Data Link Control Protocols

Agenda

- Flow control
- Error control
- High-level data link control (HDLC)

Data link control

- To achieve control on data transmission, a layer of logic is added above the physical layer, referred to as data link control (data link control protocol)

- The requirements needed for effective data transmission between TX and Rx:

1. Frame synchronization: recognizing the beginning and end of each frame

2. Flow control: not sending frames at a rate faster than RX can absorb them

3. Error control: bit error be corrected

4. Addressing: used in shared links

Data link control

5. Control and data on the same link: control information should be recognizable from data

6. Link management: initiation, maintenance, and termination of a sustained data exchange requires coordination and cooperation of stations

- None of these requirements are satisfied by encoding techniques in previous chapter

- We look at flow control and error control mechanisms

Data link control

5. Control and data on the same link: control information should be recognizable from data

6. Link management: initiation, maintenance, and termination of a sustained data exchange requires coordination and cooperation of stations

- None of these requirements are satisfied by encoding techniques introduced in previous chapter

- We look at flow control and error control mechanisms

Flow control

- The receiver allocates a data buffer of some maximum length for a data transfer
- The receiver does an amount of processing before delivering data to a higher-level software → frames are queued

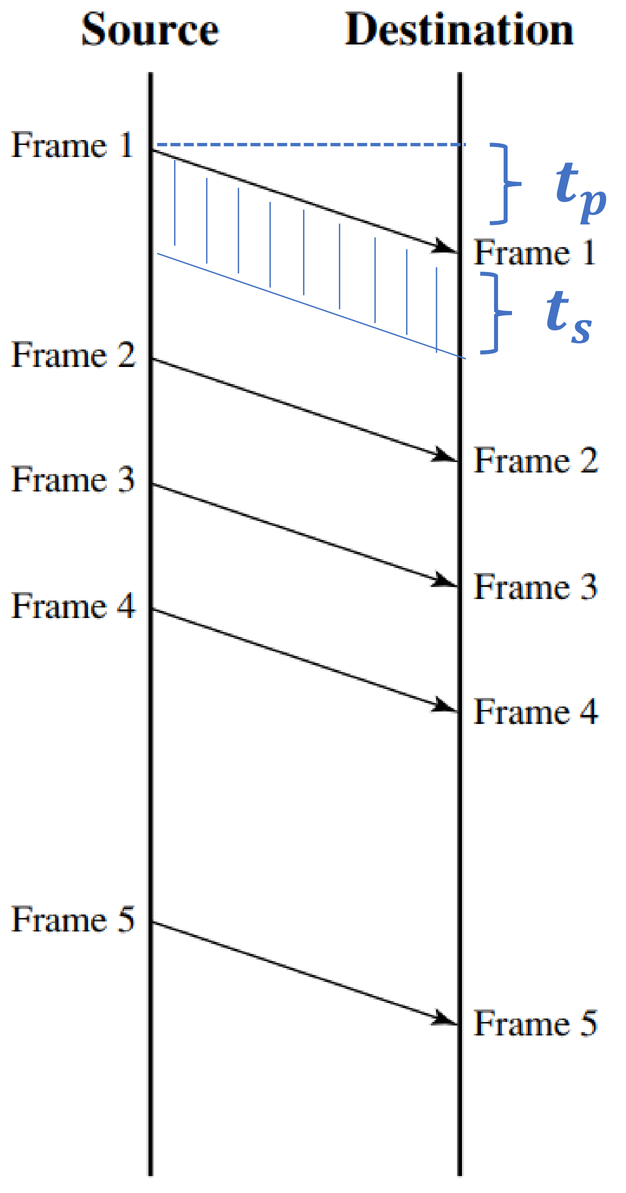
- The flow control mechanism assures that the buffer at RX does not overflow when the receiver is processing old data

Flow control

- Each arrow represents transmission of a frame
- Frame = data +control information

- Two time durations:
  - Transmission time (t_p): proportional to frame length
  - Propagation time (t_s): the time if it takes for a bit to traverse the link between TX and RX

- For flow control mechanism, we assume that the frames are received without error, in order, without loss



(a) Error-free transmission

Stop-and-wait flow control

- Transmitter sends a Frame and waits for an acknowledgment from receiver for transmitting the next frame
- If willing, receiver transmits an acknowledgment
- Receiver can stop the flow by simply withholding the ACK

- This procedure works fine if the message is transmitted in large frames
- Not efficient with small frames

- Why would the transmitter break the message in smaller pieces?

Stop-and-wait flow control

1. The buffer size of receiver is limited
2. The longer the transmission → more errors are probable → requires retransmission of the large frame

   - With smaller blocks, the error are detected sooner → less retransmissions

3. In shared mediums, not desirable to let one station occupy the link for long time → this causes long delays for other stations

- Why stop-and-wait protocol is insufficient in the presence of multiple frames?

Bit length of a link

- **Bit length of a link** is defined as follows

$$B = R \times \frac{d}{v}$$

- B: the number of bits present on the link at an instance in time when a stream of bits fully occupies the link
- R: data rate of the link in bps
- d: length or distance of the link in meters
- v: velocity of propagation in m/s
- If the transmission time is normalized to one, the propagation delay a can be written as

$$a = \frac{B}{L}$$

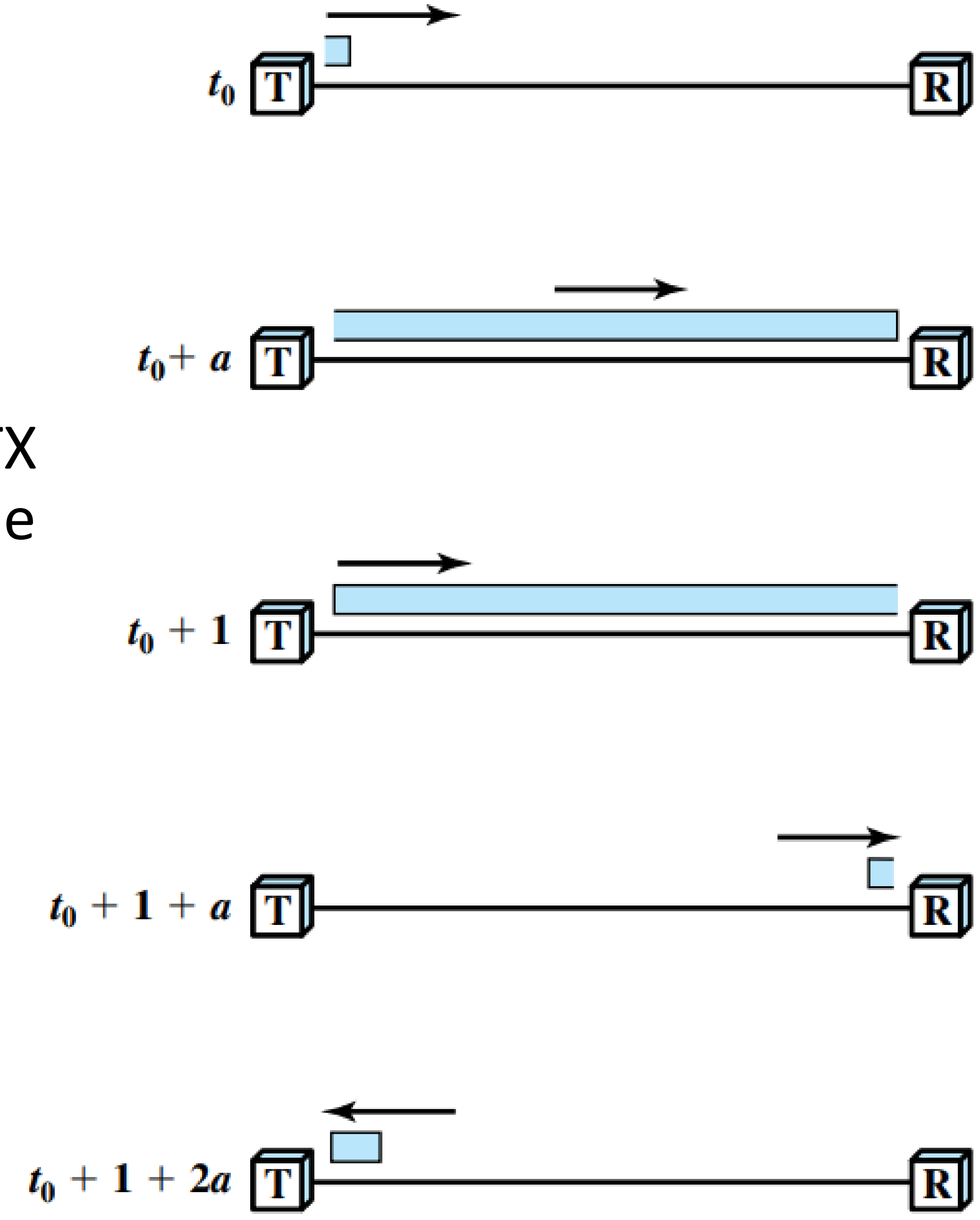
Case: a < 1

- If a < 1:

- Propagation time < transmission time

- First bit of frame has arrived at RX before TX has completed the transmission of the frame

- The link is fully utilized at some epochs
- $t_0 + a$: the first bit arrives
- $t_0 + 1$: last bit is transmitted
- $t_0 + 1 + a$: last bit is received at RX
- $t_0 + 1 + 2a$: Ack is received at TX



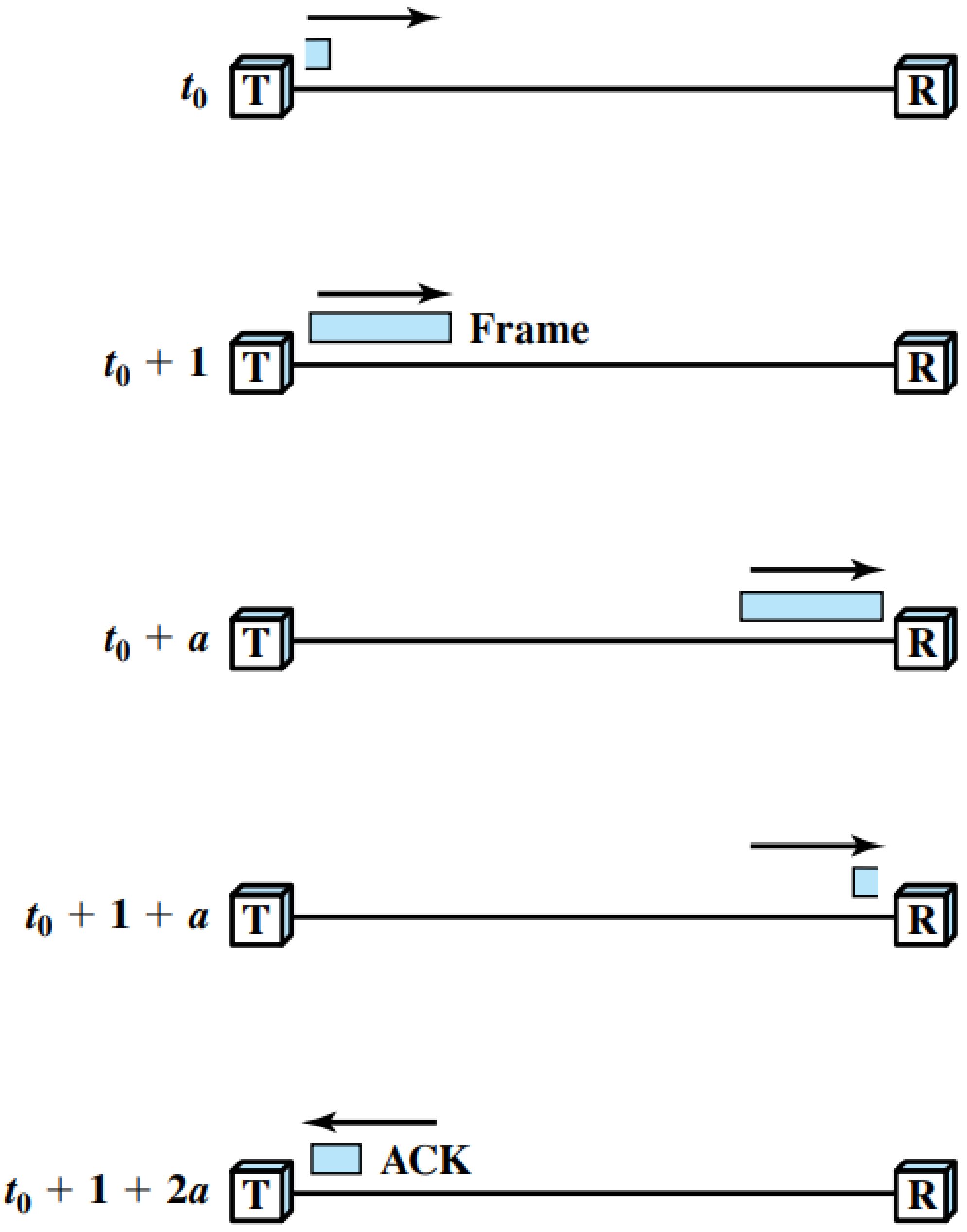
(a) $a < 1$

Case: a > 1

- If a > 1:

- Propagation time > transmission time
- TX completes the transmission of the frame before the first bit arrives at RX
- Happens in higher arrival rates or longer distances

