Module 3
Data Link Layer
Logical Link Control

BECE401L

Outline

Error Detection Techniques

ARQ protocols

Framing

HDLC

Point to Point protocol

Data Link Layer: Addressing

IP addresses as the identifiers at *the network layer*.

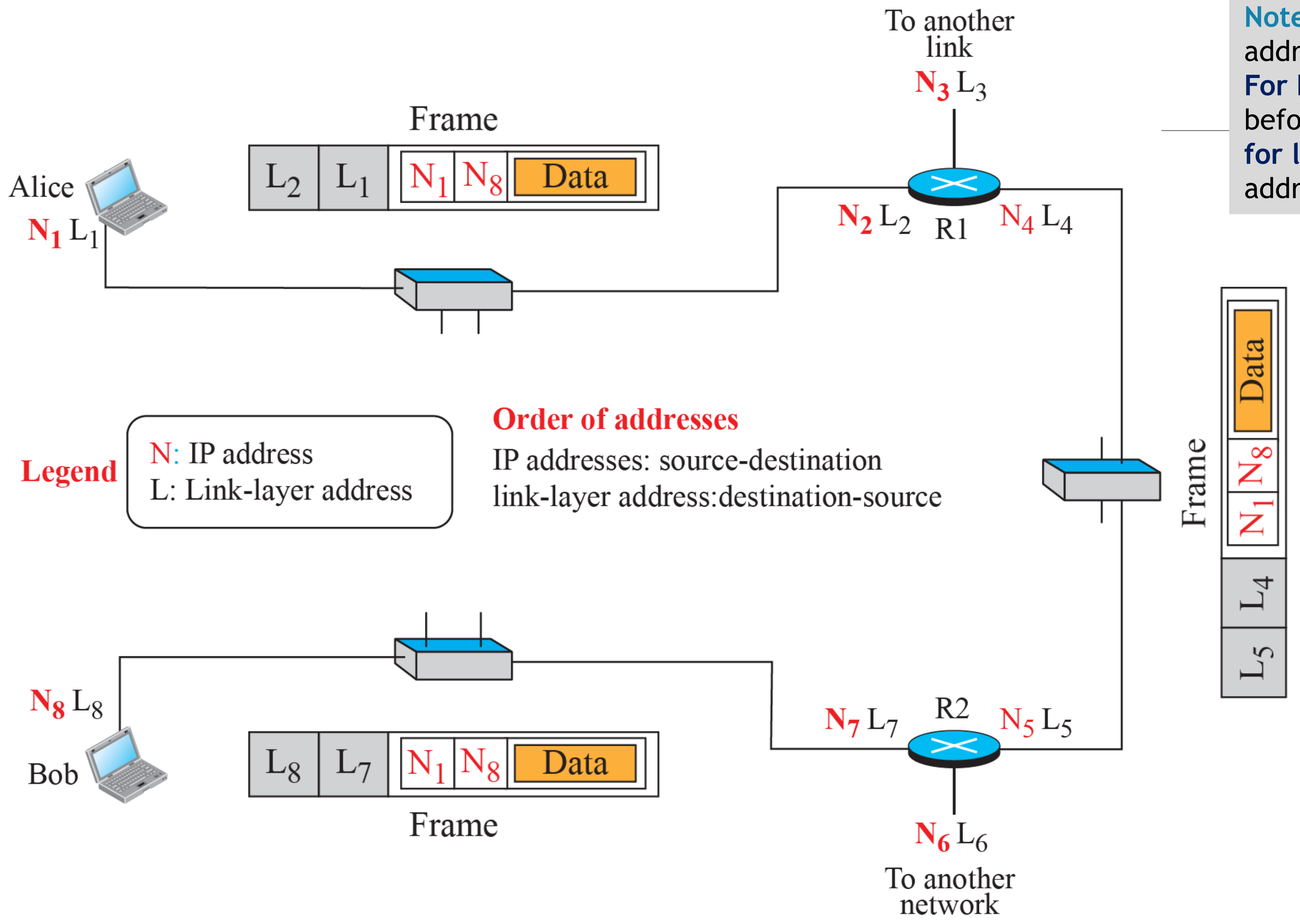
In a connectionless internetwork such as the Internet

- A datagram cannot reach its destination using only IP addresses.
- Each datagram in the Internet **may take a different path**.

Another addressing mechanism is needed in a connectionless internetwork:

- **The link-layer addresses of the two nodes.**

Link-layer address is sometimes called a ***link address***, sometimes a ***physical address***, and sometimes a ***MAC address***.



Note that the IP addresses and the link-layer addresses are not in the same order.
For IP addresses, the source address comes before the destination address;
for link-layer addresses, the destination address comes before the source.

Figure: IP addresses and link-layer addresses in a small internet

Data Link Layer: Addressing Examples

Unicast Address

- Unicasting means one-to-one communication.
- A frame with a unicast address destination is ***destined only for one entity in the link***.

Example:

The unicast link-layer addresses in the most common LAN, Ethernet, are 48 bits (six bytes) that are presented as 12 hexadecimal digits separated by colons; for example, the following is a link-layer address of a computer. The second digit needs to be an odd number.

A3:34:45:11:92:F1

Data Link Layer: Addressing Examples

Multicast Address

- Multicasting means one-to-many communication.

Example:

The second digit, however, needs to be an even number in hexadecimal. The following shows a multicast address:

A2:34:45:11:92:F1

Data Link Layer: Addressing Examples

Broadcast Address

- Broadcasting means one-to-all communication.
- A frame with a destination broadcast address is sent to all entities in the link.

Example:

The broadcast link-layer addresses in the most common LAN, Ethernet, are 48 bits, all 1s, that are presented as 12 hexadecimal digits separated by colons. The following shows a broadcast address:

FF:FF:FF:FF:FF:FF

Data Link Layer: Address Resolution Protocol

IP address of the receiving node is required.

- By a **sender node has an IP datagram** to **send to another node** in a link.
- **Source host knows the IP address** of **the default router**.
- The **last router knows the IP address** of **the destination host**.

However, **the IP address of the next node is not helpful in moving a frame** through a link; Link-layer address of the next node.

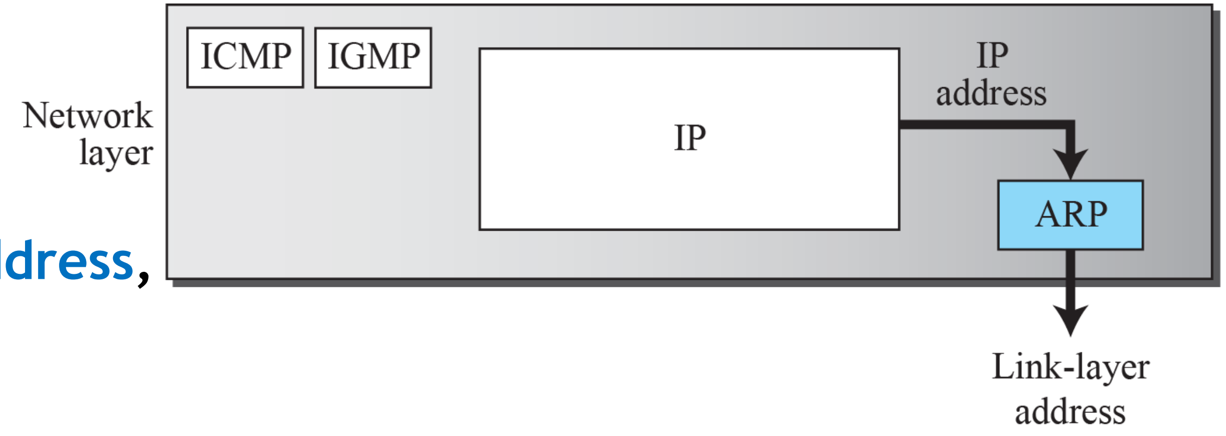
This functionality is provided by Address Resolution Protocol (ARP).

- ARP is auxiliary protocols defined in the network layer.
- Maps an IP address to a logical-link address.

ARP accepts an IP address from the IP protocol,

Maps the address to the corresponding link-layer address,

and Passes it to the data-link layer.



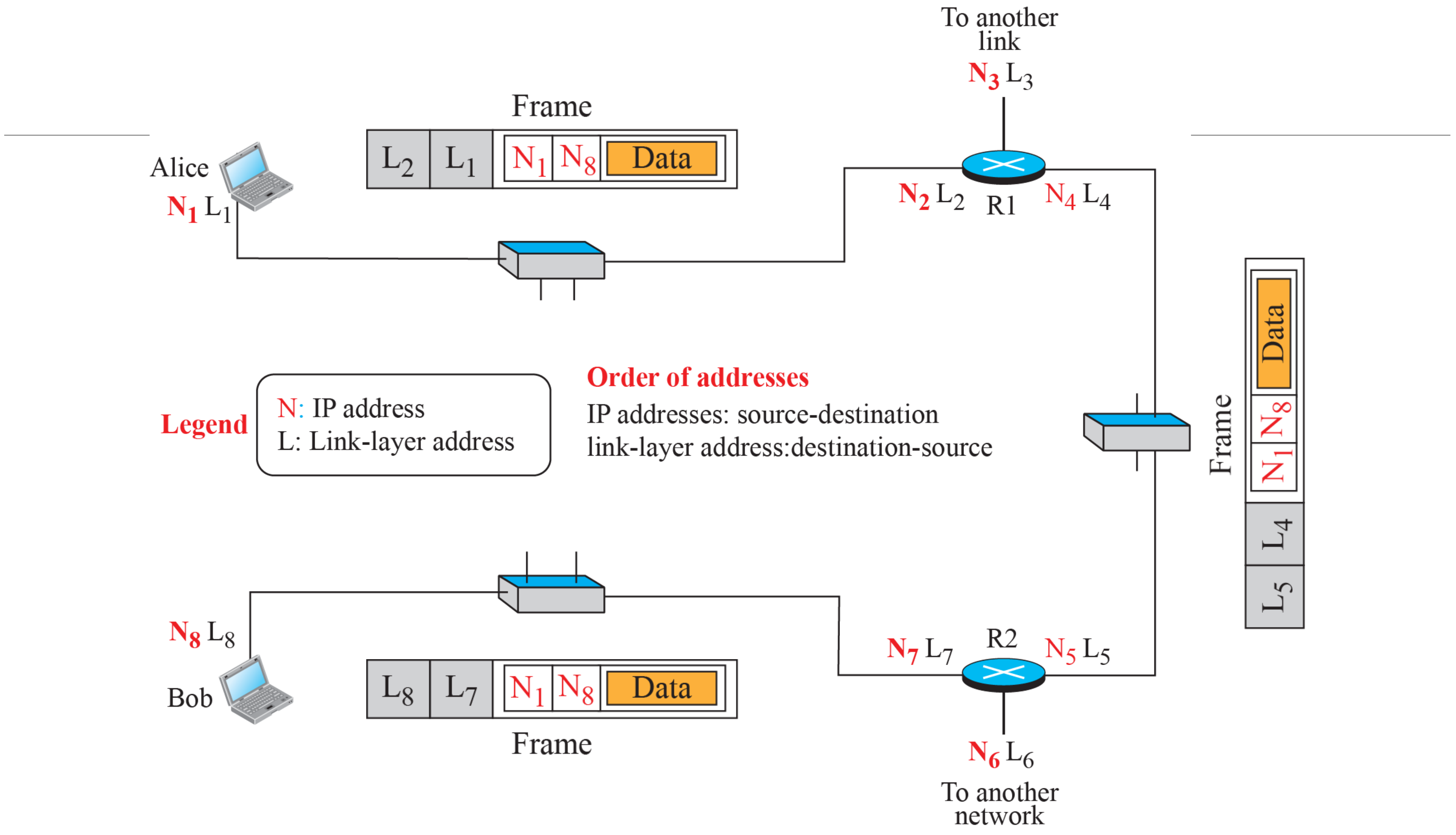
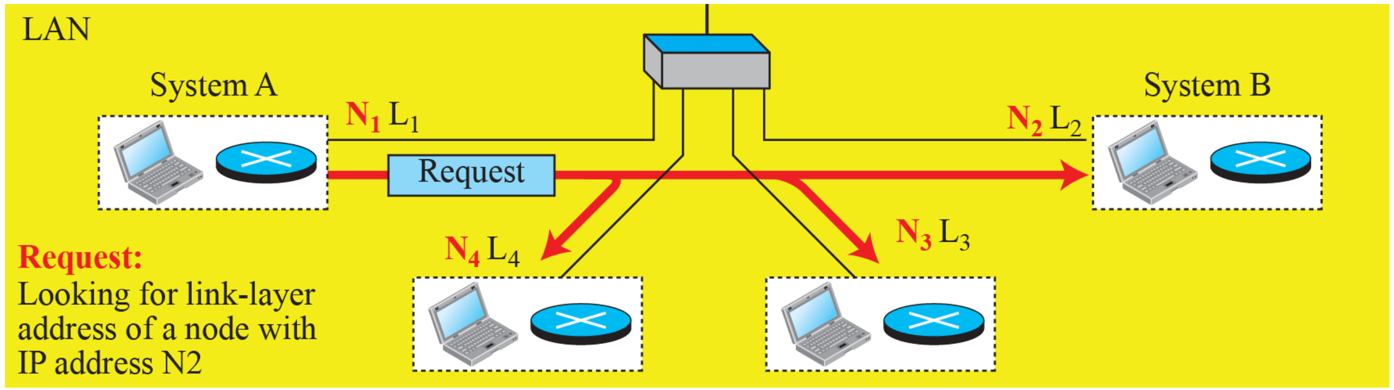


Figure: IP addresses and link-layer addresses in a small internet

ARP Operation

- Anytime a host or a router **needs to find the link-layer address of another host or router** in its network, it **sends an ARP request packet.**
  - Packet includes the **link-layer and IP addresses of the sender** and
  - IP address of the receiver.
- As the **sender does not know the link-layer address of the receiver**,
  - **Query is broadcast** over the link using the link-layer broadcast address,
  - Every host or router on the network receives and processes the ARP request packet,
  - but only the intended recipient recognizes its IP address and sends back an ARP response packet.
- The response packet contains the recipient's IP and link-layer addresses.
  - The packet is unicast directly to the node that sent the request packet.



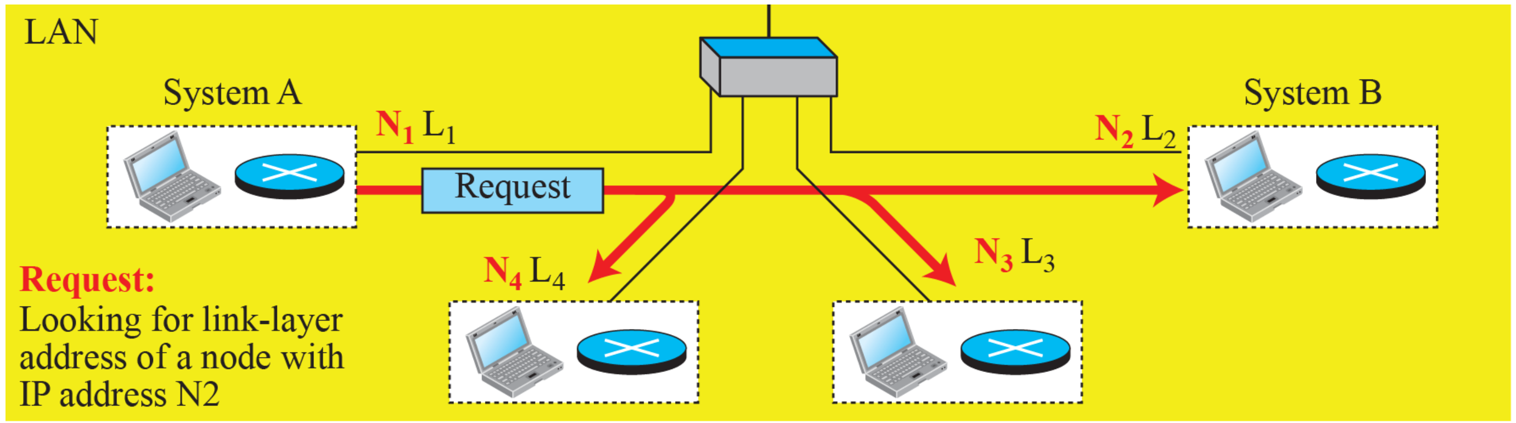
a. ARP request is broadcast

ARP Operation

In Figure A

- The system on the left (A) has a packet that needs to be delivered to another system (B) with IP address N2.
- System A needs to pass the packet to its data-link layer for the actual delivery,
  - but it does not know the physical address of the recipient.

