ADVANCED COMPUTER NETWORKS

Prof. Dr. Hasan Hüseyin BALIK

(3^{rd} Week)

Outline

- 2. Data Communications
  - 2.1. Error Detection and Correction
  - 2.2. Data Link Control Protocols
  - 2.3. Multiplexing

2.1 Error Detection and Correction

2.1.Outline

- Types of Errors
- Error Detection
- Parity Check
- The Internet Checksum
- Cyclic Redundancy Check (CRC)
- Forward Error Correction

Types of Errors

- An error occurs when a bit is altered between transmission and reception
  - Binary 1 is transmitted and binary 0 is received
  - Binary 0 is transmitted and binary 1 is received

Single bit errors

Isolated error that alters one bit but does not affect nearby bits

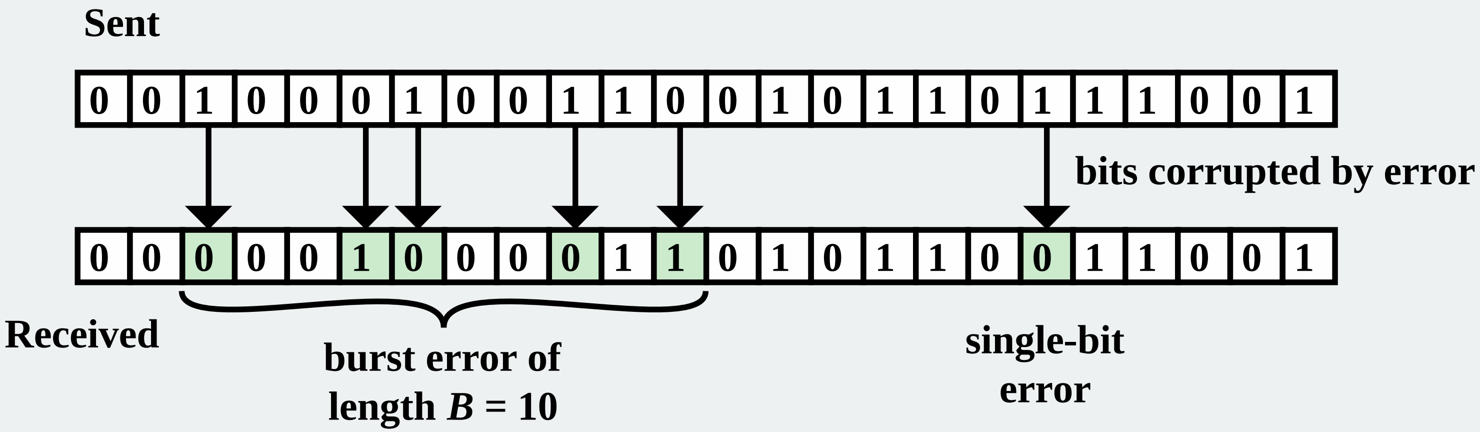
Can occur in the presence of white noise

Burst errors

Contiguous sequence of *B* bits in which the first and last bits and any number of intermediate bits are received in error

Can be caused by impulse noise or by fading in a mobile wireless environment

Effects of burst errors are greater at higher data rates



Sent: 0 0 1 0 0 0 1 0 0 1 1 0 0 1 0 1 1 0 1 1 1 0 0 1

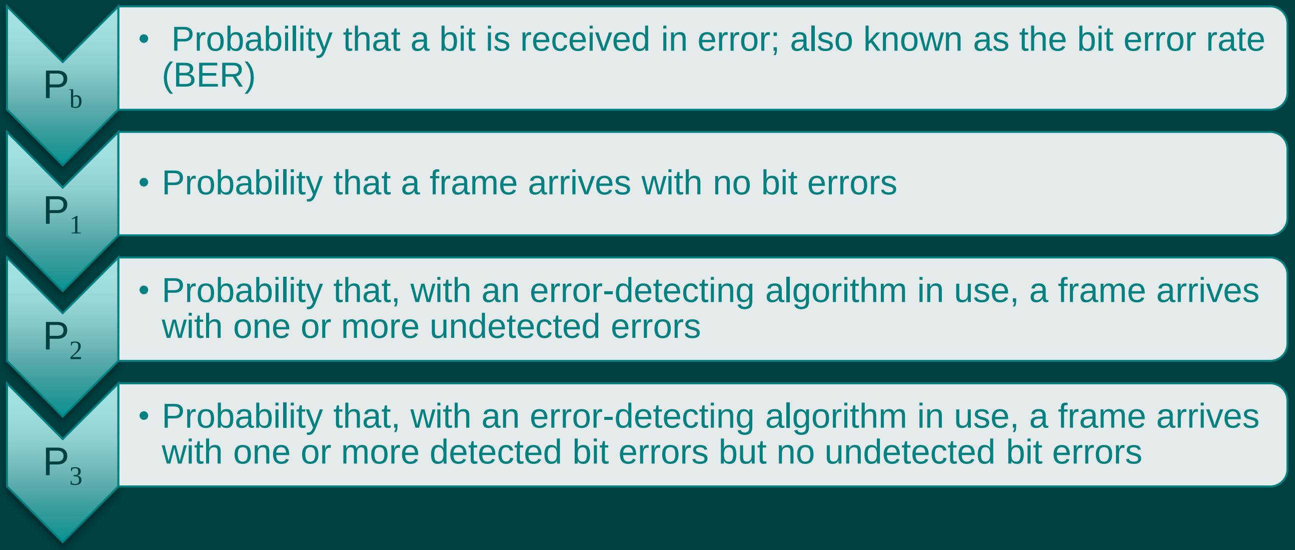
Received: 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 1 1 0 0 1 1 0 0 1

Figure 6.1 Burst and Single-Bit Errors

Error Detection

- Regardless of design you will have errors, resulting in the change of one or more bits in a transmitted frame

- Frames
  - Data transmitted as one or more contiguous sequences of bits



P_b
- Probability that a bit is received in error; also known as the bit error rate (BER)

P_1
- Probability that a frame arrives with no bit errors

P_2
- Probability that, with an error-detecting algorithm in use, a frame arrives with one or more undetected errors

