

UNIT - II

DATA LINK LAYER

Data Link Layer: Design issues, framing, error detection and correction, parity, LRC, CRC, hamming code, elementary data link protocols- Stop-and-wait, sliding window protocols.

Medium Access sublayer: ALOHA, CSMA/CD

LAN Standards: IEEE 802.3, IEEE 802.11.

Design Issues

- The data link layer is responsible for moving frames from one node to the next.
- Functions of Data Link layer
 - Framing
 - Error detection & correction
 - Flow control(DL protocols)

- The Data Link Layer break the bit stream into discrete frames and compute the checksum for each frame.
- When a Frame arrives at the destination, the checksum is recomputed.
- If the newly computed checksum is different from one computed contained in the frame, the data link layer knows that an error has occurred and takes steps to deal with it.

Figure *Data link layer*

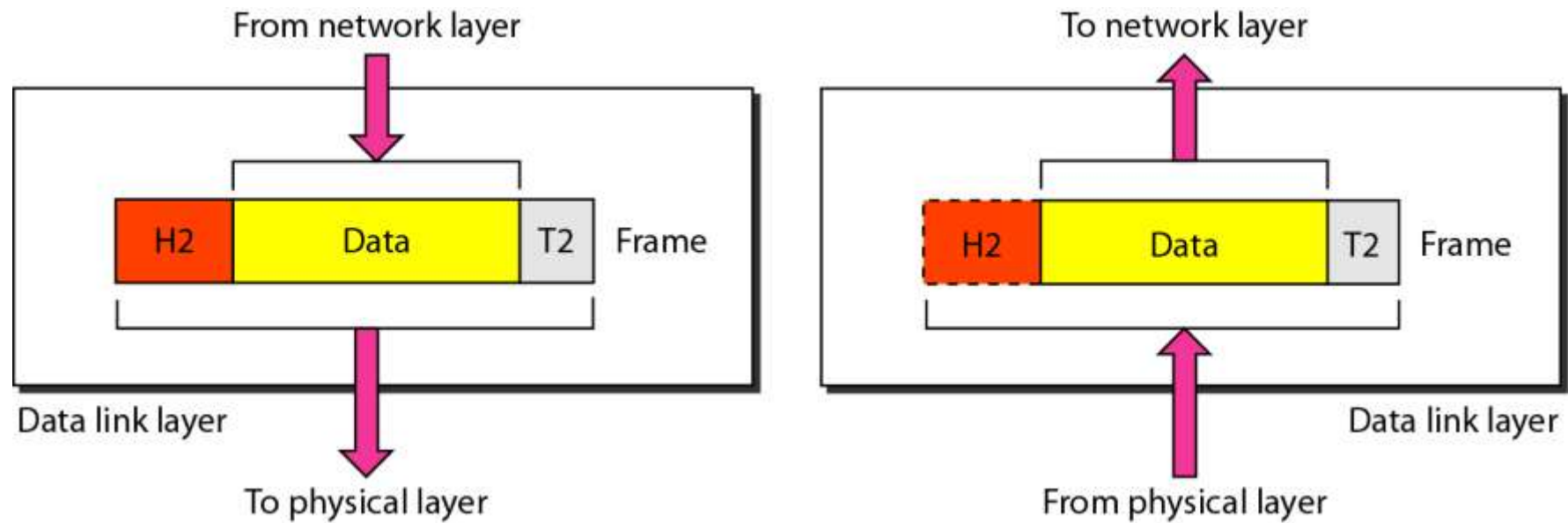
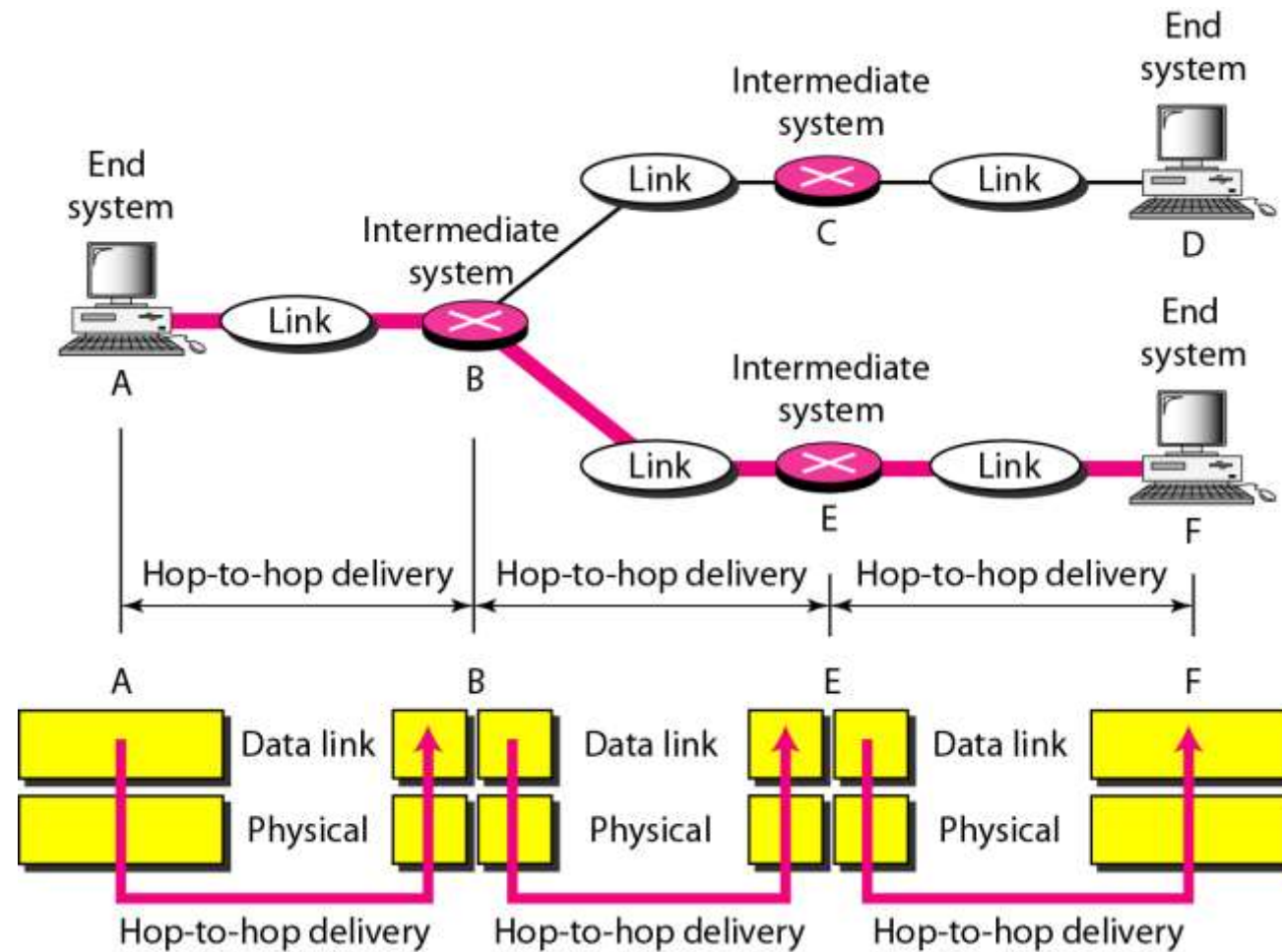
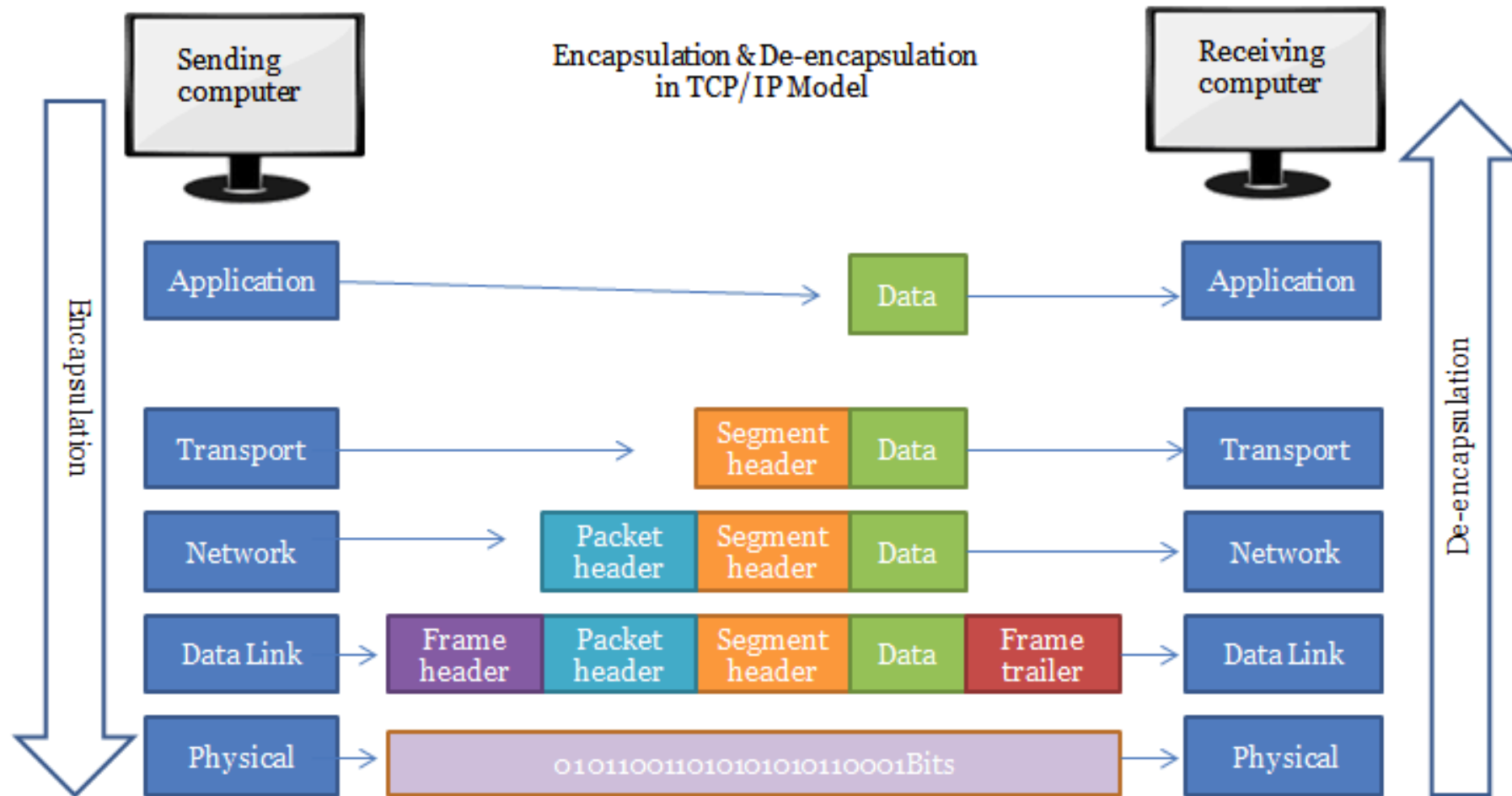


Figure *Hop-to-hop delivery*





- **Packets** are units of data in the *Network Layer* (IP in case of the Internet)
- **Frames** are units of data in the *Link Layer* (e.g. Wifi, Bluetooth, Ethernet, etc).
- The crucial difference between frame and packet is that frame is the serial collection of bits, and it encapsulates packets whereas packets are the fragmented form of data and it encapsulates segment.

BASIS FOR COMPARISON	FRAME	PACKET
Basic	Frame is the data link layer protocol data unit.	Packet is the network layer protocol data unit.
Associated OSI layer	Data link layer	Network layer
Includes	Source and destination MAC address.	Source and destination IP address.
Correlation	Segment is encapsulated within a packet.	Packet is encapsulated within a frame.

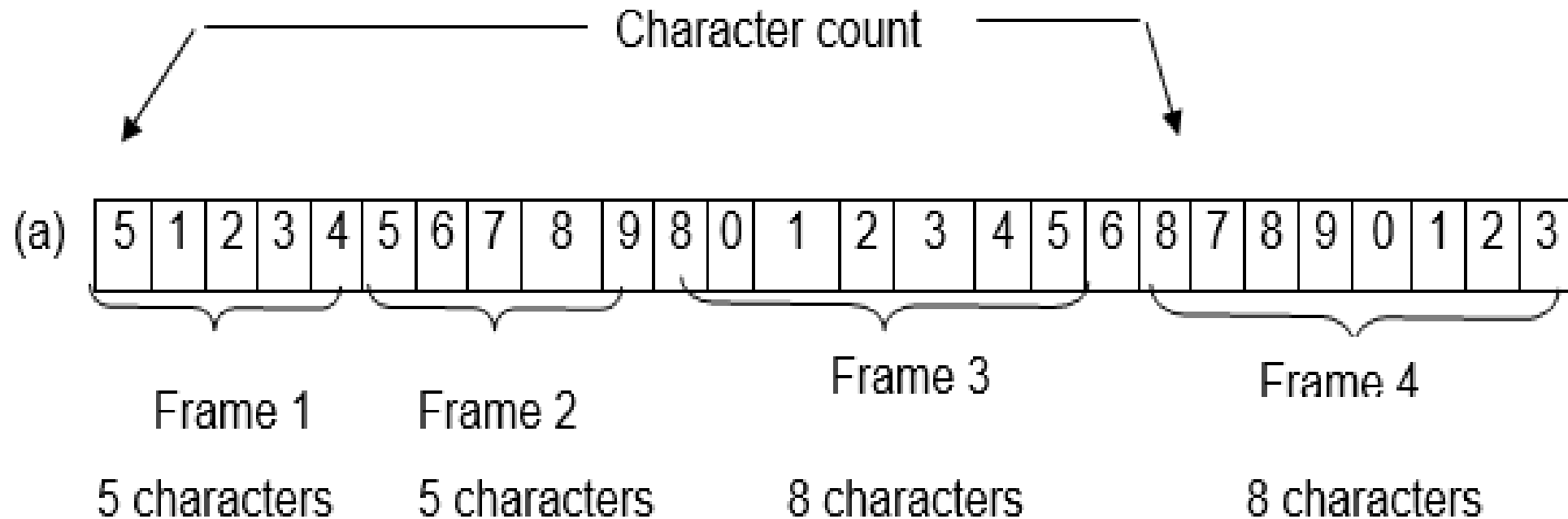
Framing Methods

- 1. CHARACTER COUNT METHOD**
- 2. STARTING AND ENDING CHARACTERS, WITH CHARACTER STUFFING**
- 3. STARTING AND ENDING FLAGS, WITH BIT STUFFING**

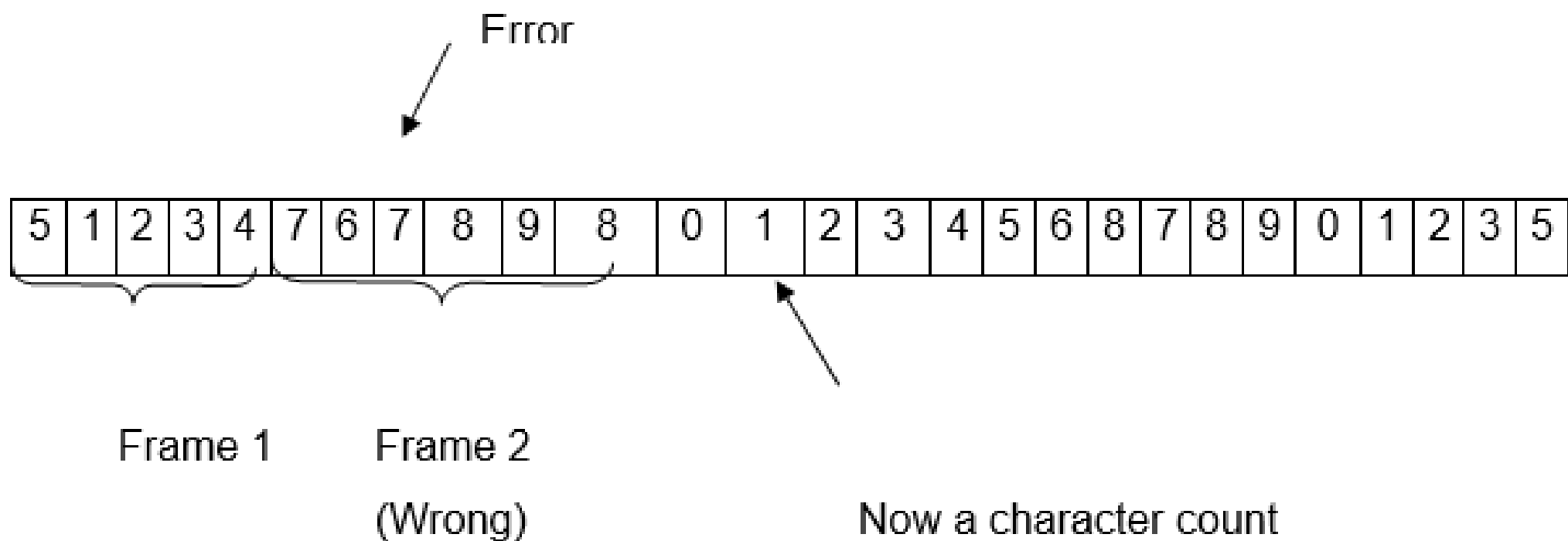
CHARATER COUNT METHOD

- In this method a field in the header will be used to specify the number of CHARACTERS in the frame.
- When data link layer at the destination sees the character count, it knows how many characters follow and hence where the end of the frame is.

