Data Link Layer

Module 2

Computer Networks

SOMAIYA VIDYAVIHAR UNIVERSITY

Somaiya TRUST

Introduction

- Data transfer with accuracy.
- Data can be corrupted during transmission.
- Some applications require that errors be detected and corrected.

Important Topics

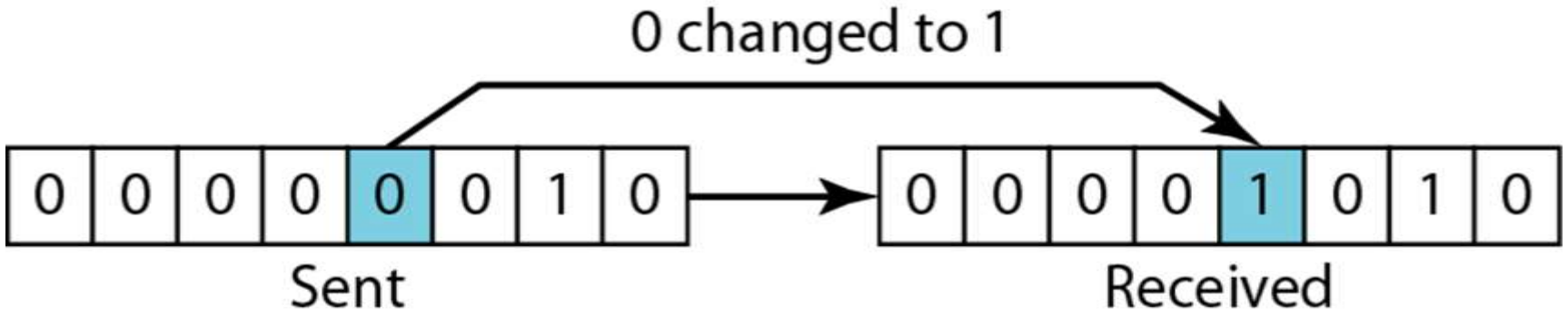
Error Correction & Detection

- Types of Errors
- Redundancy
- Détection Versus Correction
- Forward Error Correction Versus Retransmission
- Coding

Types of Errors

- Interference can change the shape of the signal
- In Single Bit error, a *0* is changed to a *1* or a *1* to a 0.
- The term ***single-bit error*** means that only 1 bit of a given data unit (such as a byte, character, or packet) is changed from 1 to 0 or from 0 to 1.

Single-Bit Error



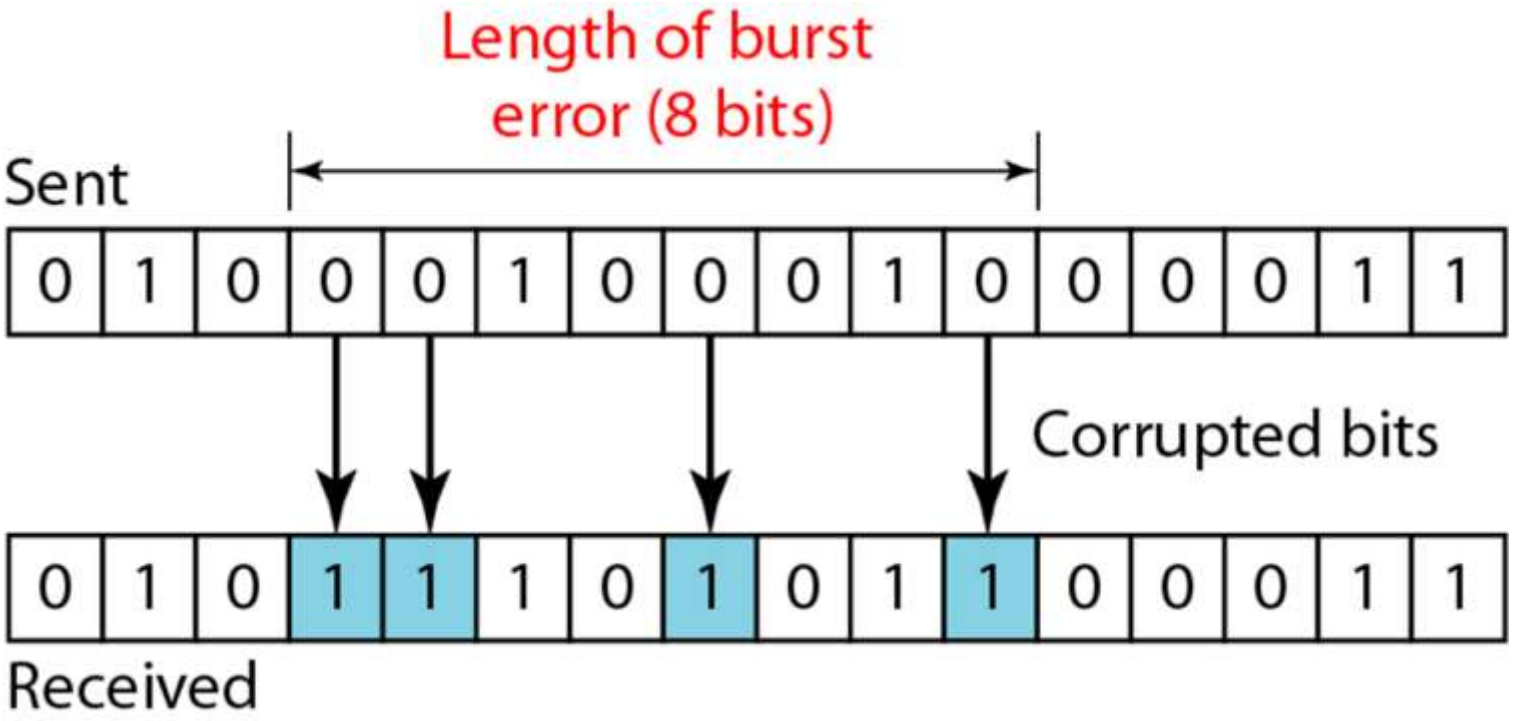
Types of Errors

- The term ***burst error*** means that 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1.

- ***Burst error*** does not necessarily mean that the errors occur in consecutive bits.

- The length of the burst is measured from the first corrupted bit to the last corrupted bit.

Burst Error of length 8



Redundancy

- Central concept in error detection and correction is Redundancy.
- To detect or correct errors, we need to send extra (redundant) bits with data.
- Redundant bits are added by the sender and removed by the receiver.

SOMAIYA VIDYAVIHAR UNIVERSITY

Detection Versus Correction

- Correction of errors is more difficult than the detection.
- In ***error detection***, we are looking only to see if any error has occurred. The answer is a simple yes or no.
- In ***error correction***, we need to know the exact number of bits that are corrupted and their location in the message.

Somaiya TRUST

Forward Error Correction Versus Retransmission

- Two main methods of error correction:
  - Forward Error Correction
  - Retransmission
- ***Forward error correction*** is the process in which the receiver tries to guess the message by using redundant bits.
- ***Retransmission*** is a technique in which the receiver detects the occurrence of an error and asks the sender to resend the message.

Coding

- Redundancy is achieved through various coding schemes
- The sender adds redundant bits through a process that creates a relationship between the redundant bits and the actual data bits.
- The receiver checks the relationships between the two sets of bits to detect or correct the errors.
- Coding schemes can be divided into two broad categories: ***Block coding*** and convolution coding

Coding

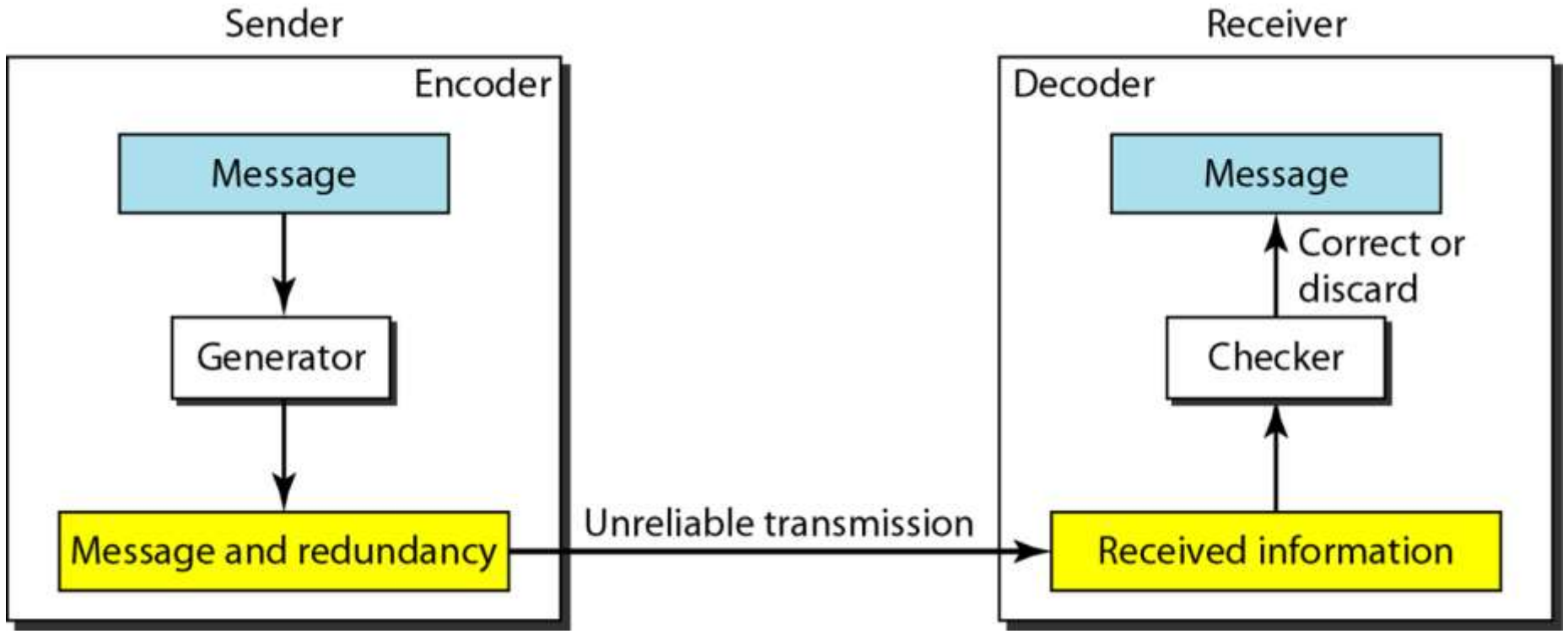
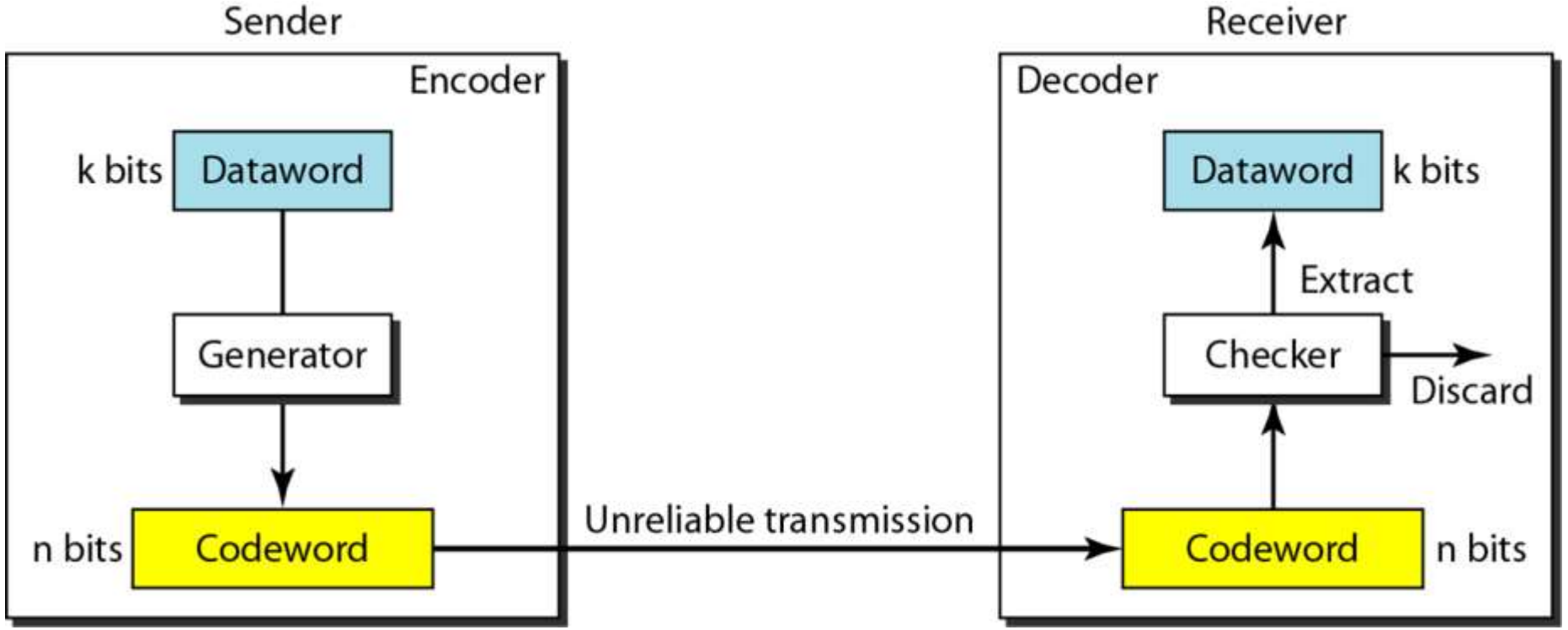


***Fig**: The structure of encoder and decoder*

Block Coding

- *In **block coding**, we divide our message into blocks, each of k bits, called **datawords**. We add r redundant bits to each block to make the length n = k + r. The resulting n-bit blocks are called **codewords**.*



Example 1

- *Let us assume that k = 2 and n = 3. Table 10.1 shows the list of datawords and codewords. Later, we will see how to derive a codeword from a dataword.*

- *Assume the sender encodes the dataword 01 as 011 and sends it to the receiver. Consider the following cases:*

1. The receiver receives 011. It is a ***valid codeword****. The receiver extracts the dataword 01 from it.*

SOMAIYA VIDYAVIHAR UNIVERSITY

Example 1(Cont.)

2. *The codeword is corrupted during transmission, and 111 is received. This is* ***not a valid codeword*** *and is discarded.*

3. *The codeword is corrupted during transmission, and 000 is received. This is a* ***valid codeword****. The receiver* ***incorrectly extracts the dataword*** *00. Two corrupted bits have made the error undetectable.*

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Example 1(Cont.)

Datawords	*Codewords*
00	000
01	011
10	101
11	110

Table 10.1 ***A code for error detection (Example 1)***

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Error Correction

- Error correction is much more difficult than error detection
- We need more redundant bits for error correction than for error detection.
- Here, the idea is the same as error detection, but the checker functions are much more complex.

Dataword	*Codeword*
00	00000
01	01011
10	10101
11	11110

Table 10.1 ***A code for error correction***

Somaiya TRUST

Hamming Distance

- One of the central concepts in coding for error control is Hamming Distance.
- The Hamming distance between two words(of same size) is the number of differences (XOR operation) between corresponding bits.
- Distance between two words x, y is represented as d(x,y)

Hamming Distance

- Let us find the Hamming distance between two pairs of words (number of differences between 2 words)

1. The Hamming distance d(000, 011) is 2 because

$$000 \oplus 011 \text{ is } 011 \text{ (two 1s)}$$

2. The Hamming distance d(10101, 11110) is 3 because

$$10101 \oplus 11110 \text{ is } 01011 \text{ (three 1s)}$$

Minimum Hamming Distance

- Measurement used for designing a code is the minimum Hamming distance.
- The minimum Hamming distance is the smallest Hamming distance between all possible pairs in a set of codewords.
- d_{min} is used to define the minimum Hamming distance in a coding scheme.

Example 2

- Find the minimum Hamming distance of the coding scheme in Table.

Datawords	*Codewords*
00	000
01	011
10	101
11	110

Solution:

We first find all Hamming distances.

$d(000, 011) = 2$ $d(000, 101) = 2$ $d(000, 110) = 2$ $d(011, 101) = 2$
$d(011, 110) = 2$ $d(101, 110) = 2$

The d_{min} in this case is 2.

Example 3

- Find the minimum Hamming distance of the coding scheme in Table.

Dataword	*Codeword*
00	00000
01	01011
10	10101
11	11110

Solution:

We first find all the Hamming distances.

$d(00000, 01011) = 3$ $d(00000, 10101) = 3$ $d(00000, 11110) = 4$
$d(01011, 10101) = 4$ $d(01011, 11110) = 3$ $d(10101, 11110) = 3$

The d_{min} in this case is 3.

SOMAIYA VIDYAVIHAR UNIVERSITY

Minimum Hamming distance for Error Detection

To guarantee the **detection** of up to *s errors* in all cases, the minimum Hamming distance in a block code must be

$$d_{min} = s + 1.$$

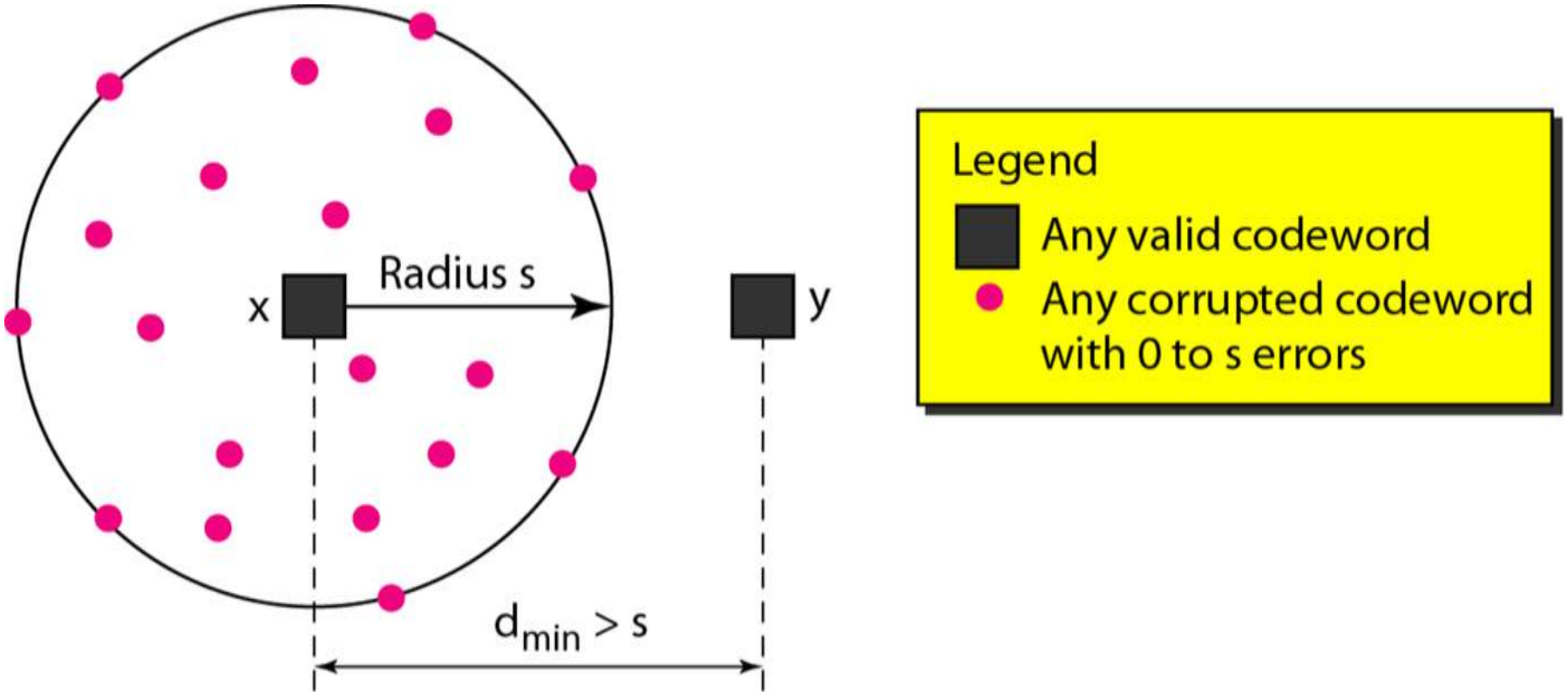
Minimum Hamming Distance for Error Correction

To guarantee **correction** of up to **t** errors in all cases, the minimum Hamming distance in a block code must be

$$d_{min} = 2t + 1.$$

Somaiya TRUST

Fig: Geometric concept for finding d_{min} in error detection



Example 4

A code scheme has a Hamming distance dmin = 4. What is the error detection and correction capability of this scheme?

Solution:

This code guarantees the detection of up to three errors (s = 3), but it can correct up to one error. In other words, if this code is used for error correction, part of its capability is wasted. Error correction codes need to have an odd minimum distance (3, 5, 7, . . .).

SOMAIYA VIDYAVIHAR UNIVERSITY

Linear Block Codes

A linear block code is a code in which the exclusive OR (addition modulo-2) of two valid codewords creates another valid codeword.

Minimum Distance for Linear Block Codes:

The minimum Hamming distance is the number of 1s in the nonzero valid codeword with the smallest number of 1s.

Example: *refer previous examples*

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Some Linear Block Codes

1. Simple Parity-Check Code
2. Hamming Codes

Simple Parity-Check Code:

- a k-bit dataword is changed to an n-bit codeword where n = k + 1.

- The extra bit, called the parity bit, is selected to make the total number of 1s in the codeword even/odd.

- The minimum Hamming distance for this category is d_{min} =2, which means that the code is a single-bit error-detecting code; it cannot correct any error.

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Example

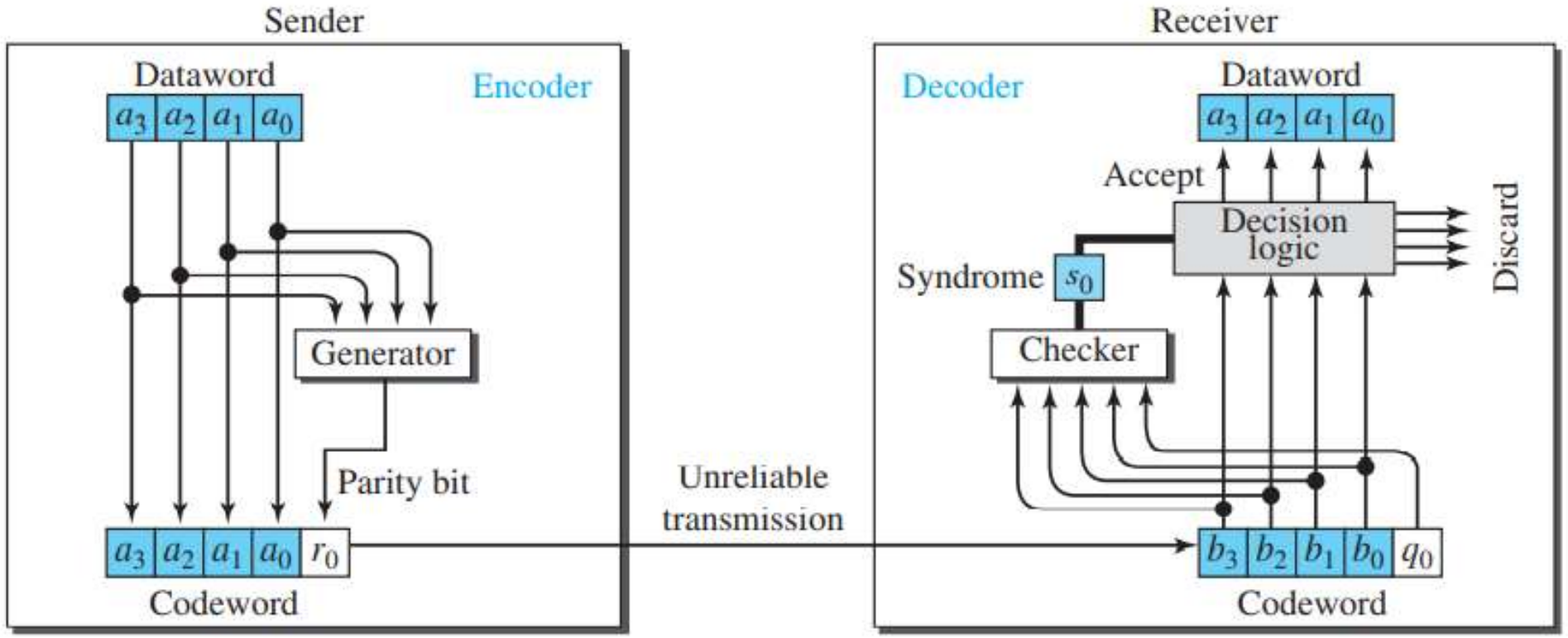
Table 10.2 *Simple parity-check code C(5, 4)*

Dataword	*Codeword*	*Dataword*	*Codeword*
0000	**00000**	1000	**10001**
0001	**00011**	1001	**10010**
0010	**00101**	1010	**10100**
0011	**00110**	1011	**10111**
0100	**01001**	1100	**11000**
0101	**01010**	1101	**11011**
0110	**01100**	1110	**11101**
0111	**01111**	1111	**11110**

Somaiya TRUST

Parity Check Code

Figure 10.4 *Encoder and decoder for simple parity-check code*



Parity Check Code

Assume the sender sends the dataword 1011. The codeword created from this dataword is 10111, which is sent to the receiver. We examine five cases:

1. No error occurs; the received codeword is 10111. The syndrome is 0. The dataword 1011 is created.

2. One single-bit error changes a1. The received codeword is 10011. The syndrome is 1. No dataword is created.

3. One single-bit error changes r0. The received codeword is 10110. The syndrome is 1. No dataword is created.

SOMAIYA VIDYAVIHAR UNIVERSITY

Parity Check Code

4. An error changes r0 and a second error changes a3. The received codeword is 00110. The syndrome is 0. The dataword 0011 is created at the receiver. The simple parity-check decoder cannot detect an even number of errors. The errors cancel each other out and give the syndrome a value of 0.

5. Three bits—a3, a2, and a1—are changed by errors. The received codeword is 01011. The syndrome is 1. The dataword is not created.

Note: A parity-check code can detect an odd number of errors.

Somaiya TRUST

Hamming Code

- Designed with d_{min} =3 i.e. detect up to 2 errors and correct up to 1-bit error.
- For example, if m =3, then n=7 and k=4. This is a Hamming code C(7, 4) with d_{min} =3.

SOMAIYA VIDYAVIHAR UNIVERSITY

Hamming Code C(7,4)

Bit position		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11	p16	d12	d13	d14	d15
Parity bit coverage	p1	✗		✗		✗		✗		✗		✗		✗		✗		✗		✗	
	p2		✗	✗			✗	✗			✗	✗			✗	✗			✗	✗	
	p4				✗	✗	✗	✗					✗	✗	✗	✗					✗
	p8								✗	✗	✗	✗	✗	✗	✗	✗					
	p16																✗	✗	✗	✗	✗

Let n be the number of information or data bits, then the number of redundant bits P is determined from the following formula,

$$2^P \geq n + P + 1$$

https://en.wikipedia.org/wiki/Hamming_code

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Cyclic Codes

- Cyclic codes are special linear block codes.
- If a codeword is cyclically shifted (rotated), the result is another codeword.
- **Cyclic Redundancy Check**:
  - A subset of cyclic codes, used for error detection.
  - Used in networks such as LANs and WANs.

Somaiya TRUST

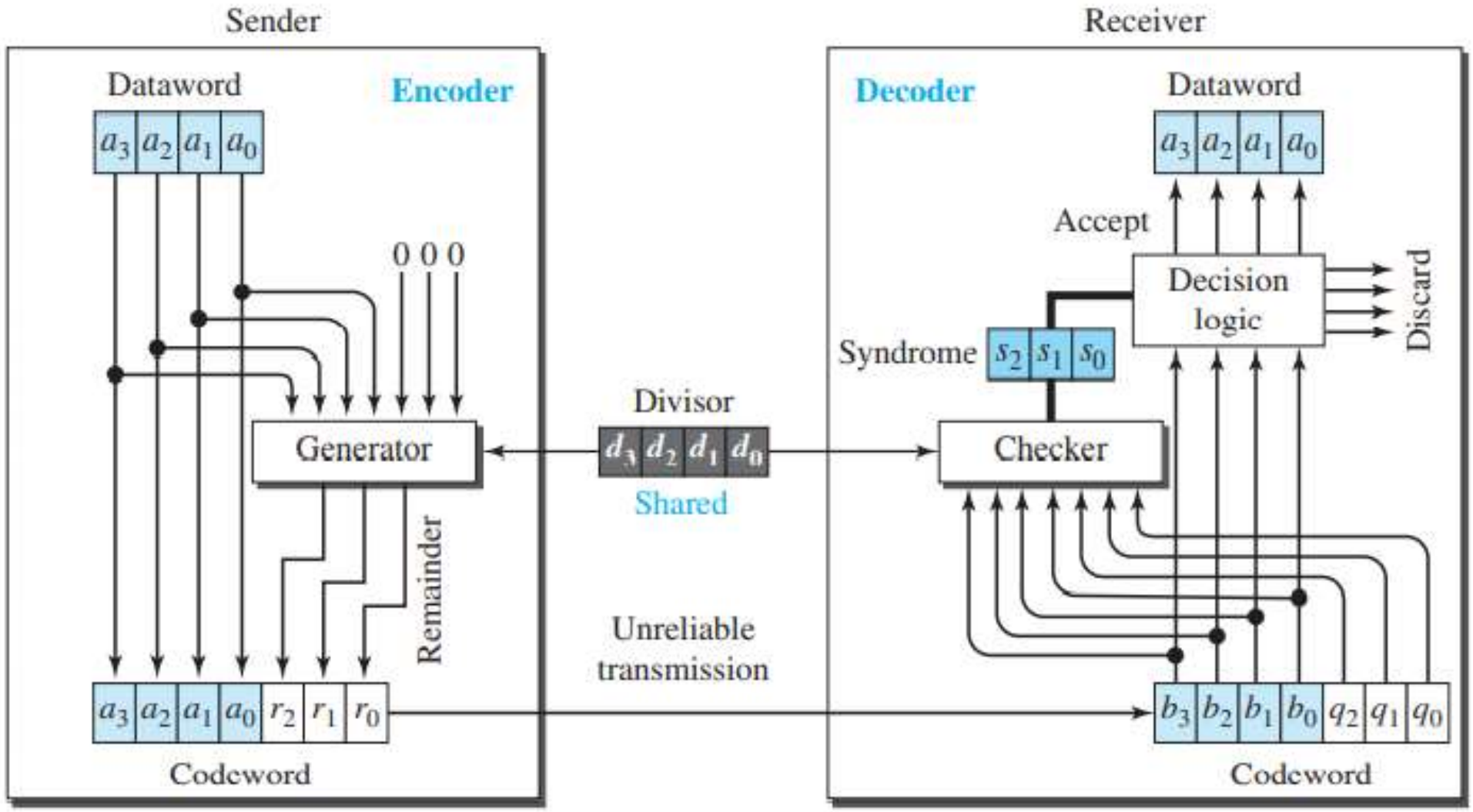
SOMAIYA VIDYAVIHAR UNIVERSITY

CRC Code with C(7,4)

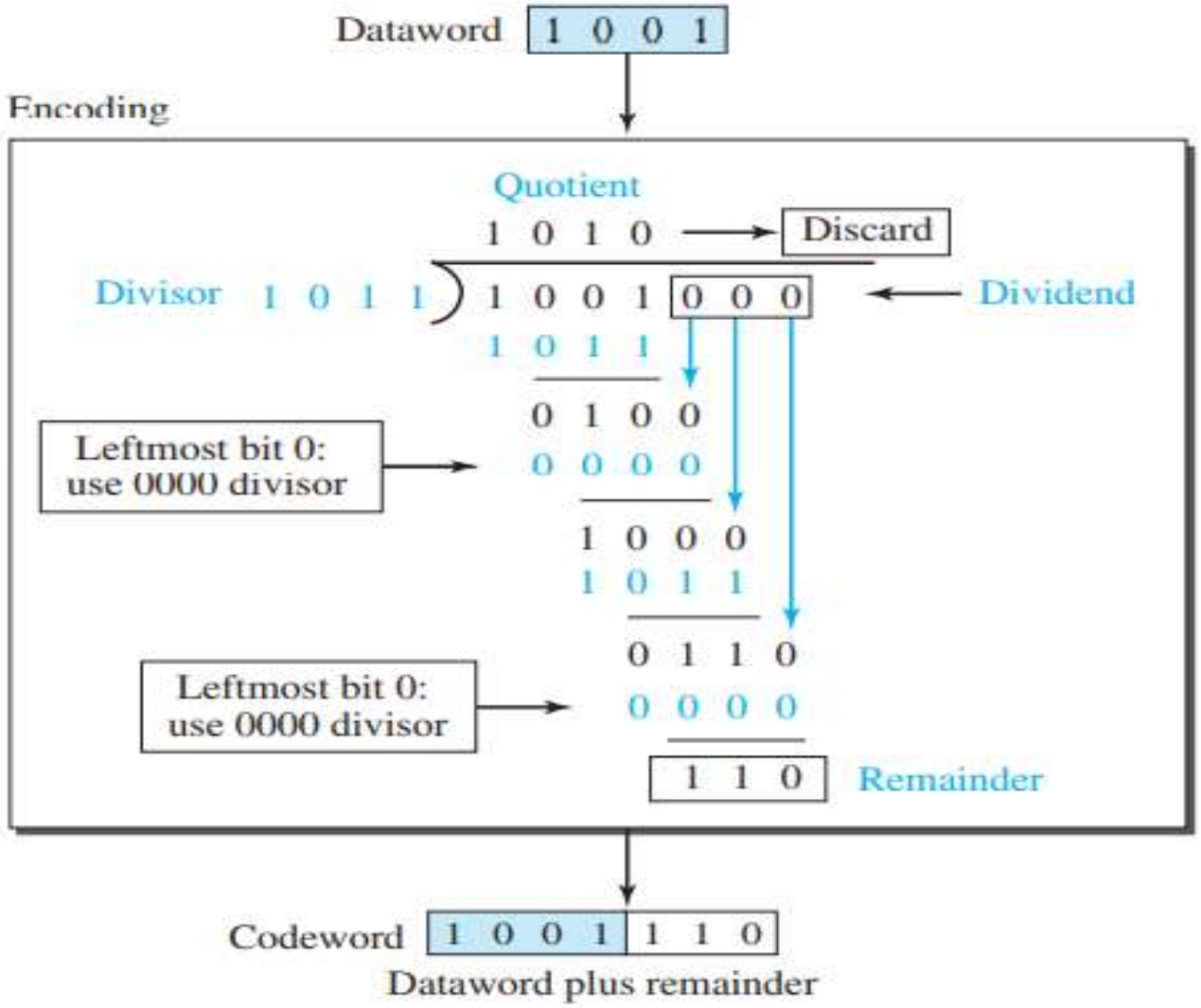
Dataword	*Codeword*	*Dataword*	*Codeword*
0000	0000000	1000	1000101
0001	0001011	1001	1001110
0010	0010110	1010	1010011
0011	0011101	1011	1011000
0100	0100111	1100	1100010
0101	0101100	1101	1101001
0110	0110001	1110	1110100
0111	0111010	1111	1111111

