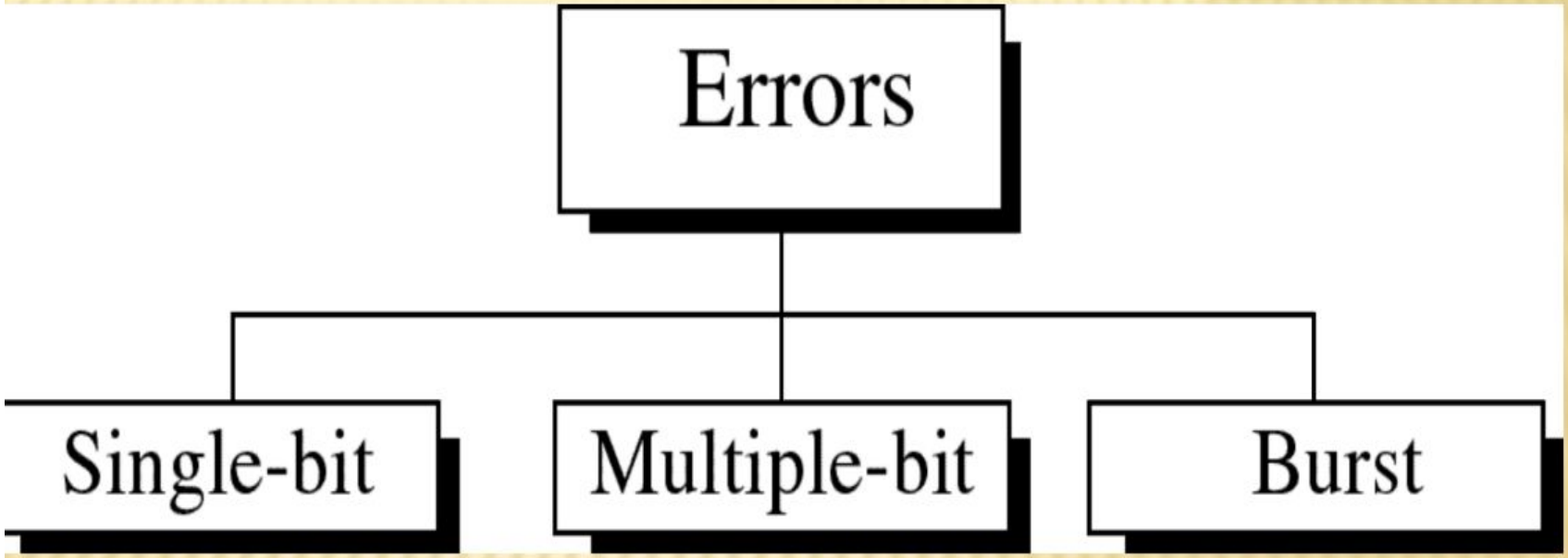
Error Detection and Correction

- Errors in data transmission can occur for a variety of reasons, such as electromagnetic interference, noise, faulty hardware, or even software glitches.
- These errors can lead to the alteration, insertion, deletion, or inversion of bits within the transmitted data, thereby compromising the accuracy and authenticity of the information

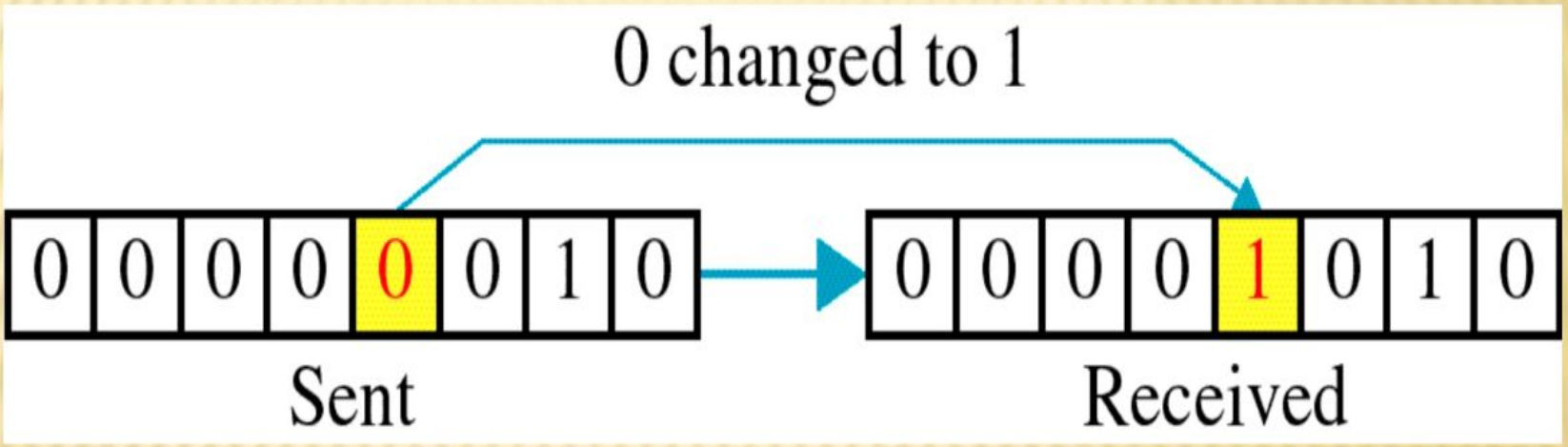
Basic concepts

- Networks must be able to transfer data from one device to another with complete accuracy.
- Data can be corrupted during transmission.
- For reliable communication, errors must be detected and corrected.
- **Error detection and correction** are implemented either at the **data link layer** or the **transport layer** of the OSI model.

