COMPUTER COMMUNICATION AND NETWORKS

BCA III SEM(NEP)

NOTES

Prepared By

Mr. Prabhu Kichadi, BE, MTECH

9880437187

UNIT – III

The Data Link Layer: Data Link Layer design issues, Error detection – Single parity checking, Checksum, polynomial codes – CRC, Error correction Hamming code, Elementary data link protocols, sliding window protocols.

The Data Link Layer: The data link layer in the OSI (Open System Interconnections) Model, is in between the physical layer and the network layer. This layer converts the raw transmission facility provided by the physical layer to a reliable and error-free link.

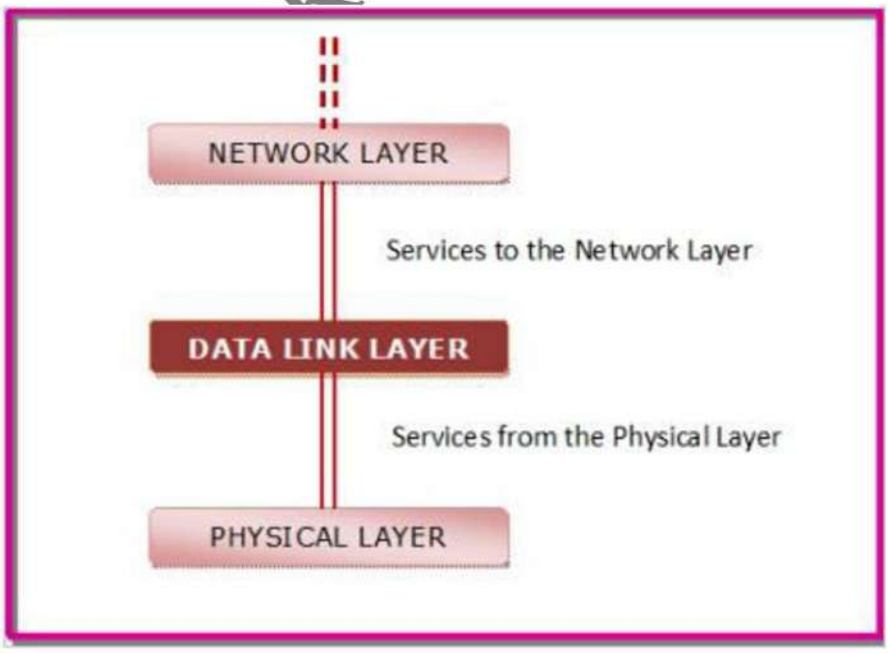
The main functions and the design issues of this layer are

1. **Providing services to the network layer**
2. **Framing**
3. **Error Control**
4. **Flow Control**

1. **Providing services to the network layer:** In the OSI Model, each layer uses the services of the layer below it and provides services to the layer above it. The data link layer uses the services offered by the physical layer. The primary function of this layer is to provide a well-defined service interface to network layer above it.

The types of services provided can be of three types –

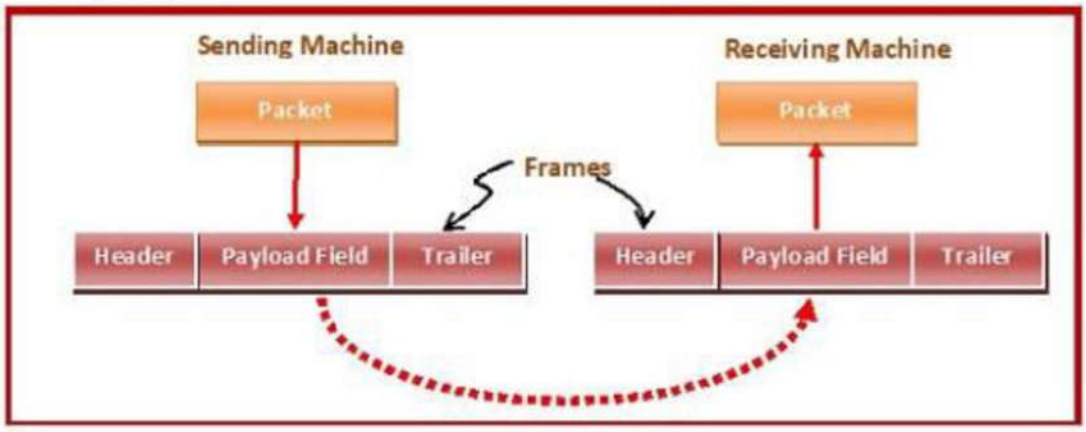
- Unacknowledged connectionless service
- Acknowledged connectionless service
- Acknowledged connection - oriented service



2. **Framing:** The data link layer encapsulates each data packet from the network layer into frames that are then transmitted.

