# Module 3

# Data Link control

# Lesson

# 3

# Flow Control and Error Control

Version 2 CSE IIT, Kharagpur

## Special Instructional Objectives:

On completion, the student will be able to:

- State the need for flow and error control
- Explain how Stop-and-wait flow control works
- Explain how Sliding-window protocol is used for flow control
- Explain how Stop-and-wait ARQ works
- Explain how Go-back-N ARQ works
- Explain how Selective-repeat ARQ works

## 3.3.1 Introduction

As we have mentioned earlier, for reliable and efficient data communication a great deal of coordination is necessary between at least two machines. Some of these are necessary because of the following constraints:

- Both sender and receiver have limited speed
- Both sender and receiver have limited memory

It is necessary to satisfy the following requirements:

- A fast sender should not overwhelm a slow receiver, which must perform a certain amount of processing before passing the data on to the higher-level software.
- If error occur during transmission, it is necessary to devise mechanism to correct it

The most important functions of Data Link layer to satisfy the above requirements are **error control** and **flow control**. Collectively, these functions are known as **data link control**, as discussed in this lesson.

**Flow Control** is a technique so that transmitter and receiver with different speed characteristics can communicate with each other. Flow control ensures that a transmitting station, such as a server with higher processing capability, does not overwhelm a receiving station, such as a desktop system, with lesser processing capability. This is where there is an orderly flow of transmitted data between the source and the destination.

**Error Control** involves both error detection and error correction. It is necessary because errors are inevitable in data communication, in spite of the use of better equipment and reliable transmission media based on the current technology. In the preceding lesson we have already discussed how errors can be detected. In this lesson we shall discuss how error control is performed based on retransmission of the corrupted data. When an error is detected, the receiver can have the specified frame retransmitted by the sender. This process is commonly known as **Automatic Repeat Request (ARQ).** For example, Internet's Unreliable Delivery Model allows packets to be discarded if network resources are not available, and demands that ARQ protocols make provisions for retransmission.

## 3.3.2 Flow Control

Modern data networks are designed to support a diverse range of hosts and communication mediums. Consider a 933 MHz Pentium-based host transmitting data to a 90 MHz 80486/SX. Obviously, the Pentium will be able to drown the slower processor with data. Likewise, consider two hosts, each using an Ethernet LAN, but with the two Ethernets connected by a 56 Kbps modem link. If one host begins transmitting to the other at Ethernet speeds, the modem link will quickly become overwhelmed. In both cases, *flow control* is needed to pace the data transfer at an acceptable speed.

Flow Control is a set of procedures that tells the sender how much data it can transmit before it must wait for an acknowledgment from the receiver. The flow of data should not be allowed to overwhelm the receiver. Receiver should also be able to inform the transmitter before its limits (this limit may be amount of memory used to store the incoming data or the processing power at the receiver end) are reached and the sender must send fewer frames. Hence, **Flow control** refers to the set of procedures used to restrict the amount of data the transmitter can send before waiting for acknowledgment.

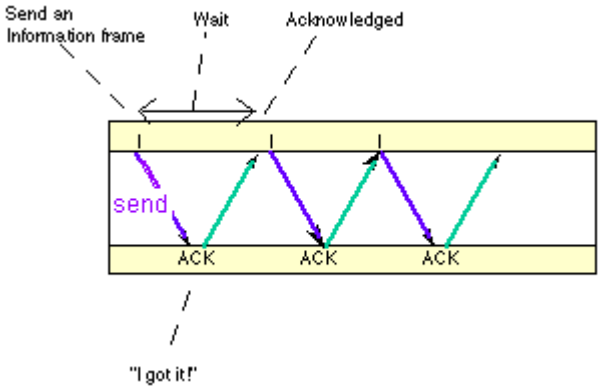
There are two methods developed for flow control namely **Stop-and-wait** and **Sliding-window**. Stop-and-wait is also known as Request/reply sometimes. Request/reply (Stop-and-wait) flow control requires each data packet to be acknowledged by the remote host before the next packet is sent. This is discussed in detail in the following subsection. **Sliding window** algorithms, used by TCP, permit multiple data packets to be in simultaneous transit, making more efficient use of network bandwidth as discussed in subsection 3.3.2.2.

### 3.3.2.1 Stop-and-Wait

This is the simplest form of flow control where a sender transmits a data frame. After receiving the frame, the receiver indicates its willingness to accept another frame by sending back an ACK frame acknowledging the frame just received. The sender must wait until it receives the ACK frame before sending the next data frame.This is sometimes referred to as *ping-pong* behavior, request/reply is simple to understand and easy to implement, but not very efficient. In LAN environment with fast links, this isn't much of a concern, but WAN links will spend most of their time idle, especially if several hops are required.

