Unit-4

Data Link Layer :

Framing Techniques:

Framing in the data link **layer separates a message from one source to a destination by adding a sender address and a destination address**.

The **destination address defines where the packet is to go**; the **sender address helps the recipient acknowledge the receipt**.

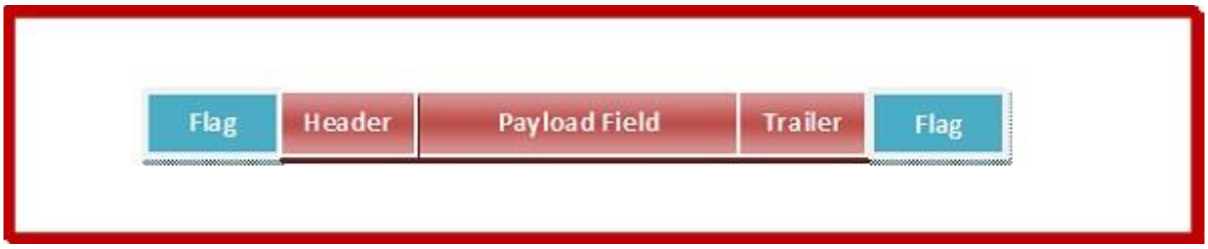
When a message is carried in one very large frame, even a single-bit error would require the retransmission of the whole message.

When a message is divided into smaller frames, a single-bit error affects only that small frame.

Parts of a Frame

A frame has the following parts −

- Frame Header − It contains the source and the destination addresses of the frame.
- Payload field − It contains the message to be delivered.
- Trailer − It contains the error detection and error correction bits.
- Flag − It marks the beginning and end of the frame.



Types of Framing

Framing can be of two types, fixed sized framing and variable sized framing.

Fixed-sized Framing

The **size of the frame is fixed** and **so the frame length acts as delimiter(boundary) of the frame**. As a result, **it does not require additional boundary bits to identify the start and end of the frame**.

Example − ATM cells.

Variable – Sized Framing

The **size of each frame to be transmitted may be different**. So **additional mechanisms are kept to mark the end of one frame and the beginning of the next frame**.

Example - It **is used in local area networks**.

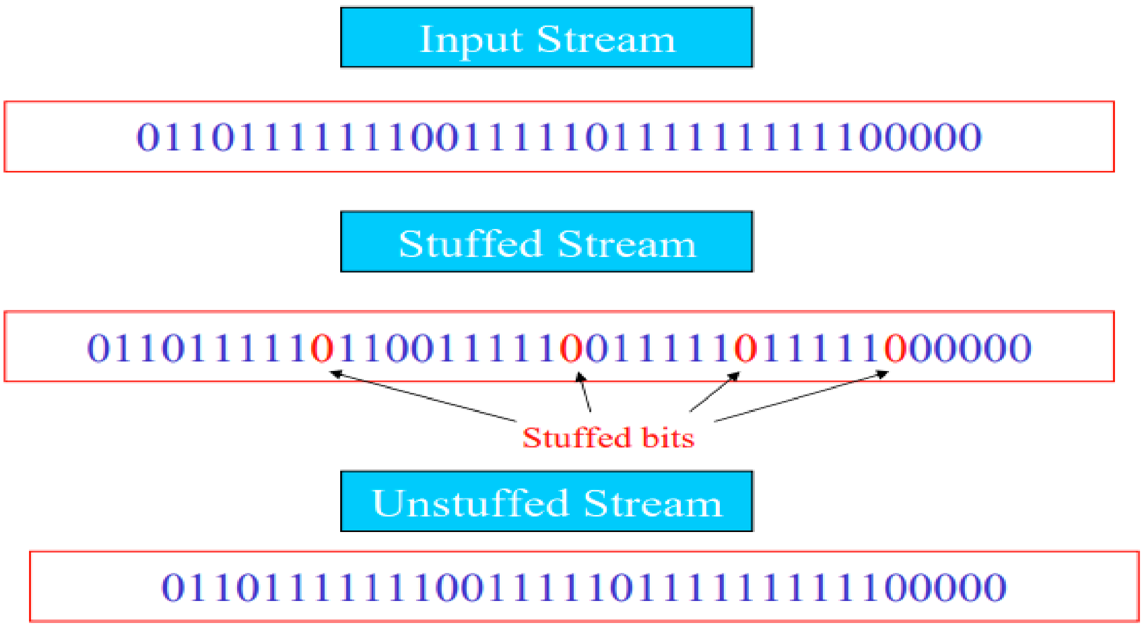
Framing Approaches in Computer Network

Talking about Framing Approaches in computer networking, there are mainly 2-different kind of approaches to Framing in the Data link layer:

1. Bit-Oriented Framing

Most protocols use a special 8-bit pattern flag 01111110 as a result of the delimiter to stipulate the beginning and so the end of the frame. Bit stuffing is completed at the sender end and bit removal at the receiver end.

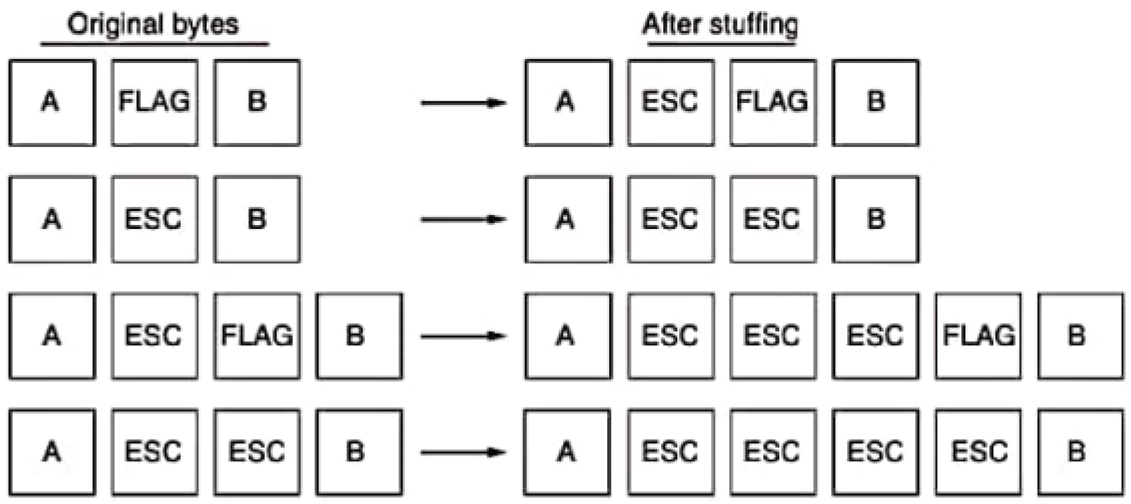
If we have **a tendency to get a zero(0) after 5 1s**. we have a tendency to tend to still stuff a zero(0). The receiver will remove the zero.



2. Byte-Oriented Framing

Byte stuffing is one of the **methods of adding an additional byte once there is a flag or escape character within the text**. Take an illustration of byte stuffing as appeared in the given diagram.

The sender **sends the frame by adding additional ESC bits** and therefore the destination machine receives the frame and it removes the extra bits to convert the frame into an identical message.



Methods of Framing

Mainly there are four different methods of framing as listed below –

1. Character Count

This methodology is needed to count the total range of characters that are available in the frame.

This process is often done by manipulating fields in the header.

This Character count methodology makes sure at the receiver end regarding the total range of characters maintain, and where the frame ends.

2. Flag Byte with Character Stuffing

The Flag Byte with Character stuffing is additionally called byte stuffing or character-oriented framing.

In byte stuffing, a particular byte that is primarily called ESC (Escape Character) which has a fixed design is usually attached to the data section of the frame.

Though the receiver detaches the ESC along with the data.

3. Starting and Ending Flags, with Bit Stuffing

Starting and Ending Flags, with Bit Stuffing, is additionally called a bit-oriented framing or bit-oriented approach.

This is typically the lodging of additional bits within the transmission unit.

Simply this is a kind of protocol control that is merely performed to interrupt the bit sample which ends up in communication.

4. Encoding Violations

Encoding violation is a technique that is used just for framing which brings some kind of repetition.

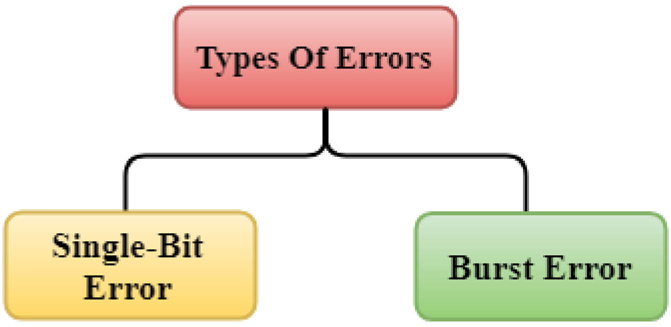
Which is used in more than one graphical or visual structure to easily encode or represent one variable of information.

Error Control

When data is transmitted from one device to another device, the system does not guarantee whether the data received by the device is identical to the data transmitted by another device.

An Error is a situation when the message received at the receiver end is not identical to the message transmitted.

