UNIVERSITY of WASHINGTON

i341 Network and Distributed Applications

DATA LINK PROTOCOLS

Rev: 2016

Data Link Protocols**:**

Data Link protocols are the rules which govern the way in which computing devices communicate at the data link level. Several data link protocols commonly used include asynch (asynchronous) protocol, bisynch (binary synchronous) protocol, SDLC (Synchronous Data Link Control) and HDLC (High Level Data Link Control) protocols.

Important Data Link Concepts **:**

Data Link (Channel) transmission modes:

Simplex	One way transmission only
Half-duplex	Two way transmission, one way at a time
Full-duplex	Two way simultaneous transmission

Virtual Channel: the logical path between network users, even though it may travel though it may travel through diverse physical routes to get there.

Asynchronous versus Synchronous transmission

Asynchronous transmission can use relatively simple and inexpensive equipment since the two stations communicating are in synchronization only for the time they are transmitting and receiving a single character. Asynchronous transmission is typical of slow speed devices, such as terminals/keyboards that do not have a buffer.

Synchronous transmission sends data in a continual stream. Devices transmit to each other with regular timing, making more efficient data link utilization. All bits in a frame are transmitted at equal time intervals. To permit synchronization, the transmitting and receiving devices must have buffering capability, making them more expensive than asynchronous devices. The actual synchronization of the transmitting and receiving devices in many systems is controlled by oscillators. Before a frame is sent, the oscillator of the receiving machine must be brought into exact phase with the oscillator of the transmitting machine

Error Checking (error detection):

To detect communication errors, the transmission must have some method of checking the validity of the message. This is done by redundancy checking, where some level of redundancy can be built into the individual characters or an entire transmission block. The two predominate error checking techniques are vertical redundancy check (VRC) and longitudinal redundancy check (LRC)

Vertical Redundancy Check (VRC):

Many transmission systems use vertical redundancy checking (VRC) in which each transmitted character is accompanied by a parity bit. If odd parity is used, the transmitter sets the parity bit to either 0 or 1 in order to make the total number of one (1) bits an odd number. If even parity is used, the transmitter sets the parity bit so that the total number of one bits is even. If the receiver would then detect a parity error in a received character, it knows a transmission error has occurred,

A potential problem with parity checking is that if more than one bit is changed, it is possible the parity check would still be correct even though a transmission error had occurred. Parity checking is not very effective at high speed transmission rates.

Figure 1 illustrates vertical redundancy checking.

Longitudinal Redundancy Check (LRC):

With longitudinal redundancy checking (LRC), redundant bits are used to check the accuracy of an entire transmitted frame or block.. The transmitter will run the entire block through an algorithm to generate a number also sent with the message. On the receiving end, when the receiver passes the same block transmitted through the same algorithm and compares it to the generated value sent with the message, if the two values match, it assumes the transmission is correct. If the values differ, it assumes an error has occurred and the block is retransmitted.

Some transmissions schemes will utilize both LRC and VRC methods, but it would be more efficient to just do LRC for frame/block checking only.

Figure 2 illustrates longitudinal redundancy checking.

Functions of Data Link Protocols:

Every specific data link protocol performs a set of functions in different ways. There are according to Martin (1988), basic functions that any data link protocol must perform to be considered a true data link protocol. These functions include:

Synchronization: The data link protocol must be capable of establishing and maintaining synchronization between the sender and receiver. This means the receiver must be capable of determining where each bit/character begins and ends.
Framing: The protocol must be capable of marking the beginning and end of each transmission frame.
Control: The protocol must perform a minimum set of control functions. For example, on a multipoint link, the sending station must be capable of identifying the receiving station to which it is transmitting data.
Error Detection: The data link protocol must be able to perform some degree of error detection and implement error recovery.

Additional Functions of Some Data Link Protocols:

Addressing: the manipulation of network addresses
Frame retransmission: specific rules for retransmitting frames
Pacing: controlling the rate of transmission, when the sender is capable of transmitting faster than the receiver can accept frames, or the receiver is in a congested state.

