Error Detection

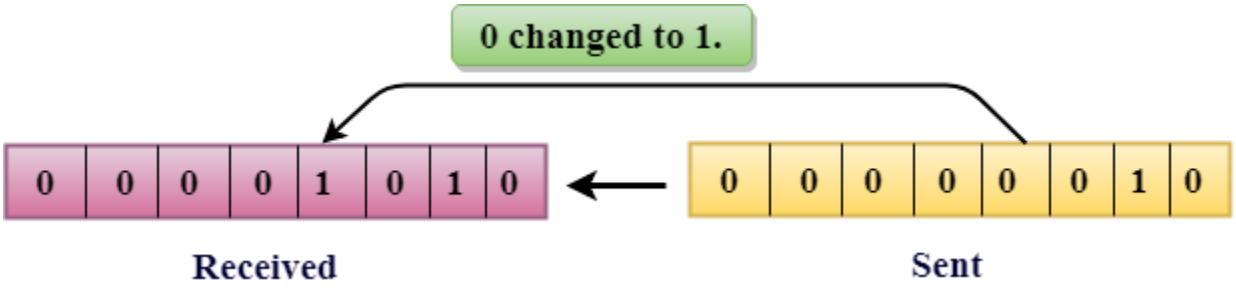
When data is transmitted from one device to another device, the system does not guarantee whether the data received by the device is identical to the data transmitted by another device. An Error is a situation when the message received at the receiver end is not identical to the message transmitted.

Types Of Errors

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

Single-Bit Error:

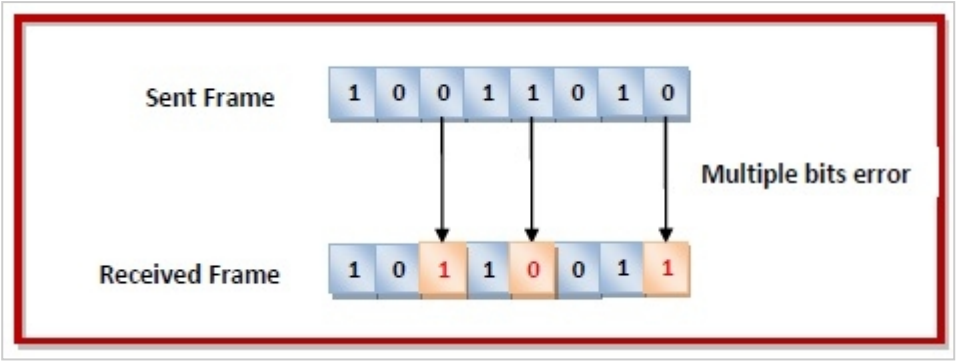
The only one bit of a given data unit is changed from 1 to 0 or from 0 to 1.



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

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.

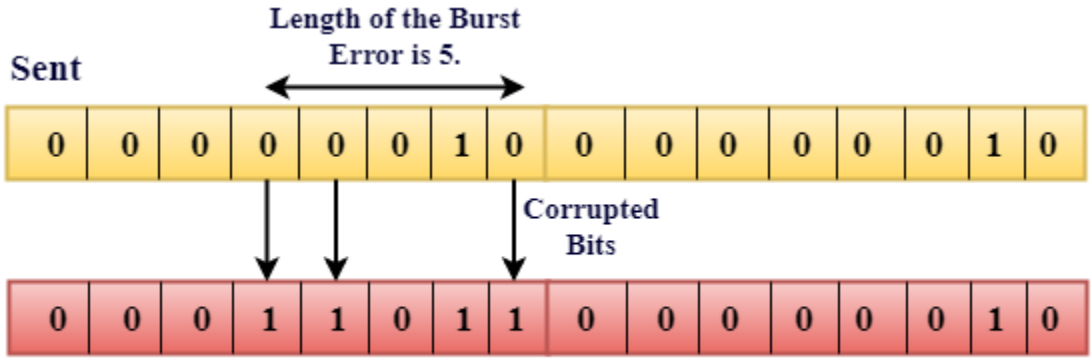
Multiple bits error – In the received frame, more than one bits are corrupted.



Burst Error:

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

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



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

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

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

Error Detection Techniques

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

Parity Check

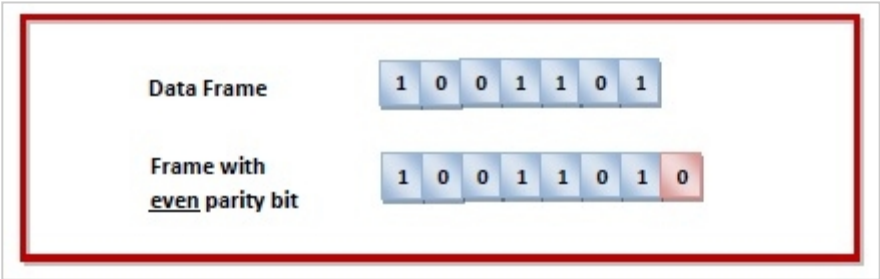
The parity check is done by adding an extra bit, called parity bit to the data to make a number of 1s either even in case of even parity or odd in case of odd parity.

