VI E&C
35-00.

CN

Computer Networks

Introduction

A computer network is a number of computers (also known as nodes) connected by some communication lines.

A network is a set of devices (often referred to a node(s)) connected by communication links.

A node can be a computer, printer or any other device capable of sending and/or receiving data generated by other nodes on the network.

Two computers connected to the network can communicate with each other through other nodes if they are not directly connected.

Nodes can be n/w devices such as switches, routers, etc

Uses of computer n/ws:-

→ Exchange of information between different computers.
→ Interconnected small computers in place of large computers
→ Communication tools (ex: email, direct communication like voice or video chatting)
→ Some applications & technologies are of distributed s/ms
Ex:- Railway Booking s/ms, distributed databases

A network is a combination of hardware & software that sends data from one location to another. The hardware consists of the physical equipment that carries

signals from one point of the network to another. The software consists of instruction sets that make the possible services that we expect from a network.

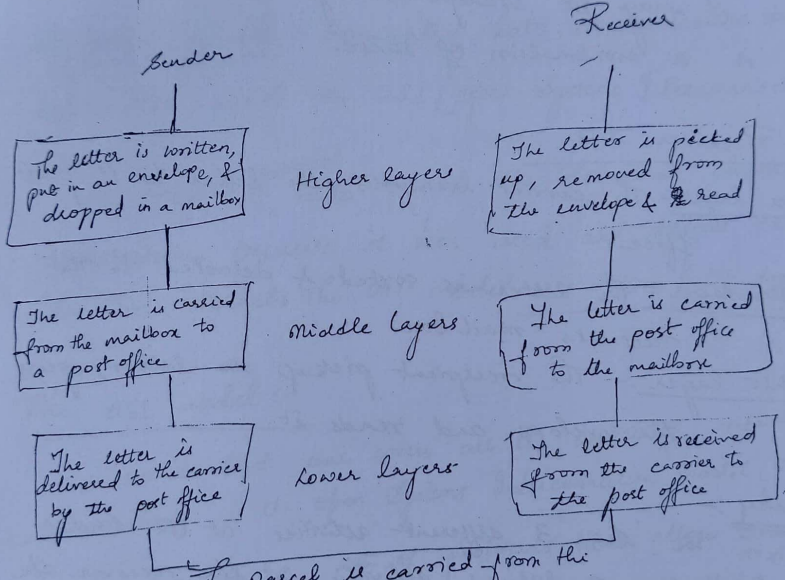
Networking tasks may be compared to solving a mathematical problem in a computer. Fundamental job of solving a problem is done by computer hardware. We need switches for memory location to read and manipulate data. The task is easier if software is available. At higher level, a program directs the problem solving process; the details of how the problem is solved by the hardware is left to the layers of the software that are called by the higher levels.

Illy Let us see the service provided by a computer network. For ex: the task of sending an email from one pt into another pt can be broken into several tasks, where each tasks are performed by a separate software package. Each s/w package uses the services of another software package. At lower layer, a signal or set of signals, is sent from the source computer to the destination computer.

Layered Tasks

Let us consider an example of layered tasks where two friends communicate through postal mail. If there were no services from post office sending a letter to a friend would be complex.

tasks involved in sending a letter



It requires a sender, receiver and a carrier that transports the letter. There is a hierarchy of tasks.

At the sender site:-

Let us consider the order in which activities takes place at the Sender site.

Higher layer → The sender writes the letter, puts in the envelope & writes the Sender & receiver addresses on the envelope & drop it mail box.

Middle layer:- The letter is picked up by the carrier and delivered to the post office.

Lower layer :- The letter is sorted at the post office, the carrier transports the letter.

On the way :-

The letter is on the way to the recepients post office. letter may go through a central office. In addition it may be transported by truck, train, aeroplane, boat, or a combination of these.

At the Receiver Site

Lower layer :- The carrier transports the letter to the post office

Middle layer :- The letter is sorted & delivered to the recepient's mail box.

Higher layer :- The recipient pickup the letter, opens the envelope and reads it.

Hirearachy :-

Here we see 3 different activities at the sender site and another 3 different activities at the receiver site The task of transporting the letter between the sender and receiver is done by carrier.

All the tasks must be done in the order of given ~~hirarchy~~ hierarchy. At the sender site letter must be written & dropped in the mailbox before it is ~~deliv~~ & ~~be~~ being picked up by the letter carrier & delivered to the post office. Nly the letter must be dropped in the recipients mailbox before it is picked & read by the recepient

Services

Each layer at the sending site uses the services of the layer immediately below it. The sender at the higher level uses the services of the middle layer. The middle layer uses the services of the lower layer & the lower layer uses the service of the carrier.

Layered model dominated data communication & networking before 1990 was the OSI (Open Systems Interconnection) model.

The TCP/IP protocol suite became the dominant commercial architecture because it was used and tested extensively in the internet. The OSI model was never fully implemented

The OSI model :-

An ISO standard that covers all the aspects of network communications is the Open Systems Interconnection model (OSI model)

An Open System is a set of protocols that allow any two different systems to communicate regardless of their underlying architecture.

The main purpose of the OSI model is to show how to facilitate communication between different systems without requiring changes to the logic of the underlying hardware & software.

Services

Each layer at the sending site uses the services of the layer immediately below it. The sender at the higher level uses the services of the middle layer. The middle layer uses the services of the lower layer & the lower layer uses the service of the carrier.

Layered model dominated data communication & N/w ing before 1990 was the OSI (Open Systems Interconnection) model.

The TCP/IP protocol suite became the dominant commercial architecture because it was used and tested extensively in the internet. The OSI model was never fully implemented

The OSI model :-

An ISO standard that covers all the aspects of network communications is the Open Systems Interconnection model (OSI model)

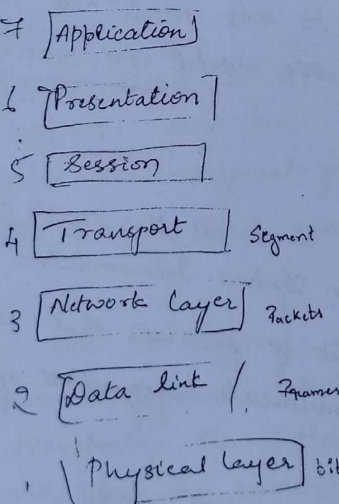
An Open System is a set of protocols that allow any two different systems to communicate regardless of their underlying architecture.

The main purpose of the OSI model is to show how to facilitate communication between different systems without requiring changes to the logic of the underlying hardware & software.

The OSI model is not protocol; it is a model for understanding & designing a network architecture that is flexible, robust, & interoperable.

The OSI model is a layered framework for the design of the network system that allows communication between all types of computer systems.

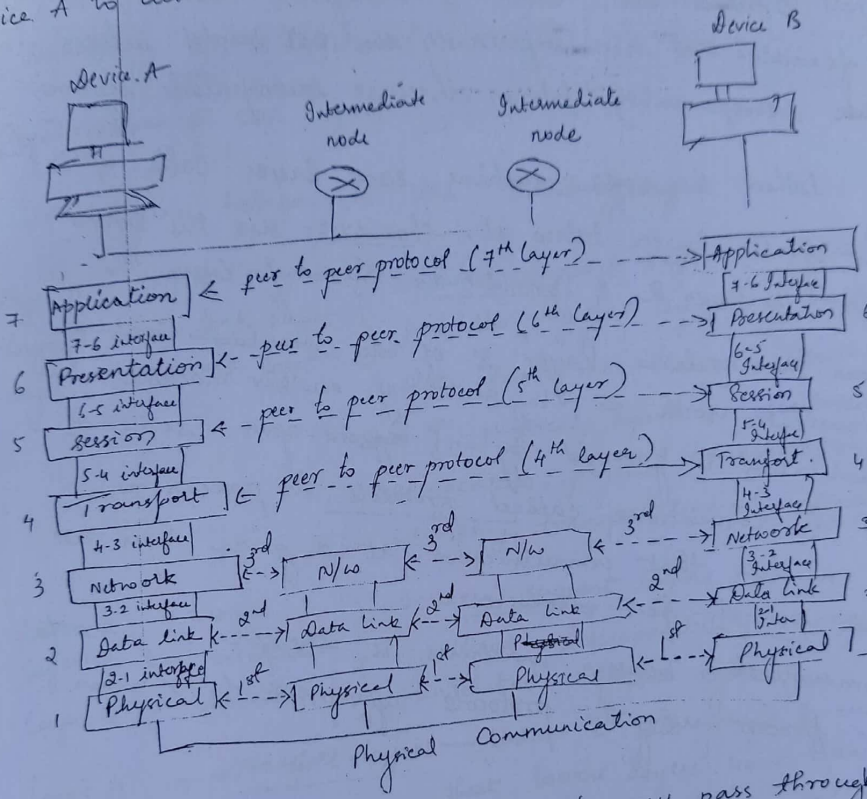
It consists of 7 separate but related layers. each of which defines a part of the process of moving information across a network.



Layered Architecture

OSI model comprises of 7 ordered layers: physical (Layer 1), data link (Layer 2), network layer (layer 3), transport layer (Layer-4), Session (Layer-5), Presentation

(Layer-6) and application (layer 7) the below figure (4)
shows the layers involved when a msg is sent from device A to device B.



As the message travels from A to B, it may pass through many intermediate nodes. These intermediate nodes usually involves only the first three layer of the OSI model.

In developing the model the designers distilled the process of transmitting data to its most fundamental elements. They identified which networking functions had related uses & collected those functions into discrete groups which became layers. Each layer defines a family of functions distinct from those of the

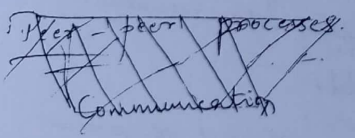
other layers.

By defining and localizing functionality in this fashion, the designers created an architecture that is both comprehensive and flexible. Most importantly the OSI model allows complete interoperability between otherwise incompatible systems.

Within a single machine each layer calls upon the services of the layer below it. For ex:- uses the services provided by layer 2 & provides the service for layer 4.

Between the machines layer x of one ~~machine~~ machines communicates with the layer x of another machine. This communication is governed by an agreed-upon series of rules & conventions called protocols. The processes on each machine that communicate at a given layer are called peer-to-peer processes.

Communication between machines is therefore a peer to peer process using the protocols appropriate to a given layer.



Interface between layers :-

→ The passing of the data & network information down through the layers of the sending device & backup through the receiving device is made possible by an interface b/w each pair of adjacent layers

→ Each interface defines the information & services a layer must provide for the layer above it.

→ Well-defined interfaces & layer functions provides modularity to a network.

Organization of the layers

→ 7 layers belongs to the 3 subgroups

Layer 1 — Physical layer
Layer 2 — Data link layer
Layer 3 — Network layer

} Network support layers — deal with the physical aspects of moving data from one device to another

Some of physical aspects are electrical specification, physical connections, physical addressing, transport timing & reliability.

Layer 5 — Session
Layer 6 — Presentation
Layer 7 — Application

} → user support layers allows interoperability among unrelated software systems.

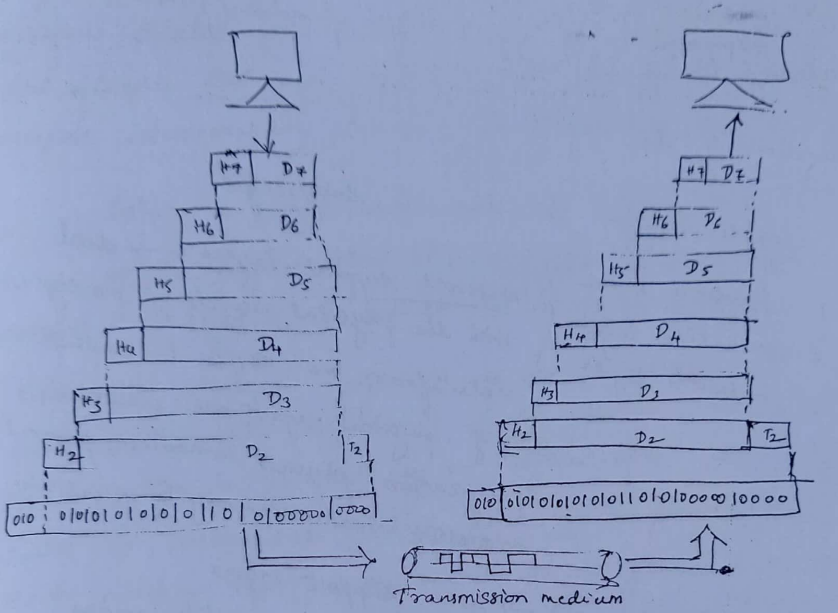
Layer 4 → Transport layer — links 2 sub groups & ensures that lowers layers have transmitted is in a form that the upper layers can use.

Upper OSI layers are ~~almost~~ always implemented in S/W

Lower layers are a combination of H/W & S/W

Physical layer is mostly hardware

An overview of the exchange of the information using the OSI model is shown in the fig.



D_7, D_6, D_5, ... & so on represents the data in the respective layers. The process begins at layer 7 then moves layer by layer downwards sequential.

Each layer adds its own header or trailer to the data from the previous layer. The layer 6 consider the Header & data of layer 7 as the data I/p for layer 6 & so on.

Usually trailer is added at the layer 2.

Formatted data at layer 1 (physical layer) is changed into electromagnetic signal and transported along a physical link.

Upon reaching the destination the signal is passed into

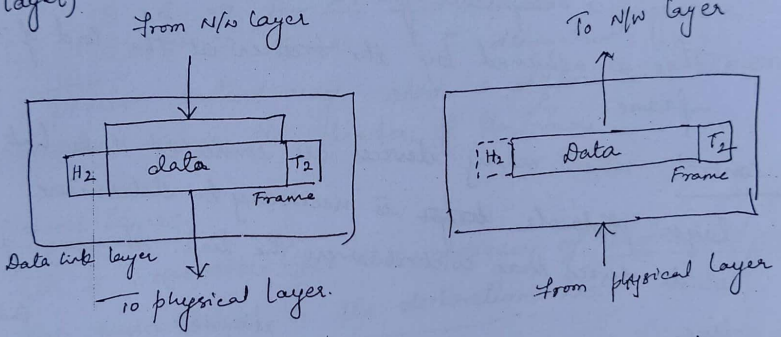
Simplex → one way communication.

