1. Explain the significance of data link layer, explain the design issues of data link layer.
[7M] **[Dec/Jan-2022-23] Understand CO3**

Ans :

Datalink is a layer in the Open System Interconnections. It is the second layer in between the physical layer and the network layer. It manages the connection between the two nodes. Data links integrate certain methods like error control, flow control, and associated link management functions.

Some of the main functions of the data link layer include providing a straightforward service interface to the network layer, framing flow control and error recognition, and frame formatting.

Types of Data Link Layers

Data Link layer is mainly of two types −

- Logical Link Control Sub-Layer (LLC)
- Media Access Control Sub-layer (MAC)

Logical Link Control Sub-Layer (LLC)

It gives logic for the data link. Therefore, it manages the synchronization, flow control, and error recognition features of the data link. LLC is used for Error Recovery and User Addressing. It executes the Control flow functioning.

Media Access Control Sub-layer (MAC)

MAC is the sub-layer of the data link. It manages the flow and is multitudinous for the transmission medium. This layer manages the channeling of data packages. MAC is used for sending the data over the network interface card.

MAC is used in recognition of errors. It accomplishes the special labeling to stations directly linked to LAN.

Design Issues of the Data Link Layer

Service Agreement to the Network Layer

The main aim of this service is to give services to the network layer. The concept of this layer is to transfer the data from the network layer on the source machine to the layer on the destination machine. Communication between the two data layers is done via Data Link Control Protocol.

Here are the important services given by the Data Link layer to the Network layer −

- Unacknowledged connectionless services
- Acknowledged connectionless service
- Acknowledged-oriented service

Framing

Service given to the network layer data link uses the services given to the physical layer. The source machine sends the data in the form of frames to the destination machine. Starting point and the endpoint of the frame should point out so that the destination machine can easily identify the frame.

The data link layer breaks the bitstream and calculates the checksum for each layer. At the destination layer, the checksum is enumerated. Therefore, breaking the bitstream by placing spaces and time gaps is known as **framing**.

It is quite difficult and dangerous to count on timing and mark the starting and endpoints of each frame. Simple techniques used for framing are −

- Character Count
- Starting and ending character with character filling
- Starting and ending flags with little fillings.

Flow Control

Flow control is done to stop the data flow at the receiver's end. The transmitter will transfer the frames very quickly to the receiver. However, the receiver will not accept them as quickly as the sender sends because the sender runs on a lightly loaded machine while the receiver runs on a heavily loaded machine.

It doesn't matter if the transmission is error-free at some point. The receiver will not be able to control the frames as they will arrive.

For stopping the transmission, a mechanism is there which requests the transmitter to block the incorrect messages.

Error Control

It is done so that there is no copying of the frames for the safe delivery of the frames at the destination. In addition, Positive and negative acceptance is sent about the incoming frames.

Therefore, if the sender gets positive acceptance, that means the frame appears safely, while negative appearance means that something is wrong with the frame and the frame will be retransferred.

The timer is put at the receiver's and sender's end. Besides, the sequence number is given to the outgoing transmission. So that receiver will easily identify that it is a retransmitted frame. It is one of the main parts of the data link layer responsibilities.

Physical Address of Frames

The data link layer adds a header to the frame to describe the sender or receiver's physical address.

2. What is need of framing, various methods for implementing framing in data link layer [7M]**[Dec/Jan-2022-23]Analyze CO3**

Ans :

Framing is a point-to-point connection between two devices that consists of a wire in which data is transmitted as a stream of bits.

- We have given two devices with an adaptor attached to each one. This adaptor will be sending the data into signals which will flow through the cable and will be received by the physical layer of other devices.
- Framing in a computer network uses frames to send/receive the data. The data link layer packs bits into frames such that each frame is distinguishable from another.
- The data link layer prepares a packet for transport across local media by encapsulating it with a header and a trailer to create a frame.
- The frame is defined as the data in telecommunications that moves between various network points.

Usually, a frame moves bit-by-bit serially and consists of a trailer field and header field that frames the information. These frames are understandable only by the data link layer.

Parts of a Frame :

1. Frame Header: It consists of the frame's source and destination address.
2. Payload Field: It contains the message to be delivered.
3. Flag: It points to the starting and the ending of the frame.
4. Trailer: It contains the error detection and correction bits.

Types of Framing in Computer Networks

Framing is further divided into two types:

1. Fixed-size Framing

In this type of framing, the size of the frame is fixed, and hence the frame length acts as a delimiter of the frame.

This framing doesn't require the additional boundary bits to identify the frame's start and end - for example: If a device is sending 200 bits of data and the 50 bits of frame size is fixed, then after receiving the 50 bits of data, the receiver will automatically know that the next 50 bits are of frame two and so on.

Drawback: It goes through internal fragmentation when the data size is less than the frame size. Padding is a solution to avoid such a situation.

2. Variable size Framing

Variable size framing is a method of dividing data into frames where each frame can have a different size. In this approach, the length of each frame is determined dynamically based on the amount of data that needs to be transmitted at a given time. For example, It is possible that out of 200 bits of data, 100 bits constitute frame 1, 25 bits constitute frame two, and the rest bits constitute frame 3.

The advantage of variable size framing is that it allows for more efficient use of network bandwidth. Smaller frames are used when less data needs to be transmitted, reducing overhead and increasing the efficiency of the data transfer. On the other hand, larger frames are used when there is a larger amount of data to be sent, which helps in maximizing data throughput.

Variable size framing is commonly used in various network protocols, especially those that require adaptive data transmission, such as streaming applications or real-time communication systems. This method enables the network to adapt to varying data requirements and optimize data transfer based on the current network conditions and data load.

3. Explain about the data link layer frame and frame fields.

[7M] **[Apr/May-2019]**

Data Link Layer Frame

A frame is a unit of communication in the data link layer. Data link layer takes the packets from the Network Layer and encapsulates them into frames. If the frame size becomes too large, then the packet may be divided into small sized frames. At receiver' end, data link layer picks up signals from hardware and assembles them into frames.

Fields of a Data Link Layer Frame

A data link layer frame has the following parts:

- **Frame Header**: It contains the source and the destination addresses of the frame and the control bytes.
- **Payload field**: It contains the message to be delivered.
- **Trailer**: It contains the error detection and error correction bits. It is also called a Frame Check Sequence (FCS).
- **Flag**: Two flag at the two ends mark the beginning and the end of the frame.

Frame Header

A frame header contains the destination address, the source address and three control fields *kind*, *seq*, and *ack* serving the following purposes:

- *kind*: This field states whether the frame is a data frame or it is used for control functions like error and flow control or link management etc.
- *seq*: This contains the sequence number of the frame for rearrangement of out – of – sequence frames and sending acknowledgments by the receiver.
- *ack*: This contains the acknowledgment number of some frame, particularly when piggybacking is used.

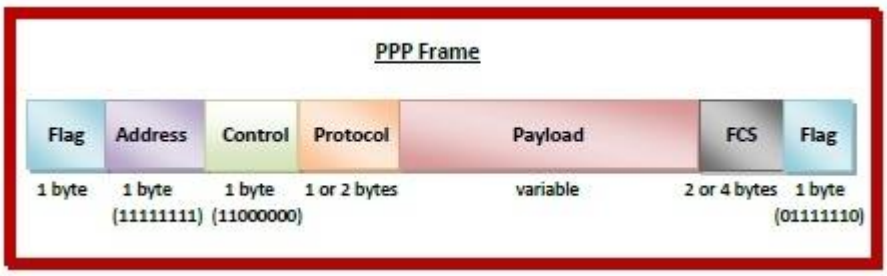
Specific Data Link Layer Frames

The structure of the data link layer frame may be specialized according to the type of protocol used. Let us study the frame structure used in two protocols: Point – to – Point Protocol (PPP) and High-level Data Link Control (HDLC).

Point – to – Point Protocol

Point – to – Point Protocol (PPP) is a communication protocol of the data link layer that is used to transmit multiprotocol data between two directly connected (point-to-point) computers. The fields of a PPP frame are:

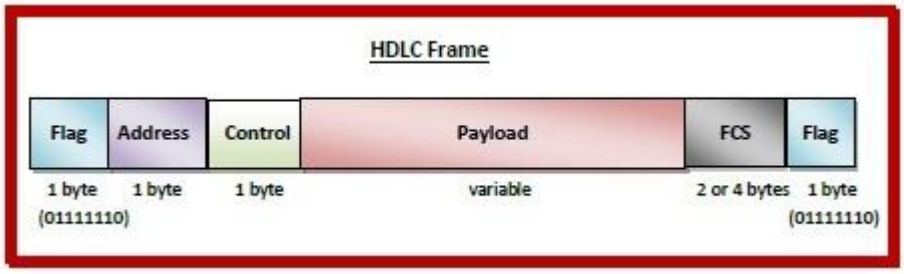
- **Flag**: It is of 1 byte that with bit pattern 01111110.
- **Address**: 1 byte which is set to 11111111 in case of the broadcast.
- **Control**: 1 byte set to a constant value of 11000000.
- **Protocol**: 1 or 2 bytes that define the type of data contained in the payload field.
- **Payload**: This carries the data from the network layer. The maximum length of the payload field is 1500 bytes.
- **FCS**: It is a 2 byte or 4 bytes frame check sequence for error detection. The standard code used is CRC (cyclic redundancy code).



High-level Data Link Control

High-level Data Link Control (HDLC) is a group of communication protocols of the data link layer for transmitting data between network points or nodes. The fields of an HDLC frame are:

- **Flag**: It is an 8-bit sequence with bit pattern 01111110.
- **Address**: It contains the address of the receiver. The address field may be from 1 byte to several bytes.
- **Control**: It is 1 or 2 bytes containing flow and error control information.
- **Payload**: This carries the data from the network layer. Its length may vary from one network to another.
- **FCS**: It is a 2 byte or 4 bytes frame check sequence for error detection. The standard code used is CRC (cyclic redundancy code)



4. Explain character, Bit stuffing for framing.

[7M]**[Dec/Jan-2022-23]**

Bit stuffing is the mechanism of inserting one or more non-information bits into a message to be transmitted, to break up the message sequence, for synchronization purpose.

