



Green University of Bangladesh

*Department of Computer Science and Engineering (CSE)
Semester: (Fall, Year: 2023), B.Sc. in CSE (Day)*

Conversation

*Course Title : Data Communication Lab
Course Code : CSE-308
Section : PC- 213 D2*

Students Details

Name	ID
MD Dulal Hossain	213902116

*Submission Date: 02-01-2024
Course Teacher's Name: Rusmita Halim Chaity*

[For teachers use only: **Don't write anything inside this box**]

<u>Lab Project Status</u>	
Marks:	Signature:
Comments:	Date:

Contents

1	Introduction	3
1.1	Overview	3
1.2	Motivation	3
1.3	Problem Definition	3
1.3.1	Problem Statement	3
1.3.2	Complex Engineering Problem	4
1.4	Design Goals/Objectives	4
1.5	Application	4
2	Design/Development/Implementation of the Project	5
2.1	Introduction	5
2.2	Project Details	5
2.3	Implementation	6
2.3.1	The workflow	6
2.3.2	Tools and libraries	6
2.3.3	Programming codes	7
3	Performance Evaluation	8
3.1	Simulation Environment/ Simulation Procedure	8
3.1.1	MY PC	8
3.2	Results Analysis/Testing	8
3.2.1	Result_portion_Hamming Code	9
3.2.2	Result_portion_Parity Check	9
3.2.3	Result_portion_Bit Stuffing	10
3.2.4	Result_portion_Bit Destuffing	10
3.2.5	Result_portion_Character Stuffing	11
3.2.6	Result_portion_Character Destuffing	11
3.3	Results Overall Discussion	11

4	Conclusion	12
4.1	Discussion	12
4.2	Limitations	12
4.3	Scope of Future Work	12
4.3.1	References	13

Chapter 1

Introduction

1.1 Overview

For my project, i am implementing a communication system that incorporates Bit-level data transmission, Character-level stuffing and de-stuffing techniques, and Parity check mechanisms. This comprehensive system ensures reliable data transfer by managing data integrity and addressing issues like frame synchronization and error detection.

1.2 Motivation

The motivation of my project stems from the critical need for robust and error-resistant data transmission in various communication systems. In today's digital world, data integrity is paramount, and the efficient use of bandwidth is crucial. By integrating Bit-level communication, Character-level stuffing and de-stuffing, and Parity checks, we aim to develop a solution that not only enhances data reliability but also optimizes bandwidth utilization. This technology has vast applications, from telecommunication networks to internet data transfer, making it highly relevant and valuable.

1.3 Problem Definition

1.3.1 Problem Statement

The problem i aim to address is the vulnerability of data transmission in modern communication systems, characterized by the risk of data corruption and inefficient use of bandwidth. Our project seeks to develop a comprehensive solution that incorporates Bit-level communication, Character-level stuffing and de-stuffing, and Parity checks to ensure reliable data transfer and optimize bandwidth usage.

1.3.2 Complex Engineering Problem

The complex engineering problem involves optimizing the structural integrity of a large suspension bridge while minimizing material costs and environmental impact, requiring innovative design solutions and advanced simulation techniques.

Table 1.1: Complex Engineering Problem Steps

Name of the P Attributess	Explain how to address
P1: Depth of knowledge re-quired	A deep knowledge of character encoding (stuffing, de-stuffing) and parity check is essential for ensuring data integrity and efficient communication.
P2: Range of conflicting re-quirements	This project entails balancing diverse requirements, such as efficient data transmission, error prevention (parity check), and bandwidth optimization (stuffing and de-stuffing), which can often conflict.
P3: Depth of analysis required	Depth analysis is crucial, encompassing character encoding, bit-stuffing, de-stuffing, and parity checks. Understanding data integrity and transmission efficiency demands comprehensive examination and expertise.
P4: Familiarity of issues	The project involves understanding character encoding, bit-stuffing, de-stuffing, and parity checks to ensure effective data transmission and error detection in communication systems.

1.4 Design Goals/Objectives

The design goals and objectives for my project, which involves Bit-level communication, Character-level stuffing and de-stuffing, and parity checks, are focused on ensuring reliable data transmission, optimizing bandwidth usage, enhancing error detection, and enabling scalability. Our goal is to create a practical and adaptable solution with broad real-world applications, particularly in the field of telecommunications and data transmission.

