Rakesh S / International Journal of Engineering Research and Applications
(IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 3, May-Jun 2012, pp.1288-1291

Design and Implementation of LRC and CRC algorithms in Netsim

Rakesh S*

*(Assistant Professor, Department of Computer Science & Engg. Siddaganga Institute of Technology, Tumkur, Visvesvaraya Technological University, Belgaum,)

Abstract

Error detection is a technique that enables reliable delivery of digital data over unreliable communication channels. Many communication channels are subject to channel noise, and thus errors may be introduced during transmission from the source to a receiver. Basically error detection mechanism contains various methods but LRC and CRC are the most commonly used methods. A longitudinal redundancy check (LRC) is a form of redundancy check that is applied independently to each of a parallel group of bit streams. A Cyclic Redundancy Check (CRC) is one of the most powerful redundancies checking technique. The problem is user knows these concepts of the LRC and CRC techniques while reading the text book but they don't know practically how LRC and CRC techniques are working. Initially on NetSim these LRC and CRC techniques are implemented as a static. That is user give the inputs of the LRC and CRC techniques to NetSim and displays the outputs on the same system where the user gives the input. So that user can't understand the concepts of the LRC and CRC techniques clearly. Animations of the LRC and CRC techniques are also presented on the NetSim but it fails to clear the concepts of the LRC and CRC techniques to the user. For the above problem, the frontend designs of LRC and CRC techniques can be created by using Java Swing on NetSim that is to change the static implementation of the LRC and CRC techniques to dynamic implementation by making NetSim to work on LAN. That is user give the inputs to NetSim by using this frontend and displays the output corresponding to this input in another system. So that user can easily understand the LRC and CRC techniques, because while the input given by the user is transmitted on the network it may or may not get changed. By analyzing the output at the other system user can easily understand the concepts of the LRC and CRC techniques. At the backend the LRC and CRC techniques are coded using C. In NetSim the C-Editor is uploaded so that user can write their own codes of LRC and CRC techniques and compile the codes by using C-Compiler.

Keywords – **CCITT, CRC, LRC, LAN, NetSim**

1. Introduction

NetSim is a first of its kind educational Network Simulation software and has proved to be an indispensable tool for network lab experiments, research and development and which takes user inputs and provides output metrics. NetSim has also been featured with Computer Networks and Internet. NetSim provides network performance metrics at various abstraction levels such as network, sub-network, node and a detailed packet trace. NetSim features a development environment with a source code editor and a compiler. Model libraries with source code are provided for user modification and options are not limited to the listing as it is possible to develop any type of protocol or device model with NetSim's protocol editing facilities. NetSim is a popular tool developed by TETCOS, in association with Indian Institute of Science, Bangalore.

The objective of this thesis is to make concepts easy and clear about difficult techniques of algorithms of computer networks with the help of animation and practical implementations of techniques and algorithms. In NetSim the techniques and algorithms of the computer networks were implemented as a static system. Because of the static implementation, user was unable to clearly understand the techniques and algorithms of the computer networks on NetSim. By designing and implementing static system to dynamic system on NetSim, the user can easily understand the techniques and algorithms of the computer networks.

This thesis will give the easy way of understanding the error detection techniques such as LRC and CRC through dynamic implementation on NetSim. In earlier days the NetSim was worked only on a static system, i.e. NetSim was implemented on single system and user must give the input data to one system where the NetSim software is installed and NetSim displays the output on the same system. Because of this the user can't understand the computer networks concepts clearly. Then the question that arises is that why we can't try to work NetSim on LAN? For this, we got the answer and we were successful to work NetSim on LAN. Once if NetSim is implemented on LAN, then user can easily identify whether the data is received with error or without error. In this project NetSim is implemented on dynamic system i.e. user can give the input data to one system, through LAN the data is transmitted to another system and the user can analyze the output displayed on another system.

The concepts of LRC and CRC techniques for error detection through text book was difficult to understand because it will give theoretical concepts about LRC and CRC techniques but not practical. In NetSim initially these LRC and CRC techniques for error detection was implemented as a static i.e. user give the inputs and see the outputs on the same system where the user can give the input and some

animations of the LRC and CRC techniques for error detection are displayed. Again the static implementation of the LRC and CRC techniques was difficult to understand. Even though through animation wise the static implementation was good but for working wise it was difficult to understand.

For the above problem, the solution is to change the static implementation of the LRC and CRC techniques to dynamic implementation. User gives the input from one system and sees the output on other system. For this, designing the frontends of the LRC and CRC techniques in NetSim. By using this frontend user gives the input and through LAN given inputs are transmitted and output will be displayed on this system design is the new version of the other system. Animation wise of the LRC and CRC techniques are also changed and uploaded in NetSim. In backend user can write their own source codes and execute their codes and give this executable file .exe as input and see the output on other system.