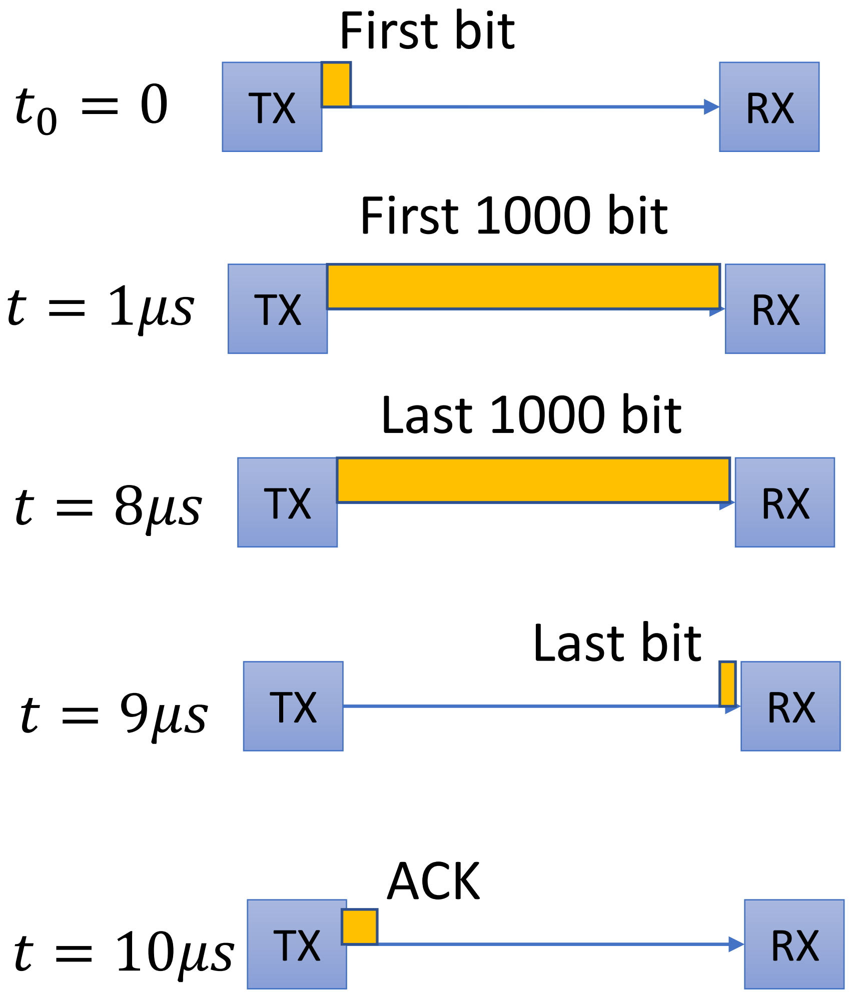
- The link is under-utilized at all epochs
- $t_0 + 1$: last bit is transmitted
- $t_0 + a$: First bit is received at RX
- $t_0 + 1 + a$: last bit is received at RX
- $t_0 + 1 + 2a$: Ack is received at TX



(b) $a > 1$

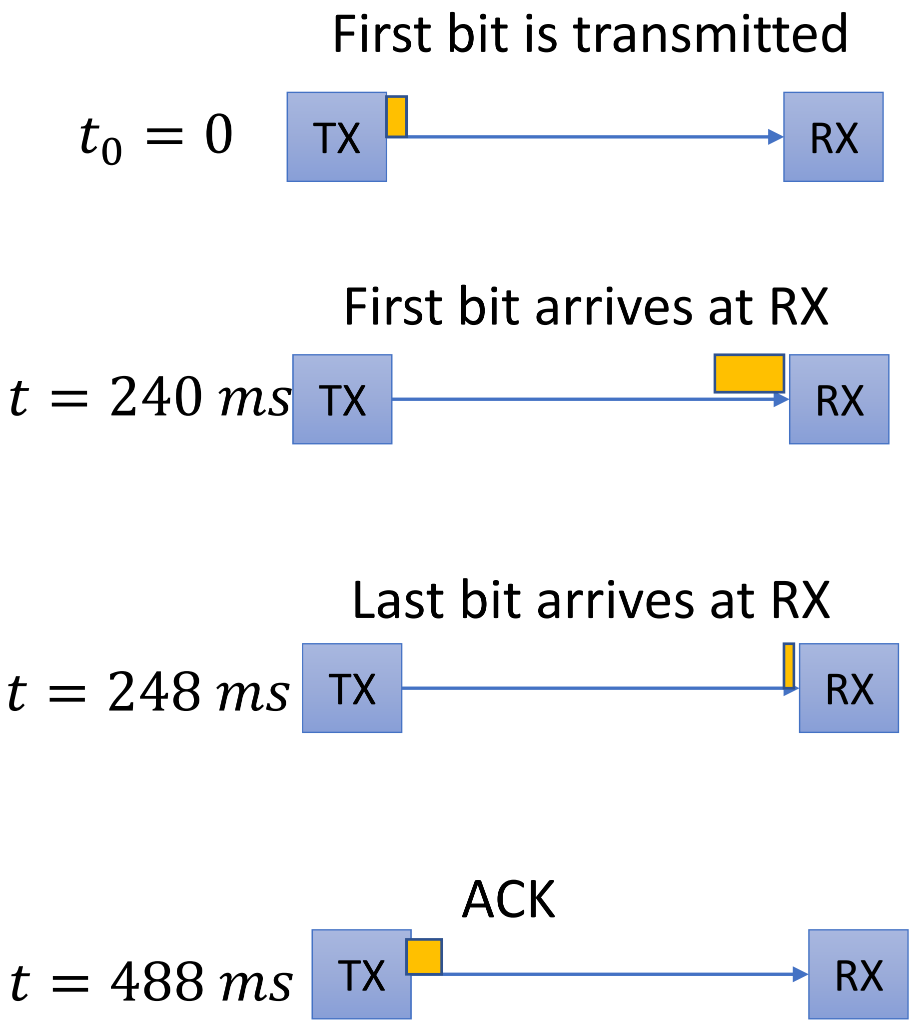
Example

- Consider a 200 m optical fiber operating at **1 Gbps**. Also, $v = 2 \times 10^8$ m/s, $L = 8000$ bits.
- $B = 10^9 \times \frac{200}{2 \times 10^8} = 1000\ bits$
- $a = \frac{1000}{8000} = 0.125$
- Transmission time = $\frac{8000}{1\ Gbps} = 8\ \mu s$
- $a = 0.125 \times 8\ \mu s = 1 \mu s$



Example

- Consider a link between two ground stations that communicate via a satellite relay at **1 Mbps**. Also, $d = 36000$ km, $L = 8000$ bits.
- $B = 10^6 \times \frac{2 \times 36 \times 10^6}{3 \times 10^8} = 240{,}000\ bits$
- $a = \frac{240{,}000}{8000} = 30$
- Transmission time = $\frac{8000}{1\ Mbps} = 8\ ms$
- $a = 30 \times 8\ ms = 240 \mu s$



Sliding window flow control

- The problem in stop-and-wait protocol

  - One frame is in transit

  - Serious inefficiency when $a > 1$

- **Solution:** allow multiple frames to be in transit at the same time
- How sliding -window flow control works:

  - Assume a full-duplex transmission link between stations A and B

  - Station B allocates a buffer space for accepting W frames

  - Station A is allowed for transmission of W frames without waiting for acknowledgements

Sliding window flow control

- Each frame has a sequence number to keep track of the frames that have been acknowledged

- When B acknowledges a frame, it sends the sequence number of the next frame it expects

  - Implicitly announces that B is prepared to receive the next W frames

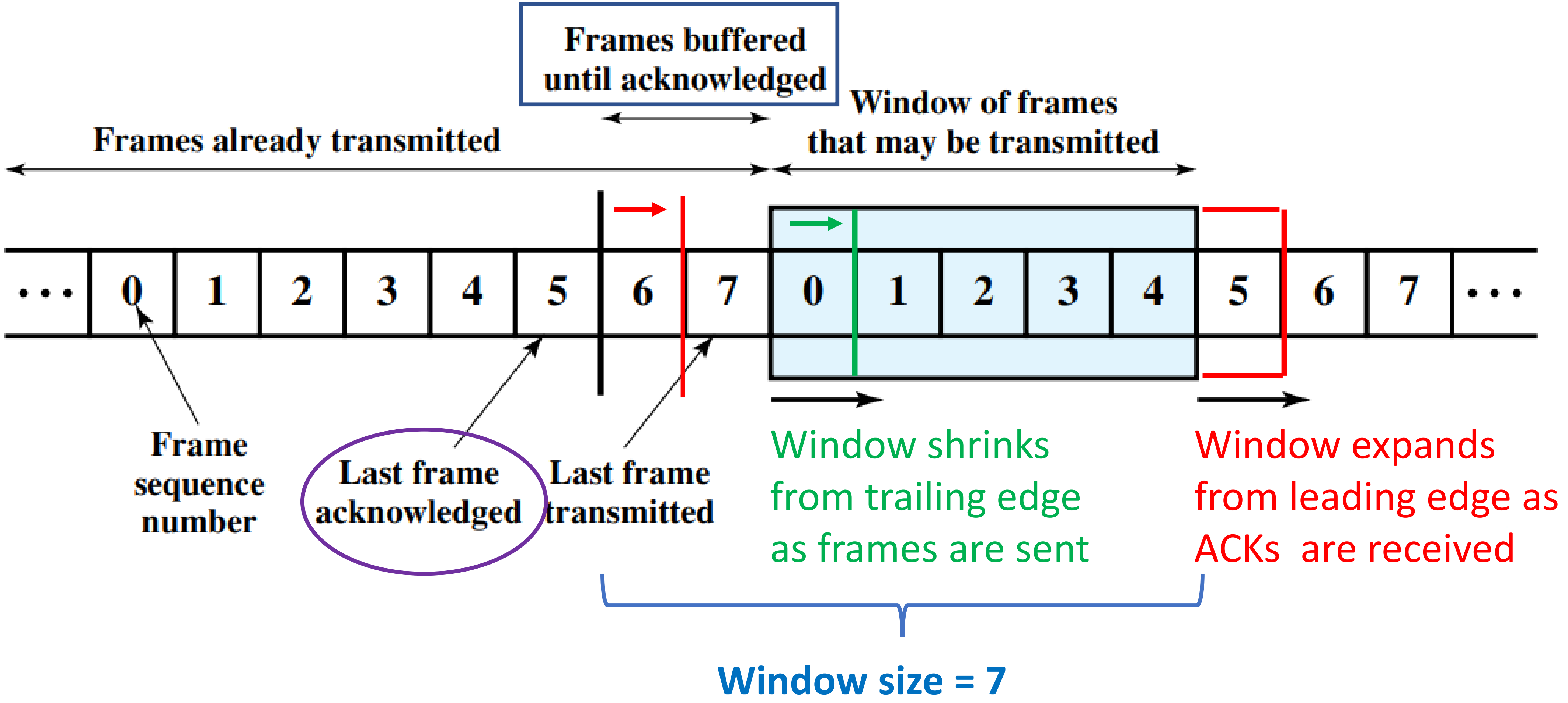
  - Used for acknowledging multiple frames
    - E.g., B receives frames 2,3,4 but acknowledges 4 with sending an ACK with sequence number 5

Sliding window flow control

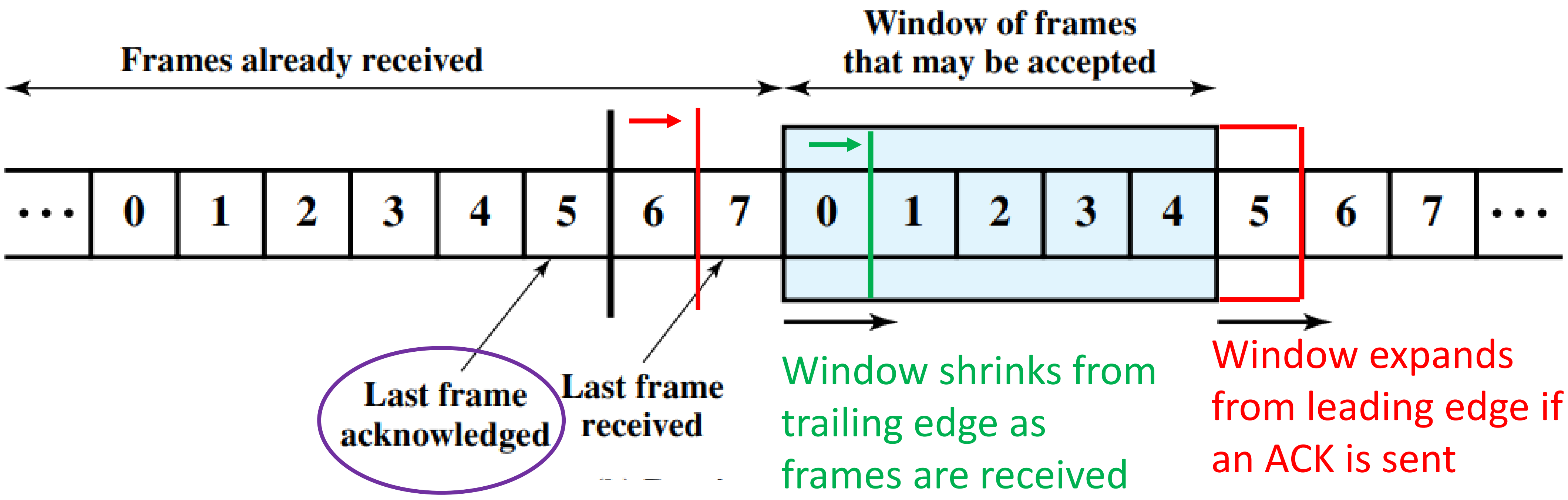
- Station A maintains a list of sequence numbers it is allowed to send
- Station B maintains a list of sequence numbers it is prepared to receive
- Each list can be thought of a window of frames → This is why it is called sliding window protocol

