

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**.*



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.



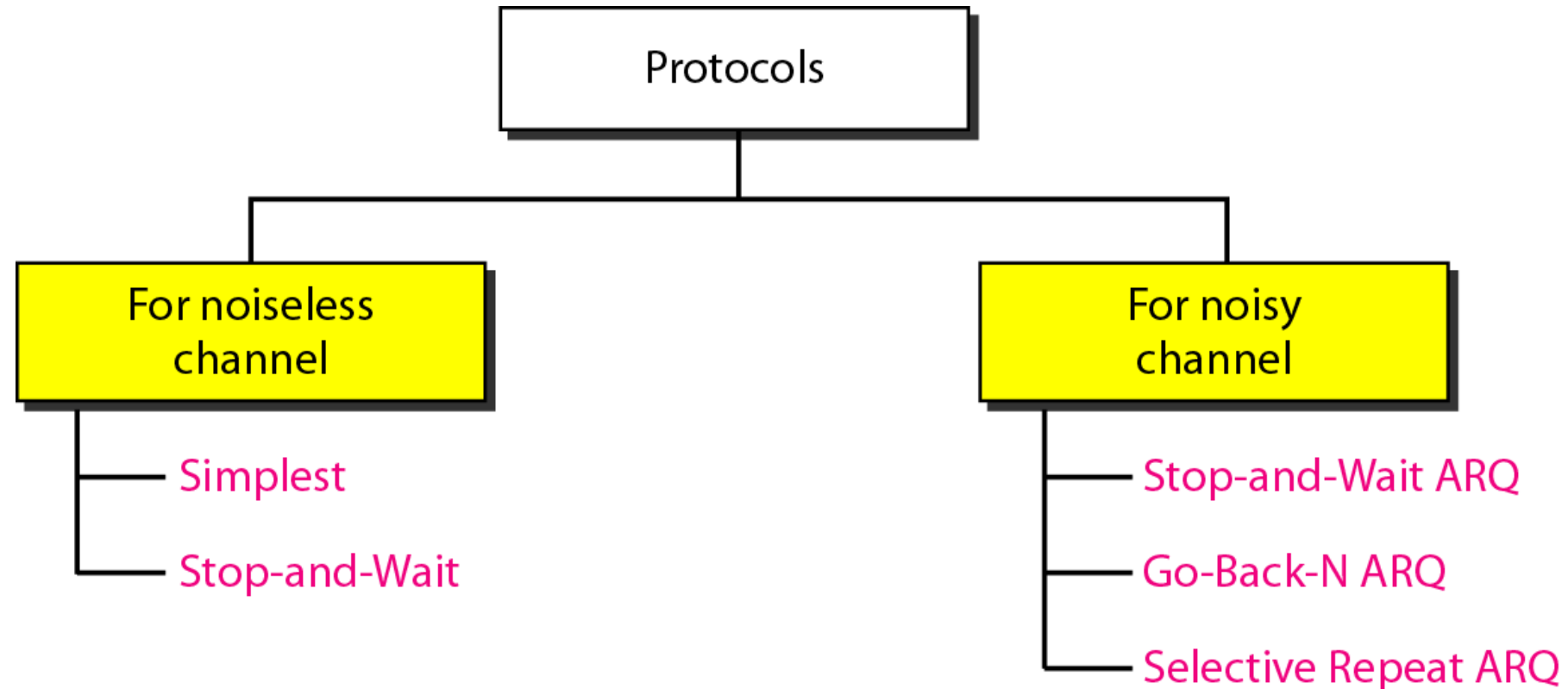
Note

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

PROTOCOLS

Now let us see how the data link layer can combine framing, flow control, and error control to achieve the delivery of data from one node to another. The protocols are normally implemented in software by using one of the common programming languages.

Figure *Taxonomy of protocols*



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.

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

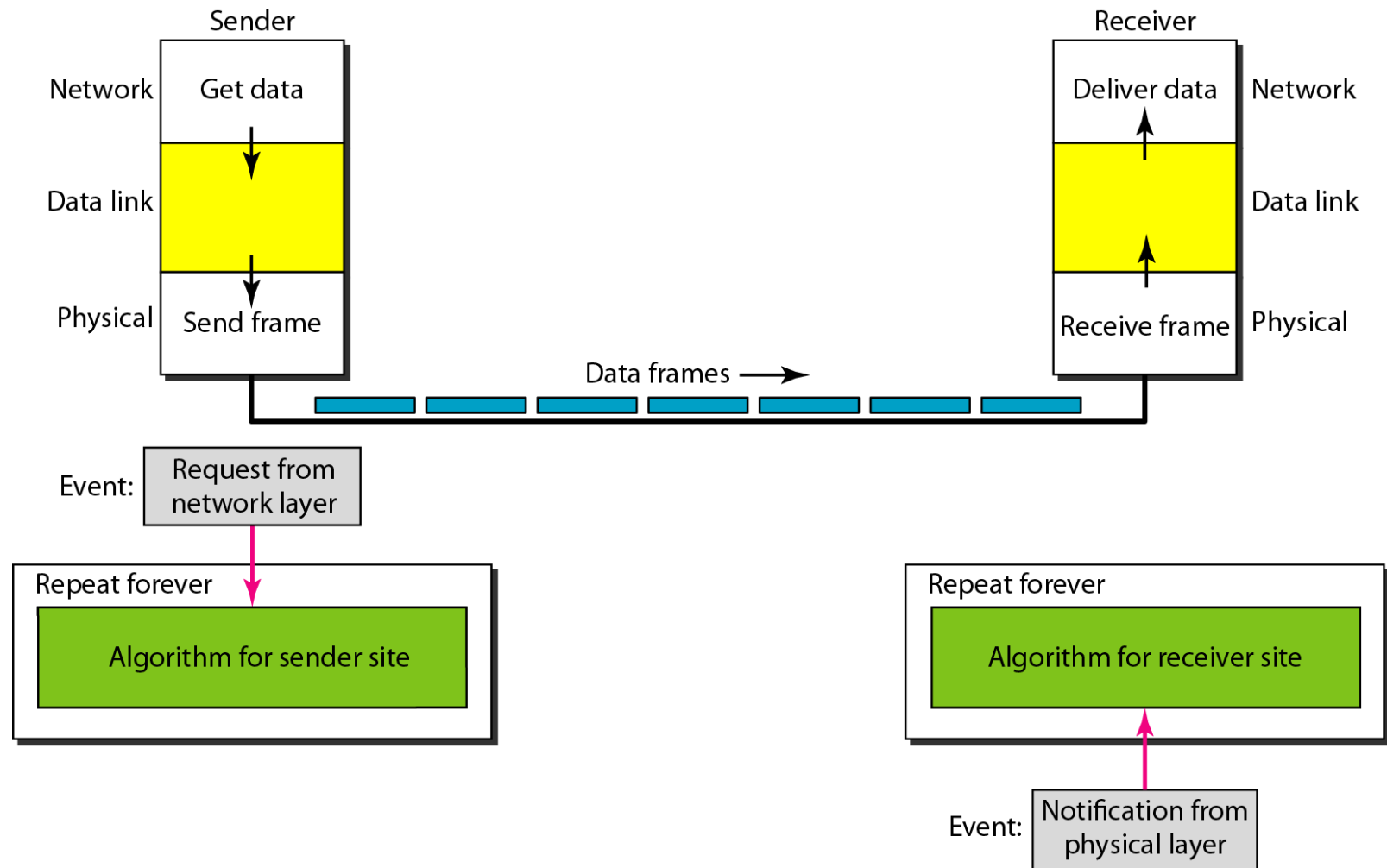


Figure 11.7 shows an example of communication using this protocol. It is very simple. The sender sends a sequence of frames without even thinking about the receiver. To send three frames, three events occur at the sender site and three events at the receiver site. Note that the data frames are shown by tilted boxes; the height of the box defines the transmission time difference between the first bit and the last bit in the frame.