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

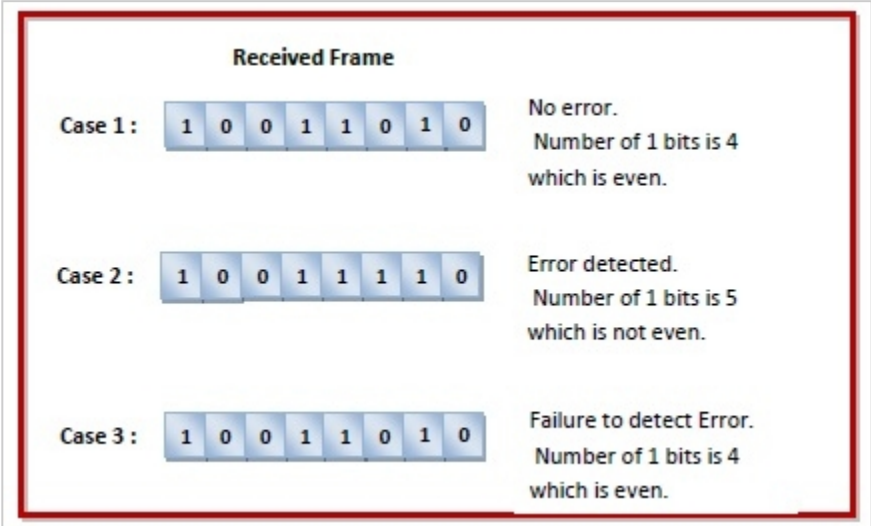
- In case of even parity: If a number of 1s is even then parity bit value is 0. If the number of 1s is odd then parity bit value is 1.

- 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.

Suppose that a sender wants to send the data 1001101 using even parity check method. It will add the parity bit as shown below.



The receiver will decide whether error has occurred by counting whether the total number of 1s is even. When the above frame is received, three cases may occur namely, no error, single bit error detection and failure to detect multiple bits error. This is illustrated as follows



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

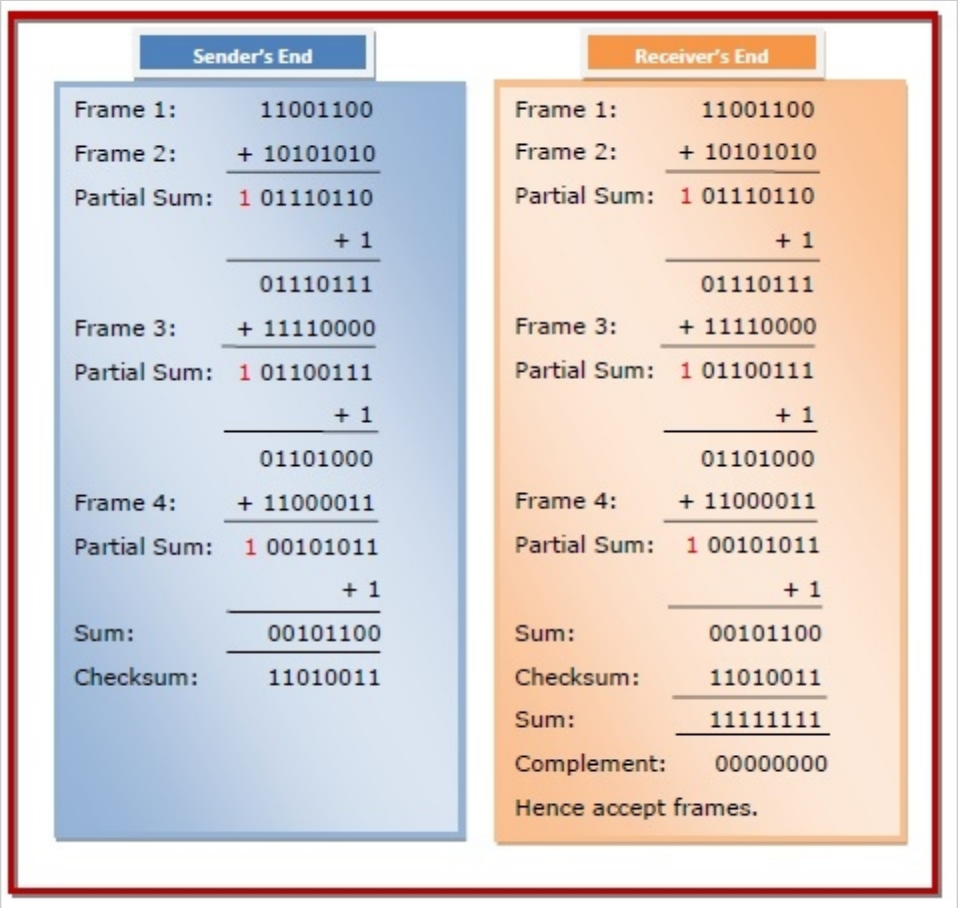
- Data is divided into fixed sized frames or segments.
- The sender adds the segments using 1's complement arithmetic to get the sum. It then complements the sum to get the checksum and sends it along with the data frames.
- The receiver adds the incoming segments along with the checksum using 1's complement arithmetic to get the sum and then complements it.
- If the result is zero, the received frames are accepted; otherwise, they are discarded.

