Data and Computer Communications

Error Detection

Mohamed Khedr

http://webmail.aast.edu/~khedr

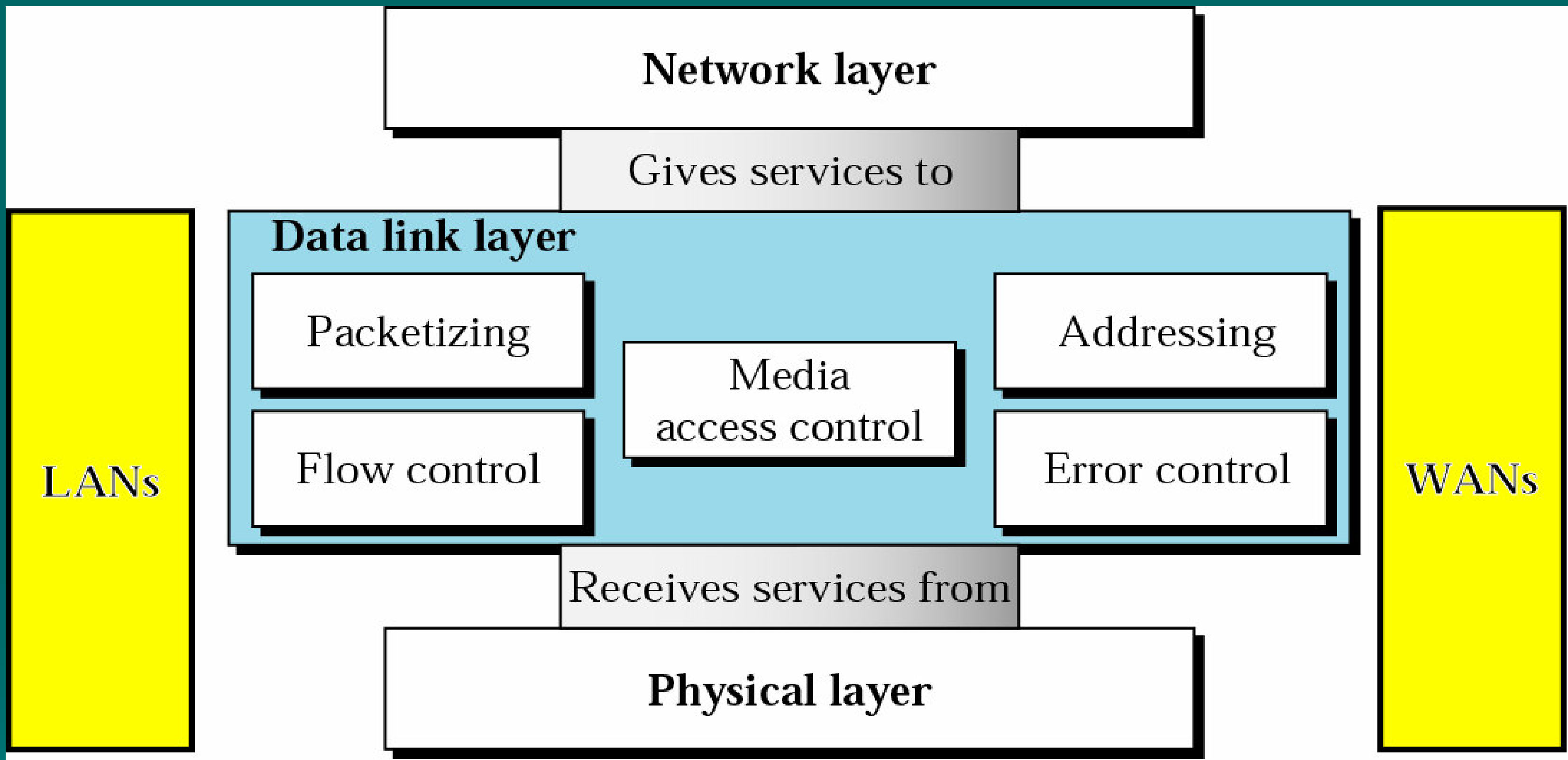
Syllabus

- Tentatively

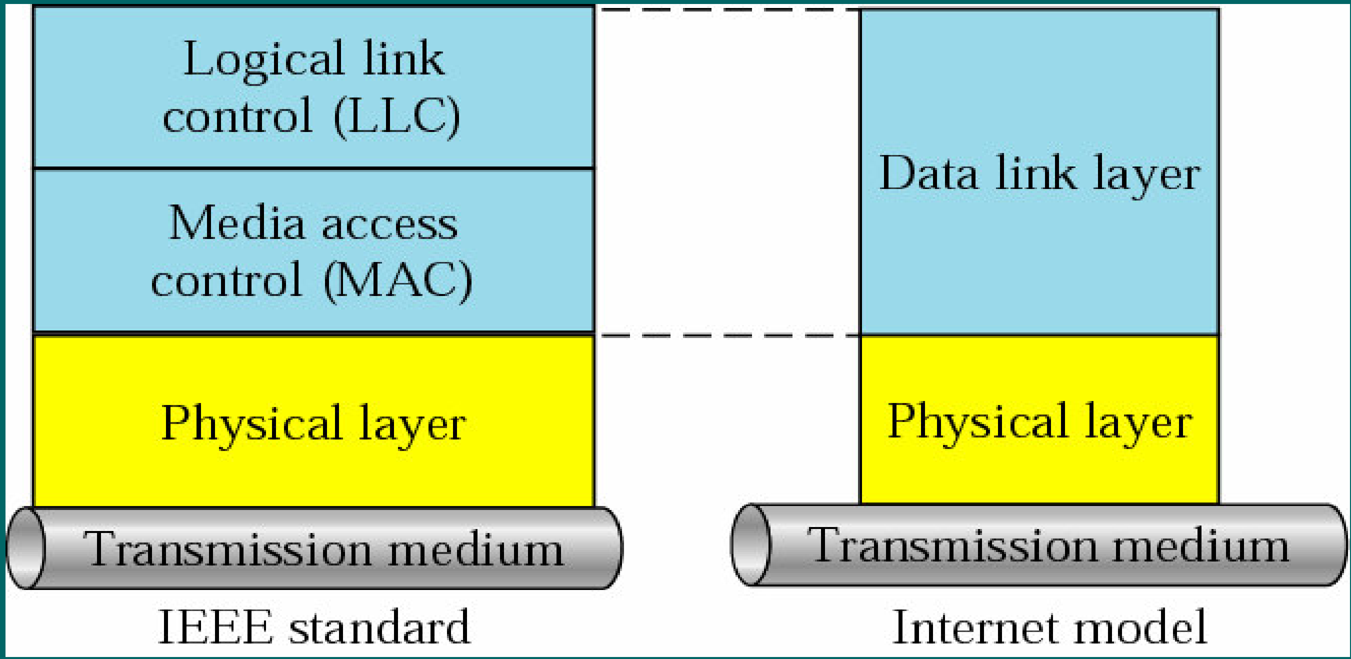
Week 1	Overview
Week 2	Data Transmission
Week 3	Signal encoding techniques
Week 4	**Error Detection**
Week 5	Error correction
Week 6	Flow Control
Week 7	Error control
Week 8	HDLC
Week 9	Multiplexing
Week 10	Spread spectrum
Week 11	Wireless channel characteristics
Week 12	OFDM
Week 13	Packet switching
Week 14	Routing
Week 15	Revision

