Purpose of Byte Stuffing

In Data Link layer, the stream of bits from physical layer are divided into data frames. The data frames can be of fixed length or variable length. In variable – length framing, the size of each frame to be transmitted may be different. So, a pattern of bits is used as a delimiter to mark the end of one frame and the beginning of the next frame. However, if the pattern occurs in the message, then mechanisms needs to be incorporated so that this situation is avoided.

The two common approaches are −

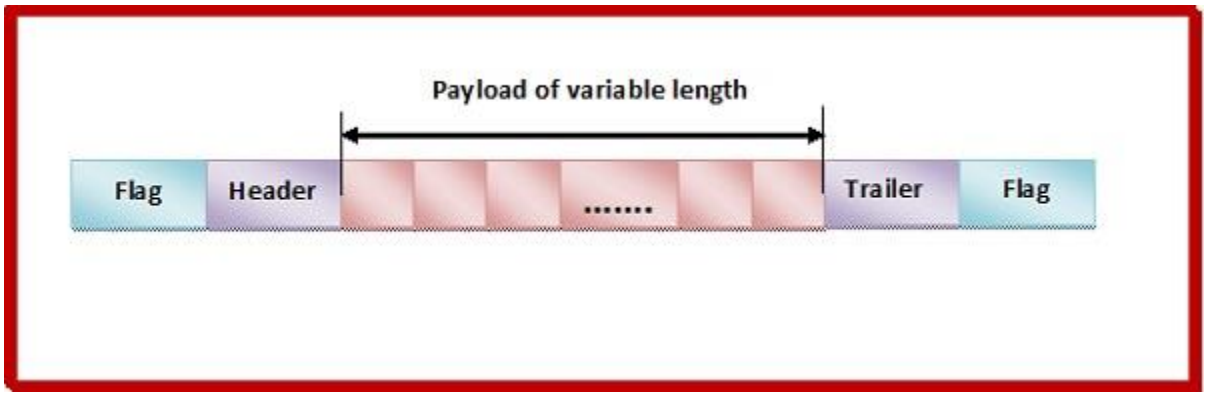
- **Byte – Stuffing** − A byte is stuffed in the message to differentiate from the delimiter. This is also called character-oriented framing.
- **Bit – Stuffing** − A pattern of bits of arbitrary length is stuffed in the message to differentiate from the delimiter. This is also called bit – oriented framing.

Frame in a Character – Oriented Framing

In character – oriented protocols, the message is coded as 8-bit characters, using codes like ASCII codes.

A frame has the following parts −

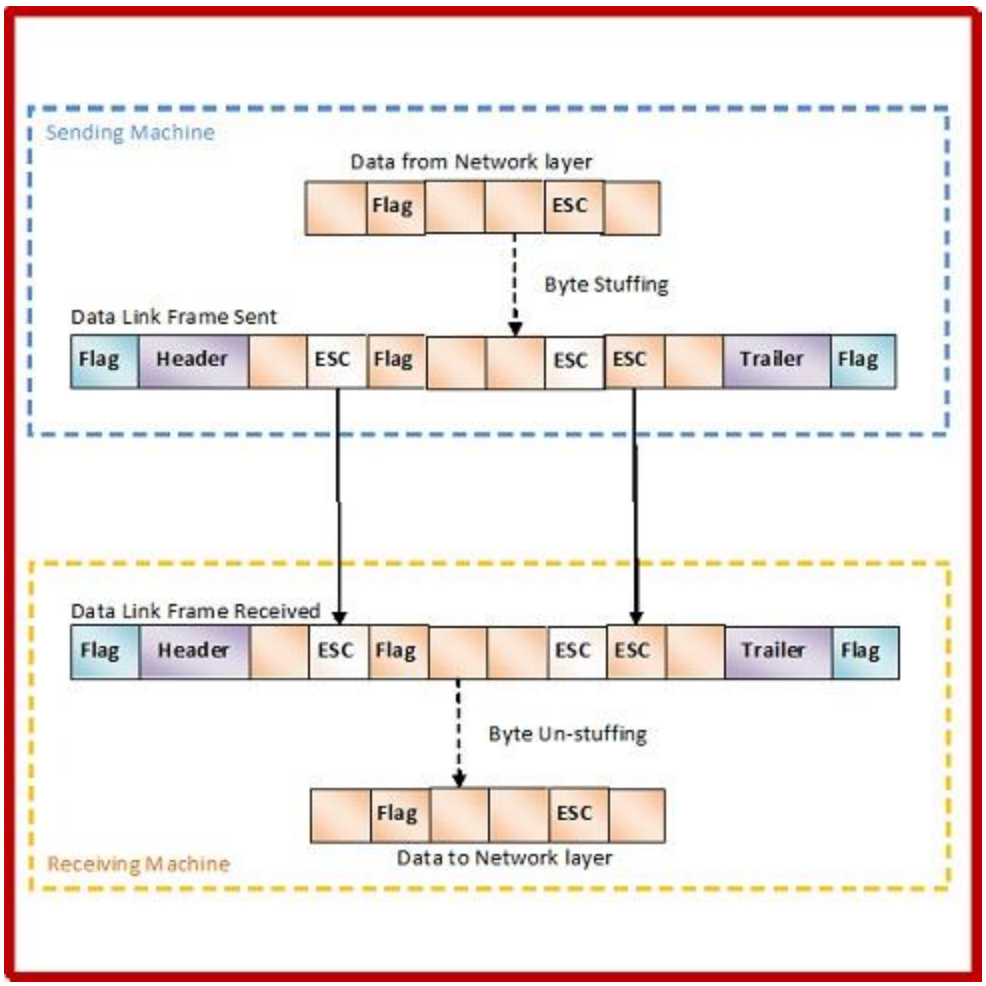
- Frame Header − It contains the source and the destination addresses of the frame.
- Payload field − It contains the message to be delivered.
- Trailer − It contains the error detection and error correction bits.
- Flags − 1- byte (8-bits) flag at the beginning and at end of the frame. It is a protocol – dependent special character, signalling the start and end of the frame.



