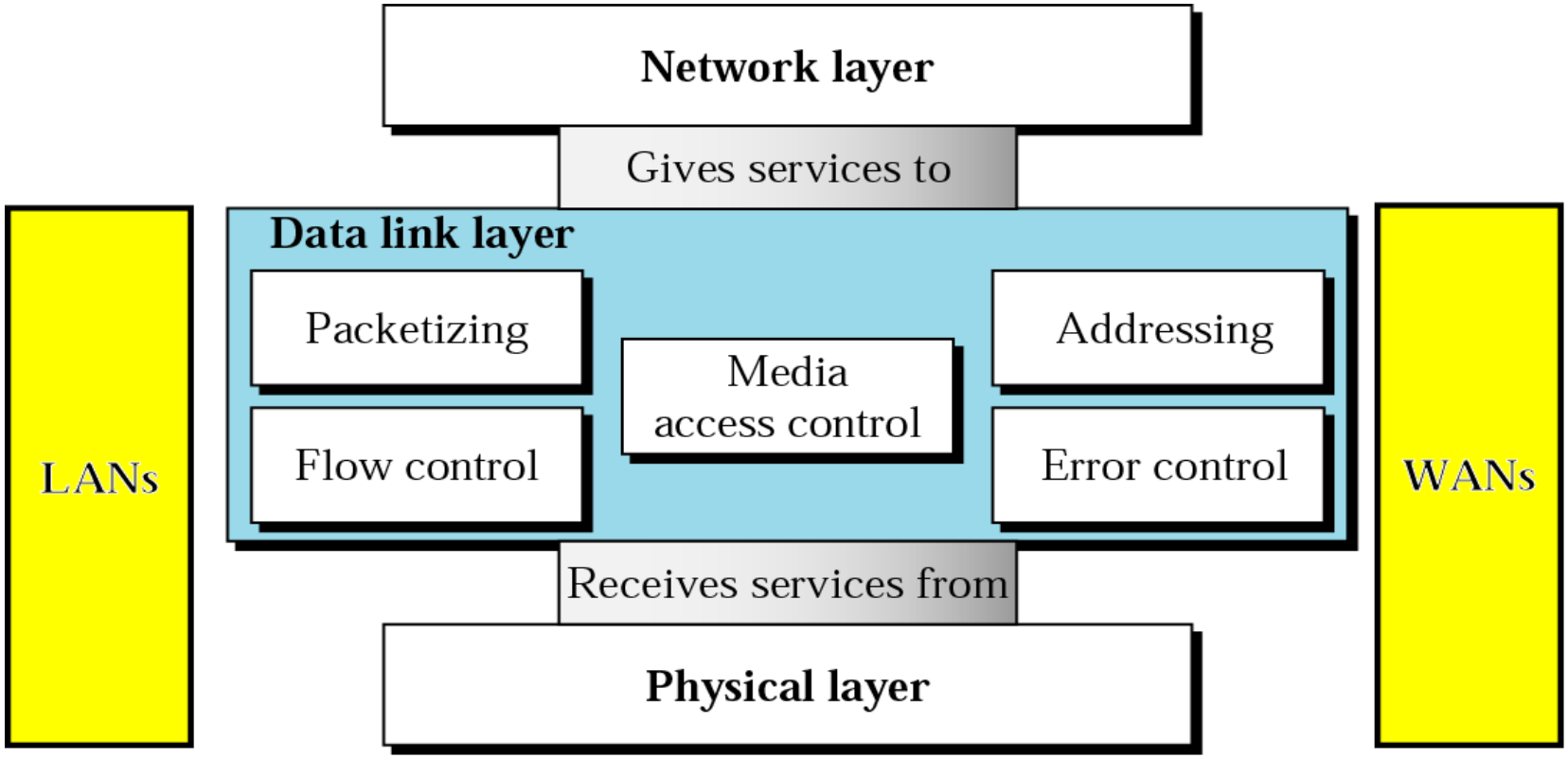
Chapter 11:
Data Link Layer:
Data Link Control

6:07 AM

Outline

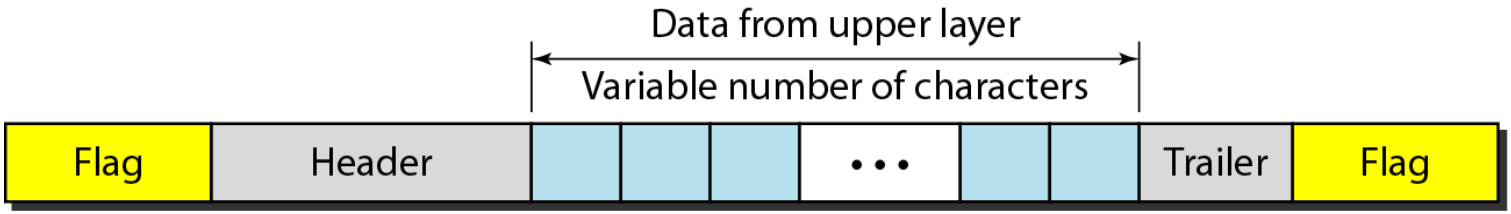
- Data link control, deals with the design and procedures for communication between two adjacent nodes: node-to-node communication which include:
  1. Framing
     - Byte vs. bit oriented protocols
  2. Data Link Control
     - Flow control
     - Error control
  3. Protocols

Data Link Layer



11-1 FRAMING

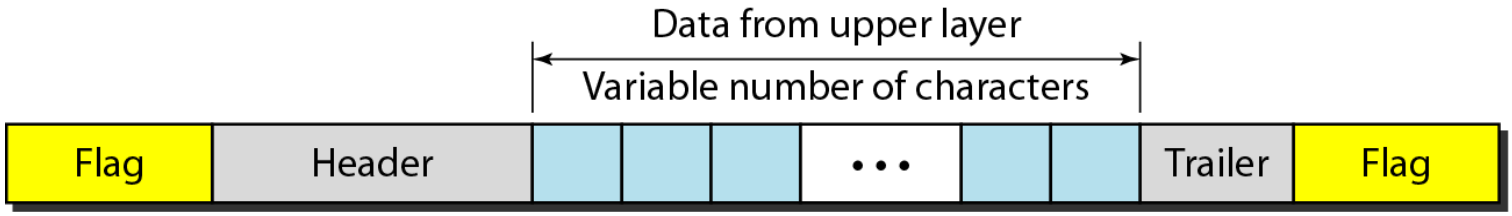
- Process of wrapping data with certain info before sending out



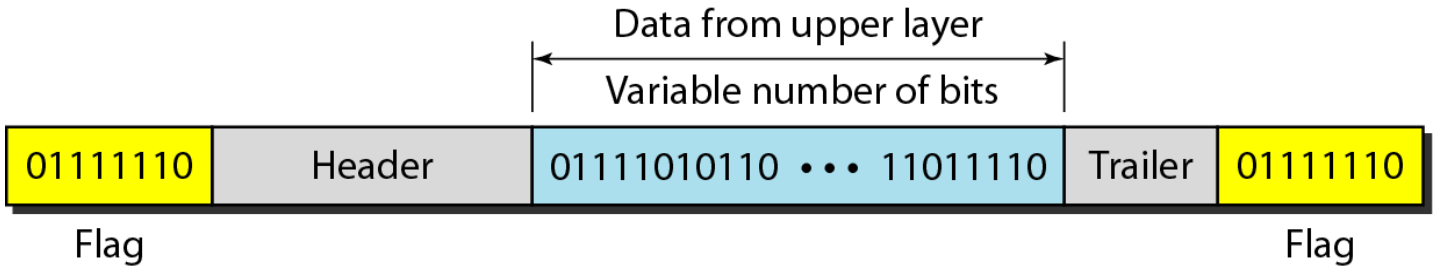
- A frame typically consists of
  - Flag: indication for start and end of a frame
  - Header: source/destination addresses, as well as other control information
  - Data: from the upper layer
  - Trailer: error detection/correction code

Byte vs. Bit Oriented

- Framing in byte-oriented protocols (popular when only text was exchanged, and for the other types of information also can be used)

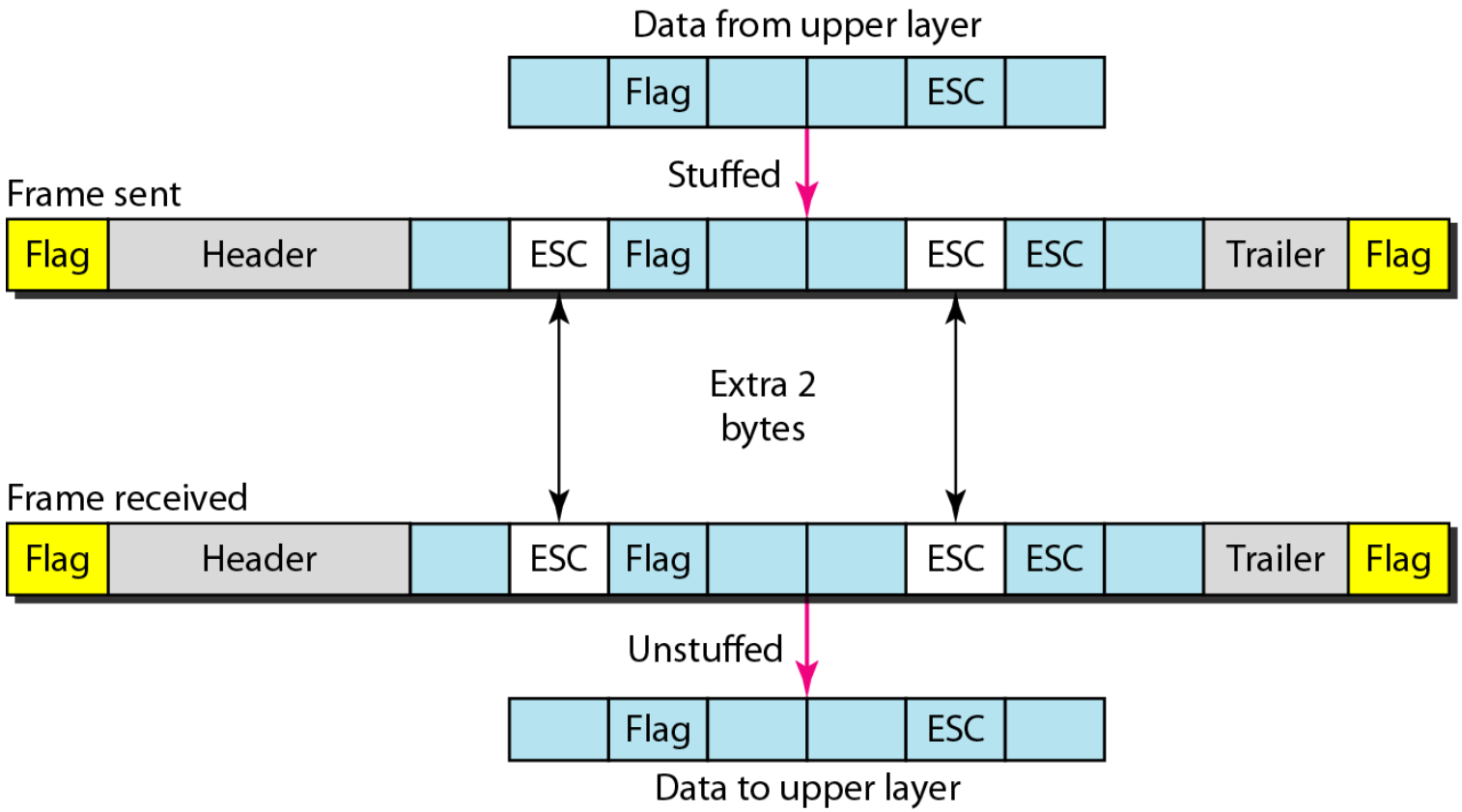


- Framing in bit-oriented protocols (text, graphs, audio, and video and so on)



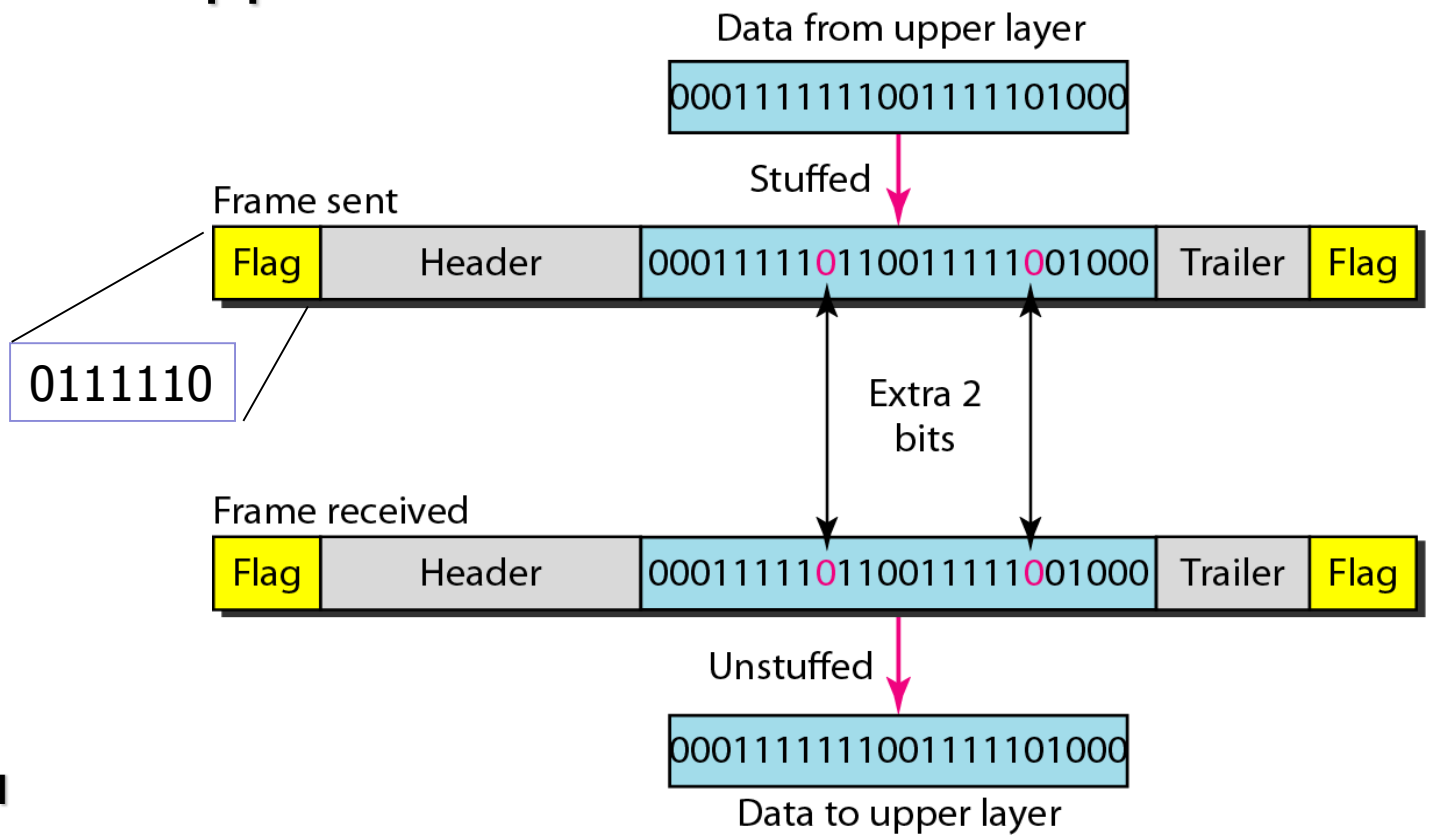
Byte Stuffing

- Process of adding extra byte whenever there is an escape or a flag character in the data



Bit Stuffing

- Process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data regardless of the value of the next bit to ensure flag sequence does not appear in the data

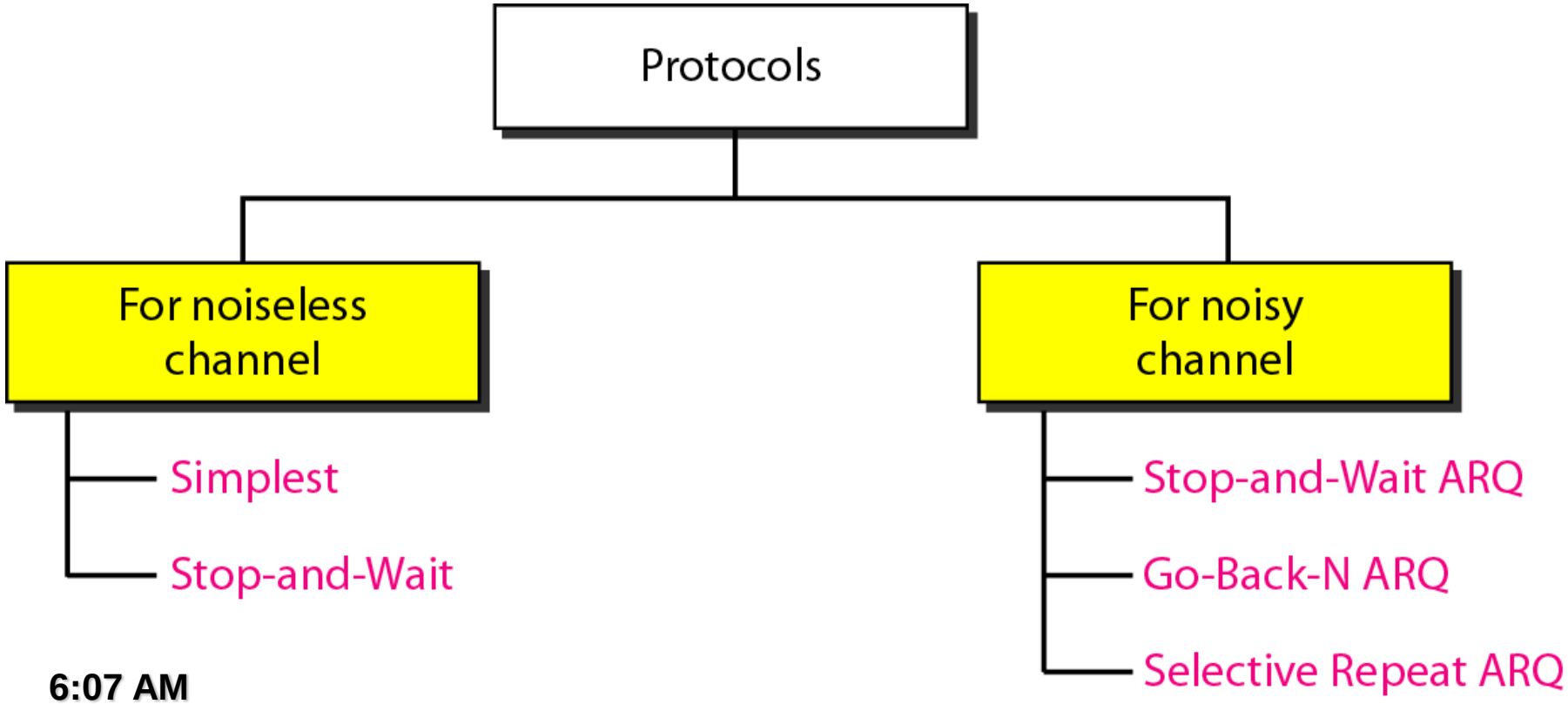


11-2 FLOW AND ERROR CONTROL

- Flow control
  - A set of procedures that tells the sender how much data can be sent before waiting for acknowledgment
- Error control
  - Includes both error detection and correction
  - Allows receiver to inform sender of lost or duplicate frames
  - Mostly based on Automatic Repeat Request (ARQ)

11-3 DATA LINK PROTOCOLS

Protocols are implemented by using one of the programming languages to define how the data link layer can combine framing, flow control, and error control to achieve the delivery of data from one node to another.