Data Link Layer

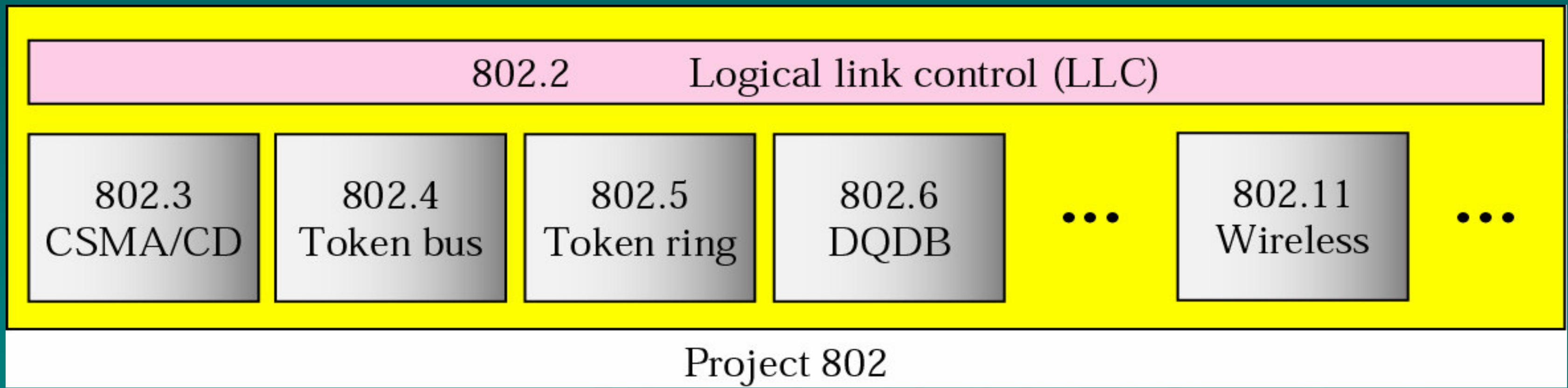
Position of the data-link layer



LLC and MAC sublayers



IEEE standards for LANs

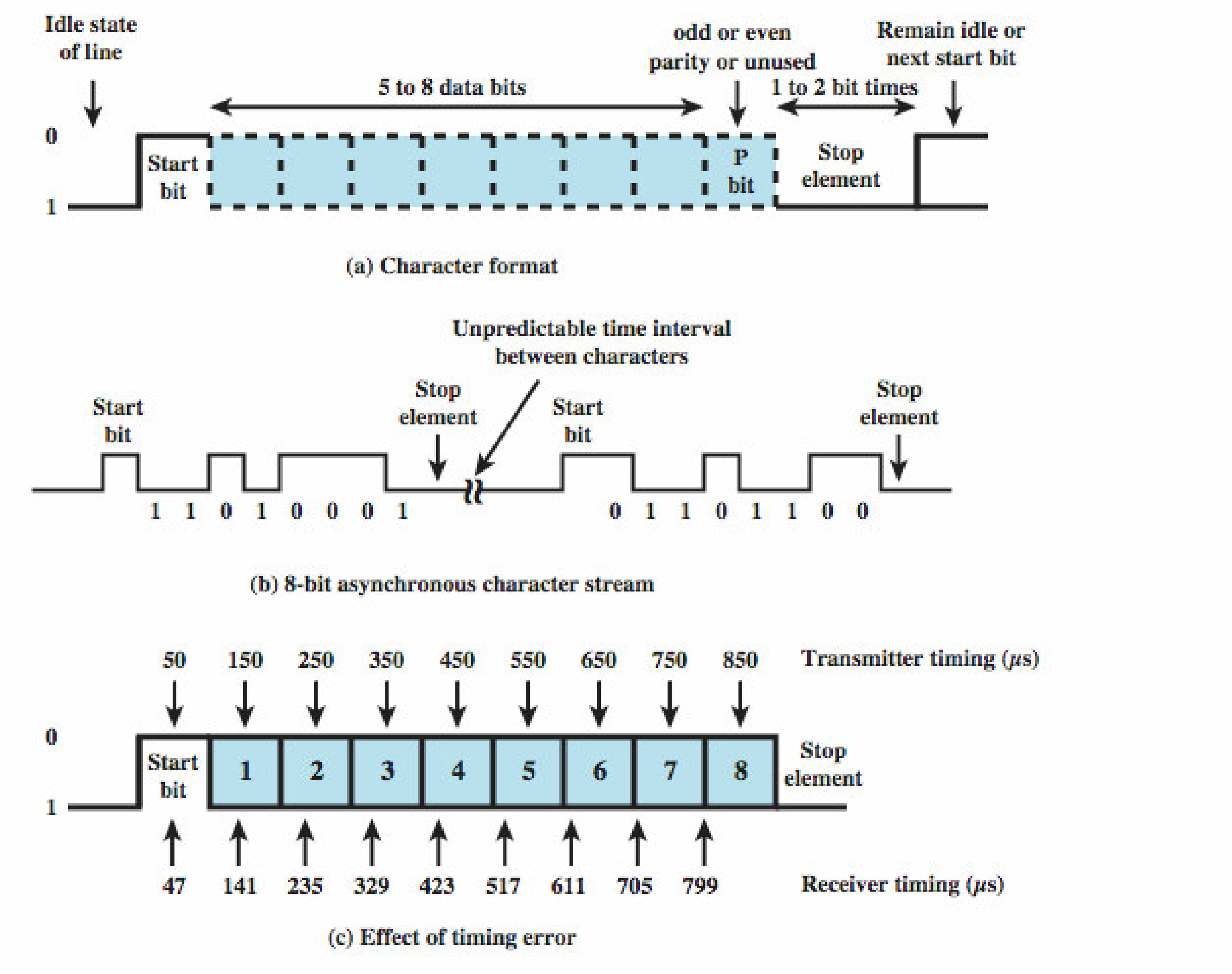


Asynchronous and Synchronous Transmission

- timing problems require a mechanism to synchronize the transmitter and receiver
  - receiver samples stream at bit intervals
  - if clocks not aligned and drifting will sample at wrong time after sufficient bits are sent
- two solutions to synchronizing clocks
  - asynchronous transmission
  - synchronous transmission

Asynchronous Transmission

- simple
- cheap
- overhead of 2 or 3 bits per char (~20%)
- good for data with large gaps (keyboard)



Synchronous Transmission

- block of data transmitted sent as a frame
- clocks must be synchronized
  - can use separate clock line
  - or embed clock signal in data
- need to indicate start and end of block
  - use preamble and postamble
- more efficient (lower overhead) than async

