UNIT –2 ERROR DETECTION,CORRECTION AND DATA LINK CONTROL

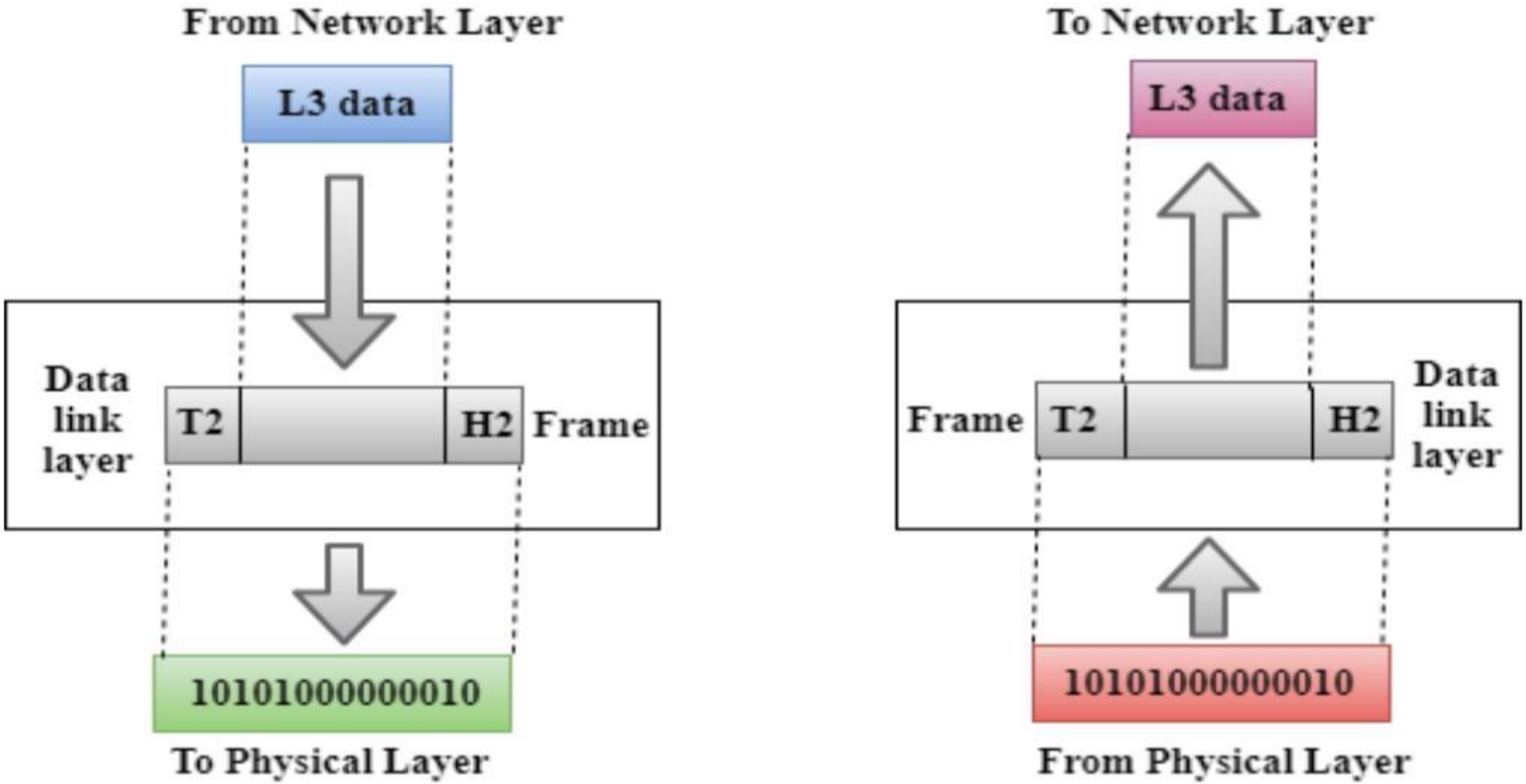
- Error Detection and Correction : Introduction, Error Detection, Error Correction
- Linear Block Codes: hamming code, Hamming Distance, parity check code
- Cyclic Codes: CRC (Polynomials), Advantages Of Cyclic Codes, Other Cyclic Codes As Examples: CHECKSUM: One's Complement, Internet Checksum
- Framing: fixed-size framing, variable size framing.
- Flow control: flow control protocols.
- Noiseless channels: simplest protocol, stop-and-wait protocol.
- Noisy channels: stop-and-wait automatic repeat request, go-back-n automatic repeat request, Selective repeat automatic repeat request, piggybacking

- **Data Link Layer**

- The Data-link layer is the second layer from the bottom in the OSI (Open System Interconnection) network architecture model.

- It is responsible for the node-to-node delivery of data.

- Its major role is to ensure error-free transmission of information.


From Network Layer
L3 data
Data link layer
T2
H2
Frame
10101000000010
To Physical Layer
To Network Layer
L3 data
Frame
T2
H2
Data link layer
10101000000010
From Physical Layer

- Two types of functions of the data link layer –

- Data link control – framing, flow & error control

- Media access control –Mac is responsible for moving data packets to & from one network interface card to another across shared channel

- Each computer has its own MAC address

- Functions of the Data-link layer

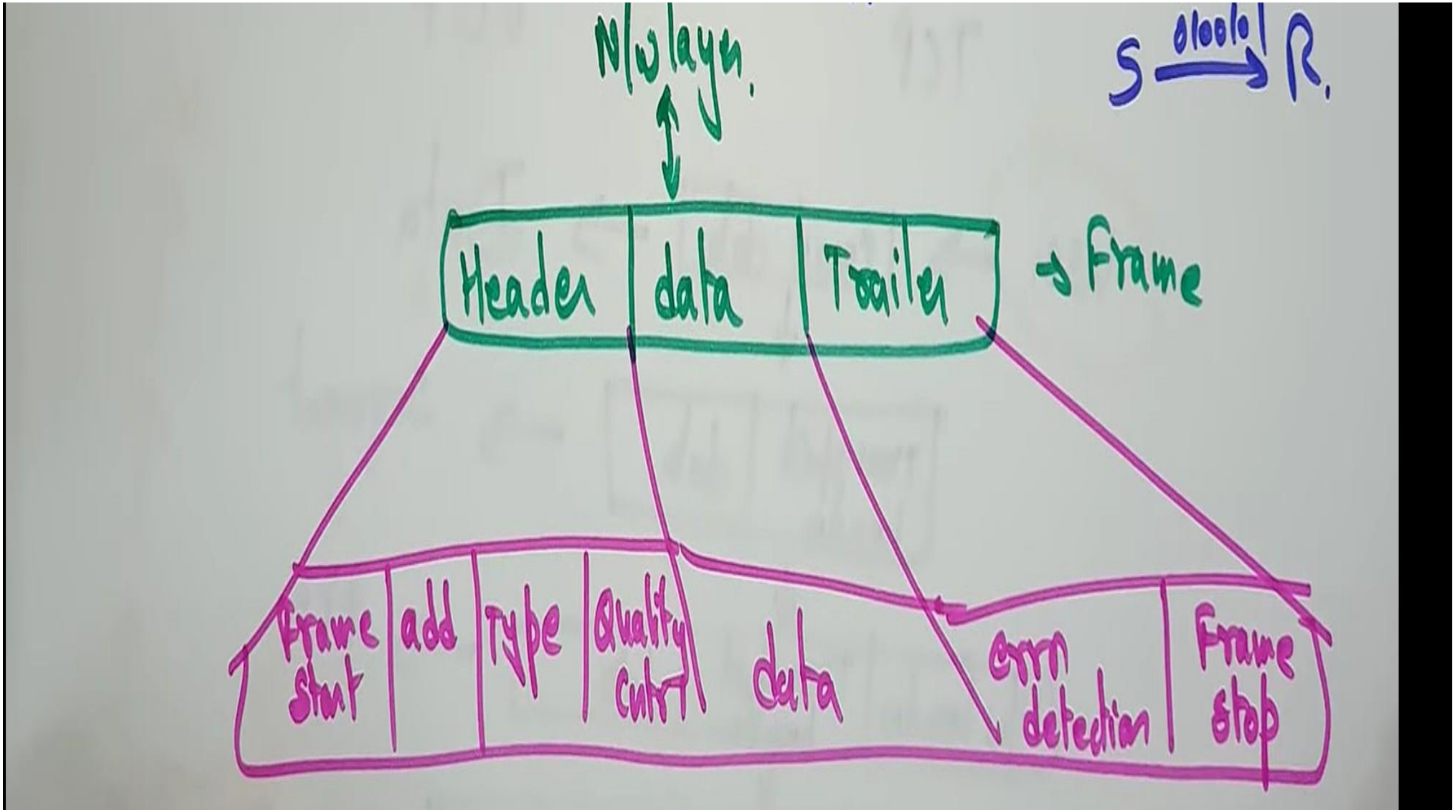
Framing

Flow Control:

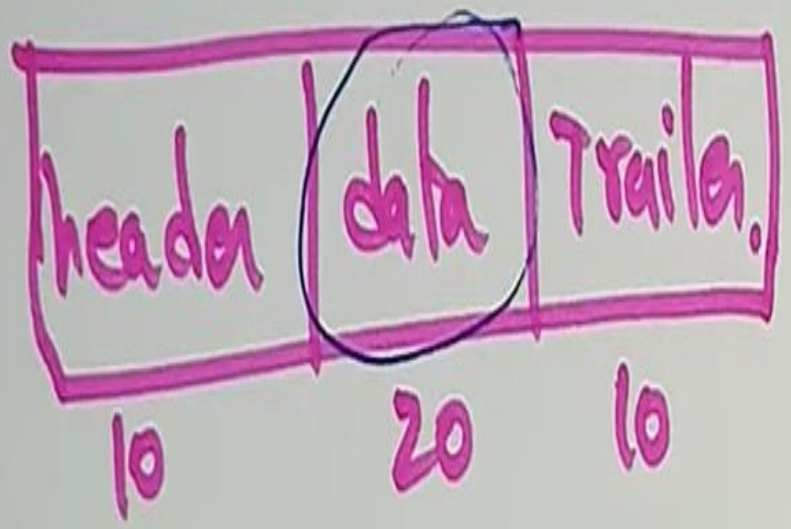
Error Control:

- **Framing:** The data link layer translates the physical's raw bit stream into packets known as Frames.

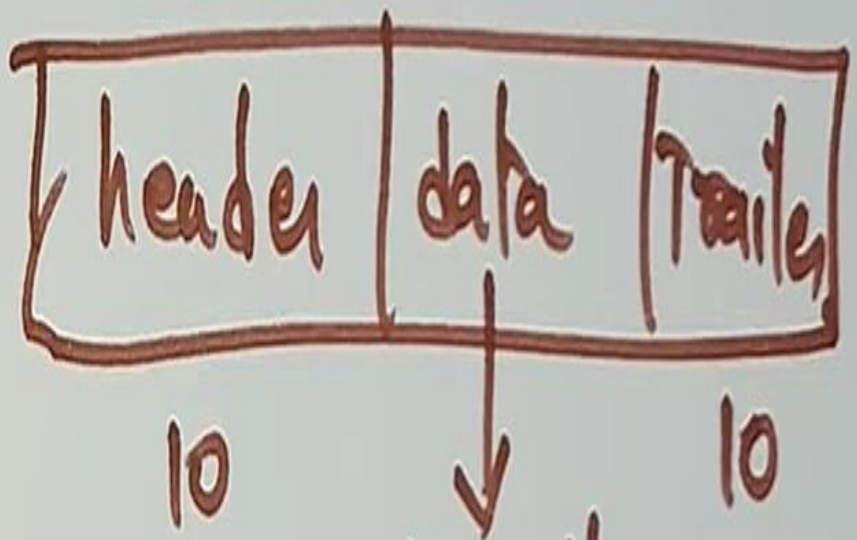
- The Data link layer adds the header and trailer to the frame.


