

Digital Cellular Systems

by

Dr. P. V. Ramana,

Professor & Head, Dept. of ECE,
Sree Vidyanikethan Engineering College, Tirupati

Unit: V

Course: Cellular & Mobile Communications

Target Group: IV B.Tech. ECE

Outline

- GSM-Introduction
- Architecture
- Frame Structure
- Channels

What is GSM ?

- Global System for Mobile Communication (GSM) is a **second generation cellular standard** developed to cater **voice** services and **data** delivery using digital modulation

GSM: History

- Developed by **Group Special Mobile** (founded 1982) which was an initiative of CEPT (**Conference of European Post and Telecommunication**)
- Aim : to replace the incompatible analog system
- Presently the responsibility of GSM standardization resides with special mobile group under ETSI (**European telecommunication Standards Institute**)
- Full set of specifications phase-I became available in 1990
- Under ETSI, GSM is named as **Global System for Mobile communication**
- Today many providers all over the world use GSM (more than 135 countries in Asia, Africa, Europe, Australia, America)
- More than 1300 million subscribers in world and 45 million subscriber in India.

GSM Services

- Tele-services
- Bearer or Data Services
- Supplementary services

Tele Services

- Telecommunication services that enable voice communication
 - via mobile phones
- Offered services
 - Mobile telephony
 - Emergency calling

Bearer Services

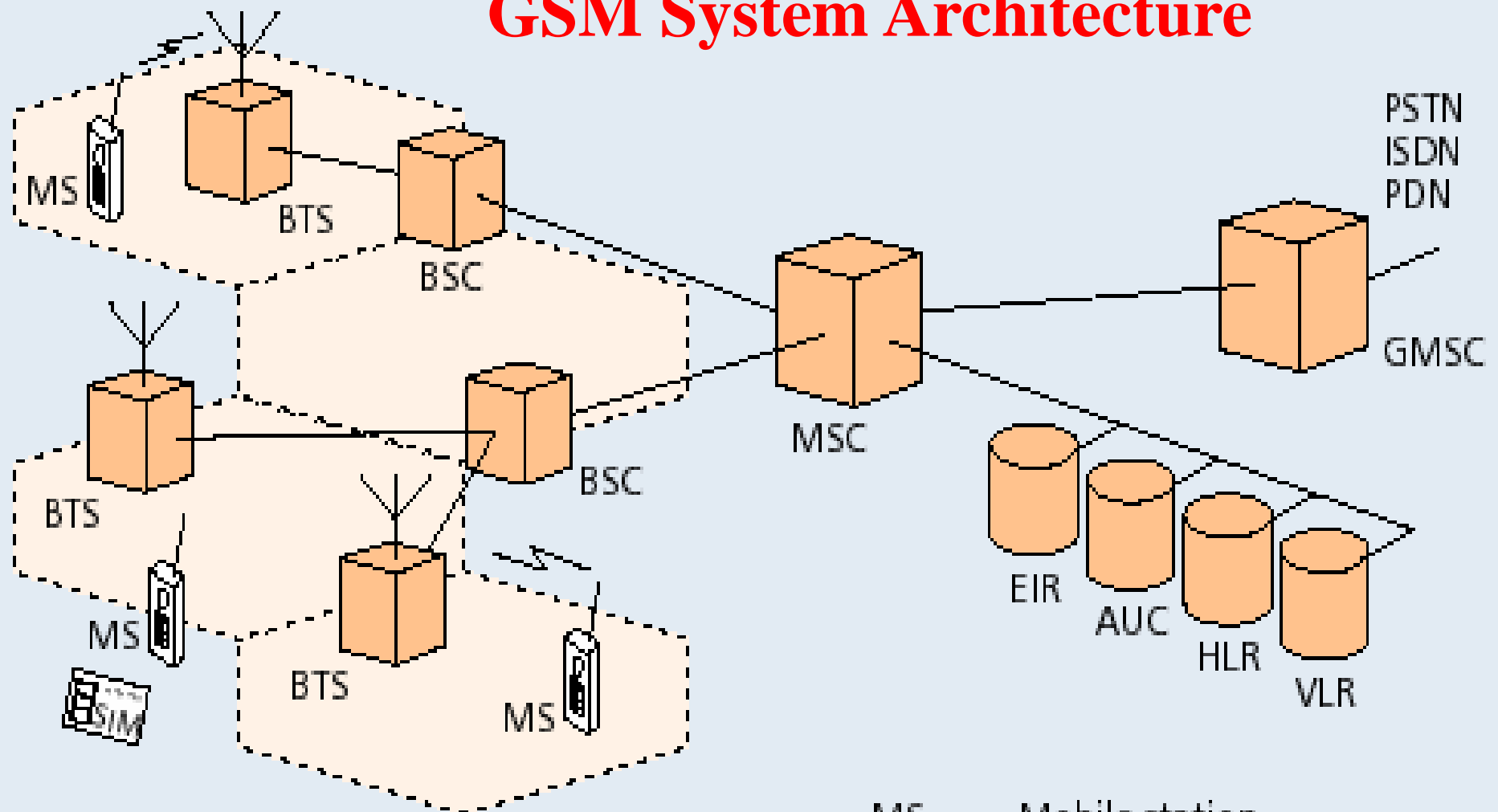
- Include various data services for information transfer between GSM and other networks like PSTN, ISDN etc at rates from 300 to 9600 bps
- Short Message Service (SMS)
 - up to 160 character alphanumeric data transmission to/from the mobile terminal
- Unified Messaging Services(UMS)
- Group 3 fax
- Voice mailbox
- Electronic mail

Supplementary Services

Call related services :

- Call Waiting- Notification of an incoming call while on the handset
- Call Hold- Put a caller on hold to take another call
- Call Barring- All calls, outgoing calls, or incoming calls
- Call Forwarding- Calls can be sent to various numbers defined by the user
- Multi Party Call Conferencing - Link multiple calls together
- CLIP – Caller line identification presentation
- CLIR – Caller line identification restriction
- CUG – Closed user group

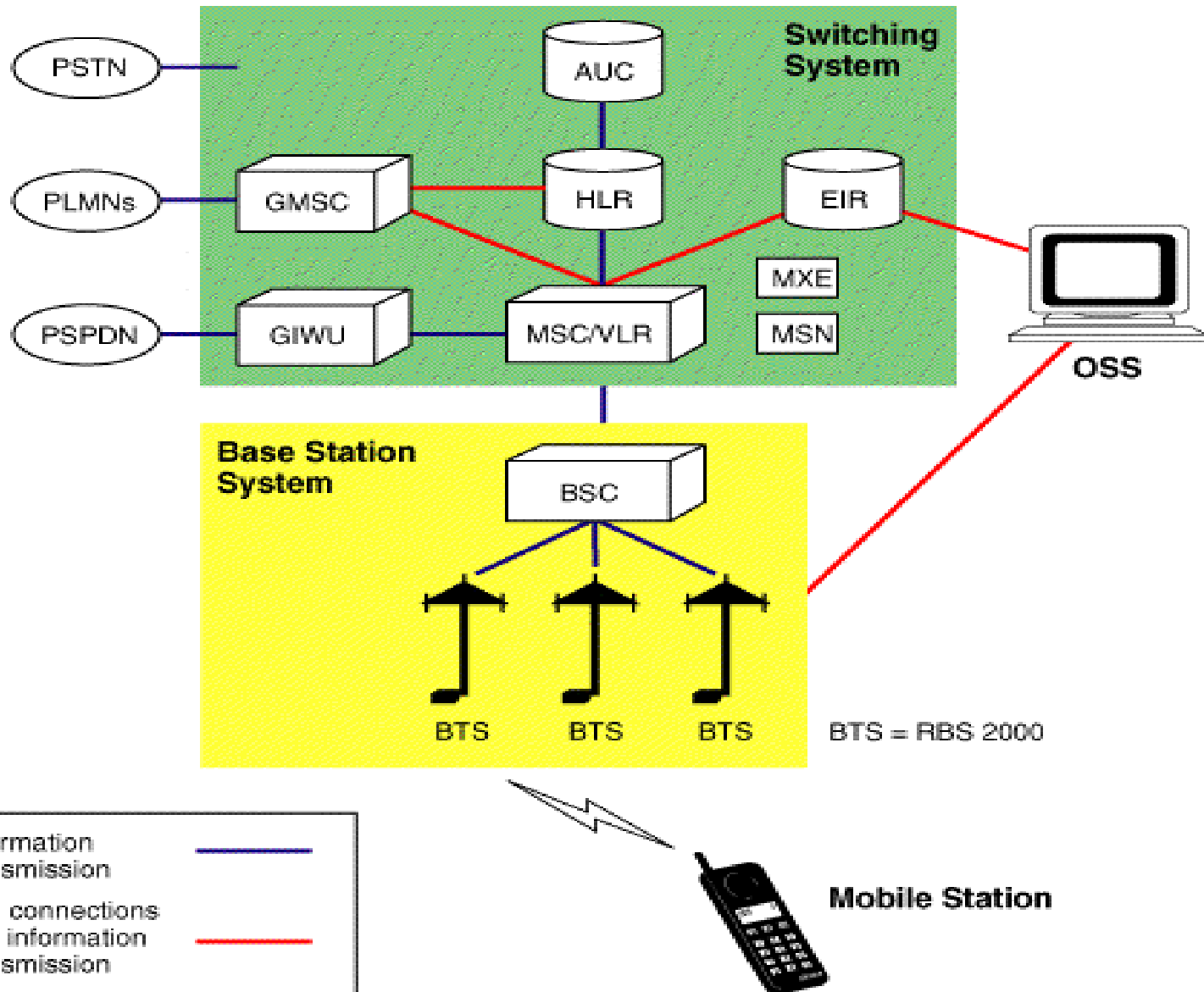
GSM System Architecture



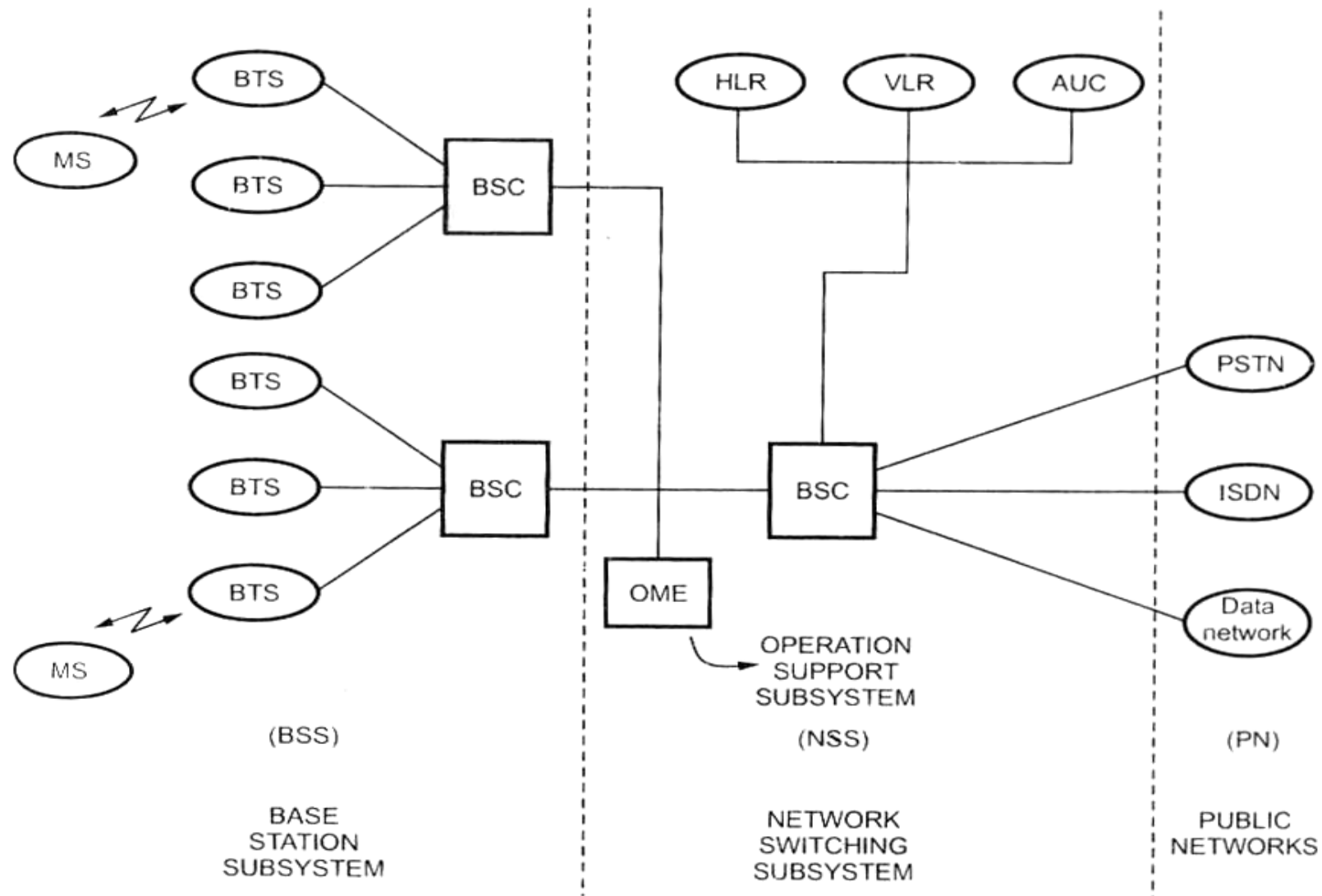
BTS Base transceiver station
 BSC Base station controller
 BSS Base station subsystem (BTS+BSC)
 MSC Mobile switching center
 GMSC Gateway MSC

MS Mobile station
 HLR Home location register
 VLR Visited location register
 EIR Equipment identity register
 AUC Authentication center

GSM Architecture



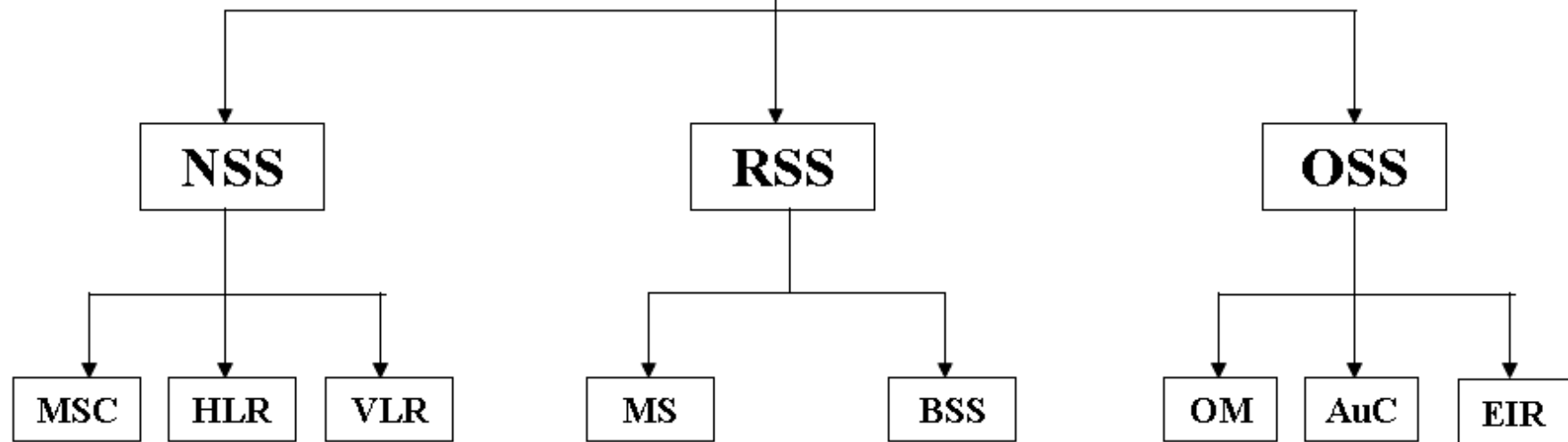
GSM System Architecture



In GSM architecture, the subsystems are

- Network and Switching Subsystems (**NSS**)
- Radio Subsystem (**RSS**)
- Operational Subsystems (**OSS**)

GSM Architecture



Network and Switching Subsystem (NSS)

- **It manages the connectivity between wireless N/Ws and public N/Ws. It consists of**
 - Mobile Switching Center (MSC)-central administration and cellular co-ordination
 - Home Location Register (HLR)- maintaining home user profile
 - Visitor Location Register (VLR)- maintaining visitors/new user profile

Mobile Switching Center (MSC)

- Heart of the network
- Manages communication between GSM and other networks
- Call setup function and basic switching
- Call routing
- Billing information and collection
- Mobility management
 - Registration
 - Location Updating
 - Inter BSS and inter MSC call handoff
- MSC does gateway function while its customer roams to other network by using HLR/VLR.

▪ **Home Location Registers (HLR)**

- Permanent database about mobile subscribers in a large service area (generally one per GSM network operator)
- Database contains IMSI, MSISDN, prepaid/postpaid, roaming restrictions, supplementary services.

▪ **Visitor Location Registers (VLR)**

- Temporary database which updates whenever new MS enters its area, by HLR database
- Controls those mobiles roaming in its area
- Reduces number of queries to HLR
- Database contains IMSI, TMSI, MSISDN, MSRN, Location Area, authentication key

Radio Subsystem

- It consists of two main parts
 - Mobile Station
 - Base Station System

The Mobile Station is made up of two entities:

1. Mobile Equipment (ME)
2. Subscriber Identity Module (SIM)

1. Mobile Equipment

- Portable, vehicle mounted, hand held device
- Uniquely identified by an IMEI (International Mobile Equipment Identity)
- Voice and data transmission
- Monitoring power and signal quality of surrounding cells for optimum handover
- Power level : 0.8W – 20 W
- 160 character long SMS.

2. Subscriber Identity Module (SIM)

- Smart card contains the International Mobile Subscriber Identity (IMSI)
- Allows user to send and receive calls and receive other subscribed services
- Encoded network identification details
- Protected by a password or PIN
- Can be moved from phone to phone – contains key information to activate the phone

Base Station Subsystem (BSS)

Base Station Subsystem is composed of two parts

1. Base Transceiver Station (**BTS**)
2. Base Station Controller (**BSC**)

1. Base Transceiver Station (BTS):

- Encodes, encrypts, multiplexes, modulates and feeds the RF signals to the antenna.
- Frequency hopping
- Communicates with Mobile station and BSC
- Consists of Transceivers (TRX) units

2. Base Station Controller (BSC)

- Manages Radio resources for BTS
- Assigns Frequency and time slots for all MS's in its area
- Handles call set up
- Transcoding and rate adaptation functionality
- Handover for each MS
- Radio Power control
- It communicates with MSC and BTS

Operation and maintenance Subsystem (OSS)

Base Station Subsystem is composed of two parts

1. Operation and Maintenance (OM)
2. Authentication Centre (Au C)
3. Equipment Identity Register (EIR)

2. Authentication Center (Au C)

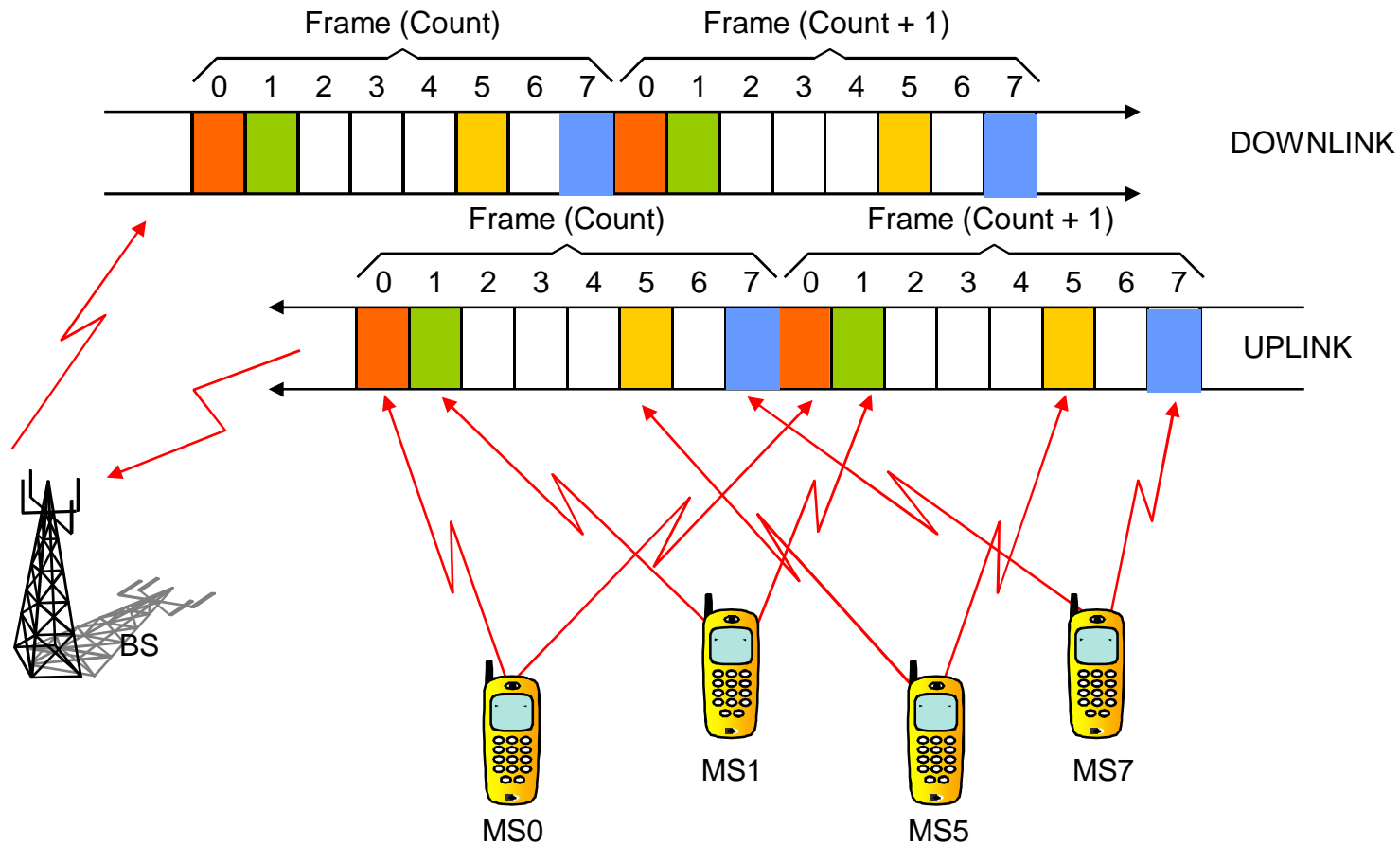
- Protects against intruders in air interface
- Maintains authentication keys and algorithms and provides security triplets
- Generally associated with HLR

3. Equipment Identity Register (EIR)

- Database that is used to track handsets using the IMEI
- Made up of three sub-classes: The White List, The Black List and the Gray List
- Only one EIR per PLMN

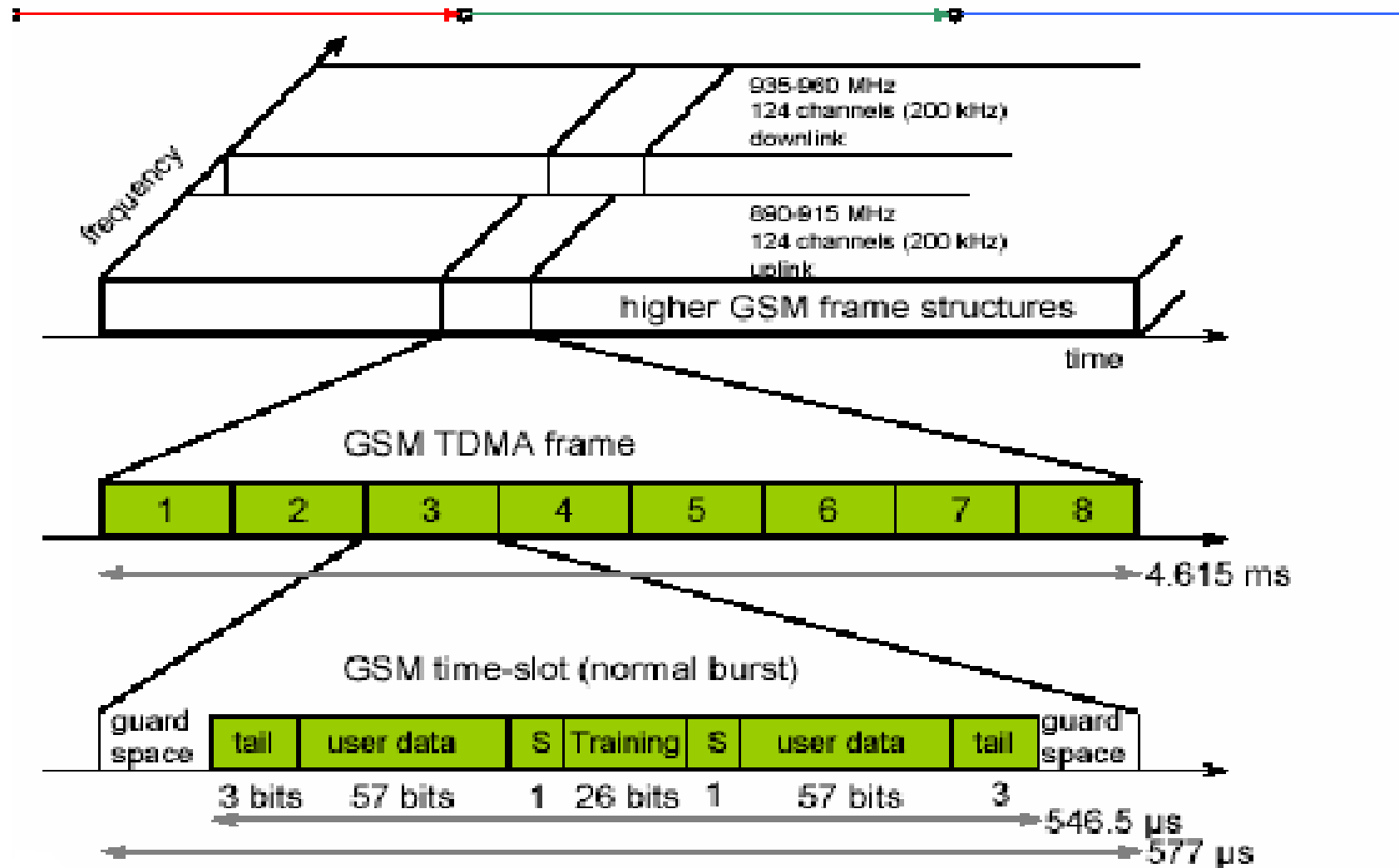
TDMA Operation in GSM

Full Rate

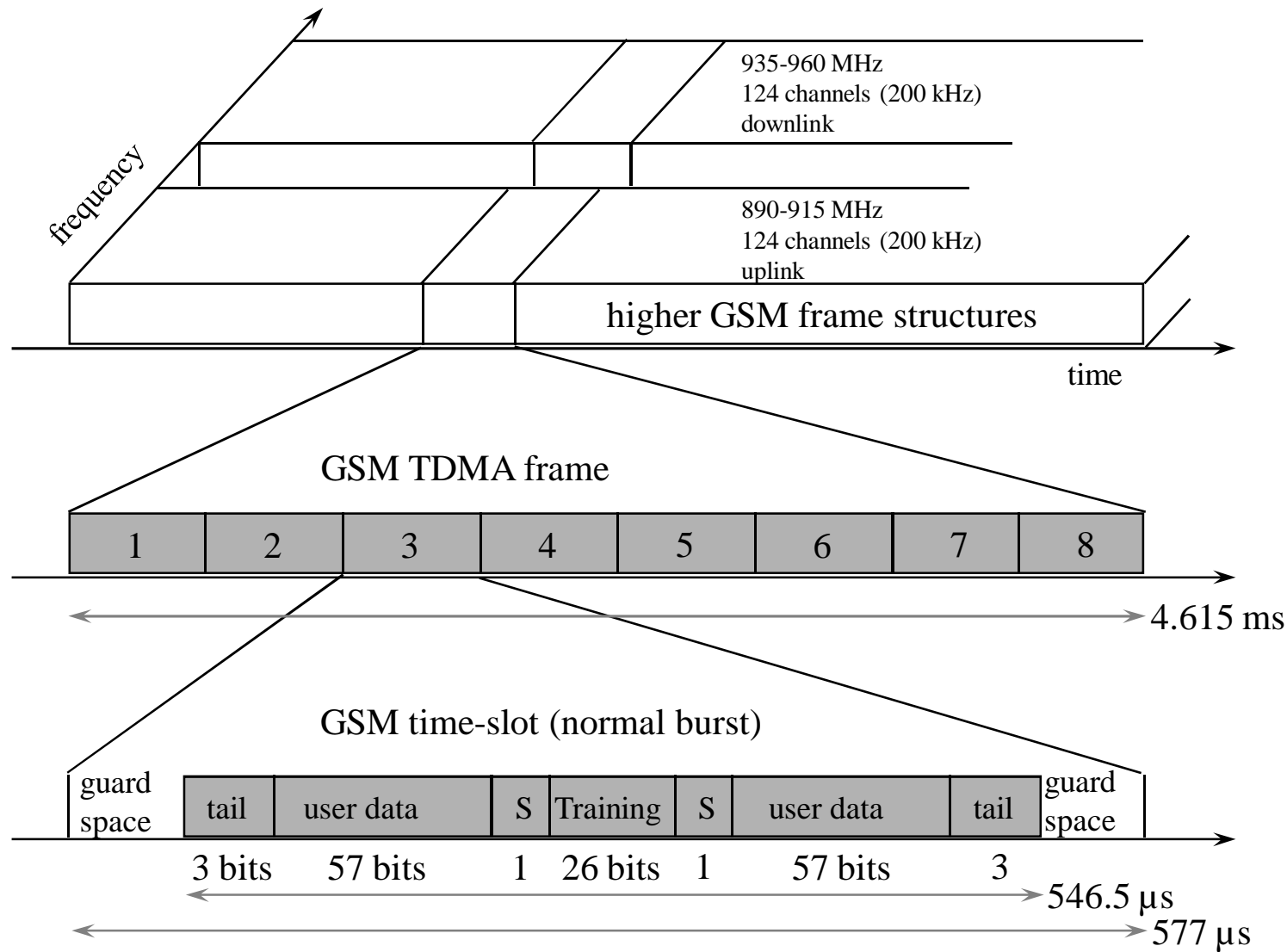


GSM-Frame Structure

GSM - TDMA/FDMA

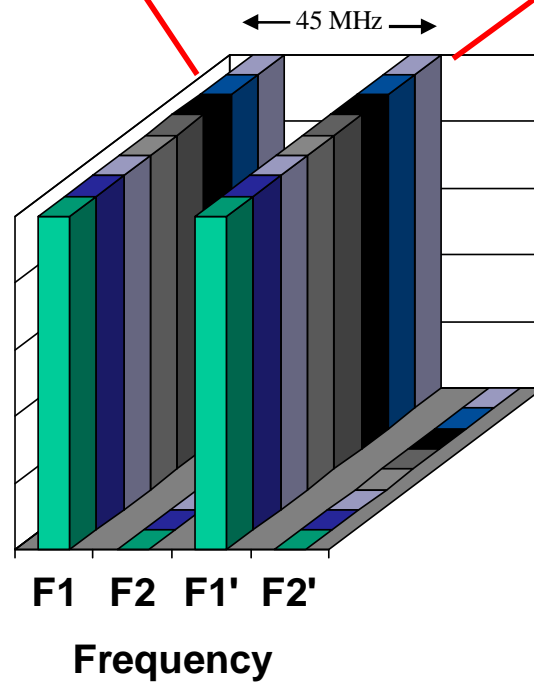


GSM - TDMA/FDMA



MS Transmission
Band : 890 – 915
MHZ

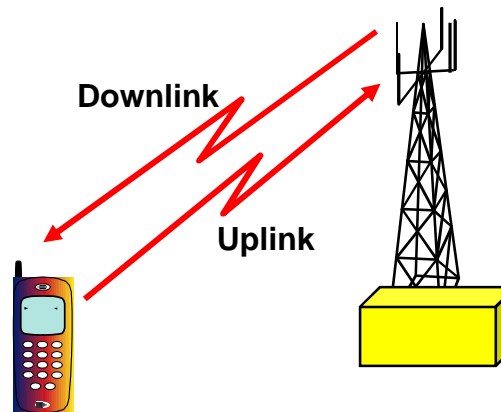
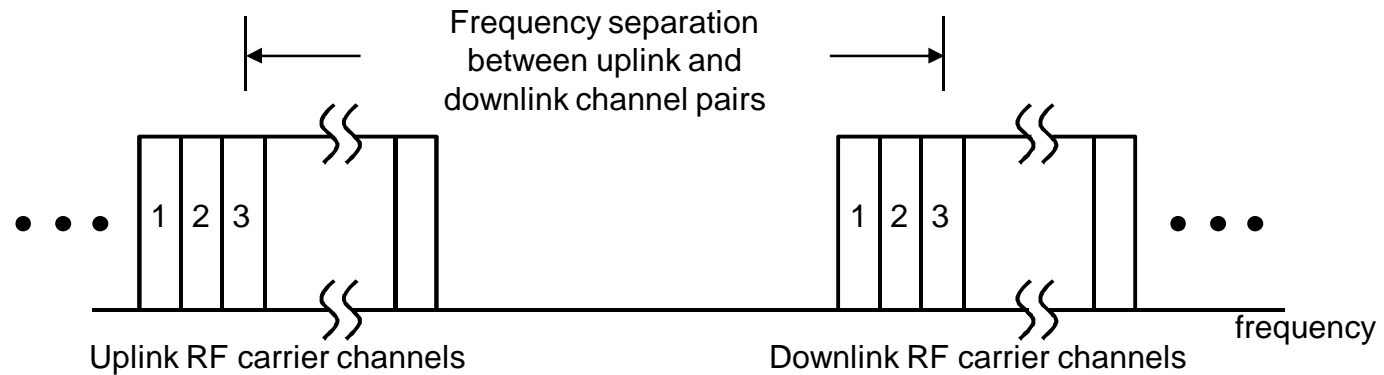
BS Transmission
Band : 935 – 960
MHZ



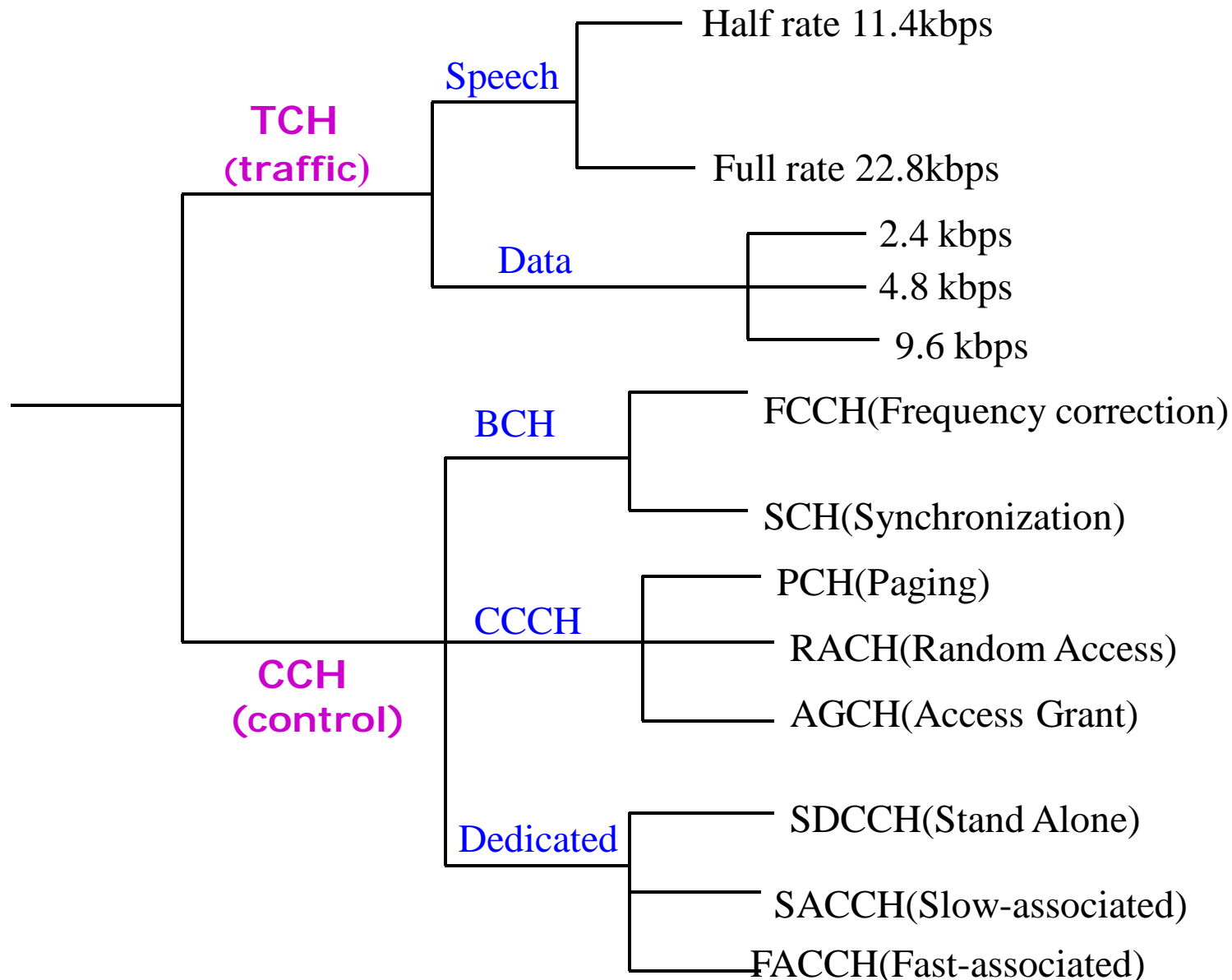
- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8

Channels for Two-Way Communications

Frequency Division Duplex



Logical Channels



Logical Channels

- **Traffic Channels (TCH)**

- TCH/F and TCH/H for full and half rate speech channels.
- TCH/9.6, TCH/4.8 and TCH/2.4 for 9.6, 4.8 and 2.4 kb/s data channels.

- **Broadcast Channels (BCH)**

- Frequency Correction Channel (FCCH),
- Synchronization Channel (SCH),
- Broadcast Control Channel (BCCH).

- **Common control channels**

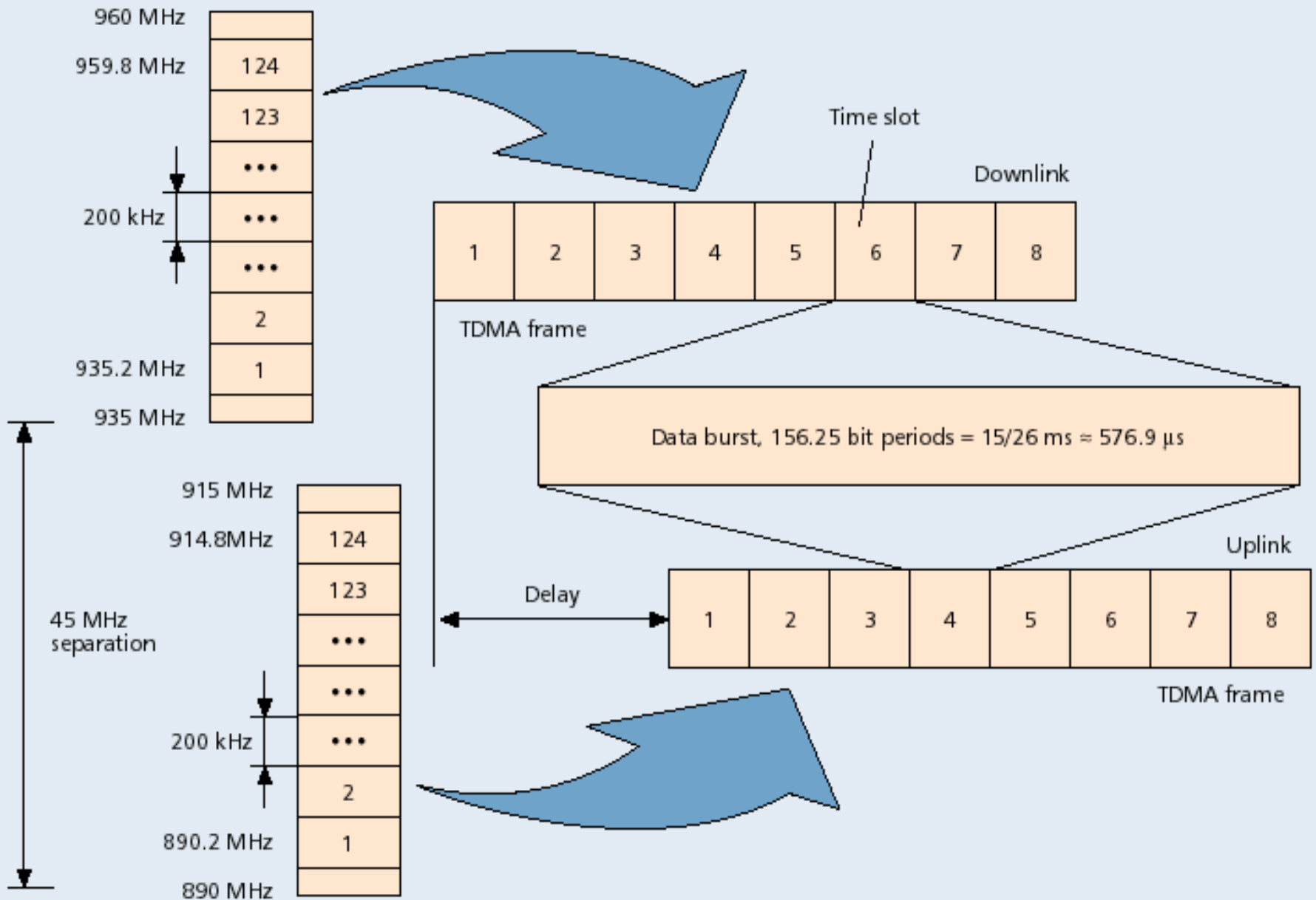
- Paging Channel (PCH),
- Access Grant Channel (AGCH),
- Random Access Channel (RACH).

- **Dedicated control channels**

- Stand-alone Dedicated Control Channel (SDCCH),
- Slow Associated Control Channel (SACCH),
- Fast Associated Control Channel (FACCH).

Number of channels in GSM

- Freq. Carrier: 200 kHz
- TDMA: 8 time slots per freq carrier
- No. of carriers = $25 \text{ MHz} / 200 \text{ kHz} = 125$
- Max no. of user channels = $125 * 8 = 1000$
- Considering guard bands = $124 * 8 = 992$ channels



Air Interface: Logical Channel

- Traffic Channel (TCH)
 - Carries user voice traffic

- Signalling Channel
 - Broadcast Channel (BCH) (unidirectional)
 - Common Control Channel (CCH) (unidirectional)
 - Dedicated/Associated Control Channel (DCCH/ACCH) (bidirectional)

BCCH

- **Broadcast Control Channel (BCCH)**
 - **BTS to MS**
 - send cell identities, organization info about common control channels, cell service available, etc
 - **Radio channel configuration**
 - Current cell + Neighbouring cells
 - **Synchronizing information**
 - Frequencies + frame numbering
 - **Registration Identifiers**
 - LA + Cell Identification (CI) + Base Station Identity Code (BSIC)

FCCH & SCH

▪ Frequency Correction Channel

- send a frequency correction data burst containing all zeros to effect a constant frequency shift of RF carrier
 - Mobile station knows which frequency to use
 - Repeated broadcast of Frequency Bursts

▪ Synchronization Channel

- send TDMA frame no. and base station identity code to synchronize MSs
 - MS knows which timeslot to use
 - Repeated broadcast of Synchronization Bursts

AGCH & PCH

▪ Access Grant Channel (AGCH)

- BTS to MS
- Used to assign an SDCCH/TCH to MS

▪ Paging Channel (PCH)

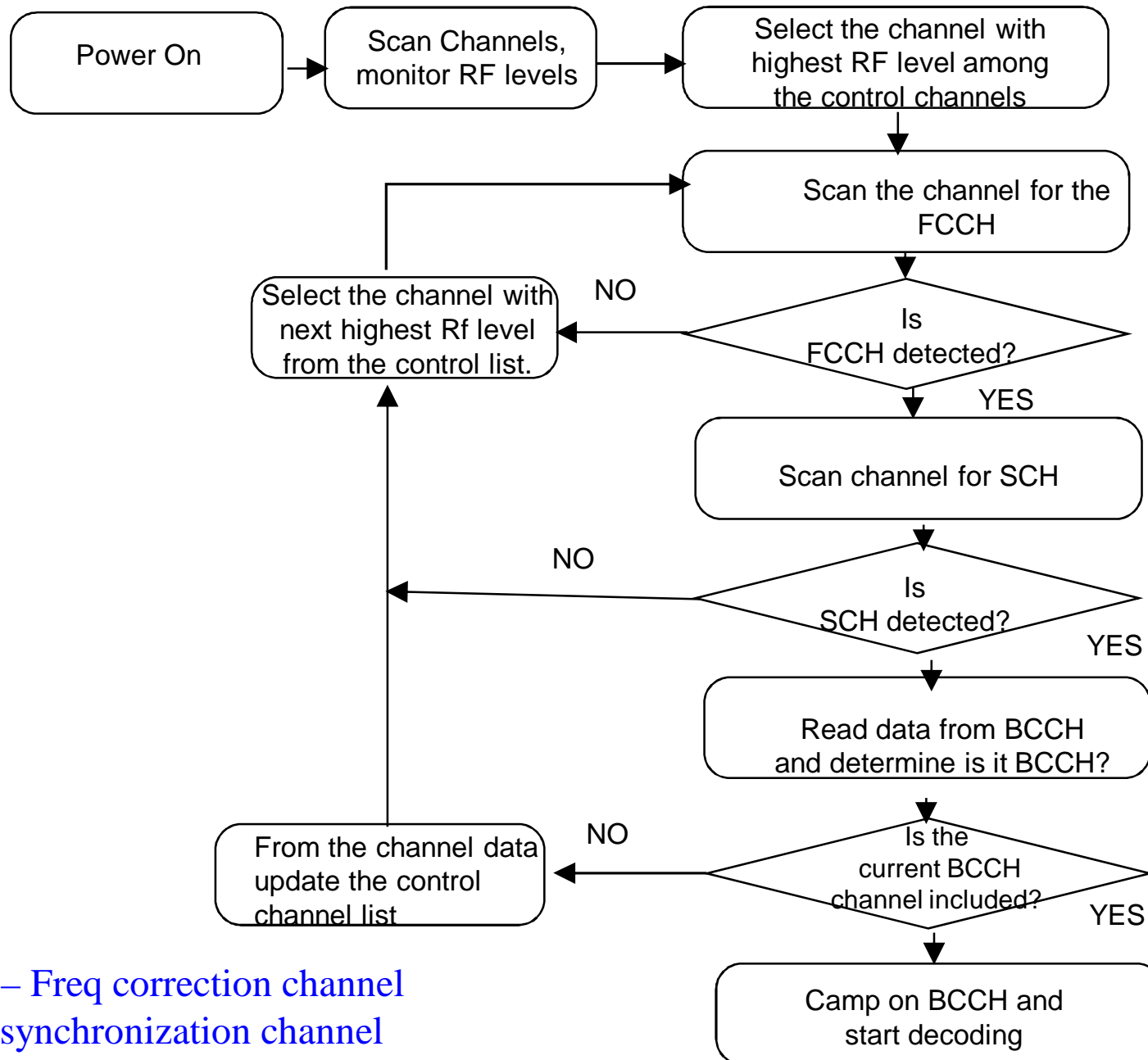
- BTS to MS
- Page MS

RACH & SDCCH

- **Random Access Channel (RACH)**
 - ✓ MS => BTS
 - ✓ Slotted Aloha
 - ✓ Request for dedicated SDCCH
- **Standalone Dedicated Control Channel (SDCCH)**
 - ✓ MS => BTS
 - ✓ Standalone; Independent of Traffic Channel
 - ✓ Used before MS is assigned a TCH

DCCH

- **DCCH (dedicated control channel):**
 - Bidirectional point-to-point main signaling channels
 - SDCCH (stand-alone dedicated control channel):
 - SACCH (slow associated control channel): for out-of-band signaling associated with a traffic channel, eg.: signal strength measurements.
 - FACCH (fast associated control channel): for preemptive signaling on a traffic channel, eg, for handoff messages
Uses timeslots which are otherwise used by the TCH`



FCCH – Freq correction channel

SCH – synchronization channel

GSM: Frequency Hopping

- **Optionally, TDMA is combined with frequency hopping to address problem of channel fading**
 - TDMA bursts are transmitted in a pre-calculated sequence of different frequencies (algorithm programmed in mobile station)
 - If a TDMA burst happens to be in a deep fade, then next burst most probably will not be so
 - Helps to make transmission quality more uniform among all subscribers

Bursts

Building unit of physical channel

Types of bursts

Normal: for transmitting messages in traffic and control channels

Frequency Correction: sent by base station for frequency correction at mobile station

Synchronization: sent by base station for synchronization

Access: for call setup

Dummy: to fill an empty timeslot in the absence of data

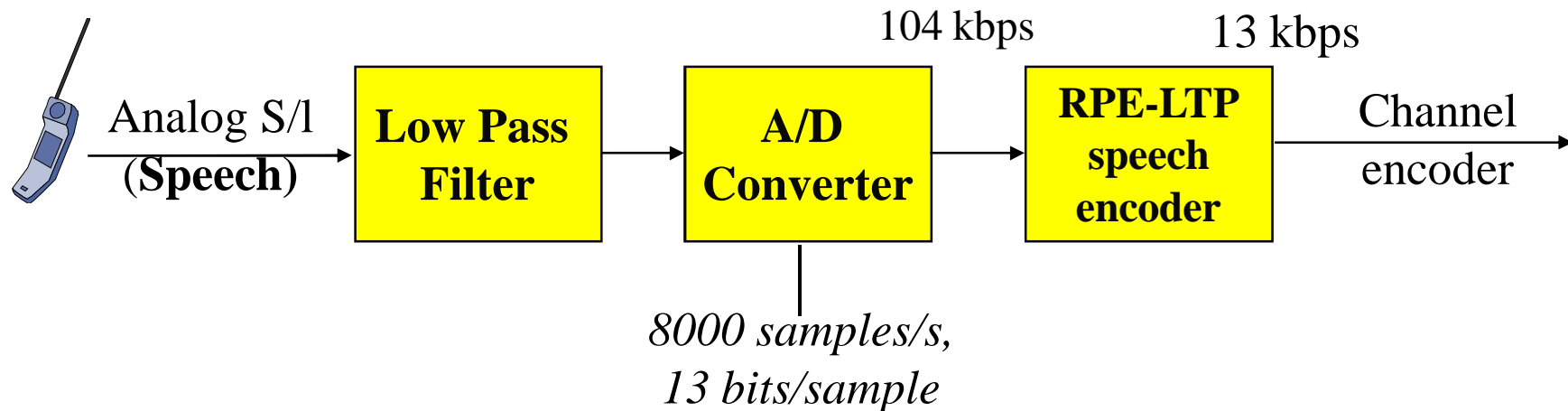
Traffic Channel

- Transfer either encoded speech or user data
- Bidirectional
- Full Rate TCH
 - Rate 22.4kbps
- **Half Rate TCH**
 - Rate 11.2 kbps

Full Rate Speech Coding

- Speech Coding for 20ms segments
 - 260 bits at the output ; Effective data rate 13kbps
- Unequal error protection
 - 182 bits are protected
 - 78 bits unprotected
- Channel Encoding
 - Codes 260 bits into (8 x 57 bit blocks) 456 bits
- Interleaving
 - 2 blocks of different set interleaved on a normal burst (save damages by error bursts)

GSM Speech Coding



SACCH & FACCH

- **Slow Associated Control Channel (SACCH)**
 - Always associated with either TCH or SDCCH
 - Information
 - Channel quality, signal power level
 - Should always be active; as proof of existence of physical radio connection
- **Fast Associated Control Channel (FACCH)**
 - Handover
 - Uses timeslots which are otherwise used by TCH (Pre-emptive multiplexing on a TCH, Stealing Flag (SF))

GSM: Channel Summary

- Logical channels
 - ✓ Traffic Channels; Control Channels

- Physical Channel
 - ✓ Time Slot Number; TDMA frame; RF Channel Sequence

- Mapping in frequency
 - ✓ 124 channels, 200KHz spacing

- Mapping in time
 - ✓ TDMA Frame, Multi Frame, Super Frame, Channel

*Thank
you*

Electromechanical Energy Conversion

By

Mr. M. Manohara

Associate Professor, Dept. of EEE

Unit: I

Course: **DC Machines**

Target Group: **II B.Tech. I Sem EEE**

Introduction

Electromechanical energy conversions – use a magnetic field as the medium of energy conversion

Electromechanical energy conversion device:

Converts electrical energy into mechanical energy

or

Converts mechanical energy into electrical energy.

Introduction

Three categories of electromechanical energy conversion devices:

- Transducers (for measurement and control)- **small motion**
Transform the signals of different forms. Examples: microphones, sensors and speakers.
- Force producing devices (**translational force**)- **limited mechanical motion**.
Produce forces mostly for linear motion drives, Example Actuators - relays, solenoids and electromagnets.
- Continuous energy conversion equipment.
Operate in rotating mode. Examples: motors and generators.

Energy Conversion Process

The principle of conservation of energy:

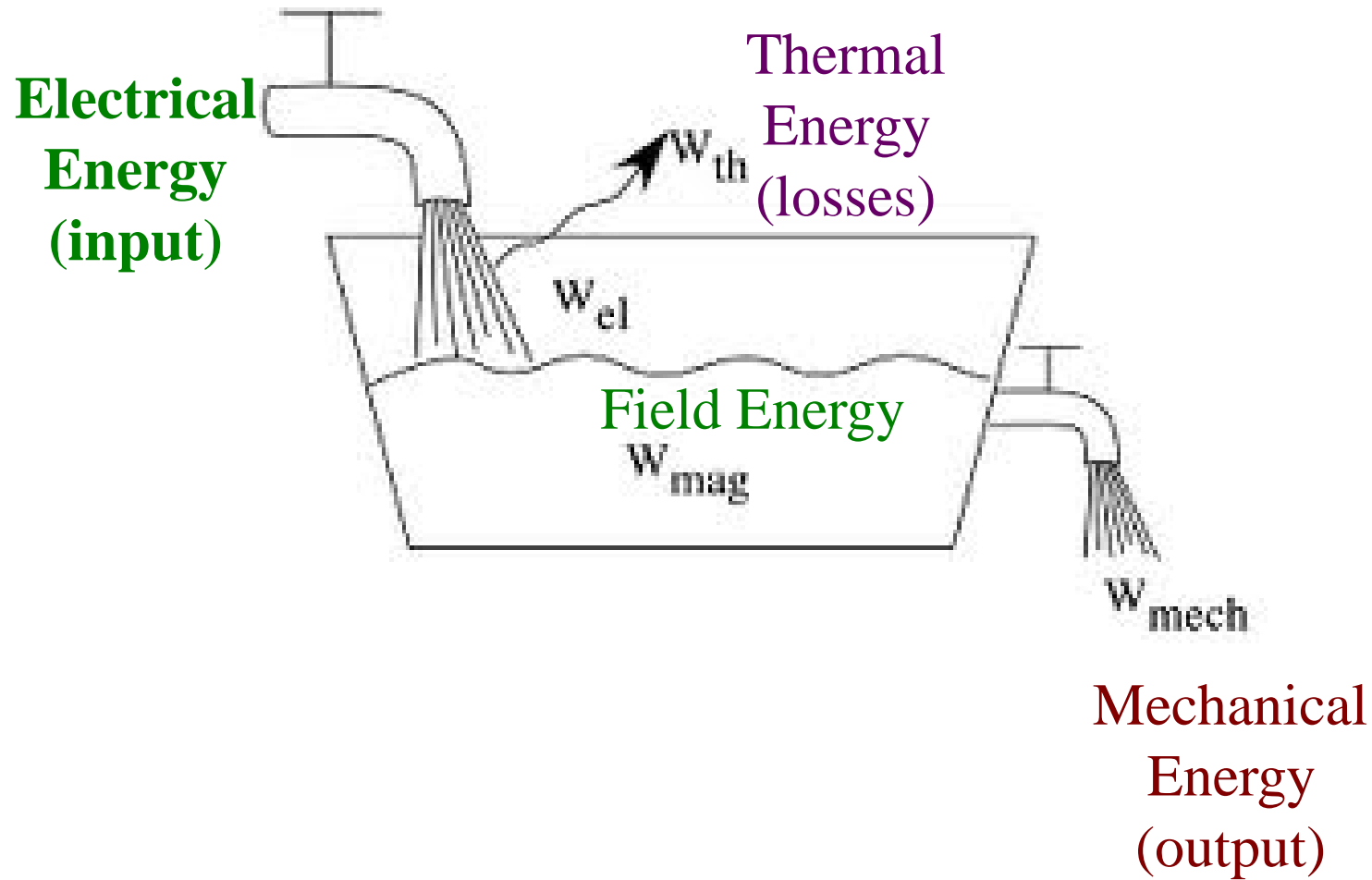
Energy can neither be created nor destroyed. It can only be changed from one form to another. Therefore total energy in a system is constant

Energy Conversion Process

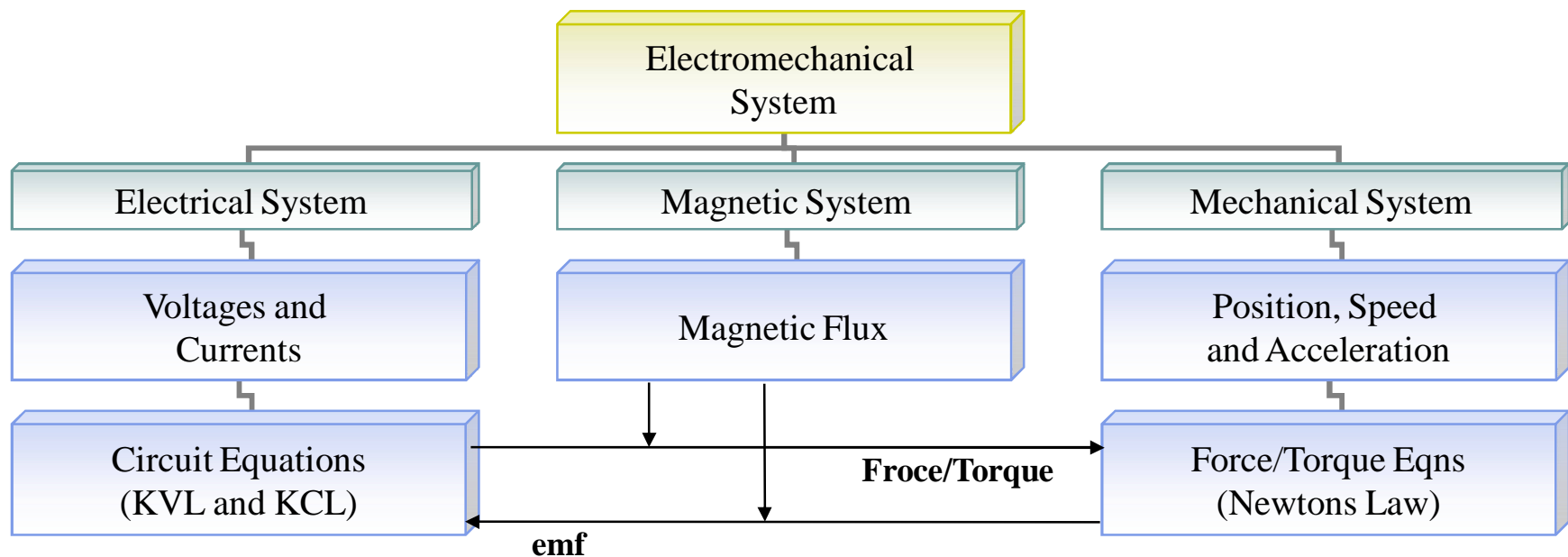
An electromechanical converter system has three essential parts:

- ① An electrical system (electric circuits such as windings)
- ② A magnetic system (magnetic field in the magnetic cores and air gaps)
- ③ A mechanical system (mechanically movable parts such as a rotor in an electrical machine).

EM Energy Conversion: Analogy

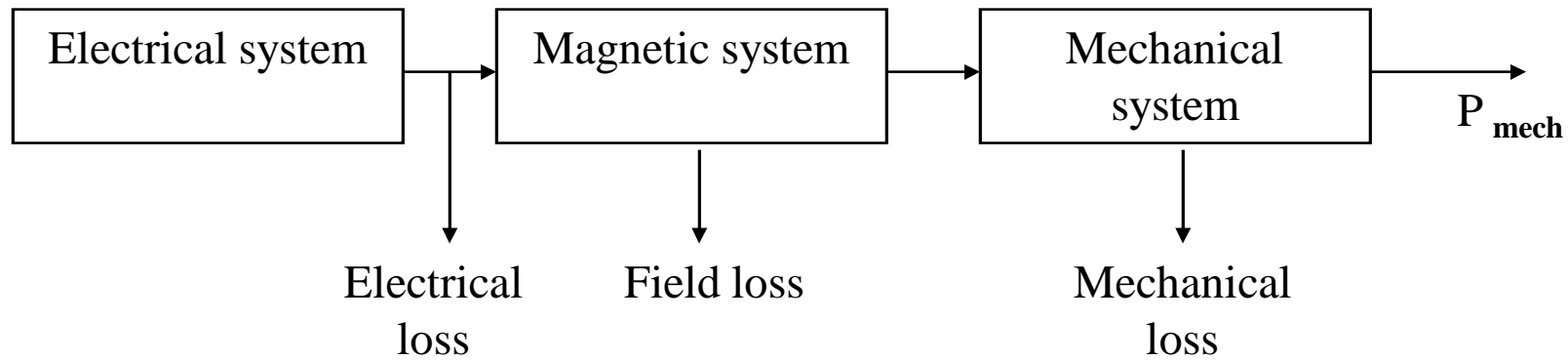


Energy Conversion Process



Concept of electromechanical system modeling

Energy Conversion Process



The energy transfer equation is as follows:

$$\left(\begin{array}{c} \text{Electrical} \\ \text{energy input} \\ \text{from sources} \end{array} \right) = \left(\begin{array}{c} \text{Mechanical} \\ \text{energy} \\ \text{output} \end{array} \right) + \left(\begin{array}{c} \text{Increase in} \\ \text{stored energy in} \\ \text{magnetic field} \end{array} \right) + \left(\begin{array}{c} \text{Energy} \\ \text{losses} \end{array} \right)$$

Energy Conversion Process

The energy balance can therefore be written as:

$$\left(\begin{array}{l} \textit{Electrical energy} \\ \textit{input from sources} \\ \textit{-resistance loss} \end{array} \right) = \left(\begin{array}{l} \textit{Mechanical energy} \\ \textit{output + friction} \\ \textit{and windage loss} \end{array} \right) + \left(\begin{array}{l} \textit{Increase in} \\ \textit{stored field} \\ \textit{energy + core loss} \end{array} \right)$$

For the lossless magnetic energy storage system in differential form,

$$\mathbf{dW}_e = \mathbf{dW}_m + \mathbf{dW}_f$$

$dW_e = i d\lambda$ = differential change in electric energy input

$dW_m = f_m dx$ = differential change in mechanical energy output

dW_f = differential change in magnetic stored energy

Energy Conversion Process

We can write

$$dW_e = e i dt; \quad e = \frac{d\lambda}{dt}$$

$$dW_e = \frac{d\lambda}{dt} i dt = i d\lambda$$

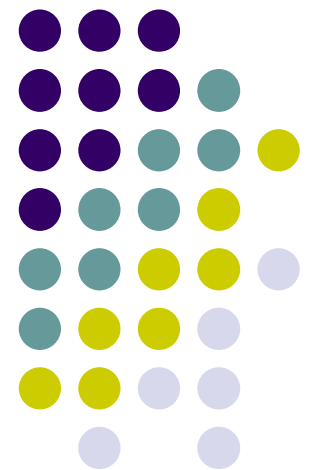
Here e is the voltage induced in the electric terminals by changing magnetic stored energy.

$$dW_e = e i dt = dW_m + dW_f$$

Together with Faraday's law for induced voltage, form the basis for the energy method.

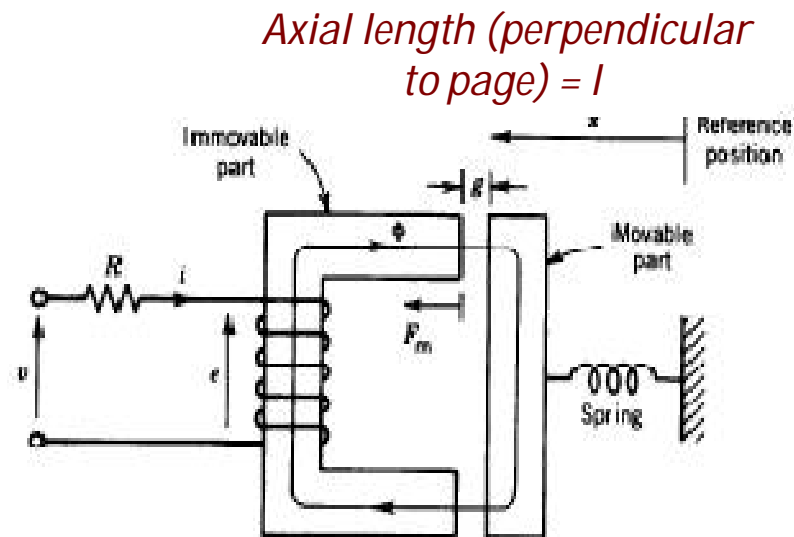
Singly-excited System

Energy, Coenergy
and Force or Torque



Energy in Magnetic System

Consider the electromechanical system below:



Schematic of an electromagnetic relay

Energy in Magnetic System

The mechanical force f_m is defined as acting from the relay upon the external mechanical system and the differential mechanical energy output of the relay is

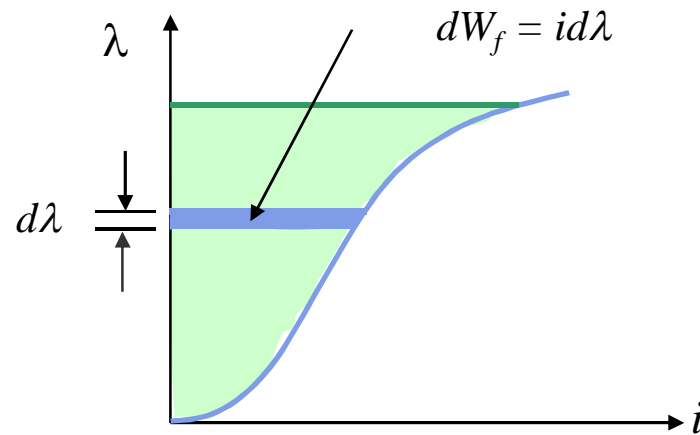
$$dW_m = f_m dx$$

Then, substitution $dW_e = id \lambda$, gives

$$dW_f = id \lambda - f_m dx$$

Value of W_f is uniquely specified by the values of λ and x , since the magnetic energy storage system is lossless.

Energy in Magnetic System



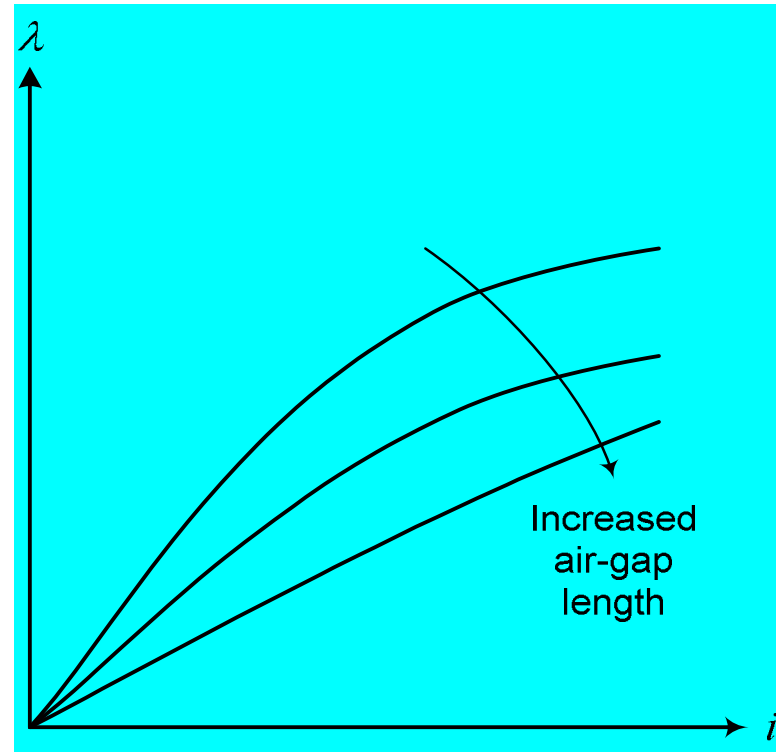
$$W_f = \int id\lambda$$

dW_f = differential change in magnetic stored energy

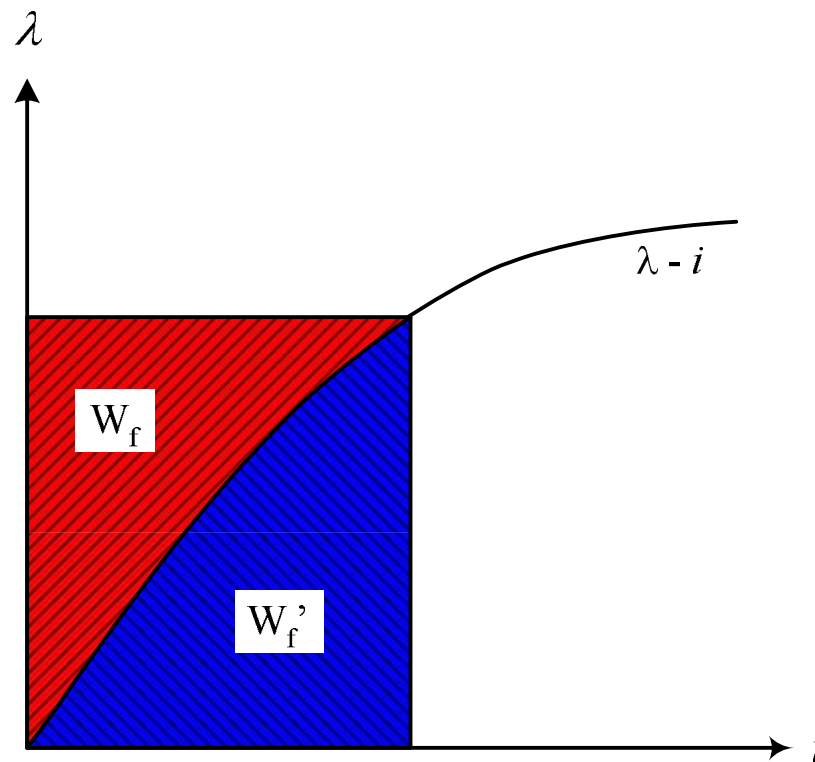
Energy and Coenergy

The λ - i characteristics of an electromagnetic system depends on the air-gap length and B-H characteristics of the magnetic material.