Half-duplex :- 2 devices can send and receive but not at the same time.

Full-duplex (simply duplex) → 2 devices can send & receive at the same time.

Data link layer :- It transforms the physical layer, a raw transmission facility, to a reliable link.

⇒ physical layer appear error-free to the upper layer (N/w layer)



The data link layer is responsible for moving frames from one hop (node) to the next.

⇒ Responsibilities

* Framing :- divides the stream of bits received from the N/w layer into manageable data units called frames
* Physical addressing :- If frames are to distributed to the s/m in N/w, the data link layer adds a header to the frame to define the sender &/or receiver of the frame. If frame is ~~intended~~ intended for a system outside sender's n/w, receiver address is the address of the

device that connects the network to the Next one.

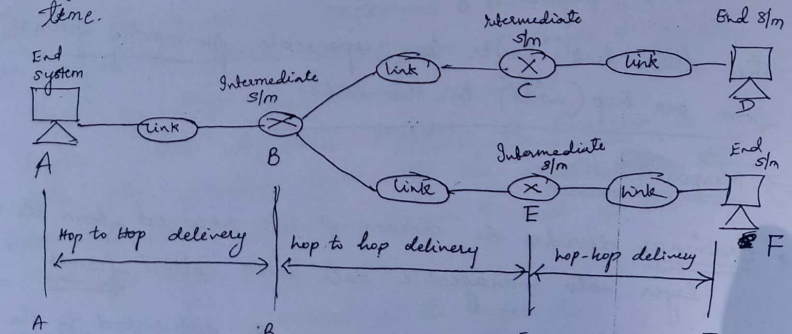
Flow control :- If the receiver absorbs the data at lesser data rate than that of the sender, the D.L layer imposes a flow control mechanism to avoid overwhelming at the Receiver.

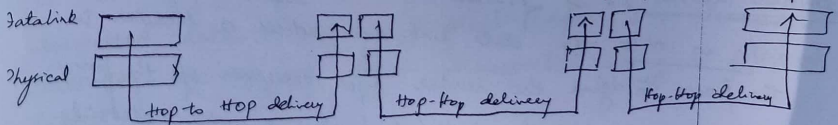
Error control :- Adds reliability by adding a mechanism to detect and re-transmit damaged or lost frames.

→ recognizes duplicate frames

→ This is achieved by the trailer at the End of the frame.

Access control :- when many devices are connected data link layer protocols ~~is~~ is necessary to determine which device has control over the link. at any given time.





Hop to Hop delivery.

The figure shows communication at the data link layer occurs between two adjacent nodes. (8)

To send data from A to F → 3 partial deliveries are made First data link layer at A sends to the D.L layer at B(router). Second the data link layer at B to E & then E to F.

Frames are exchanged b/w 3 nodes with different values of the header.

A → B ⇒ A – source, B – destination addr
B → E ⇒ B – src, E – destination
E → F ⇒ E → src & F → destination.

Trailer values may also be different if the error checking involves the header of the frame

Network Layer:-

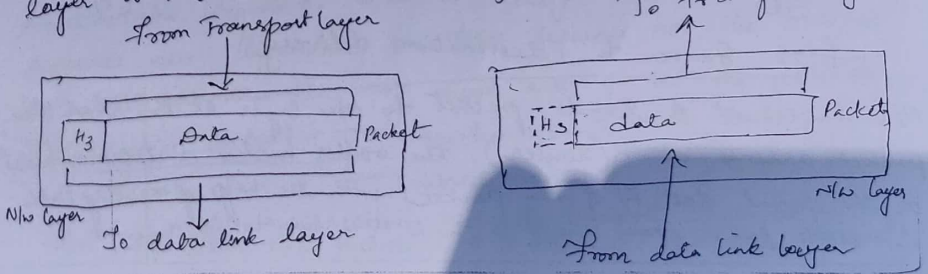
It is responsible for the delivery of individual packets from source host to the destination host.

src to dest delivery of packets possible across multiple N/ws (links).

D.L layer → delivery of N/w b/w s/ms in the same N/w

N/w layer → pt of origin to its final destination.

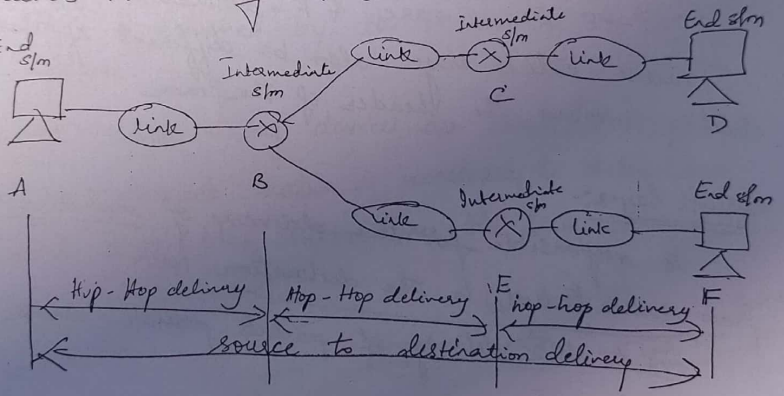
The figure shows the ~~respon~~ relationship of the ~~the~~ N/w layer to the data link & transport layer.



Responsibilities of the N/w layer

→ Logical addressing :- If a packet passes the N/w boundary we need another addressing system to help distinguish the src & dest s/m. N/w layer adds a header to the packet coming from the upper layer which includes the logical address of both sender & receiver.

→ Routing :- when independent N/w or links are connected to form the internetworks or large n/w connecting devices (called routers or switches) route or switch the packets to their final destination.



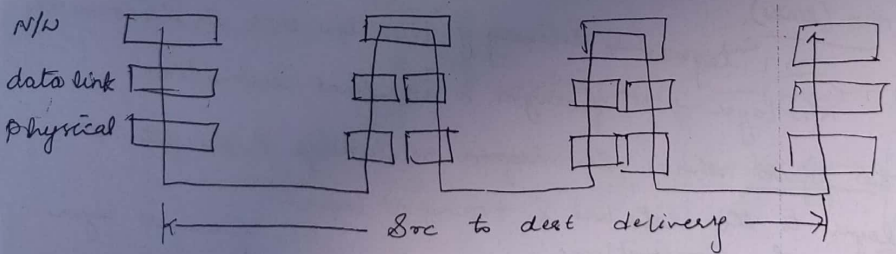
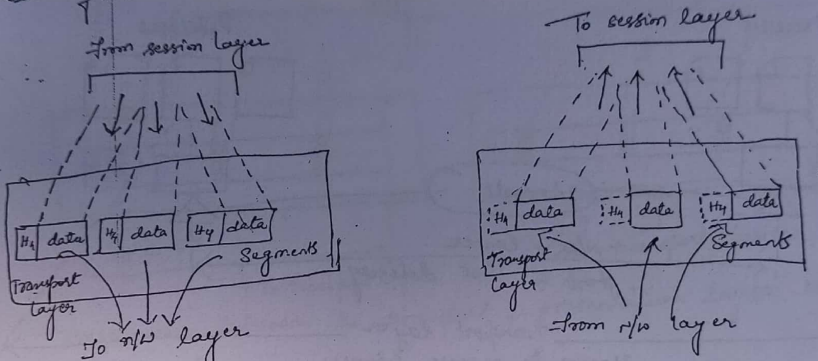


fig: Source to destination delivery.

N/w layer at A sends packet to N/w layer at B. When the packet arrives at B (router), the router makes a decision based on the final dest (F) of the packet with the help of routing table ∴ sends packet from N/w of B to E & in turn to N/w layer at F

Transport layer :-

Responsible for the delivery of msg from one process to another. ie process to process delivery of the entire message n/w layer oversees the src - destination delivery of individual packets. The transport layer ensures that the whole message arrives intact & in order, overseeing both error control & flow control at the source to destination level.



Other responsibilities :-

1) Service point addressing :- Computer runs several programs at the same time. So the msg has to be delivered to a specific process (running program) from the specific process the sender. ∴ Transport layer must add an other type of address called service type address (or port address) to the header.

2) Segmentation & Re assembly :- A message is divided into transmittable segment, with each segment containing a sequence no. Depending on these sequence nos the Transport layer re-assembles the msg upon arriving at the destination

3) Connection control :- Connectionless → treats each segment as an independent packet & delivers it
Connection oriented → makes a connection with the transport layer at the dest before delivering the packets & it is terminated after data is delivered

between the diff. encoding methods. ~~The~~ This layer ~~sender~~ at the sender changes the info from its sender-dependent format into common format. The presentation layer at the receiving machine changes the common format into its receiver-dependent format.

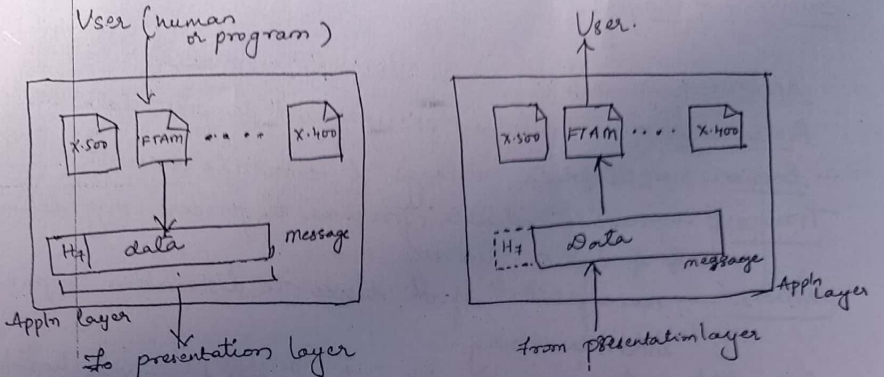
⇒ Encryption:- To carry sensitive information a s/m must be able to ensure privacy.
Encryption means sender transforms the original info to another form & sends the resulting msg over the N/w
Decryption is the reverse of encryption

⇒ Compression:- Data compression reduces the number of bits contained in the info. It is important particularly in the transmission of media such as text, audio and video.

Application Layer:-

→ It is responsible for providing services to the user.
→ It enables the user, whether human or s/w, to access ~~the~~ the N/w.
→ Provides user interface & support for services such as e-mail, remote file access & transfer, shared database management & other types of distributed info services.
→ Relationship b/w the appln layer, user & presentation layer is shown in fig.
→ No. of services are provided by the application layer for ex X.400 (msg handling services), X.500 (directory services), & File Transfer, access and management (FTAM).
~~→ The user in this example employs X.400 to send an e-mail message~~

Application layer



Specific services

1) Network virtual terminal :- It is a s/w version of a physical terminal, & it allows a user to log on to remote host.
→ The appln creates a s/w emulation of a terminal at the remote host.
→ User's computer talks to the ~~host~~ s/w terminal which in turn talks to the host & vice versa
→ Remote host believes it is communicating with one of its own terminals & allows user to log on.

(2) File transfer, access and management :- It allows a user to access files in a remote host, to retrieve files from a remote computer for use in local computer & to manage or control files

3) Mail Services → basis for e-mail forwarding & storage.

4) Directory services :- provides distributed database sources & access for global information abt various objects & services.

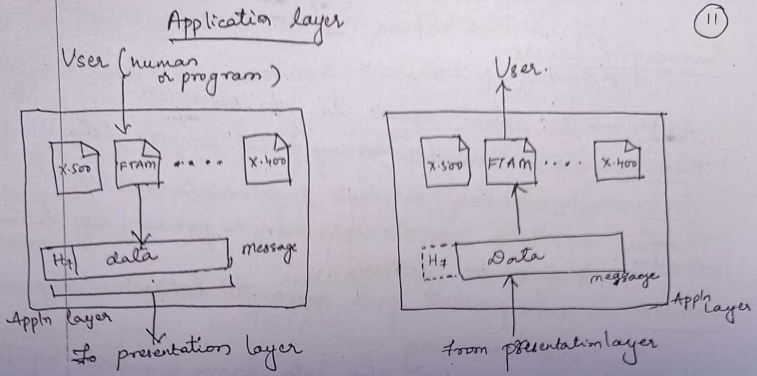
Summary of Layers

Appln → to allow access to n/w resources
Presentation → to translate, encrypt and compress data
Session → to establish, manage & terminate sessions
Transport → provide reliable process to process msg delivery & error delivery.
Network → move packets from source to destination, to provide internetworking
Data link → To organize bits into frames to provide hop-hop delivery
Physical → to transmit bits over a medium to provide mechanical & electrical specifications

TCP/IP protocol suite :-

→ It was developed prior to the OSI model.
→ TCP/IP protocol suite ~~this~~ was having 4 layers
* host to network layer
* internet
* transport
* application

TCP/IP protocol is compared to OSI model
Host to network layer is equivalent to combination of the physical & data link layer
Internet layer → equivalent to Network layer
Application layer is roughly doing the job of the session, presentation and application layers
Transport layer in TCP/IP protocol suite is taking care of the part of the duties of the session layer



Specific services

1) Network virtual terminal :- It is a s/w version of a physical terminal, & it allows a user to log on to remote host.

→ The appln creates a s/w emulation of a terminal at the remote host.

→ User's computer talks to the ~~host~~ s/w terminal which in turn talks to the host & vice versa

→ Remote host believes it is communicating with one of its own terminals & allows user to log on.

2) File transfer, access and management :- It allows a user to access files in a remote host, to retrieve files from a remote computer for use in local computer & to manage or control files

3) Mail Services → basis for e-mail forwarding & storage.

4) Directory services :- provides distributed database sources & access for global information abt various objects & services.

