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.

Q11-3. 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.

Q11-5. 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.

Q11-7. 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.

Q11-9. The packet from the network layer should be rejected.

Q11-11. 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.

Q11-13. 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.

Q11-15. 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.

Q11-17. We need two channels, but not necessarily two media. The same media can be divided into two channels using multiplexing techniques we learned before.

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

Q11-21. 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.

Q11-23. 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 $(01111101)_2$.

Problems

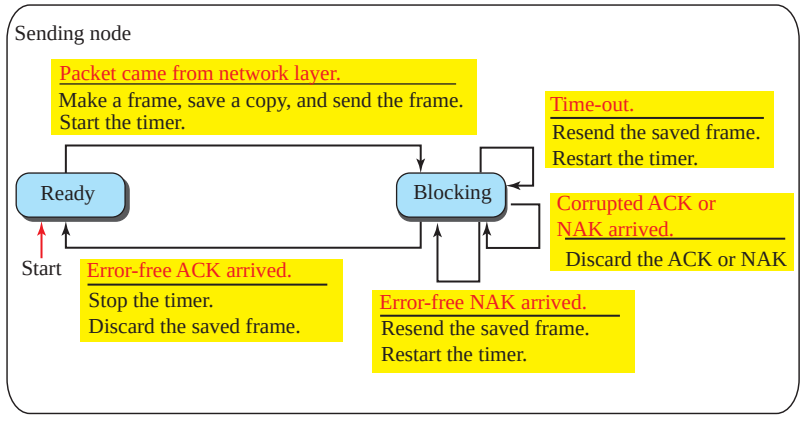
P11-1. Each escape or flag byte must be pre-stuffed with an escape byte. The following shows the result. The red bytes show the added ones.

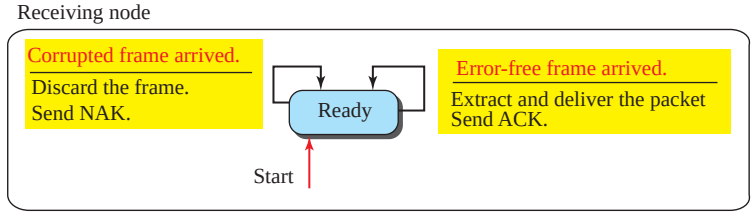
D	**E**	**E**	D	D	**E**	**F**	D	D	**E**	**E**	**E**	**E**	D	**E**	**F**	D

P11-3. The following shows the result. We inserted extra 0 after each group of five consecutive 1's.

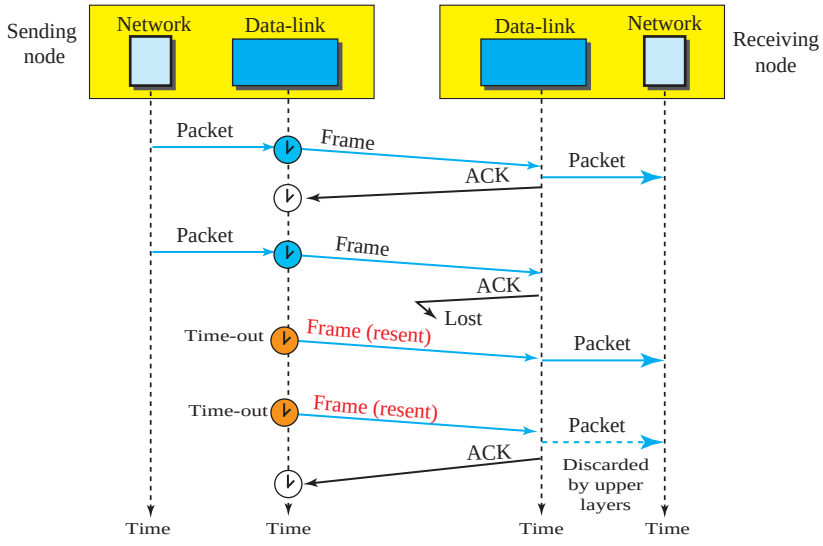
```
00011111011001111101001111101111110000111
```

P11-5. 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.





P11-7. The following figure shows the situation.



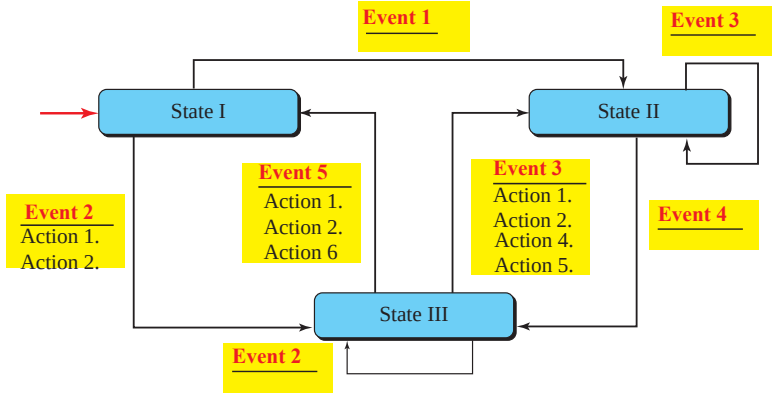
P11-9.

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

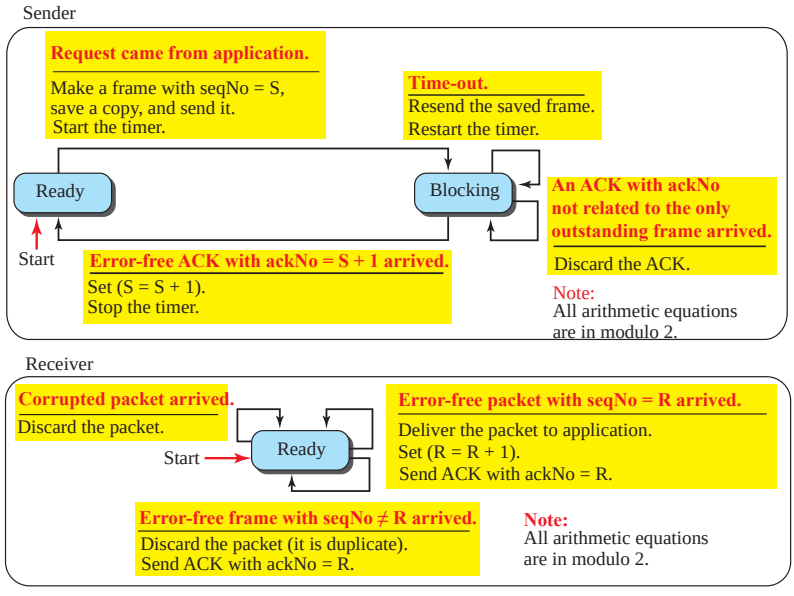
b. The sender resend a copy of the saved frame and starts a new timer.

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

P11-11. The following figure shows the states, events, actions, and transitions.



P11-13. The following figure shows the FSM.



P11-15. 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.

P11-17.

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.