Coding and Error Control

Chapter 8

Coping with Data Transmission Errors

- Error detection codes
  - Detects the presence of an error
- Automatic repeat request (ARQ) protocols
  - Block of data with error is discarded
  - Transmitter retransmits that block of data
- Error correction codes, or forward correction codes (FEC)
  - Designed to detect and correct errors

Error Detection Probabilities

- Definitions:
  - P_b : Probability of single bit error (BER)
  - P_1 : Probability that a frame arrives with no bit errors
  - P_2 : While using error detection, the probability that a frame arrives with one or more undetected errors
  - P_3 : While using error detection, the probability that a frame arrives with one or more detected bit errors but no undetected bit errors

Error Detection Probabilities

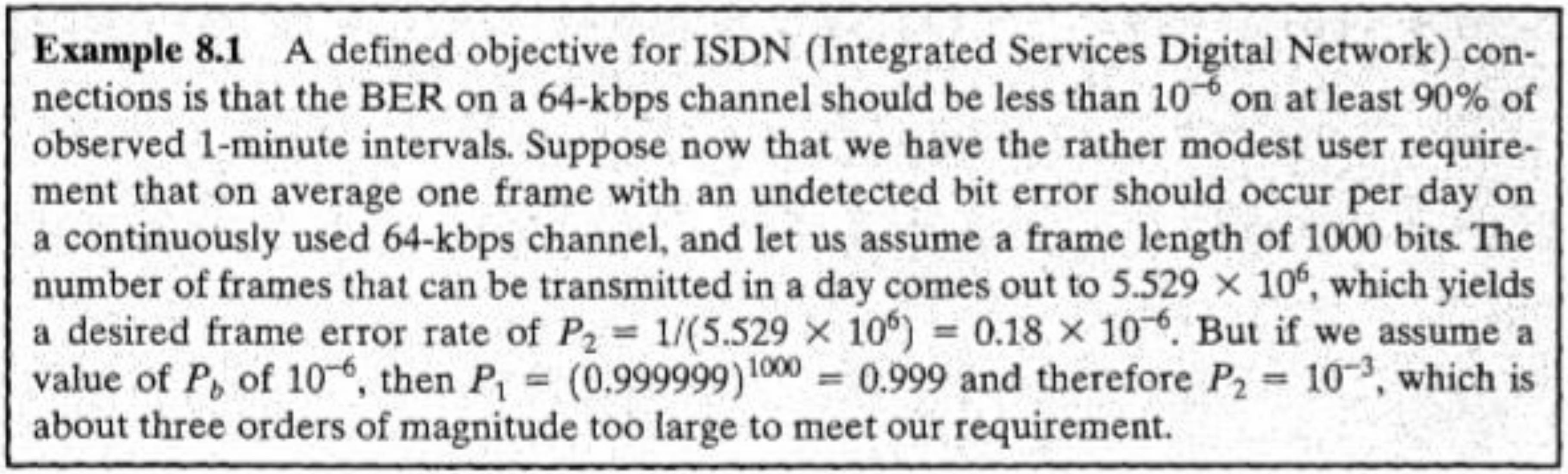
- With no error detection:

$$P_1 = (1 - P_b)^F$$

$$P_2 = 1 - P_1$$

$$P_3 = 0$$

  - F = Number of bits per frame



Example 8.1 A defined objective for ISDN (Integrated Services Digital Network) connections is that the BER on a 64-kbps channel should be less than 10^{-6} on at least 90% of observed 1-minute intervals. Suppose now that we have the rather modest user requirement that on average one frame with an undetected bit error should occur per day on a continuously used 64-kbps channel, and let us assume a frame length of 1000 bits. The number of frames that can be transmitted in a day comes out to 5.529×10^6, which yields a desired frame error rate of $P_2 = 1/(5.529 \times 10^6) = 0.18 \times 10^{-6}$. But if we assume a value of P_b of 10^{-6}, then $P_1 = (0.999999)^{1000} = 0.999$ and therefore $P_2 = 10^{-3}$, which is about three orders of magnitude too large to meet our requirement.

Error Detection Process

- Transmitter
  - For a given frame, an error-detecting code (check bits) is calculated from data bits
  - Check bits are appended to data bits

- Receiver
  - Separates incoming frame into data bits and check bits
  - Calculates check bits from received data bits
  - Compares calculated check bits against received check bits
  - Detected error occurs if mismatch

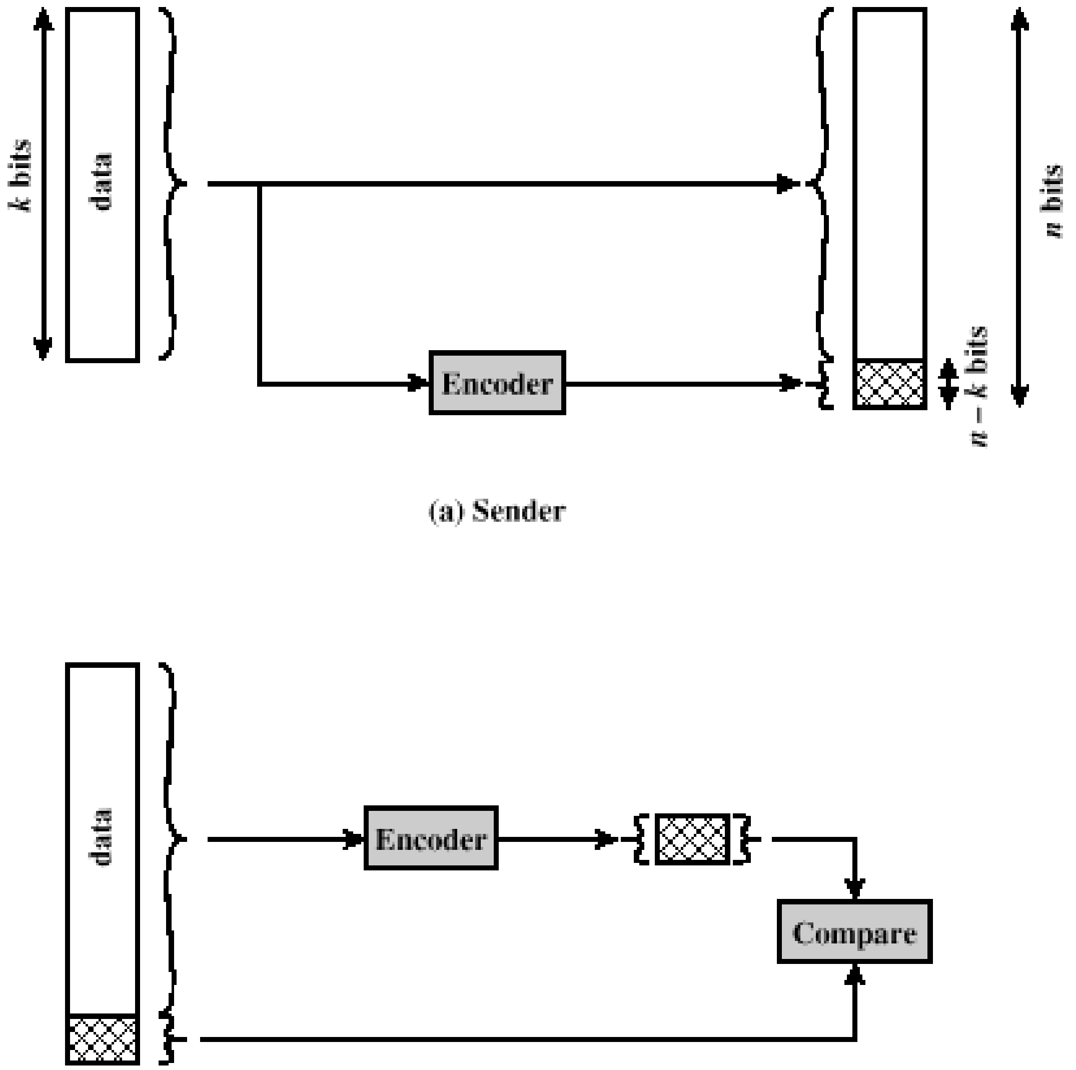


Figure 8.1 Error Detection Process

Parity Check

- Parity bit appended to a block of data
- Even parity
  - Added bit ensures an even number of 1s
- Odd parity
  - Added bit ensures an odd number of 1s
- Example, 7-bit character [1110001]
  - Even parity [11100010]
  - Odd parity [11100011]