- Sequence number occupies a field in the frames → has limited values
- Example: for a 3-bit field, the sequence number can be from 0 to 7
- For a k-bit field, the range of sequence number is from 0 to $2^k - 1$
- We will see the sliding window size is $2^k - 1$

Transmitter side



Receiver side



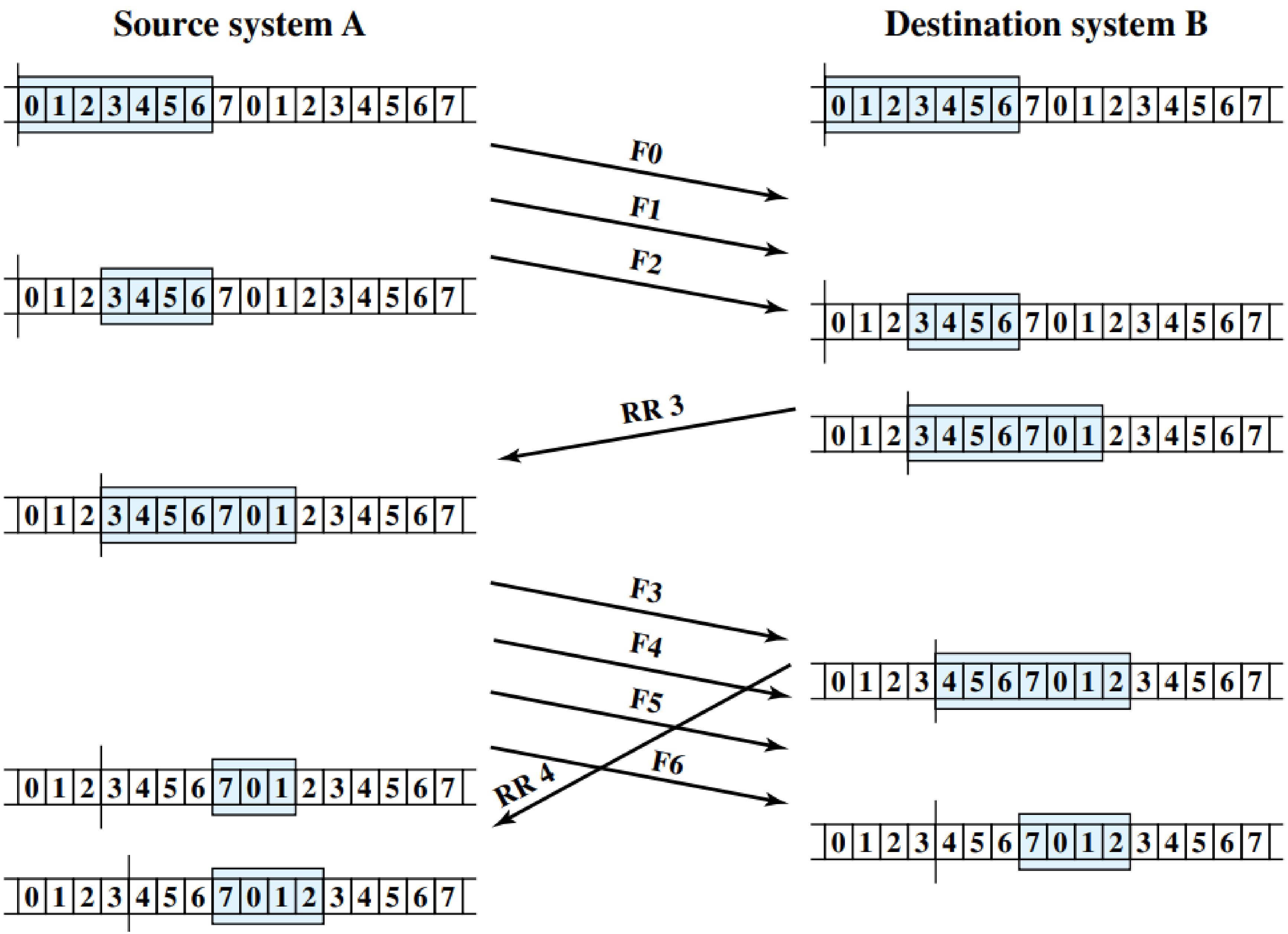


Figure 7.4 Example of a Sliding-Window Protocol

Sliding window flow control

- The sliding window may not be the maximum possible size
  - E.g., for a 3-bit sequence number, sliding window of size 5 could be used

- RX can cut off the flow of the frames by sending **Receive Not Ready (RNR)**
  - E.g., RNR 5 means: I have received all frames up to 4 but I am not ready to accept more frames
  - Send a normal Receive Ready (RR) to reopen

- Full-duplex transmission: each side needs to send data and ACK

  - **Piggybacking:** each frame includes a filed for sequence number of the frame and a field for the sequence number of acknowledgment

Sliding window flow control

- Full-duplex transmission:

  - have acknowledgment but no data → use RR and RNR

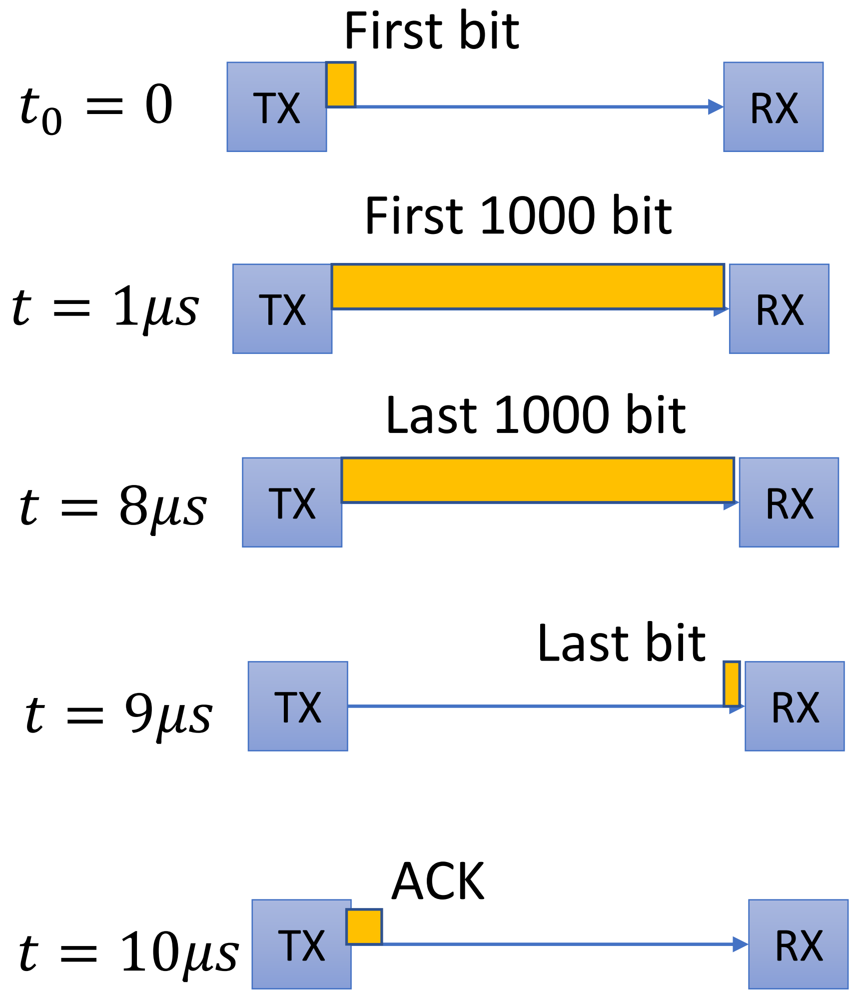
  - have acknowledgment and data → piggybacking: send both in one frame

  - have data but no new acknowledgment → repeat the last acknowledgement sequence number

- Sliding-window more efficient than stop-and-wait protocol since allows pipelining → the link is filled with multiple frames in transit

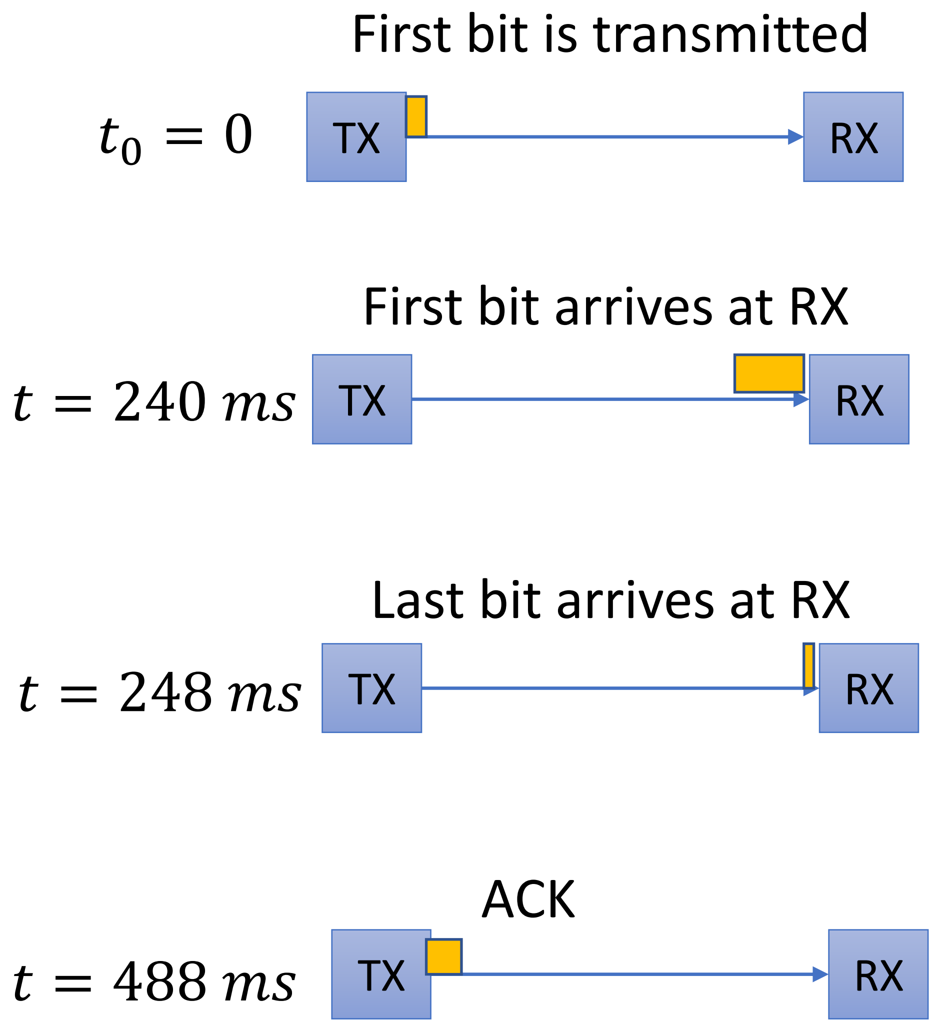
Example

- In the previous example → 10 μs take to receive an ack for the first frame
- 8 μs for transmission of each frame
- A frame and part of second can be transmitted meanwhile
- A sliding window of 2 is adequate for a continuous transmission
- Or a rate of one frame per 8 μs
- In stop-and-wait, the rate is one frame per 10 μs



Example

- It takes 488 ms for an ACK to the first frame to be received
- 8 ms for a frame transmission
- 61 frames can be transmitted before the first ACK
- With 6-bit window field or more, the transmitter can send continuously with a rate of one frame per 8 ms
- With 3-bit field, 7 frames can be sent and then wait for the first ACK → 7 frames per 488 (or one frame in 70ms)
- In stop-and-wait protocol → one frame per 488 ms

