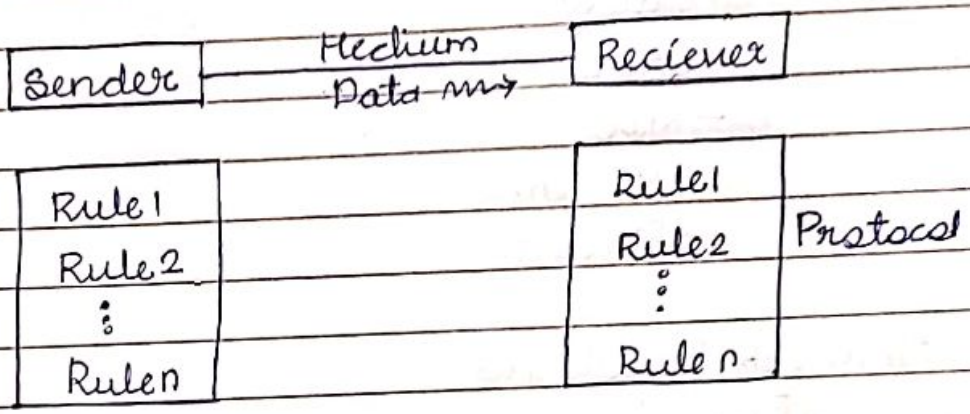
UNIT-1:

* Data communication
  - sharing of data between devices
* Data communication systems
  - consists of 5 components.



* Data flow/transmission modes
  - Simplex
  - Half duplex
  - Full duplex

Network
- set of devices connected by communication links in order to share data or to perform computational task

Network criteria
- Performance
- Reliability
- Security

Types of connections (links)
- point-to-point
- Multipoint

Networking devices
- Hubs
- Switches
- Router
- Gateway
- Bridges
- NIC
- Modem
- Firewall

Network Topologies
- Bus
- Star
- Mesh
- Ring
- Hybrid

Categories of network
- LAN (Local Area Network)
- (Metropolitan Area Network) MAN
- WAN (Wide Area Network)

OSI Reference Model

- Open System Interconnection
- Developed by ISO in 1984 (International Organization of Standardization)
- This is layered model which describes how information is transffered from one machine to other.
- It consists of 7 layers in which all the involved functions are divided into 7 distinct level with each layer performing set of dedicated functions or tasks

Application layer
Presentation layer
Session layer
Transport layer
Network layer
Data Link layer
Physical layer

1. Physical Layer
- Helps in actual transmission of data in the form of raw bits between two devices through communication links (wired / wireless)
- Hardware specification (physical cables, NIC and wireless radios) (physical, mechanical, electrical properties)
- Topology and network design
- Transmission modes
- Line configurations (P2P, mp)
- Data rate (bps)
- Synchronization at bit level
- Encoding and signalling (setting Voltage level)
  (+5 for 1 / -5 for 0)

- Data unit : bit
- Networking device : Hub, Network Interface card (NIC)
- Protocols : USB, Bluetooth
- P

2. Data Link Layer
- provides a reliable transmission of data between two devices connected by a link. This layer makes use of services or functions provided by the physical layer.
- Functions are:-
  - Flow Control → speed (achieved by waiting for acknowledgement)
  - Error control → Error Detection, Error Correction
  - Access Control (incase of multipoint links)
  - Framing
- Data unit : Frames
- Protocols : Point-to-point protocol (PPP), ATM (Asynchronous Transfer Mode)
- Devices : Switch
- Addressing Mechanism : MAC Address

3. Network Layer
- Main aim of network layer is to deliver a packets from source machine to destination machine across multiple networks (unreliable).
- other functions are:
  1) Routing : Determining how packets will be transmitted through which path. (It takes a best path) Static and dynamic (Take the path according to the network case so there is no data loss)
  2) congestion control - (handles the traffic)
  3) Internetworking
- Data unit : packet
- Devices : Router, multi layer switch
- Protocols : IP, IPV4, IPV6, ICMP.

- Addressing mechanism is IP address.

4. Transport layer
- Main aim of transport layer is to transport the message from process of source machine to the process on the destination machine.
- Identification of these processes is also handled by transport layer.
- Transport layer also handles flow and error control end to end.
- Processes are identified using port numbers.
- Total ports are 65535
  0-1023 → well known ports
  Ex: HTTP → port (80)
  FTP → port (21)
  SMTP → port
- Other functions are service point addressing, segmentation and reassembly.
- Data unit: Segment
- Device: Load Balancer, Firewall
- Protocols: TCP, UDP
- Addressing mechanism: port number

5. Session layer
- Functions
  1) session management
Main aim of session layer is to establish, maintain and to synchronize interaction between two different communicating
  2)
  3) Reestablishment of connections
  4) Synchronization using checkpoints.

5) Dialogue control (In case of video conferencing)
- Data unit: SPDU (session program data unit)
- Device: Computers
- Protocol: Net BIOS

6. Presentation Layers
- This layer is concerned with syntax and semantics transmitted between sender and reciever.
- Other functions are:
  1) Translation
  2) Encryption and Decryption
  3) Compression/Decompression.
- Data unit: PPDU (Presentation program data unit)
- Device: Computer, Encryption/Decryption Devices
- Protocols: SSL, TLS

7. Application layer
- This layer contains the application protocols which allows the user to gain access to the network (internet).
- Other functions are:
  1) mail Services
  2) Network Virtual Terminal
  3) File transfer Access Management
  4) Directory Services
- Data unit: APDU (Application program data unit)
- Protocol: TELNET, FTP, DNS, STDP.
- Device: Computer

TCP/IP Reference Model

Application layer
Transport layer
Internet layer → Network layer.
Link Layer. → physical layer + Data link layer

Critique to OSI Model and TCP/IP

1 Bad timing:
The time at which the OSI model was launched, TCP/IP model was already in use. Hence companies didn't went for new model other than TCP/IP and OSI never happened.

2. Bad Technology:
Session and presentation layer are nearly empty whereas data link and network layer are overflown. Some functions like addressing, flow control and error control reappear again and again in multiple layers.

3. Bad Implementation
Given the enormous complexity of model and protocol initial implementation of model were huge and slow. Everyone who tried them got burned (went in loss)

4. Bad Politics
OSI was widely thought to be creature of european Telecommunication and European community along with US government. But the belief was only partly used.

Critique to TCP/IP Model

1. The model does not distinguish clearly between the concepts of services, protocols and interfaces something which OSI does. Hence TCP/IP method is cannot be used as guide for designing new networks using new technologies.

2. TCP/IP model is not at all general and not suitable to describe any protocol stack other than TCP/IP to describe bluethat is completely impossible.

