# DATA LINK CONTROL

# Data Link Layer Design Issues

- Services Provided to the Network Layer
- Framing
- Error Control
- Flow Control

# 11-1 FRAMING

***The data link layer needs to pack bits into frames, so that each frame is distinguishable from another. Our postal system practices a type of framing. The simple act of inserting a letter into an envelope separates one piece of information from another; the envelope serves as the delimiter.***

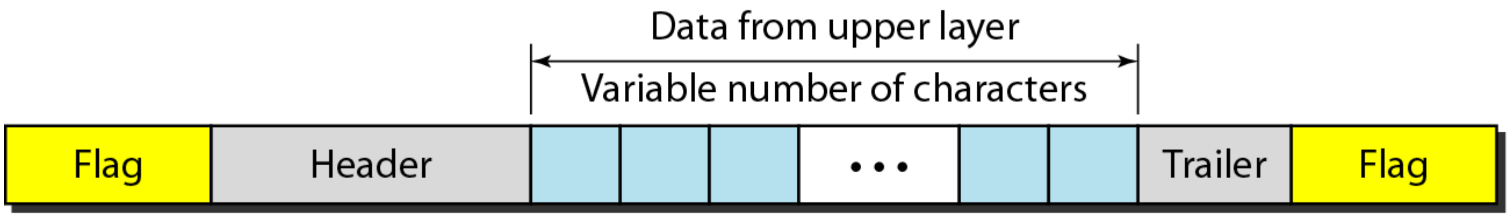
***Topics discussed in this section:***

**Fixed-Size Framing**
**Variable-Size Framing**

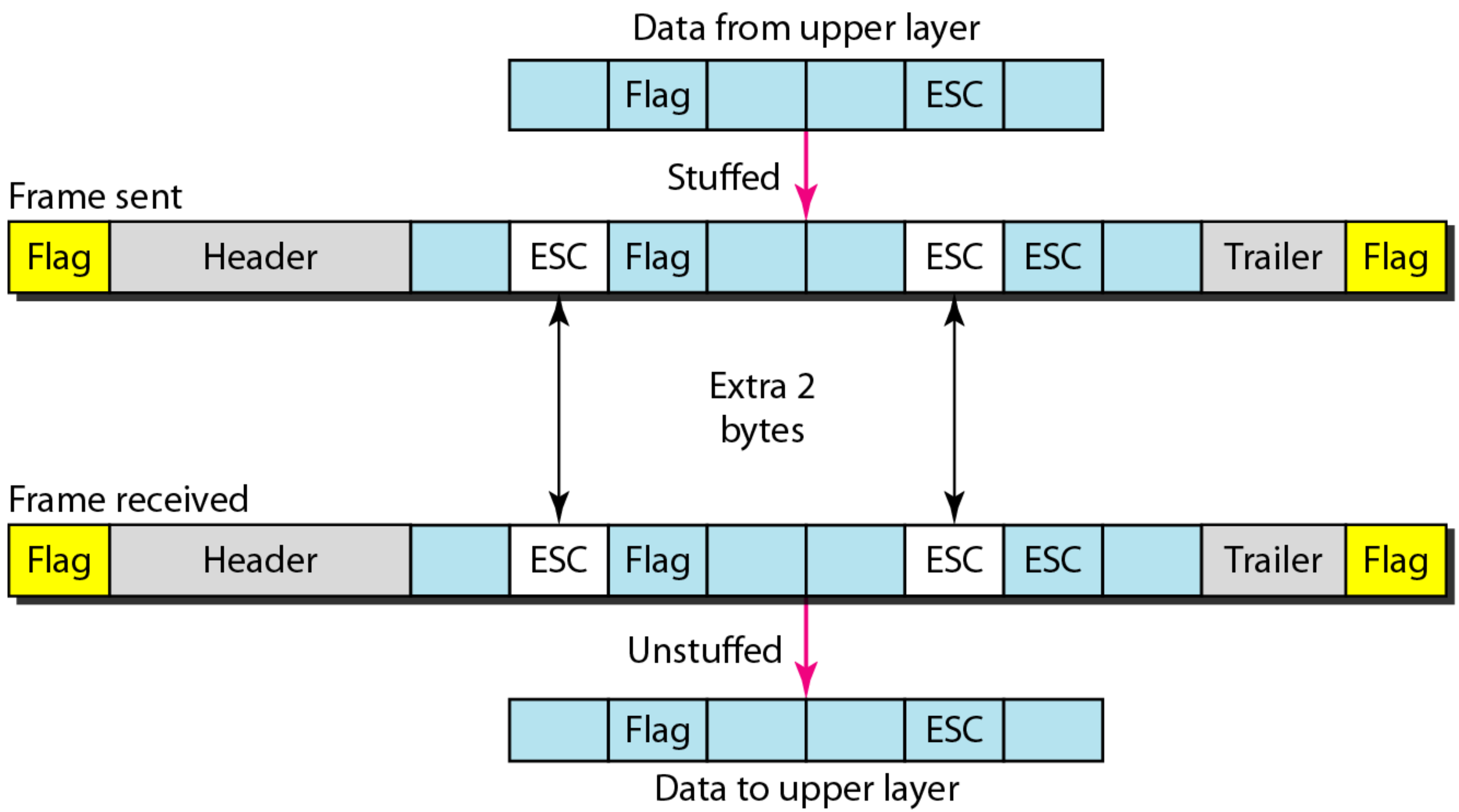
# Types of Framing

- Fixed Size Framing
- Variable Size Framing---
- Character oriented protocols, and
- Bit oriented Protocols.

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



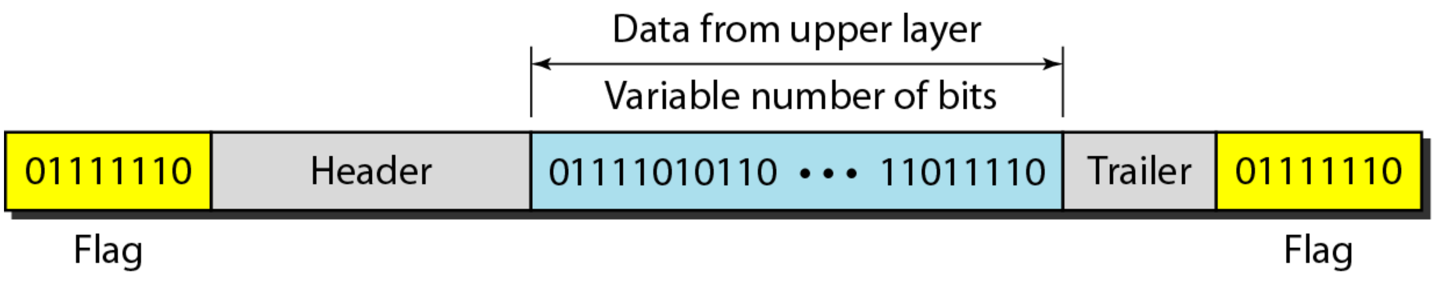
## Figure 11.2 *Byte stuffing and unstuffing*



***Note***

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

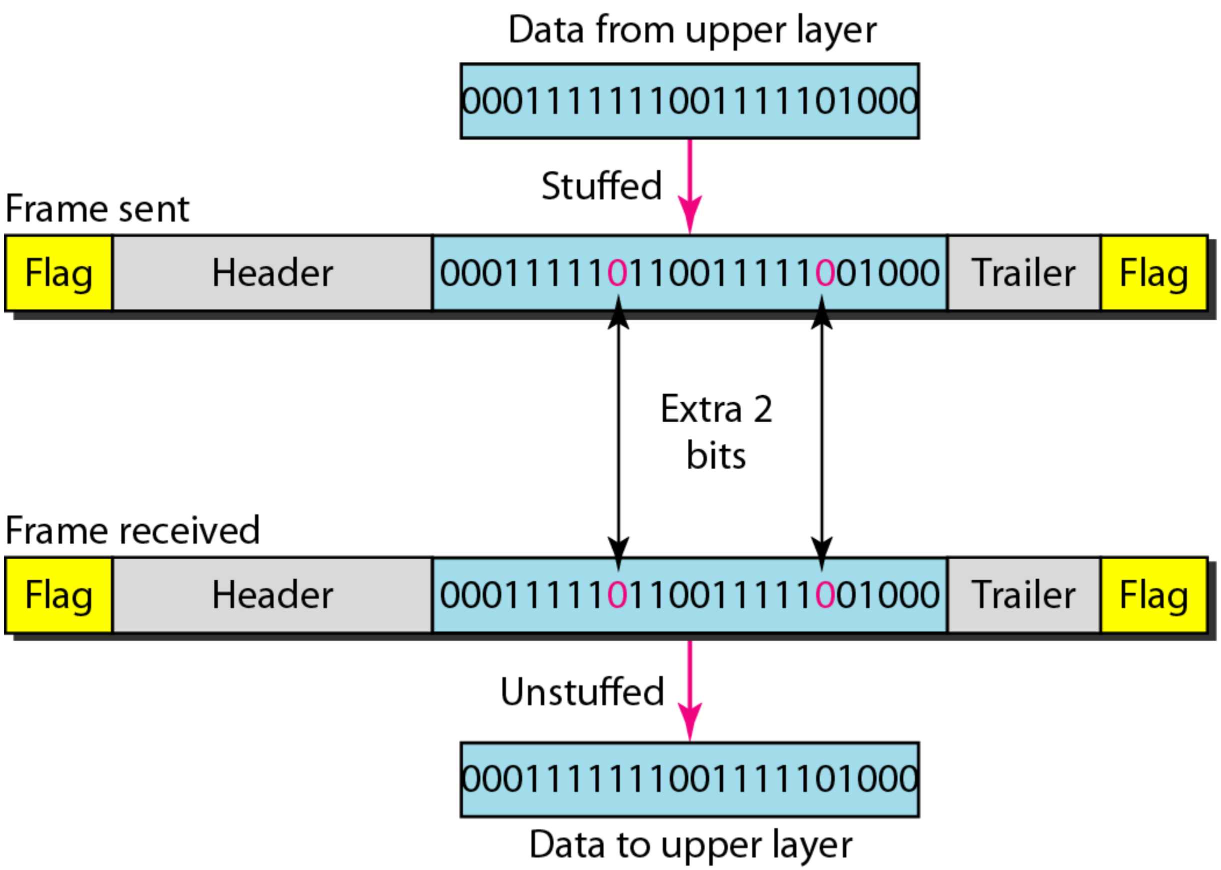
## Figure 11.3 *A frame in a bit-oriented protocol*