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.

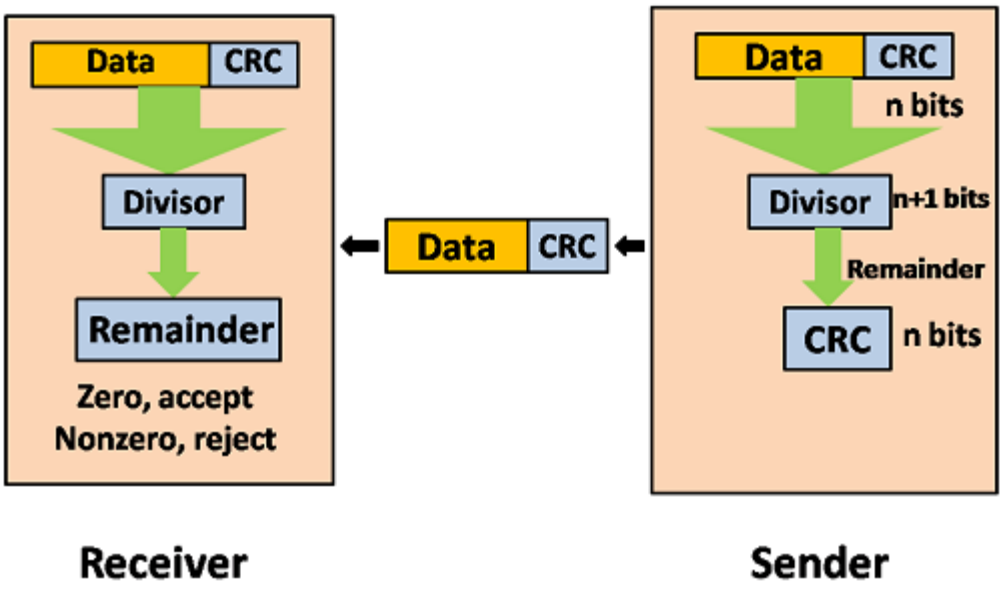
The receiver performs 1s complement arithmetic sum of all the frames including the checksum. The result is complemented and found to be 0. Hence, the receiver assumes that no error has occurred.



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.

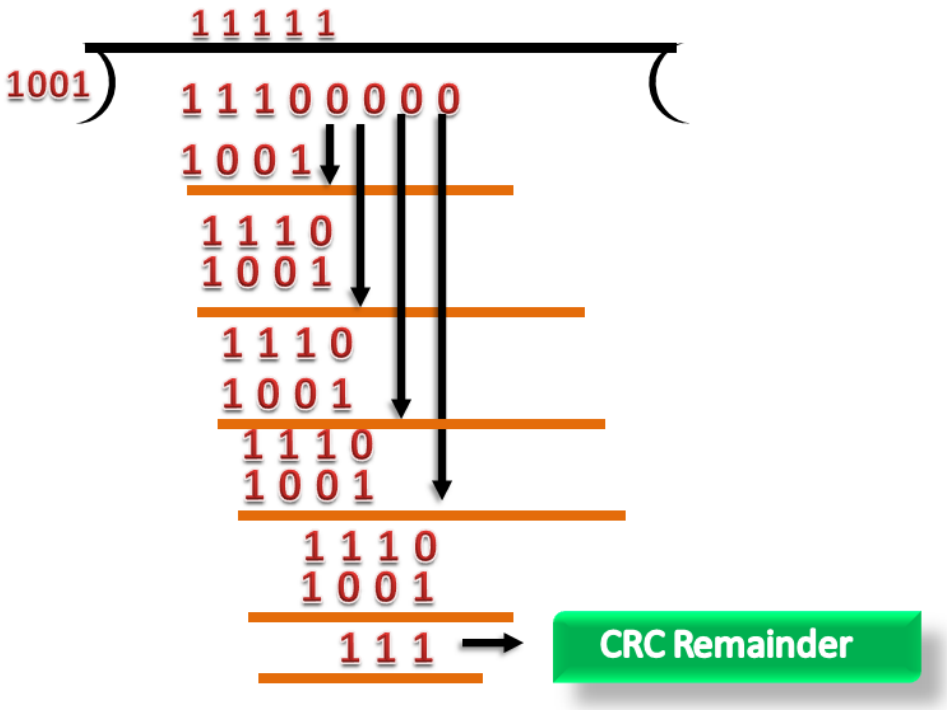
- Here, the sender performs binary division of the data segment by the divisor. It then appends the remainder called CRC bits to the end of the data segment. This makes the resulting data unit exactly divisible by the divisor.
- The receiver divides the incoming data unit by the divisor. If there is 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.



Suppose the original data is 11100 and divisor is 1001.