Figure 3.3.1 illustrates the operation of the stop-and-wait protocol. The blue arrows show the sequence of data frames being sent across the link from the sender (top to the receiver (bottom). The protocol relies on two-way transmission (full duplex or half duplex) to allow the receiver at the remote node to return frames acknowledging the successful transmission. The acknowledgements are shown in green in the diagram, and flow back to the original sender. A small processing delay may be introduced between reception of the last byte of a Data PDU and generation of the corresponding ACK.

Major drawback of Stop-and-Wait Flow Control is that only one frame can be in transmission at a time, this leads to inefficiency if propagation delay is much longer than the transmission delay.




Some protocols pretty much require stop-and-wait behavior. For example, Internet's Remote Procedure Call (RPC) Protocol is used to implement subroutine calls from a program on one machine to library routines on another machine. Since most programs are single threaded, the sender has little choice but to wait for a reply before continuing the program and possibly sending another request.

**Figure 3. 3.1** Stop-and Wait protocol

**Link Utilization in Stop-and-Wait**

Let us assume the following:

*Transmission time*: The time it takes for a station to transmit a frame (normalized to a value of 1).

*Propagation delay*: The time it takes for a bit to travel from sender to receiver (expressed as *a*).

- $a < 1$ :The frame is sufficiently long such that the first bits of the frame arrive at the destination before the source has completed transmission of the frame.
- $a > 1$: Sender completes transmission of the entire frame before the leading bits of the frame arrive at the receiver.
- The link utilization U = 1/(1+2a),

  a = Propagation time / transmission time

It is evident from the above equation that the link utilization is strongly dependent on the ratio of the propagation time to the transmission time. When the propagation time is small, as in case of LAN environment, the link utilization is good. But, in case of long propagation delays, as in case of satellite communication, the utilization can be very poor. To improve the link utilization, we can use the following (sliding-window) protocol instead of using stop-and-wait protocol.

## 3.3.2.2 Sliding Window

With the use of multiple frames for a single message, the stop-and-wait protocol does not perform well. Only one frame at a time can be in transit. In stop-and-wait flow control, if $a > 1$, serious inefficiencies result. Efficiency can be greatly improved by allowing multiple frames to be in transit at the same time. Efficiency can also be improved by making use of the full-duplex line. To keep track of the frames, sender station sends sequentially numbered frames. Since the sequence number to be used occupies a field in the frame, it should be of limited size. If the header of the frame allows k bits, the

sequence numbers range from 0 to $2^k - 1$. Sender maintains a list of sequence numbers that it is allowed to send (sender window). The size of the sender's window is at most $2^k - 1$. The sender is provided with a buffer equal to the window size. Receiver also maintains a window of size $2^k - 1$. The receiver acknowledges a frame by sending an ACK frame that includes the sequence number of the next frame expected. This also explicitly announces that it is prepared to receive the next N frames, beginning with the number specified. This scheme can be used to acknowledge multiple frames. It could receive frames 2, 3, 4 but withhold ACK until frame 4 has arrived. By returning an ACK with sequence number 5, it acknowledges frames 2, 3, 4 in one go. The receiver needs a buffer of size 1.

Sliding window algorithm is a method of flow control for network data transfers. TCP, the Internet's stream transfer protocol, uses a sliding window algorithm.

A sliding window algorithm places a buffer between the application program and the network data flow. For TCP, the buffer is typically in the operating system kernel, but this is more of an implementation detail than a hard-and-fast requirement.

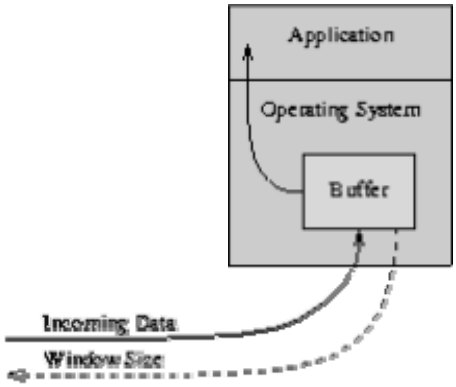



**Figure 3.3.2** Buffer in sliding window

Data received from the network is stored in the buffer, from where the application can read at its own pace. As the application reads data, buffer space is freed up to accept more input from the network. The *window* is the amount of data that can be "read ahead" - the size of the buffer, less the amount of valid data stored in it. *Window announcements* are used to inform the remote host of the current *window size*.

