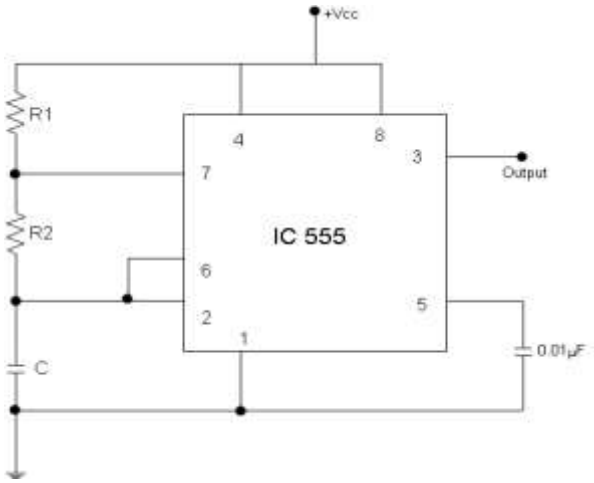


Fig. 1.19

The pins 2 and 6 are connected and hence there is no need for an external trigger pulse. It will self trigger and act as a **free running multivibrator**.

The rest of the connections are as follows: pin 8 is connected to supply voltage (VCC). Pin 3 is the output terminal and hence the output is available at this pin. Pin 4 is the external reset pin. A momentary low on this pin will reset the timer. Hence when not in use, pin 4 is usually tied to VCC

The control voltage applied at pin 5 will change the threshold voltage level. But for normal use, pin 5 is connected to ground via a capacitor (usually 0.01µF).

Operation

The following depicts the fig 1.20 e internal circuit of the IC 555 operating in astable mode. The RC timing circuit incorporates R1, R2 and C.

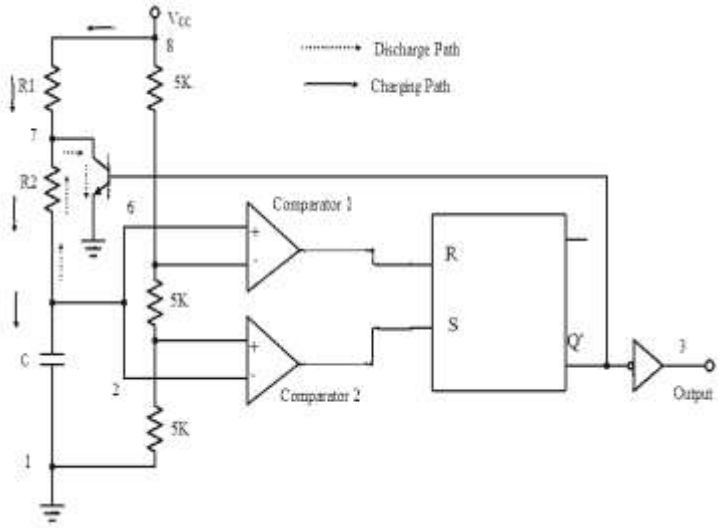


Fig. 1.20

1. Initially, on power-up, the flip-flop is RESET (and hence the output of the timer is low). As a result, the discharge transistor is driven to saturation (as it is connected to Q').
2. The capacitor C of the timing circuit is connected at Pin 7 of the IC 555 and will discharge through the transistor.
3. The output of the timer at this point is low. The voltage across the capacitor is nothing but the trigger voltage.
4. So while discharging, if the capacitor voltage becomes less than 1/3 VCC, which is the reference voltage to trigger comparator (comparator 2), the output of the comparator 2

will become high. This will SET the flip-flop and hence the output of the timer at pin 3 goes to HIGH.

5. This high output will turn OFF the transistor.
6. As a result, the capacitor C starts charging through the resistors R1 and R2. Now, the capacitor voltage is same as the threshold .
7. While charging, the capacitor voltage increases exponentially towards VCC and the moment it crosses 2/3 VCC, which is the reference voltage to threshold comparator (comparator 1), its output becomes high.
8. As a result, the flip-flop is RESET. The output of the timer falls to LOW. This low output will once again turn on the transistor
9. which provides a discharge path to the capacitor. Hence the capacitor C will discharge through the resistor R2. And hence the cycle continues.

Thus, when the capacitor is charging, the voltage across the capacitor rises exponentially and the output voltage at pin 3 is high. Similarly, when the capacitor is discharging, the voltage across the capacitor falls exponentially and the output voltage at pin 3 is low. The shape of the output waveform is a train of rectangular pulses. Shown fig 1.21

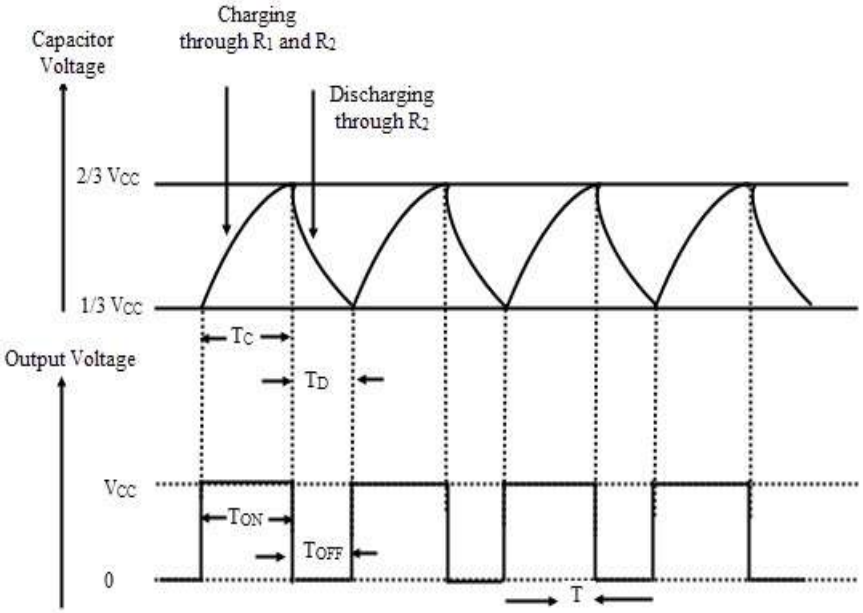


Fig. 1.21

While charging, the capacitor charges through the resistors R1 and R2. Therefore the charging time constant is (R1 + R2) C as the total resistance in the charging path is R1 + R2. While discharging, the capacitor discharges through the resistor R2 only. Hence the discharge time constant is R2C

Duty Cycle

The charging and discharging time constants depends on the values of the resistors R1 and R2. Generally, the charging time constant is more than the discharging time constant. Hence the HIGH output remains longer than the LOW output and therefore the output waveform is not symmetric. Duty cycle is the mathematical parameter that forms a relation between the high output and the low output.

Duty Cycle is defined as the ratio of time of HIGH output i.e. the ON time to the total time of a cycle.

If TON is the time for high output and T is the time period of one cycle, then the duty cycle D is given by

$$D = T_{ON}/ T$$

Therefore, percentage Duty Cycle is given by

$$\%D = (T_{ON} / T) * 100$$

T is sum of T_{ON} (charge time) and T_{OFF} (discharge time).

The value of T_{ON} or the charge time (for high output) T_C is given by

$$T_C = 0.693 * (R1 + R2) C$$

The value of T_{OFF} or the discharge time (for low output) T_D is given by

$$T_D = 0.693 * R2C$$

Therefore, the time period for one cycle T is given by

$$T = T_{ON} + T_{OFF} = T_C + T_D$$

$$T = 0.693 * (R1 + R2) C + 0.693 * R2C$$

$$T = 0.693 * (R1 + 2R2) C$$

Therefore, $\%D = (T_{ON}/ T) * 100$

$$\%D = (0.693 * (R1 + R2) C)/(0.693 * (R1 + 2R2) C) * 100$$

$$\%D = ((R1 + 2R2))/((R1 + 2R2)) * 100$$

If T = 0.693 * (R1 + R2) C, then the frequency f is given by

$$f = 1 / T = 1 / 0.693 * (R1 + 2R2) C$$

$$\mathbf{f = 1.44/((R1 + 2R2) C) Hz}$$

1.11.2 Circuit of 555 timer as Schmitt Trigger

The following circuit shows fig 1.22 the structure of a 555 timer used as a Schmitt trigger.

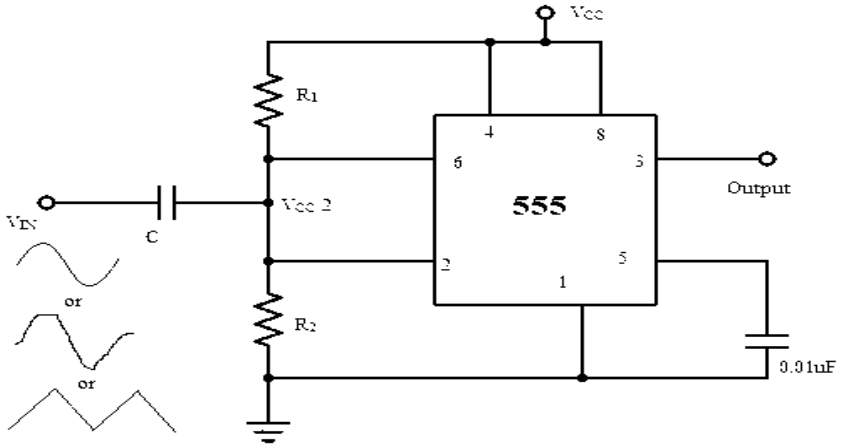


Fig. 1.22

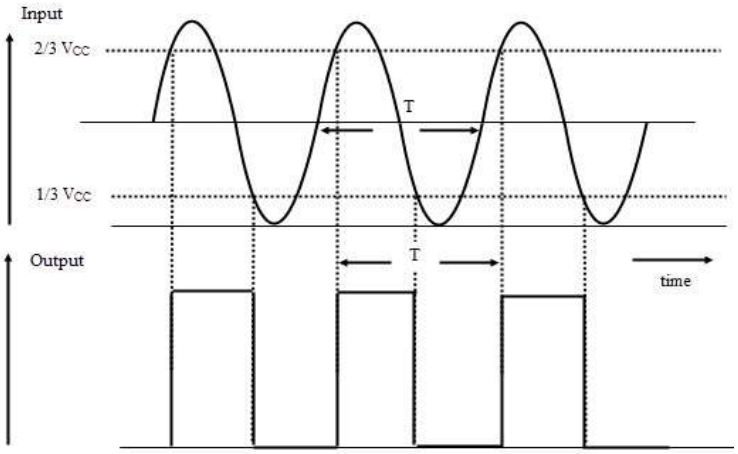


Fig 1.23

Pins 4 and 8 are connected to the supply (VCC). The pins 2 and 6 are tied together and the input is given to this common point through a capacitor C. this common point is supplied with an external bias voltage of VCC / 2 with the help of the voltage divider circuit formed by the resistors R1 and R2.

The important characteristic of the Schmitt trigger is Hysteresis. The output of the Schmitt trigger is high if the input voltage is greater than the upper threshold value and the output of the Schmitt trigger is low if the input voltage is lower than the lower threshold value shown fig 1.23

The output retains its value when the input is between the two threshold values. The usage of two threshold values is called Hysteresis and the Schmitt trigger acts as a memory element (a bistable multivibrator or a flip-flop).

The threshold values in this case are 2/3 VCC and 1/3 VCC i.e. the upper comparator trips at 2/3 VCC and the lower comparator trips at 1/3 VCC. The input voltage is compared to these threshold values by the individual comparators and the flip-flop is SET or RESET accordingly. Based on this the output becomes high or low.

When a sine wave of amplitude greater than VCC / 6 is applied at the input, the flip-flop is set and reset alternately for the positive cycle and the negative cycle. The output is a square wave and the waveforms for input sine wave and output square wave are shown above.

1.11.3 Sequential timer

This is a simple sequential timer circuit using NE555 shown fig 1.24 which can be used for psychedelic light, decoration in shops, advertisement boards at night, and in parties.
Circuit Description Sequential Timer Using NE555

In the circuit sequential timer circuit IC2, IC3 and IC4 are used as monostable multivibrators. The trigger terminals (pins 2) of these ICs are connected in chain to the previous ICs through RC differentiator network (0.022 μF capacitor and 22 KΩ resistor). IC1 is used as an astable multivibrator with adjustable duty cycle and time period.

To understand the functioning of the sequential timer circuit, supposed the circuit is put in 'all rotate' position and lamp L1 is on. When lamp L1 goes to off state, as pin 3 of IC2 goes to the low state, it triggers IC3 through C9 and R9 differentiator, and thus lamp L2 switches on. In the next period, lamp L2 while going off switches on lamp L3 and so on, thus completing the chain.

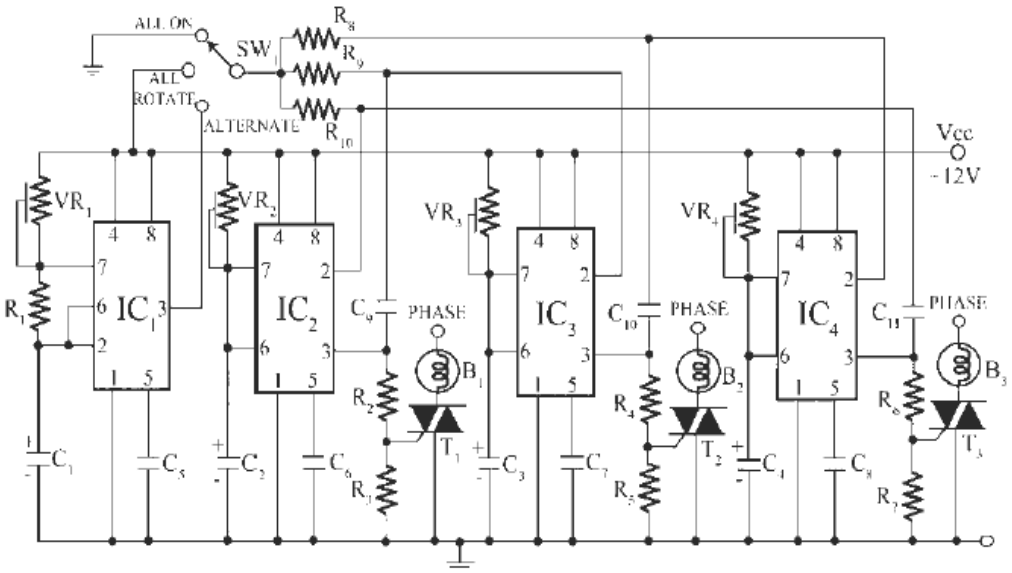


Fig 1.24

When the switch SW_1 is in 'all on' state, trigger terminal of each IC_2, IC_3 and IC_4 is clamped to the ground and thus output of each goes high, making each of the lamp L_1, L_2 and L_3 to switch on. This state is useful in checking for the fused lamps.

Since IC_1 is put as astablemultivibrator, when switch $SW_1$is put to alternate position, for some time the lamps rotate and then for some time all of them glow simultaneously. For the given component values, the time period can be increased to about four seconds. This time period can be adjusted by VR_1.The effect on 'alternate' position comes from the facts that in this case trigger terminals of IC_2, IC_3 and IC_4 are put to 'all on' and 'all rotate' positions automatically alternately, because output of IC_1 goes to Vcc and ground level alternately.Adjustment of the unit is simple. Put SW_1 to all rotate position and adjust VR_2, VR_3 and VR_4 so that each lamp glows for equal time, or the effect is most eye appeasing. Then put the switch to alternate position and adjust VR_1 for most pleasing effect.

If many small lamps are to be used, a suitable sequence for connection of the lamps can be L_1 L_2 L_3 L_1 L_2 L_3——.

Since the monistable IC_2, IC_3 and IC_4 require an initial trigger to being cycling, the circuit may not operates if switch SW_1 is in 'all rotate' position initially. In this case, switch SW_1 to the 'all on' or 'alternate' position momentarily before switching back to the 'all rotate' position.

When the switch SW1 is in 'all on' state, trigger terminal of each IC2, IC3 and IC4 is clamped to the ground and thus output of each goes high, making each of the lamp L1, L2 and L3 to switch on. This state is useful in checking for the fused lamps.

Since IC1 is put as astablemultivibrator, when switch SW1is put to alternate position, for some time the lamps rotate and then for some time all of them glow simultaneously. For the given component values, the time period can be increased to about four seconds. This time period can be adjusted by VR1.

The effect on 'alternate' position comes from the facts that in this case trigger terminals of IC2, IC3 and IC4 are put to 'all on' and 'all rotate' positions automatically alternately, because output of IC1 goes to Vcc and ground level alternately.

Adjustment of the unit is simple. Put SW1 to all rotate position and adjust VR2, VR3 and VR4 so that each lamp glows for equal time, or the effect is most eye appeasing. Then put the switch to alternate position and adjust VR1 for most pleasing effect.

If many small lamps are to be used, a suitable sequence for connection of the lamps can be L1 L2 L3 L1 L2 L3——.

Since the monistable IC2, IC3 and IC4 require an initial trigger to being cycling, the circuit may not operates if switch SW1 is in 'all rotate' position initially. In this case, switch SW1 to the 'all on' or 'alternate' position momentarily can be used as PWM before switching back to the 'all rotate' position.

555 timer can be used as PWM In analog or digital communication system, it is impossible to transmit a low frequency message signal over a channel, it causes signal distortion, attenuation or signal loss. Inorder to make the transmission ideal (almost), we have to use some techniques called modulation.

Modulation can be defined as the process of varying some characteristics of carrier signal in accordance with the instantaneous value of message signal. **Carrier signal** is a high frequency wave generated by a local oscillator, used to carry the message signal from transmitter to reciever. The message signal which is mixed with carrier wave is called modulated signal and the process is called Modulation. The modulation is an important process in a basic communication system, which increases the transmission range, reduces the size of antenna and increases the efficiency of the system by reducing noise interference.

Pulse width modulation is a process of varying the width of carrier signal according to the instantaneous amplitude value of message signal. PWM is also known as **Pulse Duration Modulation (PDM) or Pulse Length Modulation (PLM)**. It utilises the advantages of constant amplitude pulses.

CIRCUIT DIAGRAM OF PWM USING 555 TIMER

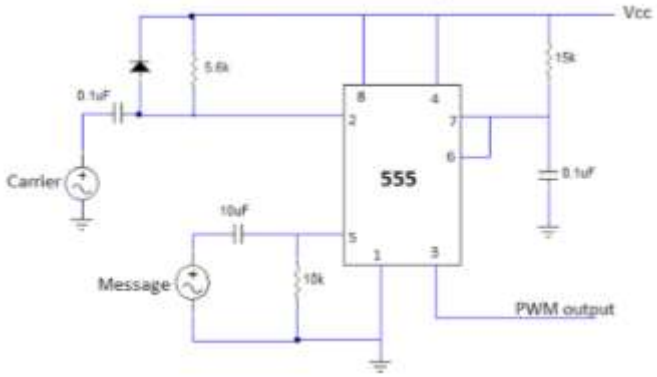
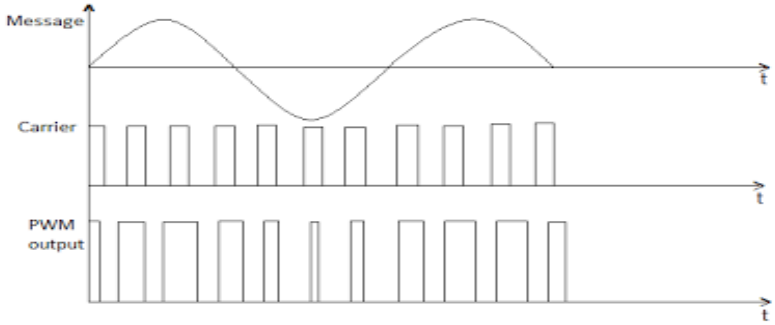


Fig 1.25

INPUT AND OUTPUT WAVEFORMS OF PWM



In the circuit diagram shown fig 1.25 the 555 IC is wired as monostable multivibrator, which varies the output period of oscillations according to the amplitude of message signal. That is, Trigger is applied to control the starting time of pulses and modulating signal is fed to control the duration of pulses.

Applications of Monostable Multivibrator

Frequency Divider

Pulse Width Modulation

Linear Ramp Generator

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

REVIEW QUESTIONS

Two Mark

1	What is an Operational Amplifier?
2	What are the advantages of Operational Amplifier?
3	Define CMMR.
4	Define slew rate.
5	Draw the symbol of 741.
6	What is gain?
7	Write the output equation of inverting amplifier.
8	Write the output equation of Non inverting amplifier.
9	What is gain of voltage follower?
10	What is use of voltage follower?
11	What is the output of summing amplifier if V1 and V2 are inputs.?
12	What are the input and feedback components of Integrator?
13	What are the input and feedback components of Differentiator?
14	What is astable mode in 555 time?
15	What is mono stable mode in 555 time?
16	What is the output of Schmitt trigger?
17	What is PWM?

Three Mark

1	What are the characteristics of Operational Amplifier?
2	Draw the block diagram of ideal Operational Amplifier
3	Write the parameters of Operational Amplifier
4	What is virtual ground?
5	Draw the pin diagram Operational Amplifier 741
6	What are the applications of Operational Amplifier?
7	Draw the circuit diagram of inverting amplifier.
8	Draw the circuit diagram of voltage follower.
9	How comparator works?
10	How zero crossing detector works?
11	Draw the input and output waveforms of Integrator.
12	Draw the input and output waveforms of Differentiator.

10 Mark

1	Explain inverting amplifier with circuit diagram and derive its output.
2	Explain Non inverting amplifier with circuit diagram and derive its output.
3	Explain summing amplifier with circuit diagram and derive its output.
4	Explain zero crossing detector with circuit diagram , input and output waveforms.
5	Explain Integrator with circuit diagram and derive its output.
6	Explain Differentiator with circuit diagram and derive its output.
7	Explain a stable mode in 555 timer with circuit diagram and derive its output
8	Explain mono stable mode in 555 timer with circuit diagram and derive its output
9	Explain Schmitt trgger in 555 timer with circuit diagram and derive its output

UNIT – II

BOOLEAN ALGEBRA

2.1 Numbering System

The study of *number systems* is important from the viewpoint of understanding how data are represented before they can be processed by any digital system including a digital computer. It is one of the most basic topics in digital electronics. In this chapter we will discuss different number systems commonly used to represent data. We will begin the discussion with the decimal number system. Although it is not important from the viewpoint of digital electronics, a brief outline of this will be given to explain some of the underlying concepts used in other number systems. This will then be followed by the more commonly used number systems such as the binary, octal and hexadecimal number systems.

The table 2.1 given below shows some examples of number system which are often used in digital circuits.

Table 2.1 examples of number system.

S.No	Types	Base/Radix	Numbers
1	Decimal number system	10	0,1,2,….9
2	Binary number system	2	0,1
3	Octal number system	8	0,1,2,3…7
4	Hexa decimal number system	16	0,1,2,….9, A,B,C,D,E,F

Table 2.1

Decimal Number System

The decimal system is composed of 10 numerals or symbols. These 10 symbols are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Using these symbols as digits of a number, we can express any quantity. The decimal system is also called the base-10 system because it has 10 digits

Weight of each digit	**10^3**	**10^2**	**10^1**	**10^0**		**10^{-1}**	**10^{-2}**	**10^{-3}**
Digit value	=1000	=100	=10	=1	•	=0.1	=0.01	0.001
Bit position	Most Significant Digit				Decimal point			Least Significant Digit

Binary Number System

In the binary system, there are only two symbols or possible digit values, 0 and 1. This base-2 system can be used to represent any quantity that can be represented in decimal or other base system

Weight of each digit	**2^3**	**2^2**	**2^1**	**2^0**		**2^{-1}**	**2^{-2}**	**2^{-3}**
Digit value	=8	=4	=2	=1	•	=0.5	=0.25	=0.125
Bit position	Most Significant Digit				Binary point			Least Significant Digit

Octal Number System

The octal number system has a base of eight, meaning that it has eight possible digits: 0,1,2,3,4,5,6,7

Weight of each digit	**8^3**	**8^2**	**8^1**	**8^0**		**8^{-1}**	**8^{-2}**	**8^{-3}**
Digit value	=512	=64	=8	=1	•	=1/8	=1/64	=1/512
Bit position	Most Significant Digit				Octal point			Least Significant Digit

Hexadecimal Number System

The hexadecimal system uses base 16. Thus, it has 16 possible digit symbols. It uses the digits 0 through 9 plus the letters A, B, C, D, E, and F as the 16 digit symbols

Weight of each digit	**16^3**	**16^2**	**16^1**	**16^0**		**16^{-1}**	**16^{-2}**	**16^{-3}**
Digit value	=4096	=256	=16	=1	•	=1/16	=1/256	=1/4096
Bit position	Most Significant Digit				Hexa Decimal point			Least Significant Digit

BCD Number System

BCD stands for binary coded decimal. A nibble is a string of four bits. BCD numbers express each decimal digit as nibble. It is a decimal number represented in binary form with 0 and 1. In BCD each decimal digit is represented by its four bit binary number. The lowest number is 0000(0) and the highest number is 1001((9)

2.2 Conversion from one number system to other

Converting from one code form to another code form is called code conversion, like converting from binary to decimal or converting from hexadecimal to decimal.

Any binary number can be converted to its decimal equivalent simply by summing together the weights of the various positions in the binary number which contain a 1.

Binary to Decimal Conversion (Integer)

Example 1:

Binary	**Decimal**
11011_2	
$(1x2^4)+(1x2^3)+(0x2^2)+(1x2^1)+1x2^0$	=16+8+0+2+1
Result	27_{10}

Example2:

Binary	**Decimal**
10110101_2	
$(1x2^7)+(0x2^6)+(1x2^5)+(1x2^4)+(0x2^3)+(1x2^2)+(0x2^1)+1x2^0$	=128+0+32+16+0+4+0+1
Result	181_{10}

Binary-To-Decimal (fraction)
Example 3:

Binary	**Decimal**
0.1101	
$(1x2^{-1})+(1x2^{-2})+(0x2^{-3})+1x2^{-4}$	=1x0.5=1x0.25+0x0.125+1x0.0625
Result	0.8125_{10}

Decimal-To-Binary Conversion (Integer)
There are 2 methods:

- **Reverse of Binary-To-Decimal Method**
- **Repeat Division**

Reverse of Binary-To-Decimal Method

Decimal-To-Binary Conversion (fraction)

In this case the given decimal number is multiplied by 2 and then records the carry in the integer position. The carry are taken in downward direction(from top to bottom) to obtain the required binary fraction. This process can be stopped either after getting the result as zero or after getting six binary digits.

Convert 0.85_{10} to binary

Multiplication	**Remainder**	**Binary**
0.85x2	= 1.7=.7 with the carry of 1	**1**
0.7x2	= 1.4=.4with the carry of 1	**1**
0.4x2	= 0.8=.8 with the carry of 0	**0**
0.8x2	= 1.6=.6 with the carry of 1	**1**
0.6x2	= 1.2=.2 with the carry of 1	**1**
0.2x2	= 0.4=.4 with the carry of 1	1
Result	0.85_{10}	$= 0.110110_2$

Decimal to octal conversion(Integer)
This method uses repeated division by 8.and writing down the remainder after each division. The remainders are taken in the reverse order(from bottom to top), to form the octal number
Example1:
Convert $177_{10}$to octal

Division	**Remainder**	**Octal**
177/8	= 22+ remainder of 1	1 (Least Significant Bit)
22/ 8	= 2 + remainder of 6	6
2 / 8	= 0 + remainder of 2	2 (Most Significant Bit)
Result	177_{10}	$= 261_8$

Decimal to octal conversion(Fraction)
In this case the given decimal number is multiplied by 8 and then records the carry in the integer position. The carry are taken in downward direction (from top to bottom) to obtain the required octal number. This process can be stopped either after getting the result as zero or after getting six digits.

Convert 0.82_{10} to binary

Multiplication	Remainder	Binary
0.82x8	= 6.56=.56 with the carry of 6	**6**
0.56x8	= 4.48=.48with the carry of 4	**4**
0.48x8	= 3.84=.84 with the carry of 3	**3**
0.84x8	= 6.72=.72with the carry of 6	**6**
0.72x8	= 5.76=.76 with the carry of 5	**5**
0.76x8	= 6.08 =.08with the carry of 6	6
Result	0.85_{10}	= 0.643656_8

Octal to Decimal conversion(Integer) Any octal number can be converted to its decimal equivalent simply by summing together the weights of the various positions in the octal number

octal	Decimal
175_8	
$1x8^2+7x8^1+5x8^0$	64+56+5
Result	125_{10}

Octal to Decimal conversion(fraction)

Decimal to Hexa- decimal conversion

Binary	Decimal
0.45_8	
$4x8^{-1}+5x8^{-2}$	=0.5+0.078125
Result	0.578125_{10}

This method uses repeated division by 16.and writing down the remainder after each division. The remainders are taken in the reverse order (from bottom to top), to form the hexa- decimal number

Example1: Convert $378_{10}$to hexa-decimal

Division	Remainder	Hexa-decimal
378/16	= 23+ remainder of 10	A (Least Significant Bit)23
23/16	= 1 + remainder of 7	7
1/16	= 0 + remainder of 1	1 (Most Significant Bit)
Result	378_{10}	= $17A_{16}$

Decimal to hexa-decimal conversion(Fraction) In this case the given decimal number is multiplied by 16 and then records the carry in the integer position. The carry are taken in downward direction (from top to bottom) to obtain the required octal number. This process can be stopped either after getting the result as zero or after getting six digits.

Convert 0.935_{10} to hexa-decimal

Multiplication	Remainder	Hexa-decimal
0.935x16	= 14.96=.96 with the carry of 14(=E)	**E**
0.96x16	= 15.36=.36with the carry of 15(=F)	**F**
0.36x16	= 5.76=.76 with the carry of 5	**5**
0.76x16	= 12.16=..16with the carry of 12(=C)	**C**
0.16x16	= 2.56=.56 with the carry of2	**2**
0.56x16	= 8.96 =..96with the carry of 8	8
Result	0.935_{10}	= $0.EF5C28_{16}$

Hexa-decimal to decimal conversion Any Hexa-decimal number can be converted to its decimal equivalent simply by summing together the weights of the various positions in the hexa-decimal number

octal	**Decimal**
$2AF_{16}$	
$2 \times (16^2) + 10 \times (16^1) + 15 \times (16^0)$	512+160+15
Result	687_{10}

Hexa-decimal to Decimal conversion(fraction)

Binary	**Decimal**
$0.B5_{16}$	
$Bx16^{-1}+5x16^{-2} = 11x16^{-1}+5x16^{-2}$	=0.6875+0.0195312
Result	0.70703112_{10}

Octal to binary conversion

In octal to binary conversion , each octal digit is converted into its equivalent three digit binary form The octal number and its equivalent three digit binary numbers are shown in table 2.2

Octal number	**Equivalent binary number**
0	**000**
1	**001**
2	**010**
3	**011**
4	**100**
5	**101**
6	**110**
7	**111**

Table 2.2

Convert an octal number326.54 to its equivalent binary number

octal	3	2	6	•	5	4
Binary	011	010	110	•	101	100

Octal number	**Equivalent binary number**
0	**000**
1	**001**
2	**010**
3	**011**
4	**100**
5	**101**
6	**110**
7	**111**

Binary to Octal Conversion

As the binary numbers are comprised of only 0 and 1 in this method, first divide the binary number into group of 3 digits. Suppose the binary numbers are not completed in the form of three digits, sufficient zeros are added to the left most side of the integer part and also

sufficient zeros are added to the right side of fractional part. That means the integer number is arranged in group of three digit from right to left. Conversely, the fraction number is arranged in group of three digit from left to right.

Example1: Let us take a binary number say 1101001.11010_2

Binary	**001**	101	001	•	110	100
octal	1	5	1	.•	6	4

Convert Hexa-decimal numberA26.F4 to its equivalent binary number

Hexa-decimal	A	2	6	•	F	4
Binary	1010	0010	0110	•	1111	0100

Binary to Hexa -decimal Conversion

As the binary numbers are comprised of only 0 and 1 in this method, first divide the binary number into group of 4 digits. Suppose the binary numbers are not completed in the form of four digits, sufficient zeros are added to the left most side of the integer part and also sufficient zeros are added to the right side of fractional part. That means the integer number is arranged in group of four digit from right to left. Conversely, the fraction number is arranged in group of four digit from left to right.

Regroup the binary number by four bits per group starting from LSB if Hexadecimal is required.

Example1:

Let us take a binary number say 1101101001.1101011_2

Binary	**0011**	0110	1001	•	1101	0110
Hexa-decimal	3	6	9	•	D	6

Decimal to BCD conversion

In this method each decimal digit is converted into its equivalent 4 digit binary form (BCD)

Convert a decimal number 881 to its equivalent BCD number

Decimal	**8**	8	1`
BCD	1000	1000	0001

BCD to Decimal conversion

In this method each BCD number in the form of four digit binary pattern is converted into its equivalent decimal number.

