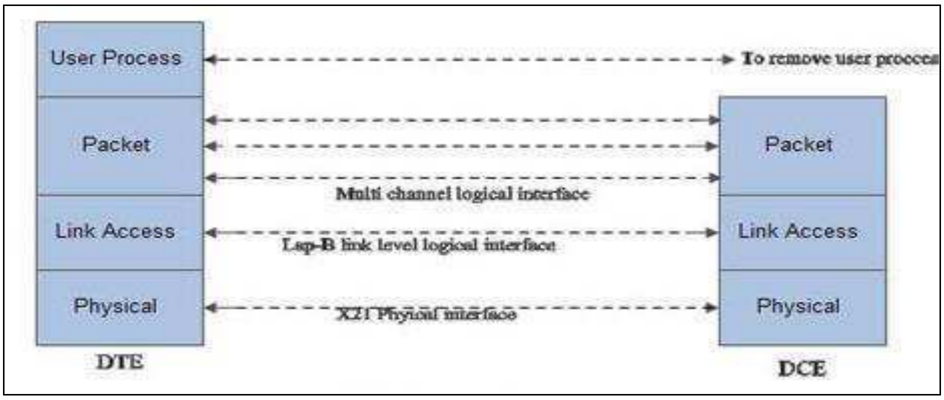
Subject Name: **Computer Network**

Subject Code: **IT-5003**

Semester: **5th**



UNIT II

DATA LINK LAYER:

Introduction

Data Link Layer is the second layer of OSI Layered Model. This layer is one of the most complicated layers and has complex functionalities and liabilities. Data link layer hides the details of underlying hardware and represents itself to the upper layer as the medium to communicate.

Data link layer works between two hosts which are directly connected in some sense. This direct connection could be a point to point or broadcast. Systems on the broadcast network are said to be on the same link. The work of data link layer tends to get more complex when it is dealing with multiple hosts on single collision domain.

Data link layer is responsible for converting data stream to signals bit by bit and to send that over the underlying hardware. At the receiving end, Data link layer picks up data from hardware which are in the form of electrical signals, assembles them in a recognizable frame format, and hands over to the upper layer.

Data link layer has two sub-layers:

- Logical Link Control: It deals with protocols, flow control, and error control
- Media Access Control: It deals with actual control of media

DATA LINK LAYER: SERVICES

- Encapsulation of network layer data packets into frames
- Frame synchronization
- Logical link control (LLC) sublayer:
- Error control (automatic repeat request, ARQ), in addition to ARQ
- Some transport-layer protocols, to forward error correction (FEC) techniques provided on the physical
- layer, and to error-detection and packet canceling provided at all layers, including the network layer.
- Flow control, in addition to the one provided on the transport layer. Data-link-layer error control is not used in LAN protocols such as Ethernet, but in modems and wireless networks.
- Media access control (MAC) sublayer:
- Multiple access protocols for channel access control, for example, CSMA/CD protocols for collision detection and re-transmission in Ethernet bus networks and hub networks, or the CSMA/CA protocol for collision avoidance in wireless networks.
- Physical addressing (MAC addressing)
- LAN switching (packet switching) including MAC filtering and spanning tree protocol
- Data packet queuing or scheduling

- Store-and-forward switching or cut-through switching
- Quality of Service (QoS) control
- Virtual LANs (VLAN)

DATA LINK LAYER: FRAMING

Since the physical layer merely accepts and transmits a stream of bits without any regard to meaning or structure, it is up to the data link layer to create and recognize frame boundaries. This can be accomplished by attaching special bit patterns to the beginning and end of the frame. If these bit patterns can accidentally occur in data, special care must be taken to make sure these patterns are not incorrectly interpreted as frame delimiters.

The four framing methods that are widely used are

- Character count
- Starting and ending characters, with character stuffing
- Starting and ending flags, with bit stuffing
- Physical layer coding violations

Character Count

This method uses a field in the header to specify the number of characters in the frame. When the data link layer at the destination sees the character count, it knows how many characters follow, and hence where the end of the frame is. The disadvantage is that if the count is garbled by a transmission error, the destination will lose synchronization and will be unable to locate the start of the next frame. So, this method is rarely used.

Character stuffing

In the second method, each frame starts with the ASCII character sequence DLE STX and ends with the sequence DLE ETX. (*where DLE is Data Link Escape, STX is Start of TeXt and ETX is End of TeXt*.) This method overcomes the drawbacks of the character count method. If the destination ever loses synchronization, it only has to look for DLE STX and DLE ETX characters. If, however, binary data is being transmitted then there exists a possibility of the characters DLE STX and DLE ETX occurring in the data. Since this can interfere with the framing, a technique called character stuffing is used. The sender's data link layer inserts an ASCII DLE character just before the DLE character in the data. The receiver's data link layer removes this DLE before this data is given to the network layer. However, character stuffing is closely associated with 8-bit characters and this is a major hurdle in transmitting arbitrary sized characters.

