BENNETT
UNIVERSITY
TIMES OF INDIA GROUP

Data Link Layer

Error Control-Error Detection Techniques

Error Detection Techniques

1. Parity Checker
2. CRC (Cyclic Redundancy Check)
3. Checksum

Parity Checker

Parity Checker:- In this approach, an extra bit is added to the data before transmission

Types of parity checker:-

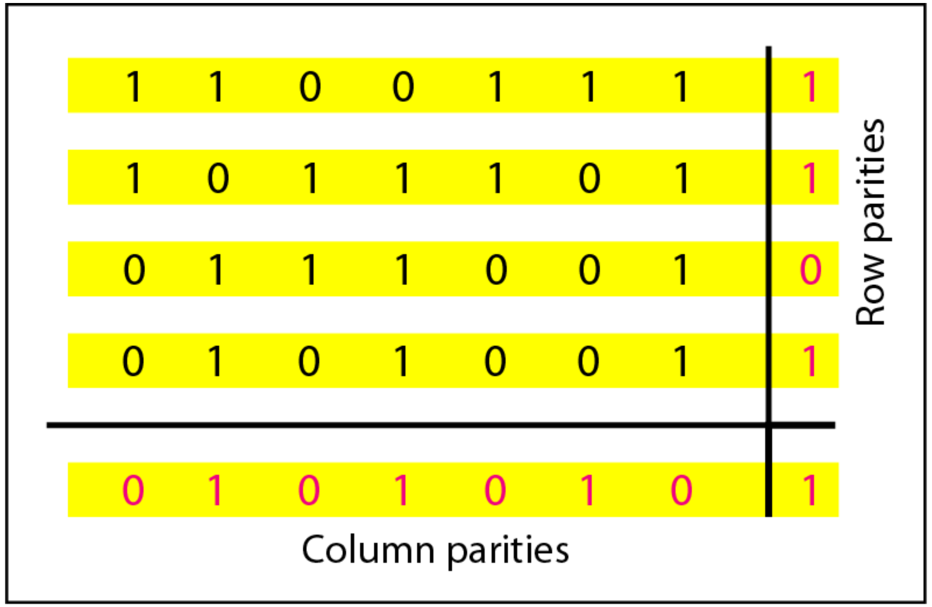
- Even Parity- Number of 1's in the given word should be even
- Odd Parity- Number of 1's in the given word should be odd

Limitations of parity checker

- Not suitable for detection of burst error
- Unable to detect the location of erroneous bit and correct the same.

Two-dimensional parity-checker

- A better approach is the two-dimensional parity check.
- In this method, the dataword is organized in a table (rows and columns).
- It can detect up to three bit error

