Fr. Conceicao Rodrigues College of Engineering

Department of Computer Engineering

Academic Term: July-Nov 2023-24

Class: T.E. (Computer B)
Subject Name: Computer Network Lab
Subject Code: CSL 502

Experiment No:	3
Date of Performance:	10/08/2023
Roll No:	9614
Name of the Student:	Aqib Firdous Khan

AIM: Implementation of bit stuffing and unstuffing algorithm

Theoretical description:

To avoid the appearance of flag pattern in the data field of the frame; thereby avoiding undue termination of the frame and loss of data bit stuffing and destuffing is used.In HDLC protocol, the flag field delimits the frame at both ends with the unique pattern 01111110. The pattern 01111110 may appear somewhere inside the frame. It may destroy the synchronization. To avoid this problem, a procedure known as bit stuffing is used. For all bits between the starting and ending flags, the transmitter inserts an extra 0 bit after occurrence of five 1s in the frame.

After detecting a starting flag, the receiver monitors the bit stream. When the pattern of five 1s appears, the sixth bit is examined. If this bit is 0, it is deleted. If the sixth bit is 1 and the seventh bit is 0, the combination is accepted as a flag. Thus, with the use of bit stuffing, arbitrary bit patterns can be inserted into the data field of the frame. Figure below shows an example of bit stuffing and unstuffing.

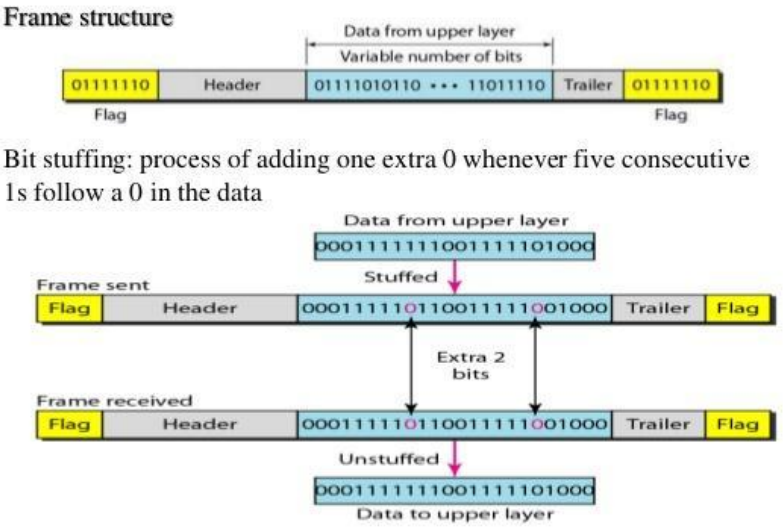


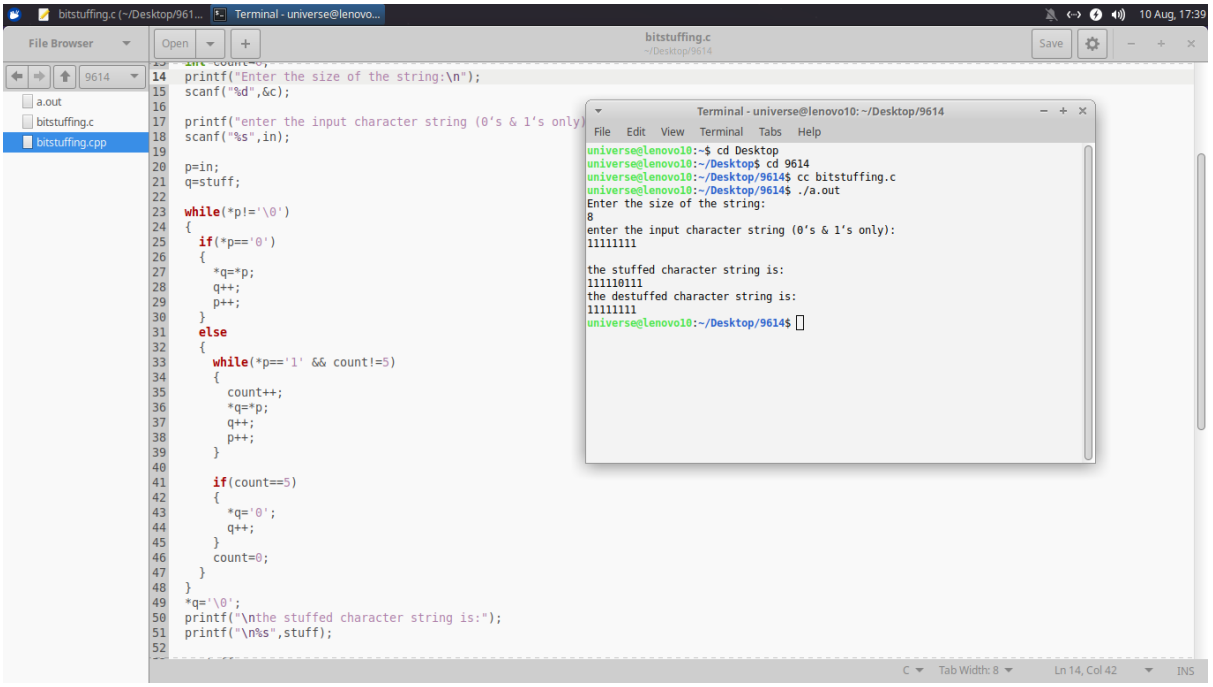
Figure 1: Example of bit stuffing and unstuffing

Write down a program to implement bit stuffing and unstuffing using C/C++/Java/Python and Attach the screenshot of the Program and its output.

