

3

Data Link Control Procedures

SIMON S. LAM

*The University of Texas at Austin
Austin, Texas*

3.1 INTRODUCTION

Data link control (DLC) protocols are concerned with the communication of data between different machines. Communication is considered to be accomplished if the intended receiver of a data message acquires access to the same serial stream of bits as that of the sender within some acceptable time delay. Thus “communication” is used throughout this chapter to mean *synchronization* of data.

Communication between “processes” within the same machine is easily accomplished with shared memory and an operating system that coordinates the communicating processes. When the communicating processes reside in different machines, however, several problems arise:

1. Communications facilities are needed to transport data from one machine to the other.
2. Coordination of the communicating processes can no longer rely on an operating system but needs to be done by the processes exchanging control messages directly between themselves.
3. The error rates of communications channels are such that comprehensive error detection and recovery techniques are needed to handle (1) and (2) properly.
4. The communications channel, especially over a long distance, is expensive and may need to be shared among concurrent “dialogues.”

We next define some terminology. Machines that communicate include various kinds of terminals and computers. They are known as *data terminal equipment* (DTEs) in the protocol literature. Each DTE has a mechanism, which we shall call a *transceiver*, with the functional capability of sending and receiving bits serially over a communications channel. Examples of transceivers are modems in common-carrier communications facilities, line driver/receivers in in-house cable networks, and so on. Together, a DTE and its transceiver will be referred to as a *station*; see Fig. 3.1 for

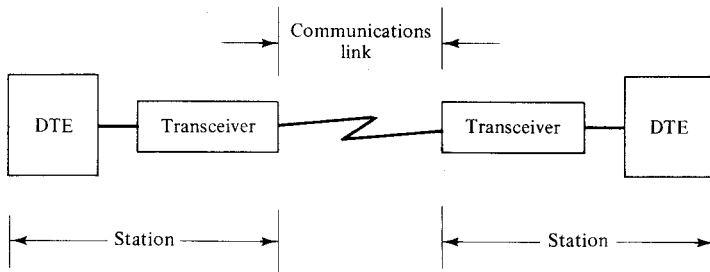


Figure 3.1. Two communicating stations.

illustration. A station is both a sender and a receiver of data messages (although not necessarily at the same time).

Although more than two stations may be involved in a “conference call” or “broadcast” mode of communication, present DLC protocols (of interest herein) deal mainly with dialogues between station pairs. To achieve communication between two stations, the synchronization functions required can be structured into a hierarchy of five levels of functions, to be referred to as the *hierarchical model* (see Fig. 3.2). Each level in the hierarchy depends on synchronism at the level below achieved over *finer grains of time*. Thus each level is offered a virtual (or real) communications channel by the level below, and in turn provides the level above with a virtual communications channel having improved characteristics. Together, the hierarchy of functions supplies the functional capabilities to bridge the gap between what is provided by communications channels and what is required for process-to-process communication.

The synchronization functions shown in Fig. 3.2 are defined in Sec. 3.2 and discussed in detail in Sec. 3.3 for the three major classes of DLC protocols: start-stop, character-oriented, and bit-oriented. Specifically, the hierarchical model is used as a framework for describing IBM’s character-oriented BSC (Binary Synchronous Communication) protocol and bit-oriented SDLC (Synchronous Data Link Control) protocol. In Sec. 3.4 the different classes of DLC protocols are compared. For bit-oriented protocols, additional functional capabilities available in the protocol standards, HDLC (High-Level Data Link Control) and ADCCP (Advanced Data Communication Control Procedure), are discussed.

The objective of this chapter is to present DLC functional requirements using the hierarchical model as a framework and various existing protocols to illustrate solution techniques. For a detailed and complete description of these protocols, the reader should consult the latest manuals for up-to-date information.

Finally, we note that the hierarchical model introduced herein provides a structured representation of functional requirements for communication between two remote processes. Thus in addition to being a useful model for DLC protocols, it can be used to model higher-level communication protocol layers as well.

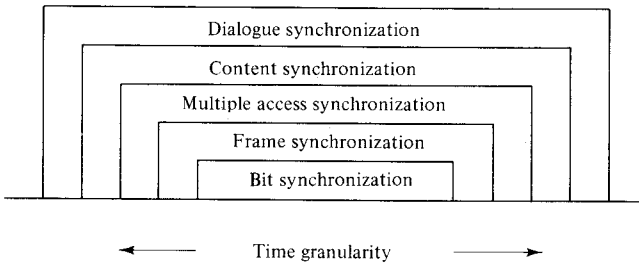


Figure 3.2. Hierarchy of communication functions.

3.2 SYNCHRONIZATION FUNCTIONS

We shall proceed from the bottom of the hierarchy shown in Fig. 3.2 upward.

Bit Synchronization

This level of functions deals with a receiver's ability to retrieve a stream of bits from an incoming analog signal by sampling for the bits at the proper times. The receiver needs to know (1) when to start sampling the first bit, and (2) the period (bit duration) between sampling consecutive bits.

Frame Synchronization

A frame is defined herein to be a stream of bits which is the basic unit of transfer of data and/or control information from a sender to a receiver. The frame synchronization level of functions establishes the conventions for a sender and receiver to delimit within a continuous stream of bits the beginning and the end of a frame. (We note that if the frame size is an integer multiple of the character size of a character-oriented system, character synchronization is also established.)

Another function at this level is error detection. Typically, to detect communication errors, redundancy is built into the bit stream to be transmitted using one of several coding techniques.

Multiple Access Synchronization

It is often necessary to share a physical communications channel among different concurrent sender-receiver pairs. In this case frames transmitted by different senders are interleaved in time. The multiple access synchronization level of functions is concerned with (1) control signals and protocols necessary to synchronize senders so as to avoid (or resolve) access conflicts, and (2) an addressing scheme for identifying senders and receivers. With multiple access synchronization, the higher-level protocol can then behave as if a dedicated (virtual) channel is provided for each sender-receiver pair.

The reader is referred to Chap. 4 for a tutorial treatment of multiple access protocols for channel sharing.

Content Synchronization

This level of functions is concerned with the information content of a frame transmitted from a sender to a receiver, in particular (1) how to differentiate between data and control information within the frame; (2) how to encode and decode control messages; (3) error control to ensure that a single error-free copy of each frame to be sent will arrive at the receiver within some acceptable duration of time; and (4) sequence control to ensure that when a data message is segmented and transported in multiple frames, the message segments are reassembled in the correct sequence.

Dialogue Synchronization

This level of functions is concerned with the initiation and subsequent termination of a dialogue between two stations during which the transfer of frames containing data can take place. Each station is both a sender and a receiver so that there are usually two sender-receiver pairs involved in a dialogue. Dialogue synchronization functions are also needed (1) to coordinate each station pair so that a sender transmits a frame only when it is in a state with the right to transmit and the receiver is in a state that can accept a new frame; and (2) to detect "nonsynchronous" conditions, attempt recovery upon detection of such conditions, and report unrecoverable conditions to higher-level protocols.

Examples of nonsynchronous conditions at the dialogue level are (1) incomplete protocol specification—a data or control message received is not among those expected by the receiver in its present state, and (2) deadlocks—mutual wait conditions between two stations.

In summary, we observe that bit and frame synchronization are necessary for the most rudimentary form of communication between two machines. Multiple access synchronization provides for the sharing of a communications channel by multiple sender-receiver pairs. Content synchronization provides for improved error characteristics, and the encoding of data and control information. Dialogue synchronization provides for data transfer in a mode of operation with desirable characteristics (flow control, interrupts, etc.)

3.3 ASYNCHRONOUS AND SYNCHRONOUS PROTOCOLS

The method for achieving bit synchronism between sender and receiver determines two basic forms of transmission: asynchronous and synchronous. In an asynchronous transmission system, bit synchronism is (in fact) maintained but only when data transmission is taking place and not during idle periods. The amount of data transmitted at a time is one character (5 to 8 bits). In a synchronous transmission system, bit synchronism is maintained at all times. DLC protocols normally used with

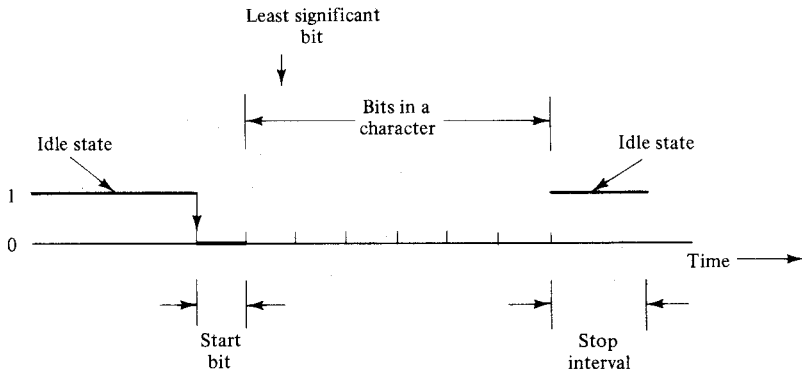


Figure 3.3. Asynchronous transmission.

a synchronous transmission system are said to be synchronous protocols, while DLC protocols normally used with an asynchronous transmission system are said to be asynchronous or start-stop protocols. We note that it is possible to use character-oriented protocols for both synchronous and asynchronous transmission systems. An example is the ANSI (American National Standards Institute) X3.28 protocol standard [ANSI 75], which is akin to the character-oriented BSC protocol to be described. However, both are much more sophisticated than early start-stop DLC protocols and will be classified as synchronous protocols in our discussions below.

