CNT3004 - Computer Network Concepts

Dr. C. Tidwell, Fall 2020

Chapter 10

Error Detection and Correction

10-1 INTRODUCTION

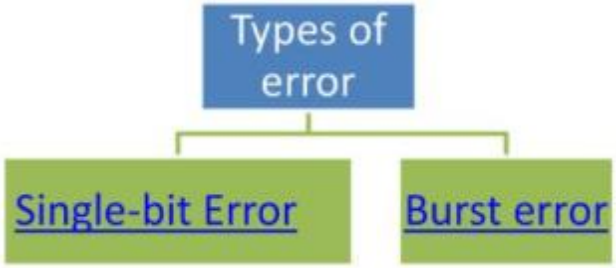
Let us first discuss some issues related, directly or indirectly, to error detection and correction.

Topics discussed in this section:

Types of Errors – single-bit and burst errors
Redundancy – sending extra bits with our data
Detection Versus Correction – correction is more difficult
Coding – used to determine if an error occurs, either through block coding or convolution coding

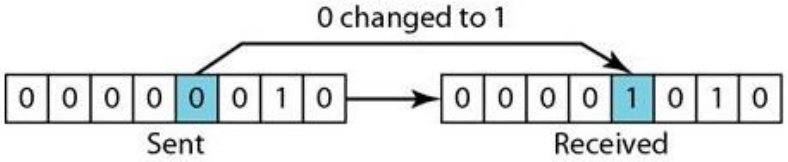
TYPES OF ERRORS

- Bits can be corrupted during transmission because of interference, fading and noise.
- Some applications require that errors be detected and corrected.
- Error detection and correction often occurred at the data link layer (layer 2).

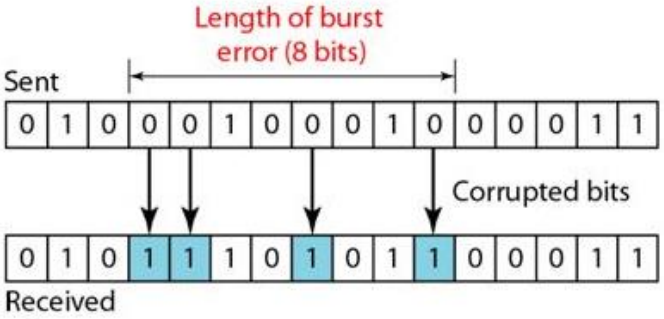


Single-bit and Burst errors

- Single-bit error
  - Only 1 bit in the data unit (packet, frame, cell) has changed.
  - Either 1 to 0, or 1 to 0.



- Burst error
  - 2 or more bits in the data unit have changed.
  - More likely to occur than the single-bit error because the duration of noise is normally longer than the duration of 1 bit.



Error Detection vs Correction

- Error detection – to check if any errors occurred. The answer is simply 'Yes' or 'No'.
- Error correction – to search the exact bits that are in error.
  - If there is a single bit error in 8-bit data unit, so there is a possible of 8 error locations.

Redundancy

- To detect or correct errors, we need to send extra (redundant) bits with data.
- These redundant bits are appended at the sender and removed at the receiver.

Coding

- Coding schemes are a technique or process on how the redundant bits are added to the actual data bis.
- Two broad coding schemes: block and convolution.
- Only block is covered.

10-2 BLOCK CODING

- *In block coding, we divide our message into blocks, each of k bits, called datawords.*
- *We add r redundant bits to each block to make the length n = k + r. The resulting n-bit blocks are called codewords.*

Topics discussed in this section:

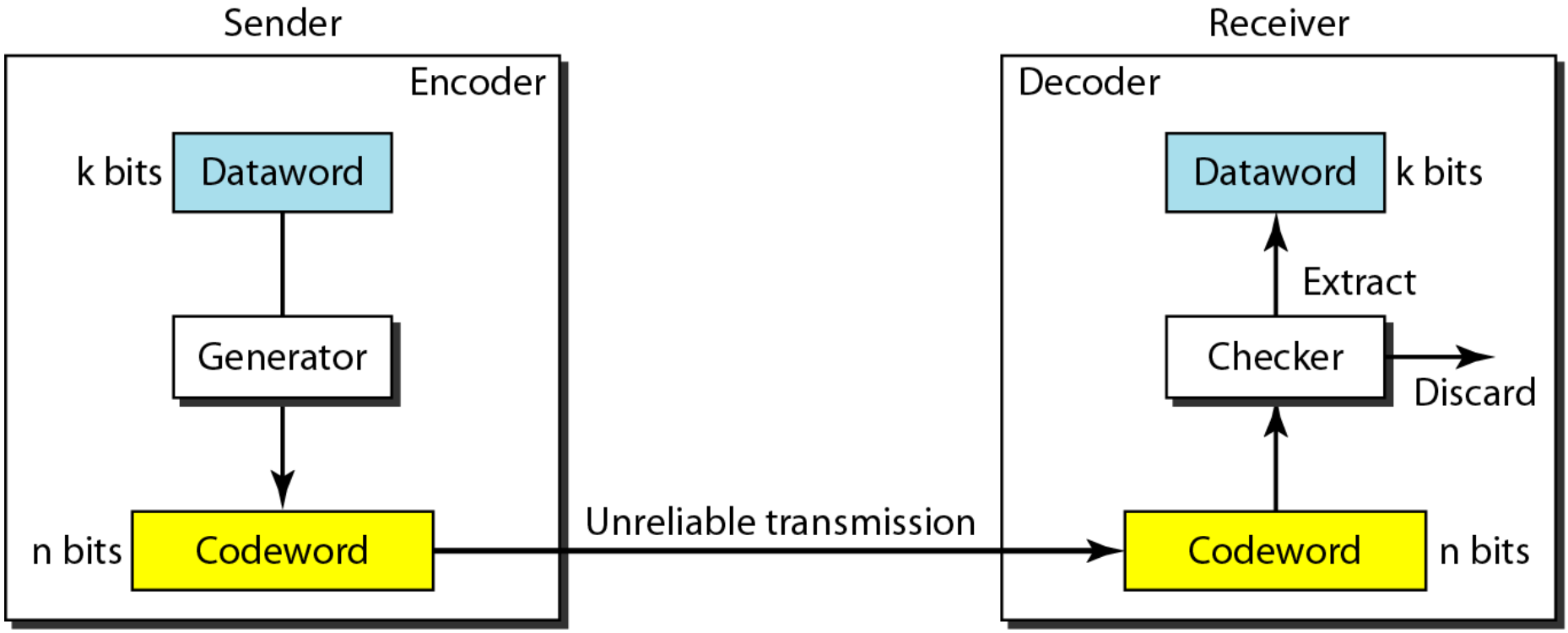
Error Detection
Error Correction
Hamming Distance

