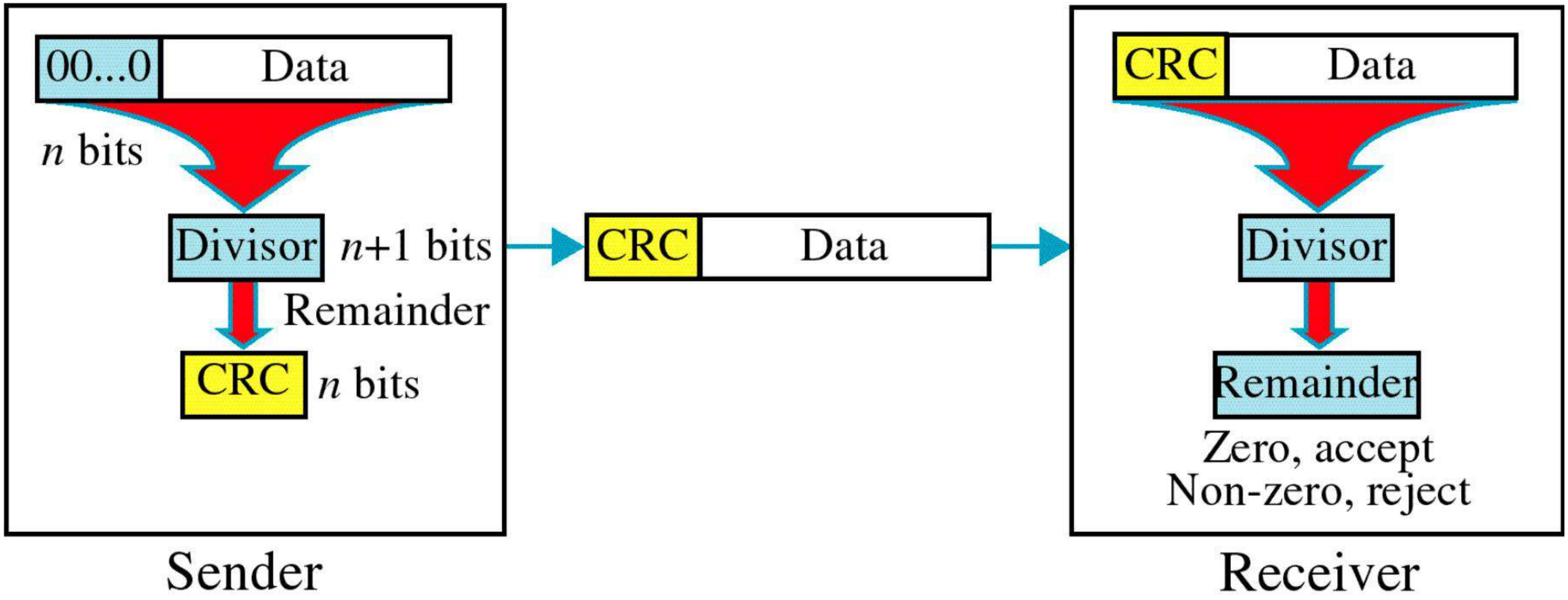
