Computer Networks 1

DATA LINK LAYER

Fatemeh Rezaei

K. N. Toosi
University of Technology

OUTLINE

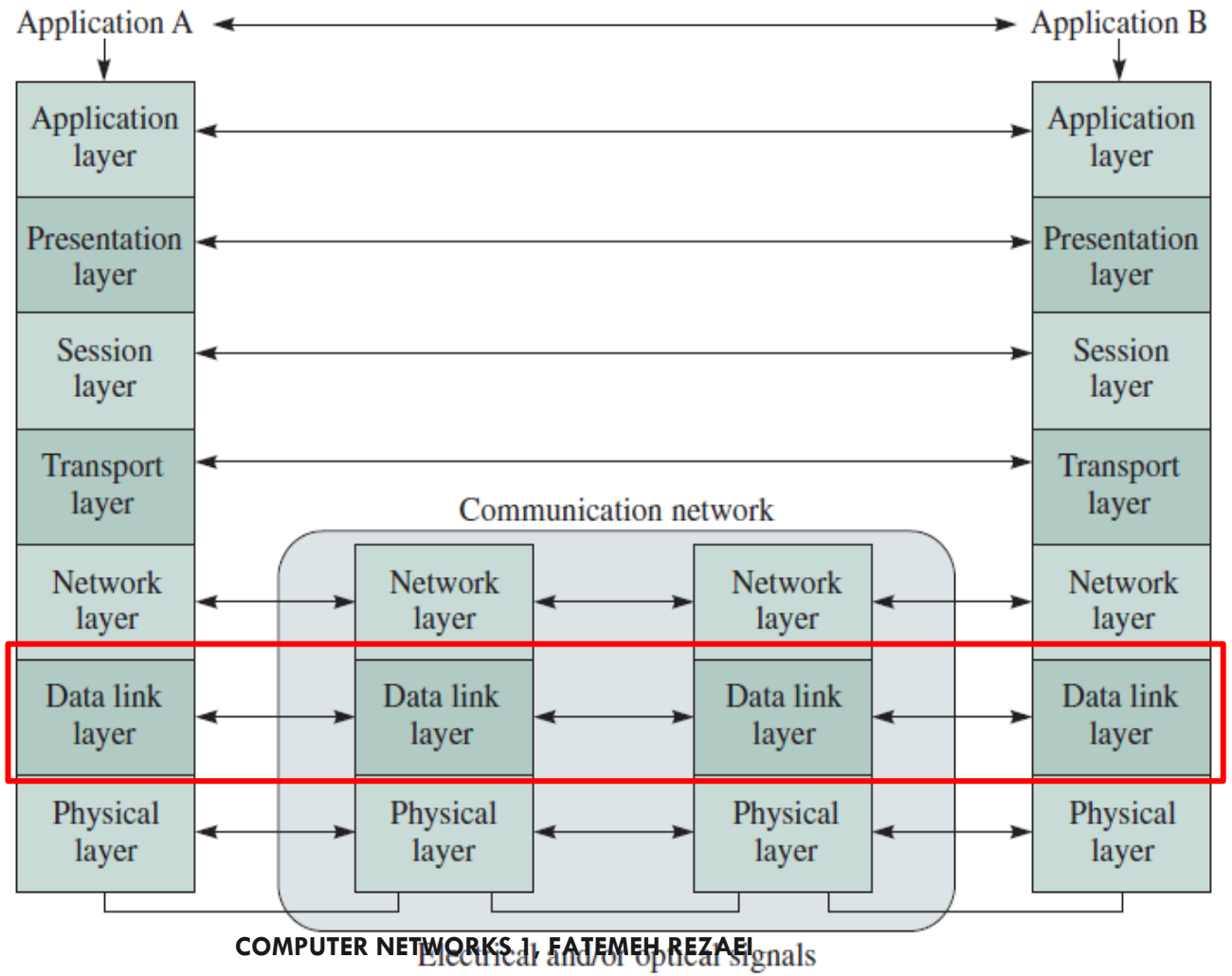
- Introduction
- Framing
- Synchronization
- Error Detection
- Error Correction

K. N. Toosi
University of Technology

OUTLINE

- Introduction
- Framing
- Synchronization
- Error Detection
- Error Correction

OSI REFERENCE MODEL



Packet

Frame

Bit

K. N. Toosi
University of Technology

DATA LINK LAYER FUNCTIONS

- Frame synchronization
  - The beginning and end of each frame must be recognizable
- Flow control
  - Regulating the frame rate between Trans/Rec
    - So that the receiver's buffers do not overflow
- Error control
  - Bit errors correction
- Addressing
  - Specifying the identity of the two stations involved in a transmission on a shared link
- Providing services to network layer

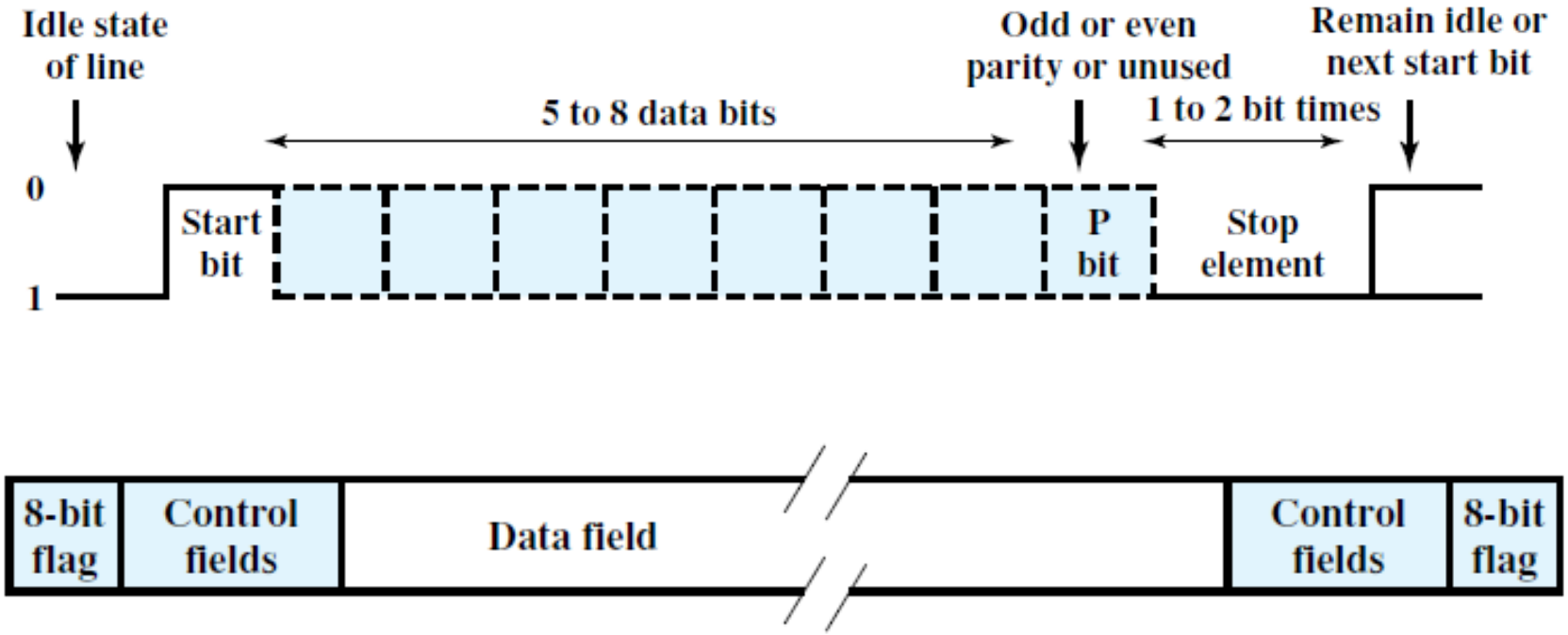
K. N. Toosi
University of Technology

OUTLINE

- Introduction
- **Framing**
- Synchronization
- Error Detection
- Error Correction