Purpose of Bit Stuffing

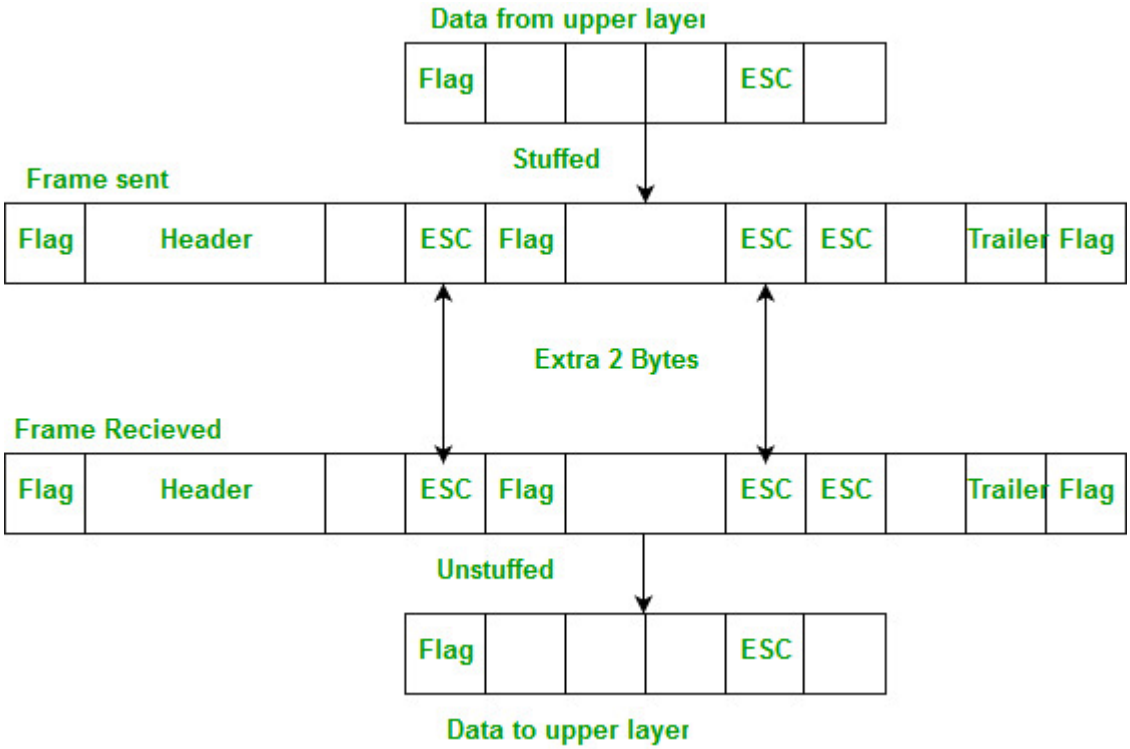
In Data Link layer, the stream of bits from the physical layer is divided into data frames. The data frames can be of fixed length or variable length. In variable - length framing, the size of each frame to be transmitted may be different. So, a pattern of bits is used as a delimiter to mark the end of one frame and the beginning of the next frame. However, if the pattern occurs in the message, then mechanisms needs to be incorporated so that this situation is avoided.

The two common approaches are −

- **Byte - Stuffing** − A byte is stuffed in the message to differentiate from the delimiter. This is also called character-oriented framing.
- **Bit - Stuffing** − A pattern of bits of arbitrary length is stuffed in the message to differentiate from the delimiter. This is also called bit - oriented framing.

Byte stuffing is a byte (usually escape character(ESC)), which has a predefined bit pattern is added to the data section of the frame when there is a character with the same pattern as the flag. Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data, not a flag. But the problem arises when the text contains one or more escape characters followed by a flag. To solve this problem, the escape characters that are part of the text are marked by another escape character i.e., if the escape character is part of the text, an extra one is added to show that the second one is part of the text.

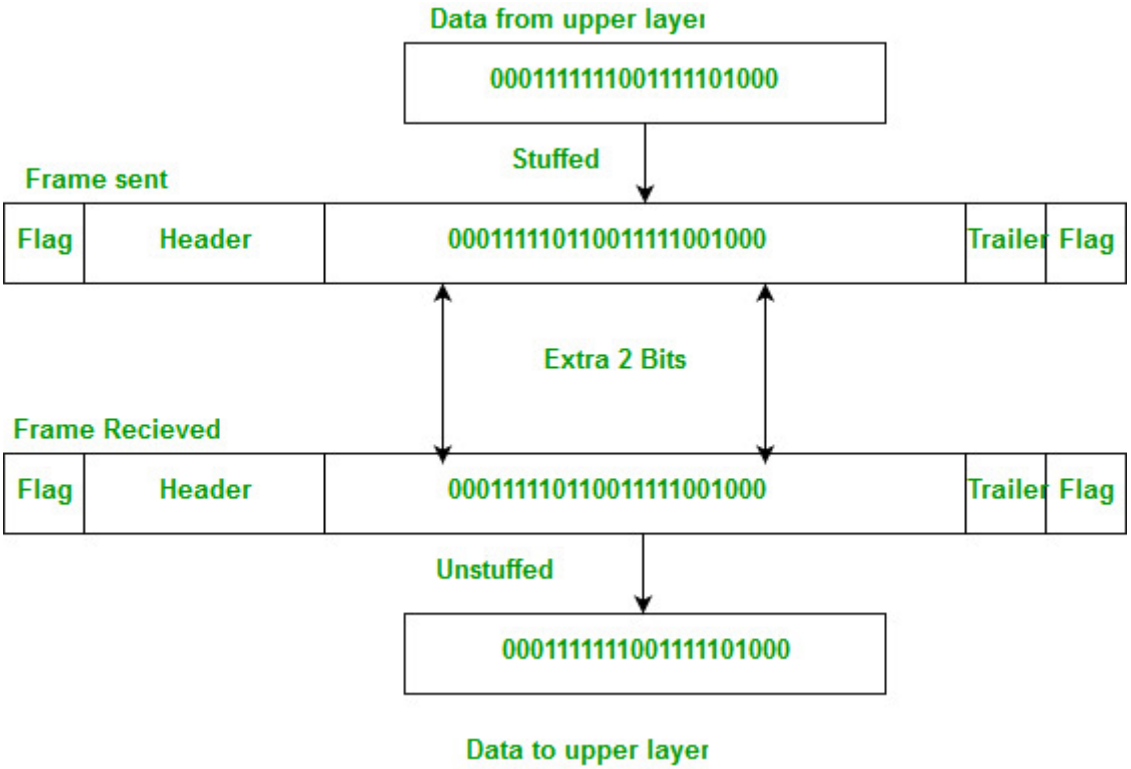
Example:



***Note** – Point-to-Point Protocol (PPP) is a byte-oriented protocol.*

Bit stuffing – Mostly flag is a special 8-bit pattern “01111110” used to define the beginning and the end of the frame. Problem with the flag is the same as that was in the case of byte stuffing. So, in this protocol what we do is, if we encounter 0 and five consecutive 1 bits, an extra 0 is added after these bits. This extra stuffed bit is removed from the data by the receiver. The extra bit is added after one 0 followed by five 1 bits regardless of the value of the next bit. Also, as the sender side always knows which sequence is data and which is flag it will only add this extra bit in the data sequence, not in the flag sequence.

Example:



Note: *High-Level Data Link Control(HDLC) is a bit-oriented protocol.*

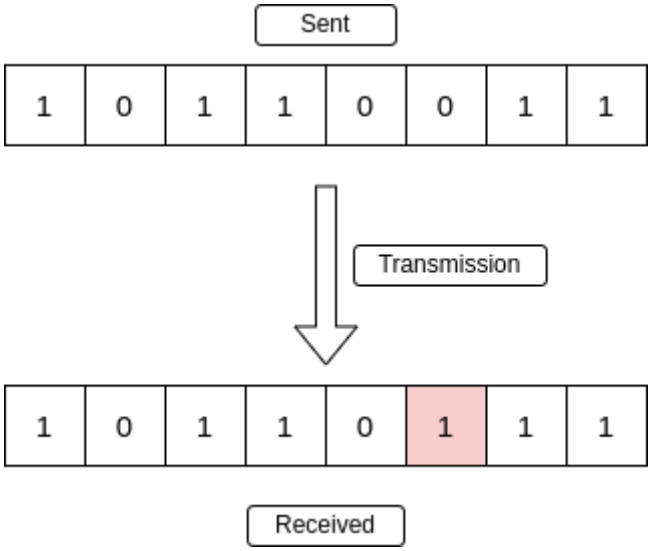
5. Explain the following error detection techniques with example
i) LRC ii) CRC iii) Checksum [7M] **[Dec/Jan-2022-23]**

Error is a condition when the receiver's information does not match the sender's information. During transmission, digital signals suffer from noise that can introduce errors in the binary bits traveling from sender to receiver. That means a 0 bit may change to 1 or a 1 bit may change to 0.
Data (Implemented either at the Data link layer or Transport Layer of the OSI Model) may get scrambled by noise or get corrupted whenever a message is transmitted. To prevent such errors, error-detection codes are added as extra data to digital messages. This helps in detecting any errors that may have occurred during message transmission.

Types of Errors

Single-Bit Error

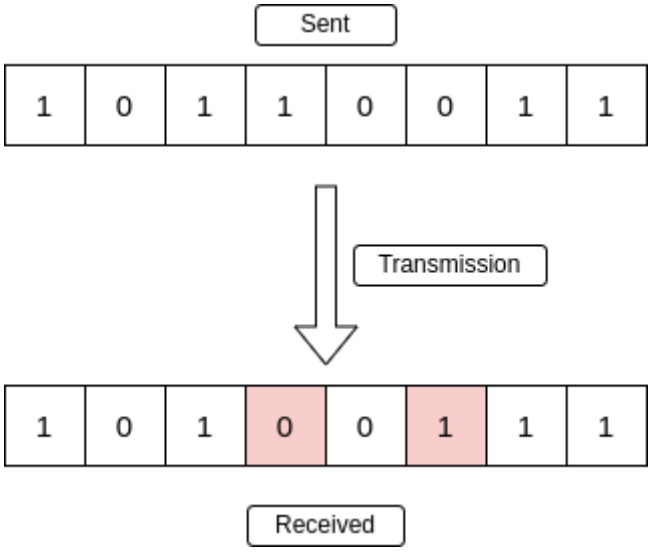
A single-bit error refers to a type of data transmission error that occurs when one bit (i.e., a single binary digit) of a transmitted data unit is altered during transmission, resulting in an incorrect or corrupted data unit.



