UNIT-2

Datalink Layer

Syllabus

UNIT – II

- Data Link Layer Design Issues, Error Detection and Correction, Elementary Data Link Control Protocols, Sliding Window Protocols, HDLC, PPP.

Data Link Layer Design Issues

- Network layer services
- Framing
- Error control
- Flow control

Data Link Layer

- Algorithms for achieving:
  - Reliable, +
  - Efficient,

    communication of a whole units – frames (as opposed to bits – Physical Layer) between two machines.
- Two machines are connected by a communication channel that acts conceptually like a wire (e.g., telephone line, coaxial cable, or wireless channel).
- Essential property of a channel that makes it “wire-like” connection is that the bits are delivered in exactly the same order in which they are sent.

Data Link Layer

- For ideal channel (no distortion, unlimited bandwidth and no delay) the job of data link layer would be trivial.

- However, limited bandwidth, distortions and delay makes this job very difficult.

Data Link Layer Design Issues

- Physical layer delivers bits of information to and from data link layer. The functions of Data Link Layer are:
  1. Providing a well-defined service interface to the network layer.
  2. Dealing with transmission errors.
  3. Regulating the flow of data so that slow receivers are not swamped by fast senders.
- Data Link layer
  - Takes the packets from Network layer, and
  - Encapsulates them into **frames**

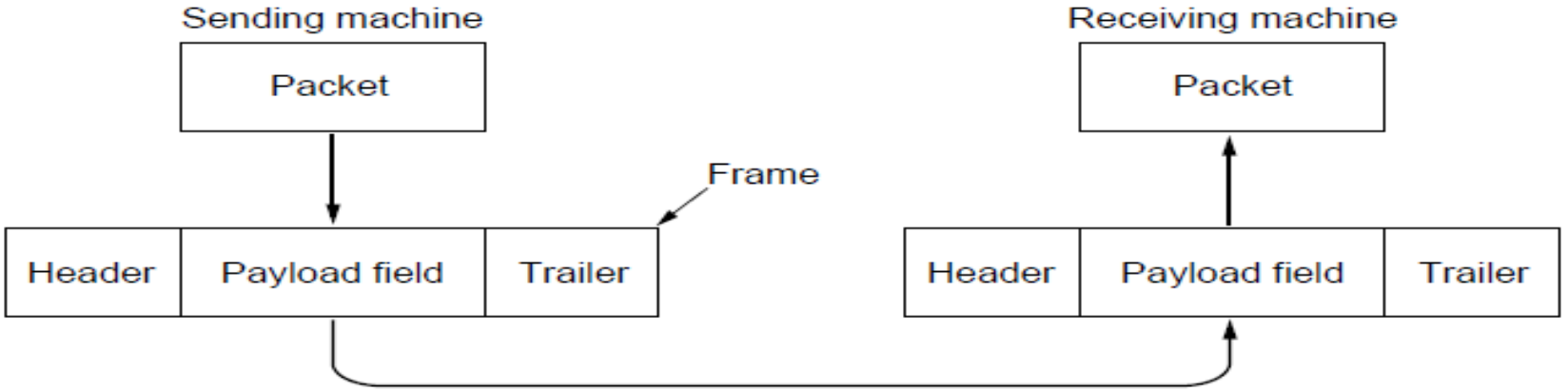
Data Link Layer Design Issues

- Each frame has a
  - frame header – a field for holding the packet, and
  - frame trailer.
- Frame Management is what Data Link Layer does.

- See figure in the next slide:

Packets and Frames

Relationship between packets and frames.

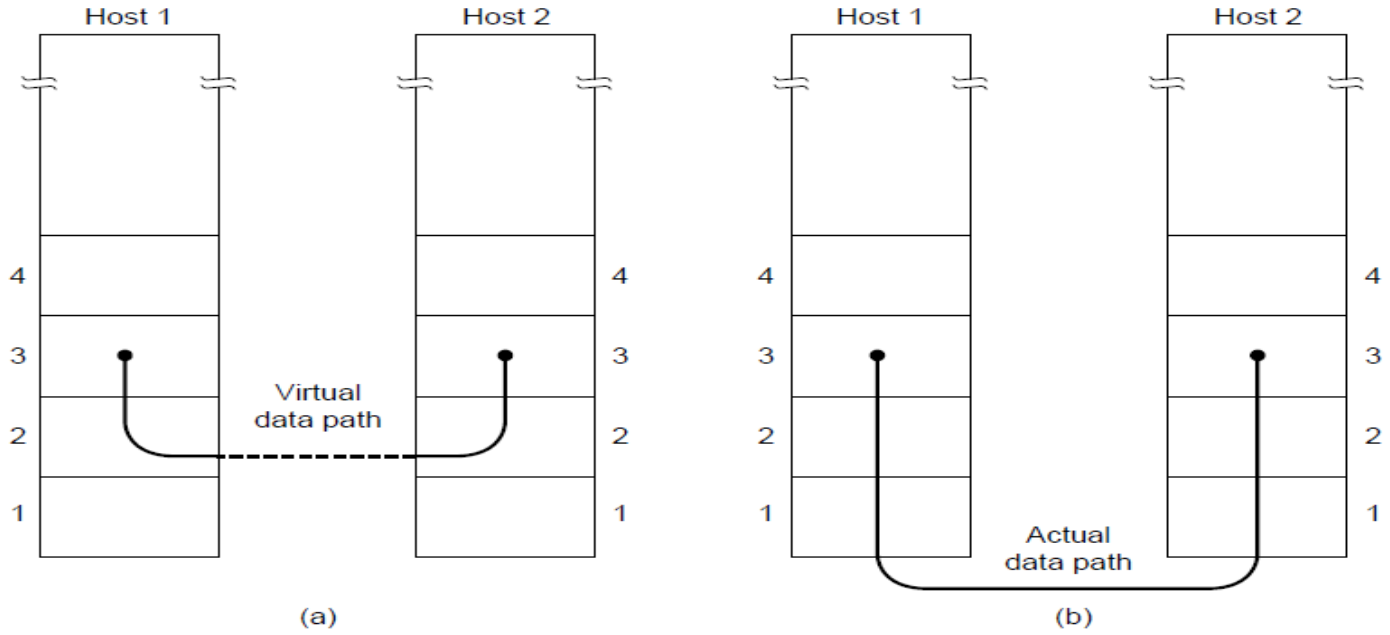


Services Provided to the Network Layer

- Principal Service Function of the data link layer is to transfer the data from the network layer on the source machine to the network layer on the destination machine.
  - Process in the network layer that hands some bits to the data link layer for transmission.
  - Job of data link layer is to transmit the bits to the destination machine so they can be handed over to the network layer there (see figure in the next slide).

Network Layer Services

(a) Virtual communication. (b) Actual communication.



Possible Services Offered

1. Unacknowledged connectionless service.
2. Acknowledged connectionless service.
3. Acknowledged connection-oriented service.

Unacknowledged Connectionless Service

- It consists of having the source machine send independent frames to the destination machine without having the destination machine acknowledge them.
- Example: Ethernet, Voice over IP, etc. in all the communication channel were real time operation is more important that quality of transmission.

Acknowledged Connectionless Service

- Each frame send by the Data Link layer is acknowledged and the sender knows if a specific frame has been received or lost.
- Typically the protocol uses a specific time period that if has passed without getting acknowledgment it will re-send the frame.
- This service is useful for commutation when an unreliable channel is being utilized (e.g., 802.11 WiFi).

- Network layer does not know frame size of the packets and other restriction of the data link layer. Hence it becomes necessary for data link layer to have some mechanism to optimize the transmission.

Acknowledged Connection Oriented Service

- Source and Destination establish a connection first.
- Each frame sent is numbered
  - Data link layer guarantees that each frame sent is indeed received.
  - It guarantees that each frame is received only once and that all frames are received in the correct order.
- Examples:
  - Satellite channel communication,
  - Long-distance telephone communication, etc.

Acknowledged Connection Oriented Service

- Three distinct phases:
  1. Connection is established by having both side initialize variables and counters needed to keep track of which frames have been received and which ones have not.
  2. One or more frames are transmitted.
  3. Finally, the connection is released – freeing up the variables, buffers, and other resources used to maintain the connection.

Framing

- To provide service to the network layer the data link layer must use the service provided to it by physical layer.
- Stream of data bits provided to data link layer is not guaranteed to be without errors.
- Errors could be:
  - Number of received bits does not match number of transmitted bits (deletion or insertion)
  - Bit Value
- It is up to data link layer to correct the errors if necessary.

Framing

- Transmission of the data link layer starts with breaking up the bit stream
  - into discrete frames
  - Computation of a checksum for each frame, and
  - Include the checksum into the frame before it is transmitted.
- Receiver computes its checksum error for a receiving frame and if it is different from the checksum that is being transmitted will have to deal with the error.

- Framing is more difficult than one could think!

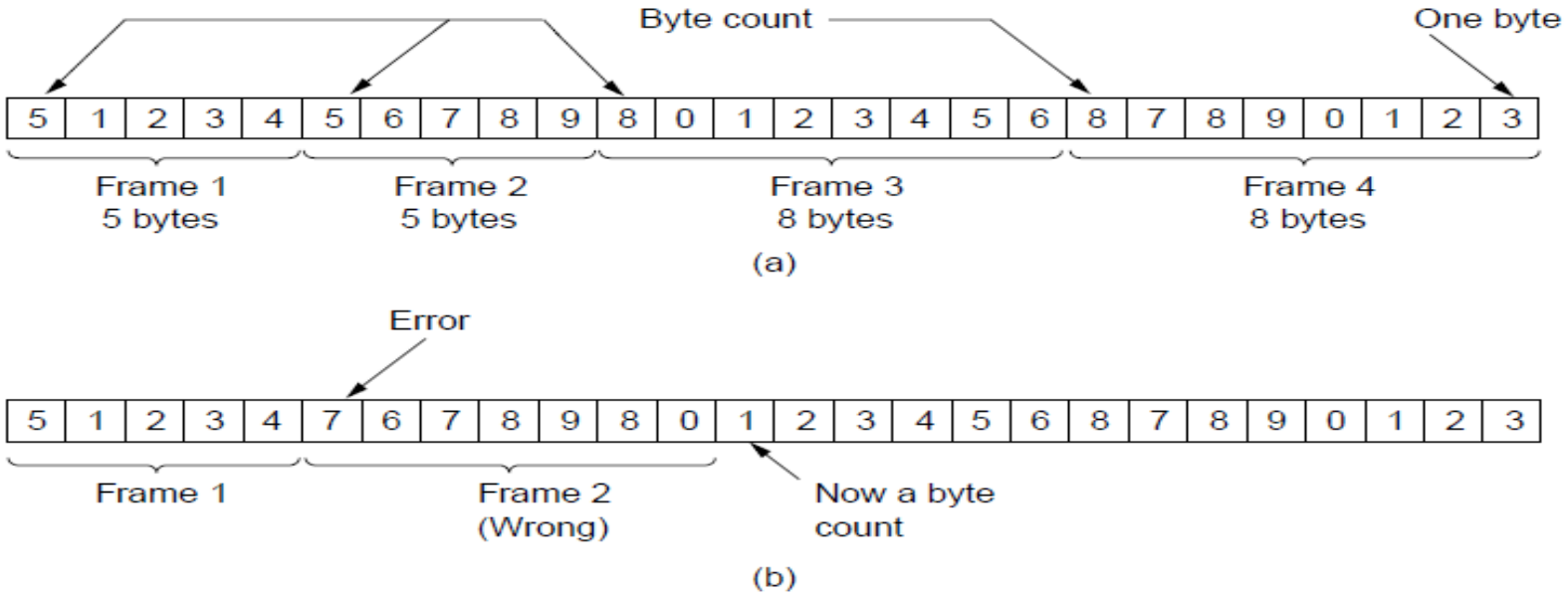
Framing Methods

1. Byte count.
2. Flag bytes with byte stuffing.
3. Flag bits with bit stuffing.
4. Physical layer coding violations.

Byte Count Framing Method

- It uses a field in the header to specify the number of bytes in the frame.
- Once the header information is being received it will be used to determine end of the frame.
- See figure in the next slide:
- Trouble with this algorithm is that when the count is incorrectly received the destination will get out of synch with transmission.
  - Destination may be able to detect that the frame is in error but it does not have a means (in this algorithm) how to correct it.