3.3.1 Asynchronous Protocols [MCNA 77]

The history of start-stop protocols lies in early teleprinter systems. Consider a communications line connecting a sender to a receiver. By convention, the line is idle when current is flowing in it, which is called the 1 state or MARK condition. The no-current state is called the 0 state or SPACE condition. To start transmitting, the line is brought to the 0 state for one bit time; this is called the *start bit*. For the next 5 to 8 bits (the number depending on the character code being used) the line is brought to the 1 or 0 state as necessary to represent the character being sent. After each character, the line is maintained at the 1 state for a minimum duration of time (equal to 1, 1.5, or 2 bit times) before the transmission of another character can begin; this is the *stop interval* (see Fig. 3.3). Thus transmission is one character at a time, with the bits representing each character preceded by a start bit and succeeded by a stop interval. The stop interval ensures that a transition occurs at the beginning of each start bit, which is used by the receiver to acquire bit synchronization.

Start-stop protocols are said to be asynchronous because a new character can be transmitted at any time (thus asynchronously) following the stop interval of the previous character. Bit synchronism, however, is maintained during the transmission of each character. The receiver determines the initial sampling time from the 1-to-0

transition of the start bit. (This can be accomplished, for example, by a “16 x clock” which samples the incoming signal at 16 times the transmission bit rate.) The sampling period between bits is known from the transmission bit rate. However, since clocks in different machines may differ slightly, the sampling of the last bit in a character may be somewhat off center. This can be corrected in the reception of the next character by retiming the 1-to-0 transition of its start bit.

Start-stop protocols are based upon the use of a character code (Baudot, ASCII, etc.) for representing control messages and data. The start bit and stop interval, in addition to providing bit synchronization, also provides character synchronization (i.e., how to partition a stream of bits into characters). Most character codes provide for some control characters which are used for various DLC control functions (mentioned above) as well as necessary device control functions.

Typically, start-stop protocols are used for terminal devices with a keyboard, with no message buffer, and little intelligence; characters are sent one at a time as keys are struck. The terminal device is controlled remotely by its “host” computer. Start-stop protocols were mostly developed in the past in conjunction with the development of terminal devices. Each terminal type would have its own protocol. Such protocols are often incompatible between terminal types.

Although a significant portion of terminal devices currently in use are nonintelligent start-stop devices, we shall not dwell on start-stop protocols much more. First, for pedagogic reasons; start-stop protocols do not provide a good illustration of the hierarchy of DLC functions discussed in Sec. 3.2. Second, the technological trend is toward more intelligent terminals which can accommodate synchronous protocols having more sophisticated functions.

3.3.2 Synchronous Protocols

The proliferation of start-stop protocols associated with numerous terminal types is a problem. Another disadvantage of asynchronous transmission is the overhead of a start bit and a stop interval associated with each character transmitted. With synchronous transmission the overhead above is eliminated by keeping bit synchronism between receiver and sender at all times, including idle periods. Most synchronous protocols are intended for terminal devices which have more buffering and processing capability than start-stop devices and therefore can implement more functions.

Bit Synchronization

When the receiver and transmitter are in proximity, bit synchronism can be maintained by a single shared clock for both the transmitter and the receiver. In other cases, bit timing will have to be recovered from the data signal at the receiver. To do so necessitates some provisions to ensure transitions in the data signal. Specifically, when a transmitter is first connected to a receiver, some number of initial transitions are necessary to establish bit synchronization. The sampling period between bits is known approximately from the data transmission rate but needs to be checked and adjusted periodically with the help of transitions in the data signal. This implies that certain bit streams may not be acceptable or that additional bits may have to be

inserted to ensure an adequate number of transitions for reliable clock recovery. In most systems, the transceiver is a synchronous modem, which will recover symbol timing and provide bit timing to the DTE in a received data timing lead. (The transmit data timing may be provided by either the DTE or modem. Some synchronous modems do not need data link control to provide transitions in the data signal; for example, some have scrambler–descrambler circuits that create transitions in the data signal.)

The subject of modulation–demodulation and clock recovery are classical communication problems at a lower level of detail and are beyond the scope of this chapter. The reader is referred to [DAVE 72]. We shall only discuss bit synchronism in terms of certain DLC functions provided for preventing the transmission of transitionless data.

Two Classes of Synchronous Protocols

There are two main classes of synchronous protocols: character-oriented and bit-oriented. A main distinction between them, as suggested by the names, is the technique for accomplishing frame synchronization. But as we shall see, they differ in their methods for multiple access synchronization, content synchronization, and dialogue synchronization as well.

Character-oriented protocols include the X3.28 standard of ANSI and IBM's BSC protocol. They are very similar and the latter is used widely by industry. In Sec. 3.3.2.1, we shall describe the main features of BSC as documented in [IBM 70] to illustrate this class of protocols.

Bit-oriented protocols include ADCCP, which is a standard of ANSI; HDLC, which is a standard of ISO (International Standards Organization); and IBM's SDLC protocol. In Sec. 3.3.2.2 we describe the version of SDLC as documented in [IBM 75] to illustrate this class of protocols.¹ We note that SDLC, HDLC, and ADCCP are basically of the same genre; the two protocol standards have evolved out of IBM's submissions to the respective standards organizations. However, additional functions have been defined (or proposed) for HDLC and ADCCP, which are not present in the current version of SDLC [IBM 75]. These functions are discussed in Sec. 3.4.

Other computer manufacturers have also proposed similar DLC protocols, including BDLC of Burroughs, UDLC of Sperry-Univac, and DDCMP of Digital Equipment Corporation. DDCMP is byte-count oriented. An excellent discussion of DDCMP may be found in [MCNA 77].

3.3.2.1 Description of a Character-Oriented Protocol—BSC [IBM 70]

Suppose that bit synchronism is maintained between a receiver and a transmitter. Since both data and control must be encoded and transported in packages of bits called frames,² the synchronization problem at the next level involves the following:

¹Excellent discussions of SDLC concepts may be found in [CYPS 78, DONN 74].

²The term "frame" has been defined in this chapter but is not defined in [IBM 70].

Bit positions 4, 5, 6, 7		Bit positions 0, 1, 2, 3															
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000	0	NUL	DLE	DS		SP	&	—					{	}	/	0	
0001	1	SOH	DC1	SOS					a	j	~		A	J		1	
0010	2	STX	DC2	FS	SYN				b	k	s		B	K	S	2	
0011	3	ETX	DC3						c	l	t		C	L	T	3	
0100	4	PF	RES	BYP	PN				d	m	u		D	M	U	4	
0101	5	HT	NL	LF	RS				e	n	v		E	N	V	5	
0110	6	LC	BS	EOB ETB	UC				f	o	w		F	O	W	6	
0111	7	DEL	IL	PRE ESC	EOT				g	p	x		G	P	X	7	
1000	8		CAN						h	q	y		H	Q	Y	8	
1001	9	RLF	EM						i	r	z		I	R	Z	9	
1010	A	SMM	CC	SM		¢	!		:								
1011	B	VT				.	\$,	#								
1100	C	FF	IFS		DC4	<	*	%	@								
1101	D	CR	IGS	ENQ	NAK	()	_	'								
1110	E	SO	IRS	ACK		+	:	>	=								
1111	F	SI	IUS	BEL	SUB	!	?	"									



Duplicate assignment

Figure 3.4. EBCDIC character assignments.

(1) upon reception of a sequence of bits, to determine which bit is the beginning and which bit is the end of a frame; and (2) error detection.

In character-oriented protocols, all data and control information are encoded as characters. The IBM manual lists three character-code sets that can be used for BSC. They are EBCDIC with 8 bits per character for 256 assignments, USASCII with 7 bits plus a parity bit per character for 128 assignments, and 6-bit Transcode with 64 assignments. As an example the EBCDIC assignments are illustrated in Fig. 3.4.

Frame Synchronization

With a character-code set, BSC data and control characters are represented by unique bit sequences. (We discuss below how transparent data, i.e., an arbitrary stream of bits, can be transmitted.) In particular, the bit sequence representing the character SYN is unique. Because of the serial-by-character nature of transmission, frame synchronization and character synchronization are accomplished at the same time. Each frame transmitted is always preceded by two or more SYN characters. The receiver will hunt for the bit pattern of SYN SYN. (We use the shorthand notation of \emptyset for SYN SYN as in [IBM 70].) When the receiver detects a sequence of two or more SYN characters in an incoming signal, the bits immediately following (6 or 8 bits, depending on the code set) will form the first character of the frame.³ The receiver may detect more than two SYN characters in a row because SYN is also used to fill any idle time in between frames. If the frame contains data characters, it ends with either one of the following control characters: ITB, ETB, or ETX followed by two BCC characters for error detection. (See below for further explanation of these control characters.) If the frame contains only control information, one or more control characters may be present, representing one of a well-defined set of control messages.