   A frame has three parts, namely –

   - Frame Header
   - Payload field that contains the data packet from network layer
   - Trailer



3. Error Control: The data link layer ensures error free link for data transmission. The issues it caters to with respect to error control are

- Dealing with transmission errors
- Sending acknowledgement frames in reliable connections
- Retransmitting lost frames
- Identifying duplicate frames and deleting them
- Controlling access to shared channels in case of broadcasting

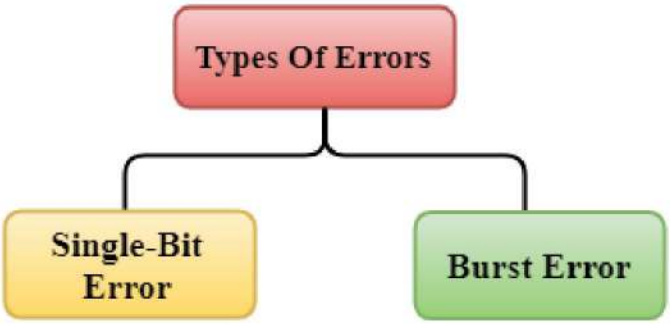
4. Flow Control: The data link layer regulates flow control so that a fast sender does not drown a slow receiver. When the sender sends frames at very high speeds, a slow receiver may not be able to handle it. There will be frame losses even if the transmission is error-free. The two common approaches for flow control are –

- Feedback based flow control
- Rate based flow control

Error Detection

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



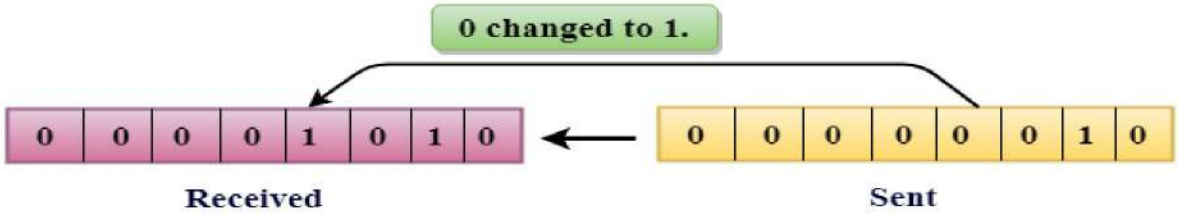
Errors can be classified into two categories:

- Single-Bit Error
- Burst Error

1. Single-Bit Error: The only one bit of a given data unit is changed from 1 to 0 or from 0 to 1.

Single bit error does not appear more likely in Serial Data Transmission. For example, Sender sends the data at 10 Mbps, this means that the bit lasts only for 1 ?s and for a single-bit error to occurred, a noise must be more than 1 ?s.

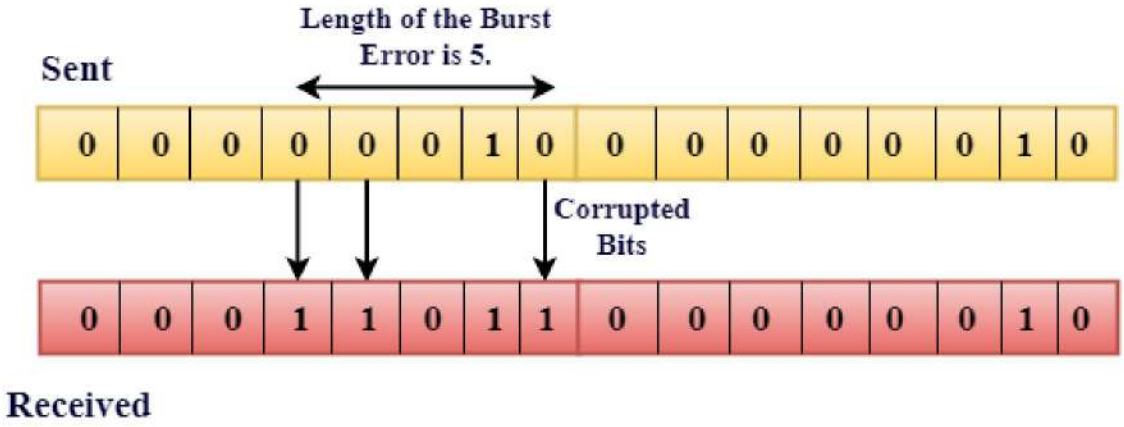
Single-Bit Error mainly occurs in Parallel Data Transmission. For example, if eight wires are used to send the eight bits of a byte, if one of the wire is noisy, then single-bit is corrupted per byte.



