

Data Link Control

- **Data Link Layer** functions - data link control and media access control
- **Data link control** - design and procedure for communication between two adjacent nodes.
- It include framing, flow and error control
- **Media access control** - manage sharing of link for communication

11-1 FRAMING

*The data link layer needs to pack bits into **frames**, so that each frame is distinguishable from another.*

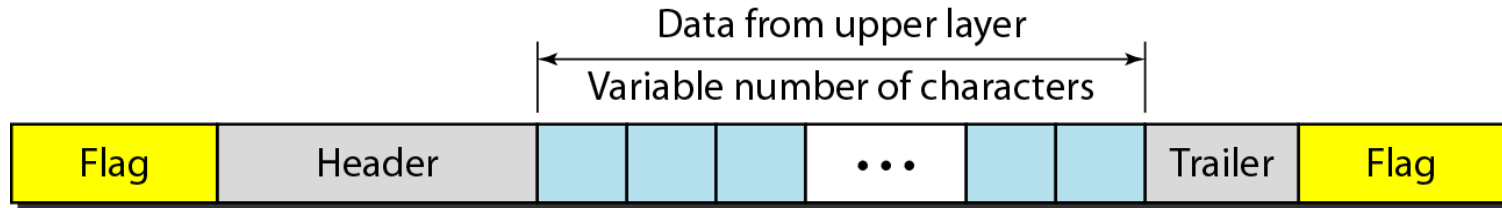
- Fixed-Size Framing

- Variable-Size Framing

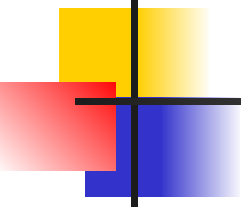
We need a way to define the end/beginning of a frame.

- Character oriented approach
- Bit-oriented approach

Figure 11.1 *A frame in a character-oriented protocol*



- Header - source and destination address, control information
- Trailer - error detection and correction redundant bits
- 8-bit flag in front and back for defining start and end of a frame



Byte stuffing is the process of adding 1 extra byte whenever there is a flag or escape character in the text.

- Flag could also be part of information in multi-media data
- The extra bit is called escape character (ESC), which has a pre-defined pattern
- Receiver whenever encounters the ESC character, it removes it and treats the next character as data

Figure 11.2 *Byte stuffing and unstuffing*

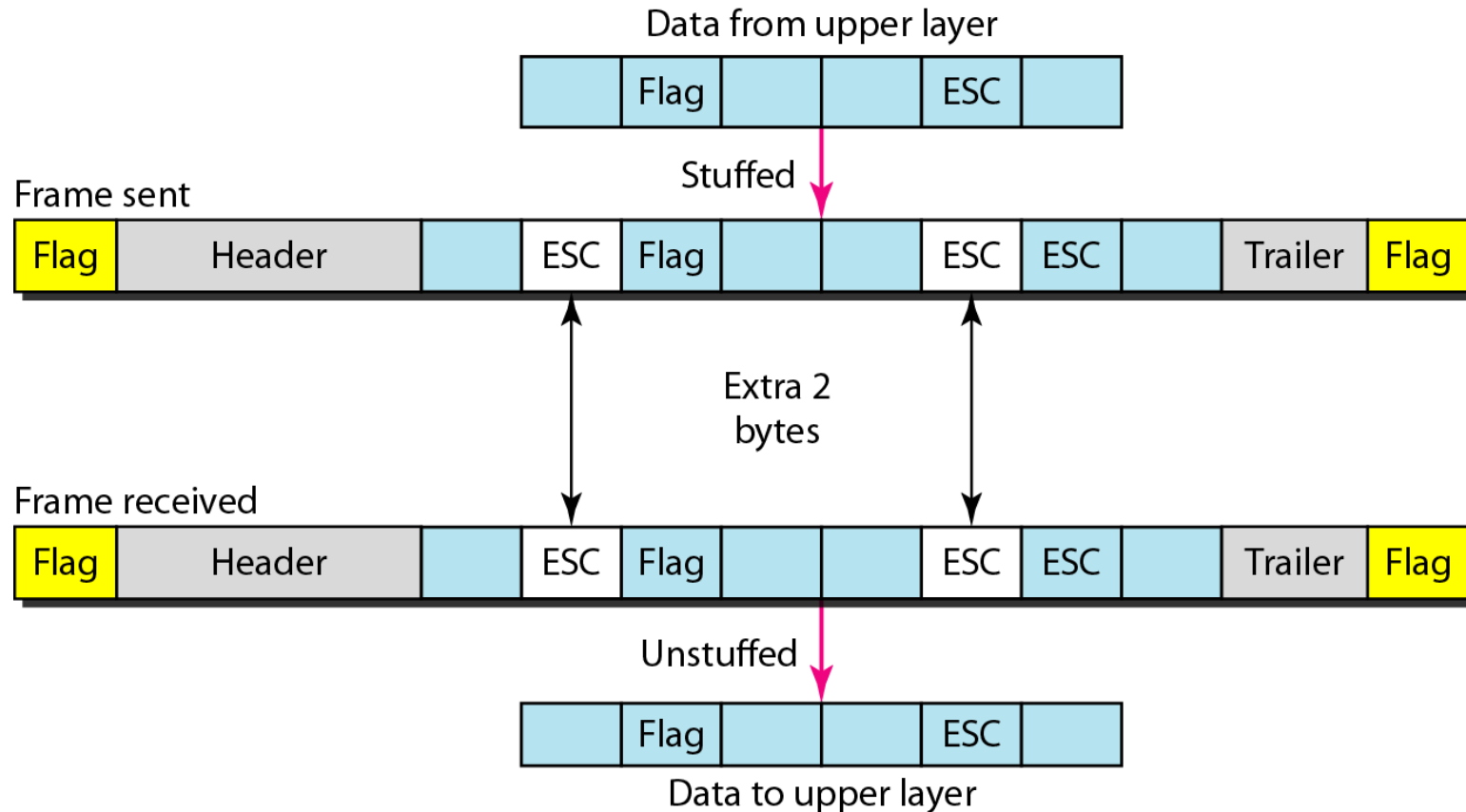
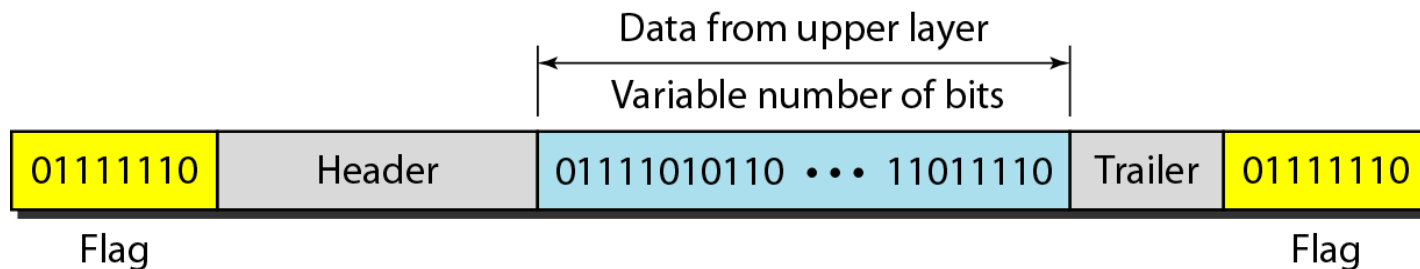
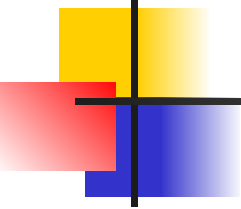


Figure 11.3 *A frame in a bit-oriented protocol*

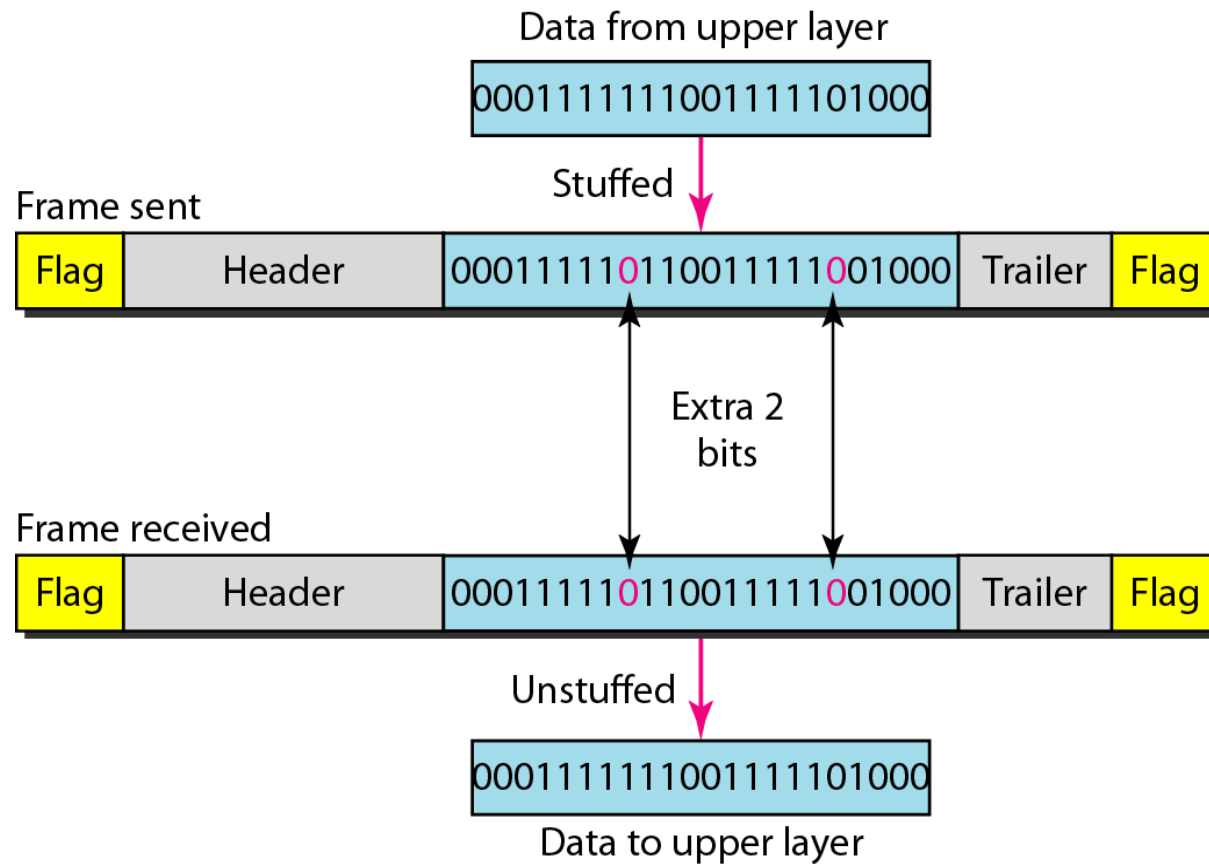
- The data section of a frame is a sequence of bits
- 8-bit fixed pattern flag is used as delimiter to define beginning/end of the frame





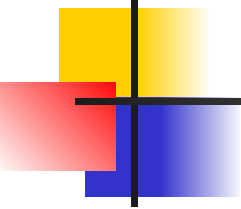
Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 011110 for a flag.

Figure 11.4 *Bit stuffing and unstuffing*

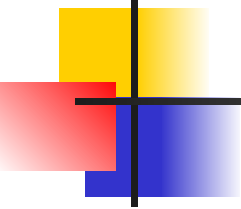


11-2 FLOW AND ERROR CONTROL

*The most important responsibilities of the data link layer are **flow control** and **error control**. Collectively, these functions are known as **data link control**.*



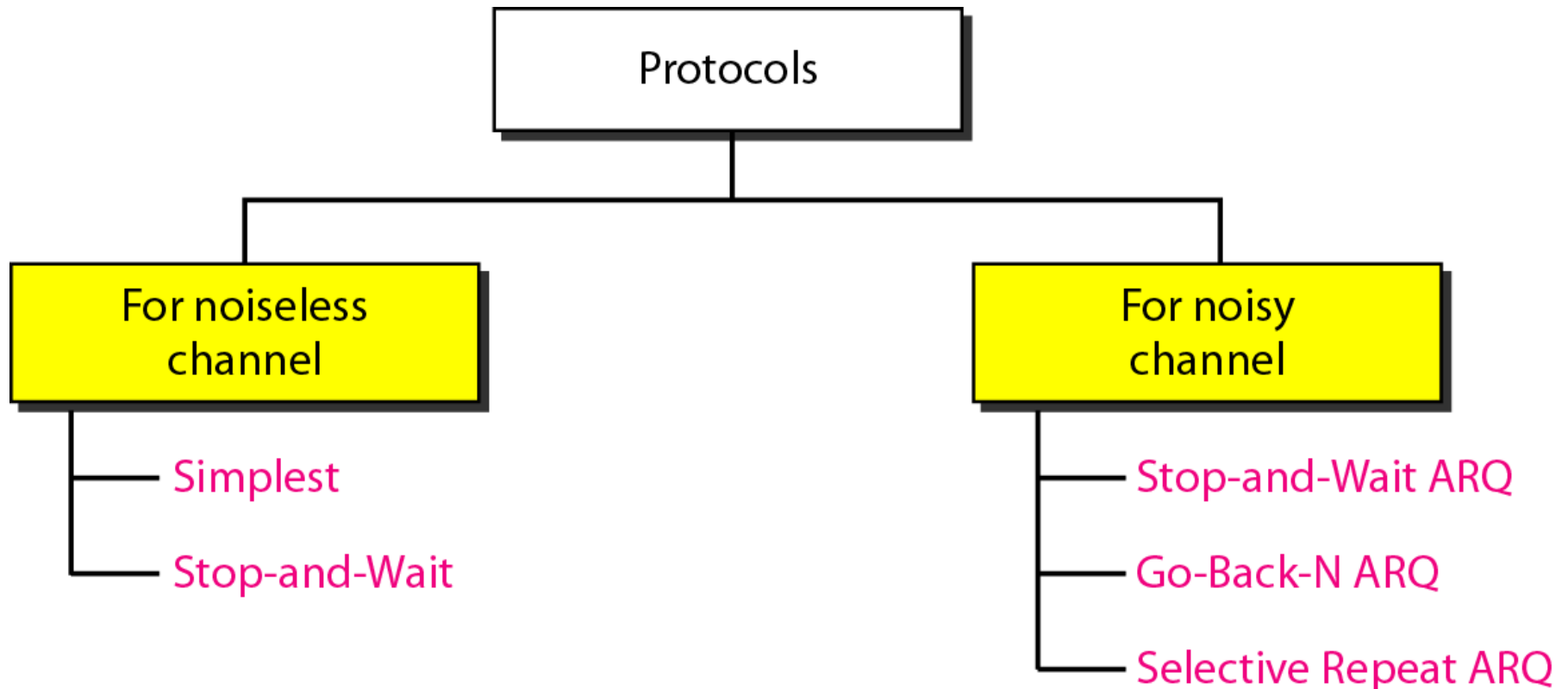
Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment.



Error control in the data link layer is based on automatic repeat request (ARQ), which is the retransmission of data.

Figure 11.5 *Taxonomy of protocols discussed in this chapter*

- *Protocols in data link layer to provide framing, flow control, and error control*



11-4 NOISELESS CHANNELS

Let us first assume we have an ideal channel in which no frames are lost, duplicated, or corrupted. We introduce two protocols for this type of channel.