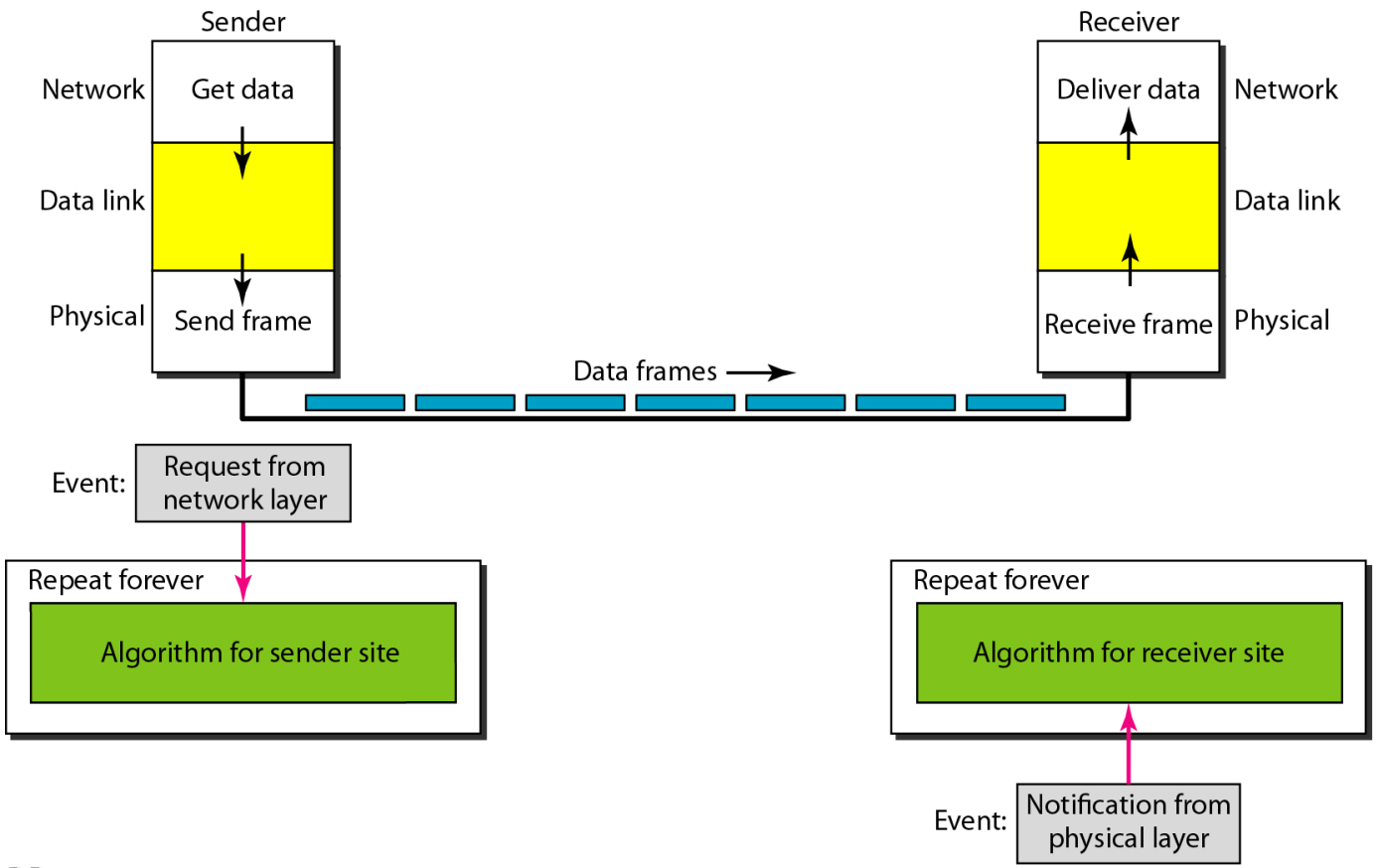


11-4 NOISELESS CHANNELS

- Assuming channel is error free (*no frames are lost, duplicated, or corrupted*)
  - Not realistic…
- No need for error control
- Two protocols for this type:
  - *Simplest Protocol*
  - *Stop-and-Wait Protocol*

"Simplest" Mechanism

- Noiseless channel (No error control)
- Unlimited buffer and speed for the receiver (No flow control)



"Simplest" : Pseudo Code

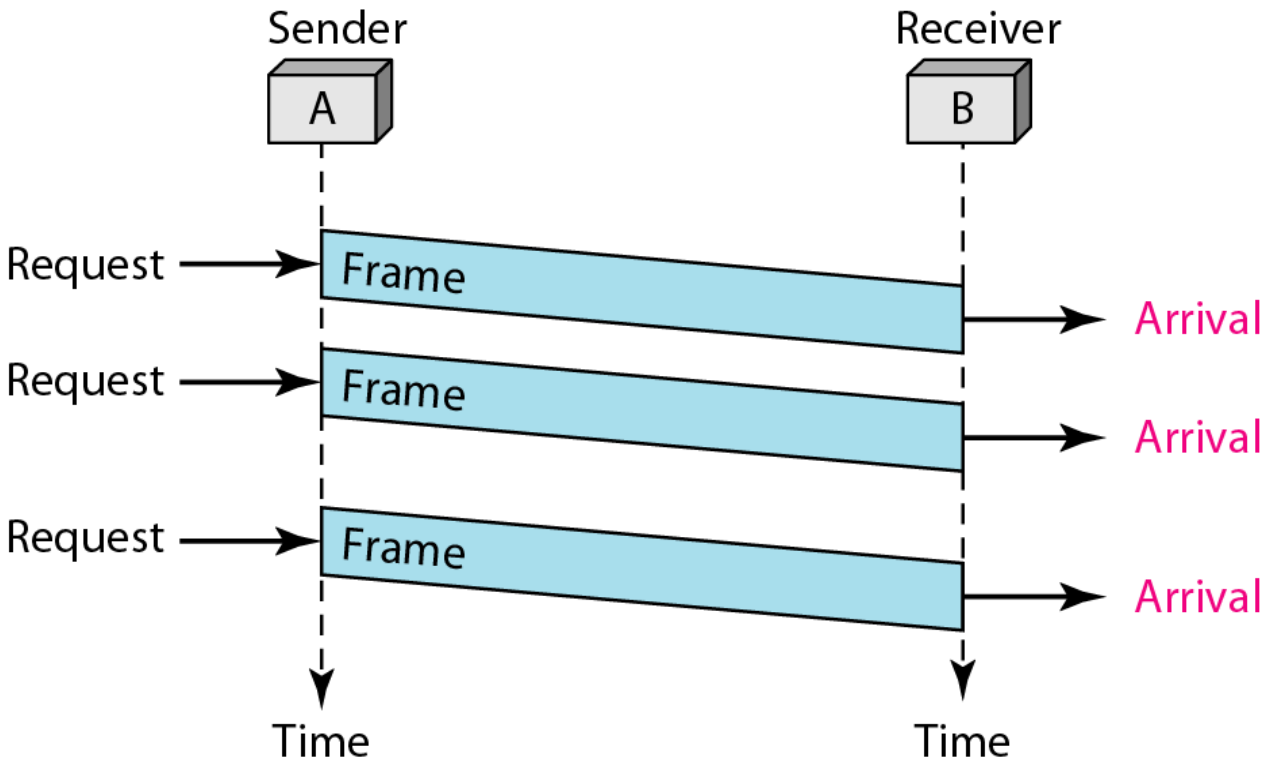
- Sender

```
while(true)                          // Repeat forever
{
  WaitForEvent();                    // Sleep until an event occurs
  if(Event(RequestToSend))           //There is a packet to send
  {
      GetData();
      MakeFrame();
      SendFrame();                   //Send the frame
  }
}
```

- Receiver

```
while(true)                          // Repeat forever
{
  WaitForEvent();                    // Sleep until an event occurs
  if(Event(ArrivalNotification))     //Data frame arrived
  {
      ReceiveFrame();
      ExtractData();
      DeliverData();                 //Deliver data to network layer
  }
}
```

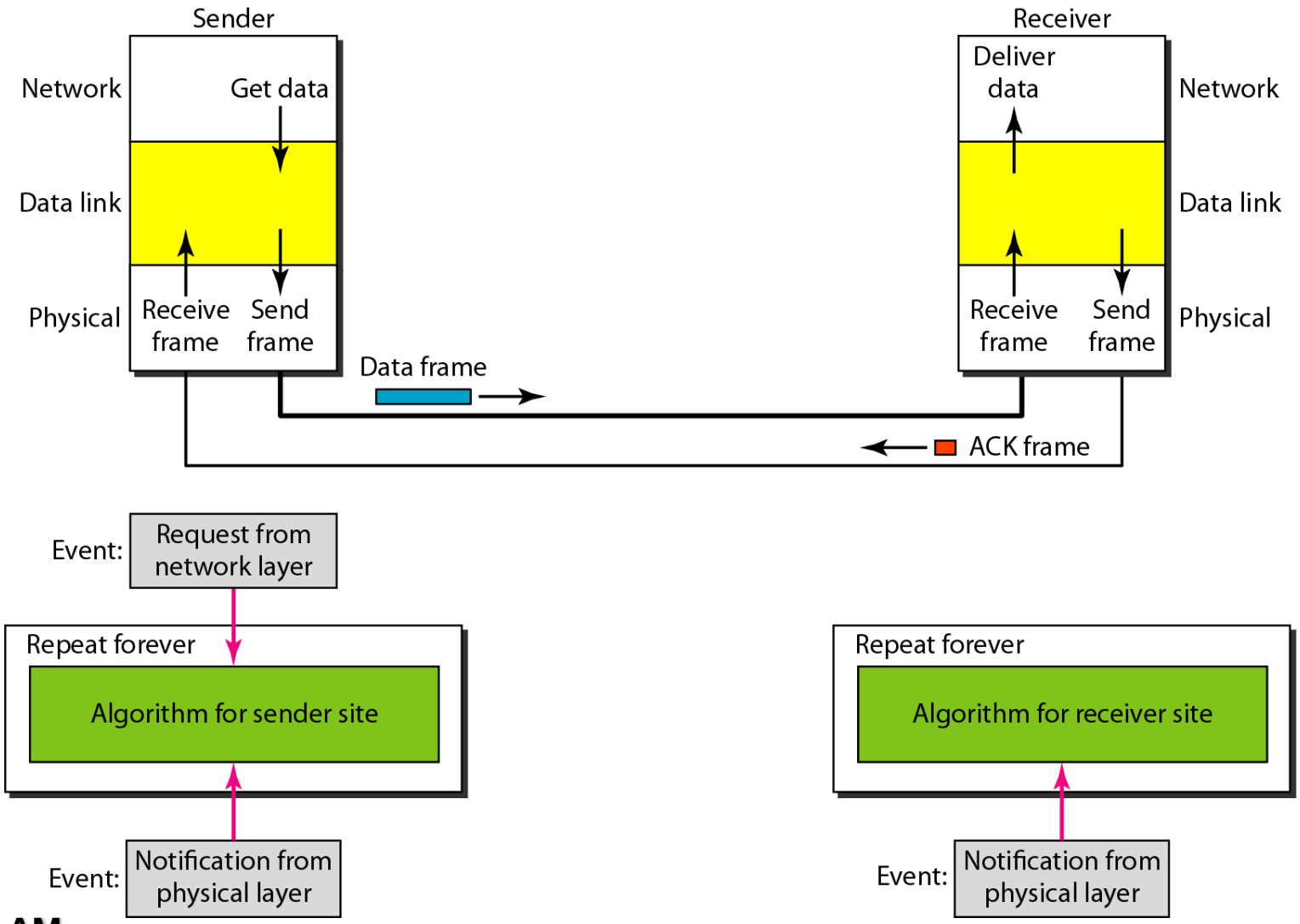
"Simplest": Flow Diagram



Stop-and-Wait Mechanism

- Still noiseless channel
- Receiver has limited buffer
  - Requires flow control
- Sender sends one frame at a time and wait for an acknowledgment

Stop-and-Wait: Overview



Stop-and-Wait: Pseudo Code

- Sender side

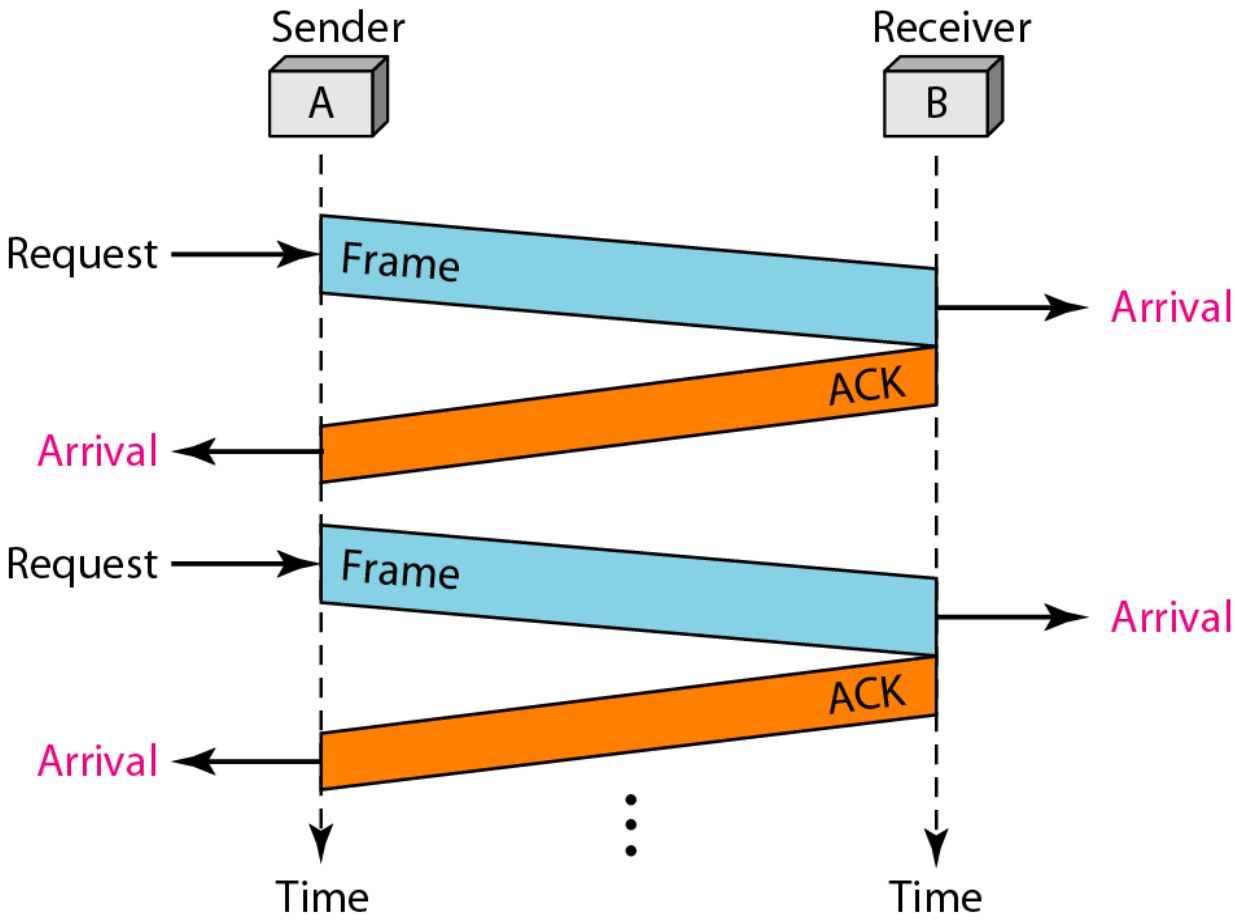
```
while(true)                          //Repeat forever
canSend = true                       //Allow the first frame to go
{
  WaitForEvent();                    // Sleep until an event occurs
  if(Event(RequestToSend) AND canSend)
  {
     GetData();
     MakeFrame();
     SendFrame();                    //Send the data frame
     canSend = false;                //Cannot send until ACK arrives
  }
  WaitForEvent();                    // Sleep until an event occurs
  if(Event(ArrivalNotification)      // An ACK has arrived
   {
     ReceiveFrame();                 //Receive the ACK frame
     canSend = true;
   }
}
```

Stop-and-Wait: Pseudo Code

- Receiver side

```
while(true)                          //Repeat forever
{
  WaitForEvent();                    // Sleep until an event occurs
  if(Event(ArrivalNotification))     //Data frame arrives
  {
     ReceiveFrame();
     ExtractData();
     Deliver(data);                  //Deliver data to network layer
     SendFrame();                    //Send an ACK frame
  }
}
```

Stop-and-Wait: Flow Diagram



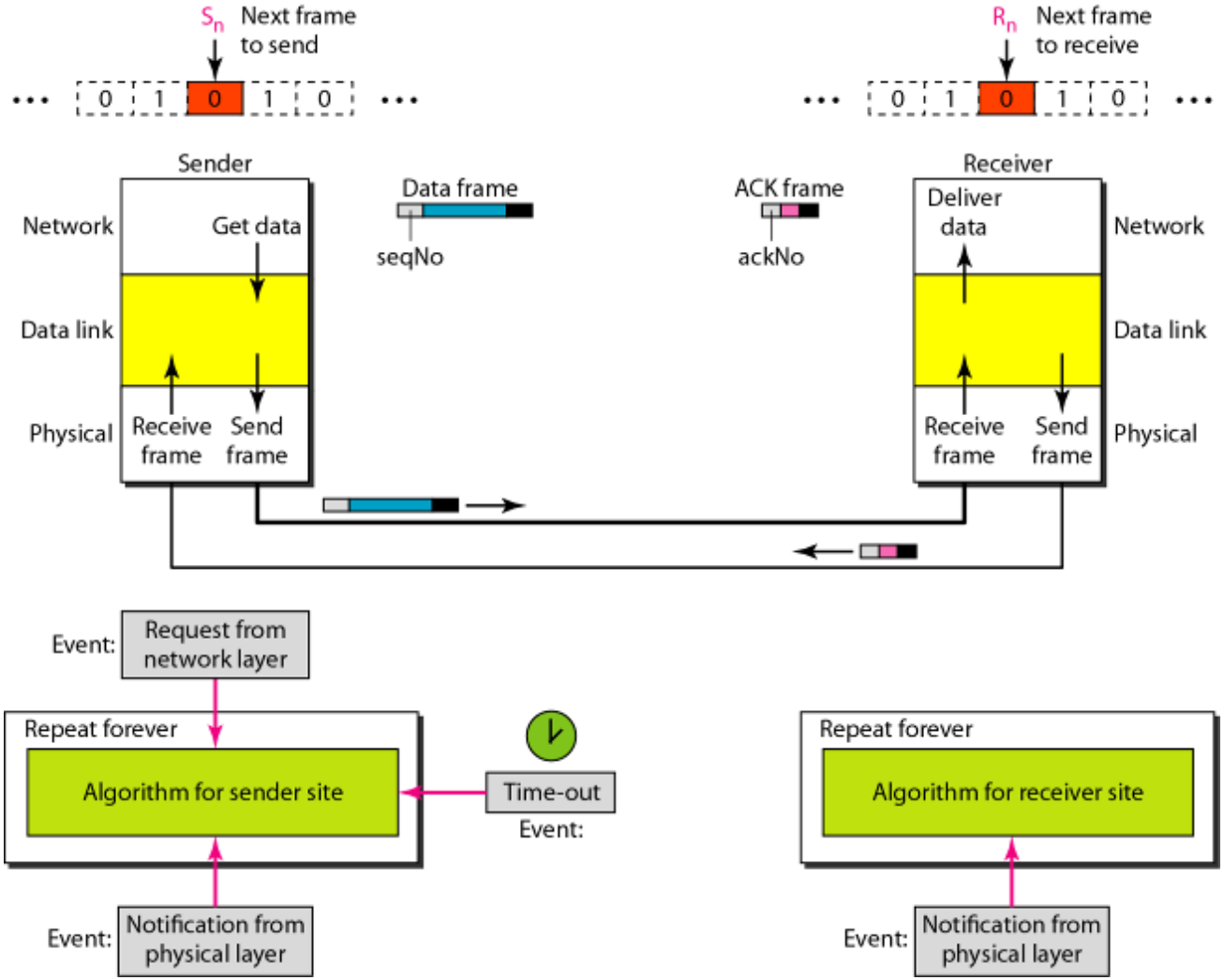
11-5 NOISY CHANNELS

- Realistic
  - Error can and will happen
  - Require error control
- Mechanisms:
  - Stop-and-Wait ARQ
  - Go-Back-N ARQ
  - Selective Repeat ARQ