Cyclic Redundancy Check (CRC)

- Transmitter
  - For a *k*-bit block, transmitter generates an (*n*-*k*)-bit frame check sequence (FCS)
  - Resulting frame of *n* bits is exactly divisible by predetermined number
- Receiver
  - Divides incoming frame by predetermined number
  - If no remainder, assumes no error

CRC using Modulo 2 Arithmetic

- Exclusive-OR (XOR) operation:

$$\begin{array}{r} 1111 \\ +\ \underline{1010} \\ 0101 \end{array} \qquad \begin{array}{r} 1111 \\ -\ \underline{0101} \\ 1010 \end{array} \qquad \begin{array}{r} 11001 \\ \underline{\times\ 11} \\ 11001 \\ \underline{11001} \\ 101011 \end{array}$$

- Parameters:
  - T = n-bit frame to be transmitted
  - D = k-bit block of data; 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
  - Q = Quotient
  - R = Remainder

CRC using Modulo 2 Arithmetic

- For *T*/*P* to have no remainder, start with:

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

- Divide $2^{n\text{-}k}D$ by *P* gives quotient and remainder

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

- Use remainder as FCS:

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

CRC using Modulo 2 Arithmetic

- Does *R* cause *T*/*P* have no remainder?

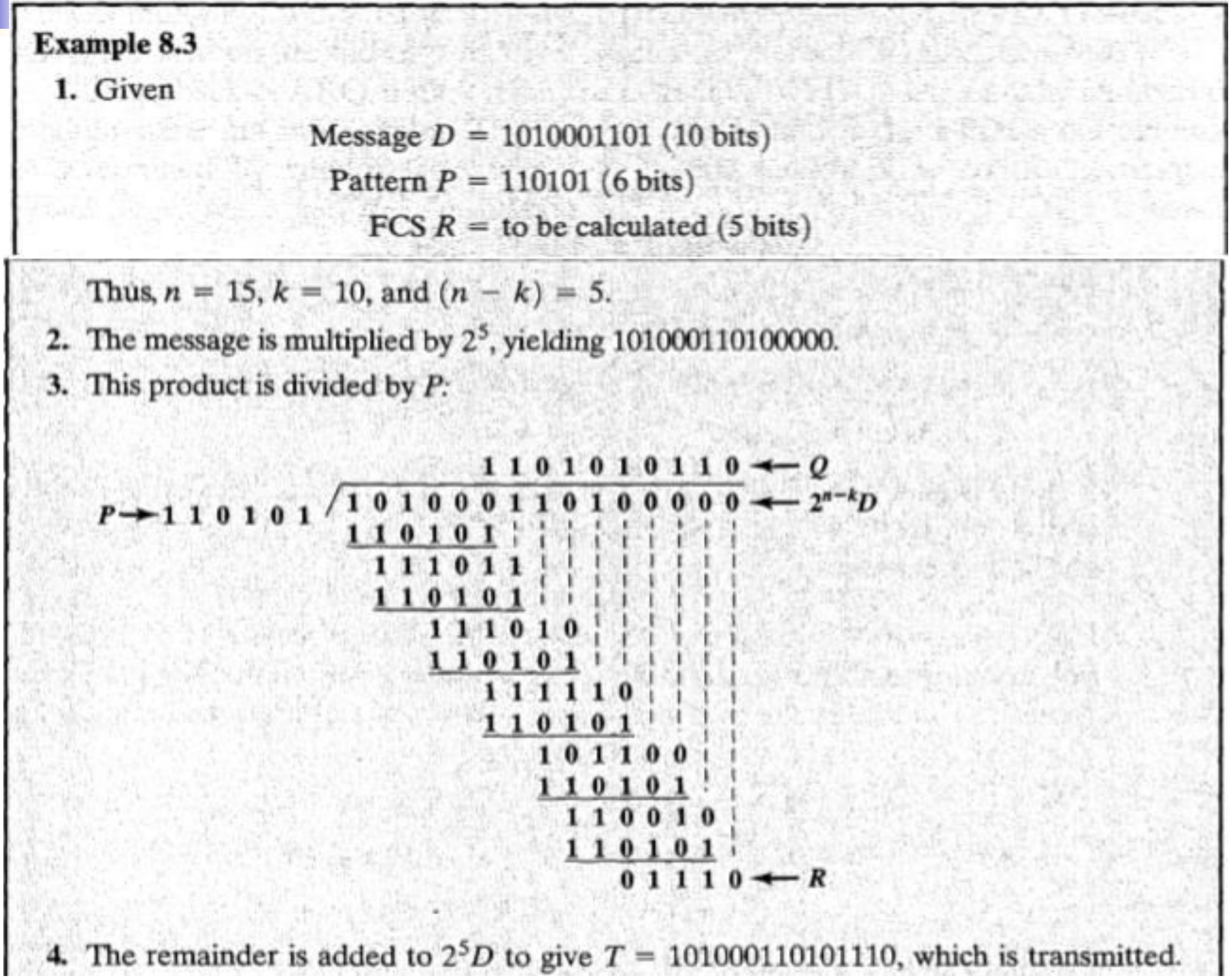
$$\frac{T}{P} = \frac{2^{n-k}D + R}{P} = \frac{2^{n-k}D}{P} + \frac{R}{P}$$

- Substituting,

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

  - No remainder, so *T* is exactly divisible by *P*

Example



Example 8.3

1. Given

Message D = 1010001101 (10 bits)
Pattern P = 110101 (6 bits)
FCS R = to be calculated (5 bits)

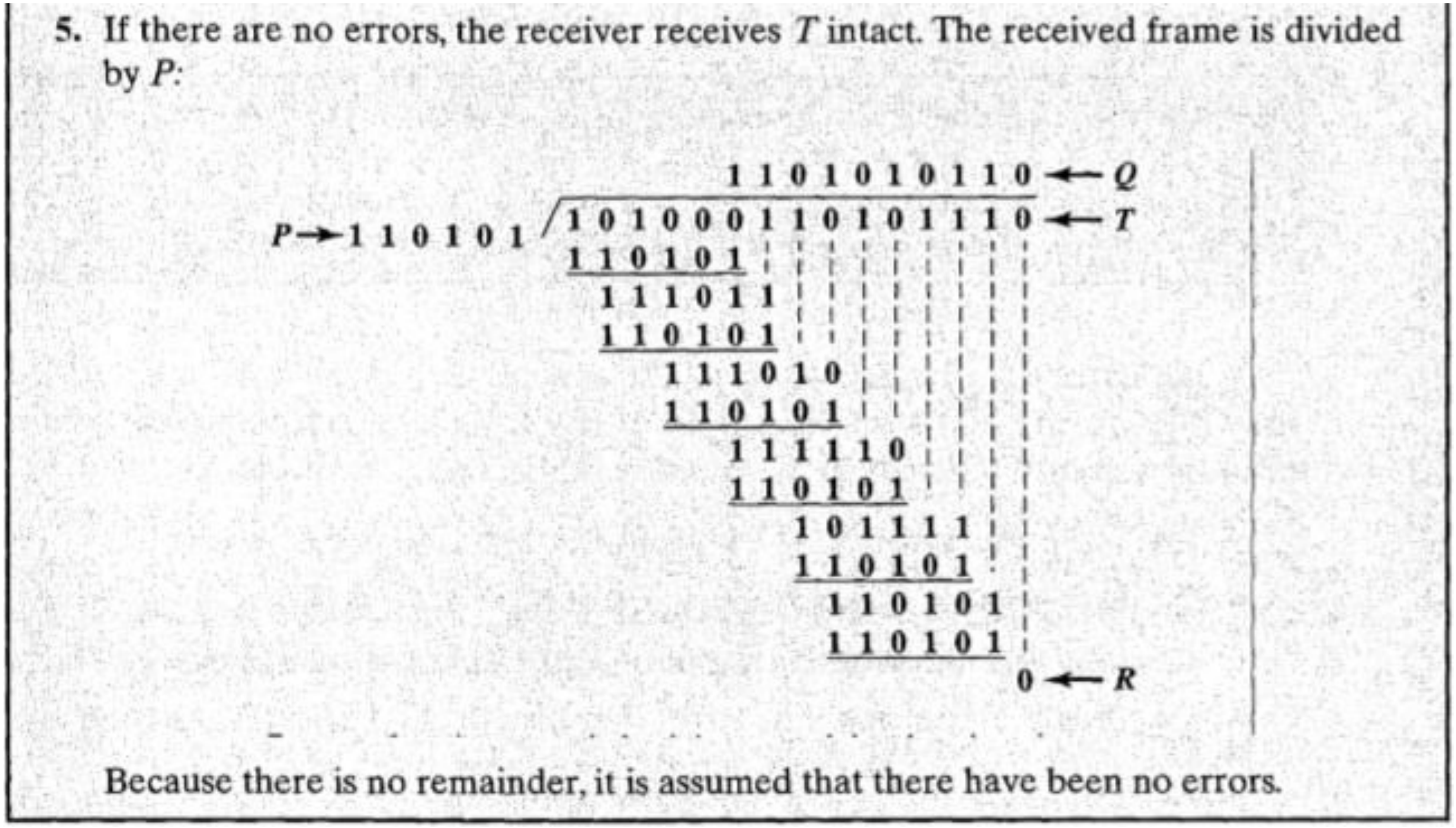
Thus, $n = 15$, $k = 10$, and $(n - k) = 5$.