We said earlier that bit synchronism may be maintained by the transceiver or DTE in a station. If it is done by the DTE, a special bit synchronization pattern needs to precede the frame synchronization pattern \emptyset . (This bit synchronization pattern can be the hex '55' '55' or the SYN SYN SYN SYN pattern that provides for the 16 transitions necessary for bit synchronization.) Bit synchronism will be assumed from now on.

To ensure that the first and last characters of a frame are properly transmitted by the transceiver, which may turn on and off abruptly, BSC requires the addition of a pad character immediately before the synchronization characters of a frame and immediately afterward. [The leading pad character may consist of alternating 0 and 1 bits (hex '55') or a SYN character. The trailing pad character consists of all 1 bits (hex 'FF').] With the frame synchronization method, BSC frames containing data and/or control information from a sender to a receiver will be in one of the following two formats:

PAD \emptyset [heading and/or data] BCC BCC PAD

PAD \emptyset [control] PAD

The two BCC characters are for error detection. The exact nature of the heading, control, and data fields in the frame formats above will be clarified in our discussions below on multiple access synchronization and content synchronization. For simplicity, we drop the leading and trailing PAD characters in our notation from now on.

Another function at the frame synchronization level is to detect the presence of

³Note that this procedure is subject to errors. First, the \emptyset sequence may be missed because of bit errors in it. On the other hand, the two-character bit pattern of \emptyset is not unique and may appear in the middle of certain three-character sequences.

errors within transmitted frames. Depending on the character-code set selected, one or more of the following three error-checking methods may be used for BSC:

VRC	Vertical redundancy checking
LRC	Longitudinal redundancy checking
CRC	Cyclic redundancy checking

With each method, two BCC characters are generated at the sender and are included at the end of each data frame. They are used for error detection at the receiver. Note from the frame formats above that BSC control messages are not protected by error checking.

CRC is a much more powerful technique for error detection than VRC or LRC and can detect many types of multiple errors. It is to be preferred since errors in communications transmission systems tend to occur in bursts [BURT 72]. CRC is available with all three BSC character-code sets. For a detailed description of LRC, VRC, and the CRC coding algorithm, the reader is referred to [MART 70, MCNA 77].

Multiple Access Synchronization

The multiple access synchronization function determines which of a population of transmitters may access a shared communications channel. If the communications channel is dedicated to a single sender-receiver pair, the problem is trivially solved. Otherwise, some multiple access protocol is required.

The method for multiple access depends on the communications link configuration. BSC permits two types of communications link configurations between stations: point to point and multipoint. We consider first a point-to-point communications link between two stations. Remember that each station is both a sender and a receiver. BSC requires a *half-duplex* mode of operation such that only one station can be transmitting at a time. Half-duplex operation is required even if the communications link consists of two separate communications channels, one for each of the two directions of transmission. Note that whenever the line “turns around,” reversing the direction of transmission, character synchronization (and possibly bit synchronization) need to be reestablished.

A point-to-point communications link is shared via a *contention* protocol. If a station wants to acquire control of the line, it sends the initialization sequence

Ø ENQ

The other station replies with

Ø ACK0	if ready to receive
Ø NAK	if not ready to receive
Ø WACK	if temporarily not ready to receive (try again later!)

Since it is possible that both stations bid for the line at about the same time, to avoid

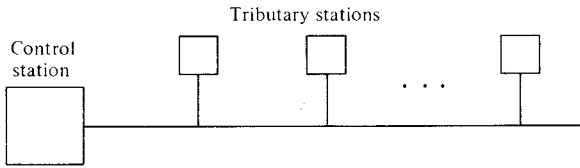


Figure 3.5. Multipoint link configuration.

any deadlock each station is designated to be primary or secondary. The primary station has high priority. When bidding for the line, the primary, but not the secondary, can continue to retry (up to a certain limit) until it gets an affirmative response.

The station with control of the line is the *transmitting station* and can transmit frames containing data to the *receiving station*. Following each such frame transmitted, the line may turn around for the receiving station to send a reply. The reply is a control message (ACK, NAK, etc.) and may not contain any data, except in the limited conversational mode (see below). A station relinquishes control of the line with the control message \emptyset EOT. For 3 seconds the other station can bid for the line without competition. After 3 seconds, both stations may bid for control of the line.

BSC also permits a multipoint communications link configuration (see Fig. 3.5), with a control station and many tributary stations sharing the same line. Data flow is always between the control station and one of the tributary stations. The communications method is again half-duplex, so that transmission of data can take place in only one direction at a time, either from the control station to a tributary or from a tributary to the control station. There are two additional functions required: (1) sharing of the line among the tributary stations; and (2) addresses for identifying the tributary stations.

The communications link is shared among the tributary stations via *polling* and *selection* protocols under the supervision of the control station. The control station directs its incoming traffic by sequentially polling each tributary station. When polled, a tributary station has control of the line and assumes the role of the transmitting station with the control station acting as the receiving station, such as previously described for a point-to-point line. The control station directs its outgoing traffic by first selecting a tributary station as the receiving station. The control station then acts as the transmitting station. In both cases, the transmitting-receiving relationship is ended when the transmitting station sends the message \emptyset EOT. If the transmitting station is a tributary station, control of the line is passed back to the control station.

The following sequence is used by the control station to poll or select a tributary station:

\emptyset EOT PAD \emptyset [polling or selection address] ENQ

The polling or selection address sequence consists of one to seven characters. It gives the tributary station address (and also a specific device address if the station has several available). The possible replies from a polled tributary station are:

1. A frame containing data
2. Ø EOT: negative reply when the station has nothing to send
3. Ø STX ENQ: temporary text delay (TTD) when the station is unable to transmit its initial data within 2 seconds; this permits the polled station to retain control of the line and avoid being timed out by the control station

The possible replies from a selected tributary station are:

1. Ø ACK0: affirmative, ready to receive
2. Ø NAK: negative, not ready to receive
3. Ø WACK: temporarily not ready to receive

Content Synchronization

With multiple access synchronization, two processes can send frames of bits to each other. The next level of functions is to enable a receiver to interpret correctly the content of a frame. There are three functions involved here:

1. To differentiate between control and data information within a frame; for control information, the receiver and sender must also agree on the encoding of a set of control messages.
2. To distinguish an arbitrary bit stream of data (transparent data), although some of the bits may be the same as control characters.
3. To remedy errors so as to ensure that a single error-free copy of each frame at the sender will arrive in sequence at the receiver within a reasonable amount of time.

Encoding of Data and Control. The separation of data and control is solved in BSC with the use of a character-code set. Data transfer is always in the form of a serial stream of characters. Characters in the code set are reserved to represent either data or control. BSC control characters are for DLC control functions as well as other high-level control functions, such as device control. (The mixing of control functions at different levels is one of the frequent criticisms of BSC.) Specifically, control of the data link is achieved through use of the following control characters and two-character sequences:

SYN	Synchronous Idle
SOH	Start of Heading
STX	Start of Text
ITB	End of Intermediate Transmission Block
ETB	End of Transmission Block
ETX	End of Text
EOT	End of Transmission
ENQ	Enquiry

ACK0/ACK1	Alternating Affirmative Acknowledgments
WACK	Wait-Before-Transmit Positive Acknowledgment
NAK	Negative Acknowledgment
DLE	Data-Link Escape
RVI	Reverse Interrupt
TTD	Temporary Text Delay
DLE EOT	Disconnect Sequence for a Switched Line

Several of the above are not defined in the character code sets but are represented as two-character control sequences. They are ACK0, ACK1, WACK, RVI, and TTD. Some minor variations in the designation of the characters and compositions of the character sequences exist among the different code sets. For example, ACK0 and ACK1 correspond to the two-character sequences DLE 0 and DLE 1 in USASCII but are coded as DLE '70' and DLE / in EBCDIC. On the other hand, TTD is represented by STX ENQ in all code sets.

BSC frames have the following formats:

```

Ø SOH [heading] STX [data] ETX BCC BCC
Ø [control character sequence]

```

Frames containing data or heading are sent only by a transmitting station to a receiving station. In these frames, SOH marks the beginning of a heading containing control information for high-level (non-DLC) functions such as message identification, routing, device control, and priority. STX marks the beginning of the data section. Either SOH [heading] or STX [data] may be absent in a frame. Each frame is terminated by one of the control characters ETX, ETB, or ITB. In all cases it is followed by two BCC characters for error detection.

Both ETX and ETB terminate frames started by STX or SOH and cause the communications channel to turn around and require a reply from the receiving station. ETX terminates a data message at the data processing level. At the DLC level, a data message may be segmented into *transmission blocks* for ease of processing and more efficient error control. Each transmission block begins with STX and ends with ETB, except for the last block of a data message, which ends with ETX.

The "SOH [heading] STX [data]" portion, where [data] can be either a complete data message or a transmission block, can be further segmented into *intermediate blocks* for increased reliability in error detection. When an intermediate block is sent in a frame, the frame is terminated by ITB, again followed by two BCC characters. The last intermediate block in a sequence is terminated by ETB or ETX as appropriate. After the first intermediate block, succeeding ones need not be preceded by STX or SOH in a frame (except in the case of transparent data to be discussed below). Frames terminated by ITB do not cause a line turnaround or require a reply from the receiver. Each sequence of intermediate blocks, comprising a data message or a transmission block, is treated as a whole by the receiving station. All BSC stations

must have the ability to receive intermediate blocks. The ability to send intermediate blocks is an option.

