IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 5, No 1, September 2011
ISSN (Online): 1694-0814
www.IJCSI.org 

# Indication of Efficient Technique for Detection of Check Bits in Hamming Code

Rahat Ullah, Jahangir Khan, Shahid Latif, Inayat Ullah

Department of Computer Science and IT

Sarhad University of Science and Information Technology (SUIT), Peshawar 25000 Pakistan

**Abstract**

When the information is transmitted through wireless or through wired channel, error may occur in data. In some techniques error may be only detected while there are such techniques which can detect as well as correct that error in the data. Hamming code is such a code which can detect a single bit error position and also correct that, this coding scheme is used now days in Wireless communication, in deep space communication etc, e.g. Fixed wireless broadband with high error rate where there is need of correction not only detection. In this coding scheme the detection of data depend on the check/Parity bits, these bits play a vital role in hamming code because they locate the position where the error has been occurred. These check bits are inserted into the data at specified position at transmitter and also at receiver side to sort out error. And to find these bits we need to design a matrix and by putting bit's information and than the value of parity bit can be determine. Normally these bits can be found by using Ex-OR logic as well as using another-logic that is AND-OR logic. In this Paper I have sketched a general Idea about these two logics that which logics performance is good and efficient. In future I will prove that by performing simulations through PSpice and will find the delays of these mentioned logics.

***Keywords:*** *AND-OR Logic, Coding, Error Correction/Detection, Check Bits, Efficient Logic.*

## 1. Introduction

Error correcting codes are used in a wide range of communication systems from deep space communication, to quality of sound in compact disks and wireless phones. If we generally discuss the main method used to recover messages that might be distorted during transmission over a noisy channel is to inhabit redundancy. We make use of redundancy present in human languages to detect and than correct errors. This happens in both oral and written communication. For example, if you read the sentence "union is strenght " you can tell that something is wrong. So we can detect an error. Moreover we can even correct it. We are doing two things here: error detection and error correction. What are the principles that we are using to achieve these goals? First, because the string "strangth" is not a valid word in English, we know that there is an error. Here, the redundancy manifests itself in the form of the fact that not every possible string is a valid word in the language. Secondly, the word "strangth" is closest to the valid word "mistake" in the language, so we conclude that it is the most likely word intended. Of course we can also use the context and meaning to detect and/or correct errors but that is an additional feature, not available to computers. When I enter the string "strangthy" to Merriam-Webster online dictionary (http://www.m-w.com ), no entry can be found for it, however, the computer comes up with a list of suggested words, the first of which is "strength". So, the computer is telling me that "strength" is the most likely word to be intended, because it is closest to the given string. This is called the maximum likelihood principle. As I typed this article on my computer I witness many instances of this principle used by my word processor. For instance, when I mistakenly typed "thre" it automatically corrected it to "three". There are also other ways redundancy is used in natural languages. As by now pointed out, Redundancy in context often enables us to detect and correct errors.. When humans communicate, redundancy, either explicitly introduced by the speaker/author or built into the language, comes into play to help the audience understand the message better by overcoming such obstacles as hearing, difficulties, noise, accent, etc.

The basics of wireless communication were properly recognized by Claude E. Shannon in 1948 [1]. In his attractive paper, Shannon assert that by proper encoding of information, error induced by a noisy channel or storage medium can be reduced to any desired level when the data rate is bounded by channel capacity. Since then, many coding techniques have been developed for wireless communication systems to achieve channel capacity. In 1950, Richard W. Hamming discovered the first class of linear block codes for error correction, only two years after Shannon's theory. Hamming codes are 1-error-correcting codes and are capable of correcting any single error over the span of the code block length [2]. Hamming codes are perfect codes and can be decoded easily with a look-up-table [3].

Actually Coding theory is the branch of mathematics concerned with accurate and efficient transfer of data across noisy channels as the theory of message sent. A transmission channel is the physical medium through which the information is transmitted, such as telephone lines, or atmosphere in the case of wireless communication. Undesirable disturbance (noise) can occur across the communication channel, causing the received information to be different from the original information sent [4]. Coding theory deals with detection and correction of transmission errors caused by noise in the channel. The primary goal of coding theory is competent encoding of information, easy transmission of encoded messages, fast decoding of received information and

correction of errors introduced in the channel [5]. Coding theory is used all the time: in reading CDs, receiving transmissions from satellites, or in cell phones[6]. So as my paper is about hamming code, so will discuss that this technique is used to detect single bit error and will also indicate the efficient logic. And will use check bits for the detection of error, on receiver as well as on transmitter side. I have found these check bits by using Ex-OR logic as well as AND-OR logic.