Initially the frontend designs of the LRC and CRC techniques are created. The frontend designs are created by using Java Swings on NetSim. In frontend there are two modes, sample mode and user mode. In sample mode, some in built library codes of the LRC and CRC techniques are uploaded on NetSim. Users select the sample mode, browse the .txt file for input and click the run button. NetSim display the output and also display the animation part of the CRC technique. In LRC technique user directly give the input, this input contains less than or equal to 8 characters and NetSim displays the corresponding output for the input. Another mode is the user mode, here user can write their own source code of the LRC and CRC techniques execute their codes and after executing .exe files are created. This .exe file acts like an input to the NetSim and click the run button, the output will display on the other system. In backend, user can write their own source codes of the LRC and CRC techniques in C-Editor and also C-Complier is provided for compilation of the codes.

2. Procedure

2.1 Steps for doing LRC:

- A block of bits is organized in a table (rows and columns).
- For example instead of sending 32 bits, we organize them in a table made of 4 rows and 8 columns.
- We then calculate the Parity bit for each column and create a new row of 8 bits which are the parity bits for the whole block.
- Note that the first parity bit in the 5th row is calculated based on all the first bits.
- The second parity bit is calculated based on all the second bits and so on..
- We then attach the 8 parity bits to the original data and send them to the receiver.

2.2 Steps for doing CRC:

Steps performed by Sender:

- Get the raw frame.
- Left shift the raw frame by n bits and divide it by divisor.
- The remainder is the CRC bit.
- Append the CRC bit to the frame and transmit.

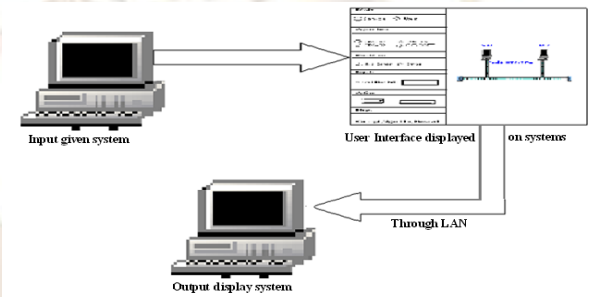
Steps performed by Receiver:

- Receive the frame.
- Divide it by divisor.
- Check the reminder.

3. System Design

3.1 System design of LRC technique on NetSim

This system design is the new version of the NetSim design of the LRC technique In this new version of the system design it has overcome the draw backs of the earlier version system design of the LRC technique on the NetSim. From the scratch it has been fully redesign the frontend design of the LRC technique in the NetSim by making the frontend design to user friendly. In frontend it has been provided Five modules which helps the user to give the inputs to NetSim.



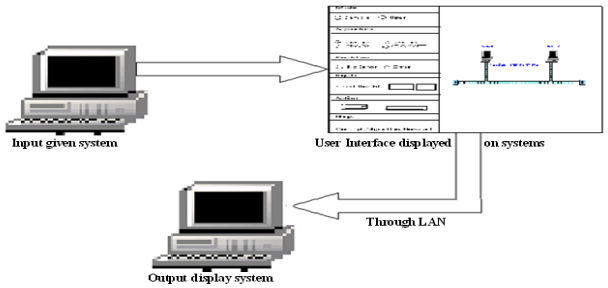
This design gives consistent look and feel newly designed frontend. Tre re-architected frontend or UI improves productivity and usability. Increased consistency in the way input parameter are modeled and how performance metrics are reported. We simplified the configuration of the NetSim. The scenario configuration has been re-designed to follow the 5-layer TCP/IP stack providing a simpler and layer wise configuration. This project provides the superior graphing to the NetSim.

In newly designed system contains 64-bit format of the LRC technique. User can give only 8-bit data, remaining 56 bits are already exists on the NetSim. These 56-bits are divided into 8 columns each and each row contains 7 bits and 8th bit is taken from the user input to each row.

3.2 System design of CRC technique on NetSim

In this system design contains all the 4 polynomials of the CRC technique such as CRC-12, CRC-16, CRC-32 and CRC-CCITT. In frontend design we have provided separate

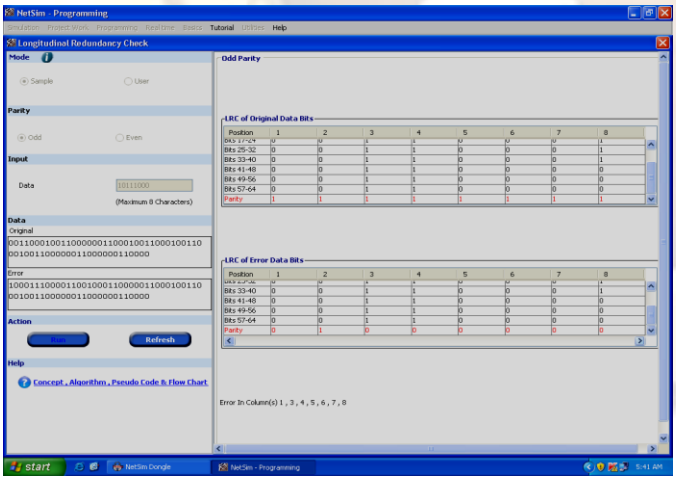
buttons for all these 4 polynomials so that user can choose any one polynomial and execute it.