Types Of Errors

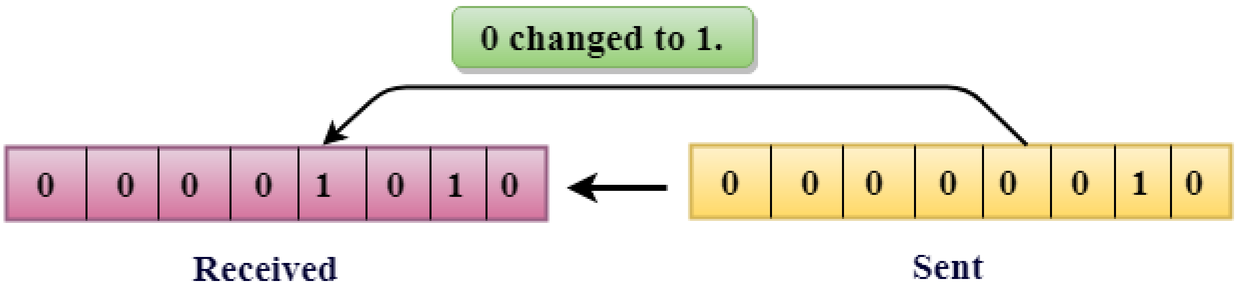


Errors can be classified into two categories:

- Single-Bit Error
- Burst Error

Single-Bit Error:

The only one bit of a given data unit is changed from 1 to 0 or from 0 to 1.

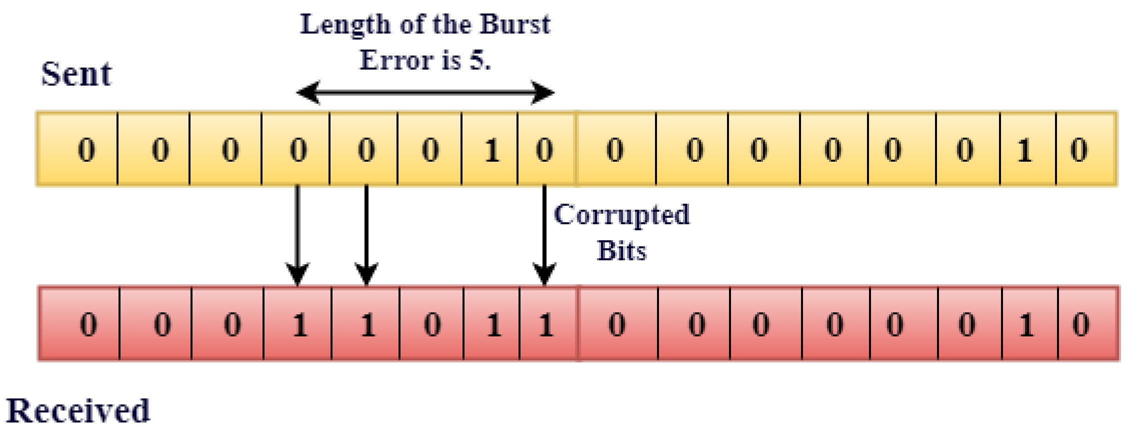


In the above figure, the message which is sent is corrupted as single-bit, i.e., 0 bit is changed to 1.

Burst Error:

The two or more bits are changed from 0 to 1 or from 1 to 0 is known as Burst Error.

The Burst Error is determined from the first corrupted bit to the last corrupted bit.



The duration of noise in Burst Error is more than the duration of noise in Single-Bit.

Burst Errors are most likely to occur in Serial Data Transmission.

The number of affected bits depends on the duration of the noise and data rate.

Error Detecting Techniques:

The most popular Error Detecting Techniques are:

- Single parity check
- Checksum
- Cyclic redundancy check

Single Parity Check-

In this technique,

- One extra bit called as **parity bit** is sent along with the original data bits.
- Parity bit helps to check if any error occurred in the data during the transmission.

Steps Involved-

Error detection using single parity check involves the following steps-

Step-01:

At sender side,

- Total number of 1’s in the data unit to be transmitted is counted.
- The total number of 1’s in the data unit is made even in case of even parity.
- The total number of 1’s in the data unit is made odd in case of odd parity.
- This is done by adding an extra bit called as **parity bit**.

Step-02:

- The newly formed code word (Original data + parity bit) is transmitted to the receiver.

Step-03:

At receiver side,

- Receiver receives the transmitted code word.
- The total number of 1’s in the received code word is counted.

Then, following cases are possible-

- If total number of 1’s is even and even parity is used, then receiver assumes that no error occurred.
- If total number of 1’s is even and odd parity is used, then receiver assumes that error occurred.
- If total number of 1’s is odd and odd parity is used, then receiver assumes that no error occurred.
- If total number of 1’s is odd and even parity is used, then receiver assumes that error occurred.

Parity Check Example-

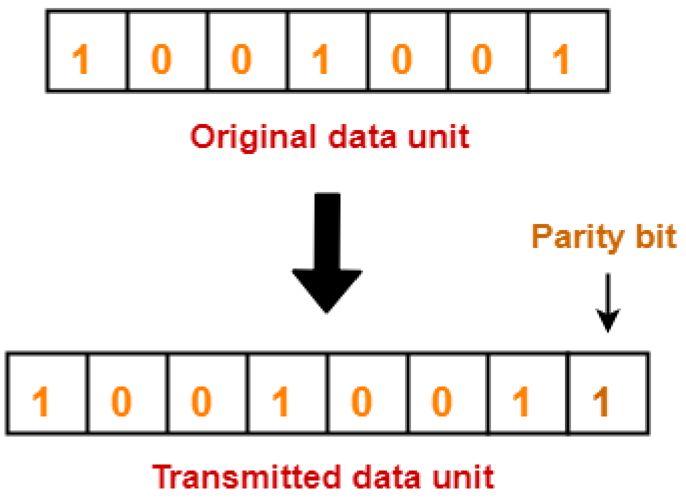
Consider the data unit to be transmitted is 1001001 and even parity is used.

Then,

At Sender Side-

- Total number of 1’s in the data unit is counted.

- Total number of 1's in the data unit = 3.
- Clearly, even parity is used and total number of 1's is odd.
- So, parity bit = 1 is added to the data unit to make total number of 1's even.
- Then, the code word 10010011 is transmitted to the receiver.



<u>At Receiver Side-</u>

- After receiving the code word, total number of 1's in the code word is counted.
- Consider receiver receives the correct code word = 10010011.
- Even parity is used and total number of 1's is even.
- So, receiver assumes that no error occurred in the data during the transmission.

Advantage-

- This technique is guaranteed to detect an odd number of bit errors (one, three, five and so on).
- If odd number of bits flip during transmission, then receiver can detect by counting the number of 1's.

Checksum-

Checksum is an error detection method.

Error detection using checksum method involves the following steps-

Step-01:

At sender side,

- If m bit checksum is used, the data unit to be transmitted is divided into segments of m bits.
- All the m bit segments are added.
- The result of the sum is then complemented using 1's complement arithmetic.
- The value so obtained is called as **checksum**.

Step-02:

- The data along with the checksum value is transmitted to the receiver.

Step-03:

At receiver side,

- If m bit checksum is being used, the received data unit is divided into segments of m bits.
- All the m bit segments are added along with the checksum value.
- The value so obtained is complemented and the result is checked.

Then, following two cases are possible-

Case-01: Result = 0

If the result is zero,

- Receiver assumes that no error occurred in the data during the transmission.
- Receiver accepts the data.

Case-02: Result ≠ 0

If the result is non-zero,

- Receiver assumes that error occurred in the data during the transmission.
- Receiver discards the data and asks the sender for retransmission.

Checksum Example-

Consider the data unit to be transmitted is-

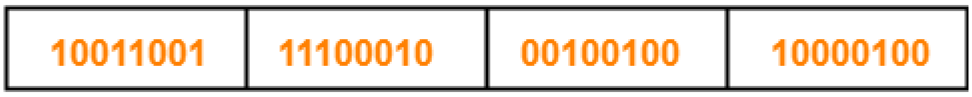
10011001111000100010010010000100

Consider 8 bit checksum is used.

Step-01:

At sender side,

The given data unit is divided into segments of 8 bits as-



10011001	11100010	00100100	10000100

Now, all the segments are added and the result is obtained as-

- 10011001 + 11100010 + 00100100 + 10000100 = 1000100011
- Since the result consists of 10 bits, so extra 2 bits are wrapped around.
- 00100011 + 10 = 00100101 (8 bits)
- Now, 1's complement is taken which is 11011010.
- Thus, checksum value = 11011010

Step-02:

- The data along with the checksum value is transmitted to the receiver.

Step-03:

At receiver side,

- The received data unit is divided into segments of 8 bits.
- All the segments along with the checksum value are added.
- Sum of all segments + Checksum value = 00100101 + 11011010 = 11111111
- Complemented value = 00000000

Since the result is 0, receiver assumes no error occurred in the data and therefore accepts it.

PRACTICE PROBLEM BASED ON CHECKSUM ERROR DETECTION METHOD-

Problem-

Checksum value of 1001001110010011 and 1001100001001101 of 16 bit segment is-

1. 1010101000011111
2. 1011111000100101
3. 1101010000011110
4. 1101010000111111

Solution-

We apply the above discussed algorithm to calculate the checksum.

- 1001001110010011 + 1001100001001101 = 10010101111100000
- Since, the result consists of 17 bits, so 1 bit is wrapped around and added to the result.
- 0010101111100000 + 1 = 0010101111100001
- Now, result consists of 16 bits.
- Now, 1’s complement is taken which is 1101010000011110
- Thus, checksum value = 1101010000011110

Thus, Option (C) is correct.

Cyclic Redundancy Check-

- Cyclic Redundancy Check (CRC) is an error detection method.
- It is based on binary division.

