30 BIT Hamming Code for Error Detection and Correction with Even Parity and Odd Parity Check Method by using VHDL

Brajesh Kumar Gupta
M.Tech Scholar
Electronics and Communication Department
Jaipur National University, Jaipur (India)

Prof. Rajeshwar Lal Dua
HOD, Electronics and Communication Department
Jaipur National University, Jaipur (India)

ABSTRACT

Hamming code error detection and correction methodology is used for error free communication in communication system. In communication system information data transferred from source to destination by channel. In between source and destination data may be corrupted due to any type of noise. To find original information we use Hamming code error detection and correction technique.
In hamming code error detection and correction technique to get error free data at destination, we encrypt information data according to even and odd parity method before transmission of information at source end.

In hamming code with even and odd parity check method by using VHDL, we transmit 25 bit information data with 5 redundancy bits from source and receive this data at destination. To find the value of these redundancy bits we have two methods, one of them is even parity method and another is odd parity method. In this paper we have written VHDL code for both methods at source as well as destination side.

At the point of destination, we receive 30 bit data, which was transmitted by source end. This receives data may be corrupted due to noise. To remove this noise we find the address of error bit then correct them. For finding the location of error bit and correct them we have again two methods one of them is even parity check method and another is odd parity check method .To find the location of error bit and correct them we write code in VHDL for destination .

In this paper we have written VHDL code for finding error location and correct error bit. We have also written code for decrypt this 30 bit encrypted data into 25 bit information data. Because of this code there is no need to use another circuit for decryption of encrypted data.
Up to today, at destination we were using one circuit for correcting error bit and another circuit for finding the information data from encrypted data. Now we can use only one circuit for correction error bit and finding the actual information data.

In this paper , we have described how we can generate 5 redundancy bit for 25 bit information data to make 30 bit data string for transmission by even and odd parity check method at source end. How we can find accurate 25 bit information data at destination from even and odd parity check method.
In this paper, we describe what is a Hamming code and how its work in communication system at source and destination. How we can generate 30 bit code for transmission and how we can get 25 bit actual information data from 30 bit received corrupted (error free) data string at destination.
Here, we have used Xilinx ISE 10.1 Simulator for simulating VHDL Code. Xilinx ISE 10.1 Simulator is a simulator which is used for simulating HDL language and schematic circuit diagram. Here we have used Xilinx simulator to simulate VHDL code for transmitter and receiver.

Keywords – Hamming code, Odd parity check method, even parity check method, Redundancy bits, VHDL language, Xilinx ISE 10.1 Simulator.

1. INTRODUCTION

In communication system, a secure data transmission from transmitter to receiver is very major issue .for error free transmission there are number of technologies. One of them is hamming code method. Hamming code works on the parity check method. Parities are two types first even parity and second odd parity .here we use both even parity and odd parity method to encrypt data before transmission.
In this paper we generate 5 redundancy bits for 25 bit information data to send 30 bit data string for transmission at source by using even and odd parity check method[1][2][3].

Suppose, we want to transmit 25 information data bit is "10111111011100110010101110" = 25`h17EE656. For this 25 bit information data we need 5 redundancy bits and these are "00111" (5h`07) and "11000"(5h`18) by using even add odd parity method respectably . After generating redundancy bits, add these bits to 25 bit information data for making 30 bit data string for transmission at source end. How we can generate 5 redundancy bits for 25 bit information data for making 30 bit data string for transmission at source end by using even and odd parity method will be discussed in details at communication with even parity method and communication with odd parity method section[1][2][3].

At destination receiver receives 30 bit data string from channel and check it, is it corrupted or not? If this data string is corrupted then receiver find the error location according to parity check method (even parity check or odd parity check method which one we use for finding error). And correct this error bit.
In this paper receiver pass only 25 bit information after correcting error and decrypted 30 bit data.
How we can find error bit location, how receiver correct this error bit and how we get 25 bit actual information data from

30 bit encrypted data string will be discussed in details in communication with even parity method and communication with odd parity method section[1][2][3].

In this paper we have used VHDL language for writing VHDL code for source and destination. At source, VHDL code is used for generating 5 redundancy bits for 25 bit information data and also written code for making 30 bit data string for transmission. At destination, we have written VHDL code for finding error bit location and for correcting that error bit. At destination we have also written VHDL code for finding 25 bit actual information data from 30 bit received encrypted data[6][7][8][9][10].
Here we have used Xilinx ISE 10.1 simulator to simulate VHDL code and given simulated results in term of input output waveforms [4] [5].

2. HAMMING CODE

Hamming code is a linear error-correcting code named after its inventor, Richard Hamming. Hamming codes can detect up to two simultaneous bit errors , and correct single-bit errors; thus, reliable communication is possible when the Hamming distance between the transmitted and received bit patterns is less than or equal to one . By contrast, the simple parity code cannot correct errors, and can only detect an odd number of errors.
In 1950 Hamming introduced the (7, 4) code. It encodes 4 data bits into 7 bits by adding three parity bits. Hamming (7, 4) can detect and correct single – bit errors. With the addition of overall parity bit, it can also detect (but not correct) double bit errors. Hamming code is an improvement on parity check method. It can correct 1 error bit only[1][2][3].

Hamming code method works only two methods (even parity, odd parity) for generating redundancy bit. In hamming code method for generating the number of redundancy bit use formula .The number of redundancy depends on the number of information data bits[1][2][3].

Formula for generating redundancy bit ----

$2^{r} >= D + r + 1$ ---------- (1)

Here r = number of redundancy bit
D = number of information data bit Calculate the number of number of redundancy bit for 25 bit of input data string by above formula We get 5 redundancy bit required.