Single-Bit Error

Multiple-Bit Error

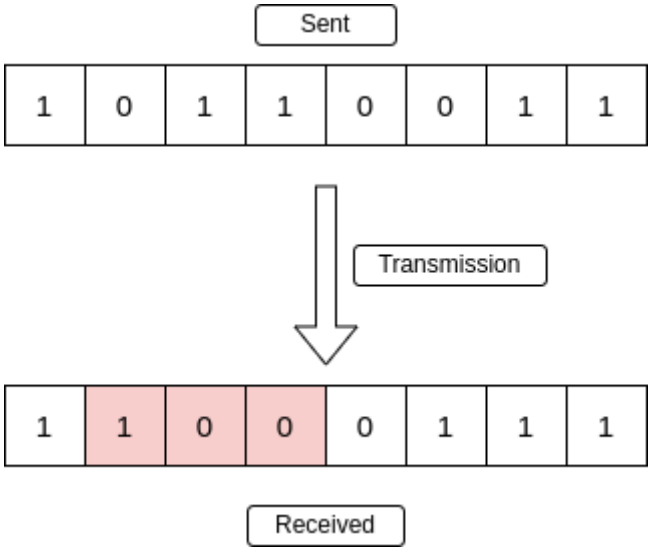
A multiple-bit error is an error type that arises when more than one bit in a data transmission is affected. Although multiple-bit errors are relatively rare when compared to single-bit errors, they can still occur, particularly in high-noise or high-interference digital environments.



Multiple-Bit Error

Burst Error

When several consecutive bits are flipped mistakenly in digital transmission, it creates a burst error. This error causes a sequence of consecutive incorrect values.



Burst Error

To detect errors, a common technique is to introduce redundancy bits that provide additional information. Various techniques for error detection include:

i) LRC
ii) CRC
iii) Checksum

LRC (Longitudinal Redundancy Check): is also known as 2-D parity check. In this method, data which the user want to send is organised into tables of rows and columns. A block of bit is divided into table or matrix of rows and columns. In order to detect an error, a redundant bit is added to the whole block and this block is transmitted to receiver. The receiver uses this redundant row to detect error. After checking the data for errors, receiver accepts the data and discards the redundant row of bits.

Example :
If a block of 32 bits is to be transmitted, it is divided into matrix of four rows and eight columns which as shown in the following figure :

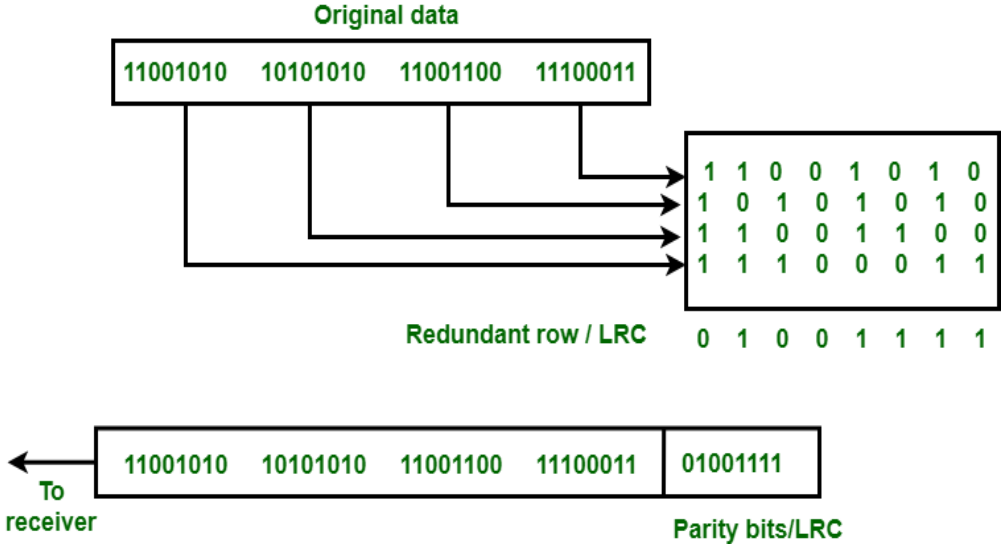


Figure: *LRC*

In this matrix of bits, a parity bit (odd or even) is calculated for each column. It means 32 bits data plus 8 redundant bits are transmitted to receiver. Whenever data reaches at the destination, receiver uses LRC to detect error in data.

Example : Suppose 32 bit data plus LRC that was being transmitted is hit by a burst error of length 5 and some bits are corrupted as shown in the following figure :

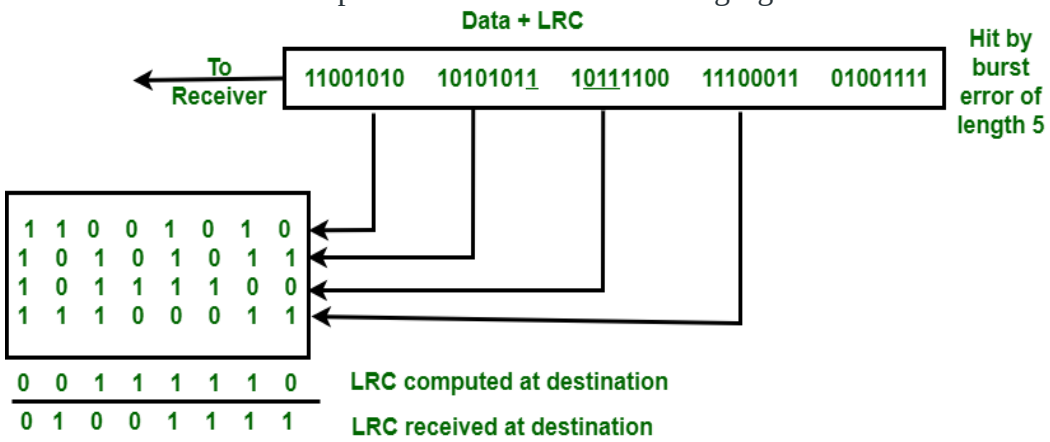


Figure : *Burst error & LRC*

The LRC received by the destination does not match with newly corrupted LRC. The destination comes to know that the data is erroneous, so it discards the data.

Advantage :
LRC is used to detect burst errors.

Disadvantage :
The main problem with LRC is that, it is not able to detect error if two bits in a data unit are damaged and two bits in exactly the same position in other data unit are also damaged.

Checksum

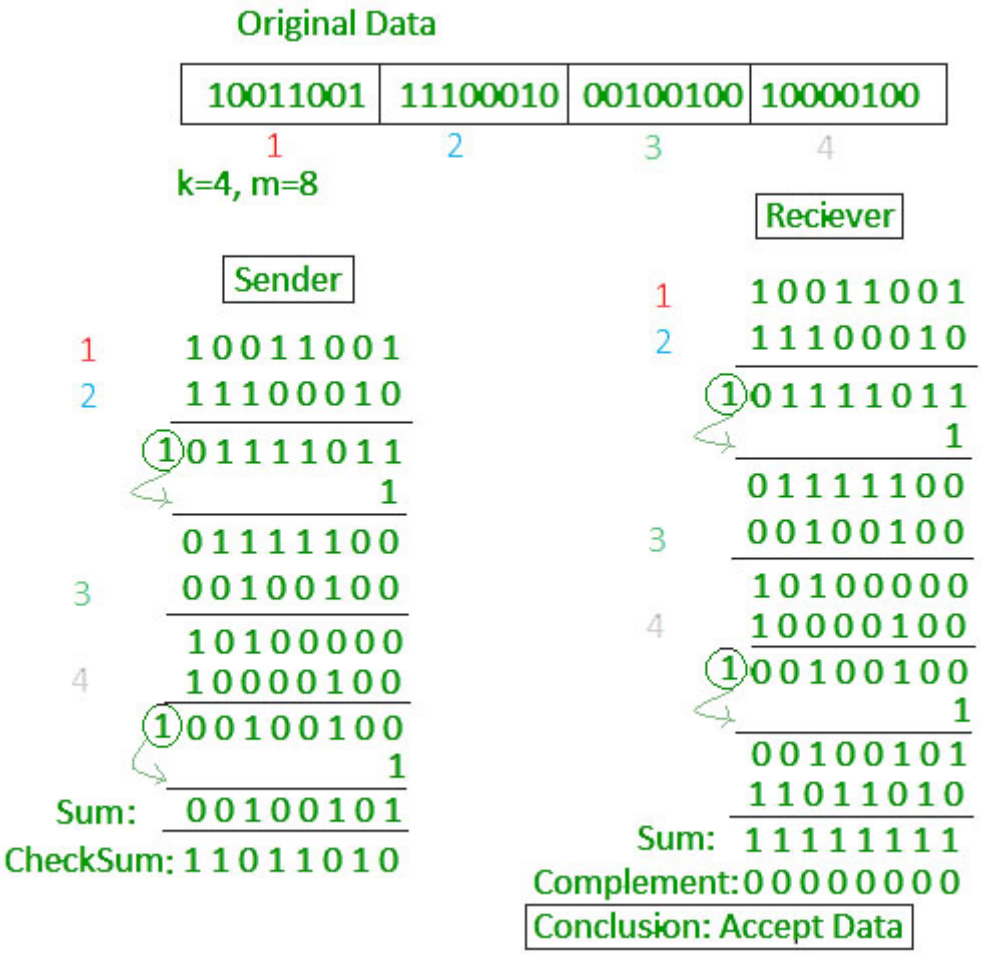
Checksum error detection is a method used to identify errors in transmitted data. The process involves dividing the data into equally sized segments and using a 1's complement to calculate the sum of these segments. The calculated sum is then sent along with the data to the receiver. At the receiver's end, the same process is repeated and if all zeroes are obtained in the sum, it means that the data is correct.

Checksum – Operation at Sender's Side

- Firstly, the data is divided into k segments each of m bits.
- On the sender's end, the segments are added using 1's complement arithmetic to get the sum. The sum is complemented to get the checksum.
- The checksum segment is sent along with the data segments.

Checksum – Operation at Receiver's Side

- At the receiver's end, all received segments are added using 1's complement arithmetic to get the sum. The sum is complemented.
- If the result is zero, the received data is accepted; otherwise discarded.