* These types of functions are indicative of the more advanced and robust data link protocols and should be considered when evaluating network architectures.

Classes of Data Link Protocols:

Character-oriented Protocols: A character oriented protocol uses a particular code set for transmission with a special code set reserved for control functions.

Bit-oriented Protocols: Most modern data link protocols (SDLC, HDLC) are bit oriented protocols, meaning they are independent of any particular code set for transmission, without any character codes reserved for control functions.

Error Checking (error detection):

To detect communication errors, the transmission must have some method of checking the validity of the message. This is done by redundancy checking, where some level of redundancy can be built into the individual characters or an entire transmission block. The two predominate error checking techniques are vertical redundancy check (VRC) and longitudinal redundancy check (LRC)

Vertical Redundancy Check (VRC):

Many transmission systems use vertical redundancy checking (VRC) in which each transmitted character is accompanied by a parity bit. If odd parity is used, the transmitter sets the parity bit to either 0 or 1 in order to make the total number of one (1) bits an odd number. If even parity is used, the transmitter sets the parity bit so that the total number of one bits is even. If the receiver would then detect a parity error in a received character, it knows a transmission error has occurred,

A potential problem with parity checking is that if more than one bit is changed, it is possible the parity check would still be correct even though a transmission error had occurred. Parity checking is not very effective at high speed transmission rates.

Figure 1 illustrates vertical redundancy checking.

Odd Parity

0	**1**	**0**	**1**	**1**	**0**	**1**	**1**
8	**7**	**6**	**5**	**4**	**3**	**2**	**1**

Parity Bit = 0, since there is an odd number of mark (1s)

Figure-1 Vertical Redundancy Checking (Parity)

Longitudinal Redundancy Check (LRC):

With longitudinal redundancy checking (LRC), redundant bits are used to check the accuracy of an entire transmitted frame or block.. The transmitter will run the entire block through an algorithm to generate a number also sent with the message. On the receiving end, when the receiver passes the same block transmitted through the same algorithm and compares it to the generated value sent with the message, if the two values match, it assumes the transmission is correct. If the values differ, it assumes an error has occurred and the block is retransmitted.

Some transmissions schemes will utilize both LRC and VRC methods, but it would be more efficient to just do LRC for frame/block checking only.

Asynchronous Protocol:

Asynchronous transmission techniques are simple and permit inexpensive terminals/devices to be designed. The protocols that use asynchronous transmission are half duplex in nature. The use of start and stop bits result in a relatively high percentage of overhead (20 % - 2 start stop bits for each character transmitted). Asynchronous transmission techniques are oriented to computer devices where communications speeds are relatively low, typically less than 19,200 Bps.

In asynchronous communication, the receiving device must be able to reestablish synchronization for every character by being able to recognize the first bit of each character. This is accomplished, as previously indicated, by the use of the start bit. The receiving station detects start bits by monitoring the link, which can either be in a 1 or 0 state. When there is not any traffic or the link (the line is idle), typically the line will remain in a 1 (bit) condition. This is known as a mark or marking condition. The opposite line status, the 0 bit condition, is known as the space or spacing condition, While in an idle state, a transmitter will continue to send a continuous stream of 1's.

When the transmitting station decides to transmit a character, it changes the link status from mark to space, in effect sending a start bit. The start bit tells the receiving device that data bits follow. The receiving device will then sample the line each time a 0 or 1 bit is sent, if the bit rate of transmission is 2400 **Bps,** this would in effect make each bit time 0.0004166 second

Figure 3 illustrates a typical asynchronous character format.

Bisynch (Binary Synchronous) Protocol

The bisynch protocol differs from asynch in that bit synchronization is established for a long duration, transmissions using bisynch can exceed Mbits of data. Bisynch requires more elaborate and thus expensive circuitry allowing the sending and receiving clocks to remain in synch for long periods of time.