For a larger air-gap length the characteristic is essentially linear. The characteristic becomes non linear as the air-gap length decreases.



Energy and Coenergy



For a particular value of air-gap length, the field energy is represented by the red area between λ axis and $\lambda - i$ characteristic. The blue area between i axis and $\lambda - i$ characteristic is known as the coenergy

Energy and Coenergy

The coenergy is defined as

$$\mathbf{W}_f' = \int_0^i \lambda \mathbf{d}\mathbf{i}$$

From the figure of $\lambda - i$ characteristic,

$$\mathbf{W}_f' + \mathbf{W}_f = \lambda i$$

Note that $W_f' > W_f$ if the $\lambda - i$ characteristic is non linear and $W_f' = W_f$ if it is linear.

The quantity of coenergy has no physical significance. However, it can be used to derive expressions for force (torque) developed in an electromagnetic system

Determination of Force from Energy

The magnetic stored energy W_f is a state function, determined uniquely by the independent state variables λ and x . This is shown explicitly by

$$dW_f(\lambda, x) = i d\lambda - f_m dx$$

Determination of Force from Energy

For any function of two independent variables $F(x_1, x_2)$, the total differential equation of F with respect to the two state variables x_1 and x_2 can be written

$$dF(x_1, x_2) = \left. \frac{\partial F(x_1, x_2)}{\partial x_1} \right|_{x_2} dx_1 + \left. \frac{\partial F(x_1, x_2)}{\partial x_2} \right|_{x_1} dx_2$$

Determination of Force from Energy

Therefore, for the total differential of W_f

$$dW_f(\lambda, x) = \left. \frac{\partial W_f(\lambda, x)}{\partial \lambda} \right|_x d\lambda + \left. \frac{\partial W_f(\lambda, x)}{\partial x} \right|_\lambda dx$$

And we know that

$$dW_f(\lambda, x) = i d\lambda - f_m dx$$

Determination of Force from Energy

By matching both equations, the current:

$$i = \left. \frac{\partial \mathcal{W}_f(\lambda, x)}{\partial \lambda} \right|_x$$

where the partial derivative is taken while holding x constant and the mechanical force:

$$f_m = \left. \frac{\partial \mathcal{W}_f(\lambda, x)}{\partial x} \right|_\lambda$$

where the partial derivative is taken while holding λ constant.

Determination of Force from Energy: Linear System

For a linear magnetic system for which $\lambda=L(x)i$:

$$W_f(\lambda, x) = \int_0^\lambda i(\lambda, x) d\lambda = \int_0^\lambda \frac{\lambda}{L(x)} d\lambda = \frac{1}{2} \frac{\lambda^2}{L(x)}$$

and the force, f_m can be found directly:

$$f_m = -\left. \frac{\partial W_f(\lambda, x)}{\partial x} \right|_\lambda = -\left. \frac{\partial}{\partial x} \left(\frac{1}{2} \frac{\lambda^2}{L(x)} \right) \right|_\lambda = \frac{\lambda^2}{2L(x)^2} \frac{dL(x)}{dx}$$

Determination of Torque from Energy

For a system with a rotating mechanical terminal, the mechanical terminal variables become the angular displacement θ and the torque T .

Therefore, equation for the torque:

$$T = - \left. \frac{\partial \mathcal{W}_f(\lambda, \theta)}{\partial \theta} \right|_{\lambda}$$

where the partial derivative is taken while holding λ constant.

Determination of Force from Coenergy

The coenergy W_f' is defined as

$$W_f'(i, x) = i\lambda - W_f(\lambda, x)$$

and the differential coenergy dW_f' :

$$dW_f'(i, x) = d(i\lambda) - dW_f(\lambda, x)$$

We know previously that

$$dW_f(\lambda, x) = i d\lambda - f_m dx$$

Determination of Force from Coenergy

By expanding $d(i\lambda)$:

$$d(i\lambda) = id\lambda + \lambda di$$

So, the differential coenergy dW_f' :

$$\begin{aligned}dW_f'(i, x) &= d(i\lambda) - dW_f(\lambda, x) \\ &= id\lambda + \lambda di - (id\lambda - f_m dx) \\ &= \lambda di + f_m dx\end{aligned}$$

Determination of Force from Coenergy

By expanding $dW_f'(i, x)$:

$$dW_f'(i, x) = \left. \frac{\partial W_f'(i, x)}{\partial i} \right|_x di + \left. \frac{\partial W_f'(i, x)}{\partial x} \right|_i dx$$

and, from the previous result:

$$dW_f'(i, x) = \lambda di + f_m dx$$

Determination of Force from Coenergy

By matching both equations, λ :

$$\lambda = \left. \frac{\partial \mathcal{W}'_f(i, x)}{\partial i} \right|_x$$

where the partial derivative is taken while holding x constant and the mechanical force:

$$f_m = \left. \frac{\partial \mathcal{W}'_f(i, x)}{\partial x} \right|_i$$

where the partial derivative is taken while holding i constant.

Determination of Force from Coenergy: Linear System

For a linear magnetic system for which $\lambda=L(x)i$:

$$W_f'(i, x) = \int_0^i \lambda(i, x) di = \int_0^i L(x)idi = L(x) \frac{i^2}{2}$$

and the force, f_m can be found directly:

$$f_m = \left. \frac{\partial W_f'(i, x)}{\partial x} \right|_i = \left. \frac{\partial}{\partial x} \left(L(x) \frac{i^2}{2} \right) \right|_i = \frac{i^2}{2} \frac{dL(x)}{dx}$$

Determination of Torque from Coenergy

For a system with a rotating mechanical terminal, the mechanical terminal variables become the angular displacement θ and the torque T .

Therefore, equation for the torque:

$$T = \left. \frac{\partial \mathcal{W}'_f(i, \theta)}{\partial \theta} \right|_i$$

where the partial derivative is taken while holding λ constant.

Determination of Force Using Energy or Coenergy?

The selection of energy or coenergy as the function to find the force is purely a matter of convenience.

They both give the same result, but one or the other may be simpler analytically, depending on the desired result and characteristics of the system being analyzed.

Direction of Force Developed

1. By using energy function: $f_m = -\frac{\partial W_f(\lambda, x)}{\partial x} \Big|_{\lambda}$

The negative sign shows that the force acts in a direction to decrease the magnetic field stored energy at constant flux.

2. By using coenergy function: $f_m = +\frac{\partial W'_f(i, x)}{\partial x} \Big|_i$

The positive sign emphasizes that the force acts in a direction to increase the coenergy at constant current.

Direction of Force Developed

3. By using inductance function:

$$f_m = + \frac{i^2}{2} \frac{dL(x)}{dx} \Big|_i$$

The positive sign emphasizes that the force acts in a direction to increase the inductance at constant current.

B-H Curve and Energy Density

In a magnetic circuit having a substantial air gap g , and high permeability of the iron core, nearly all the stored energy resides in the gap. Therefore, in most of the cases we just need to consider the energy stored in the gap. The magnetic stored energy,

$$W_f = \int_0^\lambda i d\lambda$$

in which $i = \frac{Hg}{N}$ and $d\lambda = d(N\phi) = d(NAB) = NAdB$

B-H Curve and Energy Density

Therefore,
$$W_f = \int_0^B \frac{Hg}{N} NAdB = Ag \int_0^B HdB$$

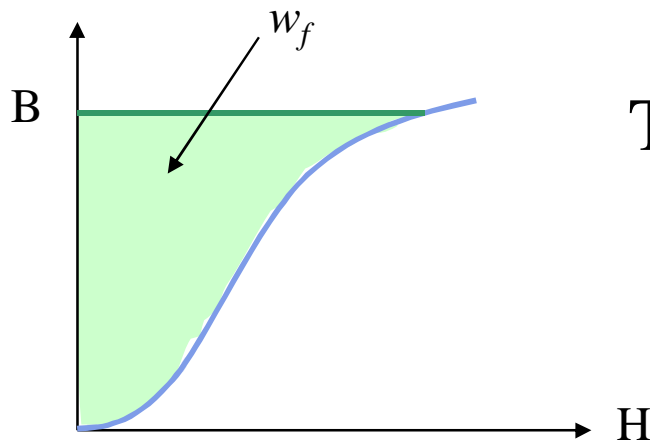
However, A_g is volume of the air gap. Dividing both sides of the above equation by the volume A_g results in

$$w_f = \frac{W_f}{A_g} = \int_0^B HdB$$

B-H Curve and Energy Density

where $w_f = \int_0^B H dB$ is energy per unit volume

w_f is known as energy density.

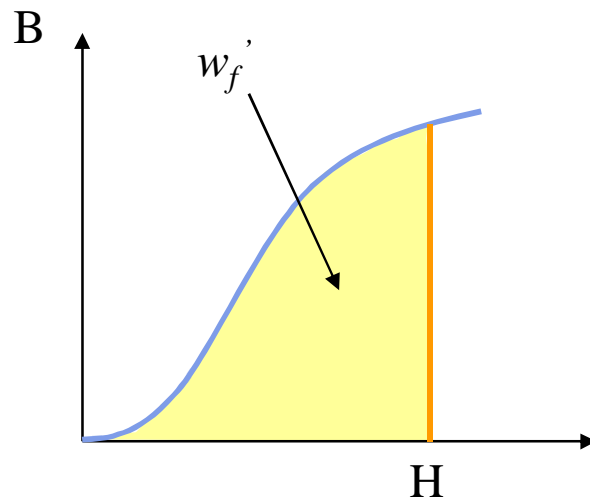


The area between the B-H curve and B axis represents the energy density in the air gap.

B-H Curve and Energy Density

In the same manner,

$$w_f' = \int_0^H B dH \quad \text{is coenergy per unit volume.}$$



The area between the B-H curve and H axis represents the coenergy density in the air gap.

B-H Curve and Energy Density

For a linear magnetic circuit, $B = \mu H$ or $H = B/\mu$, energy density:

$$w_f = \int_0^B H dB = \int_0^B \frac{B}{\mu} dB = \frac{B^2}{2\mu}$$

and coenergy density:

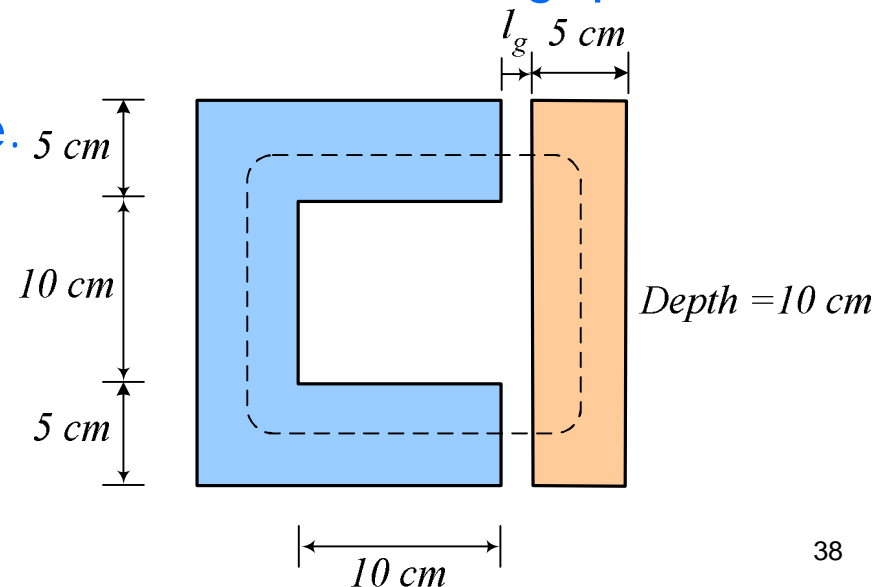
$$w_f' = \int_0^H B dH = \int_0^H \mu H dH = \frac{\mu H^2}{2}$$

In this case, it is obvious that $w_f = w_f'$.

Example 3.1 PC Sen

The dimensions of the relay system are shown in figure below. The magnetic core is made of cast steel whose B-H characteristic is shown in Figure 1.7 (pg.6). The coil has 300 turns, and the coil resistance is 6 ohms. For a fixed air-gap length $l_g = 4 \text{ mm}$, a dc source is connected to the coil to produce a flux density of 1.1 Tesla in the air-gap. Calculate

- (a) The voltage of the dc source.
- (b) The stored field energy.



Pg:99 PC Sen

Example 3.2 PC Sen

The λ - i relationship for an electromagnetic system is given by

$$i = \left(\frac{\lambda g}{0.09} \right)^2$$

which is valid for the limits $0 < i < 4$ A and $3 < g < 10$ cm. For current $i = 3$ A and air gap length $g = 5$ cm, find the mechanical force on the moving part using coenergy and energy of the field.

-124.7 Nm
pg103 sen

Example 3.3 PC Sen

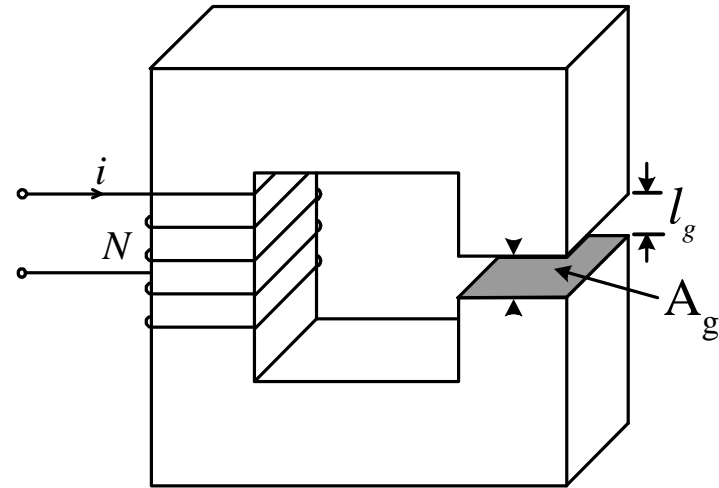
The magnetic system shown in the Figure has the following parameters:

$$N = 400, i = 3 \text{ A}$$

Width of air-gap = 2.5 cm

Depth of air-gap = 2.5 cm

Length of air-gap = 1.5 mm



Neglect the reluctance of the core, leakage flux and the fringing flux. Determine:

- The force of attraction between both sides of the air-gap
- The energy stored in the air-gap.
- Coil Inductance

Sen pg 106

Example 3.4 PC Sen

The lifting magnetic system is shown, with a square cross section area $6 \times 6 \text{ cm}^2$. The coil has 300 turns and a resistance of 6 ohms.

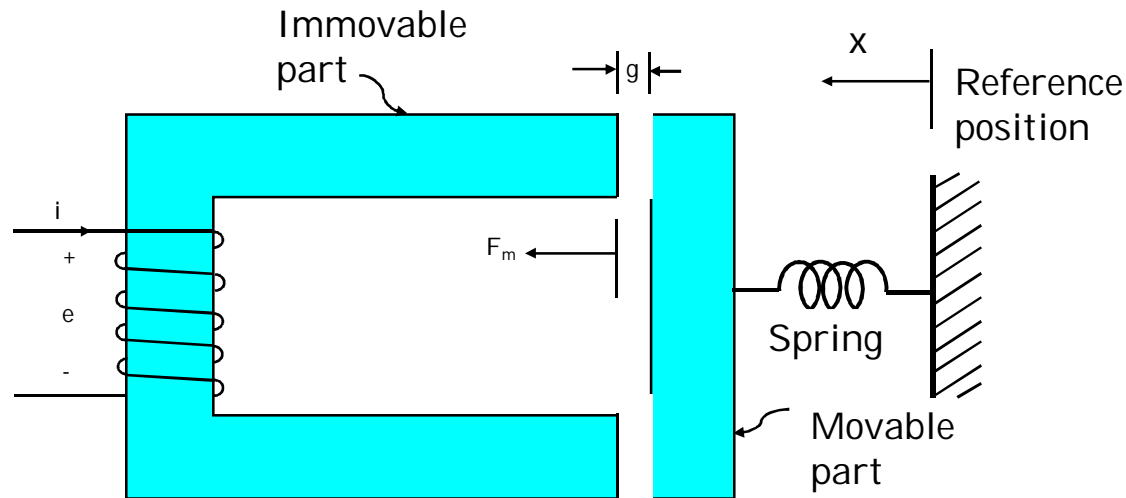
Neglect core reluctance and fringing effect.

- a) The air gap is initially 5mm and a dc source of 120 V is connected to the coil. Determine the stored field energy and the lifting force
- b) The air gap is held at 5 mm and an ac source of 120 Vrms at 60 Hz is supplied to the coil. Determine the average value of the lift force

Sen 107

Example 1

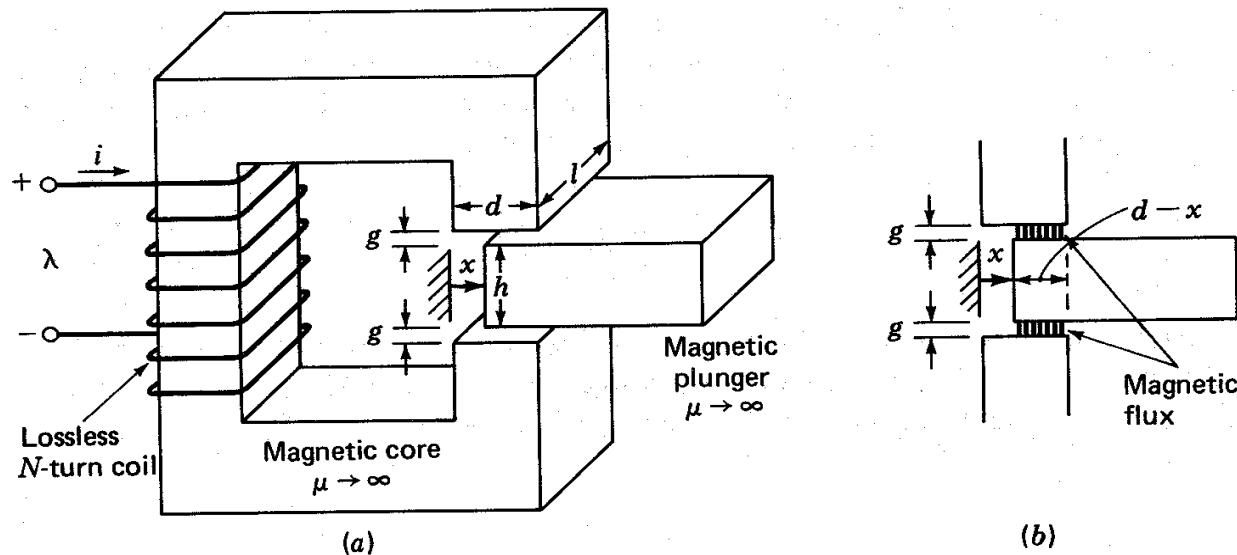
- Q. The magnetic circuit shown in Figure Q1 is made of high permeability steel so that its reluctance can be negligible. The movable part is free to move about an x-axis. The coil has 1000 turns, the area normal to the flux is ($5 \text{ cm} \times 10 \text{ cm}$), and the length of a single air gap is 5 mm.
- Derive an expression for the inductance, L , as a function of air gap, g .
 - Determine the force, F_m , for the current $i = 10 \text{ A}$.
 - The maximum flux density in the air gaps is to be limited to approximately 1.0 Tesla to avoid excessive saturation of the steel. Compute the maximum force.



Example 2

Figure below shows a relay made of infinitely-permeable magnetic material with a moveable plunger (infinitely-permeable material). The height of the plunger is much greater than air gap length ($h \gg g$). Calculate

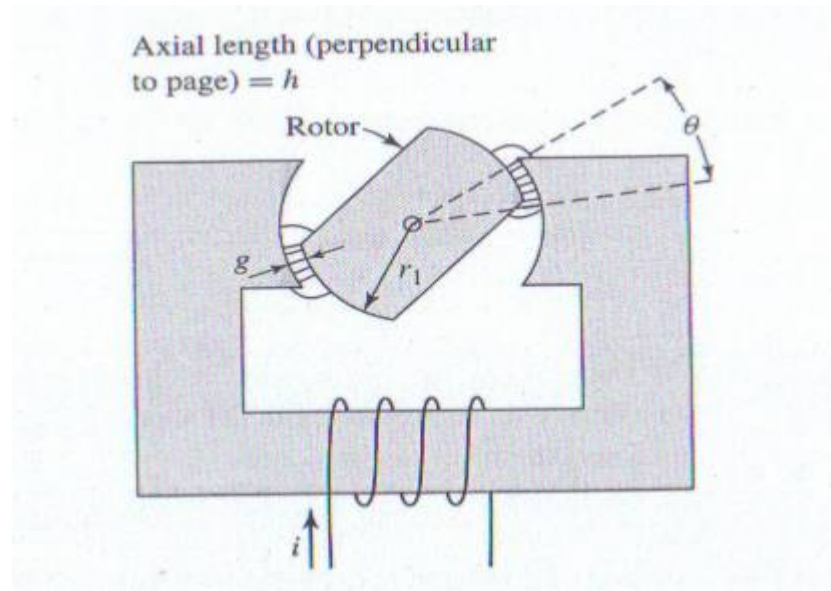
- The magnetic storage energy W_f as a function of plunger position ($0 < x < d$) for $N = 1000$ turns, $g = 2$ mm, $d = 0.15$ m, $l = 0.1$ m and $i = 10$ A.
- The generated force, F_m



b)Pg 121/
132 Fgrld

Example 3

The magnetic circuit shown is made of high-permeability electrical steel. Assume the reluctance of steel $\mu \rightarrow \infty$. Derive the expression for the torque acting on the rotor .



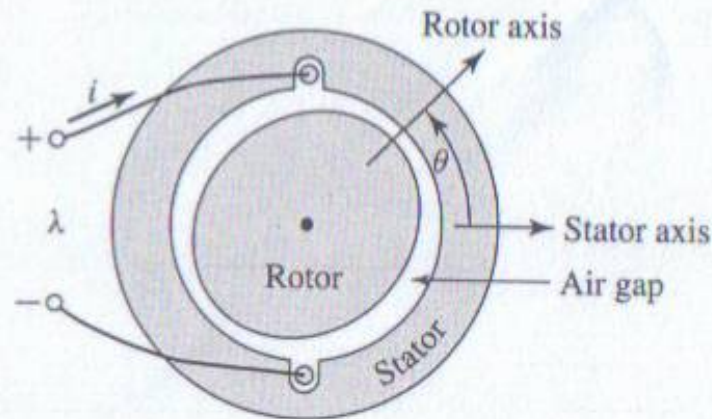
Fgrd pg
135

Example 4

The magnetic circuit below consists of a single coil stator and an oval rotor. Because of the air-gap is non uniform, the coil inductance varies with the rotor angular position.

Given coil inductance $L(\theta) = L_0 + L_2 \cos 2\theta$, where $L_0 = 10.6$ mH and $L_2 = 2.7$ mH.

Find torque as a function of θ for a coil current of 2 A.



Exercise- PC Sen pg 115 , prob.3.3

- An actuator system is shown. All dimensions are in cm. The magnetic material is cast steel, magnetisation as shown. The magnetic core and air gap have a square cross-section area. The coil has 500 turns and 4 ohm resistance.
 - a) The gap is $d = 1\text{ mm}$
 - i) Determine the coil current and supply voltage required to establish an air gap flux density of 0.5 T
 - ii. Determine the stored energy in the actuator system
 - iii. Determine the force of attraction on the armature arm
 - iv. Determine the coil inductance

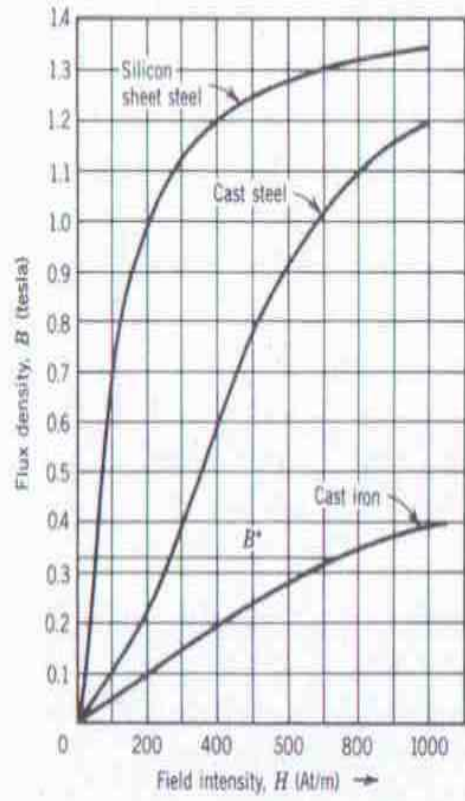


FIGURE 1.7 Magnetization curves.

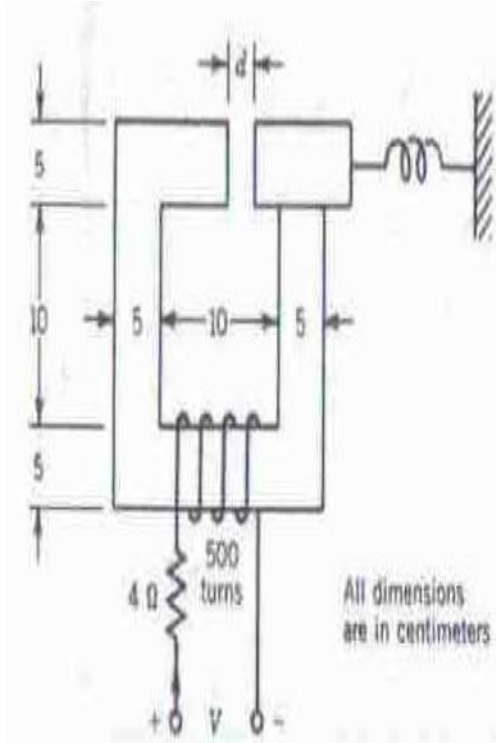
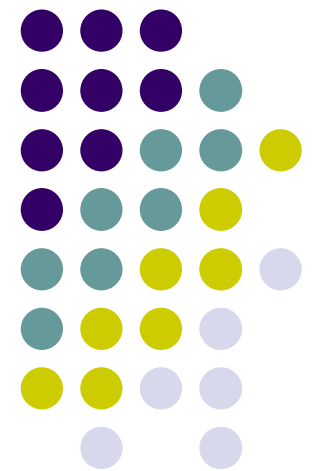


FIGURE P3.3

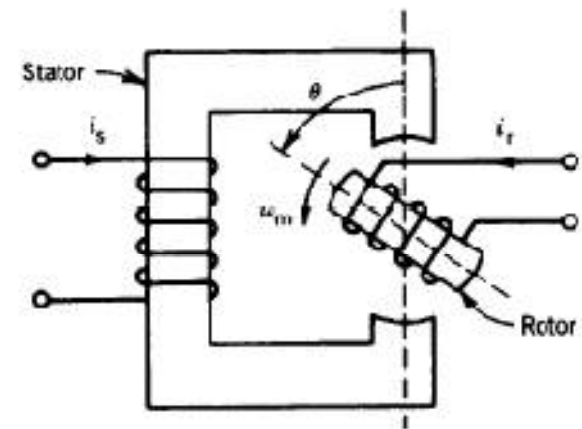
Doubly-excited Systems

Energy, Coenergy
and Force or Torque



Rotating Machines

- Most of the energy converters, particularly the higher-power ones, produce rotational motion.
- The essential part of a rotating electromagnetic system is shown in the figure.
- The fixed part is called the *stator*, the moving part is called the *rotor*.
- The rotor is mounted on a shaft and is free to rotate between the poles of the stator
- Let consider general case where both stator & rotor have windings carrying current (i_s and i_r)



Rotating Machines

- Assume general case, both stator and rotor have winding carrying currents (non-uniform air gap – silent pole rotor)
- The system stored field energy, W_f can be evaluated by establishing the stator current i_s and rotor current i_r and let system static, i.e. no mechanical output

$$dW_f = e_s i_s dt + e_r i_r dt$$

$$= i_s d\lambda_s + i_r d\lambda_r$$

$$\lambda_s = L_{ss} i_s + L_{sr} i_r$$

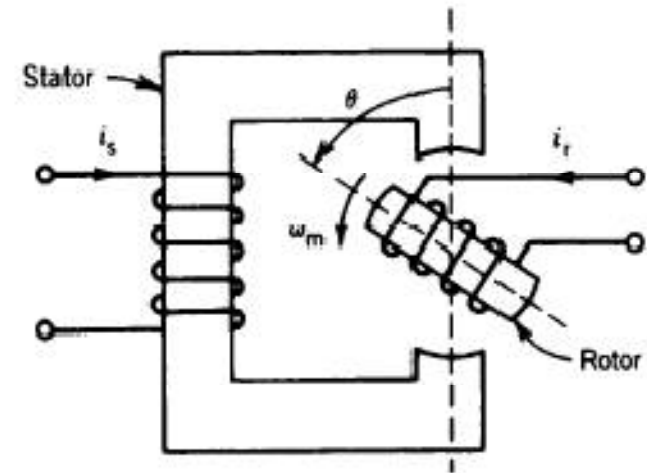
$$\lambda_r = L_{rs} i_s + L_{rr} i_r$$

$$\begin{bmatrix} \lambda_s \\ \lambda_r \end{bmatrix} = \begin{bmatrix} L_{ss} & L_{sr} \\ L_{sr} & L_{rr} \end{bmatrix} \begin{bmatrix} i_s \\ i_r \end{bmatrix}$$

Stator and rotor flux linkage λ is expressed in terms of inductances L (which depends on position rotor angle θ , $L(\theta)$)

$$dW_f = i_s d(L_{ss} i_s + L_{sr} i_r) + i_r d(L_{sr} i_s + L_{rr} i_r)$$

$$= L_{ss} i_s di_s + L_{rr} i_r di_r + L_{sr} d(i_s i_r)$$



Rotating Machines

- Stored field energy $W_f = L_{ss} \int_0^{i_s} i_s di_s + L_{rr} \int_0^{i_r} i_r di_r + L_{sr} \int_0^{i_s i_r} d(i_s i_r)$
 $= \frac{1}{2} L_{ss} i_s^2 + \frac{1}{2} L_{rr} i_r^2 + L_{sr} i_s i_r$

- Torque $T = \left. \frac{\partial W'_f(i, \theta)}{\partial \theta} \right|_{i=\text{constant}} \quad X \rightarrow \theta$

In linear system,
 coenergy = energy
 $W'_f = W_f$

$$T = \frac{1}{2} i_s^2 \frac{dL_{ss}}{d\theta} + \frac{1}{2} i_r^2 \frac{dL_{rr}}{d\theta} + i_s i_r \frac{dL_{sr}}{d\theta}$$

- First two terms represents reluctance torque; variation of self inductance (exist in both salient stator and rotor, or in either stator or rotor is salient)
- The third term represents alignment torque; variation of mutual inductance.

Reluctance Torque – It is caused by the tendency of the induced pole to align with excited pole such that the minimum reluctance is produced. At least one or both of the winding must be excited.

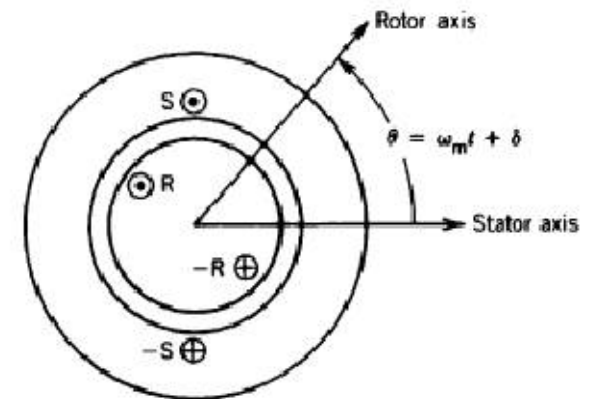
Alignment Torque – It is caused by a tendency of the excited rotor to align with excited stator so as to maximize the mutual inductance. Both winding must be excited.

Cylindrical Machines

- Reluctance machines are simple in construction, but torque developed in these machines is small.
- Cylindrical machines, although more complex in construction, produce larger torques.
- Most electrical machines are of the cylindrical type.

Cylindrical Machines

- A cross sectional view of an elementary two pole cylindrical rotating machine (uniform air gap) is shown.
- The stator and rotor windings are placed on two slots.
- In the actual machine the windings are distributed over several slots.
- If the effects of the slots are neglected, the reluctance of the magnetic path is independent of the position of the rotor.
- Assumed L_{ss} and L_{rr} are constant (i.e no reluctance torque produced).
- Alignment torque is caused by the tendency of the excited rotor to align with the excited stator, depends on mutual inductance



Cylindrical machines

- Torque produced

$$T = i_s i_r \frac{dL_{sr}}{d\theta} = i_s i_r \frac{dM \cos \theta}{d\theta} = M i_s i_r \sin \theta$$

T_m when $\theta=90^\circ$

- Mutual inductance

$$L_{sr} = M \cos \theta$$

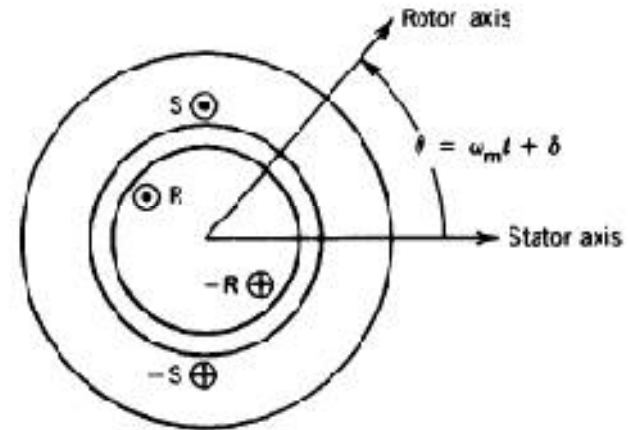
- Currents

$$i_s = I_{sm} \cos \omega_s t$$

$$i_r = I_{rm} \cos(\omega_r t + \alpha)$$

- Rotor position

$$\theta = \omega_m t + \delta$$



Where

M = peak value of mutual inductance
 θ = the angle between magnetic axis of the stator and rotor windings
 ω_m = angular velocity of rotor

Cylindrical Machines

$$T = -I_{sm} I_{rm} M \cos \omega_s t \cos(\omega_r t + \alpha) \sin(\omega_m t + \delta)$$

$$T = -\frac{I_{sm} I_{rm} M}{4} \left[\begin{array}{l} \sin\{(\omega_m + (\omega_s + \omega_r))t + \alpha + \delta\} + \\ \sin\{(\omega_m - (\omega_s + \omega_r))t - \alpha + \delta\} + \\ \sin\{(\omega_m + (\omega_s - \omega_r))t - \alpha + \delta\} + \\ \sin\{(\omega_m - (\omega_s - \omega_r))t + \alpha + \delta\} \end{array} \right]$$

- Torque in general varies sinusoidally with time
- Average value of each term is zero **unless** the coefficient of t is zero

Cylindrical Machines

- Non zero average torque exists/develop only if

$$\omega_m = \pm(\omega_s \pm \omega_r) \quad |\omega_m| = |\omega_s \pm \omega_r|$$

Machine develop torque if sum or difference of the angular speed of the stator and rotor current

Case 1: $\omega_r = 0$ $\omega_m = \omega_s$ $\alpha = 0$

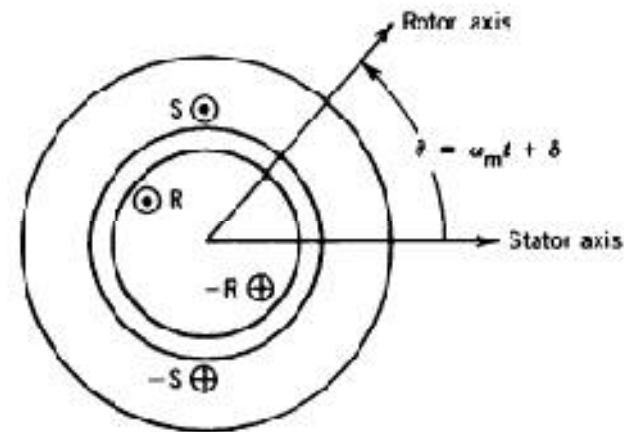
- Synchronous machine

$W_r = 0$ – Idc at rotor

$$T = -\frac{I_{sm} I_R M}{2} \{ \sin(2\omega_s t + \delta) + \sin \delta \}$$

$$T_{avg} = -\frac{I_{sm} I_R M}{2} \sin \delta$$

- Single phase machine
- Pulsating torque
- Polyphase machine minimize pulsating torque
- Not self starting ($\omega_m = 0 \rightarrow T_{avg} = 0$)



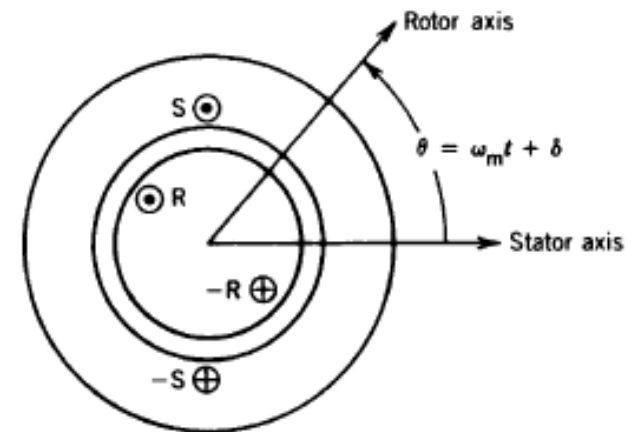
Cylindrical Machines

- Asynchronous machines $\omega_m = \omega_s - \omega_r$ $\omega_m \neq \omega_r$ $\omega_m \neq \omega_s$

$$T = -\frac{I_{sm}I_{rm}M}{4} \left[\begin{array}{l} \sin(2\omega_s t + \alpha + \delta) + \sin(-2\omega_r t - \alpha + \delta) + \\ \sin(2\omega_s t - 2\omega_r t - \alpha + \delta) + \sin(\alpha + \delta) \end{array} \right]$$

$$T_{avg} = -\frac{I_{sm}I_{rm}M}{4} \sin(\alpha + \delta)$$

- Single phase machine
- Pulsating torque
- Not self starting



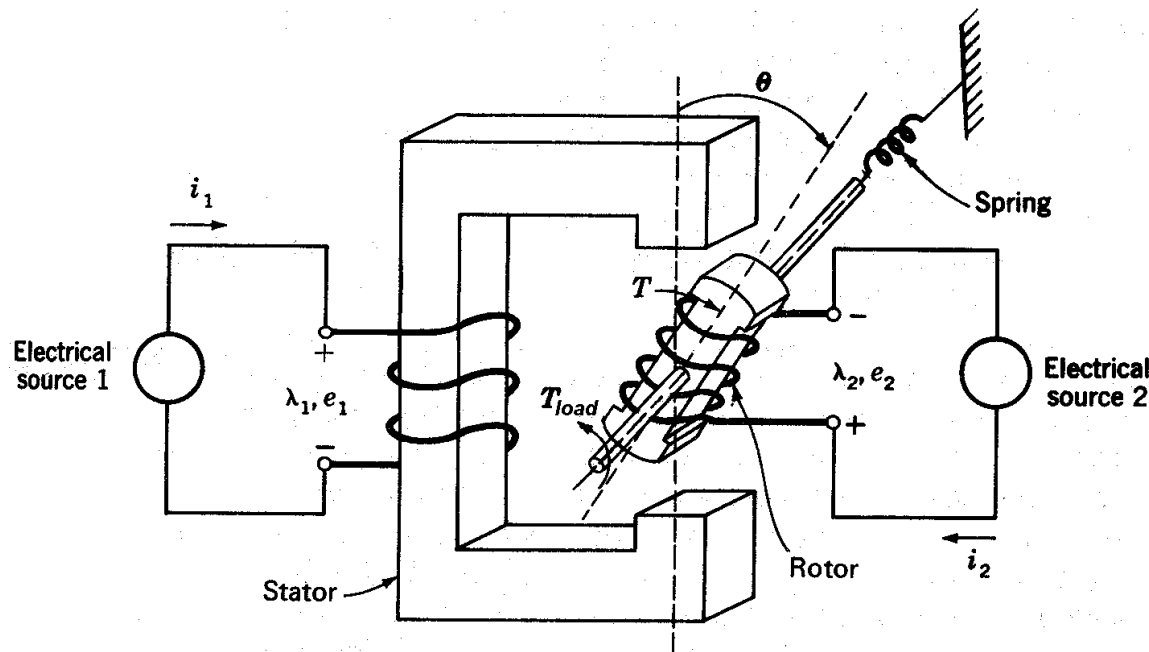
- Polyphase machine minimize pulsating torque and self starting

Example

- In a electromagnetic system, the rotor has no winding (i.e. we have a reluctance motor) and the inductance of the stator as a function of the rotor position θ is $L_{ss} = L_0 + L_2 \cos 2\theta$. The stator current is $i_s = I_{sm} \sin \omega t$
 - (a) Obtain an expression for the torque acting on the rotor
 - (b) Let $\theta = \omega_m t + \delta$, where ω_m is the angular velocity of the rotor and δ is the rotor position at $t = 0$. Find the condition for the non-zero average torque and obtain the expression for the average torque.

Example 5

In a doubly excited rotating actuator shown in figure below, the stator inductances are given as $L_{11} = (3 + \cos 2\theta)$ mH, $L_{12} = 0.3 \cos \theta$, and the rotor impedance is $L_{22} = 30 + 10 \cos 2\theta$. Find the developed torque in the system for $i_1 = 0.8$ A and $i_2 = 0.01$ A.



Fgrd pg 140

LEVEL MEASUREMENT

By

N Harathi

Assistant Professor

Electronics and Instrumentation Engineering

Unit: **II**

Course: **Industrial Instrumentation - II**

Target Group: **III B.Tech. EIE**

LEVEL MEASUREMENT

- ▶ With the wide variety of approaches to level measurement and as many as 163 suppliers offering one or more types of level-measuring instrument, identifying the right one for your application can be very difficult. In recent years, technologies that capitalized on microprocessor developments have stood out from the pack.
- ▶ For example, the tried-and-true technique of measuring the head of a liquid has gained new life thanks to “smart” differential pressure (DP) transmitters. Today’s local level-measuring instruments can include diagnostics as well as configuration and process data that can be communicated over a network to remote monitoring and control instrumentation. One model even provides local PID control.

LEVEL MEASUREMENT

- ▶ Some of the most commonly used liquid-level measurement methods are:
 - FLOAT type
 - RF capacitance
 - Conductance (conductivity)
 - Hydrostatic head/tank gauging
 - Radar
 - Ultrasonic
- ▶ Before you can decide which one is right for your application, however, you need to understand how each works and the theory behind it.

Float type Level transmitter

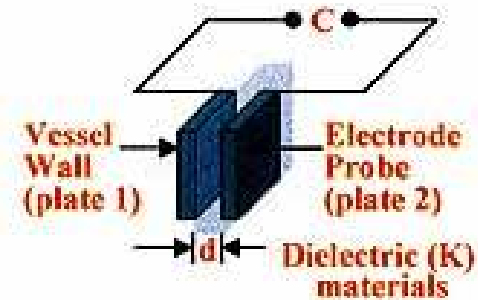


RF Capacitance

- ▶ RF (radio frequency) technology uses the electrical characteristics of a capacitor, in several different configurations, for level measurement.
- ▶ Commonly referred to as RF capacitance or simply RF, the method is suited for detecting the level of liquids, slurries, granulars, or interfaces contained in a vessel.
- ▶ Designs are available for measuring process level at a specific point, at multiple points, or continuously over the entire vessel height. Radio frequencies for all types range from 30 kHz to 1 MHz.

Capacitance Measurement Theory

- ▶ All RF level systems make use of enhancements of the same capacitance-measuring technique, and the same basic theory underlies them all.
- ▶ An electrical capacitance exists between two conductors separated by a distance, d . The first conductor can be the vessel wall (plate 1), and the second can be a measurement probe or electrode (plate 2). The two conductors have an effective area, A , normal to each other.
- ▶ Between the conductors is an insulating medium—the nonconducting material involved in the level measurement.



$$C = E \frac{KA}{d}$$

C= Capacitance in pf
E= Constant
K= Process Dielectric
A= Surface area of plates
d= Distance between plates

Capacitance Measurement Theory

- ▶ The amount of capacitance here is determined not only by the spacing and area of the conductors, but also by the electrical characteristic (relative dielectric constant, K) of the insulating material.
- ▶ The value of K affects the charge storage capacity of the system: The higher the K , the more charge it can build up.
- ▶ Dry air has a K of 1.0. Liquids and solids have considerably higher values, as shown in Table 1.

Capacitance Measurement Theory

- ▶ The capacitance for the basic capacitor arrangement shown in Figure 1 can be computed from the equation:

$$C = E (K A/d)$$

where:

C = capacitance in picofarads (pF)

E = a constant known as the absolute permittivity of free space

K = relative dielectric constant of the insulating material

A = effective area of the conductors

d = distance between the conductors

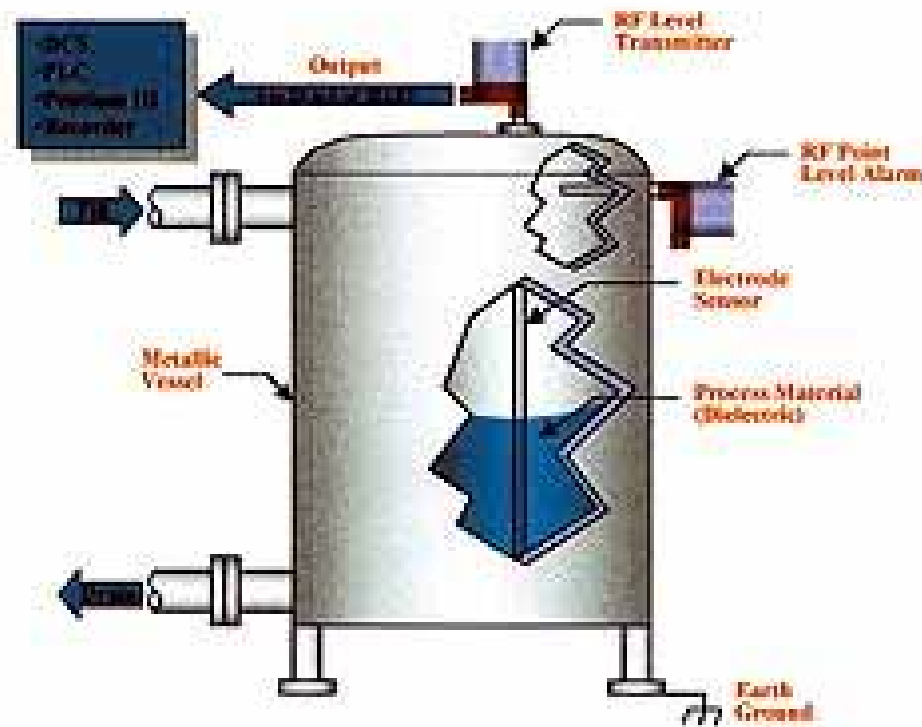
Capacitance Measurement Theory

- ▶ To apply this formula to a level-measuring system, you must assume that the process material is insulating, which, of course, is not always true.
- ▶ A bare, conductive, sensing electrode (probe) is inserted down into a tank to act as one conductor of the capacitor.
- ▶ The metal wall of the tank acts as the other.
- ▶ If the tank is nonmetallic, a conductive ground reference must be inserted into the tank to act as the other capacitor conductor.

Capacitance Measurement Theory

- ▶ With the tank empty, the insulating medium between the two conductors is air. With the tank full, the insulating material is the process liquid or solid.
- ▶ As the level rises in the tank to start covering the probe, some of the insulating effect from air changes into that from the process material, producing a change in capacitance between the sensing probe and ground.
- ▶ This capacitance is measured to provide a direct, linear measurement of tank level.

Capacitance Measurement Theory



- ▶ In the RF capacitance method of liquid level measurement, the electrode sensor connects directly to an RF transmitter outside the tank.

Capacitance Measurement Theory

- ▶ The electrode sensor, or probe, connects directly to an RF level transmitter, which is mounted outside the tank. In one design, with the probe mounted vertically, the system can be used for both continuous level measurement and simultaneous multipoint level control.
- ▶ Alternatively, for point level measurement, one or more probes can be installed horizontally through the side of the tank; Figure shows this type being used as a high-level alarm.

Capacitance Measurement Theory

TABLE 1

Dielectric Constants of Sample Substances

Substance	Value
Isopropyl alcohol	18.3
Kerosene	1.8
Kynar	8.0
Mineral oil	2.1
Pure water	80
Sand	4.0
Sugar	3.0
Teflon	2.0

Capacitance Measurement Theory



This view of a typical RF capacitance probe shows the electronic chassis enlarged to twice the size of its housing.

RF Impedance or RF Admittance.

- ▶ When another electrical characteristic, impedance, enters the picture, the result is further refinements in RF level measurement.
- ▶ Offering improved reliability and a wider range of uses, these variations of the basic RF system are called RF admittance or RF impedance. In RF or AC circuits, impedance, Z , is defined as the total opposition to current flow:
 - ▶ $Z = R + 1 / j 2 \pi f C$
where:
R = resistance in ohms
f = measurement frequency (radio frequency for RF measurement)
C = capacitance in picofarads

RF Impedance or RF Admittance.

- ▶ An RF impedance level-sensing instrument measures this total impedance rather than just the capacitance. Some level-measuring systems are referred to as RF admittance types. Admittance, A , is defined as a measure of how readily RF or AC current will flow in a circuit and is therefore the reciprocal of impedance ($A = 1/Z$). Thus, there is no basic difference between the RF impedance and RF admittance as a level-measurement technology.
- ▶ In some cases, the process material tends to build up a coating on the level-sensing probe. In such cases, which are not uncommon in level applications, a significant measurement error can occur because the instrument measures extra capacitance and resistance from the coating buildup. As a result, the sensor reports a higher, and incorrect, level instead of the actual tank level.

RF Impedance or RF Admittance.

- ▶ Note that the equation for impedance includes resistance, R . The RF impedance method can be provided with specific circuitry capable of measuring the resistance and capacitance components from the coating and the capacitive component due to the actual process material level.
- ▶ The circuitry is designed to solve a mathematical relationship electronically, thereby producing a 4–20 mA current output that is proportional only to the actual level of the process material. It is virtually unaffected by any buildup of coating on the sensing probe, enabling an RF system to continue functioning reliably and accurately.

Conductance

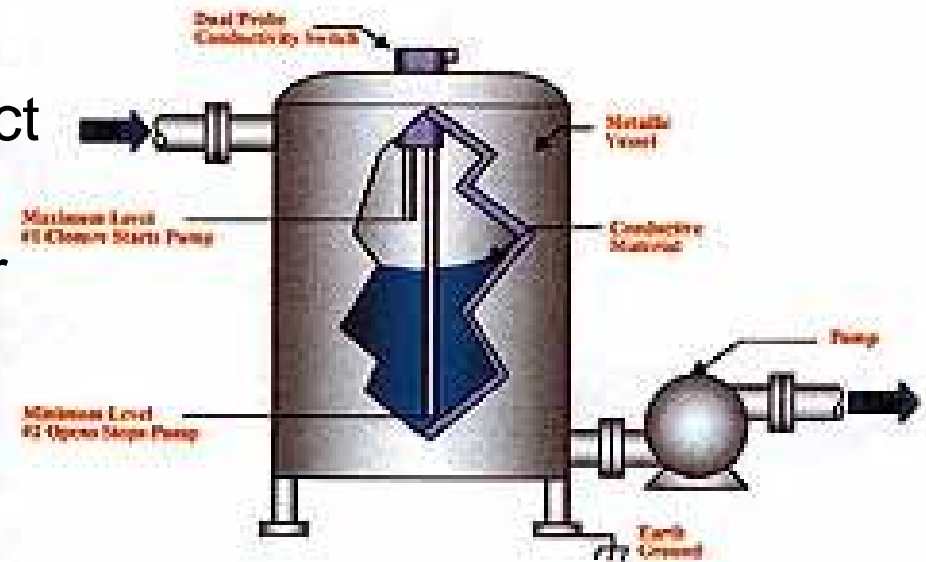
- ▶ The conductance method of liquid level measurement is based on the electrical conductance of the measured material, which is usually a liquid that can conduct a current with a low-voltage source (normally <20 V). Hence the method is also referred to as a conductivity system. Conductance is a relatively low-cost, simple method to detect and control level in a vessel.
- ▶ One common way to set up an electrical circuit is to use a dual-tip probe that eliminates the need for grounding a metal tank. Such probes are generally used for point level detection, and the detected point can be the interface between a conductive and nonconductive liquid.

Conductance

- ▶ The conductance method of liquid level measurement is based on the electrical conductance of the measured material, which is usually a liquid that can conduct a current with a low-voltage source (normally <20 V). Hence the method is also referred to as a conductivity system. Conductance is a relatively low-cost, simple method to detect and control level in a vessel.
- ▶ One common way to set up an electrical circuit is to use a dual-tip probe that eliminates the need for grounding a metal tank. Such probes are generally used for point level detection, and the detected point can be the interface between a conductive and nonconductive liquid.
- ▶ Figure 3 shows an arrangement with two dual-tip probes that detect maximum and minimum levels. When the level reaches the upper probe, a switch closes to start the discharge pump; when the level reaches the lower probe, the switch opens to stop the pump.

Conductance

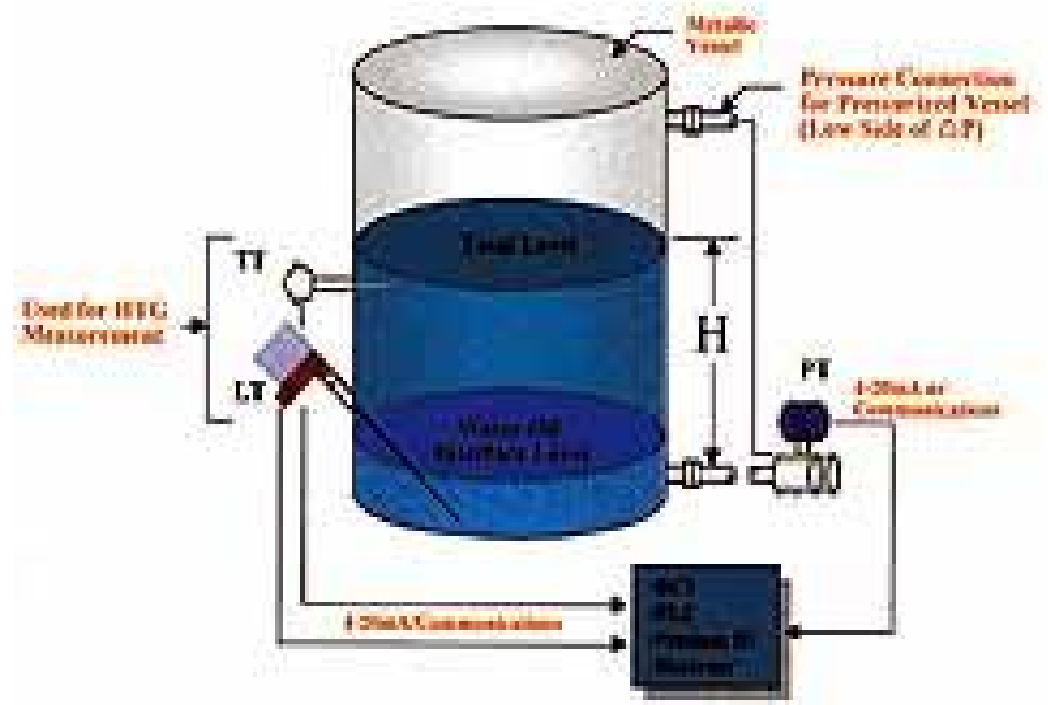
Figure shows an arrangement with two dual-tip probes that detect maximum and minimum levels. When the level reaches the upper probe, a switch closes to start the discharge pump; when the level reaches the lower probe, the switch opens to stop the pump



- ▶ In the conductive type of level measurement, two dual-tip probes detect the maximum and minimum levels in a tank.

Hydrostatic Head

- One of the oldest and most common methods of measuring liquid level is to measure the pressure exerted by a column (or head) of liquid in the vessel. The basic relationships are:
 - ▶ $H = mP/d$
 - ▶ where, in consistent units:
 - ▶ P = pressure
 - ▶ m = a constant
 - ▶ H = head
 - ▶ d = density



Hydrostatic Head

- ▶ The density of a liquid varies with temperature. For the highest precision in level measurement, the density must therefore be compensated for or expressed with relation to the actual temperature of the measured liquid.
- ▶ For decades, DP-type instruments—long before the DP cell—were used to measure liquid.
- ▶ With open vessels a pipe at or near the bottom of the vessel connects only to the high-pressure side of the meter body and the low-pressure side is open to the atmosphere.
- ▶ If the vessel is pressurized or under vacuum, the low side of the meter has a pipe connection near the top of the vessel, so that the instrument responds only to changes in the head of liquid.

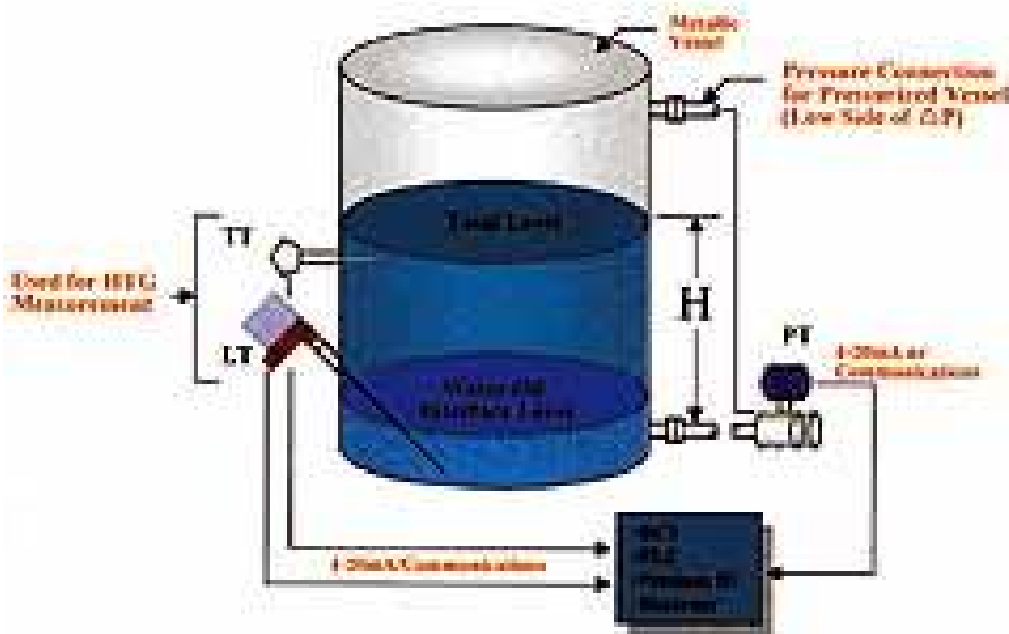


Figure shows a simplified system that incorporates only one pressure transmitter (PT) with a temperature transmitter (TT) and makes novel use of a level transmitter (LT) to detect accumulation of water at the bottom of a tank.

Mass (weight) of the tank's contents can be calculated from the hydrostatic head (measured by PT) multiplied by the tank area (obtained from a lookup table). The liquid's temperature-density relationship can be used to calculate the volume and level, provided the tank is not under pressure. Data fed into a computer system make it possible for all calculations to be automatic, with results continuously available for monitoring and accounting purposes.

Radar or Microwave

- ▶ Radar methods of level measurement are sometimes referred to as microwave types. Both use electromagnetic waves, typically in the microwave X-band (10 GHz) range. This technology is being adapted and refined for level measurement, so you should check out the latest offerings. Most applications have been designed for continuous level measurement.
- ▶ Basically, all types operate on the principle of beaming microwaves downward from a sensor located on top of the vessel. The sensor receives back a portion of the energy that is reflected off the surface of the measured medium. Travel time for the signal (called the time of flight) is used to determine level. For continuous level measurement, there are two main types of noninvasive systems, as well as one invasive type that uses a cable or rod as a wave guide and extends down into the tank's contents to near its bottom.

Radar or Microwave

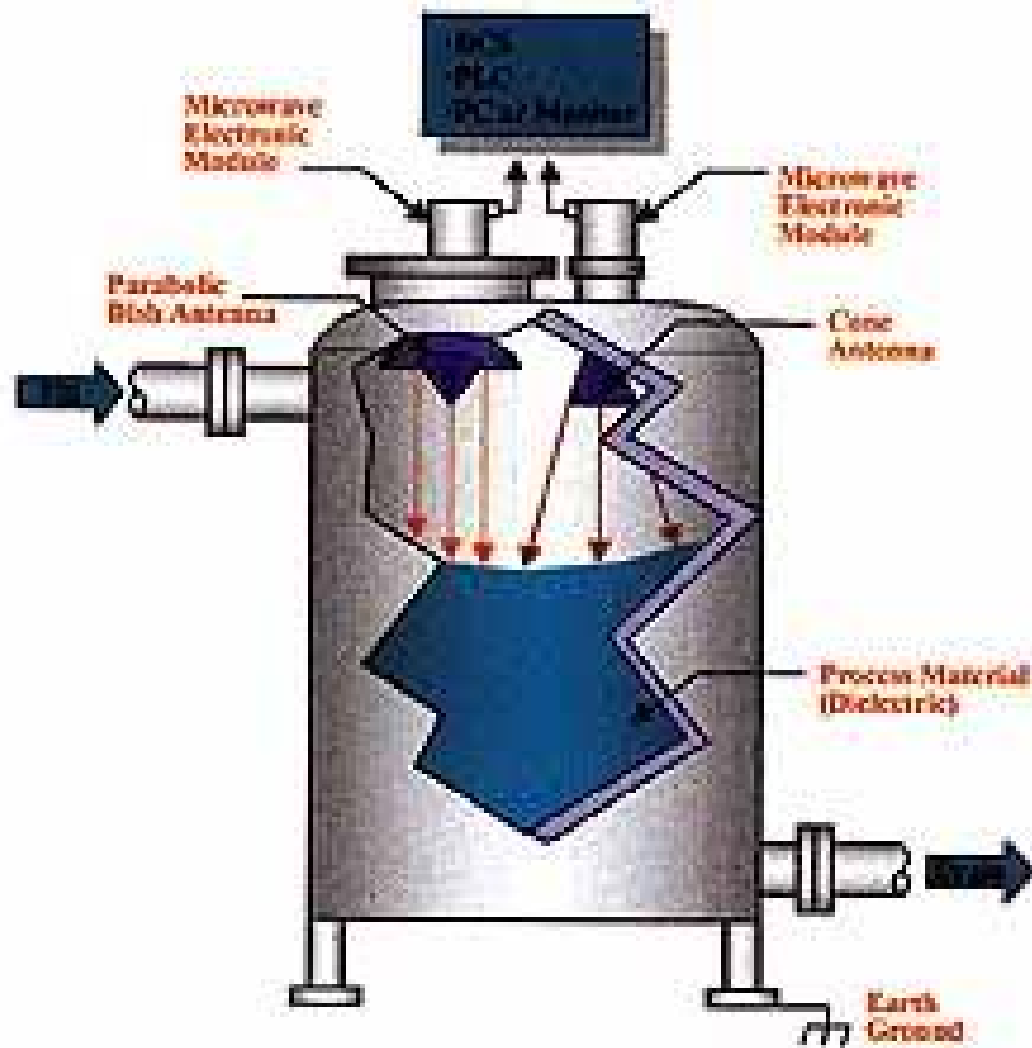
- ▶ One type of noninvasive system uses a technology called frequency-modulated continuous wave (FMCW).
- ▶ From an electronic module on top of the tank, a sensor oscillator sends down a linear frequency sweep, at a fixed bandwidth and sweep time.
- ▶ The reflected radar signal is delayed in proportion to the distance to the level surface.
- ▶ Its frequency is different from that of the transmitted signal, and the two signals blend into a new frequency proportional to distance. That new frequency is converted into a very accurate measure of liquid level.

Radar or Microwave

- ▶ The sensor outputs a frequency-modulated (FM) signal that varies from 0 to ~200 Hz as the distance ranges from 0 to 200 ft (60 m).
- ▶ An advantage of this technique is that the level-measurement signals are FM rather than AM, affording the same advantages that radio waves offer.
- ▶ Most tank noise is in the AM range and does not affect the FM signals.

Radar or Microwave

- ▶ The second noninvasive technology, pulsed radar or pulsed time-of-flight, operates on a principle very similar to that of the ultrasonic pulse method. The radar pulse is aimed at the liquid's surface and the transit time of the pulse's return is used to calculate level. Because pulse radar is lower power than FMCW, its performance can be affected by obstructions in the tank as well as foam and low-dielectric materials ($K < 2$).
- ▶ Antennas for the noninvasive methods come in two designs: parabolic dish and cone.
- ▶ The parabolic dish antenna tends to direct the signals over a wider area while the cone tends to confine the signals in a narrower downward path. The choice of one or the other, and its diameter, depends on application factors such as tank obstructions that may serve as reflectors, the presence of foam, and turbulence of the measured fluid.



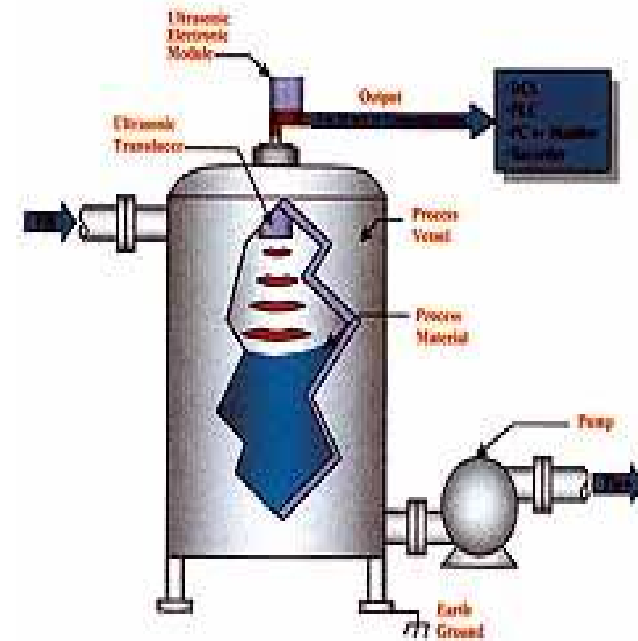
Radar (microwave) level measurement can use either of two types of antenna construction at the top of vessel.

Radar or Microwave

- ▶ Guided-wave radar (GWR) is an invasive method that uses a rod or cable to guide the micro wave as it passes down from the sensor into the material being measured and all the way to the bottom of the vessel. The basis for GWR is time-domain reflectometry (TDR), which has been used for years to locate breaks in long lengths of cable that are underground or in building walls. A TDR generator develops more than 200,000 pulses of electromagnetic energy that travel down the waveguide and back. The dielectric of the measured fluid causes a change in impedance that in turn develops a wave reflection. Transit time of pulses down and back is used as a measure of level.

Ultrasonic and Sonic

- ▶ Both ultrasonic and sonic level instruments operate on the basic principle of using sound waves to determine fluid level. The frequency range for ultrasonic methods is ~20–200 kHz, and sonic types use a frequency of 10 kHz.
- ▶ A top-of-tank mounted transducer directs waves downward in bursts onto the surface of the material whose level is to be measured. Echoes of these waves return to the transducer, which performs calculations to convert the distance of wave travel into a measure of level in the tank.



Radar or Microwave

- ▶ A piezoelectric crystal inside the transducer converts electrical pulses into sound energy that travels in the form of a wave at the established frequency and at a constant speed in a given medium. The medium is normally air over the material's surface but it could be a blanket of nitrogen or some other vapor.
- ▶ The sound waves are emitted in bursts and received back at the transducer as echoes. The instrument measures the time for the bursts to travel down to the reflecting surface and return. This time will be proportional to the distance from the transducer to the surface and can be used to determine the level of fluid in the tank. For practical applications of this method, you must consider a number of factors. A few key points are:

Radar or Microwave

- ▶ The speed of sound through the medium (usually air) varies with the medium's temperature. The transducer may contain a temperature sensor to compensate for changes in operating temperature that would alter the speed of sound and hence the distance calculation that determines an accurate level measurement.
- ▶ The presence of heavy foam on the surface of the material can act as a sound absorbent. In some cases, the absorption may be sufficient to preclude use of the ultrasonic technique.
- ▶ Extreme turbulence of the liquid can cause fluctuating readings. Use of a damping adjustment in the instrument or a response delay may help overcome this problem.



Combinational Logic

BY

Dr.G. Sunitha

Topic: Combinational Logic

Subject: Digital Logic and Design(DLD)

II-B.Tech(CSE)

Sree Vidyanikethan Engineering College

Combinational Logic

- Logic circuits for digital systems may be combinational or sequential.
- A combinational circuit consists of input variables, logic gates, and output variables.

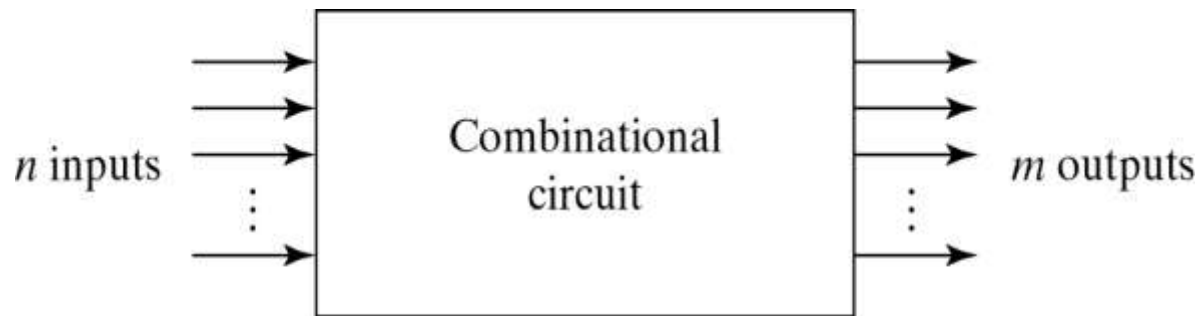


Fig. 4-1 Block Diagram of Combinational Circuit



Analysis procedure

- To obtain the output Boolean functions from a logic diagram, proceed as follows:
 1. Label all gate outputs that are a function of input variables with arbitrary symbols. Determine the Boolean functions for each gate output.
 2. Label the gates that are a function of input variables and previously labeled gates with other arbitrary symbols. Find the Boolean functions for these gates.