Types of Errors



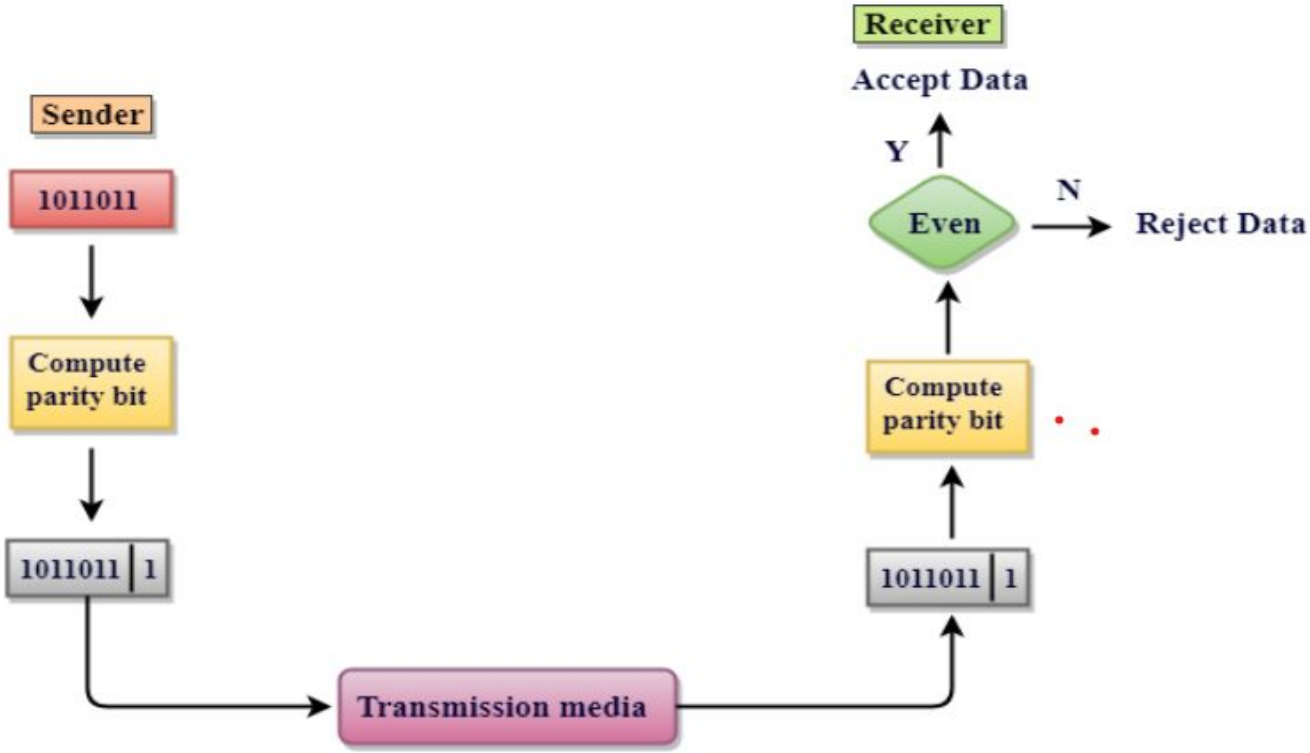
Single-bit error



Single bit errors are the **least likely** type of errors in serial data transmission because the noise must have a very short duration which is very rare. However this kind of errors can happen in parallel transmission.

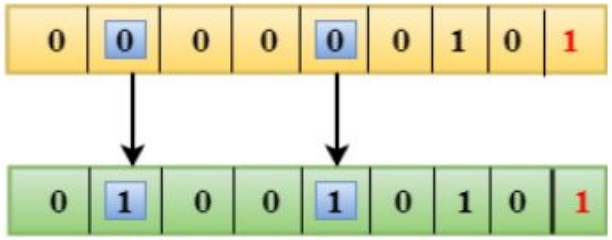
Example:

- If data is sent at 1Mbps then each bit lasts only 1/1,000,000 sec. or 1 μs.
- For a single-bit error to occur, the noise must have a duration of only 1 μs, which is very rare.