BCD	**1001**	1000	0001
Decimal	9	8	1

2.3 Boolean Algebra

Boolean algebra was invented by world famous mathematician George Boole in the year of 1854. Boolean algebra or switching algebra is a system of mathematical logic to perform different mathematical operations in binary system. These are only two elements 1 and 0 by which all the mathematical operations are to be performed. There only three basis binary operations, AND, OR and NOT by which all simple as well as complex binary

mathematical operations are to be done. Some basic logical Boolean operations- AND operation, OR operation, NOT operation,

2.4 Basics Laws And Demorgan's Theorems

2.4.1 Basic laws of Boolean Algebra

1. **OR law**
   A + 0 = A where A can be either 0 or 1.
   A + 1 = 1 where A can be either 0 or 1.
   A + $\bar{A}$ = 1 where A can be either 0 or 1.
   A + A = A where A can be either 0 or 1.
2. AND law
   A . 0 = 0 where A can be either 0 or 1.
   A . 1 = A where A can be either 0 or 1
   A . A = A where A can be either 0 or 1.
   A . $\bar{A}$ = 0 where A can be either 0 or 1.
3. **Laws of Complementation**

   (A')'=A
4. **Commutative law**

   A+B=B+A

   A.B=B.A

Here the answer of adding A+B is same as that of B+A. Similarly A.B is same as that of B.A

5. **Associative law**

   A+(B+C)=(A+B)+C

   A.(B.C)=(A.B).C

6. **Distributive law**

   A+BC=(A+B)(A+C)

   A.(B+C)=A.B+A.C

7. **Some of the following theorems are solved by the basic laws are**

   1.A+AB=A

   2.A.(A+B)=A

   3.A+A'B=A+B

   4.(A+B)(A+B')=A

   5. AB+AB'=A

2.4.2 Demorgan's Theorems

i) $\overline{\mathbf{A+B}} = \bar{\mathbf{A}} \cdot \bar{\mathbf{B}}$ Demorgan's first theorem states that the complement of a sum equals the product of the complements.

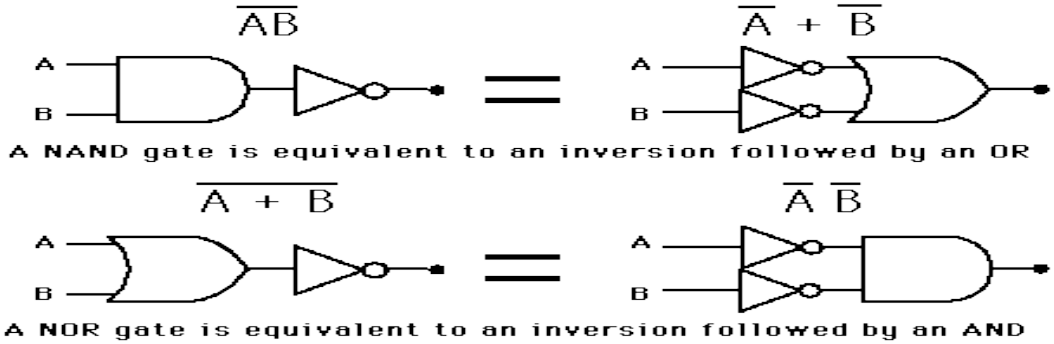


Fig 2.1

These equalities can be easily proved as shown in truth tables.

INPUTS		LHS		RHS		
A	B	A+B	$\overline{(A+B)}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

Table 2.4

(ii) $\overline{A}\ \overline{B} = \overline{A} + \overline{B}$

Demorgan's second theorem states that the complement of a product equals the sum of the complement.

INPUTS		LHS		RHS		
A	B	A.B	$\overline{AB}$	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

Table 2.5

2.5 Logic Gates

The basic elements thatmakes up digital system are called as logic gates. The most common logic gates are OR, AND, Not, NAND and NOR gates. OR, AND ,NOT are called fundamental gates. NAND,NOR gates are called as universal gates. Exclusive –OR gate is another logic circuit which can be constructed using and,OR and NOT gate

Logical Addition (OR Gate)

If A and B are the input logic variable and Y is the output Variable, the truth table for a two input OR gate is given in Figure.2.2

(a) Truth table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

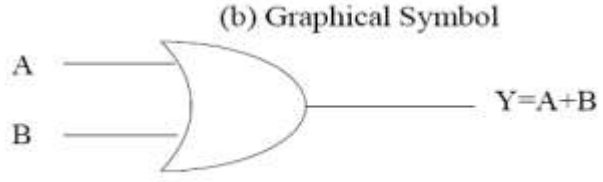
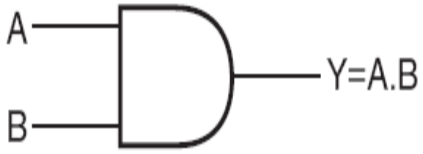


Fig 2.2

The OR gate produces the inclusive – OR function, that is, the output is 1. If input A or input B or both inputs are 1, otherwise the output is a 0.
Y=A+B+C+D+E+……

Logical Multiplication (AND Gate)
The AND gate has one or more inputs and a single output. The output of an AND gate is equal to the multiplication of its inputs. The truth table for a two input AND gate is shown in Figure. . The output is 1 if both A and B are 1 and 0 otherwise. In general, in an n input AND gate, only if all the inputs are at logic 1, the output will be 1. That is Y = A,B
The Input – Output relationships can be written as



A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Fig 2.3

Logical Inversion: Complement (NOT gate)

The complement function is nothing but inversion, 0 is changed to 1 and 1 to 0. The inverter circuit is also referred to as a NOT gate and it has a single input and single output. The truth table for a NOT gate is shown 2.4

(a) Truth Table

A	Y
0	1
1	0

(b) Graphic Symbol

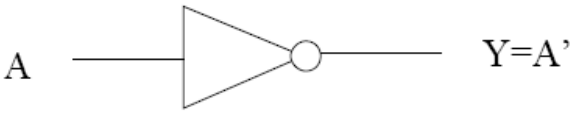


Fig 2.4

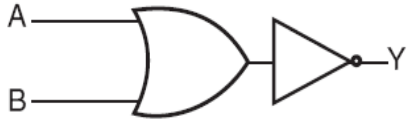
The NOT logic can be written as, Y=A' or Y=A. The small circle in the output of the graphic symbol of an inverter designates a logic complement.

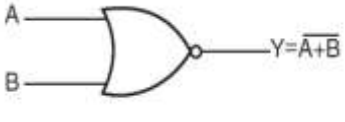
Secondary Gates

NOR Gate

The NOR function is the complement of OR function and uses the OR graphic symbol followed by a small circle.

A NOR gate is shown in Figure 2.5 .Figure shows the NOR gate block diagram symbol with inputs A, B, and the output $\overline{A+B}$. This shows the NOR gate's output will be a 1 only when all the inputs are 0's. If any input represents a 1, then the output of a NOR gate will be a 0.





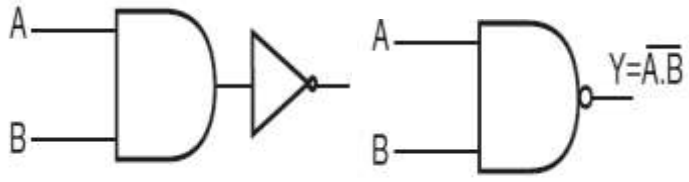
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Fig 2.5

NAND Gate

The NAND function is the complement of the AND function, as indicated by a graphic symbol that consists of an AND graphic symbol followed by a small circle.

If the inputs are A, B then the output of the AND gate will be A.B and the complement of this is (A.B.) $\overline{A.B}$ as shown in the figure . The truth table for a NAND gate is shown in fig 2.6.



A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Fig 2.6

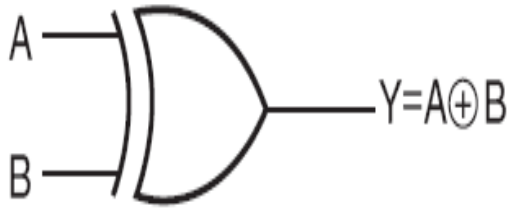
EX-OR Gate

The exclusive-OR gate has a graphical symbol similar to that of the OR gate, except for the additional curved line on the input side. The equivalence, or exclusive-NOR, gate is the complement of the exclusive-OR, as indicated by the small circle on the output side of the graphic symbol.

The exclusive – OR (XOR), denoted by $\oplus$, is a logical operation that performs the

Following Boolean operation**:** $Y = (A \oplus B) = \overline{A}B + A\overline{B}$

It is equal to 1 if only A is equal to 1 or if only y is equal to 1, but not when both are equal to 1, The truth table for a XOR gate is shown in the Figure.2.7 and fig 2.8



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Fig 2.7

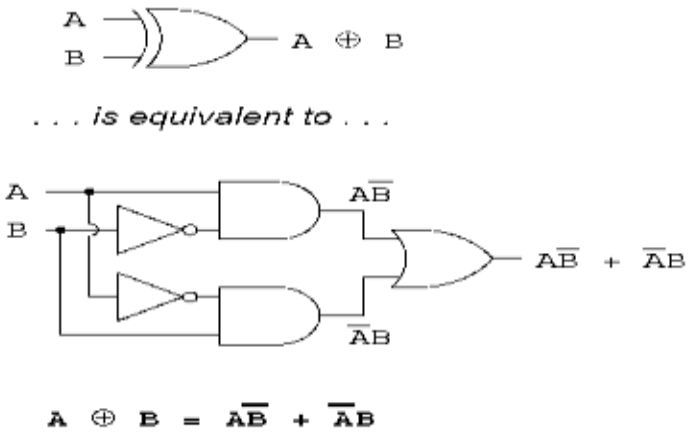


Fig 2.8

2.6 Realization of gates using universal gates NAND and NOR

OR, AND and NOT gates are the three basic logic gates as they together can be used to construct the logic circuit for any given Boolean expression. That is, it is possible to use either only NAND gates or only NOR gates to implement any Boolean expression. **This is so because a combination of NAND gates or a combination of NOR gates can be used to perform functions of any of the basic logic gates. It is for this reason that NAND and NOR gates are universal gates**.

NOT gate using NAND gate

NOT gate is constructed using NAND gate by connecting two inputs together as shown in figure 2.9

Input	Output
0	1
1	0

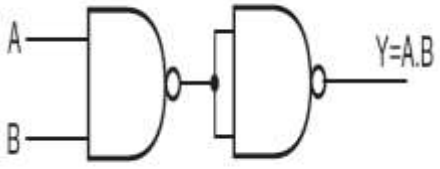
Fig 2.9

In NOT gate, the output Y= $\bar{A}$

=($\bar{A}.\bar{A}$) (i.e $\bar{A}=\bar{A}.\bar{A}$)

AND gate using NAND gates

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

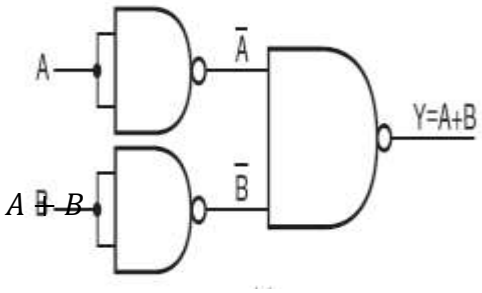


$$Y = \overline{\overline{AB}}$$

$$= AB$$

Fig 2.10

OR gate using NAND gates



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

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

Fig 2.11

NOT gate using NOR gate

NOT gate is constructed using NAND gate by connecting two inputs together as shown in figure 2.12

Input	Output
0	1
1	0

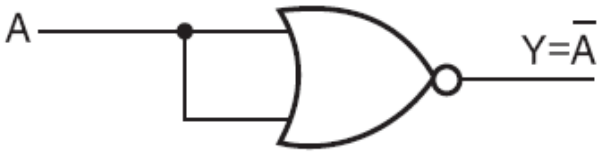


Fig 2.12

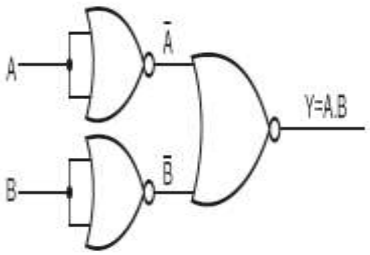
In NOT gate, the output Y=$\overline{A}$

$$=(\bar{A}.\bar{A}) \quad (\text{i.e } \bar{A}=\bar{A}.\bar{A})$$

AND gate using NOR gates

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

$A.B$

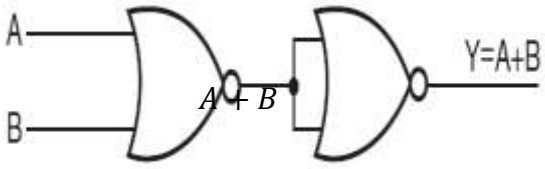


$$Y = A.B$$
$$Y = \overline{(\bar{A} + \overline{B})} = \bar{\bar{A}}.\bar{\bar{B}} =$$

Fig 2.13

OR gate using NOR gates

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1



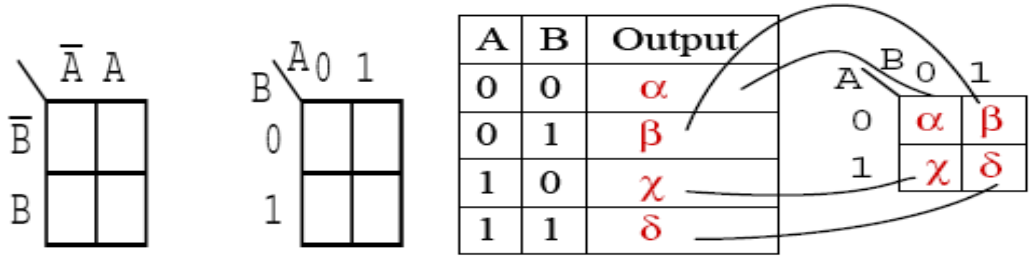
$$Y = \overline{\overline{A + B}} =$$

Fig 2.14

2.7 Problems using 2,3 and 4 variables

A karnaugh map is a visual display of the fundamental products needed for a sum of products solution. The karnaugh map method provides simple and straight forward procedure to minimize Boolean expressions. in sum of product form. The karnaugh map(k-map) for two, three,four variables are different . The k-map for different variables are discussed below. The map is a diagram made up of squares. Each square represents a **minterm**. **Karnaugh maps reduce logic functions more quickly and easily compared to Boolean algebra.**

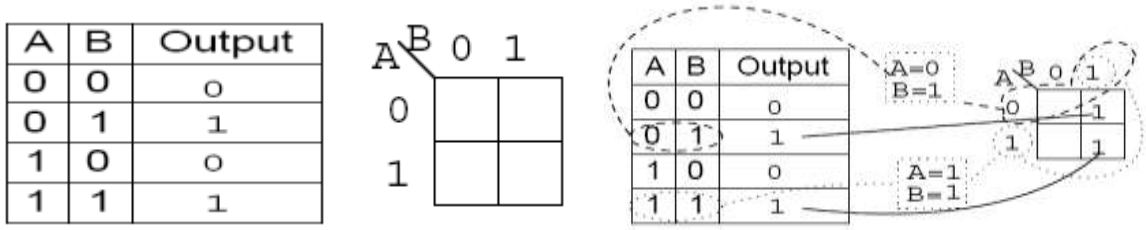
Two –variables map Two inputs A and B can take on values of either 0 or 1, high or low, open or closed, True or False, as the case may be. There are **2power2= 4** combinations of inputs producing an output. These four outputs may be recorded in the truth table, or in the Karnaugh map. Look at the Karnaugh map as being a rearranged truth table. The Output of the Boolean equation may be computed by the laws of Boolean algebra and transferred to the truth table or Karnaugh map. If A and B are inputs, then we make a K map as follows



A	B	Output
0	0	α
0	1	β
1	0	χ
1	1	δ

Table 2.6

Example:
Transfer the contents of the truth table to the Karnaugh map



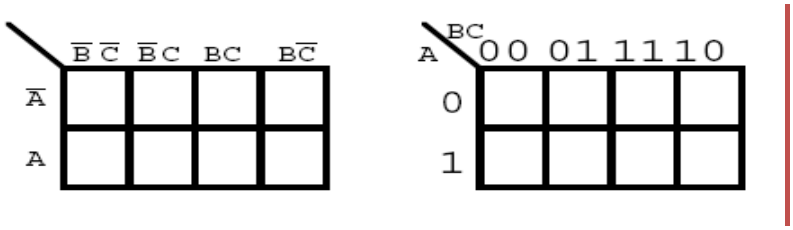
A	B	Output
0	0	0
0	1	1
1	0	0
1	1	1

Table 2.7

In the k-map, first note the variables and their complements The vertical column has A followed by A' and horizontal row has B followed by B'. Then take the combination of inputs, which produce the output as 1. For given truth table output are 1 at A'B and AB input combinations. Enter 1 in the spaces of A'B and AB at the k-map because the corresponding output are high. The remaining spaces are entered with 0s.

Three variable k-map An example of a three variable truth table and their corresponding k-map are shown in fig The horizontal row are labeled as B'C', B'C, BC, BC'. This order is not a binary progress of 00, 01,10 and 11. In the k-map the variables are assigned in a sequence of only one variable changes from complemented to uncomplemented form (or vice versa). The vertical column is marked as A' and A.

For 3 variables: 2 power 3 = 8



Example :

Input			Output
A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

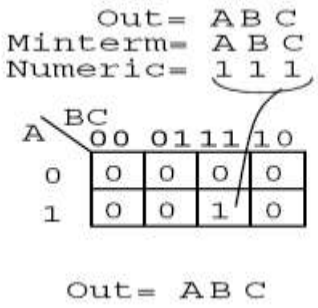


Table 2.7

The fundamental product for ABC is 1. Enter 1 on this variable position of k-map. The remaining spaces are filled with 0s.

Four variable k-map

For 4 variables: 2 power 4 = 16

An example of a four variable truth table and their corresponding k-map are shown in fig The horizontal row are labeled as C'D', C'D, CD, CD'. This order is not a binary progress of 00, 01,10 and 11. In the k-map the variables are assigned in a sequence of only one variable changes from complemented to uncomplemented form (or vice versa). The vertical column is marked as A'B', A'B, AB, AB'..

Input				Output
A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

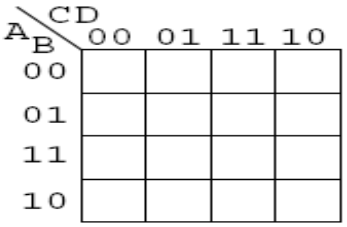


Table 2.8

The output 1 is appeared at the input combinations of ABCD. Enter 1 on this variable position of k-map. The remaining spaces are filled with 0s.

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

Table 2.9

Looping

The expression for the output that can be simplified properly by combining 1s in the k-map is called looping Pairs, quads and octets are the group of looping used in k-map for simplification.

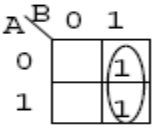
Rules

1. **Two adjacent cells have 1s, form pair.**
2. **Four adjacent cells have 1s, form Square.(QUADS)**
3. **Four corner cells have 1s, form square. .(QUADS)**
4. **Eight adjacent cells have 1s form OCTETS.**
5. **Two corner cells have 1s, form pair.**
6. **Otherwise takes single cell.**

a)pairs

Example:

For the Karnaugh map in the above problem, write the Boolean expression.



Out = B

Solution:

1. Look for adjacent cells, that is, above or to the side of a cell. Diagonal cells are not adjacent. Adjacent cells will have one or more Boolean variables in common.
2. Group (circle) the two 1s in the column
3. Find the variable(s) top and/or side which are the same for the group, Write this as the result. It is B in our case.
4. Ignore variable(s) which are not the same for a cell group. In our case A varies, is both 1
5. And ignore Boolean A.
6. ignore any variable not associated with cells containing 1s. B' has no ones under it.ignore B'
7. Result Out = B

Example:

For the Truth table 2.10 below, transfer the outputs to the Karnaugh, then write the Boolean expression for the result.

A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

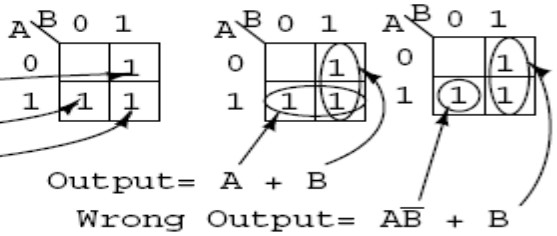
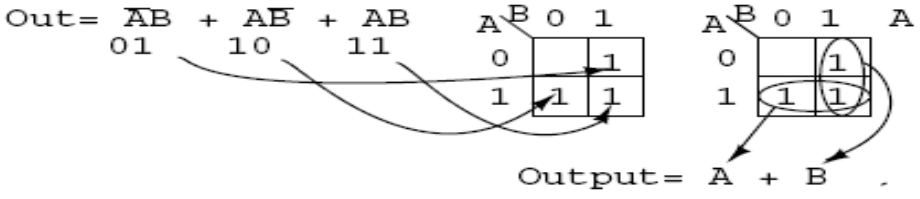


Table 2.10

Solution:

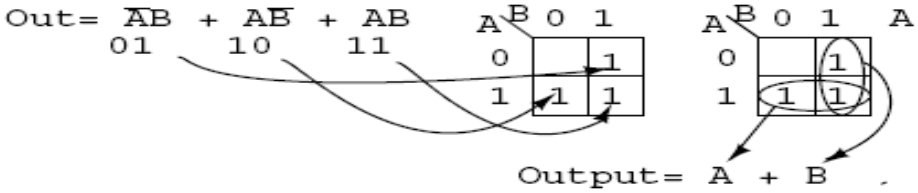
1 Transfer the 1s from the locations in the Truth table to the corresponding locations in the K-map.
2 Group (circle) the two 1's in the column under B=1
3 Group (circle) the two 1's in the row right of A=1
4 Write product term for ¯first group = B
5 Write product term for second group = A
6 Write Sum-Of-Products of above two terms Output = A+B

Example: Fill in the Karnaugh map for the Boolean expression below, then write the Boolean expression
for the result.



Example: Fill in the Karnaugh map for the given function below, then write the Boolean expression for the result.

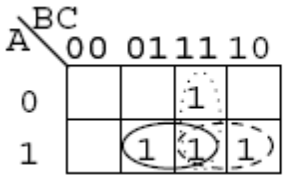
F (A,B)= { 1,2,3}



Example : Simplify the following Boolean algebra.

$\bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$

Solution:



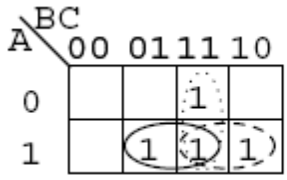
Output = AB + BC + AC

Example : Simplify the following Boolean function

F(A,B,C) = { 3,5,,6,7}

Solution

1 Transfer the product terms to the Karnaugh map
2 Form groups of cells as in previous examples
3 Write Boolean expression for groups as in previous examples
4 Draw simplified logic diagram



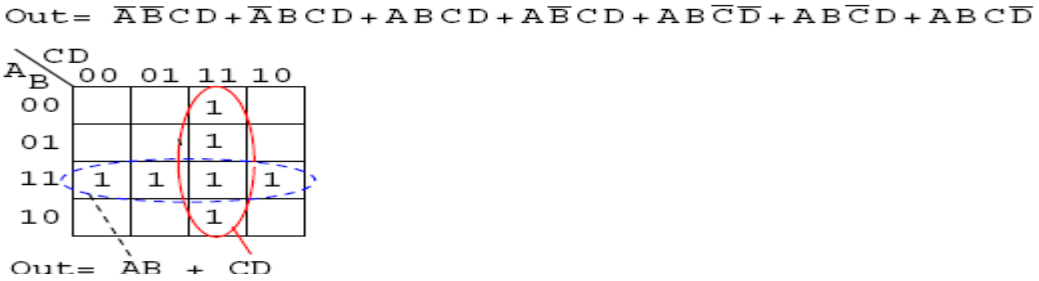
Output = AB + BC + AC

In the **pair looping,** only one variable is in uncomplemented and complemented form, hence pairs eliminate single variables. **The pair eliminates only one variable and their complement.**

b)Quads

A k-map that contains a group of four 1's that are adjacent to each other in a form of line or square is called quad.

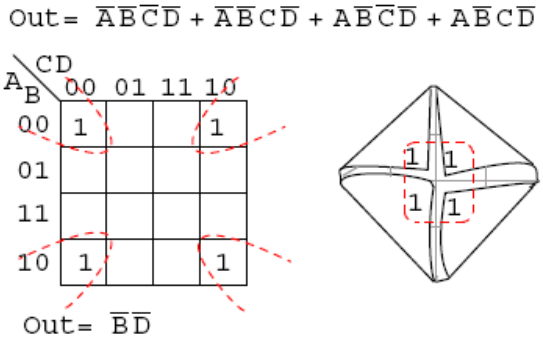
Example: Simplify the following Boolean expression.



The other product terms are placed in a similar manner. Encircling the largest groups possible, two groups of four are shown above. The dashed horizontal group corresponds the the simplified product term AB. The vertical group corresponds to Boolean CD. Since there are two groups, there will be two product terms in the **Sum-Of-Products result of Out=AB+CD.**

Example: Simplify the following Boolean expression.

The four cells above are a group of four because they all have the Boolean variables B' and D' in common. In other words, B=0 for the four cells, and D=0 for the four cells. The other variables (A, B) are 0 in some cases, 1 in other cases with respect to the four corner cells. Thus, these variables (A, B) are not involved with this group of four. This single group comes out of the map as one product term for the simplified result: Out=B'D'



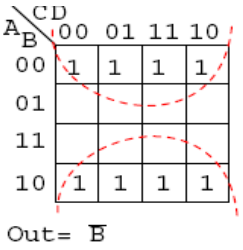
The quad eliminates two variables and their complements.

c)Octets

A k-map that contains a group of eight 1's that are adjacent to each other in a form of line or square is called octet.

Example: For the K-map below, roll the top and bottom edges into a cylinder forming eight adjacent cells.

F(A,B,C,D)={ 0,1,2,3,8,9,10,11}

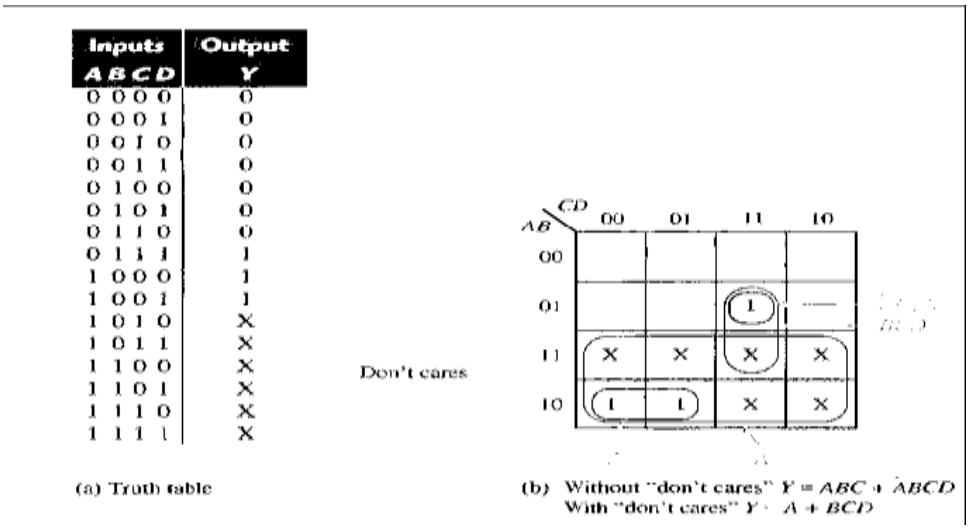


The above group of eight has one Boolean variable in common: B=0. Therefore, the one group of eight is covered by one p-term: B'. The original eight term Boolean expression simplifies to Out=B'

The octet eliminates three variables and their complements.

Don't Care Conditions

Some logic circuits can be designed so that there are certain input conditions that do not produce any specified output level i.e 0 or 1. That means certain input condition of some logic circuits produce the output as neither 0 nor 1. This type of input is in an unspecified form called don't cares. The truth table contain some don't care output for some input conditions.



Inputs A B C D	Output Y
0 0 0 0	0
0 0 0 1	0
0 0 1 0	0
0 0 1 1	0
0 1 0 0	0
0 1 0 1	0
0 1 1 0	0
0 1 1 1	1
1 0 0 0	1
1 0 0 1	1
1 0 1 0	X
1 0 1 1	X
1 1 0 0	X
1 1 0 1	X
1 1 1 0	X
1 1 1 1	X

(a) Truth table

(b) Without "don't cares" $Y = ABC + \bar{A}BCD$
With "don't cares" $Y = A + BCD$

Table 2.11

2.8 Boolean expression for outputs

Example1: Derive the Boolean expression for the output of given logic diagram shown fig 2.15

Step 1:

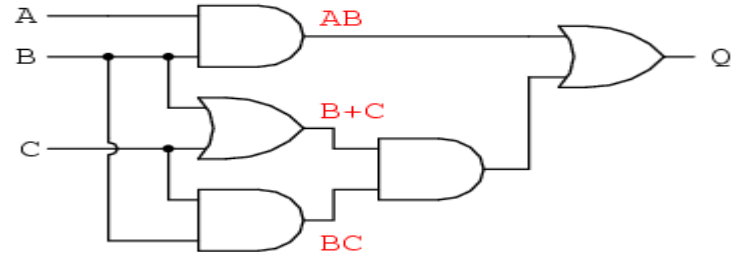
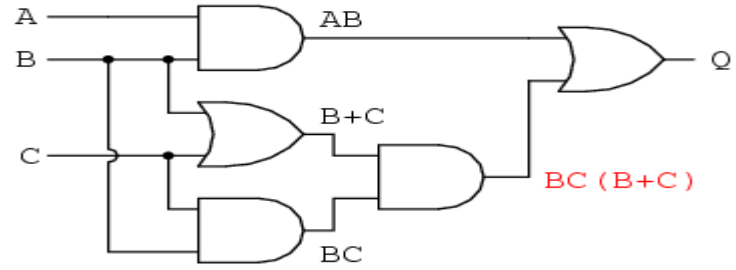
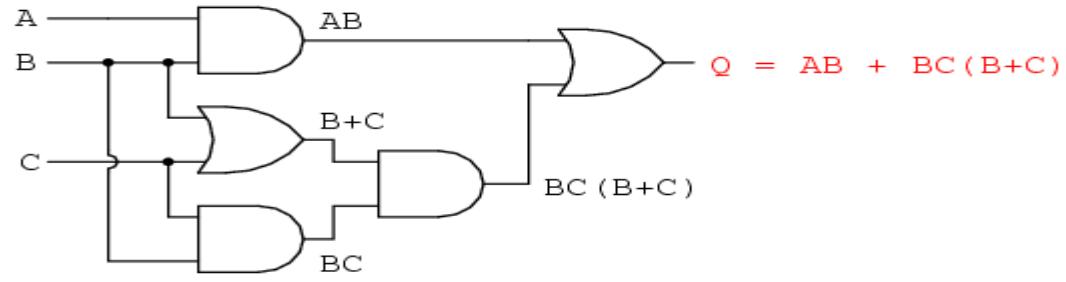


Fig 2.15

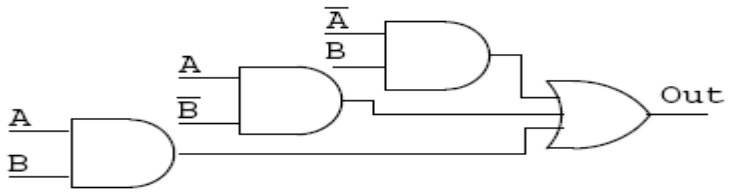
Step 2:



Step 3:



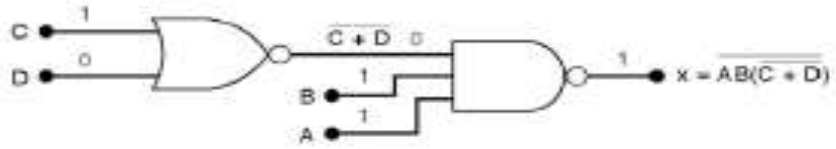
Example 2: Derive the Boolean expression for the output of given logic diagram shown fig 2.16



$Out = \overline{A}B + A\overline{B} + AB$

Fig 2.16

Example 3: Derive the Boolean expression for the output of given logic diagram.