CRC Generator-

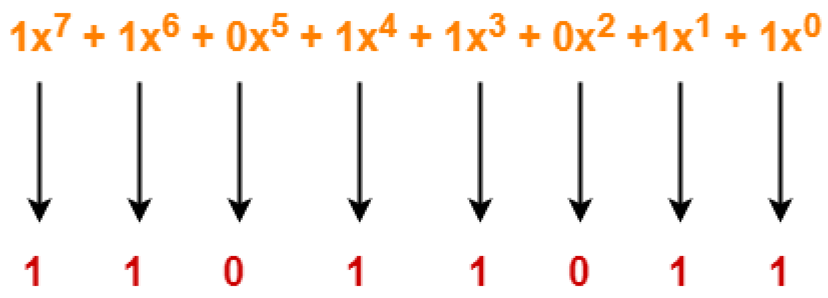
- CRC generator is an algebraic polynomial represented as a bit pattern.
- Bit pattern is obtained from the CRC generator using the following rule-

The power of each term gives the position of the bit and the coefficient gives the value of the bit.

Example-

Consider the CRC generator is $x^7 + x^6 + x^4 + x^3 + x + 1$.

The corresponding binary pattern is obtained as-



Thus, for the given CRC generator, the corresponding binary pattern is 11011011.

Properties Of CRC Generator-

The algebraic polynomial chosen as a CRC generator should have at least the following properties-

Rule-01:

- It should not be divisible by x.
- This condition guarantees that all the burst errors of length equal to the length of polynomial are detected.

Rule-02:

- It should be divisible by x+1.
- This condition guarantees that all the burst errors affecting an odd number of bits are detected.

Important Notes-

If the CRC generator is chosen according to the above rules, then-

- CRC can detect all single-bit errors
- CRC can detect all double-bit errors provided the divisor contains at least three logic 1's.
- CRC can detect any odd number of errors provided the divisor is a factor of x+1.
- CRC can detect all burst error of length less than the degree of the polynomial.
- CRC can detect most of the larger burst errors with a high probability.

Steps Involved-

Error detection using CRC technique involves the following steps-

Step-01: Calculation Of CRC At Sender Side-

At sender side,

- A string of n 0’s is appended to the data unit to be transmitted.
- Here, n is one less than the number of bits in CRC generator.
- Binary division is performed of the resultant string with the CRC generator.
- After division, the remainder so obtained is called as **CRC**.
- It may be noted that CRC also consists of n bits.

Step-02: Appending CRC To Data Unit-

At sender side,

- The CRC is obtained after the binary division.
- The string of n 0’s appended to the data unit earlier is replaced by the CRC remainder.

Step-03: Transmission To Receiver-

- The newly formed code word (Original data + CRC) is transmitted to the receiver.

Step-04: Checking at Receiver Side-

At receiver side,

- The transmitted code word is received.
- The received code word is divided with the same CRC generator.
- On division, the remainder so obtained is checked.

The following two cases are possible-

Case-01: Remainder = 0

If the remainder is zero,

- Receiver assumes that no error occurred in the data during the transmission.

- Receiver accepts the data.

Case-02: Remainder ≠ 0

If the remainder is non-zero,

- Receiver assumes that some error occurred in the data during the transmission.
- Receiver rejects the data and asks the sender for retransmission.

PRACTICE PROBLEMS BASED ON CYCLIC REDUNDANCY CHECK (CRC)-

Problem-01:

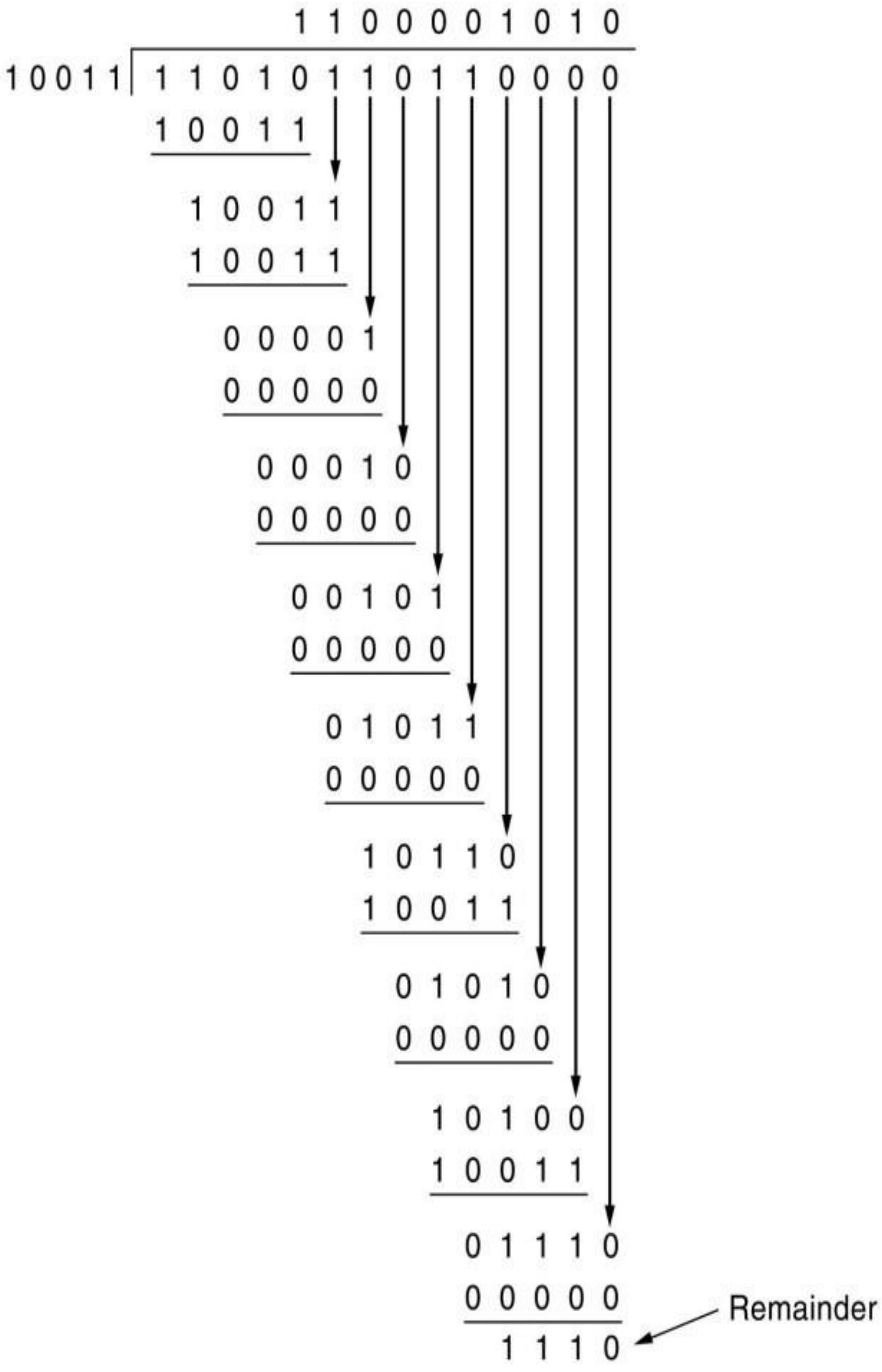
A bit stream 1101011011 is transmitted using the standard CRC method. The generator polynomial is x^4+x+1. What is the actual bit string transmitted?

Solution-

- The generator polynomial $G(x) = x^4 + x + 1$ is encoded as 10011.
- Clearly, the generator polynomial consists of 5 bits.
- So, a string of 4 zeroes is appended to the bit stream to be transmitted.
- The resulting bit stream is 1101011011**0000**.

Now, the binary division is performed as-

From here, CRC = 1110.



Now,

- The code word to be transmitted is obtained by replacing the last 4 zeroes of 11010110110**000** with the CRC.

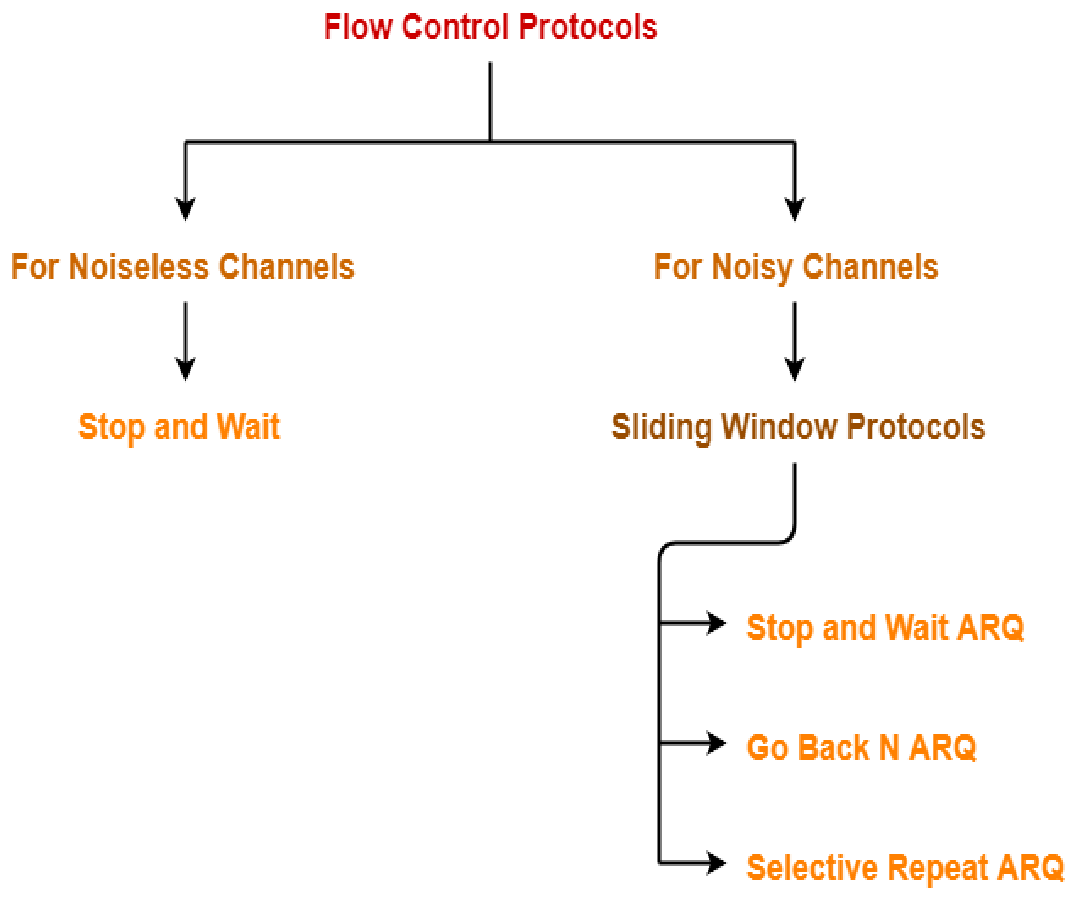
Thus, the code word transmitted to the receiver = 1101011011**1110**

Flow Control

A set of procedures which are used for restricting the amount of data that a sender can send to the receiver.