Summary of Layers

Appln → to allow access to N/w resources
Presentation → to translate, encrypt and compress data
Session → Establish, manage & terminate sessions
Transport → provide reliable process to process msg delivery & error delivery.
Network — move packets from source to destination, to provide internetworking
Data link → To organize bits into frames to provide hop-hop delivery
Physical — to transmit bits over a medium to provide mechanical & electrical specifications

1.4 TCP/IP protocol suite :-

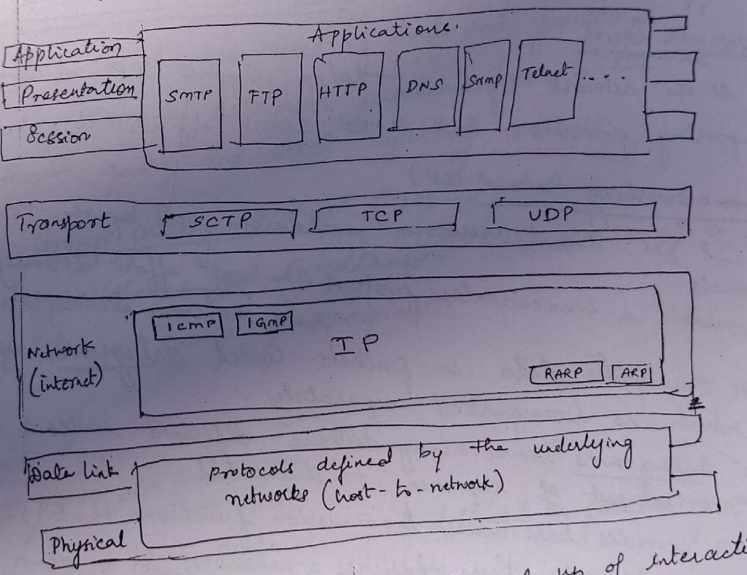
→ It was developed prior to the OSI model.

→ TCP/IP protocol suite ~~was~~ was having 4 layers
* host to network layer
* internet
* transport
* application

TCP/IP protocol is compared to OSI model
Host to network layer is equivalent to combination of the physical & data link layer
Internet layer → equivalent to Network layer
Application layer is roughly doing the job of the session, presentation and application layers
Transport layer in TCP/IP protocol suite is taking care of the part of the duties of the session layer

Here we assume that TCP/IP protocol suite has 5 layers: physical, data link, network, transport & application.

The first four layers provide physical standards, network interfaces, internetworkings and transport functions that correspond to the first four layers of the OSI model. 3 topmost layer are indicated by the single layer called application layer in TCP/IP.



→ TCP/IP is a hierarchial protocol made up of interactive modules, each of which provides a specific functionality, however modules are not interdependent.

→ It contains relatively independent protocols that can be mixed and matched depending on the system needs.

→ Hierarchial means that each upper-level protocol is supported by one or more lower level protocols.

3 protocols at transport layer — TCP [Transmission Control Protocol], UDP [User Datagram Protocol], SCTP [Stream Control Transmission Protocol]

Network layer → main protocol is Internetworking Protocol (IP)

Physical and Data Link Layers

→ It supports all the standard & proprietary protocols, but it has no defined specific protocol.

→ N/w in a TCP/IP internetwork can be a local area n/w or a wide area n/w.

Network layer

At the network layer TCP/IP supports IP which uses 4 supporting protocols: ARP, RARP, ICMP & IGMP.

Internetworking Protocol (IP)

→ IP is the transmission mechanism used by the TCP/IP protocols

→ unreliable & connectionless protocol — best effort delivery service. means no error checking & tracking

→ It transports data in packets called datagrams, each of which is transported separately.

* datagrams travel along different routes & can arrive out of sequence or duplicated.

→ IP provides bare-bones transmission functions that free the user to add only those facilities necessary for a given appln.

∴ allows maximum efficiency.

ARP [Address Resolution Protocol]

→ It is used to associate a logical address with a physical address.

→ on a physical n/w, such as LAN, each device on a link is identified by a physical or station address usually imprinted on the network interface card (NIC)

→ ~~Rtarp~~ to find the physical address of the node when its internet address is known.

Reverse Address Resolution Protocol (RARP)

→ It allows a host to discover its internet address when it knows only its physical address.
→ Used when a computer is connected to a network for the first time.

Internet Control Message Protocol (ICMP)

→ It is mechanism used by hosts and gateways to send notification of datagram problems back to the sender
→ Sends query and error reporting messages.

Internet Group message protocol (IGMP)

It facilitates the simultaneous transmission of a message to a group of recipients.

Transport Layer:

→ 2 protocols [TCP, UDP]

IP is host-host protocol → it delivers a packet from one physical device to another.

TCP & UDP are transport level protocols → for the delivery of message from a process (running program) to another process.

UDP [User datagram Protocol]

→ It is the simpler of the ~~two~~ 2 standard TCP/IP transport protocols.
→ process to process ~~com~~ protocol which adds only port addresses, checksum error control, & length info to the data from upper layer

Transmission Control Protocol [TCP]

→ provides full transport layer services to application

→ It is reliable stream transport protocol

Stream in this context means connection oriented.

→ At the sending end of each transmission, TCP divides a stream of data into smaller units called segments where each segment includes a sequence number for reordering after receipt, together with an acknowledgement number for the segments received.

→ Segments are carried across internet inside IP datagrams.

→ At receiver end, TCP collects each datagram & reorders the transmission based on sequence numbers

Stream Control Transmission Protocol (SCTP)

→ It supports for newer applications such as voice over the Internet.

→ It combines the best features of UDP & TCP

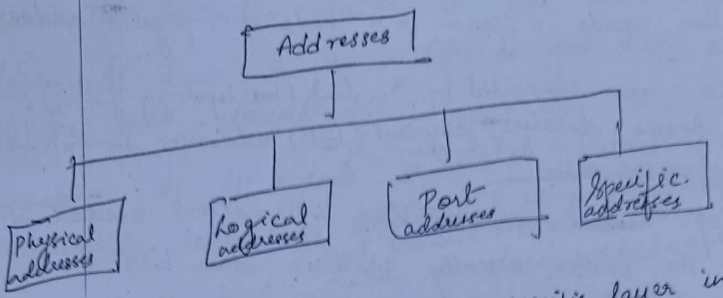
Application layer :-

→ It is equivalent to the combined session, presentation, & application layers in the OSI model. Many protocols are defined at this layer.

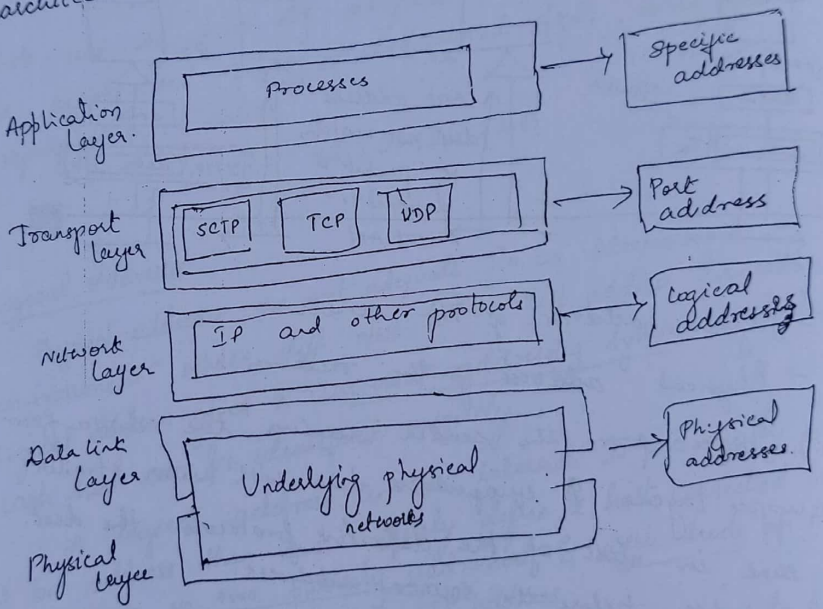
Addressing :-

4 levels of addresses are used in an internet employing TCP/IP protocol :

- Physical (link) addresses
- Logical (IP) "
- Port "
- Specific addresses



Each address is related to the specific layer in TCP/IP architecture as shown in fig



Physical addresses

It is also known as the link address, it is the address of a node as defined by its LAN or WAN

→ It is included in the frame by D.L layer.

→ It is the lowest level address.

→ has authority over the n/w. The size and format vary depending on the network.

Ex:- → Ethernet uses a 6-byte physical address (Nw interface card) imprinted on NIC.

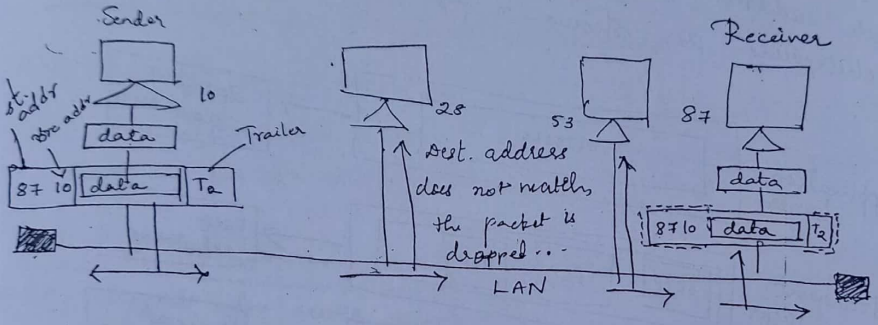
→ Local talk (apple), has a 1 byte dynamic address that change each time the station comes up.

Let us consider an example where a node with physical address 10 sends a frame to a node with physical address 87.

→ 2 nodes are connected by a link (bus topology LAN).

At DLL frame contains physical (link) addresses in the header. ~~The rest of information~~

The rest of header contains other information needed at this level. The trailer usually contains extra bits needed for error detection.



10 → Physical address of the sender

87 → Physical address of the receiver.

D.L layer from the sender receives the data from its upper layers. It encapsulates data with header & trailer. Note that in most of the data link protocols, the dest. address comes before the source address.

In this example isolated LAN with bus topology is considered where the frames are propagated in both the directions (left & right). The frame dies when the end terminal is reached (if it terminated appropriately). The frame propagated is sent to each & every station & the each of the station receives the frame &

checks it for the match in the destination address. If there is no match in the dest. addr the frames are dropped.

However if it finds a match between the ~~source &~~ destination address of its physical address frame is received checked, header & trailer are dropped & data is decapsulated and sent to the upper layers.

Let us see how ~~the~~ physical address 48 bits (6-byte) is represented using 12 hexadecimal digits, every byte is separated by a colon

07:01:02:01:2C:2B

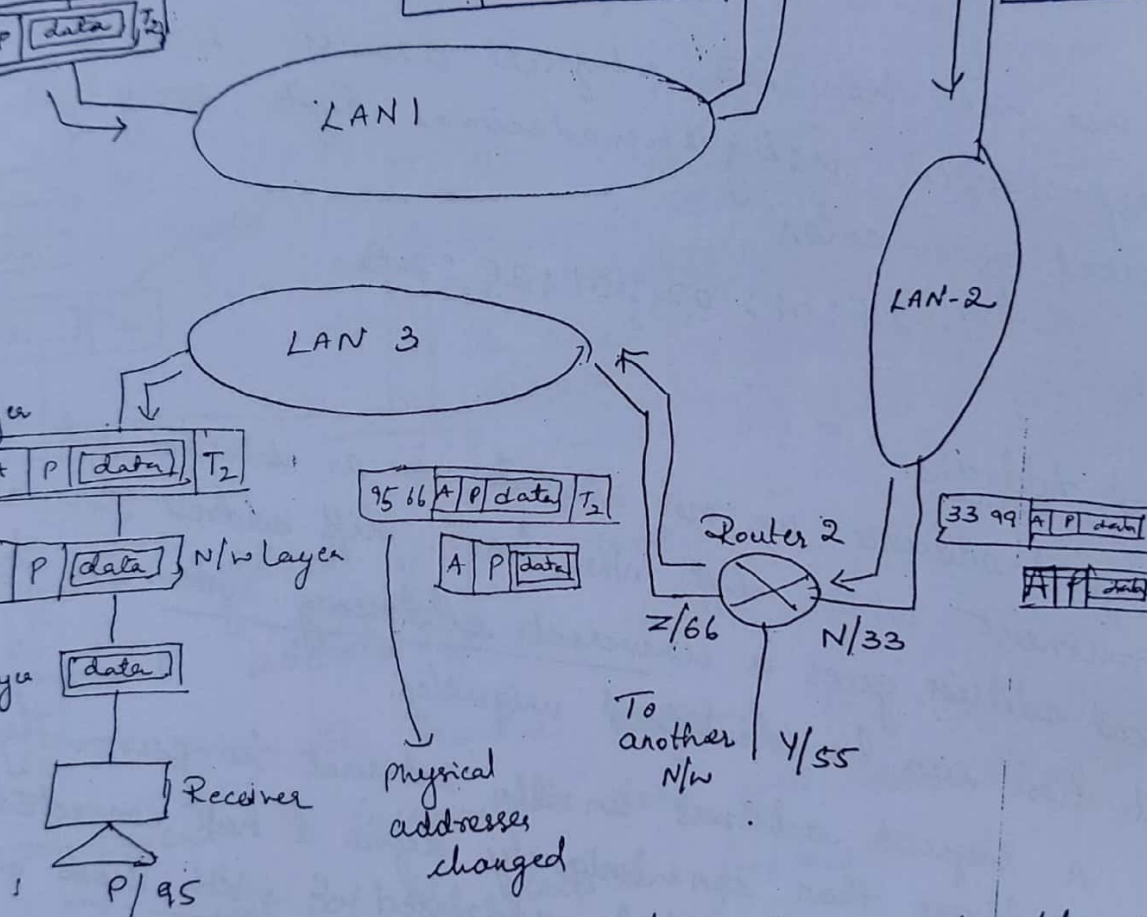
Logical Addresses

Physical addresses are not adequate in an internetwork environment where diff n/ws have diff address formats. Logical address gives a universal addressing system in which each host can be identified uniquely.

A logical address in the internet is currently a 32 bit address than can uniquely define a host connected to the Internet. No two publicly addressed & visible hosts on the internet can have the same IP address.

Physical addresses ~~will~~ will change from hop to hop, but the logical addresses usually remain the same.

Let us take an example of a part of an internet with two routers connecting 3 LAN's. Each device has a pair of addresses (physical and logical) for each connection. Each Stn (computer) is connected to only one link & ∴



mputer with logical address A and physical addr
communicate with logical address P and physical
95 [Both logical & physical address are numericals h
take as alphabets for logical address]

der encapsulates its data in a packet at the netwo
& adds 2 logical address A & P. In most p
he logical source address comes before the logical

destination address.

→ n/w layer needs to find physical addr. of the next hop before the packet can be delivered.

After consulting routing table n/w layer finds the logical address of the router i.e next hop to be F.

ARP finds the physical address of router 1 that corresponds to logical address of 20. Now n/w layer passes this to d.l. layer which encapsulates the packet with physical destination address 20 & physical src addr 10.