Analysis procedure

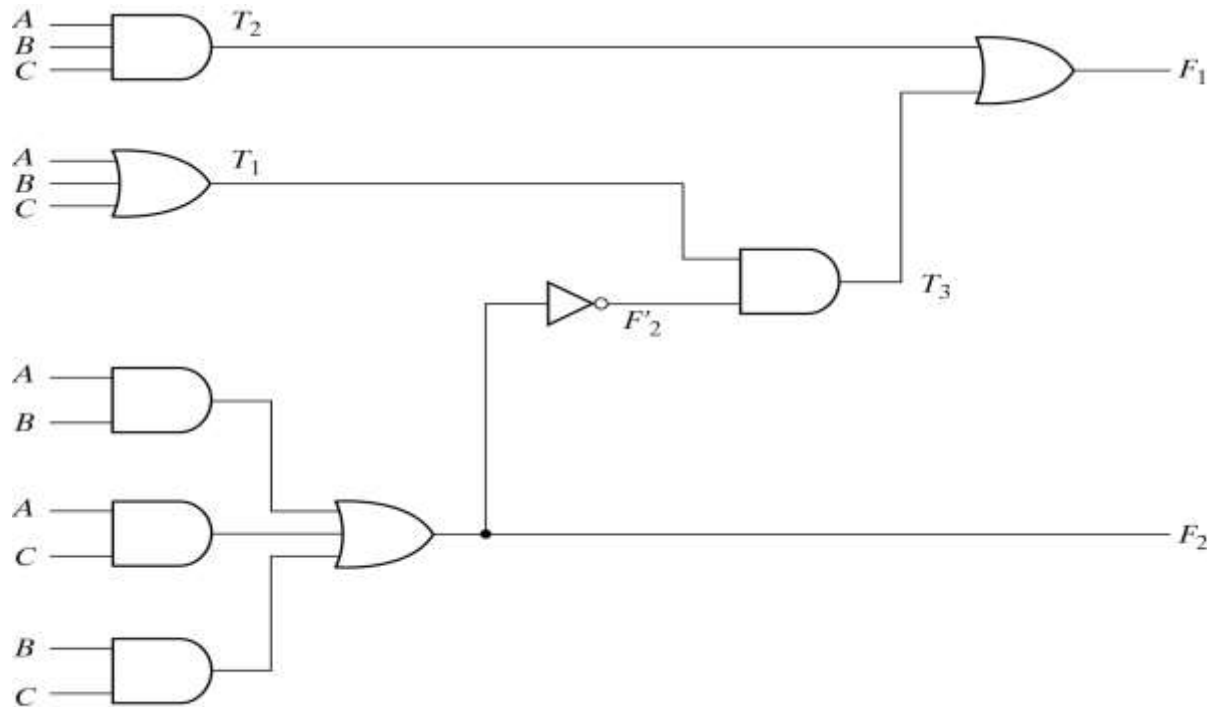
3. Repeat the process outlined in step 2 until the outputs of the circuit are obtained.
4. By repeated substitution of previously defined functions, obtain the output Boolean functions in terms of input variables.

Example

$$F_2 = AB + AC + BC; \quad T_1 = A + B + C; \quad T_2 = ABC; \quad T_3 = F_2'T_1;$$

$$F_1 = T_3 + T_2$$

$$F_1 = T_3 + T_2 = F_2'T_1 + ABC = A'BC' + A'B'C + AB'C' + ABC$$



Derive truth table from logic diagram

- We can derive the truth table in Table 3-1 by using the circuit of Fig.3-2.

Table 4-1
Truth Table for the Logic Diagram of Fig. 4-2

<i>A</i>	<i>B</i>	<i>C</i>	<i>F</i> ₂	<i>F</i> ₂	<i>T</i> ₁	<i>T</i> ₂	<i>T</i> ₃	<i>F</i> ₁
0	0	0	0	1	0	0	0	0
0	0	1	0	1	1	0	1	1
0	1	0	0	1	1	0	1	1
0	1	1	1	0	1	0	0	0
1	0	0	0	1	1	0	1	1
1	0	1	1	0	1	0	0	0
1	1	0	1	0	1	0	0	0
1	1	1	1	0	1	1	0	1

Design procedure

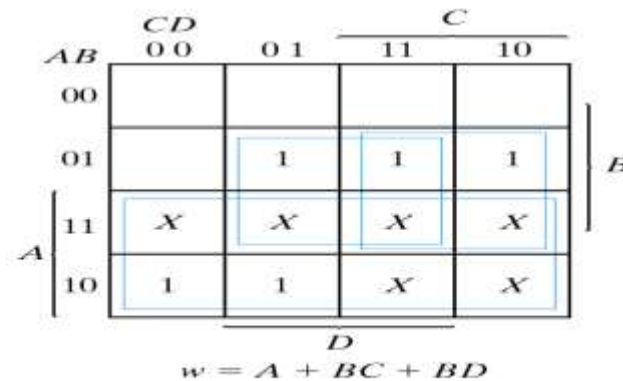
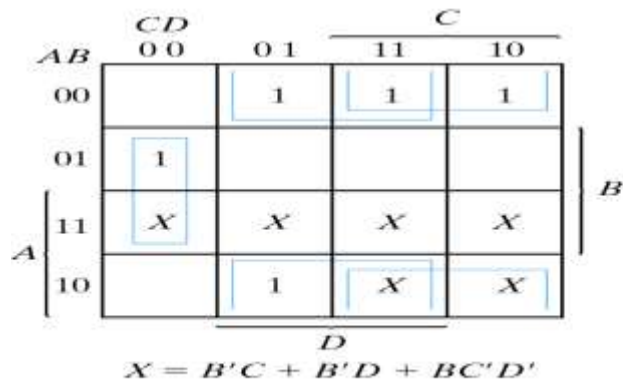
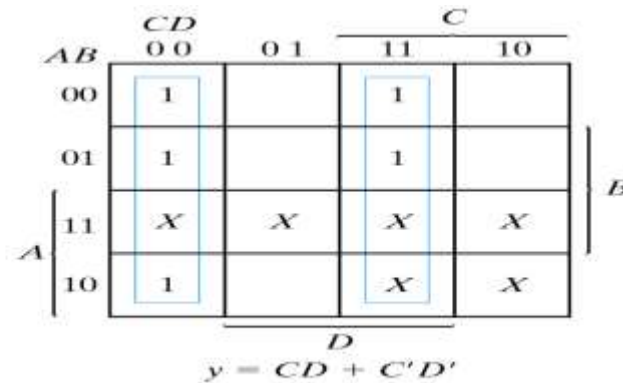
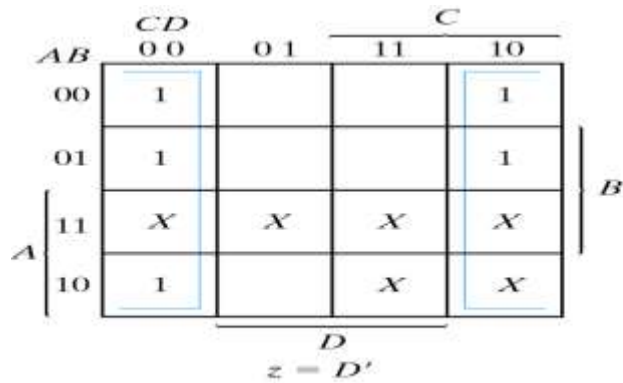
1. Table 3-2 is a Code-Conversion example, first, we can list the relation of the BCD and Excess-3 codes in the truth table.

Table 4-2
Truth Table for Code-Conversion Example

Input BCD				Output Excess-3 Code			
A	B	C	D	w	x	y	z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

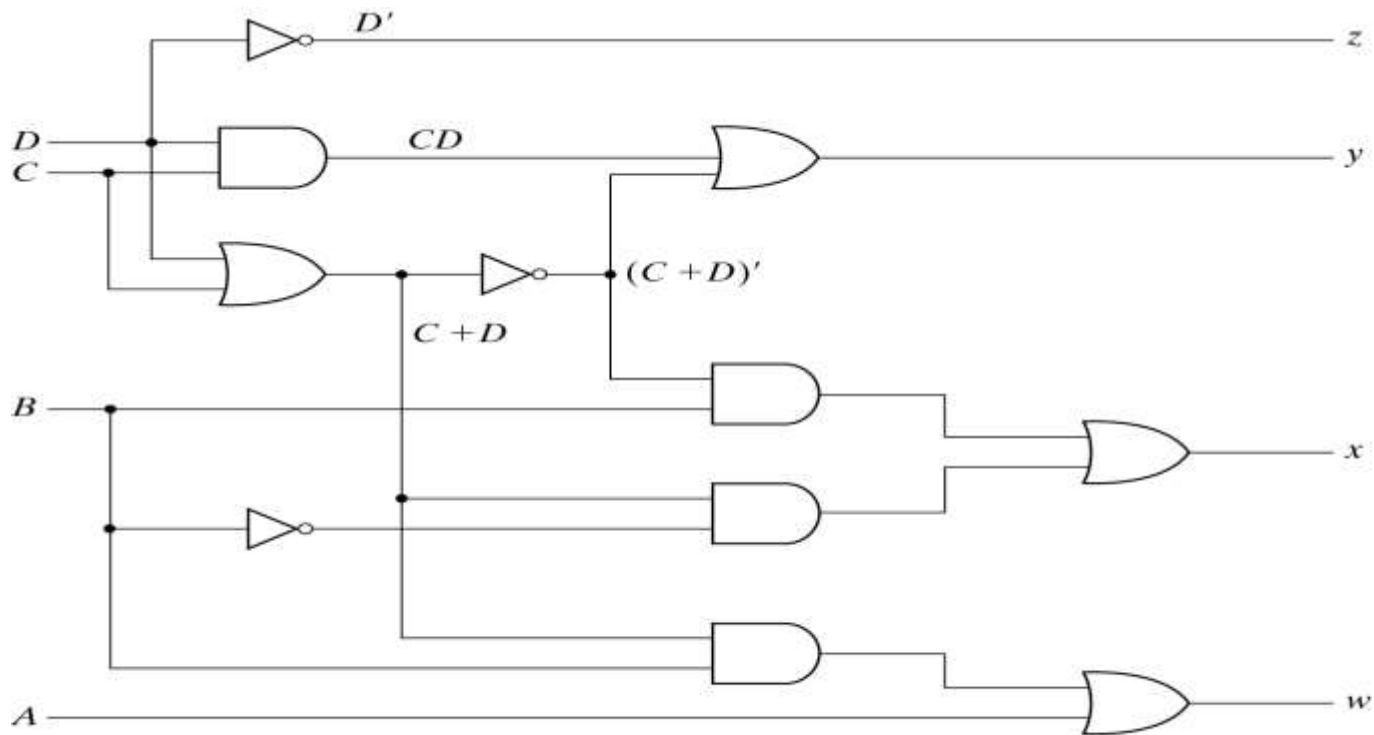
Karnaugh map

2. For each symbol of the Excess-3 code, we use 1's to draw the map for simplifying Boolean function.



Circuit implementation

$$z = D'; \quad y = CD + C'D' = CD + (C + D)'$$
$$x = B'C + B'D + BC'D' = B'(C + D) + B(C + D)'$$
$$w = A + BC + BD = A + B(C + D)$$



Binary Adder-Subtractor

- A combinational circuit that performs the addition of two bits is called a **half adder**.
- The truth table for the half adder is listed below:

Table 4-3
Half Adder

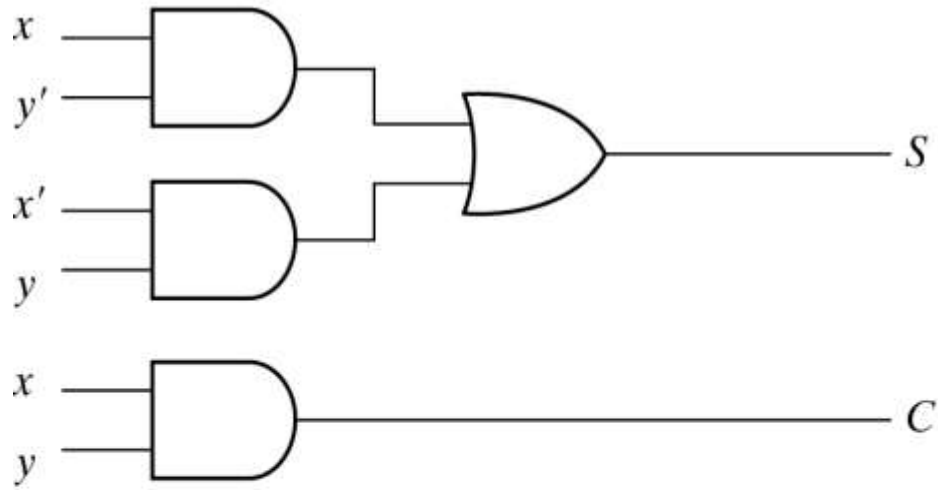
<i>x</i>	<i>y</i>	<i>C</i>	<i>S</i>
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

S: Sum
C: Carry

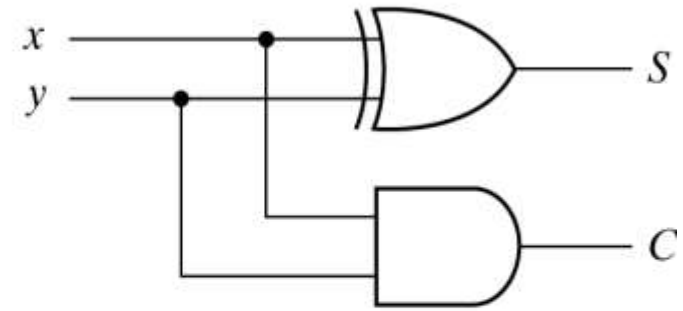
$$S = x'y + xy'$$

$$C = xy$$

Implementation of Half-Adder



$$(a) S = xy' + x'y$$
$$C = xy$$



$$(b) S = x \oplus y$$
$$C = xy$$

Fig. 4-5 Implementation of Half-Adder

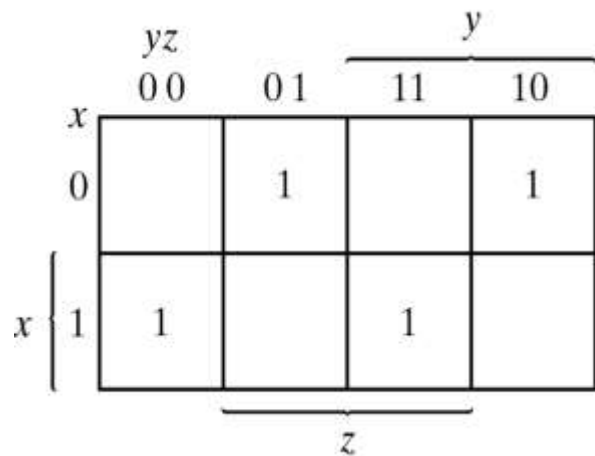
Full-Adder

- One that performs the addition of three bits (two significant bits and a previous carry) is a **full adder**.

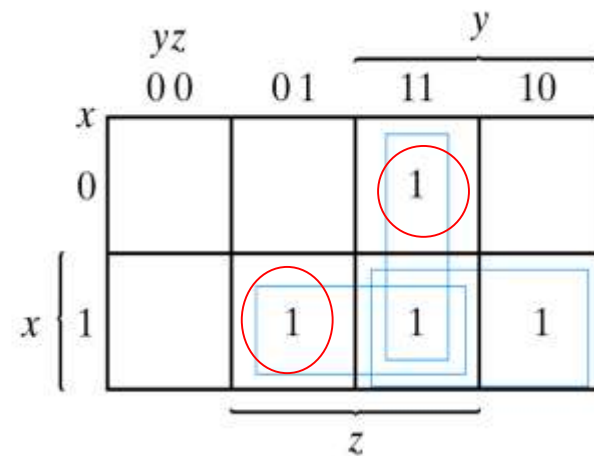
Table 4-4
Full Adder

<i>x</i>	<i>y</i>	<i>z</i>	<i>C</i>	<i>S</i>
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Simplified Expressions



$$S = x'y'z + x'yz' + xy'z' + xyz$$



$$C = xy + xz + yz$$

$$= xy + xy'z + x'yz$$

Fig. 4-6 Maps for Full Adder

$$S = x'y'z + x'yz' + xy'z' + xyz$$

$$C = xy + xz + yz$$

Full adder implemented in SOP

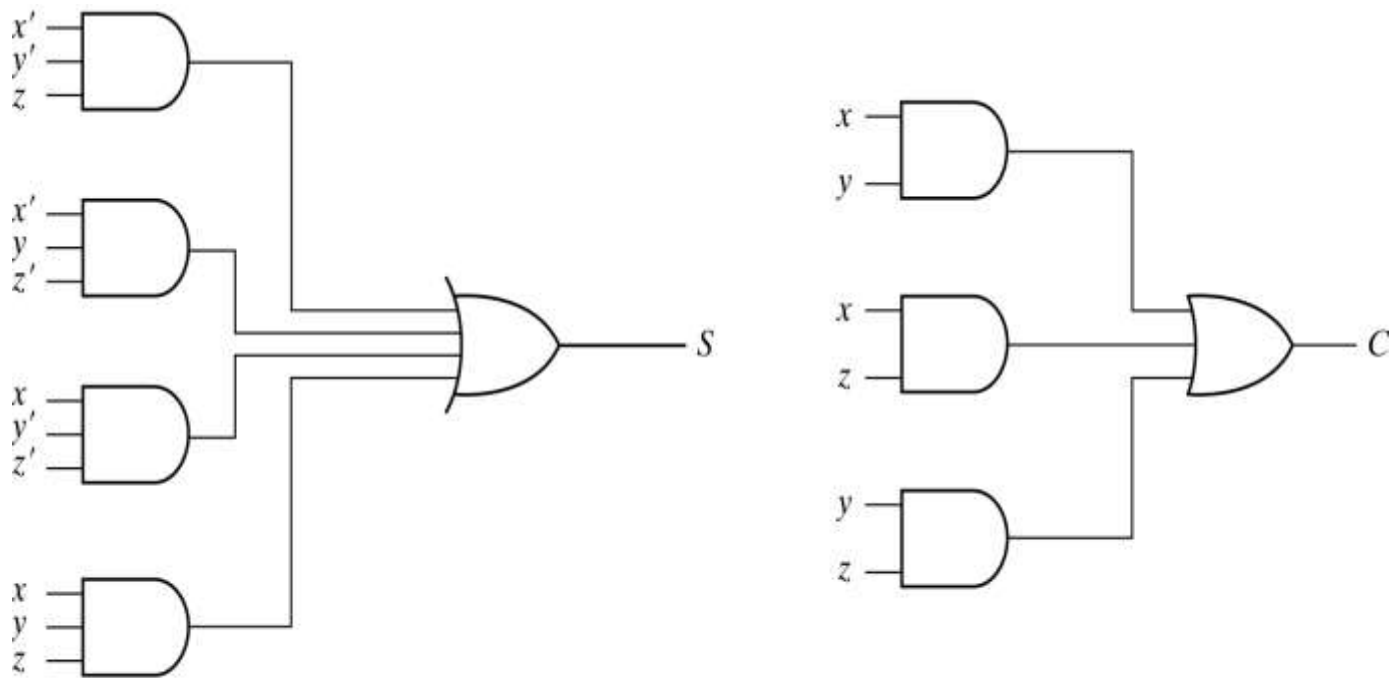


Fig. 4-7 Implementation of Full Adder in Sum of Products

Another implementation

- Full-adder can also be implemented with **two half adders and one OR gate** (Carry Look-Ahead adder).

$$S = z \oplus (x \oplus y)$$

$$= z'(xy' + x'y) + z(xy' + x'y)'$$

$$= xy'z' + x'yz' + xyz + x'y'z$$

$$C = z(xy' + x'y) + xy = xy'z + x'yz + xy$$

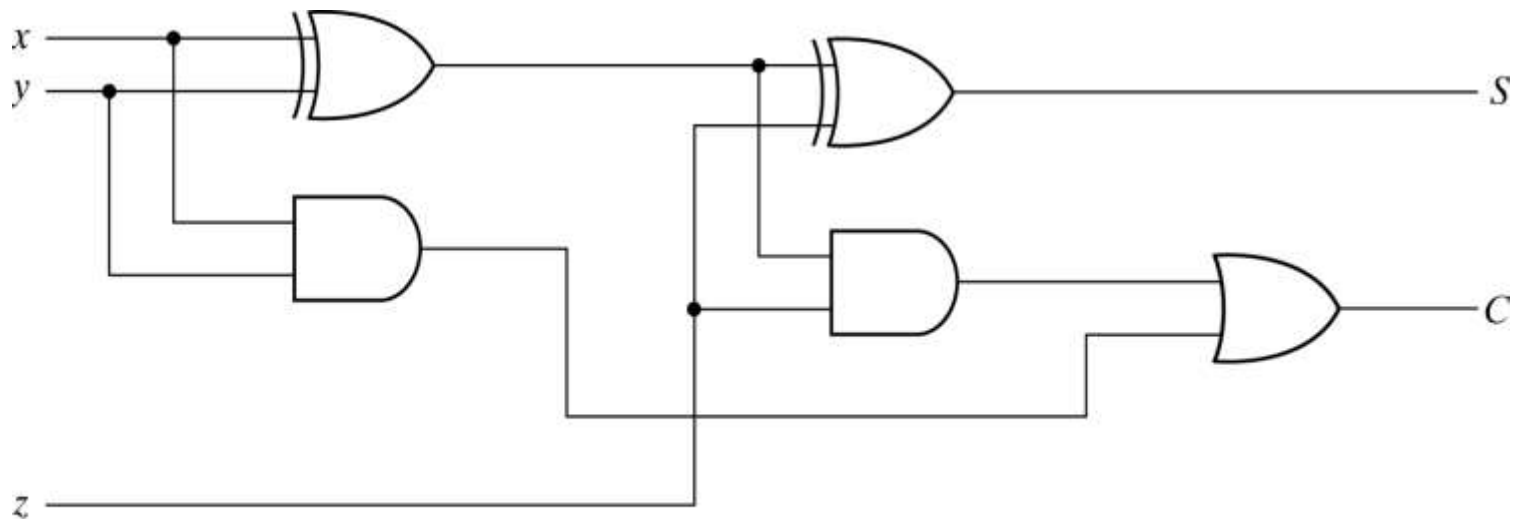
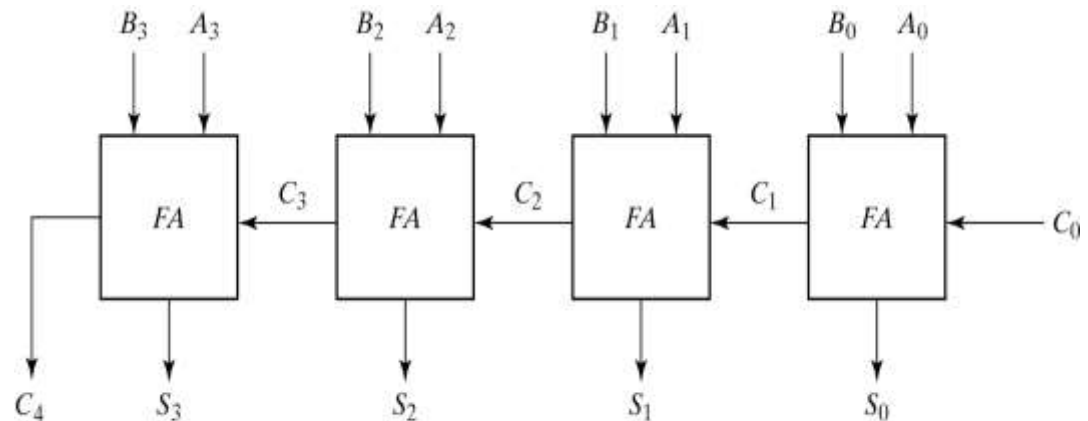


Fig. 1.8 Implementation of Full Adder with Two Half Adders and an OR Gate

Binary adder

- This is also called **Ripple Carry Adder**, because of the construction with full adders are connected in cascade.

<i>Subscript i:</i>	3	2	1	0	
Input carry	0	1	1	0	C_i
Augend	1	0	1	1	A_i
Addend	0	0	1	1	B_i
Sum	1	1	1	0	S_i
Output carry	0	0	1	1	C_{i+1}



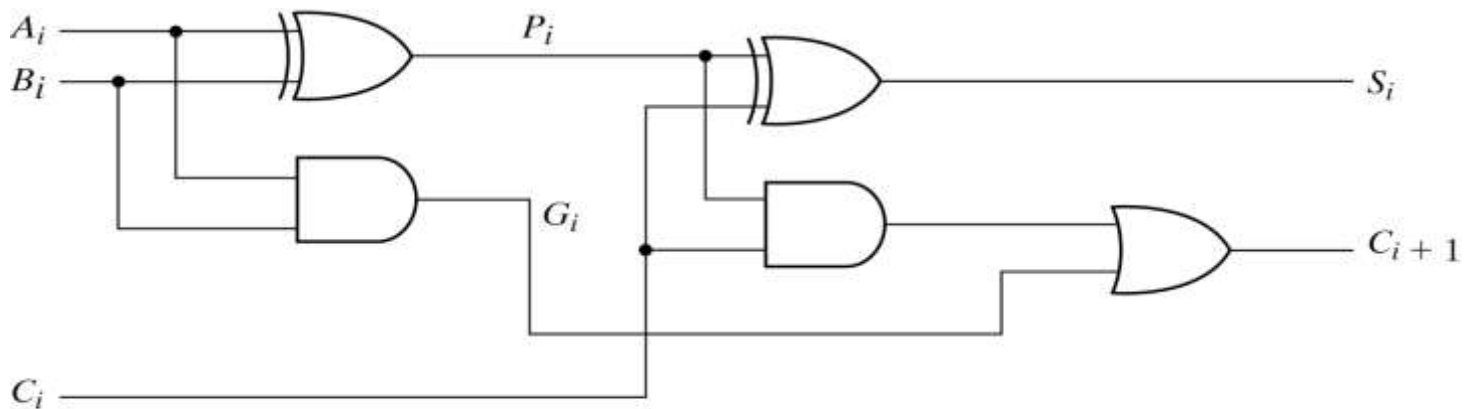


Carry Propagation

- Fig.3-9 causes a **unstable** factor on **carry bit**, and produces a **longest propagation delay**.
- The signal from C_i to the output carry C_{i+1} , **propagates through an AND and OR gates**, so, for an n-bit RCA, there are **2n** gate levels for the carry to propagate from input to output.

Carry Propagation

- Because the propagation delay will affect the output signals on different time, so the signals are **given enough time to get the precise and stable outputs**.
- The most widely used technique employs the principle of **carry look-ahead** to **improve the speed of the algorithm**.





Boolean functions

$$P_i = A_i \oplus B_i \quad \text{steady state value}$$

$$G_i = A_i B_i \quad \text{steady state value}$$

Output sum and carry

$$S_i = P_i \oplus C_i$$

$$C_{i+1} = G_i + P_i C_i$$

G_i : carry generate P_i : carry propagate

C_0 = input carry

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1 = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

- C_3 does not have to wait for C_2 and C_1 to propagate.

Logic diagram of carry look-ahead generator

- C_3 is propagated at the same time as C_2 and C_1 .

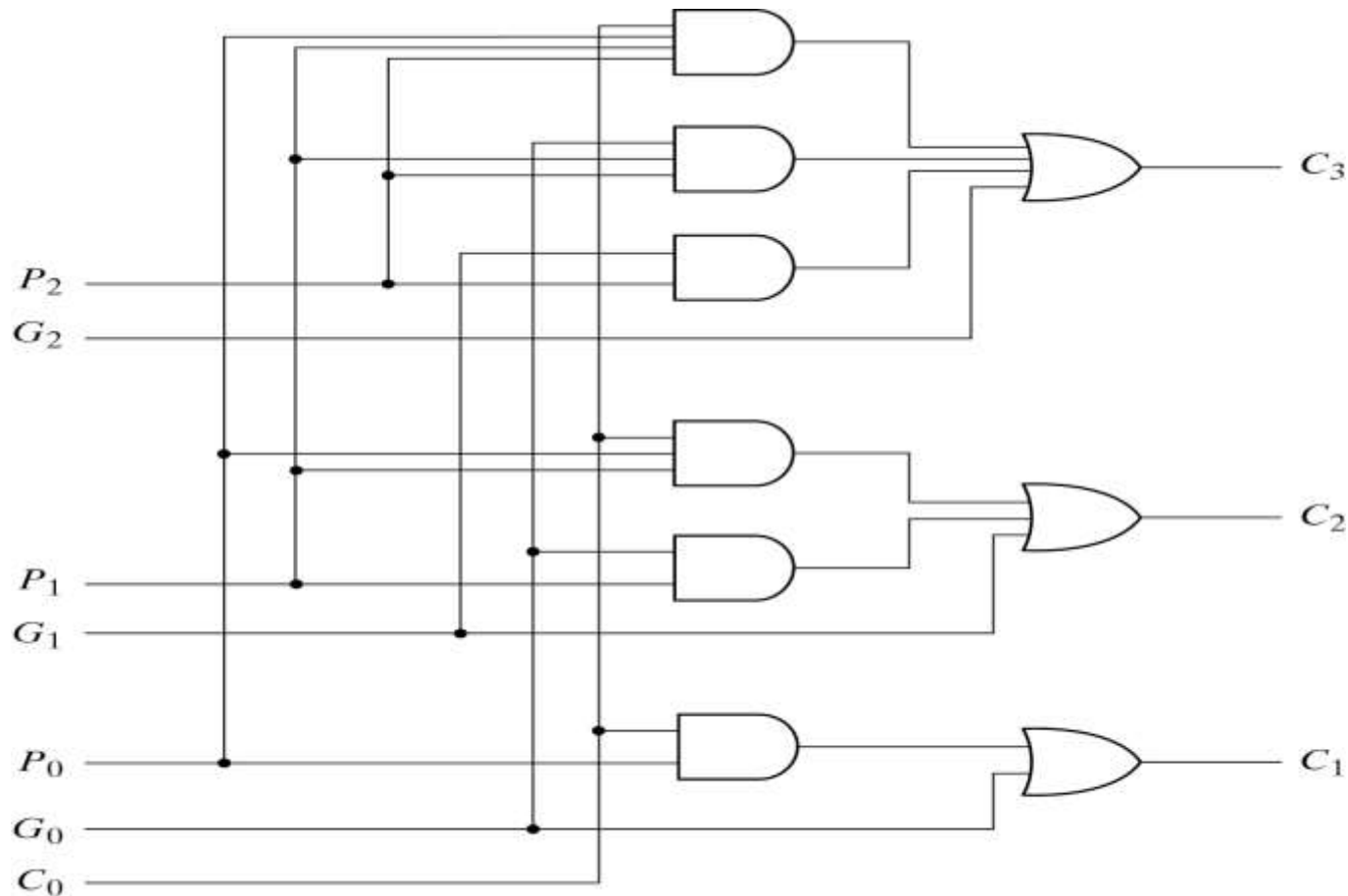


Fig. 4-11 Logic Diagram of Carry Lookahead Generator

3-bit adder with carry lookahead

- Delay time of n-bit CLAA = XOR + (AND + OR) + XOR

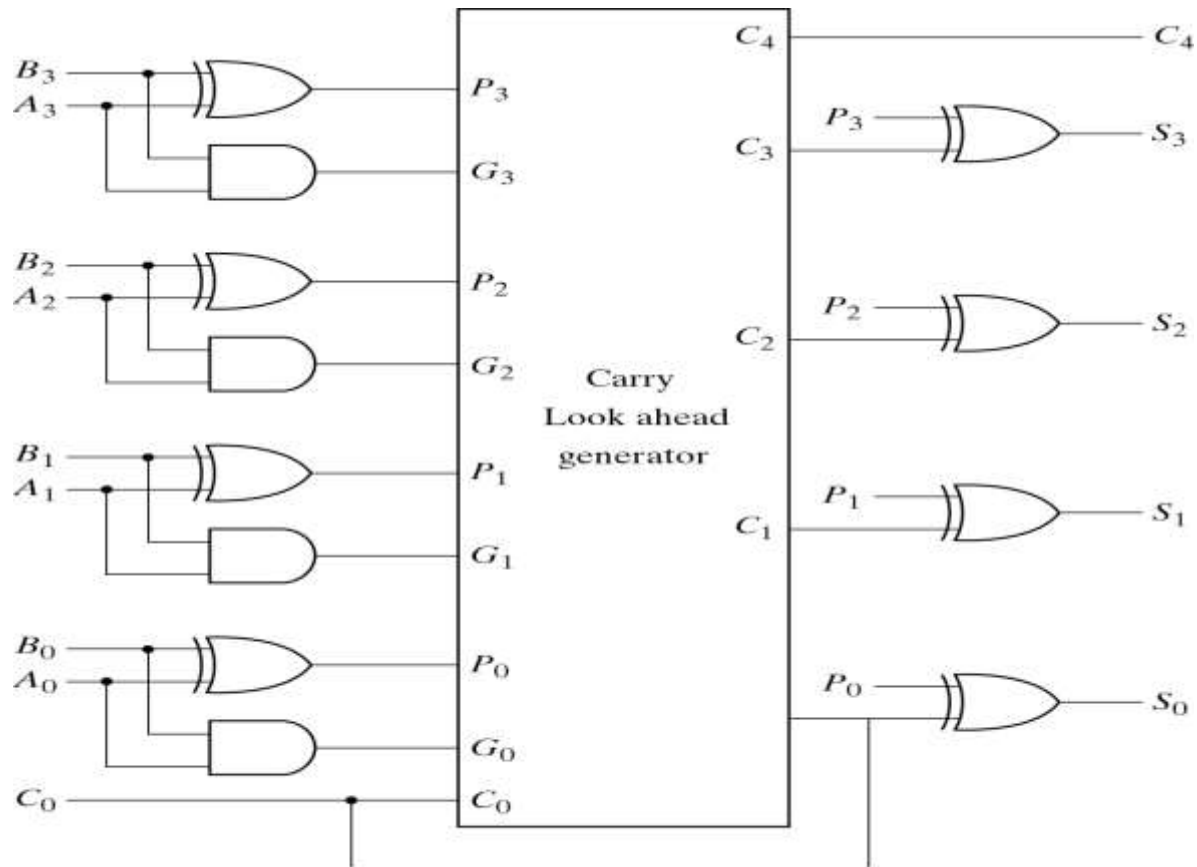


Fig. 4-12 4-Bit Adder with Carry Lookahead
Combinational Logic

Binary subtractor

$M = 1 \rightarrow$ subtractor ; $M = 0 \rightarrow$ adder

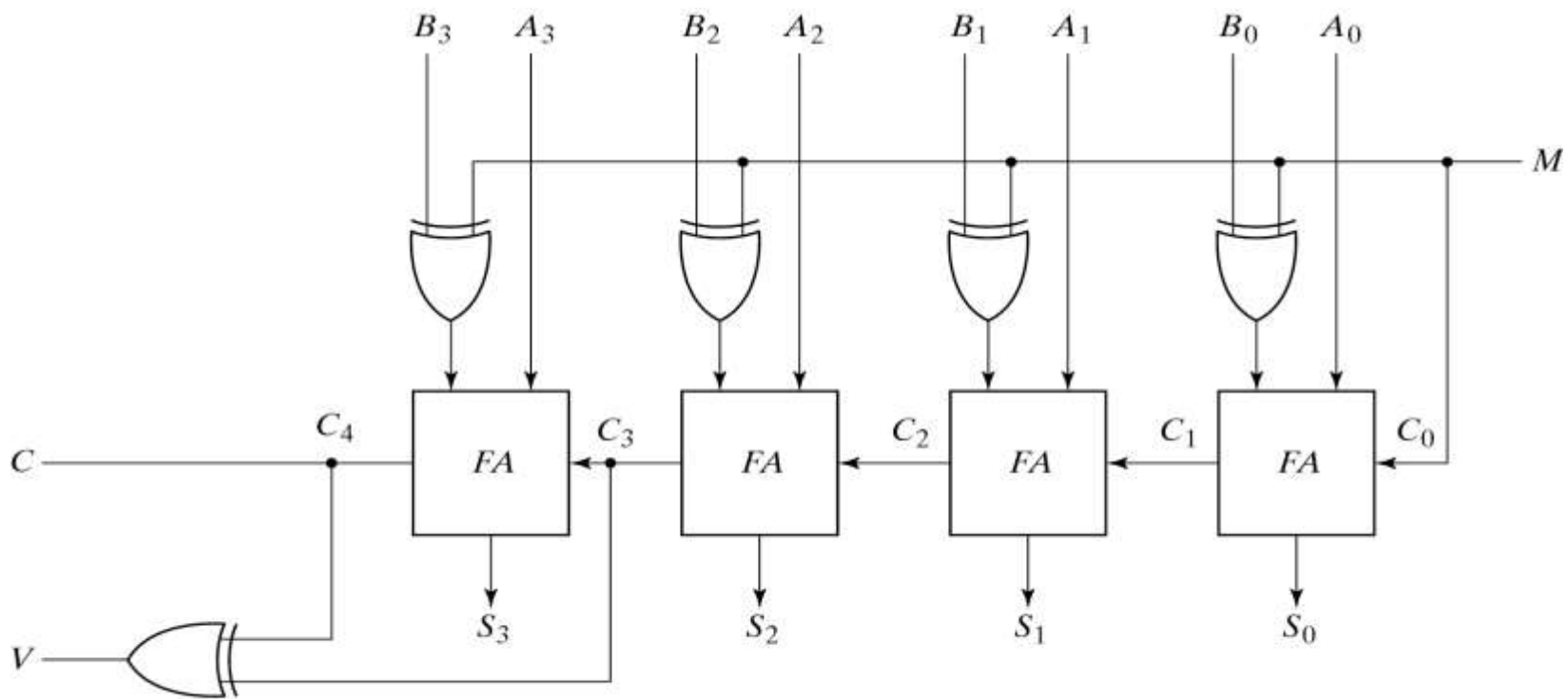


Fig. 4-13 4-Bit Adder Subtractor



Overflow

- It is **worth** noting Fig.3-13 that binary numbers in the **signed-complement system are added and subtracted** by the same basic addition and subtraction rules **as unsigned numbers**.
- Overflow is a problem in digital computers because the number of bits that hold the number is finite and a result that contains $n+1$ bits cannot be accommodated.



Overflow on signed and unsigned

- When two **unsigned** numbers are added, an overflow is detected from the **end carry out of the MSB position**.
- When two **signed** numbers are added, the sign bit is treated as part of the number and the end carry does not indicate an overflow.
- An **overflow can't occur** after an addition if one number is **positive** and the other is **negative**.
- An overflow may occur if the two numbers added are both positive or both negative.

Decimal adder

BCD adder can't exceed 9 on each input digit. K is the carry.

Table 4-5
Derivation of BCD Adder

Binary Sum					BCD Sum					Decimal
K	Z ₈	Z ₄	Z ₂	Z ₁	C	S ₈	S ₄	S ₂	S ₁	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19



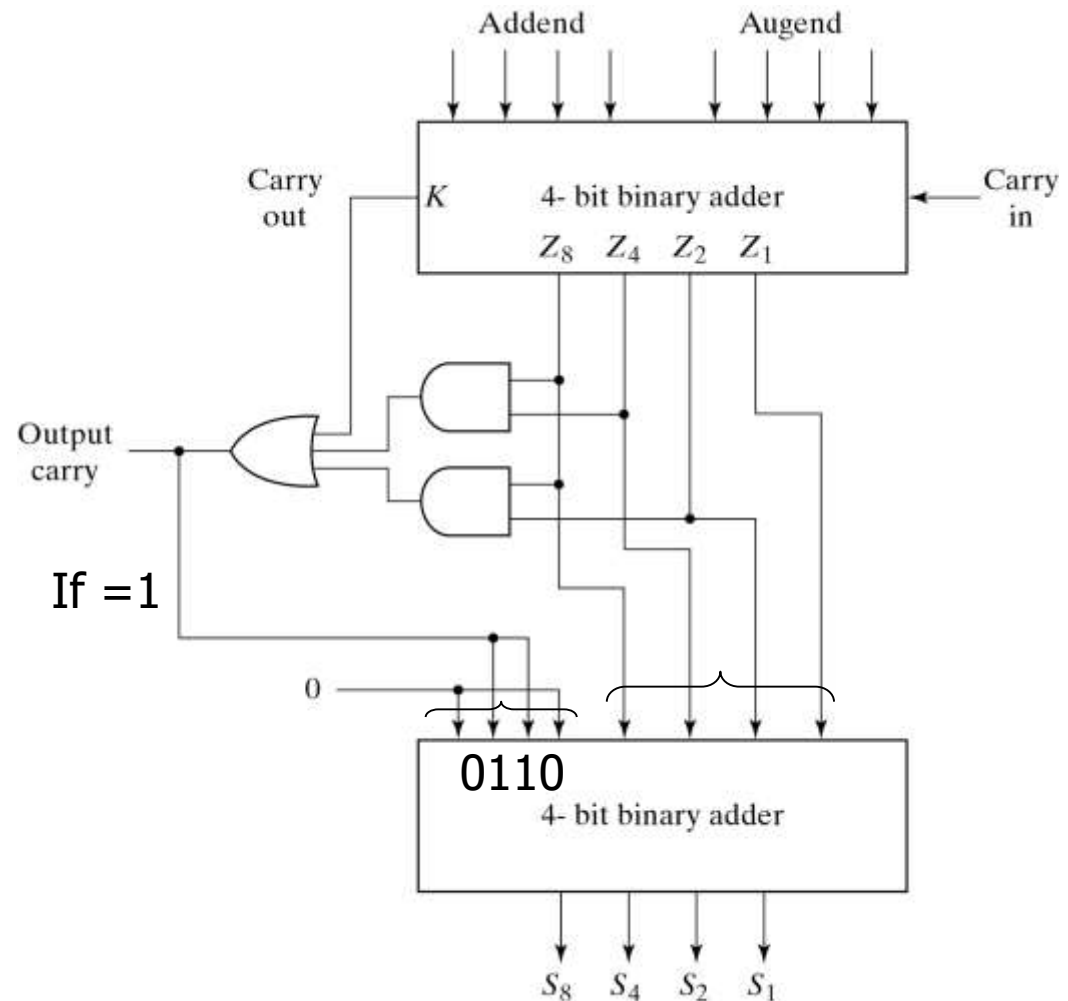
Rules of BCD adder

- When the binary sum is **greater than 1001**, we obtain a **non-valid BCD** representation.
- The **addition of binary 6(0110)** to the binary sum **converts it to the correct BCD** representation and also produces an output carry as required.
- To distinguish them from binary 1000 and 1001, which also have a 1 in position Z_8 , we specify further that either Z_3 or Z_2 must have a 1.

$$C = K + Z_8Z_3 + Z_8Z_2$$

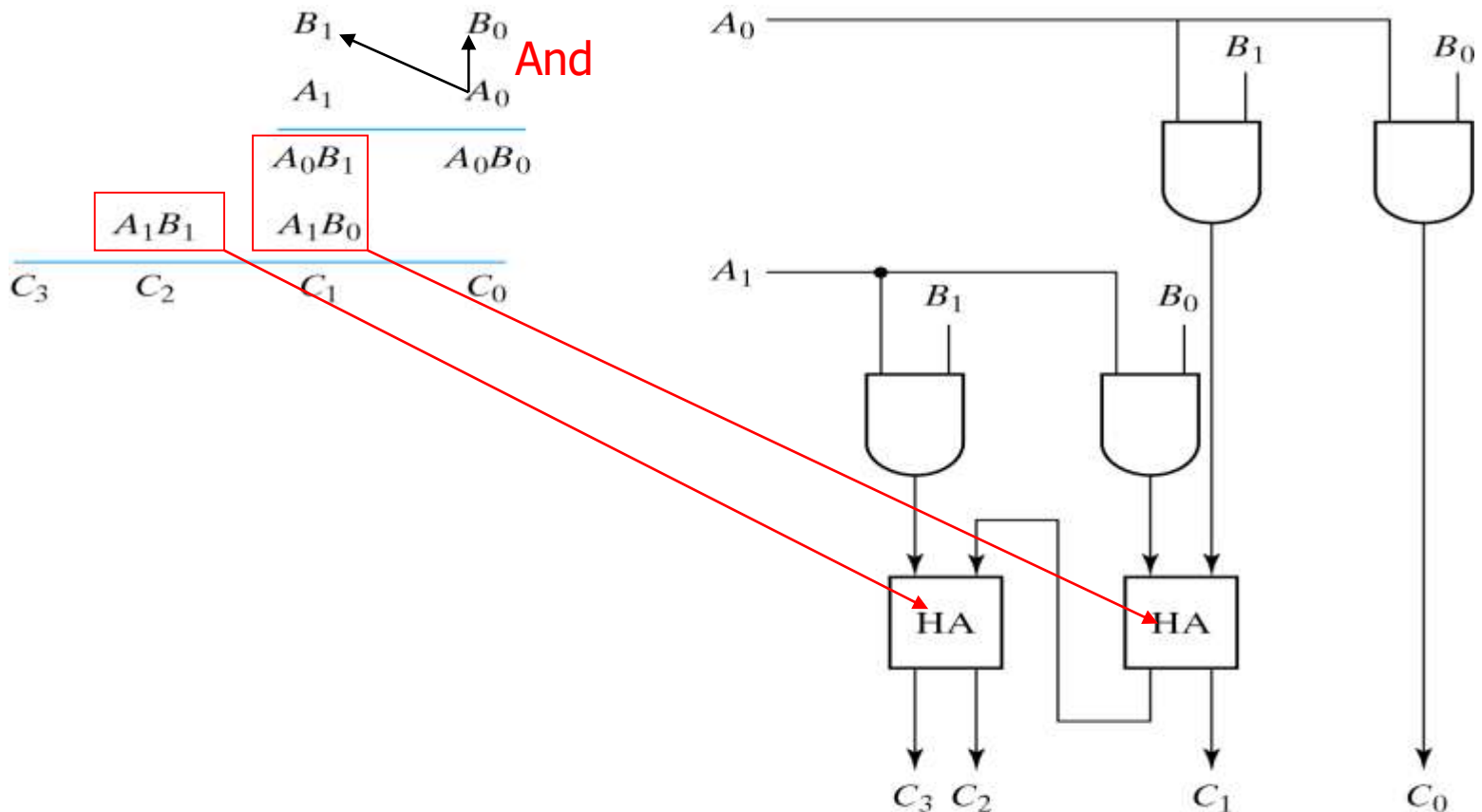
Implementation of BCD adder

- A decimal parallel adder that adds n decimal digits needs n BCD adder stages.
- The **output carry from one stage** must **be connected** to the input carry of the next higher-order stage.



Binary multiplier

- Usually there are **more bits** in the partial products and it is necessary to use **full adders** to produce the sum of the partial products.



3-bit by 3-bit binary multiplier

- For **J multiplier** bits and **K multiplicand** bits we need **(J X K) AND gates** and **(J – 1) K-bit adders** to produce a product of J+K bits.
- K=3 and J=3, we need 12 AND gates and two 3-bit adders.

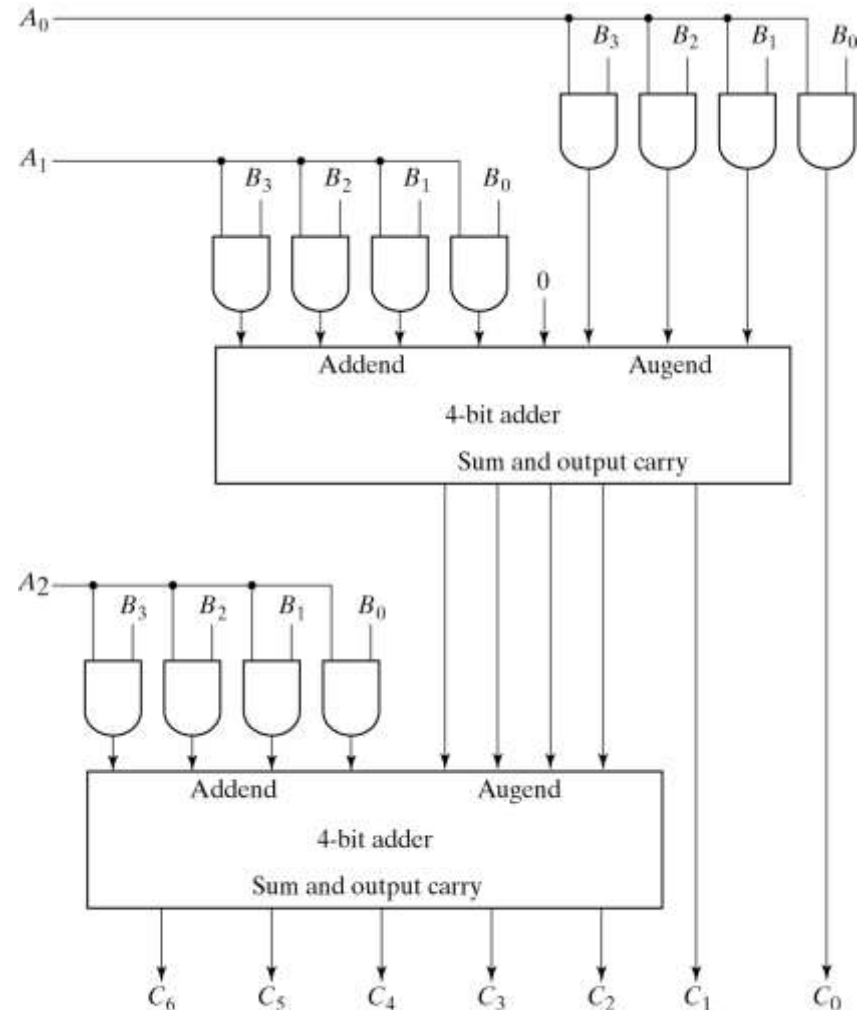


Fig. 4-16 4-Bit by 3-Bit Binary Multiplier

Magnitude comparator

- The equality relation of each pair of bits can be expressed logically with an exclusive-NOR function as:

$$A = A_3A_2A_1A_0 ; B = B_3B_2B_1B_0$$

$$x_i = A_i B_i + A_i' B_i' \quad \text{for } i = 0, 1, 2, 3$$

$$(A = B) = x_3 x_2 x_1 x_0$$

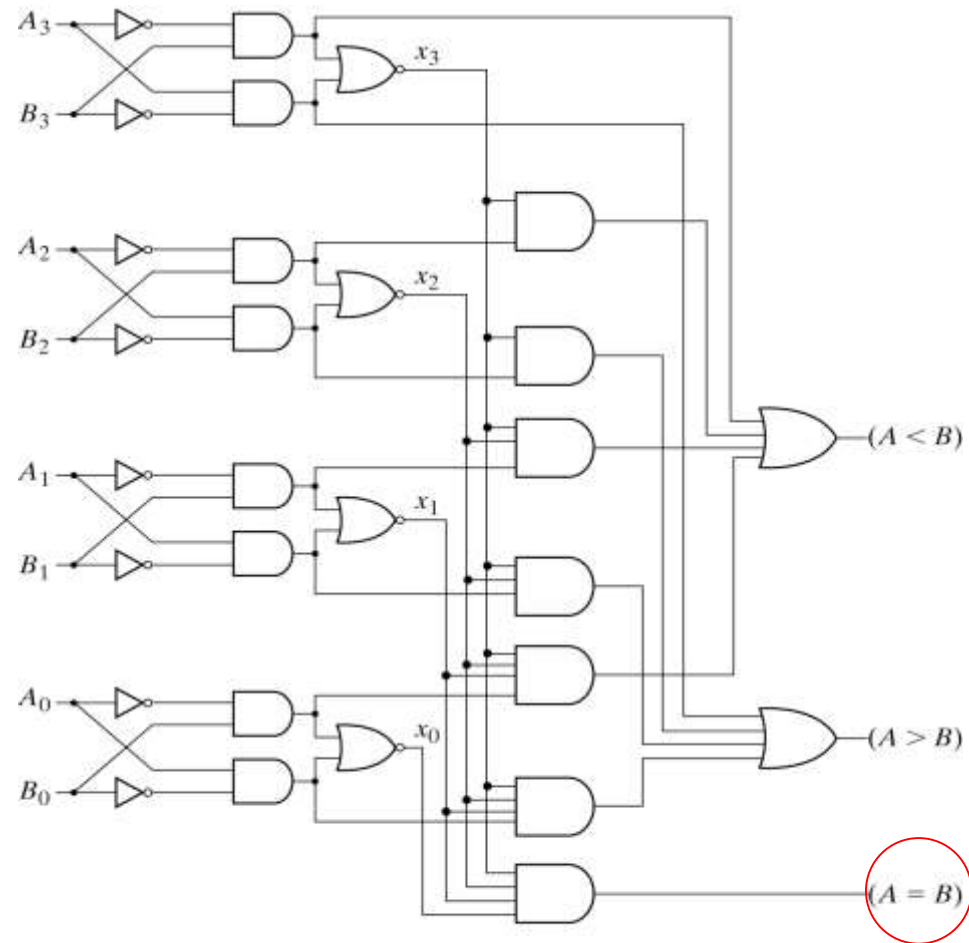


Fig. 4-17 4-Bit Magnitude Comparator

Magnitude comparator

- We inspect the relative magnitudes of pairs of MSB. If equal, we compare the next lower significant pair of digits until a pair of unequal digits is reached.
- If the corresponding digit of A is 1 and that of B is 0, we conclude that $A > B$.

$$(A > B) =$$

$$A_3 B'_3 + x_3 A_2 B'_2 + x_3 x_2 A_1 B'_1 + x_3 x_2 x_1 A_0 B'_0$$

$$(A < B) =$$

$$A'_3 B_3 + x_3 A'_2 B_2 + x_3 x_2 A'_1 B_1 + x_3 x_2 x_1 A'_0 B_0$$

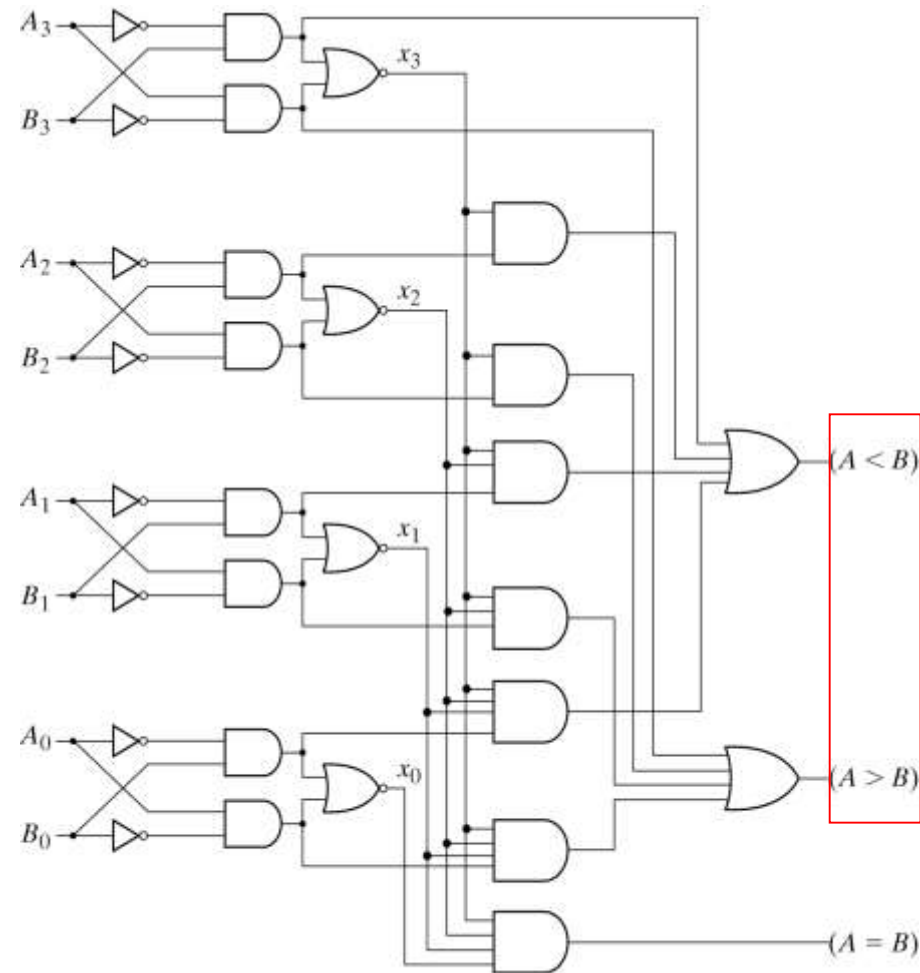


Fig. 4-17 4-Bit Magnitude Comparator



Decoders

- The decoder is called n-to-m-line decoder, where $m \leq 2^n$.
- the decoder is also used in conjunction with other code converters such as a BCD-to-seven_segment decoder.
- 3-to-8 line decoder: For each possible input combination, there are seven outputs that are equal to 0 and only one that is equal to 1.

Implementation and truth table

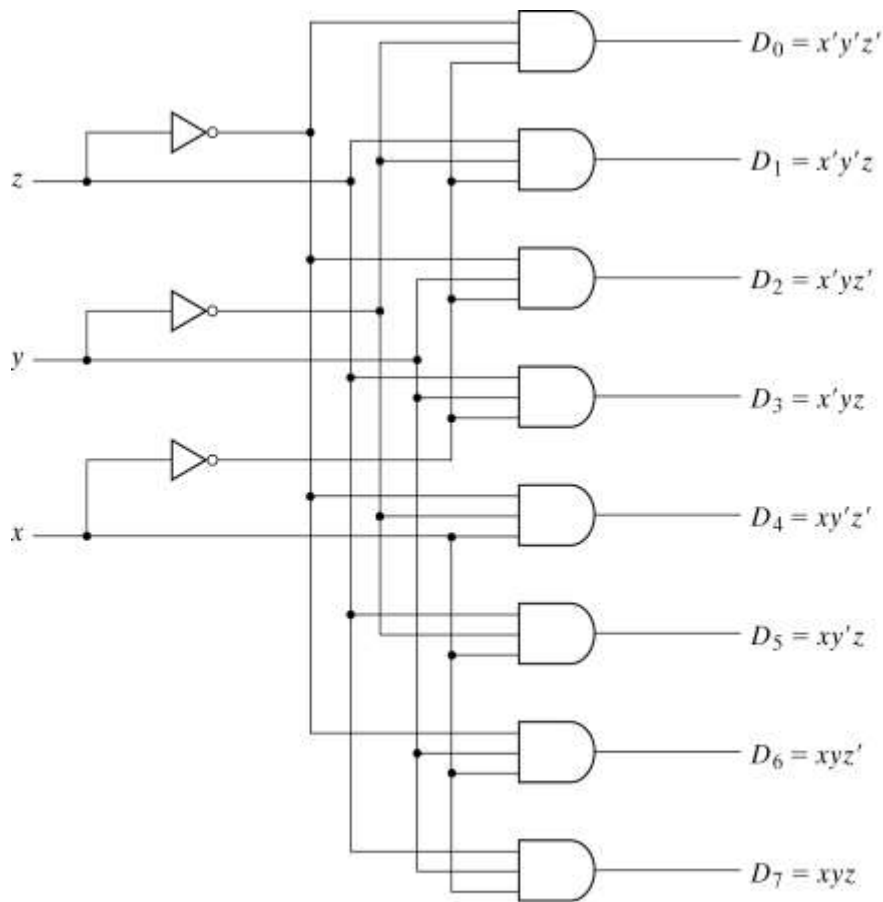


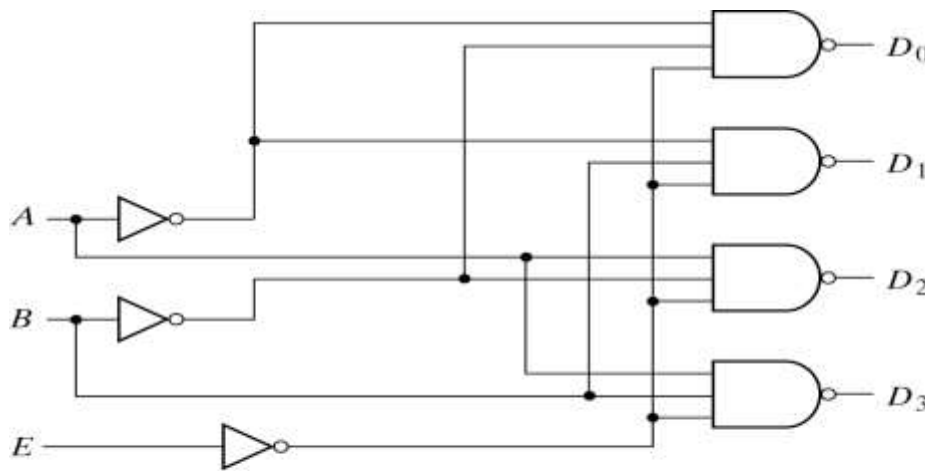
Fig. 4-18 3-to-8-Line Decoder

Table 4-6
Truth Table of a 3-to-8-Line Decoder

Inputs			Outputs							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Decoder with enable input

- Some decoders are constructed with NAND gates, it becomes more economical to generate the decoder minterms in their complemented form.
- As indicated by the truth table, only one output can be equal to 0 at any given time, all other outputs are equal to 1.



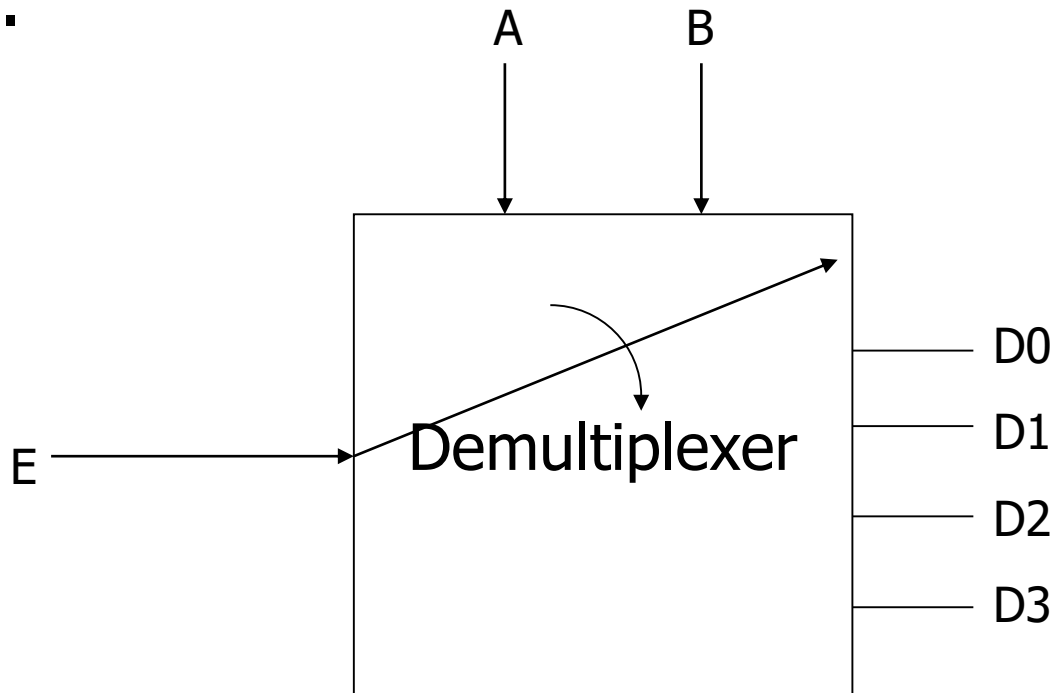
(a) Logic diagram

E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

(b) Truth table

Demultiplexer

- A decoder with an enable input is referred to as a decoder/demultiplexer.
- The truth table of demultiplexer is the same with decoder.



3-to-8 decoder with enable implement the 3-to-16 decoder

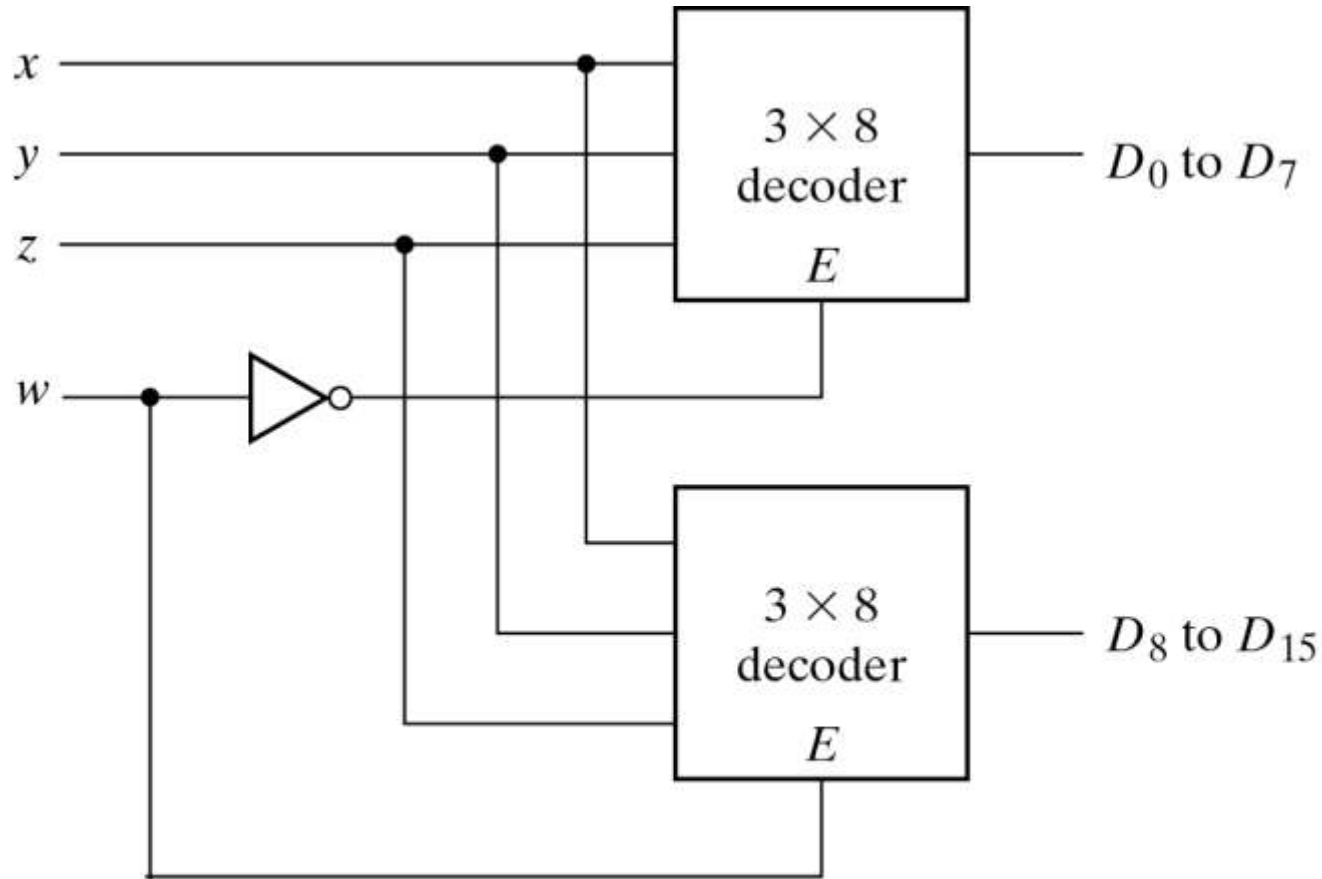


Fig. 4.20 4 × 16 Decoder Constructed with Two 3 × 8 Decoders

Implementation of a Full Adder with a Decoder

- From table 3-3, we obtain the functions for the combinational circuit in sum of minterms:

$$S(x, y, z) = \Sigma(1, 2, 3, 7)$$

$$C(x, y, z) = \Sigma(3, 5, 6, 7)$$

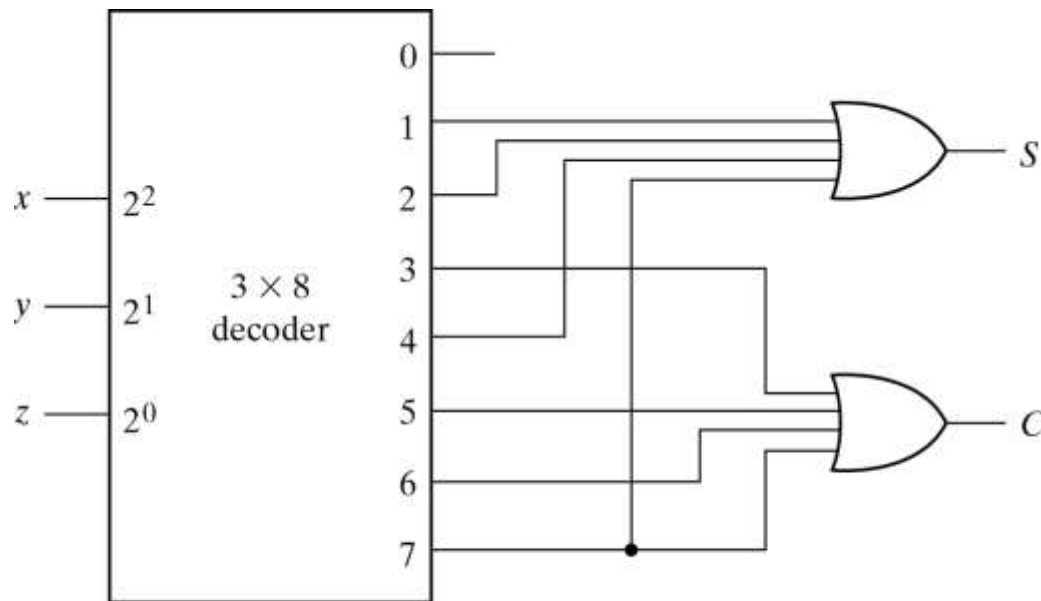


Fig. 4-21 Implementation of a Full Adder with a Decoder

3-9. Encoders

- An **encoder** is the **inverse operation of a decoder**.
- We can derive the Boolean functions by table 3-7

$$z = D_1 + D_3 + D_5 + D_7$$

$$y = D_2 + D_3 + D_6 + D_7$$

$$x = D_3 + D_5 + D_6 + D_7$$

Table 4-7
Truth Table of Octal-to-Binary Encoder

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1



Priority encoder

- If two **inputs** are **active simultaneously**, the **output** produces an **undefined combination**. We can establish an input **priority** to ensure that only one input is encoded.
- **Another ambiguity** in the octal-to-binary encoder is that an **output with all 0's** is generated when **all the inputs are 0**; the output is the same as when D_0 is equal to 1.
- The discrepancy tables on Table 3-7 and Table 3-8 can **resolve aforesaid condition by providing one more output** to indicate that at least one input is equal to 1.

Priority encoder

$V=0 \rightarrow$ no valid inputs

$V=1 \rightarrow$ valid inputs

X's in output columns represent **don't-care** conditions

X's in the input columns are useful for representing a truth table in condensed form.

Instead of listing all 16 minterms of four variables.