2. 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.

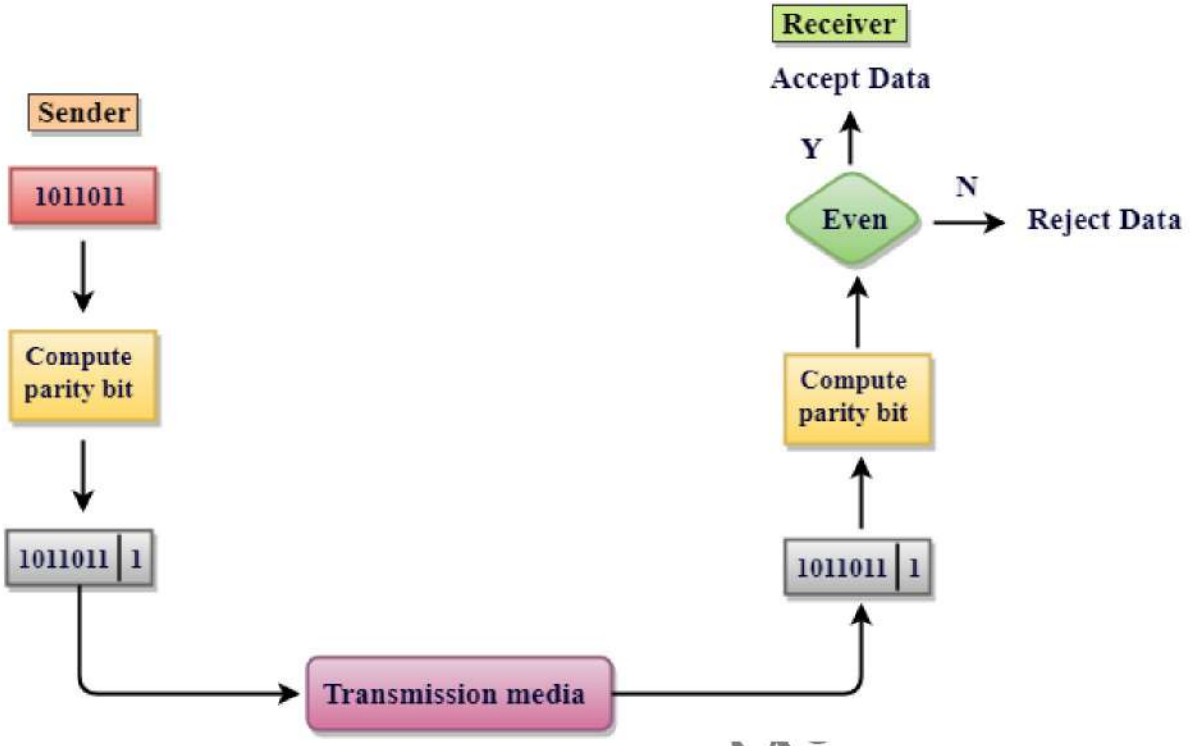


Error Detecting Techniques:

- **Single Parity Check**
- **Checksum**
- **Cyclic Redundancy Check**

1. Single Parity Check:

- Single Parity checking is the simple mechanism and inexpensive to detect the errors.
- In this technique, a redundant bit is also known as a parity bit which is appended at the end of the data unit so that the number of 1s becomes even. Therefore, the total number of transmitted bits would be 9 bits.
- If the number of 1s bits is odd, then parity bit 1 is appended and if the number of 1s bits is even, then parity bit 0 is appended at the end of the data unit.
- At the receiving end, the parity bit is calculated from the received data bits and compared with the received parity bit.
- This technique generates the total number of 1s even, so it is known as even-parity checking



2. Checksum: This is a block code method where a checksum is created based on the data values in the data blocks to be transmitted using some algorithm and appended to the data. When the receiver gets this data, a new checksum is calculated and compared with the existing checksum. A non-match indicates an error.

For error detection by checksums, data is divided into fixed sized frames or segments.

- **Sender's End** – The sender adds the segments using 1's complement arithmetic to get the sum. It then complements the sum to get the checksum and sends it along with the data frames.
- **Receiver's End** – The receiver adds the incoming segments along with the checksum using 1's complement arithmetic to get the sum and then complements it.

If the result is zero, the received frames are accepted; otherwise they are discarded.

