CHAPTER 10

Error Detection and Correction

Solutions to Odd-numbered Review Questions and Exercises

Review Questions

1. In a ***single bit error*** only one bit of a data unit is corrupted; in a **burst error** more than one bit is corrupted (not necessarily contiguous).
3. In ***forward error correction***, the receiver tries to correct the corrupted codeword; in ***error detection by retransmission***, the corrupted message is discarded (the sender needs to retransmit the message).
5. The ***Hamming distance*** between two words (of the same size) is the number of differences between the corresponding bits. The Hamming distance can easily be found if we apply the XOR operation on the two words and count the number of 1s in the result. The ***minimum Hamming distance*** is the smallest Hamming distance between all possible pairs in a set of words.
7.
   a. The only relationship between the size of the codeword and dataword is the one based on the definition: **n = k + r**., where ***n*** is the size of the codeword, ***k*** is the size of the dataword, and ***r*** is the size of the remainder.

   b. The ***remainder*** is always ***one bit smaller*** than the ***divisor***.

   c. The ***degree*** of the generator polynomial is ***one less than*** the size of the ***divisor***. For example, the CRC-32 generator (with the polynomial of degree 32) uses a 33-bit divisor.

   d. The ***degree*** of the generator polynomial is the ***same as*** the size of the remainder (length of checkbits). For example, CRC-32 (with the polynomial of degree 32) creates a remainder of 32 bits.
9. ***At least three types of error*** cannot be detected by the current checksum calculation. First, if two data items are swapped during transmission, the sum and the checksum values will not change. Second, if the value of one data item is increased (intentionally or maliciously) and the value of another one is decreased (intentionally or maliciously) the same amount, the sum and the checksum cannot detect these changes. Third, if one or more data items is changed in such a way that the change is a multiple of $2^{16} – 1$, the sum or the checksum cannot detect the changes.

Exercises

11. We can say that **(vulnerable bits) = (data rate) × (burst duration)**

a.	vulnerable bits	$= (1{,}500) \times (2 \times 10^{-3})$	= **3 bits**
b.	vulnerable bits	$= (12 \times 10^{3}) \times (2 \times 10^{-3})$	= **24 bits**
c.	vulnerable bits	$= (100 \times 10^{3}) \times (2 \times 10^{-3})$	= **200 bits**
d.	vulnerable bits	$= (100 \times 10^{6}) \times (2 \times 10^{-3})$	= **200,000 bits**

Comment: The last example shows how a noise of small duration can affect so many bits if the data rate is high.

13. The codeword for dataword **10** is **101**. This codeword will be changed to **010** if a 3-bit burst error occurs. This pattern is not one of the valid codewords, so the receiver detects the error and discards the received pattern.

15.
 a. d (10000, 00000) = **1**
 b. d (10101, 10000) = **2**
 c. d (1111, 1111) = **0**
 d. d (000, 000) = **0**

Comment: Part c and d show that the distance between a codeword and itself is 0.

17.
 a. **01**
 b. **error**
 c. **00**
 d. **error**

19. We check five random cases. All are in the code.

I.	(1st)	$\oplus$	(2nd)	=	(2nd)
II.	(2nd)	$\oplus$	(3th)	=	(4th)
III.	(3rd)	$\oplus$	(4th)	=	(2nd)
IV.	(4th)	$\oplus$	(5th)	=	(8th)
V.	(5th)	$\oplus$	(6th)	=	(2nd)

21. We show the dataword, codeword, the corrupted codeword, the syndrome, and the interpretation of each case:
 a. Dataword: 0100 → Codeword: 0100011 → Corrupted: **1100011** → $s_2s_1s_0$ = 110
 Change b_3 (Table 10.5) → Corrected codeword: **0100011** → dataword: 0100
 The dataword is correctly found.
 b. Dataword: 0111 → Codeword: 0111001 → Corrupted: **0011001** → $s_2s_1s_0$ = 011
 Change b_2 (Table 10.5) → Corrected codeword: **0111001**→ dataword: 0111
 The dataword is correctly found.
 c. Dataword: 1111 → Codeword: 1111111 → Corrupted: **0111110** → $s_2s_1s_0$ = 111
 Change b_1 (Table 10.5) → Corrected codeword: **0101110**→ dataword: 0101
 The dataword is found, but it is **incorrect**. C(7,4) cannot correct two errors.

d. Dataword: 0000 $\rightarrow$ Codeword: 0000000 $\rightarrow$ Corrupted: **1100001** $\rightarrow$ $s_2s_1s_0 = 100$ Change q_2 (Table 10.5) $\rightarrow$ Corrected codeword: **1100101** $\rightarrow$ dataword: 1100 The dataword is found, but it is **incorrect**. C(7,4) cannot correct three errors.

23. We need to find $k = 2^m - 1 - m \geq 11$. We use *trial and error* to find the right answer:

a. Let m = 1 $\quad k = 2^m - 1 - m = 2^1 - 1 - 1 = 0$ (not acceptable)

b. Let m = 2 $\quad k = 2^m - 1 - m = 2^2 - 1 - 2 = 1$ (not acceptable)

c. Let m = 3 $\quad k = 2^m - 1 - m = 2^3 - 1 - 3 = 4$ (not acceptable)

d. Let m = 4 $\quad k = 2^m - 1 - m = 2^4 - 1 - 4 = 11$ (acceptable)

Comment: The code is **C(15, 11)** with **$d_{min} = 3$**.