Note

Error detection using block coding can detect a change in the original codeword if the following conditions are met:

1. **The receiver has a list of valid codewords.**
2. **The original codeword has changed to an invalid one.**

Figure 10.2 *Process of error detection in block coding*



Example 10.1

- Let us assume that k = 2 and n = 3. Table 10.1 shows the list of datawords and codewords. Later, we will see how to derive a codeword from a dataword.
- Assume the sender encodes the dataword 01 as 011 and sends it to the receiver. Consider the following cases:
  1. The receiver receives 011. It is a valid codeword. The receiver extracts the dataword 01 from it.
  2. The codeword is corrupted during transmission, and 111 is received (the leftmost bit is corrupted). This is not a valid codeword and is discarded.
  3. The codeword is corrupted during transmission, and 000 is received (the right two bits are corrupted). This is a valid codeword. The receiver incorrectly extracts the dataword 00. Two corrupted bits have made the error undetectable.

Table 10.1: A code for error detection in Example 10.1

Datawords	*Codewords*	*Datawords*	*Codewords*
00	000	10	101
01	011	11	110

Example 10.2

- Let us find the Hamming distance between two pairs of words.
  1. The hamming distance d(000,011) is 2 because (000 + 011) is 011 (two 1s).
  2. The Hamming distance d(10101, 11110) is 3 because (10101 + 11110) is 01011 (three 1s).

Example 10.3

- In our first code (Table 10.1), the numbers of 1s in the nonzero codewords are 2, 2, and 2. So the minimum Hamming distance is $d_{min} = 2$.
- This code guarantees detection of only a single error. For example, if the third codeword (101) is sent and one error occurs, the received codeword does not match any valid codeword. If two errors occur, however, the received codeword may match a valid codeword and the errors are not detected.

Table 10.1: A code for error detection in Example 10.1

Datawords	*Codewords*	*Datawords*	*Codewords*
00	000	10	101
01	011	11	110

Note

An error-detecting code can detect only the types of errors for which it is designed; other types of errors may remain undetected.

Note

The Hamming distance between two words is the number of differences between corresponding bits.

Note

The minimum Hamming distance is the smallest Hamming distance between all possible pairs in a set of words.

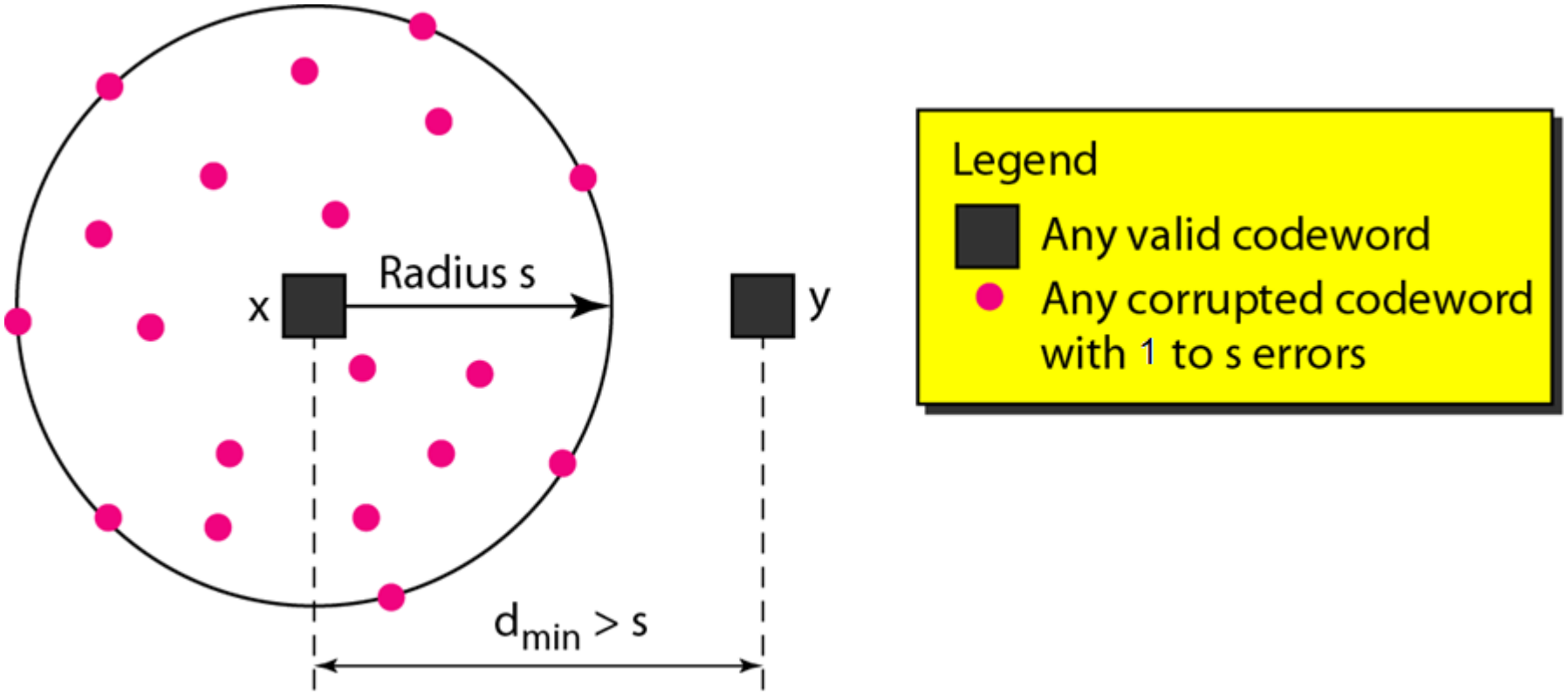
Note

To guarantee the detection of up to s errors in all cases, the minimum Hamming distance in a block code must be $d_{min} >= s + 1$.

