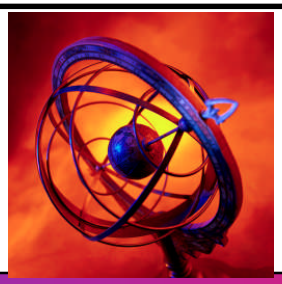




OSI Data Layer
HDLC: High-Level Data Link Control

Professor Richard Harris

School of Engineering and Advanced Technology (SEAT)

Learning Objectives

You will be able to:

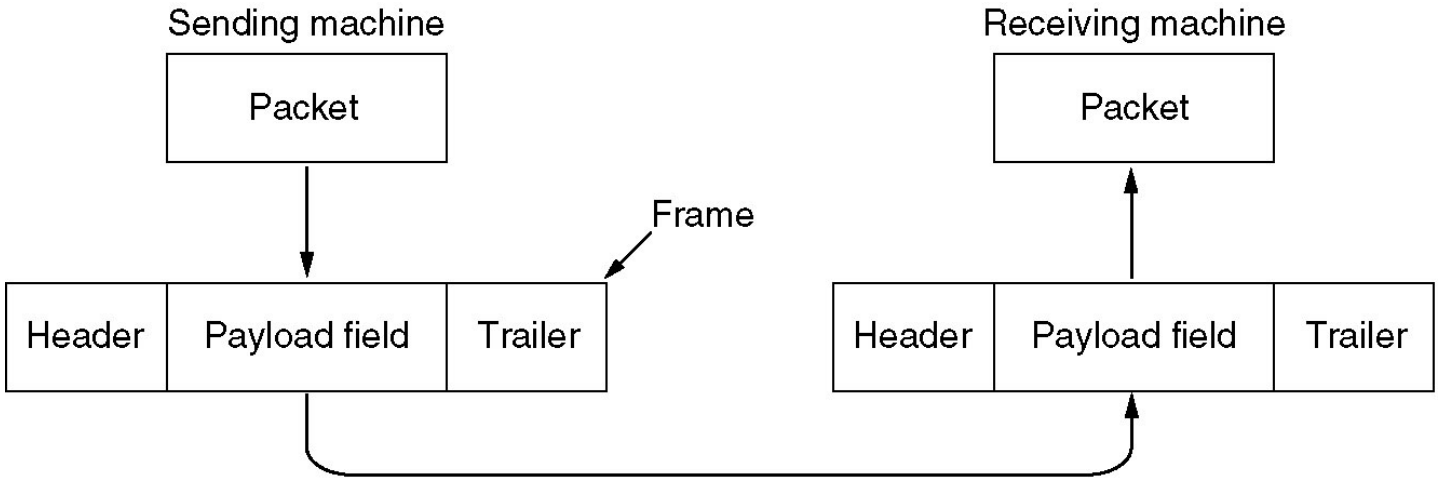
- Discuss the services provided by the Data Link Layer
- Discuss the services provided to the Network Layer
- Explain the differences between connection and connectionless services
- Provide an overview of the HDLC
- Discuss methods of detecting and correcting errors in the Data Link Layer
- Prepare a simple CRC code for a given specification
- Explain what is meant by the parity check sequence and a syndrome and how they are used for error detection and correction

References

- Forouzan, “Data Communications and Networking”, 4th Edition
- Tanenbaum, “Computer Networks”, 4th Edition
- Cisco CCNA1 Module 10 - part 1
- Stallings, William 2000 ‘Data and Computer Communications’, Prentice Hall, Sixth Edition
- Russell, Travis 1997 ‘Telecommunications Protocols’, McGraw Hill

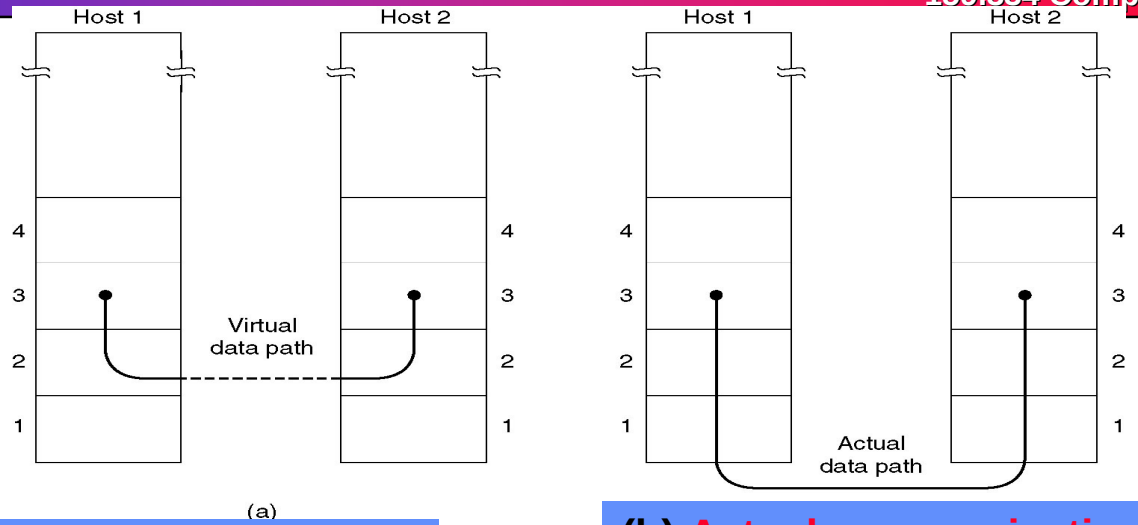
Services Provided to the Network Layer (1)