N/w layer.
S
01001
R.
Header
data
Trailer
→ Frame
Frame start
add
type
Quality cntrl
data
error detection
Frame stop

① fixed length



② Variable length

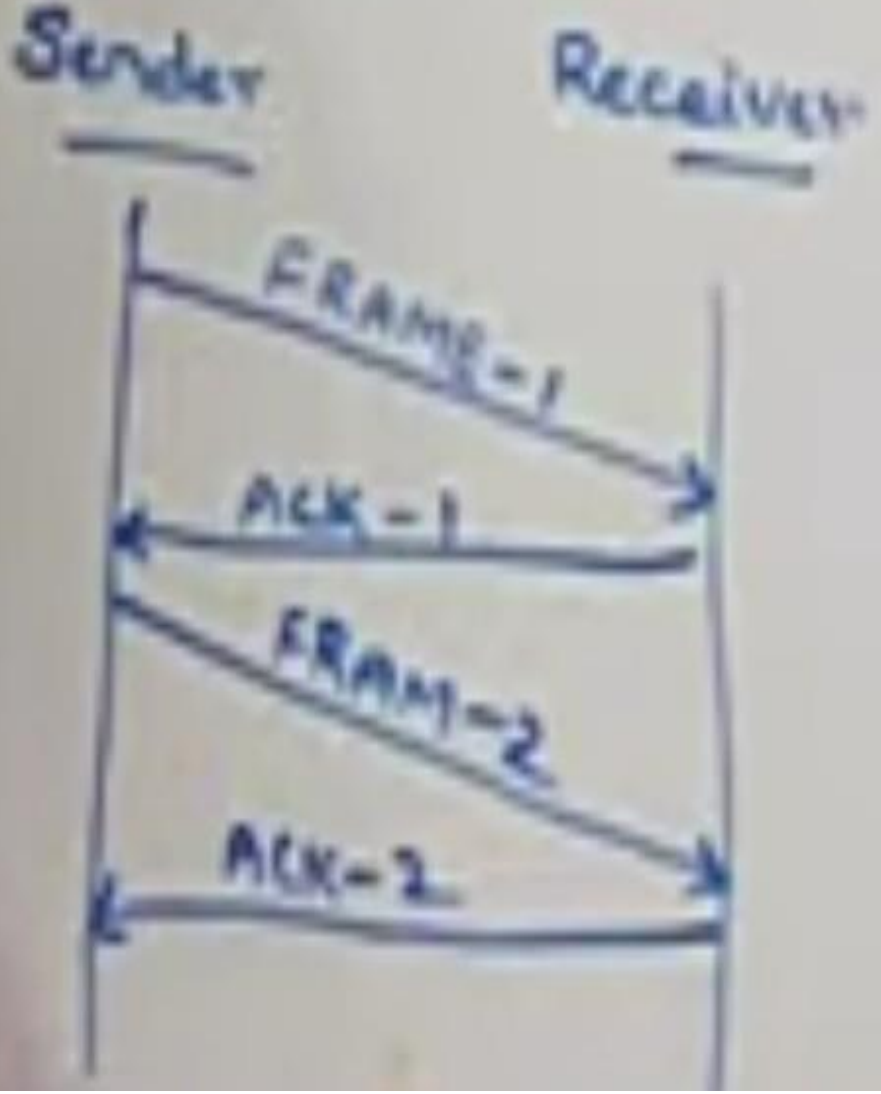


- **Flow Control:** Flow control is the main functionality of the Data-link layer.

- It is the technique through which the constant data rate is maintained on both the sides so that no data get corrupted.

① STOP AND WAIT

② SLIDING WINDOW



- **Error Control:** Error control is achieved by adding a calculated value CRC Redundancy Check)

- that is placed to the Data link layer's trailer

If any error seems to occur, then the receiver sends the acknowledgment for the retransmission of the corrupted frames.

ERROR CONTROL

* Retransmission

Phases of Error Control

① Error Detection

② Acknowledgement (ACK)

   ⓐ positive ACK

   ⓑ Negative ACK

③ Retransmission

Errors

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

- Types of Errors
- •*Single-bit error*
- •*Burst error*

Multiple bit error

- Single bit error : there is only one bit corrupted (Means change from 0 to 1 or 1 to 0)

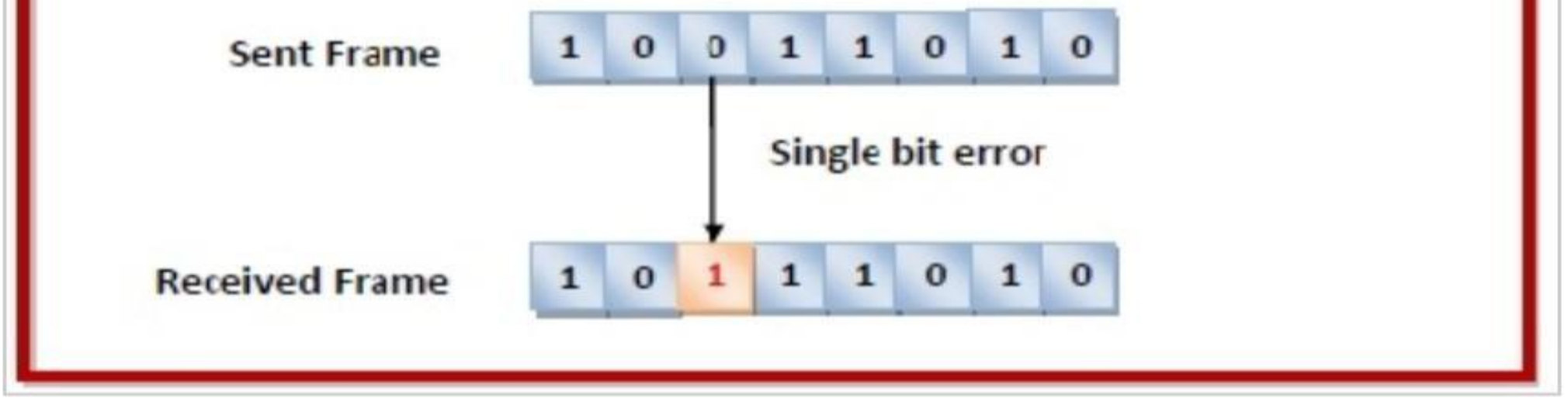
- Sent data : 10110**0**11
- Received data 10110**1**11

- Multiple bit error : frame is received with more than one corrupted bits

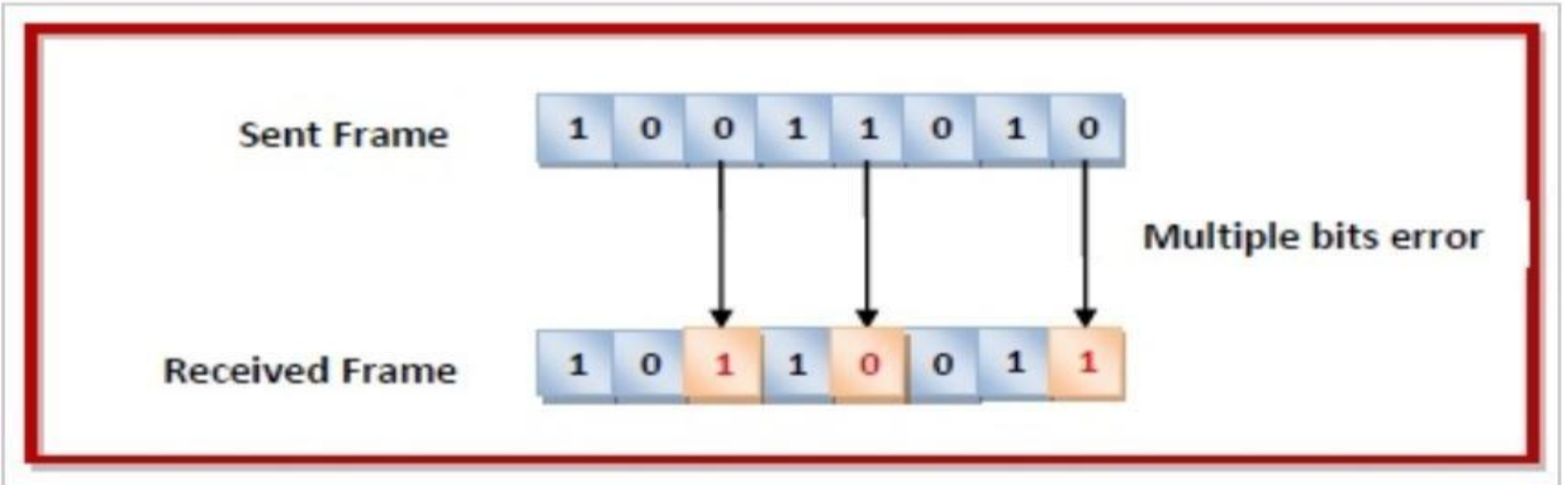
- Example :
- Sent data : 101**1**0**0**11
- Received data : 101**0**0**1**11

- Burst error : two or more consecutive bits are corrupted

- Example :
- Sent data : 1**011**0011
- Received data : 1**100**0111



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



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

What Are Advantages Of FEC ?

(1) It reduces the errors in the transmission system quickly.

(2) No need for re-transmission of data.

(2) It enable for long distance transmission with accuracy.

(4) Low communication power is needed for it.

The disadvantage with FEC is its Latency.

Error Detection Techniques:

Some popular techniques for error detection are:

1. **Simple Parity check**
2. **Two-dimensional Parity check**
3. **Cyclic redundancy check**
4. **Checksum**

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

Parity Check

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

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

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

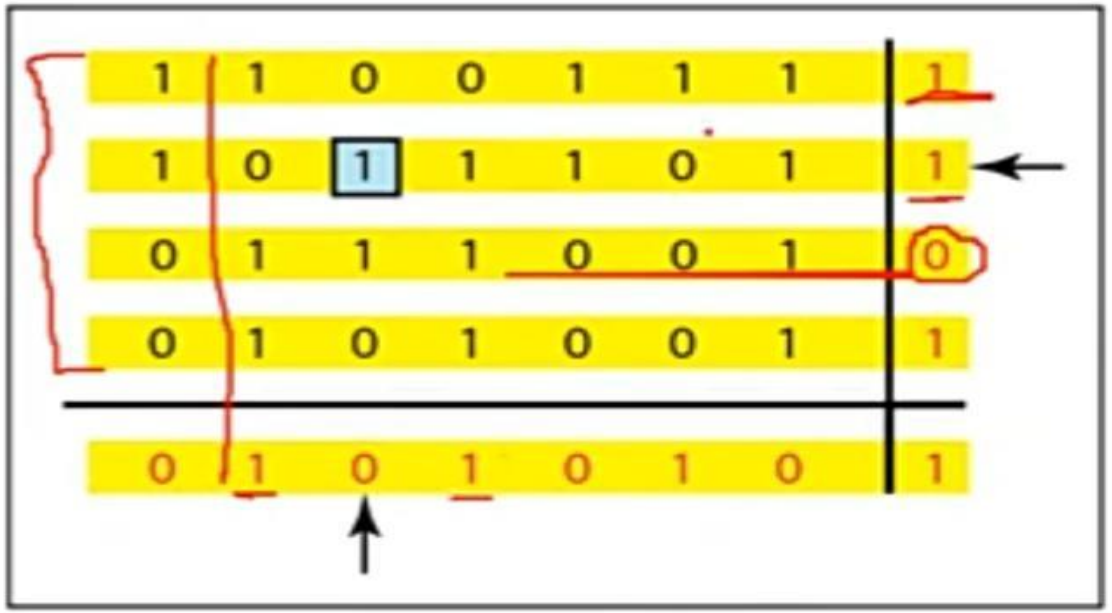
Parity Bit

- A single bit is appended to each data chunk
  - makes the number of 1 bits even/odd
- Example: even parity
  - 1000000(1)
  - 1111101(0)
  - 1001001(1)
- Example: odd parity
  - 1000000(0)
  - 1111101(1)
  - 1001001(0)