Why?

More than s-bit error is possible to detect, but not guaranteed.

Figure 10.3 *Geometric concept for finding d_{min} in error detection*



Example – Hamming Distance

Let us find the Hamming distance between two pairs of words.

1. The Hamming distance d(000, 011) is 2 because

$000 \oplus 011$ is 011 (two 1s)

2. The Hamming distance d(10101, 11110) is 3 because

$10101 \oplus 11110$ is 01011 (three 1s)

Example

Find the minimum Hamming distance of the coding scheme in Table 10.1.

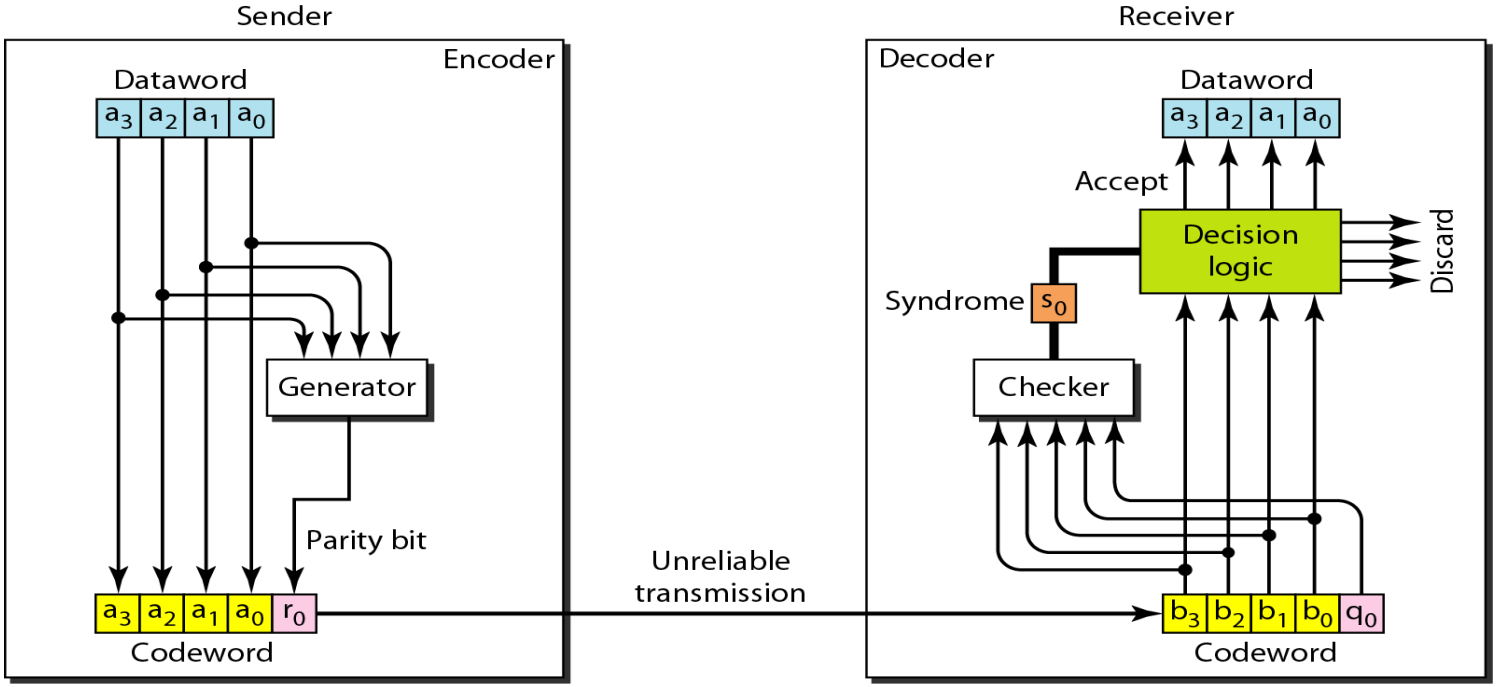
Datawords	Codewords
00	000
01	011
10	101
11	110

Solution

We first find all Hamming distances.

(000 + 011 = 2 (2 1's); 000 + 101 = 2; 000 + 101 = 2; 000 + 110 = 2; 011 + 101 = 2; etc.) We are looking for the number of 1's in the codewords.

Figure 10.4 *Encoder and decoder for simple parity-check code*



$r_0 = a_3 + a_2 + a_1 + a_0$

If the number of 1s is even, the result r_0 is 0; if the number of 1s is odd, the results r_0 is 1

$s_0 = b_3 + b_2 + b_1 + b_0 + q_0$

If the number of 1s is even, the syndrome s_0 is 0; if the number of 1s is odd, the syndrome s_0 is 1

10-3 CYCLIC CODES

***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.*

Topics discussed in this section:

Cyclic Redundancy Check
Polynomials
Cyclic Code Encoder Using Polynomials
Cyclic Code Analysis
Hardware Implementation

Note

Cyclic Redundancy Check (CRC) is used to detect errors in LANs and WANs. It does this by cyclically shifting that results in another codeword.