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.

Example (cont.)



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.

Wireless Transmission Errors

- Error detection requires retransmission
- Retransmission inadequate for wireless applications:
  - Error rate on wireless link can be high, results in a large number of retransmissions
  - Long propagation delay compared to transmission time (ex. Satellite communication)

Block Error Correction Codes

- Transmitter
  - Forward error correction (FEC) encoder maps each k-bit block into an n-bit block codeword
  - Codeword is transmitted; analog for wireless transmission
- Receiver
  - Incoming signal is demodulated
  - Block passed through an FEC decoder

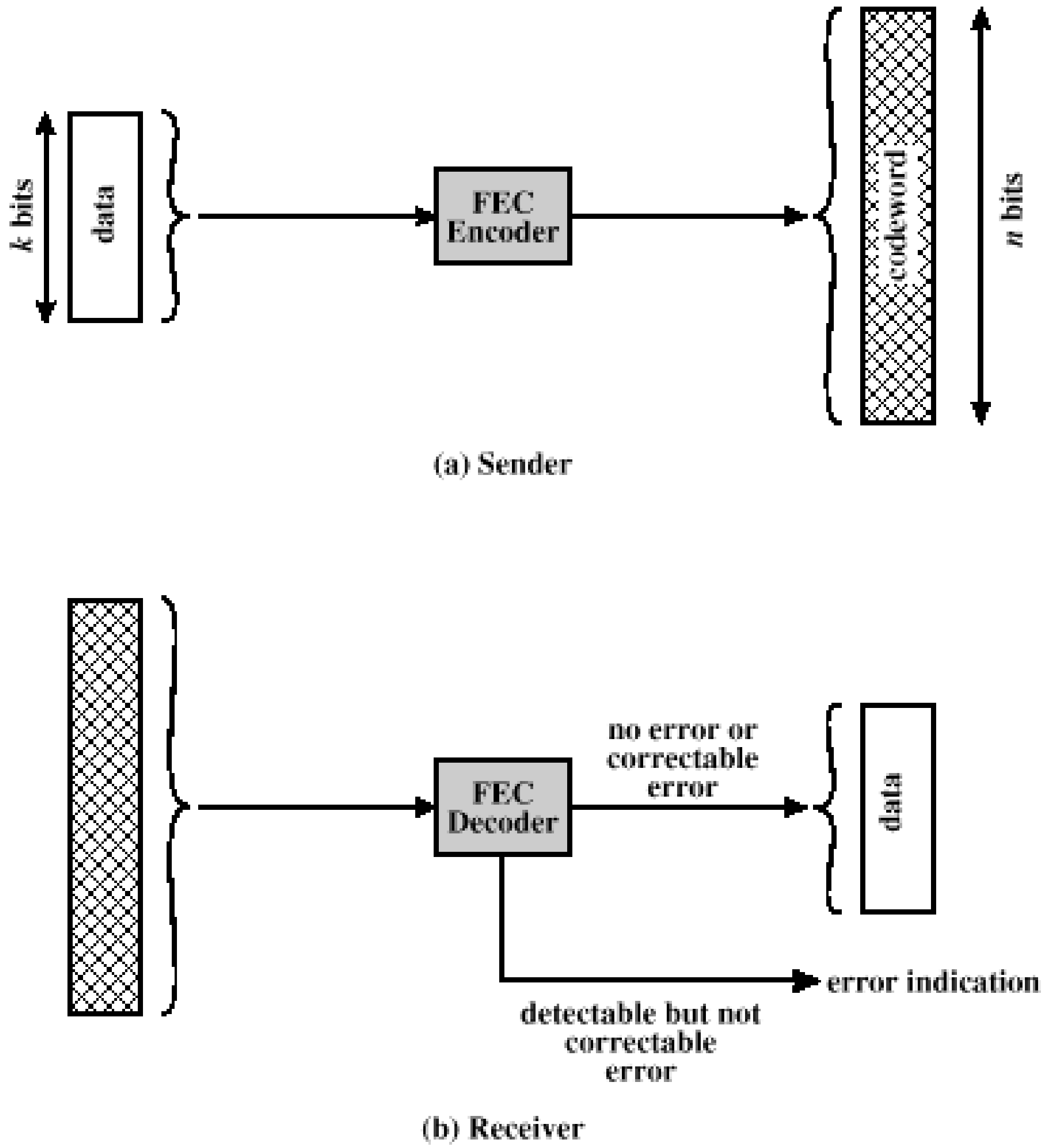


Figure 8.5 Forward Error Correction Process

FEC Decoder Outcomes:

- No errors present
  - Codeword produced by decoder matches original codeword
- Decoder detects and corrects bit errors
- Decoder detects but cannot correct bit errors; reports uncorrectable error
- Decoder cannot detect bit errors, even though there are errors