Flow Control Protocols-

There are various flow control protocols which are classified as-



Stop and Wait Protocol-

Stop and Wait Protocol is the simplest flow control protocol.

It works under the following assumptions-

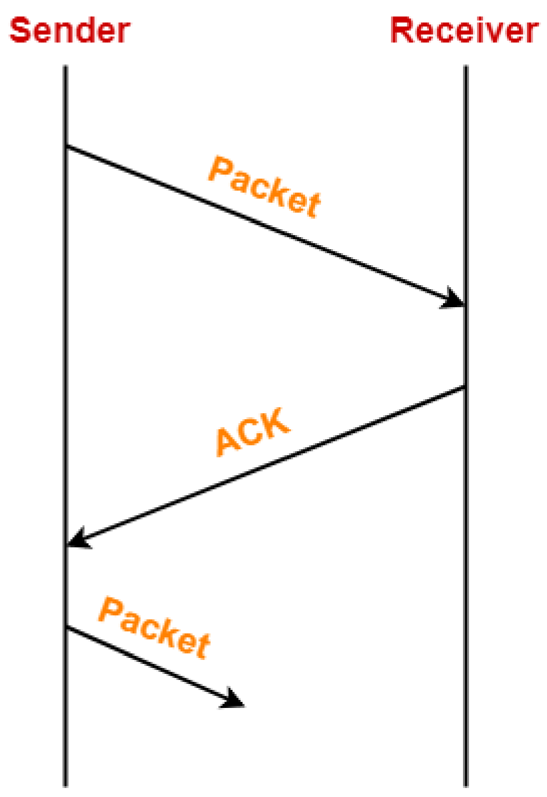
- Communication channel is perfect.
- No error occurs during transmission.

Working-

The working of a stop and wait protocol may be explained as-

- Sender sends a data packet to the receiver.
- Sender stops and waits for the acknowledgement for the sent packet from the receiver.
- Receiver receives and processes the data packet.
- Receiver sends an acknowledgement to the sender.
- After receiving the acknowledgement, sender sends the next data packet to the receiver.

These steps are illustrated below-



Stop and Wait Protocol

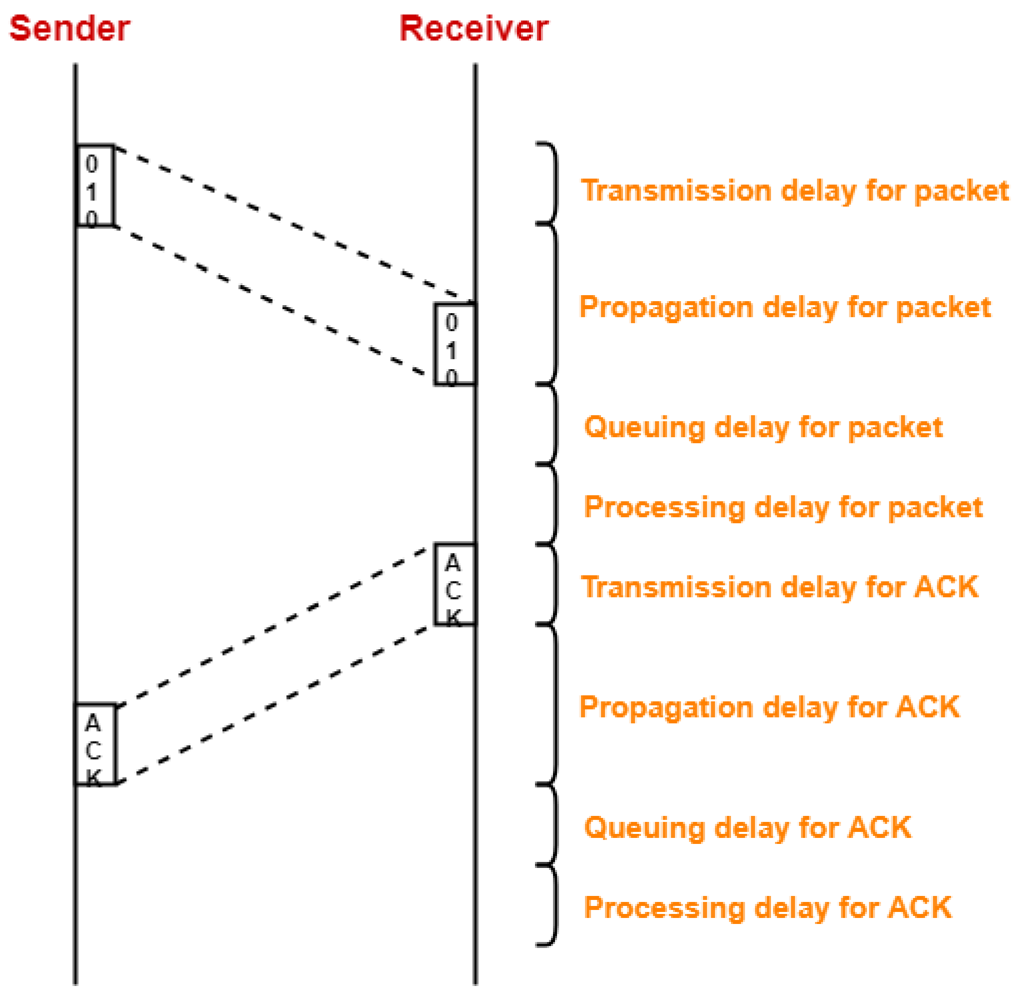
Analysis-

Now, let us analyze in depth how the transmission is actually carried out-

- Sender puts the data packet on the transmission link.
- Data packet propagates towards the receiver's end.

- Data packet reaches the receiver and waits in its buffer.
- Receiver processes the data packet.
- Receiver puts the acknowledgement on the transmission link.
- Acknowledgement propagates towards the sender's end.
- Acknowledgement reaches the sender and waits in its buffer.
- Sender processes the acknowledgement.

These steps are illustrated below-


Sender
Receiver
0
1
0
0
1
0
A
C
K
A
C
K
Transmission delay for packet
Propagation delay for packet
Queuing delay for packet
Processing delay for packet
Transmission delay for ACK
Propagation delay for ACK
Queuing delay for ACK
Processing delay for ACK

Stop and Wait Protocol

Stop and Wait ARQ-

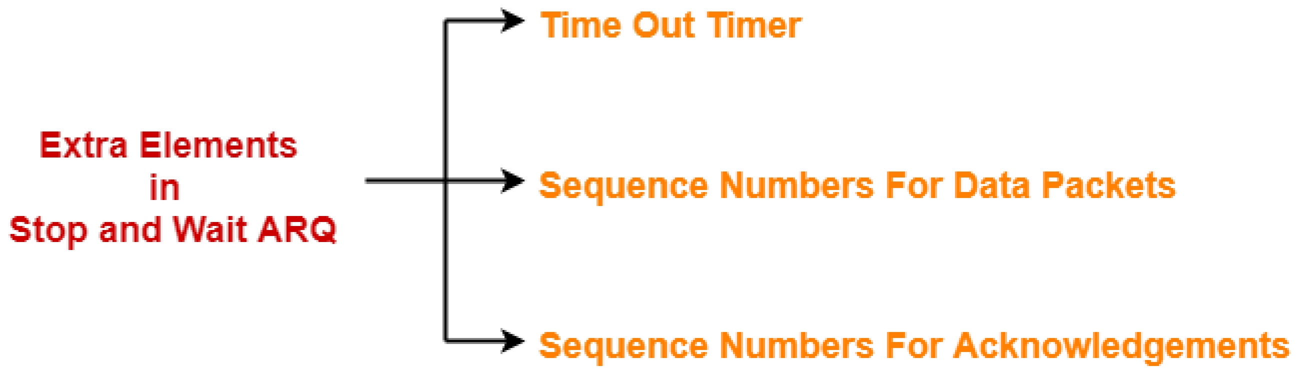
Stop and Wait ARQ is an improved and modified version of Stop and Wait protocol.

Stop and Wait ARQ assumes-

- The communication channel is noisy.
- Errors may get introduced in the data during the transmission.

Working-

- Stop and wait ARQ works similar to stop and wait protocol.
- It provides a solution to all the limitations of stop and wait protocol.
- Stop and wait ARQ includes the following three extra elements.


Extra Elements in Stop and Wait ARQ
Time Out Timer
Sequence Numbers For Data Packets
Sequence Numbers For Acknowledgements

Thus, we can say-

Stop and Wait ARQ

= Stop and Wait Protocol + Time Out Timer + Sequence Numbers for Data Packets and Acknowledgements

Number of Sequence Numbers Required-

Stop and wait ARQ is a one bit sliding window protocol where-

- Sender window size = 1
- Receiver window size = 1

Thus, in stop and wait ARQ,

Minimum number of sequence numbers required

= Sender Window Size + Receiver Window Size

= 1 + 1

= 2

Thus,

- Minimum number of sequence numbers required in Stop and Wait ARQ = 2.
- The two sequence numbers used are 0 and 1.

