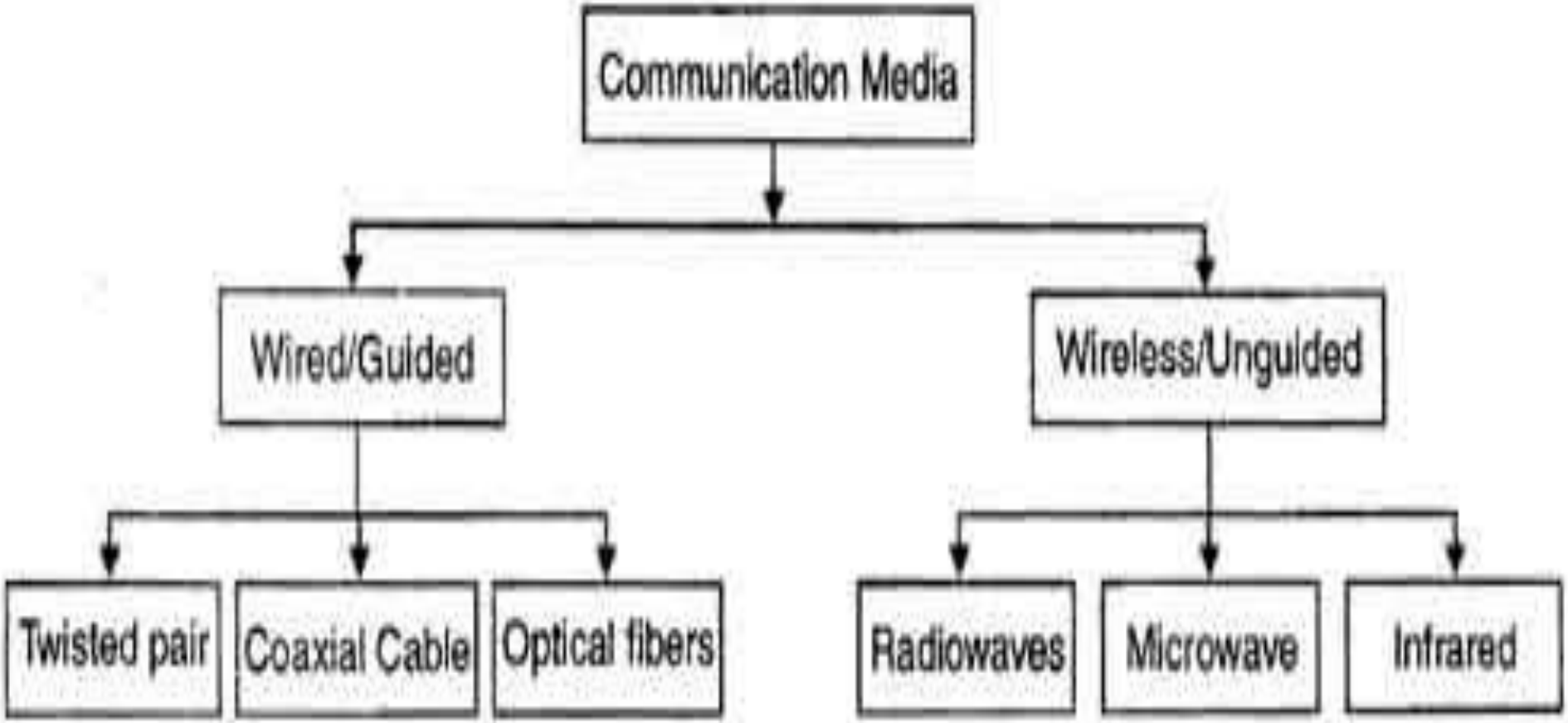
2. Physical and Data Link Layer

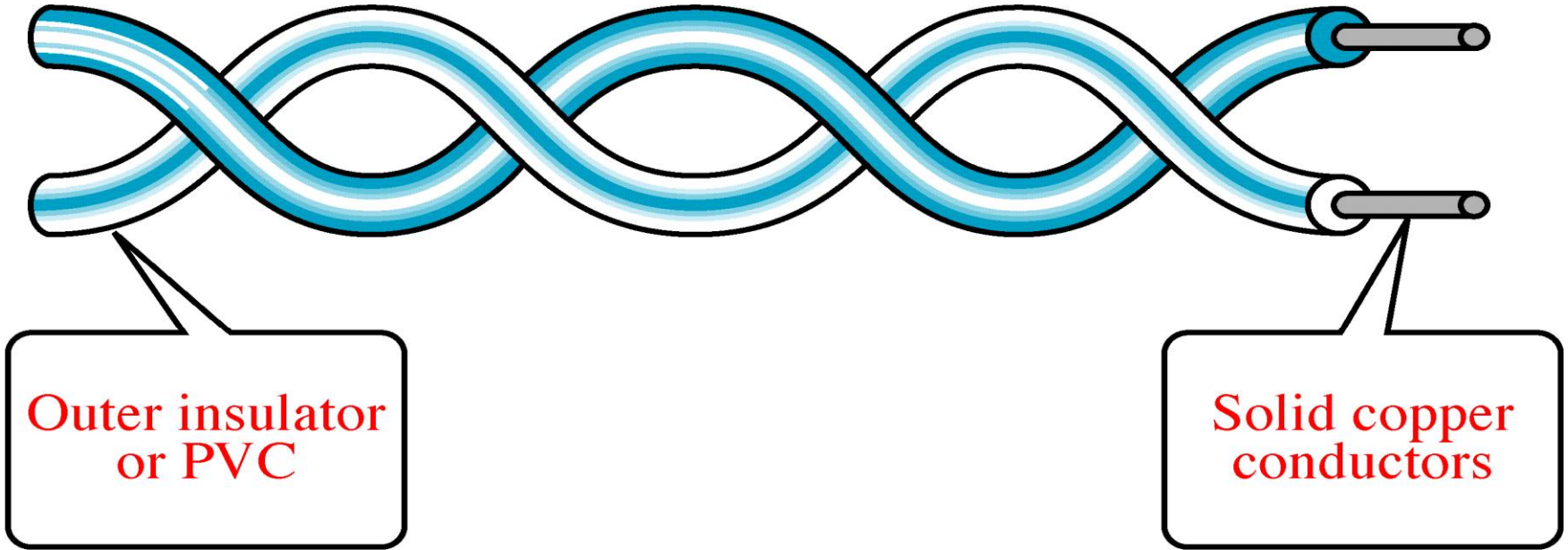
Physical layer

Transmission Media



Transmission media

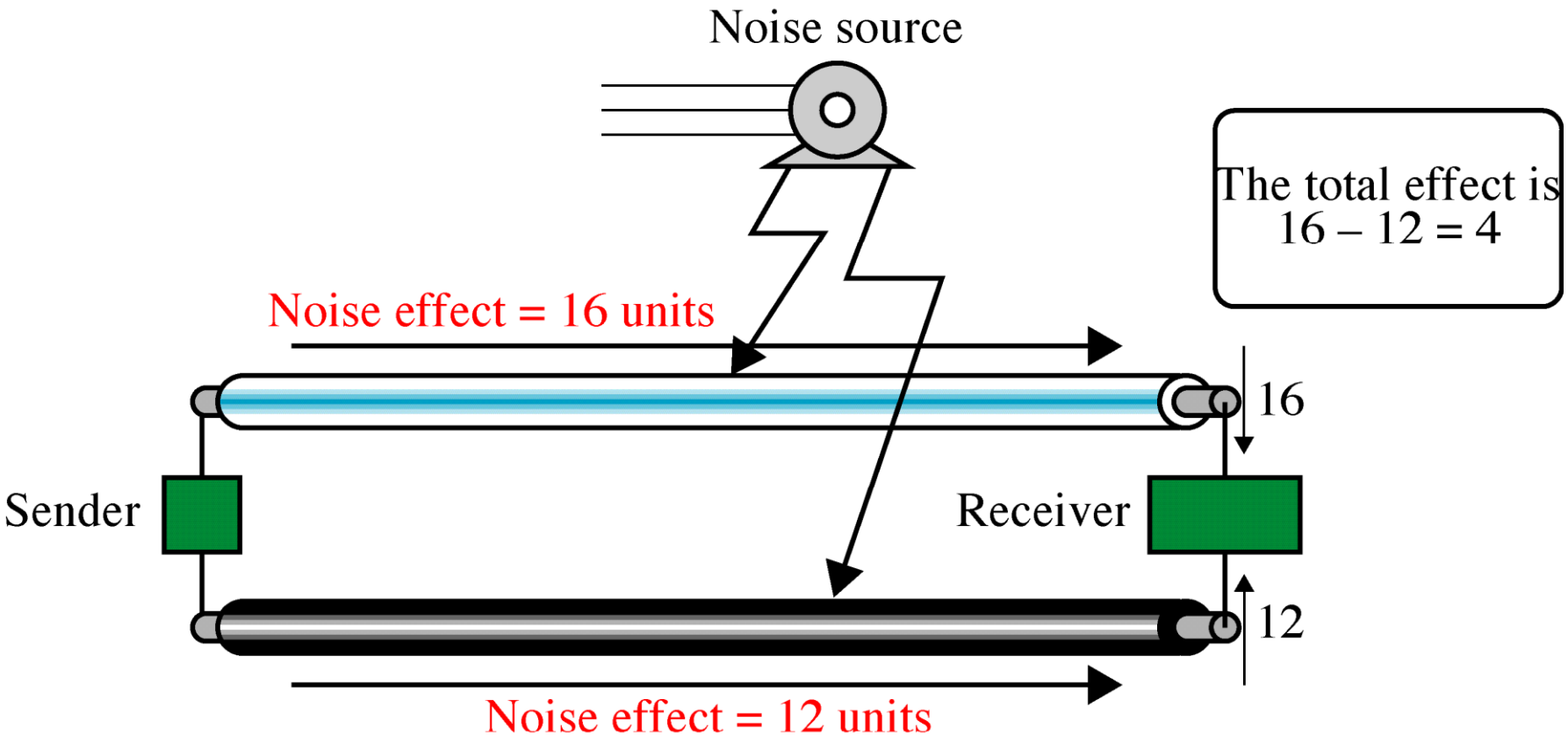
- Guided media
  - Guided media provide a connection from one device to another device.
  - It include twisted-pair cable, coaxial cable and fiber-optic cable.
- Unguided media
  - Unguided media transport electromagnetic waves without using physical conductor.
- Twisted – pair cable
  - A twisted pair consist of two insulated copper wires, about 1mm thick.
  - The wires are twisted together in helical form, just like your DNA molecule.



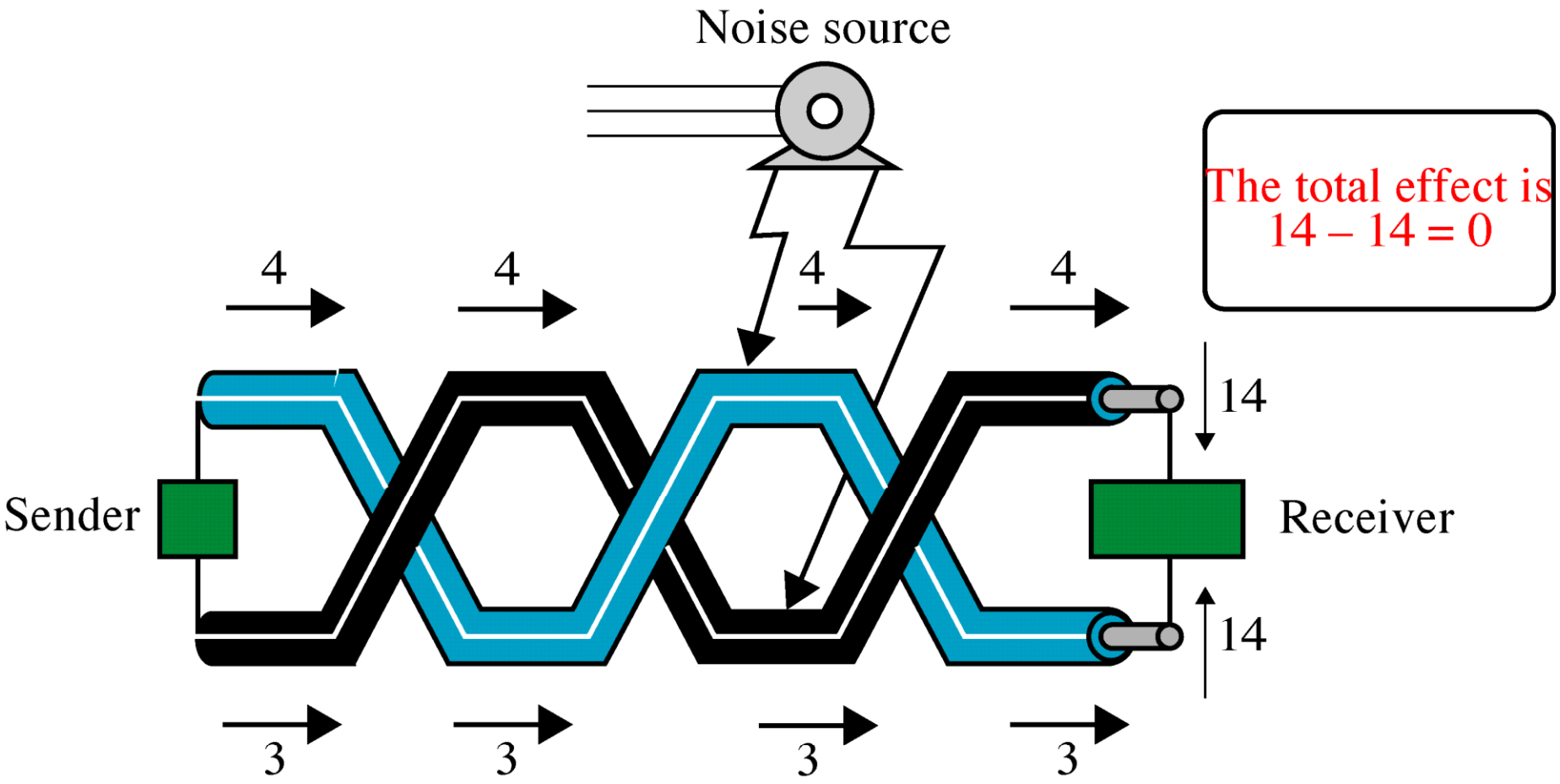
Transmission media

- One wire is used to carry signals to the receiver.
- Other is used only as a ground reference. Receiver uses difference between two wire.
- Interference may affect both wires and create unwanted signals.
- E.g. in straight cable noise on both wire are different so it may create difference on receiver.
- By twisting a pair balance is maintained. (in one twist one wire is closer to noise source and other is farther, but in next twist the reverse is true.)

Transmission Media



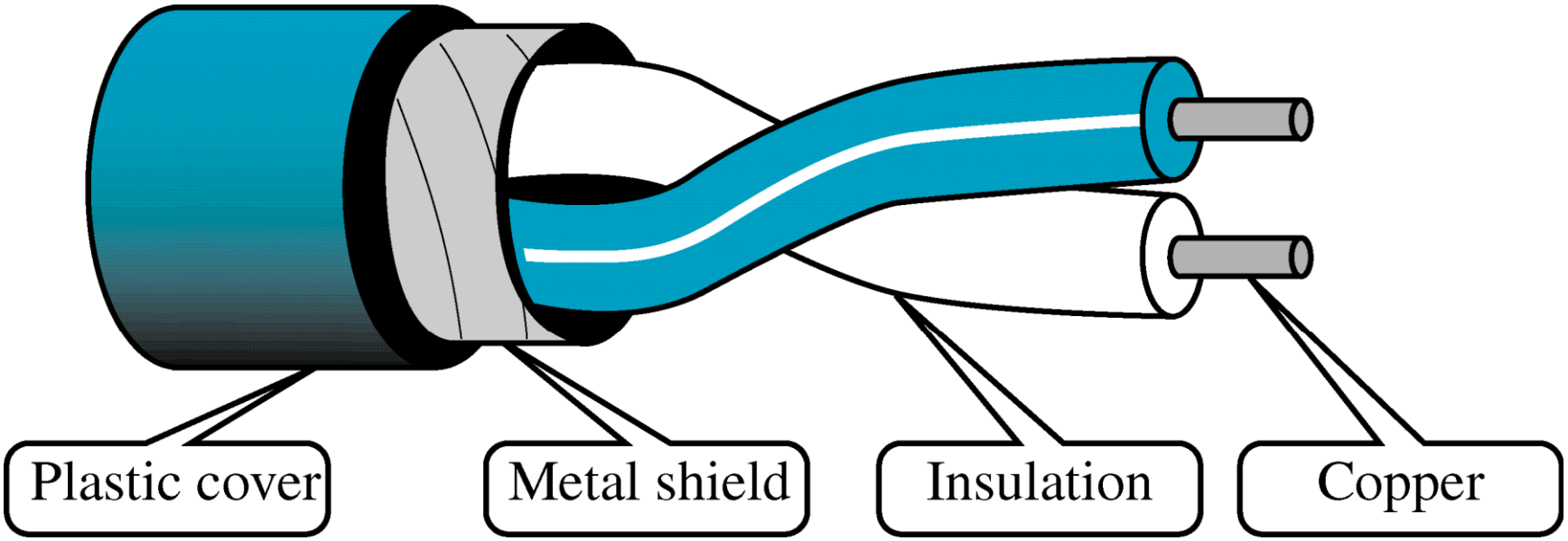
Transmission Media



Transmission media

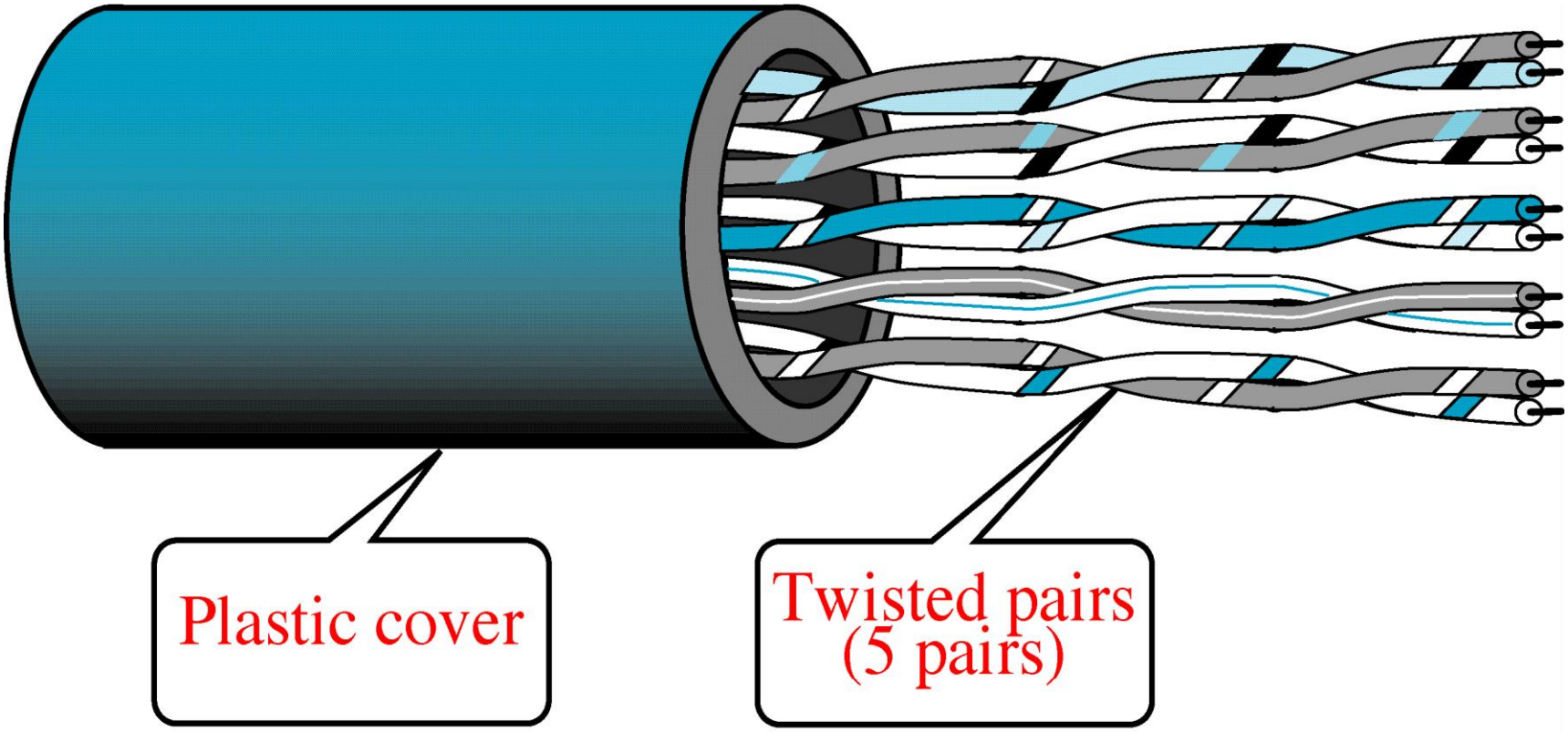
- Types of twisted pair
  - Shielded twisted pair cable
  - Unshielded twisted pair cable

- **Shielded twisted pair cable**



Transmission media

- **Unshielded twisted pair cable**



Transmission media

- **Categories of unshielded twisted pair cable**
  - Total seven categories of unshielded twisted pair cable are available

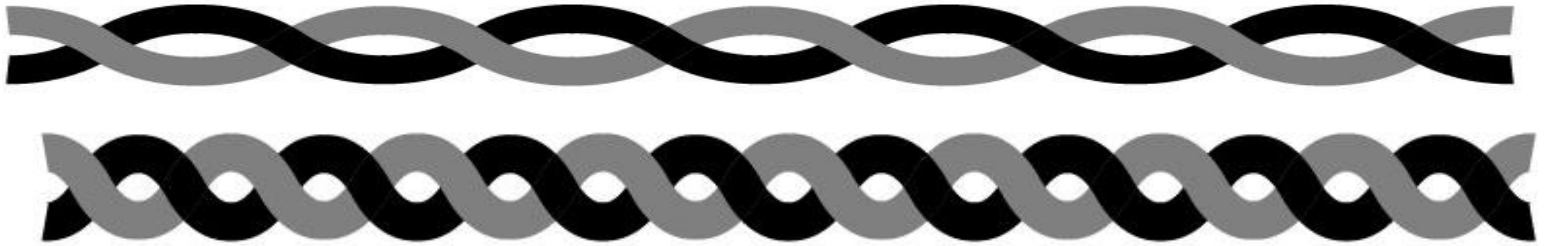
(a)

(b)

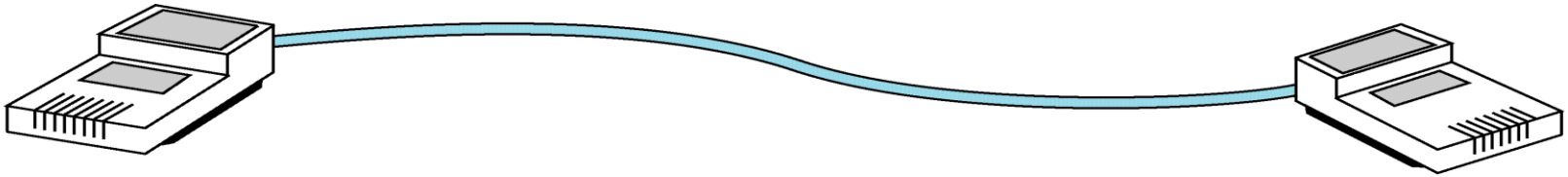
  - (a) category-3 UTP
  - (b) category-5 UTP

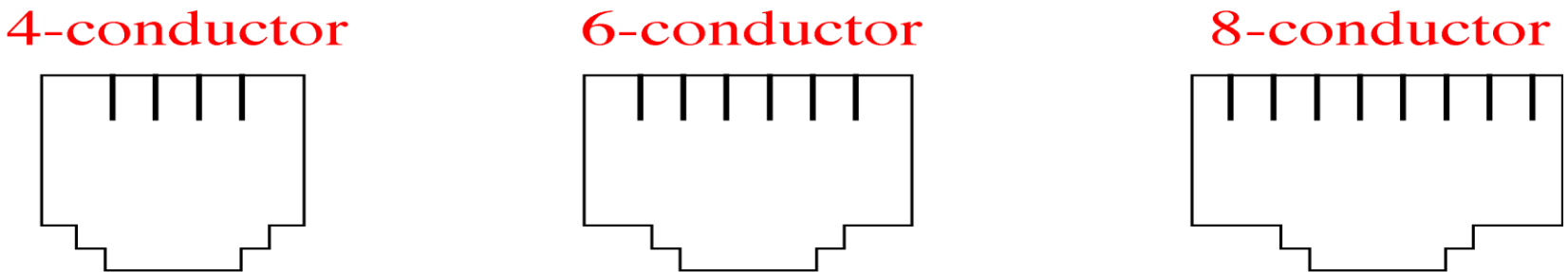
Transmission media

- **Categories of unshielded twisted pair cable**
  - Total seven categories of unshielded twisted pair cable are available in which 1 as lowest and 7 as highest.