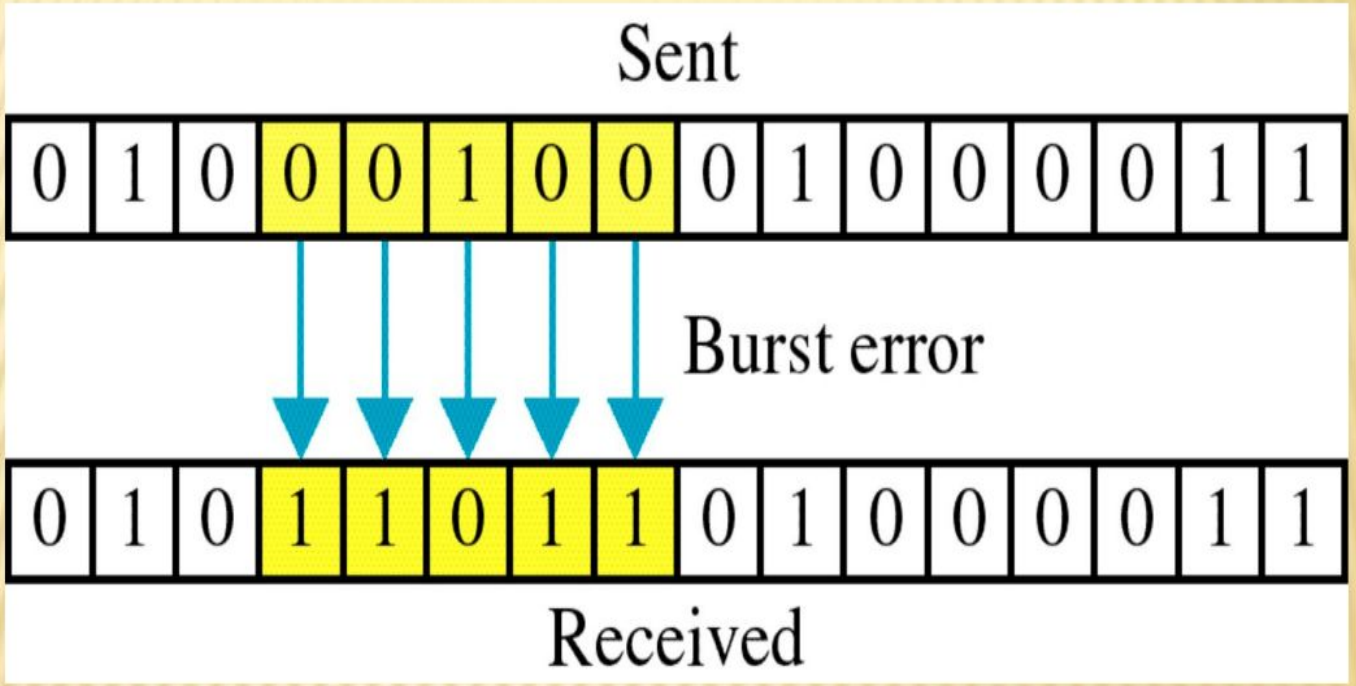


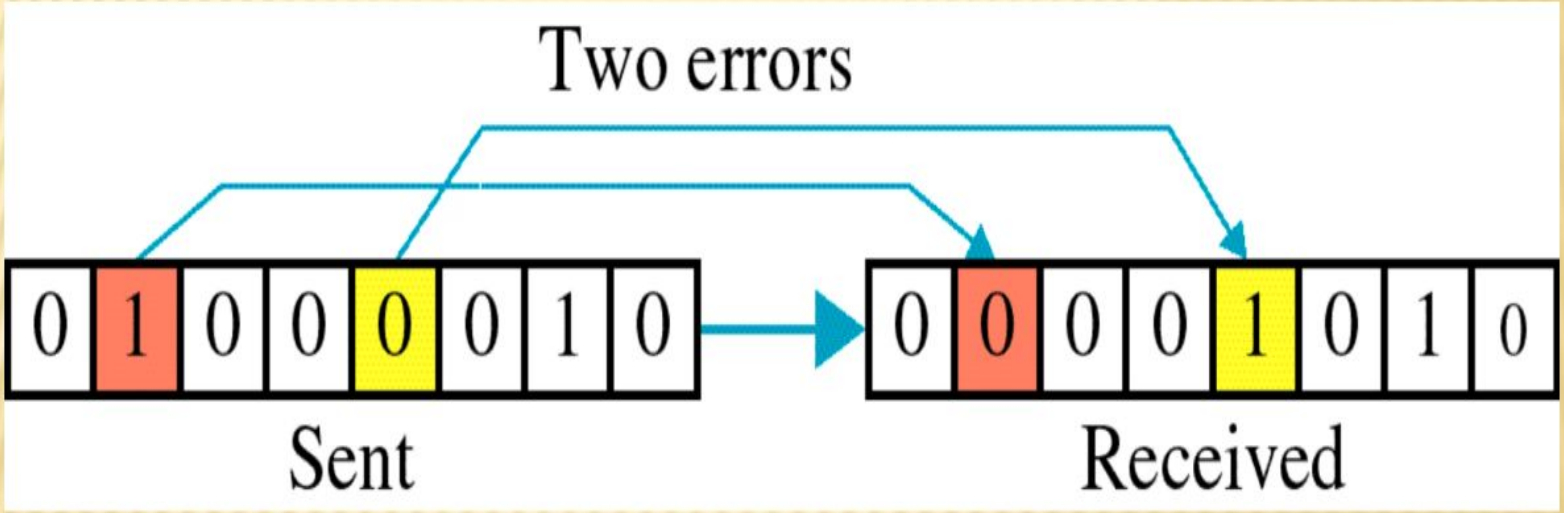
Drawbacks Of Single Parity Checking

- It can only detect single-bit errors which are very rare.
- If two bits are interchanged, then it cannot detect the errors.



Burst error


Sent
0 1 0 0 0 1 0 0 0 1 0 0 0 0 1 1
Burst error
0 1 0 1 1 0 1 1 0 1 0 0 0 0 1 1
Received


Two errors
0 1 0 0 0 0 1 0
0 0 0 0 1 0 1 0
Sent
Received

- **Burst error is most likely to happen in serial transmission** since the duration of noise is normally longer than the duration of a bit.
- The number of bits affected depends on the data rate and duration of noise.