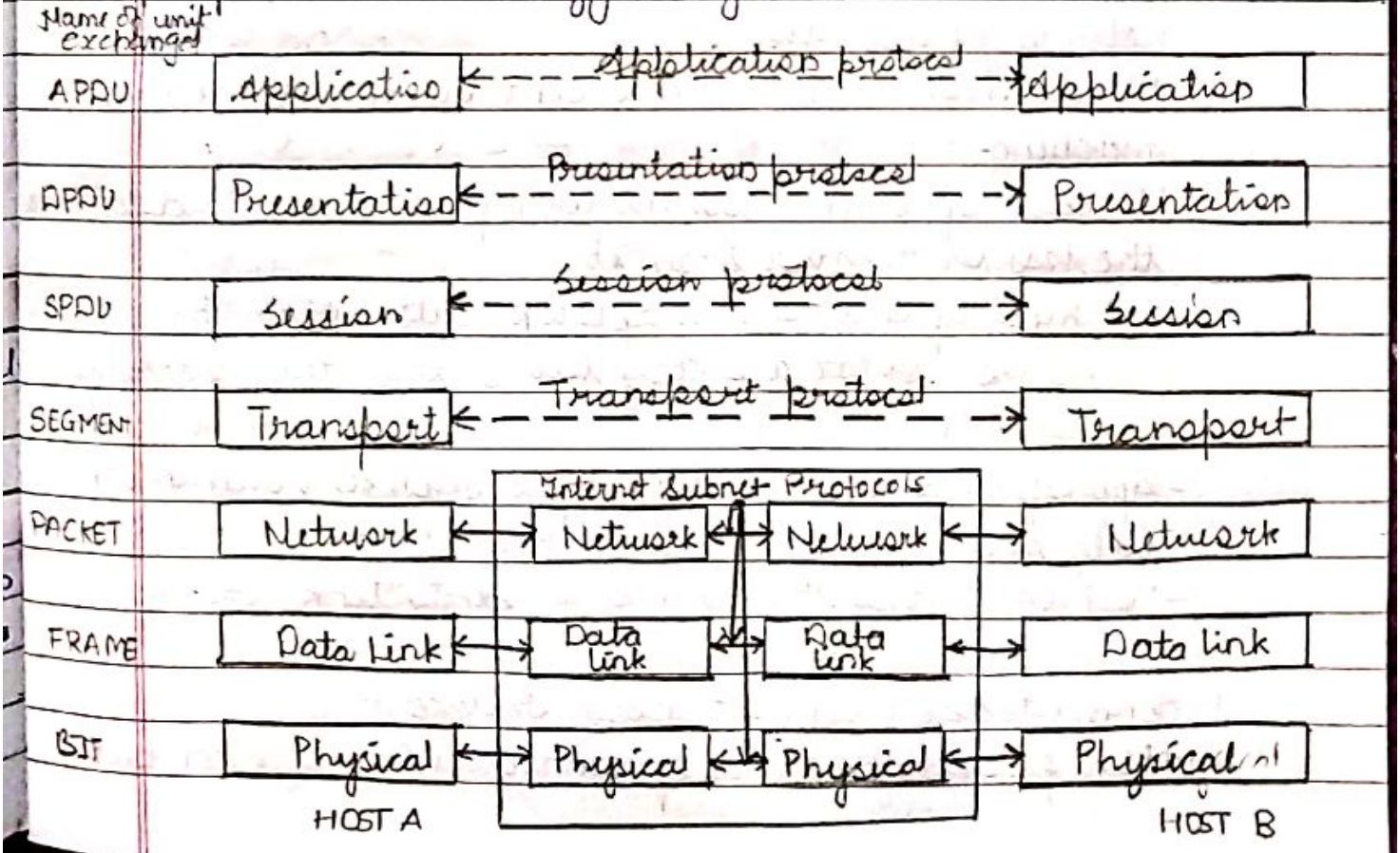
3. Link layer is an interface between network and data link layer. Not layer.

4. TCP/IP Model doesn't distinguish between physical and data link layer. Actually they are completely different layers. Physical layer deal with transmission properties whereas data link deals with framing and reliable data transfer. Proper model should include both these as seperate layers. TCP/IP model should not do this.

* OSI Model vls TCP/IP.

OSI	TCP/IP
• Open system Interconnection	• Transmission Control Protocol Internet Protocol
• 7 layers	• 4 layer.
• Invented after TCP/IP	• Invented before OSI
• Clearly distinguishes between protocols, layers interfaces and services.	• No clear distinction between layers, services and interfa
• OSI was devised before	• With TCP/IP protocols came

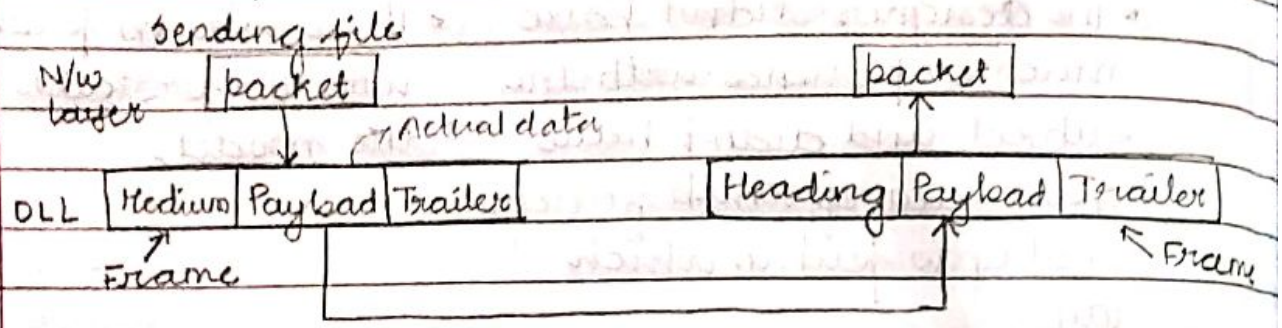
the corresponding protocols were invented. Hence this model wasn't based towards one particular set of protocol	first and then this model
• The designer didn't have much experience with the subject and didn't have good idea of which functio-nality to put in which layer.	• There was no problem with the protocols fitting the model.
• Transport layer is connection oriented	• Transport layer can be connection oriented or connectionless
• Network layer is connectionless as well as connection oriented.	• Network layer is connection-less.
• Protocols are hidden in OSI model and can be easily replaced as technology changes	• replacing the protocol is not easy.



1

Data Link Layer

* Relationship between packets and frames



* Services provided by Data link layer to network layer

1. unacknowledged connectionless services
- Source machine sends independent frames without connection setup phase hence there is no connection release phase also.
- No acknowledgments are sent by the destination machine.
- In case of frame loss, no attempt is made to detect the loss or recover from it.
- This kind of services are appropriate when the chances of error are very low so that the recovery is left to higher level.
- Appropriate for real time topic such as voice in which late data are worst then bad data.
- Most LAN's use this service in data link layer.

2. Acknowledged connectionless service
- Still follows the rules of connectionless services but is

this case, the reciever sends back an acknowledgement to the sender for each frame, thus making it reliable to some extent
- In case if sender doesn't get an acknowledgement for particular frame within certain amount of time then it resends the particular frame.
- Useful over unreliable channels such as wireless systems.

3 Acknowledged Connection-Oriented Services
- Three important phases or steps. as
i) Connection setup phase in which the the required ressources (bandwidth, buffer space, CPU cycle) is allocated in advance.
ii) Data transfer phase is which actual transmission takes place.
iii) Connection release phase in which the connection allocated are released.
- Each frame is numbered for which the reciever sends an acknowledgement back to sends.
- In case of frame loss the sender resends the same frame.

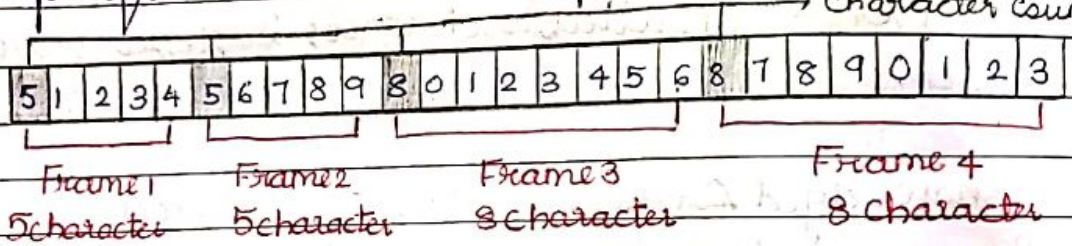
* Framing
- Dividing the continues bit strings into distinct frames. This construction of frames is an important function of DLL.
- Various framing methods are.
1. character count
2. Flag bytes with byte stuffing
3. Flag bytes with bit stuffing
4. Physical layer coding violations

5. Combination of multiple methods.

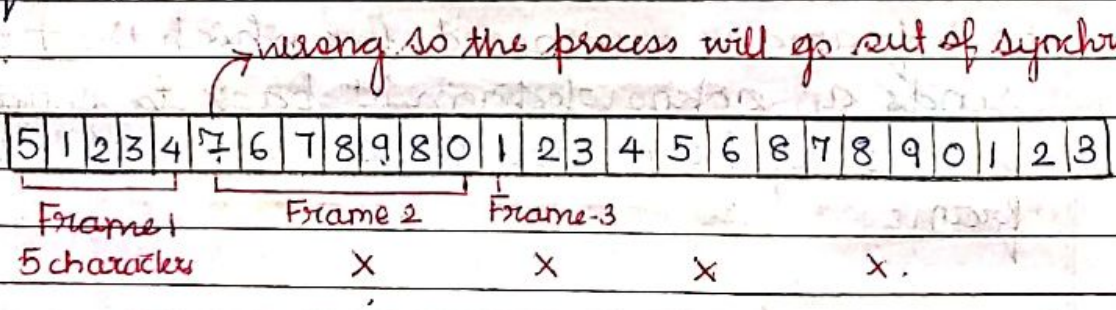
Methods:

1. Character Count

- Number of characters present in the frame are specified in the header part of that frame.