P_3
- Probability that, with an error-detecting algorithm in use, a frame arrives with one or more detected bit errors but no undetected bit errors

- The probability that a frame arrives with no bit errors decreases when the probability of a single bit error increases

- The probability that a frame arrives with no bit errors decreases with increasing frame length
  - The longer the frame, the more bits it has and the higher the probability that one of these is in error

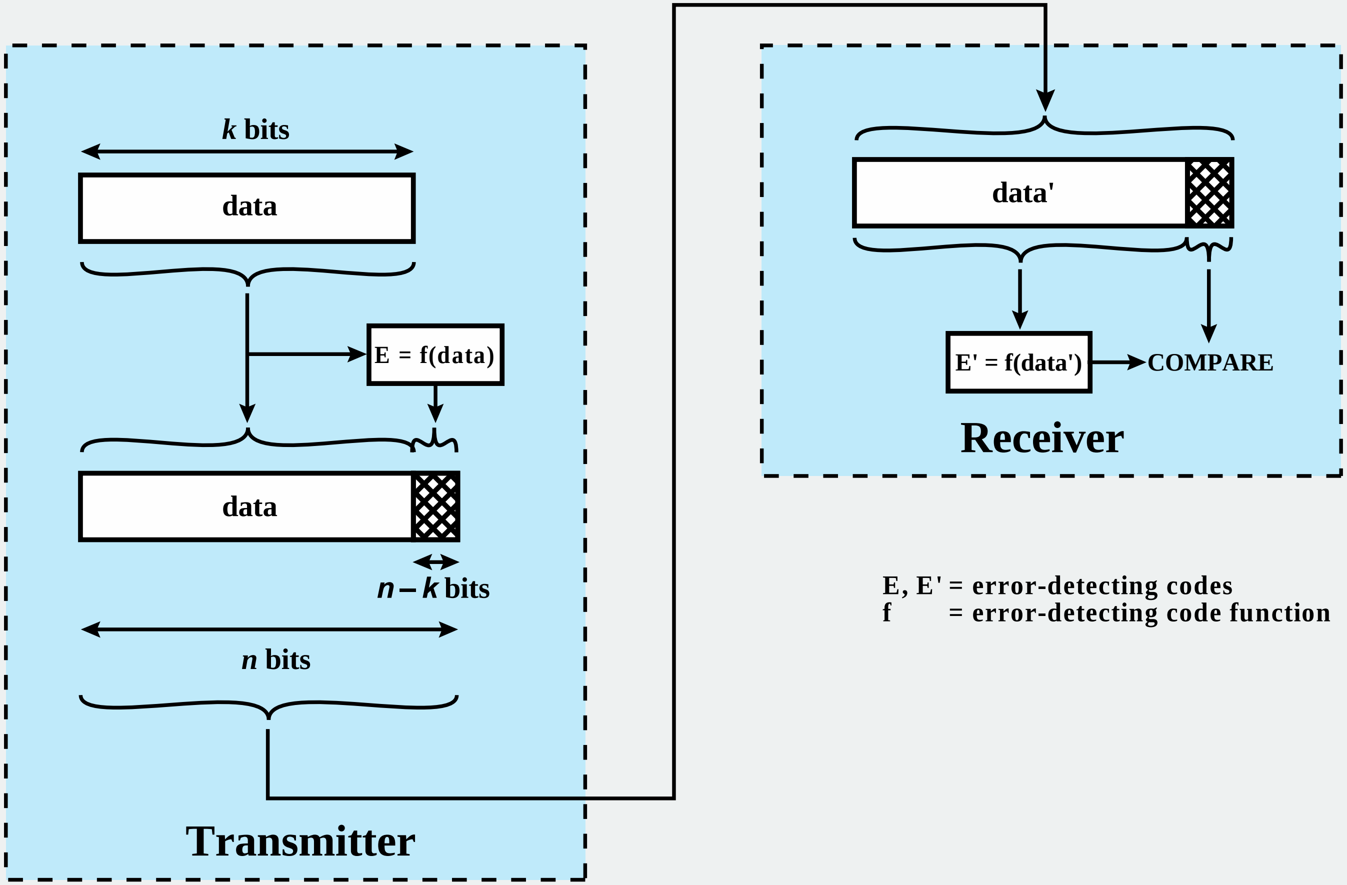
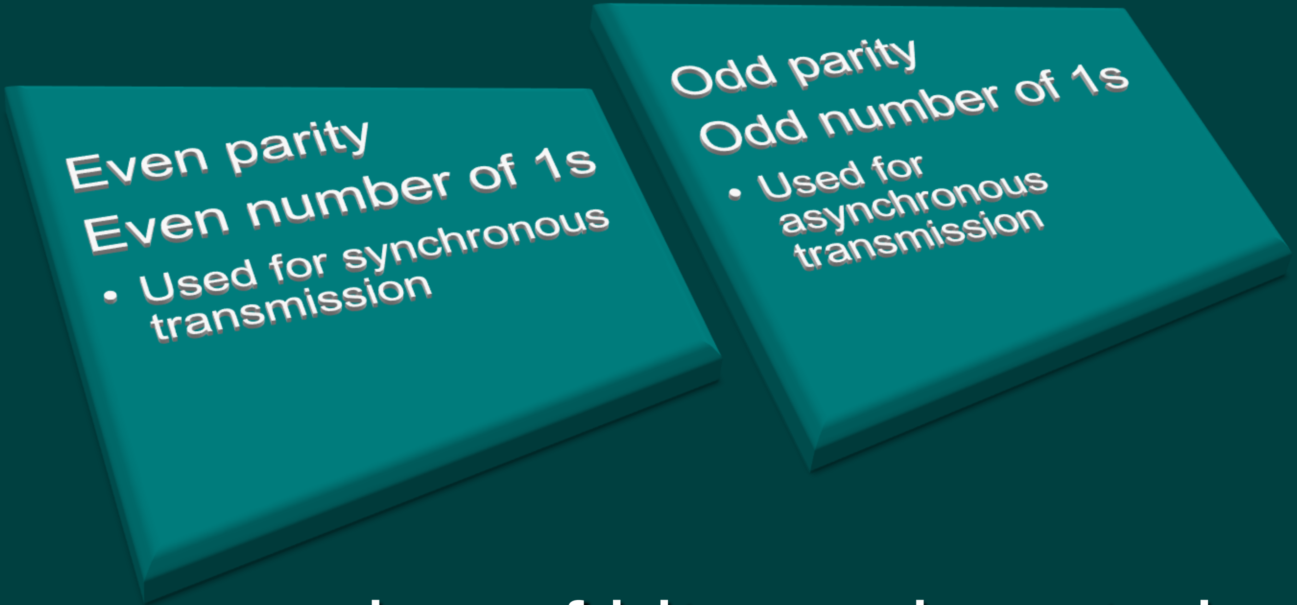