Go back N Protocol-

Go back N protocol is an implementation of a sliding window protocol.

The features and working of this protocol are explained in the following points-

Point-01:

In Go back N, sender window size is N and receiver window size is always 1.

In Go back N,

- Sender window size = N. Example in Go back 10, sender window size will be 10.
- Receiver window size is always 1 for any value of N.

Point-02:

Go back N uses cumulative acknowledgements.

In Go back N,

- Receiver maintains an acknowledgement timer.
- Each time the receiver receives a new frame, it starts a new acknowledgement timer.
- After the timer expires, receiver sends the cumulative acknowledgement for all the frames that are unacknowledged at that moment.

NOTE-

- A new acknowledgement timer does not start after the expiry of old acknowledgement timer.
- It starts after a new frame is received.

Point-03:

Go back N may use independent acknowledgements too.

- The above point does not mean that Go back N can not use independent acknowledgements.
- Go back N may use independent acknowledgements too if required.
- The kind of acknowledgement used depends on the expiry of acknowledgement timer.

Example-

- Consider after the expiry of acknowledgement timer, there is only one frame left to be acknowledged.
- Then, Go back N sends the independent acknowledgement for that frame.

Point-04:

Go back N does not accept the corrupted frames and silently discards them.

In Go back N,

- If receiver receives a frame that is corrupted, then it silently discards that frame.
- The correct frame is retransmitted by the sender after the time out timer expires.
- Silently discarding a frame means-

“Simply rejecting the frame and not taking any action”

(like not sending a NACK to the sender to send the correct frame)

Point-05:

Go back N does not accept out of order frames and silently discards them.

In Go back N,

- If receiver receives a frame whose sequence number is not what the receiver expects, then it silently discards that frame.

- All the following frames are also discarded.
- This is because receiver window size is 1 and therefore receiver can not accept out of order frames.

Point-06:

Go back N leads to retransmission of entire window if for any frame, no ACK is received by the sender.

In Go back N,

- Receiver silently discards the frame if it founds the frame to be either corrupted or out of order.
- It does not send any acknowledgement for such frame.
- It silently discards the following frames too.

Thus,

- If for any particular frame, sender does not receive any acknowledgement, then it understands that along with that frame, all the following frames must also have been discarded by the receiver.
- So, sender has to retransmit all the following frames too along with that particular frame.
- Thus, it leads to the retransmission of entire window.
- That is why, the protocol has been named as "**Go back N**".

Point-07:

Go back N leads to retransmission of lost frames after expiry of time out timer.

In Go back N,

- Consider a frame being sent to the receiver is lost on the way.
- Then, it is retransmitted only after time out timer expires for that frame at sender's side.

Selective Repeat Protocol-

Selective Repeat protocol or SR protocol is an implementation of a sliding window protocol.

The features and working of this protocol are explained in the following points-

Point-01:

In SR protocol, sender window size is always same as receiver window size.

In SR protocol,

- Sender window size = Receiver window size
- The size is of course greater than 1 otherwise the protocol will become **Stop and Wait ARQ**.
- If n bits are available for sequence numbers, then-

Sender window size = Receiver window size = $2^n/2 = 2^{n-1}$

Point-02:

SR protocol uses independent acknowledgements only.

In SR protocol,

- Receiver acknowledges each frame independently.
- As receiver receives a new frame from the sender, it sends its acknowledgement.

Point-03:

SR protocol does not accept the corrupted frames but does not silently discard them.

In SR protocol,

- If receiver receives a frame that is corrupted, then it does not silently discard that frame.
- Receiver handles the situation efficiently by sending a negative acknowledgement (NACK).
- Negative acknowledgement allows early retransmission of the corrupted frame.
- It also avoids waiting for the time out timer to expire at the sender side to retransmit the frame.

Point-05:

SR protocol accepts the out of order frames.

In SR protocol,

- Consider receiver receives a frame whose sequence number is not what the receiver expects.
- Then, it does not discard that frame rather accepts it and keeps it in its window.

Point-06:

SR protocol requires sorting at the receiver's side.

In SR protocol,

- Receiver window is implemented as a linked list.
- When receiver receives a new frame, it places the new frame at the end of the linked list.
- When the received frames are out of order, receiver performs the sorting.
- Sorting sorts the frames in the correct order.

Point-07:

SR protocol requires searching at the sender's side.

In SR protocol,

- Receiver does not reject the out of order frames.
- Receiver accepts the out of order frames and sort them later.
- Thus, only the missing frame has to be sent by the sender.
- For sending the missing frame, sender performs searching and finds the missing frame.
- Then, sender selectively repeats that frame.
- Thus, only the selected frame is repeated and not the entire window.
- That is why, the protocol has been named as "**Selective Repeat Protocol**".

Point-08:

SR protocol leads to retransmission of lost frames after expiry of time out timer.

In SR protocol,

- Consider a frame being sent to the receiver is lost on the way.
- Then, it is retransmitted only after time out timer expires for that frame at sender's side.

Access Control Methods :

CSMA / CD-

CSMA / CD stands for Carrier Sense Multiple Access / Collision Detection.

This access control method works as follows-

Step-01: Sensing the Carrier-

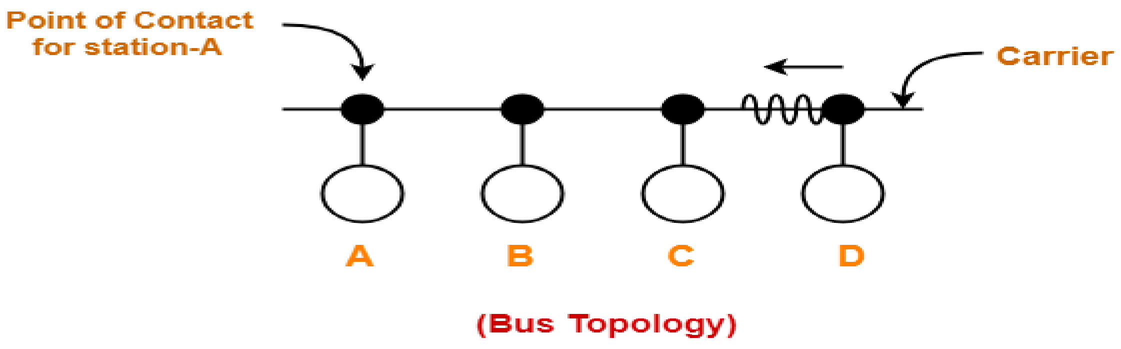
- Any station willing to transmit the data senses the carrier.
- If it finds the carrier free, it starts transmitting its data packet otherwise not.

How?

- Each station can sense the carrier only at its point of contact with the carrier.
- It is not possible for any station to sense the entire carrier.
- Thus, there is a huge possibility that a station might sense the carrier free even when it is actually not.

Example-

Consider the following scenario-



At the current instance,

- If station A senses the carrier at its point of contact, then it will find the carrier free.
- But the carrier is actually not free because station D is already transmitting its data.
- If station A starts transmitting its data now, then it might lead to a collision with the data transmitted by station D.

Step-02: Detecting the Collision-

In CSMA / CD,

- It is the responsibility of the transmitting station to detect the collision.
- For detecting the collision, CSMA / CD implements the following condition.
- This condition is followed by each station-

Transmission delay >= 2 x Propagation delay

Meaning-

According to this condition,

- Each station must transmit the data packet of size whose transmission delay is at least twice its propagation delay.
- If the size of data packet is smaller, then collision detection would not be possible.

Length Of Data Packet-

We know-

- Transmission delay = Length of data packet (L) / Bandwidth (B)
- Propagation delay = Distance between the two stations (D) / Propagation speed (V)

Substituting values in the above condition, we get-

$$L / B >= 2 \times D / V$$

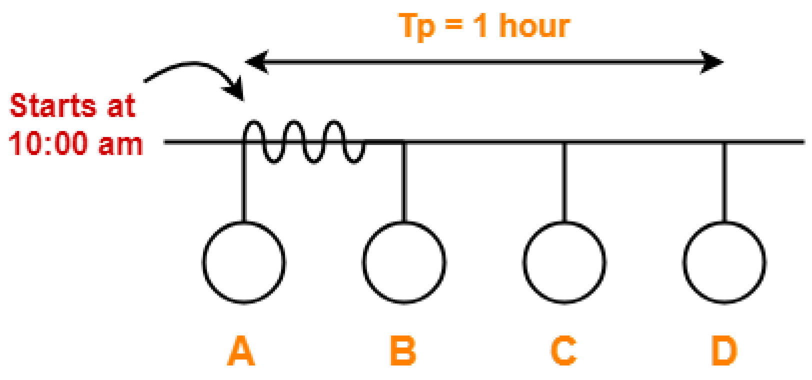
Thus,

$$\boxed{L >= 2 \times B \times D / V}$$

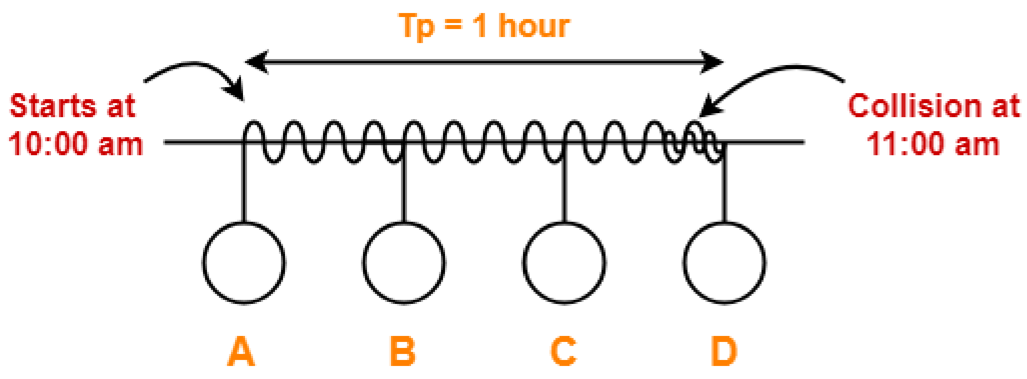
Understanding the Condition To Detect Collision With Example

- Consider at time 10:00 am, station A senses the carrier.
- It finds the carrier free and starts transmitting its data packet to station D.
- Let the propagation delay be 1 hour.