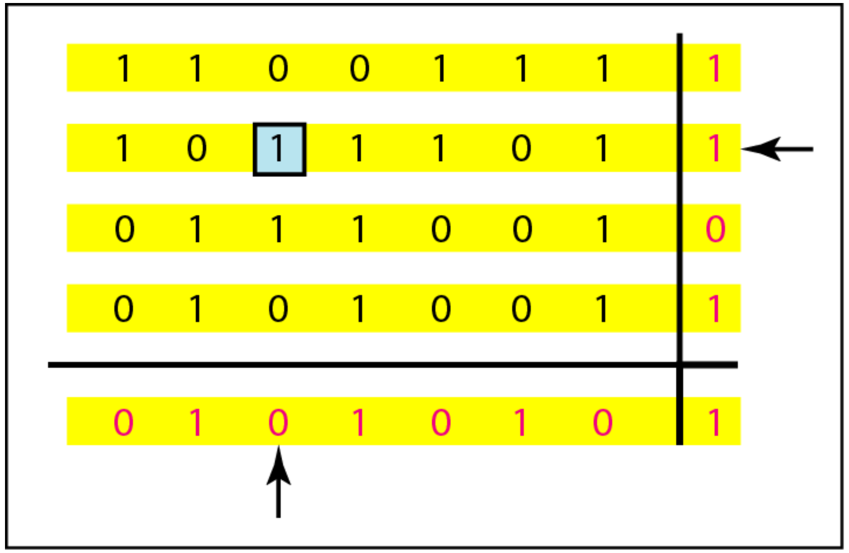


1	1	0	0	1	1	1	1
1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

Row parities
Column parities

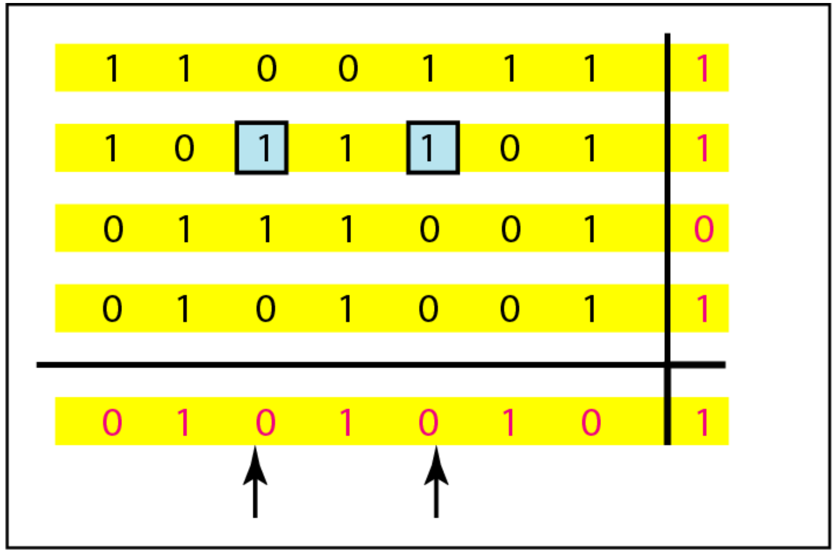
a. Design of row and column parities

Two-dimensional parity-check code



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

b. One error affects two parities

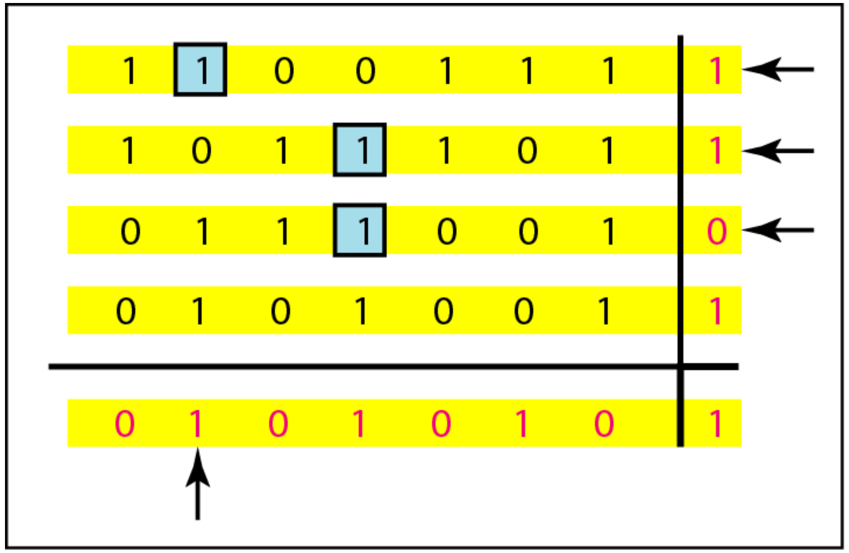


1	1	0	0	1	1	1	1
1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

c. Two errors affect two parities

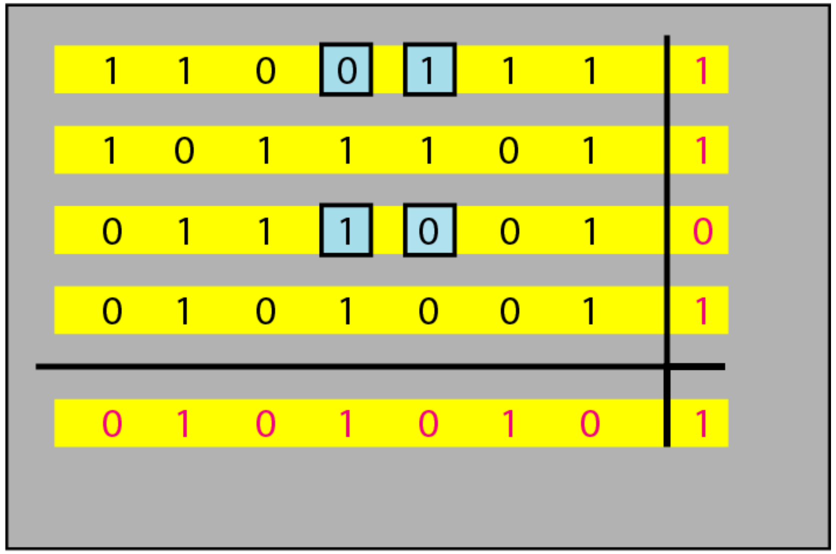
Two-dimensional parity-check code

- Error affecting 4 bit may not be detected



1	$\boxed{1}$	0	0	1	1	1	1
1	0	1	$\boxed{1}$	1	0	1	1