  - (a) category-3 UTP
  - (b) category-5 UTP
- **Connectors**
  - Most common UTP connector is RJ45.



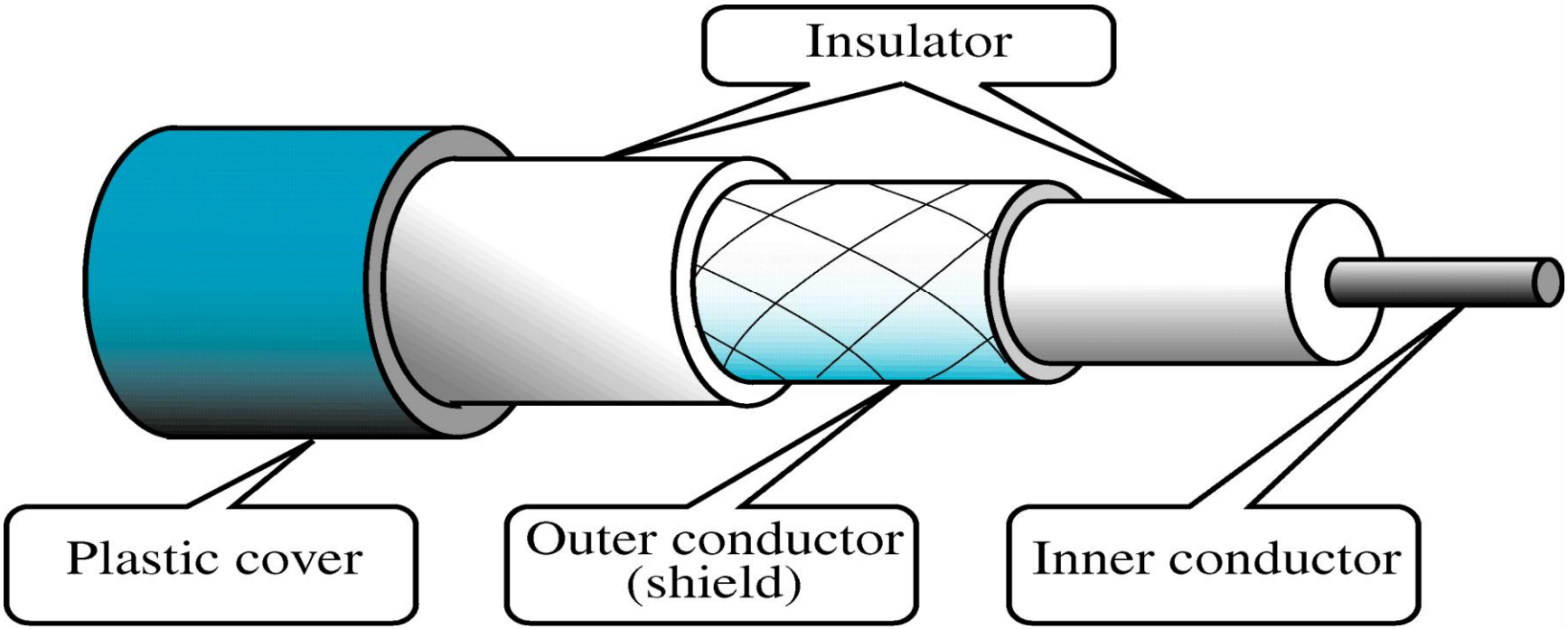


Transmission media

- **Performance**
  - Twisted pair can pass vide range of frequency.
  - Gauge is a measure of the thickness of the cable.
  - Thick wire provide more bandwidth.
- **Application**
  - Twisted pair cables can be used in telephone line.
  - Local Area Network
    - 10BaseT and 100BaseT twisted pair cables are used for LAN connection.

Transmission media

- **Coaxial Cable**
  - Coaxial cable carries signals of higher frequency ranges than those in twisted pair cable.
  - Instead of having two wires, coax has a central core conductor of solid copper.



Transmission media

  - The copper core is enclosed in an insulating sheath.
  - Insulating sheath is covered in an outer conductor of metal foil, braid or combination of two.
  - Outer conductor is enclosed in insulating sheath.
    - Outer conductor gives protection against noise and protection to the second conductor which completes the circuit.
  - And the whole cable is protected by plastic cover.
- Categories of coaxial cable
  - RG-59 – cable TV
  - RG-58 – Thin Ethernet
  - RG-11 – Thick Ethernet
- Coaxial cable connector
  - BNC connector (Bayon-neill connector)
    - Used to connect end of the cable
  - BNCT connector
    - Used in Ethernet networks
  - BNC Terminator
    - Used at the end of the cable to prevent the reflection of the signal.

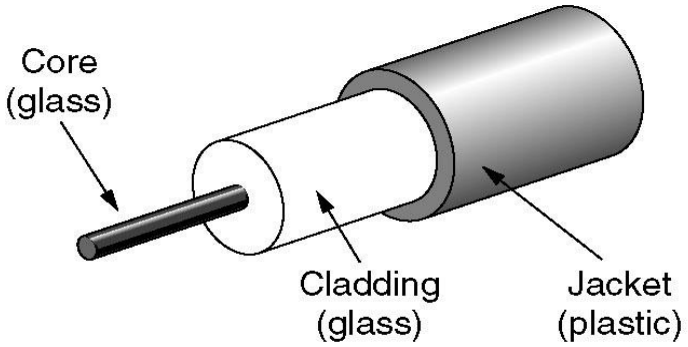
Transmission media

Transmission media

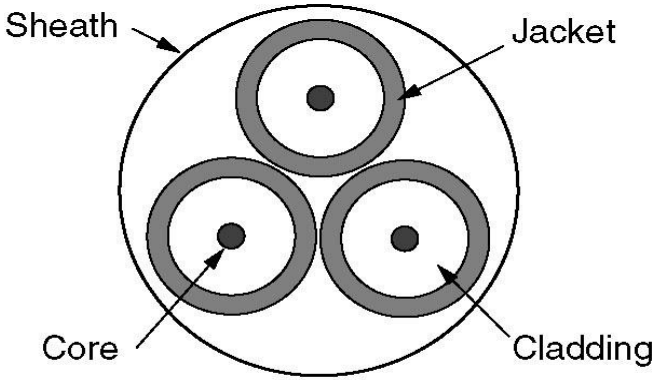
- Performance
  - Coaxial cable has a much higher bandwidth than the twisted pair.
- Application
  - Telephone lines
  - Cable TV networks
  - Traditional Ethernet network

Transmission media

- **Fiber optic cable**
  - A fiber optic cable is made up of glass or plastic and transmit signals in form of light.



(a)

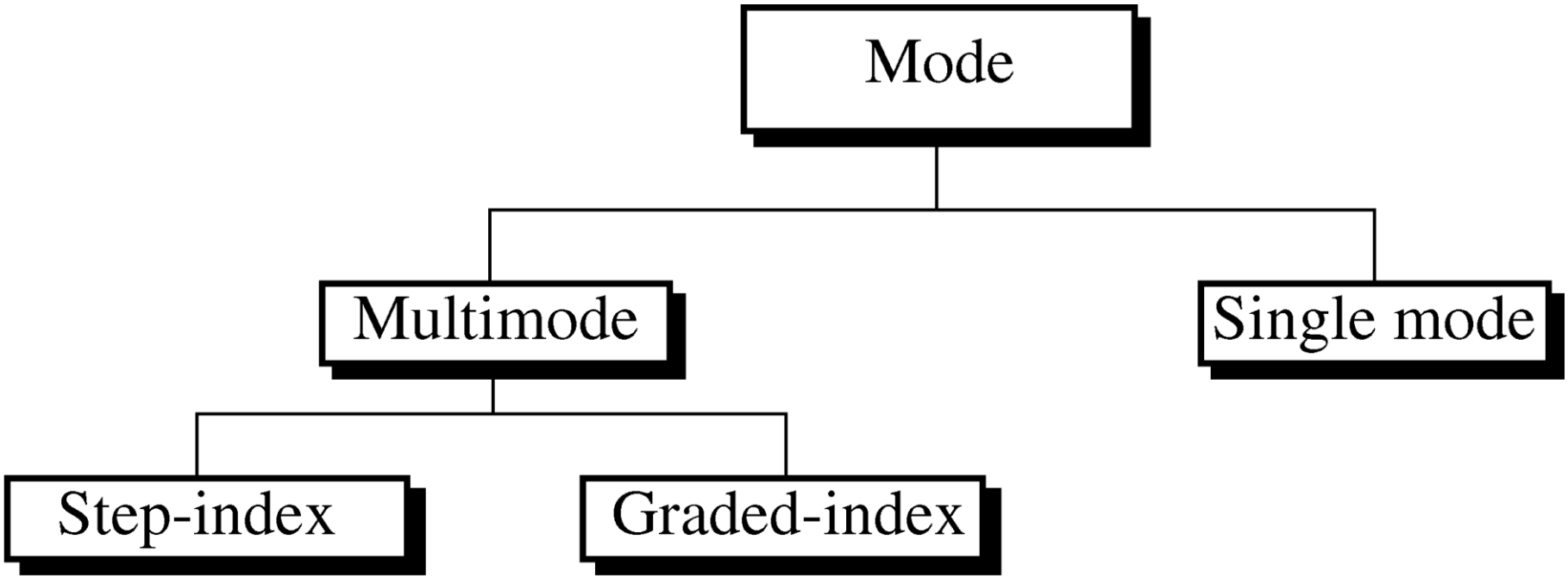


(b)

  - (a) side view of single fiber
  - (b) end view of a sheath with three fiber
- Light travel in straight line as long as it is moving from single uniform substance.
- If light move from one substance to another substance the ray changes the direction.

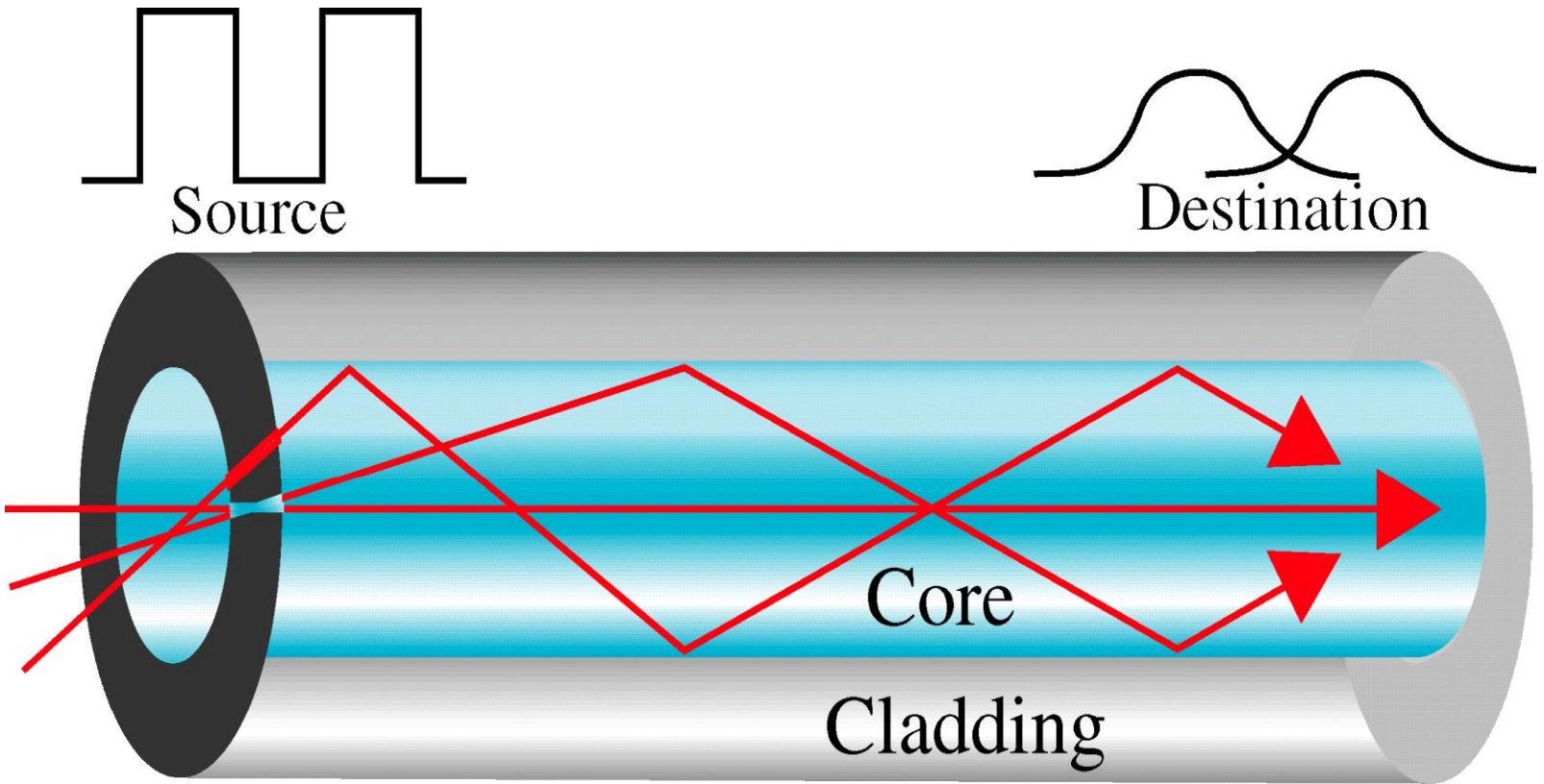
Transmission media

- **Propagation mode**



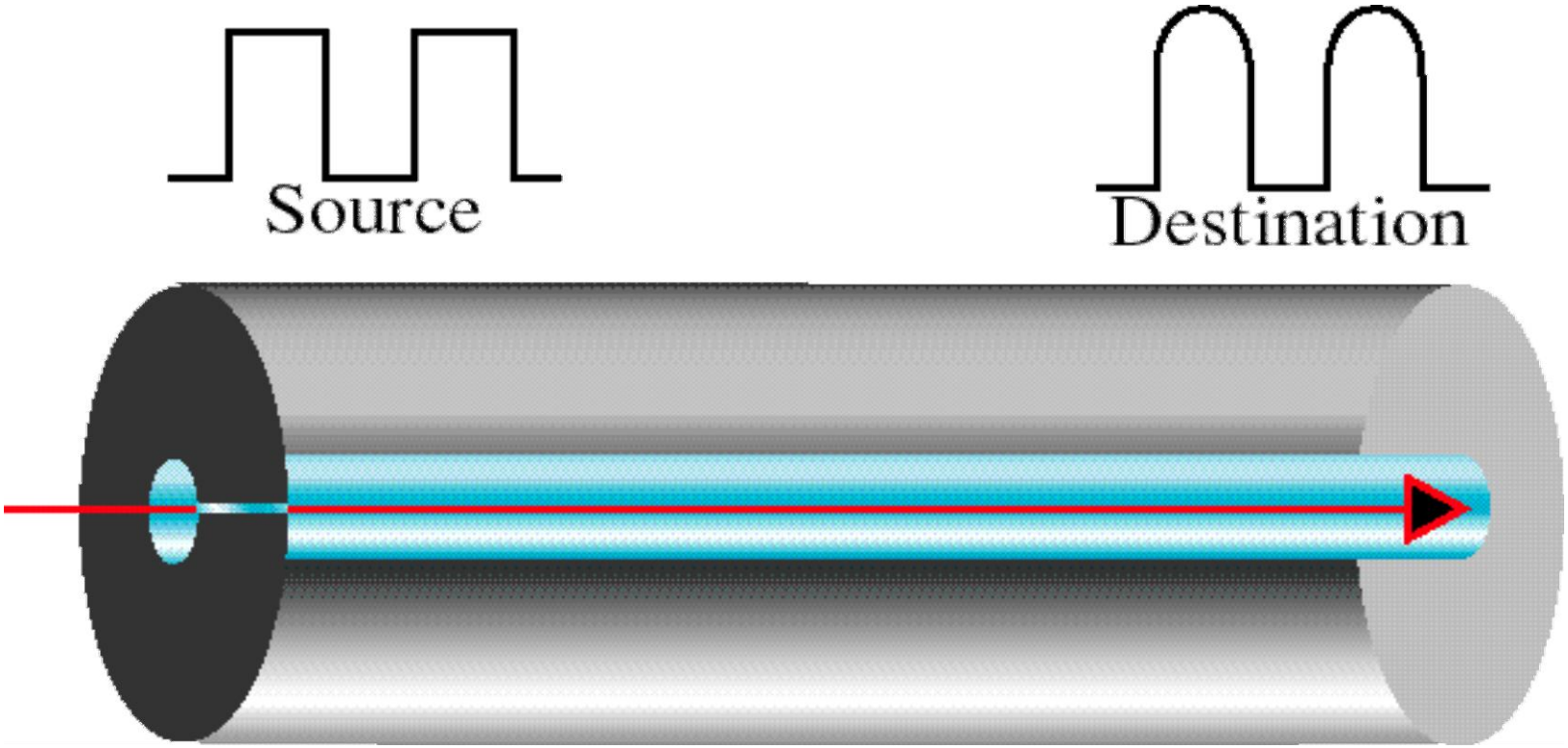
Transmission media

- Multimode
  - Multiple beams from a light source move through the core in different path.
- Single mode
  - Single beam from a light source move through core.
- Propagation of beams almost horizontal.
  - **Multimode step-index**



Transmission media

- **Single mode**



Transmission media

- SC connector (Subscribe channel connector)
  - Used for cable TV
- ST connector (Straight tip connector)
  - Connecting cable in networking
- MT-RJ connector same size as RJ45.

Transmission media

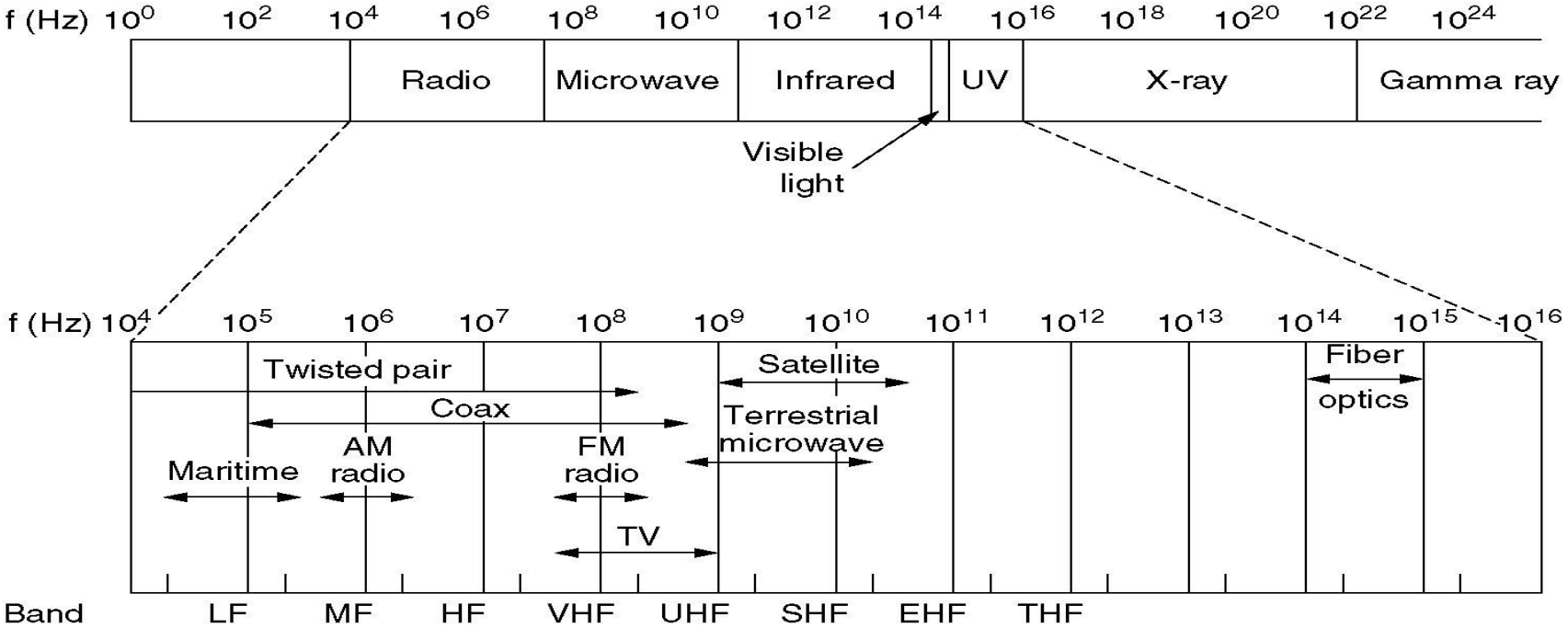
- Applications
  - Often found in backbone network its wide bandwidth is cost effective
  - Some cable TV companies uses combination of fiber & coaxial cable.
  - Fast Ethernet uses 100baseFX and 1000baseFX fiber optic cable.
- Advantage of fiber optic
  - High bandwidth
    - Fiber optic provide higher bandwidth than twisted pair & coaxial cable.
  - Less Signal attenuation
    - Transmission distance is grater than that of other guided media
    - In twisted pair repeaters are required at every 5KM.
    - While in fiber optic repeaters are required at every 50KM.
  - Immunity to electromagnetic interference
    - Electromagnetic noise cannot affect fiber optic cable
  - Resistance to corrosive material
    - Resistance power is more than copper cable
  - Light weight
    - Much lighter than copper

Transmission media

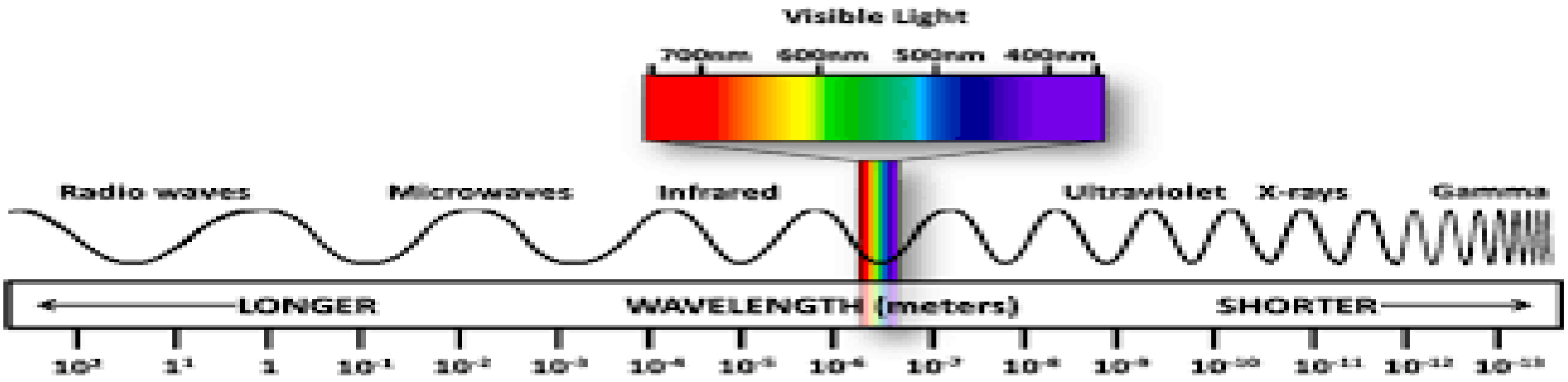
- Grater immunity to tapping
  - Intruder can not tape fiber optic cable
- Disadvantage
  - Installation and maintenance
    - New technology so that expert engineer is required
  - Unidirectional light propagation
    - If we need bidirectional line than two fibers are needed
  - Cost
    - Cable and interfaces are relatively more expencive