Table 4-8

Truth Table of a Priority Encoder

Inputs				Outputs		
D_0	D_1	D_2	D_3	x	y	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

3-input priority encoder

- Implementation of table 3-8

$$x = D_2 + D_3$$

$$y = D_3 + D_1 D'_2$$

$$V = D_0 + D_1 + D_2 + D_3$$

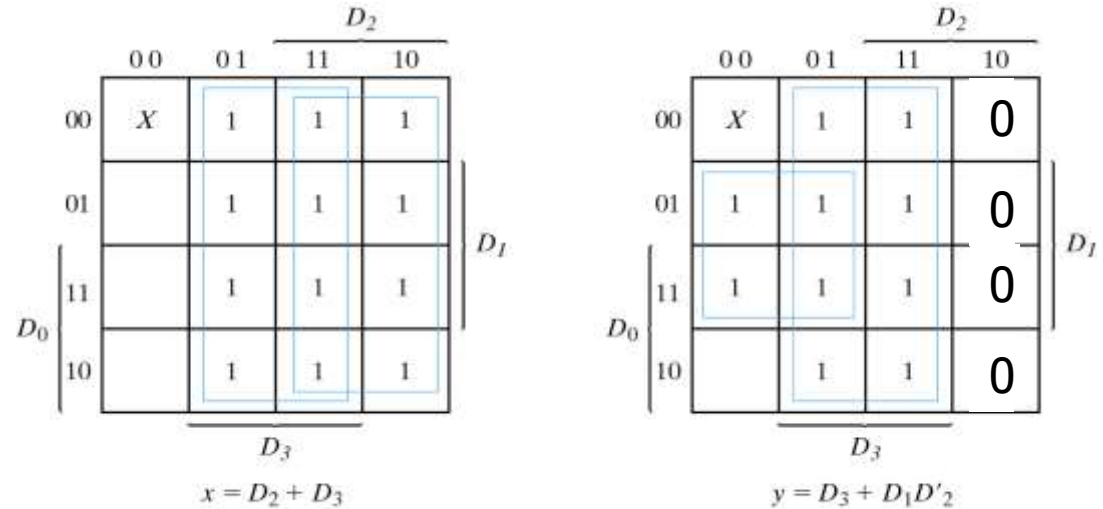
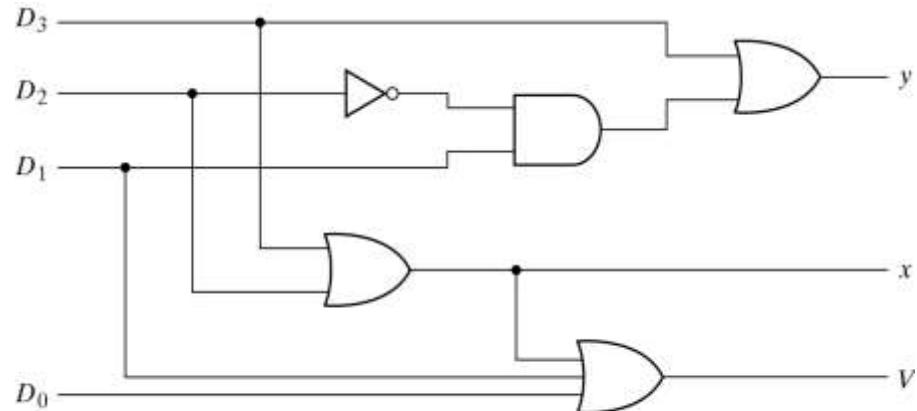


Fig. 4-22 Maps for a Priority Encoder



Multiplexers

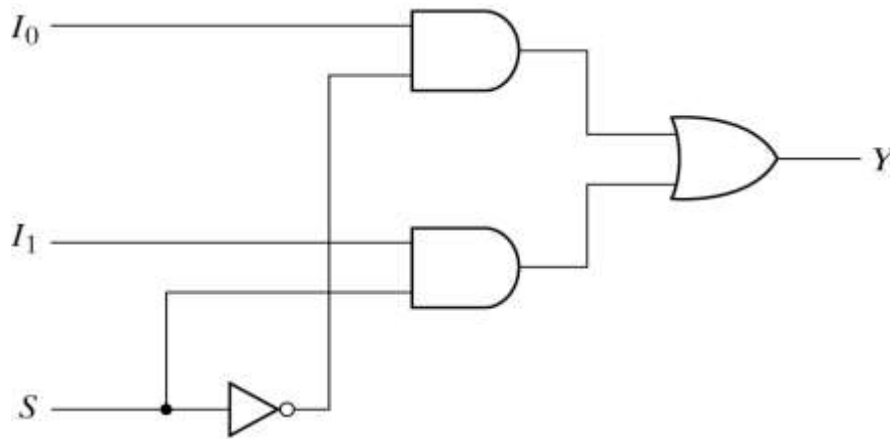
$$S = 0, Y = I_0$$

$$S = 1, Y = I_1$$

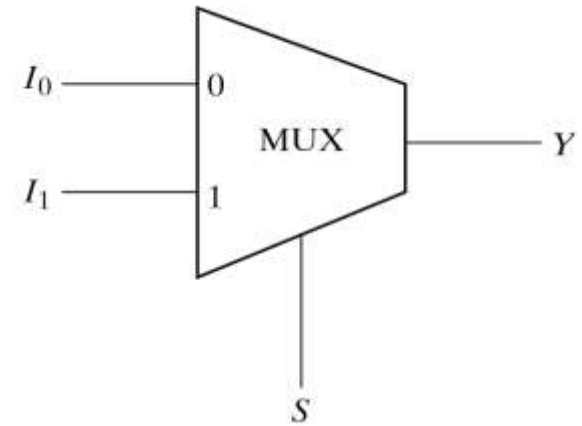
Truth Table \rightarrow

S	Y
0	I_0
1	I_1

$$Y = S'I_0 + SI_1$$



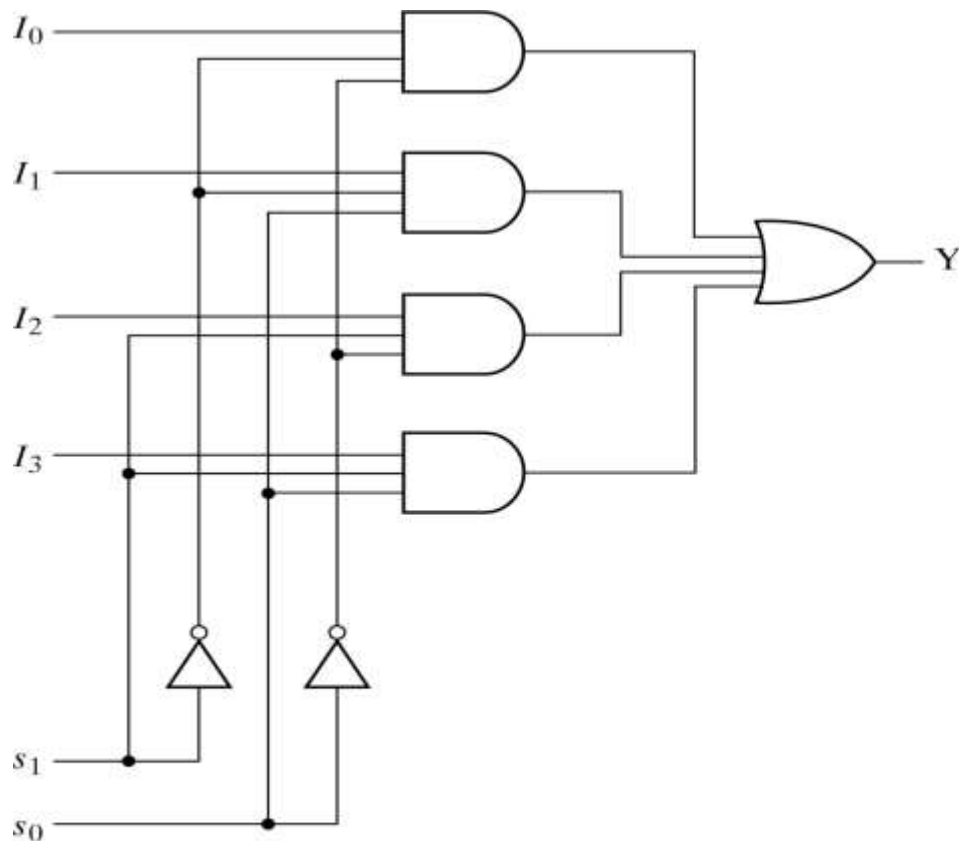
(a) Logic diagram



(b) Block diagram

Fig. 4-24 2-to-1-Line Multiplexer

3-to-1 Line Multiplexer



(a) Logic diagram

s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

(b) Function table

Fig. 4-25 4-to-1-Line Multiplexer
Combinational Logic

Quadruple 2-to-1 Line Multiplexer

- Multiplexer circuits can be combined with common selection inputs to provide multiple-bit selection logic. Compare with Fig3-23.

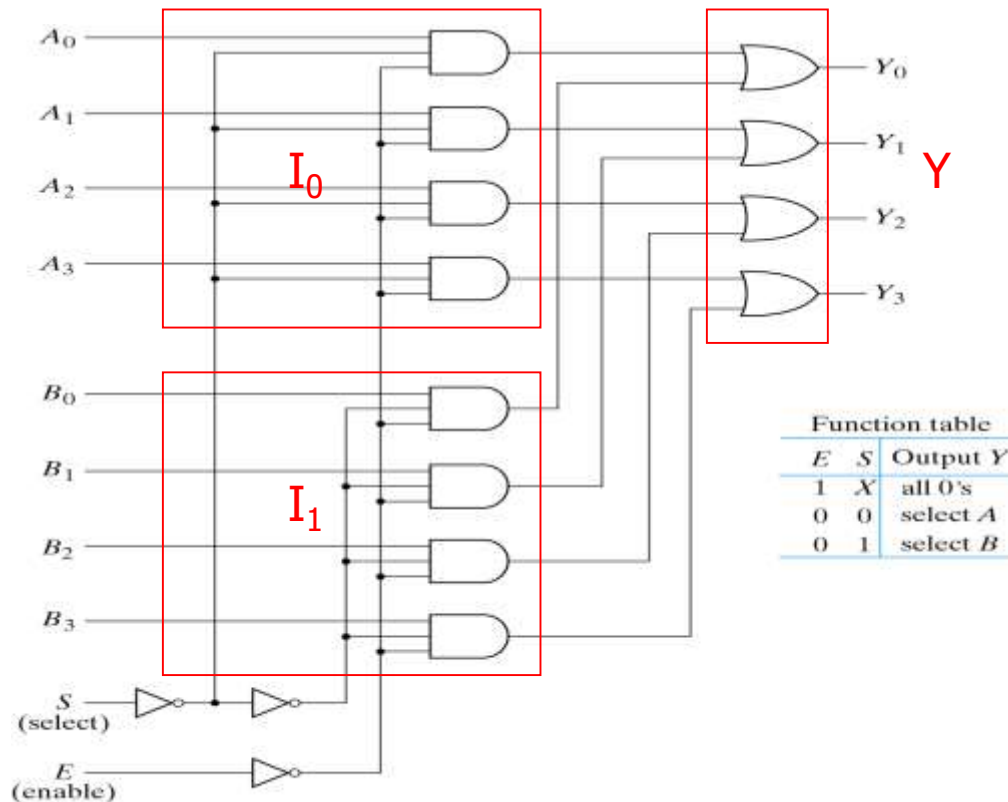


Fig. 4-26 Quadruple 2-to-1-Line Multiplexer

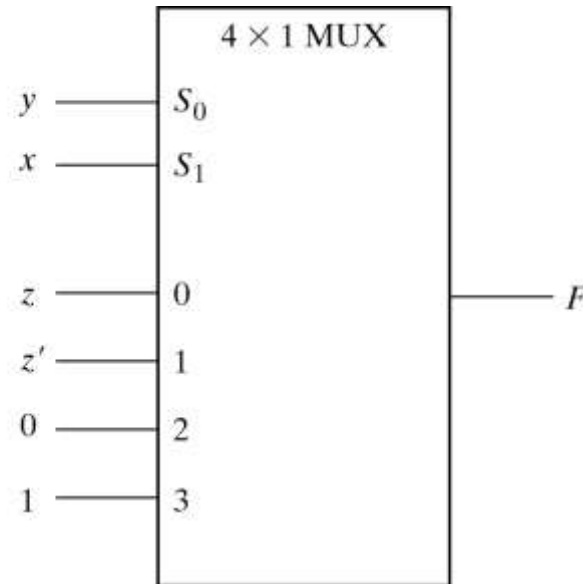
Boolean function implementation

- A more efficient method for implementing a Boolean function of n variables with a multiplexer that has $n-1$ selection inputs.

$$F(x, y, z) = \Sigma(1,2,6,7)$$

x	y	z	F	
0	0	0	0	$F = z$
0	0	1	1	
0	1	0	1	$F = z'$
0	1	1	0	
1	0	0	0	$F = 0$
1	0	1	0	
1	1	0	1	$F = 1$
1	1	1	1	

(a) Truth table

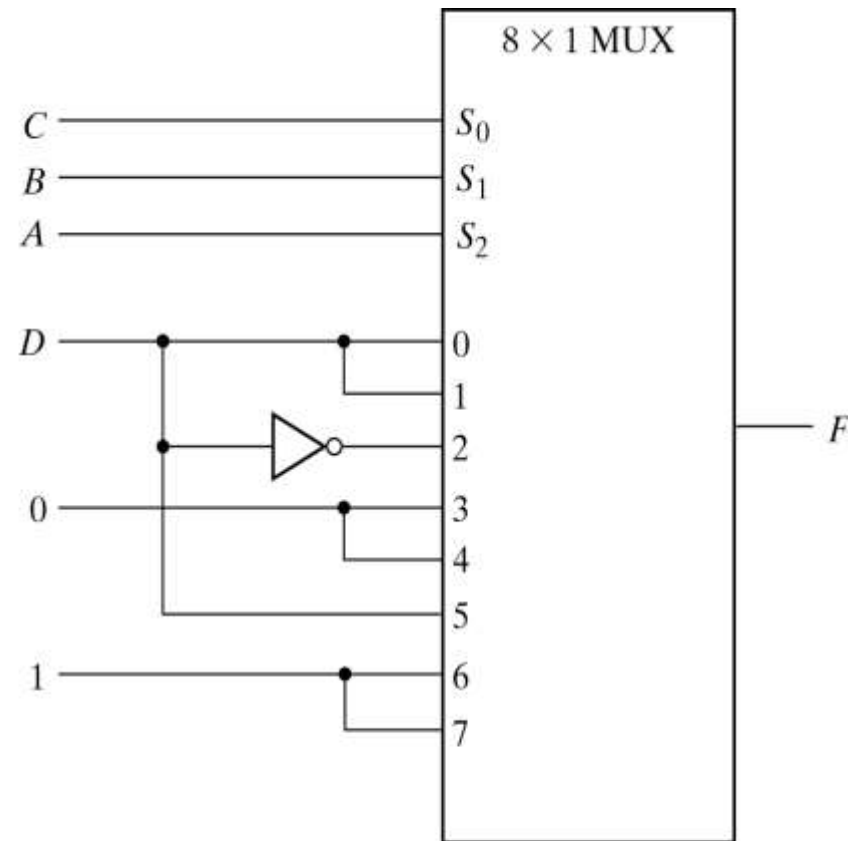


(b) Multiplexer implementation

3-input function with a multiplexer

$$F(A, B, C, D) = \Sigma(1, 3, 3, 11, 12, 13, 13, 15)$$

A	B	C	D	F	
0	0	0	0	0	
0	0	0	1	1	$F = D$
0	0	1	0	0	$F = D$
0	0	1	1	1	
0	1	0	0	1	$F = D'$
0	1	0	1	0	
0	1	1	0	0	$F = 0$
0	1	1	1	0	
1	0	0	0	0	$F = 0$
1	0	0	1	0	
1	0	1	0	0	$F = D$
1	0	1	1	1	
1	1	0	0	1	$F = 1$
1	1	0	1	1	
1	1	1	0	1	$F = 1$
1	1	1	1	1	



Three-State Gates

- A multiplexer can be constructed with three-state gates.

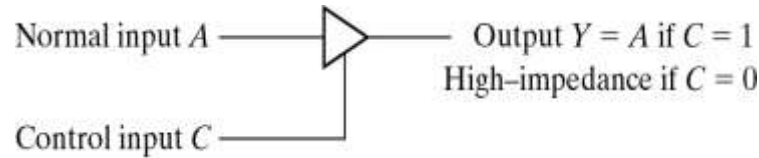


Fig. 4-29 Graphic Symbol for a Three-State Buffer

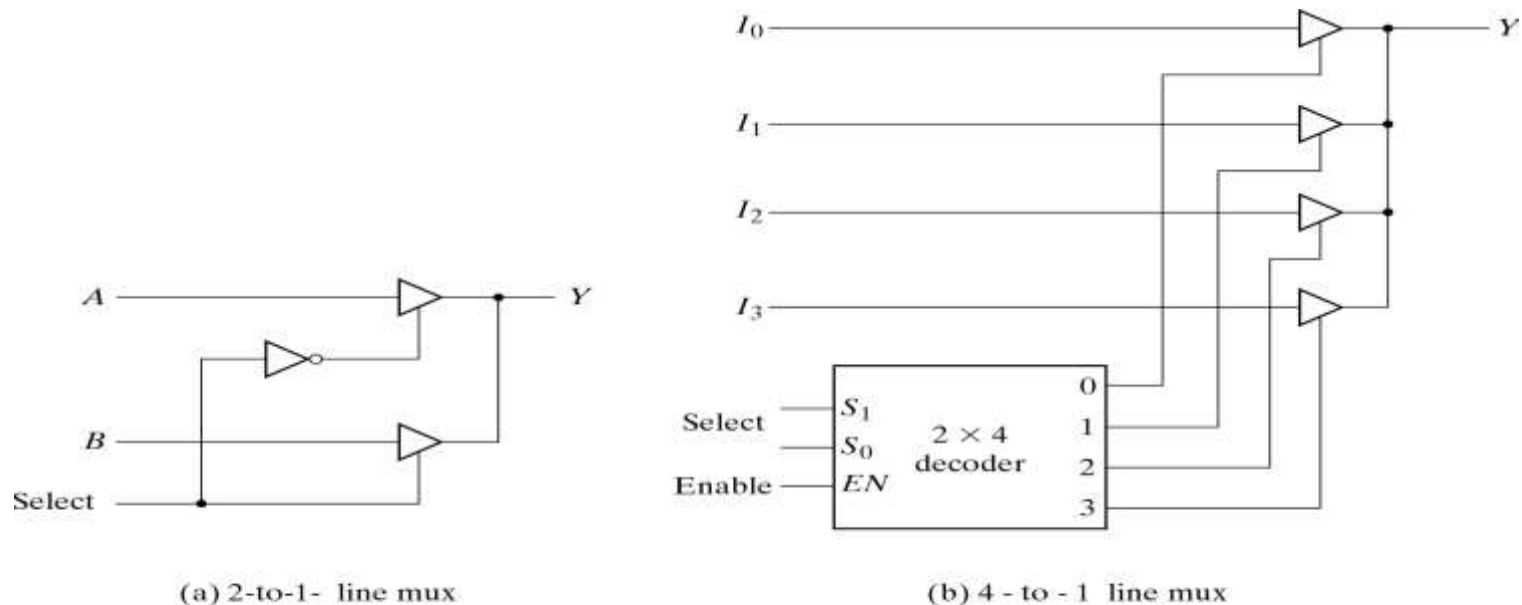


Fig. 4-30 Multiplexers with Three-State Gates
Combinational Logic



HDL for combinational circuits

- A module can be described in any one of the following modeling techniques:
 1. **Gate-level** modeling using instantiation of **primitive gates** and **user-defined modules**.
 2. **Dataflow** modeling using continuous assignment statements with **keyword assign**.
 3. **Behavioral** modeling using procedural assignment statements with **keyword always**.

Gate-level Modeling

- A circuit is specified by its logic gates and their interconnection.
- Verilog recognizes **12 basic gates** as **predefined primitives**.
- The logic values of each gate may be 1, 0, x(unknown), z(high-impedance).

Table 4-9

Truth Table for Predefined Primitive Gates

and	0	1	x	z
0	0	0	0	0
1	0	1	x	x
x	0	x	x	x
z	0	x	x	x

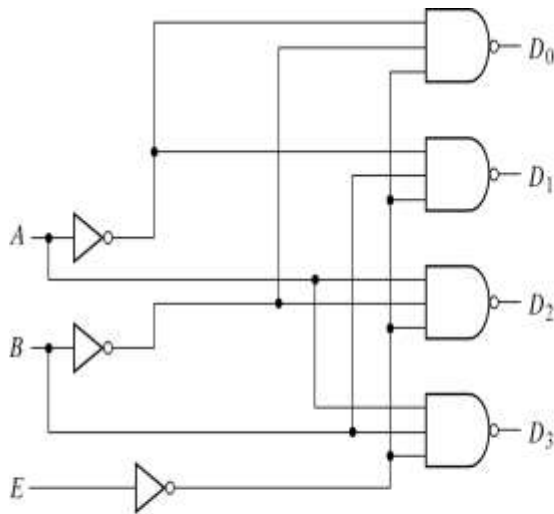
or	0	1	x	z
0	0	1	x	x
1	1	1	1	1
x	x	1	x	x
z	x	1	x	x

xor	0	1	x	z
0	0	1	x	x
1	1	0	x	x
x	x	x	x	x
z	x	x	x	x

not	input	output
	0	1
	1	0
	x	x
	z	x

Gate-level description on Verilog code

The **wire** declaration is for internal connections.



(a) Logic diagram

E	A	B	D ₀	D ₁	D ₂	D ₃
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

(b) Truth table

Fig. 4-19 2-to-4-Line Decoder with Enable Input

HDL Example 4-1

```
//Gate-level description of a 2-to-4-line decoder  
//Figure 4-19
```

```
module decoder_g1 (A,B,E,D);  
  input A,B,E;  
  output [0:3]D;  
  wire Anot,Bnot,Enot;  
  not  
    n1 (Anot,A),  
    n2 (Bnot,B),  
    n3 (Enot,E);  
  nand  
    n4 (D[0],Anot,Bnot,Enot),  
    n5 (D[1],Anot,B,Enot),  
    n6 (D[2],A,Bnot,Enot),  
    n7 (D[3],A,B,Enot);  
endmodule
```



Design methodologies

- There are two basic types of design methodologies: **top-down** and **bottom-up**.
- Top-down: the top-level block is defined and then the sub-blocks necessary to build the top-level block are identified.(Fig.3-9 binary adder)
- Bottom-up: the building blocks are first identified and then combined to build the top-level block.(Example 3-2 3-bit adder)

A bottom-up hierarchical description

HDL Example 4-2

```
//Gate-level hierarchical description of 4-bit adder
// Description of half adder (see Fig 4-5b)
module halfadder (S,C,x,y);
    input x,y;
    output S,C;
    //Instantiate primitive gates
    xor (S,x,y);
    and (C,x,y);
endmodule
```



Full-adder

```
//Description of full adder (see Fig 4-8)
module fulladder (S,C,x,y,z);
    input x,y,z;
    output S,C;
    wire S1,D1,D2; //Outputs of first XOR and two AND gates
//Instantiate the halfadder
    halfadder HA1 (S1,D1,x,y),
                HA2 (S,D2,S1,z);
    or g1(C,D2,D1);
endmodule
```



3-bit adder

```
//Description of 4-bit adder (see Fig 4-9)
module _4bit_adder (S,C4,A,B,C0);
    input [3:0] A,B;
    input C0;
    output [3:0] S;
    output C4;
    wire C1,C2,C3; //Intermediate carries
//Instantiate the fulladder
    fulladder FA0 (S[0],C1,A[0],B[0],C0),
                FA1 (S[1],C2,A[1],B[1],C1),
                FA2 (S[2],C3,A[2],B[2],C2),
                FA3 (S[3],C4,A[3],B[3],C3);

endmodule
```


Three state gates

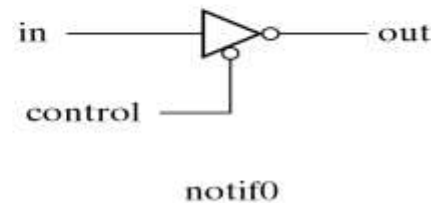
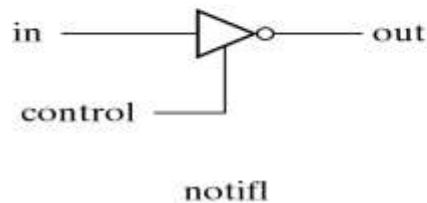
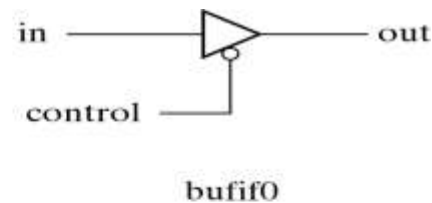
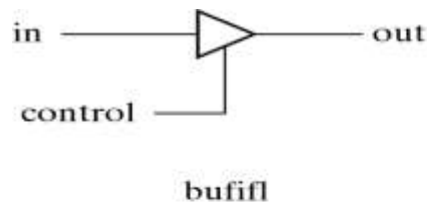
Gates statement: gate name(output, input, control)

>> **bufif1(OUT, A, control);**

A = OUT when control = 1, OUT = z when control = 0;

>> **notif0(Y, B, enable);**

Y = B' when enable = 0, Y = z when enable = 1;



2-to-1 multiplexer

- HDL uses the keyword **tri** to indicate that the output has multiple drivers.

```
module muxtri (A, B, select, OUT);  
  input A,B,select;  
  output OUT;  
  tri OUT;  
  bufif1 (OUT,A,select);  
  bufif0 (OUT,B,select);  
endmodule
```

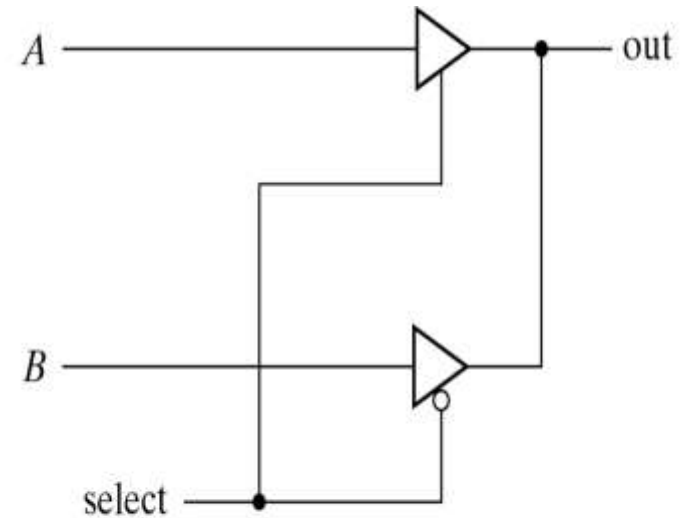


Fig. 4-32 2-to-1-Line Multiplexer with Three-State Buffers



Dataflow modeling

- It uses a number of operators that act on operands to produce desired results. Verilog HDL provides about 30 operator types.

Table 3-10

Symbol	Operation
+	binary addition
-	binary subtraction
&	bit-wise AND
	bit-wise OR
^	bit-wise XOR
~	bit-wise NOT
==	equality
>	greater than
<	less than
{ }	concatenation
?:	conditional

Dataflow modeling

- A continuous **assignment** is a statement that assigns a value to a net.
- The data type **net** is used in Verilog HDL to represent a physical connection **between circuit elements**.
- A **net** defines a gate output declared by an **output** or **wire**.

HDL Example 4-3

```
//Dataflow description of a 2-to-4-line decoder
//See Fig. 4-19
module decoder_df (A,B,E,D);
    input A,B,E;
    output [0:3] D;
    assign D[0] = ~(~A & ~B & ~E),
           D[1] = ~(~A & B & ~E),
           D[2] = ~(A & ~B & ~E),
           D[3] = ~(A & B & ~E);
endmodule
```



Dataflow description of 3-bit adder

HDL Example 3-3

```
//Dataflow description of 3-bit adder
module binary_adder (A,B,Cin,SUM,Cout);
input [3:0] A,B;
input Cin;
output [3:0] SUM;
output Cout;
assign {Cout,SUM} = A + B +Cin;
endmodule
```

Data flow description of a 3-bit comparator

HDL Example 4-5

```
//Dataflow description of a 4-bit comparator.  
module magcomp (A, B, ALSB, AGTB, AEQB);  
    input [3:0] A, B;  
    output ALTB, AGTB, AEQB;  
    assign ALTB = (A < B),  
           AGTB = (A > B),  
           AEQB = (A == B);  
endmodule
```

Dataflow description of 2-1 multiplexer

- Conditional operator(? :)
- Condition? true-expression : false-expression;

HDL Example 4-6

```
//Dataflow description of 2-to-1-line multiplexer
module mux2x1_df (A,B,select,OUT);
    input A,B,select;
    output OUT;
    assign OUT = select ? A : B;
endmodule
```



Behavioral modeling

- It is used mostly to describe **sequential circuits**, but can be used also to describe **combinational circuits**.
- **Behavioral** descriptions use the keyword **always** followed by a list of procedural assignment statements.
- The **target output** of procedural assignment statements must be of the **reg** data type. Contrary to the **wire** data type, where the target output of an assignment may be **continuously updated**, a **reg** data type **retains its value until a new value is assigned**.

Behavioral description of 2-1 multiplexer

HDL Example 4-7

```
//Behavioral description of 2-to-1-line multiplexer
module mux2x1_bh(A,B,select,OUT);
    input A,B,select;
    output OUT;
    reg OUT;
    always @ (select or A or B)
        if (select == 1) OUT = A;
        else OUT = B;
endmodule
```

3-to-1-line Multiplexer

HDL Example 4-8

```
//Behavioral description of 4-to-1- line multiplexer
//Describes the function table of Fig. 4-25(b).
module mux4x1_bh (i0,i1,i2,i3,select,y);
    input i0,i1,i2,i3;
    input [1:0] select;
    output y;
    reg y;
    always @ (i0 or i1 or i2 or i3 or select)
        case (select)
            2'b00: y = i0;
            2'b01: y = i1;
            2'b10: y = i2;
            2'b11: y = i3;
        endcase
endmodule
```



Writing a simple test bench

- In addition to the always statement, test benches use the initial statement to provide stimulus to the circuit under test.
- The **always** statement executes **repeatedly** in a loop. The **initial** statement executes **only once** starting from simulation **time=0** and may continue with any operations that are **delayed by a given number of time** units as specified by the **symbol #**.

For example:

```
initial
begin
    A = 0; B = 0;
    #10 A = 1;
    #20 A = 0; B = 1;
end
```

Stimulus and design modules interaction

- The signals of test bench as inputs to the design module are **reg** data type, the outputs of design module to test bench are **wire** data type.

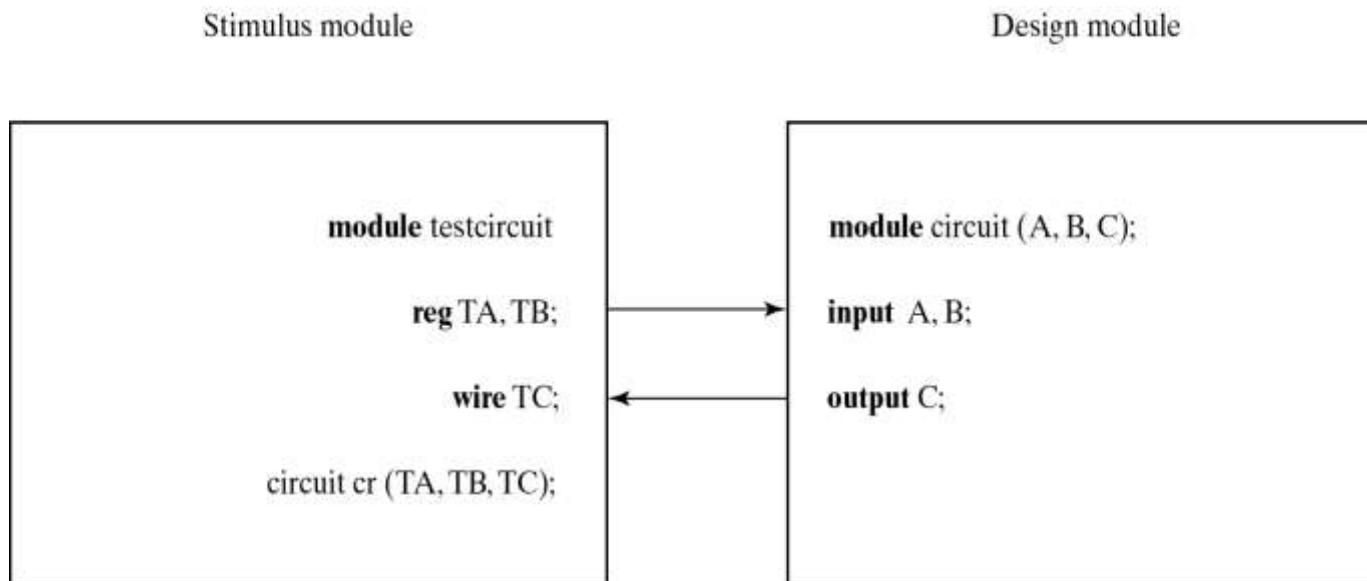


Fig. 4-33 Stimulus and Design Modules Interaction

Example

HDL Example 4-9

```
//Stimulus for mux2x1_df.
module testmux;
  reg TA,TB,TS; //inputs for mux
  wire Y; //output from mux
  mux2x1_df mx (TA,TB,TS,Y); // instantiate mux
  initial
  begin
    TS = 1; TA = 0; TB = 1;
    #10 TA = 1; TB = 0;
    #10 TS = 0;
    #10 TA = 0; TB = 1;
  end
  initial
    $monitor("select = %b A = %b B = %b OUT = %b time = %0d",
            TS, TA, TB, Y, $time);
endmodule
```

```
//Dataflow description of 2-to-1-line multiplexer
//from Example 4-6
module mux2x1_df (A,B,select,OUT);
  input A,B,select;
  output OUT;
  assign OUT = select ? A : B;
endmodule
```

Simulation log:

```
select = 1 A = 0 B = 1 OUT = 0 time = 0
select = 1 A = 1 B = 0 OUT = 1 time = 10
select = 0 A = 1 B = 0 OUT = 0 time = 20
select = 0 A = 0 B = 1 OUT = 1 time = 30
```

Gate Level of Verilog Code of Fig.3-2

HDL Example 4-10

//Gate-level description of circuit of Fig. 4-2

```
module analysis (A,B,C,F1,F2);  
  input  A,B,C;  
  output F1,F2;  
  wire  T1,T2,T3,F2not,E1,E2,E3;  
  or  g1 (T1,A,B,C);  
  and g2 (T2,A,B,C);  
  and g3 (E1,A,B);  
  and g4 (E2,A,C);  
  and g5 (E3,B,C);  
  or  g6 (F2,E1,E2,E3);  
  not g7 (F2not,F2);  
  and g8 (T3,T1,F2not);  
  or  g9 (F1,T2,T3);  
endmodule
```

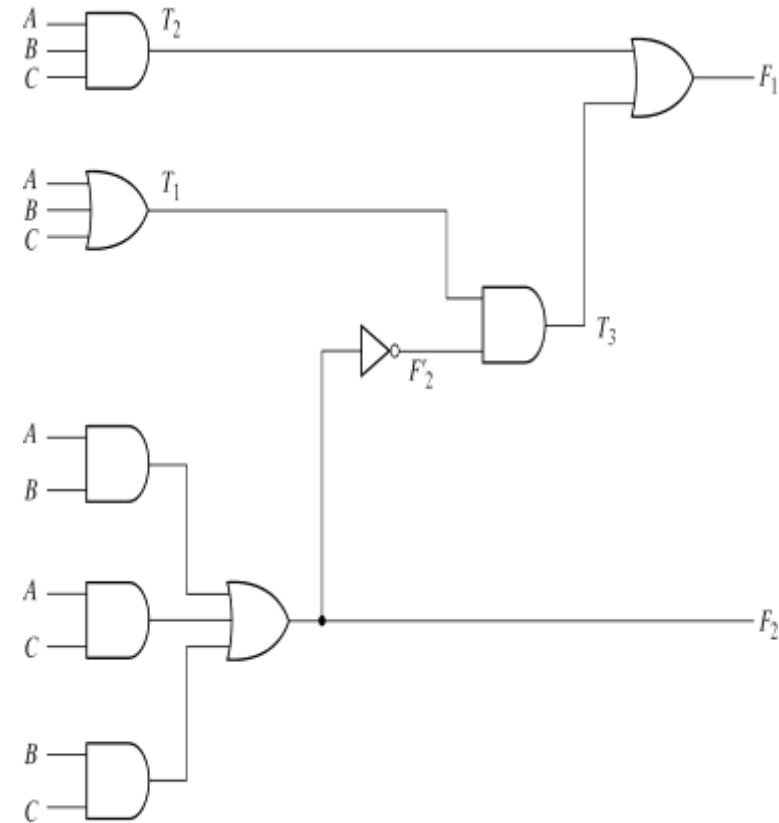


Fig. 4-2 Logic Diagram for Analysis Example

Test Bench of the Figure 3-2

```
//Stimulus to analyze the circuit
module test_circuit;
  reg [2:0]D;
  wire F1,F2;
  analysis fig42(D[2],D[1],D[0],F1,F2);
  initial
    begin
      D = 3'b000;
      repeat(7)
        #10 D = D + 1'b1;
    end
  initial
    $monitor ("ABC = %b F1 = %b F2 =%b ",D, F1, F2);
endmodule
```

Simulation log:

```
ABC = 000 F1 = 0 F2 =0
ABC = 001 F1 = 1 F2 =0
ABC = 010 F1 = 1 F2 =0
ABC = 011 F1 = 0 F2 =1
ABC = 100 F1 = 1 F2 =0
ABC = 101 F1 = 0 F2 =1
ABC = 110 F1 = 0 F2 =1
ABC = 111 F1 = 1 F2 =1
```

Introduction to Cloud Computing

Mr. Asadi Srinivasulu

Associate Professor

Dept. of Information Technology

Unit: I

Subject : Cloud Computing

Target Group: IV B.Tech. I Sem – IT

Cloud Computing

By Mr. Asadi Srinivasulu, Associate Professor

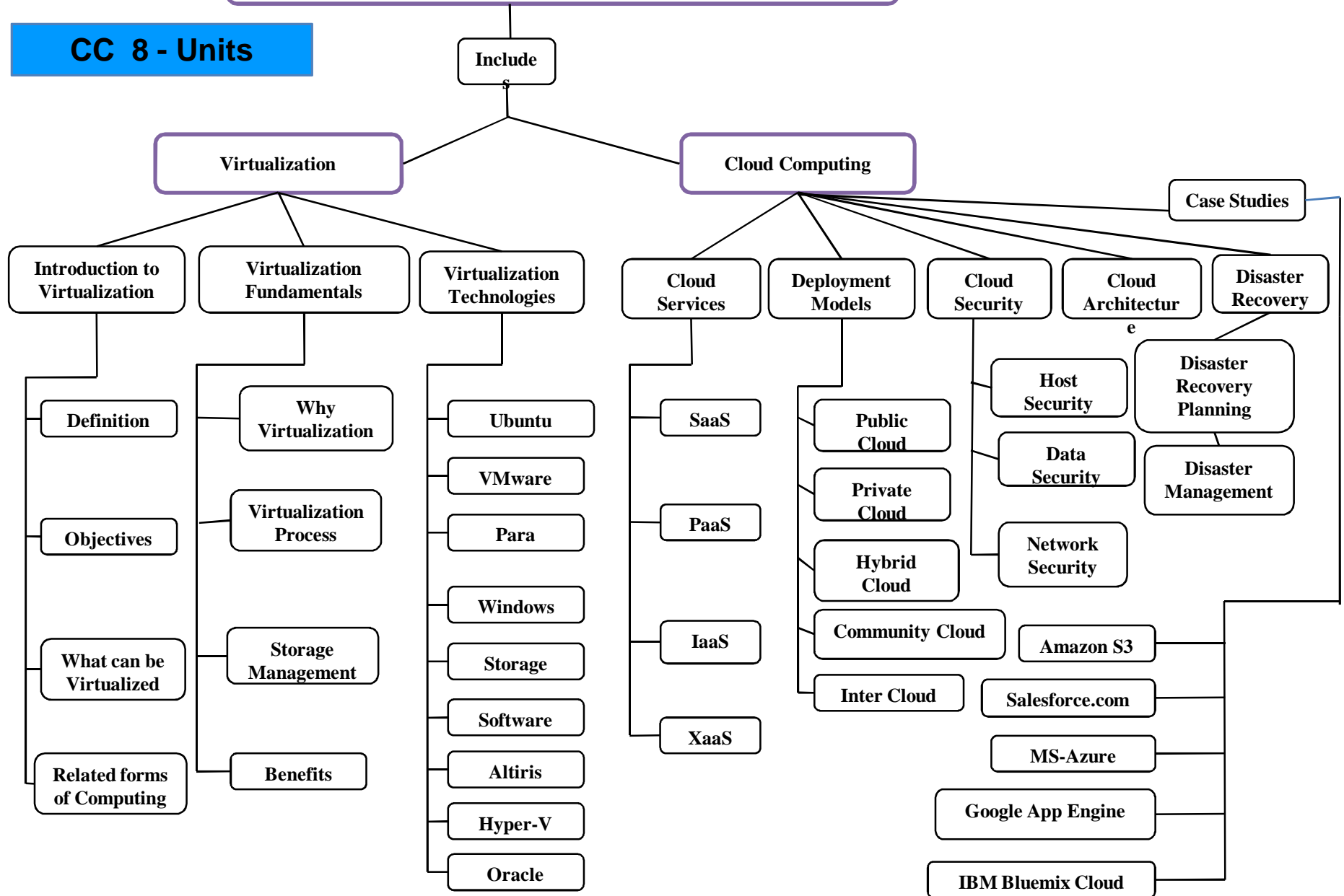
Syllabus

- Unit I : Introduction to Cloud Computing
- Unit II : Cloud Computing Architecture
- Unit III : Introduction to Virtualization
- Unit VI : Virtualization Technologies
- Unit V : Cloud Security
- Unit VI : Disaster Recovery
- Unit VII : Graph Reduction
- Unit VIII : Case Studies (Google App Engine, OBIEE, MS Azure, and Yahoo Hadoop)

VIRTUALIZATION AND CLOUD COMPUTING

CC 8 - Units

CC 8 - Units



1. Cloud Computing Definition

- ✓ “A Model for enabling ubiquitous, convenient, on-demand network access through Internet to a shared pool of configurable computing resources i.e. networks, servers, storage, applications, and services that can be rapidly provisioned and released with minimal management effort or service provider interaction.”
(NIST, USA)
- ✓ **Ex:** Web-based applications such as Google’s Gmail can be accessed in real time from an Internet-connected machine anywhere in the world.

- ✓ **Cloud** is a **Metaphor** for the Internet.
- ✓ Cloud Computing is Internet based computing.
- ✓ Derived from depiction in **Network diagrams(TCP/IP)**.
- ✓ **Cloud is an Internet based development** of applications and services.
- ✓ Cloud Computing is a **Multitenant** Architecture
- ✓ Cloud computing is a **5-4-3** Formula i.e. 5 Characteristics, 4 Infrastructural Models and 3 major Services.
- ✓ Introduced initially by **J.C.K Licklider** and **John. McCarthy** in **1961**.
- ✓ It provides the resources that are as services, those services are **on-demand services** i.e. **Pay and use**
- ✓ Cloud computing is form of **Distributed Computing**.
- ✓ The Heart of the Cloud computing is **“Data center and Virtualization”**.

What is Cloud Computing

Multi-tenant solution provided by vendor

Automated backups, uptime, SLA, maintenance

Automated upgrades

Elastic, pay as you go – *scale up or down*

Modern web based integration

Web and mobile - *access from anywhere*

- ✓ Infrastructure models are also called as Deployment Models.
- ✓ Cloud computing is a **5-4-3** Formula

❖ **Five Characteristics**

1. On Demand Service
2. Broad Network Access
3. Resource Pooling
4. Rapid Elasticity
5. Measured Service

OBR²M

❖ **Four Infrastructure/Deployment Models**

1. Public cloud
2. Private cloud
3. Hybrid cloud
4. Community cloud/Inter Cloud

PPHC

❖ **Three Major Services**

1. Software as a Service (SaaS)
2. Platform as a Service (PaaS)
3. Infrastructure as a Service (IaaS)

SPI

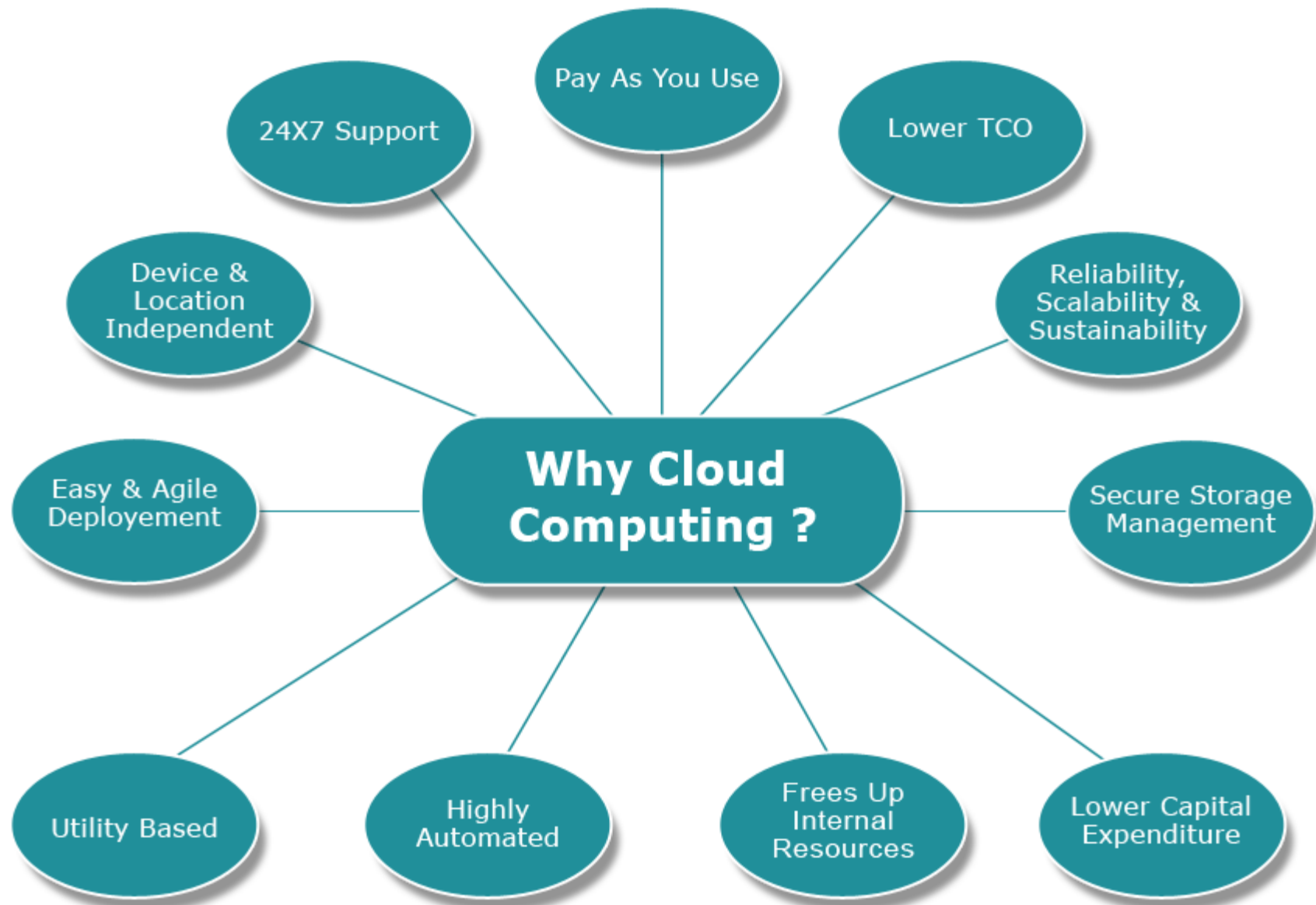


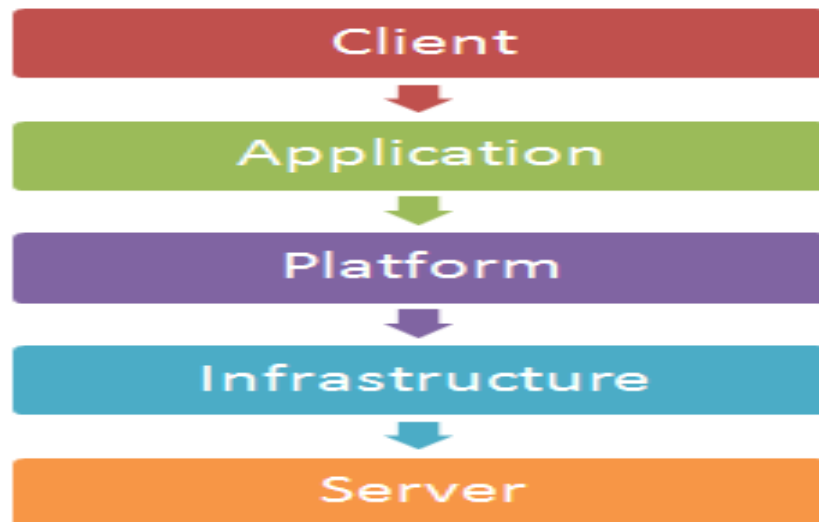
Fig: Why Cloud Computing

- ✓ **Cloud Computing** is a style of computing in which massively scalable IT-related capabilities are provided "as a service" using Internet technologies to multiple external customers.
- ✓ Cloud computing has two important sections
 - ✓ 1) Front end
 - ✓ 2) Back end.
- ✓ **Front end** : Cloud client , Thin client interface, cloud end-user
- ✓ **Connectivity**: Internet.
- ✓ **Back end** : Datacenter and Virtualization software

2) Cloud Computing Layers

Major cloud computing layers are:

- i. **Servers:** Are the physical servers where the data is stored.
- ii. **Infrastructure:** It's a platform virtualization environment.
- iii. **Platform:** Simply it's where you write the code of applications that consume infrastructure.
- iv. **Application:** Delivers Software over the internet, eliminating the need to install and run the application on customer's own computers.
- v. **Client:** Computer Hardware or Computer Software that uses the developed application.



3) Cloud Computing Components

- ✓ “A Model for enabling ubiquitous, convenient, on-demand network access through Internet to a shared pool of configurable computing resources i.e. networks, servers, storage, applications, and services that can be rapidly provisioned and released with minimal management effort or service provider interaction.”

- ✓ Cloud Computing components are
 - a) Client
 - b) Datacenters
 - c) Distributed servers
 - d) Cloud consumer
 - e) Cloud Service Provider(CSP)
 - f) Cloud computing SLA
 - g) Cloud broker
 - h) Cloud carrier

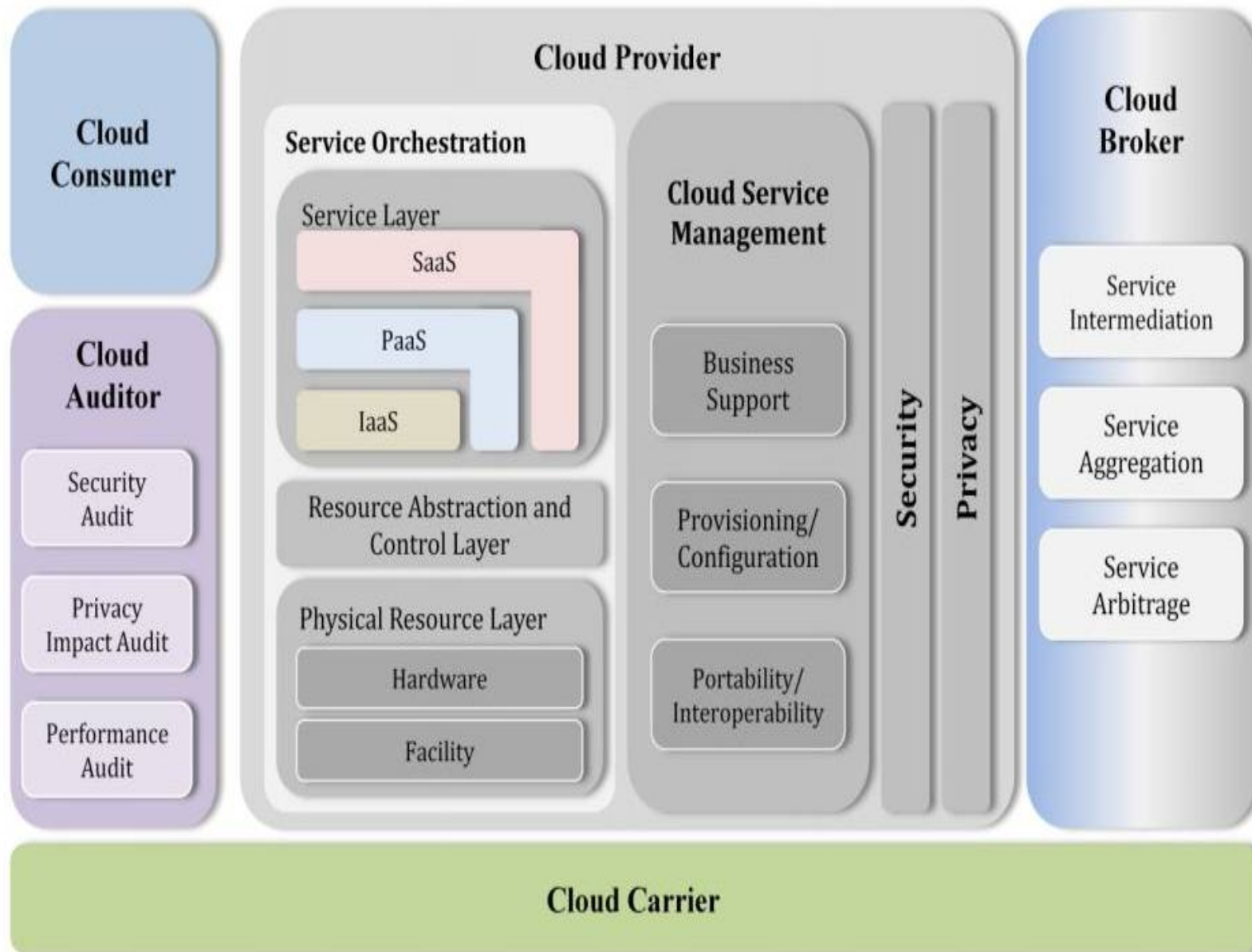


Fig: NIST Reference Model with Components

- a) Client:** It's a computers / devices that sits on desk.
- ✓ End users are those who interact with clients to avail cloud services.
 - ✓ **Ex:** Smart devices, Thin clients, Thick Clients
- b) Data Center:** integration or collection of servers where the applications to which you subscribe is housed.
- ✓ Room full of servers at one place that can be accesses via internet – virtual servers.
- c) Distributed Servers:** Its not mandatory to have all servers in one place they are geographically spread.
- ✓ This gives CSP more options and security.

- d) **Cloud Consumer:** Person or organization that maintains a business relationship with, and uses service from, Cloud Service Providers.
- e) **Cloud Provider:** Person, organization or entity responsible for making a service available to service consumers.
- f) **Cloud Carrier:** The intermediary that provides connectivity and transport of cloud services between Cloud Providers and Cloud Consumers.
- g) **Cloud Broker:** An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers.
- h) **Cloud Auditor:** A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.

4) Cloud Characteristics

- 1. On-Demand Self-Service:** Customers can automatically provision computing capabilities and resources on their own when needed without necessitating any human intervention.
- 2. Broad Network Access:** Access and capabilities are available over the network through standard devices, such as cell phones, laptops, PDAs, etc.
- 3. Resource Pooling:** Resources such as network bandwidth, virtual machines, memory, processing power, storage capacity, etc. are pooled together to serve multiple customers using a multi-tenant model. That is, virtual and physical resources are dynamically assigned and reassigned based on need and customers' demands.
- 4. Rapid Elasticity:** Depending on demand, resources and capabilities can be quickly and automatically deployed and scaled at any quantity and at any time.
- 5. Measured Service:** Customer usage of the vendor's resources and services are automatically monitored, controlled and reported offering a high level of transparency for the customer and vendor.

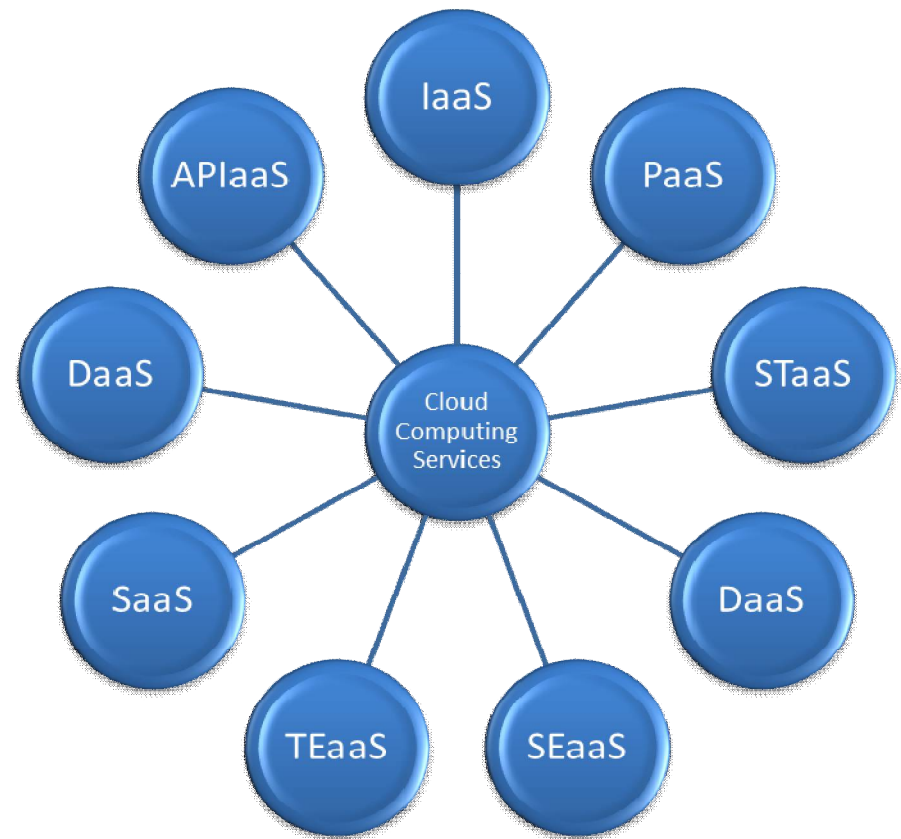


Fig: Cloud Computing Characteristics & Services

5) Cloud Computing Services: These services are delivering **Cloud Computing** Software, Platform, infrastructure – Software's, platforms, servers, storage, network and operating systems – as an on-demand **service**.

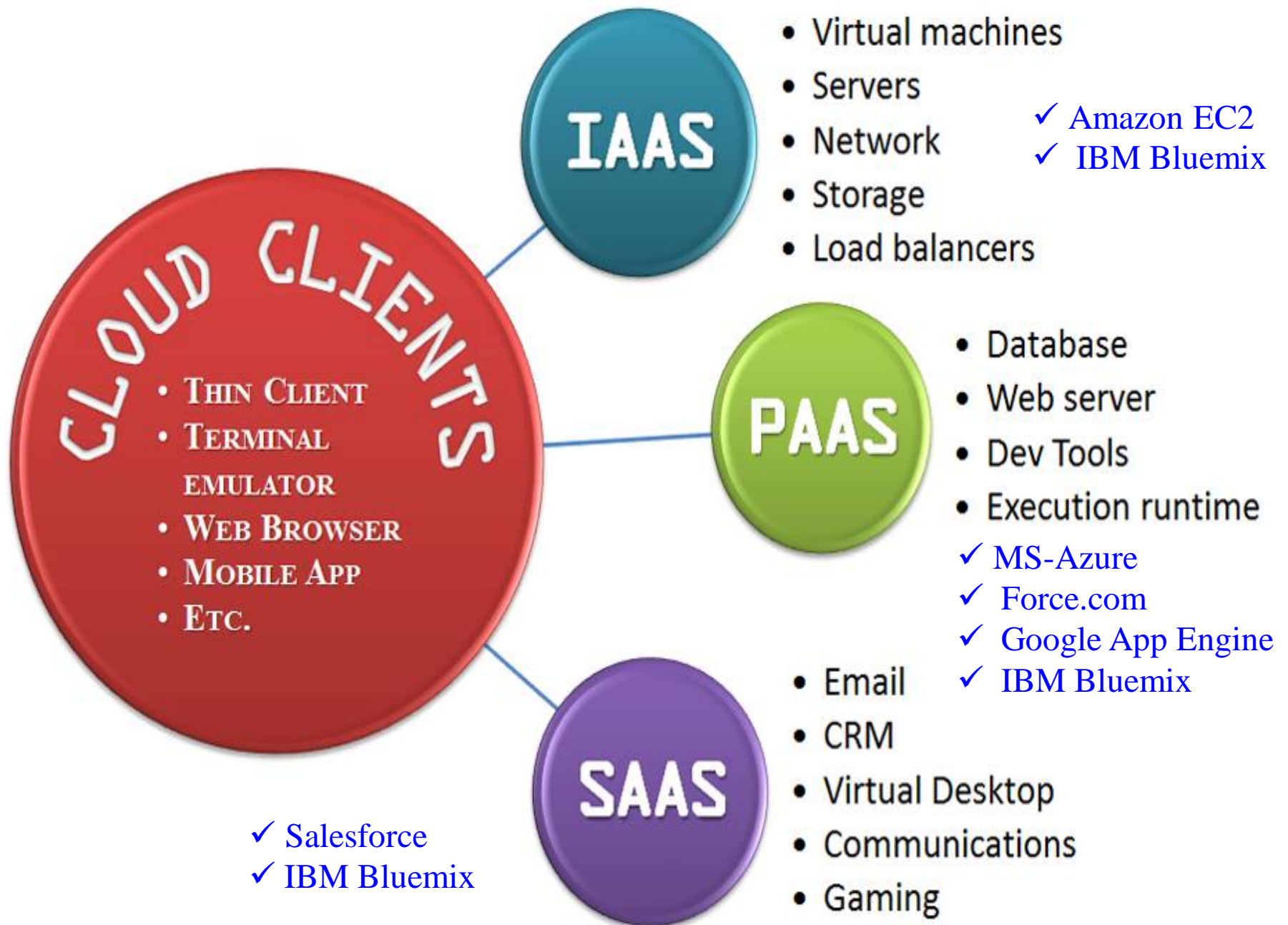
✓ The Cloud Computing services are

1. Software as a Service (SaaS)
2. Platform as a Service (PaaS)
3. Infrastructure as a Service (IaaS)
4. Architecture-as-a-Service (AaaS)
5. Database as a Service (DBaaS)
6. Identity-as-a-Service(IDaaS)
7. Testing-as-a-Service(TaaS)
8. Network as a Service (NaaS)
9. Storage-as-a-Service(STaaS)
10. Anything as a Service (XaaS)



1) SaaS (Software-as-a-Service):

- ✓ Software as a Service (SaaS) provides business processes and applications, including CRM, e-mails, collaboration, and so on.
- ✓ Software deployment model whereby a provider licenses an application to customers for use as a service on demand.
- ✓ Software as a Service (SaaS) application is offered as a service.
- ✓ SaaS managed by provider such as Salesforce.com.
- ✓ SaaS helps in optimizing the cost and delivery in exchange of negligible customization and represents a shift of operational risks from the consumer to the provider.
- ✓ SaaS is sometimes referred to as “on-demand software” and is usually priced on a pay-per-use basis. SaaS providers price applications using a subscription fee.



- ✓ Who Uses It : Business Users
- ✓ Why use it : To complete business tasks

Examples:

- ❖ Salesforce.com
- ❖ Microsoft Office 365
- ❖ G-Mail
- ❖ Google app's,
- ❖ Social networks.
- ❖ Cisco, Zoho planner,
- ❖ Google Docs.
- ❖ ERP, VoIP, BI, Supply chain

Characteristics of SaaS include

- ✓ The application is hosted centrally.
- ✓ Outsourcing hardware and software support to the cloud provider.
- ✓ Enhancing the potential of an organization to reduce its IT operational costs.
- ✓ No need to install new software to release updates, any update can be executed by the cloud provider itself not the customers.
- ✓ Software testing takes place at a faster rate as SaaS applications have only one configuration.

2) PaaS: Platform-as-a-Service

- ✓ Platform as a Service (PaaS) is another service model of cloud computing that provides application execution services like application runtime, storage, and integration.
- ✓ PaaS follows a resourceful and responsive approach to operate scale-out applications and make these applications profitable.
- ✓ The consumer controls software deployment and configuration settings.
- ✓ Optimized IT and developer tools offered through Platform as a Service (PaaS) for Database and Testing Environments.
- ✓ It provides computing platform is offered as a Service.
- ✓ It supplies tools and development environment.
- ✓ Customers interact with platform through API.

- ✓ **Who Uses It:** Developers and Deployers
- ✓ **Why use it :** Create or deploy applications and services for users

Examples:

- ❖ MS-Azure, Force.com,
- ❖ Google App Engine
- ❖ Force.com
- ❖ IBM Bluemix

Characteristics of PaaS include

- ✓ Facilitation of hosting capabilities
- ✓ Designing and developing the application
- ✓ Integrating web services and databases
- ✓ Providing security, scalability and storage
- ✓ Versioning the application and application instrumentation
- ✓ Testing and deployment facilities.

3) IaaS (infrastructure-as-a-Service)

- ✓ It is also known as Hardware as a Service ([HaaS](#)), Infrastructure as a Service (IaaS) is a category of cloud computing in which an organization outsources the equipment used to support operations, including storage, servers hardware and networking components.
- ✓ The service provider is the owner of the equipment and is responsible for configuring , running and maintaining it.
- ✓ IaaS offers a standardized, dynamic, flexible and sometimes virtualized environment for the end users.
- ✓ It provides storage, database management, computing capabilities as a on-demand service by the major vendors.
- ✓ It provides on-demand, highly scalable computing, storage and hosting services.
- ✓ It provides delivers storage and computing power.
- ✓ It provides rids of virtualized Servers, Storage devices and Network resources.

- ✓ **Who Uses It:** System Managers
- ✓ **Why use it:** Create platforms for service and application test, development, integration and deployment.
- ❖ **Examples:**
 - ✓ Amazon services - EC2 & S3
 - ✓ IBM Bluemix
 - ✓ GoGrid, 3Terra

Characteristics of IaaS include

- ✓ Virtualization of Desktop
- ✓ Internet availability
- ✓ Use of billing model
- ✓ Computerized administrative tasks
- ✓ Utility computing service
- ✓ Policy-based services
- ✓ Active scaling

4.Architecture-as-a-Service(AaaS): The social application architecture is simulates the Cloud computing architecture. The social networking process involved with Varsity of networks such of Corporate Network, Home area network , Wireless Personal area network n Internet and vehicular area network.

✓ **Eg:** Facebook, Twitter

5.Database-as-a-Service(DaaS):Database-as-a- Service traditionally supported on-premise database management functions are releasing cloud versions of their products.

✓ **Eg:** Oracle OBIEE, MS-SQL,MYSQL, IBMDB2 etc.

6.Identity-as-a-Service(IaaS): Identity-as-a-Service refers to the management of identities in the cloud, apart from the applications and providers that use them.

✓ **Eg:** MS-Azure

7.Testing as a Service(TaaS): It provides online testing platform.

✓ **Ex:** Hp Load Test, Soasta, IBM-RFT, HP-QTP

8.Network as a Service (NaaS): Communications assets and massive subscriber base, the global (mobile) telecom network can tap this market and provide valuable, revenue generating cloud computing capabilities in the form of Network as a Service (NaaS).

✓ **Eg:** Motorola Cloud, Facebook, LinkedIn, Twitter etc.

9.Storage-as-a-Service(StaaS): Storage as a Service is a business model in which a large company rents space in their storage infrastructure to a smaller company or individual.

✓ **E.g.:** Amazon and EMC, GoGrid, 3 Terra, IBM Cloud

10: Any thing as a Service(XaaS): any kind of resources that are provisioned as a service is called anything as a service.

✓ **E.g.:** Amazon and EMC, Salesforce.com, MS-Azure etc.

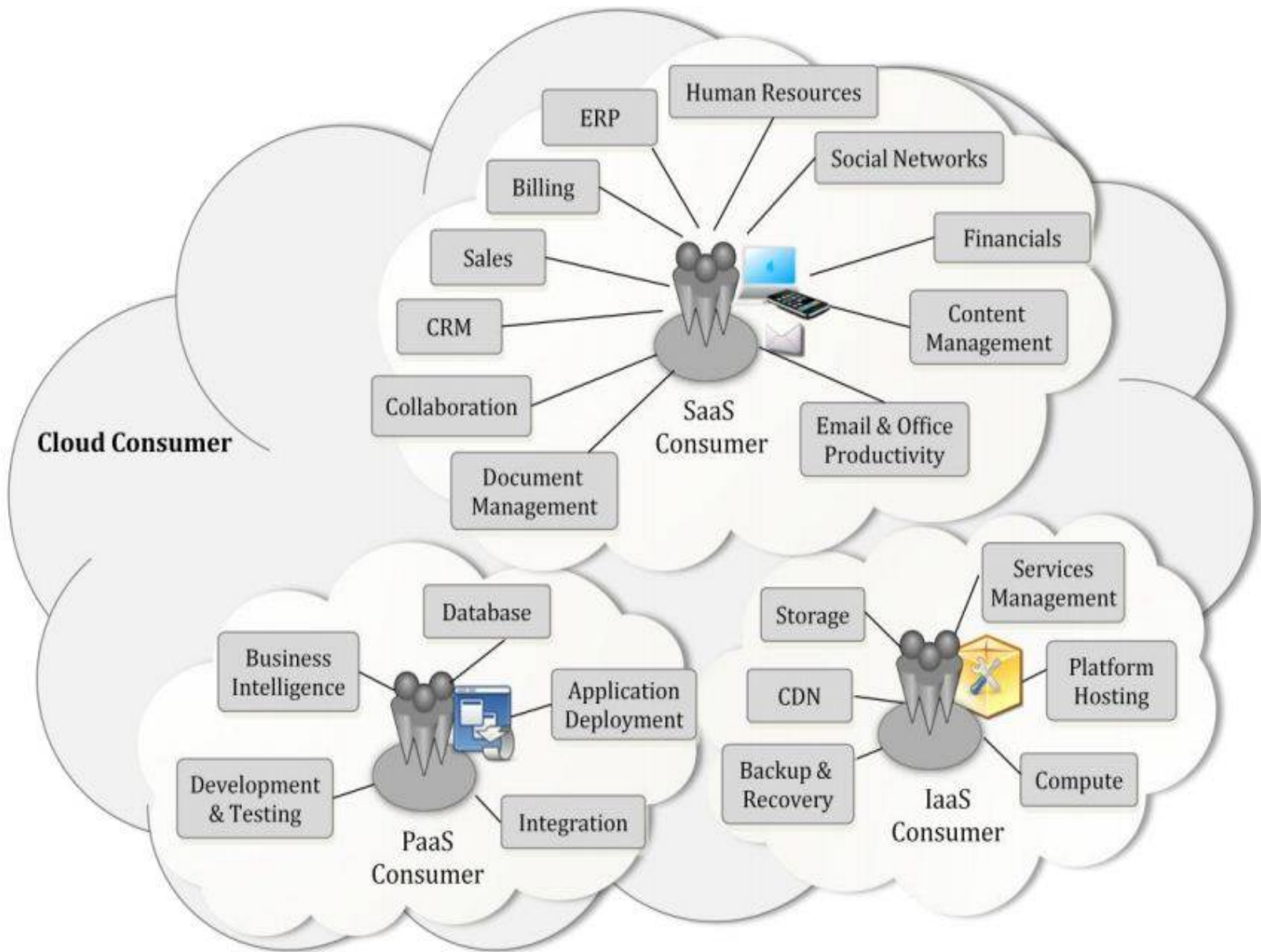


Fig: Cloud Services overview and Applications

Benefits and Limitations of Cloud Computing

- ✓ Reducing Hardware Cost
- ✓ Reducing Software Cost
- ✓ Reduced implementation and maintenance costs.
- ✓ Increased Mobility for a global workforce.
- ✓ Flexible and scalable infrastructures.
- ✓ Quick-time-to-market.
- ✓ IT department transformation (Innovation Vs Implem'n)
- ✓ Greening of the data centers.
- ✓ Increasing availability of high-performance applications to small/medium size businesses.



