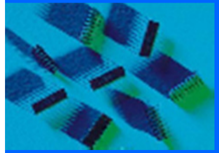


Lecture 3
Data link - *Error Detection and Correction*

Textbook: Ch.9 and Ch.10

Main Topics

- **Ch 9 Data Link Layer**
  - **9.1 Nodes and Links**
- **Ch 10 Error Detection and Correction**
  - **10.1 Error Detection and Correction**
    - **Types of Errors**
  - **10.2 Linear Block Codes**
    - **Parity Check**
  - **10.3 Cyclic Codes**
    - **Cyclic Redundancy Check**

Data-link layer

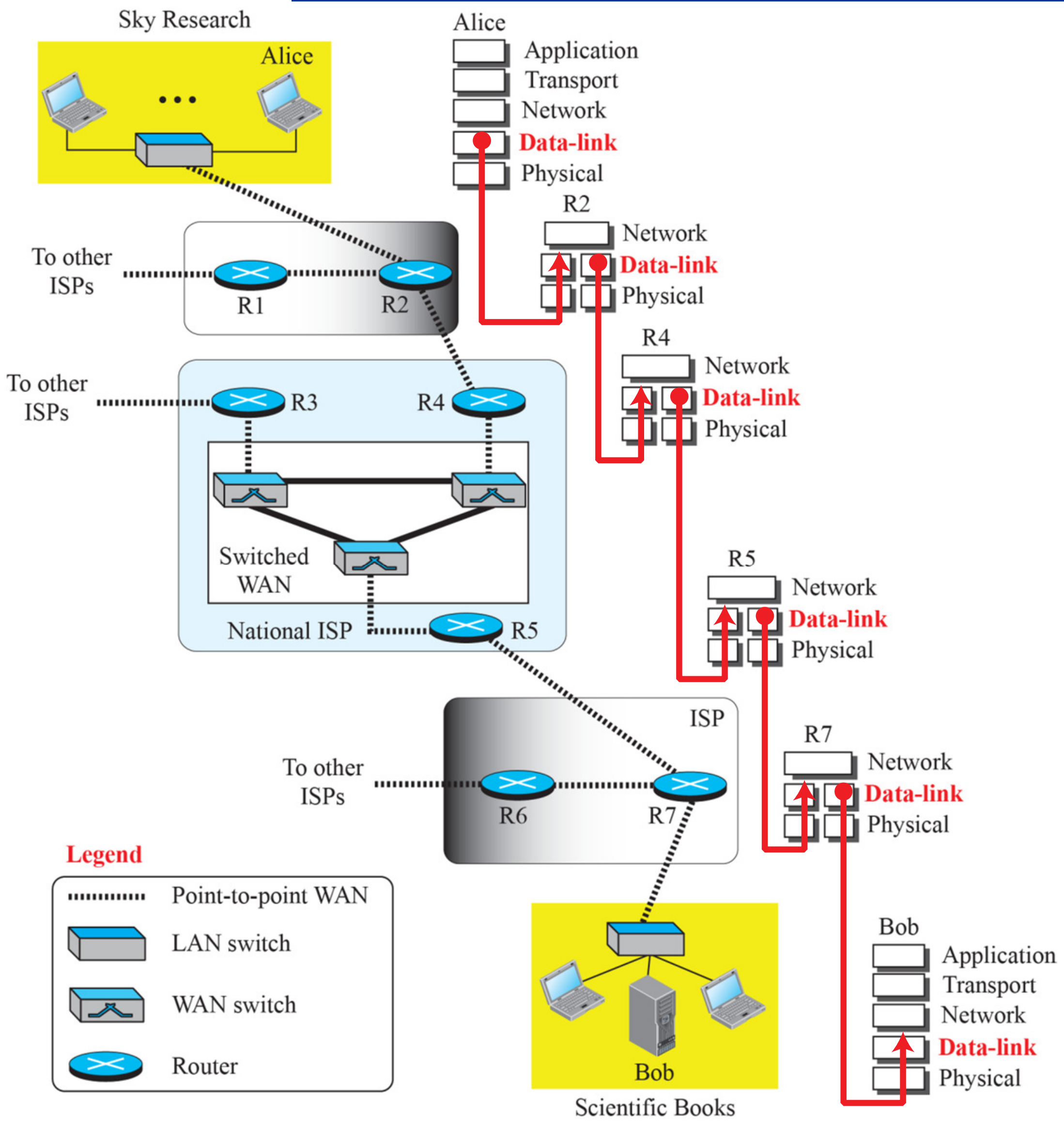
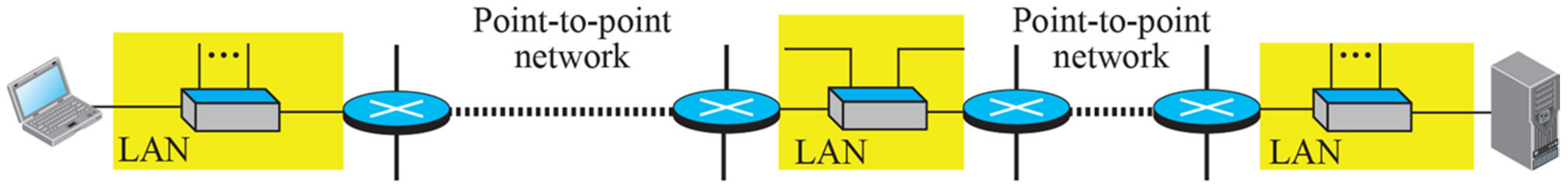


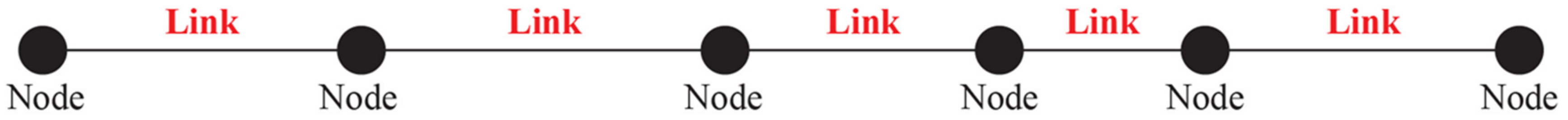
Figure 9.1: Communication at the data-link layer

9.1 Nodes and Links

- Communication at the data-link layer is ***node-to-node.***
- It is customary to refer to the two *end hosts* and the *routers* as ***nodes*** and the *networks* in between as ***links***.