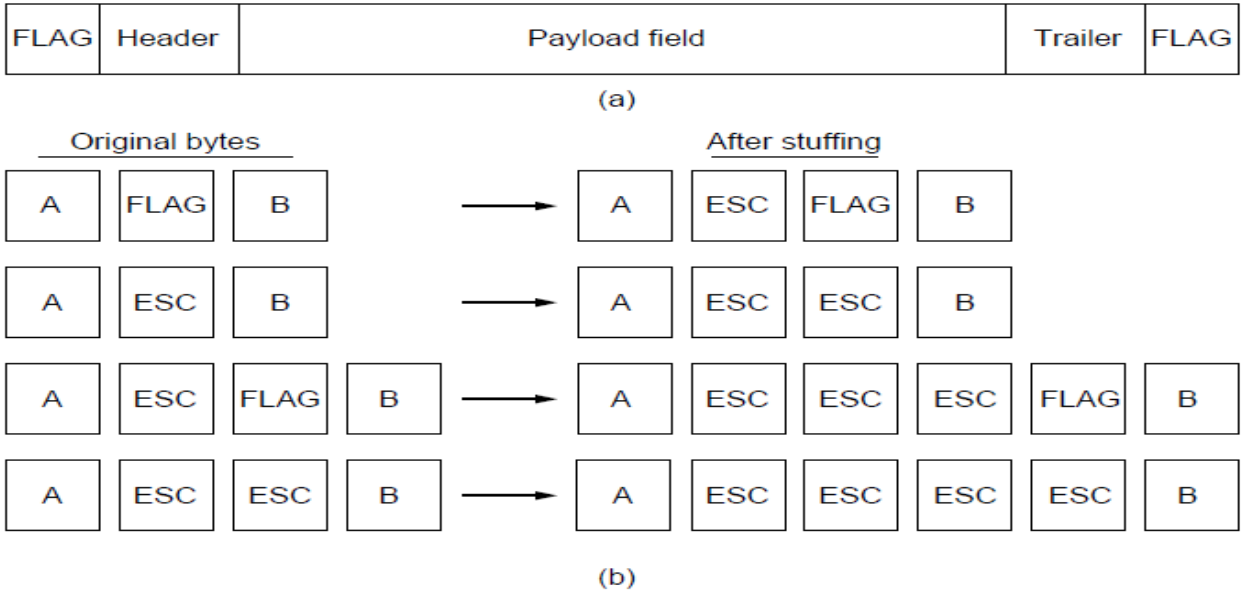
Framing (1)


Byte count
One byte
5 1 2 3 4 5 6 7 8 9 8 0 1 2 3 4 5 6 8 7 8 9 0 1 2 3
Frame 1
5 bytes
Frame 2
5 bytes
Frame 3
8 bytes
Frame 4
8 bytes
(a)
Error
5 1 2 3 4 7 6 7 8 9 8 0 1 2 3 4 5 6 8 7 8 9 0 1 2 3
Frame 1
Frame 2
(Wrong)
Now a byte
count
(b)

Flag Bytes with Byte Stuffing Framing Method

- This methods gets around the boundary detection of the frame by having each appended by the frame start and frame end special bytes.
- If they are the same (beginning and ending byte in the frame) they are called **flag byte**.
- In the next slide figure this byte is shown as FLAG.
- If the actual data contains a byte that is identical to the FLAG byte (e.g., picture, data stream, etc.) the convention that can be used is to have escape character inserted just before the “FLAG” character.

Framing (2)

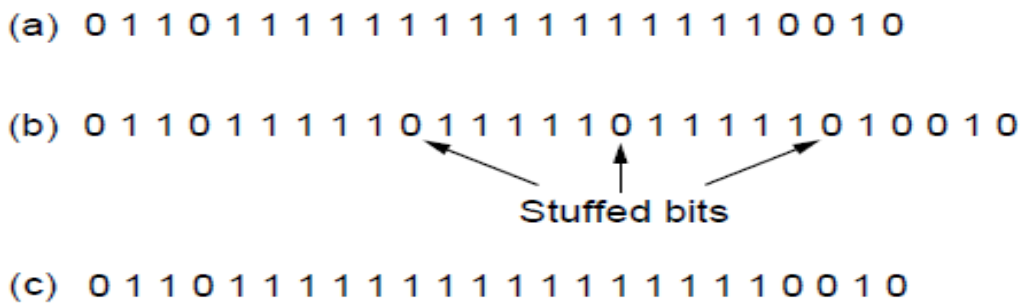


- A frame delimited by flag bytes.
- Four examples of byte sequences before and after byte stuffing.

Flag Bits with Bit Stuffing Framing Method

- This methods achieves the same thing as Byte Stuffing method by using Bits (1) instead of Bytes (8 Bits).
- It was developed for High-level Data Link Control (HDLC) protocol.
- Each frames begins and ends with a special bit pattern:
  - 01111110 or 0x7E <- Flag Byte
  - Whenever the sender's data link layer encounters five consecutive 1s in the data it automatically stuffs a 0 bit into the outgoing bit stream.
  - USB uses bit stuffing.

Framing (3)



Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after destuffing.

Framing

- Many data link protocols use a combination of presented methods for safety. For example in Ethernet and 802.11 each frame begin with a well-defined pattern called a preamble.
- Preamble is typically 72 bits long.
- It is then followed by a length field.

Error Control

- After solving the marking of the frame with start and end the data link layer has to handle eventual errors in transmission or detection.
  - Ensuring that all frames are delivered to the network layer at the destination and in proper order.
- Unacknowledged connectionless service: it is OK for the sender to output frames regardless of its reception.
- Reliable connection-oriented service: it is NOT OK.

Error Control

- Reliable connection-oriented service usually will provide a sender with some feedback about what is happening at the other end of the line.
  - Receiver Sends Back Special Control Frames.
  - If the Sender Receives positive Acknowledgment it will know that the frame has arrived safely.
- Timer and Frame Sequence Number for the Sender is Necessary to handle the case when there is no response (positive or negative) from the Receiver .

Flow Control

- Important Design issue for the cases when the sender is running on a fast powerful computer and receiver is running on a slow low-end machine.
- Two approaches:
  1. Feedback-based flow control
  2. Rate-based flow control

Feedback-based Flow Control

- Receiver sends back information to the sender giving it permission to send more data, or
- Telling sender how receiver is doing.

Rate-based Flow Control

Built in mechanism that limits the rate at which sender may transmit data, without the need for feedback from the receiver.

Error Detection and Correction

- Two basic strategies to deal with errors:
  1. Include only enough redundancy to allow the receiver to deduce that an error has occurred (but not which error).**Error detecting codes.**
  2. Include enough redundant information to enable the receiver to deduce what the transmitted data must have been. **Error correcting codes.**

Error Detection and Correction

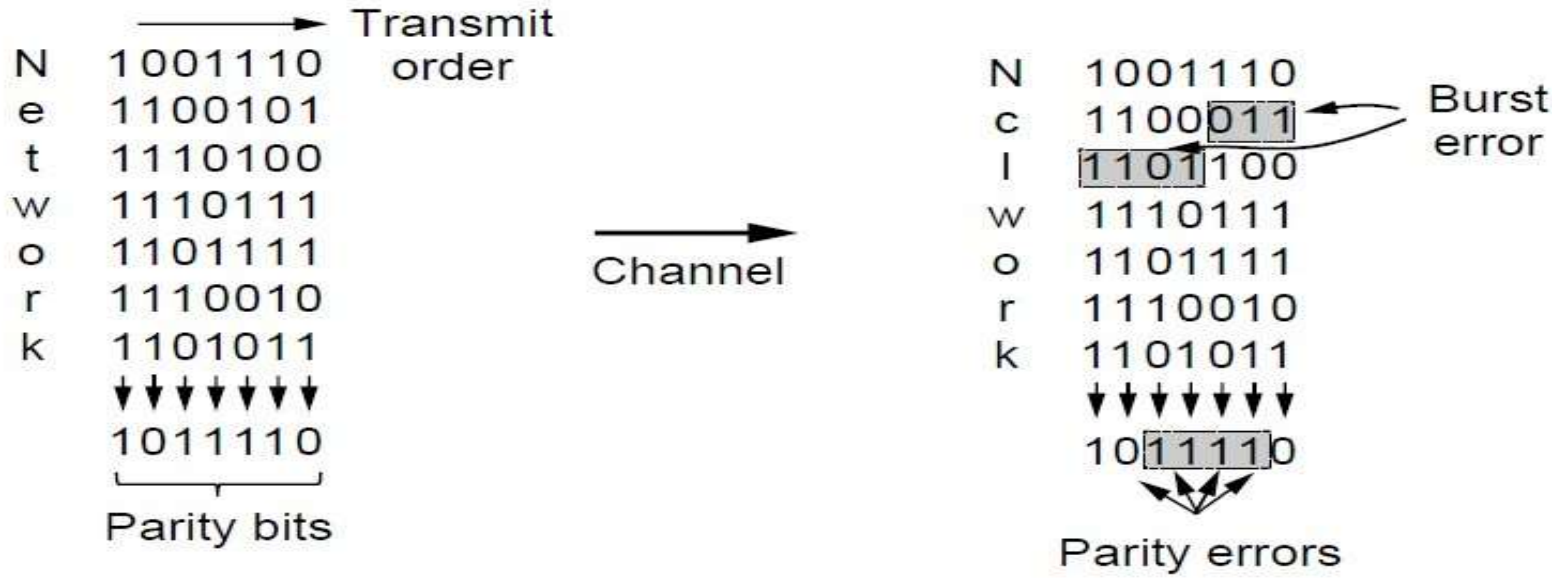
- Error codes are examined in Link Layer because this is the first place that we have run up against the problem of reliability transmitting groups of bits.
  - Codes are reused because reliability is an overall concern.
  - The error correcting code are also seen in the physical layer for noise channels.
  - Commonly they are used in link, network and transport layer.