This paper is organized in such away that the first portion explains the brief introduction about occurring of errors in the information and that how the concept of information theory and than coding techniques came into being, the second portion contains information about types of errors, third portion is about error exposure, fourth portion is about Hamming Code that how the error's position will be detected and corrected using the two logics, and the last section I have discuss the propagation delay of logic gates and explained that which logic should be used to find the parity/check bits and that my future goal is to do these simulations in PSpice.

## 2. Types of Errors

When binary data is transmitted and processed from source to destination, which can be altered due to channel's noise. Two types of errors could be possible. Single bit error and a burst error [7]. Single bit will be inverted in a single bit error and more than one bit inversion indicated burst errors[7].

## 3. Error Exposure

When we be on familiar terms with what type of error can situate in the data, will we be able to recognize one when we see it? If we have a copy of the projected message for transmission, than of course we will be able to identify. But if we have no copy than we will be incapable to find the error, until we decipher the data and find that it is of no significance, by making no sagacity. If we use this concept in the engine than it will be so costly, slow of dubious value. We don't need a system than decode whatever comes in, then sit around trying to make a decision if the engine really intended to use the word chshr in the middle of an array of whether information. What we need is a means that is simple and utterly objective.

### 2.1 Redundancy

In this technique extra bit is added with the Information.. Figure.1 shows that how the extra bits are added to the information to check the accuracy in the information at the receiving side. Once the data stream has been generated, it passes through a device that analyses it and the redundant bit are appropriately generates. The data is know enlarged by adding redundant check, and travels to the receiver. The receiver puts the entire stream to the checking function. If the received bit stream passes the checking criteria, the data portion of the data unit is accepted and the redundant bits are unnecessary and than discarded [7].

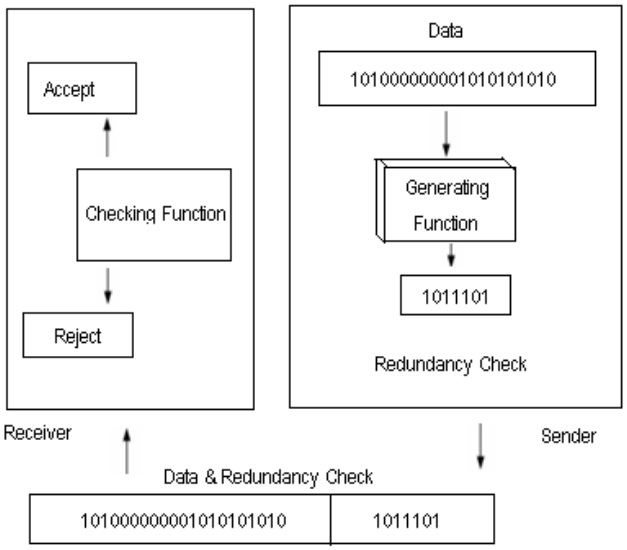



Fig. 1 Process of Using Redundant Bits

Four Types of Redundancy checks are used in data communication; Vertical Redundancy Check(VRC), Longitudinal Redundancy Check(LRC), Cyclic Redundancy Check(CRC), and Checksum.

## 4. Hamming Code

Data that is transmitted over a communication channel can be scratched; their bits can be masked or inverted by noise. To get rid of retransmitting the data we need to detect and correction of error in the data. Some simple codes can detect but not correct errors [7,8]; Others can detect and correct one or more errors[2]. This paper addresses one Hamming code that can correct a single-bit error which is detected with the help of parity/check bits using two logics and than indicated the efficient one. The Hamming single-bit correction code is implemented by adding check bits also called parity bits to the output message according to the following pattern:

i) The message bits are numbered from left to right, starting at 1.
ii) Every bit whose number is a power of 2 (bits 1, 2, 4, 8,…) is a check bit[8].
iii) The other are the output message bits (bits 3, 5, 6, 7, 9,…) contain the data bits, in order.
Each check bit establishes even parity over itself and a group of data bits and can be fine with the help of Ex-OR gate[9] or also by using AND-OR logic because both the circuits have same performance[10], figure.4. A data bit is in a check bit's group if the binary representation of the data bit's number contains a 1 in the position of the check bit's weight. For instance, the data bits associated with check bit 2 are all those with a 1 in the 2's position of their binary bit number—bits 2, 3, 6, 7, and so forth. Fig.2 shows an 8-bit data message. It shows the assignment of data bits and check bits to the bits of a12-bit output message. The horizontal lines below the output message box have dots to indicate the assignment of output

bits to check groups. Each check group has a ones count box, with a parity bit that is inserted into the output message..