0	1	1	$\boxed{1}$	0	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

d. Three errors affect four parities



1	1	0	$\boxed{0}$	$\boxed{1}$	1	1	1
1	0	1	1	1	0	1	1
0	1	1	$\boxed{1}$	$\boxed{0}$	0	1	0
0	1	0	1	0	0	1	1
0	1	0	1	0	1	0	1

e. Four errors cannot be detected

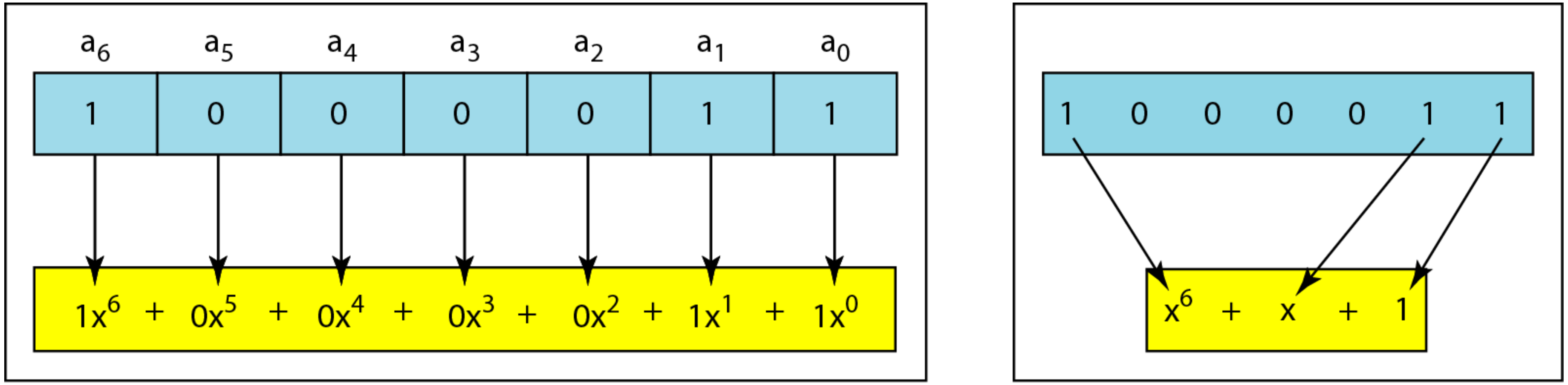
Cyclic Redundancy Check

Cyclic codes are special linear block codes with one extra property. In a cyclic code, if a codeword is cyclically shifted (rotated), the result is another codeword.

- *It is based on the concept of binary division*

Cyclic Redundancy Check

- The divisor in a cyclic code is normally called the generator polynomial or simply the generator.