→ Frame is received by every device on LAN 1 & router 1 matches its address with destination physical addr.

→ Router decapsulates & gets the logical address P. Since logical address doesn't match the packet is forwarded.

→ Now the router consults the routing table & ARP to find the physical dest. addr. of the next hop. & then creates new frame, encapsulates the packet & sends to router 2.

→ Src phy addr change from 10 to 99 & dest address from 20 to 33.

The logical source and destination addresses must remain the same. Else pack will be lost.

→ n/w at the router 2 the physical address are changed & a new frame is sent to the destination where the match between the logical address is found and the data is decapsulated from packet and delivered to the upper layer.

The physical address will change from hop to hop, but the logical address usually remains the same.

Port Addresses :-

IP address & physical addr. are necessary for a quantity of data to travel from source to dest host.

→ End objective of Internet communication is a process communicating with another process.

For. example :- Computer A communicate with computer C using TELNET at the same time Computer A communicates with computer B by using File Transfer Protocol (FTP).

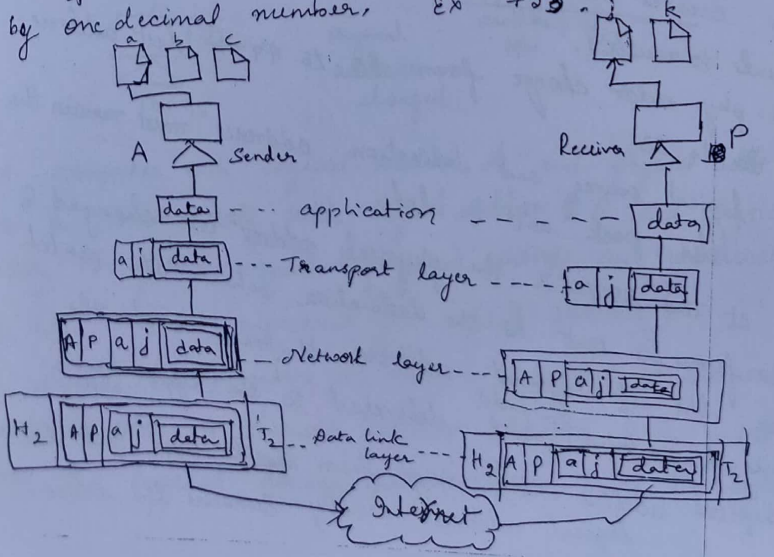
For these processes to receive data simultaneously we must label the different processes i.e they need addresses.

In TCP/IP architecture the label assigned to a process is called port address.

Port address in TCP/IP is 16 bit length.

The physical addresses change from hop to hop, but the logical and port addresses usually remain the same.

A port address of 16-bit is being represented by one decimal number. Ex 753.



Consider the fig above where 2 computers communicating via the internet.

→ Sending computer is running 3 processes at a given time a, b & c.

→ Receiving computer is running 2 process j & k at a given time.

→ Process 'a' of sending machine (computer) has to communicate with process 'j' of receiving machine.

→ ~~Although~~ Although both the computers are using the same application, (FTP) the port address is diff. : one is the client and other is the server program.

→ Transport layer encapsulates the data from appln layer into a packet and adds 2 port addresses (a & j). Source & destination.

→ Packet from Transport layer is encapsulated in the other packet at the N/w layer with logical src & logical dest. addresses (A & P).

→ Finally this packet is encapsulated in a frame ~~with~~ with the ~~source~~ physical source and destination addresses of the next hop

Specific Addresses → Some applns have user-friendly addresses that are designed for that specific address.

→ Ex:- email addresses, the Universal Resource Locator (URL)

In this ~~email address~~ the first defines the recepient of an e-mail, the second is used to find a document on the world wide web.

These addresses change based on ~~physical~~ logical address

& the corresponding port of the sending computer.

Telephone Networks

Telephone N/W uses circuit switching.
→ In the beginning entire telephone network was referred as "Plain Old Telephone Systems", was originally an analog S/m using analog signals to transmit voice. Now the telephone N/W carry data in addition to voice.
The N/W is now digital as well as analog.

Major components of Telephone N/W

The telephone N/W is made of three major components:
- local loops
- trunks
- Switching offices

It has several level of switching offices like end offices, tandem offices & regional offices as shown in fig

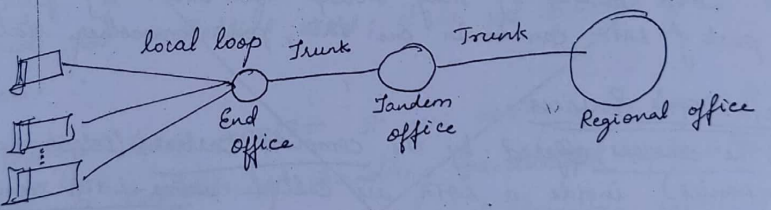


fig: A telephone system

Local loops:- It is a twisted pair cable that connects the subscriber telephone to the nearest end office or local central office.
→ Bandwidth of the local loop used for voice is 4000Hz (4kHz).
→ Telephone number associated with each loop is given by the first three digit endocabes the local telephone number define the office, & the next four digits define the local loop number.

Trunks are the transmission media that handles the communication between offices.

→ It handles hundreds or thousands of connections through multiplexing.

→ Transmission is usually through optical fibres or satellite links.

Switching offices

To avoid having a permanent physical link between any two subscribers, the telephone company has switches located in a switching office.

→ A switch connects several local loops or trunks and allows a connection between different subscribers.

LATAs [Local Access Transport Areas]:- After the divestiture of 1984 US was divided into more than 200 LATAs

→ A LATA can be a small or large metropolitan area.

→ A small state may have one single LATA ; a large state may have several LATAs.

→ A LATA boundary may overlap the boundary of a state, part of LATA can be in one state, part in another state.

Intra-LATA Services

→ The services offered by the common carriers (telephone companies) inside a LATA are called intra-LATA services. The carrier that handles these services is called a local exchange carrier (LEC).

Before the telecommunications Act of 1996 intra-LATA services were granted to one single carrier.

After 1996 more than one carrier could provide services inside a LATA.

The carrier that provided services before 1996 owns the cabling system (local loops) & is called the incumbent local exchange carrier (ILEC).

Digital Subscriber Line (DSL)

DSL provide higher speed access to the Internet, with the high speed digital communication over the existing local loops.

DSL Technology is a set of technologies
- ADSL
- VDSL
- HDSL
- SDSL

where each set is replaced by X.DSL where X is A, V, H, S.

A → Asymmetric DSL
V → Very high bit rate DSL
H → High bit rate DSL
S → Symmetric DSL

ADSL

ADSL :- It is like a 56K modem, provides higher speed (bit rate) in the downstream direction (from Internet to the resident) than in the upstream dirn. i.e. the reason it is called asymmetric.

ADSL is an asymmetric communication technology designed for residential users; it is not suitable for business.

Using existing local loops:- It uses existing local loops. It is capable of handling bandwidth upto 1.1 MHz using the twisted pair local loop but the filter installed at the end office of the telephone company where each local loop is limits the bandwidth to 4kHz.

Adaptive Technology:-

Factors affecting the bandwidth of ADSL are
→ distance b/w the residence & the switching office
→ Size of the cable.
→ Signalling used & so. on.

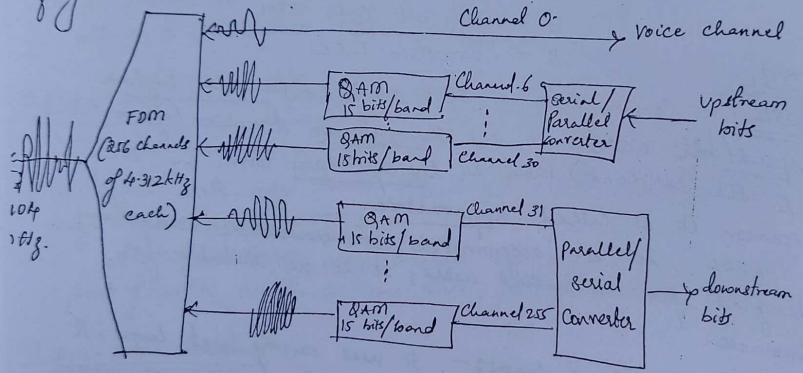
∴ An adaptive technology test the condition & bandwidth availability of the line before the settling on data rate.

ADSL is an adaptive technology. The system uses a data rate based on condition of the local loop line.

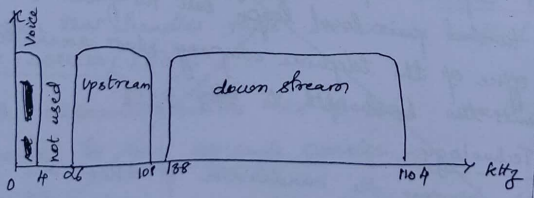
Discrete Multitone technique (DMT) It is the standard modulation technique used in DMT which combines QAM & FDM (Quadrature Amplitude Modulation) (frequency division multiplexing)

Each s/m decides on its bandwidth division.

Available bandwidth of 1.104 MHz is divided into 256 channels Each channel uses a bandwidth of 4.312 kHz as shown in fig



Bandwidth can be divided ~~into~~ as shown in fig.



Voice :- Channel 0 is reserved for voice communication

Idle :- Channel 1 to 5 are not used & provides a gap b/w voice & data communication.

Upstream data & control :-

Channel 6-30 (25 channels) are used for upstream data & transfer & control.

1 channel → for control

24 channels → for data transfer.

24 channels each using 4kHz (out of 4.312 kHz) with QAM modulation, we have $24 \times 4000 \times 15$ or a 1.44 Mbps bandwidth in upstream.

The data rate is normally below 500kbps :: some carriers are deleted at frequencies where noise level is large. i.e some channels are not used.

Downstream data and control :- Channels 31 to 255 (225 channels) are used.

1 channel is used for control

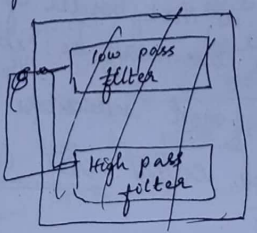
224 channels are used for data.

$224 \times 4000 \times 15$, or 13.4 Mbps.

Datarate is normally below 8 Mbps :: some channels may be unused.

Customer Site : ADSL modem :-

The figure below shows the ADSL modem installed at customer site. Local loops connects to a splitter which separates voice & data communication. The modem, modulates & demodulates the ~~downstream~~ data using DMT & creates downstream & upstream channels.



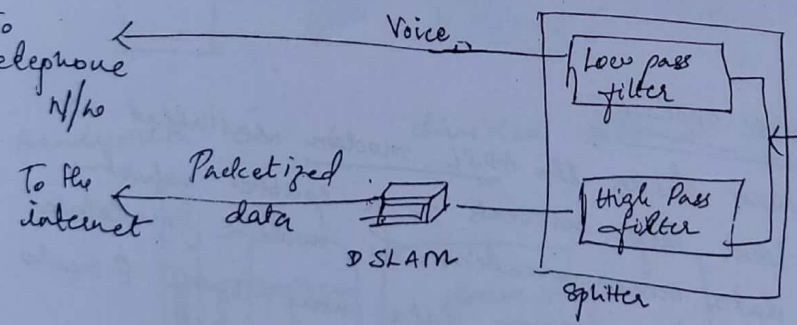
splitter

ADSL modem

ADSL line is expensive as splitter needs to be installed t the customer premises by a technician, Voice line can use he existing home wiring but data line needs a professional install. This led to an alternate technology Universal DSL (or ADSL lite).

Telephone Company Site : DSLAM

Instead of ADSL modem in a telephone company a device called a digital subscriber line access multiplexer (DSLAM) is installed. It also packetizes the data to be sent to the Internet. The configuration is shown below.



ADSL Lite (Universal ADSL or splitterless ADSL)

The installation of splitters at the border of the premises the new wiring for the data line can be expensive & impractical enough to dissuade most subscribers. Thus a newer version of ADSL technology called ADSL lite is available for subscribers.

ADSL lite modem is plugged in directly to the telephone jack & connected to computer.

It uses 256 DMT carriers with 8 bit modulation (instead of 15 bit). It can provide maximum ~~downspeed~~ stream data rate of 1.5 Mbps and an upstream data rate of 512 kbps.

HDSL [High bit rate digital subscriber line]

It was designed as an alternative to the T-1 line (1.544 Mbps) which uses alternate mark inversion (AMI) encoding, which is very susceptible to attenuation at high frequencies which limits the length of T-1 line to 3200 ft (1 km). For longer distances a repeater is necessary which increased the cost.

HDSL uses 2B1Q encoding which is less susceptible to attenuation. Data rate of 1.544 Mbps (Sometimes upto 2 Mbps) can be achieved without repeaters upto a distance of 12000 ft (3.86 km). It uses 2 twisted pairs to achieve full-duplex transmission.

SDSL → One twisted pair version of HDSL. It provides full duplex symmetric communication supporting upto ~~768~~ kbps in each direction. It provides symmetric communication alternative to ADSL.

VDSL → [Very high speed DSL] : An alternative approach to ADSL, uses co-axial, fibre-optic or twisted pair cable for short distances with DMT modulation techniques.

Range of bit rates (25 to 55 Mbps) for ~~down~~ upstream communication at distances of 3000 to 10000 ft.

~~Down~~ Upstream rate is normally 3.2 Mbps.

