DATA LINK CONTROL

A LITTLE HEADS UP !

2 Main Functions of Data Link Layer

DATA LINK CONTROL (DLC)

DLC is the service provided by the Data Link layer of function defined in the Open Systems Interconnection (OSI) model for network communication. The Data Link layer is responsible for providing reliable data transfer across one physical link within the network.

Some of its primary functions include:

1. defining **frames**

2. performing error detection or ECC on those frames

3. performing **flow control** (to prevent a fast sender from overwhelming a slow receiver).

3

MEDIA ACCESS CONTROL (MAC)

Media access control (MAC) is a sublayer of the data link layer (DLL) in the seven-layer OSI network reference model. MAC is responsible for the transmission of data packets to and from the network-interface card, and to and from another remotely shared channel.

FRAMING

Framing

in the data link layer separates a message from one source to a destination, or from other messages to other destinations, by adding a sender address and a destination address. The destination address defines where the packet is to go; the sender address helps the recipient acknowledge the receipt.

Fixed-Size Framing

Frames can be of fixed or variable size. In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. An example of this type of framing is the ATM wide-area network, which uses frames of fixed size called cells.

Variable-Size Framing

Character-Oriented

Protocols In a character-oriented protocol, data to be carried are 8-bit characters from a coding system such as ASCII . The header, which normally carries the source and destination addresses and other control information, and the trailer, which carries error detection or error correction redundant bits, are also multiples of 8 bits. To separate one frame from the next, an 8-bit (I-byte) flag is added at the beginning and the end of a frame. The flag, composed of protocol-dependent special characters, signals the start or end of a frame.

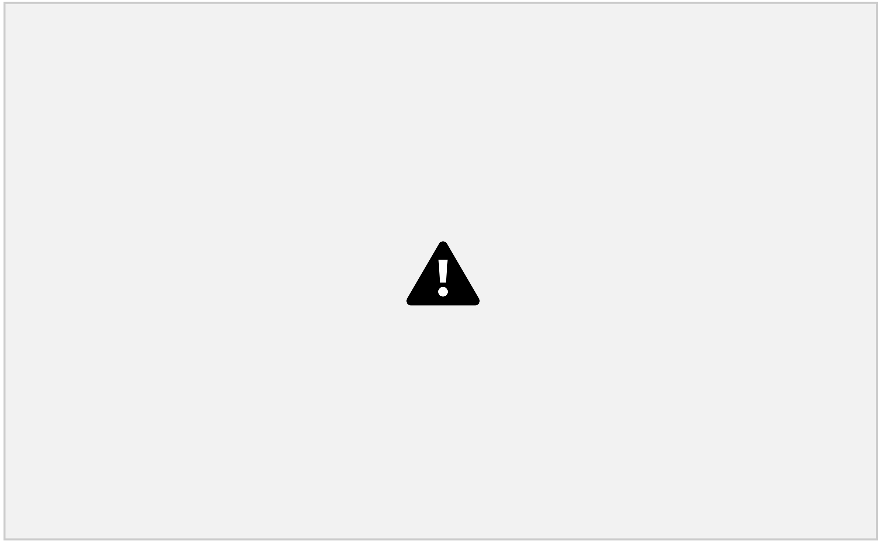
Bit-Oriented Protocols

Bit stuffing

Flow Control

Flow control

Error control - Error control in the data link layer is based on automatic repeat request, which is the retransmission of data



PROTOCOLS

CHANNELS

NOISELESS

Simplest Protocol

Figure 11.7 ***Flow diagram for Example 11.1***

STOP-AND-WAIT PROTOCOL, AUTOMATIC

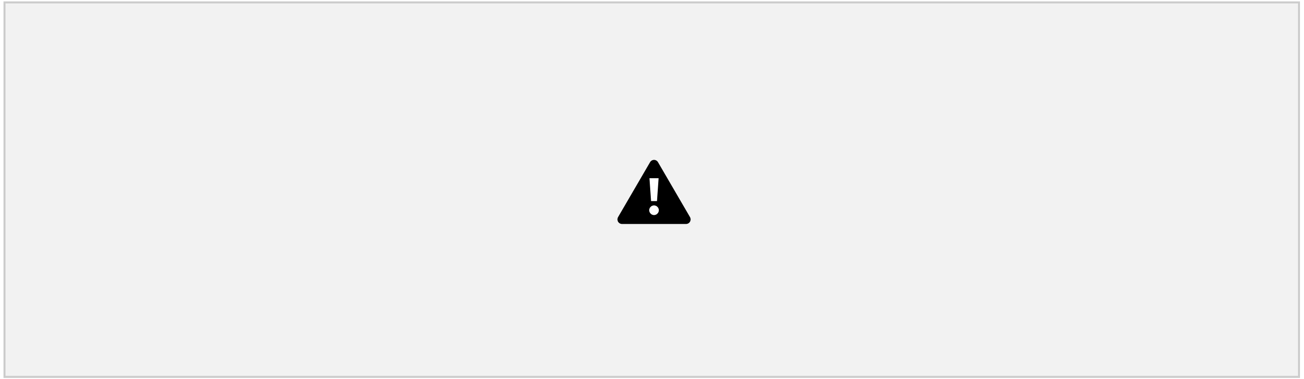
Stop-and-Wait Protocol, 2

In normal operation

- *S* transmits a frame and wait for an ACK from *D* during a give time.
- *D* transmits an ACK after receiving an error-free frame.

