Computer Network(CSC 503)

Shilpa Ingoley

Lecture 13

Different error-detecting codes

1. Parity.
2. Checksums.
3. Cyclic Redundancy Checks (CRCs).

Cyclic Redundancy Check(CRC)

CRC or Cyclic Redundancy Check is a method of **detecting errors** in communication channel.

- 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
- Bit sequences can be written as polynomials with the coefficients 0 and 1.
- A frame with **k** bits is considered as a polynomial of degree k−1.
- The most significant bit is the coefficient of xk−1the next bit is the coefficient of xk−2.

Example:

$$\begin{aligned} M(x) &= 1*x^7 + 0*x^6 + 0*x^5 + 1*x^4 + 1*x^3 + 0*x^2 + 1*x^1 + 0*x^0 \\ &= x^7 + x^4 + x^3 + x^1 \end{aligned}$$

- The bit sequence **10011010** corresponds to this polynomial:
- Sending and receiving messages can be imagined as an exchange of polynomials

Contd…

- The Data Link Layer protocol specifies a generator polynomial G(x).Generator Polynomial is available on both sender and receiver side.
- *G(x) is a polynomial of degree k*
- If e.g. *G(x) = x3 + x2 + x0*
- *= 1101, then k = 3*
- Therefore, the generator polynomial is of degree 3
- The degree of the generator polynomial is equal to the **number of bits minus one**.
- If for a frame, the CRC need to be calculated, **n 0 bits are appended** to the frame

n corresponds to the degree of the generator polynomial

Contd...

Generator polynomial:	100110

- The generator polynomial has 6 digits
    - Therefore, five 0 bits are appended

Frame (payload):	10101
Frame with appended 0 bits:	1010100000

Contd…

Steps:

- **Sender Side (Generation of Encoded Data from Data and Generator Polynomial (or Key)):**
- ⏵The binary data is first augmented by adding n zeros in the end of the data. (n-degree of the generator polynomial)
- ⏵Use ***modulo-2 binary division*** to divide binary data by the key and store remainder of division.
- ⏵Append the remainder at the end of the data to form the encoded data and send the same.

- **Receiver Side (Check if there are errors introduced in transmission)**Perform modulo-2 division again and if remainder is 0, then there are no errors.