Summary:-

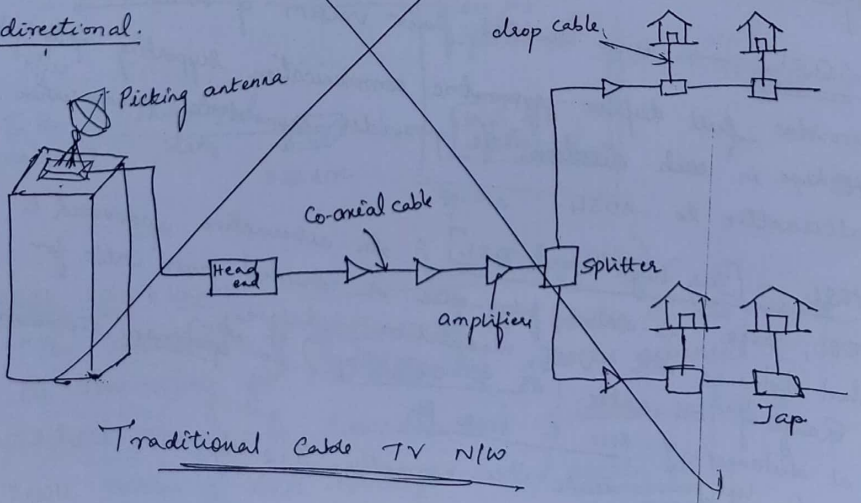
Technology	Downstream rate	Upstream rate	Distance (ft)	Twisted pairs	Line code
ADSL	1.5 - 6.1 Mbps	16-640 kbps	12,000	1	DMT
ADSL lite	1.5 Mbps	500 kbps	18,000	1	DMT
HDSL	1.05 - 2.0 Mbps	1.5 - 2.0 Mbps	12,000	2	2B1Q
SDSL	768 kbps	768 kbps	12,000	1	2B1Q
VDSL	25-55 Mbps	3.2 Mbps	3000-10,000	1	DMT

Cable TV Networks

Traditional Cable TV n/ws

Cable TV started to distribute broadcast video signals to locations with poor or no reception in the late 1940s. It was called community antenna TV (CATV) because an antenna at the top of a tall hill or building received the signals from the TV stations & distributed them, via co-axial cables, to the community.

Communication in the traditional cable TV network is unidirectional.



Traditional Cable TV N/W

Data Link Control : Chapter 11

2 main functions of the data link layer are data link control & media access control.

Data link control deals with the design and procedures for communication b/w 2 adjacent nodes: node-node communication.

Data link ~~layer~~ control functions include framing, flow & error control, & is implemented protocols that provide smooth & reliable transmission of frames b/w nodes.

Framing :-

Data link layer need to pack the bits into frames, so that each frame is distinguishable from another.

→ Framing in the data link layer separates a message from one source to a destination, or from other messages to other destinations, by adding a sender address & a destination address.

→ The sender address helps the receipient to acknowledge the receipt. The destination address defines where the packet is to go.

→ The whole msg can be packed in one frame, but this is not done because a very large frame can make the flow and error control very inefficient.

→ When a msg is carried in / very large frame, even a single bit error would require the re-transmission of the whole message. When msg is divided into smaller frames, a single bit error affects only that small frame.

Fixed-Size Framing :-

In fixed size framing there is no need for defining the boundaries of the frames; the size itself can be used as de-limiter.

Ex :- ATM wide area network → uses frames of fixed size called cells.

Variable-Size Framing :

It is prevalent in local area networks. In this we have to define the end of the frame & beginning of next.

Two approaches used in varible size framing are
- character oriented approach
- bit-oriented approach.

Character-Oriented Protocols :-

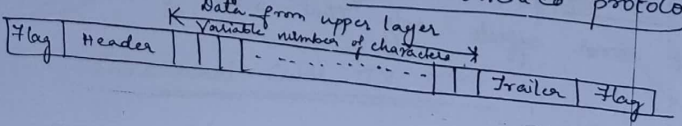
→ Data to be carried are 8-bit characters from a coding system such as ASCII.

→ Header normally carries the source & destination addresses & other control info.

→ Trailer carries error detection or error correction redundant bits, are also multiples of 8 bits.

→ To separate the frames a 8-bit (1 byte) flag is added at the beginning & the end of the frame. The flag composed of protocol-dependent special characters, signal the start of end of a frame.

→ The frame format in a character-oriented protocol is



→ It is popular when only text was exchanged by data link layers. We send other types of info such as graphs audio & video.

→ The flag can be any character not used in the text for communication. If any pattern of the flag is in the part of the information, when the receiver encounters this pattern it thinks that it is the end of the frame.

To fix this problem a byte-stuffing strategy was added to character-oriented framing.

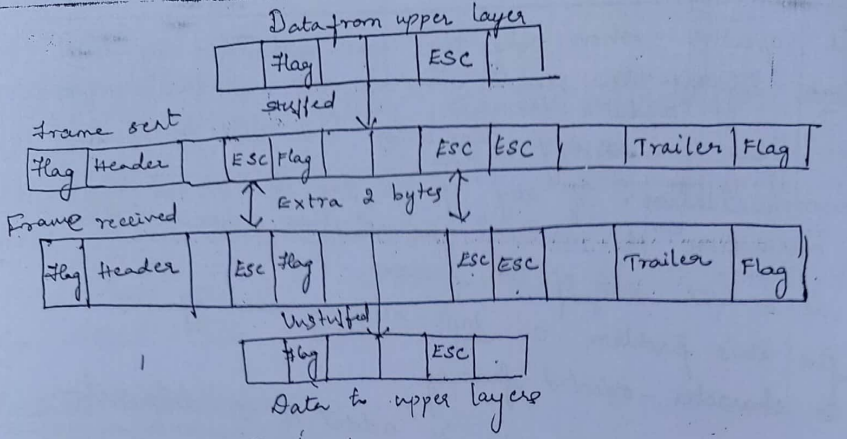
In byte stuffing special byte is added to the data section of the frame when there is character with the same pattern as the flag. The data section is stuffed with an extra byte called the escape character (ESC) which has a predefined bit pattern.

Whenever escape character is encountered by the receiver it removes it from the frame & treats the next character as data.

If the text contains one or more escape characters followed by a flag, the receiver removes the escape character but keeps the flag, which is incorrectly interpreted as end of the frame. To solve this problem the escape characters that are part of text must also be marked by another escape character. i.e if escape character is a part of the text an extra one is added to show that second one is part of the text.

Byte stuffing is the process of adding 1 extra byte whenever there is a flag or escape character.

The universal coding systems we use today such as Unicode, have 16 bit & 32 bit characters that conflicts with the 8 bit characters in character oriented protocols.

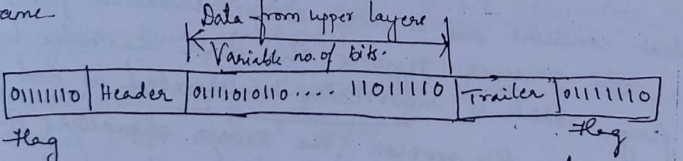


Bit oriented Protocols :-

In a bit oriented protocol, the data section of the frame is a sequence of bits to be interpreted by the upper layer as text, graphic, audio, video & so on.

→ With the headers we need a delimiter to separate one frame from the other.

→ The delimiter used in most of the protocols are a special 8-bit pattern 01111110 to define the beginning & end of the frame.



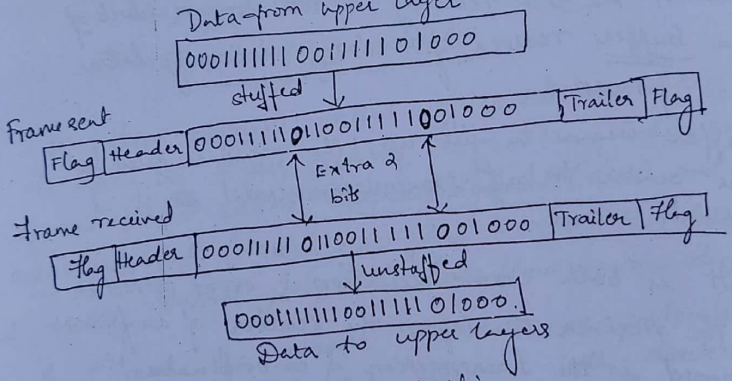
A frame in a bit oriented protocol

The existence of flag pattern in the data may cause the receiver to consider it has the end of the frame ∴ Receiver must be informed that is not the end of the frame by stuffing 1 single bit to prevent the pattern from looking like a flag. This strategy is called bit stuffing.

In bit stuffing if a '0' & 5 consecutive '1's are encountered an extra '0' is added. This extra bit will be removed at the receiver.

Bit Stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data so that the receiver doesn't mistake the pattern 01111110 for a flag.

i.e If flag 01111110 appears in data it will change to 011111010 (stuffed) & not taken as flag by receiver



Bit Stuffing & unstuffing

Flow And Error Control

The most important responsibilities of the data link layer are **flow control** & **error control**. Collectively these functions are known as **datalink control**

Flow Control :- Flow control co-ordinates the amount of data that can be sent before receiving an acknowledgement & is one of the most important duties of D.L. layer - Flow control refers to a set of ~~protocols~~ procedures need to

restrict the amount of data that the sender can send before waiting for acknowledgement.

→ The flow of data must not be overwhelmed at the receiver

→ Receiving device has a limited speed at which it can process incoming data & a limited amount of memory to store the incoming data. The receiving device must be able to inform the sending device before these limits are reached & to request that the transmitting device to send fewer frames or stop temporarily

→ Incoming data must be checked & processes before they can be used. The rate of such processing is slower than the rate of transmission ∴ each receiving device has a block of memory called a buffer reserved for storing incoming data until they are processed.

→ If the buffer begins to fill up, the receiver must be able to tell the sender to halt transmission until it is once again able to receive.

Error Control :- It is both error detection & error correction

→ It allows the receiver to inform the sender of an frames lost or damaged in the transmission & co-ordinates the re-transmission of those frames by the sender.

→ Error control primarily with error detection & re-transmission in Data Link Layer.

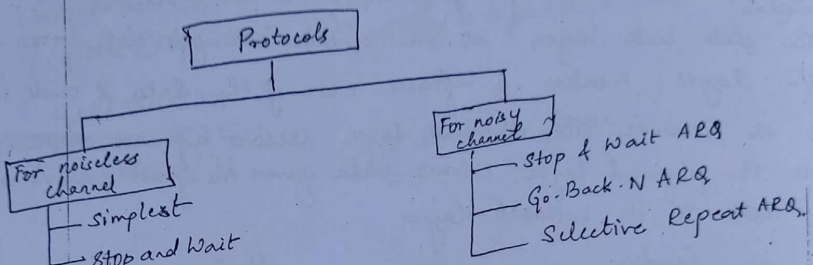
→ Any time an error is detected in an exchange, specified frames are retransmitted. This process is called automatic repeat request (ARQ)

Protocols:- The protocol using one

Protocols:-

The protocols are normally implemented in software by using one of the common programming languages.

Protocols can be used for
- Noiseless [error-free] channels
- Noisy [error-creating] channels.



Taxonomy of Protocols

All these protocols we are discussing are unidirectional. ie from the sender node to the receiver node except for the special frames called acknowledgement (ACK) and negative acknowledgement (NAK) can flow in opposite direction for flow & error control purposes. data flow in one direction.

In real life network the data link protocols are implemented as bi-directional, data flow in both directions - In these protocols the flow & error control information such as ACK & NAK is included in the data frames in a technique called piggybacking.

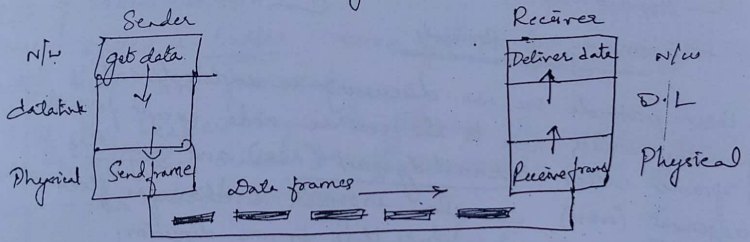
Noiseless Channels

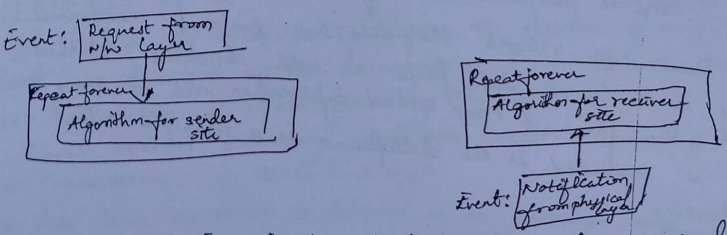
If the channel is perfect then we say it is noiseless channel.

Simplest Protocol:- It is the protocol which has no flow or error control. It is a unidirectional protocol ie from sender to receiver

→ The receiver immediately handles any frame it receives & removes the header from the frame & hands the data packet to its network layer, which also accepts the packet immediately.
→ Receiver can never be overwhelmed with incoming frames.

Design :- No need for flow control in this scheme. The data link layer at the sender site gets data from its n/w layer, makes a frame out of the data & sends it. At the receiver site datalink layer receives a frame from its physical layer extracts data from the frame, & delivers the data to its network layer





If the protocol is implemented as a procedure, we need to introduce events in the protocol. the procedures will be running continuously but action is taken only when the notification arrives [both at receiver & sender]

Algorithm

At sender site

```
While (true)                          // Repeat forever
{
  WaitforEvent();                     // Sleep until an event occurs
  if (Event(RequestToSend))           // There is a packet to send
  { GetData();
    MakeFrame();
    SendFrame();                      // Send the frame.
  }
}
```

At receiver site

```
While (true)
{
  waitForEvent();
  if (Event(ArrivalNotification))
  {
     ReceiveFrame();
     ExtractData();
     DeliverData();
  }
}
```

Analysis of algorithms → It also has an event driven algorithm

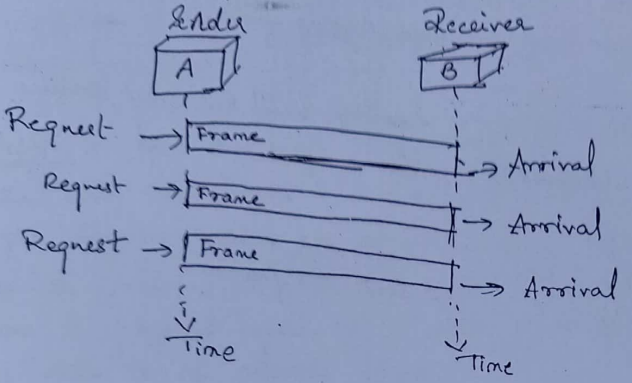
The algorithm has an infinte loop

→ event driven → it sleeps till event occurs

when the event occurs at sender the module GetData() takes a packet from n/w layer, MakeFrame() adds a header & delimiter flag to the data packet to make a frame & SendFrame() delivers frame to the physical layer for transmission.

At the Rx After the event occurs the data link layer receives the frame from physical layer using ReceiveFrame() process & extracts the data from frame using ExtractData() process & delivers the data to n/w layer using DeliverData() process

It shows an example of communication using this protocol. The sender sends a sequence of frames without even thinking about the receiver.



