CHAPTER 10 : Error Detection and Correction

Solutions to Review Questions

Review Questions

1.

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.

2. ***The value of a checksum can be all 0s*** (in binary). This happens when the value of the sum (after wrapping) becomes all 1s (in binary). ***It is almost impossible for the value of a checksum to be all 1s***. For this to happen, the value of the sum (after wrapping) must be all 0s which means all data units must be 0s.

3. ***Redundancy*** is a technique of adding extra bits to each data unit to determine the accuracy of transmission.

4. 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.

5. 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).

6. ***One's complement arithmetic*** is used to add data items in checksum calculation. In this arithmetic, when a number needs more than *n* bits, the extra bits are wrapped and added to the number. In this arithmetic, the complement of a number is made by inverting all bits.

7. A ***linear block code*** is a block code in which the exclusive-or of any two codewords results in another codeword. A ***cyclic code*** is a linear block code in which the rotation of any codeword results in another codeword.

8. The ***single parity check*** uses one redundant bit for the whole data unit. In a ***twodimensional parity check***, original data bits are organized in a table of rows and columns. The parity bit is then calculated for each column and each row.

9. 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).

10. ***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 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)

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.

14. The codeword for dataword **10** is **10101**. This codeword will be changed to **01001** if a 3-bit burst error occurs. This pattern is not one of the valid codewords, so the receiver discards the received pattern.

16. 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.

17.

a. If we rotate **0101100** one bit, the result is **0010110**, which is in the code. If we rotate **0101100** two bits, the result is **0001011**, which is in the code. And so on.

b. The XORing of the two codewords (0010110) ___ (1111111) = 1101001, which is in the code.

22.

Comment: The above shows three properties of the exclusive-or operation. First, the result of XORing two equal patterns is an all-zero pattern (part b). Second, the result of XORing of any pattern with an all-zero pattern is the original non-zero pattern (part c). Third, the result of XORing of any pattern with an all-one pattern is the complement of the original non-one pattern.

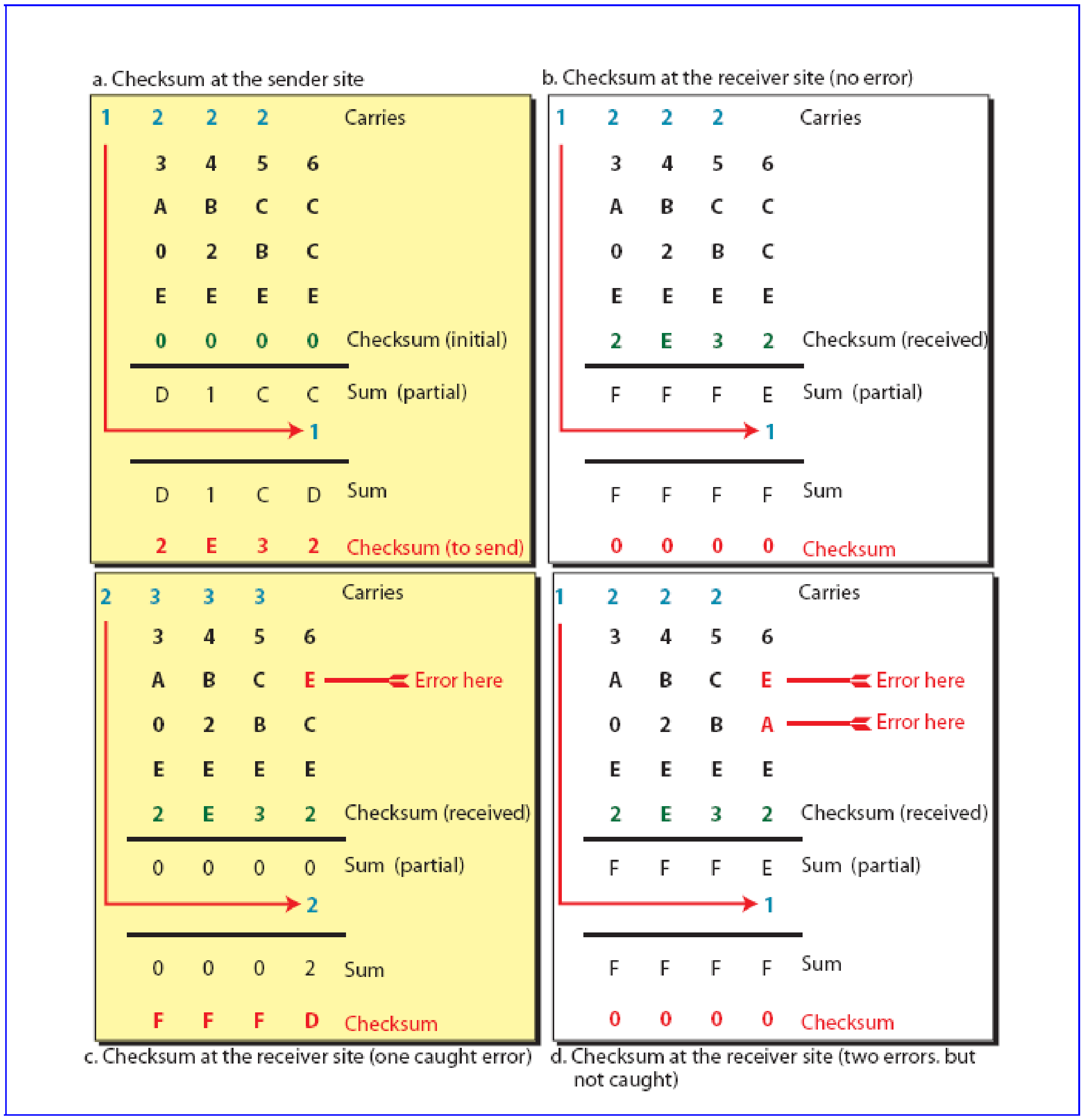
27.