A Character Stream



A Character Stream with one error



- The trouble with this algorithm is that the count can be garbed by a transmission error resulting the destination will get out of synchronization and will be unable to locate the start of the next frame.
- There is no way of finding where the next frame starts. For this reason this method is rarely used.

CHARATER STUFFING METHOD

❖ In this method each frame will start with a FLAG and ends with a FLAG.

✓ The starting flag

DLE STX ---- Data Link Escape Start of Text

✓ The ending flag

DLE ETX ---- Data link Escape End of Text.

Ex 1. The given Data ABRFCXDGJHKK12435ASBGXRR

The Data will be sent as

DLE STX ABRFCXDGJHKK12435ASBGXRR DLE ETX



starting flag



Data



ending flag

Let the code for DLE is 01010101

let the data – 0000111000110111001010101010101

code for DLE



-- what is the problem?

When ever the receiver found a DLE in the data another DLE will be stuffed immediately in the data.

- When ever the receiver found a DLE in the data another DLE will be stuffed immediately in the data.
- When the receiver receives two DLEs successively it will de stuff one DLE(thinking that it is stuffed) and data will be read.

Ex 2. The given Data

ASHGTRDXZBNHG DLE %\$#54378

The data will be sent as

DLE STX ASHGTRDXZBNHG DLE DLE %\$#54378 DLE ETX

Dis Adv:

1. 24 bits are stuffed.
2. Transmission delay.

BIT STUFFING METHOD

- In this method every frame will start with a **flag**

01111110

Ex. The given data is

01111000011111110101001111110 01111101100

The data will be sent as

01111110 01111000011111110101001111110 01111101100

FLAG



DATA

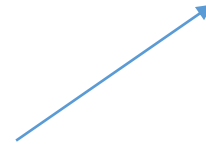


EX:

DATA : 100000111100110001000111101111110001111

01111110 100000111100110001000111101111110001111

?



In the data if there are **FIVE** consecutive ONE 's are found then a ZERO will be stuffed automatically.

After the receiver receives the data it will check for 05 successive ones.

Case i) if the sixth bit is a zero it will be de stuffed.

Case ii) if the 6th bit is a one, observes the 7th bit. If it is a zero then it is a Flag.

Case iii) if the 6th bit is a one, observes the 7th bit. If it is a one then it's an error.(07 successive 1's will not appear in any case)

Exercise

Find out how the data will be transmitted by sender for the below.

data -- 1 0 1 0 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1

1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 1 1

ABCDEFGHIJNDERTSDLEFJWQIR342590478DLEDLE918274

ERROR DETECTION & CORRECTION

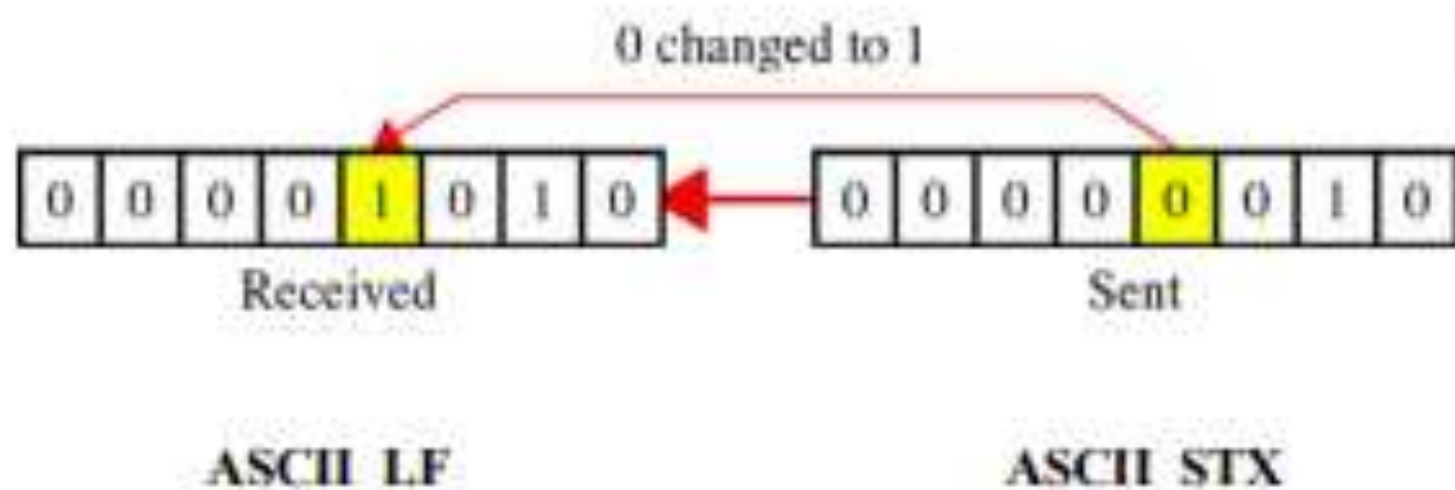
PARTY CHECK,VRC,LRC,CRC,HAMMING CODE.

Types of Errors

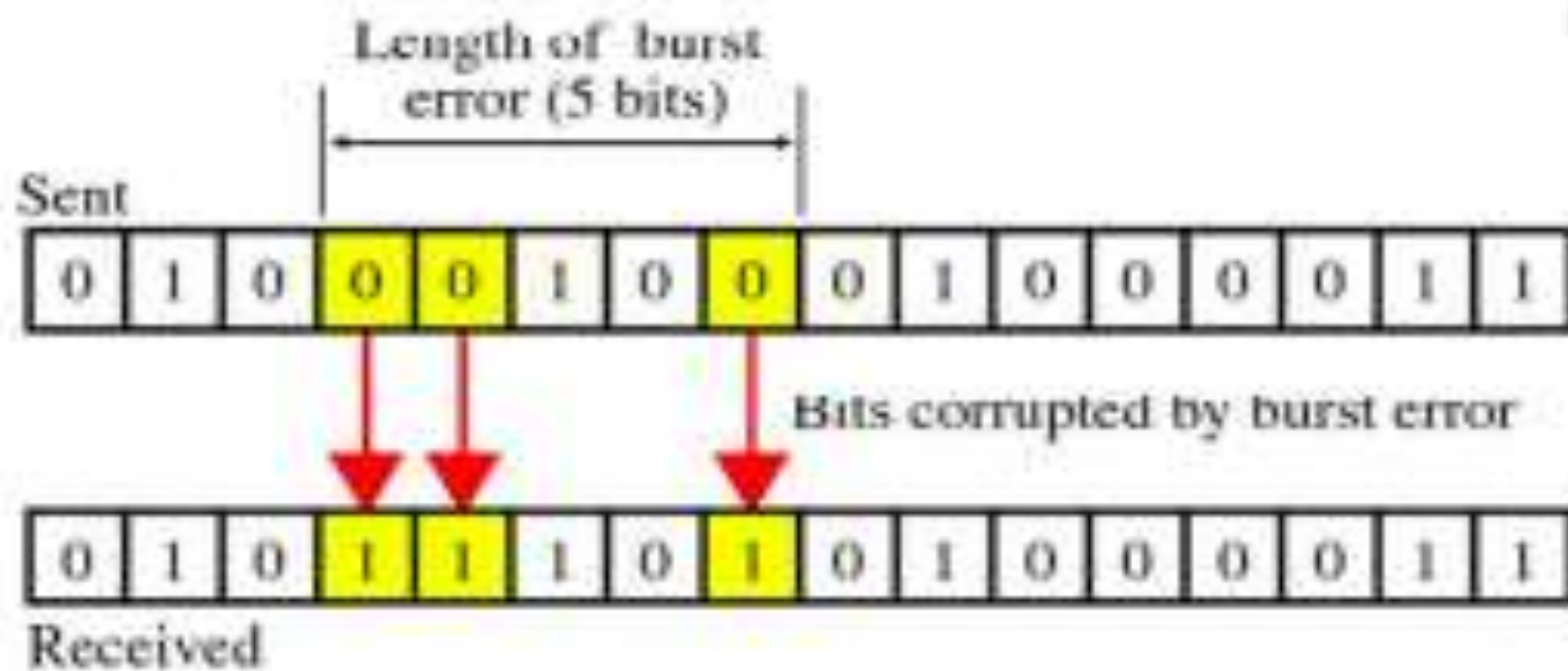
- A network system that cannot guarantee that the data received are completely identical to the data transmitted is essentially useless.
- Reliable systems must have a mechanism for detecting and correcting the errors.
- There are two types of errors: –
 1. Single -bit error: one bit is changed from 0 to 1 or from 1 to 0
 2. Burst error: multiple bits are changed.

Figure 5-2

Single-Bit Error



Burst Error of Length Five



ERROR – CORRECTING AND DETECTING CODES

- Network designers have developed two basic strategies for dealing with errors.
- One way is to include enough redundant information along with each block of data sent, to enable the receiver to deduce what the transmitted data must have been .

- The other way is to include only enough redundancy to allow the receiver to deduce that an error occurred, but not which error, and have it request a retransmission.
- The former strategy uses **Error – correcting codes** and the latter uses **Error- detecting codes**.

- The **Error – correcting** and **Error- detecting** methods are

1. PARITY METHOD
2. LRC METHOD (Longitudinal redundancy check)
3. CRC METHOD (Cyclic redundancy check)
4. HAMMING CODE METHOD

PARITY METHOD

Character code	even parity	odd parity
1100100	1100100 <u>1</u>	1100100 <u>0</u>
0011000	0011000 <u>0</u>	0011000 <u>1</u>

the transmitter & receiver agree on common parity.

- Let the transmitter & receiver agreed on Even parity.

Transmitted data -- 10101111

Received data --- 10101111

- Let the transmitter & receiver agreed on odd parity.

Transmitted data -- 10101110

Received data --- 10101110

Ex 1 Transmitted data -- 10101111 (Even parity)

Received data --- 10101011 (odd parity) ---- error occurred.

Ex 2 Transmitted data -- 11100000 (Odd parity)

Received data --- 10100000 (even parity) ---- error occurred.

Ex 3 Transmitted data -- 10101111

Received data --- 10111101

Error occurred.

- This technique can not detect an even number of bit errors (two, four, six and so on).
- If even number of bits flip during transmission, then receiver can not catch the error.

Longitudinal Redundancy Check

LRC

Longitudinal Redundancy Check(LRC)

- The frame is viewed as a block of characters arranged in 2-dimensions.
- To each character is appended a parity bit.
- In addition a parity bit is generated for each bit position across all characters i.e., an additional character is generated in which the l^{th} bit of the character is parity bit for the l^{th} bit of all other characters in the block. This can be expressed mathematically using exclusive OR(+) operation. The parity bit at the end of each character of row parity

- This can be expressed mathematically using exclusive OR(+) operation.
- The parity bit at the end of each character of row parity

$$R_j = b_{1j} + b_{2j} + \dots + b_{nj}$$

Where R_j = Parity bit of jth character

b_{ij} = ith bit in jth character


$$C_i = b_{i1} + b_{i2} + \dots + b_{in}$$

Where C_i = ith bit of parity check character

m = number of characters in a frame

- In this format the parity bits at the end of each character are referred to as the Vertical Redundancy Check(VRC) and
- the Parity check character is referred to as the Longitudinal Redundancy Check(LRC).

	<u>bit</u> 1	<u>bit</u> 2		<u>bit</u> n	Parity <u>bit</u>
Character 1	b_{11}	b_{21}		b_{n1}	R_1
Character 2	b_{12}	b_{22}		b_{n2}	R_2
Character m	b_{1m}	b_{2m}		<u>b_{nm}</u>	R_m
Parity check character	C_1	C_2		<u>C_n</u>	<u>C_{n+1}</u>


Char 1 - 1 0 1 1 1 0 0 0 0  VRC

Char 2 - 1 1 1 1 0 0 1 1

Char 3 - 0 0 1 0 1 0 0 0

Char 4 - 1 0 1 0 1 0 1 0

Char 5 - 0 1 0 1 0 1 0 1

Char - 1 0 0 1 1 1 0 0  LRC

Char 1 - 1 0 1 1 1 0 1

Char 2 - 1 0 0 1 0 0 1

Char 3 - 1 0 1 0 1 0 1

Char 4 - 1 1 1 0 1 0 1

Char 5 - 0 1 1 0 0 1 1

Find out the VRC & LRC for the above 05 characters by considering ODD parity.