Error-Detecting Codes (1)

Linear, systematic block codes

1. Parity.
2. Checksums.
3. Cyclic Redundancy Checks (CRCs).

Error-Detecting Codes (2)



2-D parity Check

Original Data

10011001	11100010	00100100	10000100

Row parities

10011001	0
11100010	0
00100100	0
10000100	0
11011011	0

Column parities

100110010	111000100	001001000	100001000	110110110

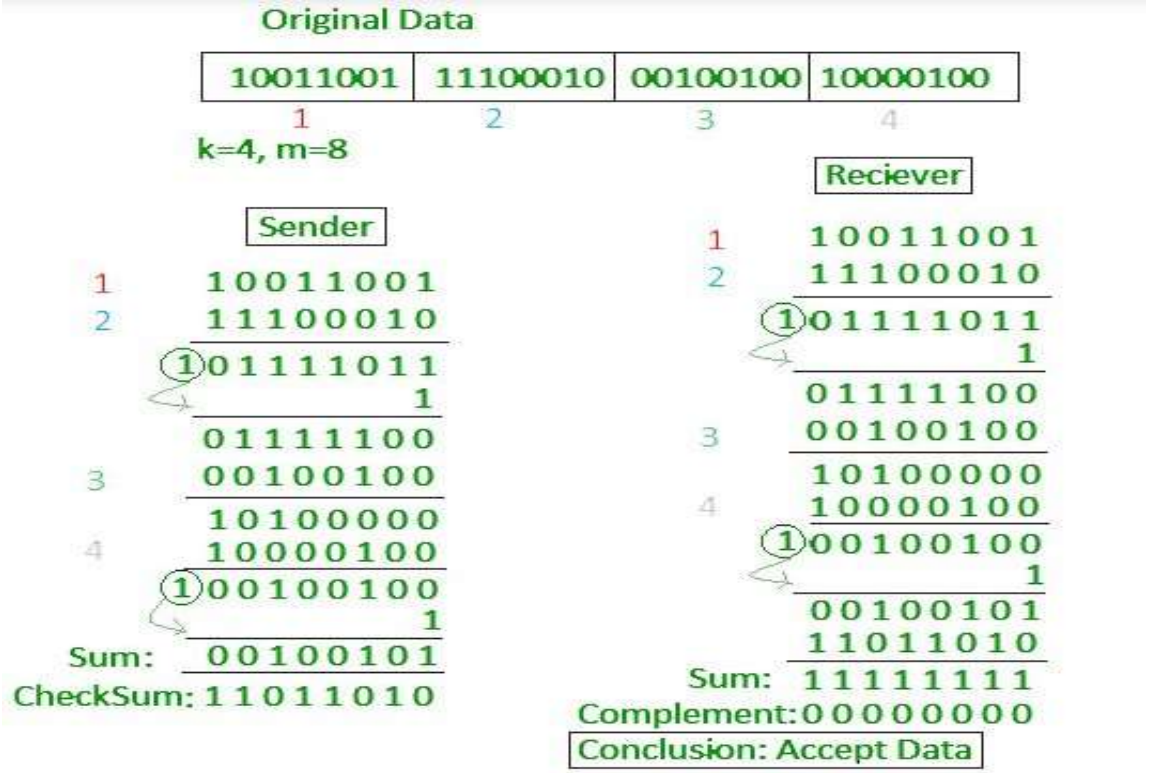
Data to be sent

Checksum

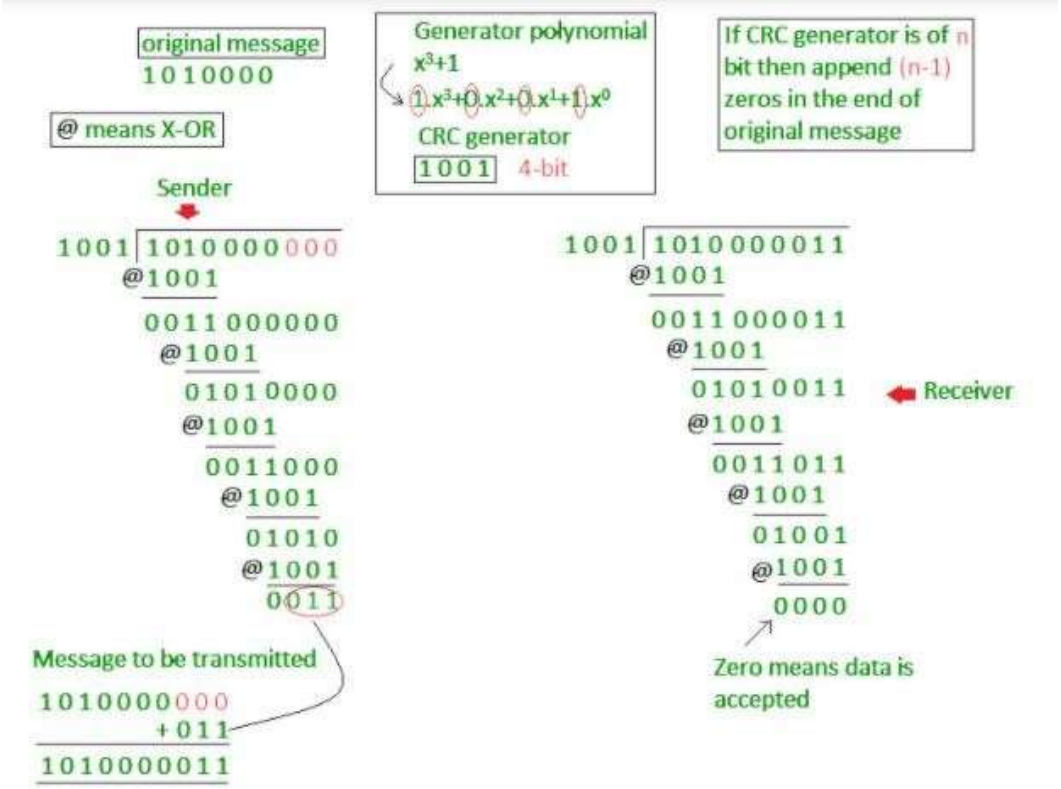
3. Checksum

- In checksum error detection scheme, the data is divided into k segments each of m bits.
- In the sender's end the segments are added using 1's complement arithmetic to get the sum. The sum is complemented to get the checksum.
- The checksum segment is sent along with the data segments.
- At the receiver's end, all received segments are added using 1's complement arithmetic to get the sum. The sum is complemented.
- If the result is zero, the received data is accepted; otherwise discarded.

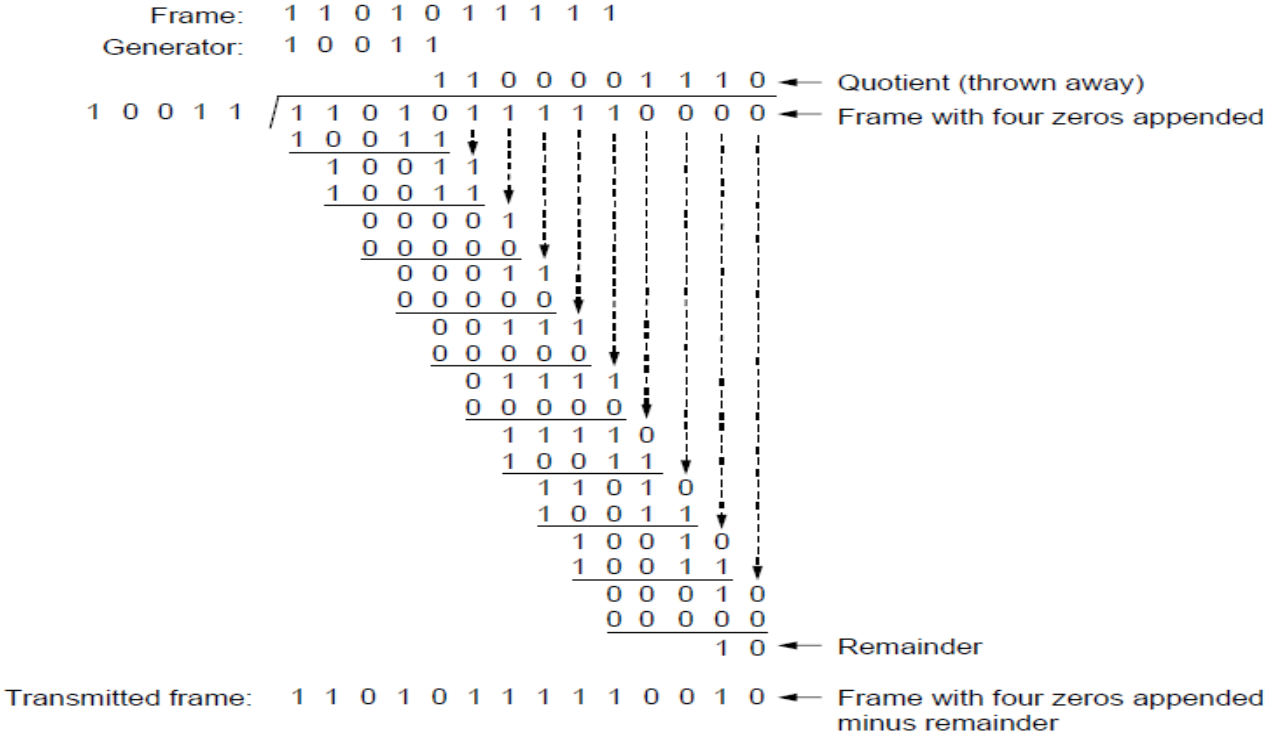
Checksum-Example



Cyclic Redundancy Check



Error-Detecting Codes (3)



Example calculation of the CRC

Error Detection and Correction

- Error codes have been developed after long fundamental research conducted in mathematics.
- Many protocol standards get codes from the large field in mathematics.

Error Detection & Correction Code (1)

1. Hamming codes.
2. Binary convolutional codes.
3. Reed-Solomon codes.
4. Low-Density Parity Check codes.

Error Detection & Correction Code

- All the codes presented in previous slide add redundancy to the information that is being sent.
- A frame consists of
  - $\boldsymbol{m}$ data bits (message) and
  - $\boldsymbol{r}$ redundant bits (check).
- ***Block code*** - the $\boldsymbol{r}$ check bits are computed solely as function of the $\boldsymbol{m}$ data bits with which they are associated.

Error Detection & Correction Code

Example

- Transmitted: 10001001
- Received: 10110001

XOR operation gives number of bits that are different.

- XOR: 00111000

- Number of bit positions in which two code words differ is called ***Hamming Distance***. It shows that two codes are ***d*** distance apart, and it will require ***d*** errors to convert one into the other.

Error Detection & Correction Code

- All 2^m possible data messages are legal, but due to the way the check bits are computers not all 2^n possible code words are used.
- Only small fraction of $2^m/2^n=1/2^r$ ***are possible will be legal code words.***
- The error-detecting and error-correcting codes of the block code depend on this Hamming distance.
- To reliably detect d error, one would need a distance $d+1$ code.
- To correct d error, one would need a distance $2d+1$ code.

Error Detection & Correction Code

Example:

- 4 valid codes:
  - 0000000000
  - 0000011111
  - 1111100000
  - 1111111111
- Minimal Distance of this code is 5 => can correct double errors and it detect quadruple errors.
- 0000000111 => single or double – bit error. Hence the receiving end must assume the original transmission was 0000011111.
- 0000000000 => had triple error that was received as 0000000111 it would be detected in error.

Error Detection & Correction Code

- One cannot perform double errors and at the same time detect quadruple errors.
- Error correction requires evaluation of each candidate code word which may be time consuming search.
- Through design this search time can be minimized.
- In theory if n = m + r, this requirement becomes:

→ $(m + r + 1) \leq 2^r$

Hamming Code

- Code word: b1 b2 b3 b4 ….
- Check bits: The bits that are powers of 2 (p1, p2, p4, p8, p16, …).
- The rest of bits (m3, m5, m6, m7, m9, …) are filled with m data bits.
- Example of the Hamming code with m = 7 data bits and r = 4 check bits is given in the next slide.

The Hamming Code

Consider a message having four data bits (D) which is to be transmitted as a 7-bit code word by adding three error control bits. This would be called a (7,4) code. The three bits to be added are three EVEN Parity bits (P), where the parity of each is computed on different subsets of the message bits as shown below.