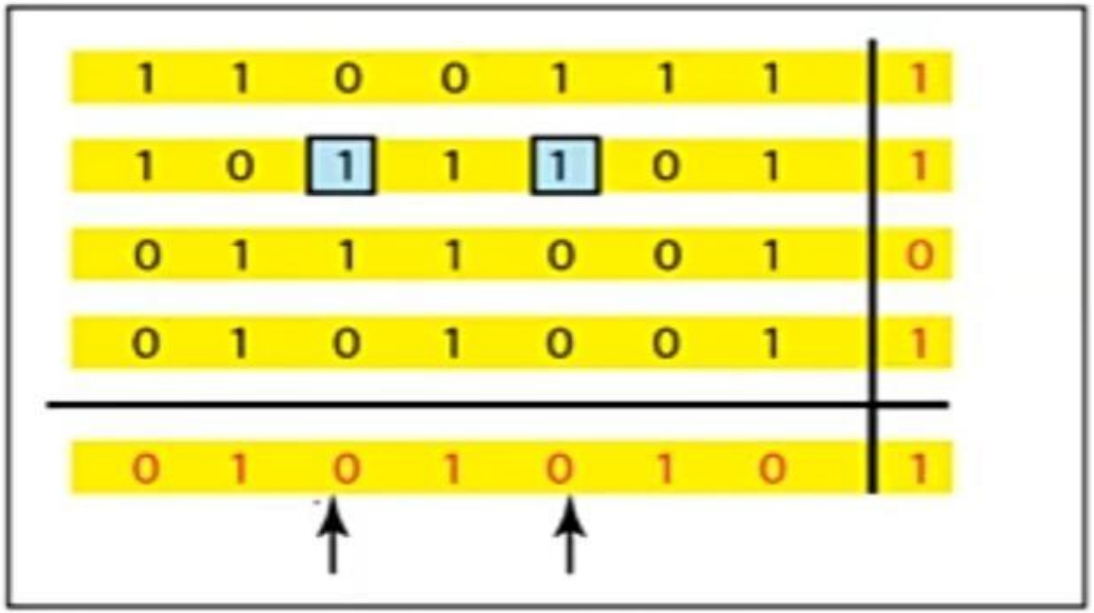
Two-dimensional Parity check

- Parity check bits are also calculated for all columns then both are sent along with the data
- At the receiving end these are compared with the parity bits calculated on the received data

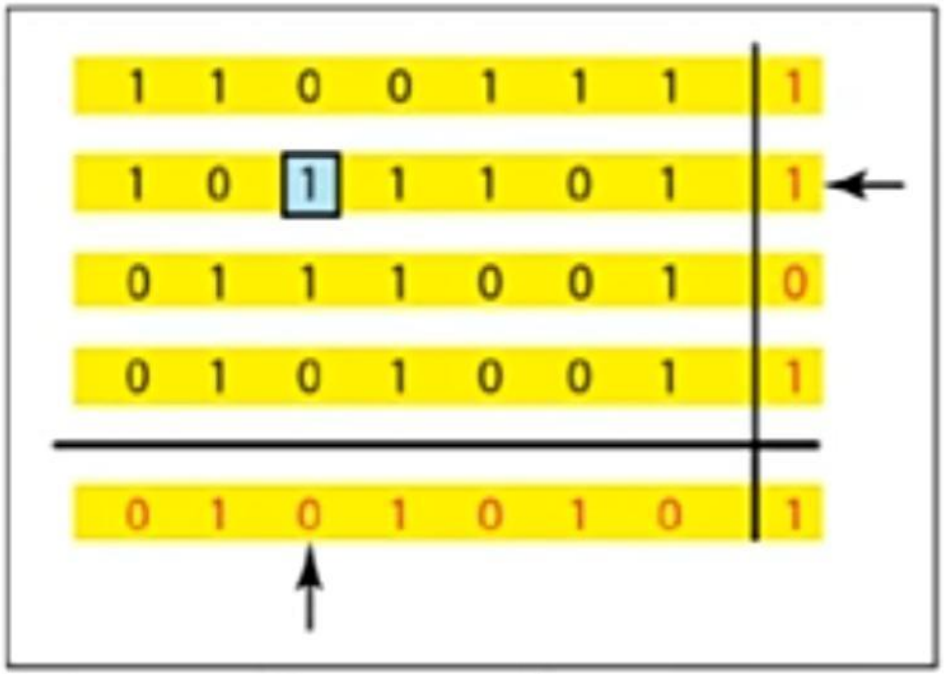
Figure *Two-dimensional parity-check code*



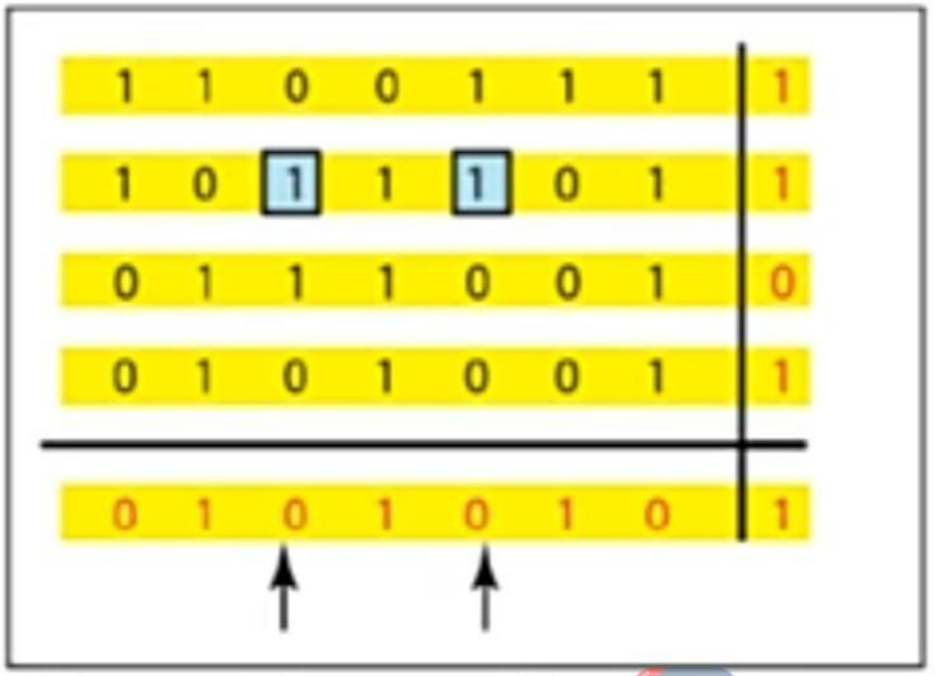
b. One error affects two parities



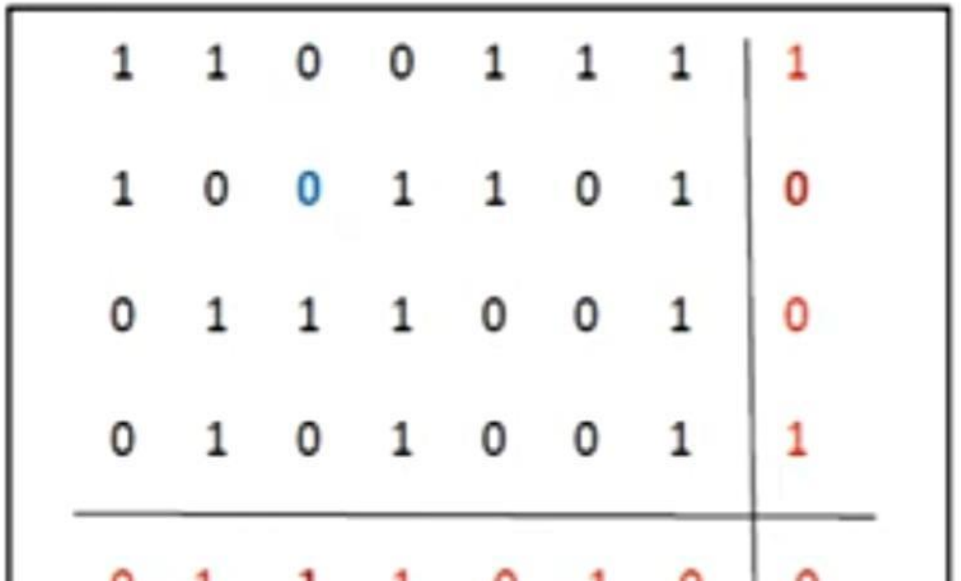
c. Two errors affect two parities

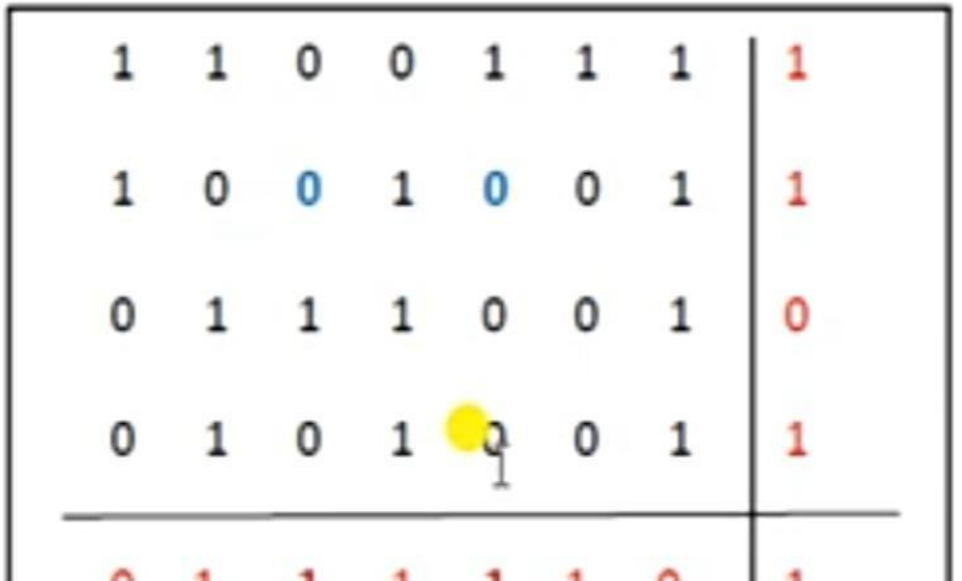


b. One error affects two parities



c. Two errors affect two parities

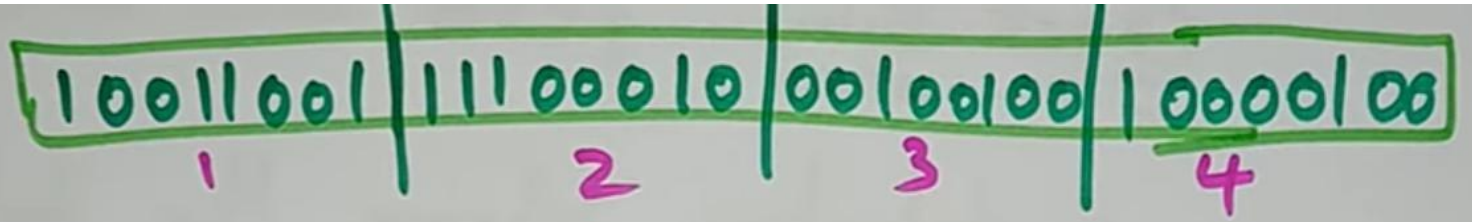




Checksum

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.



