Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Spring,Year:2025),B.Sc.in CSE (Day)

LAB REPORT NO - 2

Course Title: Data Communication Lab

Course Code:CSE307 Section:223-D1

Lab Experiment Name : Implementing Bit Stuffing and De-stuffing

Student Details

	Name	ID
1.	MD.SHAJALAL	223002088

Lab Date : 24-02-2025

Submission Date : 03-03-2025

Course Teacher's Name : Md.Samin Hossian Utsho

Lab Report Status

Marks: …… **Signature:.....**

Comments:..... **Date:.....**

1. TITLE OF THE LAB REPOT EXPERIMENT

Implementing Bit Stuffing and De-stuffing

2.Objective: To implement Bit Stuffing and De-stuffing using the C programming language.

3. Theory: Bit Stuffing is a data transmission technique used to prevent confusion with control flags in synchronous communication. When five consecutive '1' bits appear in the data, a '0' bit is inserted after them to differentiate from the flag sequence. De-stuffing is the reverse process, where these extra '0' bits are removed at the receiver's end to retrieve the original data.

4.Algorithm:

Bit Stuffing Algorithm:

1. Traverse the input bit stream.
2. Count consecutive '1's.
3. If five consecutive '1's are encountered, insert a '0'.
4. Continue until the end of the input.

Bit De-stuffing Algorithm:

1. Traverse the stuffed bit stream.
2. Count consecutive '1's.
3. If five consecutive '1's are encountered followed by a '0', remove the '0'.
4. Continue until the end of the input.

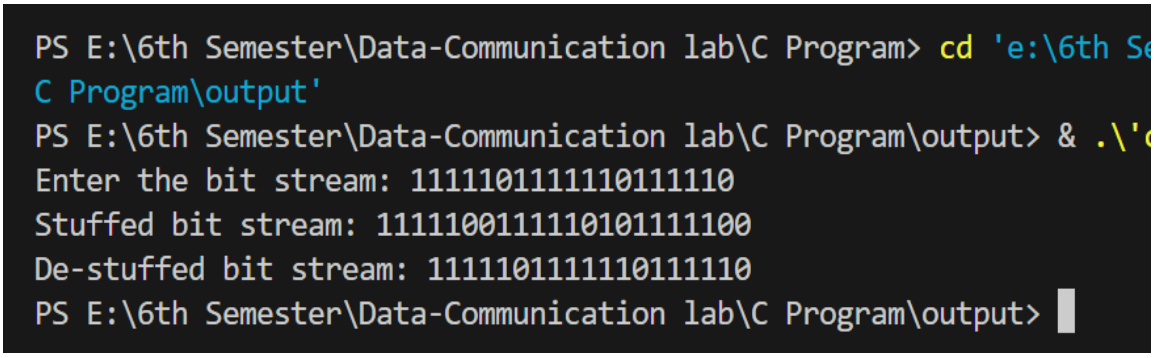
5. C Program:

Code:

```c
 #include <stdio.h>

#include <string.h>

void bitStuffing(char *input, char *stuffed) {
    int i, j = 0, count = 0;
    for (i = 0; input[i] != '\0'; i++) {
        stuffed[j++] = input[i];
        if (input[i] == '1') {
            count++;
```

```
            if (count == 5) {
                stuffed[j++] = '0';
                count = 0;
            }
        } else {
            count = 0;
        }
    }
    stuffed[j] = '\0';
}

void bitDeStuffing(char *stuffed, char *original) {
    int i, j = 0, count = 0;
    for (i = 0; stuffed[i] != '\0'; i++) {
        original[j++] = stuffed[i];
        if (stuffed[i] == '1') {
            count++;
            if (count == 5 && stuffed[i + 1] == '0') {
                i++;
                count = 0;
            }
        } else {
            count = 0;
        }
    }
    original[j] = '\0';
}

int main() {
    char input[100], stuffed[150], original[100];

    printf("Enter the bit stream: ");
    scanf("%s", input);

    bitStuffing(input, stuffed);
    printf("Stuffed bit stream: %s\n", stuffed);

    bitDeStuffing(stuffed, original);
    printf("De-stuffed bit stream: %s\n", original);

    return 0;
}
```

6.Output:



7.Conclusion: The Bit Stuffing and De-stuffing techniques were successfully implemented in C. The program correctly inserts and removes extra '0' bits while ensuring proper data transmission. The results verify the correctness of the implementation.