ERROR

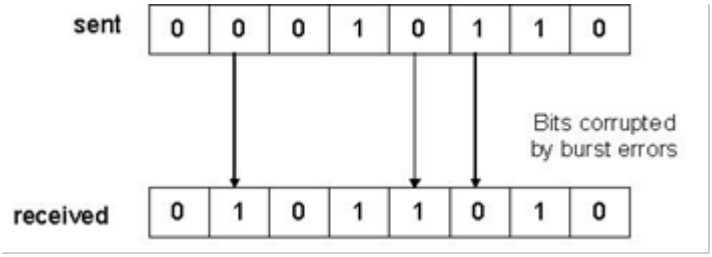
Networks must be able to transfer data from one device to another with complete accuracy. Some part of a message will be altered in transit than that the entire content will arrive intact. Many factors like line noise can alter or wipe out one or more bits of a given data unit. This is known as errors.

TYPES OF ERRORS

There are two types. They

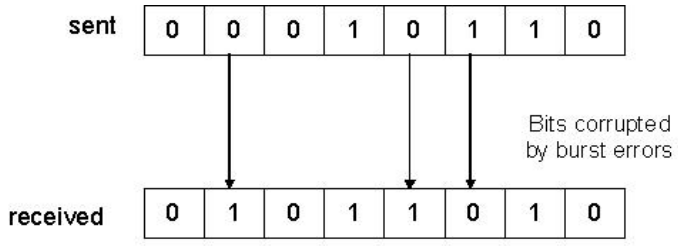
are, 1. Single Bit Error

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



2. Burst Bit Error

It means that two or more bits in the data unit have changed.



- A burst bit does not necessarily means that the errors occur in consecutive bits
- The length of the bust error is measured from the first corrupted bit to the last corrupted bit. Some bits in between may not be corrupted.

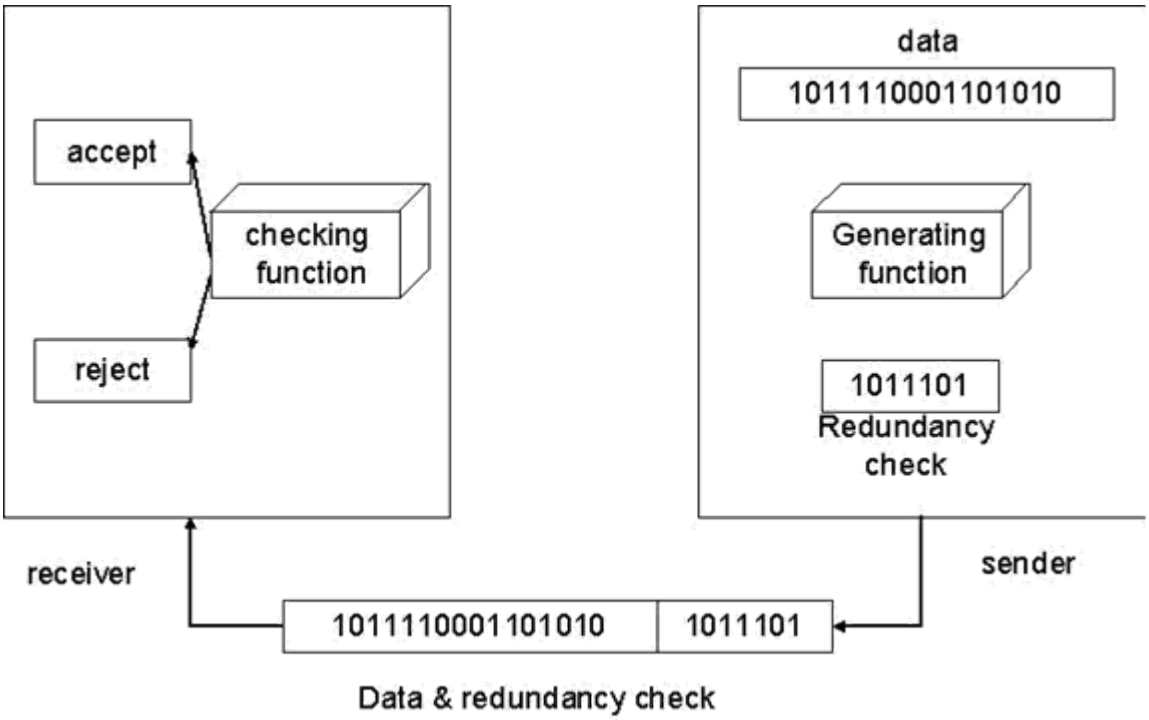
ERROR DETECTION

For reliable communication errors must be detected and corrected. For error detection we are using many mechanisms.

REDUNDANCY

One error detection mechanism is sending every data unit twice. The receiving device then would be able to do a bit for bit comparison between the two versions of the data. Any discrepancy would indicate an error, and an appropriate correction mechanism could be used.

But instead of repeating the entire data stream, a shorter group of bits may be appended to the end of each unit. This technique is called redundancy because extra bits are redundant to the information. They are discarded as soon as the accuracy of the transmission has been determined.



TYPES

Four types of redundancy checks are used in data communications. They are,

1. vertical redundancy check (VRC)
2. longitudinal redundancy check (LRC)
3. cyclic redundancy check (CRC)
4. checksum

VERTICAL REDUNDANCY CHECK:

It is also known as parity check. In this technique a redundant bit called a parity bit is appended to every data unit so that the total number of 1s in the unit including the parity bit becomes even for even parity or odd for odd parity.

In even parity, the data unit is passed through the even parity generator. It counts the number of 1s in the data unit. If odd number of 1s, then it sets 1 in the parity bit to make the number of 1s as even. If the data unit having even number of 1s then it sets in the parity bit to maintain the number of 1s as even. When it reaches its destination, the receiver puts all bits through an even parity checking function. If it counts even number of 1s than there is no error. Otherwise there is some error.

EXAMPLE:

The data is : 01010110

The VRC check : 010101100

In odd parity, the data unit is passed through the odd parity generator. It counts the number of 1s in the data unit. If even number of 1s, then it sets 1 in the parity bit to make the number of 1s as odd. If the data unit having odd number of 1s then it sets in the parity bit to maintain the number of 1s as odd. When it reaches its destination, the receiver puts all bits through an odd parity checking function. If it counts odd number of 1s than there is no error. Otherwise there is some error.