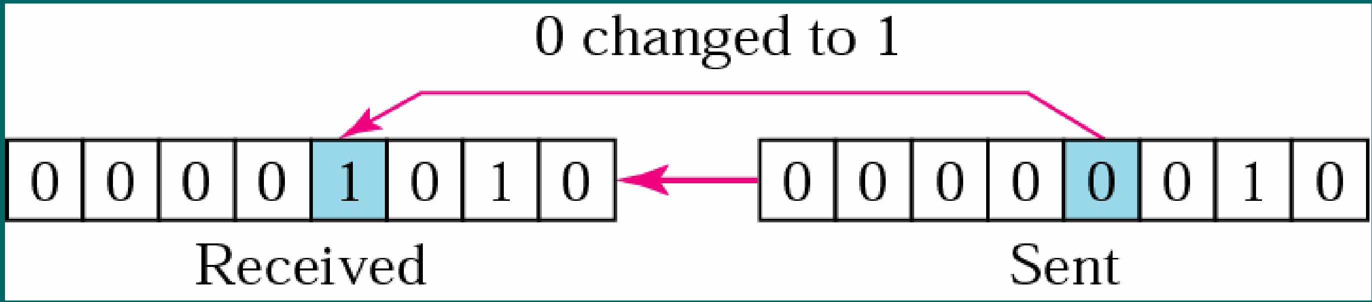
Note:

Data can be corrupted during transmission. For reliable communication, errors must be detected and corrected.

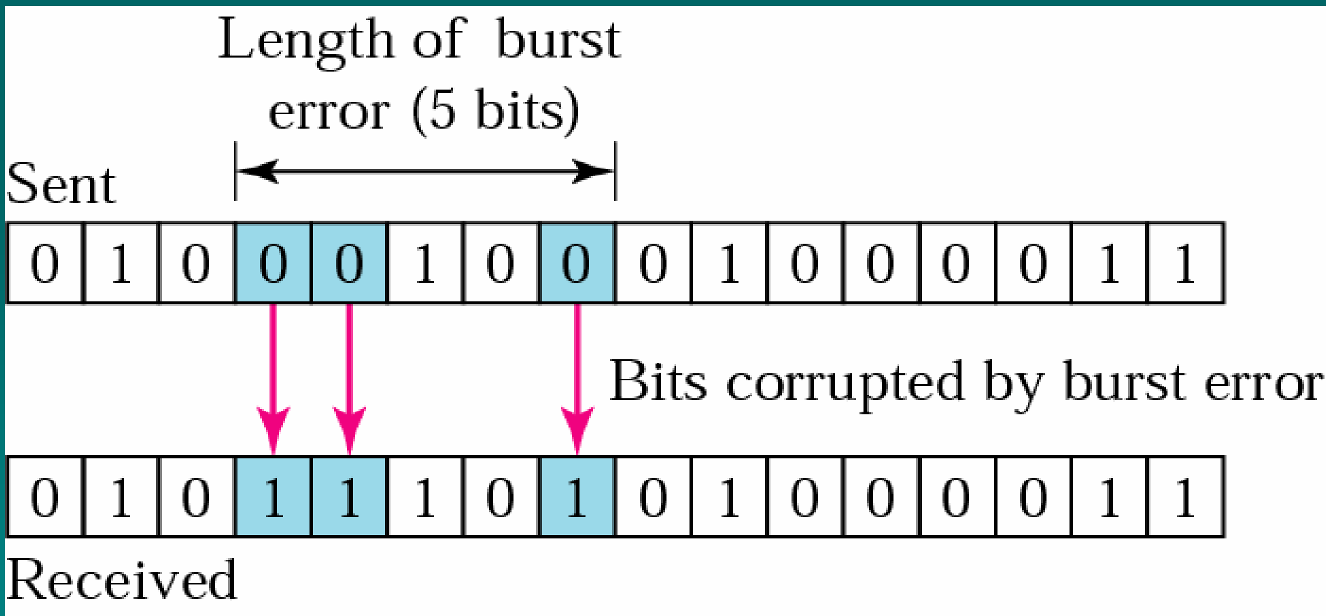
Types of Error

- an error occurs when a bit is altered between transmission and reception
- single bit errors
  - only one bit altered
  - caused by white noise
- burst errors
  - contiguous sequence of *B* bits in which first last and any number of intermediate bits in error
  - caused by impulse noise or by fading in wireless
  - effect greater at higher data rates

10.1 ***Single-bit error***



10.2 ***Burst error of length 5***



Probabilities of Error

- P_b= Probability that a bit is received in error, AKA as BER
- P_1= Probability that a frame arrives with no bits in error
- P_2= Probability that a frame arrives with one or more undetectable error in the presence of error detection algorithm
- P_3=Probability that a frame arrives with one or more detected bit errors in the presence of error detection algorithm

Probabilities of Error

- In case of no error detection algorithm is used
  - P_3=zero
  - $P_1=(1-P_b)^n$
  - $P_2=1-P_1$