Example:

- If data is sent at rate = 1Kbps then a noise of 1/100 sec can affect 10 bits.(1/100*1000)

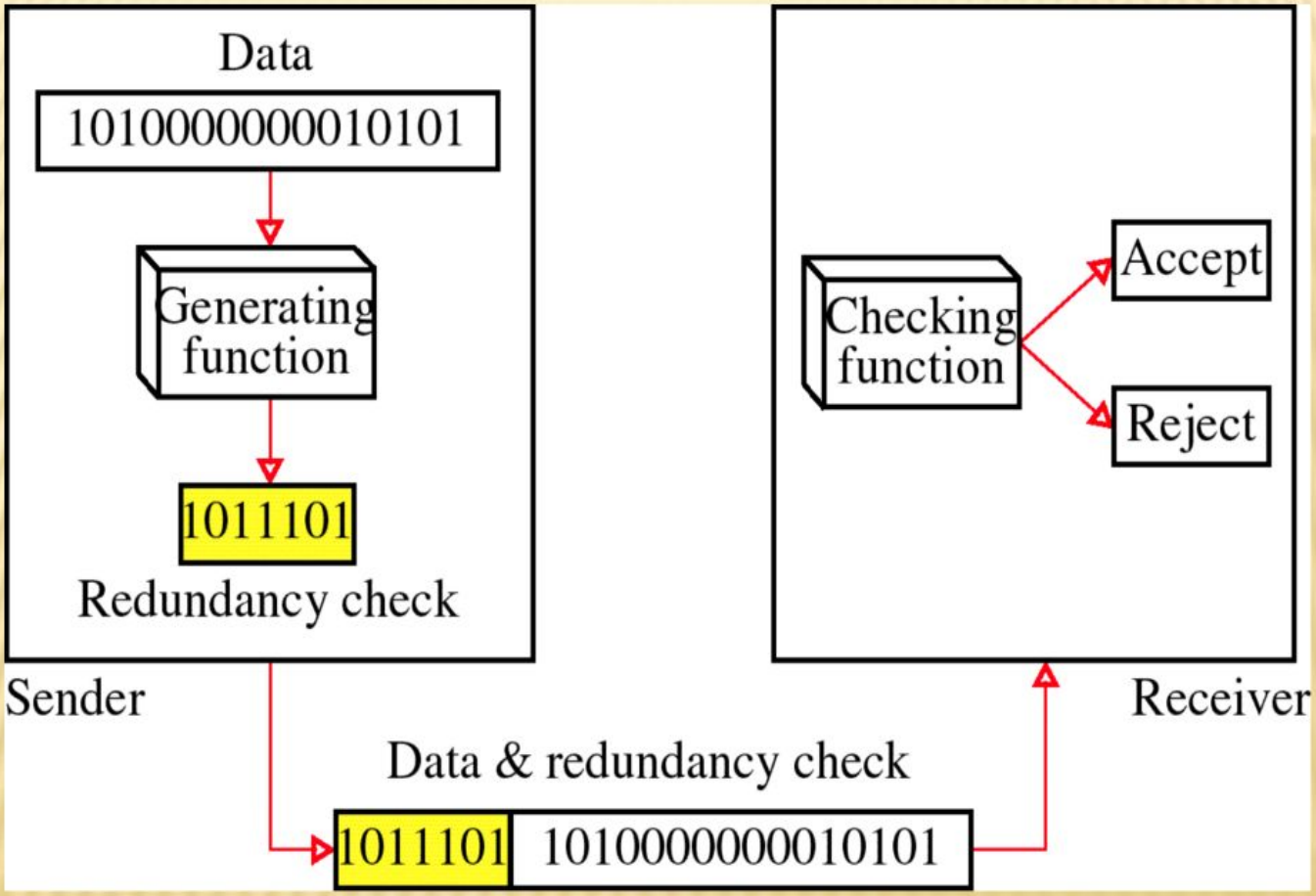
- If same data is sent at rate = 1Mbps then a noise of 1/100 sec can affect 10,000 bits.($1/100*10^6$)

ERROR DETECTION

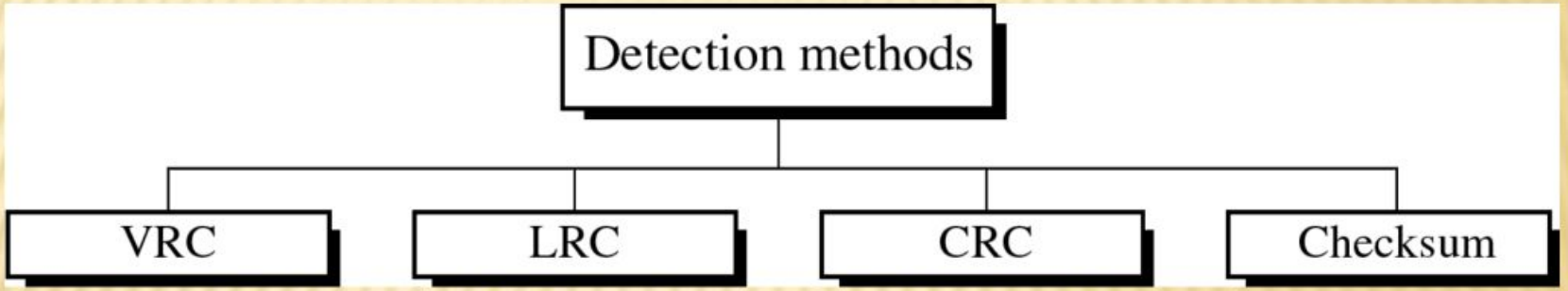
Error detection means to decide whether the received data is correct or not without having a copy of the original message.

Error detection **uses the concept of redundancy**, **which means** adding extra bits for detecting errors at the destination.