Figure 6.2 Error Detection Process

Parity Check

- The simplest error detecting scheme is to append a parity bit to the end of a block of data



Even parity

Even number of 1s

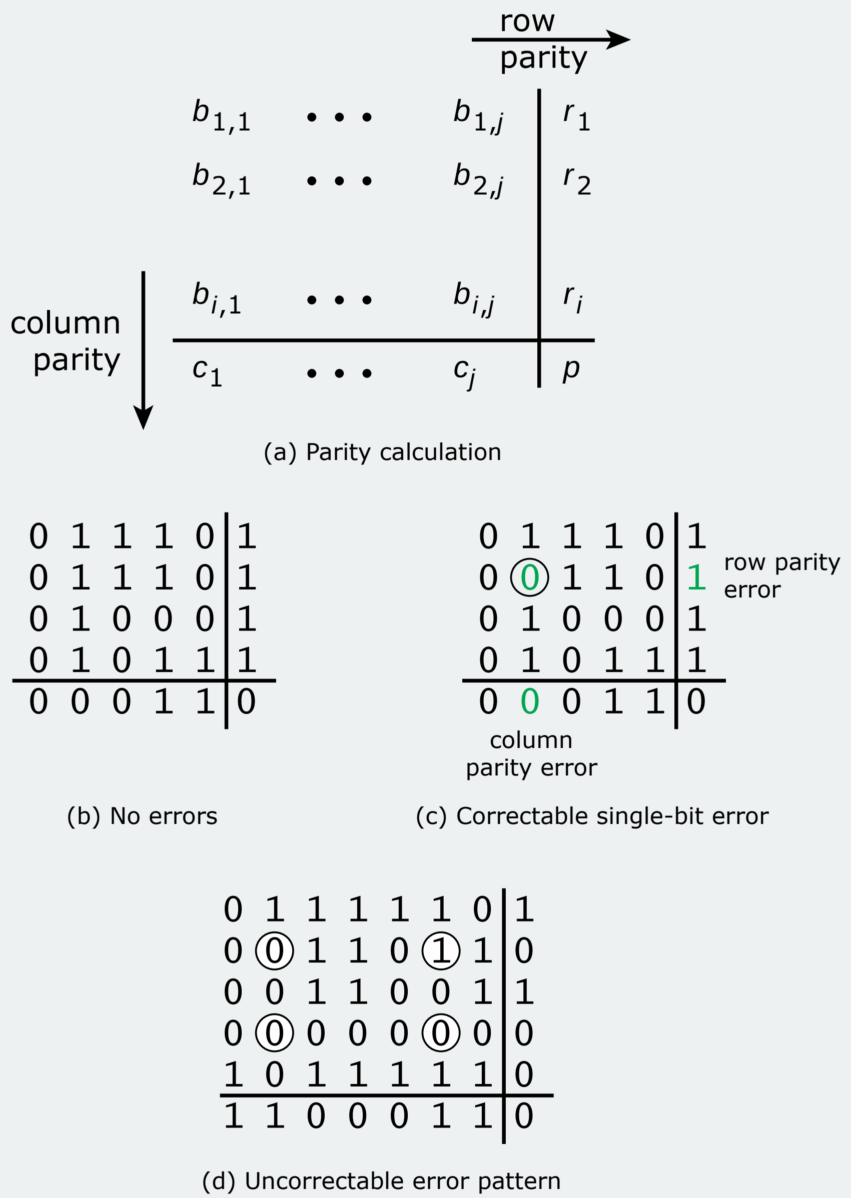
- Used for synchronous transmission

Odd parity

Odd number of 1s

- Used for asynchronous transmission

- If any even number of bits are inverted due to error, an undetected error occurs


row parity
column parity
$b_{1,1}$ • • • $b_{1,j}$ | r_1
$b_{2,1}$ • • • $b_{2,j}$ | r_2
$b_{i,1}$ • • • $b_{i,j}$ | r_i
c_1 • • • c_j | p
row parity error
column parity error

(a) Parity calculation

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

(b) No errors

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

(c) Correctable single-bit error

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

(d) Uncorrectable error pattern

Figure 6.3 A Two-Dimensional Even Parity Scheme

The Internet Checksum

- Error detecting code used in many Internet standard protocols, including IP, TCP, and UDP
- Ones-complement operation
  - Replace 0 digits with 1 digits and 1 digits with 0 digits
- Ones-complement addition
  - The two numbers are treated as unsigned binary integers and added
  - If there is a carry out of the leftmost bit, add 1 to the sum (end-around carry)
- To verify
  - the ones-complement sum is computed over the same set of octets, including the checksum field. If the result is all 1 bits (- 0 in ones-complement arithmetic), the check succeeds.