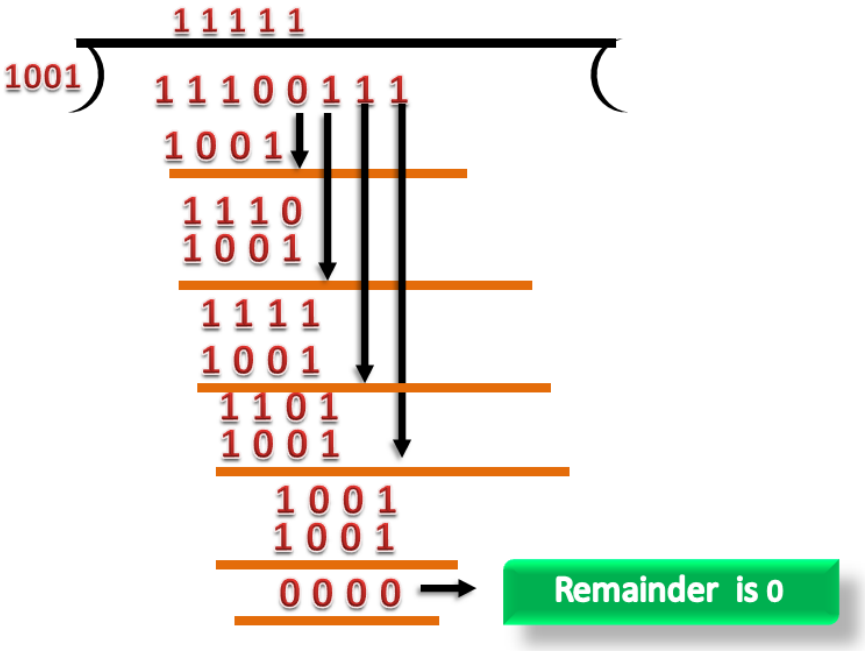
CRC Generator

- A CRC generator uses a modulo-2 division. Firstly, three zeroes are appended at the end of the data as the length of the divisor is 4 and we know that the length of the string 0s to be appended is always one less than the length of the divisor.
- Now, the string becomes 11100000, and the resultant string is divided by the divisor 1001.
- The remainder generated from the binary division is known as CRC remainder. The generated value of the CRC remainder is 111.
- CRC remainder replaces the appended string of 0s at the end of the data unit, and the final string would be 11100111 which is sent across the network.



CRC Checker

- The functionality of the CRC checker is similar to the CRC generator.
- When the string 11100111 is received at the receiving end, then CRC checker performs the modulo-2 division.
- A string is divided by the same divisor, i.e., 1001.
- In this case, CRC checker generates the remainder of zero. Therefore, the data is accepted.



Longitudinal Redundancy Check (LRC):

Longitudinal Redundancy Check (LRC) 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 :

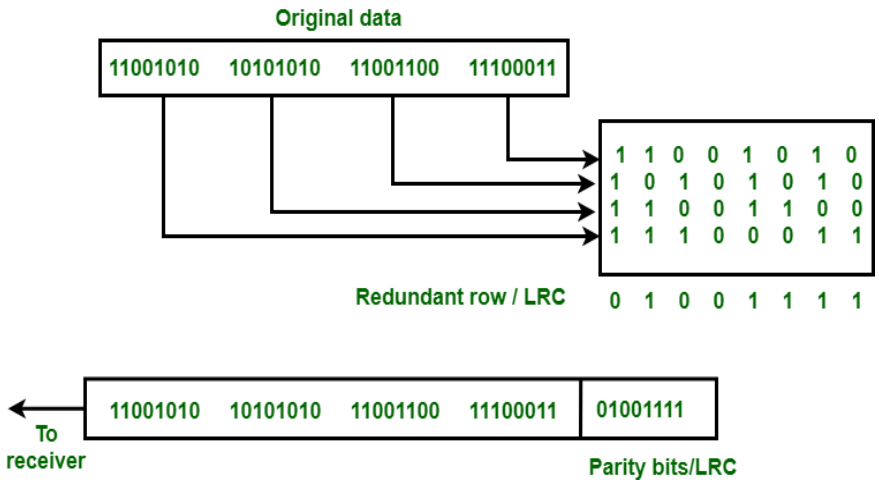


Figure: *LRC*

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.

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. Ex. Hamming Code

Hamming Code

Parity bits: The bit which is appended to the original data of binary bits so that the total number of 1s is even or odd.

Even parity: To check for even parity, if the total number of 1s is even, then the value of the parity bit is 0. If the total number of 1s occurrences is odd, then the value of the parity bit is 1.

Odd Parity: To check for odd parity, if the total number of 1s is even, then the value of parity bit is 1. If the total number of 1s is odd, then the value of parity bit is 0.

Algorithm of Hamming code:

- An information of 'd' bits are added to the redundant bits 'r' to form d+r.
- The location of each of the (d+r) digits is assigned a decimal value.
- The 'r' bits are placed in the positions 1,2,.....2^{k-1}.
- At the receiving end, the parity bits are recalculated. The decimal value of the parity bits determines the position of an error.

Relationship b/w Error position & binary number.

Error Position	Binary Number
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Let's understand the concept of Hamming code through an example:

Suppose the original data is 1010 which is to be sent.