***Note***

**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 0111110 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.***

***Topics discussed in this section:***

**Flow Control**
**Error Control**

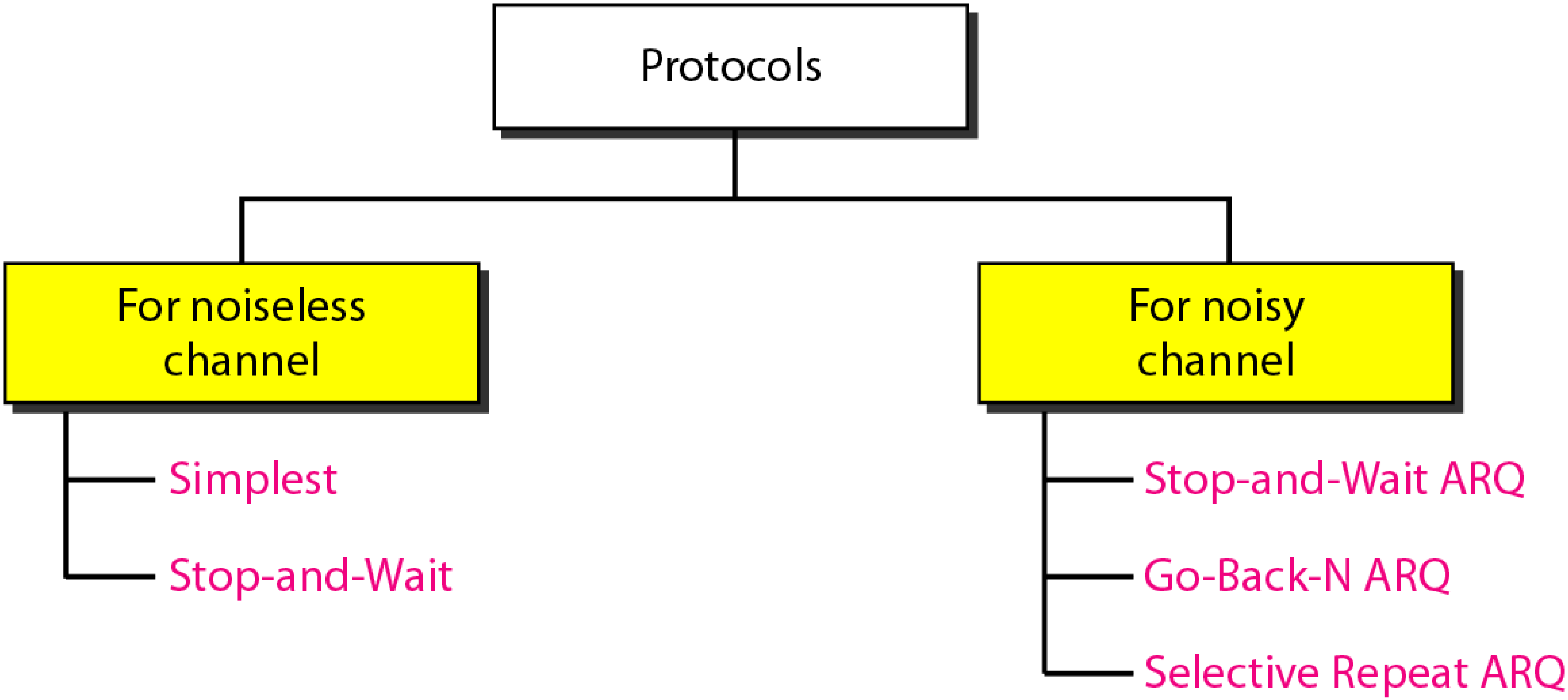
***Note***

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

***Note***

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

## Figure 11.5 *Taxonomy of protocols discussed in this chapter*



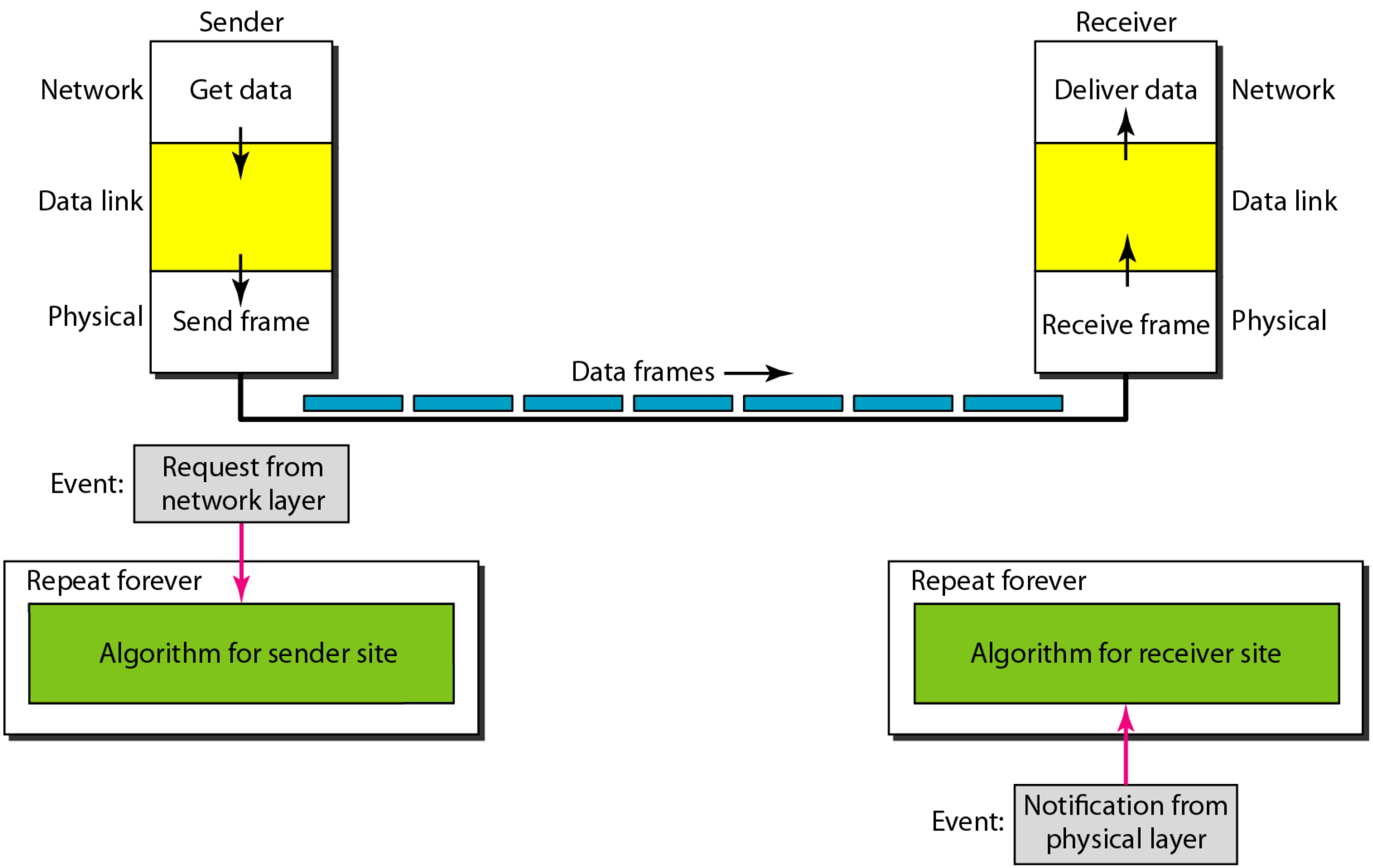
# 11-4 NOISELESS CHANNELS

***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.***

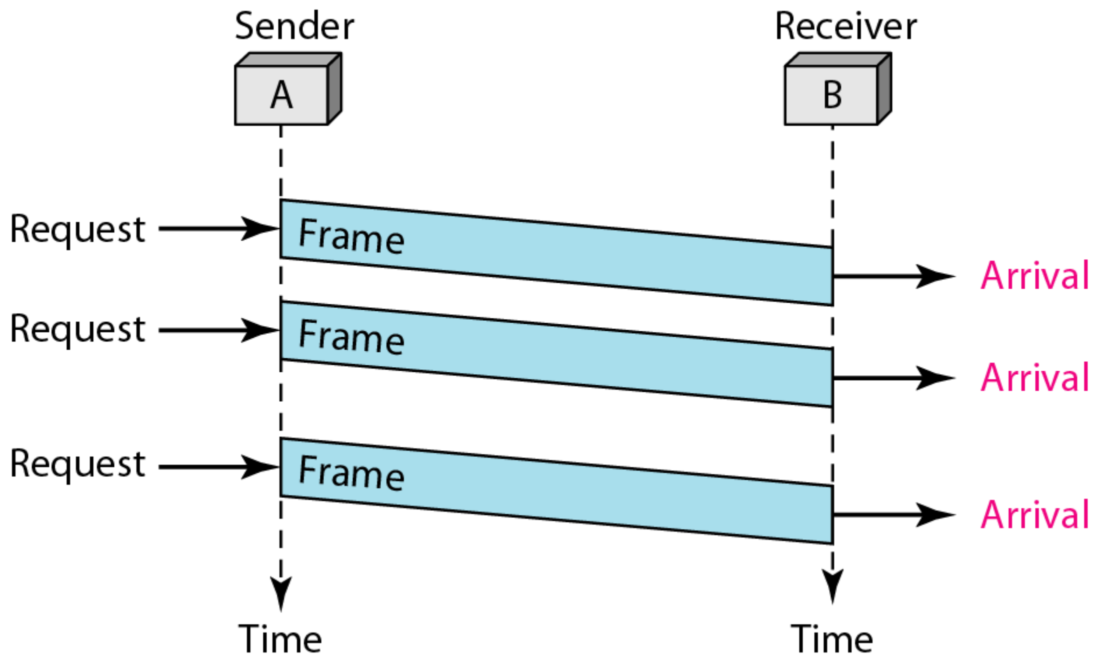
***Topics discussed in this section:***

**Simplest Protocol**
**Stop-and-Wait Protocol**

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



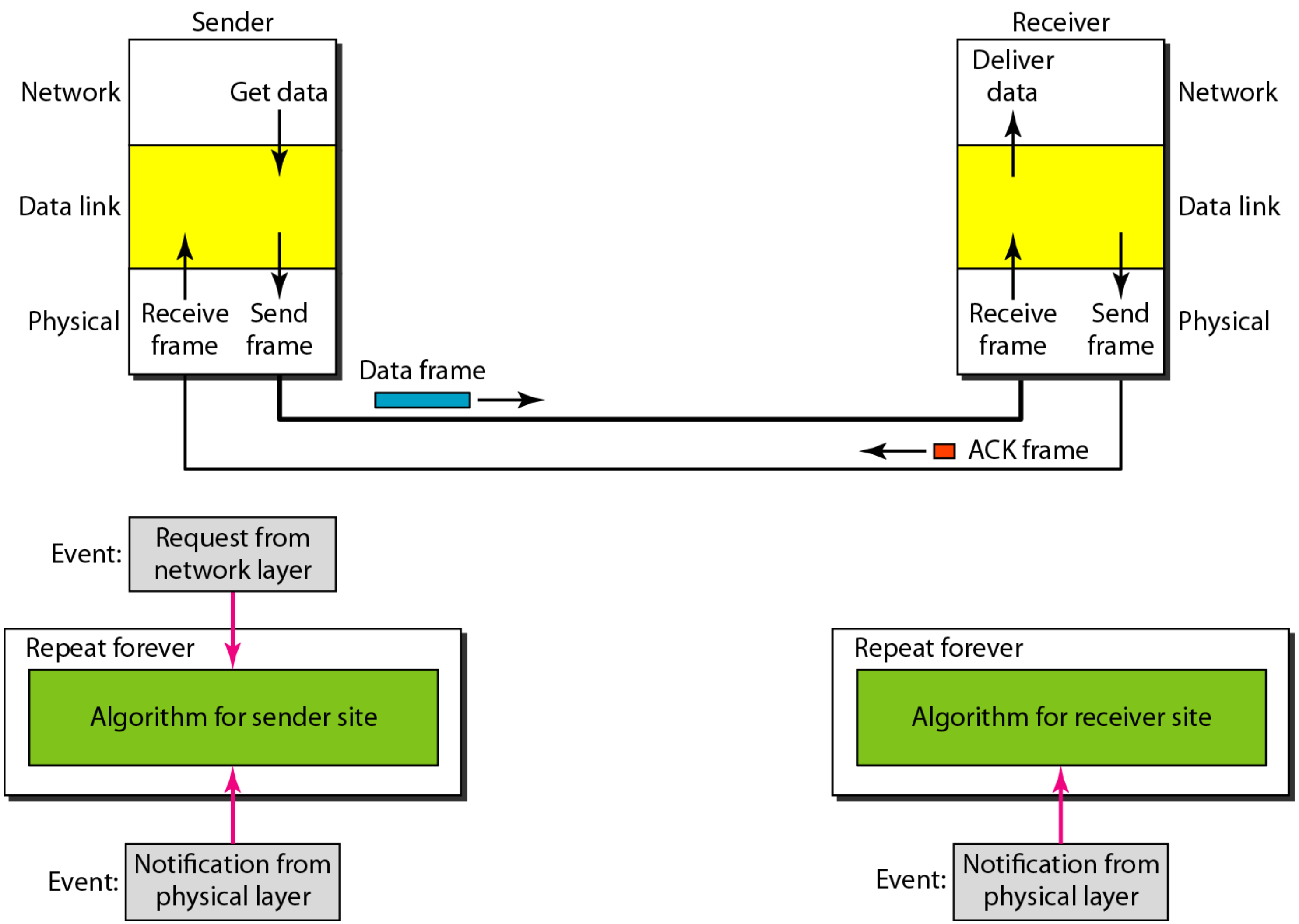
## **Figure 11.7** *Flow diagram for Example 11.1*



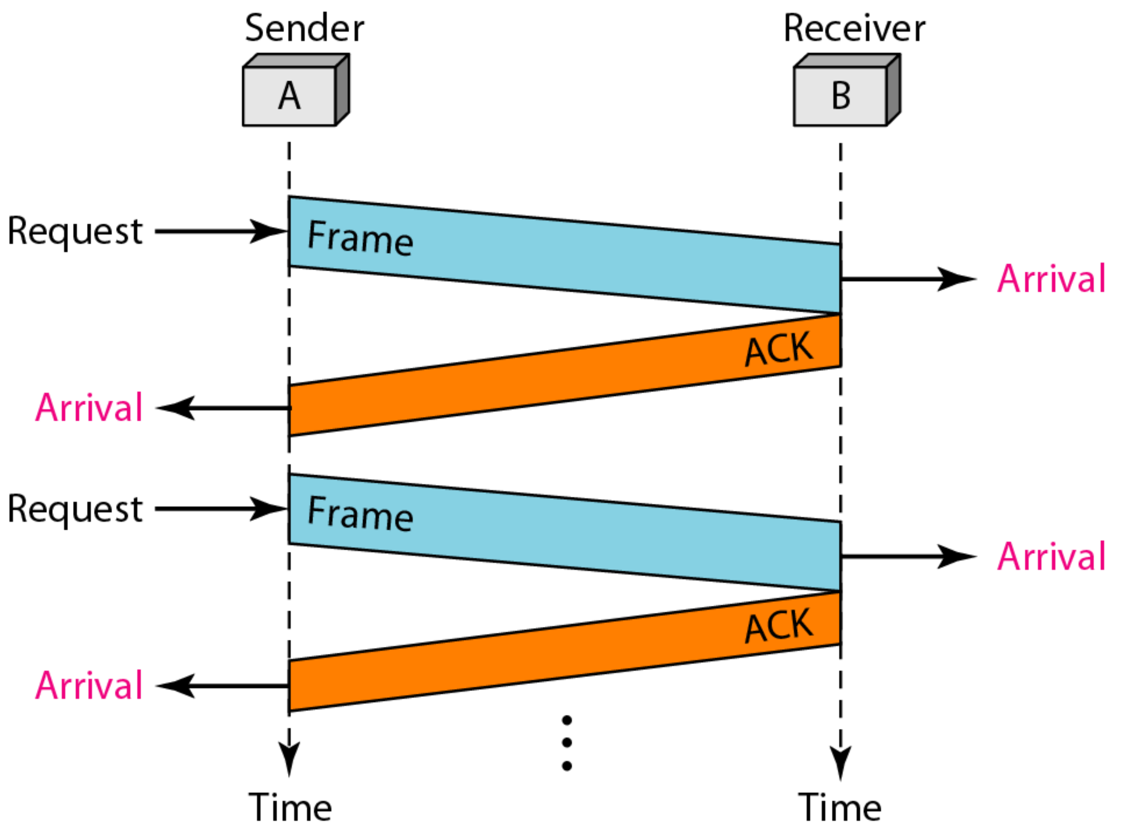
# *Stop-and-Wait Protocol*

- Sender sends one frame, stops until it gets confirmation from receiver.

