**Assignment No.3**

## Title:

Write a program for error detection and correction for 7/8 bits ASCII codes using Hamming Codes or CRC. Demonstrate the packets captured traces using Wireshark Packet Analyzer Tool for peer to peer mode.( 50% students will perform Hamming Code and others will perform CRC) (Use C/C++)

## 1. Prerequisite:

1. Data Link Layer: Roles, Protocols(Ethernet)
2. C/C++ Programming Syntax
3. Wireshark Tool

## Learning Objectives:

* + Students will able to understand Data Link Layer of OSI Model
  + Students will able to understand hamming code and CRC.

## Theory

**Computer Networks Error Detection and Correction**

A condition when the receiver’s information does not matches with the sender’s information. During transmission, digital signals suffer from noise that can introduce errors in the binary bits traveling from sender to receiver. That means a 0 bit may change to 1 or a 1 bit may change to 0.

Some popular techniques for error detection are:

### Simple Parity check

1. Two-dimensional Parity check
2. Checksum

### Cyclic redundancy check

1. **Hamming Codes**

Hamming code is a set of error-correction codes that can be used to detect and correct [bit](http://searchcio-midmarket.techtarget.com/definition/bit) errors that can occur when computer data is moved or stored. Like other error-correction code, Hamming code makes use of the concept of [parity](http://searchstorage.techtarget.com/definition/parity) and [parity bit](http://searchstorage.techtarget.com/definition/parity)s, which are bits that are added to data so that the validity of the data can be checked when it is read or after it has been received in a data transmission. Using more than one parity bit, an error-correction code can not only identify a single bit error in the data unit, but also its location in the data unit.

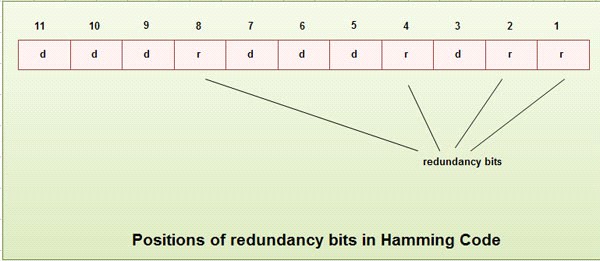
### Calculating the Hamming Code

**Determining the positions of redundancy bits**

We know that to detect errors in a 7 bit code, 4 redundant bits are required.

Now, the next task is to determine the positions at which these redundancy bits will be placed within the data unit.

* + These redundancy bits are placed at the positions which correspond to the power of 2.
  + For example in case of 7 bit data, 4 redundancy bits are required, so making total number of bits as 11. The redundancy bits are placed in position 1, 2, 4 and 8 as shown in fig.



* In Hamming code, each r bit is the VRC for one combination of data bits. rl is the VRC bit for one combination of data bits, r2 is the VRC for another combination of data bits and so on.
* Each data bit may be included in more than one VRC calculation.
* rl bit is calculated using all bits positions whose binary representation includes a 1 in the rightmost position.
* bit calculated using all the bit positions with a 1 in the second position and so on.
* Therefore the various r bits are parity bits for different combination of bits.

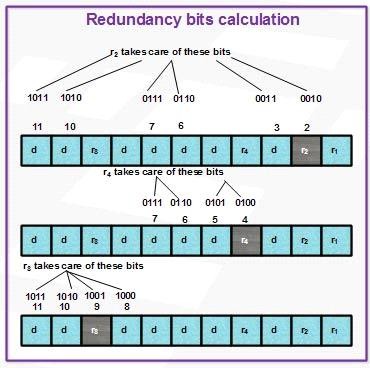
The various combinations are: rl : bits 1,3,5, 7, 9, 11

r2 : bits 2, 3, 6, 7, 10, 11

r4 : bits 4, 5, 6, 7

r8 : bits 8, 9, 10, 11

### Example of Hamming Code Generation



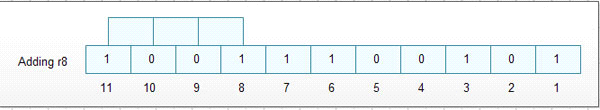
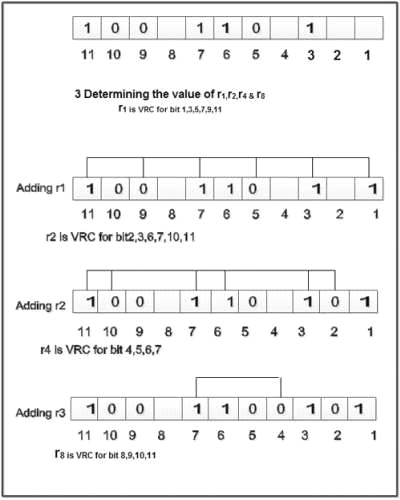
Suppose a binary data 1001101 is to be transmitted. To implement hamming code for this, following steps are used:

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

Therefore no. of redundancy bits = 4

2r > m + r + 1 24 > 7 + 4 + 1

1. 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

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

Data received: 1 00 1 0 1 00 1 0 1 (seventh bit changed)

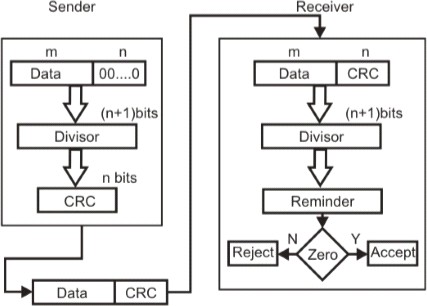
The receive takes the transmission and recalculates four new VRCs using the same set of bits used by sender plus the relevant parity (r) bit for each set as shown in fig.

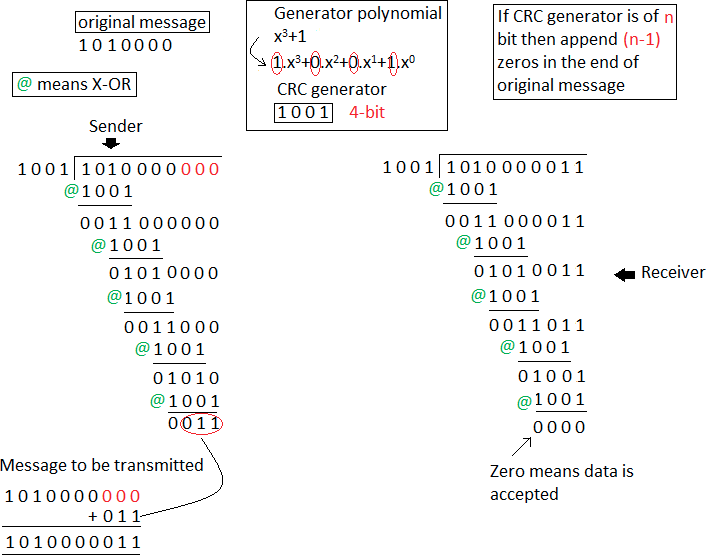
Then it assembles the new parity values into a binary number in order of r position (r8, r4, *r2,*

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.

### Cyclic redundancy check (CRC)

* + CRC is based on binary division.
  + In CRC, a sequence of redundant bits, called cyclic redundancy check bits, are appended to the end of data unit so that the resulting data unit becomes exactly divisible by a second, predetermined binary number.
  + At the destination, the incoming data unit is divided by the same number. If at this step there is no remainder, the data unit is assumed to be correct and is therefore accepted.
  + A remainder indicates that the data unit has been damaged in transit and therefore must be rejected.



## Conclusion:

Hence we have studied error detection and correction using Hamming Codes and CRC.