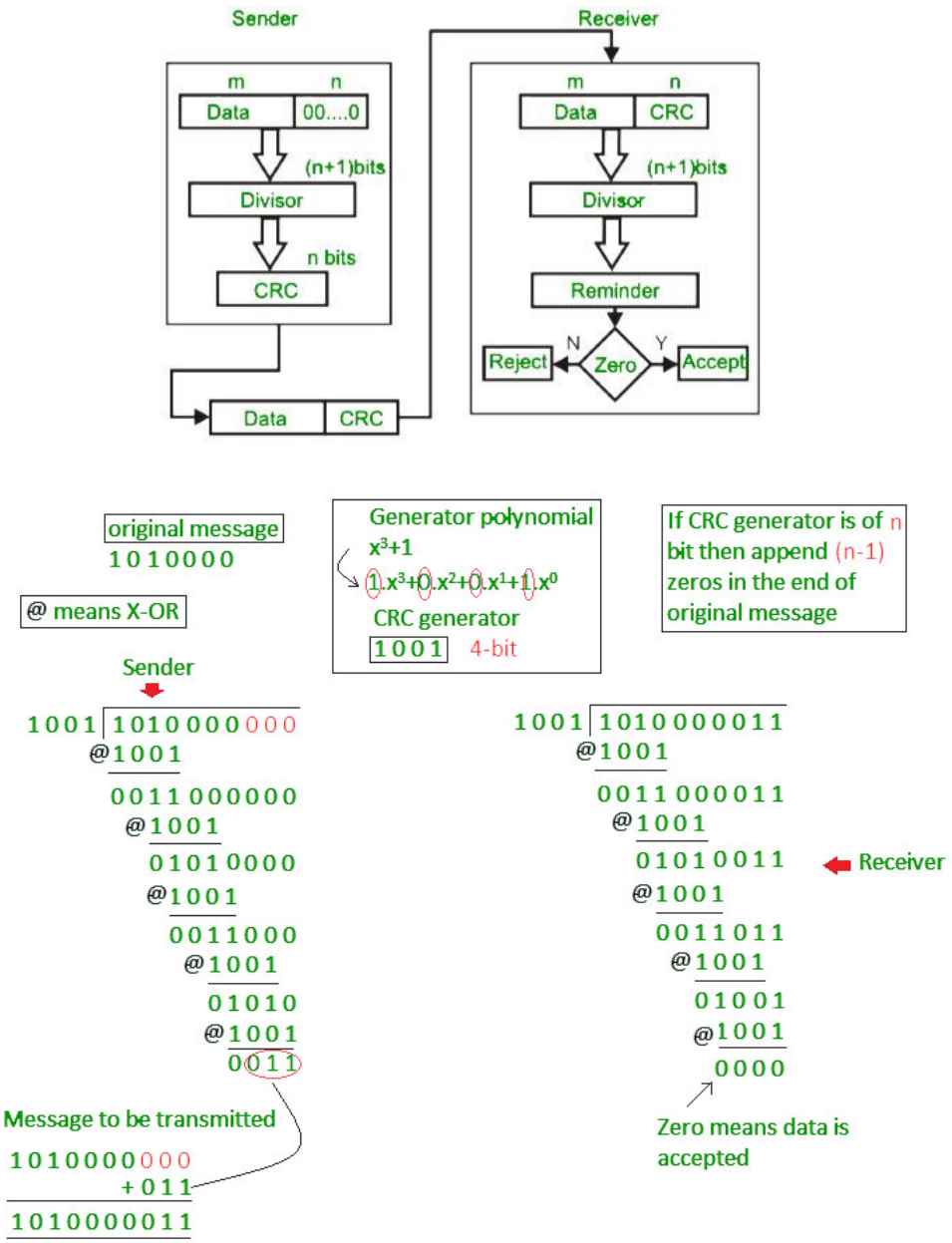


Disadvantages

- If one or more bits of a segment are damaged and the corresponding bit or bits of opposite value in a second segment are also damaged.

Cyclic Redundancy Check (CRC)

- Unlike the checksum scheme, which is based on addition, CRC is based on binary division.
- In CRC, a sequence of redundant bits, called cyclic redundancy check bits, are appended to the end of the data unit so that the resulting data unit becomes exactly divisible by a second, predetermined binary number.
- At the destination, the incoming data unit is divided by the same number. If at this step there is no remainder, the data unit is assumed to be correct and is therefore accepted.
- A remainder indicates that the data unit has been damaged in transit and therefore must be rejected.



Advantages :

Increased Data Reliability: Error detection ensures that the data transmitted over the network is reliable, accurate, and free from errors. This ensures that the recipient receives the same data that was transmitted by the sender.
Improved Network Performance: Error detection mechanisms can help to identify and isolate network issues that are causing errors. This can help to improve the overall performance of the network and reduce downtime.
Enhanced Data Security: Error detection can also help to ensure that the data transmitted over the network is secure and has not been tampered with.

Disadvantages:
Overhead: Error detection requires additional resources and processing power, which can lead to increased overhead on the network. This can result in slower network performance and increased latency.
False Positives: Error detection mechanisms can sometimes generate false positives, which can result in unnecessary retransmission of data. This can further increase the overhead on the network.
Limited Error Correction: Error detection can only identify errors but cannot correct them. This means that the recipient must rely on the sender to retransmit the data, which can lead to further delays and increased network overhead.

6.What is meant by error in data link layer? Discuss about error detection and correction methods in data link layer, [7M] **[Dec/Jan-2022-23]**

Data-link layer uses error control techniques to ensure that frames, i.e. bit streams of data, are transmitted from the source to the destination with a certain extent of accuracy.

Errors
When bits are transmitted over the computer network, they are subject to get corrupted due to interference and network problems. The corrupted bits leads to spurious data being received by the destination and are called errors.

Types of errors :

Errors can be of three types, namely single bit errors, multiple bit errors, and burst errors.

- **Single bit error** − In the received frame, only one bit has been corrupted, i.e. either changed from 0 to 1 or from 1 to 0.
- **Multiple bits error** − In the received frame, more than one bits are corrupted.
- **Burst error** − In the received frame, more than one consecutive bits are corrupted.

Error Control
Error control can be done in two ways

- **Error detection** − Error detection involves checking whether any error has occurred or not. The number of error bits and the type of error does not matter.
- **Error correction** − Error correction involves ascertaining the exact number of bits that has been corrupted and the location of the corrupted bits.

For both error detection and error correction, the sender needs to send some additional bits along with the data bits. The receiver performs necessary checks based upon the additional redundant bits. If it finds that the data is free from errors, it removes the redundant bits before passing the message to the upper layers.

Error Detection Techniques
There are three main techniques for detecting errors in frames: Parity Check, Checksum and Cyclic Redundancy Check (CRC).

Parity Check
The parity check is done by adding an extra bit, called parity bit to the data to make a number of 1s either even in case of even parity or odd in case of odd parity.

While creating a frame, the sender counts the number of 1s in it and adds the parity bit in the following way

- In case of even parity: If a number of 1s is even then parity bit value is 0. If the number of 1s is odd then parity bit value is 1.
- In case of odd parity: If a number of 1s is odd then parity bit value is 0. If a number of 1s is even then parity bit value is 1.

On receiving a frame, the receiver counts the number of 1s in it. In case of even parity check, if the count of 1s is even, the frame is accepted, otherwise, it is rejected. A similar rule is adopted for odd parity check.

The parity check is suitable for single bit error detection only.

Checksum

In this error detection scheme, the following procedure is applied

- Data is divided into fixed sized frames or segments.
- 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.
- 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.

Cyclic Redundancy Check (CRC)

Cyclic Redundancy Check (CRC) involves binary division of the data bits being sent by a predetermined divisor agreed upon by the communicating system. The divisor is generated using polynomials.

- Here, the sender performs binary division of the data segment by the divisor. It then appends the remainder called CRC bits to the end of the data segment. This makes the resulting data unit exactly divisible by the divisor.
- The receiver divides the incoming data unit by the divisor. If there is no remainder, the data unit is assumed to be correct and is accepted. Otherwise, it is understood that the data is corrupted and is therefore rejected.

Error Correction Techniques

Error correction techniques find out the exact number of bits that have been corrupted and as well as their locations. There are two principle ways

- **Backward Error Correction (Retransmission)** − If the receiver detects an error in the incoming frame, it requests the sender to retransmit the frame. It is a relatively simple technique. But it can be efficiently used only where retransmitting is not expensive as in fiber optics and the time for retransmission is low relative to the requirements of the application.
- **Forward Error Correction** − If the receiver detects some error in the incoming frame, it executes error-correcting code that generates the actual frame. This saves bandwidth required for retransmission. It is inevitable in real-time systems. However, if there are too many errors, the frames need to be retransmitted.

The four main error correction codes are :

- Hamming Codes
- Binary Convolution Code

- Reed – Solomon Code
- Low-Density Parity-Check Code

7. Explain the CRC error detection technique using generation polynomial x^4+x^3+1 data is 11100011. [7M] **[Dec/Jan-2022-23] Understand CO3**

Cyclic Redundancy Check (CRC) is an error-detection technique commonly used in data communication to detect errors in transmitted data. It involves appending a CRC code to the data, which is generated using a specific polynomial. When the data is received, the receiver also calculates the CRC code using the same polynomial and checks if it matches the received CRC code. If they match, it's likely that the data was transmitted without errors. If they don't match, it indicates that an error occurred during transmission.

In your example, you have the following information:

1. Data: 11100011
2. Generation Polynomial: x^4 + x^3 + 1

To calculate the CRC code for the given data, you can follow these steps:

Step 1: Append Zeros
Append zeros to the data to make its length equal to the degree of the polynomial minus one. In this case, the polynomial is of degree 4, so you need to append four zeros to the data:

Data: 111000110000

Step 2: Perform Binary Polynomial Division
Perform binary polynomial division using XOR operations. Divide the modified data by the generation polynomial, keeping track of the remainder. The remainder is the CRC code. Here's how the division works:

```
          11010 (generation polynomial)
        ____________________
111000110000 | 111000110000
          - 11010
           ____________
             1010110000
             -  11010
              _________
                10010000
                -  11010
                 _______
                  1001010
```

```
    - 11010
    _________
      110000
      - 11010
      _______
        00110
```

The remainder after the division is 00110. This is the CRC code.