FRAMING

- Each block of data is formatted as a frame



DEFINITIONS

- Data are sent in a sequence of frames
  - Frame containing a portion of the data and some control information
- Transmission time
  - The time it takes for a station to emit all of the bits of a frame onto the medium
  - Proportional to the length of the frame
- Propagation time
  - The time it takes for a bit to traverse the link between source and destination

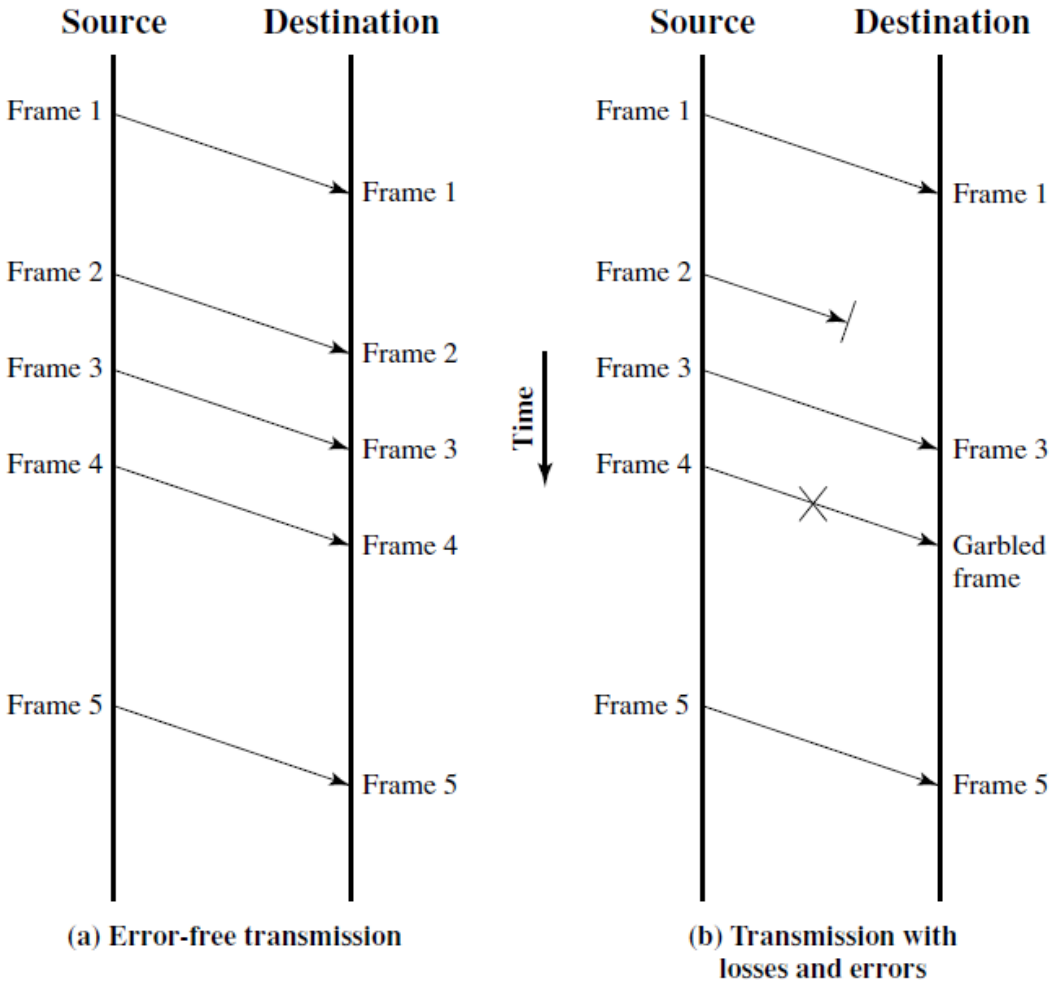
K. N. Toosi
University of Technology

FRAME TRANSMISSION

- Error free transmission
  - All frames that are transmitted are successfully received
  - No frames are lost and none arrive with errors
  - Frames arrive in the same order they are sent
  - Transmitted frame suffers an arbitrary and variable amount of delay before reception
- Transmission with loses and errors

FRAME TRANSMISSION

VERTICAL-TIME SEQUENCE DIAGRAM



(a) Error-free transmission

(b) Transmission with losses and errors

K. N. Toosi
University of Technology

FRAME SIZE

- A source will break up a large block of data into smaller blocks
- Transmitting the data in many frames
- Why?
  - Limited buffer size of the receiver
  - The longer the transmission, the more likely that there will be an error
    - Necessitating retransmission of the entire frame
  - With smaller frames
    - Errors are detected sooner
    - Smaller amount of data needs to be retransmitted
  - On a shared medium, e.g LAN,
    - Desirable not to permit one station to occupy the medium for an extended period
    - Causing long delays at the other sending stations

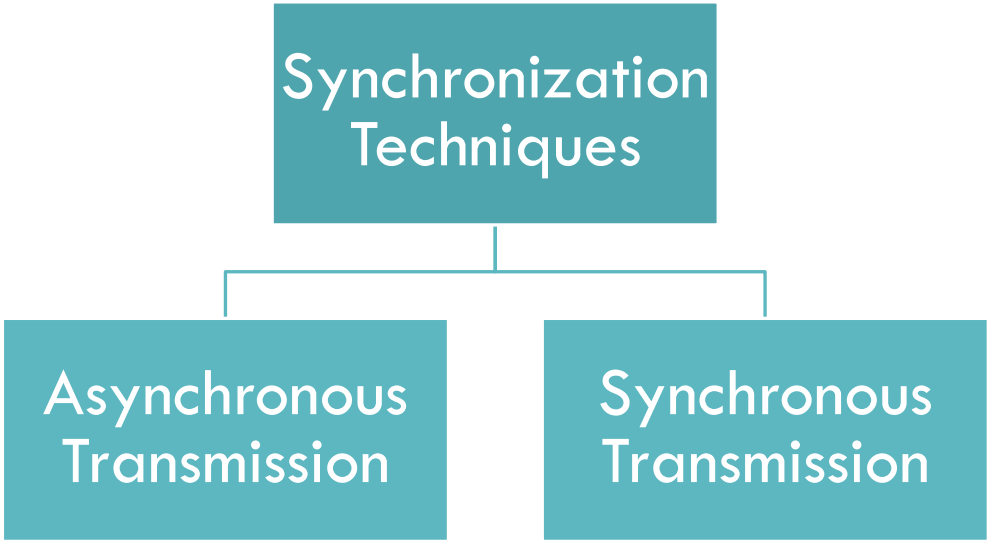
K. N. Toosi
University of Technology

OUTLINE

- Introduction
- Framing
- **Synchronization**
- Error Detection
- Error Correction