Transmission media

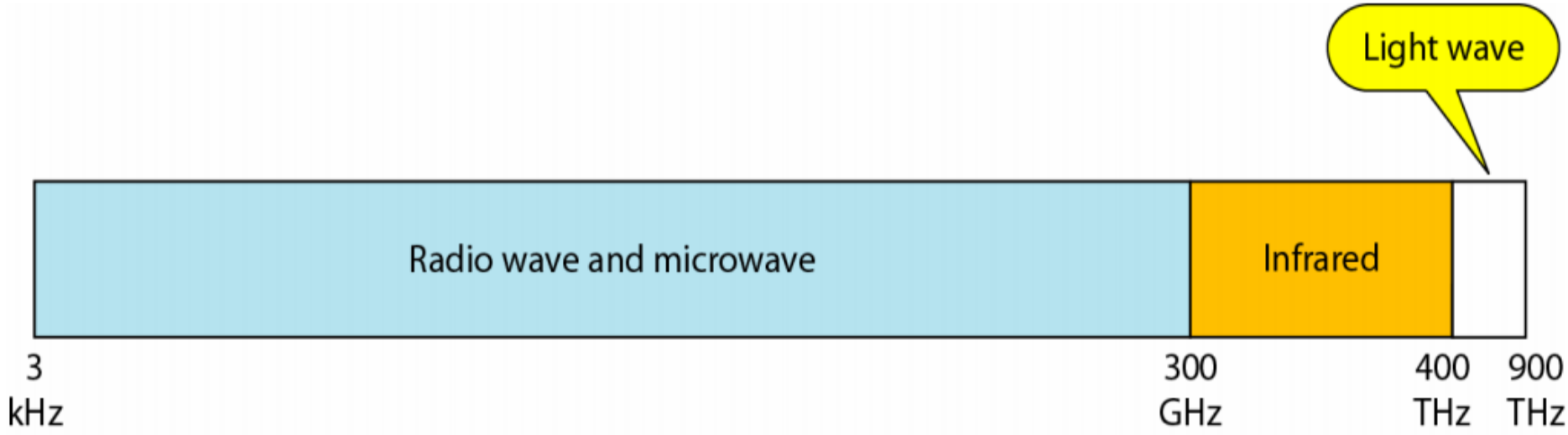
- Unguided media: Wireless
  - Unguided media transport electromagnetic waves without using physical conductor.
  - Often called wireless communication.
  - Signals are normally broadcast through free space.
  - When electrons (signal) moves, they create electromagnetic waves.
  - " Number of cycle per second of electron wave is called its frequency and it measured in HZ".
  - Electromagnetic spectrum



Transmission media

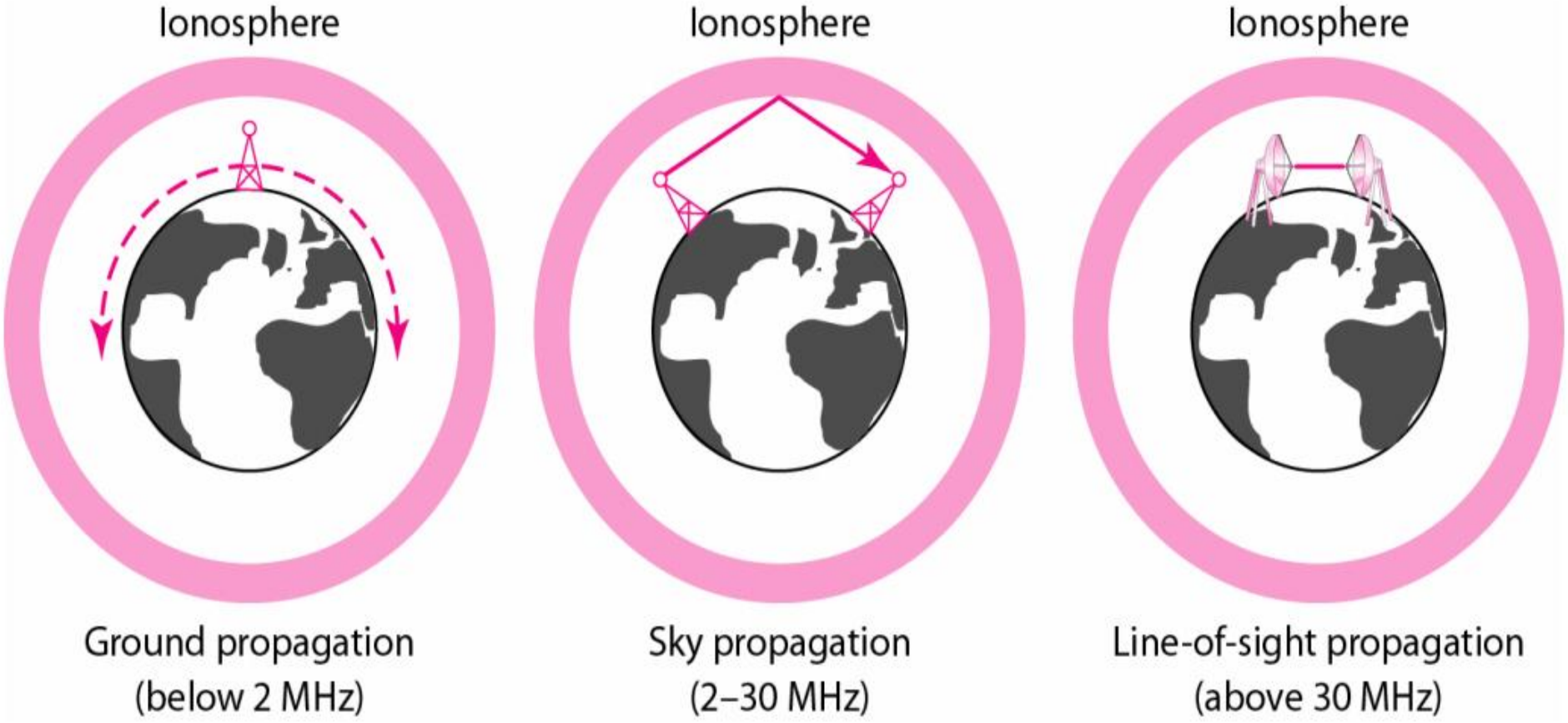


- Ranging from 3KHz to 900THz used for wireless communication.



Transmission media

- Propagation Type



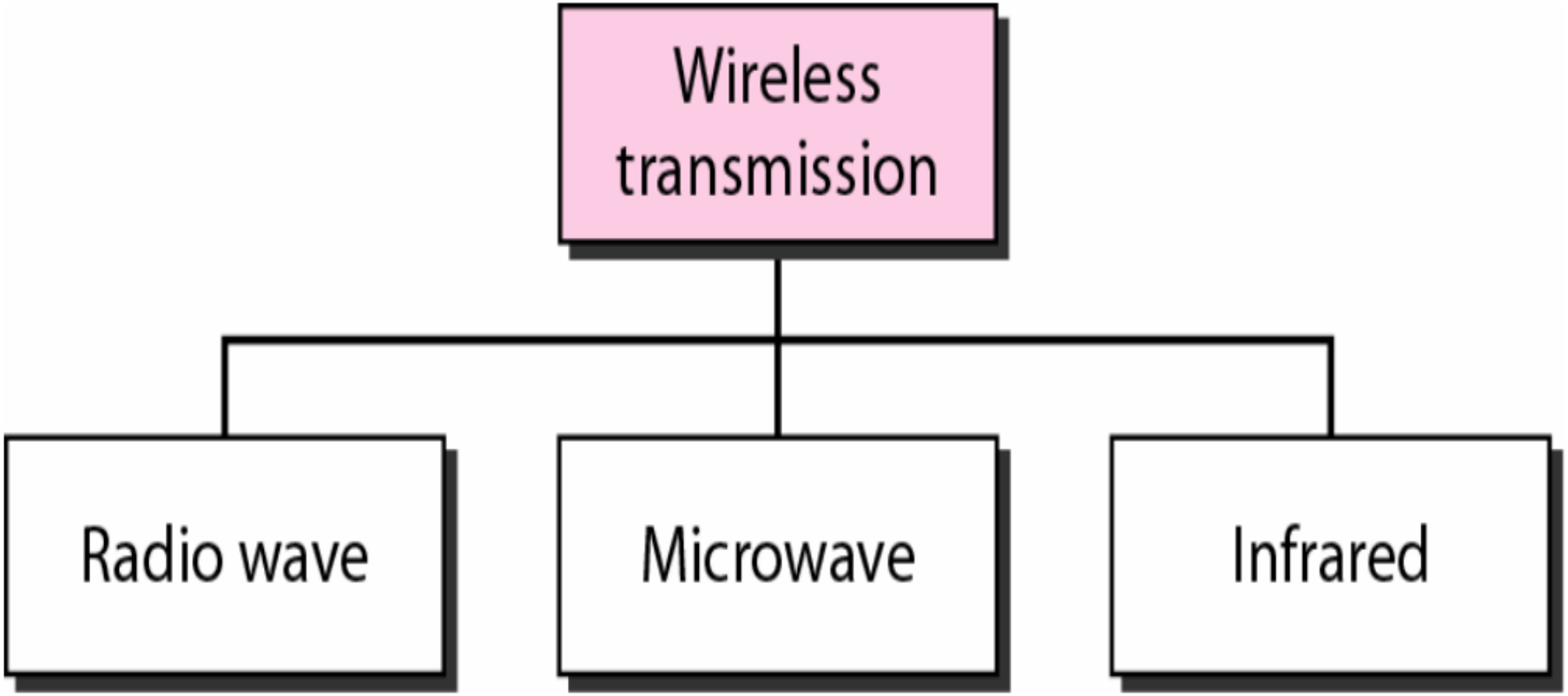
Transmission media

- Ground propagation
  - Radio waves travels through very lowest portion of the atmosphere.
- Sky Propagation
  - Higher frequency radio waves radiate upward into the ionosphere (waves reflects back)
- Line of sight propagation
  - Very high frequency signals are transmitted in straight lines directly from antenna to antenna.
- **Range of frequency is known as band.**

Transmission media

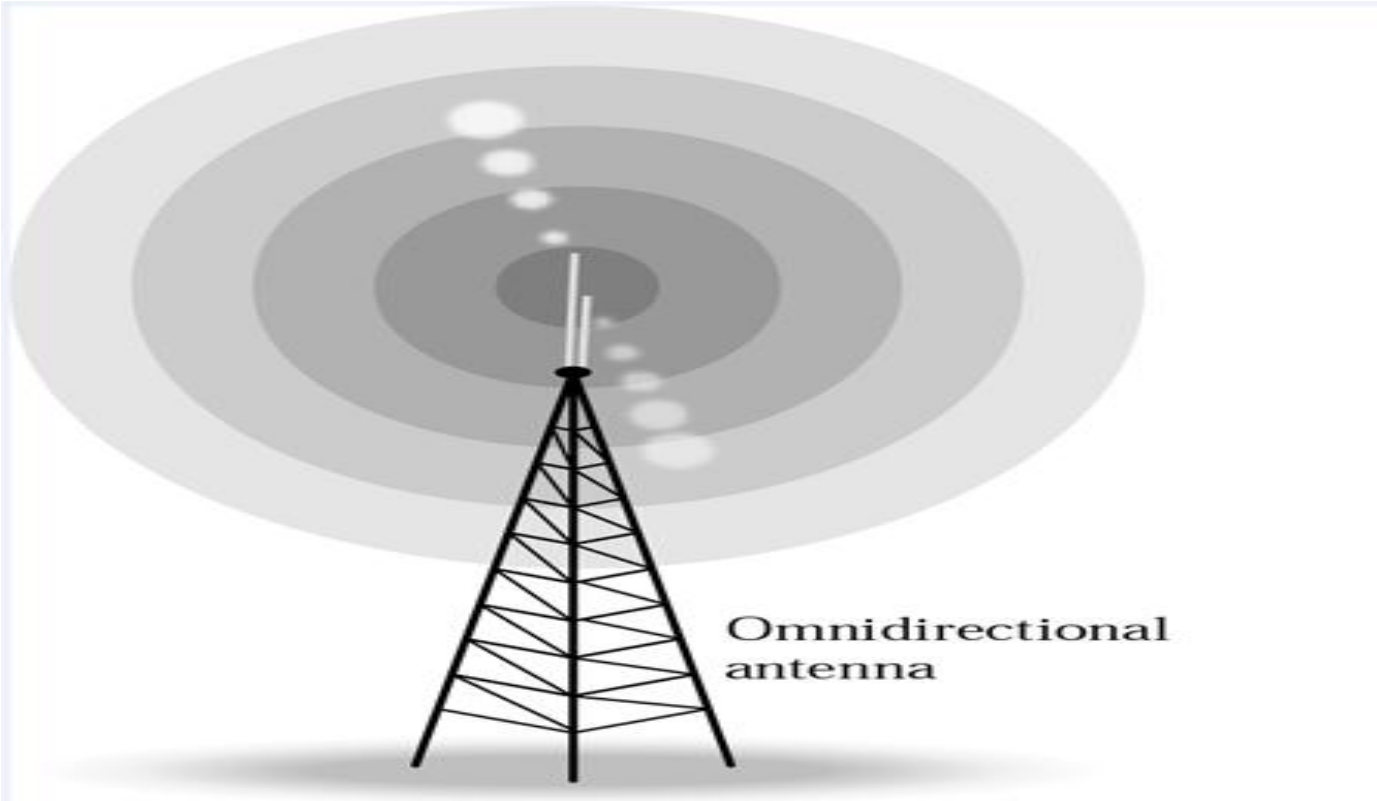
Band	*Range*	*Propagation*	*Application*
VLF (very low frequency)	3–30 kHz	Ground	Long-range radio navigation
LF (low frequency)	30–300 kHz	Ground	Radio beacons and navigational locators
MF (middle frequency)	300 kHz–3 MHz	Sky	AM radio
HF (high frequency)	3–30 MHz	Sky	Citizens band (CB), ship/aircraft communication
VHF (very high frequency)	30–300 MHz	Sky and line-of-sight	VHF TV, FM radio
UHF (ultrahigh frequency)	300 MHz–3 GHz	Line-of-sight	UHF TV, cellular phones, paging, satellite
SHF (superhigh frequency)	3–30 GHz	Line-of-sight	Satellite communication
EHF (extremely high frequency)	30–300 GHz	Line-of-sight	Radar, satellite

Transmission media



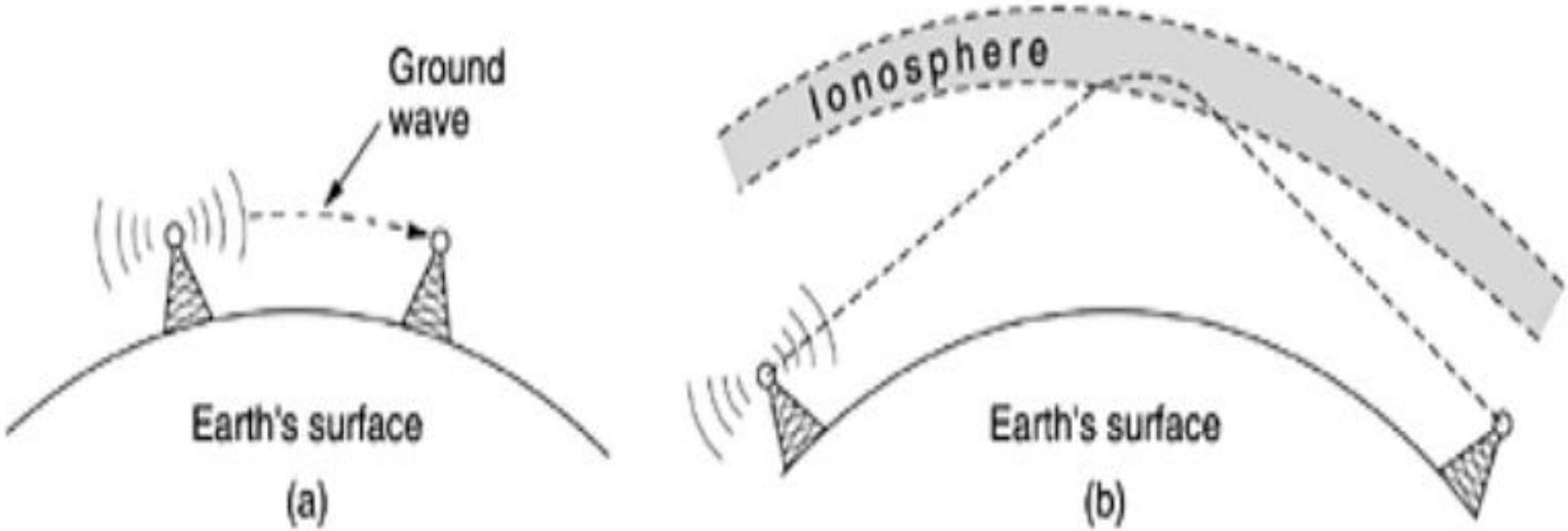
Transmission media

- Radio waves
  - Electromagnetic waves ranging in frequencies between 3KHz and 1GHz are normally called radio waves.
  - Radio waves are omnidirectional.
  - Antenna transmits radio waves, they are propagated in all direction.
  - Sending and receiving antennas do not have to be aligned.



Transmission media

- Radio wave propagate in sky mode.
- It can travel long distances.
- Radio is good for long distance broadcasting.
- Radio waves are of low and medium frequencies.
- It can penetrate walls.



Transmission media

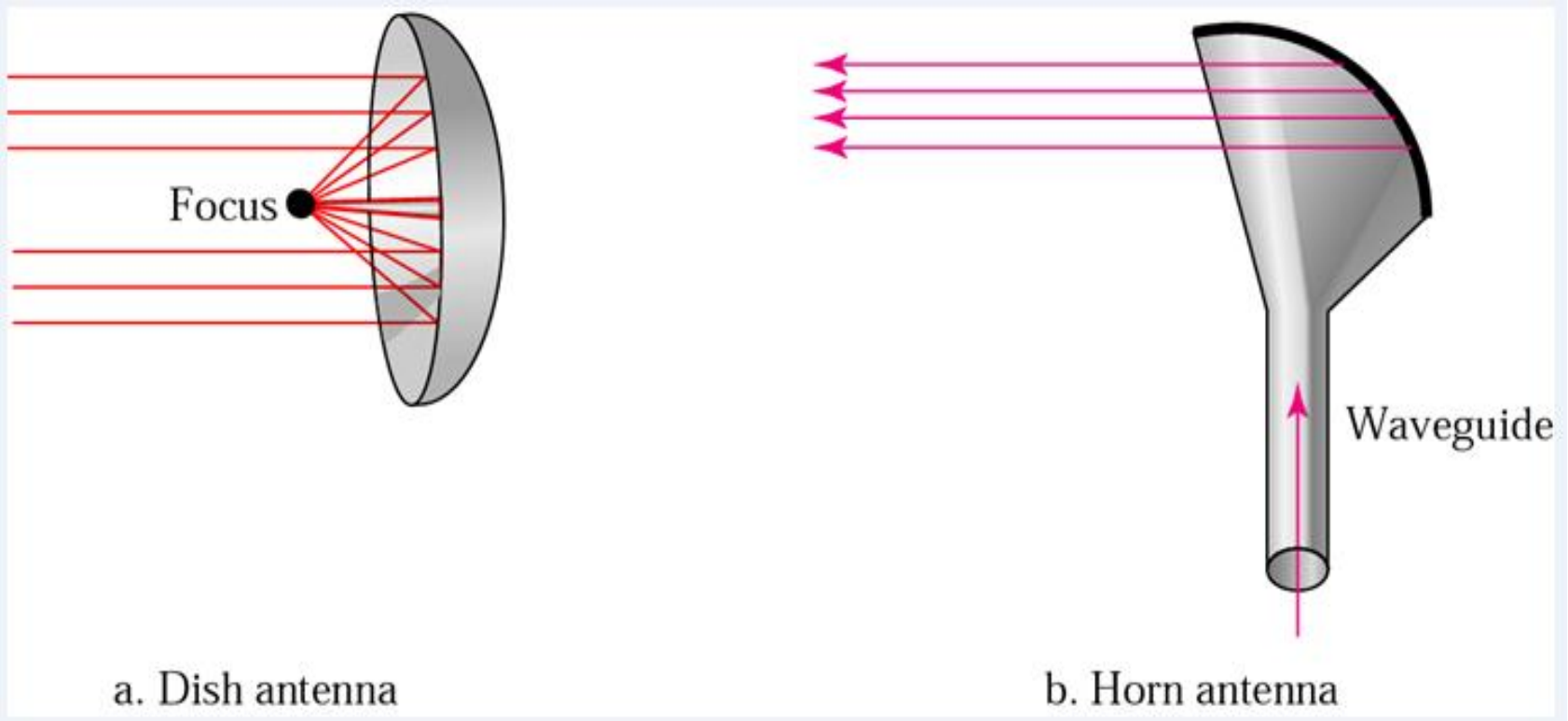
- Applications
  - AM and FM radio
  - Television
  - Maritime radio
  - Cordless phone
- Disadvantage
  - Omnidirectional propagation

Transmission media

- Microwaves
  - Electromagnetics waves having frequencies between 1 and 300GHz are called microwaves.
  - Microwaves are unidirectional.
  - Sending and receiving antenna need to be aligned.
- Characteristics
  - Microwave propagation is line of sight.
  - Towers need to be very tall.
  - Repeaters are required for long distance communication.
  - Very high frequency microwave can penetrate wall.
  - Bands are relatively wide almost 299GHz.
  - To use band requires permission from authorities.

Transmission media

- Unidirectional antennas



- Parabolic dish antenna
  - Every line parallel to line of sight, reflects off the curve at angles such that all lines intersect in common point called the focus.
  - Parabolic dish works as funnel – catching the waves and direct them to common point.

Transmission media

- Horn antenna
  - Looks like gigantic scoop.
  - Outgoing transmissions are broadcast up a stem and deflected upward in a series of narrow parallel beams by the curved head.
- Application
  - Cellular phone
  - Satellite networks
  - Wireless LANs

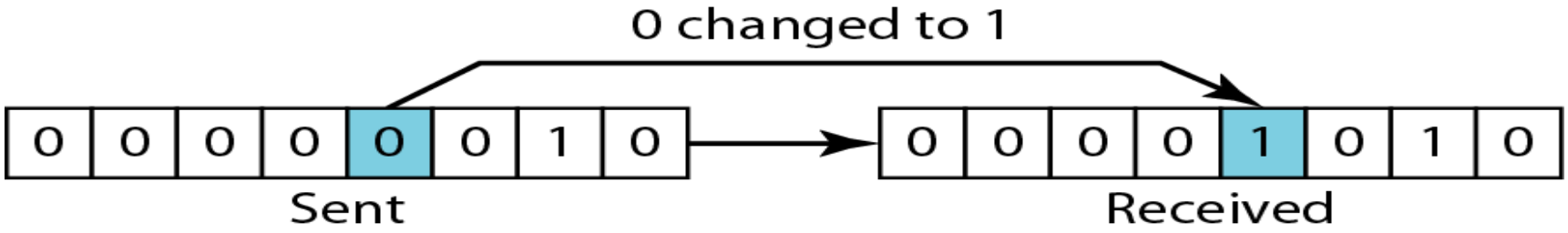
Transmission media

- Infrared waves
  - Infrared signals can be used for short range communication in a closed area using line of sight propagation.
  - Frequencies from 300GHz 400THz
  - Wavelength from 1mm to 770mm.
  - Can not be used outside the building
  - Eg. TV remote
- Applications
  - Transmit digital data at very high rate.
  - IrDA(Infrared Data Association) sponsors the use of infrared waves for communication between devices like keyboard, mice, PCs and printers.
  - Some manufacturers provide IrDA port that allow communication between wireless keyboard and PC.

Data Link Layer

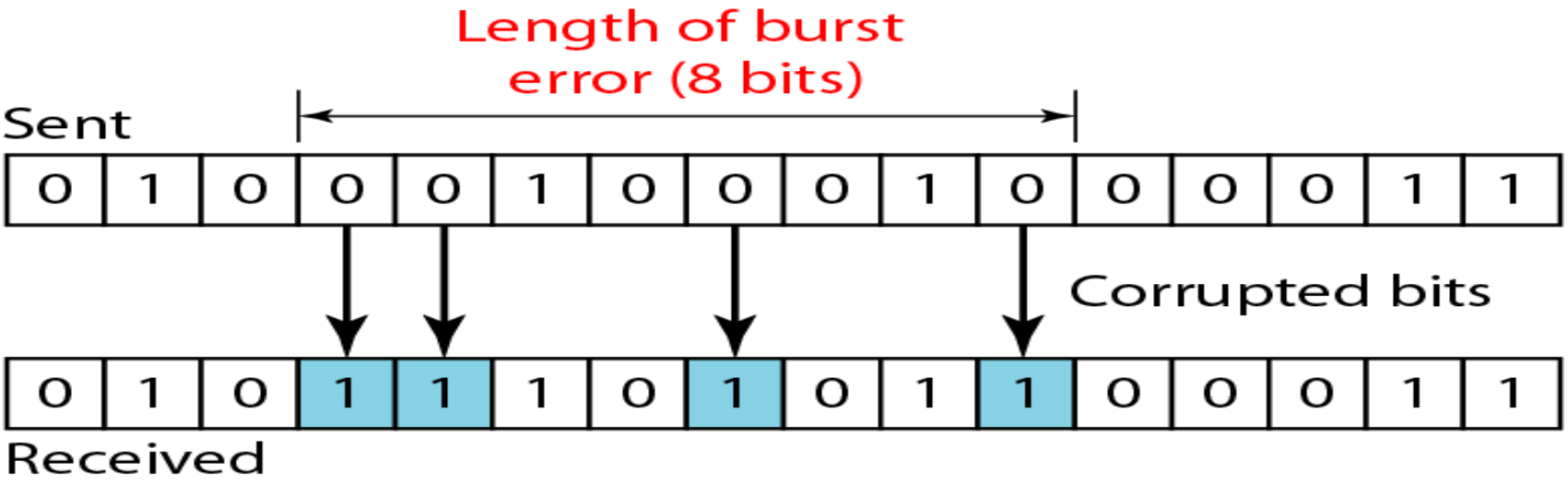
Error Detection

- **Data can be corrupted during transmission. Some application require that errors be detected and corrected.**
- There are two type of error.
  - Single bit error
  - Burst error
- Single bit error
  - In a single bit error, only 1 bit in the data unit has changed.
  - The least likely type of error in serial data transmission.