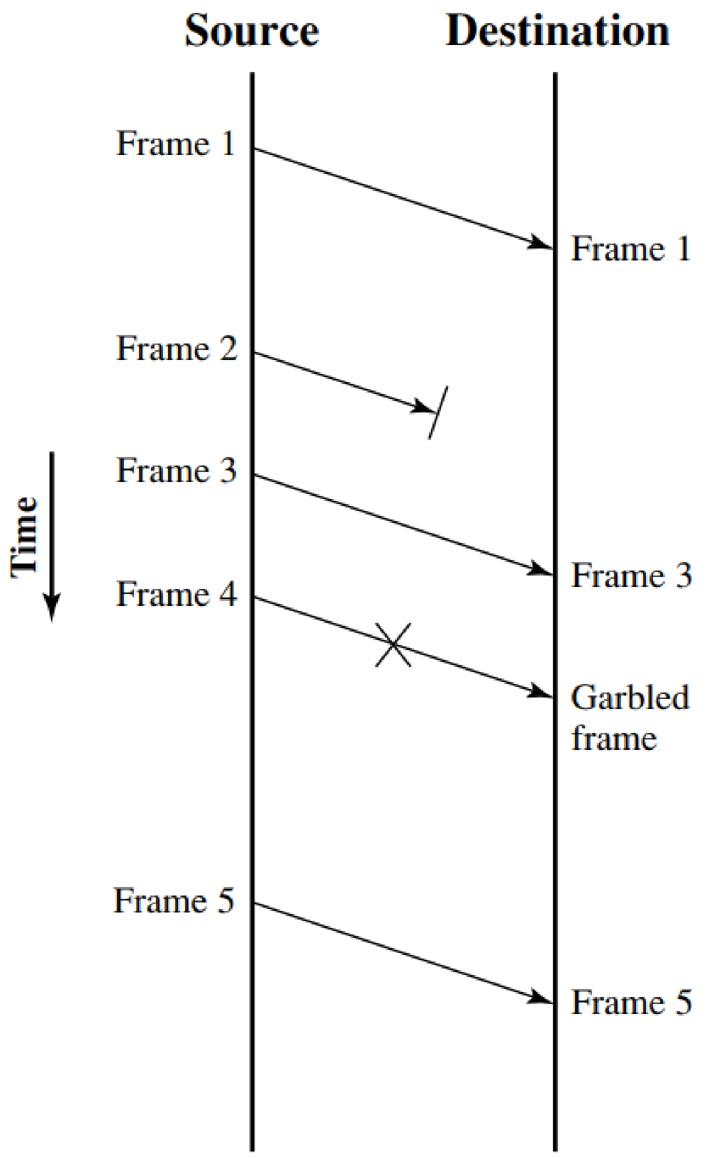


Error control

- Mechanisms to detect and correct errors
- We assume the frames arrive in order with variable delays
- Two types of error are considered

1. **Lost frame:** A frame fails to arrive at the other side, e.g., due to severe noise

1. **Damaged frame:** A recognizable frame does arrive, but some of the bits are in error



(b) Transmission with losses and errors

Error control

- Error control mechanisms are based on one or more of following techniques

1. Error detection: as discussed in previous chapter
2. Positive acknowledgment → The destination returns a positive acknowledgment to successfully received, error-free frames
3. Retransmission after timeout → The source retransmits a frame if no ACK is received after a predetermined amount of time.
4. Negative acknowledgment and retransmission → The destination returns a negative acknowledgment to frames in which an error is detected. The source retransmits such frames.

Error control

- All are referred to as **automatic repeat request (ARQ)**;
- The effect of ARQ is to turn an unreliable data link into a reliable one.

- Three versions of ARQ have been standardized

1. Stop-and-wait ARQ
2. Go-back-N ARQ
3. Selective-reject ARQ

- All based on flow control techniques we discussed

Stop-and-wait ARQ

- Transmitter send a frame and waits until an ACK is received
- No other new frame can be transmitted
- Two types of error can happen

1. Damaged frame:

❖ Receiver detects the error using error-detecting schemes and discard the frame

- A timer is used at transmitter

❖ The timer expires → retransmits the frame

- Transmitter should keep a copy of the frame

Stop-and-wait ARQ

2. Damaged ACK:

- ❖ Station A send a frame
- ❖ The frame is received correctly by B
- ❖ Station B sends an ACK
- ❖ ACK is damaged and not recognizable by A
- ❖ Station A times out and retransmit the frame
- ❖ Station B receives two copies of a frame as if they are separate

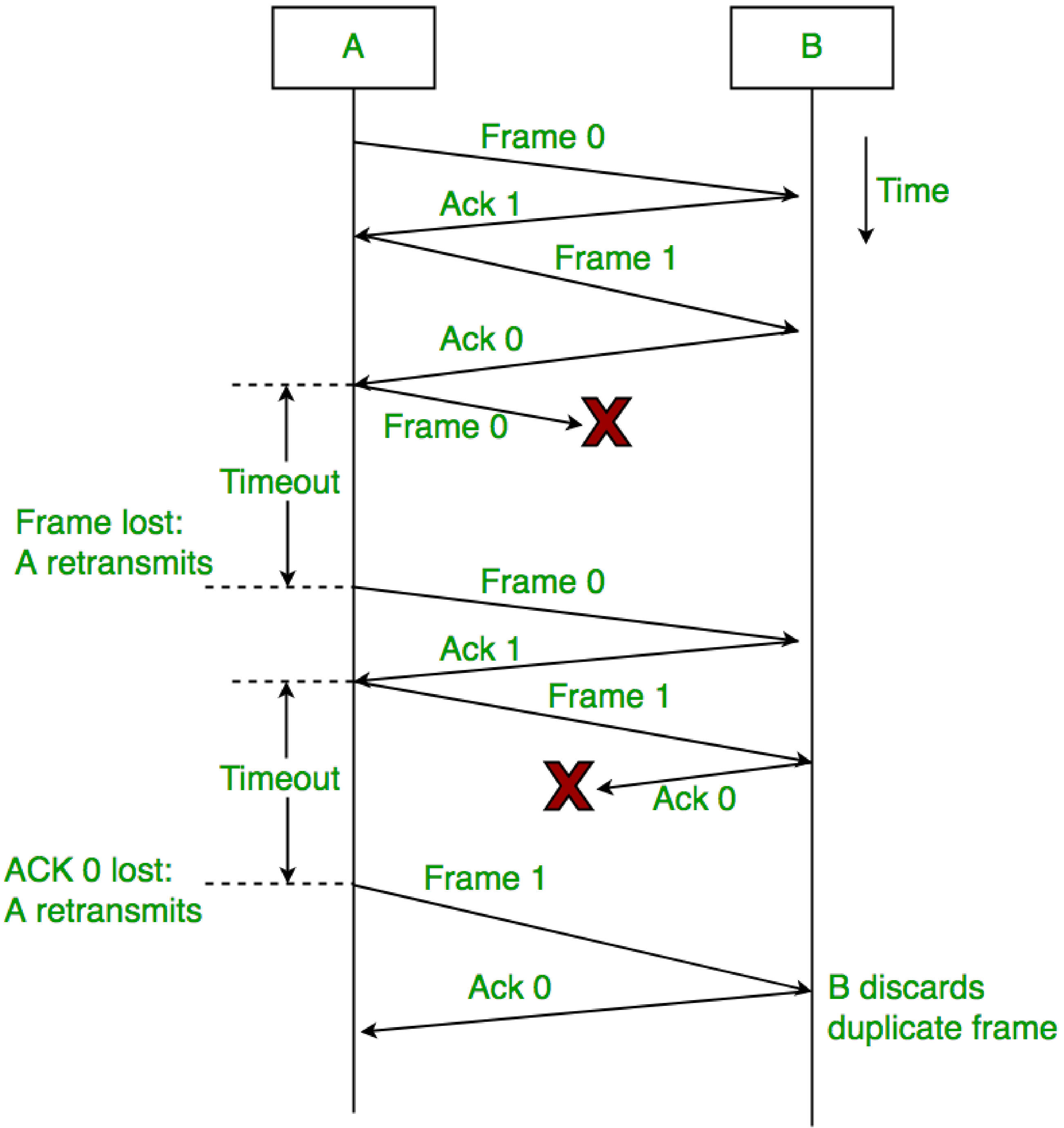
- To avoid this problem the frames are alternatively labeled with 0 and 1

  - Positive ACKs are in the form ACK0 and ACK1.

  - ACK0 acknowledges Frame 0 and indicates that B is waiting for sequence number 1

Stop-and-wait ARQ

- Main advantage: Simplicity
- Main disadvantage: Inefficient
- Sliding window protocol can be adapted to provide more efficient line use



Go-back-N ARQ

- Based on sliding window flow control

- **Principals:**

1. A sends a series of frames sequentially numbered modulo some maximum value
   - E.g., maximum value = 8 → 0,1,2,…,7,0,1,2,..

2. The number of unacknowledged frames = window size = N
   - E.g., window size = 7 and Seq. number of the last acknowledged frame is 5 → 6,7,0,1,2,3,4
   - E.g., window size = 5 and Seq. number of the last acknowledged frame is 5 → 6,7,0,1,2

Go-back-N ARQ

- **Principals:**

3. If no error happens, B sends RR or piggybacks the ACK message

4. If error happens B sends a negative ACK, called REJ

5. B discards the damaged frame and all subsequent frames

6. When A receives REJ, retransmits the frame in error and all subsequent frames
7. A sets an acknowledgment timer for the frame just transmitted

Possible scenarios

- Suppose B has received frame i-1 successfully and A has just transmitted frame I

- Three possible scenarios:

1. Damaged frame → the received frame is invalid → B discard the frame and take no action

2. Damaged RR → RR is lost

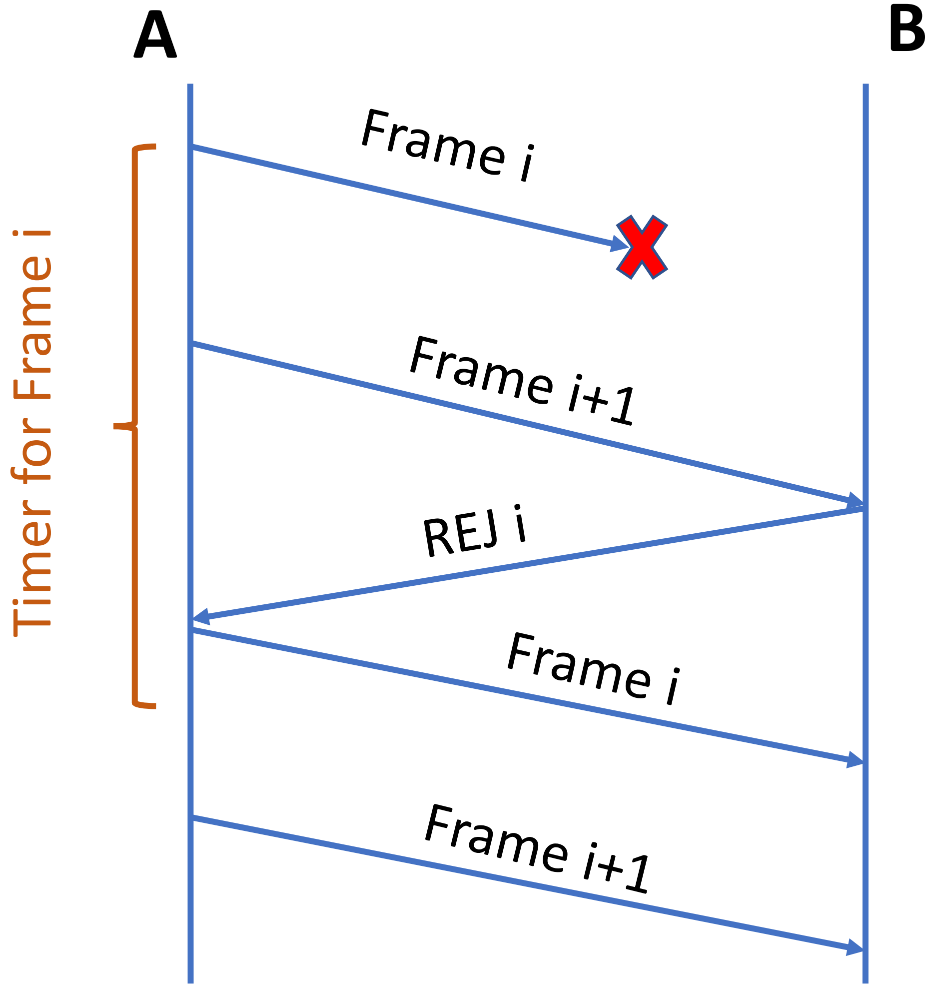
3. Damaged REJ → REJ is lost

Damaged frame: Case 1

- **A transmits an additional frame i+1 before the timer expires**

- B receives frame i+1 out of order and sends REJ I

- A retransmits frame i and i+1



Damaged frame: Case 2

- **A Sends no additional frame**

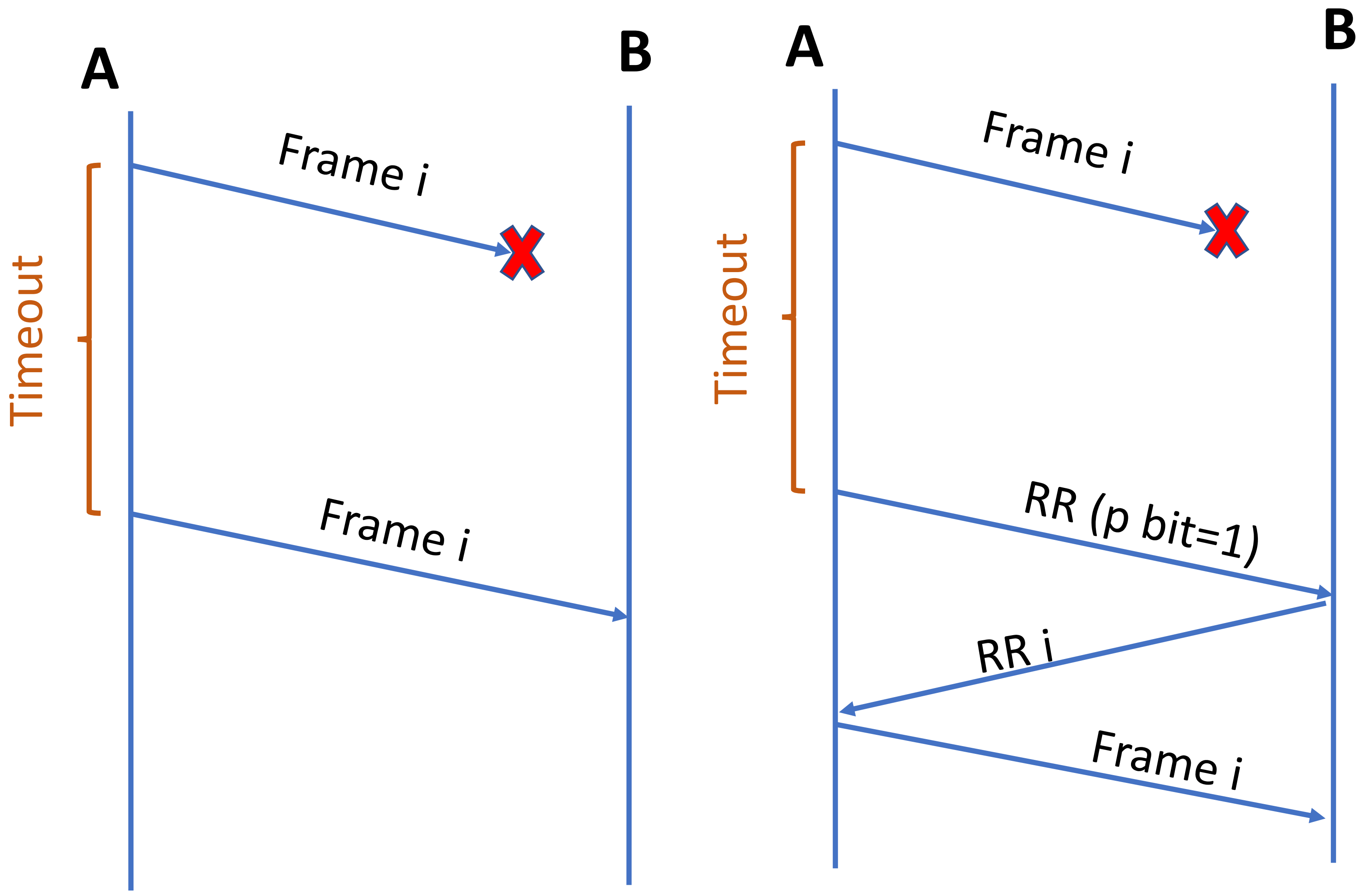
- Timer at A expires

1. A sends an RR message which have a P bit equal to one, asking B which frame it is expecting to receive

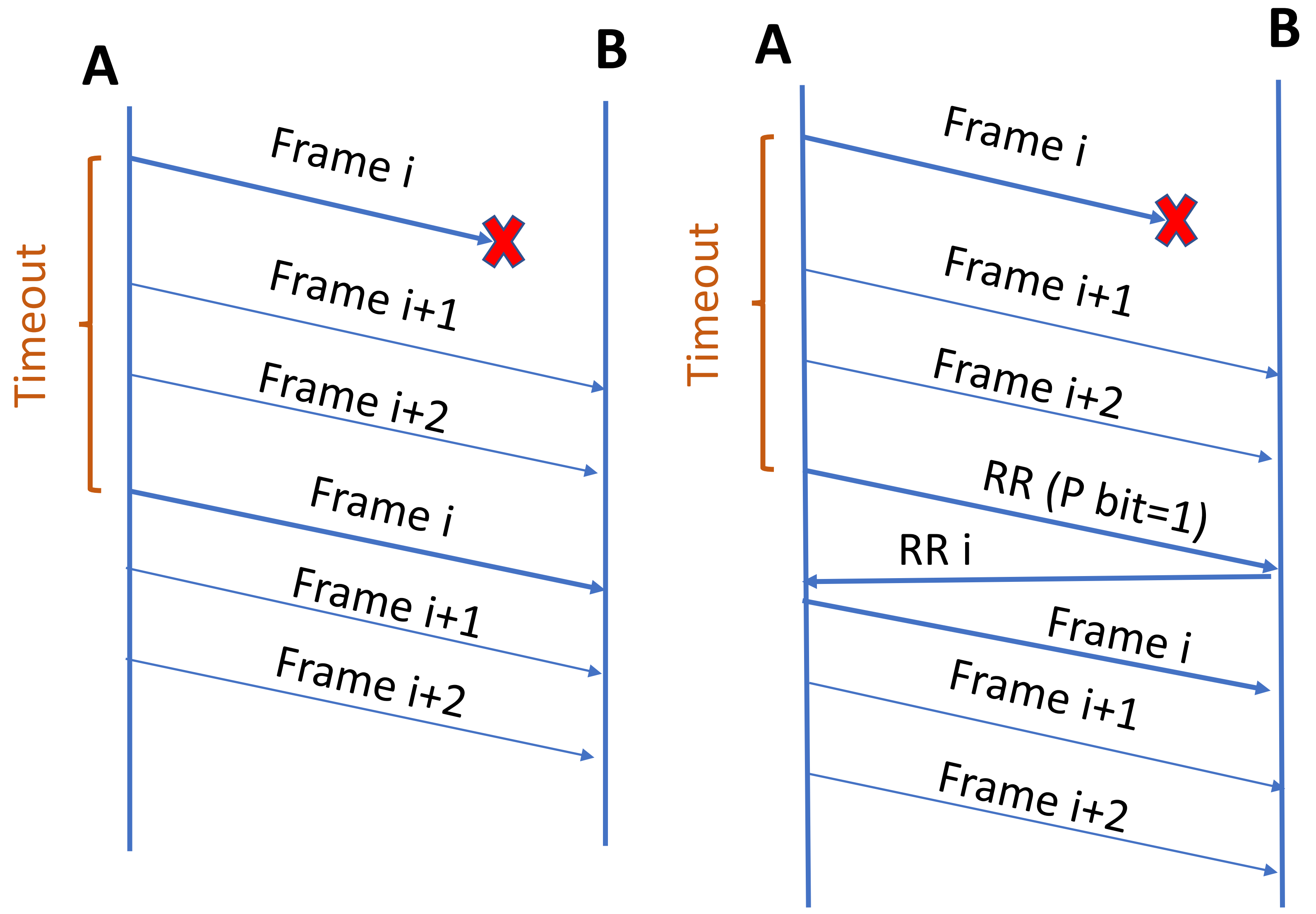
   - B sends RR i

   - A retransmits frame I

2. A retransmits frame i

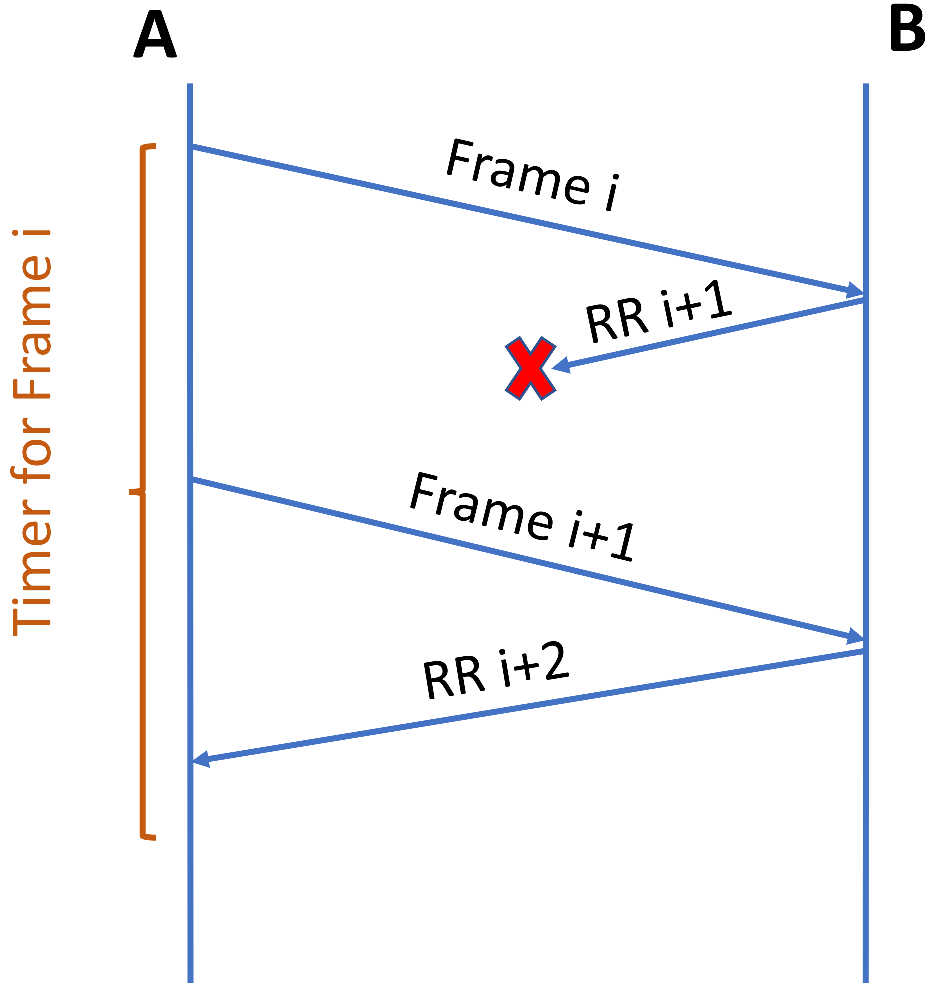


Damaged frame: Case 2



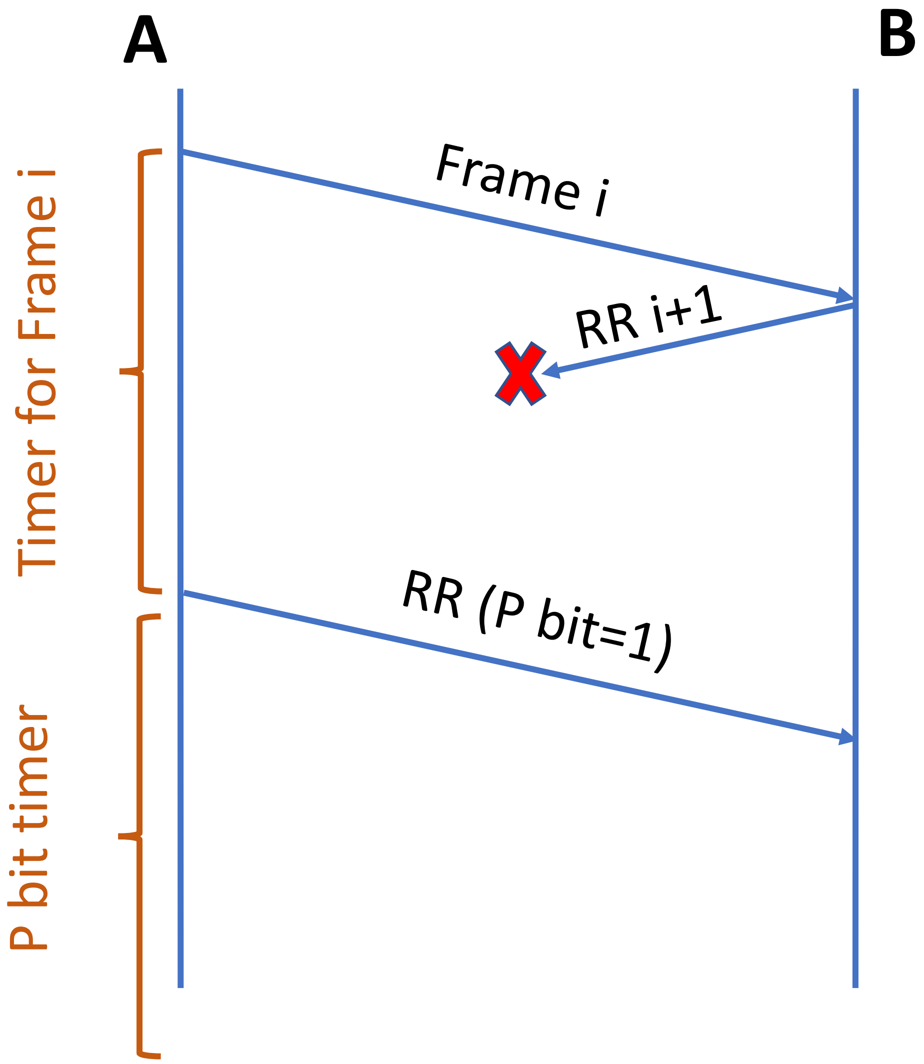
Damaged RR: Case 1

- B receives i and sends RR i+1
- RR i+1 suffers an error
- Before the timer expires, A send another frame and receives RR