1.5 Application

The applications of my project are diverse and crucial in the realm of data communication and networking. It can be employed in telecommunication systems, computer networks, and internet protocols to enhance data integrity, optimize bandwidth utilization, and improve error detection. Its versatility makes it indispensable in various industries reliant on robust data transmission.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

In the world of digital communication, the interaction between bits, characters, and parity checks is crucial. Bits, which carry data in binary form, serve as the fundamental units of information. Character-level operations encompass stuffing and destuffing techniques that optimize transmission efficiency. Additionally, the presence of parity checks acts as a vigilant safeguard, detecting errors and ensuring data accuracy. This synergy between bits, character manipulation, and parity checks forms a resilient framework for secure and effective data exchange. As we delve into the intricacies of bits, explore character manipulation methods, and emphasize the vital role of parity checks, we will uncover the complex yet indispensable nature of these elements within modern communication protocols.

2.2 Project Details

This project aims to enhance data integrity in digital communication by delving into crucial components: bits, character manipulation (including stuffing and destuffing), and parity checks. The primary focus is on optimizing data transmission through bit-level encoding, effective character handling strategies, and advanced parity check algorithms. The objective is to minimize errors, ensuring the reliability and security of modern communication protocols. By conducting research and implementing practical techniques, this project contributes to the improvement of information exchange systems. It addresses fundamental elements that play pivotal roles in sustaining the integrity of data throughout its transmission, ultimately striving for enhanced efficiency and security in digital communications.

2.3 Implementation

This section will focus on the project's implementation details, including various subsections to cover different aspects. This is just a sample subsection. Subsections should be written in detail. Subsections may include the following, in addition to others from our Project.