Fig: Cloud Computing Benefits

Limitations of Cloud Computing

- ✓ Secured Data Management
- ✓ Privacy and Reliability
- ✓ Performance
- ✓ Availability
- ✓ Ownership Rights
- ✓ High speed access to -
internet and standardiza



Advantages of Cloud Computing

1. **Reduce on infrastructure:** Maintain easy access to your information with minimal upfront spending. Pay as you go (weekly, quarterly or yearly), based on demand.
2. **Globalize your workforce for low:** People worldwide can access the cloud, provided they have an Internet connection.
3. **Streamline processes:** Get more work done in less time with less people.
4. **Reduce capital costs:** There's no need to spend big money on hardware, software or licensing fees.
5. **Improve accessibility:** You have access anytime, anywhere, making your life so much easier!
6. **Monitor projects more effectively:** Stay within budget and ahead of completion cycle times.
7. **Less personnel training is needed:** It takes fewer people to do more work on a cloud, with a minimal learning curve on hardware and software issues.
8. **Minimize licensing new software:** Stretch and grow without the need to buy expensive software licenses or programs.
9. **Improved flexibility:** Provides users with more flexible options in choosing the services.

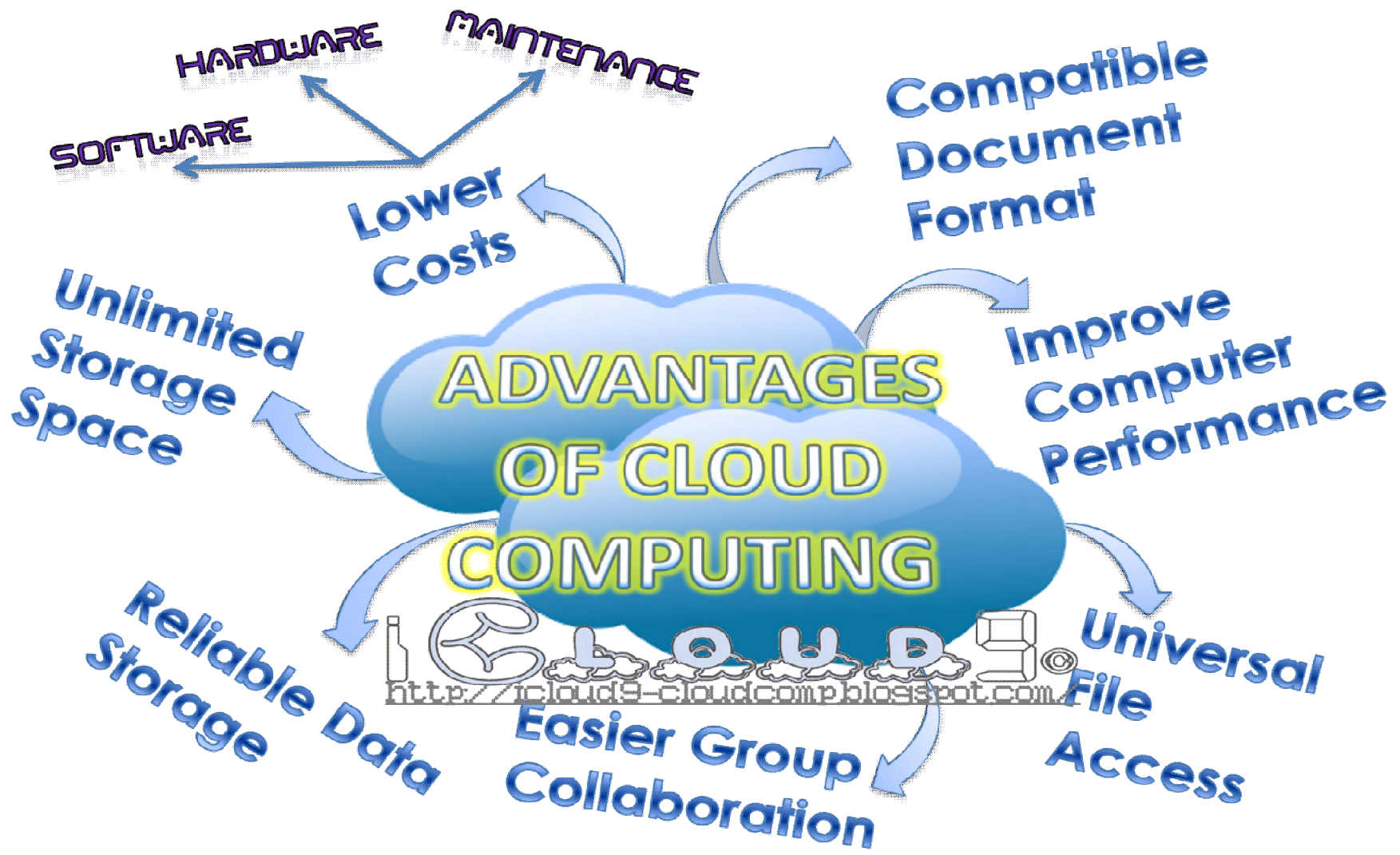


Fig: Advantages of Cloud Computing

Disadvantages of Cloud Computing

- 1. Performance:** One of the major problems with the cloud computing technique is that the application performance may suffer.
- 2. Security:** Cloud hosting does not provide you as much secure environment as you need.
- 3. Redundancy:** One of the misconceptions of cloud hosting is that it's hosted "in the sky and not in a datacenter," which is not true.
- 4. Cost:** One of the misconceptions of cloud hosting is that it is cost effective which is partially true.
- 5. Sometimes frustrating:** Exploration of next-generation IT models requires a thrill-seeking strength of mind and technical perspicacity.



Fig: Disadvantages of Cloud Computing

Applications

- ✓ Web Apps (Face book)
- ✓ SaaS (Google Apps, Salesforce)
- ✓ Software + Services (MS Online Services)
- ✓ P2P (Skype)
- ✓ Backup and Recovery
- ✓ Organizational Applications
- ✓ Distributed data computing
- ✓ Online Banking
- ✓ Online Gaming
- ✓ E-commerce
- ✓ Financial organizations
- ✓ Health care sectors
- ✓ Education
- ✓ Agriculture
- ✓ Government



6. CLOUD INFRASTRUCTURAL MODELS

- ✓ Cloud infrastructure model is also called as cloud **Deployment Model**.
- ✓ Cloud infrastructure model is also called **Types of Clouds**.
- ✓ Cloud **Infrastructure Model** can be implemented through infrastructure as a service to platform as a service.

✓ Four fundamental elements of Commercial Cloud

✓ Computing.

✓ Storage.

✓ Database.

✓ Networking

✓ E.g. Amazon offers: EC2, S3, SQS, Cloudfront, SDB

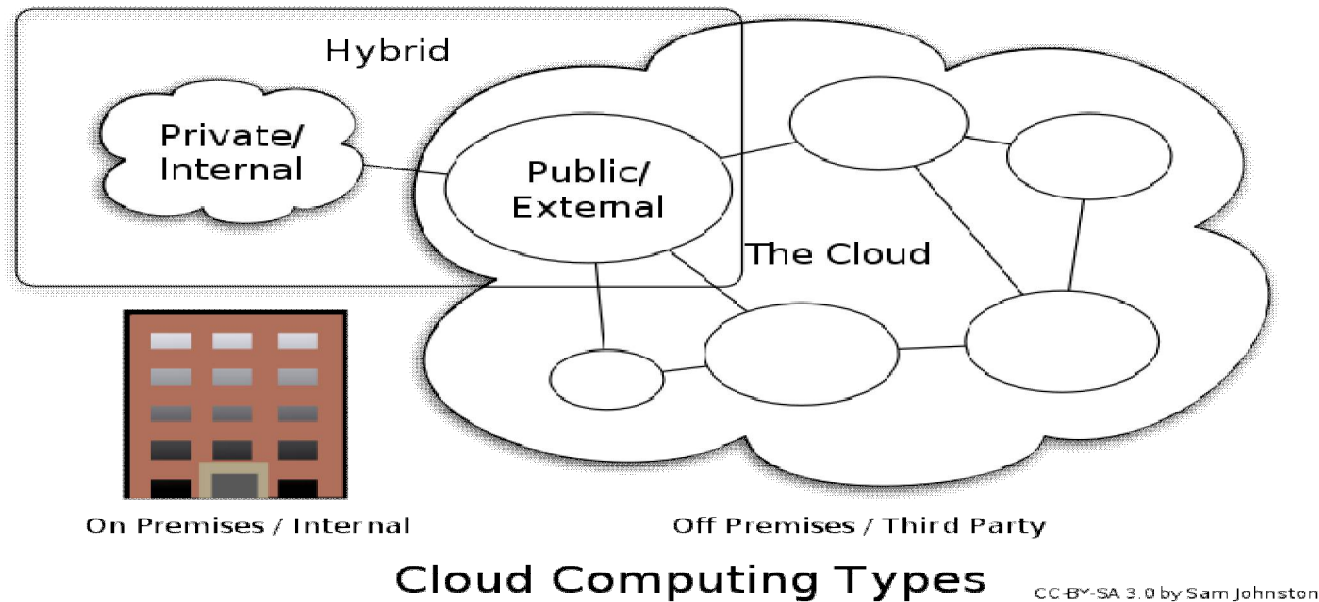
✓ EC 2

✓ Simple storage service

✓ SDB is DB as a service

✓ SQS for MPI among M/Cs

✓ Cloudfront for content delivery



❖ Cloud Computing deployment Models are:

1. Public Cloud
2. Private Cloud
3. Hybrid Cloud
4. Community Cloud

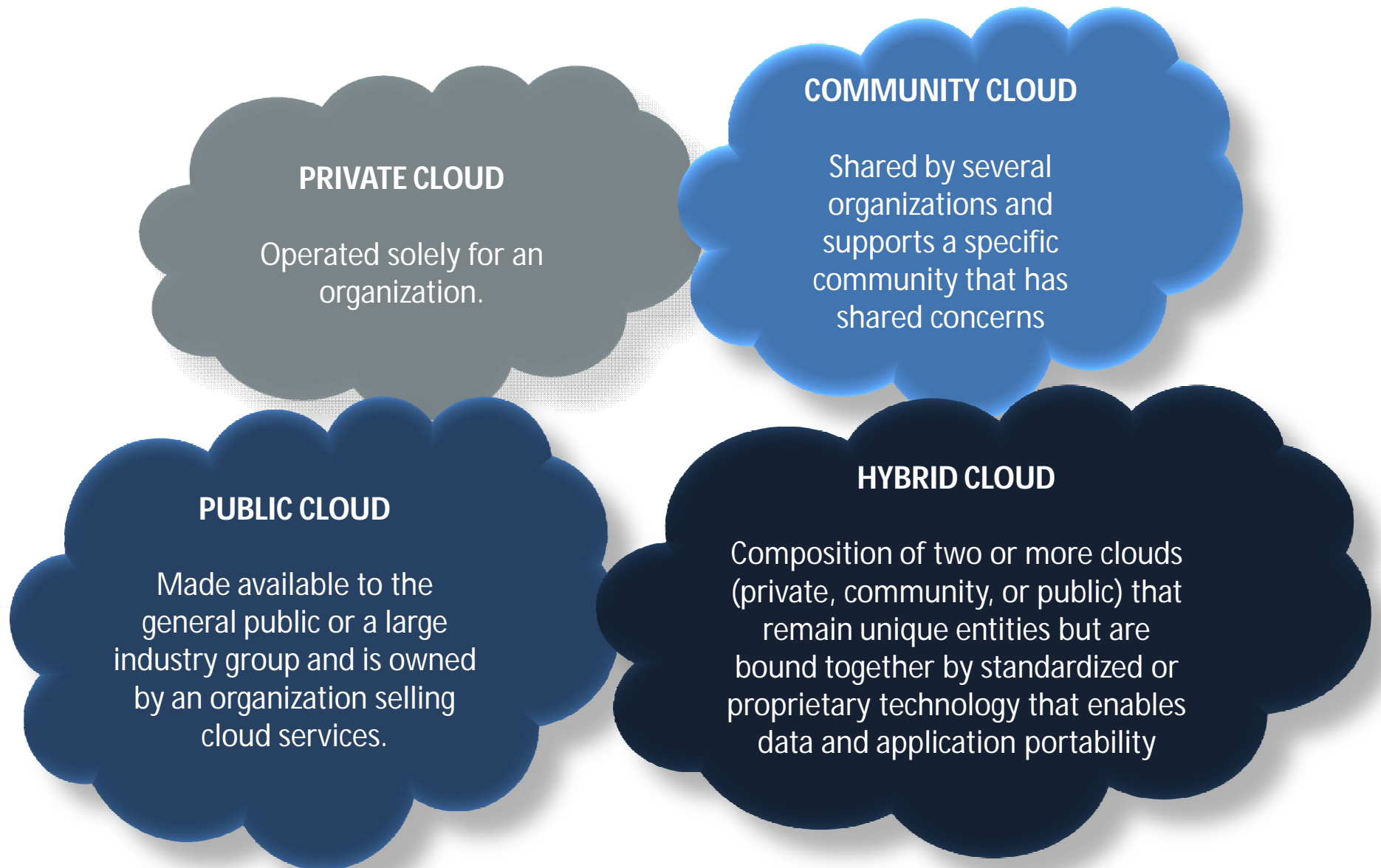
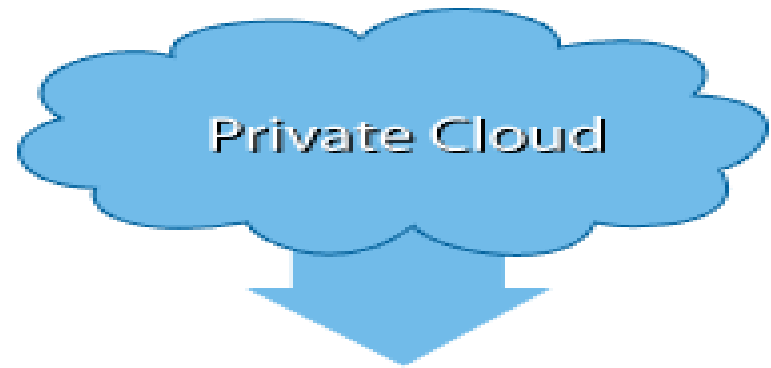


Fig: Deployment Model Overview

- 1. Private Cloud:** Private Cloud is also known as **Internal Cloud and Enterprise owned or leased.**
 - ✓ Private Cloud is infrastructure **operated solely for a single organization**, whether managed internally or by a third party and hosted internally or externally.
 - ✓ **Ex:** MS-Azure, Salesforce, Amazon S3.

- 2. Public Cloud:** Public cloud is also known as **External Cloud and Sold to the public, mega-scale infrastructure.**
 - ✓ Public cloud describes cloud computing in the traditional mainstream sense, whereby resources are dynamically provisioned to the general public on a fine-grained.
 - ✓ Ex: Gmail, Google docs, Drop box, Google Drive etc;



1. Made available to the general public or a large industry group and is owned by an organization selling cloud services.
2. Hosted at a Service provider site.
3. Supports multiple customers.
4. Often utilizes shared Infrastructure.
5. Supports connectivity over the Internet
6. Suited for Information i.e. not sensitive.
7. It can be cheaper than private cloud
8. Ex: Gmail, Google docs, Drop box, Google Drive, BSNL Portal Cloud, IRCTC Cloud etc.

1. Operated solely for an organization.
2. Hosted at a Service provider site.
3. Supports one customer.
4. Does not utilizes shared Infrastructure.
5. Connectivity over private Network or fiber or the Internet
6. Suited for Information that requires a high level security.
7. It can be higher than public cloud.
8. Ex: MS-Azure, Salesforce, Amazon [EC2](#) or [S3](#)

3. Hybrid Cloud: Hybrid cloud is a Composition of two or more clouds.

✓ Hybrid cloud is a composition of two or more clouds (private, public or community) that remain unique entities but are bound together, offering the benefits of multiple deployment models.

✓ Hybrid cloud increases flexibility of computing.

✓ **Ex:** MS-Azure, Salesforce, IBM Blue cloud.

4. Community Cloud: Community Cloud is also known as External Cloud and Shared infrastructure for specific community.

- ✓ Community cloud shares infrastructure between several organizations from a specific community with common concerns (security, compliance, jurisdiction, etc.), whether managed internally or by a third-party and hosted internally or externally.
- ✓ **Ex:** Health cloud, E-Gov, Social networks.

- 5. Inter Cloud:** The Inter Cloud is an interconnected global "Cloud of clouds" and an extension of the Internet "Network of networks" on which it is based. Inter Cloud is a Scaling of Applications across multiple Cloud Computing Environments.
- 6. Virtual Private Cloud (VPC):** An IT solution platform that integrates local and commercial cloud infrastructure under a single management.
- 7. Regional Clouds:** Geographical Cloud Computing - Response based scaling on a regional basis and Regional Scaling and Monitoring.
- 8. Offline Clouds:** What happens when your lose connectivity and What happens when your infrastructure goes down.

7. Cloud Computing Protocols

- ✓ Among a handful of application protocols (HTTP, FTP, etc.), would there be a very few protocols that will see much wider usage as cloud computing becomes more and more popular.
- ✓ With the advent of cloud computing, networking protocols such as TCP, UDP, SNMP, etc are more relevant than ever.
- ✓ Cloud computing works because of networking. Without these networking protocols, we would not have "cloud."
- ✓ **It consists of common standards and protocols:**
 - i) **Common standards:**
 1. The Open Cloud Consortium(OCC)
 2. The Distributed Management Task Force(DMTF)
 3. Open Virtualization format(OVF)
 - ii) **Common protocols in cloud:**
 1. Standards for Messaging and Communications (SMTP, POP, IMAP, Atom Publishing Protocol, HTTP, SIMPLE, XMPP)
 2. Standards for Application Developers (Browser, Data, Solution stacks)
 3. Standards for Security(SAML, OAuth, Open ID, SSL/TLS)

i) Common standards are:

1. **OCC(open cloud infrastructure)**: Its an standard organization where many universities and it sectors collaboratively work to frame the interoperable standards. **Working Groups**: Terra flow test bed, open science data cloud(OSDC).
2. **DMTF(Distributed management task force)**: It provides standards to develop, test the systems. **Working groups**: VMAN(virtualization management Initiative)
3. **OVF(Open Virtualization Format)**: Simplifies interoperability, security and virtual m/c life cycle management by describing open, portable, efficient format for packaging and distribution of one/more virtual appliances. Advantages are simplified installation and deployment process and VM packaging portability.

ii) Common Protocols for cloud computing:

1. **Standards for Messaging and Communication:** These are protocols: SMTP, FTP, POP, IMAP((Internet Message Access Protocol), Atom Publishing Protocol, RSS(Really simple syndication), HTTP, XMPP (Extensible Messaging Presence Protocol), SOAP(Simple Object Access Protocol)
2. **Standards for Application Developers:** Browser(AJAX), Data (XML, JSON-Java script object notation), Solution stacks (LAMP-Linux Apache My SQL PHP, LAPP- Linux Apache Postgre SQL PHP)
3. **Standards for Security** uses following protocols: (SAML(Security Assertion Markup Language) OAuth, Open ID, SSL/TLS(Transport layer security)

8. CLOUD COMPUTING ARCHITECTURE: Cloud computing architecture refers to the components and subcomponents required for cloud computing, these components typically consist of a front end platform i.e. fat client, thin client, mobile device, back end platforms i.e. servers, storage, a cloud based delivery, and a network i.e. Internet, Intranet, Inter-cloud, combined, these components are called as **Cloud Computing Architecture**.

- ✓ Cloud Computing Architecture is a structure, which comprises the software and hardware components integrated each other is called **Cloud Computing Architecture**.
- ✓ A set of structures of a system with software elements and relations, properties among them is called as **Cloud Computing Architecture**.
- ✓ It is the responsibility of the back end to provide built-in security mechanism, traffic control and protocols.

- ✓ The server employs certain protocols, known as middleware, helps the connected devices to communicate with each other.
- ✓ The Cloud Computing architecture comprises of many cloud components, each of them are loosely coupled, it can broadly divide the cloud architecture into two parts:
 - ✓ **i) Front End:** It refers to the client part of cloud computing system. It consists of interfaces and applications that are required to access the cloud computing platforms, **E.g.,** Web Browser.
 - ✓ **ii) Back End:** It refers to the cloud itself. It consists of all the resources required to provide cloud computing services. It comprises of huge **data storage, virtual machines, security mechanism, services, deployment models, servers, etc.**

✓ Cloud Computing Architecture requirements are

1. User Requirements
2. Enterprise Requirements
3. Provider Requirements

1) User Requirements:

✓ These are

- ✓ User consumption based billing and metering
- ✓ User centric privacy
- ✓ SLA
- ✓ Adaptability and Learning
- ✓ User experience

2) Enterprise Requirements:

✓ These are

- ✓ Cloud Deployment
- ✓ Security
- ✓ Cloudeconomics
- ✓ Data Governance
- ✓ Data Migration

- ✓ Business Process Management
- ✓ Third Party Engagement
- ✓ Transferable Skills

3) Provider Requirements:

- ✓ These are
 - ✓ Service Delivery Models
 - ✓ Service centric Issues
 - ✓ Fault Tolerance
 - ✓ Data Management, Storage and Processing
 - ✓ Virtualization Management
 - ✓ Load Balancing

- ✓ Object of **NIST** reference arch is to illustrate and understand various cloud services in the context of an overall cloud computing conceptual model.
- ✓ There are majorly divided into **Cloud Computing Architecture** is
 1. Cloud Consumer
 2. Cloud Provider
 3. Cloud Auditor
 4. Cloud Carrier
 5. Cloud Broker

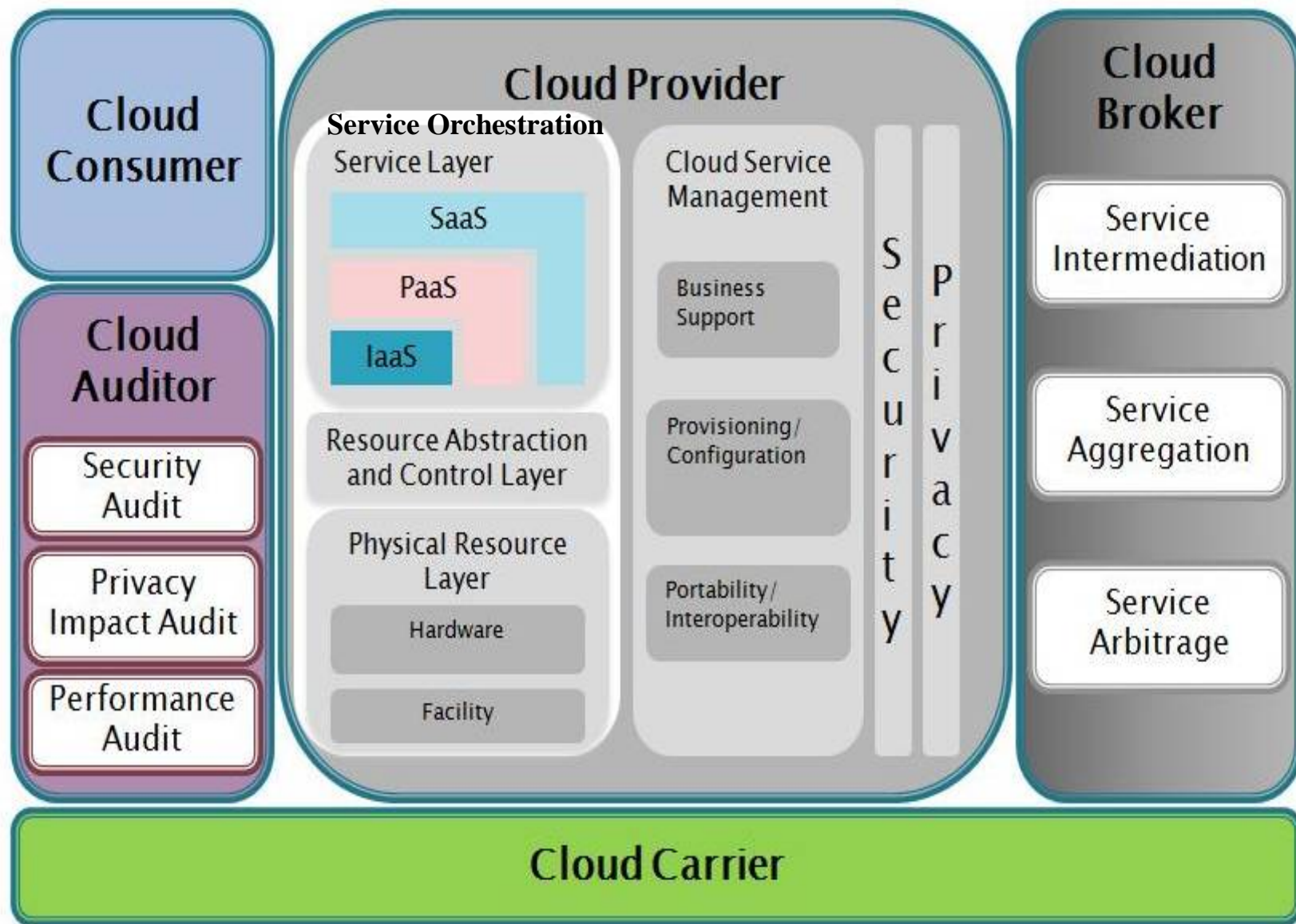


Fig: Cloud Computing Architecture (NIST)

- 1. Cloud Consumer:** Person or organization that maintains a business relationship with, and uses service from, Cloud Service Providers.
- 2. Cloud Provider:** Person, organization or entity responsible for making a service available to service consumers.
- 3. Cloud Carrier:** The intermediary that provides connectivity and transport of cloud services between Cloud Providers and Cloud Consumers.
- 4. Cloud Broker:** An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers.
- 5. Cloud Auditor:** A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.

- ✓ **Cloud Application Architectures** will help you determine whether and how to put your applications into these virtualized services, with critical guidance on issues of cost, availability, performance, scaling, privacy, and security.
- ✓ **Cloud Application Architecture consists of**
 - a) Fundamental Cloud Architectures
 - b) Cloud Application Components
 - c) Multi-Tenancy
 - d) Cloud Integration
 - e) Grid Computing
 - f) Transactional Computing
 - g) On-Demand Computing
 - h) Distributed Computing
 - i) Utility Computing
 - j) Cluster Computing

- a) **Fundamental Cloud Architectures:** It refers to the **components** and **subcomponents** required for cloud computing, these components typically consist of a front end platform (**fat client, thin client, mobile device**), back end platforms (**servers, storage**), a cloud based delivery, and a network (**Internet, Intranet, Intercloud**), combined, these components make up cloud computing architecture.
- b) **Cloud Application Components:** These are **clients, services, application, platform, storage and infrastructure**.
- c) **Multi-Tenancy:** It is an architecture in which a single instance of a software application serves **multiple customers**.
- d) **Cloud Integration:** It is the process of configuring **multiple application programs** to share data in the cloud.
- e) **Grid Computing:** It is the collection of computer resources from multiple locations to reach a **common goal** and grid can be thought of as a **distributed system** with non-interactive workloads that involve a large number of files.

- f) **Transactional Computing:** It is information processing that is divided into individual, indivisible operations called transactions. each transaction must **succeed** or **fail** as a complete unit; it can never be only partially complete.
- g) **On-Demand Computing:** It is an increasingly popular **enterprise model** in which computing resources are made available to the user as needed.
- h) **Distributed Computing:** It is a model in which **components located on networked computers communicate** and coordinate their actions by passing messages, the components interact with each other in order to achieve a common goal.
- i) **Utility Computing:** It is a service **provisioning model** in which a service provider makes computing resources and infrastructure management available to the customer as needed, and charges them for specific usage rather than a flat rate.
- j) **Cluster Computing:** It consists of a set of **loosely** or **tightly** connected computers that work together so that, in many respects, they can be viewed as a single system.

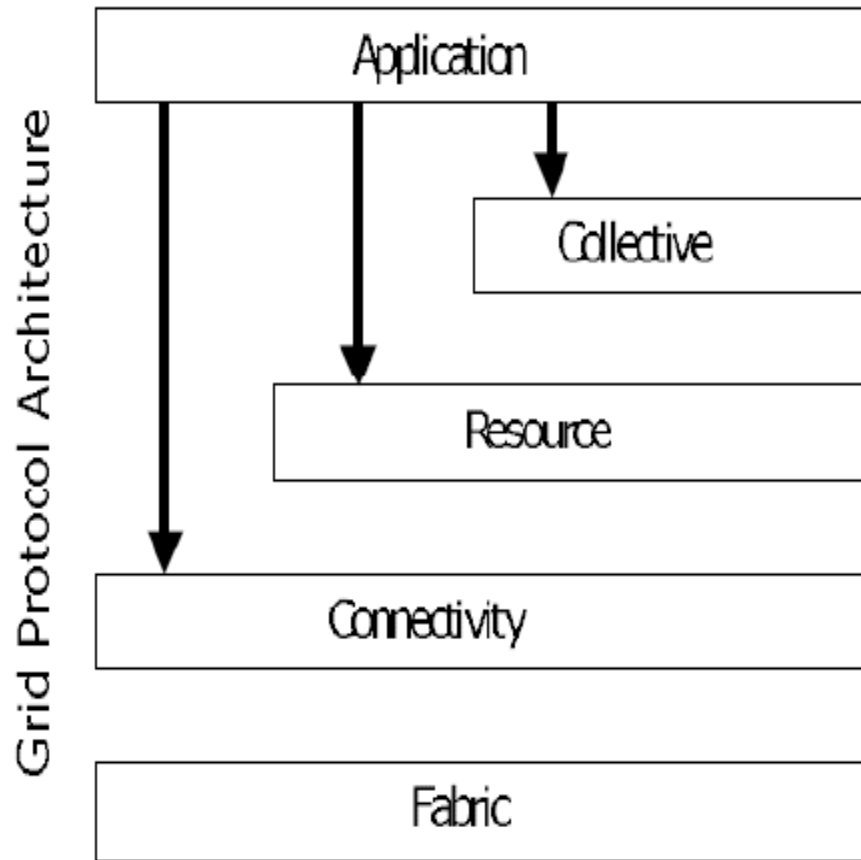
GRID COMPUTING

VS.

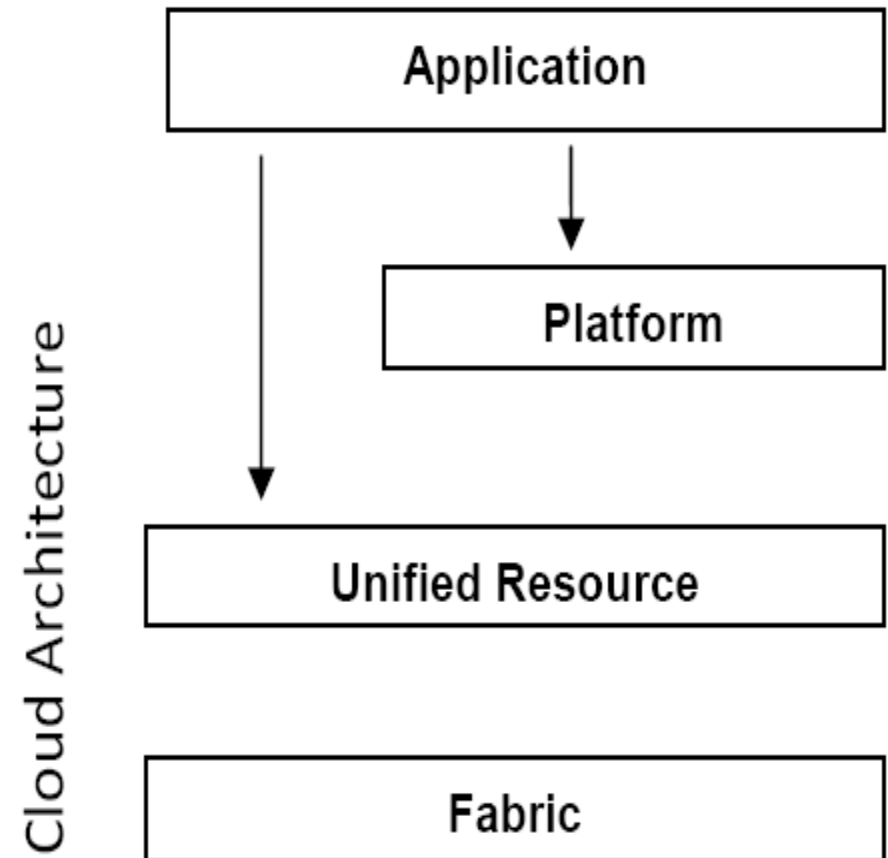
CLOUD COMPUTING

1. The concept of **grids** was proposed by **Ian Foster** in the 1970's.
 2. Grids enable access to **shared** computing power and **storage capacity** from your desktop
 3. Governments - Providers and users are usually publicly funded research organizations, **Ex:** National Grid Initiatives.
 4. In computing centers **distributed** across different sites, countries and continents.
 5. Grids are an **open source** technology.
 6. **Benefits:** Collaboration, transparency, Resilience, ownership.
 7. **Drawbacks:** Reliability, complexity, commercial.
 8. Architecture is service oriented
1. The concept of Cloud Computing was proposed by J.C.R.Licklider and John McCarthy in the 1960's.
 2. Clouds enable access to **leased computing power** and storage capacity from your desktop
 3. The cloud provider pays for the computing resources; the user pays to use them. **Ex:** Amazon, Google
 4. The cloud providers private data centers which are often **centralized** in a few locations with excellent network connections and cheap electrical power.
 5. Clouds are a **proprietary** technology.
 6. **Benefits:** Flexibility, Reliability, Ease of use.
 7. **Drawbacks:** Security, opacity, rigidity, provider lock in.
 8. Architecture is user chosen

9. Grid Architecture



9. Cloud Architecture



DISTRIBUTED COMPUTING

VS.

CLOUD COMPUTING

1. The concept of **grids** was proposed by **Ian Foster** in the 1970's.
 2. Grids enable access to **shared** computing power and **storage capacity** from your desktop
 3. Governments - Providers and users are usually publicly funded research organizations, **Ex:** National Grid Initiatives.
 4. In computing centers **distributed** across different sites, countries and continents.
 5. Grids are an **open source** technology.
 6. **Benefits:** Collaboration, transparency, Resilience, ownership.
 7. **Drawbacks:** Reliability, complexity, commercial.
 8. Architecture is service oriented
1. The concept of Cloud Computing was proposed by J.C.R.Licklider and John McCarthy in the 1960's.
 2. Clouds enable access to **leased computing power** and storage capacity from your desktop
 3. The cloud provider pays for the computing resources; the user pays to use them. **Ex:** Amazon, Google
 4. The cloud providers private data centers which are often **centralized** in a few locations with excellent network connections and cheap electrical power.
 5. Clouds are a **proprietary** technology.
 6. **Benefits:** Flexibility, Reliability, Ease of use.
 7. **Drawbacks:** Security, opacity, rigidity, provider lock in.
 8. Architecture is user chosen

9) On-Demand Services

- ✓ On-demand self-service makes IT resource capacity within a cloud infrastructure appear infinite to users.
- ✓ A consumer can unilaterally provision computing capabilities, such as **server time** and **network storage**.
- ✓ Any resources cloud end users get as a service on **leased basis** demanding cloud vendor to fulfill their business needs.
- ✓ Customers can automatically provisioned computing capabilities and resources on their own when needed without much human intervention.
- ✓ On-demand self service is also related to utility computing and the **pay-as-you-go subscription** method.

Examples:

- ✓ Video on demand.
- ✓ Hardware on demand.
- ✓ Application on demand.
- ✓ Software on demand.
- ✓ Storage on demand.
- ✓ Computing on demand.

- ✓ A consumer can unilaterally provision computing capabilities such as server time and network storage as needed automatically without requiring human interaction with each service provider
- ✓ Cloud computing enables end users to provision **computing power, storage, networks** and **software** in a simple and flexible way.
- ✓ It consists of
 1. Application Architecture of On-Demand Service in Cloud Computing .
 2. Distributed Management and Resource Status Monitoring Technologies .
 3. Modeling Technology of On-Demand Services with Context Awareness .
 4. Automatic Service Composition Technology on Demand .
 5. On-Demand Service Based on Complex Adaptive System.

QUIZ - I

1. Write few real time applications where cloud computing can be used?

Ans) Drop Box, Gmail, Facebook

2. Name the Early contributor for cloud computing in the 1960s developing ARPANET project?

Ans) J.C.K Licklider and John McCarthy

3. Name the first person to use cloud term in India: _____

Ans) Ramanathan Chellappa

4. Choose the appropriate top cloud service provider that provides computing as a service?

a) Amazon EC2 b) MS-Azure c) Salesforce.com d) All

5. Identify the next trend after cloud technology?

a) Cloud computing b) IOT and Big Data c) both a) & b) d) All

6. Illustrate four important features of platform virtualization?

Ans) Parallel Processing, Vector Processing, Symmetric Multiprocessing and Massively Parallel Processing issues

7. Distinguish the Cloud vs. Internet?

Ans) Network of networks used for online communication Vs.

Cloud is considered as delivery of resources available as a service.

8. Name two popular Virtualization Technologies?

Ans) VMware, Hyper-V, Xen and Virtual Iron

9. Define Cloud Computing?

Ans) “A Model for enabling ubiquitous, convenient, on-demand network access (through Internet) to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.”

10. Illustrate with an example where Salesforce.com application can be used?

Ans) Web applications, Call center Applications

11. Define on demand service/computing?

Ans) On-demand self service is also related to utility computing and the pay-as-you-go subscription method.

12. Give examples of SaaS based applications used in daily life?

Ans) Web applications, Call center Applications

13. Categorize the major services of Cloud Computing?

Ans) SaaS, PaaS, IaaS

14. State different Cloud Computing Layers?

Ans) Client, Server

15. Tell another name of Deployment Models?

Ans) Infrastructural Models

16. Write major research issue in Cloud Computing?

Ans) Security and Availability

17. Extend or Expand SLA _____ and CSP _____

Ans) Service Level Agreement, Cloud Service Provider

18. List few Cloud Computing components?

Ans) Client, Data center, Cloud Auditor, Cloud Carrier

19. Describe 5-4-3 formulae of Cloud Computing?

Ans) 5 Characteristics, 4 Deployment Models and 3 services

20. Compare and contrast of the Public Cloud vs. Private Cloud?

Ans) Many users can access in Public Cloud, only employees of that organization access private Cloud

UNIT-V – Sorting and Hashing

Mergesort and Quicksort

Dr. M Naresh Babu

Sree Vidyanikethan Engineering College
(Autonomous)

Email: nareshmuppalaneni@gmail.com

Sorting algorithms

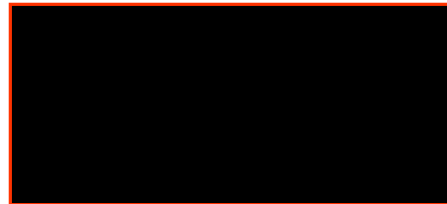
- Insertion, selection and bubble sort have quadratic worst-case performance
- The faster comparison based algorithm ?
 $O(n \log n)$
- Mergesort and Quicksort

Merge Sort

- Apply divide-and-conquer to sorting problem
- Problem: Given n elements, sort elements into non-decreasing order
- Divide-and-Conquer:
 - If $n=1$ terminate (every one-element list is already sorted)
 - If $n>1$, partition elements into two or more sub-collections; sort each; combine into a single sorted list
- How do we partition?

Partitioning - Choice 1

- First $n-1$ elements into set A, last element set B
- Sort A using this partitioning scheme recursively
 - B already sorted
- Combine A and B using method Insert() (= insertion into sorted array)
- Leads to recursive version of InsertionSort()
 - Number of comparisons: $O(n^2)$
 - Best case = $n-1$
 - Worst case =



Partitioning - Choice 2

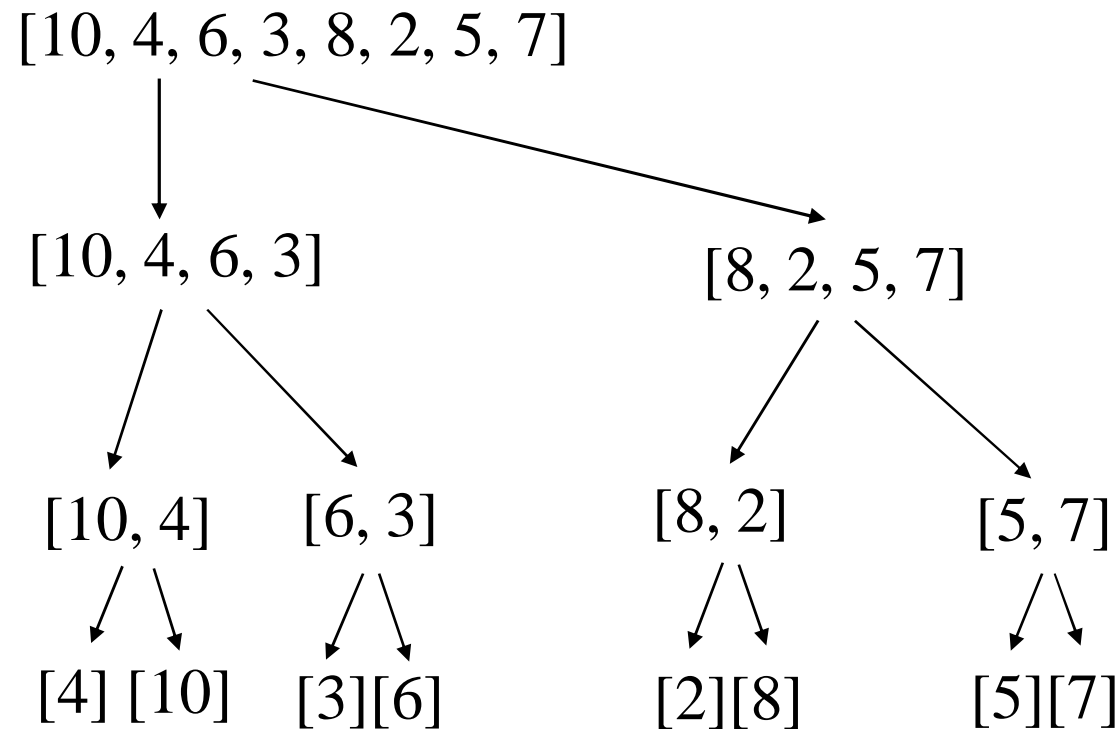
- Put element with largest key in B, remaining elements in A
- Sort A recursively
- To combine sorted A and B, append B to sorted A
 - Use Max() to find largest element → recursive SelectionSort()
 - Use bubbling process to find and move largest element to right-most position → recursive BubbleSort()
- All $O(n^2)$

Partitioning - Choice 3

- Let's try to achieve balanced partitioning
- A gets $n/2$ elements, B gets rest half
- Sort A and B recursively
- Combine sorted A and B using a process called *merge*, which combines two sorted lists into one
 - How? We will see soon

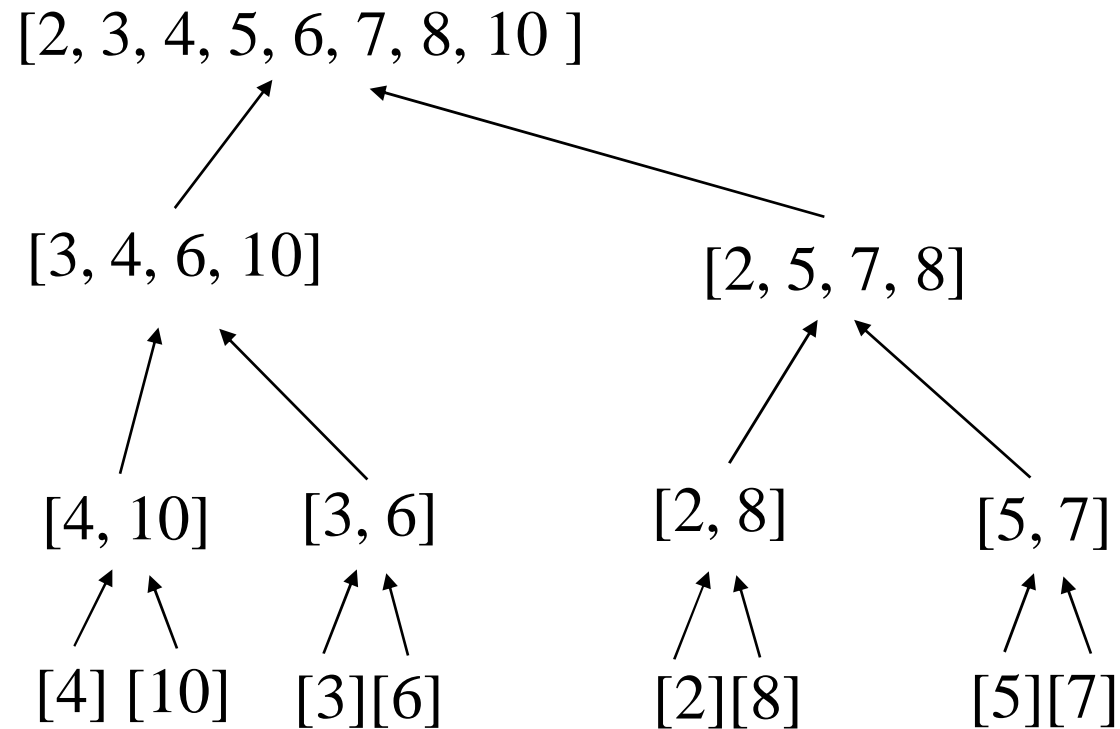
Example

- Partition into lists of size $n/2$



Example Cont'd

- Merge



Static Method mergeSort()

```
Public static void mergeSort(Comparable []a, int left,
    int right)
{
    // sort a[left:right]
    if (left < right)
    { // at least two elements
        int mid = (left+right)/2;    //midpoint
        mergeSort(a, left, mid);
        mergeSort(a, mid + 1, right);
        merge(a, b, left, mid, right); //merge from a to b
        copy(b, a, left, right);    //copy result back to a
    }
}
```

Merge Function

Evaluation

- Recurrence equation:
- Assume n is a power of 2

$$T(n) = \begin{cases} c_1 & \text{if } n=1 \\ 2T(n/2) + c_2n & \text{if } n>1, n=2^k \end{cases}$$

Solution

By Substitution:

$$T(n) = 2T(n/2) + c_2n$$

$$T(n/2) = 2T(n/4) + c_2n/2$$

$$T(n) = 4T(n/4) + 2 c_2n$$

$$T(n) = 8T(n/8) + 3 c_2n$$

$$T(n) = 2^i T(n/2^i) + i c_2n$$

Assuming $n = 2^k$, expansion halts when we get $T(1)$ on right side; this happens when $i=k$ $T(n) = 2^k T(1) + k c_2n$

Since $2^k = n$, we know $k = \log n$; since $T(1) = c_1$, we get

$$T(n) = c_1n + c_2n \log n;$$

thus an upper bound for $T(n)$ is $O(n \log n)$

Quicksort Algorithm

Given an array of n elements (e.g., integers):

- If array only contains one element, return
- Else
 - pick one element to use as *pivot*.
 - Partition elements into two sub-arrays:
 - Elements less than or equal to pivot
 - Elements greater than pivot
 - Quicksort two sub-arrays
 - Return results

Example

We are given array of n integers to sort:

40	20	10	80	60	50	7	30	100
----	----	----	----	----	----	---	----	-----

Pick Pivot Element

There are a number of ways to pick the pivot element. In this example, we will use the first element in the array:

40	20	10	80	60	50	7	30	100
----	----	----	----	----	----	---	----	-----

Partitioning Array

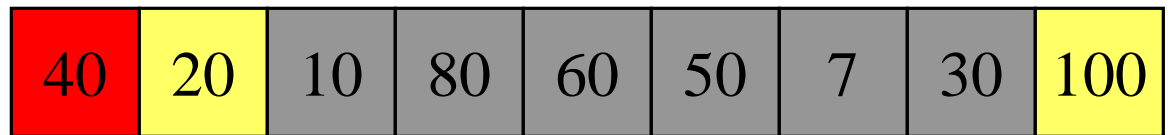
Given a pivot, partition the elements of the array such that the resulting array consists of:

1. One sub-array that contains elements \geq pivot
2. Another sub-array that contains elements $<$ pivot

The sub-arrays are stored in the original data array.

Partitioning loops through, swapping elements below/above pivot.

pivot_index = 0

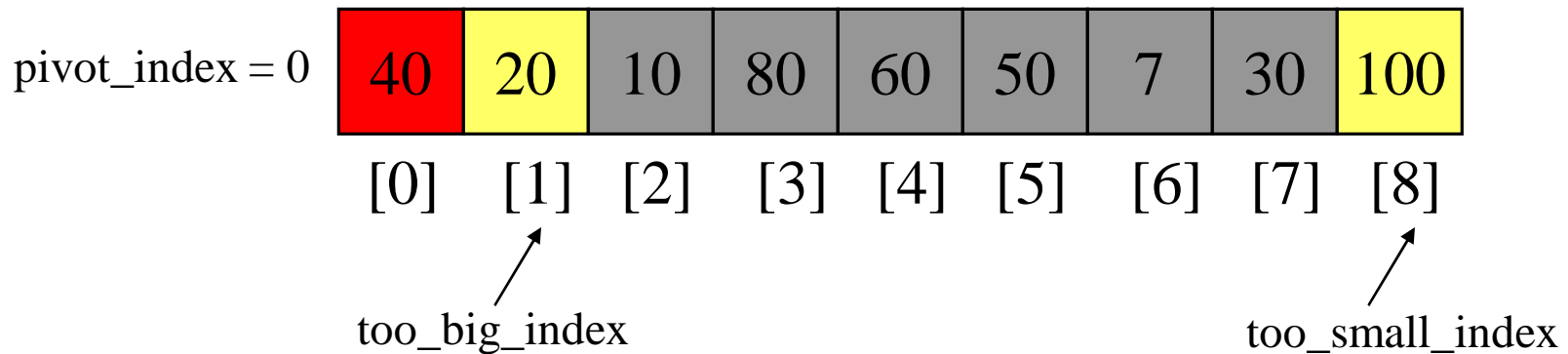


[0] [1] [2] [3] [4] [5] [6] [7] [8]

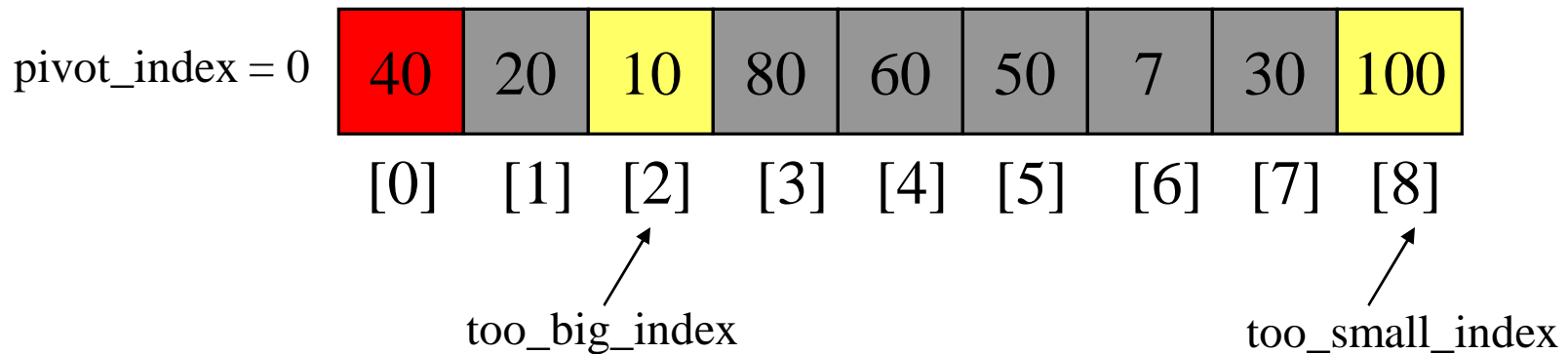
too_big_index

too_small_index

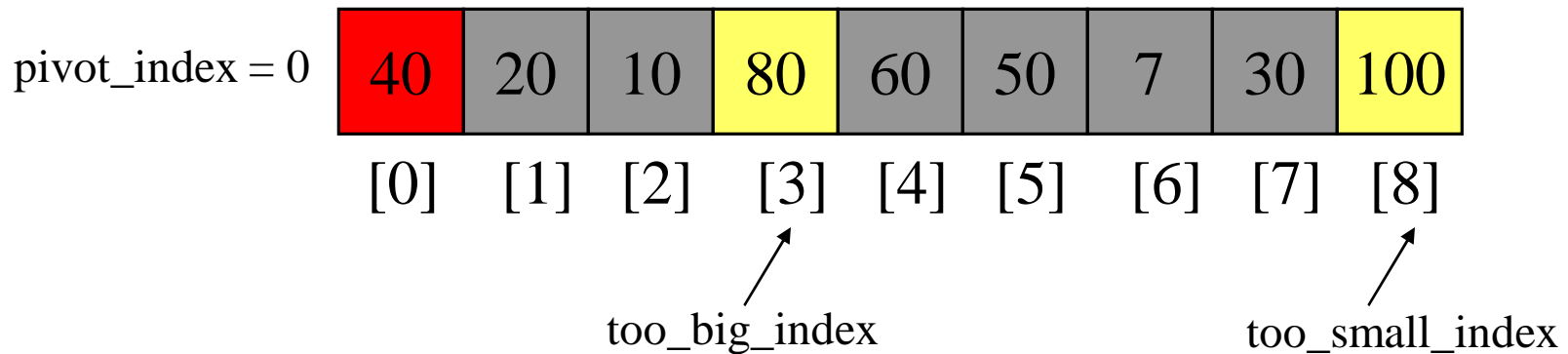
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$



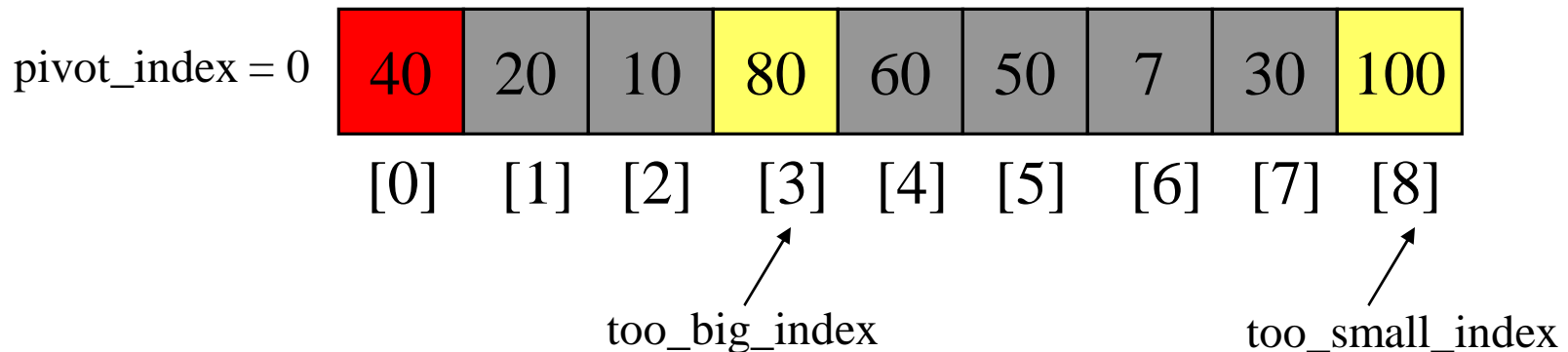
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$



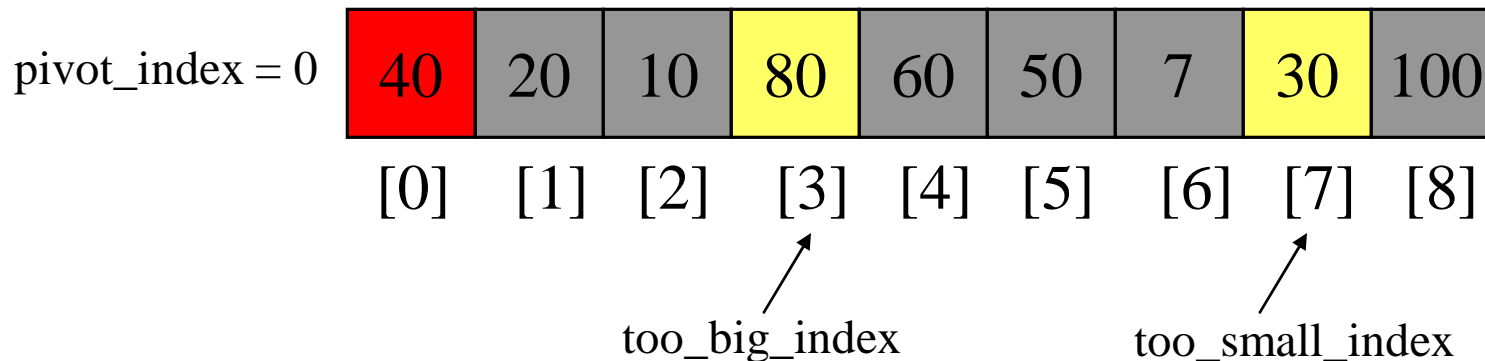
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$



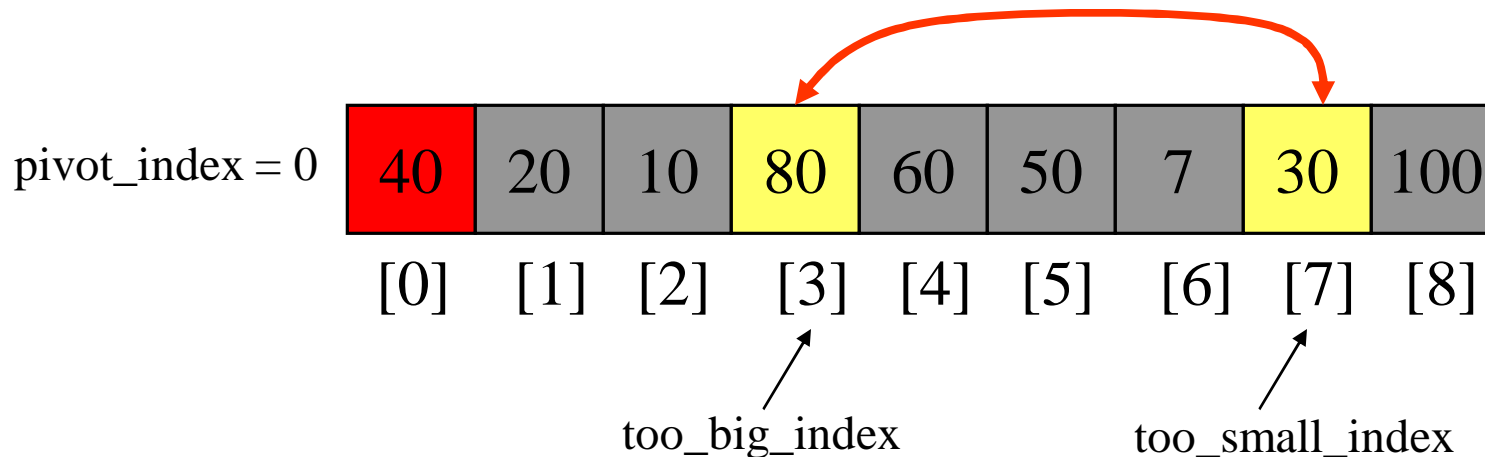
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$



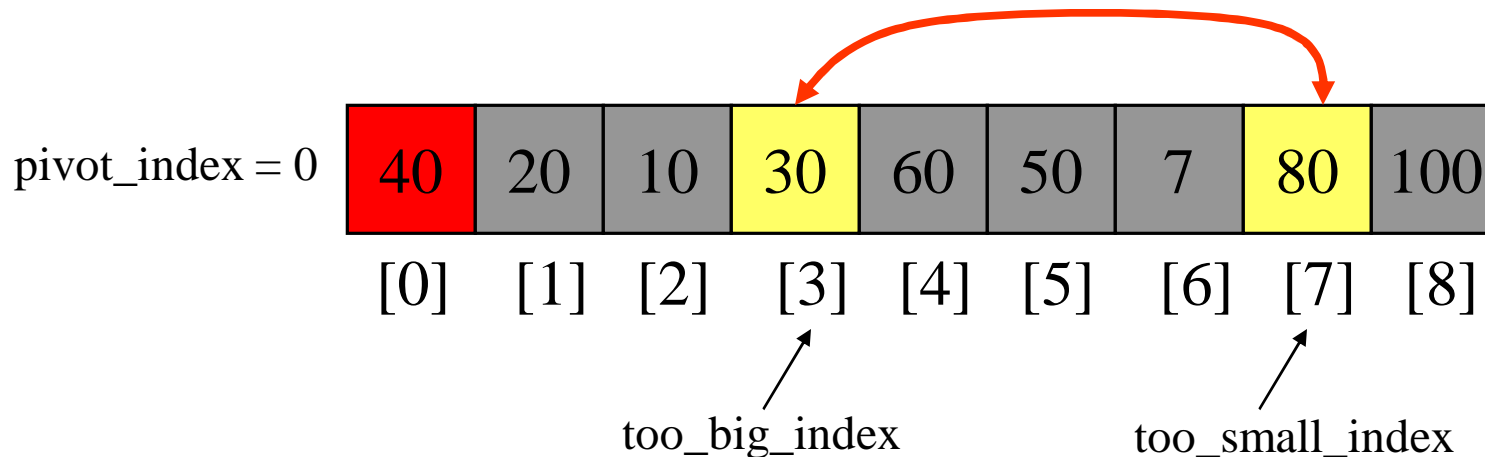
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$



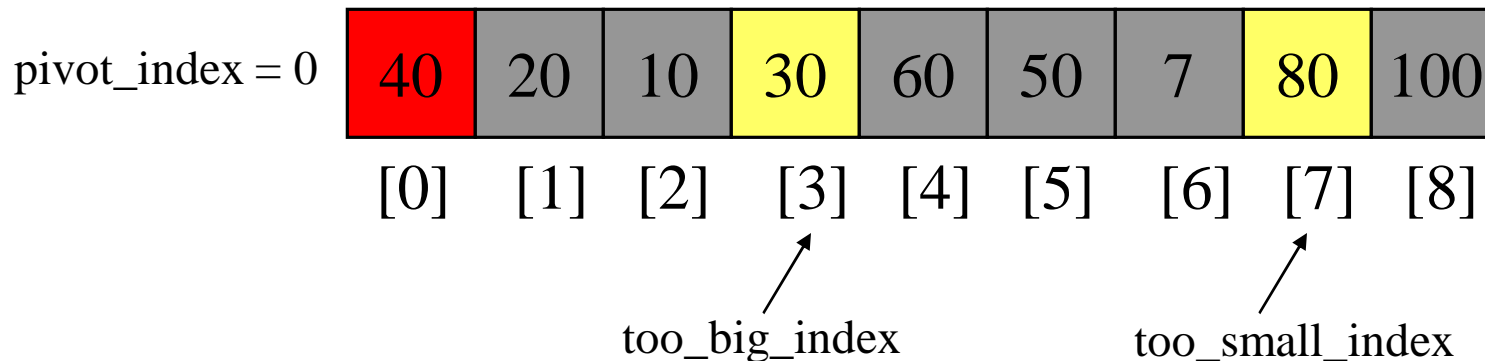
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$



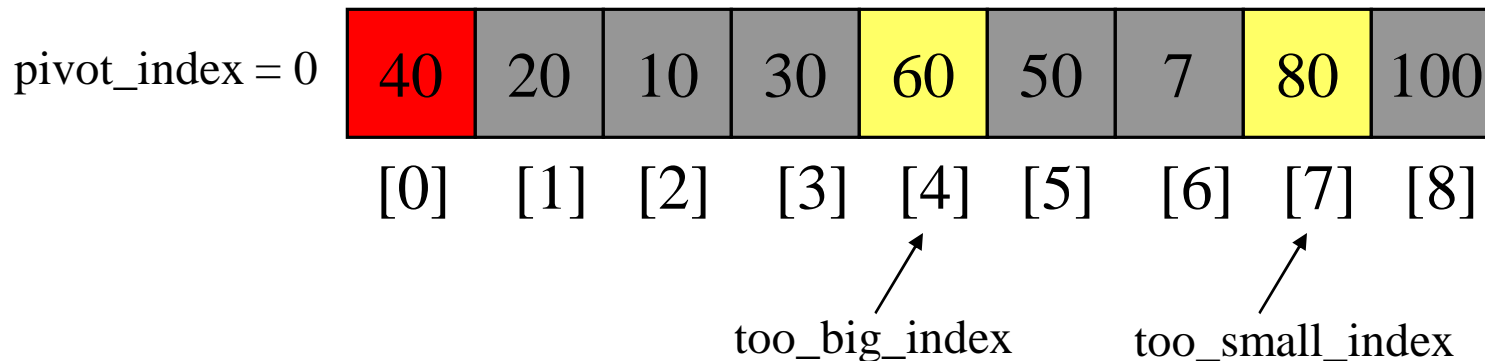
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$



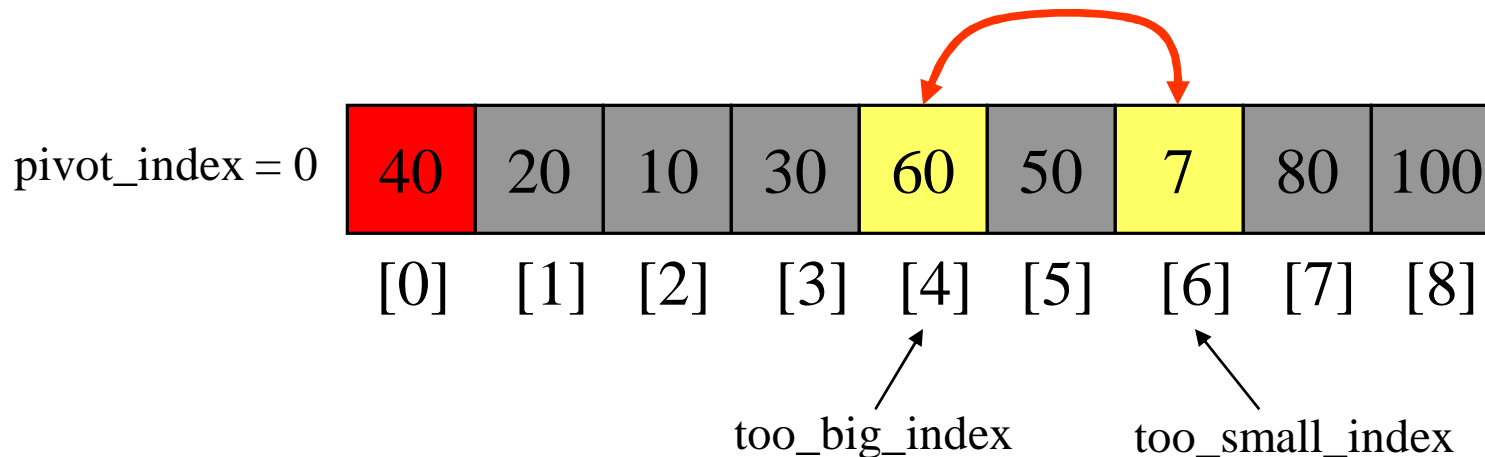
- 1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



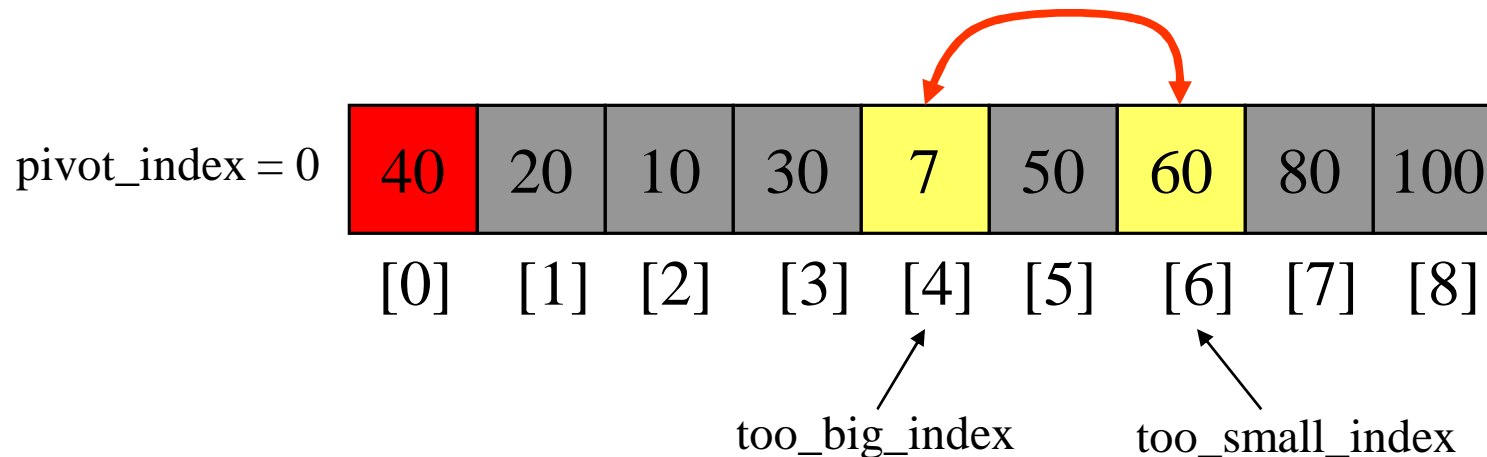
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 ++too_big_index
- 2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 --too_small_index
3. If $\text{too_big_index} < \text{too_small_index}$
swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



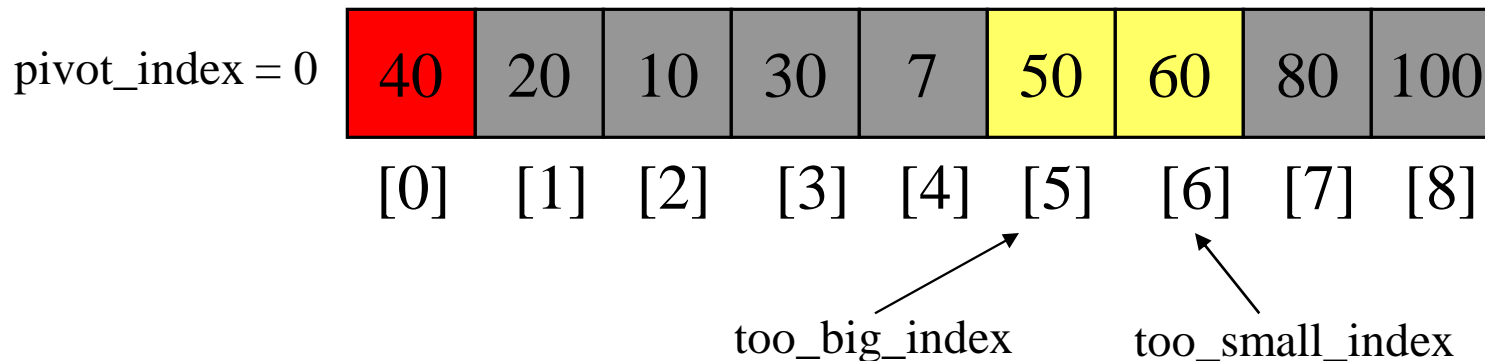
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
- 3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