10.2 Detection

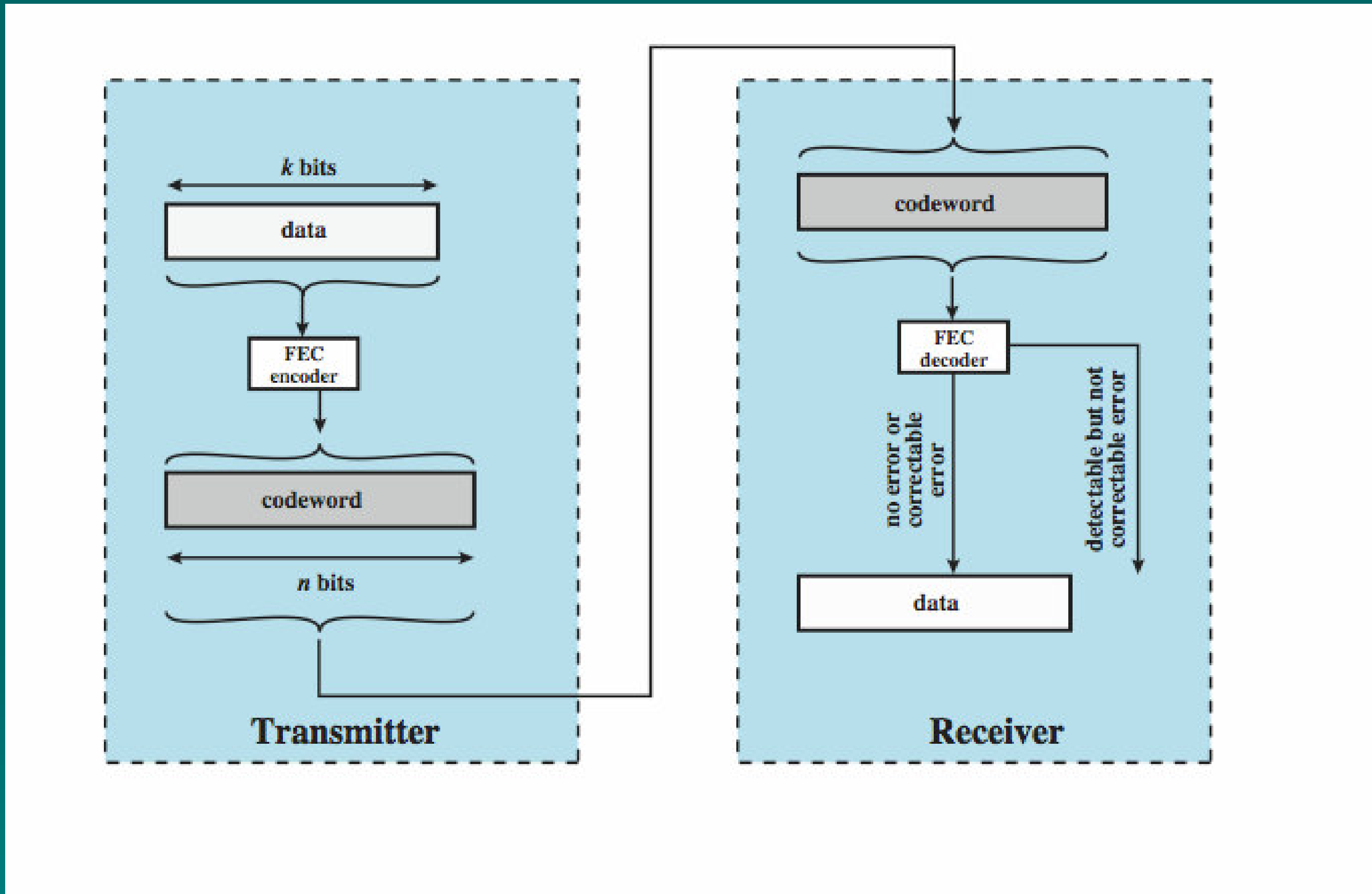
Redundancy

Parity Check

Cyclic Redundancy Check (CRC)

Checksum

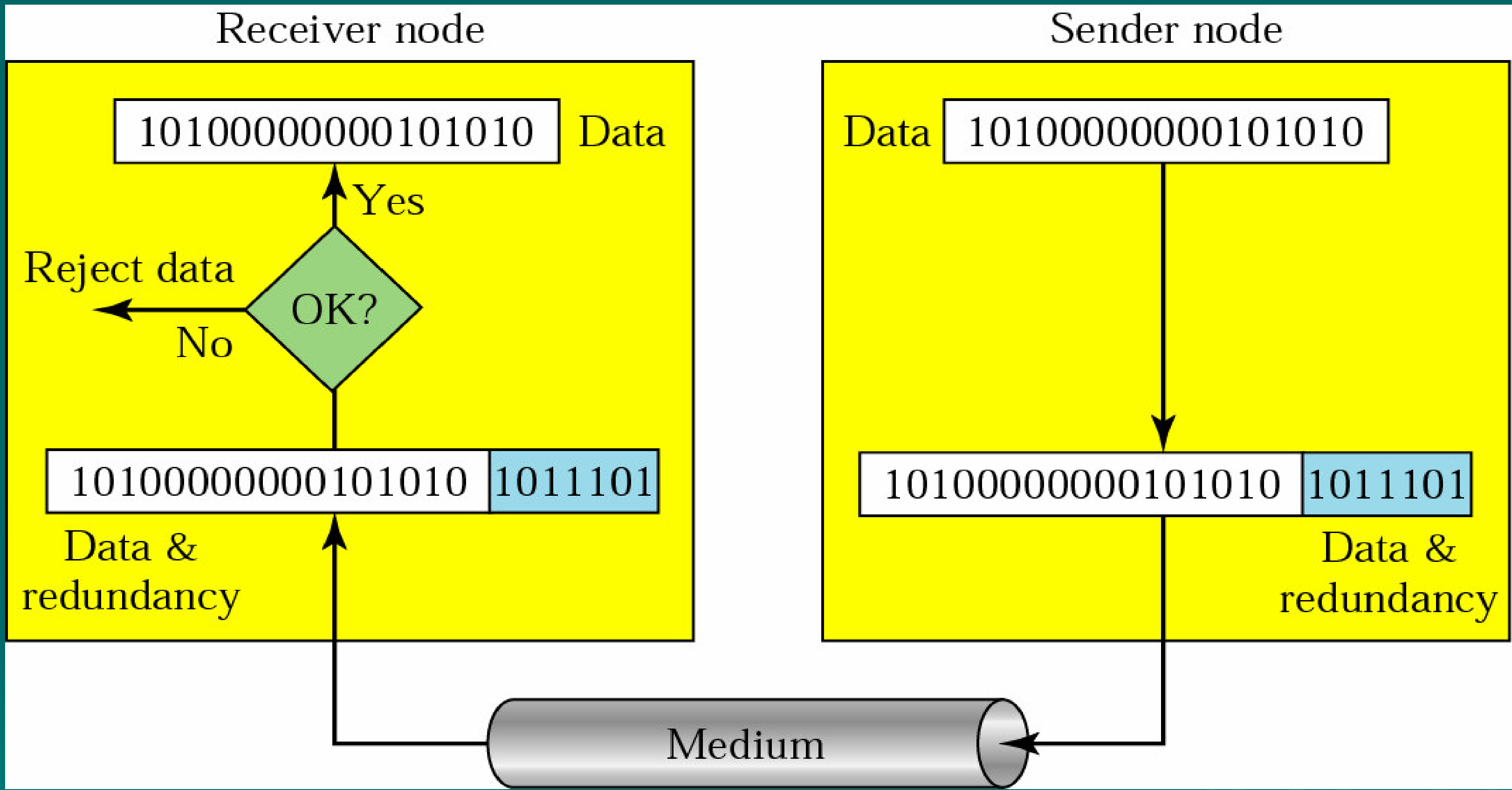
Error Detection Process



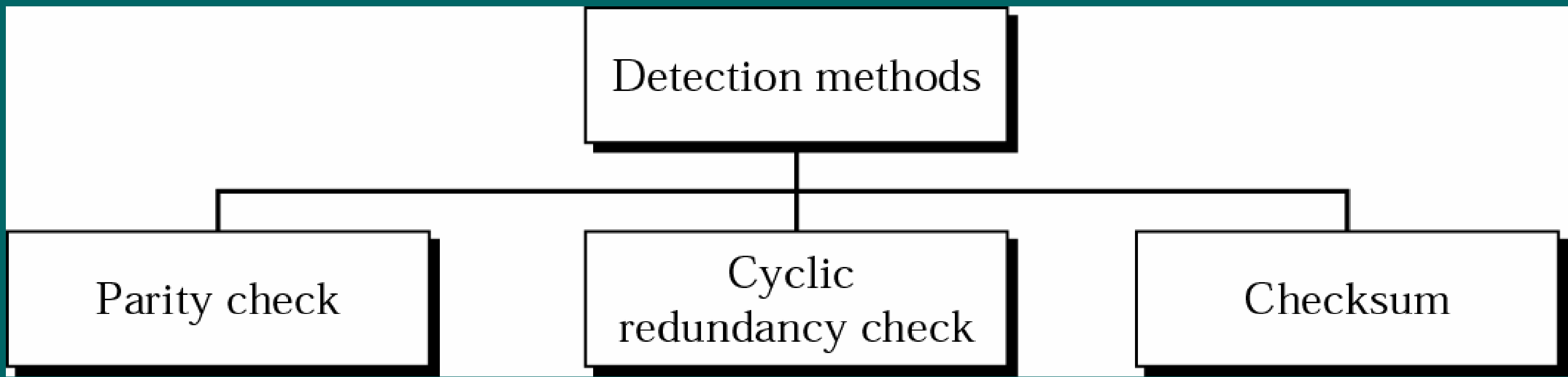
Note:

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

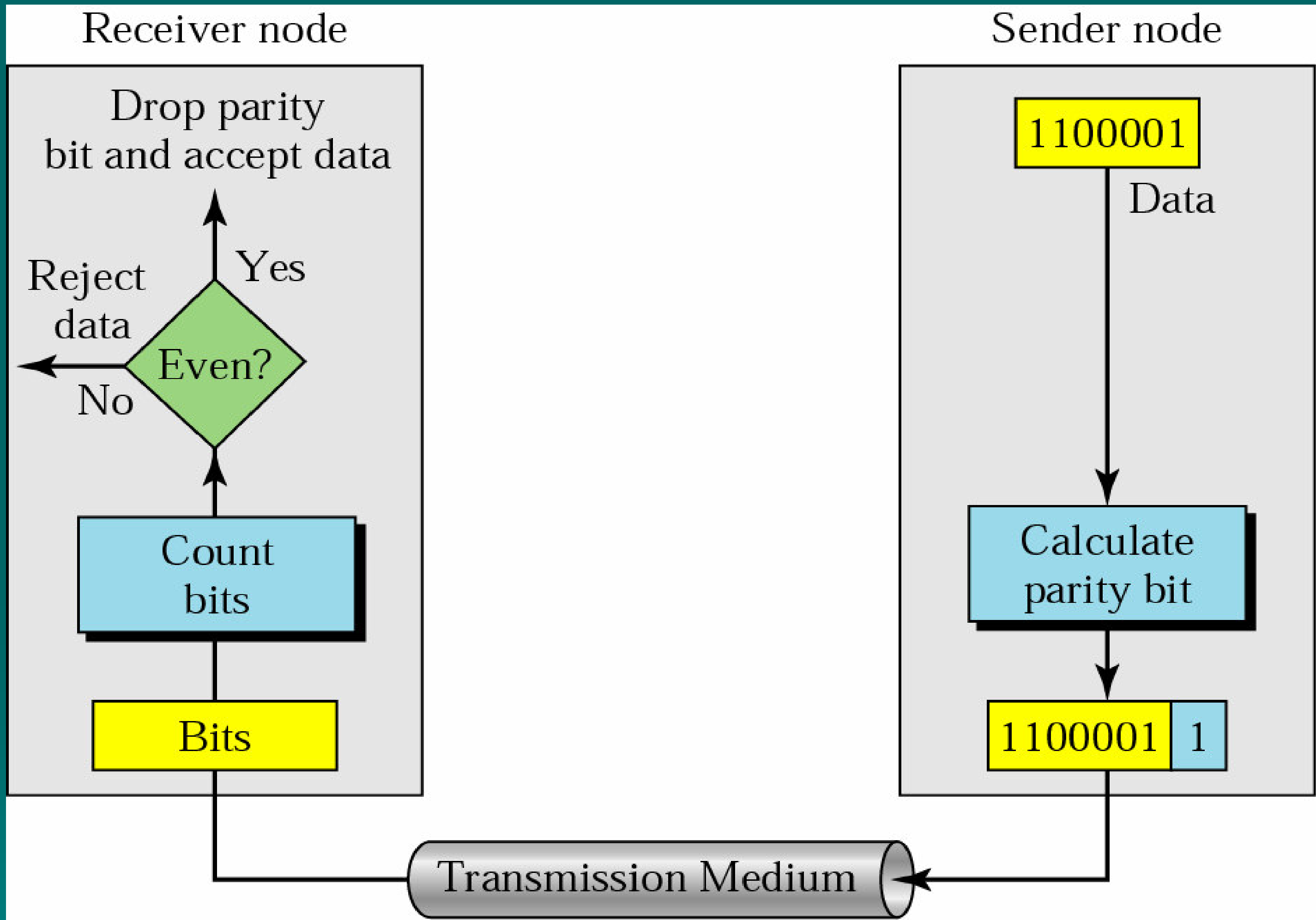
10.3 *Redundancy*



10.4 *Detection methods*



10.5 *Even-parity concept*



Note:

In parity check, a parity bit is added to every data unit so that the total number of 1s is even (or odd for odd-parity).

Example 1

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

1110111 1101111 1110010 1101100 1100100

The following shows the actual bits sent

1110111**0** 1101111**0** 1110010**0** 1101100**0** 1100100**1**

Example 2

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 1s in each character and comes up with even numbers (6, 6, 4, 4, 4). The data are accepted.

Example 3

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

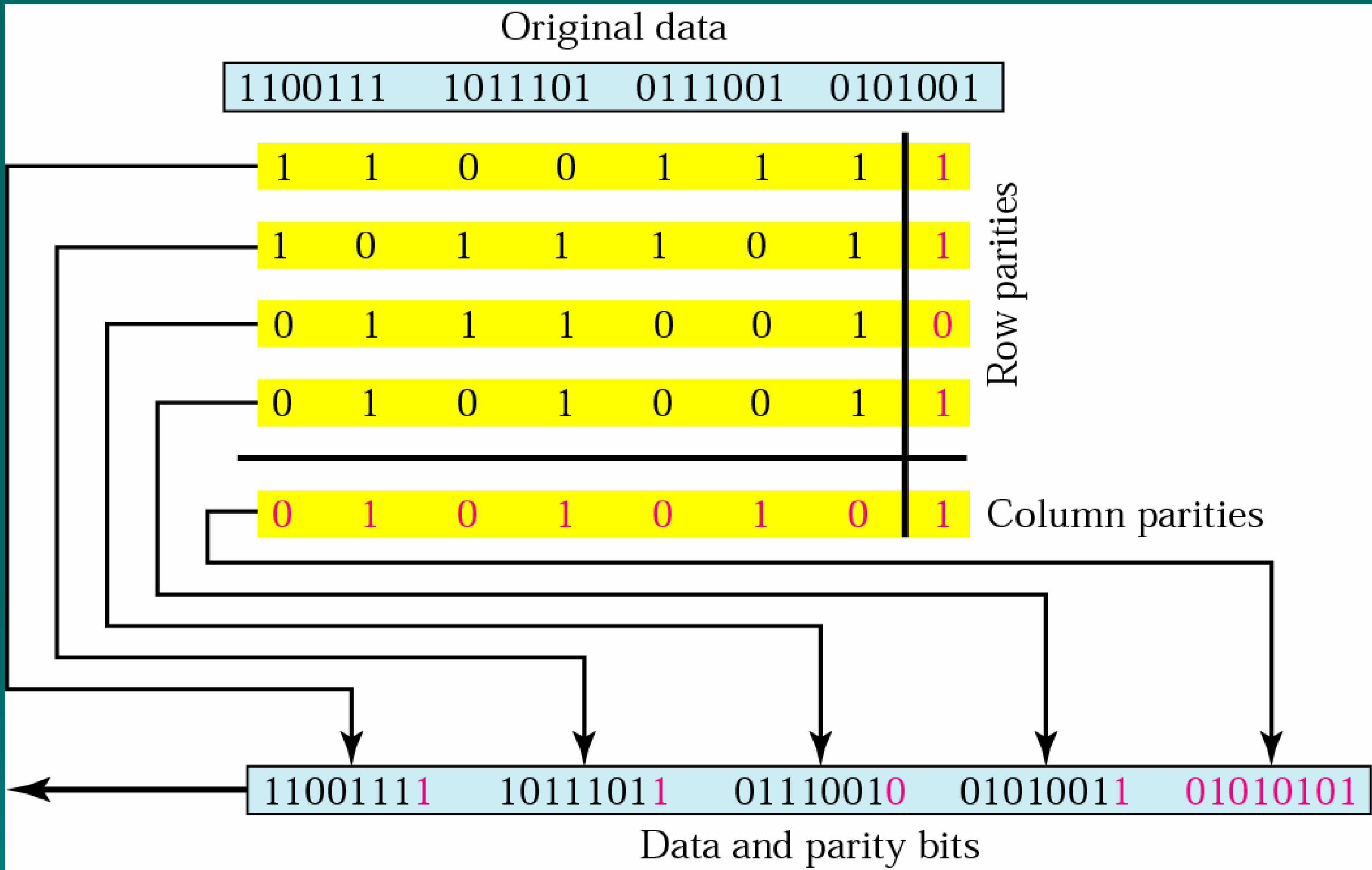
11111110 11011110 11101100 11011000 11001001