SYNCHRONIZATION

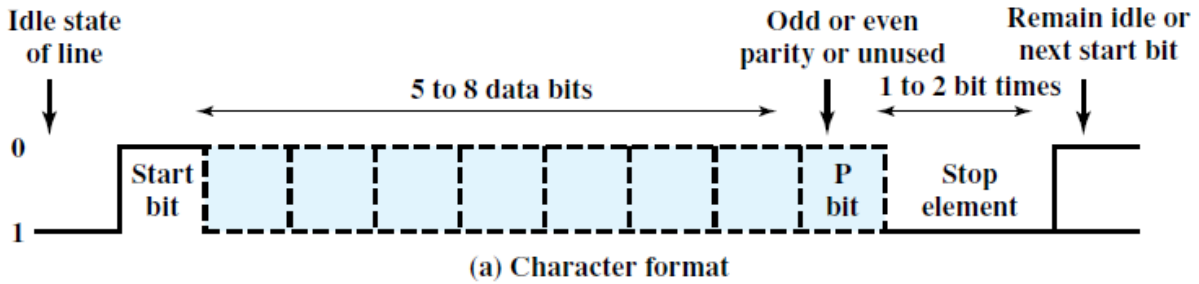
- The receiver must know the bit rate
  - Can sample the line at appropriate intervals
  - To determine the value of each received bit



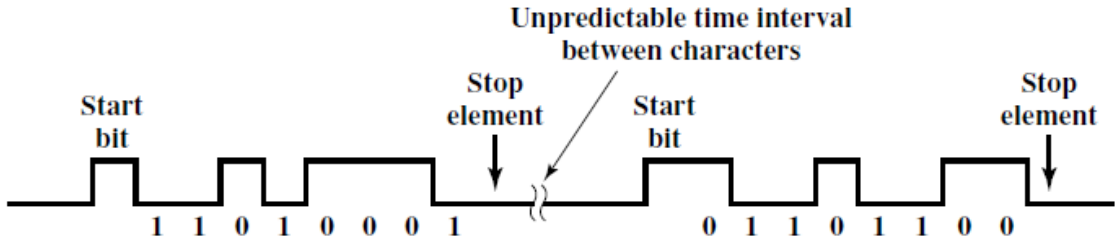
ASYNCHRONOUS TRANSMISSION

- Each character of data is treated independently
- Each character begins with a start bit
  - Alerts the receiver that a character is arriving
- Frame
  - The character plus start bit and stop element
- At the receiver
  - Sampling each bit in the character
  - Looking for the beginning of the next character

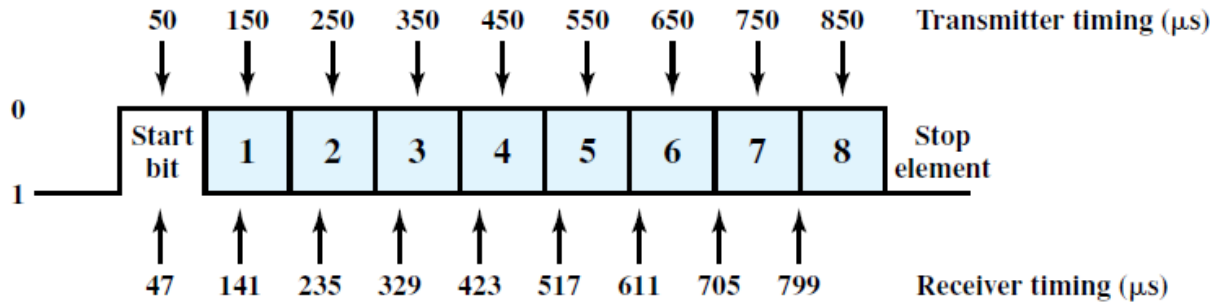
ASYNCHRONOUS TRANSMISSION EXAMPLE



(a) Character format



(b) 8-bit asynchronous character stream



(c) Effect of timing error

ASYNCHRONOUS TRANSMISSION ERRORS

- Cumulative timing error
  - The receiver's clock might eventually drift out of synchronization with the transmitter's clock
- Results in two errors
  - Bit Error
    - The last sampled bit is incorrectly received
  - Framing error
    - The bit count may now be out of alignment
    - If bit 7 is a 1 and bit 8 is a 0, bit 8 could be mistaken for a start bit
    - Can also occur if some noise condition causes the false appearance of a start bit during the idle state

ASYNCHRONOUS TRANSMISSION PROS. AND CONS.

- Simple and Cheap
- Requires an overhead of two to three bits per character
- For an 8-bit character
  - With no parity bit
  - Using a 1-bit-long stop element
  - 2 out of every 10 bits convey no information
  - The overhead is 20%
- The percentage overhead could be reduced
  - by sending larger blocks of bits between the start bit and stop element
- Larger block of bits, Greater Cumulative timing error
- Not work well for long blocks of data

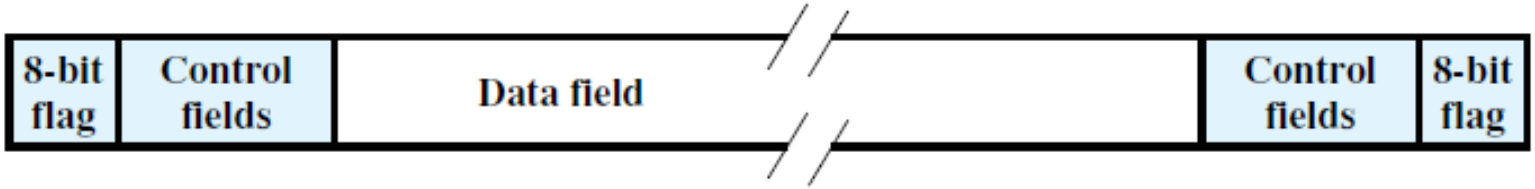
K. N. Toosi
University of Technology

SYNCHRONOUS TRANSMISSION

- Transmitter and receiver clocks must be synchronized
- Separate clock line
  - Transmitter or receiver pulses the line regularly with one short pulse per bit time
  - The other side uses these regular pulses as a clock
  - Working well over short distances
  - Not well over longer distances
    - Timing errors due to channel impairments
- Embedding the clocking information in the data signal
  - Digital signals
    - Manchester/differential Manchester encoding
  - Analog signals
    - Carrier

SYNCHRONOUS TRANSMISSION FRAMING

- Beginning with a preamble bit pattern (called a flag)
- Generally ends with a postamble bit pattern
- Control fields
  - Containing data link control protocol information
- Data field
  - Variable length for most protocols



SYNCHRONOUS TRANSMISSION EFFICIENCY

- Control information, preamble, and postamble in synchronous transmission are typically less than 100 bits
- HDLC (High-Level Data Link Control)
  - Overhead: 48 bits of control, preamble, and postamble
  - For a 1000-character block of data, each frame consists of 48 bits of overhead and 8000 bits of data
  - Percentage overhead:
    - 48/8048 * 100% = 0.6%
- Asynchronous transmission requires 20% or more overhead
- Sending data in large blocks is more efficient than sending data one character at a time
- For sizable blocks of data, synchronous transmission is far more efficient than asynchronous