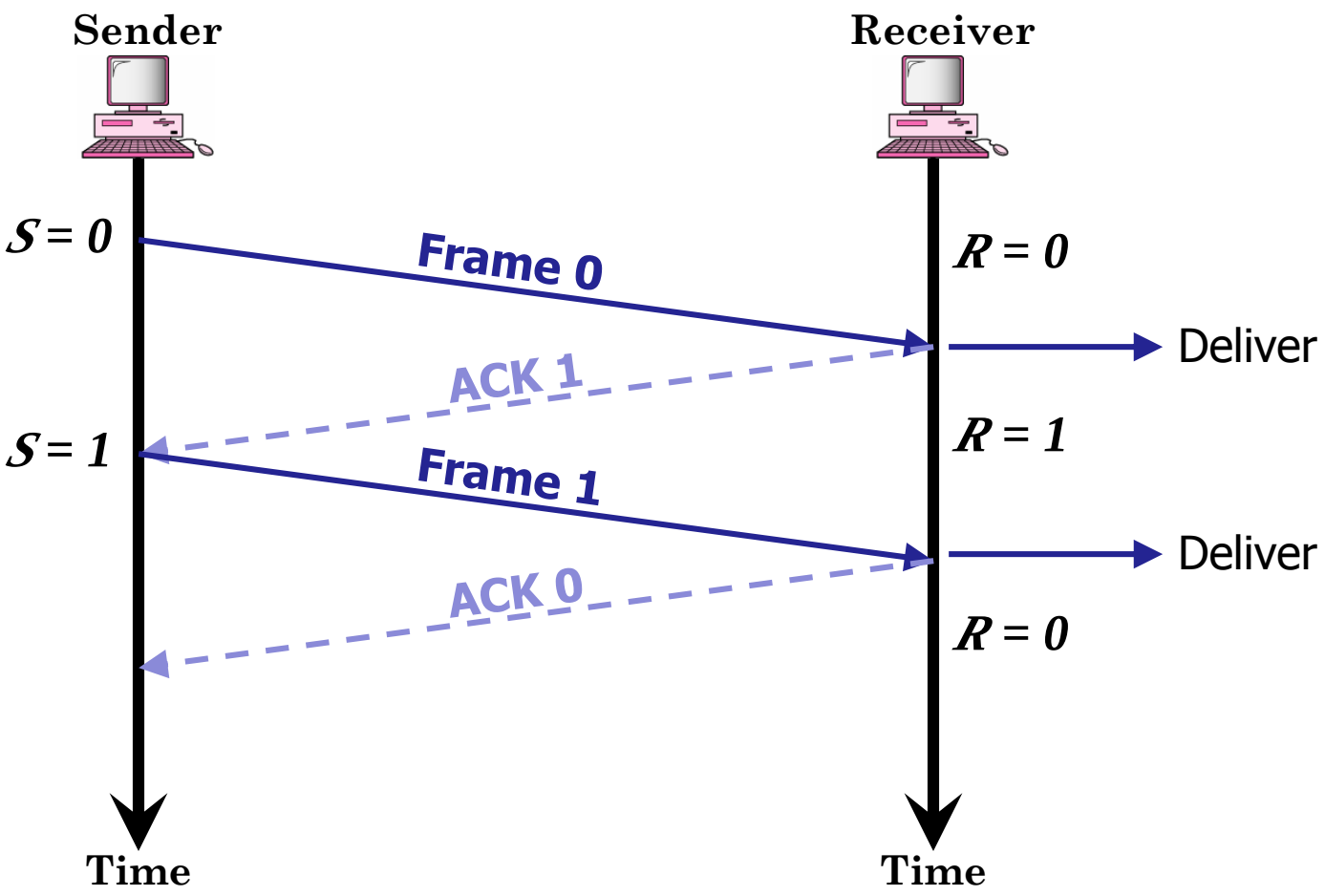
Stop-and-Wait ARQ

- Sender keeps a copy of sent frame until successful delivery is ensured
- Receiver responds with an ack when it successfully receives a frame
- Both data and ack frames must be numbered
- When sender does not receive an ack within certain time, it assumes frame is lost, then retransmits the same frame.

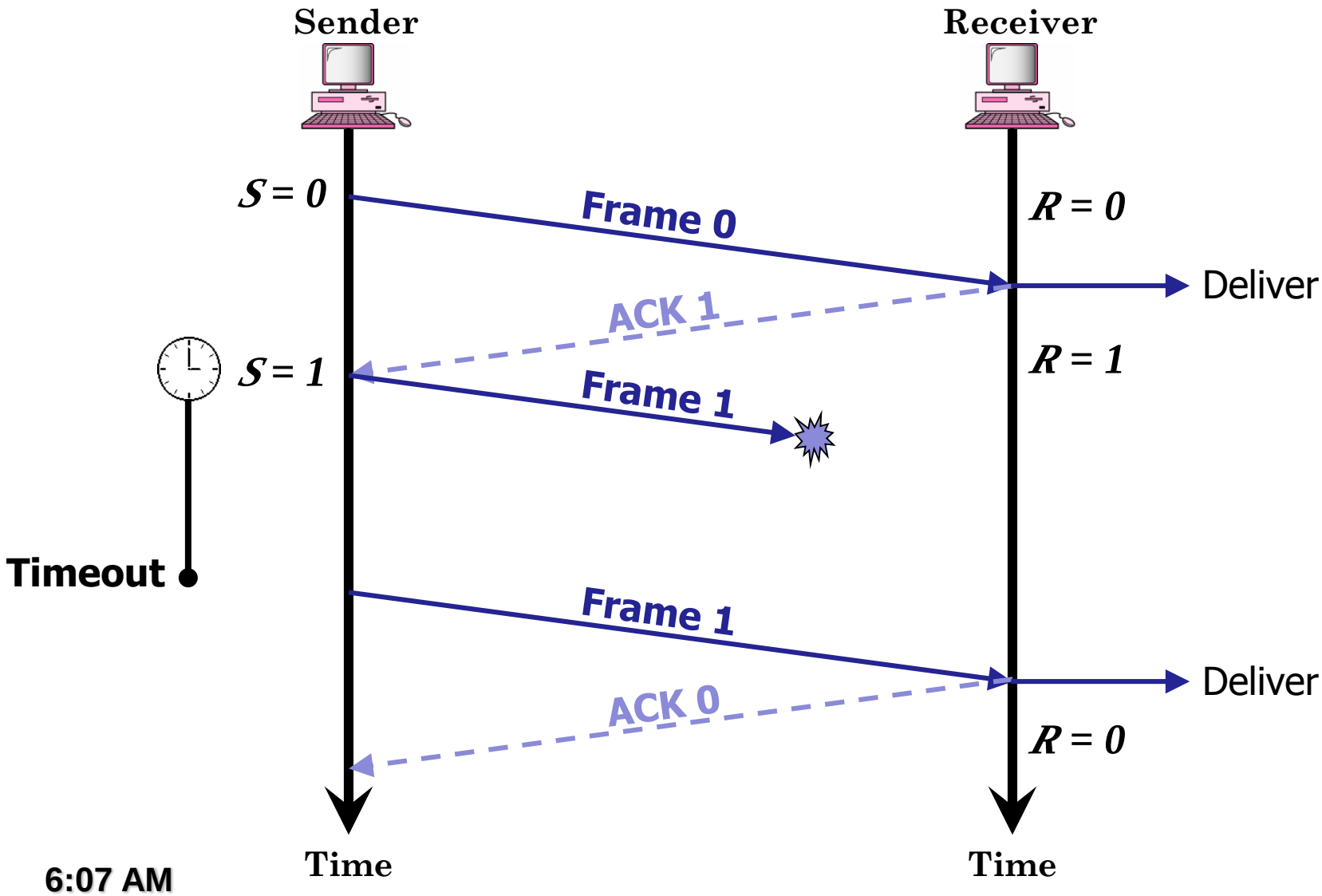
Stop-and-Wait ARQ



Flow Diagram: Normal Operation

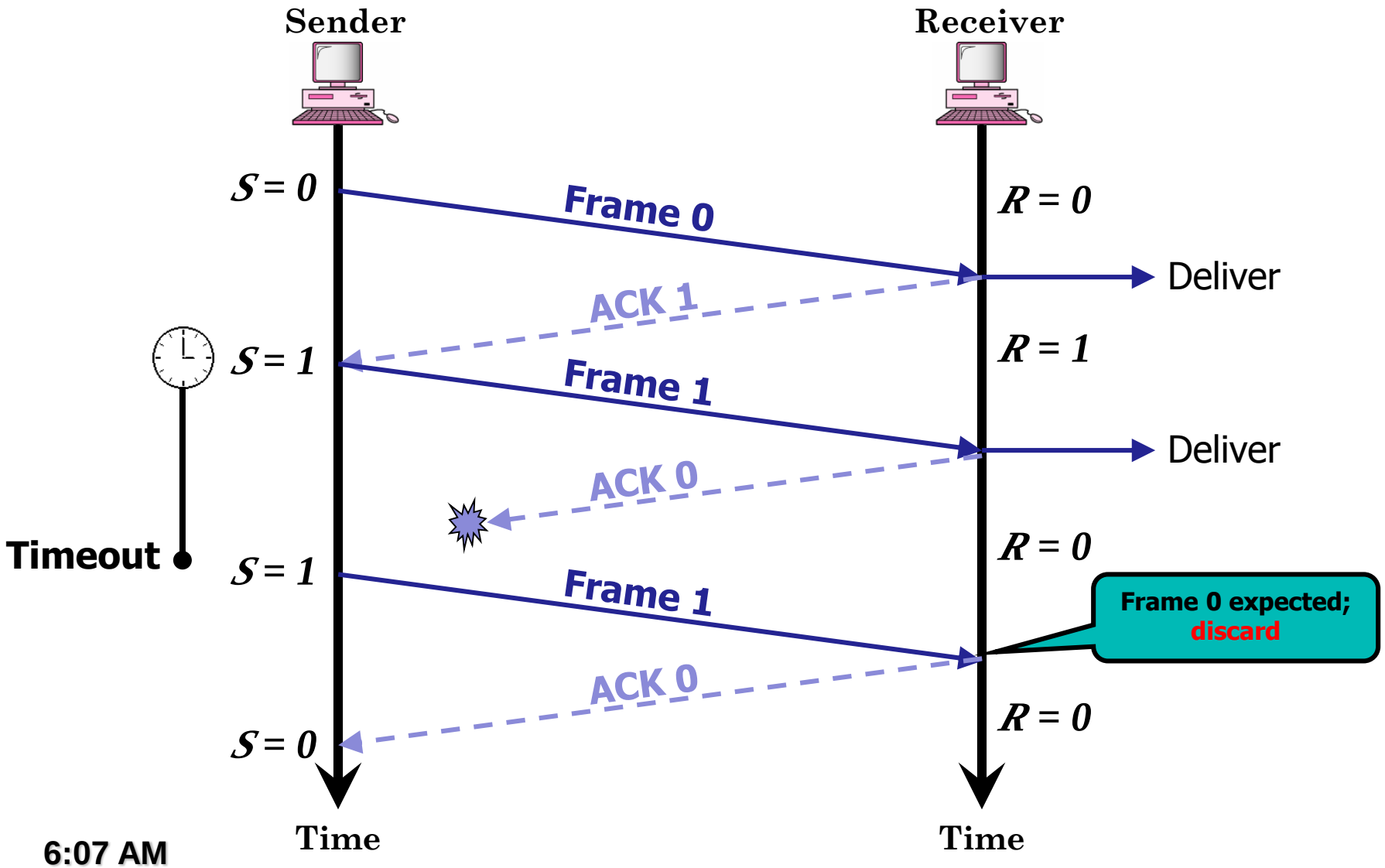


Flow Diagram: Lost Frame



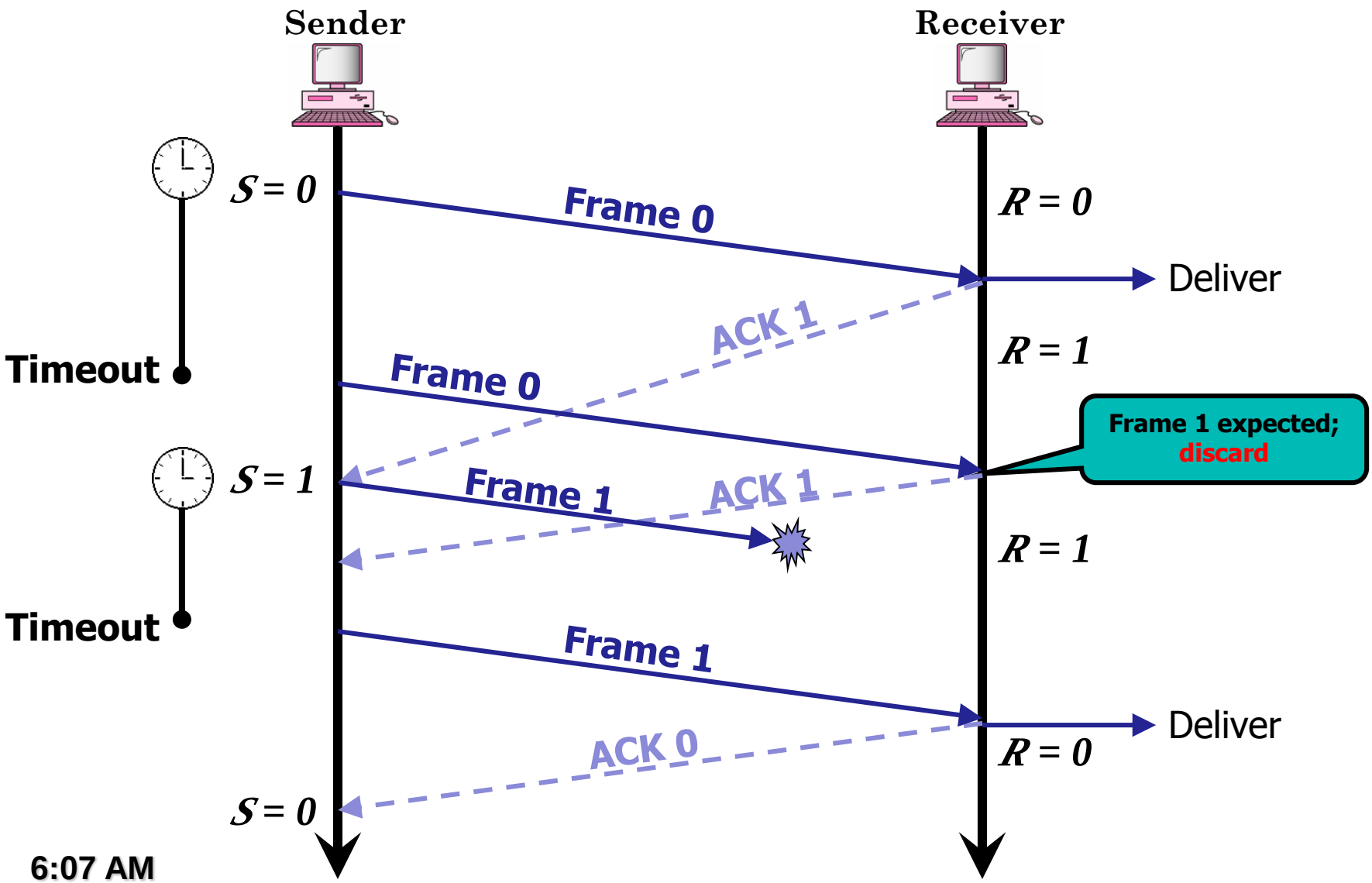
6:07 AM

Flow Diagram: Lost ACK



6:07 AM

Flow Diagram: Delayed ACK



6:07 AM

Go-Back-N ARQ

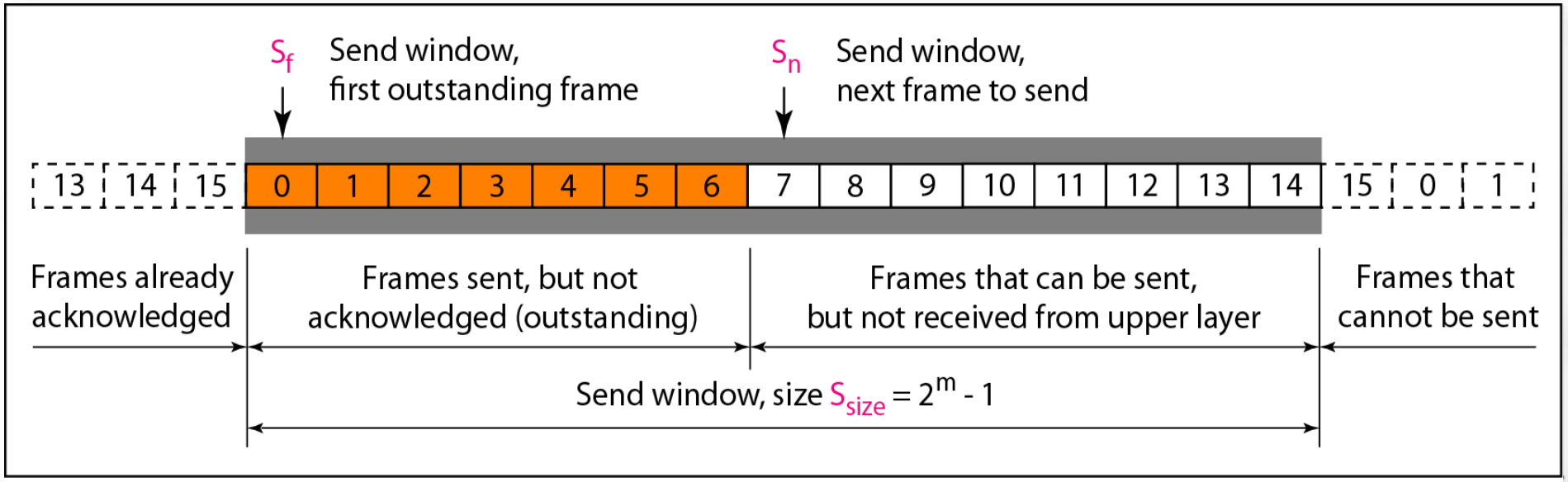
- Allows multiple frames to be sent before waiting for ACK
  - These frames must be numbered differently
  - Frame numbers are called Sequence numbers
- Frames must be received in the correct order
- If a frame is lost, the lost frame and all of the following frames must be retransmitted

Sequence Numbers

- Frame header contains m bits for sequence number
- That allows up to 2^m different frame numbers
- How big should m be???