Figure *Flow diagram for Example*

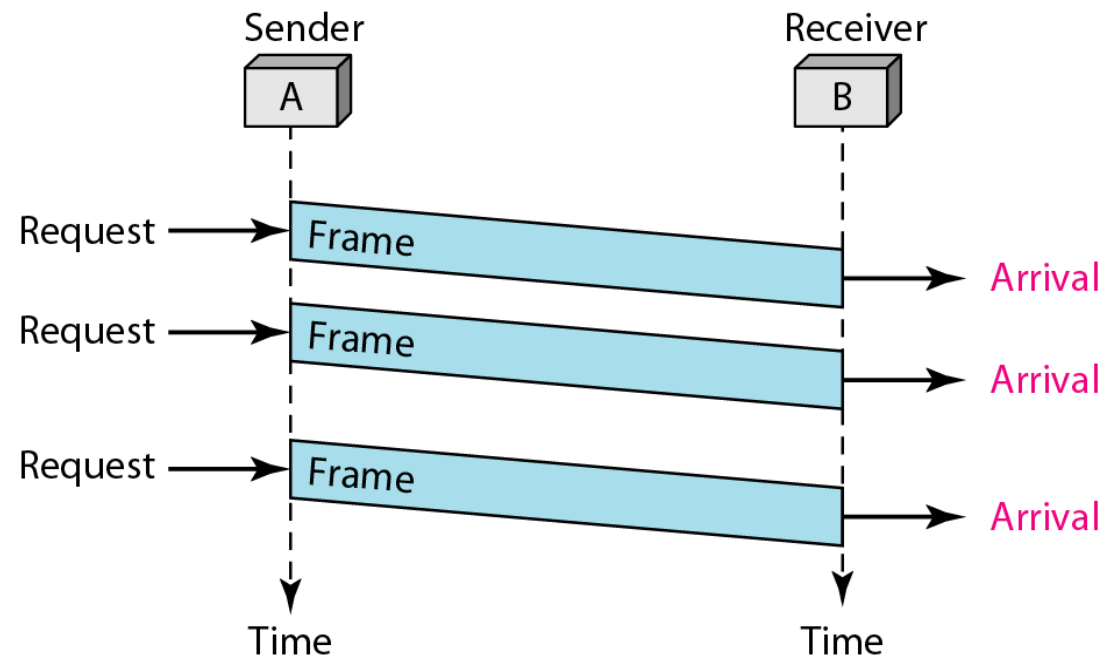


Figure *Design of Stop-and-Wait Protocol*

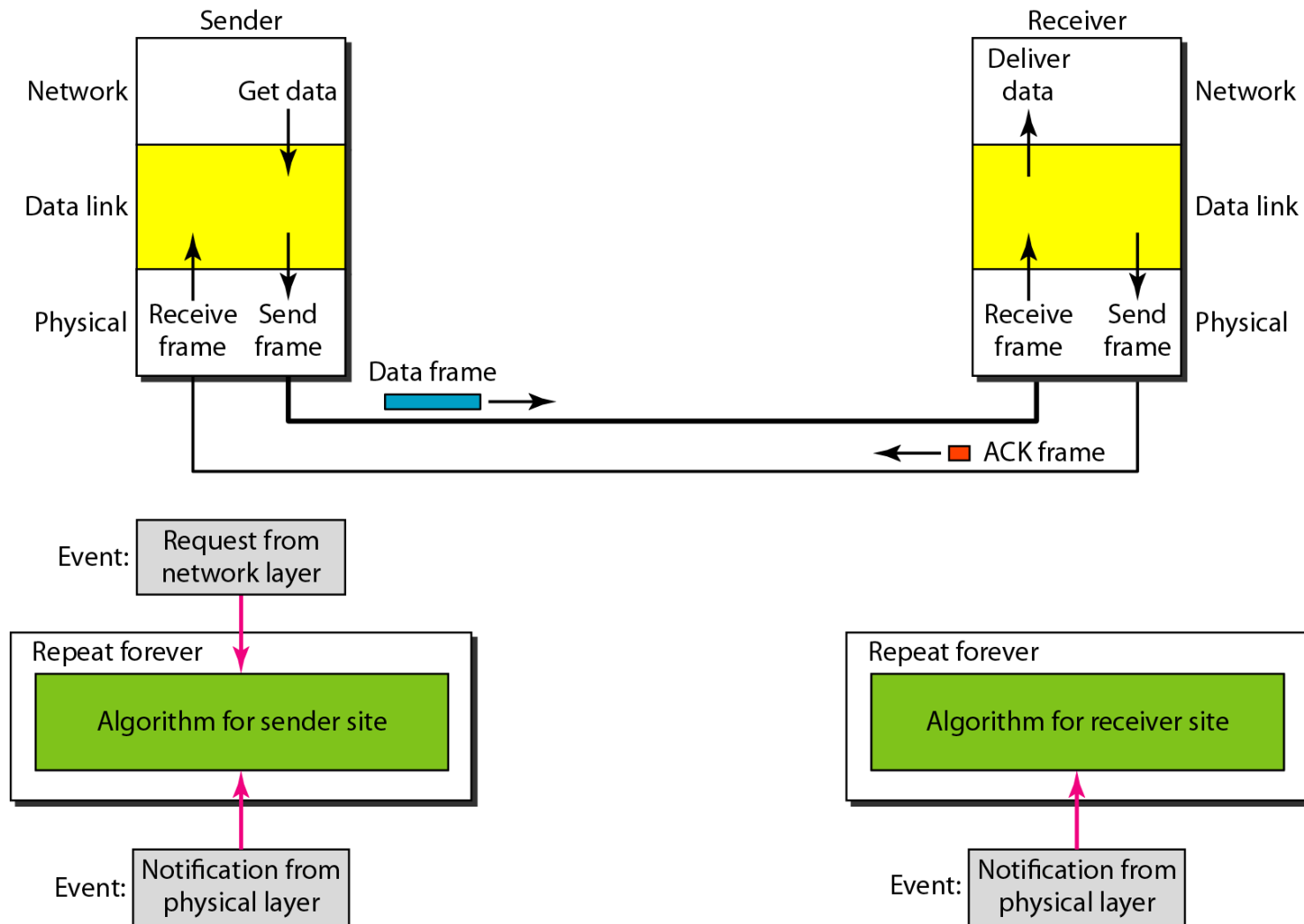
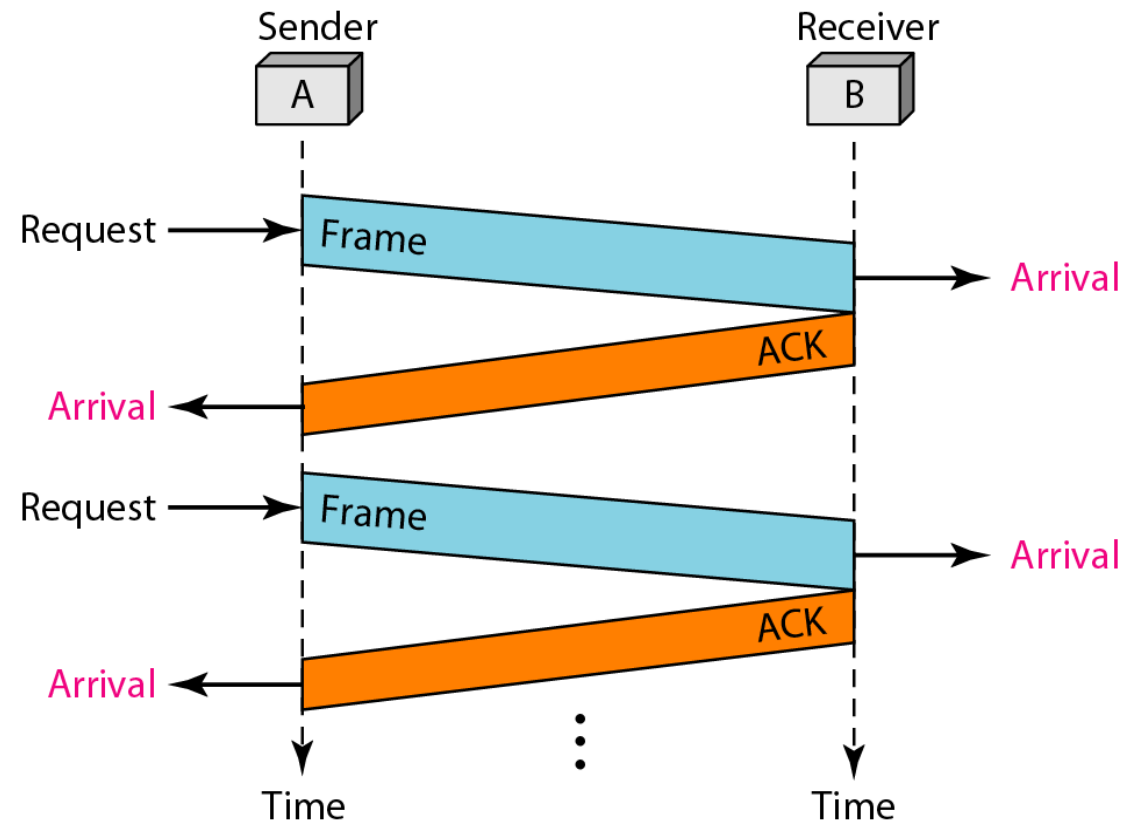
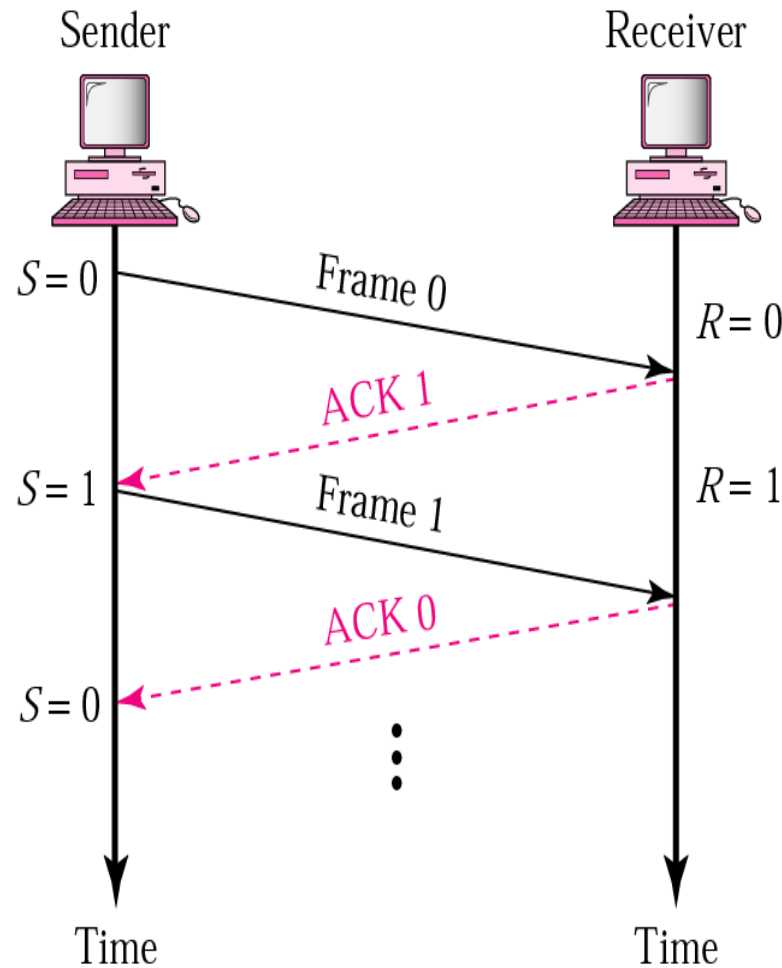