- Burst error
  - A burst error means that 2 or more bits in the data unit have changed.
  - More likely to occur than a single bit error.

Error Detection



- **Error Detection methods**
- **Error Detection** techniques allow detecting such errors,
- **Error correction** enables reconstruction of original data
- Three methods are available
  - Parity check
    - VRC – Vertical Redundancy Check (parity check)
    - LRC - Longitudinal Redundancy check
  - Checksum
  - CRC – Cyclic Redundancy Check

Parity bits

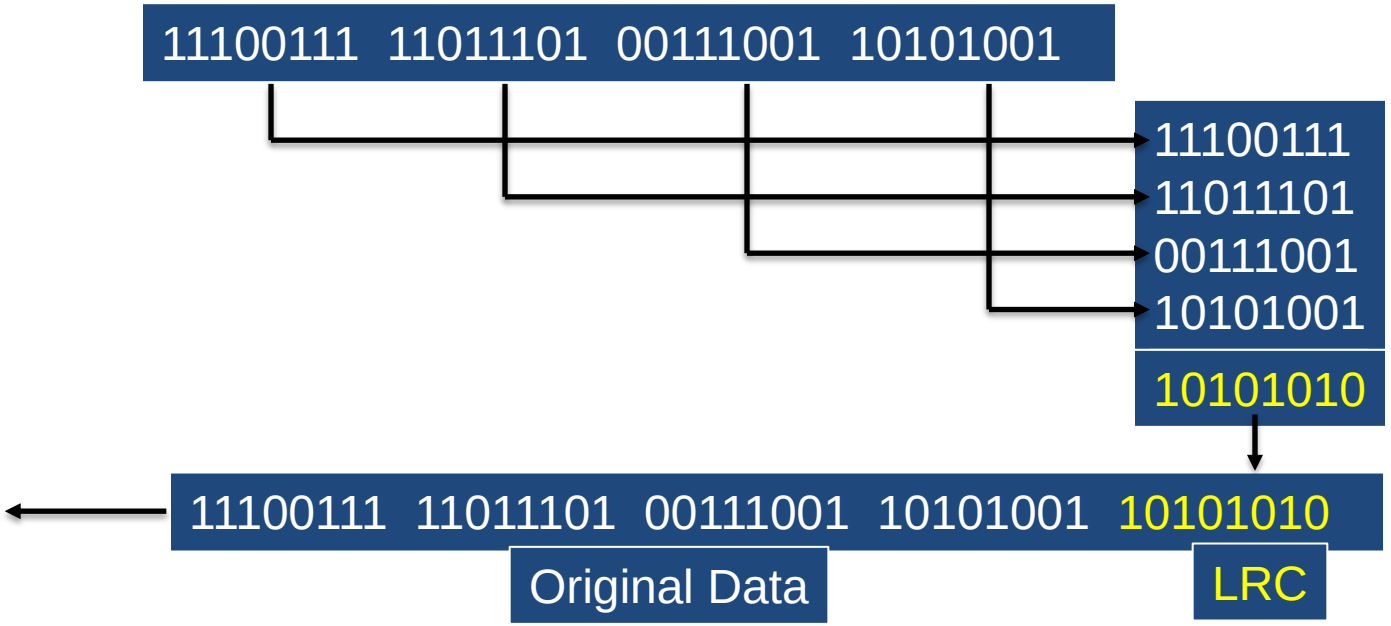
- There are two variants of parity bits:
  - Even parity bit
  - Odd parity bit
- When using even parity, the parity bit is set to 1 if the number of ones in a given set of bit (not including parity bit) is odd, making the number of ones in the entire set of bits (including the parity bit) even.
- If the number of one bits is already even, it is set to a 0.
  - 0110011 -> 01100110
  - 0111011 -> 01110111
- Odd parity
  - When using odd parity, the parity bit is set to 1 if the number of ones in a given set of bits (not including the parity bit) is even, keeping the number of ones in the entire set of bits (including the parity bit) odd.
  - And when the number of set bits is already odd, parity bit is set to 0.
    - 0110011 -> 01100111
    - 0111011 -> 01110110

Vertical Redundancy Check (VRC)

- Appending a single bit at the end of data block such that the number of ones is even.
- Example
  - 0110011 -> 01100110
  - 0111011 -> 01110111
- VRC is known as **parity check**.

Longitudinal Redundancy Check (LRC)

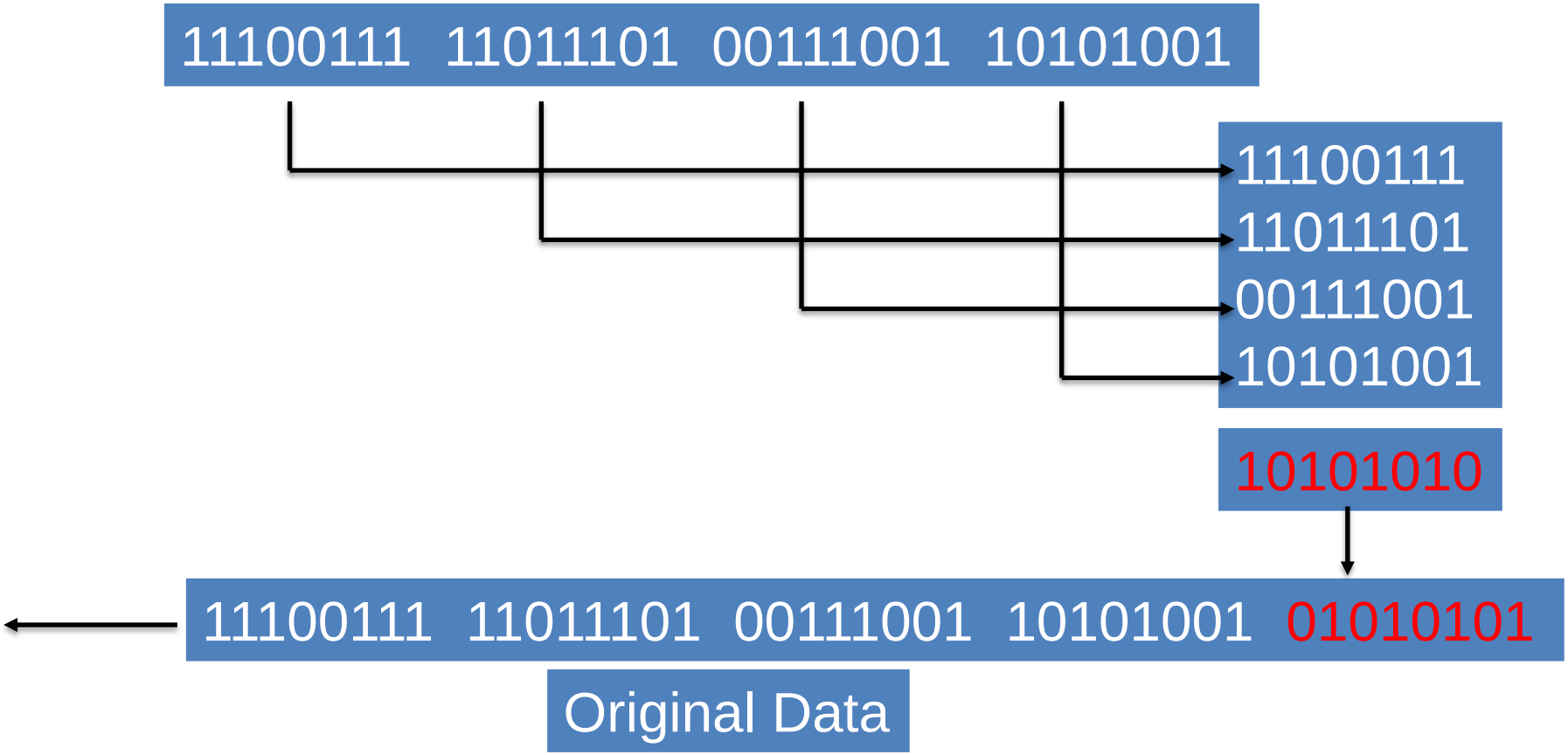
- Sender side:
  - Do the modulo addition of the given number (XOR)
  - Attach calculated LRC answer with data.
  - Send the combined data.
- Receiver side
  - Do modulo addition of all the numbers without including LRC.
  - Compare calculated data with LRC.
  - If both are same, than it is assumes no error otherwise there is an error.
- Organize data into a table and create a parity for each column



Checksum

- Sender side:
  - Do the addition of the given numbers
  - Do the 1's complement of the result
  - Attach calculated 1's complemented answer with data.
  - Send the combined data.
- Receiver side:
  - Do addition of all the numbers including the checksum.
  - Calculate 1's complement of answer.
  - If result is 0 than it assumes no error otherwise there is an error.

Checksum


11100111 11011101 00111001 10101001
11100111
11011101
00111001
10101001
10101010
11100111 11011101 00111001 10101001 01010101
Original Data

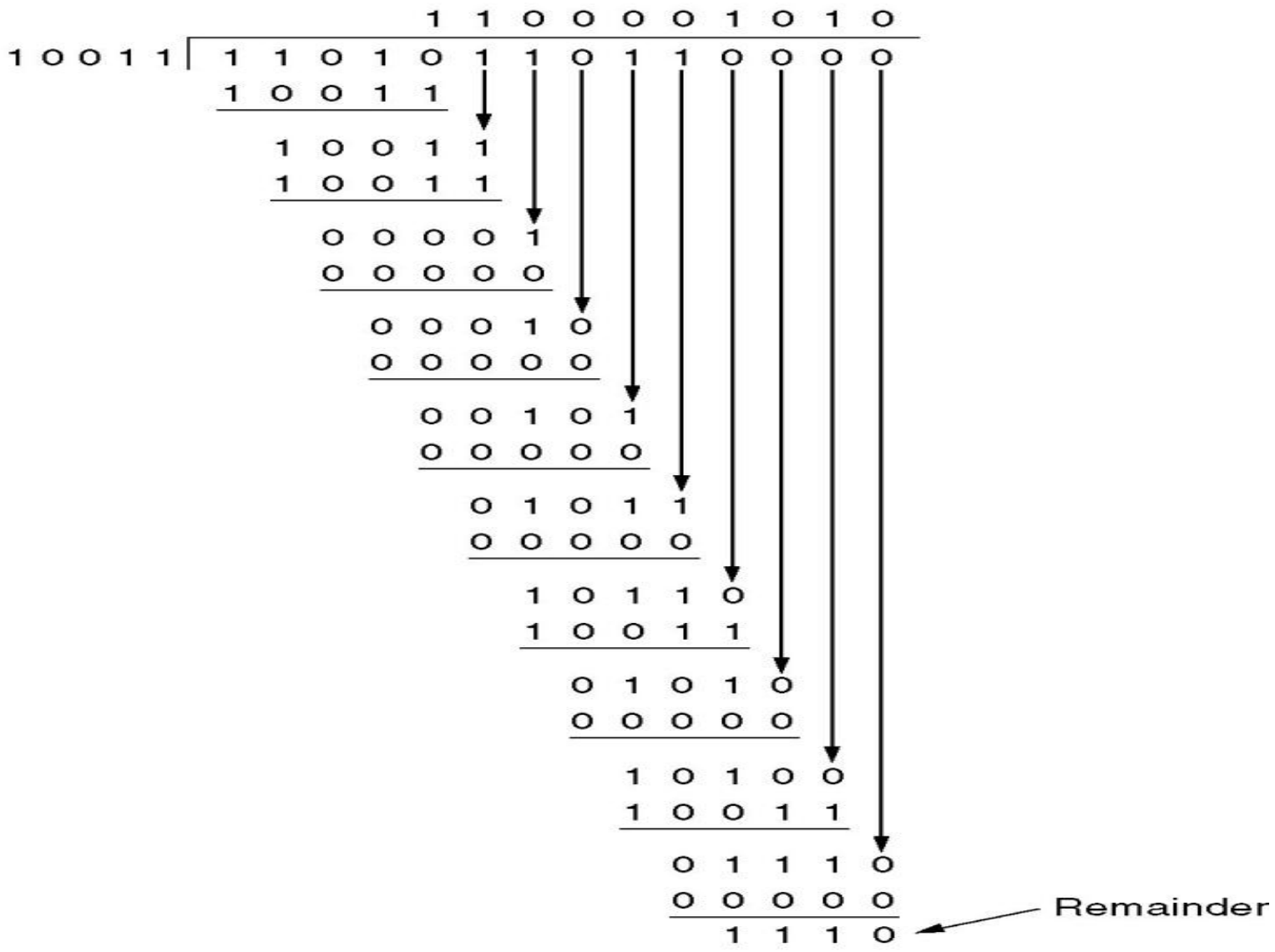
Cyclic Redundancy Check (CRC)

- It is also known as polynomial code.
- Cyclic Redundancy Check
  - Bit Stream M(x)
  - Generator Polynomial G(x)
- Steps to do CRC
  - Let r be the degree of G(x), append r zero to lower order end of the frame, so m+r bits.
  - Divide the lower order bit stream by G(x)
  - Add the reminder to the lower order bit stream.
  - Transmit the data.
- Eg. Frame 1101011011 using the generator
  - G(x)=X^4+X+1
- Sender side
  - M(x)=110011 -> X^5+X^4+X+1 (6bits)
  - G(x)=11001 -> X^4+X^3+1 (5 bits, r=4)
    - 4 bits of redundancy
  - From M(x) -> 1100110000 divide M(x) by G(x) to find C(x).

Frame : 1 1 0 1 0 1 1 0 1 1

Generator: 1 0 0 1 1

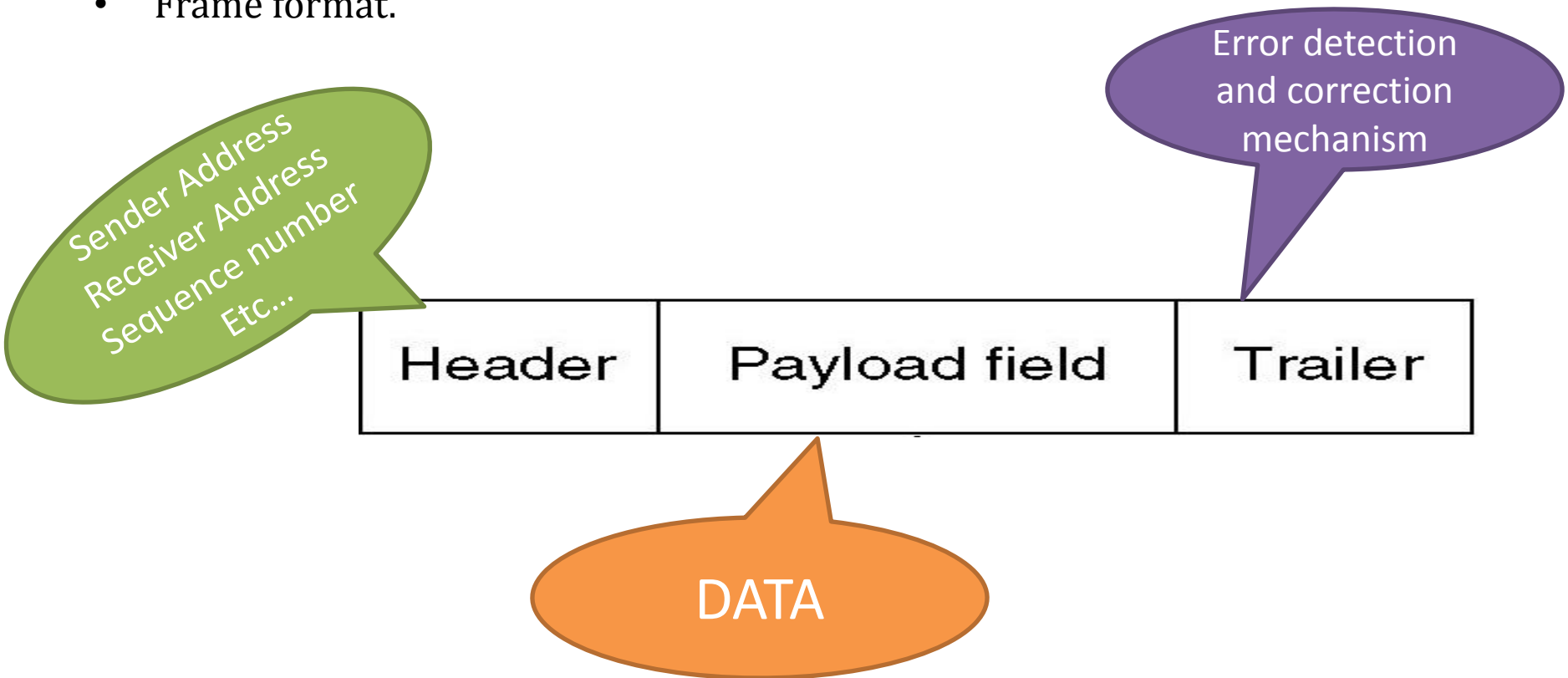
Message after 4 zero bits are appended: 1 1 0 1 0 1 1 0 1 1 0 0 0 0



Transmitted frame: 1 1 0 1 0 1 1 0 1 1 1 1 1 0

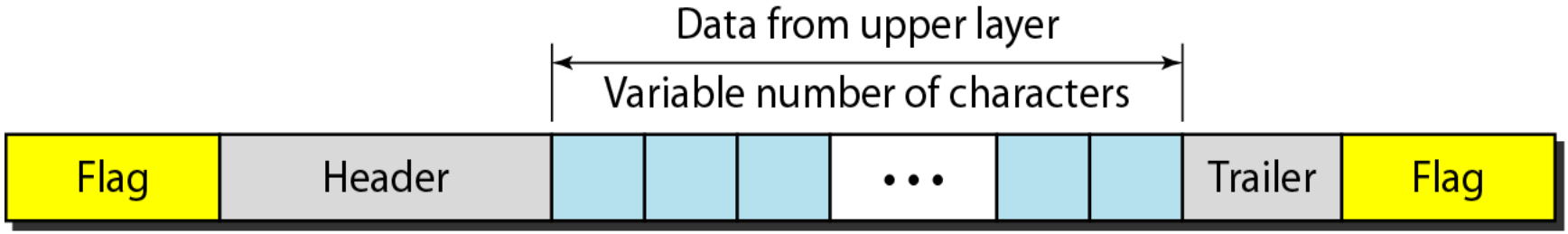
Framing

- The data link layer need to pack bits into frame, so that each frame is distinguishable from another.
- “Framing separates the messages by adding a sender address and receiver address”.
- Message could be packed in frame.
- When a message is carried in one large frame, even a single bit error would require retransmission of the whole message.
- Frame format.



Framing

- There are two type of framing
  - Fixed size framing
    - No need to define boundaries of the frames, the size itself can be used as a delimiter..
  - Variable size framing
    - We need to define the end of the frame and beginning of the next.
- Framing techniques
  - Character oriented approach
    - Character count
    - Byte stuffing
  - Bit oriented approach
    - Bit stuffing
- **Character count**



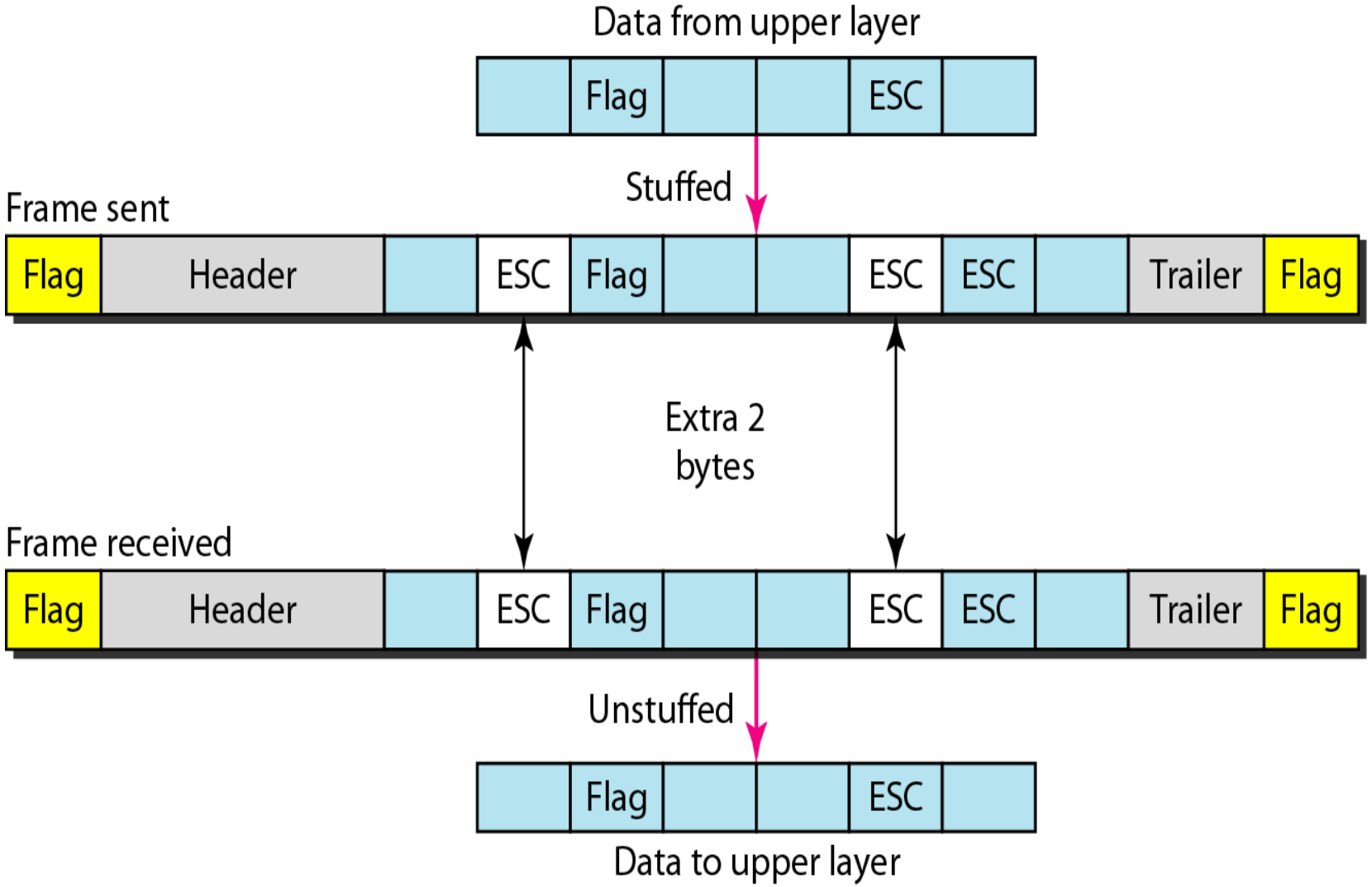
- Data to be carried are 8-bit character.

Character Count

- **Header**: carries the source and destination address and other control information.
- **Trailer**: which carries error detection and error correction redundant bits, are multiples of 8 bits.
- **Flag**: flag is added at beginning and ending of the frame to separate one frame from the next.
  - The flag composed of protocol dependent special character that signals the start or end of the frame.
- **Problem with character count**
  - It is popular when only text was exchanged by the data link layer.
  - This technique is not used, if we send other types of information such as graph, audio and video.
  - For that byte stuffing strategy was added with character count framing.