Somaiya TRUST

CRC Code encoder and decoder



Division in CRC encoder

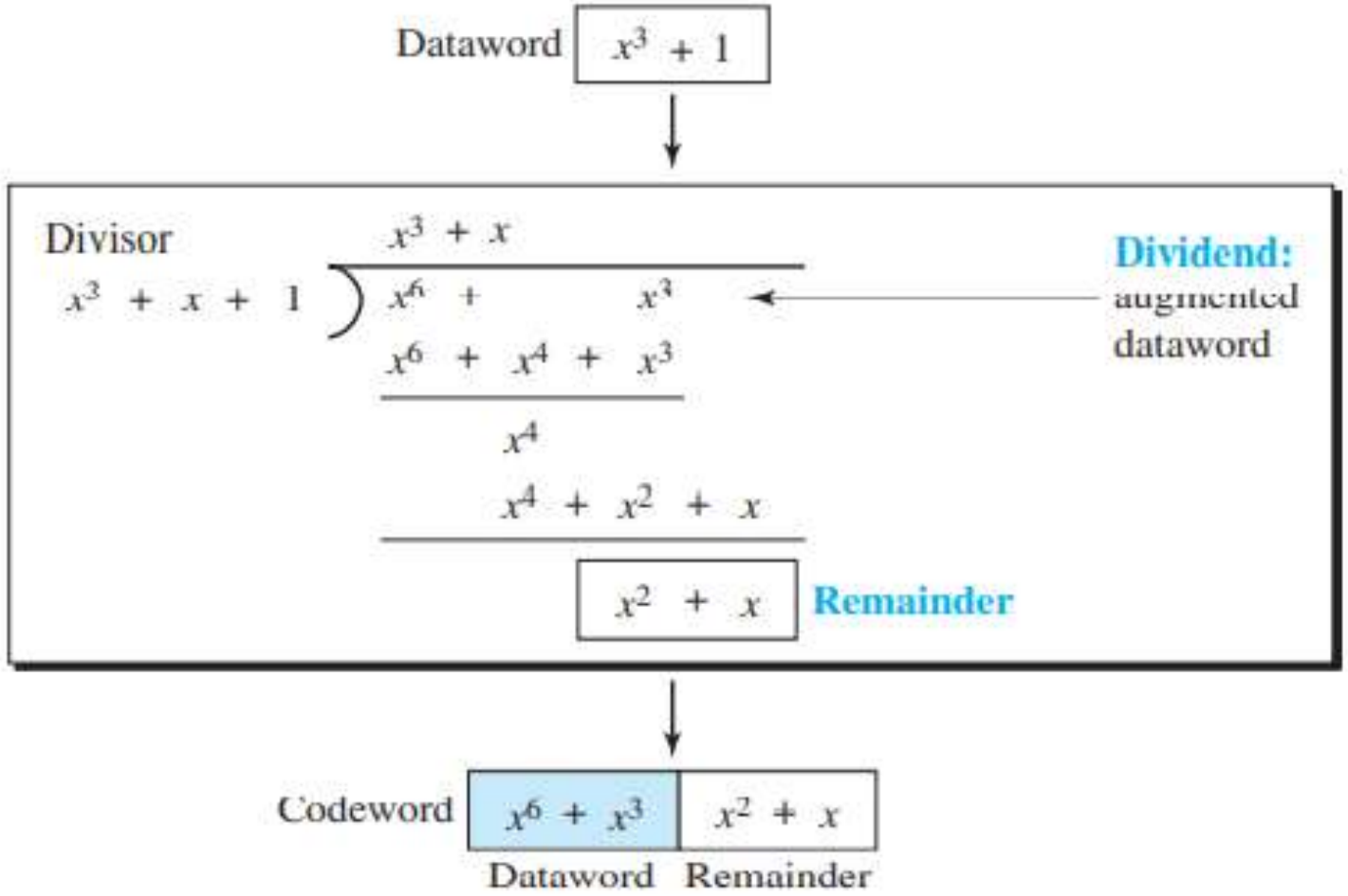


Note:

Multiply: AND
Subtract: XOR

SOMAIYA VIDYAVIHAR UNIVERSITY

CRC Division using polynomial



The divisor in a cyclic code is normally called the generator polynomial or simply the generator

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Standard polynomials

Name	*Polynomial*	*Used in*
CRC-8	$x^8 + x^2 + x + 1$ **100000111**	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$ **11000110101**	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$ **1000100000100001**	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ **10000010011000001000111011011011 1**	LANs

The divisor in a cyclic code is normally called the generator polynomial or simply the generator

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Checksum

Internet Checksum:

- Traditionally, the Internet has used a 16-bit checksum

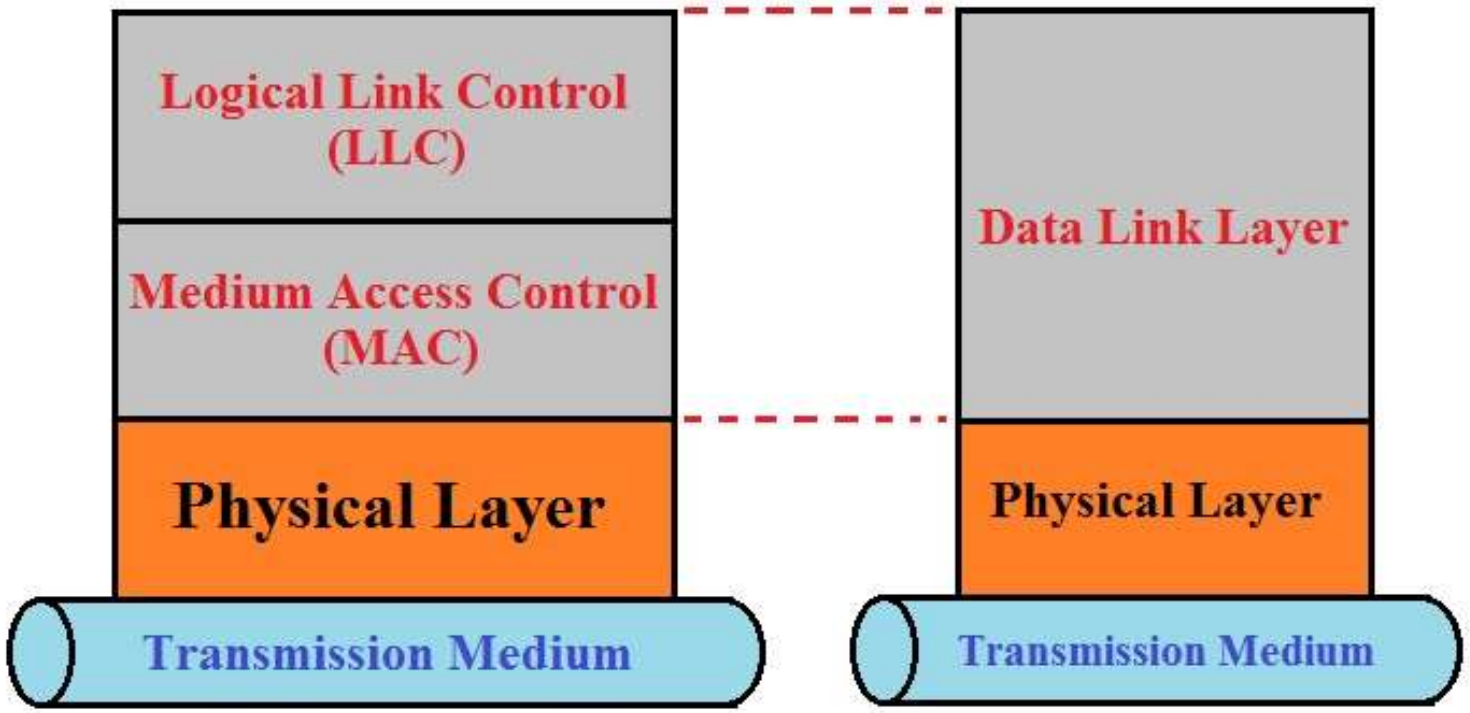
- The sender and the receiver follow the steps depicted in Table

Sender	*Receiver*
1. The message is divided into 16-bit words.	1. The message and the checksum are received.
2. The value of the checksum word is initially set to zero.	2. The message is divided into 16-bit words.
3. All words including the checksum are added using one's complement addition.	3. All words are added using one's complement addition.
4. The sum is complemented and becomes the checksum.	4. The sum is complemented and becomes the new checksum.
5. The checksum is sent with the data.	5. If the value of the checksum is 0, the message is accepted; otherwise, it is rejected.

Somaiya TRUST

Data Link Layer

Data Link Layer is Divided into two sub layers:



SOMAIYA VIDYAVIHAR UNIVERSITY

Data Link Layer

Logical Link Control (LLC Sublayer):

- Controls frame synchronization, flow control an error Checking.

Media Access Control (MAC Sublayer):

- Responsible for moving data packets from one interface to another across the shared channel.

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Framing

The data link layer needs to pack bits into frames, so that each frame is distinguishable from another.

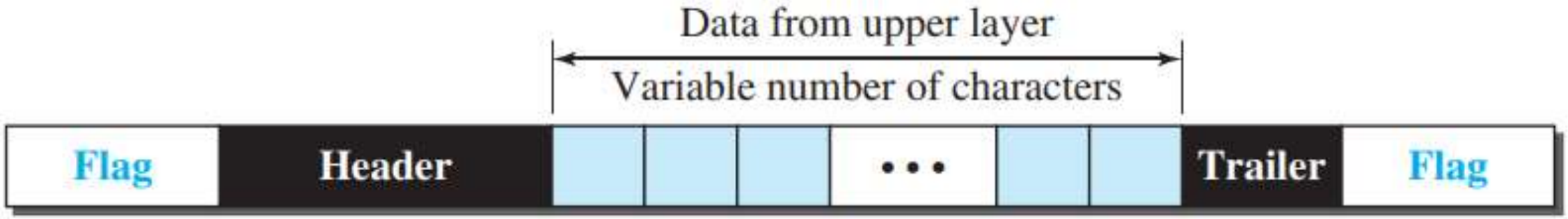
Example: Postal system: letter and envelope(delimiter)

- Fixed Size Framing: Size itself act as a delimiter
- Variable Size Framing: need a way to define end of one frame and beginning of next.
  - Character (Byte) Oriented Framing
  - Bit Oriented Framing

Somaiya TRUST

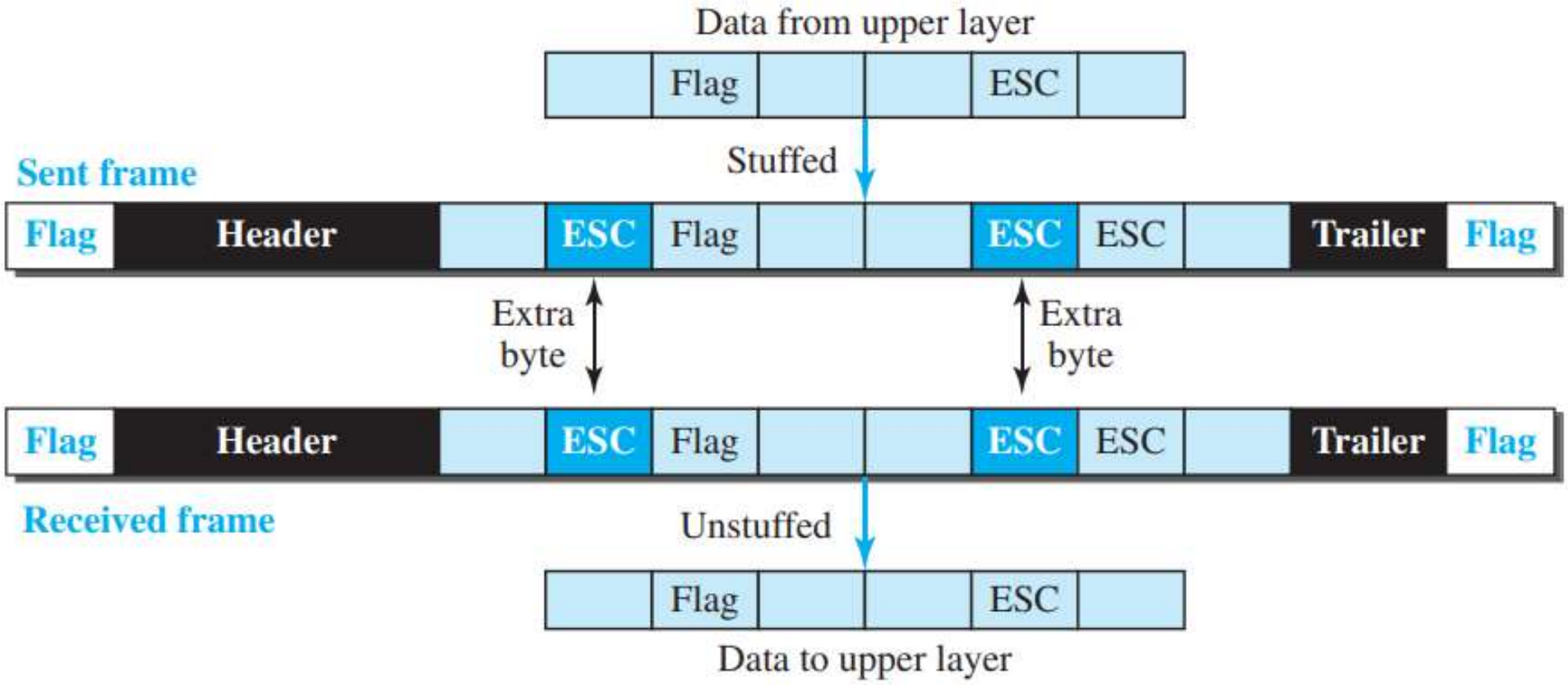
Character Oriented Framing

- In Character (Byte) oriented Framing, data to be carried are 8-bit characters.
- Header and Trailer are also multiples of 8-bit.
- To separate one from the next, flag is added at the beginning and end of the frame.



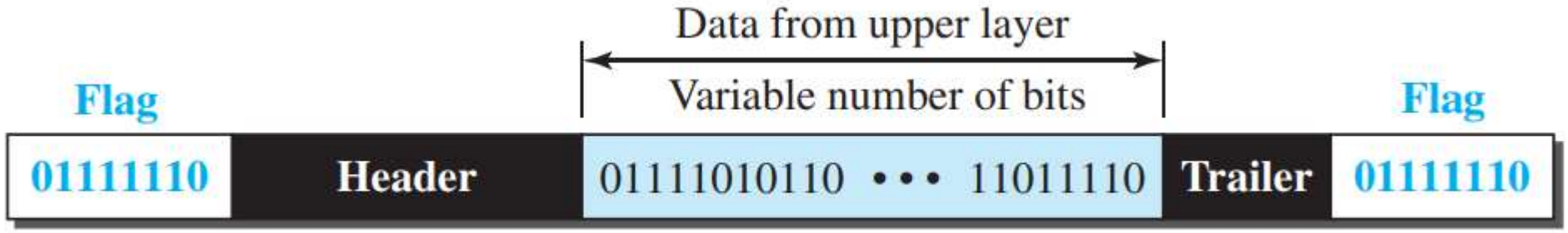
Character Oriented Framing

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



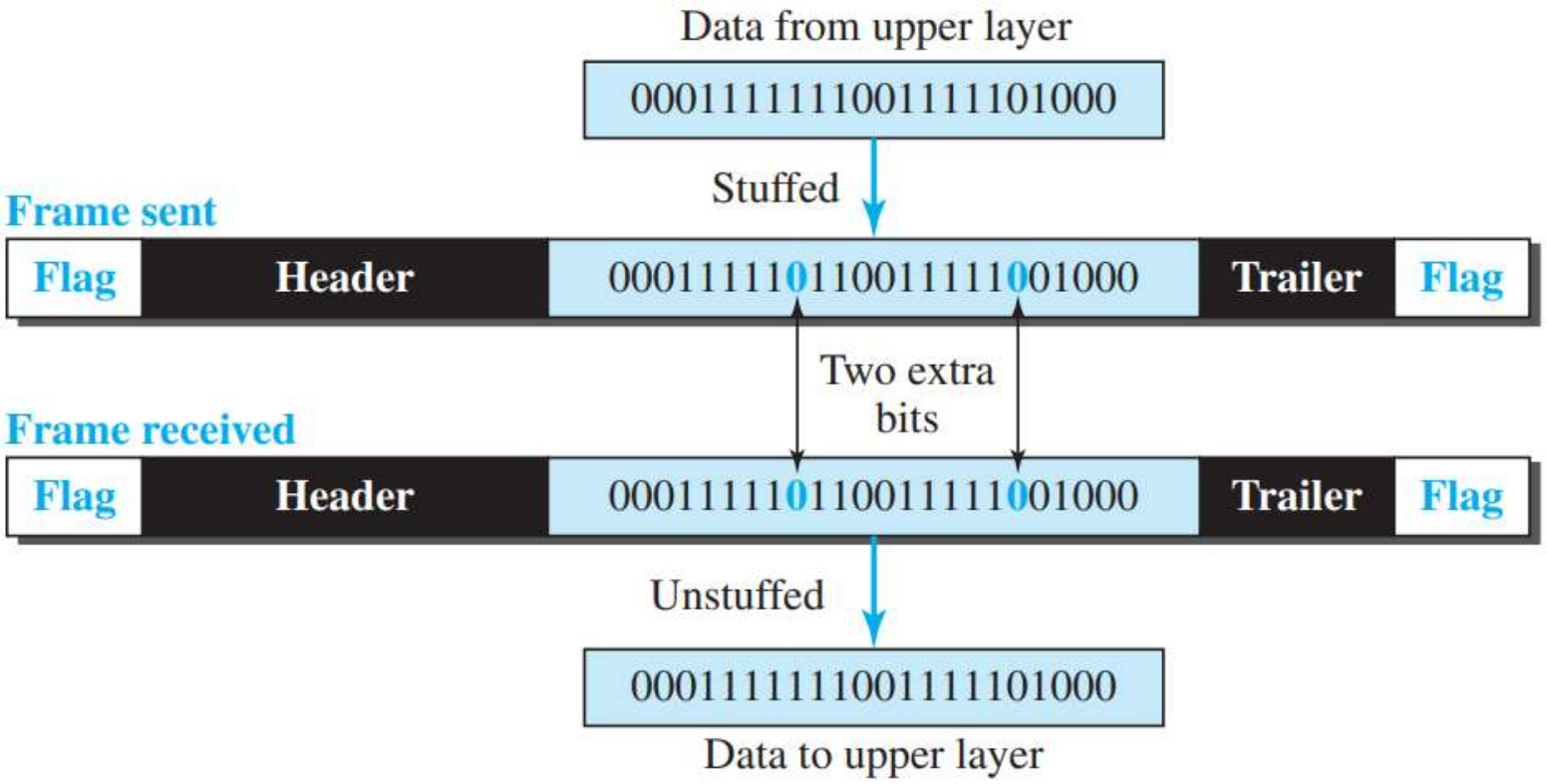
Bit-Oriented Framing

- In Bit-oriented Framing, a special 8-bit pattern flag, **01111110**, is used as the delimiter to define the beginning and the end of the frame.



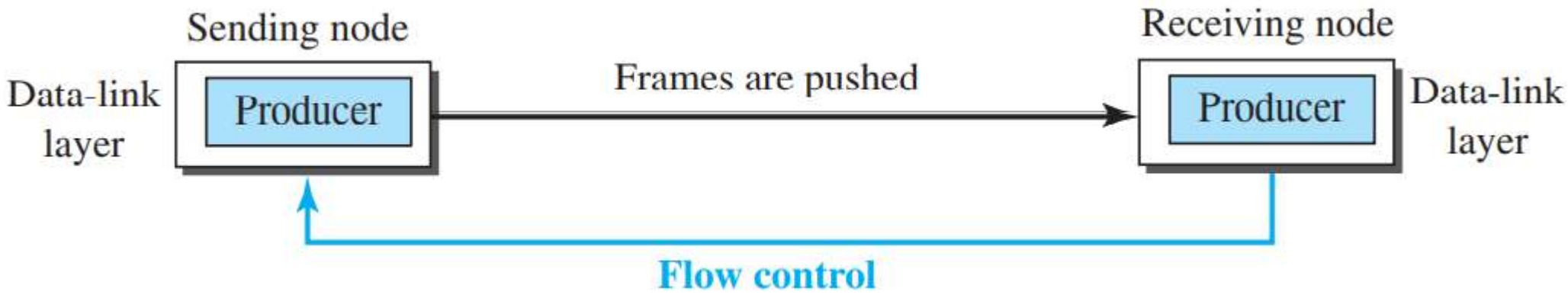
Bit Stuffing and unstuffing

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



Flow Control and Error Control

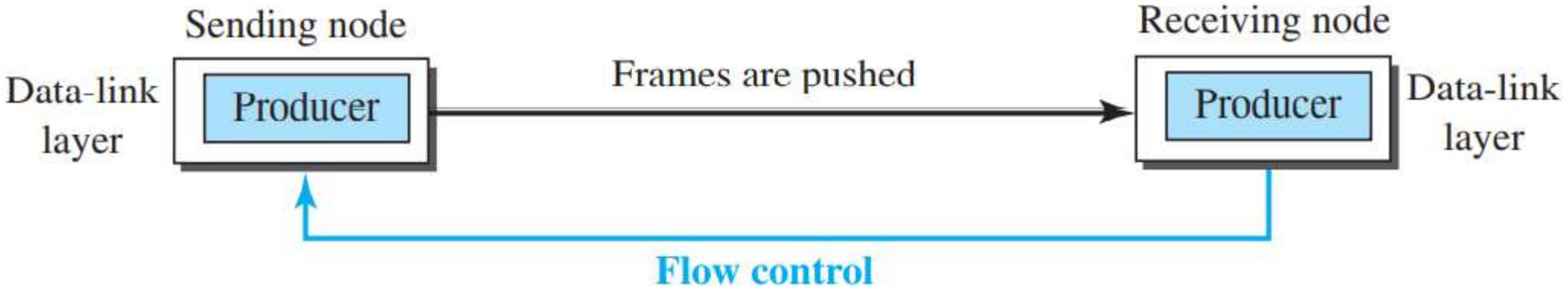
One of the responsibilities of the data link control sublayer is flow and error control at the data-link layer. Collectively, these functions are known as data link control. waiting for acknowledgment.



Flow Control and Error Control

Flow Control at Data Link Layer: Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before.

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

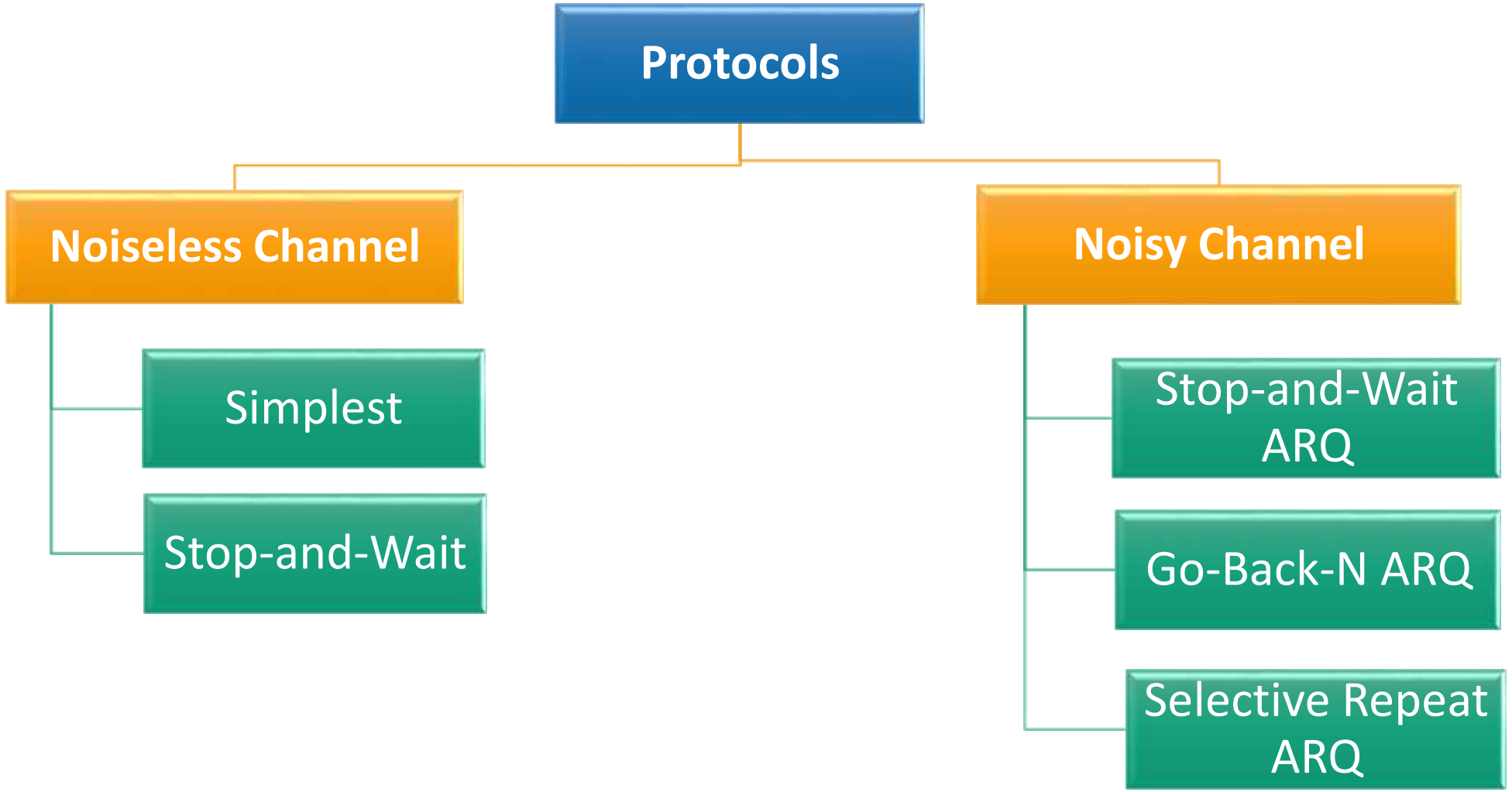


SOMAIYA VIDYAVIHAR UNIVERSITY

Flow Control and Error Control

- Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment.
- Flow of data must not allow to overwhelm the receiver.
- Receiving device have limited speed at which it can process incoming data and limited amount of memory to store incoming data.
- Error control is both error detection and correction.
- Error control in the data link layer is based on automatic repeat request, which is the retransmission of data.

Somaiya TRUST


Protocols
Noiseless Channel
Noisy Channel
Simplest
Stop-and-Wait
Stop-and-Wait ARQ
Go-Back-N ARQ
Selective Repeat ARQ

SOMAIYA VIDYAVIHAR UNIVERSITY

Noiseless Channel

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.

- Simplest Protocol
- Stop-and-Wait Protocol

Somaiya TRUST

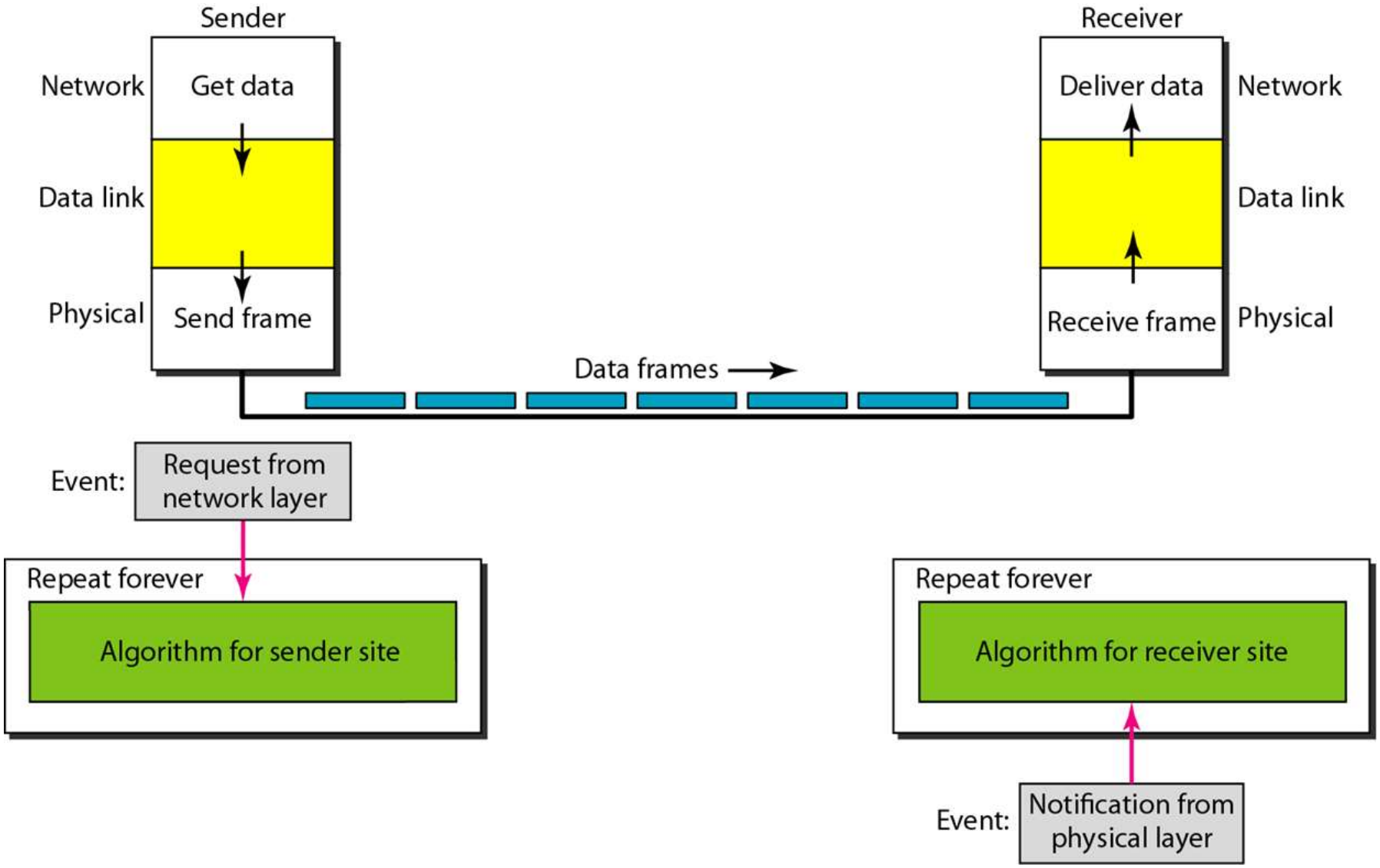
SOMAIYA VIDYAVIHAR UNIVERSITY

Simplest Protocol

- Cannot be used in real life
- No flow and error control
- Receiver can handle any frame with negligible processing time.
- Receiver can never be overwhelmed with incoming frames.

Somaiya TRUST

Simplest Protocol



Sender-site algorithm for the simplest protocol

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

Receiver-site algorithm for the simplest protocol

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

Simplest Protocol *(Example)*

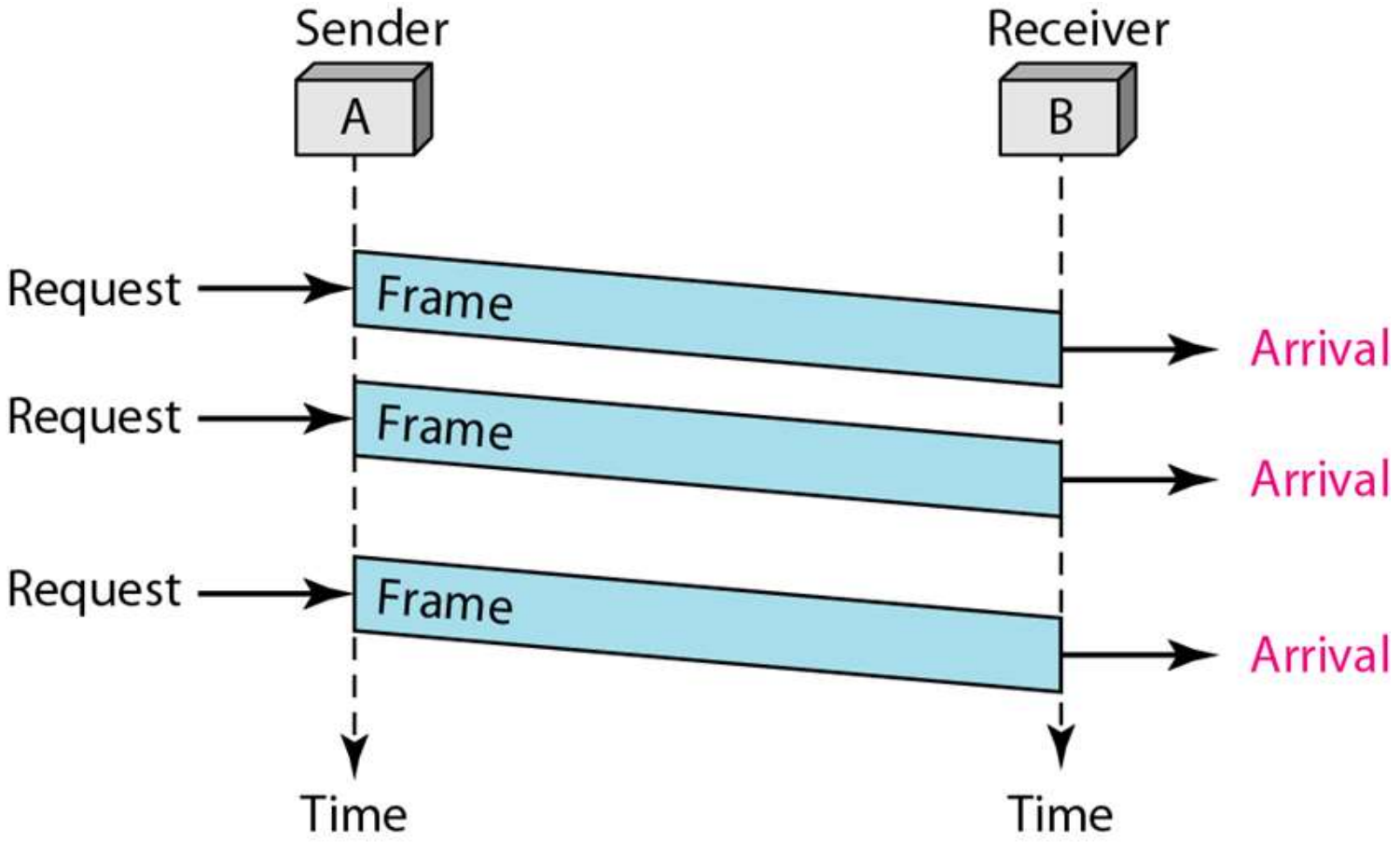
***Figure** shows an example of communication using this protocol.*

- *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.