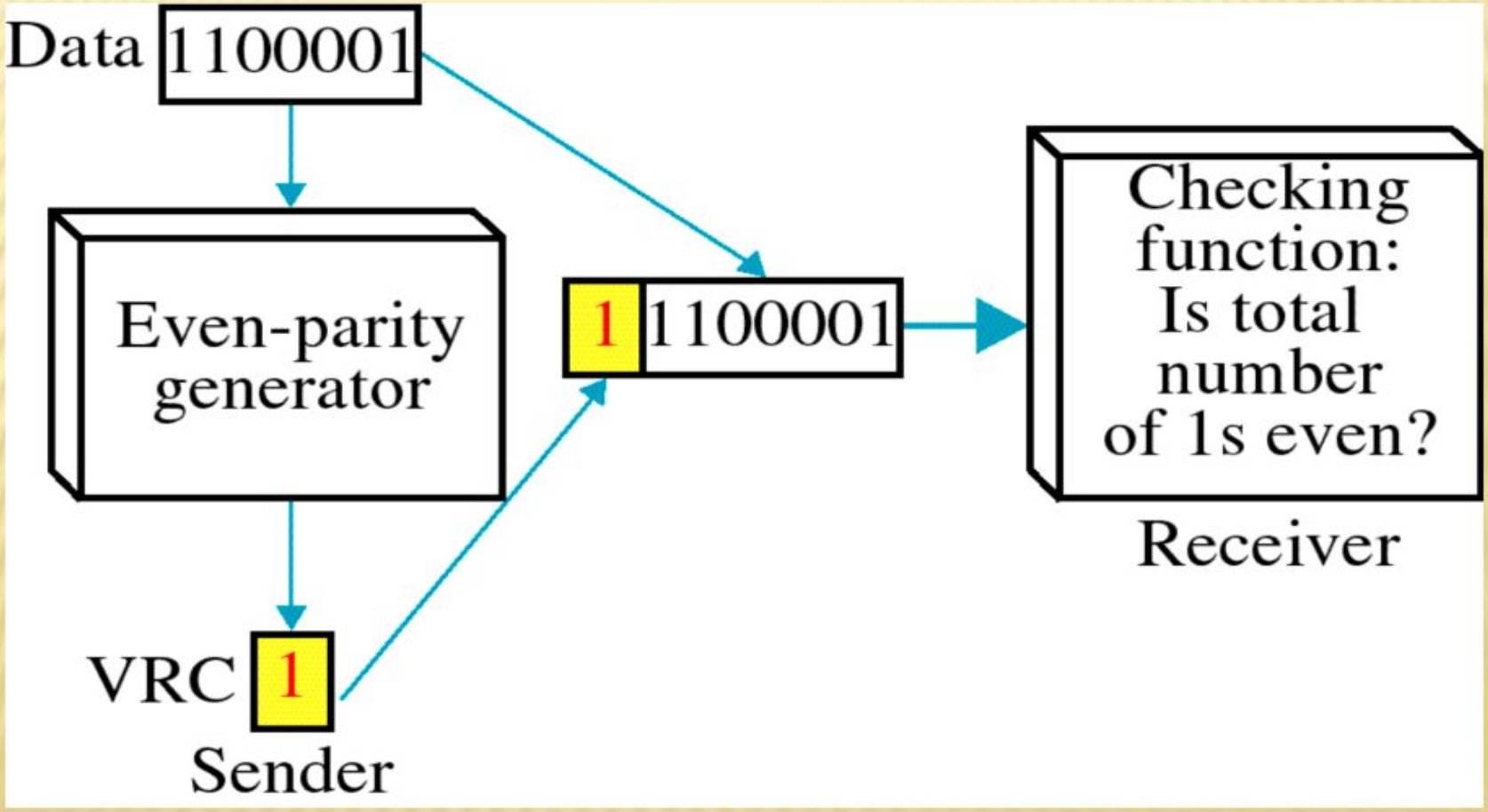
Redundancy


Data
1010000000010101
Generating function
1011101
Redundancy check
Sender
Checking function
Accept
Reject
Receiver
Data & redundancy check
1011101
1010000000010101

Four types of redundancy checks are used in data communications



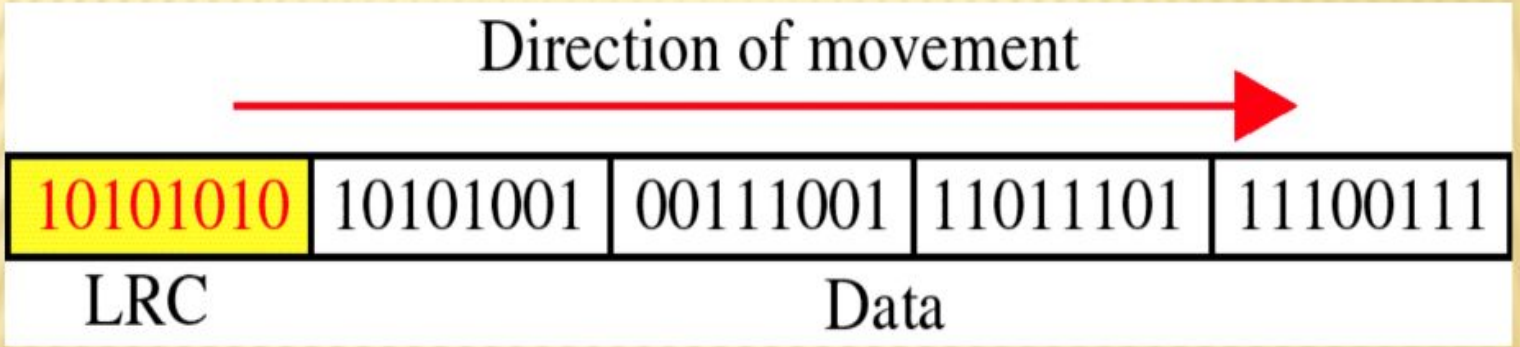
Vertical Redundancy Check
VRC



PERFORMANCE

- It can detect single bit error
- It can detect burst errors only if the total number of errors is odd.