Example 4:Derive the Boolean expression for the output of given logic diagram Shown fig 2.17

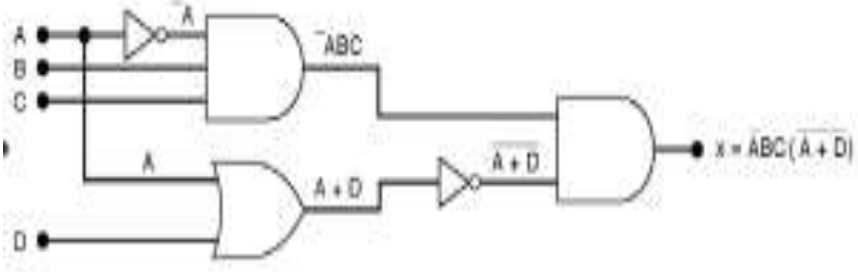


Fig 2.17

Example 5: Derive the Boolean expression for the output of given logic diagram shown fig 2.18

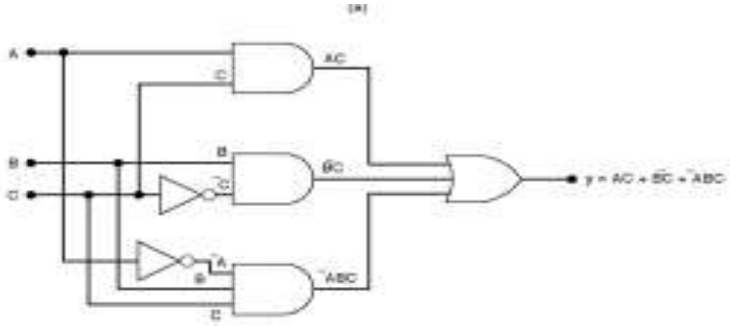
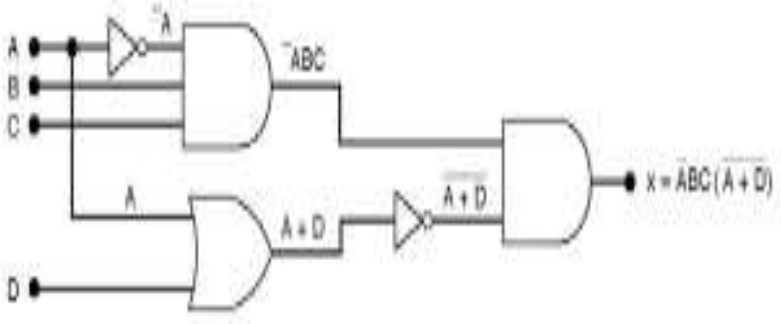


Fig 2.18

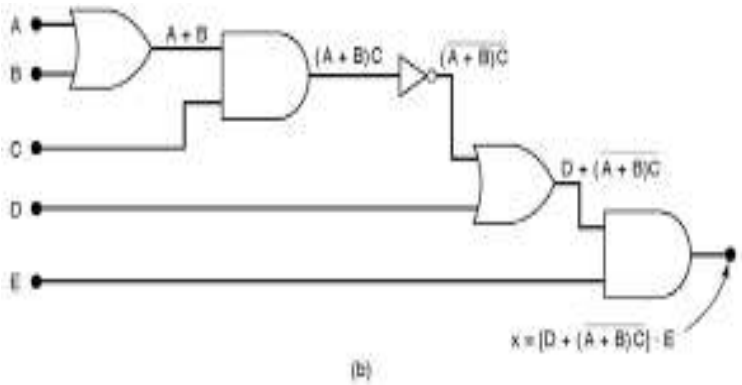
Example 6:

Derive the Boolean expression for the output of given logic diagram shown fig 2.19



shown fig 2.19

Example 7:
Derive the Boolean expression for the output of given logic diagram shown fig 2.20



shown fig 2.20

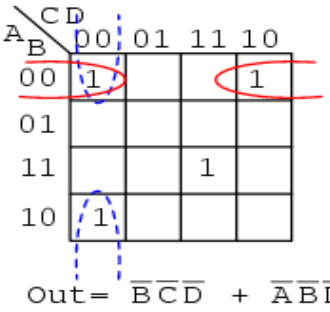
2.9 Simplification of Boolean expression using Karnaugh map (up to 4 variables)

The procedure for simplifying a Boolean expression is given below

1. Construct a k-map and place 1's in these squares corresponding th the 1's in the truth table. Place 0's in the other squares.
2. Encircle (loop) the possible octets, quads and pairs.
3. If any isolated 1's remains, encircle each
4. Write the Boolean expression corresponding to the octet, quad and pair loops.

Example1: Simplify the given logic equation by using k-map

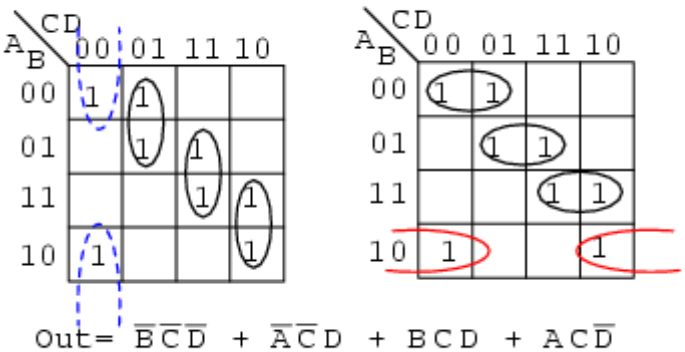
$$Out= \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}C\overline{D} + A\overline{B}\overline{C}\overline{D} + ABCD$$



$$Out= \overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}\overline{D} + ABCD$$

Example 2: Simplify the given logic equation by using k-map

$$Out= \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}\overline{C}D + \overline{A}B\overline{C}D + \overline{A}BCD + ABCD + ABC\overline{D} + A\overline{B}\overline{C}\overline{D} + A\overline{B}C\overline{D}$$

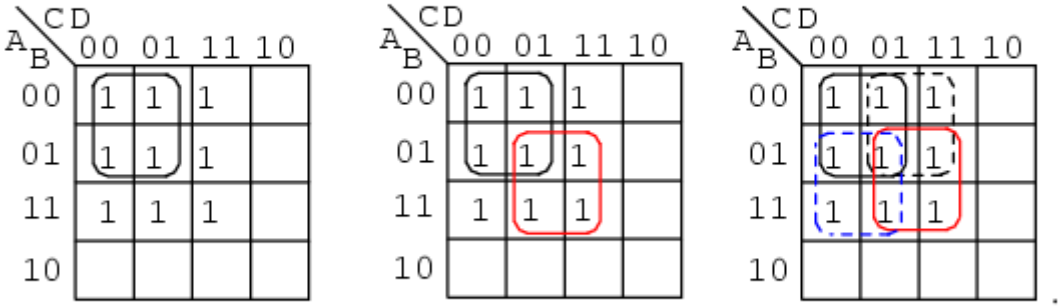


$$Out= \overline{B}\overline{C}\overline{D} + \overline{A}\overline{C}D + BCD + AC\overline{D}$$

$$Out= \overline{A}\overline{B}\overline{C} + \overline{A}BD + ABC + A\overline{B}\overline{D}$$

Example 3: Simplify the given logic equation by using k-map

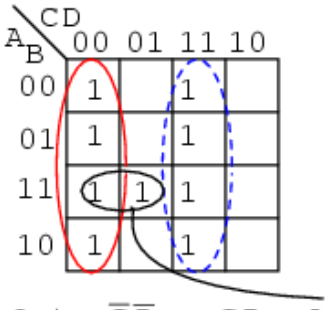
$$Out = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + AB\bar{C}\bar{D} + AB\bar{C}D + ABCD$$



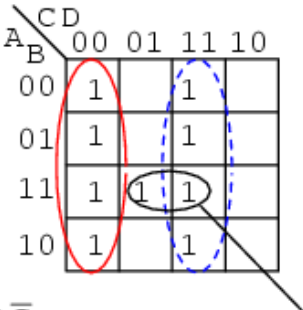
$$Out = \bar{A}\bar{C} + \bar{A}D + B\bar{C} + BD$$

Example 4: Simplify the given logic equation by using k-map

$$Out = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}CD + A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D + A\bar{B}C\bar{D} + A\bar{B}CD + AB\bar{C}\bar{D}$$



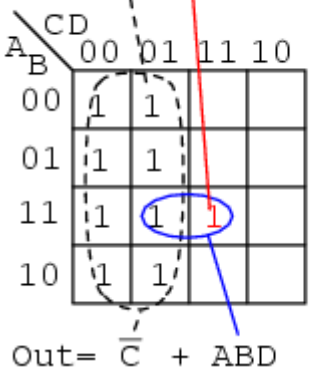
$$Out = \bar{C}\bar{D} + CD + AB\bar{C}$$



$$Out = \bar{C}\bar{D} + CD + ABD$$

Example 5: Simplify the given logic equation by using k-map

$$Out = \bar{C} + ABCD$$



$$Out = \bar{C} + ABD$$

Simplification by Boolean Algebra

$$Out = \bar{C} + ABCD$$

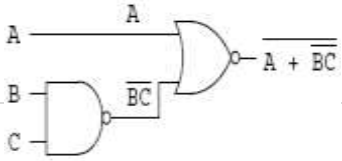
Applying rule $\mathbf{A + \bar{A}B = A + B}$ to the $\bar{C} + ABCD$ term

$$Out = \bar{C} + ABD$$

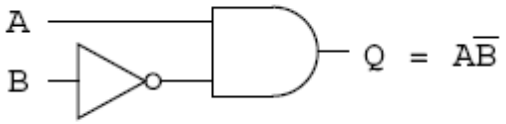
2.10 Constructing logic circuits for the Boolean expressions

Example 1: Draw the logic diagram for the given Boolean expression

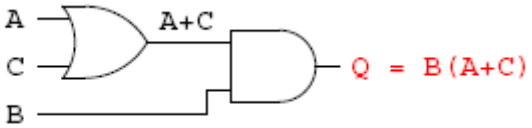
Y=(A+(BC)')'



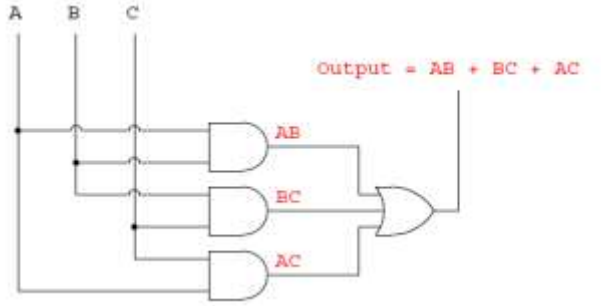
Example 2: Draw the logic diagram for the given Boolean expression Q=AB'



Example 3: Draw the logic diagram for the given Boolean expression Q=B(A+C)

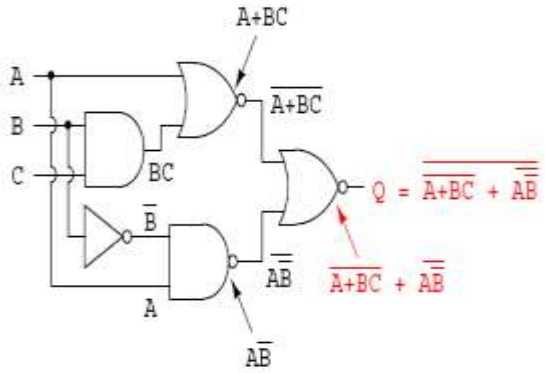


Example 4: Draw the logic diagram for the given Boolean expression Y=AB+BC+AC



Example 5: Draw the logic diagram for the given boolean expression

Y=((A+BC)'+(AB')')'



%%%%%%%%%%%%%%%%%%%%%%%%%%%%

REVIEW QUESTIONS

Two Mark

1	What is the equivalent binary of given 12_{10}?
2	What is the equivalent decimal of given 101?
3	What is the equivalent binary of given 9_8?
4	What is the equivalent binary of given B_h?
5	What is the equivalent decimal of given 10_8?
6	What is the equivalent decimal of given $1F_h$?
7	What is logic gate?
8	What are the types of logic gates?
9	Give the truth table of AND gate.
10	Give the truth table of NAND gate.
11	Give the truth table of NOR gate.
12	What is Universal gate?
13	What are Universal gate?
14	Derive NOT gate from NAND gate.
15	Derive NOT gate from NOR gate.
16	What is K map?
17	What is OCTETS?
18	What is pair?

Three Mark

1	State the basic laws of Boolean algebra.
2	State De morgans theorems.
3	Draw the symbol EX OR gate and give its truth table.
4	Draw the symbol AND gate and give its truth table.
5	Draw the symbol NAND gate and give its truth table.
6	Draw the symbol NOR gate and give its truth table.
7	Realize AND gate from NAND gate.
8	Realize AND gate from NOR gate.
9	Realize OR gate from NAND gate.
10	Realize OR gate from NOR gate.
11	Construct 3X3 K map with ABC=1.
12	What are the rules in K map for simplification?
13	Simplify the $\bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$ using K map

10 Mark

1	The given decimal number 2345_{10} can be converted into following a) Binary b) octal c) hexadecimal
2	State and explain De morgans theorem.
3	Explain with truth tables for following gates a) AND b) OR c) Ex or d) NOT
4	Realize the AND , OR and NOT gates using universal gates.
5	Simplify the following Boolean expression using K map $\bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D} + \bar{A}BC\bar{D} + A\bar{B}\bar{C}\bar{D} + A\bar{B}CD + AB\bar{C}\bar{D} + ABCD$

UNIT – III

COMBINATIONAL LOGIC

3.1 COMBINATIONAL LOGIC:

Two types of operation that are performed on binary data include arithmetic and logic operations. Basic arithmetic operations include addition, subtraction, multiplication and division. AND, OR and NOT are the basic logic functions.

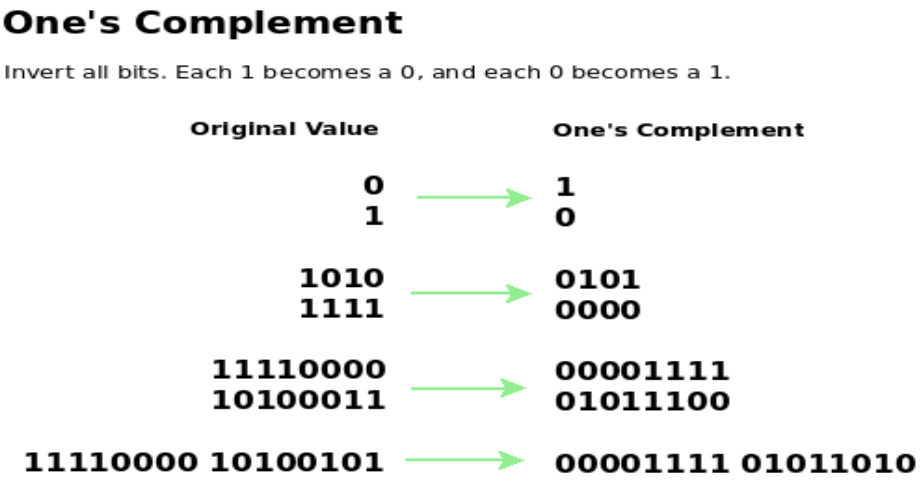
1s and 2s complement concept One's complement and two's complement are two important binary concepts. Two's complement is especially important because it allows us to represent signed numbers in binary, and one's complement is the interim step to finding the two's complement.

Two's complement also provides an easier way to subtract numbers using addition instead of using the longer, more involving subtraction

One's Complement: If all bits in a binary number are inverted by changing each 1 to 0 and each 0 to 1.

Original One's Complement

10011001 --> 01100110
10000001 --> 01111110
11110000 --> 00001111
11111111 --> 00000000



Two's Complement (Binary Additive Inverse)

The two's complement is a method for representing positive and negative integer values in binary. The useful part of two's complement is that it automatically includes the sign bit.

Rule: To form the two's complement, add 1 to the one's complement.

Step 1: Begin with the original binary value

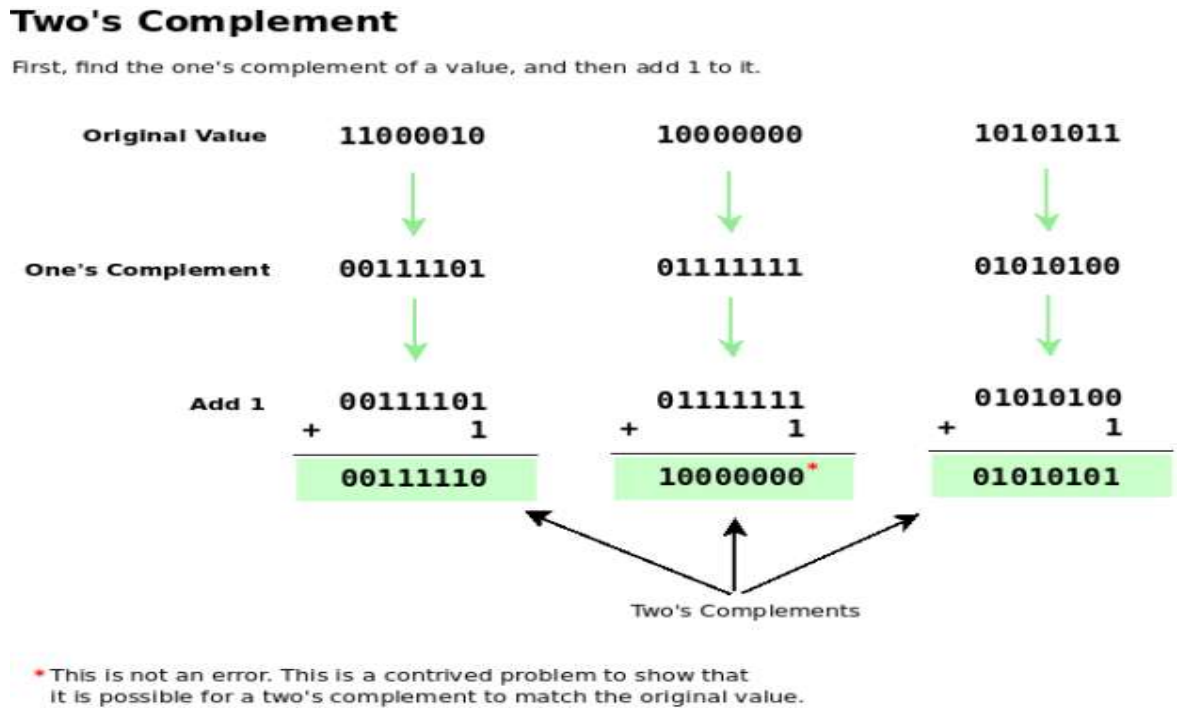
10011001 Original byte

Step 2: Find the one's complement

01100110 One's complement

Step 3: Add 1 to the one's complement

```
  01100110          One's complement
+         1                   Add 1
 ------------
  01100111  <---  Two's complement
```



What is -65d in binary?

Two's complement allows us to represent signed negative values in binary, so here is an introductory demonstration on how to convert a negative decimal value to its negative equivalent in binary using two's complement.

Step 1: Convert 65_{10} to binary. Ignore the sign for now. Use the absolute value. The absolute value of -65_{10} is 65_{10}

65_{10}--> $(01000001)_2$

Step 2: Convert 01000001 to its one's complement.

01000001 --> 10111110

Step 3: Convert 10111110b to its two's complement by adding 1 to the one's complement.

```
  10111110
+         1
 ------------
  10111111  <---  Two's complement
```

10111111b is -65_{10} in binary. We know this it true because if we add $(01000001)_2$ (+65d) to 10111111b (-65d) and *ignore the carry bit*, the sum is 0, which is what we obtain if we add +65 + (-65) = 0.

```
  01000001  +65
+ 10111111  -65
--------------------
 100000000b   0d
 ^
 |
```

Ignore the carry bit for now. What matters is that original number of bits (D7-D0) are all 0.

We will examine signed binary values in more detail later. For now, understand the difference between one's complement and two's complement and practice converting between them.

Practice

Convert to one's complement:

1010

11110000

10111100 11000000

10100001

Convert to two's complement:

1010

11110000

10000000

011111111

Basic Rules of Binary Addition and Subtraction

we can write the basic rules of

binary addition as follows:

1. 0 + 0 = 0.
2. 0 + 1 = 1.
3. 1 + 0 = 1.
4. 1 + 1 = 0 with a carry of '1' to the next more significant bit.
5. 1 + 1 + 1 = 1 with a carry of '1' to the next more significant bit.

The basic principles of binary subtraction include the following:

1. 0 - 0 = 0.
2. 1 - 0 = 1.
3. 1 - 1 = 0.
4. 0 - 1 = 1 with a borrow of 1 from the next more significant bit.

Addition of Larger-Bit Binary Numbers

The addition of larger binary integers, fractions or mixed binary numbers is performed columnwise in just the same way as in the case of decimal numbers. In the case of binary numbers, however, we follow the basic rules of addition of two or three binary digits, as outlined earlier. The process of adding two larger-bit binary numbers can be best illustrated with the help of an example.

Consider two generalized four-bit binary numbers (A3 A2 A1 A0_ and (B3 B2 B1 B0_, with A0 and B0 representing the LSB and A3 and B3 representing the MSB of the two numbers. The addition of these two numbers is performed as follows. We begin with the LSB position. We add the LSB bits and record the sum S0 below these bits in the same column and take the carry C0, if any, to the next columnof bits. For instance, if A0 = 1 and B0 = 0, then S0 = 1 and C0 = 0. Next we add the bits A1 and B1 and the carry C0 from the previous addition. The process continues until we reach the MSB bits. The four steps are shown ahead. C0, C1, C2 and C3 are carrys, if any, produced as a result of adding first, second, third and fourth column bits respectively, starting from LSB and roceeding towards MSB. A similar procedure is followed when the given numbers have both integer as well as fractional parts:

1.

		(C_0)	
A_3	A_2	A_1	A_0
B_3	B_2	B_1	B_0
			S_0

2.

	(C_1)	(C_0)	
A_3	A_2	A_1	A_0
B_3	B_2	B_1	B_0
		S_1	S_0

3.

(C_2)	(C_1)	(C_0)	
A_3	A_2	A_1	A_0
B_3	B_2	B_1	B_0
	S_2	S_1	S_0

4.

	(C_2)	(C_1)	(C_0)	
	A_3	A_2	A_1	A_0
	B_3	B_2	B_1	B_0
C_3	S_3	S_2	S_1	S_0

Addition Using the 2's Complement Method

The 2's complement is the most commonly used code for processing positive and negative binary numbers. It forms the basis of arithmetic circuits in modern computers. When the decimal numbers to be added are expressed in 2's complement form, the addition of these numbers, following the basic laws of binary addition, gives correct results. Final carry obtained, if any, while adding MSBs should be disregarded. To illustrate this, we will consider the following four different cases:

1. Both the numbers are positive.
2. Larger of the two numbers is positive.
3. The larger of the two numbers is negative.
4. Both the numbers are negative.

Case 1

- Consider the decimal numbers +37 and +18.
- The 2's complement of +37 in eight-bit representation = 00100101.
- The 2's complement of +18 in eight-bit representation = 00010010.
- The addition of the two numbers, that is, +37 and +18, is performed as follows

$$\begin{array}{r} 00100101 \\ +\,00010010 \\ \hline 00110111 \end{array}$$

- The decimal equivalent of $(00110111)_2$ is (+55), which is the correct answer.

Case 3

- Consider the two decimal numbers +18 and −37.
- −37 in 2's complement form in eight−bit representation = 11011011.
- +18 in 2's complement form in eight−bit representation = 00010010.
- The addition of the two numbers, that is, −37 and +18, is performed as follows:

$$\begin{array}{r} 11011011 \\ +\,00010010 \\ \hline 11101101 \end{array}$$

- The decimal equivalent of $(11101101)_2$, which is in 2's complement form, is −19, which is the correct answer. 2's complement representation was discussed in detail in Chapter 1 on number systems.

Case 4

- Consider the two decimal numbers −18 and −37.
- −18 in 2's complement form is 11101110.
- −37 in 2's complement form is 11011011.
- The addition of the two numbers, that is, −37 and −18, is performed as follows:

$$\begin{array}{r} 11011011 \\ +\,11101110 \\ \hline 11001001 \end{array}$$

- The final carry in the ninth bit position is disregarded.
- The decimal equivalent of $(11001001)_2$, which is in 2's complement form, is −55, which is the correct answer.

Subtraction of Larger-Bit Binary Numbers

Subtraction is also done columnwise in the same way as in the case of the decimal number system. In the first step, we subtract the LSBs and subsequently proceed towards the MSB. Wherever the subtrahend (the bit to be subtracted) is larger than the minuend, we borrow from the next adjacent higher bit position having a '1'. As an example, let us go through different steps of subtracting (1001)2 from (1100)2. In this case, '1' is borrowed from the second MSB position, leaving a '0' in that position. The borrow is first brought to the third MSB position to make it '10'. Out of '10' in this position, '1' is taken to the LSB position to make '10' there, leaving a '1' in the third MSB position. 10−1 in the LSB column gives '1', 1−0 in the third MSB column gives '1', 0−0 in the second MSB column gives '0' and 1−1 in the MSB also gives '0' to
complete subtraction. Subtraction of mixed numbers is also done in the same manner. The above-mentioned steps are summarized as follows:

1.	1	1	0	0	2.	1	1	0	0
	1	0	0	1		1	0	0	1
				1				1	1
3.	1	1	0	0	4.	1	1	0	0
	1	0	0	1		1	0	0	1
		0	1	1		0	0	1	1

Subtraction Using 2's Complement Arithmetic

Subtraction is similar to addition. Adding 2's complement of the subtrahend to the minuend and disregarding the carry, if any, achieves subtraction. The process is illustrated by considering six different cases:

1. Both minuend and subtrahend are positive. The subtrahend is the smaller of the two.

2. Both minuend and subtrahend are positive. The subtrahend is the larger of the two.

3. The minuend is positive. The subtrahend is negative and smaller in magnitude.

4. The minuend is positive. The subtrahend is negative and greater in magnitude.

5. Both minuend and subtrahend are negative. The minuend is the smaller of the two.

6. Both minuend and subtrahend are negative. The minuend is the larger of the two.

Case 1

- Let us subtract +14 from +24.
- The 2's complement representation of +24 = 00011000.
- The 2's complement representation of +14 = 00001110.
- Now, the 2's complement of the subtrahend (i.e. +14) is 11110010.
- Therefore, +24 − (+14) is given by

$$\begin{array}{r} 00011000 \\ +\ 11110010 \\ \hline 00001010 \end{array}$$

with the final carry disregarded.

- The decimal equivalent of $(00001010)_2$ is +10, which is the correct answer.

Case 2

- Let us subtract +24 from +14.
- The 2's complement representation of +14 = 00001110.
- The 2's complement representation of +24 = 00011000.
- The 2's complement of the subtrahend (i.e. +24) = 11101000.
- Therefore, +14 − (+24) is given by

$$\begin{array}{r} 00001110 \\ +\ 11101000 \\ \hline 11110110 \end{array}$$

- The decimal equivalent of $(11110110)_2$, which is of course in 2's complement form, is −10 which is the correct answer.

Case 3

- Let us subtract −14 from +24.
- The 2's complement representation of +24 = 00011000 = minuend.
- The 2's complement representation of −14 = 11110010 = subtrahend.
- The 2's complement of the subtrahend (i.e. −14) = 00001110.
- Therefore, +24 − (−14) is performed as follows:

$$\begin{array}{r} 00011000 \\ +\ 00001110 \\ \hline 00100110 \end{array}$$

- The decimal equivalent of $(00100110)_2$ is +38, which is the correct answer.

Case 4

- Let us subtract −24 from +14.
- The 2's complement representation of +14 = 00001110 = minuend.
- The 2's complement representation of −24 = 11101000 = subtrahend.
- The 2's complement of the subtrahend (i.e. −24) = 00011000.
- Therefore, +14 − (−24) is performed as follows:

$$\begin{array}{r} 00001110 \\ +\ 00011000 \\ \hline 00100110 \end{array}$$

- The decimal equivalent of $(00100110)_2$ is +38, which is the correct answer.

Case 5

- Let us subtract −14 from −24.
- The 2's complement representation of −24 = 11101000 = minuend.
- The 2's complement representation of −14=11110010 = subtrahend.
- The 2's complement of the subtrahend = 00001110.
- Therefore, −24 − (−14) is given as follows:

$$\begin{array}{r} 11101000 \\ +\ 00001110 \\ \hline 11110110 \end{array}$$

- The decimal equivalent of $(11110110)_2$, which is in 2's complement form, is −10, which is the correct answer.

Case 6

- Let us subtract −24 from −14.
- The 2's complement representation of −14 = 11110010 = minuend.
- The 2's complement representation of −24=11101000 = subtrahend.
- The 2's complement of the subtrahend = 00011000.
- Therefore, −14 − (−24) is given as follows:

$$\begin{array}{r} 11110010 \\ +\,00011000 \\ \hline 00001010 \end{array}$$

with the final carry disregarded.

- The decimal equivalent of $(00001010)_2$, which is in 2's complement form, is +10, which is the correct answer.

The different steps to be followed to do subtraction in 2's complement arithmetic are summarized as follows:

1. Represent the minuend and subtrahend in 2's complement form.

2. Find the 2's complement of the subtrahend.

3. Add the 2's complement of the subtrahend to the minuend.

4. Disregard the final carry, if any.

5. The result is in 2's complement form.

6. 2's complement notation can be used to perform subtraction when the expected result of subtraction lies in the range from −2n−1 to +(2n−1− 1), n being the number of bits used to represent the numbers.

Exercise 1

Subtract (1110.011)2 from (11011.11)2 using basic rules of binary subtraction and verify the result by showing equivalent decimal subtraction.

Exercise 2

Subtract (a) (−64)10 from (+32)10 and (b) (29.A)16 from (4F.B)16. Use 2's complement arithmetic.

Sign-Bit Magnitude

In the sign-bit magnitude representation of positive and negative decimal numbers, the MSB represents the 'sign', with a '0' denoting a plus sign and a '1' denoting a minus sign. The remaining bits represent the magnitude. In eight-bit representation, while MSB represents the sign, the remaining seven bits represent the magnitude.

3.2 Arithmetic Circuits

A *combinational circuit* is one where the output at any time depends only on the present combination of inputs at that point of time with total disregard to the past state of the inputs shown in fig 3.1. The logic gate is the most basic building block of combinational logic. The logical function performed by a combinational circuit is fully defined by a set of Boolean expressions.

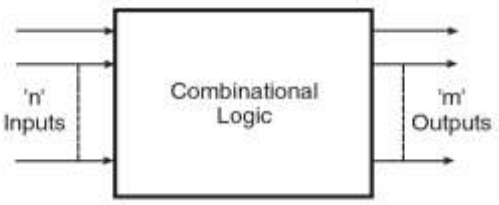


Fig 3.1

The other category of logic circuits, called *sequential logic circuits*, comprises both logic gates and memory elements such as flip-flops. Owing to the presence of memory elements, the output in a sequential circuit depends upon not only the present but also the past state of inputs.

Figure shows the block schematic representation of a generalized combinational circuit having n input variables and m output variables or simply outputs. Since the number of input variables is n, there are 2^n possible combinations of bits at the input. Each output can be expressed in terms of input variables by a Boolean expression, with the result that the generalized system of Fig. 3.1 can be expressed by m Boolean expressions. As an illustration, Boolean expressions describing the function of a four-input OR/NOR gate are given as

$$Y_1 \text{ (OR output)} = A + B + C + D \quad \text{and} \quad Y_2 \text{ (NOR output)} = \overline{A + B + C + D}$$

Arithmetic circuits

A lo*gic circuit which is used to perform arithmetic operations like addition, subtraction, multiplication,division etc is called* ***arithmetic circuits***

Exa*mples:* Half adder , Full adder , Half sub tractor , Full sub tractor.

3.2.1 Half-Adder

A *half-adder* is an arithmetic circuit block that can be used to add two bits. Such a circuit thus has two inputs that represent the two bits to be added and two outputs, with one producing the SUM output and the other producing the CARRY. Figure 3.2 shows the truth table of a half-adder, showing all possible input combinations and the corresponding outputs.