## **Figure 11.8** *Design of Stop-and-Wait Protocol*



## 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.***

***Topics discussed in this section:***

**Stop-and-Wait Automatic Repeat Request**
**Go-Back-N Automatic Repeat Request**
**Selective Repeat Automatic Repeat Request**

***Note***

**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.**

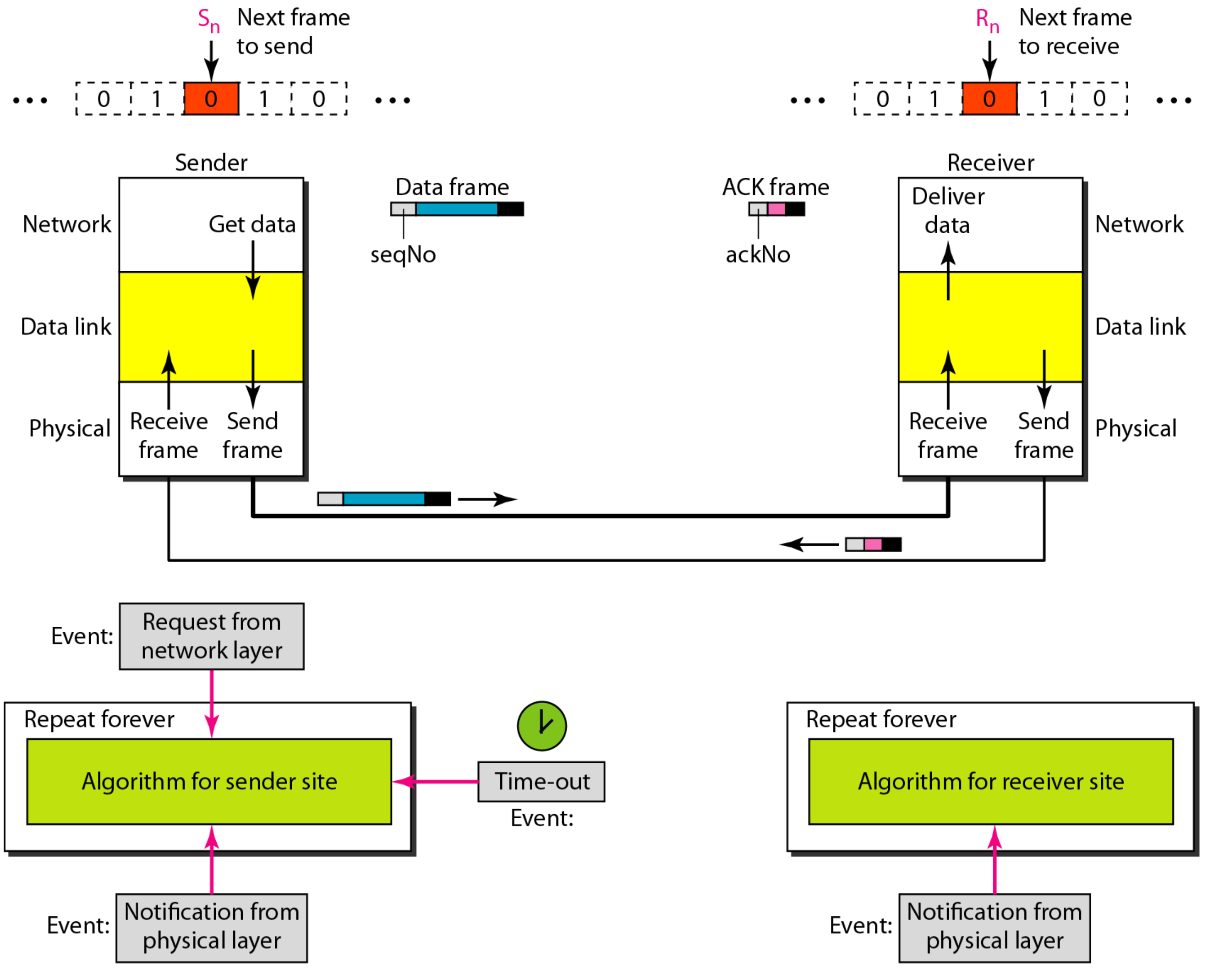
***Note***

**In Stop-and-Wait ARQ, we use sequence numbers to number the frames. The sequence numbers are based on modulo-2 arithmetic.**

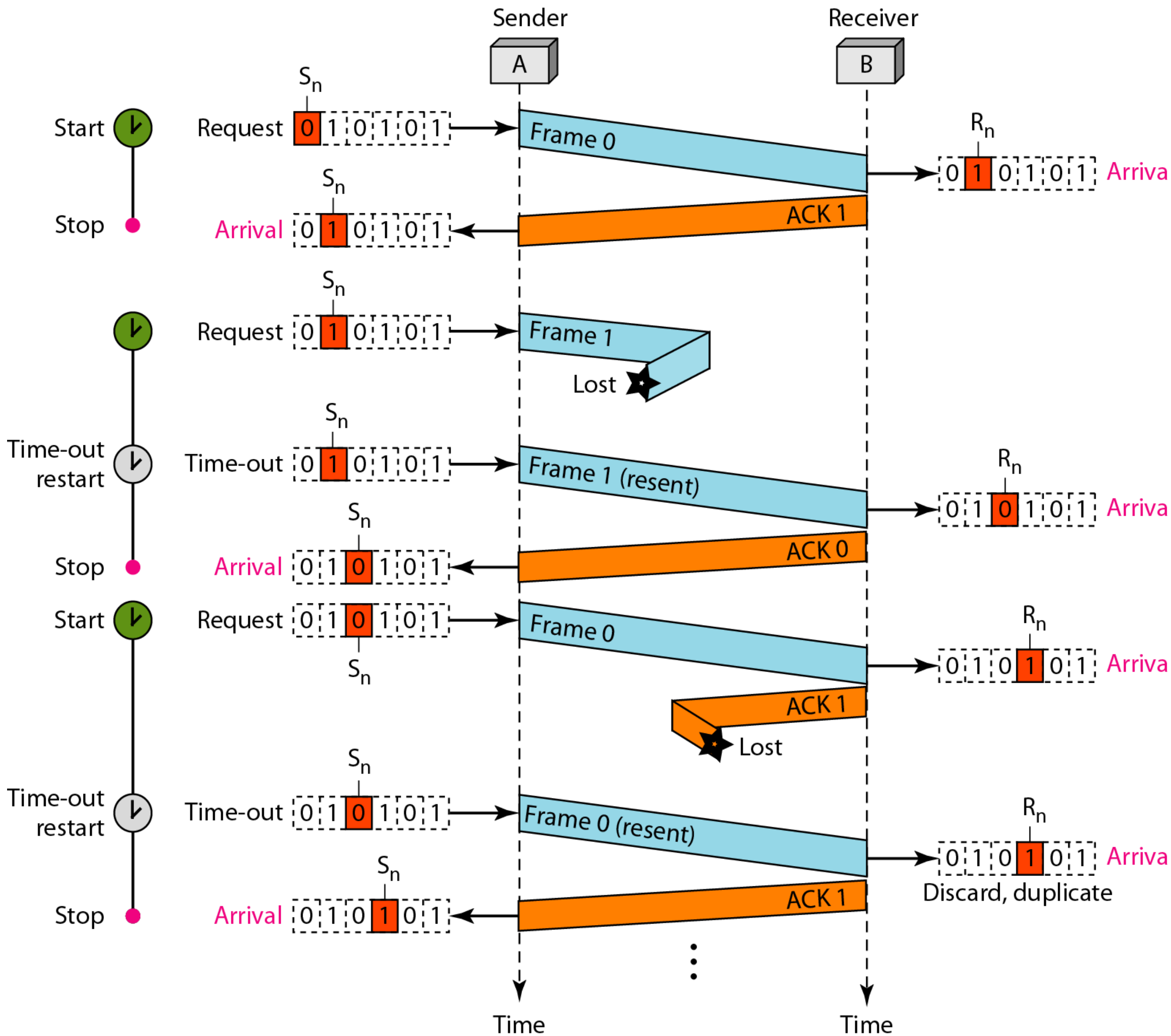
***Note***

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

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



## **Figure 11.11** *Flow diagram for Example 11.3*



# Disadvantage *Stop-and-Wait ARQ Protocol*

- Inefficient---if channel is thick and long
- Thick means high bandwidth
- Long means roundtrip delay
- Product of both is bandwidth delay.
- Bandwidth delay is number of bits we can send while waiting for news from 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**

$$\boxed{(1 \times 10^6) \times (20 \times 10^{-3}) = 20{,}000 \text{ bits}}$$

# Example 11.4 (continued)

***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.***

# Pipelining

- Task begins before end of first task.
- ***Stop-and-Wait ARQ does not use pipelining but other two techniques do.***
- ***This improves efficiency.***

# Go-Back-N Protocol

- This sends multiple frames before receiving acknowledgment from receiver.