Partial sum	0001 F203 F204
Partial sum	F204 F4F5 1E6F9
Carry	E6F9 1 E6FA
Partial sum	E6FA F6F7 1DDF1
Carry	DDF1 1 DDF2
Ones complement of the result	220D

(a) Checksum calculation by sender

Partial sum	0001 F203 F204
Partial sum	F204 F4F5 1E6F9
Carry	E6F9 1 E6FA
Partial sum	E6FA F6F7 1DDF1
Carry	DDF1 1 DDF2
Partial sum	DDF2 220D FFFF

(b) Checksum verification by receiver

Figure 6.4 Example of Internet Checksum

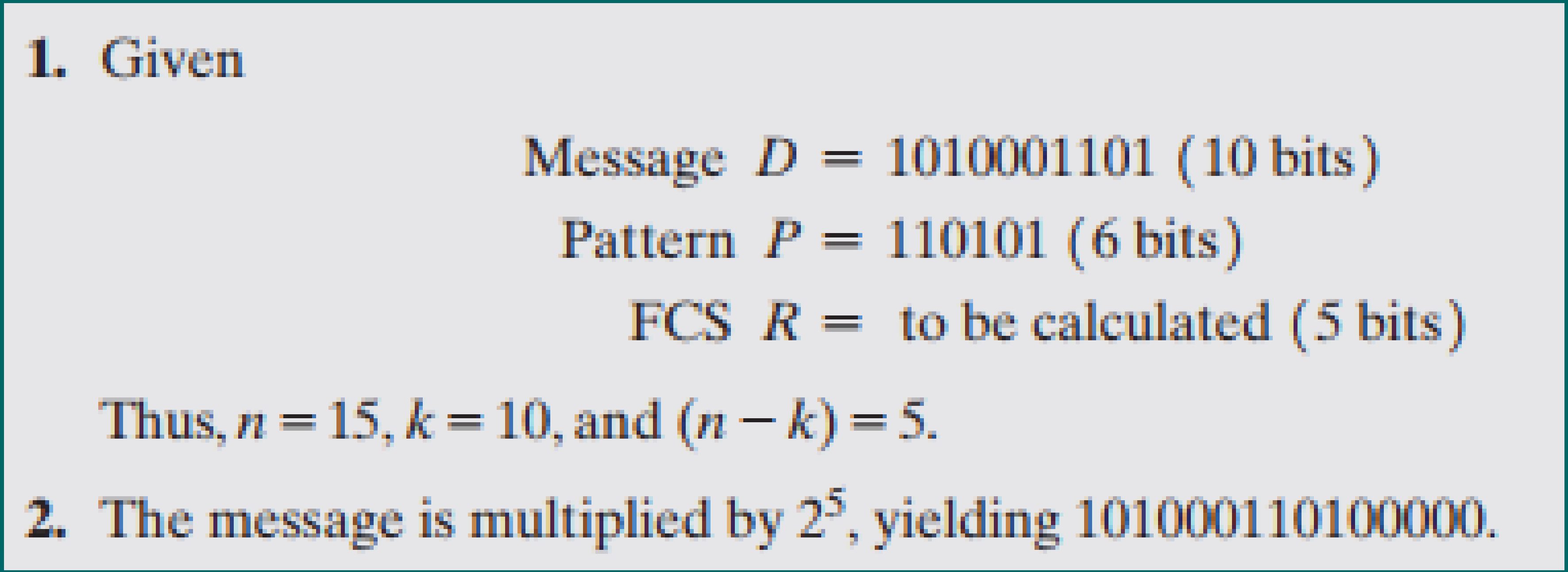
Cyclic Redundancy Check (CRC)

- One of the most common and powerful error-detecting codes
- Given a *k* bit block of bits, the transmitter generates an (*n* – *k*) bit frame check sequence (FCS) which is exactly divisible by some predetermined number
- Receiver divides the incoming frame by that number
  - If there is no remainder, assume there is no error

CRC Process

- Modulo 2 arithmetic
  - Uses binary addition with no carries
  - Just the exclusiveOR (XOR) operation
- Polynomials
  - Express all values as polynomials in a dummy variable X, with binary coefficients
  - Coefficients correspond to the bits in the binary number

- Digital logic
  - Dividing circuit consisting of XOR gates and a shift register
  - Shift register is a string of 1-bit storage devices
  - Each device has an output line, which indicates the value currently stored, and an input line
  - At discrete time instants, known as clock times, the value in the storage device is replaced by the value indicated by its input line
  - The entire register is clocked simultaneously, causing a 1-bit shift along the entire register