Types of noiseless protocols -

- **Simplest Protocol (No flow control, No error control)**
- **Stop-and-Wait Protocol (Flow control, No error control)**

Figure 11.6 *The design of the simplest protocol with no flow or error control*

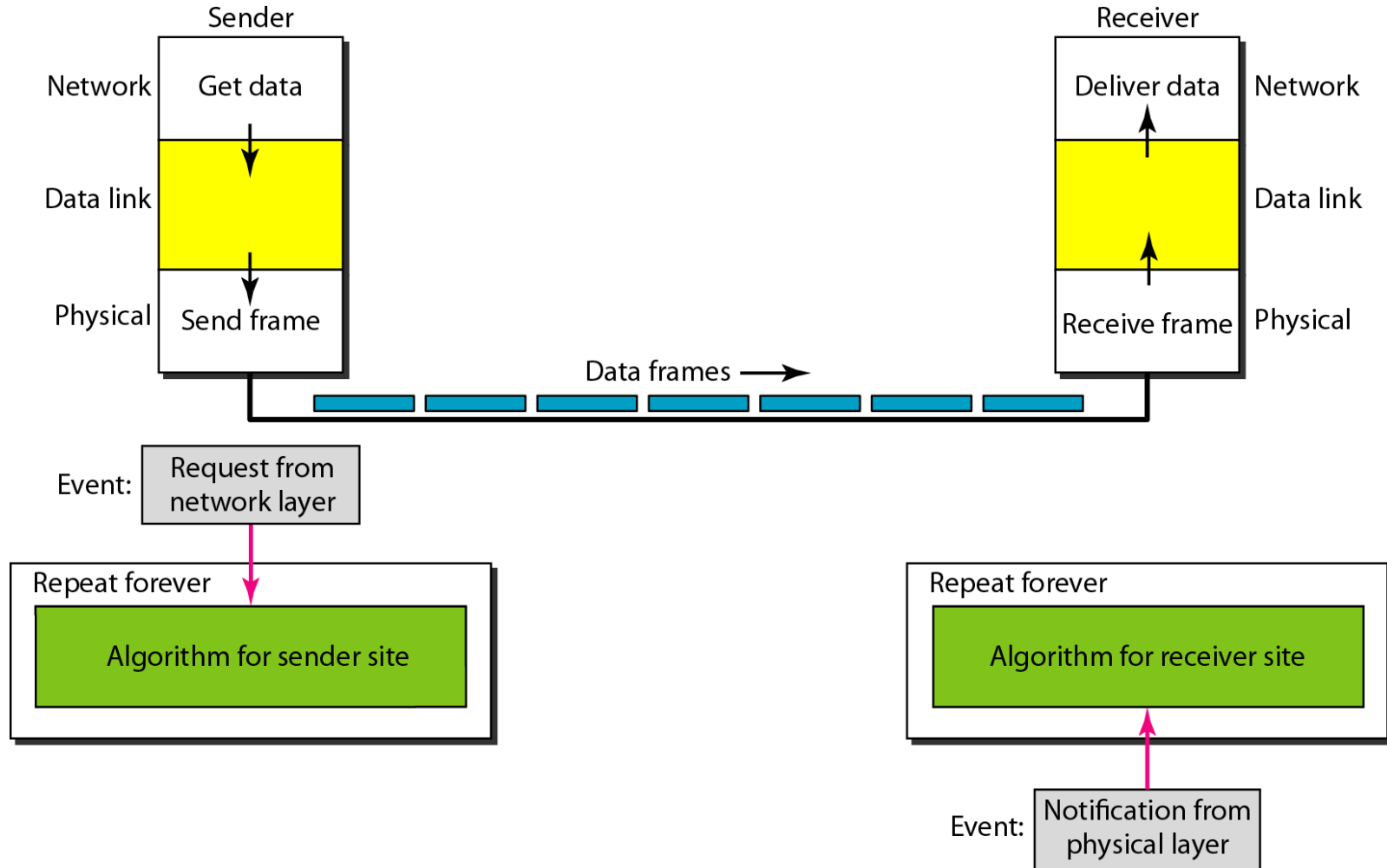
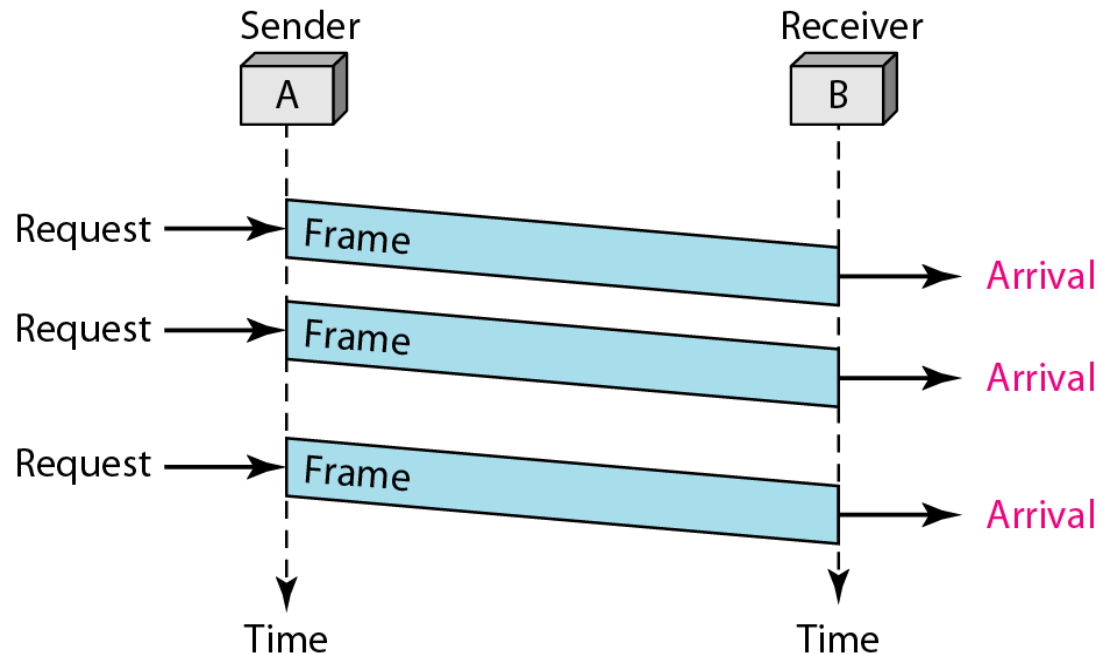


Figure 11.7 *Flow diagram for Example 11.1*



- Problem with this kind of communication is that receiver can overwhelmed with packets from the sender
- This may result in either discarding of frames or DoS.

Figure 11.8 *Design of Stop-and-Wait Protocol (for flow control)*

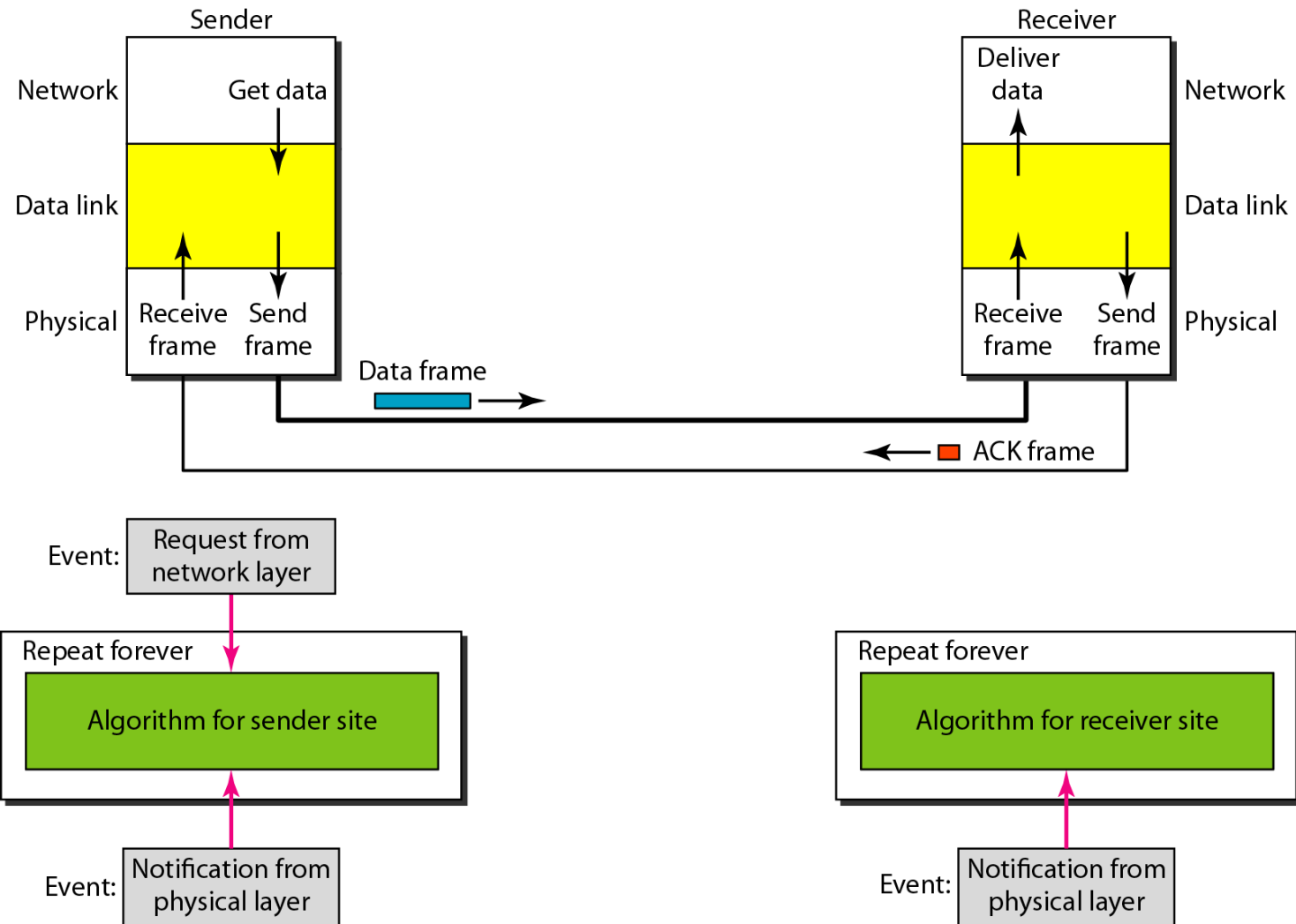
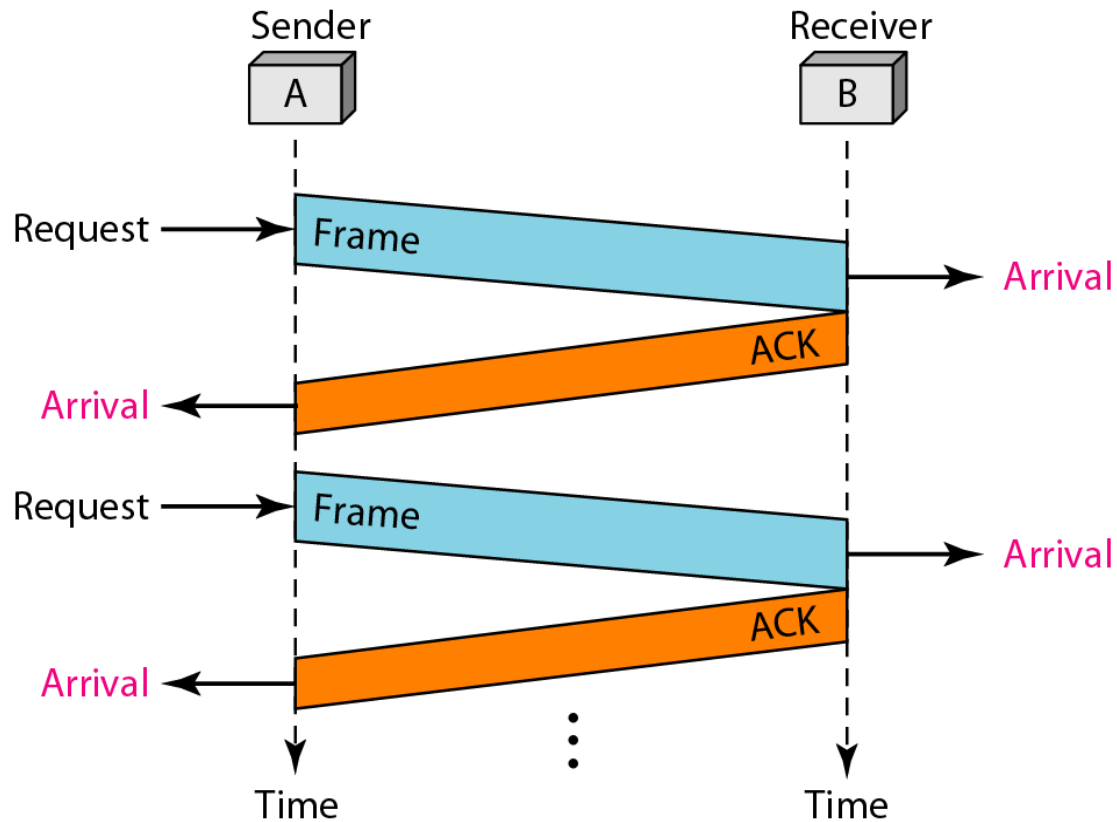


Figure 11.9 *Flow diagram for Example 11.2*



11-5 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.

Stop-and-Wait Automatic Repeat Request

Go-Back-N Automatic Repeat Request

Selective Repeat Automatic Repeat Request

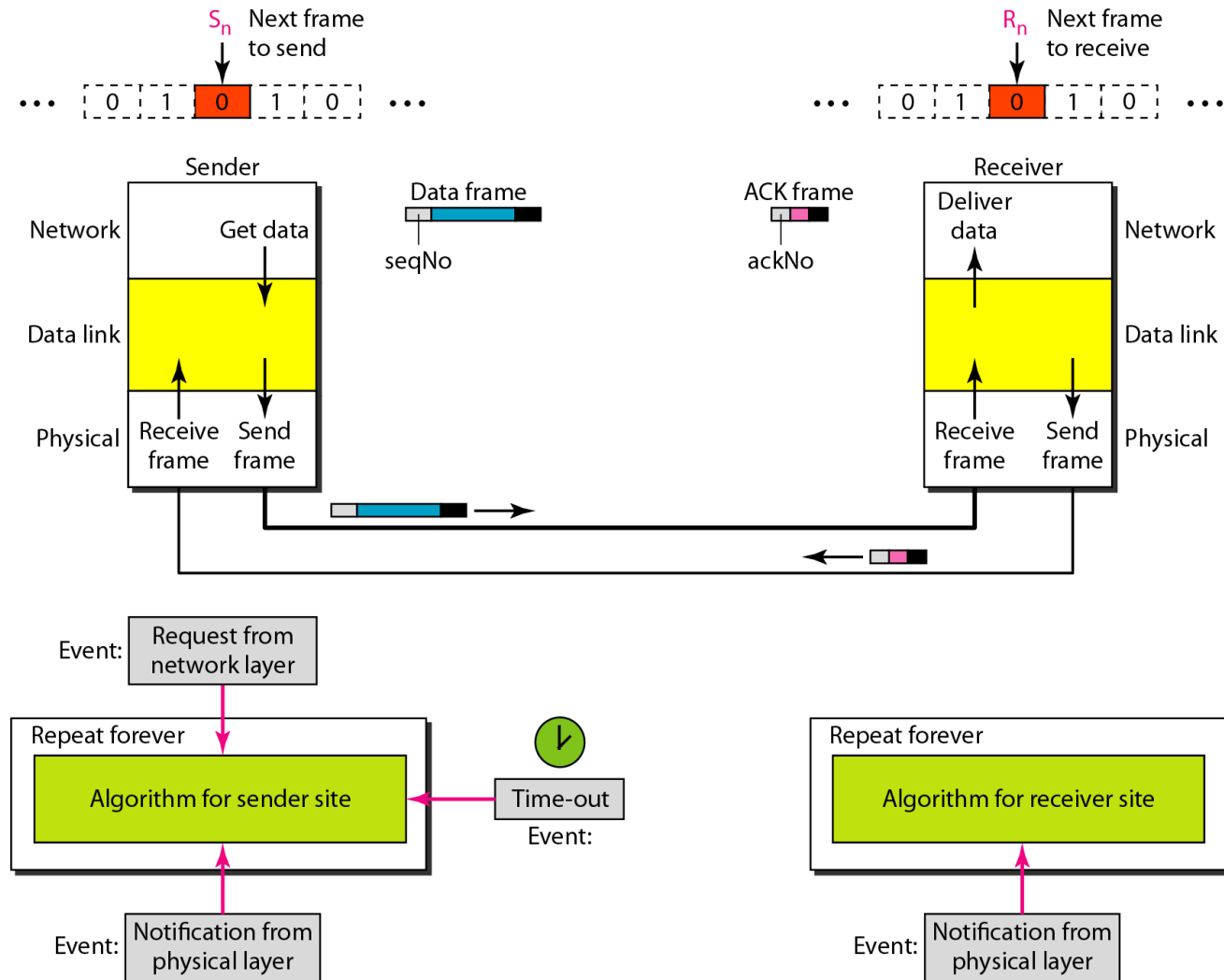


Stop and Wait Automatic Repeat Request

- 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.**

Figure 11.10 *Design of the Stop-and-Wait ARQ Protocol*



The diagram illustrates two network communication scenarios between a Sender (A) and a Receiver (B) over time.