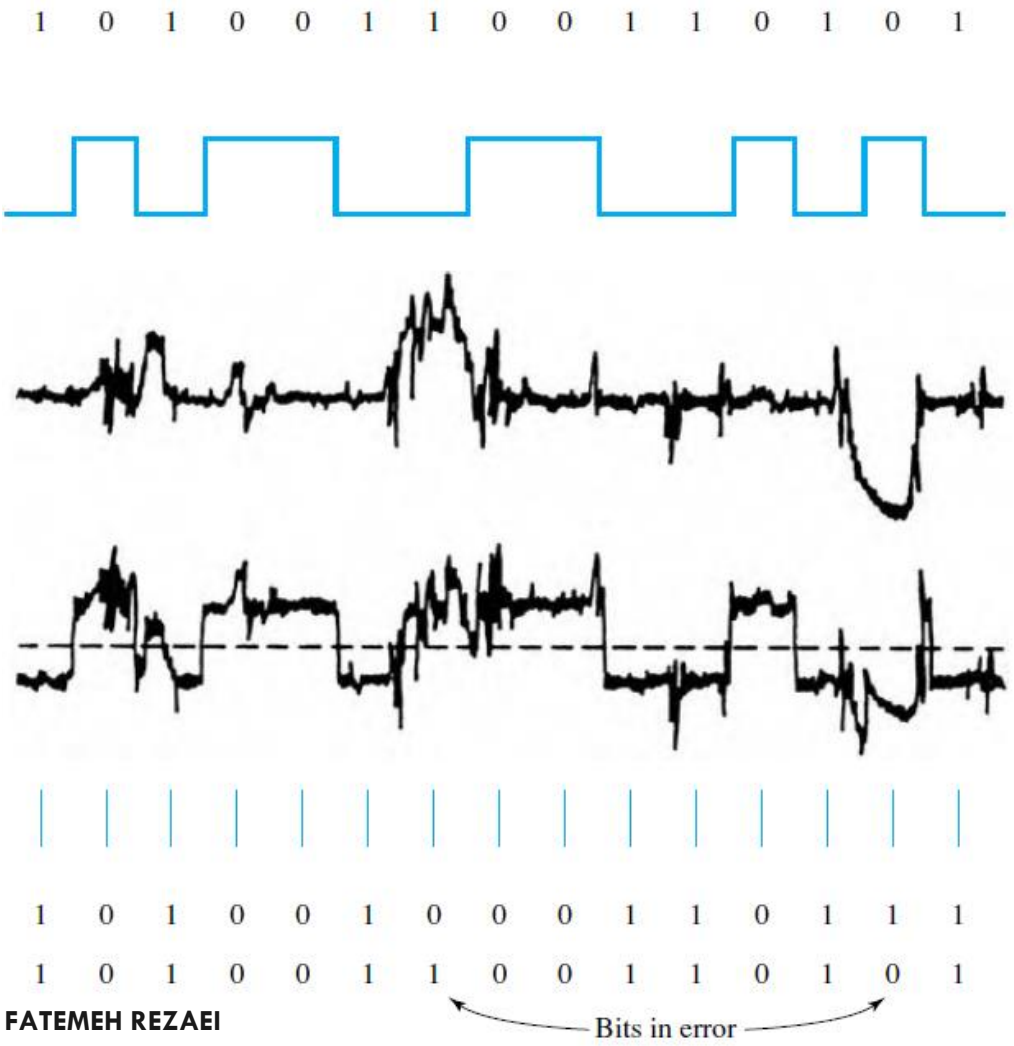
K. N. Toosi
University of Technology

OUTLINE

- Introduction
- Framing
- Synchronization
- **Error Detection**
- Error Correction

TYPES OF ERRORS IN DIGITAL TRANSMISSION

- Error
  - A bit alternation between transmission and reception
- Single-bit error
  - Isolated error condition that alters one bit but does not affect nearby bits
  - Can be caused by white noise

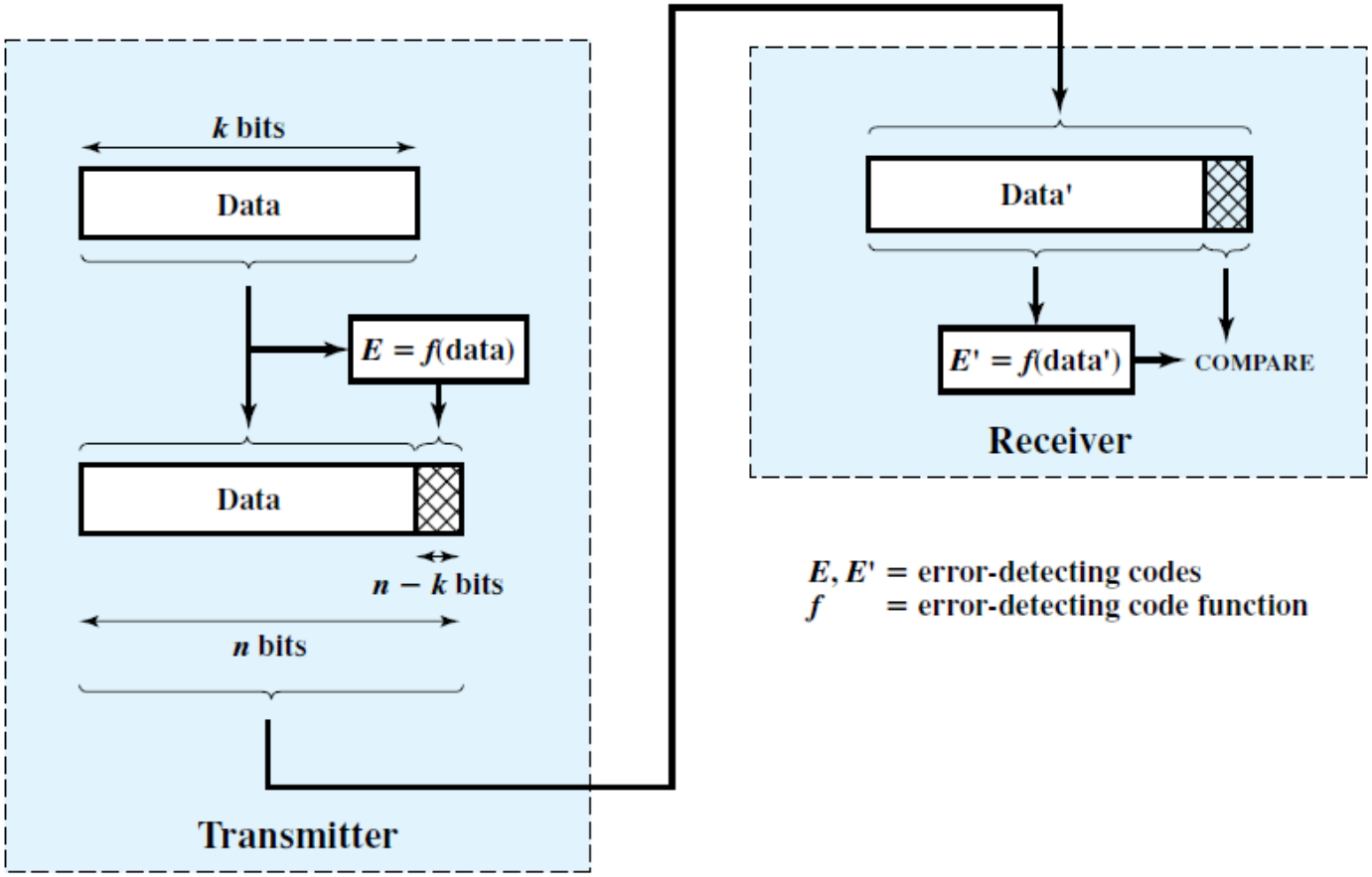


BURST ERRORS

- A cluster of bits in which a number of errors occur
  - Not necessarily all of the bits in the cluster suffer an error
- More common and more difficult to deal with
- Caused by Impulse noise or fading
- Greater effects at higher data rates
- EX: An impulse noise or a fading event of 1 μs
  - At a data rate of 10 Mbps: A resulting error burst of 10 bits
  - At a data rate of 100 Mbps: A resulting error burst of 100 bits