- It uses the **services of ARP** by asking the ARP protocol to send *a broadcast ARP request packet to ask for the physical address of a system with an IP address of N2*.



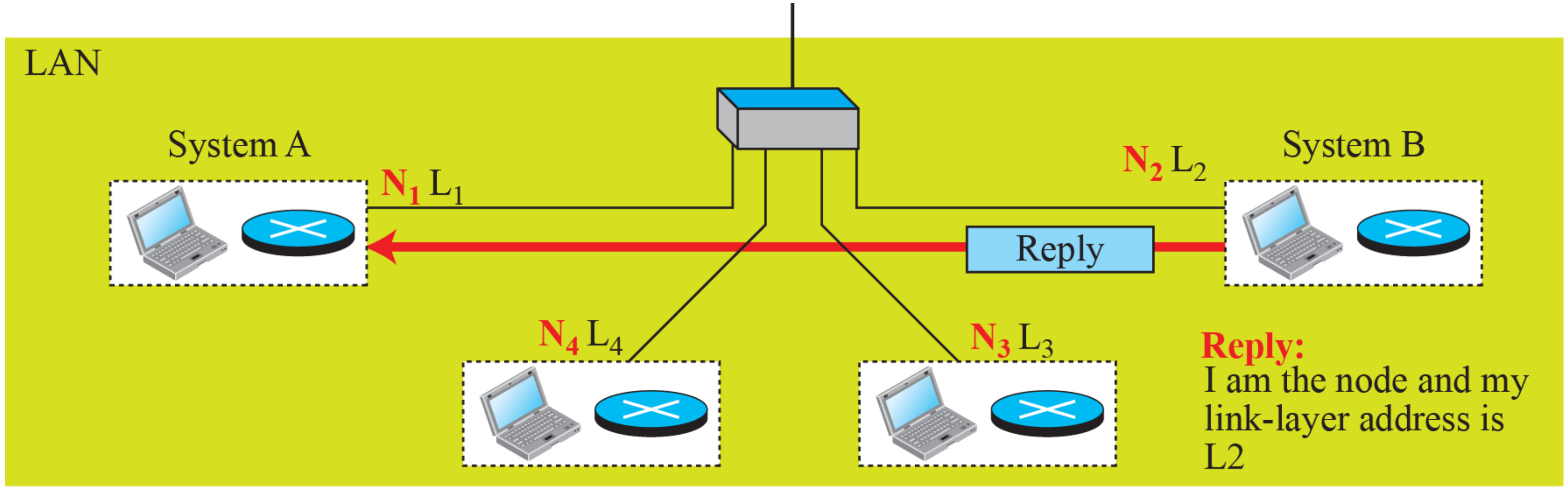
a. ARP request is broadcast

ARP Operation

This **packet is received by every system on the physical network**, but **only system B will answer it**, as shown in Figure B.

System B sends an ARP reply packet that includes its physical address.

Now system A can send all the packets it has for this destination using the physical address it received.



b. ARP reply is unicast

ARP Operation: Caching

Question:

If system A can broadcast a frame to find the link layer address of system B, why can't system A send the datagram for system B using a broadcast frame?

In other words, instead of sending one broadcast frame (ARP request), one unicast frame (ARP response), and another unicast frame (for sending the datagram), system A can encapsulate the datagram and send it to the network.

System B receives it and keep it; other systems discard it.

Answer: Efficiency.
It is probable that system A has more than one datagram to send to system B in a short period of time.

ARP Operation: Caching

Answer: Example

If system B is supposed to receive a long e-mail or a long file, the data do not fit in one datagram.
Let us assume that there are 20 systems connected to the network (link): system A, system B, and 18 other systems.
We also assume that system A has 10 datagrams to send to system B in one second.

a. **Without using ARP,**

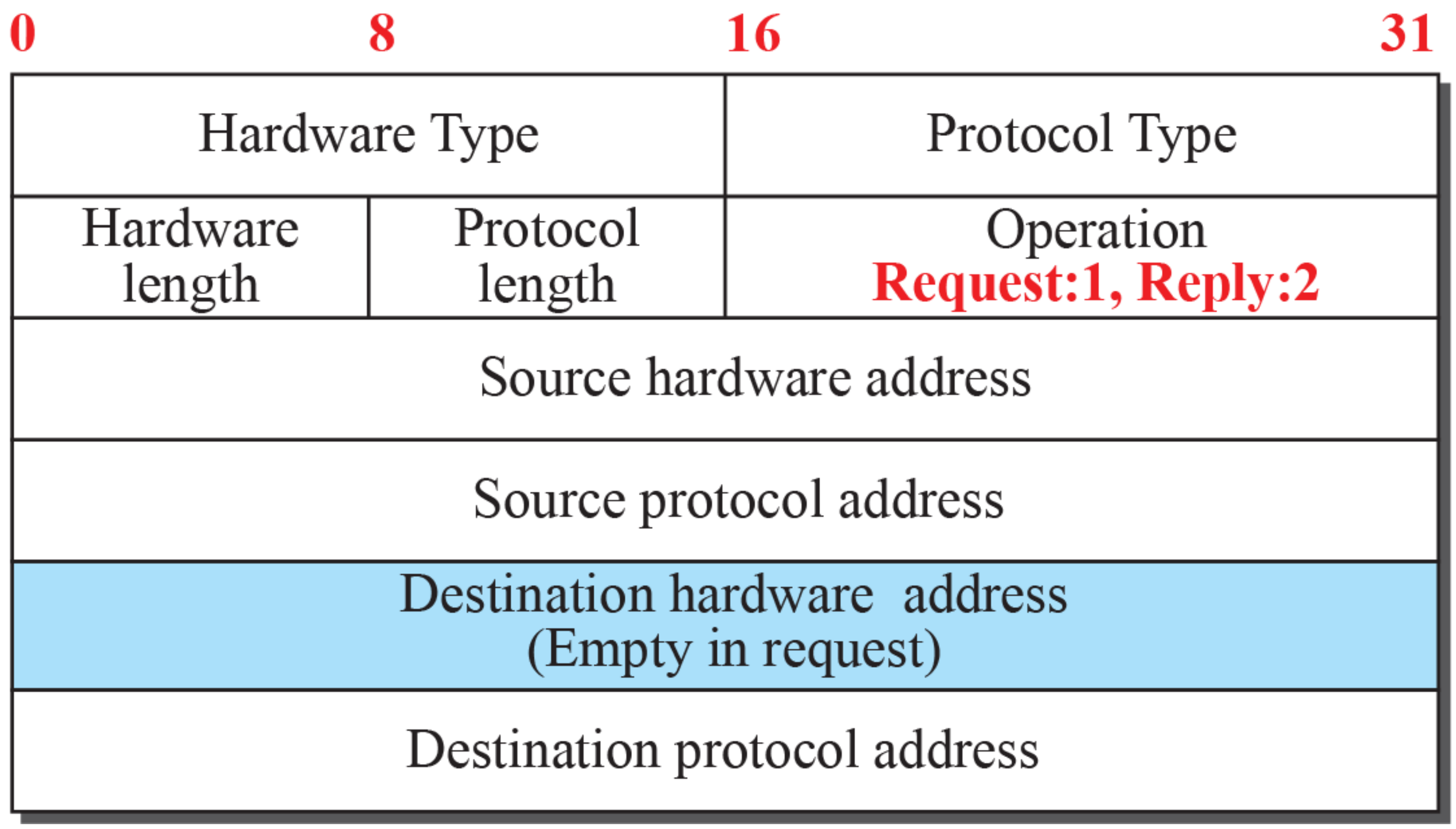
- System A needs to send 10 broadcast frames.
- Each of the *18 other systems need to receive the frames, decapsulate the frames, remove the datagram and pass it to their network-layer to find out the datagrams do not belong to them.*
- This means processing and discarding 180 broadcast frames.

b. **With ARP,**

- System A needs to send **only one broadcast frame**.
- Each of the *18 other systems need to receive the frames, decapsulate the frames, remove the ARP message and pass the message to their ARP protocol to find that the frame must be discarded*.
- This means processing and discarding only 18 (instead of 180) broadcast frames.
- After system B responds with its own data-link address,
- ***System A can store the link-layer address in its cache memory.***
- The ***rest of the nine frames are only unicast***.
- Since processing broadcast frames is expensive (time consuming).

ARP Packet

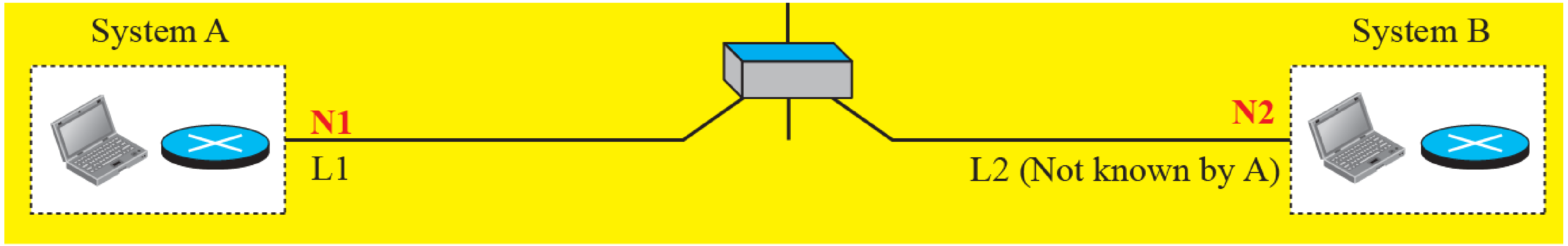
Hardware: LAN or WAN protocol
Protocol: Network-layer protocol



Data Link Layer: **ARP Example**

Example:

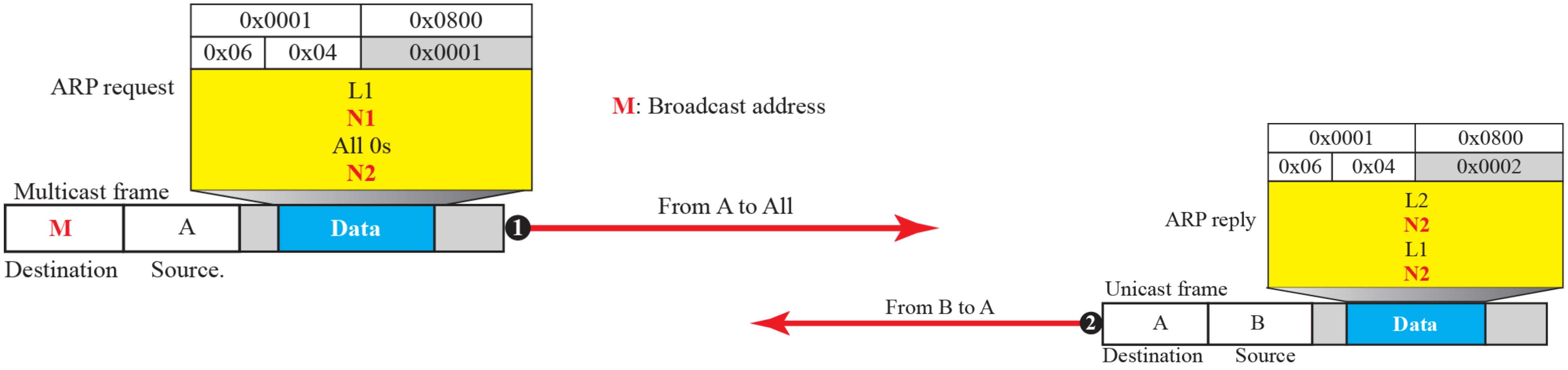
A host with IP address N1 and MAC address L1 has a packet to send to another host with IP address N2 and physical address L2 (which is unknown to the first host). The two hosts are on the same network.



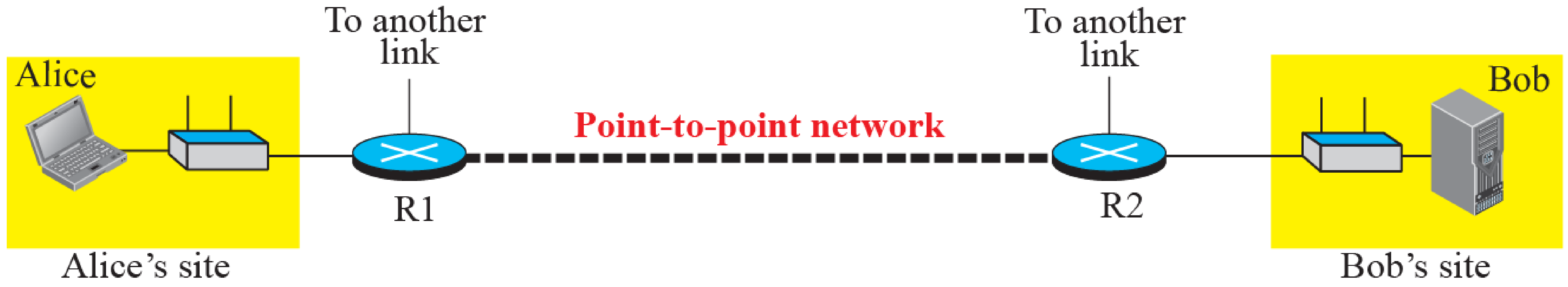
Data Link Layer: ARP Example

Example:

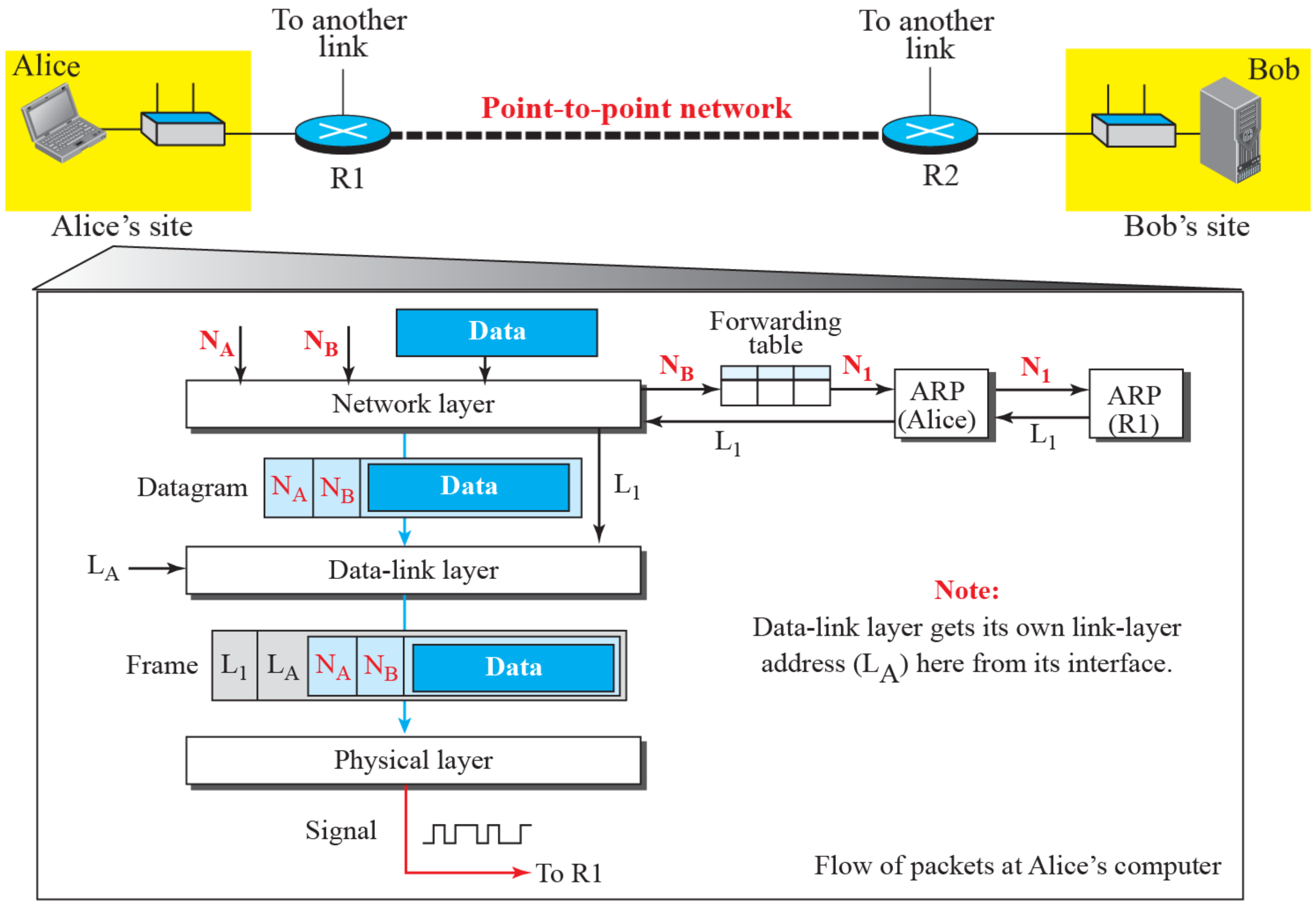
A host with IP address N1 and MAC address L1 has a packet to send to another host with IP address N2 and physical address L2 (which is unknown to the first host). The two hosts are on the same network. Figure shows the ARP request and response messages.:



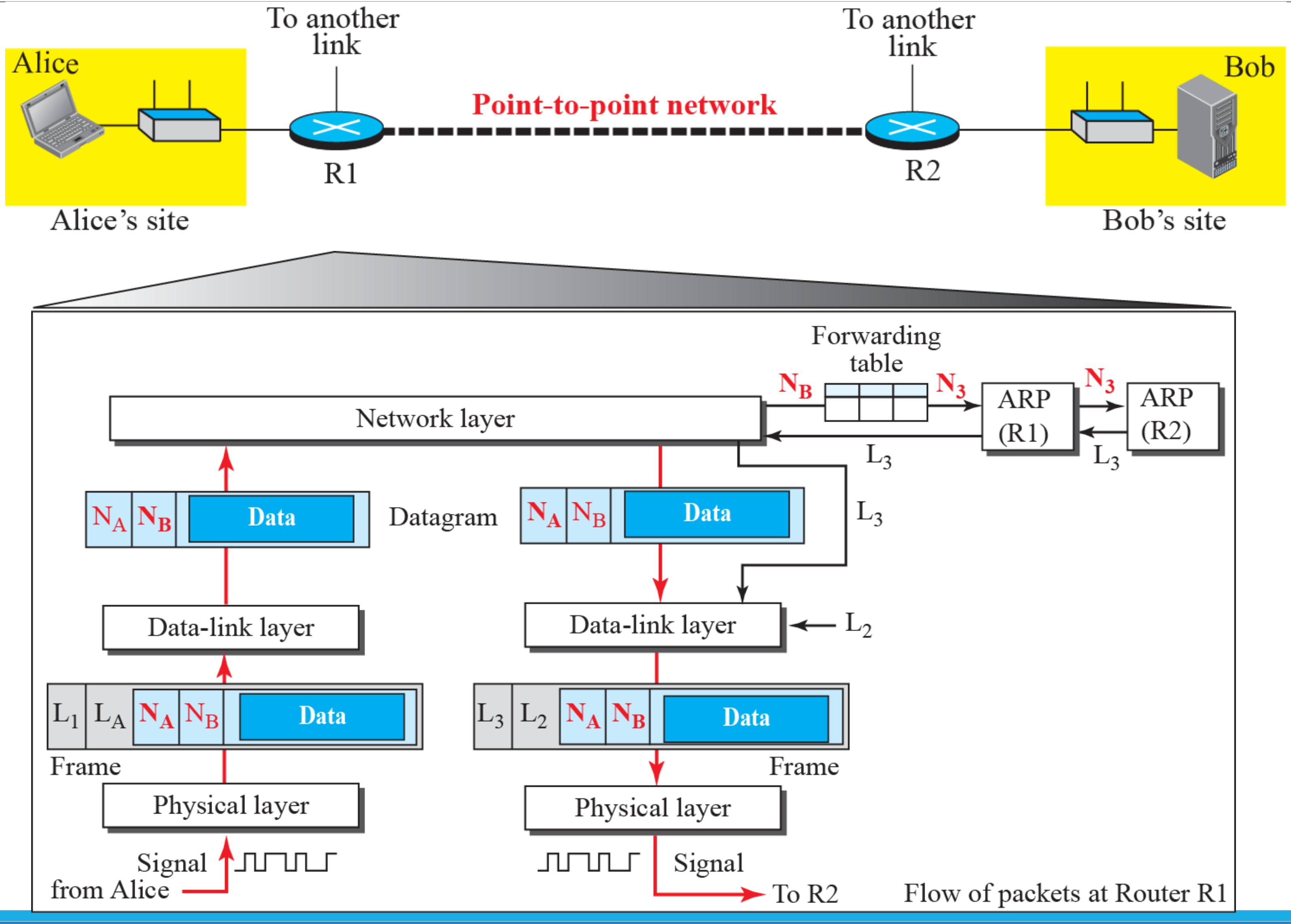
ARP Example


To another link
To another link
Alice
Bob
Point-to-point network
R1
R2
Alice's site
Bob's site

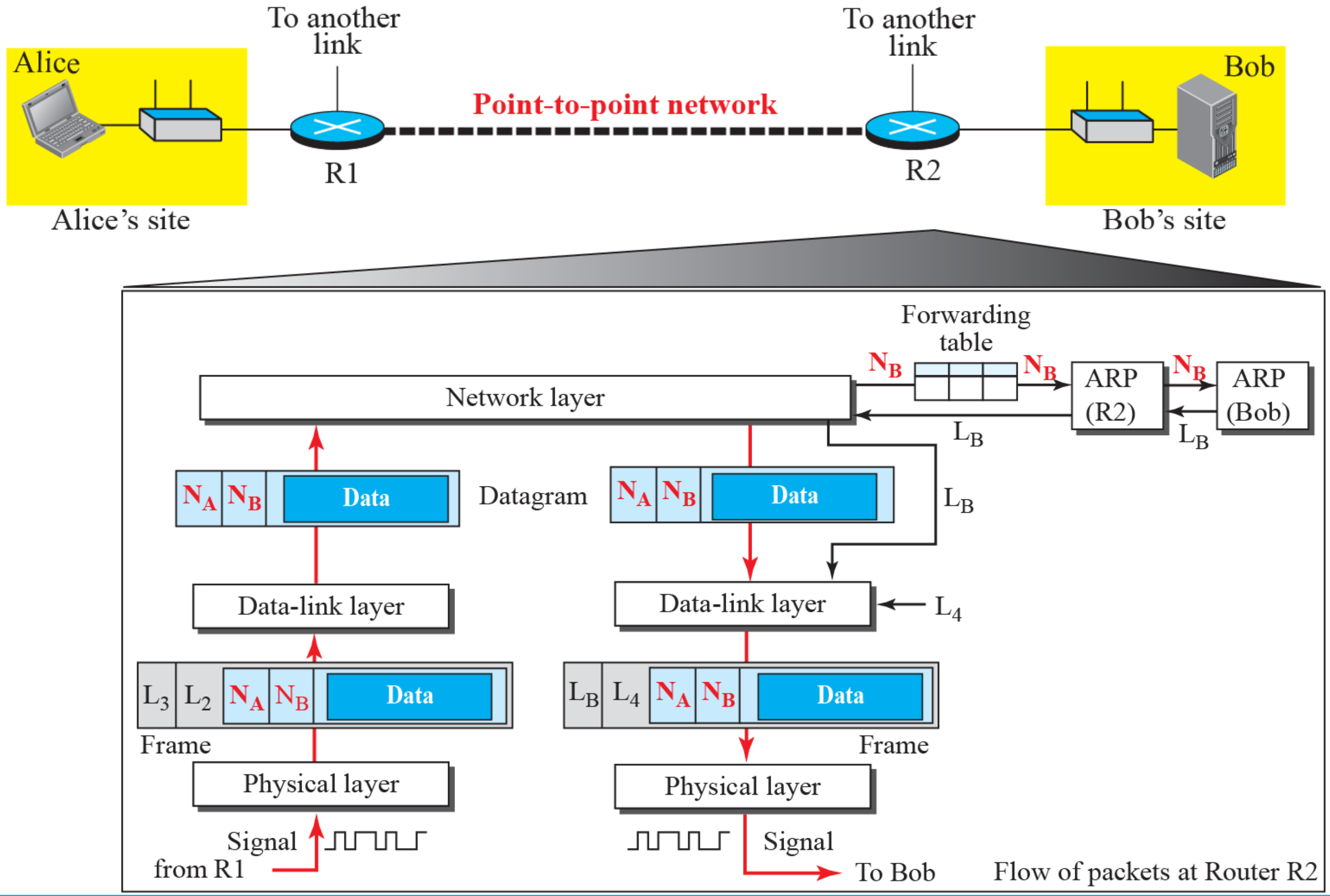
Flow of packets at Alice site



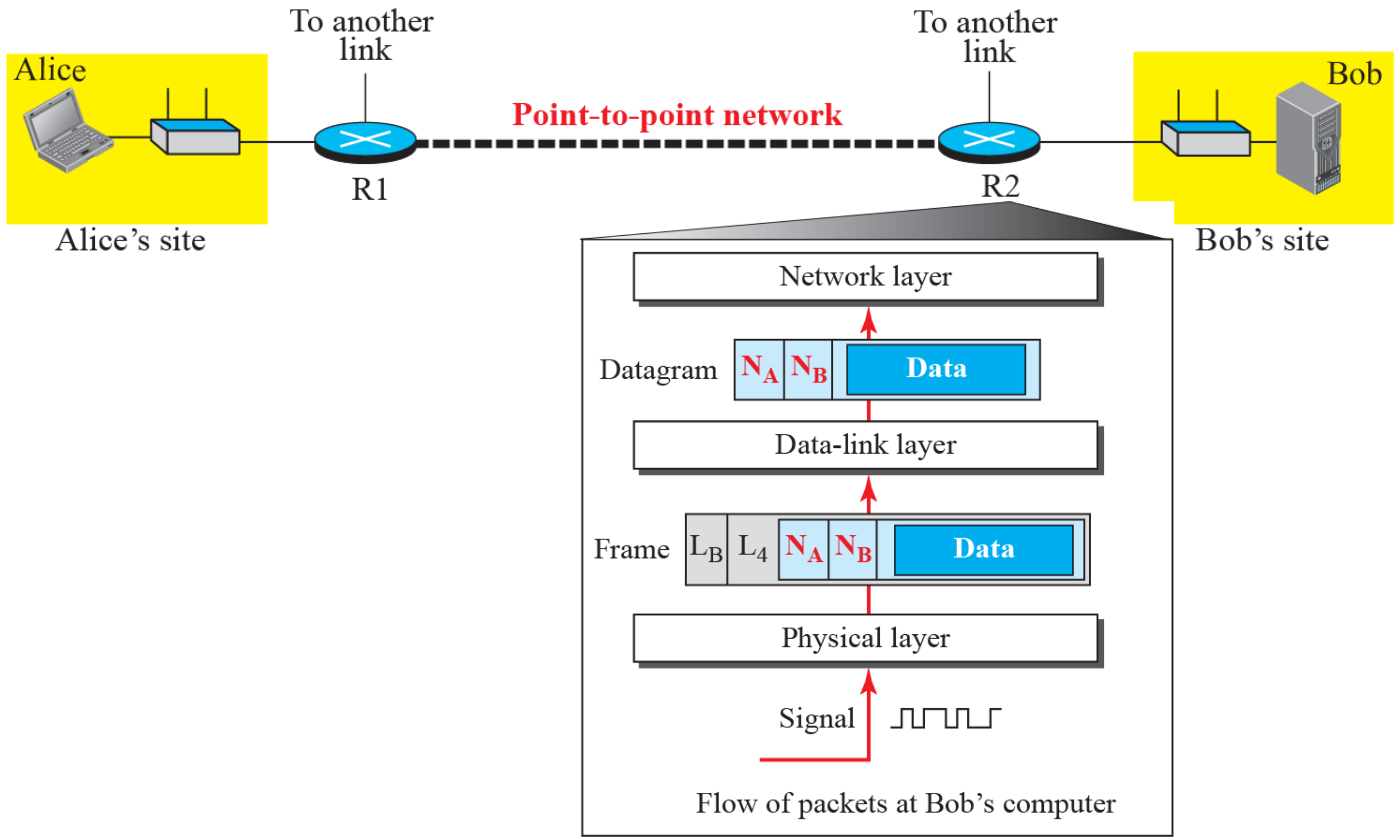
Flow of activities at Router R1



Flow of activities at router R2



Activities at Bob's site



Some Questions

- If the IP address of a router does not appear in any datagram sent from a source to a destination, why do we need to assign IP addresses to routers?
- Why do we need more than one IP address in a router, one for each interface?
- How are the source and destination IP addresses in a packet determined?
- How are the source and destination link-layer addresses determined for each link?

Data Link Layer

Data Link Control

Outline

Data Link Control

DLC Services

- Framing
  - Character Oriented Framing
  - Bit Oriented Framing
- Error & Flow Control

Data Link Control (DLC) : Services

Data link control (DLC)

- Deals with procedures for communication between **two adjacent nodes** – node-to-node communication
- No matter whether the link is **dedicated** **or** **broadcast**.

Data link control functions:

Framing and

Flow and Error Control.

DLC Services:
Framing

Data-link layer:

- Needs to **pack bits** into **frames**,
- So that **each frame is distinguishable** from another.

Example:

- Our **postal system** practices a type of framing.
- Act of **inserting a letter into an envelope** separates one piece of information from another; **the envelope serves as the delimiter**.
- **Each envelope** defines the sender and receiver addresses,

- ***Framing* in the data-link layer**
  - Separates a message from one source to a destination by adding a sender address and a destination address.

DLC Services:
Frame Size

- **Frames can be of fixed or variable size.**
- In *fixed-size framing:*
  - There is no need for defining the boundaries of the frames;
  - The size itself can be used as a delimiter.
  - **Example:** ATM WAN, *uses frames of fixed size called cells*.

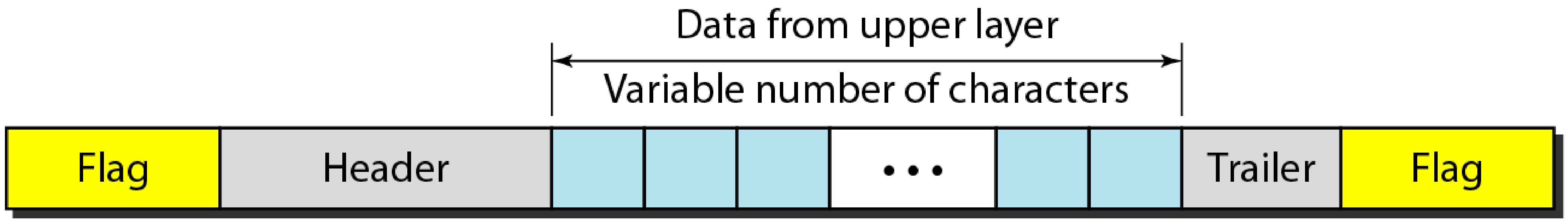
- Main Concern *variable-size framing,*
  - Prevalent in LANs.
  - Need a way to define the end of one frame and the beginning of the next.
  - Two approaches were used for this purpose:
    - A character-oriented approach and
    - A bit-oriented approach.

DLC Services:

Character Oriented Framing

In character-oriented (or byte-oriented) framing,

- Data to be carried are 8-bit characters (ASCII).
- Header: Carries the **source and destination addresses** and other **control information.**
- Trailer: Carries **error detection redundant bits** (multiple of 8 bits)
- Frames are Separated:
  - An **8-bit (1-byte) flag** is added at the **beginning & the end** of a frame.
  - The flag, composed of protocol-dependent special characters, signals the start or end of a frame.