2.1 Redundancy

To detect or correct the error we have to use some extra bits. These extra bits are called redundancy bits. We add these redundancy bits to the information data at the source end and remove at destination end. Presence of redundancy bit allows the receiver to detect or correct corrupted bits. The concept of including extra information in the transmission for error detection is a good one. But in place of repeating the entire data stream, a shorter group of bits may be added to the end of each unit. This technique is called redundancy because the extra bits are redundant to the information [7]

3. COMMUNICATION WITH EVEN PARITY CHECK METHOD

In communication system need two main part one them is source for sending data and another is destination for receives transmitted data. even parity check method count the number of one`s if number of one`s are even add zero (0) else add one (1). [1][2][3]

3.1 Source Section With Even Parity Method

In this paper, here I want to transmit 25 bit information data string. To transmit 25 bit information data need minimum 5 redundancy bit according to equation (1). Suppose , these redundancy bits are r(1),r(2),r(3),r(4),r(5). To find the value Redundancy bit, H ere we use even parity check method. The value of redundancy bit can be finding by XORING of different location of information data bit for different redundancy bit. the property of XOR gate is that if number of one`s are even in input its shows the output zero else its shows output one. By using this property we can easily find the number of one`s in a given string are even or odd for a particular redundancy bit. [6][7][8][9][10]

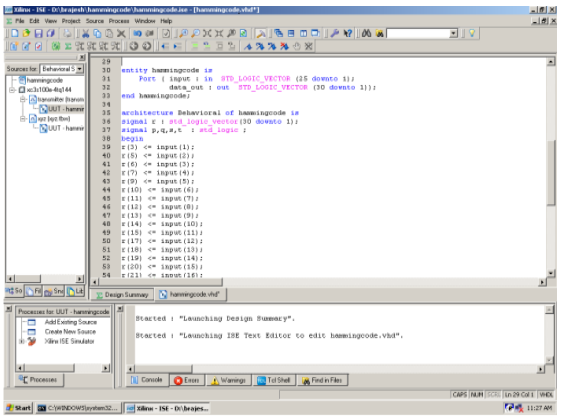
Suppose, Here we apply information data bit is "10111111011100110010101110" = 25`h17EE656. Before transmission of information data bits we need to add 5 redundancy bits.
Calculation for redundancy bit r(1) , By XORING input bit address given below
r (1) = 1,2,4,5,7,9,11,12,14,16,18,20,22,24.

Here the number of one`s are 9, this is a even number so according to even parity method
The value of r(1) = '1'
Calculation for redundancy bit r(2) , r(4) ,r(8) ,r (16)
r(2) = 1,3,4,6,7,10,11,13,14,17,18,21,22,25
r(4) = 2,3,4,8,9,10,11,15,16,17,18,23,24,25
r(8) = 5,6,7,8,9,10,11,19,20,21,22,23,24,25
r(16) = 12,13,14,15,16,17,18,19,20,21,22,23,24,25
For calculating the redundancy bit of r(2),r(4),r(8),r(16) count the number of one`s for appropriate redundancy bit according to given formula above [1][2][3].
The value of r (2) is 1(the number of one`s are 9) value of r (4) is 1(the number of one`s are 9) the value of r (8) is 0 (the number of one`s are 10) the value of r (16) is 0 (the number of one`s are 10). [1][2][3][6][7]

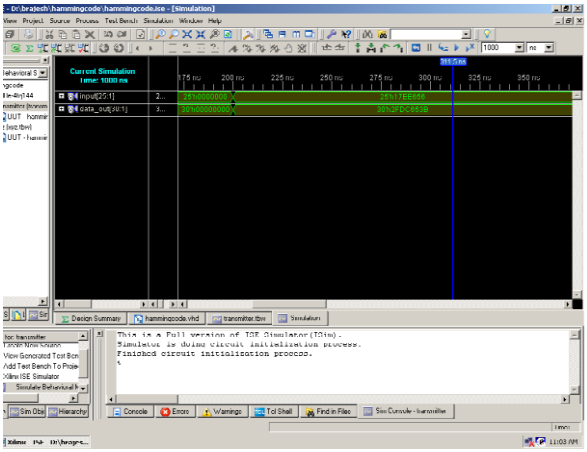
The calculation of redundancy bit is done by VHDL code written in Xilinx ISE 10.1 project navigator window. And simulate this VHDL code by using Xilinx ISE 10.1 Simulator and get the value of redundancy bit. Now we know the value of redundancy bits are "00111" 5h`17. [5][6][8]
After calculation of redundancy bit, add these bits in information data and get encrypted 30 bit data string for transmission.

The encrypted30 bit data string is.... 101111110111000110010100111011 = 30`h2FDC653B. Input and output simulated results shown bellow in Xilinx ISE 10.1 simulation window. In Xilinx ISE 10.1 simulation window 25 bit information input data string and 30 bit encrypted data represented by input (25: 1) and data_out(30:1) respectively.[5][6][10]