Sending Window

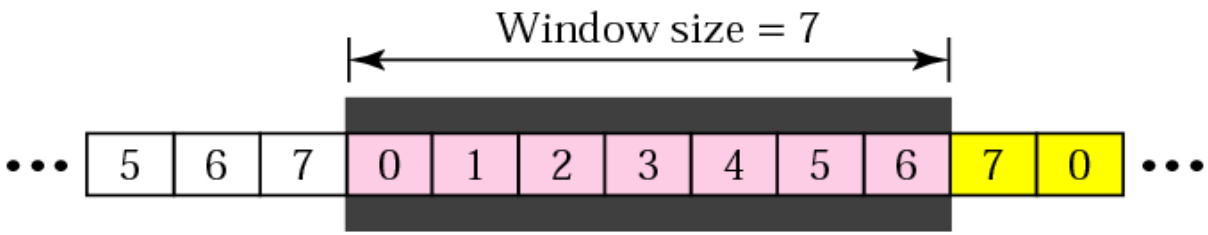
- Sending more than one frame at once requires sender to buffer multiple frames
  - Known as "sending window" which has three variables: S_f , S_n , S_{size}.



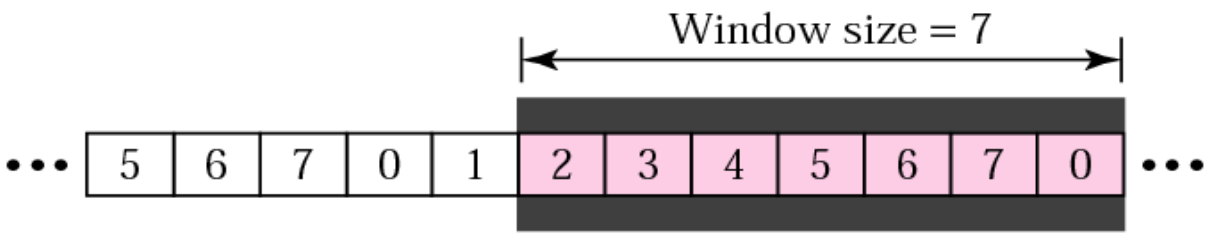
a. Send window before sliding

"Sliding" Window

- Once the first frames in the window is ACKed
  - ACKed frames are removed from the buffer
  - More frames are transmitted
  - Result: The window slides to the right



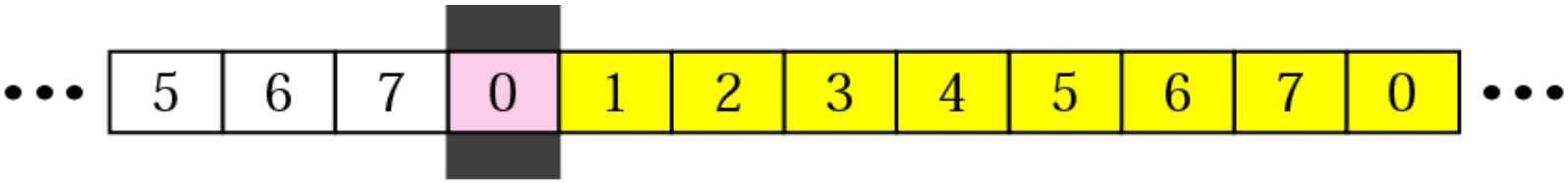
a. Before sliding



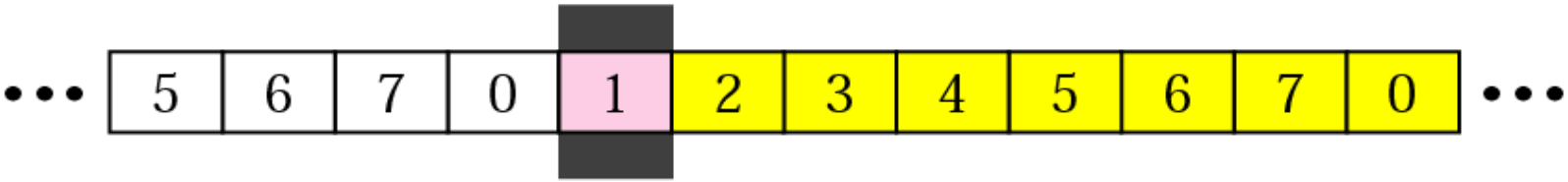
b. After sliding two frames

Receiving Window

- Receiver expects one frame at a time

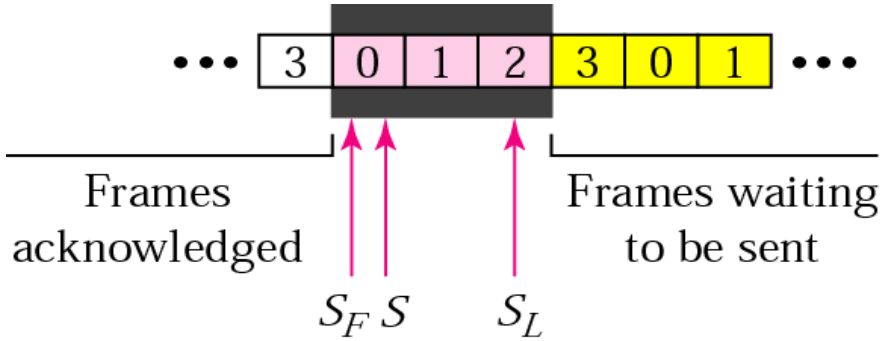


a. Before sliding

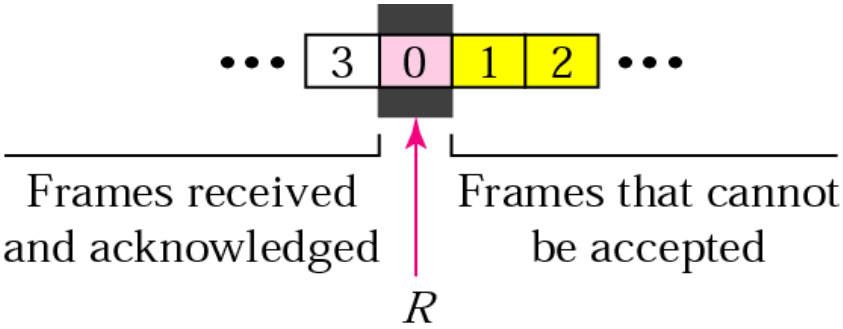


b. After sliding

Send vs. Receive Windows



a. Sender window

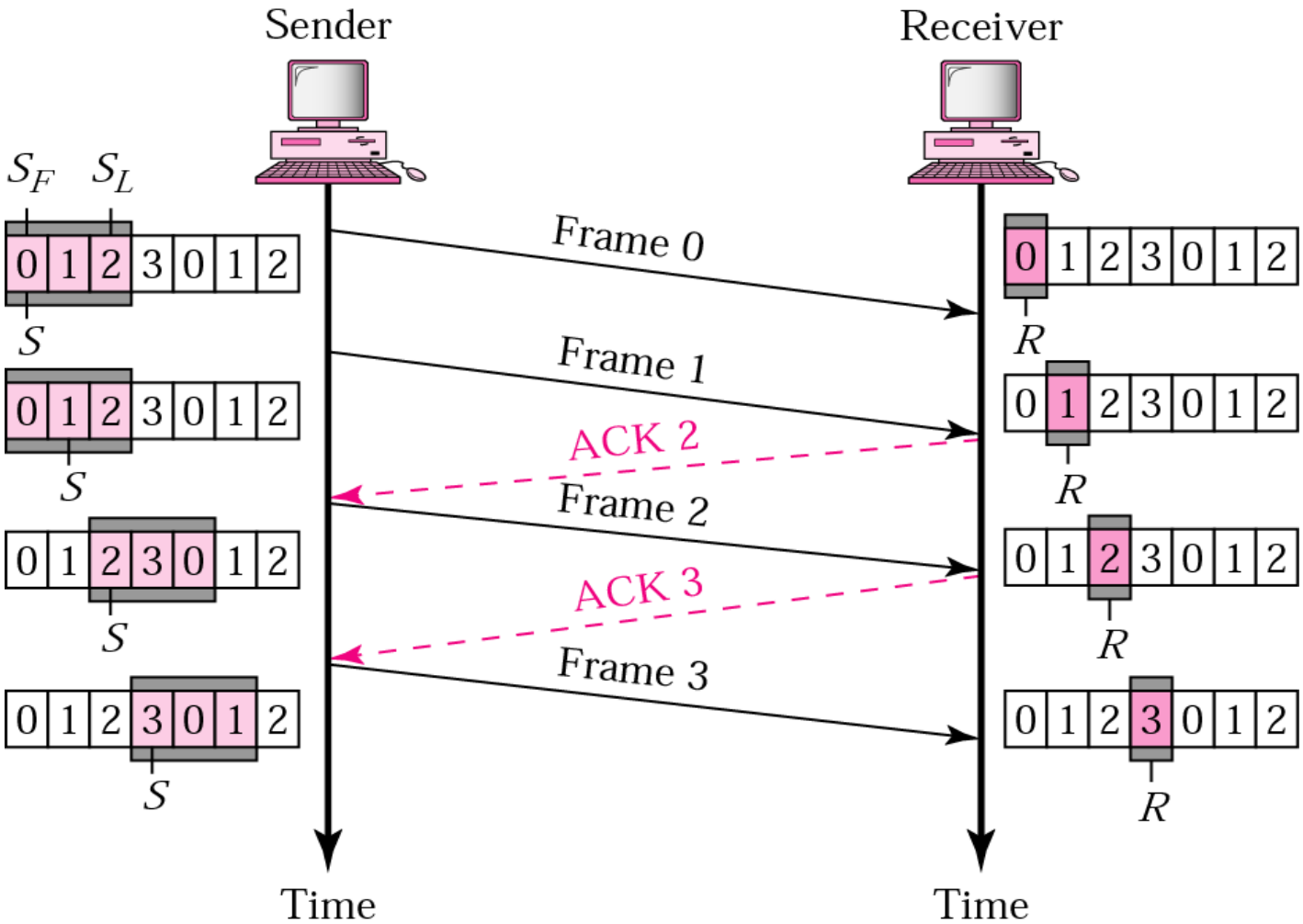


b. Receiver window

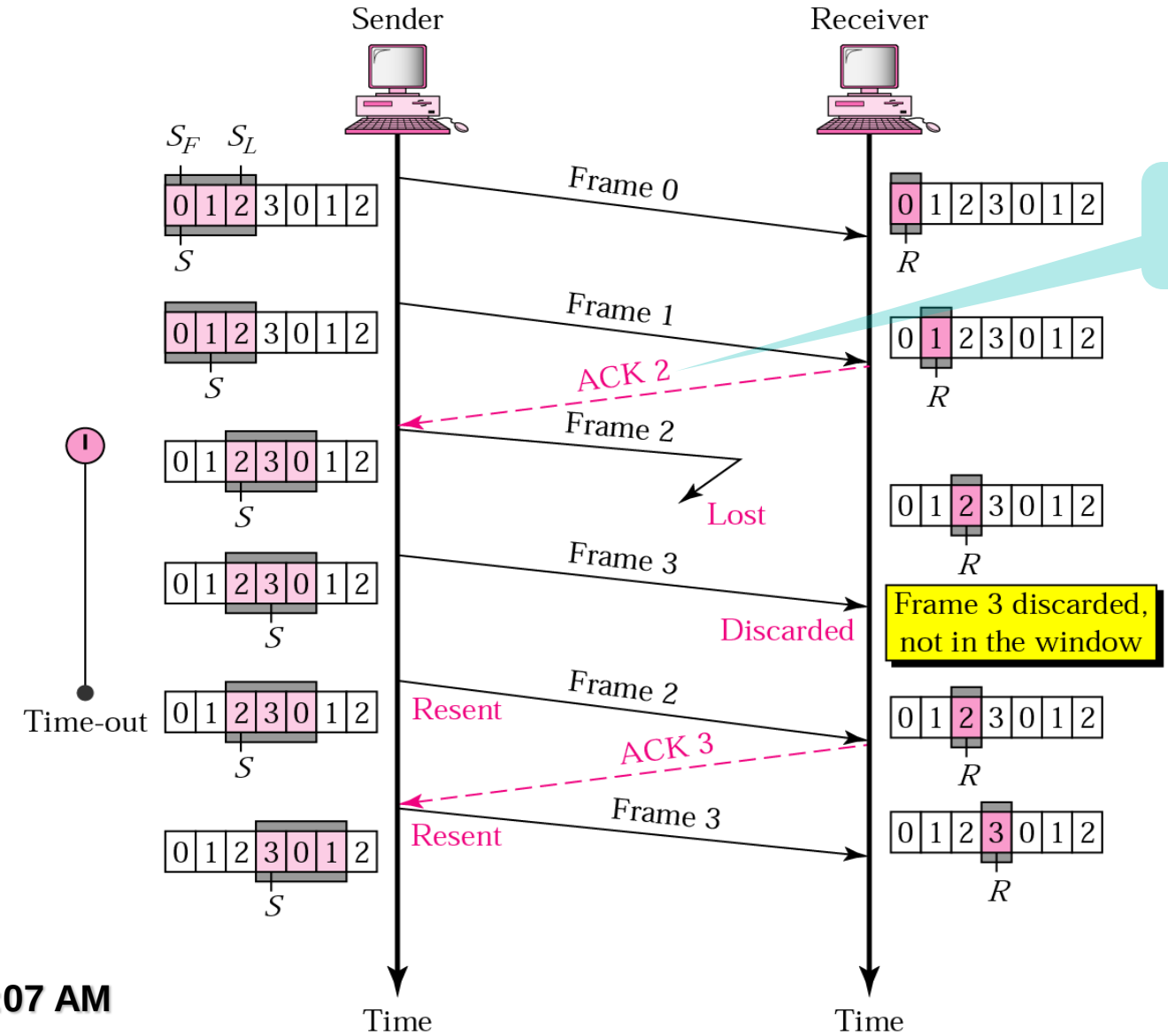
Go-Back-N: Window Sizes

- For m-bit sequence numbers
- Send window size: at most 2^m-1
  - → Up to 2^m-1 frames can be sent without ACK
- Receive window size: 1
  - → Frames must be received in order

Go-Back-N: Normal Operation



Go-Back-N: Lost Frame

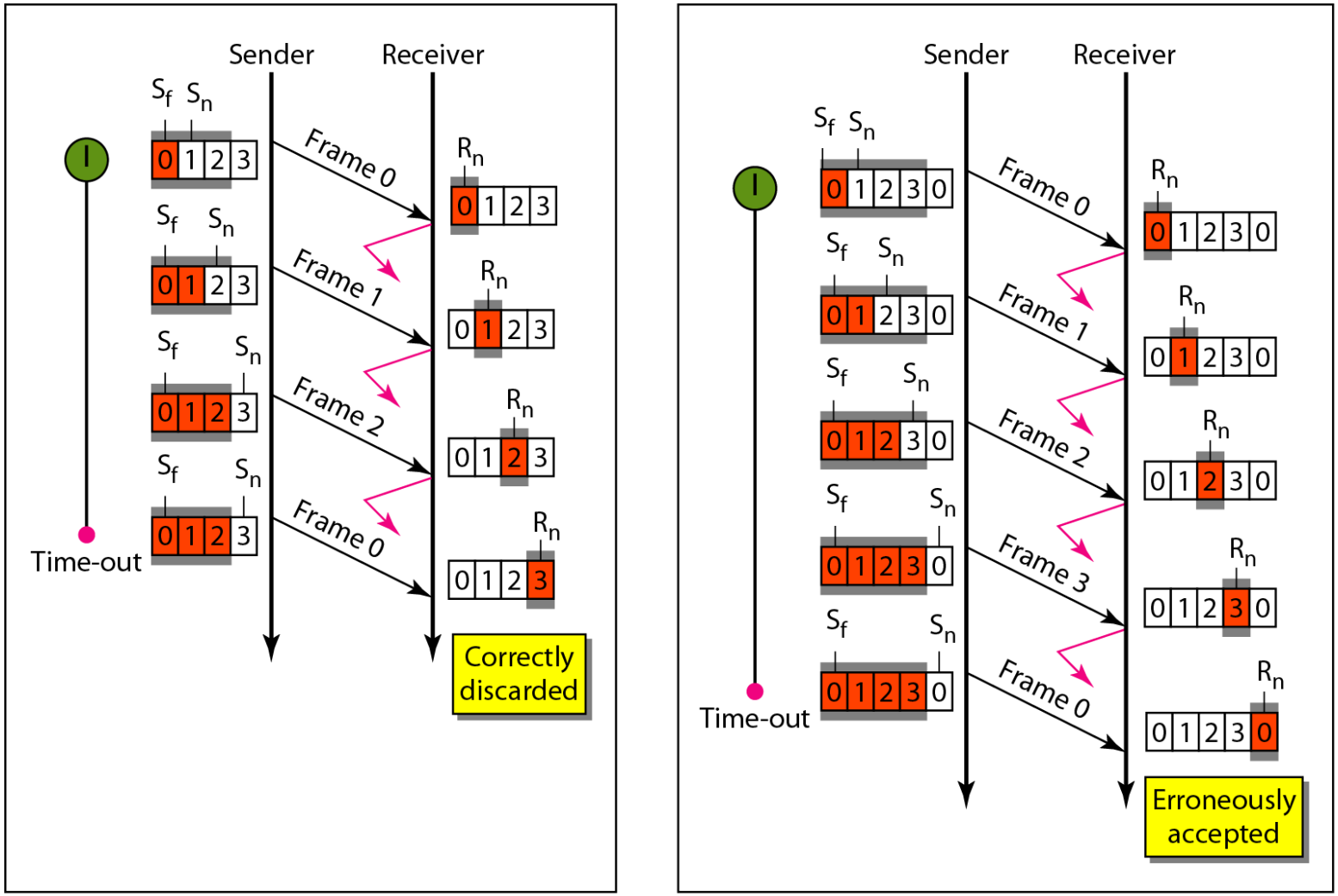


ACKs are cumulative

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 11.15: Lost ACK: *Window size for Go-Back-N ARQ*



a. Window size < 2^m

b. Window size = 2^m

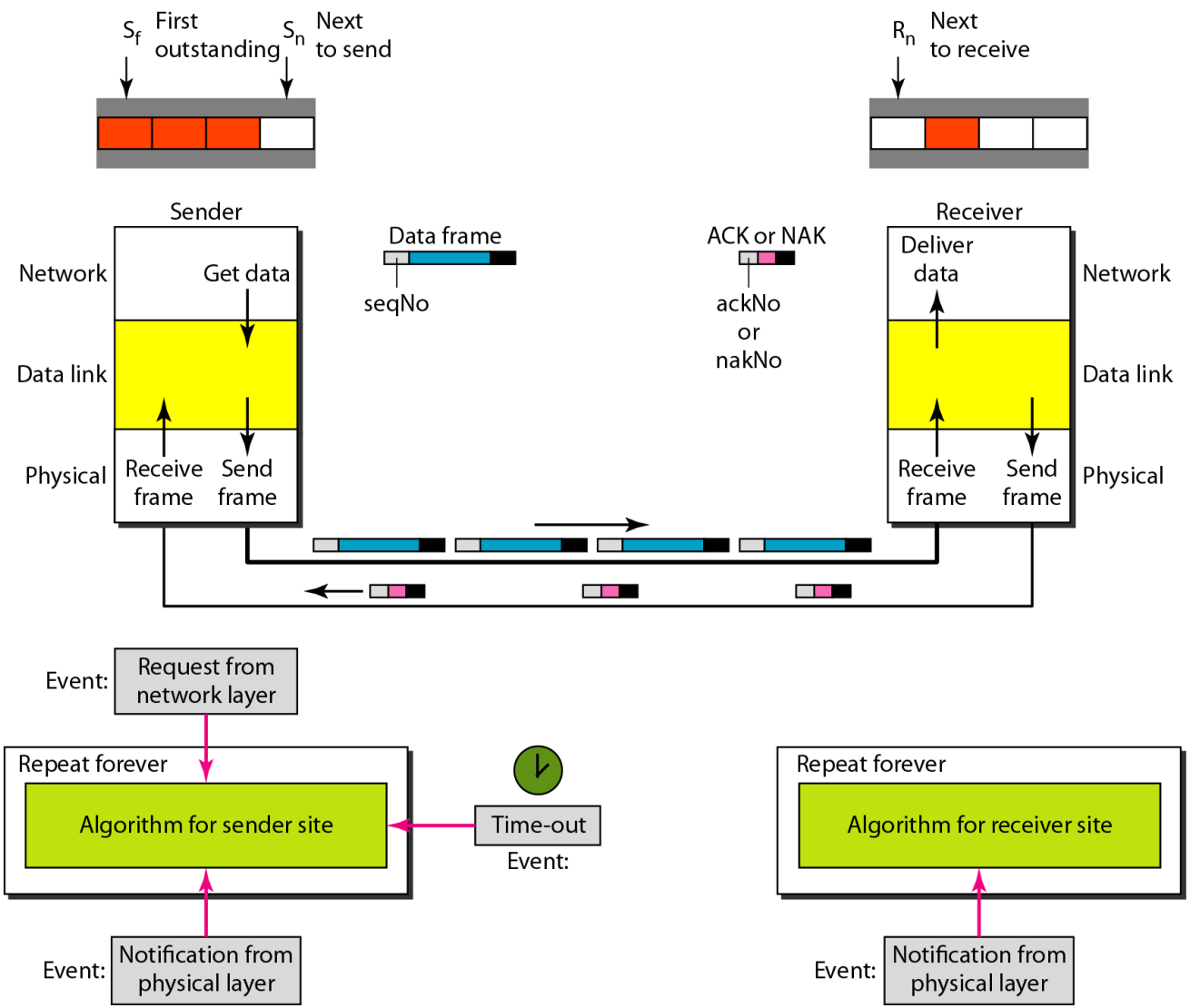
Selective Repeat ARQ

- Go-Back-N always discards out-of-order frames
  - Losing one frame may result in retransmission of multiple frames
  - Very inefficient in noisy link
- Selective Repeat ARQ allows frames to be received out of order
  - Therefore, receive window > 1