25.

a. 101110 $\rightarrow$ $x^5 + x^3 + x^2 + x$

b. 101110 $\rightarrow$ 101110**000** (Three 0s are added to the right)

c. $x^3 \times (x^5 + x^3 + x^2 + x) = x^8 + x^6 + x^5 + x^4$

d. 101110 $\rightarrow$ 10 (The four rightmost bits are deleted)

e. $x^{-4} \times (x^5 + x^3 + x^2 + x) = x$ (Note that negative powers are deleted)

27. CRC-8 generator is $x^8 + x^2 + x + 1$.

a. It has more than one term and the coefficient of x^0 is 1. It can detect a single-bit error.

b. The polynomial is of degree 8, which means that the number of checkbits (remainder) r = 8. It will detect all burst errors of size 8 or less.

c. Burst errors of size 9 are detected most of the time, but they slip by with probability $(1/2)^{r-1}$ or $(1/2)^{8-1} \approx 0.008$. This means **8 out of 1000** burst errors of size 9 are left undetected.

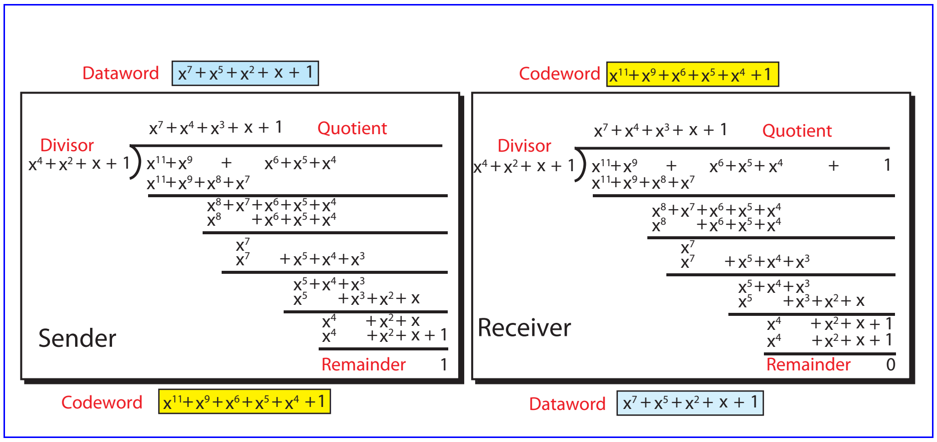
d. Burst errors of size 15 are detected most of the time, but they slip by with probability $(1/2)^r$ or $(1/2)^8 \approx 0.004$. This means **4 out of 1000** burst errors of size 15 are left undetected.

29. We need to add all bits modulo-2 (XORing). However, it is simpler to count the number of 1s and make them even by adding a 0 or a 1. We have shown the parity bit in the codeword in color and separate for emphasis.

	Dataword		Number of 1s		Parity	Codeword
a.	1001011	$\rightarrow$	4 (even)	$\rightarrow$	**0**	**0** 1001011
b.	0001100	$\rightarrow$	2 (even)	$\rightarrow$	**0**	**0** 0001100
c.	1000000	$\rightarrow$	1 (odd)	$\rightarrow$	**1**	**1** 1000000
d.	1110111	$\rightarrow$	6 (even)	$\rightarrow$	**0**	**0** 1110111

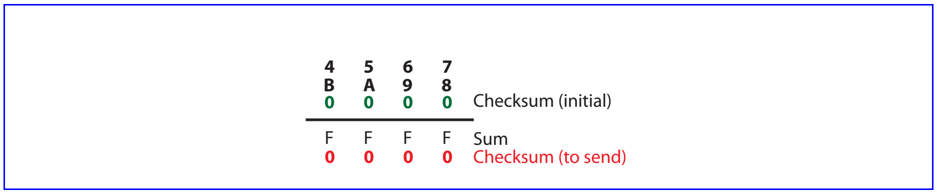
31. Figure 10.1 shows the generation of the codeword at the sender and the checking of the received codeword at the receiver using polynomial division.

Figure 10.1 *Solution to Exercise 31*


Dataword $x^7+x^5+x^2+x+1$
Codeword $x^{11}+x^9+x^6+x^5+x^4+1$
Divisor x^4+x^2+x+1
Quotient $x^7+x^4+x^3+x+1$
Sender
Remainder 1
Codeword $x^{11}+x^9+x^6+x^5+x^4+1$
Receiver
Remainder 0
Dataword $x^7+x^5+x^2+x+1$

33. Figure 10.2 shows the checksum to send (0x0000). This example shows that the checksum ***can be all 0s***. ***It can be all 1s only if all data items are all 0, which means no data at all***.

Figure 10.2 *Solution to Exercise 33*


4 5 6 7
B A 9 8
0 0 0 0 Checksum (initial)
F F F F Sum
0 0 0 0 Checksum (to send)