Byte Stuffing Mechanism

If the pattern of the flag byte is present in the message byte, there should be a strategy so that the receiver does not consider the pattern as the end of the frame. In character – oriented protocol, the mechanism adopted is byte stuffing.

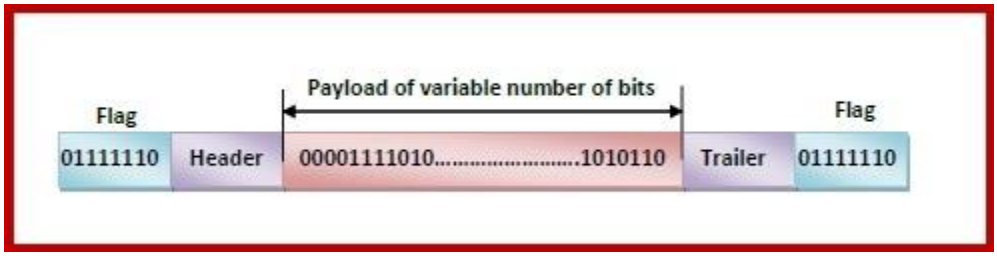
In byte stuffing, a special byte called the escape character (ESC) is stuffed before every byte in the message with the same pattern as the flag byte. If the ESC sequence is found in the message byte, then another ESC byte is stuffed before it.



Frame in a Bit - Oriented Protocol

In bit-oriented protocols, the message is coded as a sequence of bits, which are interpreted in the upper layers as text, graphics, audio, video etc. A frame has the following parts −