a. A small part of the Internet



b. Nodes and links

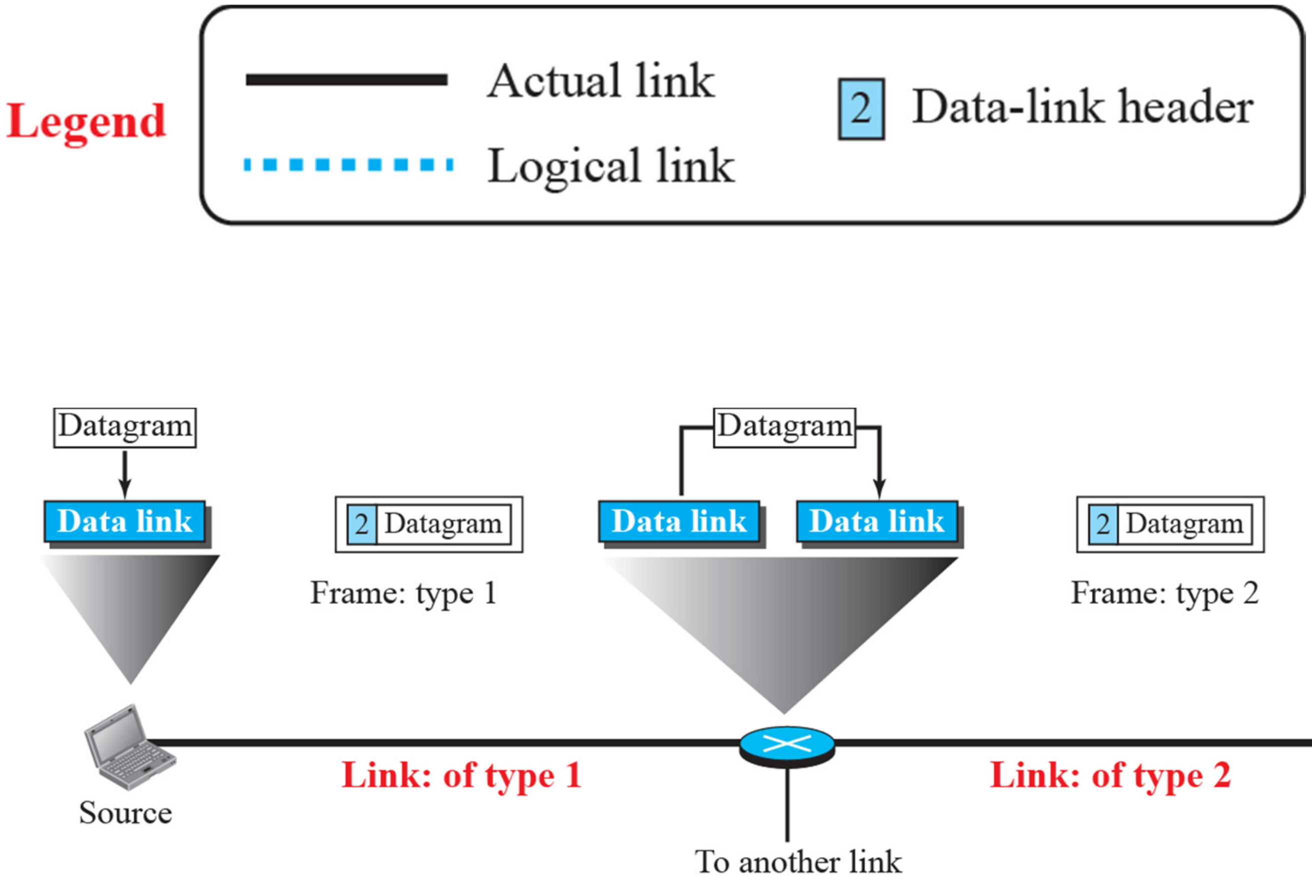
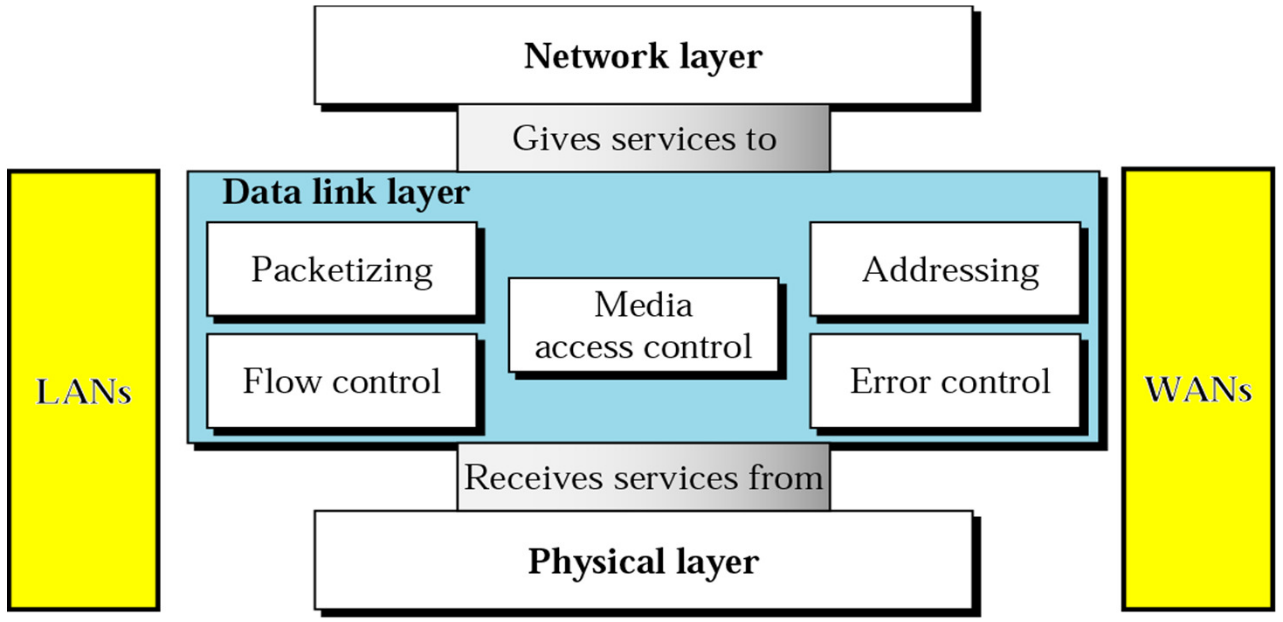