k=4
m=8

Sender

1 → 10011001

2 → 11100010

① 01111101 1

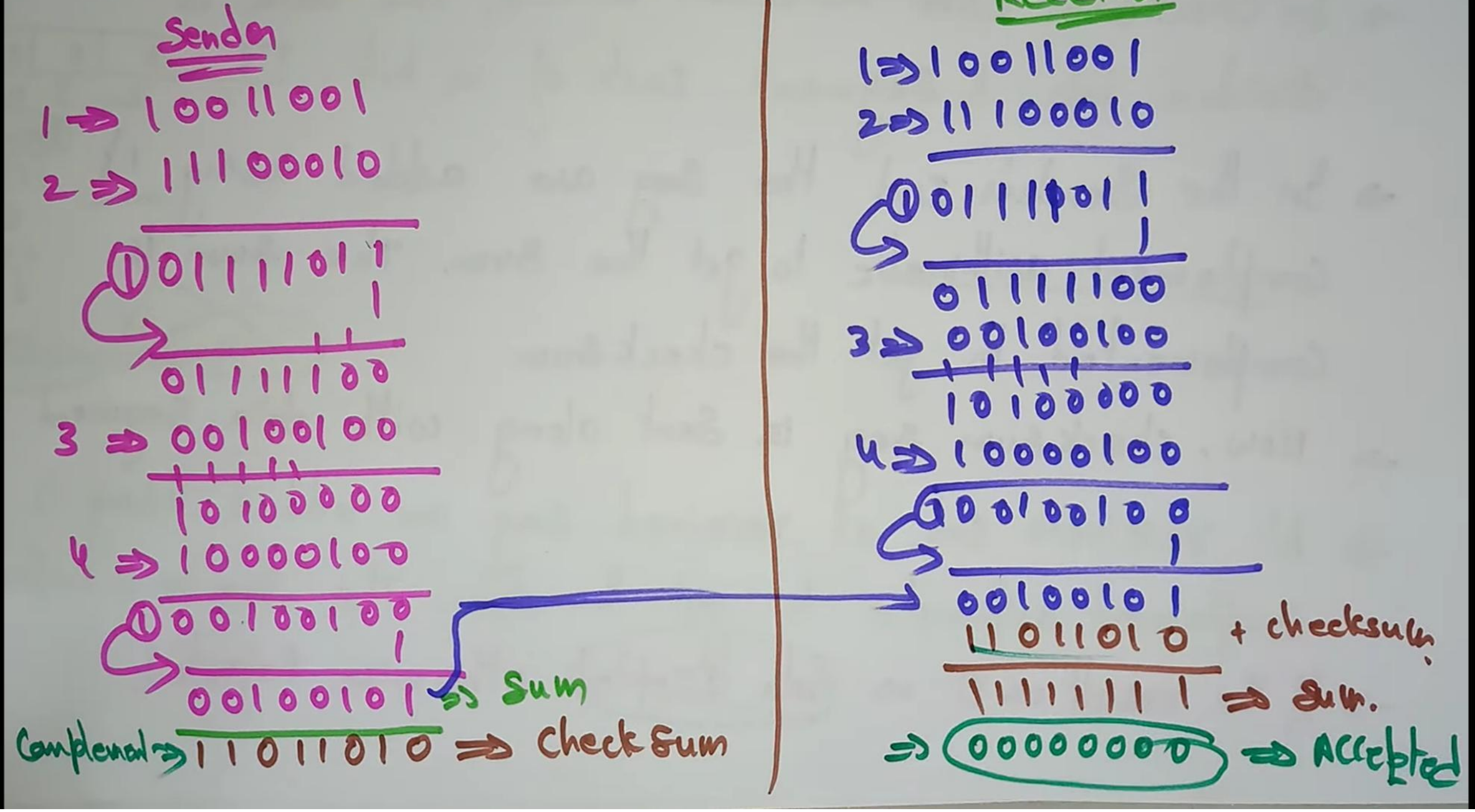
01111100

3 → 00100100

10100000

4 → 10000100

① 00100100


Sender
1 → 10011001
2 → 11100010
01111101
01111100
3 → 00100100
10100000
4 → 10000100
00100100
00100101
Sum
Complement → 11011010 ⇒ Checksum
Receiver
1 ⇒ 10011001
2 ⇒ 11100010
01111101
01111100
3 ⇒ 00100100
10100000
4 ⇒ 10000100
00100100
00100101
11011010 + checksum
11111111 ⇒ sum.
⇒ 00000000 ⇒ Accepted

Concept behind Checksum

- Suppose our data is a list of five 4-bit numbers that we want to send to a destination.
- In addition to sending these numbers, we send the sum of the numbers.
- For **example**, if the set of numbers is (7, 11, 12, 0, 6), we send (7, 11, 12, 0, 6, 36), where 36 is the sum of the original numbers.
- The receiver adds the five numbers and compares the result with the sum.
- If the two are the same, the receiver assumes no error, accepts the five numbers, and discards the sum. Otherwise, there is an error somewhere and the data are not accepted.

Checksum for Error Detection:- In this each word is added to the previous word and total sum [Checksum] is calculated. Then the checksum is transmitted along with the data.

Idea of checksum:- (7, 11, 12, 0, 6) -to send.

W1 + W2 + ... + W5 → Checksum

7 + 11 + 12 + 0 + 6 = 36

(Transmitter)

Data	C.S

(Receiver)

Data	C.S

Calculate Checksum → C.S1

Compare

if C.S = C.S1 → No Error

7, 11, 12, 0, 6	-36

(TR) → (RE)

Calculate checksum (7 + 11 + 12 + 0 + 6) = 36

-36 + 36 = 0 ✓ No = Error.

Checksum using one's Complement:-

Checksum = 36

10 0 1 0 0] 36

10

0 1 1 0 (6)

Inverse → 1 0 0 1 (9)

7, 11, 12, 0, 6	9

7 + 11 + 12 + 0 + 6 + 9 = 45

10 1 1 0 1] 45

10

1 1 1 1

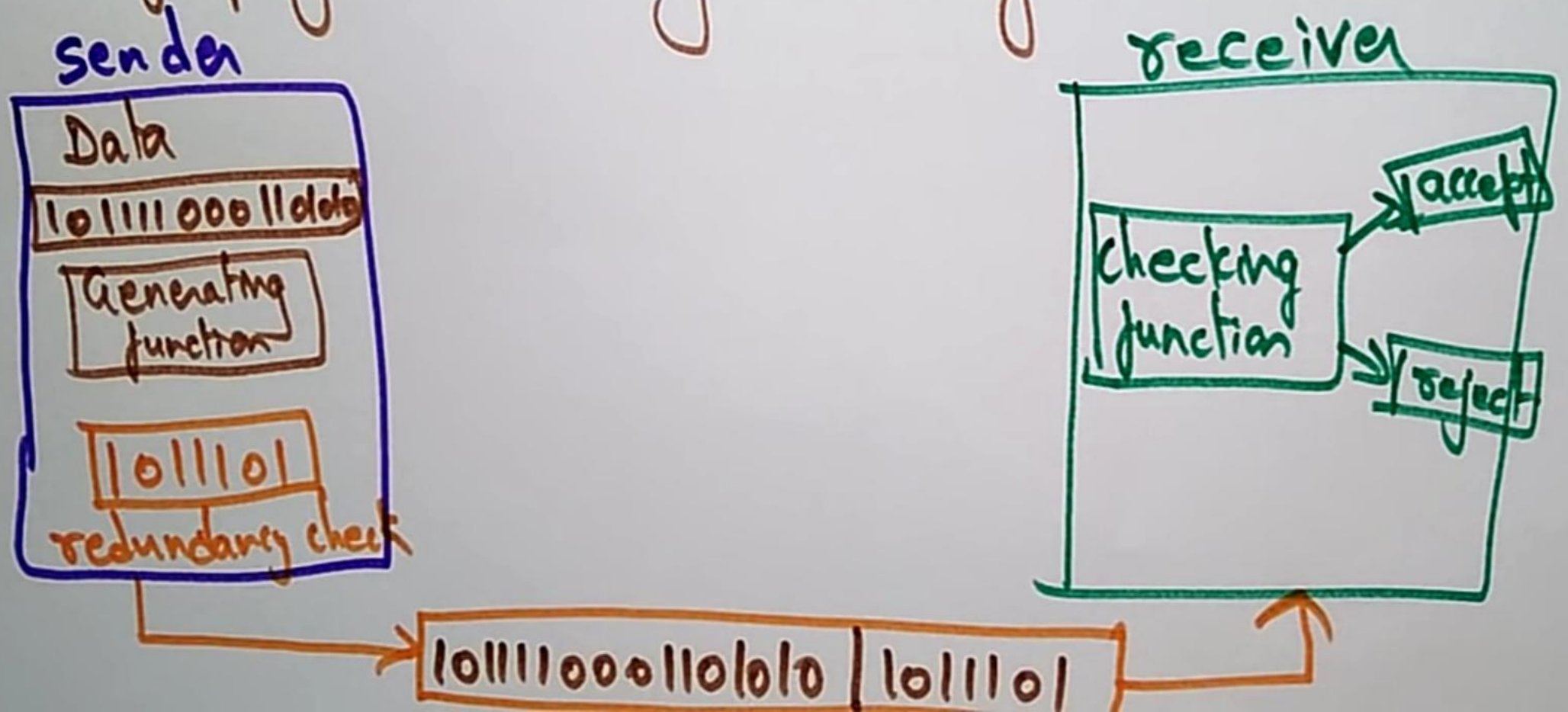
Inverse → 0 0 0 0

No = Error

① **Redundancy** :- It is the main technique to detect errors.

⇓

group of bits adding to end of data unit



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

Cyclic Redundancy Check

① Data

② CRC Generator

③ CRC bits

Sender

Receiver

* CRC Generator ⟶ n bits (4)

* CRC bits ⟶ (n-1) (3)

$$\text{CRC bits} = \frac{\text{Data} + (n-1)\text{ 0's}}{\text{CRC Generator}}$$

(Reminder)

$$\frac{\text{Data} + 000}{\text{CRC Generator}}$$

Data + CRC ⟶

$$\frac{\text{Data+CRC}}{\text{CRC Generator}}$$

⇓

000

↓

remainder

Cyclic Redundancy Check

① Data ⟶ 101101

② CRC Generator ⟶ 1101

③ CRC bits ⟶ 3

④ Division — XOR

DATA + CRC ⟹

CRC bits =

```
1101 | 101101000
       1101
       -----
        0110 0
        1101
        -----
         000 1 0 0
             1 1 0 1
             -------
             0 0 0 1 0
```

CRC = 010

101101010 ⟹

Receiver

DATA = 101101010

DATA | CRC

CRC Generator = 1101

```
1101 | 101101010
       1101
       -----
        0110 1 0
        1101
        -----
         000 1 1 0
             1 1 0 1
             -------
             0 0 0 0 0
```

Cyclic Redundancy Check

① Data ⟶ 101101

② CRC Generator ⟶ 1101

③ CRC bits ⟶ 3

④ Division — XOR

DATA + CRC ⟹

CRC bits =

```
1101 | 101101000
     + 1101
       ------
       01100
        1101
        ----
        00011
         1101
         ----
         00010
```

CRC = 010

101101010 ⟹

Receiver

DATA = 110101010

DATA CRC

CRC Generator = 1101

```
1101 | 110101010
       1101
       ----
       0000
        0101
        1101
        ----
        1000
        1101
        ----
        0101
         1101
         ----
         0111   X
```

☞ Advantages of cyclic codes

The advantage of cyclic codes over most of the other codes are as under :

(i) They are easy to encode.

(ii) They possess a well defined mathematical structure which has led to development of very efficient decoding schemes for them.

(iii) The methods c that are to be used for error detection and correction are simpler and easy to implement.

(iv) These methods do not need look-up table decoding.

(v) It is possible to detect the error bursts using the cyclic codes.

☞ Drawbacks of cyclic codes

Even though the error detection is simpler, the error correction is slightly more complicated. This is due to the complexity of the combination logic circuit used for error correction.

Hamming Code

- It is error detecting & correcting code
- It uses a no. of parity bits located at certain positions.

No. of parity bits

Let m = no. of bits in the message.

P = no. of parity bits is determined as

$$2^P \geq m + P + 1$$

10110

Locations of Parity bits in the code

Let the code is of 7 bits (m) 1101

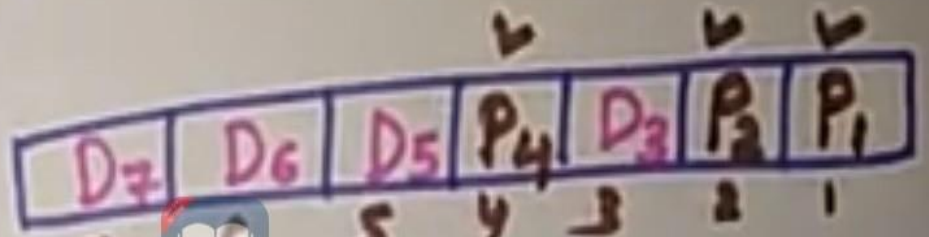
$2^0, 2^1, 2^2, 2^3$

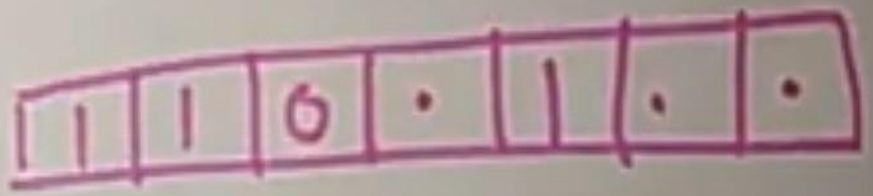
1, 2, 4, 8

m = 4

P = 3

7





Value of Parity bits

$m = 4$
$p = 3$
7

D_7	D_6	D_5	P_4	D_3	P_2	P_1
111	110	101	100	011	010	001

P_1 — D_3 D_5 D_7

P_2 — D_3 D_6 D_7

P_4 — D_5 D_6 D_7

① Encode the binary word 1011 into seven bit even parity Hamming code.

