DATA LINK CONTROL PROTOCOLS*

Simon S. Lam

TR-124 January 1980

Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712

* To appear as a chapter in Computer Communication: State of the Art and Direction for the Future, W. Chou (Editor), Prentice-Hall. This work was supported by National Science Foundation under grant No. ENG 78-01803.

ABSTRACT

Data link control functional requirements are presented as a structured set of synchronization functions (to be referred to as the hierarchical model). The three main classes of data link control protocols (asynchronous, character-oriented synchronous and bit-oriented synchronous) are introduced and compared. Using the hierarchical model as a framework, the BSC and SDLC protocols are presented to illustrate basic principles of the two classes of synchronous protocols. Additional capabilities available with the bit-oriented protocol standards, HDLC and ADCCP, are also described.

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 paper 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 upon 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 equipments (DTE) in the protocols 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 etc. Together, a DTE and its transceiver, will be referred to as a station; see Figure 1 for illustration. A station is both a sender and a receiver of data messages (though not necessarily at the same time).

Although more than 2 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 Figure 2.) Each level in the hierarchy depends upon 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 Figure 2 are defined in Section 2 and are discussed in detail in Section 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 Section 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 paper 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.

2. SYNCHRONIZATION FUNCTIONS

We shall proceed from the bottom of the hierarchy shown in Figure 2 upwards.

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 [LAM 79] 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; (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 2 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. Generally, dialogue synchronization functions are 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 its receiver is in a state that can accept a new frame; (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 definition -- 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 the course of a dialogue, each station is in one of several states. Finite-state machines have been used to model dialogue level protocols. (For a rigorous treatment of this subject, see [BOCH 78, ZAFI 78, WEST 78].) Theoretically, if these protocols have been correctly designed, nonsynchronous conditions at this functional level will occur only as a result of nonsynchronous conditions at a lower functional level (e.g. messages being lost or out-of-sequence at the content synchronization level). In practice, rigorous verification of realistic protocols is difficult

and nonsynchronous conditions may also result from improperly designed dialogue level protocols.

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. 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 1 character (5-8 bits). In a synchronous transmission system, bit synchronism is maintained at all times. DLC protocols normally used with 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.

3.1 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 upon 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 Figure 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 "16X 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 develop 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 upon 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 Section 2. Second, the technological trend is towards more intelligent terminals which can accommodate synchronous protocols having more sophisticated functions.

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 above overhead 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 close 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 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 paper. 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 above names, is the technique for accomplishing frame synchronization. But as we shall see, they differ in the 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 Section 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 draft standard of ISO (International Standards Organization) and IBM's SDLC protocol. In Section 3.2.2, we shall describe the current version of SDLC as documented in [IBM 75] to illustrate this class of protocols.* 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, that are not present in the current version of SDLC [IBM 75]. These functions are discussed in Section 4.

---

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

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 concepts may be found in [MCNA 77].

3.2.1 DESCRIPTION OF A CHARACTER-ORIENTED PROTOCOL--BSC [IBM 70]

Suppose that bit synchronism is maintained between a receiver and transmitter. Since both data and control must be encoded and transported in packages of bits called frames*, the synchronization problem at the next level involves the following: (1) upon reception of a sequence of bits, to determine which bit is the beginning and which bit is the end of a frame; (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 eight bits per character for 256 assignments, USASCII with seven bits plus a parity bit per character for 128 assignments and six-bit Transcode with 64 assignments. As an example the EBCDIC assignments are illustrated in Figure 4.

Frame Synchronization

With a character code set, BSC data and control characters are represented by unique bit sequences. (We shall 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. Due to 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 shall use the shorthand notation

---

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

of Ø 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 (six or eight bits depending on the code set) will form the first character of the frame**. 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, then 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, then 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 Ø. (This bit synchronization pattern can be the hex '55' '55' or the SYN SYN SYN SYN pattern which provides for the 16 transitions necessary for bit synchronization.) Bit synchronism will be assumed from now on.