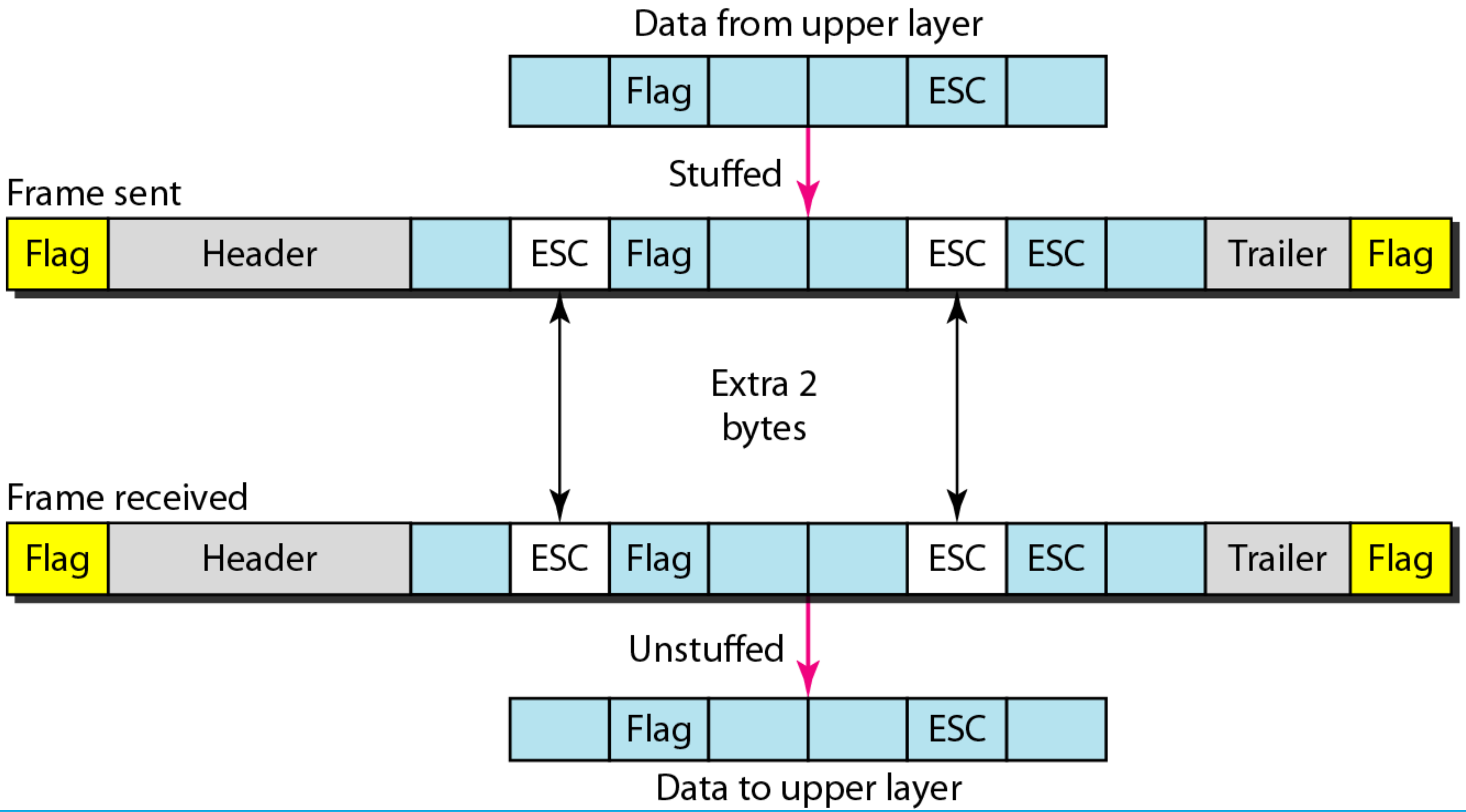


DLC Services:

Character Oriented Framing

Byte Stuffing and Unstuffing

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



DLC Services:

Character Oriented Framing

Byte Stuffing and Unstuffing

- Character-oriented framing was popular when only text was exchanged by the data-link layers.
  - The *flag* could be selected to be any character not used for text communication.
- However, *other types of information such as images, audio, and video;*
  - Any ***character*** used for the ***flag*** could also be ***part of the information***.
  - If this happens, the receiver, when it ***encounters this pattern*** in the ***middle of the data***, thinks it has ***reached the end of the frame***.
- **To fix this problem:** A byte-stuffing strategy was added to character-oriented framing.
- **Byte stuffing** (or character stuffing),
  - A ***special byte*** is added to the ***data section of the frame when there is a character with the same pattern as the flag***.
  - The data section is stuffed with an extra byte.
  - This byte is usually called the *escape character (ESC)* and has a predefined bit pattern.

DLC Services:

Character Oriented Framing

Byte Stuffing and Unstuffing

- Whenever the receiver encounters the ESC character,
  - It removes it from the data section and
  - Treats the next character as data, not as a delimiting flag.

- Byte stuffing by the escape character
  - Allows the presence of the flag in the data section of the frame, but it creates another problem.
  - *What happens if the text contains one or more escape characters followed by a byte with the same pattern as the flag?*
    - The *receiver removes* the *escape character*, but *keeps the next byte*, which is *incorrectly interpreted* as the *end* of the *frame*.

DLC Services:

Character Oriented Framing

Byte Stuffing and Unstuffing

- To solve this problem, the ***escape characters*** that are ***part*** of the ***text*** must also be ***marked*** by ***another escape character***.

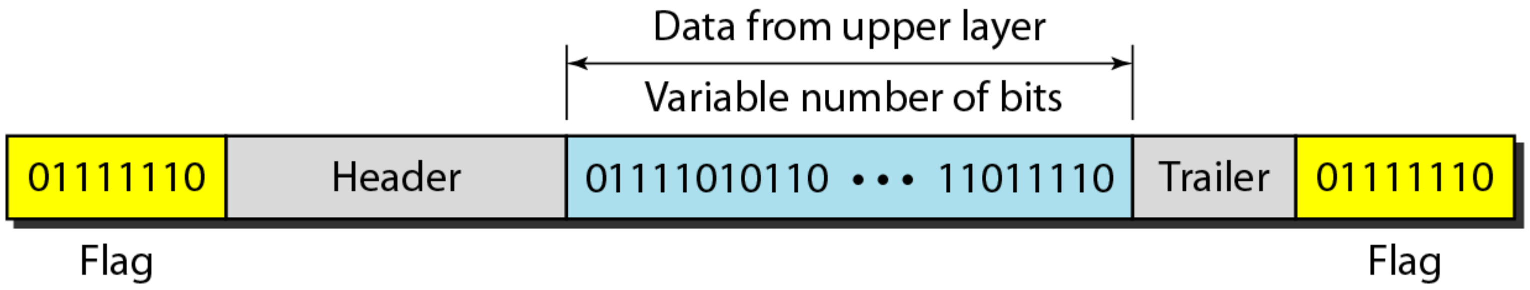
- In other words, if the **escape character is part of the text**, an **extra one** is added to **show** that the **second one is part of the text**.

- **Character-oriented protocols** present **another problem in data communications.**

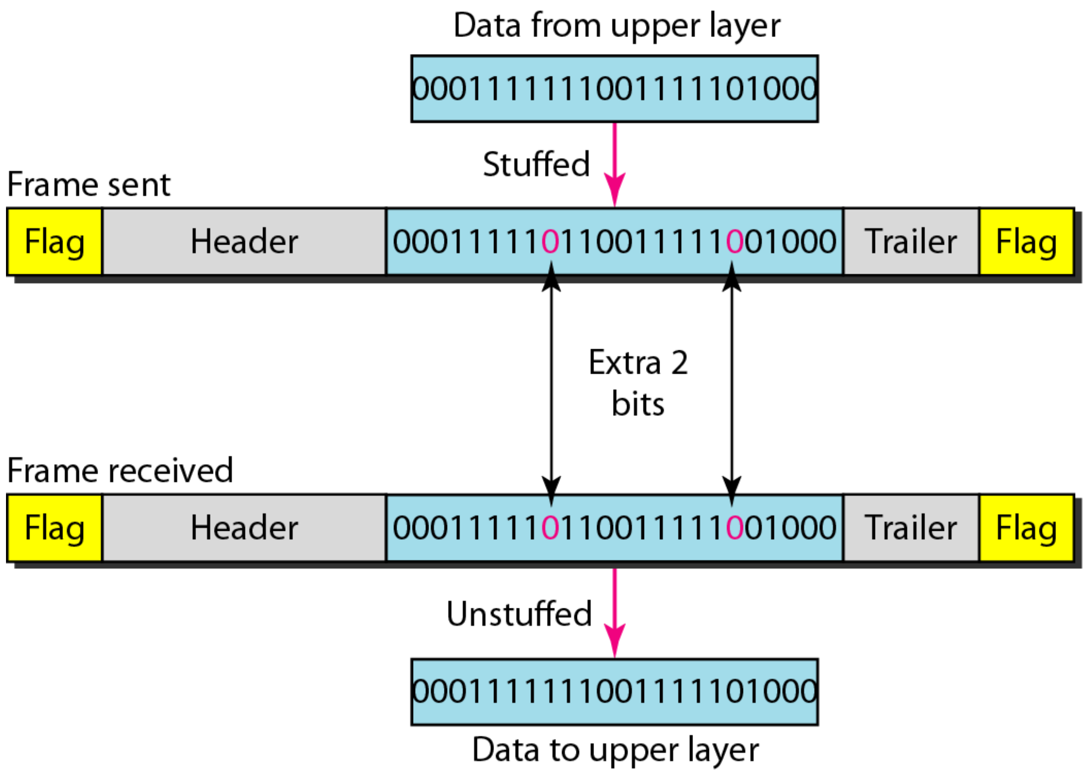
- The universal coding systems in use today, such as **Unicode**, have 16-bit and 32-bit characters that conflict with 8-bit characters.

DLC Services:

Bit Oriented Framing



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.



DLC Services:
Bit Oriented Framing

- This **flag can create the same type of problem** we saw in the **character-oriented protocols**.

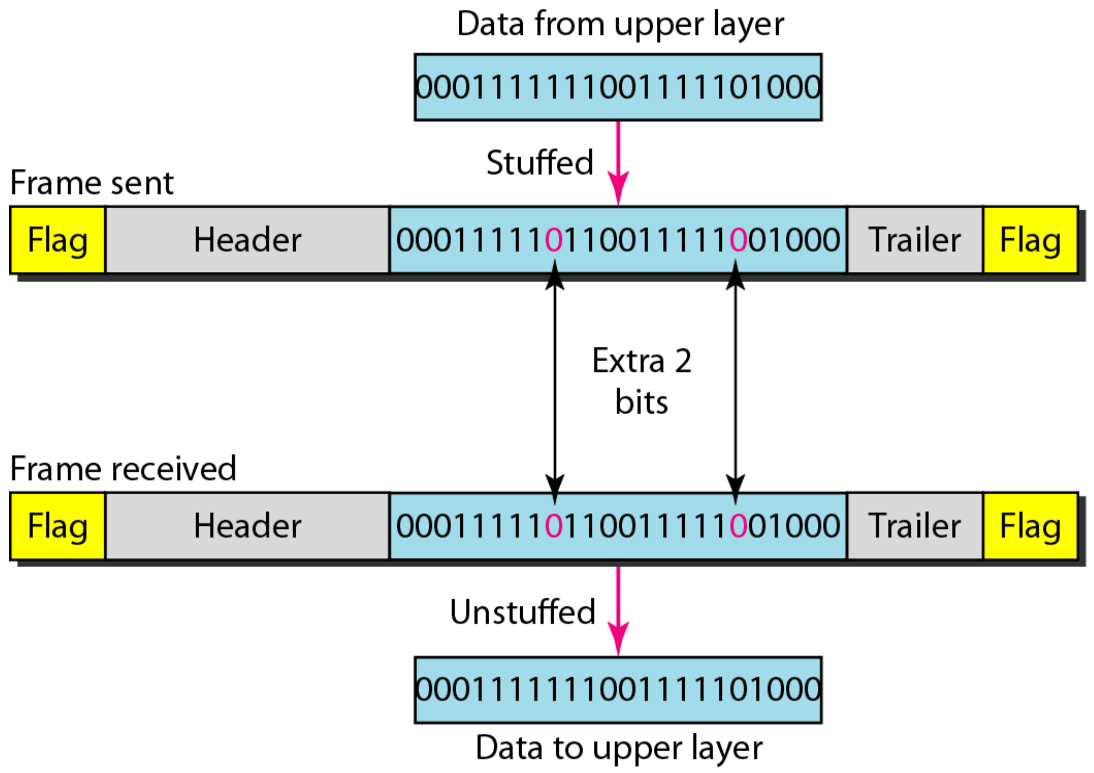
- That is, **if the flag pattern appears in the data**, we need to *somehow inform* the receiver that this is *not the end of the frame*.

- We do this by **stuffing 1 single bit** (instead of 1 byte) to prevent the pattern from looking like a flag.

The strategy is called **bit stuffing.**

- In bit stuffing, if a 0 and five consecutive 1 bits are encountered, an **extra 0 is added**.
- This extra stuffed bit is eventually removed from the data by the receiver.

Note that the extra bit is added after one 0 followed by five 1s regardless of the value of the next bit.



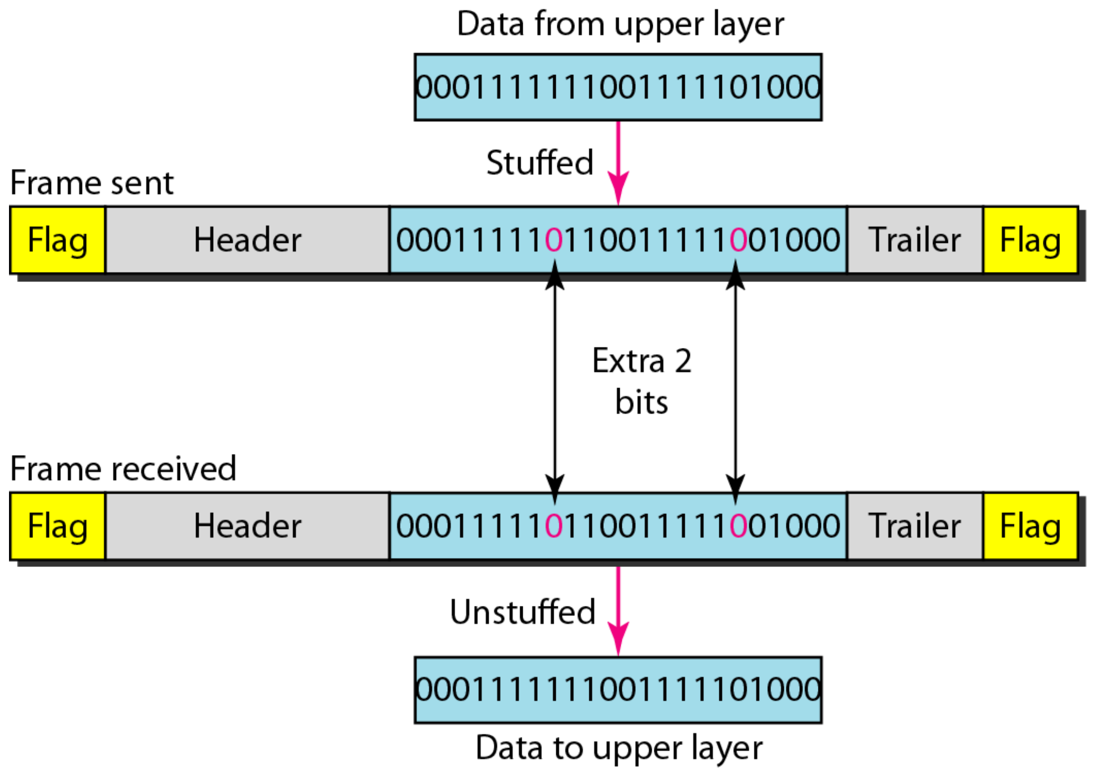
DLC Services:

Bit Oriented Framing

This guarantees that the flag field sequence does not inadvertently appear in the frame.

Figure shows bit stuffing at the sender and bit removal at the receiver.

- Note that even if we have a 0 after five 1s, we still stuff a 0.
- The 0 will be removed by the receiver.
- This means that if the flaglike pattern 01111110 appears in the data,
- it will change to 011111010 (stuffed) and is not mistaken for a flag by the receiver.
- The real flag 01111110 is not stuffed by the sender and is recognized by the receiver.