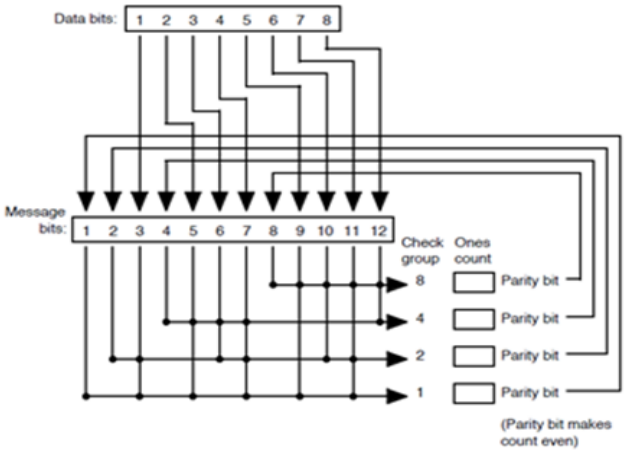



Fig.2 Indicating Positions for Parity bits

To find the value of check/parity bit, that which parity should be zero and which should be one, we will design a matrix having columns equal to the bits contained in the whole stream of data including parity bits according to the order given in the given figure. And will have rows equal to the number of parity bits used in the data, for example if we have used four parity bits, than it will have four rows, for five parity bits it will have five rows and so on.Fig.3 shows.

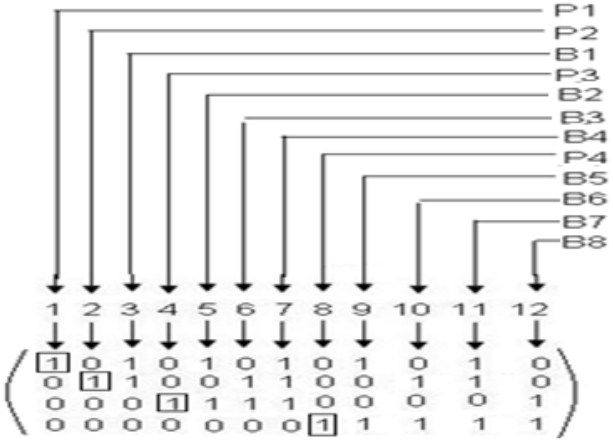



Fig.3 Designing Matrix for Parity bits

Inbox bits in Fig.5 are the starting bits for P1, P2, P3, P4. For P1 we have used 1010101….., for P2 , 1100110011….., for P3, 11110000111100., for P4, 11111111000000001111111100, and so on[16].

Now to calculate the values of P1, P2, P3, P4 from the matrix by Ex-OR or by using another logic AND-OR. Because as the figure.4 shows the outputs of both the circuits are same.

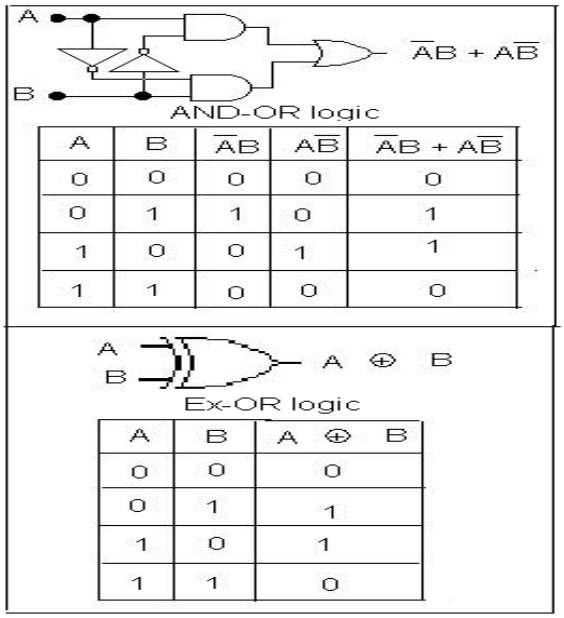
For first parity bit, from the matrix, we get the following information about the bits