Bit Synchronization: To achieve the required synchronization, two characters called PADs (no acronym available) are sent before each message to ensure that sending and receiving stations are in bit synchronization before transmission starts. According to Martin (1988), the PAD characters consist of sequences of characters containing alternate zero and one bits, such as hexidecimal AA (10101010) or hexidecimal 55 (01010101).

* Beware when discussing the bisynch protocol, a specific IBM product name is generally used to refer to what variation of the protocol is meant. 3270 bisynch differs from 2780 and 3780 bisynch.

Line control characters: There are specific codes used to control bisynch transmission, known as mnemonic codes. These mnemonic codes are used to identify each line control function. In some cases the same mnemonic will be used in both the ASCII and EBCDIC character sets to allow transmission of either character code over the same data link protocol.

Control Code Function	BSC Mnemonic	EBCDIC Mnemonic	ASCII Mnem.
Start of Heading	SOH	SOH	SOH
Start of Text	SOT	SOT	SOT
End of trans. block	ETB	ETB	ETB
End of Text	ETX	ETX	ETX
End of transmission	EOT	EOT	EOT
Negative acknowledgement	NAK	NAK	NAK
Positive acknowledgement	ACK	ACK	ACK
Synchronous Idle	SYN	SYN	SYN
Mandatory disconnect	DISC	DLE EOT	same

Data Link Pacing Techniques:

Some protocols can make use of pacing techniques, to deal with potential congestion of end devices such as front end processors, or the inability of the end device to receive data as fast as the sending station can transmit it.

XON/XOFF: In asynchronous transmission, one such technique is utilized, especially with printer devices who have the capability to receive data at 2400 Bps (for example), but do not have the capability to print data that fast. When the printer buffer fills, the devices needs to notify the sender that it can not accept any more data, so the sender will stop transmitting new data until the printer empties its buffer. This technique uses two control codes called XON/XOFF. The codes XON (**T**ransmitter **O**n) and XOFF (**T**ransmitter **O**ff). Most implementations use the ASCII DC1 code to represent XON and DC3 to represent XOFF.

SDLC (Synchronous Data Link Control) printers have a VPACING parameter in their sysgen statement to provide such a pacing capability. When IBM 37X5 FEPs communicate with each other, they utilize a pacing technique known as virtual route window size to control the amount of data they may be sending to a congested peer 37X5.

Bit Oriented Protocols:

SDLC (Synchronous Data Link Control)

SDLC is the main data link protocol that is used to implement networks conforming to the IBM Systems Network Architecture (SNA). SDLC is a subset of ISO's HDLC and is known for its superior network error recovery, network management visibility, and performance over satellite links (modulo feature).

SNA and SDLC will be covered in more detail in the Systems component lecture on the SNA architecture and components.

LAP and LAPB (Link Access Protocol and Link Access Protocol- Balanced)

The protocols for LAP and LAPB document the data link layer function of the X.25 Recommendation of the CCITT. LAP and LAPB are compatible subsets of HDLC.

HDLC (High Level Data Link Control)

The standard for HDLC was developed by the International Standards Organization and is a well documented, and popular data link protocol. HDLC is used by the majority of bridge and router vendors as their data link protocol to interconnect geographically distributed LANs.

HDLC (High Level Data Link Control)

HDLC is a bit oriented protocol specification published by ISO (international Standards Organization). HDLC is widely utilized throughout the world, it provides many functions and supports a broad range of applications. HDLC is also the basis for many other widely used protocols (such as SDLC, LAPB and LLC) used in the data communications industry. HDLC Figure-1 illustrates the HDLC family.