Simplest Protocol *(Example)*

Figure: *Flow Diagram*



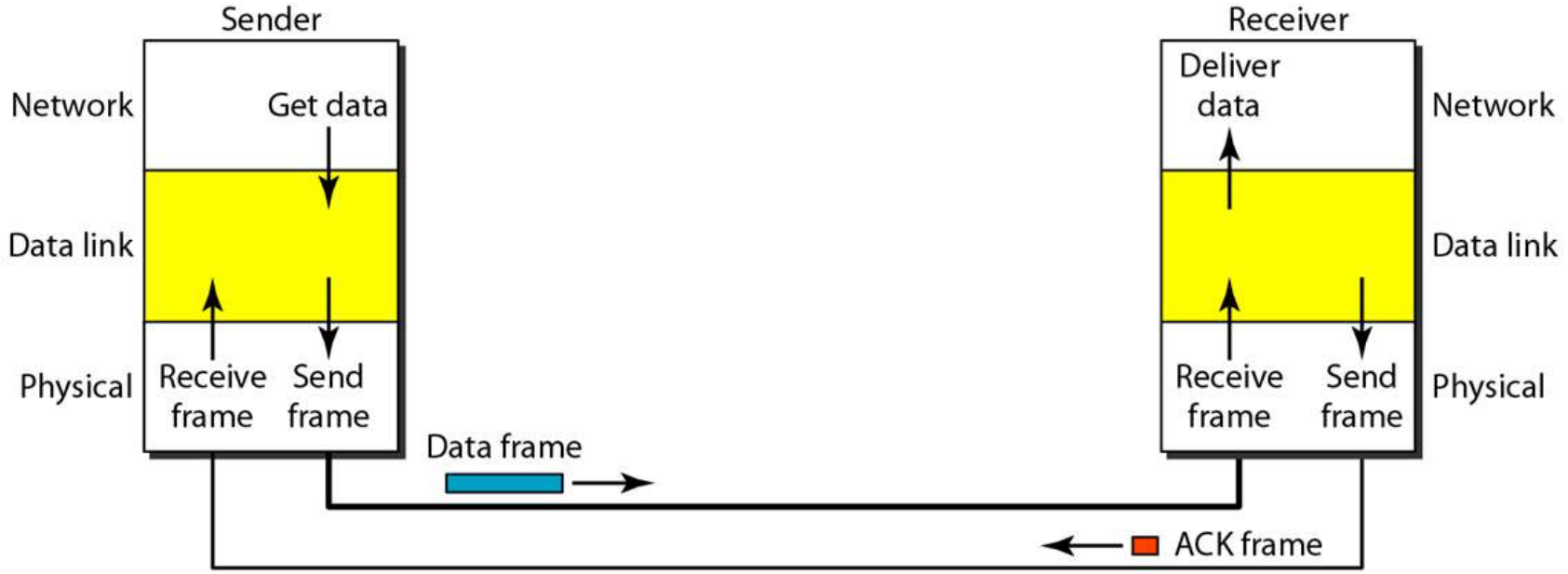
SOMAIYA VIDYAVIHAR UNIVERSITY

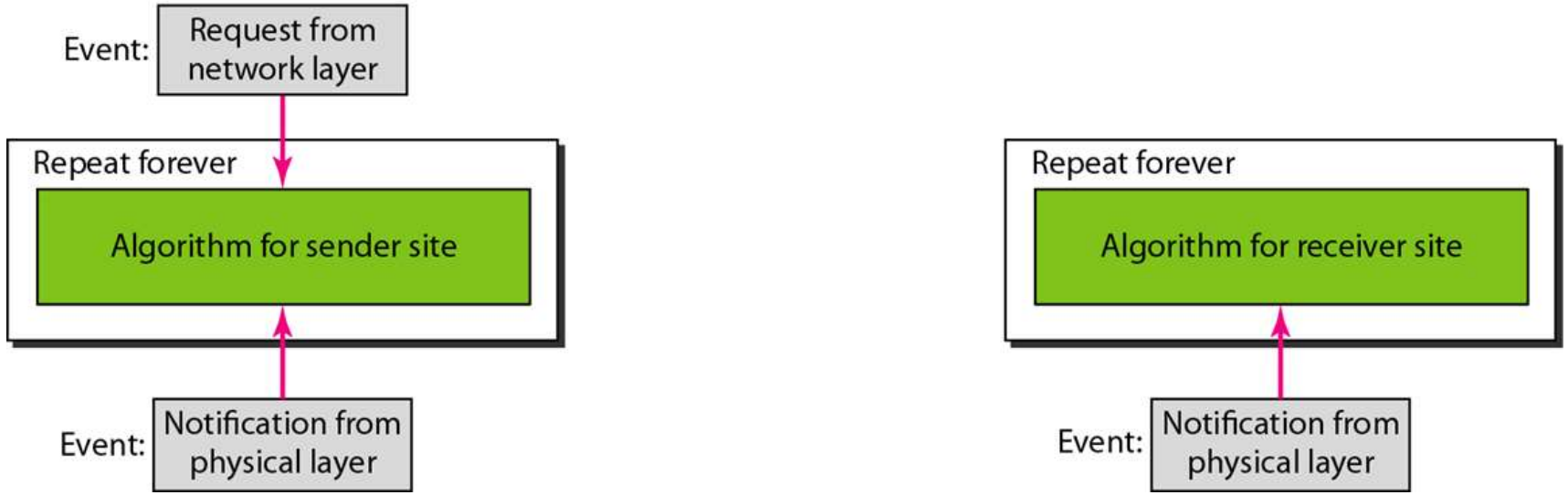
Stop and Wait Protocol

- If frames arrive faster than they can be processed, frames must be stored until their use.
- To prevent receiver from becoming overwhelmed, receiver needs to tell sender to slow down (ACK).
- Sender sends one frame, stops until it receives confirmation from receiver and then send next frame.

Somaiya TRUST

Stop and Wait Protocol





SOMAIYA VIDYAVIHAR UNIVERSITY

Stop and Wait Protocol

```
1  while(true)                          //Repeat forever
2  canSend = true                       //Allow the first frame to go
3  {
4   WaitForEvent();                     // Sleep until an event occurs
5   if(Event(RequestToSend) AND canSend)
6   {
7      GetData();
8      MakeFrame();
9      SendFrame();                     //Send the data frame
10     canSend = false;                 //Cannot send until ACK arrives
11  }
12  WaitForEvent();                     // Sleep until an event occurs
13  if(Event(ArrivalNotification)       // An ACK has arrived
14   {
15     ReceiveFrame();                  //Receive the ACK frame
16     canSend = true;
17   }
18 }
```

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Stop and Wait Protocol

```
1  while(true)                              //Repeat forever
2  {
3    WaitForEvent();                        // Sleep until an event occurs
4    if(Event(ArrivalNotification))         //Data frame arrives
5    {
6       ReceiveFrame();
7       ExtractData();
8       Deliver(data);                      //Deliver data to network layer
9       SendFrame();                        //Send an ACK frame
10   }
11 }
```

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Stop and Wait Protocol *(Example)*

Figure *shows an example of communication using this protocol. It is still very simple.*

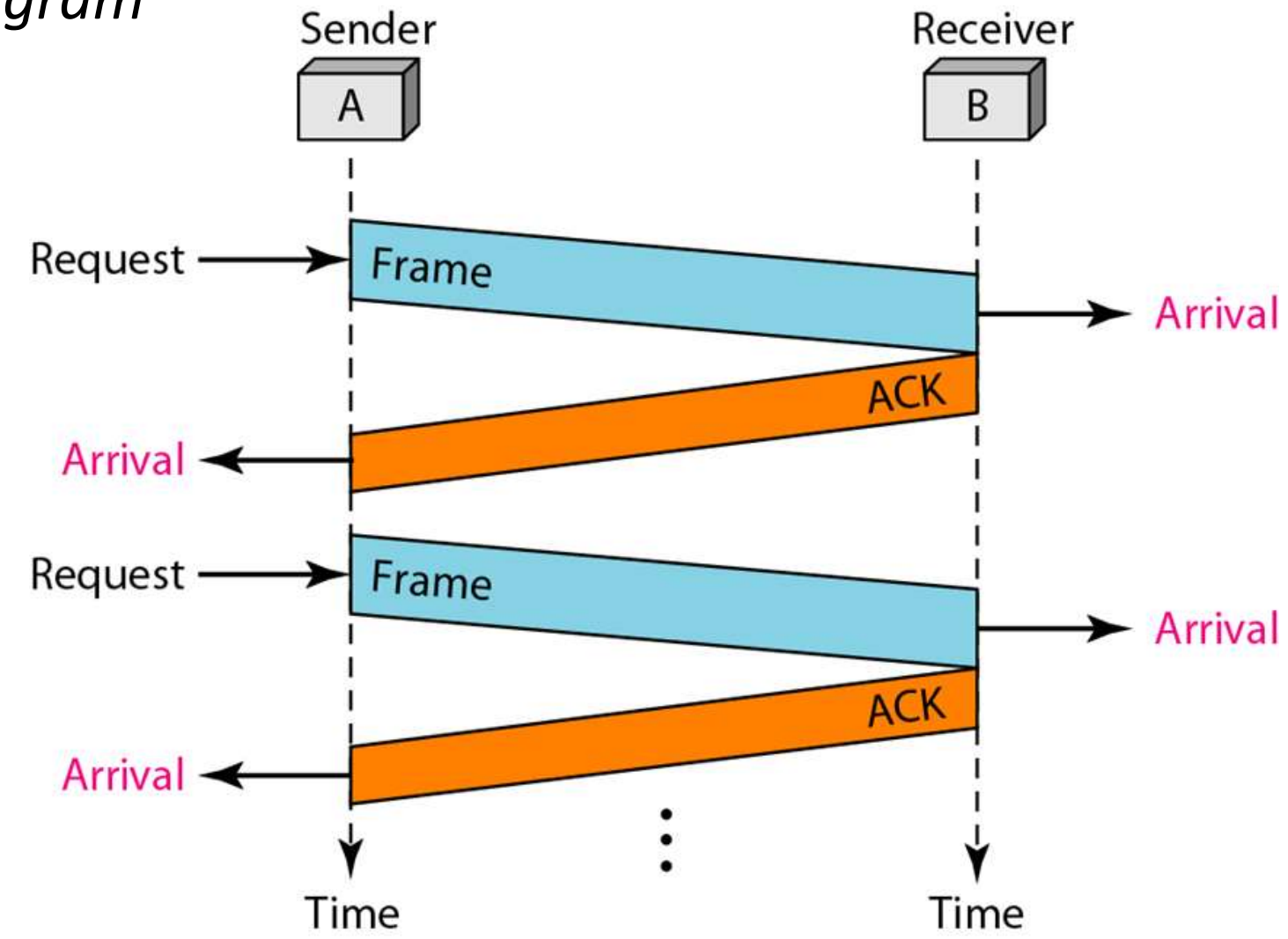
- *The sender sends one frame and waits for feedback from the receiver.*
- *When the ACK arrives, the sender sends the next frame.*

Note that sending two frames in the protocol involves the sender in four events and the receiver in two events.

Somaiya TRUST

Stop and Wait Protocol *(Example)*

Figure: *Flow Diagram*



SOMAIYA VIDYAVIHAR UNIVERSITY

Noisy Channel

Although the Stop-and-Wait Protocol gives us an idea of how to add flow control to its predecessor, noiseless channels are nonexistent. We discuss three protocols in this section that use error control.

- Stop-and-Wait Automatic Repeat Request
- Go-Back-N Automatic Repeat Request
- Selective Repeat Automatic Repeat Request

Somaiya TRUST

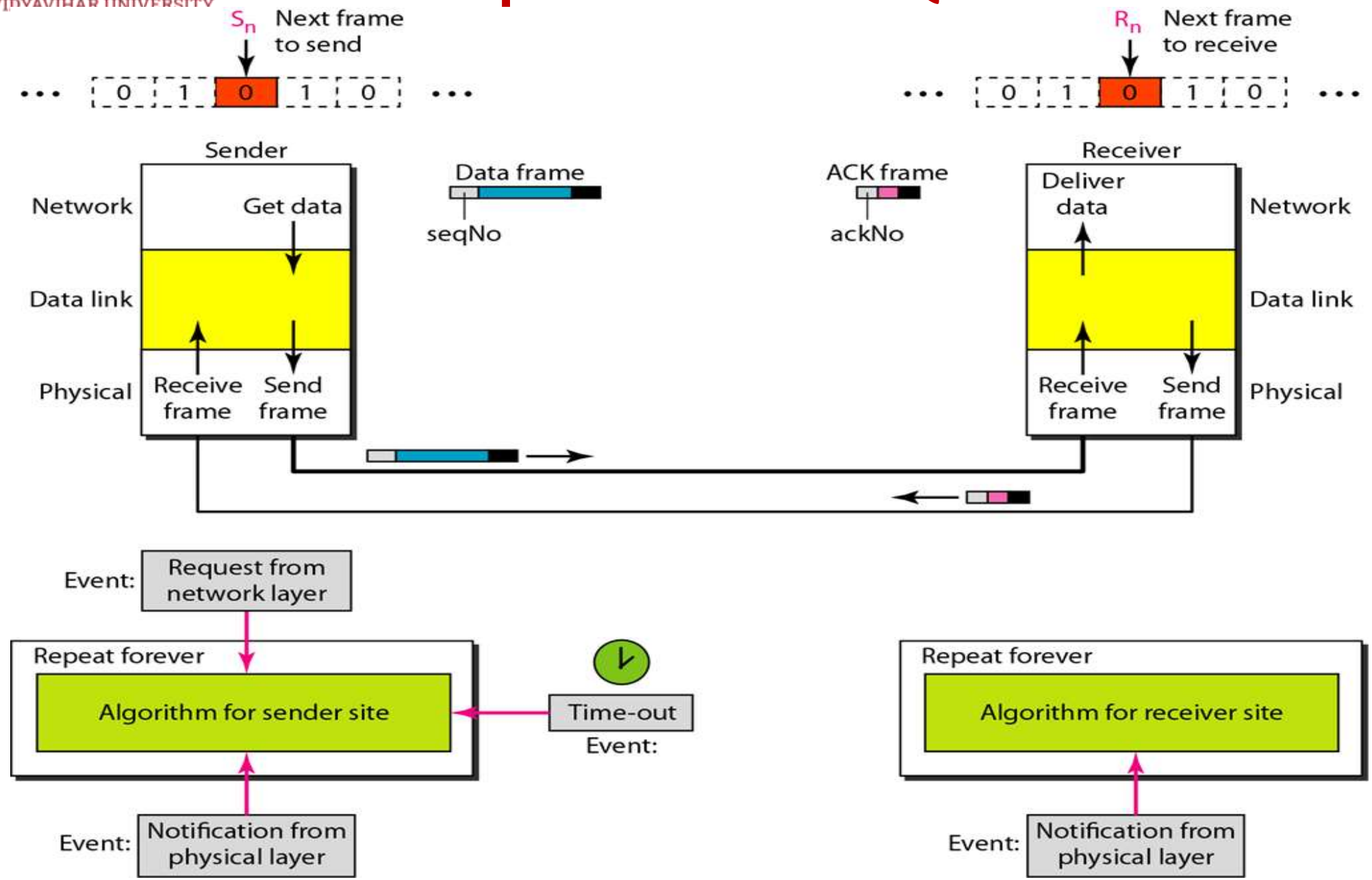
SOMAIYA VIDYAVIHAR UNIVERSITY

Stop And Wait ARQ

- Error correction in Stop-and-Wait ARQ is done by keeping a copy of the sent frame and retransmitting of the frame when the timer expires.
- In Stop-and-Wait ARQ, we use sequence numbers to number the frames. The sequence numbers are based on modulo-2 arithmetic.
- In Stop-and-Wait ARQ, the acknowledgment number always announces in modulo-2 arithmetic the sequence number of the next frame expected.

Somaiya TRUST

Stop And Wait ARQ



SOMAIYA VIDYAVIHAR UNIVERSITY

Algorithm: Sender site Algorithm for Stop And Wait ARQ Protocol

```
1  Sn = 0;                              // Frame 0 should be sent first
2  canSend = true;                      // Allow the first request to go
3  while(true)                          // Repeat forever
4  {
5    WaitForEvent();                    // Sleep until an event occurs
6    if(Event(RequestToSend) AND canSend)
7    {
8       GetData();
9       MakeFrame(Sn);                        //The seqNo is Sn
10      StoreFrame(Sn);                       //Keep copy
11      SendFrame(Sn);
12      StartTimer();
13      Sn = Sn + 1;
14      canSend = false;
15   }
16   WaitForEvent();                          // Sleep
```

(Continues…)

Somaiya TRUST

Algorithm: Sender site Algorithm for Stop And Wait ARQ Protocol

```
17      if(Event(ArrivalNotification)          // An ACK has arrived
18      {
19        ReceiveFrame(ackNo);                 //Receive the ACK frame
20        if(not corrupted AND ackNo == Sn)    //Valid ACK
21          {
22            Stoptimer();
23            PurgeFrame(Sn-1);                //Copy is not needed
24            canSend = true;
25          }
26       }
27
28      if(Event(TimeOut)                      // The timer expired
29      {
30       StartTimer();
31       ResendFrame(Sn-1);                    //Resend a copy check
32      }
33  }
```

Algorithm: Receiver site Algorithm for Stop And Wait ARQ Protocol

```
1  Rn = 0;                          // Frame 0 expected to arrive first
2  while(true)
3  {
4    WaitForEvent();                // Sleep until an event occurs
5    if(Event(ArrivalNotification))  //Data frame arrives
6    {
7       ReceiveFrame();
8       if(corrupted(frame));
9          sleep();
10      if(seqNo == Rn)              //Valid data frame
11      {
12       ExtractData();
13        DeliverData();             //Deliver data
14        Rn = Rn + 1;
15      }
16       SendFrame(Rn);             //Send an ACK
17   }
18 }
```

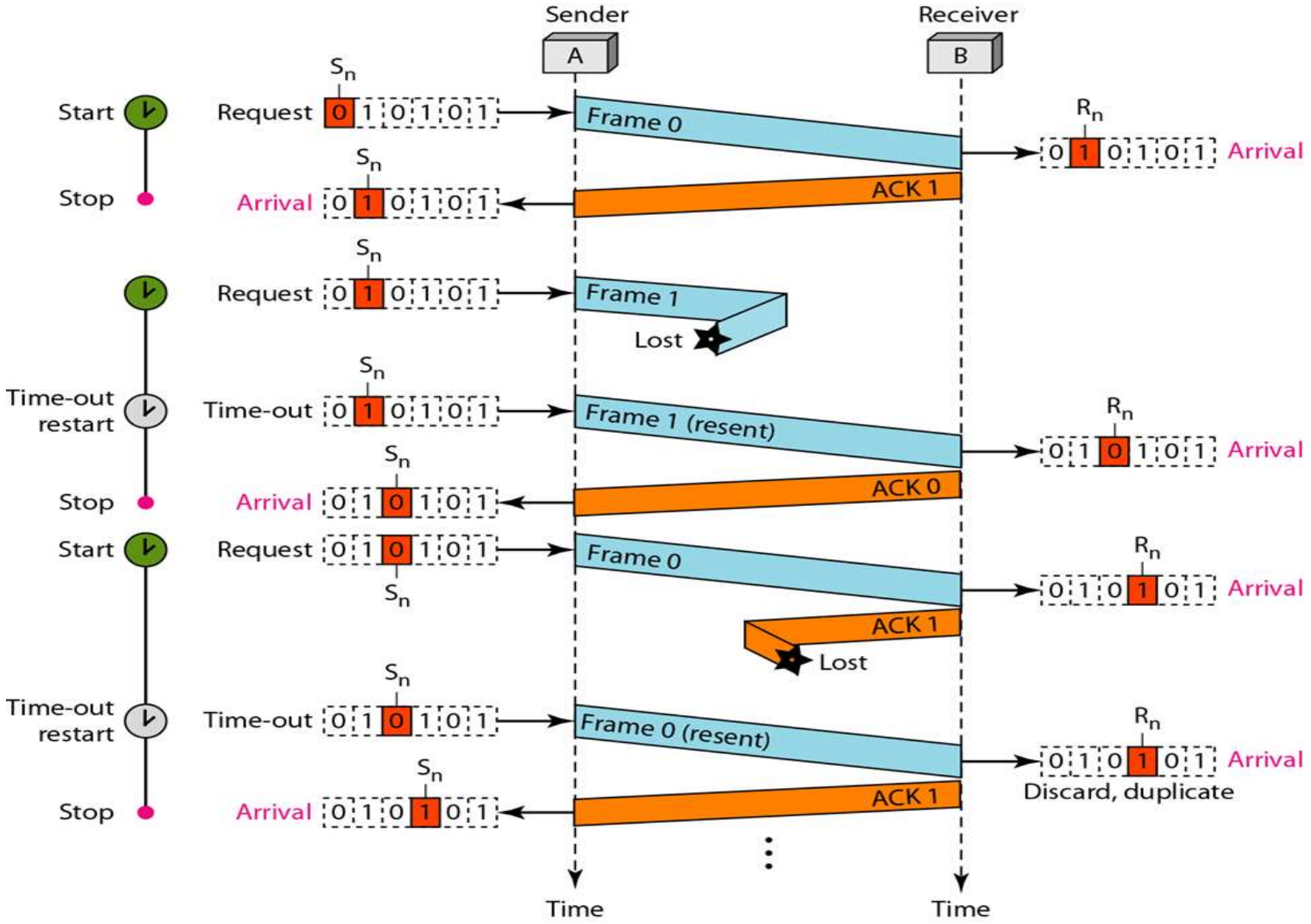
Stop And Wait ARQ *(Example)*

***Figure** shows an example of Stop-and-Wait ARQ.*

- *Frame 0 is sent and acknowledged.*
- *Frame 1 is lost and resent after the time-out.*
- *The resent frame 1 is acknowledged and the timer stops.*
- *Frame 0 is sent and acknowledged, but the acknowledgment is lost. The sender has no idea if the frame or the acknowledgment is lost, so after the time-out, it resends frame 0, which is acknowledged.*

SOMAIYA VIDYAVIHAR UNIVERSITY

Stop And Wait ARQ *(Example)*



Stop And Wait ARQ

Efficiency:

- *Very inefficient if channel is **thick** (large bandwidth)and **long** (round trip delay is long).*

- *The product of these two is called the **bandwidth-delay product** (volume of the pipe in bits).*

- *The **bandwidth-delay product** is a measure of the number of bits we can send out of our system while waiting for ACK from the receiver.*

Stop And Wait ARQ

Example 1:

Assume that, in a Stop-and-Wait ARQ system, the bandwidth of the line is 1 Mbps, and 1 bit takes 20 ms to make a round trip. What is the bandwidth-delay product? If the system data frames are 1000 bits in length, what is the utilization percentage of the link?

Solution:

The bandwidth-delay product is:

$$(1 \times 10^6) \times (20 \times 10^{-3}) = 20{,}000 \text{ bits}$$

Stop And Wait ARQ

The system can send 20,000 bits during the time it takes for the data to go from the sender to the receiver and then back again. However, the system sends only 1000 bits.

We can say that the ***link utilization*** *is only* ***1000/20,000****, or* ***5 percent****.*

For this reason, for a link with a high bandwidth or long delay, the use of Stop-and-Wait ARQ wastes the capacity of the link.

Stop And Wait ARQ

Pipelining:

- *No pipelining in Stop-And-Wait ARQ.*
- *We need to wait for a frame to reach the destination and be acknowledged before next frame can be sent.*
- *Pipelining is applicable to next two protocols, wherein several frames can be sent before receiving acknowledgement for previous frames.*
- *Pipelining improves efficiency.*

SOMAIYA VIDYAVIHAR UNIVERSITY

Go-Back-N ARQ

- To improve the efficiency of transmission (filling the pipe), multiple frames must be in transition while waiting for acknowledgment.
- This protocol can send several frames before receiving acknowledgments it keep a copy of these frames until the acknowledgments arrive.
- In the Go-Back-N Protocol, the **sequence numbers** are modulo 2^m i.e. the sequence numbers range from **0 to $2^m - 1$**, where m is the size of the sequence number field in bits.

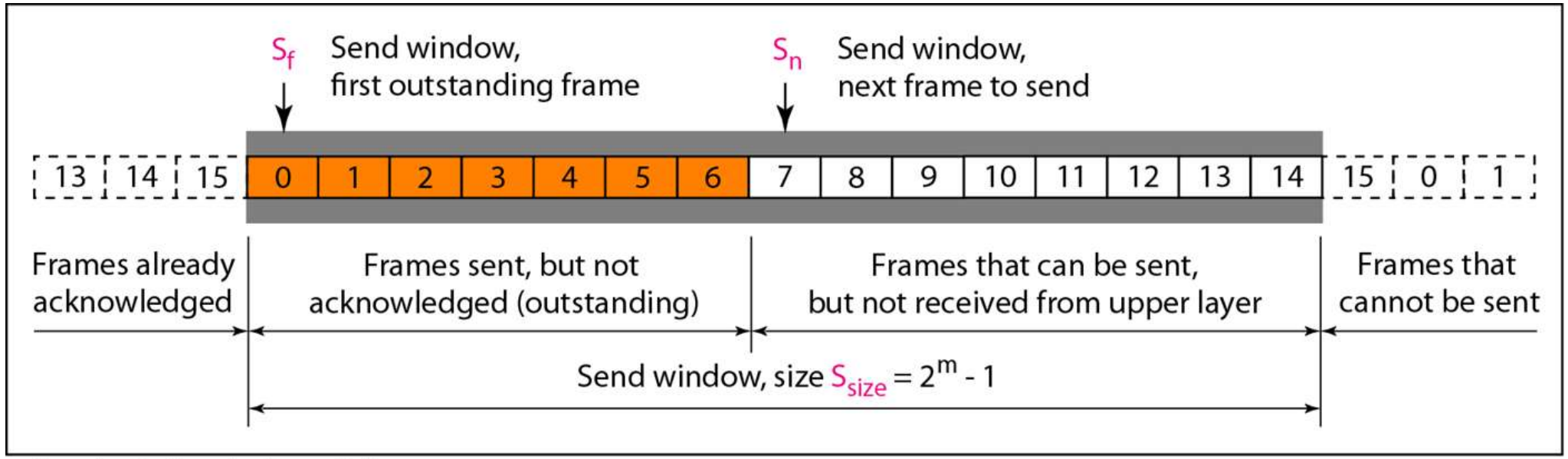
Somaiya TRUST

Go-Back-N ARQ

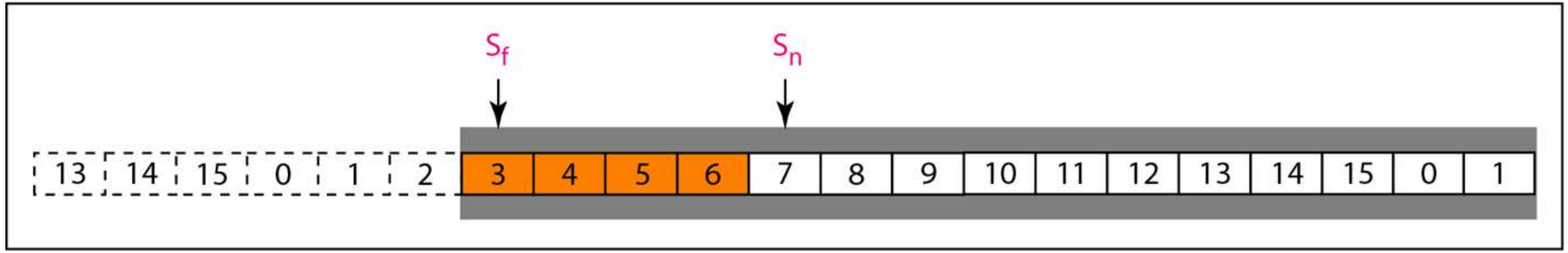
Sliding Window:

- Sliding window is an abstract concept that defines the range of sequence numbers.
- The range which is the concern of the sender is called the send sliding window; the range that is the concern of the receiver is called the receive sliding window.
- The window at any time divides the possible sequence numbers into four regions.

Send Window for Go-Back-N ARQ:



a. Send window before sliding



b. Send window after sliding

Go-Back-N ARQ

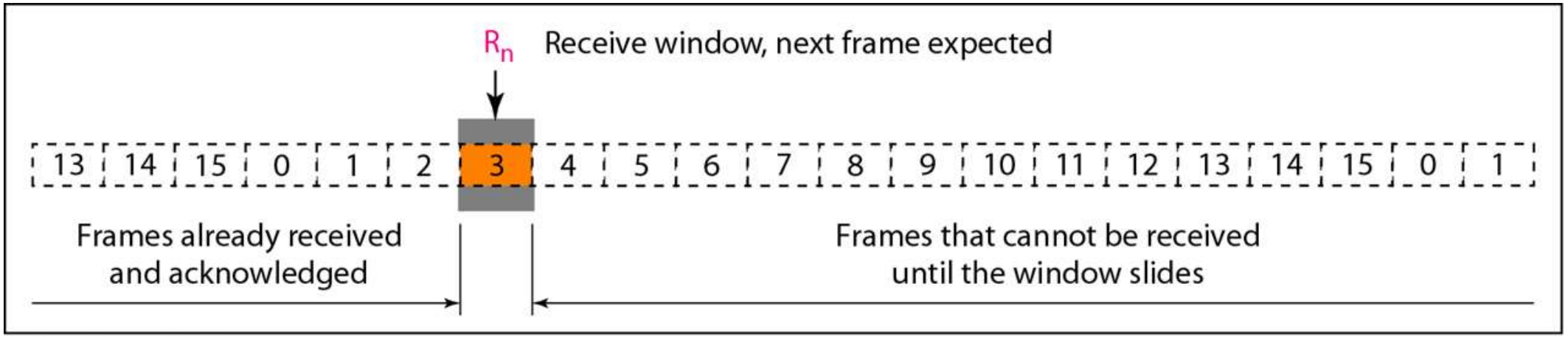
Send Window :

- Sliding window is an abstract concept that defines the range of sequence numbers.
- The send window can slide one or more slots when a valid acknowledgment arrives.

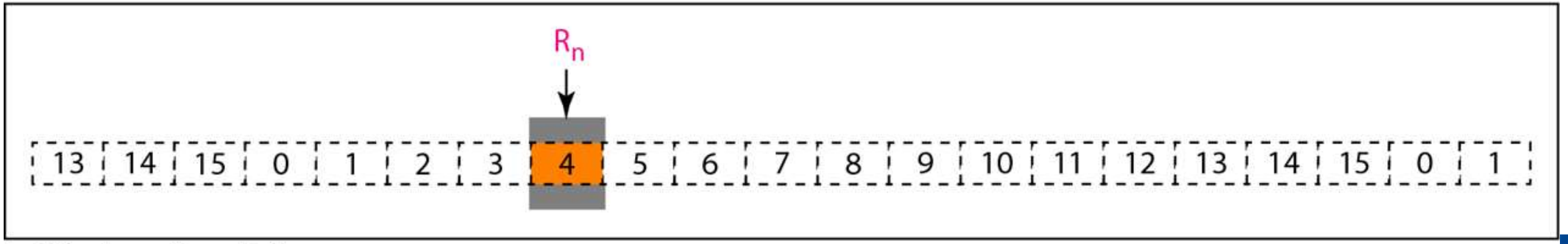
Receive Window :

- 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.

Receive Window for Go-Back-N ARQ:

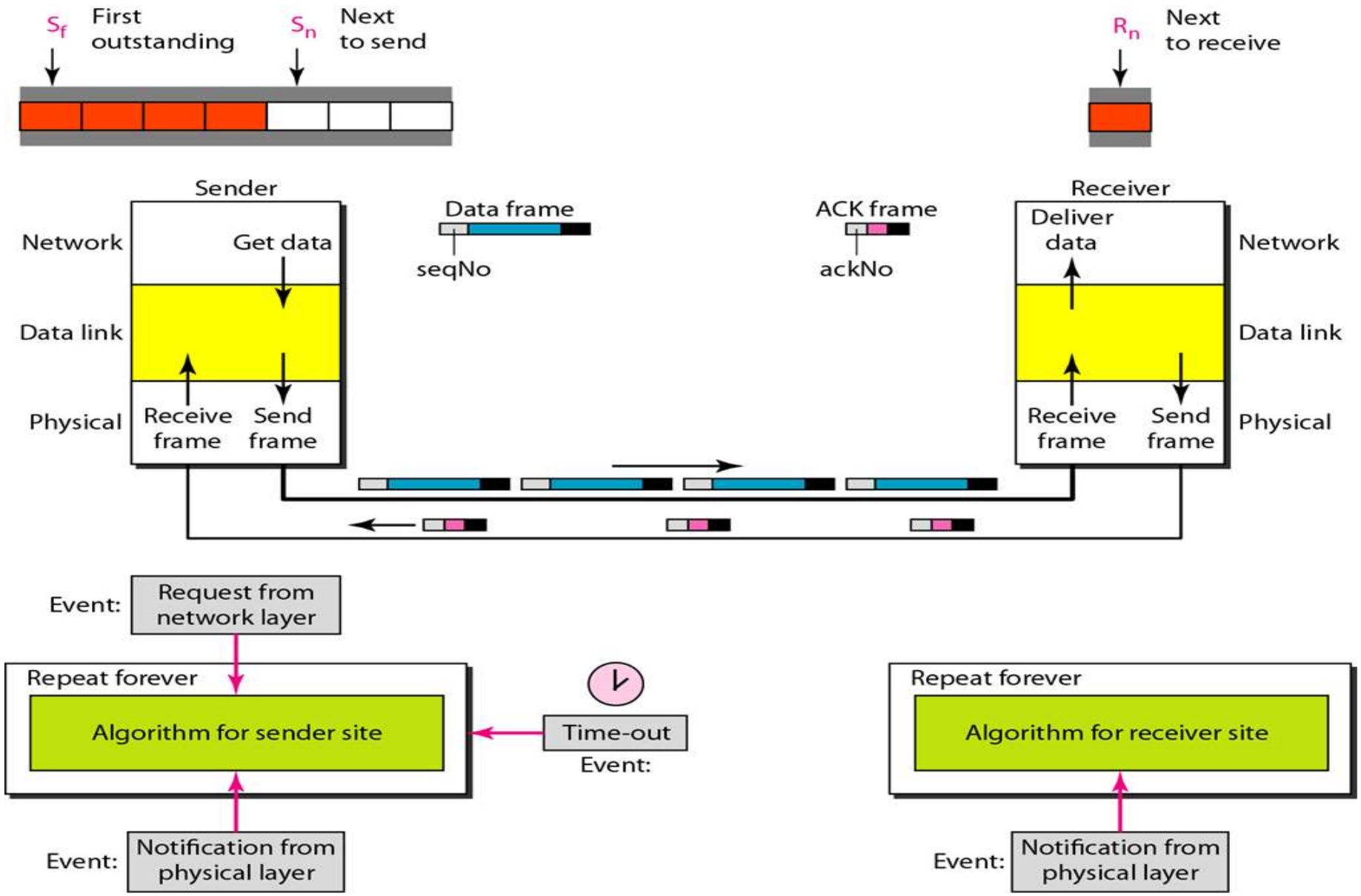


a. Receive window



b. Window after sliding

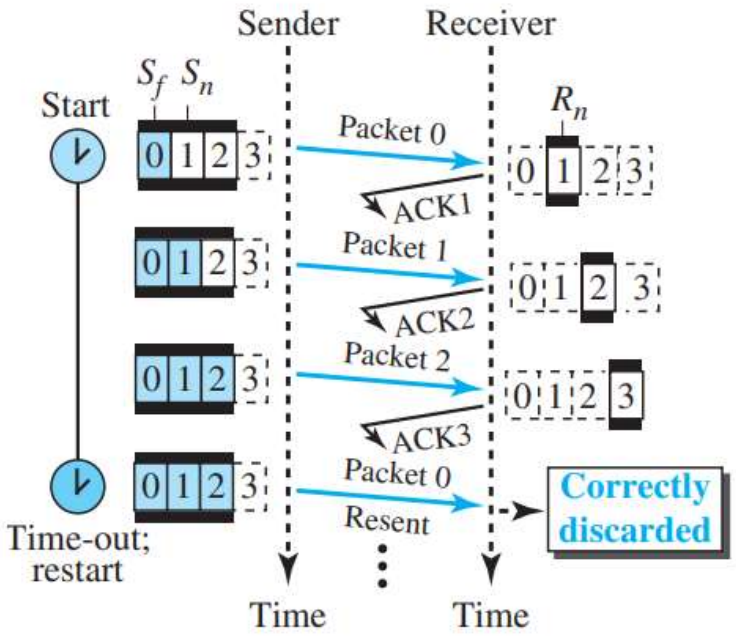
Go-Back-N ARQ


S_f First outstanding
S_n Next to send
R_n Next to receive
Sender
Receiver
Network
Get data
Data link
Physical
Receive frame
Send frame
Data frame
seqNo
ACK frame
ackNo
Deliver data
Network
Data link
Physical
Receive frame
Send frame
Event:
Request from network layer
Repeat forever
Algorithm for sender site
Time-out
Event:
Event:
Notification from physical layer
Repeat forever
Algorithm for receiver site
Event:
Notification from physical layer

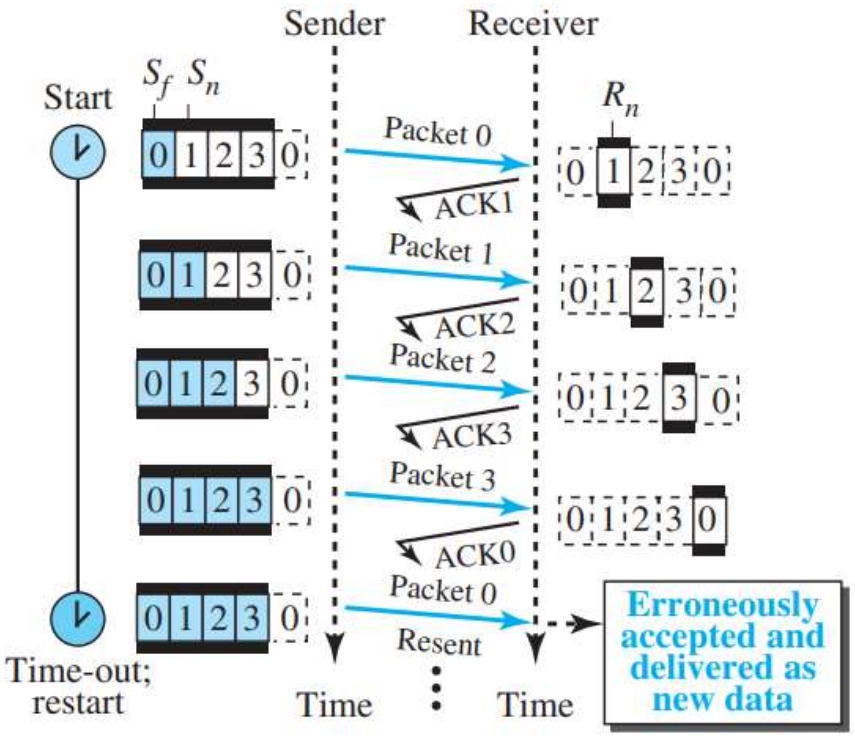
Go-Back-N ARQ

NOTE:

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



a. Send window of size $< 2^m$



b. Send window of size $= 2^m$

Algorithm: *Go-Back-N sender algorithm*

```
1  Sw = 2^m - 1;
2  Sf = 0;
3  Sn = 0;
4
5  while (true)                          //Repeat forever
6  {
7   WaitForEvent();
8    if(Event(RequestToSend))            //A packet to send
9    {
10      if(Sn-Sf >= Sw)                  //If window is full
11            Sleep();
12      GetData();
13      MakeFrame(Sn);
14      StoreFrame(Sn);
15      SendFrame(Sn);
16      Sn = Sn + 1;
17      if(timer not running)
18           StartTimer();
19   }
20
```

(continues..)