ERROR DETECTION

- Determination of whether errors are present in a received word



SINGLE PARITY CHECK

- The simplest error-detecting scheme
- Appending a parity bit to the end of a block of data
- Ex: Character transmission
  - A parity bit is attached to each 7-bit IRA character
  - The value of this bit is selected so that the character has an even number of 1s (even parity)
- All codewords have even number of 1s
- Receiver checks to see if number of 1s is even

ERROR DETECTION

- An error pattern is undetectable if and only if it causes the received word to be a valid codeword other than that which was transmitted
- Ex.
  - In single-parity-check, error will be undetectable when the number of bits in error is even

EXAMPLE OF SINGLE PARITY CHECK

- Information (7 bits): (0, 1, 0, 1, 1, 0, 0)
- Parity Bit: $b_8 = 0 + 1 + 0 + 1 + 1 + 0 = 1$
- Codeword (8 bits): (0, 1, 0, 1, 1, 0, 0, 1)

- If single error in bit 3 : (0, 1, 1, 1, 1, 0, 0, 1)
  - # of 1's =5, odd
  - Error detected

- If errors in bits 3 and 5: (0, 1, 1, 1, 0, 0, 0, 1)
  - # of 1's =4, even
  - Error not detected

SINGLE PARITY CHECK PERFORMANCE

- Redundancy
  - Single parity check code adds 1 redundant bit per k information bits
  - Overhead $= 1/(k + 1)$
- Coverage
  - All error patterns that change an odd number of bits are detectable
  - All even-numbered patterns are undetectable

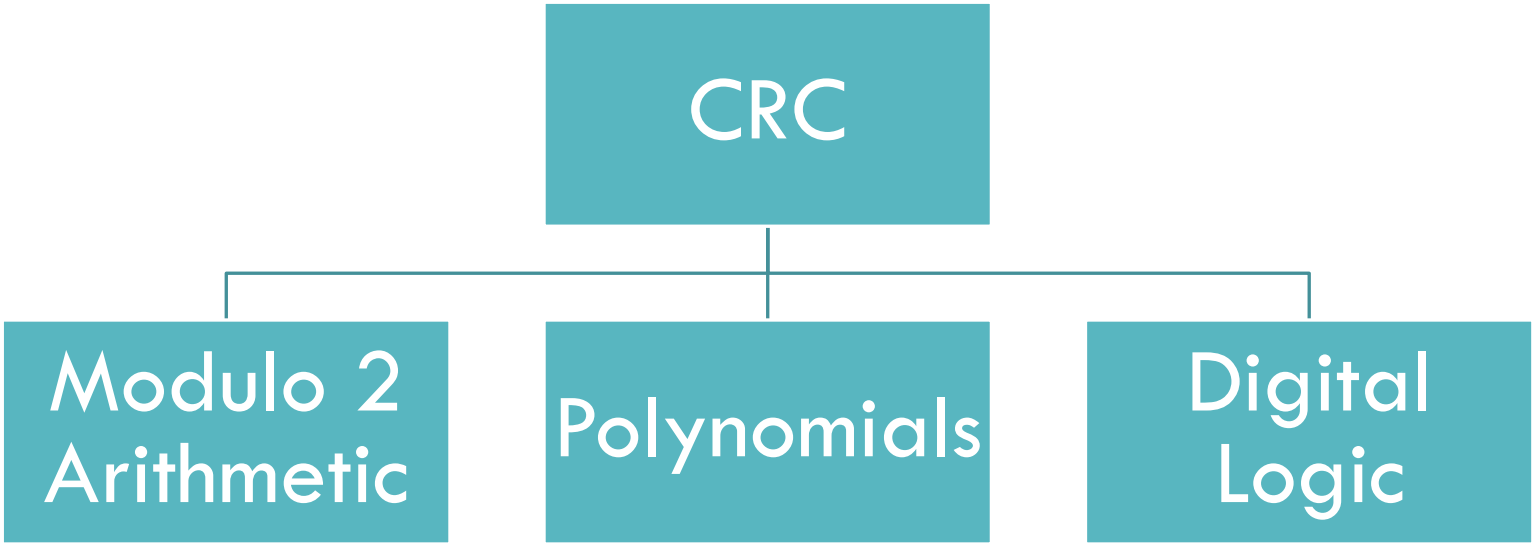
BLOCK CODES

- **(n,k) codes:** n-bit blocks (codewords) are used to conveys k-info-bit block
  - message : k-tuple $\underline{b}=(b_1,b_2,\ldots,b_k)$
  - code word : n-tuple $\underline{c}=(c_1,c_2,\ldots,c_n)$
- Rate: $R = k/n$

(6,3) Binary Block Code

Messages	Code words
(0 0 0)	(0 0 0 0 0 0)
(1 0 0)	(1 1 0 1 0 0)
(0 1 0)	(0 1 1 0 1 0)
(1 1 0)	(1 0 1 1 1 0)
(0 0 1)	(1 1 1 0 0 1)
(1 0 1)	(0 0 1 1 0 1)
(0 1 1)	(1 0 0 0 1 1)
(1 1 1)	(0 1 0 1 1 1)

CYCLIC REDUNDANCY CHECK (CRC)



CYCLIC CODES

- Binary vectors map to polynomials

$$(i_0, i_1, i_2, \ldots, i_{k-2}, i_{k-1}) \rightarrow i_0 + i_1 X + i_2 X^2 + \ldots + i_{k-2} X^{k-2} + i_{k-1} X^{k-1}$$

- Cyclic Shift
  - $\mathbf{i}=(i_0, i_1, i_2, \ldots, i_{n-1})$
  - Cyclic shift of **i**:
  - $\mathbf{i}^{(1)}=(i_{n-1}, i_0, i_1, \ldots, i_{n-2})$
    cyclically shifting the components of **i** *one place to the right*
  - Cyclically shifting **i** *k places to the right*
  - $\mathbf{i}^{(k)}=(i_{n-k}, i_{n-k+1}, \ldots, i_{n-1}, i_0, i_1, \ldots, i_{n-k-1})$
- Cyclic code
  - An (n,k) linear code C is called a cyclic code if every cyclic shift of a codeword in C is also a codeword in C

CRC USING MODULO 2 ARITHMETIC

T = n-bit frame to be transmitted

D = k-bit block of data, or message, the first k bits of T

F = $(n - k)$-bit FCS, the last $(n - k)$ bits of T

P = pattern of $n - k + 1$ bits; this is the predetermined divisor

$$T = 2^{n-k}D + F$$

$$\frac{2^{n-k}D}{P} = Q + \frac{R}{P}$$

$$T = 2^{n-k}D + R$$

$$\frac{T}{P} = \frac{2^{n-k}D + R}{P} = \frac{2^{n-k}D}{P} + \frac{R}{P}$$

$$\frac{T}{P} = Q + \frac{R}{P} + \frac{R}{P}$$

$$\frac{T}{P} = Q + \frac{R + R}{P} = Q$$

ERROR IN CRC

T = transmitted frame

E = error pattern with 1s in positions where errors occur

T_r = received frame

$\oplus$ = bitwise exclusive-OR(XOR)

$$T_r = T \oplus E$$

- If there is an error
  - The receiver will fail to detect the error, if and only if
    - T_r is divisible by P
    - Equivalent to E divisible by P

K. N. Toosi
University of Technology

POLYNOMIAL CODES

- Polynomials instead of vectors for codewords
- Polynomial arithmetic instead of check sums
- Implemented using shift-register circuits
- Most data communications standards use polynomial codes for error detection (e.g. Ethernet)
- Basis for powerful error-correction methods