To ensure that the first and last character 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 afterwards. (The leading pad character may consist of alternating 0 and 1 bits (hex '55') or a SYN character.

**Note that this procedure is subject to errors. First the Ø sequence may be missed because of bit errors in it. On the other hand, the 2-character bit pattern of Ø is not unique and may appear in the middle of certain 3-character sequences.

The trailing pad character consists of all one bits (hex 'FF').) With the above 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 Ø [heading and/or data] BCC BCC PAD

PAD Ø [control] PAD

The exact nature of the heading, control and data fields in the above frame formats will be clarified in our discussions below on multiple access synchronization and content synchronization. For simplicity, we shall 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 errors within transmitted frames. Depending upon 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

VRC or LRC can only detect single errors. CRC is a much more powerful technique for error detection and can detect 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.

With Cyclic Redundancy Checking, two BCC characters are generated at the sender using the bit sequence in a frame to be checked and a generator bit sequence (polynomial). The two BCC characters are appended to the end of the frame. At the receiver, two BCC characters are generated by the same algorithm and they are compared with the BCC characters received.

A discrepancy indicates an error in the received frame bit sequence. For a detailed discription of LRC, VRC and the CRC coding algorithm, the reader is referred to [MART 70]. Note from the above frame formats that BSC control messages are not protected by CRC error checking.

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 upon the communications link configuration. BSC permits two types of communications link configurations between stations: point-to-point and multipoint. We shall first consider 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 ACKO 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 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 Ø EOT. For three seconds the other station can bid for the line without competition. After three seconds, both stations may bid for control of the line.

BSC also permits a multipoint communications link configuration (see Figure 5), with a control station and many tributary stations sharing the same line (which is effectively a broadcast medium). 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 only take place in 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; (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 Ø 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:

Ø EOT PAD Ø [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

a. a frame containing data

b. Ø EOT negative reply when the station has nothing to send

c. 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

a. Ø ACKO affirmative, ready to receive

b. Ø NAK negative, not ready to receive

c. 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 correctly interpret 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, receiver and sender must also agree upon 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
ACKO/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 ACKO, 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, ACKO 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 nor 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

Ø [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 to denote a control function. In particular, the transparent text mode is initiated by DLE STX and is terminated by either 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 non-transparent mode of operation if that intermediate block is not the first one.) Thus a frame in a transparent text mode may look like

Ø 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. In all cases, the inserted DLE characters need to be stripped off at the receiver.

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 Ø ACKO or Ø ACK1 indicating "data accepted, ready for more," or Ø 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 at the next higher level, dialogue synchronization, 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 Ø ENQ. The receiving station repeats the previous response: NAK, ACKO or ACK1 as appropriate.

Two positive acknowledgment responses are used for the following reason. ACKO is used as an affirmative response to the transmitting station's initial ENQ and for affirmative response to all even-number transmission blocks. ACK1 is used for all odd-number transmission blocks. The odd or even property of a block does not change even when it is retransmitted. As a result the alternating affirmative replies are a means of detecting 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 always preserved at the receiver.

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 a line turn around 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

- Ø ENQ Who are you?
- Ø α...ENQ I am α... . Who are you?

The called station replies positively with

- Ø α...ACKO

Where the called station identity α...is optional. The called station replys negatively with

- Ø NAK
- Ø WACK

The identity sequences α...may be 2 to 15 characters long. The minimum 2-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 following disconnect message

Ø DLE EOT

A call may also be disconnected by the disconnect time-out 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 two seconds to avoid the three second receive time-out 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 interupt)

RVI is a positive acknowledgment used in place of ACKO or ACK1 (depending upon the current odd-even count). RVI is transmitted by a receiving station to request termination or the current transmit-receive relationship. The transmitting station treats RVI as a positive acknowledgment (0 or 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 one-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 three-second time out 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 two-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 Figure 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 results in interfaces which are somewhat awkward at places and far from being minimal. We shall illustrate this observation by pointing out some strong dependencies between the BSC functional levels described above.

1. The half-duplex mode of operation facilitates the logical design of dialogue interactions between two communicating stations. Its effect, however, 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 (for flow control purposes) 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.