Total number of data bits 'd' = 4

Number of redundant bits r : $2^r >= d+r+1$

$2^r >= 4+r+1$

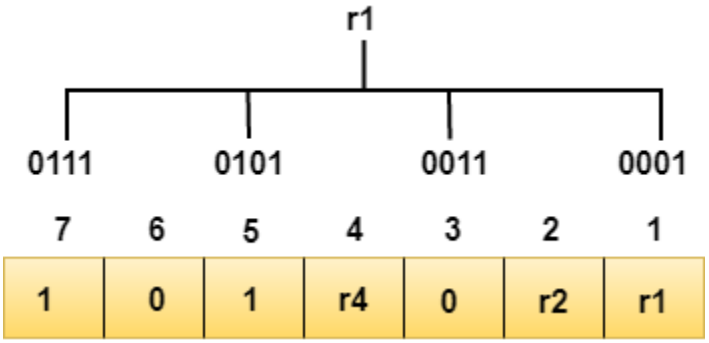
Therefore, the value of r is 3 that satisfies the above relation.

Total number of bits = d+r = 4+3 = 7;

Determining the Parity bits

Determining the r1 bit

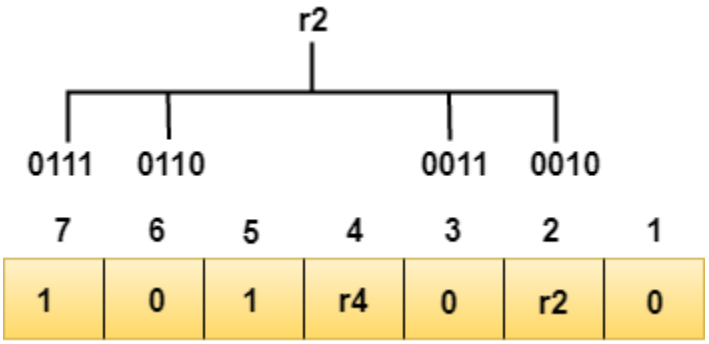
The r1 bit is calculated by performing a parity check on the bit positions whose binary representation includes 1 in the first position.



The total number of 1 at these bit positions corresponding to r1 is **even, therefore, the value of the r1 bit is 0**.

Determining r2 bit

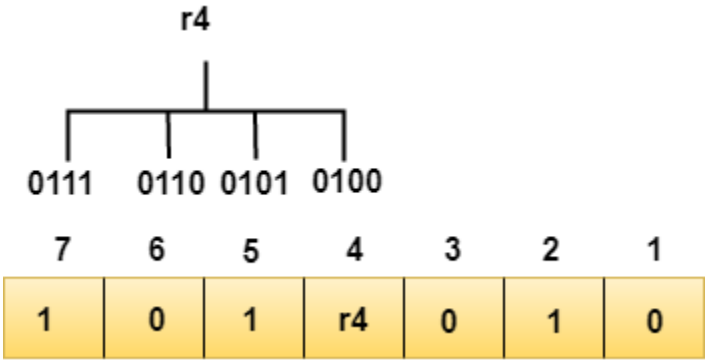
The r2 bit is calculated by performing a parity check on the bit positions whose binary representation includes 1 in the second position.



The total number of 1 at these bit positions corresponding to r2 is **odd, therefore, the value of the r2 bit is 1**.

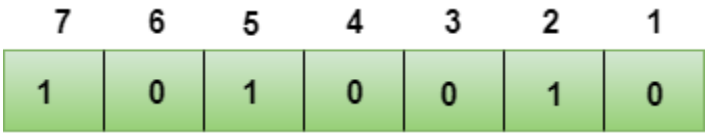
Determining r4 bit

The r4 bit is calculated by performing a parity check on the bit positions whose binary representation includes 1 in the third position.



The total number of 1 at these bit positions corresponding to r4 is **even, therefore, the value of the r4 bit is 0**.

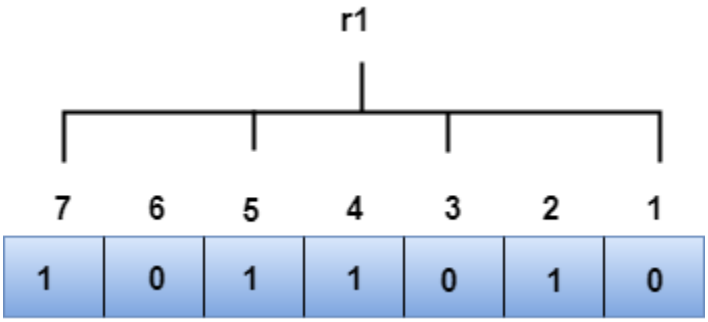
Data transferred is given below:



Suppose the 4th bit is changed from 0 to 1 at the receiving end, then parity bits are recalculated.