Figure shows an example of communication using this protocol. It is still very simple. The sender sends one frame and waits for feedback from the receiver. When the ACK arrives, the sender sends the next frame.

Figure *Flow diagram for Example*

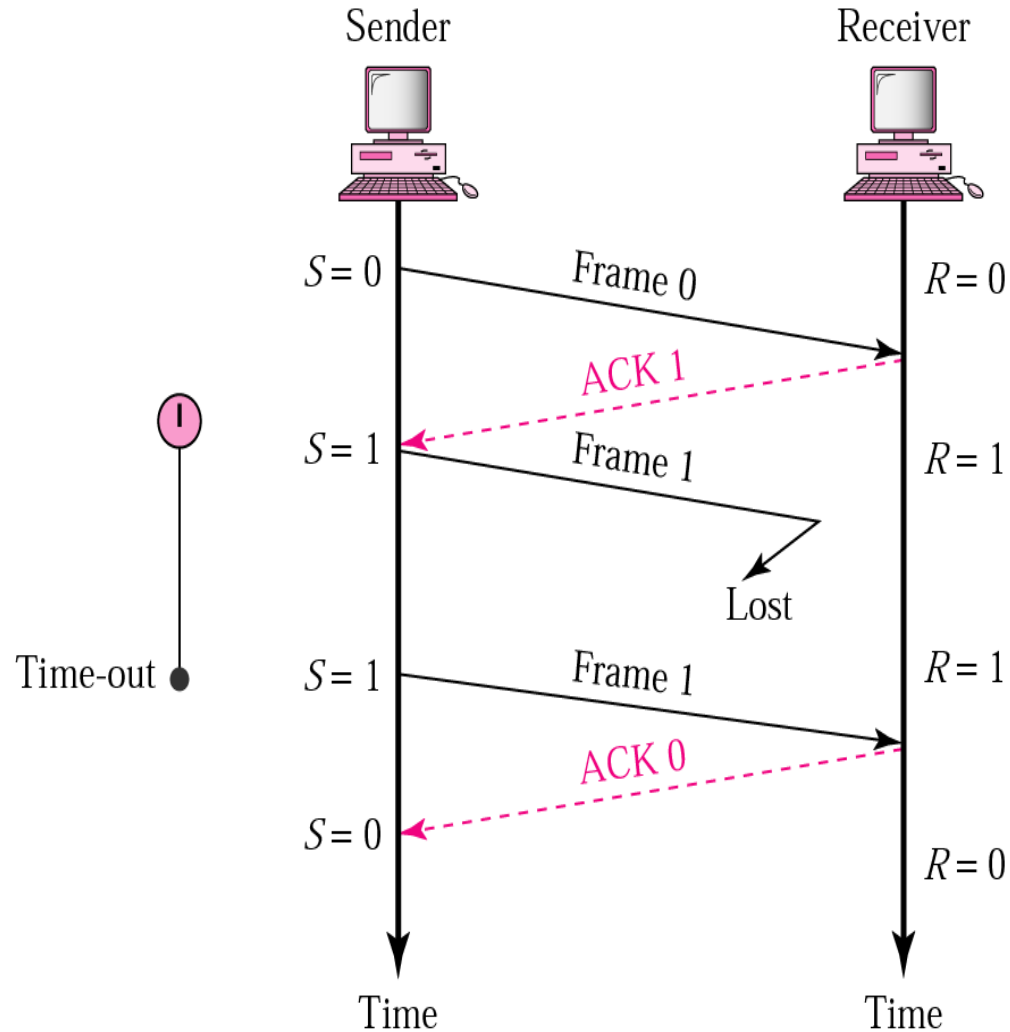


Stop-and-Wait



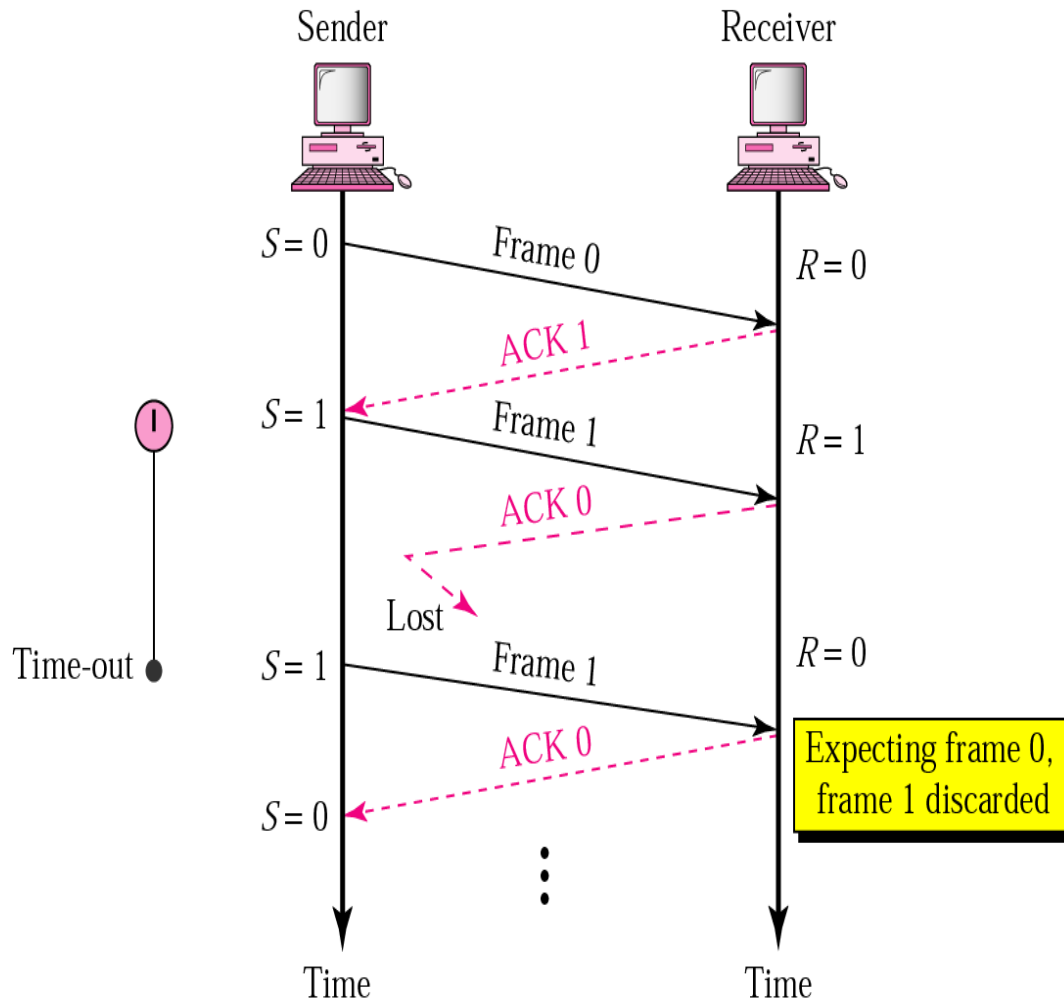
- Sender keeps a copy of the last frame until it receives an acknowledgement.
- For identification, both data frames and acknowledgements (ACK) frames are numbered alternatively 0 and 1.
- Sender has a control variable (S) that holds the number of the recently sent frame. (0 or 1)
- Receiver has a control variable R that holds the number of the next frame expected (0 or 1).
- Sender starts a timer when it sends a frame. If an ACK is not received within a allocated time period, the sender assumes that the frame was lost or damaged and resends it
- Receiver send only positive ACK if the frame is intact.
- ACK number always defines the number of the next expected frame