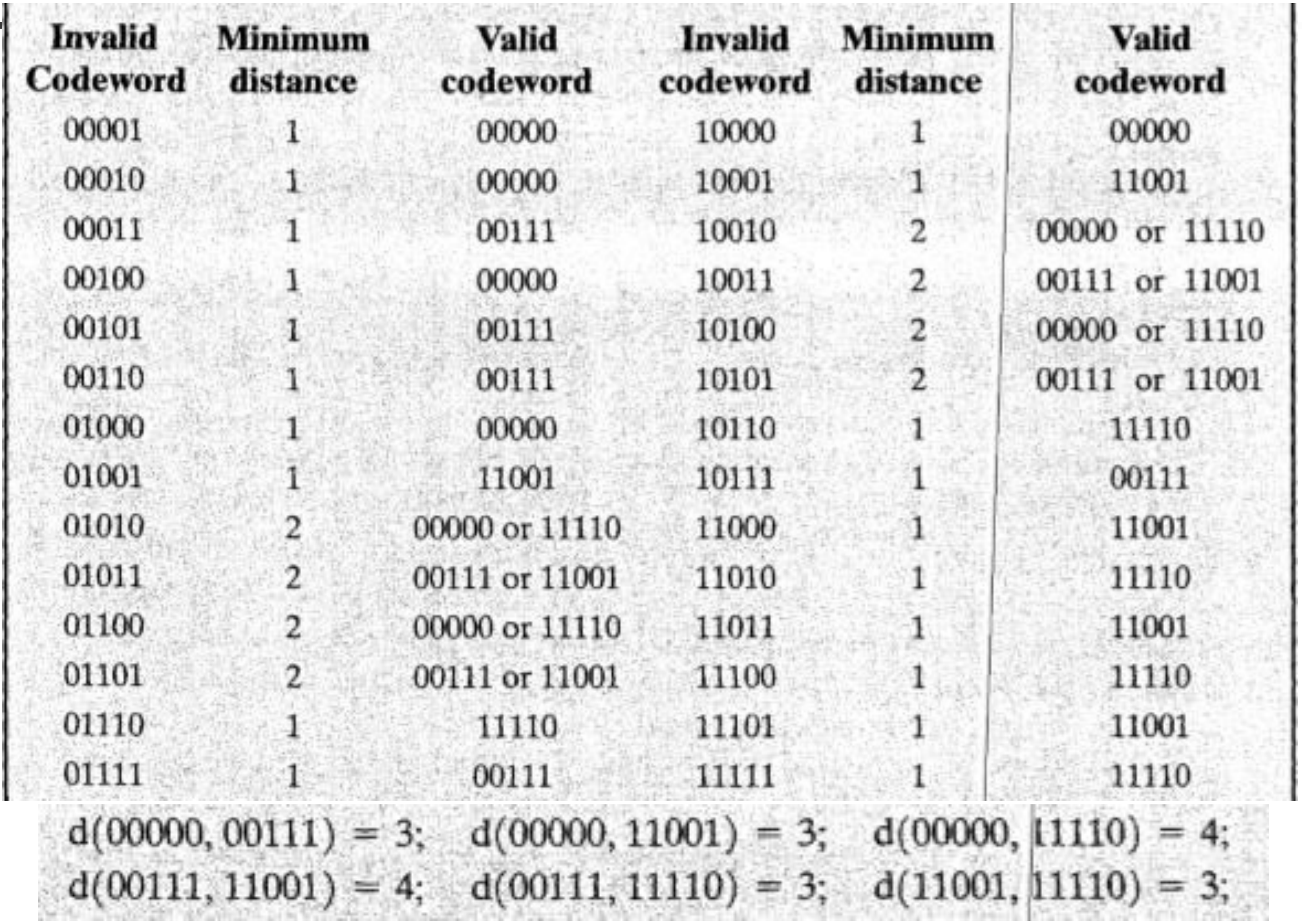
Block Code Principles

- Hamming distance – for 2 *n*-bit binary sequences, the number of different bits:
  - E.g., v_1=011011; v_2=110001; $d(v_1, v_2)$=3
- For *K*=2 and *n*=5, we can make the following assignment:

Data block	Codeword
00	00000
01	00111
10	11001
11	11110

- If the received codeword has the value of: 00100 ?

Hamming distance (Cont.)



Invalid Codeword	Minimum distance	Valid codeword	Invalid codeword	Minimum distance	Valid codeword
00001	1	00000	10000	1	00000
00010	1	00000	10001	1	11001
00011	1	00111	10010	2	00000 or 11110
00100	1	00000	10011	2	00111 or 11001
00101	1	00111	10100	2	00000 or 11110
00110	1	00111	10101	2	00111 or 11001
01000	1	00000	10110	1	11110
01001	1	11001	10111	1	00111
01010	2	00000 or 11110	11000	1	11001
01011	2	00111 or 11001	11010	1	11110
01100	2	00000 or 11110	11011	1	11001
01101	2	00111 or 11001	11100	1	11110
01110	1	11110	11101	1	11001
01111	1	00111	11111	1	11110

d(00000, 00111) = 3; d(00000, 11001) = 3; d(00000, 11110) = 4;
d(00111, 11001) = 4; d(00111, 11110) = 3; d(11001, 11110) = 3;

Max number of correctable errors: $t = \left\lfloor \dfrac{d_{\min} - 1}{2} \right\rfloor$

Block Code Principles

- Redundancy – ratio of redundant bits to data bits
- Code rate – ratio of data bits to total bits
- Coding gain – the reduction in the required E_b/N_0 to achieve a specified BER of an error-correcting coded system

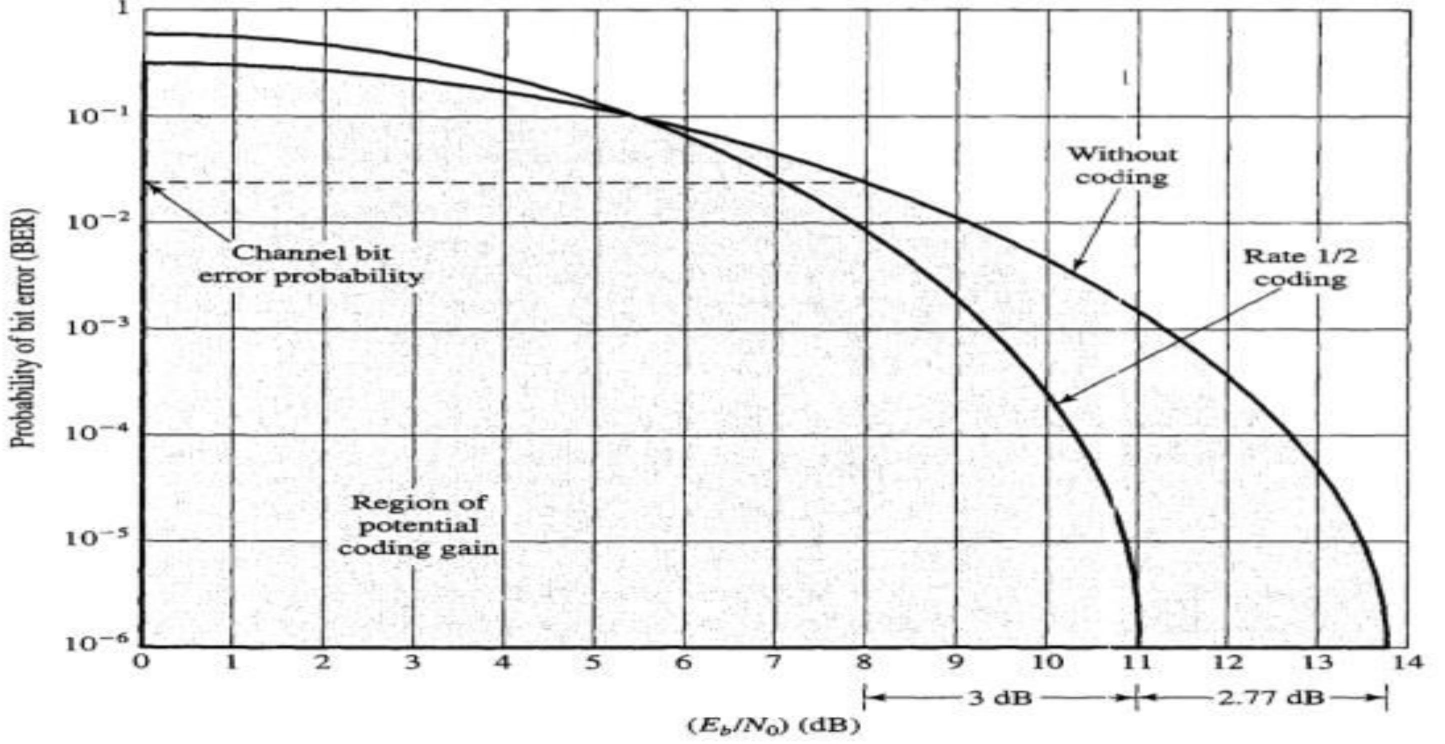


Figure 8.6 How Coding Improves System Performance

FEC Design

- The design of a block code involves a number of considerations:
- For given values of *n* and *k*, we would like the largest possible value of d_{min}.
- The code should be relatively easy to encode and decode, requiring minimal memory and processing time.
- The number of extra bits, (n - k), needs to be small, to reduce bandwidth.
- The number of extra bits, (n - k), needs to be large, to reduce error-rate.

Hamming Code

- Designed to correct single bit errors
- Family of (n, k) block error-correcting codes with parameters:
  - Block length: $n = 2^m - 1$
  - Number of data bits: $k = 2^m - m - 1$
  - Number of check bits: $n - k = m$
  - Minimum distance: $d_{\min} = 3$
- Single-error-correcting (SEC) code
  - SEC double-error-detecting (SEC-DED) code

Hamming Code Process

- Encoding: k data bits + $(n - k)$ check bits
  - Check bits: XORed the positions of ones in the data bits
  - Inserted in the positions that are a power of 2
- Decoding: compares received $(n - k)$ bits with calculated $(n - k)$ bits using XOR:
  - Resulting $(n - k)$ bits called *syndrome word*
  - Syndrome range is between 0 and $2^{(n-k)}-1$
  - Each bit of syndrome indicates a match (0) or conflict (1) in that bit position

Hamming Code

- Hamming Code generates a syndrome with the following characteristics:
  - If the syndrome contains all 0s, no error has been detected.
  - If the syndrome contains one and only one bit set to 1 , then an error has occurred in one of the check bits. No correction is needed.
  - If the syndrome contains more than one bit set to 1, then the syndrome indicates the position of the data bit in error. This data bit is inverted for correction.