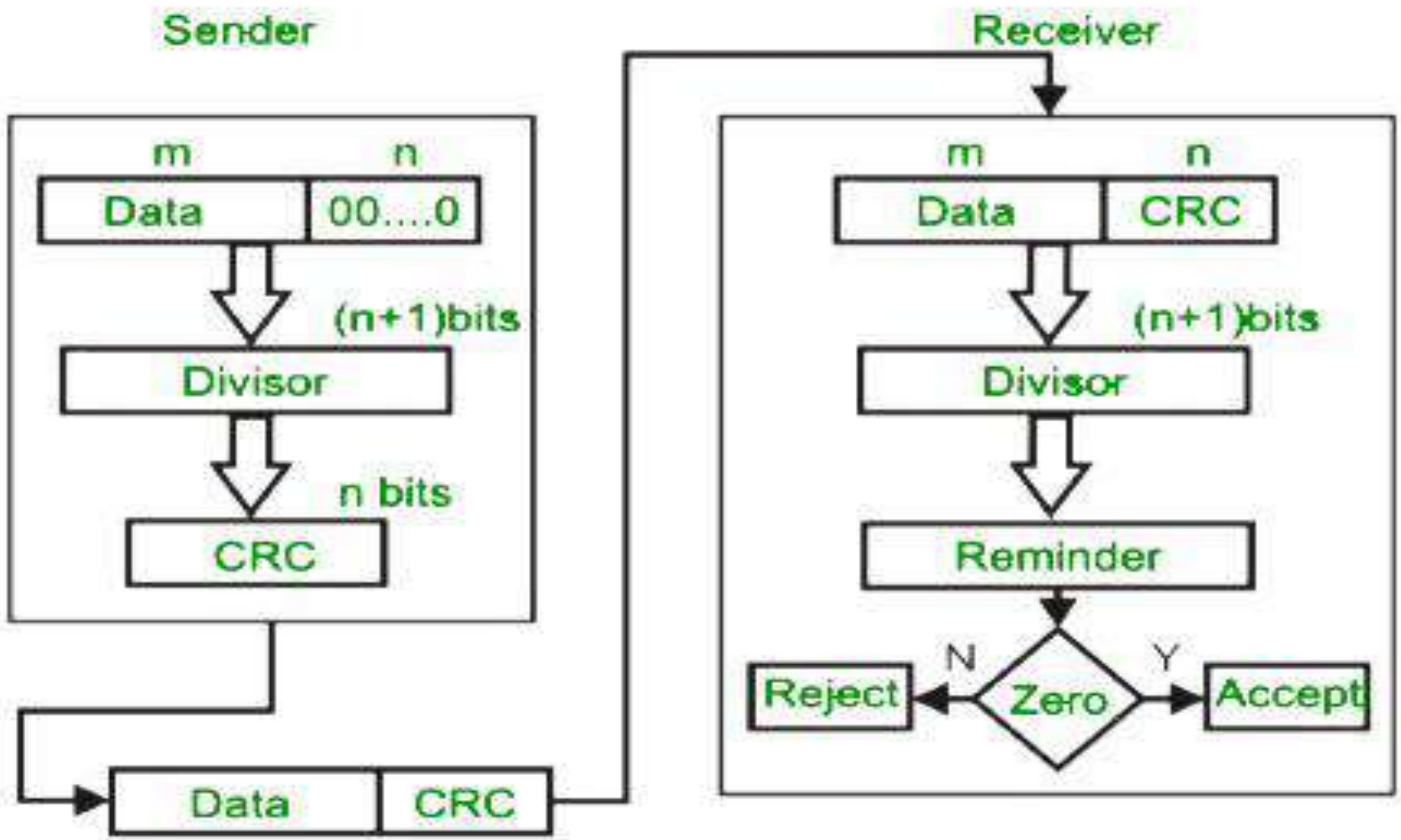
Cyclic Redundancy Check
CRC