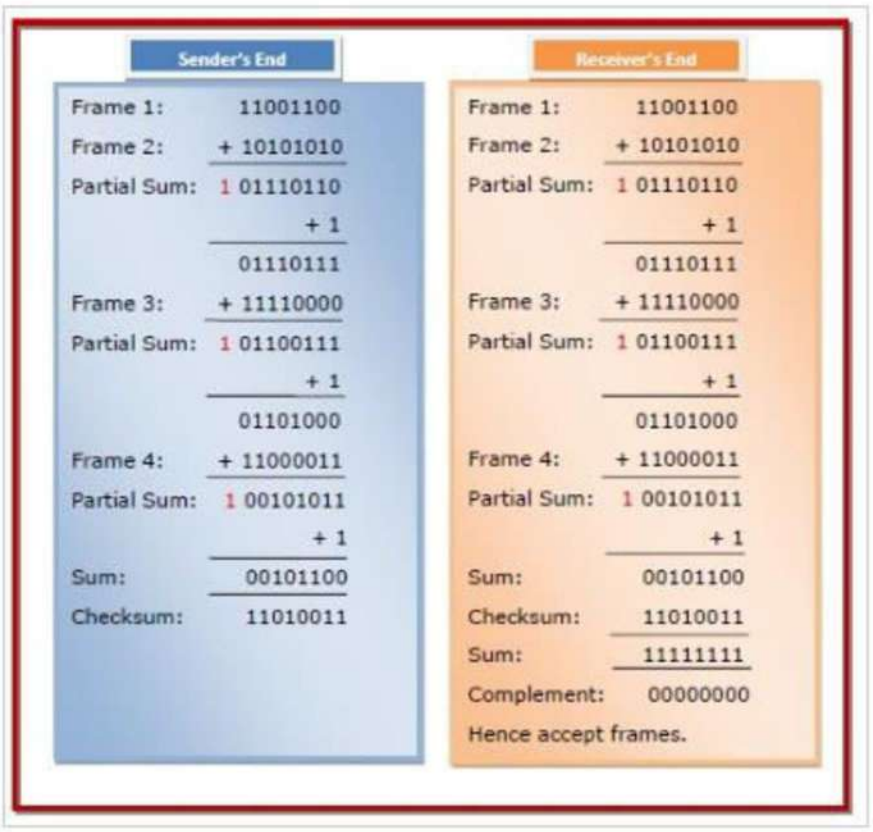
Example

Suppose that the sender wants to send 4 frames each of 8 bits, where the frames are 11001100, 10101010, 11110000 and 11000011.

The sender adds the bits using 1s complement arithmetic. While adding two numbers using 1s complement arithmetic, if there is a carry over, it is added to the sum.

After adding all the 4 frames, the sender complements the sum to get the checksum, 11010011, and sends it along with the data frames.

The receiver performs 1s complement arithmetic sum of all the frames including the checksum. The result is complemented and found to be 0. Hence, the receiver assumes that no error has occurred.



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.

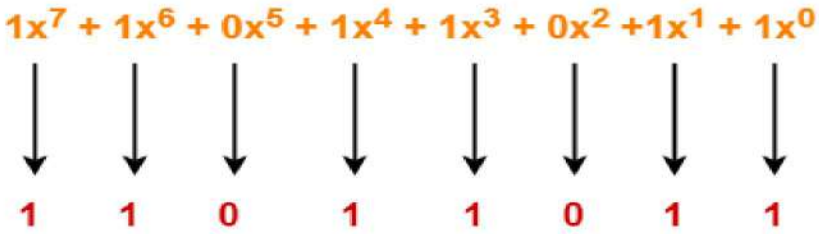
3. Cyclic Redundancy Check(CRC):

- CRC is also going to generate some bits called redundant bits. When the message and divisor(Generator Polynomial) is given.

- CRC or Cyclic Redundancy Check in computer networks is an error detection method. CRC Generator is used to generate CRC

At sender side:

Polynomial Representation:



Algorithm:

At Sender side,

1. Assume that original data unit as Original Message O(x).

2. For a given G(x), calculate the number of redundant bits (Zero) using

Redundant zero bits = n-1 number of G(x) bits.

3. Append (n-1) zero bits to the original message.

M(x) = O(x) + Redundant Bits.