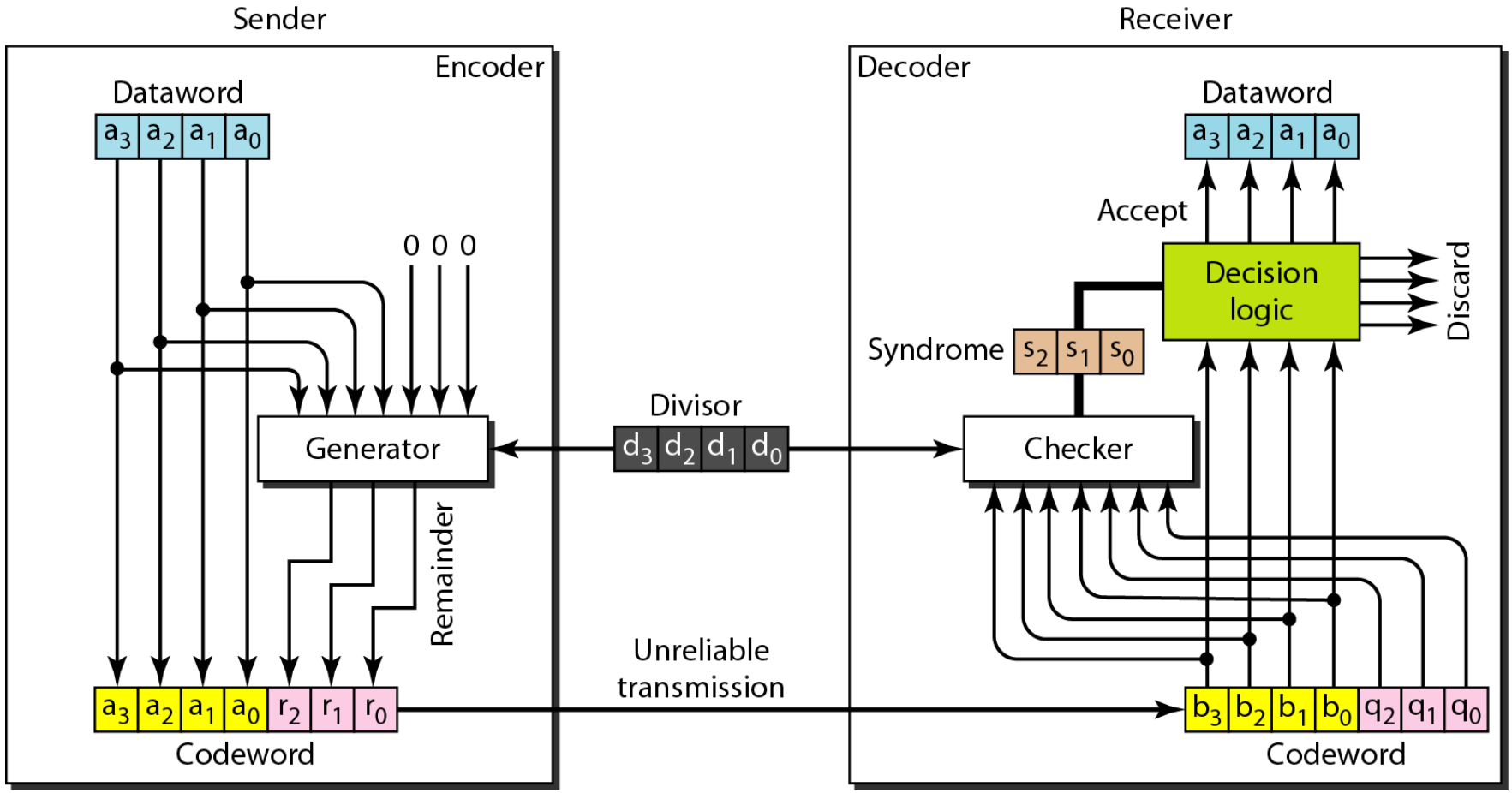
CRC (Cyclic Redundancy Check)

- We can create cyclic codes to correct errors. In this course, we simply discuss a subset of cyclic codes called the cyclic redundancy check (CRC), which is used in networks such as LANs and WANs.

Table 10.3 ***A CRC code with C(7, 4)***

Datawords	*Codewords*	*Datawords*	*Codewords*
0000	00000	1000	10001
0001	00011	1001	10010
0010	00101	1010	10100
0011	00110	1011	10111
0100	01001	1100	11000
0101	01010	1101	11011
0110	01100	1110	11101
0111	01111	1111	11110

Figure 10.5 *CRC encoder and decoder*



Note

A simple parity-check code can detect an odd number of errors.