Algorithm: *Go-Back-N sender algorithm* *(continued)*

```
21    if(Event(ArrivalNotification))  //ACK arrives
22    {
23       Receive(ACK);
24       if(corrupted(ACK))
25            Sleep();
26       if((ackNo>Sf)&&(ackNo<=Sn))   //If a valid ACK
27       While(Sf <= ackNo)
28         {
29          PurgeFrame(Sf);
30          Sf = Sf + 1;
31         }
32         StopTimer();
33    }
34
35    if(Event(TimeOut))                  //The timer expires
36    {
37     StartTimer();
38     Temp = Sf;
39     while(Temp < Sn);
40       {
41        SendFrame(Sf);
42        Sf = Sf + 1;
43       }
44    }
45  }
```

Algorithm: *Go-Back-N receiver algorithm*

```
1  Rn = 0;
2
3  while (true)                          //Repeat forever
4  {
5    WaitForEvent();
6
7    if(Event(ArrivalNotification)) /Data frame arrives
8    {
9       Receive(Frame);
10      if(corrupted(Frame))
11           Sleep();
12      if(seqNo == Rn)                  //If expected frame
13      {
14        DeliverData();                 //Deliver data
15        Rn = Rn + 1;                   //Slide window
16        SendACK(Rn);
17      }
18   }
19 }
```

Go-Back-N ARQ

Example:

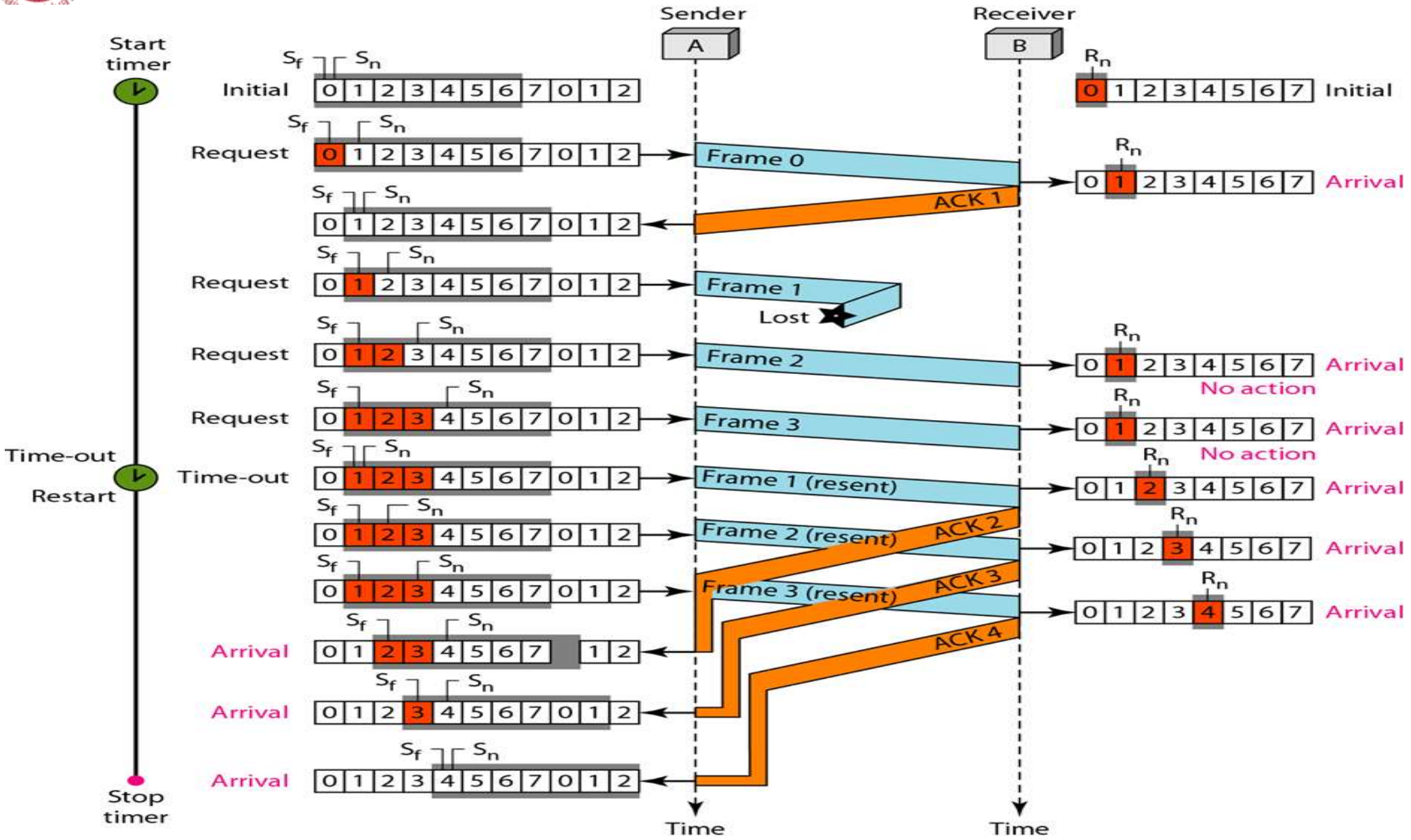
- *The physical layer must wait until this event is completed and the data link layer goes back to its sleeping state.*

- *We have shown a vertical line to indicate the delay. It is the same story with ACK 3; but when ACK 3 arrives, the sender is busy responding to ACK 2. It happens again when ACK 4 arrives.*

- *Note that before the second timer expires, all outstanding frames have been sent and the timer is stopped.*

SOMAIYA VIDYAVIHAR UNIVERSITY

Go-Back-N ARQ



Selective Repeat ARQ

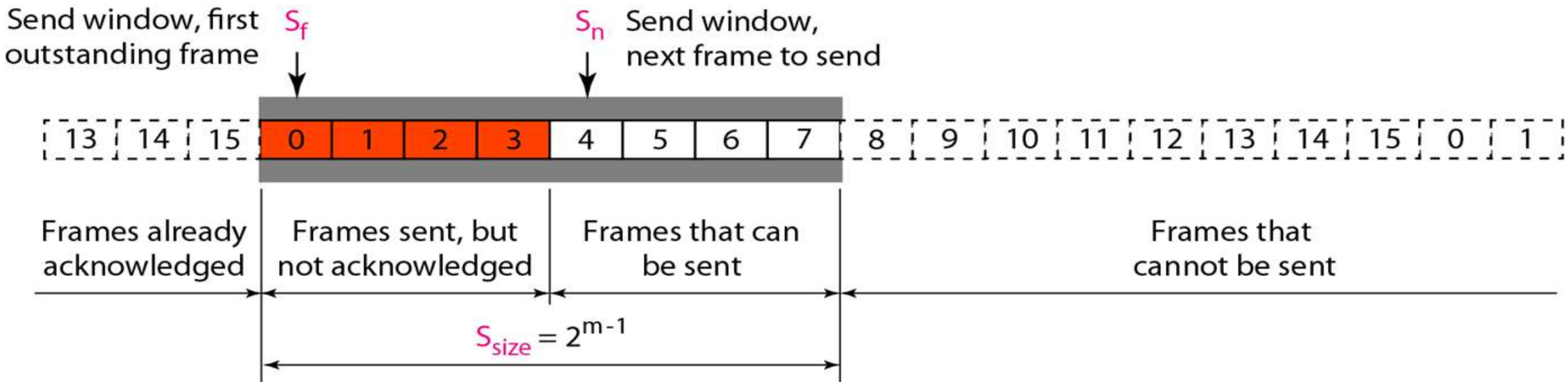
- Go-Back-N ARQ simplifies the process at the receiver site.
- The receiver keeps track of only one variable, and there is no need to buffer out-of-order frames; they are simply discarded.
- However, this protocol is very inefficient for a noisy link.
- In a noisy link a frame has a higher probability of damage, which means the resending of multiple frames. This resending uses up the bandwidth and slows down the transmission.

Selective Repeat ARQ

- Mechanism that does not resend N frames when just one frame is damaged; only the damaged frame is resent.
- More efficient for noisy links, but the processing at the receiver is more complex.
- the size of the **send window** is much smaller; it is 2^{m-1}.
- sizes of the **send window and receive window are the same**.

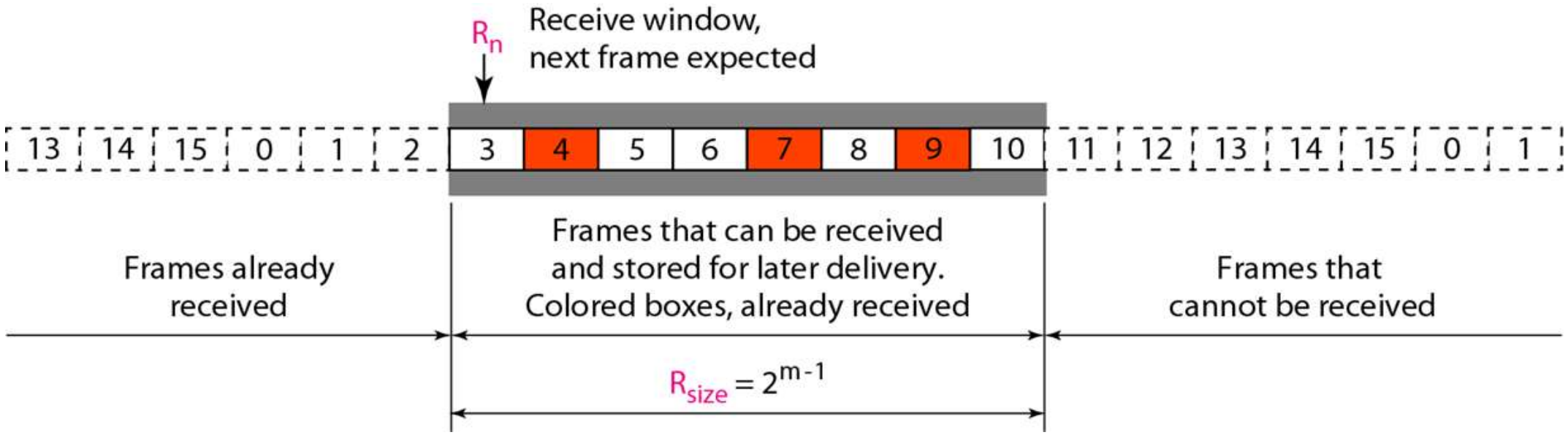
Selective Repeat ARQ

Send window for Selective Repeat ARQ:

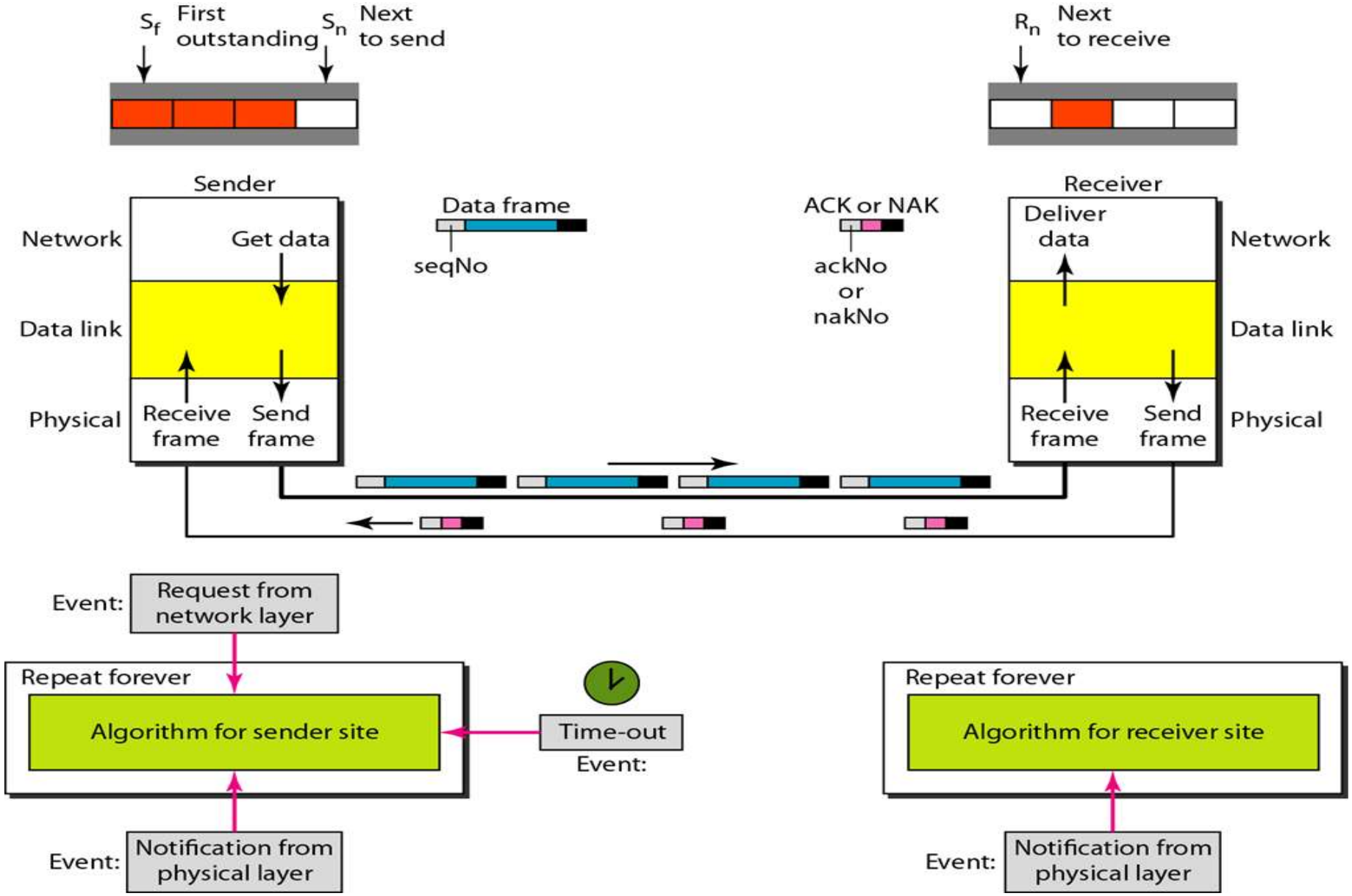


Selective Repeat ARQ

Receive window for Selective Repeat ARQ:

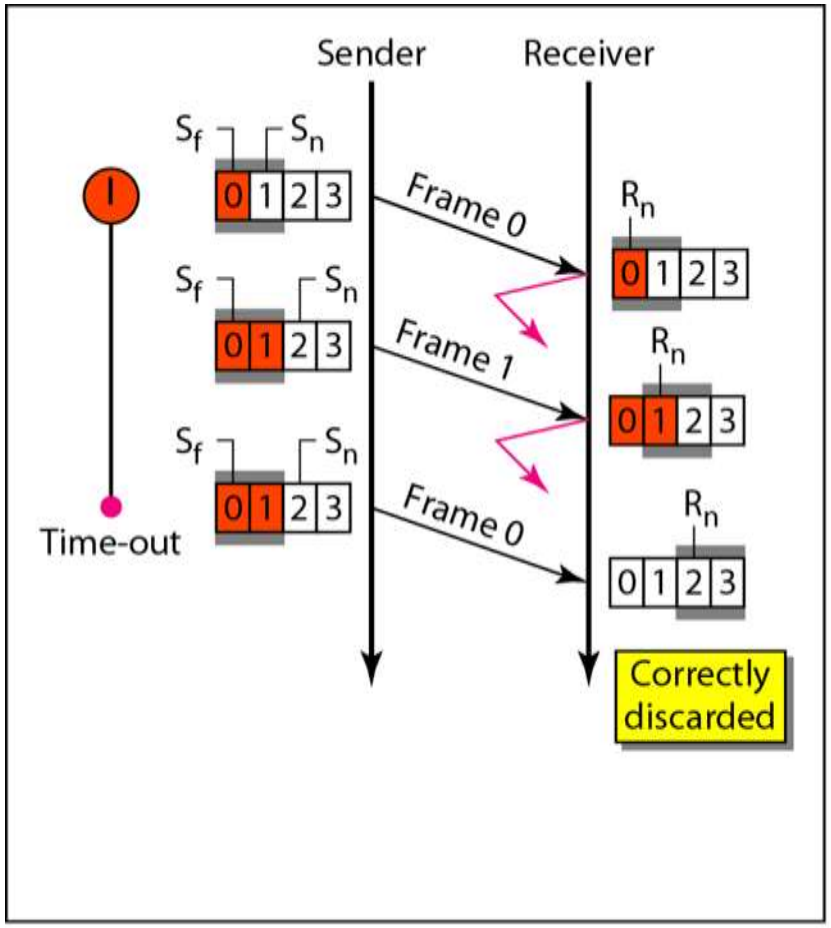


Selective Repeat ARQ

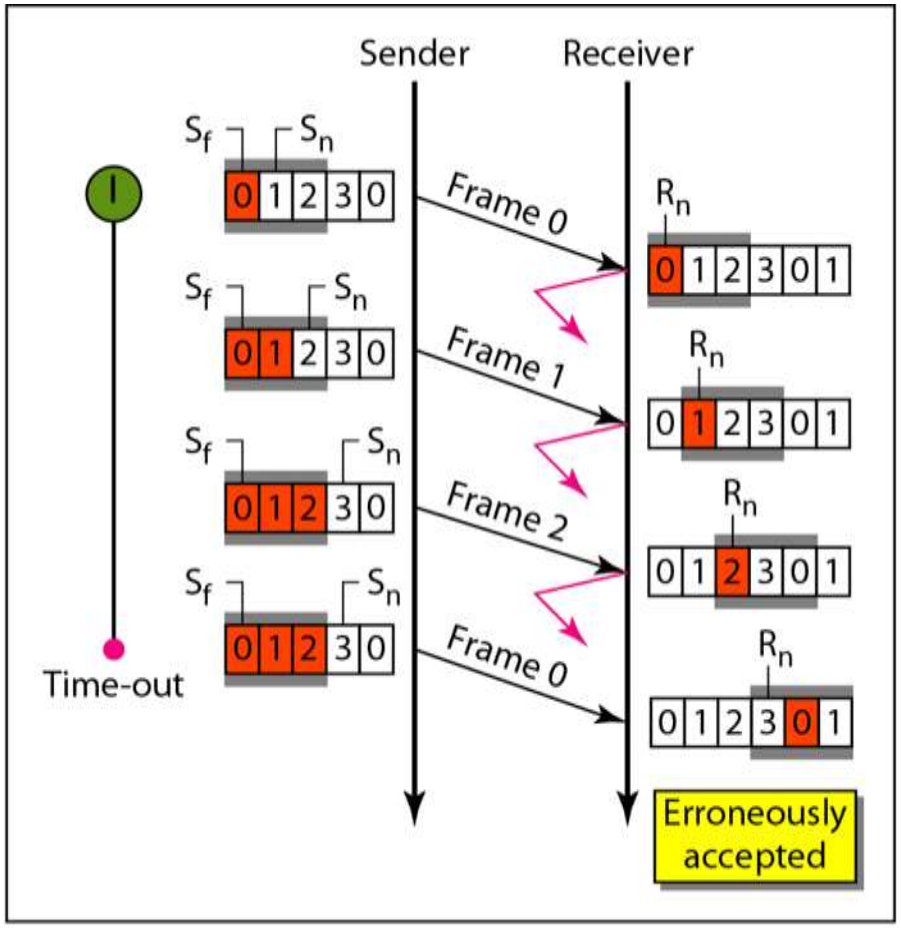


Selective Repeat ARQ

Selective Repeat ARQ, window size:



a. Window size = 2^{m-1}



b. Window size > 2^{m-1}

SOMAIYA VIDYAVIHAR UNIVERSITY

Selective Repeat ARQ

Sender Site SR ARQ Algorithm:

```
1  Sw = 2^(m-1) ;
2  Sf = 0;
3  Sn = 0;
4
5  while (true)                          //Repeat forever
6  {
7    WaitForEvent();
8    if(Event(RequestToSend))            //There is a packet to send
9    {
10     if(Sn-Sf >= Sw)                   //If window is full
11           Sleep();
12     GetData();
13     MakeFrame(Sn);
14     StoreFrame(Sn);
15     SendFrame(Sn);
16     Sn = Sn + 1;
17     StartTimer(Sn);
18   }
19
```

Somaiya TRUST

Selective Repeat ARQ

Sender Site SR ARQ Algorithm:

```
20  if(Event(ArrivalNotification)) //ACK arrives
21  {
22      Receive(frame);                 //Receive ACK or NAK
23      if(corrupted(frame))
24           Sleep();
25      if (FrameType == NAK)
26          if (nakNo between Sf and Sn)
27          {
28           resend(nakNo);
29           StartTimer(nakNo);
30          }
31      if (FrameType == ACK)
32          if (ackNo between Sf and Sn)
33          {
34            while(sf < ackNo)
35            {
36             Purge(sf);
37             StopTimer(sf);
38             Sf = Sf + 1;
39            }
40          }
41  }
```

Selective Repeat ARQ

Sender Site SR ARQ Algorithm:

```
42
43    if(Event(TimeOut(t)))              //The timer expires
44    {
45     StartTimer(t);
46     SendFrame(t);
47    }
48  }
```

SOMAIYA VIDYAVIHAR UNIVERSITY

Selective Repeat ARQ

Receiver Site SR ARQ Algorithm:

```
1  Rn = 0;
2  NakSent = false;
3  AckNeeded = false;
4  Repeat(for all slots)
5      Marked(slot) = false;
6
7  while (true)                                  //Repeat forever
8  {
9    WaitForEvent();
10
11   if(Event(ArrivalNotification))              /Data frame arrives
12   {
13      Receive(Frame);
14      if(corrupted(Frame))&& (NOT NakSent)
15      {
16       SendNAK(Rn);
17       NakSent = true;
18       Sleep();
19      }
20      if(seqNo <> Rn)&& (NOT NakSent)
21      {
22       SendNAK(Rn);
```

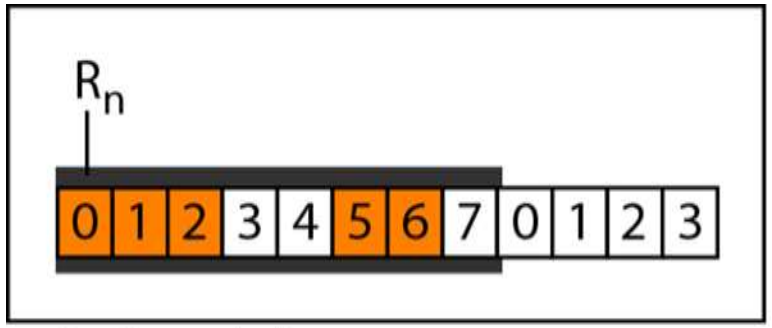
Selective Repeat ARQ

Receiver Site SR AR Algorithm

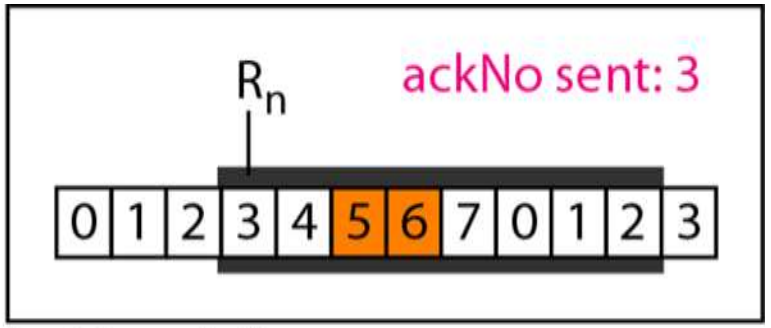
```
23        NakSent = true;
24        if ((seqNo in window)&&(!Marked(seqNo))
25        {
26         StoreFrame(seqNo)
27         Marked(seqNo)= true;
28         while(Marked(Rn))
29         {
30          DeliverData(Rn);
31          Purge(Rn);
32          Rn = Rn + 1;
33          AckNeeded = true;
34         }
35          if(AckNeeded);
36          {
37          SendAck(Rn);
38          AckNeeded = false;
39          NakSent = false;
40          }
41        }
42       }
43   }
44 }
```

Selective Repeat ARQ

Delivery of data in Selective Repeat ARQ:



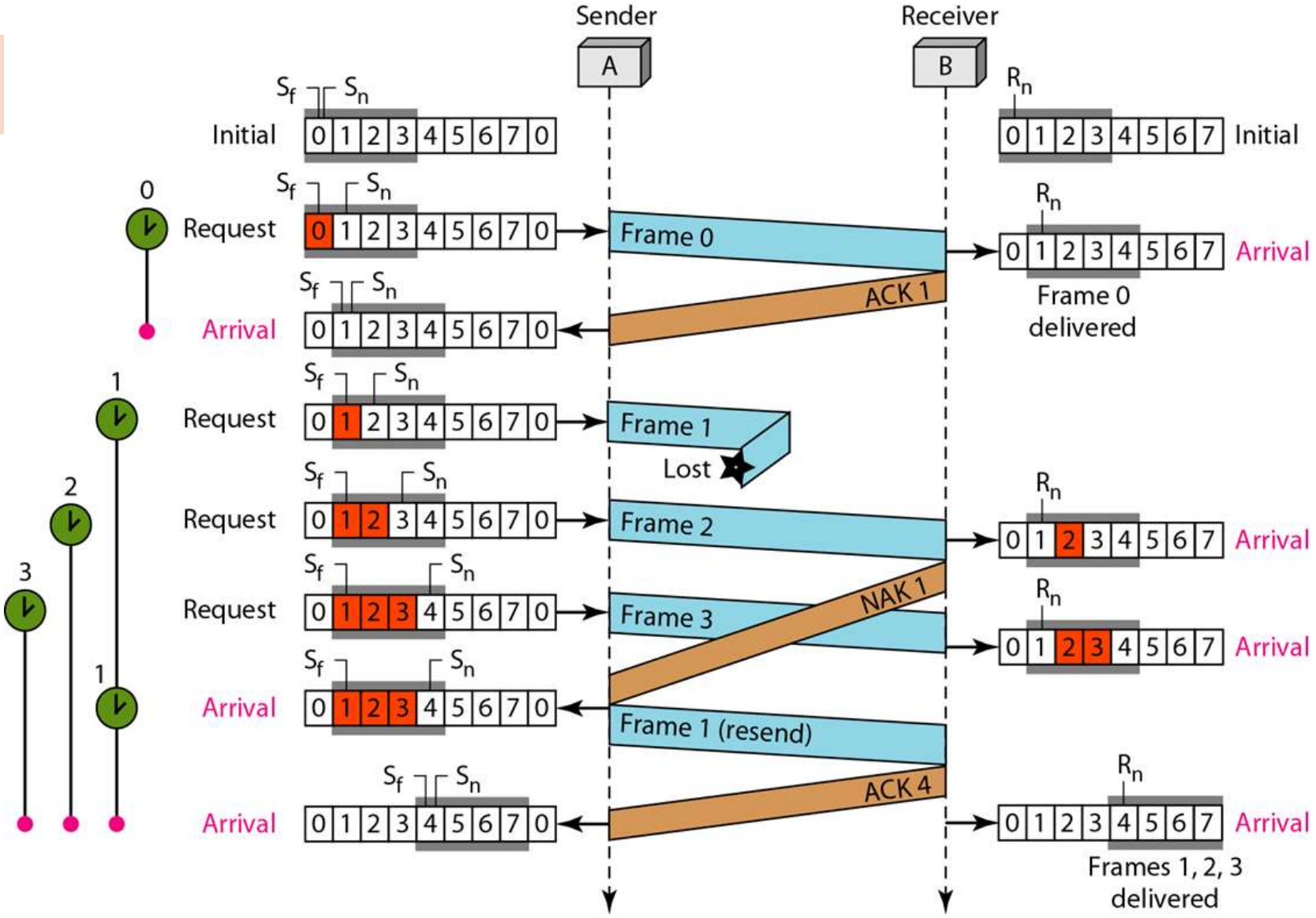
a. Before delivery



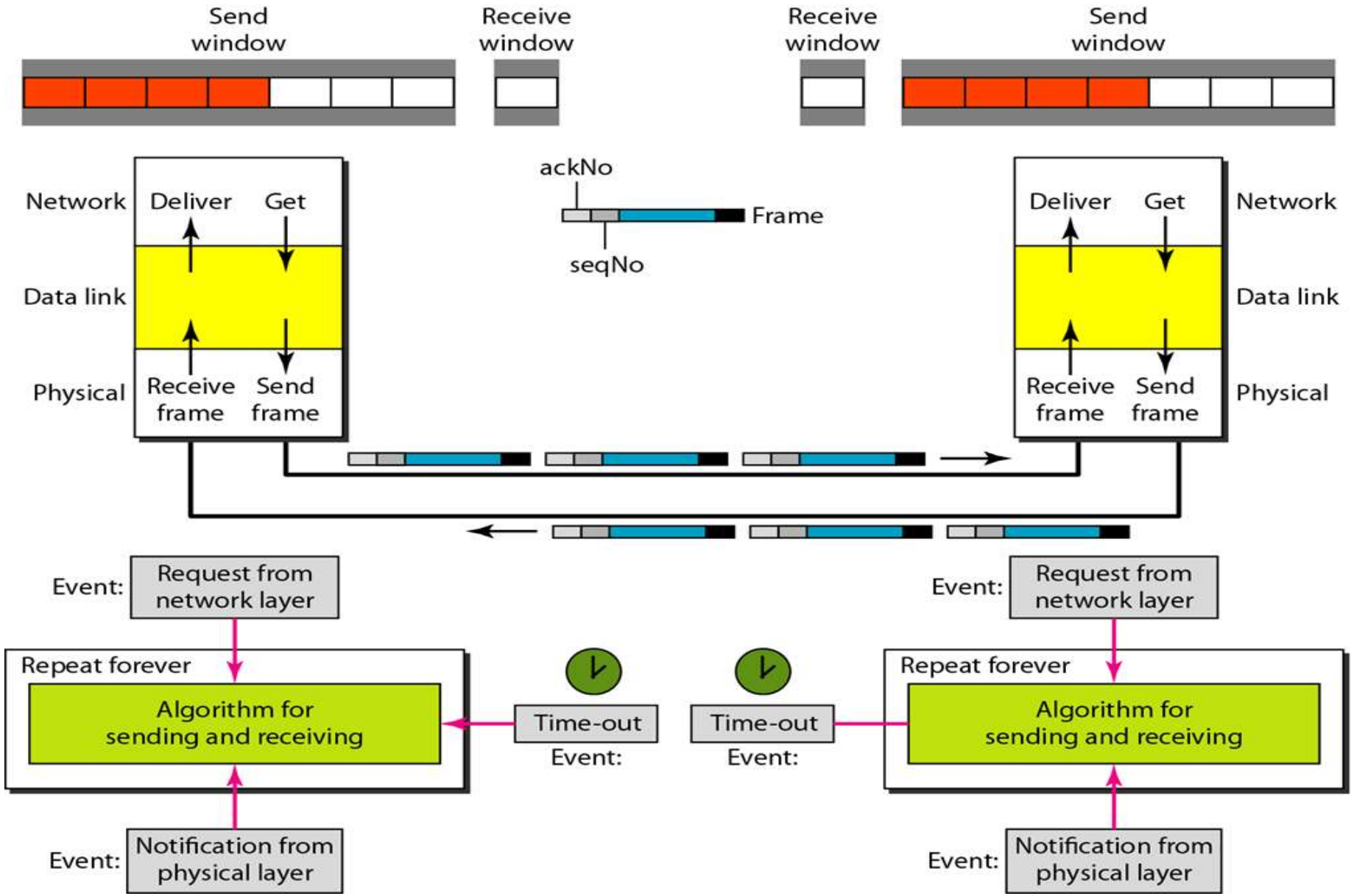
b. After delivery

Selective Repeat ARQ

Flow Diagram:



Piggybacking in Go Back N ARQ



HDLC

- **High-level Data Link Control (HDLC)** is a bit-oriented protocol for communication over point-to-point and multipoint links.

- It implements the ARQ mechanisms.