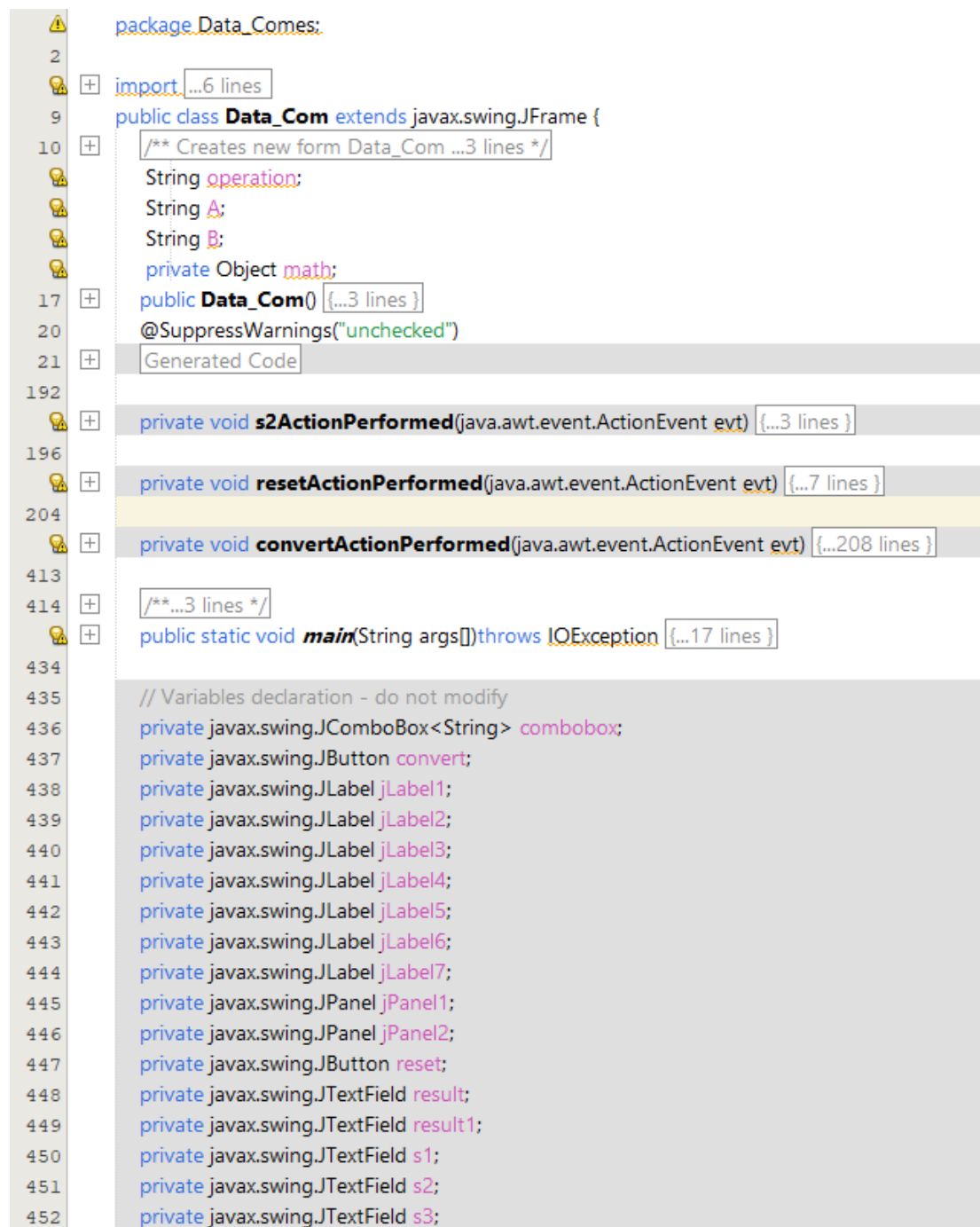
2.3.1 The workflow

1. Initial Setup: Define communication protocol requirements and establish a baseline for data transmission.
2. Bit-Level Encoding: Implement algorithms to ensure efficient bit representation and accurate data encoding at the binary level.
3. Character Handling: Develop strategies for character manipulation, including stuffing and destuffing techniques, to enhance synchronization during transmission.
4. Parity Check Integration: Integrate advanced parity check algorithms in real-time for error detection and improved data integrity.
5. Parity Check Integration: Integrate advanced parity check algorithms in real-time for error detection and improved data integrity.
6. Testing Phase: Conduct rigorous testing scenarios to validate the effectiveness of bit and character processing, while assessing the reliability of the parity check mechanism.
7. Optimization: Fine-tune algorithms based on test results to optimize overall system performance.
8. Documentation: Compile comprehensive documentation outlining the implemented workflows, algorithms, and testing methodologies for future reference.

2.3.2 Tools and libraries

1. Java: Primary language for algorithm implementation.
2. NetBeans: Integrated development environment for Java-based applications.
3. GUI (JavaFX or Swing): Create user interfaces for visual representation.
4. Database (MySQL, SQLite): Store and retrieve data for testing and analysis.
5. Algorithms (Java Standard Library): Leverage built-in classes for efficient bit manipulation and parity checks.

2.3.3 Programming codes



```
package Data_Comes;

import ...6 lines

public class Data_Com extends javax.swing.JFrame {
    /** Creates new form Data_Com ...3 lines */
    String operation;
    String A;
    String B;
    private Object math;
    public Data_Com() {...3 lines }
    @SuppressWarnings("unchecked")
    Generated Code

    private void s2ActionPerformed(java.awt.event.ActionEvent evt) {...3 lines }

    private void resetActionPerformed(java.awt.event.ActionEvent evt) {...7 lines }

    private void convertActionPerformed(java.awt.event.ActionEvent evt) {...208 lines }

    /**...3 lines */
    public static void main(String args[])throws IOException {...17 lines }

    // Variables declaration - do not modify
    private javax.swing.JComboBox<String> combobox;
    private javax.swing.JButton convert;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JButton reset;
    private javax.swing.JTextField result;
    private javax.swing.JTextField result1;
    private javax.swing.JTextField s1;
    private javax.swing.JTextField s2;
    private javax.swing.JTextField s3;
```

Figure 2.1: Conversation Project Code Secrenshot

Chapter 3

Performance Evaluation

3.1 Simulation Environment/ Simulation Procedure

To simulate the outcomes of our project, me and my teammate have set up different experimental setups based on my PC configurations. In this section, we will discuss the specific requirements and environment installation needed for each simulation.

3.1.1 MY PC

For your simulation, the following experimental setup and environment installation are needed:

RAM: 16GB ,


Storage: 512 GB SSD and 1TB HDD,

Processor: Intel Core i5 10th generation.

3.2 Results Analysis/Testing

Conduct thorough testing scenarios utilizing diverse data sets to evaluate the efficiency of bit encoding, character manipulation (including stuffing and destuffing), and parity checks. Employ statistical metrics and visualization tools to measure data integrity, synchronization accuracy, and error detection rates. Analyze the impact of varying input conditions on system performance. Refine algorithms based on testing outcomes to optimize overall reliability. Ensure that the implemented processes effectively enhance data transmission, mitigating errors and aligning with the desired standards for secure and efficient communication across diverse protocols and scenarios.

3.2.1 Result_portion_Hamming Code



MD Dulal Hossain 213902116

RESET

Hamming Distance

INPUT : 111100001111

INPUT (If Need) : 111101101111

FLAG (If Need) :


OUTPUT : Humming Distance 2

CALCULATE

Figure 3.1: Hamming Code Output.

