Computer Network

Lecture-17

Dharmendra Kumar (Associate Professor)
Department of Computer Science and Engineering
United College of Engineering and Research,
Prayagraj

Unit-2

Data Link Control

- The two main functions of the data link layer are data link control and media access control. The first, data link control, deals with the design of procedures for communication between two adjacent nodes: node-to-node communication. The second function of the data link layer is media access control, or how to share the link.

- Data link control functions include framing, flow and error control protocols that provide smooth and reliable transmission of frames between nodes.

Data Link Layer

Error Detection and Correction

Error Detection and Correction

Single bit error

In single bit error, only 1 bit in the data unit has changed.

Burst error

A burst error means that 2 or more bits in the data unit have changed.

Redundancy

To detect or correct errors, we need to send extra bits with data.

Block Coding

In block coding, we divide our message into blocks, each of k bits, called datawords. We add r redundant bits to each block to make the length n = k + r. The resulting n-bit blocks are called codewords.

Error Detection and Correction

Hamming Distance

- 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 1's in the result.

Example

Find the Hamming distance between the following words:-

a = 10101110 and b = 01010100

Minimum Hamming Distance

The minimum Hamming distance is the smallest Hamming distance between all possible pairs. We use d_{min} to define the minimum Hamming distance in a coding scheme.

Example

Find the minimum Hamming distance for the following set of words:-

{ 00000, 10101, 01011, 11110 }.

Error Detection and Correction

Minimum Hamming Distance for Error Detection

To guarantee the detection of up to **s** errors in all cases, the minimum Hamming distance in a block code must be

$$d_{min} = s + 1.$$

Minimum Hamming Distance for Error Correction

To guarantee the correction of up to t errors in all cases, the minimum Hamming distance in a block code must be

$$d_{min} = 2t + 1.$$

Error Detection and Correction

Simple Parity-Check Code

- In this code, a k-bit dataword is changed to an k+1-bit codeword.
- The extra bit, called the parity bit.
- It is selected to make the total number of 1’s in the codeword even.

Note:

A simple parity-check code is a single-bit error-detecting code in which $n = k + 1$ with $d_{min} = 2$.

Error Detection and Correction

Hamming Code

Hamming code is a set of error-correction codes that can be used to **detect and correct the errors** that can occur when the data is moved or stored from the sender to the receiver.

It is **technique developed by R.W. Hamming for error correction**.

Redundant bits –

The number of redundant bits can be calculated using the following formula:

$$2^r >= m+r+1$$

Where, r = number of redundant bits, and

m = number of data bits

Error Detection and Correction

Algorithm of Hamming code

1. Write the bit positions starting from 1.
2. All the bit positions that are a power of 2 are marked as parity bits (1, 2, 4, 8, etc).
3. All the other bit positions are marked as data bits.
4. Each data bit is included in a unique set of parity bits, as determined its bit position in binary form.
   - Parity bit 1 covers all the bits positions whose binary representation includes a 1 in the least significant position (1, 3, 5, 7, 9, 11, etc).
   - Parity bit 2 covers all the bits positions whose binary representation includes a 1 in the second position from the least significant bit (2, 3, 6, 7, 10, 11, etc).
   - Parity bit 4 covers all the bits positions whose binary representation includes a 1 in the third position from the least significant bit (4–7, 12–15, 20–23, etc).
   - Parity bit 8 covers all the bits positions whose binary representation includes a 1 in the fourth position from the least significant bit bits (8–15, 24–31, 40–47, etc).

Error Detection and Correction

- In general, each parity bit covers all bits where the bitwise AND of the parity position and the bit position is non-zero.

5. Set a parity bit to 1 if the total number of ones in the positions it checks is odd.
6. Set a parity bit to 0 if the total number of ones in the positions it checks is even.

Error Detection and Correction

Ex. Construct an even parity Hamming code word for a data word 1011001.

Solution:

Step-1: First we compute the number of redundant bits r.