In case of a frame dropped

- *S* retransmits the frame after the time-out period expires.



Stop-and-Wait Protocol, 3

In case of frame garbled

- D does nothing, and then S will retransmit.

In case of ACK dropped

- S retransmits the frame
- D receives the frame and transmits another ACK. - D will receive duplicated frames!

Add a sequence number to the frame

So, D will discard the second copy of the frame, but resend ACK.

NOISY CHANNELS

Although the Stop-and-Wait Protocol gives us an idea of how to add flow control to its predecessor, noiseless channels are nonexistent. We discuss three protocols in this section that use error control.

IMPORTANT NOTES

- Error correction in Stop-and-Wait ARQ is done by keeping a copy of the sent frame and retransmitting of the frame when the timer expires.
- In Stop-and-Wait ARQ, we use sequence numbers to number the frames. The sequence numbers are based on modulo-2 arithmetic.

.

- In Stop-and-Wait ARQ, the acknowledgment number always announces in modulo-2 arithmetic the sequence number of the next frame expected.

Stop-and-Wait Protocol, 1

- In case of delayed ACK or prematured time-out. • S will retransmit the frame, say FR-0, after the time-out. • S receives an ACK, but what it is for? either for the first frame or for the retransmitted frame?
  - S sends the next frame, say FR-1 and receive an ACK. What

Size of Sequence Number

- The number of frames to be sent is arbitrary, but the space for the sequence is finite (in header)

1-bit sequence number is good enough for Stop-and-Wait.

- Slast := # of frame in buffer of S (or to being sent).

  Dnext := # of frame that D expects to receive

Figure 11.10 *Design of the*

Stop-and-Wait ARQ Protocol

Algorithm 11.5 *Sender-site algorithm for Stop-and*

Wait ARQ

Algorithm 11.6 *Receiver-site algorithm for Stop-and-Wait ARQ Protocol*

31

Efficiency

`The Stop-and-Wait ARQ discussed in the previous section is very inefficient if our channel is *thick and long. By thick, we mean that our channel has a large bandwidth; by long,* we mean the round-trip delay is long. The product of these two is called the bandwidth-delay product. We can think of the . channel as a pipe. The bandwidth-delay product then is the volume of the pipe in bits. The pipe is always there. If we do not use it, we are inefficient. The bandwidth-delay product is a measure of the number of bits we can send out of our system while waiting for news from the receiver.

Example 11.4

Assume that, in a Stop-and-Wait ARQ system, the bandwidth of the line is 1 Mbps, and 1 bit takes 20 ms to make a round trip. What is the bandwidth-delay product? If the system data frames are 1000 bits in length, what is the utilization percentage of the link?

Solution

The bandwidth-delay product is .

$(1*10^6)*(20*10^{-3})$=20,000 bits.

The system can send 20,000 bits during the time it takes for the data to go from the sender to the receiver and then back again. However, the system sends only 1000 bits. We can say that the link utilization is only 1000/20,000, or 5 percent. For this reason, for a link with a high bandwidth or long delay, the use of Stop-and-Wait ARQ wastes the capacity of the link.

11.5

Example

What is the utilization percentage of the link in Example 11.4 if we have a protocol that can send up to 15 frames before stopping and worrying about the acknowledgments?

Solution

The bandwidth-delay product is still 20,000 bits.

The system can send up to 15 frames or 15,000 bits during a round trip. This means the utilization is 15,000/20,000, or 75 percent. Of course, if there are damaged frames, the utilization percentage is much less because frames have to be resent.

Pipelining

In networking and in other areas, a task is often begun before the previous task has ended. This is known as pipelining. There is no pipelining in Stop-and-Wait ARQ because we need to wait for a frame to reach the destination and be acknowledged before the next frame can be sent. However, pipelining does apply to our next two protocols because

GO-BACK-N PROTOCOL, AUTOMATIC REPEAT REQUEST

Motivation

- Inefficiency of Stop-and-Wait protocol
- S continuously sends enough frames so that the channel is kept busy while S waits for ACK.

Procedure

- S has a limit on the number of outstanding frames (S_W), and each frame is associated with a timer
- S sends frame 0 followed by additional frames, so S_W frames are outstanding.