4. Perform binary with X-OR operation on M(x)

i.e M(x) / G(x)

5. Finally what we get Reminder is the CRC Code bits.

6. Replace redundant bits with CRC bits, now it forms a transmitting message to receiver.

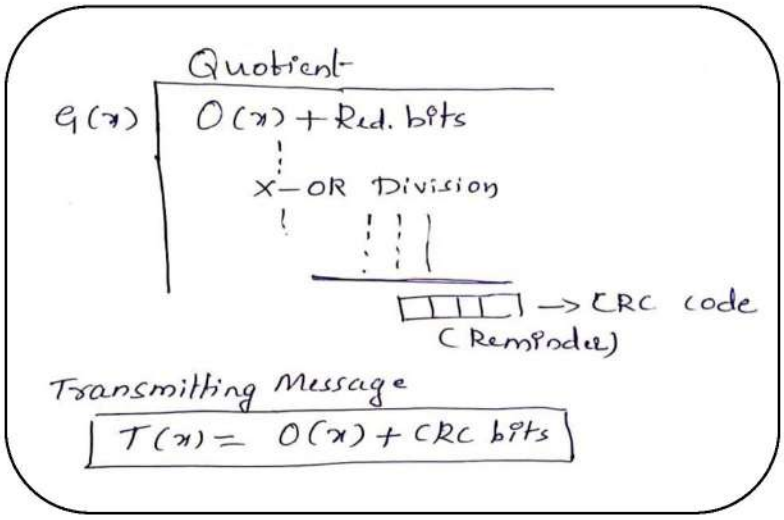
T(x) = O(x) + CRC Bits.

At Receiver side,

7. Perform same Binary division with X-OR Operation, T(x) / G(x).

8. If we get reminder as 0000 It means that There is NO ERROR in the transmitted message, else it has an ERROR in the message.

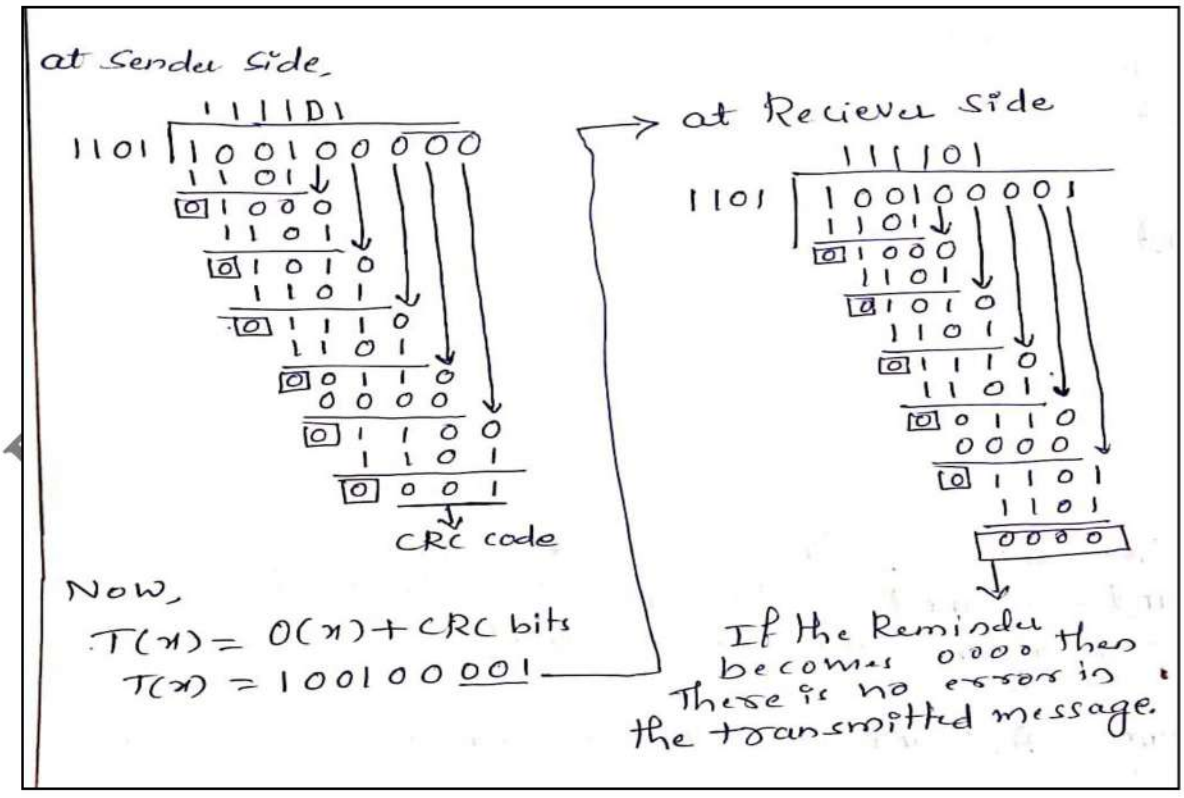
Snippet:



Ex1: For the data block(unit) 100100 with the divisor 1101

Solution:

O(x) = 100100
G(x) = 1101
Redundant Bits = 4 -1 = 3
M(x) = O(x) + Redundant bits=>
M(x) = 1 0 0 1 0 0 0 0 0
Perform Binary Division with X-OR Operation.