Stop and Wait Protocol:-

To prevent the receiver from becoming overwhelmed with frames there must be a feedback from the receiver to the sender to tell sender to slow down.

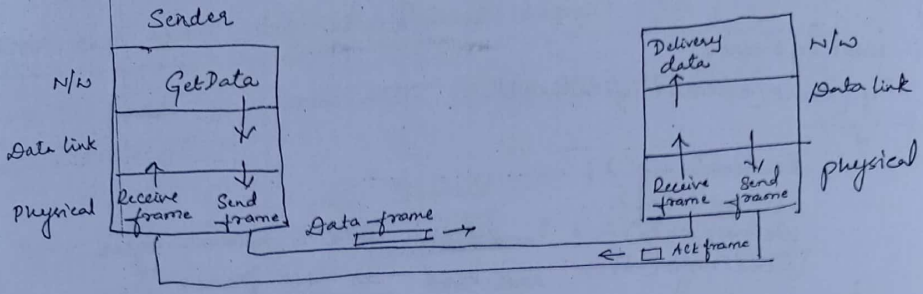
The sender sends one frame, stops until it receives confirmation from the receiver & then sends the next frame through the protocol called as STOP & WAIT protocol

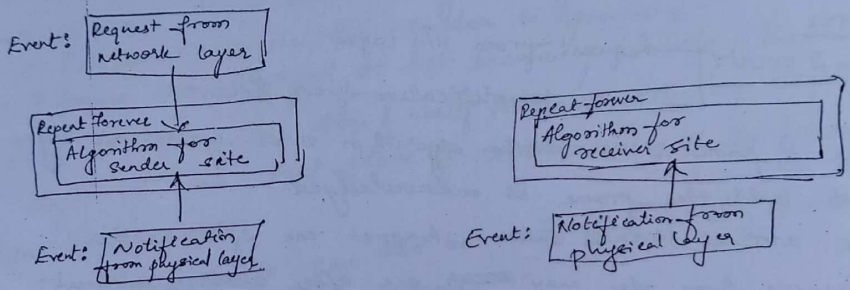
It is unidirectional for data frames but auxillary ACK frames (simple acknowledgement) travel from the other direction

flow control is added to our previous protocol.

Design :- Comparing the mechanism of this protocol with ~~stop & wait~~ Simplest protocol, we see the traffic on the forward channel & reverse channel

→ It is either one data frame on the forward channel or one ACK frame on the reverse ∴ it is half-duplex





Algorithms

At sender site:-

```
while(true)
canSend = true;
{
  waitForEvent();
  if (Event(RequestToSend) AND canSend)
  {
      GetData();
      MakeFrame();
      SendFrame();
      CanSend = false;          // cant send acknowledgement arrives
  }
  WaitForEvent();
  if (Event(ArrivalNotification))   // An Ack has arrived
  {
      ReceiveFrame();           // Receive the Ack frame
      canSend = true;
  }
}
```

At Receiver Site

```
while(true)
{
    WaitForEvent();
    if (Event(ArrivalNotification))
    {
        ReceiveFrame();
        ExtractData();
        Deliver(data);          // Deliver data to network layer
        SendFrame();            // send an Ack frame.
    }
```

Analysis

Sender → 2 events → Request from N/w layer
                  → Arrival notification from Receiver

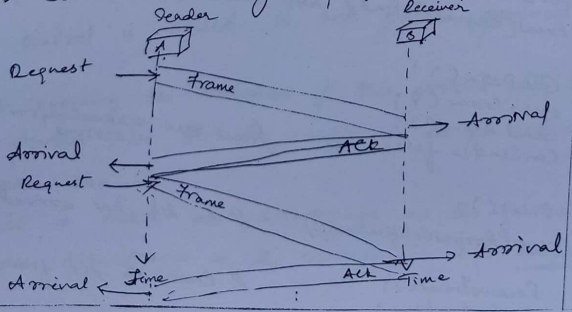
After a frame is sent, the algorithm must ignore n/w layer request until the frame is acknowledged.

→ 2 arrival events cannot happen one after another

→ Requests from N/w may occur one after another without arrival event so we consider a variable canSend → which is true when the acknowledge is received from the receiver

At Receiver → the receiver sends an ACK frame to acknowledge the receipt & allows the sender to send the next frame.

Ex:- Communication using stop & wait protocol



Noisy chan
Stop

Noisy Channels:-

STOP and WAIT Automatic Repeat Request (Stop & wait ARQ)

→ It adds a simple error control mechanism to the Stop-and Wait Protocol.

Assumptions in the Noisy Channels:-

→ data transfer is unidirectional, but half duplex
→ Both Sender & Receiver N/w layer are always ready
→ Receiver doesn't have enough storage space
→ Receiver is slower than sender in processing
→ frames are damaged or lost because of the noise in the channel

Error correction in Stop & wait ARQ is done by keeping a copy of the sent frame & re-transmitting of the frame when timer expires.

→ An ACK frame can also be corrupted & lost, it also needs redundancy bits & a sequence no.

Sequence numbers → protocol specifies that frames need to be numbered. This is done by using sequence nos.

→ In this protocol we use sequence nos to number the frames. The sequence numbers are based on modulo-2 arithmetic.

Acknowledgement no.:- It always announces the sequence number of the next frame expected by the receiver in modulo-2 arithmetic.

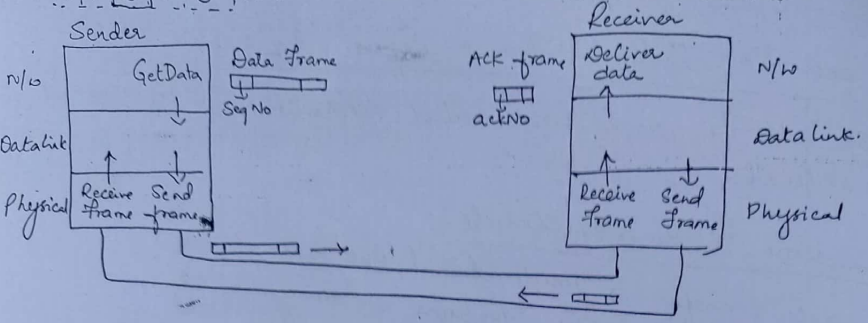
Ex:- If a frame 0 has ~~sent then the~~ arrived safe & sound the receiver sends an acknowledgment ~~frame~~ with Acknowledge 1 (which means frame 1 is expected) which is the sequence number of the next frame expected.

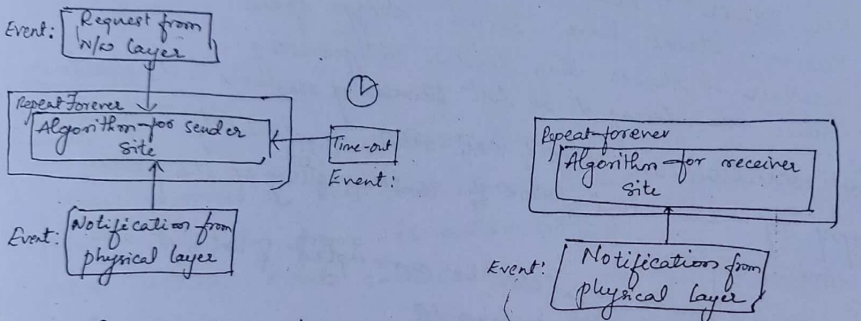
Design :- Sn next frame to send

... 0 1 0 1 0 ...

Rn Next frame to receive.

... 0 1 0 1 0 1 ...





→ Sending device keeps a copy of the last frame transmitted until it receives an acknowledgment for that frame.

→ A data frame uses a seqNo; an ACK frame uses an ackNo. The sender has a control variable Sn which holds the sequence no. for the next frame to be sent (0 or 1)

→ The receiver has a control variable, which we call Rn. that holds the number of the next frame expected.

→ When frame is sent or received, Sn & Rn are incremented respectively in modulo-2 arithmetic.

→ 3 events occurs at the sender site & 1 event occur at the receiver site.

Algorithms

Sender site algorithm

```
Sn = 0;                          // Frame 0 should be sent first
canSend = true;                  // Allow the first request to go.
while (true)
{
   WaitForEvent();
   if (Event(RequestToSend) AND canSend)
   {
      GetData();
      MakeFrame(Sn);             // The seqNo. is Sn
      StoreFrame(Sn);            // Keep copy.
      SendFrame(Sn);
      StartTimer();
      Sn = Sn + 1;
      canSend = false;
   }
   WaitForEvent();
   if (Event(ArrivalNotification))        // An ACK has arrived
   {
      ReceiveFrame(ackNo);                // Receive the ACK frame
      if (not corrupted AND ackNo == Sn) // Valid ACK.
      {
         StopTimer();
         PurgeFrame(Sn-1);        // Copy is not needed.
         canSend = true;
      }
   }
   if (Event(TimeOut))                    // Timer expired
   {
      StartTimer();
      ResendFrame(Sn-1);          // Resend a copy check
   }
```

Receiver-Site algorithm

```
Rn = 0;                                   // Frame 0 expected to arrive first
while (true)
{
    WaitForEvent();
    if (Event(ArrivalNotification))
    {
        ReceiveFrame();
        if (corrupted(frame))
            Sleep();
        if (seqNo == Rn)                  // Valid Data frame.
        {
            ExtractData();
            DeliverData();
            Rn = Rn + 1;
        }
        SendFrame(Rn);
    }
}
```

→ The frame is stored until it reaches the receiver safe. This copy is used for resending the corrupt or lost frame.

→ canSend variable is used to prevent the network layer from making a request before the previous frame is received safe & sound.

→ Once the ACK frame is received at the sender the stored frame is purged & the timer is stopped. Frame is resent when the timer expires.

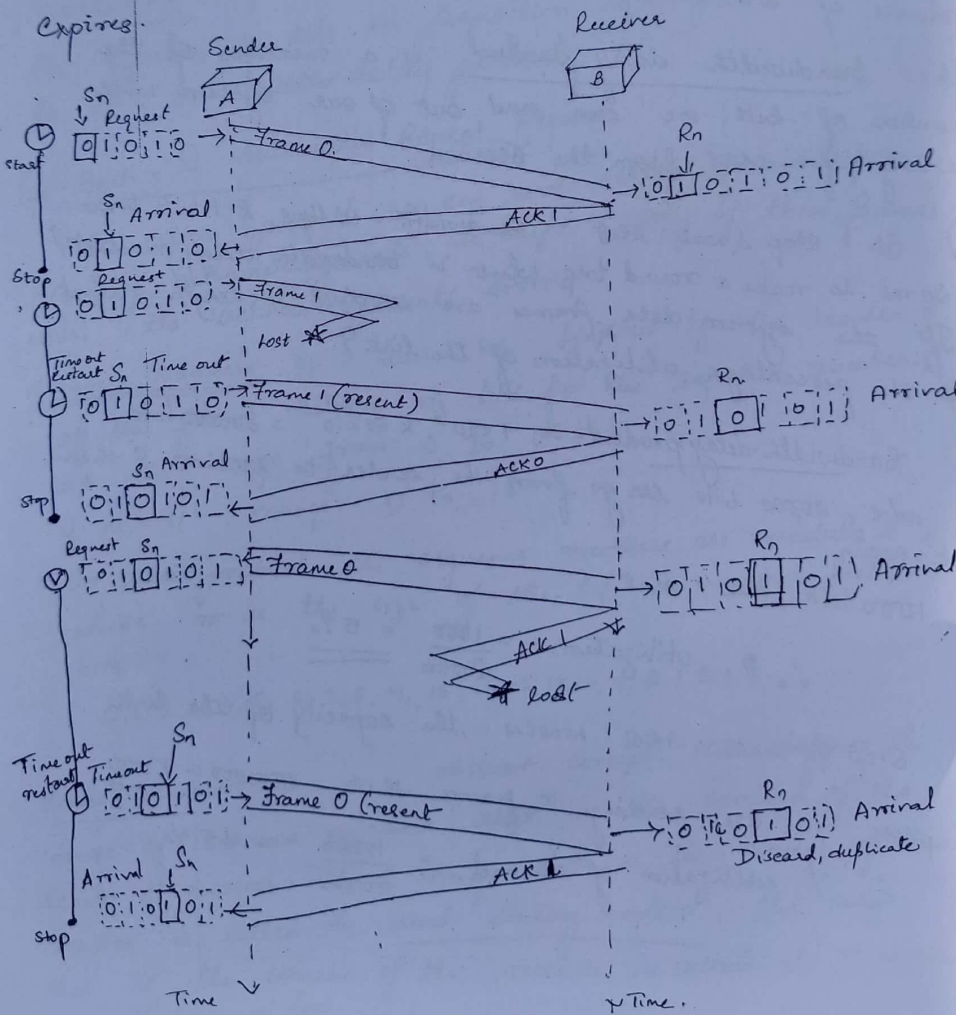
→ At the receiver even if the sequence no. of the data frame does not match the next frame expected, an ACK frame is sent to the sender to reconfirm the previous ACK instead of confirming the frame received.

Example :- Flotille diagram is shown in the fig. below.

Frame 0 is sent & acknowledged.

Frame 1 → lost & resent after time-out. Resent frame is acknowledged & the timer stops.

Frame 0 is sent & acknowledged but the acknowledgement is lost, the sender has no idea whether the frame or ack. is lost so it reserds the frame 0 after timer expires.



Efficiency :-

Stop & wait ARQ is very inefficient if our channel is thick & long

thick → our channel has a large bandwidth

long → round-trip delay is long.

The product of these two are called bandwidth-delay product

We think channel as a pipe & delay product is the volume of bits in pipe.

The bandwidth delay product is a measure of the number of bits we can send out of our system while waiting for news from the receiver.

ex1) In Stop & wait ARQ Bandwidth = 1 Mbps & 1 bit takes 20 ms to make a round trip. What is bandwidth delay product? If the system data frames are 1000 bits in length, what is the percentage utilization of the link?

Bandwidth-delay product = $1 \times 10^6 \times 20 \times 10^{-3} = 20{,}000$ bits

i.e 20,000 bits can go from the sender to receiver & then back again.

1000 bit are sent

$\therefore$ % utilization $= \frac{1000}{20000} = 5\%$

Stop & wait ARQ wastes the capacity of the link.