The Boolean expressions for the SUM and CARRY outputs are given by the equations

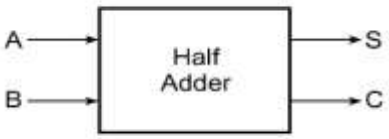
$$\text{SUM } S = A.\overline{B} + \overline{A}.B$$

$$\text{CARRY } C = A.B$$

the first one representing the SUM output of an EX-OR gate and the second one representing the CARRY output of an AND gate

truth table block diagram

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



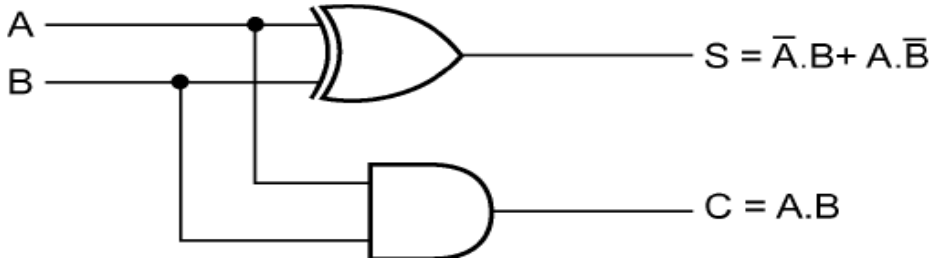
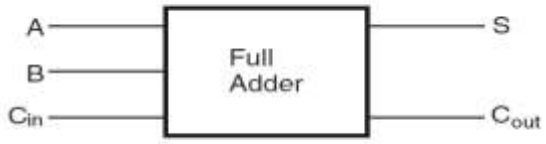


Fig 3.2

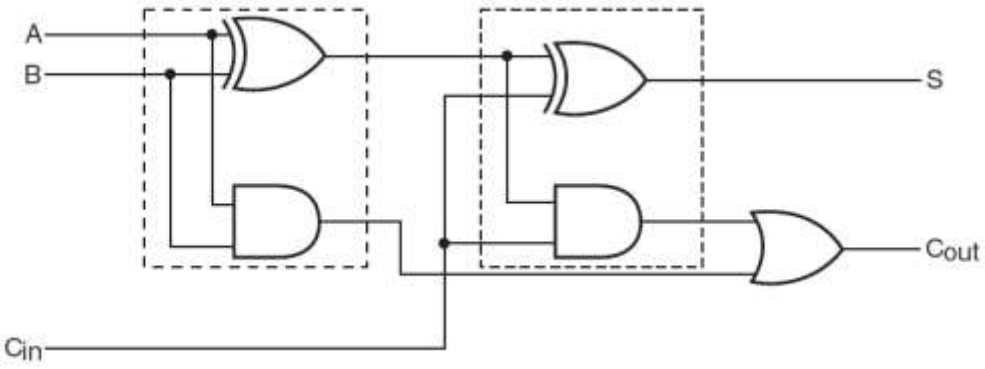
3.2.2 Full Adder

A *full adder* circuit is an arithmetic circuit block that can be used to add three bits to produce a SUM and a CARRY output. Such a building block shown fig 3.3 becomes a necessity when it comes to adding binary numbers with a large number of bits. The full adder circuit overcomes the limitation of the half-adder, which can be used to add two bits only.

Block diagram Truthtable



A	B	C_{in}	SUM (S)	C_{out}
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



Logic diagram

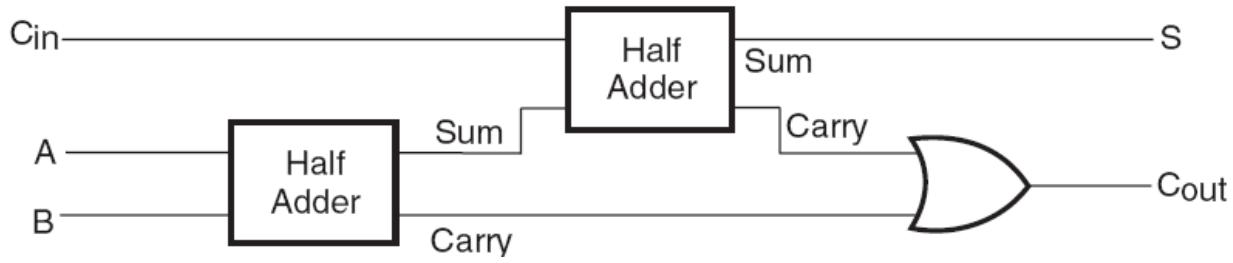


Fig 3.3

$$S = \overline{A}.\overline{B}.C_{\text{in}} + \overline{A}.B.\overline{C}_{\text{in}} + A.\overline{B}.\overline{C}_{\text{in}} + A.B.C_{\text{in}}$$

$$C_{\text{out}} = \overline{A}.B.C_{\text{in}} + A.\overline{B}.C_{\text{in}} + A.B.\overline{C}_{\text{in}} + A.B.C_{\text{in}}$$

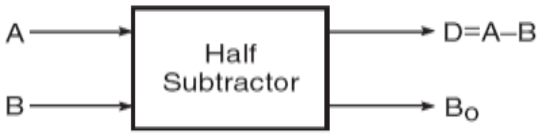
the simplified Boolean expression for Cout is given by the equation

$$C_{\text{out}} = B.C_{\text{in}} + A.B + A.C_{\text{in}}$$

3.2.3 Half-Subtractor

A *half-subtractor* is a combinational circuit that can be used to subtract one binary digit from another to produce a DIFFERENCE output and a BORROW output. The BORROW output here specifies whether a '1' has been borrowed to perform the subtraction. The truth table of a half-subtractor, as shown in Fig 3.4 The Boolean expressions for the two outputs are given by the equations

Block diagram



Truth table

A	B	D	B_o
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

LOGICDIAGRAM

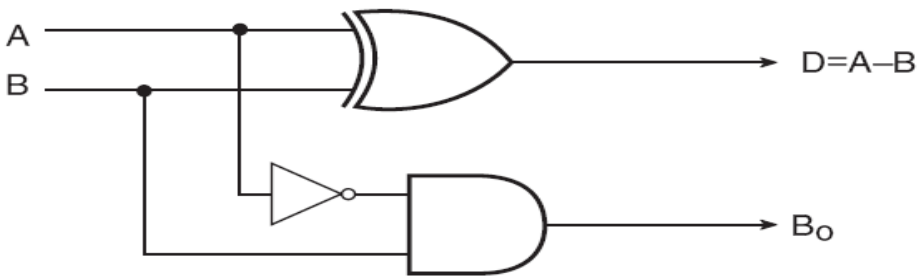


Fig 3.4

While the expression for the DIFFERENCE (D_ output is that of an EX-OR gate, the expression for the BORROW output (Bo_ is that of an AND gate with input A complemented before it is fed to the gate. Figure 3.4 shows the logic implementation of a

half-subtractor. Comparing a half-subtractor with a half-adder, we find that the expressions for the SUM and DIFFERENCE outputs are just the same. The expression for BORROW in the case of the half-subtractor is also similar to what we have for CARRY in the case of the half-adder. If the input A, that is, the minuend, is complemented, an AND gate can be used to implement the BORROW output.

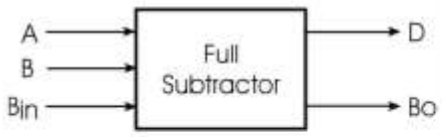
3.2.4 Full Subtractor A *full subtractor* performs subtraction operation on three bits, There are two outputs, namely the DIFFERENCE output D and the BORROW output Bo. The BORROW output bit tells whether the minuend bit needs to borrow a '1' from the next possible higher minuend bit. Figure 3.5 shows the truth table of a full subtractor.

The Boolean expressions for the two output variables are given by the equations

$$D = \overline{A}.\overline{B}.B_{\text{in}} + \overline{A}.B.\overline{B}_{\text{in}} + A.\overline{B}.\overline{B}_{\text{in}} + A.B.B_{\text{in}}$$

$$B_{\text{o}} = \overline{A}.\overline{B}.B_{\text{in}} + \overline{A}.B.\overline{B}_{\text{in}} + \overline{A}.B.B_{\text{in}} + A.B.B_{\text{in}}$$

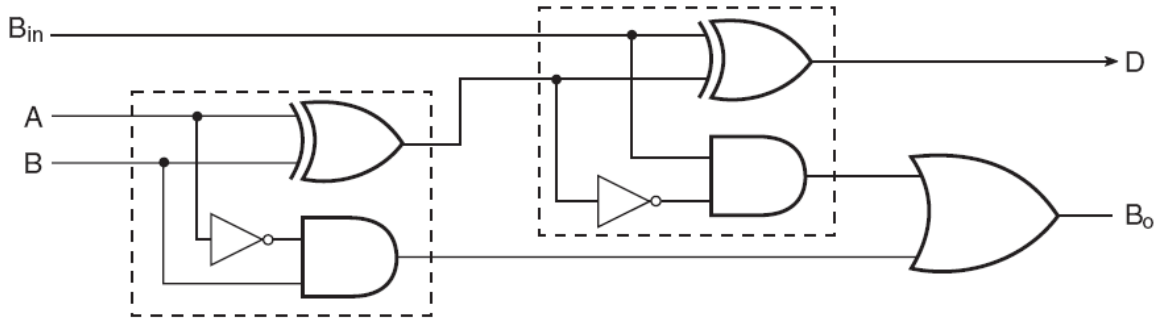
Block diagram



Truthtable

Minuend (A)	Subtrahend (B)	Borrow In (B_{in})	Difference (D)	Borrow Out (B_o)
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

Logic diagram of Full subtractor using two half subtractors



Block diagram of full subtractor using two half subtractors

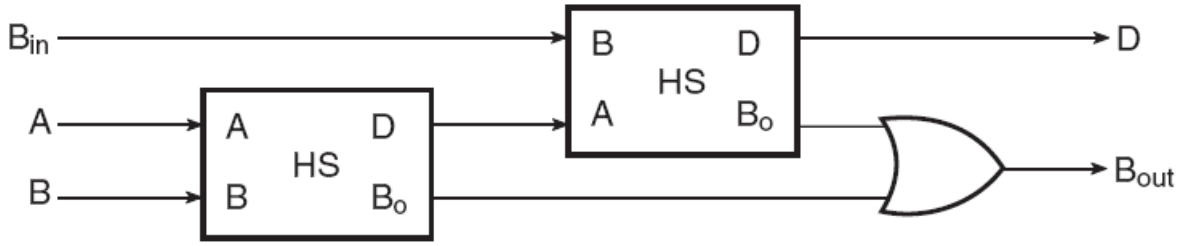


Fig 3.5

3.3 Multiplexers

A *multiplexer* or *MUX*, also called a *data selector*, is a combinational circuit with more than one input line, one output line and more than one selection line. It means many into one**A multiplexer selects binary information present on any one of the input lines, depending upon the logic status of the selection inputs**. Hence it is called as data selector. If there are n selection lines, then the number of maximum possible input lines is 2^n and the multiplexer is referred to as a 2^n to 1

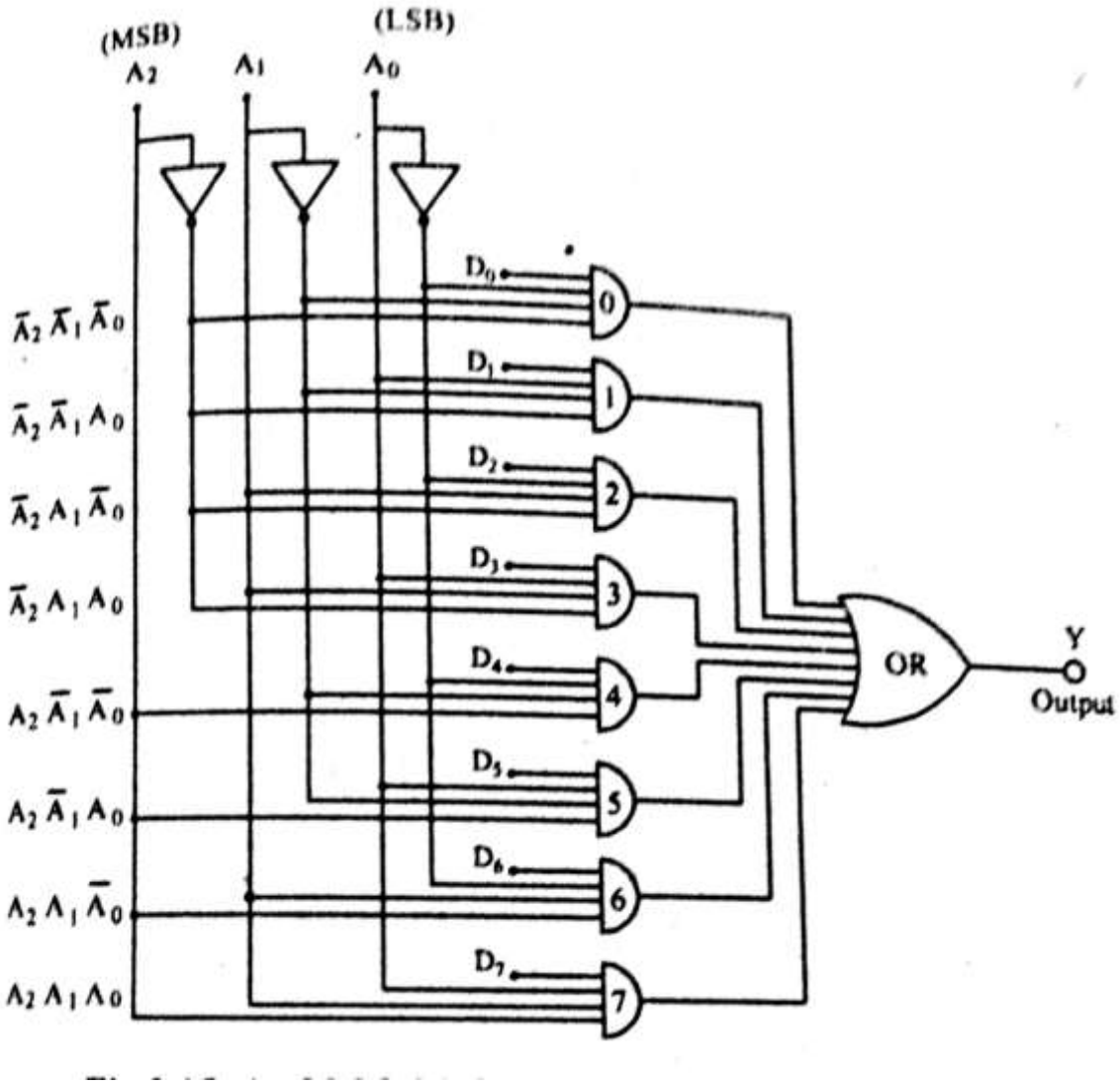


Fig 3.6

Logic diagram shown fig 3.6 is a 8:1 Multiplexer

It consists of three control or address lines named as A_0,A_1 and A_0 and 8 data input lines D_0,D_1,D_2,D_3,D_4,D_5,D_6 and D_7. Y is the output. Depending upon the value of A_2,A_1 and A_0,only one of the input is transmitted to the output.

For example,

When $A_2A_1A_0$=000, the upper AND gate is only enabled while the other gates are disabled.Therefore data bit D_0 is transmitted to the output.ie $Y=D_0$ and so on.

Truth table

Address input			Enabled AND gate	OUTPUT
A_2	A_1	A0		
0	0	0	0	D_0
0	0	1	1	D_1
0	1	0	2	D_2
0	1	1	3	D_3
1	0	0	4	D_4
1	0	1	5	D_5
1	1	0	6	D_6
1	1	1	7	D_7

3.4 Encoders:

It is a digital logic circuit which converts decimal input to a binary coded decimal output. The number of output lines in encoder is less than the number of input lines.

LOGIC DIAGRAM.

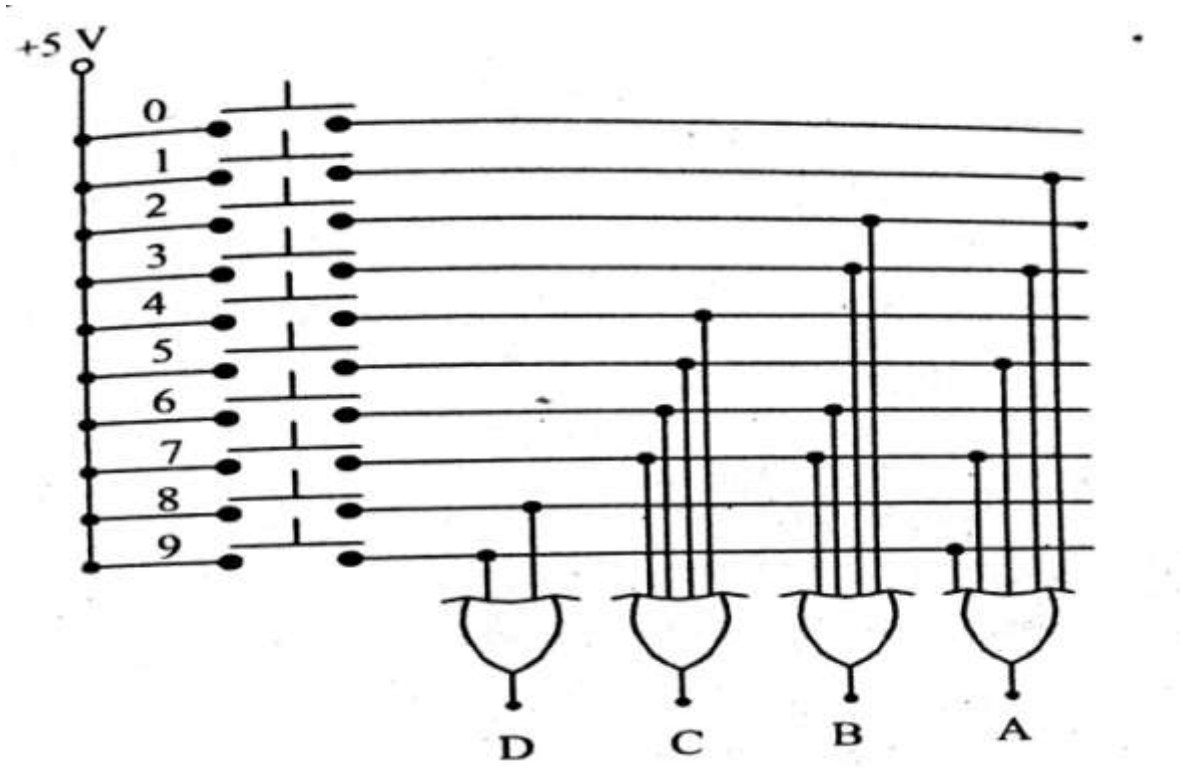


Fig 3.7

It has ten input lines from 0 to 9 and four output lines A,B,C,D.The inputs are represented by switches (decimal number) and the output of the OR gate represents the output (binary coded decimal).The decimal input is applied to the encoder by using push-button switches shown fig 3.7

For example,if switch 1 is pressed,the output of gate A only goes high .Therefore the output DCBA=0001.If switch 2 is pressed , the output of gate B goes high. Therefore the output DCBA=0010 and so on.

TRUTHTABLE.

Enabed decimal input	BCD OUTPUT			
	D	C	B	A
O	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

3.5 Demultiplexers

A *demultiplexer* is a combinational logic circuit with one input line and many output line. It means that an **input line, 2^n output lines and n select lines.** It routes the information present on the input line to any of the output lines. The output line that gets the information present on the input line is decided by the status of the selection lines.

The input data is distributed to any one of the selected output shown fig 3.8 .Hence it is called as data distributor.It has 3 control lines A_2,A_1,A_0, one data input line D, eight output lines Y_0,Y_1,Y_2,Y_3,Y_4,Y_5,Y_6 and Y_7.

TRUTHTABLE

CONTROL ADDRESS			OUTPUT							
A_2	A_1	A_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	00	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

For example,

When A_2,A_1A_0 =000,the AND gate 0 is enabled and the other ANDgates are disabled.Therefoe ,the input is transmitted as Y_0.Similarly when A_2,A_1A_0 =111,the AND gate 8 is enabled and the remaining gates are disabled.So, the output is transmitted via Y_7.

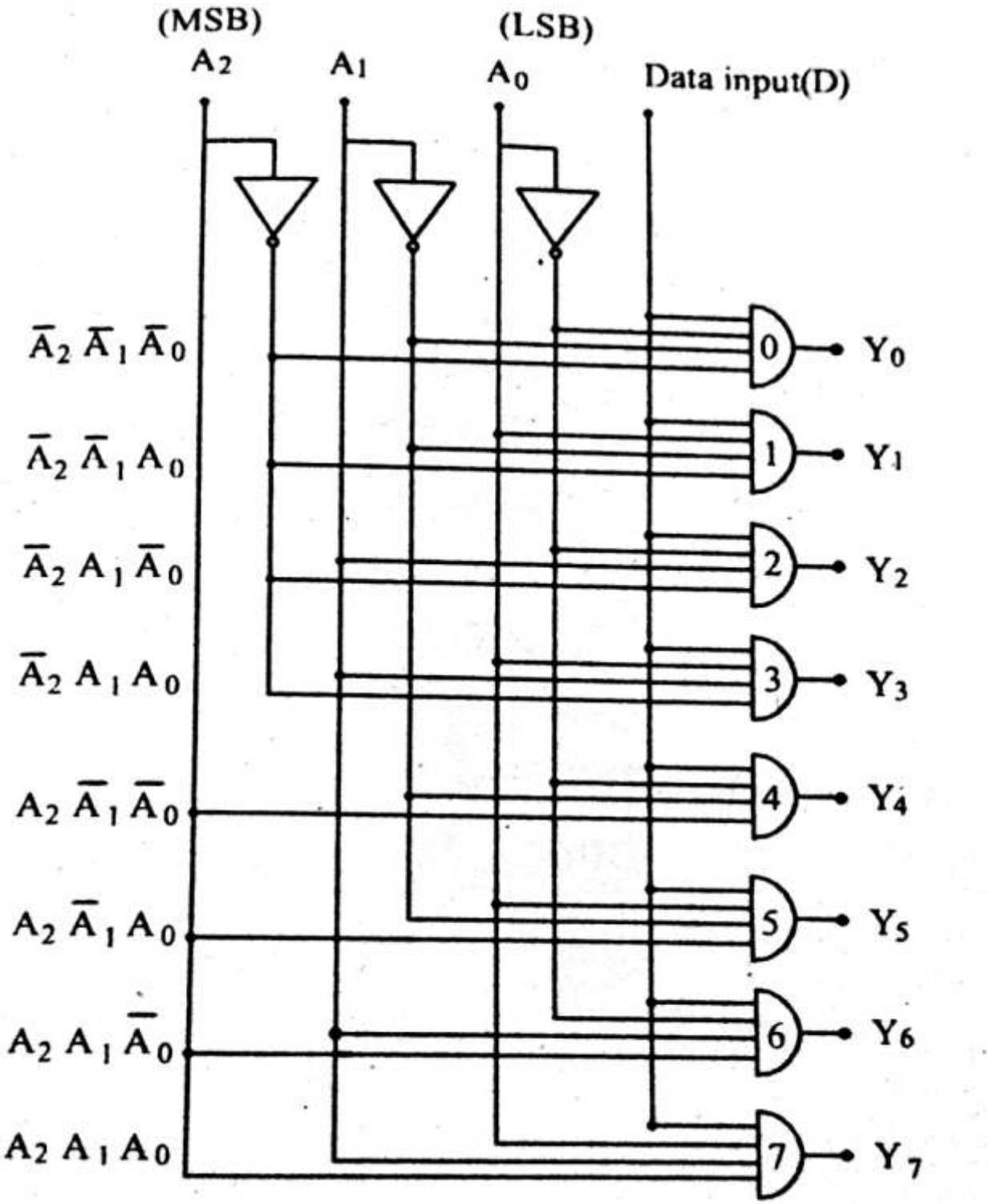
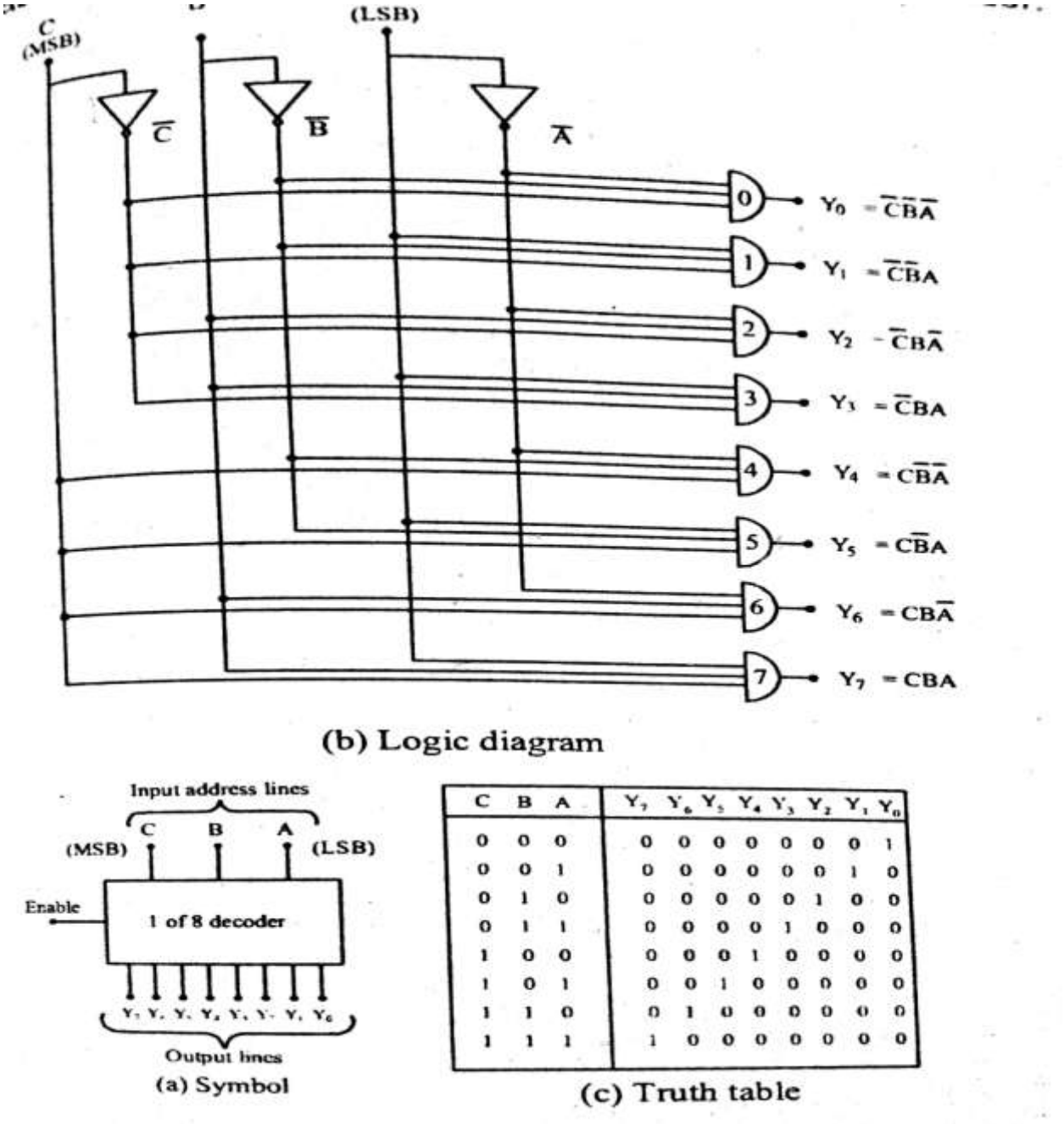


Fig 3.8

3.6 DECODER_:. A *decoder* is a combinational logic circuit with less input line and many output line without any data line. *decoder* is a special case of a demultiplexer without the datainput line.** It means that an **input line, 2^n output lines and n select lines.** I

1 of 8 decoder consists of three control lines, one conrol line and eight output line. It is also called as 3 to 8 decodershown fig 3.9.'

For example,CBA =000,the AND gate 0 is enabled.The output Y_0 is selected.If CBA=111, the AND gate 8 is enabled and the output Y_7 is transmitted.



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

(c) Truth table

Fig 3.9

3.7 Parity generation and checking circuits.

EX-OR and EX-NOR logic gates are commonly used in parity generation and checking circuits. Figures 3.10 respectively show even and odd parity generator circuits for four-bit data.

The parity check operation can also be performed by similar circuits. Figures respectively show simple even and odd parity check circuits for a four-bit data stream. In the circuits shown in Fig., a logic '0' at the output signifies correct parity and a logic '1' signifies one-bit error. Parity generator/checker circuits are available in IC form. 74180 in TTL and 40101 in CMOS are nine-bit odd/even parity generator/checker ICs.

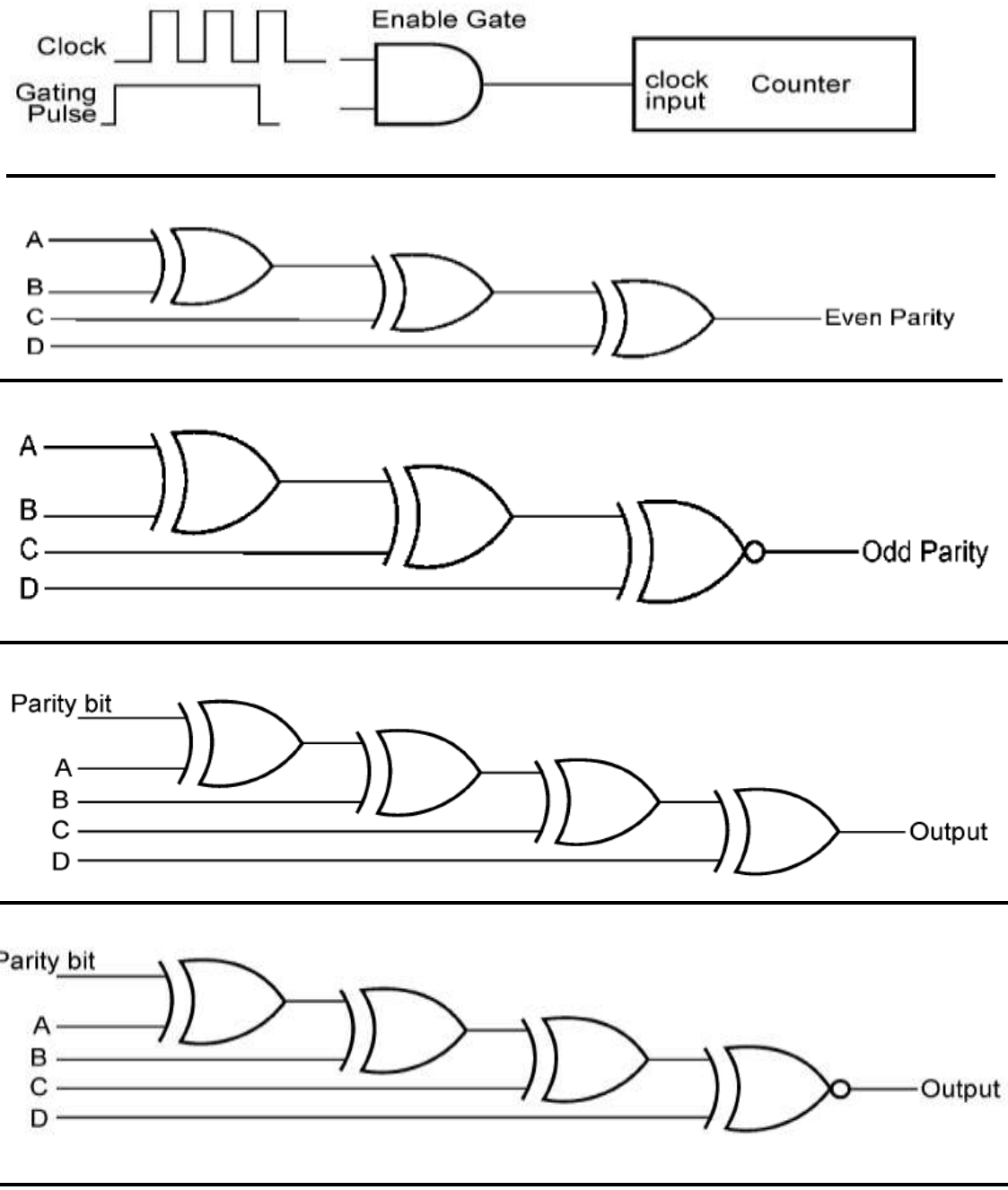


Fig 3.10

3.8 DIGITAL COMPARATOR

It is a logic circuit which is used to compare two binary numbers A and B.(ie the binary numbers $A < B, A > B, A = B$. It is otherwise called as binary comparator.