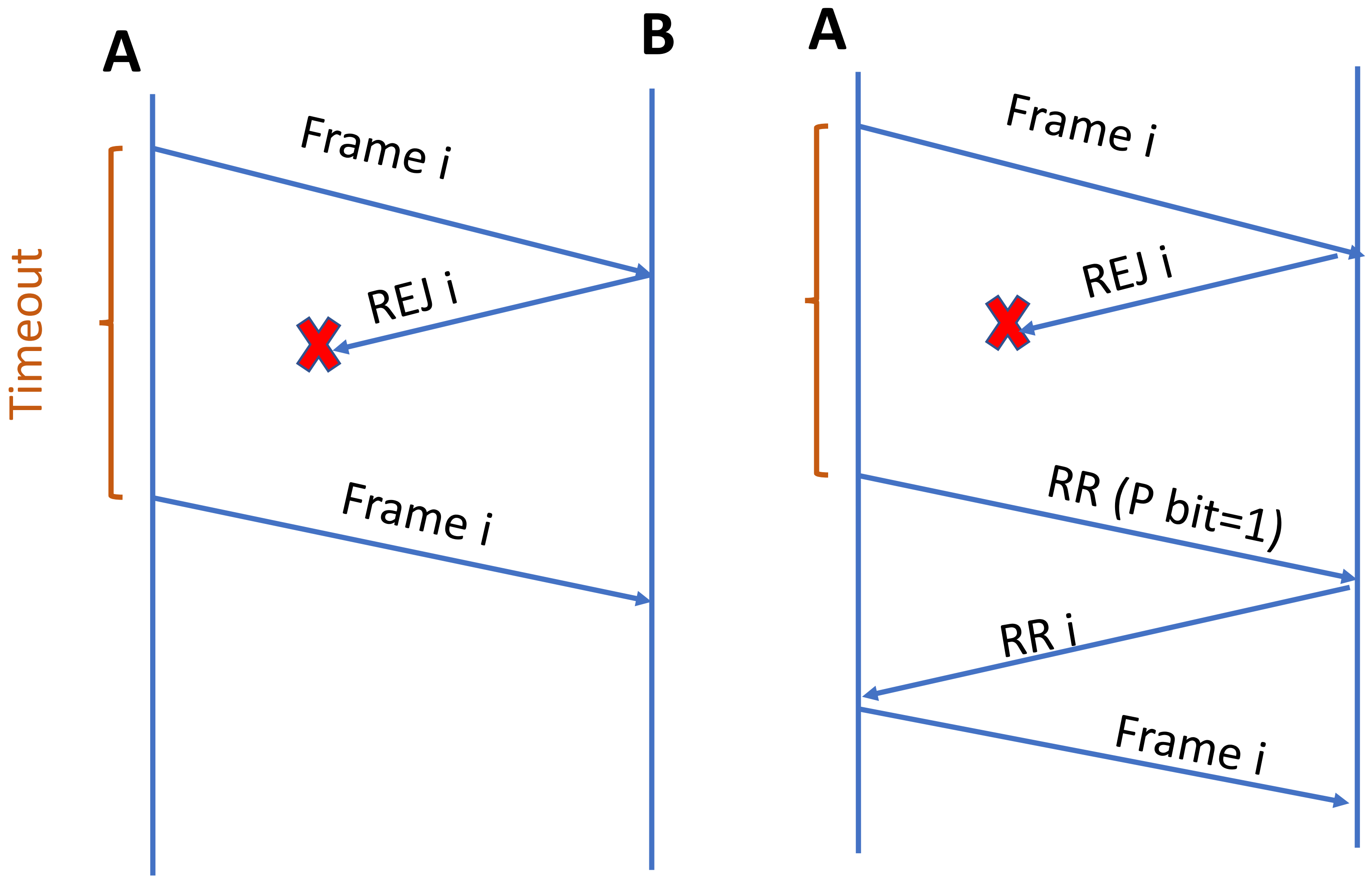
Damaged RR: Case 2

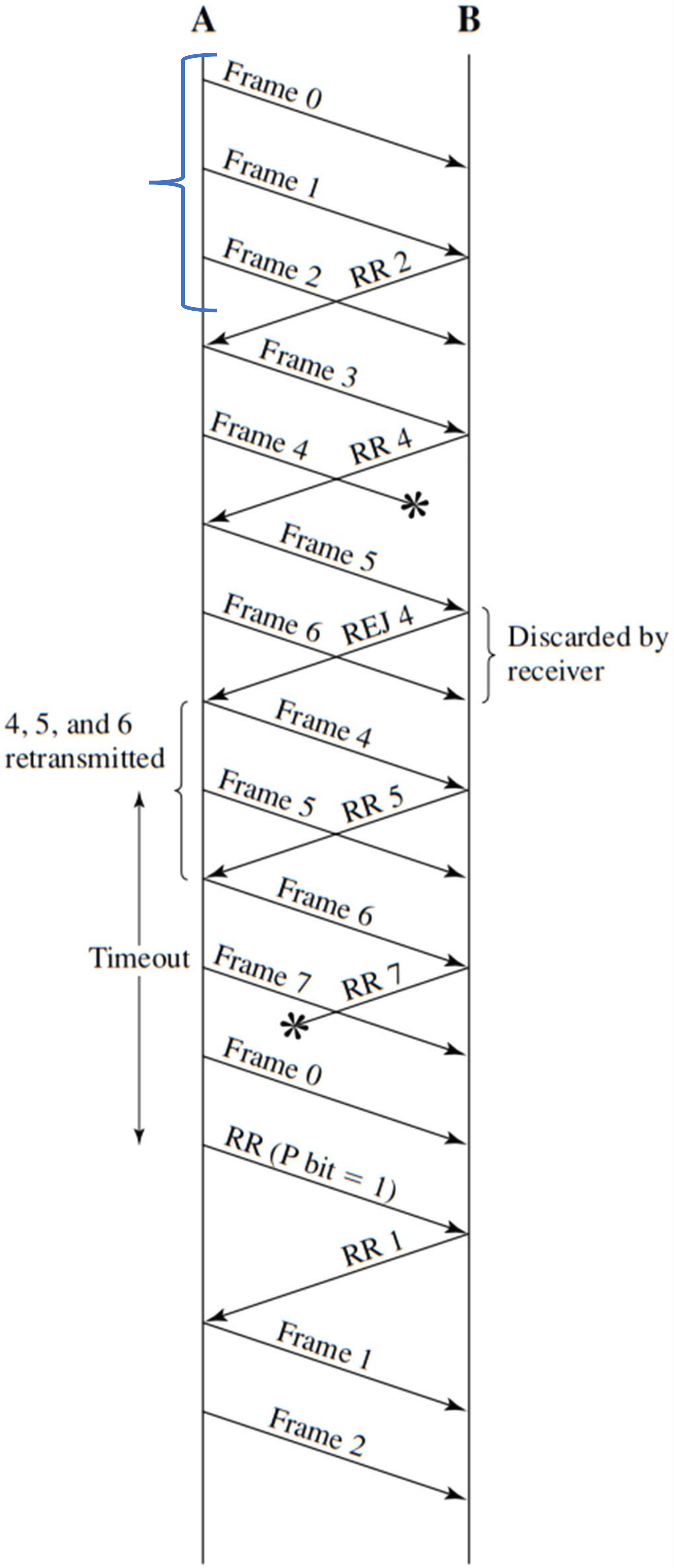
- B receives i and sends RR i+1
- RR i+1 suffers an error
- Timer expires
- A send an RR message with P bit =1 and sets a P bit timer
- If RR is not responded or the response is lost, issue new RR and sets another P bit timer
- Does this for a maximum number of times



Damaged REJ

- Similar to damaged frame with timeout





(a) Go-back-N ARQ

Window size in Go-back-N ARQ

- If k-bit field is dedicated to sequence number

  → Sequence numbers: 0 to $2^k - 1$

  → Window size: $2^k - 1$

- Suppose there is a full-duplex link between A and B
- k = 3 and N (window size) = $2^k = 8$

- A sends frame 0 and receives RR 1 → A sends 1,2,3,4,5,6,7,0
- B is piggybacking → it must fill the acknowledgment field
  - If no new ACKs are available, the previous one is duplicated

Window size in Go-back-N ARQ

- If it send RR1, two meaning are possible

1. B has received all frames error-less and waiting for frame 1

   - **New cumulative ACK** : 1,2,3,4,5,6,7,0 all received successfully → RR 1

2. B has lost all frames and is just repeating its previous ACK

   - **Repeating ACK**: Frame 1 received in error → RR 1

- To avoid the above situation, the window size is set to 7

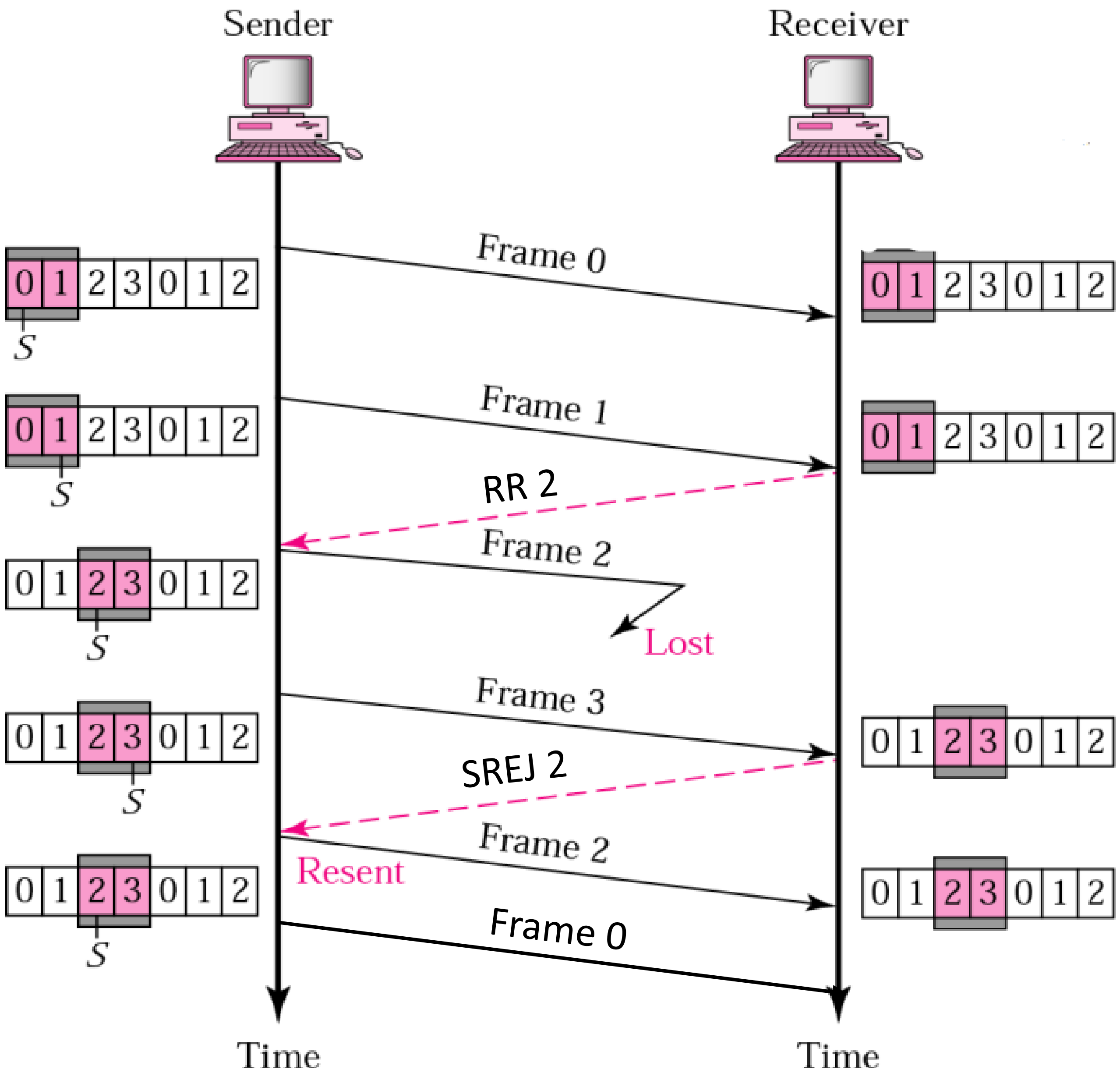
  - A transmits 1,2,3,4,5,6,7

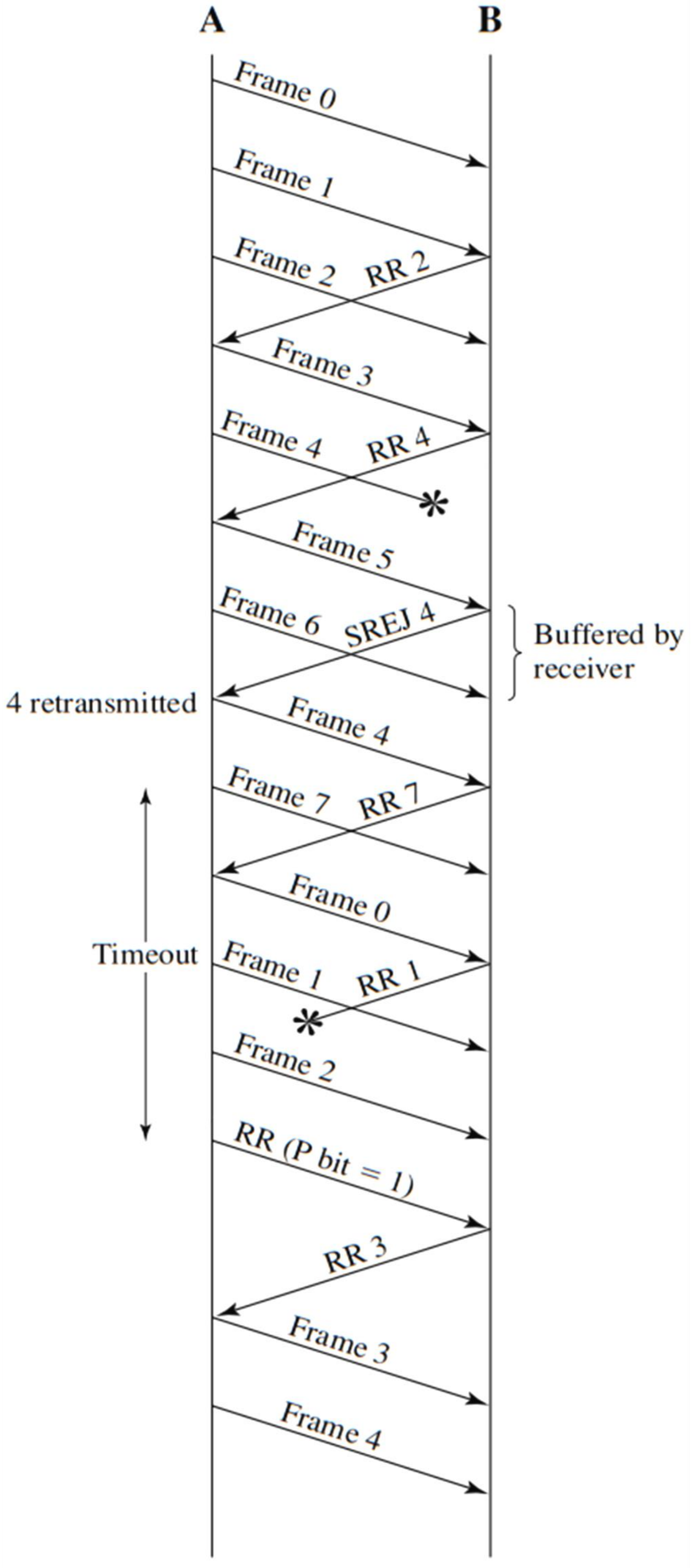
  - B sends **New cumulative ACK:** RR 0