Bit stuffing

The third method allows data frames to contain an arbitrary number of bits and allows character codes with an arbitrary number of bits per character. At the start and end of each frame is a flag byte consisting of the special bit pattern 01111110. Whenever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a zero bit into the outgoing bit stream. This technique is called bit stuffing. When the receiver sees five consecutive 1s in the incoming data stream, followed by a zero bit, it automatically stuffs the 0 bit. The boundary between two frames can be determined by locating the flag pattern.

Physical layer coding violations

The final framing method is physical layer coding violations and is applicable to networks in which the encoding on the physical medium contains some redundancy. In such cases normally, a 1 bit is a high-low pair and a 0 bit is a low-high pair. The combinations of low- low and high-high which are not used for data may be used for marking frame boundaries.

DATALINK LAYER: FLOW CONTROL & ERROR CONTROL

Flow Control

Consider a conversation with your friend. One of you listens while the other speaks. You might nod your head as you listen or you might interject with a "Whoa, slow down, you're talking too fast!" This actually flows control. Some of us are better at it than others, but we all do it to some degree.

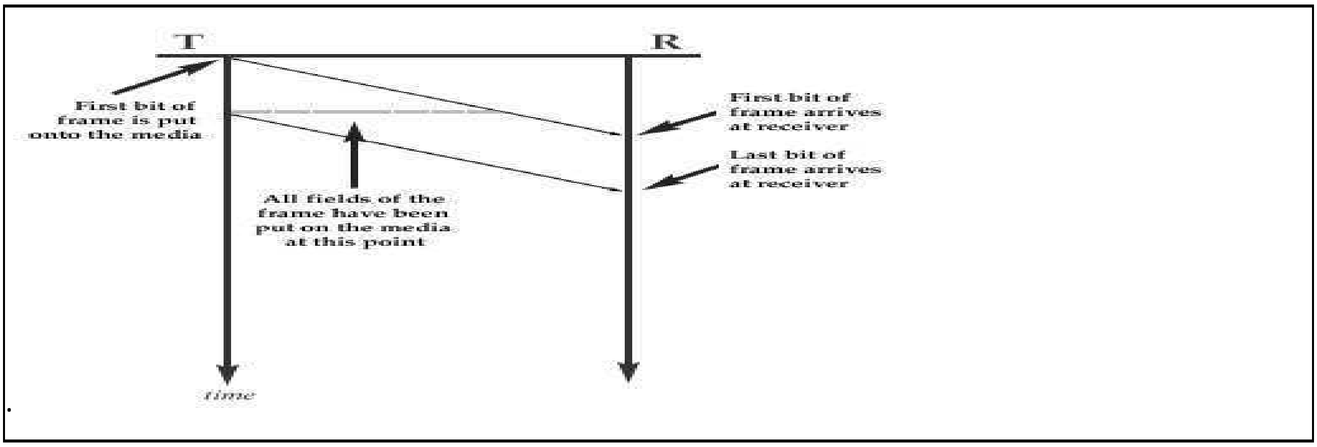
You nod to indicate you understood and are ready for the next morsel of information or you tell your friend when they are going too fast. That's flow control.

Stop and Wait for Flow Control

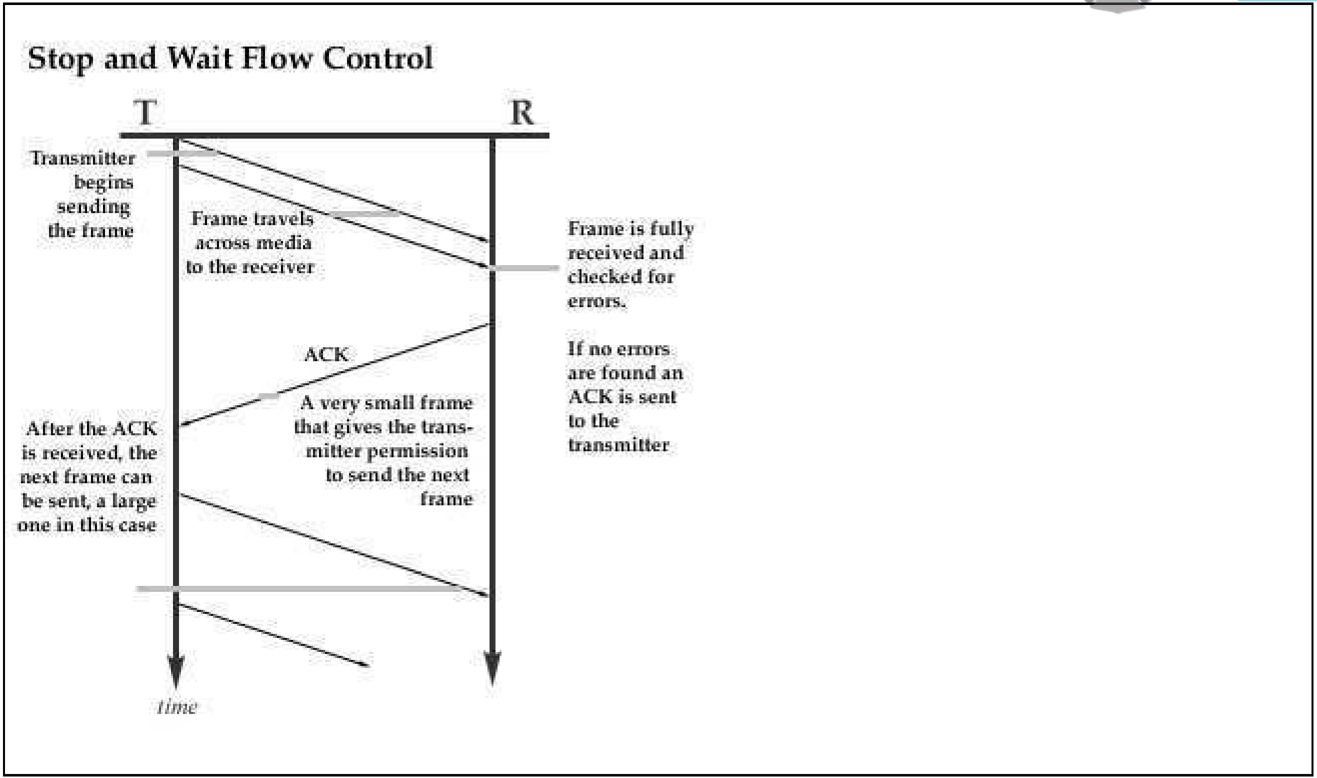
Flow control must also be used for data communications. The simplest form of flow control is called "Stop and Wait Flow Control." This is very simple. the transmitter sends one frame. Then the transmitter stops and waits for the receiver to confirm that the frame arrived without error. The receiver will confirm that the frame arrived by sending a message back to acknowledge the frame. This message is referred to as an ACK (acknowledgment) or RR (receiver ready).