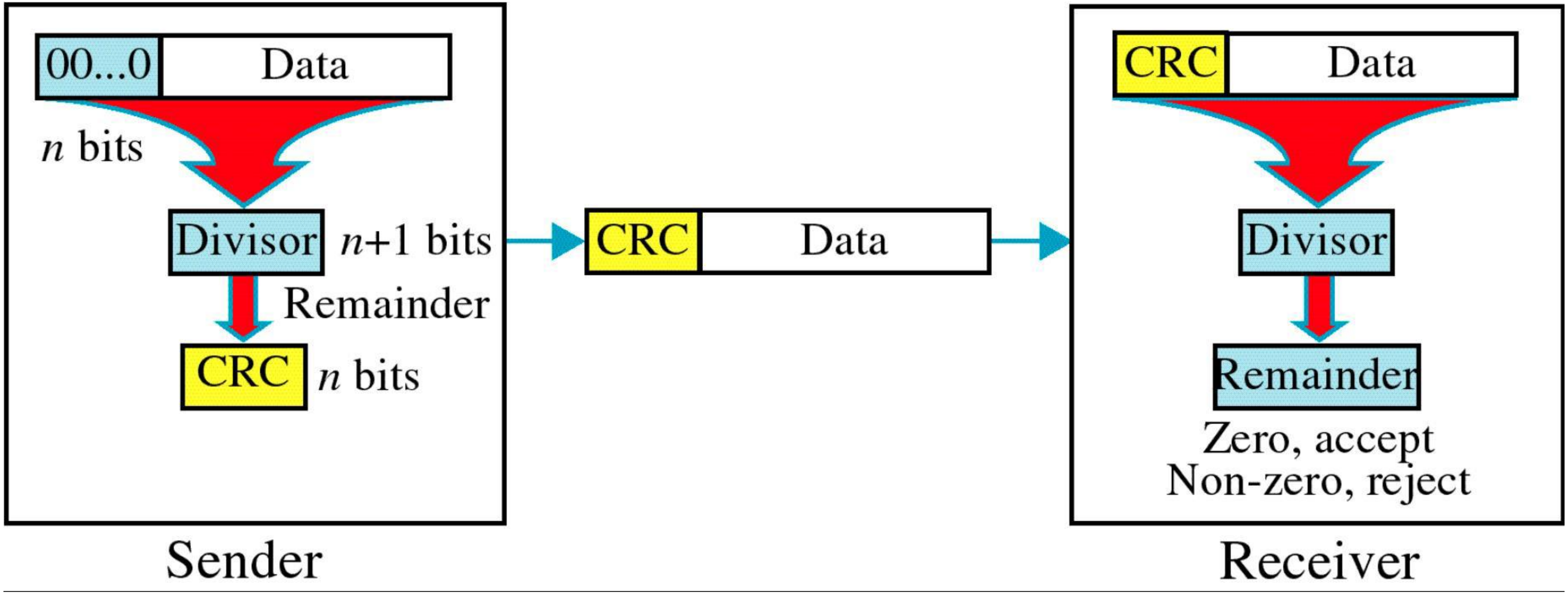
Contd…

- **Modulo 2 Division:** The process of modulo-2 binary division is the same as the familiar division process we use for decimal numbers. Just that **instead of subtraction, we use XOR** here.

$$
\begin{array}{r} 10011011 \\ +\ 11001010 \\ \hline 01010001 \end{array}
\qquad
\begin{array}{r} 00110011 \\ +\ 11001101 \\ \hline 11111110 \end{array}
\qquad
\begin{array}{r} 11110000 \\ -\ 10100110 \\ \hline 01010110 \end{array}
\qquad
\begin{array}{r} 01010101 \\ -\ 10101111 \\ \hline 11111010 \end{array}
$$

- ▶In each step, a copy of the divisor is XORed with the n+1 bits of the dividend (or key).
- ▶The result of the XOR operation (remainder) is n bits, which is used for the next step after 1 extra bit is pulled down to make it n+1 bits long.
- ▶When there are **no bits left to pull down, we have a result**.
- The n-bit remainder which is appended at the sender side.