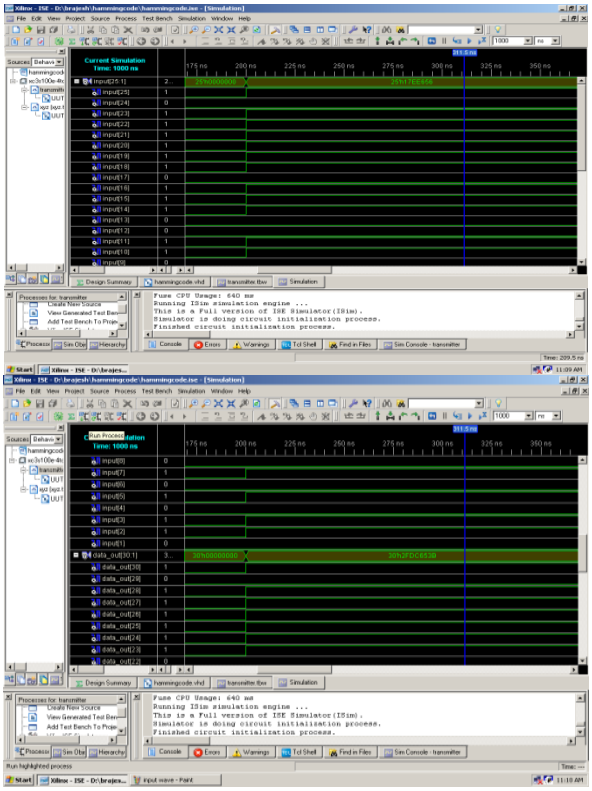
VHDL code for source end shown in given below Xilinx ISE 10.1 project navigator window. [5][6]

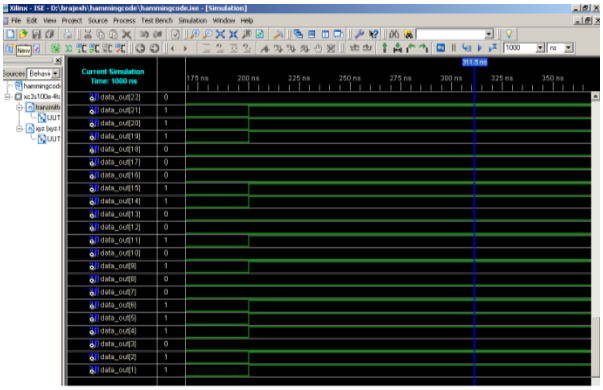


VHDL code for even parity check method at source



Xilinx ISE 10.1 window shows Input output wave form in Hexadecimal format at source end





Xilinx ISE 10.1 simulation windows shows input output wave form for source end in binary format

3.2 Destination Section With Even Parity Method

After adding redundancy bit in 25 bit information data, 30bit encrypted information data is transmit by transmitter at source end. At destination Receiver receives 30 bit encrypted data and check any error is occurred or not. If any error is occurred, receiver fined the error location and corrects this error bit. Number of address of error bit are same the number of redundancy bit added by transmitter before transmitting data.

In this paper we add 5 redundancy bit so that number of bits in the address of error is also 5 bit .Its generate the address of each location of received data.[2][3][8][9]

Suppose the name of the bit present in address of error is err_add then name of all bit are....

Err_add(1) = 1,3,5,7,11,13,15,17,19,21,23,27,29

Err_add(2)=2,3,6,7,10,11,14,15,18,19,22,23,26,27,30

Err_add(3)=4,5,6,7,12,13,14,15,20,21,22,23,28,29,30

Err_add(4)=8,9,10,11,12,13,14,15,24,25,26,27,28,29,30

Err_add(5)=16,17,18,19,20,21,22,23,24,25,26,27,28,29,30

Suppose a transmitter of source end transmit data string after adding redundancy bit are 10111111011100011001010011101l = 30`h2FDC653B but due to some noise at destination receiver receives error data. Now receiver find the location of error bit, after finding the error location correct that error bit and find actual encrypted data which is transmitted by transmitter at source end.[2][10]

Suppose, transmitter of source end transmit data is 30`h2FDC653B(10111**1**1101110001100101001110l1) and at destination receiver received error data is 30`h2EDC653B(10111**0**1101110001100101001110l1) now receiver find the address of this error bit. [1][2][3][7][9]

Method for calculation of finding error bit location count the number of one`s in encrypted information data string is received by receiver according to Err_add bits. If number of one`s are even add zero in Err_add bit location else add one. [7][8][9]

Calculation for first bit of err_add(1) , count the number of one`s in these location in received data received by receiver 1,3,5,7,11,13,15,17,19,21,23,27,29.the number of one`s are 9 in these location .so that the value of Err_add is '1'[1] [7][10]

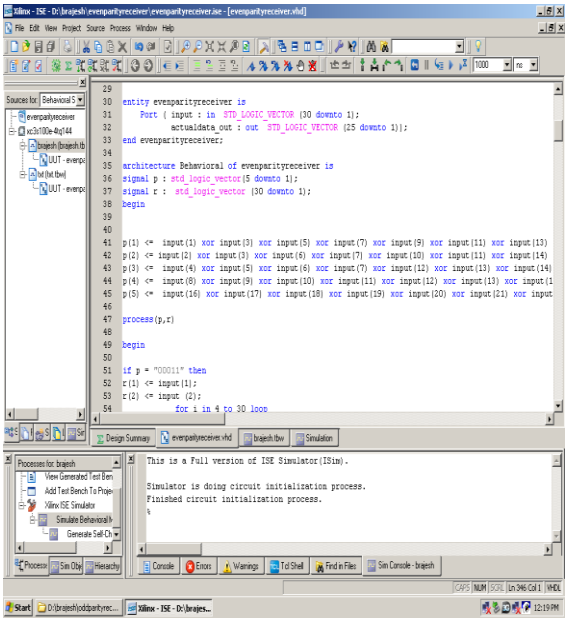
Calculation for Err_add(2) same method apply for calculation of these bit . count the number of one`s in 2,3,6,7,10,11,14,15,18,19,22,23,26,27,30 in these address.