Relationship between packets and frames.



The primary task of the Data Link Layer is the safe delivery of data handed by the source Network Layer to the Network Layer at the destination

Services Provided to the Network Layer (2)

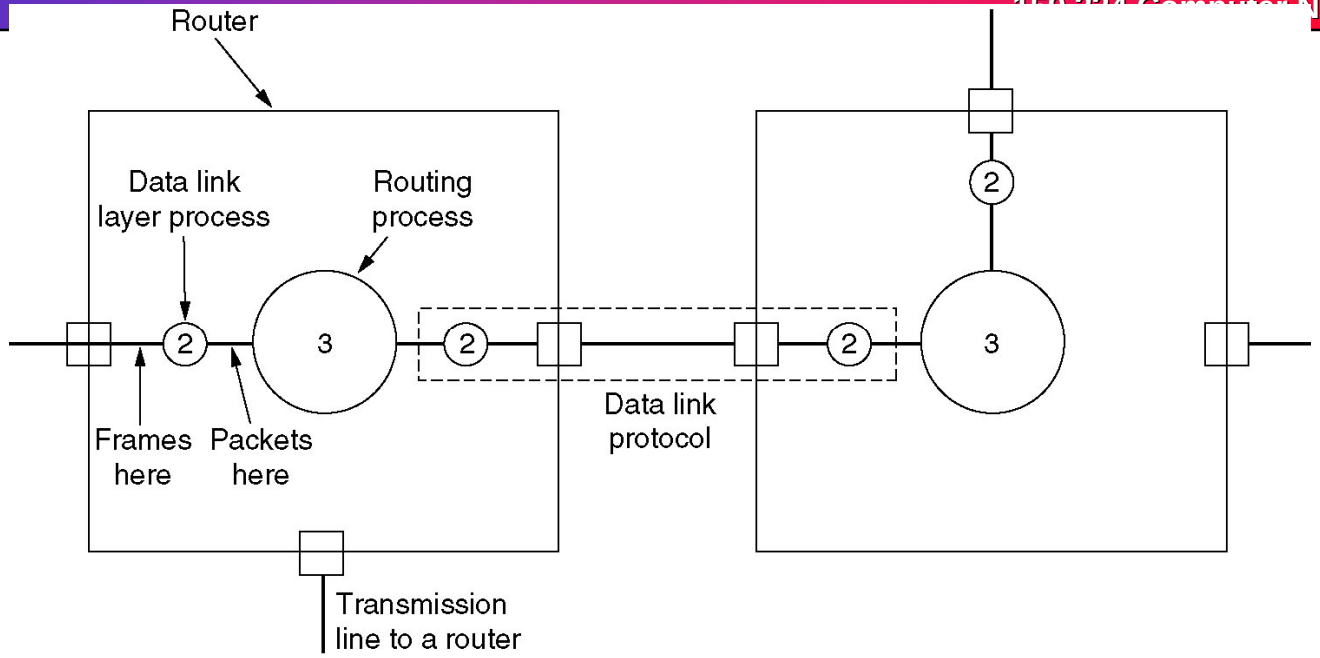


(a) Virtual communication.

(b) Actual communication.

The actual path of the data is as shown in (b) but from the Data Link Layer point of view, the two processes at the source and destination Data Link Layers communicate as shown in (a).