Figure 10.6 *Division in CRC encoder*

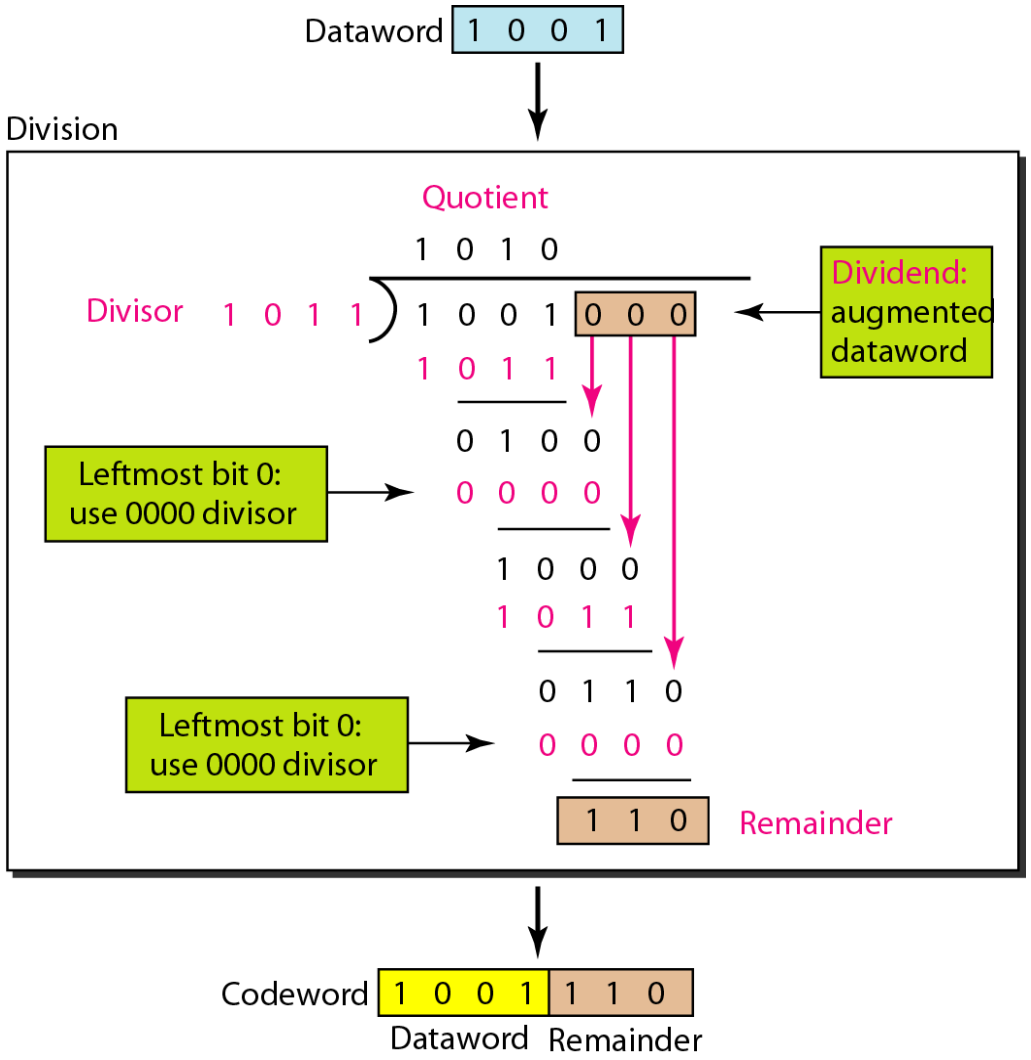
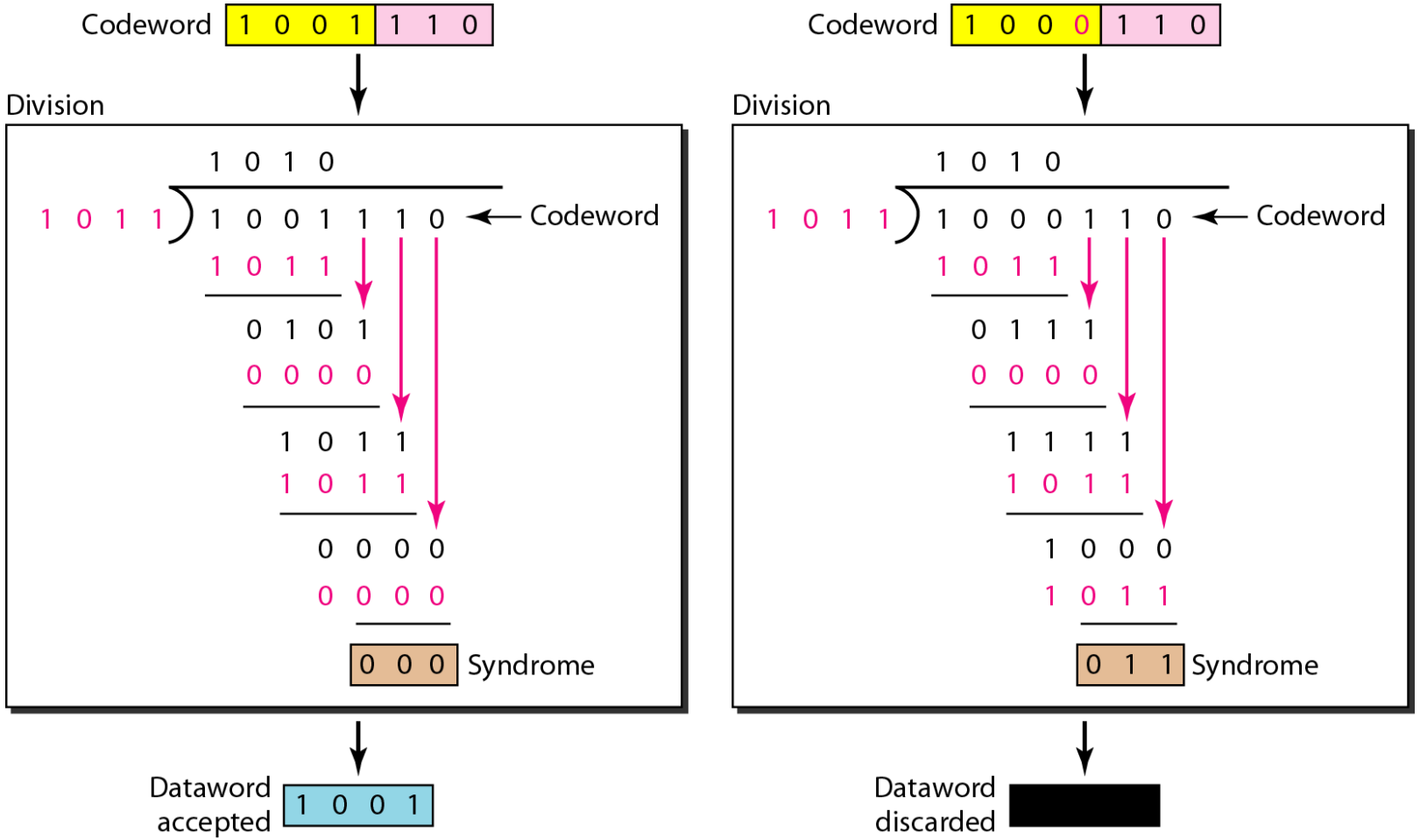