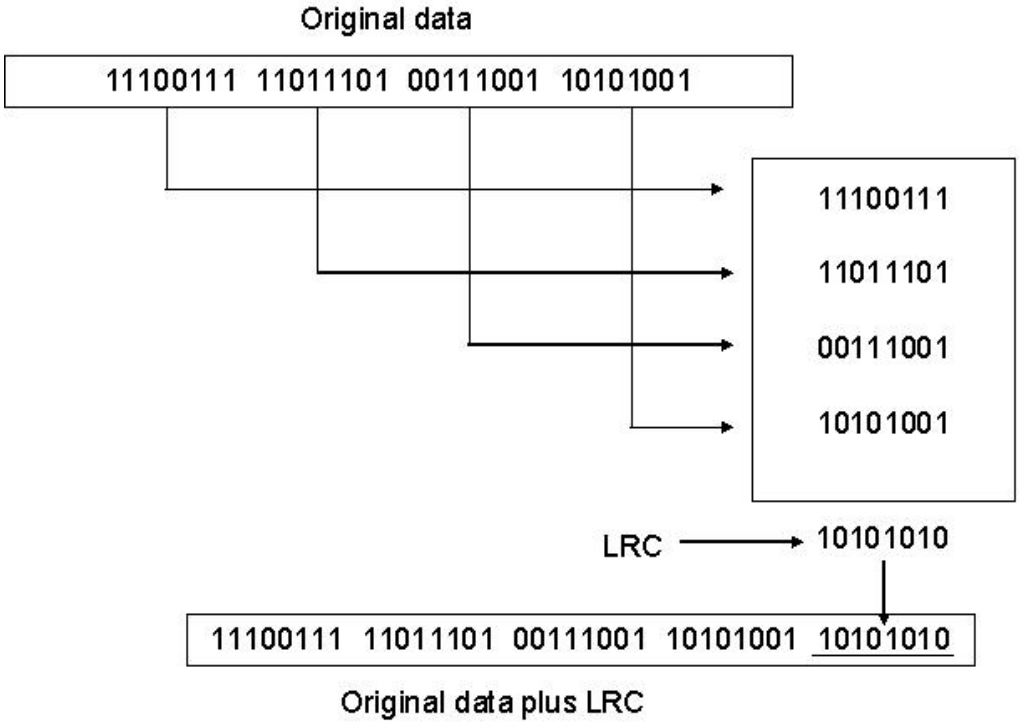
EXAMPLE

The data is: 01010110

The VRC check: 01010111

LONGITUDINAL REDUNDANCY CHECK

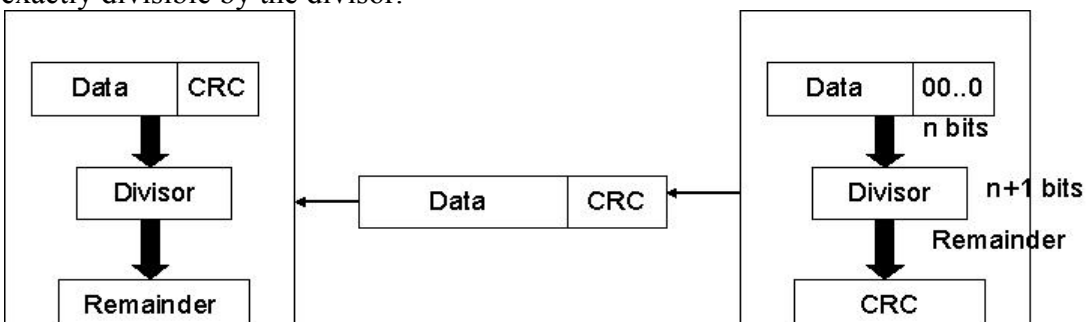
In this, a block of bits is organized in a table (rows and columns). For example, instead of sending a block of 32 bits, we organize them in a table made of four roes and eight columns. We then calculate the parity bit for each column and create a new row of eight bits which are the parity bits for the whole block



CYCLIC REDUNDANCY CHECK

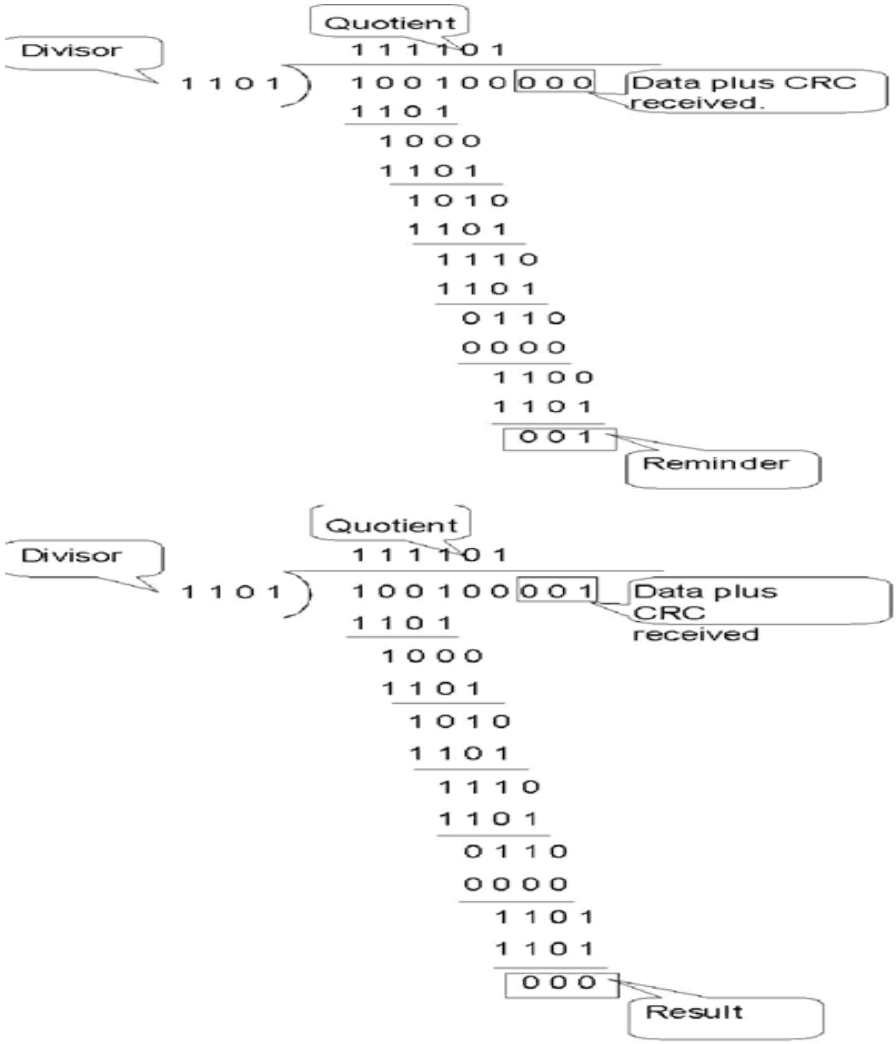
CRC is based on binary division. In this a sequence of redundant bits, called CRC remainder is appended to the end of a data unit so that the resulting data unit becomes exactly divisible by a second predetermined binary number. At its destination, the incoming data unit is divided by the same number. If at this step there is no reminder, the data unit is assumed to be intact and therefore accepted. A remainder indicates that the data unit has been changed in transit and therefore must be rejected.

Here, the remainder is the CRC. It must have exactly one less bit than the divisor, and appending it to the end of the data string must make the resulting bit sequence exactly divisible by the divisor.



First, a string of n-1 0s is appended to the data unit. The number of 0s is one less than the number of bits in the divisor which is n bits. Then the newly elongated data unit is divided by the divisor using a process called binary division. The remainder is CRC. The CRC is replaces the appended 0s at the end of the data unit.

The data unit arrives at the receiver first, followed by the CRC. The receiver treats whole string as the data unit and divides it by the same divisor that was used to find the CRC remainder. If the remainder is 0 then the data unit is error free. Otherwise it having some error and it must be discarded.



CHECKSUM

The error detection method used by the higher layer protocols is called checksum. It consists of two arts. They are,

1. checksum generator
2. checksum checker

Checksum Generator:

In the sender, the checksum generator subdivides the data unit into equal segments of n bits. These segments are added with each other by using one‟s complement arithmetic in such a way that the total is also n bits long. That total is then complemented and appended to the end of the data unit.

Checksum Checker:

The receiver subdivides the data unit as above and adds all segments together and complements the result. If the extended data unit is intact, the total value found by adding the data segments and the checksum field should be zero. Otherwise the packet contains an error and the receiver rejects it.