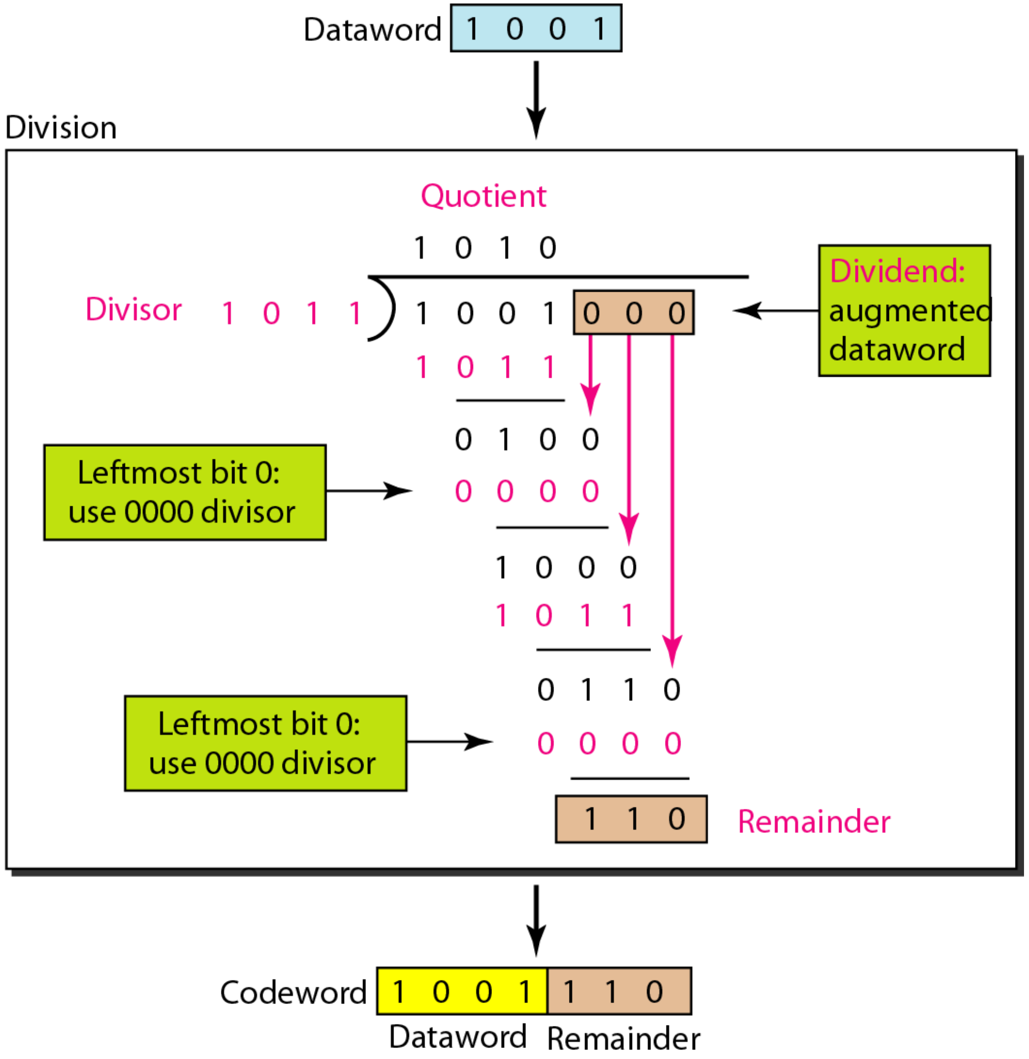
a_6 a_5 a_4 a_3 a_2 a_1 a_0
1 0 0 0 0 1 1
$1x^6 + 0x^5 + 0x^4 + 0x^3 + 0x^2 + 1x^1 + 1x^0$
1 0 0 0 0 1 1
$x^6 + x + 1$

