CHAPTER 11

PRACTICE SET

Questions

Q11-1. The data link layer needs to pack bits into *frames*. Framing divides a message into smaller entities to make flow and error control more manageable.

Q11-2. *Piggybacking* is used to improve the efficiency of bidirectional transmission. When a frame is carrying data from A to B, it can also carry control information about the frame(s) received from B; when a frame is carrying data from B to A, it can also carry control information about the frames received from A.

Q11-3. PPP is normally used to create a point-to-point communication between a main computer and a desktop computer such as the case when a desktop computer at home is connected to the main computer of an ISP. When we talk about the system, we are referring to the ISP computer; when we talk about the user, we are referring to the desktop computer.

Q11-4. In variable-size framing, flags are needed to separate a frame from the previous one and the next one.

Q11-5. The slop of the line between two vertical time line shows the time spent to move from one situation to another. We assume that encapsulation of the packets in frames, or vice versa, takes a very short time (negligible). We have shown this as the horizontal line. On the other hand, sending a frame from one station to another through media takes a loner time. We have shown this as a diagonal line.

Q11-6. The events are not shown in Figure 11-21 because they are either explicit or internal.

Q11-7. The timer belongs to the sender entity. The *ready* or *blocking* is the state of the sender. However, in this protocol, the sender does not use the timer when it is in ready state. It only uses when it is in the blocking state.

Q11-8. In one-way communication, the data frames are moving from the sender to the receiver; the acknowledgments are moving from the receiver to the sender. Since we require that the data frames be acknowledged, we need a timer at the sender site to resend lost or corrupted data frames. Since acknowledgments frame are not required to be acknowledged (vicious circle), we do not need a timer at the receiver site. If an acknowledgment frame is lost, the sender interprets the situation as the loss of the data frame and resend the data frame.

Q11-9. The flags are the delimiters of the original frames. We need first to unstuff the frame to remove extra bits. The flags are removed later when we want to deliver data to the upper layer.

Q11-10. The flags are the delimiters of the original frames. We need first to unstuff the frame to remove extra characters. The flags are removed later when we want to deliver data to the upper layer.

Q11-11. Byte-oriented protocols use *byte-stuffing* to be able to carry an 8-bit pattern that is the same as the flag. Byte-stuffing adds an extra character to the data section of the frame to escape the flag-like pattern. Bit-oriented protocols use *bit-stuffing* to be able to carry patterns similar to the flag. Bit-stuffing adds an extra bit to the data section of the frame whenever a sequence of bits is similar to the flag.

Q11-12. PPP is used for point-to-point communication. The sender and receiver of the packet are located at the two end of the line. PPP uses an address field to be compatible with HDLC protocol. It is set to the $(11111111)_2$ to define a dummy address (broadcasting which is never used in a point-to-point protocol).

Q11-13. As we said in answer to the question Q11-10, theoretically, there should be only one frame in transit at any time in the Stop-and-Wait protocol. However, after a timeout, if the sent frame is not acknowledged, the sender assumes that the frame is lost and sends a copy of the frame, but this does not mean that two frames are in transit. This is the situation in Figure 11.13 when Frame 1 is lost.

Q11-14. In a *byte-oriented protocol*, data to be carried are 8-bit characters from a coding system. Byte-oriented protocols were popular when only text was exchanged by the data link layers. In a *bit-oriented protocol*, the data section of a frame is a sequence of bits. Bit-oriented protocols are more popular today because we need to send text, graphic, audio, and video which can be better represented by a bit pattern than a sequence of characters.

Q11-15. PPP uses two different one-byte values as the flag and the escape bytes. The flag byte is $(01111110)_2$, but the escape byte is $(01111101)_2$.

Q11-16. The Simple protocol is designed for error-free medium (which never materialized). The Ethernet protocol is an example of Stop-and-Wait protocol. If a packet is lost or corrupted, it should be resent. However, there is no explicit acknowledgment in the Ethernet. If a packet is lost or corrupted, the upper-layer protocols force the data-link to resend the frame.

Q11-17. The packet from the network layer should be rejected.

Q11-18. *Flow control* refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment. *Error control* refers to a set of procedures used to detect and correct errors.

Q11-19. 19 In this protocol, the S-frame is used for acknowledgment.

Q11-20. *HDLC* is a *bit-oriented protocol* for communication over point-to-point and multipoint links. *PPP* is a byte-oriented protocol used for point-to-point links.