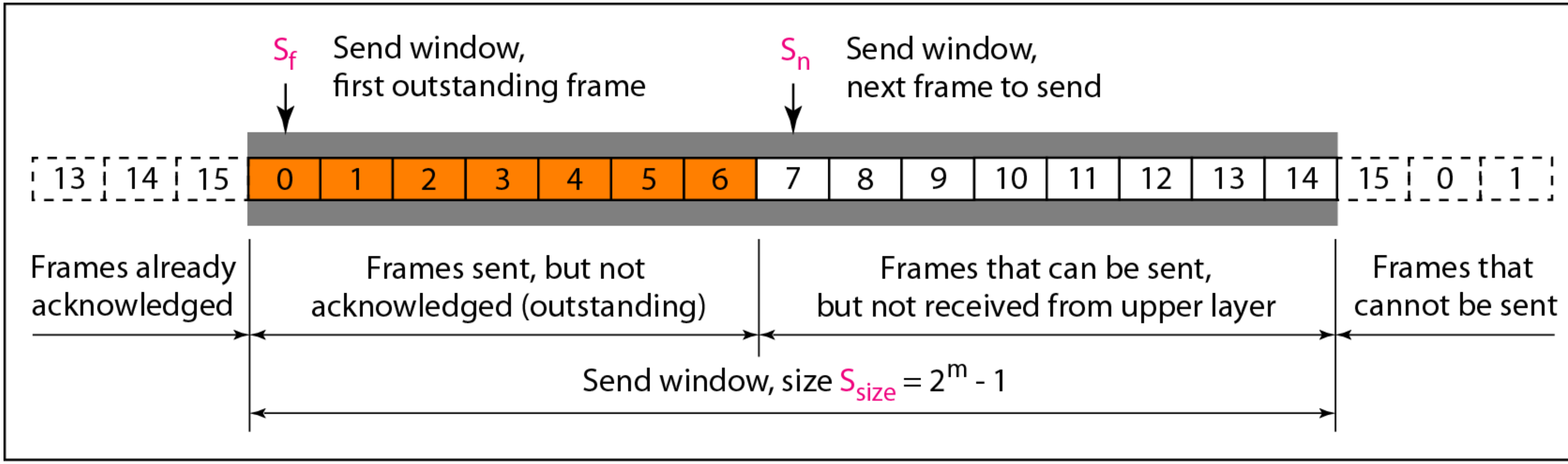
**Note**

**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.**

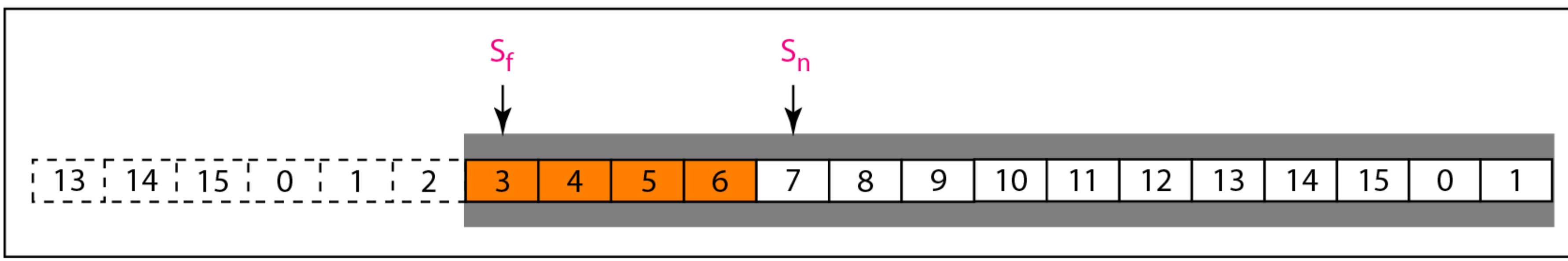
# Sliding Window

- Defines the range of sequence numbers that is concern of sender and receiver.
- The range which is concern of sender is called sender sliding window.
- The range which is concern of receiver is called receiver sliding window.