a. Binary pattern and polynomial

b. Short form

Division in CRC encoder

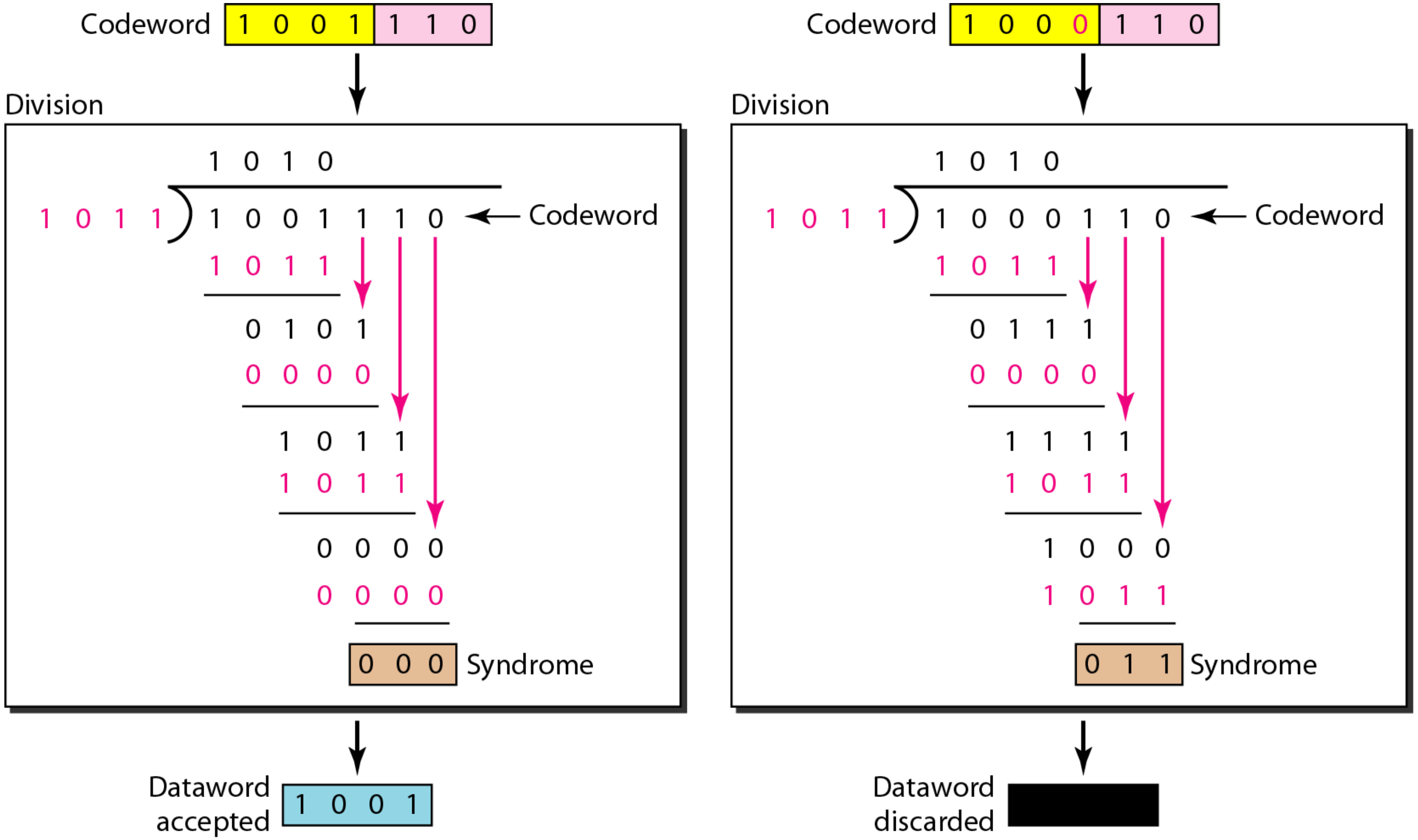
- *Message= 1001*
- *Sender appends number of zeros in message equals to degree of generator polynomial*
- *Divide the appended message with generated polynomial using modulo 2 arithmetic*
- *Remainder becomes CRC*
- *Remove the appended zeros from message and append the calculated CRC*
- *Send the code word to receiver*