Stop-and-Wait ARQ, lost frame



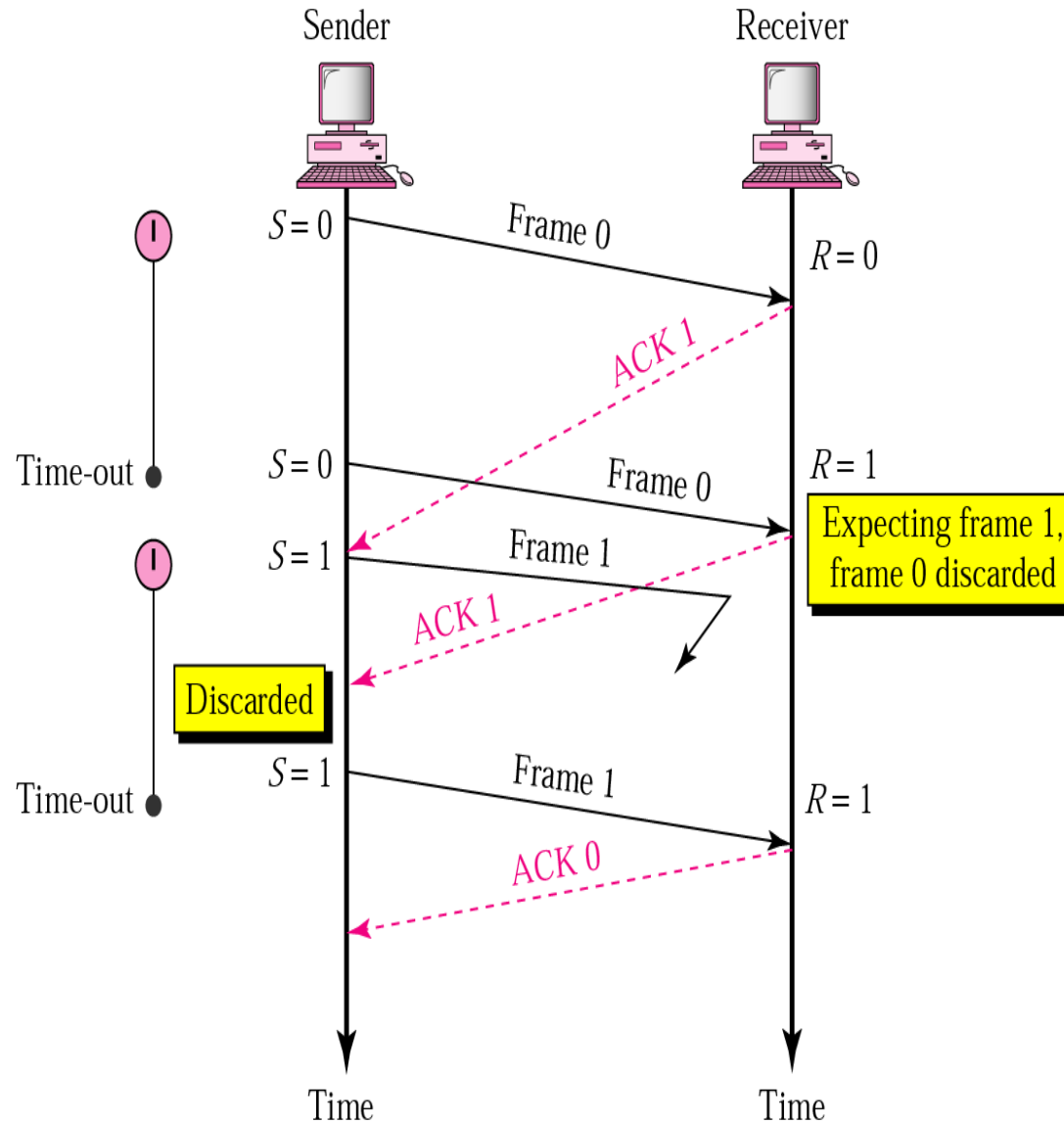
- When a receiver receives a damaged frame, it discards it and keeps its value of R .
- After the timer at the sender expires, another copy of frame 1 is sent.

Stop-and-wait, lost ACK frame



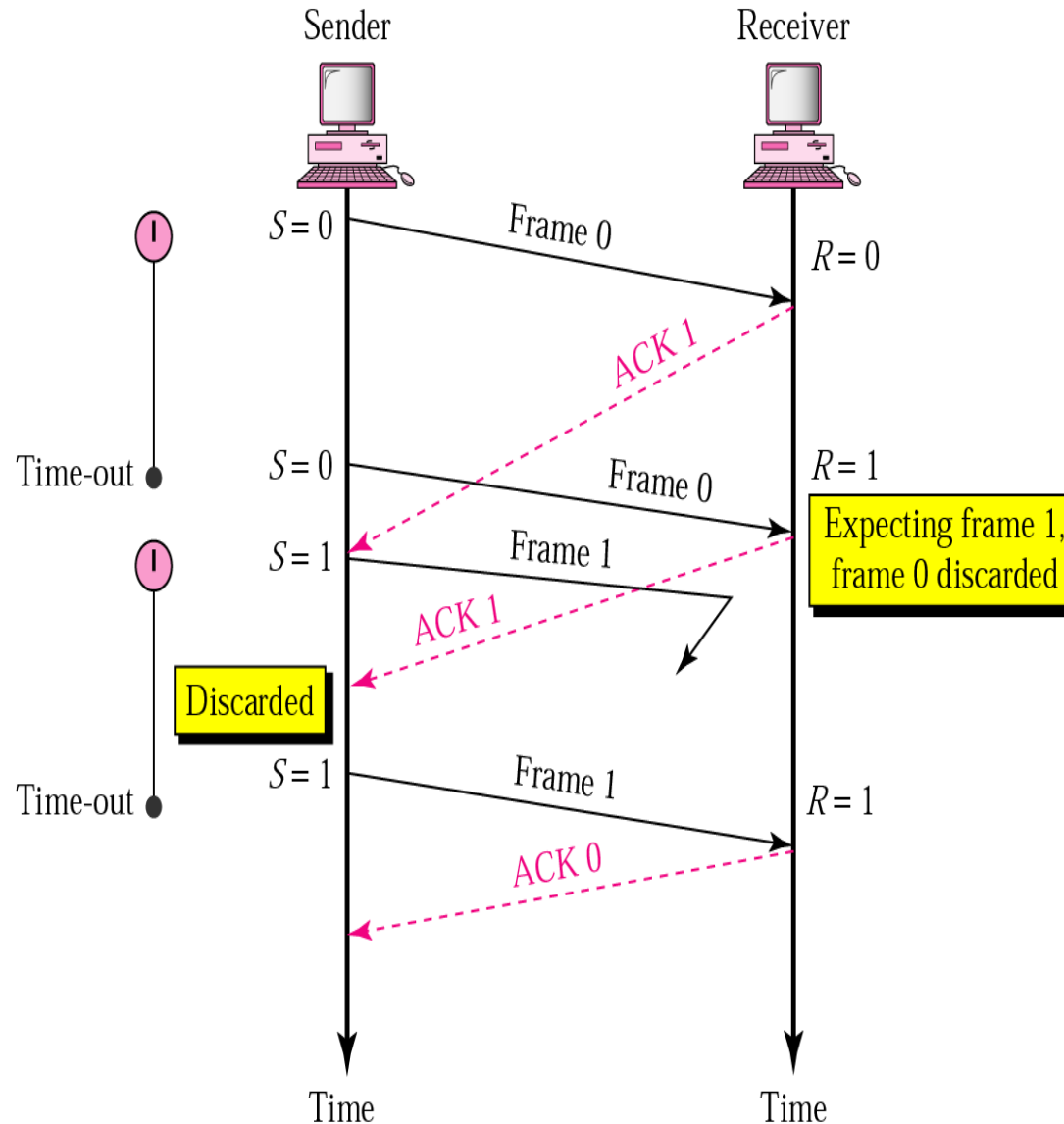
- If the sender receives a damaged ACK, it discards it.
- When the timer of the sender expires, the sender retransmits frame 1.
- Receiver has already received frame 1 and expecting to receive frame 0 ($R=0$). Therefore it discards the second copy of frame 1.