3.2.2 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 Figure 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 (which may or may not be data independent) or the DTE. In either case, our description of data link control will only be concerned 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 Fs 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 following any succession of five consecutive ones inside a frame following the beginning F. The receiver removes a zero that follows a received sequence of five continuous ones. We shall see below that with unique flags, the transmission of transparent data is always possible rather than handled as a special case.

The content 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 preceeding 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 zeroes are not protected by FCS. At the receiver, the deletion of inserted zeroes 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 one 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 ones 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 non-switched
2. Point-to-point duplex line, non-switched
3. Multipoint duplex line, non-switched
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 present, cannot communicate with each other. We remind the reader that sometimes a station has the capability to implement half-duplex protocols only 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 to consider now the format 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 as shown in Figure 6. For multiple access synchronization we are presently only interested in the address field and the p/f bit in the control field.*

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

---

*We use lower case letters p and f to avoid confusion with F which denotes a flag.

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 Figure 6.) If the frame is from secondary to primary, then the A field identifies the sender. If the frame is from primary to secondary, then 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 1s 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 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 discussion of 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. Signals 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 Figure 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 (non-sequenced 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 then sends a binary zero followed by continuous ones 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.

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 ones. The first down-loop secondary station that has been polled receives the go-ahead when it has counted a zero and seven

consecutive ones in its input signal. It repeats the first six ones 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 ones now creates a go-ahead pattern again. The access right to the shared channel is then passed on to the next downstream secondary station who has been polled. Finally, when the primary station receives seven or more continuous ones, 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 upon 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 Figure 8, where bit 7 differentiates between I frames and the others; bit 6 differentiates

between S 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_r and N_s are for error and sequence control of I frames. The remaining bits (marked * and ** in Figure 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 Figure 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 only subject 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 currently defined SDLC control messages is shown in Figure 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 eight-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 shall next talk 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 always 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 three-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 $N_s(1)$ and $N_r(1)$ at station 1, $N_s(2)$ and $N_r(2)$ at station 2. Suppose an I frame is sent from station 1 to station 2. It contains $N_s(1)$ which is the sequence number of that particular I frame. At station 2, if an error is detected in the frame, that frame is simply rejected. Suppose it is error-free. $N_s(1)$ is then compared to $N_r(2)$ which is the sequence number expected by station 2. If they agree, the frame is accepted and $N_r(2)$ is incremented by 1. If $N_s(1) \neq N_r(2)$, the received frame is rejected. (There are two possible cases. $N_s(1) < N_r(2)$ means that the received frame is a duplicate. On the other hand $N_s(1) > N_r(2)$ means that the received frame

is out of sequence and some I frames are missing. Since $N_s(1)$ and $N_s(2)$ 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 a positive acknowledgment from station 1 to station 2 confirming correct reception of previous 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 always 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 above discussion, 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.

7. To guard against errors in the beginning and ending flags of a frame, additional transformations are defined for the generation of the FCS bits. (See [IBM 75].)

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. The primary station has ultimate responsibility for the dialogue. 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 is a secondary on another data link.

The responses of a secondary depend upon 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 procedures 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 N_r and N_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 sequence control considerations dis-

cussed 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 around the line, thus alternating control of the line between and 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.

The above is a description of the behavior of a synchronized dialogue. A dialogue may get out of synchronization when one, or both, stations become uncertain of the status of the other, or the combined state is not predefined. Nonsynchronous conditions could occur if there were flaws in the design of the dialogue protocols, or because of nonsynchronous conditions at a lower level. Some examples of the latter are lost frames, and frames received out of sequence, which have not been recovered at the content synchronization level. Timeouts and retries are the means for recovering synchronization. Retries may be made by the primary or secondary station under various non-synchronous conditions. In general, the primary has the final responsibility for recovery or, if after a number of retries without success, for reporting to a higher level protocol. With the primary station having the final responsibility, deadlocks as a result of mutual waits between primary and secondary are avoided.