Figure 9.3: A communication with only three nodes

Service of the data-link layer

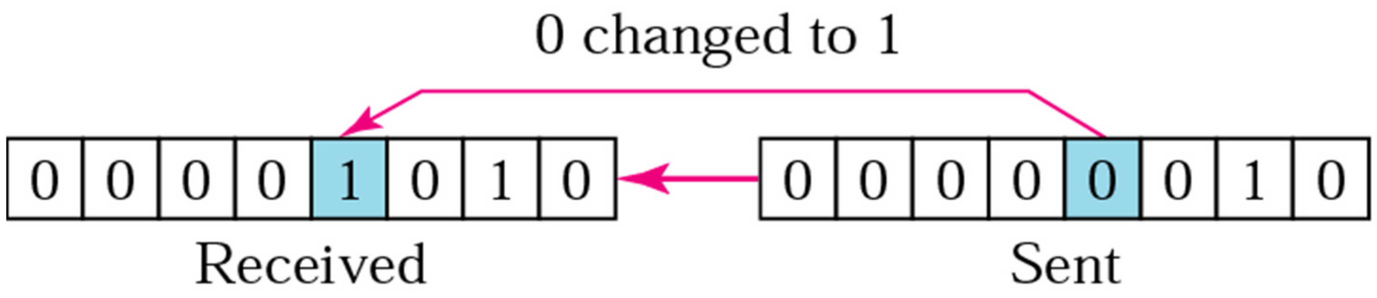
- The data-link layer is located between the physical and the network layers.
- The data-link layer provides services to the network layer; it receives services from the physical layer



Ch 10 Error Detection and Correction

10.1 Types of Errors

1. Single-bit error

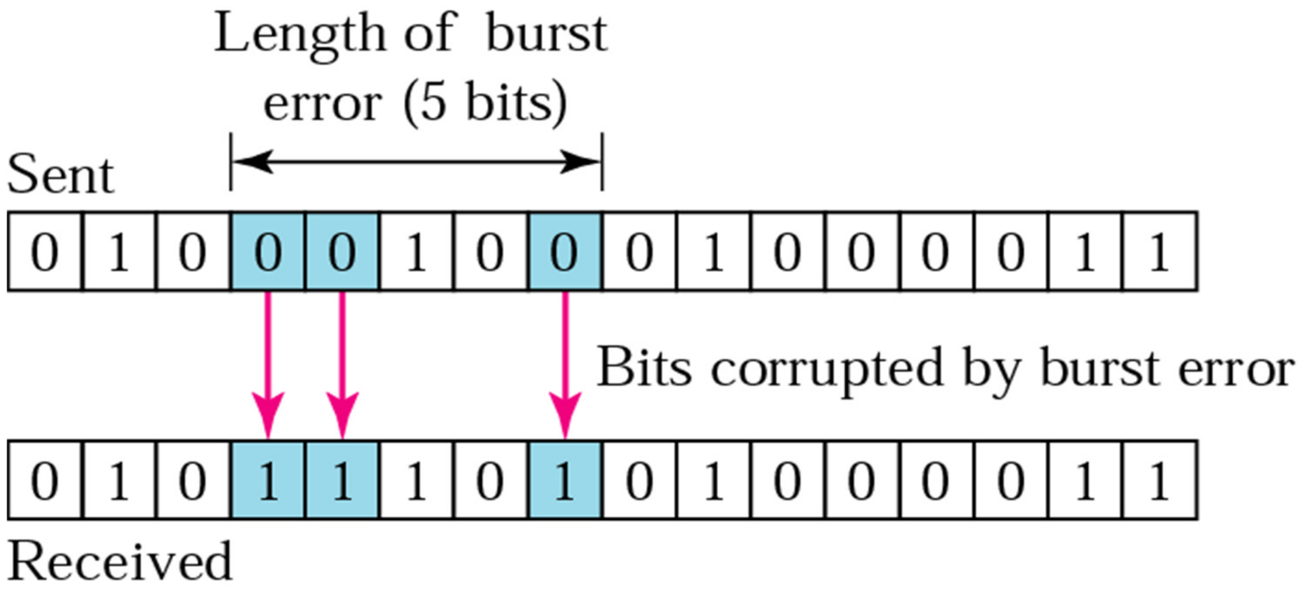


In a single-bit error, only one bit in the data unit has changed.

Types of Errors

2. Burst error

A burst error means that 2 or more bits in the data unit have changed.