7	6	5	4	3	2	1	
D	D	D	P	D	P	P	7-BIT CODEWORD
D	-	D	-	D	-	P	(EVEN PARITY)
D	D	-	-	D	P	-	(EVEN PARITY)
D	D	D	P	-	-	-	(EVEN PARITY)

Hamming Code

- For example, the message 1101 would be sent as 1100110, since:

7	6	5	4	3	2	1	
1	1	0	0	1	1	0	7-BIT CODEWORD
1	-	0	-	1	-	**0**	(EVEN PARITY)
1	1	-	-	1	**1**	-	(EVEN PARITY)
1	1	0	**0**	-	-	-	(EVEN PARITY)

Hamming Code

- It may now be observed that if an error occurs in any of the seven bits, that error will affect different combinations of the three parity bits depending on the bit position.

- For example, suppose the above message 1100110 is sent and a single bit error occurs such that the code word 1110110 is received:

transmitted message		received message
1 1 0 0 1 1 0	---------->	1 1 1 0 1 1 0
BIT: 7 6 5 4 3 2 1		BIT: 7 6 5 4 3 2 1

The above error (in bit 5) can be corrected by examining which of the three parity bits was affected by the bad bit:

Hamming Code

7	6	5	4	3	2	1			
1	1	1	**0**	1	**1**	**0**	7-BIT CODEWORD		
1	-	1	-	1	-	0	(EVEN PARITY)	**NOT!**	**1**
1	1	-	-	1	1	-	(EVEN PARITY)	**OK!**	**0**
1	1	1	0	-	-	-	(EVEN PARITY)	**NOT!**	**1**

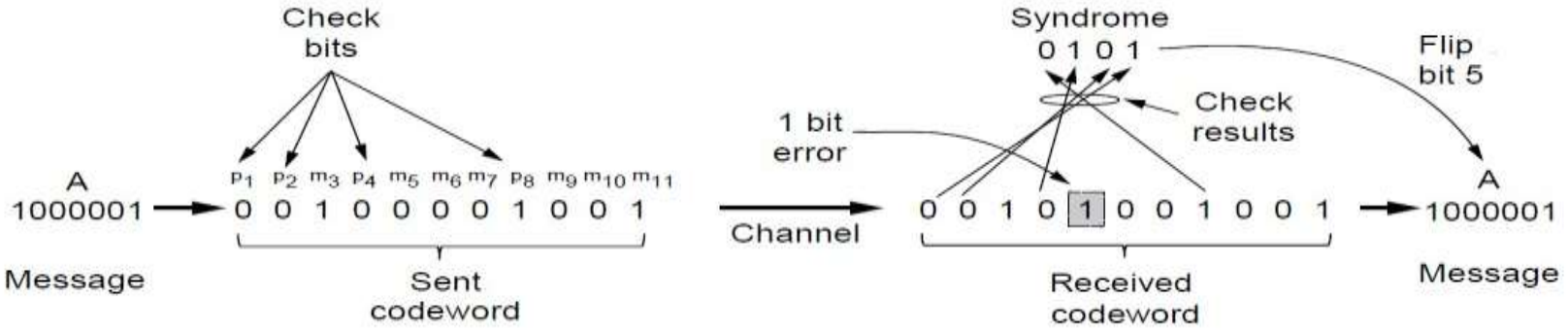
Hamming Code

- *In fact, the bad parity bits labeled **101** point directly to the bad bit since **101** binary equals **5**.* Examination of the 'parity circles' confirms that any single bit error could be corrected in this way.
- The value of the Hamming code can be summarized:

1. Detection of 2 bit errors (assuming no correction is attempted);
2. Correction of single bit errors;
3. Cost of 3 bits added to a 4-bit message.

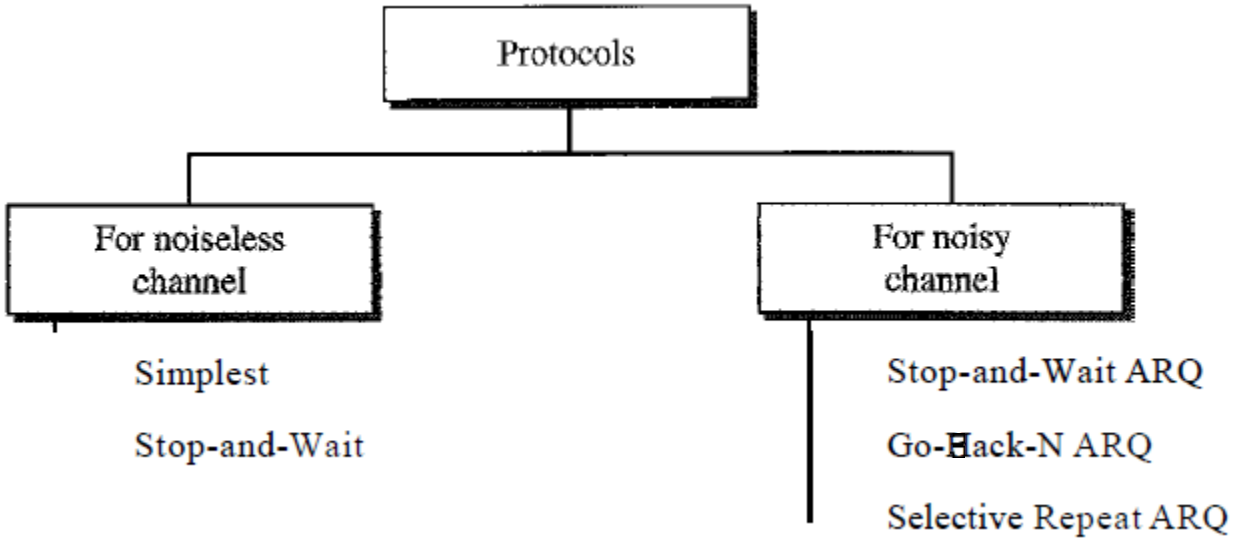
- The ability to correct single bit errors comes at a cost which is less than sending the entire message twice. (Recall that simply sending a message twice accomplishes no error correction.)

Error Detection Codes (2)



Example of an (11, 7) Hamming code correcting a single-bit error.

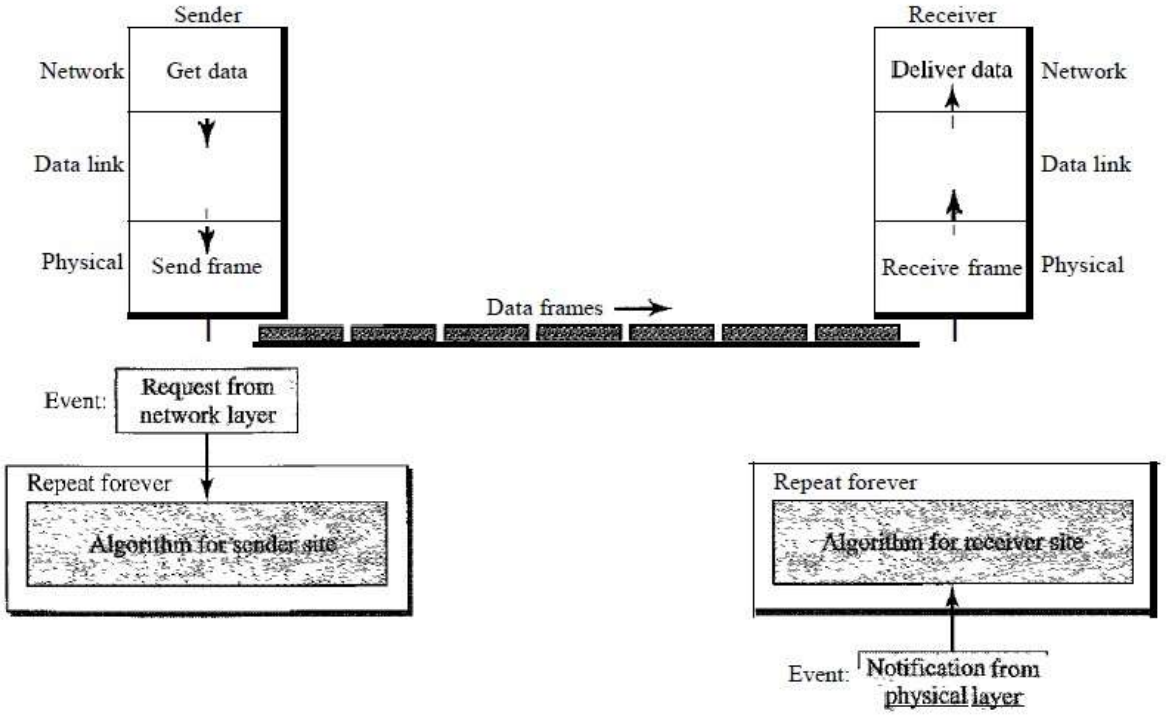
Flow Control and Error control



un realistic

Utopian simplex protocol(Noiseless channel)

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



Utopian simplex protocol

Algorithm 11.1 *Sender-site algorithm for the simplest protocol*

```
1  while (true)                          // Repeat forever
2  {
3    WaitForEvent()i                     // Sleep until an event occurs
4    if(Event(RequestToSend»             //There is a packet to send
5    {
6        GetData()i
7        MakeFrame()i
8        SendFrame()i                    //Send the frame
9    }
10 }
```