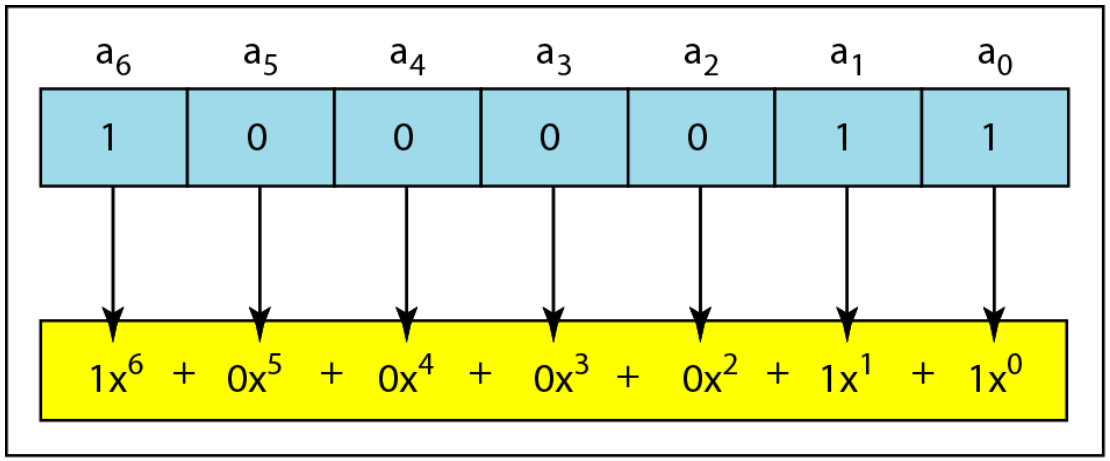
Figure 10.7 *Division in the CRC decoder for two cases*



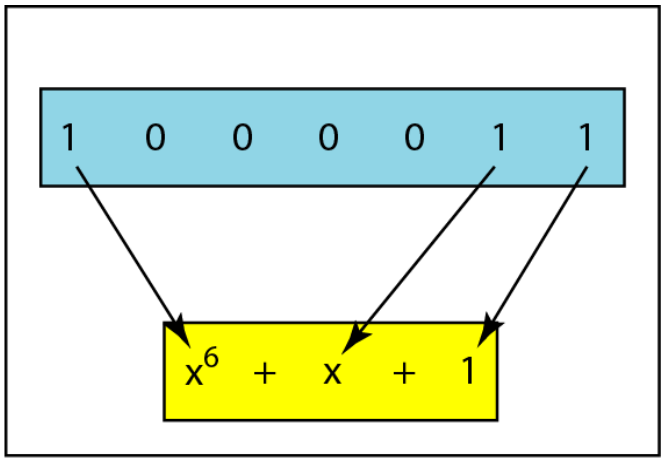
Note

A pattern of 1s and 0s can be represented as a polynomial with coefficients of 0 and 1.
The power of each term shows the position of the bit; the coefficient shows the value of the bit.
Figure 10.8 shows a binary pattern and its polynomial representation.

Figure 10.8 *A polynomial to represent a binary word*



a. Binary pattern and polynomial



b. Short form

Note

The degree of a polynomial is the highest power in the polynomial. For example, $x^6 + x + 1$ is 6.

Table 10.4 *Standard polynomials*

Name	*Polynomial*	*Used in*
CRC-8	$x^8 + x^2 + x + 1$ **100000111**	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$ **11000110101**	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$ **1000100000100001**	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ **10000010011000001000111011011011**	LANs

Advantages of Cyclic Codes

- Has a very good performance in detecting single-bit errors, double errors, an odd number of errors, and burst errors.
- Can easily be implemented in hardware and software.
- Fast when implemented in hardware. This has made cyclic codes a good candidate for many networks.

Other Cyclic Codes

- The cyclic codes we have discussed in this section are very simple. The check bits and syndromes can be calculated by simple algebra.
- There are, however, more powerful polynomials that are based on abstract algebra involving Galois fields. These are beyond the scope of this course. One of the most interesting of these codes is the Reed-Solomon code used today for both detection and correction.

10-4 CHECKSUM

The last error detection method we discuss here is called the checksum. The checksum is used in the Internet by several protocols although not at the data link layer. However, we briefly discuss it here to complete our discussion on error checking

Topics discussed in this section:

Idea/Concept

Other Approaches

Example 10.11

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.

Example 10.12

We can 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.