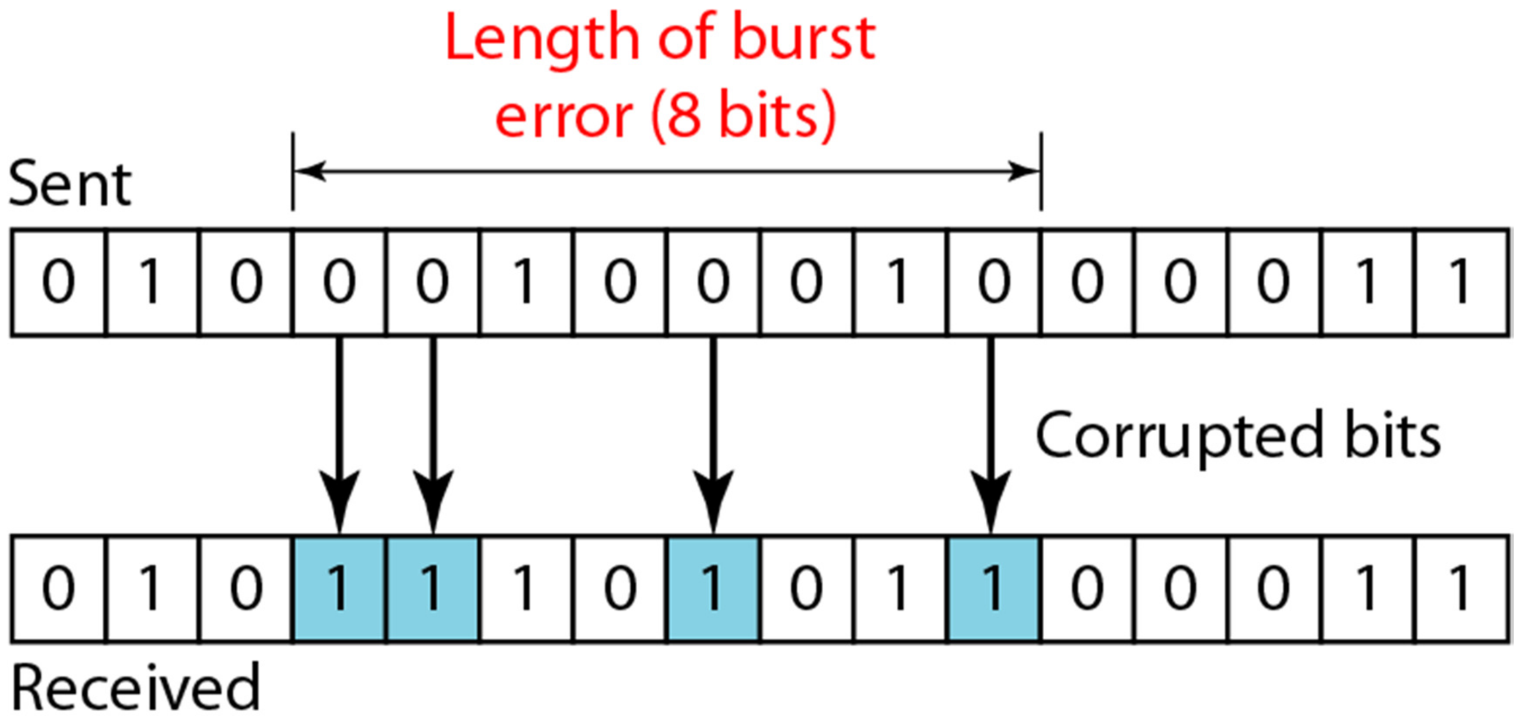


Burst Error

- Burst Error - the *string* of bits between 2 successive corrupted bits including the two bits in error
  - More likely to happen due to noise duration is usually longer than the duration of 1 bit

- Burst error length B - the last corrupted bit in a burst and the first corrupted bit in the following burst must be separated by B+1or more correct bits

Burst error of length 8



Assume the remaining bits are correct

Error Detection

Redundancy

- **Parity Check**
- **Cyclic Redundancy Check (CRC)**

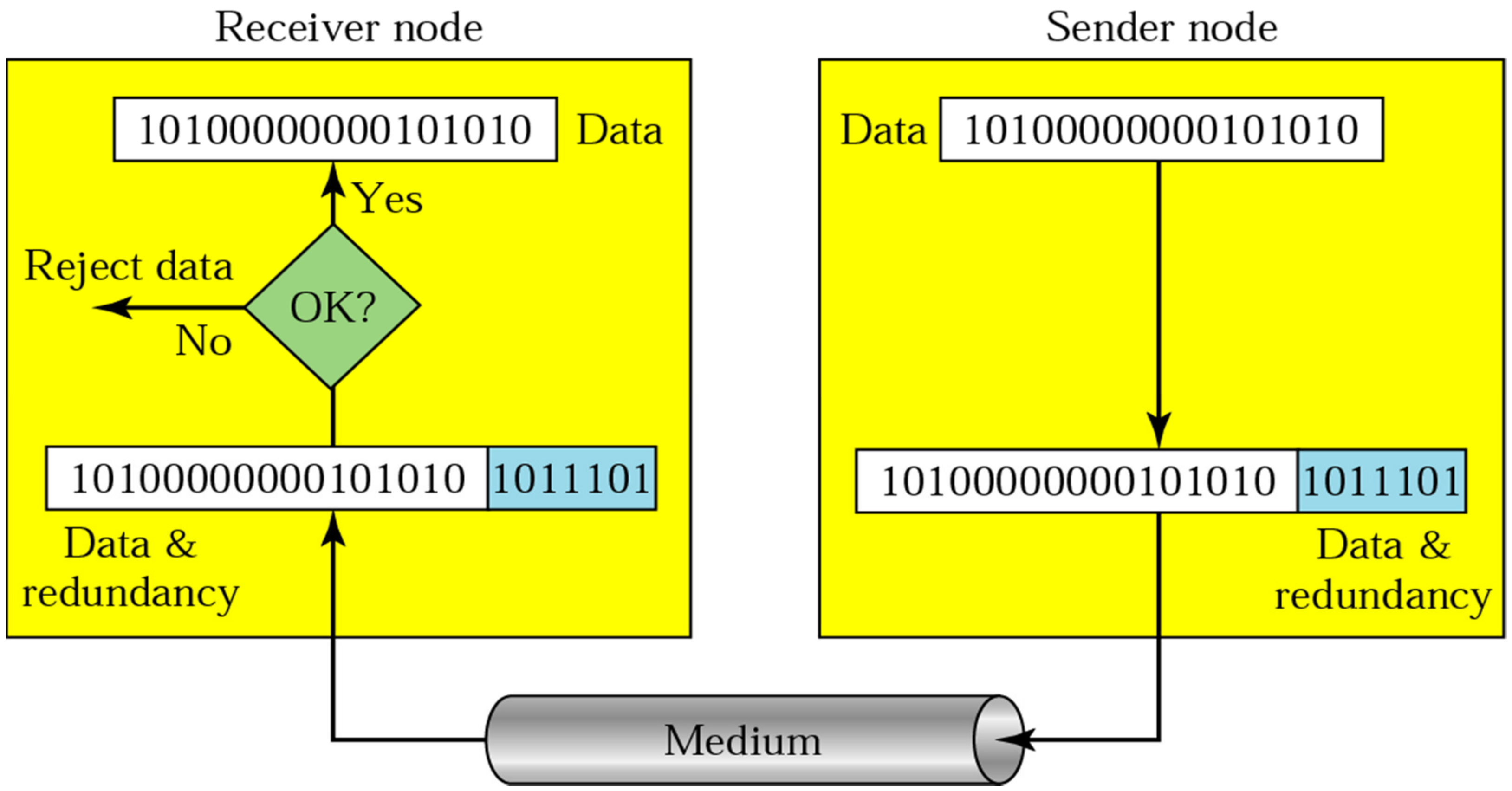
Error detection uses the concept of redundancy, which means adding extra bits for detecting errors at the destination.

Error Detection (Feedback Error Control)

- Each character or frame includes only sufficient additional information to enable the receiver to detect errors
- a retransmission control scheme is used
- the predominant method because of less additional bits required

Redundancy

- ***Adding extra bits for detecting errors at the destination***



10.2 Linear Block Code – Parity Check Method

- Most common and simple for character-oriented transmission
- The data bits in each character are inspected prior to transmission and the parity bit is computed
- The parity bit is then added so that the total number of binary 1s is either odd or even:
  - If odd parity is used, no. of 1s is odd.
  - If even parity is used, no. of 1s is even.