Example:

8-bit data block: 00111001

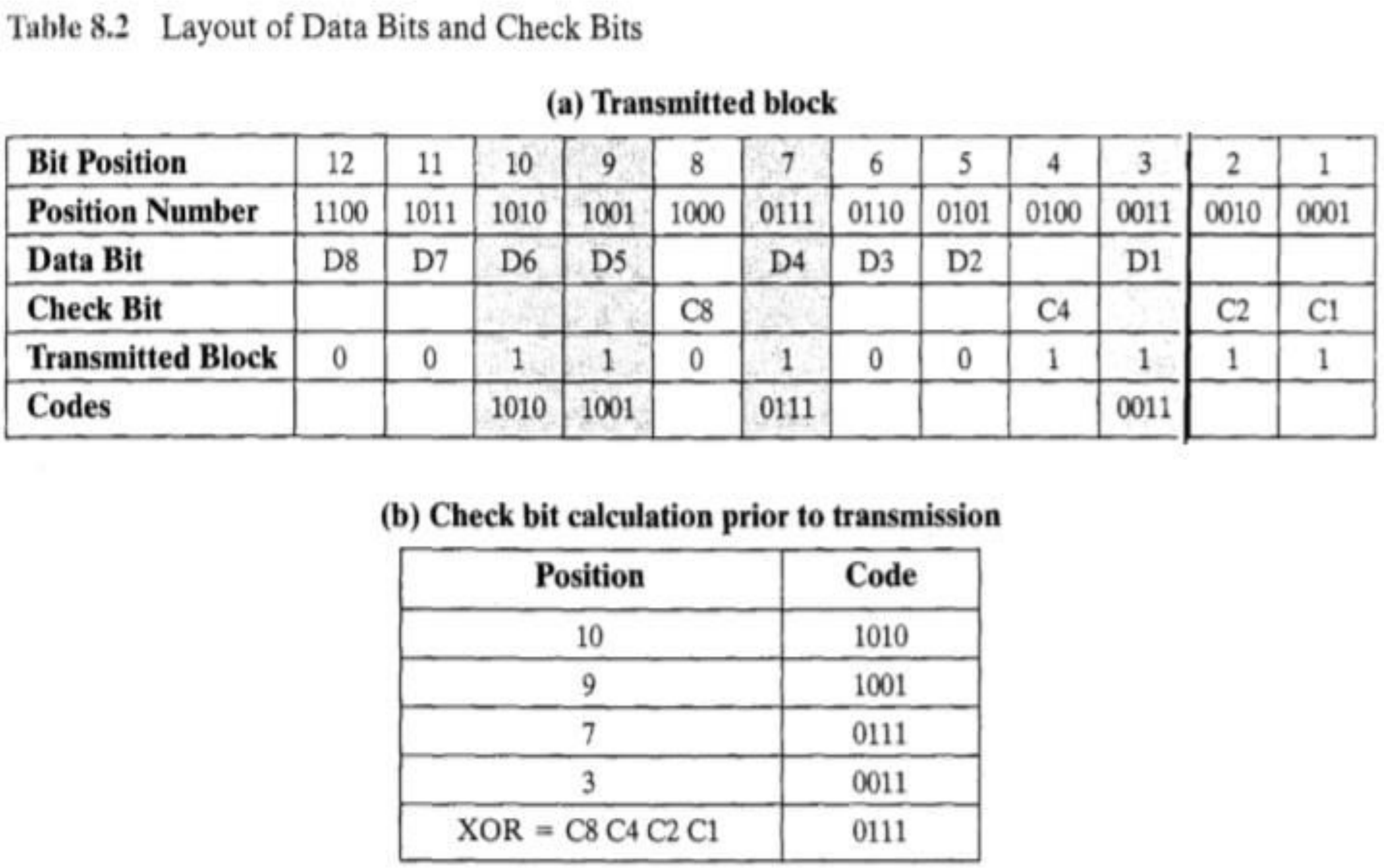


Table 8.2 Layout of Data Bits and Check Bits

(a) Transmitted block

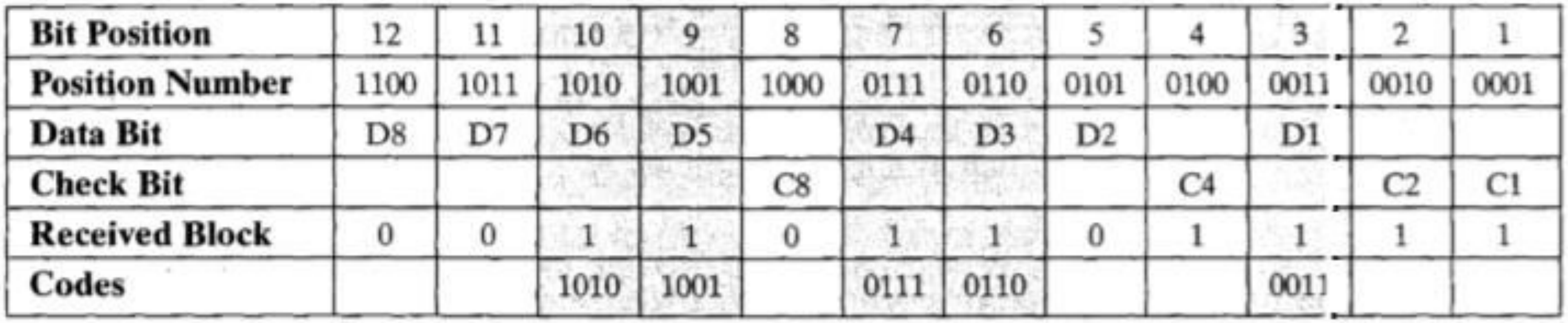
Bit Position	12	11	10	9	8	7	6	5	4	3	2	1
Position Number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data Bit	D8	D7	D6	D5		D4	D3	D2		D1		
Check Bit					C8				C4		C2	C1
Transmitted Block	0	0	1	1	0	1	0	0	1	1	1	1
Codes			1010	1001		0111				0011		

(b) Check bit calculation prior to transmission

Position	Code
10	1010
9	1001
7	0111
3	0011
XOR = C8 C4 C2 C1	0111

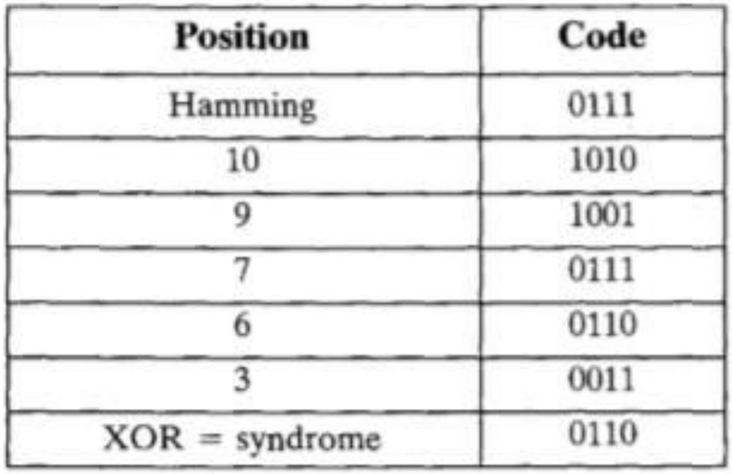
Example (cont.)