Step 3: Append the CRC Code to the Data
Append the CRC code (00110) to the original data:

Data + CRC Code: 1110001100110

Now, you can transmit this entire sequence (data + CRC code) to the receiver.

Step 4: Verification at the Receiver
When the receiver receives the data, it performs the same polynomial division using the received data and the same generation polynomial:

```
          11010 (generation polynomial)
        _____________________
1110001100110 | 1110001100110
          - 11010
          ____________
            1010110000
            - 11010
            _________
              10010000
              - 11010
              _______
                1001010
                - 11010
                _______
                  110000
                  - 11010
                  _______
                    00110
```

The remainder at the receiver is also 00110.

Step 5: Check for Errors

If the remainder calculated by the receiver matches the received CRC code (00110), it indicates that there were no errors in transmission. If they don't match, it suggests that an error occurred during data transmission.

In this case, since the remainders match, the receiver can conclude that the data was received without errors.

8. A bit stream 10011101 is transmitted using the standard CRC method. The generator polynomial is x3+1. Show the actual bit string transmitted. Suppose the third bit from the left is inverted during transmission. Show that this error is detected at the receiver end. [7M] **[Dec/Jan-2022-23]**

To determine the actual bit string transmitted and demonstrate error detection using CRC, we'll follow these steps:

1. Initial Message: The given bit stream is 10011101, and the generator polynomial is x^3 + 1.

2. Encoding with CRC:

- Append 3 zeros (since the degree of the generator polynomial is 3) to the end of the message: 10011101000.

- Perform polynomial division to calculate the CRC code. Here's a step-by-step calculation:

```
  1010011010   <-- Dividend (Message + Padding)
/ 1101         <-- Divisor (Generator Polynomial x^3 + 1)
 ------------
  1010
   1010
    ----
      0000
```

- The remainder of the division (0000) is the CRC code.

3. Transmitted Message: The actual bit string transmitted is the concatenation of the original message and the CRC code. So, it is 10011101000.

4. Error During Transmission: Suppose the third bit from the left (counting from 1) is inverted during transmission. This means that the bit at position 3 (which is '0' in the original message) is changed to '1'. The transmitted message becomes 10111101000.

5. Error Detection at Receiver End:

- The receiver receives the message 10111101000.

- The receiver performs the same CRC division:

```
  1011110100   <-- Received Message
/ 1101         <-- Divisor (Generator Polynomial x^3 + 1)
 ------------
  1110
   1101
    ----
     100
```

- Since there is a non-zero remainder (100), this indicates an error in the received message.

- The receiver knows that if the remainder is non-zero after CRC division, an error has occurred during transmission. This is how CRC detects errors.

In this case, the error inverting the third bit was successfully detected at the receiver end because the remainder after CRC division was non-zero.

9. A sender sends series of frames to the same destination using 5-bit sequence number. If the sequence number starts with 0, what is the sequence number after sending 100 frames?
[7M] **[Jun/Jul-2022]**

If the sender is using a 5-bit sequence number starting with 0 and sends 100 frames, the sequence number will wrap around after reaching its maximum value.

Here's how you can calculate the sequence number after sending 100 frames:

1. With 5 bits, you can represent sequence numbers from 00000 (0 in decimal) to 11111 (31 in decimal).

2. After sending 100 frames, you need to find the remainder when dividing 100 by 32 (2^5, as there are 5 bits available). This remainder will tell you how many times the sequence number has cycled through:

100 % 32 = 4

3. Now, add this remainder to the starting sequence number (0):

0 + 4 = 4

So, after sending 100 frames, the sequence number will be 4.

10. A bit stream 10011101 is transmitted using CRC method .The generator polynomial is X 3+1. Show that the actual bit transmitted. Suppose that third bit from left is inverted during the transmission. Show that this error is detected at the receiver's end.

[7M] **[Apr/May-2019] Evaluate CO3**

To determine the actual bit string transmitted and demonstrate error detection using CRC, we'll follow these steps:

1. Initial Message: The given bit stream is 10011101, and the generator polynomial is X^3 +1.

2. Encoding with CRC:

- Append 3 zeros (since the degree of the generator polynomial is 3) to the end of the message: 10011101000.

- Perform polynomial division to calculate the CRC code. Here's a step-by-step calculation:

```
  10011101000   <-- Dividend (Message + Padding)
/ 1101          <-- Divisor (Generator Polynomial X^3 + 1)
  ------------
  1001
    1101
    ----
      1010
```

- The remainder of the division (1010) is the CRC code.

3. Transmitted Message: The actual bit string transmitted is the concatenation of the original message and the CRC code. So, it is 100111010001010.

4. Error During Transmission: Suppose the third bit from the left (counting from 1) is inverted during transmission. This means that the bit at position 3 (which is '0' in the original message) is changed to '1'. The transmitted message becomes 101111010001010.

5. Error Detection at Receiver End:

- The receiver receives the message 101111010001010.

- The receiver performs the same CRC division:

```
    101111010001010   <-- Received Message
/ 1101                <-- Divisor (Generator Polynomial X^3 + 1)
  ----------------
  1101
    1101
    ----
```

0000

- Since there is a zero remainder, this indicates that no error has occurred during transmission.

In this case, the error inverting the third bit was not detected at the receiver end because the remainder after CRC division was zero.

11. Consider the following data: 1100111 1011101 0111001 0101001 Find Row and Column parities using even parity. Show how the following errors can be detected:

(i) An error at (R3,C3) (ii) Two errors at (R3,C4) and (R3,C6)

(iii) Three errors at (R2,C4), (R2,C5) and (R3,C4)

Explain error correction. When sender transmits the data to the receiver. Data be m = 100110 and find out the 6th position of an error between the transmission.

[7M] **[Dec/Jan-2022-23]**

To detect errors using row and column parities with even parity, we first calculate both the row and column parities for the given data matrix.

Given data matrix:
1100111
1011101
0111001
0101001

Calculating Row Parities (Horizontal):

For each row, count the number of '1' bits and add an extra bit to make the total even.

1. Row 1: 1100111 -> 1 1001110 (even parity added)
2. Row 2: 1011101 -> 10 11101 (even parity added)
3. Row 3: 0111001 -> 011 10010 (even parity added)
4. Row 4: 0101001 -> 0101 0010 (even parity added)

Calculating Column Parities (Vertical):

For each column, count the number of '1' bits and add an extra bit to make the total even.

1. Column 1: 1 10 01 01
2. Column 2: 1 01 11 01
3. Column 3: 0 11 11 00
4. Column 4: 0 00 00 01
5. Column 5: 1 11 00 01
6. Column 6: 0 01 01 01
7. Column 7: 0 01 00 00

Now let's address the three error scenarios:

(i) An error at (R3, C3):
If there's an error at (R3, C3), it means the bit in the third row and third column is flipped. This will cause both the row parity for Row 3 and the column parity for Column 3 to become odd, indicating an error.

(ii) Two errors at (R3, C4) and (R3, C6):
If there are errors at (R3, C4) and (R3, C6), these will affect both the row parity for Row 3 and the column parity for Columns 4 and 6. Since these parities will become odd, this combination of errors will also be detected.

(iii) Three errors at (R2, C4), (R2, C5), and (R3, C4):
If there are three errors at these positions, it will affect row parities for Row 2 and Row 3, as well as column parity for Column 4. All of these parities will become odd, indicating the presence of multiple errors.

Error Correction:
Even parity can detect errors, but it cannot correct them. It can only indicate that errors have occurred. To correct errors, you would need additional information, such as error-correcting codes (ECCs) or redundancy in the data.

Finding the 6th Position Error in "100110":
To find the 6th position error in "100110," we need to calculate the even parity for this 6-bit data:

1. "100110" -> 1001100 (even parity added)

Now, calculate the parity bit for this 7-bit data:

- Count the number of '1' bits in "1001100" -> There are 3 '1's.
- Add an extra bit to make the total even -> "11001100."

The presence of an error will be indicated if this 7-bit data has an odd number of '1's. In this case, with 3 '1's, it's even, indicating no error in the 6th position.

12. Explain flow control mechanism in data link layer.

[7M][Reg Feb 2022 Set-1]

Flow control in the data link layer is a mechanism used to manage the rate of data transmission between two devices to prevent data loss and congestion. It ensures that data is sent at a rate that the receiving device can handle without being overwhelmed. Flow control mechanisms work together with error detection and correction to ensure reliable data transfer. Two common flow control methods are:

1. Stop-and-Wait Flow Control: In this method, the sender sends a single data frame and then waits for an acknowledgment (ACK) from the receiver before sending the next frame. If the receiver successfully receives the frame, it sends an ACK; otherwise, it requests the sender to retransmit the frame. This approach ensures that the sender doesn't overwhelm the receiver.

2. Sliding Window Flow Control: This method allows the sender to transmit multiple frames before waiting for acknowledgments. The receiver maintains a "window" indicating the range of acceptable sequence numbers. As long as the sender's sequence numbers fall within this window, it can send frames. The receiver acknowledges received frames and updates the window. Sliding window flow control increases the efficiency of data transfer by reducing the need for frequent acknowledgments.

Both of these flow control methods help ensure efficient and reliable data transmission at the data link layer without overloading the receiver or causing data loss due to congestion.

13. Discuss data link layer protocols for noiseless channels and noisy channels in detail.
[**7M**] **[Dec/Jan-2022-23]**

Data Link Layer Protocols for Noiseless Channels:

1. Stop-and-Wait ARQ (Automatic Repeat reQuest):
- Operation: In a noiseless channel, Stop-and-Wait ARQ is a simple protocol where the sender sends one frame and waits for an acknowledgment (ACK) from the receiver. If the receiver successfully receives the frame, it sends an ACK. If the frame is lost or corrupted, the sender retransmits it.
- Efficiency: It's straightforward but not very efficient because it doesn't take advantage of the full channel capacity.

2. Go-Back-N ARQ:
- Operation: This protocol allows the sender to send multiple frames (a window of frames) before waiting for acknowledgments. The receiver acknowledges the frames it receives successfully. If a frame is lost or corrupted, the receiver discards it and all subsequent frames until the expected one arrives. The sender retransmits the lost frame and subsequent frames.
- Efficiency: It's more efficient than Stop-and-Wait ARQ as it utilizes the channel better.