**Sender sliding Window:** At any instant, the sender is permitted to send frames with sequence numbers in a certain range (the sending window) as shown in Fig. 3.3.3.

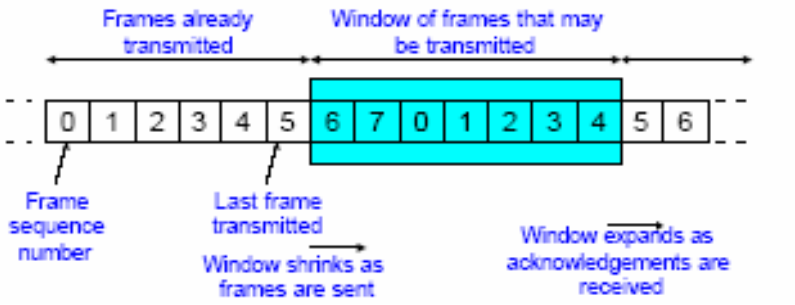



**Figure 3.3.3** Sender's window

**Receiver sliding Window:** The receiver always maintains a window of size 1 as shown in Fig. 3.3.4. It looks for a specific frame (frame 4 as shown in the figure) to arrive in a specific order. If it receives any other frame (out of order), it is discarded and it needs to be resent. However, the receiver window also slides by one as the specific frame is received and accepted as shown in the figure. The receiver acknowledges a frame by sending an ACK frame that includes the sequence number of the next frame expected. This also explicitly announces that it is prepared to receive the next N frames, beginning with the number specified. This scheme can be used to acknowledge multiple frames. It could receive frames 2, 3, 4 but withhold ACK until frame 4 has arrived. By returning an ACK with sequence number 5, it acknowledges frames 2, 3, 4 at one time. The receiver needs a buffer of size 1.

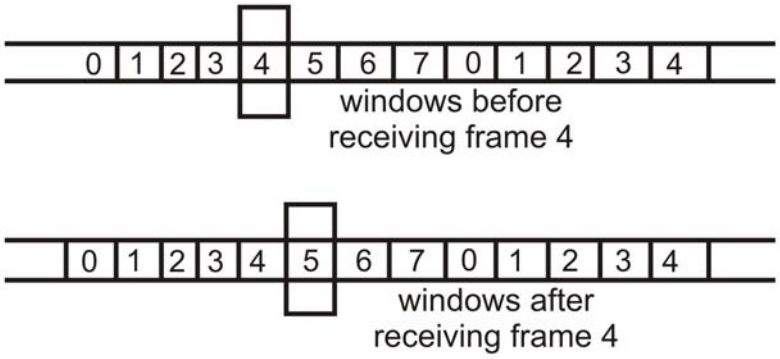



**Figure 3.3.4** Receiver sliding window

On the other hand, if the local application can process data at the rate it's being transferred; sliding window still gives us an advantage. If the window size is larger than the packet size, then multiple packets can be outstanding in the network, since the sender knows that buffer space is available on the receiver to hold all of them. Ideally, a steady-state condition can be reached where a series of packets (in the forward direction) and window announcements (in the reverse direction) are constantly in transit. As each new window announcement is received by the sender, more data packets are transmitted. As the application reads data from the buffer (remember, we're assuming the application can keep up with the network), more window announcements are generated. Keeping a series of data packets in transit ensures the efficient use of network resources.

Hence, Sliding Window Flow Control

- Allows transmission of multiple frames
- Assigns each frame a k-bit sequence number
- Range of sequence number is $[0 \ldots 2^k-1]$, i.e., frames are counted modulo 2k.

The link utilization in case of Sliding Window Protocol

U = 1, for $N > 2a + 1$

$N/(1+2a)$, for $N < 2a + 1$

Where N = the window size,

and a = Propagation time / transmission time

## 3.3.3 Error Control Techniques

When an error is detected in a message, the receiver sends a request to the transmitter to retransmit the ill-fated message or packet. The most popular retransmission scheme is known as Automatic-Repeat-Request (ARQ). Such schemes, where receiver asks transmitter to re-transmit if it detects an error, are known as reverse error correction techniques. There exist three popular ARQ techniques, as shown in Fig. 3.3.5.