Scenario 1: Stop-and-Wait Protocol

- Start:** The Sender requests Frame 0. The frame is received by the Receiver.
- Stop:** The Sender receives ACK 1 from the Receiver.
- Time-out:** The Sender requests Frame 1. The frame is lost.
- Stop:** The Sender receives ACK 0 from the Receiver.
- Start:** The Sender requests Frame 0. The frame is received by the Receiver.
- Time-out:** The Sender requests Frame 0 (resent). The frame is received by the Receiver.
- Stop:** The Sender receives ACK 1 from the Receiver.

Scenario 2: Go-Back-N Protocol

- Start:** The Sender requests Frame 0. The frame is received by the Receiver.
- Time-out:** The Sender requests Frame 0 (resent). The frame is received by the Receiver.
- Stop:** The Sender receives ACK 1 from the Receiver. The Receiver discards the duplicate frame.

The diagram uses a vertical timeline for each party, with events marked by circles (Start, Stop, Time-out) and rectangles (Request, Arrival). Data frames are represented by blue trapezoids, and acknowledgments by orange trapezoids. Lost frames are marked with a black star.

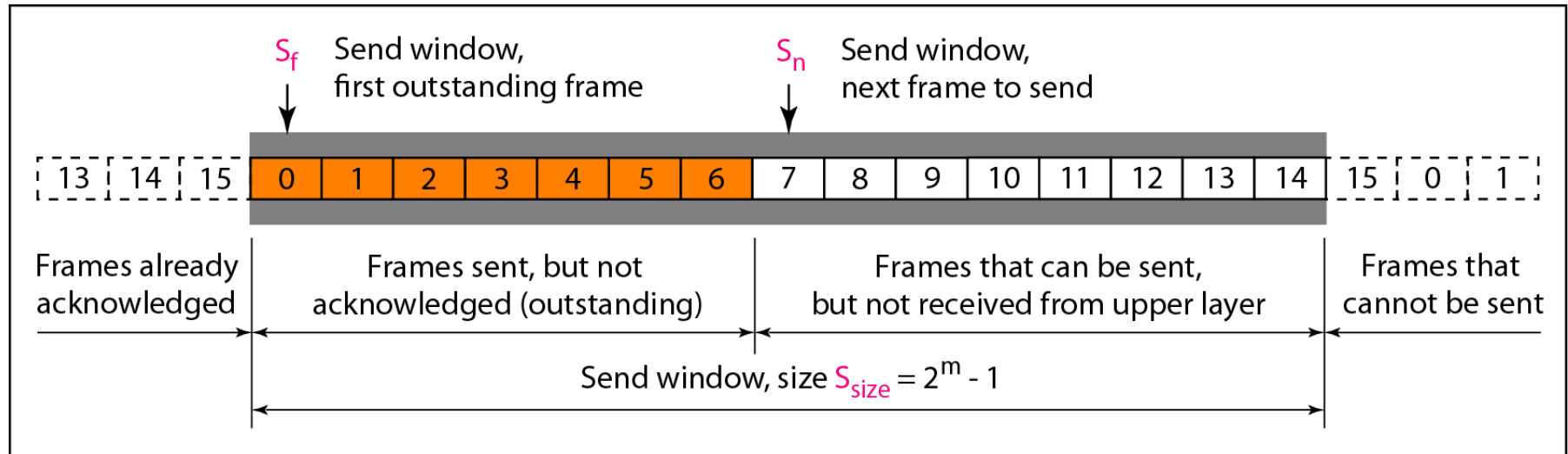


Go-Back-N Automatic Repeat Request

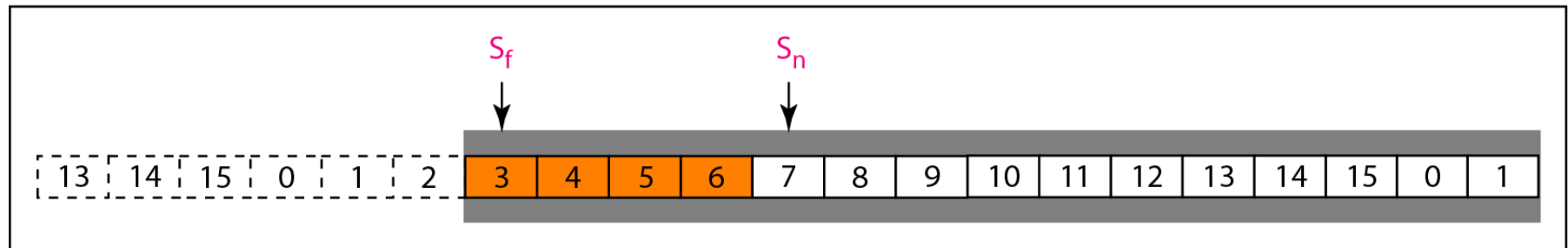
- To improve efficiency of transmission, multiple frames must be in transmission while waiting for the acknowledgment
- We keep copy of these frames until the acknowledgement arrives

In the Go-Back-N Protocol, the sequence numbers are modulo 2^m , where m is the size of the sequence number field in bits.

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



a. Send window before sliding

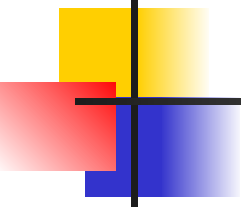


b. Send window after sliding



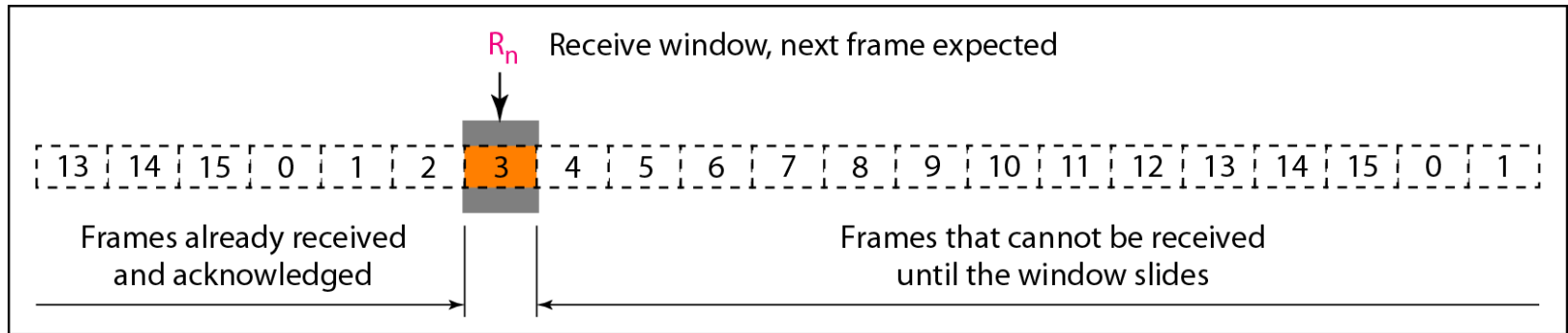
Note

The send window is an abstract concept defining an imaginary box of size $2^m - 1$ with three variables: S_f , S_n , and S_{size} .

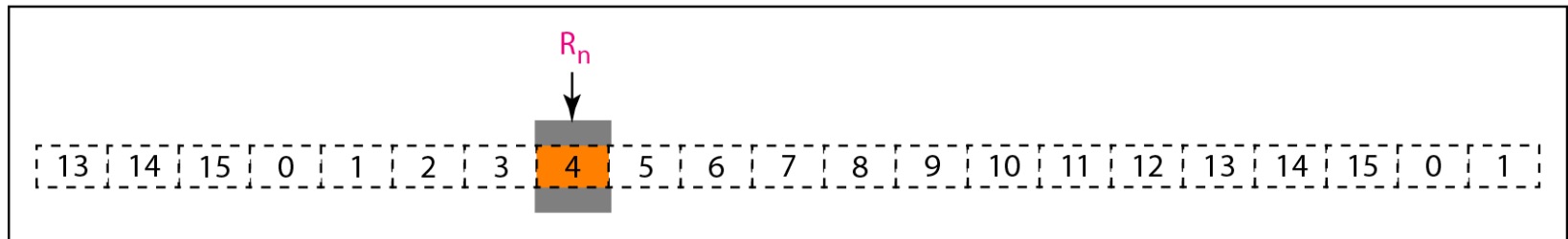


The send window can slide one or more slots when a valid acknowledgment arrives.

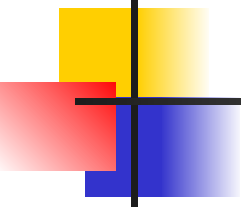
Figure 11.13 *Receive window for Go-Back-N ARQ*



a. Receive window



b. Window after sliding



The receive window is an abstract concept defining an imaginary box of size 1 with one single variable R_n .

**The window slides
when a correct frame has arrived; sliding
occurs one slot at a time.**

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

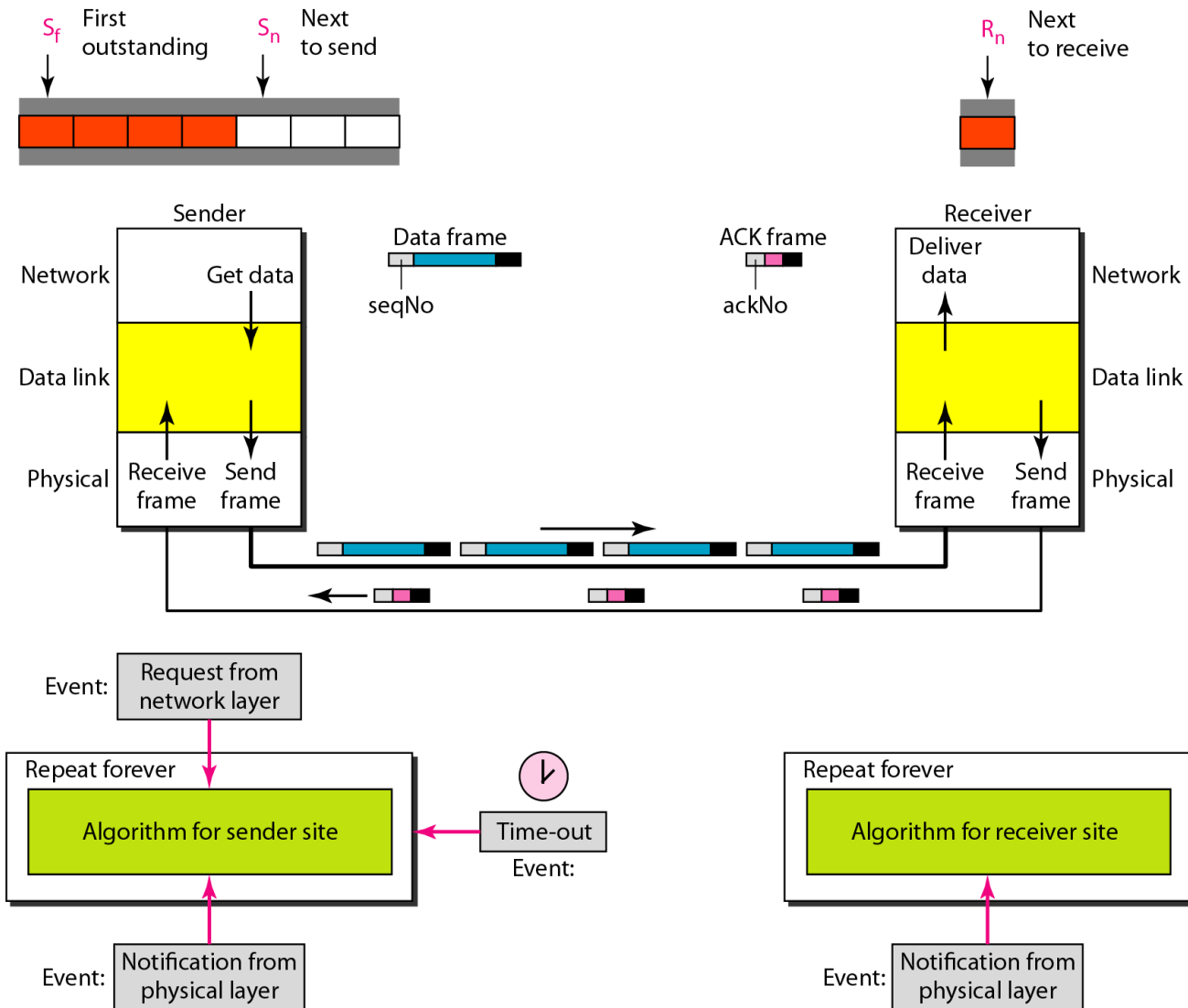
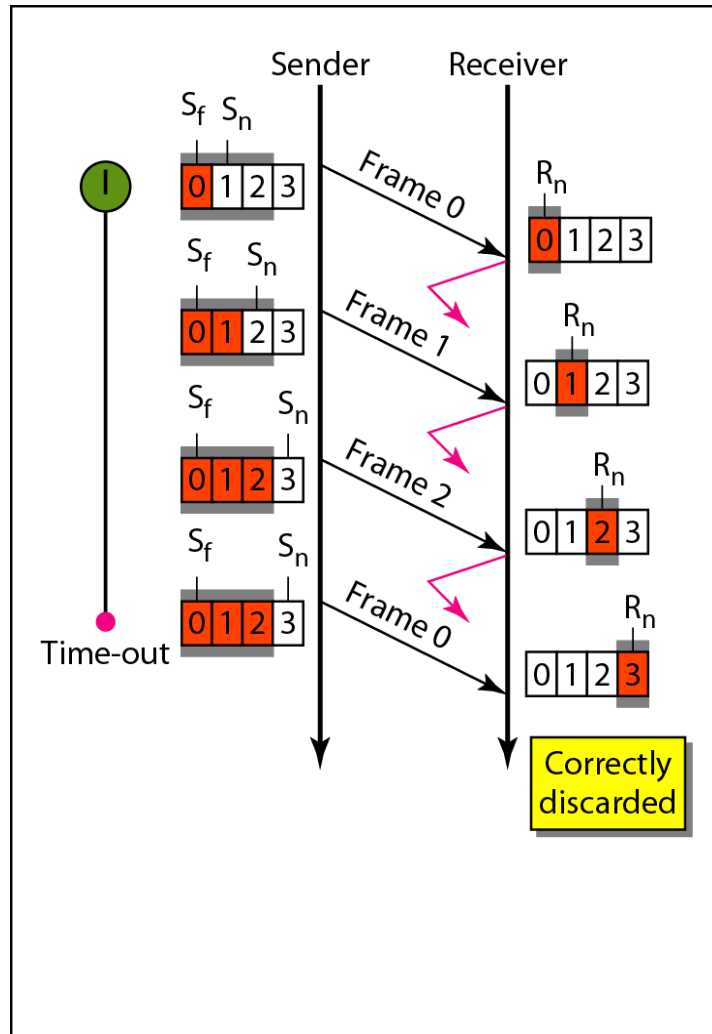
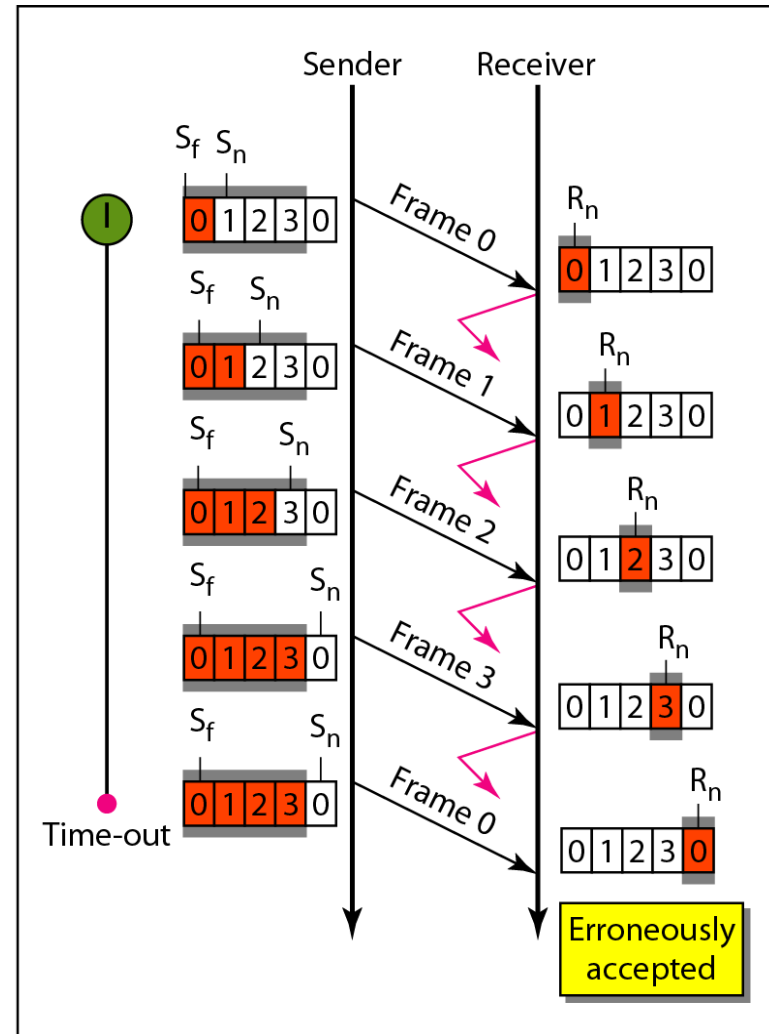