$P1=3 \oplus 5 \oplus 7 \oplus 9 \oplus 11$

$=B1 \oplus B2 \oplus B4 \oplus B5 \oplus B7$

$=1 \oplus 0 \oplus 0 \oplus 1 \oplus 0$

After performing the two logics as fig.4 shows we got the parity bit “0”, as figure. 5 shows.




| A | B | $\overline{A}B$ | $A\overline{B}$ | $\overline{A}B + A\overline{B}$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |

| A | B | A ⊕ B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Fig.4 Showing the Two equivalent logics

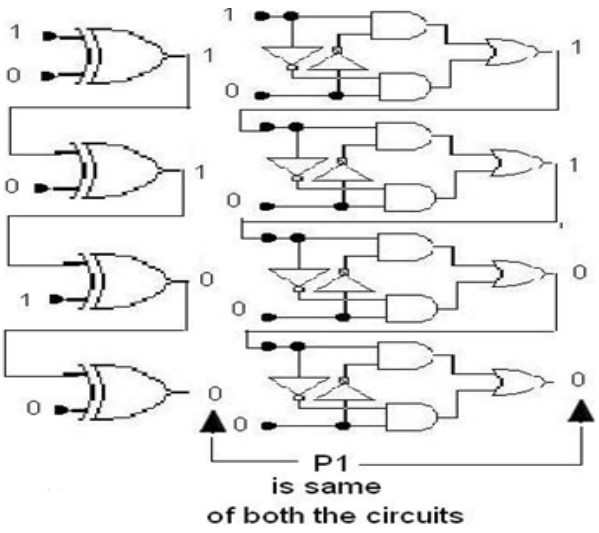



Fig. 5 Detection of First Parity bit

For second parity, from the matrix, we get the following information about the bits.

$P2=3 \oplus 6 \oplus 7 \oplus 10 \oplus 11$

$=B1 \oplus B3 \oplus B4 \oplus B6 \oplus B7$

$=1 \oplus 1 \oplus 0 \oplus 1 \oplus 0$

After performing the same logics we will get the parity bit "1".

For third parity bit, from the matrix, we get the following information.

$P3=5 \oplus 6 \oplus 7 \oplus 12$

$=B2 \oplus B3 \oplus B4 \oplus B8$

$=0 \oplus 1 \oplus 0 \oplus 1$

Here the resultant parity bit will be "0".

For the 4th parity/check bit we get the following information.

$P4=5 \oplus 6 \oplus 7 \oplus 8$

$=B5 \oplus B6 \oplus B7 \oplus B8$

$=1 \oplus 1 \oplus 0 \oplus 1$

It will give the parity bit "1", by using the two logics.

Now we are ready to send our data, but first we need to insert the parity bits in the data at specified positions, as (bits 1, 2, 4, 8), fig.6. The data becomes

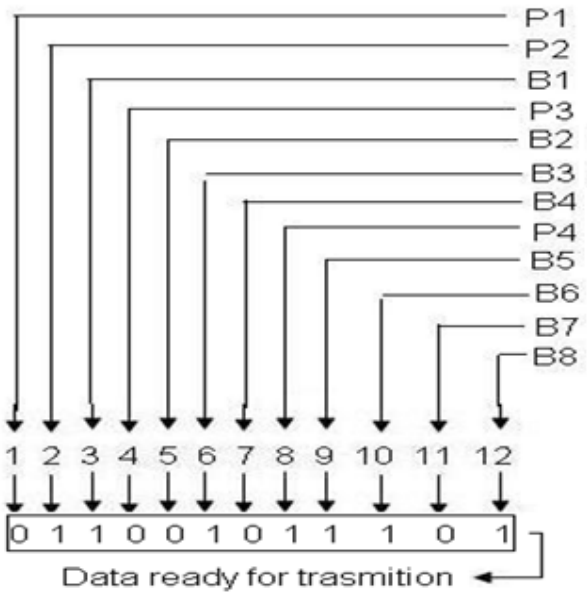



Fig. 6 Parity bits are inserted in the Data

When the message reaches the destination, we know the parity bits positions, so the data is collected from positions 3, 5, 6 ,7 ,9 ,10 ,11 ,12. Suppose the received message is 0 1 1 0 0 0 1 1 1 0 1, as the parity bits are at positions 1,2,4, and at 8 from left to right, so collect the data bits for decoding process, which is 10001101. The same process will be repeated, the matrix will be design, and than will find a parity bits at receiving side. As we have already did these things at transmitter side, we will find the parity bits by putting the received data bits. Which given below, in fig.7.