1011

- m = 4 p = 3

P_1 P_2 P_4

1	0	1	0	1	0	1
D_7	D_6	D_5	P_4	D_3	P_2	P_1

P_1	D_3	D_5	D_7
1	1	1	1

P_2	D_3	D_6	D_7
0	1	0	1

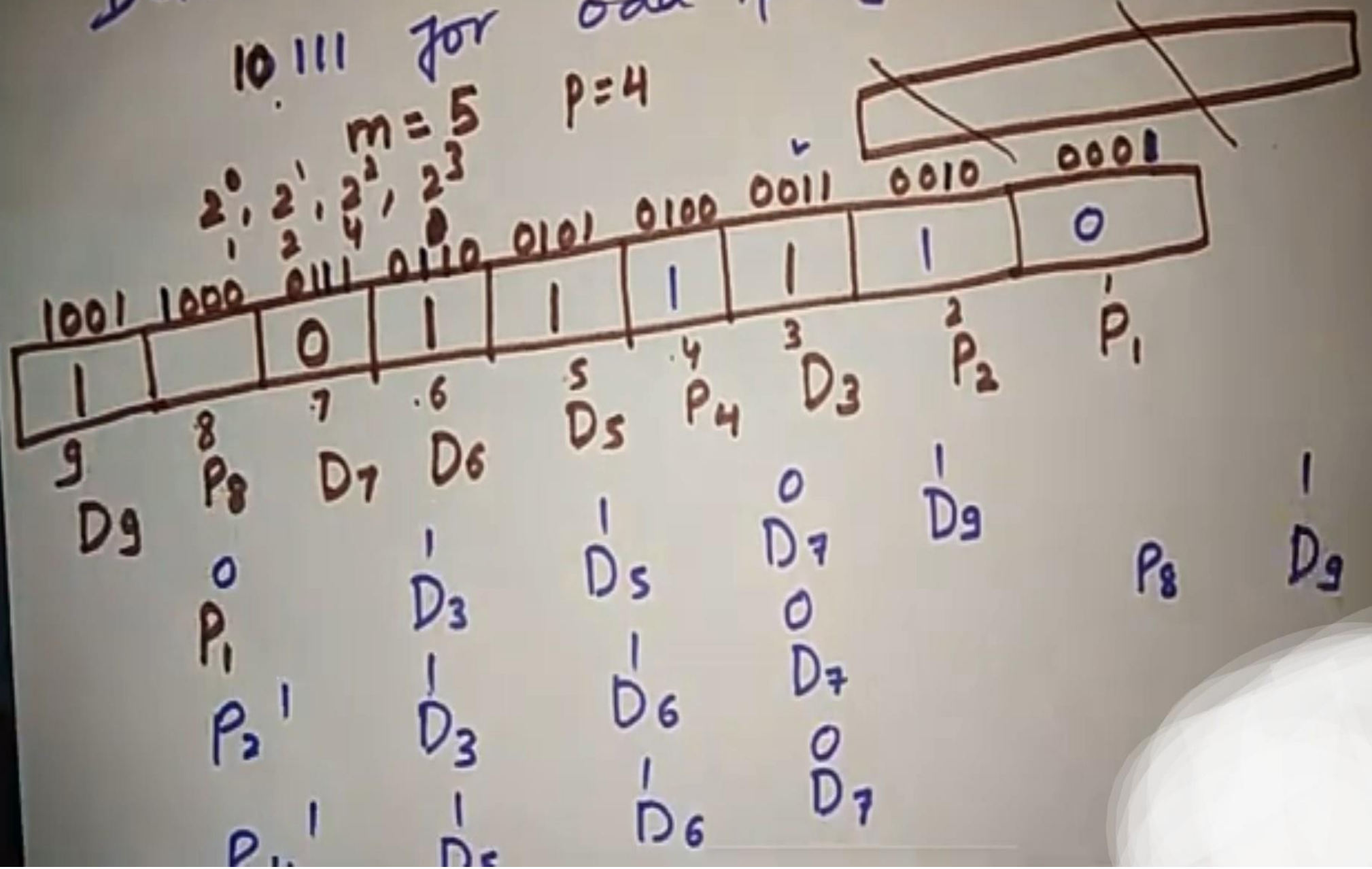
P_4	D_5	D_6	D_7
0	1	0	1

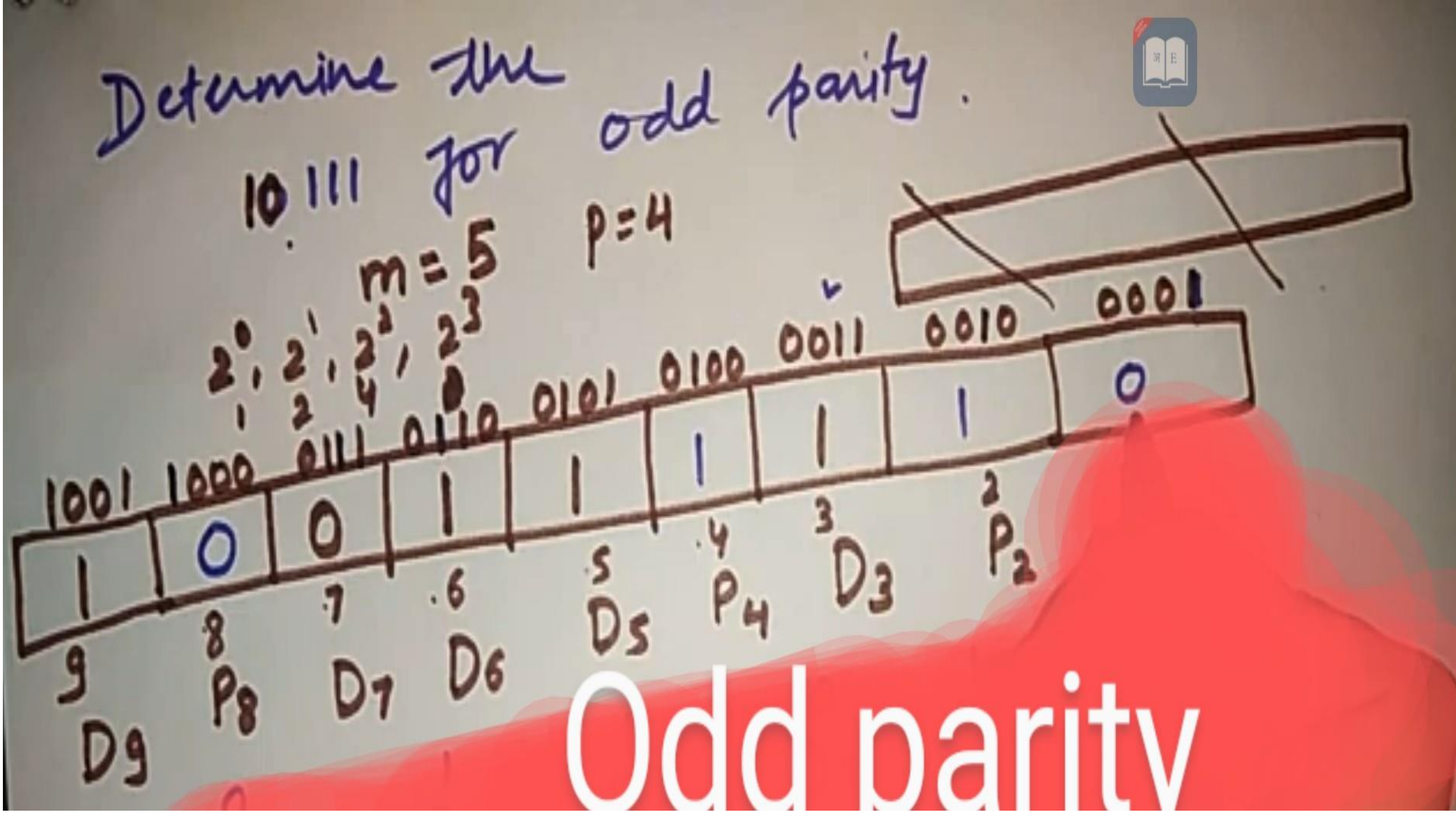
1010101

Determine the

10111 for odd parity.

m = 5 p = 4




Determine the
10111 for odd parity.
m = 5
p = 4
2^0, 2^1, 2^2, 2^3
1 2 4 8
1001 1000 0111 0110 0101 0100 0011 0010 0001
1 0 0 1 1 1 1 1 0
9 8 7 6 5 4 3 2
D9 P8 D7 D6 D5 P4 D3 P2
Odd parity

Parts of a Frame

A frame has the following parts –

- Frame Header – It contains the source and the destination addresses of the frame.
- Payload field – It contains the message to be delivered.
- Trailer – It contains the error detection and error correction bits.
- Flag – It marks the beginning and end of the frame.

Framing

- ★ When node A wishes to transmit a frame to node B, it tells its adaptor to transmit a frame from the node's memory.
- ★ This results in a sequence of bits being sent over the link.
- ★ The adaptor on node B then collects together the sequence of bits arriving on the link and deposits the corresponding frame in B's memory.

Types of framing

1. Fixed-size framing.

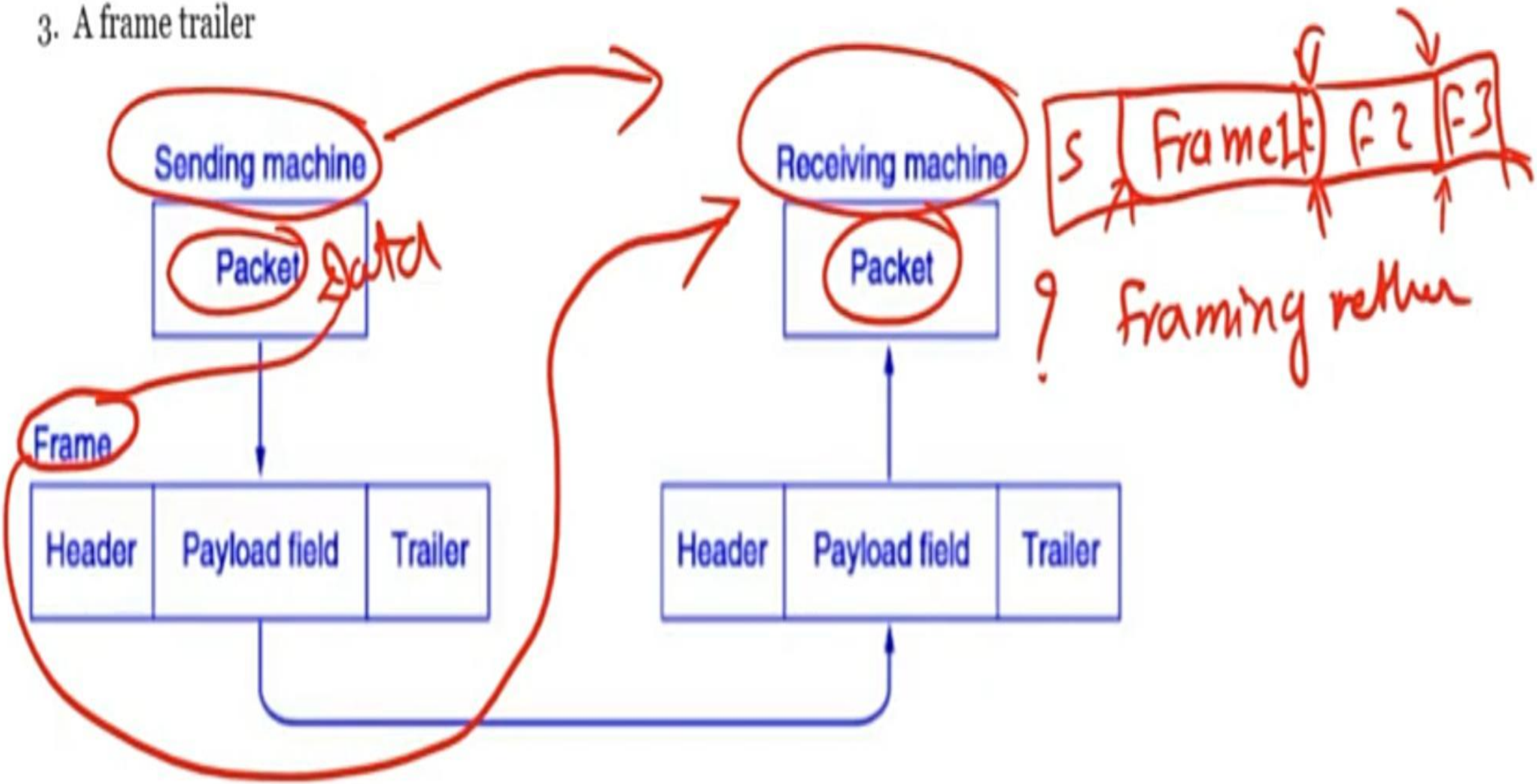
- ★ Here the size of the frame is fixed and so the frame length acts as delimiter of the frame.
- ★ Consequently, it does not require additional boundary bits to identify the start and end of the frame.