Services Provided to the Network Layer (3)

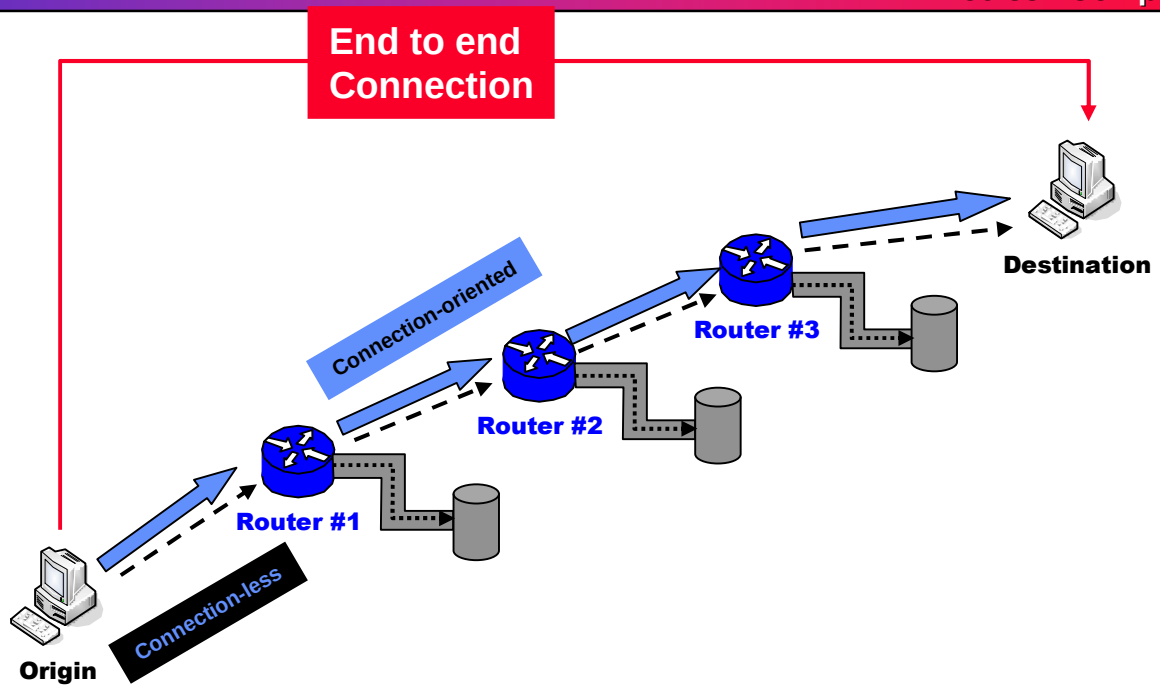


- There are three cases that we shall consider:
  - Unacknowledged connectionless service
  - Acknowledged connectionless service
  - Acknowledged connection-oriented service

Connection-oriented service

- The source and the destination establish a logical connection before any data frames are transmitted by having both sides initialize variables and counters needed to keep track of which frames have been received and which ones have not.
- Each frame sent is numbered and the DLL guarantees the safe delivery of all the data frames in the right order. In addition, one frame will be received only once - unlike in the case of connectionless service where many copies of the same may be sent and received several times, i.e. if the acknowledgements are lost.
- Here, in the connection establishment phase, both sides initialize variables, buffers and counters needed to keep track of data. In the next phase data frames are actually transmitted. Then, connection is released, freeing up the variables, buffers, counters and other resources (for example, transmission media) used to maintain the link, which is the final phase

Comparing Connection and Connectionless Services



Connection-oriented: Requires a session connection (analogous to a phone call) to be established before any data can be sent.

Connectionless: Does not require a session connection between sender and receiver. The sender simply starts sending packets (called datagrams) to the destination.

Unacknowledged Connectionless Service

- The source sends frames independently to the destination but the destination does not acknowledge them
- No logical connection is established for the duration of the transfer of a single data block received from the NL
- If some frames are lost due to noise, recovery is up to the higher layers
- This service is suitable for voice transmission - in which case bad data is preferred to late data or for links using highly reliable physical layers such as optical fibre

Acknowledged Connectionless Service

- Each frame sent by the source Data Link Layer is individually acknowledged by the destination Data Link Layer.
- If a frame has not reached the destination safely, it can be sent again.
- No logical connection is made before or released after the exchange of data

Protocol Types

- Protocols are a set of rules that govern the orderly flow of information between two devices. In serial communications, the information can be transmitted in either synchronous or asynchronous format.
- As the communications system becomes more complex, the need for speed and efficiency rises. Synchronous transmission is generally used. A closer look at synchronous transmission methods and protocols that serve the function of the data link layer (level 2 of the OSI model) is necessary to understand the development and maintenance of data communication systems.
- Synchronous data link protocols can be subdivided into two categories:
  - Bit-oriented protocols
  - Byte-oriented protocols