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




a. Send window before sliding




b. Send window after sliding

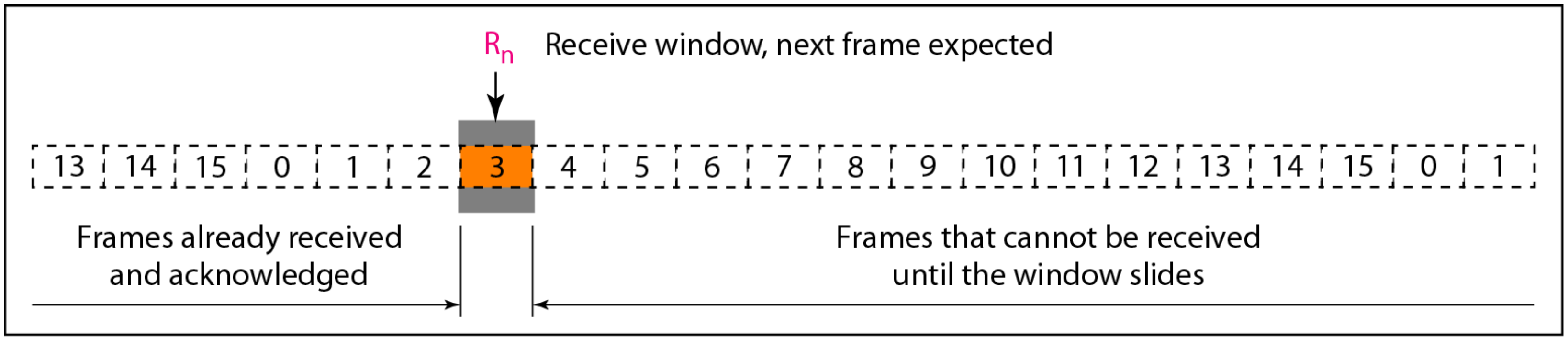
***Note***

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

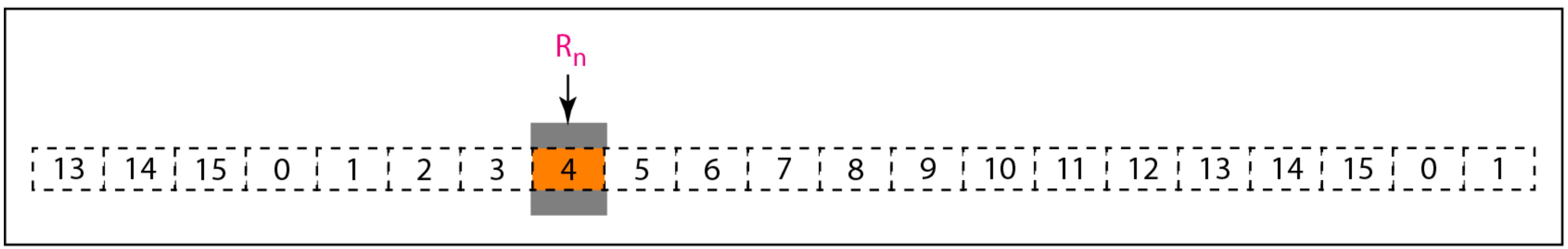
***Note***

**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

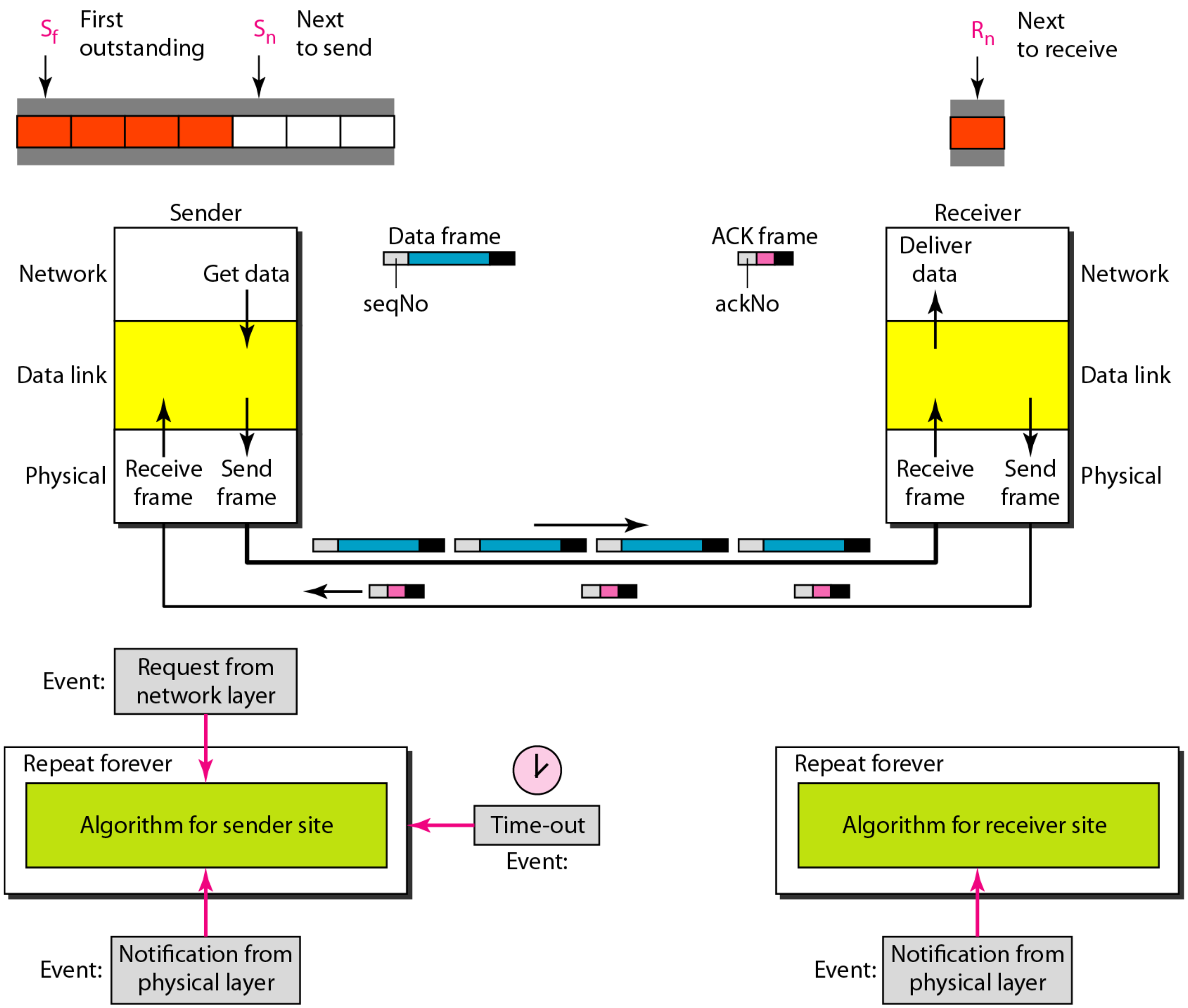