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 secondary station.
4. Retries are made to achieve initial online 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 upon 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 upon 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 upon 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 ones 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 ones, 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 ones).

Comments on the Structure of SDLC

We found that the SDLC protocol can be decomposed to fit into the hierarchical functional levels in Figure 2 very well with one obvious exception. That is the shared use of the p/f bit by the multiple access and dialogue 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 and to provide a check point for error recovery action.

4. THE EVOLUTION OF PROTOCOLS

We have examined in this paper 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 which 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 towards the use of more and more intelligent buffered terminal devices. Furthermore, the development of packet switching networks pose 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 underscored by the emergence of the protocol standards.

In this section we shall first provide a critical review of the different classes of protocols described earlier. We shall then summarize the new functional capabilities proposed for HDLC and ADCCP, which 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 upon 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 upon 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 synchronization 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 that 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, often times, 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 and 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 upon 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 is 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 only send

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 [ISO 78]. Data link control being the lowest protocol layer* 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 discussion of HDLC and ADCCP below.) Transparent data is 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

---

*According to [ISO 78], physical control, which is concerned with electrical characteristics, is actually the lowest functional layer.

lines (available with BSC). Up to seven information frames can be transmitted before a positive acknowledgment is required. The above considerations all contribute towards more efficient utilization of the communications facilities than 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 76, 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 proposed for HDLC and ADCCP. We shall discuss some of these extended capabilities in this section.* 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 new control messages. Seven-bit fields are used for N_s and N_r counts thus permitting the transmission of up to 128 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 number of frames that can be enroute from a sender to a receiver is equal to the throughput rate times the average delay

*Since 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.

from the sender to the reciever). 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_r field (but acceptance of frames numbered up to and including N_r-1.) However, unlike REJ, SREJ demands retransmission of only the specific information frame numbered N_r, 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. However 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 only take place 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; they have identical command and response abilities. The stations are called combined stations rather than primary and secondary stations. Each station on a point-to-point link would

have equal ability to perform error recovery, including resetting the mode of operation. Both the asynchronous response mode and asynchronous balanced mode are 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, for example, [ISO 78]) 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.

We illustrate the above idea in Figure 10 by considering a somewhat simplified communications architecture for internetworking with 3 protocol layers. Note that the functional levels of dialogue, content and multiple access synchronization are reiterated three times; once in the DLC protocol layer and 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 mutliple 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, etc., which are not the same as before for DLC. The dialogue and content synchronization levels still deal with flow control, error and sequence control, etc., as we have 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 National Science Foundation under Grant No. ENG78-01803. The author would like to thank Dr. Wushow Chou of North Carolina State University and Dr. Vint Cerf of the Defense Advanced Research Projects Agency for their many helpful comments on this paper. 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 76 American National Standards Institute, "Proposed American National Standard for Advanced Data Communication Control Procedures," Document X3S34/589 (Sixth Draft), October 1976.

BOCH 78 Bochman, G. V. , "Finite State Description of Communication Protocols," Computer Networks, Vol. 2, 1978.

BURT 72 Burton, H. O. and D. D. Sullivan, "Errors and Error Control," Proc. of the IEEE, Nov. 1972.

CYPS 78 Cypser, R. J., Communications Architecture for Distributed Systems, Addison-Wesley, 1978.

CERF 78 Cerf, V. G. and P. T. Kirstein, "Issues in Packet-Network Interconnection," Proc. of the IEEE, Vol. 66, Nov. 1978.

DONN 74 Donnan, R. A. and J. R. Kersey, "Synchronous data link control: A perspective," IBM Systems Journal, May 1974.

DAVE 72 Davey, J. R., "Modems," Proc. of the IEEE, Nov. 1972.

IBM 70 IBM Corp., General Information - Binary Synchronous Communications, Manual No. GA27-3004-2, Third Edition, Oct. 1970.

IBM 75 IBM Corp., Synchronous Data Link Control General Information, Manual No. GA27-3093-1, May 1975.