BINARY POLYNOMIAL ARITHMETIC

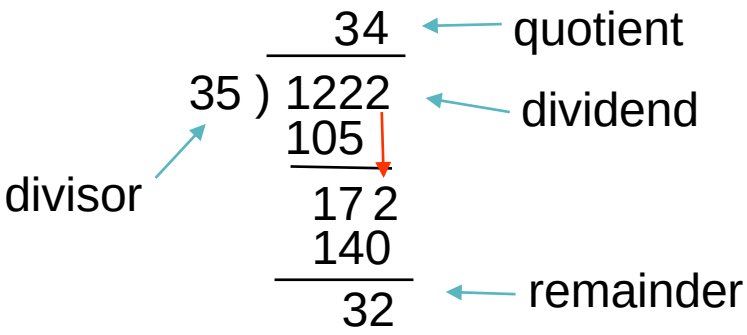
Addition:

$$(x^7 + x^6 + 1) + (x^6 + x^5) = x^7 + x^6 + x^6 + x^5 + 1$$
$$= x^7 + (1+1)x^6 + x^5 + 1$$
$$= x^7 + x^5 + 1 \quad \text{since } 1+1=0 \bmod 2$$

Multiplication:

$$(x+1)(x^2 + x + 1) = x(x^2 + x + 1) + 1(x^2 + x + 1)$$
$$= (x^3 + x^2 + x) + (x^2 + x + 1)$$
$$= x^3 + 1$$

BINARY POLYNOMIAL DIVISION



dividend = quotient x divisor +remainder

$1222 = 34 \times 35 + 32$

- Polynomial Division

$$x^3 + x^2 + x = q(x) \text{ quotient}$$

$$x^3 + x + 1 \;)\; x^6 + x^5$$

divisor

dividend

$$x^6 + x^4 + x^3$$

$$x^5 + x^4 + x^3$$

$$x^5 + x^3 + x^2$$

$$x^4 + x^2$$

$$x^4 + x^2 + x$$

$$x = r(x) \text{ remainder}$$

Note: Degree of r(x) is less than degree of divisor

POLYNOMIAL CODING

- k information bits define polynomial of degree $k - 1$

$$i(x) = i_{k-1}x^{k-1} + i_{k-2}x^{k-2} + \ldots + i_2x^2 + i_1x + i_0$$

- To construct n bit codeword
  - Codeword polynomial c(x) of degree n – 1
  - Code has binary generating polynomial g(x) of degree n–k

$$g(x) = x^{n-k} + g_{n-k-1}x^{n-k-1} + \ldots + g_2x^2 + g_1x + 1$$

$\vec{g}$: Frame check sequence (FCS)

POLYNOMIAL CODING

- Steps:

1. Multiply $i(x)$ by x^{n-k}

(puts zeros in $(n-k)$ low-order positions)

2. Divide $x^{n-k}\ i(x)$ by $g(x)$

Quotient Remainder

$$x^{n-k}\ i(x) = g(x)q(x) + r(x)$$

3. Add remainder $r(x)$ to $x^{n-k}\ i(x)$

(puts check bits in the $n-k$ low-order positions)

$$\underbrace{c(x)}_{n \text{ bits}} = \underbrace{x^{n-k}}_{k \text{ bits}}\, i(x) + \underbrace{r(x)}_{n\text{-}k \text{ bits}}$$

POLYNOMIAL CODING

- Finding *remainder polynomial* of at most degree $n-k-1$

$$g(x) \;\overline{)\; x^{n-k}\, i(x)} \quad \begin{matrix} q(x) \\ \hline \\ r(x) \end{matrix}$$

$$x^{n-k} i(x) = q(x)g(x) + r(x)$$

- *Codeword polynomial* of degree $n-1$

$$c(x) = x^{n-k} i(x) + r(x)$$

$$= q(x)g(x) + r(x) + r(x) = q(x)g(x)$$

POLYNOMIAL CODING

- All codewords satisfy the following pattern:

$$c(x) = x^{n-k}i(x) + r(x) = q(x)g(x) + r(x) + r(x) = q(x)g(x)$$

- All codewords are a multiple of *g(x)*
- Receiver should divide received n-tuple by g(x) and check if remainder is zero
- If remainder is nonzero, then received n-tuple is not a codeword

CRC EXAMPLE

- Generator polynomial: $g(x) = x^3 + x + 1$ $\quad n–k = 3$
- Information: $(1,1,0,0) \rightarrow i(x) = x^3 + x^2$ $\quad k = 4$
- Encoding: $x^3 i(x) = x^6 + x^5$

$$
\begin{array}{r|l}
 & x^3+x^2+x \\
\hline
x^3+x+1 & x^6+x^5 \\
 & x^6+\quad\ x^4+x^3 \\
\hline
 & \quad x^5+x^4+x^3 \\
 & \quad x^5+\quad\ x^3+x^2 \\
\hline
 & \qquad\quad x^4+\quad x^2 \\
 & \qquad\quad x^4+\quad x^2+x \\
\hline
 & \qquad\qquad\qquad\ x
\end{array}
$$

$$
\begin{array}{r|l}
 & 1110 \\
\hline
1011 & 1100000 \\
 & 1011 \\
\hline
 & \quad 1110 \\
 & \quad 1011 \\
\hline
 & \qquad 1010 \\
 & \qquad 1011 \\
\hline
 & \qquad\quad 010
\end{array}
$$

- Transmitted codeword: $c(x) = x^6 + x^5 + x$ $\quad \underline{c} = (1,1,0,0,0,1,0)$

WIDELY USED VERSIONS OF THE GENERATING POLYNOMIAL

$$\text{CRC-12} = X^{12} + X^{11} + X^3 + X^2 + X + 1$$

$$\text{CRC-16} = X^{16} + X^{15} + X^2 + 1$$

$$\text{CRC-CCITT} = X^{16} + X^{12} + X^5 + 1$$

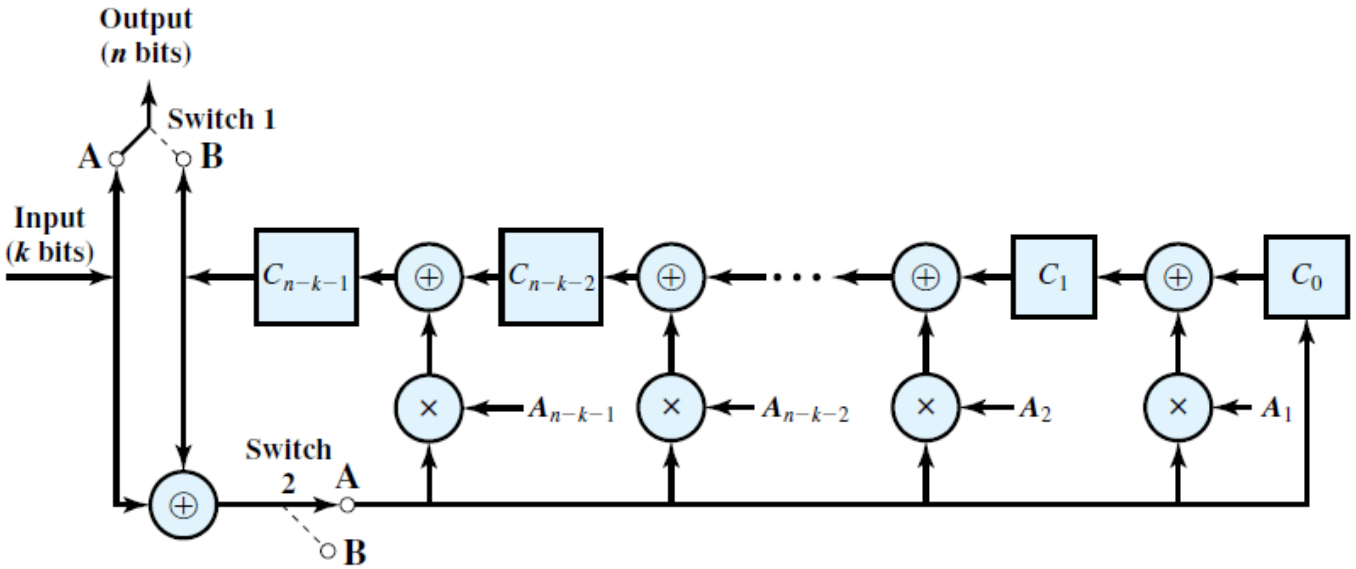
$$\text{CRC-32} = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$