Frames containing control information only have the format

\emptyset [control character sequence]

We have seen the use of some of these control messages for multiple access synchronization. We shall encounter some others in the following discussions.

Transparent Data. So far in our description of BSC, the characters that can be sent in the data portion of a frame have been assumed to be limited to a subset of the characters in the character-code set. To send an arbitrary stream of characters (transparent data) without some of the characters taking on any control meaning, BSC provides the *transparent text mode* of operation within a frame. During the transparent text mode, any control characters transmitted must be preceded by DLE to be recognized as denoting a control function. In particular, the transparent text mode is initiated by DLE STX and is terminated by any of DLE ETX, DLE ETB, or DLE ITB. (In the last case, if the succeeding intermediate block is also transparent, that intermediate block must begin with DLE STX. Recall that a beginning STX is not required of intermediate blocks in the nontransparent mode of operation if that intermediate block is not the first one.) Thus a frame in a transparent text mode may look like

\emptyset DLE STX [transparent data] DLE ETX BCC BCC

If a control character is needed within the transparent data, a DLE character is inserted in front of it. Also, if a DLE character appears in the transparent data (with no control meaning), an additional DLE must also be inserted in front of it lest it be misinterpreted as preceding a control character. The inserted DLC characters need to be stripped off at the receiver. The following procedure may be used by the receiver:

Step 1. Hunt for a DLE character in the input stream. When a DLE is received, discard it and go to Step 2.

Step 2. If the next character is a DLE *then* accept it as a data character and go to Step 1,

else accept the next character as a control character and go to Step 1.

Error Control. One of the content synchronization functions is to recover from transmission errors. BSC does so with an automatic-repeat-request (ARQ) technique. When a transmitting station ends a frame containing data with ETB or ETX, a reply is required of the receiving station. It can reply with \emptyset ACK0 or \emptyset ACK1, indicating "data accepted, ready for more," or \emptyset NAK, indicating "data not accepted, retransmission necessary." The line then turns around again for the transmitting station to send a frame. If necessary, retransmission of a frame of data is attempted a number of times following the initial NAK, after which recovery action is required. If the transmitting station receives no reply after sending a frame of data, due to either the

frame or the reply being garbled, the transmitting station can request a reply from the receiving station by sending the control message \emptyset ENQ. The receiving station repeats the previous response: NAK, ACK0, or ACK1, as appropriate.

Two positive acknowledgment responses are used for the following reason. ACK0 is used as an affirmative response to the transmitting station's initial ENQ and for an affirmative response to all even-number transmission blocks. ACK1 is used for all odd-number transmission blocks. The odd or even count of a block does not change even when it is retransmitted. As a result the alternating affirmative replies enable the sender to detect the loss of a transmission block. Also, since transmission blocks are sent one at a time until successfully received and acknowledged, the original sequence of transmission blocks at the sender is preserved at the receiver. However, a transmission block does not actually contain a bit indicating it to be even or odd. It is thus not possible for the receiver to detect a duplicate transmission block. (A duplicate would be sent if, for instance, an ACK was garbled and turned into a NAK.)

Recall that only frames containing heading/data are protected by error detection. Control messages are not checked. The BSC manual mentions some provisions for using trailing pad characters for format checking which serves as a limited means of error detection.

The "accumulation" for the BCC characters begins right after the first STX or SOH character following either a line turnaround or a previous ITB termination of an intermediate block. All characters following that, including control characters, are used in the accumulation, with the exception of SYN idle characters inserted for bit and character synchronization considerations (see below).

Dialogue Synchronization

With content synchronization, it is now possible to send frames containing data from one station to another. There also exists a set of control messages which can be used for coordinating the actions and responses of two communicating stations in a dialogue fashion. The dialogue level functions consist of:

1. Initial dialogue synchronization
2. Recovery methods in the event of loss of synchronism and
3. Dialogue termination

Initial dialogue synchronization in BSC is trivially accomplished once multiple access synchronization has been established, with one station being in the "transmit" state and the other in the "receive" state.

In the case of switched-network (dial-up) operation, additional initial synchronization procedures are needed immediately following the establishment of a communications circuit. The calling station sends either one of the following messages:

\emptyset ENQ	Who are you?
$\emptyset \alpha \dots$ ENQ	I am $\alpha \dots$. Who are you?

The called station replies positively with

Ø $\alpha \dots$ ACK0

where the called station identity $\alpha \dots$ is optional. The called station replies negatively with

Ø NAK

Ø WACK

The identity sequences $\alpha \dots$ may be 2 to 15 characters long. The minimum two-character sequence must consist of the same character transmitted twice (for reliability).

A dial-up call between two stations can be terminated by either one of the stations transmitting the disconnect message

Ø DLE EOT

A call may also be disconnected by the disconnect timeout to be described below.

Apart from the transmit and receive states, the stations may transit to other states as a result of the following control messages.

TTD (Temporary Text Delay). The TTD control sequence is sent by a transmitting station when it wishes to retain control of the line but is not ready to transmit data. It is sent after approximately 2 seconds to avoid the 3-second receive timeout at the receiving station. The receiving station responds NAK and waits for transmission to begin again. TTD can be repeated one or more times. TTD can also be used to abort a transmission. If following the receiving station's reply of NAK, the transmitting station sends EOT, the transmit-receive relationship is terminated. (This is called the forward abort sequence.)

RVI (Reverse Interrupt). RVI is a positive acknowledgment used in place of ACK0 or ACK1 (depending on the current odd-even count). RVI is transmitted by a receiving station to request termination of the current transmit-receive relationship. The transmitting station treats RVI as a positive acknowledgment (0 to 1 as appropriate) and proceeds to transmit all data that might prevent it from becoming a receiving station. The ability to receive RVI is mandatory, but the ability to transmit RVI is optional.

WACK (Wait-Before-Transmit Positive Acknowledgment). WACK allows a receiving station to indicate a "temporarily not ready to receive" condition to the transmitting station. The normal transmitting station response to WACK is ENQ. The receiving station may continue to respond with WACK. The transmitting station has the option of terminating the dialogue by EOT or disconnecting a circuit with DLE EOT. The ability to receive WACK is mandatory. The ability to send WACK is optional.

Timeouts. Timeouts are used to attempt recovery from possible nonsynchronous conditions. There are four specific timeouts used in BSC:

1. *Transmit timeout.* This is a nominal 1-second timeout that establishes the rate at which synchronous idles are automatically inserted into frames containing heading and data. Ordinarily, SYN SYN is inserted every second. For transparent data, DLE SYN is inserted instead. These insertions permit character synchronization to be checked and in the case of SYN SYN, reestablished. If bit synchronism is maintained by the DTE, insertion of DLE SYN is required at least every 84 characters to ensure bit synchronism amid transitionless data. The inserted synchronous idle characters are not included in BCC accumulation for error detection.
2. *Receive timeout.* This is a nominal 3-second timeout and is used to limit the waiting time tolerated by a station to receive a reply as well as for other functions.
3. *Disconnect timeout.* This timeout is used optionally on switched network connections. It is used to prevent a station from holding on to a connection for a prolonged period of inactivity. After 20 seconds of inactivity, the station will disconnect from the switched network.
4. *Continue timeout.* This is a nominal 2-second timeout associated with the control messages TTD and WACK.

Additional BSC Features

1. A frame containing heading/data can be terminated prematurely by using an ENQ character which signals the receiving station to disregard the current frame. NAK is always the reply from the receiving station.
2. BSC permits a "limited conversational mode" whereby a frame containing heading/data can be sent in place of a positive acknowledgment by the receiving station as a reply to a frame that ended with ETX or DLE ETX. The transmitting station must then reply with a control message and not with another conversational reply.

Comments on the Structure of BSC

We have given a description of the BSC protocol using the hierarchical functional levels in Fig. 3.2 as a framework for elaboration. Ideally, a well-structured protocol should have minimal interfaces defined between adjacent functional levels so that functions and implementation techniques at one level can be changed without affecting adjacent levels if the interfaces are maintained to be the same. Since BSC was not originally designed with the hierarchical functional levels in mind, the decomposition of BSC functions into different levels such as we have done is somewhat awkward at places. We illustrate this observation by pointing out some dependencies between the BSC functional levels described earlier in this section.

1. The half-duplex mode of operation facilitates the logical design of dialogue

interactions between two communicating stations. Its effect pervades all functional levels. For instance, bit and character synchronization need to be reestablished whenever a line turns around. The stop-and-wait form of ARQ technique is used for error control.

2. The transmit–receive coordination of two communicating stations at the dialogue level is directly coupled to the transmit–receive coordination at the multiple access level of all stations sharing a multipoint link. The dialogue-level control messages WACK and TTD are used as an integral part of the multiple access protocol.

3.3.2.2 Description of a Bit-Oriented Protocol —SDLC [IBM 75]

The SDLC protocol will also be described in terms of the hierarchy of synchronization functional levels in Fig. 3.2; however, techniques used in SDLC for solving these synchronization problems are different from BSC and other character-oriented protocols.

Again, we begin on the premise that bit synchronism is maintained by the transceivers or the DTE. In either case, our description of data link control will be concerned only with avoiding the transmission of transitionless data.

