**U19EC046 | CN | LAB 4**

Hamming.java

**public class hamming {**

**static void print(int arr[]) {**

**for (int i=arr.length-1; i>0; i--) {**

**System.out.print(arr[i] + " ");**

**}**

**System.out.println();**

**}**

**static boolean isParityBit(int i) {**

**double N = Math.log(i) / Math.log(2);**

**int X = (int) N;**

**double temp = N - X;**

**return temp > 0 ? false : true;**

**}**

**static int redundantBitsCount(int n) {**

**int r = 1;**

**while (Math.pow(2, r) < (n + r + 1)) {**

**r++;**

**}**

**return r;**

**}**

**static int[] getDataParityArray(String str, int M, int r) {**

**int[] ar = new int[r + M + 1];**

**int j = M;**

**for (int i = 1; i < ar.length; i++) {**

**ar[i] = isParityBit(i) ? 0 : (int) (str.charAt(--j) - '0');**

**}**

**return ar;**

**}**

**static int[] hammingGenerator(String str) {**

**int M = str.length();**

**int r = redundantBitsCount(M);**

**int[] ar = getDataParityArray(str, M, r);**

**for (int i = 0; i < r; i++) {**

**int x = (int) Math.pow(2, i);**

**for (int j = 1; j < ar.length; j++) {**

**if (((j >> i) & 1) == 1) {**

**if (x != j)**

**ar[x] = ar[x] ^ ar[j];**

**}**

**}**

**System.out.println("P" + x + " = " + ar[x]);**

**}**

**return ar;**

**}**

**public static void main(String[] args) {**

***// input message***

**String str = "1001101";**

**System.out.println("Generated hamming code ");**

**int[] ar = hammingGenerator(str);**

**print(ar);**

**}**

**}**

Output:

Generated hamming code

P1 = 1

P2 = 0

P4 = 0

P8 = 1

1 0 0 1 1 1 0 0 1 0 1

**Example Scenario**

Suppose a binary data 1001101 is to be transmitted. To implement hamming code for this, following steps are used:

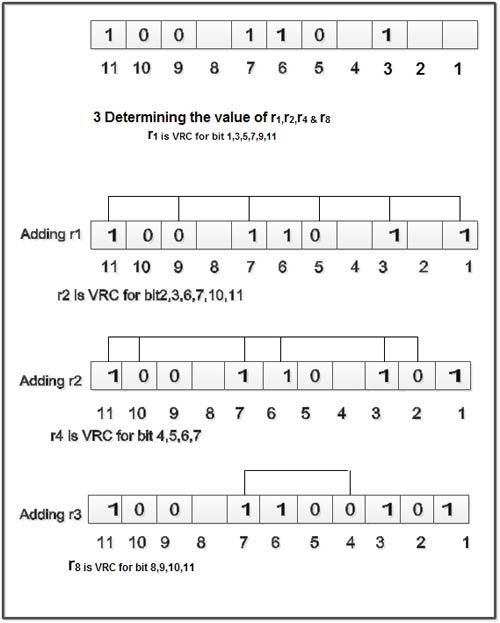
1. Calculating the number of redundancy bits required. Since number of data bits is 7, the value of r is calculated as

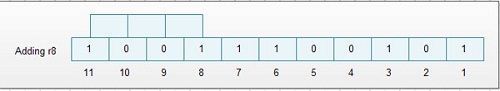
2r > m + r + 1

24 > 7 + 4 + 1

Therefore no. of redundancy bits = 4

2. Determining the positions of various data bits and redundancy bits. The various r bits are placed at the position that corresponds to the power of 2 *i.e.* 1, 2, 4, 8

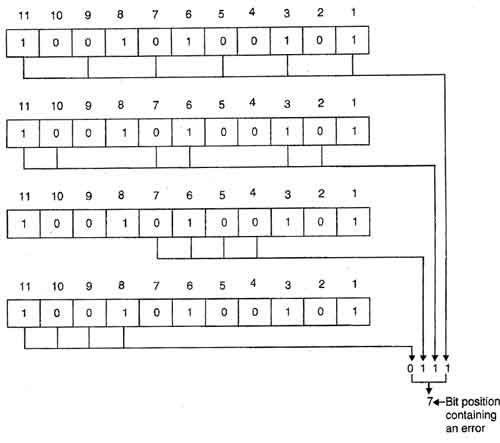




4. Thus data 1 0 0 1 1 1 0 0 1 0 1 with be transmitted.

## **Error Detection & Correction**

Considering a case of above discussed example, if bit number 7 has been changed from 1 to 0.The data will be erroneous.



Data sent: 1 0 0 1 1 1 0 0 1 0 1

Data received: 1 0 0 1 0 1 0 0 1 0 1 (seventh bit changed)

The receive takes the transmission and recalculates four new VRCs using the same set of bits used by sender plus the relevant parity (r) bit for each set as shown in fig.

Then it assembles the new parity values into a binary number in order of r position (r8, r4, *r2,*r1).

In this example, this step gives us the binary number 0111. This corresponds to decimal 7. Therefore bit number 7 contains an error. To correct this error, bit 7 is reversed from 0 to 1.