(We are considering station D for the worst case)



- Let us consider the scenario at time 10:59:59:59 when the packet is about to reach the station D.
- At this time, station D senses the carrier.
- It finds the carrier free and starts transmitting its data packet.
- Now, as soon as station D starts transmitting its data packet, a collision occurs with the data packet of station A at time 11:00 am.



- After collision occurs, the collided signal starts travelling in the backward direction.
- The collided signal takes 1 hour to reach the station A after the collision has occurred.
- For station A to detect the collided signal, it must be still transmitting the data.
- So, transmission delay of station A must be >= 1 hour + 1 hour >= 2 hours to detect the collision.
- That is why, for detecting the collision, condition is $T_t >= 2T_p$.

Two cases are possible-

Case-01:

If no collided signal comes back during the transmission,

- It indicates that no collision has occurred.
- The data packet is transmitted successfully.

Case-02:

If the collided signal comes back during the transmission,

- It indicates that the collision has occurred.
- The data packet is not transmitted successfully.
- Step-03 is followed.

Step-03: Releasing Jam Signal-

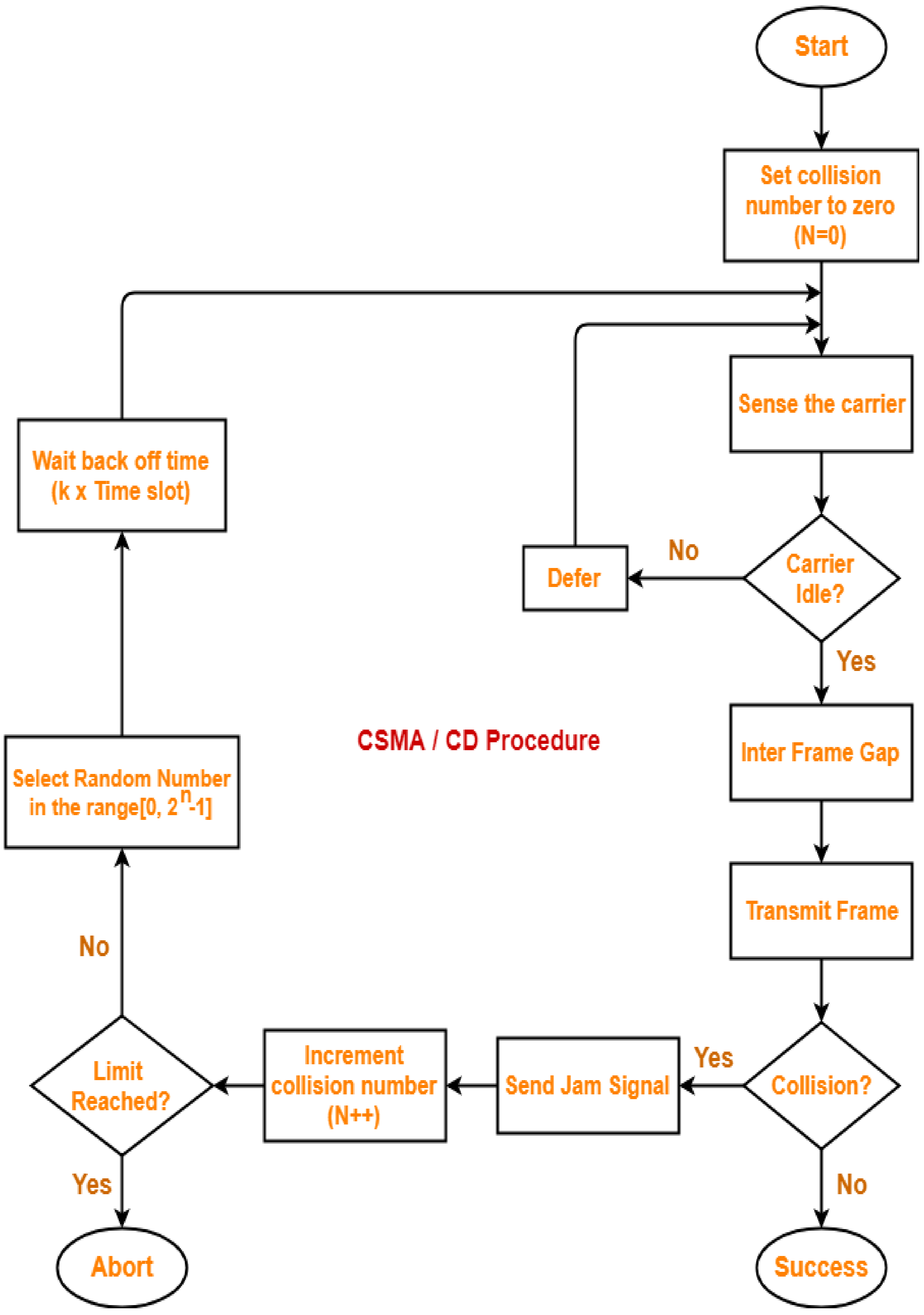
- Jam signal is a 48 bit signal.
- It is released by the transmitting stations as soon as they detect a collision.
- It alerts the other stations not to transmit their data immediately after the collision.
- Otherwise, there is a possibility of collision again with the same data packet.
- Ethernet sends the jam signal at a frequency other than the frequency of data signals.
- This ensures that jam signal does not collide with the data signals undergone collision.

Step-04: Waiting For Back Off Time-

- After the collision, the transmitting station waits for some random amount of time called as **back off time**.
- After back off time, it tries transmitting the data packet again.
- If again the collision occurs, then station again waits for some random back off time and then tries again.
- The station keeps trying until the back off time reaches its limit.
- After the limit is reached, station aborts the transmission.
- Back off time is calculated using **Back Off Algorithm**.

CSMA / CD Flowchart-

The following CSMA / CD flowchart represents the CSMA / CD procedure-



Token Passing

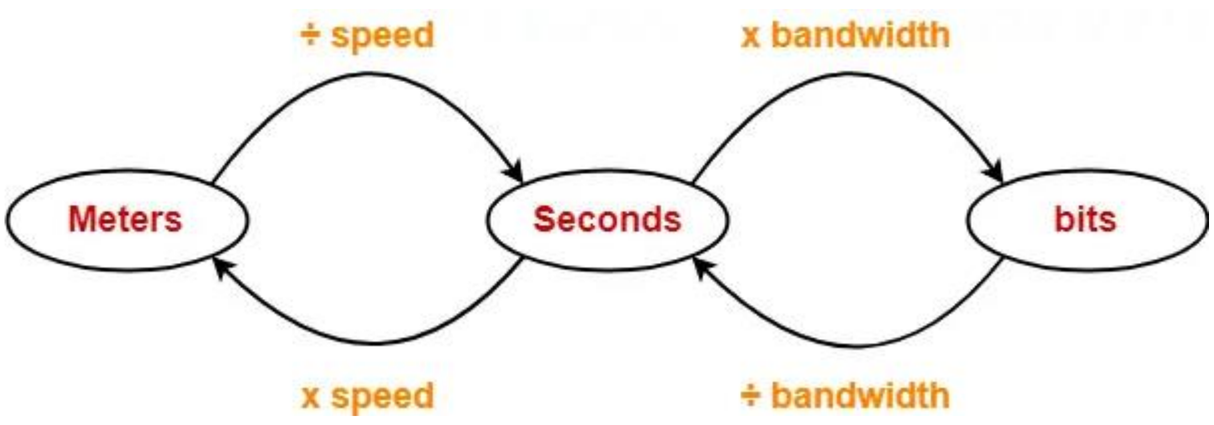
Before discussing Token Passing, let us discuss few important concepts required for the discussion.

Time Conversions-

In token passing,

Time may be expressed in seconds, bits or meters.

To convert the time from one unit to another, we use the following conversion chart-



Conversion Chart

Token Passing Terminology-

The following terms are frequently used-

1. Token
2. Ring Latency
3. Cycle Time

1. Token-

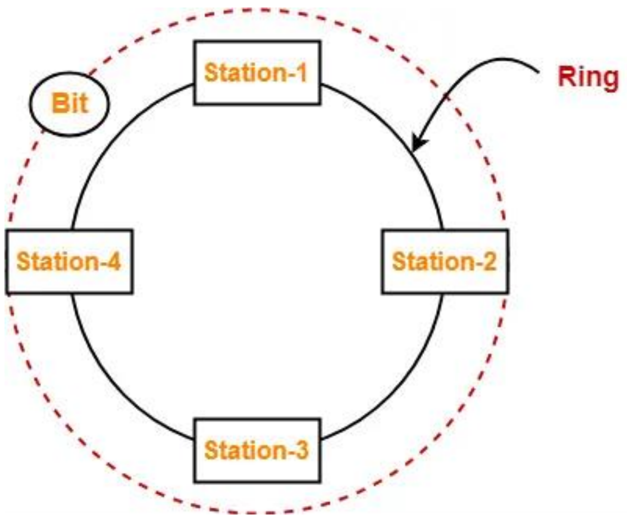
- A token is a small message composed of a special bit pattern.