Stop-and-wait, delayed ACK frame



- The ACK can be delayed at the receiver or due to some problem
- It is received after the timer for frame 0 has expired.
- Sender retransmitted a copy of frame 0. However, $R=1$ means receiver expects to see frame 1. Receiver discards the duplicate frame 0.
- Sender receives 2 ACKs, it discards the second ACK.

Stop-and-wait, delayed ACK frame



- The ACK can be delayed at the receiver or due to some problem
- It is received after the timer for frame 0 has expired.
- Sender retransmitted a copy of frame 0. However, $R=1$ means receiver expects to see frame 1. Receiver discards the duplicate frame 0.
- Sender receives 2 ACKs, it discards the second ACK.

Disadvantage of Stop-and-Wait

- In stop-and-wait, at any point in time, there is only one frame that is sent and waiting to be acknowledged.
- This is not a good use of transmission medium.
- To improve efficiency, multiple frames should be in transition while waiting for ACK.
- Two protocols use the above concept,
 - **Go-Back-N ARQ**
 - **Selective Repeat ARQ**

Problems

1. If the bandwidth of the line is 1.5 Mbps, RTT is 45 msec and packet size is 1 KB, then find the link utilization in stop and wait.
2. A channel has a bit rate of 4 Kbps and one way propagation delay of 20 msec. The channel uses stop and wait protocol. The transmission time of the acknowledgement frame is negligible. To get a channel efficiency of at least 50%, the minimum frame size should be:

-
3. Consider a MAN with average source and destination 20 Km apart and one way delay of 100 μ sec. At what data rate does the round trip delay equals the transmission delay for a 1 KB packet?
4. If the packet size is 1 KB and propagation time is 15 msec, the channel capacity is 10^9 b/sec, then find the transmission time and utilization of sender in stop and wait protocol.

Go-Back-N ARQ

We can send up to W frames before worrying about ACKs.

We keep a copy of these frames until the ACKs arrive.

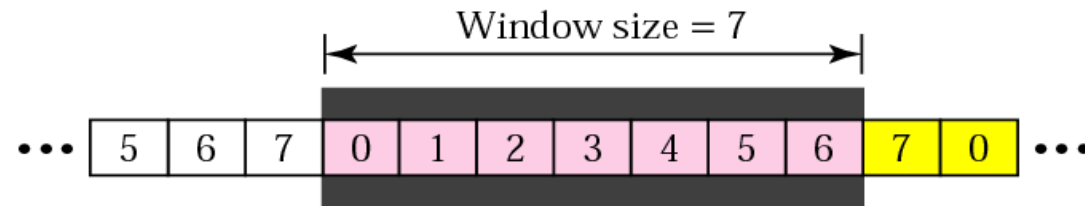
This procedure requires additional features to be added to Stop-and-Wait ARQ.

Sequence Numbers

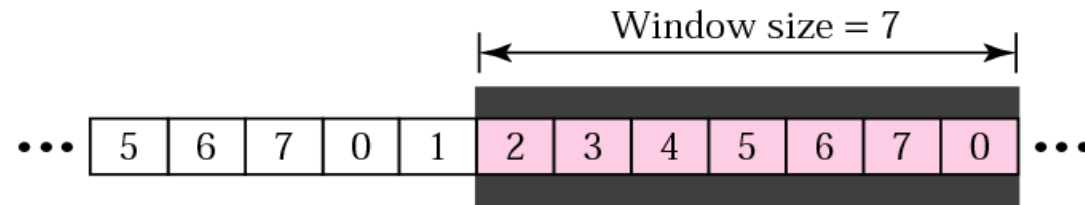
- Frames from a sender are numbered sequentially.
- We need to set a limit since we need to include the sequence number of each frame in the header.
- If the header of the frame allows m bits for sequence number, the sequence numbers range from 0 to $2^m - 1$. for $m = 3$, sequence numbers are: 1, 2, 3, 4, 5, 6, 7.
- We can repeat the sequence number.
- Sequence numbers are:
0, 1, 2, 3, 4, 5, 6, 7, 0, 1, 2, 3, 4, 5, 6, 7, 0, 1, ...

Sender Sliding Window

- At the sending site, to hold the outstanding frames until they are acknowledged, we use the concept of a window.
- The size of the window is at most $2^m - 1$ where m is the number of bits for the sequence number.
- Size of the window can be variable, e.g. TCP.
- The window slides to include new unsent frames when the correct ACKs are received



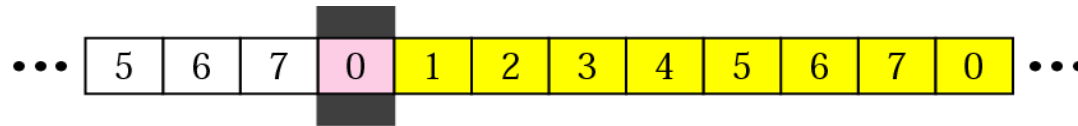
a. Before sliding



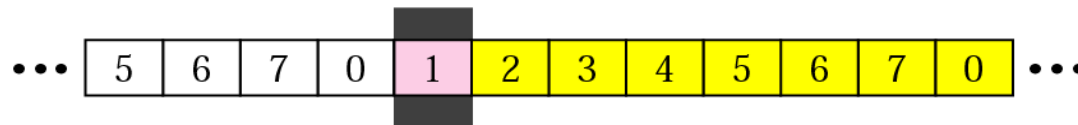
b. After sliding two frames

Receiver Sliding Window

- Size of the window at the receiving site is always 1 in this protocol.
- Receiver is always looking for a specific frame to arrive in a specific order.
- Any frame arriving out of order is discarded and needs to be resent.
- Receiver window slides as shown in fig. Receiver is waiting for frame 0 in part a.



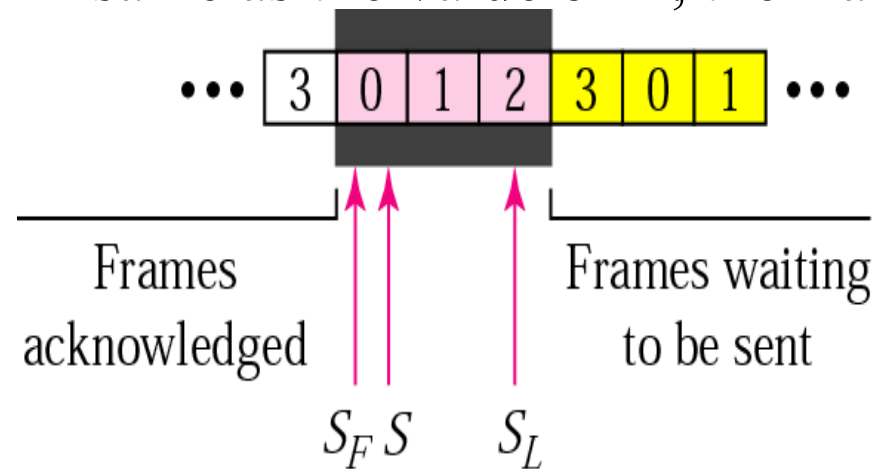
a. Before sliding



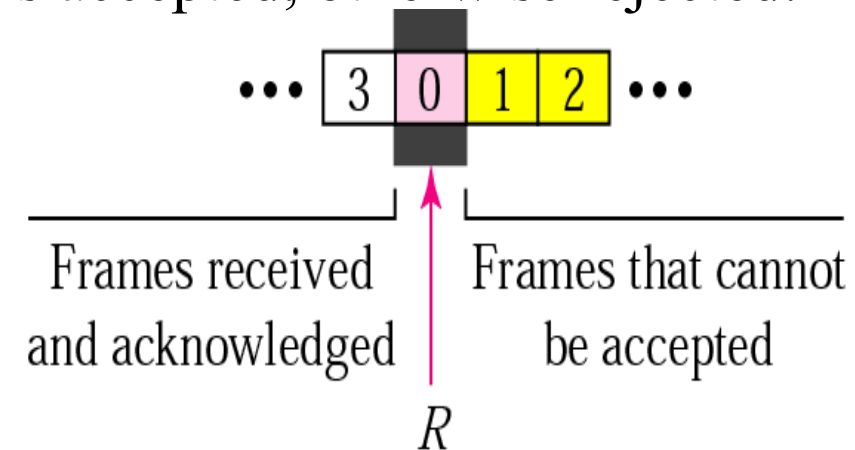
b. After sliding

Control Variables

- Sender has 3 variables: S , S_F , and S_L
- S holds the sequence number of recently sent frame
- S_F holds the sequence number of the first frame
- S_L holds the sequence number of the last frame
- Receiver only has the one variable, R , that holds the sequence number of the frame it expects to receive. If the seq. no. is the same as the value of R , the frame is accepted, otherwise rejected.



a. Sender window



b. Receiver window

Acknowledgement

Receiver sends positive ACK if a frame arrived safe and in order.