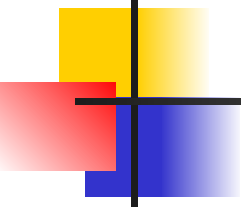
Figure 11.15 *Window size for Go-Back-N ARQ*



a. Window size $< 2^m$



b. Window size $= 2^m$



In Go-Back-N ARQ, the size of the send window must be less than 2^m ; the size of the receiver window is always 1.

Flow diagram to explain cumulative acknowledgements

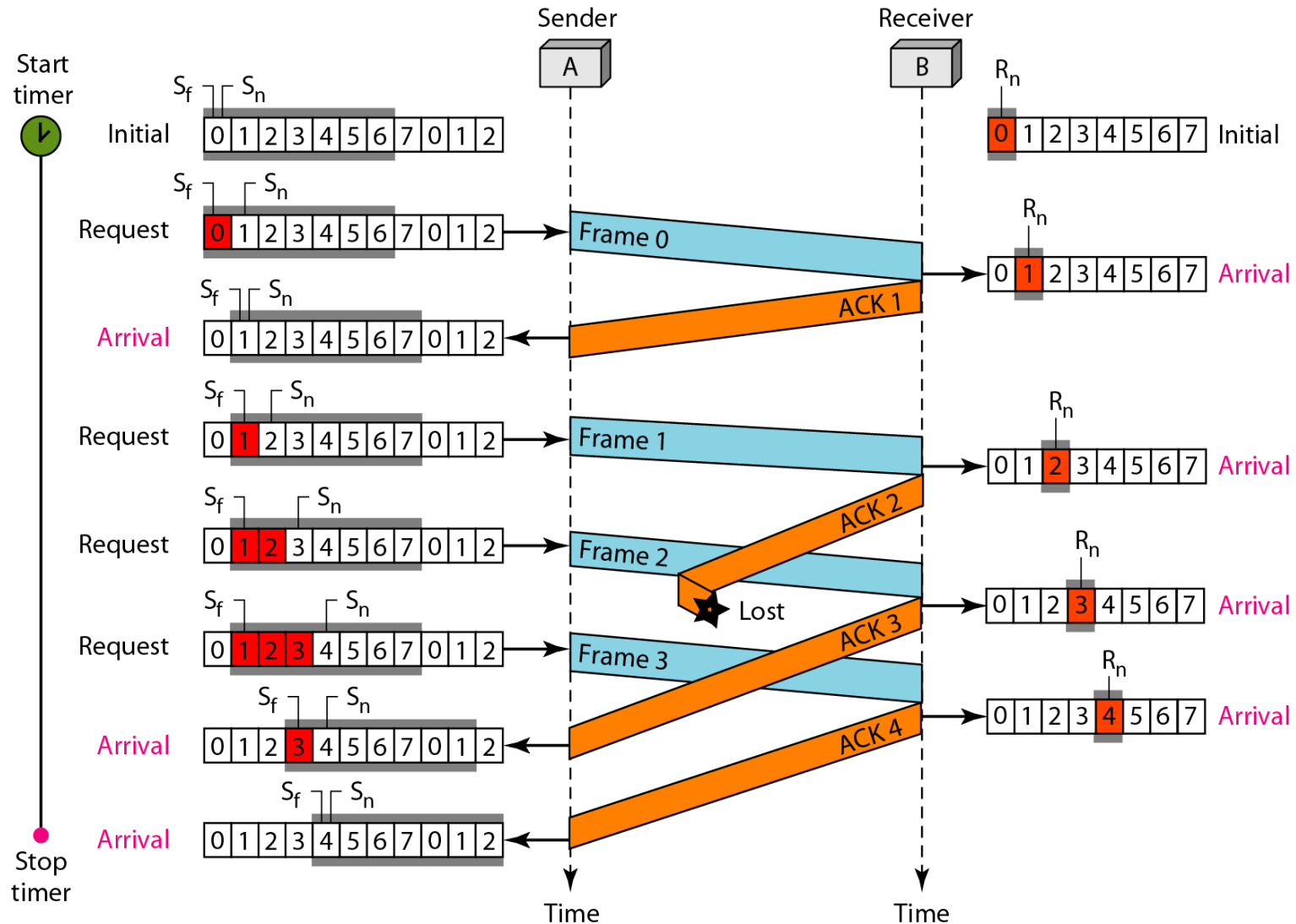
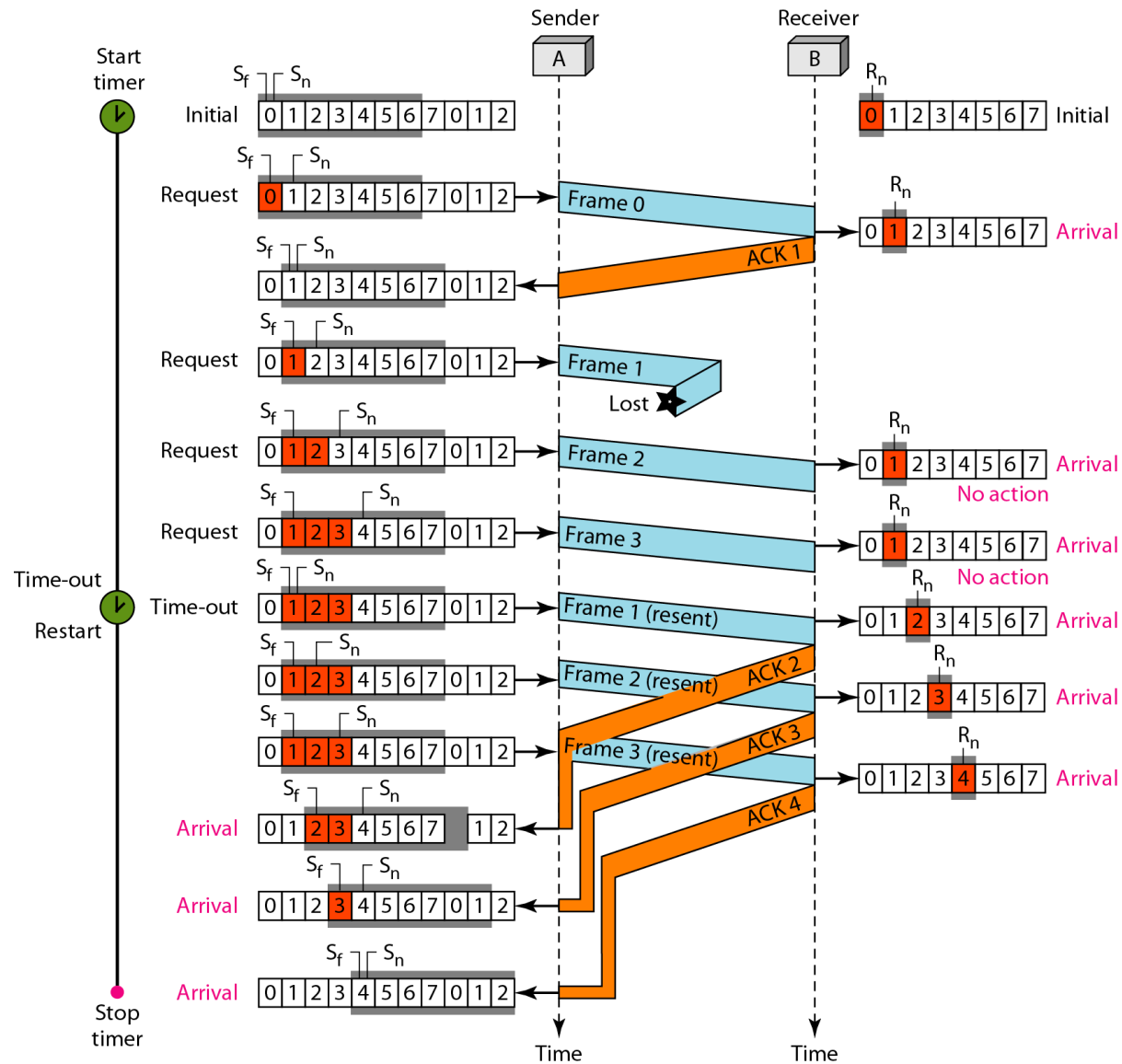
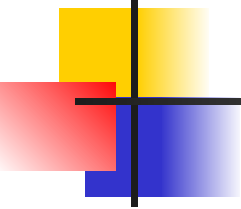


Figure 11.17 *Flow diagram to explain timeout concept*





Stop-and-Wait ARQ is a special case of Go-Back-N ARQ in which the size of the send window is 1.

- Go-Back-N maintain a window of size $(2^m)-1$ at sender location and a window of size 1 at receiver end
- Receiver cannot accept out of order frames as it do not has buffer
- In Selective repeat ARQ, sender and receiver both maintain a window size of $2^{(m-1)}$