CODE:

```
#include<stdio.h>
#include<stdlib.h>
#define MAXSIZE 100

int main()
{
  char *p,*q;
  char temp;
  char in[MAXSIZE];
  char stuff[MAXSIZE];
  char destuff[MAXSIZE];
  int c;
  int count=0;
  printf("Enter the size of the string:\n");
  scanf("%d",&c);

  printf("enter the input character string (0's & 1's only):\n");
  scanf("%s",in);

  p=in;
  q=stuff;

  while(*p!='\0')
  {
    if(*p=='0')
    {
```

```
      *q=*p;
      q++;
      p++;
    }
    else
    {
      while(*p=='1' && count!=5)
      {
        count++;
        *q=*p;
        q++;
        p++;
      }

      if(count==5)
      {
        *q='0';
        q++;
      }
      count=0;
    }
  }
  *q='\0';
  printf("\nthe stuffed character string is:");
  printf("\n%s",stuff);

  p=stuff;
  q=destuff;
  while(*p!='\0')
  {
    if(*p=='0')
    {
      *q=*p;
      q++;
      p++;
    }
    else
    {
      while(*p=='1' && count!=5)
      {
        count++;
        *q=*p;
        q++;
        p++;
      }
      if(count==5)
      {
        p++;
      }
      count=0;
    }
  }
  *q='\0';
  printf("\nthe destuffed character string is:");
  printf("\n%s\n",destuff);
  return 0;

}
```



Conclusion:

Post lab questions:

1. Explain the HDLC protocol of data link later in detail.

Ans: HDLC, or High-Level Data Link Control, is a widely used protocol at the data link layer (Layer 2) of the OSI model. It's a bit-oriented protocol that's used for point-to-point and multipoint communication links. Originally developed by the International Organization for Standardization (ISO), HDLC has become a foundation for many other protocols, including LAPB (Link Access Procedure, Balanced) and PPP (Point-to-Point Protocol).

Here are the key features and aspects of the HDLC protocol:

1. Frame Structure: HDLC frames are made up of a header, data, and a trailer. The header contains control information, such as addressing and flow control, while the trailer includes error-checking mechanisms.

Frame Types:

- Information Frames (I-Frames): Carry user data and control information

for error detection and flow control.
- Supervisory Frames (S-Frames): Used for flow control and error control.
- Unnumbered Frames (U-Frames): Carry control information like link establishment, maintenance, and termination.

2. Synchronous Communication: HDLC is synchronous, meaning it uses synchronized clocks at both ends of the communication link. This ensures that data is transmitted in a predictable manner.
Full-Duplex or Half-Duplex Operation: HDLC can operate in both full-duplex (simultaneous two-way communication) and half-duplex (alternating two-way communication) modes.
3. Flow Control:

- Positive Acknowledgment with Retransmission (PAR): The sender waits for an acknowledgment (ACK) from the receiver before sending the next frame. If no ACK is received, the sender retransmits the frame.
- Selective Reject (SREJ): The receiver can request retransmission of specific frames, rather than the whole sequence.

4. Error Control: HDLC provides error detection and correction through various mechanisms, including frame sequencing, checksums, and cyclic redundancy checks (CRC).
Addressing:

- Normal Response Mode (NRM): Used for point-to-point connections, where one device acts as the primary sender and receiver.
- Asynchronous Response Mode (ARM): Used for multipoint connections, where multiple secondary devices communicate with a primary device.

5. Bit Stuffing: To ensure proper frame synchronization, HDLC uses bit stuffing. A '0' bit is added after every five consecutive '1' bits in the data, and the receiver removes the stuffed '0' bits to recover the original data.
Transparency: HDLC supports transparency, which means it can transmit control characters and data without ambiguity, even if they match the control characters used by the protocol.
Connection Establishment and Termination: HDLC supports procedures for establishing, maintaining, and terminating connections between devices.
Variants: HDLC has inspired several variants, such as LAPB used in X.25 networks and PPP used for point-to-point connections, including dial-up and DSL.

HDLC's reliability, efficiency, and versatility have led to its adoption in various applications and networks. While it's still used in some contexts, newer protocols like PPP and its derivatives have gained wider popularity due to their expanded features and compatibility with modern network environments.

2. Differentiate between CSMA/CD and CSMA/CA.

Ans:

Aspect	CSMA/CD (Carrier Sense Multiple Access with Collision Detection)	CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance)
Medium Type	Wired Ethernet networks	Wireless networks
Collision Handling	Detects collisions and handles them by stopping transmission, random backoff	Aims to avoid collisions through coordinated communication and acknowledgments
Detection Mechanism	Listens for signal changes to detect collisions	Listens and waits for interframe space before transmitting
Protocol for Wired Networks	Yes	No (typically not used)
Protocol for Wireless Networks	No (not suitable for wireless)	Yes
ACKnowledgment and Retransmission	No, as collisions are detected and handled	Yes, devices wait for ACKs and retransmit if necessary

Efficiency	More efficient in wired networks, but efficiency decreases with network size	Less efficient due to overhead of RTS, CTS, ACK, and waiting times
Main Focus	Collision detection and backoff mechanisms	Collision avoidance through coordinated communication and acknowledgments