DLC Services:
Flow & Error Control

Most important responsibilities of the data link layer:

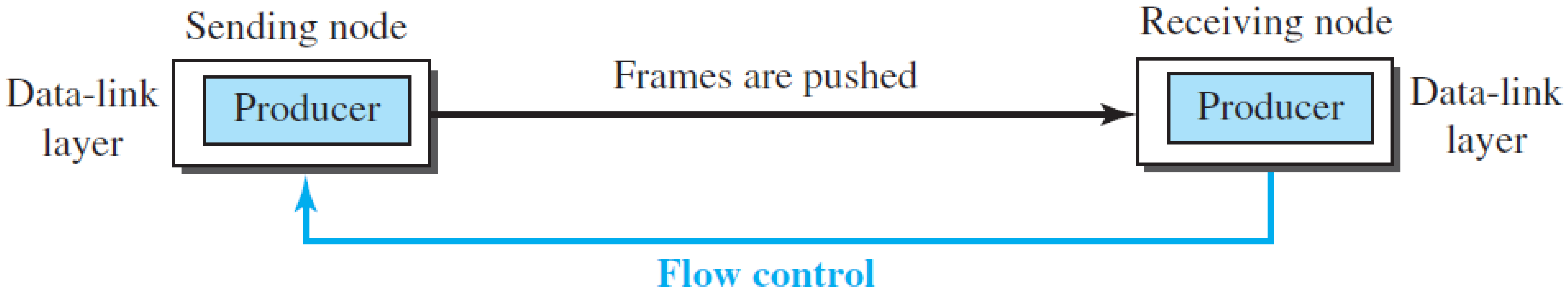
- Flow control and Error control.
- Collectively, these functions are known as ***data link control (DLC).***

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:

- Based on automatic repeat request, which is the retransmission of data.



DLC Services:
Flow & Error Control

Error Control

- Need to **implement error control at the data-link layer**
  - Prevent the receiving node from delivering corrupted packets to its network layer.

Can be implemented using one of the following two methods.

- In both methods, a CRC is added to the frame header by the sender & checked by the receiver.
- First method,
  - If the frame is corrupted, it is silently discarded;
  - If it is not corrupted, the packet is delivered to the network layer.
  - Used mostly in wired LANs such as Ethernet.
- Second method,
  - If the frame is corrupted, it is silently discarded;
  - If it is not corrupted, an acknowledgment is sent to the sender

Combination of Flow and Error Control

- The acknowledgment that is sent for flow control
  - *Can also be used for error control to tell the sender the packet has arrived uncorrupted.*
- Lack of acknowledgment: *Means that there is a problem in the sent frame*

Data Link Layer

Data Link Layer Protocols

Outline

Simple Protocol

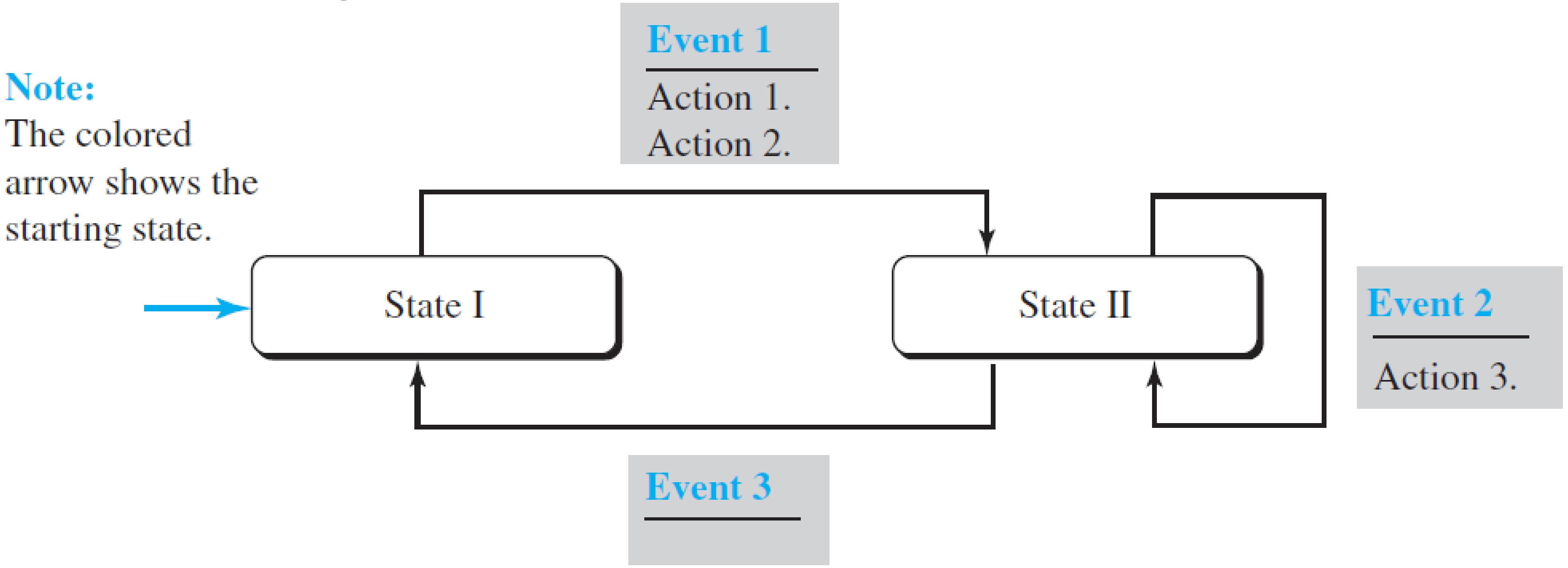
Stop & Wait Protocol

HDLC

Data Link Control Protocols

Four Protocols:

- Simple
- Stop-and-Wait
- Go-Back-N
- Selective Repeat

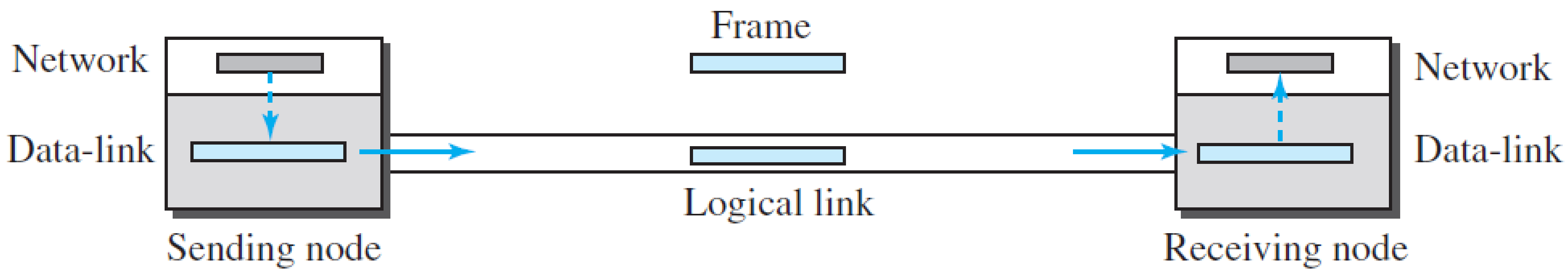


Data Link Control Protocols

Simple Protocol

- Offers **neither flow nor error control**.
- Assume that the receiver can immediately handle any frame it receives.
  - The ***receiver can never be overwhelmed with incoming frames***.
- **Procedure:**
  - DL layer at the **Tx** gets a packet from its network layer, makes a frame, & sends the frame.

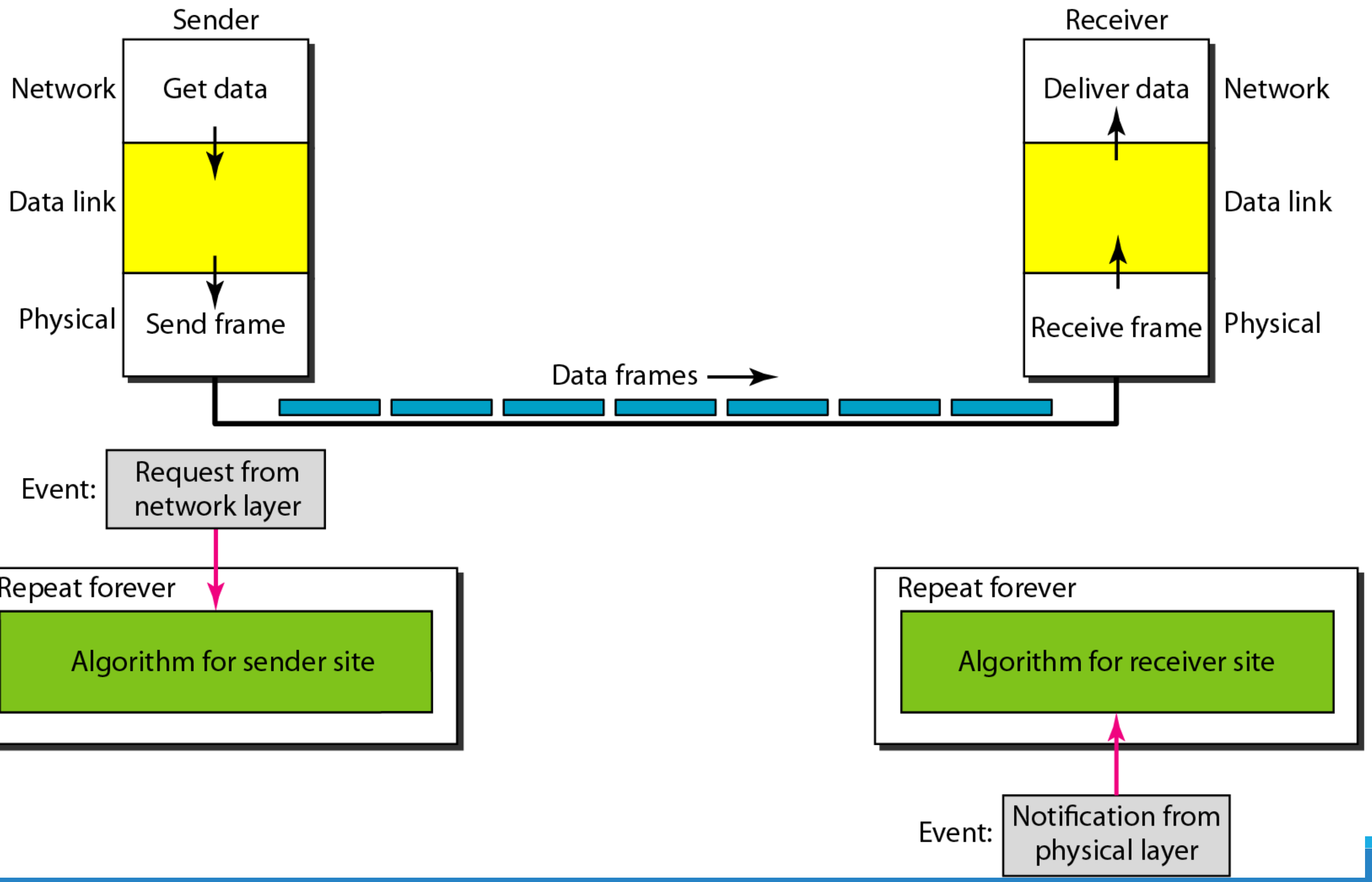
  DFL layer at the **Rx** receives a frame, extracts the packet from the frame, and delivers the packet to its network layer.



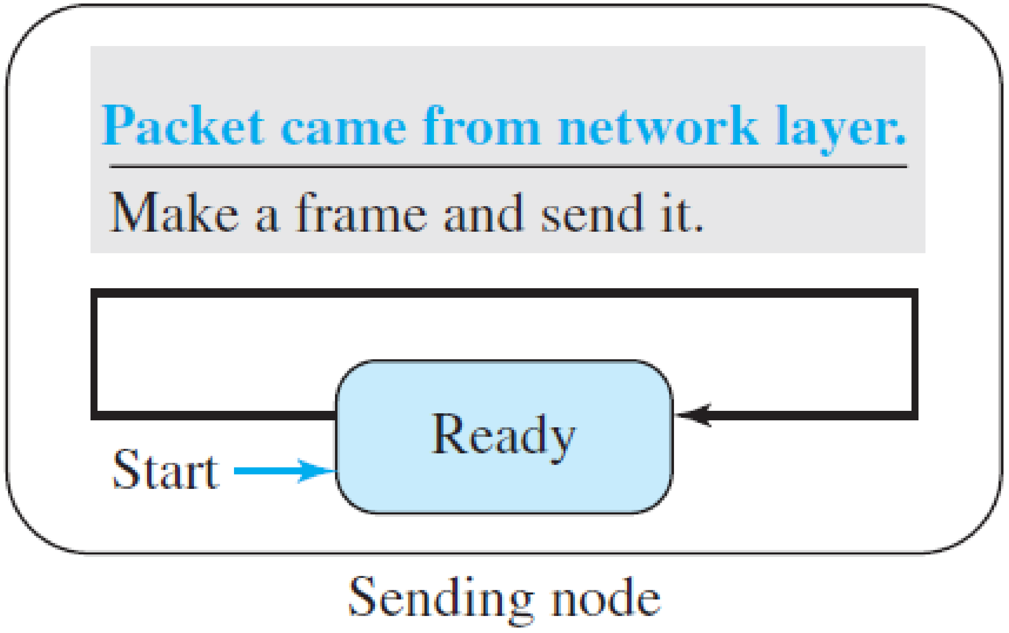
Data Link Control Protocols

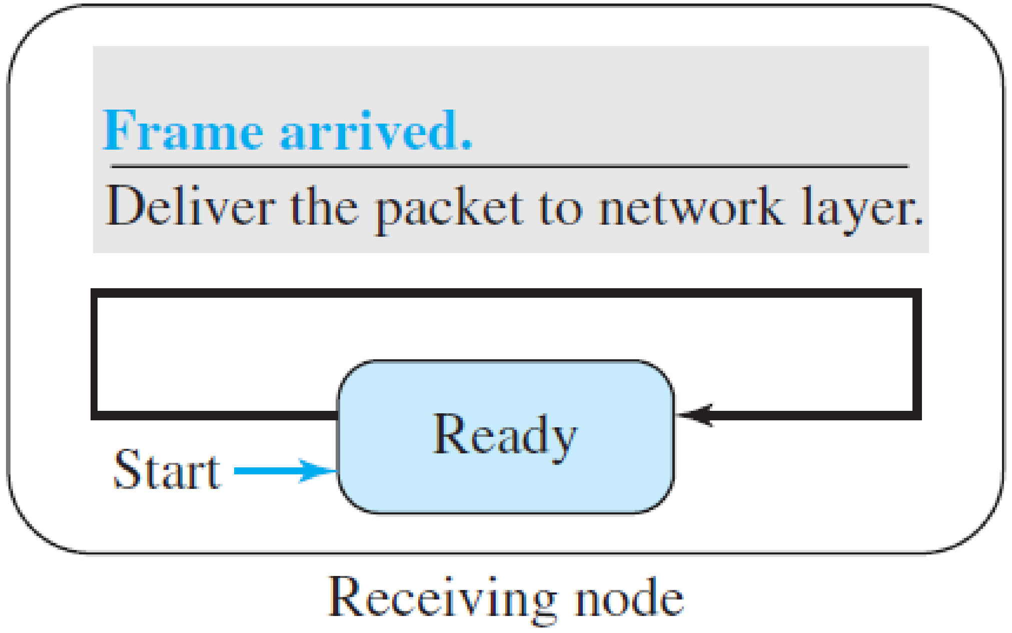
Simple Protocol

- Offers **neither flow nor error control**.