- Problem with this method is due to some transmission error or noise, the character count value can be garbled or changed. For example if character count of 5 in second frame becomes 7, then the destination will go out of synchronization and we will be unable to locate the correct start of next frame.



2. FLAG bytes with byte stuff

- Each frame starts and ends with a special byte known as a flag byte which is used as a delimiter.
  FLAG = 01111110.
- Two consecutive flag bytes indicates end of one frame and start of next frame.
- This method solves the synchronization problem of the first method.

- In case if reciever ever loses synchronization he can just search for flag byte, to find the end of current frame.
- If flag byte is present as a part of data itself then ESC byte is inserted before the flag data byte. This is known as byte stuffing.
- At reciever end when it encounters an ESC byte it discards this escape byte and considers the immediate next byte as part of data.
- Same method is used in case if escape byte is present as part of data.

Examples:

Construct the frames using byte stuffing methods

1) data = A B

The frame is as follows:-

FLAG	A	B	FLAG

2) data = x a b y

The frame is as follows:-

FLAG	x	a	b	y	FLAG

3) data = A FLAG B

The frame is as follows:-

FLAG A ESC FLAG B FLAG

↳ stuffed byte.

4) data = A P FLAG FLAG Q

The frame is as follows:-

FLAG A P ESC FLAG ESC FLAG Q FLAG

↳ stuffed byte ↳ stuffed byte

5) data = A ESC B

FLAG	A	ESC	ESC	B	FAG

6) data = A ESC ESC FLAG B

FLAG A ESC ESC ESC ESC ESC FLAG B FLAG

↳ stuffed byte ↳ stuffed byte ↳ stuffed byte

Example 2:
Consider the following frames recieved by reciever. Find the original data assuming byte stuffing method.

1) Frame = FLAG A ESC ESC B FLAG
   Data : A ESC B

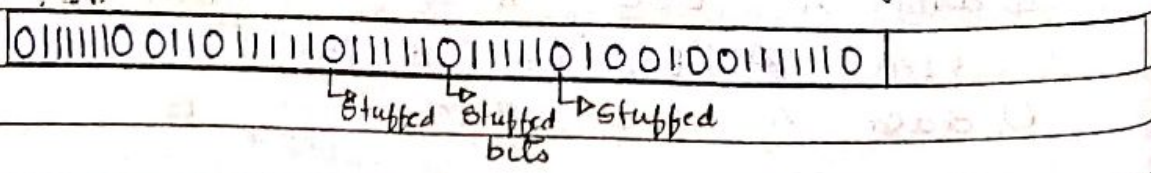
2) Frame = FLAG A ESC ESC ESC ESC B FLAG
   Data : A ESC ESC B

Note: This framing method is used in point-to-point protocol (PPP).

- Main disadvantage of this method is stuffing is done at byte level hence may consume more space.
- This problem is solved using next method in which stuffing is done at bit level.

3. FLAG bytes with bit stuffing
- Each frame begins and ends with a FLAG byte (delimiter).
- At sender side, if data contains five consecutive 1's then it automatically inserts stuffed 0 bit after these five consecutive 1's.
- At reciever end, when it gets five consecutive 1's followed by a 0, it understands that this 0 is a stuffed bit and hence it discards this bit.

Example 1:
Consider the data = 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0
Construct the frame using bit stuffing.



Example 2:

Consider the recieved frame as 01111110000111110110110000111110011111110. Find the original data.

Data: 000111111110110000111111

4 Physical layer coding Vislations

- This method is applicable to networks in which encoding on physical medium contains some redundancies.
- This redundant signals can be used to implement framing concept.

5. Combinatiso of multiple method.

- This is the combination of the previous methods.

* Types of Errors

1 Single bit error

- only one bit gets changed during transmission.
- Example: 11101011 (Data sent) 11100101 (Data recieved.)

2. Burst errors

- Two or more bits gets changed during transmission
- Example: 10110100101 (Data sent) 10010101101 (Data recieved.)

length of bit error = 8 (Last bit number - First bit number)

* Error Control

- can be classified as error detection and error correction.
- In case of Error detection reciever applies some method to find whether the data is correct or not.

If the received data is correct it will accept and give to network layer. In case if there is error in data then the reciever asks sender for Retransmission. Hence it is called as Backward

Methods are:-
1. Parity check
2. 2-D parity
3. checksum
4. Cyclic Redundancy check

- In case of error correction the reciever applies the method and check if the data is valid or not.
- If it is valid, it accepts and gives data to network layer, in case if error is detected, the reciever itself will apply error correcting algorithm to correction the Error. Hence it is known as Forward error control.
Ex: Hamming code.
- Error control is achieved by sending some redundant/extra bit along with actual data. Combination of data and redundant bits is known as codeword. This codeword is sent to reciever.
- The redundant bits allows the reciever to detect or correct the errors if present.

* Error Detection

1. Parity check.
- can be odd or even
- along with actual data one extra bit (parity bit) is sent to the reciever.

- In case of even parity check, the value of p bit is chosen in such a way that total number of 1s in codeword becomes even.
- In case of odd parity check, the value of p bit is chosen in such a way that total number of 1s in codeword becomes odd.
- Example 1:

Assuming even parity check construct the codeword.

1) data = 1001001

Codeword = 1001001 1

2) data = 10101010

codeword = 10101010 0

- Example 2:

Assuming odd parity check construct the codeword.

1) data = 11001010 1

Codeword = 11001010 10

2) data = 11001011 1

Codeword = 11001011 11

- Example 3:

Consider the following codewords recieved by reciever Detect the error if present. Incase of no error find the actual data. Assume even parity check

1) codeword = 11010101

No error.

The actual data is 11010101

2) codeword = 10110101 0

Error present.

- Note: This method can detect only single bit error. Cannot detect burst error.

2. 2-D Parity Check.

- Performance of previous method can be improved

by using this methods which organises bit in form of table. P bits are calculated for each row and for each column and then sent to each data.
- At recieving ends, this are compared with the p bits computed by the reciever
- Example 1:

Data: 10110011 10101011 01011010 11010101 (Assume even parity).

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

Codeword: 101100111 101010111 010110100 110101011 100101111
- This methods can detect burst errors of n bits.
- If two bits in one data units are damaged and two bits in exact same unit in another unit is also damaged then this method won't detect this error.

3. Checksum

At sender side

step1: Data is divided into k segments each of m bits.

step2: All this segments are added using 1's compliment to get the final sum.

step3: This sum is complimented to get the checksum.

step4: This checksum is sent along with original data.

At reciever side

step1: All the recieved segments are added using

1's complement to get sum

Step2: This sum is complemented.

If result = 0,
then "data is correct"
else
"error present."

- Example:

Consider data = 10110011 10101011 01011010 11010101

Construct the codeword using checksum method.

Solution:

1) Dividing the above data into 4 segments each of 8-bits.

10110011	10101011	01011010	11010101
S1	S2	S3	S4

2) Addition of all the segments to generate sum.

```
  10110011 (S1)
+ 10101011 (S2)
  01011110
+ 1
  01011111 (S1+S2)
+ 01011010 (S3)
  10111001 (S1+S2+S3)
+ 11010101 (S4)
  11000111 0
+ 1
  10001111 → sum
```

3) Compliment the sum : 01110000

4) Therefore the codeword is :

10110011 10101011 01011010 11010101	01110000

↳ checksum.

Example 2:
Consider the same above codeword and find whether the given codeword is correct or not.

Solution:

1) Addition of all the segments using 1's compliment.

```
  1 0 1 1 0 0 1 1 (S1)
+ 1 0 1 0 1 0 1 1 (S2)
  ---------------
  0 1 0 1 1 1 1 0 (S1+S2)
+               1
  ---------------
  0 1 0 1 1 1 1 1 (S1+S2)
+ 0 1 0 1 1 0 1 0 (S3)
  ---------------
  1 0 1 1 1 0 0 1 (S1+S2+S3)
+ 1 1 0 1 0 1 0 1 (S4)
  ---------------
  1 0 0 0 1 1 1 0
+               1
  ---------------
  1 0 0 0 1 1 1 1 (S1+S2+S3+S4)
+ 0 1 1 1 0 0 0 0 (S5)
  ---------------
  1 1 1 1 1 1 1 1 (Sum)
```

2) Complement the sum

3) 00000000.

result = 00000000

since (result == 0)

Therefore Data is correct.

- checksum detects all the error involving odd and even number of bits

* Error Correction

- Hamming distance between two codeword is the number of bit positions by which the two codeword differ

Example 1:
Find the hamming distance between 10001001 and 10110001.

