CST303: MODULE II

Module II (8 hours)

Data Link layer Design Issues – Flow Control and ARQ techniques. Data link Protocols – HDLC. DLL in Internet. MAC Sub layer – IEEE 802 FOR LANs & MANs, IEEE 802.3, 802.4, 802.5. Bridges - Switches – High Speed LANs - Gigabit Ethernet. Wireless LANs - 802.11 a/b/g/n, 802.15.PPP .

DATA LINK LAYER DESIGN ISSUES

- Uses services of physical layer to send & receive bits over communication channel.

1. Providing a well-defined **service** interface to the network layer.

2. Dealing with **transmission errors** (framing & error control)

3. Regulating the **flow of data** so that slow receivers are not swamped by fast senders.

- takes packets from network layers & encapsulates them into frames for transmission

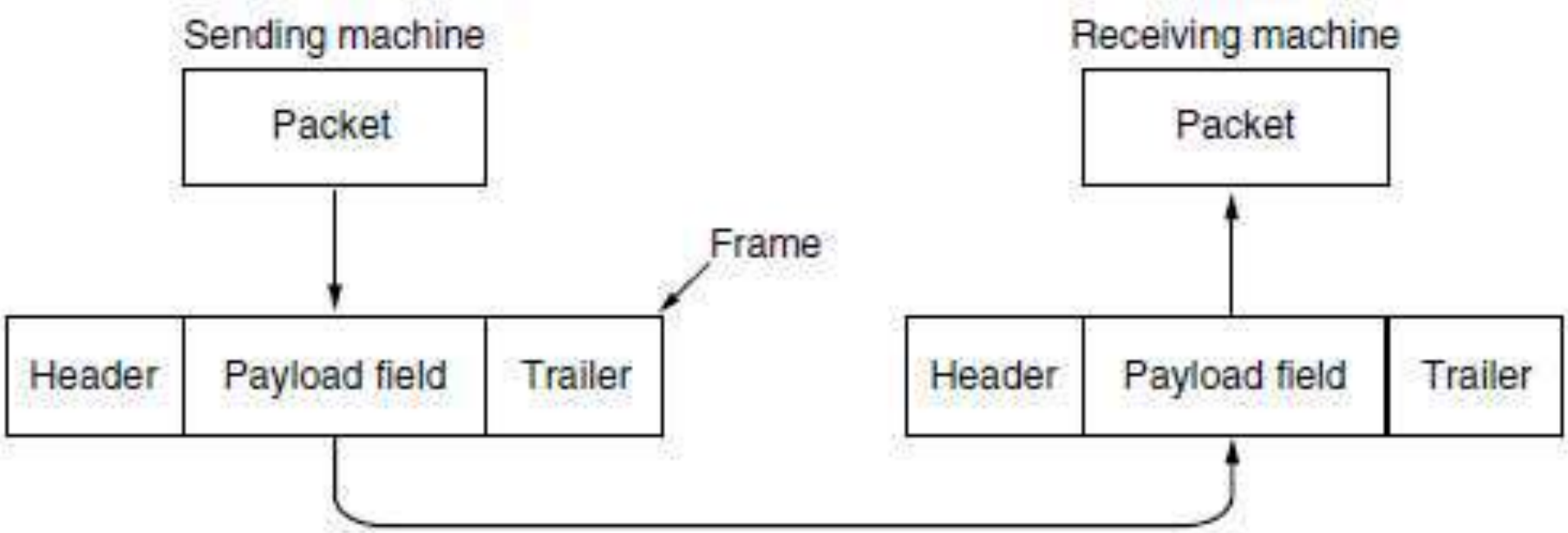


Figure 3-1. Relationship between packets and frames.