Bit-oriented Protocols

- In bit-oriented protocols, special groups of uniquely defined bit patterns are used to control the framing, error checking and flow of data between devices.
- The data in bit-oriented protocols may be of any content and may not necessarily represent an encoded character set such as ASCII or EBCDIC.
- An example would be raw data from an A/D converter.
- Special transmission schemes must be employed so that the receiver can distinguish between the actual control characters for framing and error control versus the raw data patterns from the A/D converter that may coincidentally take on the same bit pattern as the control character

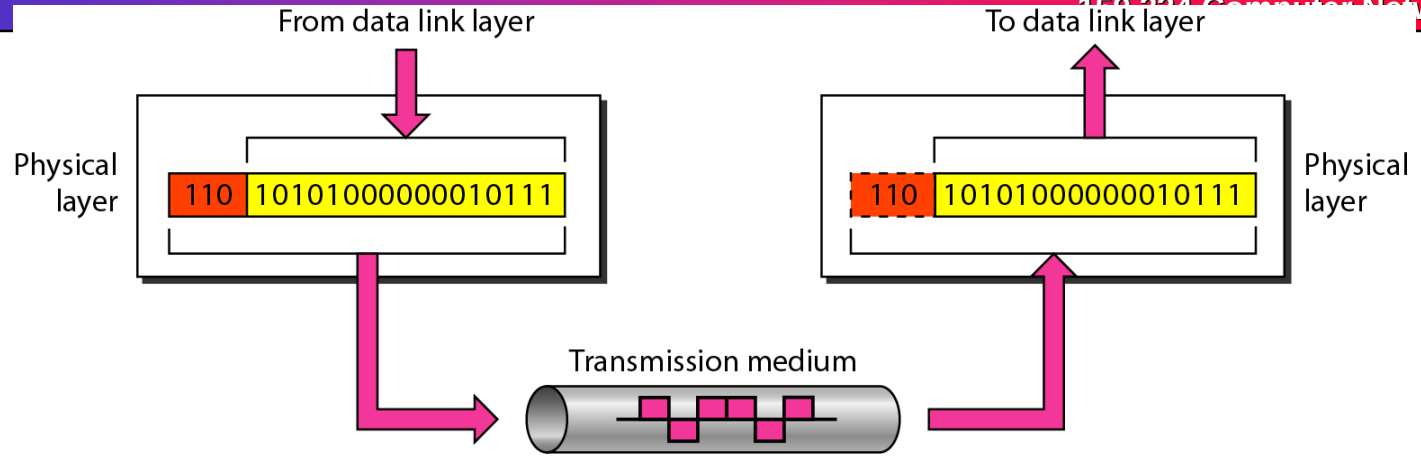
Byte-oriented Protocols

- In byte-oriented protocols, the transmission of data blocks is controlled by ASCII or EBCDIC control characters such as SYN, SOH, and ETX. Control characters, like the data characters, are uniquely defined as part of the ASCII or EBCDIC character set. (See next slide for a list.)
- They are placed at the beginning and end of the transmitted data block for purposes of framing and error control.
- Since the actual data within the block are typically ASCII or EBCDIC, the receiving device can distinguish between data and control.