3. Selective Repeat ARQ:
- Operation: Similar to Go-Back-N, this protocol uses a window of frames. However, unlike Go-Back-N, the receiver individually acknowledges each frame it receives, even if they arrive out of order. The sender only retransmits frames that were not acknowledged.
- Efficiency: It's more efficient and reduces unnecessary retransmissions compared to Go-Back-N in noiseless channels.

Data Link Layer Protocols for Noisy Channels:

1. Error Detection and Correction Codes:

- Operation: These protocols involve the use of error-detecting and error-correcting codes such as CRC (Cyclic Redundancy Check) and Hamming codes. These codes add extra bits to the data to detect and, in some cases, correct errors. If an error is detected, the receiver can request retransmission.

- Efficiency: They are effective in noisy channels but add some overhead due to the extra bits.

2. Selective Reject ARQ:

- Operation: In a noisy channel, Selective Reject ARQ is similar to Selective Repeat ARQ. The difference is that the receiver acknowledges all received frames but requests retransmission only for the frames with errors. This reduces unnecessary retransmissions.

- Efficiency: It is efficient in noisy channels as it minimizes retransmissions.

3. Automatic Repeat reQuest (ARQ):

- Operation: In noisy channels, ARQ protocols like Go-Back-N can still be used, but they require the sender to retransmit frames even if only one is in error. This is less efficient but works in noisy conditions.

- Efficiency: While less efficient than selective schemes, it's still a viable option in noisy channels.

In noisy channels, the choice of protocol depends on the trade-off between efficiency and complexity. Protocols like Selective Repeat ARQ and error correction codes are more efficient but come with added complexity, while simpler protocols like Go-Back-N ARQ may be used when efficiency is not the primary concern. The choice also depends on factors like channel characteristics and the required reliability of the data link layer.

14. Explain the difference between flow control and error control.

[7M][June/July 2022 Set-1]

Difference between Flow Control and Error Control :

S.NO.	Flow control	Error control
1.	Flow control is meant only for the transmission of data from sender to receiver.	Error control is meant for the transmission of error free data from sender to receiver.

2.	For Flow control there are two approaches : Feedback-based Flow Control and Rate-based Flow Control.	To detect error in data, the approaches are : Checksum, Cyclic Redundancy Check and Parity Checking. To correct error in data, the approaches are : Hamming code, Binary Convolution codes, Reed-Solomon code, Low-Density Parity Check codes.
3.	It prevents the loss of data and avoid over running of receive buffers.	It is used to detect and correct the error occurred in the code.
4.	Example of Flow Control techniques are : Stop & Wait Protocol and Sliding Window Protocol.	Example of Error Control techniques are : Stop & Wait ARQ and Sliding Window ARQ (Go-back-N ARQ, Selected Repeat ARQ).

15. Infer the need of buffers on both sender and receiver ends.

[7M][July 2023 Set-1]

Buffers are essential on both sender and receiver ends of a communication system for several reasons:

1. **Data Rate Mismatch:** The sender and receiver may operate at different speeds. Buffers allow the sender to continue transmitting data even if the receiver is temporarily slower, preventing data loss.

2. **Varying Processing Times:** Both sender and receiver devices might need time to process incoming and outgoing data. Buffers temporarily store data, ensuring it's not lost during processing delays.

3. **Network Congestion:** In network communication, data can encounter congestion or delays in transit. Buffers help in managing these delays and ensure that data arrives intact and in the correct order.

4. **Error Handling:** Buffers provide a space to temporarily hold data that might need retransmission due to errors. This helps maintain data integrity.

5. **Asynchronous Operation:** In cases where sender and receiver operate independently or asynchronously, buffers act as a bridge between them, allowing data to be stored until the other end is ready to receive or process it.

6. **Smooth Data Flow:** Buffers can smooth out variations in data transmission rates, preventing abrupt starts and stops, which can be disruptive to the communication process.

7. **Efficiency:** Buffers can improve system efficiency by allowing data to be transmitted and processed in larger, more efficient chunks rather than small, frequent transfers.

In summary, buffers are crucial components in data communication systems, ensuring the efficient and reliable transfer of data between sender and receiver by accommodating variations in speed, processing, and network conditions.

16. Explain in detail about Point-to-Point Protocol.

[7M] **[Dec/Jan-2022-23]**

Point - to - Point Protocol (PPP) is a communication protocol of the data link layer that is used to transmit multiprotocol data between two directly connected (point-to-point) computers. It is a byte - oriented protocol that is widely used in broadband communications having heavy loads and high speeds. Since it is a data link layer protocol, data is transmitted in frames. It is also known as RFC 1661.

Services Provided by PPP
The main services provided by Point - to - Point Protocol are −

Defining the frame format of the data to be transmitted.

Defining the procedure of establishing link between two points and exchange of data.

Stating the method of encapsulation of network layer data in the frame.

Stating authentication rules of the communicating devices.

Providing address for network communication.

Providing connections over multiple links.

Supporting a variety of network layer protocols by providing a range os services.

Components of PPP

Point - to - Point Protocol is a layered protocol having three components −

Encapsulation Component − It encapsulates the datagram so that it can be transmitted over the specified physical layer.

Link Control Protocol (LCP) − It is responsible for establishing, configuring, testing, maintaining and terminating links for transmission. It also imparts negotiation for set up of options and use of features by the two endpoints of the links.

Authentication Protocols (AP) − These protocols authenticate endpoints for use of services. The two authentication protocols of PPP are −

Password Authentication Protocol (PAP)

Challenge Handshake Authentication Protocol (CHAP)

Network Control Protocols (NCPs) − These protocols are used for negotiating the parameters and facilities for the network layer. For every higher-layer protocol supported by PPP, one NCP is there. Some of the NCPs of PPP are −

Internet Protocol Control Protocol (IPCP)

OSI Network Layer Control Protocol (OSINLCP)

Internetwork Packet Exchange Control Protocol (IPXCP)

DECnet Phase IV Control Protocol (DNCP)

NetBIOS Frames Control Protocol (NBFCP)

IPv6 Control Protocol (IPV6CP)

PPP Frame

PPP is a byte - oriented protocol where each field of the frame is composed of one or more bytes. The fields of a PPP frame are −

- **Flag** − 1 byte that marks the beginning and the end of the frame. The bit pattern of the flag is 01111110.
- **Address** − 1 byte which is set to 11111111 in case of broadcast.
- **Control** − 1 byte set to a constant value of 11000000.
- **Protocol** − 1 or 2 bytes that define the type of data contained in the payload field.
- **Payload** − This carries the data from the network layer. The maximum length of the payload field is 1500 bytes. However, this may be negotiated between the endpoints of communication.
- **FCS** − It is a 2 byte or 4 bytes frame check sequence for error detection. The standard code used is CRC (cyclic redundancy code)

Byte Stuffing in PPP Frame − Byte stuffing is used is PPP payload field whenever the flag sequence appears in the message, so that the receiver does not consider it as the end of the frame. The escape byte, 01111101, is stuffed before every byte that contains the same byte as the flag byte or the escape byte. The receiver on receiving the message removes the escape byte before passing it onto the network layer.

17. Explain the elementary datalink layer protocols.
    i) Simplex protocol ii) simplex stop and wait iii) simplex protocol for noisy channel [7M] [**Apr/May-2019]**

Elementary data link layer protocols are basic communication protocols that operate at the data link layer (Layer 2) of the OSI model. They are often used for educational purposes to understand fundamental concepts in data communication. Let's explain the three mentioned protocols:

****i) Simplex Protocol:****
- Simplex communication is the most basic form of communication, where data flows in only one direction, like a one-way street.
- In a simplex protocol, data is transmitted from a sender to a receiver, but there's no feedback from the receiver to the sender.
- It's similar to a radio or television broadcast, where the sender (e.g., a TV station) transmits data, but the receiver (e.g., a TV set) doesn't send any data back.
- Simplex is suitable for applications where one-way communication suffices, and there's no need for acknowledgment or error correction.

****ii) Simplex Stop-and-Wait Protocol:****
- The Simplex Stop-and-Wait protocol is an extension of the simplex protocol that introduces flow control and acknowledgment.
- In this protocol, the sender sends a data frame to the receiver and waits for an acknowledgment (ACK).

- The receiver acknowledges the received frame by sending an ACK back to the sender. If the frame is corrupted or lost, the sender waits for a timeout before retransmitting the frame.
- It ensures that data is reliably delivered, even over unreliable channels, by using acknowledgments and retransmissions.

****iii) Simplex Protocol for Noisy Channel:****
- This variant of the simplex protocol is designed for communication over noisy channels, where data is prone to errors.
- It incorporates error detection and correction mechanisms to increase the reliability of data transmission.
- Typically, it includes techniques like adding error-checking codes (e.g., CRC or checksum) to data frames and having the receiver check these codes for errors.
- If an error is detected, the receiver may request the sender to retransmit the frame.
- This protocol is suitable for situations where data integrity is critical, and the channel is error-prone.

In summary, elementary data link layer protocols like simplex, simplex stop-and-wait, and simplex for noisy channels provide varying levels of communication capabilities. Simplex is unidirectional, simplex stop-and-wait introduces reliability with acknowledgment, and simplex for noisy channels adds error detection and correction for use in noisy communication environments. These protocols help in understanding fundamental concepts of data communication and are building blocks for more complex data link layer protocols.

18. Explain the Sliding window flow control mechanisms. Explain the drawbacks of Stop and wait? How they overcome by Sliding window protocol.

[7M]**[Dec/Jan-2022-23]**

Sliding Window Flow Control Mechanisms:

Sliding window flow control mechanisms are used in data communication to manage the flow of data between a sender and a receiver efficiently. Unlike the simple Stop-and-Wait protocol, sliding window protocols allow multiple frames to be in transit simultaneously, improving data transfer efficiency. The sender maintains a "window" of frames that can be sent before waiting for acknowledgments.

Here's an explanation of sliding window flow control and how it overcomes the drawbacks of the Stop-and-Wait protocol:

****Drawbacks of Stop-and-Wait:****
1. ****Inefficiency:** Stop-and-Wait is inefficient because the sender can only transmit one frame at a time. This underutilizes available bandwidth, especially in high-speed networks.**

2. **Low Throughput: Due to the inefficiency, the protocol results in low throughput, making it impractical for high-speed connections.**

3. **Increased Latency: Each frame must be acknowledged before the next frame is sent, increasing latency in data transmission.**

****How Sliding Window Overcomes Stop-and-Wait Drawbacks:****

1. **Increased Throughput:**
- Sliding window protocols allow multiple frames to be in transit simultaneously, maximizing the utilization of the communication channel.
- The sender can send a window's worth of frames (e.g., 5 frames) before waiting for acknowledgments, greatly improving throughput.