1. Services Provided to the Network Layer

- Principal Service → transferring data from network layer on the source machine to the network layer on the destination machine.
- Virtual Communication
- Actual Communication

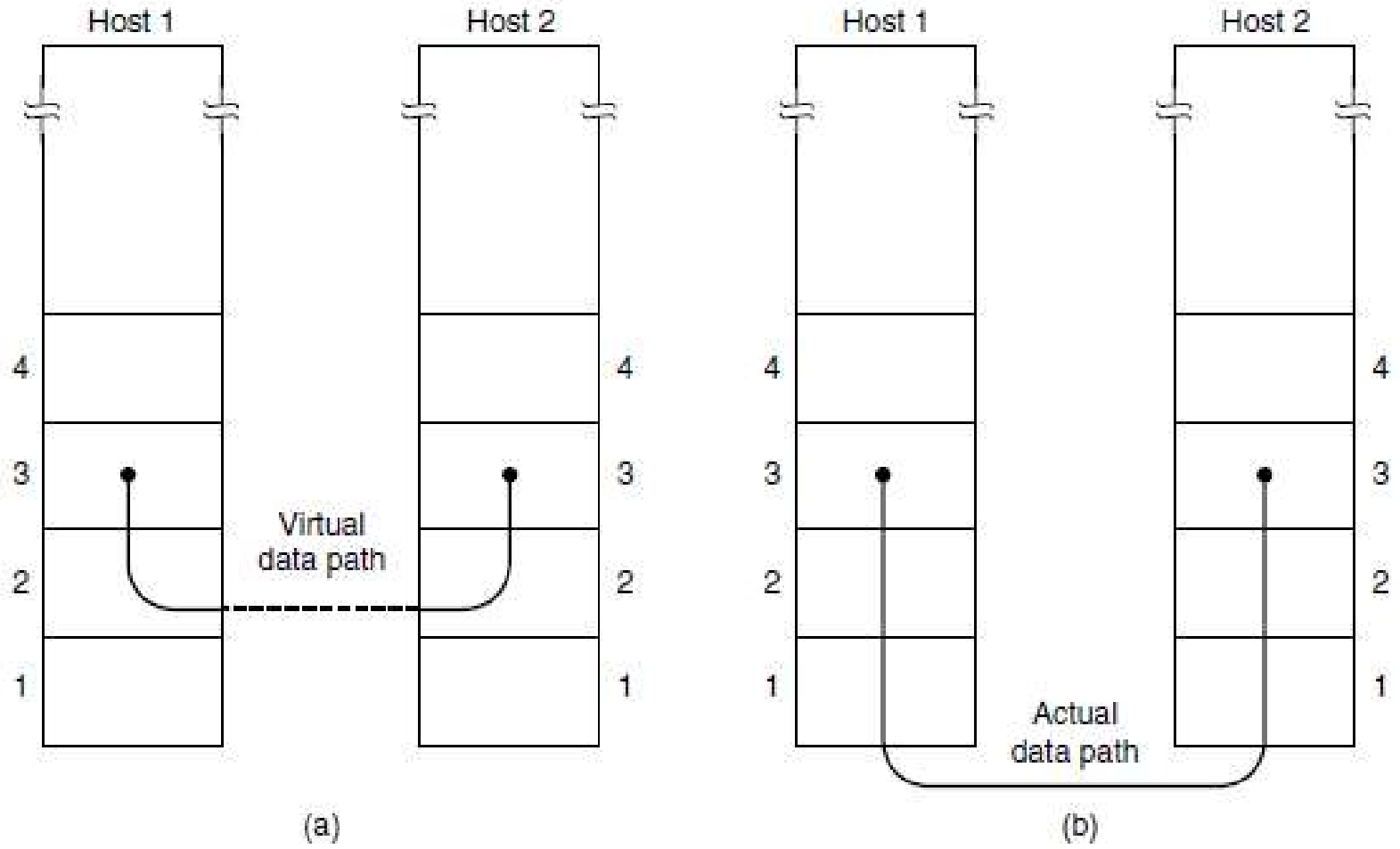


Figure 3-2. (a) Virtual communication. (b) Actual communication.

- 3 reasonable possibilities that we will consider in turn are:
  1. Unacknowledged connectionless service.
     - Eg: Ethernet
  2. Acknowledged connectionless service.
     - Eg: 802.11 WiFi
  3. Acknowledged connection-oriented service.
     - Eg: Satellite channel, long distance telephone circuit

Framing

- accept a raw bit stream and attempt to deliver it to the destination.
- break up the bit stream into discrete frames.
- compute a short token called a checksum for each frame.
- include the checksum in the frame when it is transmitted.
- destination → checksum is computed
- If the newly computed checksum is different from the one contained in the frame, the data link layer knows that an error has occurred.