2. Variable-size framing.

- ★ Here, the size of each frame to be transmitted may be different.

- Two methods of framing
- Character oriented framing method
- Bit oriented framing method
- Why use this method in framing technique?

3. A frame trailer



Character - Oriented Framing

In character - oriented framing, data is transmitted as a sequence of bytes, from an 8-bit coding system like ASCII. The parts of a frame in a character - oriented framing are –

Frame Header – It contains the source and the destination addresses of the frame in form of bytes.

Payload field – It contains the message to be delivered. It is a variable sequence of data bytes.

Trailer – It contains the bytes for error detection and error correction.

Flags – Flags are the frame delimiters signalling the start and end of the frame. It is of 1- byte denoting a protocol - dependent special character.

2. Flag bytes with byte stuffing framing method:

1. In this method a flag byte is used as both the starting and ending of a frame. See in the figure below.
2. Two consecutive flag bytes indicate the end of one frame and the start of the next frame.
3. If the receiver ever loses synchronization it can just search for two flag bytes to find the end of the current frame and the start the next frame.

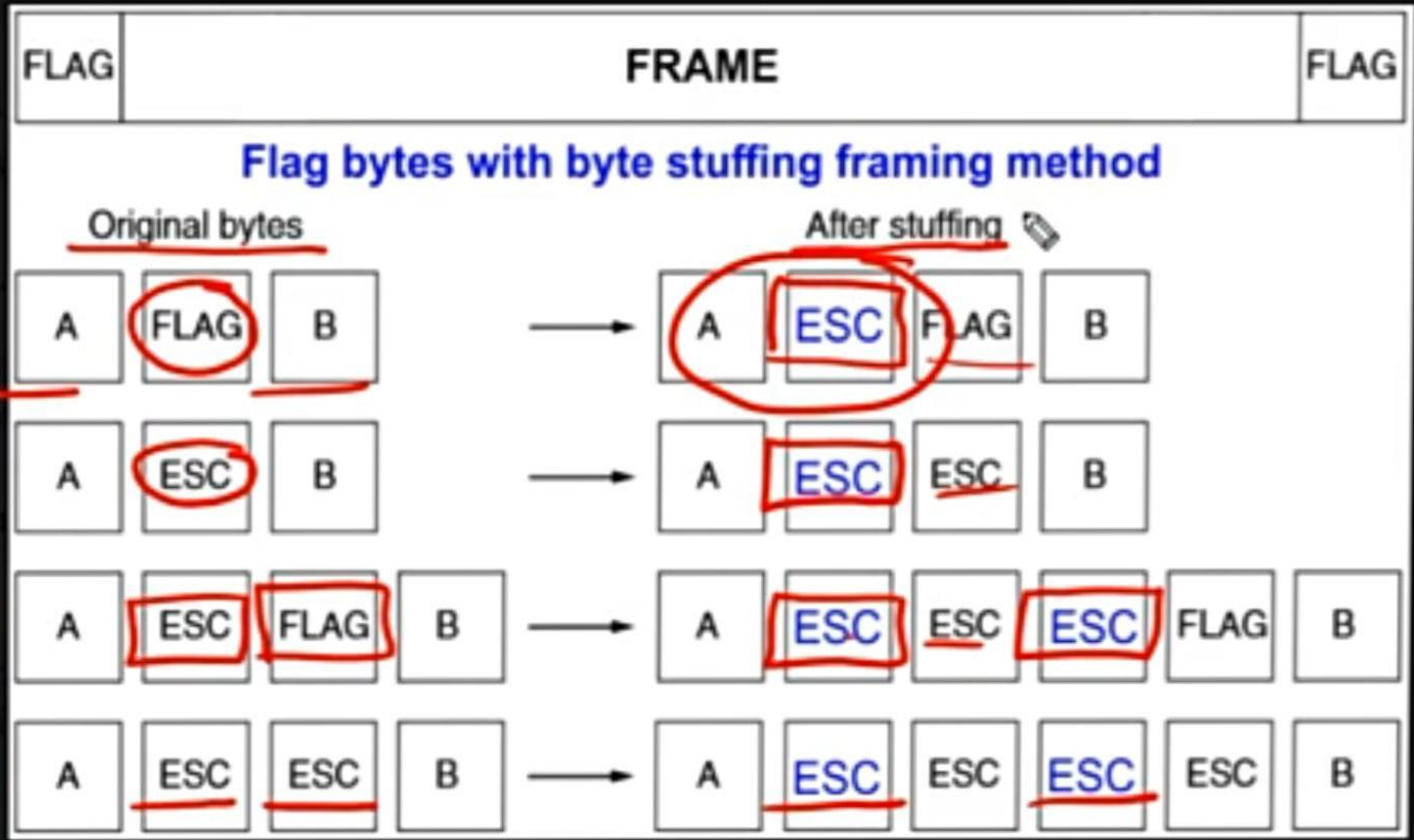
What if accidental unwanted flag bytes occur in the data?

To prevent from wrong flag bytes, escape byte (ESC) is inserted just before each "accidental" flag byte in the data. As shown in the figure below.

The data link layer on the receiving end removes the ESC before giving the data to the network layer. This technique is called **byte stuffing**.

What if accidental unwanted ESC occur in the data?

This unwanted ESC is also stuffed with an escape byte (ESC). As shown in the figure below.


FLAG
FRAME
FLAG
Flag bytes with byte stuffing framing method
Original bytes
After stuffing
A
FLAG
B
A
ESC
FLAG
B
A
ESC
B
A
ESC
ESC
B
A
ESC
FLAG
B
A
ESC
ESC
ESC
FLAG
B
A
ESC
ESC
B
A
ESC
ESC
ESC
ESC
B

Bit-oriented framing

In bit-oriented framing, data is transmitted as a sequence of bits that can be interpreted in the upper layers both as text as well as multimedia data.

The parts of a frame in a character - oriented framing are –

Frame Header – It contains bits denoting the source and the destination addresses of the frame.

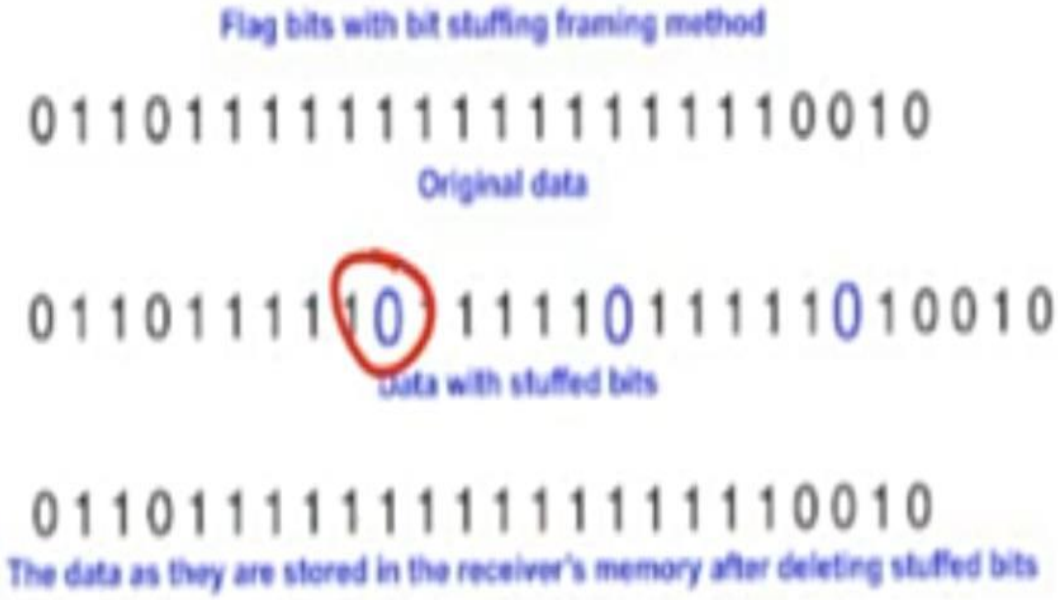
Payload field – It contains the message to be delivered. It is a variable sequence of bits.

Trailer – It contains the error detection and error correction bits.

Flags – Flags are a bit pattern that act as the frame delimiters signalling the start and end of the frame. It is generally of 8-bits and comprises of six or more consecutive 1s. Most protocols use the 8-bit pattern 01111110 as flag.

3. Flag bits with bit stuffing framing method: Bit level

1. In this method bit stuffing is used.
2. When sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a 0 bit.
3. At receiver end this stuffed 0 bit automatically deleted. As shown in the figure below.



Flag bits with bit stuffing framing method

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

Original data

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

Data with stuffed bits

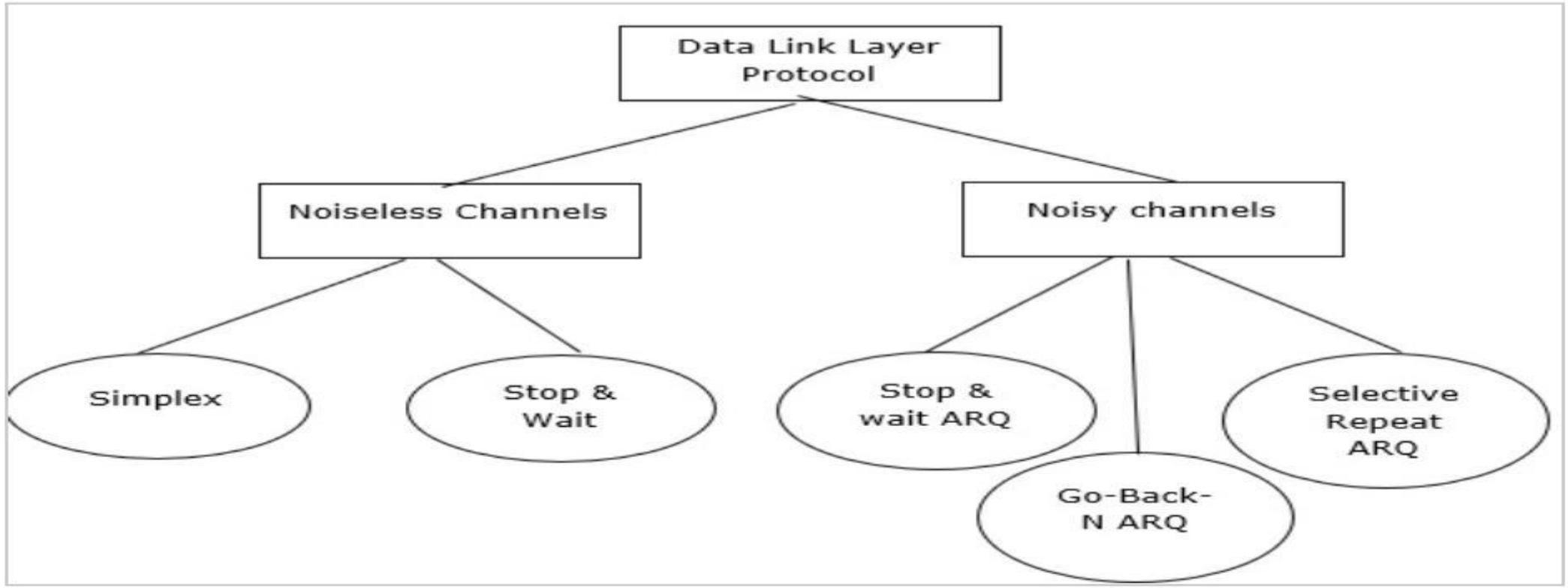
0 1 1 0 1 1 1 1 1 * 1 1 1 1 1 * 1 1 1 1 1 * 1 0 0 1 0