BISYNC Control Characters

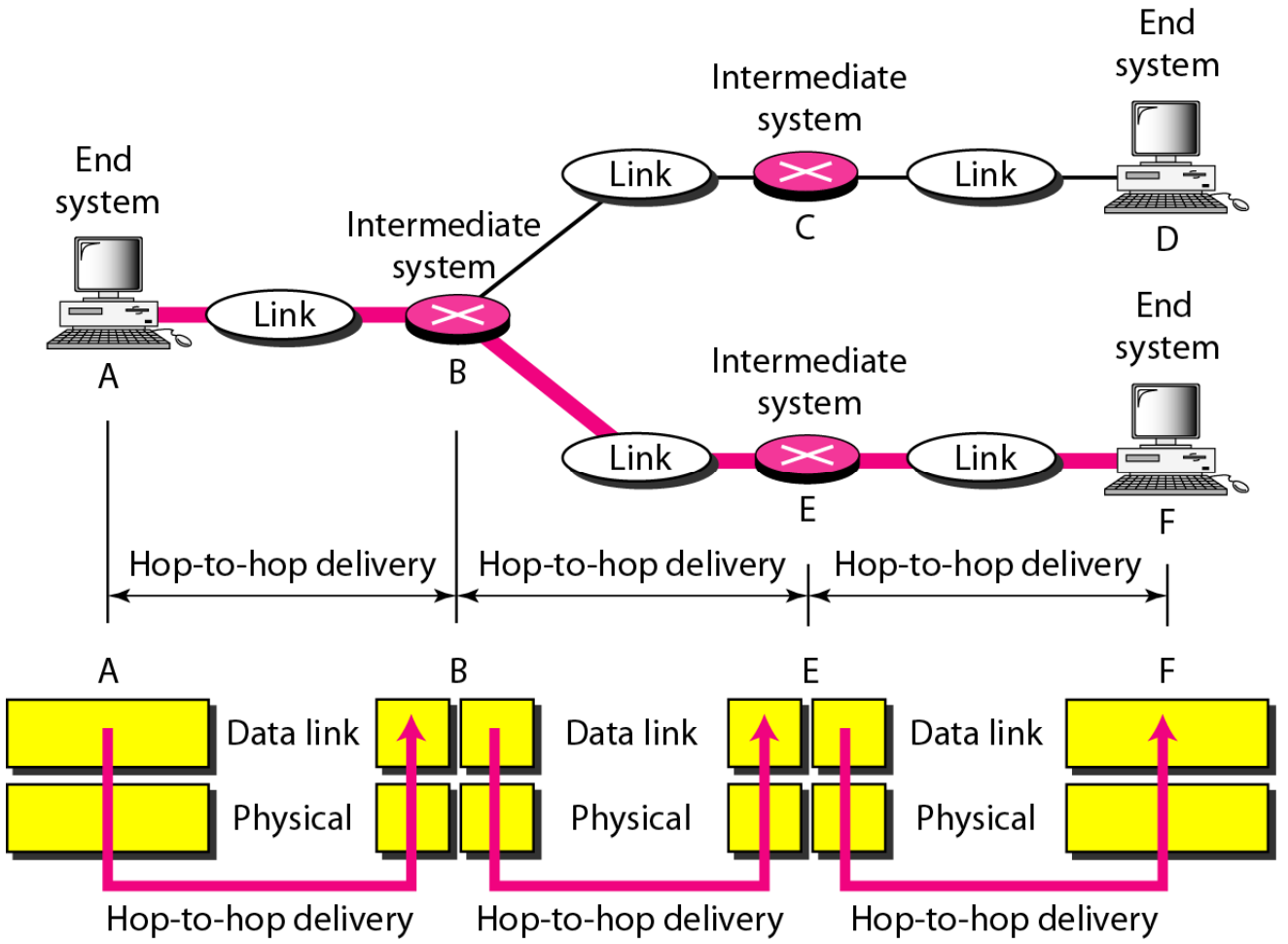
ACK	Affirmative acknowledgement
DLE	Data Link Escape
ENQ	Enquiry
EOT	End of Transmission
ETB	End of Transmission Block
ETX	End of Text
ITB	End of intermediate transmission block
NAK	Negative Acknowledgement
SOH	Start of Header
STX	Start of Text
SYN	Synchronous Idle
WACK	Wait before positive acknowledgement

Protocols at the Data Link Layer

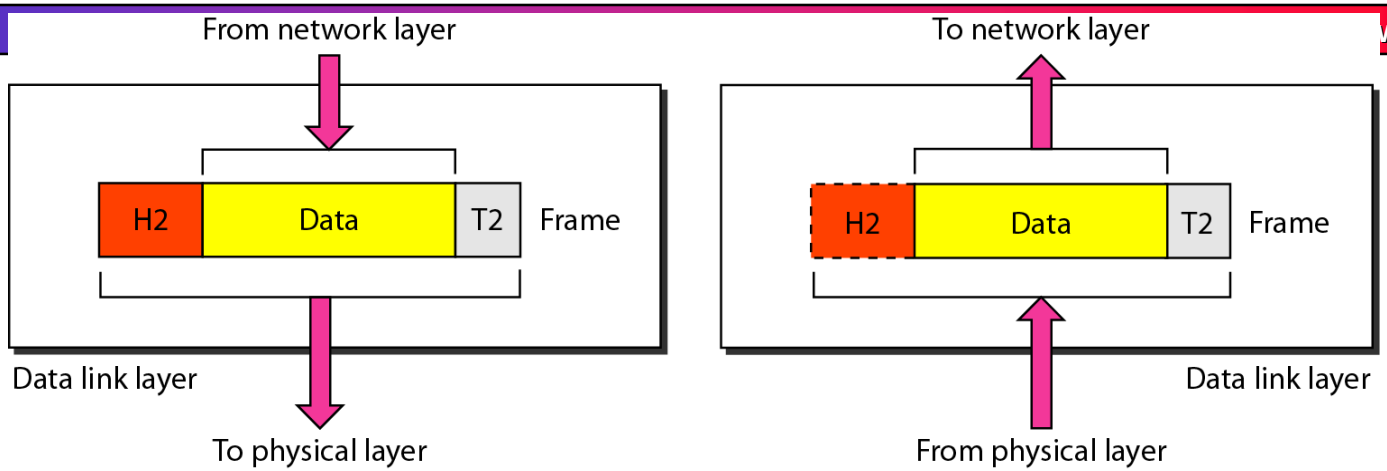


- In order to achieve these goals, some agreement between the two computers or nodes connected by a single transmission medium (twisted pair, radio), must be reached without which we would say no link has been established between two devices
- These agreements or protocols form the Data Link Layer and it is concerned with data transfer between two nodes or devices only
- There are no other devices in between two communicating devices other than repeaters, i.e. in an optical fibre.
- A satellite transponder can be considered a repeater but upward and downward paths use different carrier frequencies, hence they are treated as two data links