Parity Check Method (Sender side)

- Sender obtains data from upper layer
- Determine the parity bit (either 0 or 1) so that the total number of binary 1s is either odd or even:
  - If odd parity is used, no. of 1s is odd.
  - If even parity is used, no. of 1s is even.
- Add the parity bit to data and send it out

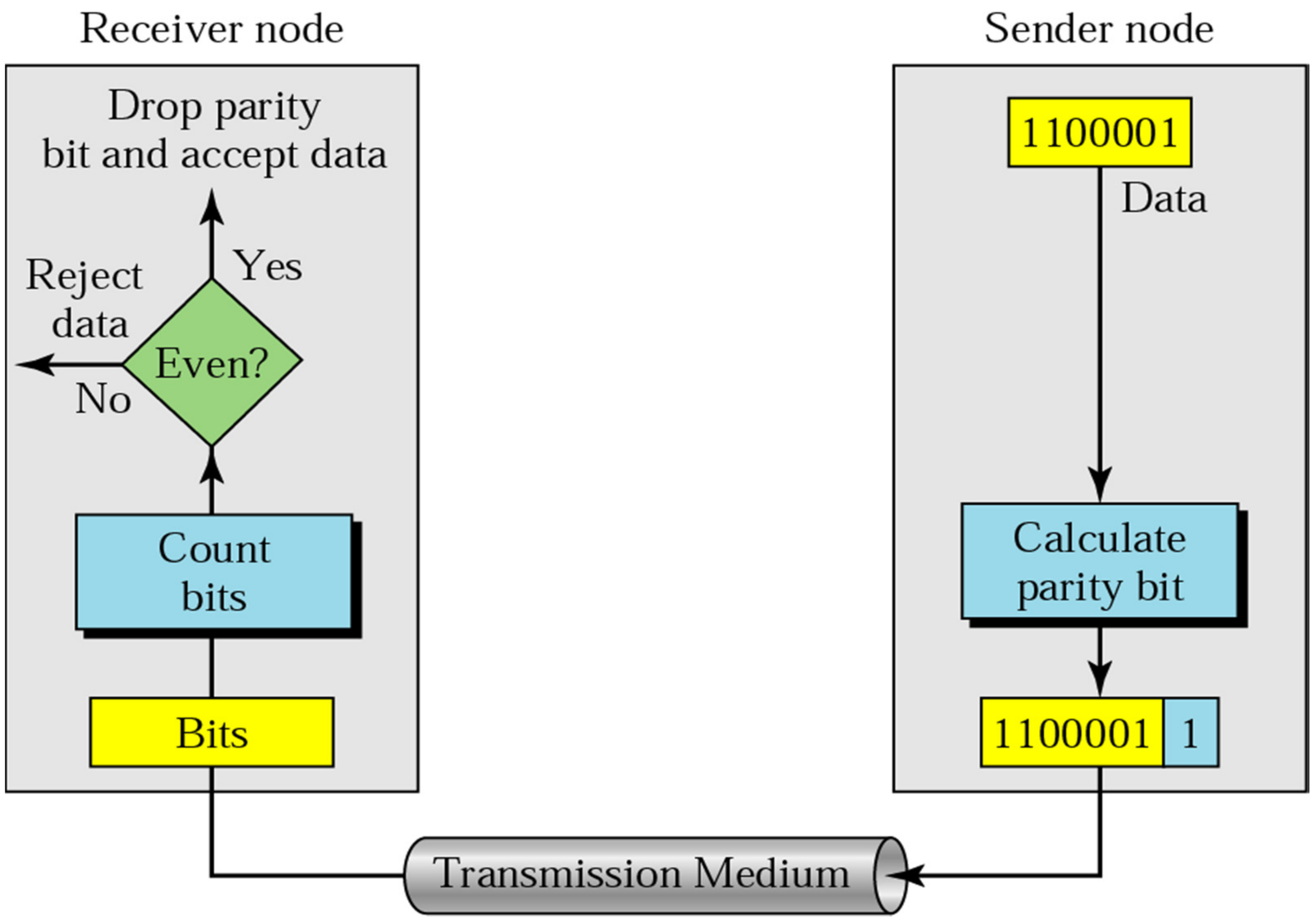
Parity Check Method (Receiver side)

- Receiver receives bits from lower layer
- Count the number of 1s
- Error is detected if the number does not match, i.e.
  - Even number of 1s in odd parity
  - Odd number of 1s in even parity

Limitation

- Single parity bit is used
- Can only detect burst errors with odd number of error bits
- If even number of bits are corrupted, the errors cannot be detected

Even-parity concept



Example

Suppose the sender wants to send the word *world*. In 7-bit ASCII the five characters are coded as

1110111 1101111 1110010 1101100 1100100

The following shows the actual bits sent using even parity:

1110111**<u>0</u>** 1101111**<u>0</u>** 1110010**<u>0</u>** 1101100**<u>0</u>** 1100100**<u>1</u>**

Example (no error)

Now suppose the word *world* in Example 1 is received by the receiver without being corrupted in transmission.

11101110   11011110   11100100   11011000   11001001

The receiver counts the number of 1s in each character and comes up with even numbers (6, 6, 4, 4, 4).

The data are accepted.

Example (with error)