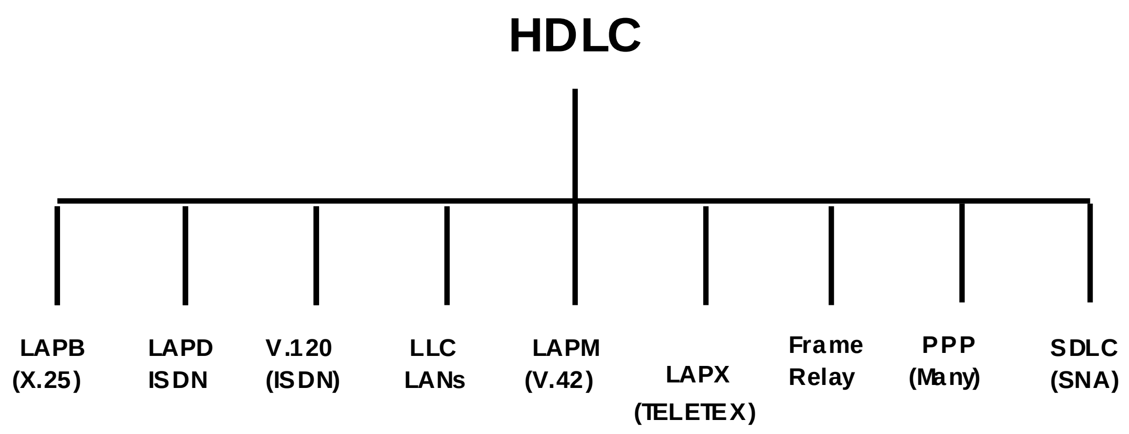

Figure-1 (HDLC Family)

HDLC defines three types of stations, two link configurations and three data modes of operation.

The three station types are:

Primary Station: has the responsibility for controlling the operation of the link. Frames issued by the primary station are called commands.

Secondary Station: operates under control of the primary station. Frames issued by the secondary station(s) are called responses. The primary station maintains a logical link with each secondary station on the line.

Combined Station: combines the features of primary and secondary stations. A combined station may issue commands and responses.

The two link configurations are:

Unbalanced configuration: used in point to point and multipoint operation. This configuration consists of one primary and one or more secondary stations and supports both half and full duplex transmission.

Balanced configuration: used only in point to point operation. This configuration consists of two combined stations and supports both half and full duplex transmission.

The three data transfer modes are:

Normal Response Mode (NRM): This is an unbalanced configuration. The primary may initiate data transfer to a secondary device, but a secondary may only transmit data in response to a poll from the primary.

Asynchronous Balanced Mode (ABM): This is a balanced configuration. Either combined station may initiate transmission without receiving permission from the other side.

Asynchronous Response Mode (ARM): This is an unbalanced configuration. In ARM, the secondary may initiate data transfer without explicit permission of the primary (ie send a response without waiting for a command). The primary still retains responsibility for the line, including initialization, error recovery and logical disconnection.

The Normal Response Mode is used on multidrop lines where a number of terminals are connected to a computer. The computer polls each terminal for input. NRM is also used on point to point links, especially for links connecting terminals to computers. The Asynchronous Balanced Mode makes more efficient use of a full duplex point to point link, since there is no polling overhead. The Asynchronous Response Mode is rarely used; it is applicable to hub polling and special situations when a secondary device may need to initiate transmission.

HDLC Frame Structure:

HDLC uses synchronous transmission, and all transmissions are in frames. One frame format is sufficient for all types of HDLC data and control exchanges. The HDLC frame has the following fields:

Flag	8 Bits
Address	One or More Octets
Control	8 or 16 Bits
Information	Variable

Frame Check Sequence (FCS)	16 or 32 bits
Flag	

Flag Fields: delimit the frame at both ends with the unique pattern 01111110. A single flag may be used as the closing flag for one frame and the opening flag for the next. All active stations attached on the link are continuously hunting for the flag sequence to synchronize on the start of a frame. While receiving a frame, a station continues to hunt for the sequence to determine the end of the frame. Since HDLC frame allows arbitrary bit patterns, there is no assurance that the pattern 01111110 will not appear somewhere inside the actual HDLC frame, and this would destroy frame level synchronization. To avoid this problem, a procedure knows as **bit stuffing** is utilized. In bit stuffing, the transmitter will always insert an extra 0 after each occurrence of five 1s in the frame (with the exception of the flag fields). After detecting a start flag, the receiver monitors the bit stream. When a pattern of five 0s is detected, the sixth bit is examined. If this bit is 0, it is deleted. If the sixth bit is a 1 and the seventh bit a 0, the combination is accepted as a flag. If the sixth and seventh bits are both 1,, the sending station is signaling an abort condition.