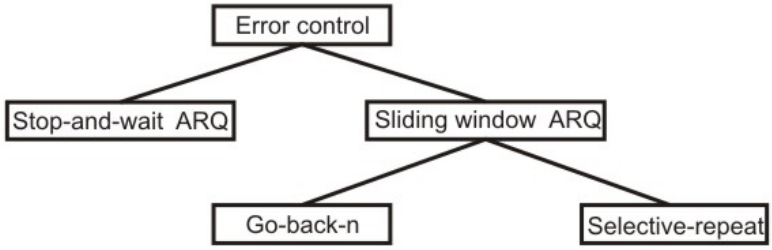



Figure 3.3.5 Error control techniques

### 3.3.3.1 Stop-and-Wait ARQ

In Stop-and-Wait ARQ, which is simplest among all protocols, the sender (say station A) transmits a frame and then waits till it receives positive acknowledgement (ACK) or negative acknowledgement (NACK) from the receiver (say station B). Station B sends an ACK if the frame is received correctly, otherwise it sends NACK. Station A sends a new frame after receiving ACK; otherwise it retransmits the old frame, if it receives a NACK. This is illustrated in Fig 3.3.6.

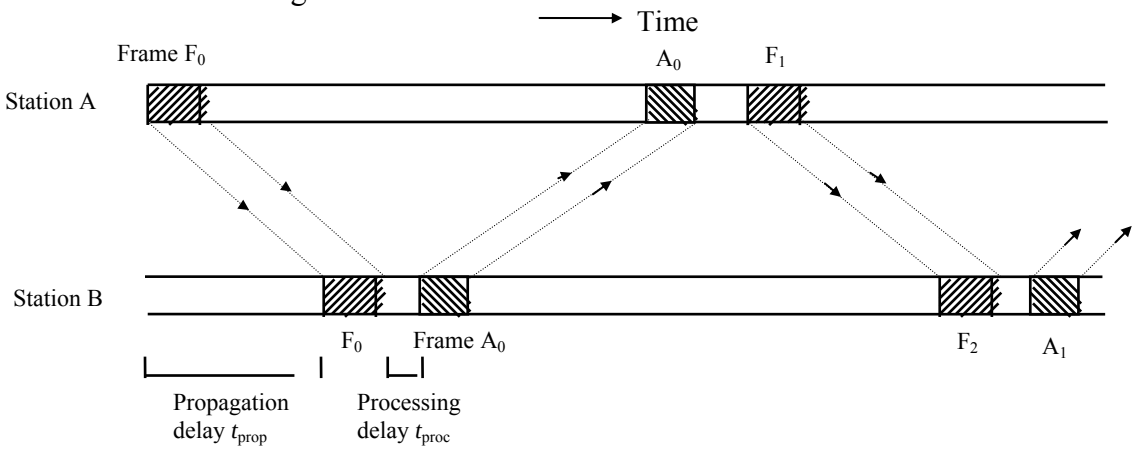



**Figure 3.3.6** Stop-And-Wait ARQ technique

To tackle the problem of a lost or damaged frame, the sender is equipped with a timer. In case of a lost ACK, the sender transmits the old frame. In the Fig. 3.3.7, the second PDU of Data is lost during transmission. The sender is unaware of this loss, but starts a timer after sending each PDU. Normally an ACK PDU is received before the

timer expires. In this case no ACK is received, and the timer counts down to zero and triggers retransmission of the same PDU by the sender. The sender always starts a timer following transmission, but in the second transmission receives an ACK PDU before the timer expires, finally indicating that the data has now been received by the remote node.

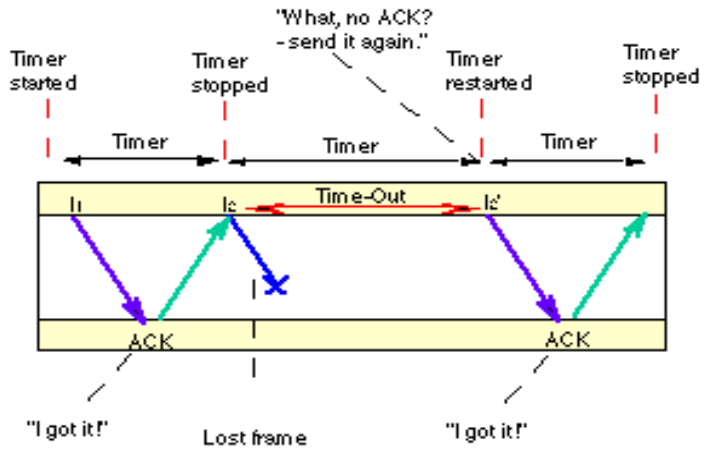



**Figure 3.3.7** Retransmission due to lost frame

The receiver now can identify that it has received a duplicate frame from the label of the frame and it is discarded

To tackle the problem of damaged frames, say a frame that has been corrupted during the transmission due to noise, there is a concept of NACK frames, i.e. Negative Acknowledge frames. Receiver transmits a NACK frame to the sender if it founds the received frame to be corrupted. When a NACK is received by a transmitter before the time-out, the old frame is sent again as shown in Fig. 3.3.8.