VRC ---- 0, 0, 1, 0, 1

LRC ---- 1 1 1 1 0 0 0

Char 1 1 0 1 1 1 1 0

Char 2 0 1 0 0 0 1 1

Char 3 1 0 1 1 0 1 0

Char 4 0 0 0 0 1 1 0

Char 5 1 0 0 0 0 0 1

Char 6 0 0 0 0 0 0 1

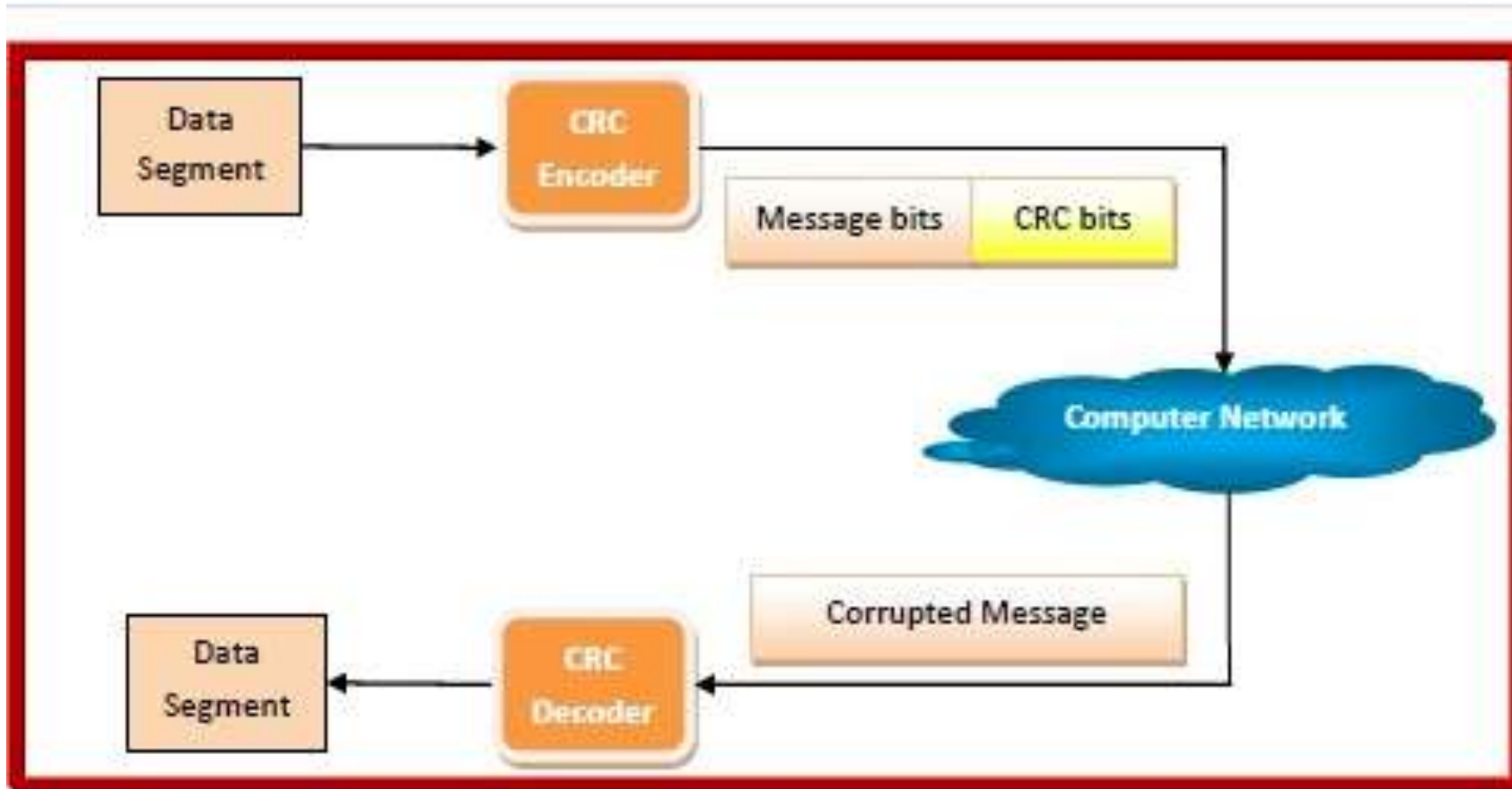
Char 7 0 1 1 1 0 0 0

1. Find out VRC & LRC for the above data by considering ODD parity.
2. The received check sum is 01000100. Find out in which character the error occurred.

Cyclic Redundancy Check

CRC

Cyclic Redundancy Check(CRC)



CRC Method

1. The frame is expressed in the form of a Polynomial Form.

$$x^7 + x^6 + x^4 + x^3 + x + 1.$$

2. Both the sender and receiver will agree upon a generator polynomial $G(x)$ in advance. $G(x) = x^4 + x + 1$

3. Let 'r' be the degree of $G(x)$. Append 'r' zero bits to the lower – order end of frame now it contains $m+r$ bits.

4. Divide the bit string by $G(x)$ using Mod 2 operation.

5. Transmitted frame $[T(x)] = \text{frame} + \text{remainder}$

6. Divide $T(x)$ by $G(x)$ at the receiver end. If the result is a zero, then the frame is transmitted correctly.

DATA - $x^7 + x^6 + x^4 + x^3 + x + 1$.

How to convert?

--- Where ever the power is there keep 1 no power is Zero.

power of	7	6	5	4	3	2	1	0
	1	1	0	1	1	0	1	1

Generator --- $x^4 + x + 1$

power of	4	3	2	1	0
	1	0	0	1	1

Ex.

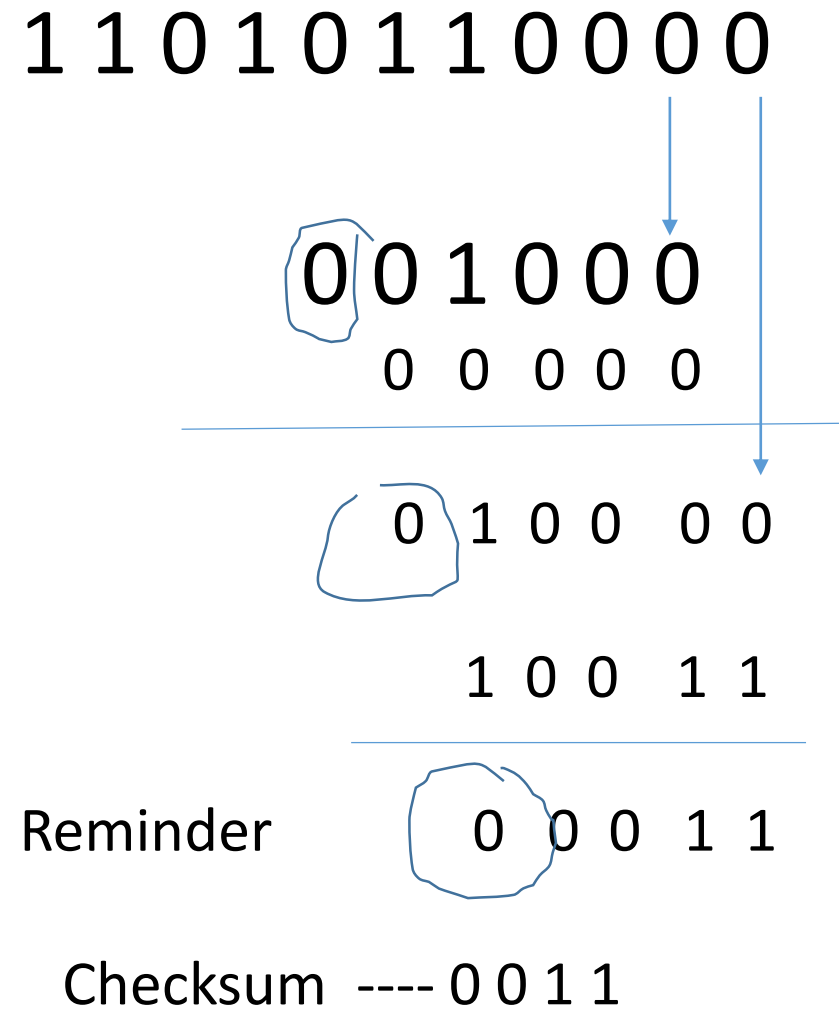
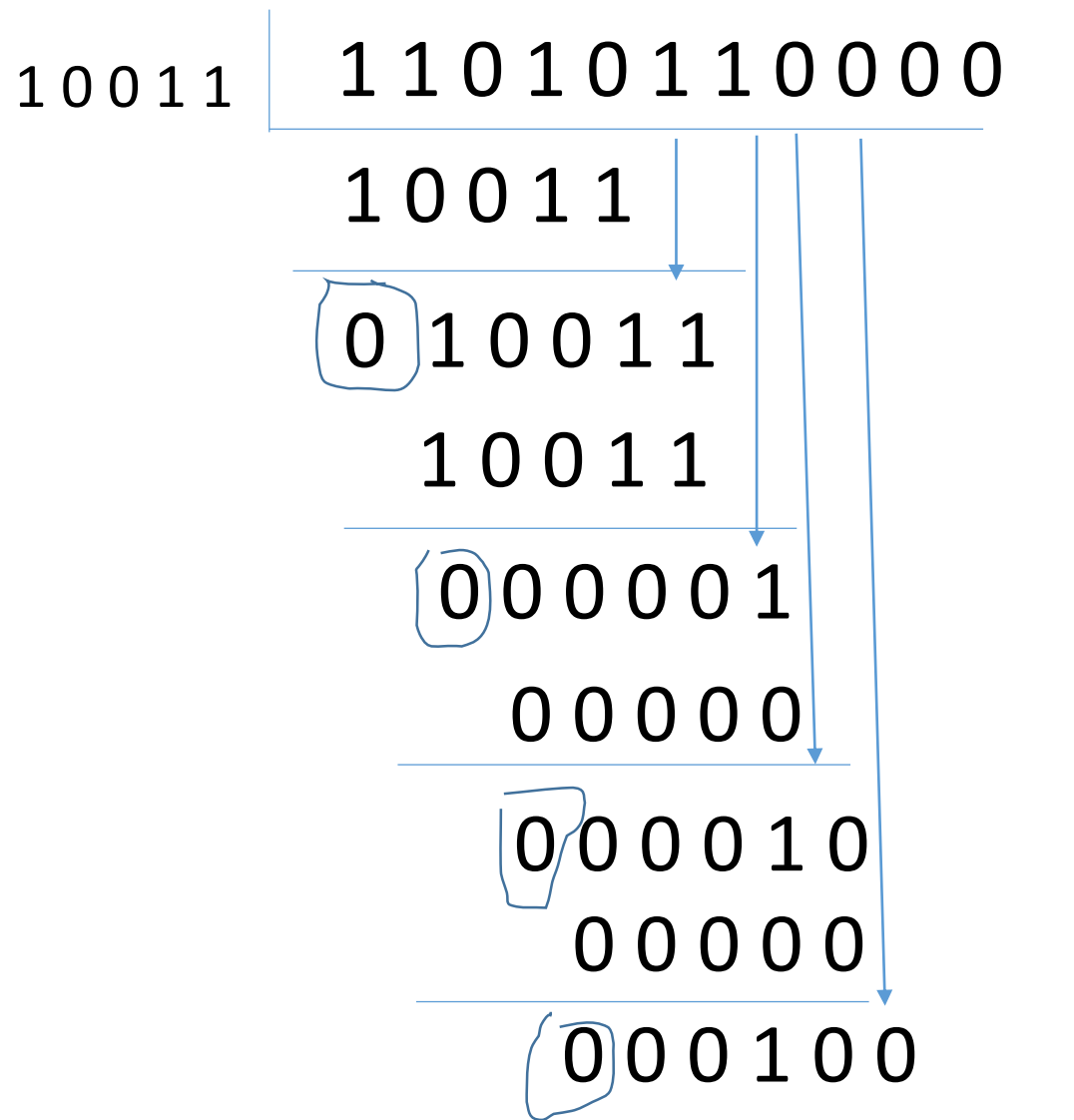
Frame : 1101011(7 bits)

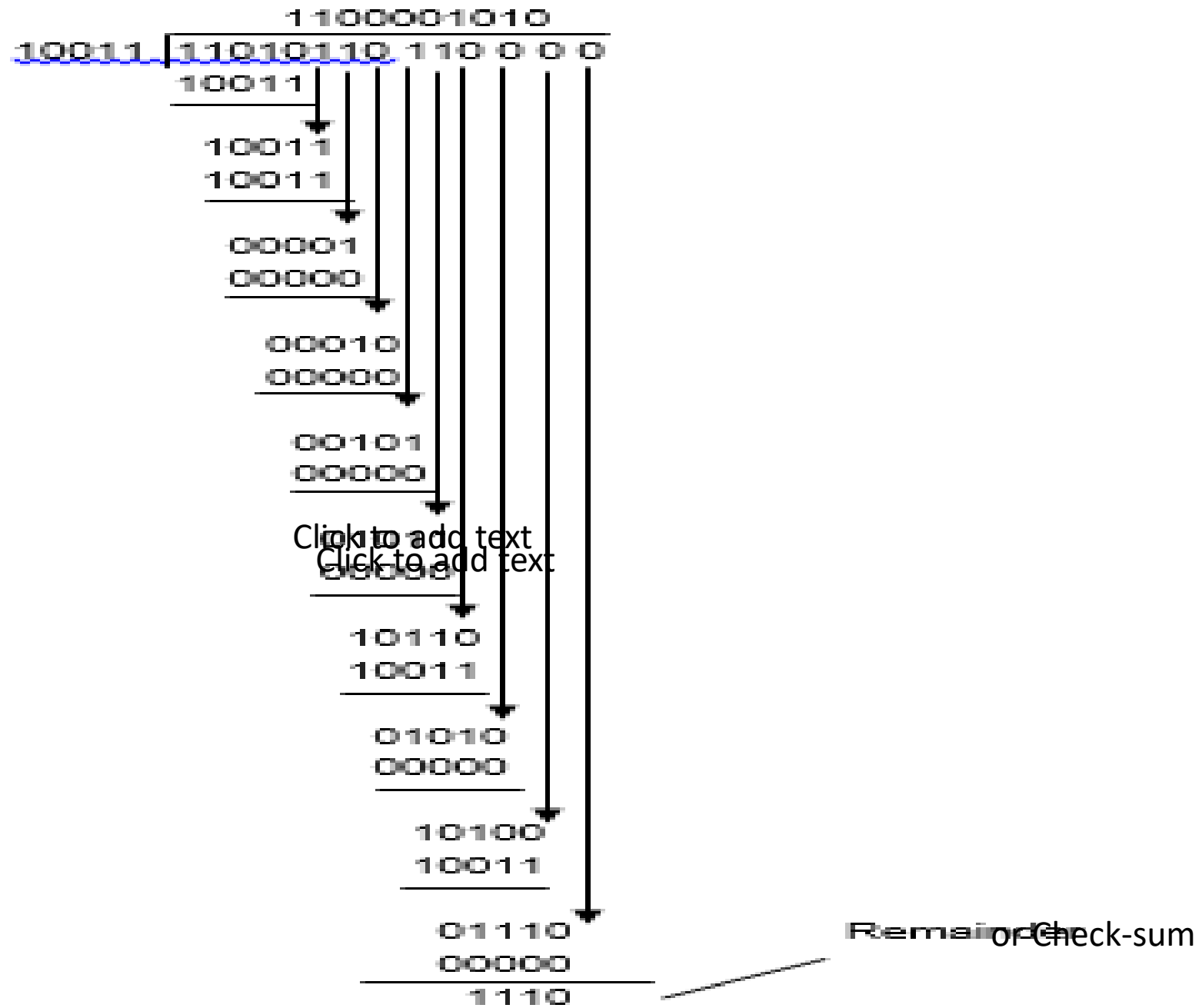
Generator : 10011

Append 'r' zero bits

$$r = 4$$

Message after appending 4 zero bits : 11010110000(11 bits)