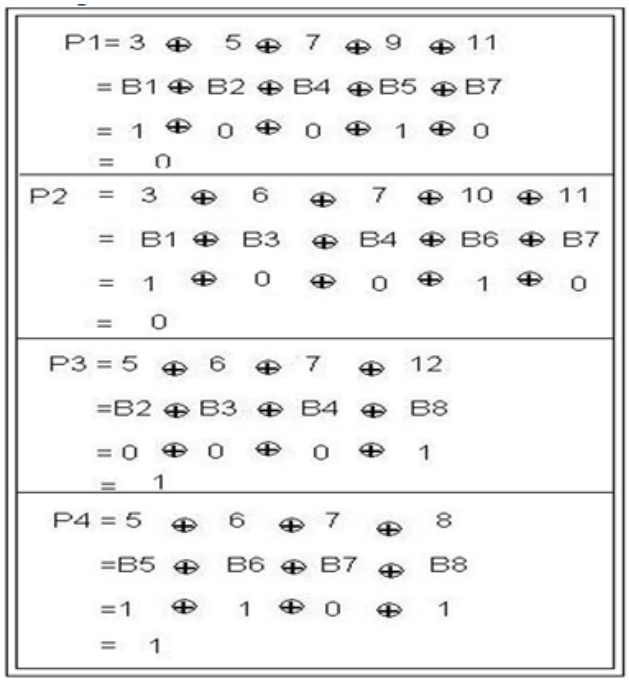



Fig. 7 Detection of Parity bits at receiving side

To check error in the data through the channel, we will compare the parity bits at receiver and transmitter ends that we have at the source and destination, if it gives the answer zero, that received data is error free, otherwise it will locate the poison.

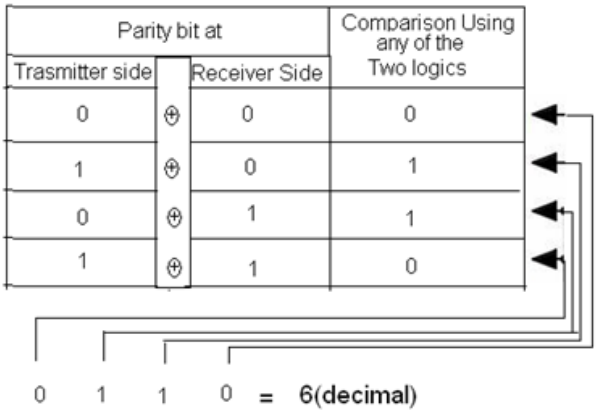



Fig. 8 Detecting the location of error

Fig.8 shows that the error was occurred at position 6, and was corrected. As shown in figure. 9.

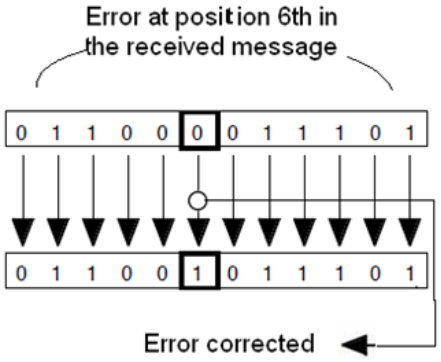



Fig. 9 Received message is corrected

## 5. Indication of Efficient Logic

Propagation Delay is the length of time taken for the quantity of interest to reach its destination. In computer networks, and Electronics, propagation delay is the amount of time it takes for the head of the signal to travel from the sender to the receiver over a medium. It can be computed as the ratio between the link length and the propagation speed over the specific medium [11]. Propagation delay = d / s where d is the distance and s is the wave propagation speed. In wireless communication, s=c, i.e. the speed of light. In copper wire, the speed s generally ranges from .59c to .77c [12, 13]. This delay is the major obstacle in the development of high-speed computers and is called the interconnect bottleneck in IC systems.

In electronics, digital circuits and digital electronics, the propagation delay, or gate delay, is the length of time starting from when the input to a logic gate becomes stable and valid, to the time that the output of that logic gate is stable and valid. Often this refers to the time required for the output to reach from 10% to 90% of its final output level when the input changes. Reducing gate delays in digital circuits allows them to process data at a faster rate and improve overall performance.