R1 bit

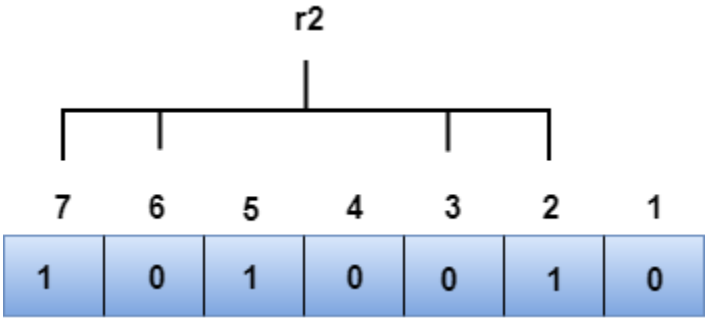
The bit positions of the r1 bit are 1,3,5,7



We observe from the above figure that the binary representation of r1 is 1100. Now, we perform the even-parity check, the total number of 1s appearing in the r1 bit is an even number. Therefore, the value of r1 is 0.

R2 bit

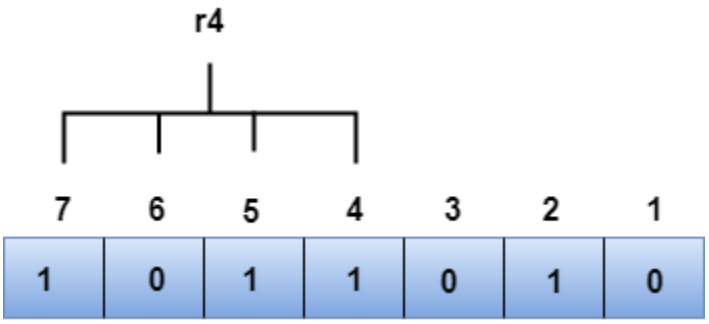
The bit positions of r2 bit are 2,3,6,7.



We observe from the above figure that the binary representation of r2 is 1001. Now, we perform the even-parity check, the total number of 1s appearing in the r2 bit is an even number. Therefore, the value of r2 is 0.

R4 bit

The bit positions of r4 bit are 4,5,6,7.



We observe from the above figure that the binary representation of r4 is 1011. Now, we perform the even-parity check, the total number of 1s appearing in the r4 bit is an odd number. Therefore, the value of r4 is 1.

- ***The binary representation of redundant bits, i.e., r4r2r1 is 100, and its corresponding decimal value is 4. Therefore, the error occurs in a 4^{th} bit position. The bit value must be changed from 1 to 0 to correct the error.***

Error control in data link layer is the process of detecting and correcting data frames that have been corrupted or lost during transmission.

In case of lost or corrupted frames, the receiver does not receive the correct data-frame and sender is ignorant about the loss. Data link layer follows a technique to detect transit errors and take necessary actions, which is retransmission of frames whenever error is detected or frame is lost. The process is called Automatic Repeat Request (ARQ).

Phases in Error Control

The error control mechanism in data link layer involves the following phases –

Detection of Error – Transmission error, if any, is detected by either the sender or the receiver.

Acknowledgment – acknowledgment may be positive or negative.

Positive ACK – On receiving a correct frame, the receiver sends a positive acknowledge.

Negative ACK – On receiving a damaged frame or a duplicate frame, the receiver sends a negative acknowledgment back to the sender.

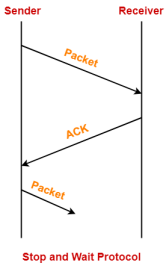
Retransmission – The sender maintains a clock and sets a timeout period. If an acknowledgment of a data-frame previously transmitted does not arrive before the timeout, or a negative acknowledgment is received, the sender retransmits the frame.

Error Control Techniques

There are three main techniques for error control –

In stop and wait protocol,

Sender sends one data packet and then waits for its acknowledgement.
Sender sends the next packet only after it receives the acknowledgement for the previous packet.



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

Stop and Wait ARQ
This protocol involves the following transitions –

A timeout counter is maintained by the sender, which is started when a frame is sent.

If the sender receives acknowledgment of the sent frame within time, the sender is confirmed about successful delivery of the frame. It then transmits the next frame in queue.

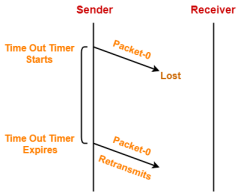
If the sender does not receive the acknowledgment within time, the sender assumes that either the frame or its acknowledgment is lost in transit. It then retransmits the frame.

If the sender receives a negative acknowledgment, the sender retransmits the frame.

How Stop and Wait ARQ Solves All Problems?

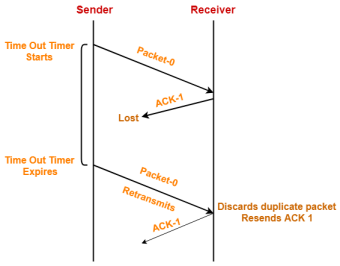
1. Problem of Lost Data Packet-

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.
This prevents the occurrence of deadlock.

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. Delayed Acknowledgement:
This is resolved by introducing sequence numbers for acknowledgement also.

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

Go-Back-N ARQ

The working principle of this protocol is –

The sender has buffers called sending window.

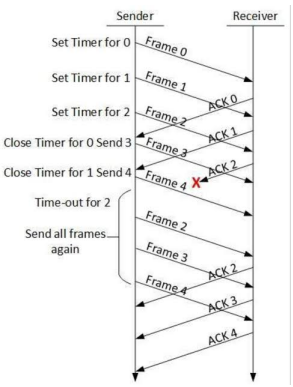
The sender sends multiple frames based upon the sending-window size, without receiving the acknowledgment of the previous ones.