Transmitted frame: 11010110111110

CRC

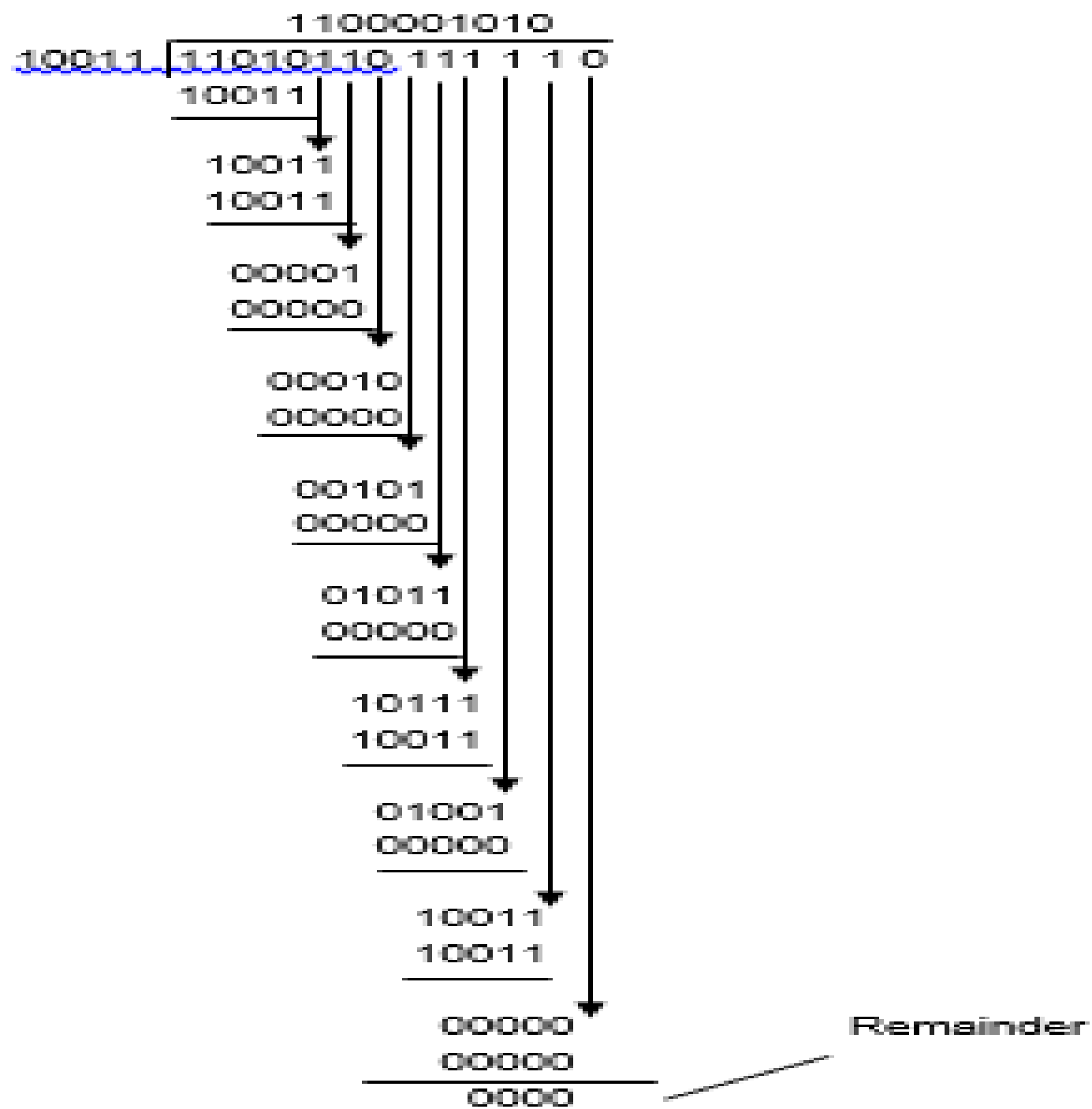
Original Frame(data) --- 1101011011

Reminder(checksum) -- 1110

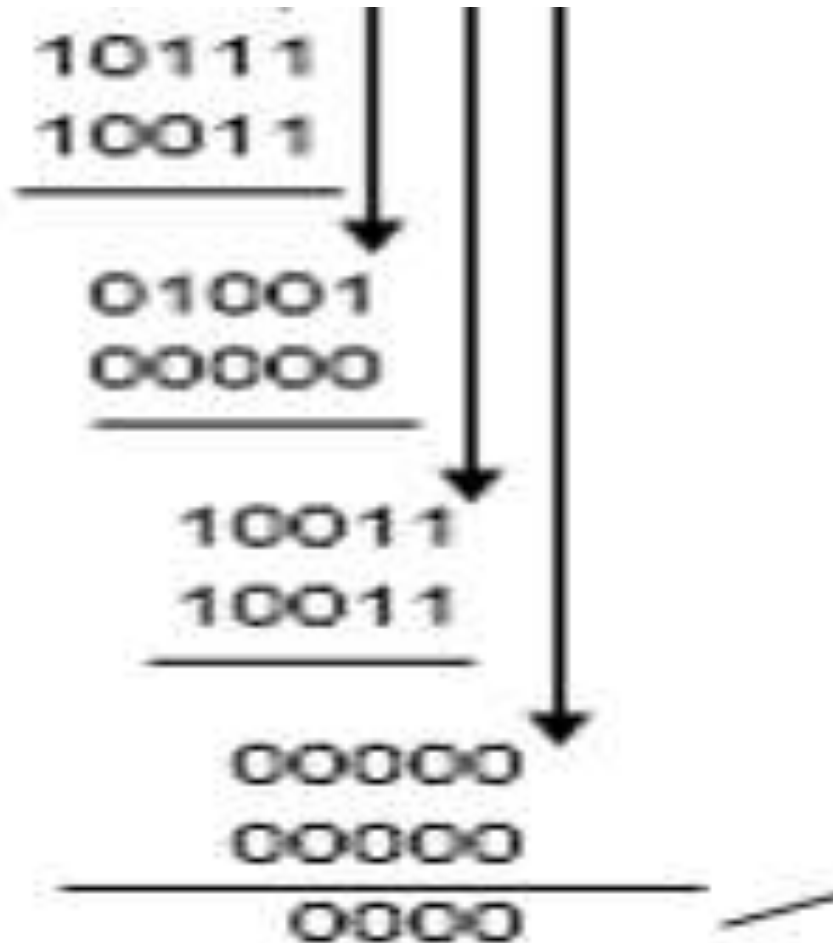
Transmitted Data --- data + reminder

1101011011 1110

- The transmitted data is divided again at receiver by the same generator.
- If the reminder is zero then the received data is correct.
- Otherwise error occurred.



Transmitted data 1101011011 11^{Transmitted as 0}10



10011 } 1101011011110 0

Diagram illustrating the division of 11010110111100 by 10011:

$$\begin{array}{r}
 100110 \\
 \underline{10011} \\
 000010 \\
 \underline{00000} \\
 00010 \text{ reminder}
 \end{array}$$

Reminder is not zero.
Hence there is an error.

HAMMING CODE

HAMMING CODE

- Hamming codes provide another method for error correction.
- Error bits, called Hamming bits, are inserted into message bits at random locations.
- It is believed that the randomness of their locations reduces the odds that these Hamming bits themselves would be in error.

- This is based on a mathematical assumption that because there are so many more message bits compared with Hamming bits, there is a greater chance for a message bit to be in error than for a Hamming bit to be wrong.
- Determining the placement and binary value of the Hamming bits can be implemented using hardware, but it is often more practical to implement them using software

- The number of bits in a message (M) are counted and used to solve the following equation to determine the number of Hamming bits (H) to be used:

$$2^H \geq M + H + 1$$

Where M – denotes the number of data bits

H -- denotes the number of hamming bits

Once the number of Hamming bits is determined, the actual placement of the bits into the message is performed. It is important to note that despite the random nature of the Hamming bit placements, the exact sample placements must be known and used by both the transmitter and receiver. Once the Hamming bits are inserted into their positions, the numerical values of the bit positions of the logic 1 bits in the original message are listed. The equivalent binary numbers of these values are added in the same manner as used in previous error methods by discarding all carry results. The sum produced is used as the states of the Hamming bits in the message. The numerical difference between the Hamming values transmitted and that produced at the receiver indicates the bit position that contains a bad bit, which is then inverted to correct it.

Ex. The given data

00010001100101(14- bits)

The number of hamming codes

$$2^H \geq M + H + 1$$

H = ? M = 14

to satisfy this equation the minimum H should be 5

i.e. Five hamming code bits need to be incorporated in data bits.

0 0 0 1 0 0 0 1 1 0 H 0 H 1 H 0 H 1 H

- Find the positions where binary 1's(or 0's) are present in the data to be transmitted. (including Hamming bit position)
- Add using mod 2 operation (Ex-OR).
- The result will give the Hamming code at the transmitter end.

1's place in data

Binary equivalent

2

0 0 0 1 0

6

0 0 1 1 0

11

0 1 0 1 1

12

0 1 1 0 0

16

1 0 0 0 0

Hamming code



1 0 0 1 1

0 0 0 1 0 0 0 1 1 0 H 0 H 1 H 0 H 1 H



The data transmitted is 0 0 0 1 0 0 0 1 1 0 0 1 0 1 0 0 1 1 1

At the receiver, hamming code will be calculated.

If both code are same no error. Otherwise an error occurred.

Lets assume that the bit at 13th place is inverted to 1 (from zero) while transmitting.

0 0 0 1 0 1 0 1 1 0 0 1 0 1 0 0 1 1 1

1 0 0 1 0 0 0 1 1 0 H 0 H 1 H 0 H 1 H

1's place in data

Binary equivalent

2

0 0 0 1 0

6

0 0 1 1 0

11

0 1 0 1 1

12

0 1 1 0 0

13

0 1 1 0 1

16

1 0 0 0 0

Hamming code



1 1 1 1 0

Hamming code at Transmitter: 1 0 0 1 1

Hamming code at Receiver : 1 1 1 1 0

Hence error occurred. How to find out the error?

1 0 0 1 1

1 1 1 1 0

0 1 1 0 1

Perform Mod 2 operation
On both the codes }

- The resultant code after Mod 2 operation

0 1 1 0 1

Decimal equivalent for this code is 13.

Hence an error is occurred at 13th place.

Invert the bit at place 13th.

DATA LINK PROTOCOLS

STOP & WAIT PROTOCOL,
PIGGYBACKING
SLIDING WIDOW PROTOCOL

Unrestricted Simplex Protocol

❑ In this protocol the following assumptions are made

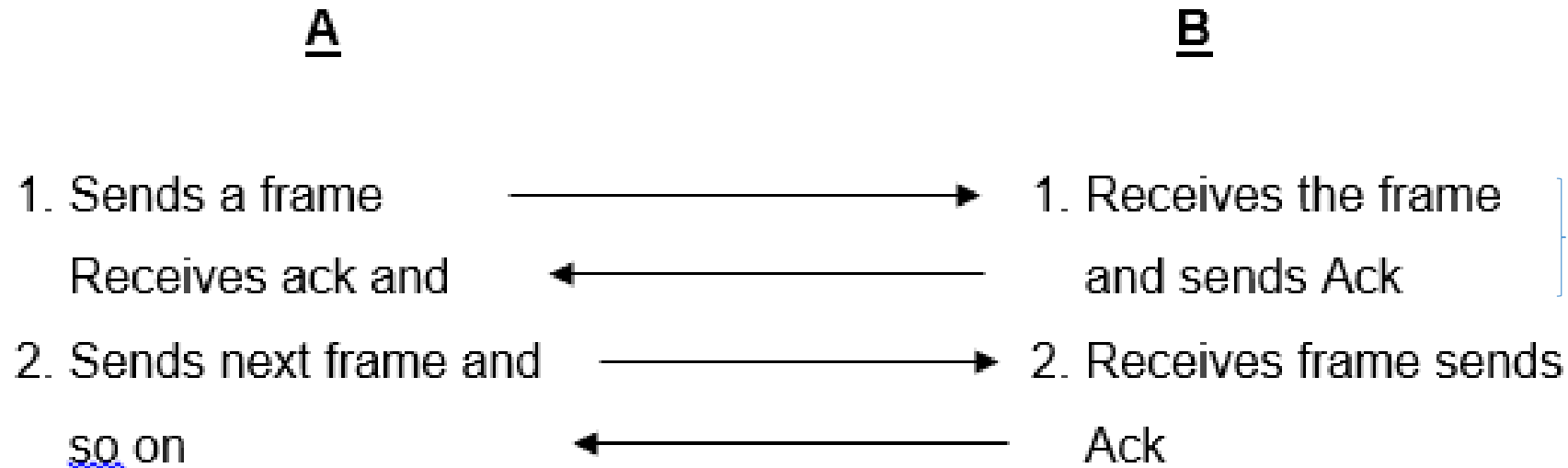
- Data transmission is simplex i.e. transmitted in one direction only.
- Both transmitting and receiving network layers are ready.
- Processing time is ignored.
- Infinite buffer space is available.
- An error free channel.

This is an unrealistic protocol, which has a nickname “Utopia”.

A simplex stop and wait protocol

The following assumptions are made

- a) Error free channel.
- b. Data transmission simplex

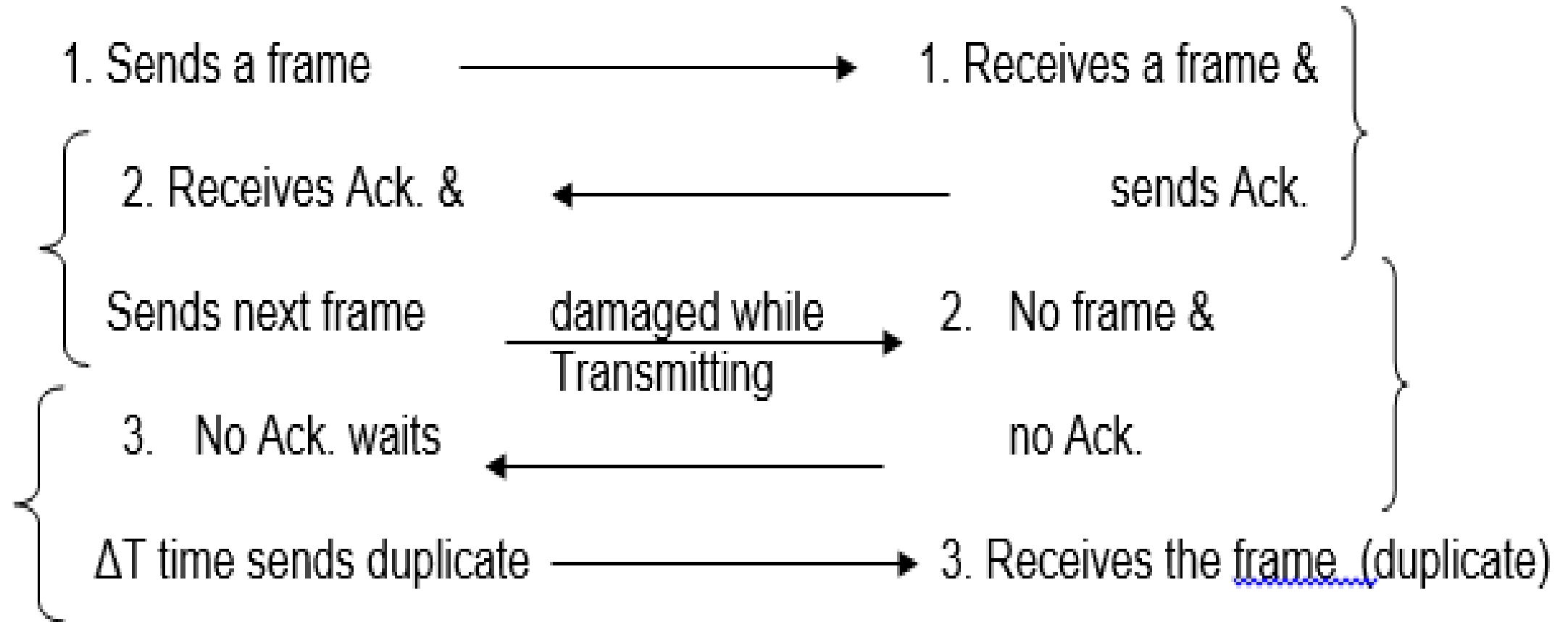


Since the transmitter waits for Δt time for an Ack this protocol is called stop and wait protocol.

3. A simplex protocol for a noisy channel

A

B



When this protocol fails?

A

1. Sends a frame

2. Ack. is not received

Waits some time & sends duplicate

B

1. Receives & }
Sends Ack.

Receives duplicate frame



At this situation protocol fails

- because the receiver receives a duplicate frame and there is no way to find out whether the receiver frame is original or duplicate.
- Hence, the protocol fails at this situation.

What is required?

- It is needed some way for the Receiver to distinguish a frame and a duplicate.
- To achieve this sender has to put a sequence number in the header of each frame it sends.
- The Receiver can check the sequence number of each arriving frame to see if it is a new frame or a duplicate.

- Here a question arises?

-- What is the minimum number of bits needed for the sequence number?

The ambiguity is between a frame and its successor.