- Breaking up the bit stream into frames is more difficult than it at first appears.

- 4 methods:

1. Character Count
2. Flag bytes with byte stuffing
3. Starting & ending flags, with bit stuffing
4. Physical layer coding violations

1. Character Count

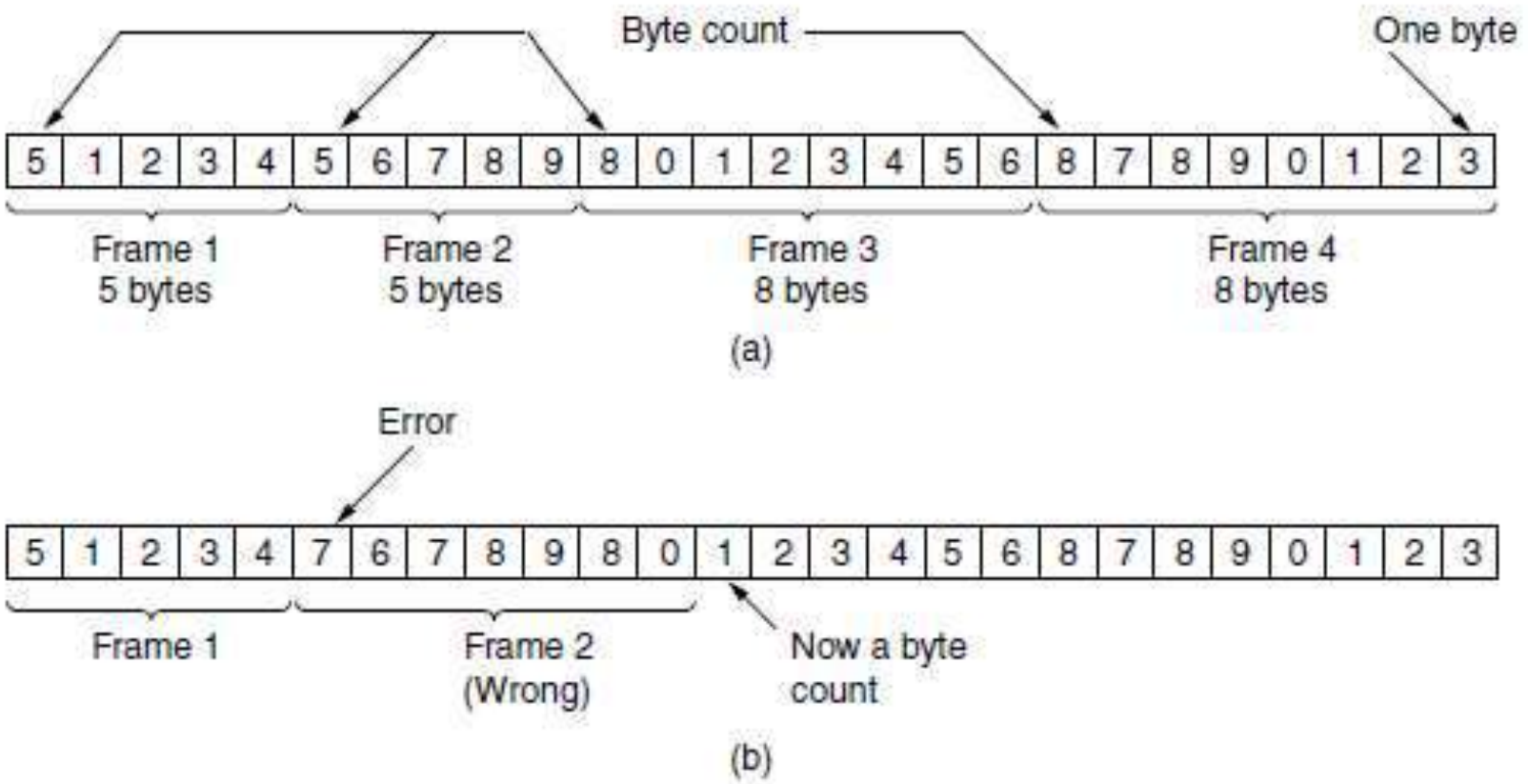
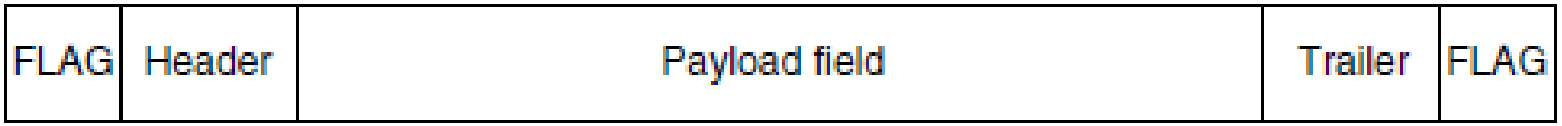