The number of one`s are 10 so that the value of Err_add is ‘0’.[1][2][3]

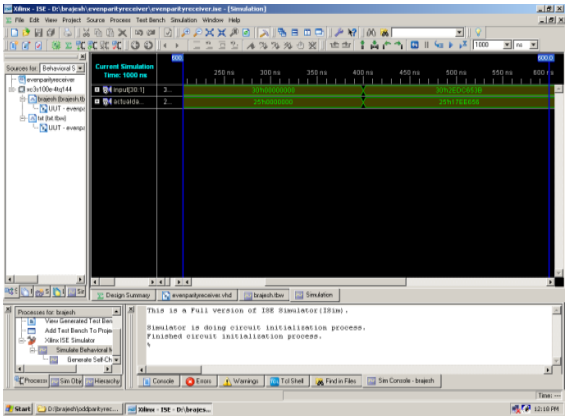
Calculation for Err_add(3) , count the number of one`s in 4,5,6,7,12,13,14,15,20,21,22,23,28,29,30 in these location given above . the number of one`s are 10 so the value of Err_add is ‘0’. [2][3]
Calculation for Err_add(4) , the value of Err_add is ‘1’ because of the number of one`s in 8,9,10,11,12,13,14,15,24,25,26,27,28,29,30 these address are 9.[1][2][3]

Calculation for Err_add(5) , for calculate the value of Err_add(5) count the number of one`s in for address of 16,17,18,19,20,21,22,23,24,25,26,27,28,29,30 location of received data by the receiver . the number of ones are ‘9’ so that the value of Err_add(5) is ‘1’.[1][2][3]
Finally we get the address of error location is Err_add = 11001 (5`h19 ) .after getting the location of error bit receiver correct that error bit by replacing zero by one and one by zero. And we get actual encrypted 30 bit data is 30`h2FDC653B (10111**1**1101110001100101001110 11) transmitted by transmitter at source end. Now we find actual “101111101110011001010110” = 25`h17EE656 25 bit information data from 30 bit encrypted data string.[8][9][10]
We write VHDL code for finding the error bit location, correcting error bit and decrypt this encrypted data. VHDL code for destination end shown in given below Xilinx ISE 10.1 project Navigator window. [5][6][7]

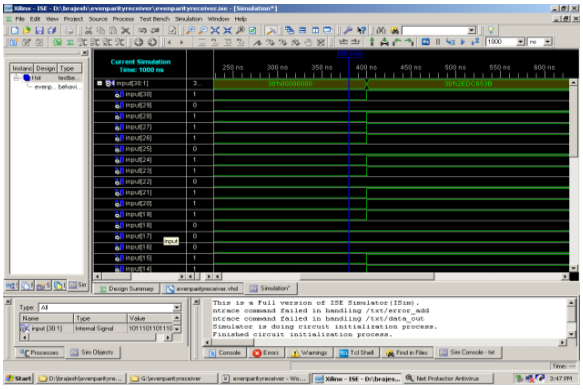
Simulated results for destination end shown below. Xilinx ISE 10.1 Simulation window shows 30 bit receives encrypted data string and 25 bit actual error free information data string, whose want to be transmit.[5][6][7]

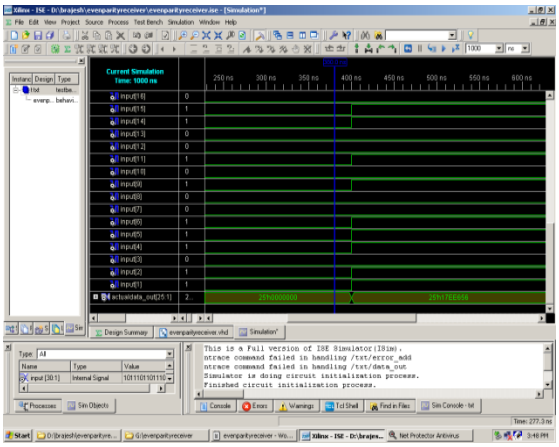


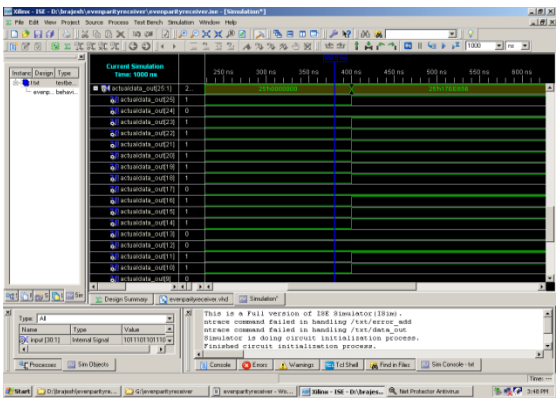
VHDL code for even parity check method at destination

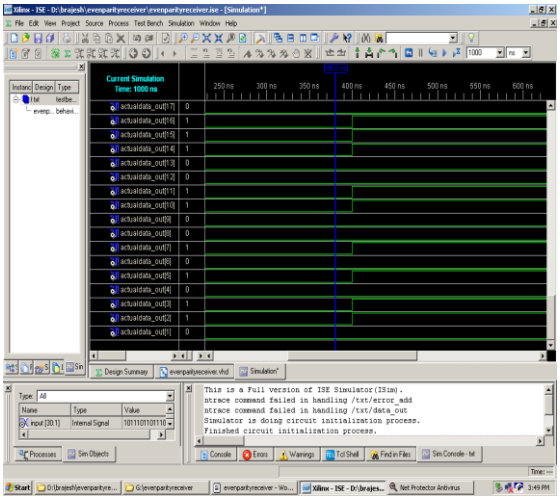


Xilinx ISE 10.1 simulation window shows the input and output waveform at destination in Hexadecimal format