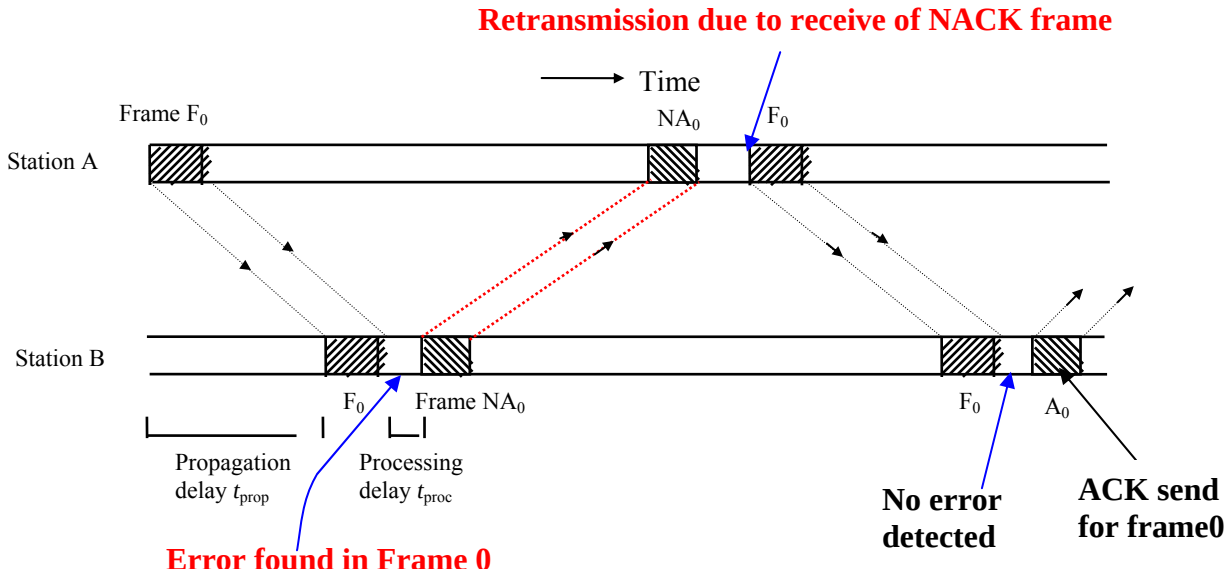



**Figure 3.3.8** Retransmission due to damaged frame

The main advantage of stop-and-wait ARQ is its simplicity. It also requires minimum buffer size. However, it makes highly inefficient use of communication links, particularly when 'a' is large.

### 3.3.3.2 Go-back-N ARQ

The most popular ARQ protocol is the go-back-N ARQ, where the sender sends the frames continuously without waiting for acknowledgement. That is why it is also called as *continuous ARQ*. As the receiver receives the frames, it keeps on sending ACKs or a NACK, in case a frame is incorrectly received. When the sender receives a NACK, it retransmits the frame in error plus all the succeeding frames as shown in Fig.3.3.9. Hence, the name of the protocol is go-back-N ARQ. If a frame is lost, the receiver sends NAK after receiving the next frame as shown in Fig. 3.3.10. In case there is long delay before sending the NAK, the sender will resend the lost frame after its timer times out. If the ACK frame sent by the receiver is lost, the sender resends the frames after its timer times out as shown in Fig. 3.3.11.

Assuming full-duplex transmission, the receiving end sends piggybacked acknowledgement by using some number in the ACK field of its data frame. Let us assume that a 3-bit sequence number is used and suppose that a station sends frame 0 and gets back an RR1, and then sends frames 1, 2, 3, 4, 5, 6, 7, 0 and gets another RR1.This might either mean that RR1 is a cumulative ACK or all 8 frames were damaged. This ambiguity can be overcome if the maximum window size is limited to 7, i.e. for a k-bit sequence number field it is limited to $2^k$-1. The number N (=$2^k$-1) specifies how many frames can be sent without receiving acknowledgement.