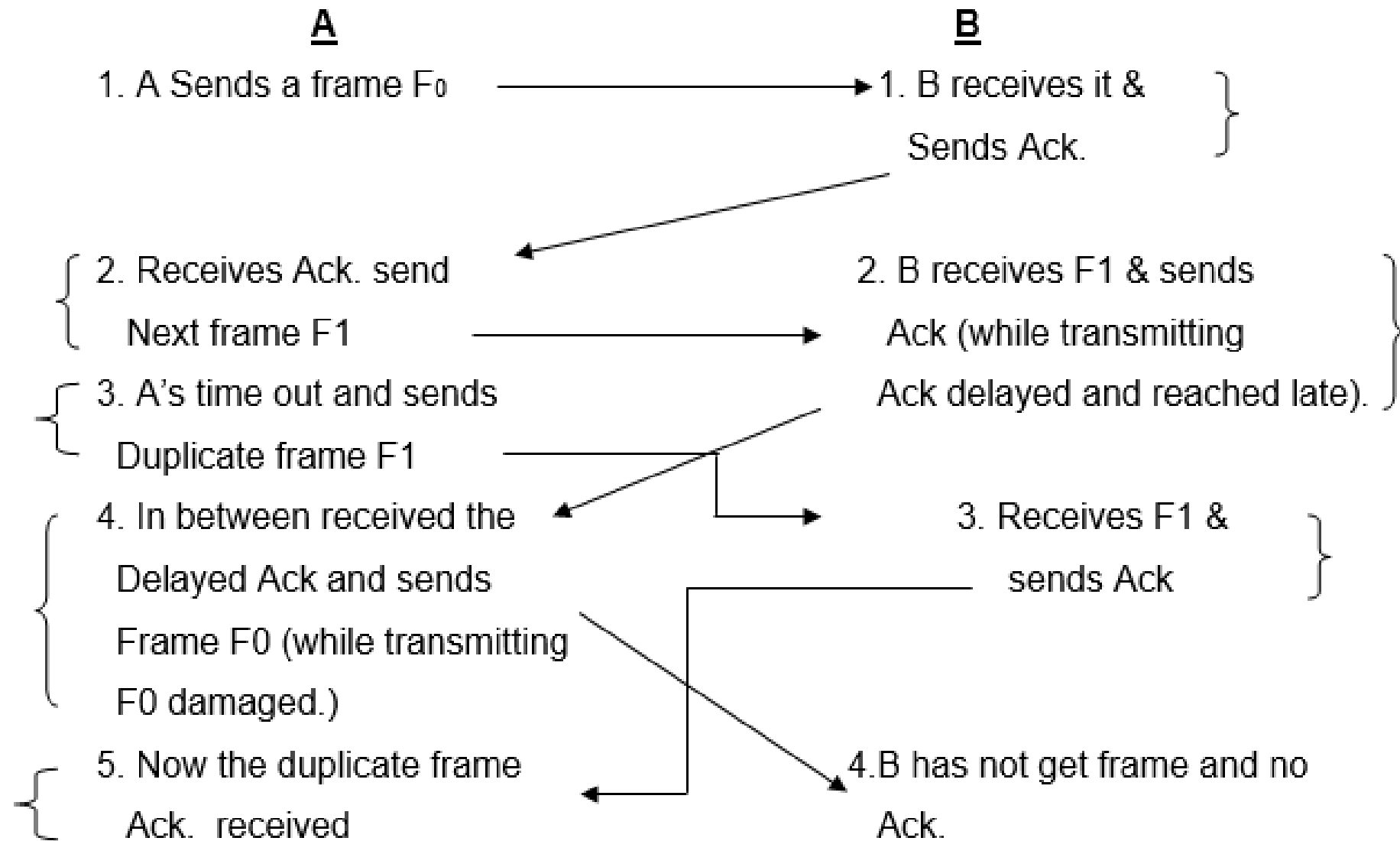
Hence a

1-bit sequence number (0 or 1) is therefore sufficient. At each instant of time, the receiver expects a particular sequence number next.

- Any arriving frame containing wrong sequence number is rejected as a duplicate.
- When a frame containing the correct sequence number arrives, it is accepted, passed to the network layer and then expected sequence number is incremented
- i.e. 0 becomes 1 and one becomes 0.

- Protocols in which a sender waits for a positive acknowledgement before advancing to the next data item are often called PAR (positive ack with retransmission) or ARQ (automatic repeat request).

When this protocol fails?



PIGGY BACKING

- In most practical situations there is a need of transmitting data in both directions.
- This can be achieved by full duplex transmission.
- If this is done we have two separate physical circuits each with a 'forward' and 'reverse' channel. In both cases, the reverse channel is almost wasted.
- To overcome this problem a technique called **piggy backing** is used.

- The technique of temporarily delaying outgoing acknowledgements, so that they can be hooked onto the next outgoing data frame is known as **piggy backing**.

- Complication with piggybacking?
- How long the data link layer should wait ?
- Longer than the sender's timeout period, the frame will be retransmitted, defeating the whole purpose of having acknowledgements.

SLIDING WINDOW PROTOCOLS

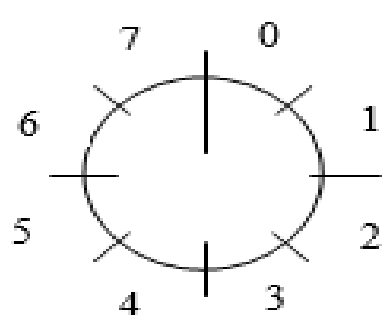
- In all sliding window protocols, each outbound frame contains a sequence number, ranging from 0 up to some maximum(n).
- The sequence number fits nicely in an n -bit field.
- The stop-and-wait sliding window protocol uses $n=1$, restricting the sequence numbers to 0 and 1, but more sophisticated versions can use arbitrary n .

- The essence of all sliding window protocols is
 - that at any instant of time, the sender maintains a set of sequence numbers corresponding to frames it is permitted to send.
- These frames are said to fall within the sending window.

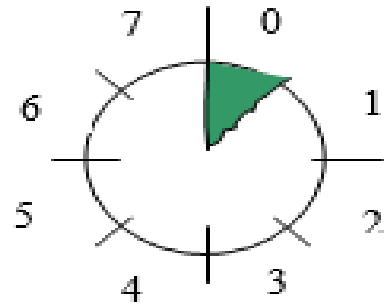
- Similarly the receiver also maintains a receiving window corresponding to the set of frames it is permitted to accept.
- The sender's window and the receiver's window need not have the same lower and upper limits, or even have the same size.
- In some protocols they are fixed in size, but in others they can grow or shrink as frames are sent and received.

- The sequence numbers within the sender's window represent frames sent but as yet not acknowledged.
- Whenever a new packet arrives from the network layer, it is given the next highest sequence number, and the upper edge of the window is advanced by one.
- When an acknowledgement comes in, the lower edge is advanced by one.
- In this way the continuously maintains a list of unacknowledged frames.

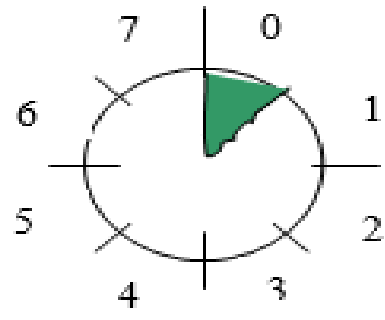
Sender



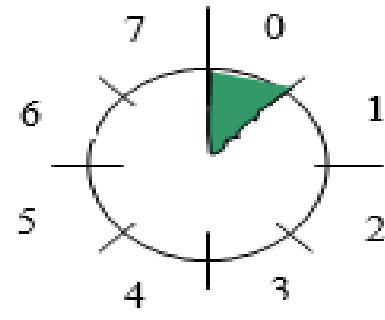
Receiver



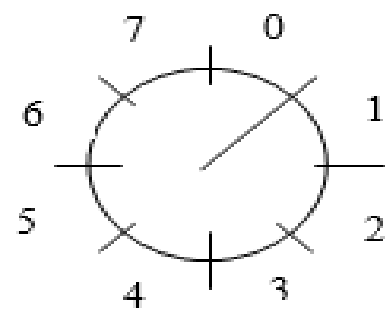
(a)



(b)



(c)

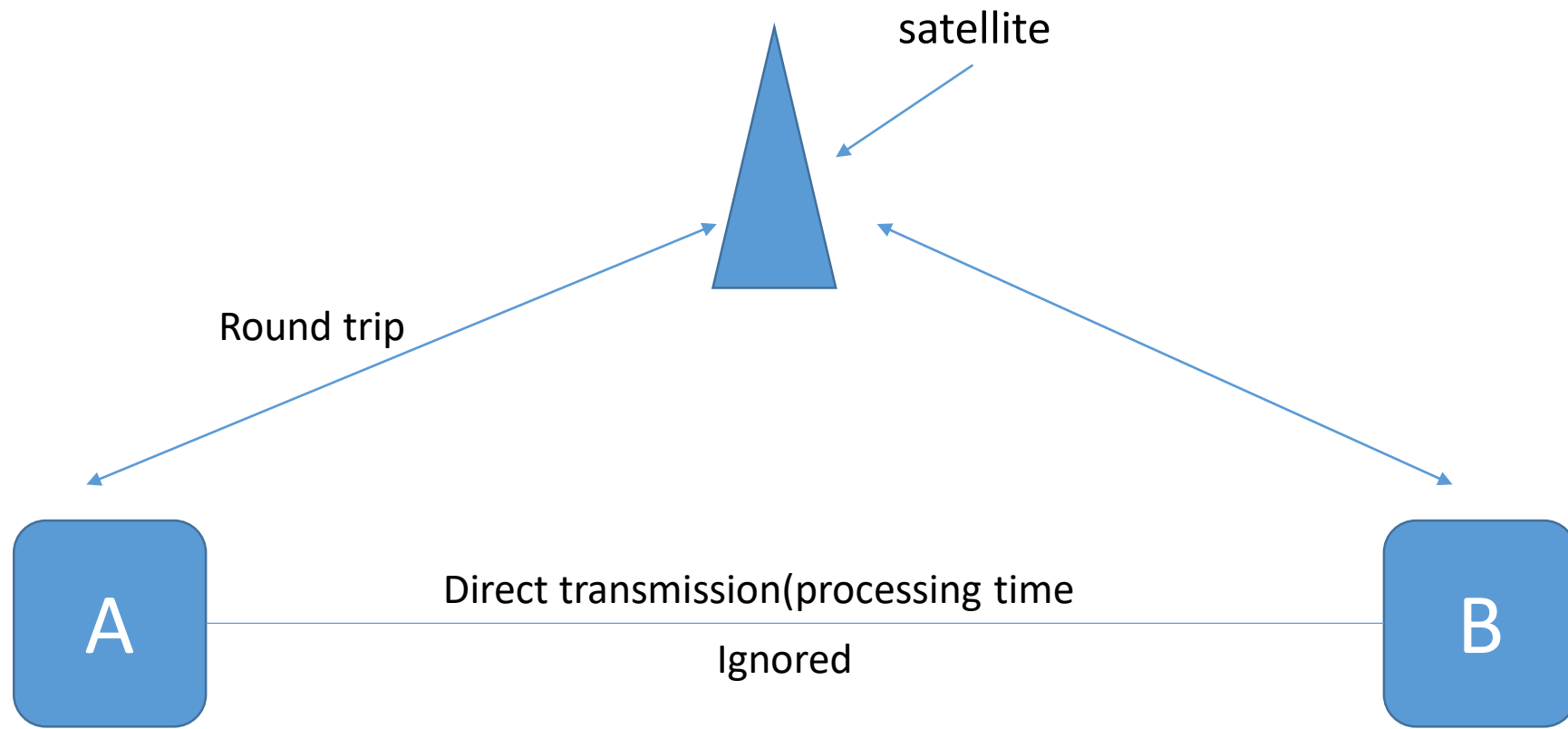


(d)

(a) Initially (b) After the first frame has been sent (c) After the first frame has been received. (d) After the first acknowledgement has been received.

PIPELINING

1. Upto now we made the assumption that the transmission time required for a frame to arrive at the receiver plus the transmission time for the acknowledgement to come back is negligible.
2. Sometimes this is not true, when there is a long round trip propagation time is there.
3. In these cases round trip propagation time can have important implications for the efficiency of the bandwidth utilization.



A → satellite → B → satellite → A

Let the channel capacity = b bps

Let the frame size = l bits

Let the round trip delay = R secs

To send one frame the time required will be l/b secs

Due to round trip delay the time taken will be $(l/b + R)$ Sec = $\left[\frac{l + R b}{b} \text{ Sec} \right]$

The channel utilization = $\frac{\frac{l}{b}}{\frac{l + R b}{b}}$

Consider the below example.

Let the channel capacity $b = 50 \text{ Kbps}$.

Round trip propagation delay $= 500 \text{ ms}$

Frame size $= 1000 \text{ bits}$

Without considering the round trip propagation delay

For one frame the time taken will be $= 1000/50 \text{ k}$
 $= 20 \text{ ms}$

Let the round trip propagation delay = 500 m secs

Considering the round trip propagation delay 500 m secs

For one frame the time taken will be = 500 ms + 20 ms

$$= 520 \text{ ms}$$

$$\text{The channel utilization} = \left[\frac{20}{520} \right] * 100 = 4 \%$$

96% of channel time is wasted.

How to overcome?

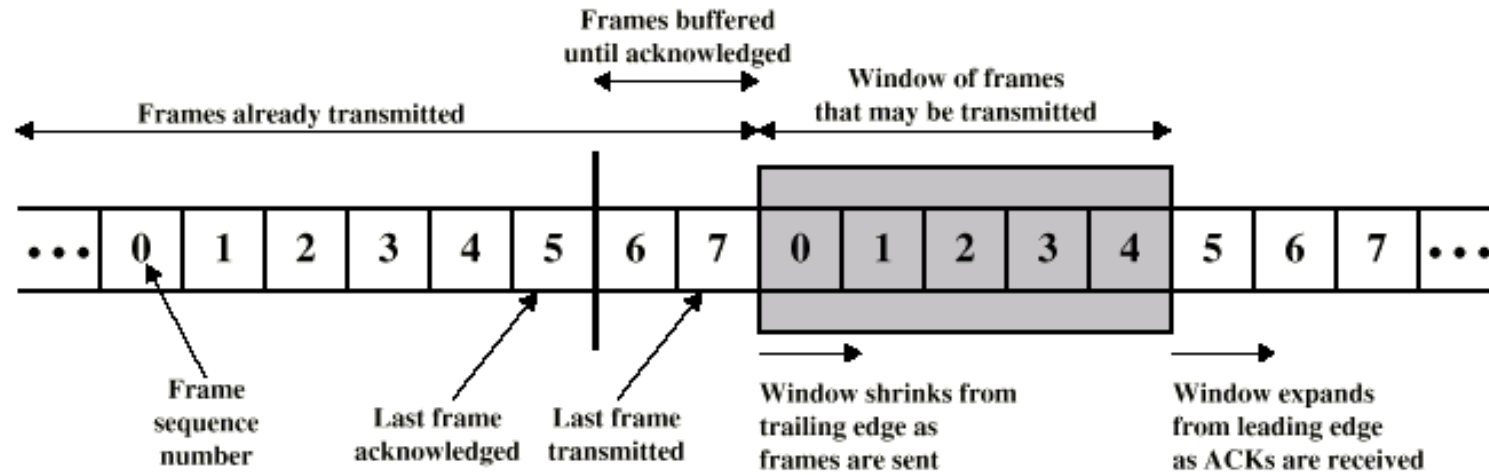
PIPELINING??

- In this technique, the sender is allowed to transmit up to 'w' frames before blocking, instead of just 1.
- With an appropriate choice of w the sender will be able to continuously transmit frames for a time equal to the round trip transmit time without filling up the window.
- In the above example w would be at least 26 frames.