The receiver receives frames one by one. It keeps track of incoming frame’s sequence number and sends the corresponding acknowledgment frames.

After the sender has sent all the frames in window, it checks up to what sequence number it has received positive acknowledgment.

If the sender has received positive acknowledgment for all the frames, it sends next set of frames.

If sender receives NACK or has not receive any ACK for a particular frame, it retransmits all the frames after which it does not receive any positive ACK.



Selective Repeat ARQ

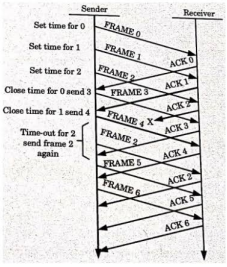
Both the sender and the receiver have buffers called sending window and receiving window respectively.
The sender sends multiple frames based upon the sending-window size, without receiving the acknowledgment of the previous ones.

The receiver also receives multiple frames within the receiving window size.

The receiver keeps track of incoming frame's sequence numbers, buffers the frames in memory.

It sends ACK for all successfully received frames and sends NACK for only frames which are missing or damaged.

The sender in this case, sends only packet for which NACK is received.



HDLC
HDLC (High-Level Data Link Control) is a bit-oriented protocol that is used for communication over the point-to-point and multipoint links. This protocol implements the mechanism of ARQ(Automatic Repeat Request). With the help of the HDLC protocol,full-duplex communication is possible.

HDLC is the most widely used protocol and offers reliability, efficiency, and a high level of Flexibility.

In order to make the HDLC protocol applicable for various network configurations, there are three types of stations and these are as follows:

Primary Station This station mainly looks after data like management. In the case of the communication between the primary and secondary station; it is the responsibility of the primary station to connect and disconnect the data link. The frames issued by the primary station are commonly known as commands.

Secondary Station The secondary station operates under the control of the primary station. The Frames issued by the secondary stations are commonly known as responses.

Combined Station The combined station acts as both Primary stations as well as Secondary stations. The combined station issues both commands as well as responses.

Transfer Modes in HDLC
The HDLC protocol offers two modes of transfer that mainly can be used in different configurations. These are as follows:

Normal Response Mode(NRM)
Asynchronous Balanced Mode(ABM)
Asynchronous Response Mode(ARM)

COMPUTER NETWORK BASICS
Introduction To Computer Networks
Uses of Computer Networks
Line Configuration
Types of Network Topology
Transmission Modes
Transmission Mediums
Bounded/Guided Transmission Media
UnBounded/UnGuided Transmission Media

Types of Communication Networks
Connection Oriented and Connectionless Services
NETWORK LAYER
Quality of Service(QoS)
Network Layer
IGMP Protocol
REFERENCE MODELS
Reference Models
PHYSICAL LAYER
Digital Transmission
Multiplexing
Switching
Circuit-Switched
Message-Switched Networks
Packet Switching
DATA LINK LAYER
Error Correction
Data Link Control
Flow and Error
Simplest Protocol
Stop-and-Wait Protocol
Go-Back-N Automatic Repeat
Sliding Window Protocol
HDLC Protocol
Point-to-Point Protocol
Multiple Access in DL
Channelization Protocols
Gigabit Ethernet
Random Access Protocol
Controlled Access Protocols
Carrier Sense Multiple Access
TRANSPORT LAYER
Transport Layer
Telnet vs SSH
UDP Protocol
TCP vs UDP
TCP - Protocol
ISO/OSI REFERENCE MODEL
Introduction to Reference Models
ISO/OSI Model

OSI Model: Physical Layer
OSI Model: Datalink Layer
OSI Model: Network Layer
OSI Model: Transport Layer
OSI Model: Session Layer
OSI Model: Presentation Layer
OSI Model: Application Layer
TCP/IP REFERENCE MODELCOMPUTER NETWORKS
The TCP/IP Reference Model
Difference between OSI and TCP/IP Model
Key Terms
SESSION LAYER
Session Layer
COMPUTER NETWORKS
Components of Computer Networks
Features of Computer Network
Protocols and Standards
Connection Oriented and Connectionless
Transmission Modes
OSI Vs TCP/IP
PRESENTATION LAYER
Presentation Layer
APPLICATION LAYER
HTTP Protocol
FTP Protocol
SMTP Protocol
POP Protocol
SNMP Protocol
TELNET
Electronic Mail
MIME Protocol
World Wide Web
SSH
DNS Protocol
Home
Computer Networks
HDLC Protocol
HDLC Protocol
In this tutorial, we will be covering the HDLC protocol in the data link layer of the OSI Model.

HDLC (High-Level Data Link Control) is a bit-oriented protocol that is used for communication over the point-to-point and multipoint links. This protocol implements the mechanism of ARQ(Automatic Repeat Request). With the help of the HDLC protocol,full-duplex communication is possible.

HDLC is the most widely used protocol and offers reliability, efficiency, and a high level of Flexibility.