ISO 78 International Standards Organization, "Provisional Model of Oper- Systems Architecture," ISO/TC97/SC16 N34; reprinted in Computer Communication Review, July 1978.

LAM 79 Lam, S. S., "Multiple Access Protocols," Technical Report TR-88, Department of Computer Sciences, University of Texas at Austin, Jan. 1979.

MART 70 Martin, J., Teleprocessing Network Organization, Prentice-Hall, 1970.

MCNA 77 McNamara, J. E., Technical Aspects of Data Communication, Digital Equipment Corp., 1977.

WEST 78 West, C. H., "An Automated Technique of Communication Protocol Validation," IEEE Trans. on Commun., August 1978.

ZAFI 78 Zafiropulo, P., "Protocol Validation by Duologue-Matrix Analysis," IEEE Trans. on Commun., August 1978.

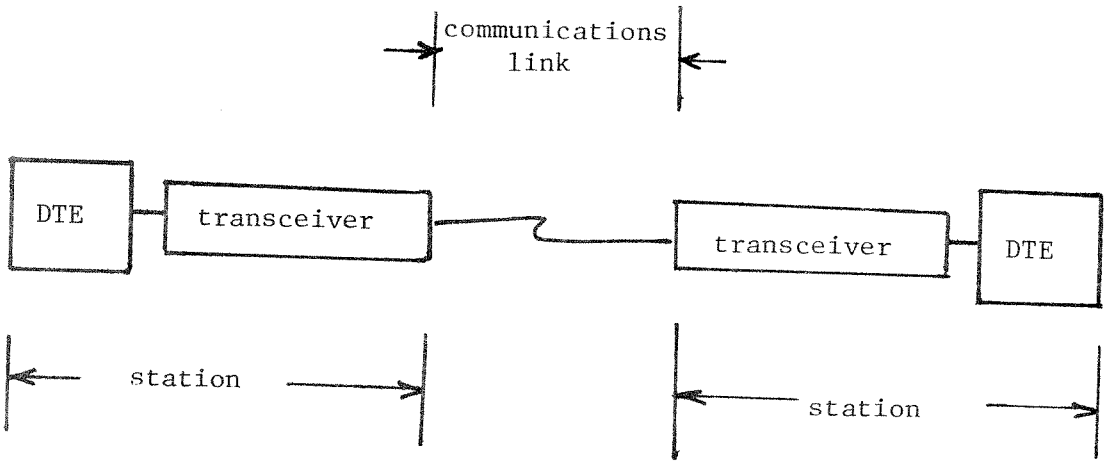

Fig. 1. Two communicating stations.

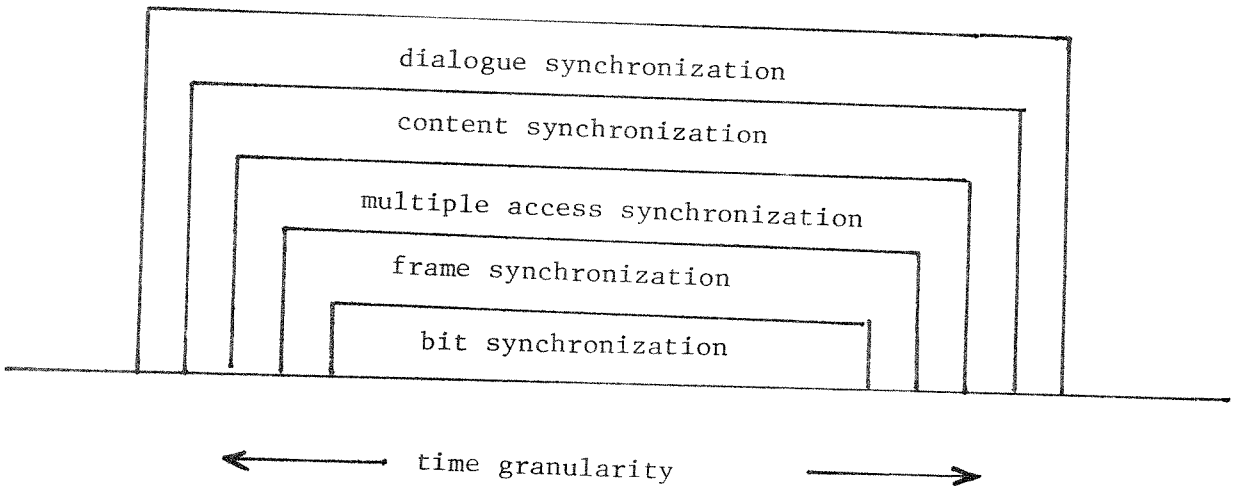

Fig. 2. A hierarchy of communication functions.