Contd…

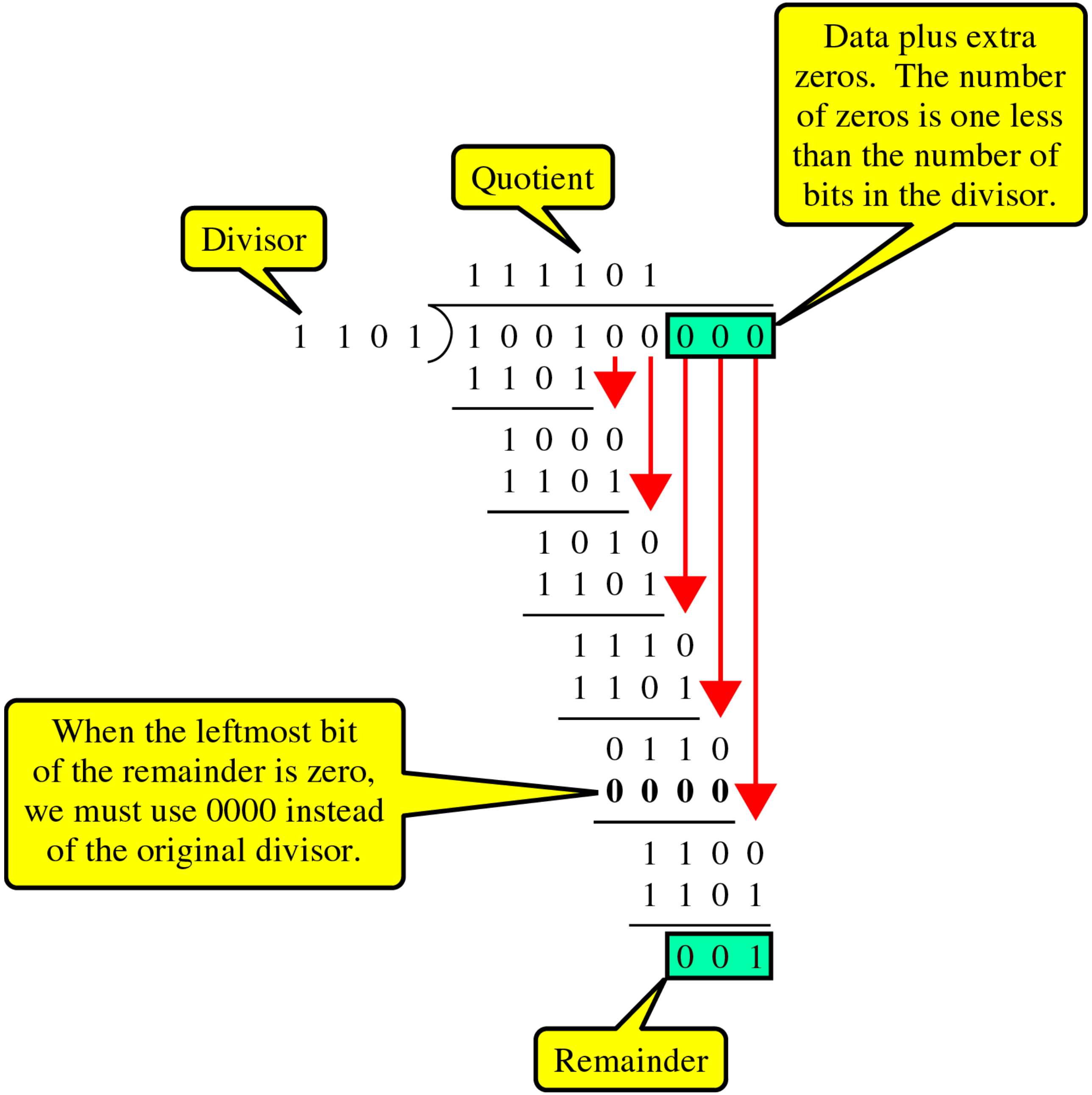

00...0
Data
n bits
Divisor
n+1 bits
Remainder
CRC
n bits
CRC
Data
CRC
Data
Divisor
Remainder
Zero, accept
Non-zero, reject
Sender
Receiver

Contd…

- Data= 100100
- G(x)= 1 1 0 1

Contd...

- CRC generator
- uses modular-2 division.