Hop to Hop Delivery

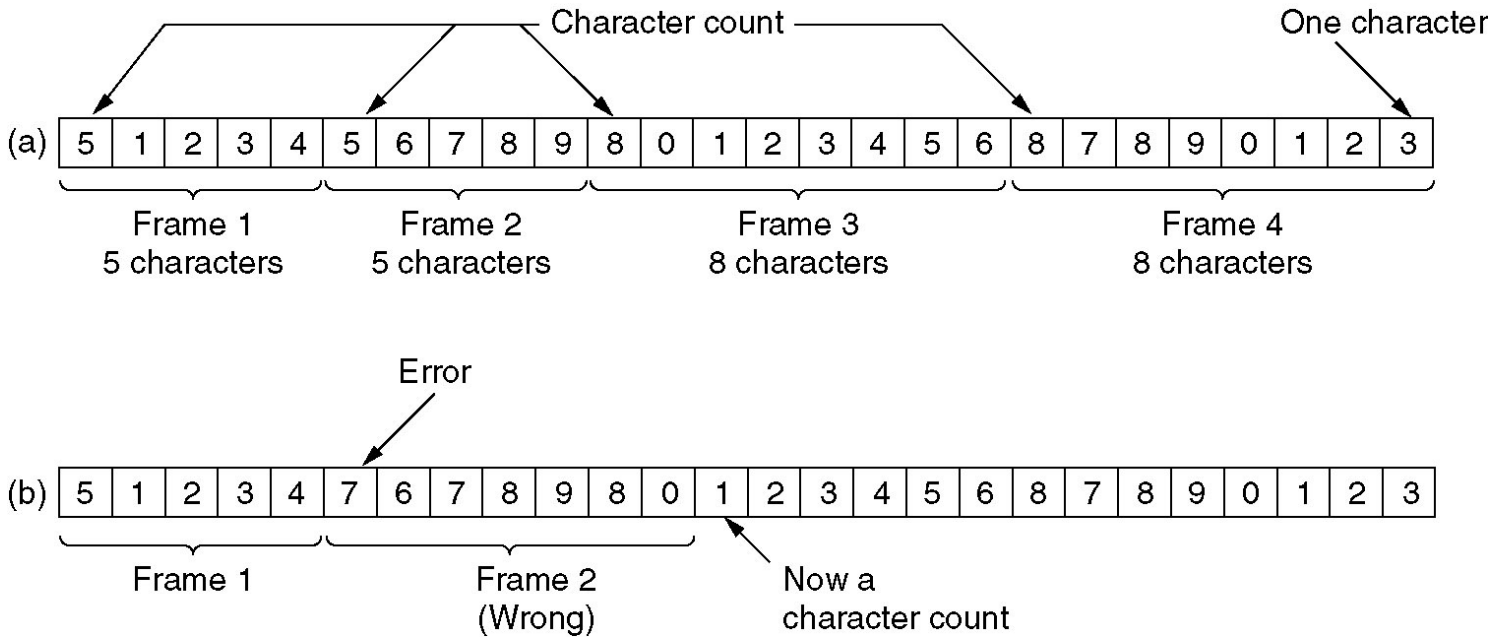


Framing - 1



- To provide required services to the NL, the DLL in turn uses the services of the Physical layer which accepts a raw bit stream and transmits them to the receiver at the other end.
- The physical layer is not concerned whether the bit stream contains data or control information or how they are framed. In addition, there may be bit errors introduced by the channel
- It is up to the data link layer to detect and correct errors and recover all the data that was sent
- What DLL does is to group the data into frames which also contain adequate information to carry out the above tasks
- Therefore, the DLL must use suitable methods for identifying and synchronizing frames

Framing - 2



Generic Frame:

Framing with flag bytes and bit stuffing (1)