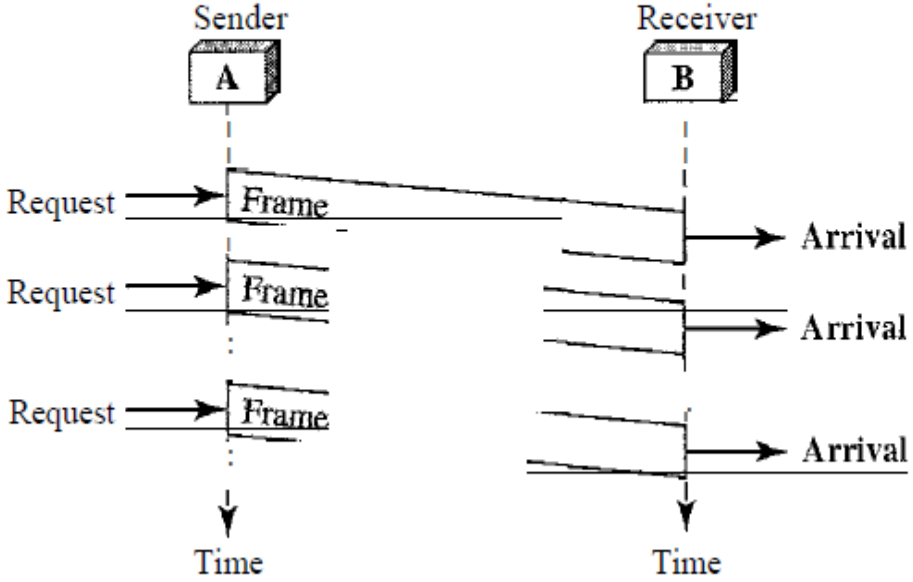
Utopian simplex protocol

Algorithm 11.2 *Receiver-site algorithm for the simplest protocol*

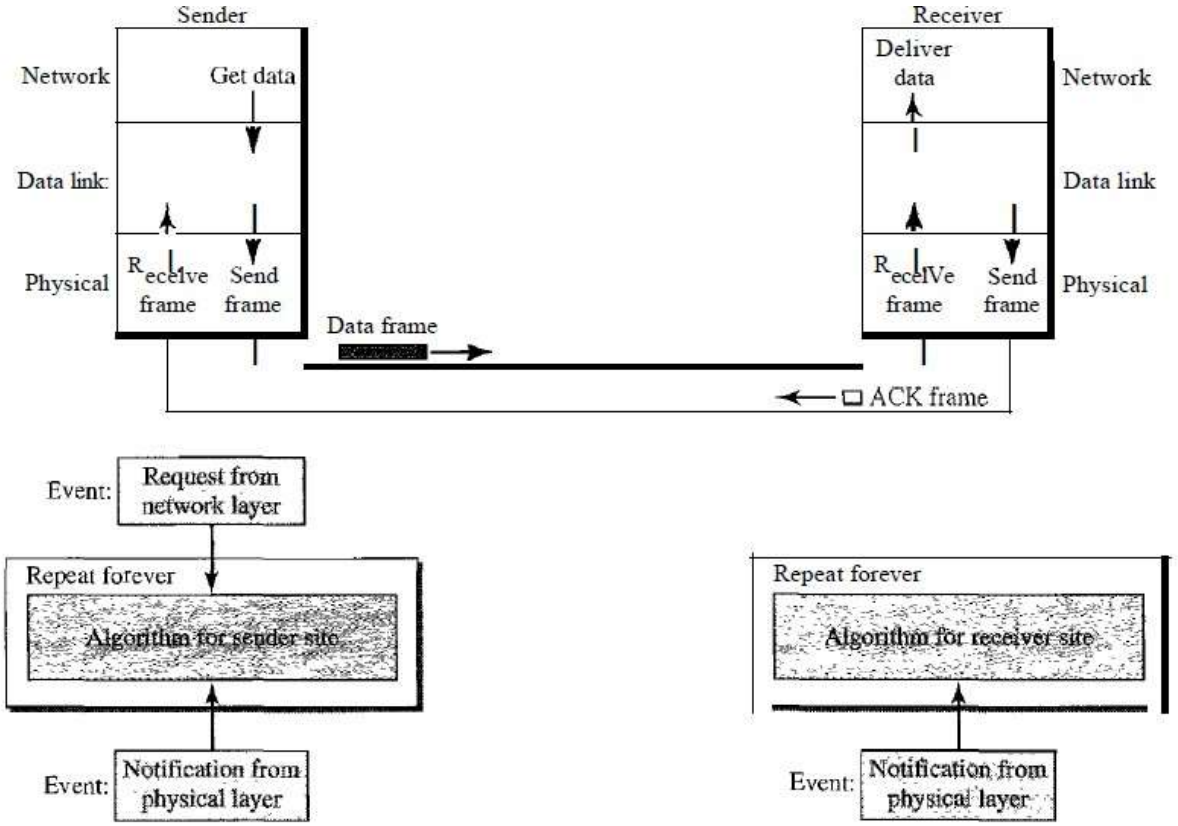
```
1  while(true)                          // Repeat forever
2  {
3    WaitForEvent()i                    II Sleep until an event occurs
4    if(Event(ArrivalNotification»      IIData frame arrived
5    {
6        ReceiveFrame()i
7        ExtractData()i
8        DeliverData ()i                //Deliver data to network layez
9    }
10 }
```

Utopian Simplex protocol(Noiseless channel)

Flow diagram



Stop and wait protocol- Half Duplex



Stop and Wait protocol

Algorithm 11.3 *Sender-site algorithm for Stop-and-Wait Protocol*

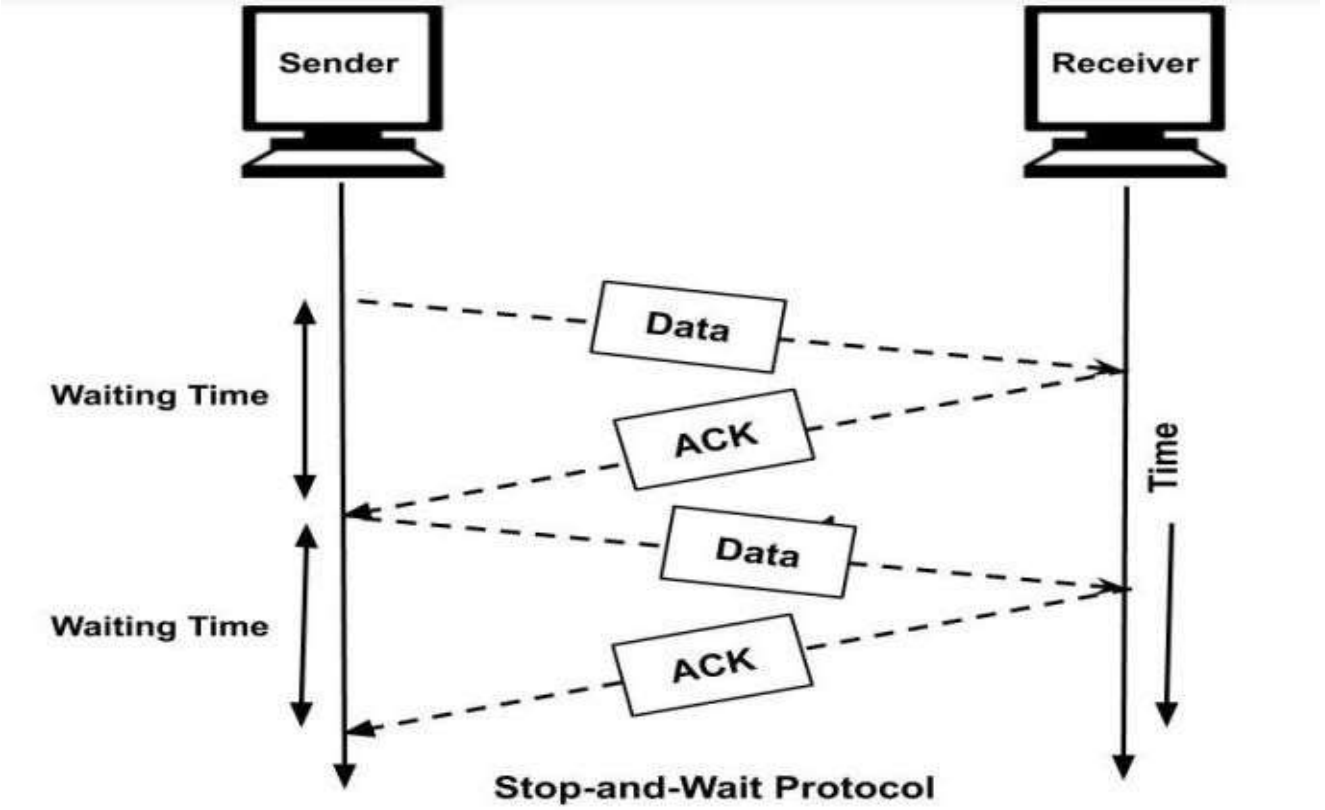
```
1  while(true)                          IIRepeat forever
2  canSend = true                       IIAllow the first frame to go
3  {
4    WaitForEvent()i                    II Sleep until an event occurs
5    if(Event(RequestToSend) AND canSend}
6    {
7       GetData()i
8       MakeFrame();
9       SendFrame()i                    I/Send the data frame
10      canSend = false;                I/cannot send until ACK arrives
11   }
12   WaitForEvent()i                    II Sleep until an event occurs
13   if(Event(ArrivalNotification)      /1 An ACK has arrived
14    {
15      ReceiveFrame();                 I/Receive the ACK frame
16      canSend = true;
17    }
```

Stop and Wait protocol

Algorithm 11.4 *Receiver-site algorithm for Stop-and-Wait Protocol*

```
1  while (true)                          IIRepeat forever
2  {
3    WaitForEvent();                     II Sleep until an event occurs
4    if(Event(ArrivalNotification)}      IIData frame arrives
5    {
6        ReceiveFrame(};
7        ExtractData(}i
8        Deliver(data};                  /IDeliver data to network layex
9        SendFrame();                    IISend an ACK frame
10   }
11 }
```

Flow diagram



Protocols for Noisy channels

- Stop and Wait ARQ
- Go back-N- ARQ
- Selective Repeat ARQ

Automatic repeat request

Sliding Window Protocols

Stop-and-Wait ARQ

- *It is the simplest flow and error control mechanism . A transmitter sends a frame then stops and waits for an acknowledgment.*
- *Stop-and-Wait ARQ has the following features:*
  - ✓ *The sending device keeps a copy of the sent frame transmitted until it receives an acknowledgment(ACK)*
  - ✓ *The sender starts a timer when it sends a frame. If an ACK is not received within an allocated time period, the sender resends it*
  - ✓ *Both frames and acknowledgment (ACK) are numbered alternately 0 and 1 (two sequence number only)*
  - ✓ *This numbering allows for identification of frames in case of duplicate transmission*

Stop-and-Wait ARQ

- *The acknowledgment number defines the number of next expected frame. (frame 0 received ACK 1 is sent)*
- *A damage or lost frame treated by the same manner by the receiver*
- *If the receiver detects an error in the received frame, or receives a frame out of order it simply discards the frame*
- *The receiver send only positive ACK for frames received safe; it is silent about the frames damage or lost.*
- *The sender has a control variable S that holds the number of most recently sent frame (0 or 1). The receiver has control variable R, that holds the number of the next frame expected (0,or 1)*

Stop-and-Wait ARQ

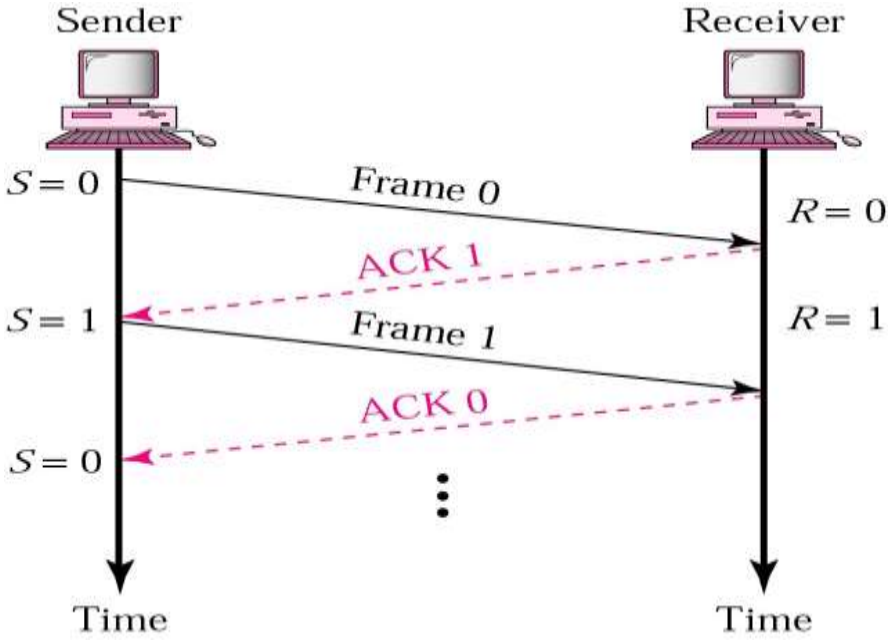
Cases of Operations:

1. *Normal operation*
2. *The frame is lost*
3. *The Acknowledgment (ACK) is lost*
4. *The Ack is delayed*

Stop-and-Wait ARQ

Normal operation

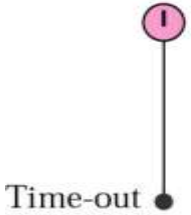
- *The sender will not send the next frame until it is sure that the current one is correctly receive*
- *sequence number is necessary to check for duplicated frames*