Q11-21. The answer is negative. We need to distinguish between a *byte* and a *character.* It is better to think of a byte as the data unit at the data-link layer and the character as the data unit at the application layer. The application layer was designed to use one byte representing a character (ASCII). Today the tendency is to use two or more bytes as representing one character (to show characters in other languages and for other purposes). This does not mean that if a character is represented as two or more bytes at the application layer, we need to change the protocol at the data-link layer. A character in the application layer may change to two bytes (or more), but at the data-link layer, we treat them as two separate bytes. The size of the flag or other control characters remain the same.

Q11-22. The Simple protocol is designed for error-free environment, which means there is no need for CRC.

Q11-23. We need two channels, but not necessarily two media. The same media can be divided into two channels using multiplexing techniques we learned before.

Q11-24. Theoretically, there should be only one frame in transit at any time in the Stop-and-Wait protocol. However, after a timeout, if the sent frame is not acknowledged, the sender assumes that the frame is lost and sends a copy of the frame, but this does not mean that two frames are in transit. This is the situation in Figure 11.12 when the second frame is lost.

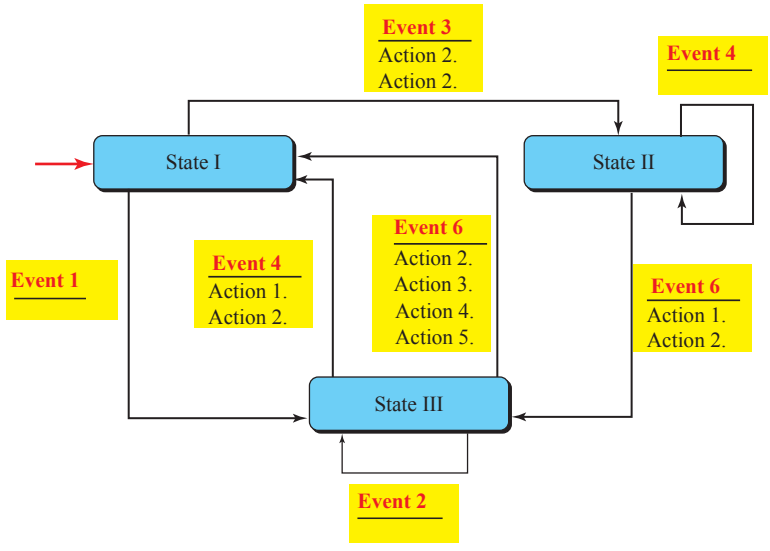
Problems

P11-1. The user and the system exchange LCP packets in this phase.

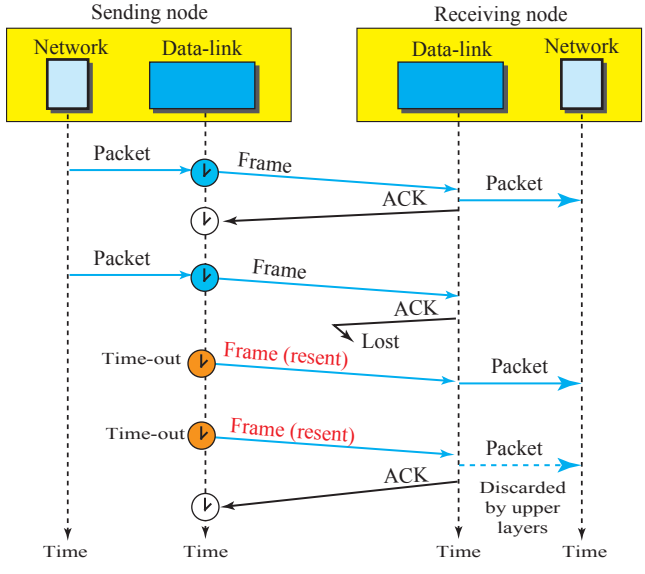
The user sends a *configuration request* packet.

The system responds with either a *configuration ack* or *configuration nak* packet.

P11-2. The following figure shows the states, events, actions, and transitions.



P11-3. The following figure shows the situation.



P11-4.

a. In this case, the time-out is less than the round-trip time. The first frame cannot be acknowledged before it is timed out. The sender sends the first frame over and over. The communication is useless.

b. In this case, a frame is acknowledged on time unless it is lost or encounters unusual delay. This is the ideal situation. The time-out is set correctly.

c. This is the marginal case. If there is a slight delay in a frame or acknowledgment, the frame is timed out and will be resent.

P11-5. Each escape or flag byte must be pre-stuffed with an escape byte. The following shows the result. The red bytes show the added ones.

D	**E**	**E**	D	D	**E**	**E**	D	D	**E**	**E**	**E**	**F**	D	**E**	**F**	D

P11-6.

a. The packet is discarded. The receiver in this one-directional setup is not supposed to get a packet from the upper layer; it is supposed to deliver the packet to the upper layer.

b. The frame is discarded.

c. The acknowledgment is discarded. The receiver is supposed to receive data frames and send acknowledgment.