If the frames are damaged/out of order, receiver is silent and discard all subsequent frames until it receives the one it is expecting.

The silence of the receiver causes the timer of the unacknowledged frame to expire.

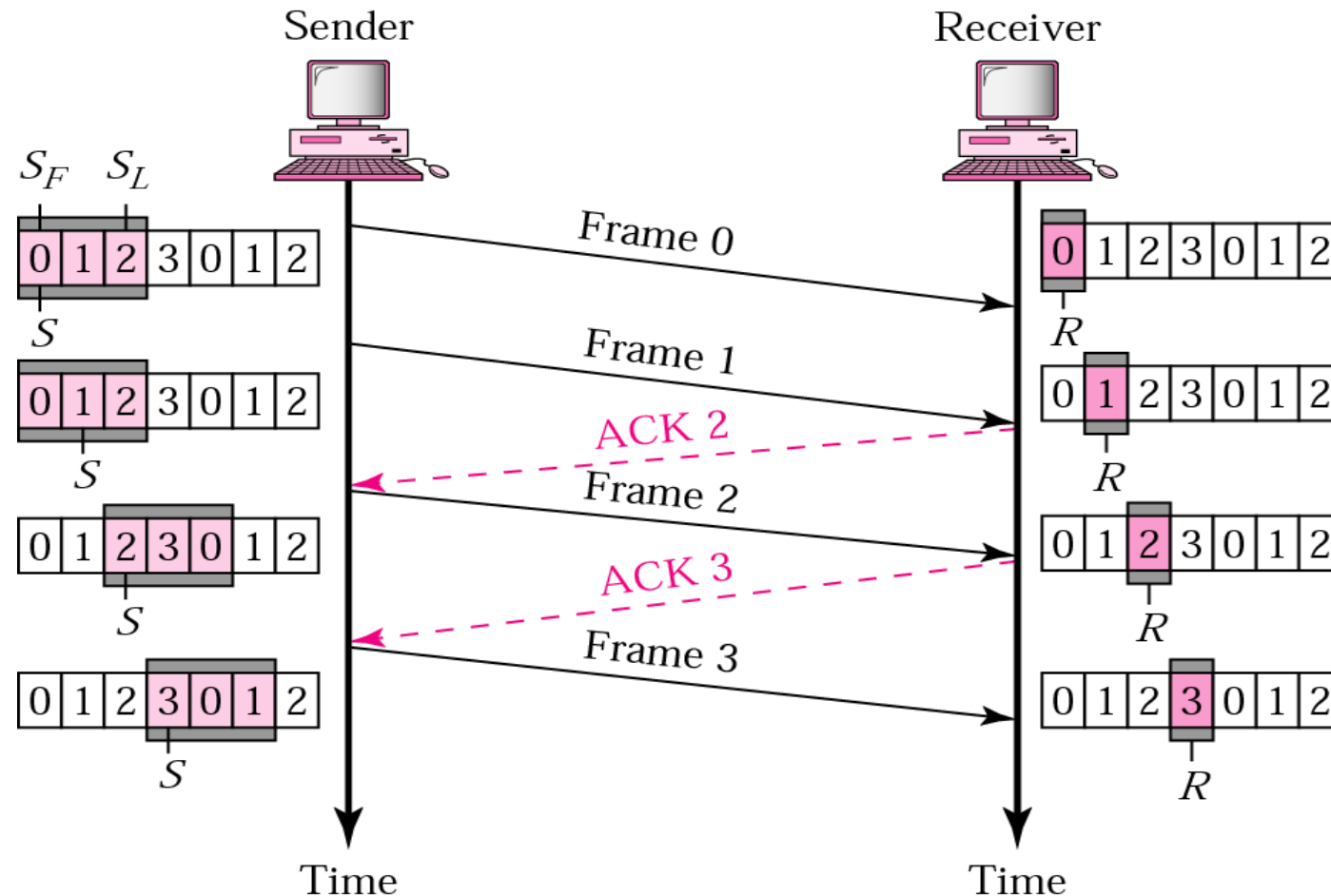
Then the sender resends all frames, beginning with the one with the expired timer.

For example, suppose the sender has sent frame 6, but the timer for frame 3 expires (i.e. frame 3 has not been acknowledged), then the sender goes back and sends frames 3, 4, 5, 6 again. Thus it is called Go-Back-N-ARQ

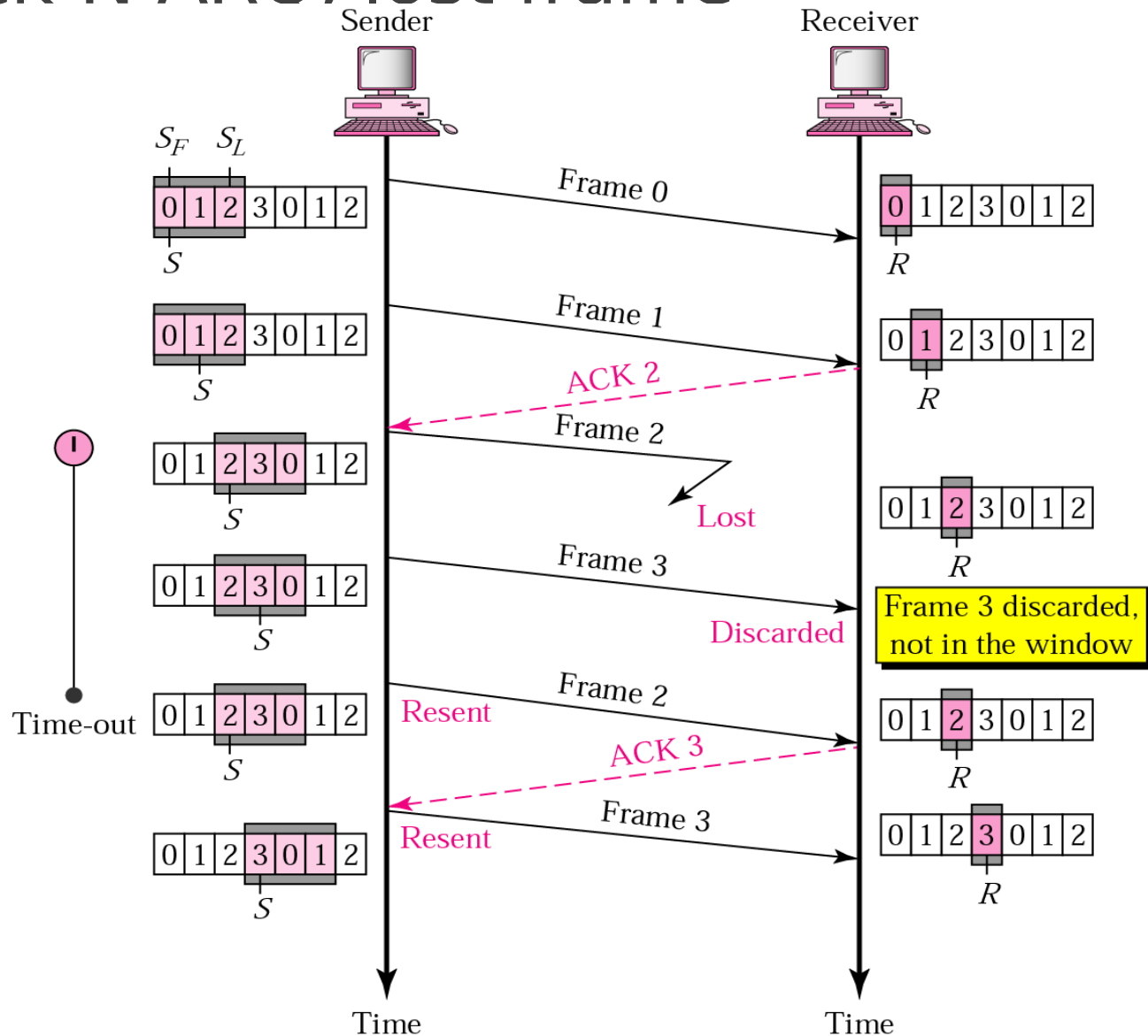
The receiver does not have to acknowledge each frame received, it can send one cumulative ACK for several frames.