Ex2: Generate the CRC Code for the message 1101011011 using polynomial generator 10011.

Ex3: A bit stream 10011101 is transmitted using the standard CRC method. The generator polynomial is x3+1.

Error correction: Error correction is the process of detecting errors in transmitted messages and reconstructing the original error-free data. Error correction ensures that corrected and error-free messages are obtained at the receiver side.

Hamming Code:

- It can applied to data units of any length, and it is used to detect and correct single bit errors.
- It contains Data bits + Parity(Redundant bits).
- Positions of the Parity bits are determined by 2^n where n = 0, 1, 2, 3, 4, etc.
- The bits positions are 7, 6, 5, 4,

Redundant bits: Redundant bits are extra binary bits that are generated and added to the information-carrying bits of data transfer to ensure that no bits were lost during the data transfer.

Hamming code structure:

- **All bit positions that are power of 2 are marked as parity bits. Ex(2^0, 2^1, 2^2, 2^3 that is 1, 2, 4, 8, 16 ……..)**
- **Lets consider 7 bit data unit, then the structure of the hamming code will be,**

D7	D6	D5	P4	D3	P2	P1
7	6	5	4	3	2	1

- Next step is to determine the value of the parity bit based type of parity(even or odd).
- The value of parity is
- P1 value is decide by the D3, D5, D7 bits(Even or odd parity)
- P2 value is decided by the D3, D6, D7
- P4 value is decided D5, D6, D7.

Parity bits must be sent along with data bits.

Ex1: data set 1011, calculate hamming code parity bits.

- Given 7 bit hamming code structure,

1	0	1	0	1	0	1
d7	d6	d5	p4	d3	p2	p1

calculate the the values for parity bits assuming even parity.
p1-> d3,d5, d7 => 1 1 1 then p1 => 1
p2-> d3,d6, d7 => 1 0 1 then p2 => 0
p4-> d5,d6, d7 => 1 0 1 then p4 => 0

Assuming because of noise, an error 1 0 1 0 1 0 1 is interpreted as
1 1 1 0 1 0 1

At receiver side,
Calculate the parity bits:
P1 -> 1 1 1 => 1 valid
P2 -> 1 1 1 => 0 invalid
P4 -> 1 1 1 => 0 invalid.
Hence the data is having an error.

Example: If the 7 bit hamming code word received by a receiver is 1011011. Assuming the even parity state whether the received code word is correct or wrong. If wrong locate the bit having an error.

Given data code word is :
1 0 1 1 0 1 1

1	0	1	1	0	1	1
D7	**d6**	**d5**	**p4**	**d3**	**p2**	**p1**

Calculate parity bits:

P1 => p1,d3,d5,d7 => 1 0 1 1 => hence p1 => 1

P2 => p2, d3,d6,d7 => 1 0 0 1 => hence p2 => 0
P3 => p4,d5,d6,d7 => 1 1 0 1 => hence p4 => 1

(p4,p2,p1) => $1\ 0\ 1_{(2)} = 5_{(10)}$

Hence, we find the location of the bit error, ie 5th position d5 value must be corrected to 1 0 0 1 0 1 1

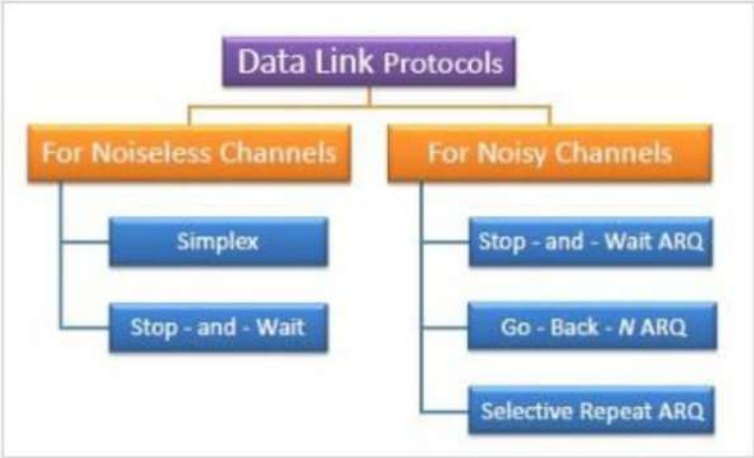
Elementary data link protocols

Protocols in the data link layer are designed so that this layer can perform its basic functions: framing, error control and flow control. Framing is the process of dividing bit - streams from physical layer into data frames whose size ranges from a few hundred to a few thousand bytes.

Framing: Framing is the process of dividing bit - streams from physical layer into data frames.