Once the transmitter gets the acknowledgment it can send the next frame. In other words, frames are sent one at a time and each must be confirmed before the next one can be sent.

Take a look at the following diagram. The transmitter is on the left (T) and the receiver (R) is on the right. They are separated by some distance. The vertical scale represents time, with time zero at the top. A frame is transmitted and it takes some time to travel to the receiver. The black arrows crossing the diagram show the leading edge and trailing edge of the frame as it travels to the receiver. Take a look at Stop and Wait for Flow Control in the next diagram. Notice that much of the time, the receiver and the transmitter are doing nothing but waiting.


T
R
First bit of frame is put onto the media
First bit of frame arrives at receiver
Last bit of frame arrives at receiver
All fields of the frame have been put on the media at this point
time

.


Stop and Wait Flow Control
T
R
Transmitter begins sending the frame
Frame travels across media to the receiver
Frame is fully received and checked for errors.
If no errors are found an ACK is sent to the transmitter
ACK
A very small frame that gives the transmitter permission to send the next frame
After the ACK is received, the next frame can be sent, a large one in this case
time

Frames are usually numbered to keep track of them. Acknowledgments are also numbered. Stop and Wait for Flow Control need only number the frames as frame 0 and frame 1. The acknowledgments are numbered ACK0 and ACK1.

Look how much time the transmitter is doing nothing in the above diagram. When a frame is small, much of the time is spent waiting for it to travel across the network and for the ACK to return.

One solution to this problem of poor efficiency is to send larger frames. When frames are larger, more time is spent sending data. Larger frames are more efficient as long as there are few errors. If there is an error the entire frame has to be resent. The larger the frame the more likely there will be an error in it, so there is a point where no benefits are found by increasing the frame size. When there are frequent errors very small frames might be appropriate.

Another solution to the poor efficiency is another form of flow control called Sliding Window Flow Control.

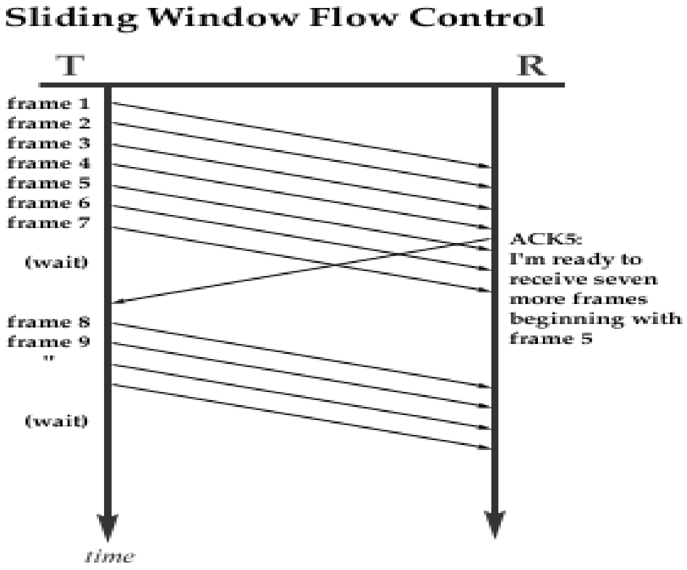
Sliding Window Flow Control

With a relatively error-free link, sliding window flow control is more efficient than stop and wait. The improvement is that the transmitter is granted permission to send more than one frame in a row before it has to stop and wait for an acknowledgment.