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.

Cyclic Redundancy Check



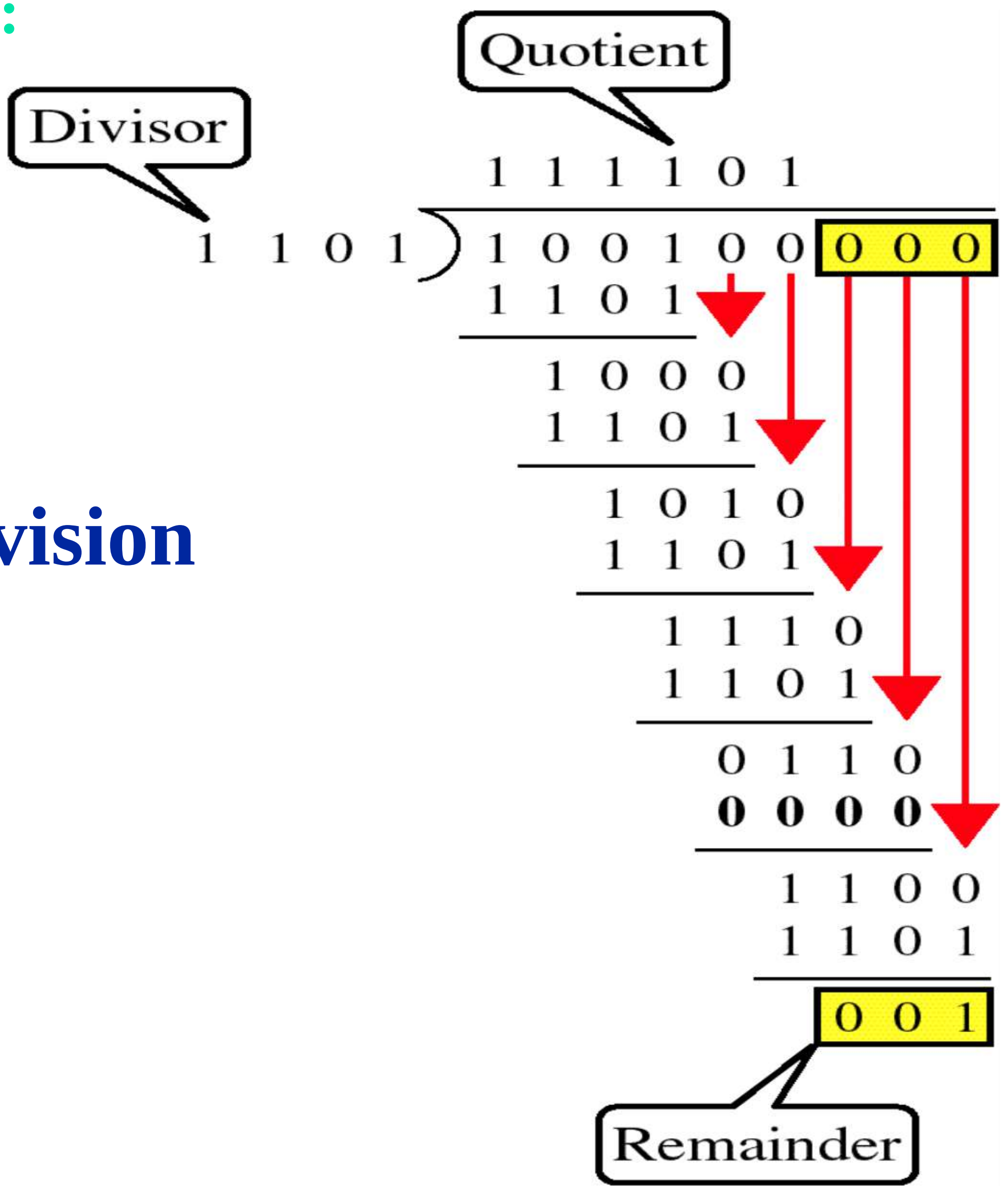
```
             1101010110  <- Q
       ________________
110101 ) 101000110100000
         110101
         ------
          111011
          110101
         ------
           111010
           110101
         -------
            111110
            110101
         -------
             101100
             110101
         ----------
              110010
              110101
         -----------
               01110 <- R
```

```
             1101010110
       ________________
110101 ) 101000110101110
         110101
         ------
          1110111
          1101101
         ------
           111010
           110101
         -------
            111110