Figure 11.18 *Send window for Selective Repeat ARQ*

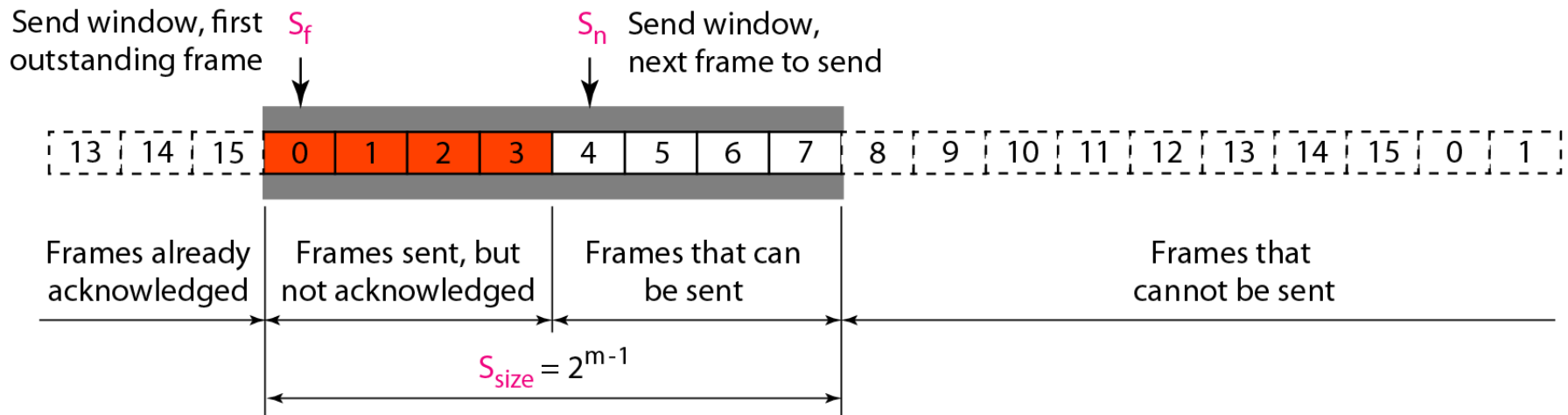


Figure 11.19 *Receive window for Selective Repeat ARQ*

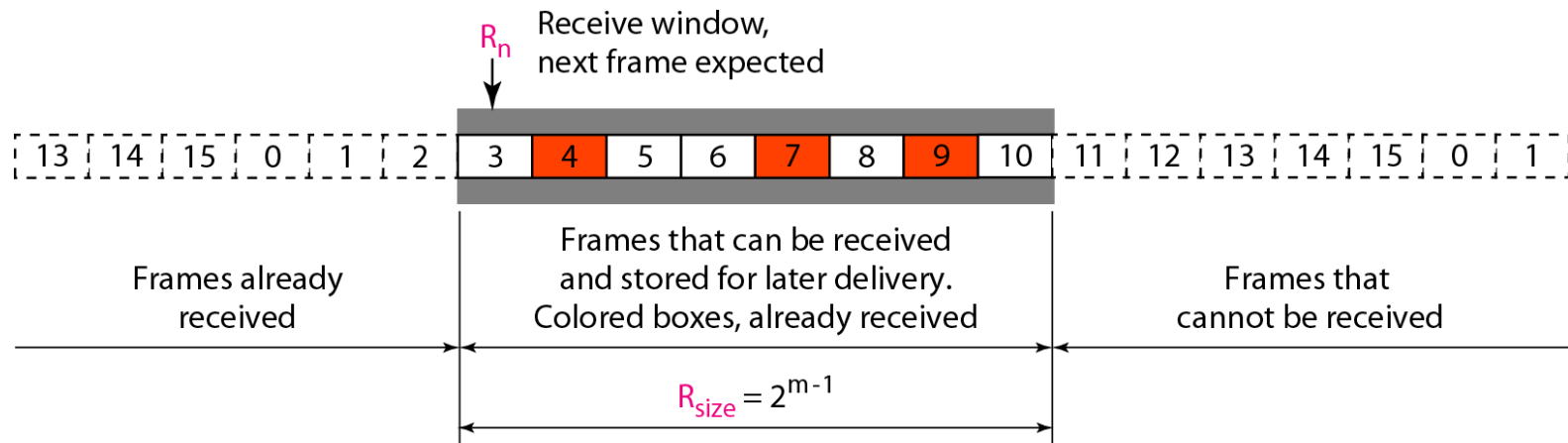
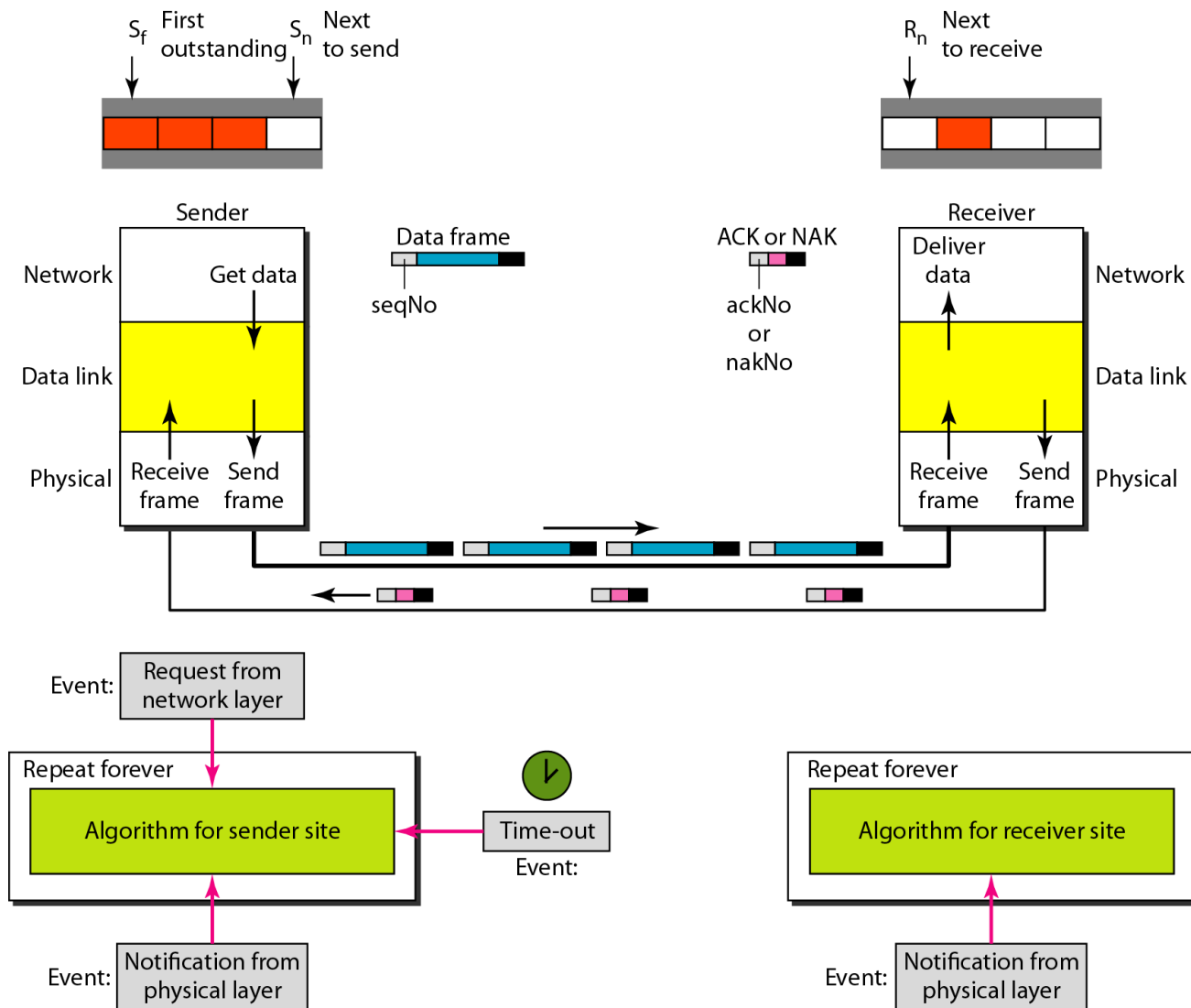


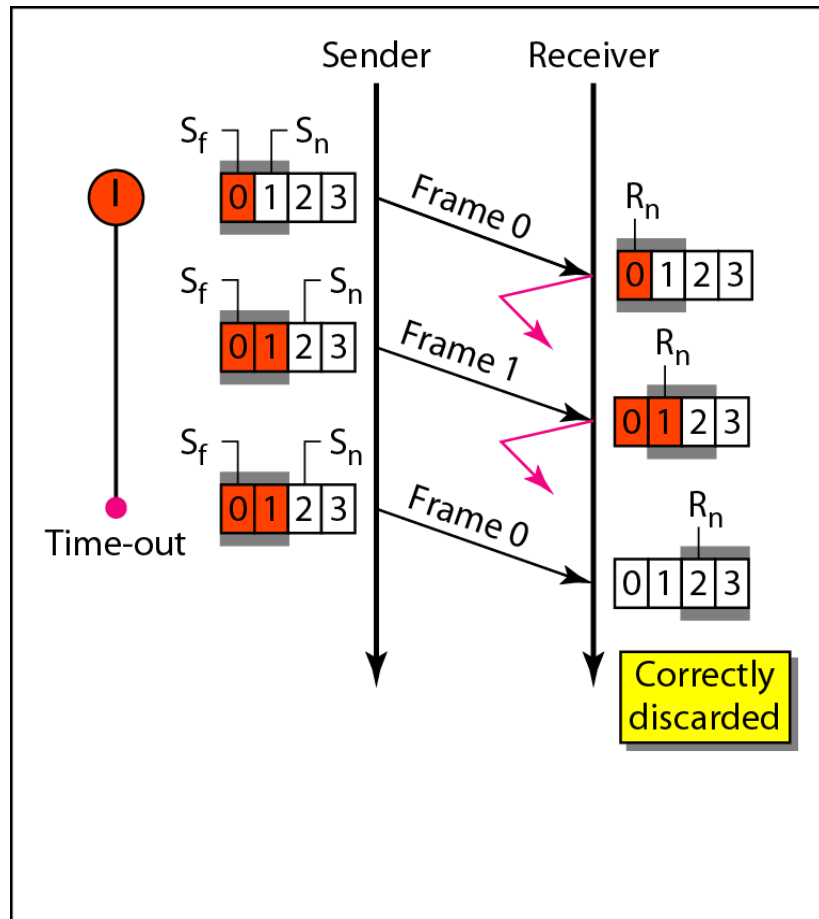
Figure 11.20 *Design of Selective Repeat ARQ*



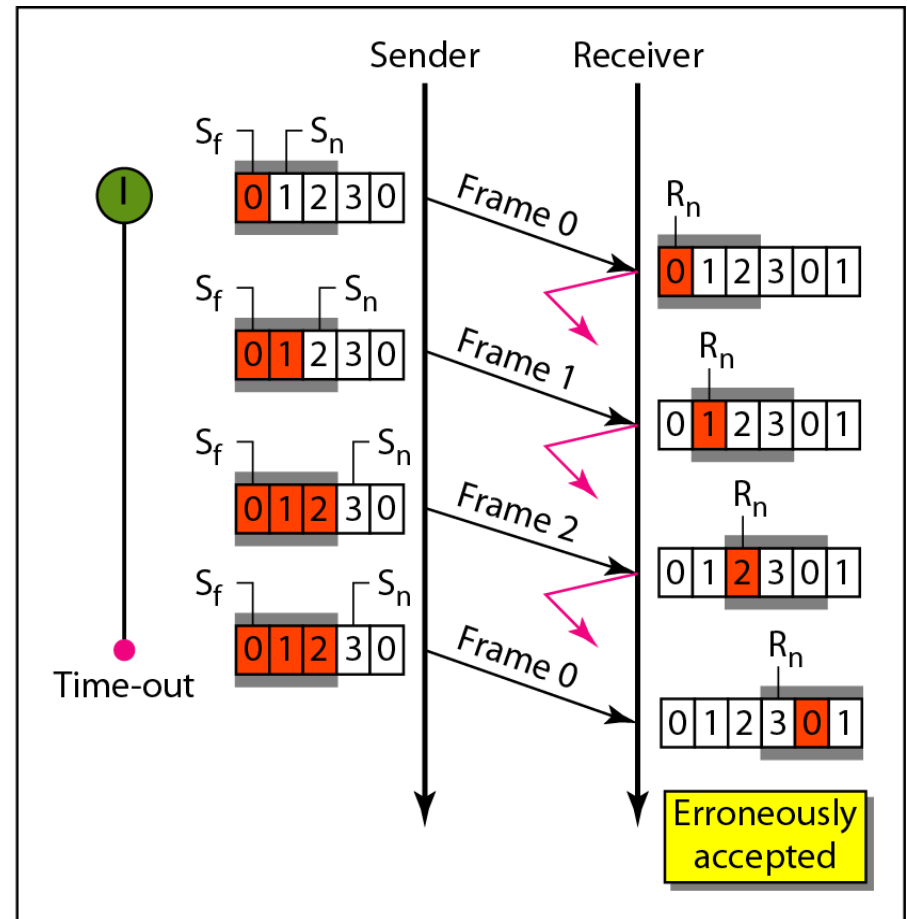


In Selective Repeat ARQ, the size of the sender and receiver window must be at most one-half of 2^m .

Figure 11.21 *Selective Repeat ARQ, window size*

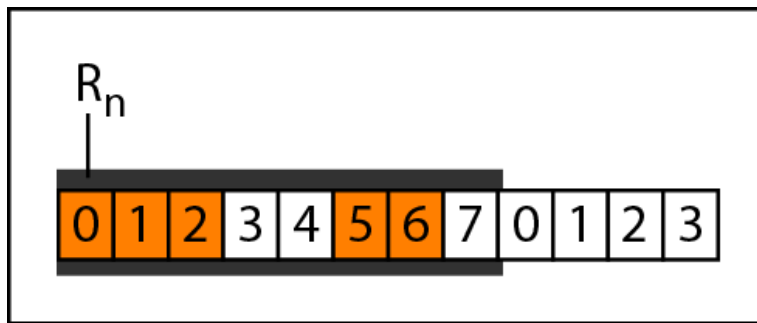


a. Window size = $2^m - 1$

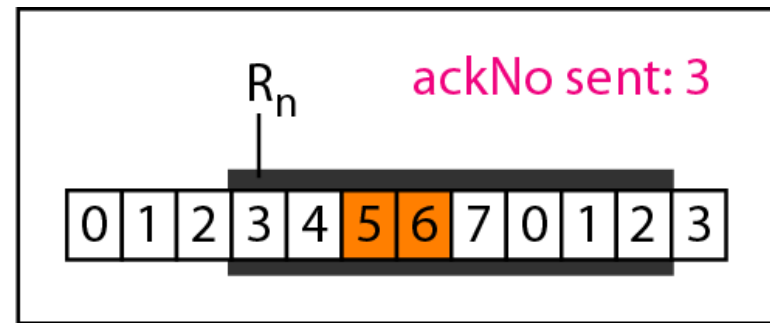


b. Window size > $2^m - 1$

Figure 11.22 *Delivery of data in Selective Repeat ARQ*



a. Before delivery



b. After delivery

Figure 11.23 *Flow diagram for Example 11.8*

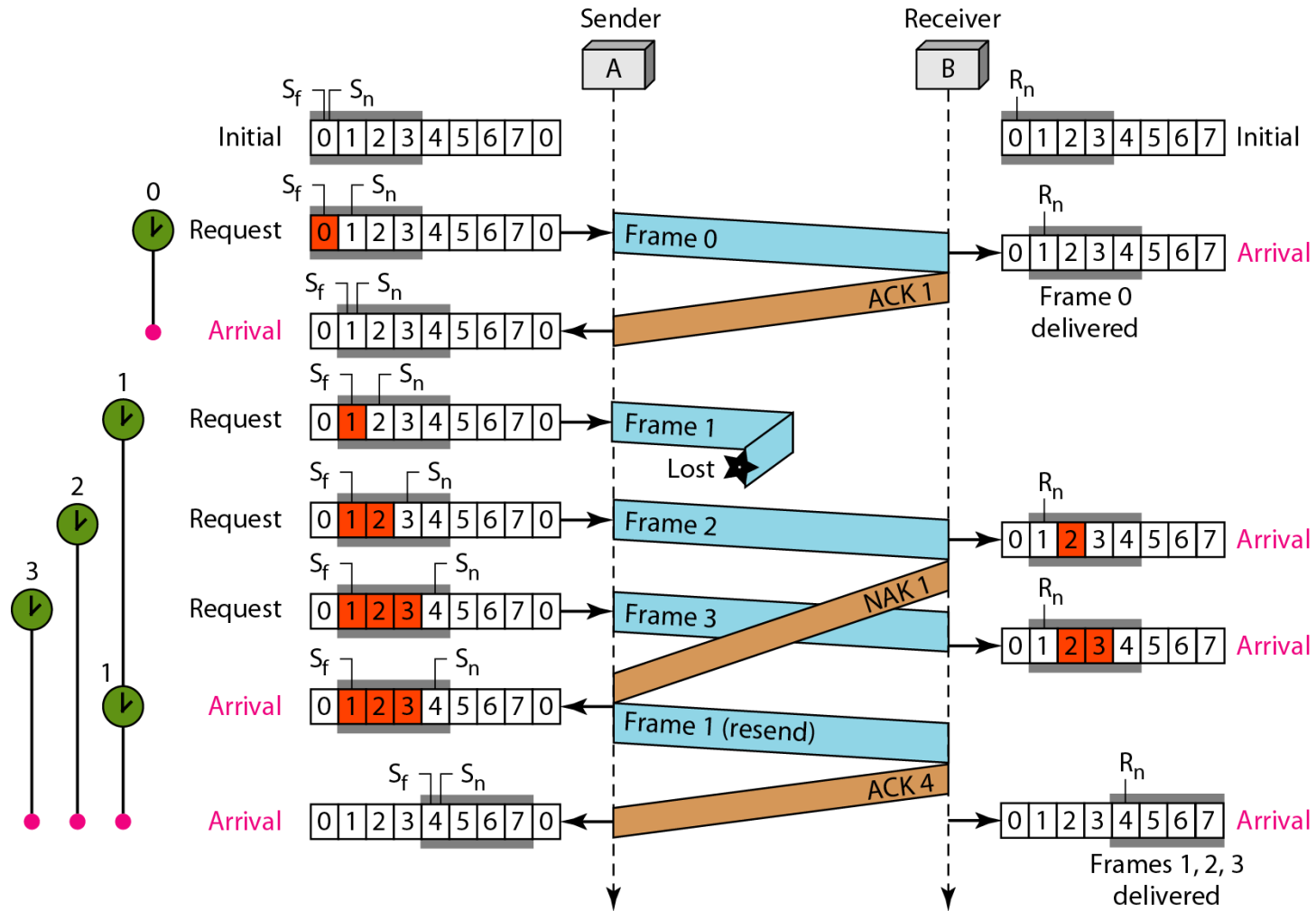
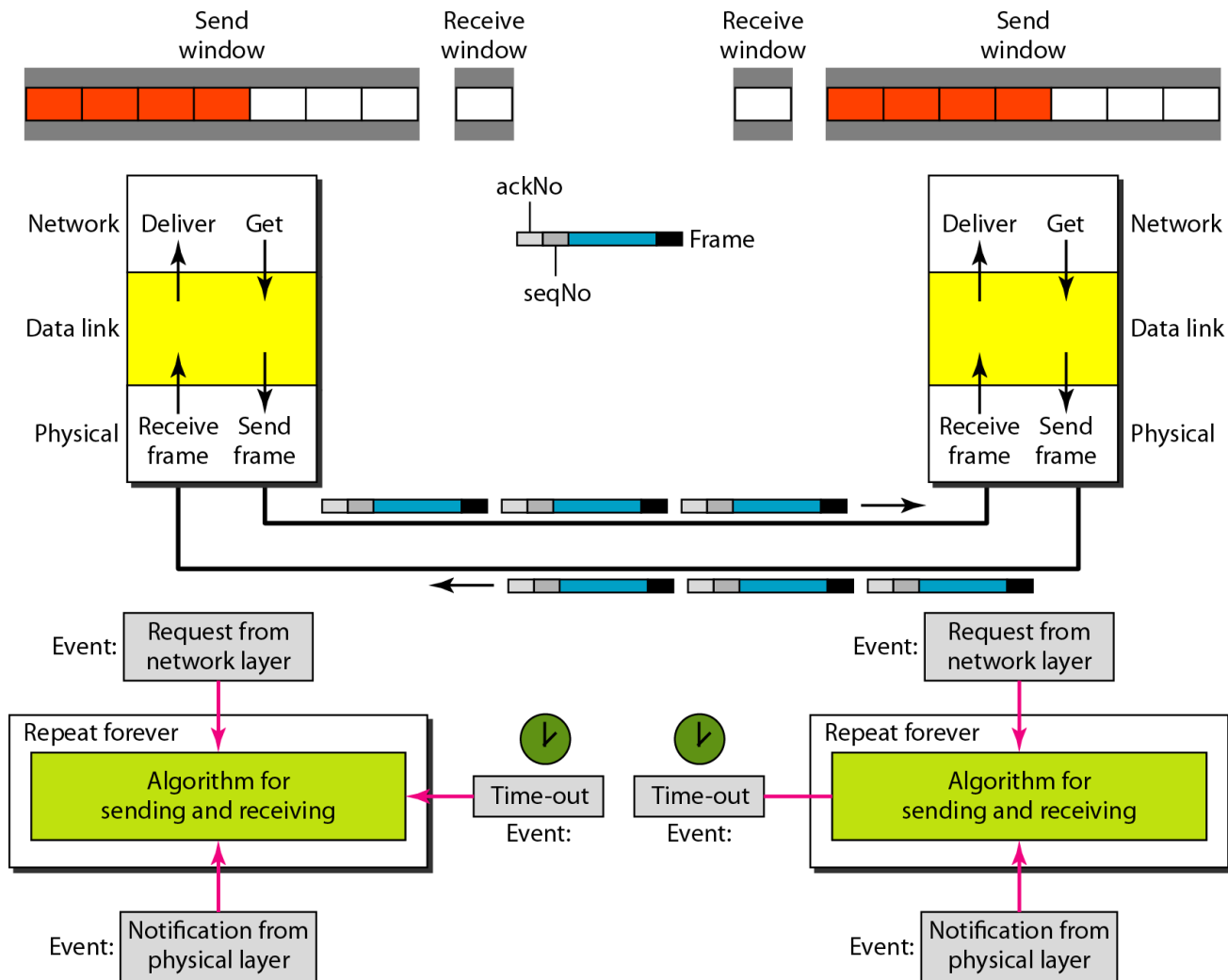


Figure 11.24 *Design of piggybacking in Go-Back-N ARQ*



11-6 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.