Address Field: used to identify the secondary station that transmitted or is to receive the frame. This field is not required for point to point links, but is always included to make all frames uniform.

Control Field: HDLC defines three types of frames, each with a different control field format. Information frames (I-frames) carry data to be transmitted for the station, known as user data. Flow and error control information, using the ARQ mechanism, may be piggybacked on an information frame. Supervisory frames (S-frames) provide the ARQ (Automatic Repeat Request) mechanism when piggybacking is not used. Unnumbered frames (U-frames) provide supplement link control functions. the first one or two bits of the control field serves to identify the frame type.

Information Frames: The basic operation of HDLC involves the exchange of I-frames containing user data. Each I-frame contains the sequence number of the transmitted frame as well as a piggybacked positive acknowledgement. The acknowledgement is the sequence number of the **next** frame expected. A maximum window size of 7 or 127 is supported. The I-frame also contains a poll/final (P/F) bit. The bit is a poll bit for commands from the primary, and a final bit from the secondary for responses. In normal response mode, the primary issues a poll giving permission to send by setting the poll bit to 1, and the secondary sets the final bit to 1 on the last I-frame if its response. In asynchronous response mode (ARM) and asynchronous balanced mode (ABM), the P/F bit is sometimes used to coordinate the exchange of S-frames and U-frames.

Supervisory Frames: The supervisory frame is used for flow and error control. Both Go-back-N ARQ (REJ) and selective reject (SREJ) are allowed. A frame is normally acknowledged with a receive ready (RR) when an I-frame is not available for piggybacking. Additionally, a receive not ready)RNR) is used to accept a frame but request that no more I-frames are sent until a subsequent RR is used. For RR, RNR and REJ frames, N(R) indicates the sequence number of the next expected I-frame. For SREJ, N(R) is the sequence number of the rejected frame.

Unnumbered Frames: are used for a variety of control functions. These frames do not carry sequence numbers nor alter the flow or sequence of numbered I-frames. Unnumbered frames can be grouped into the following categories:

Mode setting commands and responses
Information transfer commands and responses
Recovery commands and responses
Miscellaneous commands and responses

Examples of Mode Commands:

Disconnect (DISC) - transmitting station suspending operation
Request Initialization mode (RIM) - station not ready, req init.
Disconnect Mode (DM) - respond station logically disconnected
SNRM
SNRME
SARM
SABM
SABM

Examples of Recovery Commands:

Frame Reject (FRMR) error in received frame
Reset (RSET) clear FRMR condition

Examples of Miscellaneous Commands

XID (Exchange ID) - two stations to exchange station IDs
TEST - Test Command echoed with Test Response, simple means of testing the link and addressed station are still functioning.

HDLC Timers:

Network Component vendors implement link level timers in their products, specifically the network operating system application software. Examples include: a) in the IBM SNA environment, the IBM 37X5 (FEP) software operating system called NCP (Network Control Program) , and b) OS/2 Communications Manager.

HDLC defines two timers, they are:

1) T1
2) T2

Most implementations use T1 in some fashion. T2 is used, but not as frequently as T1. The timers are used as follows:

T1: A primary station issues a P bit and checks whether a response is received to the P bit within a defined time. Thus function is controlled by the T1 timer and is called the "wait for F" time-out.

T2: A station in ARM mode that issues I-frames checks whether acknowledgements are received within a timer period. This function is controlled by timer T2 and is called "wait for N(R)" time-out.

Figure-2 (Stallings, 1994) illustrates the HDLC frame format.