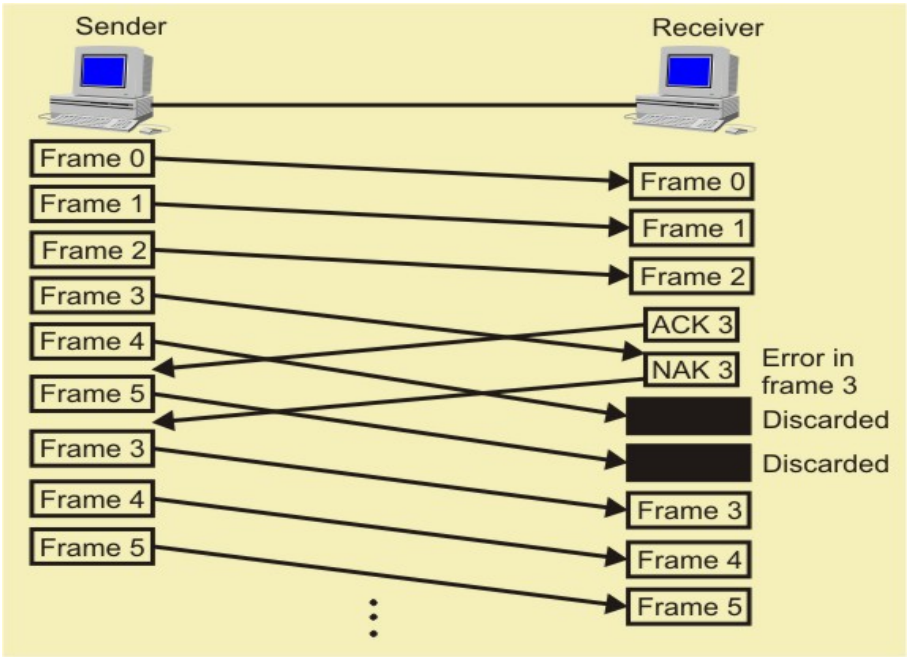



**Figure 3.3.9** Frames in error in go-Back-N ARQ

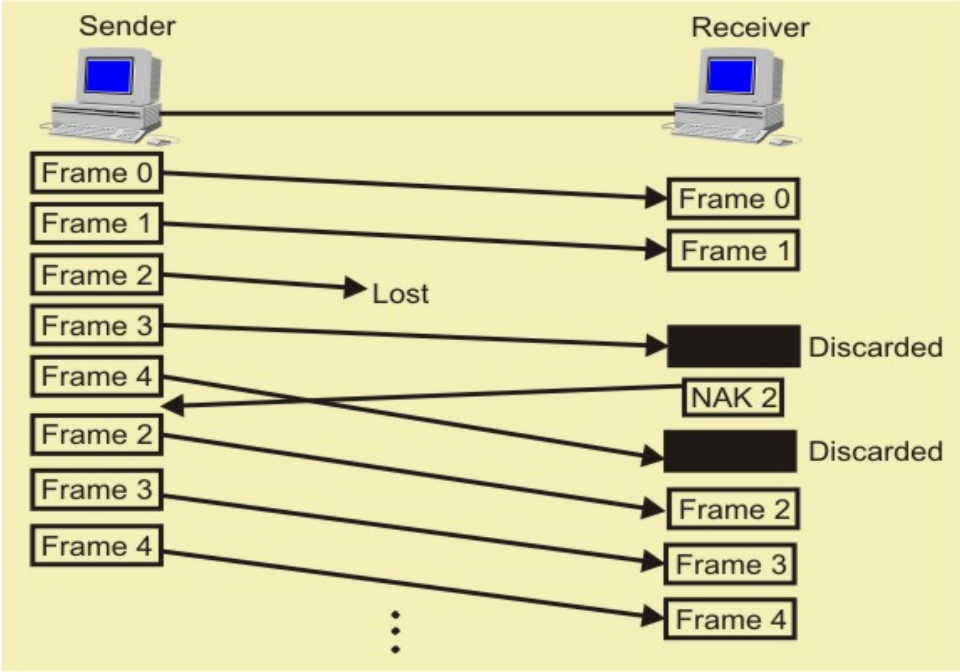



**Figure 3.3.10** Lost Frames in Go-Back-N ARQ

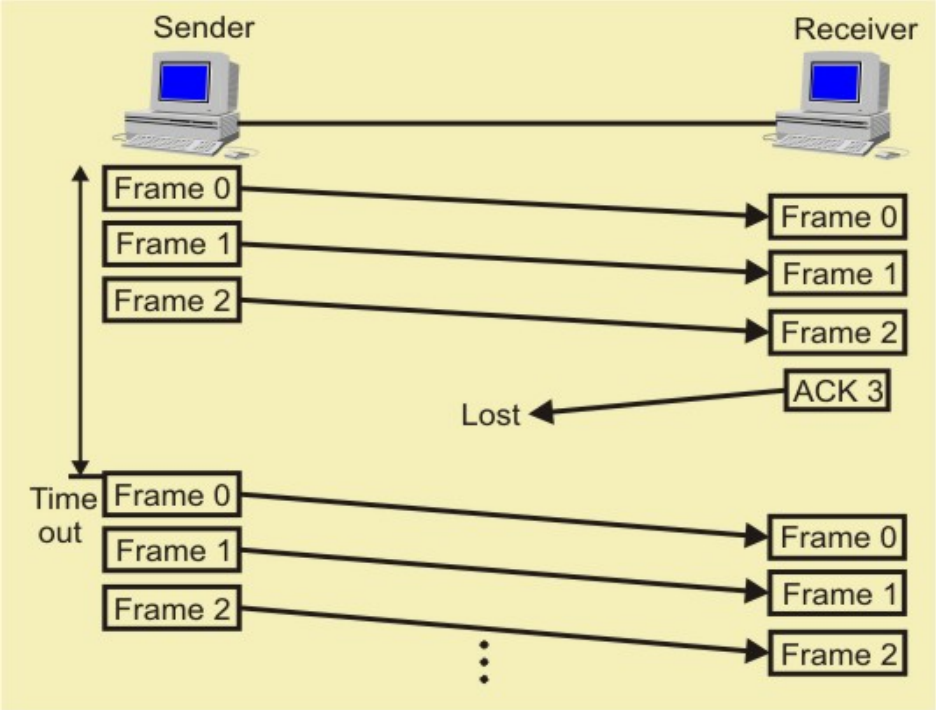



**Figure 3.3.11** Lost ACK in Go-Back-N ARQ

If no acknowledgement is received after sending N frames, the sender takes the help of a timer. After the time-out, it resumes retransmission. The go-back-N protocol

also takes care of damaged frames and damaged ACKs. This scheme is little more complex than the previous one but gives much higher throughput.

Assuming full-duplex transmission, the receiving end sends piggybacked acknowledgement by using some number in the ACK field of its data frame. Let us assume that a 3-bit sequence number is used and suppose that a station sends frame 0 and gets back an RR1, and then sends frames 1, 2, 3, 4, 5, 6, 7, 0 and gets another RR1.This might either mean that RR1 is a cumulative ACK or all 8 frames were damaged. This ambiguity can be overcome if the maximum window size is limited to 7, i.e. for a k-bit sequence number field it is limited to $2^k$-1. The number N (=$2^k$-1) specifies how many frames can be sent without receiving acknowledgement. If no acknowledgement is received after sending N frames, the sender takes the help of a timer. After the time-out, it resumes retransmission. The go-back-N protocol also takes care of damaged frames and damaged ACKs. This scheme is little more complex than the previous one but gives much higher throughput.

### 3.3.3.3 Selective-Repeat ARQ

The selective-repetitive ARQ scheme retransmits only those for which NAKs are received or for which timer has expired, this is shown in the Fig.3.3.12. This is the most efficient among the ARQ schemes, but the sender must be more complex so that it can send out-of-order frames. The receiver also must have storage space to store the post-NAK frames and processing power to reinsert frames in proper sequence.