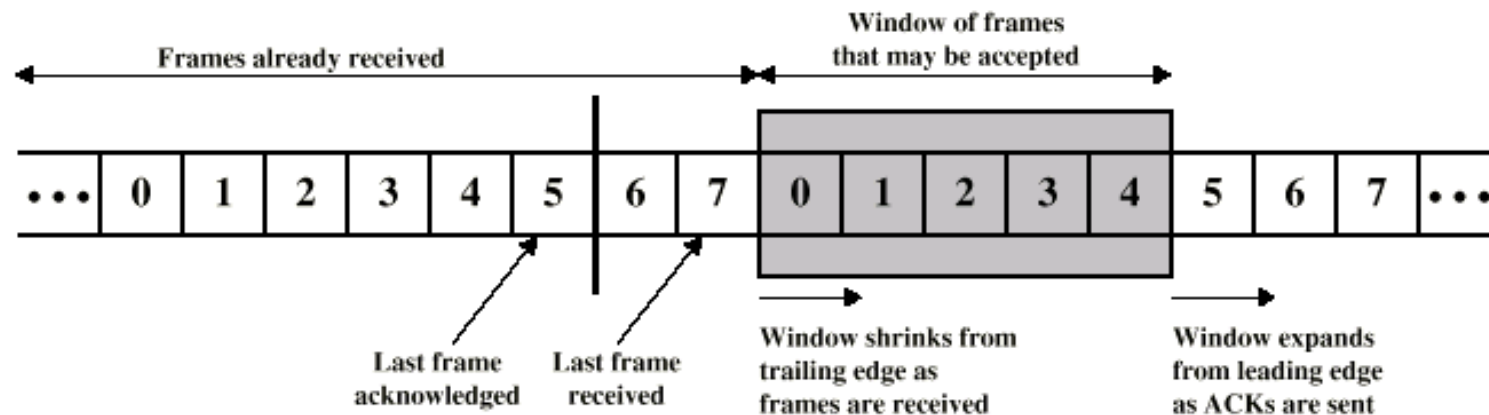
$$520/20 = 26 \text{ frames}$$

- By the time it has finished sending 26 frames, at $t = 520$ ms, the ack. for frame 0 will have just arrived. Thereafter ack will arrive every 20 ms, so the sender always gets permission to continue just when it needs it.
- Hence, we can say the sender window size is 26.

Sliding Window Diagram

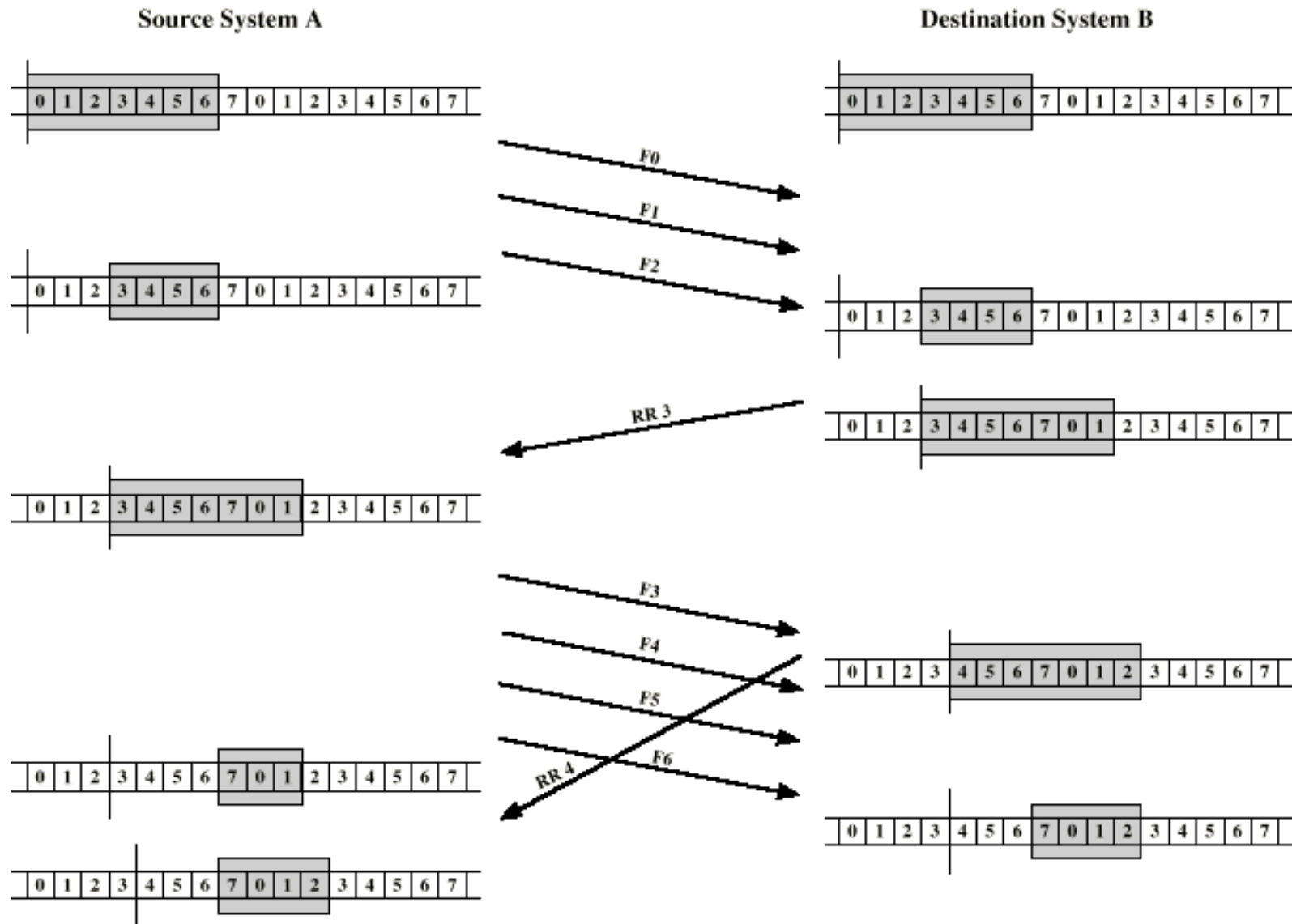


(a) Sender's perspective



(b) Receiver's perspective

Example Sliding Window



Example :

Let the frame size is 2000 bits. The channel capacity is 100kbps. Find out the window size for sliding window protocol if the propagation delay is 100msecs.

Frame size = 2000 bits

channel capacity = 100 Kbps

time taken for send a frame = $\left[\frac{2000}{100K} \right] = 20 \text{ m secs}$

Example :

A channel has a bit rate of 4 kbps and a propagation delay of 20msec. For what range of frame sizes does stop and wait give an efficiency of **at least 50 %** ?

bit rate = 4 Kbps

Propagation delay = 20 m secs.

If $I = bR$ the efficiency will be 50%.

so $I = (4 \times 10^3) * (20 \times 10^{-3}) = 80$ bits

Frame size = 80 bits.

Problem :

Consider an error free 64 kbps channel with a propagation delay of 540 m secs used to send 512 – byte data frames in one direction, with very short acknowledgements coming back the other way. What is the maximum throughput for window sizes of 1, 7, 15, and 127?

- Channel capacity = 64 k bps
propagation delay = 540 m secs
frame size = 512 bytes = $512 \times 8 = 4096$ bits

For one frame time taken = $\frac{4096}{64 \times 10^3} = 64$ m secs.

with the propagation delay of 540 m secs time to sent one frame will be
= $540 + 64 = 604$ m secs

Number of frames can be sent = $604/64 = 9$ frames.

Through put = number of bits sent per sec.

For a window size of 1, we can send 4096 bits per 604 m secs.

Through put = $4096/604 \text{ m s} = 6781 \text{ bps}$

For a window size of 7, $7 \times 6781 = 47,467 \text{ bps}$

- Pipelining frames over an unreliable channel raises some serious issues.
- First, what happens if a frame in the middle of a long stream is damaged or lost?
- When a damaged frame arrives at the receiver, it obviously should be discarded, but what should the receiver do with all the correct frames following it?
- There are two basic approaches to dealing with errors.
 1. Go Back 'n'(GBN)
 2. Selective repeat or Selective Reject

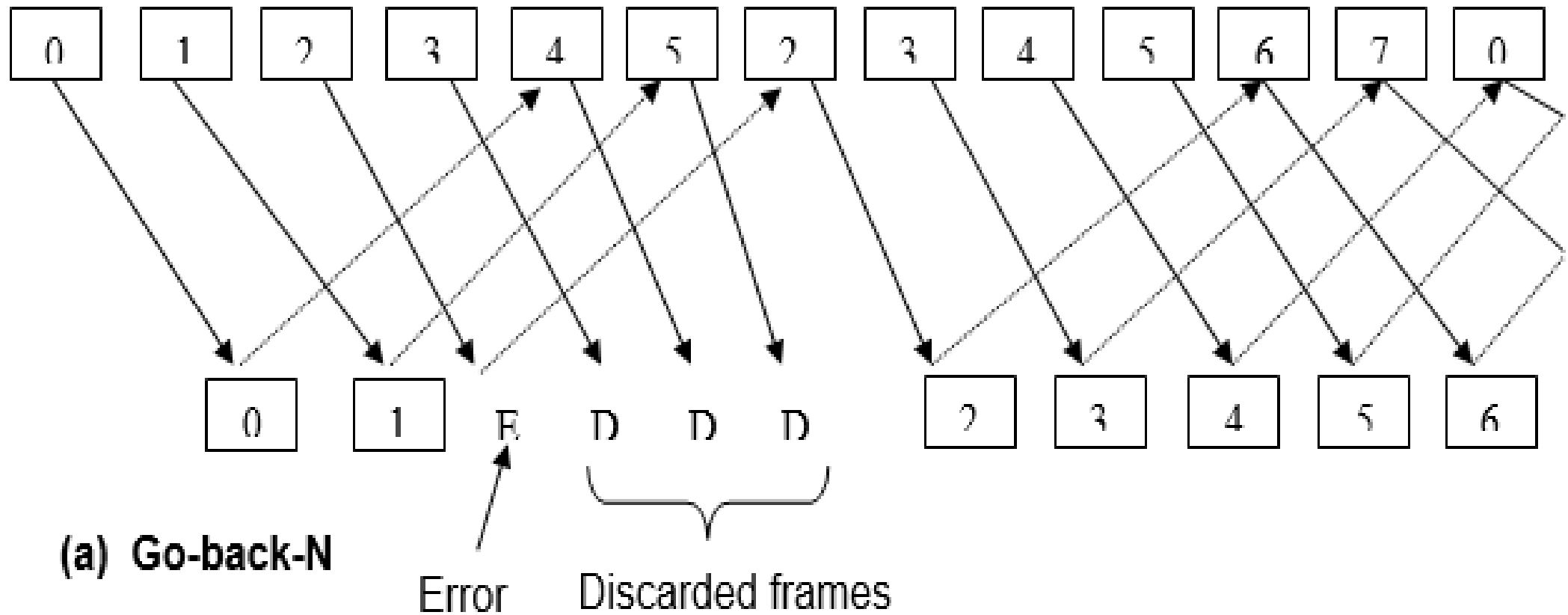
GO – BACK 'N'(GBN)

- ❑ what exactly happens in GBN?
- ❑ Consider the Example given below.
- ❑ We have sender window size of 4.
- ❑ Now the sender has sent the packets 0, 1, 2 and 3.
- ❑ After acknowledging the packets 0 and 1, receiver is now expecting packet 2 and sender window has also slides to further transmit the packets 4 and 5

- . Now suppose the packet 2 is lost in the network, Receiver will discard all the packets which sender has transmitted after packet 2 as it is expecting sequence number of 2.
- On the sender side for every packet send there is a time out timer which will expire for packet number 2.
- Now from the last transmitted packet 5 sender will go back to the packet number 2 in the current window and transmit all the packets till packet number 5.
- That's why it is called Go Back N.

- Go back means sender has to go back N places from the last transmitted packet in the unacknowledged window and not from the point where the packet is lost.

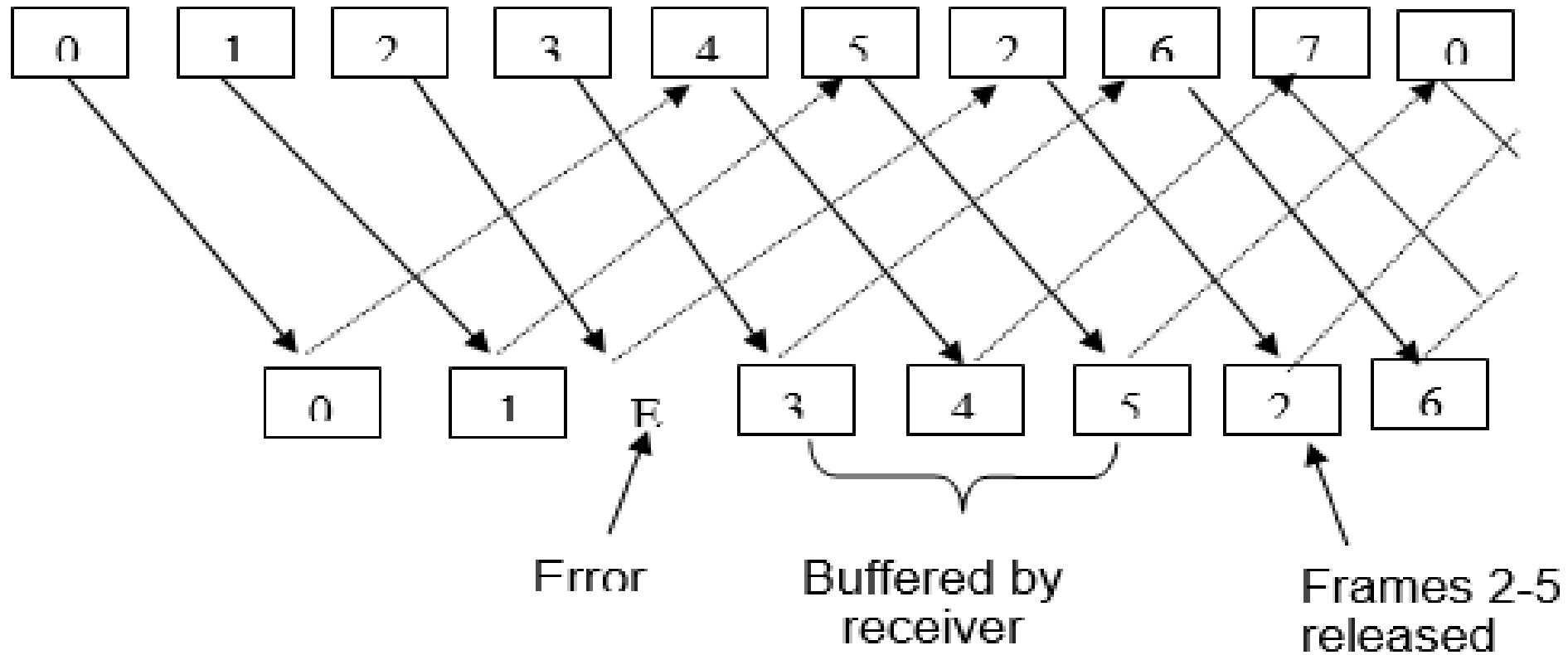
GO – BACK ‘N’



- The go-back-n protocol works well if errors are less, but if the line is poor it wastes a lot of bandwidth on retransmitted frames. An alternative strategy, the selective repeat protocol, is to allow the receiver to accept and buffer the frames following a damaged or lost one.
- Selective Repeat attempts to retransmit only those packets that are actually lost (due to errors) :

- Sender's Windows (W_s) = Receiver's Windows (W_r).
- Sender can transmit new packets as long as their number is within W of all un ACKed packets.
- Sender retransmit un-ACKed packets after a timeout – Or upon a NAK if NAK is employed.
- Receiver ACKs all correct packets.
- Receiver stores correct packets until they can be delivered in order to the higher layer.