EX.OR gate is a basic comparator circuit. It consists of two inputs namely A and B and three outputs namely D ,E AND C shown fig 3.11

The output E is enabled when $A = B$

The output D is enabied when $A < B$

The output C is enabled when $A > B$

Logic diagram

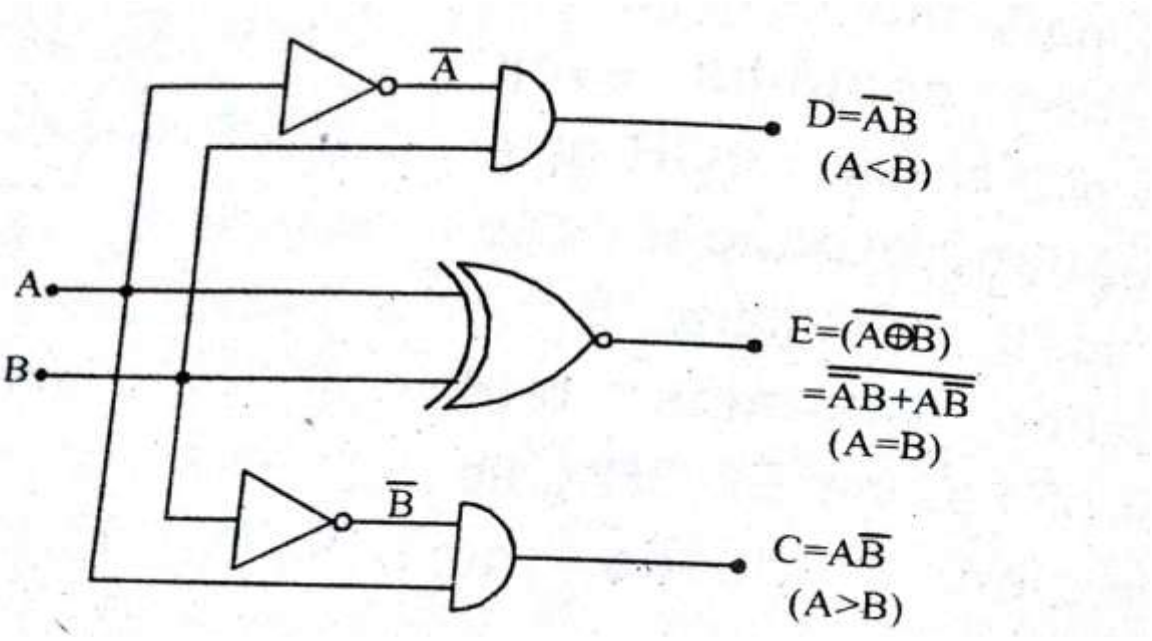


Fig 3.11

Truth table

Inputs		OUTPUTS		
A	**B**	**C** $A > B$	**D** $A < B$	**E** $A = B$
0	**0**	**0**	**0**	**1**
0	**1**	**0**	**1**	**0**
1	**0**	**1**	**0**	**0**
1	**1**	**0**	**0**	**1**

3.9 ARITHMETIC LOGIC UNIT

It is a logic circuit which can perform a set of both arithmetic logic operations.The basic arithmetic operations performed by the ALU are addition, subtraction, multiplication, division etc and logical operations performed by ALU are complement (NOT gate), AND, OR, EX-OR etc.

ALU has a number of selection lines used to select a particular operation in the ALU unit.The selection lines are decoded within the ALU.

It has four data inputs from A and four data inputs from B. These inputs are combined to produce the output Y.

The mode select input line S_2 is used to select arithmetic or logic operation to be performed.

The two function select lines S_1, S_0 are used to select particular arithmetic or logic operation to be performed.

TABLE:

Selection				output	Function
S_2	S_1	S_0	C_{IN}		
0	0	0	0	F=A+1	Increment A
0	0	0	1	F=A-1	Decrement A
0	0	1	0	F=A+B	Addition
0	0	1	1	F=A+B+1	Addition with carry
0	1	0	0	F=A-B	Subtraction
0	1	0	1	F=A-B-1	Subtraction with carry
0	1	1	0	F=A* B	Multiplication
0	1	1	1	$F = A \div B$	Division
1	0	0	X	F=A V B	A OR B
1	0	1	X	$F = A \cup B$	A AND B
1	1	0	X	$F = A \forall B$	A EX-OR B
1	1	1	X	F=$\bar{A}$	Complement A

Select s_2 = 0 for arithmetic operation and s_2 =1 for logic operation. The three input lines S_2, S_1,S_0 are used to select four arithmetic operations and four logic operations. **By using C_{IN} , the number of arithmetic operations are doubled.**

The input carry and output carry are used only in arithmetic operations.

C_{OUT} =1 when borrow is created in subtraction otherwise C_{OUT} = 0

3.10 BCD TO SEVEN SEGMENT DECODER

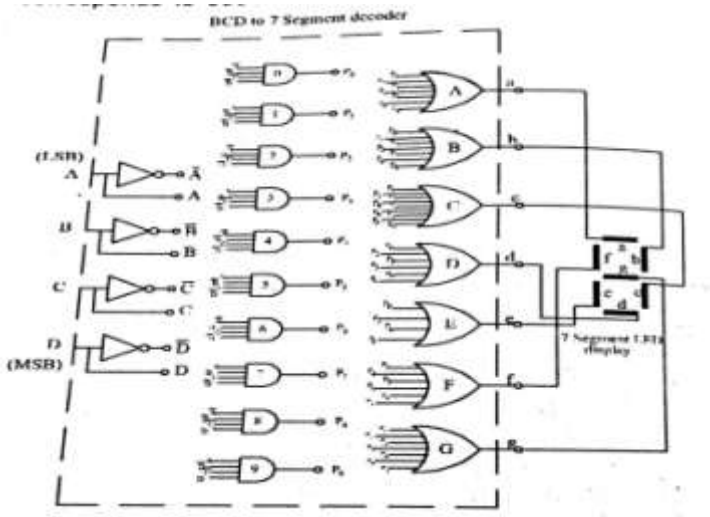
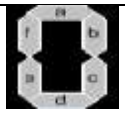
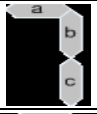
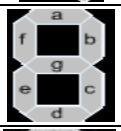
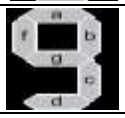


Fig 3.12

It is an alphanumeric display which can display digits from 0 to 9. Seven LEDS are placed in an organized manner. By controlling the current flow through each LED, some segments will be light and others will be dark.hence,the desired pattern will be displayed.It is in the form of (i) common anode type (ii) common cathode type.

It can be used to drive a seven segment decoder. It contains four input lines A,B,C,D and 7 output lines a to g.The number of output lines are more than number of input lines.So ,it is called as decoder.Its logic diagram consists of 4 NOT gates to produce complement inputs and 10 AND gates to display the digits 0 to 9. Among the 10 AND gates ,only one AND gate is enabled at a time which depends upon the applied data input shown fig 3.12

Truthtable

Decimal digit	**Pattern**	**Bcd code**				**Seven segment code**						
		D	**C**	**B**	**A**	**a**	**b**	**C**	**d**	**E**	**f**	**g**
0		**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**
1		**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**0**	**0**
2		**0**	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**
3		**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**
4		**0**	**1**	**0**	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**1**
5		**0**	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**	**1**
6		**0**	**1**	**1**	**0**	**1**	**0**	**1**	**1**	**1**	**1**	**1**
7		**0**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**
8		**1**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
9		**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**

3.11 Digital logic families

A group of compatible ICs with the same logic levels and supply voltages for performing various logic functions have been fabricated using a specific circuit configuration is called as digital family.

Two types

(i) unipolar

(ii) bipolar.

There are two types of bipolar operations namely

(i) Saturated-transistors in the IC are driven into saturation. - eg TTL, DTL.I^2L

(ii) Non-saturated- transistors not driven into saturation.
eg CMOS,NMOS,PMOS

Characteristics of digital ICs

1. Speed of operation
2. Power dissipation
3. Fan in
4. Fan out
5. Noise immunity
6. Package density

1. **speed of operation**

The speed operation of an IC expressed in terms of propagation delay

Propagation delay is defined as the time taken for the output of gate to change after the inputs have changed

The time difference between the application of input and appearance of output is also called as propagation delay

2. Power dissipation

Power dissipation is the nature of the power consumed by a logic gate when fully driven by all its inputs,it is expressed in milliwatts.

3. Fan in

The number of inputs connected to a gate is known as fan in of the gate.

5. Fan out

The maximum number of standard logic inputs that an output can drive reliably

6. Noise immunity

The circuit ability to tolerate noise without causing spurious changes in the output voltage is called noise immunity.

The quantitative measure of the noise immunity is called noise margin.

7. Packaged density

The number of devices that can be fabricated per unit area of a chip is called packaged density.

Transistor-transistor logic (TTL logic)

The basic TTL logic circuit is NAND gate shown fig 3.13

Q_1 is a multiple- emitter input transistor contains two emitter terminals.Each emitter acts like a diode.The overall circuit acts like a 2 input NAND gate.The output transistors Q_3 and Q_4 are connected in a totem pole arrangement(ie one npn transistor is connected in series with the another npn transistor.)

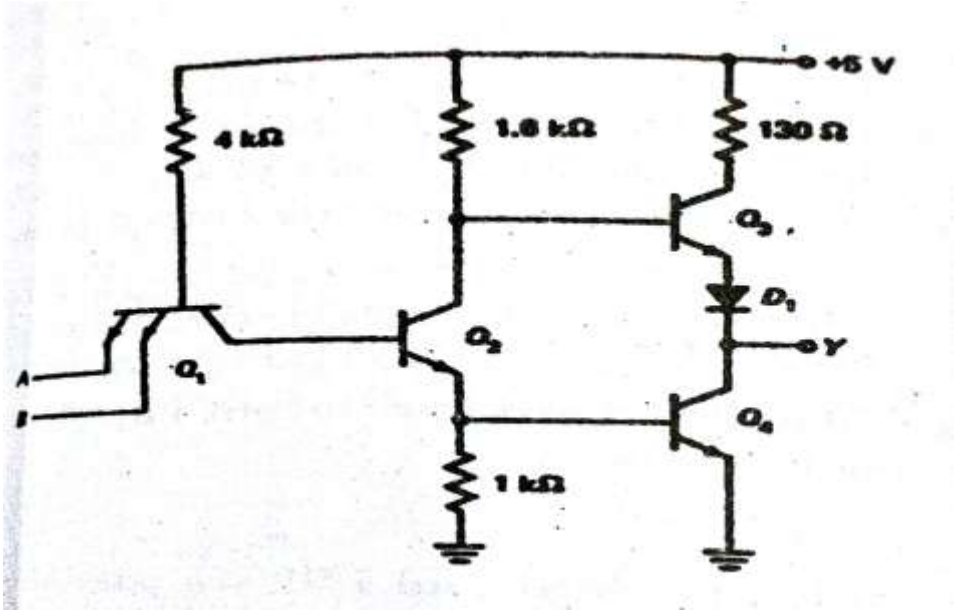


Fig 3.13

Input A	Input B	Output Y
0	0	1
0	1	1
1	0	1
1	1	0

With the totem pole connection,either Q_3 or Q_4 is on.When Q_3 is on, the output is high. When Q_4 is on, the output is low.

The input voltages A and B are either low or high.

If A or B is low, the base of voltage of Q_1 is reduced which inturn reduces the base voltage of Q_2. So , Q_2 is in cut off.Due to Q_2 open, Q_4 goes into cut off.Now the base voltage of Q_3 goes high. Q_3 act as a emitter follower.The output becomes high.

If A and B are both high , the emitter diodes of D_1 stop conducting and the collector diode goes into forward conduction. This forces the base voltage of Q_2 to high . It increases the saturation of Q_4.

Propagation delay = 10 ns

Fan in = 10

Fan out = 2

Noise immunity = Good

Advantages

1.compatibility with other ICs.

2.speed of operation is high.

3.Good noise immunity.

4.High level integraton are possible.

5.variety of circuit functions are possible.

Disadvantages

1.wired output capability not possible.

2.It generate switching transients.

CMOS LOGIC (complementary metal oxide semiconductor logic)

CMOS NAND gate consists of one N channel and P channel device connected in push-pull operation. Because of push-pull operation, one device is normally on and the other is normally off.

Q_1 and Q_2 form one complementary connection and Q_3 and Q_4 form another connection shown fig 3.14.

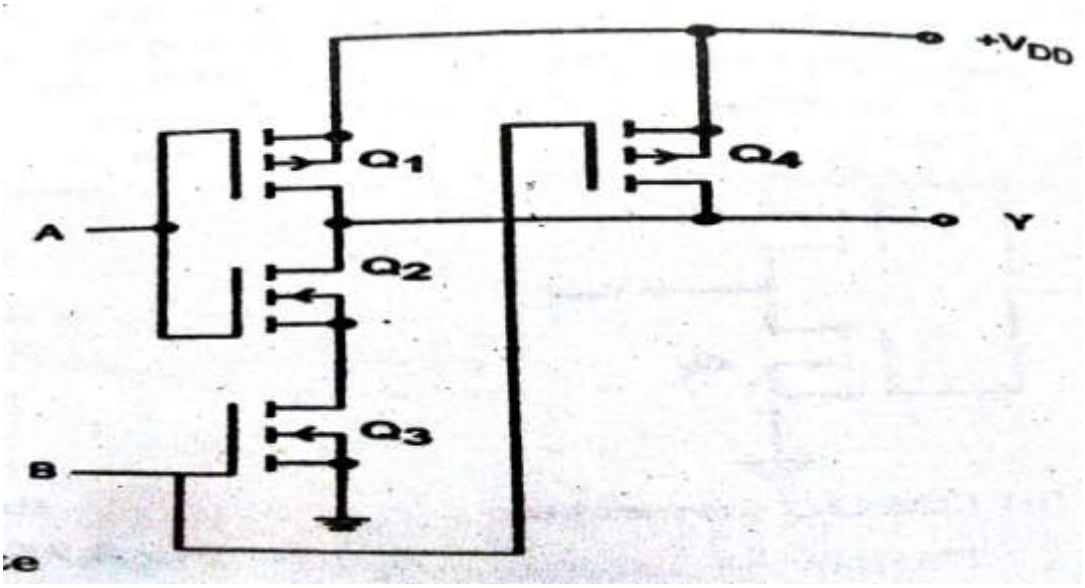


Fig 3.14

If A= low,it will close Q_1 and open Q_2

A= high,it will open Q_1 and close Q_2

B=low ,it will open Q_3 and close Q_4

B=high,it will close Q_3 and open Q_4

Operation

Case 1

A=0 and B=0, now Q_1 is closed and output goes high.

Case 2

A=0 and B=1,Since Ais low, Q_1 is closed and the output is high.

Case 3

A=1 and B=0, now Q_4 is closed and output goes high

Case4

A=1 and B=1,Since Ais low, Q_1 is closed and the output is low.

When the inputs are high, Q_2 and Q_3 are closed pulling the output to the ground.

Input A	Input B	Output Y
0	0	1
0	1	1
1	0	1
1	1	0

%%%%%%%%%%%%%%%%%%%%

REVIEW QUESTIONS

Two Mark

1	What is 1 s complement?
2	What is 2 s complement?
3	What is the use of 2 s complement?
4	What is signed number?
5	Give the rules of binary addition.
6	Give the rules of binary subtraction.
7	Find 1 s complement of 1001100.
8	Find 2 s complement of 10011.
9	What is half adder?
10	What is full adder?
11	What is parity bit?
12	What is decoder?
13	What is encoder?
14	What is demultiplexer?
15	What is multiplexer?
16	What is fan in?
17	What is fan out?

Three Mark

1	What is the concept of signed number?
2	Subtract 1011 from 1000 using 2 s complement method.
3	Draw the circuit diagram of half adder.
4	Draw the circuit diagram of full adder.
5	Give the truth tables of half adder.
6	Give the truth tables of full adder.
7	What is CMOS?

10 Mark

1	Explain half adder circuit with truth table.
2	Explain full adder circuit with truth table.
3	Explain half subtract or circuit with truth table.
4	Explain full subtract or circuit with truth table.
5	Explain 3 to 8 Decoder circuit with truth table.
6	Explain BCD to seven segment decoder with circuit and truth table.
7	Explain multiplexer with circuit and truth table.
8	Explain Demultiplexer with circuit and truth table.

UNIT – IV

SEQUENTIAL LOGIC

4.1. Flip=Flops

A ***combinational circuit*** is one where the output at any time depends only on the present combination of inputs at that point of time with total disregard to the past state of the inputs. The logic gate is the most basic building block of combinational logic. The logical function performed by a combinational circuit is fully defined by a set of Boolean expressions.

The other category of logic circuits, called ***sequential logic circuits***, comprises **both logic gates and memory elements such as flip-flops. Owing to the presence of memory elements, the output in a sequential circuit depends upon not only the present but also the past state of inputs. All the flip-flops are sequential logic circuits.**

Flip-flop is a bistable logic element with one or more inputs and two outputs. The outputs are complement to each other. A flip-flop can store one bit of binary data as 1 or 0. The Flip-flop are divided into the following types

i. RS flip-flop
ii. Clocked RS flip-flop
iii. JK flip-flop
iv. D flip-flop
v. T flip-flop

4.2.R-S Flip-Flop

The R-S flip-flop is the most basic of all flip-flops. The letters 'R' and 'S' here stand for RESET and SET. When the flip-flop is SET, its Q output goes to a '1' state, and when it is RESET it goes to a '0' state. The Q output is the complement of the $\bar{Q}$ output at all times.

A flip-flop, as stated earlier, **is a bistable circuit**. Both of its output states are stable. The circuit remains in a particular output state indefinitely until something is done to change that output status.

Referring to the bistable multivibrator circuit, these two states were those of the output transistor in saturation (representing a LOW output) and in cut-off (representing a HIGH output). If the LOW and HIGH outputs are respectively regarded as '0' and '1', then the output can either be a '0' or a '1'. Since either a '0' or a '1' can be held indefinitely until the circuit is appropriately triggered to go to the other state, the circuit is said to have memory.

It is capable of storing one binary digit or one bit of digital information. Also, if we recall the functioning of the bistable multivibrator circuit, we find that, when one of the transistors was in saturation, the other was in cut-off. This implies that, if we had taken outputs from the collectors of both transistors, then the two outputs would be complementary. In the flip-flops of various types that are available in IC form, we will see that all these devices offer complementary outputs usually designated as Q and Q_

The logic diagram and truth table of RS flip-flop are shown in figure. It is also called RS latch. It contains two inputs (R and S) and two outputs(Q and Q̄). Both outputs are complement to each other.

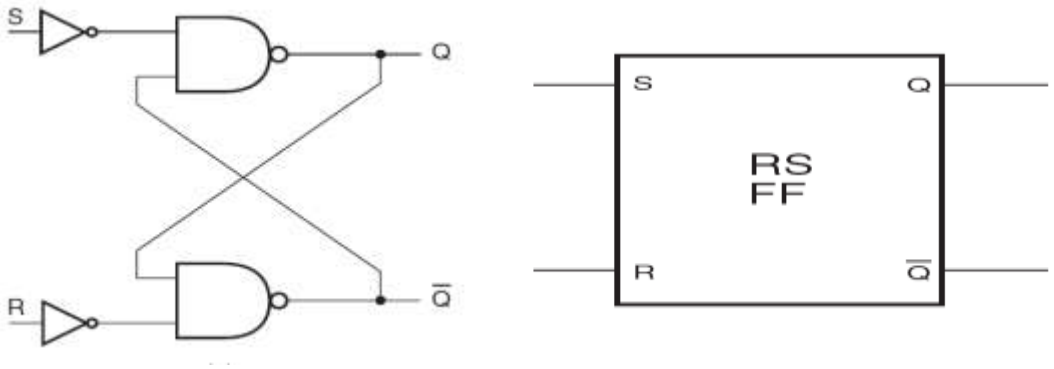


Fig 4.1

There are four conditions that exist in RS flip-flop in accordance with the input signals.

i.When R=0 and S=0

The input combination R = S = 0, the output would be previous state

ii.When R=0 and S=1

The input combination R = 0 and S=1, the output Q would be 1 (high) and other output would be 0(low). This condition is called 'set".

Input		Output		Condition
R	S	Q	Q̄	
0	0	Qn-1	Q̄n-1	Not used
0	1	1	0	Set
1	0	0	1	Reset
1	1	Forbidden state		No change

Truth table 4.1

iii.When R=1 and S=0

The input combination R = 1 and S=0, the output Q would be 0 (low) and other output would be 1(High). This condition is called 'Reset".

iv.When R=1 and S=1

The Q and Q̄ outputs both remains in logic 1 state . There is a race between Q and Q̄ outputs. This is called race condition.And this case should be eliminated. Hence it is called Forbidden state.

Clocked* R-S *Flip-Flop

In the case of a clocked R-S flip-flop, the outputs change states as per the inputs only on the occurrence of a clock pulse. The clocked flip-flop could be a level-triggered one or an edge-triggered one.

The basic flip-flop is the same as that shown in Fig. The two NAND gates at the input have been used to couple the R and S inputs to the flip-flop inputs under the control of the clock signal.

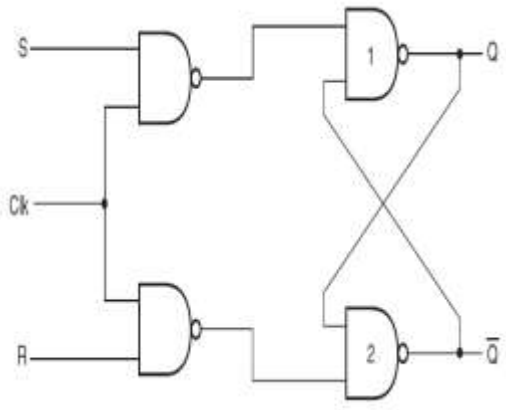


Fig 4.2

When the clock signal is HIGH, the two NAND gates are enabled and the S and R inputs are passed on to flip-flop inputs with their status complemented. The outputs can now change states as per the status of R and S at the flip-flop inputs. Refer **Truth table 4.2**

i.When R=0 and S=0

The input combination **R = S = 0,** the output would be previous state.

ii.When R=0 and S=1

For instance, when **R = 0 and S = 1** it will be passed on as 0 and 1 respectively when the clock is HIGH. When the clock is LOW, the two NAND gates produce a '1' at their outputs,

Clock	Input		Output		Condition
	R	S	Q	$\bar{Q}$	
0					No change
1	0	0	Forbidden		No change
1	0	1	1	0	Set
1	1	0	0	1	Reset
1	1	1	Not permissible		Invalid

Truth table 4.2

iii.When R=1 and S=0

The input combination **R= 1 and S=0,** the output Q would be 0 (low) and other output would be 1(high).

iv.When R=1 and S=1

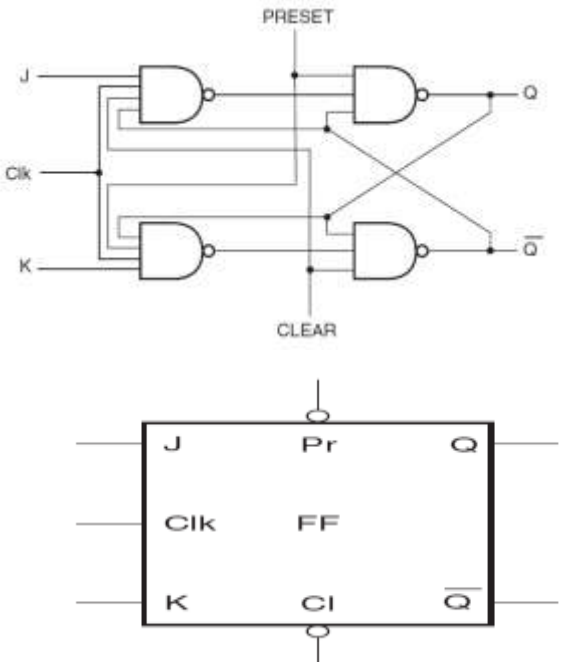
The input combination **R = S = 1** would be forbidden as SET and RESET inputs in an R-S flip-flop cannot be active at the same time

Fig 4.3

4.3.J-K *Flip-Flop with PRESET and CLEAR Inputs*

In the Rs flip-flop and clocked Rs flip-flop if the inputs R and S go to 1 simultaneously the outputs are indeterminant. This ambiguity is removed in the JK flip-flop.

It is often necessary to clear a flip-flop to a logic '0' state (Qn = 0) or preset it to a logic '1' state (Qn =1). An example of how this is realized is shown in Fig.. 4.3



The **flip-flop is cleared** (that is, Qn = 0) whenever the **CLEAR input is '0'** and the PRESET input is '1'.

The **flip-flop is preset** to the logic '1' state whenever **the PRESET input is '0'** and the CLEAR input is '1'. Here, the CLEAR and PRESET inputs are active LOW. Figure shows the circuit symbol ofthis presettable, clearable, clocked J-K flip-flop.Figure shows the fnctiontable of such a flip-flop. Refer **Truth table 4.3**

i.When J=0 and K=0

The input combination K = J = 0, the output would be previous state.

ii.When J=0 and K=1

The input combination J = 0 and K=1, the output Q would be 0 (low) and other output would be 1(high).

Clock	Input		Output		Condition
	J	K	Q	$\bar{Q}$	
1	0	0	Qn-1	Qn-1	No change
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	$\overline{Qn-1}$	Qn-1	Toggle

Truth table 4.3

iii.When J=1 and K=0

For instance, when J = 1 and K = 0 it will be passed on as 0 and 1 respectively when the clock is HIGH. When the clock is LOW, the two NAND gates produce a '1' at their outputs,

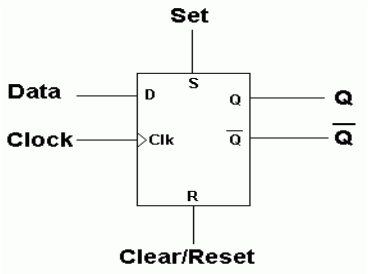
iv.When J=1 and K=0

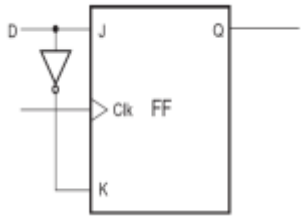
The input combination J = K = 1 would be Toggle (That is changes output for every clock pulse).

4.,4.D Flip-flop(delay flip-flop or Data flip-flop)

The D flip-flop is a Delay flip-flop or data flip-flop. It contains one input and two outputs.. The symbol connection diagram and truth table are shown in figure. In order to convert JKMS flip-flop into D flip-flop a NOT gate is connected in between J and K inputs.

In a D flip-flop, the data on the D input are transferred to the Q output on the positive- or negative-going transition of the clock signal, depending upon the flip-flop, and this logic state is held at the output until we get the next effective clock transition.





Clock	Inputs	Output		Condition
	D	Q	$\bar{Q}$	
1	1	1	0	Set
1	0	0	1	Reset

Fig 4.4 **Truth table 4.4**

When D=1: which makes the inputs as **J=1 and K=0.** At the application of negative edge of the clock input, the output of flip-flop will become **Q=1and Q-0(Set)**

When D=0: which makes the inputs as **J=0 and K=1.** At the application of negative edge of the clock input, the output of flip-flop will become **Q=0and Q=1(Reset)**

This flip-flop is a single bit memory unit used to store single bit binary data.

Flip-Flop Applications

Flip-flops are used in a variety of application circuits, the most common among these being the ***frequency division and counting*** circuits and ***data storage and transfer*** circuits. Both these applications use a cascaded arrangement of flip-flops with or without some additional combinational logic to perform the desired function. Counters and registers are available in IC form for a variety of digital circuit applications.

Other applications of flip-flops include their use for switch **debouncing,** where even an unclocked flip-flop (such as a NAND or a NOR latch) can be used, for synchronizing asynchronous inputs with the clock input and for identification of edges of synchronous inputs.

4.5.Toggle Flip-Flop (*T* Flip-Flop)

T-flip-flop contains one input and two outputs. The output of a *toggle flip-flop*, also called a T flip-flop, changes state every time it is triggered at its T input, called the toggle input. That is, the output becomes ‘1’ if it was ‘0’ and ‘0’ if it was ‘1’. Figures 4.5 respectively show the circuit symbols of positive edge-triggered and negative edge-triggered T flip-flops, along with their function tables. Refer **Truth table 4.4**

In order to convert JKMS flip-flop into T flip-flop, the J and K inputs are connected together. when both J and K inputs of the flip-flop are tied to their active level (‘1’ level if J and K are active when HIGH, and ‘0’ level when J and K are active when LOW), the flip-

flop behaves like a toggle flip-flop, with its clock input serving as the T input. Figure shows the use of a J-K flip-flop as a T flip-flop.

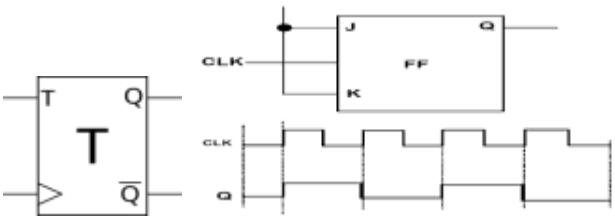


Fig 4.5

Clock	Input	Output		Condition
	T	**Qn**	$\bar{Q}$n	
1	1	Q$\bar{n}$-1	$\bar{Q}$n-1	Set
1	0	Qn-1	$\bar{Q}$n-1	Reset

Truth table 4.4

When T=0: which makes the inputs as **J=0 and K=0.** At the application of clock input, the output of flip-flop will become **No change.**

When T=1: which makes the inputs as **J=1 and K=1.** At the application of the clock input, the output of flip-flop will become **Toggle**.

The input and output waveforms at T=1 condition are shown in figure above. It is called divide by 2 counter because the output signal frequency is one half (1/2) of the input clock signal frequency.

4.6.Master–Slave Flip-Flops

Whenever the width of the pulse clocking the flip-flop is greater than the **propagation delay** of the flip-flop, the change in state at the output is not reliable. In the case of edge-triggered flip-flops, this pulse width would be the trigger pulse width generated by the edge detector portion of the flip-flop and not the pulse width of the input clock signal. This phenomenon is referred to as the ***race problem***. As the propagation delays are normally very small, the likelihood of the occurrence of a race condition is reasonably high. One way to get over this problem is to use a ***master–slave*** **configuration**.

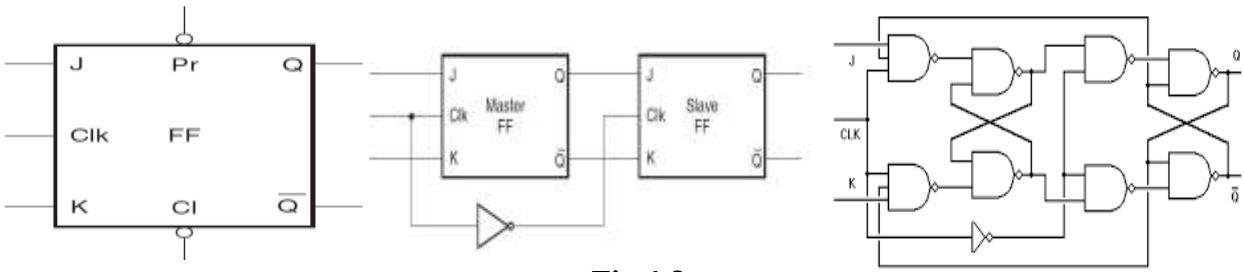


Fig 4.6

Truth table 4.5

Clock	Input		Output		Condition
	J	K	Q	$\bar{Q}$	
1	0	0	Qn-1	$\overline{Qn-1}$	No change
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	$\overline{\text{Qn-1}}$	Qn-1	Toggle

Figure 4.6 shows a master–slave flip-flop constructed with two J-K flip-flops. The first flip-flop is called the **master flip-flop** and the second is called the **slave**.

The clock to the slave flip-flop is the complement of the clock to the master flip-flop. When the clock pulse is present, the master flip-flop is enabled while the slave flip-flop is disabled.

As a result, the master flip-flop can change state while the slave flip-flop cannot. When the clock goes LOW, the master flip-flop gets disabled while the slave flip-flop is enabled. Therefore, the slave J-K flip-flop changes state as per the logic states at its J and K inputs. The contents of the master flip-flop are therefore transferred to the slave flip-flop, and the master flip-flop, being disabled, can acquire new inputs without affecting the output. As would be clear from the description above, a master– slave flip-flop is a pulse-triggered flip-flop and not an edge-triggered one.

Refer **Truth table 4.5**

i.When J=0 and K=0

The input combination K = J = 0, the output would be previous state.

ii.When J=0 and K=1

The input combination J = 0 and K=1, the output Q would be 0 (low) and other output would be 1(high).

iii.When J=1 and K=0

For instance, when J = 1 and K = 0 it will be passed on as 0 and 1 respectively when the clock is HIGH. When the clock is LOW, the two NAND gates produce a '1' at their outputs,

iv.When J=1 and K=1

The input combination J = K = 1 would be Toggle (That is changes output for every clock pulse)

In fact, the J-K master slave flip-flop can be used to **construct any other flip-flop**. That is why it is also sometimes referred to as a ***universal flip-flop***.