Xilinx ISE10.1 simulation windows shows Input output wave form for destination end in binary format

4. COMMUNICATIONS WITH ODD PARITY CHECK METHOD

In communication with odd parity information data encrypted by odd parity method at source end then transmit it by transmitter. At destination end receiver receives this transmitted data and check any error is occurred or not with odd parity method.[1][2][3]
In odd parity method count the number of one`s in a given string, if number of one`s are even add zero else add one for encrypt information data.[1][2][3]

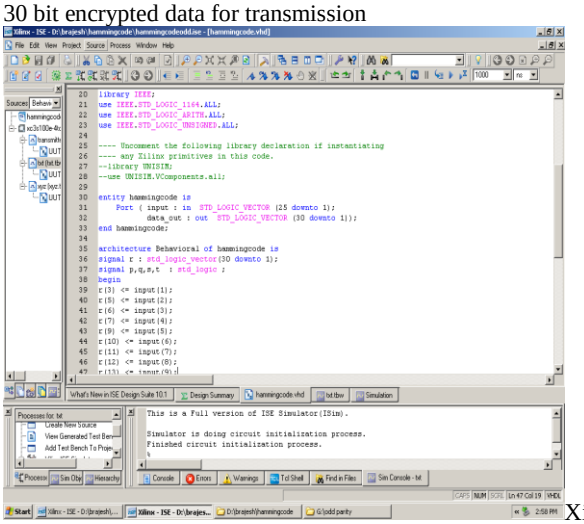
Here we want to send 25 bit information data at source end , to transmit 25 bit information data according to hamming code we need minimum 5 redundancy bit described in hamming code section above. How we can find the value of redundancy bit, how can find the error bit location described in source section and destination section respectably.[1][2][3]

4.1 Source Section With Odd Parity Method

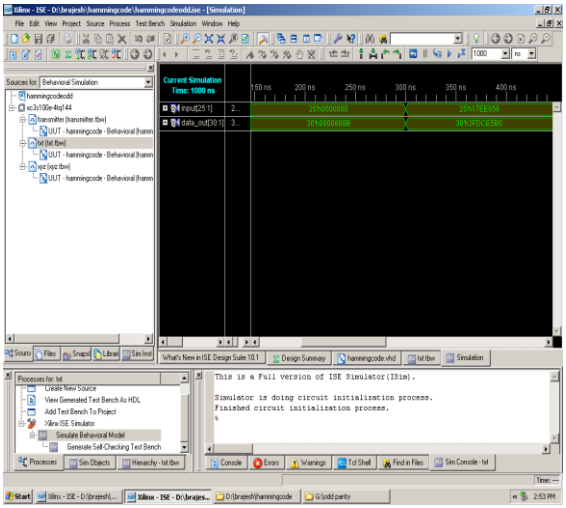
To transmit 25 bit information data at source end need to add minimum 5 redundancy bits. Suppose , These redundancy bits are r(1),r(2),r(4),r(8),r(16).
r (1) = 1,2,4,5,7,9,11,12,14,16,18,20,22,24.
r(2) = 1,3,4,6,7,10,11,13,14,17,18,21,22,25
r(4) = 2,3,4,8,9,10,11,15,16,17,18,23,24,25
r(8) = 5,6,7,8,9,10,11,19,20,21,22,23,24,25
r(16) = 12,13,14,15,16,17,18,19,20,21,22,23,24,25

To find the number of one`s are even or odd for redundancy bits r(1),r(2),r(4),r(8),r(16) in VHDL code use XOR and NOT gate . After finding the value of these redundancy bits , add these redundancy bits in 25 bit information input data to make 30 bit encrypted data string for transmission by transmitter at source end.[7][8]
Suppose, we want to transmit 25 bit information data is 25`h17EE656 (10111111011100110010101110), Now for generate the value of redundancy bit write VHDL code. VHDL code for source end shown in given below Xilinx ISE 10.1 project Navigator window. After writing VHDL code simulate it by Xilinx ISE 10.1 simulator and get the value of r(1),r(2),r(4),r(8),r(16). We get value of r(1) is 0 (zero) ,r(2) is 0(zero),r(4) is 0(zero),r(8) is 1(one) and r(16) is 1 (one) . Now add these redundancy bits to 25 bit information data and find encrypted 30 bit data is 30`h2FDCE5B0 (101111101111001010110110000) for transmission. The VHDL code simulated results are given below in Xilinx ISE 10.1 simulation window. [5][6][7][8][9][10]

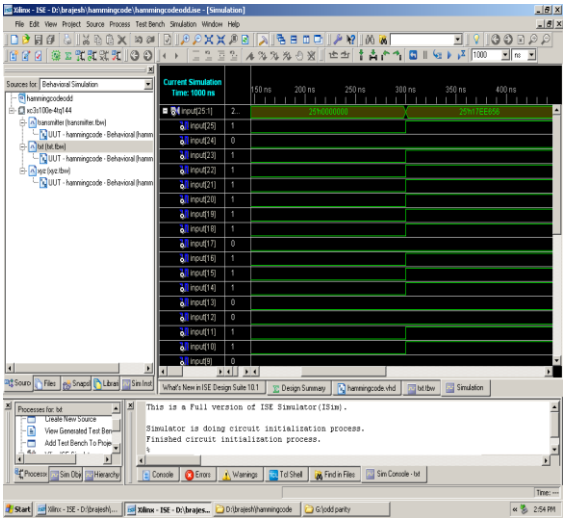
Here input means 25 bit information data and data_out means 30 bit encrypted data for transmission