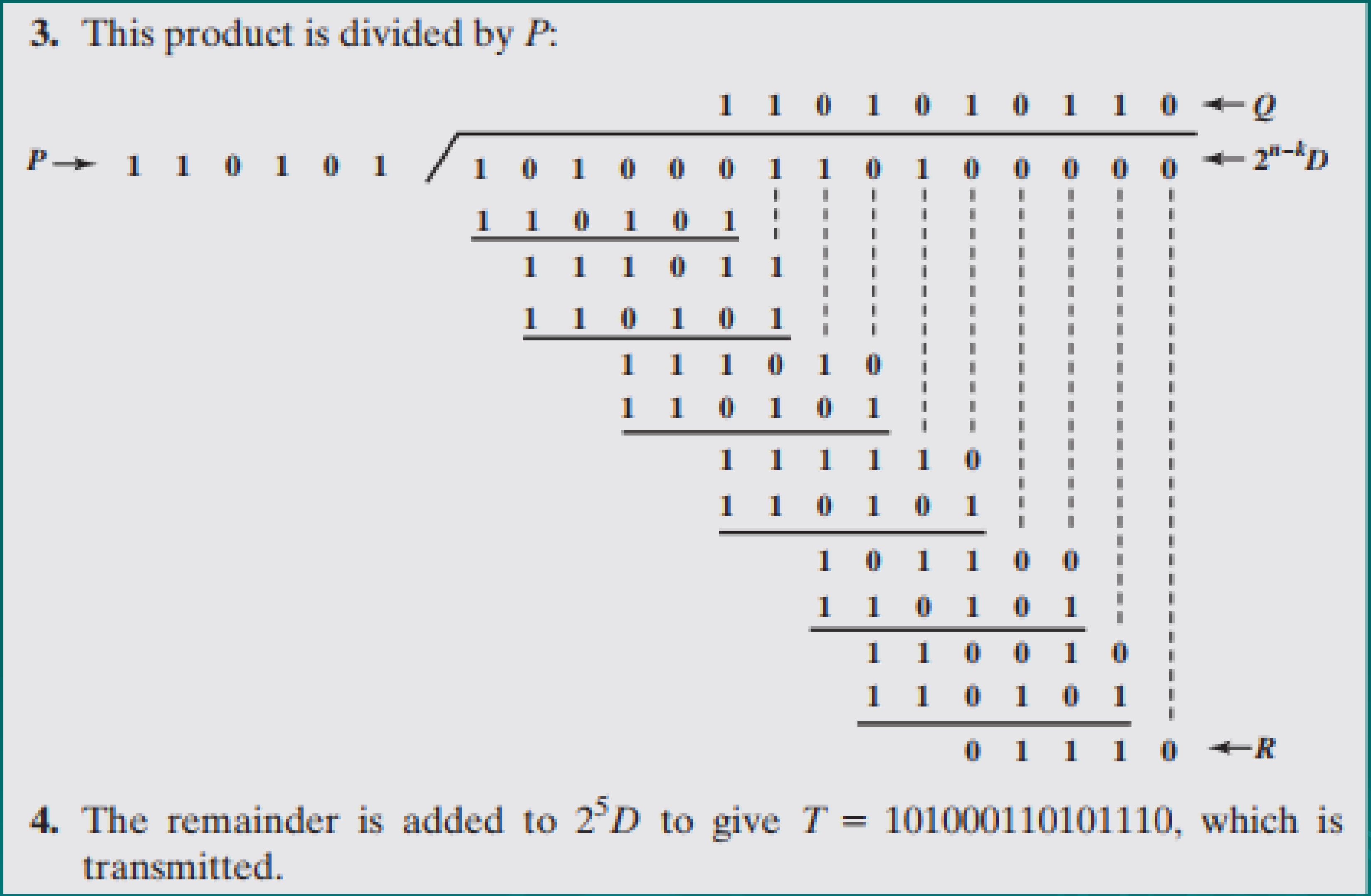


1. Given

$$\begin{aligned} \text{Message } D &= 1010001101 \ (10 \text{ bits}) \\ \text{Pattern } P &= 110101 \ (6 \text{ bits}) \\ \text{FCS } R &= \text{ to be calculated } (5 \text{ bits}) \end{aligned}$$

Thus, $n = 15$, $k = 10$, and $(n - k) = 5$.

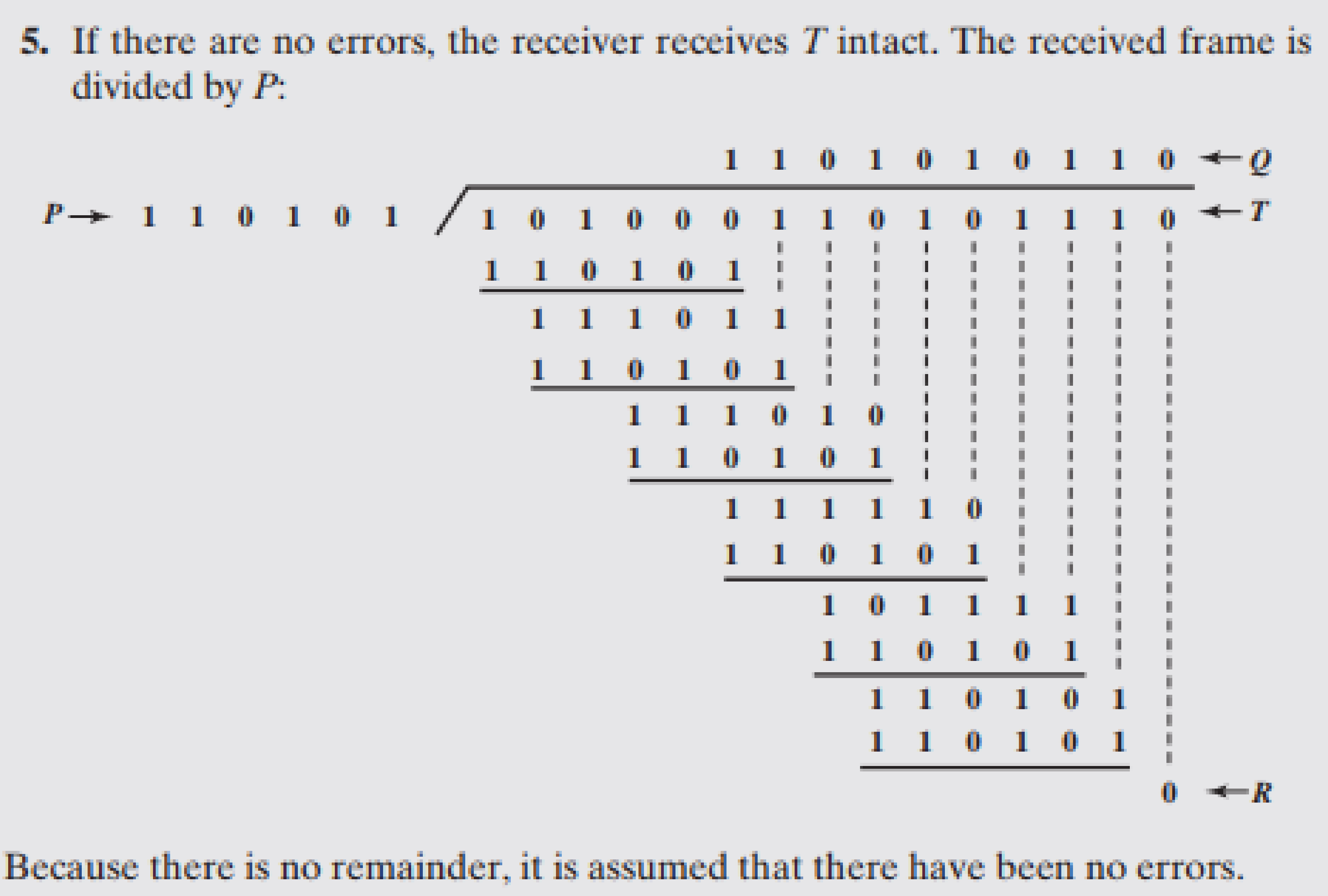
2. The message is multiplied by 2^5, yielding 101000110100000.