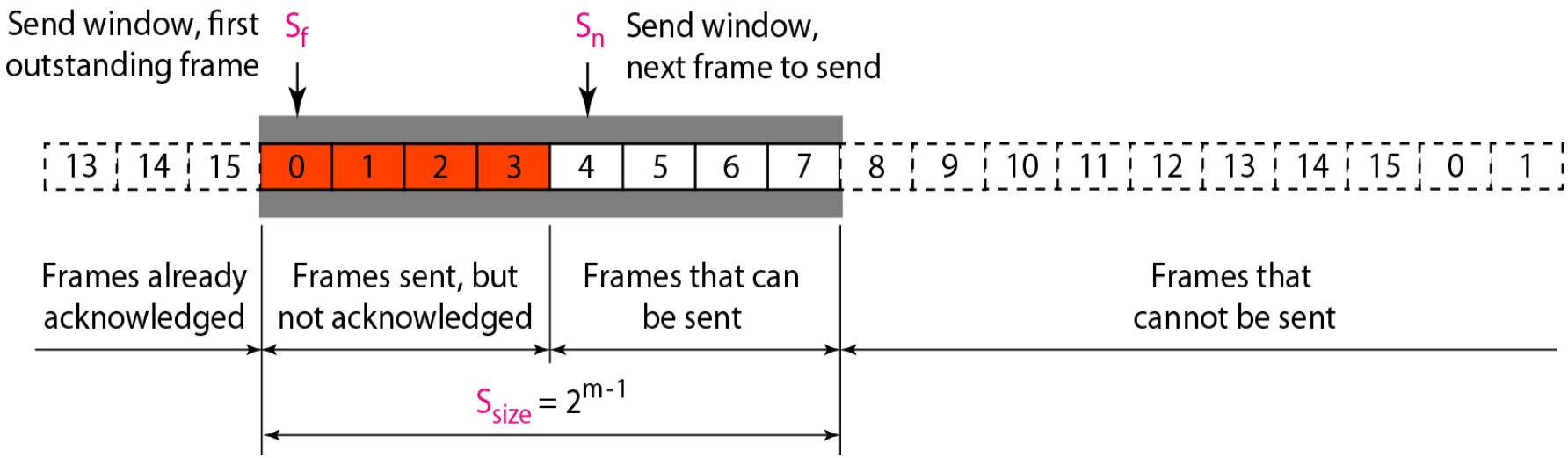
Send and Receive Windows

- Sender and receiver share window space equally
- For m-bit sequence numbers
  - Send window: up to 2^{m-1}
  - Receive window: up to 2^{m-1}

Figure 11.20 *Design of Selective Repeat ARQ*



Send Window


Send window, first outstanding frame
S_f
S_n
Send window, next frame to send
13 14 15 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 1
Frames already acknowledged
Frames sent, but not acknowledged
Frames that can be sent
Frames that cannot be sent
$S_{size} = 2^{m-1}$

Receive Window

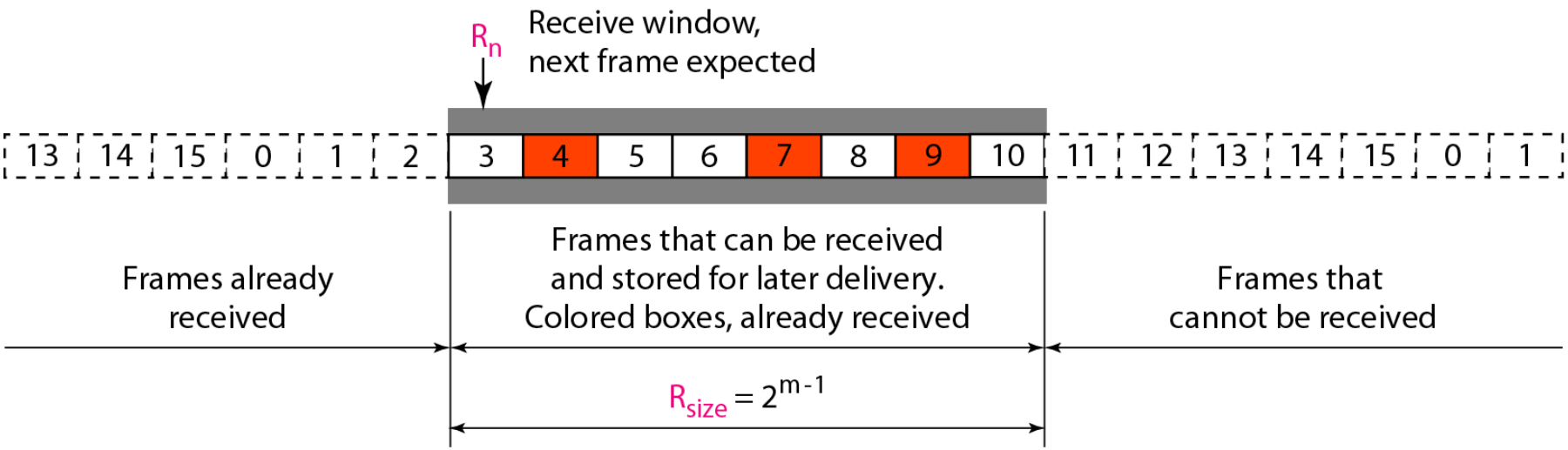
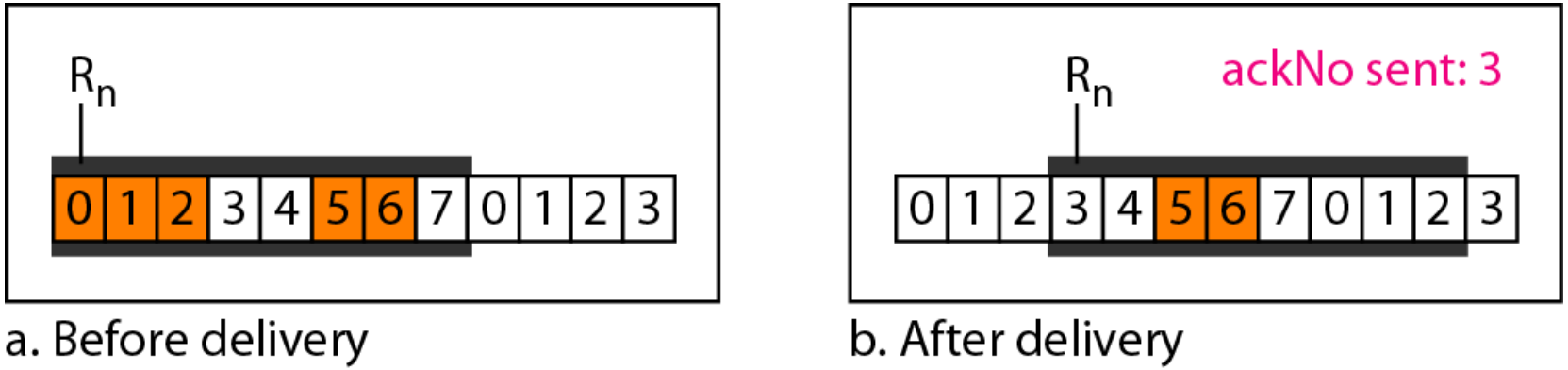


Figure 11.22 *Delivery of data in Selective Repeat ARQ*

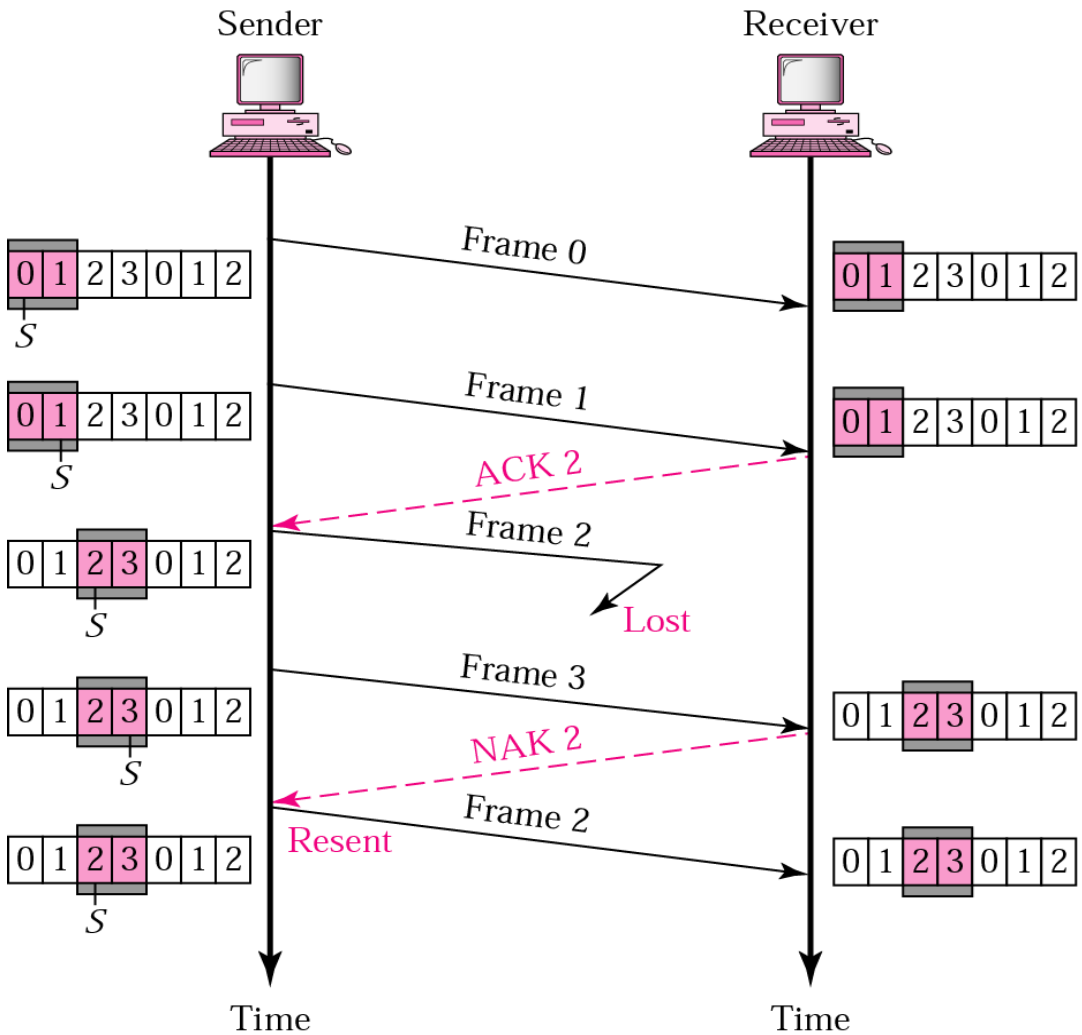


a. Before delivery

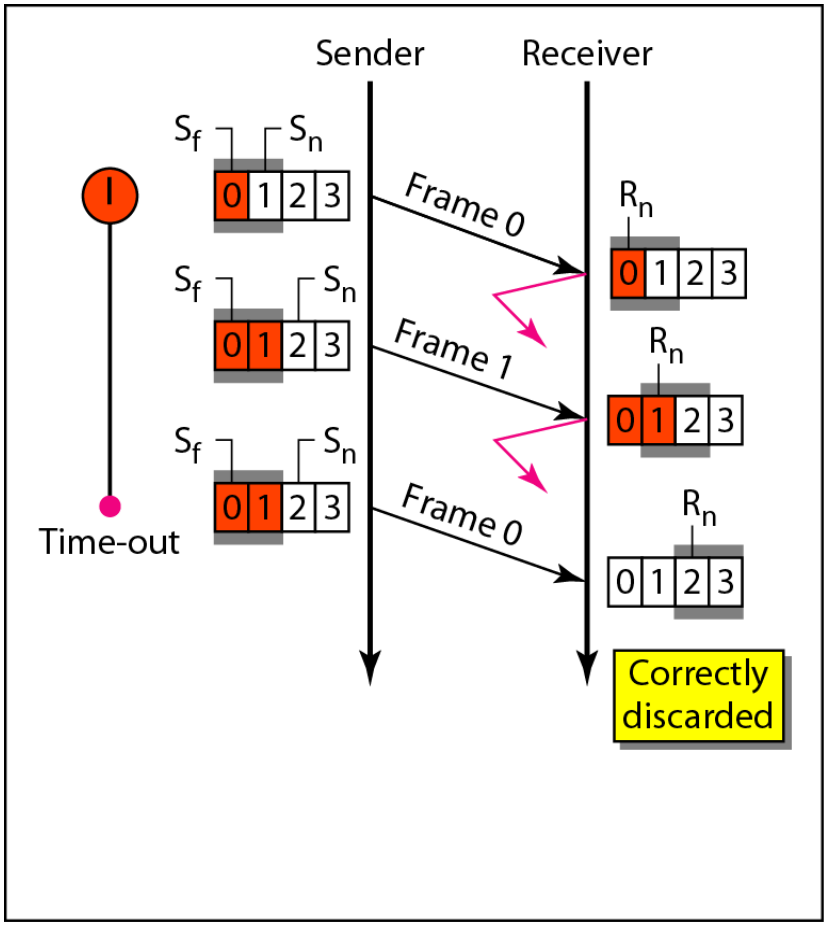
b. After delivery

Negative ACK

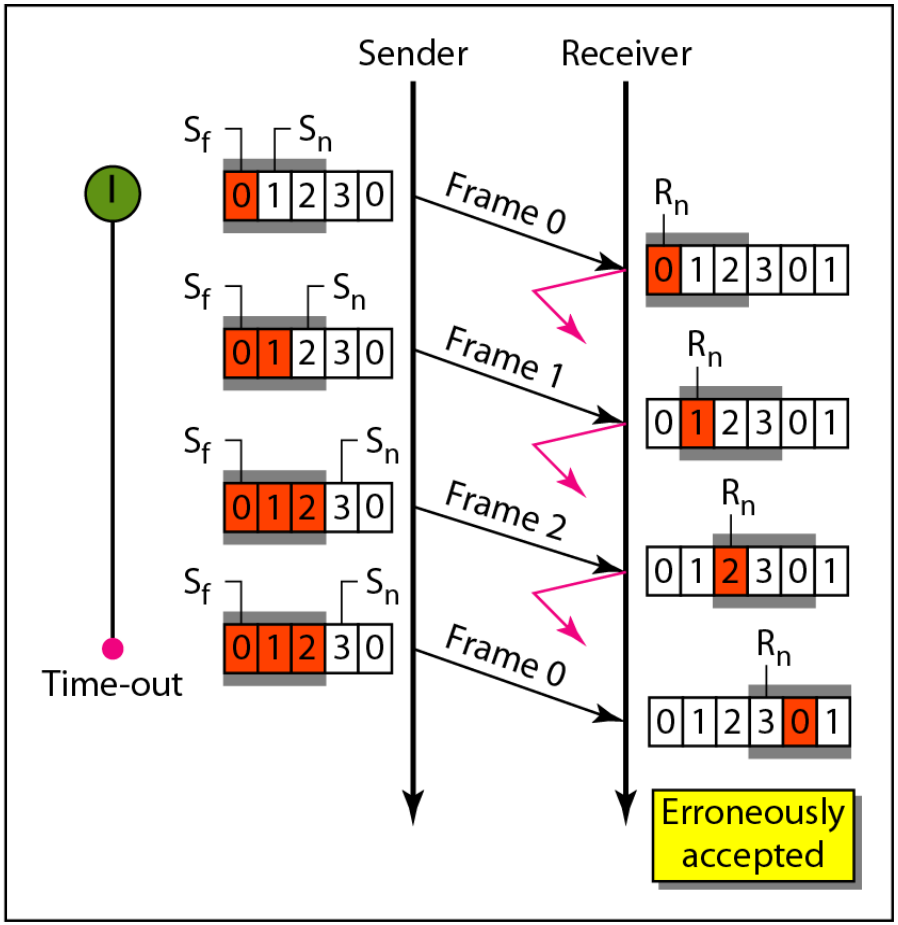
- Used by receiver to indicate missing frame



Selective Repeat: Window Size



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



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

Note:

Refer to the Textbook for protocol's Pseudo Codes.