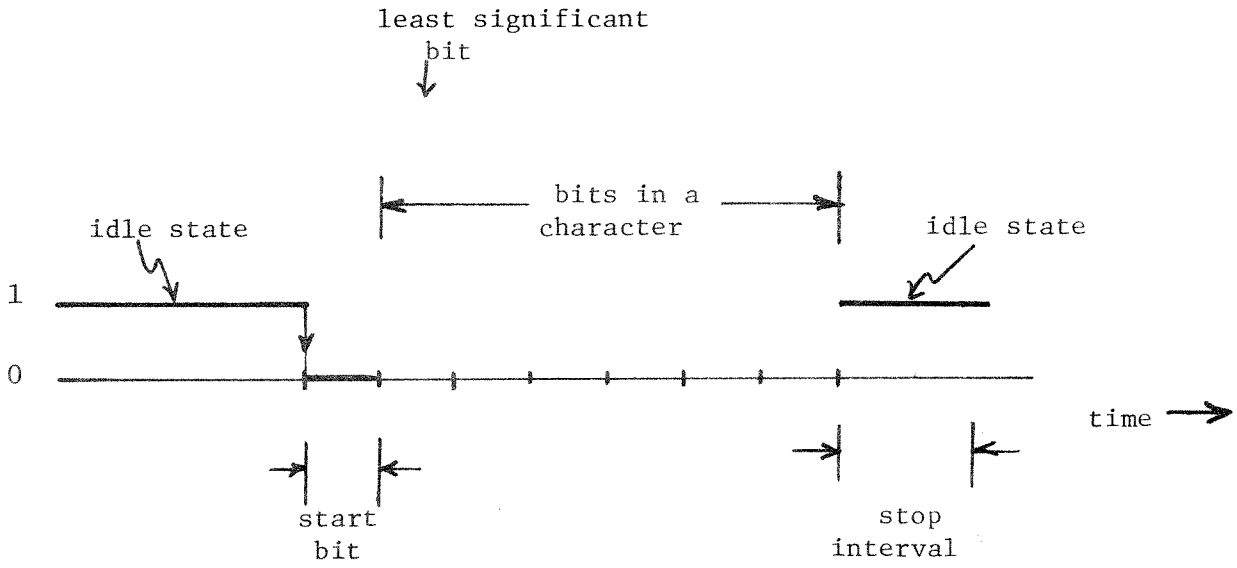

Fig. 3. Asynchronous transmission.

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

Fig. 4. EBCDIC character assignments.

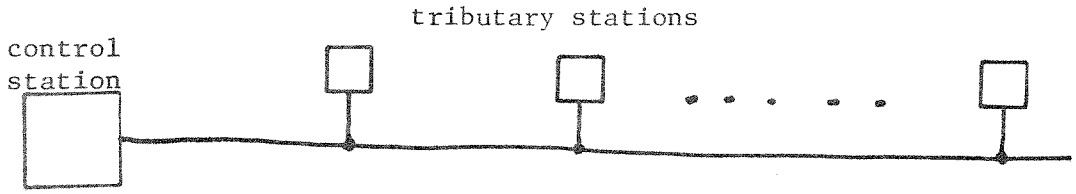

Fig. 5. Multipoint link configuration.