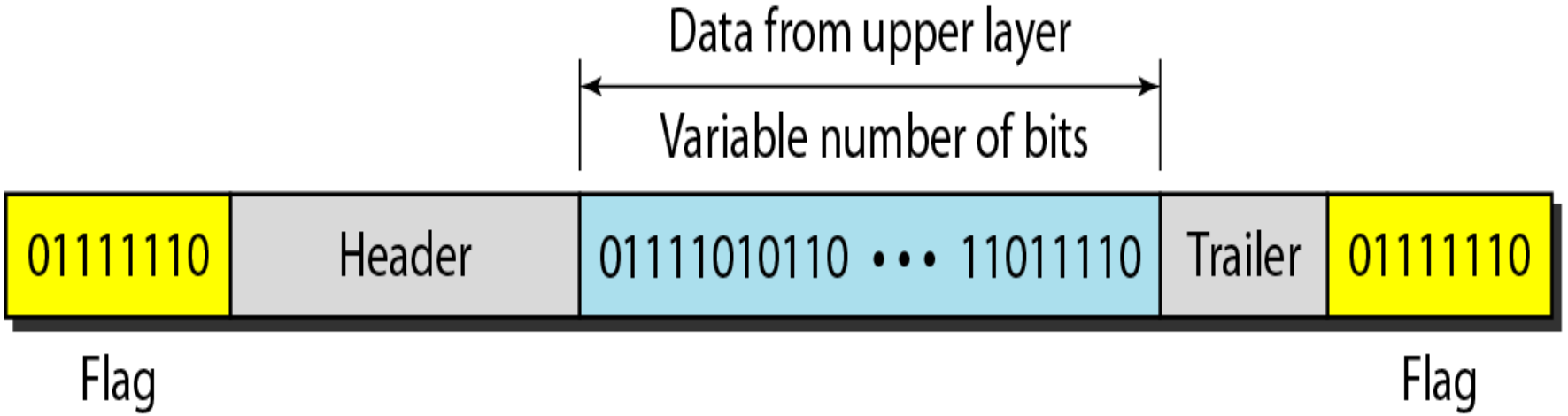
Byte Stuffing

- A special byte is added to the data section of the frame when there is a character with same pattern as the flag.
- The data section is stuffed with an extra byte. This byte is usually called the **escape character (ESC)** which has predefined bit pattern.
- Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data.
- **Byte stuffing is the process of adding 1 extra byte whenever there is a flag or escape character in the text.**
- Problem with byte stuffing.
  - If text contains one or more escape characters followed by flag, receiver removes the escape character but keeps the flag, which is incorrectly interpreted as the end of the frame.
- Solution of this problem.
  - the escape character that are the part of the text must also be marked by another escape character.


Data from upper layer
Flag
ESC
Stuffed
Frame sent
Flag
Header
ESC
Flag
ESC
ESC
Trailer
Flag
Extra 2
bytes
Frame received
Flag
Header
ESC
Flag
ESC
ESC
Trailer
Flag
Unstuffed
Flag
ESC
Data to upper layer

Bit Stuffing

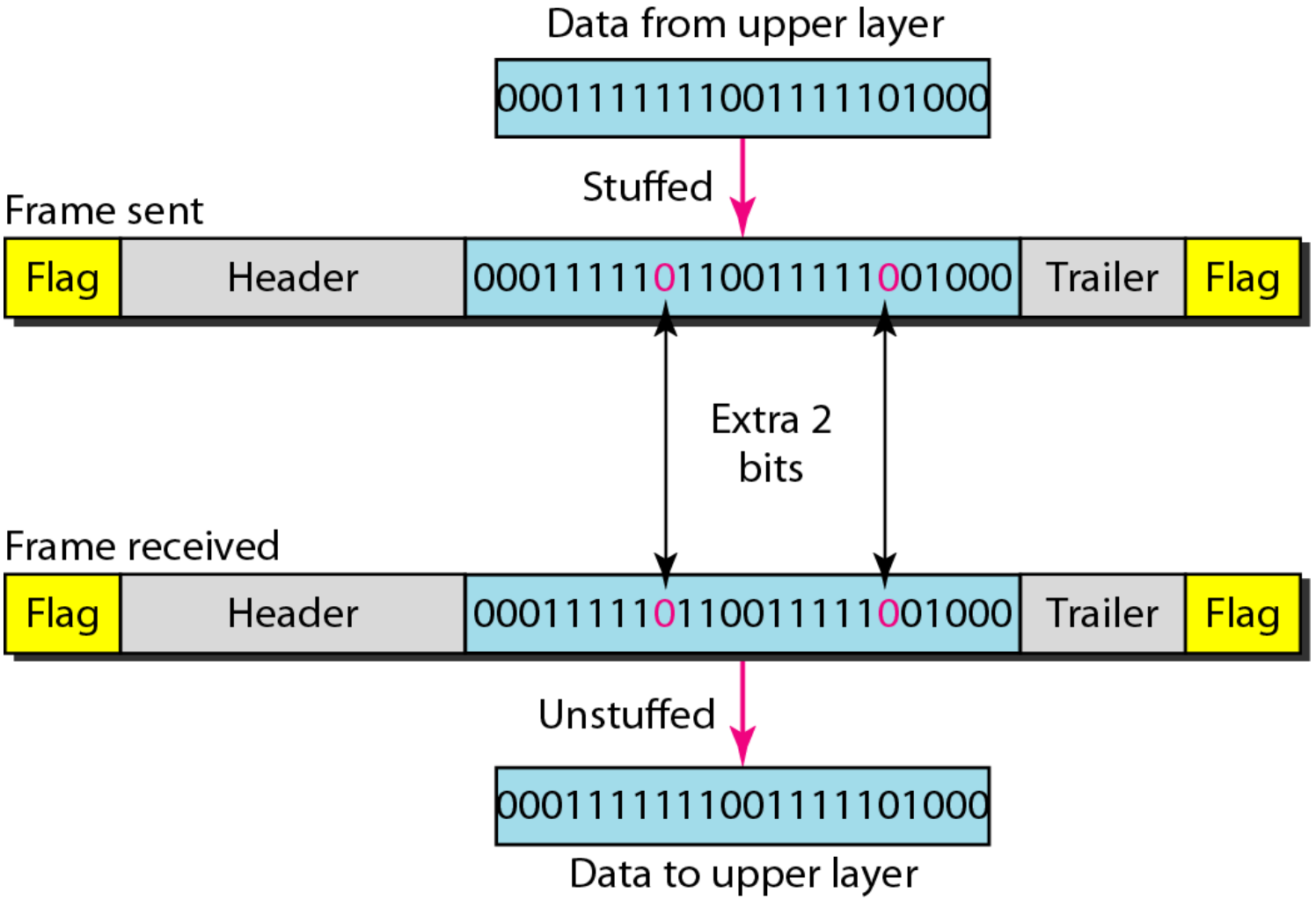
- The data section of a frame is a sequence of bits to be interpreted by the upper layer as text, graphics, audio, video and so on.
- We still need a delimiter to separate one frame from the other. Most protocol use a special 8-bit pattern flag 01111110 as the delimiter to define the begging and the end of the frame.



- This flag can create same problem we saw in the byte-oriented protocols that is, if the flag pattern appears in the data, we need to inform the receiver that this is not the end of the frame.
- We do it by stuffing 1 single bit (instead of 1 byte) to prevent the pattern from looking like flag. This strategy is called **bit stuffing**.

Bit Stuffing

- Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag.



Flow Control & Error Control

- The most important responsibility of data link layer are flow control and error control. Collectively, these functions are known as data link control.

Note

Flow control refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment.

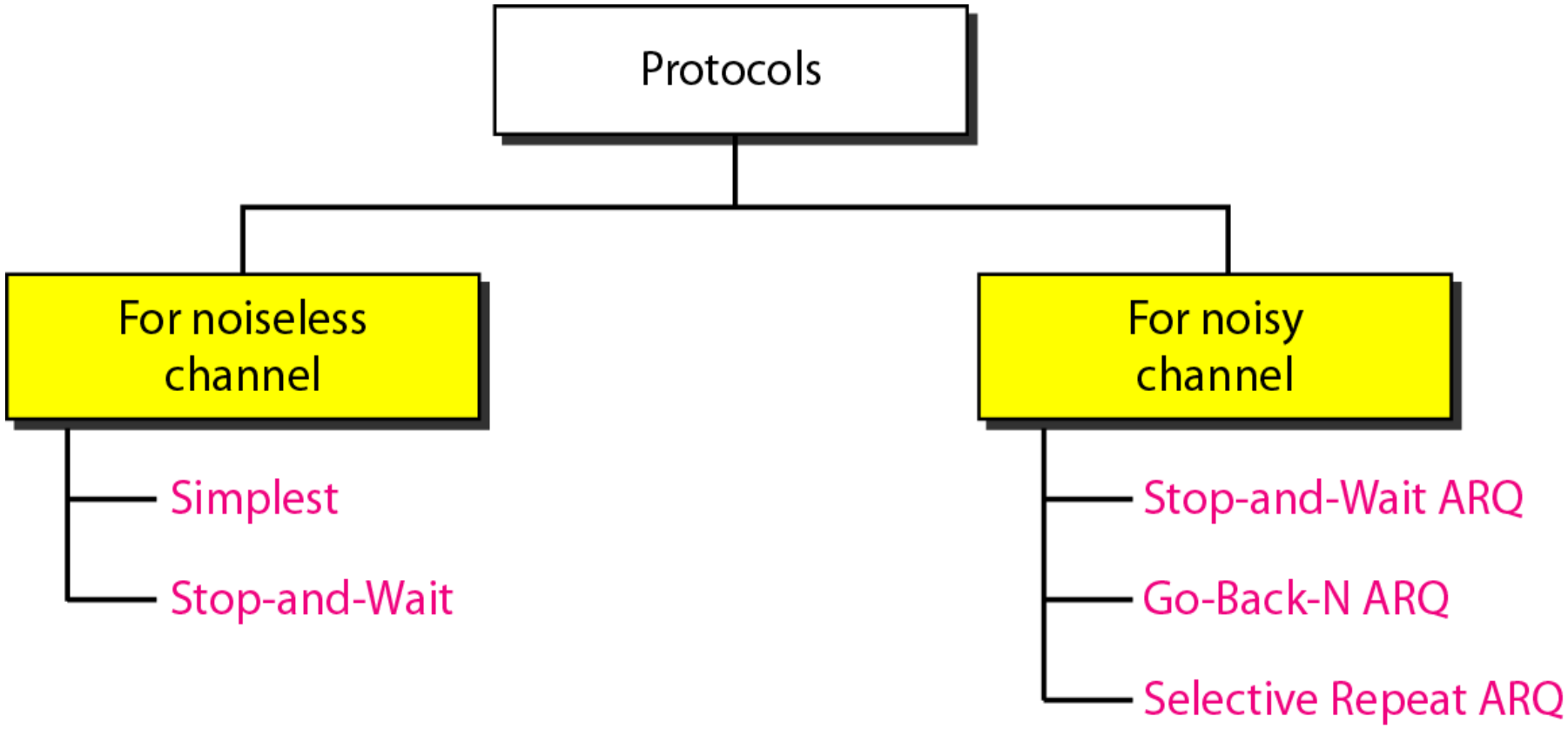
- Any time an error is detected in an exchange, specified frames are retransmitted. This process is called **Automatic Repeat Request**.

Note

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

Protocols

- Now let us see how the data link layer can combine framing, flow control, and error control to achieve the delivery of data from one node to another.
- The protocols are normally implemented in software by using one of the common programming language.
- To make our discussion language free, we have written in pseudo code a version of protocol that concentrates mostly on the procedure instead of delving into details of language rules.



Noiseless channels

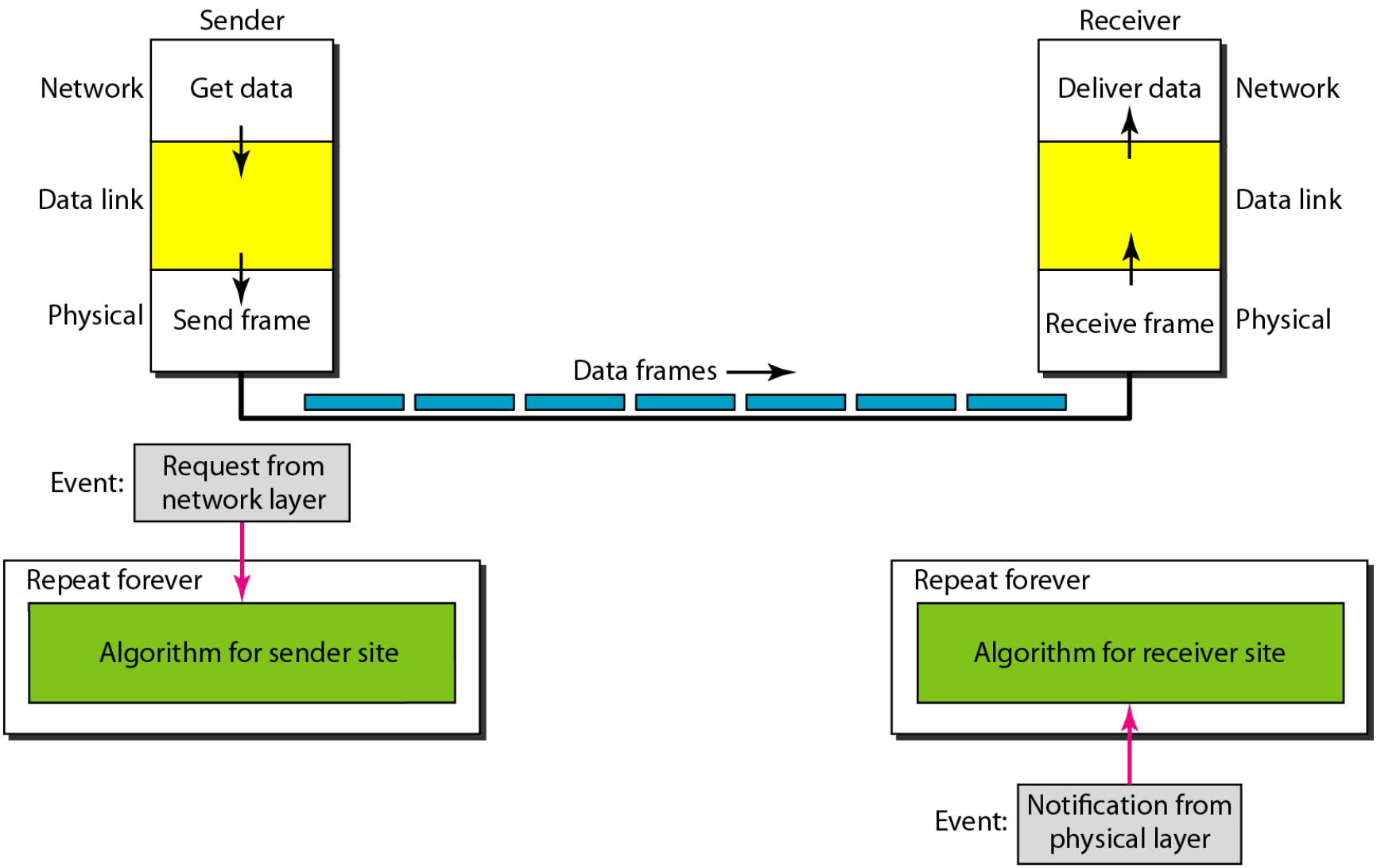
- Let us first assume we have an ideal channel in which no frames are lost, duplicated or corrupted. We introduce two protocols for this type of channel.
- Two types:
  - Simplest protocol
  - Stop – and – wait protocol.

Simplest Protocol

- Name simplest protocol because of it is one that has **no flow** or **no error control**.
- It is unidirectional protocol in which data frames are travelling in one direction.
- Receiver can never be overwhelmed with incoming frame.
  - Receiver can immediately handle any frame it can receives with negligible time.
- Design
  - The data link layer at the sender side gets data from it's network layer, make a frame out of the data and send it.
  - The data link layer at the receiver side receives a frame from it's physical layer. Extracts data from frame, and deliver to its network layers.
  - Two events:
    - Procedure at sender side:
      - There is no action until there is request from network layer.
    - Procedure at receiver side
      - No action until notification from physical layer arrives.

Simplest Protocol

- ***The design of the simplest protocol with no flow and error control***



Simplest Protocol

- ***Sender side algorithm for the simplest protocol***

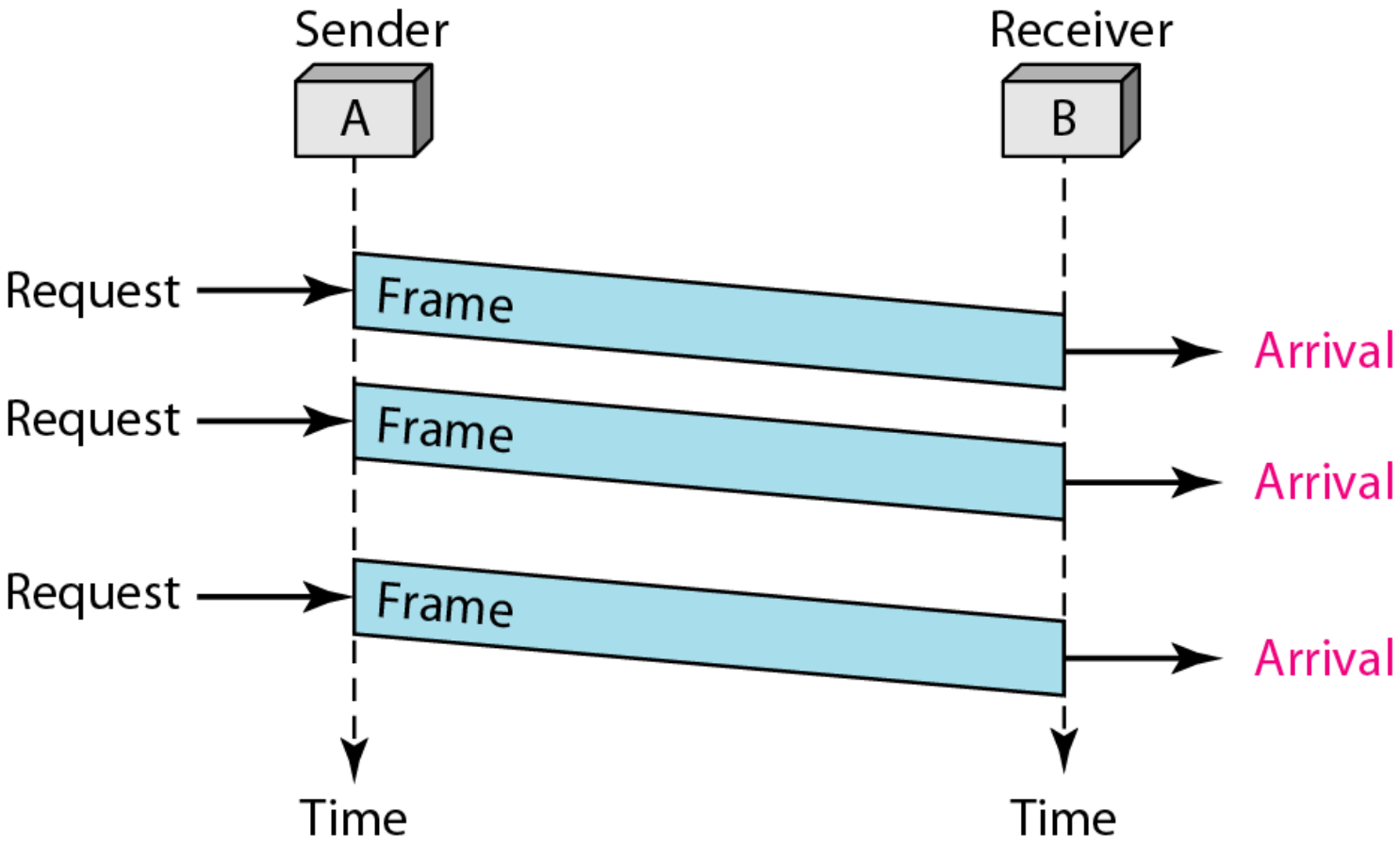
```
1  while(true)                          // Repeat forever
2  {
3    WaitForEvent();                    // Sleep until an event occurs
4    if(Event(RequestToSend))           //There is a packet to send
5    {
6       GetData();
7       MakeFrame();
8       SendFrame();                    //Send the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

- ***Receiver side algorithm for 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

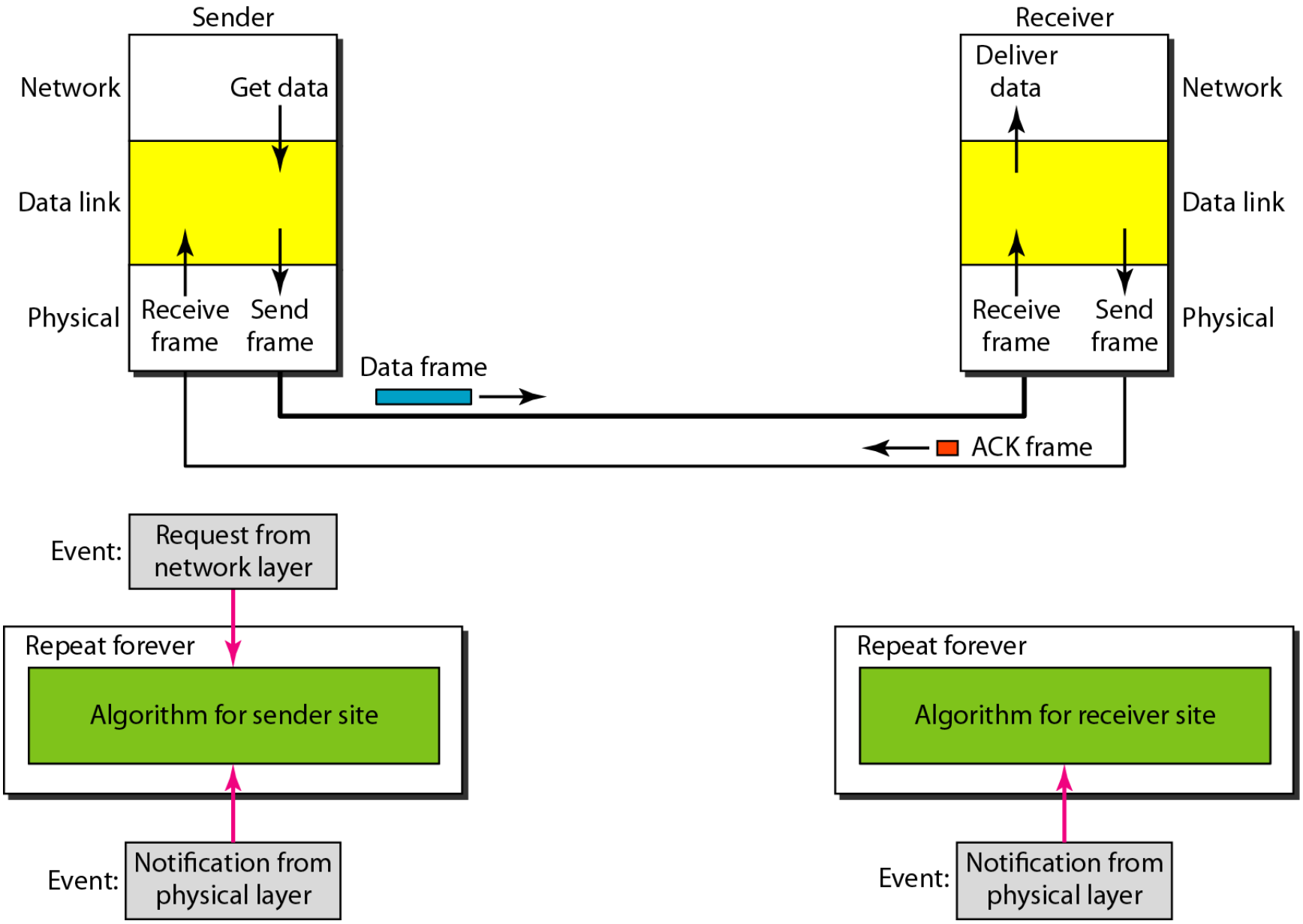
- Example



Stop – and – wait Protocol

- Name because the sender sends one frame, stop until it receives confirmation from receiver, and than send the next frame.
- ACK frame (token of acknowledgement) in other direction.
- Design of stop – and – wait protocol
  - In this traffic on the forward channel (sender to receiver) and reversed channel.
  - At nay time, there is either one data frame on the forward channel or one ACK on the reverse channel.
  - It uses half duplex mode.

Stop – and – wait Protocol



Stop – and – wait Protocol

- *Sender-side algorithm for 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 }
```