Go-Back-N ARQ, normal operation

- The sender keeps track of the outstanding frames and updates the variables and windows as the ACKs arrive.



Go-Back-N ARO. lost frame



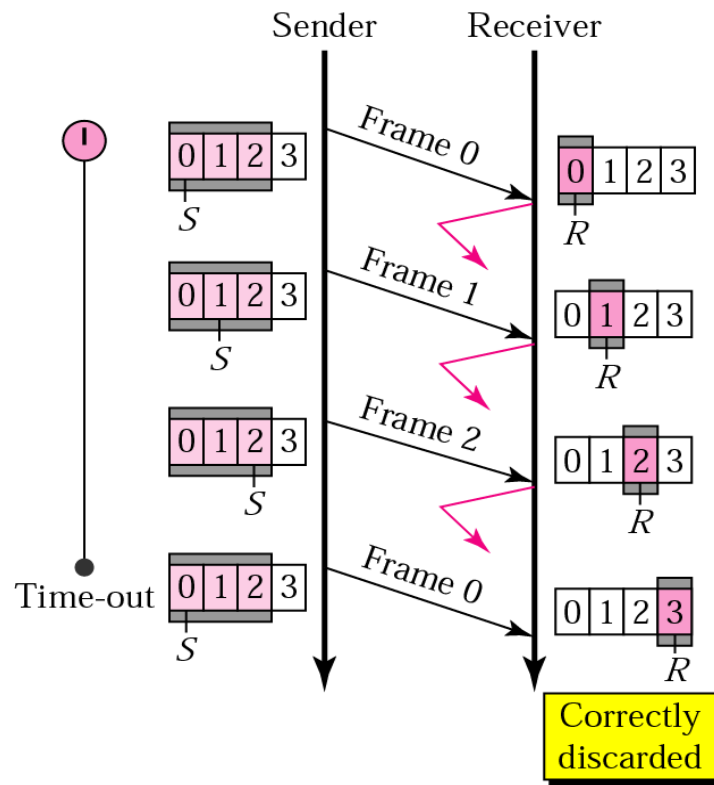
- Frame 2 is lost
- When the receiver receives frame 3, it discards frame 3 as it is expecting frame 2 (according to window).
- After the timer for frame 2 expires at the sender site, the sender sends frame 2 and 3. (go back to 2)

Go-Back-N ARQ, damaged/lost/delayed ACK

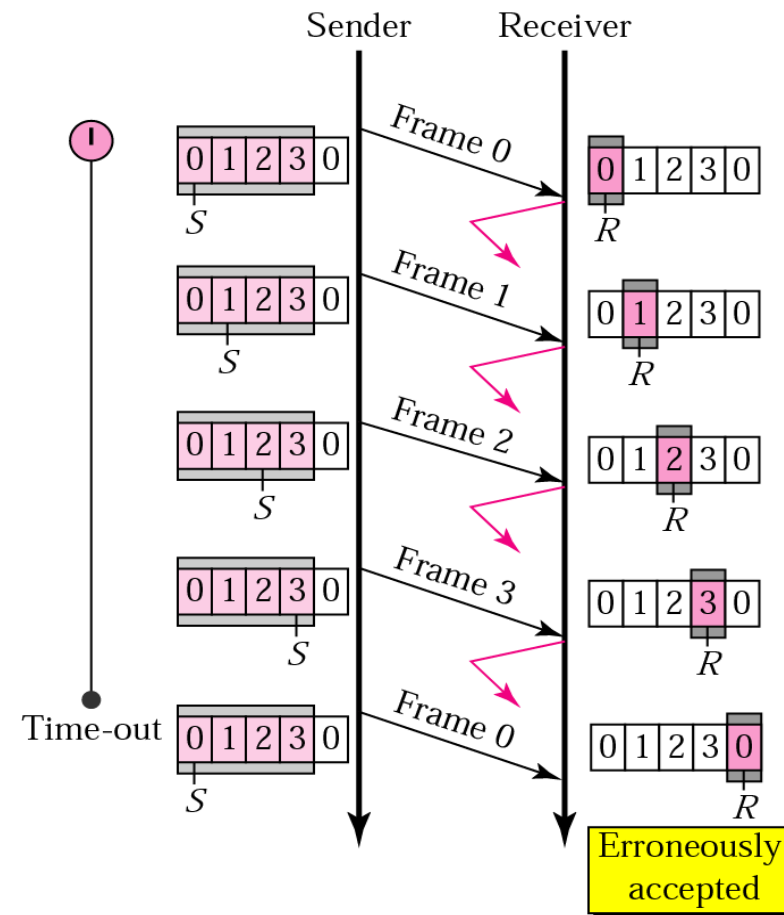
- If an ACK is damaged/lost, we can have two situations:
- If the next ACK arrives before the expiration of any timer, there is no need for retransmission of frames because ACKs are cumulative in this protocol.
- If ACK1, ACK2, and ACK3 are lost, ACK4 covers them if it arrives before the timer expires.
- If ACK4 arrives after time-out, the last frame and all the frames after that are resent.
- Receiver never resends an ACK.
- A delayed ACK also triggers the resending of frames

Go-Back-N ARQ, sender window size

- Size of the sender window must be less than 2^m . Size of the receiver is always 1. If $m = 2$, window size = $2^m - 1 = 3$.
- Fig compares a window size of 3 and 4.



a. Window size $< 2^m$

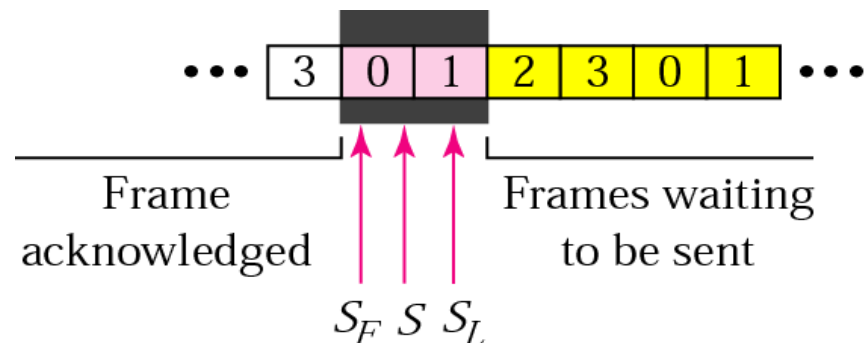


b. Window size $= 2^m$

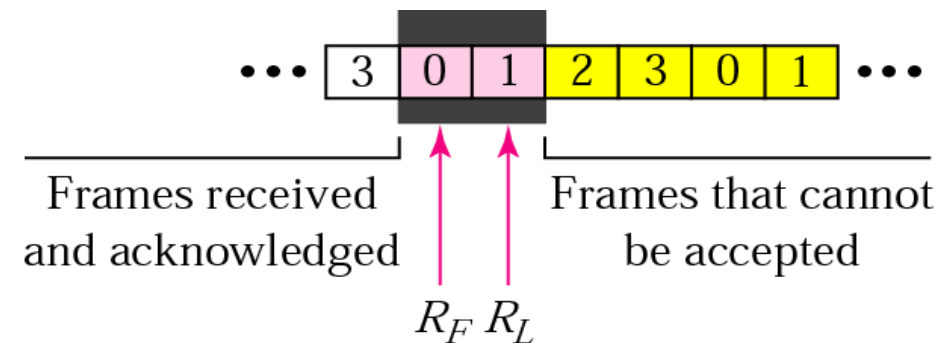
Accepts
as the
1st
frame in
the next
cycle-an
error

Selective Repeat ARQ, sender and receiver windows

- Go-Back-N ARQ simplifies the process at the receiver site. Receiver only keeps track of only one variable, and there is no need to buffer out-of-order frames, they are simply discarded.
- However, Go-Back-N ARQ protocol is inefficient for noisy link. It bandwidth inefficient and slows down the transmission.
- In Selective Repeat ARQ, only the damaged frame is resent. More bandwidth efficient but more complex processing at receiver.
- It defines a negative ACK (NAK) to report the sequence number of a damaged frame before the timer expires.