EXAMPLE

At the sender

Data unit: 10101001 00111001

10101001

00111001

Sum 1100010

Checksum 00011101

At the receiver

1)

Received data: 10101001 00111001

00011101 10101001 00111001

00011101

Sum 11111111

Complement 00000000

It means that the patter is ok.

2)

Received data: 1010111 111001 00011101

	10101111
	11111001
	00011101
Result	11000101
Carry	1
Sum	11000110
Complement	00111001

It means that the patter is corrupted.

ERROR CORRECTION

Error correction is handled in two ways. In one, when an error is discovered, the receiver can have the sender retransmit the entire data unit. In the other, a receiver can use an error correcting code, which automatically corrects certain errors.

Types of error correction:

1. Single bit error correction
2. Burst bit error correction

Single Bit Error Correction

To correct a single bit error in an ASCII character, the error correction code must determine which of the seven bits has changed. In this case we have to determine eight different states: no error, error in position 1, error in position 2, error in position 3, error in position 4, error in position 5, error in position 6, error in position 7. It looks like a three bit redundancy code should be adequate because three bits can show eight different states. But what if an error occurs in the redundancy bits? Seven bits of data and three bits of redundancy bits equal 10 bits. So three bits are not adequate.

To calculate the number of redundancy bits (r) required to correct a given number of data bits (m) we must find a relationship between m and r.

If the total number of bits in a transmittable unit is m+r then r must be able to indicate at least m+r+1 different state. Of these, one state means no error and m+r states indicate the location of an error in each of the m+r positions.

So m+r+1 state must be discoverable by r bits. And r bits can indicate 2^r different states. Therefore, 2^r must be equal to or greater than m+r+1;

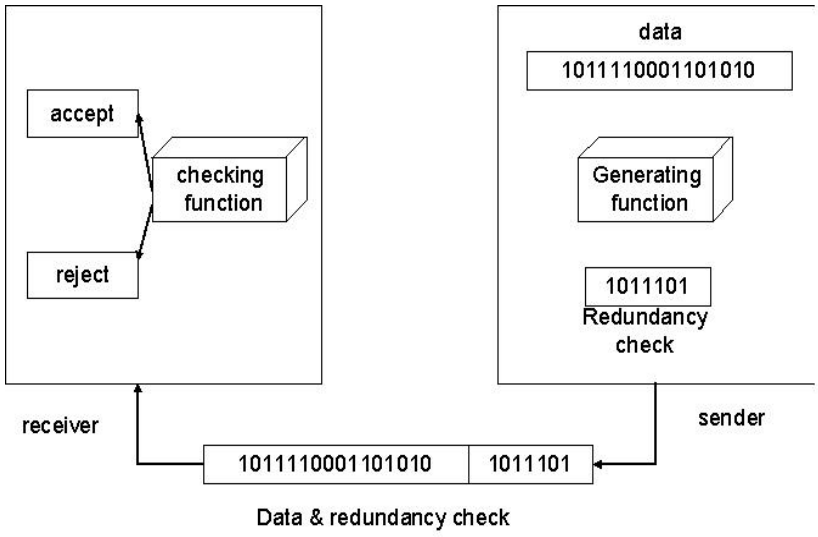
$$2^r >= m+r+1$$

NUMBER OF DATA BITS (M)	NUMBER OF REDUNDANCY BITS (R)	TOTAL BITS (M+R)
1	2	3
2	3	5
3	3	6
4	3	7
5	4	9
6	4	10
7	4	11

Hamming Code:

The hamming code can be applied to data units of any length and uses the relationship between data and redundancy bits.

Positions of redundancy bits in hamming code



The combinations used to calculate each of the four r values for a seven bit data sequence are as follows:

r1 :1,3,5,7,9,11

r2 : 2,3,6,7,10,11

r3 : 4,5,6,7

r4 : 8,9,10,11

Here, r1 bit is calculated using all bit positions whose binary representation includes a 1 in the rightmost position (0001, 0011, 0101, 0111, 1001, and 1011). The r2 bit is calculated using all bit positions with a 1 in the second position (0010, 0011, 0110, 0111, 1010 and 1011), and for r3 1 at third bit position (0100, 0101, 0110 and 0111) for r4 1 at fourth bit position (1000, 1001, 1010 and 1011).

Calculating the r Values:

In the first step, we place each bit of the original character in its appropriate positions in the 11 bit unit. Then, we calculate the even parities for the various bit combinations. The parity value of each combination is the value of the corresponding r bit. For example r1 is calculated to provide even parity for a combination of bits 3, 5, 7, 9, 11.

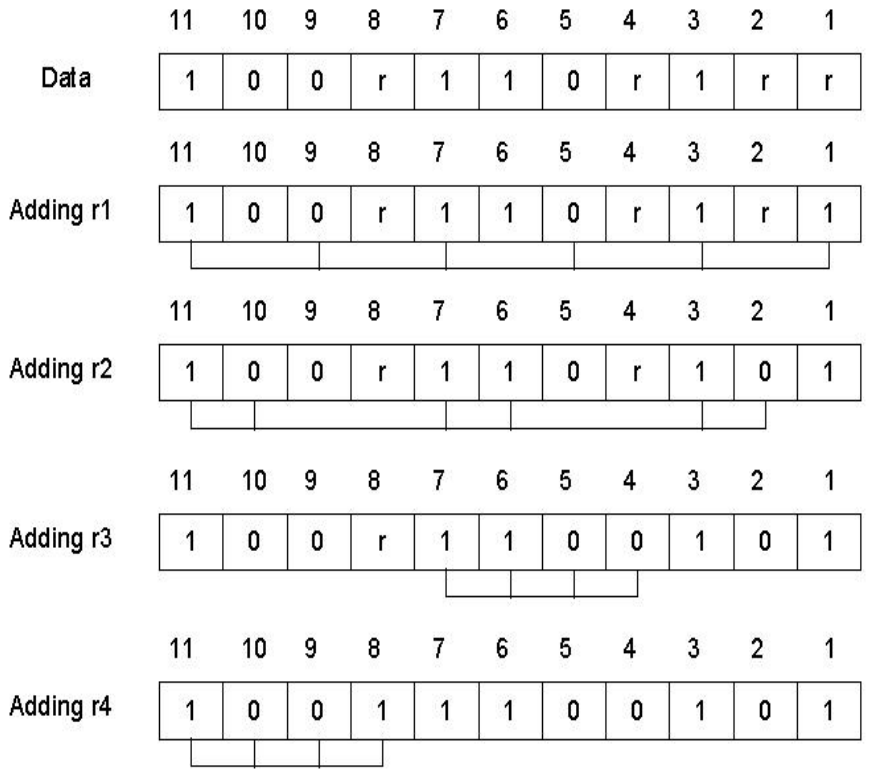
Error Detection and Correction:

Example:

At the sender:

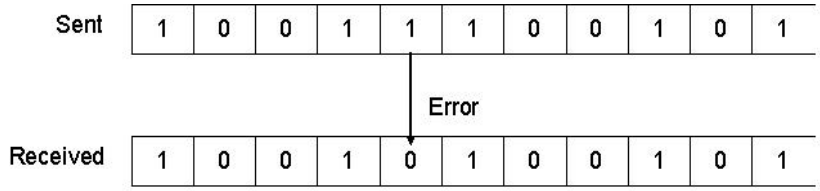
Data to be sent: 1001101

Redundancy bit calculation:



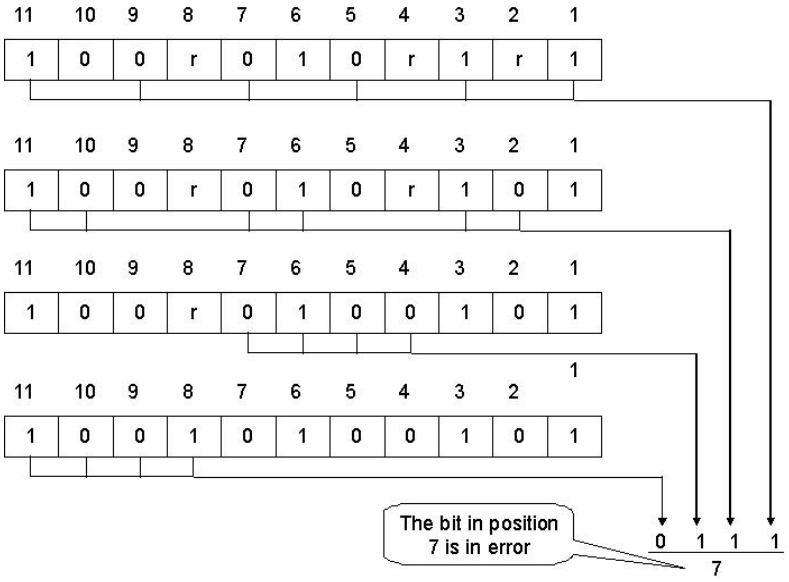
Data sent with redundancy bits: 10011100101

During transmission:



At the receiver:

The receiver takes the transmission and recalculates four new r values using the same set of bits used by the sender plus the relevant parity (r) bit for each set. Then it assembles the new parity values into a binary number in order of r position (r8, r4, r2, r1).



Once the bit is identified, the receiver can reverse its value and correct the error.

Burst Bit Error Correction:

A hamming code can be designed to correct burst errors of certain length. The number of redundancy bits required to make these corrections, however, is dramatically higher than that required for single bit errors. To correct double bit errors, for example, we must take into consideration that the two bits can be a combination of any two bits in the entire sequence. Three bit correction means any three bits in the entire sequence and so on.

FUNCTIONS OF DATA LINK LAYER:

The data link layer is responsible for the following functions. They are,

1. Line discipline or Access control
2. Flow control
3. Error control
4. Framing

LINE DISCIPLINE

Communications requires at least two devices, one to send and one to receive. If both devices are ready to send some information and put their signals on the link then the two signals collides each other and became nothing. To avoid such a situation the data link layer use a mechanism called line discipline.

Line discipline coordinates the link system. It determines which device can send and when it can send. It answers then question, who should send now?

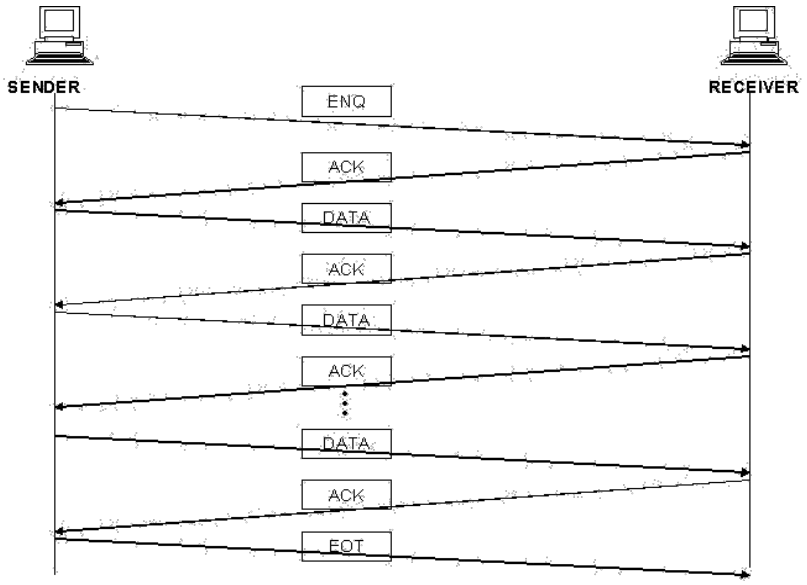
Line discipline can serve in two ways:

1. enquiry / acknowledgement (ENQ / ACK)
2. poll / select (POLL / SELECT)

ENQ / ACK:

This method is used in peer to peer communications. That is where there is a dedicated link between two devices.

The initiator first transmits a frame called an enquiry (ENQ) asking I the receiver is available to receive data. The receiver must answer either with an acknowledgement (ACK) frame if it ready to accept or with a negative acknowledgement (NAK) frame if it is not ready. If the response is positive, the initiator is free to send its data. Otherwise it waits, and try again. Once all its data have been transmitted, the sending system finishes with an end of transmission (EOT) frame.



POLL / SELECT

This method of line discipline works with topologies where one device is designated as primary and the other devices are secondary.

Whenever a multi point link consists of a primary device and multiple secondary devices using a single transmission line, all exchanges must be made through the primary device even when the ultimate destination is a secondary. The primary device controls the link. The secondary devices follow its instructions. The primary device only determines which device is allowed to use the channel at a given time.

The primary asks the secondary if they have anything to send; this function is called polling. And the primary tells the target secondary to get ready to receive; this function is called selecting.

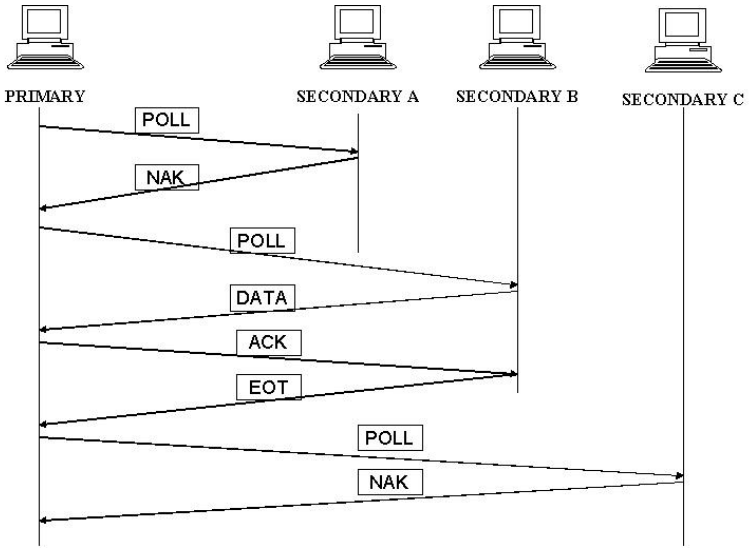
POLL:

This function is used by the primary device to solicit transmission from the secondary devices. The secondary are not allowed to transmit data unless asked by the primary device.

When the primary ready to receive data, it must ask (poll) each device in turn if it has anything to send. If the secondary have data to transmit it sends the data frame otherwise sends a negative acknowledgment (NAK).