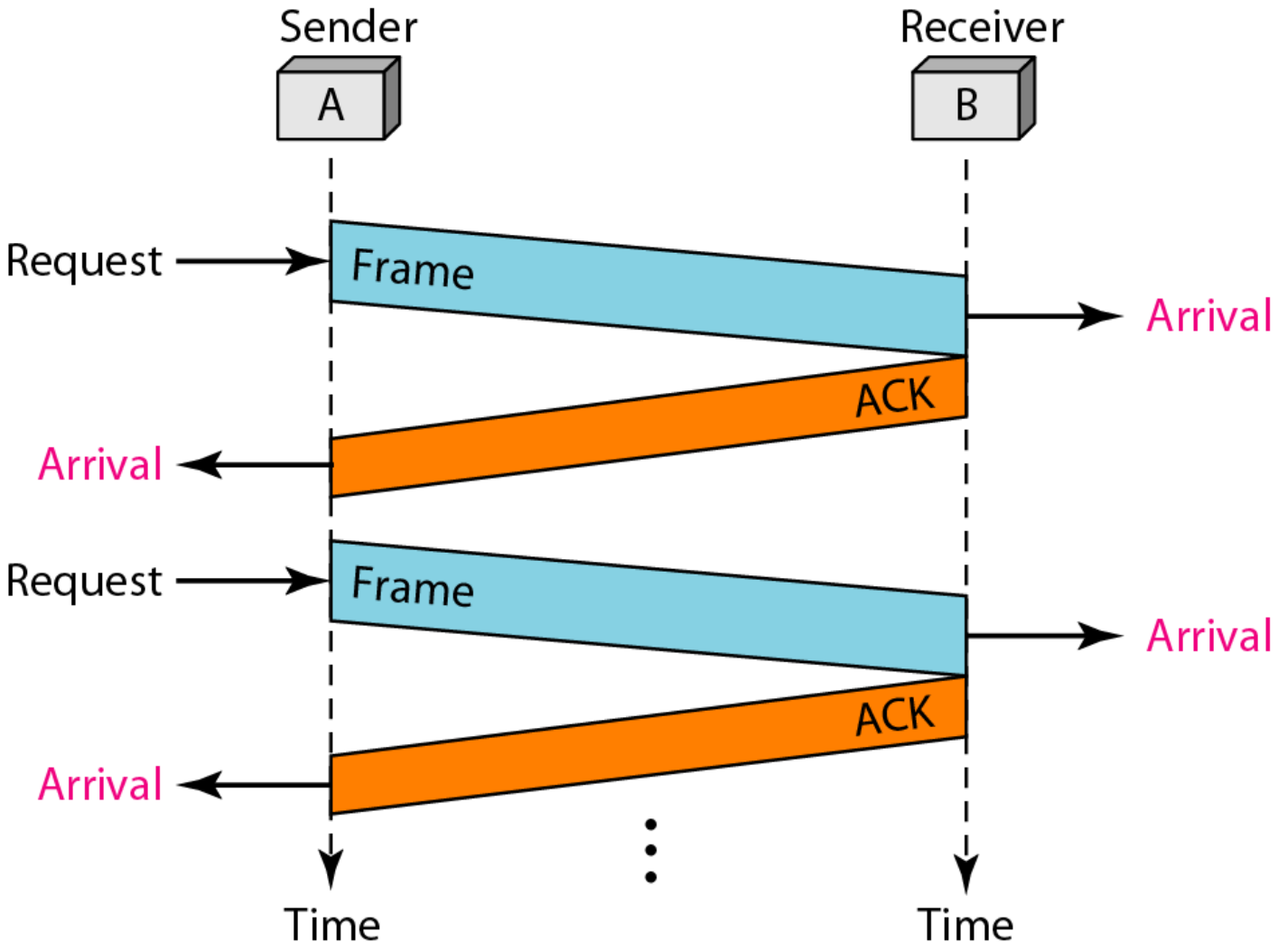
Stop – and – wait Protocol

- *Receiver-side algorithm for 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 }
```

Stop – and – wait Protocol

- *Example*



Noisy Channel

- Although the stop – and – wait protocol gives us an idea of how to add flow control to its predecessor, noiseless channel are nonexistent.
- 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.

Stop – and – Wait Automatic Repeat Request

- It adds simple error control mechanism to the stop – and – wait protocol.
- When frame arrives at the receiver site, it is checked and if it corrupted, it is discarded.
- *If receiver does not respond when there is an error, how can the sender know which frame to resend?*
- The sender keeps a copy of the sent frame, at the same time, it starts timer.
- If the timer expires and there is no ACK for sent, the frame is resent, the copy is held, and the timer restarted.

Note

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.

Stop – and – Wait Automatic Repeat Request

- **Sequence number**
- A field is added to the data frame to hold sequence number of that frame.
- If we decide that field is m bit long, the sequence number start from 0 to 2^m-1 and then are repeated.
- We have used x as a sequence number, we need to use x+1 after that. There is no need for x+2.

Note

In Stop-and-Wait ARQ, we use sequence numbers to number the frames.
The sequence numbers are based on modulo-2 arithmetic.

Stop – and – Wait Automatic Repeat Request

- Assumes that sender has sent frame number x.
- **Three things can happen.**
  - The frame arrives safe and receiver sends an acknowledgement. The acknowledgement arrives at the sender sites causing the sender to send the next frame numbered x+1.
  - The frame arrives safe and receiver sends an acknowledgment, but acknowledgement is corrupted or lost. The sender resend the frame (number x) after time out. The receiver can recognize this fact because it expects frame x+1 but x was received.
  - The frame is corrupted or even arrives at the receiver site, the sender resends the frame (number x) after the time out.
- **Acknowledgement numbers**
  - The acknowledgement numbers always announce the sequence number of next frame excepted by receiver.
  - Eg. If frame 0 has arrived safe and sound, the receiver sends an ACK frame with acknowledgement 1 (means frame 1 is expected next) and vice versa.

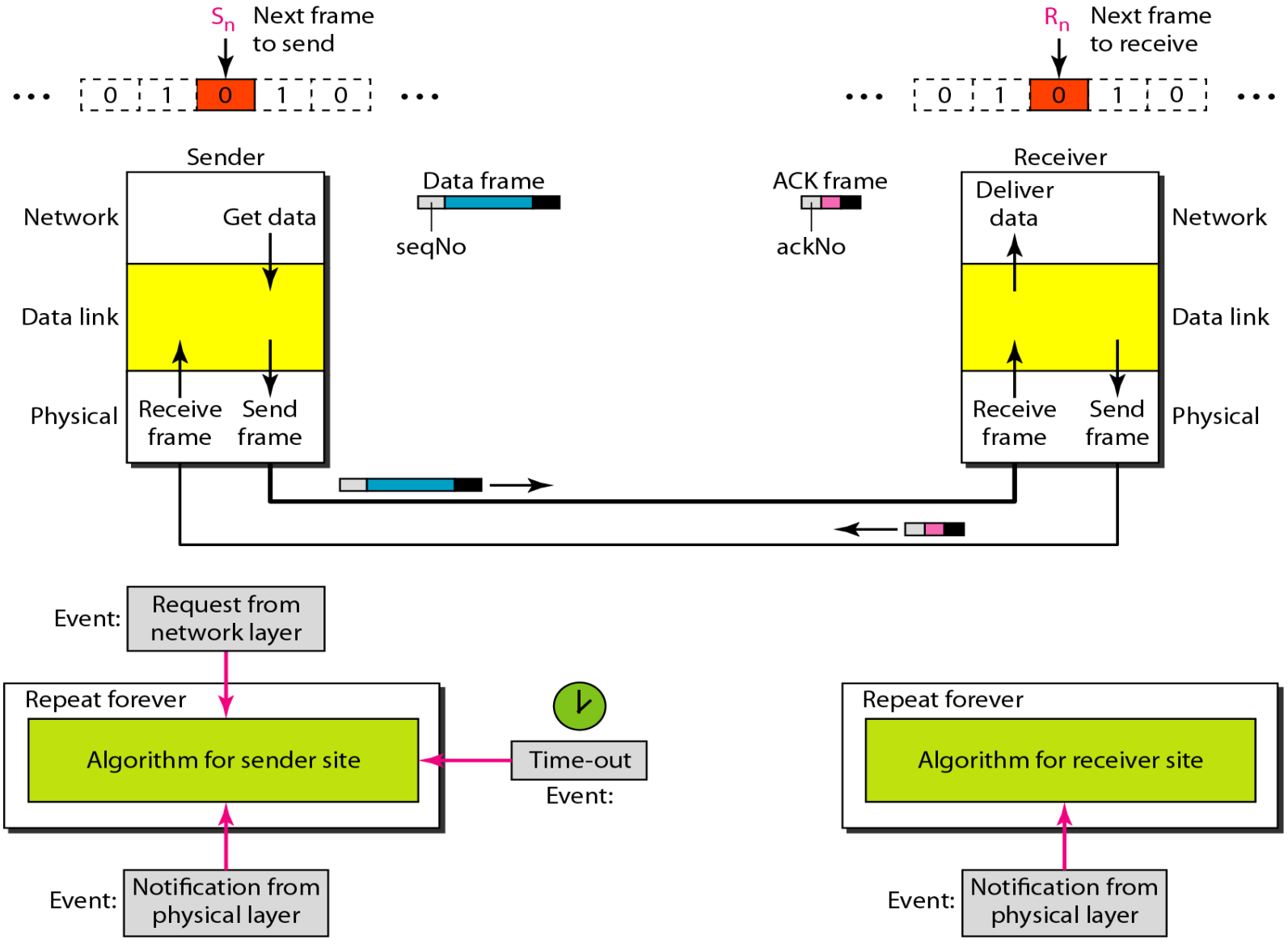
Stop – and – Wait Automatic Repeat Request

Note

In Stop-and-Wait ARQ, the acknowledgment number always announces in modulo-2 arithmetic the sequence number of the next frame expected.

Stop – and – Wait Automatic Repeat Request

- *Design of stop – and – wait ARQ protocol*



Stop – and – Wait Automatic Repeat Request

- Sender site algorithm for stop – and – wait ARQ

```
 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
```

Stop – and – Wait Automatic Repeat Request

- Sender site algorithm for stop – and – wait ARQ

(Continued)

```
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  }
```

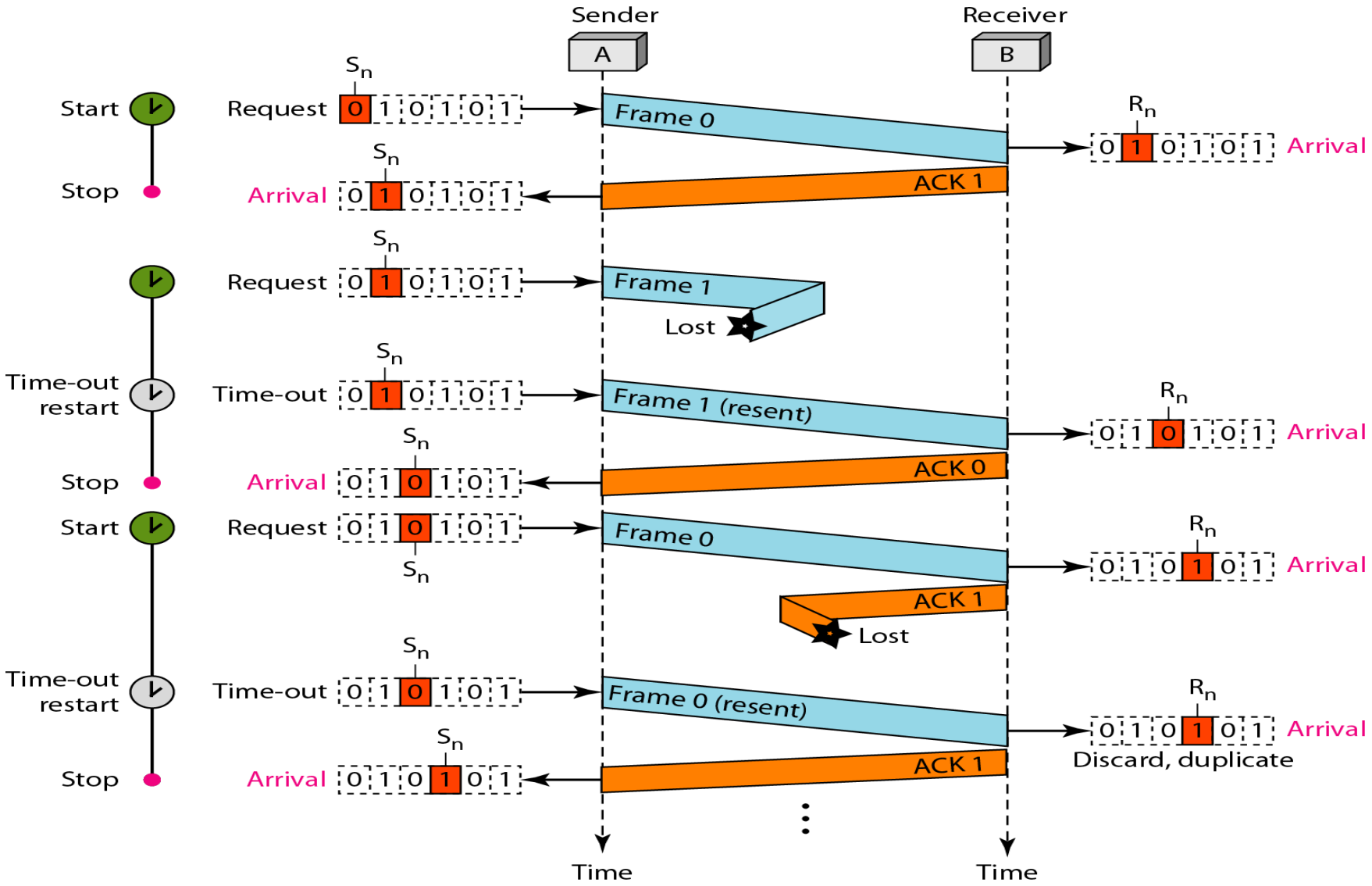
Stop – and – Wait Automatic Repeat Request

- Receiver site algorithm for stop – and – wait ARQ

```
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 Automatic Repeat Request

- Example



Go Back N ARQ

- To improve the efficiency of transmission, multiple frame must be in transmission while waiting for acknowledgement.
- We need to let more than one frame to keep the channel busy while the sender is waiting for acknowledgement.
- We achieve it by Go Back N ARQ protocol.
- In this protocol we can send several frames before receiving acknowledgement; we keep a copy of these frames until the acknowledgement arrives.
- It include concepts like
  - Sequence number
  - Sliding window
  - Timer
  - Acknowledgement
  - Resending frame
- **Sequence number**
  - Frames from a sending station are numbered sequentially.
  - We need to include the sequence number of each frame in header, we need to set an limit.
  - If m=4 then sequence number are 0 to 15. then we can repeat sequence.

Go Back N ARQ

- 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,0,1,2,3,4,……15.

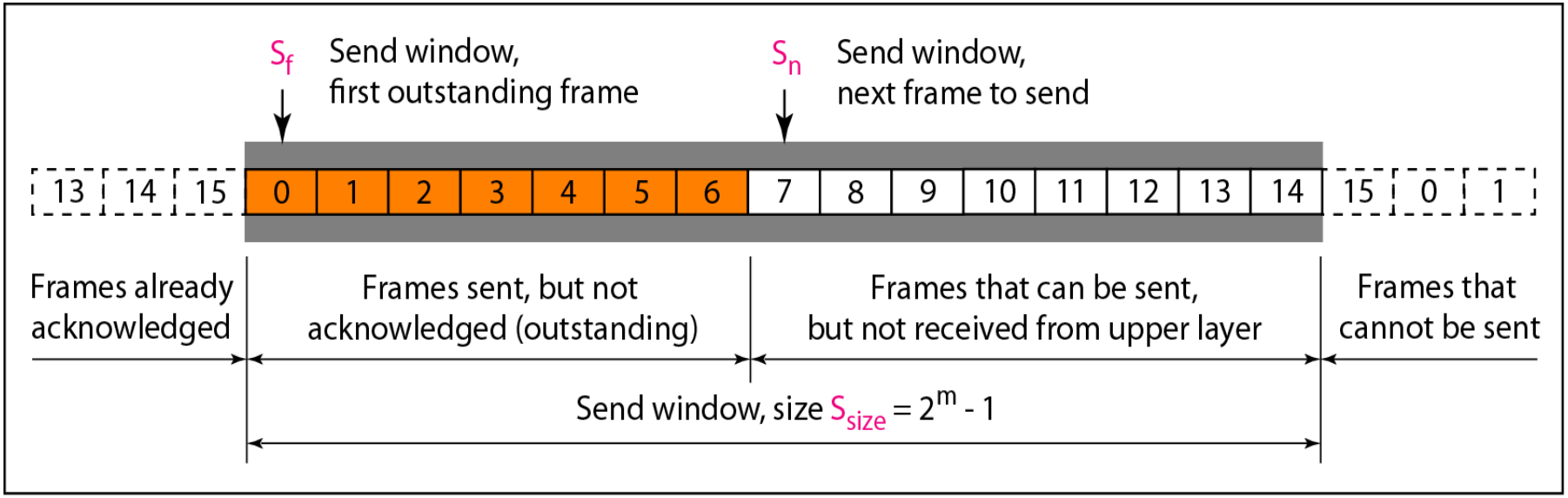
Note

In the Go-Back-N Protocol, the sequence numbers are modulo 2^m,
where m is the size of the sequence number field in bits.

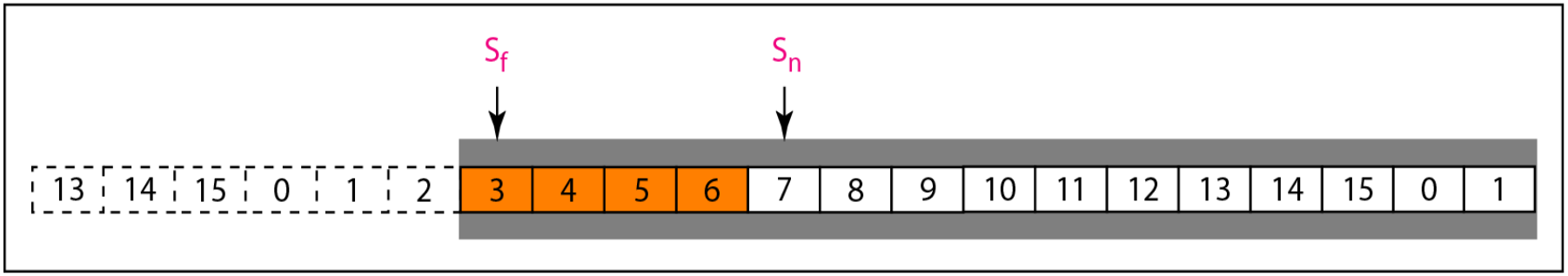
- **Sliding Window**
  - It is a abstract concept that defines the range of sequence numbers that is the concern of the sender and receiver.
  - The sender and receiver need to deal with only part of possible sequence number.
  - The range which is the concern of the sender is called **sender sliding window**.
  - The range which is the concern of the receiver is called **receiver sliding window**.
  - In each window position, some of these sequence number defines
    - Frame that have been send
    - Frame that can be send

Go Back N ARQ

- *Send window for Go Back N ARQ*



a. Send window before sliding



b. Send window after sliding

Go Back N ARQ

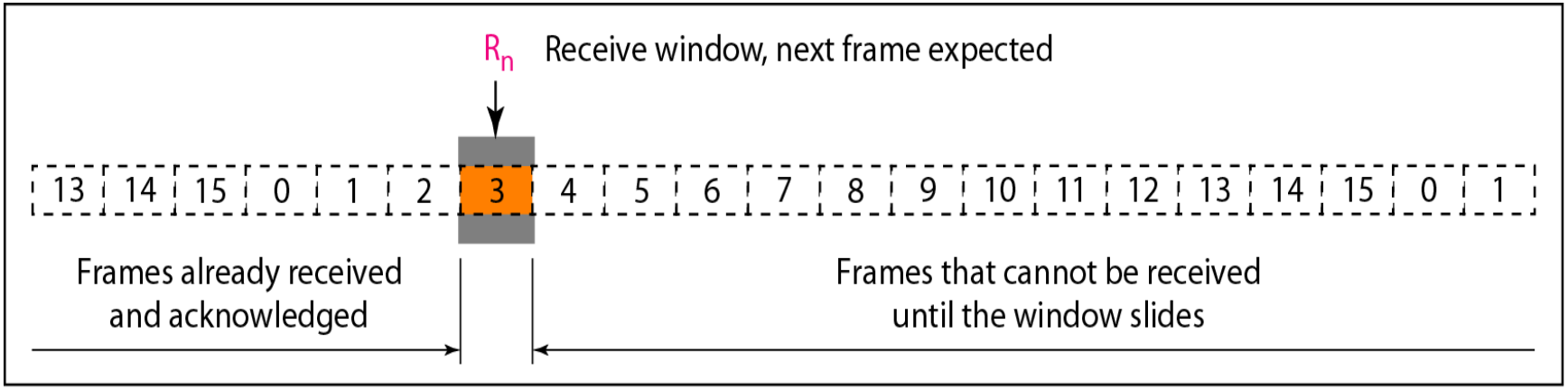
Note

The send window is an abstract concept defining an imaginary box of size $2^m - 1$ with three variables: S_f, S_n, and S_{size}.

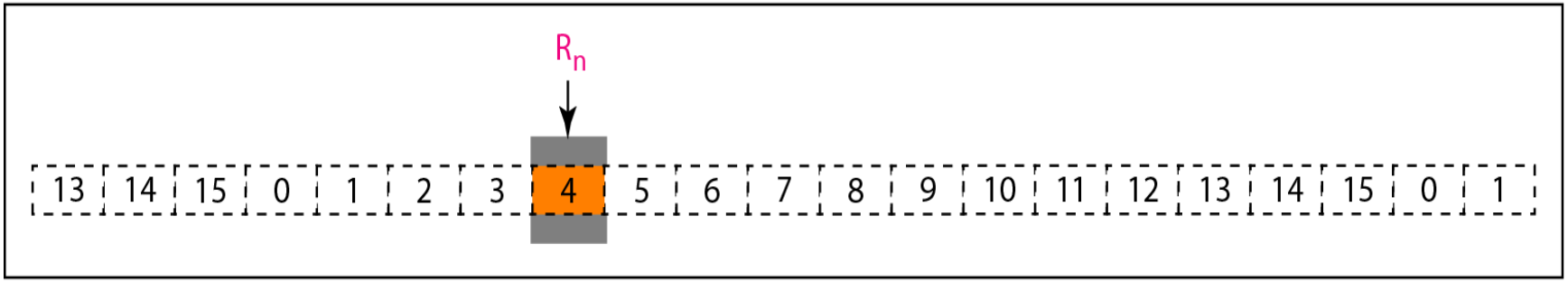
- The window defines 3 variables:
  - S_f (Send Window, first outstanding frame)
    - Defines sequence number of first (oldest) outstanding frame.
  - S_n (send window, next frame to be sent)
    - Holds sequence number that will assigned to next frame to be sent.
  - S_{size} (send window, size)
    - Define size of window, which is fixed in our protocol.