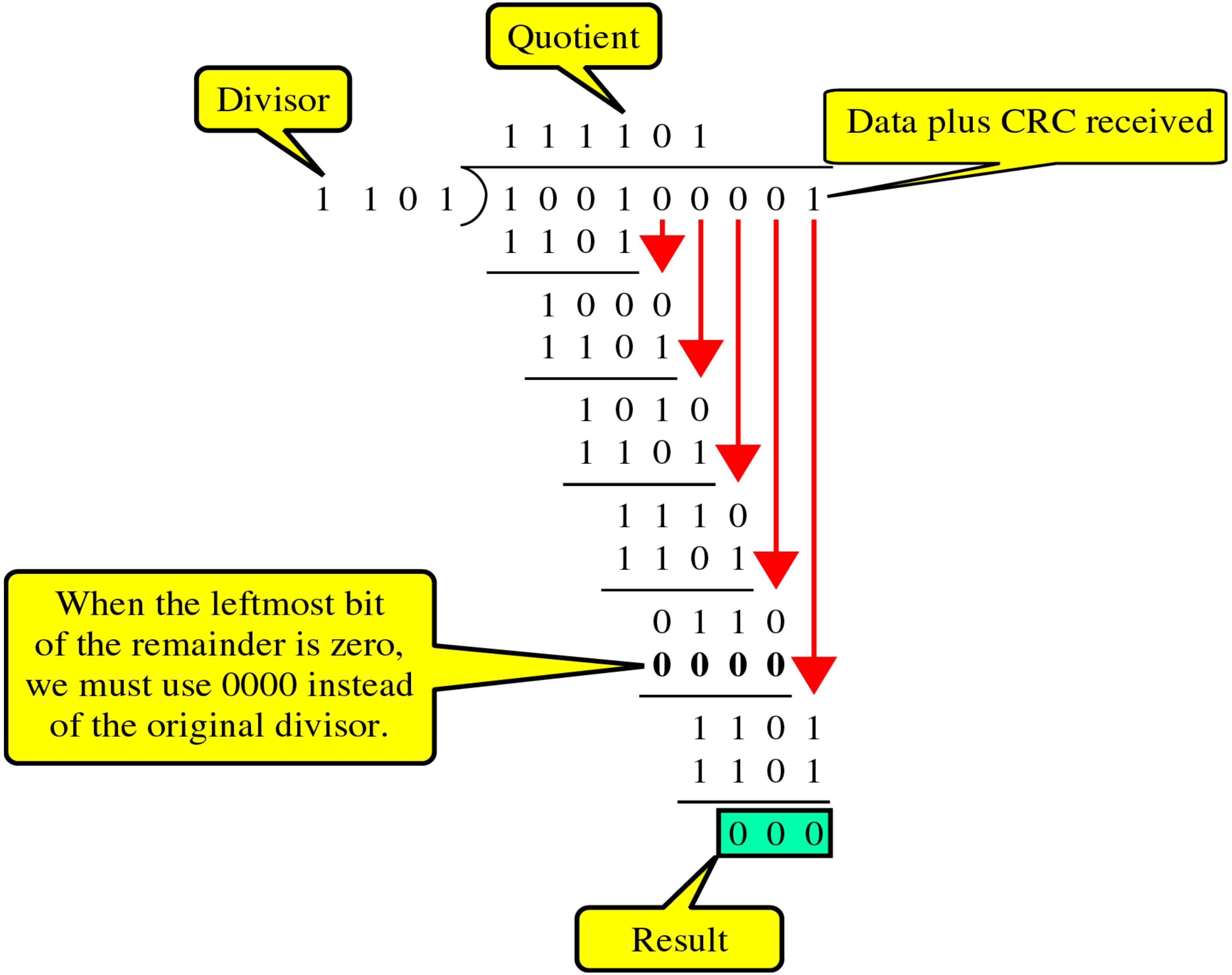
Contd…

At receiver side

Data reciveived=

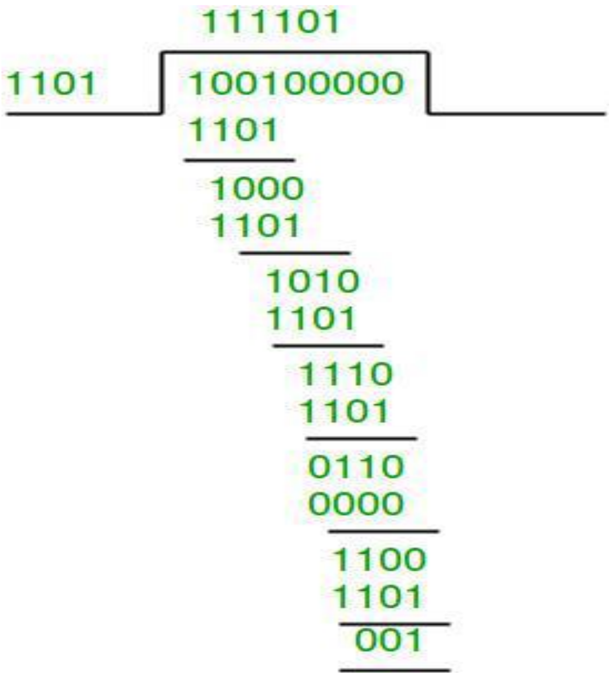
100100001

- G(x)= 1 1 0 1



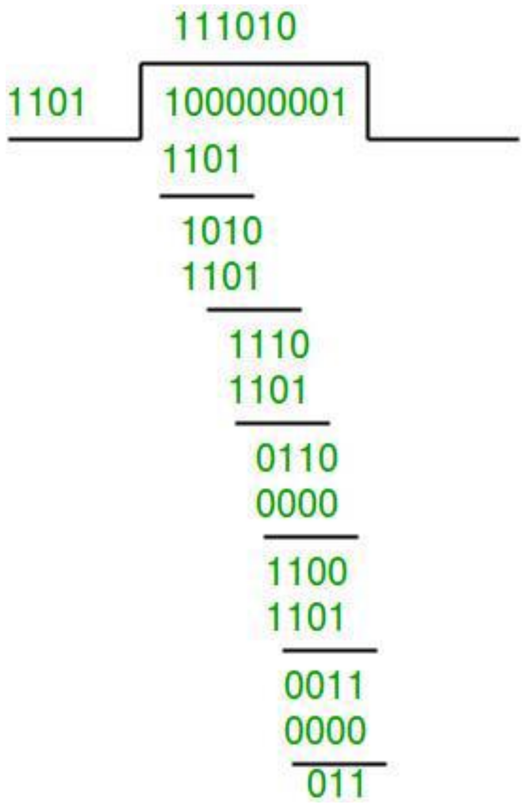
Contd…

- **Therefore, the remainder is all zeros. Hence, the data received has no error.**
- **Example 2:** (Error in transmission)
- ▸Data word to be sent -100100 Key :G(x)= 1101



Contd…

- Therefore, the remainder is 001 and hence the code word sent is 100100001.
- **Receiver Side**
- ▸Let there be error in transmission media
- ▸Code word received at the receiver side -100000001