Let's say that the transmitter and the receiver agree that seven frames can be sent at a time. The transmitter starts out with permission to send seven frames. We'll define the number of frames that can be sent as the "window". Frames 1 through 7 may be sent immediately.

If all seven frames are sent, the transmitter has to stop and wait. The receiver may send an ACK when it is ready to receive another 7 frames. The ACK will set the transmitter's window back to seven sendable frames. In the case of the receiver has sent an ACK5. This message means that frames 1 through 4 were received correctly and that the receiver is "expecting frame 5 next."

When the transmitter gets ACK5 is knows that beginning with frame 5 is may send seven frames. Some of those frames may have already been sent, though, so the transmitter sends the remaining frames in the window.



The transmitter has to keep track of which frames in the window have been sent and which can be sent. It has to know which frames have been acknowledged as well. After sending 7 frames in the above diagram, the transmitter has to stop and wait. Once ACK5 arrives it marks frames 1 to 4 as acknowledged and "slides" its seven-frame window down to include frames 5 to 11. Having already sent frames 5, 6, and 7, the transmitter now sends 8 through 11. Then the transmitter waits again.

The receiver may choose not to acknowledge frames so that the transmitter will exhaust its supply of sendable frames and stop transmitting. This is how the receiver controls the pace of incoming frames. When the receiver is ready to receive another 7 frames, it can send an ACK.

If there are no problems with the incoming pace, the receiver sends frequent ACKs so that the transmitter never exhausts its window. This is where the efficiency is accomplished. The transmitter can send continuously when the receiver processes frame more quickly than they arrive because the receivers buffer space never fills up.

Computers work in binary. So, to keep count of frames they will use several bits in the control field of the frame for the send number. They will also have a few bits for the received number. Since both sides will have information to send, full duplex transmission, both sides will have a window, send numbers and receiver numbers.

If three bits are reserved for the send number, then ...

$L = 2^n$

$L = 2^3$

$L = 8$

... the window size is 8. For most systems, when the last frame is sent, the number rolls over and starts from zero again. Remember that counting in binary starts will zero, not one. Not all sequence numbers roll over to zero upon reaching the window size number. The window size can be smaller, even much smaller than the frame sequence numbers.

This rollover could cause a problem. What if the transmitter receives an ACK3, then sends frames 3, 4, 5, 6, 7, 0, 1, and 2, and then receives an ACK3? Is this a repeat ACK3 or a new ACK 3? For this reason, the transmitter will always set its window to one less than the maximum. That way the problem cannot occur. This means that:

- 3-bit sequence fields permit a window size of 7, not 8
- 8-bit sequence fields permit a window size of 255, not 256

In practice, the most common window sizes at layer 2, the data link layer, are 7 and 127. However, on low-error links of great distance, it is beneficial to increase the window size.

In summary:

- The window slides when acknowledgments are received
- Only frames inside the window may be sent.
- Sent frames remain in the window until an acknowledgment moves the window.
- window size = 2^n -1

Error Control

So, the receiver can control the pace of incoming frames by carefully timing acknowledgments. But, what happens if a frame is received in error? Of course, some set of procedures must be defined so that errors can be dealt with.

Three types of error control are discussed in your text. All are collectively called automatic repeat request or ARQ. The three types of error control are:

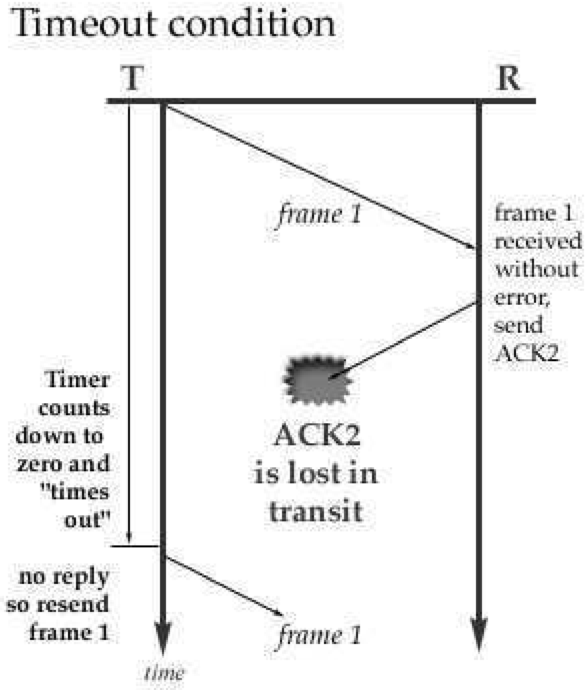
- Stop and Wait ARQ
- Go Back N ARQ
- Selective Reject ARQ

Please don't confuse Stop and Wait Flow Control with Stop and Wait ARQ. These are different concepts that are part of a stop and wait transmission protocol.