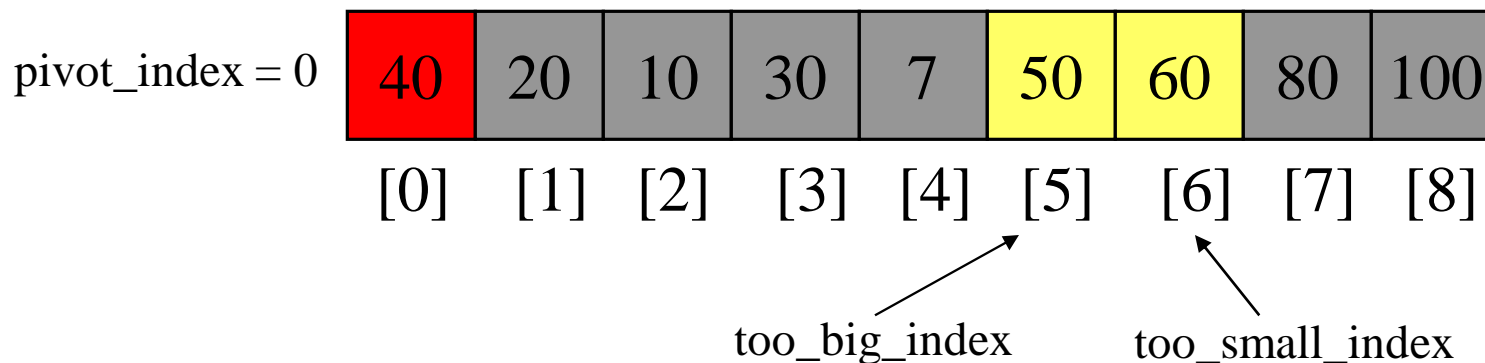
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
- 3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



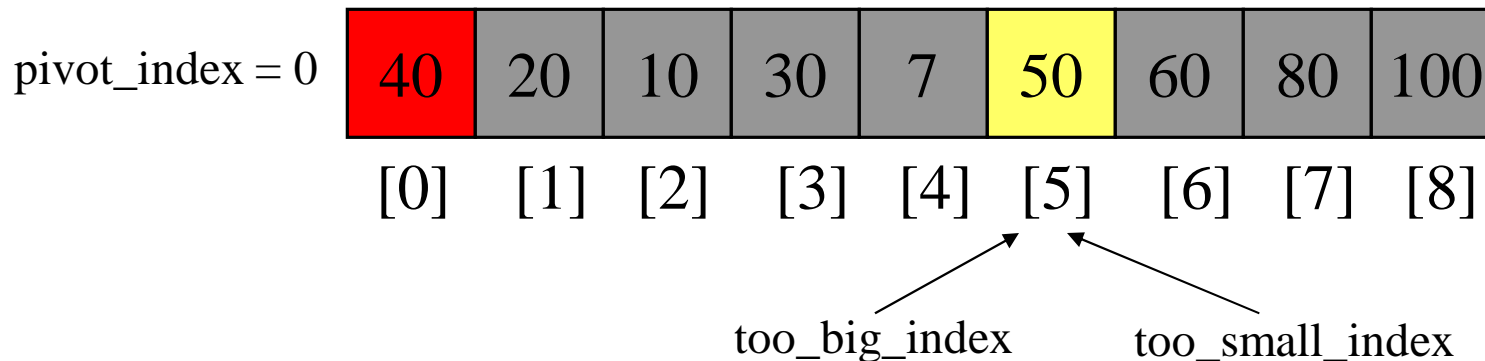
- 1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



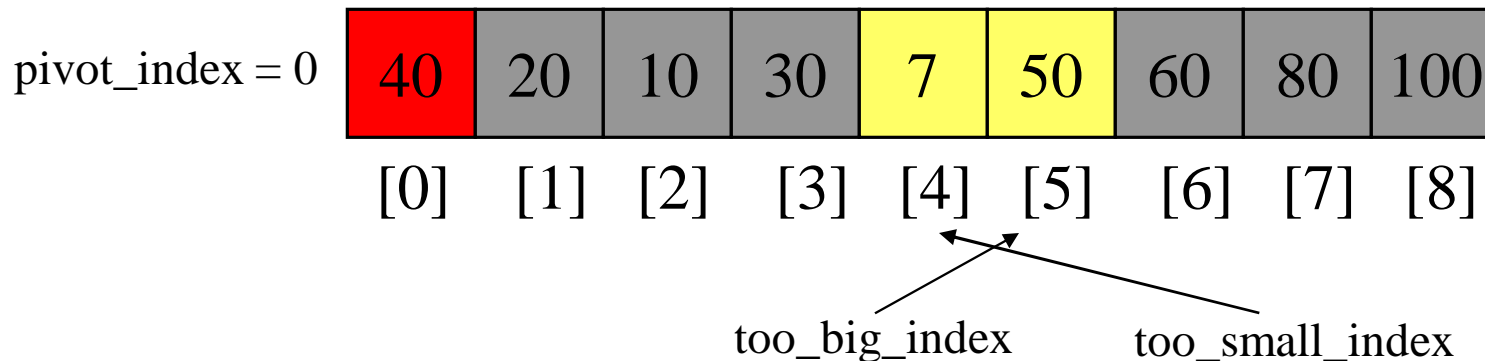
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
- 2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



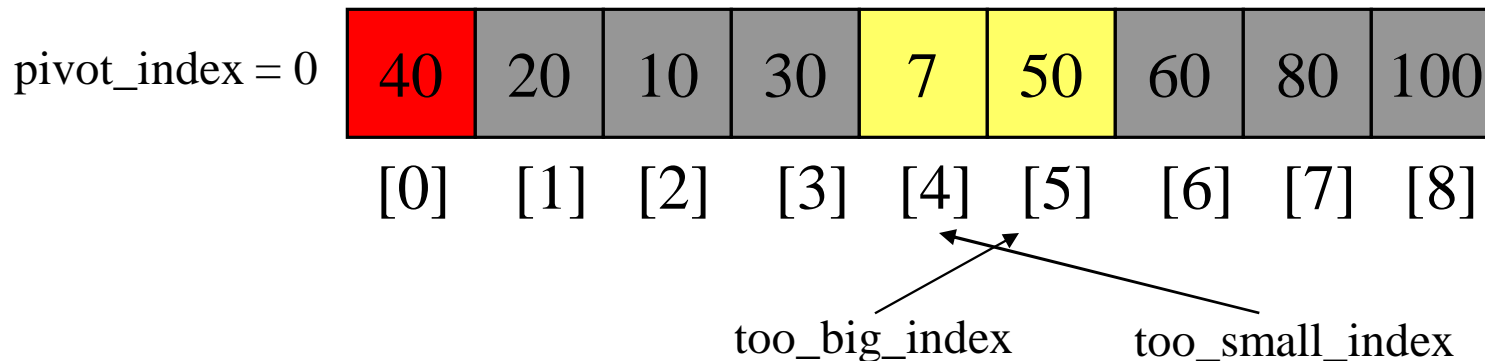
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
- 2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



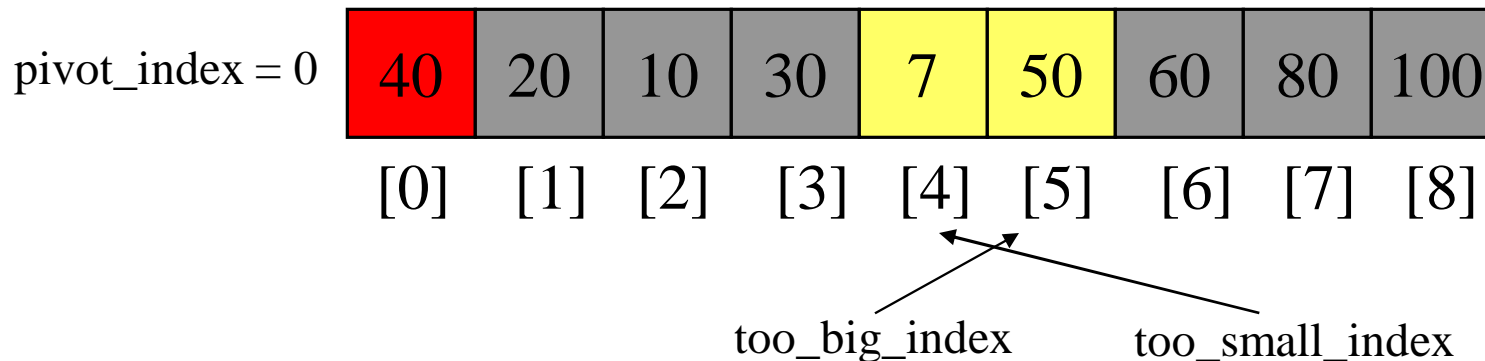
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
- 2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



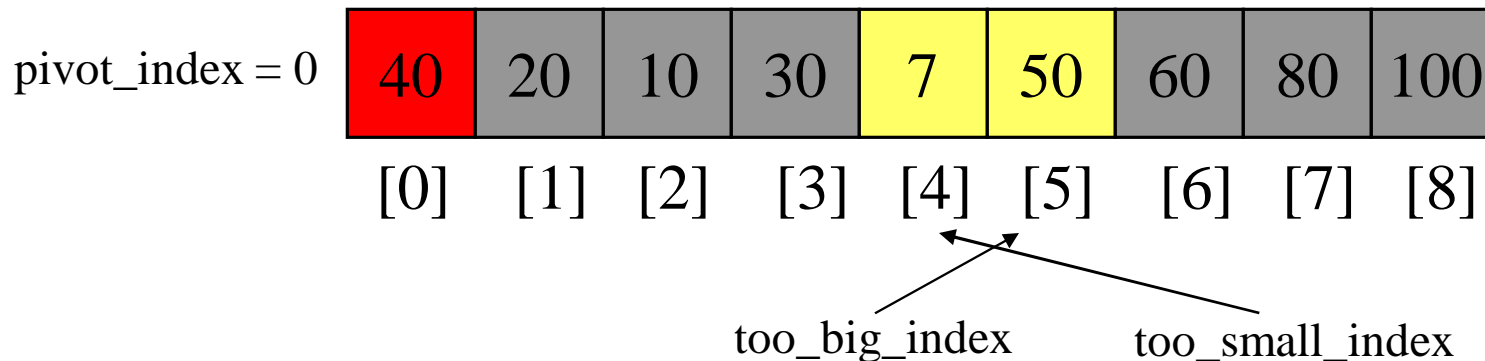
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
- 3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



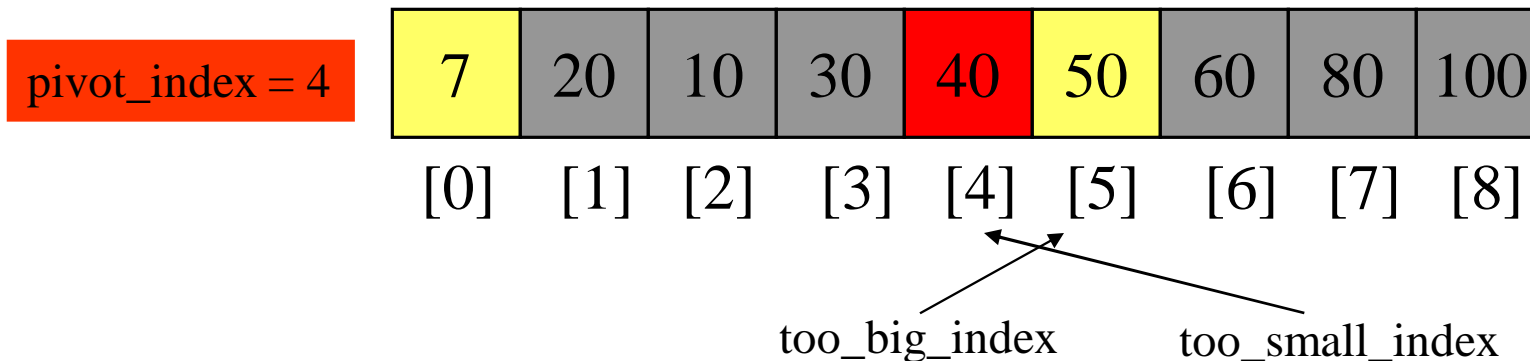
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
- 4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.



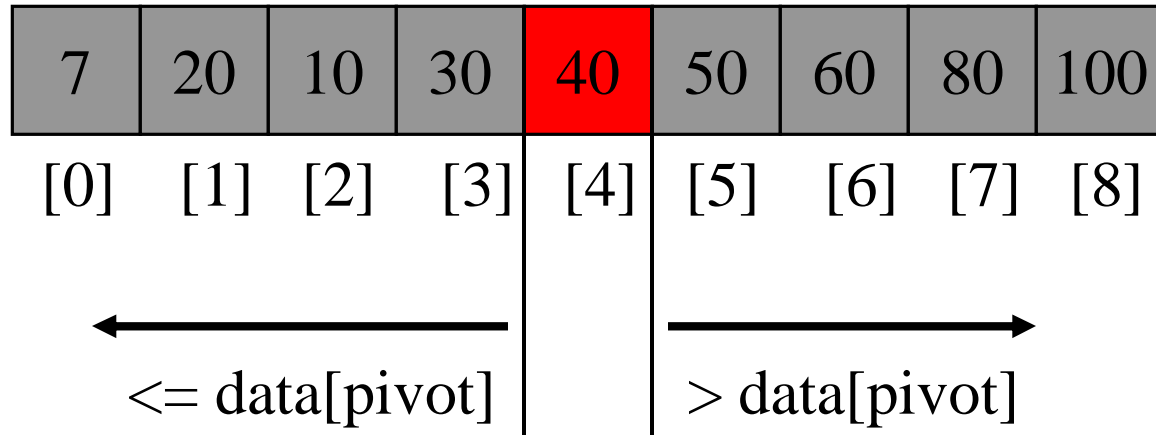
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.
- 5. Swap $\text{data}[\text{too_small_index}]$ and $\text{data}[\text{pivot_index}]$



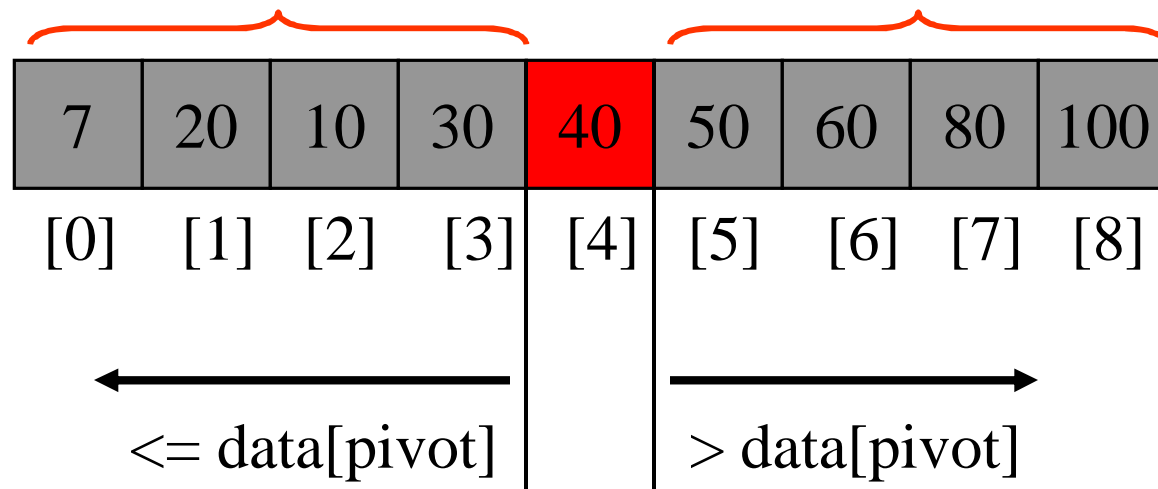
1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.
- 5. Swap $\text{data}[\text{too_small_index}]$ and $\text{data}[\text{pivot_index}]$



Partition Result



Recursion: Quicksort Sub-arrays



Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- What is best case running time?

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- What is best case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays of size $n/2$
 2. Quicksort each sub-array

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- What is best case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays of size $n/2$
 2. Quicksort each sub-array
 - Depth of recursion tree?

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- What is best case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays of size $n/2$
 2. Quicksort each sub-array
 - Depth of recursion tree? $O(\log_2 n)$

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- What is best case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays of size $n/2$
 2. Quicksort each sub-array
 - Depth of recursion tree? $O(\log_2 n)$
 - Number of accesses in partition?

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- What is best case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays of size $n/2$
 2. Quicksort each sub-array
 - Depth of recursion tree? $O(\log_2 n)$
 - Number of accesses in partition? $O(n)$

Quicksort Analysis

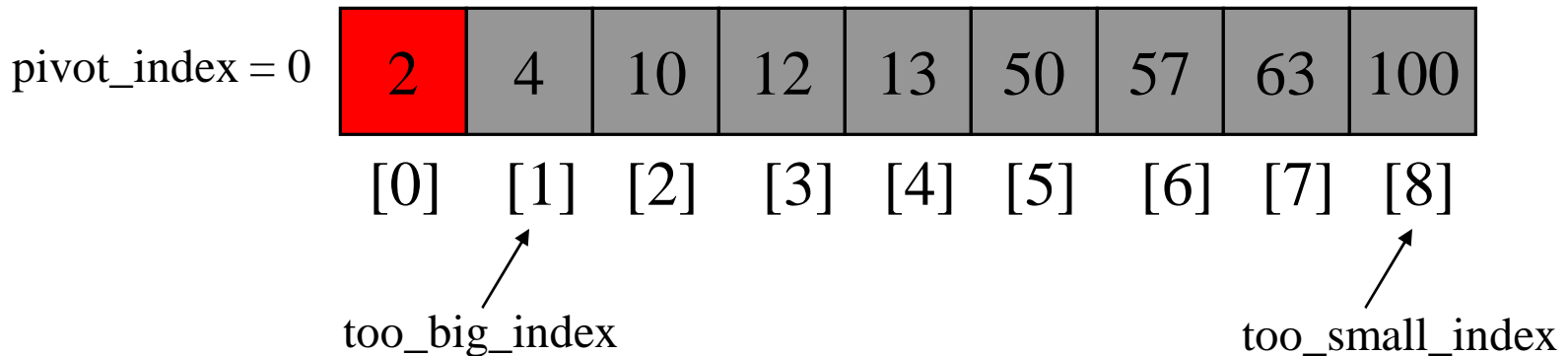
- Assume that keys are random, uniformly distributed.
- Best case running time: $O(n \log_2 n)$

Quicksort Analysis

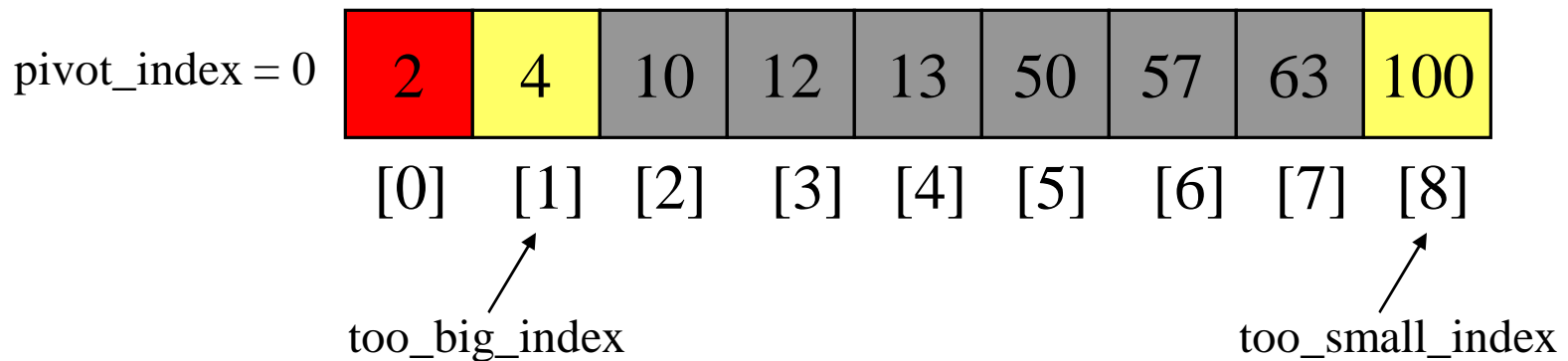
- Assume that keys are random, uniformly distributed.
- Best case running time: $O(n \log_2 n)$
- Worst case running time?

Quicksort: Worst Case

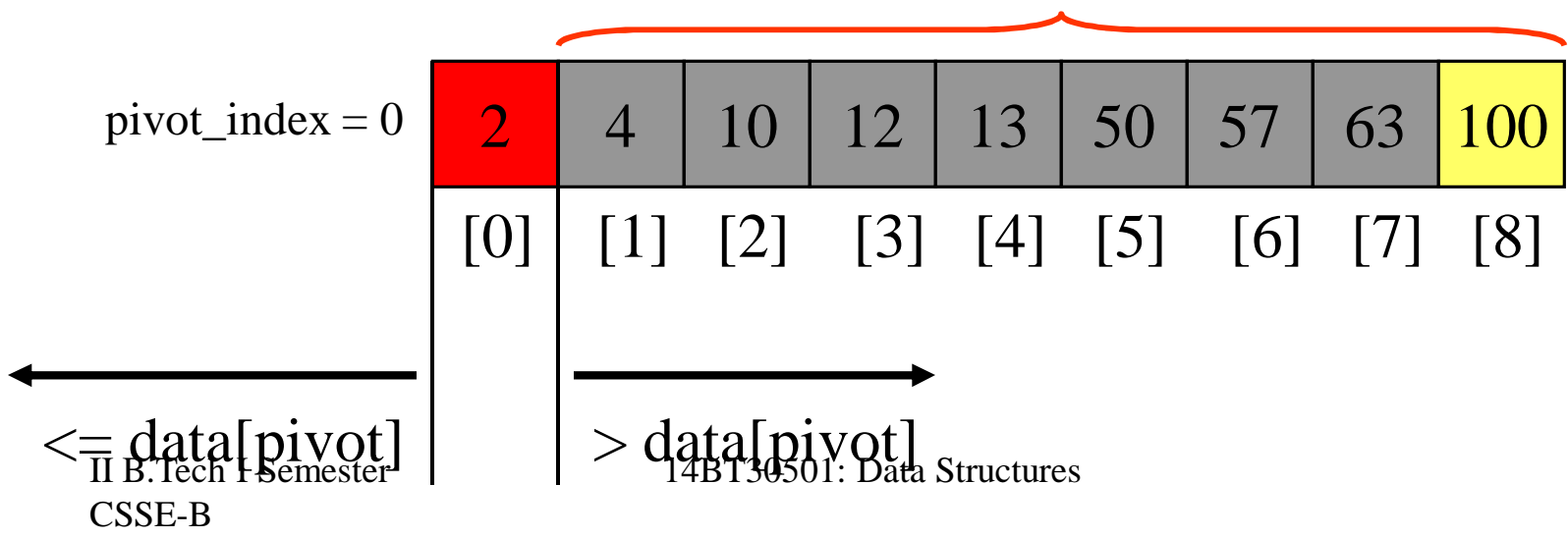
- Assume first element is chosen as pivot.
- Assume we get array that is already in order:



- 1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 ++too_big_index
- 2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 --too_small_index
- 3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
- 4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.
- 5. Swap $\text{data}[\text{too_small_index}]$ and $\text{data}[\text{pivot_index}]$



1. While $\text{data}[\text{too_big_index}] \leq \text{data}[\text{pivot}]$
 $++\text{too_big_index}$
2. While $\text{data}[\text{too_small_index}] > \text{data}[\text{pivot}]$
 $--\text{too_small_index}$
3. If $\text{too_big_index} < \text{too_small_index}$
 swap $\text{data}[\text{too_big_index}]$ and $\text{data}[\text{too_small_index}]$
4. While $\text{too_small_index} > \text{too_big_index}$, go to 1.
- 5. Swap $\text{data}[\text{too_small_index}]$ and $\text{data}[\text{pivot_index}]$



Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- Best case running time: $O(n \log_2 n)$
- Worst case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays:
 - one sub-array of size 0
 - the other sub-array of size $n-1$
 2. Quicksort each sub-array
 - Depth of recursion tree?

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- Best case running time: $O(n \log_2 n)$
- Worst case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays:
 - one sub-array of size 0
 - the other sub-array of size $n-1$
 2. Quicksort each sub-array
 - Depth of recursion tree? $O(n)$

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- Best case running time: $O(n \log_2 n)$
- Worst case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays:
 - one sub-array of size 0
 - the other sub-array of size $n-1$
 2. Quicksort each sub-array
 - Depth of recursion tree? $O(n)$
 - Number of accesses per partition?

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- Best case running time: $O(n \log_2 n)$
- Worst case running time?
 - Recursion:
 1. Partition splits array in two sub-arrays:
 - one sub-array of size 0
 - the other sub-array of size $n-1$
 2. Quicksort each sub-array
 - Depth of recursion tree? $O(n)$
 - Number of accesses per partition? $O(n)$

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- Best case running time: $O(n \log_2 n)$
- Worst case running time: $O(n^2)!!!$

Quicksort Analysis

- Assume that keys are random, uniformly distributed.
- Best case running time: $O(n \log_2 n)$
- Worst case running time: $O(n^2)!!!$
- What can we do to avoid worst case?

Improved Pivot Selection

Pick median value of three elements from data array:
 $\text{data}[0]$, $\text{data}[n/2]$, and $\text{data}[n-1]$.

Use this median value as pivot.

Improving Performance of Quicksort

- Improved selection of pivot.
- For sub-arrays of size 3 or less, apply brute force search:
 - Sub-array of size 1: trivial
 - Sub-array of size 2:
 - if($\text{data}[\text{first}] > \text{data}[\text{second}]$) swap them
 - Sub-array of size 3: left as an exercise.

SHORING

Mr. B. Hari Krishna
Assistant Professor
Department of Civil Engineering
Sree Vidyanikethan Engineering College

Unit-III
II B.Tech I semester
Dept. of Civil Engineering

SHORING

Definition:

It is the construction of a temporary structure to support temporarily an unsafe structure.

Shoring acts as lateral support to the walls.

When we need shoring?

- If there is any bulging of wall due to bad workmanship.
- When a wall cracks due to unequal settlements of foundation and the cracked wall need repairs.
- When an adjacent structure is to be dismantled.
- When openings are to be made or enlarged in the wall.

Types of shores

- 1) Raking shores
- 2) Flying or horizontal shores
- 3) Dead or vertical shores

Raking shores

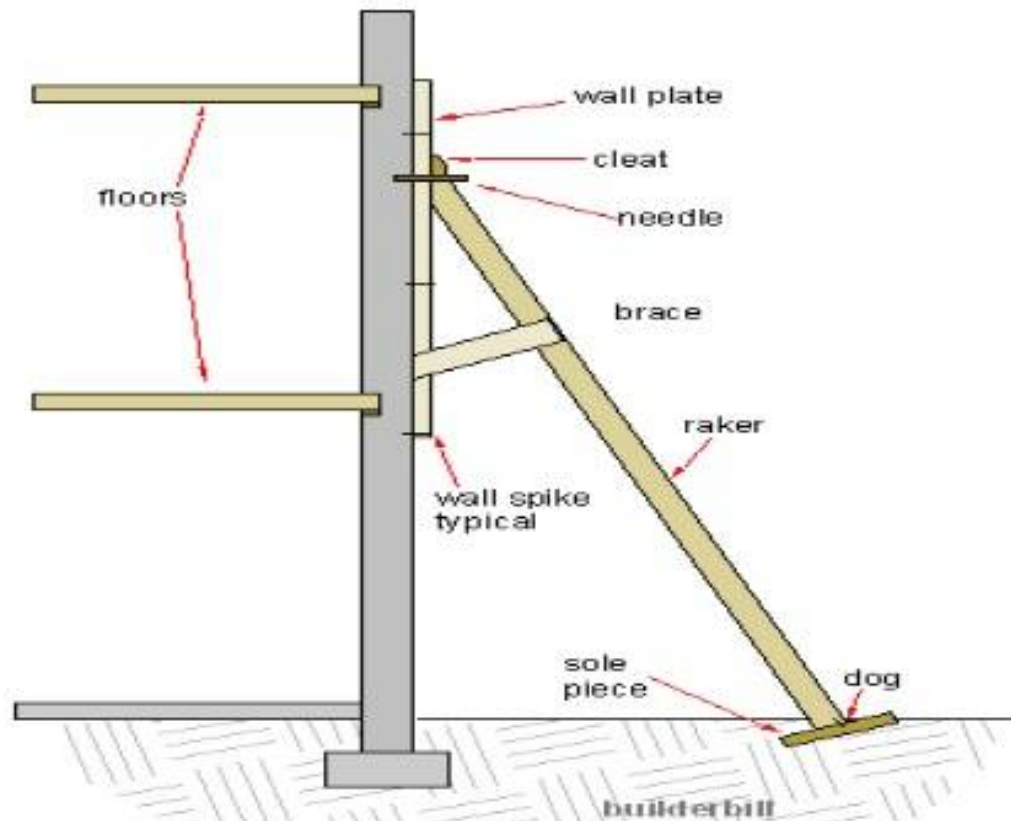
- In this method, inclined members known as rakers are used to give lateral supports to walls.
- A raking shore consists of the following components:
 - 1)Rakers or inclined member
 - 2)Wall plate
 - 3)Needles
 - 4)Cleats
 - 5)Bracing
 - 6)Sole plate

Raking shores – Erection methodology

• **The following points are to be kept in view for the use of the raking shores:**

- a) Rakers are to be inclined to the ground at angle of 45 degrees to make them more effective. In practice the angle may vary from 45 to 75 degrees.
- b) The top raker should not be inclined steeper than 75 degrees.
- c) The size of the rakers is to be decided on the basis of anticipated thrust from the wall.
- d) If longer length of wall needs support, shoring may be spaced at 3 to 4.5 m spacing, depending upon the requirement

RAKING OR INCLINED SHORES



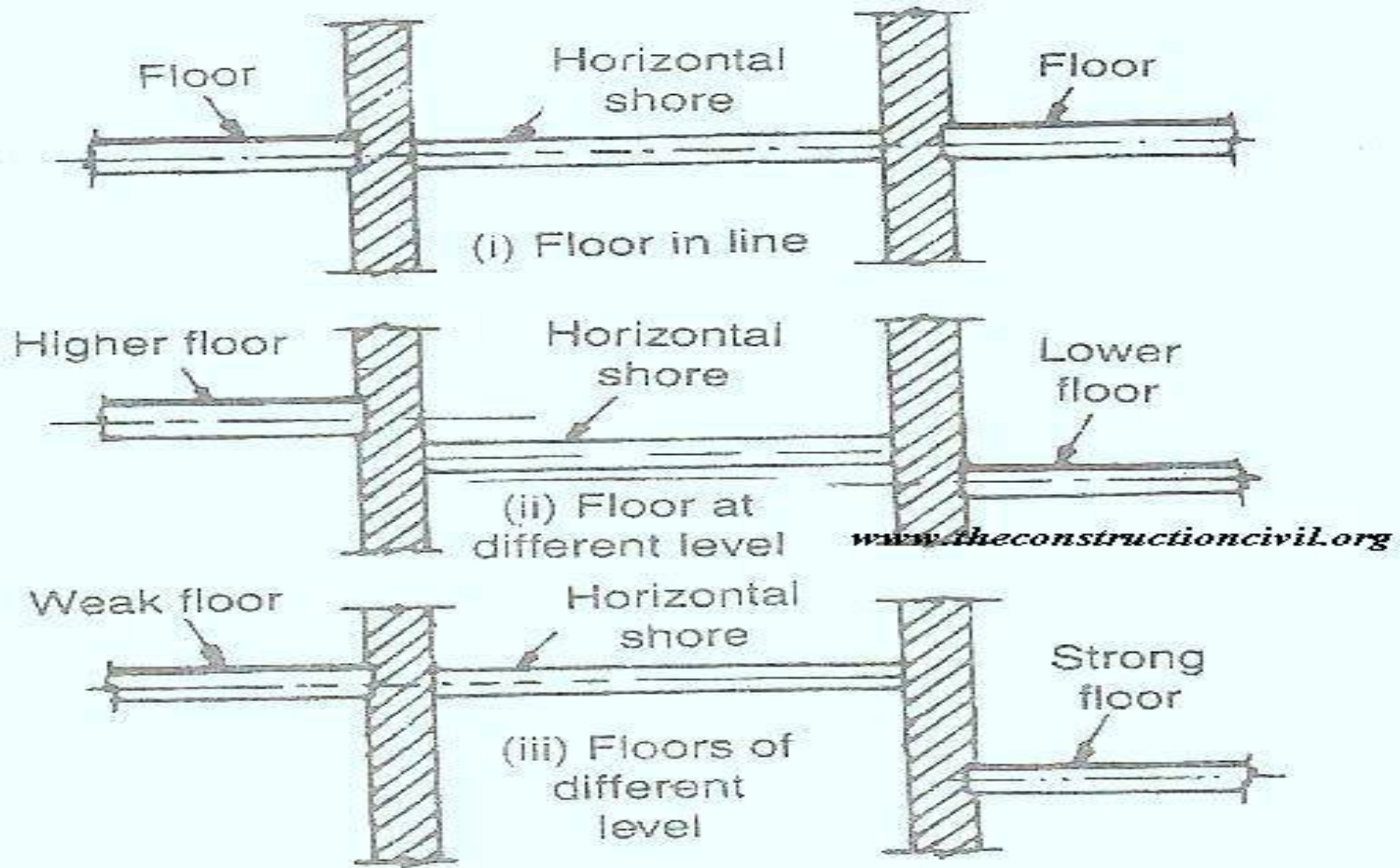
A Single Raking Timber Shore

Flying or horizontal shores

- These shores are used to give horizontal support to two adjacent, parallel party walls which have become unsafe due to removal or collapse of the intermediate building.
- All types of arrangements of supporting the unsafe structure in which the shores do not reach the ground fall under this category.
- If the walls are quite near to each other (distance up to 9 m), **single flying shores** can be constructed.
- If the distance between the walls is more **compound or double flying shore** may be provided.

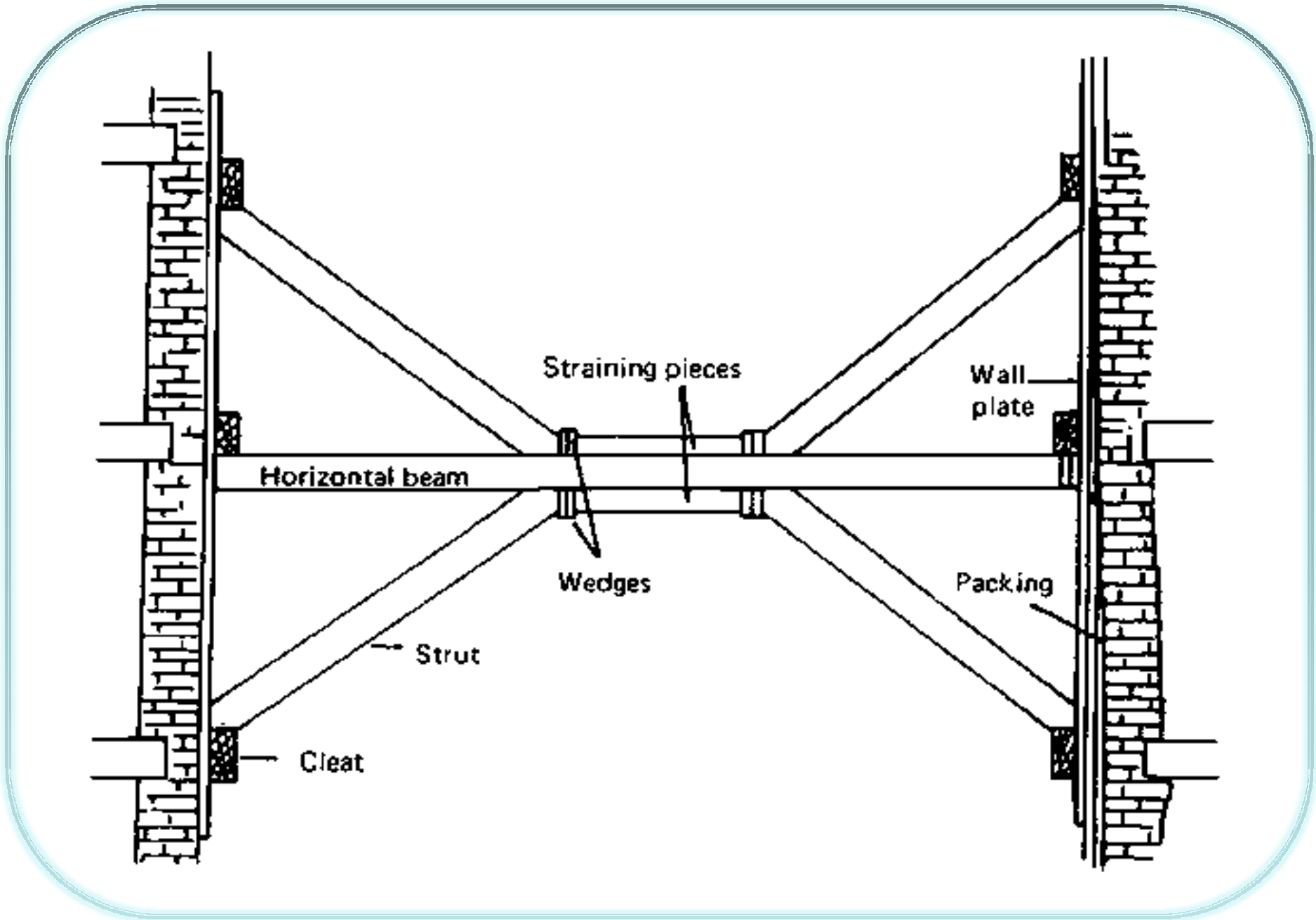
Flying or horizontal shores

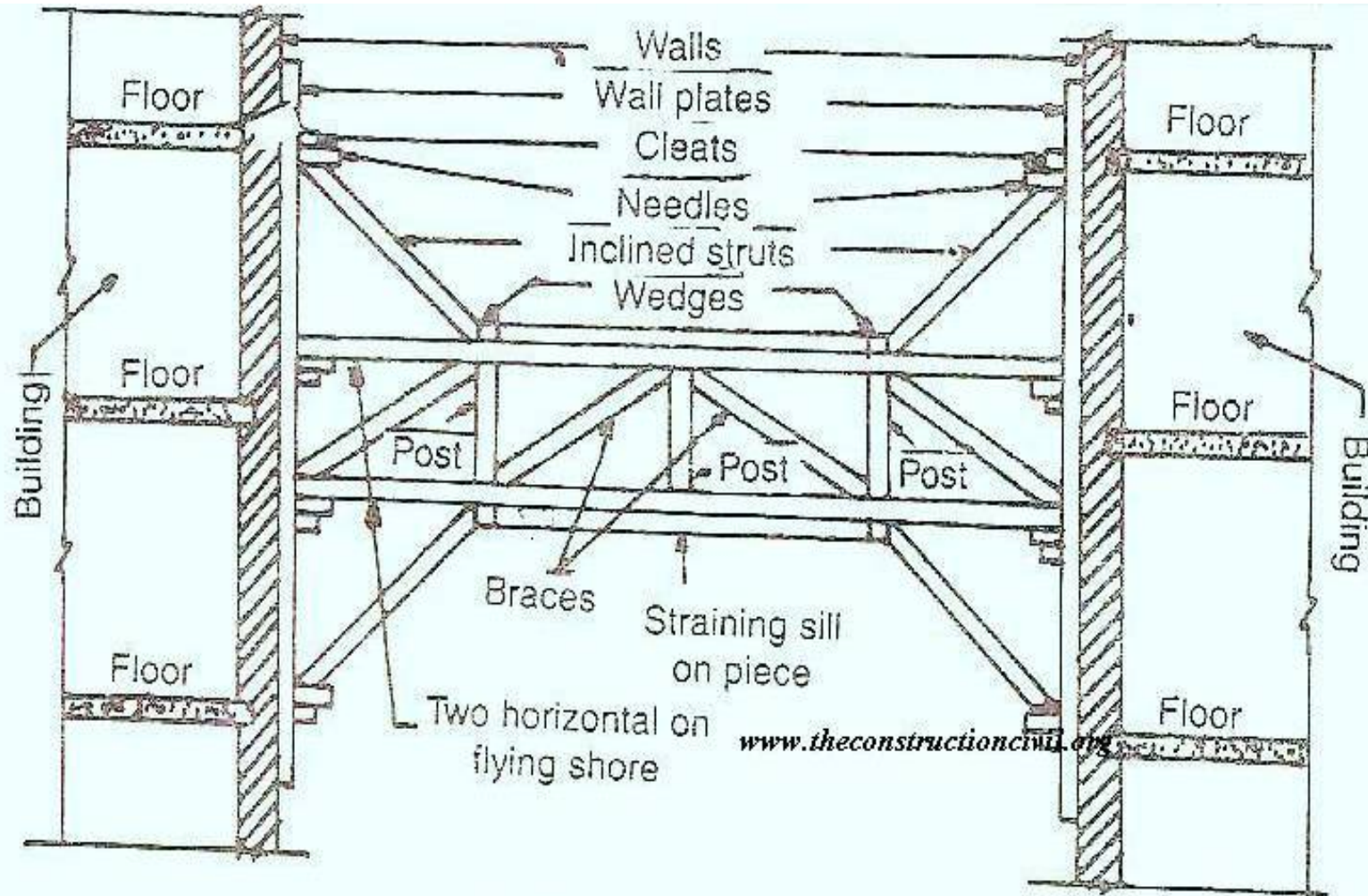
- It consists of following components;
 - 1) Wall plates
 - 2) Needles
 - 3) Cleats
 - 4) Struts
 - 5) Horizontal shore
 - 6) Straining pieces
 - 7) Folding wedges



www.theconstructioncivil.org

Shore positions relative to floor levels and floor strength.





SECTIONAL ELEVATION
 Details of double flying shore or horizontal shore.

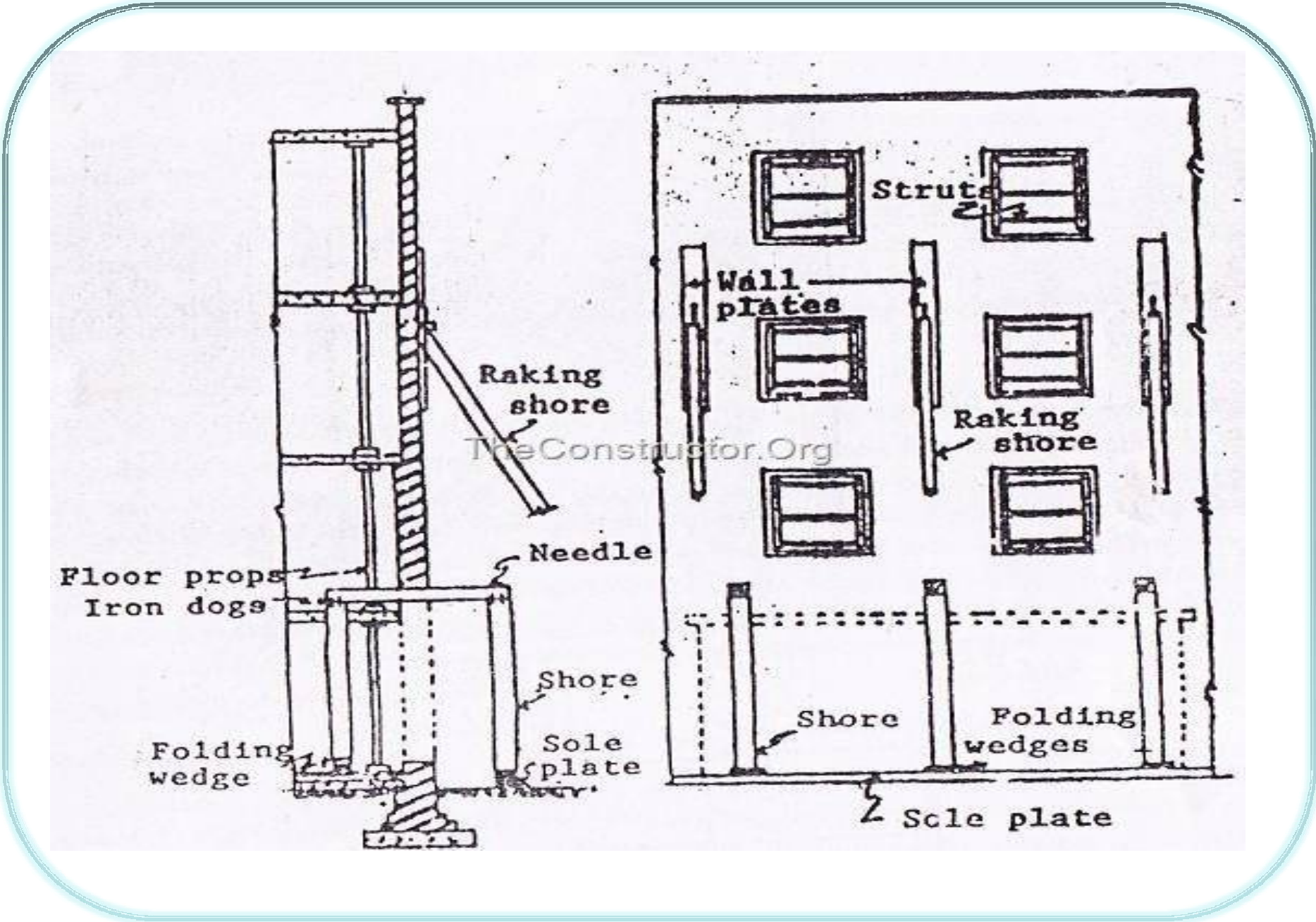
Dead or vertical shores

- These shores consists of vertical members known as dead shores supporting horizontal members known as needles.
- The needles transfer the load of the wall to the dead shores.
- This type of shoring is provided under the following circumstances:
 - a)To build the defective lower part of the wall
 - b)To rebuild or deepen the existing foundation.
 - c)To make large openings in the existing wall at the lower level.

Dead or vertical shores

- i) The sequence of removals should be
 - a) Needles
 - b) Strutting from opening
 - c) Floor strutting inside
 - d) Raking shore if any.

An interval of 2 days should be allowed between each one of these removal operations.



Shoring and scaffolding

Thank You

ARTIFICIAL INTELLIGENCE AND ROBOTICS

Search Strategies

By

Mr.G.V.V.S.Reddy Prasad,

Assistant Professor,
Dept of Mechanical Engineering.

Target Group: III B.Tech, II – Sem (Open Elective)



SREE VIDYANIKETHAN ENGINEERING COLLEGE

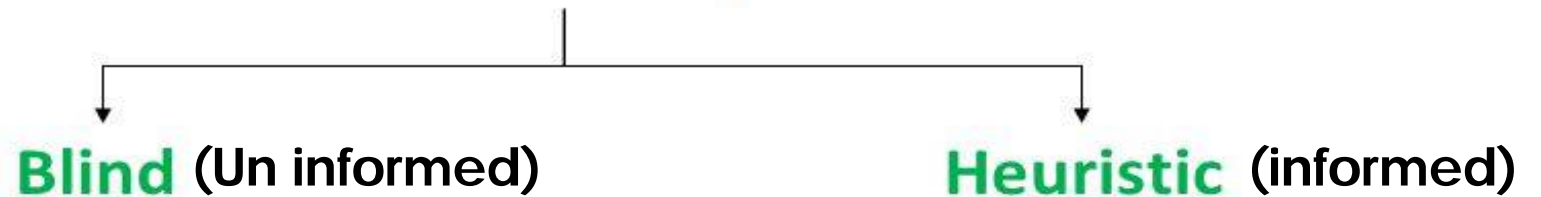
AUTONOMOUS)

Sree Sainath Nagar, Tirupati – 517 102, A.P.

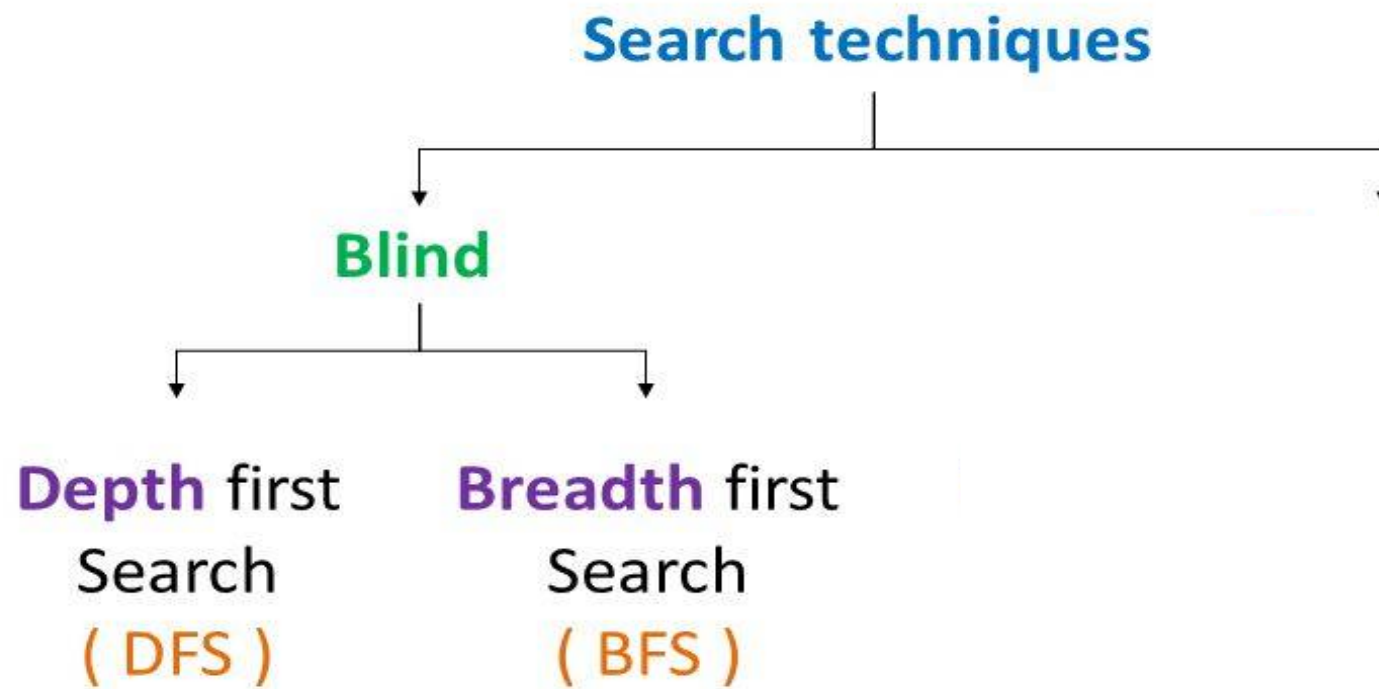
DEPARTMENT OF MECHANICAL ENGINEERING

SEARCH TECHNIQUES

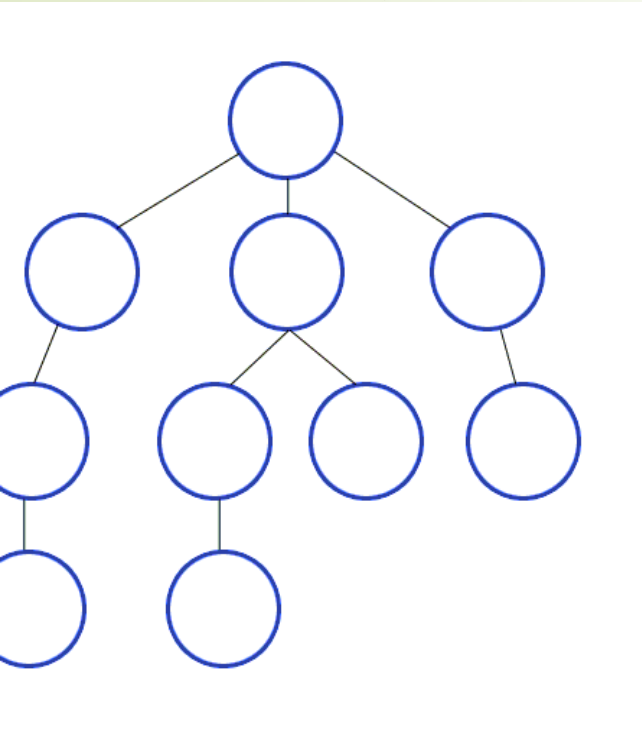
Search techniques



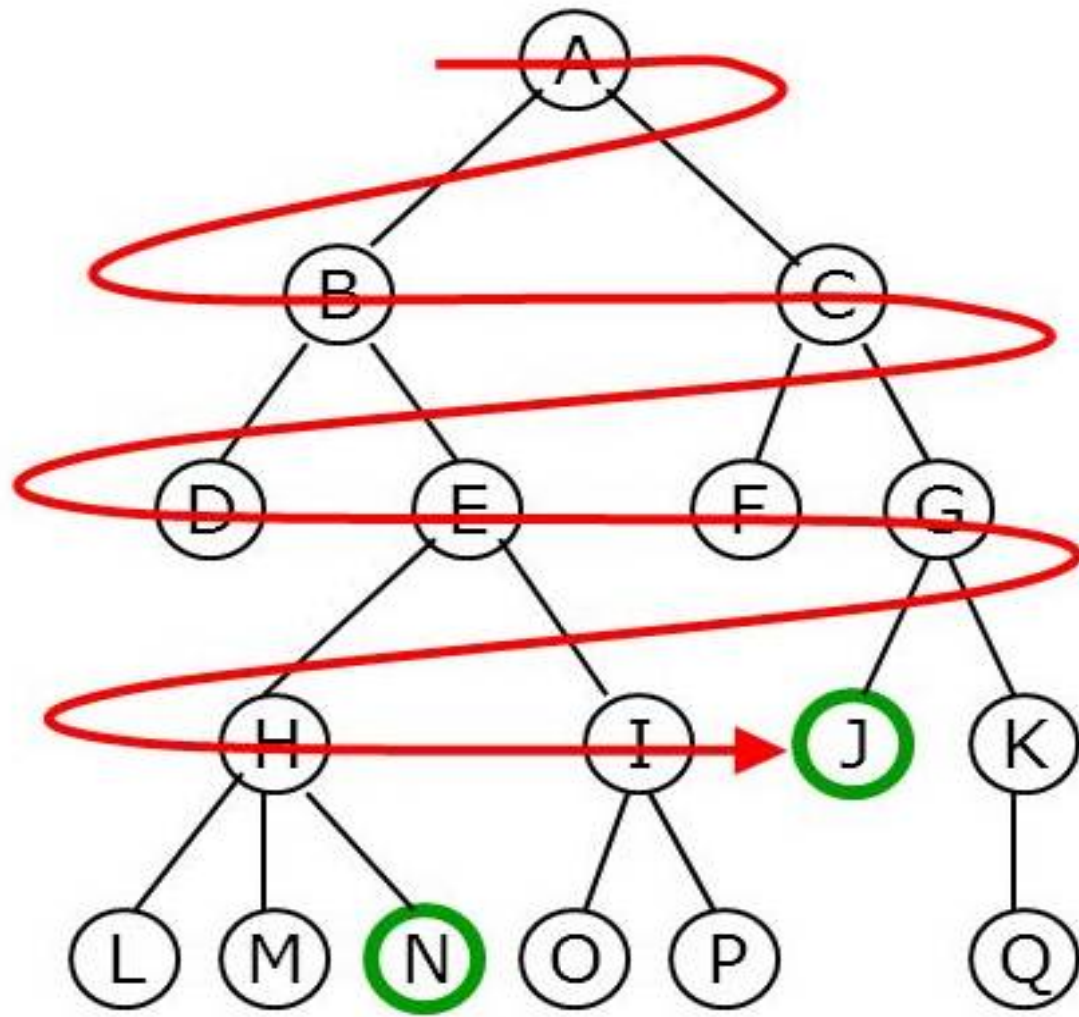
SEARCH TECHNIQUES



BREADTH FIRST SEARCH



- *Root node is expanded first*
- *Then all the nodes generated by the root node are expanded next*
- *....so on...*
- *Mathematically: all the Nodes at depth "d" are expanded before the node "d+1"*

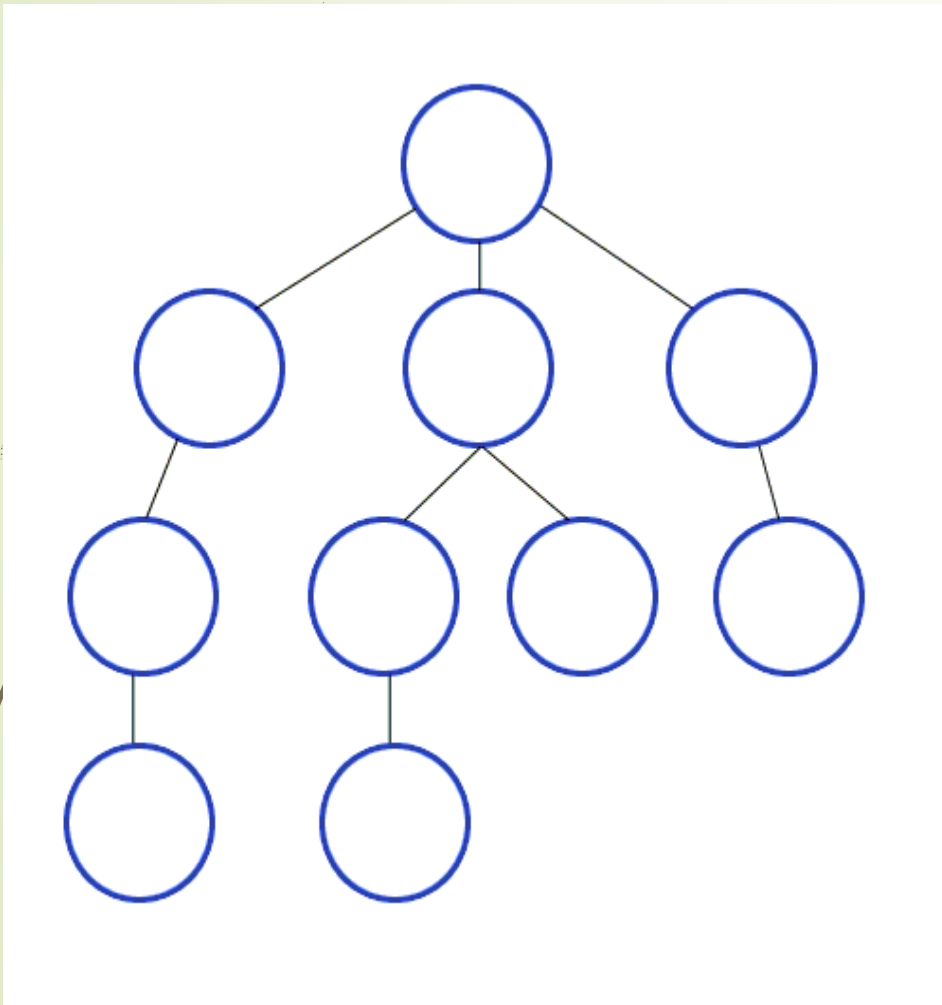


Breadth first search



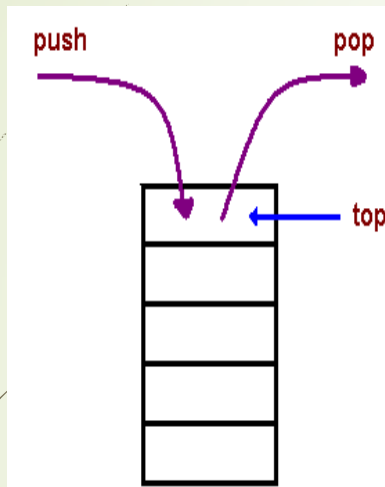
DEPTH FIRST SEARCH

Depth first search

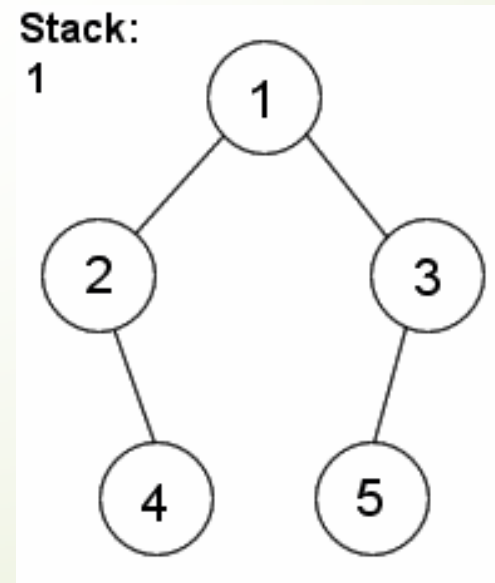


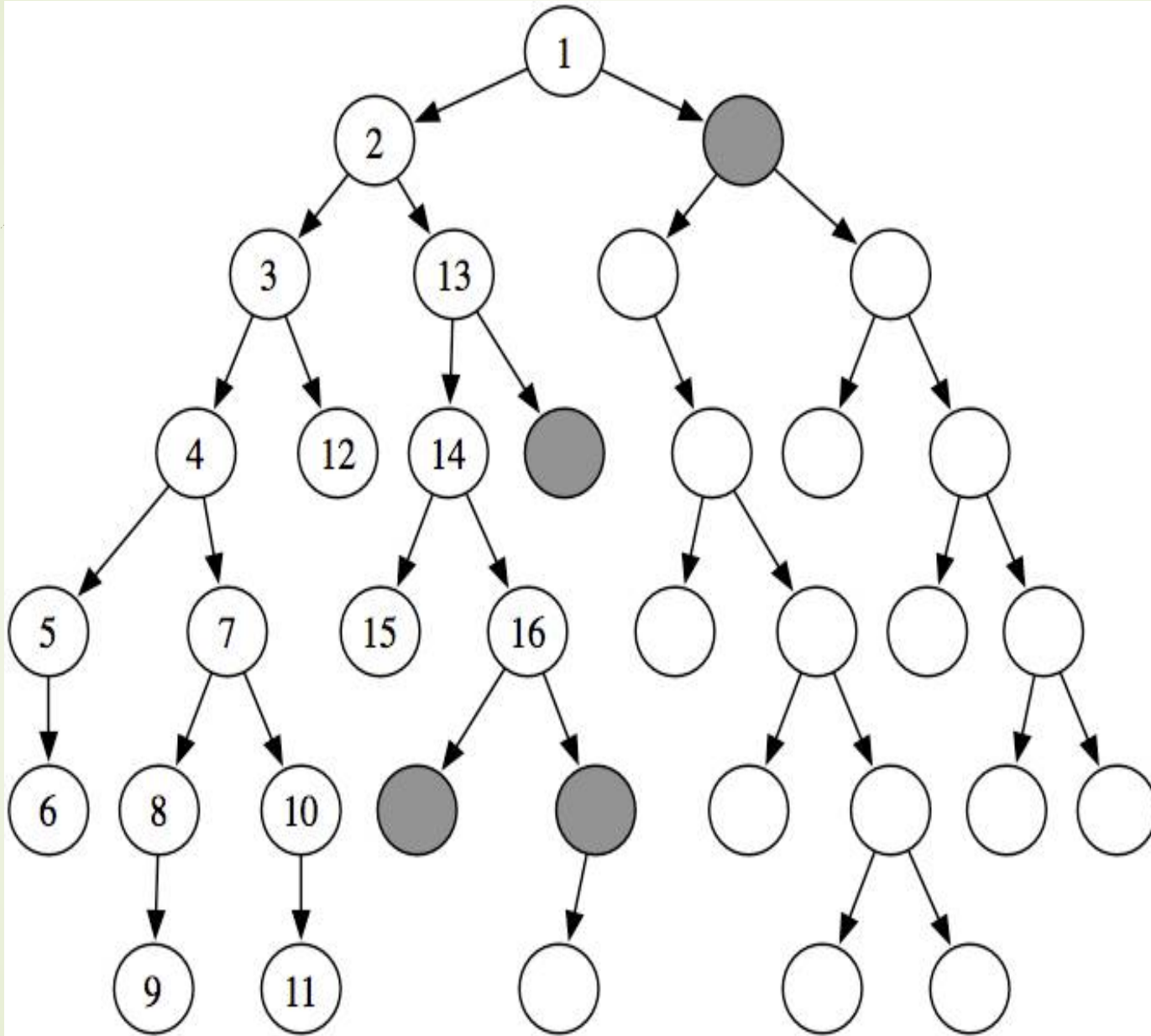
- Searches deeper into problem space
- Expands one of the nodes at the deepest level of the tree
- i.e., Expands the deepest unexpanded node
- Implements Stack principle: LIFO

STACK principle:



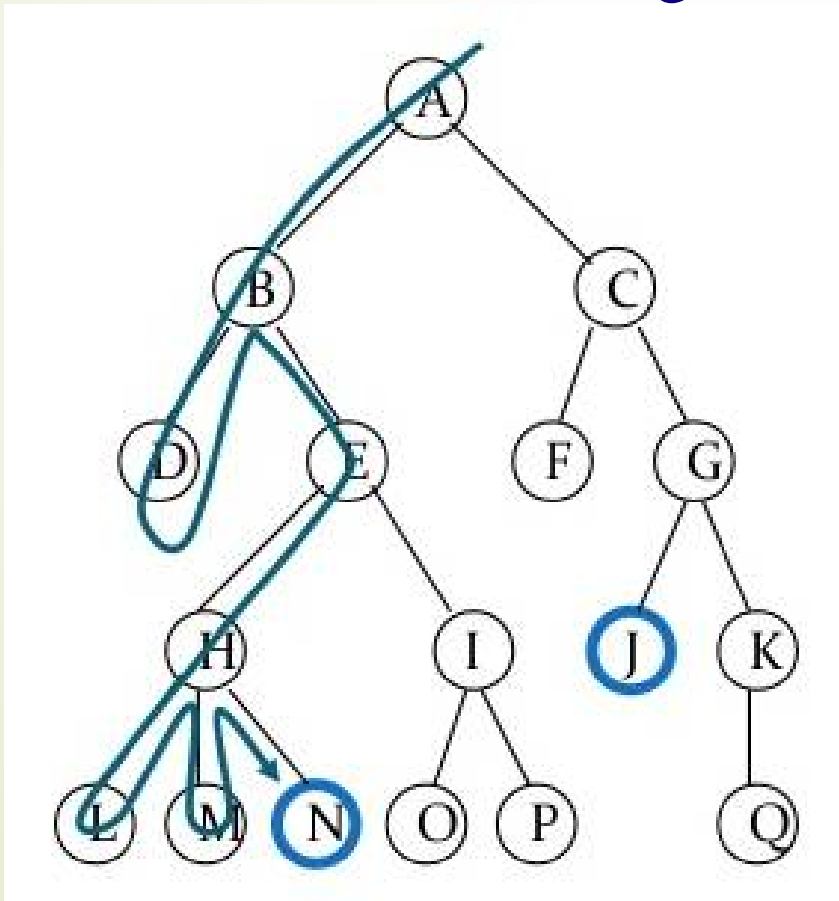
- Stack is a container of objects that are inserted and removed according to the last-in first-out (LIFO) principle
- **push** the item into the stack, and **pop** the item out of the stack
- elements can be added and removed from the stack only at the top





Depth first search

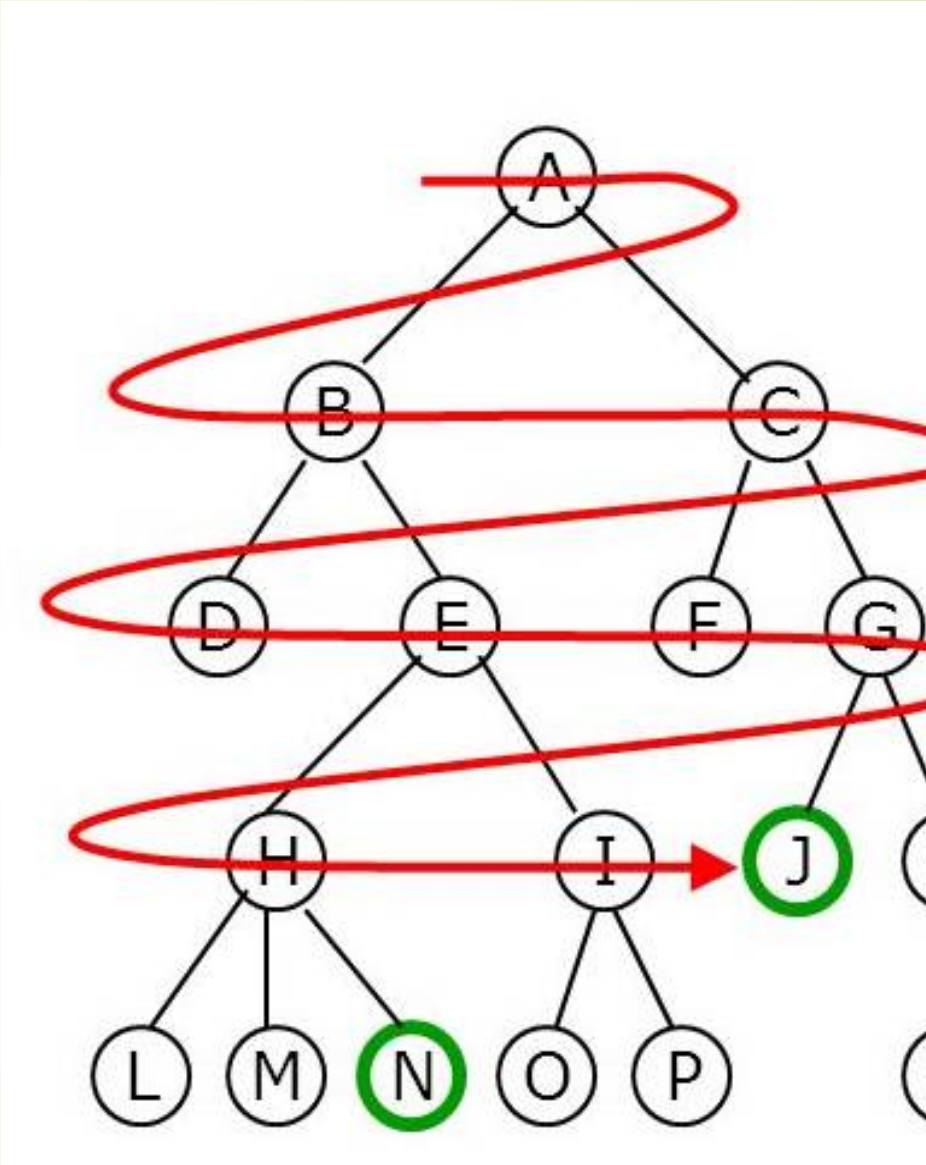
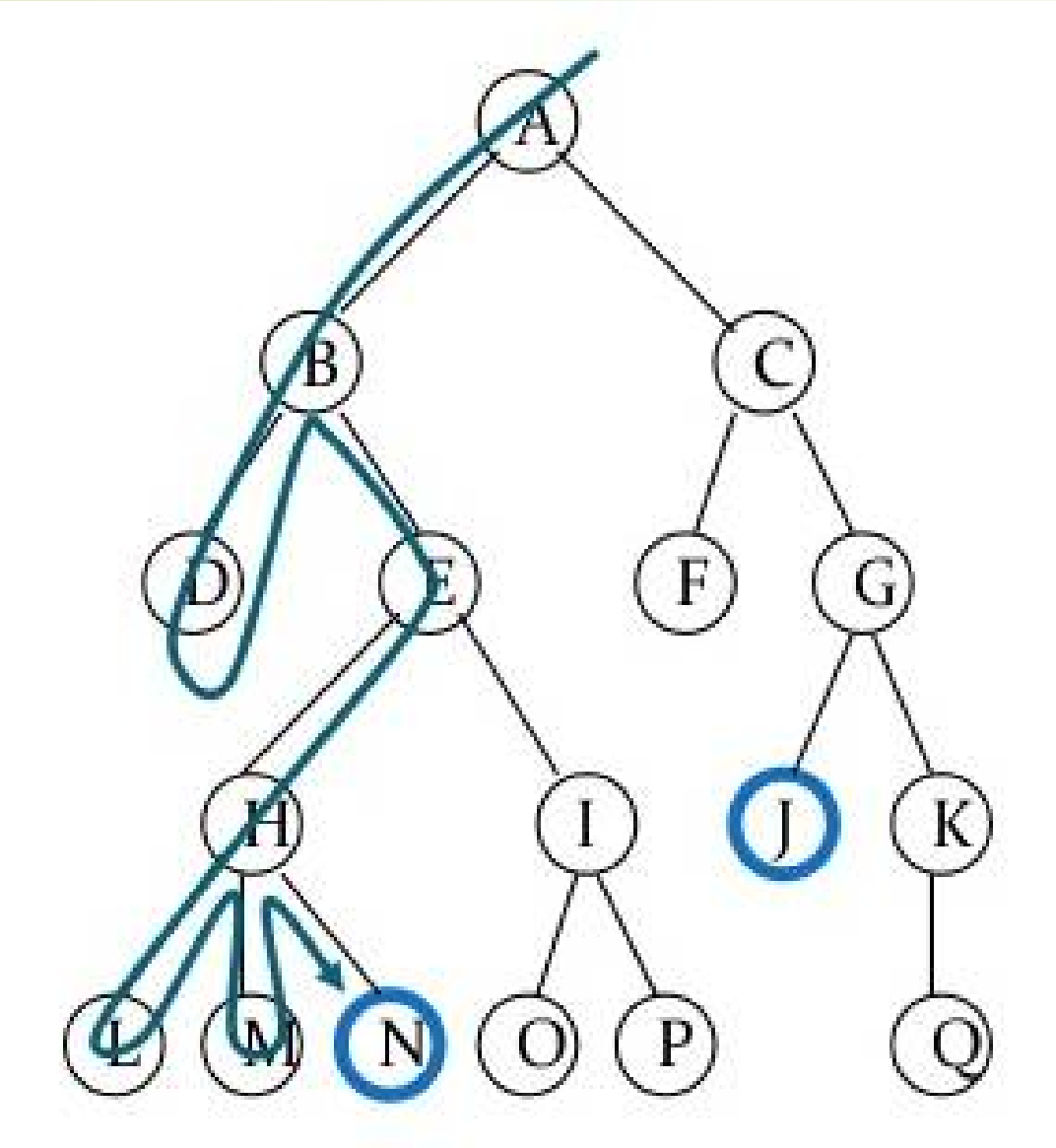
- explores as far as possible along each branch before **backtracking**.

