Chapter 2
Data Link Control Protocols

2.1. OVERVIEW

Data Link Control (DLC) protocols provide functions of frame, multiple access, content, and dialog synchronization for communication using a physical link. Except for start-stop protocols, bit synchronism is assumed to be provided separately by hardware devices (modems, transceivers, etc.) that implement the physical layer function.

DLC protocols fall into two main categories: character-oriented protocols and bit-oriented protocols. In character-oriented protocols, all control and data information are exchanged in the form of characters chosen from a character code set (e.g., ASCII and EBCDIC). A popular example is IBM's BSC [IBM 70] which is akin to the ANSI X3.28 protocol standard [ANSI 75]. Bit-oriented protocols do not depend upon the use of any character code set. Bit-oriented protocols include HDLC which is a standard of ISO [ISO 78, ISO 79a, ISO 79b, ISO 80a, ISO 80b], ADCCP which is a standard of ANSI [ANSI 79] and IBM's SDLC [IBM 75]. All three protocols are essentially the same with only minor differences among them. The two protocol standards evolved out of IBM's submissions to the respective standards organizations. Some additional functions have been defined for HDLC and ADCCP, which were not available in the version of SDLC described in [IBM 75].

In [LAM 83], reprinted below, a brief introduction to start-stop protocols is first given. The BSC and SDLC protocols are then described in some detail; additional functional capabilities in HDLC and ADCCP are summarized. (SDLC was described instead of one of the protocol standards because this article was first written in 1979 when the protocol standards documents were not generally available.) Various computer manufacturers have also proposed similar bit-oriented DLC protocols, including DDCMP of Digital Equipment Corporation. An excellent description of DDCMP may be found in [MCNA 77].

We give some additional comments on the functions of frame, multiple access, content, and dialog synchronization and how they are implemented in different protocols.

2.2. FRAME SYNCHRONIZATION

There are two basic techniques for identifying frame boundaries. Character-oriented protocols use special control characters to do so. However, sending transparent data in such protocols requires character insertion/deletion operations. Bit-oriented protocols use a special sequence of 8 bits, 01111110 (called a flag), for marking frame boundaries. To render flags unique, bit insertion/deletion operations have to be performed (by hardware). The reader is referred to [LAM 83] for more details. The DDCMP protocol is a bit-oriented protocol, but it uses a third approach. The beginning of a frame is marked by two or more SYN characters (like character-oriented protocols), while the end of a frame is indicated by a byte count in the frame header.

Most DLC protocols use the cyclic redundancy check (CRC) method for error detection, except for start-stop protocols and some implementations of character-oriented protocols. A description of algorithms for CRC code generation and error detection can be found in Chapter 13 of [MCNA 77]. Note that a modified version of these algorithms is specified for HDLC (see the appendix of [ISO 79a]). The modification is intended to detect flags missing from between two consecutive frames.

2.3. MULTIPLE ACCESS SYNCHRONIZATION

For both classes of protocols considered above, very simple protocols (such as polling) are used for multiple access, or no protocol is necessary because each sender has its own dedicated channel. Much more sophisticated protocols are employed when the data link is actually a local area network (see Chapter 4). The subject of multiple access protocols is covered in Chapter 3.

2.4. CONTENT AND DIALOG SYNCHRONIZATION

In character-oriented protocols, the separation of control and data information is done by the use of different subsets of characters in the character code set. In bit-oriented protocols, the separation is done by positional significance; i.e., they reside in different fields within a frame.

Error control is provided by an automatic repeat request (ARQ) technique in all protocols mentioned above. In an ARQ protocol, the sender keeps a copy of a transmitted data unit. The copy is discarded upon receipt of an acknowledgement from the receiver. If no acknowledgement has been received within a timeout period, the data unit is retransmitted. There are two main categories of ARQ protocols: (1) stop-and-wait protocols, in which only one data unit can be in transit and awaiting acknowledgement at any time and (2) window protocols, in which up to N data units can be in transit and be awaiting acknowledgement. Two retransmission strategies can be used for window protocols. First, when the timer of a specific data unit awaiting acknowledgement has expired, only that particular data unit is retransmitted (selective repeat strategy). Second, the data unit whose timer has expired and all data units transmitted subsequent to it are retransmitted (sometimes referred to as the go-back-N strategy). Both classes of ARQ protocols, if implemented correctly, can also provide in-sequence delivery of data units, given certain assumptions about the behavior of the communication channels (such as finite lifetimes for data units in transit).