2. **Reduced Latency:**
- Since the sender doesn't need to wait for each frame to be acknowledged before sending the next one, sliding window protocols reduce latency in data transmission.

3. **Efficient Use of Bandwidth:**
- Sliding window protocols adapt to the available bandwidth more efficiently, ensuring that the channel is used optimally, even in high-speed networks.

4. **Selective Retransmission:**
- Sliding window protocols allow for selective retransmission of frames that are lost or corrupted. Only the affected frames need to be retransmitted, improving efficiency.

5. **Flow Control:**
- Sliding window protocols incorporate flow control mechanisms to prevent sender and receiver buffer overflows.
- The receiver advertises its buffer space to the sender, and the sender ensures not to exceed this capacity, preventing data loss.

6. **Error Handling:**
- Sliding window protocols often include error detection and correction mechanisms to ensure the integrity of transmitted data.

7. **Support for Bidirectional Communication:**
- Some sliding window protocols, like Go-Back-N and Selective Repeat, support bidirectional communication, allowing data to flow in both directions simultaneously.

In summary, sliding window flow control mechanisms, such as Go-Back-N and Selective Repeat, significantly improve data transfer efficiency and overcome the drawbacks of the Stop-and-Wait protocol. They achieve higher throughput, reduced latency, and efficient

bandwidth utilization by allowing multiple frames to be in transit simultaneously and providing mechanisms for error handling and flow control.

19. Describe about the Selective –Repeat protocol.

[7M] [**Jun/Jul-2022]**

Selective Repeat Protocol (SRP) is a type of error control protocol we use in computer networks to ensure the reliable delivery of data packets. Additionally, we use it in conjunction with the Transmission Control Protocol (TCP) to ensure that the receiver receives data transmitted over the network without errors.

In the SRP, the sender divides the data into packets and sends them to the receiver. Furthermore, the receiver sends an acknowledgment (ACK) for each packet received successfully. If the sender doesn't receive an ACK for a particular packet, it retransmits only that packet instead of the entire set of packets.

The SRP uses a window-based flow control mechanism to ensure the sender doesn't overwhelm the receiver with too many packets. Additionally, the sender and receiver maintain a window of packets. Based on the window size, the sender sends packets and waits for a specific amount of time for acknowledgment from the receiver.

The receiver, in turn, maintains a window of packets that contains the frame number it's receiving from the sender. If a frame is lost during transmission, the receiver sends the sender a negative acknowledgment attacking the frame number.

Steps :

The first step is to divide data into packets. The sender divides the data into packets of a fixed size. When the sender divides the data into packets, it assigns a unique sequence number to each packet. The numbering of packets plays a crucial role in the SRP.

The next step is to send the packets to the receiver. The receiver receives the packets and sends an acknowledgment (ACK) for each packet received successfully.

The sender and receiver maintain a window of packets indicating the number of frames we can transmit or receive at a given time. Additionally, we determine the size of the window based on the network conditions. As the sender sends packets, it updates its window to reflect the packets that have been transmitted, and the ACKs received.

However, if the sender doesn't receive an ACK for a particular packet within a certain timeout period, it retransmits only that packet instead of the entire set of packets. The receiver only accepts packets that are within its window. If the receiver receives a packet outside the window, it discards the packet.

The receiver sends selective acknowledgments (SACKs) for packets received out of order or lost. The sender processes the SACKs to determine which packets need to be retransmitted.

Finally, we continue this process until we successfully send the data packets or the number of retransmissions exceeds a predetermined threshold.

20. Explain the working of unrestricted simplex protocol, what are the restrictions placed on other protocols?

The Unrestricted Simplex Protocol is a basic communication protocol that allows data transmission in one direction only, from a sender to a receiver. Unlike full-duplex communication, where data can flow in both directions simultaneously, unrestricted simplex only permits data to be sent from one end to the other. Here's how it works:

Working of Unrestricted Simplex Protocol:

1. **Sender-Receiver Configuration:** There are two devices involved: a sender and a receiver. The sender is the device that initiates the communication and sends data, while the receiver is the device that receives the data.

2. **One-Way Data Flow:** In this protocol, data flows in one direction only, typically from the sender to the receiver. The sender can transmit data frames or packets to the receiver, but there's no provision for the receiver to send data back to the sender.

3. **No Acknowledgments:** Unlike more advanced protocols, such as Stop-and-Wait or Sliding Window, there are typically no acknowledgments or feedback from the receiver to the sender. The sender assumes that data sent is received correctly.

4. **No Error Handling:** Unrestricted simplex often lacks built-in error detection or correction mechanisms. If data is corrupted during transmission, there's no way to request retransmission or correct errors.

Restrictions Placed on Other Protocols:

Unrestricted simplex protocol has limited applicability due to its one-way data flow and lack of error handling and acknowledgment mechanisms. As a result, more advanced protocols are designed to overcome these restrictions and provide more robust and versatile communication. Here are some restrictions placed on other protocols:

1. **Stop-and-Wait Protocol:** Unlike unrestricted simplex, Stop-and-Wait introduces acknowledgments from the receiver to the sender, allowing for reliable data transmission. It also includes error detection and retransmission mechanisms.

2. **Sliding Window Protocols:** Sliding window protocols, such as Go-Back-N and Selective Repeat, address the limitations of unrestricted simplex by allowing multiple frames to be in transit simultaneously, reducing latency, and offering error detection and correction.

3. **Full-Duplex Protocols:** Full-duplex protocols, like Ethernet or TCP/IP, enable bidirectional communication, allowing data to flow in both directions simultaneously. They typically include acknowledgments, error handling, and flow control mechanisms.

4. **Error-Correction Protocols:** Some protocols, like Automatic Repeat reQuest (ARQ) or Forward Error Correction (FEC), focus specifically on detecting and correcting errors in data transmission, which unrestricted simplex lacks.

5. **Flow Control Protocols:** Modern communication protocols incorporate flow control mechanisms to manage the rate of data transfer, ensuring that sender and receiver buffers do not overflow.

In summary, unrestricted simplex protocol is a simple one-way communication method, but it lacks essential features for reliable, bidirectional, or error-tolerant communication. More advanced protocols are designed to overcome these restrictions by introducing acknowledgments, error handling, and flow control, enabling more versatile and robust data transmission.

21. Derive the sending and receiver window sizes for Go Back N and Selective-Repeat protocols. **[7M][June/July 2022]**

In Go-Back-N (GBN) and Selective Repeat (SR) protocols, both sender and receiver maintain a window to manage the flow of data. These protocols are used in data link layer for reliable data transfer in a noisy channel. Let's derive the sending and receiver window sizes for both protocols.

****1. Go-Back-N (GBN):****

In the Go-Back-N protocol:

- The sender maintains a sending window that can hold `N` frames.
- The receiver maintains a single, fixed-size receiving window that can hold `1` frame.
- `N` is the maximum number of frames that can be in transit (sent but not yet acknowledged) at any given time.

Sender Window Size (N):
- The sender can send up to `N` frames before needing to wait for acknowledgments.
- It's determined by the maximum sequence number allowed by the protocol.

Receiver Window Size (1):

- The receiver can only receive one frame at a time.
- The receiver can selectively acknowledge correctly received frames.

****2. Selective Repeat (SR):****

In the Selective Repeat protocol:

- Both the sender and receiver maintain a window.
- The sender maintains a sending window that can hold `N` frames.
- The receiver maintains a receiving window that can also hold `N` frames.
- `N` is the maximum number of frames that can be in transit (sent but not yet acknowledged) at any given time.

Sender Window Size (N):
- The sender can send up to `N` frames before needing to wait for acknowledgments.
- The sender can selectively retransmit only those frames that are not acknowledged.

Receiver Window Size (N):
- The receiver can receive and buffer `N` frames at a time.
- It can selectively acknowledge the frames that are received correctly, indicating the next expected frame.

22. Discuss about working Principle of a One-Bit Sliding Window Protocol with example. [7M] [**Jun/Jul-2022]**

A One-Bit Sliding Window Protocol, also known as the 1-bit sliding window protocol, is one of the simplest forms of sliding window protocols used for reliable data communication between a sender and a receiver. It allows the sender to transmit one frame at a time, and the receiver acknowledges the successful receipt of that frame. Here's how it works, along with an example:

****Working Principle:****

1. ****Sender Side:**
- The sender has a single-frame sending window.
- It sends one frame at a time to the receiver.
- After sending a frame, the sender waits for an acknowledgment (ACK) from the receiver.

2. ****Receiver Side:**
- The receiver also has a single-frame receiving window.
- It receives the frame from the sender.
- If the received frame is error-free, the receiver sends an ACK back to the sender.
- If the received frame has errors, the receiver discards it, and no ACK is sent.

3. **Acknowledgment (ACK):**
- The ACK is a positive acknowledgment that confirms the successful receipt of the frame.
- The sender uses the ACK to know that it can send the next frame.

4. **Timeout:**
- If the sender doesn't receive an ACK within a certain timeout period, it assumes the frame was lost or corrupted and retransmits the same frame.
- This timeout mechanism ensures reliable data transfer.

Example:

Let's illustrate the operation of the One-Bit Sliding Window Protocol with a simple example involving a sender (S) and a receiver (R).

1. Sender (S) sends Frame 0 to Receiver (R).
2. Receiver (R) successfully receives Frame 0 and sends an ACK0 back to Sender (S).
3. Sender (S) receives the ACK0 and knows that Frame 0 was successfully delivered.
4. Sender (S) can now send the next frame, which is Frame 1.
5. Receiver (R) successfully receives Frame 1 and sends an ACK1 back to Sender (S).
6. Sender (S) receives the ACK1 and knows that Frame 1 was successfully delivered.
7. Sender (S) can continue to send subsequent frames, one at a time, and wait for acknowledgments.

If an ACK is not received or is lost in transit, the sender will retransmit the frame after a timeout.

Note:
- The One-Bit Sliding Window Protocol ensures that frames are reliably transmitted one at a time.
- It may not fully utilize available bandwidth since it doesn't allow multiple frames in transit simultaneously, making it less efficient than more advanced sliding window protocols like Go-Back-N or Selective Repeat.
- However, it is simple and suitable for situations where the overhead of maintaining a larger window size is not justified.