The receiver counts the 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.

Note:

Simple parity check can detect all single-bit errors. It can detect burst errors only if the total number of errors in each data unit is odd.

10.6 *Two-dimensional parity*



Example 4

Suppose the following block is sent:

10101001   00111001   11011101   11100111   10101010

However, it is hit by a burst noise of length 8, 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.

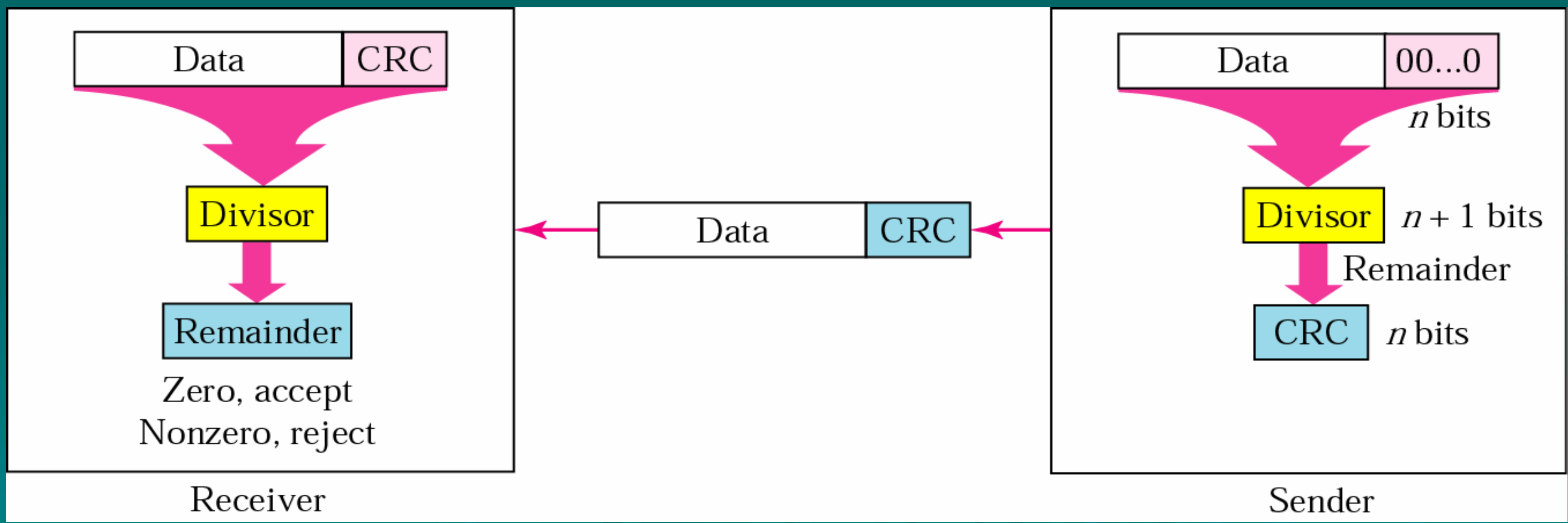
10100011   10001001   11011101   11100111   **1**0**101**0**1**0

Note:

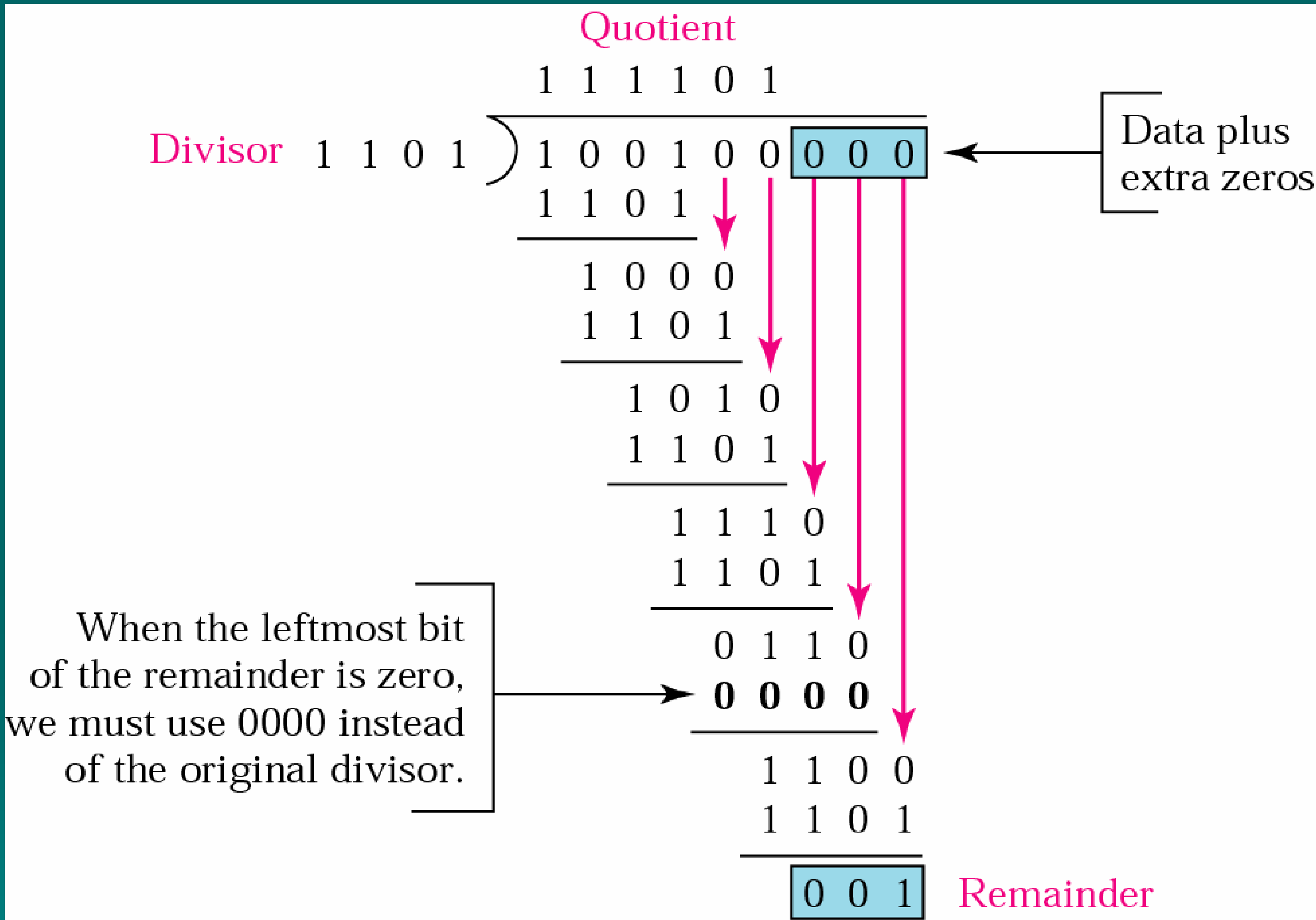
In two-dimensional parity check, a block of bits is divided into rows and a redundant row of bits is added to the whole block.