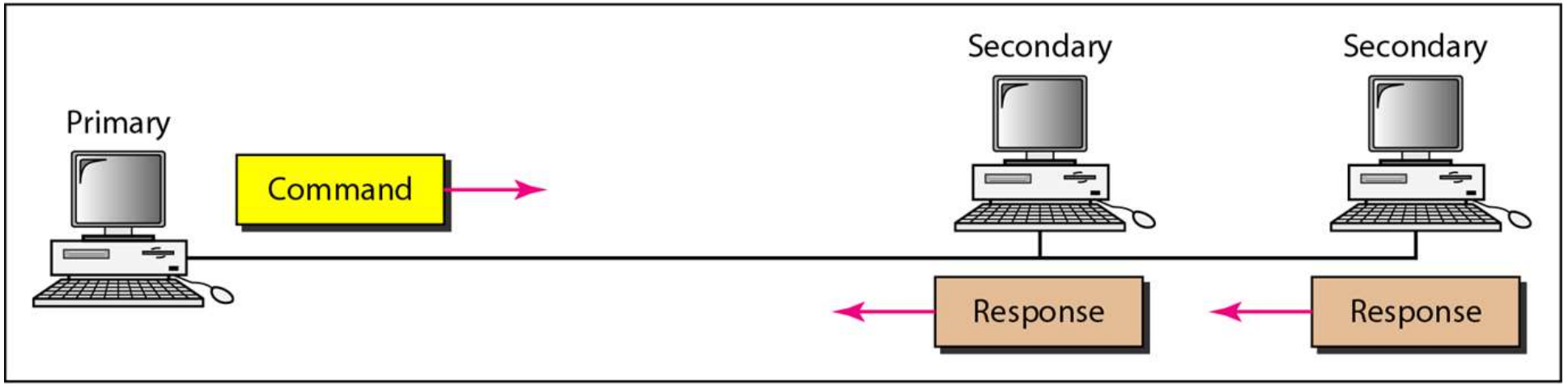
Configurations and Transfer Modes:

HDLC provides two common transfer modes that can be used in different configurations:

- *Normal response Mode(NRM)*
- *Asynchronous Balanced Mode(ABM)*

Normal Response Mode

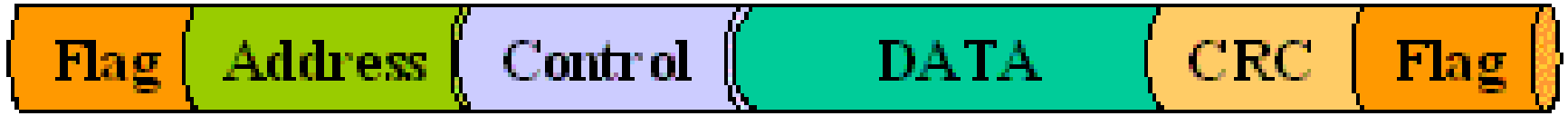
a. Point-to-point

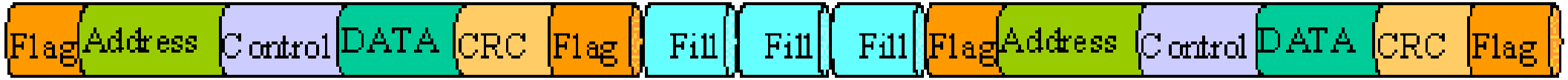


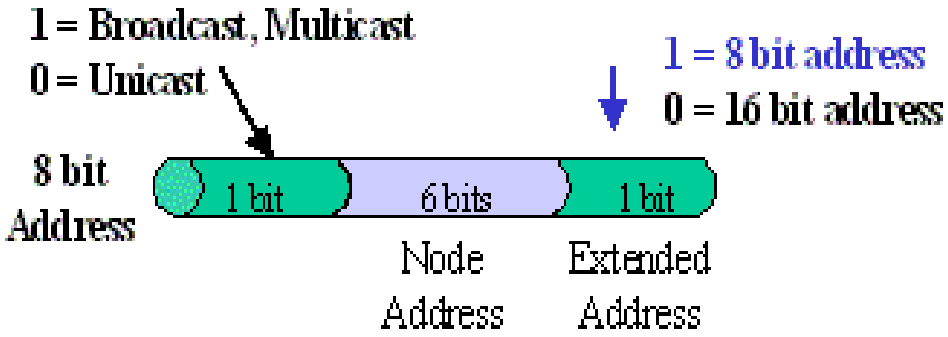
b. Multipoint

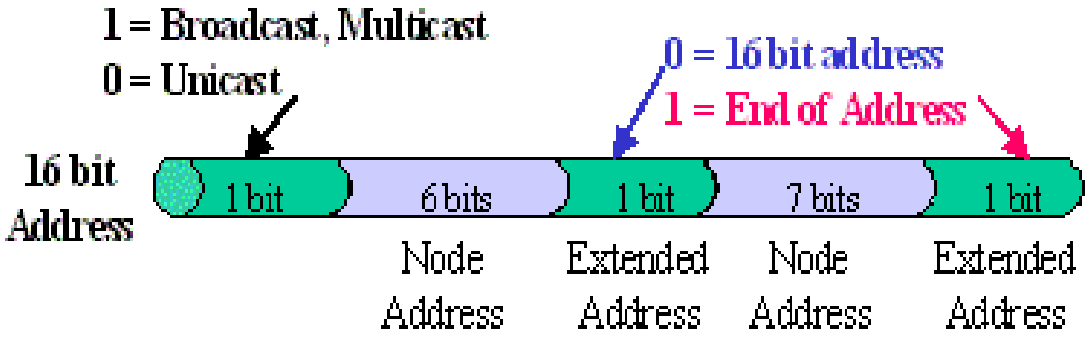
Asynchronous Balanced Mode

HDLC frames





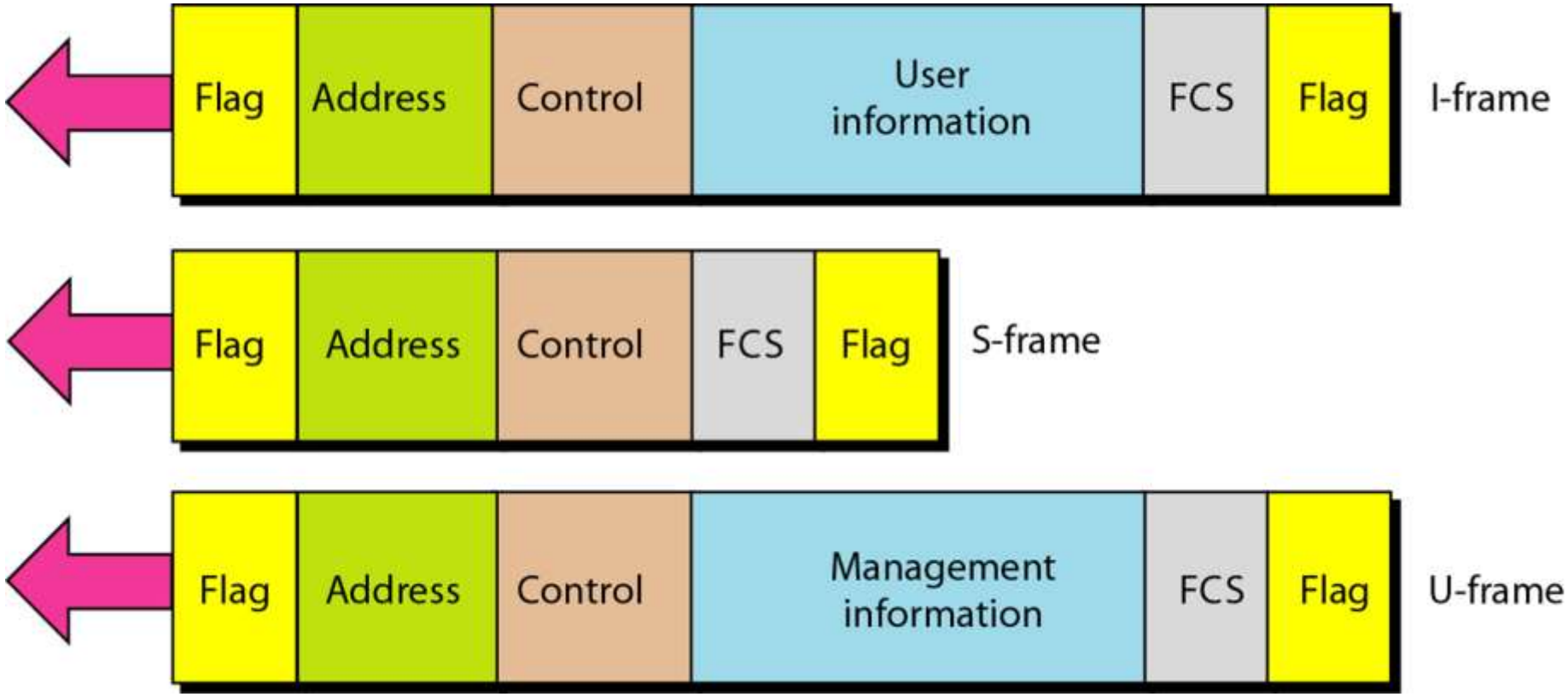




http://www.interfacebus.com/HDLC_Protocol_Description.html

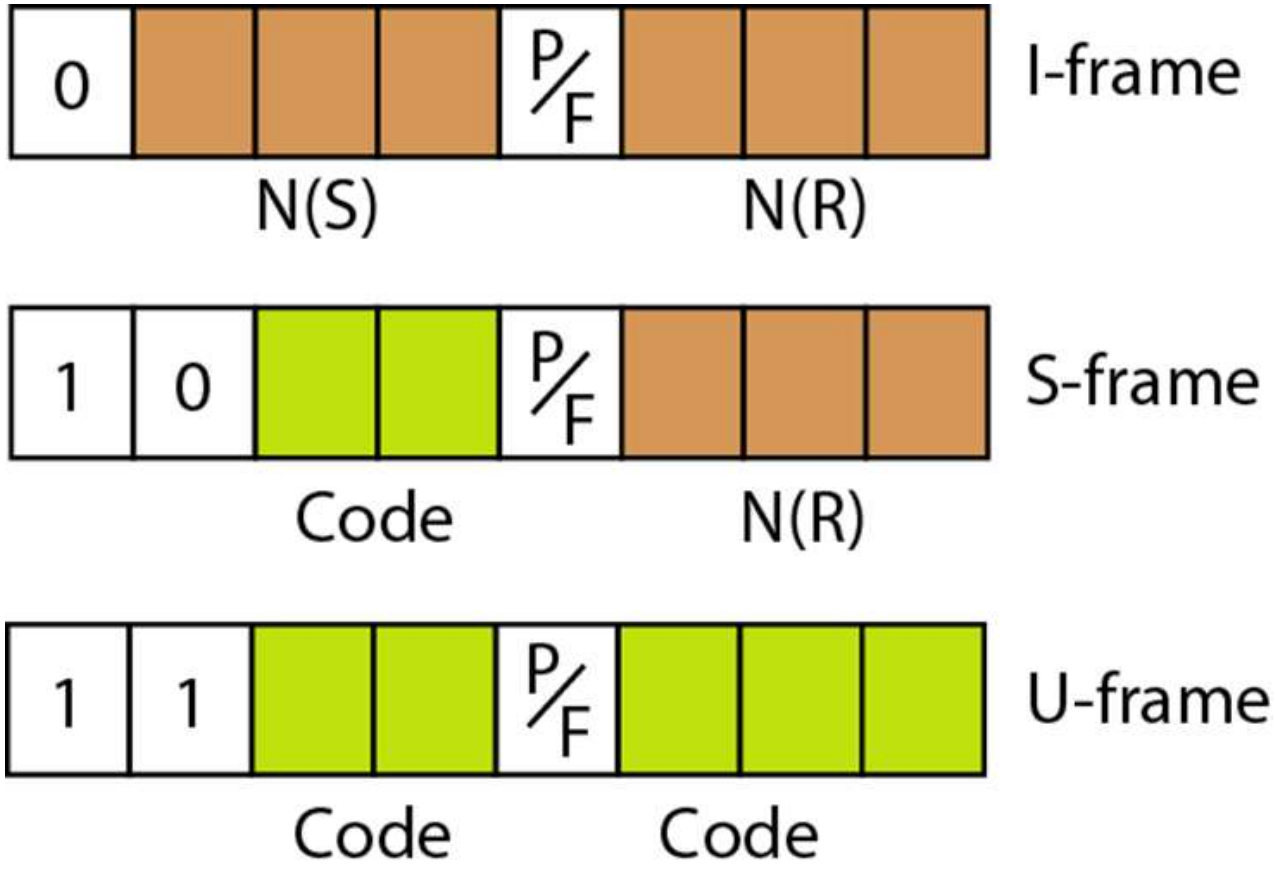
SOMAIYA VIDYAVIHAR UNIVERSITY

HDLC frames



Somaiya TRUST

Control field format for the different frame types



U-frame control command and response

Code	*Command*	*Response*	*Meaning*
00 001	SNRM		Set normal response mode
11 011	SNRME		Set normal response mode, extended
11 100	SABM	**DM**	Set asynchronous balanced mode or **disconnect mode**
11 110	SABME		Set asynchronous balanced mode, extended
00 000	UI	**UI**	Unnumbered information
00 110		**UA**	**Unnumbered acknowledgment**
00 010	DISC	**RD**	Disconnect or **request disconnect**
10 000	SIM	**RIM**	Set initialization mode or **request information mode**
00 100	UP		Unnumbered poll
11 001	RSET		Reset
11 101	XID	**XID**	Exchange ID
10 001	FRMR	**FRMR**	Frame reject

SOMAIYA VIDYAVIHAR UNIVERSITY

Multiple Access

Module 2

Data Link Layer

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

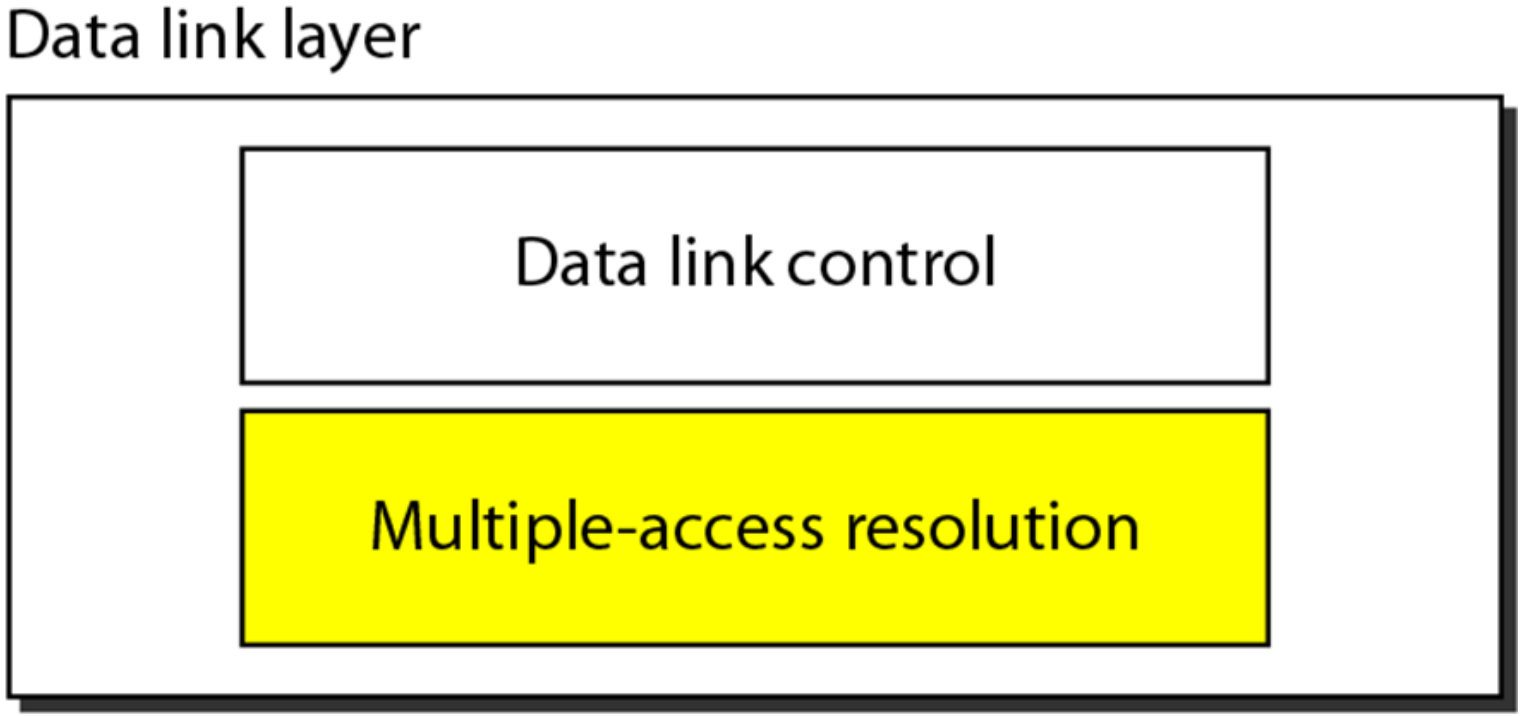
Multiple Access

Outline:

- Multiple access mechanisms
- Random access
- Controlled access
- Channelization

Somaiya TRUST

Sublayers of Data Link Layer



SOMAIYA VIDYAVIHAR UNIVERSITY

Sublayers of Data Link Layer

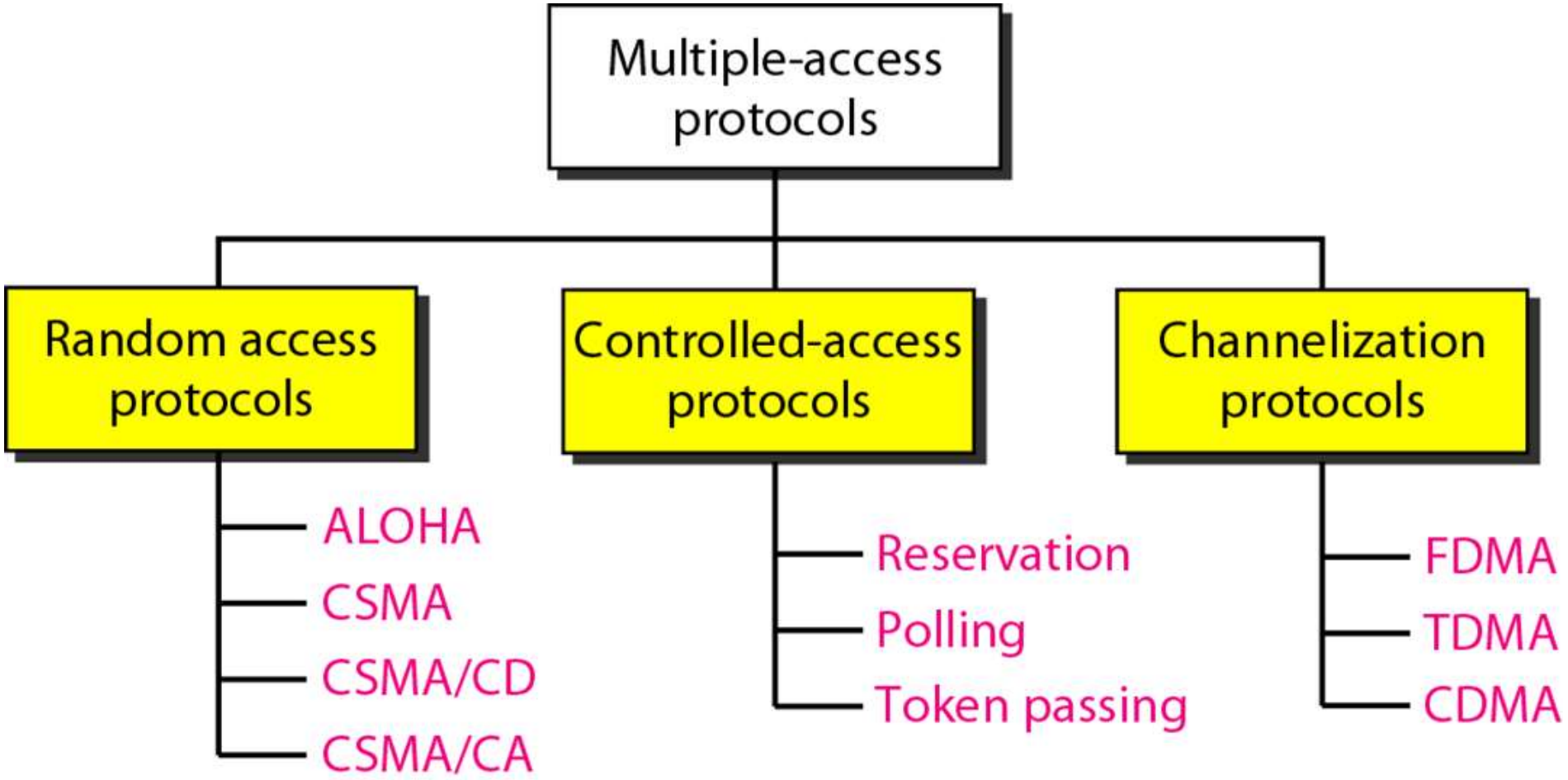
- The upper sub layer that is responsible for flow and error control is called the **logical link control (LLC) layer.**

- Lower sub layer that is mostly responsible for multiple access resolution is called the **media access control (MAC) layer.**

Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

Multiple Access Mechanisms



Somaiya TRUST

Random Access

- Also called *contention-based access*
- No station is assigned to control another
- At each instance, a station that has data to send uses a procedure defined by the protocol to make a decision on whether or not to send.

SOMAIYA VIDYAVIHAR UNIVERSITY

Random Access

- Transmission is random among the stations - *random access.*
- Stations compete with one another to access the medium - *contention methods*.
- each station has the right to the medium without being controlled by any other station
  - more than one station tries to send
  - access conflict-collision - and the frames destroyed or modified.

Somaiya TRUST

ALOHA Network

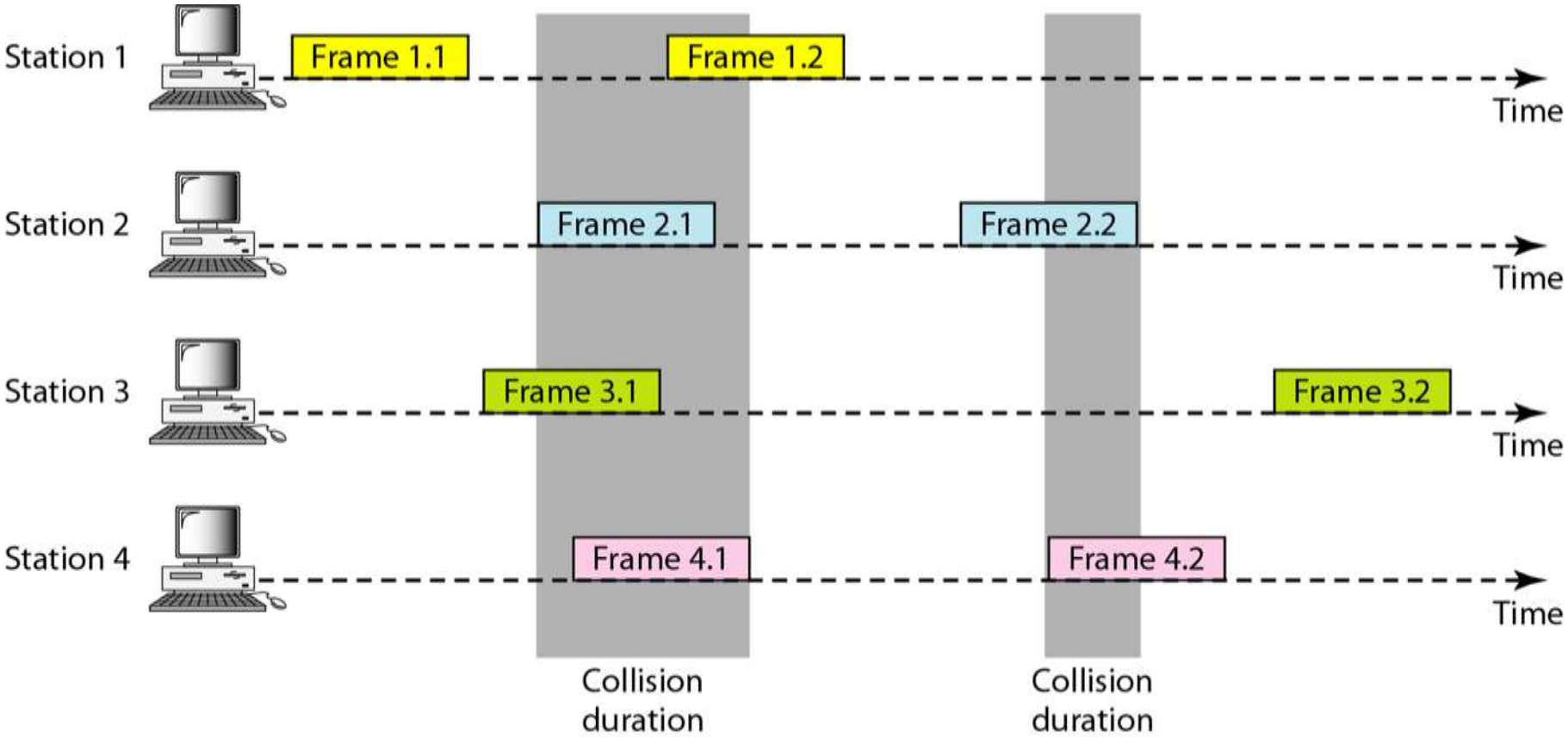
When a station sends data, another station may attempt to do so at the same time. The data from the two stations collide and become garbled.

- ✓ Pure Aloha
- ✓ Slotted Aloha

Pure ALOHA

- The idea is that each station sends a frame whenever it has a frame to send.
- Since there is only one channel to share, there is the possibility of collision between frames from different stations.
- Even if one bit of a frame coexists on the channel with one bit from another frame, there is a collision, and both will be destroyed.

Frames in Pure ALOHA



Pure ALOHA

- Resend the frames that have been destroyed during transmission - acknowledgments from the receiver.
- Timeout – resend the frame
- Pure ALOHA - when the time-out period passes, each station waits a random amount of time before resending its frame.

Pure ALOHA

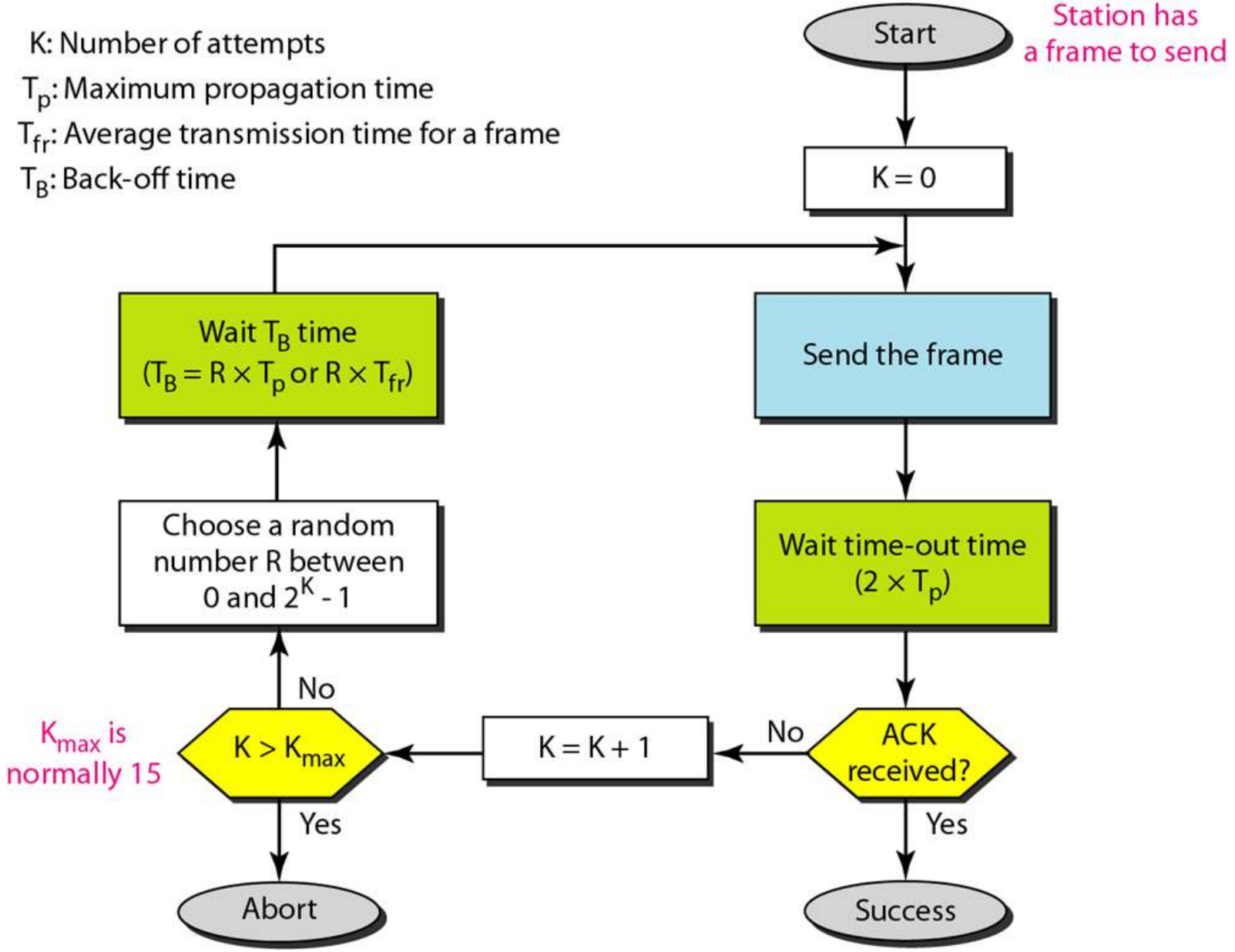
- The randomness will help avoid more collisions.
- *Random time – Back off time* T_B
- Second method to prevent congesting the channel with retransmitted frames.
- Maximum number of retransmission, K_{max} , after that give up and try later.

K: Number of attempts

T_p: Maximum propagation time

T_{fr}: Average transmission time for a frame

T_B: Back-off time



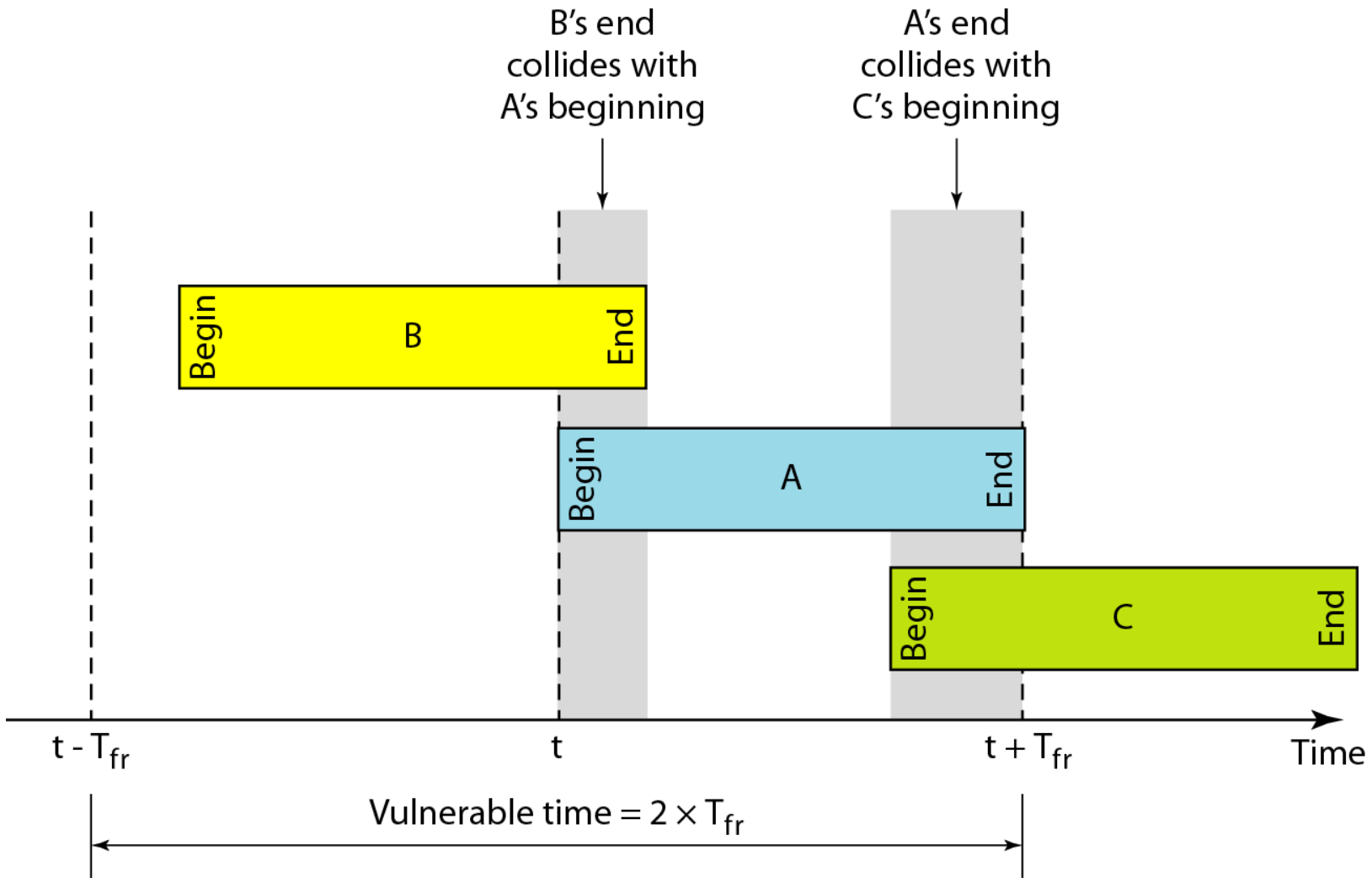
Example

Calculate possible values of T_B when stations on an ALOHA network are a maximum of 600 km apart; signal propagates at 3 x 10^8 m/s.

$$\mathbf{T_p = (600 \times 103) / (3 \times 10^8) = 2\ ms}$$

When K=1, $T_B \in \{0\ ms, 2\ ms\}$
When K=2, $T_B \in \{0\ ms, 2\ ms, 4\ ms, 6ms\}$

ALOHA: Vulnerable Time



ALOHA: Throughput

- Assume number of stations trying to transmit follow *Poisson Distribution*

- The throughput for pure ALOHA is:

$$S = G \times e^{-2G}$$

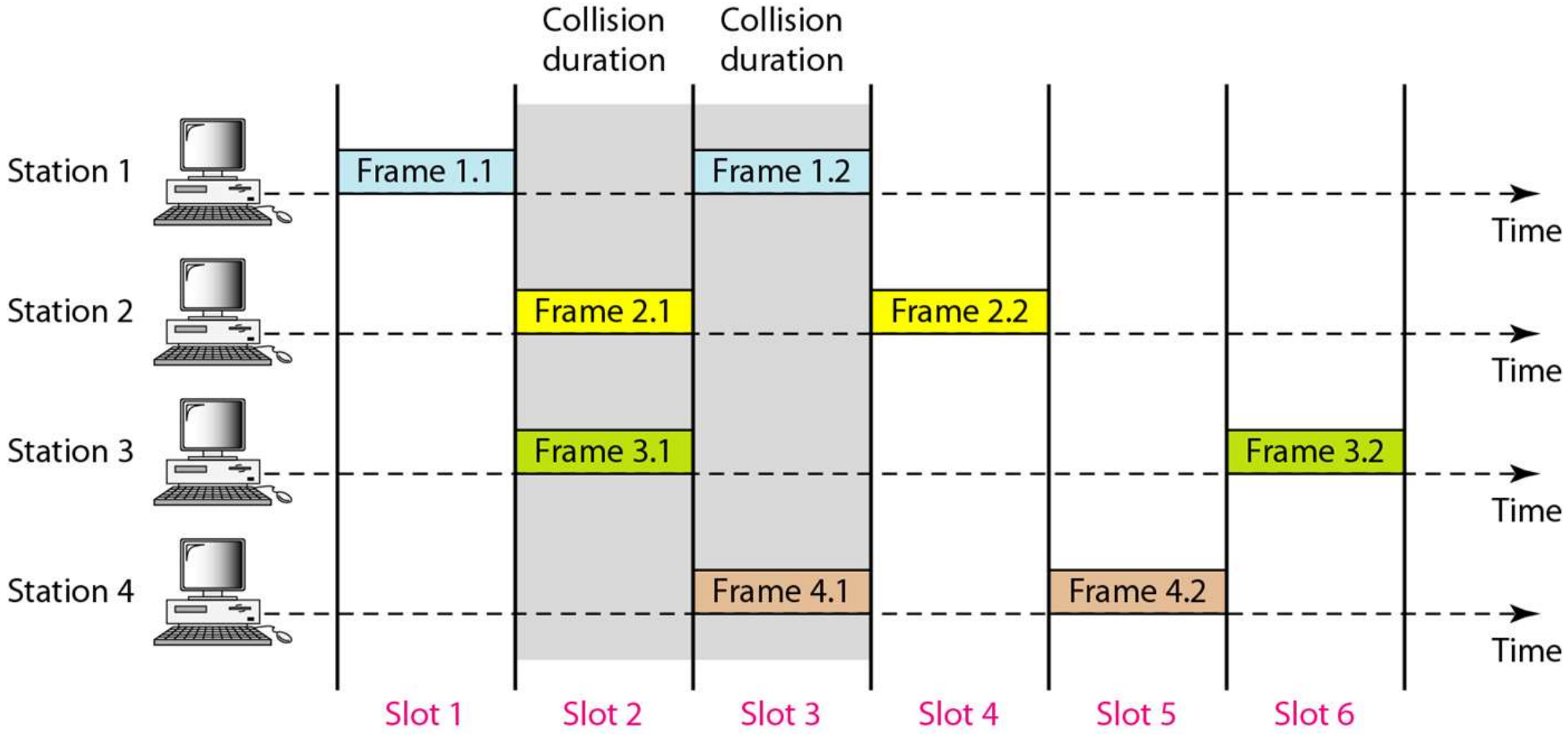
where G is the *average number of frames requested/generated by the system per frame-time*

- The maximum throughput

$$S_{max} = 0.184 \text{ when } G = 1/2$$

SOMAIYA VIDYAVIHAR UNIVERSITY

Slotted ALOHA

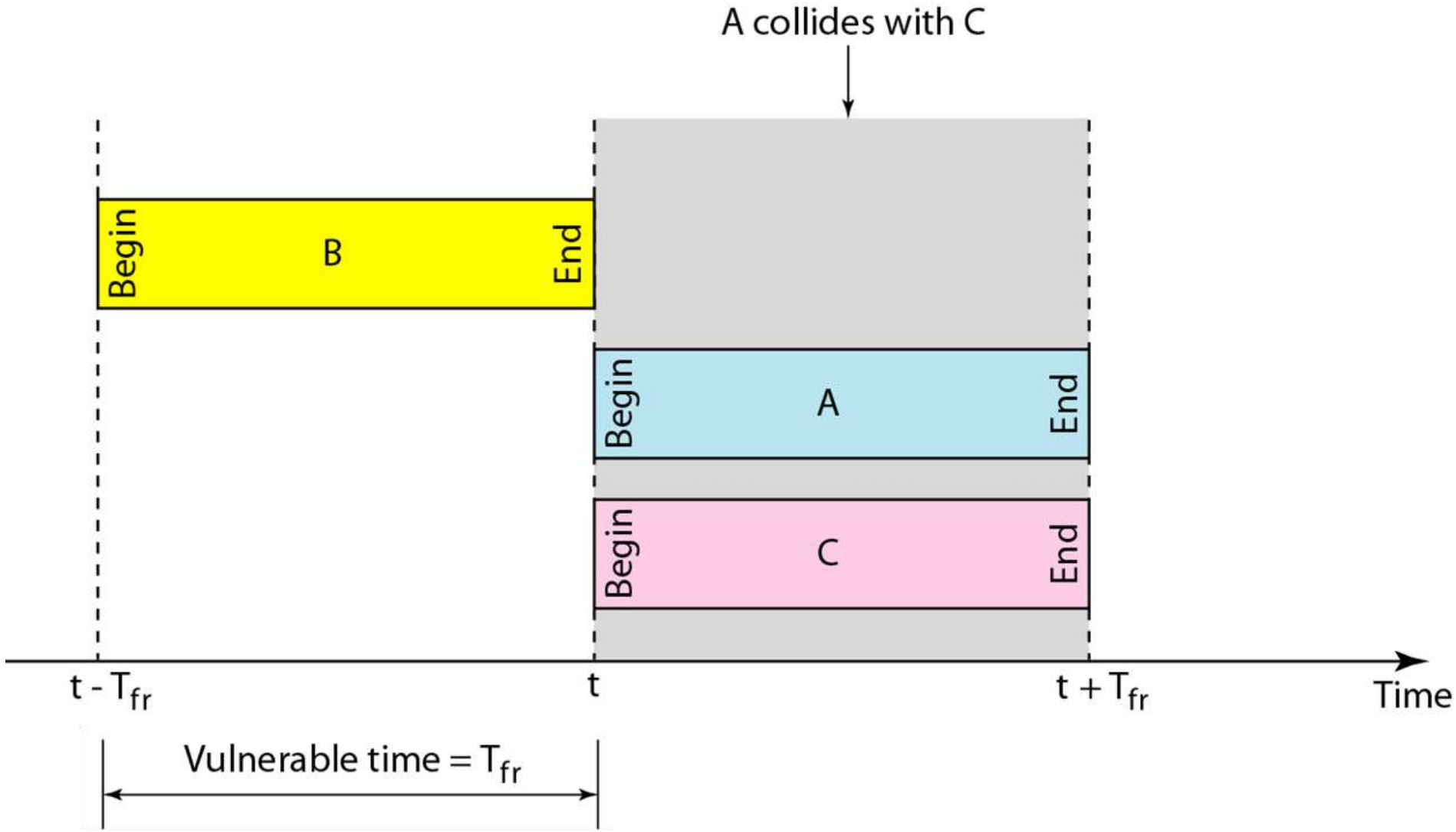


Somaiya TRUST

Slotted ALOHA

- Pure ALOHA has a vulnerable time of $2 \times T_{fr}$ - no rule that defines when the station can send.
- Slotted ALOHA was invented to *improve the efficiency* of pure ALOHA.
- Slotted ALOHA divide time into slots of T_{fr} and force the station to send only at the beginning of the time slot.