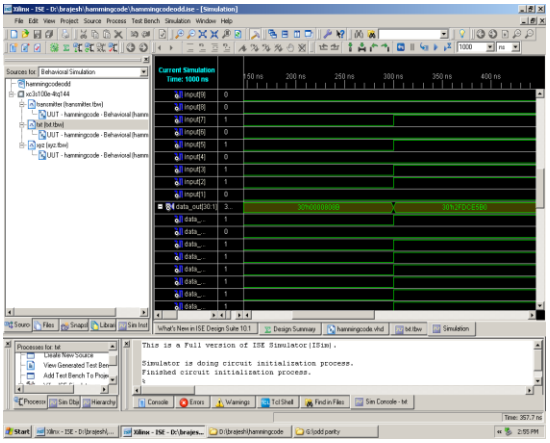


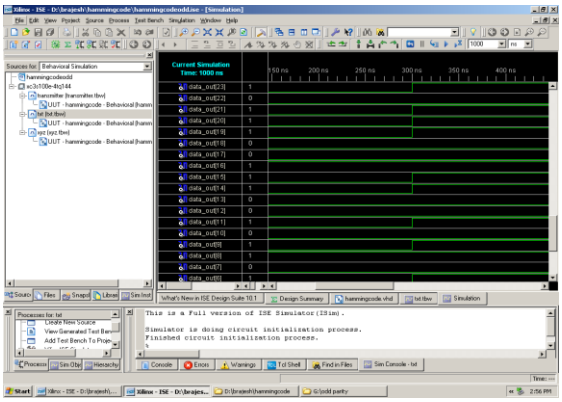
Xilinx ISE 10.1 project Navigator Window shows VHDL code for source end.

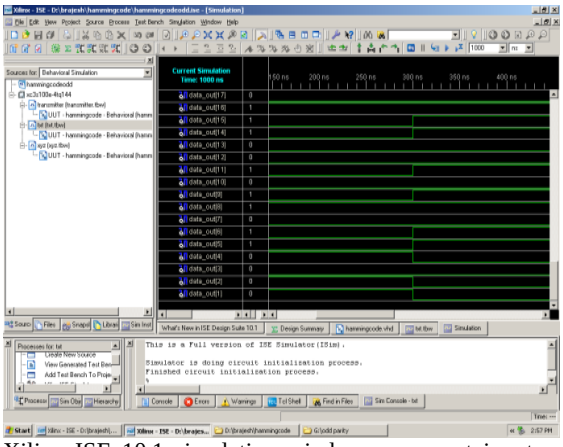


Xilinx ISE 10.1 simulation window shows Input output wave form at source end in Hexadecimal format









Xilinx ISE 10.1 simulation windows represent input output waveform for odd parity source end in binary format

4.2 Destination Section with Odd Parity Method

In destination section we receiver receives data which is transmitted by transmitter at source end. In between source end and destination end communication is possible by some medium called channel. Encrypted data travelled by this channel from source end to destination end, this channel may be noisy. Due to this noisy channel received data may be corrupted; to find the location of corrupted bit and for correcting that error bit use hamming code odd parity check method.[1][2][3]

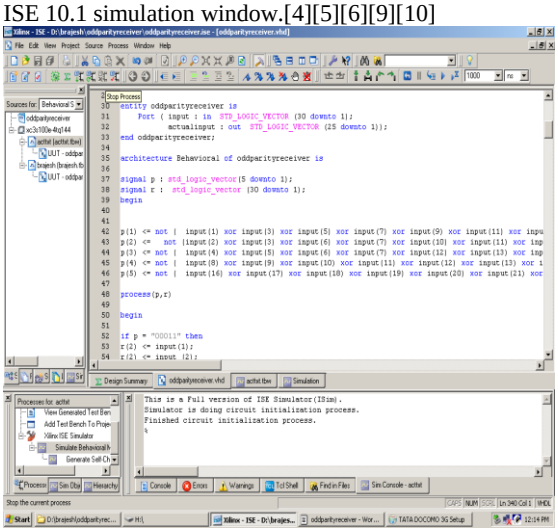
In this paper we transmit 30 bit encrypted data string by transmitter at source end and at destination end, receiver receives this data string. Transmitted data string travelled by channel from source end to destination end. This channel may be noisy; due to this noisy channel received data string may be corrupted. To find address of corrupted bit we need 5 bits[1][8][10]
Suppose the name of error bits are erroradd.
erroradd(1) = 1,3,5,7,11,13,15,17,19,21,23,27,29
erroradd(2)=2,3,6,7,10,11,14,15,18,19,22,23,26,27,30
erroradd(3)=4,5,6,7,12,13,14,15,20,21,22,23,28,29,30
erroradd(4)=8,9,10,11,12,13,14,15,24,25,26,27,28,29,30
erroradd(5)=16,17,18,19,20,21,22,23,24,25,26,27,28, 29, 30

To find the location of error bit, write code in VHDL. In hamming code error detection and correction with odd parity method by using VHDL we use XOR and NOT gate for finding the error bit location.
Suppose, at source end transmitter transmit data is 30`h2FDC653B(10111**1**1101110001100100111011) and due to noisy channel at destination end receiver receives corrupted 30 bit data string is 30`h2EDCE5B0(10111**0**1101110011100101101100000).[3][8][9][10]