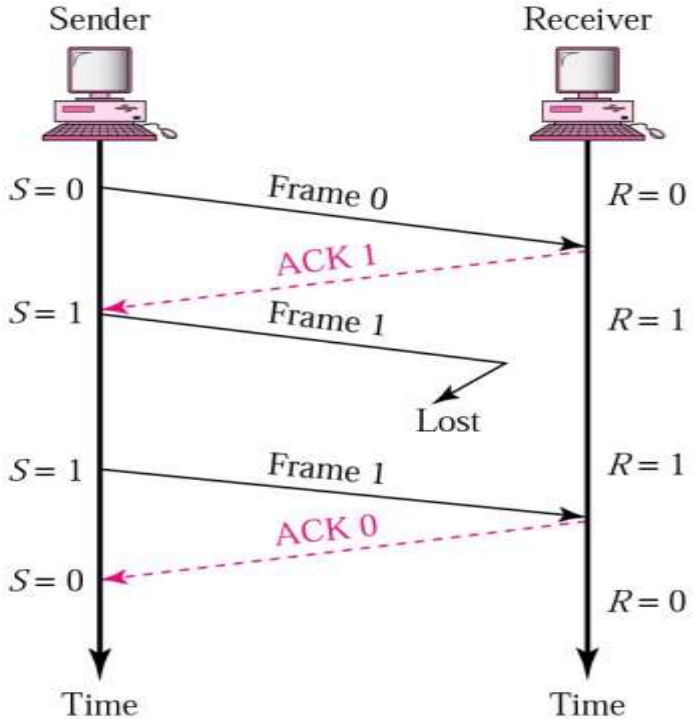


Stop and Wait ARQ

2. Lost or damaged frame

- *A damage or lost frame treated by the same manner by the receiver.*
- *No NACK when frame is corrupted / duplicate*

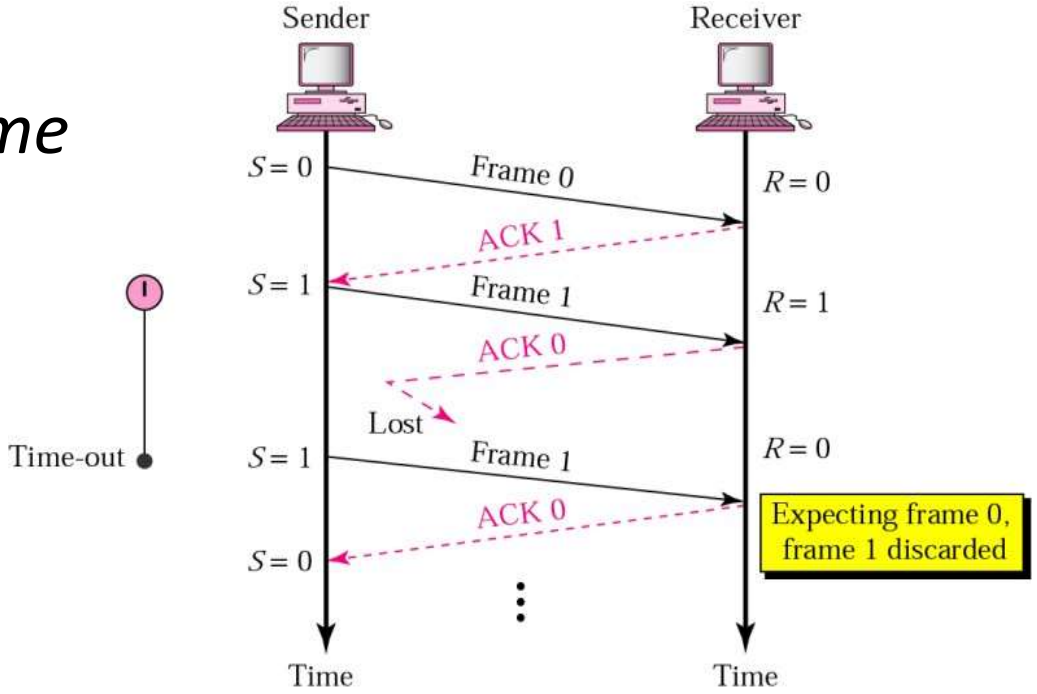




Stop-and-Wait ARQ

3. Lost ACK frame

- *Importance of frame numbering*



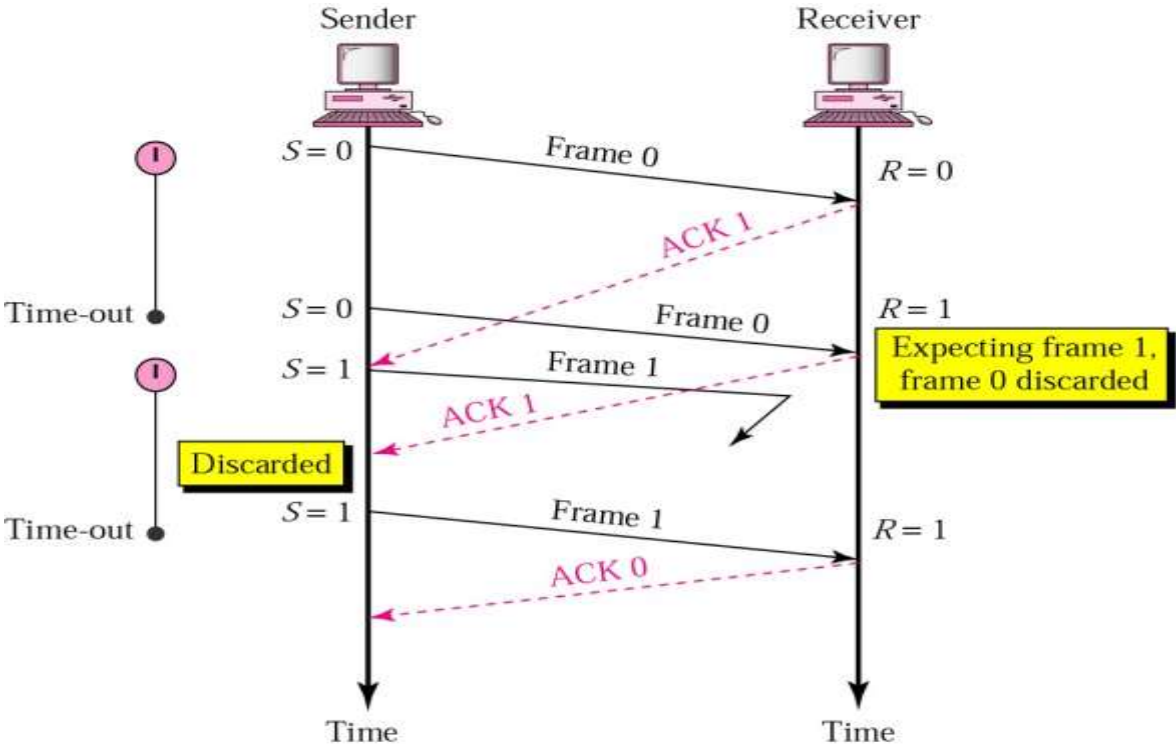
Note

In Stop and-Wait ARQ, numbering frames prevents the retaining of duplicate frames.

Stop-and-Wait ARQ

4. Delayed ACK and lost frame

- *Importance of frame numbering*

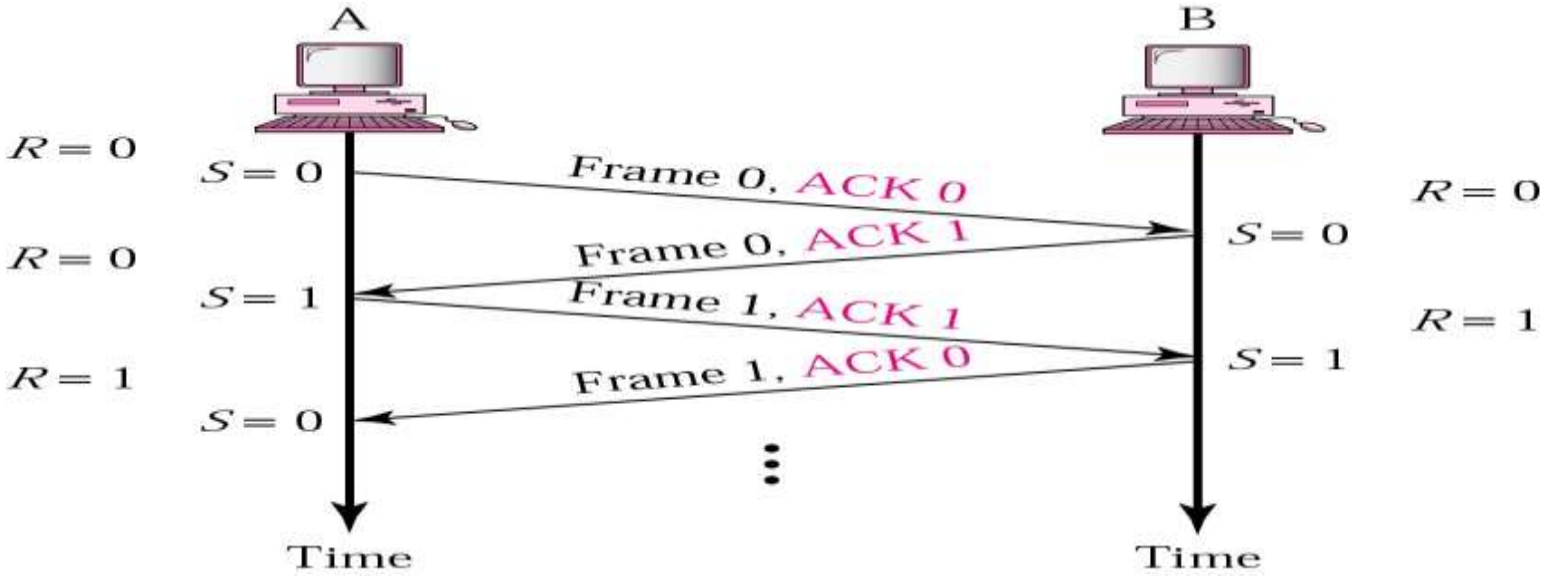


Note

Numbered acknowledgments are needed if an acknowledgment is delayed and the next frame is lost.

Piggybacking (Bidirectional transmission)

Is a method to combine a data frame with an acknowledgment.
It can save bandwidth because data frame and an ACK frame can combined into just one frame



Stop-and-Wait ARQ

After each frame sent the host must wait for an ACK

- *inefficient use of bandwidth*

To improve efficiency ACK should be sent after multiple frames

Alternatives: Sliding Window protocol

- ✓ *Go-back-N ARQ*
- ✓ *Selective Repeat ARQ*

Pipelining

Pipelining: A task is begun before the previous task has ended

- *There is no pipelining in stop and wait ARQ because we need to wait for a frame to reach the destination and be acknowledged before the next frame can be sent*
- *Pipelining improves the efficiency of the transmission*

Sliding window protocol

Sliding window protocols apply Pipelining :

- ✓ *Go-Back-N ARQ*
- ✓ *Selective Repeat ARQ*

- *Sliding window protocols improve the efficiency*
- *multiple frames should be in transition while waiting for ACK. Let more than one frame to be outstanding.*
- *Outstanding frames: frames sent but not acknowledged*
- *We can send up to W frames and keep a copy of these frames(outstanding) until the ACKs arrive.*
- *This procedures requires additional feature to be added :sliding window*

Sliding window

Sequence Numbers

- Sent frames are numbered sequentially
- Sequence number is stored in the header of the frame
- If the header of the frame allow m bits for the sequence number, the sequence numbers range from 0 to $(2^m - 1)$.

The sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits.

If $m = 3$, sequence number range from 0 to 7(8 numbers): 0, 1, 2 , 3, 4, 5, 6, 7, 0 , 1 .....