Upon receiving ACK k, S sends frames up to S_w + k.

If time-out occurs, go back to S_w previous frames (outstanding frames) which are outstanding in S, and re-transmit all those outstanding frames.

Go-Back-N Protocol, 2

- If errors occurs, the loss of transmission time for S_w frames.
  - In Stop-and-Wait, the loss was the time-out period.
- Each outstanding frame is associated with a timer.
- Outstanding frames are maintained by two variables: S_f and S_n.
  - S_f: the oldest outstanding frame.
  - S_n: the next frame to send.
  - $S_n - S_f < S_{size}$.

D has a single state variable: R_n:= the frame number to receive. If D receives frame k for k = R_n, it sends an ACK for R_n and update $R_n=R_n+1$.

Upon receiving ACK(R_n), S update $S_f = R_n$ and sends more frames such that at most S_w frames are outstanding.

Relations between the variables

$S_f \leq R_n < S_n$ and $S_n - S_f < S_w$

Go-Back-N Protocol, 3

- Size of Header (for sequence number)
  - *m* bits – capable of representing 0 to 2^m - 1.
  - If $S_w < 2m$, S and D can say which frame has been received or acknowledged without ambiguity.

Example

Assume S_w=2m, for m=2 (4)

Assume S_w=2m-1, for m=2 (3)

Go-Back-N Protocol, 4

Improvement

D sends a NAK(Rnext) after the first out-of-sequence message S retransmits from frames Rnext after receiving NAK(Rnext). Piggybacking

In bi-directional communication, ACK or NAK is piggybacked to an information frame in reverse direction.

What if there no user data for a certain period of time?

ACK timer.

Figure 11.12 *Send window for Go-Back-N ARQ*

Figure 11.13 *Receive window for Go-Back-N ARQ*

Figure 11.14 *Design of Go-Back-N ARQ*

Bac

Figure 11.15 *Window size for Go*

Algorithm 11.7 *Go-Back-N sender algorithm*

Algorithm 11.8 *Go-Back-N receiver algorithm*

Example 11.6

Figure 11.16 shows an example of Go-Back-N. This is an example of a case where the forward channel is reliable, but the reverse is not. No data frames are lost, but some ACKs are delayed and one is lost. The example also shows how cumulative acknowledgments can help if acknowledgments are delayed or lost.

After initialization, there are seven sender events. Request events are triggered by data from the network layer; arrival events are triggered by acknowledgments from the physical layer. There is no time-out event here because all outstanding frames are acknowledged before the timer expires. Note that although ACK 2 is lost, ACK 3 serves as both ACK 2 and ACK 3.

Figure 11.16 *Flow diagram for Example 11.6*

Example 11.7

Figure 11.17 shows what happens when a frame is lost. Frames 0, 1, 2, and 3 are sent. However, frame 1 is lost. The receiver receives frames 2 and 3, but they are discarded because they are received out of order. The sender receives no acknowledgment about frames 1, 2, or 3. Its timer finally expires.

The sender sends all outstanding frames (1, 2, and 3) because it

does not know what is wrong. Note that the resending of frames 1, 2, and 3 is the response to one single event. When the sender is responding to this event, it cannot accept the triggering of other events. This means that when ACK 2 arrives, the sender is still busy with sending frame 3.

Figure 11.17 *Flow diagram for Example 11.7*

NOTE

Figure 11.18 Send window for Selective Repeat ARQ

Figure 11.19 Receive window for Selective Repeat ARQ

Figure 11.21 Selective Repeat ARQ, window size

Figure 11.20 Design of Selective Repeat ARQ

Selective Repeat

In Selective Repeat ARQ, the size of the sender and receiver window must be at most one-half of 2^m.

Algorithm 11.9 Sender-site Selective Repeat algorithm

Algorithm 11.10 Receiver-site Selective Repeat algorithm

Figure 11.22 Delivery of data in Selective

Repeat ARQ

Example 11.8

Example 11.8 (continued)

Example 11.8 (continued)

sec

Example 11.8 (continued)

Figure 11.23 Flow diagram for Example 11.8

Figure 11.24 Design of piggybacking in Go-Back-N ARQ

HDLC

High-level Data Link Control (HDLC) is a bit oriented protocol for communication over point-to point and multipoint links. It implements the ARQ mechanisms we discussed in this chapter.

Figure 11.25 Normal response mode

Figure 11.26 Asynchronous balanced mode

Figure 11.27 HDLC frames

Figure 11.28 Control field format for the different frame types

Table 11.1 U-frame control command and response

Example 11.9

Figure 11.29 Example of connection and disconnection

Example 11.10

Example 11.10 (continued)

Figure 11.30 Example of piggybacking without error

Example 11.11