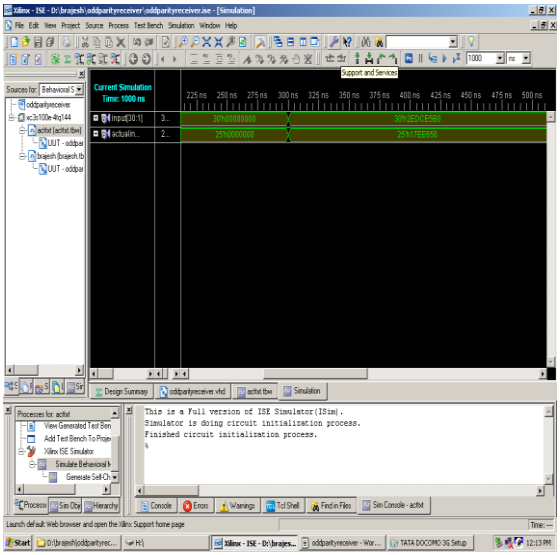
Now receiver fined location of error bit and corrects them. For finding the error bit location and correcting that error bit we have written code in VHDL. [8]

After writing code in VHDL, VHDL code is simulated by Xilinx ISE 10.1 simulator. And get the value of erroradd. The value of erroradd for received data is 5`h19 (11001) here erroradd(1) is 1(one),erroradd(2) is 0(zero) , erroradd(3) is 0 (zero) , erroradd(4) is 1(one) and erroradd(5) is 1 (one). After finding error bit location writes VHDL code for correcting error bit (replace this error bit zero by one and one by zero). [6][7][8]
After correcting this error bit, at destination end we write VHDL code for finding actual 25 bit information data from 30 bit encrypted data which is transmit by transmitter at source end.[6][7][8][9]

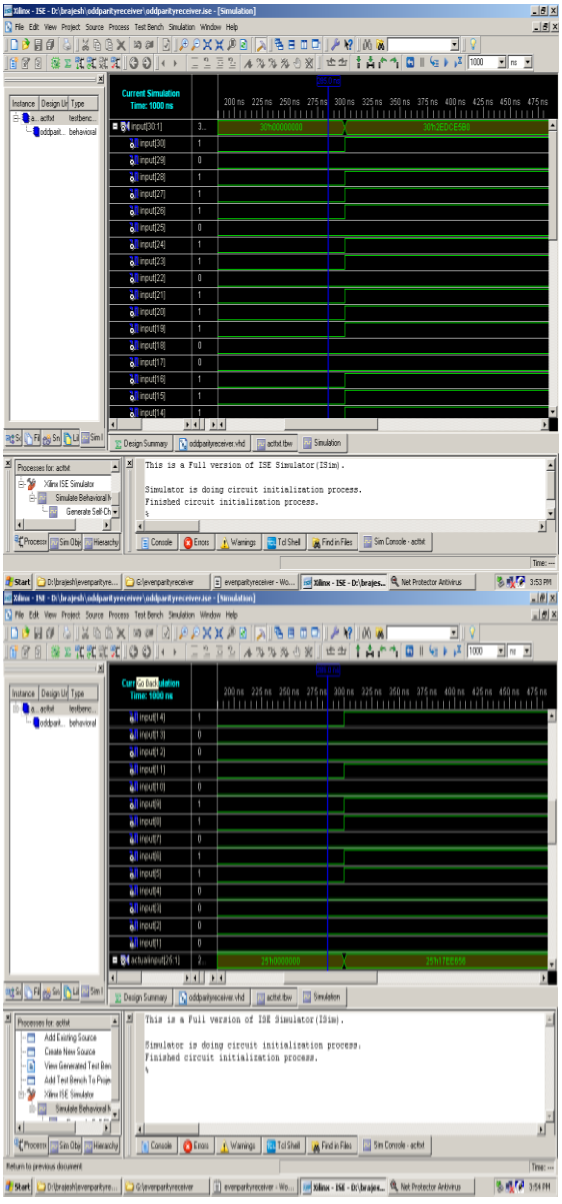
A VHDL code for finding the location of error bit, for correcting that error bit and getting the 25 bit actual information data“10111111011100110010101 10”=25`h17EE656 from 30 bit encrypted data shown below in Xilinx ISE 10.1 project navigator window.
Now Xilinx ISE 10.1 simulator used for simulates VHDL code. Simulated VHDL code results shown below in Xilinx ISE 10.1 simulation window.[4][5][6][9][10]

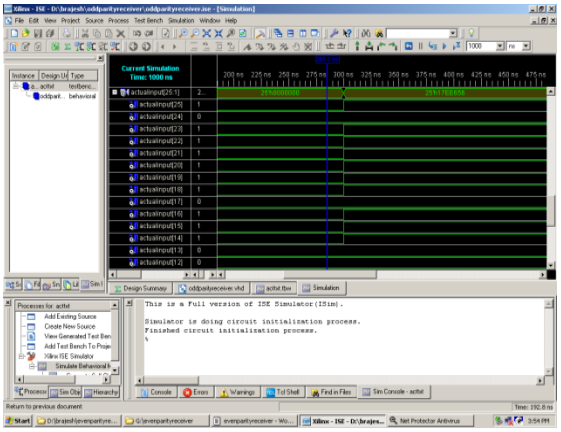


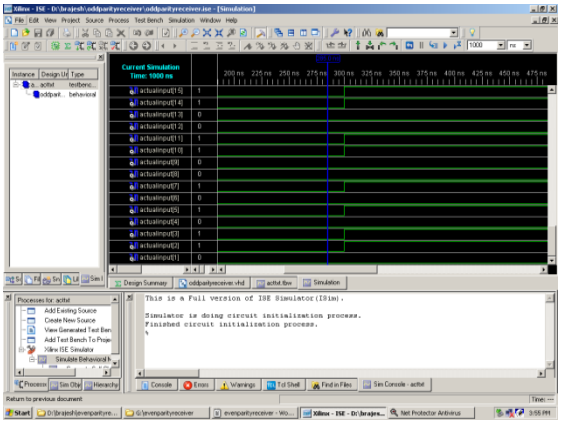
Xilinx ISE 10.1 project Navigator window shows VHDL code for odd parity check method at destination end