Frame Synchronization

A bit-oriented protocol, as its name suggests, deals with a stream of bits rather than a stream of characters. A character-code set is therefore not required. To delimit the beginning and end of a frame, a unique sequence of 8 bits 01111110 called a flag, denoted by F, is used. So each frame looks like

F [control and data] FCS F

where FCS consists of 16 CRC bits for error detection. The ending F may be followed by a frame, by another F, or by an idle condition. A series of contiguous F's may be transmitted to maintain bit synchronism and to maintain the data link in an active state. An idle state is perceived by the receiver when he receives a succession of 15 or more consecutive binary ones.

The F bit sequence 01111110 is made unique by zero insertion at the transmitting station and zero deletion at the receiving station, performed as part of the DLC protocol. The transmitter inserts a binary zero after any succession of five consecutive 1's inside a frame following the beginning F. The receiver removes a zero that follows a received sequence of five continuous 1's. We shall see below that with unique flags, the transmission of transparent data is always possible rather than handled as a special case.

The contents of every SDLC frame, with the exception of the beginning and ending flags, are protected by cyclic redundancy checking. Specifically, the generating polynomial, $x^{16} + x^{12} + x^5 + 1$, which is a CCITT standard, is used for generating the error detection bits [MCNA 77]. These bits are put into the FCS field immediately preceding the ending flag of a frame. Note, however, that the computation of the FCS

bits at the sender is done before the zero insertion operation. Therefore, the inserted zeros are not protected by FCS. At the receiver, the deletion of inserted zeros is performed before the FCS computation.

When data signal transitions are needed by the DTE or transceiver for bit synchronism considerations, SDLC specifies the zero complementing NRZI (Non-Return-to-Zero-Inverted) coding method, with which, to send a binary 1 in the data, the signal on the line remains in the same state. The signal switches to the opposite state to send a binary 0 in the data. Thus an extended period of binary zeros in the data is transmitted as continuous transitions in the channel. A long period of binary 1's in the data will have transitions in the channel as a result of the zero insertion requirement to make the flags unique. If NRZI is used, it must be used by all DTEs on the data link.

Multiple Access Synchronization

The basic data link configurations that are currently supported by the SDLC protocol are:

1. Point-to-point half-duplex line, switched or nonswitched
2. Point-to-point duplex line, nonswitched
3. Multipoint duplex line, nonswitched
4. Unidirectional loop consisting of point-to-point simplex line segments

where

- Simplex means data-carrying ability in one direction only
- Half-duplex means data-carrying ability in both directions but not at the same time
- Duplex means simultaneous bidirectional data-carrying ability

In any of the SDLC data link configurations, exactly one of the stations is designated to be the primary station of the data link. Thus each data link has a primary station and one or more secondary stations. Communication of data always takes place between the primary station and a secondary station. Secondary stations, if more than one is present, cannot communicate with each other. We remind the reader that sometimes a station has the capability to implement only half-duplex protocols, even if a full-duplex line is available. In a multipoint configuration, it is often the case that only the primary station operates duplex, while secondary stations operate half-duplex. Thus the primary may be transmitting to one secondary while simultaneously receiving from another secondary station.

With a point-to-point duplex line, both transmitter-receiver pairs have their own dedicated channels. In this case, multiple access synchronization is not necessary.

For all other configurations, sharing of a channel by multiple transmitter-receiver pairs is involved and multiple access synchronization is needed for conflict resolution. In SDLC, the encoding of control messages used for multiple access, content, and dialogue synchronization levels is integrated. Therefore, it is necessary for us to consider now the format of an SDLC frame.

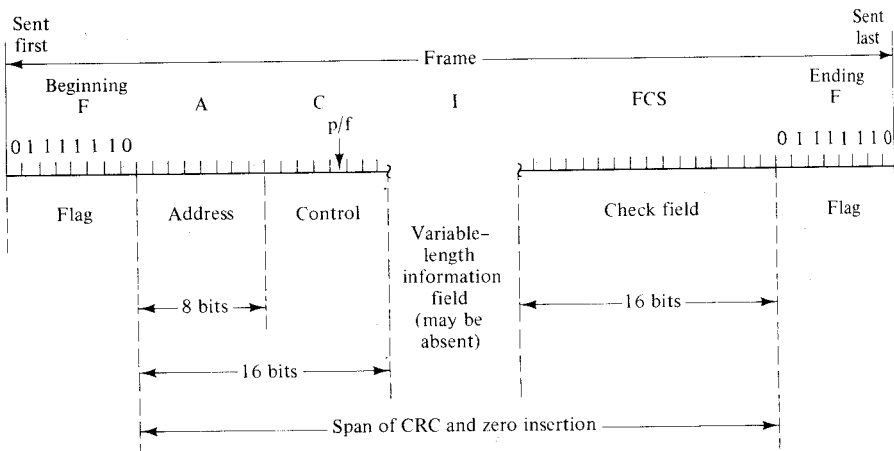


Figure 3.6. Fields of an SDLC frame.

With character-oriented DLC protocols, the character-code set usually provides for control characters. A sequence of one or more such control characters forms a control message. This approach necessitates special handling of transparent data.

Bit-oriented protocols separate control and data information by putting them in different parts of a frame. This approach, called *positional significance*, is possible because the beginning and ending flags can be uniquely identified (through zero insertion and deletion at the frame synchronization level) and thus provide reference points for locating data and control fields. Specifically, a frame consists of the fields shown in Fig. 3.6. For multiple-access synchronization we are presently interested only in the address field and the p/f-bit in the control field.⁴

In each of the SDLC data link configurations, there is a primary station and one or more secondary stations. The primary station acts as a central controller for data link access. It carries out this function using the address (A) field and the poll/final (p/f) bit in the control field. Since there is only one primary station, it never needs to be identified. The address of the secondary station in communication with the primary is always indicated in the address field (see Fig. 3.6). If the frame is from secondary to primary, the A field identifies the sender. If the frame is from primary to secondary, the A field identifies the intended receiver. In frames from the primary, a common address may be used for a group of secondary stations. In particular, an all-1's address field is often used as a broadcast or all-stations address.

In frames sent by the primary, the p/f-bit is a *poll bit*. In frames sent by a secondary station, the p/f-bit is a *final bit*. When the primary sends a frame to a secondary station with the poll bit turned on, it demands a response from the secondary station. Access right is also automatically granted to that secondary station for using the data link. The secondary station can then send a sequence of frames on the data link; the number of frames is limited only by error and sequence control

⁴We use lowercase letters p and f to avoid confusion with F, which denotes a flag.

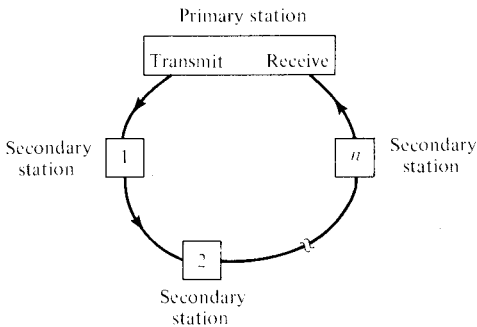


Figure 3.7. Loop configuration.

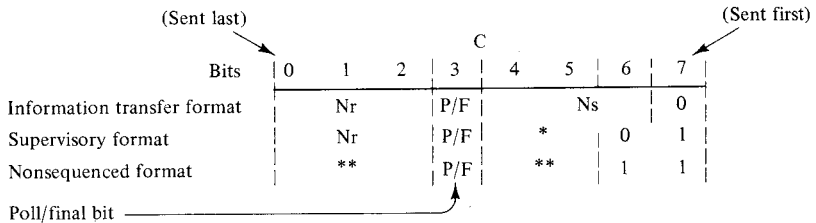
considerations to be addressed later. The secondary station relinquishes its access right when it sends a frame with the final bit turned on.

This use of the poll/final bit is adhered to for all primary-secondary dialogues even for a point-to-point full-duplex data link configuration where each transmitter-receiver pair has its own dedicated channel (see the discussion of the normal response mode in the dialogue synchronization section below).

In a point-to-point half-duplex line, the primary and secondary stations alternate use of the shared data link. In a multipoint full-duplex line, the primary-to-secondary channel is dedicated to the primary transmitter, which can send frames to any one of the secondary stations by identifying it in the address field. The secondary-to-primary channel is shared by the secondary stations and access right to the shared channel is controlled by the primary station via polling using the poll bit.

An SDLC loop consists of a series of point-to-point simplex lines. Each secondary station ordinarily acts as a repeater. Bits sent out of the primary station are relayed from one secondary station to the next one downstream until they return to the primary station (see Fig. 3.7). Such a loop is therefore unidirectional and half-duplex procedures between the primary and a secondary are used. The primary transmits a frame to any one of the secondary stations by identifying it in the address field. Or it may command a response from a secondary station by polling it.

The poll may be specific, as in point-to-point and multipoint configurations, with the poll bit turned on in a primary-to-secondary frame. Or the poll may be for a group of secondary stations with a common address. In the latter case the primary station polls by first sending an NSP (nonsequenced poll) control message with a common address in the address field (see below for the encoding of NSP and other control messages). After the NSP, the primary sends a binary zero followed by continuous 1's, which starts a *polling cycle*. If the poll bit in the NSP frame is also turned on, the addressed secondary stations are required to respond. If the poll bit is off, the addressed secondary stations are only invited to transmit. The response to an NSP control message requires turning on the final bit only if the command had the poll bit turned on.