Sliding window

used to hold the unacknowledged outstanding frames (frames sent but not acknowledged)

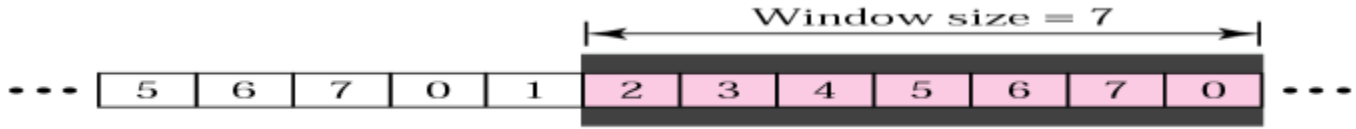
Go_Back _N ARQ

Sender sliding window

The sender window is an abstract concept defining an imaginary box of size $2^m - 1$ *(sequence numbers –1) The sender window can slide one or more slots when a valid acknowledgment arrives.*

a. Before sliding

If m = 3; sequence numbers = 8 and window size =7



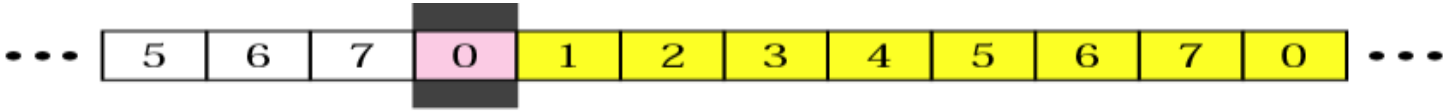
b. After sliding two frames

Acknowledged frames

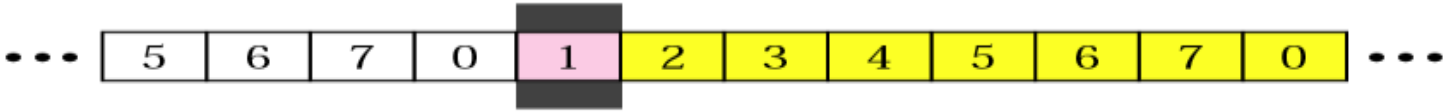
Go_Back _N ARQ

Receiver sliding window

- *The receive window is an abstract concept defining an imaginary box of size 1 with one single variable R_n.*
- *The window slides when a correct frame has arrived; sliding occurs one slot at a time.*



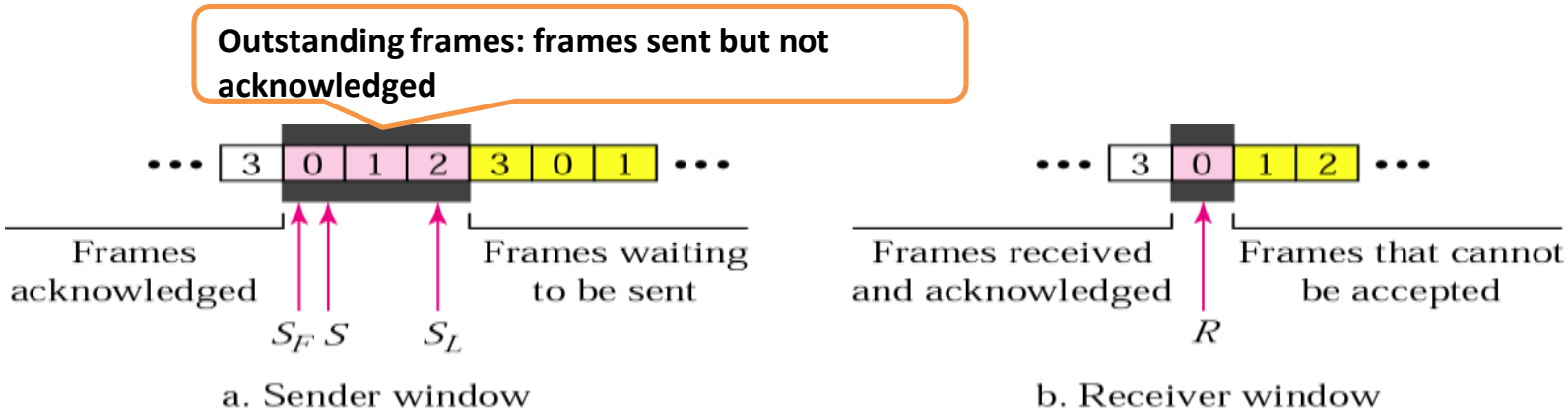
a. Before sliding



b. After sliding

Go-Back-N ARQ

control variables



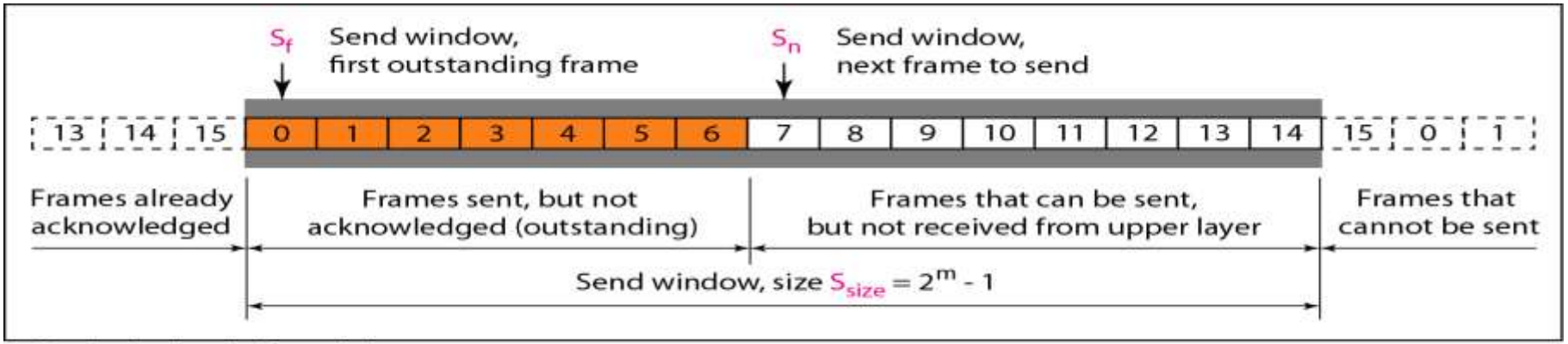
S: hold the sequence number of the recently sent frame

SF: holds sequence number of the first frame in the window

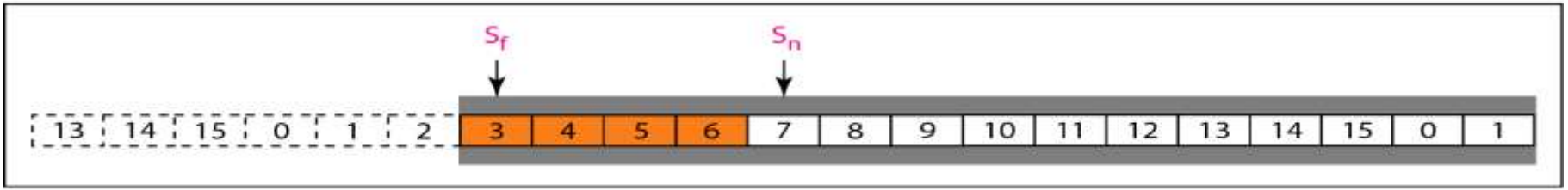
SL: holds the sequence number of the last frame

R: sequence number of the frame expected to received

Go-Back-N ARQ



a. Send window before sliding



b. Send window after sliding

Go-Back-N ARQ

In Go-Back-N ARQ we use one timer for the first outstanding frame

- *The receiver sends a positive ACK if a frame has arrived safe and in order.*
- ***if a frame is damaged or out of order ,the receiver is silent and will discard all subsequent frames***
- *When the timer of an unacknowledged frame at the sender site is expired ,* ***the sender goes back and resend all frames , beginning with the one with expired timer.(that is why the protocol is called Go-Back-N ARQ)***
- *The receiver doesn't have to acknowledge each frame received . It can send cumulative Ack for several frames*

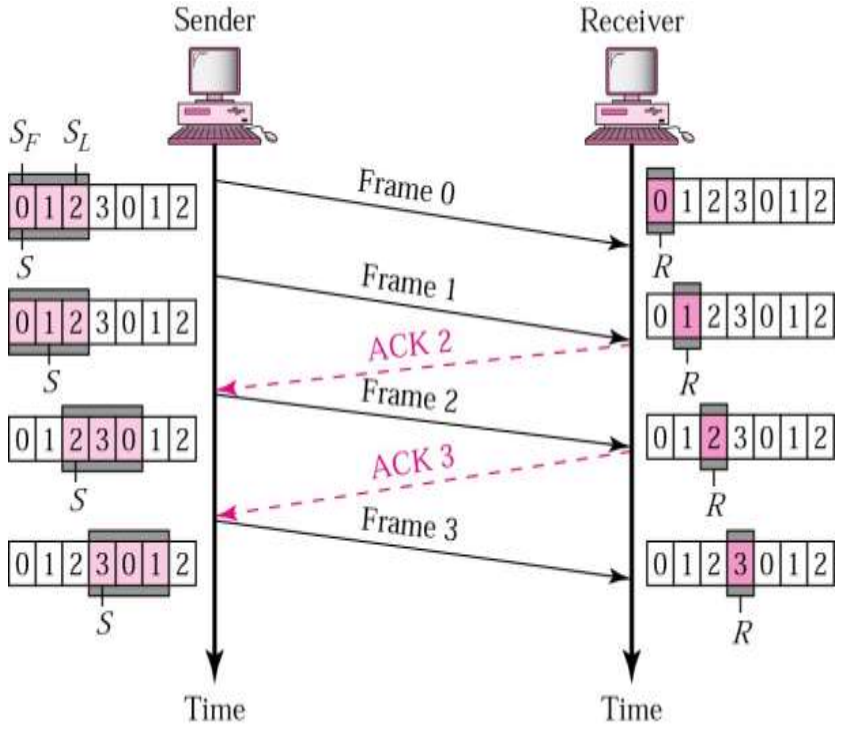
Go-Back-N ARQ

Example: The sender has sent frame 6 , and timer expires for frame 3(frame 3 has not been acknowledge); the sender goes back and resends frames 3, 4,5 and 6

Go-Back-N ARQ

Normal operation

- *How many frame scan be transmitted Without acknowledgment?*
- *ACK1 is not necessary if ACK2 is sent: Cumulative ACK*