- Since the remainder is not all zeroes, the error is detected at the receiver side.

Contd…

- **Example :3**

original message

1 0 1 0 0 0 0

Generator polynomial

x^3+1

$1.x^3+0.x^2+0.x^1+1.x^0$

CRC generator

$\boxed{1\,0\,0\,1}$ 4-bit

Sender

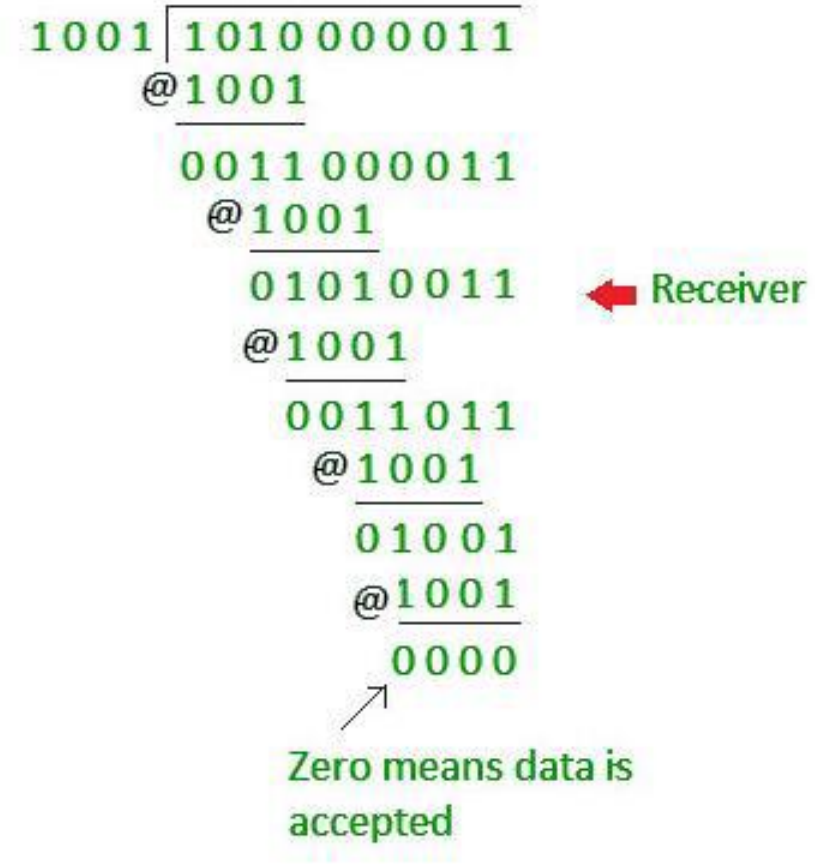
```
1001 | 1010000000
       1001
       ----
       001100
         1001
         ----
          01010
           1001
           ----
           001100
             1001
             ----
              01010
             @ 1001
               ----
               0011
```