IEEE 802 LAN standards

SHIFT-REGISTER IMPLEMENTATION

1. The register contains $n - k$ bits, equal to the length of the FCS.
2. There are up to $n - k$ XOR gates.
3. The presence or absence of a gate corresponds to the presence or absence of a term in the divisor polynomial, $P(X)$, excluding the terms 1 and X^{n-k}.

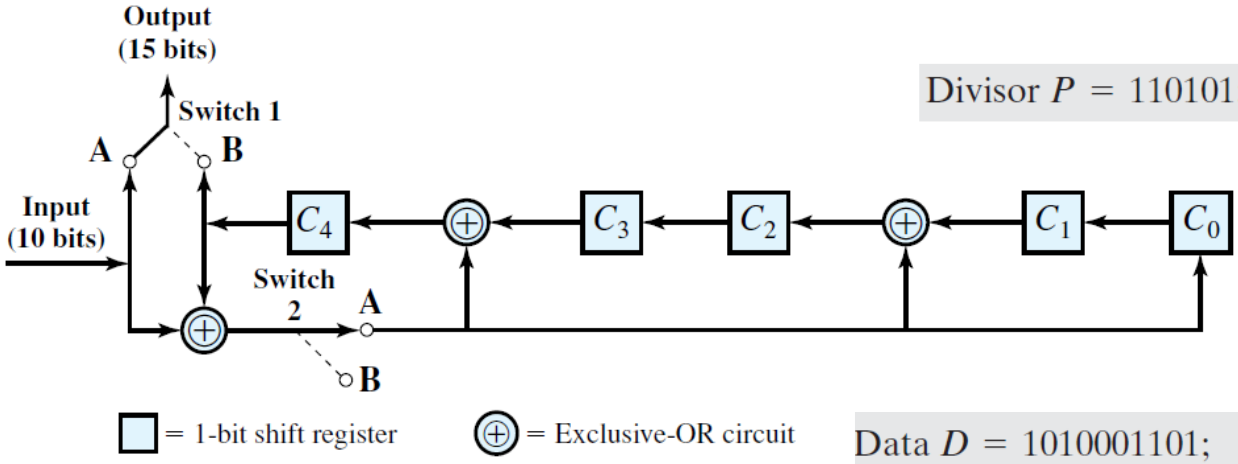
$$1 + A_1X + A_2X^2 + \cdots + A_{n-1}X^{n-k-1} + X^{n-k}$$



SHIFT-REGISTER IMPLEMENTATION

1. Accept information bits $i_{k-1}, i_{k-2}, \ldots, i_2, i_1, i_0$
2. Append $n - k$ zeros to information bits
3. Feed sequence to shift-register circuit that performs polynomial division
4. After n shifts, the shift register contains the remainder

CRC ENCODING EXAMPLE



(a) Shift-register implementation

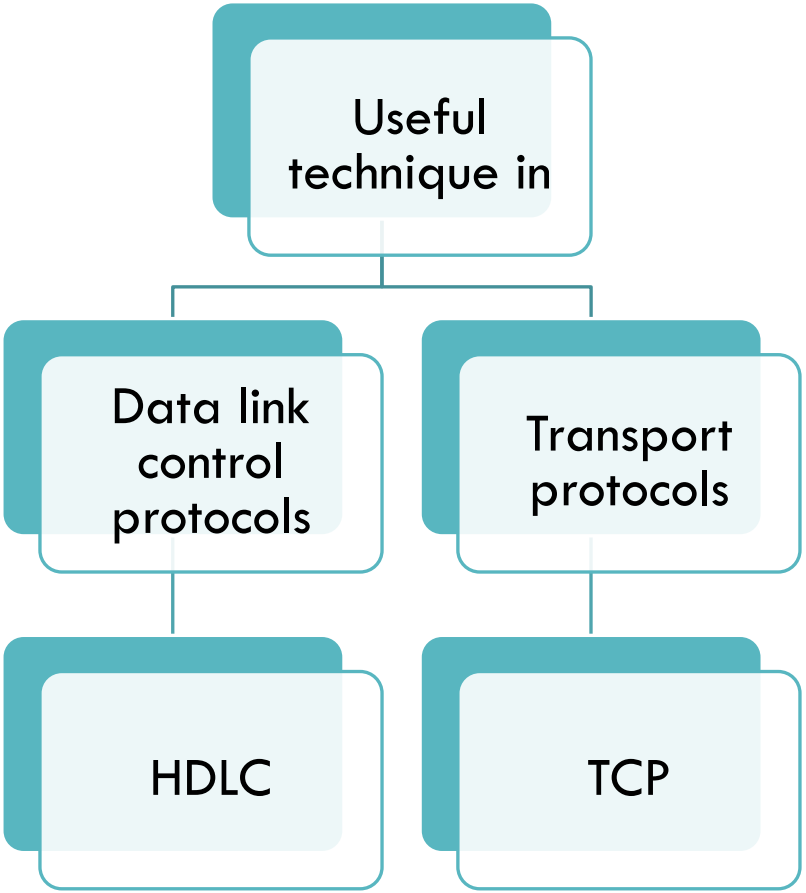
Divisor $P = 110101$; $P(X) = X^5 + X^4 + X^2 + 1$

Data $D = 1010001101$; $D(X) = X^9 + X^7 + X^3 + X^2 + 1$

	C_4	C_3	C_2	C_1	C_0	$C_4 \oplus C_3 \oplus I$	$C_4 \oplus C_1 \oplus I$	$C_4 \oplus I$	I = input	
Initial	0	0	0	0	0	1	1	1	1	
Step 1	1	0	1	0	1	1	1	1	0	
Step 2	1	1	1	1	1	1	1	0	1	
Step 3	1	1	1	1	0	0	0	1	0	
Step 4	0	1	0	0	1	1	0	0	0	
Step 5	1	0	0	1	0	1	0	1	0	Message to be sent
Step 6	1	0	0	0	1	0	0	0	1	
Step 7	0	0	0	1	0	1	0	1	1	
Step 8	1	0	0	0	1	1	1	1	0	
Step 9	1	0	1	1	1	0	1	0	1	
Step 10	0	1	1	1	0					

(b) Example with input of 1010001101

ERROR DETECTION APPLICATIONS



CORRECTING OF ERRORS USING ERROR DETECTION CODES

- Requiring retransmission of data block
- Inadequate for wireless applications
  - High bit error rate on a wireless link
    - Large number of retransmissions
  - Long propagation delay compared to the transmission time
    - Inefficient system
    - Ex. satellite links
- The common approach
  - Retransmission of the frame in error plus all subsequent frames
    - Error in a single frame necessitates retransmitting many frames