Soln:
To get hamming distance we use XOR operation and count number of 1's in the answer. This count is the hamming distance.

```
  1 0 0 0 1 0 0 1
⊕ 1 0 1 1 0 0 0 1
-----------------
  0 0 1 1 1 0 0 0
```

∴ Hamming distance = 3

Example 2:
Find the hamming

Soln:

```
    1 1 1 0 0 0 1 1 0 0 1
⊕   0 1 0 1 0 0 0 1 1 1 0
-------------------------
    1 0 1 1 0 0 1 0 1 1 1
```

∴ Hamming distance = 7

Hamming code for Error correction
- used to detect as well as correct the errors.
- The number of redundant bits needed can be calculated using following calculation

$$m + r + 1 \leq 2^r$$

where $r \rightarrow$ number of redundant bits.
$m \rightarrow$ number of message data bits.

- within the codeword, redundant bits or p bits are always inserted at positions which are power of 2.

Example 1:

Assuming even parity Construct codeword for 1001101 using hamming method.

Solution:

data = 1001101

∴ m=7

step1: To find: Number of p bits

$$m+r+1 \le 2^r$$

$r=1 \Rightarrow 7+1+1 \le 2^1 \Rightarrow 9 \le 2$ X

$r=2 \Rightarrow 7+2+1 \le 2^2 \Rightarrow 10 \le 4$ X

$r=3 \Rightarrow 7+3+1 \le 2^3 \Rightarrow 11 \le 8$ X

$r=4 \Rightarrow 7+4+1 \le 2^4 \Rightarrow 12 \le 16$ ✓

∴ r=4

∴ 4-p bits are needed.

step2: To set value of 4-p-bits

Codeword layout is

P1	P2	1	P4	0	0	1	P8	1	0	1
P1	P2	3	P4	5	6	7	P8	9	10	11

To set values of P-bits

P1 ⇒ 1,3,5,7,9,11 ⇒ ?1011 ⇒ 0

P2 ⇒ 2,3,6,7,10,11 ⇒ ?1101 ⇒ 1

P4 ⇒ 4,5,6,7 ⇒ ?001 ⇒ 1

P8 ⇒ 8,9,10,11 ⇒ ?101 ⇒ 0

Therefore the final codeword is:

0	1	1	1	0	0	1	0	1	0	1
P1	P2	3	P4	5	6	7	P8	9	10	11

Example2:

Consider message = 11010011001101 01. Construct the codeword using odd parity check.

solution:

data = 1101001100110101

∴ m=16 bits.

Step 1: To find no. number of p bits

$m + r + 1 \leq 2^r$

$r = 1 \Rightarrow 16 + 1 + 1 \leq 2^1 \Rightarrow 18 \leq 2$ X

$r = 2 \Rightarrow 16 + 2 + 1 \leq 2^2 \Rightarrow 19 \leq 4$ X

$r = 3 \Rightarrow 16 + 3 + 1 \leq 2^3 \Rightarrow 20 \leq 8$ X

$r = 4 \Rightarrow 16 + 4 + 1 \leq 2^4 \Rightarrow 21 \leq 16$ X

$r = 5 \Rightarrow 16 + 5 + 1 \leq 2^5 \Rightarrow 22 \leq 32$ ✓

$\therefore r = 5$

$\therefore$ 5-p-bits are needed.

Step 2: To set value of 5-p-bits

Codeword layout is

P1	P2	1	P4	1	0	1	P8	0	0	1	1	0	0	1	P16	1	0	1	0	1
P1	P2	3	P4	5	6	7	P8	9	10	11	12	13	14	15	P16	17	18	19	20	21

To set value of p-bits

P1 ⇒ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21 ⇒ ?1110.10.1111 ⇒ 1

P2 ⇒ 2, 3, 6, 7, 10, 11, 14, 15, 18, 19 ⇒ ?1010.10.101 ⇒ 0

P4 ⇒ 4, 5, 6, 7, 12, 13, 14, 15, 20, 21 ⇒ ?1011.00.101 ⇒ 0

P8 ⇒ 8, 9, 10, 11, 12, 13, 14, 15 ⇒ ?0011.001 ⇒ 0

P16 ⇒ 16, 17, 18, 19, 20, 21 ⇒ ?10101 ⇒ 0.

Therefore final codeword is

1	0	0	1	0	1	0	0	0	1	1	0	0	1	0	1	0	1	0	1
P1	P2	3	P4	5	6	7	P8	9	10	11	12	13	14	15	P16	17	18	19	20	21

Redundant bits

Example 3:

Consider the received codeword as 0111.0010101 using hamming method check if it contains an error. If yes, correct the error and get original data. (Even parity check).

Solution:

It is at receiver side,

0	1	1	1	0	0	1	0	1	0	1
P1	P2	3	P4	5	6	7	P8	8	9	10

P1 ⇒ 1, 3, 5, 7, 9, 11 ⇒ 0 1 0 1 1 1 ⇒ 0 ✓

P2 ⇒ 2, 3, 6, 7, 10, 11 ⇒ 1 1 0 1 0 1 ⇒ 0 ✓

P4 ⇒ 4, 5, 6, 7 ⇒ 1 0 0 1 ⇒ 0 ✓

P8 ⇒ 8, 9, 10, 11 ⇒ 0 1 0 1 ⇒ 0 ✓

Hence the codeword doesn't contain any error.

∴ The actual data is 1001101.

Example 4:

Consider the codeword as 01110010001. Using Hamming method check if it contains an error, if present correct it. (Assume even parity)

Solution:

At reciever side

0	1	1	1	0	0	1	0	0	0	1
P1	P2	3	P4	5	6	7	P8	9	10	11

P1 ⇒ 1, 3, 5, 7, 9, 11 ⇒ 0 1 0 1 0 1 ⇒ ✗ 1

P2 ⇒ 2, 3, 6, 7, 10, 11 ⇒ 1 1 0 1 0 1 ⇒ ✓ 0

P4 ⇒ 4, 5, 6, 7 ⇒ 1 0 0 1 ⇒ ✓ 0

P8 ⇒ 8, 9, 10, 11 ⇒ 0 0 0 1 ⇒ ✗ 1

Hence the position of erroneous bit is 1+8=9.

Hence the corrected codeword is

0	1	1	1	0	0	1	0	1	0	1
P1	P2	3	P4	5	6	7	P8	9	10	11

∴ The actual data is 1001101.

Q. Using cyclic Redundancy check method construct the codeword.

$data = x^9 + x^8 + x^6 + x^4 + x^3 + x^2 + x + 1$

$g(x) = x^4 + x + 1$.

Soln: $data = 1x^9 + 1x^8 + 0x^7 + 1x^6 + 0x^5 + 1x^4 + 1x^3 + 1x^2 + 1x + 1$

$= 1\,1\,0\,1\,0\,1\,1\,1\,1\,1$

$g(x) = x^4 + x + 1$

$= x^4 + 0 \cdot x^3 + 0 \cdot x^2 + x + 1$

$= 10011$

k = number of bits in $g(x) = 5$.

Number of redundant bits to be added;

$r = k - 1$

$= 5 - 1$

$= 4$

∴ 4-bits must be appended to data.

updated data is

11010111110000.

Dividing data by g(x).

```
                1100001110
        ---------------------
10011 ) 11010111110000
        10011
        -----
        01001
        00000
        -----
        10011
        10011
        -----
        00001
        00000
        -----
        00011
        00000
        -----
        00111
        00000
        -----
        01111
        00000
        -----
        11110
        10011
        -----
        11010
        10011
        -----
        10010
        10011
        -----
        00010
        00000
        -----
        00010
```

∴ The codeword after updating is

11010111110010 → Redundant bits

To verify the steps:

* Flow Control

Flow control →
- Feedback based flow control
- Rate based flow control
  - Ex: 1 minute - 5 packets
  - (built in mechanism)

Elementary Data Link Protocols are:
1. Unrestricted Simplex Protocol (Utopian)
2. Simplex stop and wait protocol for Perfect channel.
3. Simplex stop and wait protocol for noisy channel. (PAR, ARQ)

1. Unrestricted Simplex Protocol (Utopian Protocol)

Assumption:
1) Communication channel is considered as perfect hence error control is not required.
2) Infinite buffer space at sender and reciever.
3) Processing speed at sender and reciever is very high, hence flow control not needed.
4) Data transmitted in one direction only.