3. This product is divided by P:

```
                            1 1 0 1 0 1 0 1 1 0  ← Q
                  _______________________________
P → 1 1 0 1 0 1 ) 1 0 1 0 0 0 1 1 0 1 0 0 0 0 0  ← 2^{n−k}D
                  1 1 0 1 0 1
                  -----------
                    1 1 1 0 1 1
                    1 1 0 1 0 1
                    -----------
                        1 1 1 0 1 0
                        1 1 0 1 0 1
                        -----------
                            1 1 1 1 1 0
                            1 1 0 1 0 1
                            -----------
                                1 0 1 1 0 0
                                1 1 0 1 0 1
                                -----------
                                  1 1 0 0 1 0
                                  1 1 0 1 0 1
                                  -----------
                                      0 1 1 1 0  ← R
```

4. The remainder is added to 2^5D to give T = 101000110101110, which is transmitted.



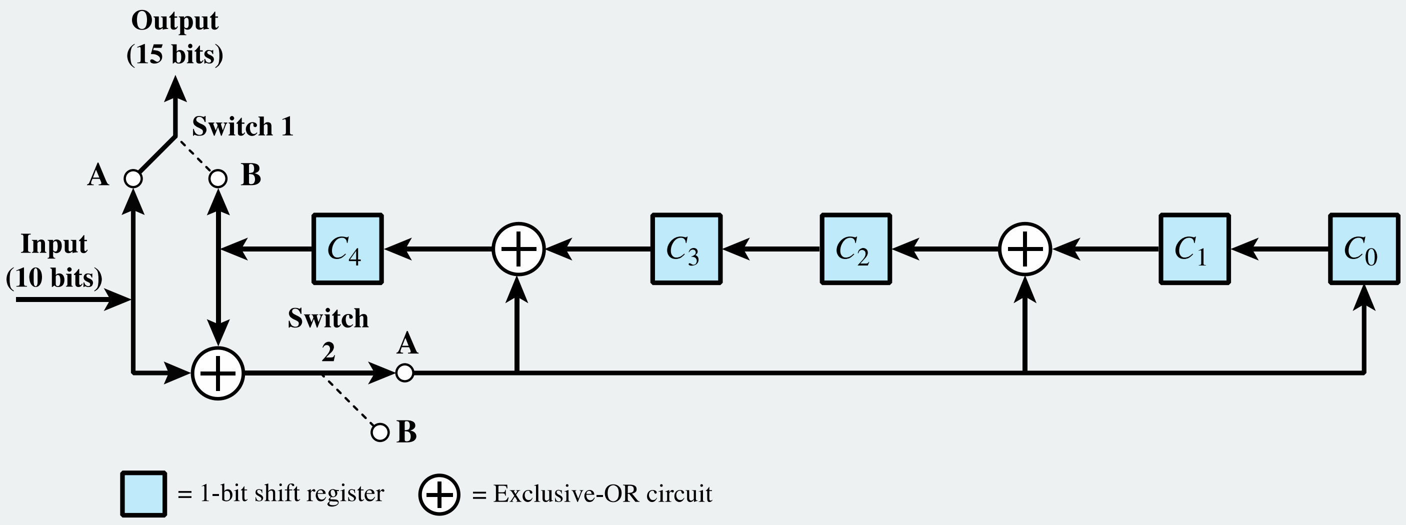
5. If there are no errors, the receiver receives *T* intact. The received frame is divided by *P*:

```
                             1 1 0 1 0 1 0 1 1 0  ← Q
                   _____________________________
P → 1 1 0 1 0 1  / 1 0 1 0 0 0 1 1 0 1 0 1 1 1 0  ← T
                   1 1 0 1 0 1
                   -----------
                     1 1 1 0 1 1
                     1 1 0 1 0 1
                     -----------
                         1 1 1 0 1 0
                         1 1 0 1 0 1
                         -----------
                             1 1 1 1 1 0
                             1 1 0 1 0 1
                             -----------
                                 1 0 1 1 1 1
                                 1 1 0 1 0 1
                                 -----------
                                   1 1 0 1 0 1
                                   1 1 0 1 0 1
                                   -----------
                                               0  ← R
```

Because there is no remainder, it is assumed that there have been no errors.

$$
\begin{array}{rllllllllllllllll}
 & \multicolumn{14}{l}{X^9+X^8+X^6+X^4+X^2+X} & \leftarrow Q(X) \\ \cline{2-15}
P(X)\rightarrow X^5+X^4+X^2+1 \;\big) & X^{14} & & X^{12} & & & & X^8+ & X^7+ & & X^5 & & & & & \leftarrow X^5D(X) \\
 & X^{14}+ & X^{13}+ & & X^{11}+ & & X^9 & & & & & & & & & \\ \cline{2-7}
 & & X^{13}+ & X^{12}+ & X^{11}+ & & X^9+ & X^8 & & & & & & & & \\
 & & X^{13}+ & X^{12}+ & & X^{10}+ & & X^8 & & & & & & & & \\ \cline{3-8}
 & & & & X^{11}+ & X^{10}+ & X^9+ & & X^7 & & & & & & & \\
 & & & & X^{11}+ & X^{10}+ & & X^8+ & & X^6 & & & & & & \\ \cline{5-10}
 & & & & & & X^9+ & X^8+ & X^7+ & X^6+ & X^5 & & & & & \\
 & & & & & & X^9+ & X^8+ & & X^6+ & & X^4 & & & & \\ \cline{7-12}
 & & & & & & & & X^7+ & & X^5+ & X^4 & & & & \\
 & & & & & & & & X^7+ & X^6+ & & X^4+ & & X^2 & & \\ \cline{9-14}
 & & & & & & & & & X^6+ & X^5+ & & & X^2 & & \\
 & & & & & & & & & X^6+ & X^5+ & & X^3+ & & X & \\ \cline{10-15}
 & & & & & & & & & & & & X^3+ & X^2+ & X & \leftarrow R(X)
\end{array}
$$

Figure 6.5 Example of Polynomial Division



(a) Shift-register implementation

	C_4	C_3	C_2	C_1	C_0	$C_4 \approx C_3 \approx I$	$C_4 \approx C_1 \approx I$	$C_4 \approx I$	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
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				

Message to be sent

(b) Example with input of 1010001101

Figure 6.6 Circuit with Shift Registers for Dividing by the Polynomial $X^5 + X^4 + X^2 + 1$