Stop and Wait ARQ

In a stop and wait system, if a frame arrives and an error check indicates that somewhere in the frame there is an error, the frame is discarded and a message is sent to notify the transmitter. The message sent is a negative acknowledgment, also called a reject message (NAK or REJ).

What happens if a frame, an ACK or an NAK are lost in transit? This could be a problem, so when a frame is sent, the transmitter starts a timer. If the timer counts down to zero before an ACK or NAK is received the transmitter resends the frame.


Timeout condition
T
R
frame 1
frame 1 received without error, send ACK2
ACK2 is lost in transit
Timer counts down to zero and "times out"
no reply so resend frame 1
frame 1
time

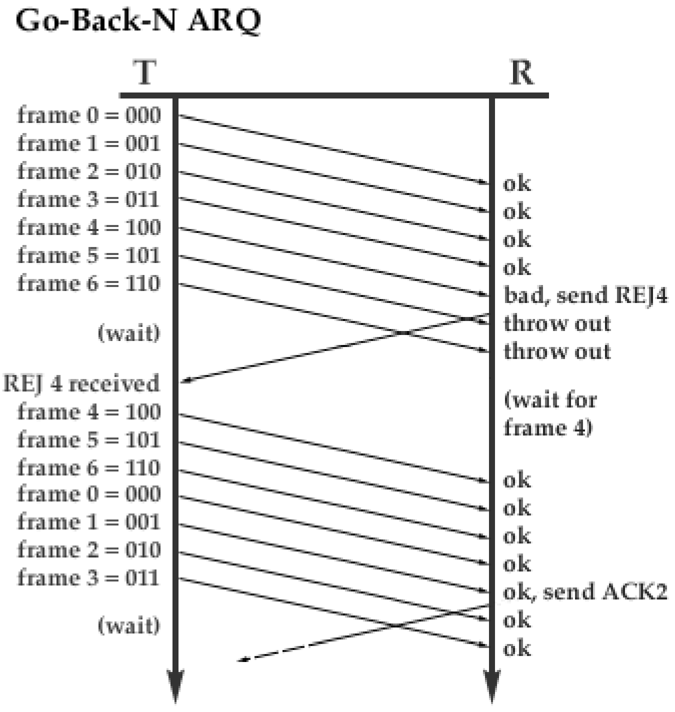
This condition is known as a "time out". Note the following:

- If a frame is lost, after the transmitter times out, the frame is present and the receiver accepts it without no knowledge of the lost frame.
- If an NAK is lost, the transmitter times out and resends the frames. the receiver accepts the retransmitted frame as if the NAK had been received normally. Ah, there is no requirement to send an NAK. The transmitter will resend the frame anyway. Sending the NAK forces the transmitter to resend immediately, rather than waiting for a timeout to occur.
- If an ACK is lost, the transmitter times out and resends the frame. The receiver already has this frame, so it discards the duplicate and resends the ACK.

Go-Back-N ARQ

The Go-Back-N ARQ error control method is used in combination with sliding window flow control. When the receiver finds a frame in error, it tells the transmitter to go back, resend that frame and all succeeding frames. After the erred frame, some succeeding frames will arrive. The receiver discards these knowing that the erred frame will arrive and the succeeding frames will also follow.

This feature guarantees that frames are received in order, but it also means that some good frames may have to be resent.



three bit sequence numbers so window size = 2^3 - 1 = 7 frames

In the above graphic, note how the transmitter begins by resending the frame that was in error and then continues by sending succeeding frames until its window is exhausted.

Selective Reject ARQ

For a system with ample memory space, Selective Reject Request ARQ may provide improved performance when compared to Go-Back-N ARQ. This error control scheme allows the receiver to selectively reject frames.

The receiver, upon finding and erred frame, will request that frame be present and only that frame is resent. The receiver has to keep track of what frames are coming in and what sequence they belong in. Reordering the frames may be required. The receiver cannot deliver frame contents to layer three until the frames have been put in order and no frames are missing in the sequence.

The transmitter also has to keep careful track of which frames have been sent, which have been acknowledged and which are available to be sent. Selective Reject ARQ works in combination with sliding window flow control.

Selective Reject ARQ is not common at the data link layer. It requires much more processing and buffer space to implement. The marginal improvement in capacity is not easy to justify in the hardware used at the data link layer. We will see Selective Reject ARQ later when we study TCP, however.

Summary

Flow Control	***Advantages***	***Disadvantages***
Stop and Wait	Very simple low buffer requirement low processor burden	very inefficient
Sliding Window	very efficient for links with low error rates	not efficient on error-prone links
Error Control	*Advantages*	*disadvantages*
Stop and Wait ARQ	easy to implement low processor burden low buffer requirement	doesn't allow sliding window flow control
Go-Back-N ARQ	easy to implement permits use of sliding window flow control	may have to retransmit good frames
Selective Reject ARQ	permits sliding window flow control, only retransmit lost or erred frames	requires more powerful processing and m

Piggybacking

Piggybacking is a bi-directional data transmission technique in the network layer (OSI model). It makes the most of the sent data frames from receiver to the emitter, adding the confirmation that the data frame sent by the sender was received successfully (ACK acknowledge). This practically means, that instead of sending an acknowledgment in an individual frame it is piggy-backed on the data frame.