Hence unrealistic

Sender procedure:

```
void sender1(void)
{
    packet buffer;
    frame S;
    while(true)
    {
        from_network_layer(&buffer);
        S.info = buffer.
        to_physical_layer(&S);
    }
}
```

→ D.L.L at sender side accepts the packet from network layer then it constructs the frame as by inserting the packet into the information part of frame. This constructed frame is given to physical

layer for transmission.

Reciever Procedure:

```
void reciever1(void)
{
    event_type event;
    frame r;
    while(true)
    {
        wait_for_event(&event);
        from_physical_layer(&r);
        to_network_layer(&r.info);
    }
}
```

→ Data link layer at the recieving end initially waits for some event to happen, the only possible event is arrival of frame at physical layer. This frame is fetched from the physical layer and stored in variable r. Only the information part of this frame (i.e packed) is sent/given to the network layer.

2. Simplex stop and wait protocol for perfect channel.

Assumptions:

1) Limited buffer space is available at sender and reciever. Hence limited processing speed and therfore flow control is required.
2) Channel is considered as perfect, hence error control is not required.
3) Flow control is implemented using acknowledge-ments which reciever sends to the user. Hence before sending the next data, sender waits

for acknowledgements of the reciever.
4) Reciever after handing over the packet to the network layer, does one extra step. This extra step is sending the acknowledgement to the sender.

Sender procedure:

```
void sender2(void)
{
   packet buffer;
   frame s;
   event_type event;
   while(true)
   {
      from_network_layer(&buffer);
      s.info = buffer;
      to_physical_layer(&s);
      wait_for_event(&event);
   }
}
```

→

Reciever procedure:

```
void Reciever2(void)
{
   frame r,s;
   event_type event;
   while(true)
   {
      wait_for_event(&event);
```

```
        from_physical_layer (&r);
        to_network_layer (&r.info);
        to_physical Layer (&s); /* sending acknowl-
            -edge back to sender.
    }
}
```

→

3 Simplex stop and wait protocol for noisy channel. (ARQ/PAR)

Assumptions:

1) A channel is assumed to be noisy, Hence the error control needs to be implemented. This will be done using sequence numbers and timer facilities.
2) Limited buffer space and processing speeds at both the ends. Hence flow control is also required which will be implemented using acknowledgement.

Sender procedure:

```
void sender3 (void)
{
    packet buffer;
    frame s;
    event_type event;
    seq_nr nfts; // next frame to send
    nfts=0;
    from_network_layer (&buffer);
    while (true)
```

```
{
    s.info = buffer;
    s.seq = nfts;
    to_physical_layer(&s);
    start_timer(s.seq);
    wait_for_event(&event);
    if(event == frame_arrival)
    {
        from_physical_layer(&s);
        if(s.ack = nfts)
        {
            stop_timer(s.ack);
            from_network_layer(&buffer);
            inc(nfts);
            /* End of inner if */
        /* End of outer if */
    /* End of while */
/* End of function */
```

Cut of sender

→ After transmitting the frame and starting the timer the sender waits for event to happen. Only three possibilities exists.

1. undamaged acknowledgement frame is recieved.
2. Damaged ack recieved.
3. Expiry of timer.

→ If a valid ack frame comes in the sender stops the timer for the sent frame, fetches next packet from network layer and advances the sequence number.

→ If damaged ack is achieved or timer expires then neither the buffer nor sequence number is changed so that duplicate can be sent.

Reciever Procedure:

```
void reciever3(void)
{
  seq_nr fe;
  frame r,s;
  event_type event;
  fe=0; // frame expected
  while(true)
  {
     wait_for_event(&event);
     if(event == frame_arrival)
     {
        from_physical_layer(&r);
        if (r.seq_no == fe)
        {
           to_network_layer(&r.info);
           inc(fe);
        }
        s.ack=fe-1;
        to_physical_layer(&s);
     }
  }
}
```

Wrong if sender packets are not accepted

→ At the reciever end when a valid frame is delivered its sequence number is checked to see if it is a duplicate, if not it is accepted past to the network layer and then acknowledgment is constructed and sent to the sender.

→ Duplicate and damaged frames are not passed to network layer.

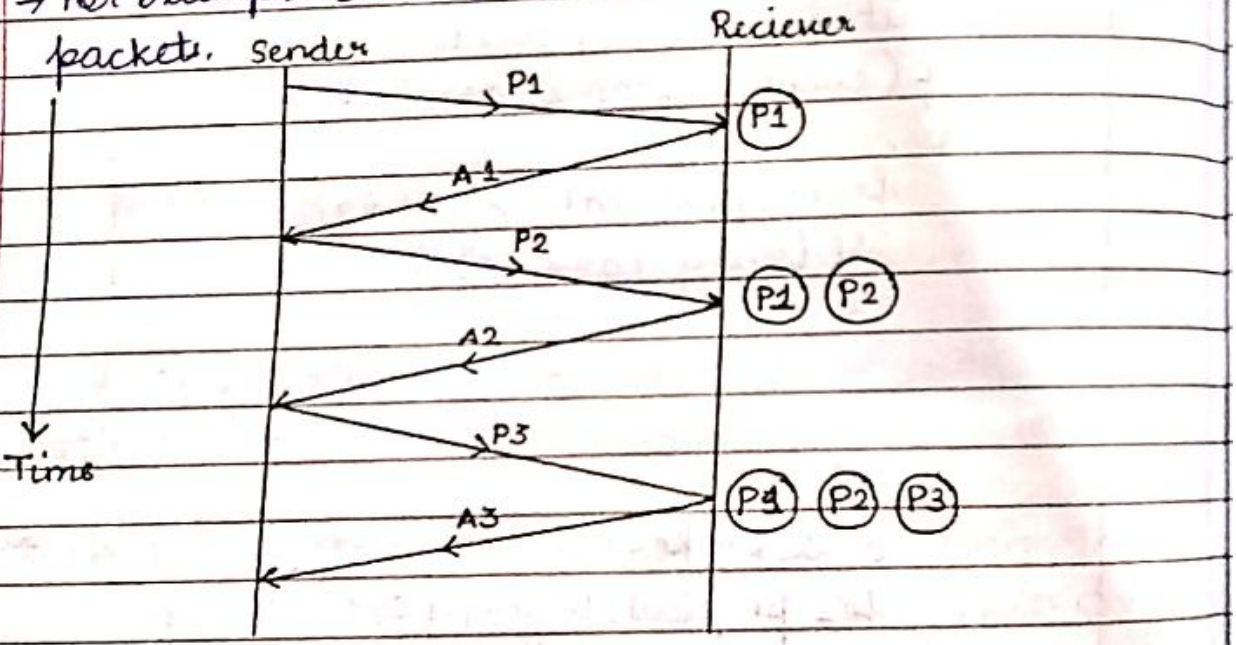
* Various cases of stop and wait ARQ Protocol

Case 1: Normal case

→ Sender sends the first packet, starts the timer, stops and waits for its acknowledgement.

→ upon getting its ack, sender turns of the timer for the previous packet and start with the next packet.

→ For example consider sender has to send three packets.