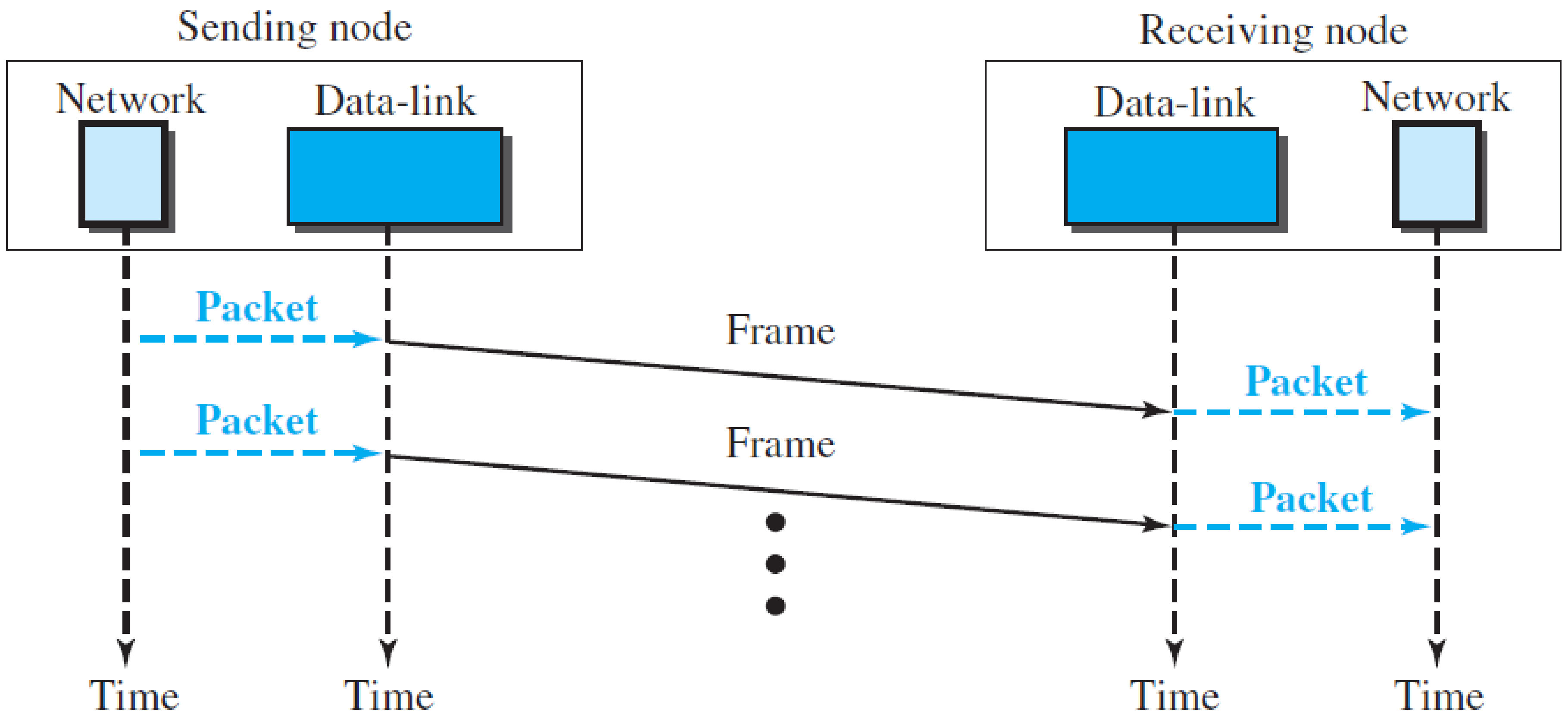
Simple Protocol : FSM





Data Link Control Protocols

Simple Protocol : FSM



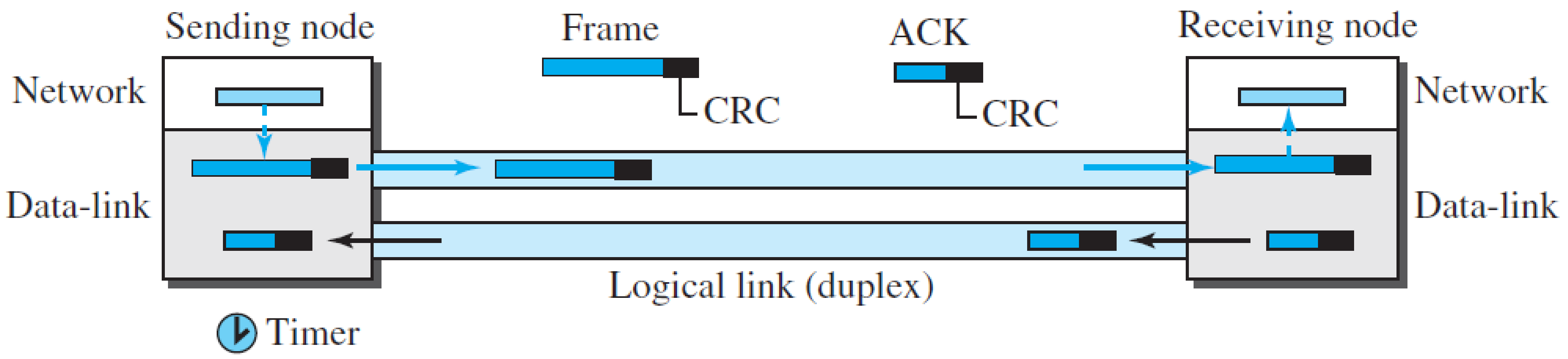
Data Link Control Protocols

Stop & Wait Protocol

- **Uses both flow and error control.**

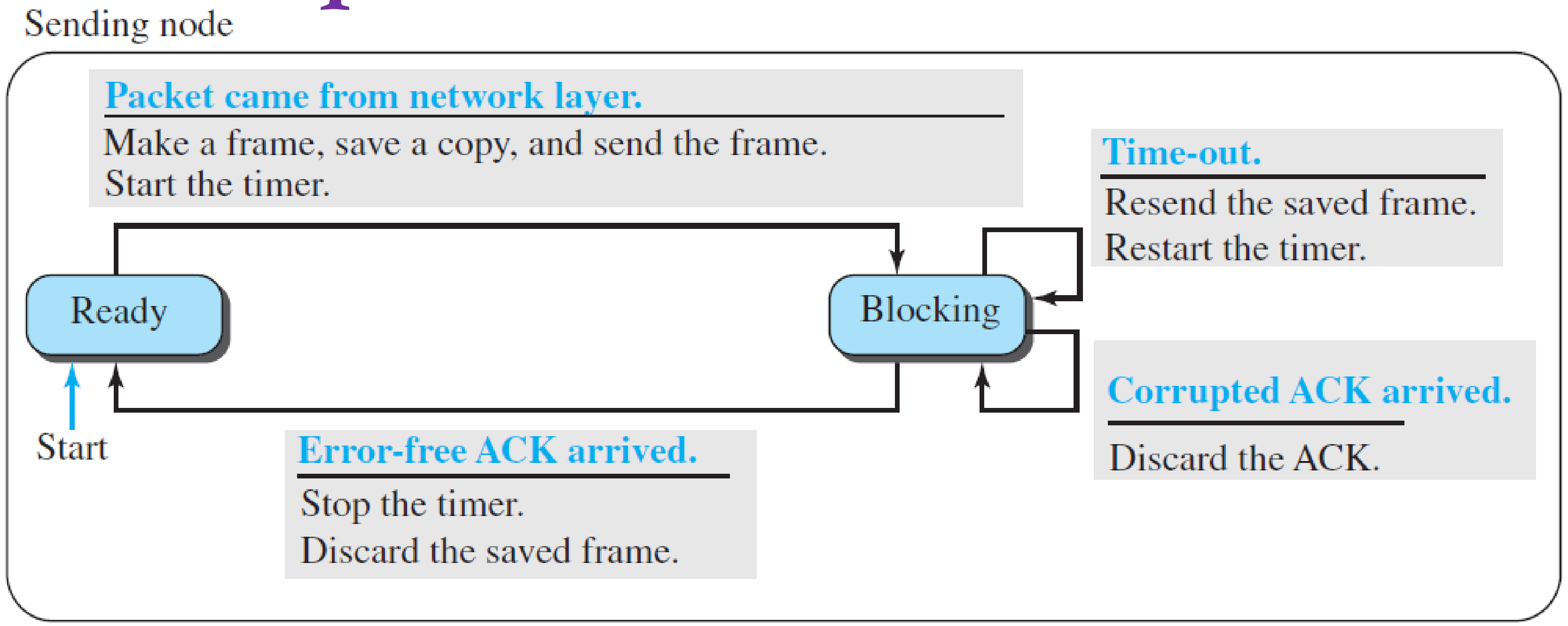
Procedure:

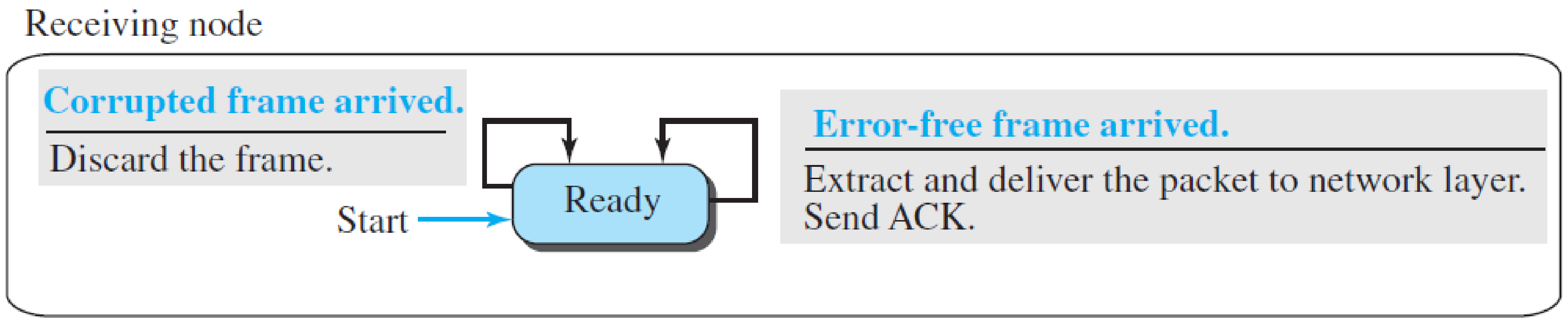
- Sender sends one frame at a time and waits for an ACK before sending the next.
- **CRC** is added to each data frame; **to detect Errors**.
- When a frame arrives at the receiver site,
  - It is checked.
  - If its CRC is incorrect, the frame is corrupted and silently discarded.
  - The silence of the receiver is a signal for the sender that a frame was either corrupted or lost.



Data Link Control Protocols

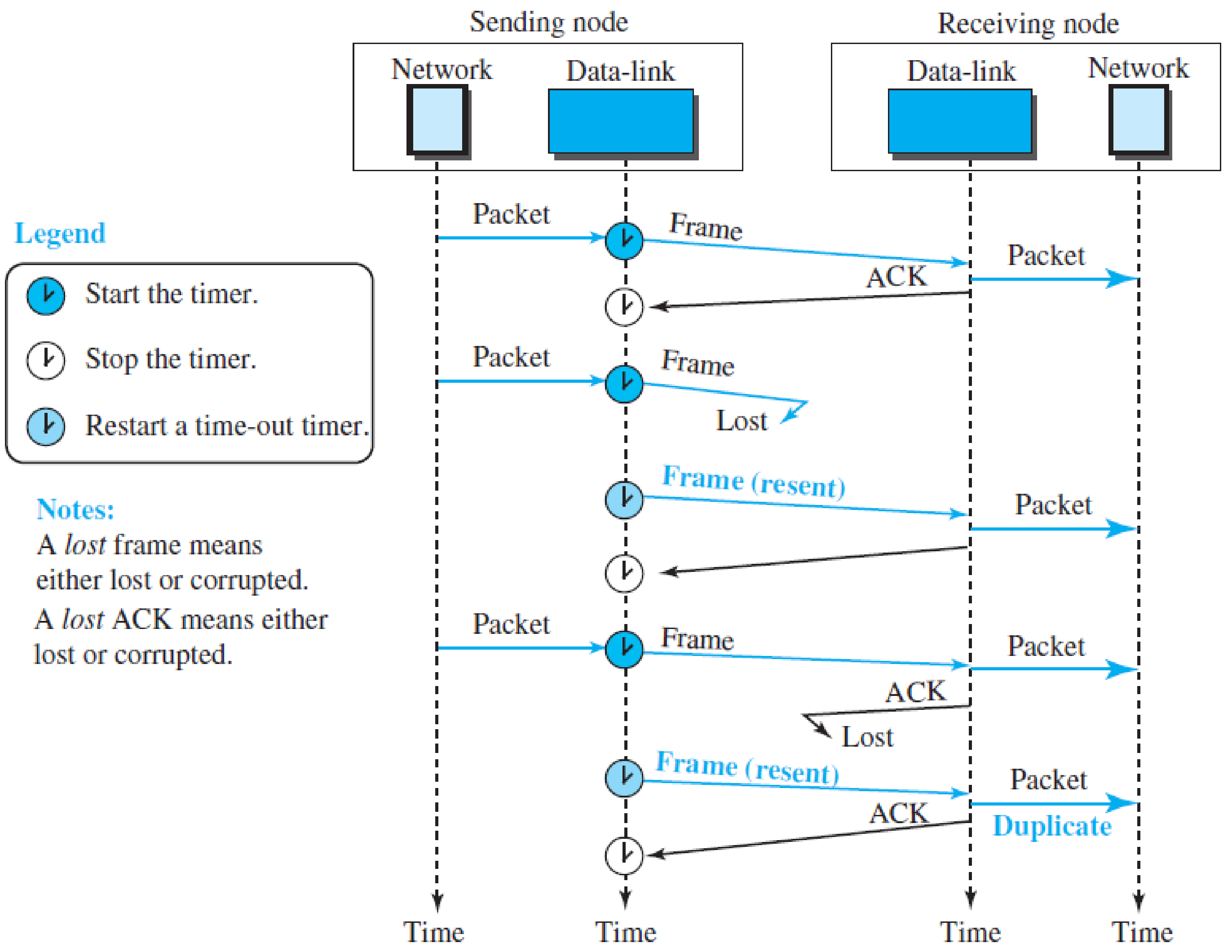
Stop & Wait Protocol : FSM





Data Link Control Protocols

Stop & Wait Protocol



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.

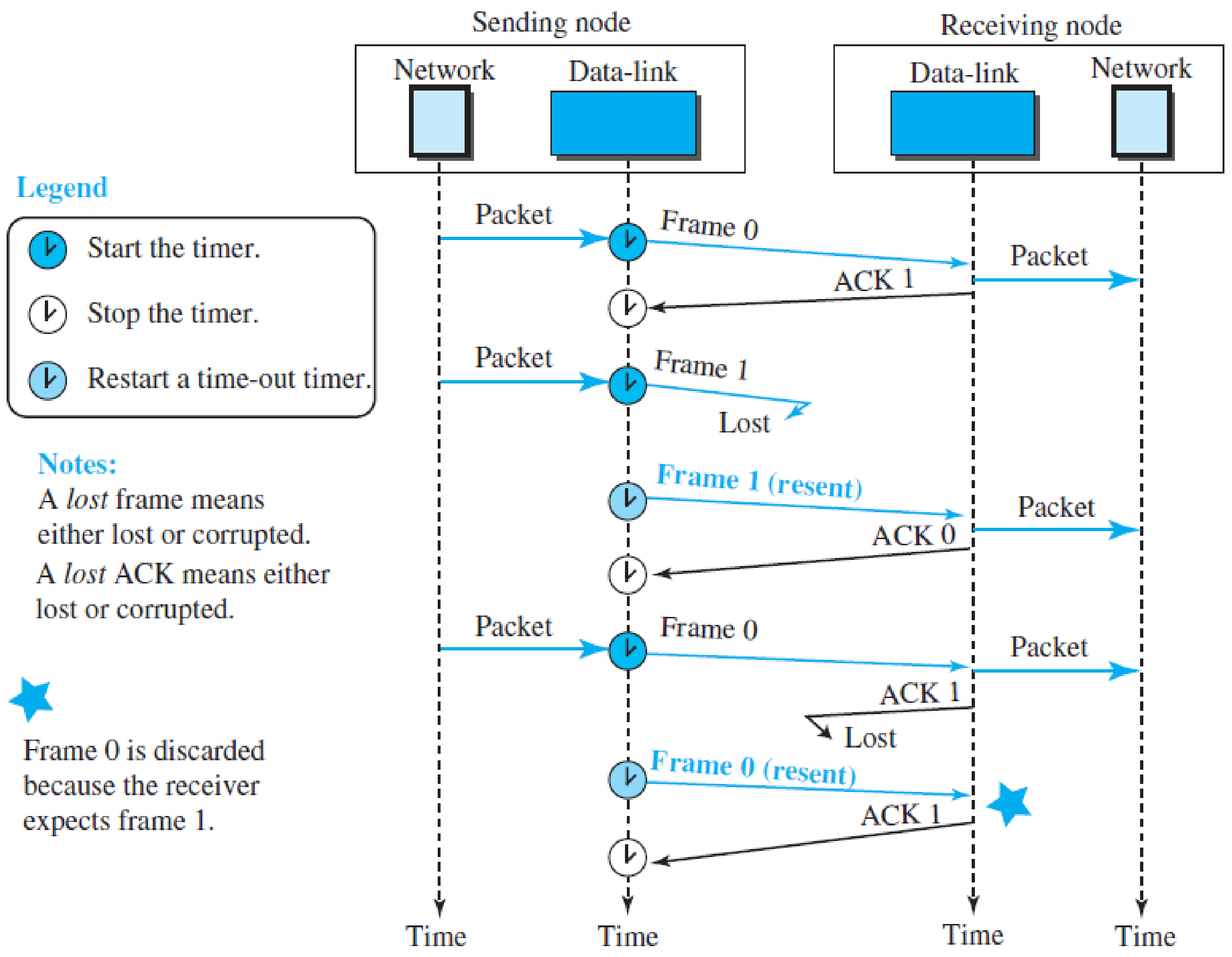
Stop & Wait Protocol

We need to add **sequence numbers** to the data frames and **acknowledgment numbers** to the ACK frames.