b. Window after sliding

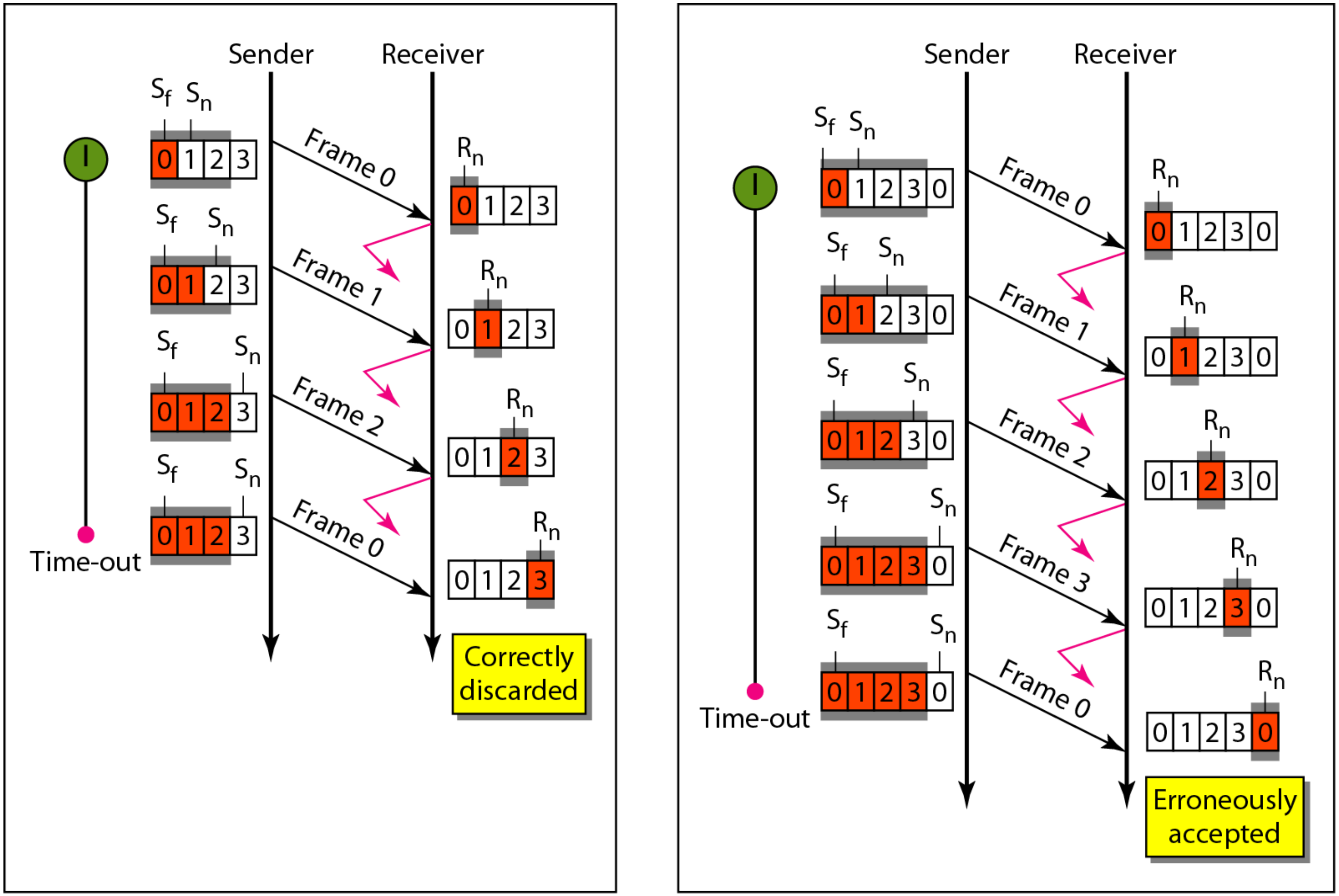
*Note*

**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*



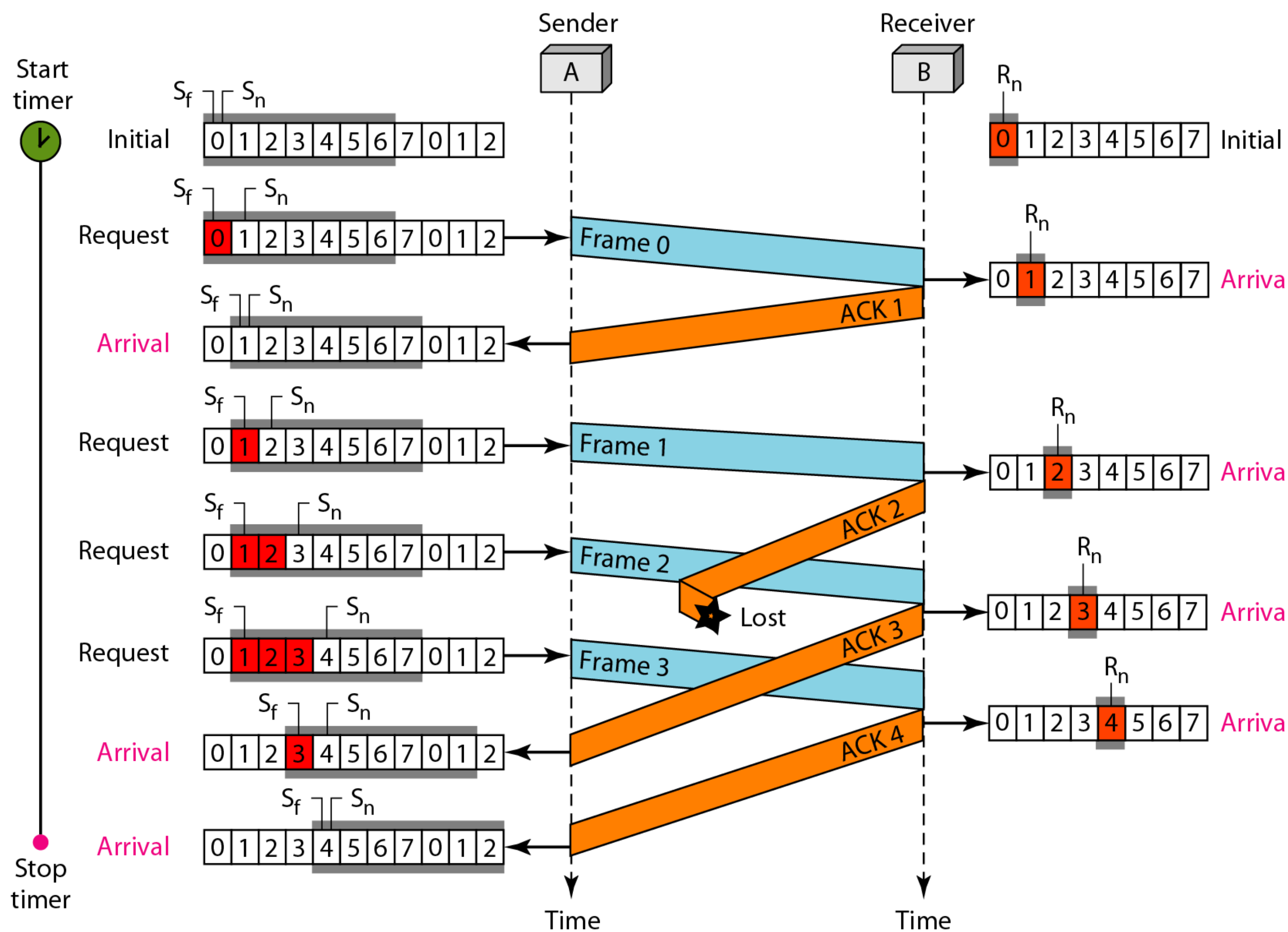

a. Window size $< 2^m$

b. Window size $= 2^m$

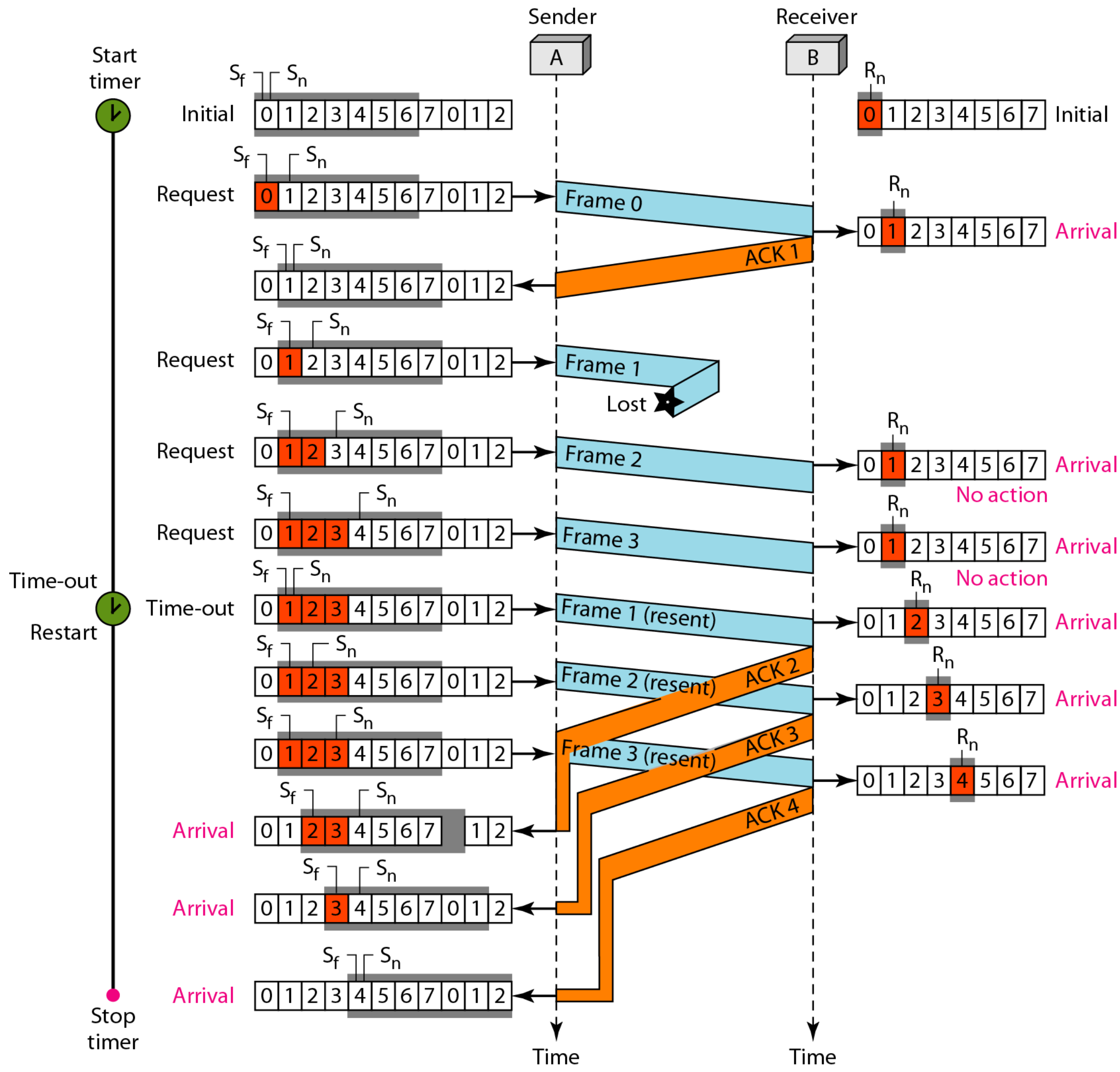
***Note***

**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.