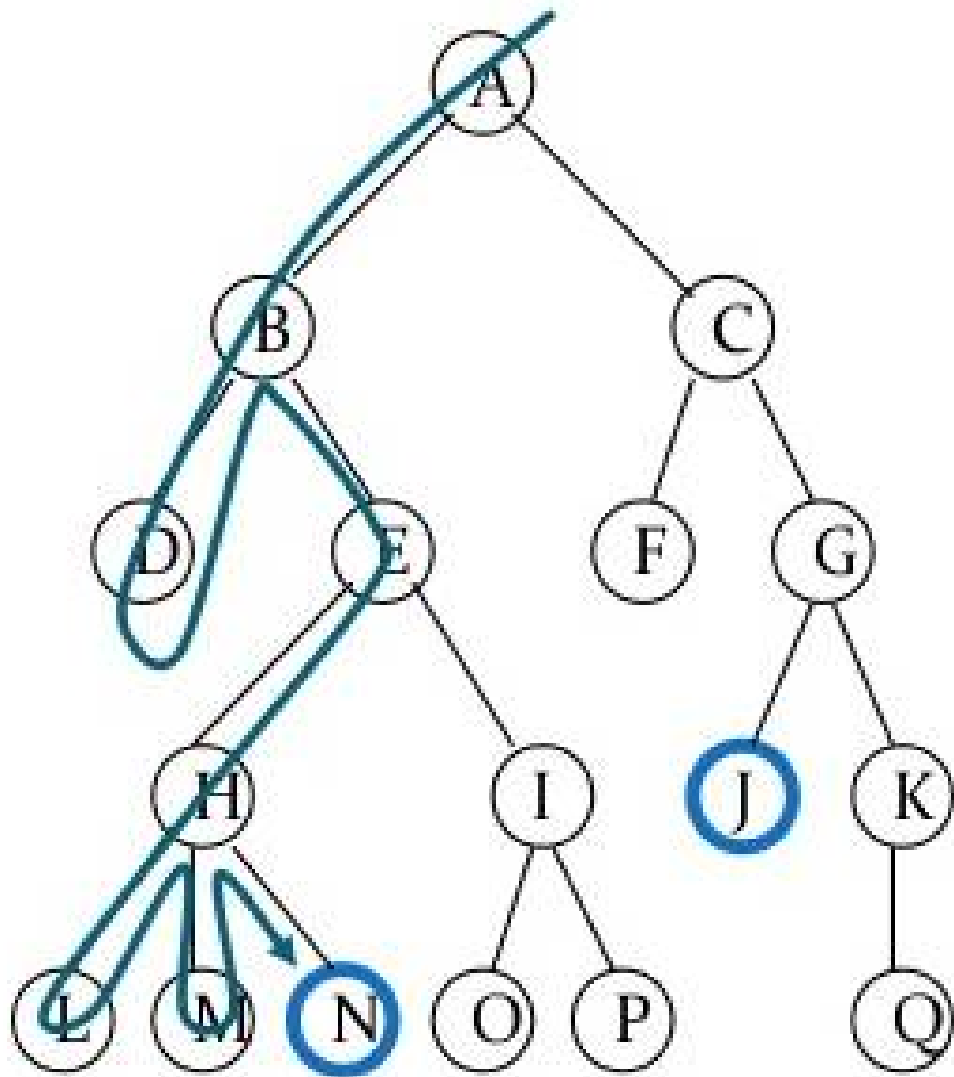


SEARCH FLOW:

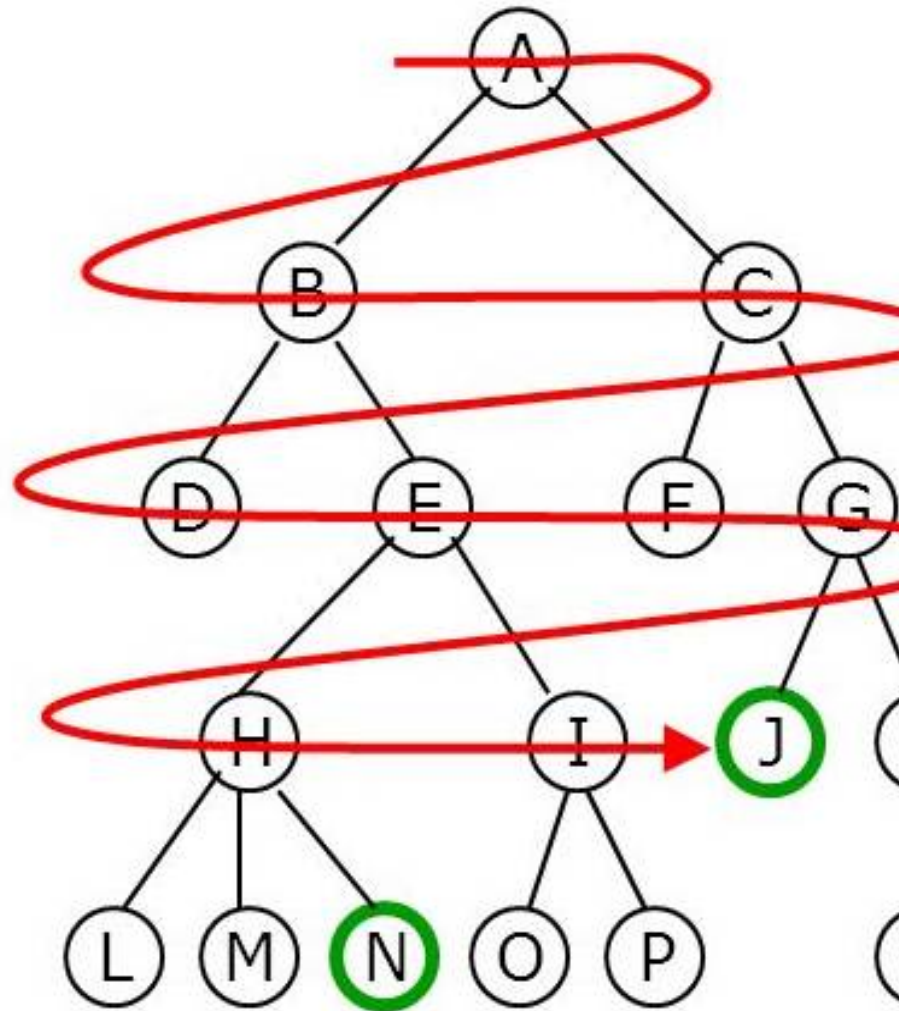
A → B → D → E → H
→ L → M → N

GOAL STATE



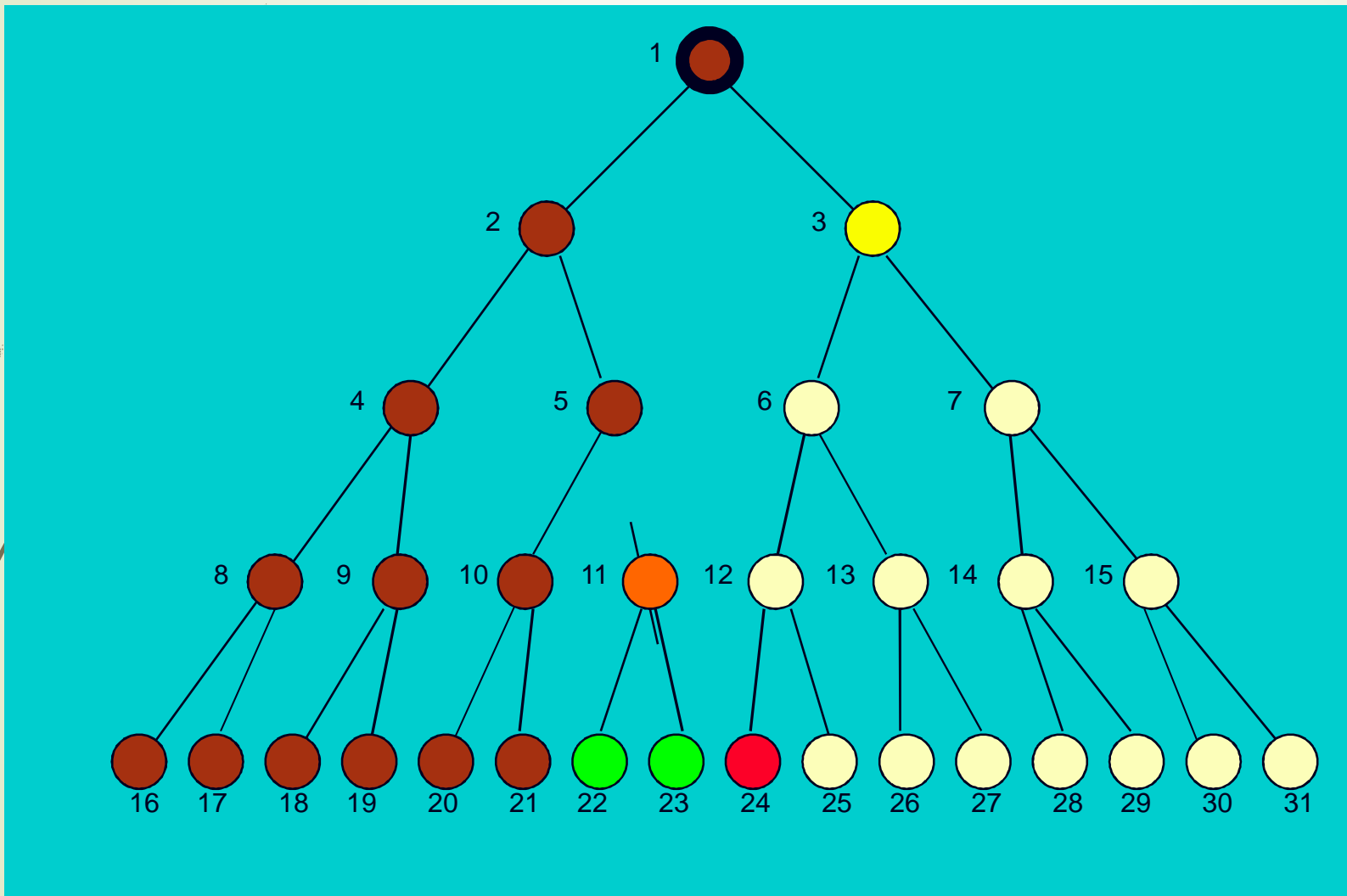


Depth first search



Breadth first search

Depth-First Snapshot



- Initial ●
- Visited ●
- Fringe ●
- Current ●
- Visible ●
- Goal ●

Fringe: [3] + [22, 23]



FEAUTURES

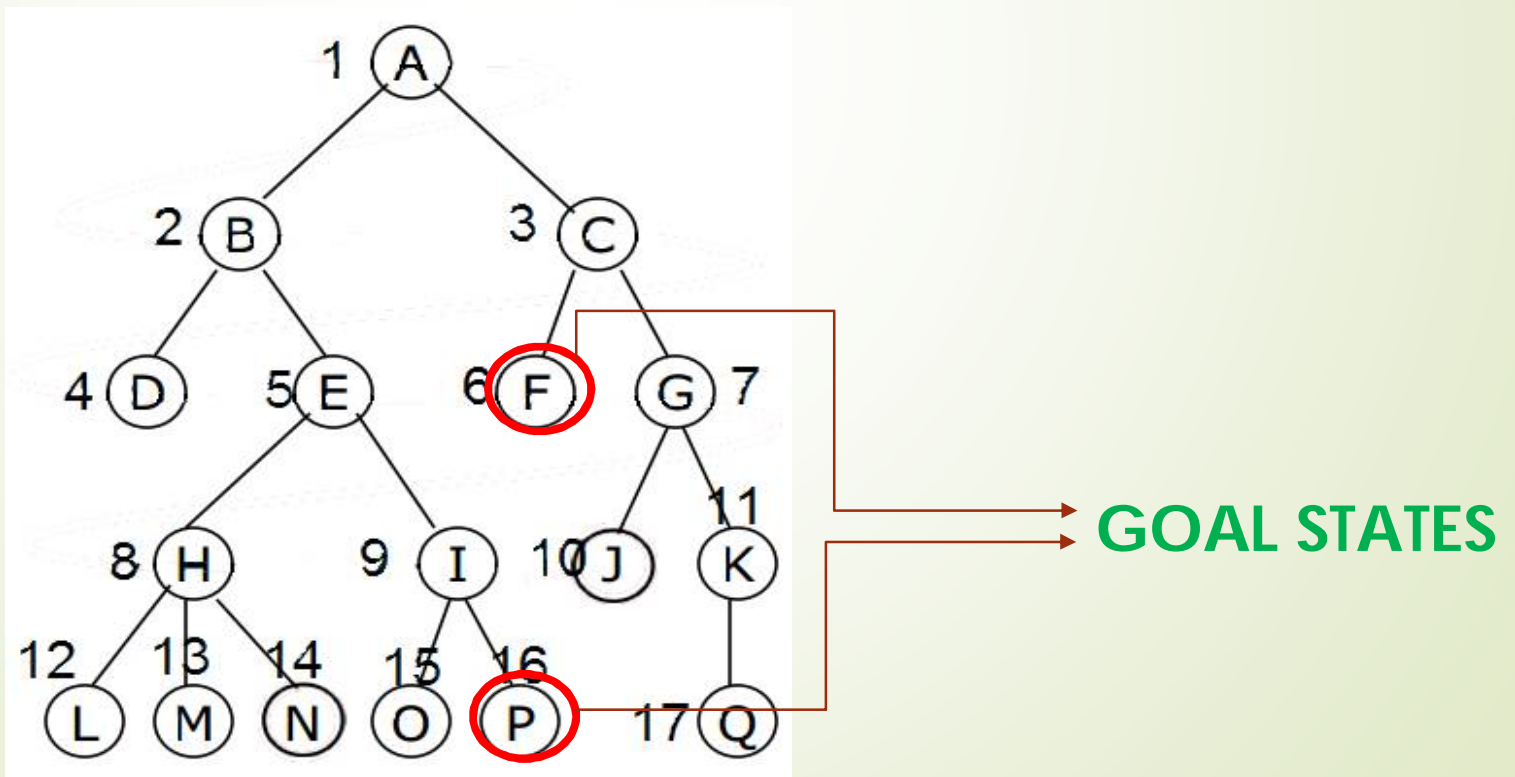
- Simple to implement
- Modest memory requirements
- Needs to store only single path from root node to a leaf node

FOR EXAMPLE....

- Consider a hypothetical state space where every state can be expanded to yield **b** new states.
- i.e., Branching factor of the search tree is **b**.
- Maximum depth= **d**
- depth-first search requires storage of only ***bm*** nodes
- in case of BFS= ***b^d***

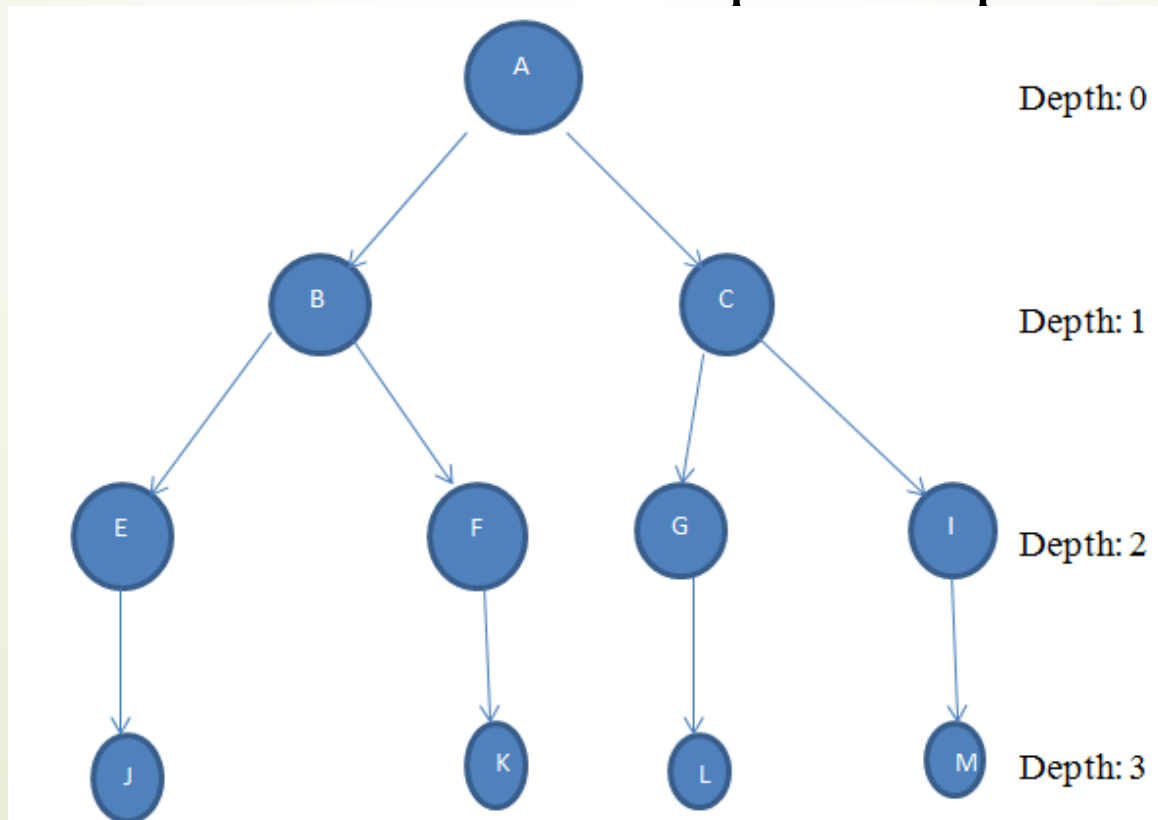
Disadvantages:

- Can get stuck going down the wrong path in an infinite loop
- Can not guarantee an optimal solution



DEPTH LIMITED SEARCH

- Applied when we know the depth
- Impose a limit on the depth of search
- Cut off on maximum depth of a path



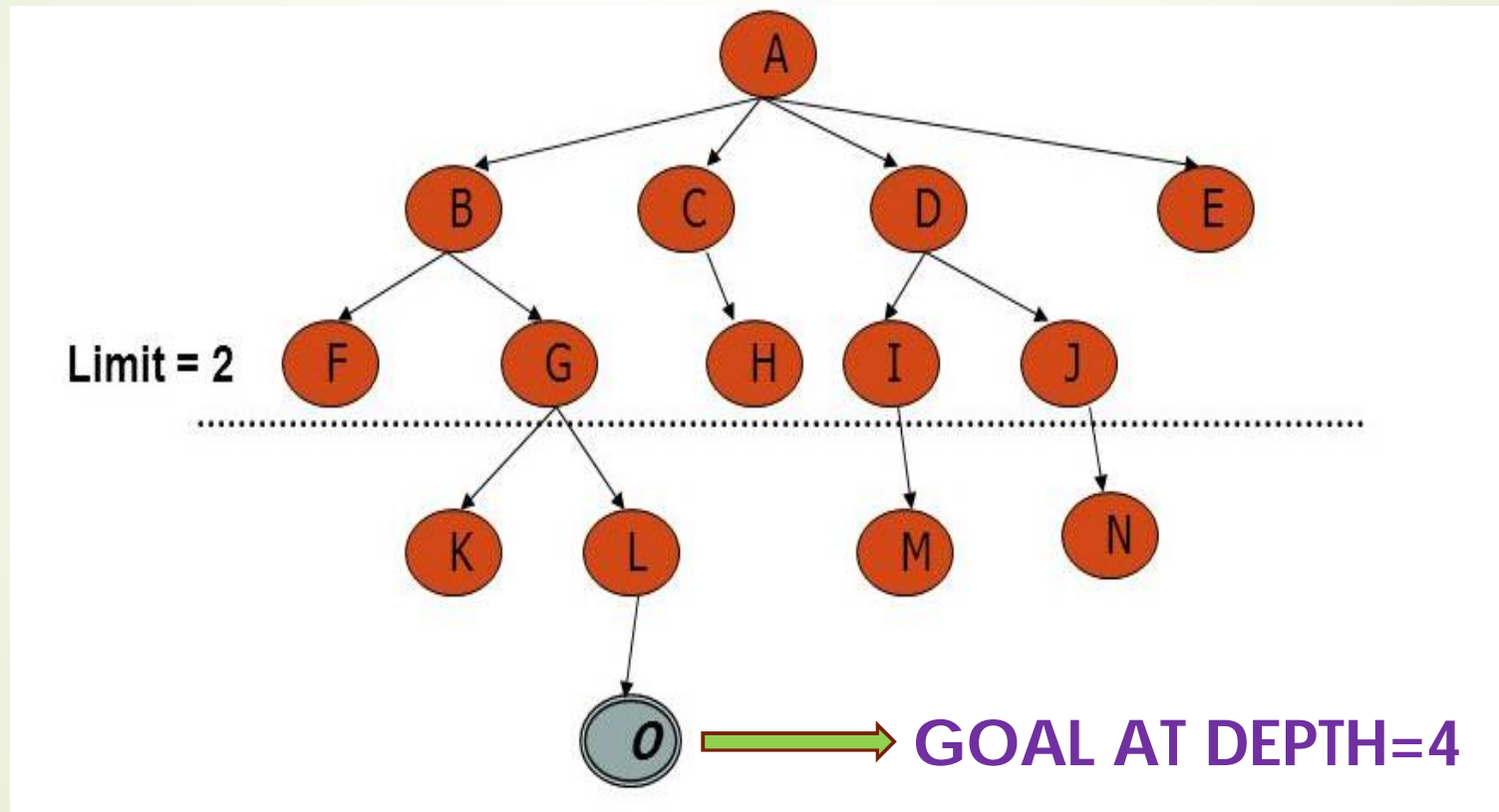


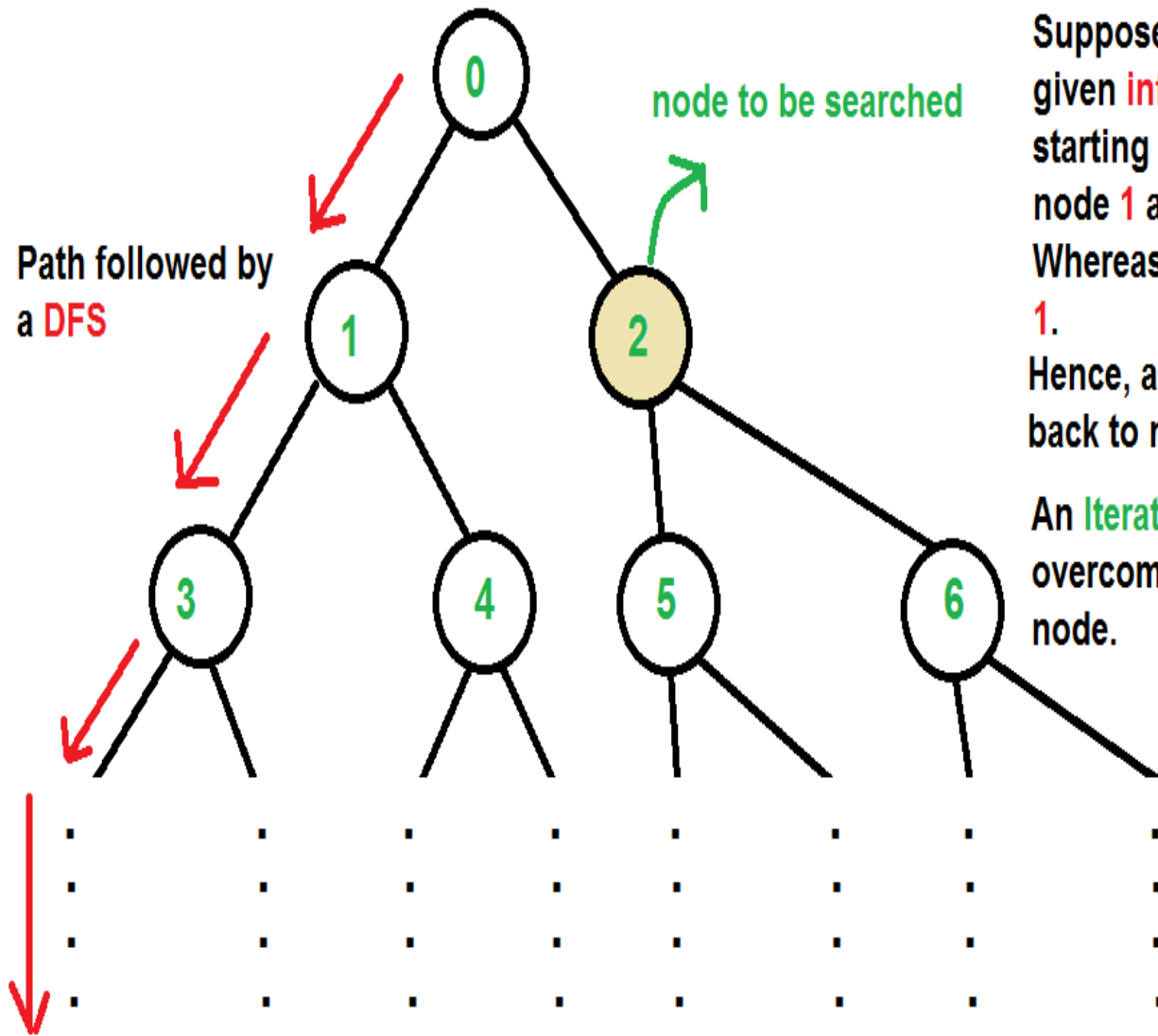
Example...

- Need to go to Hyderabad from Tirupati.
- Have a map with you, Total of 8 cities
- Then Maximum length should be 7 only.

Problem with this is....

If we choose a depth limit that is too small, then depth-limited search is not even complete.





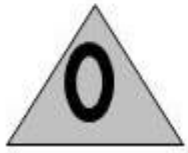
Suppose, we want to find node- '2' of the given **infinite** undirected graph/tree. A **DFS** starting from node- 0 will dive left, towards node 1 and so on. Whereas, the node 2 is just adjacent to node 1. Hence, a DFS wastes a lot of time in coming back to node 2.

An **Iterative Deepening Depth First Search** overcomes this and quickly find the required node.

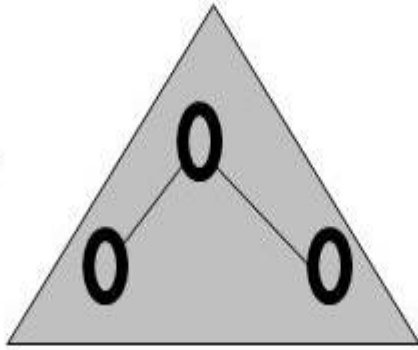


ITERATIVE DEEPENING SEARCH

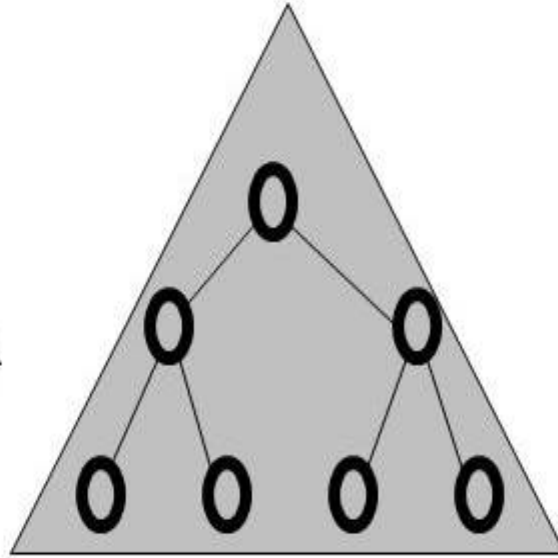
- Combines the space efficiency of depth-first search with the optimality of breadth-first methods
- Idea is to recompute the elements rather than storing them
- recomputation can be a depth-first search, which thus uses less space.



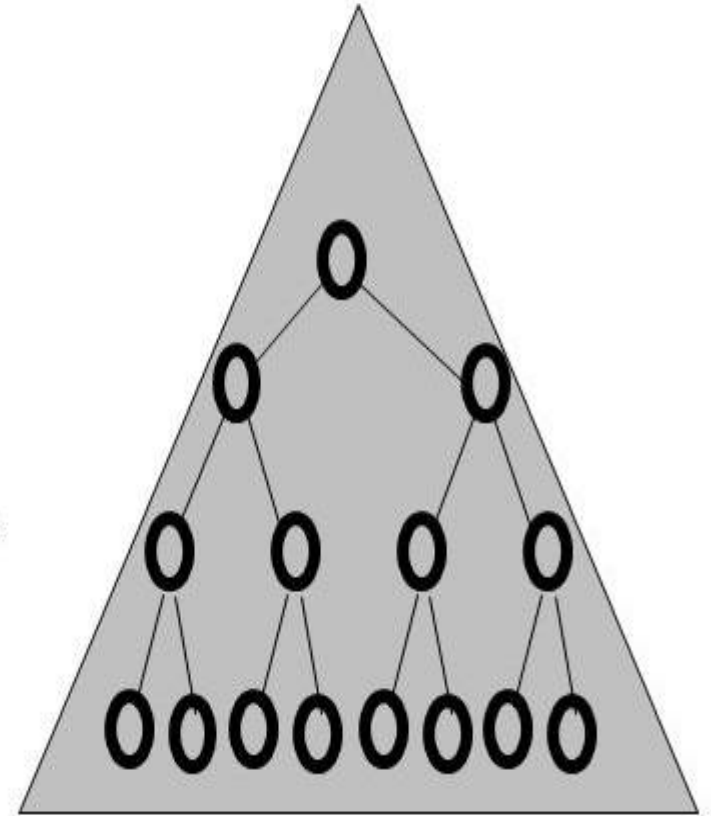
$L = 0$



$L = 1$



$L = 2$



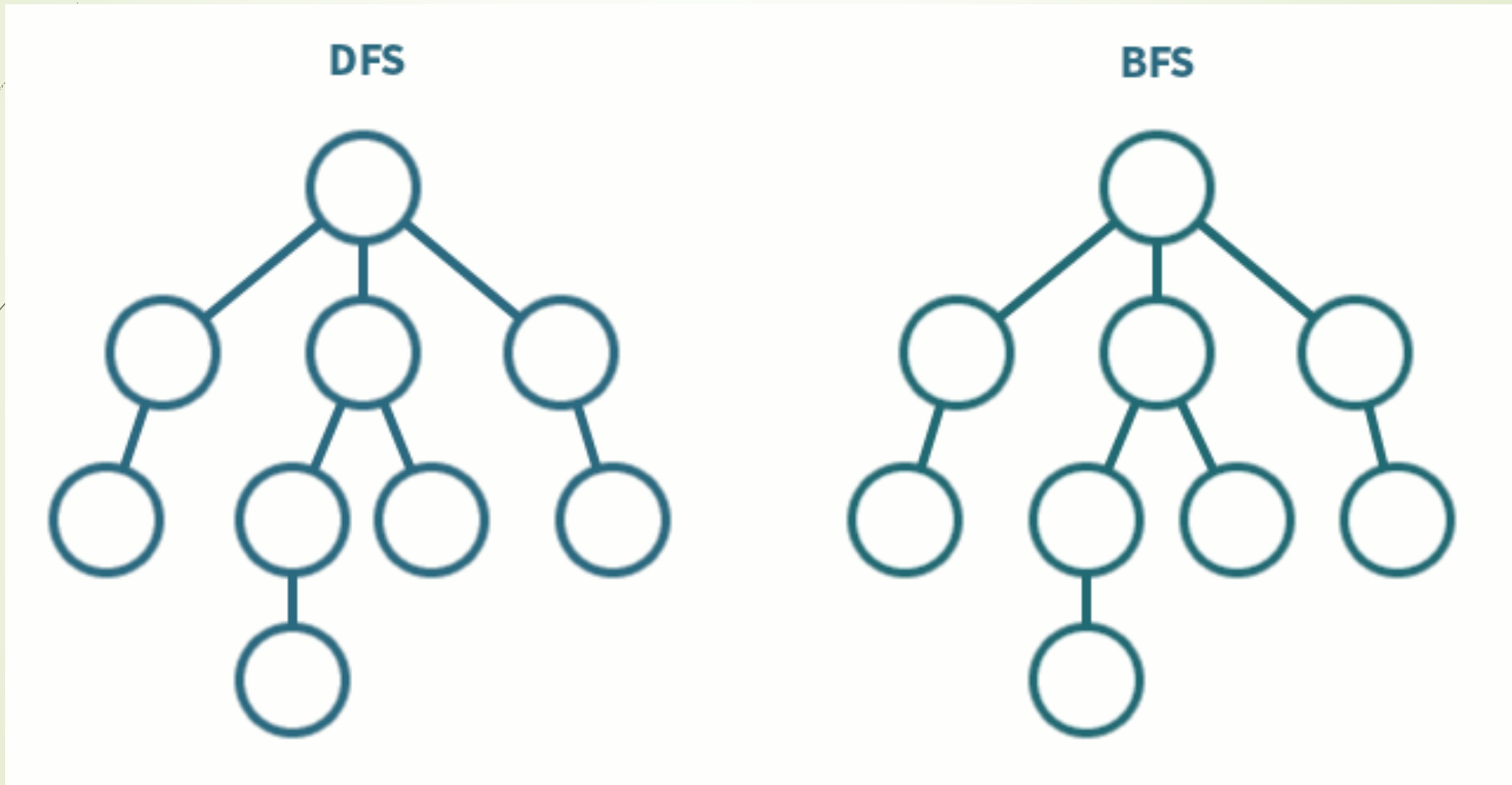
$L = 3$



ITERATIVE DEEPENING SEARCH

- optimal and complete, like breadth-first search
- modest memory requirements of depth-first search
- Preferred when there is a large search space and the depth of the solution is not known.

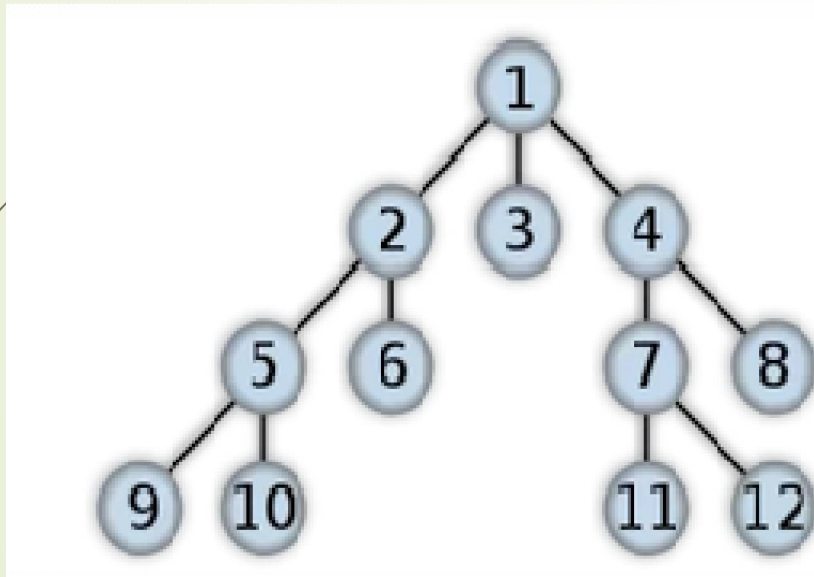
Comparison:



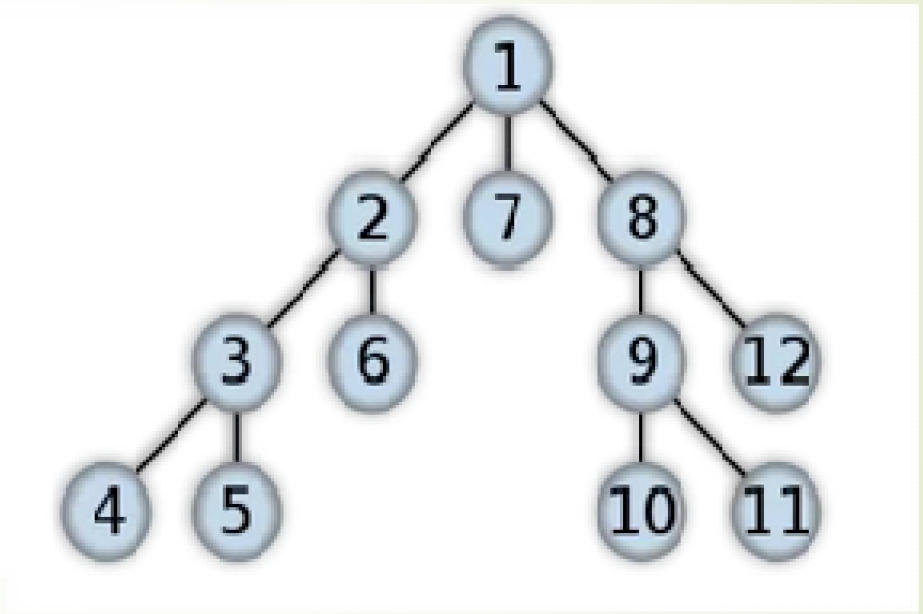
STACK is used

QUEUE is used

BFS



DFS





Evaluation criteria for the search strategies:

- **Completeness:** Is the solution guaranteed?
- **Time complexity:** How long it takes to find the solution?
- **Space complexity:** How much memory it requires?
- **Optimality:** Can it find the optimal solution when multi solution exist?

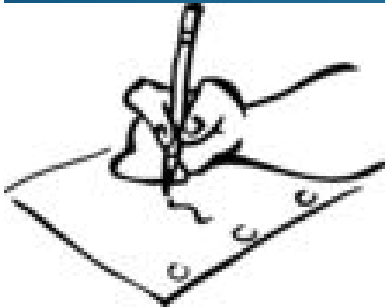
Comparison table:

CRITERIA	Breadth first	Depth first	Depth limited	Iterative Deepening
TIME	b^d	b^m	b^l	b^d
SPACE	b^d	bm	bl	bd
Optimal?	Yes	No	No	Yes
Complete?	Yes	No	Yes, (if $l > d$)	Yes

Where:

- 'b' is the branching factor; 'd' is depth of solution;
- 'm' is maximum depth of the search tree
- 'l' is the depth limit

Operators & Expressions



by

Dr. M. Lavanya

Assistant Professor(SL)

Department of MCA

Unit: I

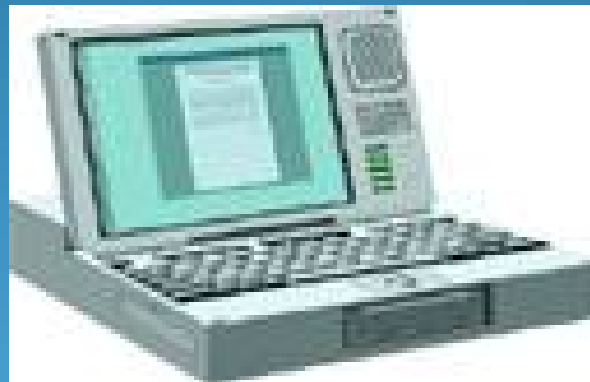
Course: **Programming in C**

Target Group: **MCA, I Sem**

Definition

“An operator is a symbol (+, -, *, /) that directs the computer to perform certain mathematical or logical manipulations and is usually used to manipulate data and variables”

Ex: $a+b$



Operators in C

1. Arithmetic operators
2. Relational operators
3. Logical operators
4. Assignment operators
5. Increment and decrement operators
6. Conditional operators
7. Bitwise operators
8. Special operators





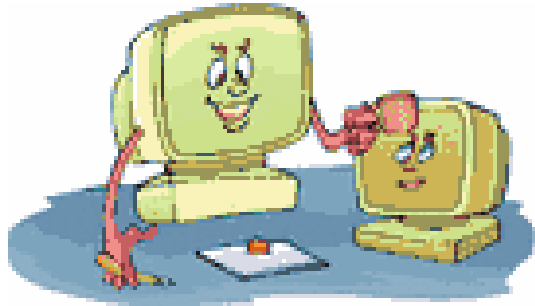
Arithmetic operators

Operator	example	Meaning
+	$a + b$	Addition –unary
-	$a - b$	Subtraction- unary
*	$a * b$	Multiplication
/	a / b	Division
%	$a \% b$	Modulo division- remainder



Relational Operators

Operator	Meaning
<	Is less than
<=	Is less than or equal to
>	Is greater than
>=	Is greater than or equal to
==	Equal to
!=	Not equal to



Logical Operators

Operator	Meaning
&&	Logical AND
	Logical OR
!	Logical NOT

Logical expression or a compound relational expression-

An expression that combines two or more relational expressions

Ex: if (a==b && b==c)



Truth Table

a	b	Value of the expression	
		a && b	a b
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Assignment operators

Syntax:

$v \text{ op} = \text{exp};$

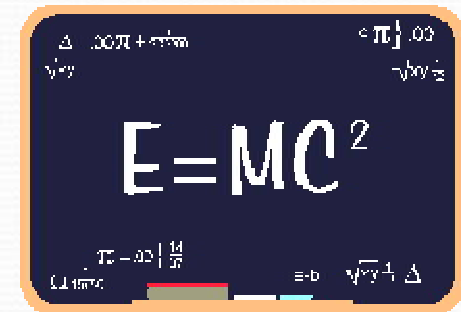
Where v = variable,

op = shorthand assignment operator

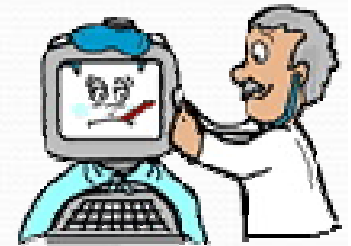
exp = expression

Ex: $x = x + 3$

$x += 3$



Shorthand Assignment operators



Simple assignment operator	Shorthand operator
$a = a + 1$	$a + = 1$
$a = a - 1$	$a - = 1$
$a = a * (m+n)$	$a * = m+n$
$a = a / (m+n)$	$a / = m+n$
$a = a \% b$	$a \% = b$

Increment & Decrement Operators

C supports 2 useful operators namely

1. Increment ++
2. Decrement – operators

The ++ operator adds a value 1 to the operand

The – operator subtracts 1 from the operand

++a or a++

--a or a--





Rules for ++ & -- operators

1. These require variables as their operands
2. When postfix either ++ or – is used with the variable in a given expression, the expression is evaluated first and then it is incremented or decremented by one
3. When prefix either ++ or – is used with the variable in a given expression, it is incremented or decremented by one first and then the expression is evaluated with the new value

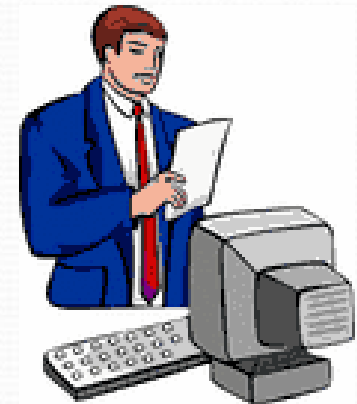
Examples for ++ & -- operators

Let the value of $a = 5$ and $b = ++a$ then
 $a = b = 6$

Let the value of $a = 5$ and $b = a++$ then
 $a = 5$ but $b = 6$

i.e.:

1. a prefix operator first adds 1 to the operand and then the result is assigned to the variable on the left
2. a postfix operator first assigns the value to the variable on left and then increments the operand.



Conditional operators

Syntax:

$\text{exp1} ? \text{exp2} : \text{exp3}$

Where exp1 , exp2 and exp3 are expressions

Working of the ? Operator:

Exp1 is evaluated first, if it is nonzero(1/true) then the expression2 is evaluated and this becomes the value of the expression,

If exp1 is false(0/zero) exp3 is evaluated and its value becomes the value of the expression

Ex: $m=2;$

$n=3$

$r=(m>n) ? m : n;$



Bitwise operators

These operators allow manipulation of data at the bit level

Operator	Meaning
&	Bitwise AND
	Bitwise OR
^	Bitwise exclusive OR
<<	Shift left
>>	Shift right



Special operators

1. Comma operator (,)
2. sizeof operator – sizeof()
3. Pointer operators – (& and *)
4. Member selection operators – (. and ->)



Arithmetic Expressions

Algebraic expression	C expression
$axb-c$	$a*b-c$
$(m+n)(x+y)$	$(m+n)*(x+y)$
$\left[\frac{ab}{c} \right]$	$a*b/c$
$3x^2+2x+1$	$3*x*x+2*x+1$
$\frac{a}{b}$	a/b
$S = \frac{a+b+c}{2}$	$S=(a+b+c)/2$

Arithmetic Expressions

Algebraic expression	C expression
$\text{area} = \sqrt{s(s-a)(s-b)(s-c)}$	$\text{area} = \text{sqrt}(s*(s-a)*(s-b)*(s-c))$
$\text{Sin} \left(\frac{b}{\sqrt{a^2 + b^2}} \right)$	$\text{sin}(b/\text{sqrt}(a*a+b*b))$
$\tau_1 = \sqrt{\left\{ \frac{\sigma_x - \sigma_y}{2} \right\} + \tau xy^2}$	$\text{tow1} = \text{sqrt}((\text{rowx} - \text{rowy})/2 + \text{tow} * \text{x} * \text{y} * \text{y})$
$\tau_1 = \sqrt{\left\{ \frac{\sigma_x - \sigma_y}{2} \right\}^2 + \tau xy^2}$	$\text{tow1} = \text{sqrt}(\text{pow}((\text{rowx} - \text{rowy})/2, 2) + \text{tow} * \text{x} * \text{y} * \text{y})$
$y = \frac{\alpha + \beta}{\sin \theta} + x $	$\text{y} = (\text{alpha} + \text{beta}) / \text{sin}(\text{theta} * 3.1416 / 180) + \text{abs}(\text{x})$



Precedence of operators

BODMAS RULE-

Brackets **o**f **D**ivision **M**ultiplication **A**ddition **S**ubtraction

Brackets will have the highest precedence and have to be evaluated first, then comes of , then comes division, multiplication, addition and finally subtraction.

C language uses some rules in evaluating the expressions and they r called as precedence rules or sometimes also referred to as hierarchy of operations, with some operators with highest precedence and some with least.

The 2 distinct priority levels of arithmetic operators in c are-

Highest priority : * / %

Lowest priority : + -



Rules for evaluation of expression

1. First parenthesized sub expression from left to right are evaluated.
2. If parentheses are nested, the evaluation begins with the innermost sub expression
3. The precedence rule is applied in determining the order of application of operators in evaluating sub expressions
4. The associatively rule is applied when 2 or more operators of the same precedence level appear in a sub expression.
5. Arithmetic expressions are evaluated from left to right using the rules of precedence
6. When parentheses are used, the expressions within parentheses assume highest priority



Hierarchy of operators

Operator	Description	Associativity
(), []	Function call, array element reference	Left to Right
+, -, ++, -- ,!,~,*,&	Unary plus, minus, increment, decrement, logical negation, 1's complement, pointer reference, address	Right to Left
*, /, %	Multiplication, division, modulus	Left to Right

Example 1

$$\begin{aligned} &\text{Evaluate } x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \text{ @ } a=1, b=-5, c=6 \\ &= \frac{-(-5) + \sqrt{(-5)(-5) - 4 \cdot 1 \cdot 6}}{2 \cdot 1} \\ &= \frac{5 + \sqrt{(-5)(-5) - 4 \cdot 1 \cdot 6}}{2 \cdot 1} \\ &= \frac{5 + \sqrt{25 - 4 \cdot 1 \cdot 6}}{2 \cdot 1} \\ &= \frac{5 + \sqrt{25 - 4 \cdot 6}}{2 \cdot 1} \\ &= \frac{5 + \sqrt{25 - 24}}{2 \cdot 1} \\ &= \frac{5 + \sqrt{1}}{2 \cdot 1} \\ &= \frac{5 + 1.0}{2 \cdot 1} \\ &= \frac{6.0}{2 \cdot 1} \\ &= 6.0/2 = 3.0 \end{aligned}$$

Example 2

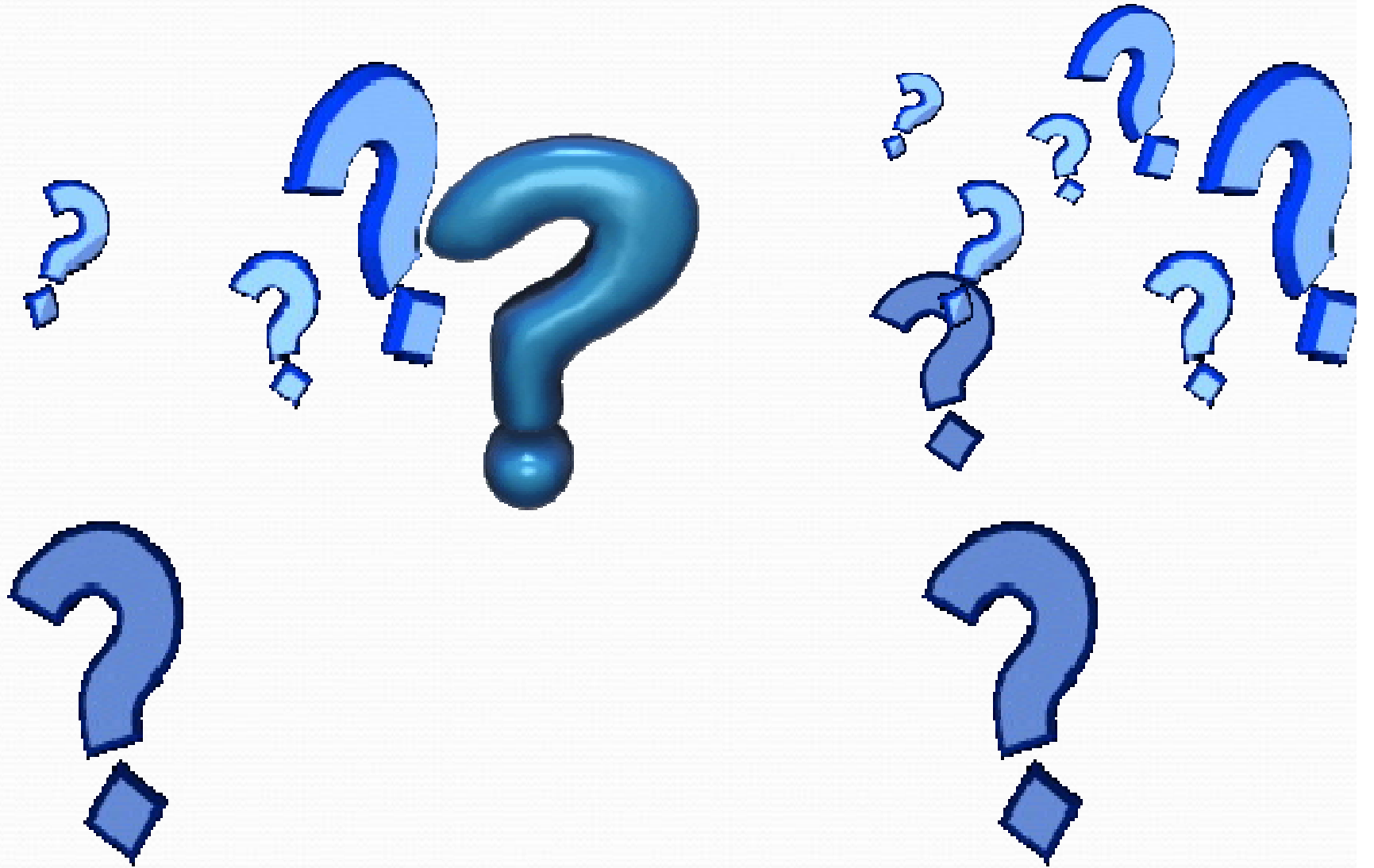
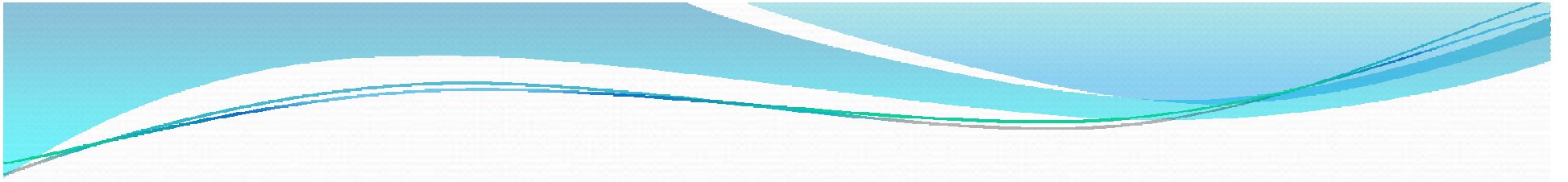
Evaluate the expression when $a=4$

$$b = a - 5 + a$$

$$= a - 5 + a$$

$$= 5 - 5$$

$$= 0$$



NANOMATERIALS

by

Dr. G. Madhu

Assistant Professor, Dept. of BS&H

Unit: **III**

Course: **Engineering Chemistry**

Target Group: **I. B.Tech.**

The study of the controlling of matter on an **atom** and **molecular** scale.

Generally nanotechnology deals with structures sized between **1 to 100 nanometer** in at least one dimension, and involves developing or modifying materials or devices within that size.

The term "**Nano**" stands for **1 billionth** of a meter in a physical scale length.

Nano chemistry is defined as ' the study of synthesis and analysis of materials in nano scale range including large organic molecules, inorganic cluster compounds and metallic or non metallic semiconductor particles'.

What is Nanoscale

$$1 \text{ nm} = 10^{-9} \text{ m}$$



12,756 Km

$$1.27 \times 10^7 \text{ m}$$

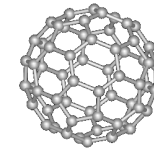
www.mathworks.com



22 cm

$$0.22 \text{ m}$$

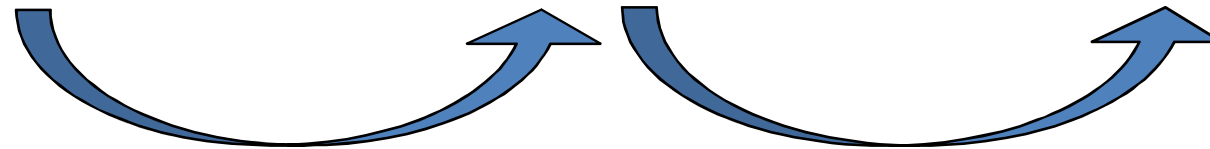
Fullerenes C_{60}



0.7 nm

$$0.7 \times 10^{-9} \text{ m}$$

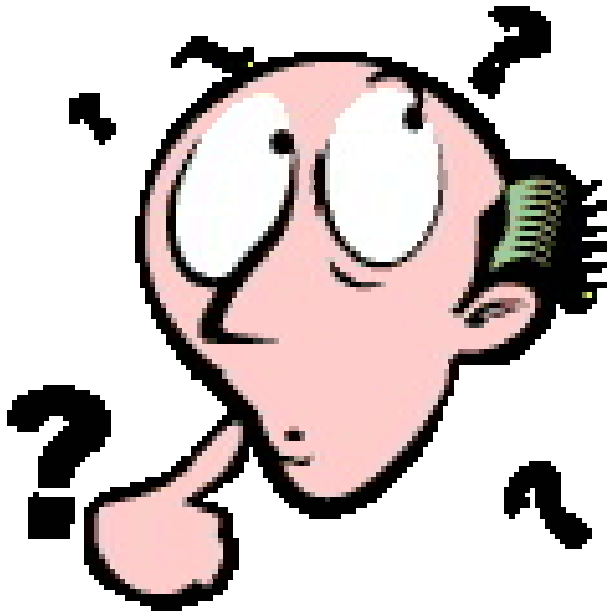
www.physics.ucr.edu



10 millions times
smaller

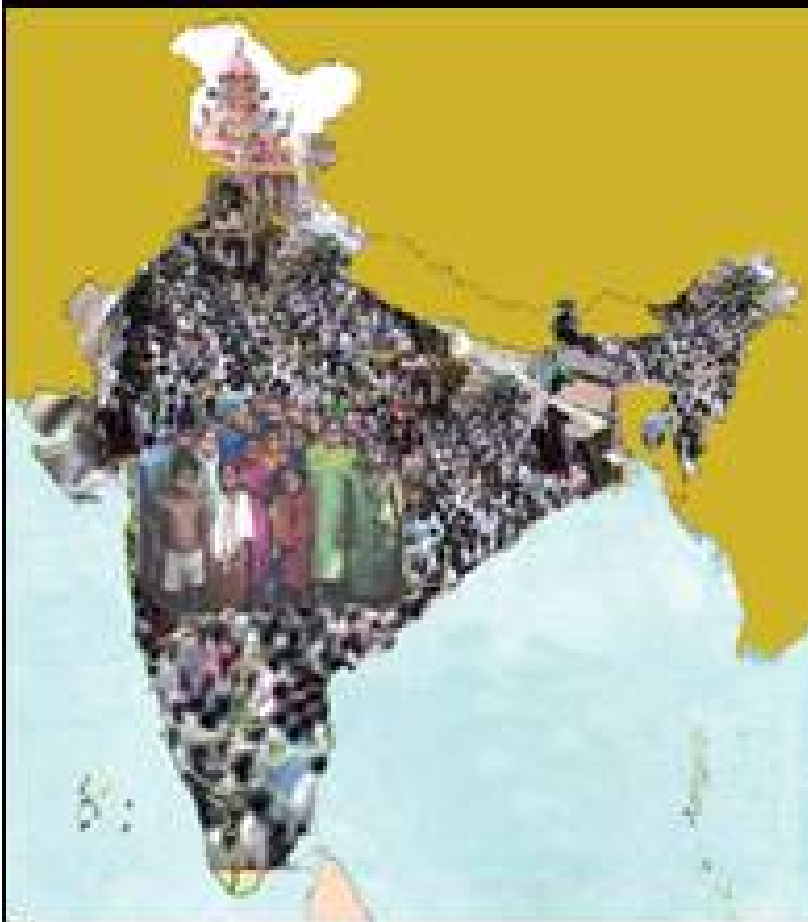
1 billion times
smaller

Understanding Size



How big (small) are we
talking about?

BEST COMPARSION



A few more familiar examples may convince you of the difficulty in imagining the size of nano-objects.

The population of India is one billion or 100 crores. Each Indian - you or me - is nano in comparison with the total population of India.





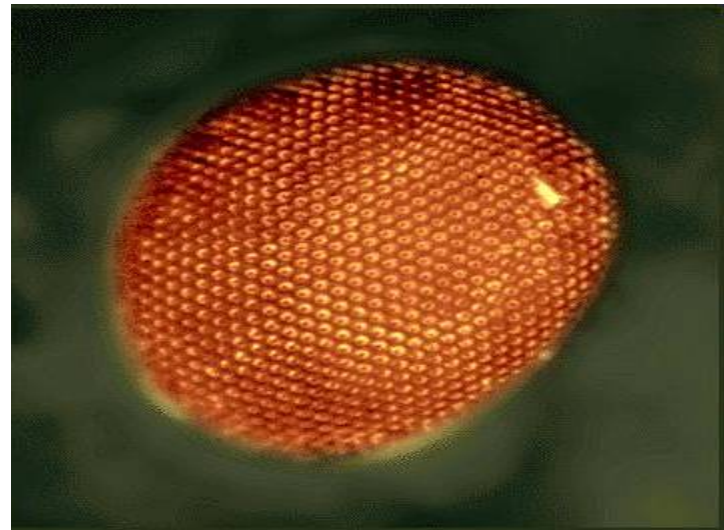
1 meter



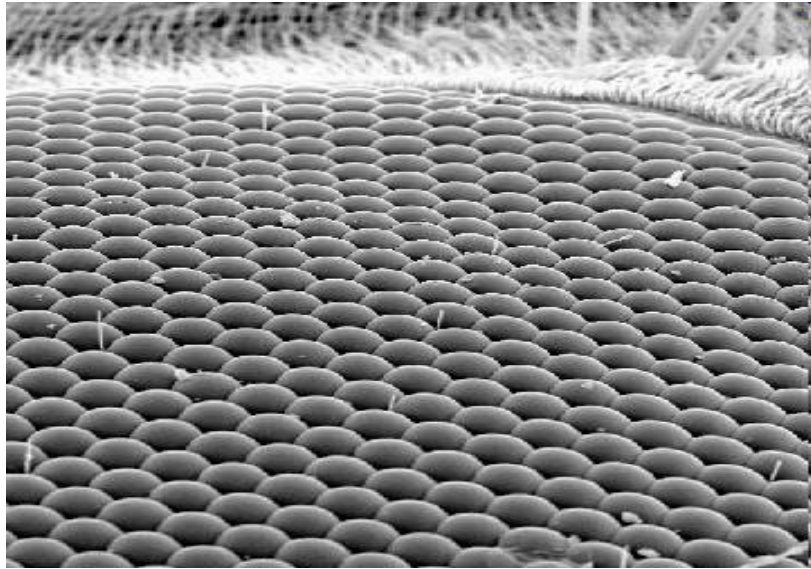
10 centimeters



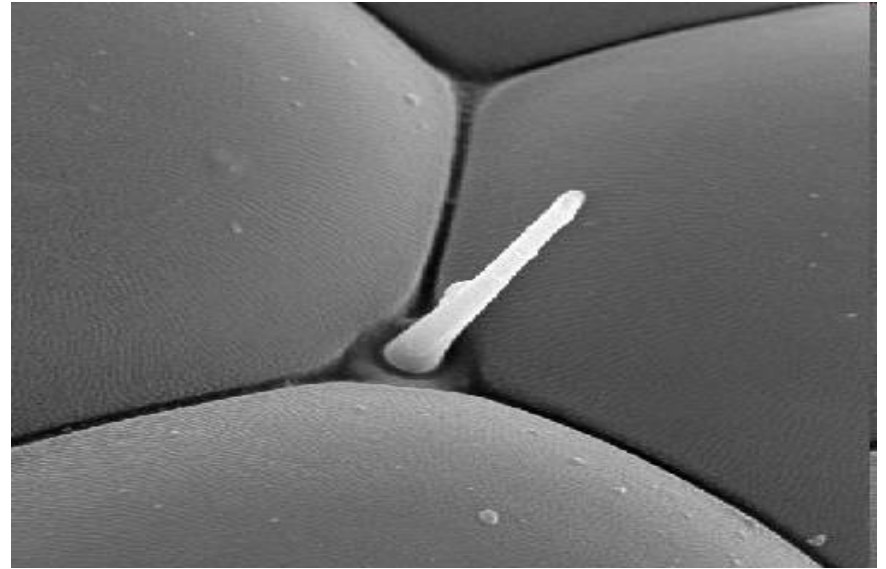
1 centimeter



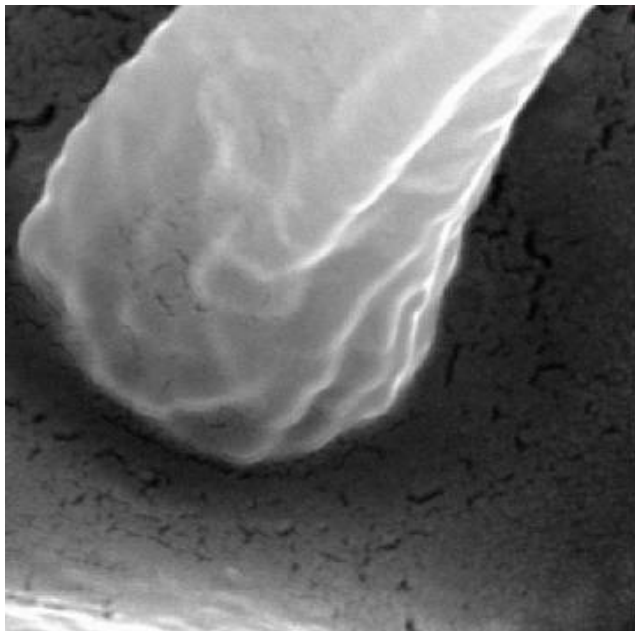
100 micrometers



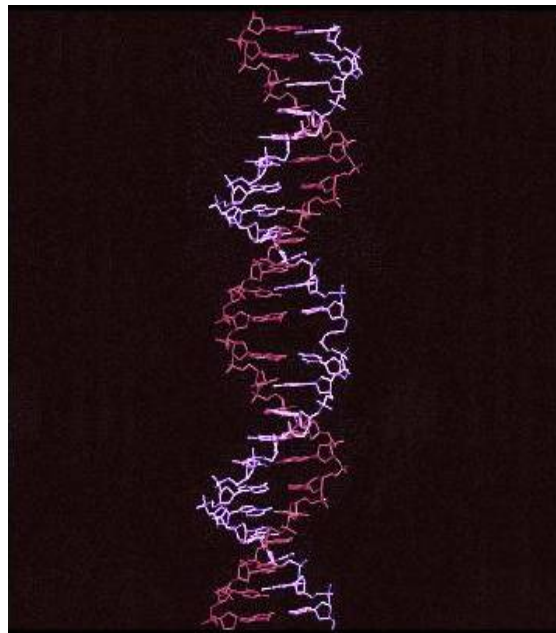
10 micrometers



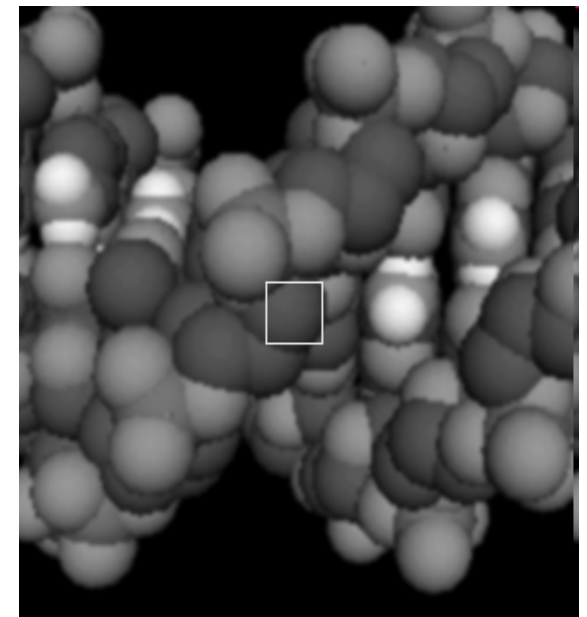
1 micrometer



100 nanometers



10 nanometers



1 nanometer

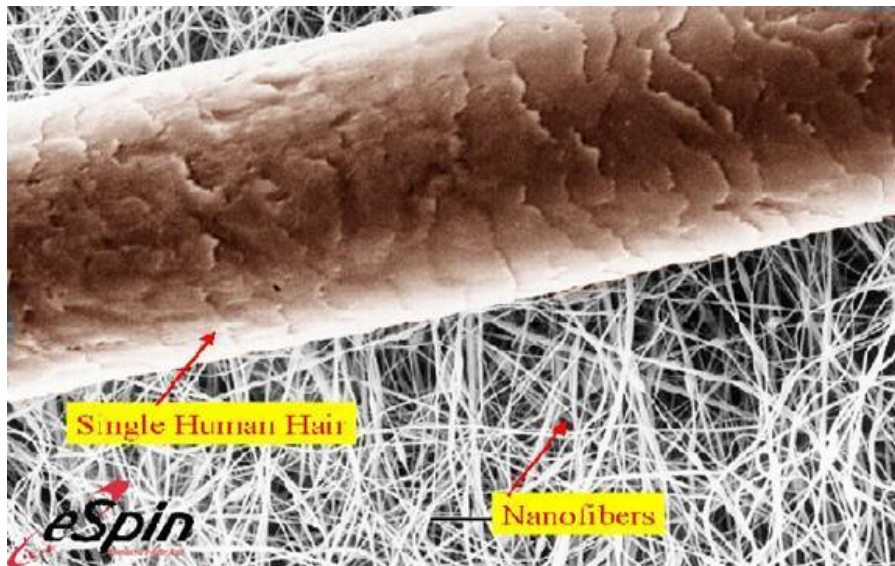
A nanometre is the width of 10 hydrogen atoms lined up side-by-side.