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

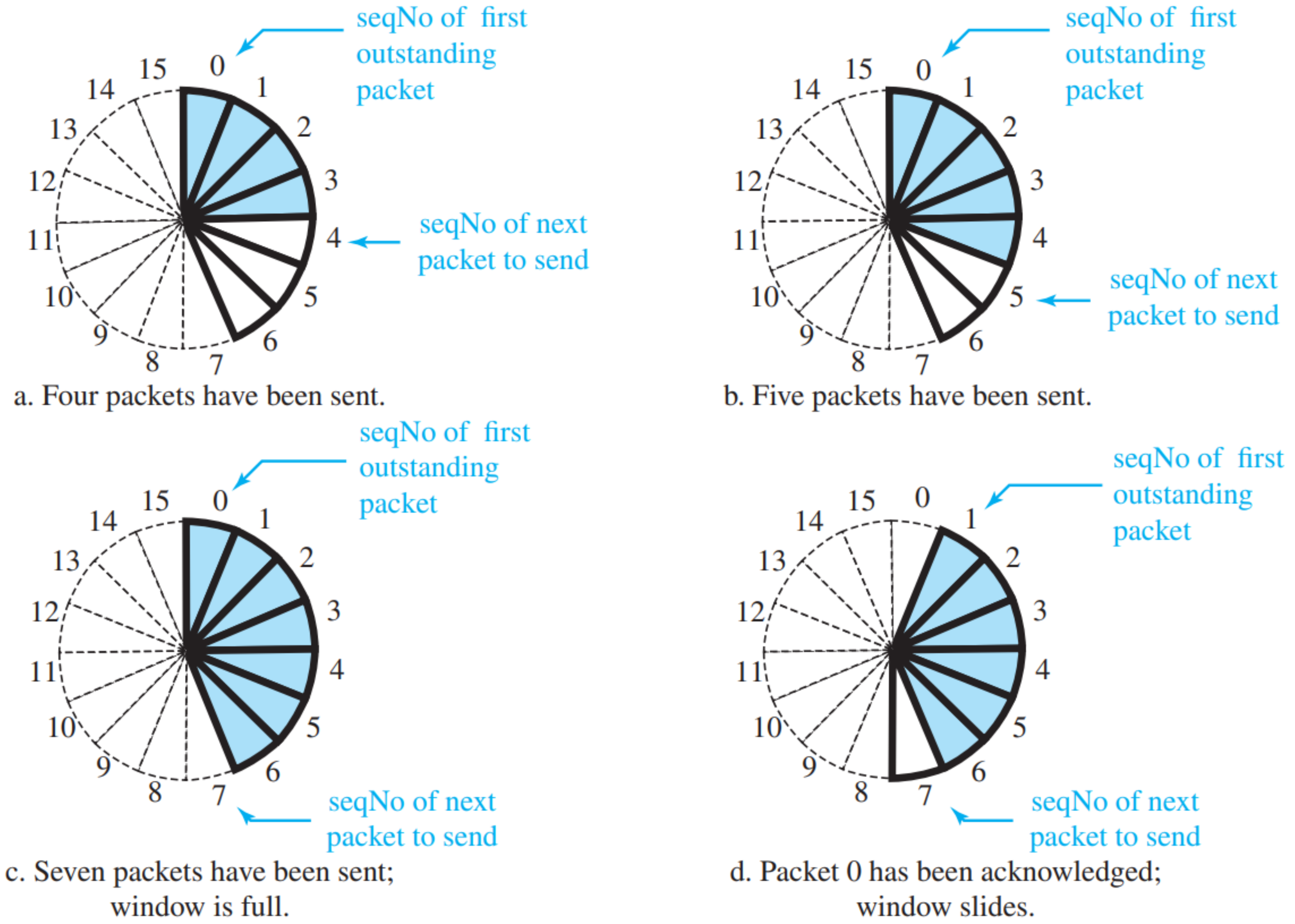
The sequence numbers start with 0, the acknowledgment numbers start with 1.

An acknowledgment number always defines the sequence number of the next frame to receive.



Go-back-N Protocol

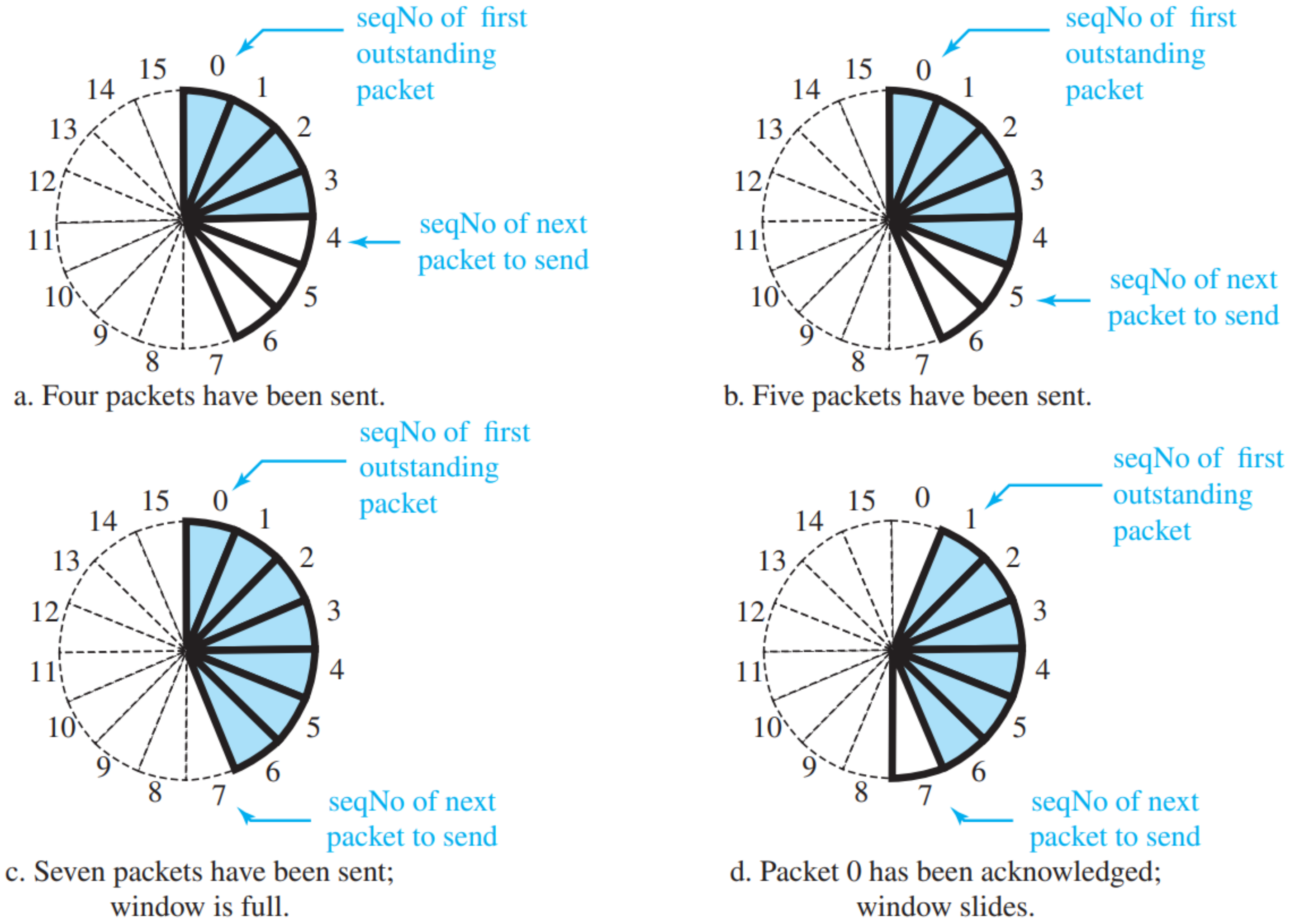
- Since the **sequence numbers use modulo 2^m**, a circle can represent the sequence numbers from 0 to $2^m - 1$ (Figure).
- **Buffer** is represented as **a set of slices**, called the sliding window, that occupies part of the circle at any time.
- ***At the sender site:***
  - When a packet is sent, the **corresponding slice is marked.**
  - **When all the slices are marked**, it means that the **buffer is full** and
  - No further messages can be accepted from the application layer.



seqNo of first outstanding packet
seqNo of next packet to send
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
a. Four packets have been sent.
b. Five packets have been sent.
c. Seven packets have been sent; window is full.
d. Packet 0 has been acknowledged; window slides.

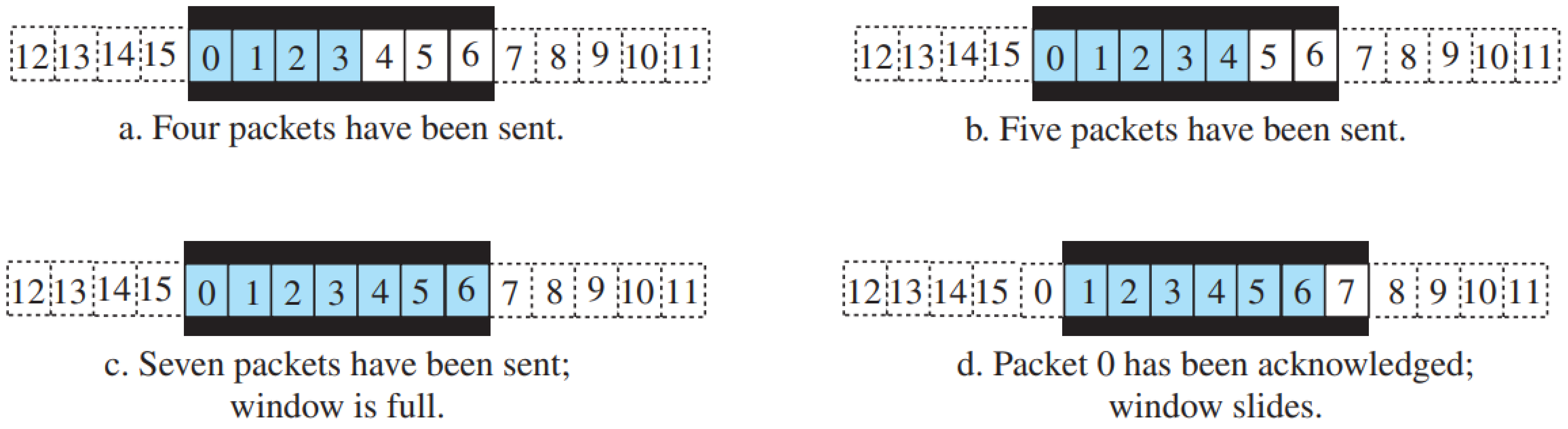
Go-back-N Protocol

- When an **acknowledgment arrives**, the **corresponding slice** is **unmarked**.
- *If some consecutive slices from the beginning of the window are unmarked,*
  - The window slides over the range of the corresponding sequence numbers to allow more free slices at the end of the window.
- Figure shows the sliding window at the sender. **The sequence numbers are in modulo 16 (m = 4) and the size of the window is 7.**
- **Note that the sliding window is just an abstraction**: the actual situation uses computer variables to hold the sequence numbers of the next packet to be sent and the last packet sent.


seqNo of first outstanding packet
seqNo of next packet to send
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
a. Four packets have been sent.
b. Five packets have been sent.
c. Seven packets have been sent; window is full.
d. Packet 0 has been acknowledged; window slides.

Go-back-N Protocol

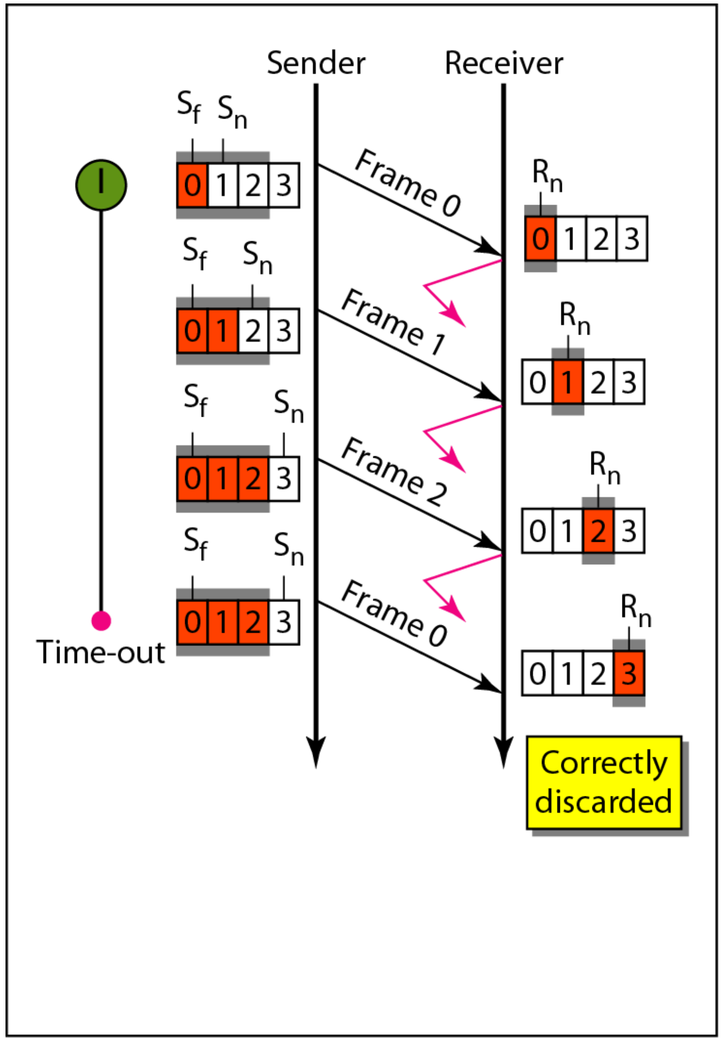
- Most protocols show the sliding window using linear representation.
- The idea is the same, but it normally takes less space on paper.
- Figure shows this representation. Both representations tell us the same thing.