Suppose we are sending 15 frames then $15 \times 1000 = 15000$ bits

$\therefore$ % utilization of channel $= \frac{15000}{20000} = 75\%$

Pipelining

A task is begun before the previous task has ended. This is known as pipelining

In ~~Stop~~ Go-Back N Automatic Repeat Request & Selective Repeat ARQ ~~the use~~ protocols uses pipelining

Pipelining improves the efficiency of the transmission if the number of bits in transition is large with respect to the bandwidth-delay product.

Go-Back-N Automatic Repeat Request:-

In this protocol we can send several frames before receiving acknowledgments, we keeps a copy of these frames until the acknowledgments arrive.

Sequence Numbers. It must be limited. If the header of the frame allows m bits for the sequence number, with the range from 0 to $2^m - 1$

if $m = 4$ range 0 to 15.

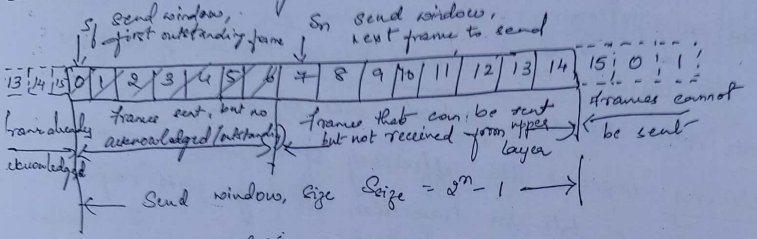
In this protocol the sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 0, 1, 2, 3 ----

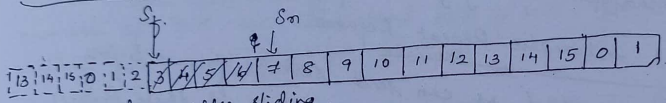
Sliding window is an abstract concept that defines the range of sequence numbers that is the concern of the sender & receiver. The range which is the concern of the sender is called the send sliding window, the range that is the concern of the receiver is called the receive sliding window

Send window is the imaginary box covering the sequence nos. frames which can be in transit. The maximum size of the window is $2^m - 1$

Consider a sliding window with size $m = 4$ i.e 15



Send window before sliding



Send window after sliding.

The window at any time divides the possible sequence numbers into 4 regions.

1st Region :- left most region belongs to frames that are already acknowledged.

2nd Region :- frames that are sent & status is unknown. These are outstanding frames

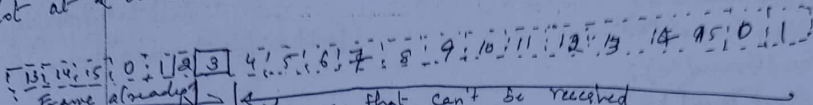
3rd Region :- frames that can be sent

4th Region :- defines frames that cannot be used until window slides.

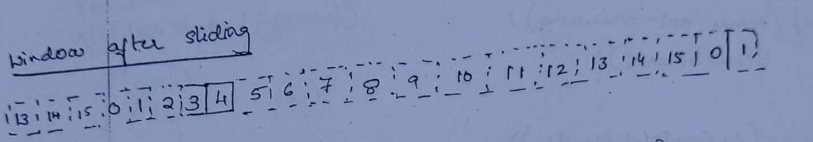
→ The send window is an abstract concept defining an imaginary box of size $2^m - 1$ with 3 variables Sf (first outstanding frame) Sn (next frame to be sent) & Ssize (send window size)

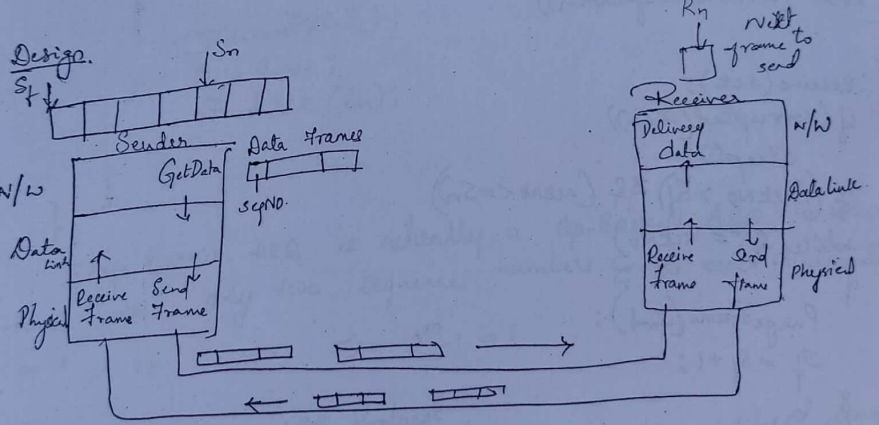
→ The send window can slide one or more slots when a valid acknowledgment arrives

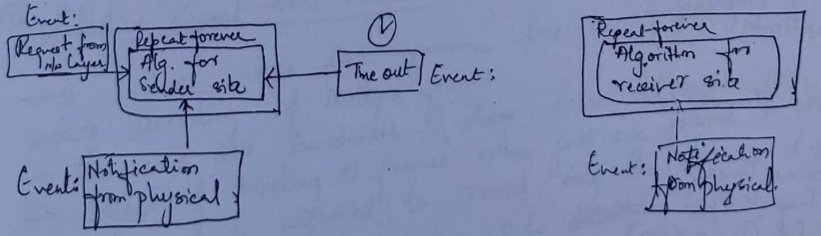
→ The receive window is an abstract concept defining an imaginary box of size 1 with one single variable R_n. The window slides when a correct frame has arrived; sliding occurs one slot at a time.



Window after sliding







Sender - Site Algorithm

```
Sw = 2^m - 1;
Sf = 0;
Sn = 0;
while (true)
{
    waitForEvent();
```

```
if(event(requestToSend))
{
    if (Sn - Sf >= Sw)          // if window is full
        Sleep();
    GetData();
    MakeFrame(Sn);
    StoreFrame(Sn);
    SendFrame(Sn);
    StartTimer();
    Sn = Sn + 1;
    if (timer not running);
        StartTimer();
}
if (Event (ArrivalNotification))
{
    Receive (Ack);
    if (corrupted(Ack))
        Sleep();
    if ((ackNo > Sf) && (ackNo <= Sn))
    while(Sf <= ackNo)
    {
        PurgeFrame(Sn-1);
        Sf = Sf + 1;
    }
    StopTimer();
}
if (Event (Timeout))
{
    startTimer();
    Temp = Sf;
    While (Temp < Sn)
    {
        SendFrame (Sf)
        Sf = Sf + 1
    }
}
```

Receiver-site algorithm :-

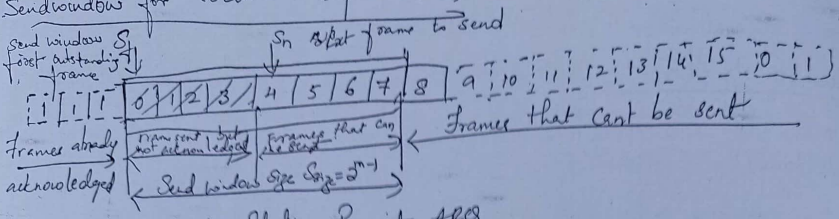
```
Rn = 0;
while (true)
{
  waitForEvent();
  if (event (ArrivalNotification))
  {
    ReceiveFrame();
    if (corrupted (frame))
      sleep();
    if (SeqNo == Rn)
    {
      DeliverData();
      Rn = Rn + 1;
      Send ACK (Rn);
    }
  }
}
```

Stop & wait ARQ is actually a Go-Back-N ARQ in which there are only two sequence numbers & the send window size is 1, i.e. $m = 1 \Rightarrow 2^m - 1 = 1$

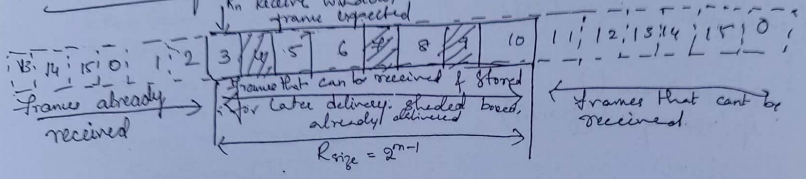
iii) Selective Repeat ARQ Protocol

→ In a noisy link frame has a probability of damage means resending of multiple frames
→ This uses more bandwidth & slows down the transmission.
→ Instead of sending N frames when just one frame is damaged, a mechanism is required to resend only the damaged frame
→ Sequence numbers range from 0 to ~~(2^m - 1)~~ 2^{m-1}
→ Both send & receive window size = 2^{m-1}

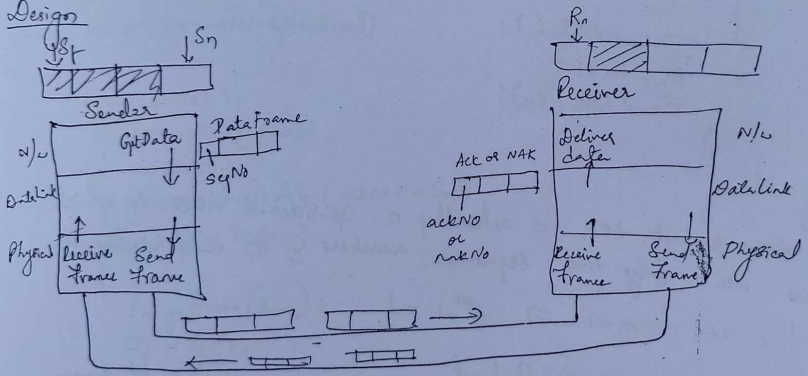
Send window for Selective Repeat ARQ

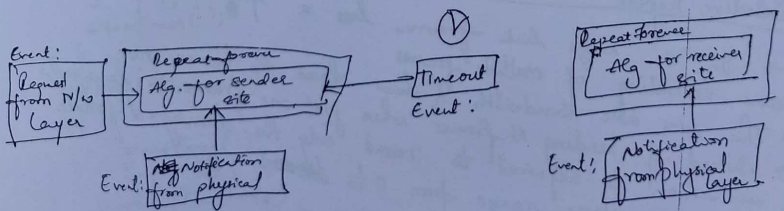


Receive window for Selective Repeat ARQ



Design





Algorithms

Sender-site algorithm

```
Sw = 2^m - 1 ;
Sf = 0;
Sn = 0;
while(true)
{
    waitForEvent();
    if (Event(requestToSend))
    {
        if (Sn - Sf >= Sw)
            Sleep();
        GetData();
        MakeFrame(Sn);
        StoreFrame(Sn);
        SendFrame(Sn);
        Sn = Sn + 1;
        StartTimer(Sn);
    }
    if (Event(ArrivalNotification))
    {
        Receive(frame);
        if (corrupted(frame))
            sleep();
        if (FrameType == NAK)
            if (nakNo between Sf & Sn)
            {
                resend(nakNo);
                StartTimer(nakNo);
            }
        if (FrameType == ACK)
            if (ackNo b/w Sf & Sn)
            {
                while(Sf <= ackNo)
                {
                    purge(Sf);
                    stopTimer(Sf);
                    Sf = Sf + 1;
                }
            }
    }
    if (Event(Timeout(t)))
    {
        startTimer(t);
        SendFrame(t);
    }
}
```

Receiver site algorithm

```
Rn = 0;
NakSent = false;
ackNeeded = false;
Repeat (for all slots)
  marked (slot) = false;
while (true)
{
wait For Event ();
if (event (Arrival Notification))
{
   Receive Frame ();
   if (corrupted (frame) && NOT nakSent)
   {
      sendNAK (Rn);
      nakSent = true;
      sleep();
   }
if (seqNo <> Rn && NOT nakSent)
{
    send Nak (Rn);
    nak Sent = true;
    if ((SeqNo in window) && (!marked(seqNo)))
    {
       store frame (seqNo);
       Marked (seqNo) = true;
       while (Marked (Rn))
          {
            Deliver data (Rn);
            purge (Rn);
            Rn = Rn + 1;
            ackNeeded = true;
          }
       if (ackNeeded)
          {
            Send Ack (Rn);
            ackNeeded = false;
            nakSent = false;
          }
    }
}
}
```

Flow control of Selective Repeat ARQ

Piggybacking :-

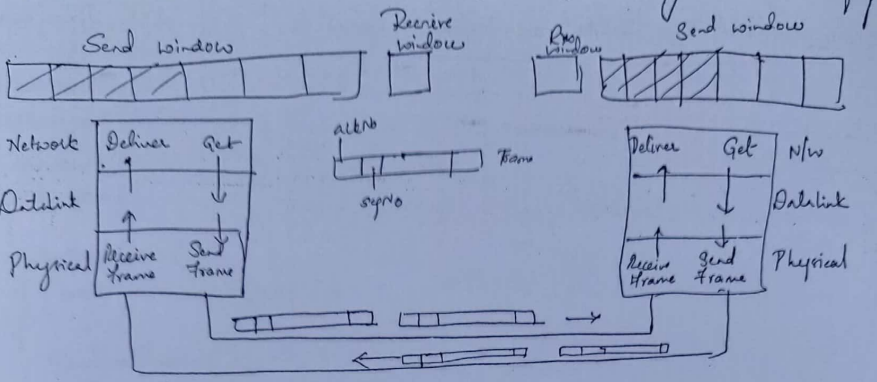
All the 3 protocols we studied are unidirectional i.e data frames flow in only one direction although control information such as Ack & NAK frames can travel in other direction.

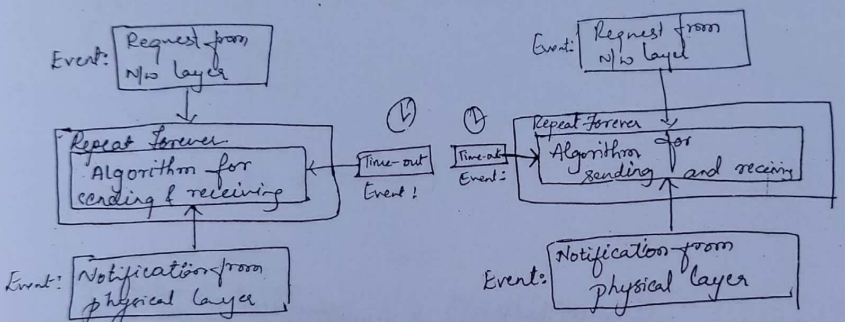
In real life data frames are normally flowing in both dirn from node A to node B and from node B. to node A. i.e control infn must also flow in both directions.