To put this in perspective, a human hair is about 80,000 nanometres wide;

a red blood cell is about 7,000 nanometres in diameter

Human Hair Greater than 80,000nm

Human Red Blood Cells=8000nm



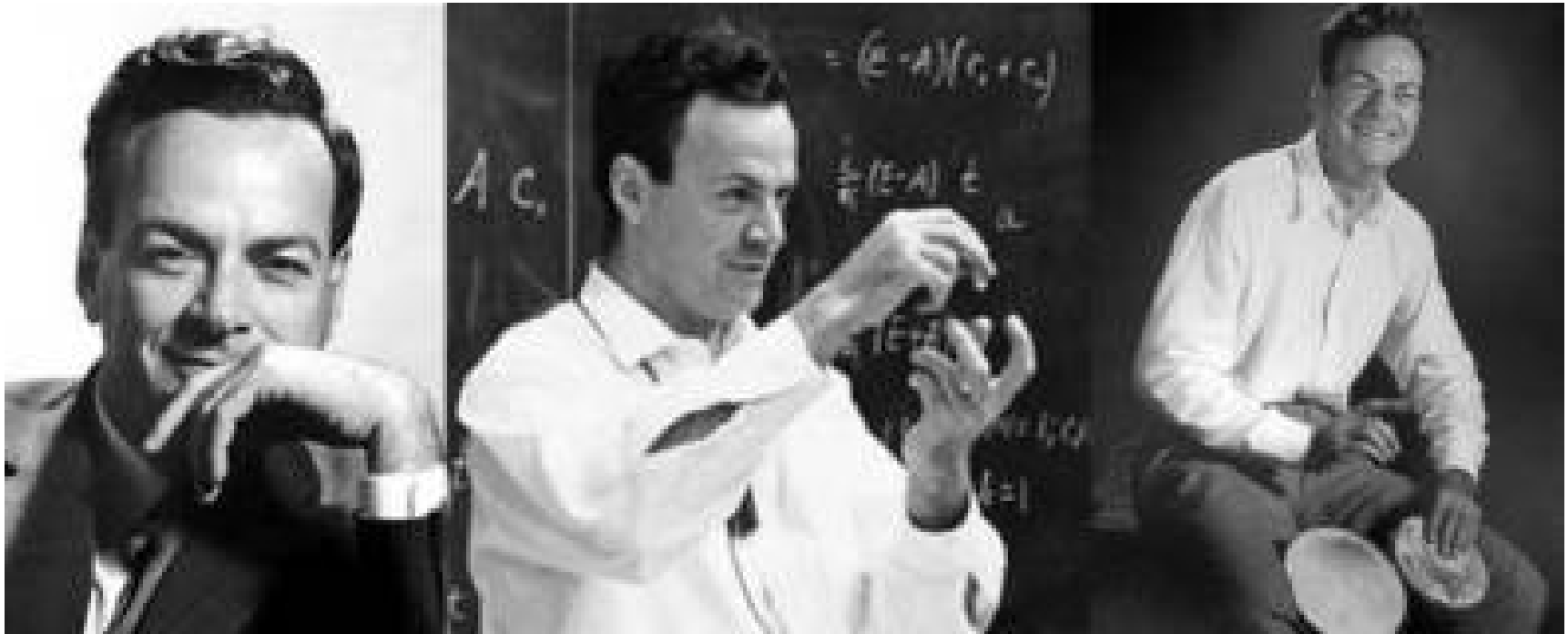
History of Nanotechnology

The use of nanomaterials had been recognized even in the fourth century A.D Roman glass markers fabricated coloured articles of glass, embedded with nano particles of silver and gold.

- ~ **2000 Years Ago** – Sulfide nanocrystals used by Greeks and Romans to dye hair
- ~ **1000 Years Ago (Middle Ages)** – Gold nanoparticles of different sizes used to produce different colors in stained glass windows

“There is plenty of room at the bottom”

by R. Feynman



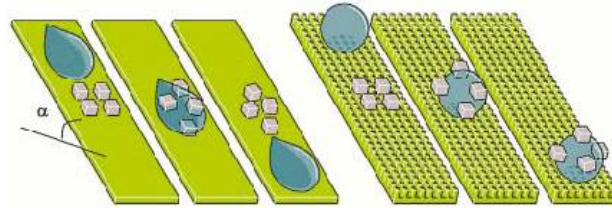
The first ever concept was presented in 1959 by the famous Professor of physics **Dr. Richard P. Feynman**

LOTUS – SELF CLEANING



Water lily

Super hydrophobicity
- Self cleaning



Surface roughness – Self cleaning



SEM



Rinsing with water

Applications -



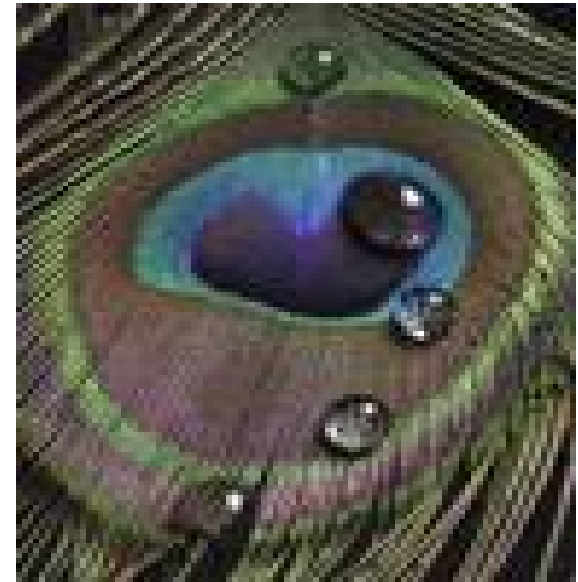
Water drops – clean surface



PEACOCK FEATHER

- Nano structured particles

- Cleaning surface



Applications



- Colour never fade
- Scratch free



Major aspects for the development of Nano technology:

- Synthesis of fullerenes(C_{60})
- Invention of Scanning Tunnelling Microscope (STM) and Atomic force microscope (AFM) for viewing.
- Characterisation and Manipulation of nano structures.
- Synthesis of carbon nanotubes, development photonic crystals in 1990s.

Some of the examples of the nanomaterials are:

- ✓ Nano particles of ceramic oxides, semi conductors
- ✓ Nano crystals and clusters of metals , quantum dots
- ✓ Nano wires and rods
- ✓ Nano porous solids of zeolites and phosphate.

CLASSIFICATION OF NANOMATERIALS

- ❖ Materials with **one** dimension in the nano scale.
Eg: Nano wires, nano tubes, bio polymers
- ❖ Materials with **two** dimension in the nano scale.
Eg: Thin films
- ❖ Materials with **three** dimension in nano scale.
Eg: fullerenes

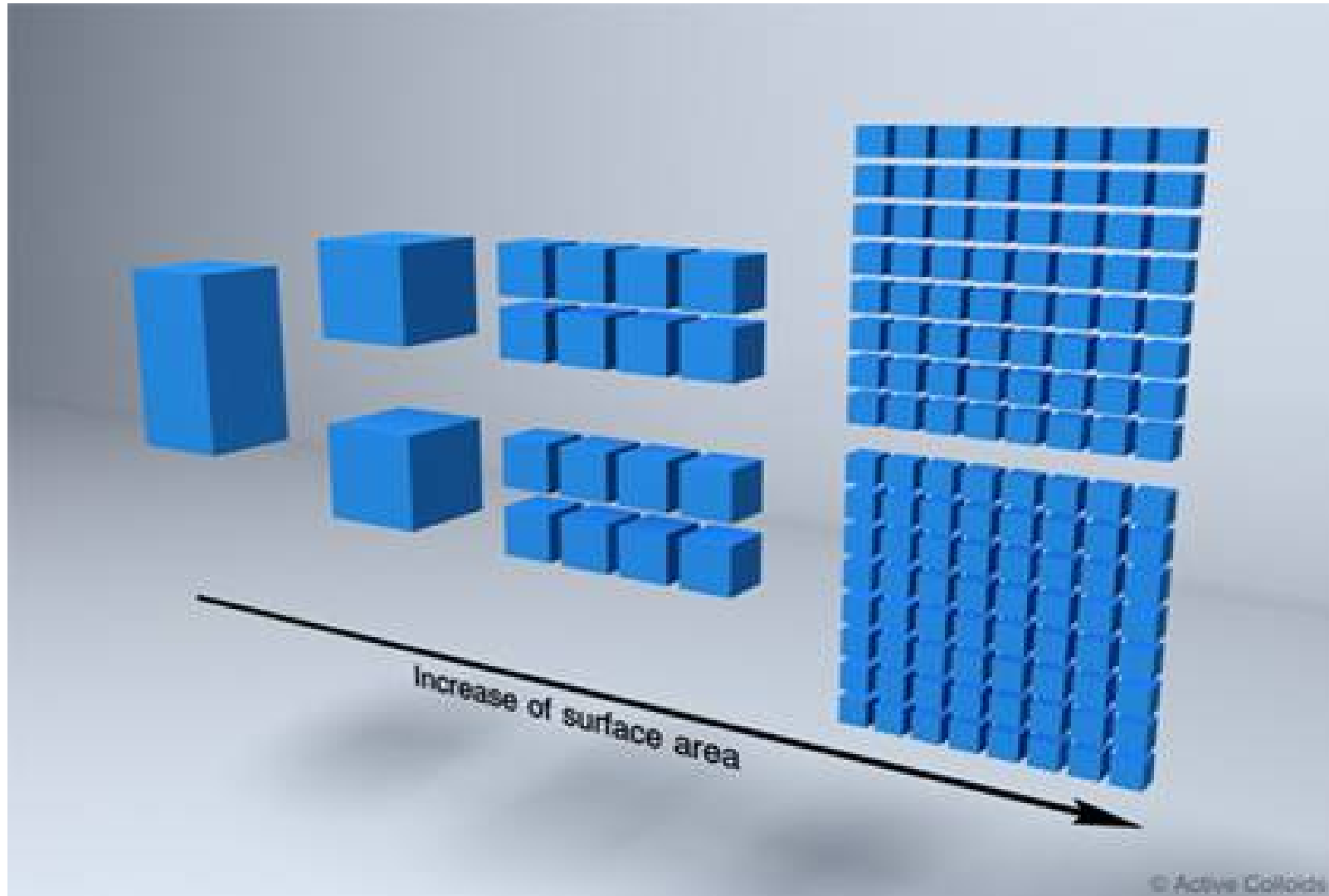
PROPERTIES OF NANOMATERIALS

The two principle factors for the properties of nanomaterials to differ significantly from other materials are:

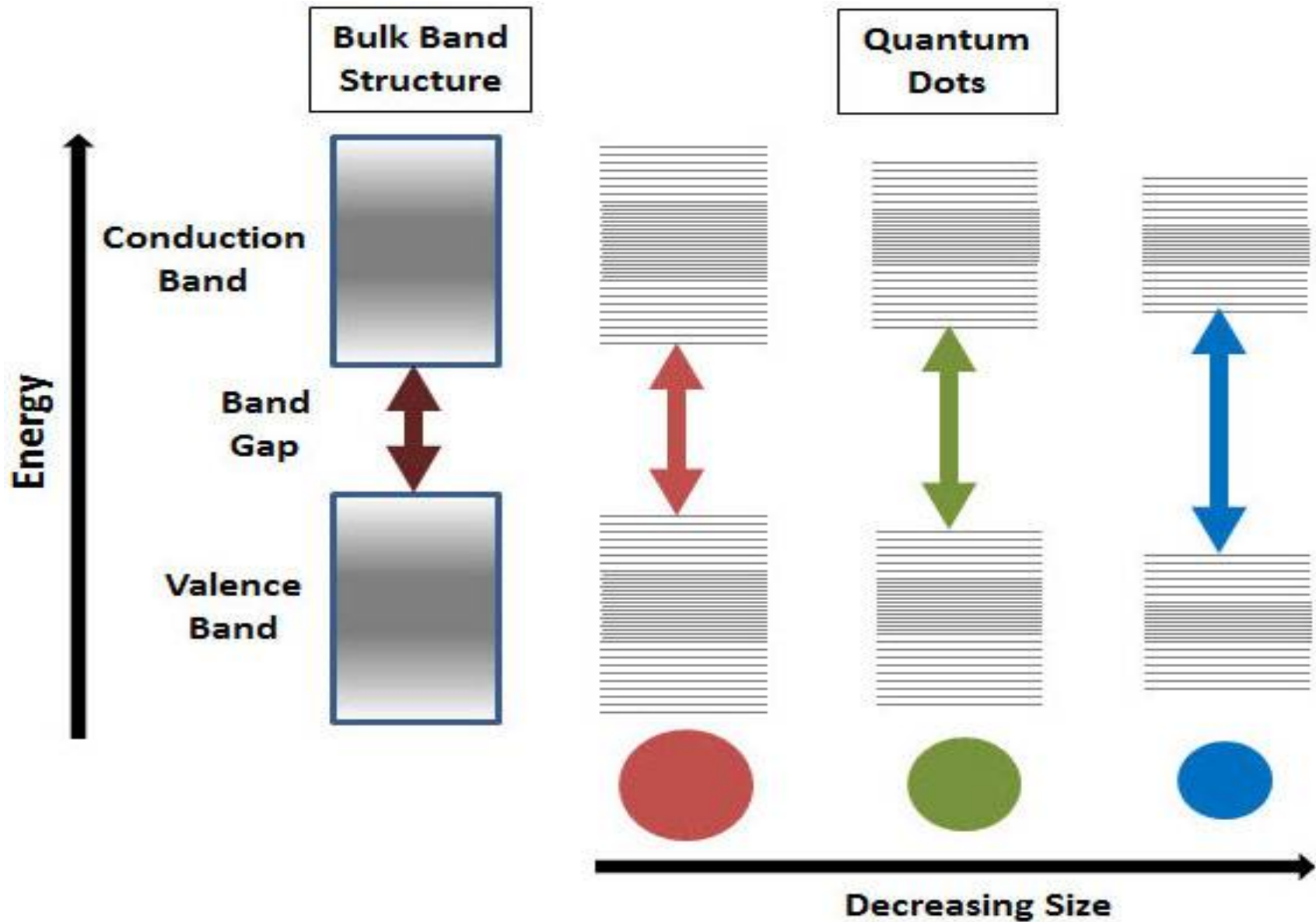
- (i) to increase relative surface area
- (ii) quantum effect

These two principle factors change or enhance properties such as reactivity, strength, electrical and optical properties.

INCREASE SURFACE AREA



QUANTUM CONFINEMENT



Size

Nano crystals have large surface area . Colloidal particles have a large surface area.

Nano crystal of 10nm size—containing 1000 atoms (approx. 15% atoms on its surface)

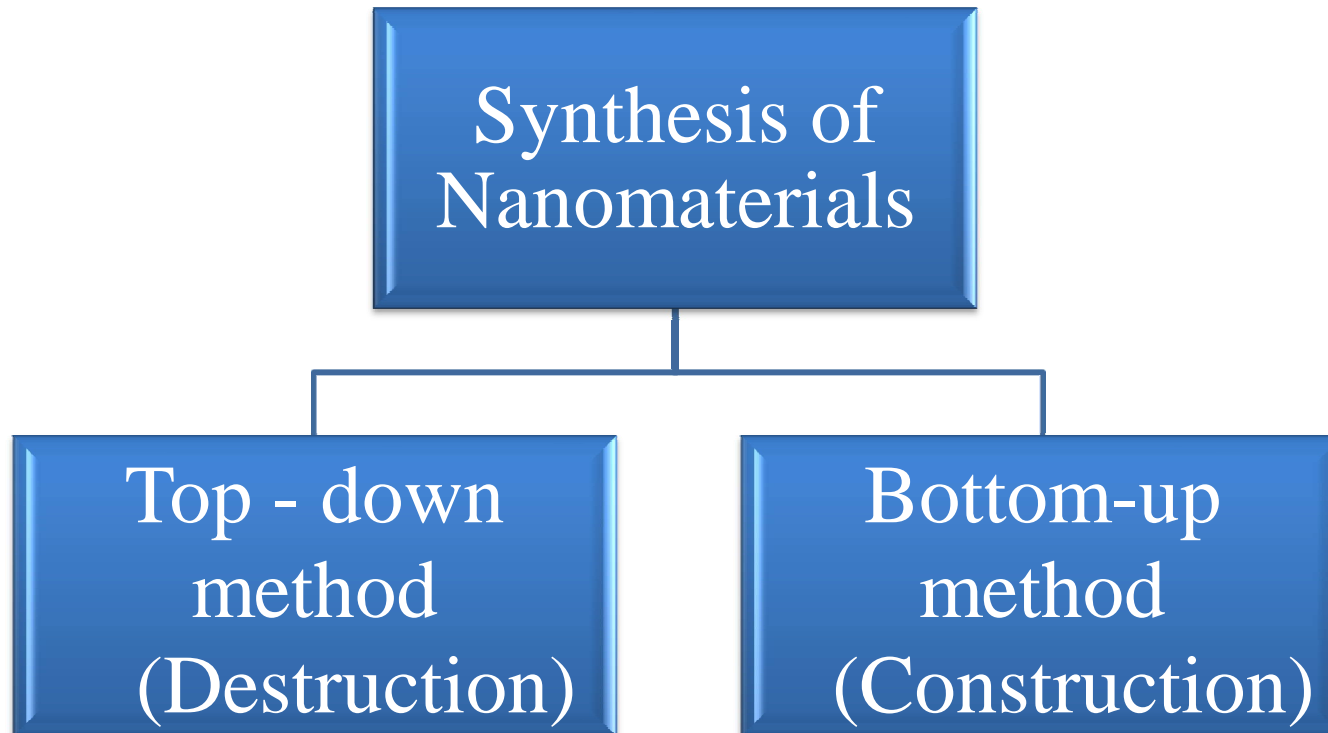
Nano crystal of 1nm size —will be about 30% of its atom on the surface.

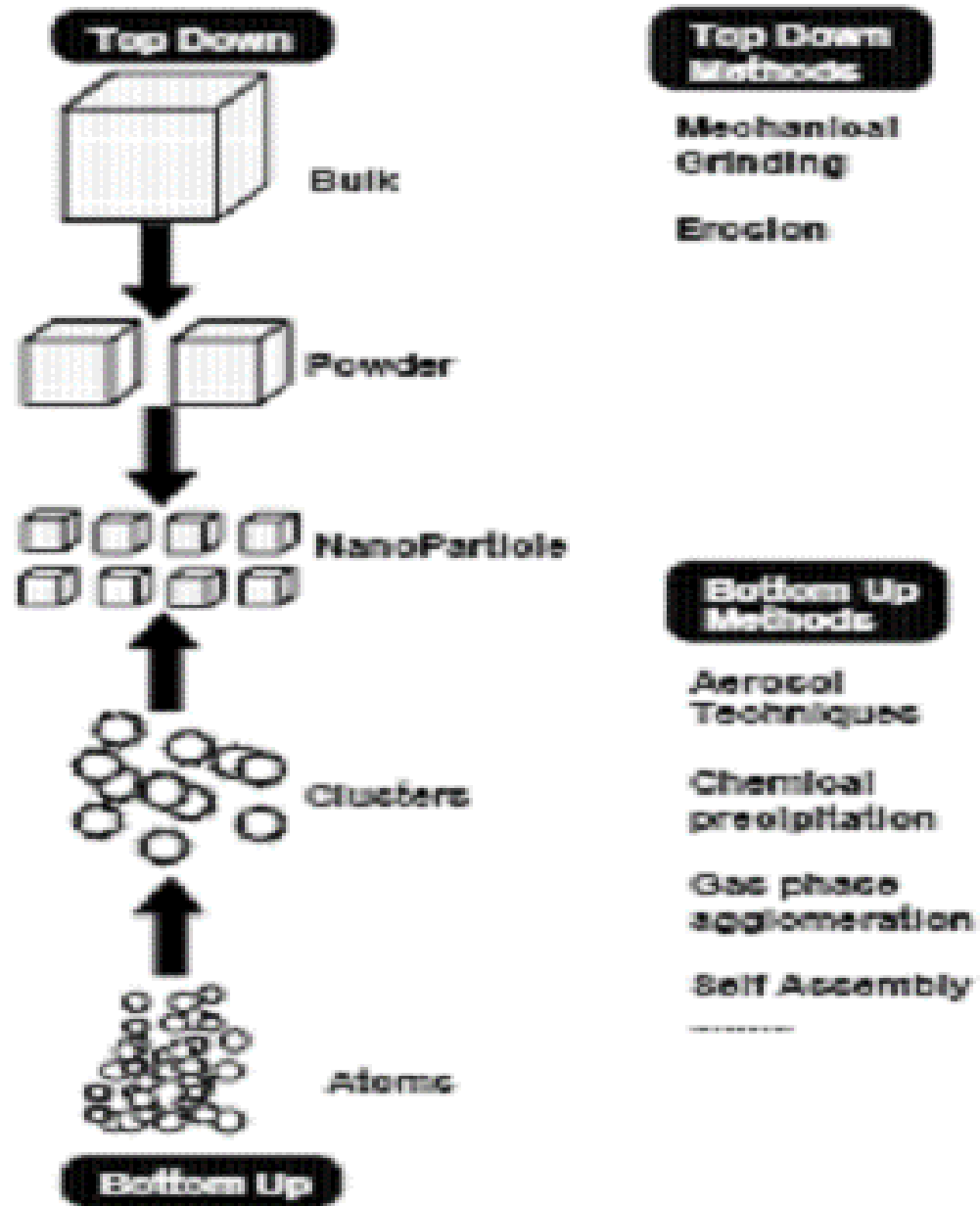
Catalytic activity

Catalytic activity of nano crystals and nano colloids has been investigated wide variety of homo and heterogeneous phases

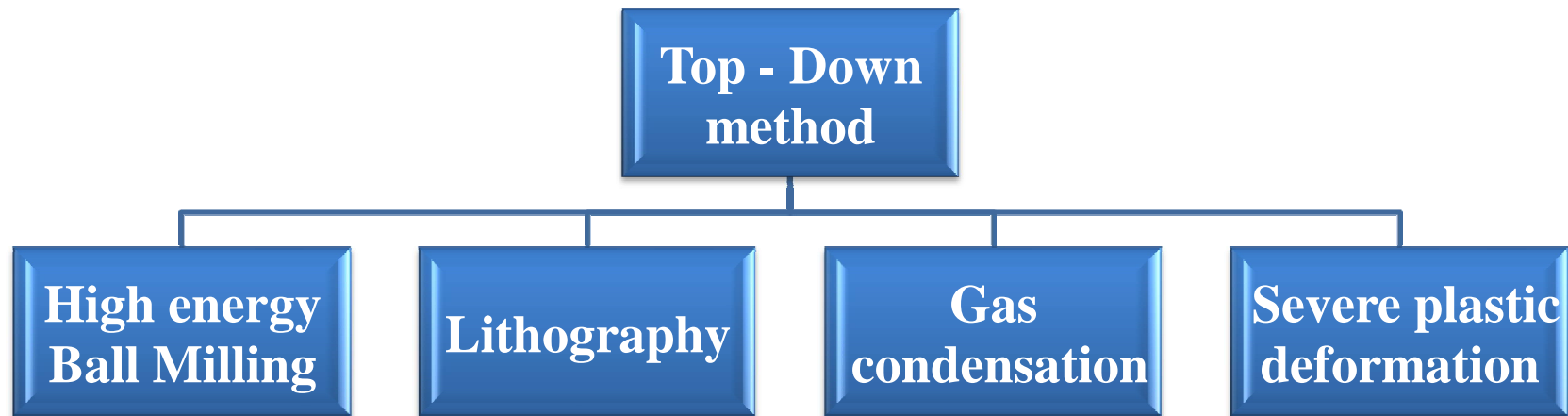
Nano particles	Catalytic reaction
Rhodium hydrosols	Hydrogenation of Olefins dissolved in organic phase
Palladium colloids	Reduction of C-C multiple bonds in presence of formic acid
Cortex Catalyst (Pd nano particle on alumina)	Olefins hydrogenation
MoS ₂	Methanation of Co + H ₂ at low temp.

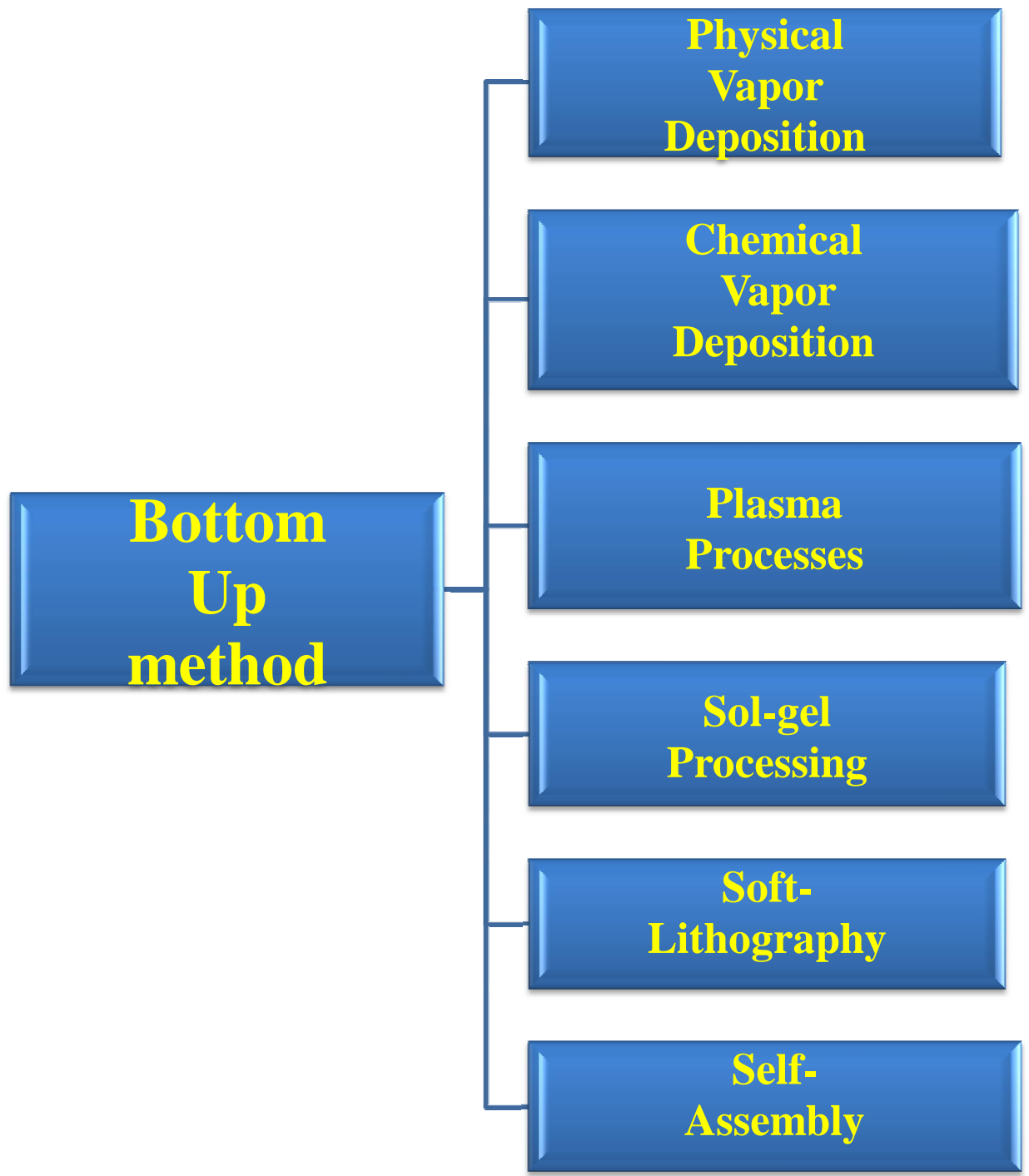
Synthesis of Nanomaterials





Synthesis of Nanomaterials





Top-down vs. bottom-up

- **Top-down methods**

begin with a pattern generated on a larger scale, then reduced to nanoscale.

- By nature, aren't cheap and quick to manufacture
- Slow and not suitable for large scale production.

- **Bottom-up methods**

start with atoms or molecules and build up to nanostructures

- Fabrication is much less expensive

S😊I-Gel Technologies for nanomaterials fabrication



Sol



Gel

🌱 Sol-gel is a chemical solution process used to make ceramic and glass materials in the form of thin films, fibers , or powders .

🌱 A **sol is a colloidal solution-** molecular suspension of solid particles of ions in a solvent.

🌱 A **gel is a semi-rigid mass** that forms when the solvent from the sol begins to evaporate and the particles or ions left behind begin to join together in a continuous network.

The precursors for synthesizing these colloids consist :

Metal alkoxides

e.g: tetramethoxysilane (TMOS) and
tetraethoxysilane (TEOS)

Metal chlorides

They readily react with water.

Sol-gel Processing

➤ The sol-gel process is a wet-chemical technique that uses either a chemical solution (**sol short for solution**) or colloidal particles (sol for nanoscale particle) to produce an **integrated network (gel)**.



➤ **Metal alkoxides** and metal chlorides are typical precursors. They undergo **hydrolysis and polycondensation reactions** to form a colloid, a system composed of nanoparticles dispersed in a solvent. The sol evolves then towards the formation of an inorganic continuous network containing a liquid phase (gel).



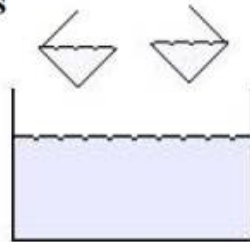
➤ Formation of a metal oxide involves connecting the metal centers with oxo (**M-O-M**), therefore generating **metal-oxo polymers** in solution.



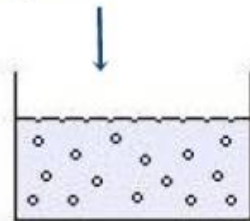
➤ After a drying process, the liquid phase is removed from the gel. Then, a thermal treatment (**calcination**) may be performed in order to favor further polycondensation and enhance mechanical properties.

Making Gel formation

Mix reactives



Hydrolysis and Condensation reactions take place

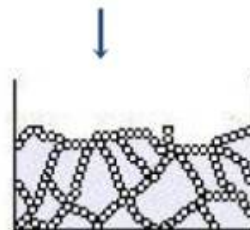


Sol

Gelification



Gel






Hydrolysis



Condensation



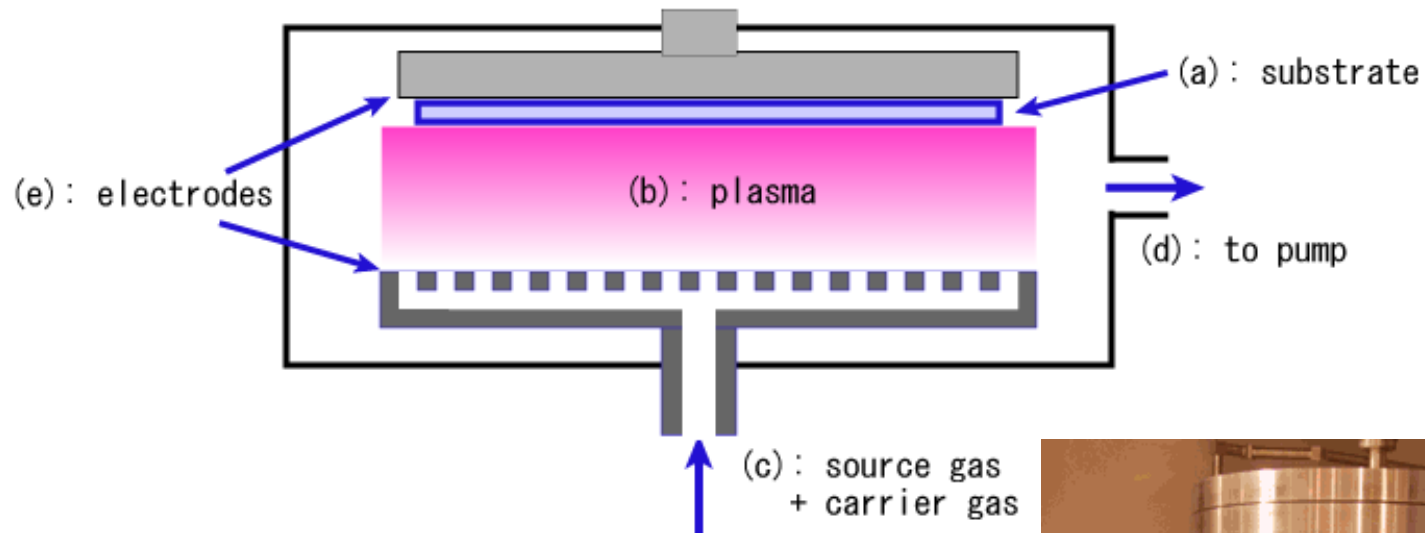
ADVANTAGES OF SOL-GEL PROCESS

-  The sol-gel approach is interesting in that it is a cheap and low-temperature technique that allows for the fine control on the product's chemical composition.
-  Can produce high purity products because the organo-metallic precursor of the desired ceramic oxides can be mixed, dissolved in a specified solvent and hydrolyzed into a sol, and subsequently a gel, the composition can be highly controllable.
-  as even small quantities of dopants, such as organic dyes and rare earth metals, can be introduced in the sol and end up in the final product finely dispersed.

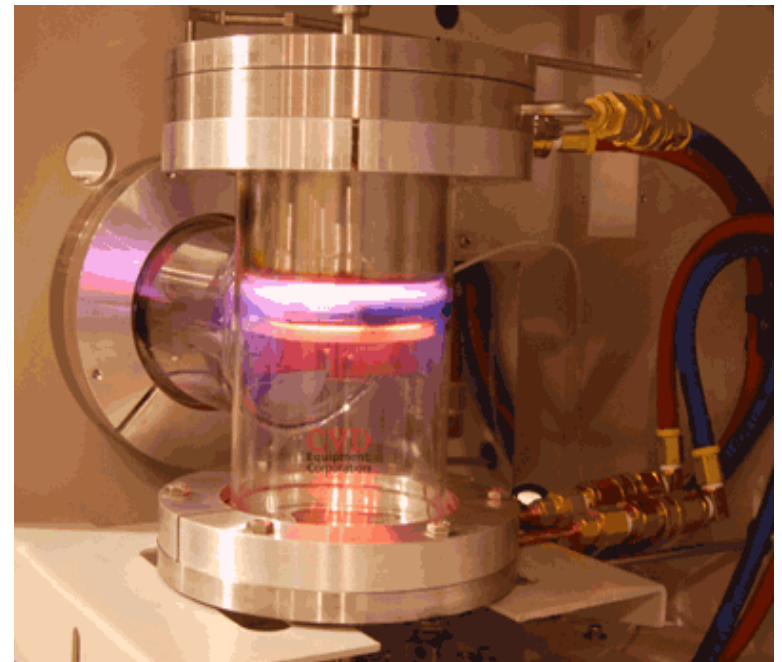
CHEMICAL VAPOUR DEPOSITION

- Chemical vapor deposition (CVD) is a chemical process used to produce high-purity, high-performance solid materials or coatings
- In a typical CVD process, the substrate is exposed to one or more **volatile precursors** which react and decompose on the substrate surface to produce the desired deposit
- **Precursors** include Halides (eg TiCl_4), Hydrides (eg SiH_4) and other compounds etc
- During this process, volatile by-products are also produced, which are removed by gas flow through the reaction chamber.

CHEMICAL VAPOUR DEPOSITION



Plasma enhanced CVD



CHEMICAL VAPOUR DEPOSITION - TYPES

- *Plasma Enhanced CVD (PE-CVD)*
- *Metal Organic CVD (MO-CVD)*
- *Atmospheric pressure CVD (AP-CVD)*
- *Low-pressure CVD (LP-CVD)*
- *Ultrahigh vacuum CVD (UHV-CVD)*
- *Aerosol assisted CVD (AA-CVD)*
- *Direct liquid injection CVD (DLICVD)*

PLASMA ENHANCED CVD

- Plasma-enhanced chemical vapor deposition (PECVD) is a process used to **deposit thin films** from a gas state (vapor) to a solid state on a substrate.
- Chemical reactions are involved in the process, which occur after creation of a plasma of the reacting gases.
- The plasma is generally created by **RF (AC) frequency** or **DC discharge** between two electrodes, the space between which is filled with the reacting gases.
- The helping hand of the Plasma helps in increasing the film quality at low temperature and pressure.

PLASMA.....?

- A plasma can be created by heating a gas or subjecting it to a strong electromagnetic field applied with a laser or microwave generator.
- This decreases or increases the number of electrons, creating positive or negative charged particles called ions, and is accompanied by the dissociation of molecular bonds, if present.

The presence of a significant number of charge carriers makes plasma electrically conductive so that it responds strongly to electromagnetic fields.

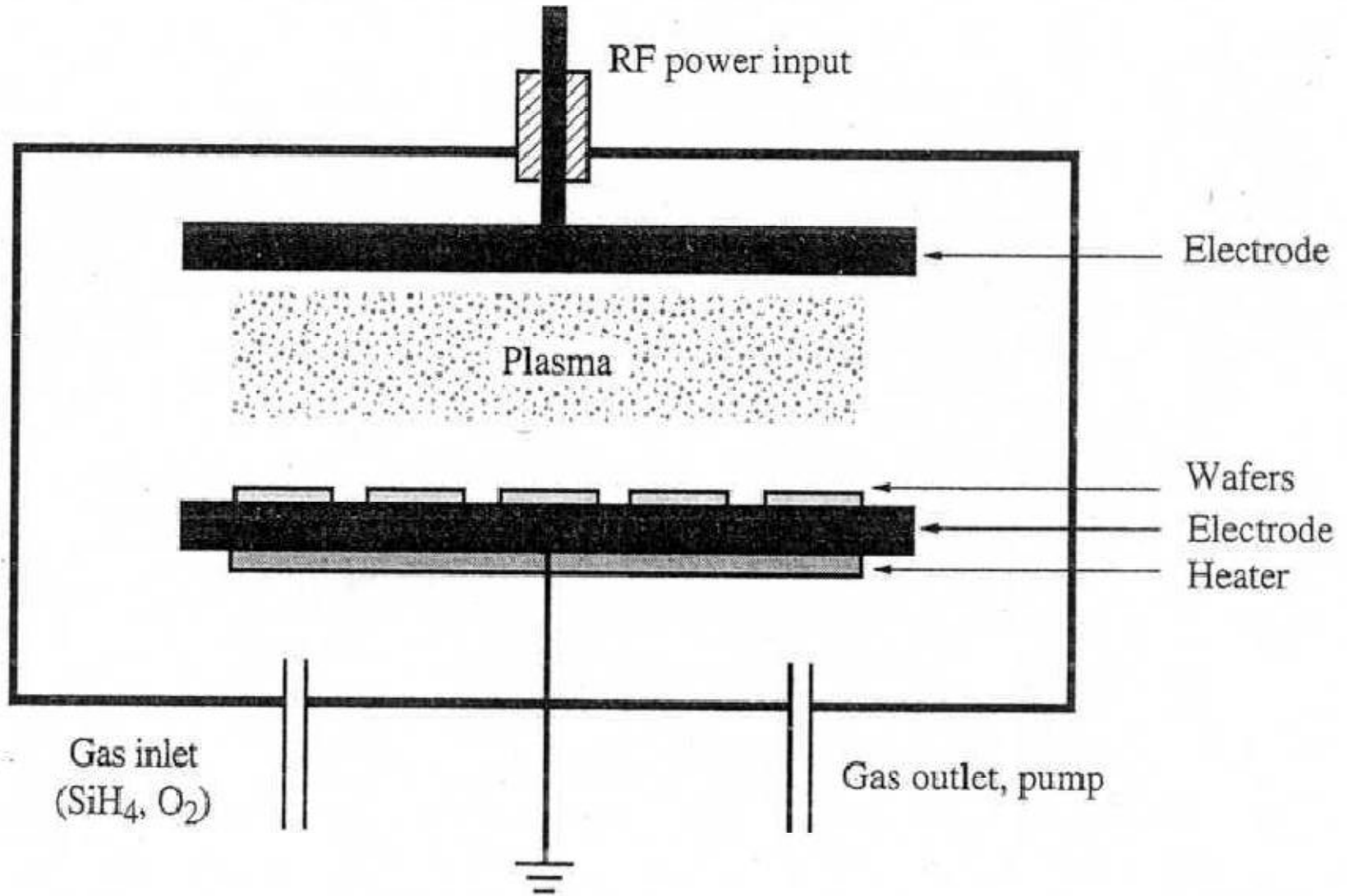
Like gas, plasma does not have a definite shape or a definite volume unless enclosed in a container.

Unlike gas, under the influence of a magnetic field, it may form structures such as filaments, beams and double layers.

CONTINUED..

- PECVD uses electrical energy which is transferred to the gas mixture.
- This transforms the gas mixture into reactive radicals, ions, neutral atoms and molecules, and other highly excited species. ---- **PLASMA**
- These atomic and molecular fragments interact with a substrate and, depending on the nature of these interactions, either etching or deposition processes occur at the substrate.
- Some of the desirable properties of PECVD films are good adhesion, low pinhole density and uniformity.

SCHEMATIC DIAGRAM OF PECVD



Silicon Dioxide: Nano Thin Film



Silicon Nitride:



Metal Nano thin film:



REACTORS USED IN PECVD

- **REINBERG TYPE REACTOR (DIRECT):**

- ❖ Reactants, by-products, substrates and plasma are in the same space.
- ❖ Capacitive-coupled Radio Frequency plasma.
- ❖ Rotating substrates are present.

- **DOWNSTREAM REACTOR (INDIRECT):**

- ❖ Plasma is generated in a separate chamber and is pumped into the deposition chamber.
- ❖ Allows better control of purity and film quality when compared to the Direct type.

ADVANTAGES OF CHEMICAL VAPOUR DEPOSITION

- Variable shaped surfaces, given reasonable access to the coating powders or gases, such as screw threads, blind holes or channels or recesses, can be coated evenly without build-up on edges.
- Versatile –any element or compound can be deposited.
- High Purity can be obtained.
- High Density – nearly 100% of theoretical value.
- Economical in production, since many parts can be coated at the same time.

Nanotechnology Applications

Information Technology



- Smaller, faster, more energy efficient and powerful computing and other IT-based systems

Energy



- More efficient and cost effective technologies for energy production
 - Solar cells
 - Fuel cells
 - Batteries
 - Bio fuels

Medicine



- Cancer treatment
- Bone treatment
- Drug delivery
- Appetite control
- Drug development
- Medical tools
- Diagnostic tests
- Imaging

Consumer Goods

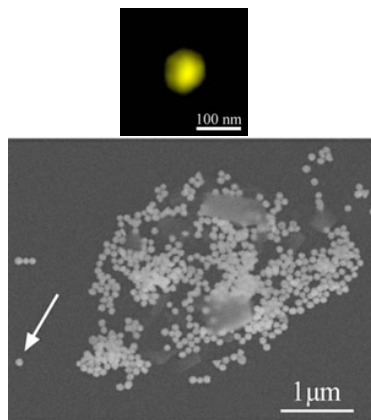


- Foods and beverages
 - Advanced packaging materials, sensors, and lab-on-chips for food quality testing
- Appliances and textiles
 - Stain proof, water proof and wrinkle free textiles
- Household and cosmetics
 - Self-cleaning and scratch free products, paints, and better cosmetics

Nanoscale Materials

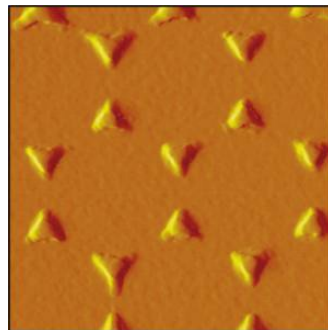
Nanoscale materials have feature size less than 100 nm – utilized in nanoscale structures, devices and systems

Nanoparticles and Structures



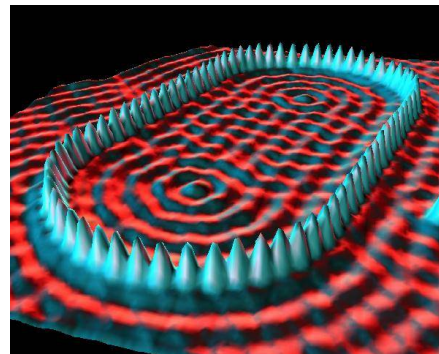
Gold nanoparticles

– TU Dresden/ESRF, 2008



Silver nanoparticles –

Northwestern Univ., 2002



A stadium shaped “quantum corral” made by positioning iron atoms on a copper surface – IBM Corp., 1993.



A 3-dimensional nanostructure grown by controlled nucleation of Silicon-carbide nanowires on Gallium catalyst particles

– Univ. of Cambridge, 2007

Disadvantages of Nanomaterial :-

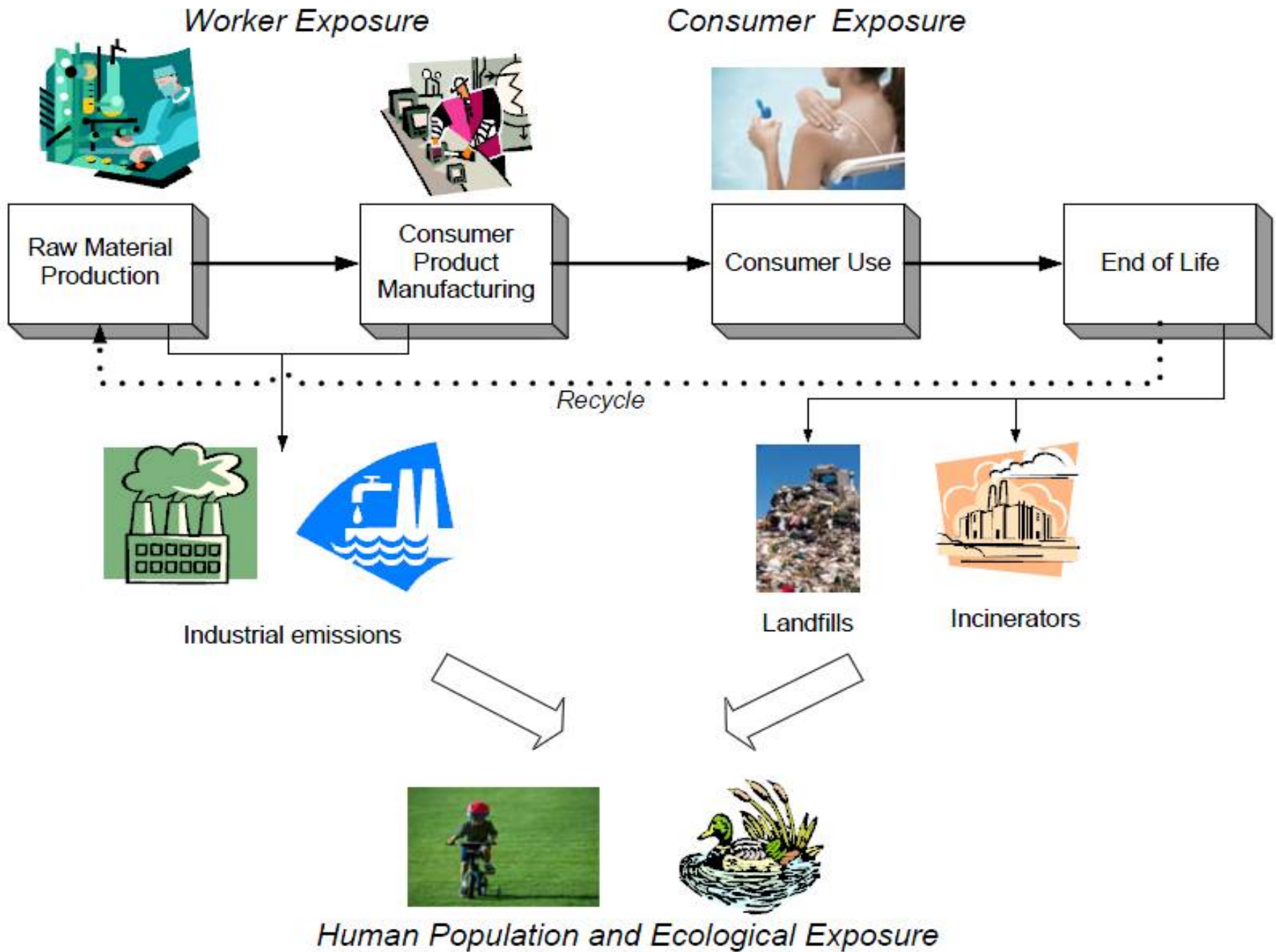
- The biggest disadvantage is that Nanomaterial is actually Very expensive, so not everyone can buy or afford it.
- Many of the people in the world are aware of the advancements that various field have made since the introduction of Nanotechnology into the world. But little do they know of the hidden dangers and potential risks involved with Nanotechnology running under the carpet.
- In present time we can use many types of items which is made by nanomaterial, it is very dangerous for our environment.



Nanomaterials :-

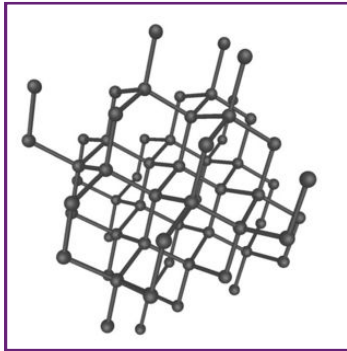
A Health Hazard

- Health hazards may be caused either due to direct use of products containing nanomaterials, or in the process of production or even accidentally.
- A big disadvantage for us is that research on the toxicity of nanomaterial is not taking place as fast as the material themselves are being developed and put to use.
- For example **Lux**, has conducted a research based on how nanomaterial currently used in day-to-day items like **Soap, Sunscreen lotions, cosmetics** etc will have an effect on our health.

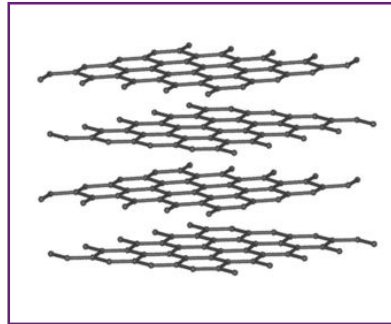


Did You Know?

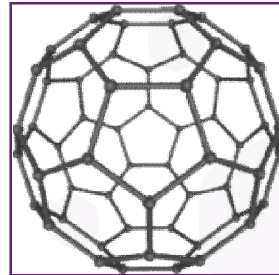
Allotropes of carbon have different covalent bonding arrangements.



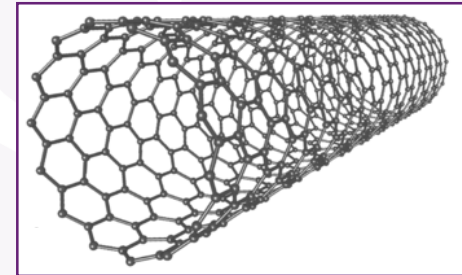
diamond



graphite



buckyball



nanotube

- Carbon atoms form covalent bonds by sharing outer shell electrons with each other
- Diamond, graphite, buckyballs and carbon nanotubes all have different covalent arrangements of carbon atoms
- The differing covalent arrangements of carbon atoms lead to the different properties of carbon allotropes.



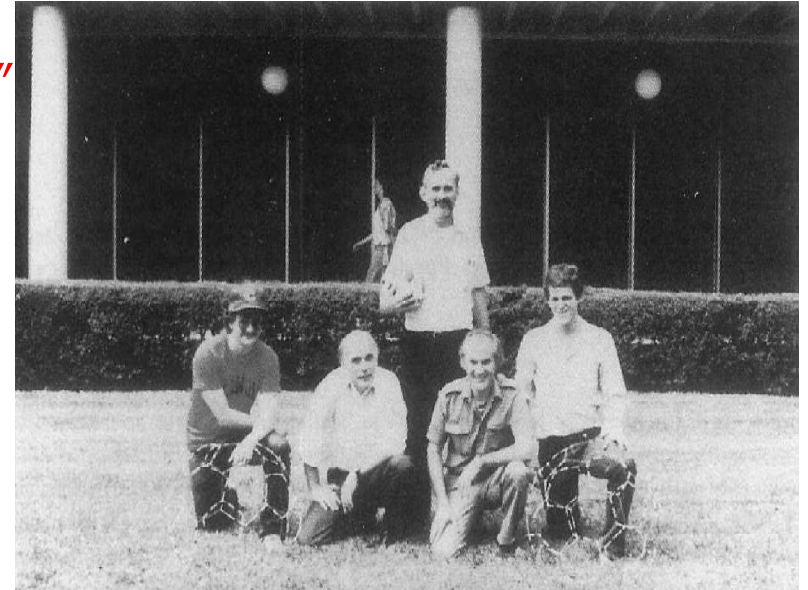
Fullerene

"The most symmetrical large molecule"

- Discovered in 1985
- Nobel prize Chemistry 1996

standing: Curl, kneeling (left to right):

O'Brian, Smalley, Kroto and Heath

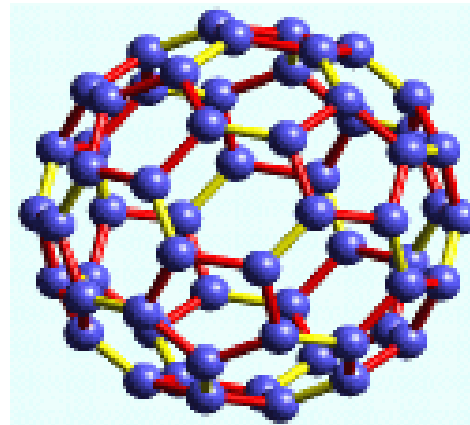


What is Fullerene?

A **Fullerene** is any molecule composed entirely of carbon, in the form of a hollow sphere, ellipsoid, or tube. Spherical fullerenes are also called **buckyballs**.

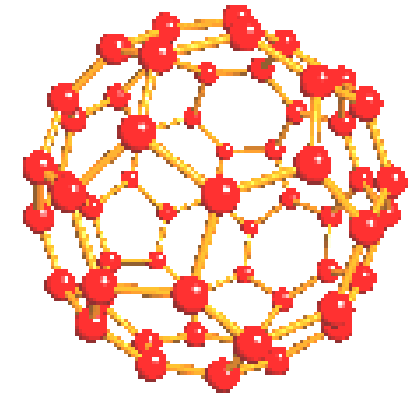


~1 nm

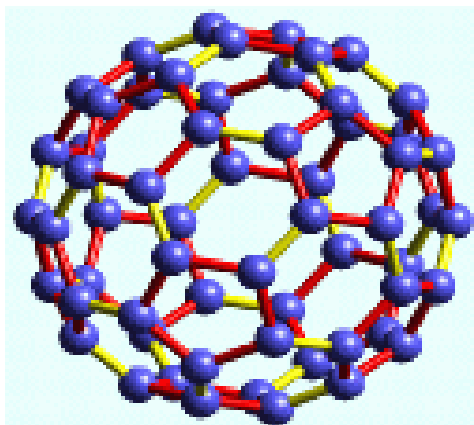


Discovery of Carbon C⁶⁰

- 1985, Robert F. Curl, ... **discovered a new form of carbon**, that **60 or 70** carbon atoms could **cluster** together to form a **cage-like molecule**.
- The molecular structure resembled the pattern of a **soccer ball** or the geodesic designs of **Buckminster Fullerenes**. Thus the **name buckyballs** or **fullerenes**.
- Since then the discovery has led to **new research** in polymers, semiconductors, and other various areas.
- **Nobel Prize** to their finders in 1996



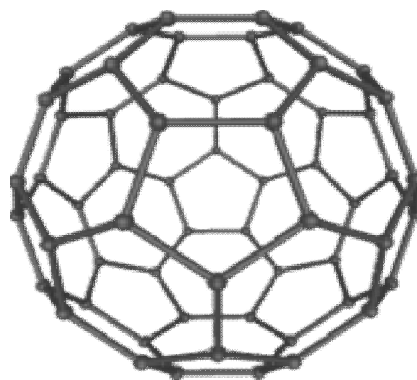
The substance of the Fullerene was **named** after the **American inventor, architects and philosophers Richard Buckminster Fuller** (1895 till 1983). As an architect, R.B.Fuller designed the constructions which exist of **5-corners and 6-corners**, for example, the American pavilion to the Expo in '67 in Montréal([geodesic dome](#)).



Richard Buckminster Fuller, c. 1917

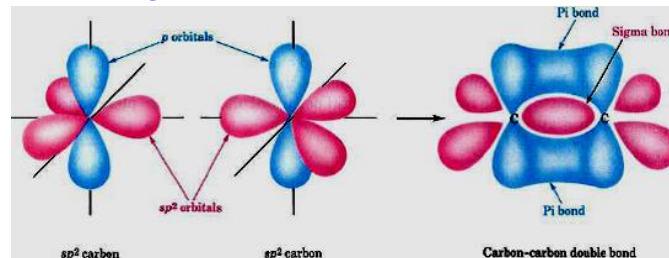


C₆₀ in solution



Properties of fullerene:

- ✓ The fullerenes are even number of **sp² hybrid carbon atoms** over the surface of a hollow cage.



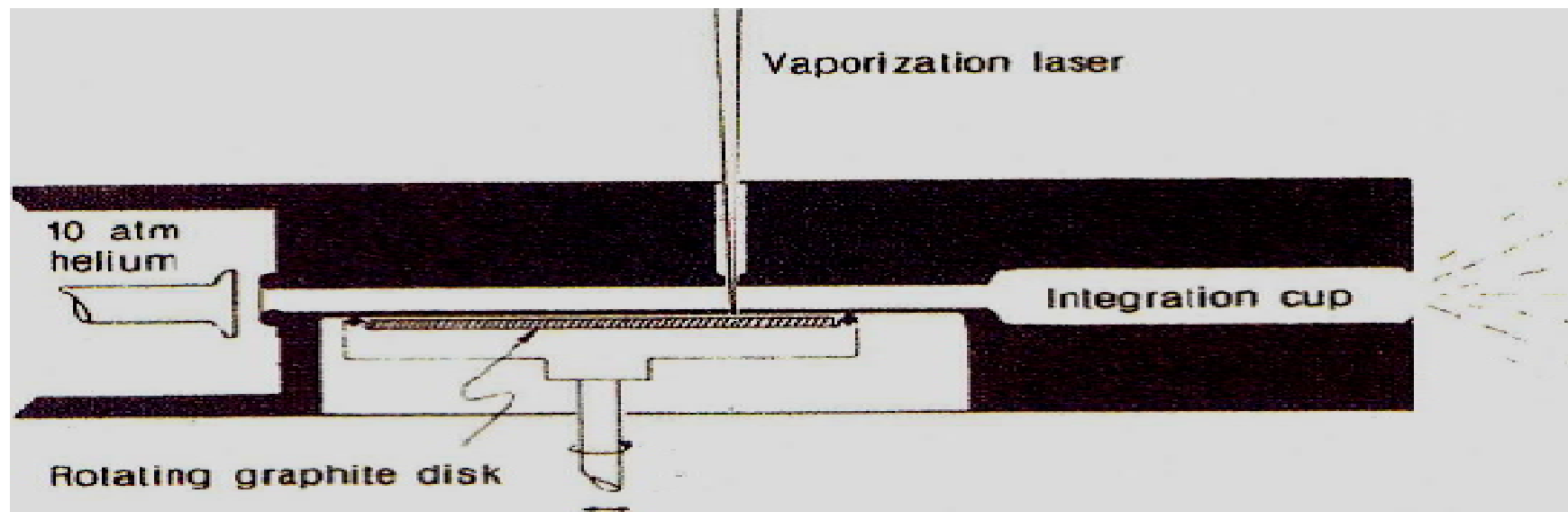
- ✓ The structure of C₆₀ is truncated **icosahedron** which looks like a soccer ball with **12 pentagons** and **20 hexagons**.
- ✓ The bond length in a pentagon is **1.45Å** and that bond between pentagons is **1.40Å**.
- ✓ At room temperature the solubility of C₆₀ is **2.8mg/ml** in **toluene**.
- ✓ It is **poor conductor of electricity** but when react with **reducing agent** (alkali) the resulting compound has high electrical conductivity.

How are fullerenes made?

Fullerenes can be prepared in simple process, graphite rods are vaporized in an **inert** atmosphere (Helium), by passing a high electric current through them.

This produces a light condensate called fullerene soot.

Extraction through toluene, separation and purification through column chromatography.



Schematic cross-sectional drawing of the supersonic laser-vaporization nozzle used in the discovery of fullerenes

Applications

- Researchers have found that water-soluble derivatives of fullerenes inhibit the **HIV-1** protease (enzyme responsible for the development of the virus) and are therefore useful in fighting the HIV virus that leads to **AIDS**.
- Elements can be bonded with C_{60} or other fullerenes to create more diverse materials, including **superconductors** and **insulators**.
- Non-linear optical devices
- Micro electronic devices.

Carbon Nanotube

Allotropes of carbon (graphite , diamond, Amorphous carbon and Fullerene) (cylindrical members of the fullerene structural family)

It was discovered by S. Iijima in 1991

Carbon nanotubes are **fullerene-related** structures which consist of **graphene cylinders** **closed** at either end with caps containing **pentagonal rings**

Nano tubes characterised as

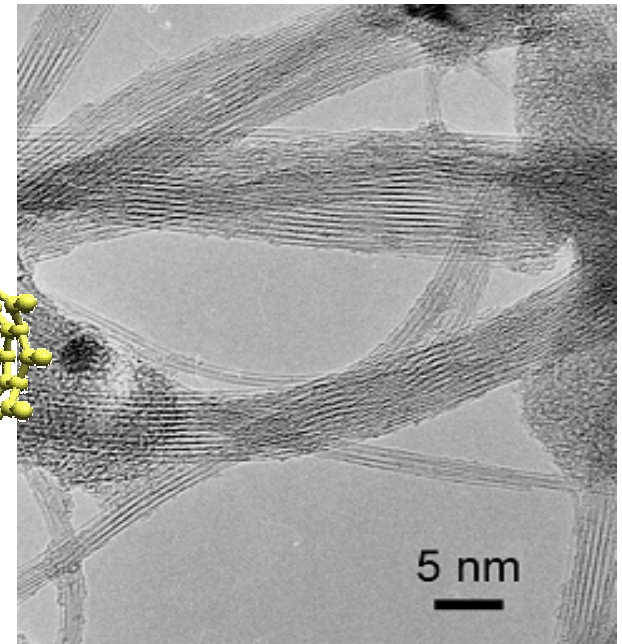
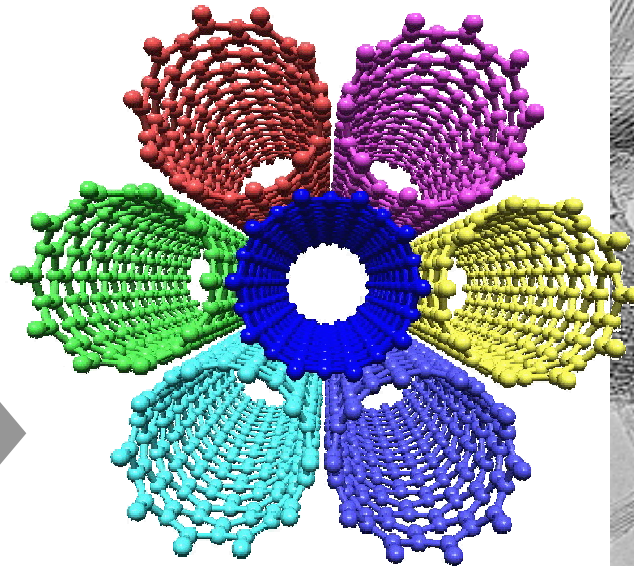
Single Walled nano tubes(SWNTs)

Multi-walled nano tubes(MWNTs)

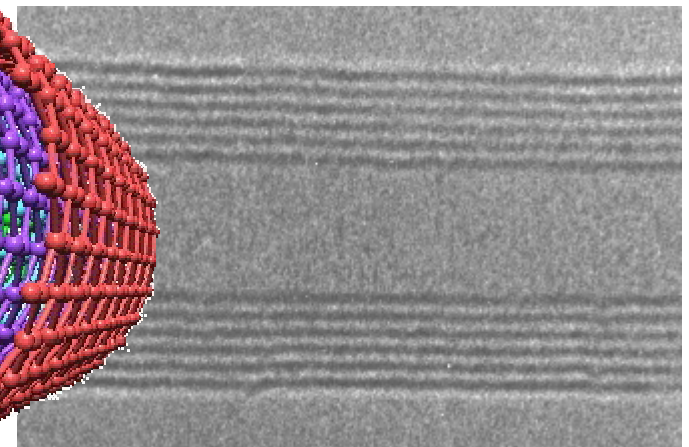
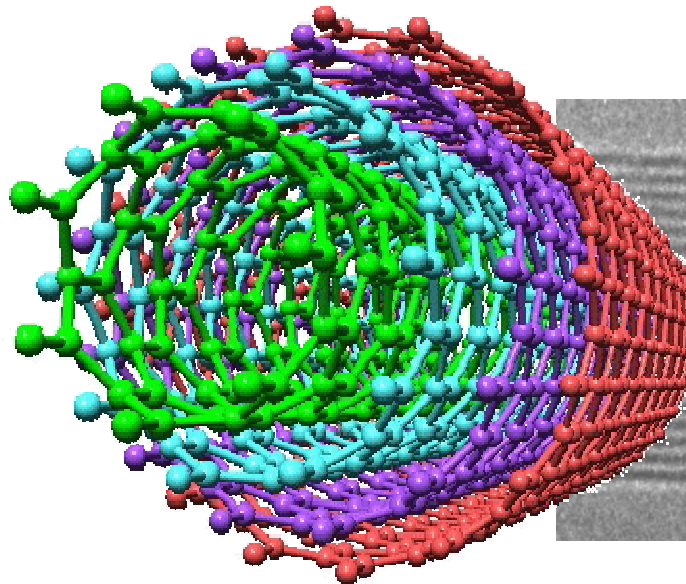
Types of Carbon nanotubes

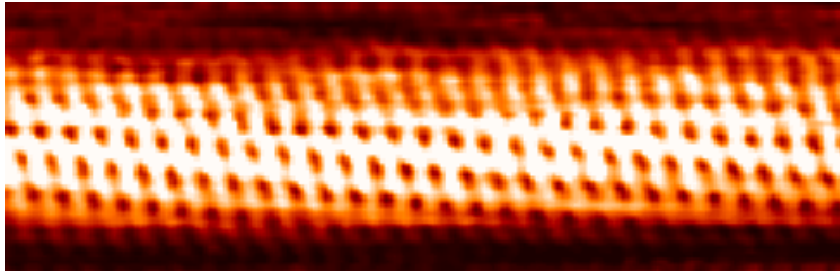
Two main types of carbon nanotubes:

Single-walled nanotubes (SWNTs) consist of a single graphite sheet seamlessly wrapped into a cylindrical tube.

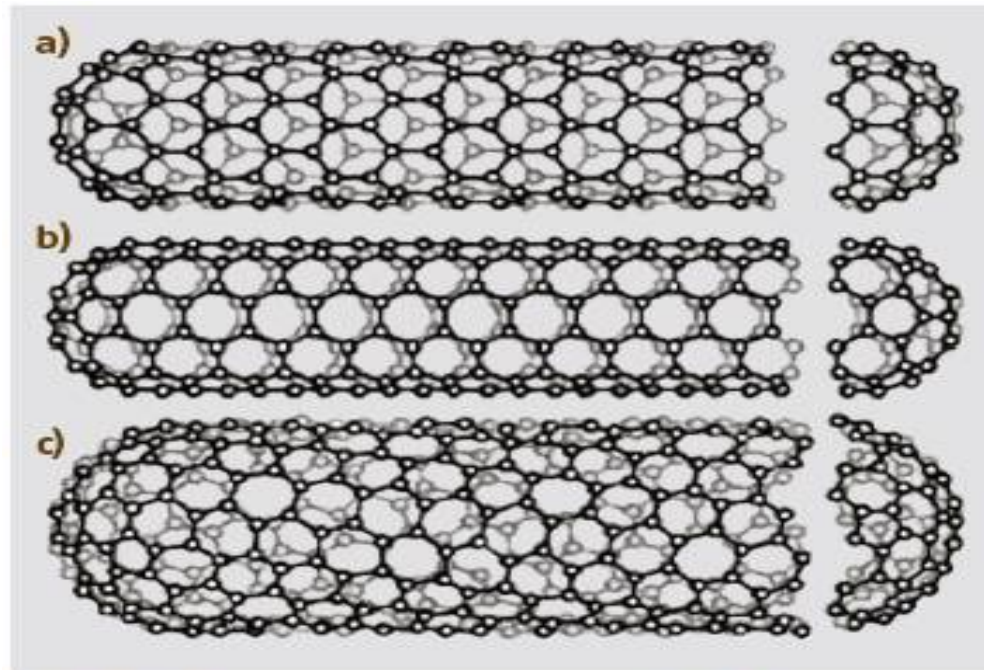
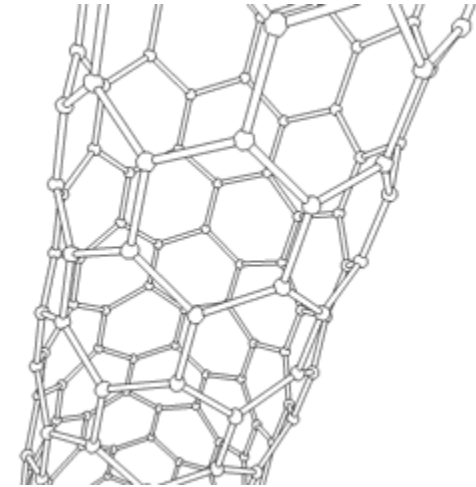


Multiwalled nanotubes (MWNTs) comprise an array of such nanotubes (more than one wall) that are concentrically nested with in.





Discovered in 1991 by the Japanese electron microscopist Sumio Iijima.



$(n,0)$
zigzag nanotube (9, 0)

(n,n)
armchair nanotube (5, 5)

(n,m)
helical (chiral)
nanotube (10,5)

Fig. 3.2a–c Sketch of three different SWNT structures as examples for (a) a zig-zag-type nanotube, (b) an armchair-type nanotube, (c) a helical nanotube (adapted from [3.13])

Properties of Nanotubes

The chemical bonding of a nanotubes is based on **Sp²** hybridisation similar to those of graphite.

Strength: It is strongest and stiffest materials.

MWNTs found their tensile strength, i.e., **63** giga pascal

Temperature Stability:

CNT is estimated to be up to **2800°C** in vaccum and about **750°C** in air of temperature stability.

Hardness

Standard single-walled carbon nanotubes can withstand a pressure up to **24 GPa** without deformation. They then undergo a transformation to superhard phase nanotubes. Maximum pressures measured using current experimental techniques are around **55 GPa**.

Defects

As with any material, the existence of a **crystallographic defect** affects the material properties.

Defects can occur in the form of atomic **vacancies**. High levels of such defects can lower the tensile strength by up to 85%. An important example is the **Stone Wales defect**, which creates a pentagon and heptagon pair by rearrangement of the bonds.

APPLICATIONS:

- Nano tubes field emission transistors for use as switching components in computers.
- Field emission light devices for fluorescent displays.

Morph: Concept video from Nokia and Cambridge Nanoscience Centre



<http://www.nokia.com/A4879144>

THANK YOU