K. N. Toosi
University of Technology

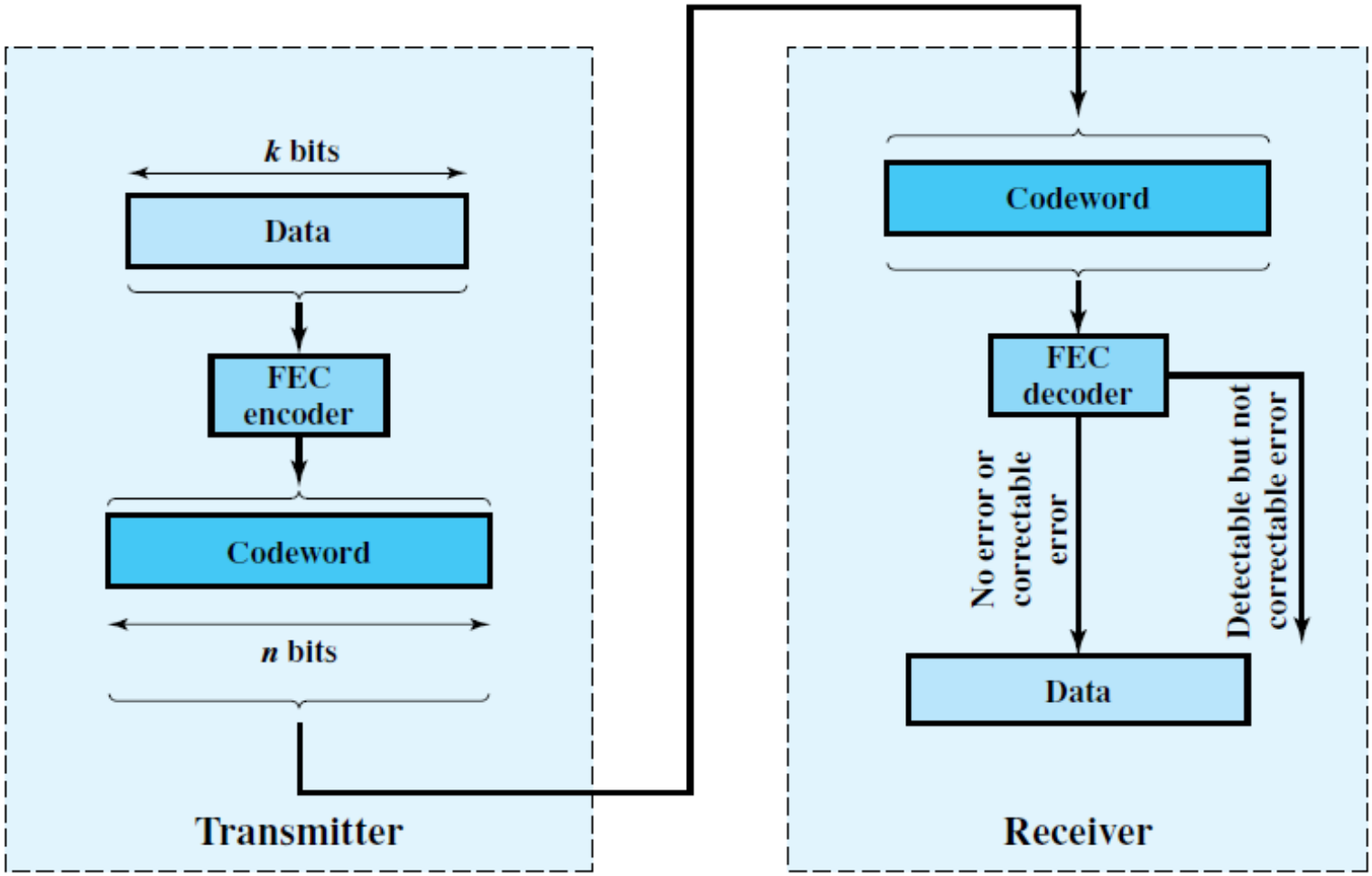
OUTLINE

- Introduction
- Framing
- Synchronization
- Error Detection
- **Error Correction**

ERROR CORRECTION

- Enabling the receiver to correct errors on the basis of the transmitted bits
- By adding redundancy to the transmitted message
- Similar layout as shown for error-detecting codes
  - Mapping the *k*-bit input into an *n*-bit codeword
- FEC (forward error correction) algorithms
  - Original *k* bits not appear in the codeword

ERROR CORRECTION BLOCK DIAGRAM



K. N. Toosi
University of Technology

POSSIBLE OUTCOMES OF FEC DECODER

1. No bit errors

- Input to the FEC decoder is identical to the original codeword
- Decoder produces the original data block

2. Correctable error patterns

- Detect and correct those errors
- FEC decoder is able to map the incoming codeword into the original data block

3. Detectable but not correctable error patterns

- Decoder can detect but not correct the errors
- Decoder reports an uncorrectable error

4. Not detectable error patterns

- Decoder does not detect that any errors have occurred
- Decoder maps the incoming codeword into a wrong data block

BLOCK CODE TECHNIQUE FOR ERROR CORRECTION

- An (*n*, *k*) block code encodes *k* data bits into *n*-bit codewords
  - Adding *n*-*k* redundant bits
- Each valid codeword reproduces the original *k* data bits
- Goal: To design a function

$$v_c = f(v_d)$$

v_d: Vector of *k* data bits

v_c: Vector of *n* codeword bits

- Redundancy of the code: $\frac{n-k}{k}$
  - Ratio of redundant bits to data bits

CODE RATE

- Ratio of data bits to total bits $\frac{k}{n}$
  - How much additional bandwidth is required
  - To carry data at the same data rate as without the code
- Ex. A code rate of 1/2
  - Requires double the transmission capacity of an uncoded system
    - To maintain the same data rate
  - If the data rate input to the encoder is 1 Mbps
    - The output from the encoder must be at a rate of 2 Mbps to keep up

HAMMING DISTANCE

- v_1, v_2: n-bit binary sequences
- Hamming distance
  - The number of bits in which v_1 and v_2 disagree
  - $d(v_1, v_2)$
  - Ex.

v_1 =(1 , 0 , 0 , 0 , 0 , 1 , 1)

v_2 =(0 , 1 , 0 , 0 , 1 , 0 , 1)

$$d(v_1, v_2) = 4$$

BLOCK CODE PERFORMANCE

- (n, k) block code
  - $s = 2^n$ possible codewords $w_1, w_2, \ldots, w_s$
  - 2^k valid codewords
- The minimum distance of the code

$$d_{\min} = \min_{i \neq j} [d(\mathbf{w}_i, \mathbf{w}_j)]$$

- For a code with minimum distance $d_{\min}$
  - Number of errors that can be detected per codeword:

$$d\text{min}-1$$

  - Maximum number of guaranteed correctable errors per codeword:

$$\left\lfloor \frac{d\text{min}-1}{2} \right\rfloor$$

K. N. Toosi
University of Technology

BLOCK CODE DESIGN CONCERNS

1. The largest possible value of d_{min}
   - More correctable errors
2. Relatively easy to encode and decode
   - Minimal memory and processing time
3. Small number of extra bits (*n-k*)
   - Reducing bandwidth
4. Large number of extra bits (*n-k*)
   - Reducing error rate

CODING PERFORMANCE

- E_b/N_0
  - Ratio of signal energy per bit to noise power density per Hertz
- Coding gain
  - The reduction, in decibels, in the required E_b/N_0
  - To achieve a specified BER of an error-correcting coded system
  - Compared to an uncoded system using the same modulation

IMPROVING SYSTEM PERFORMANCE BY CODING