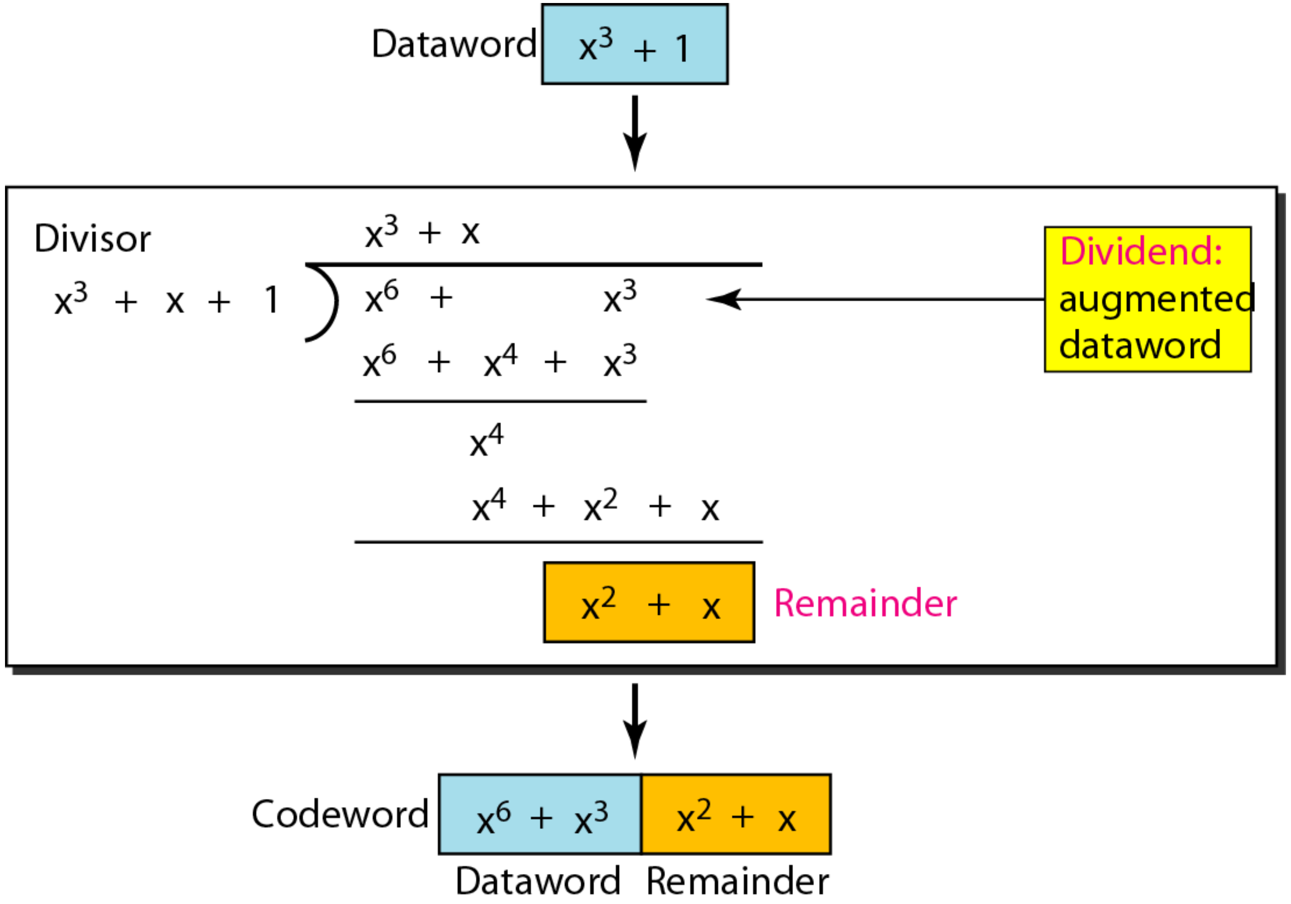
Division in the CRC decoder for two cases

At receiver Side

- *Receiver takes the code word*
- *It divides the codeword with same generator polynomial using modulo 2 arithmetic*
- *If remainder contains all zero bits, message is accepted otherwise it is discarded*



CRC division using polynomials



Cyclic Redundancy Check

- If the generator has more than one term and the coefficient of x^0 is 1, all single errors can be caught.
- If a generator cannot divide $x^t + 1$ (t between 0 and n – 1), then all isolated double errors can be detected.
- A generator that contains a factor of *x* + 1 can detect all odd-numbered errors.