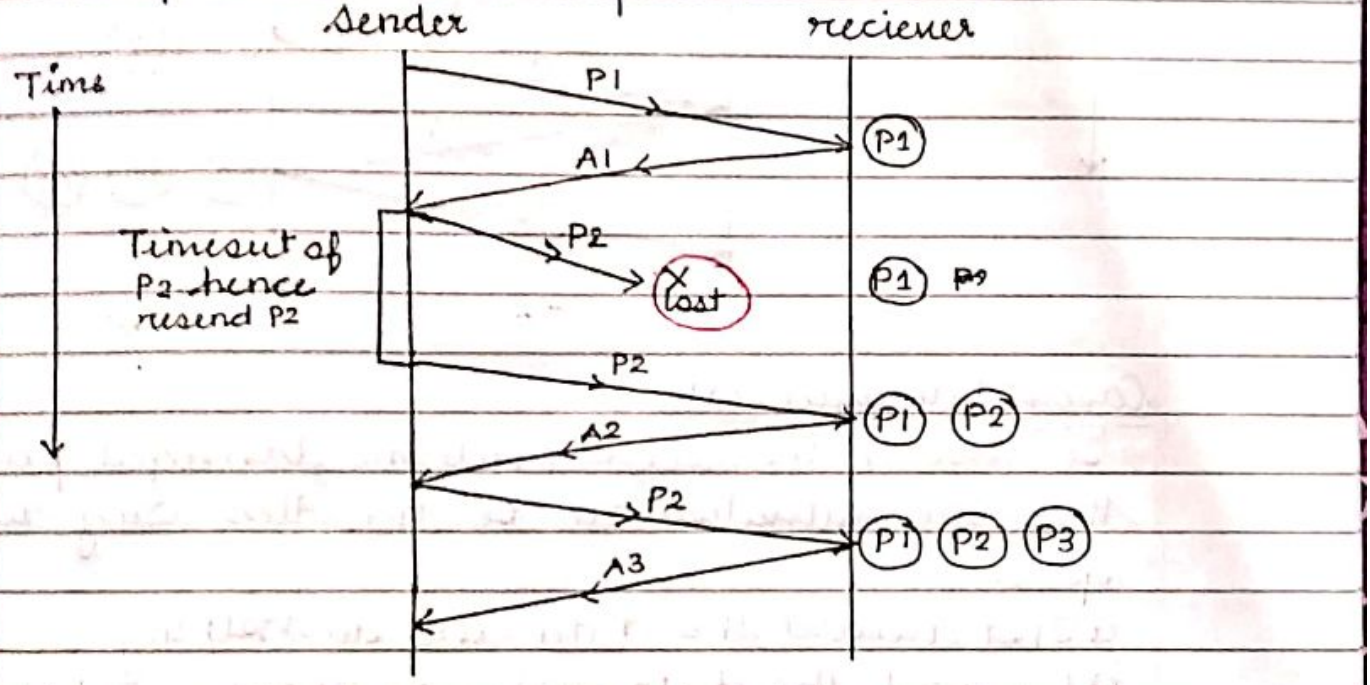
Case 2: packet lost

→ Because of some transmission issues incase if a particular packet is lost then sender continuously waits for its acknowledgement and it goes in a deadlock state.

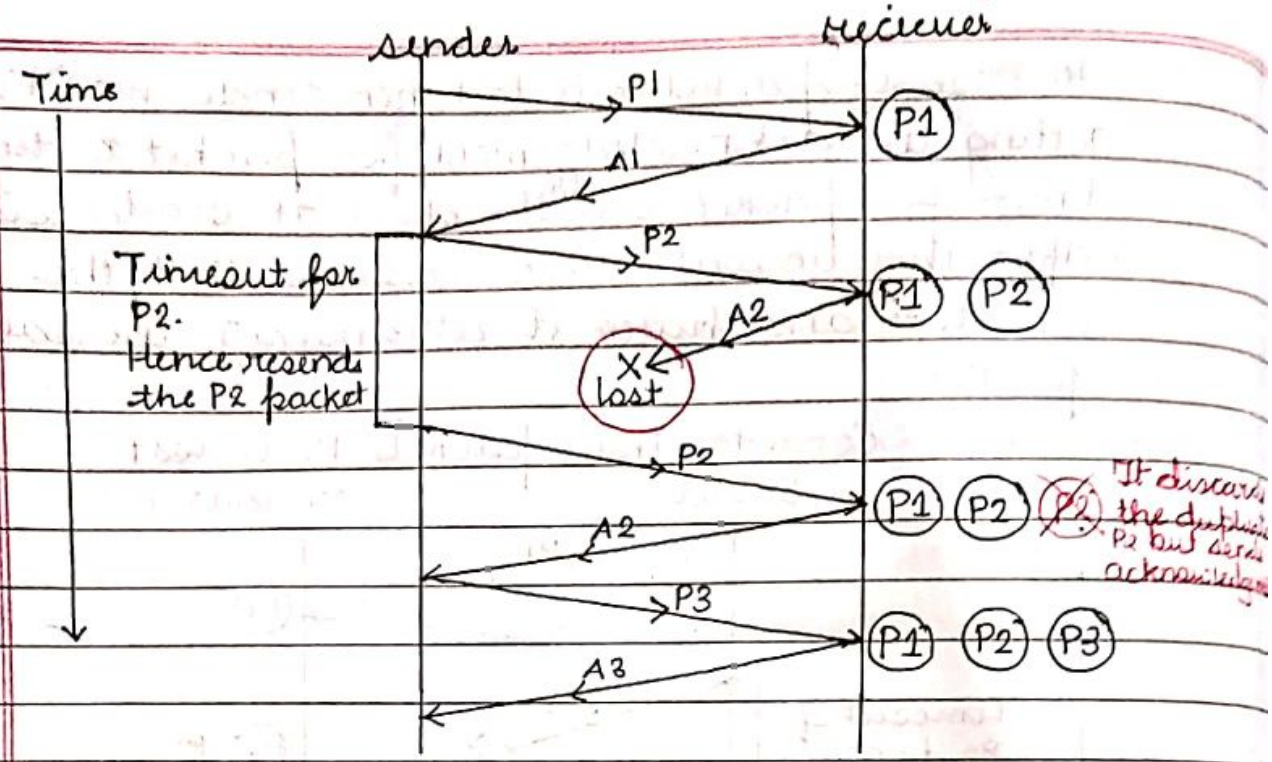
→ This problem can be solved by using a timeout timer at the sender side.

→ Sender sends a packet x to reciever and starts the timer; if ack for packet x is recieved before the expiry of timer then sender turns of the timer for x and proceed with next packet.

→ In case if packet x is lost then sender won't be getting an acknowledgement for packet x. Hence the timer for packet x will expire at sender side.
→ After this timeout, sender assumes that this packet was lost and hence it retransmits the same packet
→ Example: Consider three packets. P2 is lost.



Case 3: Acknowledgement Lost
→ If an acknowledgement for packet x is lost then there will be a timeout for the packet at sender. And because of this timer sender will assume that packet x was lost during transmission (false conclusion).
→ Now sender retransmits the packet x and reciever gets duplicate copy of this packet. The reciever discards the duplicate the copy but sends the acknowledgement to the sender.
→ Example: Consider three packets P1, P2 and P3.
The acknowledgement A2 is lost.



Case 4: Damaged Data

→ In case if the reciever gets the damaged packet then this situation can be handled using two options:

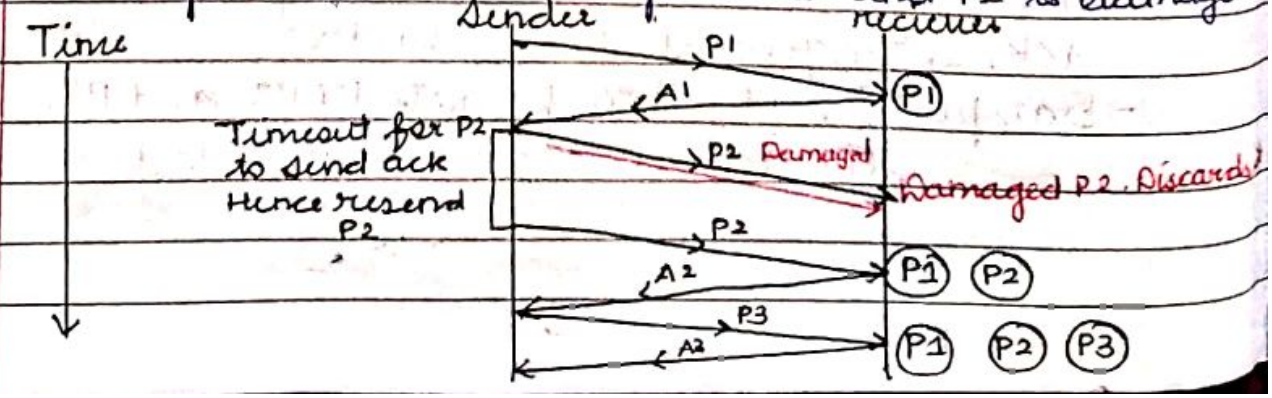
a) Just discard the data and do nothing.

b) Discard the data and send negative acknowledg

4.a: Discard the data and do nothing:

→ Using this method, the reciever discards the recieved damaged data without sending the acknowledgement to the sender. Hence timeout occurs at the sender side after which the sender resends the data.