Go Back N ARQ

- **Receiver window**
  - Receiver window make sure that the correct data frames are received and that the correct acknowledgment are sent.
  - The size of receiver window is always 1.
  - The receiver always looking for the arrival of a specific frame.
  - Any frame arriving out of the order is discarded and need to be resend.



a. Receive window



b. Window after sliding

Note

Go Back N ARQ

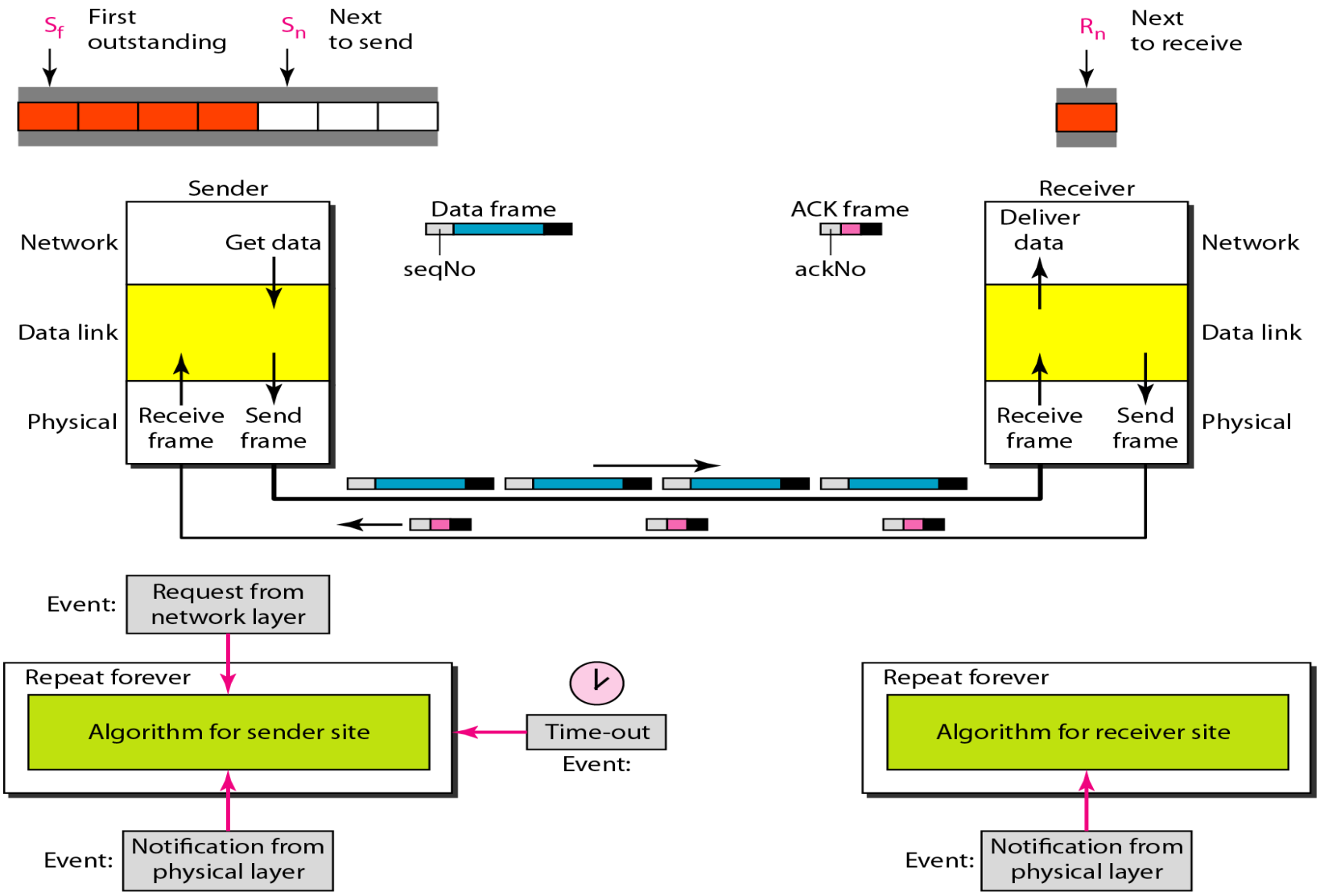
The receive window is an abstract concept defining an imaginary box of size 1 with one single variable R_n. The window slides when a correct frame has arrived; sliding occurs one slot at a time.

- R_n (Receive window, next frame expected)
- The receiver window also slides, but only one at a time.
- When correct frame is received, the window slides.
- **Timer**
  - Only 1 timer for each sent frame.
  - The timer for first outstanding frame always expires first.
  - We send all outstanding frame when timer expires.

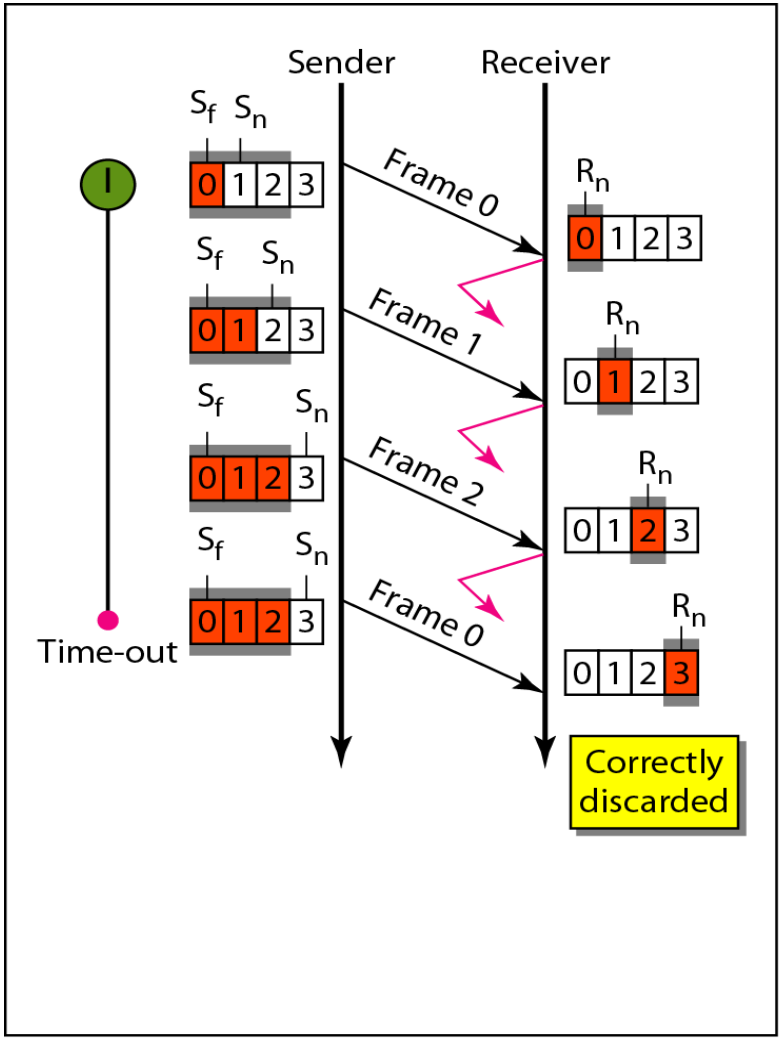
Go Back N ARQ

- **Acknowledgment**
  - The receiver sends positive acknowledgment if frame has arrived safe and sound in order.
  - If frame is damage or out of order, the receiver discard all the frame until it receives the one it is expecting.
  - This cause timer to expire.
- Resending Frame
  - When timer expires, resending all the outstanding frames.
  - This process is called Go Back N ARQ.

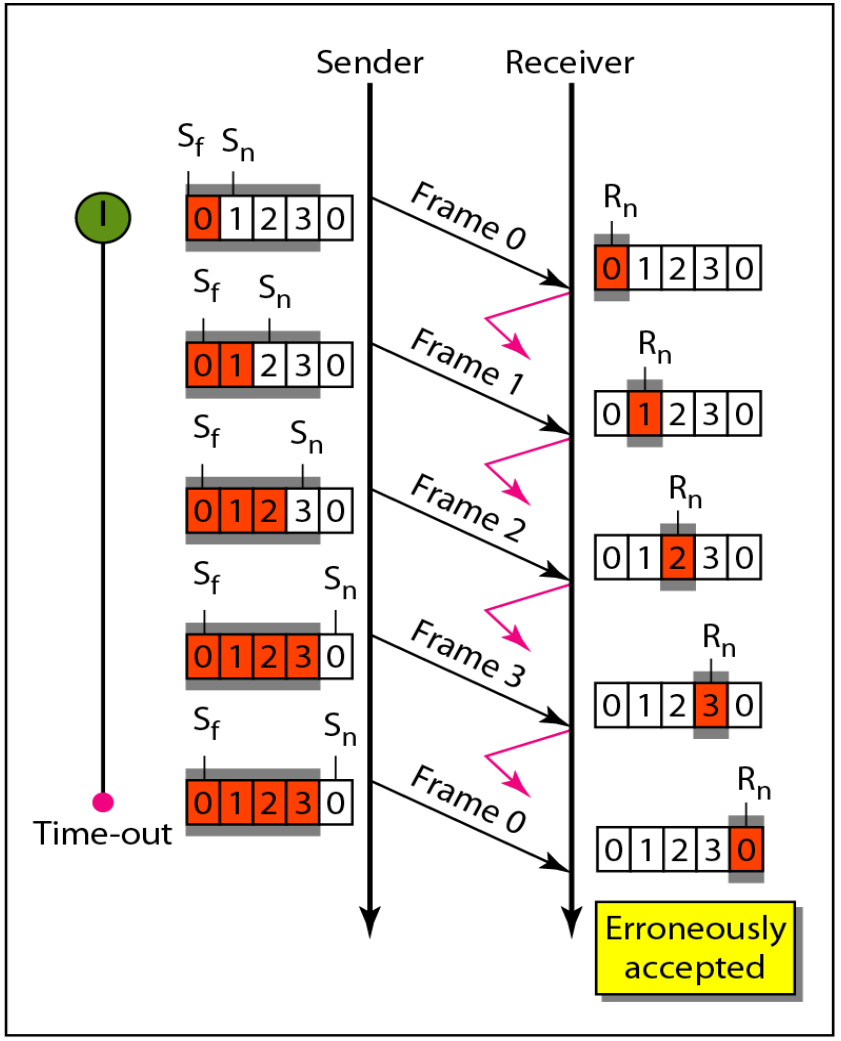
Go Back N ARQ


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



a. Window size < 2^m



b. Window size = 2^m

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
```

Go Back N Sender algorithm

```
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 }
```

Go Back N Receiver algorithm

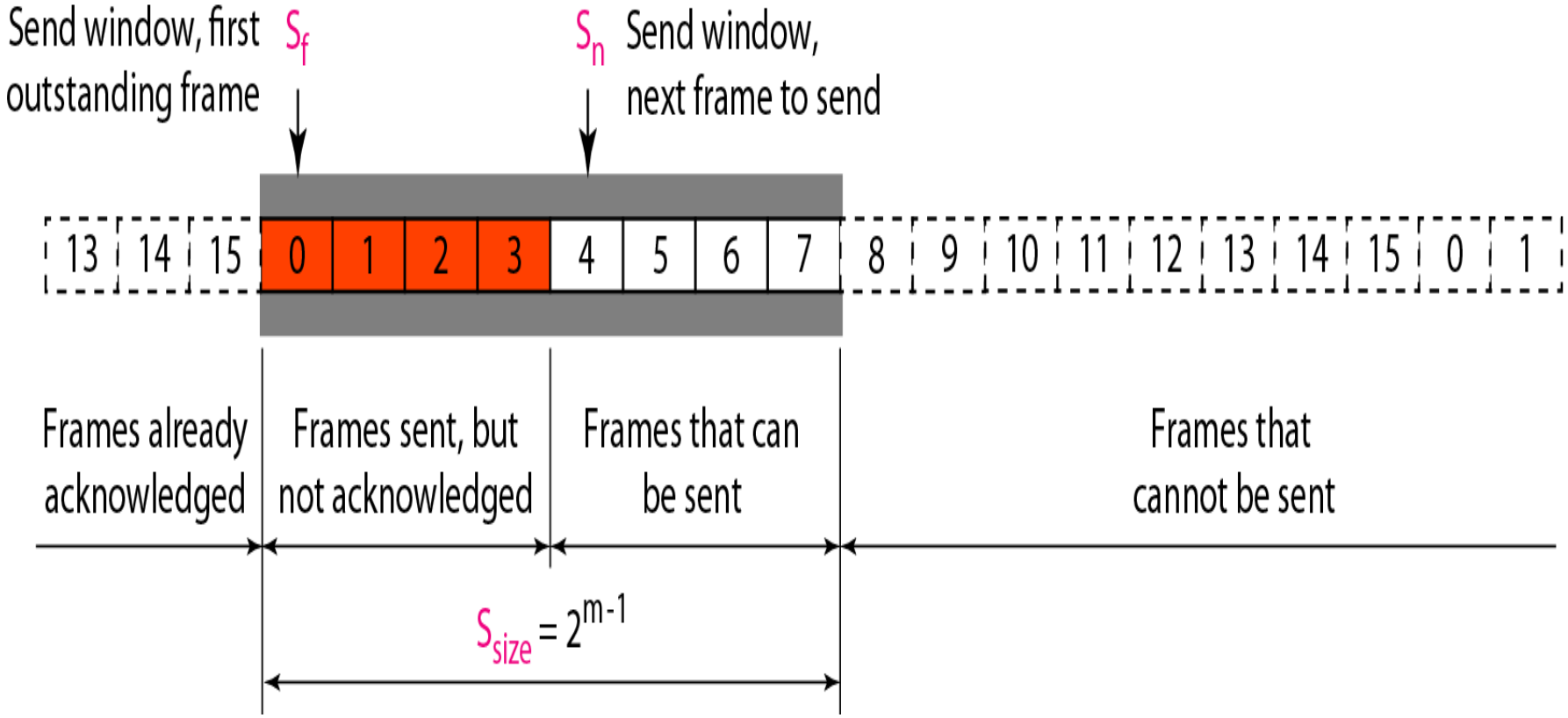
```
1  Rn = 0;
2
3  while (true)                          //Repeat forever
4  {
5    WaitForEvent();
6
7    if(Event(ArrivalNotification)) /Data frame arrives
8    {
9       Receive(Frame);
10      if(corrupted(Frame))
11           Sleep();
12      if(seqNo == Rn)                  //If expected frame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

Selective Repeat ARQ

- Go Back N simplifies the process at the receiver site.
  - The receiver keep the track of only one variable, and there is no need to buffer out-of-order frame.
- This protocol is very inefficient for noise link.
- In this higher probability of damage, which means the resending of multiple frame.
- So there is another mechanism that does not resent N frame when just one frame is damage. Only damage frame is resent. This mechanism is called Selective Repeat ARQ.
- **Windows**
  - It uses 2 window: send window and receive window.
  - The size of send window in 2^{m-1}.
  - The receiver window is same size as send window.
  - Eg. M=4, the size of sequence number is 0-15 but size of window is just 8.
  - This protocol uses same variable as Go Back N.

Selective Repeat ARQ

- ***Send window for Selective Repeat ARQ.***

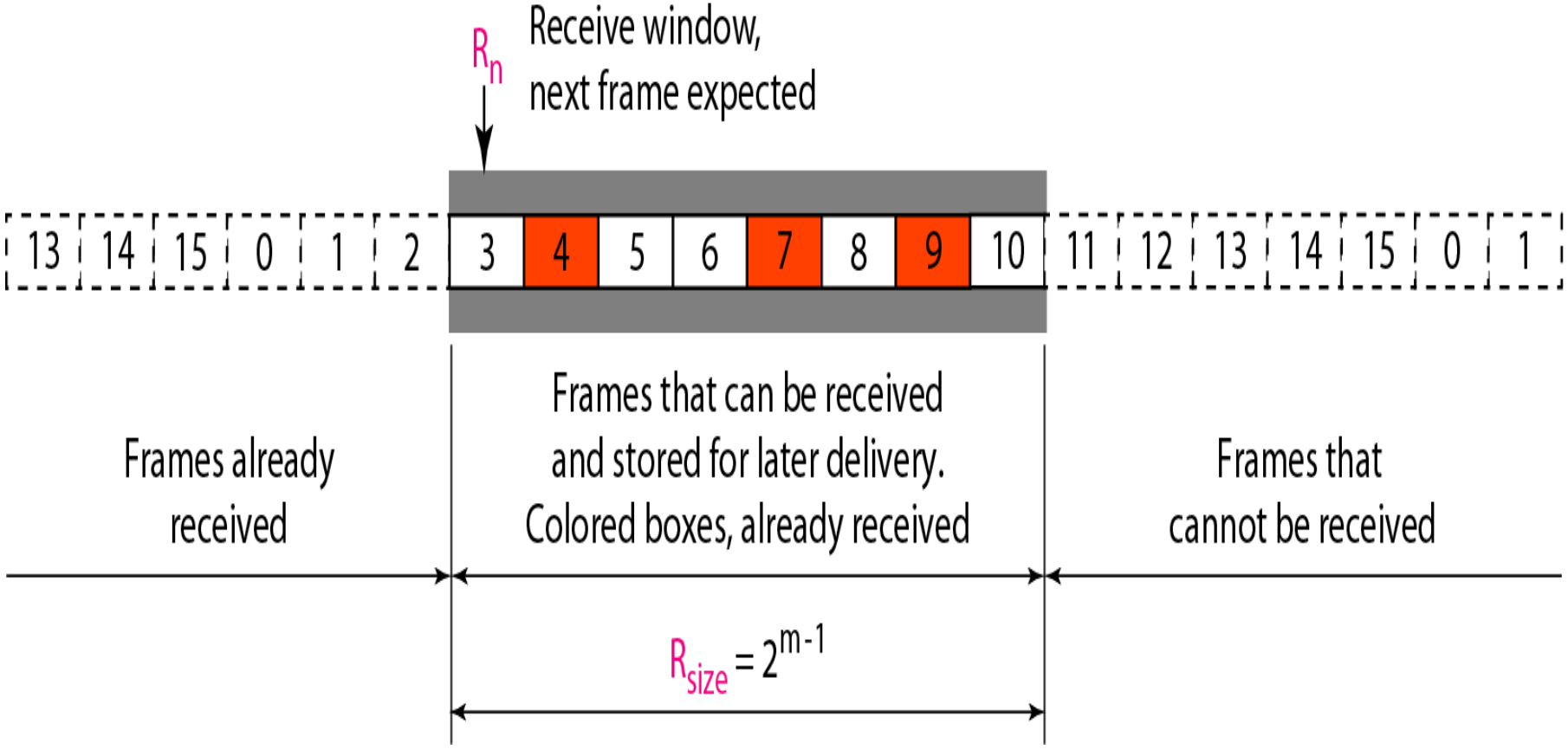


- **Receiver window**
  - Size of receiver window is same as the size of send window (2^{m-1}).
  - The Selective Repeat protocol allows as many frame as the size of the receiver window to arrive out of the order.

Selective Repeat ARQ

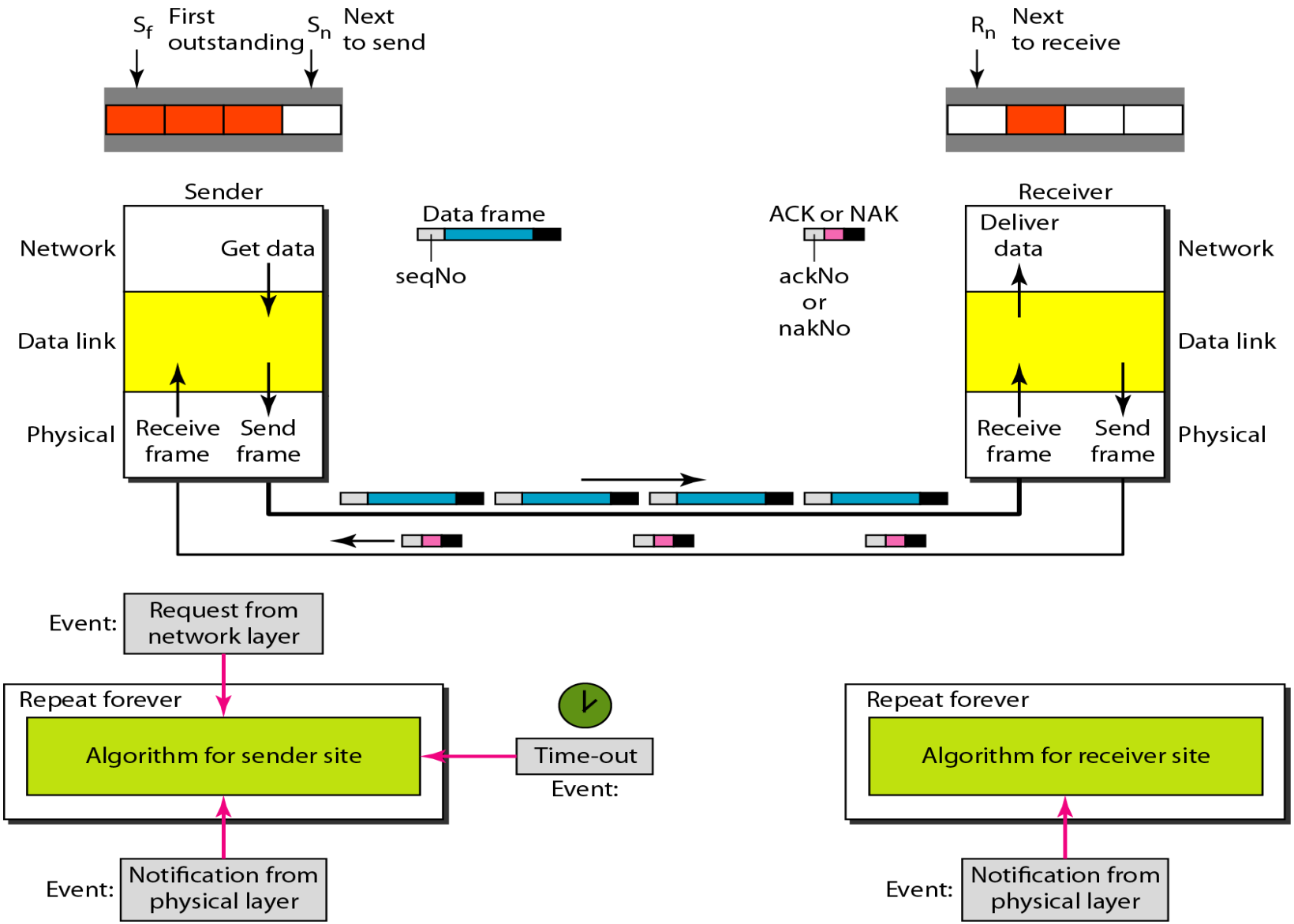
- Kept until there is set of in-order frames to be delivered to the network layer.
  - Because the size of the send window and receive window are same.

- Receive window for Selective Repeat ARQ



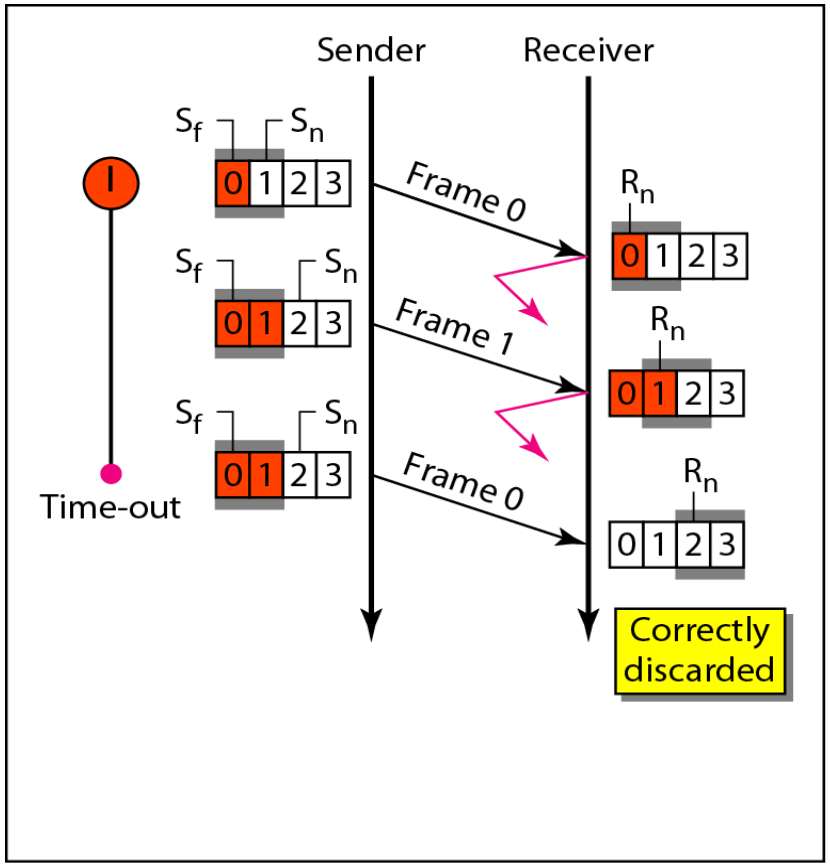
Selective Repeat ARQ

- Design of Selective Repeat ARQ

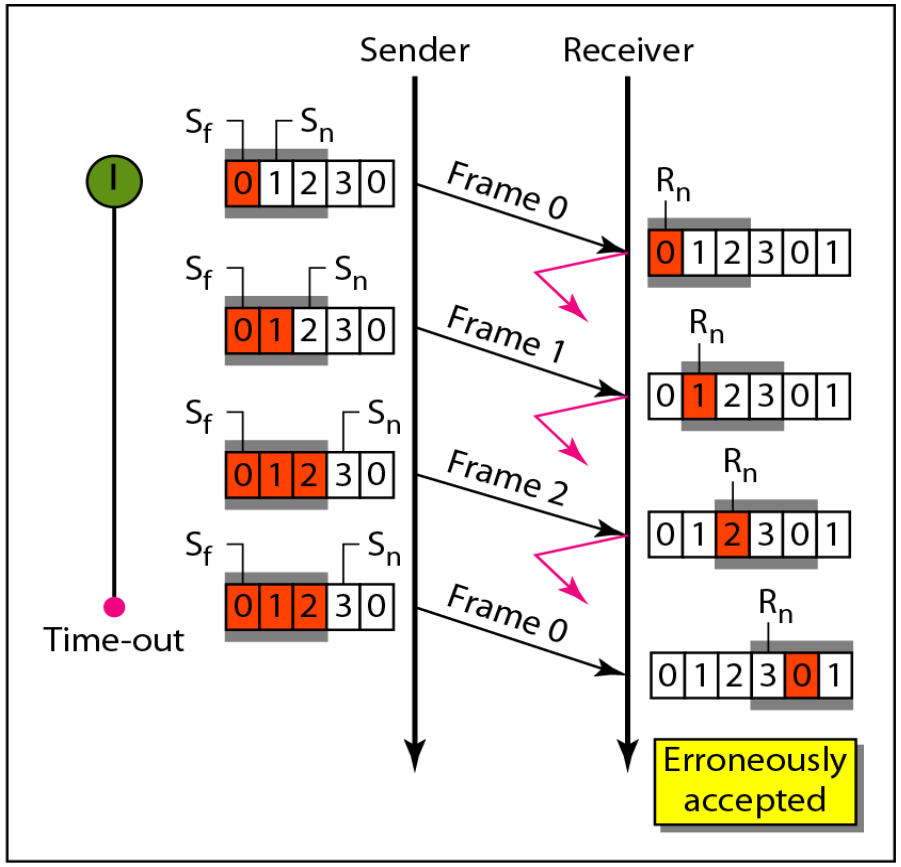


Selective Repeat ARQ

- Window size
  - Why size of the sender and receiver window must be at most half of 2^m.