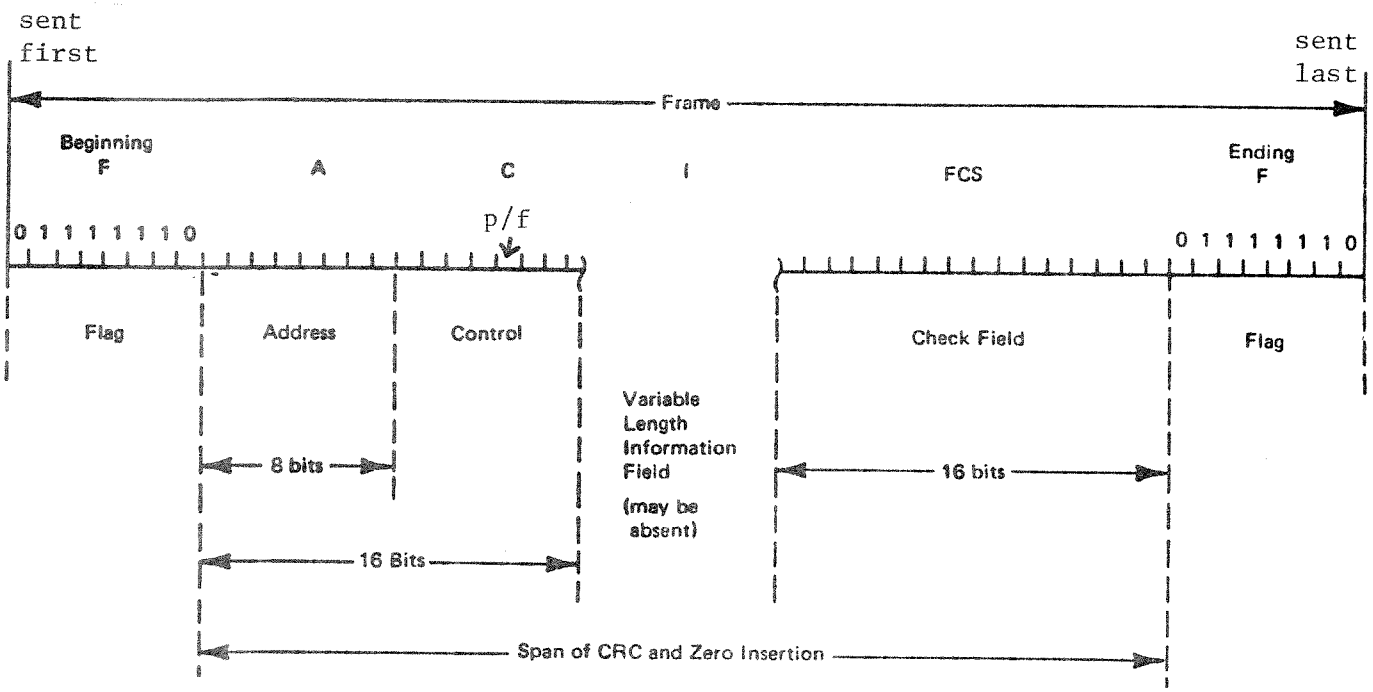

Fig. 6. Fields of an SDLC frame.

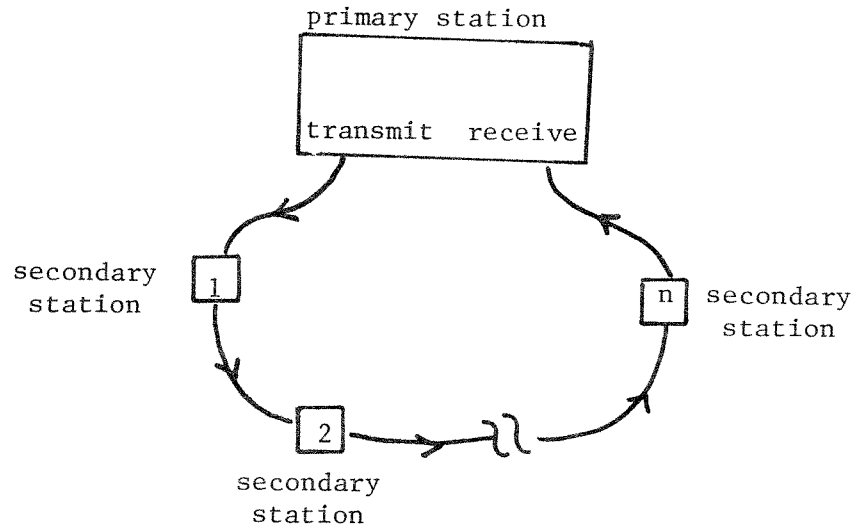

Fig. 7. Loop configuration.