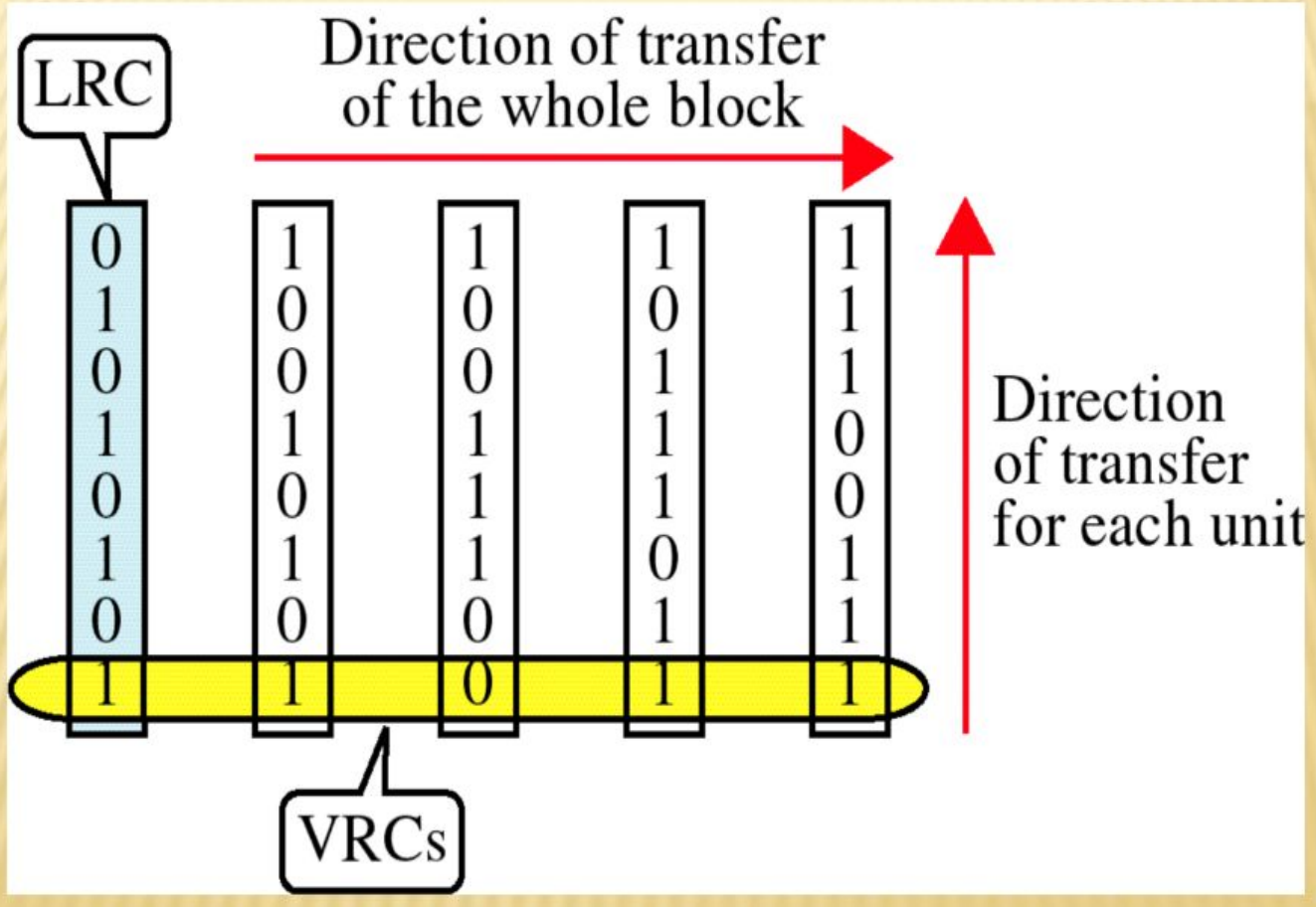
Longitudinal Redundancy Check
LRC



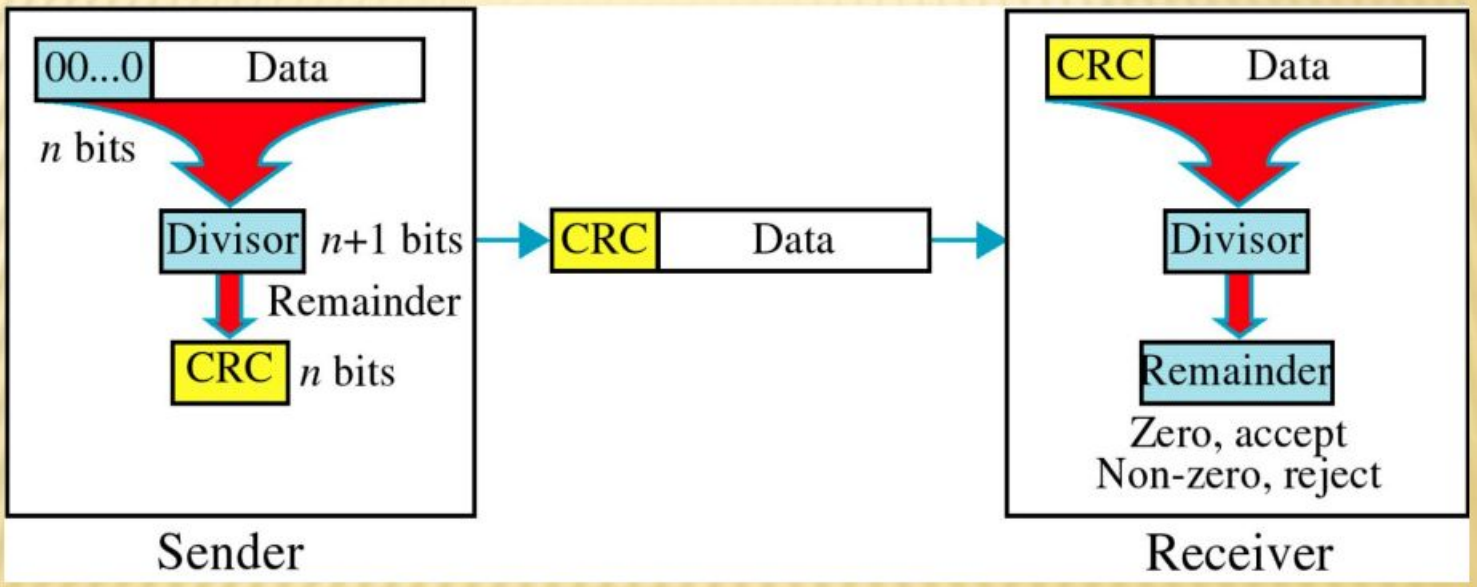
Performance

- LCR increases the likelihood of detecting burst errors.
- If two bits in one data units are damaged and two bits in exactly the same positions in another data unit are also damaged, the LRC checker will not detect an error.

VRC and LRC


LRC
Direction of transfer
of the whole block
0 1 0 1 0 1 0 1
1 0 0 1 0 1 0 1
1 0 0 1 1 1 1 0
1 0 1 1 1 0 1 1
1 1 1 0 0 1 1 1