Slotted ALOHA: Vulnerable Time



Slotted ALOHA: Throughput

- The throughput for Slotted ALOHA is

$$S = G \times e^{-G}$$

where G is the average number of frames requested per frame-time

- The maximum throughput

$$S_{max} = 0.368 \text{ when } G = 1$$

Pure ALOHA: Throughput

Q. A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces

a. 1000 frames per second?
b. 500 frames per second?
c. 250 frames per second?

Pure ALOHA: Throughput calculation

The frame transmission time is 200/200 kbps or 1 ms.

a. If the system creates 1000 frames per second, or 1 frame per millisecond, then G = 1. In this case $S = G \times e^{-2G} = 0.135$ (13.5 percent). This means that the throughput is 1000 × 0.135 = 135 frames. Only 135 frames out of 1000 will probably survive.

b. If the system creates 500 frames per second, or 1/2 frames per millisecond, then G = 1/2. In this case $S = G \times e^{-2G} = 0.184$ (18.4 percent). This means that the throughput is 500 × 0.184 = 92 and that only 92 frames out of 500 will probably survive. Note that this is the maximum throughput case, percentagewise.

c. If the system creates 250 frames per second, or 1/4 frames per millisecond, then G = 1/4. In this case $S = G \times e^{-2G} = 0.152$ (15.2 percent). This means that the throughput is 250 × 0.152 = 38. Only 38 frames out of 250 will probably survive.

SOMAIYA VIDYAVIHAR UNIVERSITY

Slotted ALOHA: Throughput Calculation

Q. A slotted ALOHA network transmits 200-bit frames using a shared channel with a 200-kbps bandwidth. Find the throughput if the system (all stations together) produces

a. 1000 frames per second.
b. 500 frames per second.
c. 250 frames per second.

Somaiya TRUST

Slotted ALOHA: Throughput Calculation

This situation is similar to the previous exercise except that the network is using slotted ALOHA instead of pure ALOHA. The frame transmission time is 200/200 kbps or 1 ms.

a. In this case G is 1. So, $S = G \times e^{-G} = 0.368$ (36.8 percent). This means that the throughput is $1000 \times 0.0368 = 368$ frames. Only 368 out of 1000 frames will probably survive. Note that this is the maximum throughput case, percentagewise.
b. Here G is 1/2. In this case $S = G \times e^{-G} = 0.303$ (30.3 percent). This means that the throughput is $500 \times 0.0303 = 151$. Only 151 frames out of 500 will probably survive.
c. Now G is 1/4. In this case $S = G \times e^{-G} = 0.195$ (19.5 percent). This means that the throughput is $250 \times 0.195 = 49$. Only 49 frames out of 250 will probably survive.

CSMA

- **C**arrier **S**ense **M**ultiple **A**ccess

  "Listen before talk“

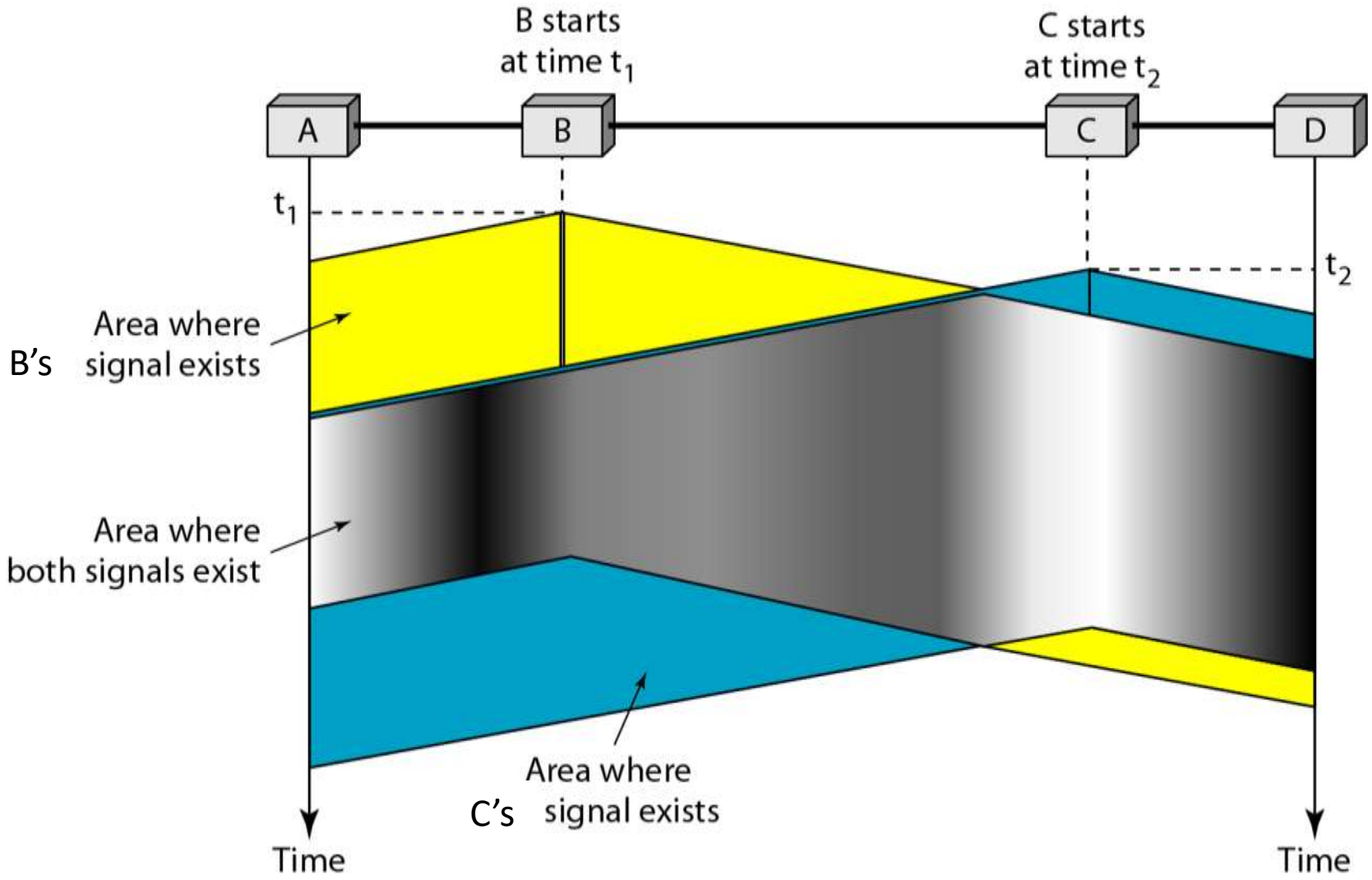
- Reduce the possibility of collision, but *cannot completely eliminate* it

- Carrier sense multiple access (CSMA) requires that each station first check the state of the medium before sending.

CSMA

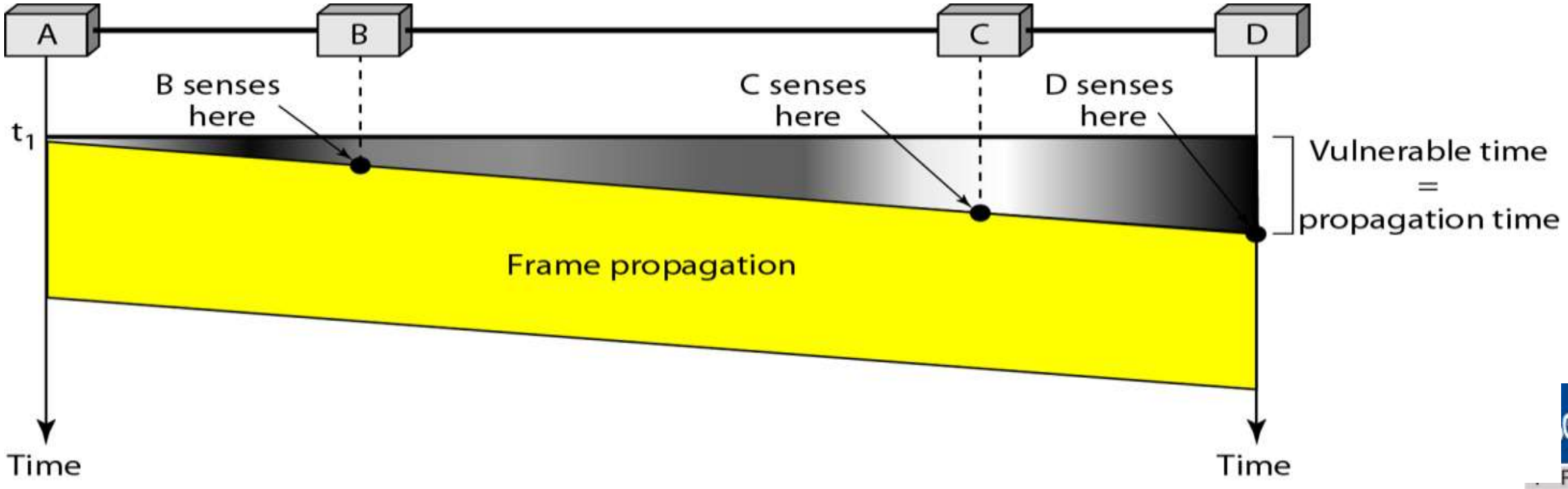
- The possibility of collision still exists because of propagation delay when a station sends a frame, it still takes time (although very short) for the first bit to reach every station and sense it.

Collision in CSMA

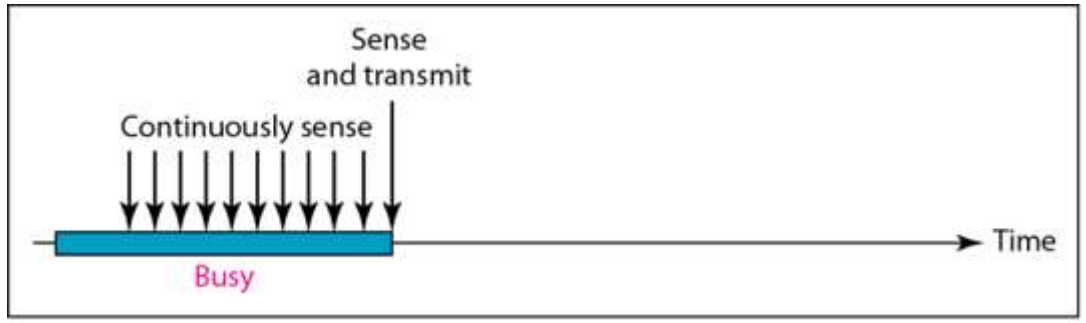


Collision in CSMA

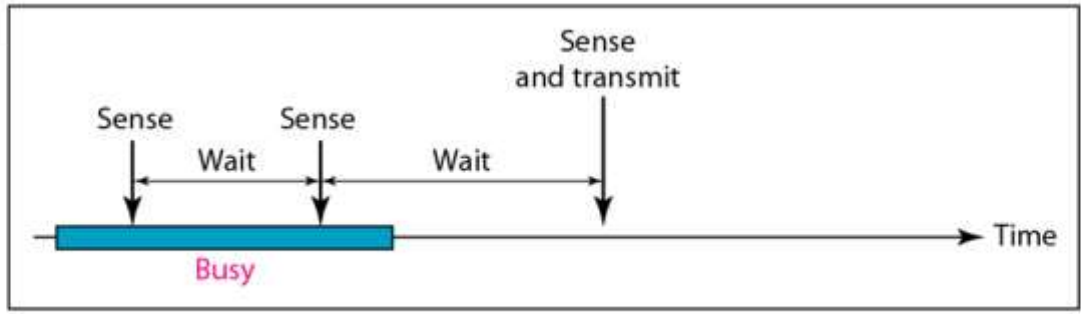
The vulnerable time for CSMA is the propagation time Tp



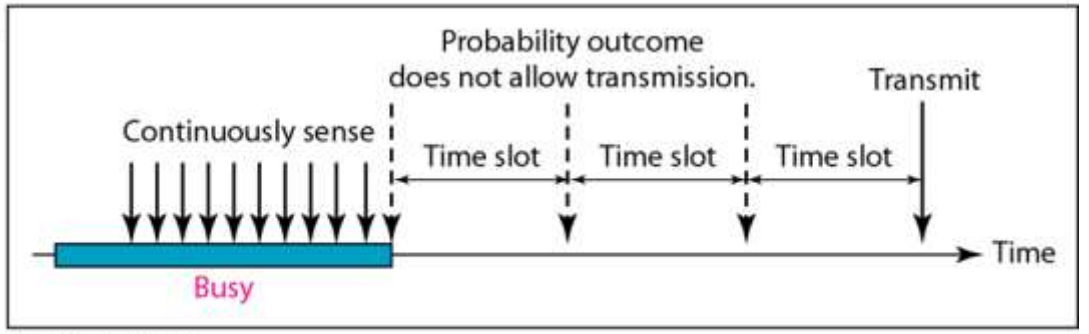
Persistence Methods



a. 1-persistent



b. Nonpersistent

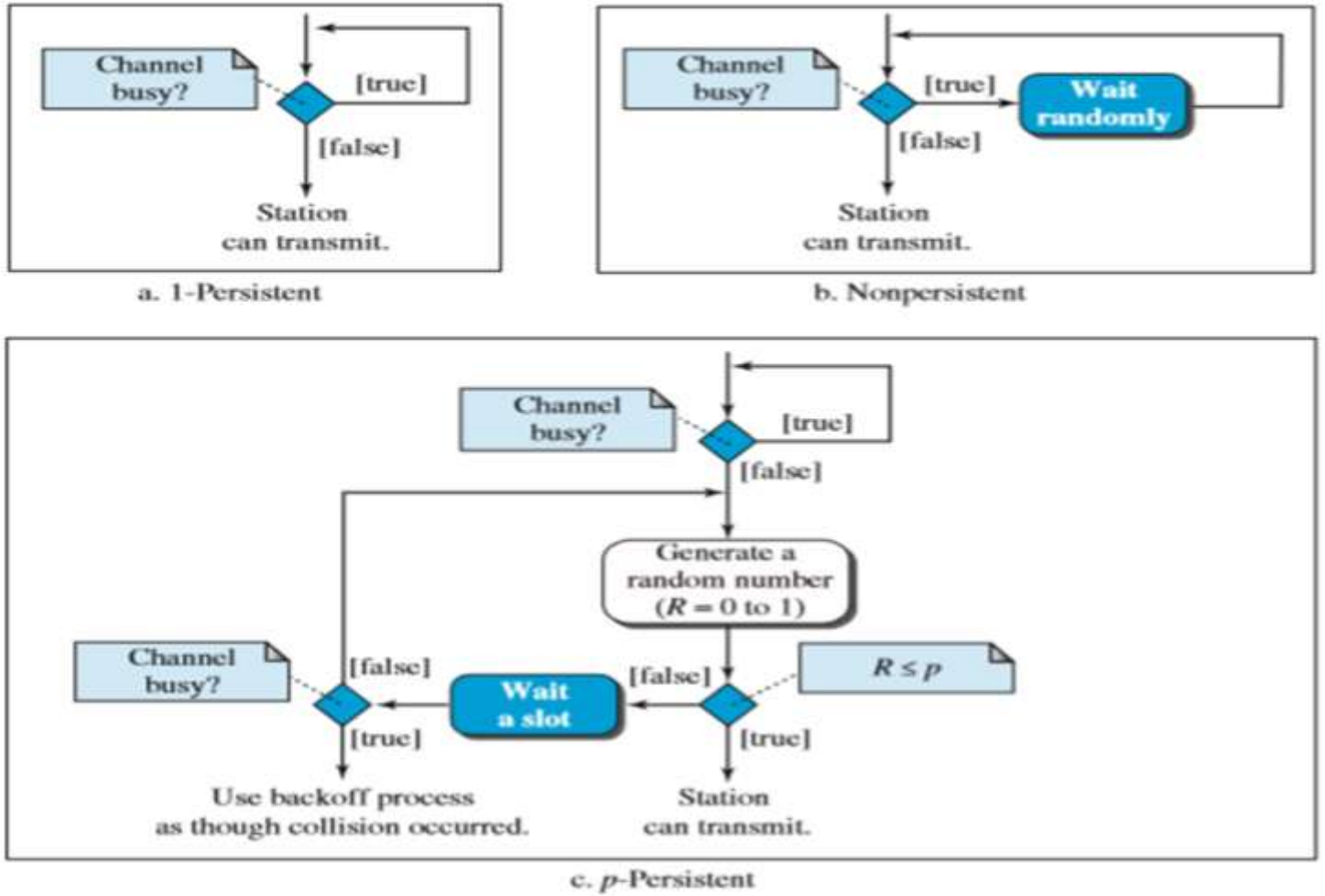


c. p-persistent

What a station does when channel is idle or busy

Persistence Methods

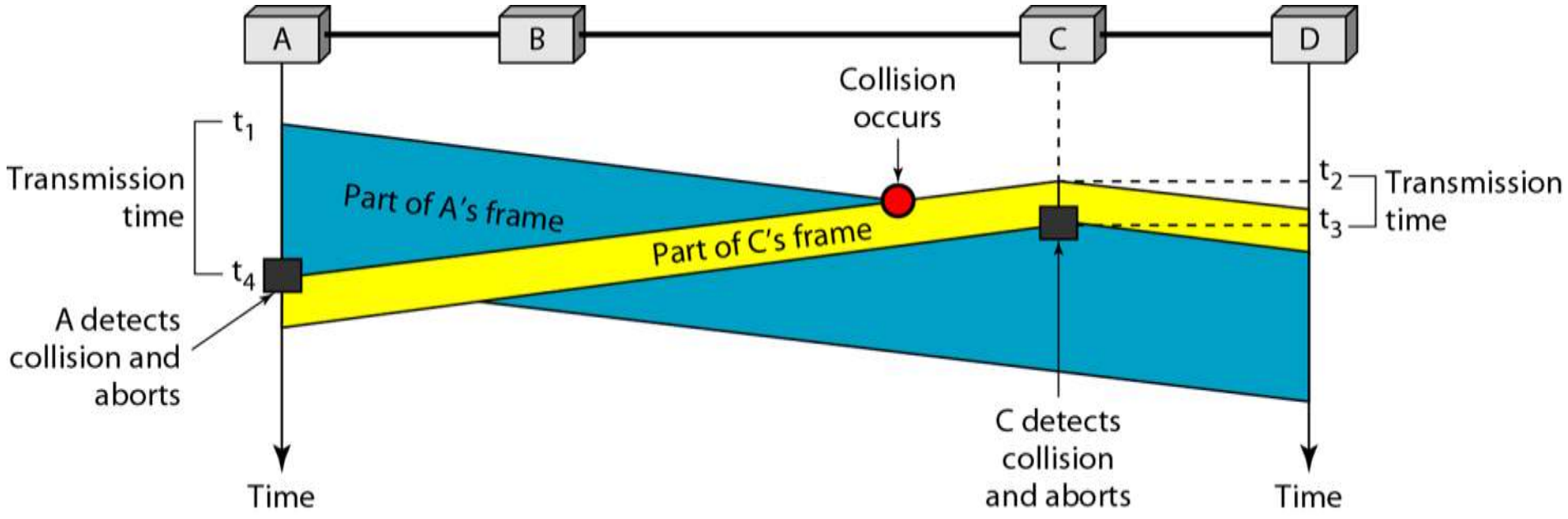
Figure 12.10 *Flow diagram for three persistence methods*



CSMA/CD

- **C**arrier **S**ense **M**ultiple **A**ccess with **C**ollision **D**etection

- Station monitors channel while sending a frame.



CSMA/CD

- (CSMA/CD) augments the algorithm to handle the collision.
- a station monitors the medium after it sends a frame to see if the transmission was successful.
- For CSMA/CD to work, we need a restriction on the frame size.

CSMA/CD

- Before sending last bit, station must detect a collision because once entire frame is sent, station does not keep copy of frame.

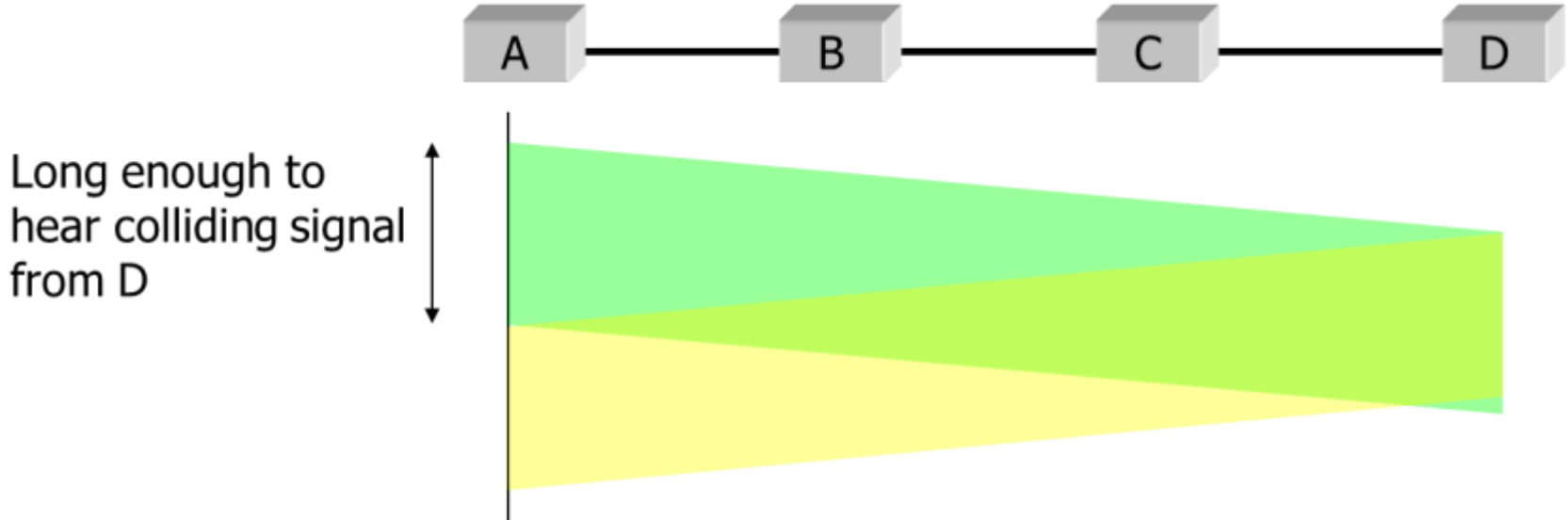
- Frame Transmission time T_{fr} must be atleast

$$2^* T_P$$

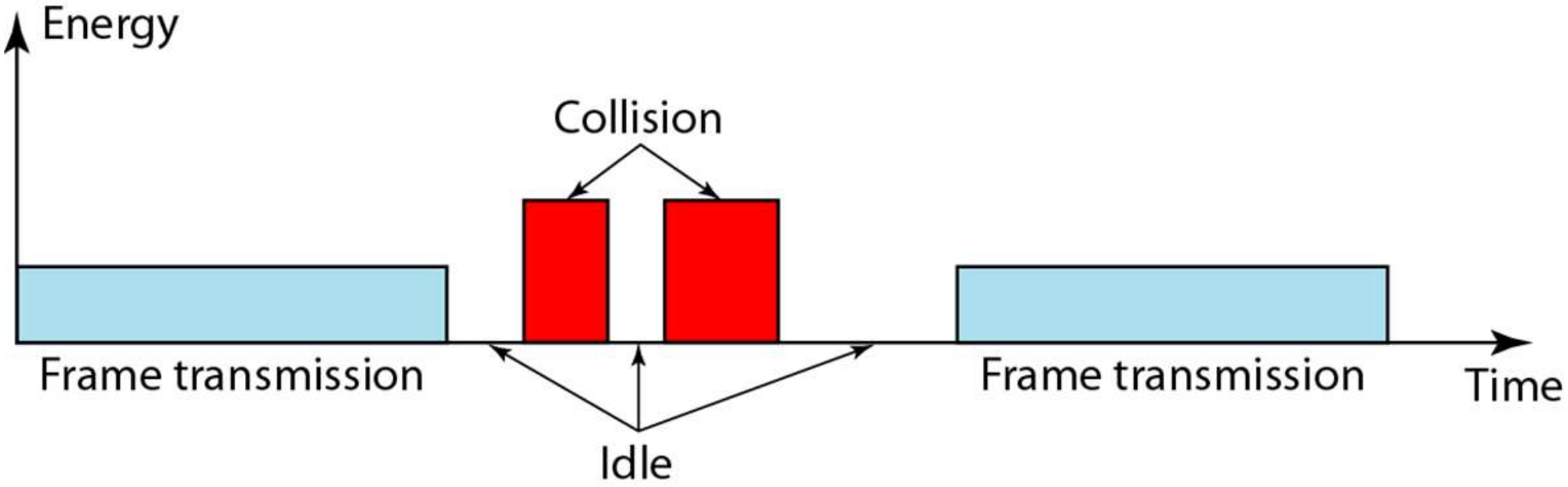
i.e. twice the maximum Propagation time.

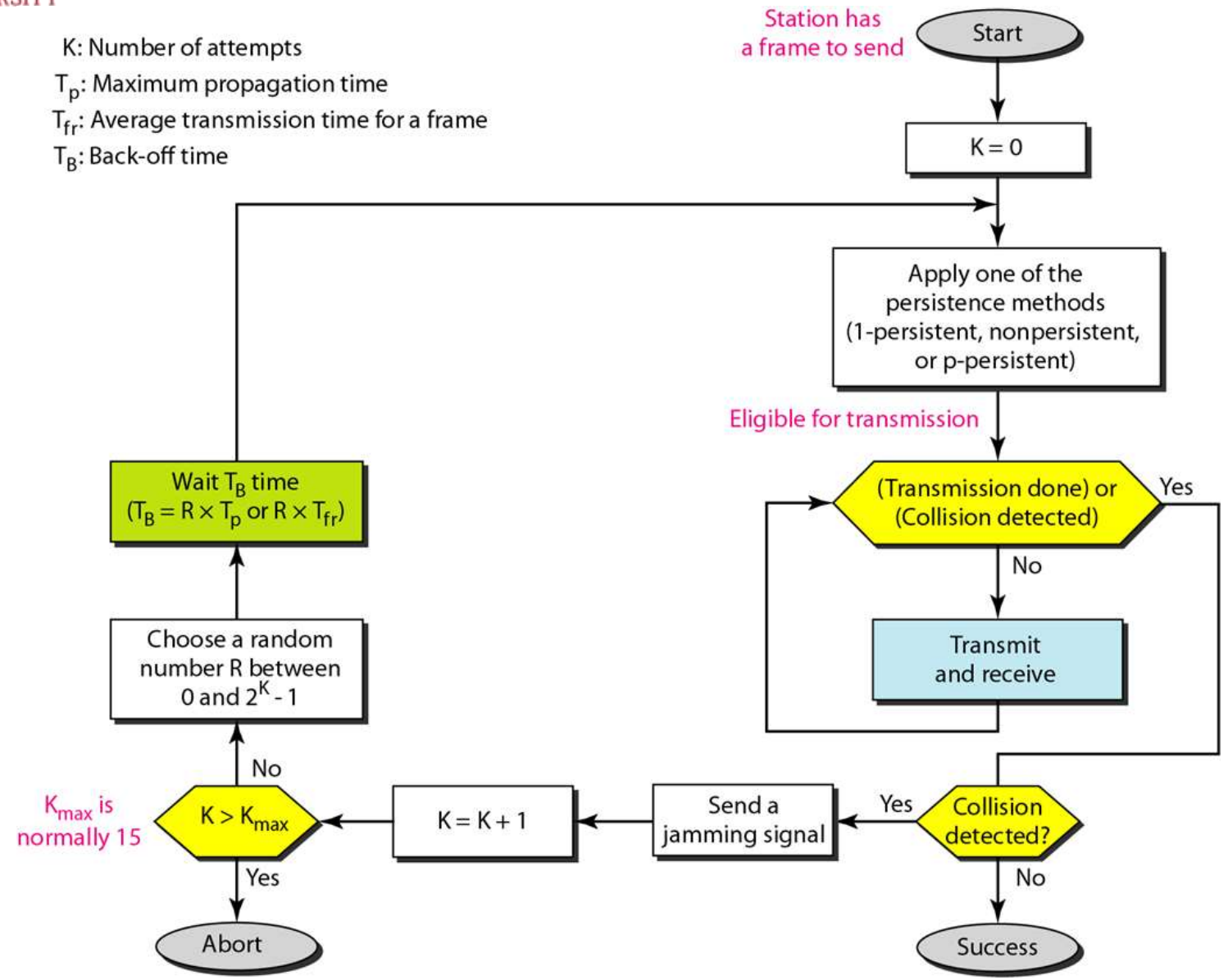
CSMA/CD: Minimum Frame Size

- Each frame must be large enough for a sender to detect a collision.
- Worst case scenario:
  - "A" is transmitting
  - "D" starts transmitting just before A's signal arrives



CSMA/CD: Energy Levels




K: Number of attempts
T_p: Maximum propagation time
T_{fr}: Average transmission time for a frame
T_B: Back-off time
Station has a frame to send
Start
K = 0
Apply one of the persistence methods (1-persistent, nonpersistent, or p-persistent)
Eligible for transmission
(Transmission done) or (Collision detected)
Yes
No
Transmit and receive
Collision detected?
Yes
Send a jamming signal
K = K + 1
$K > K_{max}$
No
K_{max} is normally 15
Yes
Abort
No
Success
Choose a random number R between 0 and $2^K - 1$
Wait T_B time ($T_B = R \times T_p$ or $R \times T_{fr}$)

Difference ALOHA and CSMA/CD

- first difference is the addition of the persistence process (for sensing the medium).

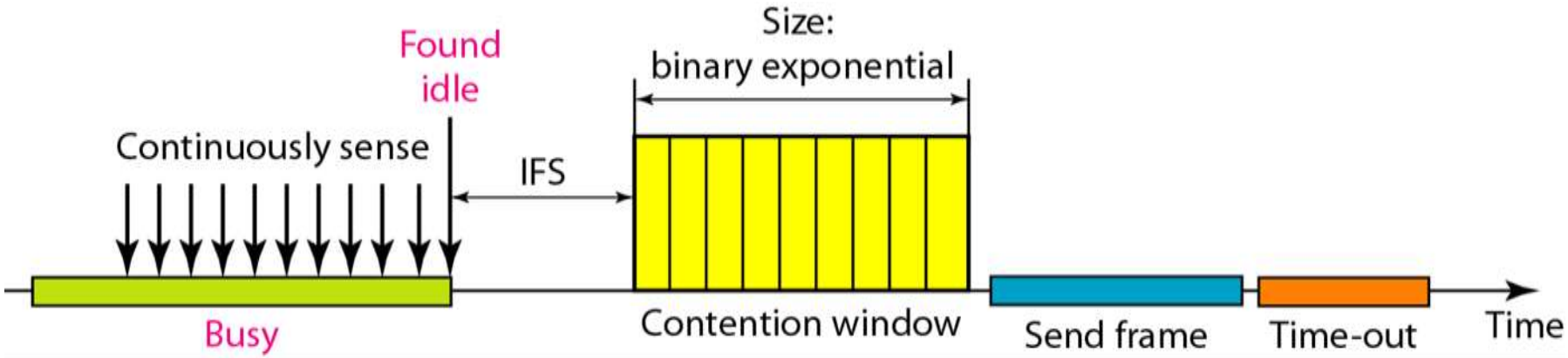
- second difference is the frame transmission.
  - In ALOHA, *first transmit the entire frame and then wait for an acknowledgment*.
  - In CSMA/CD, *transmission and collision detection is a continuous process.*

- The third difference is the sending of a short jamming signal that enforces the collision in case other stations have not yet sensed the collision.