(c) Received block



Bit Position	12	11	10	9	8	7	6	5	4	3	2	1
Position Number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data Bit	D8	D7	D6	D5		D4	D3	D2		D1		
Check Bit					C8				C4		C2	C1
Received Block	0	0	1	1	0	1	1	0	1	1	1	1
Codes			1010	1001		0111	0110			0011		

(d) Check bit calculation after reception



Position	Code
Hamming	0111
10	1010
9	1001
7	0111
6	0110
3	0011
XOR = syndrome	0110

Cyclic Codes

- Can be encoded and decoded using linear feedback shift registers (LFSRs)
- For cyclic codes, a valid codeword $(c_0, c_1, \ldots, c_{n-1})$, shifted right one bit, is also a valid codeword $(c_{n-1}, c_0, \ldots, c_{n-2})$
- Takes fixed-length input (k) and produces fixed-length check code (n-k)
  - In contrast, CRC error-detecting code accepts arbitrary length input for fixed-length check code

BCH Codes

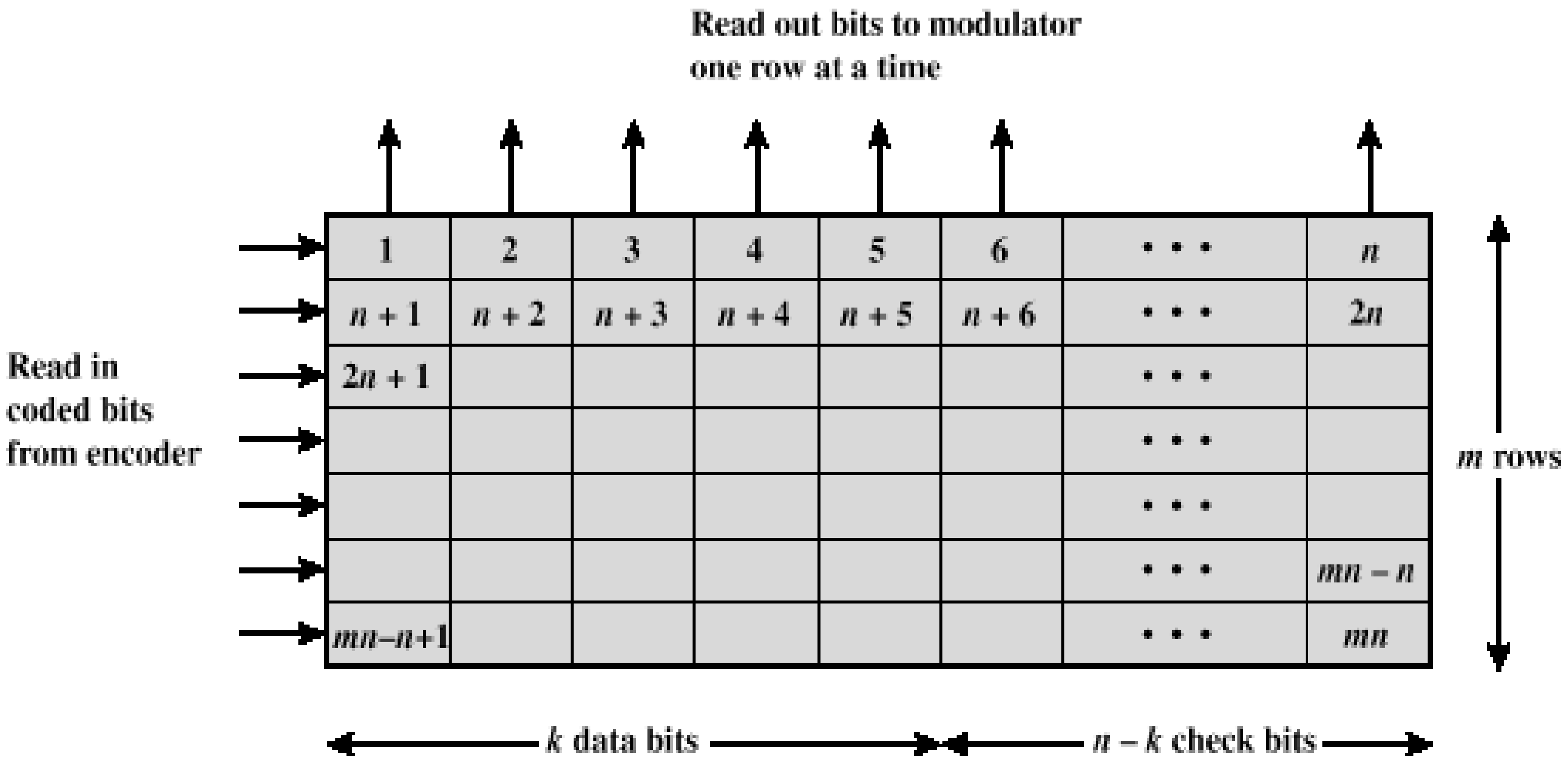
- For positive pair of integers m and t, a (n, k) BCH code has parameters:
  - Block length: $n = 2^m - 1$
  - Number of check bits: $n - k$ £ mt
  - Minimum distance:$d_{\min}$ ³ $2t + 1$
- Correct combinations of t or fewer errors
- Flexibility in choice of parameters
  - Block length, code rate

Reed-Solomon Codes

- Subclass of nonbinary BCH codes
- Data processed in chunks of m bits, called symbols
- An (n, k) RS code has parameters:
  - Symbol length: m bits per symbol
  - Block length: $n = 2^m - 1$ symbols $= m(2^m - 1)$ bits
  - Data length: k symbols
  - Size of check code: $n - k = 2t$ symbols $= m(2t)$ bits
  - Minimum distance: $d_{\text{min}} = 2t + 1$ symbols

Block Interleaving

- Data written to and read from memory in different orders
- Data bits and corresponding check bits are interspersed with bits from other blocks
- At receiver, data are deinterleaved to recover original order
- A burst error that may occur is spread out over a number of blocks, making error correction possible

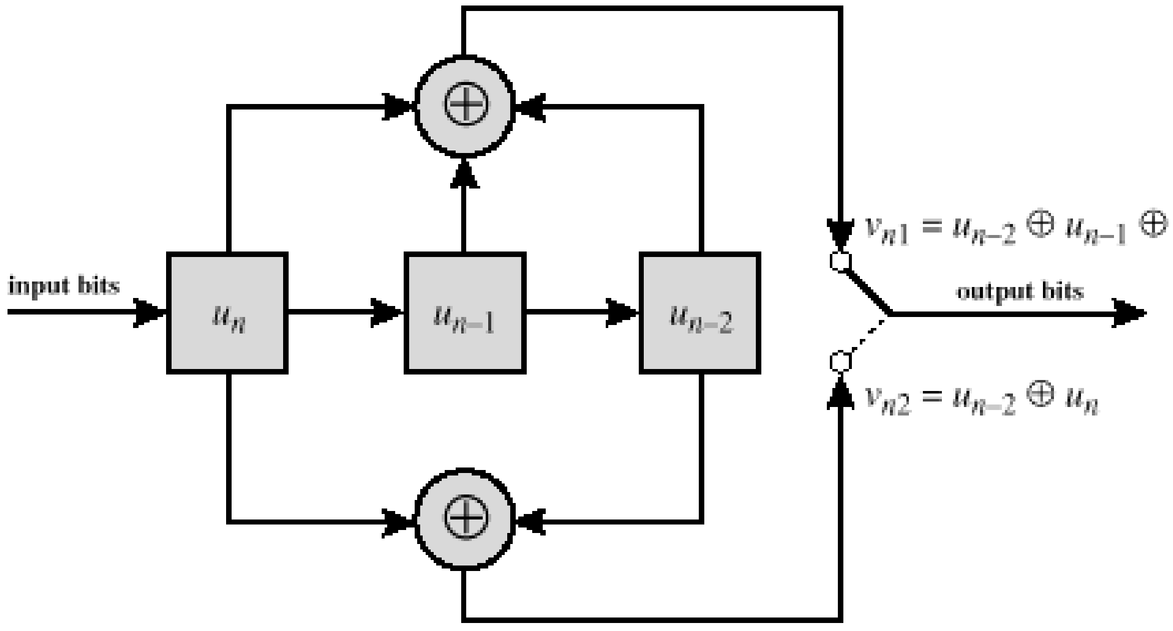


Note: The numbers in the matrix indicate the order in which bits are read in.
Interleaver output sequence: 1, $n + 1$, $2n + 1$, …