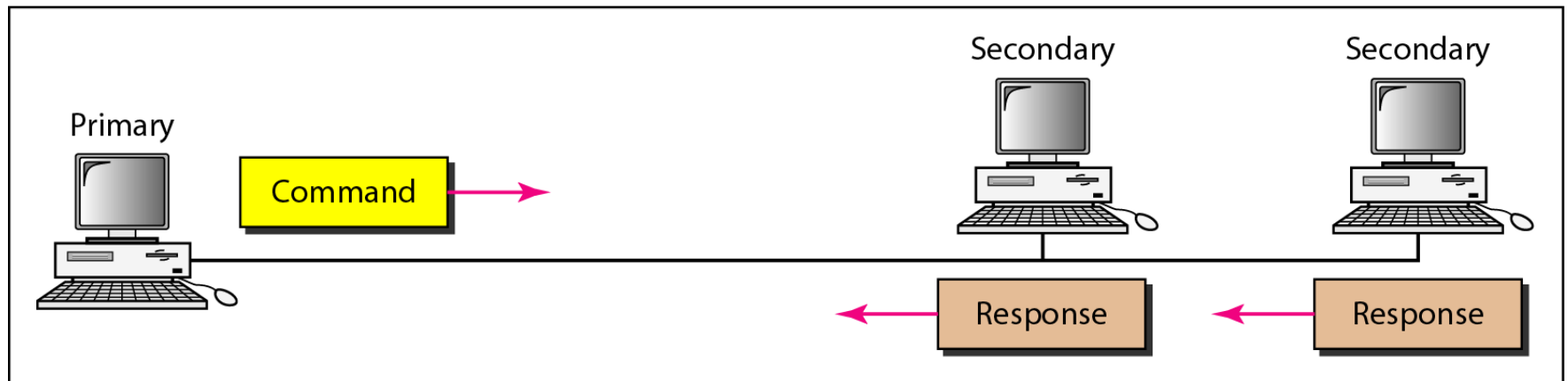
Two type of Transfer Modes for using HDLC in different configuration -

- Normal Response mode
- Asynchronous Balanced mode

Figure 11.25 *Normal response mode*



a. Point-to-point



b. Multipoint

Figure 11.26 *Asynchronous balanced mode*

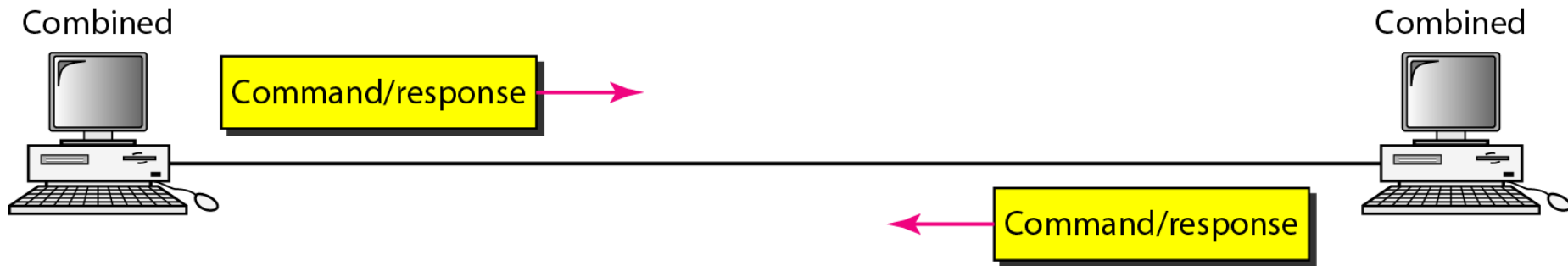


Figure 11.27 *HDLC frames (to support all the options possible in modes)*

Information frames (user data and control information),
Supervisory frames (control information) and Unnumbered
frames (system/link management)
Frame Check Sequence (FCS) - CRC

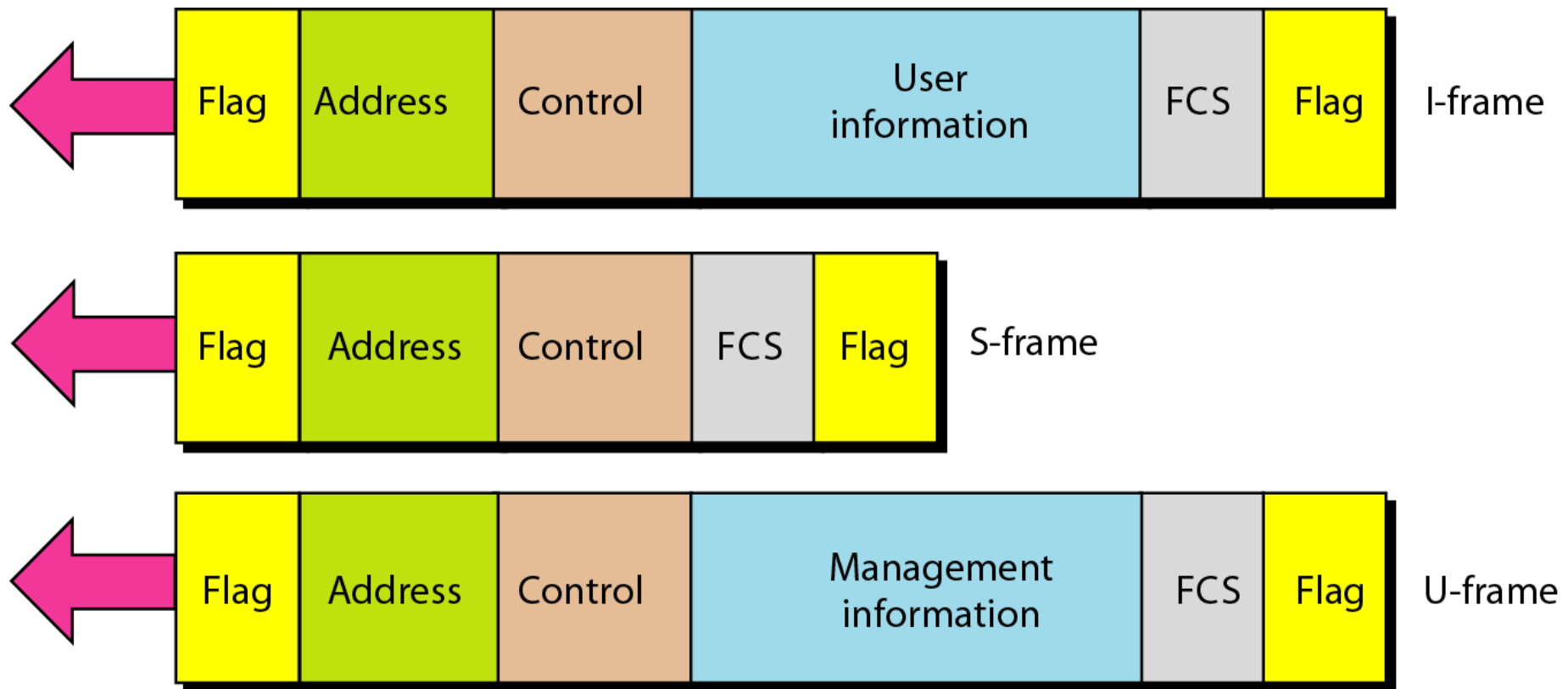
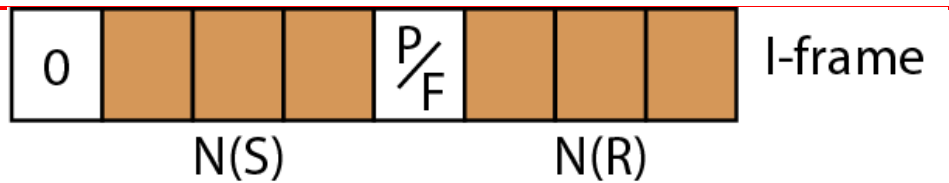
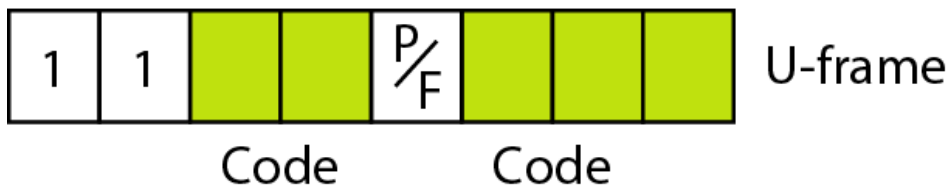
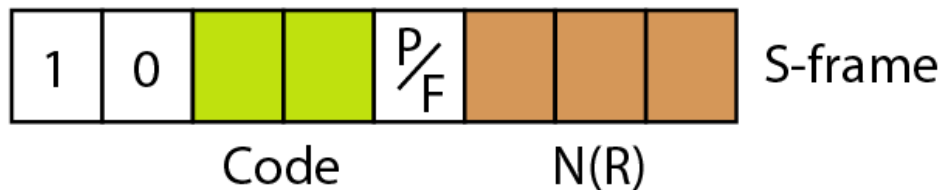


Figure 11.28 *Control field format for the different frame types*



P- primary to secondary
F- secondary to primary



S-frame types-

- Receive ready (RR) - 00
- Receive not ready (RNR) - 10
- Reject (REJ) - for Go-Back-N - 01
- Selective Reject (SREJ) - for Selective Repeat - 11

Table 11.1 *U-frame control command and response*

<i>Code</i>	<i>Command</i>	<i>Response</i>	<i>Meaning</i>
00 001	SNRM		Set normal response mode
11 011	SNRME		Set normal response mode, extended
11 100	SABM	DM	Set asynchronous balanced mode or disconnect mode
11 110	SABME		Set asynchronous balanced mode, extended
00 000	UI	UI	Unnumbered information
00 110		UA	Unnumbered acknowledgment
00 010	DISC	RD	Disconnect or request disconnect
10 000	SIM	RIM	Set initialization mode or request information mode
00 100	UP		Unnumbered poll
11 001	RSET		Reset
11 101	XID	XID	Exchange ID
10 001	FRMR	FRMR	Frame reject