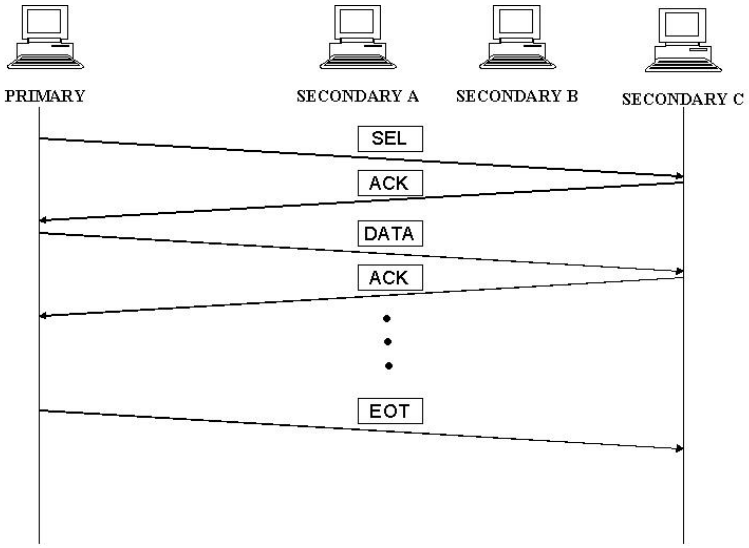
The primary then polls the next secondary. When the response is positive (a data frame), the primary reads the frame and returns an acknowledgment (ACK).

There are two possibilities to terminate the transmission: either the secondary sends all data, finishing with an EOT frame, or the primary says “timer‟s up”. Then the primary cal polls the remaining devices.



SELECT:

This mode of function is used whenever the primary device has something to send. It alerts the intended secondary device get ready to receive data. Before sending data it sends the select (SEL) frame. The receiver returns an ACK frame. Then the primary sends data.



FLOW CONTROL AND ERROR CONTROL

FLOW CONTROL

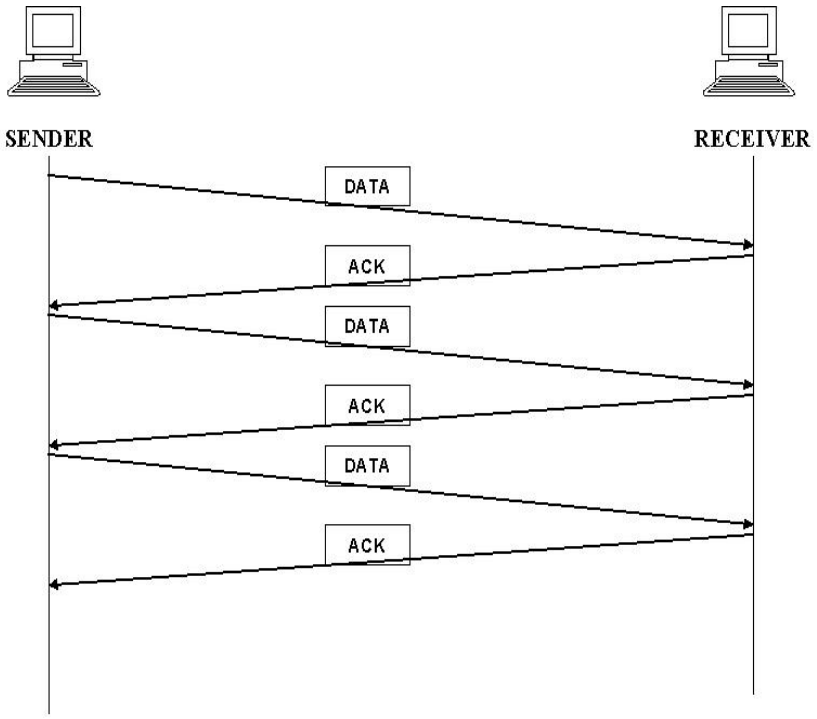
It refers to a set of procedures used to restrict the amount of data flow between sending and receiving stations. It tells the sender how much data it can transmit before it must wait for an acknowledgement from the receiver.

There are two methods are used. They are,

1. stop and wait
2. sliding window

STOP AND WAIT:

In this method the sender waits for acknowledgment after every frame it sends. Only after an acknowledgment has been received, then the sender sends the next frame.

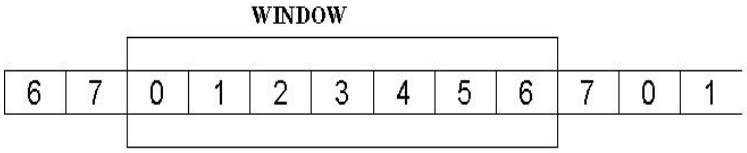


The advantage is simplicity. The disadvantage is inefficiency.

SLIDING WINDOW:

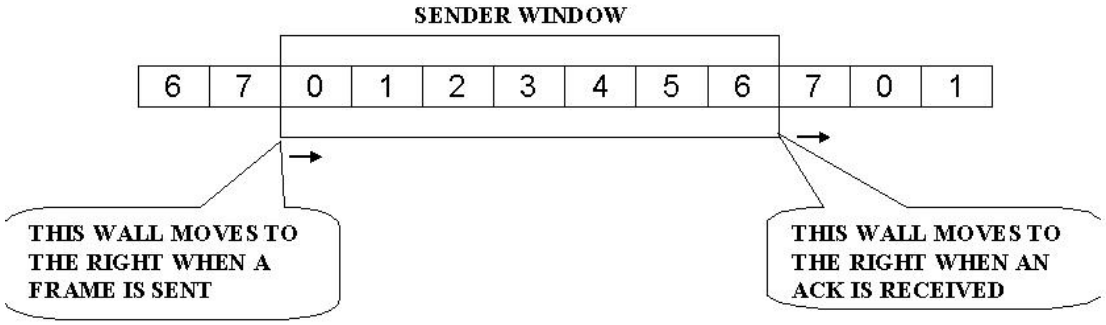
In this method, the sender can transmit several frames before needing an acknowledgment. The receiver acknowledges only some of the frames, using a single ACK to confirm the receipt of multiple data frames.

The sliding window refers to imaginary boxes at both the sender and receiver. This window provides the upper limit on the number of frames that can be transmitted before requiring an acknowledgement. To identify each frame the sliding window scheme introduces the sequence number. The frames are numbered as 0 to n-1. And the size of the window is n-1. Here the size of the window is 7 and the frames are numbered as 0,1,2,3,4,5,6,7.