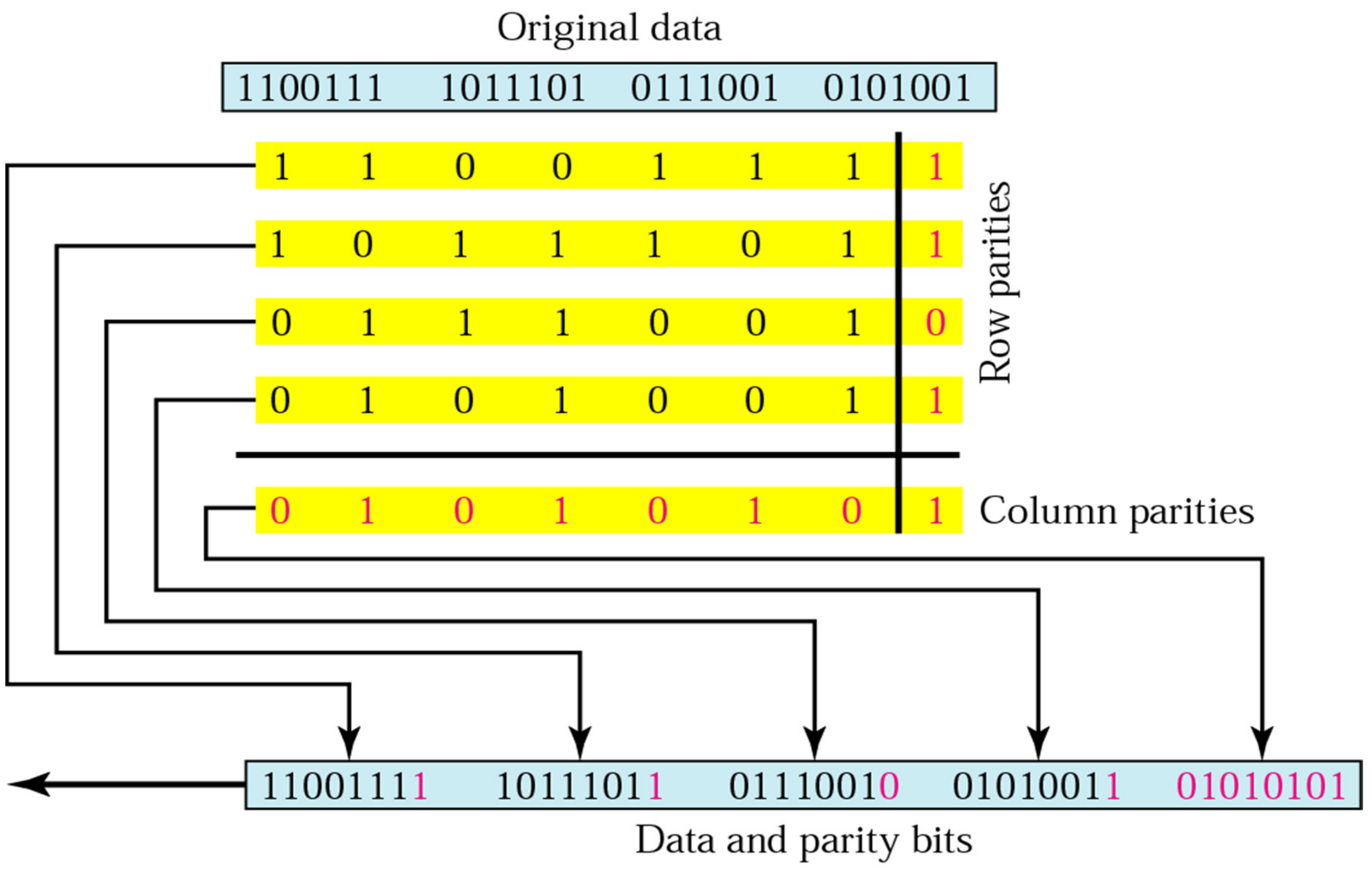
Now suppose the word *world* in Example 1 is corrupted during transmission.

11111110 11011110 11101100 11011000 11001001

The receiver counts the number of 1s in each character and comes up with even and odd numbers (7, 6, 5, 4, 4).

The receiver knows that the data are corrupted, discards them, and asks for retransmission.

Two-dimensional parity check



Two-dimensional parity check

- Organize the bits in rows and columns
- Calculate the parity bit for each row
  - same as the single-bit parity check method
  - odd parity or even parity
- Calculate the parity bit for each column
  - also for the parity bit of each row

Example

Suppose the following block is sent:

10101001 00111001 11011101 11100111 10101010

However, it is hit by a noise with 8-bit duration, and some bits are corrupted.

1010**0011** **1000**1001 11011101 11100111 10101010

When the receiver checks the parity bits, some of the bits do not follow the even-parity rule and the whole block is discarded.

10100011 1000100**1** 11011101 11100111 **1**0**101**0**1**0

What kind of errors can (/cannot) be detected by two-dimensional parity check?

one pattern of error that remains elusive. If two bits in one data unit are damaged and two bits in exactly same position in another data unit are also damaged, the 2-D Parity check checker will not detect an error.

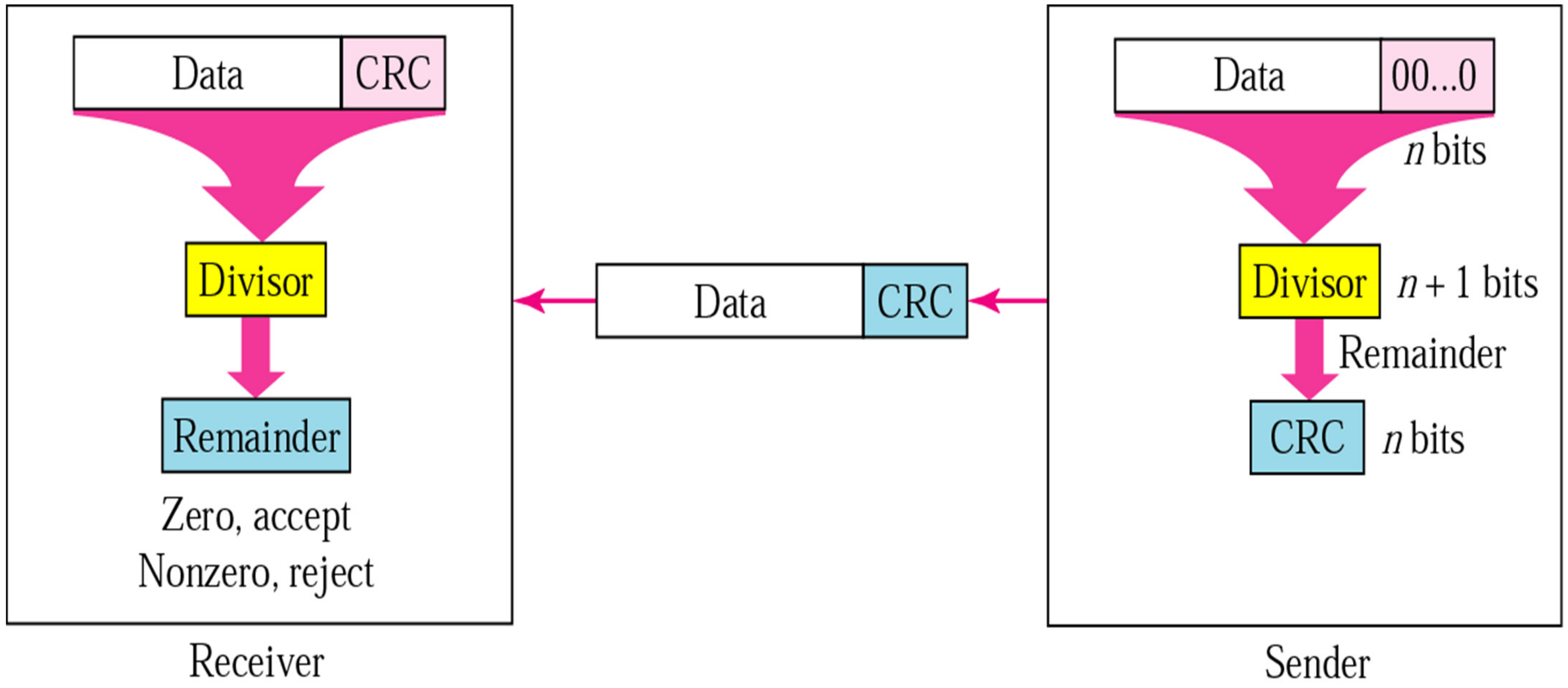
10.3 Cyclic Codes – Cyclic Redundancy Check (CRC)