Cyclic Redundancy Check

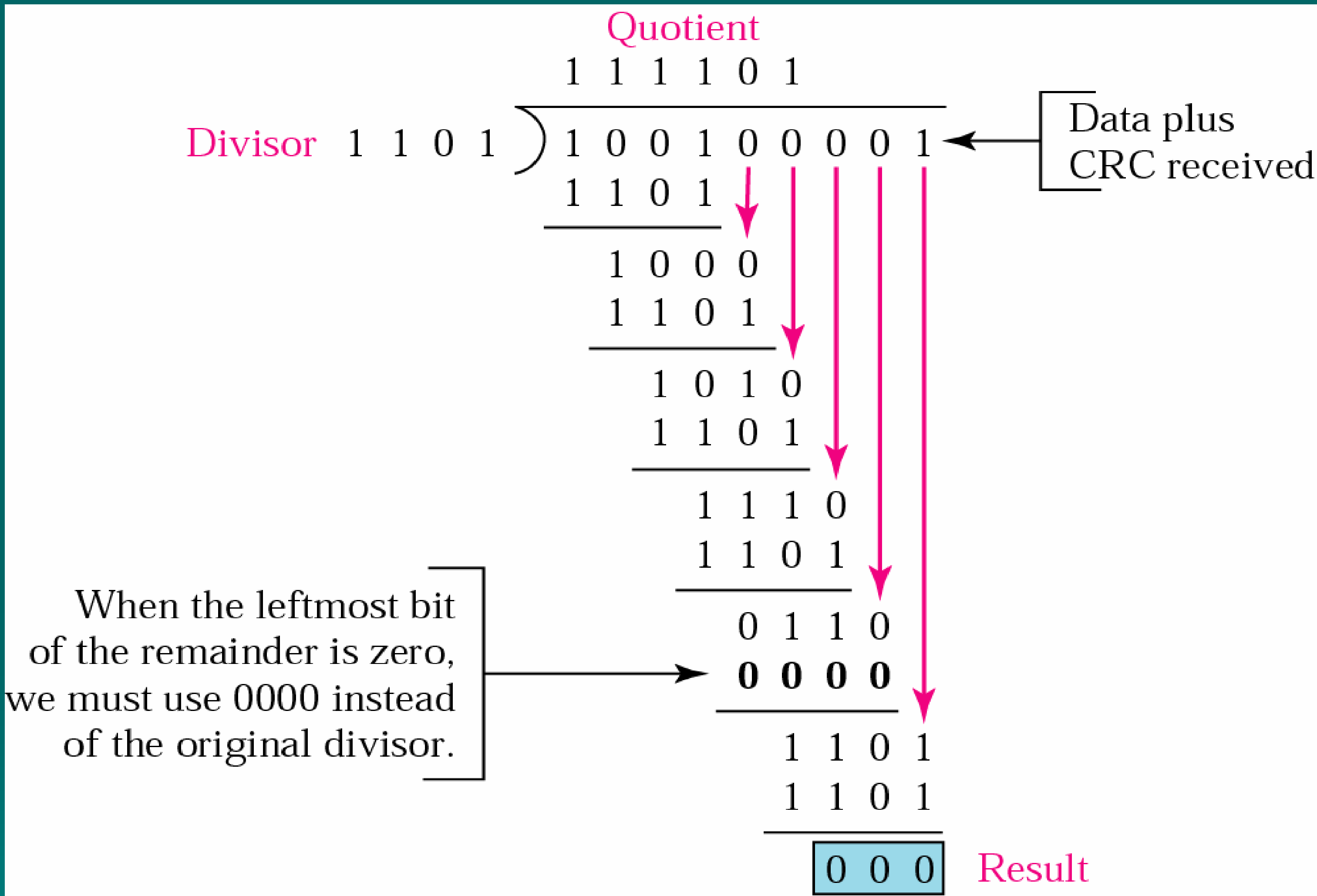
- one of most common and powerful checks
- for block of *k* bits transmitter generates an *n* bit frame check sequence (FCS)
- transmits *k+n* bits which is exactly divisible by some number
- receiver divides frame by that number
  - if no remainder, assume no error
  - for math, see Stallings chapter 6



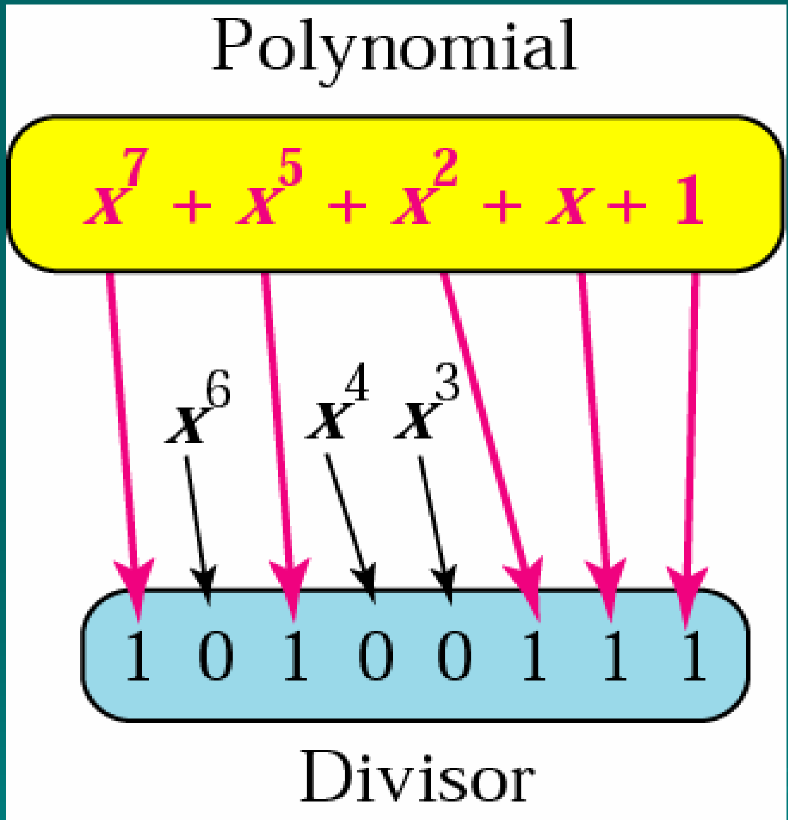
10.8 *Binary division in a CRC generator*