When the user selects the CRC-12 polynomial, it computes the 12-bit value. User create the .txt file and in this file user store the input data up to 5000 bytes. If CRC-12 polynomial means user can store the data of 12-bits for easy understanding. This file acts like a input and we have provided the file browser on the frontend. By using this browser user can browse the .txt file and can give this file as a input to the NetSim. Similary for all the remaining polynomials user can follow the above procedure.

4. Implementation

In frontend design, GUI's are designed with the help of Java Swings on NetSim. In Java Swings there exist many built in functions to design a GUI. With the help of these built in functions frontend of the LRC and CRC techniques are created. The frontend of the LRC technique consists of six modules. They are: Mode, Parity, Input, Data, Action, and Help. The Mode module contains Sample and User modes and Parity module contains Even parity and Odd parity modes.



In Sample mode, when the user wants to know how the LRC technique is working then the user can click on this mode. Here some samples of built in codes are uploaded. When the user clicks on this mode some animations of the LRC technique is displayed on the output window of the NetSim on another system for the easy understanding of these techniques.

In User mode, when the user click on this mode another new window will be opened. In this new window two options are provided, one for creating the file and writing source codes in C-Editor and another for selecting the .exe file. This .exe file is created after executing the source codes in C-Editor. This .exe file is given as input in input mode and this input data will be transmitted on network using LAN and displays the output in output window of the NetSim on another system. That is through LAN, PC to PC communication is established so that input is given in one PC and output will be displayed on the other PC.

When the user click on Sample mode, Even parity mode and user can give the input bits in Input mode then built in codes are executed by taking this user input bits and appending the even parity according to the user input bits and this is transmitted on the network through LAN and displays the output in NetSim on another system. Similarly when the user click on Sample mode, Odd parity mode and user can give the input bits in Input mode then built in codes are executed by taking this user input bits and appending the odd parity according to the user input bits and this is transmitted on the network through LAN and displays the output in NetSim on another system.

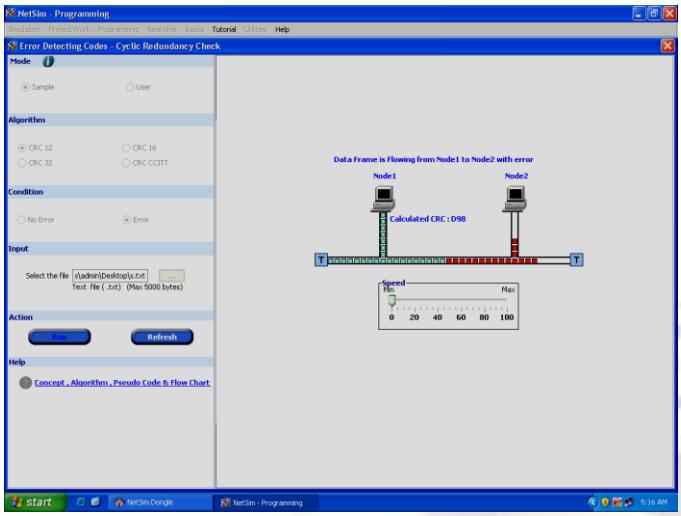
The input will have only 8-bits but the built in codes will work on 64-bits format. So that we need to upload the remaining 56-bits i.e. these 56-bits are constant and these 56-bits can be divided by 8 columns and each column contains 7-bits and the 8th bit is taken from the user input bits in the output display on the NetSim i.e. the user input bits are presented in 8th column in the output.

The Original data field can display up to 64-bits since the built in bits of the LRC technique contains 56-bits and user can give 8-bits as input. So all these 56 bits plus 8-bits together known as original data and these 64-bits are displayed in the Original data field. The Error field also displays up to 64-bits.

In Action module, two buttons are created. The first button is RUN button and second button is REFRESH button.

The help module contains the hyperlink to Concepts, Algorithms and Flow charts for the LRC technique. If the user wants any information about LRC technique, then they can click on this link and it will display all the details about the LRC technique.