Here, number of bits in the given data word (1011001), m = 7

Therefore, we compute r as following:-

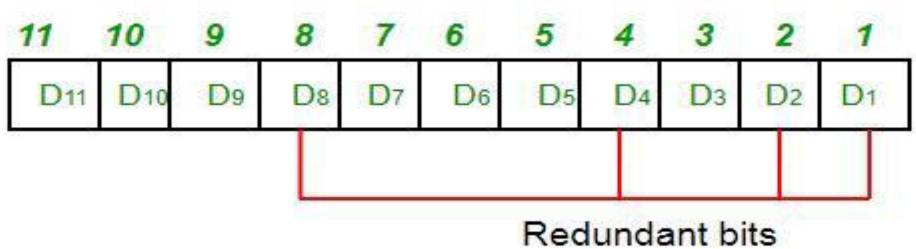
$$2^r >= m+r+1 \Rightarrow 2^r >= 7+r+1$$

Minimum value of r which satisfies above inequality = 4.

Therefore, r=4.

Step-2: Now, we compute the position of redundant bits in the codeword.

These redundancy bits are placed at positions that correspond to the power of 2. Therefore, the position these redundant bits will be 1, 2, 4 and 8.



Error Detection and Correction

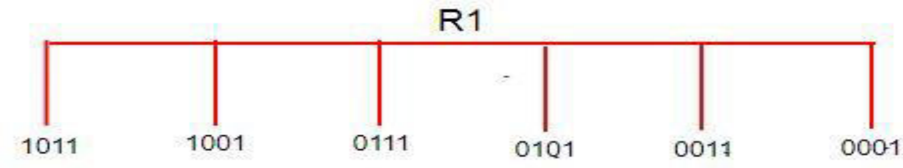
Step-3: Now, we compute the codeword.

Since the data to be transmitted is 1011001, therefore the bits will be placed as follows:

11	10	9	8	7	6	5	4	3	2	1
1	0	1	R8	1	0	0	R4	1	R2	R1

Determining the Parity bits:

R1 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the least significant position i.e. positions 1, 3, 5, 7, 9, 11.

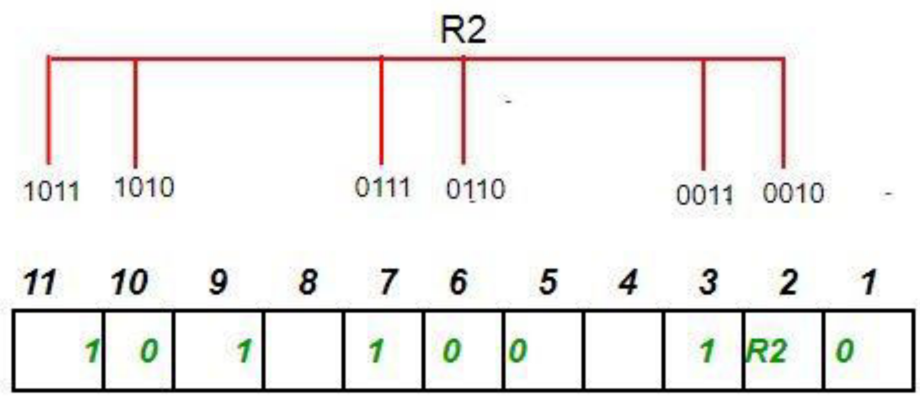


11	10	9	8	7	6	5	4	3	2	1
1	0	1		1	0	0		1		R1

To find the redundant bit R1, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R1 is an even number the value of R1 (parity bit's value) = **0**

Error Detection and Correction

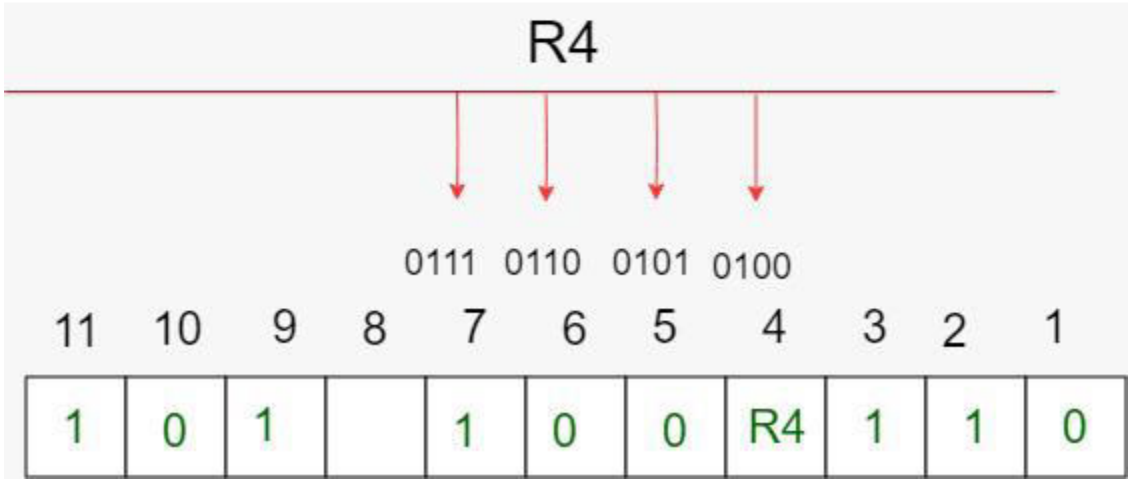
R2 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the second position from the least significant bit. R2: bits 2,3,6,7,10,11.