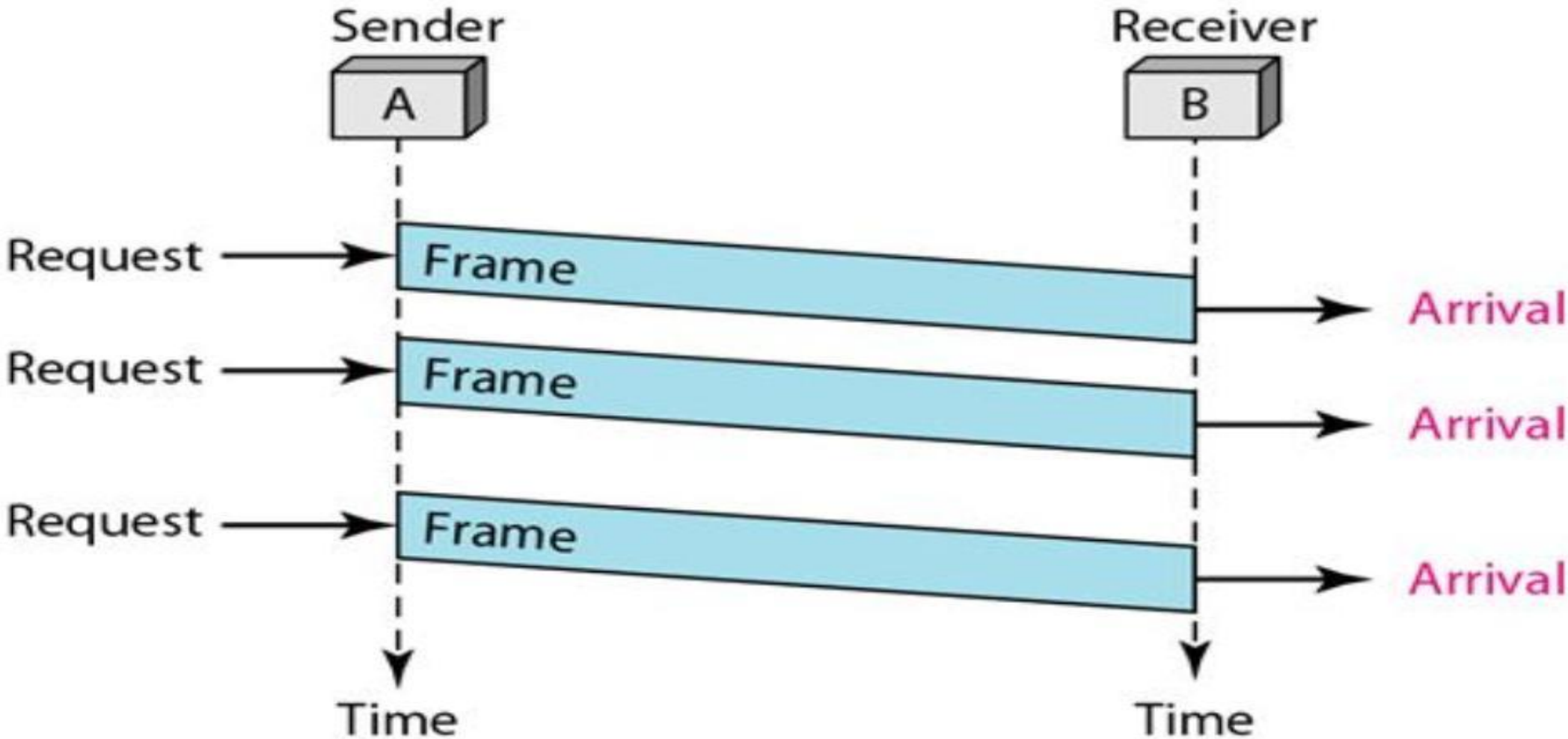
The data as they are stored in the receiver's memory after deleting stuffed b

Top

Data link layer protocols are divided into two categories based on whether the transmission channel is noiseless or noisy.

The data link layer protocol is diagrammatically represented below –




Sender
A
Receiver
B
Request
Frame
Arrival
Request
Frame
Arrival
Request
Frame
Arrival
Time
Time

Simplest Protocol

Step 1 – Simplest protocol that does not have flow or error control.

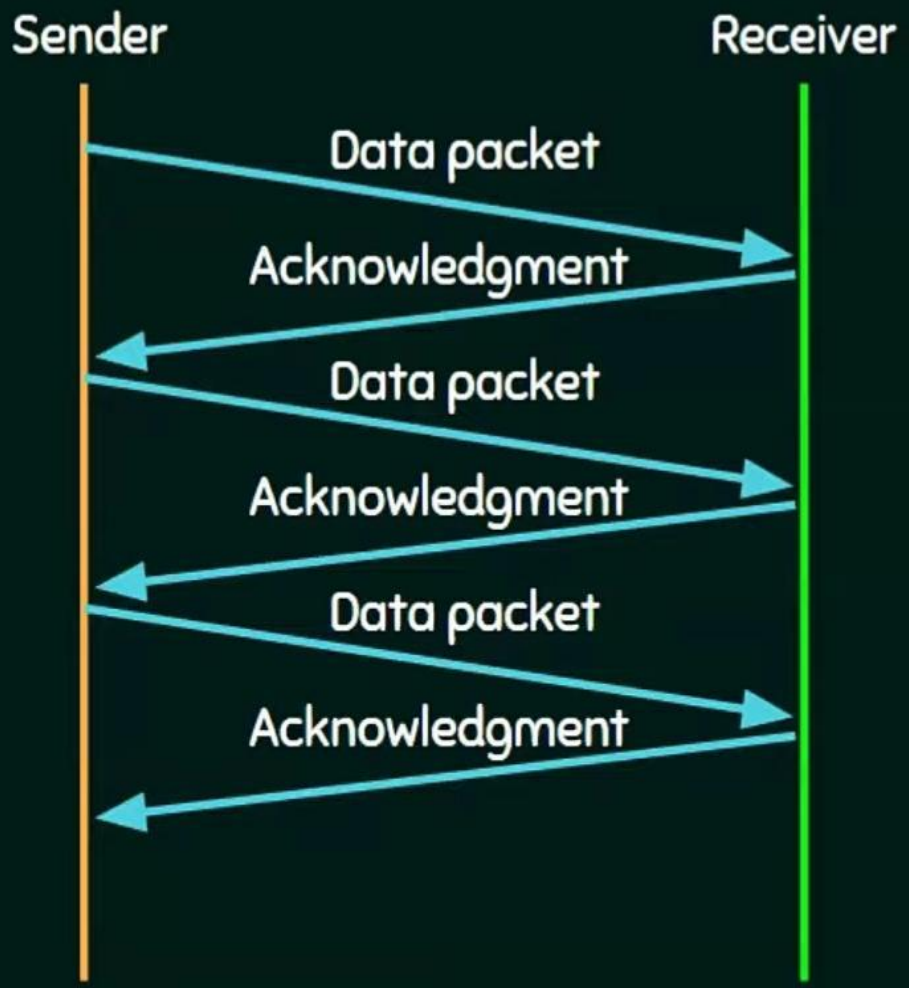
Step 2 – It is a unidirectional protocol where data frames are traveling in one direction that is from the sender to receiver.

Step 3 – Let us assume that the receiver can handle any frame it receives with a processing time that is small enough to be negligible, the data link layer of the receiver immediately removes the header from the frame and hands the data packet to its network layer, which can also accept the packet immediately.

Stop-and-Wait Protocol

- ★ Stop - and - Wait protocol is data link layer protocol for transmission of frames over noiseless channels.
- ★ It provides unidirectional data transmission with flow control facilities but without error control facilities.
- ★ The idea of stop-and-wait protocol is straightforward.
- ★ After transmitting one frame, the sender waits for an acknowledgement before transmitting the next frame.

Stop-and-Wait Protocol



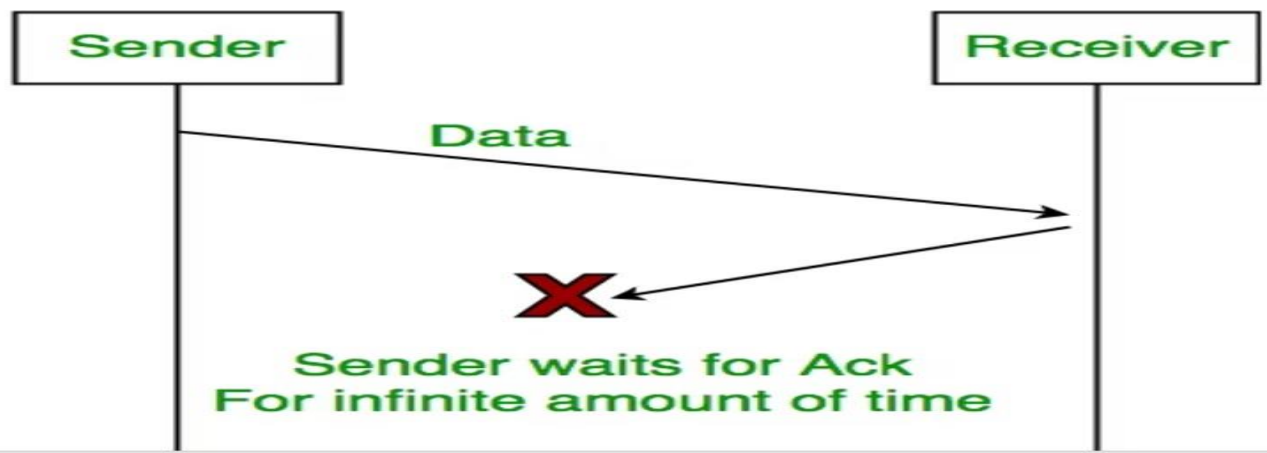
Stop-and-Wait ARQ Protocol

- ★ Idea of stop-and-wait protocol is straightforward.
- ★ After transmitting one frame, the sender waits for an acknowledgement before transmitting the next frame.
- ★ If the acknowledgement does not arrive after a certain period of time, the sender times out and retransmits the original frame.
- ★ Stop-and-Wait ARQ = Stop-and-Wait + Timeout Timer + Sequence number

Problems :

1. Lost Data

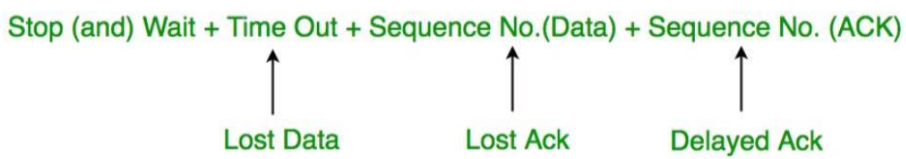
2. Lost Acknowledgement:



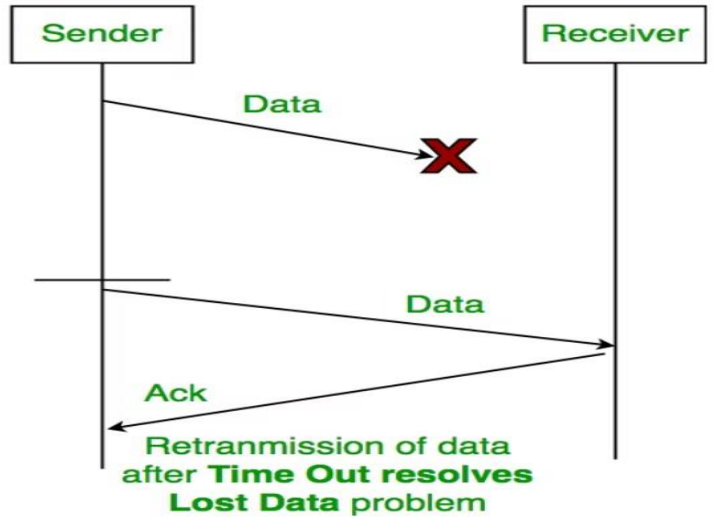
3. Delayed Acknowledgement/Data: After a timeout on the sender side, a long-delayed acknowledgement might be wrongly considered as acknowledgement of some other recent packet.

Stop and Wait for ARQ (Automatic Repeat Request)

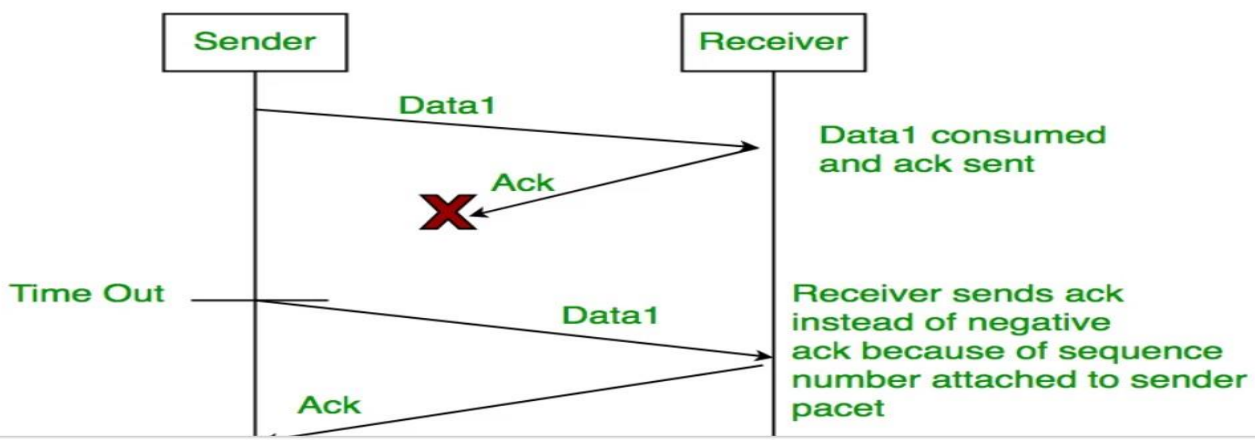
The above 3 problems are resolved by Stop and Wait for ARQ (Automatic Repeat Request) that does both error control and flow control.