            110101
         -------
             101100
             110101
         ----------
              110101
              110101
         -----------
                   0 <- R
```

Cyclic Redundancy Check

- **At Sender Side:**

Binary Division



Cyclic Redundancy Check

- **Three right most bits of your reminder are the CRC bits(001)**
- **Attach 001 CRC bits with data and then send to receiver.**

100100 001

Cyclic Redundancy Check

- **At Receiver Side:**

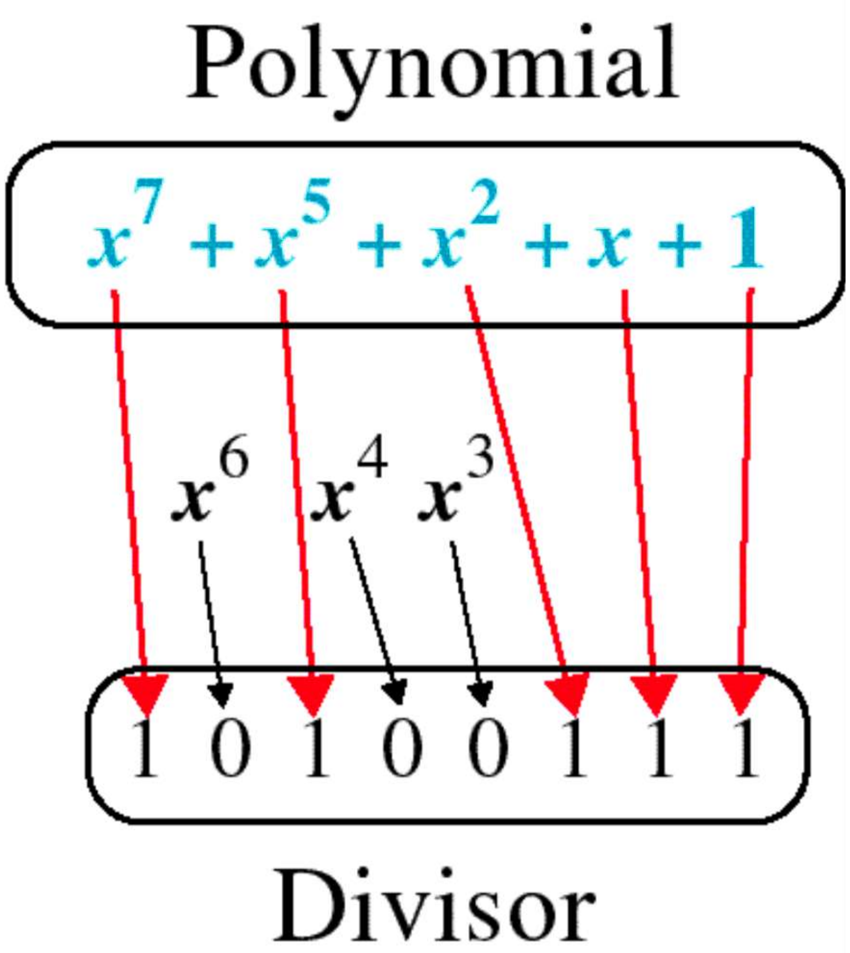
```
1101 )   100100001
         1101
          01000
           1101
           01010
            1101
             01110
              1101|
               001101
                 1101
                 0000  ← All zero's in reminder means No ERROR
```