To find the redundant bit R2, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R2 is odd the value of R2(parity bit's value)**=1**

Error Detection and Correction

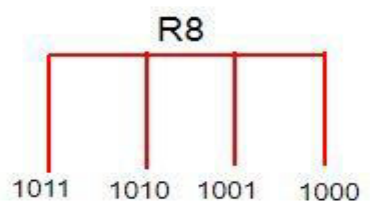
R4 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the third position from the least significant bit. R4: bits 4, 5, 6, 7.



To find the redundant bit R4, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R4 is odd the value of R4(parity bit's value) **= 1.**

Error Detection and Correction

R8 bit is calculated using parity check at all the bits positions whose binary representation includes a 1 in the fourth position from the least significant bit. R8: bit 8, 9, 10, 11.



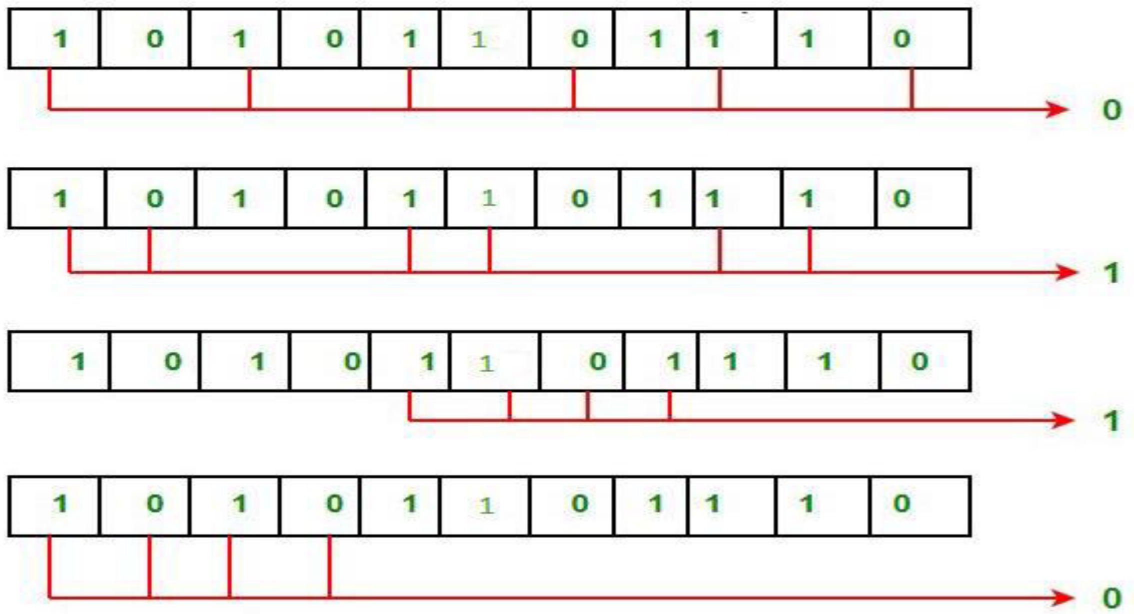
11	10	9	8	7	6	5	4	3	2	1
1	0	1	R8	1	0	0	1	1	1	0

To find the redundant bit R8, we check for even parity. Since the total number of 1's in all the bit positions corresponding to R8 is an even number the value of R8(parity bit's value)**= 0**. **Thus, the data transferred is:**

11	10	9	8	7	6	5	4	3	2	1
1	0	1	0	1	0	0	1	1	1	0

Error Detection and Correction

Error detection and correction: Suppose in the above example the 6th bit is changed from 0 to 1 during data transmission, then it gives new parity values in the binary number:



The bits give the binary number **0110** whose decimal representation is **6.** Thus, bit **6** contains an error. **To correct the error** the 6th bit is changed from **1 to 0.**