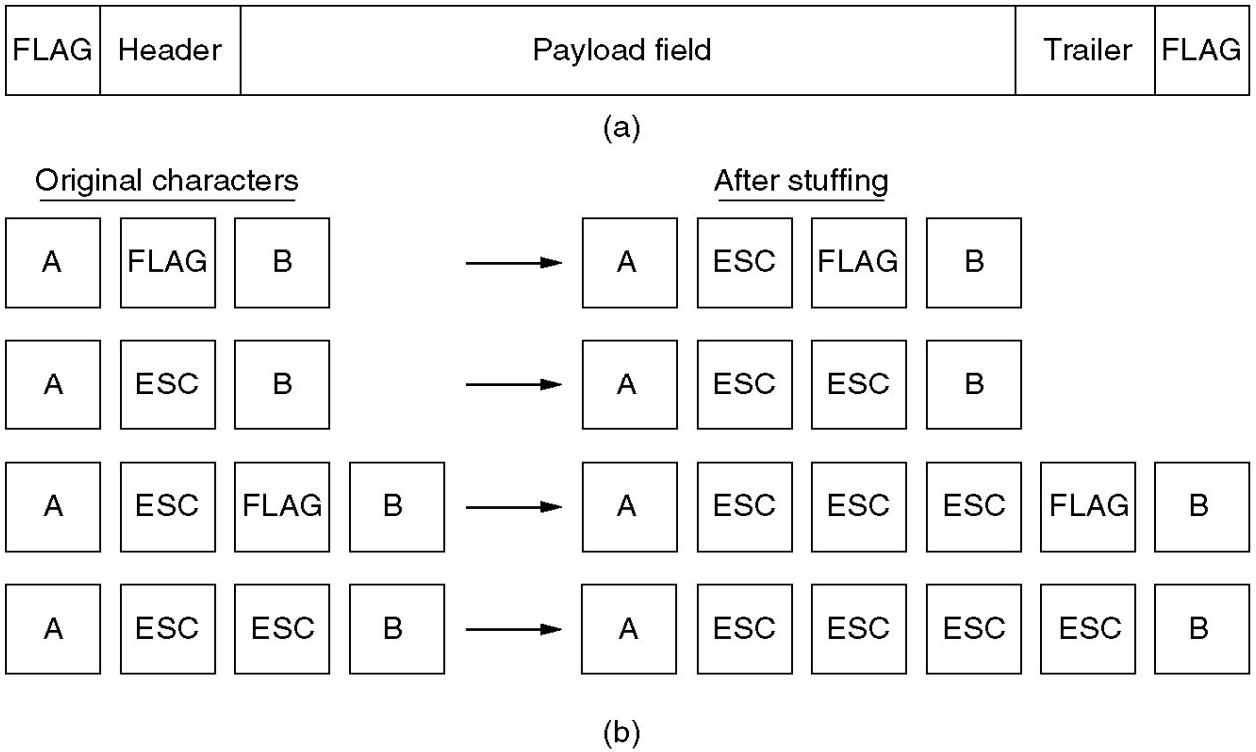
BISYNC Message Block Format

- The block of data and control information appears as a stream of characters, each 8-bits long.
- The frame begins and ends with a unique character called preamble and post-amble. The preamble character is called SYN, and the receiver starts accepting the character stream until the post-amble character is found, which indicates the end of the frame. The receiver then looks for the next SYN pattern.
  - The example above is for the BISYNC protocol
- Disadvantage: tied to the use of 8-bit characters. UNICODE uses 16 bits.

Problem with flag bytes

- It can easily happen that the flag byte's bit pattern occurs in the data.
- Solution:
  - Use byte stuffing, ie. have the sender's data link layer insert a special escape byte (ESC) just before each accidental flag byte in the data.
  - The data link layer at the receiving end removes the escape byte before the data are presented to the network layer.
- Problem:
  - If an escape byte occurs in the middle of the data, it is stuffed with an escape byte too.
    - Thus, any single escape byte is part of the escape sequence; whereas, a double escape byte indicates that a single escape occurred naturally in the data.

Examples of framing using a flag byte



(a) A frame delimited by flag bytes.
(b) Four examples of byte sequences before and after stuffing.

Framing with starting and ending flags with bit stuffing (1)

- Here, the block of data and control information are not interpreted as characters but considered as a sequence of bits
- The preamble and the post-amble is a known sequence referred to as a flag, which is usually 8-bits long
- The receiver (actually the DLL) looks for the flags to locate the beginning and the end of the frame
- A typical flag sequence is '01111110'. However, it is possible that the data itself may contain the above sequence causing misidentification of frame boundaries. This can cause complete loss of frame synchronization
- One way of solving this problem is to, before transmission, add a 0 bit whenever a five consecutive 1's are discovered, which is called bit stuffing. At the receiver, if five consecutive 1s followed by a 0 is encountered, the 0 bit is removed (destuffed)
- Therefore, the flags allow the identification of frame boundaries, as they never occur within the frame as in the figure

Framing with starting and ending flags with bit stuffing (2)

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0

Stuffed bits

(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Bit stuffing

(a) The original data.