Figure 8.8 Block Interleaving

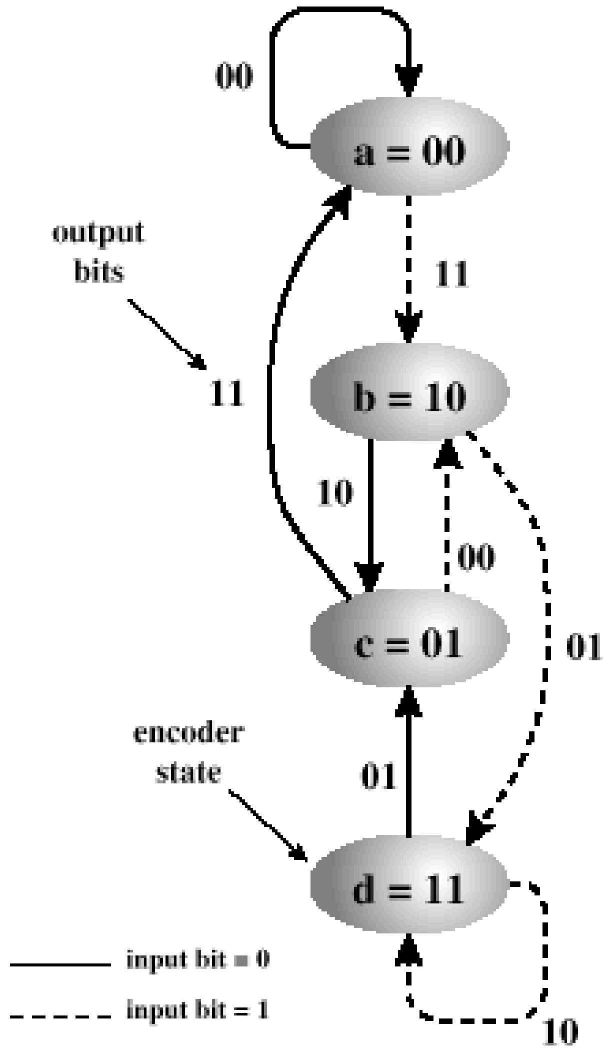
Convolutional Codes

- Generates redundant bits continuously
- Error checking and correcting carried out continuously
  - (n, k, K) code
    - Input processes k bits at a time
    - Output produces n bits for every k input bits
    - K = constraint factor
    - k and n generally very small
  - n-bit output of (n, k, K) code depends on:
    - Current block of k input bits
    - Previous K-1 blocks of k input bits



input bits
u_n
u_{n-1}
u_{n-2}
$v_{n1} = u_{n-2} \oplus u_{n-1} \oplus u_n$
output bits
$v_{n2} = u_{n-2} \oplus u_n$

(a) Encoder shift register



(b) Encoder state diagram

Figure 8.9 Convolutional Encoder with $(n, k, K) = (2, 1, 3)$

Decoding

- Trellis diagram – expanded encoder diagram
- Viterbi code – error correction algorithm
  - Compares received sequence with all possible transmitted sequences
  - Algorithm chooses path through trellis whose coded sequence differs from received sequence in the fewest number of places
  - Once a valid path is selected as the correct path, the decoder can recover the input data bits from the output code bits

Automatic Repeat Request

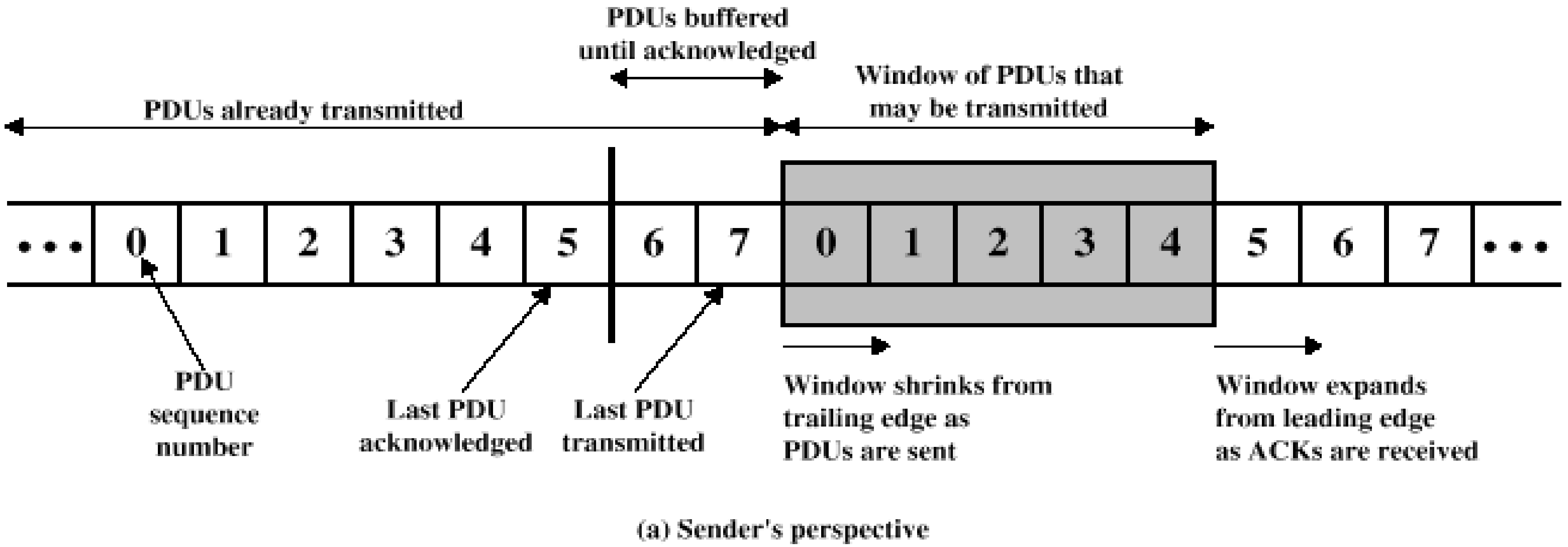
- Mechanism used in data link control and transport protocols
- Relies on use of an error detection code (such as CRC)
- Flow Control
- Error Control

Flow Control

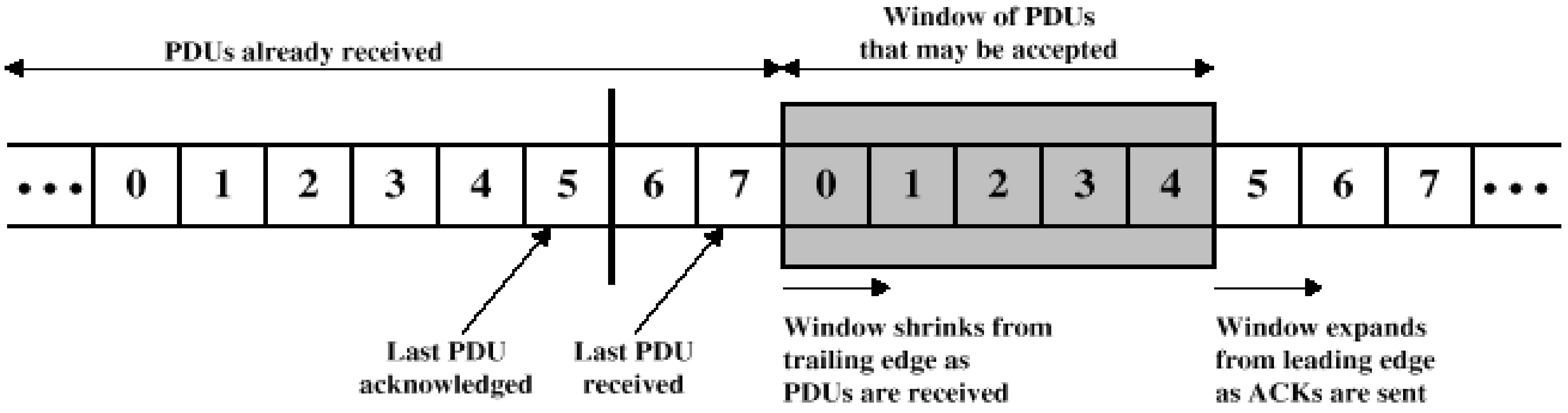
- Assures that transmitting entity does not overwhelm a receiving entity with data
- Protocols with flow control mechanism allow multiple PDUs in transit at the same time
- PDUs arrive in same order they are sent
- Sliding-window flow control:
  - Transmitter maintains list (window) of sequence numbers allowed to send
  - Receiver maintains list allowed to receive