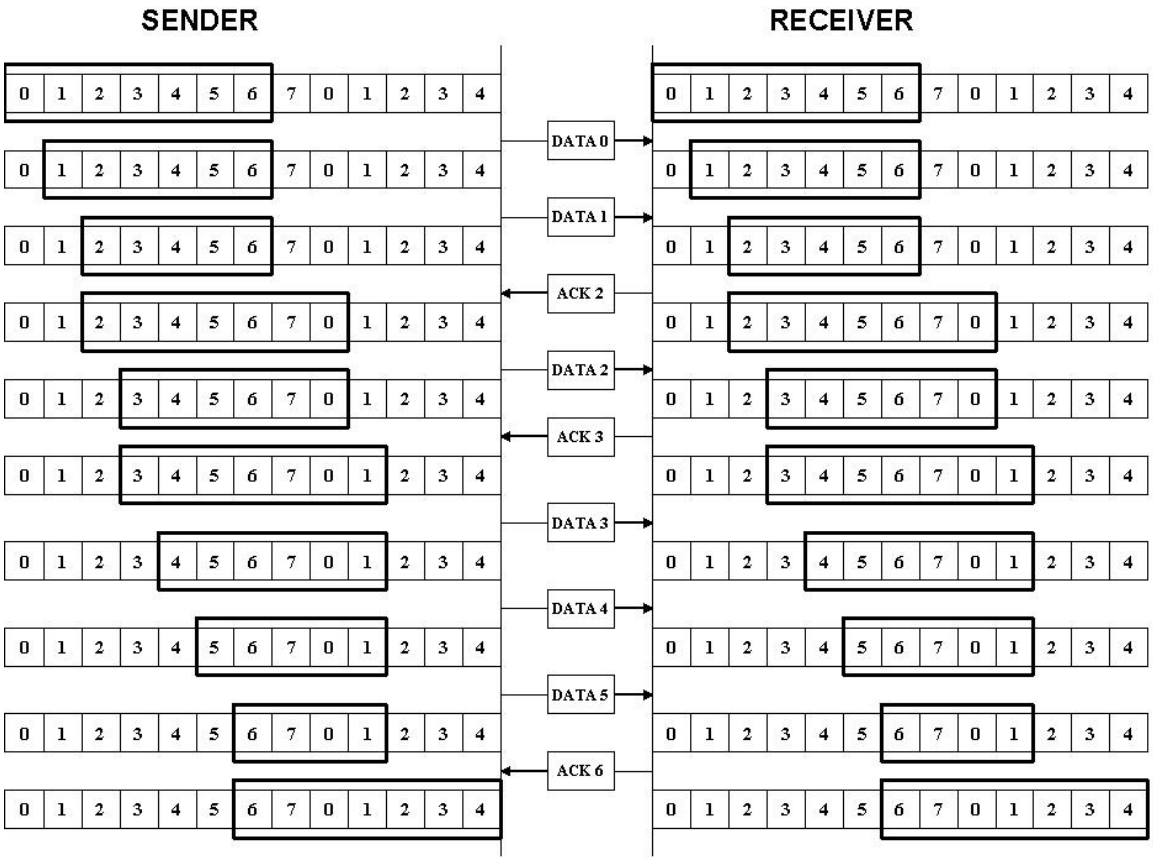


SENDER WINDOW:

At the beginning the sender‟s window contains n-1 frames. As frames are sent out the left boundary of the window moves inward, shrinking the size of the window. Once an ACK receives the window expands at the right side boundary to allow in a number of new frames equal to number of frames acknowledged by that ACK.



EXAMPLE:



ERROR CONTROL

Error control is implemented in such a way that every time an error is detected, a negative acknowledgement is returned and the specified frame is retransmitted. This process is called **automatic repeat request (ARQ).**

The error control is implemented with the flow control mechanism. So there are two types in error control. They are,

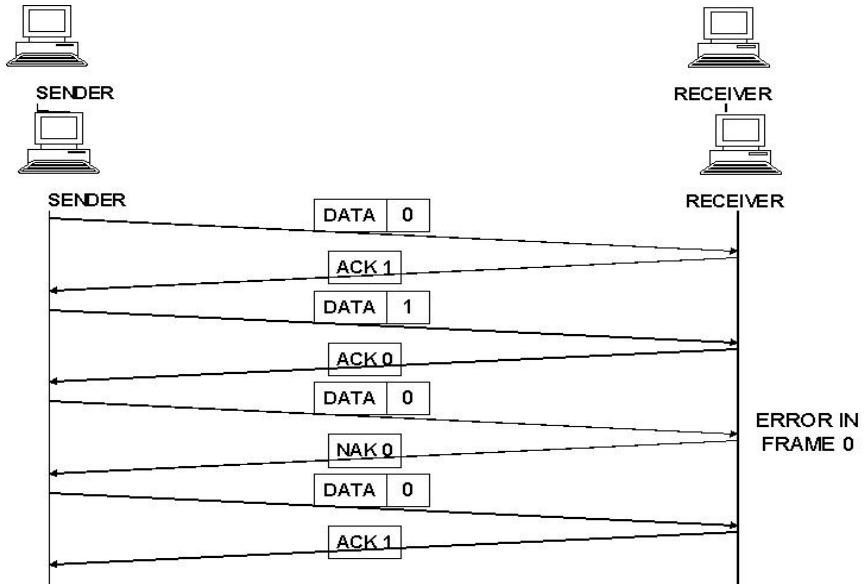
1. stop and wait ARQ
2. sliding window ARQ

STOP AND WAIT ARQ:

It is a form of stop and wait flow control, extended to include retransmission of data in case of lost or damaged frames.

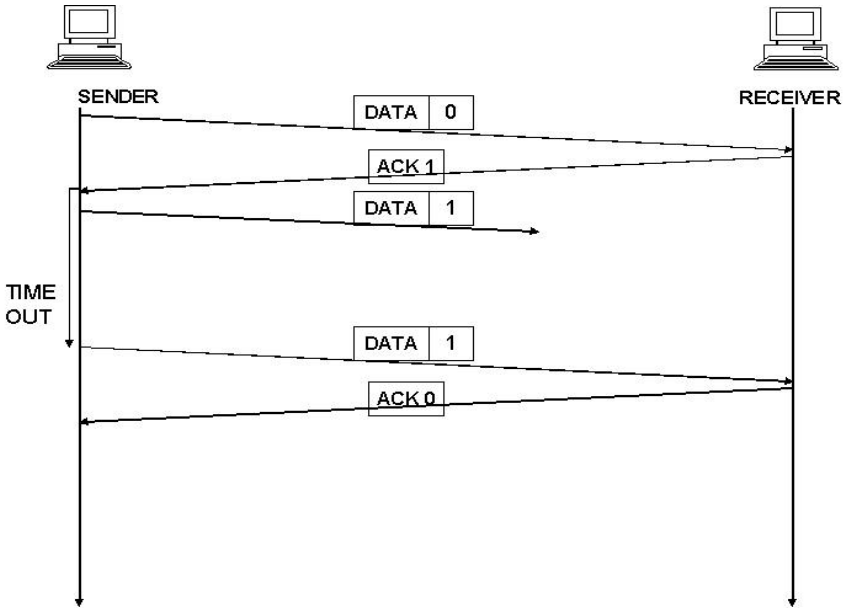
DAMAGED FRAME:

When a frame is discovered by the receiver to contain an error, it returns a NAK frame and the sender retransmits the last frame.



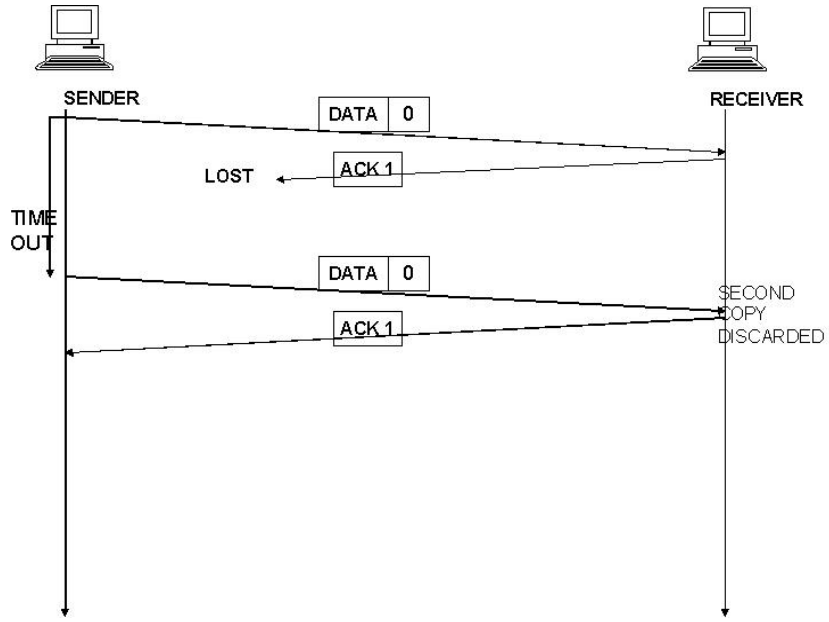
LOST DATA FRAME:

The sender is equipped with a timer that starts every time a data frame is transmitted. If the frame lost in transmission the receiver can never acknowledge it. The sending device waits for an ACK or NAK frame until its timer goes off, then it tries again. It retransmits the last data frame.