a. For error detection $\rightarrow$ $d_{min} = s + 1 = 2 + 1 = 3$
b. For error correction $\rightarrow$ $d_{min} = 2t + 1 = 2 \times 2 + 1 = 5$
c.
   For error section $\rightarrow$ $d_{min} = s + 1 = 3 + 1 = 4$
   For error correction $\rightarrow$ $d_{min} = 2t + 1 = 2 \times 2 + 1 = 5$
   Therefore d_{min} should be 5.
d.
   For error detection $\rightarrow$ $d_{min} = s + 1 = 6 + 1 = 7$
   For error correction $\rightarrow$ $d_{min} = 2t + 1 = 2 \times 2 + 1 = 5$
   Therefore d_{min} should be 7.

30. Figure 10.3 shows the four situations.

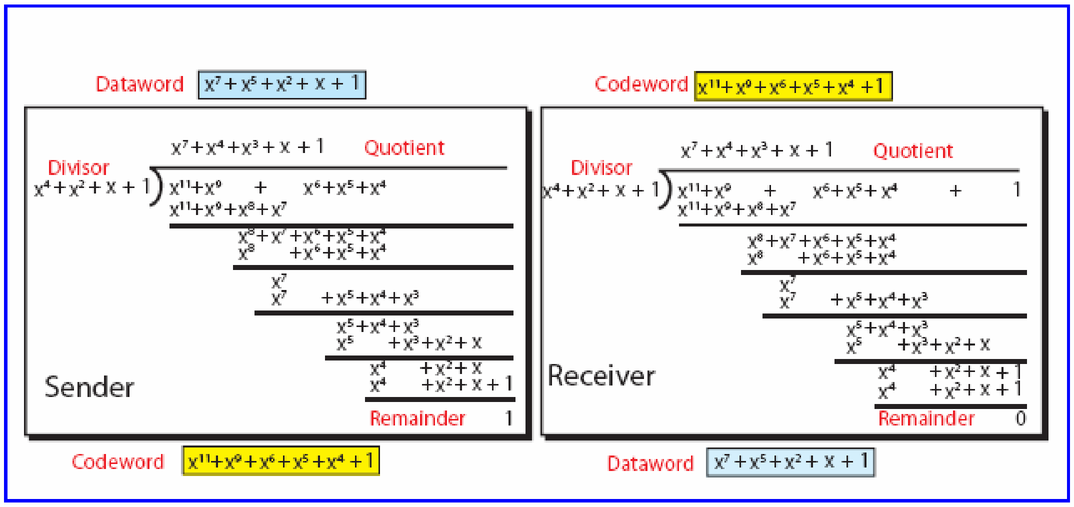
a. In part a, we calculate the checksum to be sent (0x2E32)
b. In part b, there is no error in transition. The receiver recalculates the checksum to be all 0x0000. The receiver correctly assumes that there is no error.
c. In part c, there is one single error in transition. The receiver calculates the checksum to be 0FFFD. The receiver correctly assumes that there is some error and discards the packet.
d. In part d, there are two errors that cancel the effect of each other. The receiver calculates the checksum to be 0x0000. The receiver erroneously assumes that there is no error and accepts the packet. This is an example that shows that the checksum may slip in finding some types of errors.

Figure 10.3 *Solution to Exercise 30*



32. 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.
33. Figure 10.2 shows the generation of the codeword at the sender and the checking of the received codeword at the receiver using polynomial division.

Figure 10.2 *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 $x^7+x^5+x^2+x+1$