Bidirectional Transmission

- Data are transferred both ways (*each node has two windows: send window and receive window*)
- ACK are "piggybacked" with data frames

Go-Back-N ARQ using piggybacking

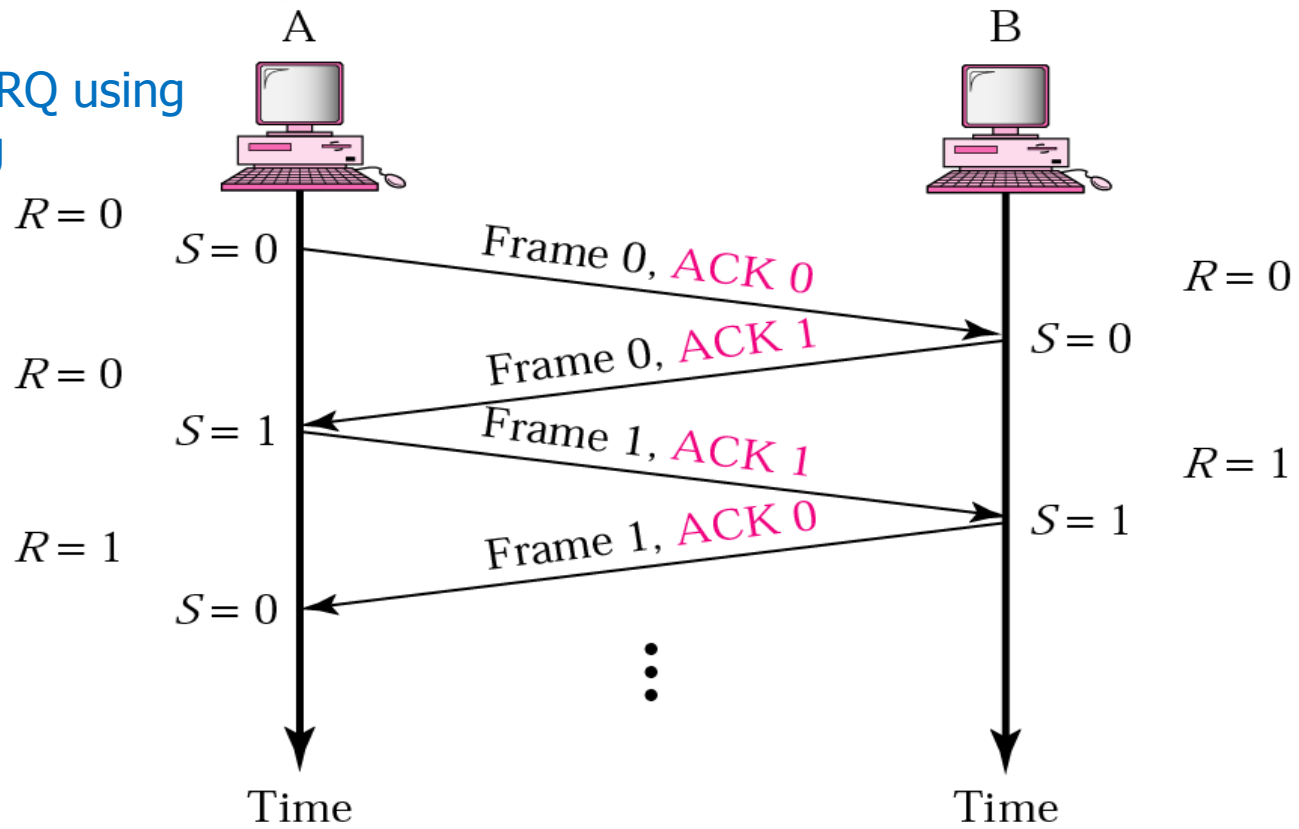
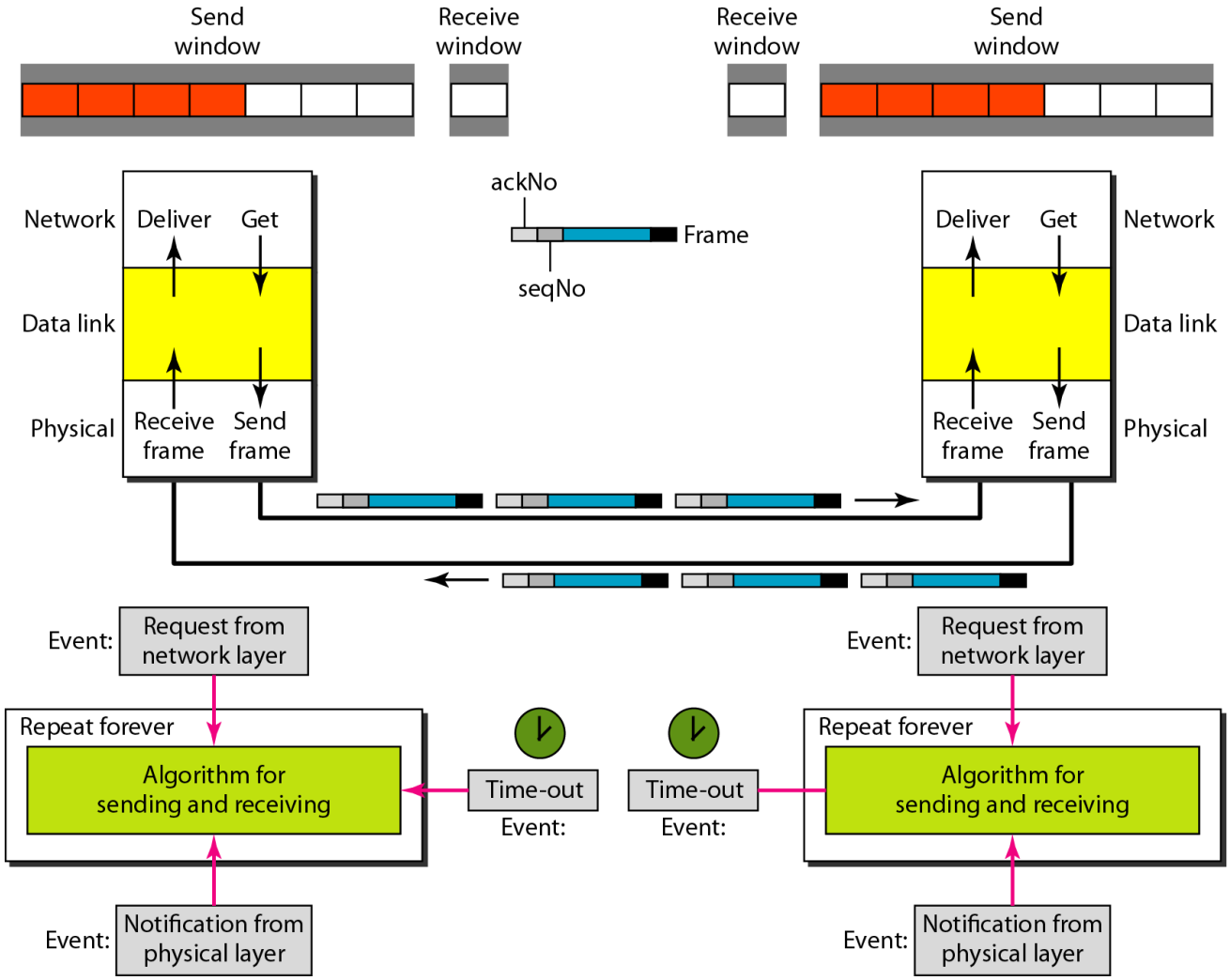


Figure 11.24 *Design of piggybacking in Go-Back-N ARQ*



Data Link Control Protocols:
HDLC and PPP

6:07 AM

11-6 HDLC

- **H**igh-level **D**ata **L**ink **C**ontrol
- Bit-oriented protocol
- Support both
  - Point-to-point links
  - Multipoint links
- Two common transfer modes:
  - Normal Response Mode (NRM)
  - Asynchronous Balanced Mode (ABM).

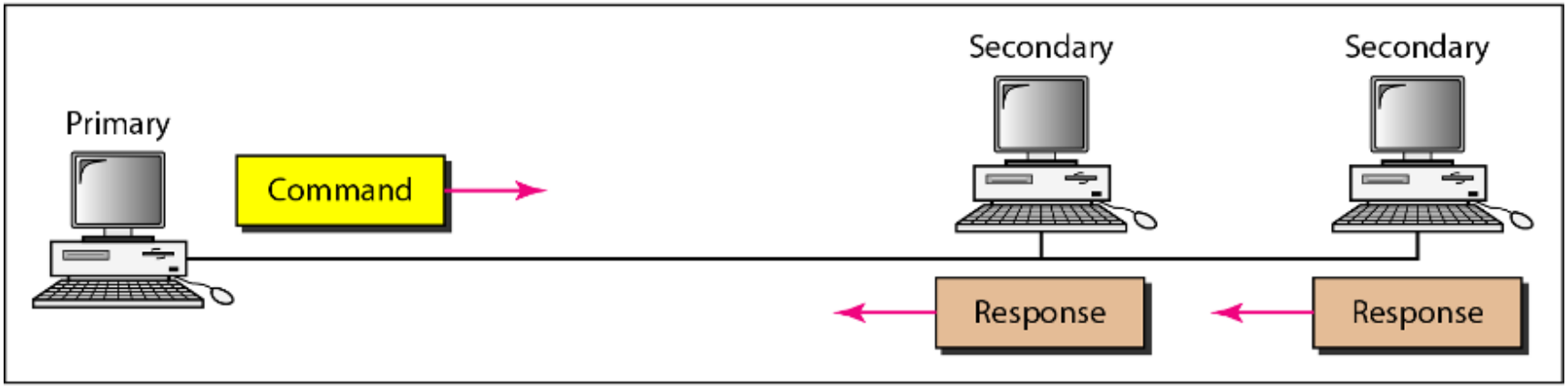
Normal Response Mode

- "NRM" Used in both point-to-point and multi-point link
- We have one primary station and multiple secondary stations. A primary station can send commands; a secondary station can only respond.

Normal Response Mode

- "NRM" Used in both point-to-point and multi-point link

a. Point-to-point



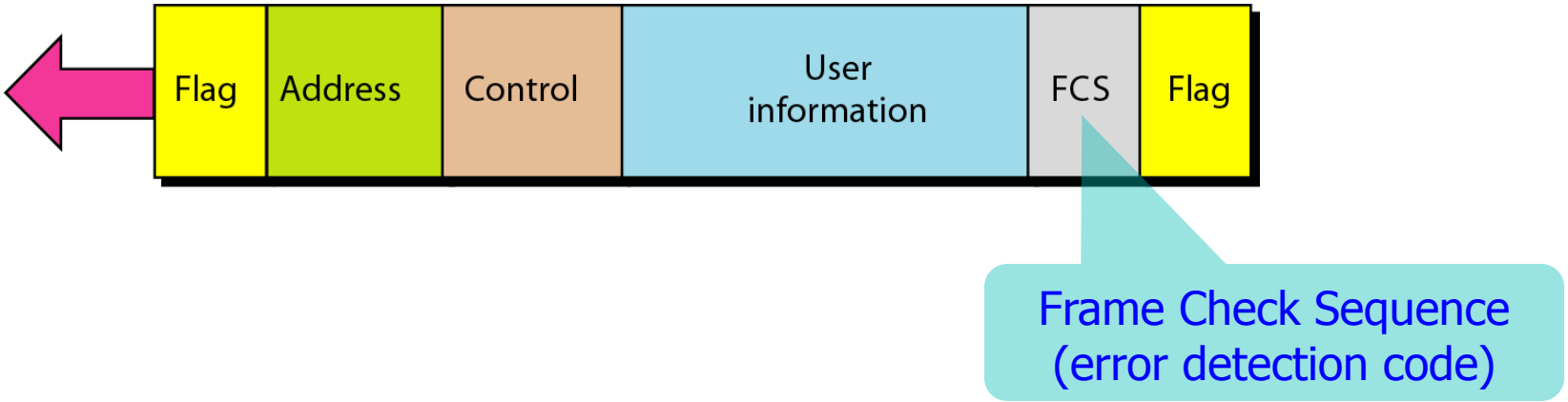
b. Multipoint

Asynchronous Balance Mode

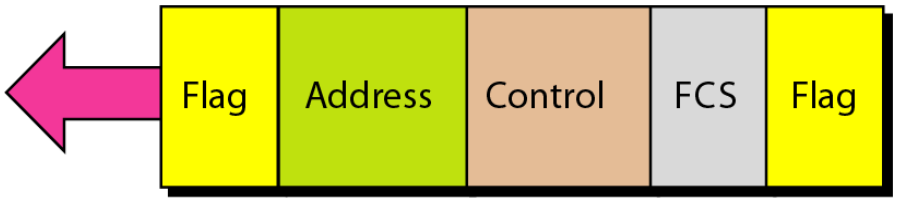
- "ABM"
- Supports only point-to-point links
- Each station is both primary and secondary

HDLC Frames

- Information frame (I-frame): used to transport user data and control information relating to user data (piggybacking)

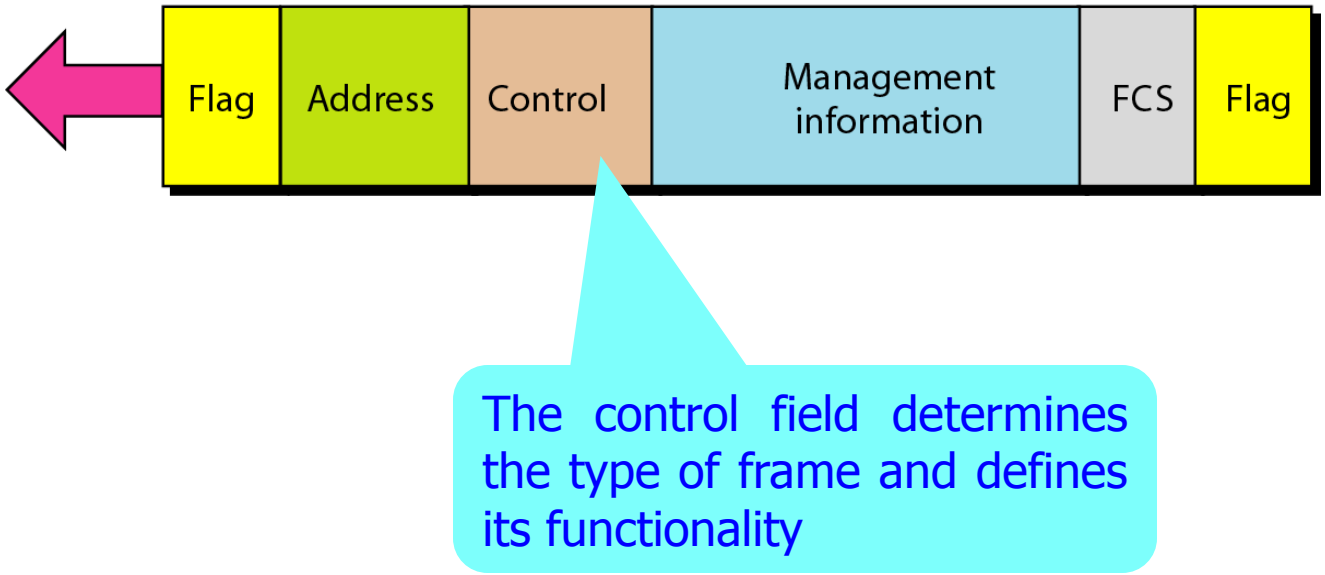


- Supervisory frame (S-frame): used only to transport control information (flow and error control whenever piggybacking is either impossible or inappropriate)

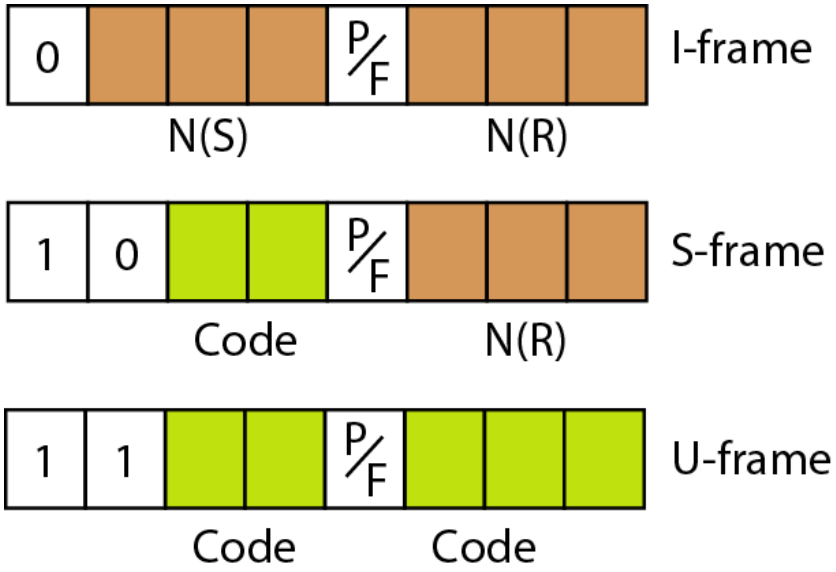


HDLC Frames

- Unnumbered frame (U-frame): reserved for system management. Information carried by U-frames is intended for session management or managing the link itself.



Control Field Format

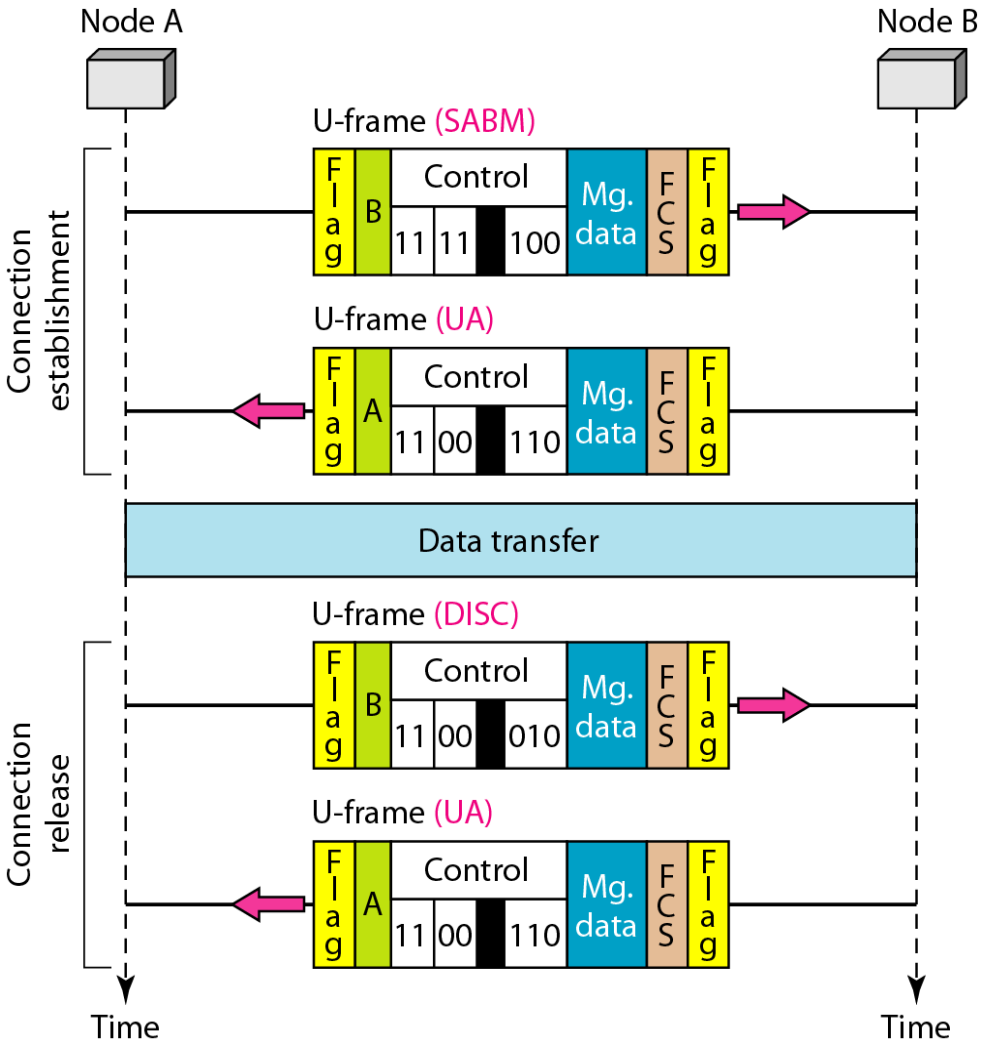


- N(S) – 3-bits Frame sequence number
- N(R) – 3-bits Ack sequence number
- P/F [Poll (primary → secondary) - Final (secondary → primary)] (*It means the address field contains the address of the Sender or Receiver*).)
- S-frame Code: [00= Receive ready (RR), 10= Receive not ready (RNR), 01= Reject (REJ)-GBN, 11= Selective reject (SREJ)-SR]