(b) The data as they appear on the line.

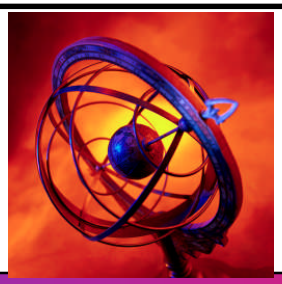
(c) The data as they are stored in receiver's memory after destuffing.

Error Control - 1

- For the connection oriented service, the DLL must guarantee that all the frames sent arrive at the receiver in the proper order, and only one correct copy of each frame is expected at the receiver
- The receiver usually sends an acknowledgement (ACK) or a negative acknowledgement (NACK) back to the transmitter stating whether a received frame arrived safely or not
- These ACKs and NACKs are typically very short in length and do not consume the channel capacity
- To stop the loss of a frame or an ACK/NACK due to channel errors causing the transmitter to wait forever to transmit the next frame, a time out procedure is used. If after certain time, an ACK or NACK is not reached the transmitter, the frame is sent again
- However, if only the ACK is lost, the sender transmits a copy of the same frame causing duplication and waste of channel capacity

Error Control - 2

- By introducing sequence numbers to all frames, passing of a duplicate frame to the NL can be prevented
- The receiver can acknowledge each frame individually which allows the transmitter to continue sending other frames without waiting for an acknowledgement of a particular frame.
- This kind of error control mechanism is referred to as an Automatic Repeat reQuest (ARQ) scheme.


HDLC: High-Level Data Link Control

Professor Richard Harris

School of Engineering and Advanced Technology

Massey University

Wikipedia: HDLC

- **From Wikipedia, the free encyclopedia:**
  - High-Level Data Link Control (HDLC) is a bit-oriented synchronous data link layer protocol developed by the International Organization for Standardization (ISO). The original ISO standards for HDLC were:
    - ISO 3309 — Frame Structure
    - ISO 4335 — Elements of Procedure
    - ISO 6159 — Unbalanced Classes of Procedure
    - ISO 6256 — Balanced Classes of Procedure
  - The current standard for HDLC is ISO 13239, which replaces all of those standards.
  - HDLC provides both connection-oriented and connectionless service.
  - HDLC can be used for point to multipoint connections, but is now used almost exclusively to connect one device to another, using what is known as Asynchronous Balanced Mode (ABM). The other modes are Normal Response Mode (NRM) and Asynchronous Response Mode (ARM).

Frame Format of HDLC

Bits	8	8	8	⩾ 0	16	8
	0 1 1 1 1 1 1 0	Address	Control	Data	Checksum	0 1 1 1 1 1 1 0

- The address field is used to identify one of the terminals on lines with multiple terminals.
- Control field is used for sequence numbers, acknowledgements and other purposes
- Checksum field is a cyclic redundancy code
- Data field may contain any information. It may be arbitrarily long.

Control Field of HDLC

Bits	1	3	1	3
(a)	0	Seq	P/F	Next
(b)	1	0	Type	P/F	Next
(c)	1	1	Type	P/F	Modifier

Control field of:

(a) An information frame.

(b) A supervisory frame.

(c) An unnumbered frame.

- There are 3 kinds of frames: Information, Supervisory and Unnumbered
- The contents of the control field for these three kinds are shown above
  - The *Seq* field is the frame sequence number.
  - The *Next* field is a piggybacked acknowledgement. All the protocols adhere to the convention that instead of piggybacking the number of the last frame received correctly, they use the number of the first frame not yet received.
  - The *P/F* bit stands for *Poll/Final.* It is used when a computer is inviting the terminal to send data.
- All the frames sent by the terminal, except the final one, have the *P/F* bit set to *P*. The final one is set to *F*.

Type field of HDLC

- The Supervisory frames are distinguished by the Type field.
- Type 0 is an acknowledgement frame (officially RECEIVE READY) used to indicate the next frame expected.
- Type 1 is a negative acknowledgement (REJECT), used to indicate that a transmission error has been detected. The Next field indicates the first frame in sequence not received correctly. The sender is required to retransmit all outstanding frames starting at Next.
- Type 2 is RECEIVE NOT READY. It acknowledges all frames up to but not including Next, just as RECEIVE READY does, but it tells the sender to stop sending. Type 2 is intended to signal certain temporary problems with the receiver, such as a shortage of buffers, and not as an alternative to the sliding window flow control.
- Type 3 is the SELECTIVE REJECT. It calls for retransmission of only the frame specified.