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



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

Selective Repeat ARQ

- Sender side Selective repeat 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
```

Selective Repeat ARQ

- Sender side Selective repeat 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 side Selective repeat Algorithm

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

Selective Repeat ARQ

- Receiver side Selective repeat 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 side Selective repeat 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);
```

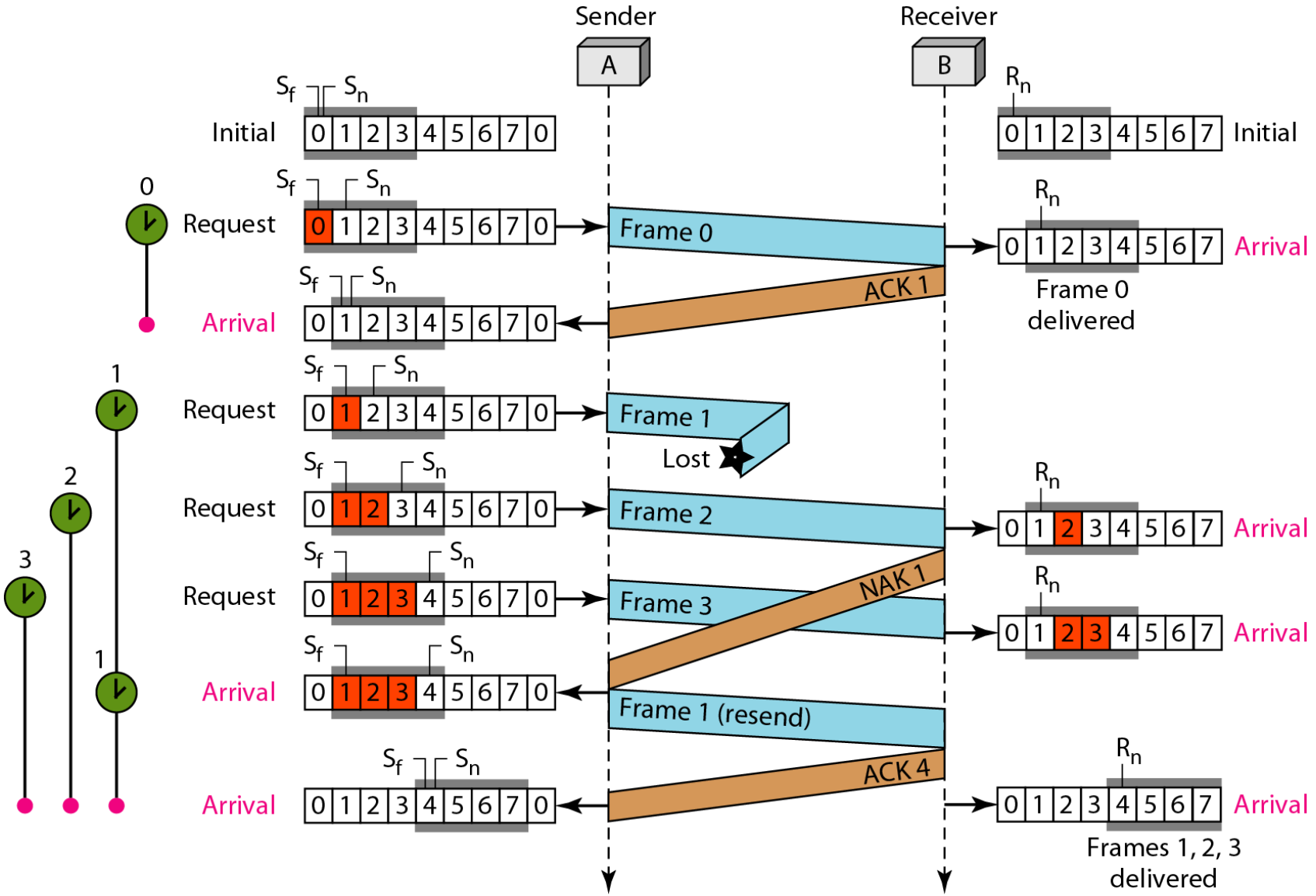
Selective Repeat ARQ

- Receiver side Selective repeat Algorithm

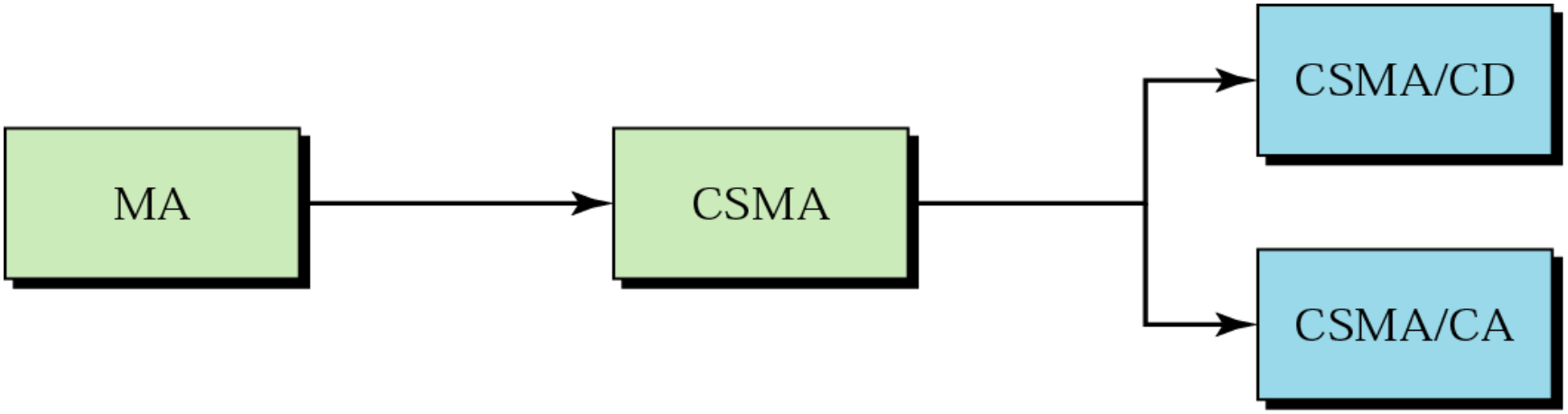
```
23        NakSent = true;
24        if ((seqNo in window)&&(!Marked(seqNo))
25        {
26         StoreFrame(seqNo)
27         Marked(seqNo)= true;
28         while(Marked(Rn))
29         {
30          DeliverData(Rn);
31          Purge(Rn);
32          Rn = Rn + 1;
33          AckNeeded = true;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

Selective Repeat ARQ

- Flow diagram for example



Multiple Access



Carrier Sense Multiple Access Protocol – CSMA

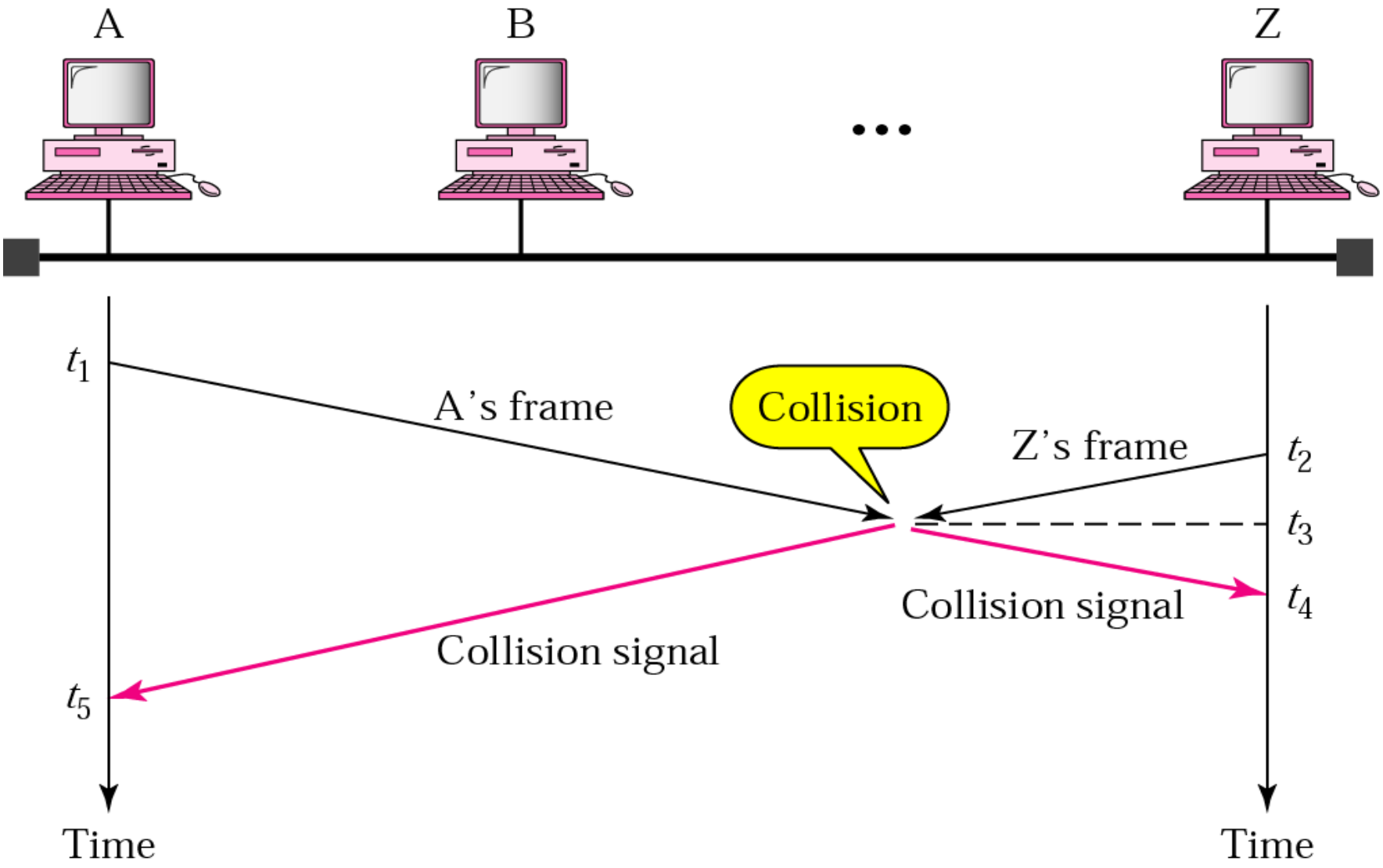
- CSMA requires that each station first listen to the medium (or check state of medium) before sending.
- CSMA based on the principle "Sense before transmit" or "Listen before talk".

Protocols in which stations listen for a carrier (i.e., a transmission channel) and act accordingly are called **Carrier Sense Protocol**.

- It can reduce possibility of collision, but it can not eliminate it.
- Possibility of collision still exist.
  - When station send frames, it still takes time for the first bit to reach very station and for every station to send it.
  - Station may sense the medium and find it ideal, the first bit send by another station has not yet been received.

Carrier Sense Multiple Access Protocol – CSMA

- Sense/time model of collision in CSMA

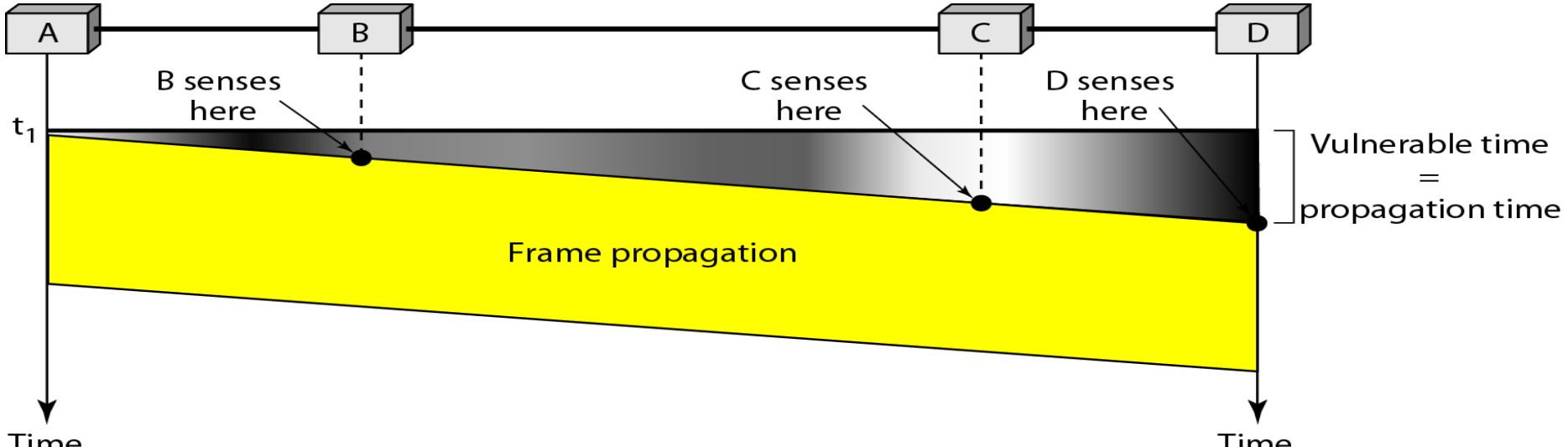


Carrier Sense Multiple Access Protocol – CSMA

- Propagation time T_p .

This is time needed for a signal to propagate from one end of medium to other.

- Vulnerable time:
  - It is a propagation time T_p
  - Time needed for a signal to propagate from one end to other. If other station can transmit during this time, a collision will happen.



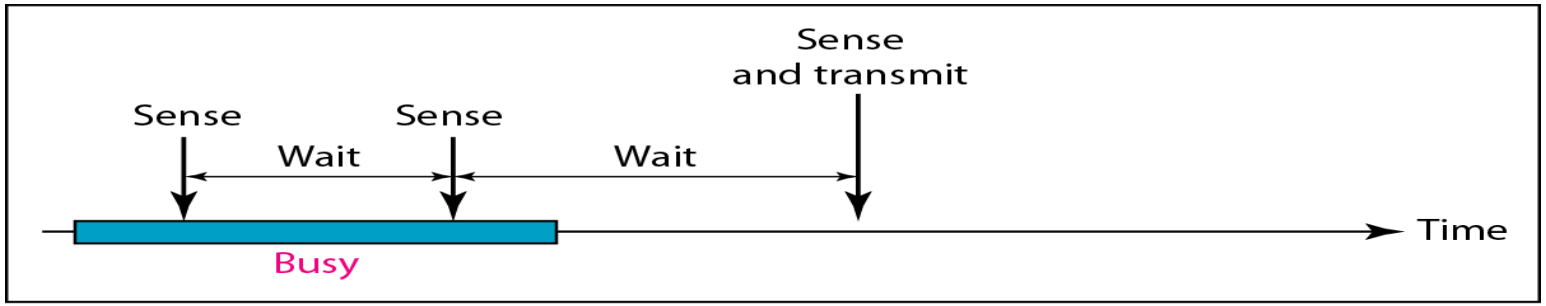
Carrier Sense Multiple Access Protocol – CSMA

- Persistence method
  - What should a station can do if channel is busy?
  - What should a station can do if channel is ideal?
  - Three methods are available to give answer of these question:
    - 1-persistant method
    - Non persistent method
    - P-persistent method

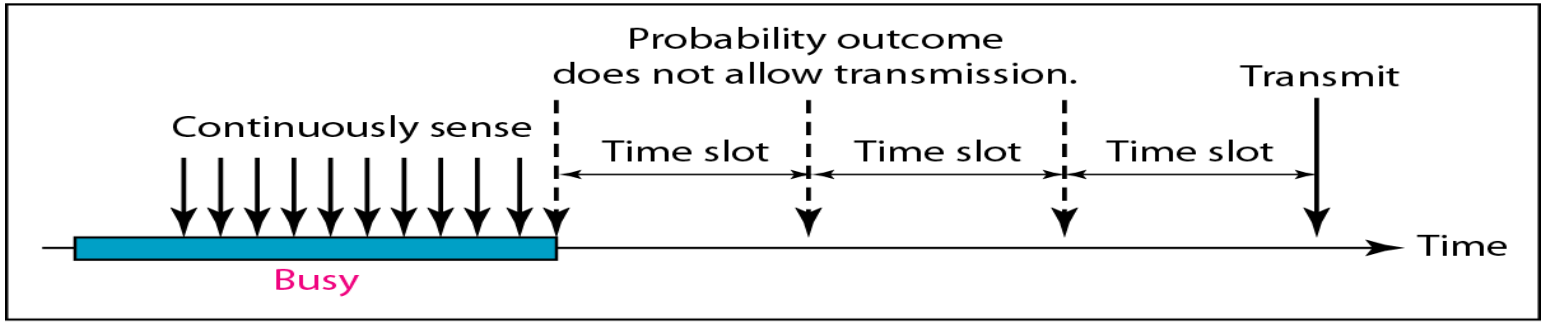
Carrier Sense Multiple Access Protocol – CSMA

- Behavior of three persistent method

a. 1-persistent



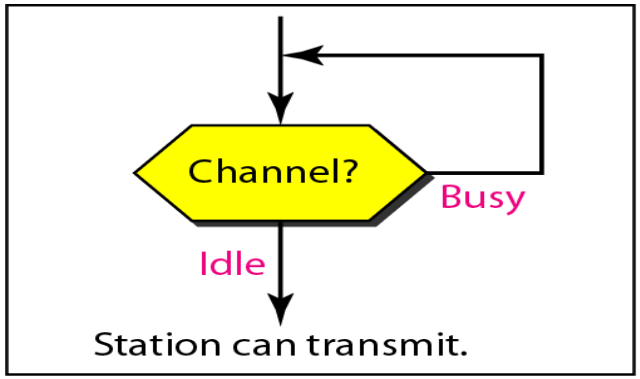
b. Nonpersistent



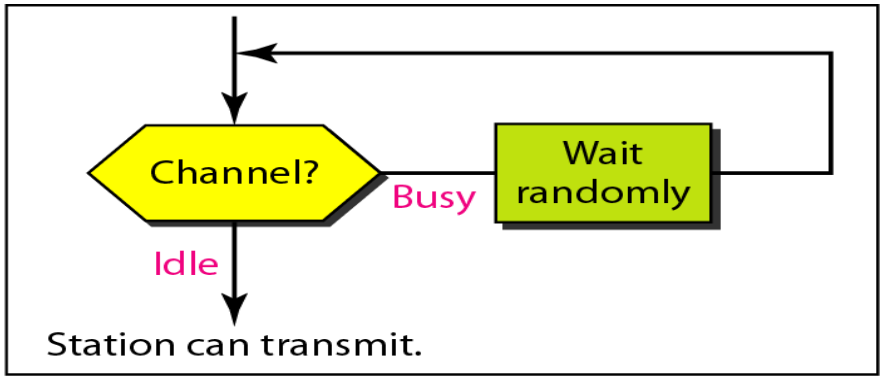
c. p-persistent

Carrier Sense Multiple Access Protocol – CSMA

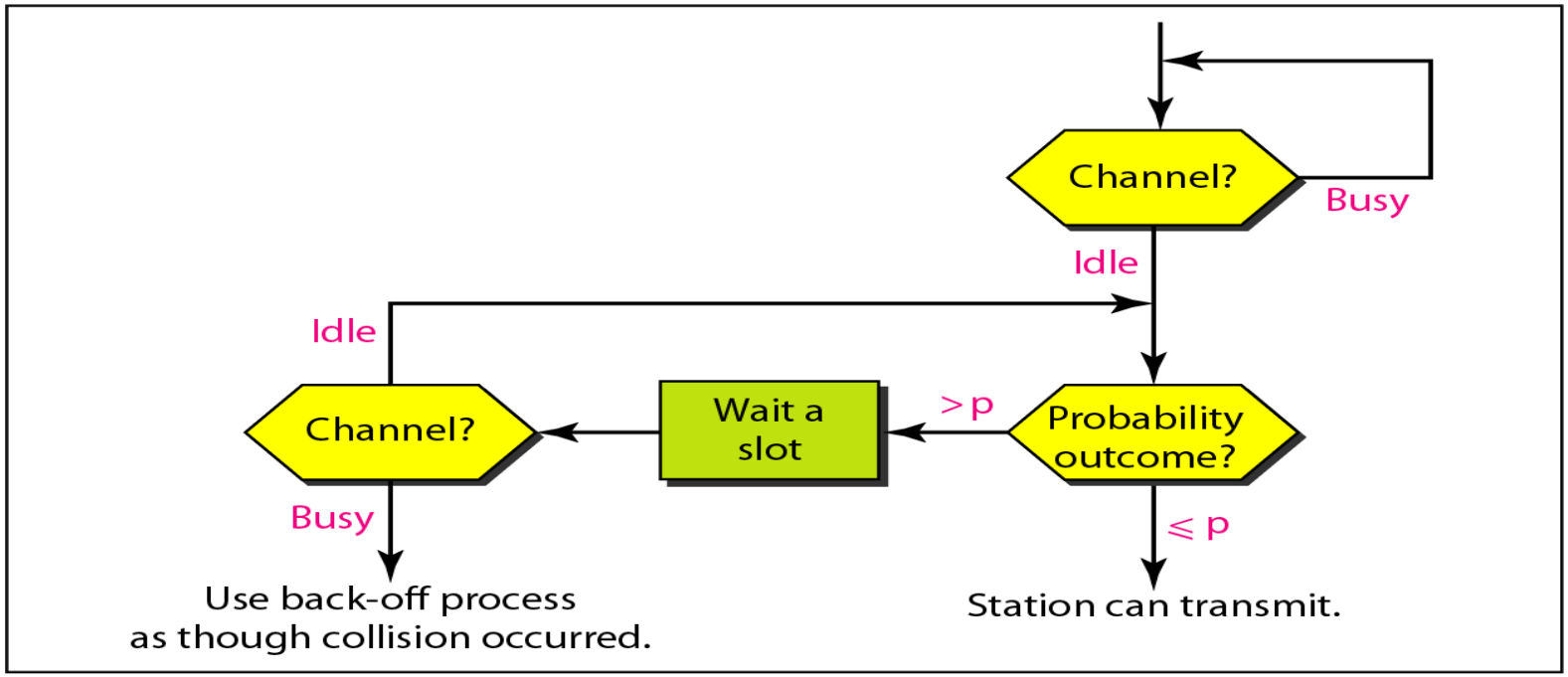
- Flow diagram of three persistent method



a. 1-persistent



b. Nonpersistent



c. p-persistent

Carrier Sense Multiple Access Protocol – CSMA

- 1- Persistent
  - If the station find line ideal it sends its frame immediately with probability 1.
  - This channel has highest chance of collision because
    - Two or more station may finds the line ideal and send their frames immediately.
- Non-persistent
  - If the line is ideal, it send immediately.
  - If line is busy, it wait a random amount of time and then sense the line again.
  - It reduces chance of collision because
    - It is unlikely that two or more station will wait the same amount of time and retry to send simultaneously.
  - It reduces the efficiency of network because
    - Medium remain ideal when there may be station with frame to send.
- P-persistent
  - It combines the advantage of two other method.
    - It reduces chance of collision and improve efficiency.