*Codes for supervisory commands/responses
 **Codes for nonsequenced commands/responses

Figure 3.8. SDLC control field formats.

Thus a polling cycle begins with the transmission of an NSP control frame by the primary, followed by a “go-ahead” pattern consisting of a binary zero followed by continuous 1’s. The first down-loop secondary station that has been polled receives the go-ahead when it has counted a zero and seven consecutive 1’s in its input signal. It repeats the first six 1’s and changes the seventh one to a binary zero. Thus an SDLC flag is generated. That secondary station then suspends the repeater function and transmits from itself to the primary station one or more frames. When it is finished, it resumes the repeater function. The ending binary zero in the ending flag of the last frame followed by continuous 1’s now creates a go-ahead pattern again. The access right to the shared channel is then passed on to the next downstream secondary station that has been polled. Finally, when the primary station receives seven or more continuous 1’s, the polling cycle is complete and the primary station is back in control of the data link.

Content Synchronization

As before, the functions included in this functional layer are: the separation of data and control, the encoding of control messages, error control, and the sequencing of frames.

The first function is provided using the positional significance approach discussed earlier: control and data are transmitted in different fields of a frame. The positions of these fields are measured from the beginning and end of the frame marked by unique flags. Unlike character-oriented protocols, the data transmitted are always considered to be transparent data and no special handling is required.

Depending on the values of bits 6 and 7 in the control field of a frame, SDLC defines three different kinds of frames with different formats:

- Information transfer (I) frames
- Supervisory (S) frames
- Nonsequenced (NS) frames

The control field formats of these three kinds of frames are shown in Fig. 3.8, where bit 7 differentiates between I-frames and the others; bit 6 differentiates between S-

Format (note 1)	Binary configuration		Acronym	Command	Response	I-Field prohibited	Resets Nr and Ns	Confirms frames through Nr-1	Defining characteristics
	Sent last	Sent first							
NS	000	P/F	0011	NSI	X	X			Command or response that requires nonsequenced information
	000	F	0111	RQI		X	X		Initialization needed; expect SIM
	000	P	0111	SIM	X		X	X	Set initialization mode; the using system prescribes the procedures
	100	P	0011	SNRM	X		X	X	Set normal response mode; transmit on command
	000	F	1111	ROL		X	X		This station is off line
	010	P	0011	DISC	X		X		Do not transmit or receive information
	011	F	0011	NSA		X	X		Acknowledge NS commands
	100	F	0111	CMDR		X			Nonvalid command received; must receive SNRM, DISC, or SIM
	101	P/F	1111	XID	X	X			System identification in I field
	001	0/1	0011	NSP	X		X		Response optional if no P-bit
	111	P/F	0011	TEST	X	X			Check pattern in I field
S	Nr	P/F	0001	RR	X	X	X	X	Ready to receive
	Nr	P/F	0101	RNR	X	X	X	X	Not ready to receive
	Nr	P/F	1001	REJ	X	X	X	X	Transmit or retransmit, starting with frame Nr
I	Nr	P/F	Ns 0	I	X	X		X	Sequenced I-frame

Note 1: NS = nonsequenced, S = supervisory, I = information

Figure 3.9. Summary of SDLC commands and responses

and NS-frames. The p/f-bit in the control field has been encountered earlier for multiple-access synchronization. It is also used for some dialogue synchronization functions. The bit sequences N_T and N_S are for error and sequence control of I-frames. The remaining bits (marked * and ** in Fig. 3.8) are used to encode various control messages. The rest of the frame for all three formats are identical except for the I-field (see Fig. 3.6). In I-frames, the I-field can accommodate an arbitrarily long sequence of data bits, which must be a multiple of 8 bits. Otherwise, its length is subject only to error detection performance considerations. The I-field is not permitted in S-frames, while the I-field may be used to transmit link management data in NS-frames.

A listing of the encoding of SDLC control messages defined in [IBM 75] is shown in Fig. 3.9. We note that not all the bit sequences have been assigned. Hence additional control messages may be defined in the future as it becomes necessary.

A significant advantage of bit-oriented protocols over character-oriented protocols is the ease with which the set of control messages can be expanded. Recall that in BSC, the character-code set determines the available control characters. To define additional control messages one must employ longer and longer sequences of control characters. Character-oriented protocols are inefficient since only a very small

number of the 8-bit sequences in a character is used for encoding control information. While with the positional significance approach in bit-oriented protocols, theoretically all 2^8 possible bit sequences can be used to encode control information.

We talk next about the error and sequence control functions of SDLC. Recall that error detection is performed for all frames. In addition, all I-frames are numbered in sequence. Sequence numbering is used to detect lost (garbled) frames or duplicated frames so as to guarantee that I-frames are delivered in their original order.

Recall that in BSC, data frames must be transmitted and acknowledged one at a time. Hence their order is preserved. Furthermore, the use of ACK0 and ACK1 for odd- and even-numbered transmission blocks ensures that a missing frame will be detected.

In SDLC, a transmitter can transmit up to seven I-frames before receiving positive acknowledgment. This is possible because the sequence numbering fields N_S and N_R are 3-bit sequences and have a counting capacity of 8.

Each station sequentially numbers every I-frame it transmits. This count is known as N_S , which is the sequence number (modulo 8) of the transmitted frame. The same station also keeps track, sequentially, of error-free I-frames it has received. N_R is the sequence number (modulo 8) of the next I-frame that it expects.

Consider two communicating stations, 1 and 2, with transmit and receive counts $V_S(1)$ and $V_R(1)$ at station 1, $V_S(2)$ and $V_R(2)$ at station 2. Suppose that an I-frame is sent from station 1 to station 2. It contains $N_S(1) = V_S(1)$, which is the sequence number of that particular I-frame. $V_S(1)$ is then incremented by 1. At station 2, if an error is detected in the frame, that frame is simply rejected. Suppose that it is error free. $N_S(1)$ is then compared to $V_R(2)$, which is the sequence number expected by station 2. If they agree, the frame is accepted and $V_R(2)$ is incremented by 1. If $N_S(1) \neq V_R(2)$, the received frame is rejected. [There are two possible cases. $N_S(1) < V_R(2)$ means that the received frame is a duplicate. On the other hand, $N_S(1) > V_R(2)$ means that the received frame is out of sequence and some I-frames are missing. Since $N_S(1)$ and $V_R(2)$ are modulo 8 numbers, the two cases cannot really be distinguished. However, the ability to distinguish the two cases is not necessary to achieve the objective of error and sequence control.]

Next consider the $N_R(1)$ field in the I-frame from station 1 to station 2. $N_R(1)$ is set equal to $V_R(1)$ and is a positive acknowledgment from station 1 to station 2 confirming correct reception of previously sent I-frames, from station 2 to station 1, with contiguous sequence numbers (modulo 8) up to and including $N_R(1) - 1$. The positive acknowledgment indicated by $N_R(1)$ is accepted as long as the I-frame is error free, even though it may be out of sequence. Buffers containing acknowledged frames may then be released.

We note several things regarding error and sequence control:

1. On a duplex data link, a transmitter can transmit I-frames continuously as long as it receives positive acknowledgments from incoming I-frames so that it does not have seven outstanding unacknowledged I-frames. On a half-duplex data

link, at most seven I-frames can be transmitted at a time before the line turns around for an acknowledgment.

2. A single positive acknowledgment can acknowledge up to seven frames at the same time. This is unlike BSC, in which data frames must be acknowledged individually.
3. If an error is detected in any I-frame, all I-frames following it need to be retransmitted even if they have been correctly received. This protocol ensures that the original order of the I-frames is preserved at the receiver.
4. Supervisory and nonsequenced frames are not numbered. They are not therefore protected by sequence numbering against loss, duplication, or out-of-order reception. All frames, however, have CRC error detection.
5. Supervisory frames have the N_R field and can therefore be used to provide positive acknowledgments.
6. In the discussion above, the $N_R(1)$ count received by station 2 acknowledges correct reception of I-frames up to and including $N_R(1) - 1$. Note that this serves as a positive acknowledgment. Alternatively, the supervisory frame REJ can be used by station 1 to demand station 2 to transmit or retransmit starting with frame $N_R(1)$. In this respect, REJ is like a negative acknowledgment.

Dialogue Synchronization

In both multipoint and loop configurations, there are multiple secondary stations which may be engaging in concurrent dialogues with a single primary station. However, having achieved multiple access synchronization, we can consider each dialogue individually. A dialogue involves only two stations: a primary and a secondary. Control messages from primary to secondary are called *commands*. Control messages from secondary to primary are called *responses*. The primary always transmits first to initiate a dialogue, although on a switched data link either one may call the other first to make the physical connection. We note that in a network environment, a station may be attached to more than one data link; it can be a primary on one data link but a secondary on another data link.

The responses of a secondary depend on its "mode" status. The primary station of a dialogue can command one of three modes at the secondary station:

Normal disconnected mode (NDM)

Initialization mode

Normal response mode (NRM)

A secondary station that receives a DISC command assumes NDM; it also assumes NDM when power is turned on or when a switched connection is initially made.