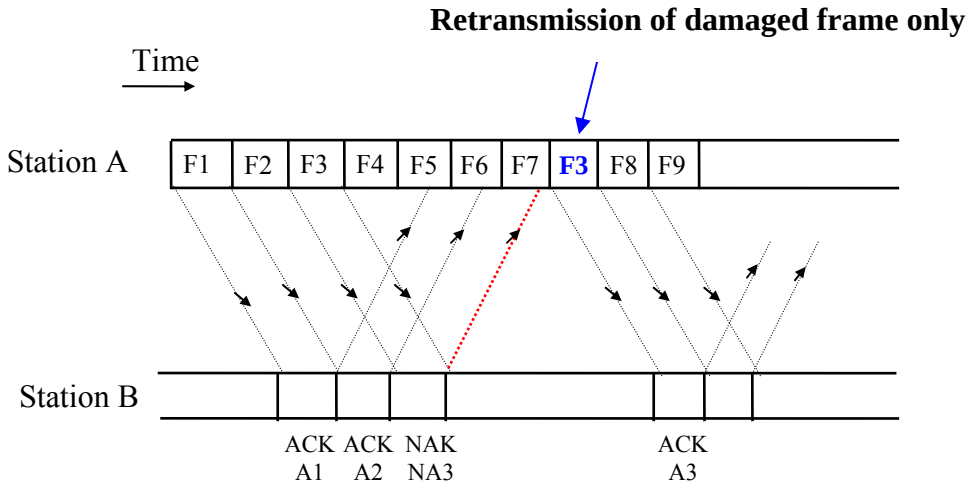



**Figure 3.3.12** Selective-repeat Reject

## Fill In The Blanks

1. _____Control is a technique for speed-matching of transmitter and receiver.
2. _____ Control provides for the addition of binary digits (redundant bits) that can be used to identify if there has been an error in the transmission of one or more bits.
3. There are two methods developed for flow control namely _____ and _____.
4. Stop-And-Wait is also known as _____.
5. Sliding Window Flow Control allows transmission of _____ frames.
6. Sliding Window Flow Control assigns each frame a _____-bit sequence number.
7. Sliding window ARQ can be of two types namely, _____ and _____.

## Short Answer Questions

**1. What are the key functions of error control techniques?**

**Ans:** There are basically two types of errors, namely, (a) Damaged Frame (b) Lost Frame. The key functions for error control techniques are as follows:

- Error detection
- Sending of positive acknowledgement (ACK) by the receiver for no error
- Sending of negative acknowledgement (NAK) by the receiver for error
- Setting of timer for lost frame
- Numbering of frames

**2. Why is flow control needed?**

**Ans:** In case of data communication between a sender and a receiver, it may so happen that the rate at which data is transmitted by a fast sender is not acceptable by a slow receiver. IN such a situation, there is a need of flow control so that a fast transmitter does not overwhelm a slow receiver.

**3. Mention key advantages and disadvantages of stop-and-wait ARQ technique?**

**Ans:** Advantages of stop-and-wait ARQ are:

a. Simple to implement
b. Frame numbering is modulo-2, i.e. only 1 bit is required.

The main disadvantage of stop-and-wait ARQ is that when the propagation delay is long, it is extremely inefficient.

**4. Consider the use of 10 K-bit size frames on a 10 Mbps satellite channel with 270 ms delay. What is the link utilization for stop-and-wait ARQ technique assuming P =** $10^{-3}$**?**

**Ans:** Link utilization = (1-P) / (1+2a)
Where a = (Propagation Time) / (Transmission Time)
Propagation time = 270 msec
Transmission time = (frame length) / (data rate)
= (10 K-bit) / (10 Mbps)
= 1 msec
Hence, a = 270/1 = 270
Link utilization = 0.999/(1+2*270) ≈0.0018 =0.18%

**5. What is the channel utilization for the go-back-N protocol with window size of 7 for the problem 3?**

**Ans:** Channel utilization for go-back-N
= N(1 – P) / (1 + 2a)(1-P+NP)
P = probability of single frame error ≈ $10^{-3}$
Channel utilization ≈ 0.01285 = 1.285%

**6. In what way selective-repeat is better than go-back-N ARQ technique?**

**Ans :** In selective-repeat scheme only the frame in error is retransmitted rather than transmitting all the subsequent frames. Hence it is more efficient than go-back-N ARQ technique.

**7. In what situation Stop-and-Wait protocol works efficiently?**

**Ans:** In case of Stop-and-Wait protocol, the transmitter after sending a frame waits for the acknowledgement from the receiver before sending the next frame. This protocol works efficiently for long frames, where propagation time is small compared to the transmission time of the frame.

**8. How the inefficiency of Stop-and-Wait protocol is overcome in sliding window protocol?**

**Ans:** The Stop-and-Wait protocol is inefficient when large numbers of small packets are send by the transmitter since the transmitter has to wait for the acknowledgement of each individual packet before sending the next one. This problem can be overcome by sliding window protocol. In sliding window protocol multiple frames (up to a fixed number of frames) are send before receiving an acknowledgement from the receiver.

**9. What is piggybacking? What is its advantage?**

**Ans:** In practice, the link between receiver and transmitter is full duplex and usually both transmitter and receiver stations send data to each over. So, instead of sending separate acknowledgement packets, a portion (few bits) of the data frames can be used for acknowledgement. This phenomenon is known as piggybacking.
The piggybacking helps in better channel utilization. Further, multi-frame acknowledgement can be done.

**10. For a k-bit numbering scheme, what is the range of sequence numbers used in sliding window protocol?**

**Ans:** For k-bit numbering scheme, the total number of frames, N, in the sliding window can be given as follows (using modulo-k).

$N = 2^k - 1$

Hence the range of sequence numbers is: 0, 1, 2, and 3 … $2^k - 1$