Piggybacking data is a bit different from Sliding Window Protocol used in the OSI model. In the data frame itself, we incorporate one additional field for acknowledgment (called ACK).

Whenever party A wants to send data to party B, it will send the data along with this ACK field. Considering the sliding window here of size 8 bits, if A has received frames up to 5 correctly (from B), and wants to send frames starting from frame 6, it will send ACK6 with the data.

Three rules govern the piggybacking data transfer.

- If station A wants to send both data and an acknowledgment, it keeps both fields there.
- If station A wants to send just the acknowledgment, then a separate ACK frame is sent.
- If station A wants to send just the data, then the last acknowledgment field is sent along with the data.

Station B simply ignores this duplicate ACK frame upon receiving.

BIT ORIENTED PROTOCOLS

A bit-oriented protocol is a communications protocol that sees the transmitted data as an opaque stream of bits with no semantics or meaning. Control codes are defined in terms of bit sequences instead of characters. The bit-oriented protocol can transfer data frames regardless of frame contents. It can also be stated as "bit stuffing" this technique allows the data frames to contain an arbitrary number of bits and allows character codes with an arbitrary number of bits per character.

Synchronous framing High-Level Data Link Control is a popular bit-oriented protocol.

HDLC

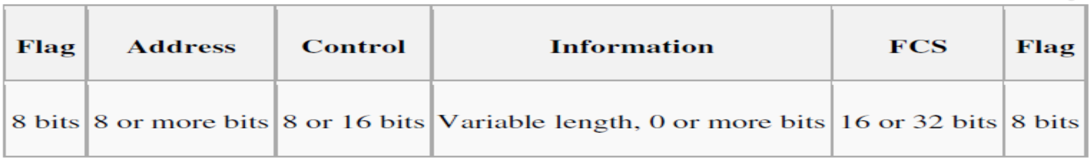
High-Level Data Link Control (HDLC) is a bit-oriented code-transparent synchronous data link layer protocol developed by the International Organization for Standardization (ISO).

HDLC (High-level Data Link Control) is a group of protocols or rules for transmitting data between network points (sometimes called nodes). In HDLC, data is organized into a unit (called a frame) and sent across a network to a destination that verifies its successful arrival. The HDLC protocol also manages the flow or pacing at which data is sent. HDLC is one of the most commonly-used protocols in what is layer 2 of the industry communication reference model called Open Systems Interconnection (OSI). (Layer 1 is the detailed physical level that involves actually generating and receiving the electronic signals. Layer 3 is the higher level that has knowledge about the network, including access to router tables that indicate where to forward or send data. On sending, programming in layer 3 creates a frame that usually contains source and destination network addresses. HDLC (layer 2) encapsulates the layer 3 frame, adding data link control information to a new, larger frame.

HDLC SUBSET	USES
NRM (Normal Response Mode)	Multipoint networks that typically use SDLC
LAP (Link Access Procedure)	Early X.25 implementations
LAPB (Link Access Procedure, Balanced)	Current X.25 implementations
LAPD (Link Access Procedure for the Integrated Services Digital Network D channel)	ISDN D channel and frame relay
LAPM (Link Access Procedure for Modems)	Error-correcting modems (specified a

The original ISO standards for HDLC are:

- ISO 3309 – Frame Structure
- ISO 4335 – Elements of Procedure
- ISO 6159 – Unbalanced Classes of Procedure
- ISO 6256 – Balanced Classes of Procedure



Flag	Address	Control	Information	FCS	Flag
8 bits	8 or more bits	8 or 16 bits	Variable length, 0 or more bits	16 or 32 bits	8 bits

The current standard for HDLC is ISO 13239, which replaces all 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 original master-slave modes Normal Response Mode (NRM) and Asynchronous Response Mode (ARM) are rarely used.

FRAMING

HDLC frames can be transmitted over synchronous or asynchronous serial communication links. Those links have no mechanism to mark the beginning or end of a frame, so the beginning and end of each frame should be identified. This is done by using a frame delimiter, or flag, which is a unique sequence of bits that is guaranteed not to be seen inside a frame. This sequence is '01111110', or, in hexadecimal notation, 0x7E. Each frame begins and ends with a frame delimiter. A frame delimiter at the end of a frame may also mark the start of the next frame. A sequence of 7 or more consecutive 1-bits within a frame will cause the frame to be aborted.

When no frames are being transmitted on a simplex or full-duplex synchronous link, a frame delimiter is continuously transmitted on the link. Using the standard NRZI encoding from bits to line levels (0 bit = transition, 1 bit = no transition), this generates one of two continuous waveforms, depending on the initial state:

This is used by modems to train and synchronize their clocks via phase-locked loops. Some protocols allow the 0-bit at the end of a frame delimiter to be shared with the start of the next frame delimiter, i.e. '011111101111110'.