- It represents the permission to send the data packet.

- A station is allowed to transmit a data packet if and only if it possess the token otherwise not.

2. Ring Latency-

Time taken by a bit to complete one revolution of the ring is called as ring latency.



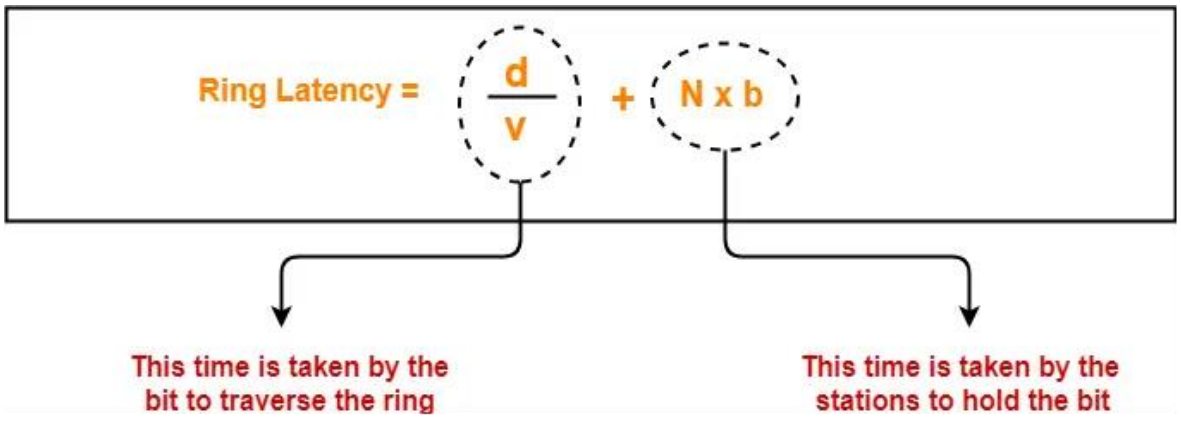
Let us derive the expression for ring latency.

If-

- Length of the ring = d
- Speed of the bit = v
- Number of stations = N
- Bit delay at each station = b

(Bit delay is the time for which a station holds the bit before transmitting to the other side)

Then-


Ring Latency = d / v + N x b
This time is taken by the bit to traverse the ring
This time is taken by the stations to hold the bit

Notes-

- d / v is the propagation delay (Tp) expressed in seconds.
- Generally, bit delay is expressed in bits.
- So, both the terms (d / v and N x b) have different units.
- While calculating the ring latency, both the terms are brought into the same unit.

After conversion, we have-

$$\text{Ring Latency} = \left(\frac{d}{v} + \frac{N \times b}{B}\right) \text{ sec}$$

$$= \left(T_p + \frac{N \times b}{B}\right) \text{ sec}$$

OR

$$\text{Ring Latency} = \left(\frac{d \times B}{v} + N \times b\right) \text{ bits}$$

$$= \left(T_p \times B + N \times b\right) \text{ bits}$$

3. Cycle Time-

Time taken by the token to complete one revolution of the ring is called as cycle time.

If-

- Length of the ring = d
- Speed of the bit = v
- Number of stations = N
- Token Holding Time = THT

  (Token Holding Time is the time for which a station holds the token before transmitting to the other side)

Then-

$$\text{Cycle Time} = \frac{d}{v} + N \times THT$$

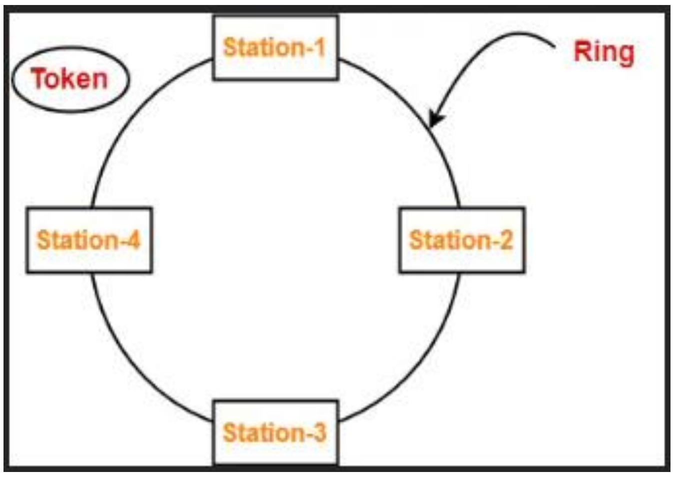
$$= T_p + N \times THT$$

Now, we start discussing about Token Passing Access Control Method.

Token Passing-

In this access control method,

- All the stations are logically connected to each other in the form of a ring.
- The access of stations to the transmission link is governed by a token.
- A station is allowed to transmit a data packet if and only if it possess the token otherwise not.
- Each station passes the token to its neighboring station either clockwise or anti-clockwise.



Assumptions-

Token passing method assumes-

- Each station in the ring has the data to send.
- Each station sends exactly one data packet after acquiring the token.

Efficiency-

Efficiency (η) = Useful Time / Total Time

In one cycle,

Useful time = Sum of transmission delay of N stations since each station sends 1 data packet = N x Tt

Total Time = Cycle time = Tp + N x THT

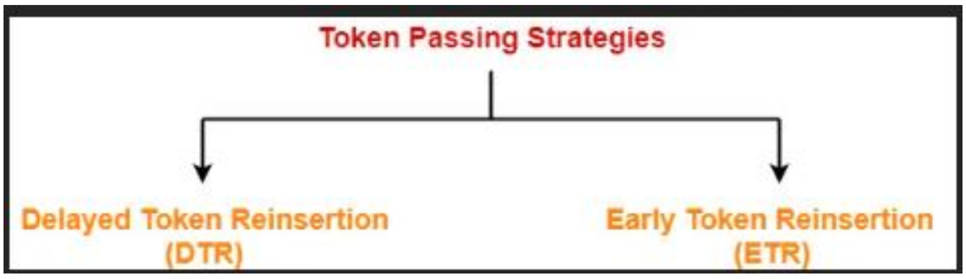
Thus,

$$\text{Efficiency } (\eta) = \frac{N \times T_t}{T_p + N \times THT}$$

Token Holding Time depends on the strategy implemented.

Token Passing Strategies-

The following 2 strategies are used in token passing-


Token Passing Strategies
Delayed Token Reinsertion (DTR)
Early Token Reinsertion (ETR)

1. Delayed Token Reinsertion (DTR)
2. Early Token Reinsertion (ETR)

1. Delayed Token Reinsertion -

In this strategy,

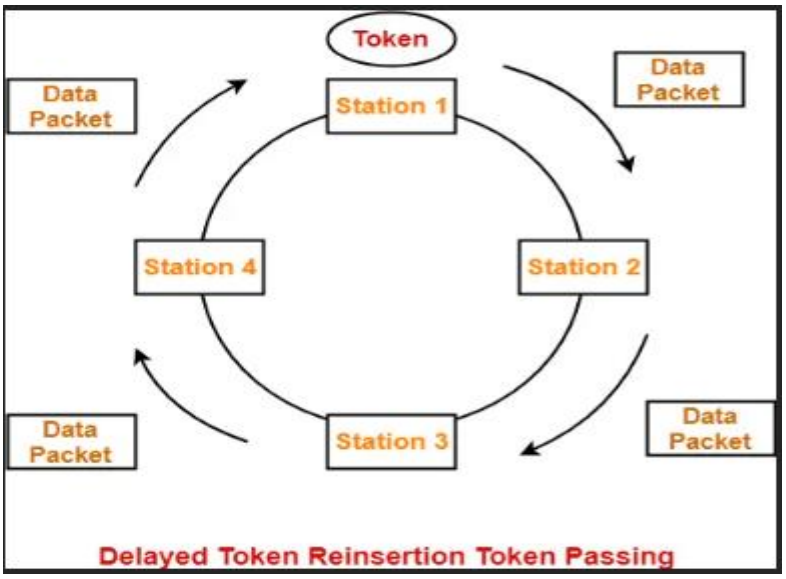
- Station keeps holding the token until the last bit of the data packet transmitted by it takes the complete revolution of the ring and comes back to it.

Working-

After a station acquires the token,

- It transmits its data packet.
- It holds the token until the data packet reaches back to it.
- After data packet reaches to it, it discards its data packet as its journey is completed.
- It releases the token.

The following diagram illustrates these steps for station-1. Same procedure is repeated at every station.



Token Holding Time-

$$\boxed{\text{Token Holding Time (THT)} = \text{Transmission delay} + \text{Ring Latency}}$$

We know,

Ring Latency = Tp + N x bit delay

Assuming bit delay = 0 (in most cases), we get-

$$\boxed{\text{Token Holding Time} = Tt + Tp}$$

Efficiency-

Substituting THT = Tt + Tp in the efficiency expression, we get-

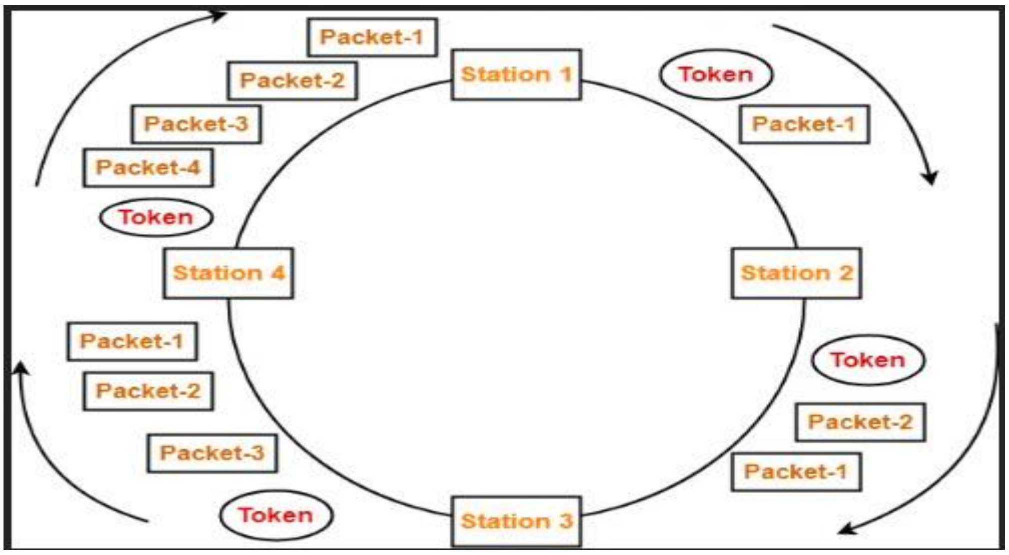
$$\text{Efficiency } (\eta) = \frac{N \times T_t}{T_p + N \times (T_t + T_p)}$$

2. Early Token Reinsertion-

In this strategy,

Station releases the token immediately after putting its data packet to be transmitted on the ring.