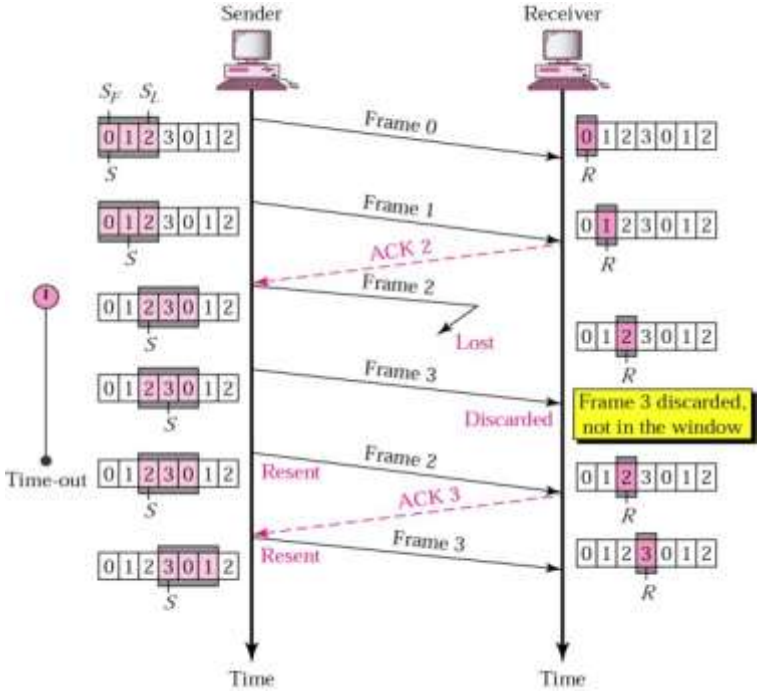


Go-Back-N ARQ

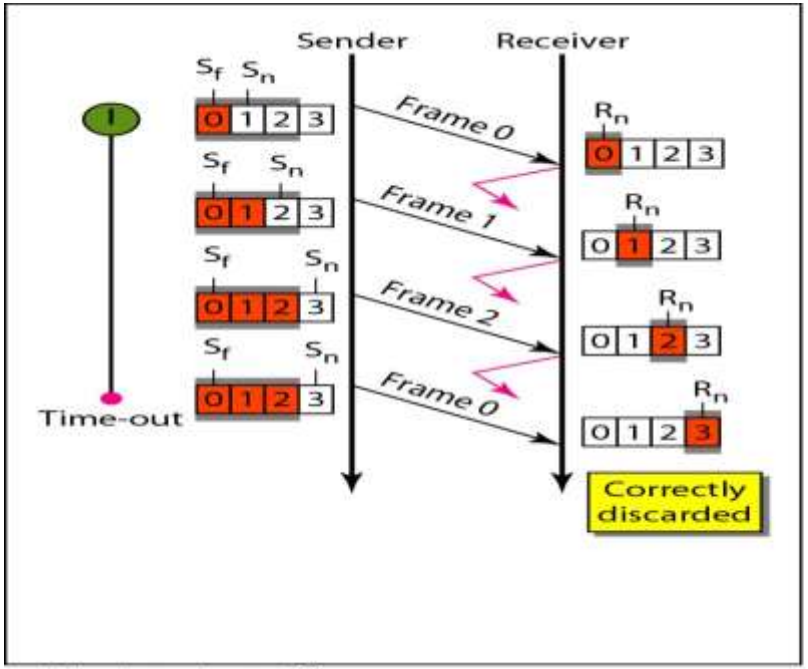
Damage or Lost Frame

Correctly received out of order packets are not Buffered

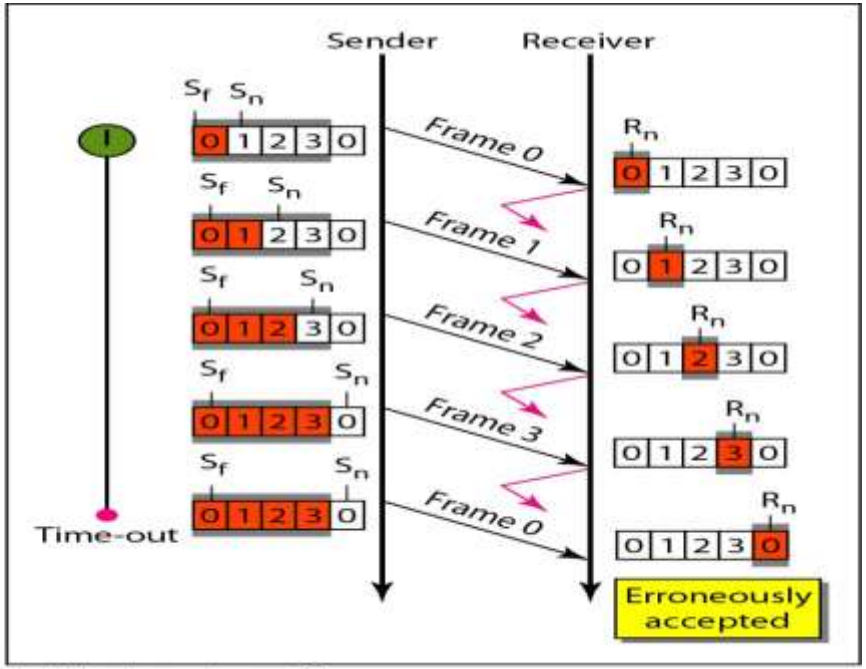
What is the disadvantage of this?



Go-Back-N ARQ



a. Window size $< 2^m$



b. Window size $= 2^m$

Go-Back-N ARQ

In Go-Back-N ARQ, the size of the sender window must be less than $2^m = (2^m - 1)$ *; the size of the receiver window is always is 1..*

Bidirectional transmission : piggybacking

As Stop-and-Wait we can use piggybacking to improve the efficiency of bidirectional transmission . Each direction needs both a sender window and a receiver window.

Note

Stop-and-Wait ARQ is a special case of Go-Back-N ARQ in which the size of the send window is 1

Selective Repeat ARQ

*Go-Back-N ARQ is inefficient of a **noisy** link.*

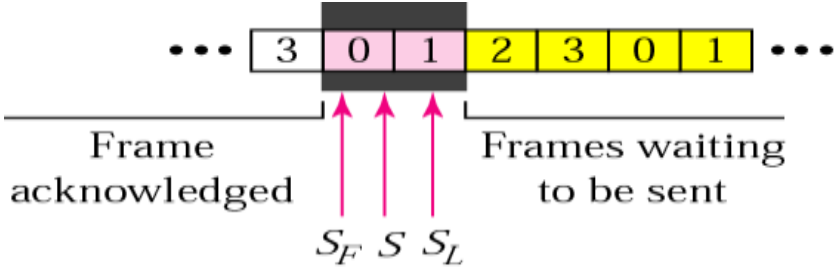
- ***In a noisy link frames have higher probability of damage** , which means the resending of multiple frames.*
- ***this resending consumes the bandwidth and slow down the transmission .***

***Solution**:*

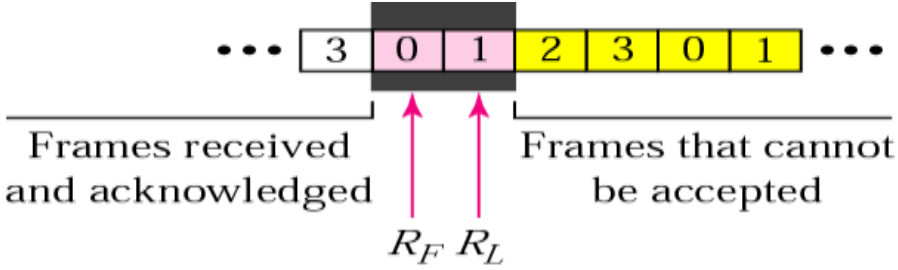
- *Selective Repeat ARQ protocol : retransmit only the damage frame*
- *It defines a negative Acknowledgement (NAK) that report the sequence number of a damaged frame before the timer expires*
- *It is more efficient for noisy link, but the processing at the receiver is more complex*

Selective Repeat ARQ

- The window size is reduced to one half of 2^m
- Sender window size = receiver window size = 2^m /2
- Window size = sequence number/2
- If m = 2, Window size = 4/2=2
- Sequence number = 0, 1, 2 , 3



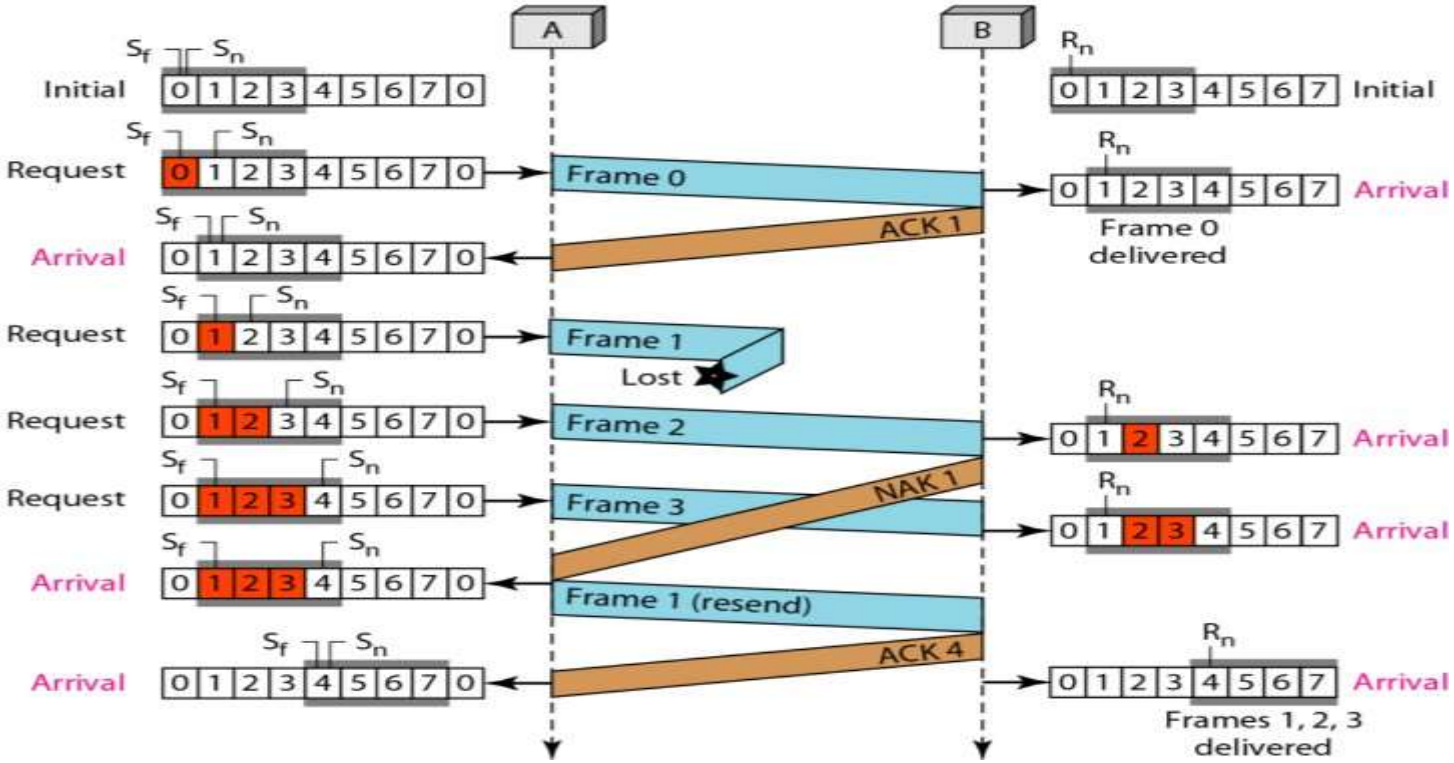
a. Sender window



b. Receiver window

Selective Repeat ARQ

Lost Frame



m=3
Sequences no=2m =8 : 0,1,2 ,3,4,5,6,7Window size =2m/2= 8/2=4

Selective Repeat ARQ

*At the receiver site we need to distinguish between the acceptance of a frame and its delivery to the network layer

* At the second arrival , frame 2 arrives and is stored and marked , but it can not be delivered because frame 1 is missing

* At the next arrival , frame 3 arrives and is marked and stored , but still none of the frames can be delivered .

*Only at the last arrival , when finally a copy of frame 1 arrives , can frames 1 , 2 , and 3 be delivered to the network layer.

*There are two conditions for the delivery of frames to the network layer: First , a set of consecutive frames must have arrived. Second, the set starts from the beginning of the window .

Selective Repeat ARQ

The next point is about the ACKs

- *Notice that only two ACKs are sent here.*
- *The first one acknowledges only the first frame;*
- *The second one acknowledges three frames.*
- *In Selective Repeat, ACKs are sent when data are delivered to the network layer. If the data belonging to n frames are delivered in one shot , only one ACK is sent for all of them.*

Note

In Selective Repeat ARQ, the size of In the sender and receiver window must be at most one-half of 2^m.

Selective Repeat ARQ

m=2