Direction
of transfer
for each unit
VRCs

Cyclic Redundancy Check
CRC



CYCLIC REDUNDANCY CHECK

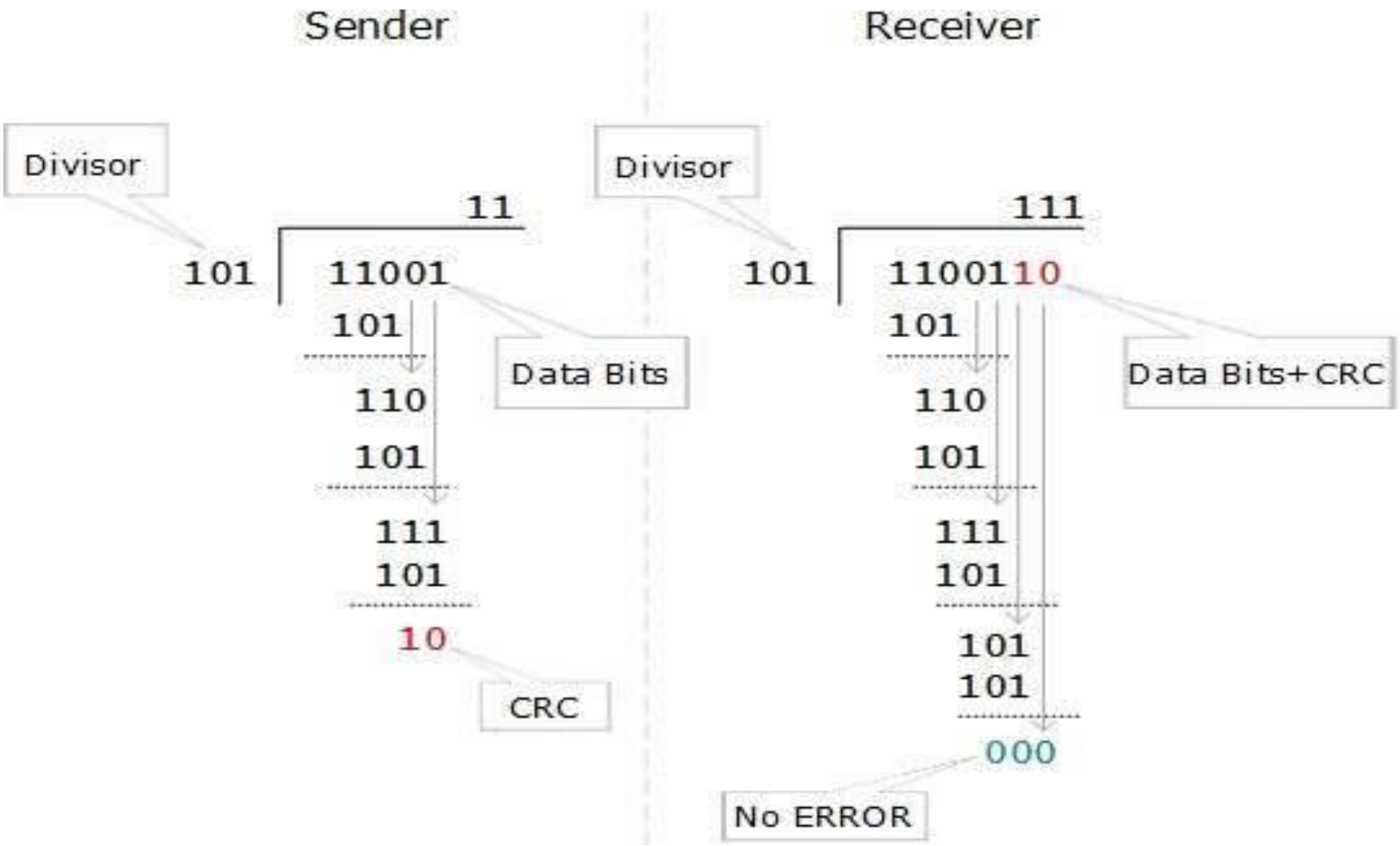
- Given a k-bit frame or message, the transmitter generates an n-bit sequence, known as a *frame check sequence (FCS)*, so that the resulting frame, consisting of (k+n) bits, is exactly divisible by some predetermined number.
- The receiver then divides the incoming frame by the same number and, if there is no remainder, assumes that there was no error.