Figure 11.29 *Example of connection and disconnection*

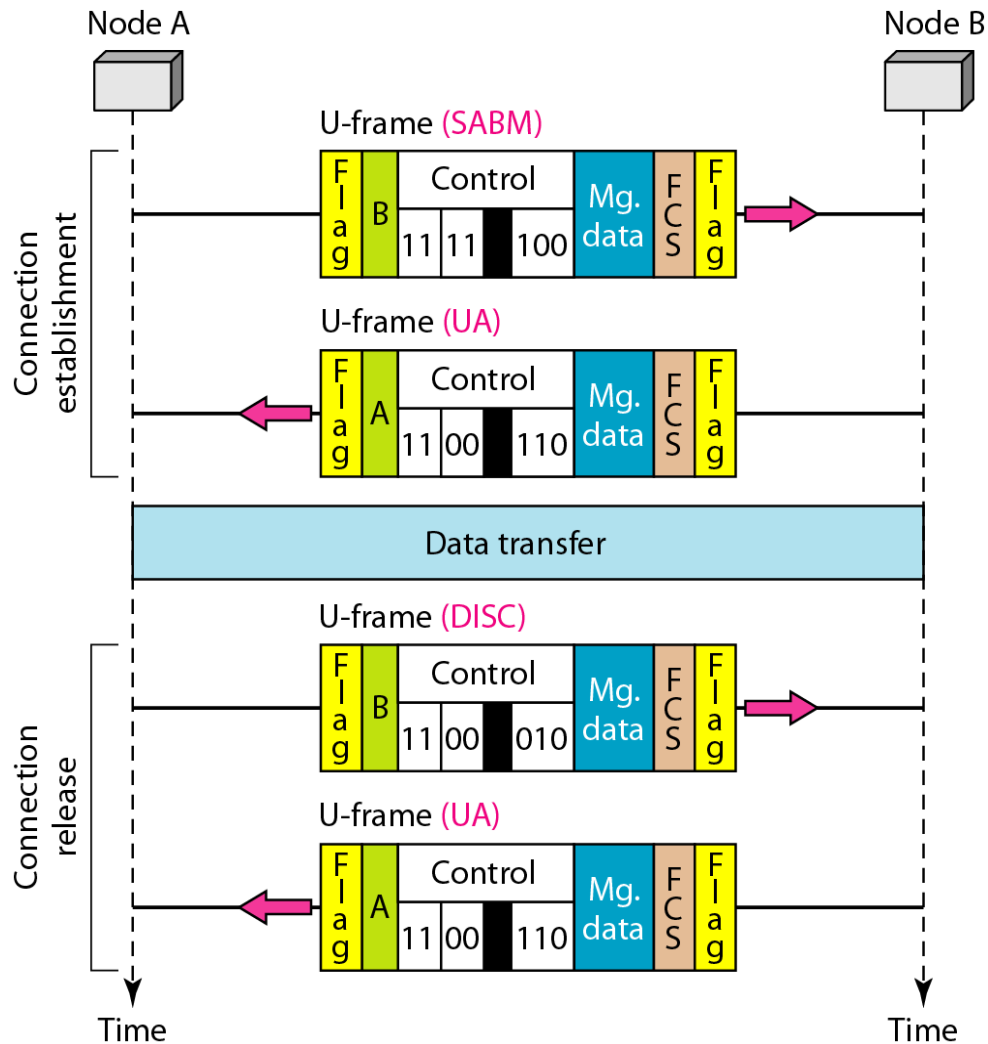


Figure 11.30 *Example of piggybacking without error*

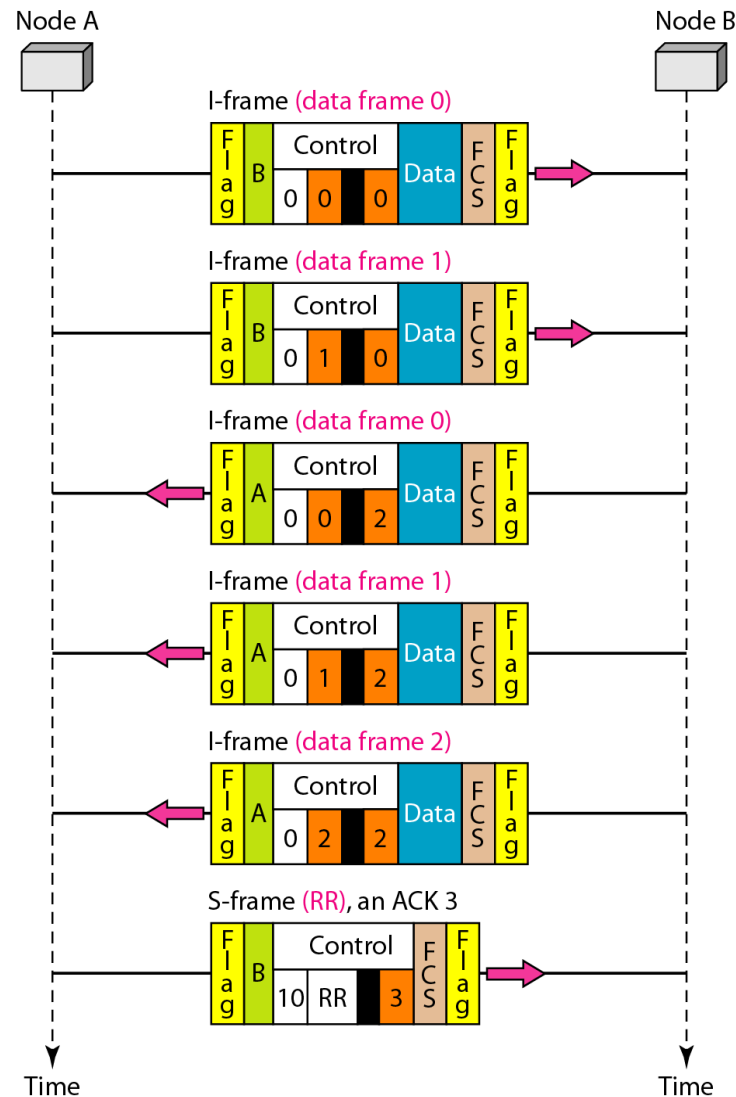
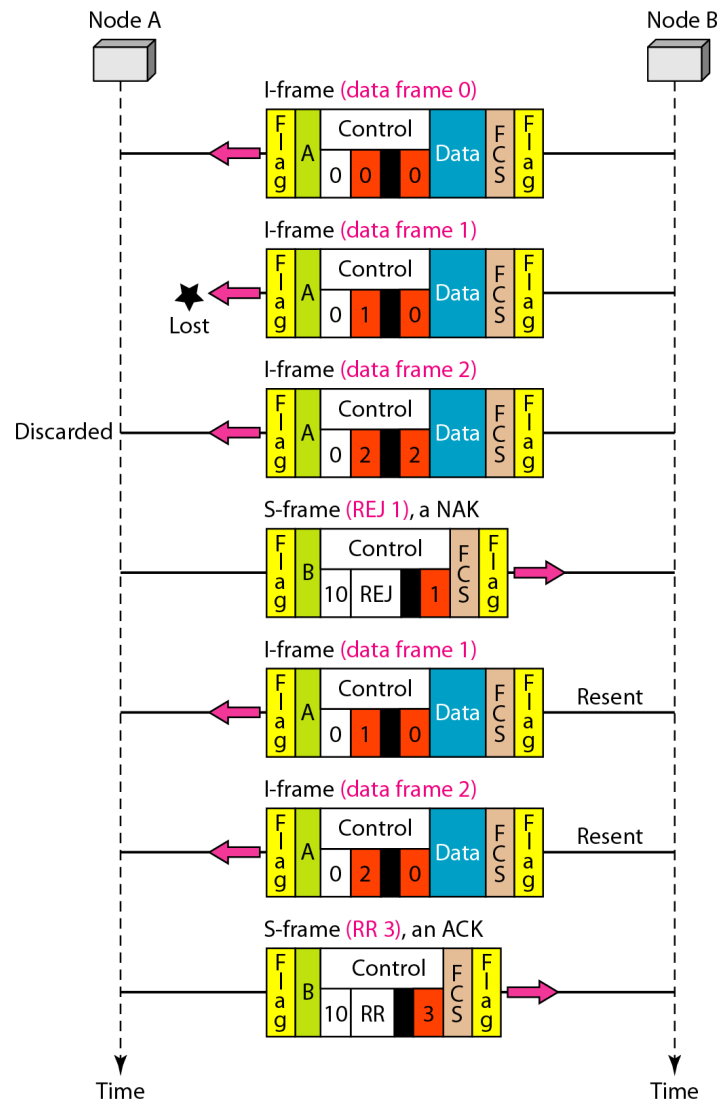


Figure 11.31 *Example of piggybacking with error*



11-7 POINT-TO-POINT PROTOCOL

*Although HDLC is a general protocol that can be used for both point-to-point and multipoint configurations, one of the most common protocols for point-to-point access is the **Point-to-Point Protocol (PPP)**. PPP is a **byte-oriented** protocol.*

Framing

Transition Phases

Multiplexing

Multilink PPP

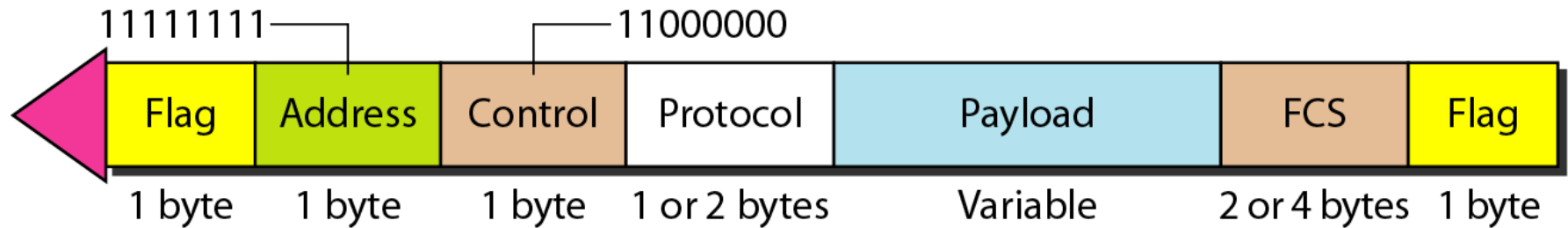


Services provided by PPP-

- Define format of frame
- Define negotiation for establishment of the link
- Define encapsulation of network layer data
- Define authentication between two devices
- Provide connection over multiple links
- PPP do not provide flow control and uses CRC for error control

**PPP is a byte-oriented protocol using
byte stuffing with the escape byte
01111101.**

Figure 11.32 *PPP frame format*



- Flag - 01111110
- Address is always 11111111, broadcasting address
- Control is 11000000, more like unnumbered frame in HDLC
- Protocol define the what is their in payload: data/information
- Data/Information - max 1500 bytes and padding is mandatory
- FCS- Frame check sequence - 2 or 4 byte CRC