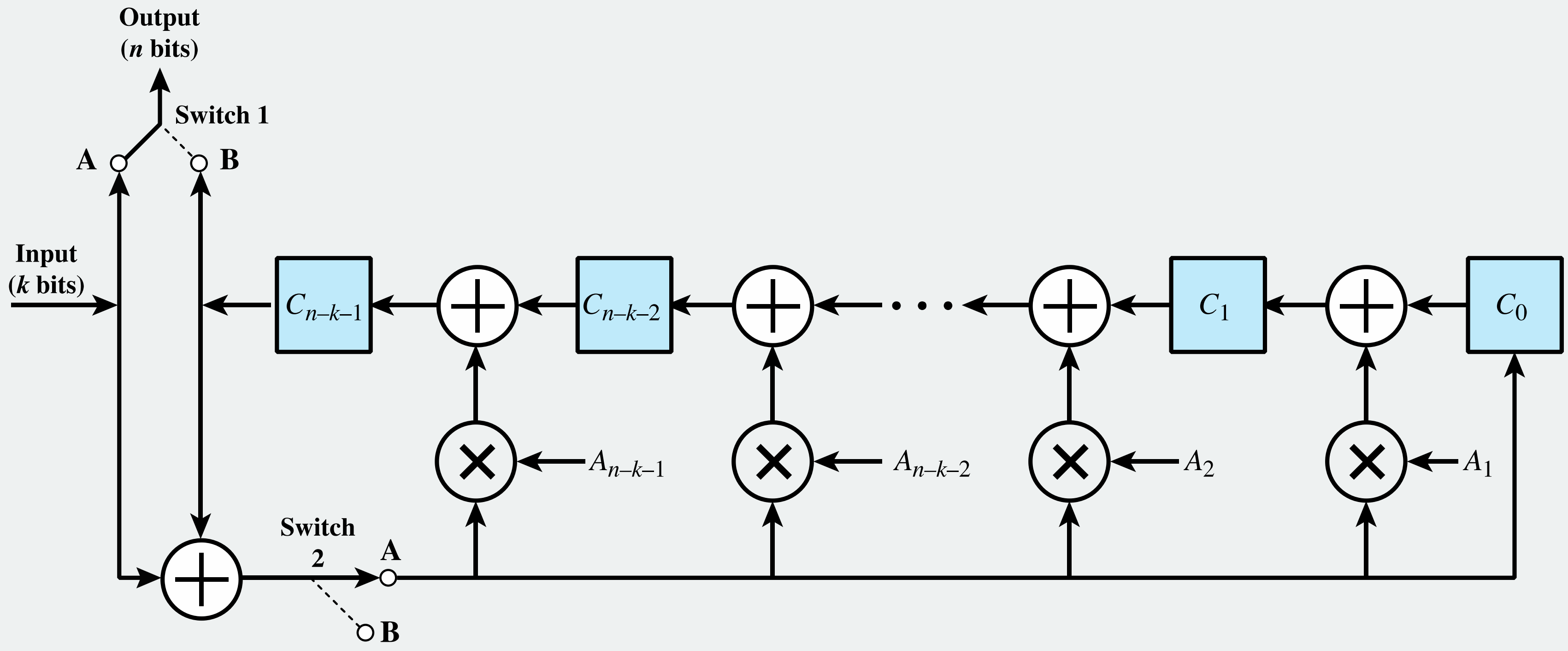


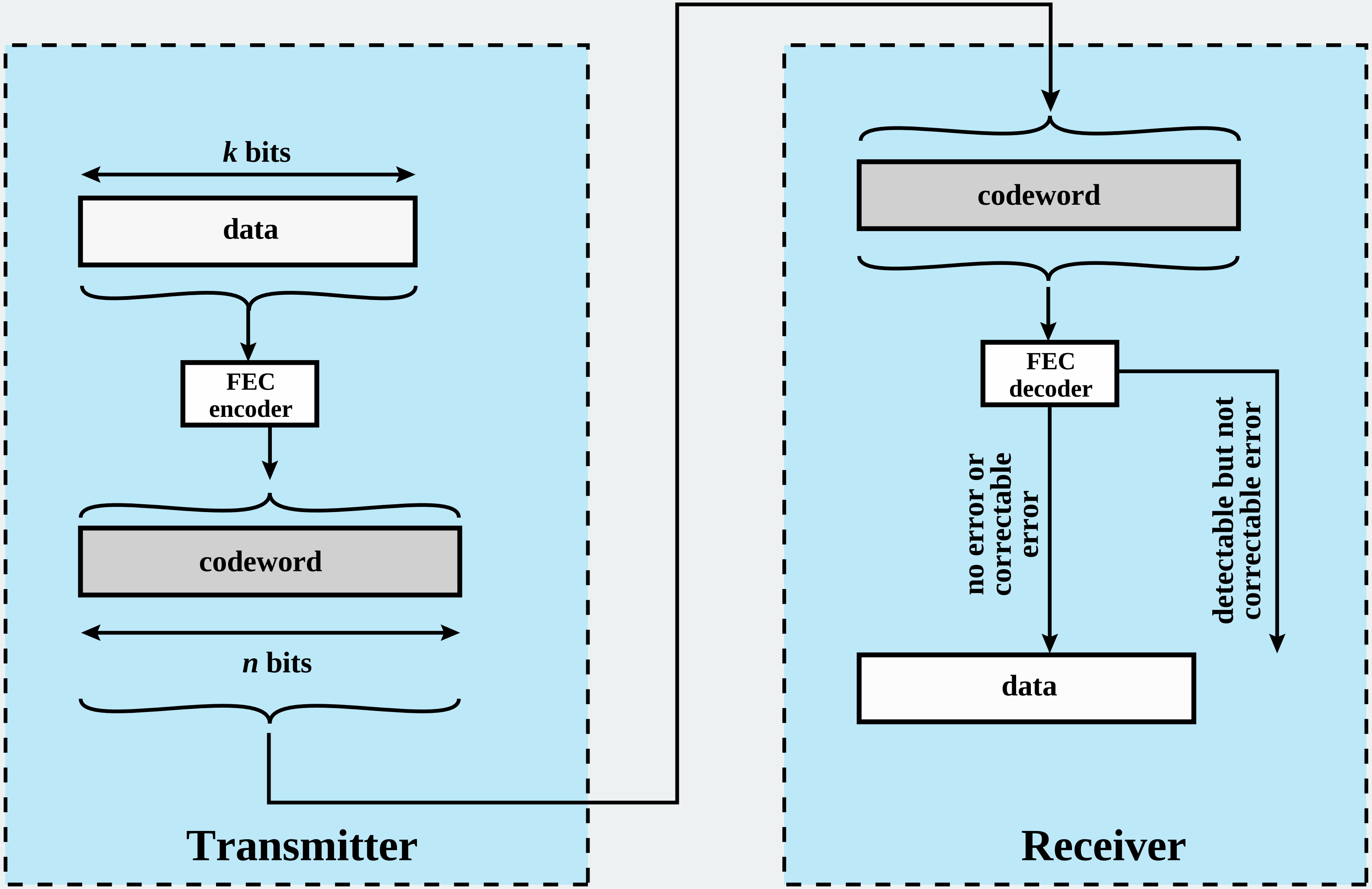
Figure 6.7 General CRC Architecture to Implement Divisor $(1 + A_1X + A_2X^2 + \ldots + A_{n-k-1}X^{n-k-1} + X^{n-k})$