4.7. Edge-Triggered Flip-Flops

In a ***level-triggered*** **flip-flop**, the output responds to the data present at the inputs during the time the clock pulse level is HIGH (or LOW). That is, any changes at the input during the time the clock is active (HIGH or LOW) are reflected at the output as per its function table.

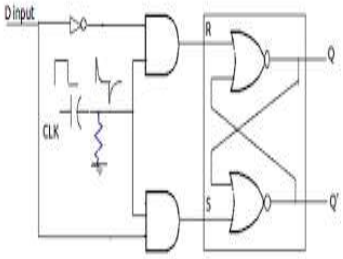
In an ***edge-triggered*** **flip-flop**, the output responds to the data at the inputs only on LOW-to-HIGH or HIGH-to-LOW transition of the clock signal.

The flip-flop in the two cases is referred to as positive edge triggered and negative edge triggered respectively. Any changes in the input during the time the clock pulse is HIGH (or LOW) do not have any effect on the output. In the case of an edge triggered flip-flop, an edge detector circuit transforms the clock input into a very narrow pulse that is a few nanoseconds wide.

This narrow pulse coincides with either LOW-to-HIGH or HIGH-to-LOW transition of the clock input, depending upon whether it is a positive edge-triggered flip-flop or a negative edge-triggered flip-flop. This pulse is so narrow that the operation of the flip–flop can be considered to have occurred on the edge itself.

4.7.1. Positive Edge-Triggered Flip-Flops

The circuit diagram of positive edge triggered flip-flop is shown in figure. The clock signal is applied to the clock input of the flip-flop through RC network. The RC network is operated as a differentiator. This RC network produces a continuous positive and negative spike signal at its output.



Clock	Input		Output		Condition
	J	K	Q	$\bar{Q}$	
-ve edge	X	X	Qn-1	$\overline{Qn-1}$	No change
+ve edge	0	0	Qn-1	$\overline{Qn-1}$	No change
+ve edge	0	1	0	1	Reset
+ve edge	1	0	1	0	Set
+ve edge	1	1	$\overline{Qn-1}$	Qn-1	Toggle

Fig 4.7 **Truth table 4.6**

When the clock input is low, this flip-flop will be in the inactive state and it will also retain its previous outputs. This flip-flop is triggered only at positive edge of the clock input. The output condition of this flip-flop is shown in the table.

4.8.Counters

A counter is a circuit used to count a repeated set of values, like clock pulses. In this case, the **counters** are mainly used in counting applications, where they either measure the time interval between two unknown time instants or measure the frequency of a given signal.

A flip-flop can store one bit of information either 1 or 0. A group of cascaded flip-flops called registers is required to store binary information. In counter flip-flop are connected in cascaded manner. Generally the JKMS flip-flop, T flip-flop and D flip-flop are used in counter.

Counters can be classified into two types. They are

i. **Asynchronous Counter or ripple counter or serial counter**
ii. **Synchronous Counter or parallel counter**

In a serial counter each flip-flop is triggered by the previous flip-flop and thus the counter has a cumulative settling time. In synchronous counter the flip-flop are triggered by a single clock pulse simultaneously.

Classification of counters

(i) **Asynchronous Counter** A *ripple counter serial counter*, is a cascaded arrangement of flip-flops where the output of one flip-flop drives the clock input of the following flip-flop. The number of flip-flops in the cascaded rrangement depends upon the number of different logic states that it goes through before it repeats the sequence, a parameter known as the modulus of the counter.

(ii) **Synchronous Counter** In a *synchronous counter*, also known as a *parallel counter*, all the flip-flops in the counter change state at the same time in synchronism with the input clock signal. The clock signal in this case is simultaneously applied to the clock inputs of all the flip-flops. The delay involved in this case is equal to the propagation delay of one flip-flop only, irrespective of the number of flip-flops used to construct the counter. In other words, the delay is independent of the size of the counter.

Modulus of a Counter

The *modulus* (MOD number) of a counter is the number of different logic states it goes through before it comes back to the initial state to repeat the count sequence. An n-bit counter that counts through all its natural states and does not skip any of the states has a modulus of 2^n

1. We can see that such counters have a modulus that is an integral power of 2, that is, 2, 4, 8, 16 and so on. These can be modified with the help of additional combinational logic to get a modulus of less than 2^n
2. To determine the number of flip-flops required to build a counter having a given modulus, identify the smallest integer m that is either equal to or greater than the desired modulus and is also equal to an integral power of 2.
3. For instance, if the desired modulus is 10, which is the case in a decade counter, the smallest integer greater than or equal to 10 and which is also an integral power of 2 is 16. The number of flip-flops in this case would be 4, as 16 = 24.
4. On the same lines, the number of flip-flops required to construct counters with MOD numbers of 3, 6, 14, 28 and 63 would be 2, 3, 4, 5 and 6 respectively.
5. In general, the arrangement of a minimum number of N flip-flops can be used to construct any counter with a modulus given by the equation.

$$(2^{N-1} + 1) \leq modulus \leq 2^N$$

4 .8.1 Asynchronous Binary Counter (Ripple Counter)

The operation of a binary ripple counter can be best explained with the help of a typical counter of this type. Figure shows a four-bit ripple counter implemented with negative edge-triggered *J-K* flip-flops wired as toggle flip-flops.

The output of the first flip-flop feeds the clock input of the second, and the output of the second flip-flop feeds the clock input of the third, the output of which in turn feeds the clock input of the fourth flip-flop. The outputs of the four flip-flops are designated as Q0 (LSB flip-flop), Q1, Q2 and Q3 (MSB flip-flop).

Figure shows the waveforms appearing at Q0, Q1, Q2 and Q3 outputs as the clock signal goes through successive cycles of trigger pulses. The counter functions as follows.

Let us assume that all the flip-flops are initially cleared to the '0' state.

On HIGH-to-LOW transition of the first clock pulse, Q0 goes from '0' to '1' owing to the toggling action. As the flip-flops used are negative edge-triggered ones, the '0' to '1' transition of Q0 does not trigger flip-flop FF1. FF1, along with FF2 and FF3, remains in its '0' state. So, on the occurrence of the first negative-going clock transition, Q0 = 1, Q1 = 0, Q2 = 0 and Q3 = 0.

On the HIGH-to-LOW transition of the second clock pulse, Q0 toggles again. That is, it goes from '1' to '0'. This '1' to '0' transition at the Q0 output triggers FF1, the output Q1 of which goes from '0' to '1'. The Q2 and Q3 outputs remain unaffected.

Therefore, immediately after the occurrence of the second HIGH-to-LOW transition of the clock signal, Q0 = 0, Q1 = 1, Q2 = 0 and Q3 = 0.

On similar lines, we can explain the logic status of Q0, Q1, Q2 and Q3 outputs immediately after subsequent clock transitions. The logic status of outputs for the first 16 relevant (HIGH-to-LOW in the present case) clock signal transitions is summarized in Table.

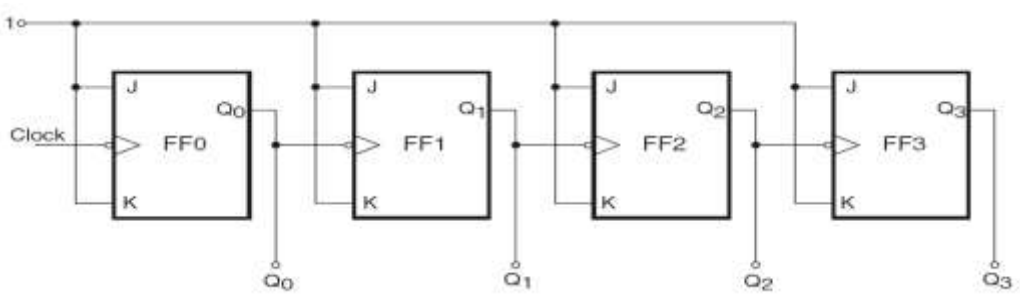
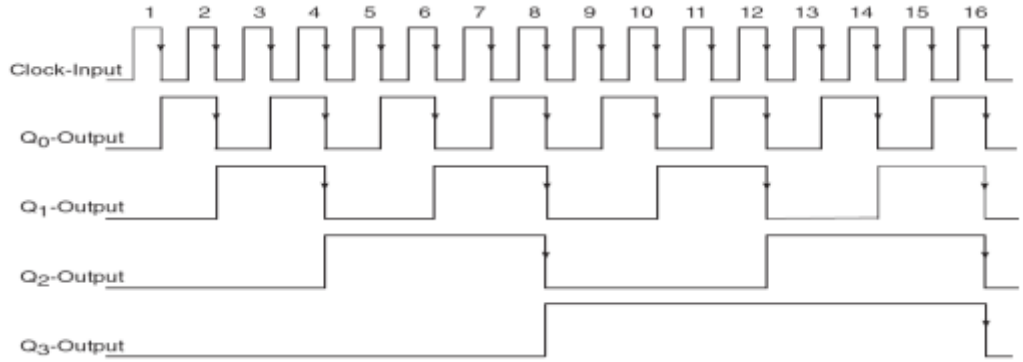


Fig 4.8



Output logic states for different clock signal transitions for a four-bit binary ripple counter.

Clock	Q3	Q2	Q1	Q0
Reset	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Truth table 4.7

Thus, we see that the counter goes through 16 distinct states from 0000 to 1111 and then, on the occurrence of the desired transition of the sixteenth clock pulse, it resets to the original state of 0000 from where it had started. In general, if we had N flip-flops, we could count up to 2N pulses before the counter resets to the initial state. We can also see from the Q0, Q1, Q2 and Q3 waveforms, as shown. in Fig. 11.2(b), that the frequencies of the Q0, Q1, Q2 and Q3 waveforms are f/2,f/4, f/8 and f/16 respectively. Here, f is the frequency of the clock input.

In the case of a four-bit counter of the type shown in Fig. 11.2(a), outputs are available at f/2 from the Q0 output, at f/4 from the Q1 output, at f/8 from the Q2 output and at f/16 from the Q3 output. It may be noted that frequency division is one of the major applications of counters.

Decade Counter (BCD Counter)

A *decade counter* is one that goes through 10 unique output combinations and then resets as the clock proceeds further. Since it is an MOD-10 counter, it can be constructed with a minimum of four flip-flops. A four-bit counter would have 16 states.

By skipping any of the six states by using some kind of feedback or some kind of additional logic, we can convert a normal four-bit binary counter into a decade counter. A decade counter does necessarily count from 0000 to 1001.

A decade counter is otherwise called divide by 10^{th} counter, mod 10 counter or BCD counter. The logic diagram of decade counter is shown in figure. 4.9

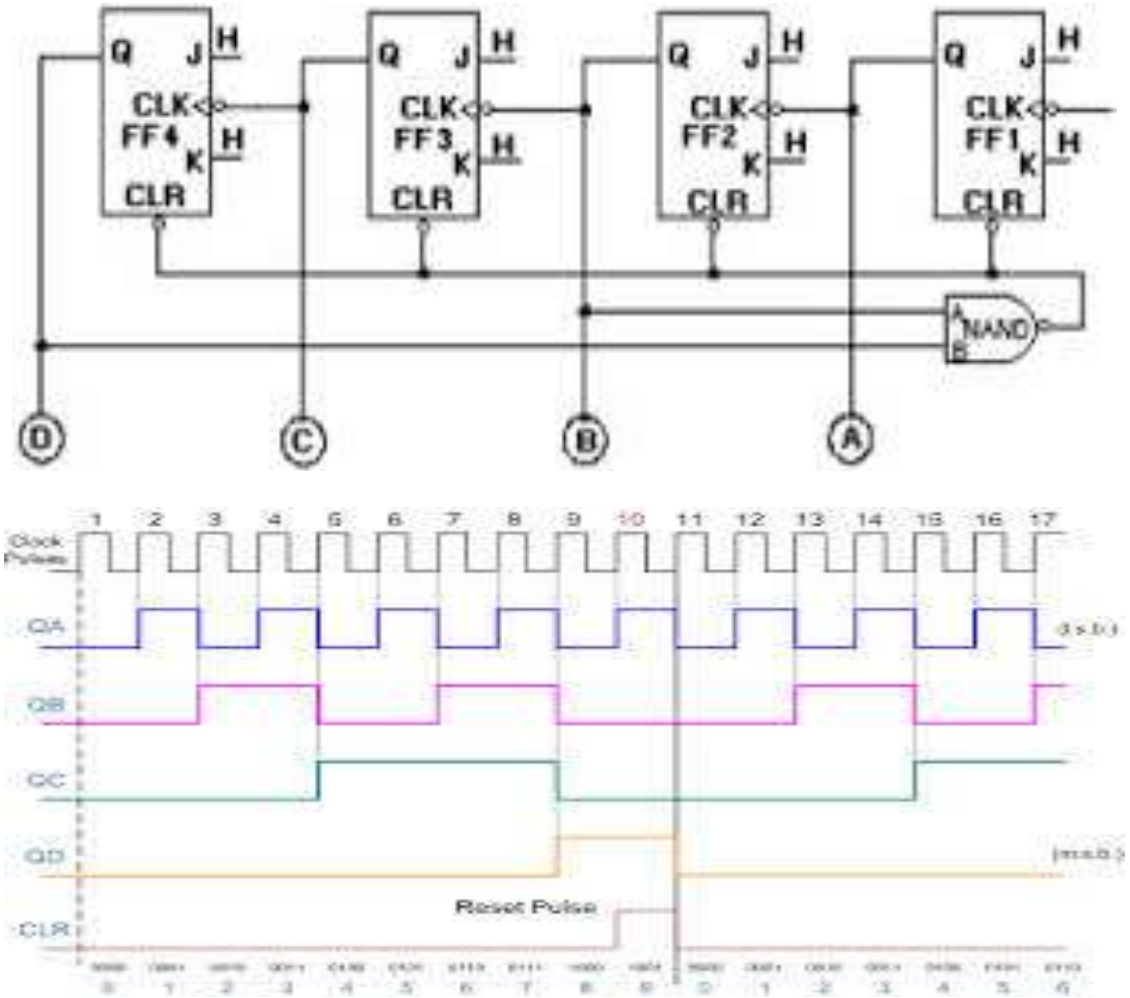


Fig 4.9

lock	Q_D	Q_C	Q_B	Q_A
Reset	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	0	0	0	0
11	0	0	0	1

Truth table 4.8

The counter is reset at 10^{th} clock pulse by using two input NAND gate. At the tenth clock pulse, an ordinary four bit counter produces the count **as $Q_DQ_CQQ_BQ_{A=1010}$**.

That means the outputs of Q_D and Q_B are only in high level at the 10^{th} clock pulse, before that it is not possible. Hence the output terminals of Q_D and Q_B are connected to the input of two input NAND gate and its output is also connected to the reset terminal**.**

This counter operates similar as a 4 bit ripple counter while counting from 0000 to 1001. On the 10^{th} clock pulse the output goes to 1010. That means $Q_{D=}$ $Q_{B=1}$. This makes the

output of NAND gate as 0. Immediately the reset terminal makes the output of the counter as $Q_DQ_CQQ_BQ_A$=0000.

MOD-N/Divide-by-N Counters

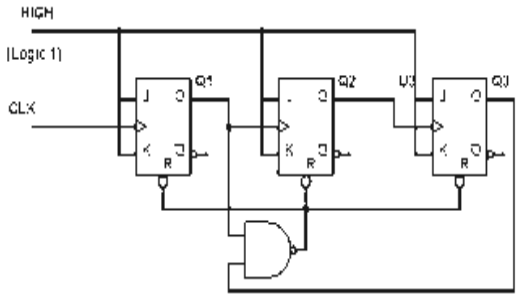
A counter which is reset at the nth clock pulse is called Mod n counter or divide by n counter. Logic gates that are connected externally make to reset the counter at the nth clock pulse.

Normal binary counter counts from 0 to 2^N - 1, where N is the number of bits/flip-flops in the counter. In some cases, we want it to count to numbers other than 2^N - 1. This can be done by allowing the counter to skip states that are normally part of the counting sequence. There are a few methods of doing this. One of the most common methods is to use the CLEAR input on the flip-flops.

An extra NAND gate is generally used for making divide by n counter. For this first we must know, which flip-flop outputs are actually in high (1) level in the nth clock pulse. Then connect the outputs of that flip-flop to the input terminal of NAND gate. The output terminal of the NAND gate is connected to the reset terminal of the counter. Thus at the nth clock pulse, all the inputs connected to the NAND gate are high, makes the output as 0 and to reset the counter. After applying the next clock pulse the counter will start the count from its initial level. The number of flip-flop needed for this counter depends upon the n value.

Modulo 5 counter

A counter which is reset at the fifth clock pulse is called mod 5 counters or divides by 5 counters. The circuit diagram of mod 5 counter is shown in figure. The counter contains three flip-flops shown fig 4.10



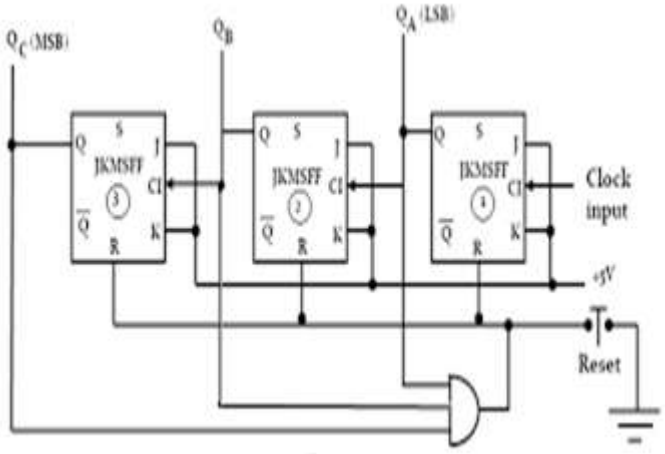
Clock	Q_3	Q_2	Q_1
Reset	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	0	0	0
7	0	0	1

Fig 4.10 **Truth table 4.9**

A 3 bit n binary counter is normally counting from 000 to111. The actual output of 3 bit counter at the 5^{th} clock pulse is 101. A two input NAND gate is used to make a MOD 5 counter. The output of first and third flip-flop is connected to the\\input of the NAND gate and its output is connected to the reset terminal of the counter. In normal counting sequences from 00 to 100 the output of NAND gate is high. At the 5^{th} clock pulse the counter value of this counter reaches 101. This value makes all the input of NAND gate as high level. It produces the output of NAND gate as zero. Hence the counter is reset at the 5^{th} clock pulse, which produces at he output as 000

Modulo 7 counter

A counter which is reset at the 7th clock pulse is called mod 7 counters or divides by 7 counters. The circuit diagram of mod 7 counter is shown in figure. The counter contains three flip-flops shown fig 4.11



Clock	Q_C	Q_B	Q_A
Reset	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	0	0	0
8	0	0	1

Fig 4.11 **Truth table 4.10**

A 3 bit n binary counter is normally counting from 000 to111. The actual output of 3 bit counter at the 7th clock pulse is 111. A two input NAND gate is used to make a MOD 7 counter. The output of all the flip-flop is connected to the\\input of the NAND gate and its output is connected to the reset terminal of the counter. In normal counting sequences from 00 to 110 the output of NAND gate is high. At the 7th clock pulse the counter value of this counter reaches 111. This value makes all the input of NAND gate as high level. It produces the output of NAND gate as zero. Hence the counter is reset at the 7th clock pulse, which produces at the output as 000

UP/DOWN Counters

Counters are also available in integrated circuit form as UP/DOWN counters, which can be made to operate as either UP or DOWN counters., An **UP** counter is one that counts upwards or in the forward direction by one LSB every time it is clocked. A four-bit binary UP counter will count as 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111, 0000, 0001, --- and so on.

A **DOWN** counter counts in the reverse direction or downwards by one LSB every time it is clocked. The four-bit binary DOWN counter will count as 0000, 1111, 1110, 1101, 1100, 1011, 1010, 1001, 1000, 0111, 0110, 0101, 0100, 0011, 0010, 0001, 0000, 1111, --- and so on.

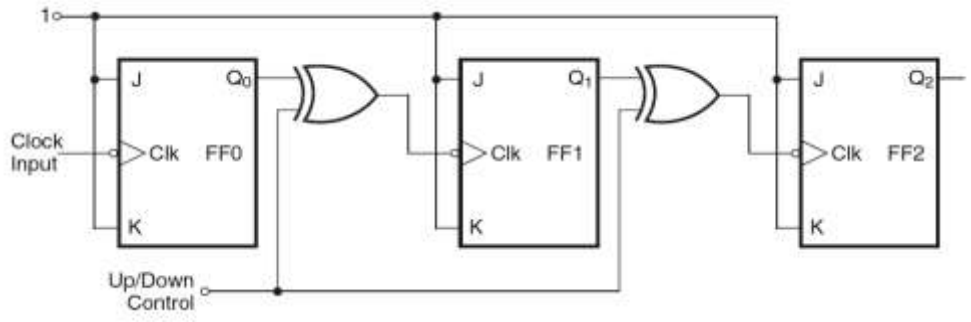


Fig 4.12

Figure shows a three-bit binary UP/DOWN counter. This is only one possible logic arrangement. As we can see, the counter counts upwards when **UP control is logic '1' and DOWN control is logic '0'**. In this case the clock input of each flip-flop other than the LSB flip-flop is fed from the normal output of the immediately preceding flip-flop.

The counter counts downwards when the **UP control input is logic '0' and DOWN control is logic '1'**. In this case, the clock input of each flip-flop other than the LSB flip-flop

is fed from the complemented output of the immediately preceding flip-flop. Figure shows another possible configuration for a three-bit binary ripple UP/DOWN counter. It has a common control input. When this input is in logic '1' state the counter counts downwards, and when it is in logic '0' state it counts upwards.

Presettable Counters

Presettable counters are those that can be preset to any starting count either asynchronously (independently of the clock signal) or synchronously (with the active transition of the clock signal). The presetting operation is achieved with the help of PRESET and CLEAR (or MASTER RESET) inputs available on the flip-flops.

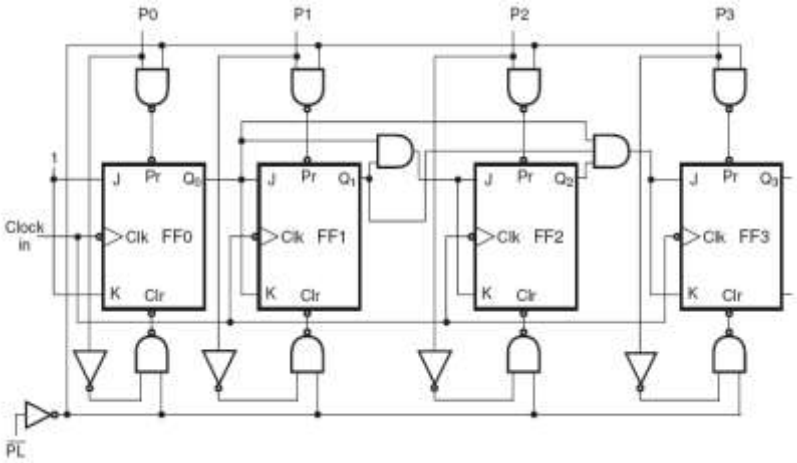


Fig 4.13

The presetting operation is also known as the 'preloading' or simply the 'loading' operation.

Presettable counters can be UP counters, DOWN counters or UP/DOWN counters. Additional inputs/outputs available on a presettable UP/DOWN counter usually include PRESET inputs, from where any desired count can be loaded, parallel load (*PL*) inputs, which when active allow the PRESET inputs to be loaded onto the counter outputs, and terminal count (*TC*) outputs, which become active when the counter reaches the terminal count.

Figure 4.13 shows the logic diagram of a four-bit presettable synchronous UP counter. An ordinary four bit counter count the value from 0000 to 1111. In a 4 bit presetable counter, we can be able to start the counting value from any intermirttent level in between 0000 to 1111. The needed intermittent value is set by using the loading facilities.
The data available on P3, P2, P1 and P0 inputs are loaded onto the counter when the parallel load **eeee**(PL_input goes LOW).

When the PL input goes LOW, one of the inputs of all NAND gates, including the four NAND gates connected to the PRESET inputs and the four NAND gates connected to the CLEAR inputs, goes to the logic '1' state. What reaches the PRESET inputs of FF3, FF2, FF1 and FF0 is **eeee**P3_ P2_ P1 and P0 respectively, and what reaches their CLEAR inputs is P3, P2, P1 and P0 respectively. Since PRESET and CLEAR are active LOW inputs, the counter flip-flops FF3, FF2, FF1 and FF0 will respectively be loaded with P3, P2, P1 and P0.

For example, if **P3 = 1,** the PRESET and CLEAR inputs of FF3 will be in the '0' and '1' logic states respectively. This implies that the **Q3 output will go to the logic '1' state**. Thus, FF3 has been loaded with P3.

Similarly, if **P3 = 0,** the PRESET and CLEAR inputs of flip-flop FF3 will be in the '1' and '0' states respectively. The flip-flop output (**Q3 output) will be cleared to the '0' state**. Again, the flip-flop is loaded with P3 logic status when the PL input becomes active.
Refer **Truth table 4.11**

Clock	Q_3	Q_2	Q_1	Q_0
Loading	0	1	1	0
1	0	1	1	1
2	1	0	0	0
3	1	0	0	1
4	1	0	1	0
5	1	0	1	1
6	1	1	0	0
7	1	1	0	1
8	1	1	1	0
9	1	1	1	1
10	0	0	0	0
11	0	0	0	1

Truth table 4.11

Suppose we ant to start the count from 0110, apply the input to the preset terminal and makes the PL_ input goes LOW. Now the output of the counter goes to 0110.

After that the first clock pulse makes the output as 0111., the second clock pulse makes the output 1000 and soon. In this manner the counter counts upward direction.

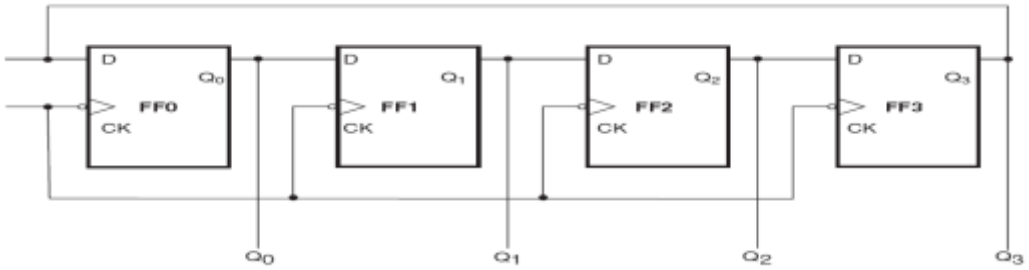
At the ninth clock pulse the output goes to 1111. The continuous 10^{th} pulse makes the output as 0000, reset the counter. After that if if the load line is nor enabled, the counter will start its counting sequence from 0001 and soon

Counter ICs 74190, 74191, 74192 and 74193 are asynchronously presettable synchronous UP/DOWN counters.

Ring Counter

A *ring counter* is obtained from a shift register by directly feeding back the true output of the lastt flip-flop to the data input terminal of the input flip-flop. If D flip-flops are being used to construct the shift register, the **ring counter, also called a circulating register,** can be constructed by feeding back the Q output of the last flip-flop back to the D input of the input flip-flop shown fig 4.14

If *J-K* flip-flops are being used, the Q and Q outputs of the output flip-flop are respectively fed back to the J and K inputs of the input flip-flop. Figure shows the logic diagram of a four-bit ring counter.



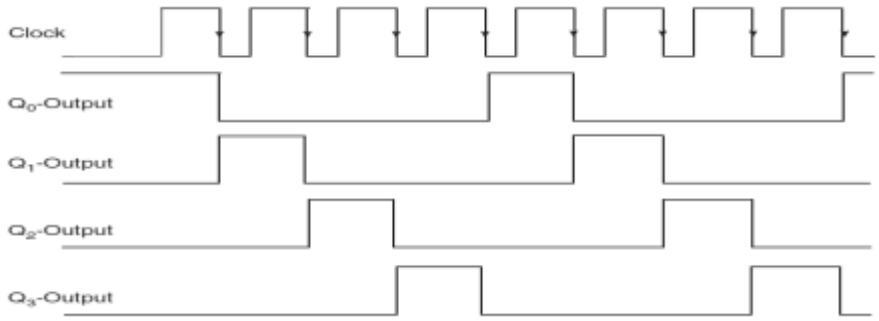


Fig 4.14

Let us assume that flip-flop FF0 is initially set to the logic '1' state and all other flip-flops are reset to the logic '0' state. The counter output is therefore **1000.**

With the **first clock pulse**, this '1' gets shifted to the second flip-flop output and the counter output becomes **0100.**

Similarly, with the second and third clock pulses, the counter output will become 0010 and 0001. With the fourth clock pulse, the counter output will again become 1000. The count cycle repeats in the subsequent clock pulses.

Clock	Q_0	**Q_1**	Q_2	Q_3
Initial	1	0	0	0
1	0	1	0	0
2	0	0	1	0
3	0	0	0	1
4	1	0	0	0
5	0	1	0	0

Truth table 4.12

Johnson counter (Shift Counter)

A shift counter on the other hand is constructed by having an inverse feedback in a shift register.

For instance, if we connect the Q output of the last flip-flop back to the K input of the first flip-flop and the Q_ output of the lastr flip-flop to the J input of the first flip-flop in a serial shift register, the result is a shift counter, also called a ***Johnson counter***. Figure shows the logic diagram of a basic four-bit shift counter.

Let us assume that the counter is initially reset to all 0s. With the first clock cycle, the outputs will become 1000. With the second, third and fourth clock cycles, the outputs will respectively be 1100, 1110 and 1111. The fifth clock cycle will change the counter output to 0111. The sixth, seventh and eighth clock pulses successively change the outputs to 0011, 0001 and 0000. Thus, one count cycle is completed in eight cycles.

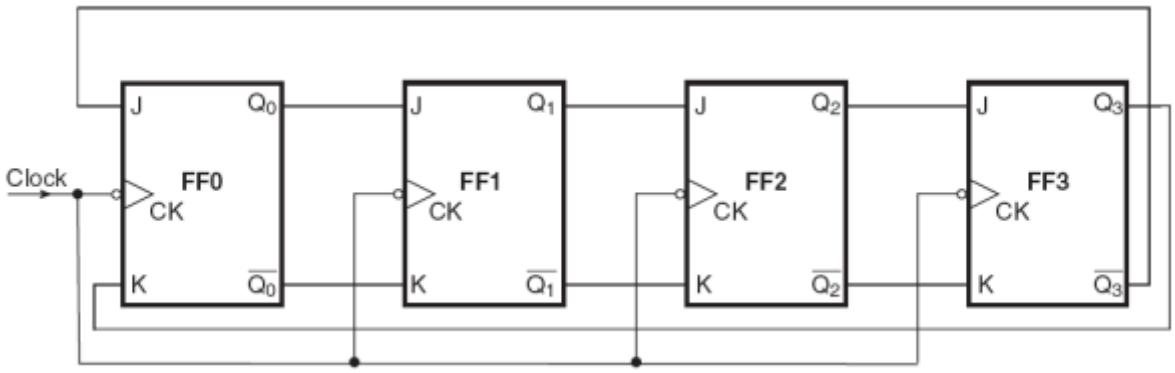
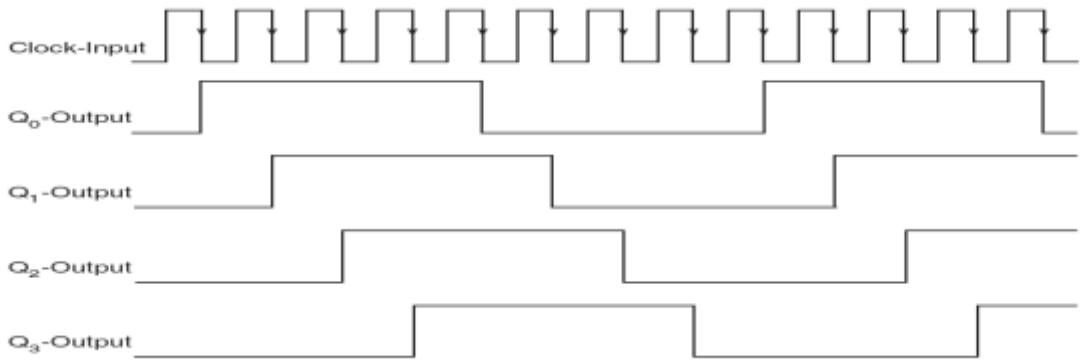


Fig 4.15



Clock	Q_3	Q_2	Q_1	Q_0
Reset	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
8	0	0	0	0
9	1	0	0	0

Truth table 4.13

Let us assume that the counter is initially reset to all 0s. With the first clock cycle, the outputs will become 1000. With the second, third and fourth clock cycles, the outputs will respectively be 1100, 1110 and 1111.