Working-



Step-01: At Station-1:

Station-1

- Acquires the token
- Transmits packet-1
- Releases the token

Step-02: At Station-2:

Station-2

- Receives packet-1
- Transmits packet-1
- Acquires the token
- Transmits packet-2
- Releases the token

Step-03: At Station-3:

Station-3

- Receives packet-1
- Transmits packet-1
- Receives packet-2
- Transmits packet-2
- Acquires the token
- Transmits packet-3
- Releases the token

Step-04: At Station-4:

Station-4

- Receives packet-1
- Transmits packet-1
- Receives packet-2
- Transmits packet-2
- Receives packet-3
- Transmits packet-3
- Acquires the token
- Transmits packet-4
- Releases the token

Step-05: At Station-1:

- Receives packet-1
- Discards packet-1 (as its journey is completed)
- Receives packet-2

- Transmits packet-2
- Receives packet-3
- Transmits packet-3
- Receives packet-4
- Transmits packet-4
- Acquires the token
- Transmits packet-1 (new)
- Releases the token

In this manner, the cycle continues.

Token Holding Time-

Token Holding Time (THT) = Transmission delay of data packet = Tt

Efficiency-

Substituting THT = Tt in the efficiency expression, we get-

$$\text{Efficiency } (\eta) = \frac{N \times T_t}{T_p + N \times T_t}$$

OR

$$\text{Efficiency } (\eta) = \frac{1}{1 + \frac{a}{N}}$$

Differences between DTR and ETR-

Delay Token Retransmission (DTR)	**Early Token Retransmission (ETR)**
Each station holds the token until its data packet reaches back to it.	Each station releases the token immediately after putting its data packet on the ring.

There exists only one data packet on the ring at any given instance.	There exists more than one data packet on the ring at any given instance.
It is more reliable than ETR.	It is less reliable than DTR.
It has low efficiency as compared to ETR.	It has high efficiency as compared to ETR.

Important Notes-

Note-01:

In token passing,

- It is the responsibility of each transmitting station to remove its own data packet from the ring.

Note-02:

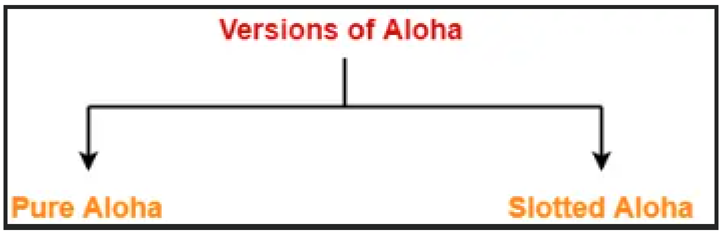
- While solving questions,

  If the strategy used is not mentioned, then consider Early Token Retransmission strategy.

Aloha

There are two different versions of Aloha -

1. Pure Aloha
2. Slotted Aloha



1. Pure Aloha-

- It allows the stations to transmit data at any time whenever they want.
- After transmitting the data packet, station waits for some time.

Then, following 2 cases are possible-

Case-01:

- Transmitting station receives an acknowledgement from the receiving station.

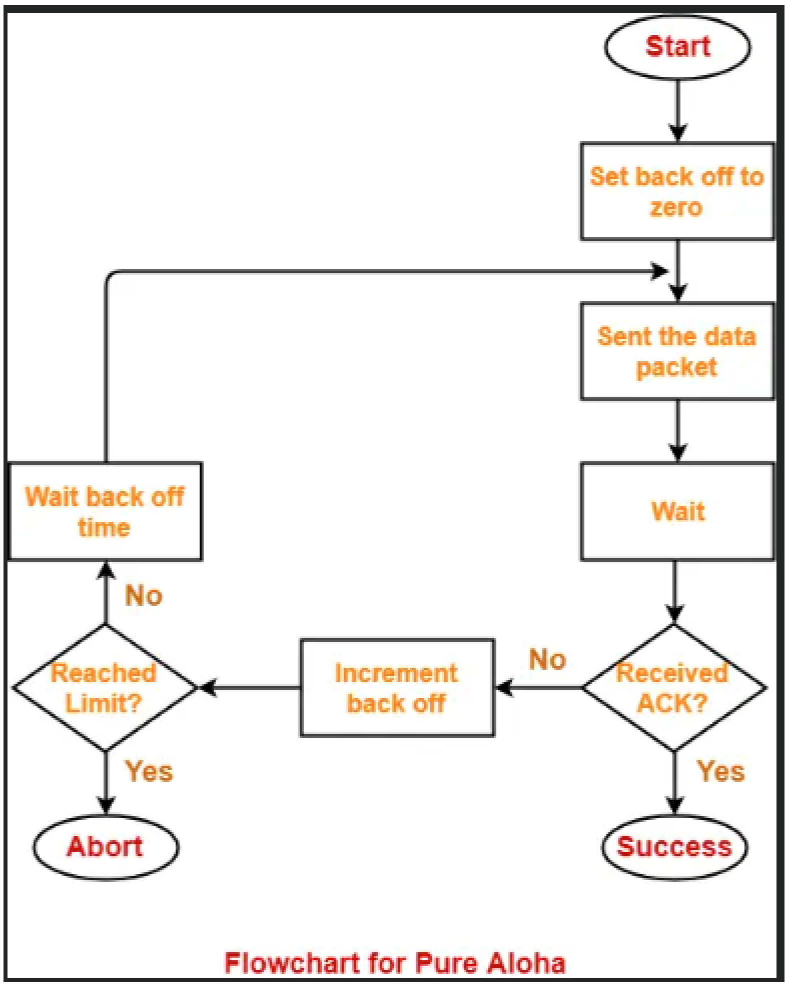
- In this case, transmitting station assumes that the transmission is successful.

Case-02:

- Transmitting station does not receive any acknowledgement within specified time from the receiving station.
- In this case, transmitting station assumes that the transmission is unsuccessful.

Then,

- Transmitting station uses a Back Off Strategy and waits for some random amount of time.
- After back off time, it transmits the data packet again.
- It keeps trying until the back off limit is reached after which it aborts the transmission.



Efficiency -

$$\text{Efficiency of Pure Aloha } (\eta) = G \times e^{-2G}$$

where G = Number of stations willing to transmit data

Maximum Efficiency -

For maximum efficiency,

- We put dη / dG = 0
- Maximum value of η occurs at G = ½
- Substituting G = 1/2 in the above expression, we get-

Maximum efficiency of Pure Aloha = 1/2 x $e^{-2 \times 1/2}$

= 1 / 2e

= 0.184

= 18.4%

Thus,

$\boxed{\text{Maximum Efficiency of Pure Aloha } (\eta) = 18.4\%}$

The maximum efficiency of Pure Aloha is very less due to large number of collisions.

2. Slotted Aloha-

- Slotted Aloha divides the time of shared channel into discrete intervals called as time slots.
- Any station can transmit its data in any time slot.
- The only condition is that station must start its transmission from the beginning of the time slot.
- If the beginning of the slot is missed, then station has to wait until the beginning of the next time slot.
- A collision may occur if two or more stations try to transmit data at the beginning of the same time slot.

Efficiency-

$\boxed{\text{Efficiency of Slotted Aloha } (\eta) = G \text{ x e-G}}$

where G = Number of stations willing to transmit data at the beginning of the same time slot

Maximum Efficiency-

For maximum efficiency,

- We put dη / dG = 0
- Maximum value of η occurs at G = 1
- Substituting G = 1 in the above expression, we get-

Maximum efficiency of Slotted Aloha

= 1 x e^{-1}

= 1 / e

= 0.368

= 36.8%

Thus,

$\boxed{\text{Maximum Efficiency of Slotted Aloha } (\eta) = 36.8\%}$

The maximum efficiency of Slotted Aloha is high due to less number of collisions.

Difference Between Pure Aloha And Slotted Aloha-

Pure Aloha	**Slotted Aloha**
Any station can transmit the data at any time.	Any station can transmit the data at the beginning of any time slot.
The time is continuous and not globally synchronized.	The time is discrete and globally synchronized.
Vulnerable time in which collision may occur = 2 x Tt	Vulnerable time in which collision may occur = Tt
Probability of successful transmission of data packet = G x e^{-2G}	Probability of successful transmission of data packet = G x e^{-G}
Maximum efficiency = 18.4% (Occurs at G = 1/2)	Maximum efficiency = 36.8% (Occurs at G = 1)
The main advantage of pure aloha is its simplicity in implementation.	The main advantage of slotted aloha is that it reduces the number of collisions to half and doubles the efficiency of pure aloha.