In order to make the HDLC protocol applicable for various network configurations, there are three types of stations and these are as follows:

Primary Station This station mainly looks after data like management. In the case of the communication between the primary and secondary station; it is the responsibility of the primary station to connect and disconnect the data link. The frames issued by the primary station are commonly known as commands.

Secondary Station The secondary station operates under the control of the primary station. The Frames issued by the secondary stations are commonly known as responses.

Combined Station The combined station acts as both Primary stations as well as Secondary stations. The combined station issues both commands as well as responses.

Link configurations in HDLC
The two link configurations are
• Unbalanced configuration: Consists of one primary and one or more sec-
ondary stations and supports both full-duplex and half-duplex transmis-
sion.
• Balanced configuration: Consists of two combined stations and supports both
full-duplex and half-duplex transmission.

Transfer Modes in HDLC
The HDLC protocol offers two modes of transfer that mainly can be used in different configurations. These are as follows:

Normal Response Mode(NRM)

Asynchronous Balance Mode(ABM)

Let us now discuss both these modes one by one:

1. Normal Response Mode(NRM)
In this mode, the configuration of the station is unbalanced. There are one primary station and multiple secondary stations. Where the primary station can send the commands and the secondary station can only respond.

This mode is used for both point-to-point as well as multiple-point links.

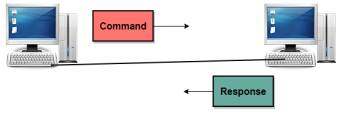


Figure: Point-to-Point

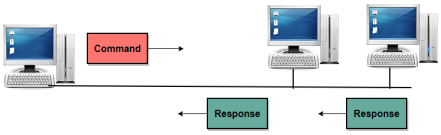


Figure: Multipoint

2. Asynchronous Balanced Mode(ABM)
In this mode, the configuration of the station is balanced. In this mode, the link is point-to-point, and each station can function as a primary and as secondary.
Used with a balanced configuration.Either combined station may initiate transmission without receiving permission from the other combined station.
Asynchronous Balanced mode(ABM) is a commonly used mode today.

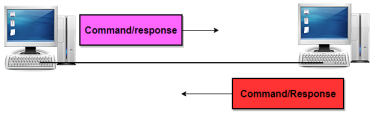


Figure: Asynchronous Balance Mode

3. Asynchronous Response mode(ARM)
Used with an unbalanced configuration. The secondary may initiate transmission without explicit permission of the primary. The primary still retains responsibility for the line, including initialization, error recovery, and logical disconnection.

HDLC Frames
In order to provide the flexibility that is necessary to support all the options possible in the modes and Configurations that are just described above. There are three types of frames defined in the HDLC:

Information Frames(I-frames) These frames are used to transport the user data and the control information that is related to the user data. If the first bit of the control field is 0 then it is identified as I-frame.

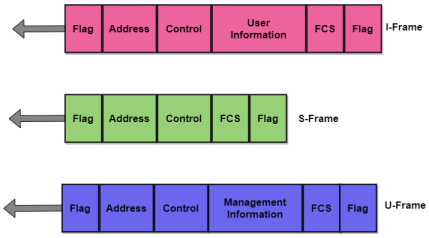
Supervisory Frames(S-frames) These frames are only used to transport the control information. If the first two bits of the control field are 1 and 0 then the frame is identified as S-frame

Unnumbered Frames(U-Frames) These frames are mainly reserved for system management. These frames are used for exchanging control information between the communicating devices.If the first two bits of the control field are 1 and 1 then the frame is identified as U-frame

Each type of frame mainly serves as an envelope for the transmission of a different type of message.

Frame Format

There are up to six fields in each HDLC frame. There is a beginning flag field, the address field then, a control field, an information field, a frame check sequence field(FCS), and an ending field.



In the case of the multiple-frame transmission, the ending flag of the one frame acts as the beginning flag of the next frame.

1. Flag Field

This field of the HDLC frame is mainly a sequence of 8-bit having the bit pattern 01111110 and it is used to identify the beginning and end of the frame. The flag field mainly serves as a synchronization pattern for the receiver.

2. Address Field

It is the second field of the HDLC frame and it mainly contains the address of the secondary station. This field can be 1 byte or several bytes long which mainly depends upon the need of the network. In case if the frame is sent by the primary station, then this field contains the address(es) of the secondary stations. If the frame is sent by the secondary station, then this field contains the address of the primary station.

3. Control Field

This is the third field of the HDLC frame and it is a 1 or 2-byte segment of the frame and is mainly used for flow control and error control. Bits interpretation in this field mainly depends upon the type of the frame.

4. Information Field

This field of the HDLC frame contains the user's data from the network layer or the management information. The length of this field varies from one network to another.

5. FCS Field

FCS means Frame check sequence and it is the error detection field in the HDLC protocol. There is a 16 bit CRC code for error detection.

Features of HDLC Protocol

Given below are some of the features of the HDLC protocol:

1.This protocol uses bits to stuff flags occurring in the data.
2.This protocol is used for point-to-point as well as multipoint link access.
3.HDLC is one of the most common protocols of the data link layer.
4.HDLC is a bit-oriented protocol.
5.This protocol implements error control as well as flow control.