SELECTIVE REJECT



b) Selective reject

Acknowledgements

There are 2 kinds of acknowledgements namely:

Cumulative **Ack**: One acknowledgement is used for many packets.

The main advantage is traffic is less.

A disadvantage is less reliability as if one ack is the loss that would mean that all the packets sent are lost.

- **Independent Ack**: If every packet is going to get acknowledgement independently. Reliability is high here but a disadvantage is that traffic is also high since for every packet we are receiving independent ack.

- **Independent Ack:**

If every packet is going to get acknowledgement independently.

- Reliability is high here but a disadvantage is that traffic is also high

- since for every packet we are receiving independent ack.

Difference between Stop and Wait protocol and Sliding Window protocol:

Stop-and-Wait Protocol

1. sender sends one frame and wait for acknowledgment from receiver side.
2. Efficiency of Stop-and-Wait Protocol is worse.
3. Sender window size of Stop-and-Wait Protocol is 1.

Sliding Window Protocol

1. sender sends more than one frame to the receiver side and re-transmits the frame(s) which is/are damaged or suspected.
2. Efficiency of sliding window protocol is better.
3. Sender window size of sliding window protocol is N.

Stop-and-Wait Protocol

- 4. Receiver window size of Stop-and-Wait Protocol is 1.
- 5. In Stop-and-Wait Protocol, sorting is not necessary.
- 6. Stop-and-Wait Protocol is half duplex.

Sliding Window Protocol

- 4. Receiver window size of sliding window protocol may be 1 or N.
- 5. In sliding window protocol, sorting may be or may not be necessary.
- 6. Sliding window protocol is full duplex.

Medium Access sublayer

ALOHA, CSMA

MEDIUM ACCESS CONTROL SUBLAYER (MAC)

- Networks can be categories in to two ways

- a) Point to point b) Broad cast channel

- In broadcast network, the key issue is how to share the channel among several users.

- Ex *a conference call with five people*

- Broadcast channels are also called as multi-access channels or random access channels.

- Multi-access channel belong to a sublayer at the DL layer called the MAC sublayer.

The Channel Allocation problem:

a) **Static channel allocation** in LANs & MANs

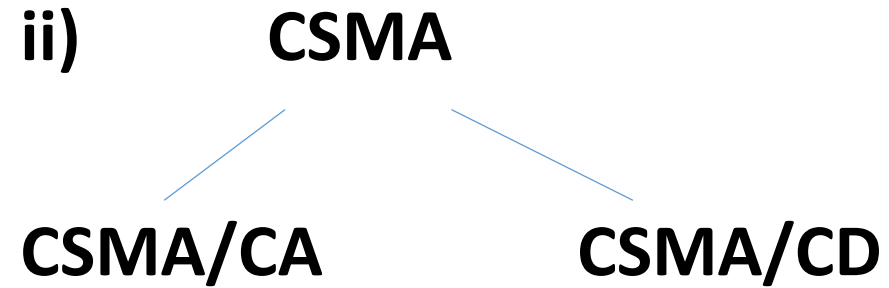
i) **FDM** ii) **TDM**

Drawbacks: -

- 1) Channel is wasted if one or more stations do not send data
- 2) If users increases this will not support.

b)Dynamic channel allocation

i) Pure **ALOHA** & Slotted **ALOHA**



Pure ALOHA

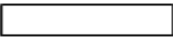
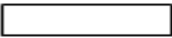
- 1970's Norman Abramson and his colleagues devised this method, used ground –based radio broad costing.
- This is called the **ALOHA** system.
- The basic idea, many users are competing for the use of a single shared channel.
- There are two versions of ALOHA
 1.Pure and 2. Slotted.

- Pure ALOHA does not require global time synchronization, where as in slotted ALOHA the time is divided into discrete slots into which all frames must fit.

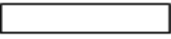
- - Let users transmit whenever they have data to send.
- There will be collisions and all collided frames will be damaged.
- Senders will be knowing through feedback property whether the frame is destroyed or by listening channel.
 - [With a LAN it is immediate, with a satellite, it will take 270m sec.]
- If the frame was destroyed, the sender waits random amount of time and again sends the frame.
- The waiting time must be random otherwise the same frame will collide over and over.

USER

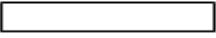
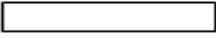
A



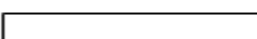
B



C



D



TIME

- Frames are transmitted at completely arbitrary times
- Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be destroyed.

We have to find out what is the efficiency of an ALOHA channel?

- Let us consider an infinite collection of interactive users sitting at their systems (stations).
- A user will always in two states **typing or waiting**.
- Let the 'Frame time' denotes the time required to transmit one fixed length frame.

- Assume that infinite population of users are generating new frames according to poisson distribution with mean N frames per frame time.
- If $N > 1$ users are generating frames at a higher rate than the channel can handle.
- For reasonable throughput $0 < N < 1$.
- In addition to new frames, the station also generates retransmission of frames.
- Old and new frames are G per frame time.
- $G \geq N$

- At low load there will be few collisions, so $G \sim N$
- Under all loads, the throughput $S = GP_o$, where P_o is the probability that a frame does not suffer a collision.
- A frame will not suffer a collision if no other frames are sent within one frame time of its start.
- Let 't' be the time required to send a frame.
- If any other user has generated a frame between time t_o and t_o+t , the end of that frame will collide with the beginning of the shaded frame.
- Similarly, any other frame started b/w t_o+t and t_o+2t will bump into the end of the shaded frame.

The probability that 'k' frames are generated during a given frame time is given by the poisson distribution:

$$P_r[k] = \frac{G^k e^{-G}}{k!}$$

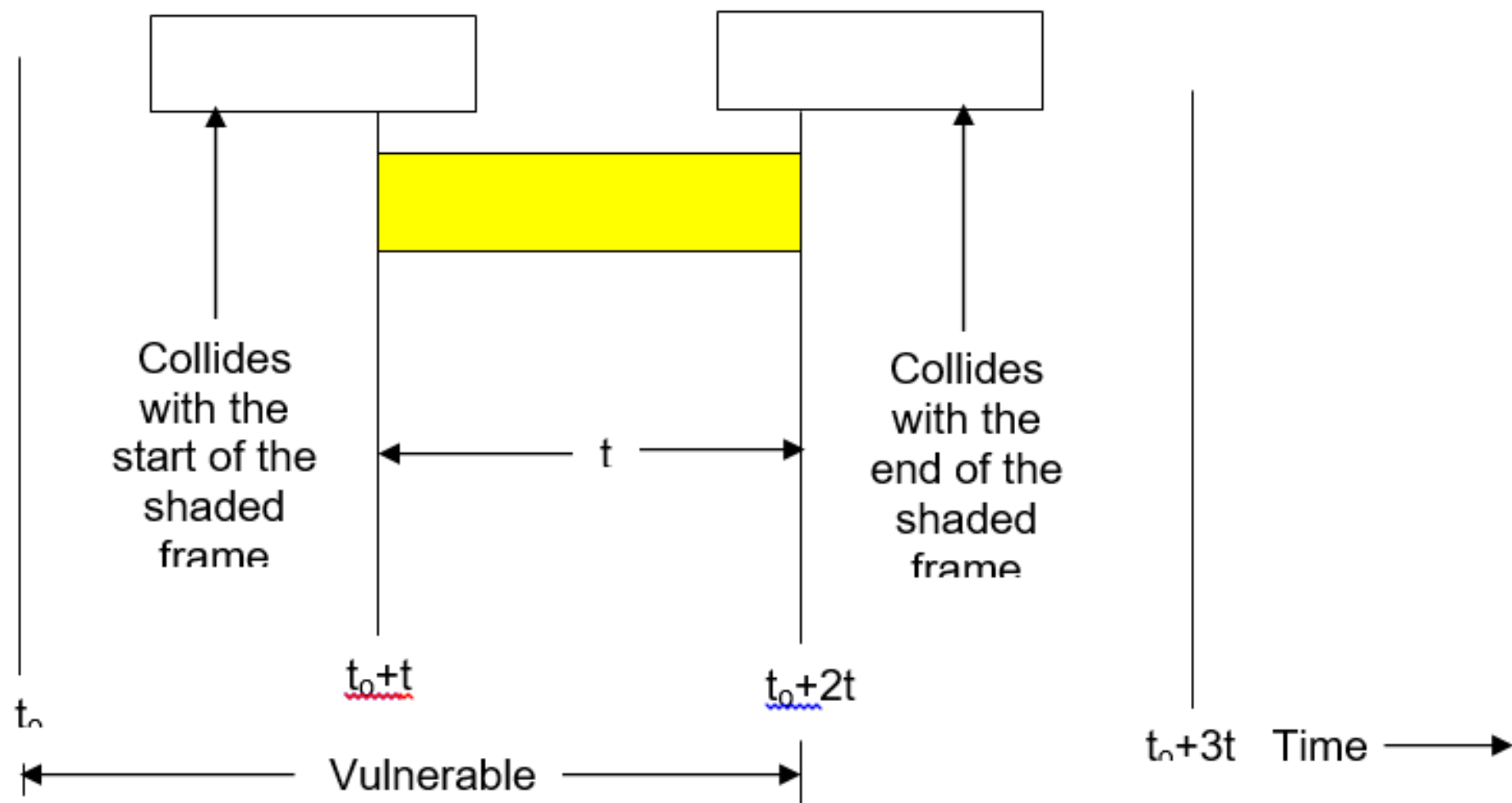
The probability of zero frames is just e^{-G}

-In an interval two frame times long, the mean number of frames generated is $2G$.

-The probability of no other traffic being initiated during the entire vulnerable period is given by $P_o = e^{-2G}$ $S = G e^{-2G}$ $[S = G P_o]$

The Maximum throughput occurs at $G=0.5$ with $S=1/2e = 0.184$

Hence the channel utilization at pure ALOHA = **18%**.

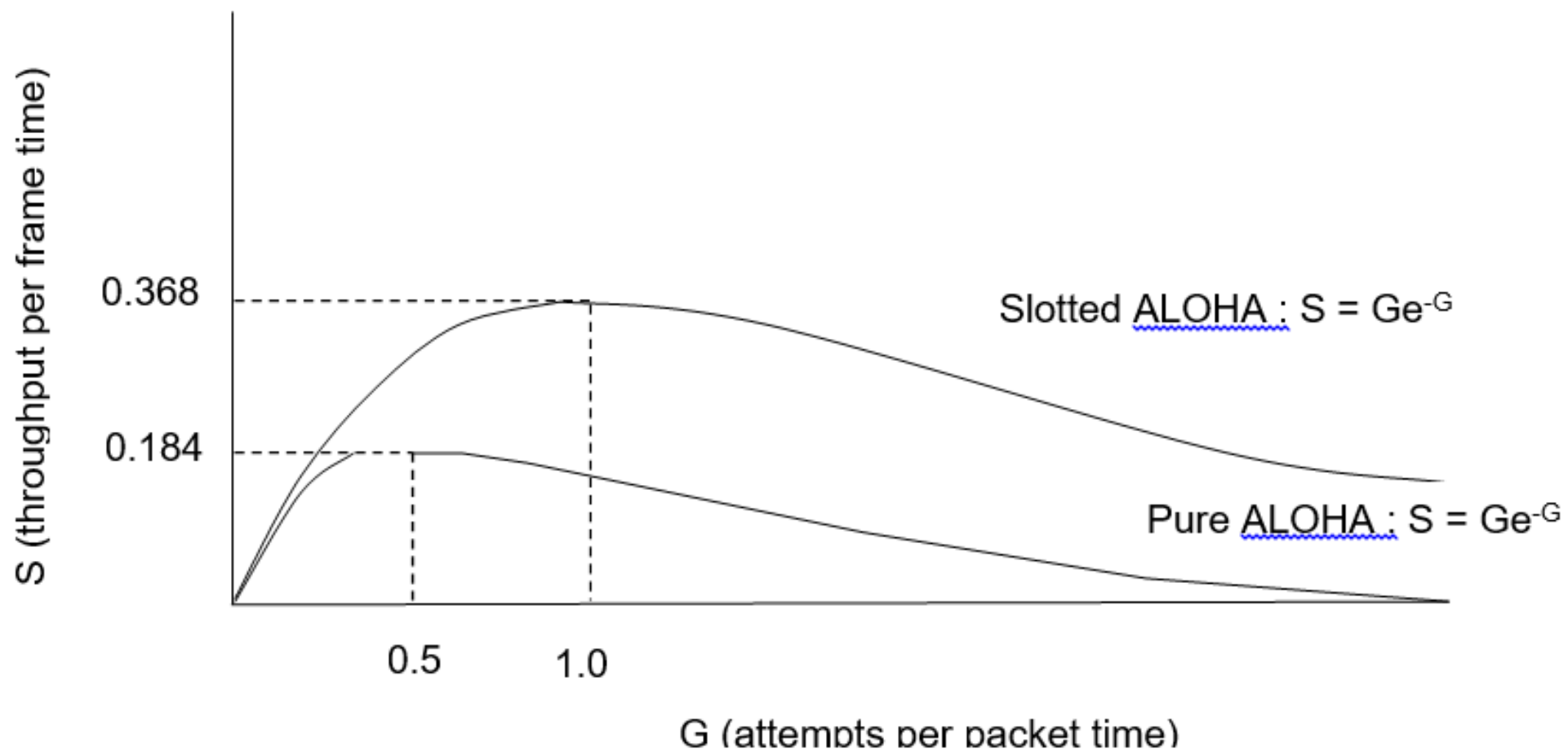


Vulnerable period for the shaded frame

- In 1972, Roberts' devised a method for doubling the capacity of ALOHA system.
- In this system the time is divided into discrete intervals, each interval corresponding to one frame.
- One way to achieve synchronization would be to have one special station emit a pip at the start of each interval, like a clock.
- In Roberts' method, which has come to be known as slotted ALOHA, in contrast to Abramson's pure ALOHA, a computer is not permitted to send whenever a carriage return is typed.
- Instead, it is required to wait for the beginning of the next slot.
- Thus the continuous pure ALOHA is turned into a discrete one.
- Since the vulnerable period is now halved, the of no other traffic during the same slot as our test frame is e^{-G} which leads to

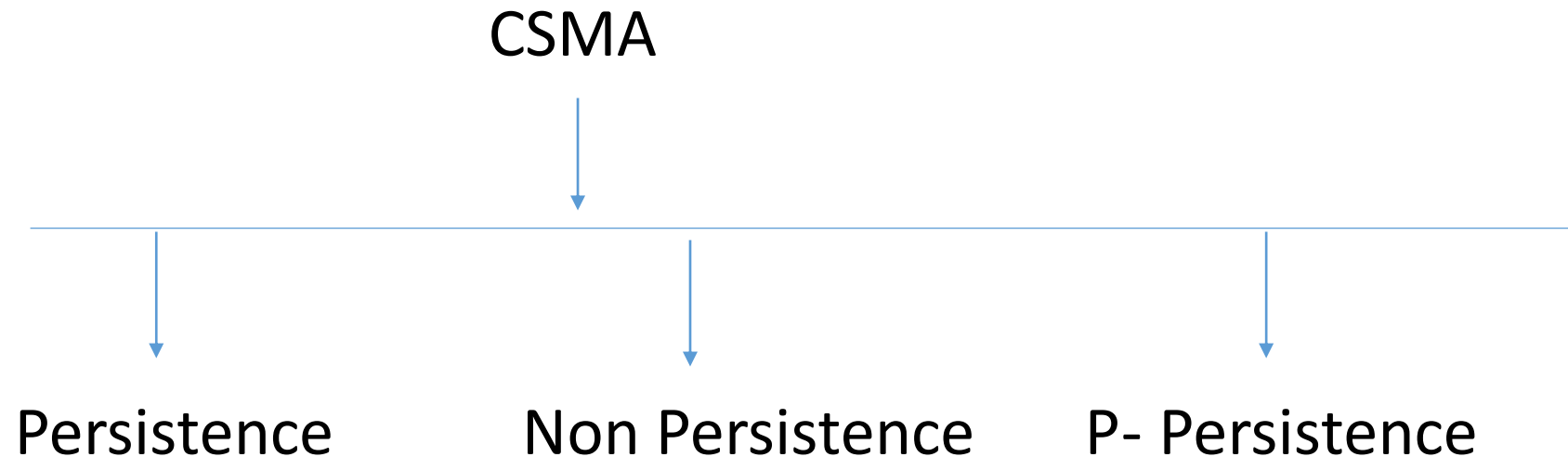
$$S = Ge^{-G}$$

- At $G=1$, slotted ALOHA will have maximum throughput.
- So $S=1/e$ or about 0.368, twice that of pure ALOHA.
- The channel utilization is 37% in slotted ALOHA.