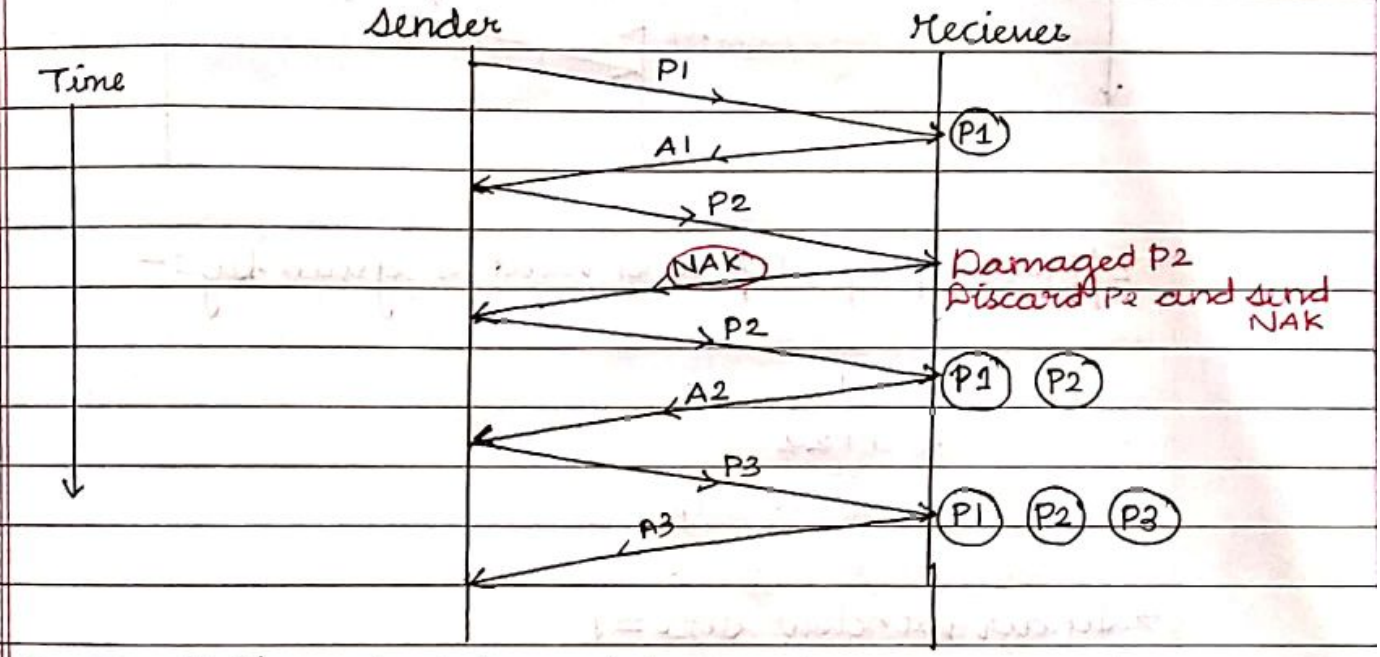
→ Example: Consider three packets and P2 is damaged



4.b: Discard the data and send negative acknowledgement.

→

→ Example:

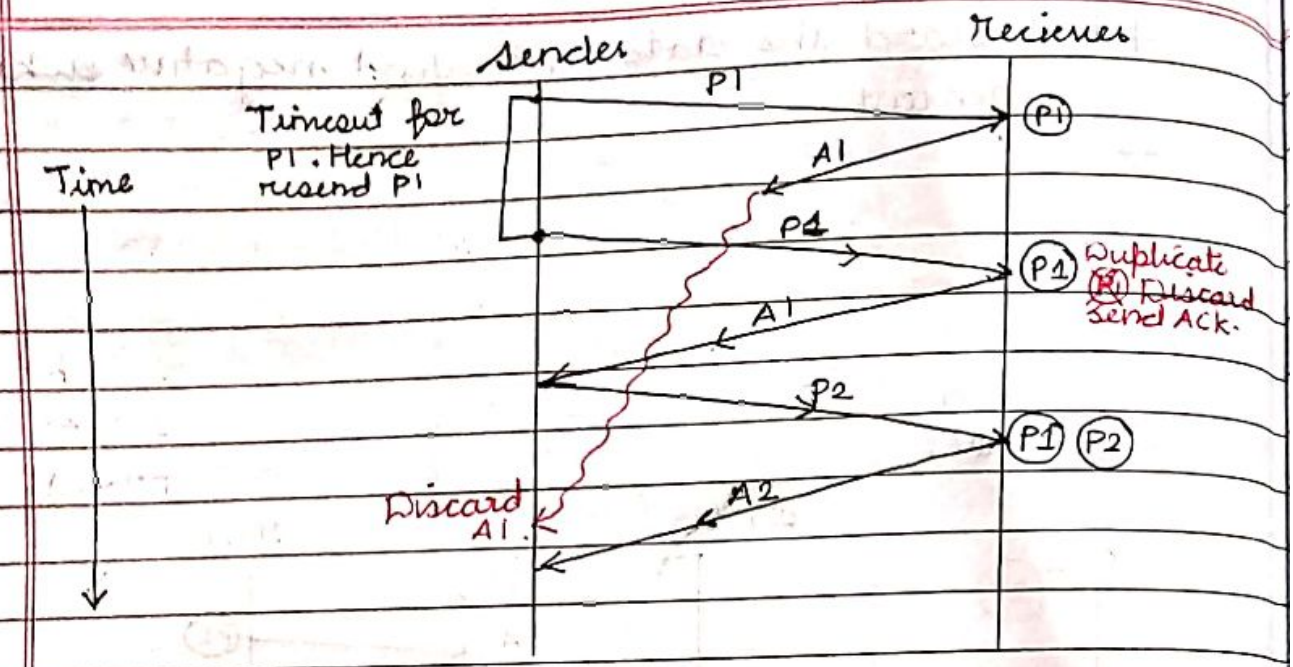


Case 5: Delayed Acknowledgement.

→ If sender gets delayed acknowledgement it identifies it using sequence number.

→ If reciever gets duplicate packet, it discards this duplicate and resends this acknowledgement.

→ Example:



→ Efficiency of stop and wait is given by :-

$$\eta = \frac{1}{1+2a}$$

where $a = \frac{T_p}{T_t}$

→ Sender window size = 1

→ Reciever window size = 1

→ Effective bandwidth / throughput = η * bandwidth.

* Types of acknowledgements

1. Cumulative Acknowledgement

→ One common ack is used for multiple packets.

→ For example for every 5th recieved packet the reciever can send the acknowledgement.

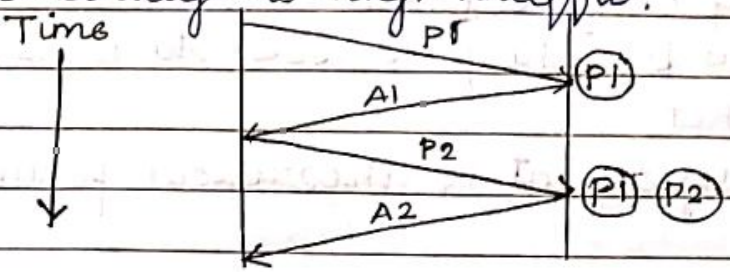
→ In this manner ACK 5 indicates that packets 1, 2, 3, 4, 5 are successfully delivered.

→ Advantage of this ack is less traffic.

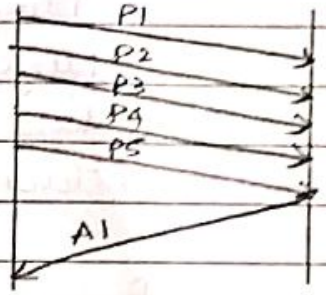
→ Disadvantage is less reliability, if one ack is lost

it means that all previous packets are lost.

2. Independent Acknowledgement
→ Each packet is acknowledged independently,
→ Advantage is high reliability
→ Disadvantage is high traffic.
→ Ex;



For cumulative



* Types of Delays

1. Transmission Delay (T_t)
→ Time required to put the entire packet on to the communication link.
→ Larger the packet, more is the transmission-time

$$T_t = \frac{L}{B}$$

where L → size/length of packet
B → Bandwidth.

2. Propagation Delay (T_p)
→ Time taken by 1 bit to travel from sender end to the reciever end.

$$T_p = \frac{D}{V}$$

where D → Distance
V → velocity

Note: Round Trip Time = 2 * T_p → Back and forth

3. Queing Delay (T_q)
→ Time taken by data packets waiting in a queue/buffer before it is taken for processing.
→ Machine dependent no theoretical formula.