Window protocols that permit multiple data units to be in transit are necessary for the efficient utilization of data links that have either relatively long transit delays or high transmission rates, or both [BURT 72].

The protocol standards manuals describe protocol architectures, not their implementations. They typically define precisely low-level functions such as error detection and frame synchronization, frame formats, and the encoding of control messages. The meanings of control messages are given only informal descriptions and are illustrated with example sequences of message exchanges. Such specifications leave many options to be decided by protocol implementors. In particular, one can choose from a variety of data link configurations and operational modes and from different subsets of control messages instead of the entire set defined. Thus it is not easy for independent implementations of such DLC protocols to be compatible. Even if they assume the same configuration and message subset, the exact procedures for implementing the functions of data transfer and connection management may be different.

Lastly, like any distributed program, it is very hard to verify a DLC protocol to be correct. The subject of protocol specification and verification is covered in Chapter 7. In [SHAN 83a], reprinted below, a version of the HDLC protocol with the Asynchronous Balanced Mode is shown. This particular protocol has been verified to have desirable correctness properties for both data transfer and connection management [SHAN 83b]. A simple version of the BSC protocol implementing data transfer and connection management functions can be found in [CHOW 83]. This protocol is guaranteed to have certain correctness properties (freedom from deadlocks and unspecified receptions, and boundedness) by the construction methodology employed therein.

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.

[CHOW 83] Chow, C. H., M. G. Gouda, and S. S. Lam, "A Discipline for Constructing Multi-Phase Communication Protocols," Technical Report TR-233, Dept. of Computer Sciences, University of Texas at Austin, June 1983 (revised, October 1983), submitted for publication.

[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 78] International Standards Organization, "Data Communication--HDLC Balanced Class of Procedures," Draft International Standard ISO/DIS 6256, 1978.

[ISO 79a] International Standards Organization, "Data Communication--High-level Data Link Control Procedures--Frame Structure," Ref. No. ISO 3309, Second Edition, 1979.

[ISO 79b] International Standards Organization, "Data Communications--HDLC Procedures--Elements of Procedures," Ref. No. ISO 4335, First Edition, 1979.

[ISO 80a] International Standards Organization, "Data Communication--High Level Data Link Control Procedures--Elements of Procedures (tertiary station) ADDENDUM 2," Draft Addendum ISO 4335/DAD2, 1980.

[ISO 80b] International Standards Organization, "Data Communication--HDLC Unbalanced Classes of Procedures," Ref. No. ISO 6159, First Edition, 1980.

*[LAM 83] Lam, S. S., "Data Link Control Procedures," Chapter 3 in *Computer Communications, Vol. 1: Principles.* W. Chou (ed.), Prentice-Hall, Englewood Cliffs, N.J., 1983.

[MCNA 77] McNamara, J. E., *Technical Aspects of Data Communication*, Digital Equipment Corp., Maynard, Mass., 1977.

*[SHAN 83a] Shankar, A. U. and S. S. Lam, "Specification and Verification of an HDLC Protocol with ARM Connection Management and Full-Duplex Data Transfer," ACM SIGCOMM '83 Symposium, University of Texas at Austin, March 1983; in *Computer Communication Review*, Vol. 13, No. 2, 1983, pp. 38-48.

[SHAN 83b] Shankar, A. U. and S. S. Lam, "An HDLC Protocol Specification and its Verification using Image Protocols., *ACM Trans. on Computer* Systems, Vol. 1, No. 4, November 1983, pp. 331-368.

(* article reprinted below.)