Flow Control

- Reasons for breaking up a block of data before transmitting:
  - Limited buffer size of receiver
  - Retransmission of PDU due to error requires smaller amounts of data to be retransmitted
  - On shared medium, larger PDUs occupy medium for extended period, causing delays at other sending stations


PDUs buffered until acknowledged
Window of PDUs that may be transmitted
PDUs already transmitted
• • • 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 • • •
PDU sequence number
Last PDU acknowledged
Last PDU transmitted
Window shrinks from trailing edge as PDUs are sent
Window expands from leading edge as ACKs are received

(a) Sender's perspective


Window of PDUs that may be accepted
PDUs already received
• • • 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 • • •
Last PDU acknowledged
Last PDU received
Window shrinks from trailing edge as PDUs are received
Window expands from leading edge as ACKs are sent

(b) Receiver's perspective

Figure 8.17 Sliding-Window Depiction

Sliding Window Protocol

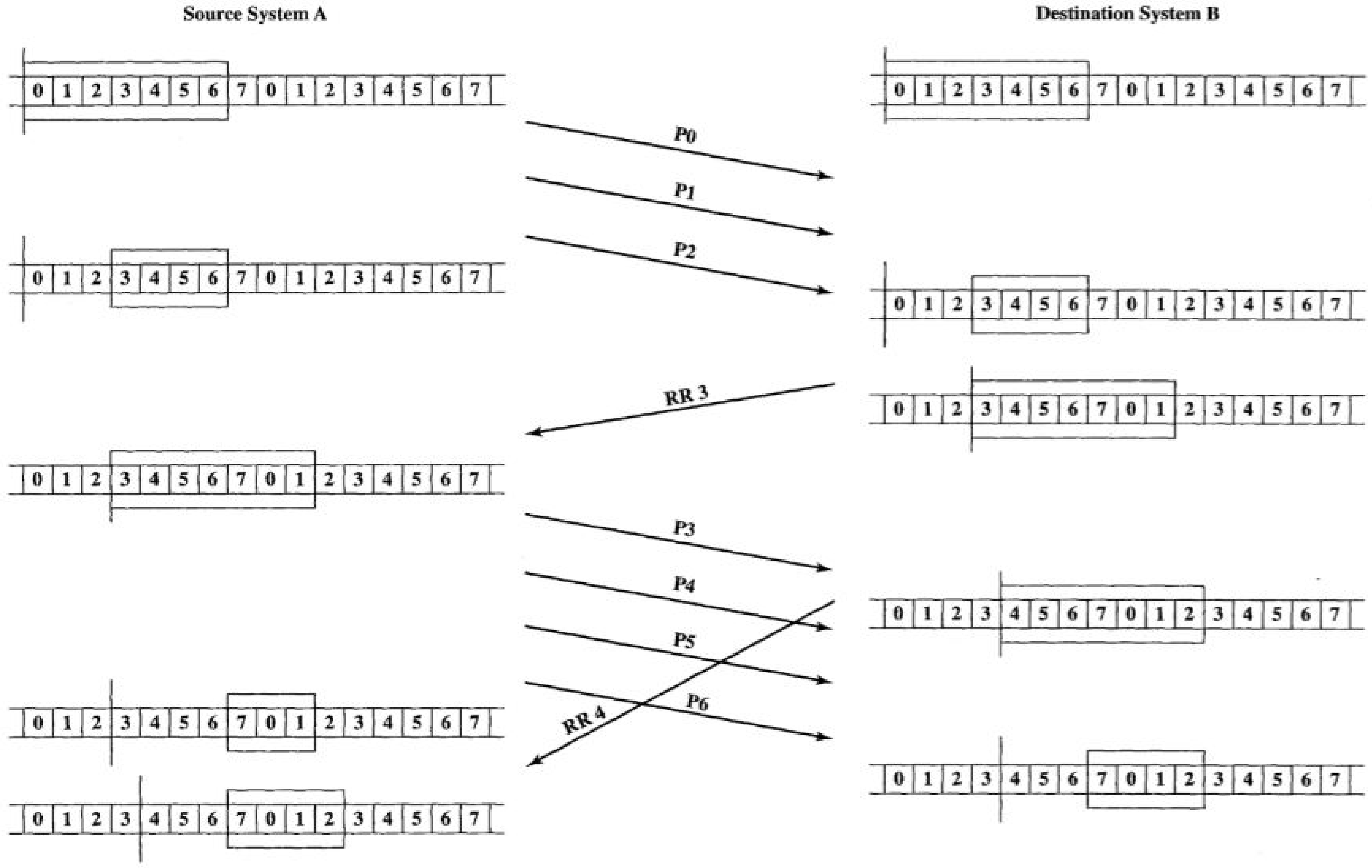


Figure 8.18 Example of a Sliding-Window Protocol

Error Control

- Mechanisms to detect and correct transmission errors

- Types of errors:
  - Lost PDU : a PDU fails to arrive
  - Damaged PDU : PDU arrives with errors

Error Control Requirements

- Error detection
  - Receiver detects errors and discards PDUs
- Positive acknowledgement
  - Destination returns acknowledgment of received, error-free PDUs
- Retransmission after timeout
  - Source retransmits unacknowledged PDU
- Negative acknowledgement and retransmission
  - Destination returns negative acknowledgment to PDUs in error

Go-back-N ARQ

- Acknowledgments
  - RR = receive ready (no errors occur)
  - REJ = reject (error detected)
- Contingencies
  - Damaged PDU
  - Damaged RR
  - Damaged REJ

Go-back-N ARQ

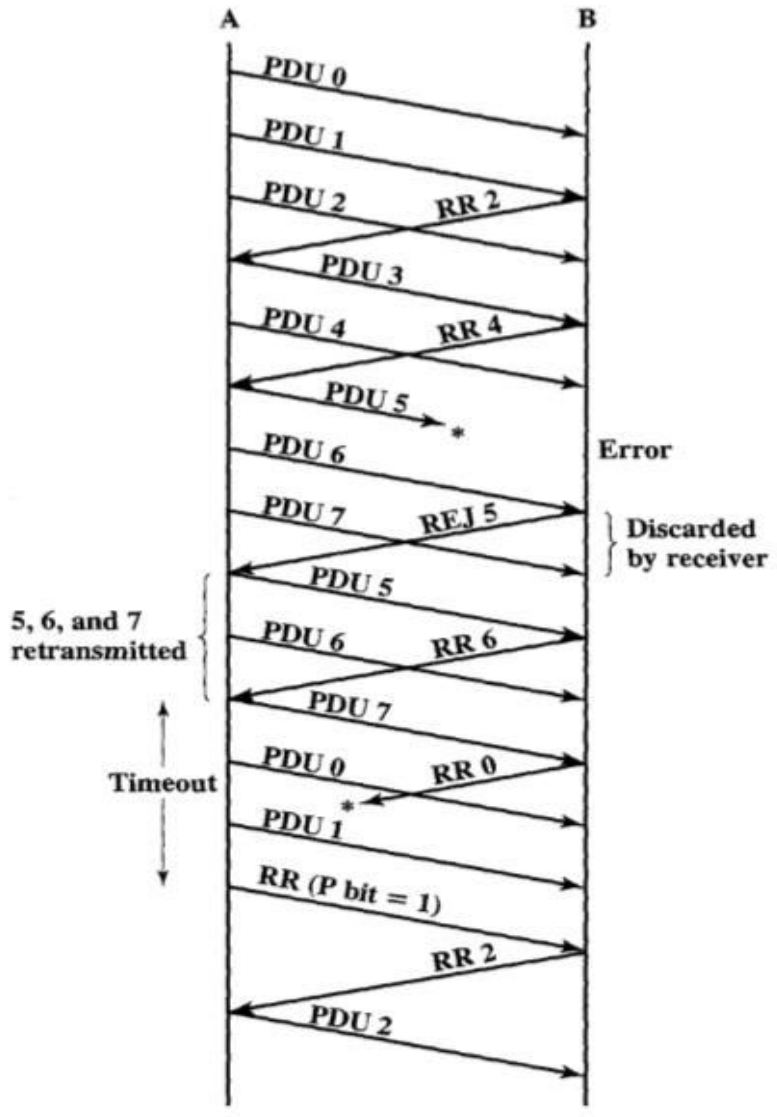


Figure 8.19 Go-back-N ARQ