4. Processing Delay (T_{proc})
→ Time taken by the processor to process the data packet.
→ Machine dependent, no theoretical formula

Problems:-

1. Consider the noiseless channel with bandwidth 3000Hz. with two signal level. Calculate maximum capacity of channel.

Soln: since its a noisless channel we use Nquist theoram.
Given
$B = 3000Hz$
2 signal levels.
$C = 2B \log_2 L$
$= 2 \times 3000 \times \log_2 2$
$= 6000 bps.$

2. Consider the noiseless channel with bandwidth 3000Hz. with four signal levels. calculate maximum

Soln: Given
noiseless channel
4 signal levels
$B = 3000Hz$
$C = 2B \log_2 L$
$= 2 \times 3000 \times \log_2 4$
$= 12000 bps.$

Increasing the number of signal level, increases signal capacity but also leads to noise (signal distortion)

3 We need to send 265kbps data over noiseless channel of bandwidth 20KHz. How many signal levels are needed.

Soln: Given

$C = 265 kbps$

$B = 20 kHz$

Since the channel is noiseless, using nquist theorm we get

$$C = 2 * B * \log_2 V$$

$$265 \times 10^3 = 2 \times 20 \times 10^3 \log_2 V$$

$$\log_2 V = \frac{265}{40}$$

$$\log_2 V = 6.625$$

$$V = (2)^{6.625}$$

$$V = 98.70$$

Note: Number of signal levels should be power of 2. Hence in this case V can be considered as either 64 or 128. If we consider V=64, channel capacity will decrease. If we consider it as 128, capacity will go beyond 265 kbps.

4. Given Signal to Noise ratio is 36dB and bandwidth as 2MHz. Calculate channel capacity.

Soln: Given

$B = 2MHz = 2 \times 10^6 Hz$

$SNR_{db} = 36 db$

$36 = 10 * \log_{10} SNR$

$3.6 = \log_{10} SNR$

$SNR = 3981$

By using Shannon's theorem

$C = B * \log_2(1+SNR)$

$= 2\times10^6 \times \log_2(1+3981)$

$= 2\times10^6 \times 11.95927775$

$= 23.9185\times10^6$ bps.

$= 23.92$ mbps

5 Consider a noisy channel with SNR=0. Calculate C

Soln: Since its a noisy channel use shannon's theorem

$C = B * \log_2(1+SNR)$

$= B * \log_2(1)$

$= B * 0$

$= 0.$

6. A telephone line has a ~~frequency~~ range from 6000Hz to 3000 Hz. SNR=3162. Calculate C.

Soln: Given

$f_1 = 6000$Hz

$f_2 = 3000$Hz

SNR=3162

∴ Bandwidth, $B = f_1 - f_2$

$= 6000 - 3000$ Hz

$= 3000$ Hz.

$C = B * \log_2(1+SNR)$

$= 3000 \times \log_2(3163)$

$= 3000 \times 11.6271$

$= 34881.233$ bps.

7. Consider bandwidth of 1MHz and SNR=63. Calculate the maximum capacity of the channel and also number of signal levels.

Soln: Note: First apply shannon's theorem and get the

capacity..

Given

$B = 1\,MHz = 1 \times 10^6\,Hz$

$SNR = 63$

Using shannon

$C = B * \log_2(1+SNR)$

$= 1 \times 10^6 \times \log_2(64)$

$= 10^6 \times 6$

$= 6\,Mbps$

Now substitute this capacity in nquist theorem to get number of signal levels.

$C = 2 * B * \log_2 V$

$6 \times 10^6 = 2 * 1 \times 10^6 \log_2 V$

$3 = \log_2 V$

$V = (2)^3$

$V = 8.$

8. If a binary signal is sent over a channel of 3kHz bandwidth whose SNR is 20db. What is the maximum achievable data rate.

Soln: Given

$B = 3\,kHz = 3 \times 10^3\,Hz$

$SNR_{db} = 20\,db$

$SNR_{db} = 10 * \log_{10} SNR$

$20 = 10 * \log_{10} SNR$

$\log_{10} SNR = 2$

$SNR = (10)^2$

$SNR = 100$

∴ Using shannon's theorem

$C = B * \log_2(1+SNR)$

$= 3 \times 10^3 \times \log_2(101)$

$= 3 \times 10^3 \times 6.6582$

$= 19.9746$ kbps

Fourier analysis

Any periodic function $g(t)$ with period T can be constructed as sum of possible number of sines and cosines.

$$g(t) = \frac{1}{2} c + \sum_{n=1}^{\infty} a_n \cdot \sin(2\pi n f t) + \sum_{n=1}^{\infty} b_n \cdot \cos(2\pi n f t) \quad \text{--- (1)}$$

where

$f = \frac{1}{T}$ is fundamental frequency

a_n = sine amplitude of nth harmonic term

b_n = cosine amplitude of

c = constant

If the period T is known and if the amplitudes are given then original function of time can be reconstructed by perform sums of eqn (1).

To get amplitude a_n multiply both sides of eqn (1) by $\sin(2\pi k f t)$ and then perform the integration from 0 to T.

$$\int_0^T \sin(2\pi k f t) \cdot \sin(2\pi n f t)\, dt = \begin{cases} 0 & \text{for } k \neq n \\ T/2 & \text{for } k = n. \end{cases}$$

Hence

$$\boxed{a_n = \frac{2}{T} \int_0^T g(t) \cdot \sin(2\pi n f t)\, dt}$$

Similarly to get value of b_n multiply both sides of eqn (1) by $\cos(2\pi k f t)$ and perform integration from 0 to T.

Hence $b_n = \frac{2}{T}\int_0^T g(t)\cdot\cos(2\pi n f t)\,dt$

$c = \frac{2}{T}\int_0^T g(t)\,dt$

Baseband, Broadband and Passband

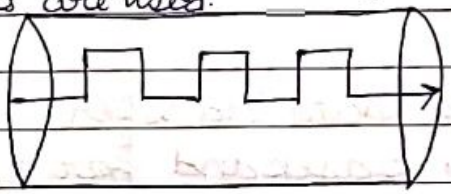
Baseband:
- In baseband data is sent as digital signal in which single channel uses entire bandwidth.
- Baseband communication is bidirectional which means that same channel can be used to send and recieve signals.
- FDM is not possible.
- used for short distance data transfer
- Ethernet standards uses baseband for LAN data transfer

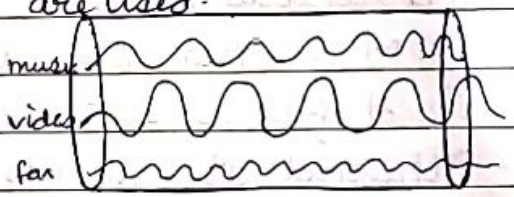
Broadband:
- It sends information in form of analog signal.
- Using FDM each transmission can be assigned portion of the bandwidth hence multiple transmission are possible at same time.
- Broadband communication is unidirectional so inorder to send and recieve two pathways are needed i.e by using two different frequency or cables.
- Used for long distance data transfer
- Home internet connection and television cables uses broadband.
- Data is sent after modulation hence modems

are required.

Difference

Baseband	Broadband
1) Sends digital signal	1) sends analog signals
2) Modulation not needed before transmission.	2) Modulation needed before transmission.
3) Bidirectional transfer	3) unidirectional transfer
4) Use for short distance transmission.	4) Used for long distance transmission.
5) Ethernet LAN standards	5) Internet and cable TV
6) FDM is not possible but we can use TDM	6) FDM is possible.
7) To boost signal strength repeaters are used.	7) To boost signal strength amplifiers are used.





Passband:

- It is the range of frequencies/wavelengths that can pass through a filter.
- Ex: Bandpass filter of a radio reciever to select the frequency of desired radio signal out of all the radio waves picked by its antennas.

Transmission Medium

Transmission medium

Guided			Unguided		
Twisted pair Cable	Coaxial cable	optical fibre	Radio waves	Micro waves	Infrared waves

Maximum Data rates of the channel (Maximum capacity of channel)

1 Nyquist Theorem for Perfect Channel / Noiseless channel.

$$C = 2B \log_2 V$$

C → maximum capacity
B → Bandwidth
V → No. of signals levels (power of 2).

2 Shannon's Theorem for Noisy channel.

$$C = B \log_2(1+SNR)$$

C → maximum capacity
B → Bandwidth
SNR → Signal to noise ratio

$$SNR_{db} = 10 * \log_{10} SNR$$

In case if SNR is given in decibels then first we have to convert it to normal SNR, then substitute and use in normal formula.