Message to be transmitted

```
1010000000
      +011
----------
1010000011
```

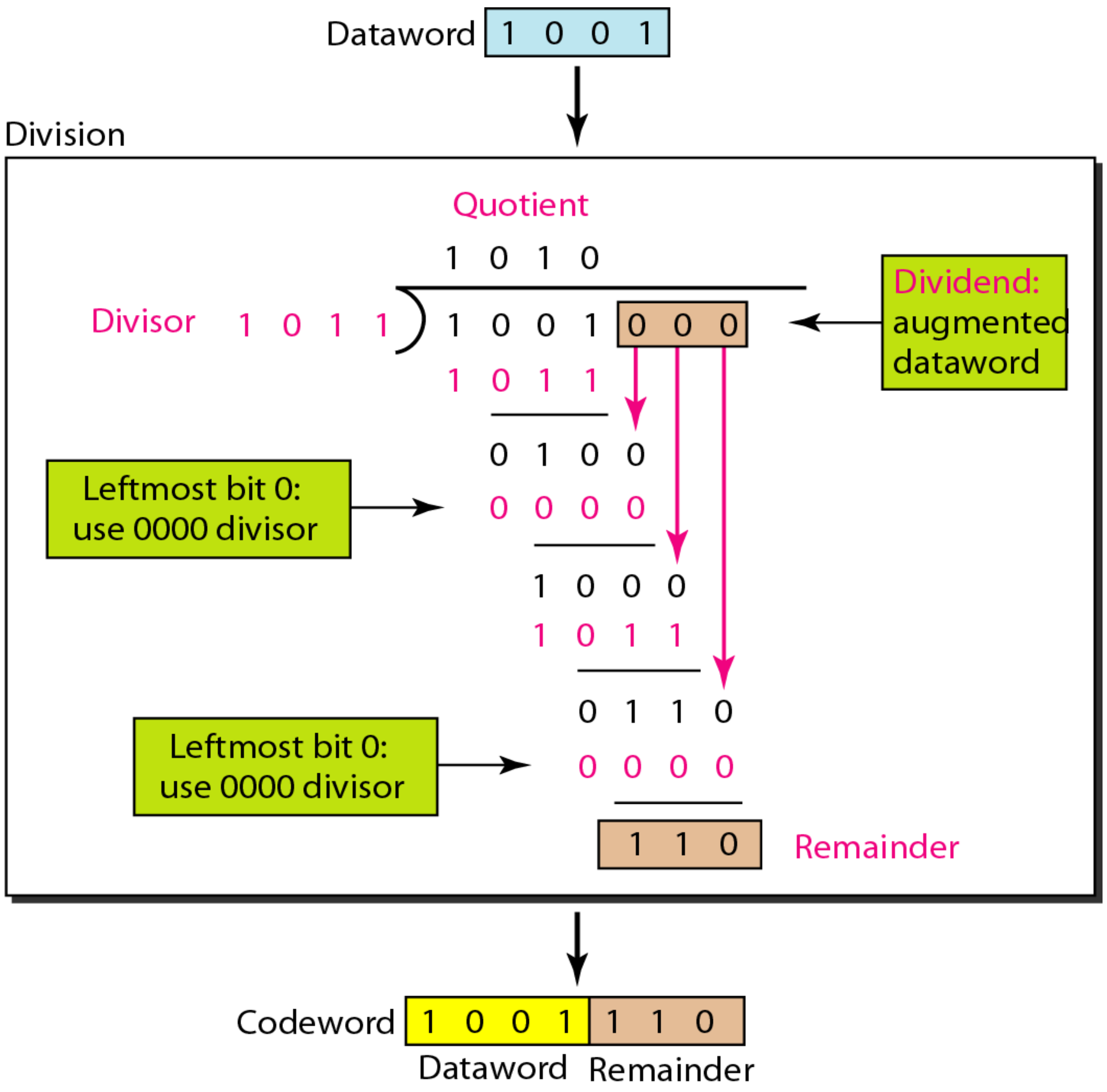
Contd...

At Receiver Side:



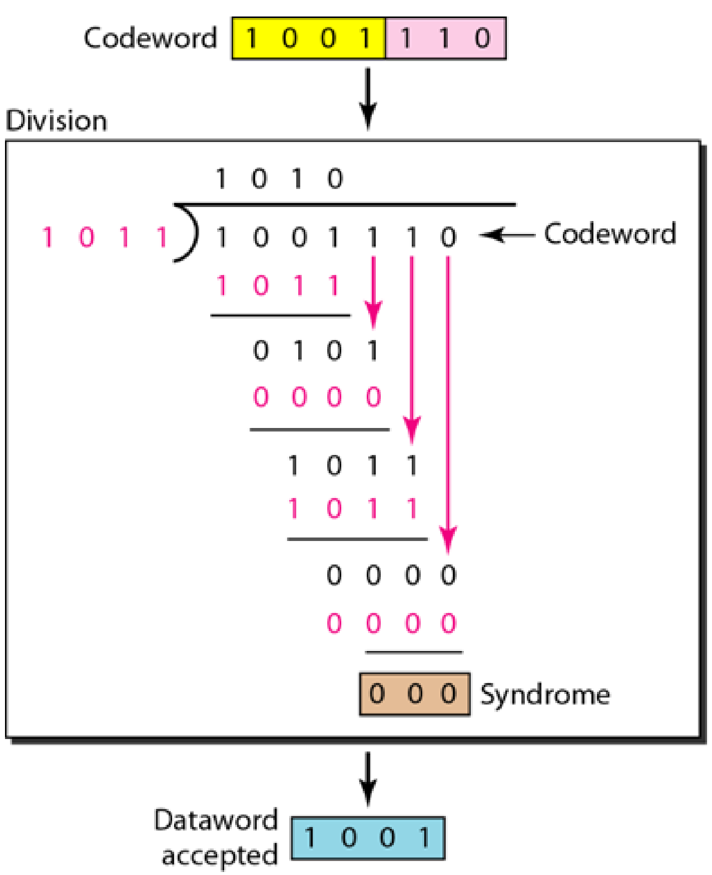
Contd...

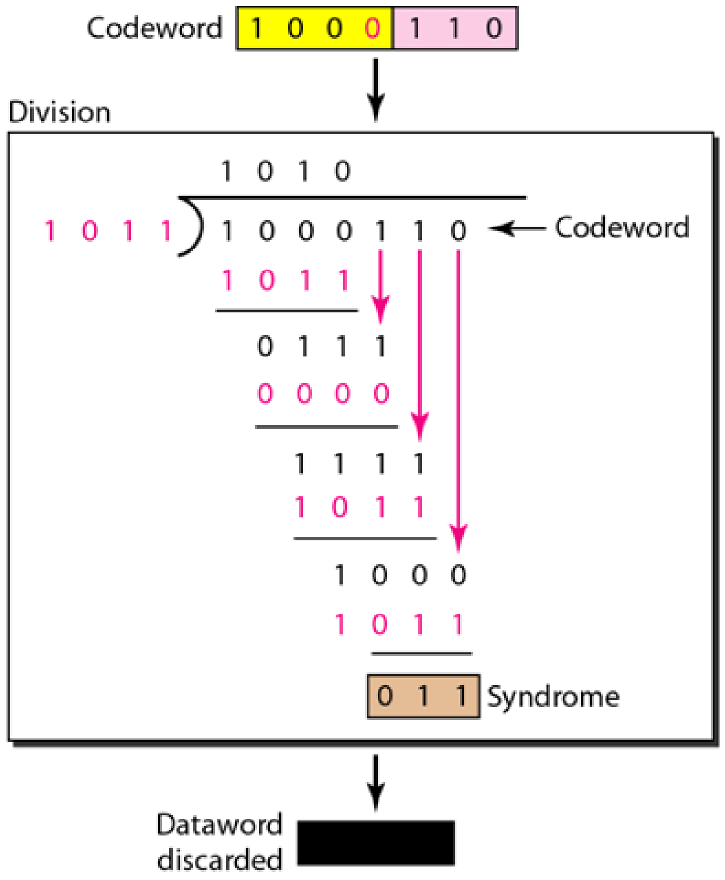
- **Example:4**
- **Data=1001**
- **G(x)=1011**



Contd...

- Data received 1001110 (to host1) and 1000110 (to Host2)
- Find which host received correct data ?





Contd...