The difference in propagation delays of logic elements is the major contributor to glitches in asynchronous circuits as a result of race conditions.

The principle of logical effort utilizes propagation delays to compare designs implementing the same logical statement.

Propagation delay increases with operating temperature, marginal supply voltage as well as an increased output load capacitance. The latter is the largest contributor to the increase of propagation delay. If the output of a logic gate is connected to a long trace or used to drive many other gates (high fanout) the propagation delay increases substantially.

Wires have an approximate propagation delay of 1 ns for every 6 in of length [14]. Logic gates can have propagation delays ranging from more than 10 ns down to the picoseconds range, depending on the technology being used[14].

As we have used two logics for the detection of Parity Bits, & because of the number of increasing gates in any circuit the propagation delay will be increases, so in these two techniques the use of Exclusive Or gate is batter than AND-OR logic. I will prove it through simulations in PSpice[15], which is my future's goal.

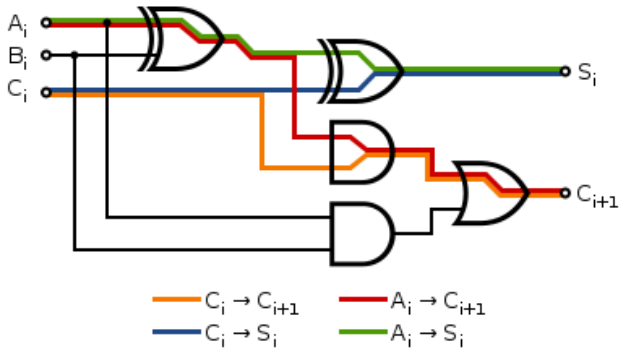



Fig.10A full adder has an overall gate delay of 3 logic gates from the inputs *A* and *B*to the carry output $C_{out}$ shown in red

For a reference consider this full adder circuit[], it is producing delay as the figure shows, when the number of gates increases & the length of circuit increases the propagation delay increases, & slow down the functionality of a circuit and vice versa.

## 6. Conclusion

The coding concept was introduced for the detection of error in data. At opening level single parity bits were used for the detection of error in the information, than multiple bits were used. But the problem was the retransmission of information in case of any error occurrence in the data. Hamming introduced a coding technique for the error detection as well as correction of that error at the receiver side. In this technique hamming added some extra bits called parity/check at some specific position, but for the detection of those parity bits Ex-OR and AND-OR logic was primarily used. I have indicate the efficient logic among the two, & in future will try to prove it through simulation that which logic producing the parity/check bits efficiently.

## References

[1] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, pp. 379-423, 623-656, July, August 1948.
[2] R. Hamming, "Error Detecting and Error Correcting Codes," *Bell System Technical Journal*, pp. 147-160, April 1950.
[3] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Application.* Prentice Hall, April 2004.
[4] Shannon C. E., 1948, A mathematical theory of communication. *Bell Syst. T.J.* 27, 379-423.
[5] Hill R., 1986, A first course in coding theory, Clarendon Press, Oxford.
[6] Huffman W. C. and Pless V., 2003, Fundamentals of Error Correcting Codes, Cambridge University Press.
[7] Behrouz Forouzan, *Data Communication and Networking*,Second Edition.
[8] Theodore F. Bogart. Jr, *Introduction to Digital Circuits*, International Edition 1992.
[9] Richard B.Wells, *Applied Coding and Information Theory for Engineers*, Published by Dorling Kindersley(India) Pvt. Ltd.
[10] Floyd & Jain, *Digital Fundamentals*, Seventh/Eighth Edition.
[11] http://en.wikipedia.org/wiki/Propagation_delay
[12] "What is propagation delay? (Ethernet Physical Layer)". *Ethernet FAQ*. 2010-10-21. Retrieved 2010-11-09.
[13] "Propagation Delay and Its Relationship to Maximum Cable Length". *Networking Glossary*. Retrieved 2010-11-09.
[14] Balch, Mark (2003). *Mcgraw Hill - Complete Digital Design A Comprehensive Guide To Digital Electronics And Computer System Architecture*. McGraw-Hill Professional. pp. 430. ISBN 9780071409278.
[15] http://edmondsonengineering.com/newbie.aspx
[16] Floyd, *Digital fundamentals, (seventh/eighth edition).*