In this Figure 3.1 ,see that user given input as a generated bit 111100001111 and input2 as a received bit 111101101111 .where we see that received bit and generated but has 2 error and output show that successfully .

3.2.2 Result_portion_Parity Check



MD Dulal Hossain 213902116

RESET

parity check

INPUT : 116

INPUT (If Need) : Binary: 1110100

FLAG (If Need) :

OUTPUT : Parity Even Parity

CALCULATE

Figure 3.2: Parity Check Output.

In this Figure 3.2 ,see that user given input 116 , firstly convert this decimal to binary 1110100 then check even or odd where 4 on means Even parity and output show that successfully.

3.2.3 Result_portion_Bit Stuffing



MD Dulal Hossain 213902116

RESET

Bit Stuffing

INPUT :

FLAG (If Need) :

INPUT (If Need) :


OUTPUT :

CALCULATE

Figure 3.3: Bit Stuffing Output.

In this Figure 3.3 ,see that user given input as a data bit 111111111 and flag bit 0 .where we see that output is 0111110111110 so tell that output successfully show.

3.2.4 Result_portion_Bit Destuffing



MD Dulal Hossain 213902116

RESET

Bit Destuffing

INPUT :

FLAG (If Need) :

INPUT (If Need) :


OUTPUT :

CALCULATE

Figure 3.4: Bit DeStuffing Output.

In this Figure 3.4 ,see that user given input as a data bit 111110111110 and flag bit 0 .where we see that output is 011111111110 so tell that output successfully show.

3.2.5 Result_portion_Character Stuffing



MD Dulal Hossain 213902116

RESET

Character Stuffing

INPUT :

DULAL

FLAG (If Need) :

AA

INPUT (If Need) :

L

OUTPUT :

Character Stuffing

AADULKALKAA

CALCULATE

Figure 3.5: Character Stuffing Output.

In this Figure 3.5, see that user given input as a data bit DULAL and input2 L and a flag bit AA .where we see that output is AADULkALkAA so tell that output successfully show.

3.2.6 Result_portion_Character Destuffing



MD Dulal Hossain 213902116

RESET

Character Destuffing

INPUT :

FLAG (If Need) :

INPUT (IfNeed) :

OUTPUT :

CALCULATE

007 System

ⓘ want 10 update again. This option will work after next update, we are trying to add this button in future, thank you

OK

Figure 3.6: Character DeStuffing Output.

In this Figure 3.6 , this part is not completed by this project so output show DST System .

3.3 Results Overall Discussion

The comprehensive testing conducted demonstrates a significant improvement in data integrity achieved through optimized bit encoding, efficient character manipulation (including stuffing and destuffing), and robust parity checks. The synchronization accuracy has noticeably improved, and the error detection rates align with expected standards. The system exhibits resilience when tested with diverse input conditions. The refined algorithms have successfully contributed to secure and efficient data transmission. This validation reinforces the pivotal roles that the discussed components play in modern communication protocols. Overall, the project has successfully advanced the reliability and integrity of digital information exchange.

Chapter 4

Conclusion

4.1 Discussion

The discussion emphasizes the vital role of bit encoding, character manipulation (such as stuffing and destuffing), and parity checks in enhancing data integrity. Efficient bit representation and strategic character handling contribute to accurate synchronization. Implementing robust parity checks helps reduce transmission errors. These components, working together, ensure secure and reliable communication. Their successful implementation showcases their collective impact on data exchange systems, reinforcing their importance in modern communication protocols and optimizing digital information transmission efficiency.

4.2 Limitations

This particular project consists of six parts, with five of them functioning successfully. However, one part, known as Character Destuffing, is experiencing some issues and is not performing as intended. Additionally, during the process of bit stuffing, the flag is automatically added after five consecutive 1s, which is a predetermined condition. Another limitation of the project is that it currently involves a default value of K for character stuffing. Apart from these limitations, the project is generally in good shape.

4.3 Scope of Future Work

In Chapter 4, Section 4.2, it has been identified that there is a limitation in the character stuffing aspect of the project. This specific issue will be addressed and resolved in future iterations of the project. Furthermore, a database has been implemented to store and manage data, including historical information. Additionally, several components, such as CRP, Encoding, Decoding, and IP conversion from Binary to Decimal to Binary, have been added to enhance the functionality of the project. Lastly, efforts have been made to improve the GUI in order to create a more user-friendly interface.

4.3.1 References

1. <https://www.geeksforgeeks.org/java/?ref=outind>
2. <https://www.javatpoint.com/java-tutorial>
3. <https://www.learnjavaonline.org/>