Link control

The link control protocol is like STR. The designers attempted to protect against simple transmission errors. The protocol requires that every message is acknowledged (ACK0/ACK1) or negatively acknowledged (NAK), so transmission of small packets has high transmission overhead. The protocol can recover from a corrupted data frame, a lost data frame, and a lost acknowledgment.

Error recovery is by retransmission of the corrupted frame. Since Bisync data packets are not serial-numbered, it's considered possible for a data frame to go missing without the receiver realizing it. Therefore, alternating ACK0s and ACK1s are deployed; if the transmitter receives the wrong ACK, it can assume a data packet (or an ACK) went missing. A potential flaw is that corruption of ACK0 into ACK1 could result in duplication of a data frame.

Error protection for ACK0 and ACK1 is weak. The Hamming distance between the two messages is only two bits.

The protocol is half-duplex (2-wire). In this environment, packets or frames of transmission are strictly unidirectional, necessitating 'turn-around' for even the simplest purposes, such as acknowledgments. Turn-around involves

- The reversal of transmission direction,
- resyncing

In a 2-wire environment, this causes a noticeable round-trip delay and reduces performance. Some datasets support full-duplex operation, and full-duplex (4-wire) can be used in many circumstances to improve performance by eliminating the turn-around time, at the added expense of 4-wire installation and support. In typical full-duplex, data packets are transmitted along one wire pair while the acknowledgments are returned along the other.

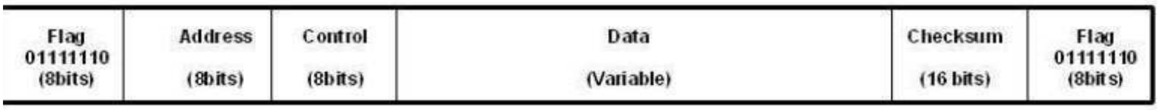
LAP AND LAPB

Link Access Procedure (LAP Link layer protocols for framing and transmitting data across point-to-point links. Was originally derived from HDLC (High-Level Data Link Control), but was later updated and renamed LAPB (LAP Balanced).

LAPB is the data link protocol for X.25.LAPB is a bit-oriented protocol derived from HDLC that ensures that frames are error free and in the right sequence. LAPB is specified in ITU-T Recommendation X.25 and ISO/IEC 7776. It can be used as a Data Link Layer protocol implementing the connection-mode data link service in the OSI Reference Model as defined by ITU-T Recommendation X.222.

LAPB is used to manage communication and packet framing between data terminal equipment (DTE) and the data circuit-terminating equipment (DCE) devices in the X.25 protocol stack. LAPB is essentially HDLC in Asynchronous Balanced Mode (ABM). LAPB sessions can be established by either the DTE or DCE. The station initiating the call is determined to be the primary, and the responding station is the secondary.

Frame types



Flag 01111110 (8bits)	Address (8bits)	Control (8bits)	Data (Variable)	Checksum (16 bits)	Flag 01111110 (8bits)

- I-Frames (Information frames): Carries upper-layer information and some control information. I-frame functions include sequencing, flow control, and error detection and recovery. I-frames carry send and receive sequence numbers.
- S-Frames (Supervisory Frames): Carries control information. S-frame functions include requesting and suspending transmissions, reporting on status, and acknowledging the receipt of I-frames. S-frames carry only receive sequence numbers.
- U-Frames (Unnumbered Frames): carries control information. U-frame functions include link setup and disconnection, as well as error reporting. U-frames carry no Sequence numbers

Frame format

Flag – The value of the flag is always 0x7E. To ensure that the bit pattern of the frame delimiter flag does not appear in the data field of the frame (and therefore cause frame misalignment), a technique known as Bit stuffing is used by both the transmitter and the receiver.

Address field – In LAPB, this field has no meaning since the protocol works in a point to point mode and the DTE network address is represented in the layer 3 packets. This byte is therefore put to a different use; it separates the link commands from the responses and can have only two values: 0x01 and 0x03. 01 identifies frames containing commands from DTE to DCE and responses to these commands from DCE to DTE. 03 is used for frames containing commands from DCE to DTE and for responses from DTE to DCE. Therefore, one side must be configured as a Layer 2 DTE and the other as a Layer 2 DCE (you must not confuse this with the more familiar Layer 1 DCE and DTE designations).

Control field – it serves to identify the type of the frame. In addition, it includes sequence numbers, control features, and error tracking according to the frame type.

Modes of operation

LAPB works in the Asynchronous Balanced Mode (ABM). This mode is balanced (i.e., no master/slave relationship) and is signified by the SABM (E)/SM frame. Each station may initialize, supervise, recover from errors, and send frames at any time. The DTE and DCE are treated as equals.

FCS – The Frame Check Sequence enables a high level of physical error control by allowing the integrity of the transmitted frame data to be checked.

Window size – LAPB supports an extended window size (modulo 128 and modulo 32768) where the maximum number of outstanding frames for acknowledgment is raised from 7 (modulo 8) to 127 (modulo 128) and 32767 (modulo 32768).