Types of Data Link Protocols

Data link protocols can be broadly divided into two categories, depending on whether the transmission channel is noiseless or noisy.



Simplex Protocol

- The Simplex protocol is hypothetical protocol designed for unidirectional data transmission over an ideal channel, i.e. a channel through which transmission can never go wrong.
- The sender simply sends all its data available onto the channel as soon as they are available its buffer. The receiver is assumed to process all incoming data instantly.
- It is hypothetical since it does not handle flow control or error control.

Stop – and – Wait Protocol

- Stop – and – Wait protocol is for noiseless channel too. It provides unidirectional data transmission without any error control facilities. However, it provides for flow control so that a fast sender does not drown a slow receiver.
- The receiver has a finite buffer size with finite processing speed. The sender can send a frame only when it has received indication from the receiver that it is available for further data processing.

Sender:

Rule 1) Send one data packet at a time.
Rule 2) Send the next packet only after receiving acknowledgement for the previous.

Receiver:

Rule 1) Send acknowledgement after receiving and consuming a data packet.
Rule 2) After consuming packet acknowledgement need to be sent (Flow Control)

For Noisy Channels:

Stop – and – Wait ARQ Protocol:

Characteristics

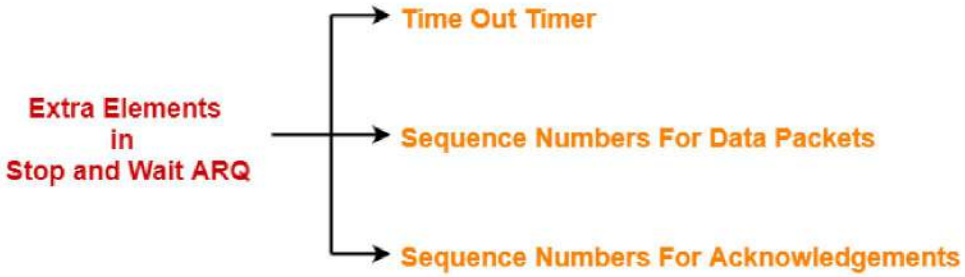
- Used in Connection-oriented communication.
- It offers error and flows control
- It is used in Data Link and Transport Layers
- Stop and Wait for ARQ mainly implements the Sliding Window Protocol concept with Window Size 1

The main problem faced by the Stop and Wait protocol is the occurrence of deadlock due to-

1. Loss of data packet
2. Loss of acknowledgement

Working-

- Stop and wait ARQ works like 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.

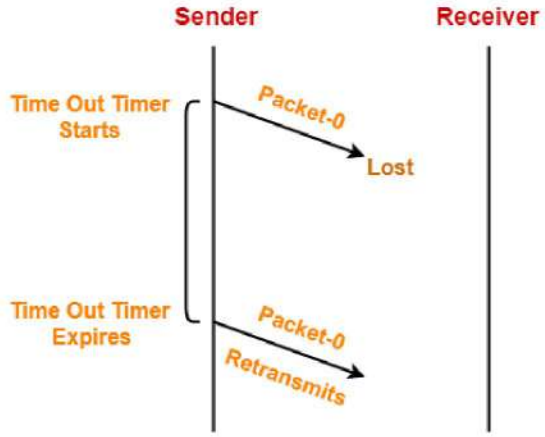


Stop and Wait ARQ = Stop and Wait Protocol + Time Out Timer + Sequence Numbers for Data Packets and Acknowledgements

How Stop and Wait ARQ Solves All Problems?

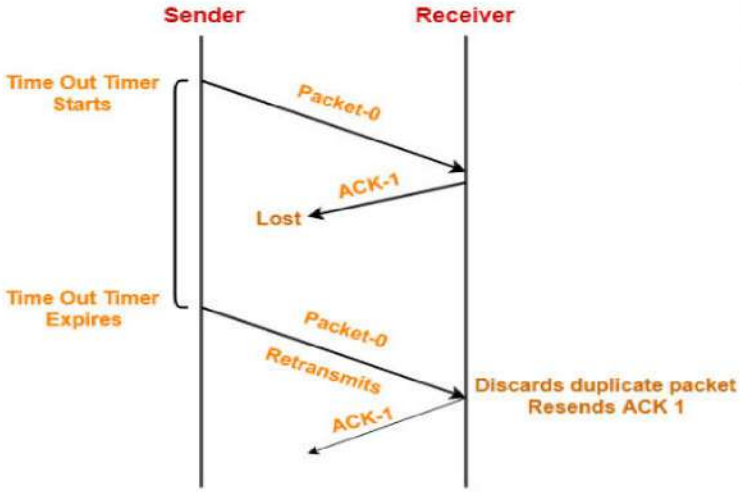
1. Problem of Lost Data Packet-

- Time out timer helps to solve the problem of lost data packet.
- After sending a data packet to the receiver, sender starts the time out timer.
- If the data packet gets acknowledged before the timer expires, sender stops the time out timer.
- If the timer goes off before receiving the acknowledgement, sender retransmits the same data packet.
- After retransmission, sender resets the timer.
- This prevents the occurrence of deadlock.