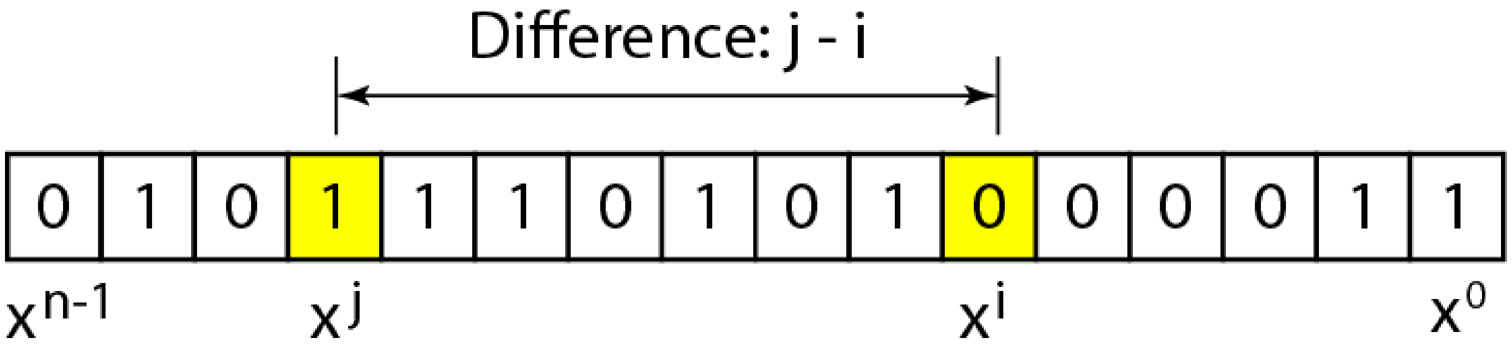
Cyclic Redundancy Check

- All burst errors with $L \leq r$ will be detected, Where r= degree of generator polynomial
- All burst errors with $L = r + 1$ will be detected with probability $1 – (1/2)^{r–1}$.
- All burst errors with $L > r + 1$ will be detected with probability $1 – (1/2)^{r}$.

A good polynomial generator needs to have the following characteristics:

1. It should have at least two terms.
2. The coefficient of the term x^0 should be 1.
3. It should not divide $x^t + 1$, for t between 2 and $n − 1$.
4. It should have the factor $x + 1$.