A technique called piggybacking is used to improve the efficiency of the bi-directional protocols.

When a frame is carrying data from A to B, it can also carry control infn about arrived (or lost) frames from B; IIIly when a frame is carrying data from B to A. it carries control infn about the arrived frames from B.

Design for the Go-Back-N-ARQ using Piggybacking is shown in fig.





Each node has 2 windows — Send window, receive window.

Both uses a timer with events — Request, Arrival, Time-out

Arrival Event here is complicated when a frame arrives, the site needs to handle control information as well as the frame itself. Both of which uses only event i.e arrival event. The request event uses only the send window at each site; the arrival event needs to use both the windows.

Piggybacking uses the same algorithm at both the sites.

HDLC [High level Data link Control]

HDLC is a bit oriented protocol for communication over point to point and multipoint links.

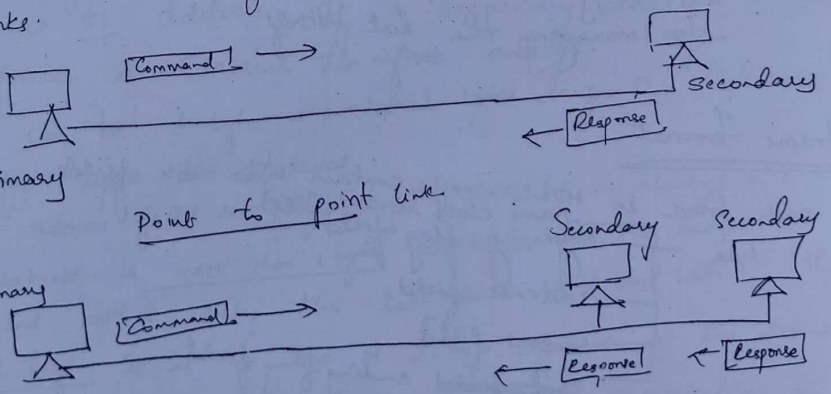
Configuration & Transfer Modes

2 common transfer modes that can be used in different configuration.

① **Normal Response Mode (NRM)**

Station configuration is unbalanced. We have one primary station & multiple secondary stations. A primary station can send commands, a secondary station can only respond.

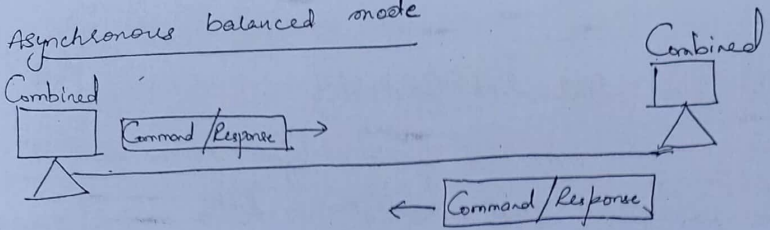
It is used for point to point & multi-point links.



Multipoint links

② **Asynchronous Balanced Mode**:- configuration is balanced. The link is point-point & each station can function as a primary & a secondary (acting as peers). This is the common mode today

Asynchronous balanced mode



Frames

HDLC defines 3 types of frames

① Information frames [I-frame] :- used to transport user data and control information related to user data (piggybacking)

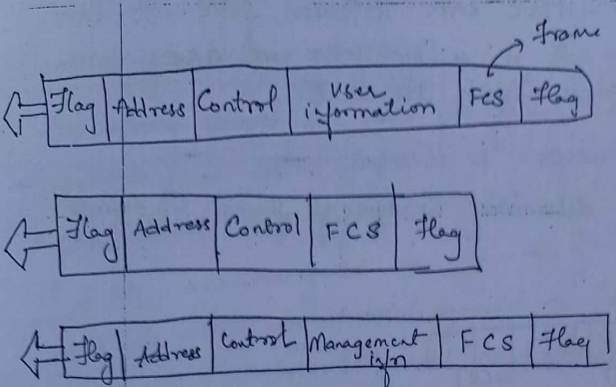
② Supervisory frames [S-frames] :- used to transport control information

③ Unnumbered frames [U-frames] :- reserved for system management. Info carried by U-frames is intended for managing the link itself.

Frame Format

Each frame in HDLC may contain upto six fields. It has
- beginning flag field
- address field
- control field
- frame check sequence (FCS)
- ending flag field

In multiple frames the ending flag of one frame can serve as the beginning flag of the next frame.



Flag Field It is an 8-bit sequence with the bit pattern 01111110 that identifies both beginning & the end of a frame & serves as a synchronization pattern for the receiver.

Address Field It contains the address of the secondary station. If a primary station created the frame it contains 'to' address. If a secondary station creates the frame it contains the 'from' address.

→ It can be 1 byte or several bytes long. One byte can identify upto 128 stations.

→ If address field is only 1 byte the last bit is always a '1'

→ If address is more than 1 byte all bytes but the last one will end with '0'; only the last byte will end with '1'

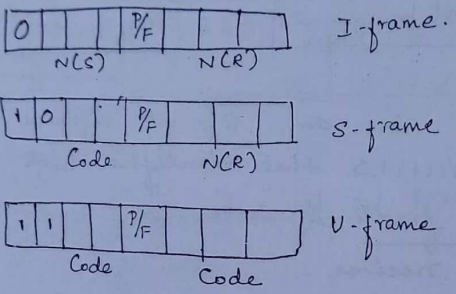
Control Field:- It is a one or 2-byte segment of the frame used for flow and error control. Its interpretation of bits differ from 1 frame type to other

Information field:- It contains the users data from N/W layer or management information. Its length can vary from one N/w to another.

FCS field → It is HDLC error detection field. It can contain either a 2 or 4 byte ITU-T CRC.
Frame check sequence.

Control Field

The control field determines the type of frame & defines its functionality.



Control Field for I-frames

I-frames include flow & error control info (piggybacking)

1st bit defines byte — '0' means of the frame is a I-frame.

next 3rd bits → N(S) → sequence number of the frame. with 3 bits we can define 0 to 7 but in extension format, in which the control field is 2 bytes, this field is larger.

last 3rd bits → N(R) → acknowledge no. when piggybacking is used

P/F bit:- Single bit b/w N(S) & N(R) is called P/F bit.

Poll when the frame is sent by a primary station to a secondary. (contains address of the receiver)

Final when the frame is sent by a secondary to a primary (contains address of the sender).

Control field for S-frames

Supervisory frames are used for flow & error control whenever piggybacking is either impossible or inappropriate.

→ They do not have information fields.

1st 2 bits → control field is 10. this frame is an S-frame.

Last 3 bits — N(R) → acknowledgement no. (ACK) or negative acknowledgement no. (NAK) depending on the type of S-frame.

2 bits called code is used to define the type of S-frame

⇒ Receive Ready → (RR) → 00

→ Frames acknowledges the receipt of a safe and sound frame or group of frames.

N(R) field → acknowledgement number.

⇒ Receive not Ready (RNR) ⇒ 10 → RNR S-frame.

RR frame with additional functions. It acknowledges the receipt of a frame or group of frames, and it announces that the receiver is busy. Can't receive more frames It is a congestion control mechanism by asking the sender to slow down. N(R) → acknowledgement no.

⇒ Reject (REJ) → 01 → code subfield.

This is a NAK frame. It is a NAK that can be used in Go-Back-N ARQ to improve the efficiency of the process by informing the sender, before the sender time expires, that the last frame is lost or damaged.

N(R) → negative Acknowledgement number

⇒ Selective reject (SREJ) :- 11 → SREJ S-frame. This is a NAK frame used in Selective Repeat ARQ. The HDLC protocol uses the term selective reject instead selective repeat.

N(R) → negative acknowledgement number

Control Field for U-Frames

→ to exchange session management & control infn. b/w connected devices.

U-frames is used for system management.

Infn carried by U-frames is contained in codes included in the control field

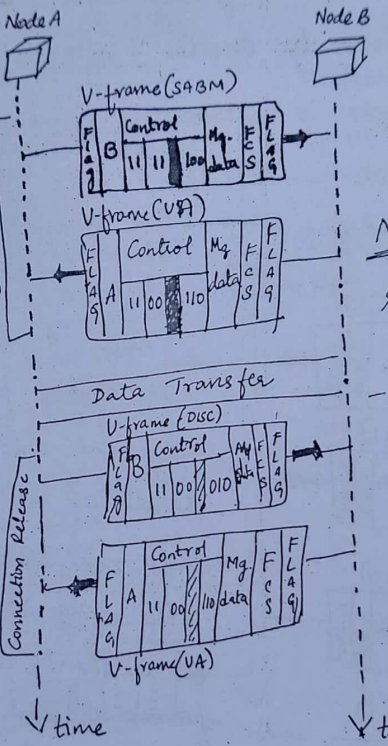
U-frame
- 2 bit prefix before P/F bit
- 3 bit suffix after P/F bit

Together these 2 segments (5 bits) can be used to create up to 32 different types of U-frames

Code	Command	Response	Meaning
00 001	SNRM		Set normal response mode
11 011	SNRME		Set normal response mode, extended
11 100	SABM	DM	Set asynchronous balanced mode or disconnect mode
11 110	SABME		Set Asynchronous balanced mode, extended.
00 000	UI	UI	Unnumbered Information
00110		UA	Unnumbered Acknowledgement
00010	DISC	RD	Disconnect or Request Disconnect
00 000	SIM	RIM	Set Initialization mode or Request information mode
00 100	UP		Unnumbered Poll
11001	RSET		Reset
11101	XID	XID	Exchange ID
10001	FRMR	FRMR	Frame Reject

Example :- Connection / Disconnection :-

U → frame → used for connection establishment and connection release.



Node-A asks for a connection with a Set asynchronous balanced mode (SABM) frame.

Node-B gives a positive response with an unnumbered acknowledgement frame (UA)

After these two exchanges, data can be transferred between the two nodes.

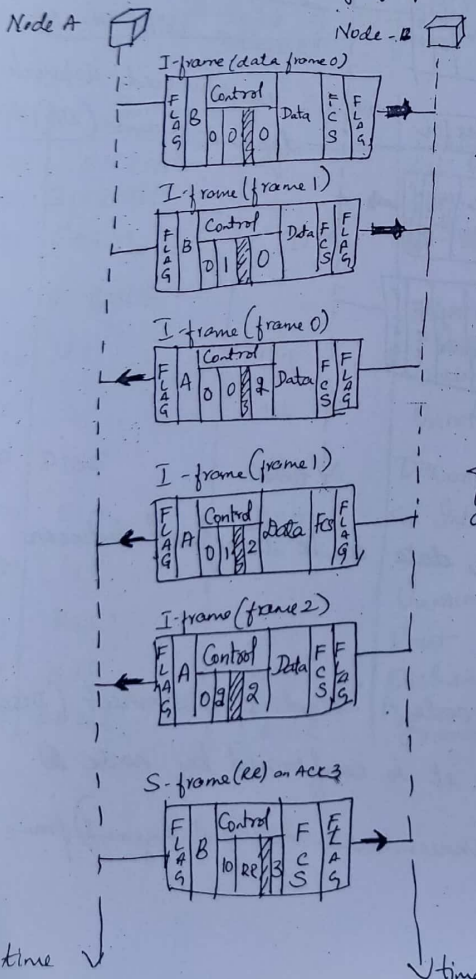
After the data transfer, node A sends a disconnect (DISC) frame to release the connection, it is confirmed by node B by responding with a UA (Unnumbered Acknowledgement) frame

Piggyb

Piggybacking without error → Fig shows an exchange using piggybacking.

Node A begins the exchange of information with an I-frame numbered 0 followed by another I-frame numbered 1.

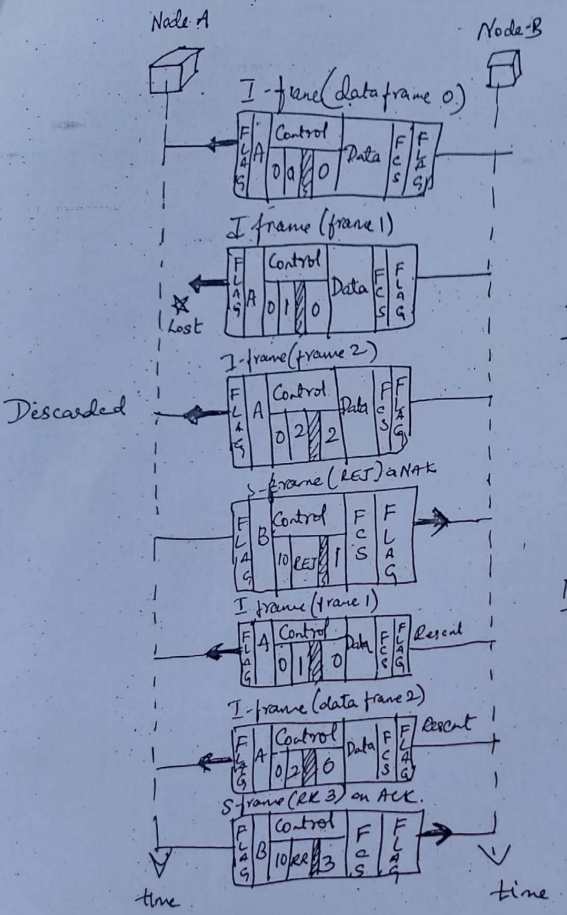
Node B piggybacks is acknowledgment of both numbers onto an I-frame of its own. → First frame with N(S) = 0, N(R) = 2 acknowledging frame 1 & 0. & expecting frame 2



Node A has sent all its data. ∴ it cannot piggyback an acknowledgement onto an I-frame and sends an S-frame instead

RR code indicates that it is still ready to receive. The number 3 is in the NCR) field tells B that frames 0, 1, and 2 have all been accepted and that A is now expecting frame 3.

Piggybacking with Error



Node B sends 3 data frames (0, 1, 2) but frame 1 is lost.

When Node A receives frame 2, it discards it and sends a REJ frame for frame 1.

Go-Back N protocol with special use of an REJ frame as an NAK frame.

NAK frame
- confirms the receipt of frame 0
- declares that frame 1 and any following frames must be resent.

Node B after receiving the REJ frame, resends 1 and 2. Node A acknowledges the receipt by sending an RR frame (ACK) with Acknowledgement No. 3