Valid commands with the p-bit turned on cause a disconnected secondary station to respond with a request for on-line status (ROL) or, if needed, a request for initialization (RQI). In the latter case, the command SIM is expected, which causes the secondary to enter initialization mode and starts predefined initialization proce-

dures stored at the secondary. The command SNRM puts the secondary in the NRM and subordinates the secondary to the primary; a secondary cannot transmit unless it has been polled by the primary. The primary and secondary transmit-receive counts V_T and V_S are reset to 0. The secondary station remains in NRM until it receives a DISC or SIM command. NSA is the affirmative response to the mode-setting commands SNRM, DISC, or SIM.

In SDLC, data transfer between primary and secondary takes place when the secondary is in NRM. In this mode, the secondary has access right to transmit only after it has received a frame from the primary with the p-bit turned on. It can then transmit one or more frames and relinquishes its access right when it turns the f-bit on in its last frame. (An exception to this procedure using the p/f-bit is the polling of secondary stations on a loop with the NSP command.) For the sequence control considerations discussed earlier, each station can have only up to seven unacknowledged I-frames outstanding. In a half-duplex link, to get a positive acknowledgment, the line must turn around. Therefore, the p- and f-bits also serve to turn the line around, thus alternating control of the line between primary and secondary. In a full-duplex link, the p- and f-bits serve only to assign and relinquish the access right of the secondary.

We have so far encountered one control message encoded in supervisory frames, REJ. There are two other control messages encoded in supervisory frames, which are used for flow control. They are RR (receive ready) and RNR (receive not ready). They can be sent by a receiver, either primary or secondary, to control the flow from the sender.

A dialogue may get out of synchronization when one or both stations become uncertain of the status of the other. Nonsynchronous conditions could occur because of lost frames, and frames received out of sequence. Timeouts and retries are the means for recovering synchronization. Some of the SDLC conditions for timeouts and retries are:

1. When the primary station transmits a frame with the p-bit turned on, response is expected within a certain time. The absence of a response may be due to no transmission from the secondary ("idle detect" condition) or garbled transmissions from the secondary ("nonproductive receive" condition). Timeouts are needed for both conditions to initiate recovery action at the primary. We note that a secondary cannot retry if there is no acknowledgment to its last frame with the f-bit turned on, since it has just given up its access right. In a loop, however, the secondary can retry following an NSP polling command.
2. Retries are made to obtain acknowledgment of a command. For example, NSA is the expected acknowledgment for SNRM, DISC, and SIM.
3. Retries are made to resume communication with a busy station following reception of RNR. Retries may be attempted by a primary or a secondary station.
4. Retries are made to achieve initial on-line status at a secondary station (ROL response).

5. Retries are made to initiate active communication at a secondary station (RQI response).
6. For a switched data link, an inactivity timeout of 20 seconds is used to alert stations of link disuse. If the timeout expires at either station, that station may attempt to alert the other station. After a user-specified number of unsuccessful attempts, the station disconnects the switched circuit.

If a nonsynchronous condition cannot be recovered by the specified number of retries, help from protocols at a level above data link control will be necessary. The type of intervention required depends on the decision-making power available beyond that of data link control. At a terminal, for example, operator intervention may be needed. Some other conditions also require help from protocols at a level above data link control:

1. If a primary station transmits a command that is not valid for the receiving secondary station, the secondary responds with a CMDR frame. The I-field of the frame contains secondary station status data that the primary needs for appropriate recovery action. Intervention from a higher level is required to analyze and act on the secondary status report. The secondary then expects a *mode-setting command from the primary station*.
2. If a secondary station's response, XID, to the exchange of station identification contains the wrong identification, intervention from a higher level is required to analyze and act on the situation.

Additional SDLC Features

1. Apart from I-frames, SDLC provides another vehicle for either primary or secondary to send data to each other. These are the NSI frames of the nonsequenced format. NSI frames are not sequenced nor are they acknowledged at the DLC level.
2. A transmitter, primary or secondary, may abort the transmission of a frame by sending eight consecutive binary 1's with no zero insertion. This abort pattern terminates a frame without an FCS field or ending flag. The abort pattern may be followed by seven additional binary 1's, which will idle the data link, or it may be followed by a flag. An aborting secondary station may not start another frame until it receives a command from the primary. In the case of a loop, a loop secondary station may abort by first transmitting the abort pattern or by simply resuming the repeater function (thus propagating continuous 1's).

Comments on the Structure of SDLC

We found that the SDLC protocol can be decomposed to fit into the hierarchical functional levels in Fig. 3.2 very well with one obvious exception. That is the shared use of the p/f-bit by different levels. At the multiple-access level, the p/f-bit is used to effect the sharing of a data link by multiple secondary stations. At the dialogue

level, the p/f-bit is used to effect the NRM dialogue interactions. The p/f-bit may also be used to provide a check point for error recovery action.

3.4 THE EVOLUTION OF DLC PROTOCOLS

In this chapter we have examined the main classes of data link control protocols in the chronological order of their development; first asynchronous or start-stop protocols, then synchronous character-oriented protocols, and finally synchronous bit-oriented protocols. Generally, synchronous protocols have a lot more functional capability than asynchronous protocols and are intended for computers and terminal devices that are intelligent and have memory for buffering relatively long messages.

Start-stop protocols were originally designed for nonintelligent unbuffered terminal devices. A significant percentage of terminals in use today are still of this type and start-stop protocols are still the only protocols that they are capable of using. However, both technological and marketing trends are pointing toward the use of more and more intelligent buffered terminal devices. Furthermore, the development of packet-switching networks poses new DLC requirements for communication between computers as equals (in addition to the traditional master-slave mode of operation). Bit-oriented synchronous protocols have been developed to meet these requirements and to take advantage of hardware capabilities not heretofore available. The importance of such protocols is emphasized by the emergence of the protocol standards.

In this section we first provide a critical review of the different classes of protocols described earlier. We then summarize functional capabilities in HDLC and ADCCP that are not available in the current version of SDLC described above.

Recall that communication has been defined herein as synchronization of data between different machines. We found that the synchronization problem can be structured into a hierarchy of functional levels, each level in the hierarchy depending on synchronism at the level below achieved over finer grains of time. The synchronization functional levels in the hierarchy are (in decreasing granularity of time):

- Dialogue synchronization
- Content synchronization
- Multiple access synchronization
- Frame synchronization
- Bit synchronization

The most basic requirements are bit and frame synchronization, so that bits of data and control information can be transported as a package (frame) from one machine to another.

Start-stop protocols generally rely on a character-code set which defines data characters and some control characters. Both control and data are transmitted one character at a time. The start bit and stop interval accomplish both bit and character synchronization. The latter can be regarded as the equivalent of frame synchroniza-

tion since characters are transmitted one at a time. Start-stop protocols have limited functions in the higher levels of the hierarchy and are typically specific to terminal device types. There has been a proliferation of start-stop protocols designed for different code sets and different terminal devices, resulting in a serious compatibility problem. Nonintelligent terminals often have to be remotely controlled by the computers they are communicating with. As a result, device control in addition to data link control characters and sequences are intermixed with data in dialogue exchanges. Data link configurations are typically point-to-point half-duplex. Not only is there little sharing of communications facilities, but often an operator needs multiple terminals to interact with different application programs even though they reside within the same host computer. Finally, the irreducible overhead of a start bit and a stop interval associated with the transmission of every character is another decided disadvantage. These are some of the reasons that prompted the development of synchronous protocols for communication between computers and sophisticated data terminals or other computers.

Synchronous protocols, both character- and bit-oriented, avoid many of the deficiencies of asynchronous protocols, and provide essentially the complete hierarchy of functions identified earlier. Character-oriented protocols, such as BSC described earlier, still depend on a character-code set. Some of the frequent criticisms of BSC (and similar character-oriented protocols) are:

1. The character-code set contains both data link control and device control characters. There is no clear separation between data link control, device control, and various high-level source-destination control functions, which need to be decoded at the data link control level.
2. Control frames with control and address information are not protected by error checking. (Some format checking is done using pad characters as a rudimentary form of error detection. For example, the control characters EOT and NAK must be followed by a trailing pad character of all 1 bits. This is done to reduce the probability of a transmission line error converting a positive acknowledgment response into an EOT or NAK response.)
3. New control messages need to be defined as sequences of existing control characters. Upward compatibility is difficult as new functions are defined.
4. Also as a result of using a character-code set, transparent data are handled as a special case, using the transparent text mode described earlier.
5. Communication is always half-duplex even if a full-duplex communications line is used. Only one unacknowledged data frame can be outstanding at a time; each such data frame is individually acknowledged. As a result, frequent turnarounds of the communications line are necessary for data transfer, giving rise to inefficient utilization of the communications line, especially when the application calls for short transmission blocks.
6. A transmitting station, with control of the line, can continue to transmit indefinitely, thus monopolizing it; the receiving station can send only control messages in reply. Some partial remedies, however, are provided; such as the limited conversational mode and the reverse interrupt (RVI) control message.