  - B sends **Repeating ACK**: RR 1

Selective-Reject ARQ

- With selective-reject ARQ, the only frames retransmitted are those that receive a negative acknowledgment, called SREJ, or those that time out.





(b) Selective-reject ARQ

Window size in Selective-reject ARQ

- If k-bit field is dedicated to sequence number

  → Sequence numbers: 0 to $2^k - 1$

  → Window size: 2^{k-1}

- Suppose k = 3 and N = 7
- A sends frame 0 through 6 to B → 0,1,2,3,4,5,6
- B receives all frames and acknowledges with RR 7

  - window at B → 7,0,1,2,3,4,5
- RR 7 is lost
- A times out and retransmits 0
- B assumes that this is a new 0 and 7 is lost → a wrong frame 0 is accepted

Window size in Selective-reject ARQ

- How is the problem solved?
- The sequence numbers in the window of transmitter and receiver should not overlap
- In our example after timeout

  - A window: 0,1,2,3,4,5,6
  - B window: 7,0,1,2,3,4,5

  0,1,2,3,4,5 exist in both windows

- If we set window size to 4 the problem is solved

  - A window: 0,1,2,3
  - B window: 4,5,6,7

- What happens in Go-back-N

  - After receiving frame 0, A rejects it since its out of order

Selective-reject vs. Go-back-N ARQ

- Selective-reject is more efficient since it minimizes the amount of retransmissions

- **Cons:**

1. Rx must have a large enough buffer to store the post-SREJ frames until the frame is error is retransmitted
2. Also Rx must contain logic to insert the received frame in proper position
3. TX needs more complex logic to transmit a frame out of sequence

- Selective-reject much less widely used than Go-back-N
- Selective-reject is useful in satellite links due to large propagation delay

High-level data link control (HDLC): Basics

- Basic Characteristic:
  - Three types of stations
  - Two link configurations
  - Three data transfer modes of operation

HDLC: Station types

1. Primary station:

   - Responsible for controlling the operation of the link
   - Frames issued called commands

2. Secondary station

   - Operates under the control of the primary station
   - Frames issued called responses

3. Combined station

   - Combines the features of primary and secondary
   - Issues both commands and responses

HDLC: Link configurations

1. Unbalanced configuration:

   - Consists of one primary and one or more secondary station

   - Supports both full-duplex and half-duplex transmissions

2. Balanced configuration:

   - Consists of two combined stations

   - Supports both full-duplex and half-duplex transmission

HDLC: Data transfer modes

1. Normal response mode (NRM):

   - Used in unbalanced configuration

   - Primary initiates data transfer to a secondary

   - Secondary transmits data only if a command from the primary is received

2. Asynchronous balanced mode (ABM):

   - Used in balanced configuration

   - Either combined station may initiate transmission

3. Asynchronous response mode (ARM):

   - Used in unbalanced configuration

   - Secondary initiates transmission without permission of the primary

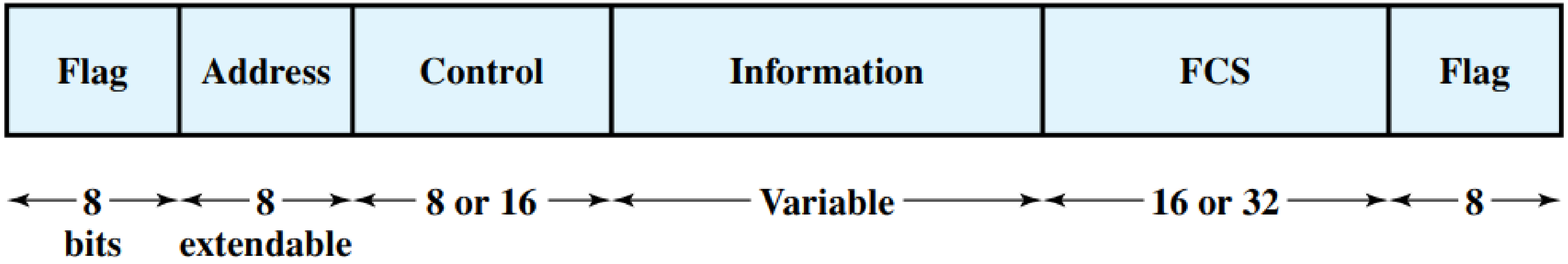
   - Primary still retains responsibility for the line

HDLC: Data transfer modes

- NRM used in multi-drop lines
  - Several terminals are connected to a host computer.
  - Computer polls each terminal for input
- NRM used in point-to-point lines
  - link connects a terminal or other peripheral to a computer

- ABM is the most widely used
  - There is no polling overhead → efficient use of link
- ARM is rarely used
  - For some special situations where a secondary needs to initiate transmission

HDLC: Frame structure



(a) Frame format

- Header = (Flagg, Address, Control)
- Trailer =(FCS,Flag)

Frame structure: Flag fields

- Flag pattern: 01111110
  - A single flag used as the closing for one frame and the opening for the next
  - Both sides look for 01111110 to synchronize on the start of a frame
  - After receiving, a station looks for 01111110 to end the frame.
- Pattern 01111110 may appear somewhere inside the frame → destroying synchronization
- Solution → **bit stuffing:** Inserts an extra 0 bit after each occurrence of five 1s in the frame (between flags).

Original pattern:

111111111111011111101111110

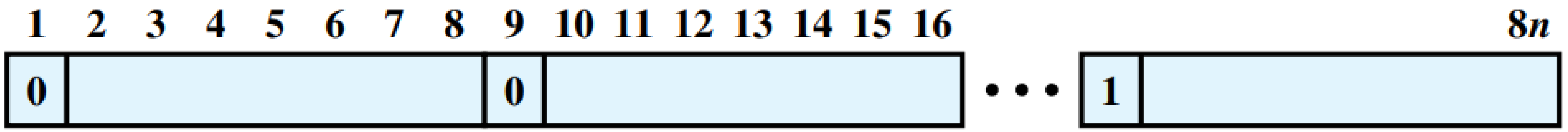
After bit-stuffing:

1111101111101101111101011111010

Frame structure: Address field

- Identifies the secondary user which is transmitting or going to receive
- 8-bit long
- Extended format: multiple of 7 bits

  - Leftmost bit of each octet 0 or 1 → 1 indicates the last octet
- Single-octet address of 11111111 targets all-stations address → used for broadcasting



(b) Extended address field

Frame structure: Control field

HDLC defines three types of frame with different control fields

1. Information frames (I-frame):

   - Carry data

   - Used to piggyback flow and error control data using ARQ

2. Supervisory frames (S-frame)

   - Provide ARQ mechanism when piggybacking is not used

3. Unnumbered frames (U-frame)

   - Provide supplemental link control functions

Frame structure: Control field (2)

- The first one or two bits of the control field serves to identify the frame type
- All control field formats contain the poll/final (P/F) bit
  - In command frames, referred to as the **P bit** and is set to 1 to poll a response frame
  - In response frames, referred to as the **F bit** and is set to 1 to indicate the response frame
- Seq. number in S- and I-frames → Basic control filed: 3-bit , Extended format: 7-bit
- U-frames → always 8-bit

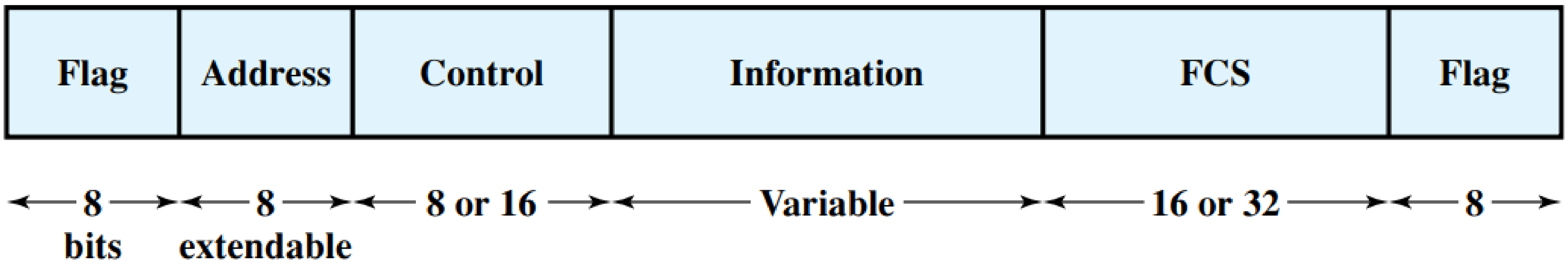
	1	2	3	4	5	6	7	8
I: Information	0	N(S)			P/F	N(R)		
S: Supervisory	1	0	S		P/F	N(R)		
U: Unnumbered	1	1	M		P/F	M		

N(S) = Send sequence number
N(R) = Receive sequence number
S = Supervisory function bits
M = Unnumbered function bits
P/F = Poll/final bit

Frame structure: Information and FCS fields

- The information field is present only in I-frames and some U-frames.
- Can contain any sequence of bits but must consist of an integral number of octets

- The frame check sequence (FCS) is an error detecting code calculated from the remaining bits of the frame, exclusive of flags.
- The normal code is the 16-bit CRC-CCITT → $\text{CRC-CCITT} = X^{16} + X^{12} + X^{5} + 1$



HDLC operation

- Exchange of I-frames, S-frames, and U-frames between two stations
- Various commands and responses defined for these frame types
- 3 Phases: 1) initialization 2) data transfer 3) termination

Name	Command/ Response	Description
Information (I)	C/R	Exchange user data
Supervisory (S)		
Receive ready (RR)	C/R	Positive acknowledgment; ready to receive I-frame
Receive not ready (RNR)	C/R	Positive acknowledgment; not ready to receive
Reject (REJ)	C/R	Negative acknowledgment; go back N
Selective reject (SREJ)	C/R	Negative acknowledgment; selective reject

Unnumbered (U)		
Set normal response/extended mode (SNRM/SNRME)	C	Set mode; extended = 7-bit sequence numbers
Set asynchronous response/extended mode (SARM/SARME)	C	Set mode; extended = 7-bit sequence numbers
Set asynchronous balanced/extended mode (SABM, SABME)	C	Set mode; extended = 7-bit sequence numbers
Set initialization mode (SIM)	C	Initialize link control functions in addressed station
Disconnect (DISC)	C	Terminate logical link connection
Unnumbered Acknowledgment (UA)	R	Acknowledge acceptance of one of the set-mode commands
Disconnected mode (DM)	R	Responder is in disconnected mode
Request disconnect (RD)	R	Request for DISC command
Request initialization mode (RIM)	R	Initialization needed; request for SIM command
Unnumbered information (UI)	C/R	Used to exchange control information
Unnumbered poll (UP)	C	Used to solicit control information
Reset (RSET)	C	Used for recovery; resets N(R), N(S)
Exchange identification (XID)	C/R	Used to request/report status
Test (TEST)	C/R	Exchange identical information fields for testing
Frame reject (FRMR)	R	Report receipt of unacceptable frame

HDLC operation: Initialization

- Either side may request initialization by issuing one of the six set mode commands
- Goals:

1. Signals the other side that initialization is requested.
2. Specifies which of the three modes (NRM,ABM,ARM) is requested.
3. Specifies whether 3- or 7-bit sequence numbers are to be used.

- If the other side accepts this request→ transmits an unnumbered acknowledged (UA) frame
- If the request is rejected → sends a disconnected mode (DM) frame

HDLC operation: Data transfer (1)

- After initialization is done → data transfer
- Both sides may begin to send user data in I-frames → starting with sequence number 0.