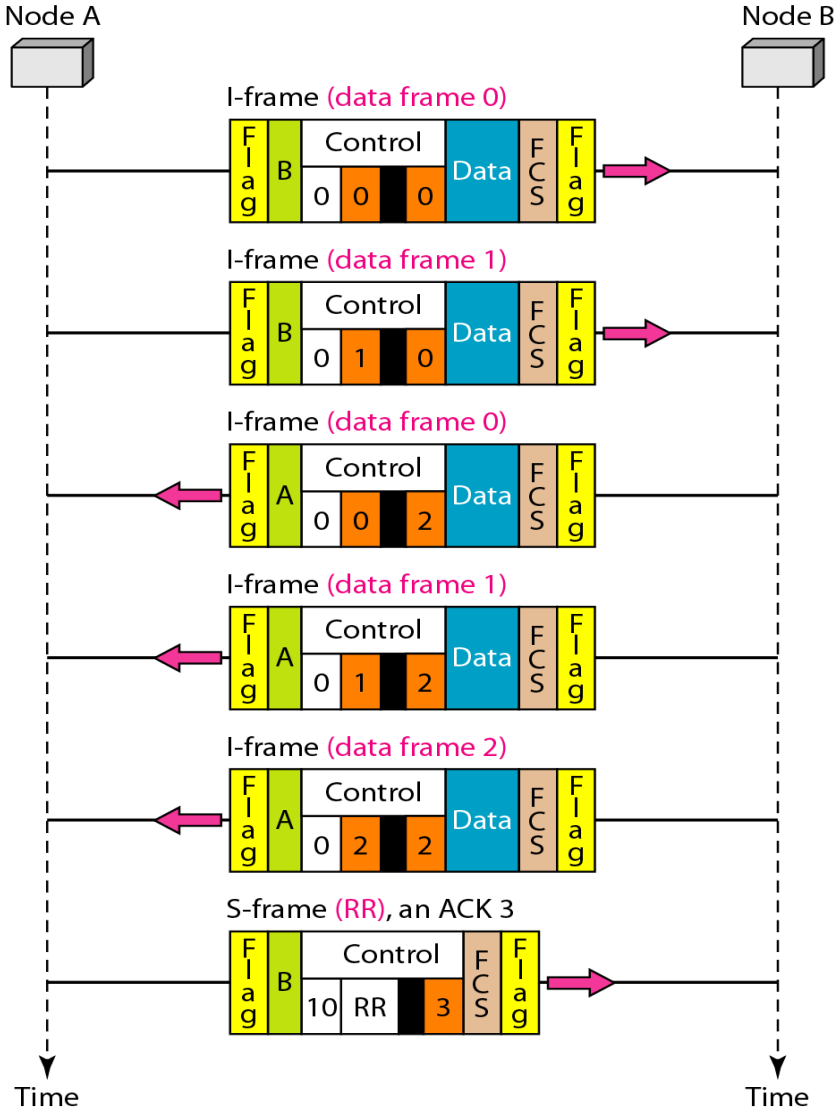
U-Frame Codes

Code	*Command*	*Response*	*Meaning*
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

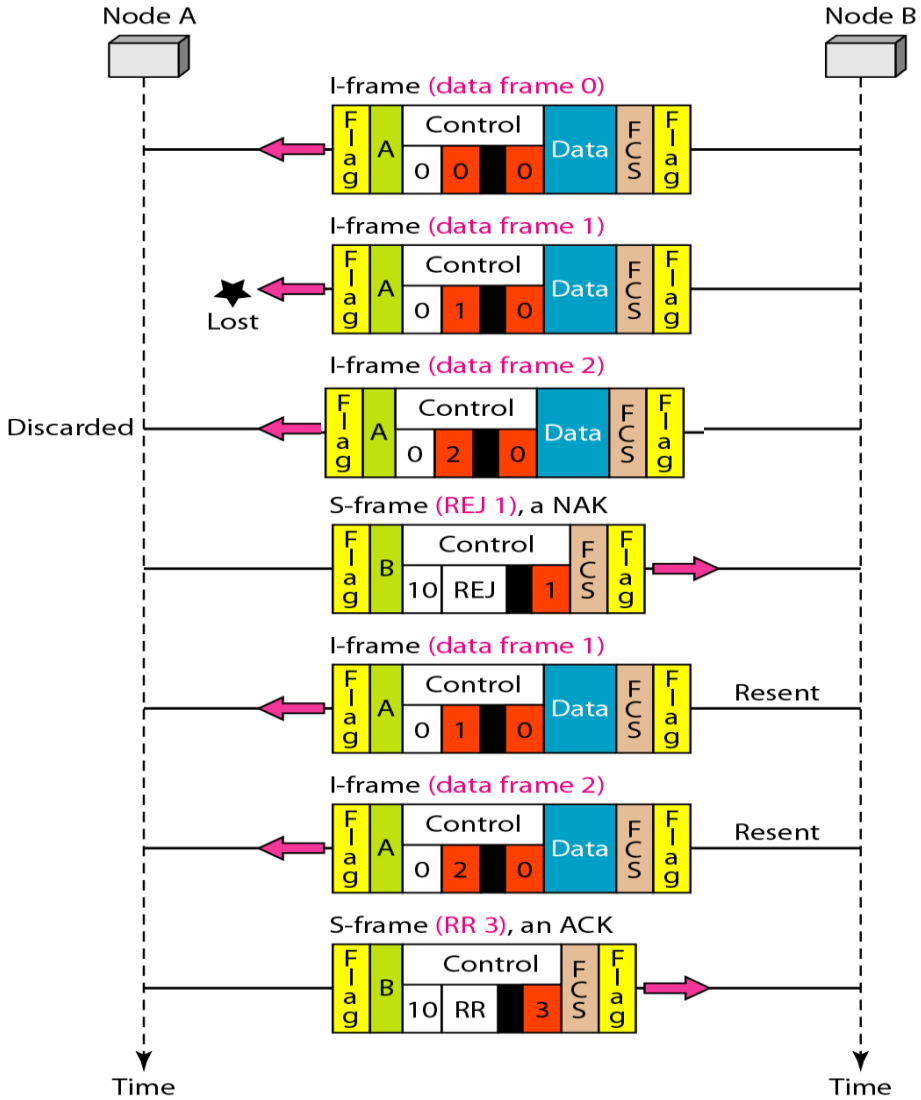
Connection and Disconnection



Data Transfer: *using piggybacking*



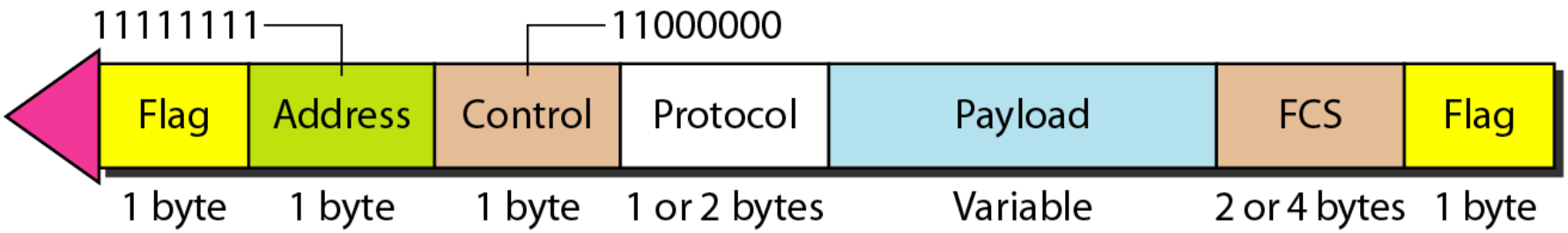
Data Transfer: With Error



11-7 POINT-TO-POINT PROTOCOL

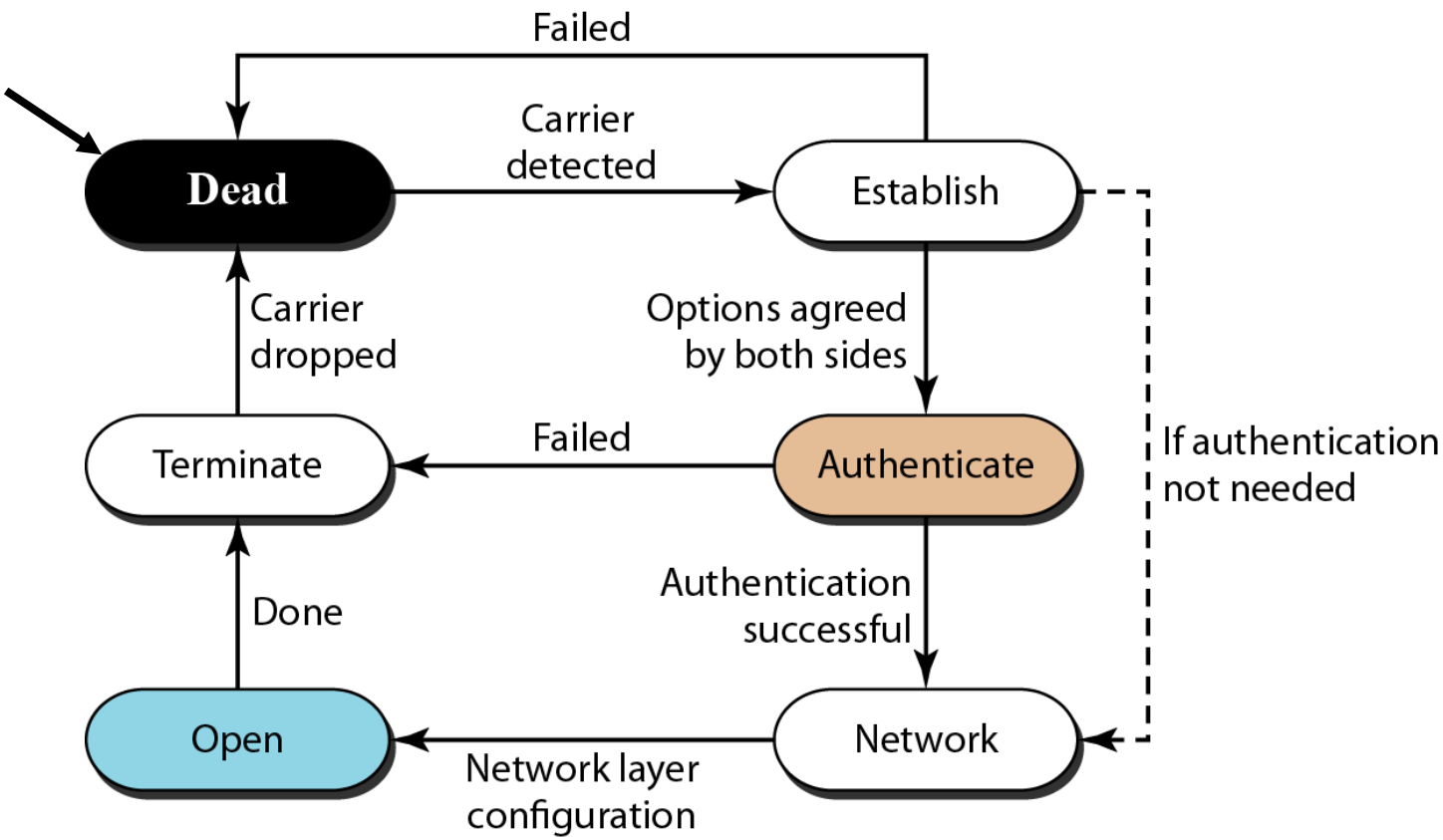
- Point-to-Point Protocol
- Byte-oriented protocol
- Most common protocol for point-to-point access
  - Dial-up access
  - ADSL
  - Cellular Network [2G (GPRS/EDGE), 3G]

PPP Frame Format



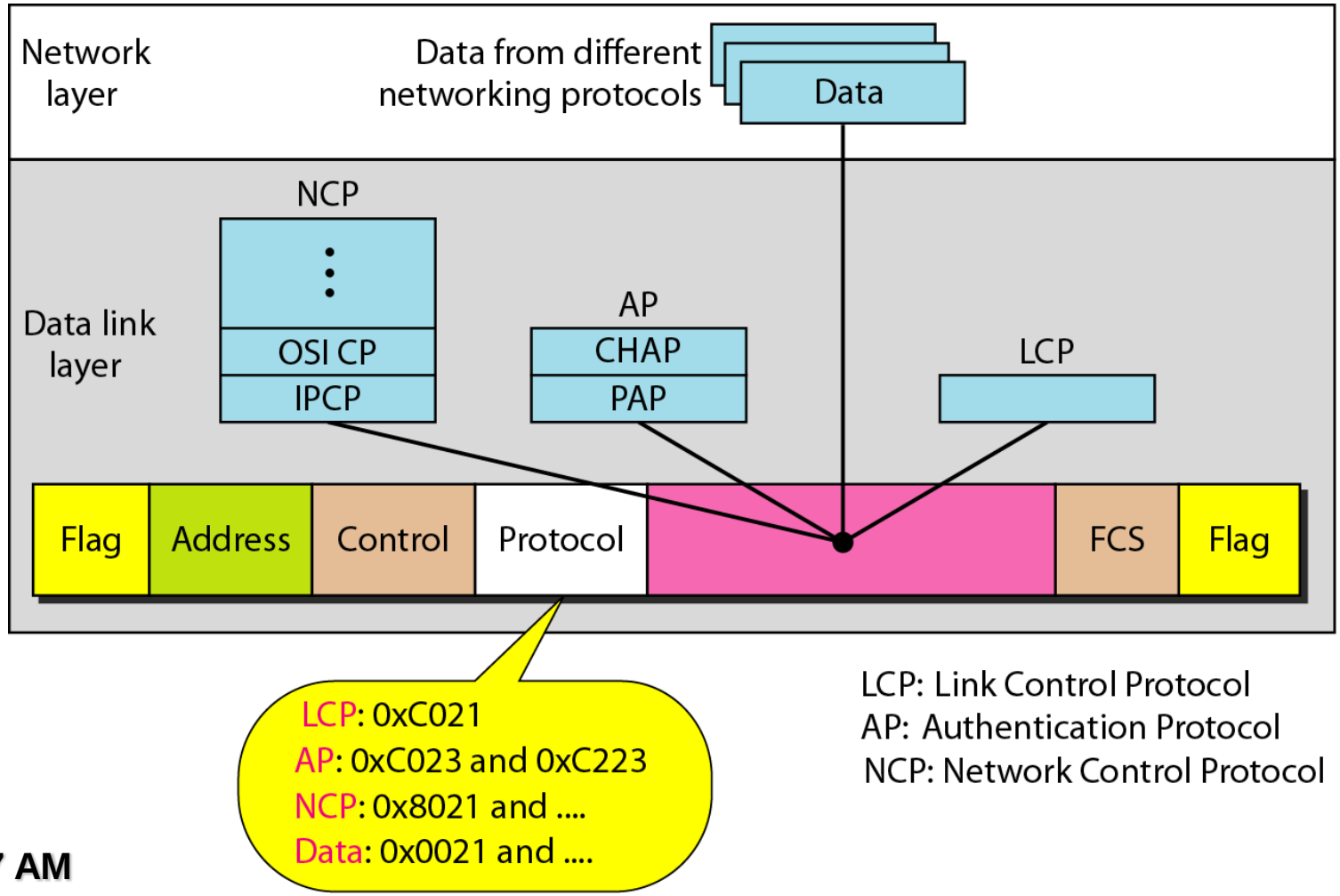
- Escape byte: 01111101
- No flow control with a very simple mechanism for error control (CRC field)
- The address and control fields have constant values and the two parties may agree to omit them.
- The protocol field defines what is being carried in the Payload (maximum 1500 bytes): either user data or other information.

PPP Transition States



Multiplexing in PPP

PPP uses another set of other protocols to establish the link, authenticate the parties involved, and carry the network layer data.

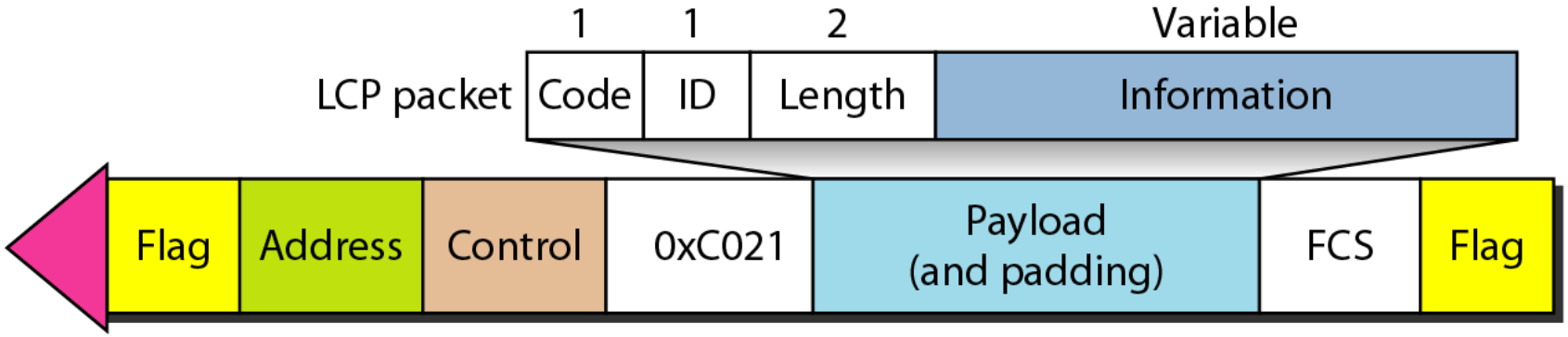


PPP Stack

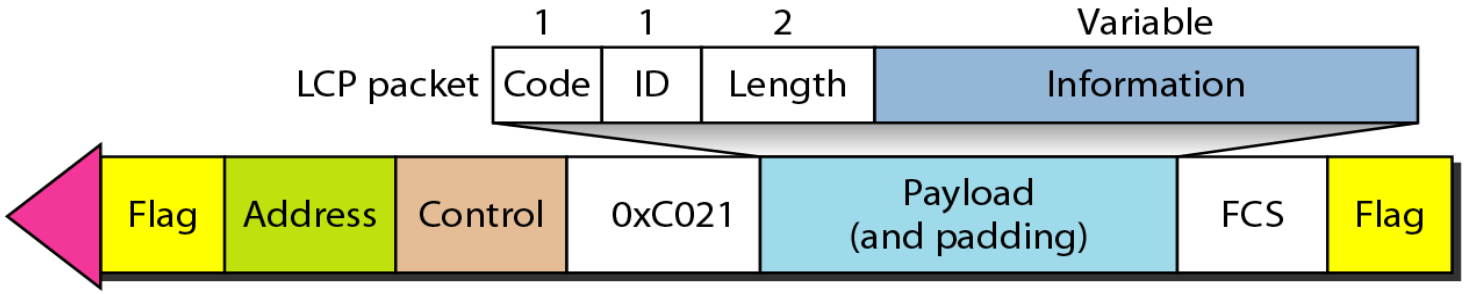
- Link Control Protocol (LCP)
- Authentication Protocol (AP)
- Network Control Protocol (NCP)
- Data Transfer

LCP: Link Control Protocol

- Responsible for establishing, maintaining, configuring, and terminating links. Also it provides negotiation mechanisms to set options between the two endpoints.