CRC

- CRC is a different approach to detect if the received frame contains valid data. This technique involves binary division of the data bits being sent.

- The divisor is generated using polynomials. The sender performs a division operation on the bits being sent and calculates the remainder. Before sending the actual bits, the sender adds the remainder at the end of the actual bits. Actual data bits plus the remainder is called a codeword. The sender transmits data bits as codewords.

CRC


Sender
Receiver
Divisor
Divisor
11
101
11001
101
110
101
111
101
10
Data Bits
CRC
111
101
1100110
101
110
101
111
101
101
101
000
Data Bits+CRC
No ERROR

Checksum

Checksum

A Checksum is an error detection technique based on the concept of redundancy.

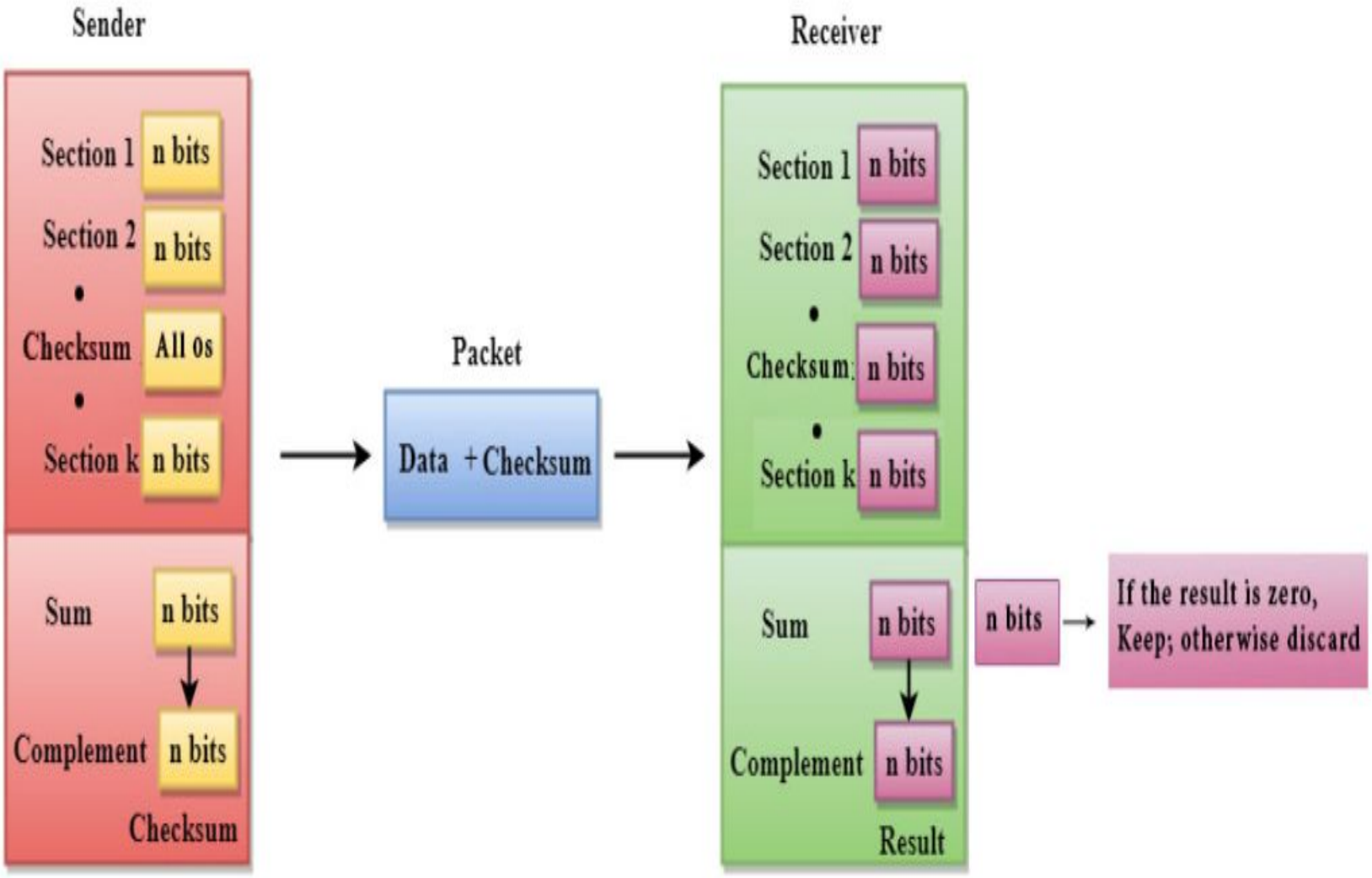
It is divided into two parts:

Checksum Generator

A Checksum is generated at the sending side. Checksum generator subdivides the data into equal segments of n bits each, and all these segments are added together by using one's complement arithmetic. The sum is complemented and appended to the original data, known as checksum field. The extended data is transmitted across the network.

Checksum Checker

A Checksum is verified at the receiving side. The receiver subdivides the incoming data into equal segments of n bits each, and all these segments are added together, and then this sum is complemented. If the complement of the sum is zero, then the data is accepted otherwise data is rejected.


Sender
Receiver
Section 1
n bits
Section 2
n bits
Checksum
All 0s
Section k
n bits
Packet
Data + Checksum
Section 1
n bits
Section 2
n bits
Checksum
n bits
Section k
n bits
Sum
n bits
Complement
n bits
Checksum
Sum
n bits
Complement
n bits
Result
n bits
If the result is zero,
Keep; otherwise discard