CSMA/CA

- **C**arrier **S**ense **M**ultiple **A**ccess with **C**ollision **A**voidance

- Used in a network where collision cannot be detected.
  - E.g., wireless LAN



IFS – Interframe Space

CSMA/CA

- need to avoid collisions on wireless networks because they cannot be detected.

- Collisions are avoided through the use of CSMA/CA's three strategies:
  - Inter-frame space
  - Contention Window
  - Acknowledgments

CSMA/CA

Interframe Space (IFS)

- collisions are avoided by deferring transmission even if the channel is found idle.

- idle channel is found - does not send immediately – waits for period of time called IFS.

- The IFS variable can also be used to prioritize stations or frame types. For example, a station that is assigned a shorter IFS has a higher priority.

SOMAIYA VIDYAVIHAR UNIVERSITY

CSMA/CA

Contention Window

- The contention window is an amount of time divided into slots.
- A station that is ready to send chooses a random number of slots as its wait time.
- No. of slots in window changes according to binary exponential back-off strategy.
- Channel set to one slot the first time and then doubles each time the station cannot detect an idle channel after the IFS time.

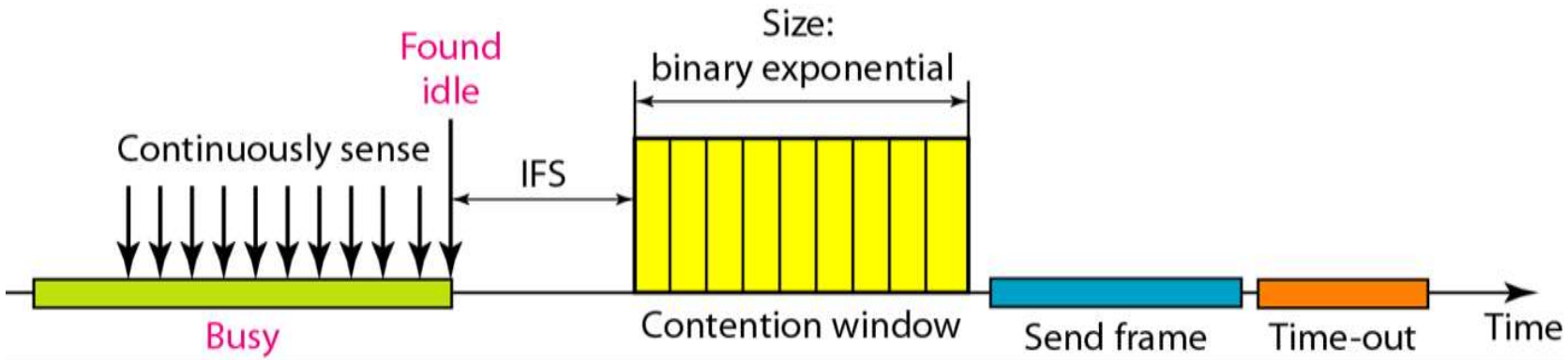
Somaiya TRUST

CSMA/CA

Contention Window

- station needs to sense the channel after each time slot.

- In CSMA/CA, if the station finds the channel busy, it does not restart the timer of the contention window; it stops the timer and restarts it when the channel becomes idle.

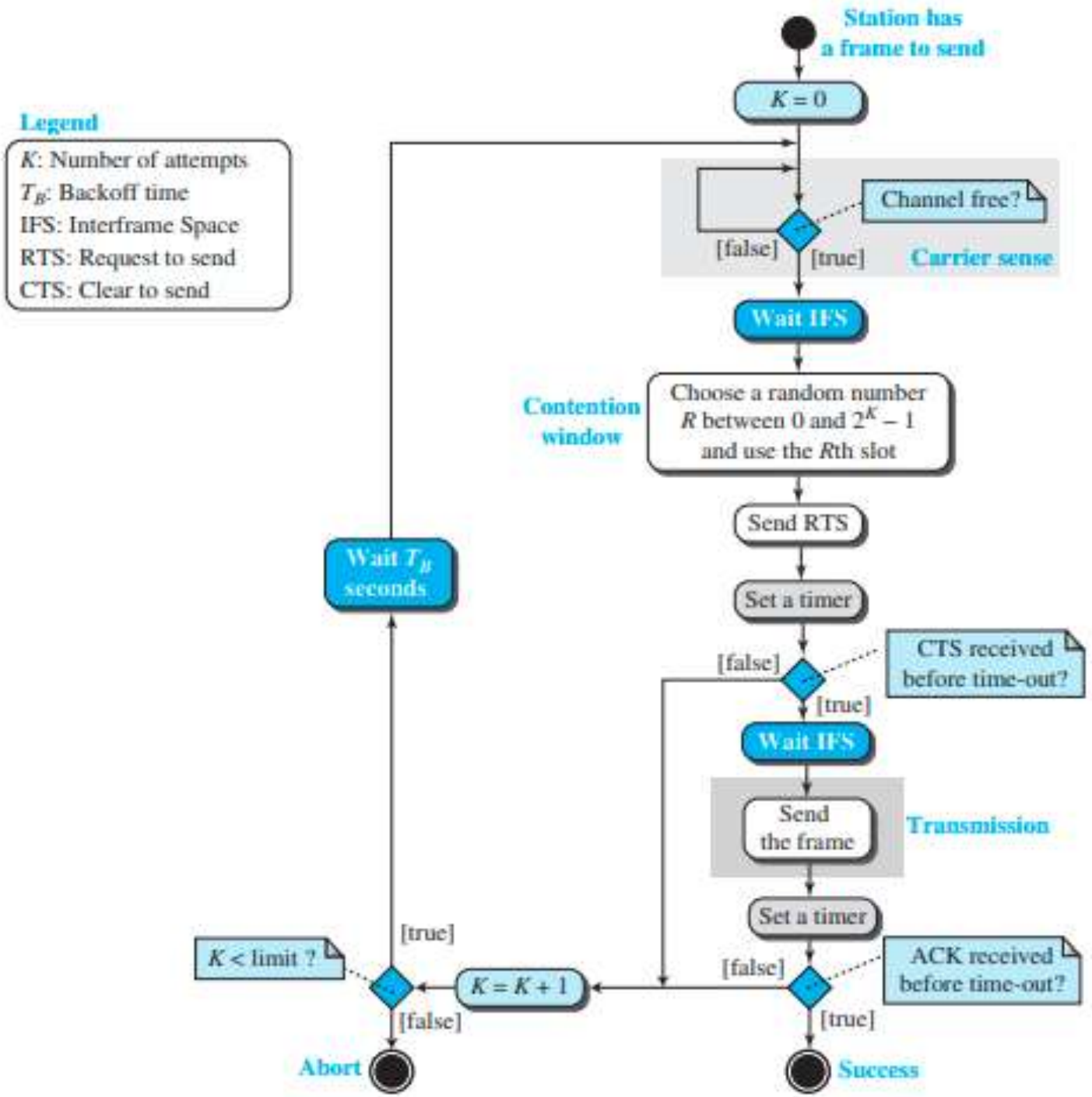


IFS – Interframe Space

CSMA/CA

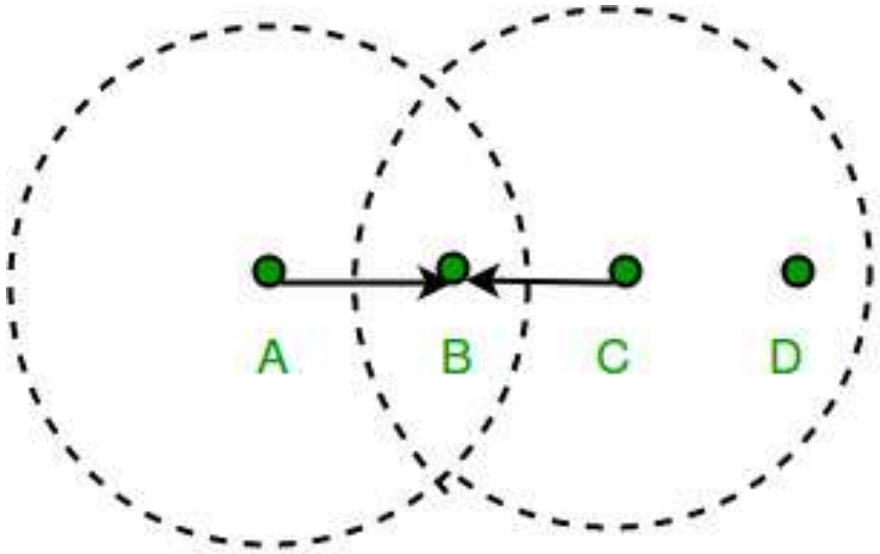
Acknowledgment

- There still may be a collision resulting in destroyed data.
- The data may be corrupted during the transmission.
- The positive acknowledgment and the time-out timer can help guarantee that the receiver has received the frame.

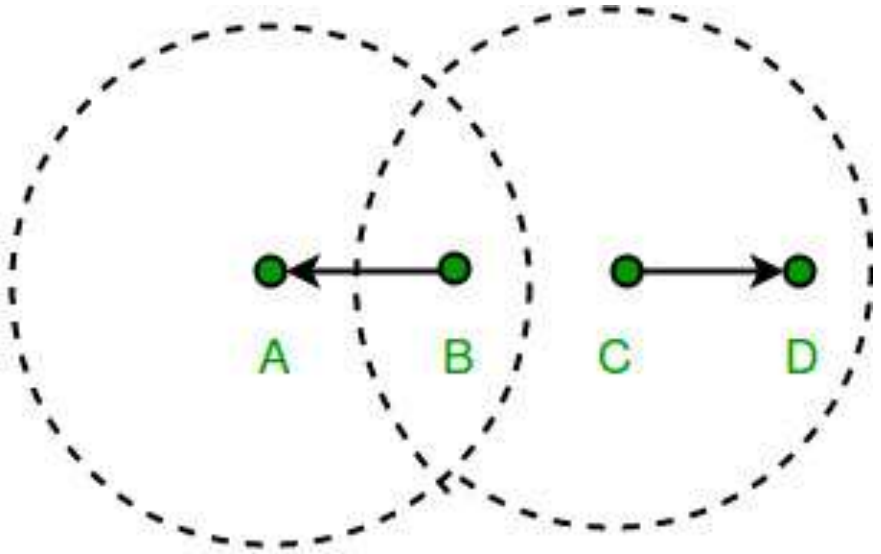

Legend
K: Number of attempts
T_B: Backoff time
IFS: Interframe Space
RTS: Request to send
CTS: Clear to send
Station has a frame to send
K = 0
Channel free?
[false]
[true]
Carrier sense
Wait IFS
Contention window
Choose a random number R between 0 and $2^K - 1$ and use the Rth slot
Send RTS
Set a timer
CTS received before time-out?
[false]
[true]
Wait IFS
Send the frame
Transmission
Set a timer
ACK received before time-out?
[false]
[true]
K = K + 1
K < limit ?
[true]
[false]
Wait T_B seconds
Abort
Success

SOMAIYA VIDYAVIHAR UNIVERSITY

CSMA/CA: Hidden and Exposed Terminal Problem**



Hidden Node Problem



Exposed Node Problem

Somaiya TRUST

Controlled Access

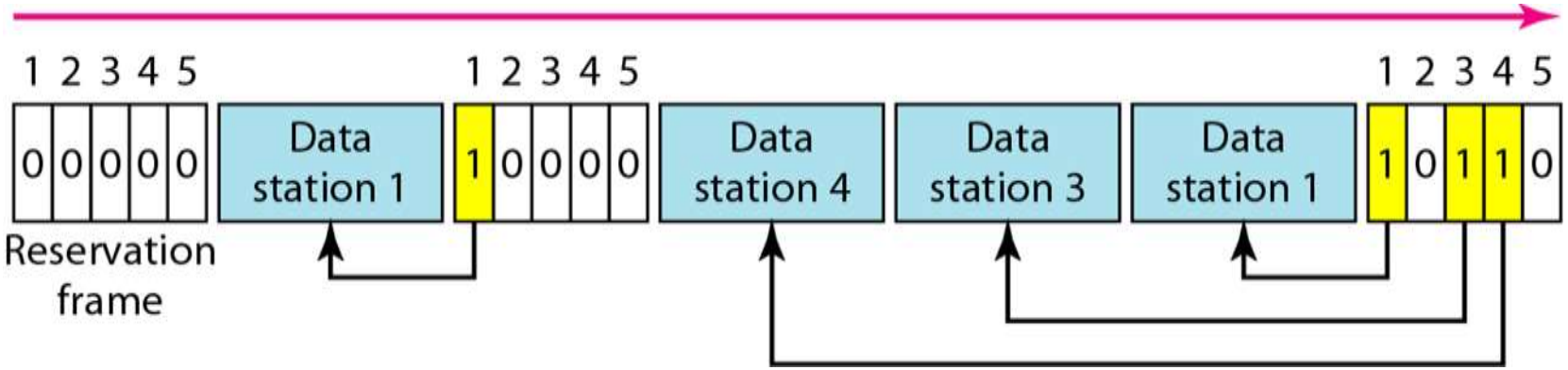
In controlled access, the stations consult one another to find which station has the right to send.

- A station must be authorized by someone (e.g., other stations) before transmitting
- Three common methods:
    - Reservation
    - Polling
    - Token passing

Reservation

- a station needs to make a reservation before sending data.
- Time is divided into intervals.
- a reservation frame precedes the data frames sent in that interval.
- N stations in the system, there are exactly N reservation mini-slots

Reservation Method



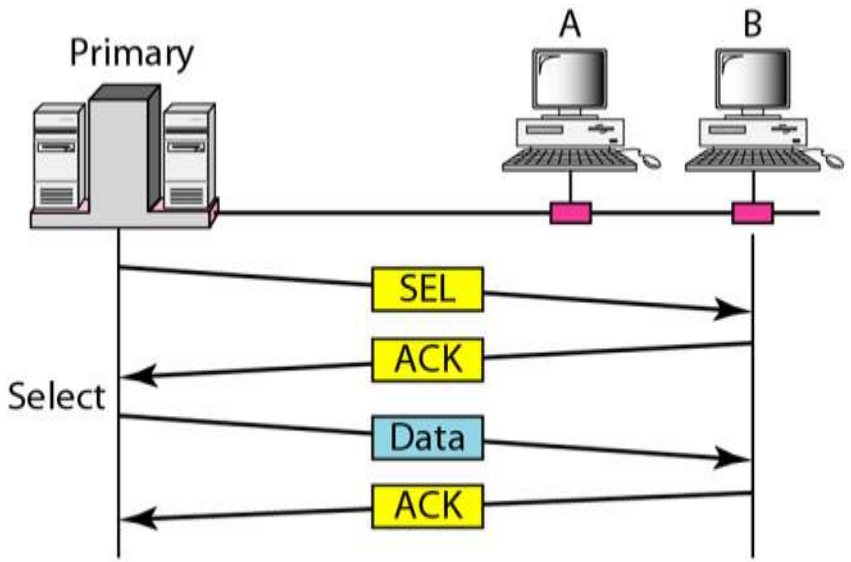
Polling Method

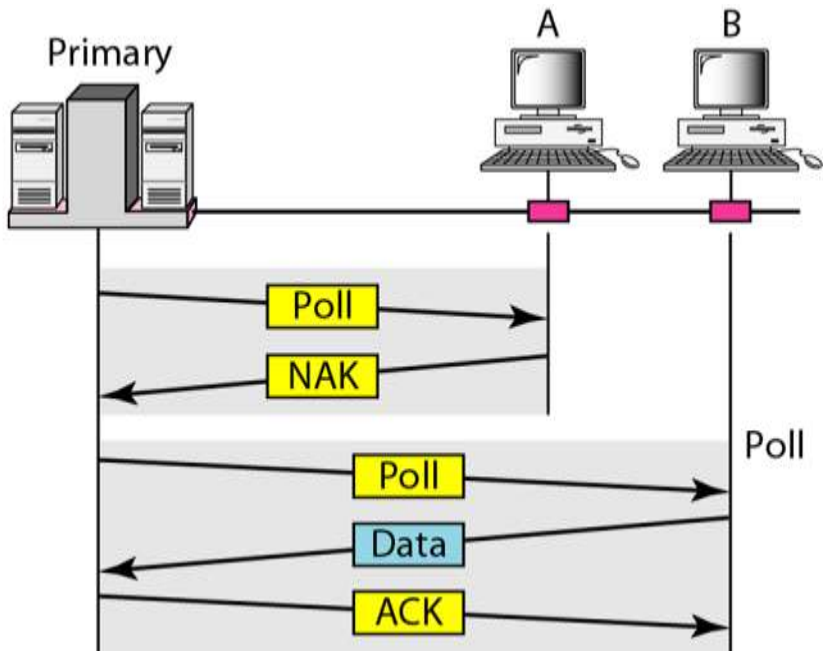
- Polling works with topologies in which one device is designated as a primary station and the other devices are secondary stations.
- All data exchanges must be made through the primary device.
- It is up to the primary device to determine which device is allowed to use the channel

Polling Method

- primary device - always the initiator of a session
- The select function is used whenever the primary device has something to send.
- The poll function is used by the primary device to solicit transmissions from the secondary devices.

Polling Method





Token Passing

- the stations in a network are organized in a logical ring.
- for each station, there is a predecessor and a successor.
- current station is the one that is accessing the channel now.
- right to this access has been passed from the predecessor to the current station.

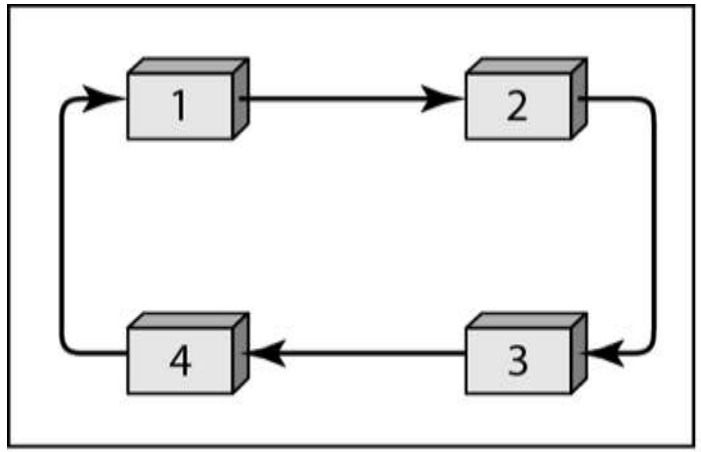
Token Passing

- right will be passed to the successor when the current station has no more data to send.
- a special packet called a token circulates through the ring.
- Token management is needed for this access method.
- Stations must be limited in the time they can have possession of the token.

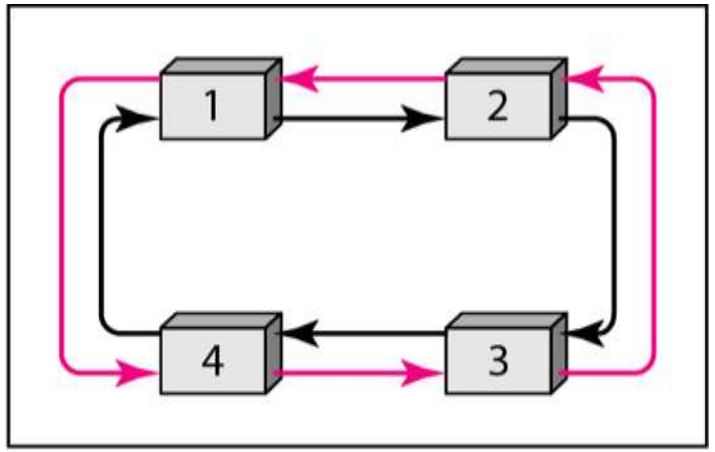
Token Passing

- token must be monitored to ensure it has not been lost or destroyed.
- assign priorities to the stations and to the types of data being transmitted.
- make low-priority stations release the token to high priority stations.

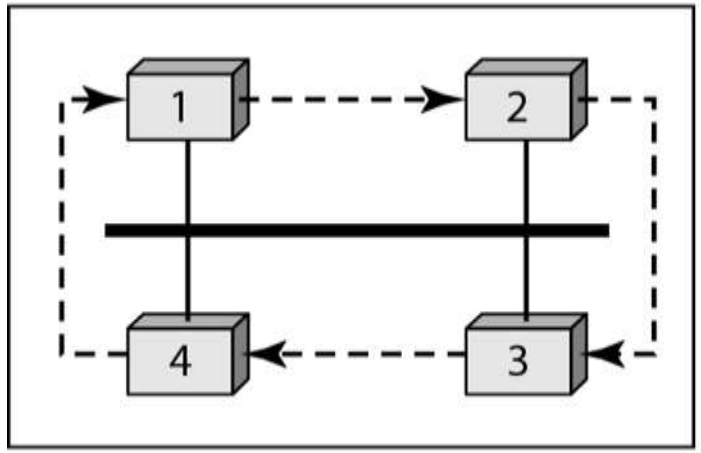
Token Passing



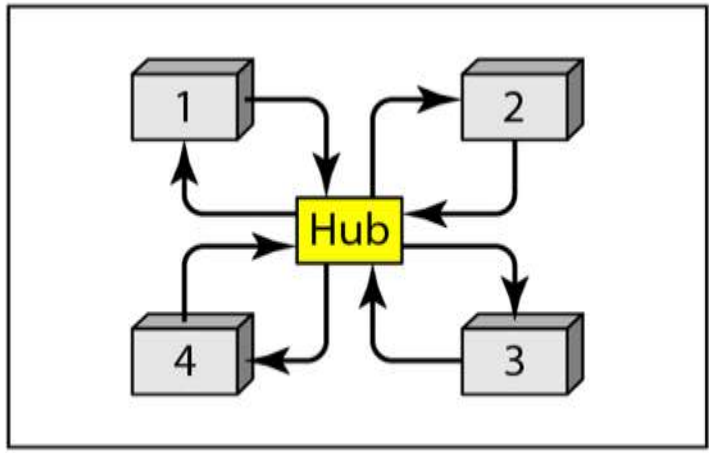
a. Physical ring



b. Dual ring



c. Bus ring



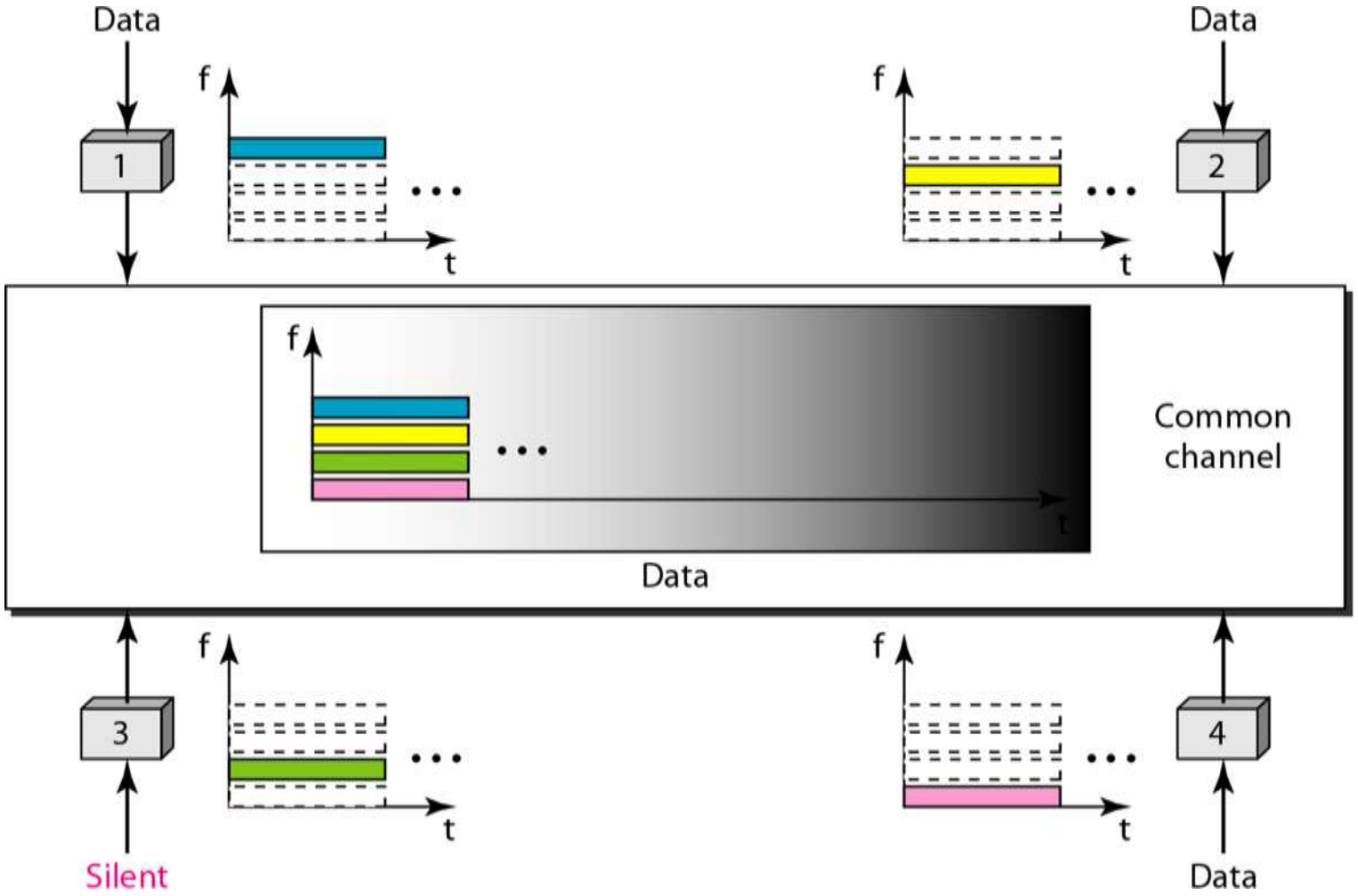
d. Star ring

Channelization

- Similar to multiplexing

- Three schemes
  - Frequency-Division Multiple Access (FDMA)
  - Time-Division Multiple Access (TDMA)
  - Code-Division Multiple Access (CDMA)

FDMA



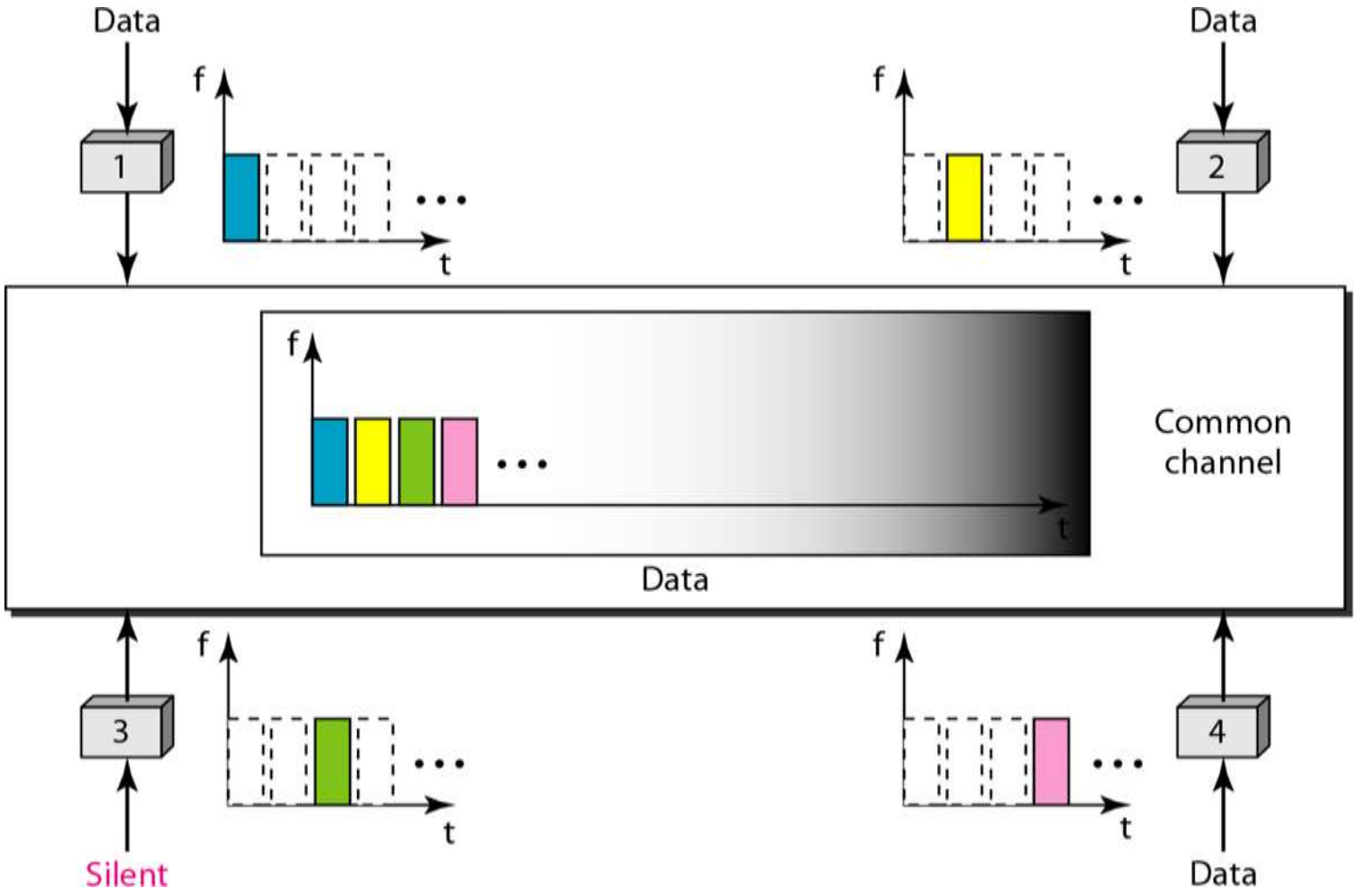
SOMAIYA VIDYAVIHAR UNIVERSITY

FDMA and FDM

- FDM is a physical layer multiplexing technique, while FDMA is a data link layer access method.
- Using FDM to allow multiple users to utilize the same bandwidth is called FDMA.
- FDM uses a physical multiplexer, while FDMA does not.

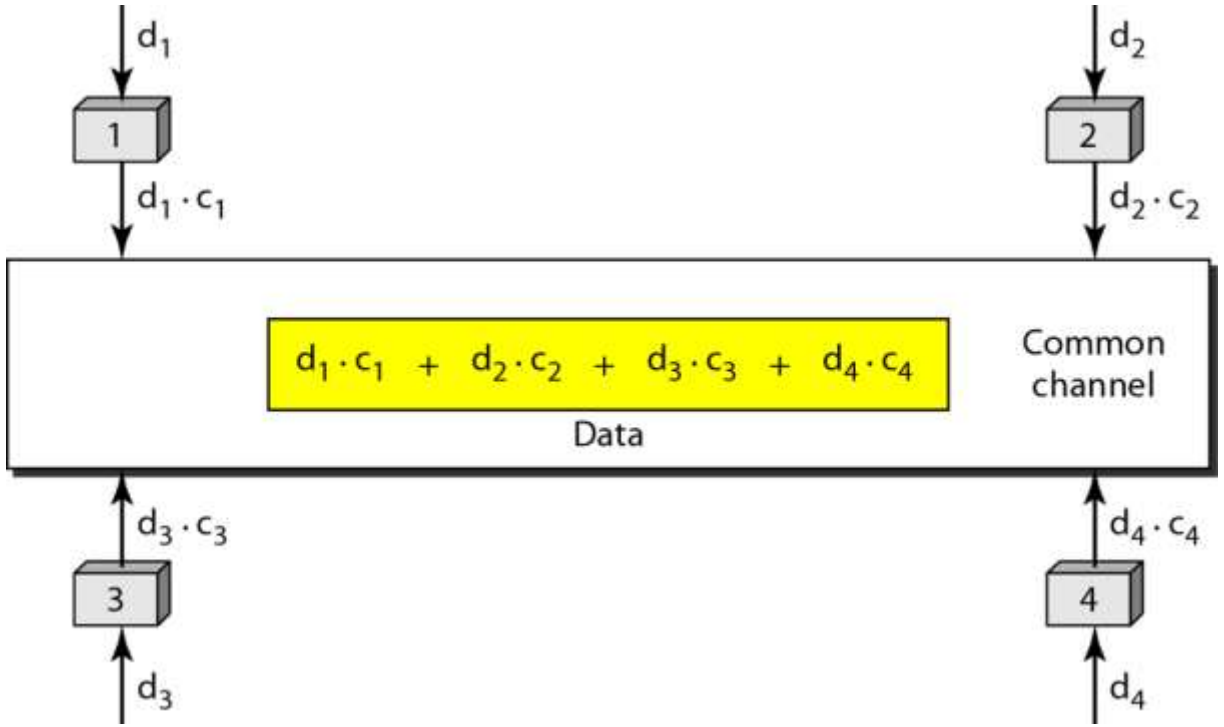
Somaiya TRUST

TDMA



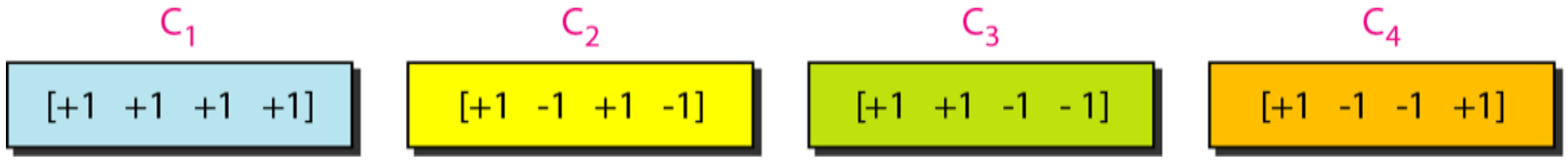
CDMA

- One channel carries all transmissions at the same time
- Each channel is separated by code.