Binary Division

Polynomial

$$\boxed{x^{7} + x^{5} + x^{2} + x + 1}$$

Polynomial and Divisor



Standard Polynomials

CRC-12

$$x^{12} + x^{11} + x^{3} + x + 1$$

CRC-16

$$x^{16} + x^{15} + x^{2} + 1$$

CRC-ITU

$$x^{16} + x^{12} + x^{5} + 1$$

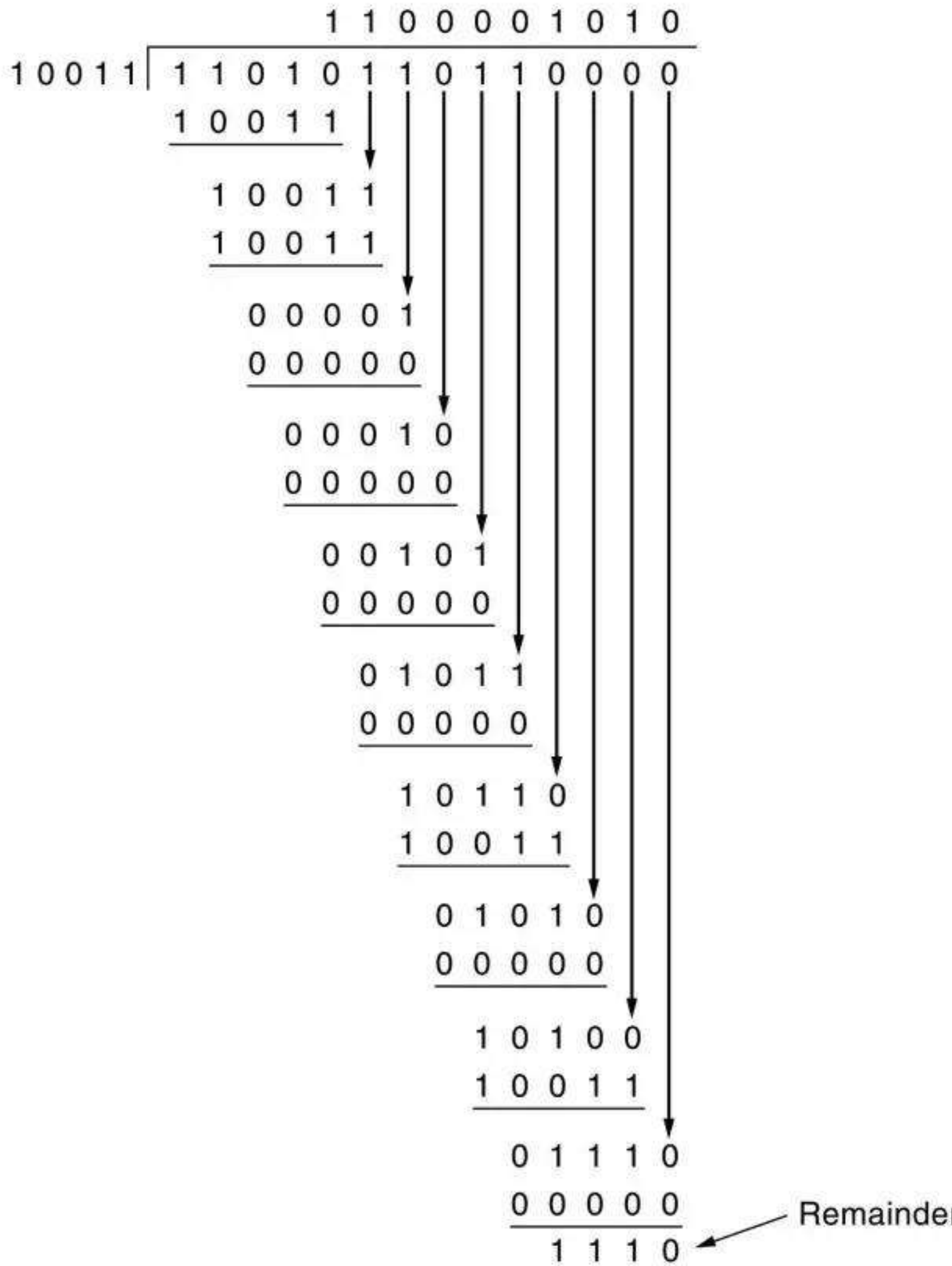
CRC-32

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

Problem 1

- A bit stream 1101011011 is transmitted using the standard CRC method. The generator polynomial is x^4+x+1. What is the actual bit string transmitted?

- **Solution-**
- The generator polynomial $G(x) = x^4 + x + 1$ is encoded as 10011.
- Clearly, the generator polynomial consists of 5 bits.
- So, a string of 4 zeroes is appended to the bit stream to be transmitted.
- The resulting bit stream is 110101101**10000**.


1 1 0 0 0 0 1 0 1 0
10011
1 1 0 1 0 1 1 0 1 1 0 0 0 0
1 0 0 1 1
1 0 0 1 1
1 0 0 1 1
0 0 0 0 1
0 0 0 0 0
0 0 0 1 0
0 0 0 0 0
0 0 1 0 1
0 0 0 0 0
0 1 0 1 1
0 0 0 0 0
1 0 1 1 0
1 0 0 1 1
0 1 0 1 0
0 0 0 0 0
1 0 1 0 0
1 0 0 1 1
0 1 1 1 0
0 0 0 0 0
1 1 1 0
Remainder

7.70

- From here, CRC = 1110.
- Now,
- The code word to be transmitted is obtained by replacing the last 4 zeroes of 11010110110000 with the CRC.
- Thus, the code word transmitted to the receiver = 11010110111110.

Problem 2

- A bit stream 10011101 is transmitted using the standard CRC method. The generator polynomial is x^3+1.

1. What is the actual bit string transmitted?
2. Suppose the third bit from the left is inverted during transmission. How will receiver detect this error?

Part-01:

The generator polynomial $G(x) = x^3 + 1$ is encoded as 1001.

- Clearly, the generator polynomial consists of 4 bits.
- So, a string of 3 zeroes is appended to the bit stream to be transmitted.
- The resulting bit stream is 10011101**000**.

```
              10001100
1001 | 10011101000
       1001
       ----
       00001
        0000
        ----
        00011
         0000
         ----
         00110
          0000
          ----
          01101
           1001
           ----
           01000
            1001
            ----
            00010
             0000
             ----
             00100
              0000
              ----
              0100  <-- CRC
```

- From here, CRC = 100.
- Now,
  - The code word to be transmitted is obtained by replacing the last 3 zeroes of 10011101**000** with the CRC.
  - Thus, the code word transmitted to the receiver = 10011101**100**.

- **Part-02:**
- According to the question,
  - Third bit from the left gets inverted during transmission.
  - So, the bit stream received by the receiver = 10111101100.
- Now,
  - Receiver receives the bit stream = 10111101100.
  - Receiver performs the binary division with the same generator polynomial as-