Forward Error Correction

- Correction of detected errors usually requires data blocks to be retransmitted
- Not appropriate for wireless applications:
  - The bit error rate (BER) on a wireless link can be quite high, which would result in a large number of retransmissions
  - Propagation delay is very long compared to the transmission time of a single frame
- Need to correct errors on basis of bits received

Codeword

- On the transmission end each *k*-bit block of data is mapped into an *n*-bit block (*n* > *k*) using a forward error correction (FEC) encoder



- Detectable, corectable error
- Detectable, not corractable errors
- Undetactable errors

Figure 6.8 Error Correction Process

Block Code Principles

- Hamming distance
  - $d(v_1, v_2)$ between two *n* –bit binary sequences v_1 and v_2 is the number of bits in which v_1 and v_2 disagree
- Redundancy of the code
  - The ratio of redundant bits to data bits $(n\text{-}k)/k$
- Code rate
  - The ratio of data bits to total bits k/n
  - Is a measure of how much additional bandwidth is required to carry data at the same data rate as without the code

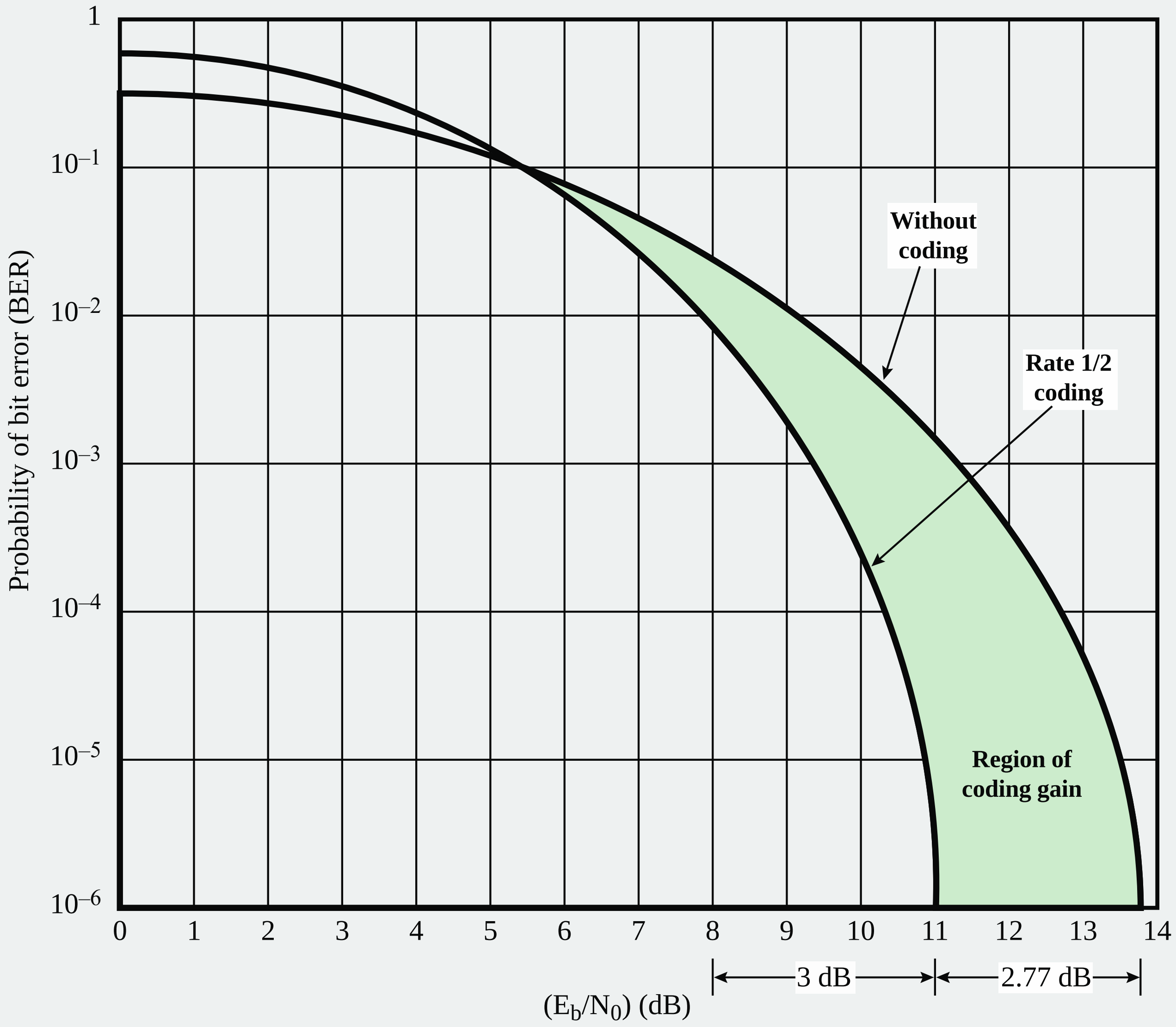


Figure 6.9 How Coding Improves System Performance