- N(S): sending sequence number
- N(R): sequence number of expected I-frame
- N(S) and N(R) numbered modulo 8 or 128

  - depending on whether 3- or 7-bit sequence numbers are used

HDLC operation: Data transfer (2)

- S-frames used for flow control and error control.

1. Receive Ready (RR): acknowledges the last received I-frame by indicating the next I-frame expected.

   - RR used when there is no data traffic

2. Receive Not Ready (RNR): acknowledges an I-frame, as with RR, but also asks to suspend transmission of I-frames

   - When again ready, sends an RR.

3. REJ: initiates the go-back-N ARQ

   - indicates retransmission of all I-frames beginning with number N(R)

4. Selective reject (SREJ): used to request retransmission of just a single frame

HDLC operation: Disconnect

- Either HDLC module can initiate a disconnect
- HDLC issues a disconnect by sending a disconnect (DISC) frame

- If remote entity accepts the disconnect → replies with a UA

  - Informs its layer 3 user that the connection has been terminated.

  - Any outstanding unacknowledged I-frames may be lost, and their recovery is the responsibility of higher layers

Example: Link setup and disconnect

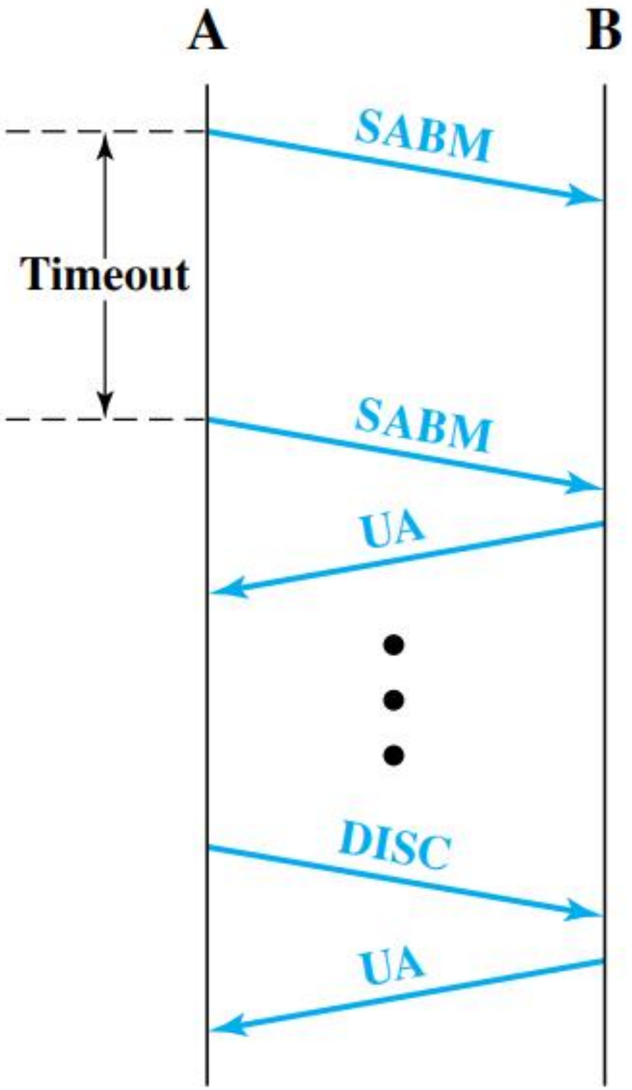
- A issues SABM
- B responses with UA, if accepts
- B responses with DM, if rejects
- If A times out → repeat the process for a maximum number of times

  - Failure → intervention of higher layers

- Disconnecting:

  - A issues DISC

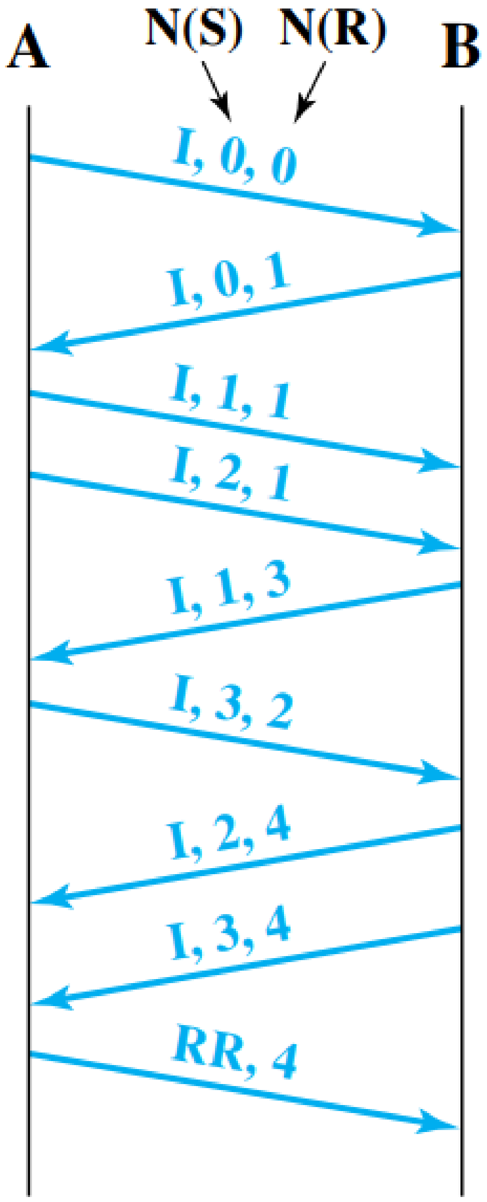
  - B responses with UA



(a) Link setup and disconnect

Example: Two-way data exchange

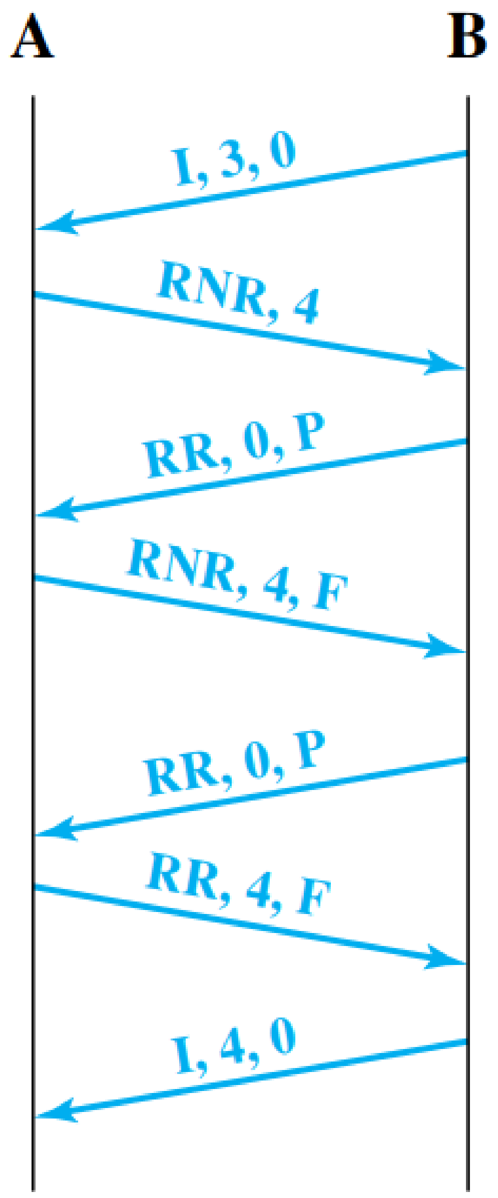
- Includes both I-frames and S-frames
- If I-frames are sent in a row with no incoming data, N(R) field is simply repeated
- If I-frames are received in a row with no outgoing frames, N(R) in the next outgoing frame must reflect the cumulative activity



(b) Two-way data exchange

Example: Busy condition

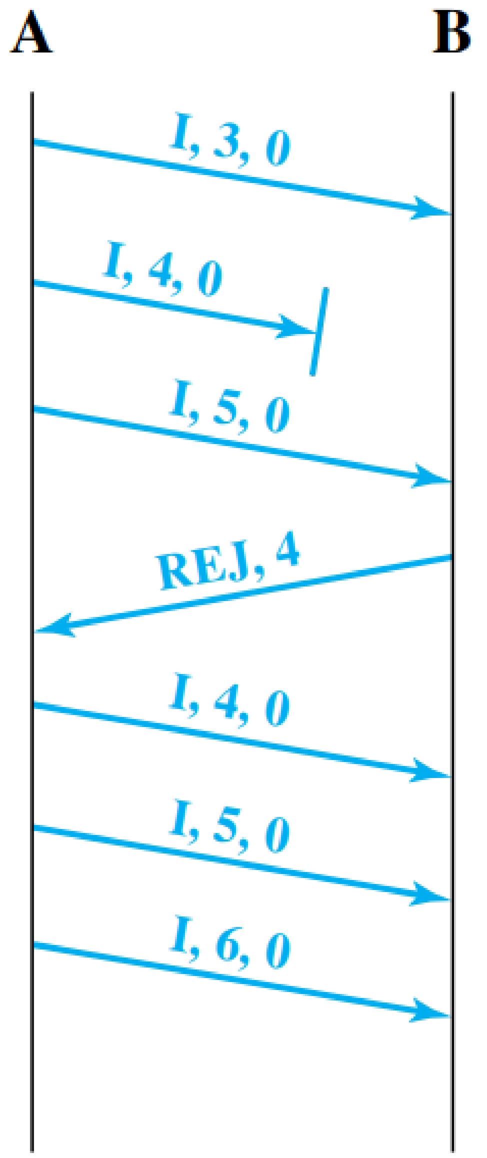
- Receiver's buffer fills up → it must halt the incoming flow of I-frames
- A issues an RNR command
- B halts transmission of I-frames
- B poll A at some periodic interval by sending an RR with the P bit set
- B responds with either an RR or an RNR
- With RR, I-frame transmission from B can resume



(c) Busy condition

Example: Reject recovery

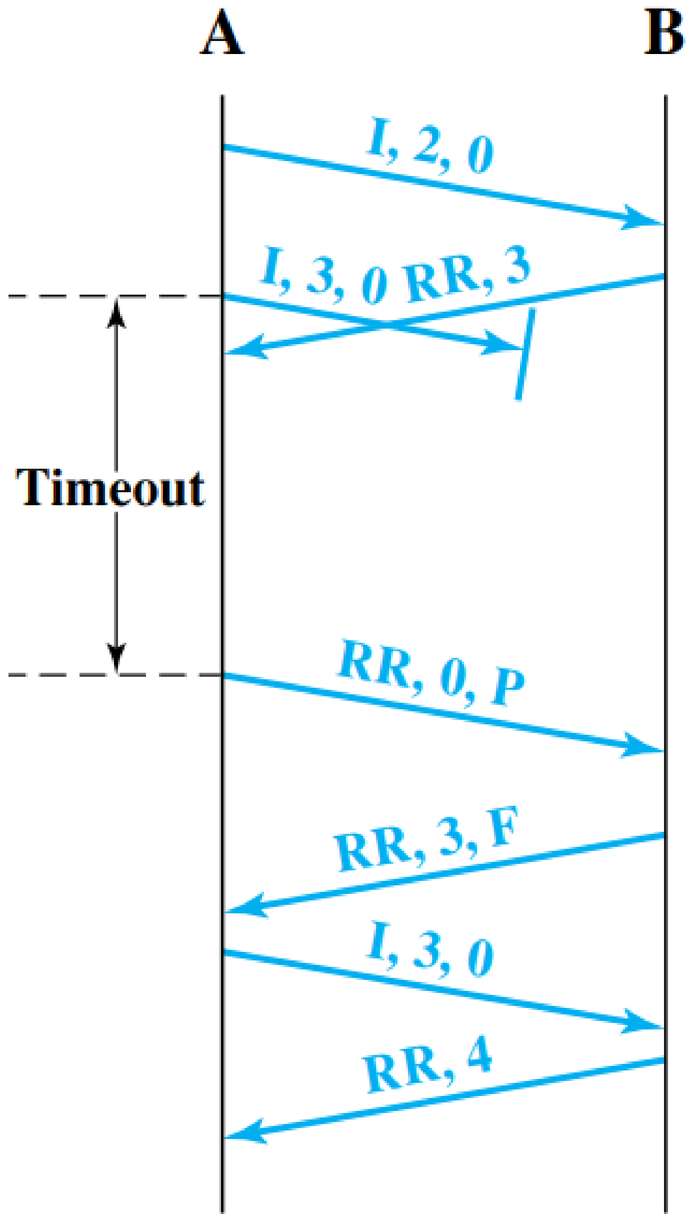
- A transmits I-frames numbered 3, 4, and 5
- Number 4 suffers an error and is lost
- B discards this frame 5 because it is out of order and sends an REJ with an N(R) of 4
- A initiates retransmission of I-frames previously sent, beginning with frame 4
- A may continue to send additional frames after the retransmitted frames



(d) Reject recovery

Example: Timeout recovery

- A transmits I-frame number 3 , with error
- B detects the error and discards it
- B cannot send an REJ, because there is no way to know if this was an I-frame
- A starts a timer
- Timer expires, A initiates recovery action
- A polls B with an RR command with the P bit set
- B sends an RR with N(R) field equal to the expected sequence number, here frame 3



(e) Timeout recovery