The fifth clock cycle will change the counter output to 0111. The sixth, seventh and eighth clock pulses successively change the outputs to 0011, 0001 and 0000. Thus, one count cycle is completed in eight cycles.

4.8.2 .Synchronous (or Parallel) Counters

Ripple counters discussed thus far in this chapter are asynchronous in nature as the different flipflops comprising the counter are not clocked simultaneously and in synchronism with the clock pulses.

The total propagation delay in such a counter is equal to the sum of propagation delays due to different flip-flops. The propagation delay becomes prohibitively large in a ripple counter with a large count. On the other hand, in a synchronous counter, all flip-flops in the counter are clocked simultaneously in synchronism with the clock, and as a consequence all flip-flops change state at the same time. The propagation delay in this case is independent of the number of flip-flops used.

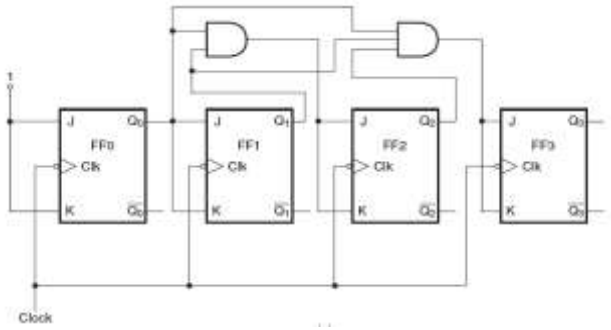


Fig 4.16

Since the different flip-flops in a synchronous counter are clocked at the same time, there needs to be additional logic circuitry to ensure that the various flip-flops toggle at the right time.

For instance, if we look at the count sequence of a four-bit binary counter shown in Table we find that flip-flop **FF0 toggles with every clock pulse, flip-flop FF1 toggles only**

when the output of FF0 is in the '1' state, flip-flop FF2 toggles only with those clock pulses when the outputs of FF0 and FF1 are both in the logic '1' state and flip-flop FF3 toggles only with those clock pulses when Q0_Q1 and Q2 are all in the logic '1' state. Such logic can be easily implemented with AND gates.

Figure 4.16 shows the schematic arrangement of a four-bit synchronous counter. The timing waveforms are shown in Fig. The diagram is self-explanatory.

A synchronous counter that counts in the **reverse or downward** sequence can be constructed in a similar manner by using complementary outputs of the flip-flops to drive the J and K inputs of the following flip-flops. Refer to the reverse or downward count sequence as given in Table. As is evident from the table, **FF0 toggles with every clock pulse, FF1 toggles only when Q0 is logic '0', FF2 toggles only when both Q0 and Q1 are in the logic '0' state and FF3 toggles only when Q0, Q1 and Q2 are in the logic '0' state.**

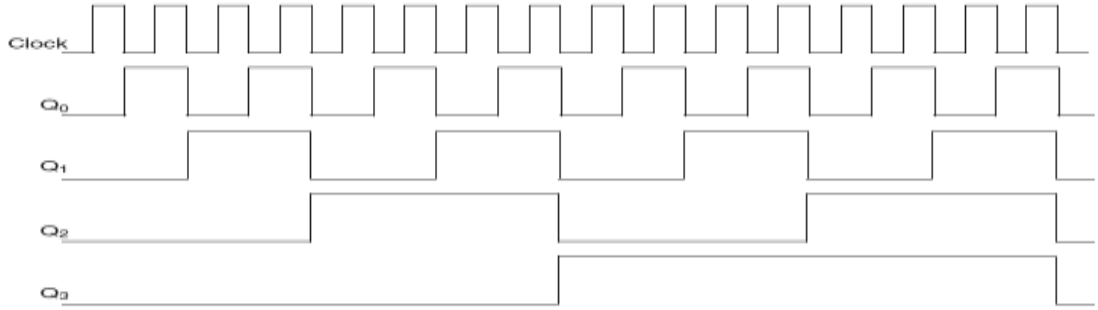


Fig 4.17

Clock	Q3	Q2	Q1	Q0
Reset	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Truth table 4.14

State Diagram

State Tables and State Diagrams

The relationship that exists among the inputs, outputs, present states and next states can be specified by either the **state table** or the **state diagram**.

State Table

The state table representation of a sequential circuit consists of three sections labelled ***present state, next state*** **and** ***output*****.** The present state designates the state of flip-flops before the occurrence of a clock pulse. The next state shows the states of flip-flops after the clock pulse, and the output section lists the value of the output variables during the present state.

State Diagram

In addition to graphical symbols, tables or equations, flip-flops can also be represented graphically by a state diagram. In this diagram, a state is represented by a circle, and the transition between states is indicated by directed lines (or arcs) connecting the circles. An example of a state diagram is shown in Figure 4.18 below

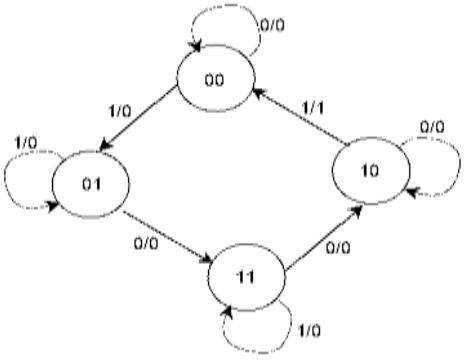


Fig 18

The binary number inside each circle identifies the state the circle represents. The directed lines are labelled with two binary numbers separated by a slash (/). The input value that causes the state transition is labelled first. The number after the slash symbol / gives the value of the output. For example, the directed line from state 00 to 01 is labelled 1/0, meaning that, if the sequential circuit is in a present state and the input is 1, then the next state is 01 and the output is 0. If it is in a present state 00 and the input is 0, it will remain in that state. A directed line connecting a circle with itself indicates that no change of state occurs. The state diagram provides exactly the same information as the state table and is obtained directly from the state table.

4.9 Shift Register

A shift register is a digital device used for storage and transfer of data.

The data to be stored could be the data appearing at the output of an encoding matrix before they are fed to the main digital system for processing or they might be the data present at the output of a microprocessor before they are fed to the driver circuitry of the output devices. The basic building block in all shift registers is the flipflop, mainly a ***D-type flip-flop***. forms an important link between the main digital system and the input/output channels.

The storage capacity of a shift register equals the total number of bits of digital data it can store, which in turn depends upon the number of flip-flops used to construct the shift register. Since each flip-flop can store one bit of data, the storage capacity of the shift register equals the number of flip-flops used. As an example, the internal architecture of an eight-bit shift register will have a cascade arrangement of eight flip-flops.

Based on the method used to load data onto and read data from shift registers, they are classified as

1 serial-in serial-out (SISO) shift registers,
2 serial-in parallel-out (SIPO) shift registers,
3 parallel-in serial-out (PISO) shift registers and
4 parallel-in parallel-out (PIPO) shift registers.

Four bit shift register

A five bit shift register is shown in figure 4.19 All flip-flops are operated in synchronous mode because clock pulses are applied to all flip-flops simultaneously. The output of each flip-flop is connected to input of next flip-flop. The data is directly applied to the input of first flip-flop.

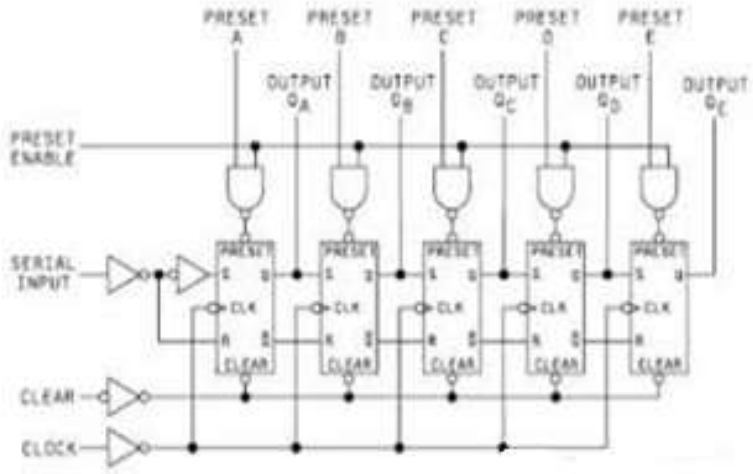


Fig 4.19

The serial input is applied to the serial input terminal, Q_E act as a serial output terminal, $Q_AQ_BQ_CQ_DQ_E$ act as parallel output terminals and ABCDE act as parallel input terminals. The five NAND gates have been provided for loading parallel five bit data into the register. Before loading the parallel input data, the registered must be cleared. A five bit shift register can store 5 single bit data. The four modes of shift register operation performed are explained below.

Serial-In Serial-Out Shift Register

The circuit functions as follows. A reset applied to the CLEAR input of all the flip-flops resets their Q outputs to 0s. The waveforms shown include the clock pulse train, the waveform representing the data to be loaded onto the shift register and the Q outputs of different flip-flops.

The shift register is a synchronous register because clock pulse is applied to all flip-flop simultaneously shown fig 4.20 The output of first flip-flop is connected to the input of second flip-flop and the output of second flip-flop is connected to the input of third flip-flop and soon. Data in is the serial input terminal and Data out is the serial output terminal.

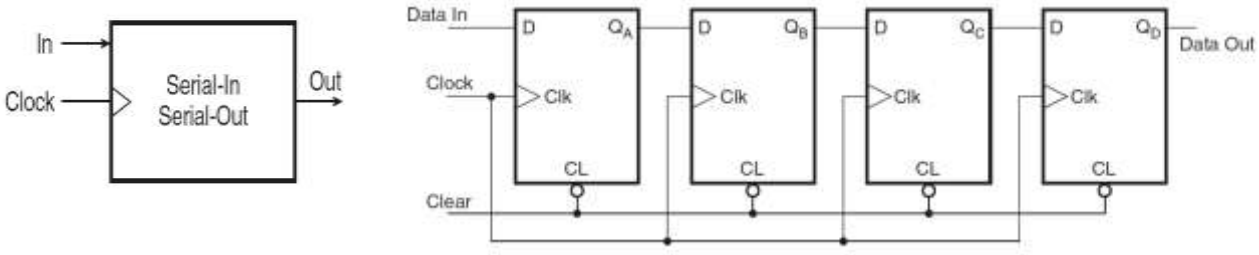


Fig 4.20

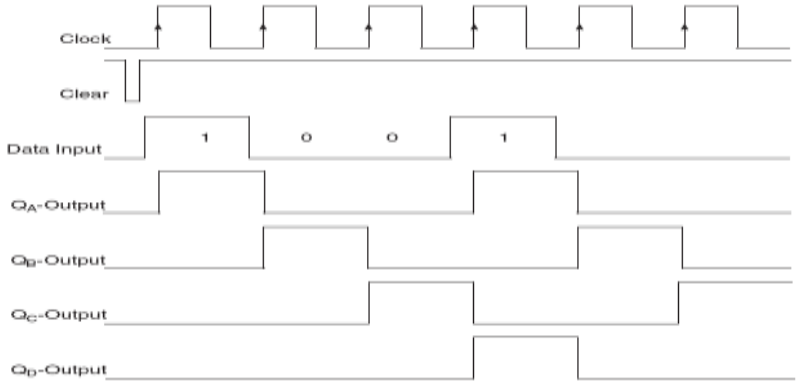


Fig 4.21

Write operation

Suppose four bit data of 1001 is written into its registers the following procedure may be followed.

1) Reset all the flip-flop
2) Apply the data input to Data in terminal one by one.
3) Clock pulse should be applied to the clock terminal after applying every data input

Clock input	Data in	Q_A	Q_B	Q_C	Q_D
Reset	-	0	0	0	0
+ve edge	1	1	0	0	0
+ve edge	0	0	1	0	0
+ve edge	0	0	0	1	0
+ve edge	1	1	0	0	1

Truth table 4.15

After applying the 4 bit data input with clock signals, all data are stored in the flip-flop.

The flip-flops shown respond to the LOW-to-HIGH transition of the clock pulses as indicated by their logic symbols.

During the first clock transition, the QA output goes from logic '0' to logic '1'.The outputs of the other three flip-flops remain in the logic '0' state as their D inputs were in the logic '0' state at the time of clock transition.

During the second clock transition, the QA output goes from logic '1' to logic '0' and the QB output goes from logic '0' to logic '1', again in accordance with the logic status of the D inputs at the time of relevant clock transition.

Thus, we have seen that a logic '1' that was present at the data input prior to the occurrence of the first clock transition has reached the QB output at the end of two clock transitions. This bit will reach the QD output at the end of four clock transitions.

Read operation

After write operation, we can be able to read the data by using Data out terminal. The clock pulse are needed for read operation. The procedure for read operation is shown below

Truth table 4.16

Clock input	Q_A	Q_B	Q_C	Q_D	Data out
-	1	0	0	1	1(first data)
+ve edge	0	1	0	0	0(second data)
+ve edge	0	0	1	0	0(third data)
+ve edge	0	0	0	1	1(fourth data)

Truth table 4.16

1) Before applying the first clock pulse, we can get the first data from its output terminal.
2) The remaining data are occurred at the output terminal after applying every sequential clock pulse.

.Serial-In Parallel-Out Shift Register

In this configuration the write operation is in serial form and read operation in parallel form. A four bit shift register serial in parallel out configuration is shown in figure. 4.22

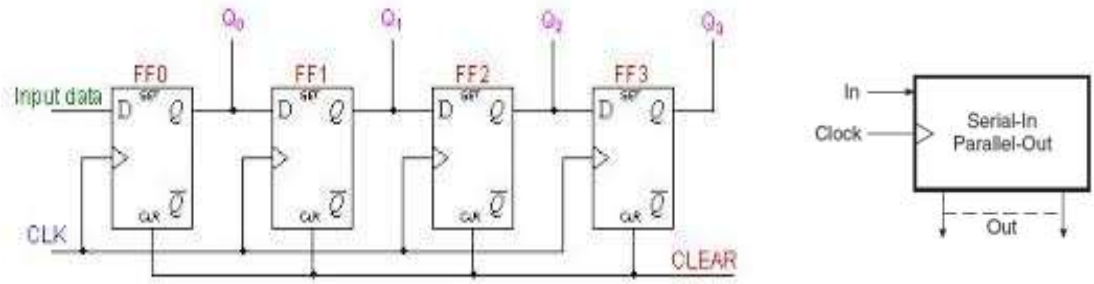


Fig 4.22

Figure 4.22 shows the logic diagram of a typical serial-in parallel-out shift register

Write operation

Suppose four bit data of 1001 is written into its registers the following procedure may be followed.

1. Reset all the flip-flop
2. Apply the data input to Data in terminal one by one.
3. Clock pulse should be applied to the clock terminal after applying every data input

Clock input	Data in	Q_0	Q_1	Q_2	Q_3
Reset	-	0	0	0	0
+ve edge	1	1	0	0	0
+ve edge	0	0	1	0	0
+ve edge	0	0	0	1	0
+ve edge	1	1	0	0	1

Truth table 4.17

After applying the 4 bit data input with clock signals, all data are stored in the flip-flop.

The flip-flops shown respond to the LOW-to-HIGH transition of the clock pulses as indicated by their logic symbols.

During the first clock transition, the QA output goes from logic ‘0’ to logic ‘1’.The outputs of the other three flip-flops remain in the logic ‘0’ state as their D inputs were in the logic ‘0’ state at the time of clock transition.

During the second clock transition, the QA output goes from logic ‘1’ to logic ‘0’ and the QB output goes from logic ‘0’ to logic ‘1’, again in accordance with the logic status of the D inputs at the time of relevant clock transition.

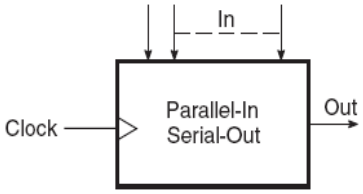
Thus, we have seen that a logic ‘1’ that was present at the data input prior to the occurrence of the first clock transition has reached the QB output at the end of two clock transitions. This bit will reach the QD output at the end of four clock transitions.

Read operation

After the write operation, we can be able to read the 4 bit data directly without using any clock signal.

Parallel-In Serial-Out Shift Register

In this configuration the write operation is in parallel form and read operation in serial form. A four bit shift register serial in parallel out configuration is shown in figure. 4.23



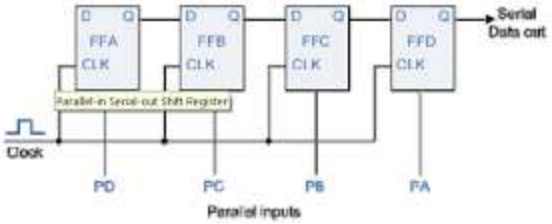


Figure: Parallel-in and Serial-out shift register

Fig 4.23

Write operation

Suppose four bit data of 1001 is written into its registers the following procedure may be followed.

1) Reset all the flip-flop
2) Apply the data input to Data in terminal one by one.

After these operations, the applied input data is stored in the registers.

Read operation

After write operation, we can be able to read the data by using Data out terminal. The clock pulse are needed for read operation. The procedure for read operation is shown below

lock input	Q_A	Q_B	Q_C	Q_D	Data out
-	1	0	0	1	1(first data)
+ve edge	0	1	0	0	0(second data)
+ve edge	0	0	1	0	0(third data)
+ve edge	0	0	0	1	1(fourth data)

Truth table 4.18

1)Before applying the first clock pulse, we can get the first data from its output terminal.
2) The remaining data are occurred at the output terminal after applying every sequential clock pulse.

Parallel-In Parallel-Out Shift Register

In this configuration both write and read operation are in parallel form. The clock pulses are not used in write and read operations. A four bit shift register parallel in parallel out configuration is shown in figure. 4.24

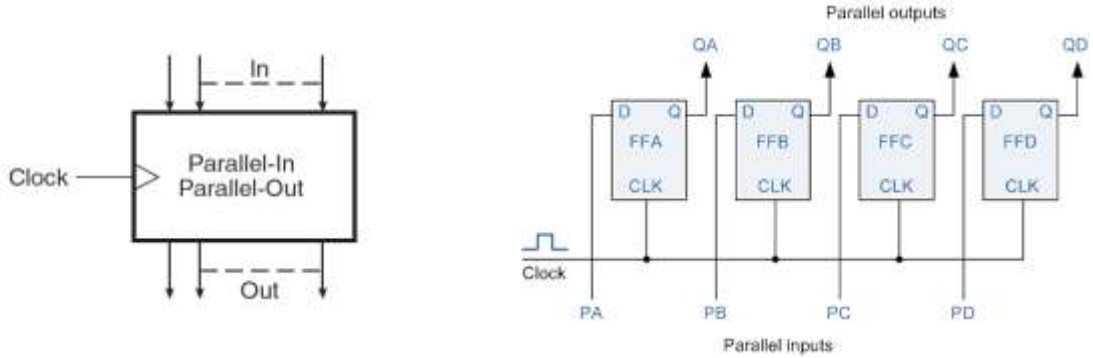


Fig 4.24

Write operation

Suppose four bit data of 1001 is written into its registers the following procedure may be followed.

1) Reset all the flip-flop
2) Apply the data input to Data in terminal one by one.

After these operations, the applied input data is stored in the registers

Read operation

After the write operation, we can be able to read the 4 bit data directly without using any clock signal.

REVIEW QUESTIONS

Two Mark

1	What is flip flop? What is its use?
2	What are the types of flip flop
3	How will get D flip flop from JK flip flop?
4	How will get T flip flop from JK flip flop?
5	Define Toggle.
6	What will happens ?when J and K =1?
7	What are the applications of counter?
8	What are the components used in counter?
9	What are the types of counter?
10	What is the use of pre settable counter?
11	What is shift register?
12	What is decade counter?
13	What are the applications of shift register?

Three Mark

1	Define Race around condition.
2	What is the concept of JKMS flip flop?
3	Give the truth table of RS flip flop.
4	Give the truth table of JK flip flop.
5	What is synchronous counter? Give example.
6	What is Asynchronous counter? Give example.
7	Give the concept of mod n counter.
8	What is shift register? What are its type?
9	Draw the wave forms of 4 bit binary counter.
10	Draw the wave forms of decade counter.

10 Mark

1	Explain RS and D flip flops with neat diagrams and truth tables.
2	Explain JK and T flip flops with neat diagrams and truth tables.
3	Explain JKMS flip flop with neat diagram and truth table.
4	Explain 4 bit binary counter with neat diagrams and waveforms.
5	Explain decade counter with neat diagrams and waveforms.
6	Explain up down counter with neat diagrams.
7	Explain Ring and Johnson counter with neat diagrams and waveforms.
8	Explain shift register with different modes of operation.

UNIT - V

D/A, A/D and Memory

5.1. D/A Convertor

A D/A converter is important not only because it is needed at the output of most digital systems, where it converts a digital signal into an analogue voltage or current so that it can be fed to a chart recorder, for instance, for measurement purposes, or a servo motor in a control application; it is also important because it forms an indispensable part of the majority of A/D converter types.

A D/A converter take digital data at its input and converts them into analogue voltage or current that is proportional to the weighted sum of digital inputs.

5.2. Basic Concepts

Any analog voltage can be expressed as a binary word by assigning voltage weights to each bit position. Considering three bit word voltage values of 4,2,1 could be assigned to each bit position as follows

Binary Word			Voltage
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Each successive binary count represents 1/7 th of the entire voltage. Here a voltage of 5.6 is represented as 110(decimal value of 6) because 5.6 is very closer to 6.

The output voltage of n-bit D/A converter is given by following equation

$$V_0=V_R(a_1 2^{-1}+a_2 2^{-2}\text{----}a_n 2^{-n})$$

Where V_R is usually a stable reference voltage and the coefficient a_1 to an are equal to 0 if a bit is OFF and equal to I if a bit is ON. In this equation, the MSB has a weight $V_R/2^n$ and weight of LSB is $V_R/2$. When all the inputs are in 1 state. V_0 will be $(V_R/2^{n-1})$ and this corresponds to the full scale output voltage of the D/A converter.

An A/D converter, too, has numerous applications. When it comes to transmitting analogue data, it forms an essential interface with a digital communication system where the analogue signal to be transmitted is digitized at the sending end with an A/D converter. It is invariably used in all digital read-out test and measuring equipment. Whether it is a digital multimeter or a digital storage oscilloscope or even a pH meter, an A/D converter is an important and essential component of all of them.

Digital-to-analogue (D/A) and analogue-to-digital (A/D) converters constitute an essential link when digital devices interface with analogue devices, and vice versa. They are important building blocks of any digital system, including both communication and non-communication systems, besides having other applications.

An ADC inputs an analog electrical signal such as voltage or current and outputs a binary number. In block diagram form, shown in fig 5.1 and 5.2

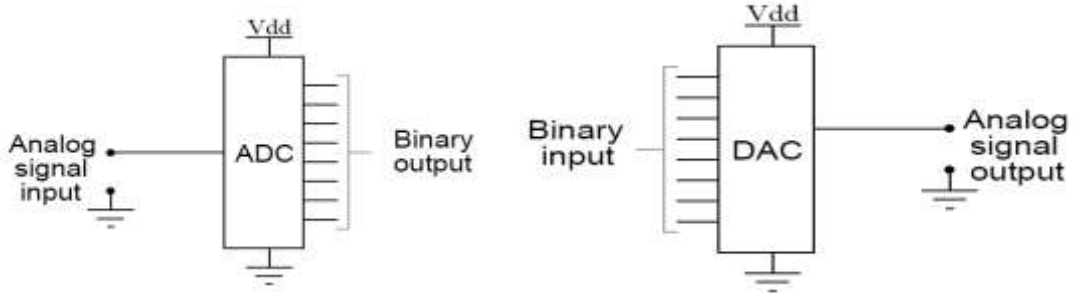


Fig 5.1

Together, they are often used in digital systems to provide complete interface with analog sensors and output devices for control systems such as those used in automotive engine controls:

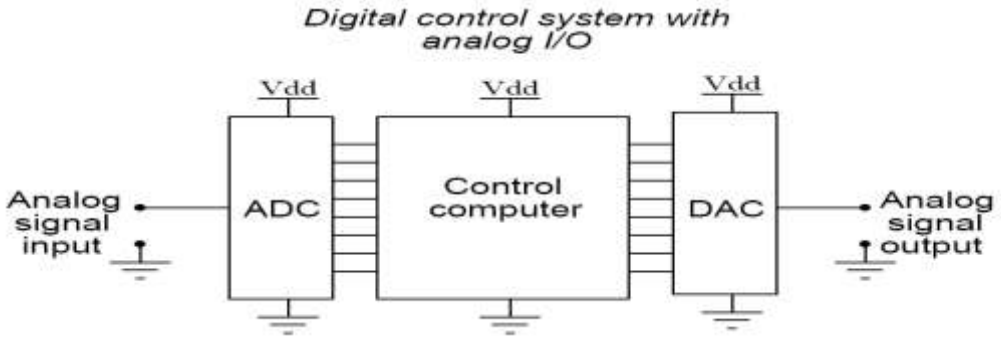
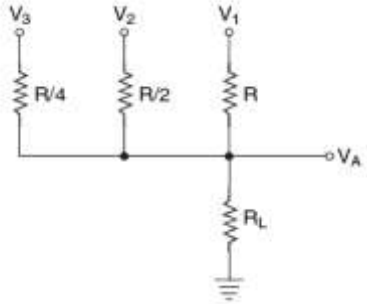


Fig 5.2

5.3. Weighted resistor D/A converter

Weighted resistor D/A converter can be used to convert a digital input into an equivalent analogue output. Figure shows one such resistive network that can convert a three-bit digital input into an analogue output. This network, however, can be extended further to enable it to perform digital-to-analogue conversion of digital data with a larger number of bits. In the network of Fig., if RL is much larger than R_ it can be proved with the help of simple network theorems that the output analogue voltage is given by



$$V_A = \frac{[V_1/R]+[V_2/(R/2)]+[V_3/(R/4)]}{[1/R]+[1/(R/2)]+[1/(R/4)]} = \frac{[V_1/R]+[2V_2/R]+[4V_3/R]}{[1/R]+[2/R]+[4/R]} = \frac{V_1+2V_2+4V_3}{7}$$

which can be further expressed as

$$V_A = \frac{V_1 \times 2^0 + V_2 \times 2^1 + V_3 \times 2^2}{2^3 - 1}$$

In general

$$V_A = \frac{V_1 \times 2^0 + V_2 \times 2^1 + V_3 \times 2^2 + \cdots + V_n \times 2^{n-1}}{2^n - 1}$$

$$V_A = \frac{V(2^0 + 2^1 + 2^2 + \cdots + 2^{n-1})}{2^n - 1} = V$$

5.4. R-2R Ladder D/A converter

The simple resistive divider network of has **two serious drawbacks**.

1. resistor in the network is of a different value. Since these networks use precision resistors, the added expense becomes unattractive.
2. the resistor used for the most significant bit (MSB) is required to handle a much larger current than the LSB resistor. For example, in a 10-bit network, the current through the MSB resistor will be about 500 times the current through the LSB resistor.

To overcome these drawbacks, a second type of resistive network called the ***binary ladder*** **(or R/2R ladder)** is used in practice.

The binary ladder, too, is a resistive network that produces an analogue output equal to the weighted sum of digital inputs. Figure shows the binary ladder network for a four-bit D/A converter. As is clear from the figure 5.4 , the ladder is made up of only two different values of resistor. This overcomes one of the drawbacks of the resistive divider network. It can be proved with the help of simple mathematics that the analogue output voltage VA in the case of binary ladder network of Fig. is given by

$$V_A = \frac{V_1 \times 2^0 + V_2 \times 2^1 + V_3 \times 2^2 + V_4 \times 2^3}{2^4}$$

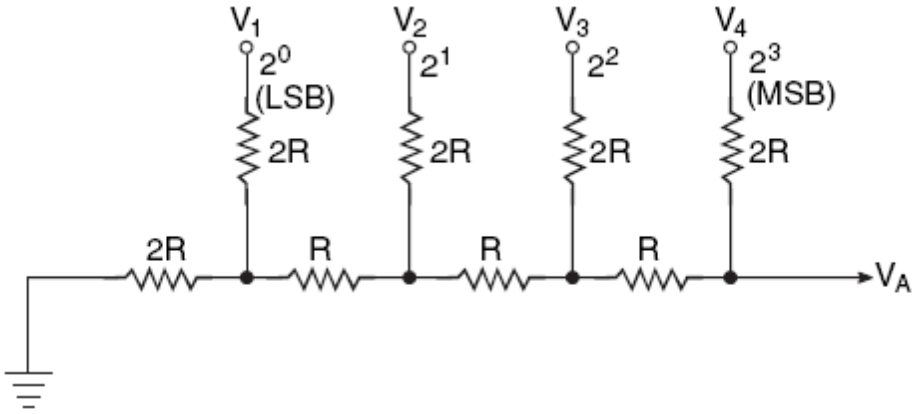


Fig 5.4

In general, for an n-bit D/A converter using a binary ladder network

$$V_A = \frac{V_1 \times 2^0 + V_2 \times 2^1 + V_3 \times 2^2 + \cdots + V_n \times 2^{n-1}}{2^n}$$

For $V_1 = V_2 = V_3 = \cdots = V_n = V$, $V_A = [(2^n - 1)/2^n]V$. For $V_1 = V_2 = V_3 = \cdots = V_n = 0$, $V_A = 0$.

The analogue output voltage in this case varies from 0 (for an all 0s input) to [(2n −1)/2n]V (for an all 1s input).

5.5. Specification of DAC IC

The major performance specifications of a D/A converter include resolution, accuracy, conversion speed, dynamic range, nonlinearity (NL) and differential nonlinearity (DNL)

Resolution

The ***resolution*** of a D/A converter is the number of states (2n_ into which the full-scale range is divided or resolved.

Accuracy

The ***accuracy*** of a D/A converter is the difference between the actual analogue output and the ideal expected output when a given digital input is applied.

Linearity

In a D/A converter, equal increment in the digital input should result in equal increments in the analog output voltage . In the actual circuit the input output relationship is not linear. This is due to the error in resistor values and voltage across the switches.

Settling Time

It is the time required for the analog output to settle within (1/2) LSB of the final value after the change in the digital input.

Conversion speed

The ***conversion speed*** of a D/A converter is expressed in terms of its settling time. The *settling time* is the time period that has elapsed for the analogue output to reach its final value within a specified error band after a digital input code change has been effected.

Temperature Sensitivity

The analog output voltage for any fixed digital input varies with temperature. This is due to the temperature sensitivitiews of the reference voltage sources, resistor etc

Dynamic Range

This is the ratio of the largest output to the smallest output, excluding zero, expressed in dB.

Nonlinearity **(NL)**

Nonlinearity **(NL)** is the maximum deviation of analogue output voltage from a straight line drawn between the end points, expressed as a percentage of the full-scale range or in terms of LSBs.

Differential nonlinearity **(DNL)**

Differential nonlinearity **(DNL)** is the worst-case deviation of any adjacent analogue outputs from the ideal one-LSB step size.

5.6. Analog to Digital conversion

A/D converter is a very important building block and has numerous applications. It forms an essential interface when it comes to analysing analogue data with a digital computer. It is an indispensable part of any digital communication system where the analogue signals to be transmitted is digitized at the sending end with the help of an A/D converter.

An A/D converter takes at its input an analogue voltage and after a certain amount of time produces a digital output code representing the analogue input. The A/D conversion process is generally more complex than the D/A conversion process.

5.7. Sampling and Quantization

Sampling Frequency and Aliasing Phenomenon If the rate at which the analogue signal to be digitized is sampled is at least twice the highest frequency in the analogue signal, which is what is embodied in the **Shannon–Nyquist sampling theorem**, then the analogue signal can be faithfully reproduced from its quantized values by using a suitable interpolation algorithm.

The accuracy of the reproduced signal is, however, limited by the quantization error. If the sampling rate is inadequate, i.e. if it is less than the Nyquist rate, then the reproduced signal is not a faithful reproduction of the original signal and these spurious signals, called **aliases,** are produced.

The frequency of an aliased signal is the difference between the signal frequency and the sampling frequency. For example, if sampled at a 1.5 kHz rate, a 2 kHz sine wave would be reconstructed as a 500 Hz sine wave. This problem is called ***aliasing*** and, in order to avoid it, the analogue input signal is low-pass filtered to remove all frequency components above half the sampling rate. This filter, called an *anti-aliasing filter*, is used in all practical A/D converters.

Quantization is the process of mapping (dividing) an analog signal into several equivalent discrete levels (ranges or steps). That means, it is the process of converting a continuous input values to a discrete output values.