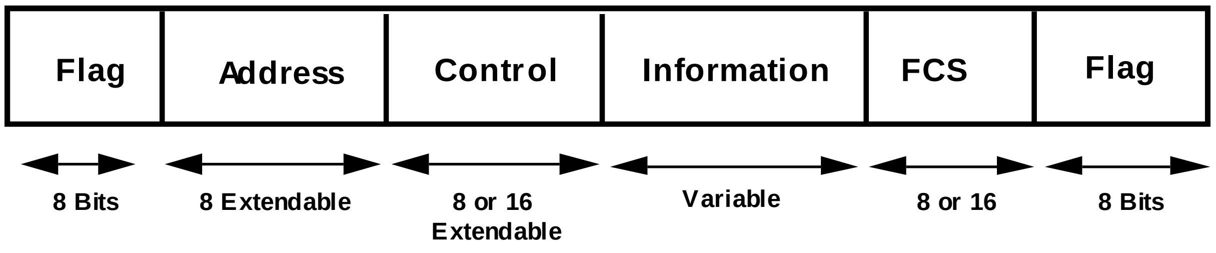

Figure-2 HDLC Frame Format

Figure-3 illustrates the HDLC Control Field format

	1	2	3	4	5	6	7	8
I: Information	0	N(S)			P/F	N(R)		
S: Supervisory	1	0	S		P/F	N(R)		
U: Unnumbered	1	1	M		P/F	M		

N (S) = Send Sequence Number
N (R) = Receive Sequence Number
S = Supervisory Function bits
M = Unnumbered function bits
P/F = Poll/Final Bit

Figure-3 HDLC Control Field Format

Sequencing of Data

Most systems offer some type of sequencing for data delivery. Whether the sequencing is simplex, as in the stop-and-wait method or complex, as used with LLC2 and TCP/IP, there will always be some type of sequencing to ensure proper data delivery.

The following describes generic sequencing - one is that is definitely used follows a method known as Automatic Return Request (ARQ).

Sequencing of transmitted data ensures that when the data is received it will be presented to the receiver of the data in food condition and in the same order in which it was sent. Imagine sending data (a lawyer's legal text of a court trial, for example) between a PC and a host. During the transfer, the data got mixed up and was received at the host in the wrong order. Without sequencing of data, the host's application would receive the data as resented to it by the sending software, and would then process the data (save it to a file, input it into a data base, etc.). If the file was saved, the file would not be saved in the way it was sent. Needless to say, in any application, misordering data can have catastrophic effects.

During the initiating of a connection, part of the handshaking that goes on is the establishment of a data window. For two communicating network stations, A and B, B will establish a window for A and A will establish a window for B. These windows are maintained by state variables. Another name for this is a *counter*. The transmitting station will maintain a send-state variable, V(S). This will contain the sequence number of the next frame to be transmitted. A receiving station will maintain a receive-state variable, V(R). This will contain the sequence number that is expected to be the next frame received. V(S) is incremented with each frame that is transmitted by a network station. This counter is also placed in a sequence field in a frame that is transmitted.

When a network station receives a frame, it will look for sequence number N(S) in the frame received. IF this field matches its V(R), then it will increment its V(R) by one and place this number in a frame N(R) of some type of an acknowledgment packet that will be transmitted back to the originator of the frame.

Sequence Counter Definitions:

V(R) Receive-state variable. A counter maintained by a network station. This counter indicates the sequence number of the next-in-sequence I PDU to be received on a connection. It is maintained in the network station and not the frame.

N(S) Sequence number of the frame (called the send sequence number). Located in the transmitted frame, this field will only be set in information (I) packets. Prior to sending an I frame, the value of N(S) is set to the value of V(S), the send-state variable. This is located in the frame and not in the network station.

V(S) Send-state variable. This number indicates the next sequence number expected on the connection. It is incremented by one for each successive I-frame transmission. It is maintained in the network station and not the frame.

N(R) Receive Sequence number. This is an acknowledgment of a previous frame. It is located in the transmitted frame. All information and supervisory frames will contain this. Prior to sending that type of frame, it is set equal to the value of the receive-state V(R) for that connection. N(R) indicates to the receiving station that the station that originated this frame accepts all frames up to the N(R) minus 1.