10.9 *Binary division in CRC checker*



$$x^7 + x^5 + x^2 + x + 1$$



- Example

D=1010001101

P=110101

Table 10.1 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
ITU-16	$x^{16} + x^{12} + x^5 + 1$	HDLC
ITU-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

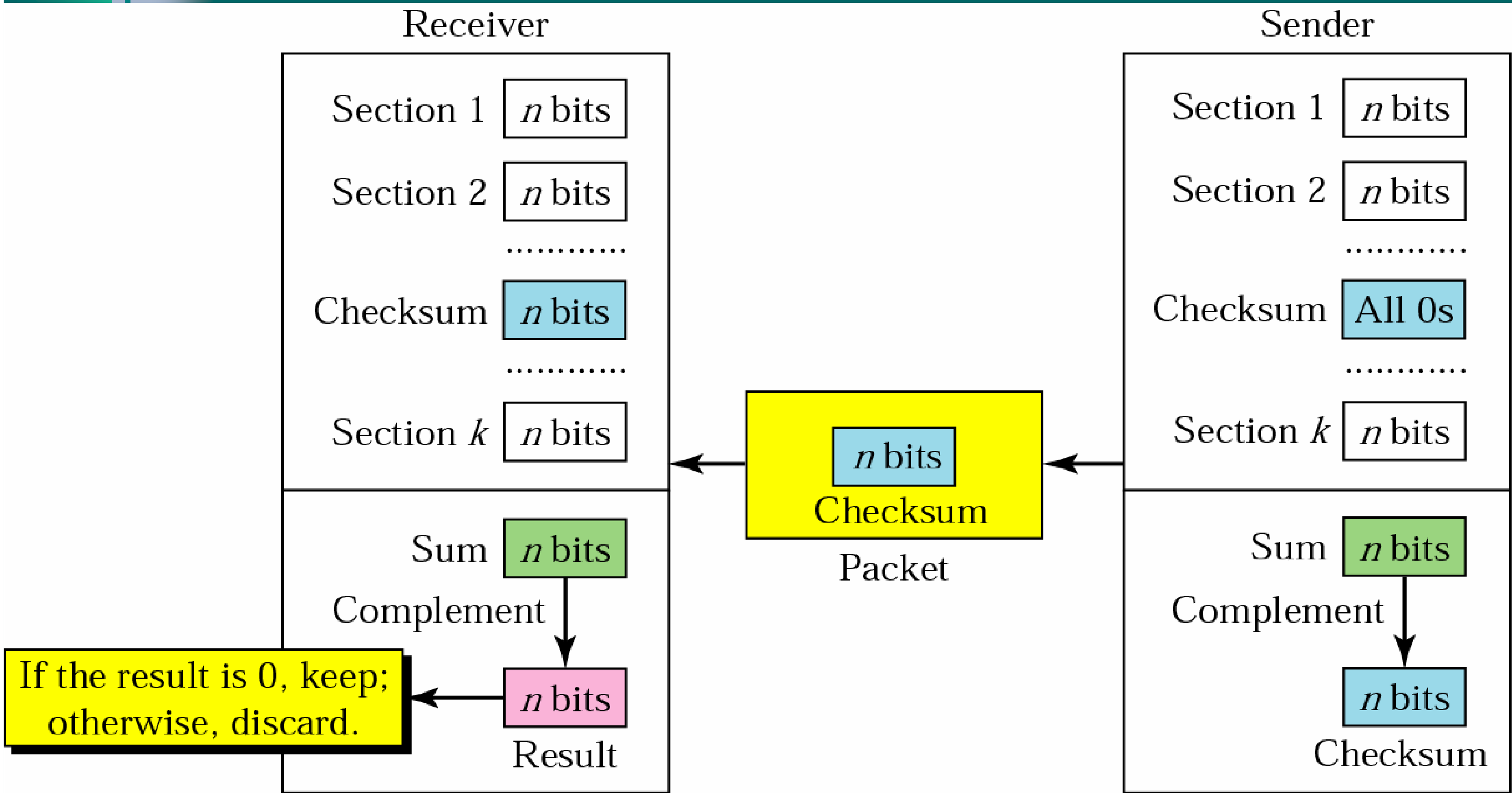
Example 6

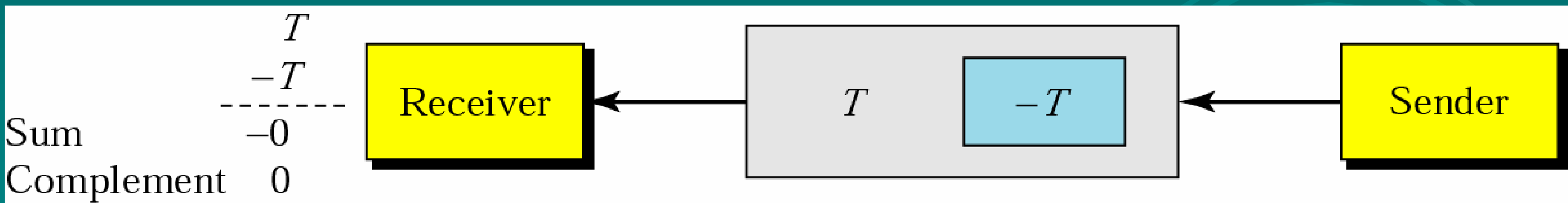
The CRC-12

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

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 the time, burst errors with a length of 12 or more.

10.12 *Checksum*





Example 7

Suppose the following block of 16 bits is to be sent using a checksum of 8 bits.

10101001   00111001

The numbers are added using one's complement

	10101001
	00111001

Sum	11100010
Checksum	**00011101**

The pattern sent is   10101001   00111001   **00011101**

Example 8

Now suppose the receiver receives the pattern sent in Example 7 and there is no error.

10101001 00111001 00011101

When the receiver adds the three sections, it will get all 1s, which, after complementing, is all 0s and shows that there is no error.

	10101001
	00111001
	00011101
Sum	11111111
Complement	**00000000** means that the pattern is OK.

Example 9

Now suppose there is a burst error of length 5 that affects 4 bits.

10101111   11111001   00011101

When the receiver adds the three sections, it gets

	10101111
	11111001
	00011101
Partial Sum	**1** 11000101
Carry	**1**
Sum	11000110
Complement	**00111001** **the pattern is corrupted.**

Thank You