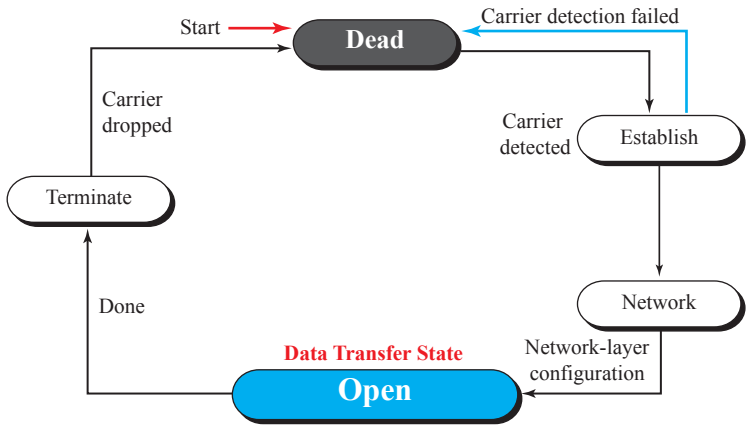
P11-7.

a. The sender stops the timer and discards the saved frame. It is ready to receive new packets from the upper layer.

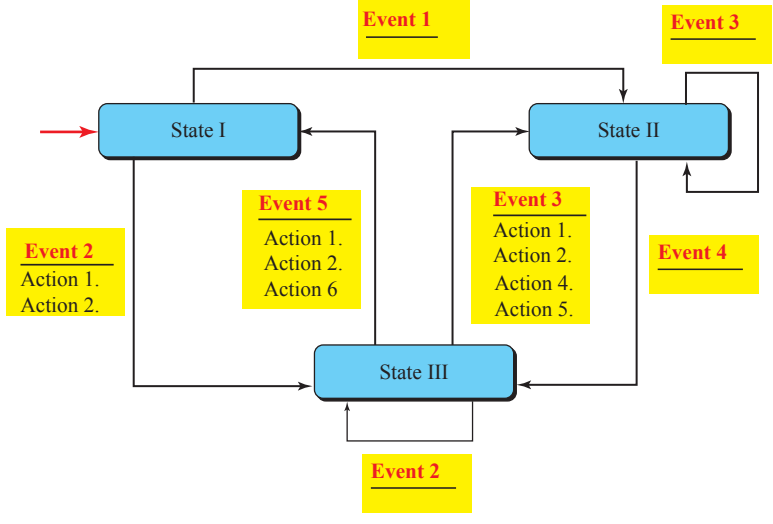
b. The sender resend a copy of the saved frame and starts a new timer.

c. This situation never happens. When the timer is running, the sender is at the blocking state, not the ready state.

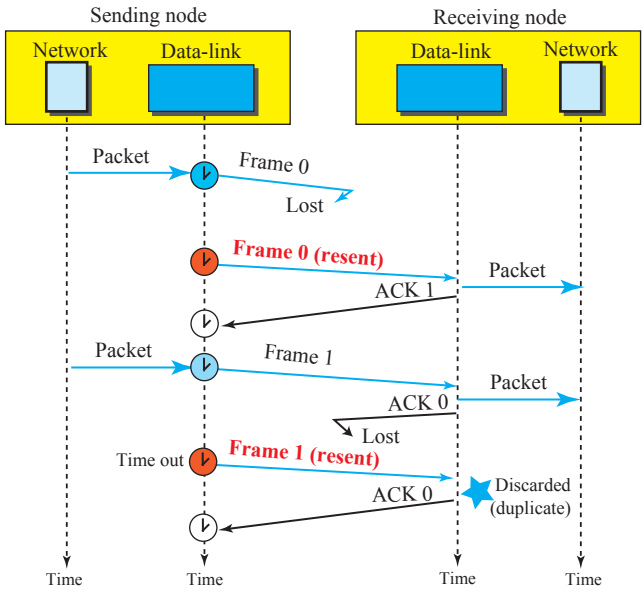
P11-8. See the following figure.



P11-9. The following figure shows the states, events, actions, and transitions.



P11-10. See the errata (Figure 11.13 instead of Figure 11.1). The following shows the situation.