Xilinx ISE 10.1 simulation window shows input output wave form at destination







Xilinx ISE10.1 simulation windows represent input output waveform for destination end in binary format

5. APPLICATION

An application of hamming code error detection and correction with even parity check and odd parity check method is that by this method there is no need to transmit data string again by transmitter at source end, if only single bit error is occurred by noisy channel because at destination end receiver can regenerate the original data string which was transmitted by transmitter at source end. Error detection and correction codes are used in many common systems including: storage devices (CD, DVD, DRAM), mobile communication (cellular telephones, wireless, microwave links), digital television, and high-speed modems (ADSL, xDSL)..

6. ADVANTAGES

Up to today, at destination end we were using two circuits for generating 25 bit actual information data which we want to transmit from source end. One for finding location of error bit and correcting that error bit, another for decrypt, received error free encrypted data.

Now we can use only one circuit for performing all operation at destination so that delay time could be reduced for regenerating 25 bit actual information data from received 30 bit encrypted corrupt data string. And the complexity of circuit configuration can reduce.

Speed of communication system also depends on the number of frame (combination of number of bit is called frame)that can be transmitted in a second. To increase the speed of communication system increases the number of frame per second or increase the number of bits in a frame.

Here we have increased the frame size to increase the number of bits in a single frame. Up to today we can transmit only 11 bit (7 bit data and 4 redundancy bit) in a frame but now we can transmit 30 bits (25 bit information data with 5 redundancy bit) in a single frame.

7. CONCLUSION

The overall conclusion of this paper is that, speed of communication system can be increased by using these methodologies; we can transmit more combination of data (more information in a single frame).

The complexity of circuit also reduced and we can use one IC in place of two IC`s for regenerating actual information data from encrypted corrupt received data at destination end.

Up to today we can transmit 7 bit information data string means only 2^7 (128) combination of data , means only 128 type of information can be transmit at a time .

Now by using 30 bit hamming code error detection and correction with odd & even parity check method we can transmit 25 bit information data string means 2^25 (33554432) combination of input data. Now we can transmit 33554432 type of information at a time.

8. REFERENCES

[1] Data communication and networking , Behrouz A. Forouzan , 2nd edition Tata McGrawHill publication.

[2] Communication Networks, available at: http://www.pragsoft.com/books/CommNetwork.pdf

[3] Xilinx ISE 8 Software Manuals and Help: http://www.eng.uwaterloo.ca/~tnaqvi/downloads/DOC/sd192/ISE8_1i_manuals.pdf

[4] Xilinx Training Course Listing, available at http://www.xilinx.com/training/xilinx-training-courses.pdf

[5] ISE 10.1 Quick Start Tutorial, available at http://www.xilinx.com/itp/xilinx10/books/docs/qst/qst.pdf

[6] VHDL (VHSIC hardware description language): http://en.wikipedia.org/wiki/VHDL

[7] VHDL Tutorial, available at http://www.vhdl-online.de/tutorial/

[8] VHDL Designer's Guide, available at http://www.doulos.com/knowhow/vhdl_designers_guide/

[9] A VHDL Primer , J. Bhasker , 3 rd edition PHI publication.

[10] Digital Logic Design with VHDL , Stephen Brown & Zvonko Vranesic , 2 nd edition TMH publication

9. AUTHOPRS PROFILE

Professor Rajeshwar Lal Dua a Fellow Life Member of IETE and also a Life member of: I.V.S & I.P.A, former "Scientist F" of the Central Electronics Engineering Research Institute (CEERI), Pilani has been one of the most well known scientists in India in the field of Vacuum Electronic Devices for over three and half decades. His professional achievements span a wide area of vacuum microwave devices ranging from crossed-field and linear-beam devices to present-day gyrotrons.
He was awarded a degree of M.Sc (Physics) and M.Sc Tech (Electronics) from BITS Pilani. He started his professional carrier in1966 at Central Electronics Engineering Research Institute (CEERI), Pilani. During this period he designed and developed a specific high power Magnetron for defence and batch produced about 100 tubes for their use. Trained the EngineeIndustries with know how transfer for further production of the same.
In 1979 he visited department of Electrical and Electronics Engineering at the University of Sheffield (UK) in the capacity of independent research worker, and Engineering Department of Cambridge University Cambridge (UK) as a visiting scientist. After having an experience of about 38 years in area of research and development in Microwave field with several papers and a patent to his credit. In 2003 retired as scientist from CEERI, PILANI & shifted to aipur and joined the profession of teaching. From last eight years he is working as professor and head of electronics department in various engineering colleges. At present he is working as head and Professor in the department of Electronics and communication engineering at JNU, Jaipur. He has guided several thesis of M.tech .of many Universities.

Mr. Brajesh Kumar Gupta , student of Mtech III sem of Jaipur National University, Jaipur ,
I have completed my B.E from "R.K.D.F institute of science & Technology ",Bhopal in 2007 in Electronics & Communication Engineering under the university of R.G.P.V Bhopal . I also done VLSI design Advanced post Graduation Diploma in DEC- 2007 from VEDANT , SCL (semiconductor of laboratory) Mohali , Chandighar,it comes under Department of Space government of India (ISRO).