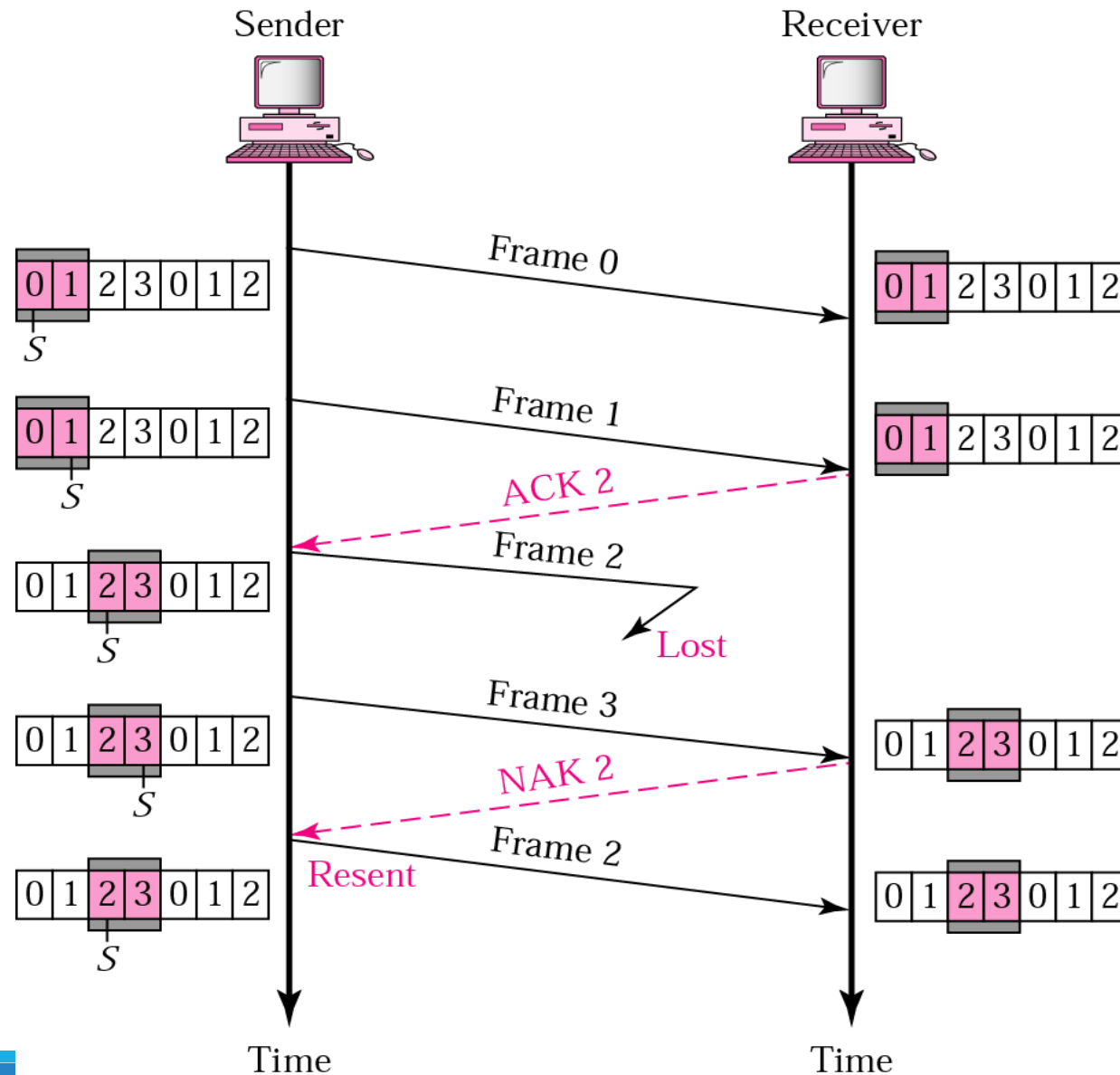


a. Sender window



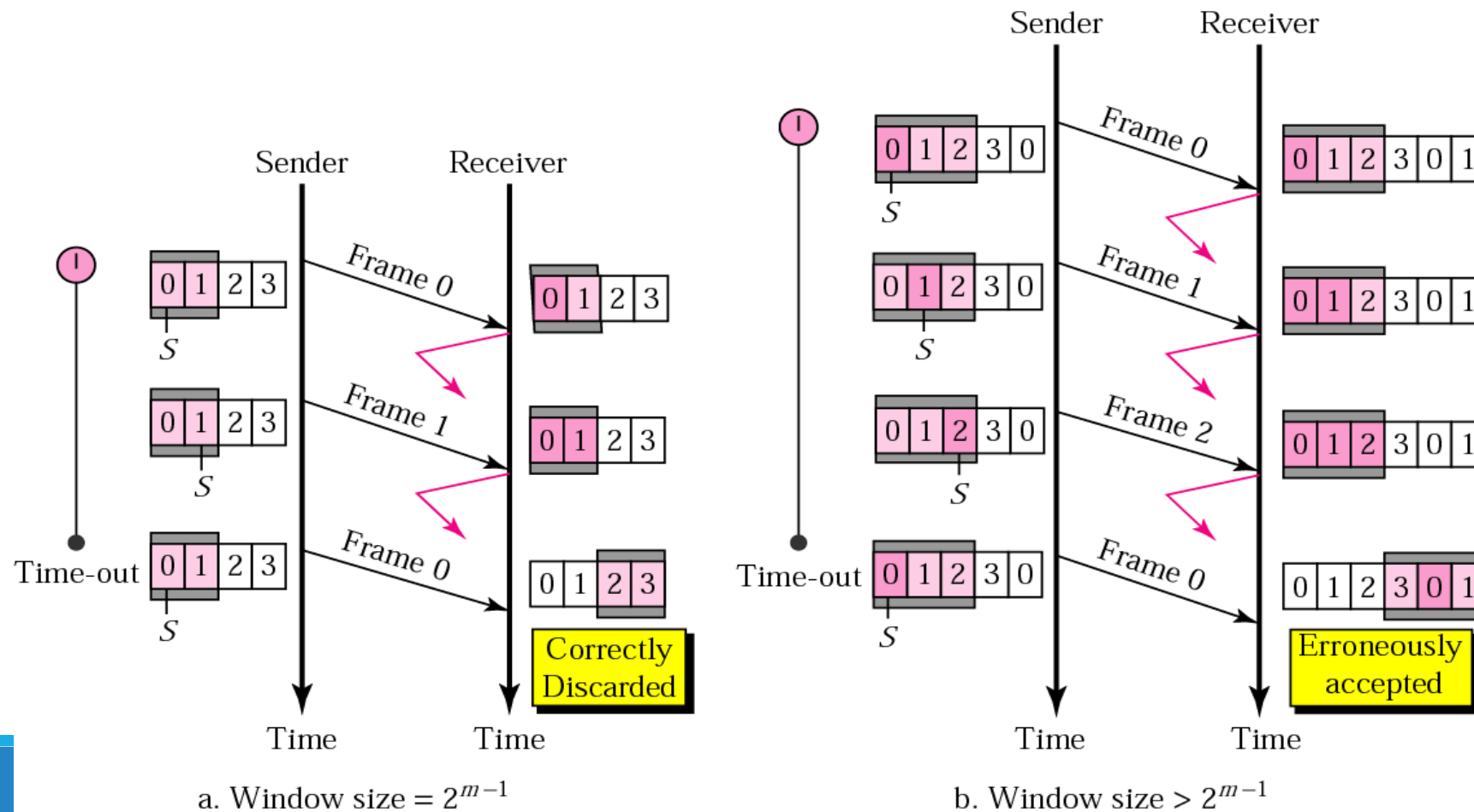
b. Receiver window

Selective Repeat ARQ, lost frame



- Frames 0 and 1 are accepted when received because they are in the range specified by the receiver window. Same for frame 3.
- Receiver sends a NAK2 to show that frame 2 has not been received and then sender resends only frame 2 and it is accepted as it is in the range of the window.

- Size of the sender and receiver windows must be at most one-half of 2^m . If $m = 2$, window size should be $2^m / 2 = 2$. Fig compares a window size of 2 with a window size of 3. Window size is 3 and all ACKs are lost, sender sends duplicate of frame 0, window of the receiver expect to receive frame 0 (part of the window), so accepts frame 0, as the 1st frame of the next cycle – an **error**.



Problems

In SR protocol, suppose frames through 0 to 4 have been transmitted. Now, imagine that 0 times out, 5 (a new frame) is transmitted, 1 times out, 2 times out and 6 (another new frame) is transmitted.

At this point, what will be the outstanding packets in sender's window?