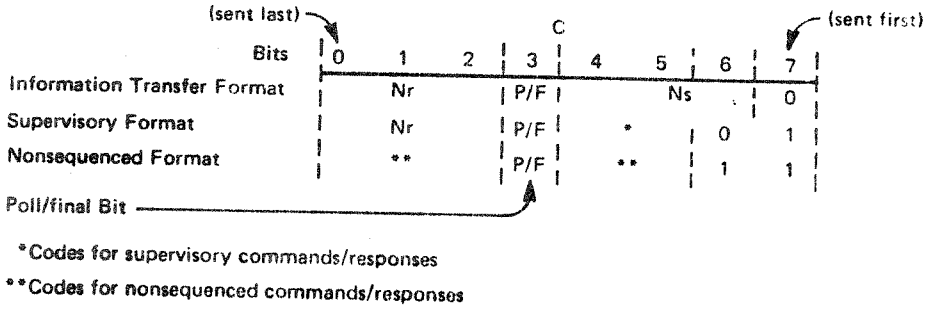

Fig. 8. SDLC control field formats.

Format (Note 1)	Sent Last ↓ Binary Configuration ↓ Sent First				Acronym	Command	Response	I-Field Prohibited	Resets Nr and Ns	Confirms frames through Nr-1	Defining Characteristics
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 offline.
	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				Non-valid 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.

Fig. 9. Summary of SDLC commands and responses.

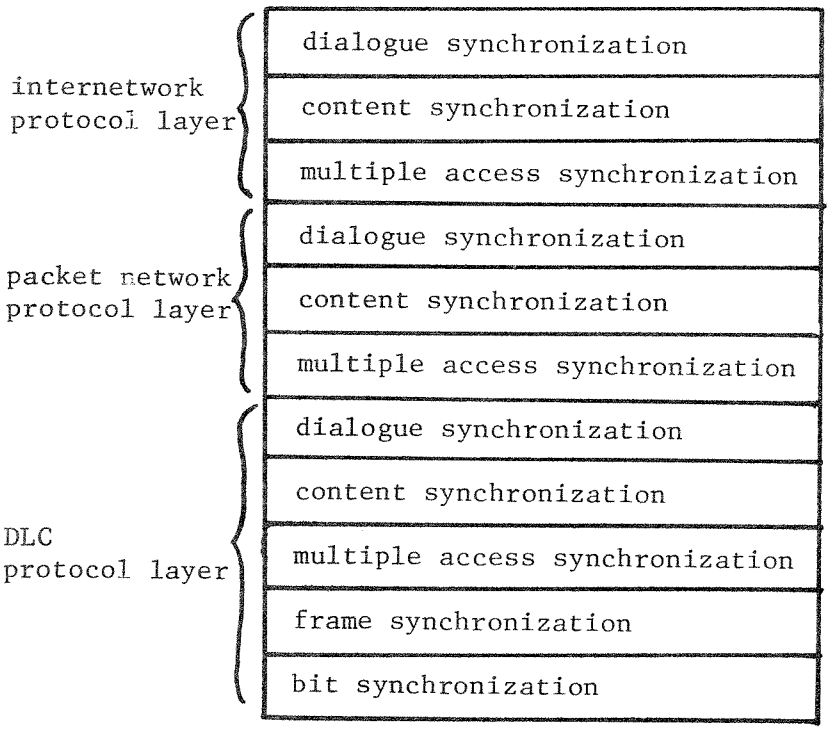

Fig. 10. Communications protocol layers and their functional requirements.