If, during the matching of V(R) to the received sequence number, there was a mismatch, and usually after a wait timer expires, the station will send a negative acknowledgment packet to the originator. In this packet will be the sequence number of its value in V(R). In HDLC, this type of frame is a REJ or SREJ frame.

Example: if a station expected one sequence number but received another, the number it expected will be transmitted back to the sender in hopes of receiving the correct packet and sequence number.

When this packet is received by the originating station, it will look at the received sequence number N(R). It will also know that it has already sent this frame, but something went wrong in the process. If the station can, it will retransmit the old frame.

What is the ordering-for-sequencing numbering? Not all protocols operate the same. For SDLC and HDLC, the sequence numbering starts at 0 and may go as high as 127 (known as modulo 128). This means that sequence numbers may go as high as 127, but then must return (wrap around) to 0. This permits 127 frames to be outstanding (not acknowledged). It does not permit 128 frames to be outstanding, for the value of V(R) is the next expected sequence number.

It also guards against wrapping, for if a station has 127 outstanding frames (0-126), a sending station may not use 0 again until it has been acknowledged. This is a highly unlikely case. Most frames will be acknowledged within a few sequence numbers.

One important consideration of window sizes is the actual size of the window. While having a modulo of 128 is nice in that 127 frames maybe transmitted with one acknowledgment, it is also a resource constraint. No transmitted frame may be erased in the sending network station's memory until it has been acknowledged. This means that a network station should have enough memory to store that amount of data until it is acknowledged.

The window can be shut down at any time, which will prevent any more frames from certain stations to be received. This allows for efficient use of resources and also allows a network station to tend to other stations on the network. In other words, it eliminates the possibility of one station hogging another station's time.This window does slide. If a window size of 7 is opened to another station and six frames are outstanding to a network station, the window will be closed. If an acknowledgment is received for six frames, the window will be opened again for six frames. This is called the *sliding window*.

An advantage of a windowing system is known as *inclusive acknowledgments*. This means that if a network station has five outstanding (unacknowledged) frames (frames 0-4), and it receives an acknowledgement frame of 5, this can mean that the destination station has received frames 0-4, and the next frame the receiving station expects to see should have a sequence number of 5.This has many advantages. For one, it keeps the receiving station from transmitting five acknowledgment frames. There will be less overhead on the network and the network stations (only one acknowledgment was transmitted instead of five).

This ability to detect and correct sequence errors is basically characterized by three types of retransmissions: **Go Back to N, Selective Repeat, and Stop and Wait**. With Selective Repeat, only the frame indicated needs to be retransmitted. The Go-Back-to N method specifies not only a specific sequence number is to be retransmitted, but also any frames before that and up to the last acknowledged sequence number. Stop and Wait means send a packet and do not transmit another until that packet has been acknowledged.

All three types have their merits.

The **Selective Rejec**t offers better bandwidth utilization in that only the out-of-sequence frame needs to be retransmitted. But the receiving network station must wait for that frame and, when it does arrive, it must reorder the data in the correct sequence before presenting it to the next layer. This consumes memory and CPU utilization.

Go-Back-to-N is a simpler method; however, it uses more network bandwidth and is generally slower that the Selective Reject.
HDLC uses the Selective Reject methods. Other network protocols use a variance of the both Go Back to N andSelective Reject.

Stop-and-Wait method has a window size of 1, for only one frame may be outstanding at a time. With this, a transmitting station will transmit a frame and will wait for an acknowledgment. It cannot transmit any more frames until it has received an acknowledgment for the previous frame. The sequence numbers used can be of two types. One can be a modulo number of some number and the other can be a 0-1 exchange. With the 0-1 exchange, a starting number is established between the two stations, say a 0. When the transmitting station transmits a packet, it will set the sequence number to a 0. When the receiver receives the packet, it will set its received sequence number to a 0 and then acknowledge the packet. When the response packet is received by the original station, it will notice the 0 sequence number and then set the next transmit sequence number to a 1. This sequence number will flip flop throughout the length of the data transfers. The only two sequence numbers used are 0 and 1.