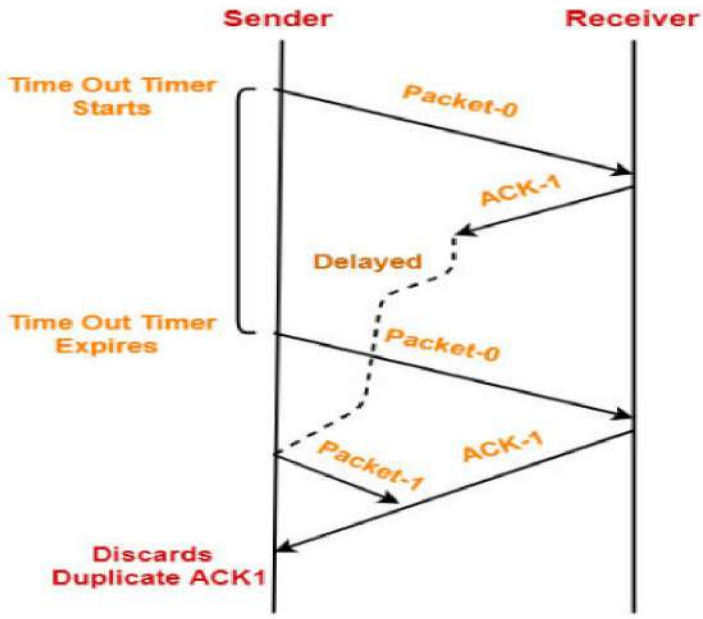
2. Problem of Lost Acknowledgement-

- Sequence number on data packets help to solve the problem of delayed acknowledgement.
- Consider the acknowledgement sent by the receiver gets lost.
- Then, sender retransmits the same data packet after its timer goes off.
- The sequence number on the data packet helps the receiver to identify the duplicate data packet.
- Receiver discards the duplicate packet and re-sends the same acknowledgement.



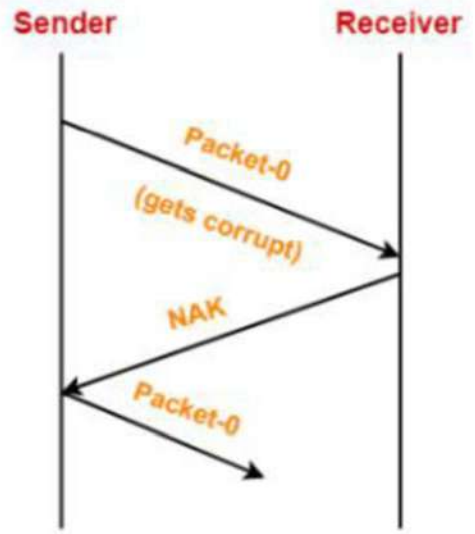
3. Problem of Delayed Acknowledgement-

Sequence number on acknowledgements help to solve the problem of delayed acknowledgement.



4. Problem of Damaged Packet-

- If receiver receives a corrupted data packet from the sender, it sends a negative acknowledgement (NAK) to the sender.
- NAK requests the sender to send the data packet again.



Stop and Wait Protocol	Stop and Wait ARQ
It assumes that the communication channel is perfect and noise free.	It assumes that the communication channel is imperfect and noisy.
Data packet sent by the sender can never get corrupt.	Data packet sent by the sender may get corrupt.
There is no concept of negative acknowledgements.	A negative acknowledgement is sent by the receiver if the data packet is found to be corrupt.
There is no concept of time out timer.	Sender starts the time out timer after sending the data packet.
There is no concept of sequence numbers.	Data packets and acknowledgements are numbered using sequence numbers.

Sliding Window Protocols:

The sliding window is a technique for sending multiple frames at a time. It controls the data packets between the two devices where reliable and gradual delivery of data frames is needed.

In this technique, each frame has sent from the sequence number. The sequence numbers are used to find the missing data in the receiver end. The purpose of the sliding window technique is to avoid duplicate data, so it uses the sequence number.

Types of Sliding Window Protocol

Sliding window protocol has two types:

1. Go-Back-N ARQ
2. Selective Repeat ARQ

Sliding window protocols have two parameters:

1. Sequence number.

2. Sliding window.