LOST ACKNOWLEDGEMENT:

The data frame was received by the receiver but the acknowledgement was lost in transmission. The sender waits until the timer goes off, then it retransmits the data frame. The receiver gets a duplicated copy of the data frame. So it knows the acknowledgement was lost so it discards the second copy.



SLIDING WINDOW ARQ

It is used to send multiple frames per time. The number of frame is according to the window size. The sliding window is an imaginary box which is reside on both sender and receiver side.

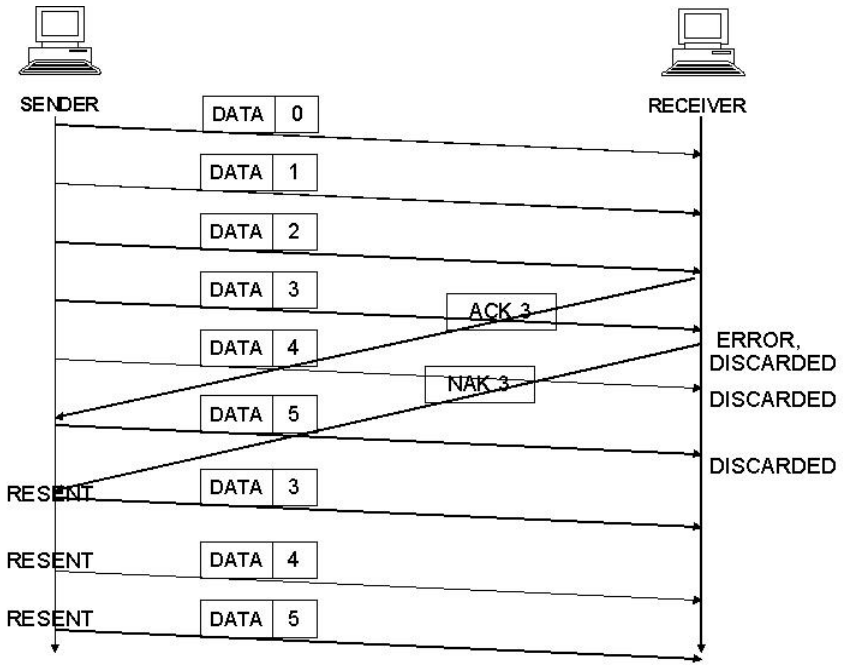
It has two types. They are,

1. go-back-n ARQ
2. selective reject ARQ

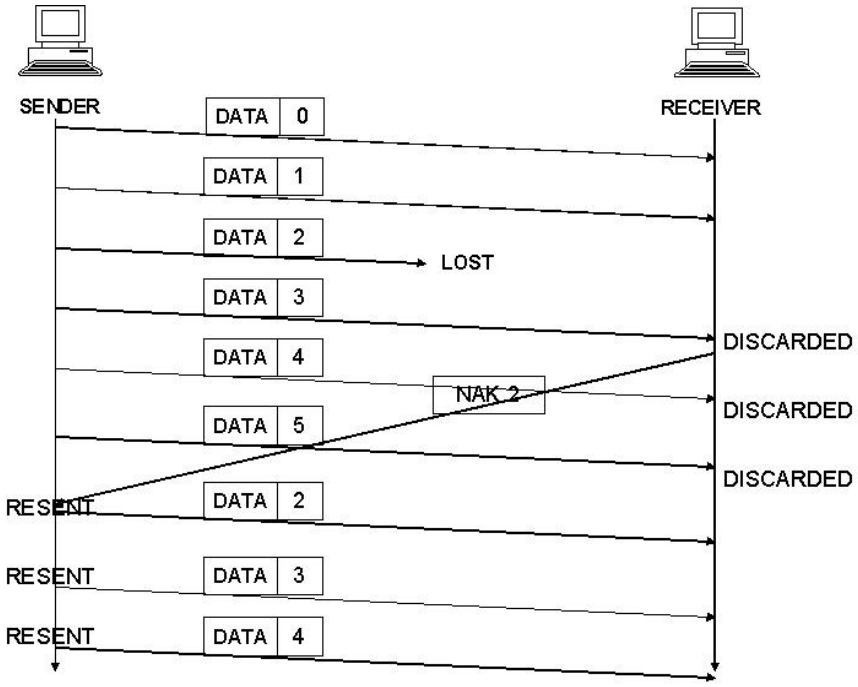
GO-BACK-N ARQ:

In this method, if one frame is lost or damaged, all frames sent since the last frame acknowledged or retransmitted.

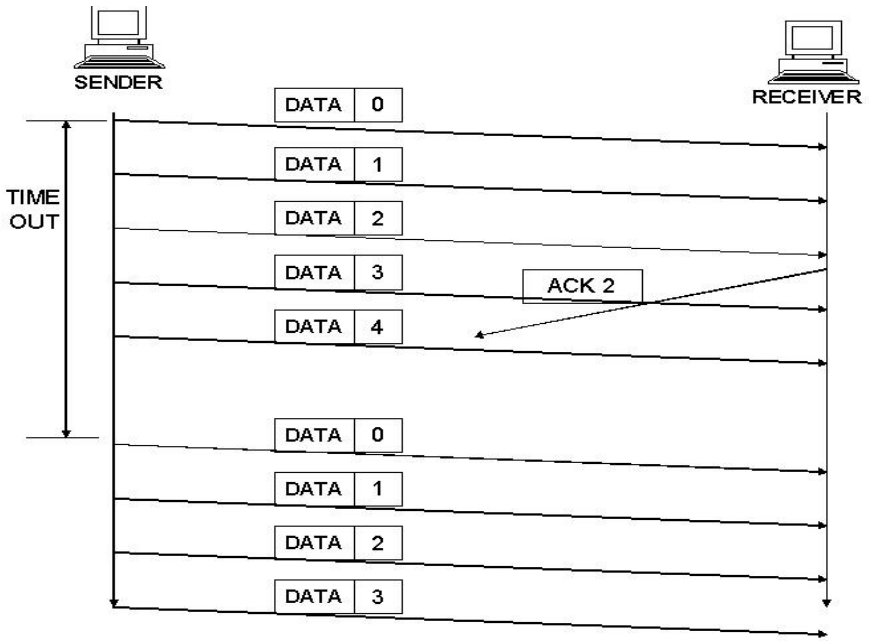
DAMAGED FRAME:



LOST FRAME:



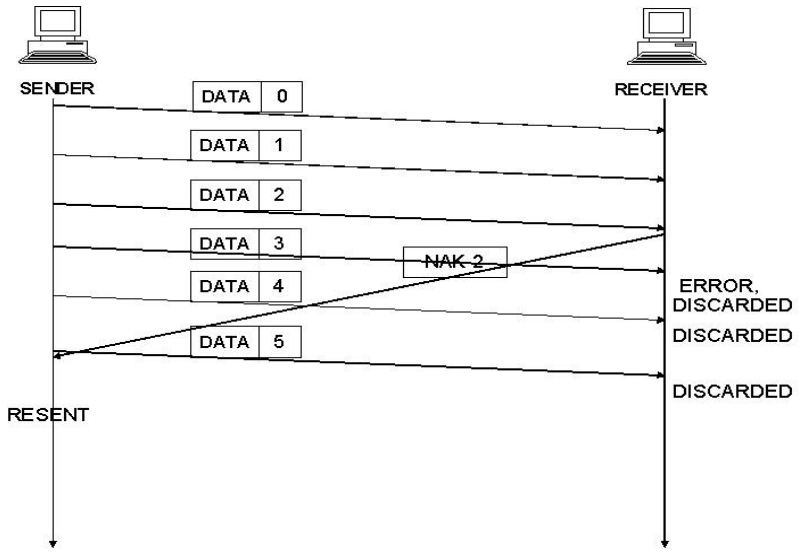
LOST ACK:



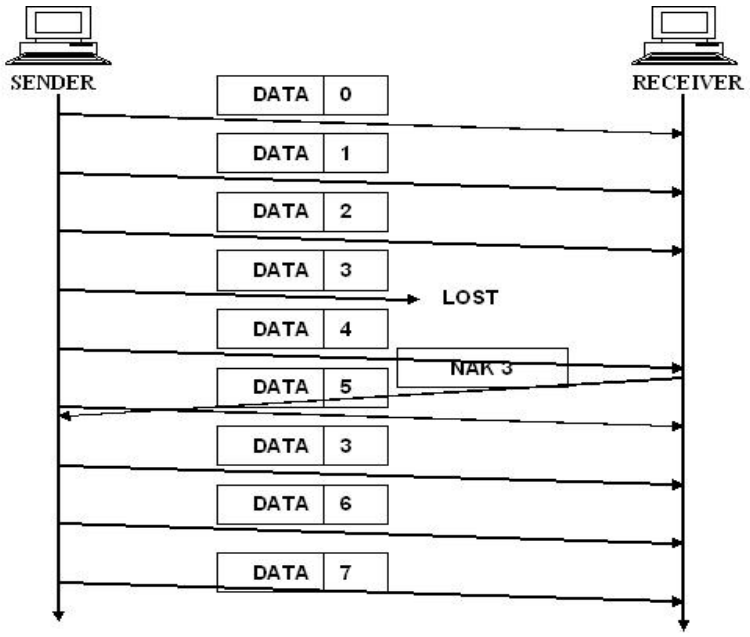
SELECTIVE REPEAT ARQ

Selective repeat ARQ re transmits only the damaged or lost frames instead of sending multiple frames. The selective transmission increases the efficiency of transmission and is more suitable for noisy link. The receiver should have sorting mechanism.

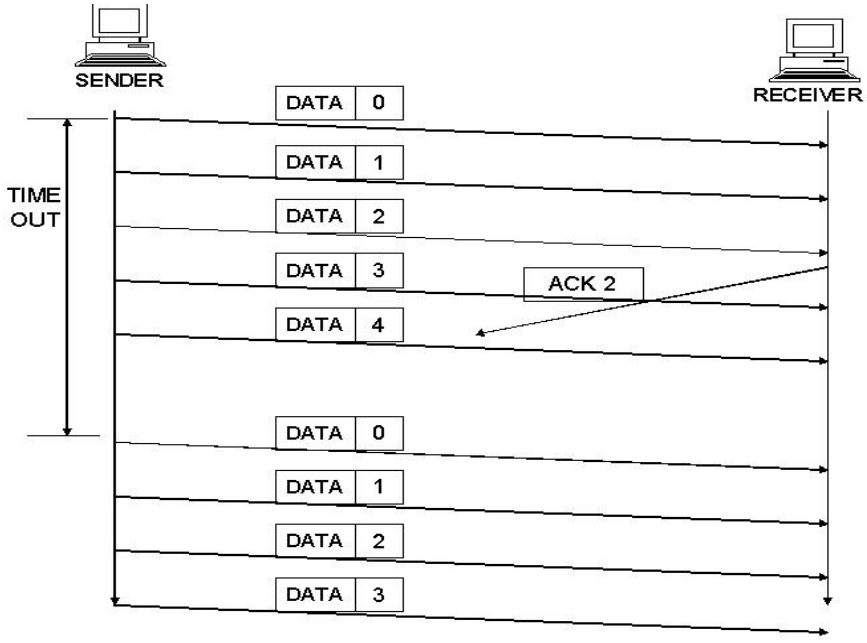
DAMAGED FRAME:



LOST FRAME



LOST ACK



FRAMING

The sream of bits are not advisible to mqaintain in networks. When an error occurs, then the entire stream have to retransmitted. To avoid this, the framing concept is used. In this, the stream of bits are divided into manageable bit units called frames. To achive, we are using several ways. They are,

1. Byte Oriented Protocols
2. Bit Oriented Protocols
3. Clock Based Protocols

1. BYTE ORIENTED PROTOCOLS:

Each frame is considered as a collection of bytes rather than a collection of bits. There are two approaches. They are,

1. Sentinel approach

In this approach it uses special characters called sentinel characters to indicate where frames start and end. This approach is called character stuffing because extra characters are inserted in the data portion of the frame.

Ex: 1. Binary Synchronous Communication (BISYNC)

2. Point to Point Protocol

2. Byte Count Approach

In this approach no of bytes in frame are counted and entered in the header. The COUNT Field specifies how many bytes are contained in the frame‟s body.

Ex: 1.Digital Data Communication Message Protocol

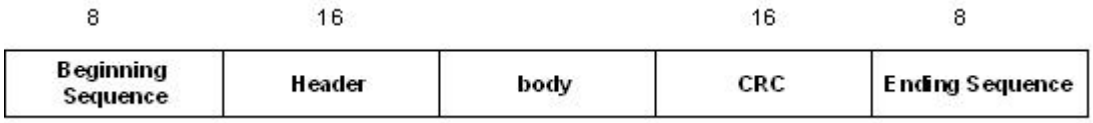
2. BIT ORIENTED PROTOCOLS:

It views the frames as a collection of bits. The Synchronous Data Link Control (SDLC) protocol developed by IBM is an example of a bit oriented protocol. It was later standardized by the ISO as the High Lever Data Link Control (HDLC)

HDLC – HIGH LEVEL DATA LINK CONTROL

It is a bit oriented data link protocol designed to support both half duplex and full duplex communication over point to point and multi point links.

FRAME FORMAT



HDLC denotes both the beginning and the end of a frame with the distinguished bit sequence 01111110. To guarantee that a special sequence does not appear in advertently anywhere else in the frame, HDLC uses a process called bit stuffing.

On the sending side, any time five consecutive 1s have been transmitted from the body of the message, the sender inserts a 0 before transmitting the next bit. On the receiver side, should five consecutive 1s arrive, the receiver makes its decision based on the next bit it sees. If the next bit is a 1, then one of the two things is true. Either this is the end of the frame or an error has been introduced. By looking at the next bit, it can conclude. If it sees a 0, then it is the end of frame. It else, then there must have an error and the whole frame has been discarded.

3. CLOCK BASED PROTOCOLS:

The Synchronous Optical NETwork (SONET) is one of the protocols using the clock based framing approach.

SONET:

It was developed by the ANSI for digital transmission over optical network. It addresses both the framing and encoding problems. A SONET frame has some special information to distinguish where the frame starts and ends.