Carrier Sense Multiple Access Protocols (CSMA)

- Protocols in which stations listen for a carrier (CHANNEL) and act accordingly are called carrier sense protocols.



Persistent CSMA

1. When a station has data to send, it first listens to the channel to see if any one else is transmitting at that moment.
2. If the channel is busy, the station waits until it become idle.
3. When the station detects an idle channel, it transmits a frame.
4. If a collision occurs, the station waits a random amount of time and starts all over again.
5. The protocol is called 1-persistent also because the station transmits with a probability of 1 when it finds the channel idle.
6. The propagation delay has an important effect on the performance of the protocol.
7. The longer the propagation delay the worse the performance of the protocol.
8. Even if the propagation delay is zero, there will be collisions.
9. If two stations listen the channel, that is idle at the same, both will send frame and there will be collision.

Non persistent CSMA

- In this, before sending, a station sense the channel.
- If no one else is sending, the station begins doing so it self.
- However, if the channel is busy,
-- the station does not continually sense it but it waits a random amount of time and repeats the process.
- This algorithms leads to better channel utilization but longer delays then 1-persistent CSMA.
- With persistent CSMA, what happens if two stations become active when a third station is busy?
- Both wait for the active station to finish, then simultaneously launch a packet, resulting a collision.
- There are two ways to handle this problem.
 1. P-persistent CSMA
 2. Exponential back off.

P-Persistent CSMA

The first technique is for a waiting station not to launch a packet immediately

- when the channel becomes idle, but first toss a coin, and send a packet only if the coin comes up heads.
- If the coin comes up tails, the station waits for some time (one slot for slotted CSMA), then repeats the process.
- The idea is that if two stations are both waiting for the medium, this reduces the chance of a collision from 100% to 25%.
- A simple generalization of the scheme is to use a biased coin, so that the probability of sending a packet when the medium becomes idle is not 0.5, but p , where $0 < p < 1$.
- We call such a scheme **P-persistent CSMA**.
- The original scheme, where $p=1$, is thus called 1-persistent CSMA.

Exponential backoff

- The key idea is that each station, after transmitting a packet, checks whether the packet transmission was successful.
- Successful transmission is indicated either by an explicit acknowledgement from the receiver or the absence of a signal from a collision detection circuit.
- If the transmission is successful, the station is done.
- Otherwise, the station retransmits the packet, simultaneously realizing that at least one other station is also contending for the medium.
- To prevent its retransmission from colliding with the other station's retransmission, each station backs off (that is, idles) for a random time chosen from the interval $[0, 2 * \text{max_propagation_delay}]$ before retransmitting its packet.
- If the retransmission also fails, then the station backs off for a random time in the interval $[0, 4 * \text{max_propagation_delay}]$, and tries again.
- Each subsequent collision doubles the back-off interval length, until the retransmission final succeeds.
- On a successful transmission, the back-off interval is reset to the initial value.
- We call this type of back-off exponential back-off

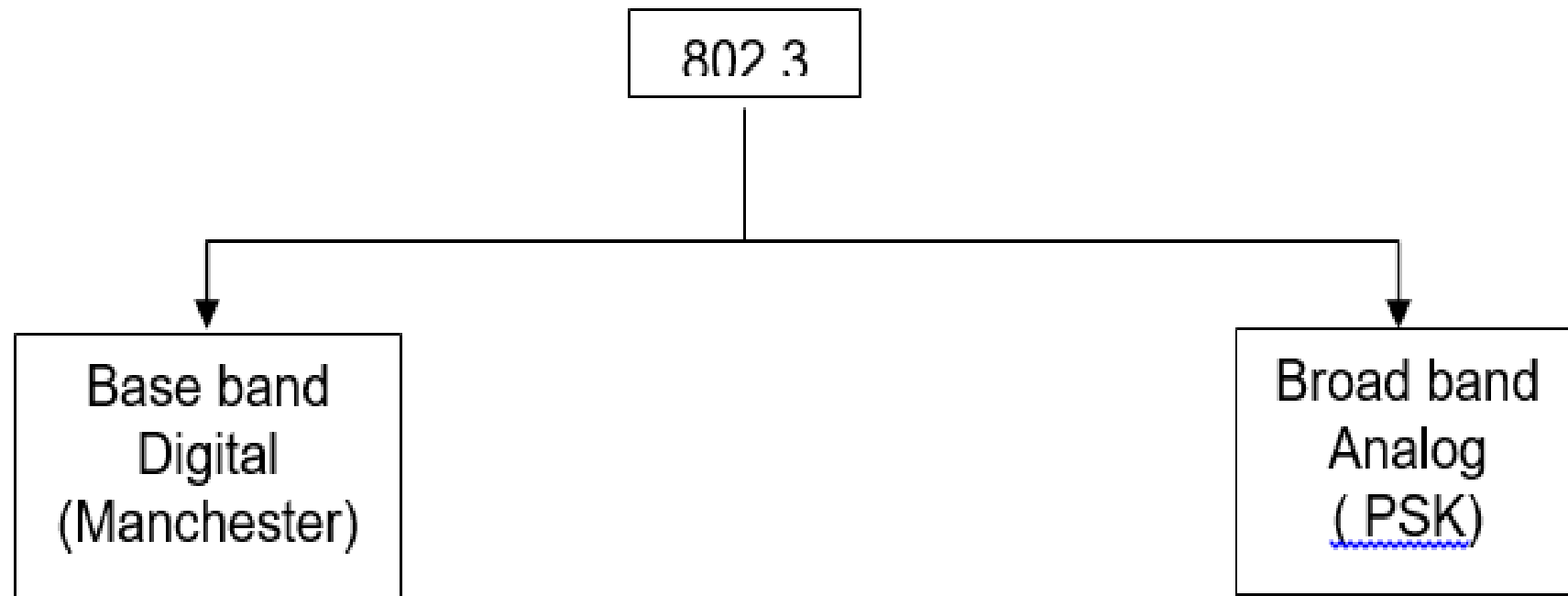
LAN Standards

IEEE 802.3

IEEE 802.11

IEEE 802.3

- The IEEE 802.3 is a 1-persistent CSMA/CD LAN.
- Xerox built a 2.94 Mbps CSMA/CD system to connect over 100 personal workstations on 1-Km cable.
- This system was called Ethernet through which electromagnetic radiation was once thought to propagate.
- Xerox DEC and Intel came with another standard for 100 Mbps Ethernet.
- This differs from old one that it runs at speeds from 1 to 10 Mbps on various media.
- The second difference between these two is in one header (802.3 length field is used for packet type in Ethernet).



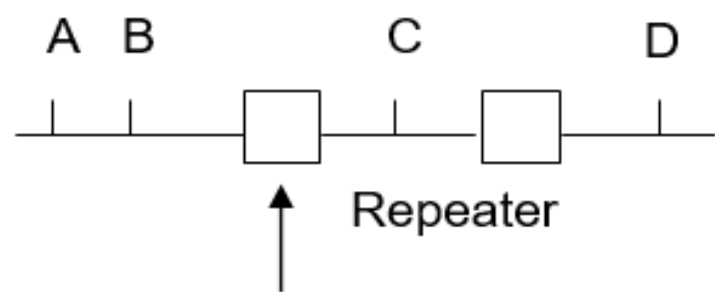
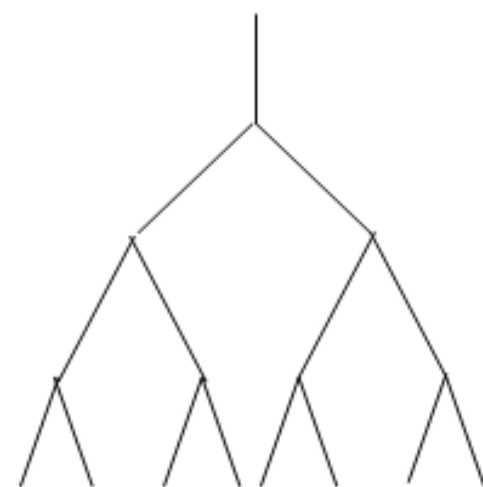
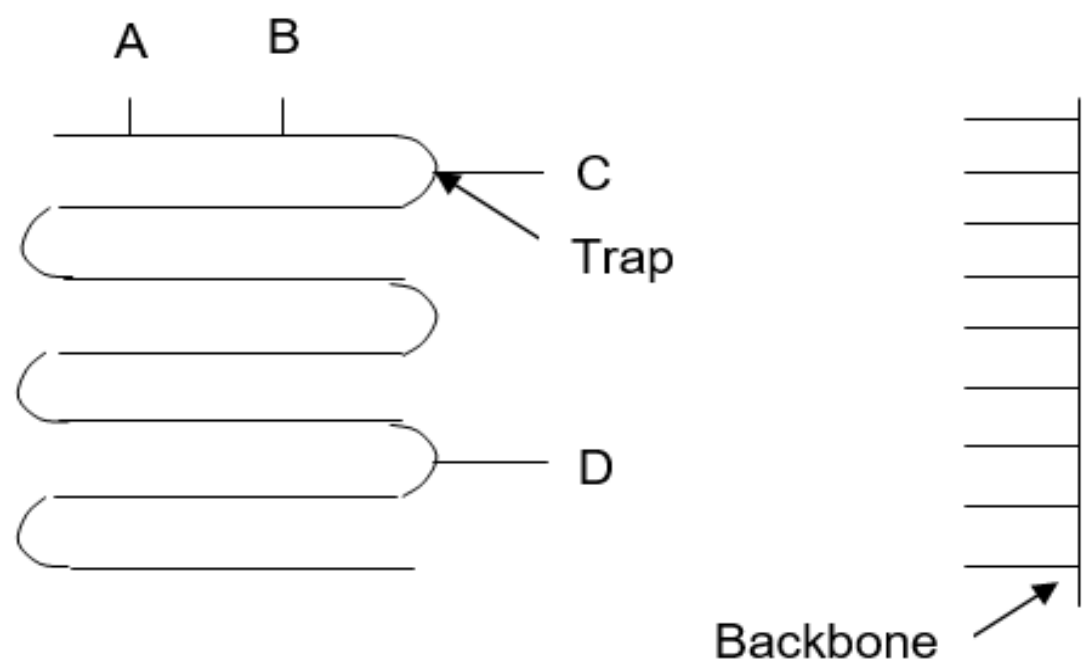
10Base5, 10Base2

10Base-T, 1Base5

100 Base-T

10 Broad 36

Name	Cable	Max. <u>segment</u>	Nodes/ <u>seg.</u>	Advantages
10Base5	Thick coax	500 m	100	Good for backbones
10Base2	Thin coax	200 m	30	Cheapest system
10Base-T	Twisted pair	100 m	1024	Easy maintenance
10Base-F	Fiber optics	2000 m	1024	Best between buildings



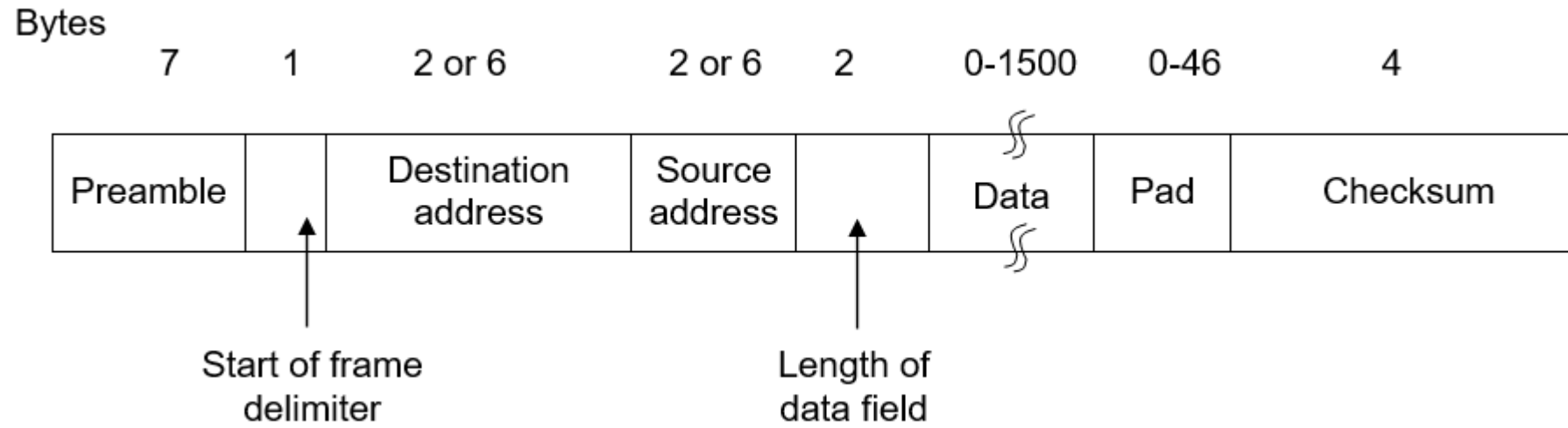
10 BASE 5: THICK ETHERNET

10 BASE 2: THIN ETHERNET (Thin net, cheap net)

10 BASE T: Twisted pair Ethernet

	10 BASE –5 (ETHERNET)	10 BASE – 2 (Cheap net)	10 BASE -Net (UTP)
Access control	CSMA/CD	CSMA/CD	CSMA/CD
Topology	Bus	Bus	Star
Message Protocol	Variable packet	Variable packet	Variable packet
Signaling rate	10 Mbps	10 Mbps	10 Mbps
Signaling type	IEEE 802.3 thick double Shielded coax	Base band RG -58 coax	UTP telephone cable
Cable impedance	50 ohms	50 ohms	NA
Minimum node separation	2.5 m	0.5 m	NA
Maximum segment length	500 m	185 m	100 m (pc to hub)
Maximum number of segments	5	2	NA
Maximum station separation	2500 m	925 m	NA

The 802.3 MAC sub layer protocol:



I) Preamble:

Each frame start with a preamble of 7 bytes each containing a bit pattern 10101010.

II) Start of frame byte:

It denotes the start of the frame itself. It contains 10101011.

III) Destination address:

This gives the destination address.

The higher order bit is zero for ordinary address and 1 for group address (Multicasting).

All bits are 1s in the destination field frame will be delivered to all stations (Broadcasting).

The 46th bit (adjacent to the high-order bit) is used to distinguish local from global addresses.

IV) Length field:

This tells how many bytes are present in the data field from 0 to 1500.

V) Data field:

This contains the actual data that the frame contains.

VI) Pad:

Valid frame must have 64 bytes long from destination to checksum. If the frame size less than 64 bytes pad field is used to fill out the frame to the minimum size.

VII) Checksum:

It is used to find out the receiver frame is correct or not. CRC will be used here.