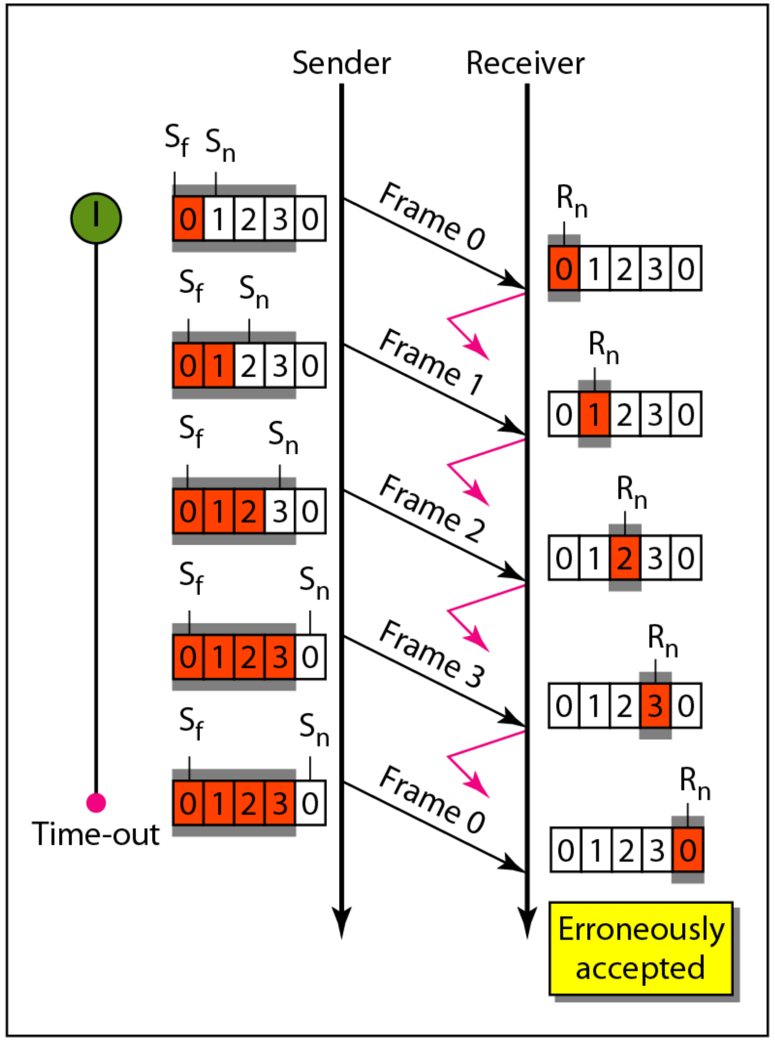
12 13 14 15 0 1 2 3 4 5 6 7 8 9 10 11
a. Four packets have been sent.
12 13 14 15 0 1 2 3 4 5 6 7 8 9 10 11
b. Five packets have been sent.
12 13 14 15 0 1 2 3 4 5 6 7 8 9 10 11
c. Seven packets have been sent; window is full.
12 13 14 15 0 1 2 3 4 5 6 7 8 9 10 11
d. Packet 0 has been acknowledged; window slides.

Sliding window in linear format

Window size for Go-Back-N ARQ



a. Window size < 2^m

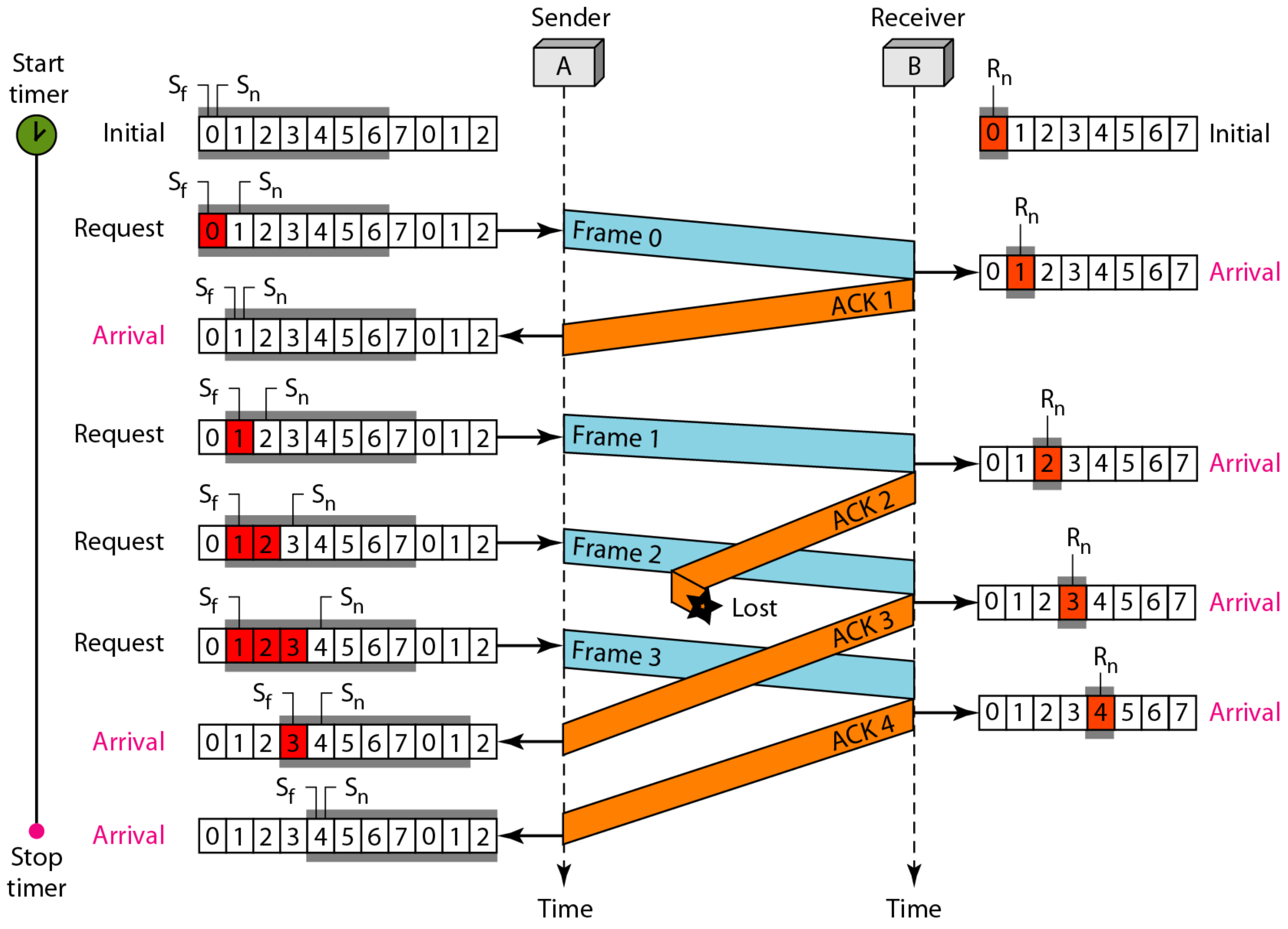


b. Window size = 2^m

- We **choose m = 2**, which *means the size of the window can be 2^m - 1, or 3*.
- Why the size of the **send window must be less than 2^m**.
- If the **size of the window is 3** (less than 2^2) **and all three acknowledgments are lost**, the **frame 0 timer expires and all three frames are resent.**
- The receiver is now expecting frame 3, not frame 0, so the duplicate frame is correctly discarded.
- **On the other hand, if the size of the window is 4 (equal to 2^2)** and all acknowledgments are lost, ***the sender will send a duplicate of frame O.***
- However, this time the window of the receiver expects to receive frame 0, so it accepts frame 0, not as a duplicate, but as the first frame in the next cycle. **This is an error.**

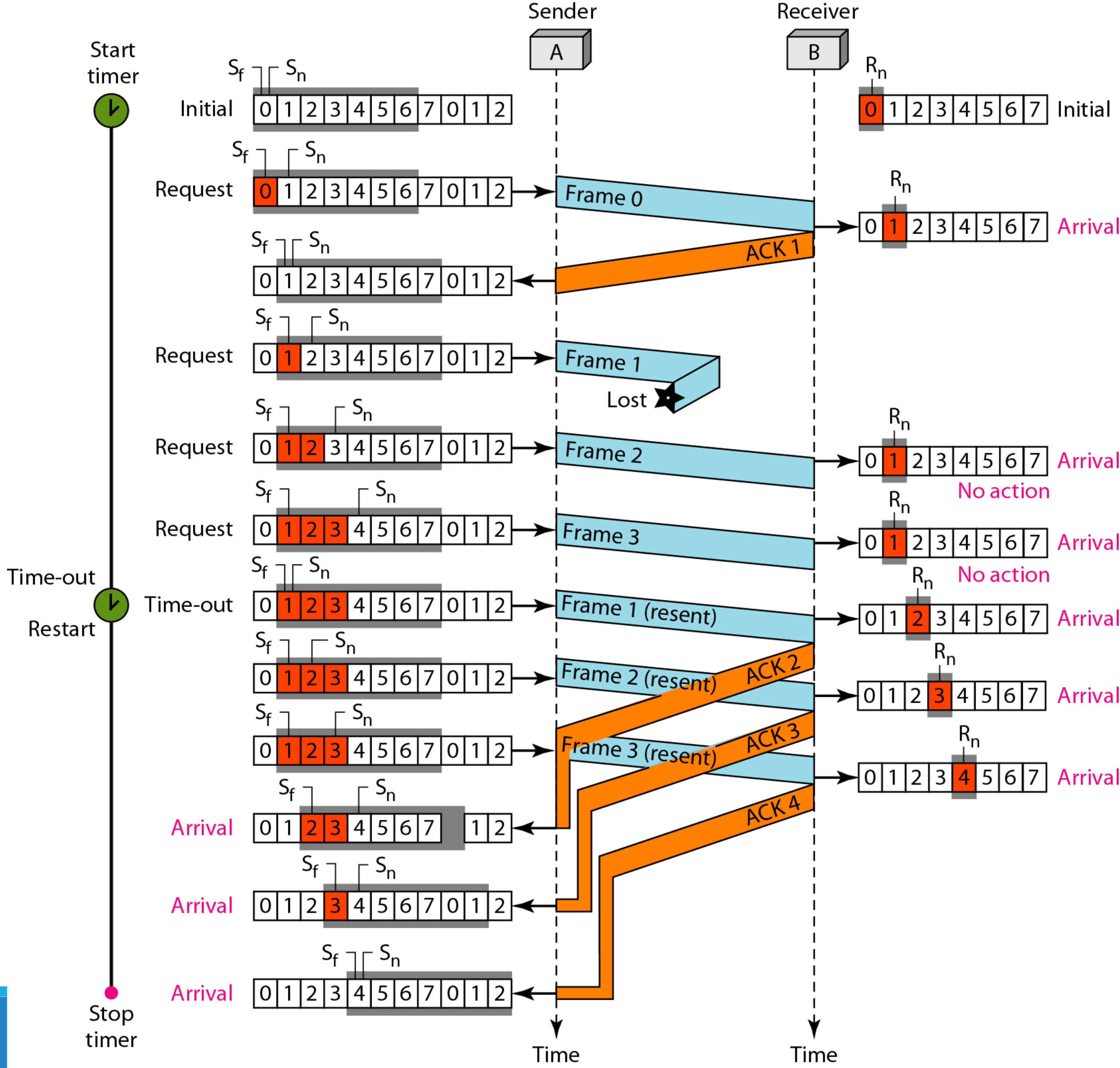
Example 1: Go-Back-N ARQ

- This is an example of a **case where the forward channel is reliable, but the reverse is not.**
- No data frames are lost, but some ACKs are delayed and one is lost.
- The example also shows *how cumulative acknowledgments can help if acknowledgments are delayed or lost*.
- After initialization, there are seven sender events.
- ***Request events*** *are triggered by data from the* ***network layer****;*
- ***Arrival events*** *are triggered by* ***acknowledgments*** *from the* ***physical layer****.*
- *There is no time-out event* ***here because all outstanding frames are acknowledged before the timer expires****.*
- *Note that although ACK 2 is lost, ACK 3 serves as both ACK 2 and ACK 3*.



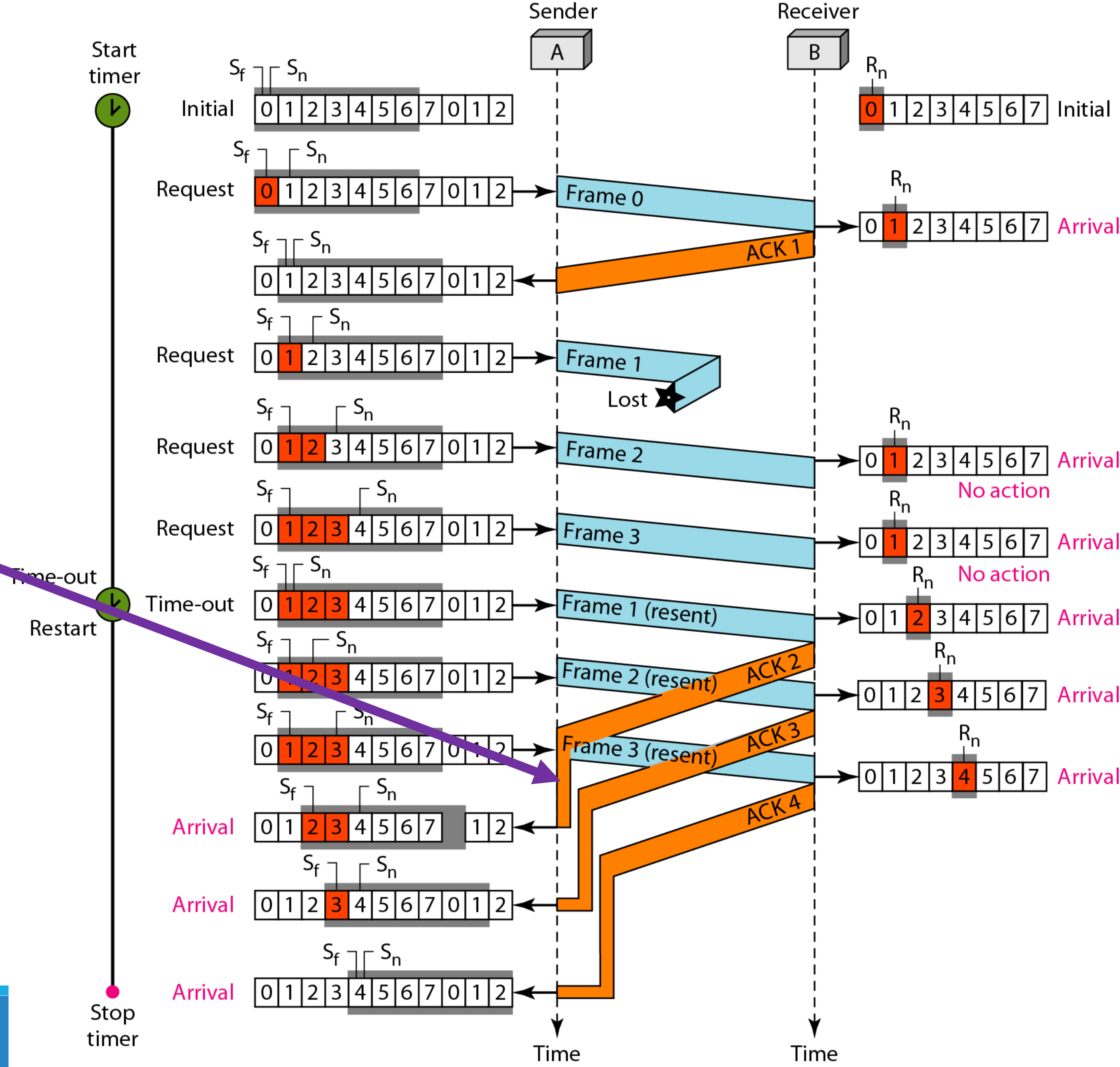
Example 2: Go-Back-N ARQ

- **Figure shows what happens when a frame is lost.**
- Frames 0, 1, 2, and 3 are sent.
- However, **frame 1 is lost**.
- The **receiver receives frames 2 and 3**, but **they are discarded** because they are received out of order.
- **Sender receives no acknowledgment about frames 1, 2, or 3.**
- Its **timer finally expires**.
- **Sender sends all outstanding frames** (1, 2, and 3) because it does not know what is wrong.
- Note that the resending of frames 1, 2, and 3 is the response to one single event.
- When **the sender is responding to this event**, ***it cannot accept the triggering of other events***.



Example 2: Go-Back-N ARQ

- When **the sender is responding to this event**, ***it cannot accept the triggering of other events***.
- This means that when ACK 2 arrives, the sender is still busy with sending frame 3.
- The **physical layer must wait until this event is completed** and the ***data link layer goes back to its sleeping state***.
- Shown a vertical line to indicate the delay.
- It is the same story with ACK 3; but when ACK 3 arrives, the sender is busy responding to ACK 2.
- It happens again when ACK 4 arrives.
- Note that *before the second timer expires, all outstanding frames have been sent and the timer is stopped*.



Reference

Forouzan, A. Behrouz. ***Data Communications & Networking***. 5th Edition. Tata McGraw-Hill Education.

Chapter 11 Data Link Control (DLC)

Topic: 11.1, 11.2, 11.3, Chapter 23 (page. 700 - 702)