Representation of two isolated single-bit errors using polynomials



Difference: j - i

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

x^{n-1} x^j x^i x^0

CHECKSUM

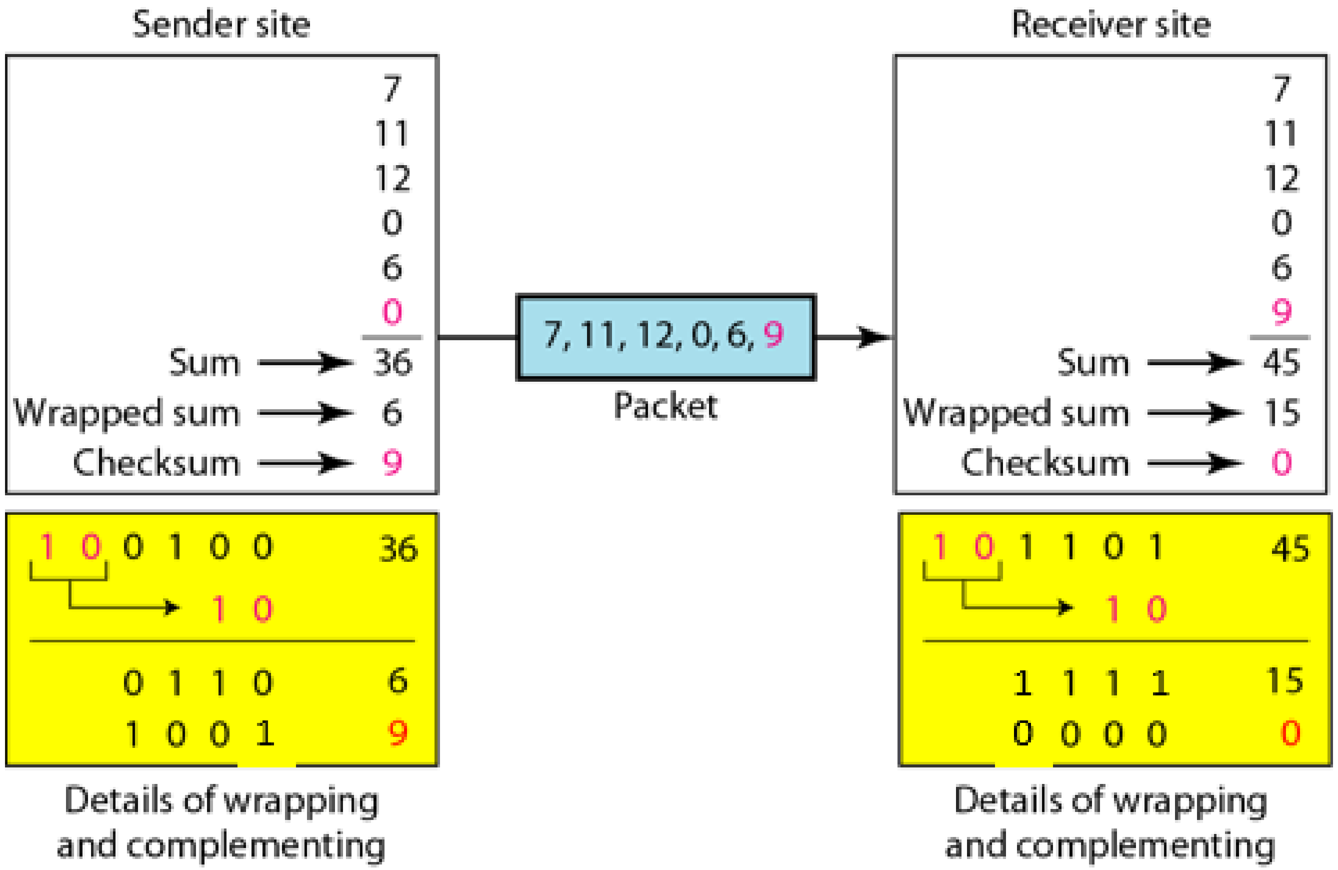
- *In this method, each data word is added to the previous data word and total sum (checksum) is calculated.*
- *Data along with checksum is then transmitted.*
- *This method detects all odd bit error; Even bit error may or may not be detected*

Idea of Checksum

- *Suppose data is a list of five 4-bit numbers (7, 11, 12, 0, 6),*
- *Sender will 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*
- *If the two are not same, the receiver assumes there is an error somewhere and the data are not accepted.*

- *To make the job of the receiver easier if we send the negative (complement) of the sum, called the checksum.*
- *In this case, we send (7, 11, 12, 0, 6, −36).*
- *The receiver can add all the numbers received (including the checksum).*
- *If the result is 0, it assumes no error; otherwise, there is an error.*

Example : Checksum



Internet Checksum

Sender

1. The message is divided into 16-bit words.
2. The value of the checksum word is set to 0.
3. All words including the checksum are added using one’s complement addition.
4. The sum is complemented and becomes the checksum.
5. The checksum is sent with the data.

Receiver

1. The message (including checksum) is divided into 16-bit words.
2. All words are added using one’s complement addition.
3. The sum is complemented and becomes the new checksum.
4. If the value of checksum is 0, the message is accepted; otherwise, it is rejected.