23. Explain in detail about the sliding window protocol using Go-Back-NA.

[7M][June/July-2022]

The Go-Back-N (GBN) protocol is a sliding window protocol used in data communication to achieve reliable and efficient data transfer over a potentially noisy channel. It allows multiple frames to be in transit simultaneously and provides for automatic retransmission of lost or corrupted frames. Below is a detailed explanation of how the Go-Back-N protocol works:

Components of the GBN Protocol:

1. **Sender Window:** The sender maintains a sending window that can hold multiple frames. The sender can transmit frames within this window.

2. **Receiver Window:** The receiver also maintains a window, called the receiving window, that can hold multiple frames. The receiver acknowledges frames within this window.

3. **Sequence Numbers:** Each frame is assigned a unique sequence number by the sender. Sequence numbers help in tracking and ordering frames.

4. **Acknowledgment (ACK):** The receiver sends cumulative acknowledgments for frames it has successfully received and correctly sequenced up to a certain point.

Working of the Go-Back-N Protocol:

1. **Sender Side:**

   - The sender can send up to 'N' frames, where 'N' is the size of the sender's window.

   - Each frame is assigned a sequence number (e.g., 0, 1, 2, ...).

   - The sender continually transmits frames within its window, even if it has unacknowledged frames.

2. **Receiver Side:**

   - The receiver has a receiving window that can hold 'N' frames.

   - It receives frames out of order, but it can buffer and acknowledge them within its window.

   - The receiver sends cumulative acknowledgments for the highest correctly received frame. If it receives frame 2 before frame 1, it will acknowledge both frame 1 and frame 2 when it receives frame 2.

   - Out-of-sequence frames are discarded.

3. **Acknowledgments (ACKs):**

- The receiver sends acknowledgments (ACKs) for correctly received frames.

- When the sender receives an ACK for a frame with sequence number 'X,' it knows that all frames with sequence numbers up to 'X' have been successfully received and can be removed from the sender's window.

4. **Retransmission:**

- If the sender's timer expires or if it receives a negative ACK (NAK) for a frame, it retransmits all unacknowledged frames in its window.

- The receiver discards duplicate frames and reorders received frames.

Example:

Let's illustrate how Go-Back-N works with a simple example involving a sender (S) and a receiver (R):

1. Sender (S) sends frames 0, 1, 2, and 3 to Receiver (R).

2. Receiver (R) receives frames 0, 1, and 2 successfully and acknowledges them.

3. However, frame 3 is lost in transit.

4. Sender (S) notices the loss of frame 3 because it doesn't receive an ACK.

5. Sender (S) retransmits frames 3, 4, 5, and 6.

6. Receiver (R) receives frames 3 and 4 and acknowledges them.

7. Receiver (R) receives frames 5 and 6 but acknowledges them together with frame 7 (cumulative ACK).

8. Sender (S) proceeds to send frames 7, 8, 9, and so on.

The Go-Back-N protocol efficiently utilizes available bandwidth by allowing multiple frames in transit but requires more complex error handling and retransmission mechanisms compared to simpler protocols like Stop-and-Wait. It ensures reliable data transfer in a potentially noisy channel.

24. With an example, explain Go Back N protocol.

[7M][June/July-2022]

The Go-Back-N (GBN) protocol is a sliding window protocol used for reliable data transfer in data communication. It allows multiple frames to be in transit simultaneously and provides automatic retransmission of lost or corrupted frames. Let's explain the Go-Back-N protocol with an example involving a sender (S) and a receiver (R):

Assumptions:

- The sender has a sending window size (N) of 4.

- The receiver has a receiving window size (N) of 4.

- Frames are numbered sequentially starting from 0.

- The channel may introduce errors, leading to frame loss.

Scenario:

1. Sender (S) sends frames 0, 1, 2, and 3 to Receiver (R).

2. Receiver (R) receives all frames correctly and in order. It acknowledges these frames with ACK4 since it expects the next frame to be 4.

```
S: [0][1][2][3]       R: [0][1][2][3]      (ACK4)
```

3. Sender (S) proceeds to send frames 4, 5, 6, and 7. However, frame 5 is lost in transit.

```
S: [0][1][2][3] [4][5][6][7]       R: [0][1][2][3]      (ACK4)
```

```

4. Receiver (R) receives frames 4, 6, and 7 but not frame 5. It detects the gap in the sequence and discards out-of-order frames. Receiver (R) sends a selective acknowledgment (SACK5) for frame 4 and informs the sender to retransmit frame 5.

```
S: [0][1][2][3] [4][5][6][7]   R: [0][1][2][3] [5] (SACK5)
```

5. Sender (S) receives SACK5 and knows that frame 5 was lost. It retransmits frame 5 and continues to send frames 8, 9, 10, and 11.

```
S: [0][1][2][3] [4][5][6][7] [5]   R: [0][1][2][3]  [5] (ACK8)
```

6. Receiver (R) receives frame 5, acknowledges it with ACK8, and expects the next frame to be 8.

```
S: [0][1][2][3] [4][5][6][7] [5][8] R: [0][1][2][3]  [5][8] (ACK8)
```

7. Sender (S) continues to send frames 8, 9, 10, and 11.

```
S: [0][1][2][3] [4][5][6][7] [5][8] [9][10][11]       R: [0][1][2][3]  [5][8] (ACK8)
```
```

```

8. The process repeats until all frames are successfully received and acknowledged.

The Go-Back-N protocol efficiently utilizes the available bandwidth by allowing multiple frames in transit. However, it also requires more complex error handling and retransmission mechanisms compared to simpler protocols like Stop-and-Wait. It ensures reliable data transfer in a potentially noisy channel, where frames may be lost or corrupted. In the example, frame 5 was lost, but the protocol ensured its successful retransmission.

**25. Compare sliding window protocols. [7M][June/July 2022] Evaluate CO3**

**Sliding window protocols are a family of protocols used in data communication to manage the flow of data between sender and receiver efficiently. They differ in how they handle window sizes, acknowledgments, and error handling. Here's a comparison of three common sliding window protocols: Stop-and-Wait, Go-Back-N, and Selective Repeat:**

**1. Stop-and-Wait:**

**- **Sender Window Size:** 1**

**- **Receiver Window Size:** 1**

**- **ACKs:** Cumulative ACKs (acknowledge the receipt of the last frame)**

**- **Retransmission:** The sender waits for an acknowledgment before sending the next frame. If no ACK is received, it retransmits the current frame.**

**- **Efficiency:** Simple but less efficient as it underutilizes bandwidth.**

**- **Error Handling:** Basic; lost or corrupted frames are retransmitted.**

**2. Go-Back-N (GBN):**

**- **Sender Window Size:** N (N frames can be sent without waiting for acknowledgment)**

**- **Receiver Window Size:** 1 (acknowledges frames in order)**

**- **ACKs:** Cumulative ACKs (acknowledge the highest correctly received frame)**

**- **Retransmission:** If a frame is lost or corrupted, the sender retransmits all frames from the lost frame onward (goes back to N frames ago).**
```

- **Efficiency: More efficient than Stop-and-Wait, especially with larger N values, as it allows pipelining.**

- **Error Handling: Less efficient for handling selective retransmission.**

3. Selective Repeat (SR):

- **Sender Window Size: N (N frames can be sent without waiting for acknowledgment)**

- **Receiver Window Size: N (acknowledges frames out of order)**

- **ACKs: Selective acknowledgments (acknowledge each correctly received frame individually)**

- **Retransmission: If a frame is lost or corrupted, the sender only retransmits that specific frame, not all subsequent frames.**

- **Efficiency: Efficient and flexible, as it allows for maximum bandwidth utilization and selective retransmission.**

- **Error Handling: Effective for handling selective retransmission, suitable for error-prone channels.**

Summary:

- Stop-and-Wait is the simplest sliding window protocol but least efficient.

- Go-Back-N is more efficient than Stop-and-Wait due to its larger sender window but can be less efficient for error handling.

- Selective Repeat is the most efficient and flexible sliding window protocol, as it allows selective acknowledgment and retransmission.

26. Draw the frame format of HDLC and explain its configuration and transfer modes. [7M][Reg Dec/Jan 2022-2023 Set-3]

High-level Data Link Control (HDLC) is a group of communication protocols of the data link layer for transmitting data between network points or nodes. Since it is a data link protocol, data is organized into frames. A frame is transmitted via the network

to the destination that verifies its successful arrival. It is a bit - oriented protocol that is applicable for both point - to - point and multipoint communications.

Transfer Modes

HDLC supports two types of transfer modes, normal response mode and asynchronous balanced mode.

Normal Response Mode (NRM) − Here, two types of stations are there, a primary station that send commands and secondary station that can respond to received commands. It is used for both point - to - point and multipoint communications.

Asynchronous Balanced Mode (ABM) − Here, the configuration is balanced, i.e. each station can both send commands and respond to commands. It is used for only point - to - point communications.

HDLC Frame

HDLC is a bit - oriented protocol where each frame contains up to six fields. The structure varies according to the type of frame. The fields of a HDLC frame are −

- **Flag** − It is an 8-bit sequence that marks the beginning and the end of the frame. The bit pattern of the flag is 01111110.
- **Address** − It contains the address of the receiver. If the frame is sent by the primary station, it contains the address(es) of the secondary station(s). If it is sent by the secondary station, it contains the address of the primary station. The address field may be from 1 byte to several bytes.
- **Control** − It is 1 or 2 bytes containing flow and error control information.
- **Payload** − This carries the data from the network layer. Its length may vary from one network to another.

- **FCS** − It is a 2 byte or 4 bytes frame check sequence for error detection. The standard code used is CRC (cyclic redundancy code)

Types of HDLC Frames

There are three types of HDLC frames. The type of frame is determined by the control field of the frame −

- **I-frame** − I-frames or Information frames carry user data from the network layer. They also include flow and error control information that is piggybacked on user data. The first bit of control field of I-frame is 0.
- **S-frame** − S-frames or Supervisory frames do not contain information field. They are used for flow and error control when piggybacking is not required. The first two bits of control field of S-frame is 10.
- **U-frame** − U-frames or Un-numbered frames are used for myriad miscellaneous functions, like link management. It may contain an information field, if required. The first two bits of control field of U-frame is 11.