- Parity check does not provide a reliable detection against error bursts
- A single set of check digits is computed and appended to the tail of each frame transmitted
- The receiver then performs a similar computation to check whether error occurs

Cyclic Redundancy Check (CRC)

- The number of check digits varies (16, 32 are most common)
- The computed check digits are called cyclic redundancy check (CRC) digits or frame check sequence (FCS) digits
- Uses property of modulo 2 arithmetic
  - uses binary addition with no carries and is effectively an XOR operation.

CRC generator and checker



CRC Procedure

1. A set of n “0”s is appended at the end of a k-bit data frame M.
   - This is the same as multiplying the frame by 2^n so we have ($M \times 2^n$)
   - $k > n$

2. ($M \times 2^n$) is divided modulo 2 by a (n+1)-bit binary number G (the generator polynomial), giving the remainder R (n-bit) which is the CRC (or FCS) digits

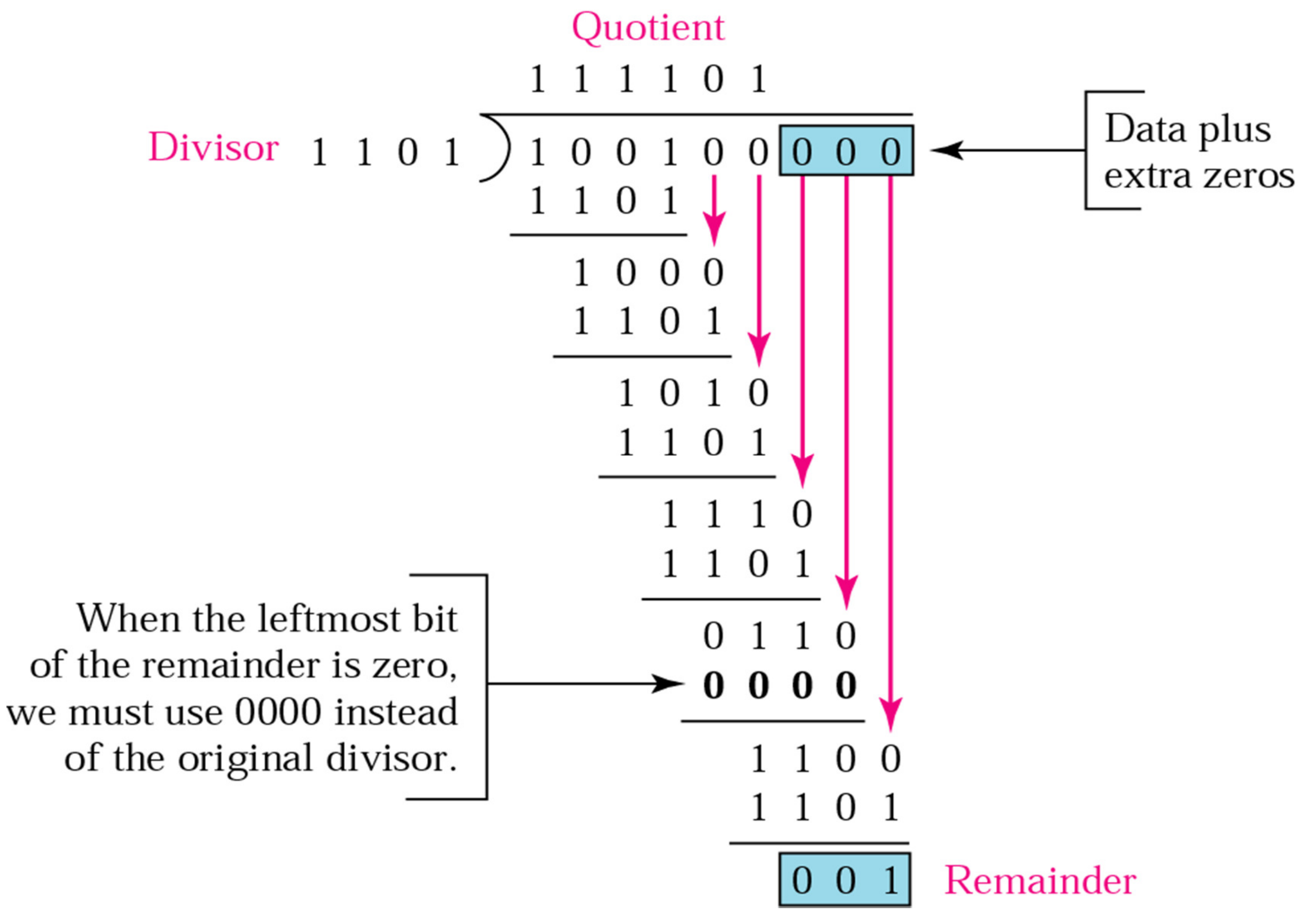
CRC Procedure

3. R is appended at the end of M and then transmit M & R together

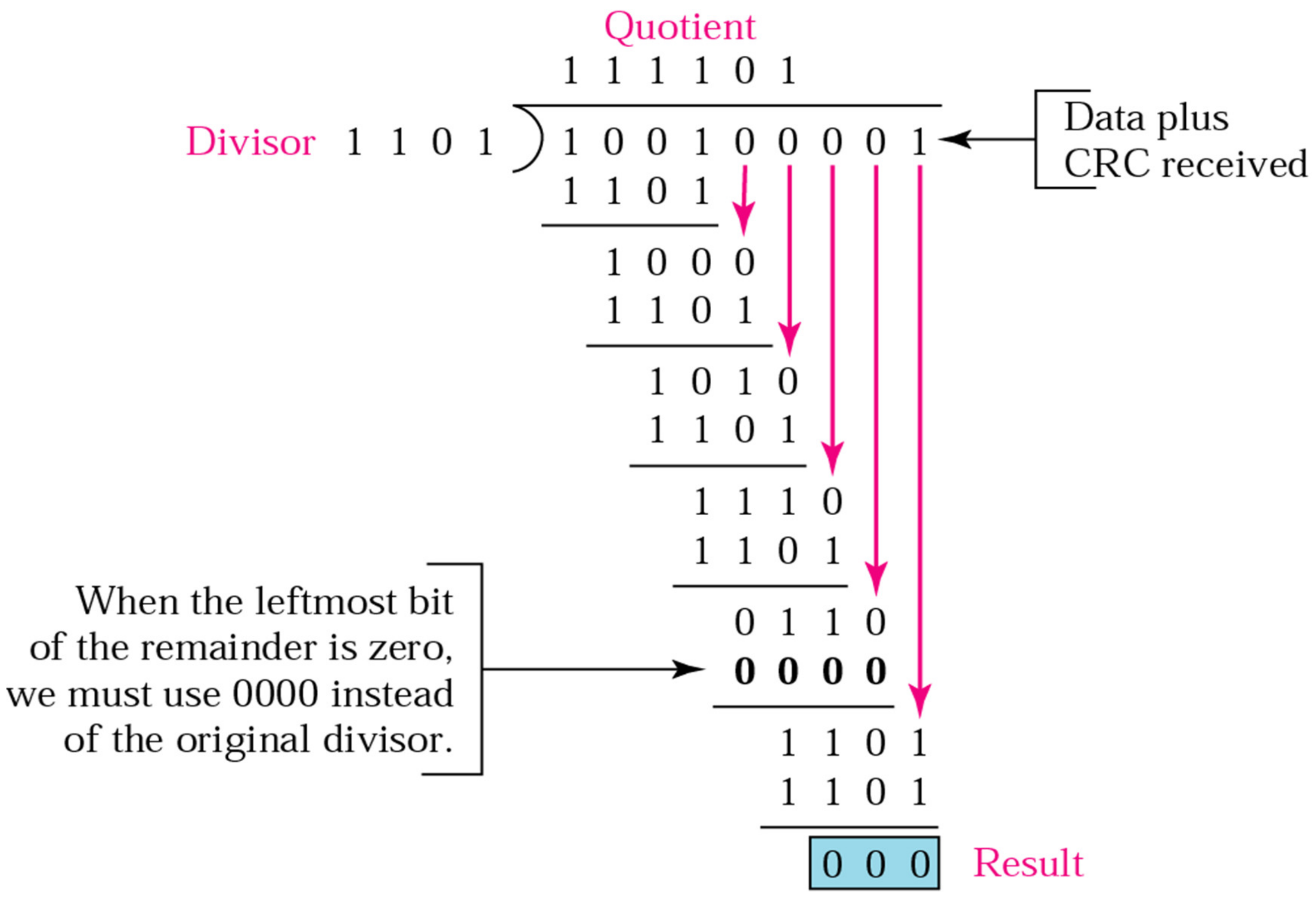
4. At the receiver, the received bit stream including M & R is again divided by the same polynomial G, i.e. (M x 2^n + R)/G

5. If the remainder = 0 then no errors; otherwise errors occur

Binary division in a CRC generator



Binary division in CRC checker



CRC Example

Message: 11100110

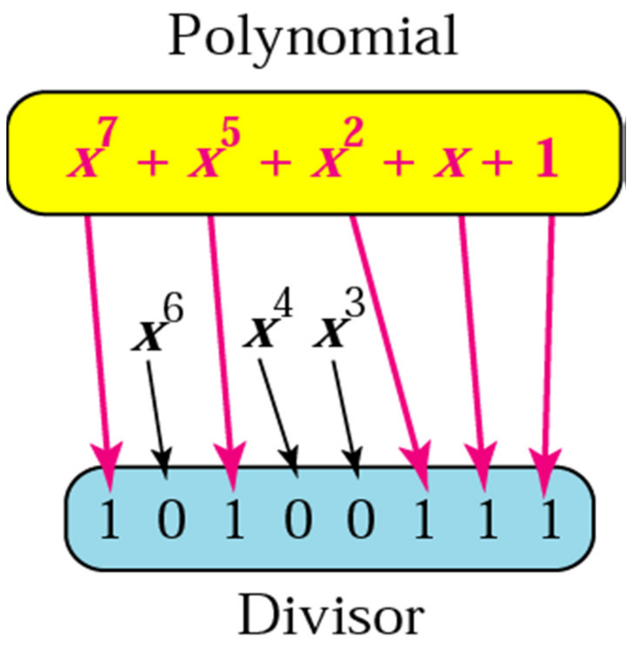
Polynomial: 11001 (i.e. $x^4 + x^3 + 1$)

What is the CRC?

- Answer: CRC should be 0110

- Division is equivalent to performing the XOR operation bit by bit in parallel as each bit in the frame is processed
- Could be implemented in hardware

A polynomial representing a divisor



The polynomial is of degree 7 but need 8 bits

Standard CRC

- Represent the generator polynomial by showing those positions that are binary 1 as powers of X
- CRC-CCITT = $X^{16} + X^{12} + X^{5} + 1(X^{0})$
  this is equivalent to 10001000000100001