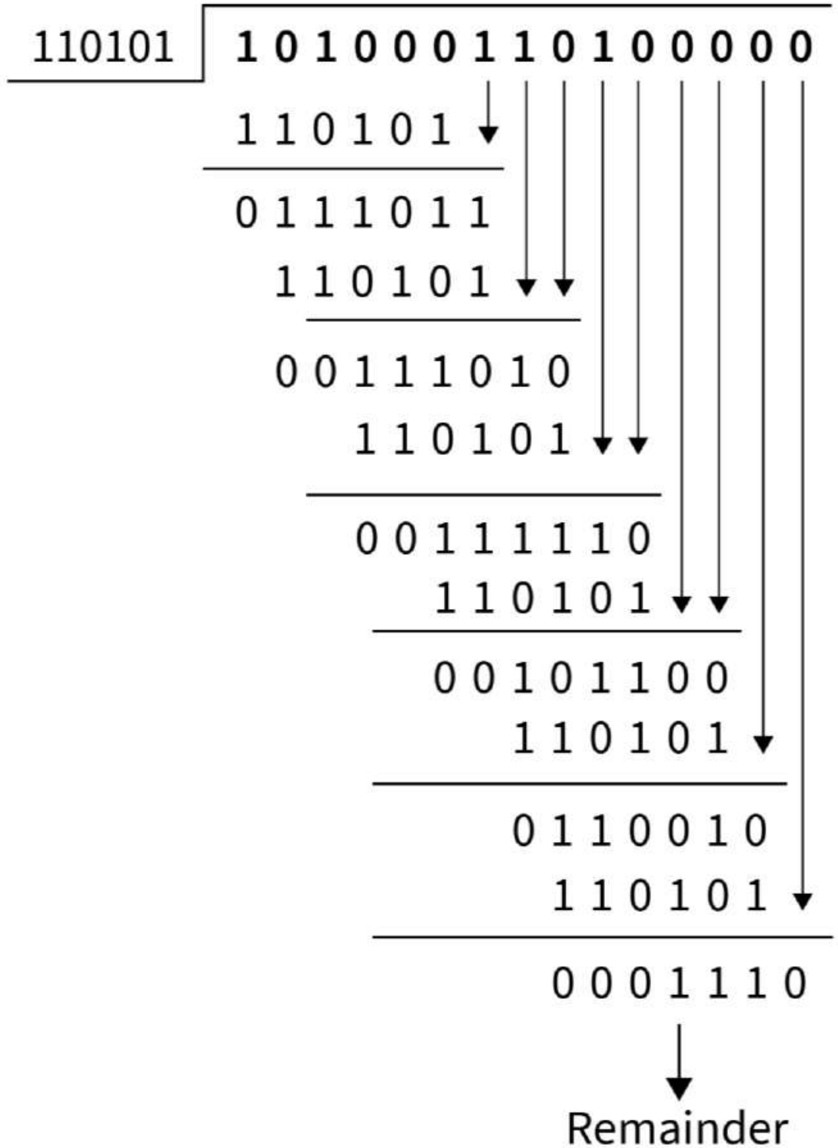
```
          1 0 1 0 1 0 0 0
        ----------------------
1 0 0 1 | 1 0 1 1 1 1 0 1 1 0 0
          1 0 0 1
          -------
          0 0 1 0 1
            0 0 0 0
            -------
            0 1 0 1 1
              1 0 0 1
              -------
              0 0 1 0 0
                0 0 0 0
                -------
                0 1 0 0 1
                  1 0 0 1
                  -------
                  0 0 0 0 1
                    0 0 0 0
                    -------
                    0 0 0 1 0
                      0 0 0 0
                      -------
                      0 0 1 0 0
                        0 0 0 0
                        -------
                        0 1 0 0   <---- Remainder
                        -------
```

7.77

- From here,
- The remainder obtained on division is a non-zero value.
- This indicates to the receiver that an error occurred in the data during the transmission.
- Therefore, receiver rejects the data and asks the sender for retransmission.

Problem 3

- Consider the message sender wants to send is 1010001101, and the generator polynomial is $x^5 + x^4 + x^2 + 1$. Find the message transmitted by the sender. If the receiver receives the message, check if the receiver receives the correct message or not.


110101
1 0 1 0 0 0 1 1 0 1 0 0 0 0 0
1 1 0 1 0 1
0 1 1 1 0 1 1
1 1 0 1 0 1
0 0 1 1 1 0 1 0
1 1 0 1 0 1
0 0 1 1 1 1 1 0
1 1 0 1 0 1
0 0 1 0 1 1 0 0
1 1 0 1 0 1
0 1 1 0 0 1 0
1 1 0 1 0 1
0 0 0 1 1 1 0
Remainder

- As we can see that the remainder is "01110" (last n bits). Now the sender sends the total data after appending the remainder to data = 1010001101 + 01110 = 101000110101110
.