Bit-oriented protocols, such as SDLC described above, provide improvements over character-oriented protocols with more functional capabilities and better solution techniques to the synchronization problems. The positional significance approach provides not only separation of control and data but also facilitates a clean separation of DLC from high-level communications protocol layers. This is consistent with the layering approach in communications network architecture, in which each protocol layer is associated with a positionally significant header and trailer of a frame [ZIMM 80]. Data link control, being the lowest protocol layer⁵ in the communications network architecture, provides the outermost header and trailer of a frame. High-level functions (such as device control and session control) are encoded in headers and trailers which are treated as part of the "information field" of a frame by data link control protocols and are thus invisible to data link control facilities.

With positionally significant control and data fields within a frame, growth in functional requirements can be accommodated by expanding the size of the control field. (See the discussion of HDLC and ADCCP below.) Transparent data are also handled as a rule rather than an exception. All control and data information are contained within frames all of which are error checked.

Communication between SDLC stations can be full or half duplex. SDLC link configurations consist of loops, in addition to point-to-point and multipoint lines (available with BSC). Up to seven information frames can be transmitted before a positive acknowledgment is required. The considerations described above all contribute toward more efficient utilization of the communications facilities than do start-stop and BSC techniques. In a half-duplex configuration, the maximum of seven outstanding unacknowledged information frames also prevents a station from monopolizing the communications line.

Additional Functional Capabilities in HDLC and ADCCP [ANSI 79, ISO 79, CYPS 78]

SDLC, as described above, and the protocol standards HDLC and ADCCP are compatible protocols with the current version of SDLC [IBM 75] having only a subset of the repertoire of commands and capabilities adopted for HDLC and ADCCP. We discuss some of these extended capabilities in this section.⁶ First we note that some of the control messages are called by different names in the protocol standards. In particular, the terminology "unnumbered" is used throughout in place of "nonsequenced"; thus NSI becomes UI (unnumbered information), NSA becomes UA (unnumbered acknowledgment), and NSP becomes UP (unnumbered poll). Also, the term FRMR (frame reject) is used instead of the SDLC term of CMDR (command reject).

HDLC and ADCCP provide for the use of an extended control field of 16 bits. Additional bits are available in supervisory and nonsequenced formats for encoding

⁵According to the OSI Reference Model [ZIMM 80], physical control, which is concerned with electrical characteristics, is actually the lowest functional layer.

⁶Since many of these extended capabilities were proposed by IBM to the standards organizations, it seems reasonable to expect that SDLC will be upgraded in the future to include them.

new control messages. Seven-bit fields are used for N_S and N_F counts, thus permitting the transmission of up to 127 information frames before positive acknowledgment. This is desirable for efficient utilization of very high speed communications channels and satellite links which have a large propagation delay (since the mean number of frames that can be enroute from a sender to a receiver is equal to the throughput rate times the average delay from the sender to the receiver). HDLC and ADCCP also include provisions for an extended address field.

An additional supervisory frame, SREJ (selective reject), is defined in the protocol standards. SREJ is a negative acknowledgment that indicates rejection of the information frame with the sequence number contained in the N_F field (but acceptance of frames numbered up to and including $N_F - 1$). However, unlike REJ, SREJ demands retransmission of only the specific information frame numbered N_F , and not subsequent frames that have also been sent. We note that this provision may alter the sequence ordering of information frames delivered from the transmitting station to the receiving station. If the original sequence of the information frames needs to be reestablished, it will have to be performed by protocols at a level higher than that of DLC. We note that in packet-switching networks, the frames exchanged by two packet-switching nodes typically belong to different "virtual circuits" and preservation of ordering is not important at the data link control level.

In the current version of SDLC, data transfer can take place only with the secondary station in the normal response mode. HDLC and ADCCP provide two additional secondary station modes for the transfer of information frames. An SARM (set asynchronous response mode) command places a secondary station in a mode in which it can initiate transmissions without having been polled first; otherwise the primary-secondary relationship between the stations is essentially the same as in NRM. The SABM (set asynchronous balanced mode) command is used to make stations into equals with identical command and response capabilities. The stations are called *combined stations* rather than primary and secondary stations. Each combined station is, in effect, a combination of a primary and a secondary station. The asynchronous balanced mode is intended primarily for point-to-point full-duplex links; note that in a packet network environment, a true full-duplex communications capability is important.

High-Level Communications Protocols

We have attempted to present data link control protocols as a structured set of functions. The hierarchical model introduced herein provides a structured representation of functional requirements for remote processes to communicate. Thus it can be used to model high-level communications protocol layers as well. In most network architectures (see, e.g., [ZIMM 80]) data link control constitutes just the lowest layer of a larger hierarchy of communications protocol layers. If we examine the functions in the hierarchy, we find that those functions within the high-level protocol layers can also be structured into functions for multiple-access synchronization, content synchronization, and dialogue synchronization, similar to what we have encountered at the data link control layer.

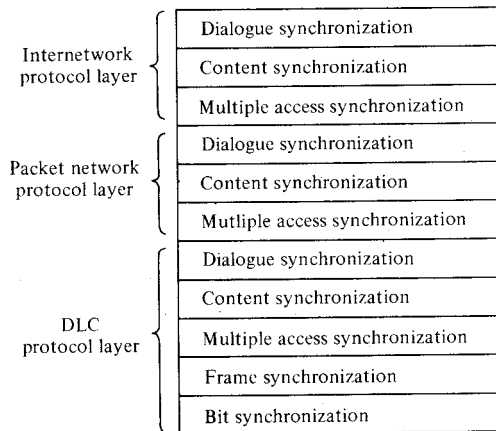


Figure 3.10. Communications protocol layers and their functional requirements.

We illustrate this idea in Fig. 3.10 by considering a somewhat simplified communications architecture for internetworking with three protocol layers. Note that the functional levels of dialogue, content, and multiple-access synchronization are reiterated three times: once in the DLC protocol layer, then in the packet network protocol layer, and finally in the internetwork protocol layer. Although the functional requirements are the same in the three protocol layers, the end points and the characteristics of the communications channel (virtual or real) assumed by different protocol layers are different. Consequently, the degrees of importance associated with various synchronization functions will not be the same in different protocol layers and the solution techniques will thus be different. Consider the end-to-end transport protocol layer of a packet-switching network. The multiple-access synchronization functional level again deals with the sharing of communications facilities by multiple sender-receiver pairs. But with a mesh topology and store-and-forward nodes, the multiple-access synchronization problems are: routing and the synchronization of routing information, congestion control, and so on, which are not the same as before for DLC. The dialogue and content synchronization levels still deal with flow control, error and sequence control, and so on, as described earlier for DLC. The specific protocols for solving these problems, however, are somewhat different because the characteristics of a packet network are different from those of a data link (at a lower level in the network). For example, flow control is probably more important for dialogue synchronization at the network level than at the DLC level, while error control is probably more important at the DLC level than at the network level (because the network transport protocols can count on reliable data links provided by DLC at the lower level). Finally, when different packet-switching networks are interconnected via gateways, it is not hard to see that the multiple access synchronization, content synchronization, and dialogue synchronization functions are needed again in internetworking protocols [CERF 78] for the network of gateways.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under Grant ECS78-01803. The author would like to thank Wushow Chou of North Carolina State University and Vint Cerf of the Defense Advanced Research Projects Agency for their helpful comments. He is particularly indebted to an anonymous reviewer who painstakingly read an early draft, pointed out many errors, and made numerous valuable suggestions for improvement.

REFERENCES

- ANSI 75 American National Standards Institute, "Procedures for the Use of the Communication Control Characters of American National Standard Code for Information Interchange in Specified Data Communication Links," ANSI X3.28-1976, December 1975.
- ANSI 79 American National Standards Institute, "American National Standard for Advanced Data Communication Control Procedures (ADCCP)," ANSI X3.66-1979, January 1979.
- BURT 72 Burton, H. O., and D. D. Sullivan, "Errors and Error Control," *Proc. IEEE*, November 1972.
- CERF 78 Cerf, V. G., and P. T. Kirstein, "Issues in Packet-Network Interconnection," *Proc. IEEE*, Vol. 66, November 1978.
- CYPS 78 Cypser, R. J., *Communications Architecture for Distributed Systems*, Addison-Wesley, Reading, Mass., 1978.
- DAVE 72 Davey, J. R., "Modems," *Proc. IEEE*, November 1972.
- DONN 74 Donnan, R. A., and J. R. Kersey, "Synchronous Data Link Control: A Perspective," *IBM Syst. J.*, May 1974.
- IBM 70 IBM Corp., *General Information — Binary Synchronous Communications*, Manual No. GA27-3004-2, 3rd. ed., October 1970.
- IBM 75 IBM Corp., *Synchronous Data Link Control — General Information*, Manual No. GA27-3093-1, May 1975.
- ISO 79 International Standards Organization, "Data Communication — High-Level Data Link Control Procedures — Frame Structure," Ref. No. ISO 3309-1979; "Data Communication — High-Level Data Link Control Procedures — Elements of Procedures," Ref. No. ISO 4335-1979, 1979.
- MART 70 Martin, J., *Teleprocessing Network Organization*, Prentice-Hall, Englewood Cliffs, N.J., 1970.
- MCNA 77 McNamara, J. E., *Technical Aspects of Data Communication*, Digital Equipment Corp., Maynard, Mass., 1977.
- ZIMM 80 Zimmermann, H., "OSI Reference Model — The ISO Model of Architecture for Open Systems Interconnection," *IEEE Trans. Commun.*, April 1980.