P11-11. The following shows the result. We inserted extra 0 after each group of five consecutive 1's.

```
00011111000001111100100011111010111100000111
```

P11-12.

a. In this case, only 16 frames are exchanged:

2 frames for link-layer establishment

2 frames for network connection

10 frames for data transfer

2 frames for termination

b. In this case, 18 frames are exchanged:

2 frames for link-layer establishment

2 frames for network connection

2 frames for PAP (*request* and *ack*)

10 frames for data transfer

2 frames for termination

c. In this case, 19 frames are exchanged:

2 frames for link-layer establishment

2 frames for network connection

3 frames for CHAP (*challenge*, *response*, and *success)*

10 frames for data transfer

2 frames for termination

P11-13.

a. The following shows the exchange of payload:

The user sends an *authentication request* packet.

The system sends either an *ack* or a *nak* packet.

b.

The system sends a *challenge* packet.

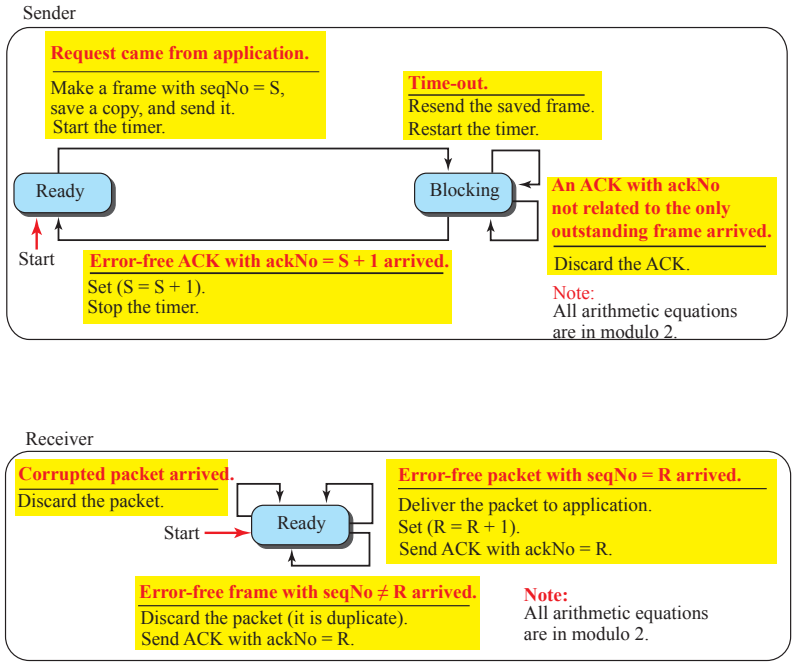
The user sends a *response* packet.

The system sends either a *success* or a *failure* packet.

P11-14. We remove the zero that comes after five 1's.

000111110000111111110100111011111000011111

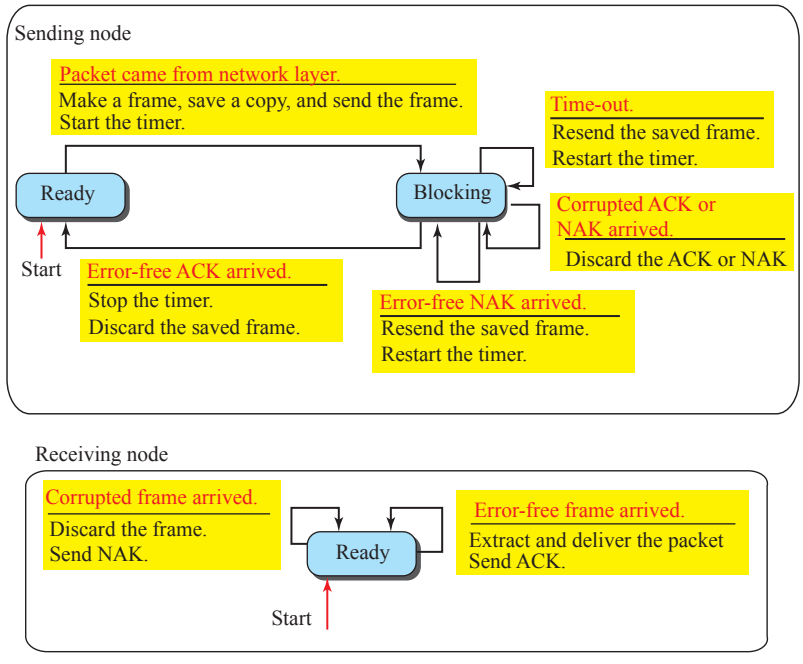
P11-15. The following figure shows the FSM.



P11-16. We scan the text until we see an E. If the next byte is E or F, we remove the scanned byte.

E	D	**F**	D	D	**F**	**E**	D	D	D

P11-17. See the errata (The figure to be changed is Figure 11.11 not Figure 11.9). We change the figure as shown below. Note that sender remains in the blocking state when it receives a NAK.



P11-18. The following figures shows the situation.