**

## **Figure 11.16** *Flow diagram for Example 11.6*



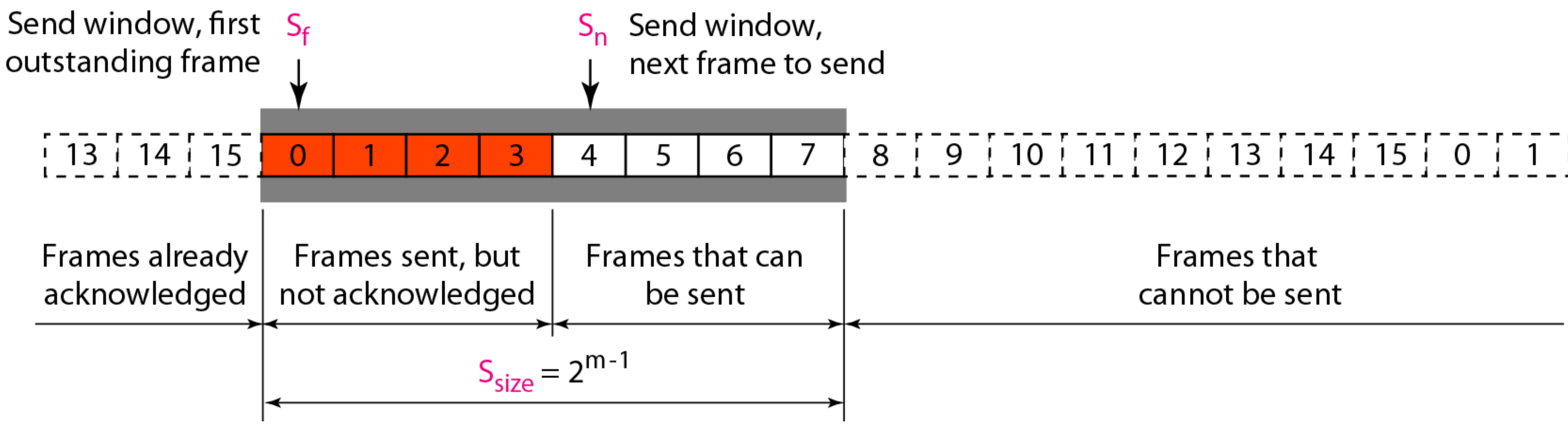
## **Figure 11.17** *Flow diagram for Example 11.7*



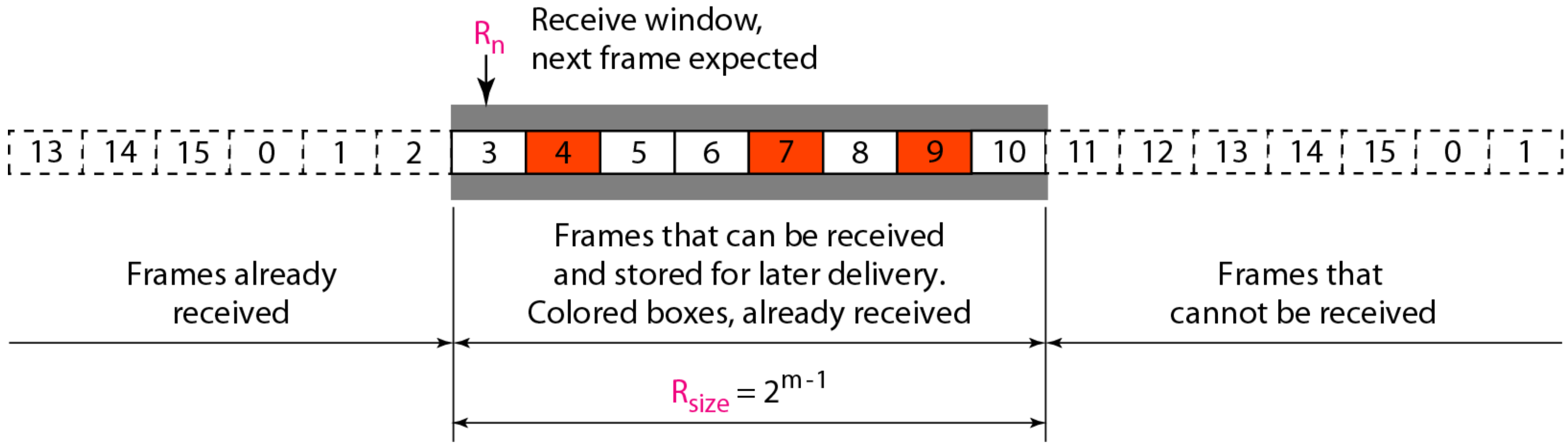
*Note*

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

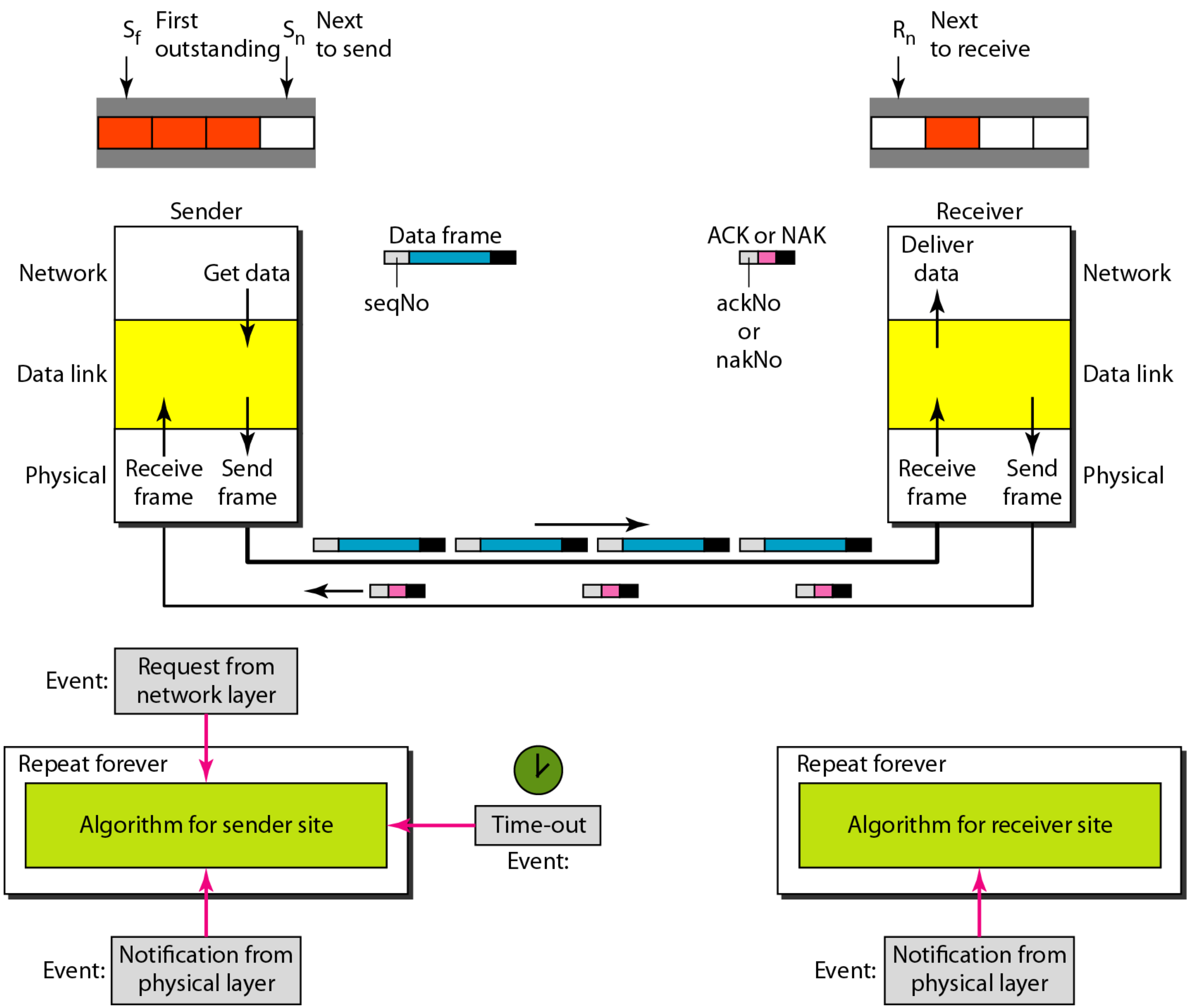
## **Figure 11.18** *Send window for Selective Repeat ARQ*