Figure 10.16 *Example 10.13*

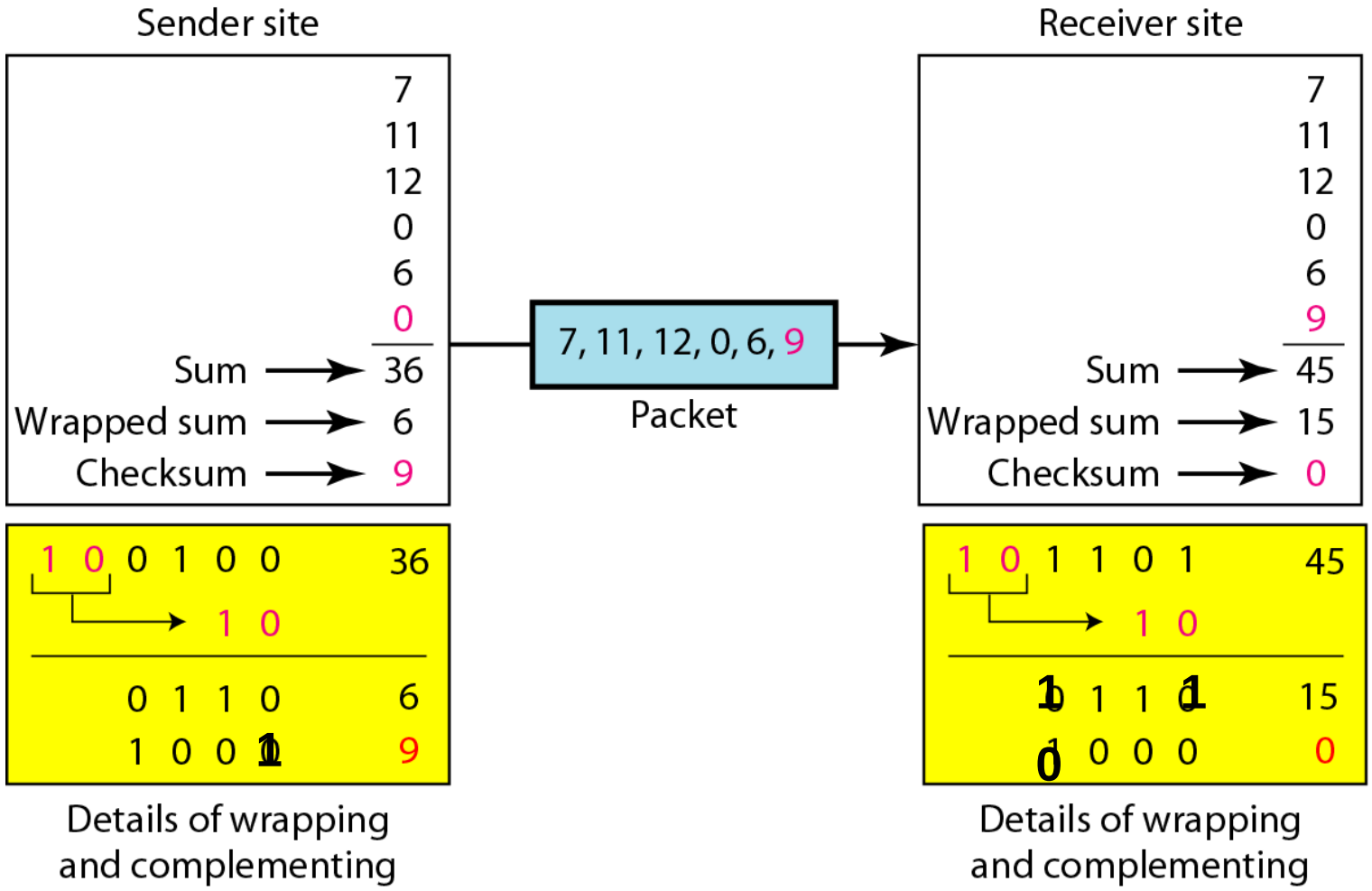
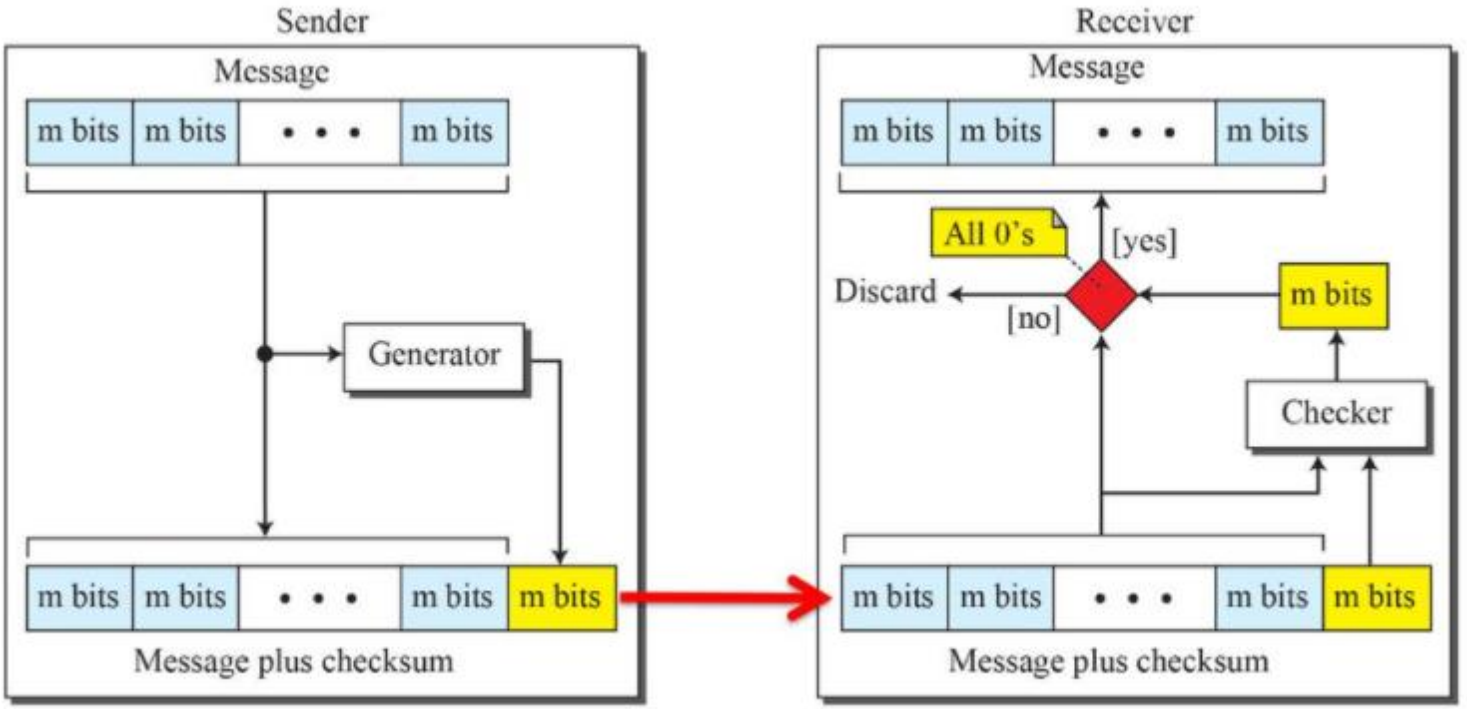


Figure 10.15: Checksum



Example 10.11

Suppose the message 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 message not accepted.