1. Time Out:



2. Sequence Number (Data)

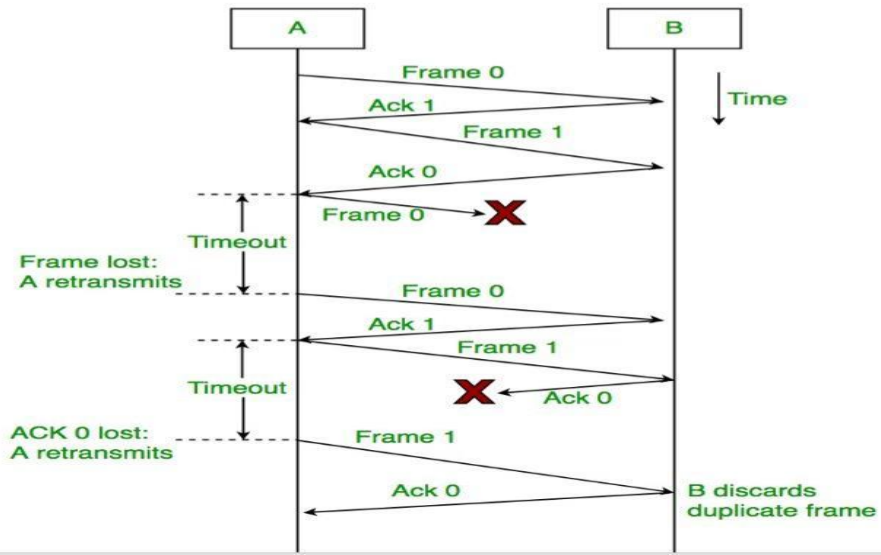


3. Delayed Acknowledgement:

This is resolved by introducing sequence numbers for acknowledgement also.

Working of Stop and Wait for ARQ:

1) Sender A sends a data frame or packet with sequence number 0.
2) Receiver B, after receiving the data frame, sends an acknowledgement with sequence number 1 (the sequence number of the next expected data frame or packet)
There is only a one-bit sequence number that implies that both sender and receiver have a buffer for one frame or packet only.



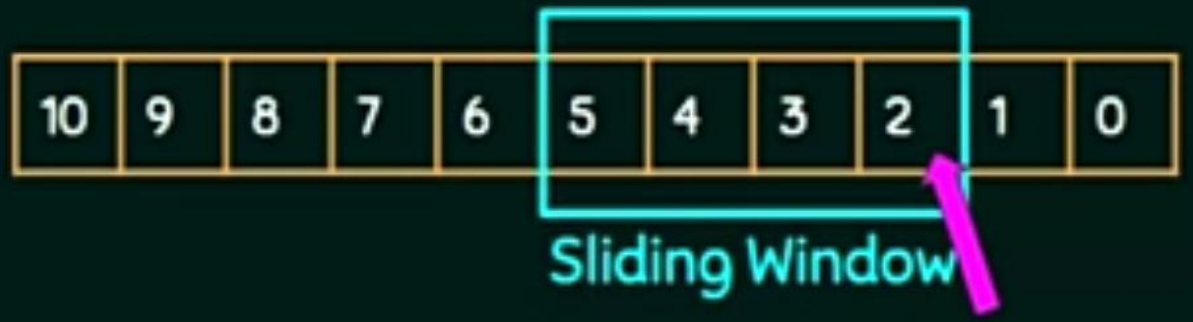
Go-Back-N ARQ

- ★ Go - Back - N ARQ uses the concept of protocol pipelining i.e. the sender can send multiple frames before receiving the acknowledgment for the first frame.
- ★ There are finite number of frames and the frames are numbered in a sequential manner.
- ★ The number of frames that can be sent depends on the window size of the sender.
- ★ If the acknowledgment of a frame is not received within an agreed upon time period, all frames in the current window are transmitted.

Go-Back-N ARQ

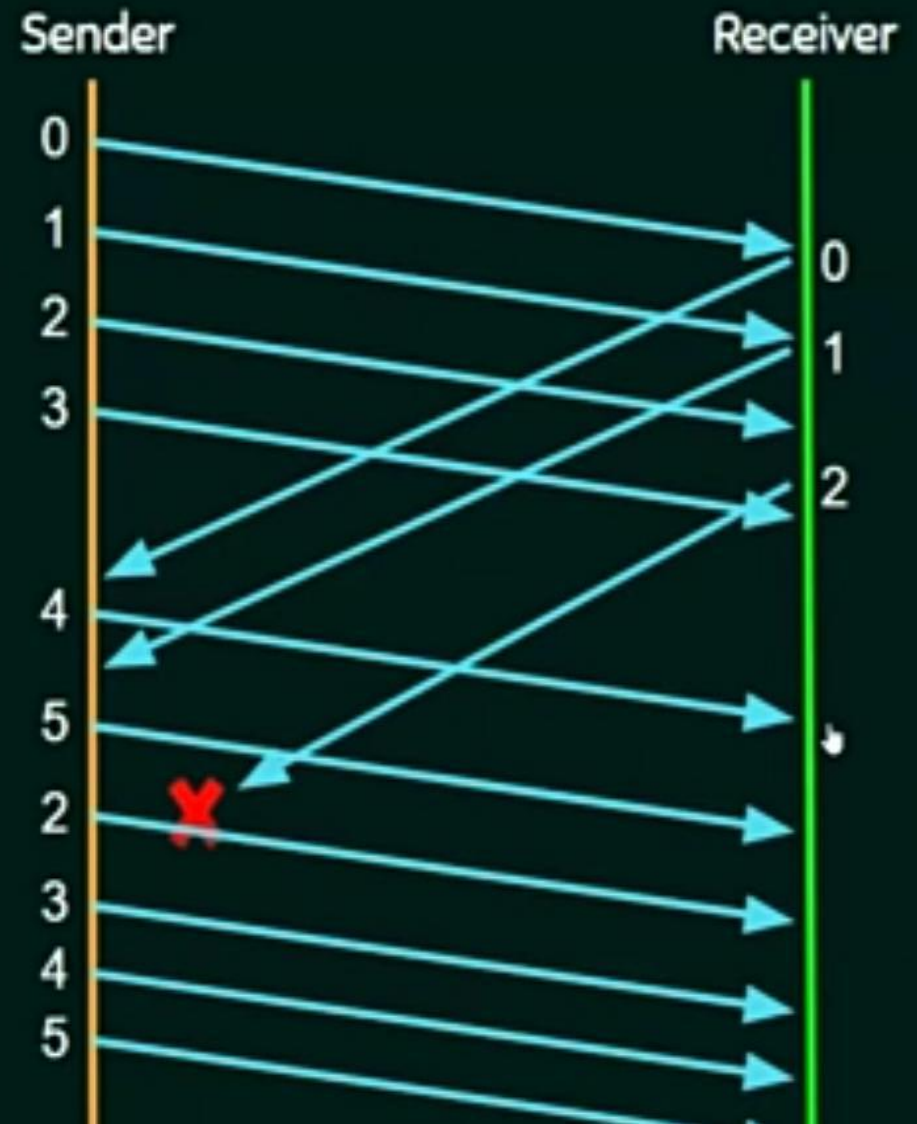
- ★ N - Sender's Window Size.
- ★ For example, if the sending window size is 4 (2^2), then the sequence numbers will be 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, and so on.
- ★ The number of bits in the sequence number is 2 to generate the binary sequence 00, 01, 10, 11.

Working of Go-Back-N ARQ



Go-Back to 2

Window Size: 4

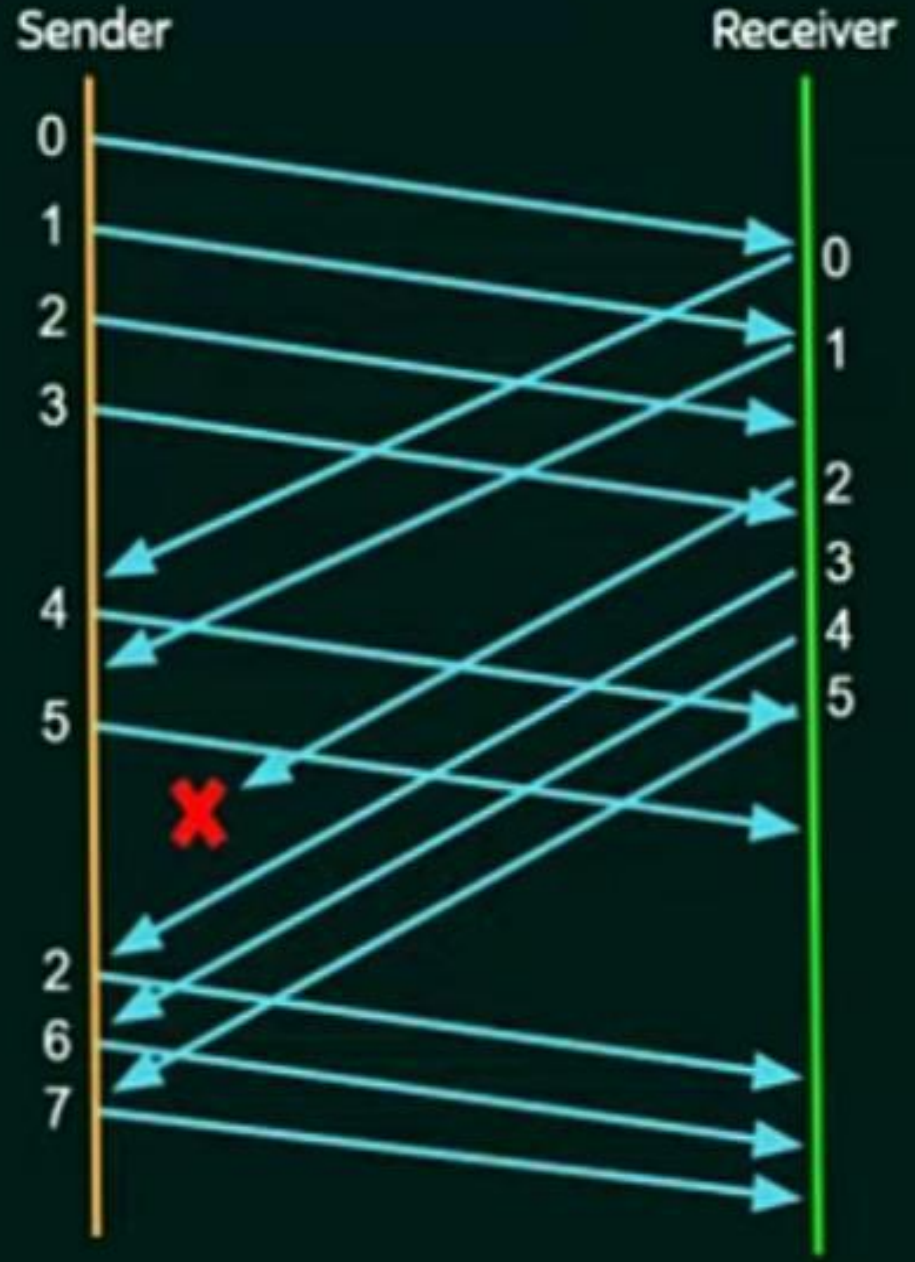


Selective Repeat ARQ

- ★ In Selective Repeat ARQ, only the erroneous or lost frames are retransmitted, while correct frames are received and buffered.
- ★ The receiver while keeping track of sequence numbers, buffers the frames in memory and sends NACK for only frame which is missing or damaged.
- ★ The sender will send/retransmit packet for which NACK is received.

Working of Selective Repeat

Window Size: 4



Piggybacking

Sending of Ack is delayed until the next data frame is available for Tx. the Ack is hooked into the outgoing dataframe. the size of Ack field is only a few bits.