LCP: Link Control Protocol

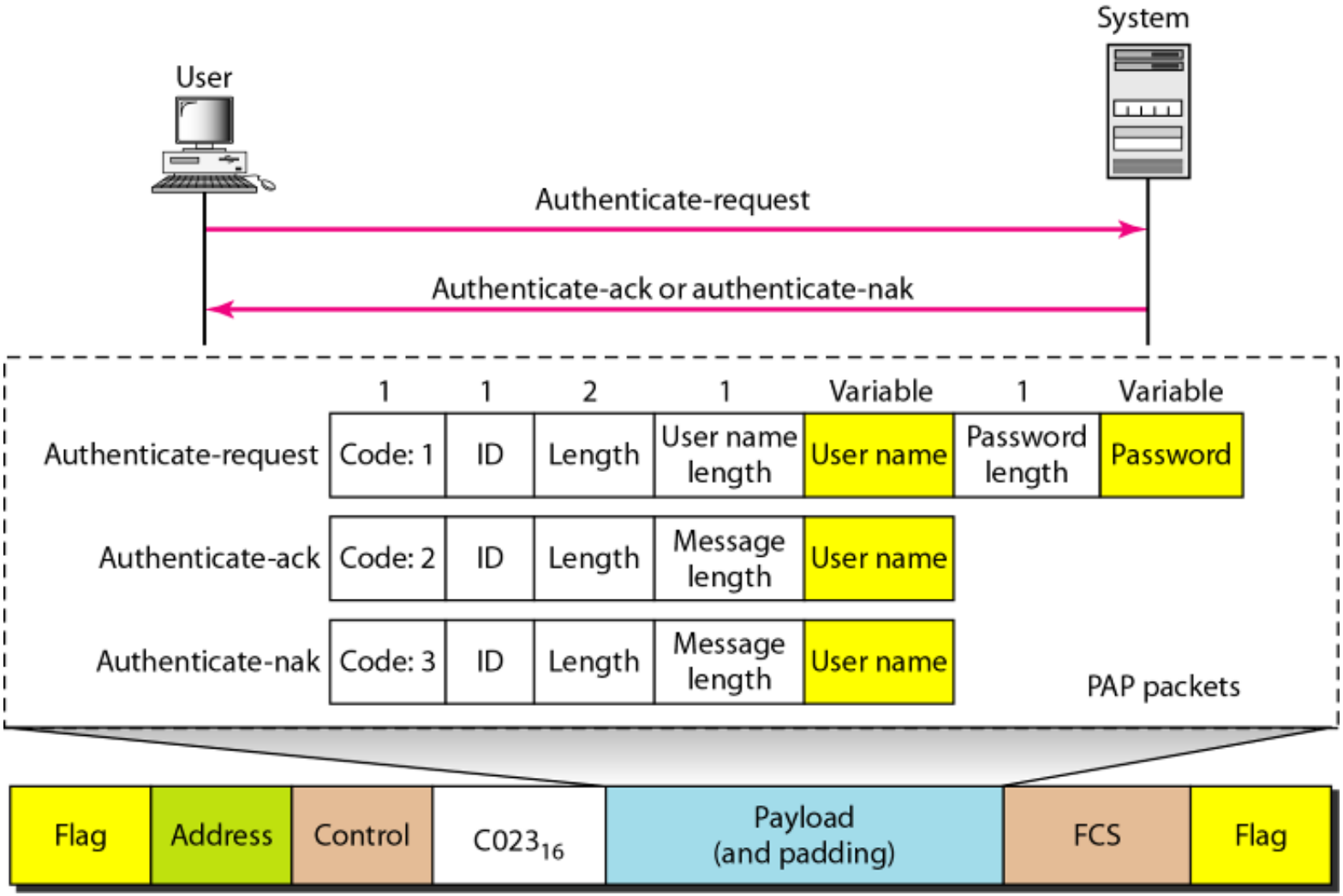


- Code field: defines the type of LCP packet. There are 11 types of packets as shown in Table 11.2.
- ID field: holds a value that matches a request with a reply.
- Length field: defines the length of the entire LCP packet.
- information field: contains options information, needed for some LCP packets (listed in Table 11.3.)

PPP Authentication

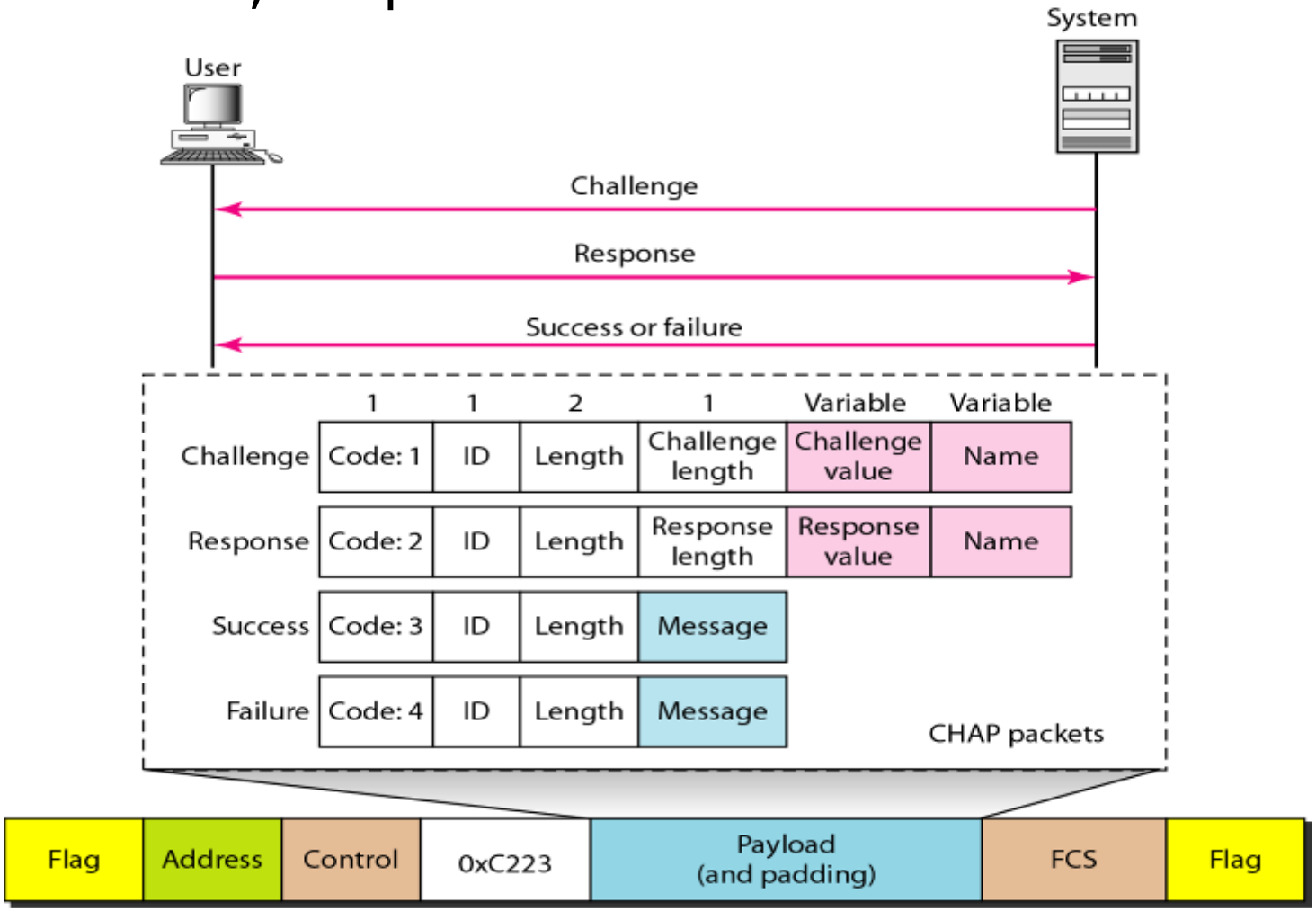
- Two protocols are supported:
  - Password Authentication Protocol (PAP)
  - Challenge Handshake Authentication Protocol (CHAP)

PAP


System
User
Authenticate-request
Authenticate-ack or authenticate-nak
1
1
2
1
Variable
1
Variable
Authenticate-request
Code: 1
ID
Length
User name length
User name
Password length
Password
Authenticate-ack
Code: 2
ID
Length
Message length
User name
Authenticate-nak
Code: 3
ID
Length
Message length
User name
PAP packets
Flag
Address
Control
C023$_{16}$
Payload (and padding)
FCS
Flag

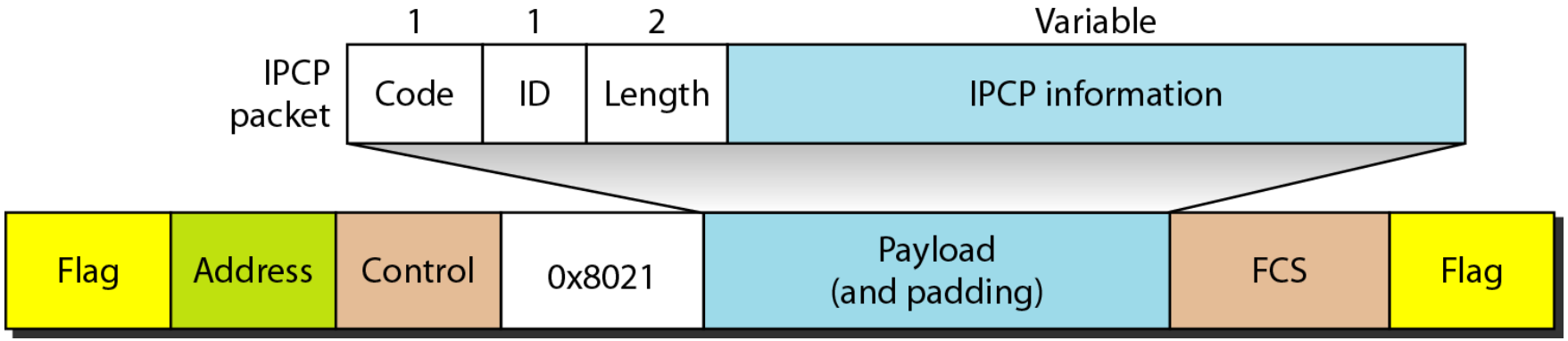
CHAP

In this method, the password is never sent online.



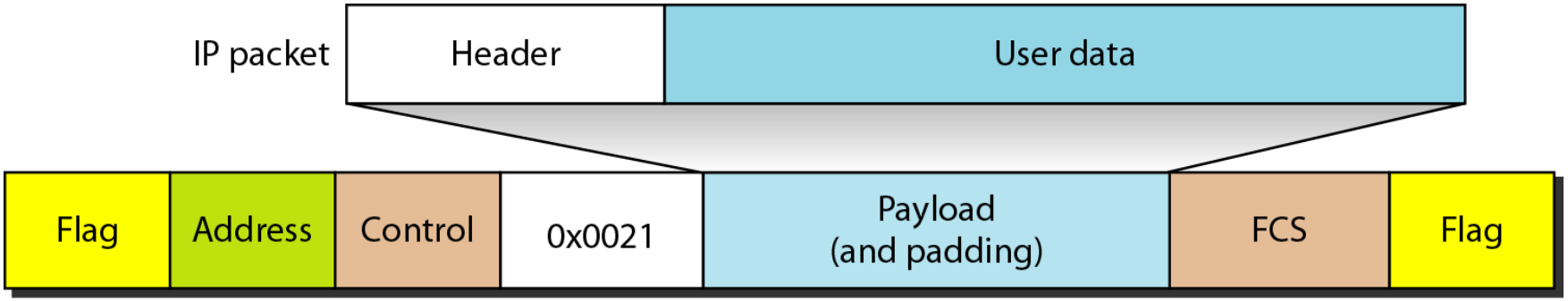
NCP: Network Control Protocol

- A set of control protocols to configures the link for carrying data packets of different network layer protocols (Internet, OSI, Xerox, DECnet) so as to be encapsulated into a PPP frame
  - Note that none of the NCP packets carry network layer data; they just configure the link at the network layer for the incoming data
- Eg. IPCP (Internetwork Protocol Control Protocol) configures the link for carrying IP data packets.

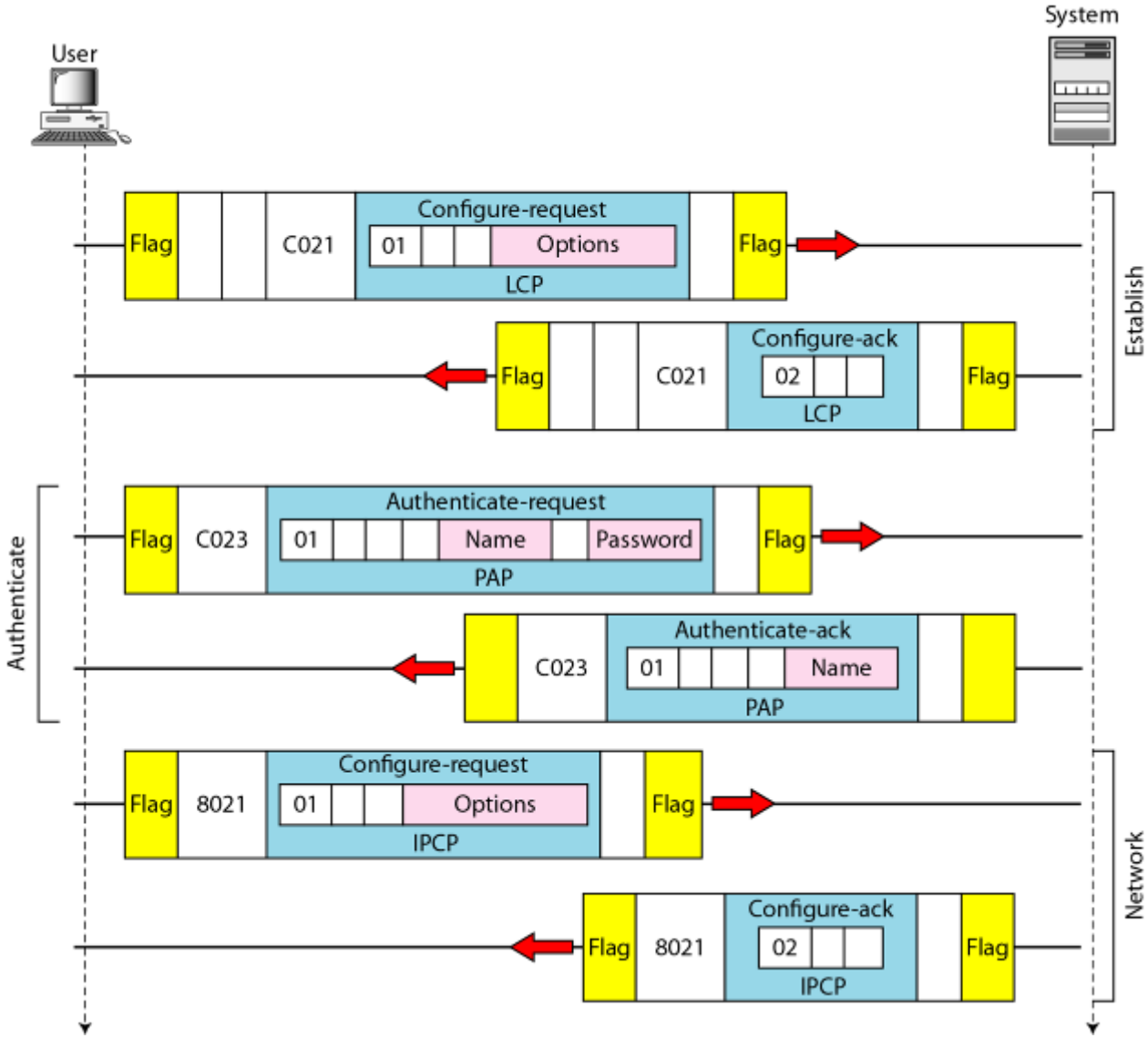


Data Encapsulation

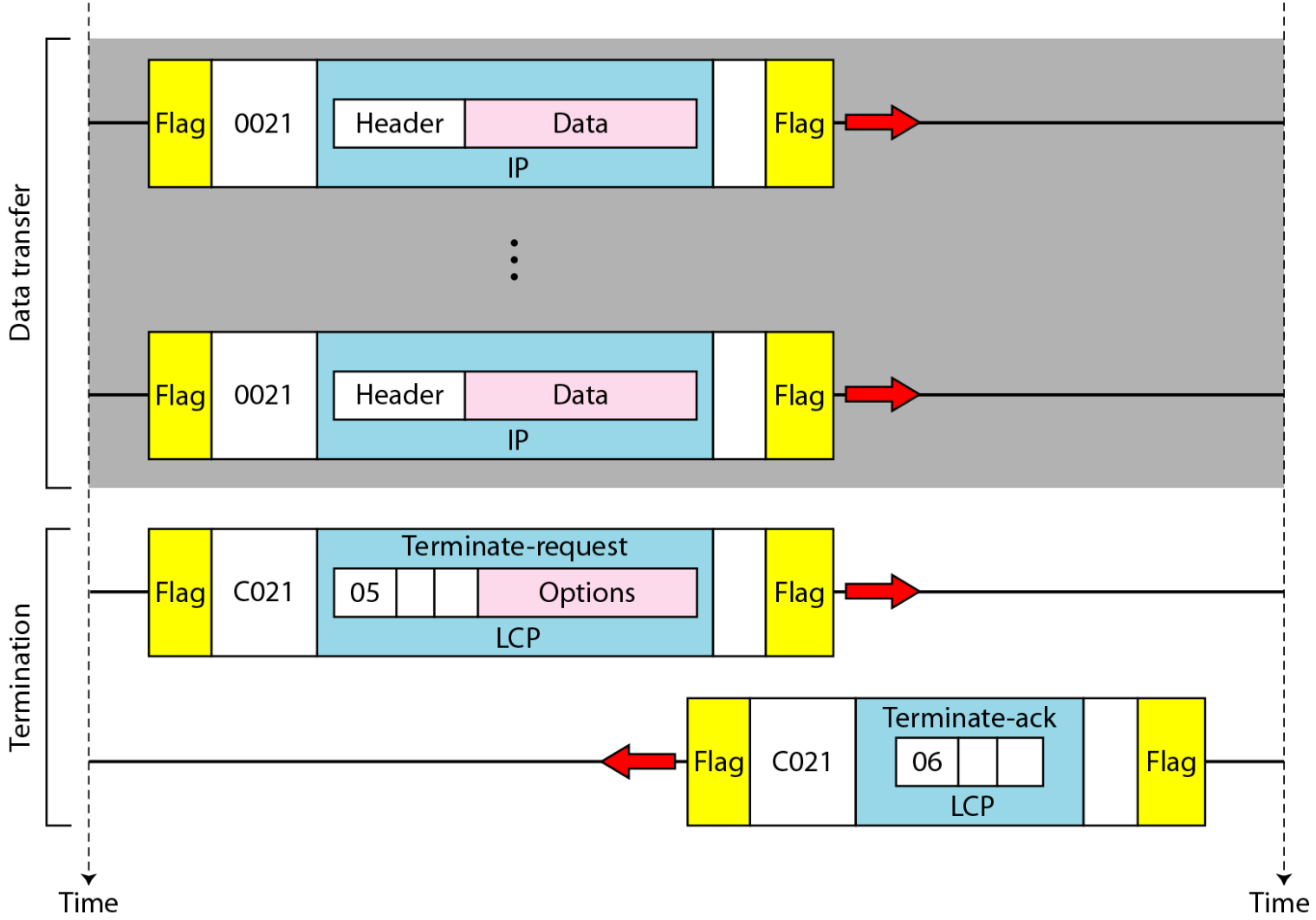
- After the network layer configuration is completed by one of the NCP protocols, the users can exchange data packets from the network layer.
- Here again, there are different protocol fields for different network layers. For example, if PPP is carrying data from the IP network layer, the field value is 0021 (note that the three rightmost digits are the same as for IPCP)



Example: PPP Session



Example: PPP Session (cont'd)



تم بحمد الله