Example 10.12

In the previous example, the decimal number 36 in binary is $(100100)_2$. To change it to a 4-bit number we add the extra leftmost bit to the right four bits as shown below.

$$\begin{array}{r} 10 \\ +\ 0100 \\ \hline 0110 \end{array} \longrightarrow 6_{10}$$

Instead of sending 36 as the sum, we can send 6 as the sum (7, 11, 12, 0, 6, 6). The receiver can add the first five numbers in one's complement arithmetic. If the result is 6, the numbers are accepted; otherwise, they are rejected.

Internet Checksum

Note

Sender site:

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.

Internet Checksum

Note

Receiver site:

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.**

10-5 FORWARD ERROR CORRECTION

Forward Error Correction is a method of obtaining error control in data transmission in which the source (transmitter) sends redundant data, and the destination (receiver) recognizes only the portion of the data that contains no apparent errors.

Topics discussed in this section:

Using Hamming Distance
Using XOR
Chunk Interleaving

Forward Error Correction (FEC)

Note

Using Hamming Distance

1. **Minimum hamming distance should be $d_{min} = s + 1$.**
2. **To detect t errors we need to have $d_{min} = 2t + 1$. Meaning, to correct 10 bits in a packet we need 2(10) + 1 or 21 bits.**

Forward Error Correction (FEC)

Note

Using XOR

1. **The other option is to use the exclusive OR operation.**
2. **The packet is divided into *N* chunks.**
3. **Send N + 1 chunks, if any chunk is lost or corrupted it can b created at the receiver.**
4. **If N = 4 then we need to send N + 1 bit or 25% extra bits.**

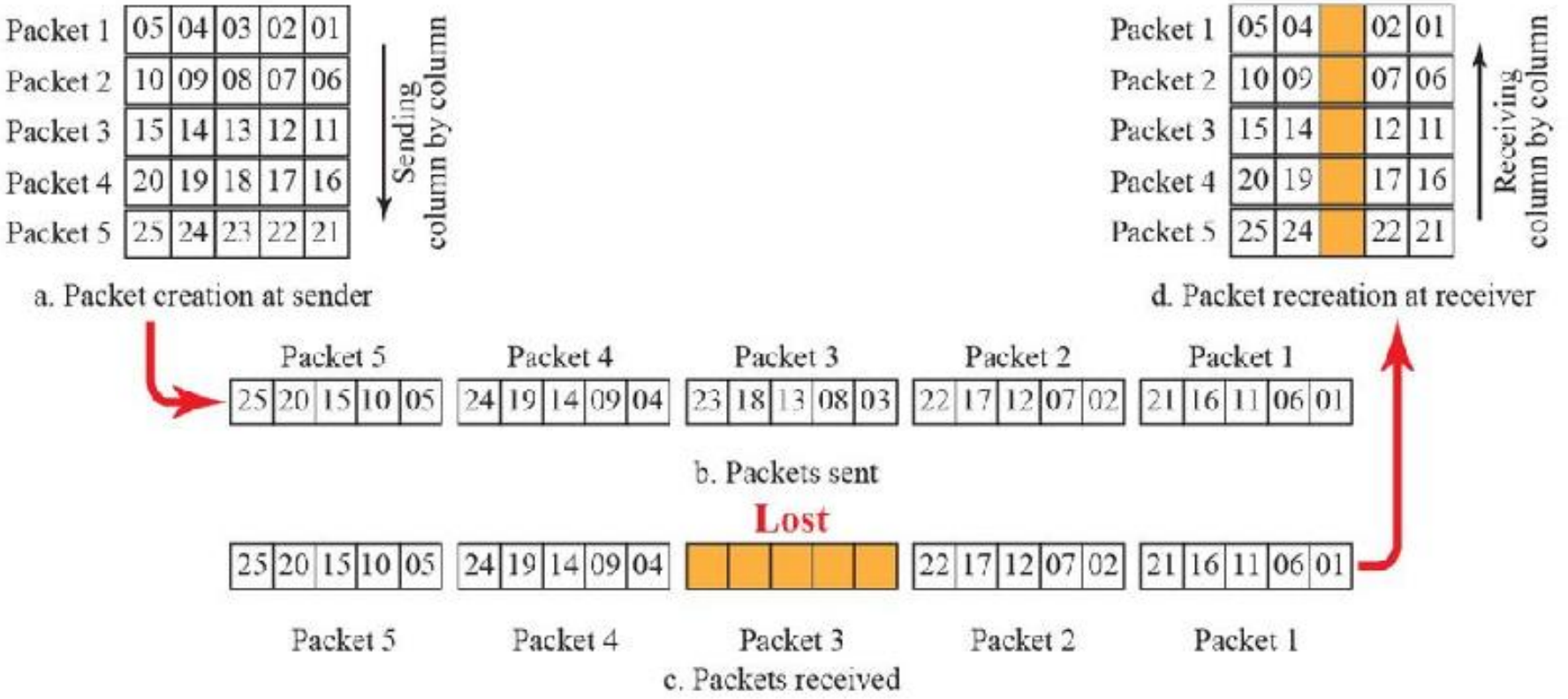
Forward Error Correction (FEC)

Note

Chunk Interleaving

1. **Used to achieve FEC in multimedia.**
2. **This permits small chunks to be missing at the receiver.**

Figure 10.21: Interleaving



CONCLUSION

- 1. Error detection and correction is an important part of data transmission.
- 2. There are single-bit and burst errors.
- 3. Several methods are used:
  - Block coding
  - Cyclic codes: CRC and Polynomials
- 4. Checksum can be applied to a message of any length.
- 5. Forward Error Correction (FEC) using Hamming distance , and XOR.