Byte Stuffing in PPP with an escape character of 1 byte
01111101

Figure 11.33 *Transition phases*

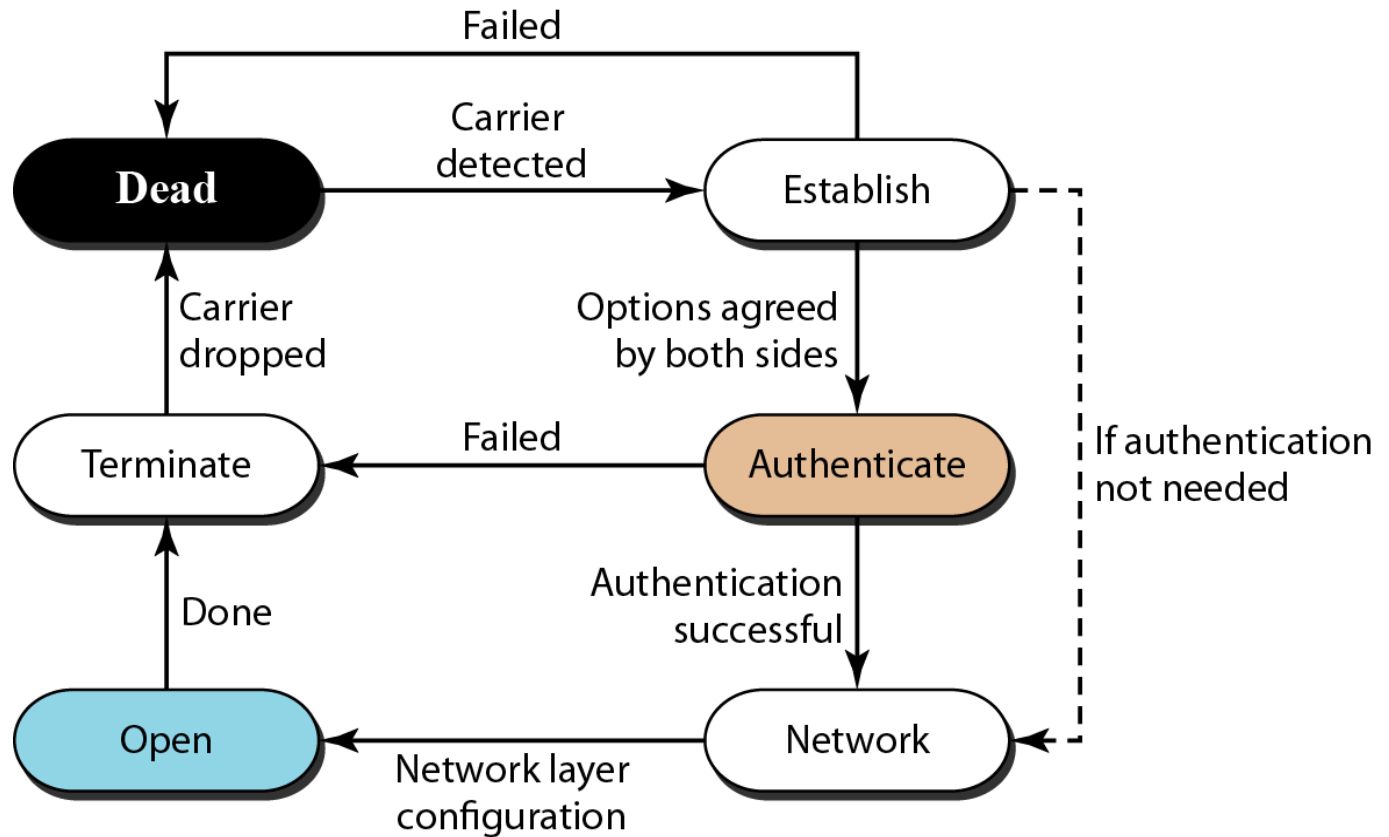


Figure 11.34 *Multiplexing in PPP*

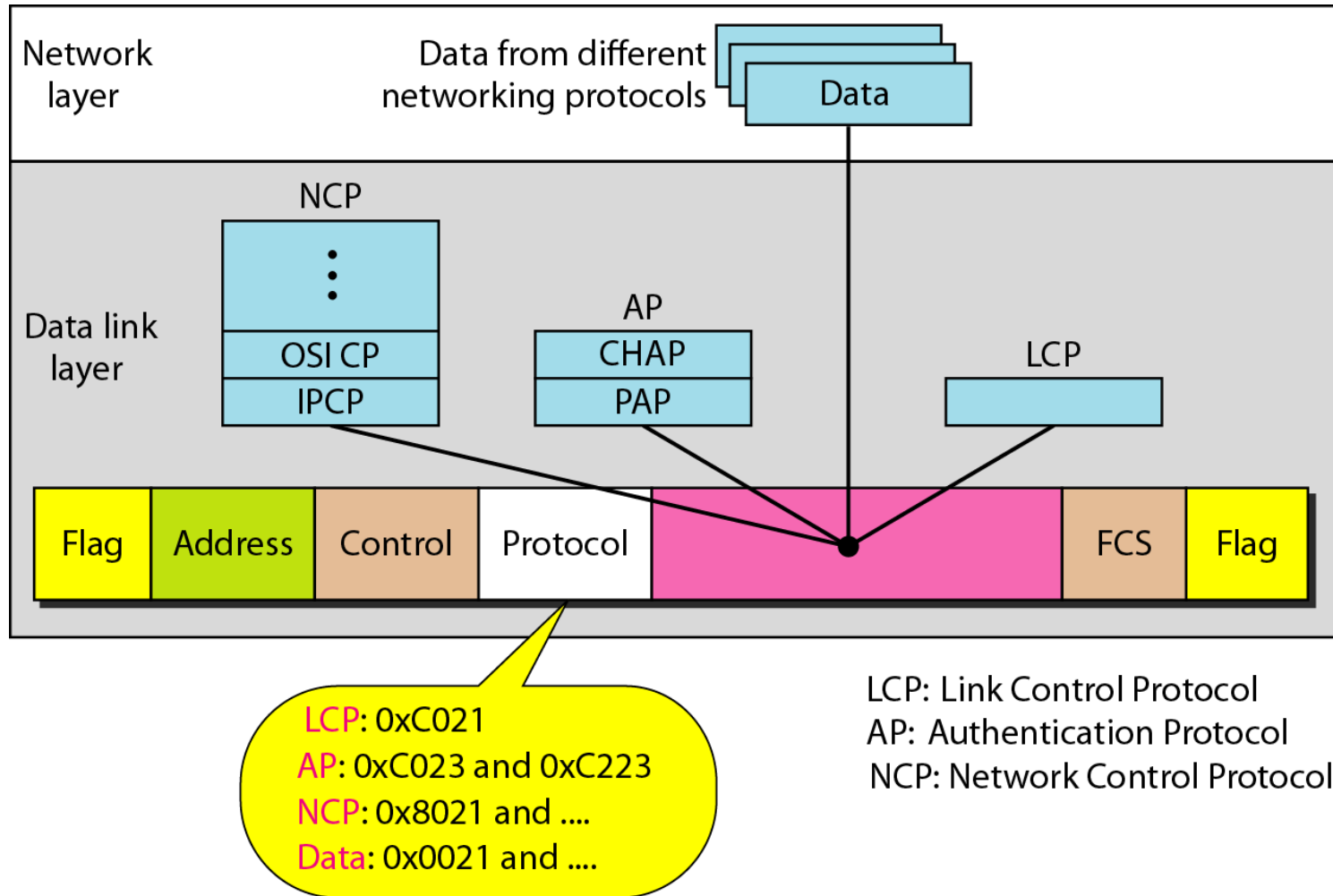


Figure 11.35 *LCP packet encapsulated in a frame*

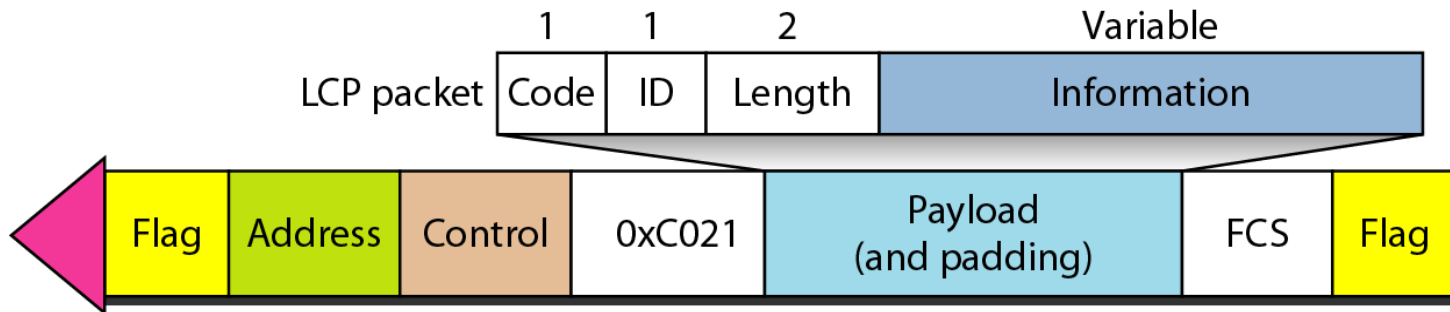


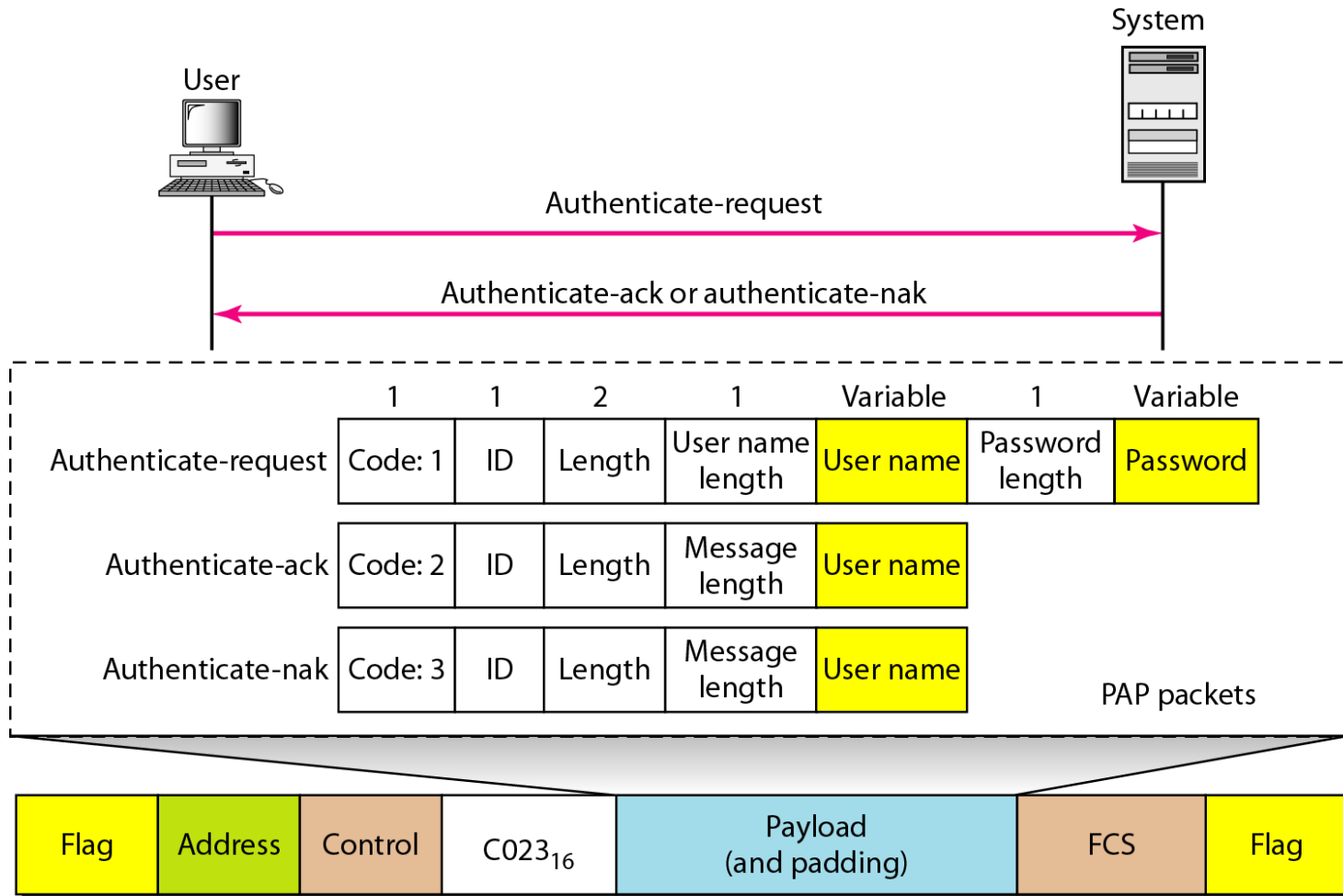
Table 11.2 *LCP packets*

<i>Code</i>	<i>Packet Type</i>	<i>Description</i>
0x01	Configure-request	Contains the list of proposed options and their values
0x02	Configure-ack	Accepts all options proposed
0x03	Configure-nak	Announces that some options are not acceptable
0x04	Configure-reject	Announces that some options are not recognized
0x05	Terminate-request	Request to shut down the line
0x06	Terminate-ack	Accept the shutdown request
0x07	Code-reject	Announces an unknown code
0x08	Protocol-reject	Announces an unknown protocol
0x09	Echo-request	A type of hello message to check if the other end is alive
0x0A	Echo-reply	The response to the echo-request message
0x0B	Discard-request	A request to discard the packet

Table 11.3 *Common options*

<i>Option</i>	<i>Default</i>
Maximum receive unit (payload field size)	1500
Authentication protocol	None
Protocol field compression	Off
Address and control field compression	Off

PAP (Password Authentication Protocol) packets encapsulated in a PPP frame



CHAP (Challenge Handshake Authentication Protocol) packets encapsulated in a PPP frame

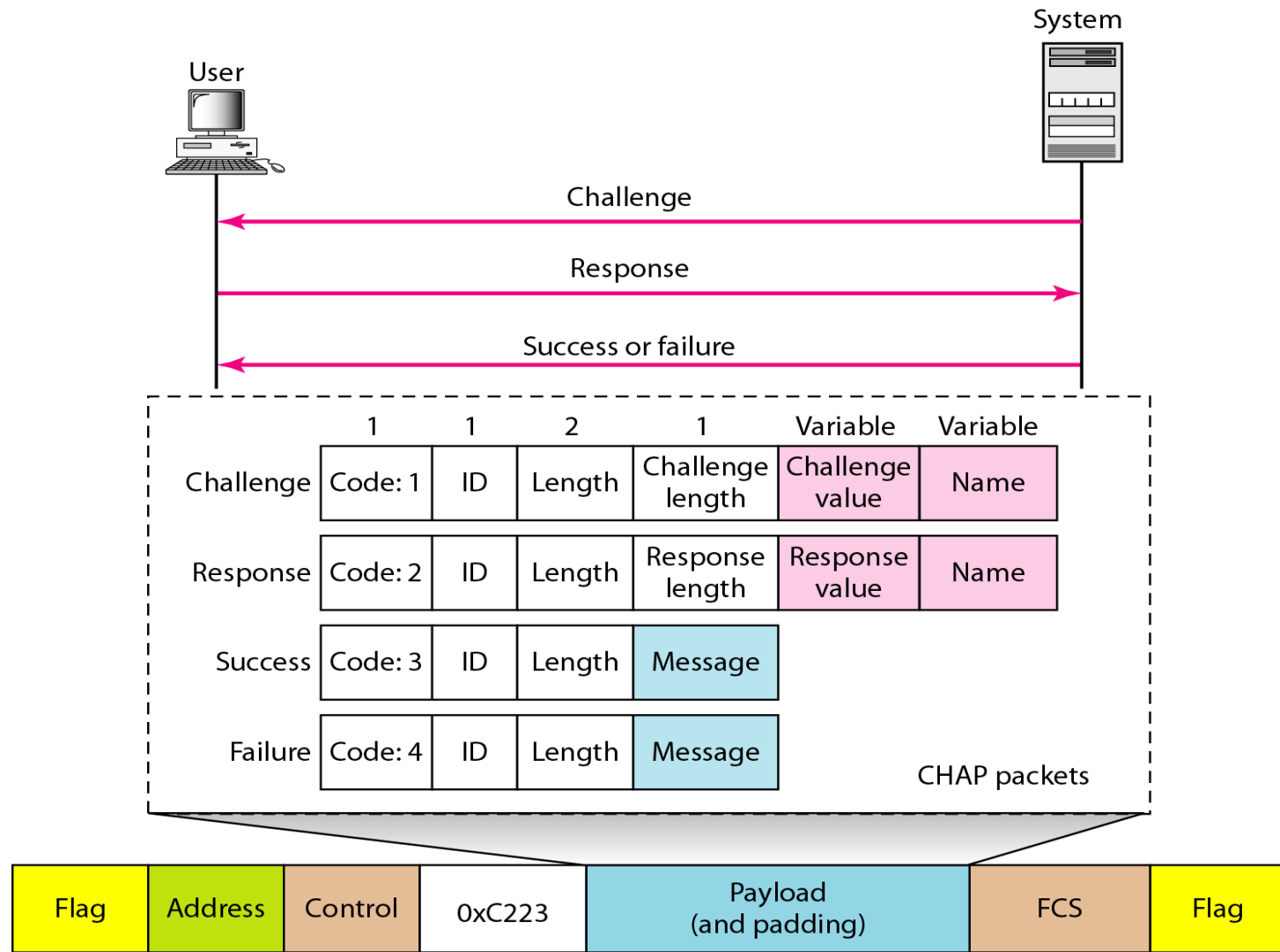


Figure 11.38 *IPCP packet encapsulated in PPP frame*

- Various NCP (Network Control Protocols) - Internet Protocol Control Protocol (IPCP), OSI, Xerox, AppleTalk, Novel, etc..

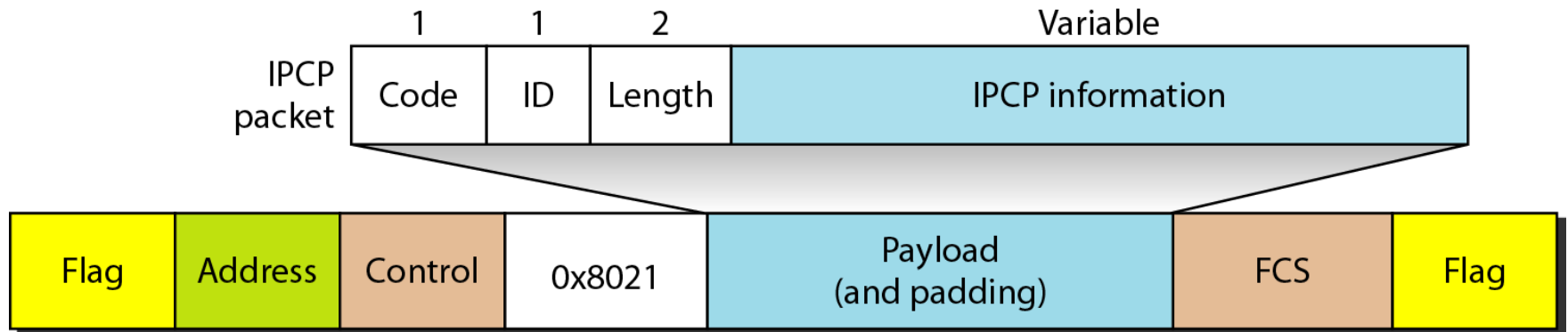


Table 11.4 *Code value for IPCP packets*

<i>Code</i>	<i>IPCP Packet</i>
0x01	Configure-request
0x02	Configure-ack
0x03	Configure-nak
0x04	Configure-reject
0x05	Terminate-request
0x06	Terminate-ack
0x07	Code-reject

Figure 11.39 *IP datagram encapsulated in a PPP frame*

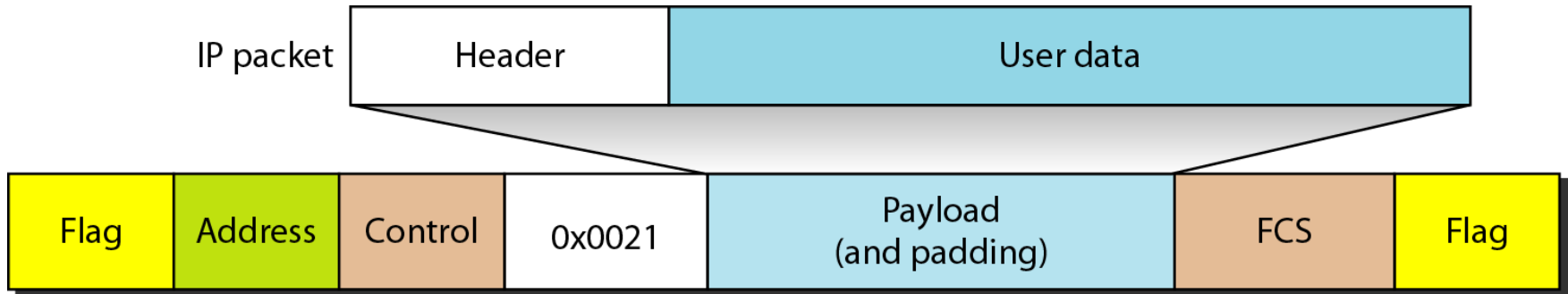


Figure 11.41 *An example*

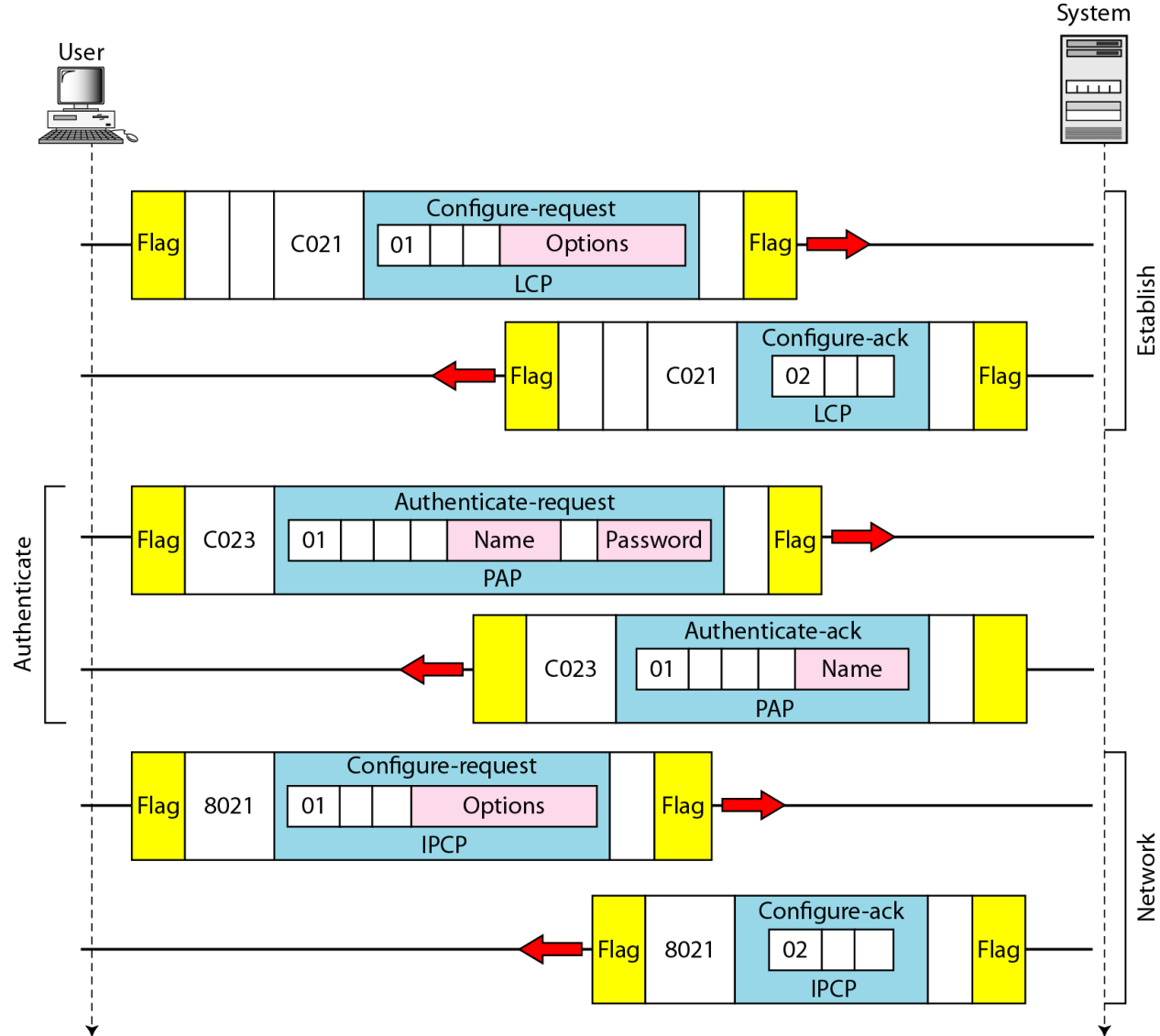


Figure 11.41 *An example (continued)*