- Also call CRC-16
- it will detect all error bursts of length less than or equal to 16 bits.

Standard CRC

- CRC-CCITT (CRC-16) is used extensively with WANs
- CRC-32 is used in most LANs
- In general, a (n+1)-bit generator polynomial, *with degree n*, will detect:
  - all single-bit, double-bit and odd number of bit errors
  - all error bursts with length <= n
  - most error bursts with length > n

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

Example

The CRC-12

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

which has a degree of 12, will detect all burst errors affecting an odd number of bits, will detect all burst errors with a length less than or equal to 12, and will detect 99.97 percent of burst errors with a length of 13 or more.

Error Correction (Forward Error Control)

- Each transmitted character or frame contains additional (redundant) information so that the receiver can detect, locate & correct the errors
  - e.g. Hamming code

Forward Error Control (FEC)

- With FEC, sufficient additional check digits are added to each transmitted message to enable the receiver
  - to detect the presence of one or more errors in a received message
  - and to locate the position of the error
- Correction is achieved simply by inverting the incorrect bit(s)

Forward Error Control (FEC)

- Number of check digits is much higher than that for just error detection
- Less efficient than retransmission with error detection
- But when the round-trip delay is long or return path is not available, FEC methods are often used

Summary

- Single-bit errors and burst errors
- Error detection by redundancy
- Parity Check
  - Odd parity and even parity
  - Two-dimentional Parity Check
- CRC / FCS
  - n-bit check digits detect burst errors <= n bits
- FEC
  - More bits are used for error location and correction

- Revision Quiz
  - http://highered.mheducation.com/sites/0073376221/student_view0/chapter10/quizzes.html