Example:

Data bits:

1101011111

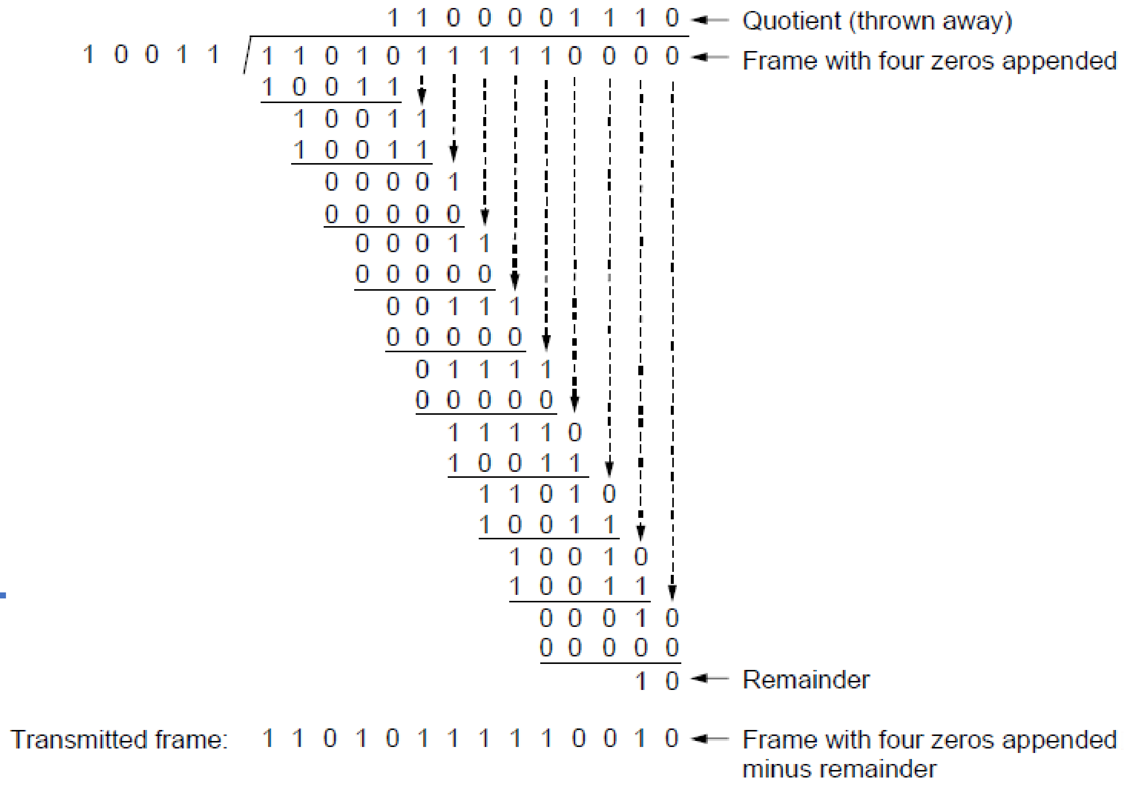
Check bits:

C(x)=x4+x1+1

C = 10011

k = 4

1 0 0 1 1 | 1 1 0 1 0 1 1 1 1 1



Contd..

- CRC can detect **all single-bit errors**
- CRC can detect all double-bit errors
- CRC can detect any odd number of errors
- CRC can detect all burst errors of less than the degree of the polynomial r
- It is said that a CRC (Cyclic Redundancy Checksum) can detect burst errors **of fewer than r + 1 bits**, where r is the degree of the polynomial. Furthermore, a burst of length greater than r + 1 bits is detected with probability $1 – 2^{-r}$.
- A **cyclic redundancy check** (**CRC**) is commonly used in digital network and storage devices to detect accidental changes to raw data.

Standard Polynomials

Name	*Polynomial*	*Application*
CRC-8	$x^8 + x^2 + x + 1$	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	LANs

The most commonly used polynomial lengths are:

- 9 bits (CRC-8)
- 17 bits (CRC-16)
- 33 bits (CRC-32)
- 65 bits (CRC-64)

CRC-8-CCITT $x^8 + x^2 + x + 1$

CRC-16-CCITT $x^{16} + x^{12} + x^5 + 1$

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

$$x^{64} + x^{62} + x^{57} + x^{55} + x^{54} + x^{53} + x^{52} + x^{47} + x^{46} + x^{45} + x^{40} + x^{39} + x^{38} + x^{37} + x^{35} + x^{33} + \\ x^{32} + x^{31} + x^{29} + x^{27} + x^{24} + x^{23} + x^{22} + x^{21} + x^{19} + x^{17} + x^{13} + x^{12} + x^{10} + x^9 + x^7 + x^4 + x + 1$$