An ADC can be modeled as two processes: **sampling** and **quantization**. Sampling converts a voltage signal (function of time) into a **discrete-time signal** (sequence of real numbers). Quantization replaces each real number with an approximation from a finite set of **discrete values** (**levels**), which is necessary for storage and processing by numerical methods. Most commonly, these discrete values are represented as fixed-point words (either proportional to the waveform values or **companded)** or floating-point words. Common word-lengths are **8-bit** (256 levels), **16-bit (**65,536 levels), **32-bit(**4.3 billion levels), and so on, though any number of quantization levels is possible (not just powers of two). Quantizing a sequence of numbers produces a sequence of quantization errors which is sometimes modeled as an additive random signal called **quantization noise** because of its stochastic behavior. The more levels a quantizer uses, the lower is its quantization noise power.

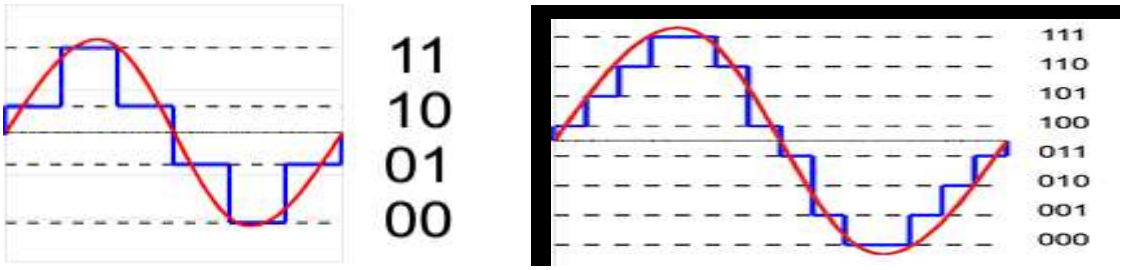


Fig 5.5

2-bit resolution with four **levels** **3-bit resolution with eight levels.**

In general, both ADC processes lose some information. So discrete-valued signals are only an approximation of the continuous-valued discrete-time signal, which is itself only an approximation of the original continuous-valued continuous-time signal. But both types of approximation errors can, in theory, be made arbitrarily small by good design.

The ***quantization error*** is inherent to the digitizing process. For a given analogue input voltage range it can be reduced by increasing the number of digitized levels. An A/D converter having an n-bit output can only identify 2n output codes while there are an infinite number of analogue input values adjacent to the LSB of the A/D converter that are assigned the same output code.

Conversion Time This is the time that elapses from the time instant of the start of the conversion signal until the conversion complete signal occurs.

5.8. Types of A/D Converter

Analogue-to-digital converters are often classified according to the conversion process or the conversion technique used to digitize the signal. Based on various conversion methodologies, common types of A/D converter include

flash or simultaneous or direct-conversion A/D converters,
half-flash A/D converters,
counter-type A/D converters,
tracking A/D converters,
successive approximation type A/D converters,
single-slope,
dual-slope and
multislope A/D converters and sigma-delta A/D converters.

5.9.Ramp type A/D converter (Counter-Type A/D Converter)

It is possible to construct higher-resolution A/D converters with a single comparator by using a variable reference voltage. One such A/D converter is the *counter-type A/D converter* represented by the block schematic of Fig. 5.6

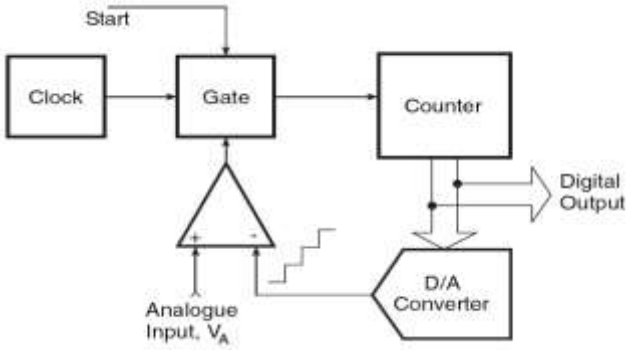


Fig 5.6

The circuit functions as follows.

To begin with, the counter is reset to all 0s. When a convert signal appears on the start line, the input gate is enabled and the clock pulses are applied to the clock input of the counter. The counter advances through its normal binary count sequence. The counter output feeds a D/A converter and the staircase waveform generated at the output of the D/A converter forms one of the inputs of the comparator.

The other input to the comparator is the analogue input signal. Whenever the D/A converter output exceeds the analogue input voltage, the comparator changes state. The gate is disabled and the counter stops. The counter output at that instant of time is then the required digital output corresponding to the analogue input signal. The counter-type A/D converter provides a very good method for digitizing to a high resolution. This method is much simpler than the simultaneous method for higher-resolution A/D converters.

The drawback with this converter is that the required conversion time is longer. Since the counter always begins from the all 0s position and counts through its normal binary sequence, it may require as many as 2n counts before conversion is complete.

The average conversion time can be taken to be $2n/2 = 2n-1$ counts. One clock cycle gives one count. As an illustration, if we have a four-bit converter and a 1 MHz clock, the average conversion time would be 8 ms. It would be as large as 0.5 ms for a 10-bit converter of this type at a 1 MHz clock rate. In fact, the conversion time doubles for each bit

5.10. Successive Approximation Type A/D Converter

The development of A/D converters has progressed in a quest to **reduce the conversion time**. The successive approximation type A/D converter aims at approximating the analogue signal to be digitized by trying only one bit at a time. The process of A/D conversion by this technique can be illustrated with the help of an example.

Let us take a four-bit successive approximation type A/D converter. Initially, the counter is reset to all 0s. The conversion process begins with the MSB being set by the start pulse. That is, the flip-flop representing the MSB is set. The counter output is converted into an equivalent analogue signal and then compared with the analogue signal to be digitized. A decision is then taken as to whether the MSB is to be left in (i.e. the flip-flop representing the MSB is to remain set) or whether it is to be taken out (i.e. the flip-flop is to be reset) when the first clock pulse sets the second MSB. Once the second MSB is set, again a comparison is made and a decision taken as to whether or not the second MSB is to remain set when the

subsequent clock pulse sets the third MSB. The process continues until we go down to the LSB. Note that, every time we make a comparison, we tend to narrow down the difference between the analogue signal to be digitized and the analogue signal representing the counter count.

Refer to the operational diagram of Fig It is clear from the diagram that, to reach any count from 0000 to 1111, the converter requires four clock cycles. In general, the number of clock cycles required for each conversion will be n for an n-bit A/D converter of this type. Figure shows a block schematic representation of a successive approximation type A/D converter. Since only one flip-flop (in the counter) is operated upon at one time, a ring counter, which is nothing but a circulating register (a serial shift register with the outputs Q and Q of the last flip-flop connected to the J and K inputs respectively of the first flip-flop), is used to do the job. Referring to Fig. 5.7, the dark lines show the sequence in which the counter arrives at the desired count, assuming that 1001 is the desired count.

This type of A/D converter is much faster than the counter-type A/D converter previously discussed. In an n-bit converter, the counter-type A/D converter on average would require 2n−1 clock cycles for each conversion, whereas a successive approximation type converter requires only n clock cycles. That is, an eight-bit A/D converter of this type operating on a 1 MHz clock has a conversion time of 8 s.

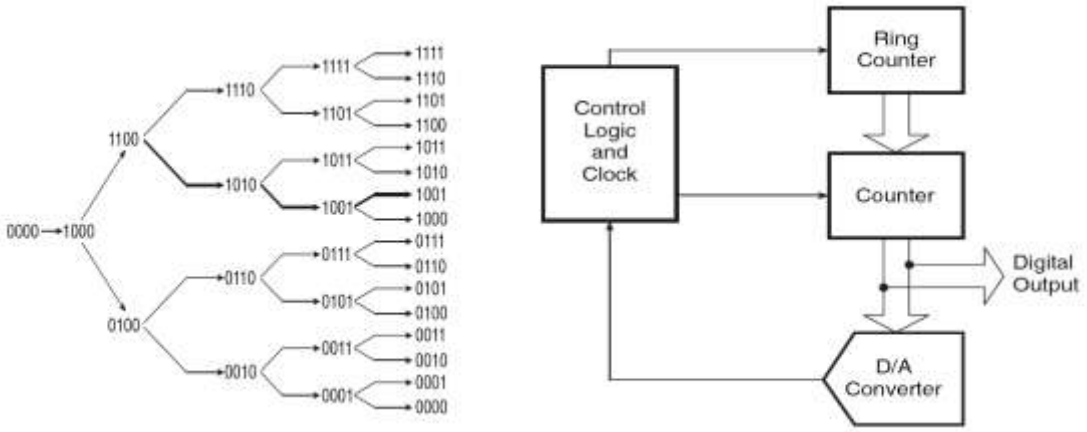


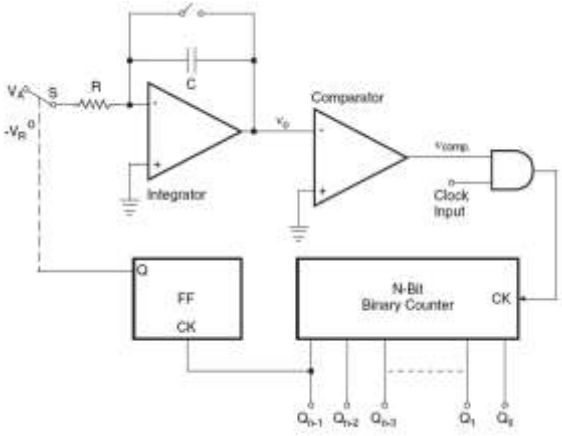
Fig 5.7

5.11. Dual slope A/D Converters

Figure shows a block schematic arrangement of a *dual-slope integrating A/D converter*. The converter works as follows. In fig 5.8 Initially, switch S is connected to the analogue input voltage VA to be digitized. The output of the integrator is mathematically given by

$$v_o = (-1/RC)\int V_A.dt = (-V_A/RC).t$$

The moment vo tends to go below zero, clock pulses reach the clock input terminal of the counter which is initially cleared to all 0s. The counter begins counting from 0000 _ _ _ 0.



At the (2n_th clock pulse, the counter is again cleared, the ‘1’ to ‘0’ transition of the MSB of the counter sets a flip-flop that controls the state of switch S which now connects the integrator input to a reference voltage of polarity opposite to that of the analogue input. The integrator output

moves in the positive direction; the counter has again started counting after being reset (at, say, t=T1_. The moment the integrator output tends to exceed zero, the counter stops as the clock pulses no longer reach the clock input of the counter. The counter output at this stage (say, at t=T2_ is proportional to the analogue input. Mathematically, it can be proved that n =(VA/VR_.2n, where n is the count recorded in the counter at t=T2. Figure illustrates the concept further with the help of relevant waveforms.

Fig 5.8

This type of A/D converter is very popular in digital voltmeters owing to its good conversion accuracy and low cost. Also, the accuracy is independent of both the integrator capacitance and the clock frequency, as they affect the negative and positive slope in the same manner. Yet another advantage of the dual-slope integrator A/D converter is that the fixed analogue input integration period results in rejection of noise frequencies present in the analogue input and having time periods that are equal to or submultiples of the integration time. The proper choice of integration time can therefore achieve excellent rejection of 50/60 Hz line ripple.

5.12 Simultaneous or Flash A/D Converters

The simultaneous method of A/D conversion is based on using a number of comparators. The number f comparators needed for n-bit A/D conversion is 2n−1. One such system capable of Converting the analogue input signal into a two-bit digital output is shown in Fig..5.9 The analogue signal to be digitized serves as one of the inputs to each of the comparators. The second input for each of the comparators is a reference input, different for each comparator. In the present case of a two-bit A/D converter, the reference voltages for the three comparators will be V/4, V/2 and 3V/4. the output status of various comparators depends upon the input analogue signal VA.

For instance, when the input VA lies between V/4 and V/2, the C1 output is HIGH whereas the C2 and C3 outputs are both LOW. The results are summarized in Table .5.1

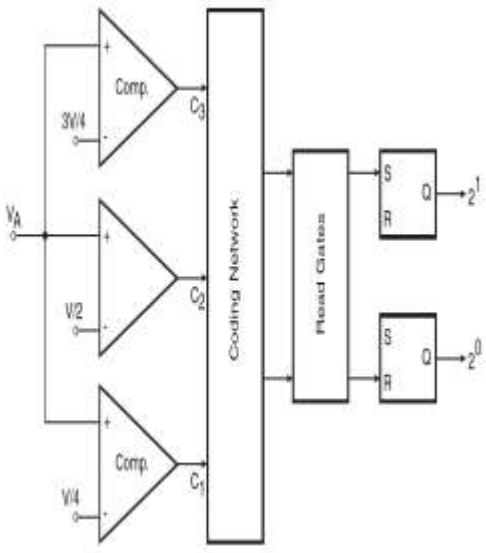


Fig 5.9

The three comparator outputs can then be fed to a coding network (comprising logic gates, etc.) to provide two bits that are the digital equivalent of the input analogue voltage. The bits at the output of the coding network can then be entered into a flip-flop register for storage. Figure shows the arrangement of a three-bit simultaneous-type A/D converter.

Input analogue voltage(v_a)	C_1	C_2	C_3	2^1	2^2
0 to $V/4$	LOW	LOW	LOW	0	0
$V/4$ to $V/2$	HIGH	LOW	LOW	0	1
$V/2$ to $3V/4$	HIGH	HIGH	LOW	1	0
$3V/4$ to V	HIGH	HIGH	HIGH	1	1

Table 5.1

5.13.Voltage to Frequency Converter

The voltage to frequency converter changes a DC input voltage into a string of pulses, whose repetition rate (frequency) is proportional to the magnitude of the input voltage. The pulses are counted for a fixed period of time and the final count is proportional to the measured voltage. The block diagram of voltage to frequency is shown in figure 5.10

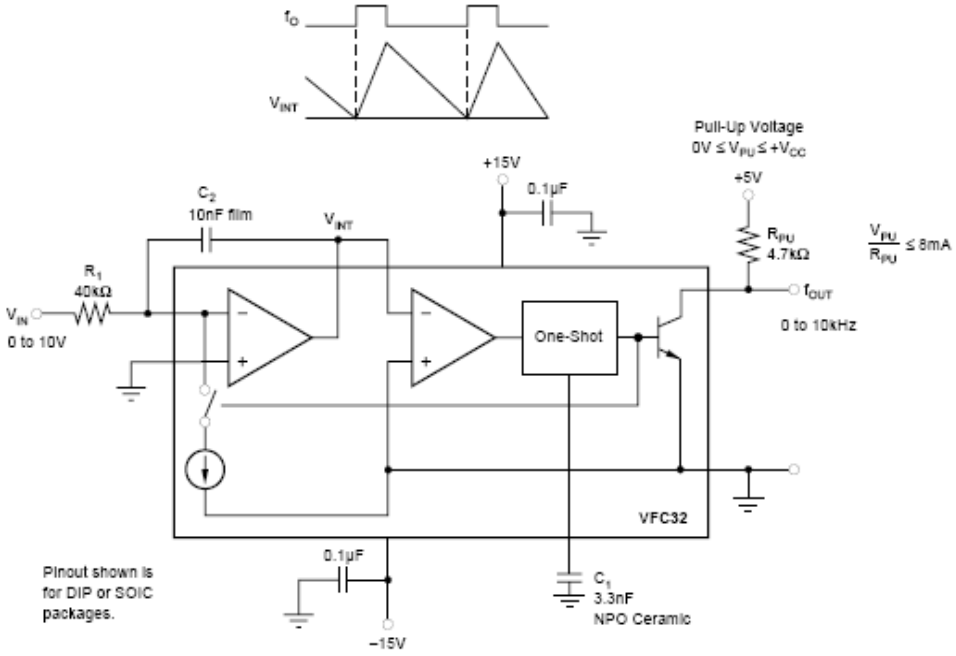


Fig 5.10

The input voltage generate a ramp signal, the slope of which is proportional to the amplitude of the input voltage. The output of the ramp generator is given to the voltage comparator in which it is compare with a reference voltage. Each time the amplitude of the ramp exceeds the absolute value of reference the comparator output goes high and trigger the monostable multivibrator (one shot), which in turn discharge the capacitor in the ramp generator and produce output pulse. The cycle repeats at a frequency that is a linear function of the amplitude of the input voltage.

Voltage to frequency converter can be considered to be a dual slope method

Where V in= V ref* t_2/t_1

But in this case V ref and t_2 are constants

Let K = Vref.t_2

V in = K*1/t1= K*F_0 ($F_0=1/t_1$)

Where F0 is the output frequency which is proportional to the input voltage Vin.

5.14. Frequency to voltage converter

The frequency to voltage converter performs the inverse operation of voltage to frequency converter.

Refer fig 5.11

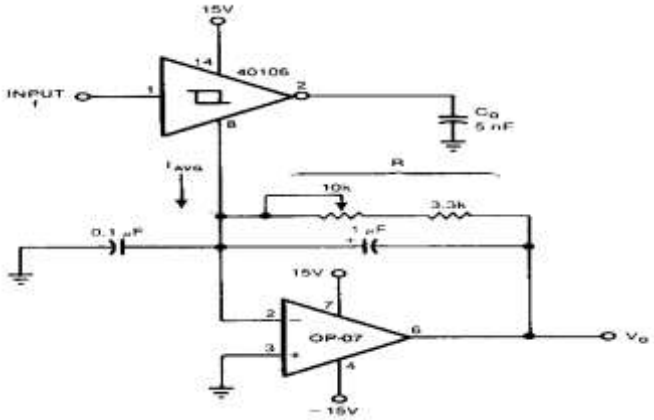


Fig 5.11

It produces an analog output voltage whose amplitude is a function of the frequency of the input signal. The input frequency is applied to the non inverting input of the comparator. The square or rectangular wave signal is applied to its input. Each time the input signal crossed zero in the negative direction, the output of the comparator goes low. This action charges the capacitor each time with precise amount of input voltage until the voltage across tit can no longer increases. On the other hand each time the input signal crosses zero in positive direction, the output of the compactor goes to high level. However the voltage across the capacitor is retained. The voltage across the capacitor is the output voltage Vo.

5.15. Specification of A/D Converter

The major performance specifications of an A/D converter include resolution, accuracy, gain and offset errors, gain and offset drifts, the sampling frequency and aliasing phenomenon, quantization error, nonlinearity, differential nonlinearity, conversion time, aperture and acquisition times and code width.

Input Impedance

The ratio of the change of input voltage to the change of input current is called input impedance

Conversion time

The time interval between the end of the start signal anfd end of conversion is called conversion time.

Differential linearity

It is a measure of variations in voltage step size that causes the converter to change from one state to other.

Drift

Drift of an A/D converter is the quality of a circuit to change the partameter with time

Range of input voltage

The voltage difference between the maximum applied input voltage to the minimum applied input voltage is known as range of input voltage.

Resolution

The resolution of an A/D converter is the quantum of the input analogue voltage change required to increment its digital output from one code to the next higher code.

Accuracy

The accuracy specification describes the maximum sum of all errors, both from analogue sources (mainly the comparator and the ladder resistors) and from the digital sources (quantization error) of the A/D converter.

Gain error

The gain error is the difference between the actual full-scale transition voltage and the ideal full-scale transition voltage. It is expressed either as a percentage of the full-scale range (% of FSR) or in LSBs.

Offset error

The offset error is the error at analogue zero for an A/D converter operating in bipolar mode. It is measured in% of FSR or in LSBs.

5.16 MEMORY

Memory is major part of computers that categories into several types. Memory is best storage part to the computer users to save information, programs and etc, The computer memory offer several kinds of storage media some of them can store data temporarily and some them can store permanently

Types

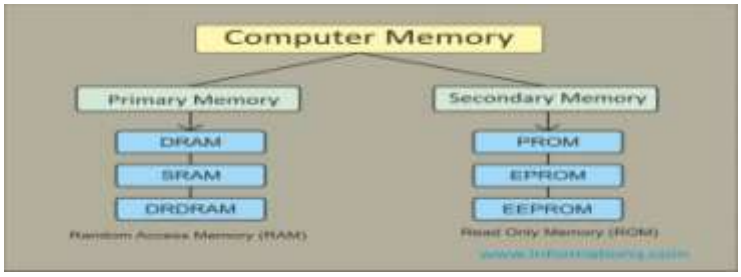


Fig 5.12

Memory is internal storage media of computer that has several names such as majorly categorized into two types, Main memory and Secondary memory.

1. Primary Memory / Volatile Memory.

2. Secondary Memory / Non Volatile Memory.

1. Primary Memory / Volatile Memory:

Primary Memory also called as volatile memory because the memory can't store the data permanently.

Random Access Memory (RAM):

The primary storage is referred to as random access memory (RAM) due to the random selection of memory locations. It performs both read and write operations on memory. If power failures happened in systems during memory access then you will lose your data permanently. So, RAM is volatile memory. RAM categorized into following types.

- DRAM - Dynamic Random Access Memory
- SRAM - Static Random Access Memory
- DRDRAM - Double data Rate Dynamic Random Access Memory

2. Secondary Memory / Non Volatile Memory:

Secondary memory is external and permanent memory that is useful to store the external storage media such as floppy disk, magnetic disks, magnetic tapes and etc cache devices. Secondary memory deals with following types of components.

Read Only Memory (ROM) :

ROM is permanent memory location that offer huge types of standards to save data. But it work with read only operation. No data lose happen whenever power failure occur during the ROM memory work in computers.

ROM memory has several models such names are following.

1. PROM: Programmable Read Only Memory (PROM) maintains large storage media but can't offer the erase features in ROM. This type of RO maintains PROM chips to write data once and read many. The programs or instructions designed in PROM can't be erased by other programs.

2. EPROM : Erasable Programmable Read Only Memory designed for recover the problems of PROM and ROM. Users can delete the data of EPROM thorough pass on ultraviolet light and it erases chip is reprogrammed.

3. EEPROM: Electrically Erasable Programmable Read Only Memory similar to the EPROM but it uses electrical beam for erase the data of ROM.

Cache Memory: Mina memory less than the access time of CPU so, the performance will decrease through less access time. Speed mismatch will decrease through maintain cache memory. Main memory can store huge amount of data but the cache memory normally kept small and low expensive cost. All types of external media like Magnetic disks, Magnetic drives and etc store in cache memory to provide quick access tools to the users

Why use an SRAM?

There are many reasons to use an SRAM or a DRAM in a system design. Design tradeoffs include density, speed, volatility, cost, and features. All of these factors should be considered before you select a RAM for your system design.

5.17 Basic Architecture of static memory

The basic architecture of a static RAM includes one or more rectangular arrays of memory cells with support circuitry to decode addresses, and implement the required read and write operations. Figure 5.13 shows a basic block diagram of a synchronous SRAM.

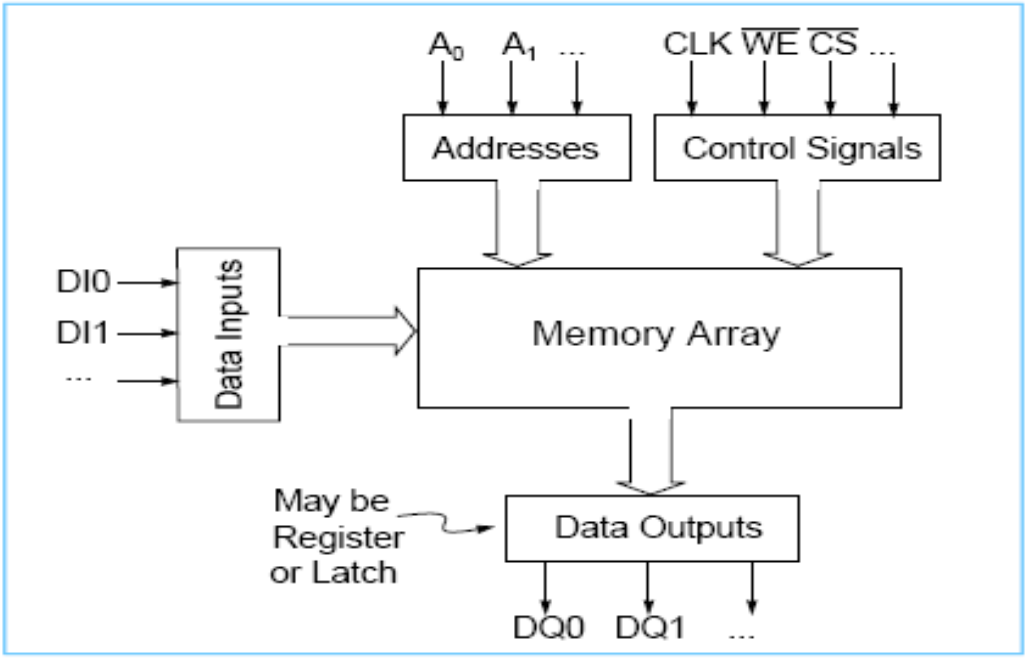


Fig 5.13

Memory Arrays

SRAM memory arrays are arranged in rows and columns of memory cells called word lines and bit lines, respectively. In IBM SRAMs, the word lines are made from poly silicon while the bit lines are metal. Each memory cell has a unique location or address defined by the intersection of a row and column. Each address is linked to a particular data input/output pin. The number of arrays on a memory chip is determined by the total size of the memory, the speed at which the memory must operate, layout and testing requirements, and the number of data I/Os on the chip.

Static Memory Cell

An SRAM memory cell is a bi-stable flip-flop made up of four to six transistors. The flip-flop may be in either of two states that can be interpreted by the support circuitry to be a 1 or a 0.

static RAM cell and its associated circuitry in block form. Each memory cell can latch, or store, data in a stable state. Information is written into and readout of the cell through the column lines. The characteristics of flip-flops keep the flip-flop in its present state and allow you to read the data out of the cell without changing its state when the row-line is activated. Similarly data is written through the column line only when the row-line is activated, so only one cell in each column is selected. A read/write control signal controls reading and writing operations. The zero or one state in the cells can be held indefinitely as long as proper power supply levels are maintained. D-type and R-S type flip-flops are commonly used for SRAMs.

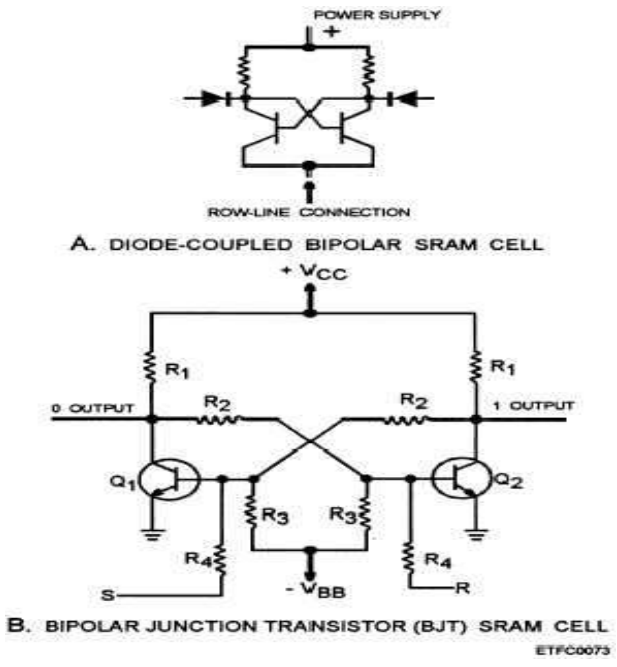


Fig 5.14

The flip-flops can be made of either bipolar or MOS transistors. MOS yields a higher density but lower access speed. Bipolar RAMs have a higher access speed but take up more space and figure 5.14 illustrate schematic diagrams of individual bipolar and MOS RAM cells. Figure 5.15, frame A, is a diode- duplex.

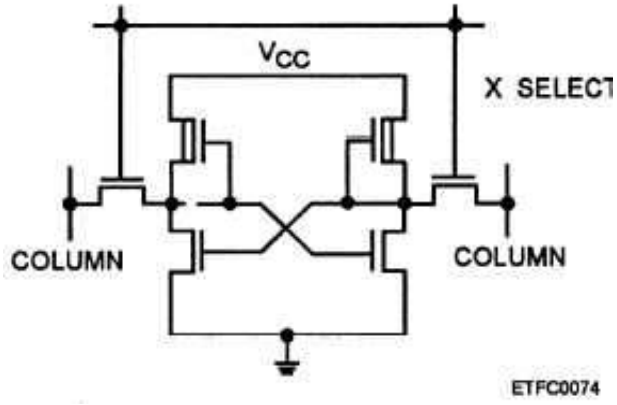


Fig 5.15

Bipolar junction transistor (BJT) SRAM cell.

bipolar static RAM cell; figure 6-28, frame B, is bipolar junction transistor (BJT) static RAM cell; and figure 5.14 is a static RAM MOS cell. As stated, the RAM chip is mounted in a logic array on a pcb. Figure 5.15 is an illustration of an IC chip, with pin connections used in a static bipolar or MOS RAM. RAM chips come in various configurations and sizes. The number of IC RAM chips needed for a computer's RAM memory is determined by the requirements and memory size of the computer. Let's use an example IC to discuss the operation of a RAM chip, which includes the architecture, address selection, and read/write cycles

5.19 What is the difference between SDRAM, DDR1, DDR2, DDR3 and DDR4?

SDRAM (Synchronous Dynamic Random Access Memory):

This enables the memory controller to know the exact clock cycle when the requested data will be ready, so the CPU no longer has to wait between memory accesses. SDRAM can stand for SDR SDRAM Single Data Rate SDRAM), where the I/O, internal clock and bus clock are the same. For example, the I/O, internal clock and bus clock of PC133 are all 133 Mhz. Single Data Rate means that SDR SDRAM can only read/write one time in a clock cycle. SDRAM have to wait for the completion of the previous command to be able to do another read/write operation.

DDR SDRAM (Double Data Rate SDRAM):

The next generation of SDRAM is DDR, which achieves greater bandwidth than the preceding single data rate SDRAM by transferring data on the rising and falling edges of the clock signal (double pumped). Effectively, it doubles the transfer rate without increasing the frequency of the clock. The transfer rate of DDR SDRAM is the double of SDR SDRAM without changing the internal clock. DDR SDRAM, as the first generation of DDR memory,

the prefetch buffer is 2bit, which is the double of SDR SDRAM. The transfer rate of DDR is between 266~400 MT/s. DDR266 and DDR400 are of this type.

DDR2 SDRAM(Double Data Rate Two SDRAM):

Its primary benefit is the ability to operate the external data bus twice as fast as DDR SDRAM. This is achieved by improved bus signal. The prefetch buffer of DDR2 is 4 bit(double of DDR SDRAM). DDR2 memory is at the same internal clock speed (133~200MHz) as DDR, but the transfer rate of DDR2 can reach 533~800 MT/s with the improved I/O bus signal. DDR2 533 and DDR2 800 memory types are on the market.

DDR3 SDRAM(Double Data Rate Three SDRAM):

DDR3 memory reduces 40% power consumption compared to current DDR2 modules, allowing for lower operating currents and voltages (1.5 V, compared to DDR2's 1.8 V or DDR's 2.5 V). The transfer rate of DDR3 is 800~1600 MT/s. DDR3's prefetch buffer width is 8 bit, whereas DDR2's is 4 bit, and DDR's is 2 bit. DDR3 also adds two functions, such as ASR (Automatic Self-Refresh) and SRT (Self-Refresh Temperature). They can make the memory control the refresh rate according to the temperature variation.

DDR4 SDRAM (Double Data Rate Fourth SDRAM):

DDR4 SDRAM provides the lower operating voltage (1.2V) and higher transfer rate. The transfer rate of DDR4 is 2133~3200 MT/s. DDR4 adds four new Bank Groups technology. Each bank group has the feature of singlehanded operation. DDR4 can process 4 data within a clock cycle, so DDR4's efficiency is better than DDR3 obviously. DDR4 also adds some functions, such as DBI (Data Bus Inversion), CRC (Cyclic Redundancy Check) and CA parity. They can enhance DDR4 memory's signal integrity, and improve the stability of data transmission/access.

%%%%%%%%%%%%%%%

REVIEW QUESTIONS

Two Mark

1	What is D/A converter?
2	What is A/D converter?
3	What are the types of D/A converter?
4	What are the types of A/D converter?
5	What is memory?
6	What is static memory?
7	What is dynamic memory?
8	What is volatile memory?
9	What is Non volatile memory?
10	What is SDRAM?
11	What is DDR RAM?

Three Mark

1	Give the basic concept of converter.
2	Give the specifications of D/A converter.
3	Give the specifications of A/D converter.
4	What are the types of memory?
5	Differentiate between SDRAM and DDR RAM.
6	Give the concept of sampling and quantization.

10 Mark

1	Explain R-2R Ladder DAC with neat diagram.
2	Explain Ramp method of ADC with neat diagram.
3	Explain successive approximation method of ADC with neat diagram.
4	Explain dual slope method of ADC with neat diagram.
5	Explain organization of static memory with neat diagram.