Protocol operation

LAPB has no master/slave node relationships. The sender uses the Poll bit in command frames to insist on an immediate response. In the response frame, this same bit becomes the receivers Final bit. The receiver always turns on the Final bit in its response to a command from the sender with the Poll bit set. The P/F bit is generally used when either end becomes unsure about proper frame sequencing because of a possible missing acknowledgment, and it is necessary to re-establish a point of reference. It is also used to trigger an acknowledgment of outstanding I-frames.

SLIP protocol

The need for a data link layer protocol to let IP operate over serial links was identified very early on in the development of TCP/IP. Engineers working on the Internet Protocol needed a way to send IP datagrams over serial connections linking computers together. To solve the problem, they created a very simple protocol that would frame IP datagrams for transmission across the serial line.

An IP datagram is passed down to SLIP, which breaks it into bytes and sends them one at a time over the link. After the last byte of the datagram, a special byte value is sent that tells the receiving device that the datagram has ended. This is called the SLIP END character, and has a byte value of 192 decimal (C0 hexadecimal, 11000000 binary). And that's basically it: take the whole datagram, send it one byte at a time, and then send the byte 192 to delimit the end of the datagram.

Working

- Break an IP datagram into bytes.
- Send the END character (value “192”) after the last byte of the datagram; in better implementations, send the END character before the first byte as well.
- If any byte to be sent in the datagram is “192”, replace it with “219 220”.
- If any byte to be sent is “219”, replace it with “219 221”.

Problems and Limitations of SLIP

- Standardized Datagram Size Specification:
- Error Detection/Correction Mechanism:
- Control Messaging:
- Type Identification:
- Address Discovery Method:
- Support for Compression:
- Security Features:

Point-to-Point Protocol (PPP)

Even as SLIP was being documented as a “nonstandard” in RFC 1055, work was underway for a newer protocol to provide full-featured IP transmission over direct links between pairs of devices. The result is the Point-to-Point Protocol (PPP), which defines a complete method for robust data link connectivity between units using serial lines or other physical layers. It includes numerous capabilities and features, including error detection, compression, authentication, encryption and much more.

SLIP was basically a “hack” to fill a specific need: bridging the gap between IP at layer three and a serial link at layer one. It “gets the job done” but doesn't provide any of the features we really want in a robust protocol for direct links between devices. PPP was developed to be a complete protocol suite that would enable fully-functional layer two connectivity to support not just IP but the transmission of other network layer protocols as well.

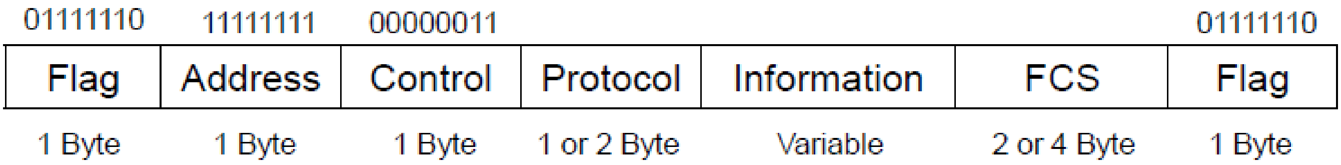
Function

PPP is a connection-oriented protocol that enables layer two links over a variety of different physical layer connections. It is supported on both synchronous and asynchronous lines, and can operate in half-duplex or full-duplex mode. It was designed to carry IP traffic but is general enough to allow any type of network layer datagram to be sent over a PPP connection.

list of PPP's strengths reads very much like a list of SLIP's weaknesses, which I explained in detail in the topic on SLIP. Some of the specific benefits of PPP compared to SLIP include:

A more comprehensive framing mechanism, compared to the single END character in SLIP.

- Specification of the encapsulated protocol, to allow multiple layer three protocols to be multiplexed on a single link.
- Error detection for each transmitted frame through the use of a CRC code in each frame header.
- A robust mechanism for negotiating link parameters, including the maximum frame size permitted.
- A method for testing links before datagram transmission takes place, and monitoring link quality.
- Support for authentication of the connection using multiple authentication protocols.
- Support for additional optional features, including compression, encryption and link aggregation (allowing two devices to use multiple physical links as if they were a single, higher-performance link).



01111110	11111111	00000011				01111110
Flag	Address	Control	Protocol	Information	FCS	Flag
1 Byte	1 Byte	1 Byte	1 or 2 Byte	Variable	2 or 4 Byte	1 Byte

Information Field: Its length is variable. It carries user data or other information.

FCS Field: It stands for Frame Check Sequence. It contains checksum. It is either 2 bytes or 4 bytes.

S. No.	SLIP	PPP
1.	SLIP stands for Serial Line Internet Protocol.	PPP stands for Point-to-Point Protocol
2.	SLIP does not perform error detection & correction.	PPP performs error detection & correction.
3.	SLIP supports only IP.	PPP supports multiple protocols.
4.	IP address is assigned statically.	IP address is assigned dynamically
5.	SLIP does not provide any authentication.	PPP provides authentication.
6.	SLIP is not approved Internet standard.	PPP is approved Internet standard.