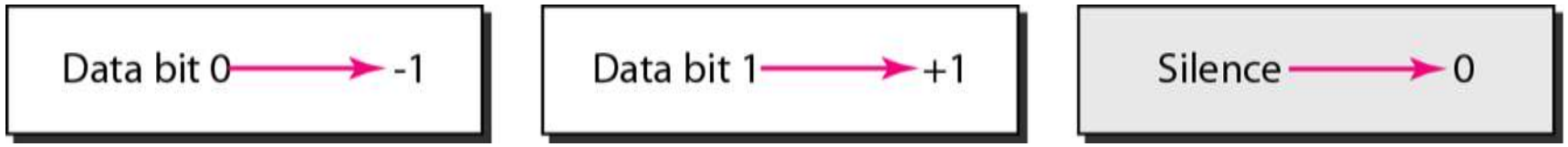
CDMA: Chip Sequences

- Each station is assigned a unique chip sequence

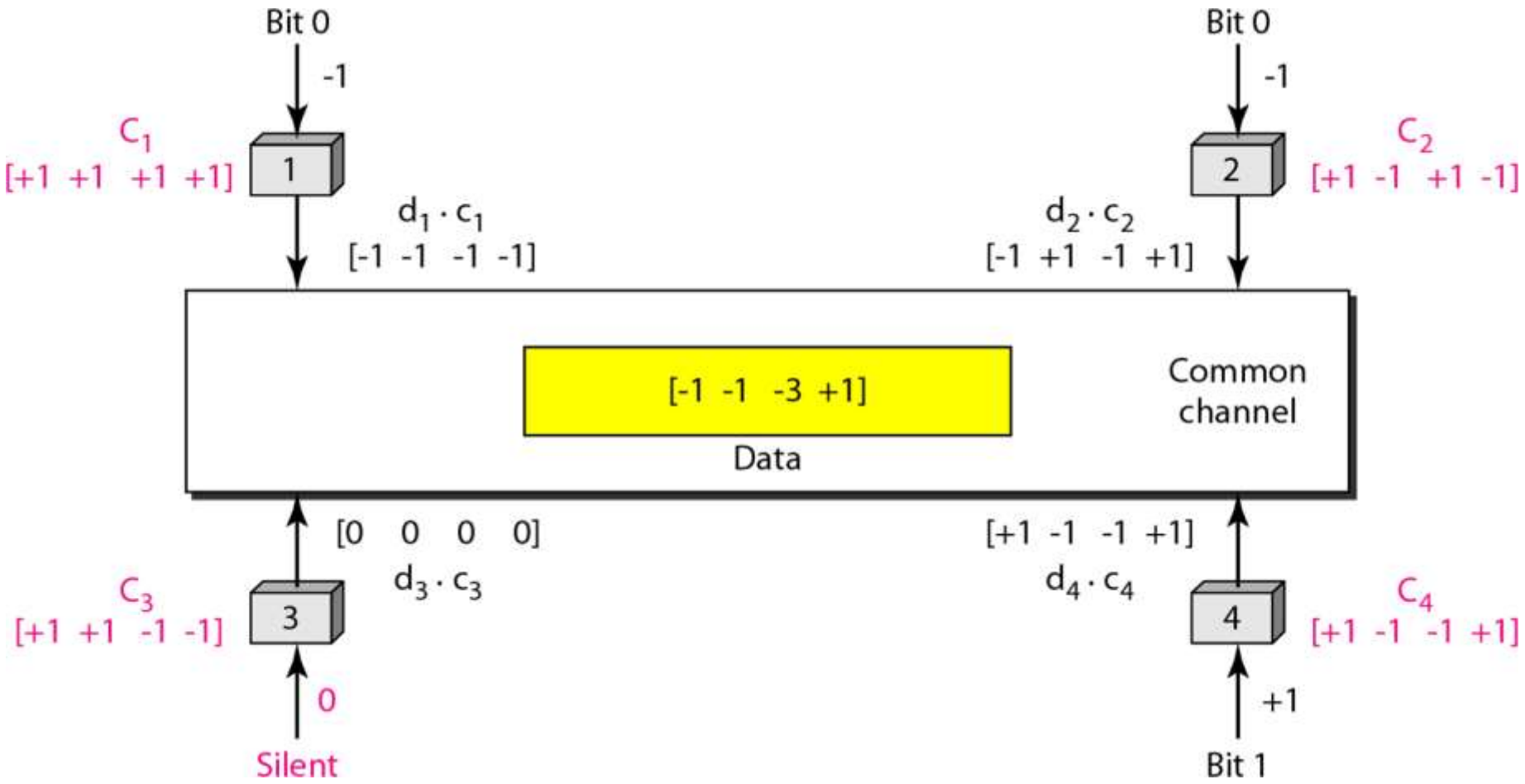


- Chip sequences are *orthogonal vectors*
  - Inner product of any pair must be zero
- With N stations, sequences must have the following properties:
  - They are of length N
  - Their self inner product is always N

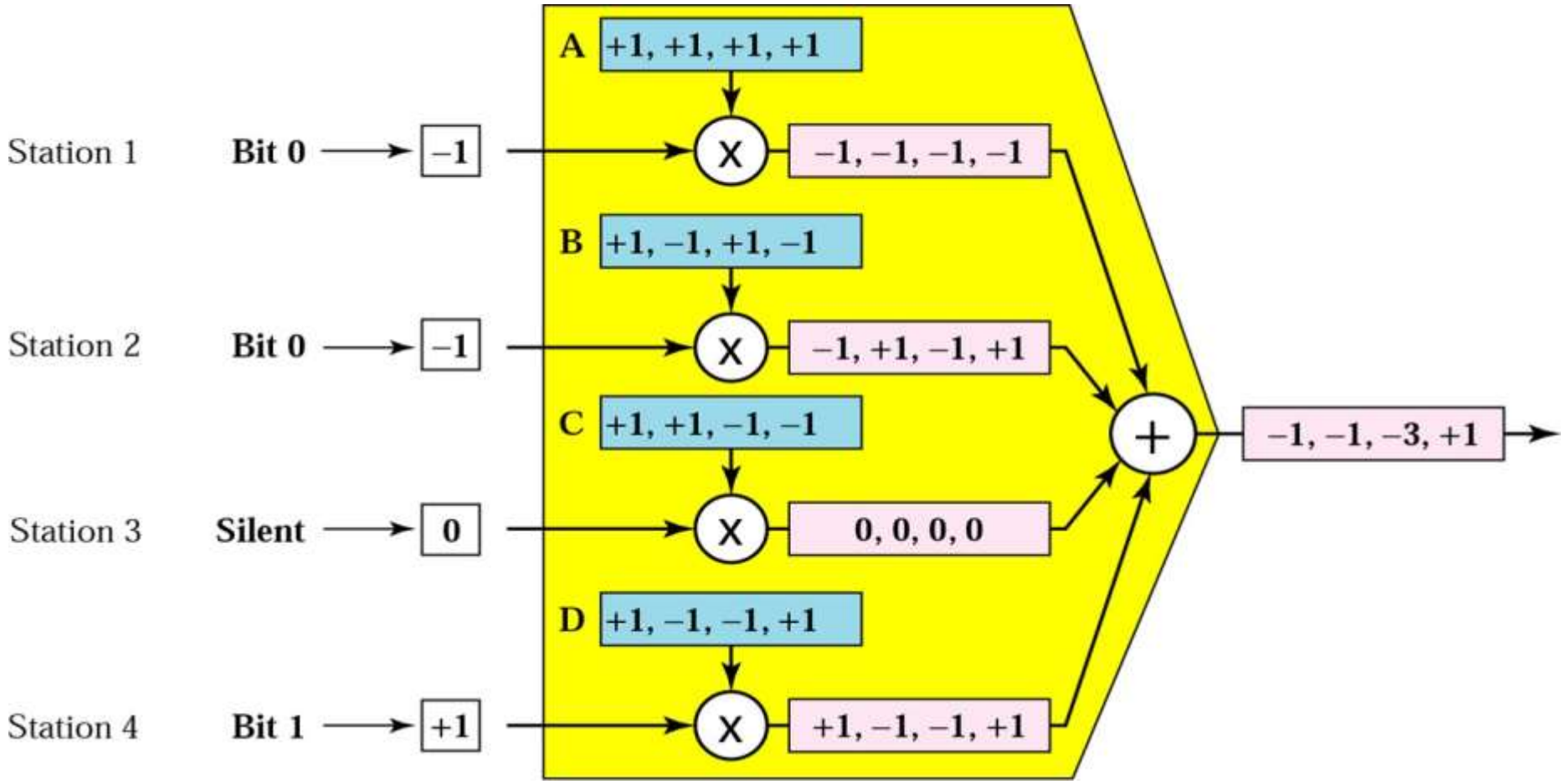
CDMA: Bit Representation



Transmission in CDMA

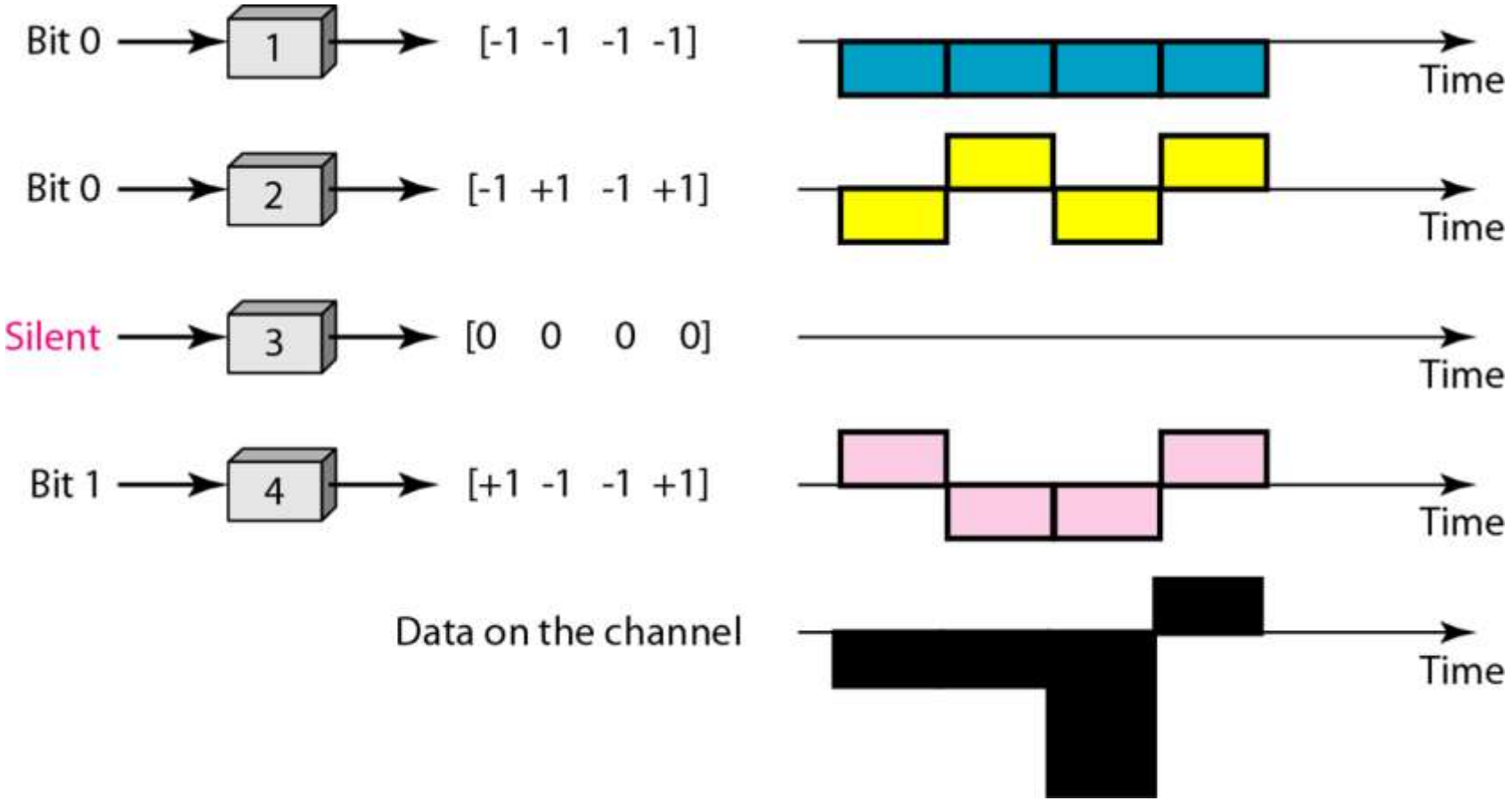


CDMA Encoding



SOMAIYA VIDYAVIHAR UNIVERSITY

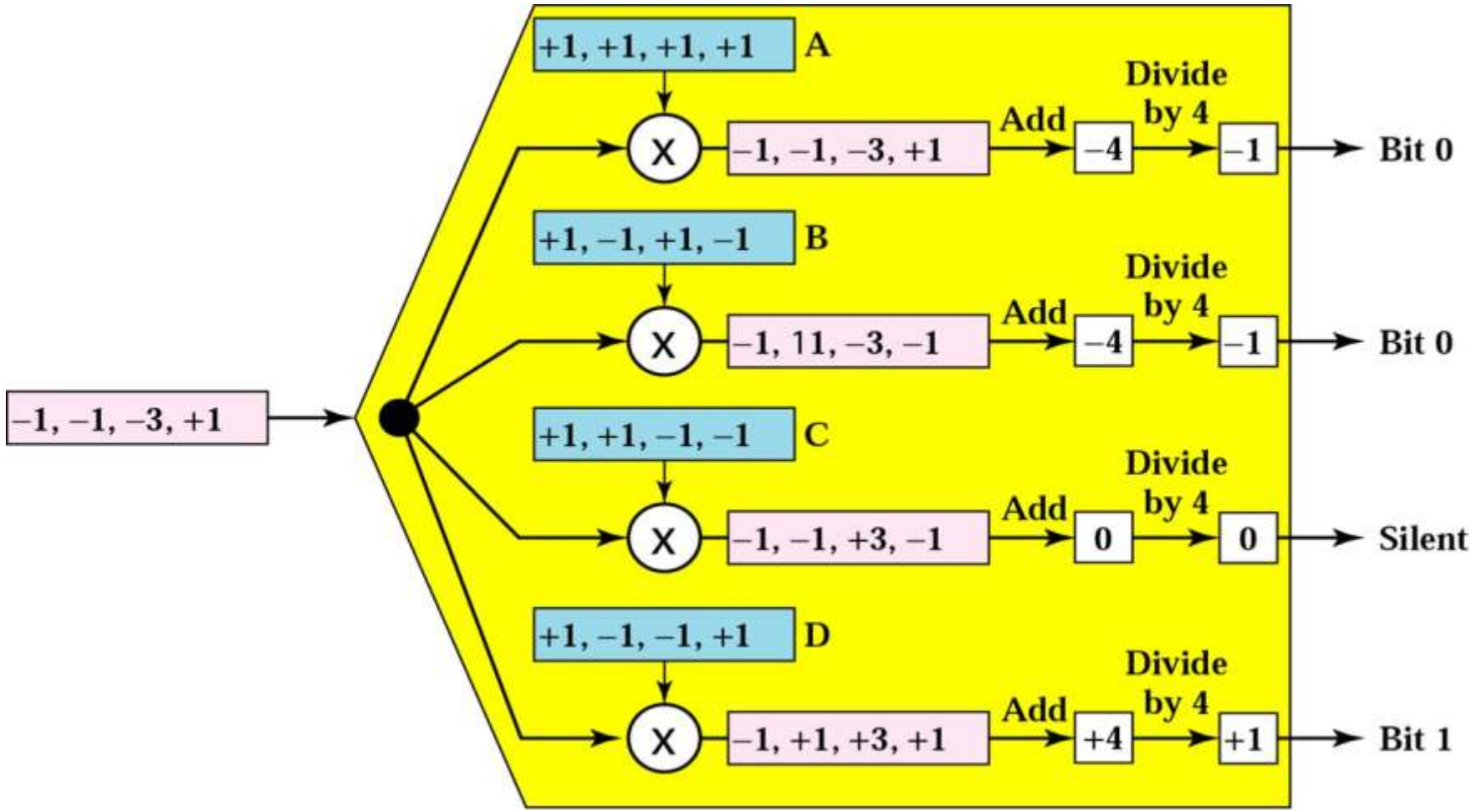
Signal Created by CDMA



Somaiya TRUST

SOMAIYA VIDYAVIHAR UNIVERSITY

CDMA Decoding



Somaiya TRUST

Sequence Generation

- Common method: *Walsh Table*
  - Number of sequences is always a power of two

$$W_1 = [+1] \qquad W_{2N} = \begin{bmatrix} W_N & W_N \\ W_N & \overline{W_N} \end{bmatrix}$$

a. Two basic rules

$$W_1 = [+1]$$

$$W_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}$$

$$W_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$$

b. Generation of W_1, W_2, and W_4

Example: Walsh Table

1. Find chip sequences for eight stations

2. There are 80 stations in a CDMA network. What is the length of the sequences generated by Walsh Table?

Chapter 13

Wired LANs: Ethernet

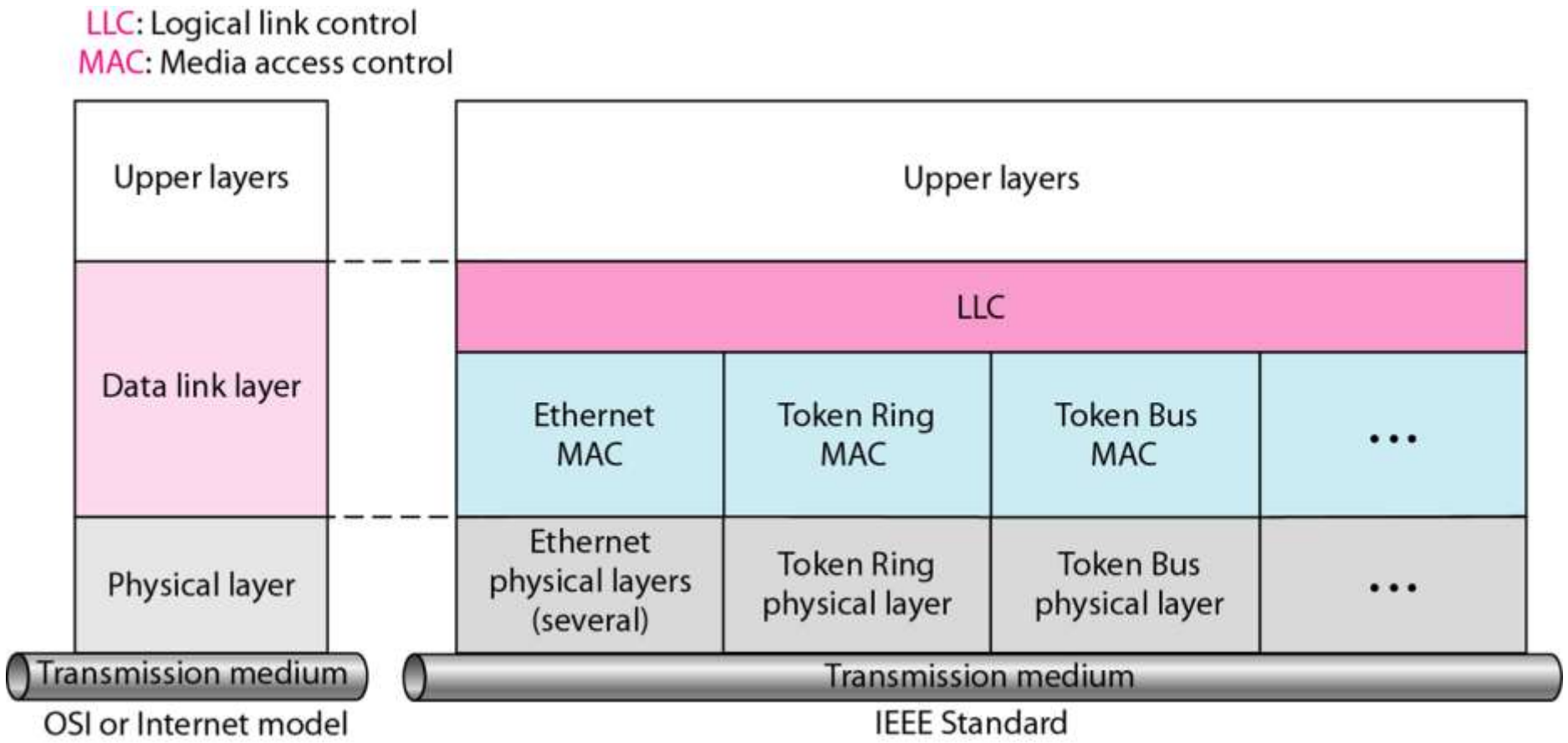
IEEE STANDARDS

In 1985, the Computer Society of the IEEE started a project, called Project 802, to set standards to enable intercommunication among equipment from a variety of manufacturers. Project 802 is a way of specifying functions of the physical layer and the data link layer of major LAN protocols.

Topics discussed in this section:

Data Link Layer
Physical Layer

Figure 13.1 *IEEE standard for LANs*



13-2 STANDARD ETHERNET

The original Ethernet was created in 1976 at Xerox's Palo Alto Research Center (PARC). Since then, it has gone through four generations. We briefly discuss the Standard (or traditional) Ethernet in this section.

Topics discussed in this section:

MAC Sublayer
Physical Layer

Figure 13.3 *Ethernet evolution through four generations*

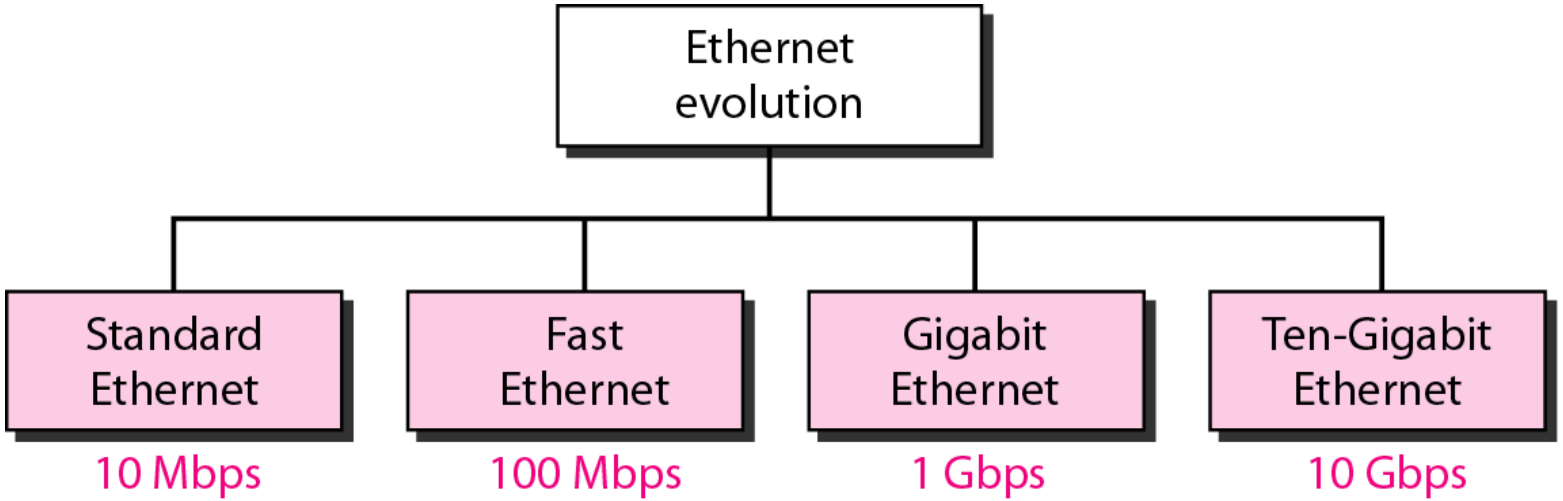
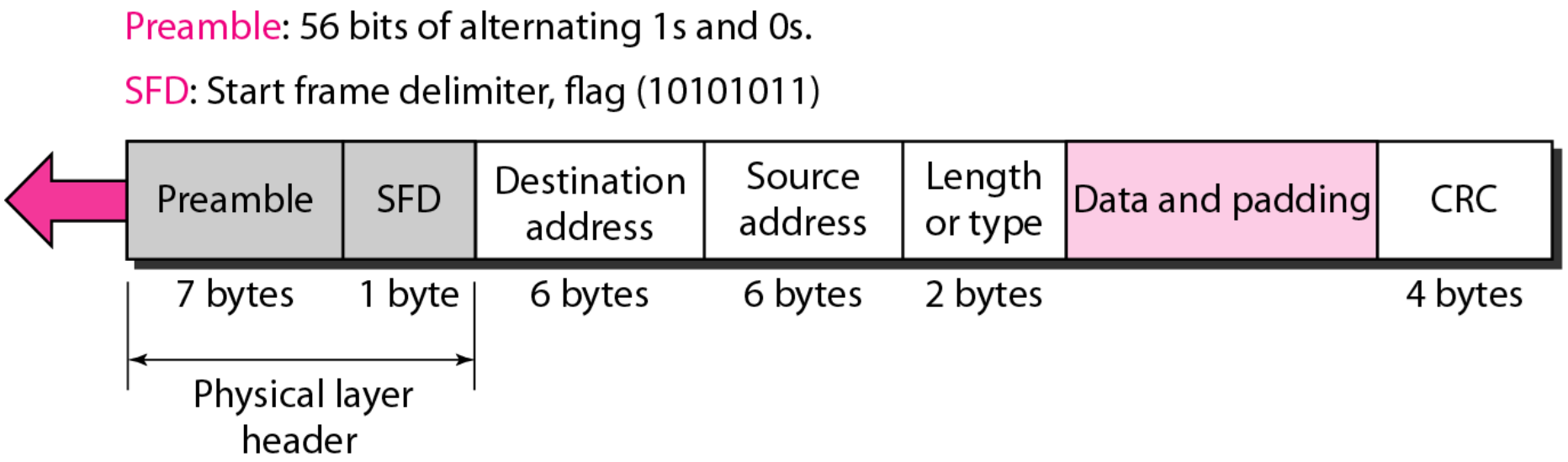


Figure 13.4 *802.3 MAC frame*



802.3 MAC frame

- The *Preamble*
  - consists of seven bytes all of the form 10101010,
  - used by the receiver to allow it to establish bit synchronization.
- The *Start frame delimiter*
  - is a single byte, 10101011, which is a frame flag, indicating the start of a frame.
  - It also signifies that the Destination MAC Address field begins with the next byte.
- Destination MAC Address.
- Source MAC Address.

802.3 MAC frame

- Length or Type
  - defines the type of protocol inside the frame, for example IPv4 or IPv6,ARP,OSPF etc.
  - it indicates the number of bytes of data in the frame's payload, and can be anything from 0 to 1500 bytes.
  - Frames must be at least 64 bytes long, not including the preamble, so, if the data field is shorter than 46 bytes, it must be compensated by the *Padding* field.

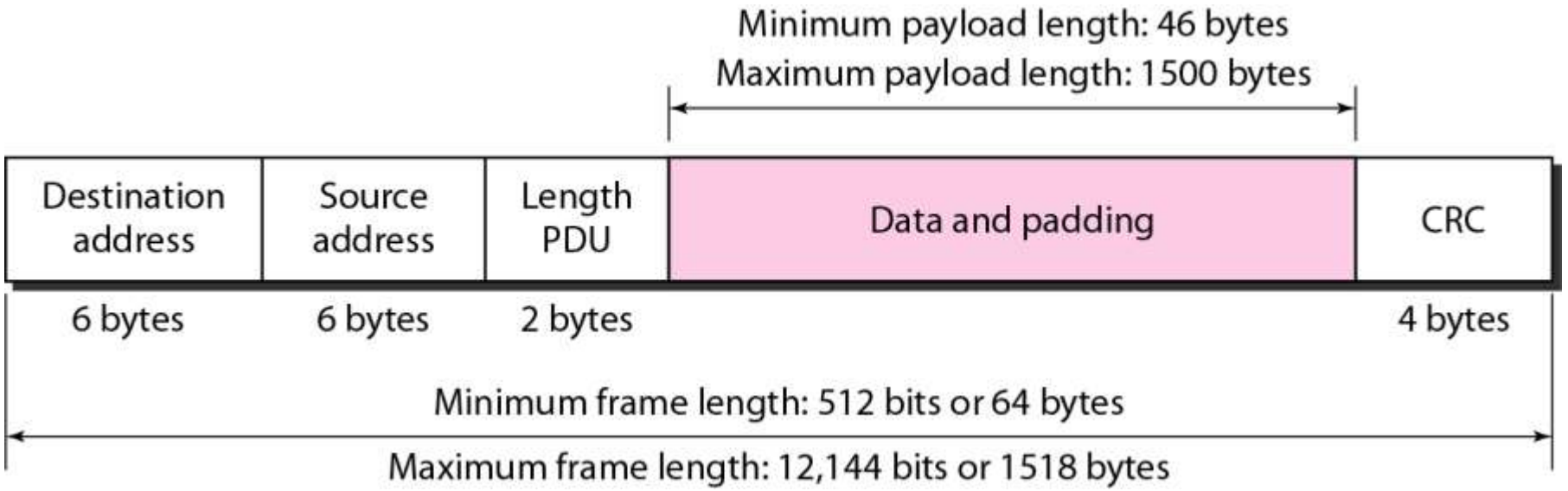
802.3 MAC frame

- Data
  - Field carries data encapsulated from upper-layer protools.
  - it is minimum of 46 bytes and maximum of 1500 bytes.
  - If data is more than 1500 bytes, then it should be fragmented and encapsulated in more than one frames.
  - If less than 46 bytes, it need to be padded with extra 0’s

802.3 MAC frame

- CRC
  - Contains error detection information.
  - CRC is calculated over addresses, types and data field.
  - Receiver calculates CRC and finds that it is not zero, it discards the frame.

Figure 13.5 *Minimum and maximum lengths*



Note

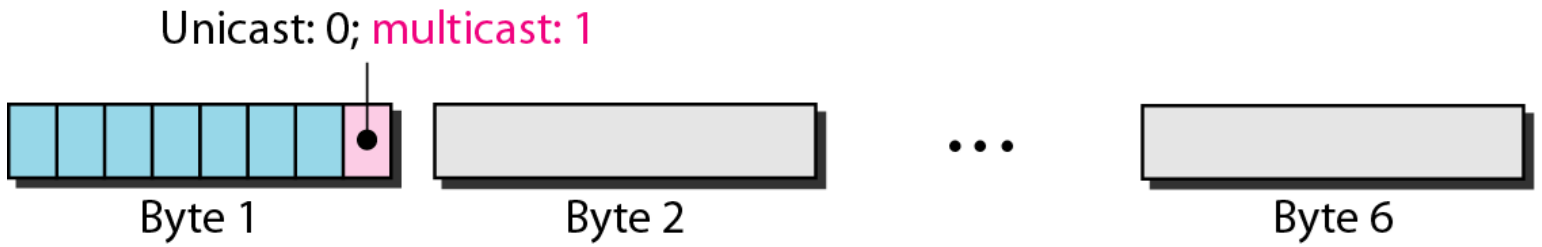
Frame length:

Minimum: 64 bytes (512 bits)

Maximum: 1518 bytes (12,144 bits)

Figure 13.6 *Example of an Ethernet address in hexadecimal notation*

Figure 13.7 *Unicast and multicast addresses*



Note

The least significant bit of the first byte defines the type of address. If the bit is 0, the address is unicast; otherwise, it is multicast.

Note

The broadcast destination address is a special case of the multicast address in which all bits are 1s.

Example 13.1

Define the type of the following destination addresses:

a. 4A:30:10:21:10:1A *b. 47:20:1B:2E:08:EE*

c. FF:FF:FF:FF:FF:FF

Solution

To find the type of the address, we need to look at the second hexadecimal digit from the left. If it is even, the address is unicast. If it is odd, the address is multicast. If all digits are F's, the address is broadcast. Therefore, we have the following:

a. This is a unicast address because A in binary is 1010.

b. This is a multicast address because 7 in binary is 0111.

c. This is a broadcast address because all digits are F's.

Example 13.2

Show how the address 47:20:1B:2E:08:EE is sent out on line.

Solution

The address is sent left-to-right, byte by byte; for each byte, it is sent right-to-left, bit by bit, as shown below:

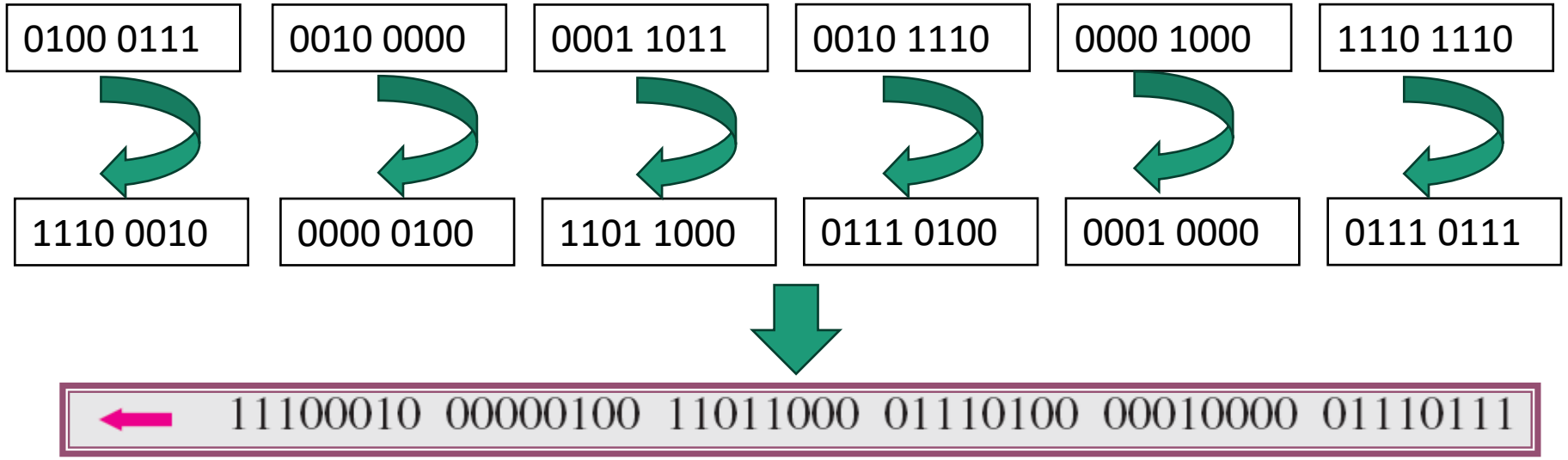


Figure 13.8 *Categories of Standard Ethernet*

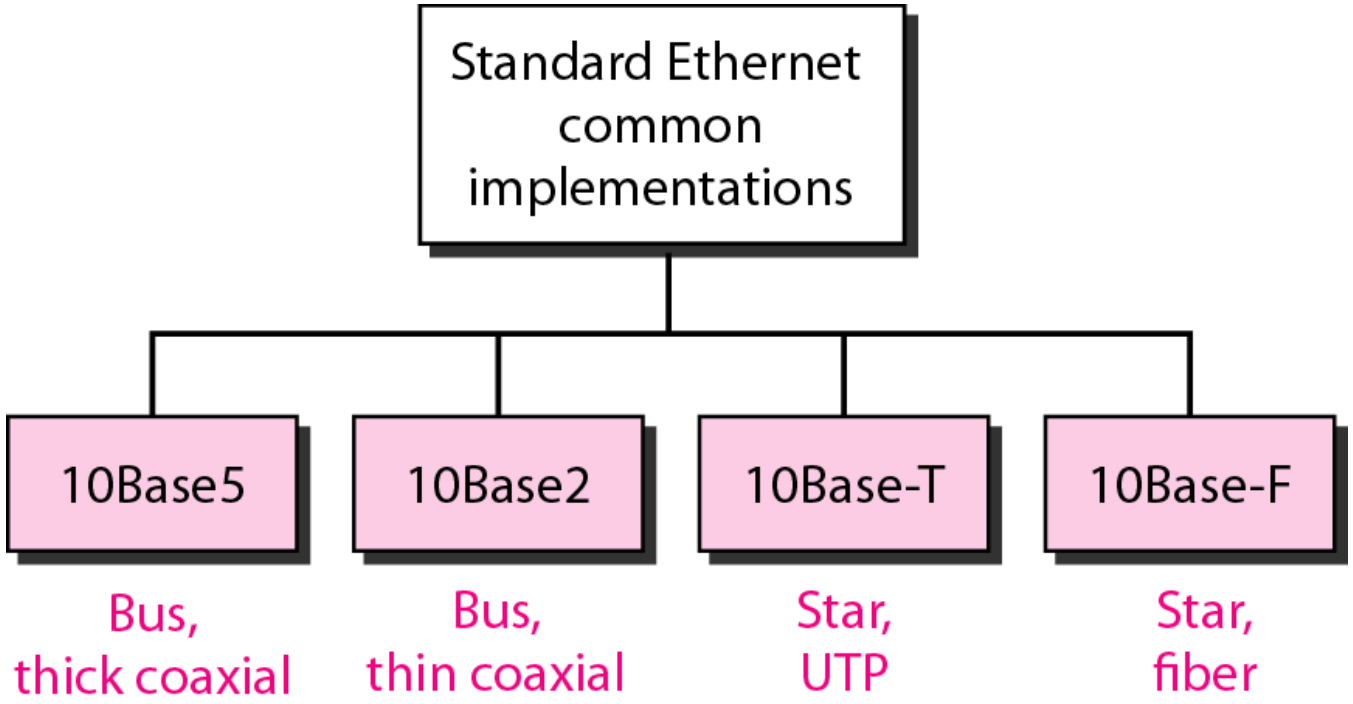


Table 13.4 *Summary of Ten-Gigabit Ethernet implementations*

Characteristics	*10GBase-S*	*10GBase-L*	*10GBase-E*
Media	Short-wave 850-nm multimode	Long-wave 1310-nm single mode	Extended 1550-mm single mode
Maximum length	300 m	10 km	40 km