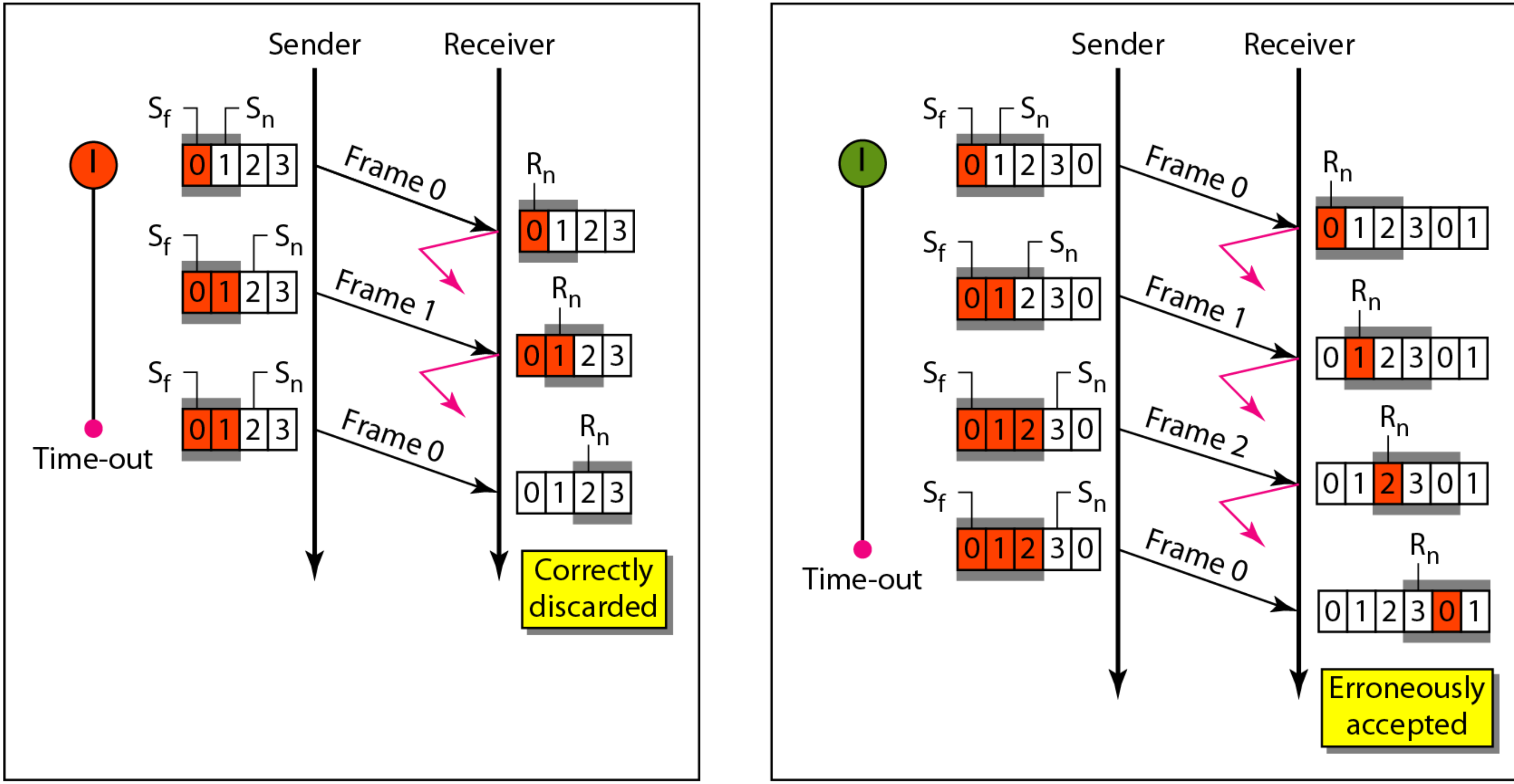
## **Figure 11.19** *Receive window for Selective Repeat ARQ*



# Figure 11.20 *Design of Selective Repeat ARQ*



## Figure 11.21 *Selective Repeat ARQ, window size*



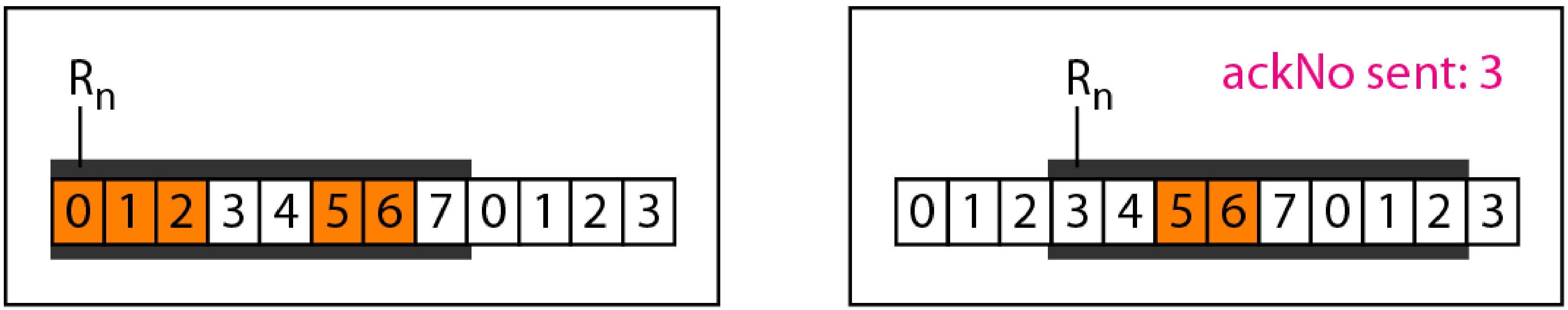

a. Window size = $2^{m-1}$

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

*Note*

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

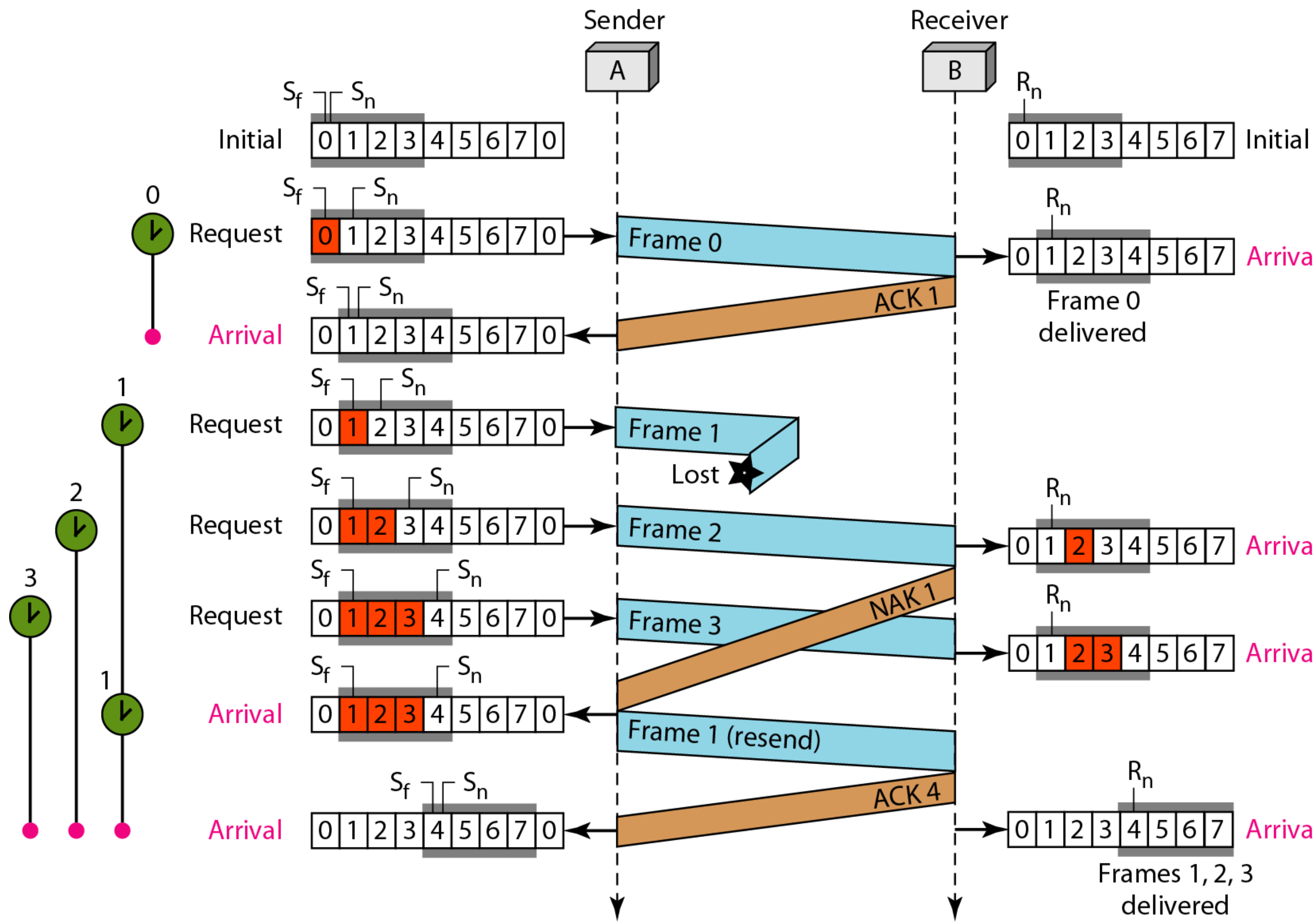
## **Figure 11.22** *Delivery of data in Selective Repeat ARQ*




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*