The frontend of the CRC technique also contains six modules. They are: Mode, Algorithm, Condition, Input, Action and Help. The Mode module is same as that of the LRC technique. The Algorithm module contains CRC-12, CRC-16, CRC-32 and CRC-CCITT polynomials. The Condition module contains No Error and Error modes. The Action and Help modules are same as that of LRC technique.



When the user selects the Sample in Mode module and CRC-12 or CRC-16 or CRC-32 or CRC-CCITT in Algorithm module and gives the .txt file as input in Input module, then the built in code of the CRC-12 or CRC-16 or CRC-32 or CRC-CCITT will be running in the NetSim and stores the output in the output .txt file (this file is present in the application path of the NetSim) on another system i.e. the output is transmitted on the network through LAN and stores the output in the file. The animation about the CRC-12 or CRC-16 or CRC-32 or CRC-CCITT polynomial is also displayed.

When the user selects the User in Mode module then user can write their own source codes in the C-Editor for the CRC-12 or CRC-16 or CRC-32 or CRC-CCITT polynomials and execute the code in C-Compiler and user gets the .exe file after executing the CRC-12 or CRC-16 or CRC-32 or CRC-CCITT code. Then the selection the CRC-12 or CRC-16 or CRC-32 or CRC-CCITT polynomials in Algorithm module gives the .exe file as input to the NetSim in Input module. The CRC-12 or CRC-16 or CRC-32 or CRC-CCITT codes written by the user will be running in the NetSim. The output will be transmitted on network through LAN and stores this output in the file (this file is present in the application path of the NetSim) on another system. The animation about the CRC-12 or CRC-16 or CRC-32 or CRC-CCITT polynomial is also displayed.

In Input mode, the user can give input either .txt file format or .exe file format only to the NetSim. Here the sizes of the files are fixed to 5000 bytes. If the user gives more than this size then an error message will be displayed. For the loading of the file to the NetSim, we created one browse button. By using this button user can browse the .txt file or .exe file and can give this file as input to the NetSim. If the user can select the Sample mode in the Mode module then they can give .txt file as input to the NetSim and if they can select the User mode in the Mode module then they can give .exe file as input to the NetSim.

5. Conclusion

In this thesis dissertation, "Design and Implementation of Error Detection Mechanism in NetSim" under Java environment using Java Swings and C language, an attempt has been made to facilitate the user to select different modes on different modules in NetSim for the LRC and CRC techniques according to the user requirements. If the user wants to know about how the LRC and CRC techniques are working, then the user can select the Sample mode and can give appropriate inputs to NetSim and if the user wants to write his own source codes then he can select the User mode and can give executable file as input to the NetSim.

This thesis was started with the requirement analysis in which collecting the materials from the books, internet and other sources. The designing and the system implementation comprises the developing of source codes for the LRC and CRC techniques. After this deployed the codes of the LRC and CRC techniques to NetSim software and found it is working satisfactory as per the requirement.

6. References

[1] Dennis McGrath, Doug Hill, Amy Hunt, Mark Ryan, and Timothy Smith "NetSim: A Distributed Network Simulation to Support Cyber Exercises" Institute for Security Technology Studies, Dartmouth College 2005, [Online]. Available FTP: http://www.ists.dartmouth.edu/library/59.pdf

[2] W. Todd Sneed, Van Miller, Brian Baker "TVA Browns Ferry Simulator EHC System Upgrade Using Woodward's NetSim™ Simulation Package" nHance Technologies, Lynchburg, VA 2008, [Online]. Available FTP: http://www.nhancetech.com/nht_web.nsf/vwTechPapers/TVABrownsFerrySimulatorEHCSystemUpgradeUsingWoodwardsNetSimSimulationPackage/$FILE/BFNetsim.pdf!OpenElement

[3] Dr. Alptekin Erkollar1 and MBM Birgit J. Oberer," The NETSIM Concept and Global Optimization" 1Department of Business Organization and Business Informatics, University of Applied Sciences Wiener Neustadt, Austria, 2006, [Online]. Available FTP: http://www.citeseerx.ist.psu.edu/10.1.1.131.3604.pdf

[4] Erkollar, A "The NETSIM Modeling Concept" In (M. Engeli, V. Hrdliczka ed.) Proc. ASIM'98, Zurich, Switzerland, 1998, Germany, 1996, pp. 323-330.

[5] Stigge, Martin; Plötz, Henryk; Müller, Wolf; Redlich, Jens-Peter (May 2006). *Reversing CRC – Theory and Practice*. Berlin: Humboldt University Berlin. pp. 24. http://sar.informatik.hu-berlin.de/research/publications/SAR-PR-2006-05/SAR-PR-2006-05_.pdf. Retrieved 21 July 2009.