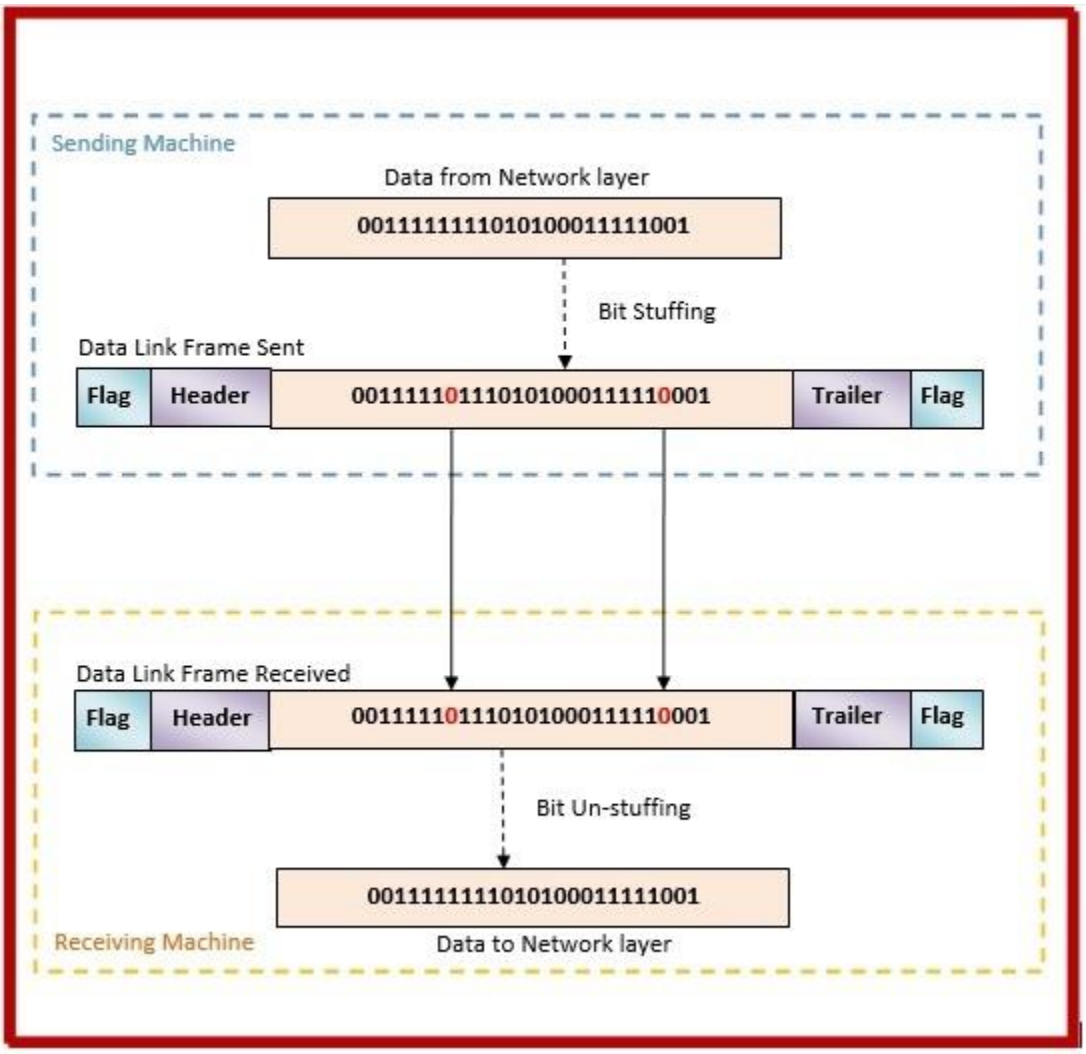
- **Frame Header** − It contains the source and the destination addresses of the frame.
- **Payload field** − It contains the message to be delivered.
- **Trailer** − It contains the error detection and error correction bits.
- **Flags** − A bit pattern that defines the beginning and end bits in a frame. It is generally of 8-bits. Most protocols use the 8-bit pattern 01111110 as flag.



Bit Stuffing Mechanism

In a data link frame, the delimiting flag sequence generally contains six or more consecutive 1s. In order to differentiate the message from the flag in case of the same sequence, a single bit is stuffed in the message. Whenever a 0 bit is followed by five consecutive 1bits in the message, an extra 0 bit is stuffed at the end of the five 1s.

When the receiver receives the message, it removes the stuffed 0s after each sequence of five 1s. The un-stuffed message is then sent to the upper layers.



Mechanisms of byte stuffing versus bit stuffing

Byte Stuffing Mechanism

If the pattern of the flag byte is present in the message byte sequence, there should be a strategy so that the receiver does not consider the pattern as the end of the frame. Here, a special byte called the escape character (ESC) is stuffed before every byte in the message with the same pattern as the flag byte. If the ESC sequence is found in the message byte, then another ESC byte is stuffed before it.

Bit Stuffing Mechanism

Here, the delimiting flag sequence generally contains six or more consecutive 1s. Most protocols use the 8-bit pattern 01111110 as flag. In order to differentiate the message from the flag in case of same sequence, a single bit is stuffed in the message. Whenever a 0 bit is followed by five consecutive 1bits in the message, an extra 0 bit is stuffed at the end of the five 1s. When the receiver receives the message, it removes the stuffed 0s after each sequence of five 1s. The un-stuffed message is then sent to the upper layers.