Figure 3-3. A byte stream. (a) Without errors. (b) With one error.

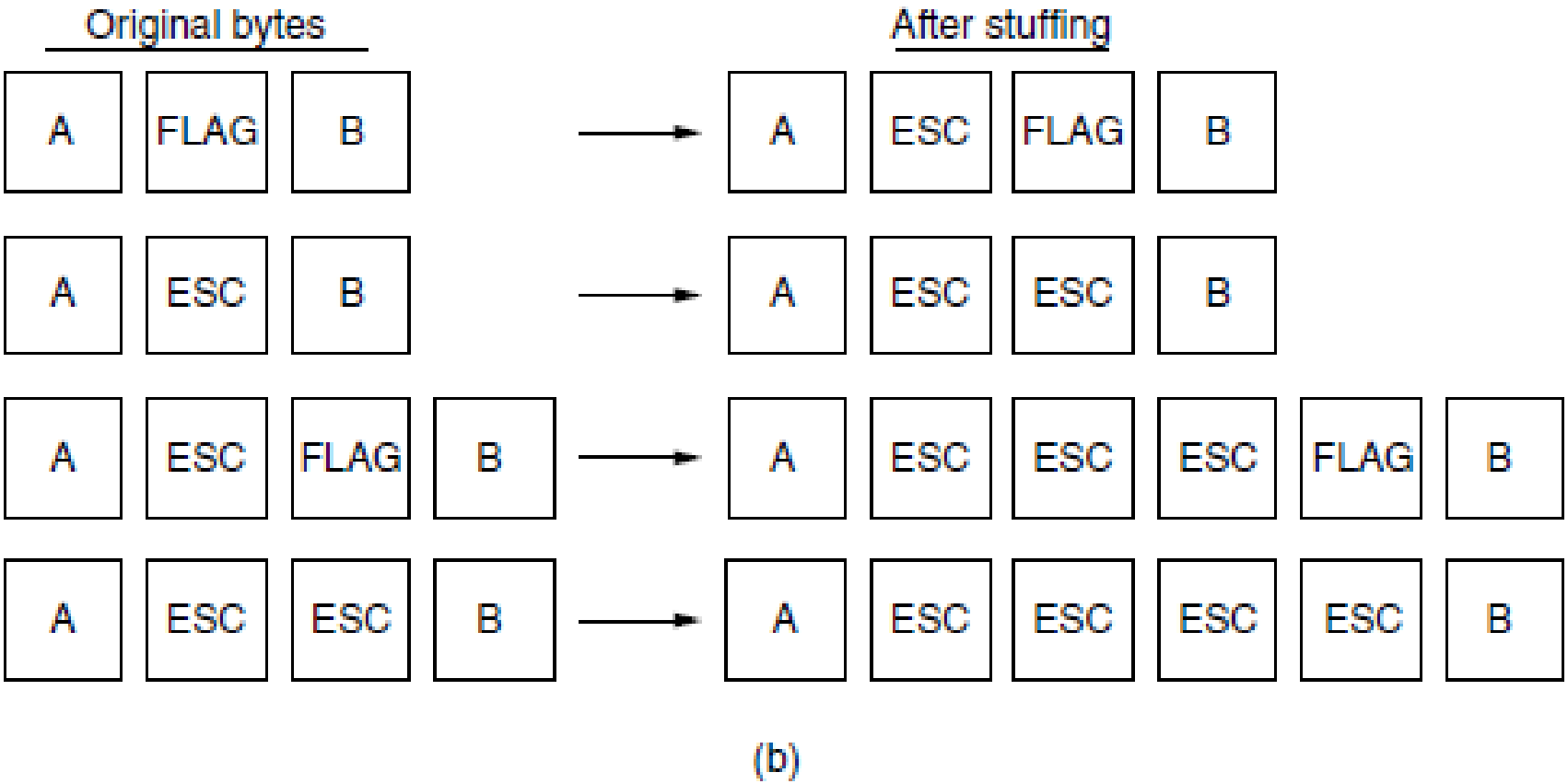
2. Flag bytes with byte stuffing

- each frame start & end with special bytes
- using same byte called, flag byte.



(a)

- flag bit same as data.
  - Resolved by inserting ESC
  - Byte stuffing or character stuffing


Original bytes
After stuffing
A
FLAG
B
A
ESC
FLAG
B
A
ESC
B
A
ESC
ESC
B
A
ESC
FLAG
B
A
ESC
ESC
ESC
FLAG
B
A
ESC
ESC
B
A
ESC
ESC
ESC
ESC
B

(b)

3. Starting and ending flags, with bit stuffing

- done at bit level, HDLC protocol
- each frame begins and ends with a special bit pattern, 01111110 or 0x7E in hexadecimal.
- whenever the sender's data link layer encounters five consecutive 1s in the data, it automatically stuffs a 0 bit into the outgoing bit stream.
- When the receiver sees five consecutive incoming 1 bits, followed by a 0 bit,it automatically destuffs (i.e., deletes) the 0 bit

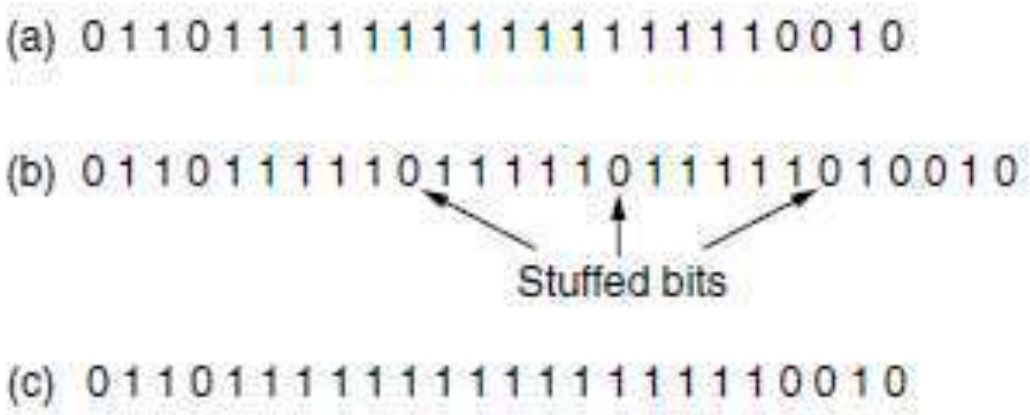


Figure 3-5. Bit stuffing. (a) The original data. (b) The data as they appear on the line. (c) The data as they are stored in the receiver's memory after destuffing.

4. Physical Layer coding violation

- encoding of bits as signals often includes redundancy to help the receiver.
- use some reserved signals to indicate the start and end of frames.
- because they are reserved signals, it is easy to find the start and end of frames and there is no need to stuff the data.
- A common pattern used for Ethernet and 802.11 is to have a frame begin with a well-defined pattern called a **preamble.**

ERROR CONTROL

- To make sure all frames are eventually delivered to the network layer at the destination in the proper order

- reliable delivery → feedback
- positive ack's & negative ack's

- noise burst → frame vanish completely
- frame lost forever → malfunctioning hardware
  - introduce timers
  - if frame/ack is lost timer will go off→ alerting sender of potential problem.

- transmit frame again
- multiple frames may be received
- Use of sequence numbers to outgoing frames

- Managing the timers & sequence numbers ensures that each frame is ultimately passed to the network layer at the destination exactly once.

- Error control is both error detection & error correction.

- Any time an error is detected in an exchange, specific frames are retransmitted.

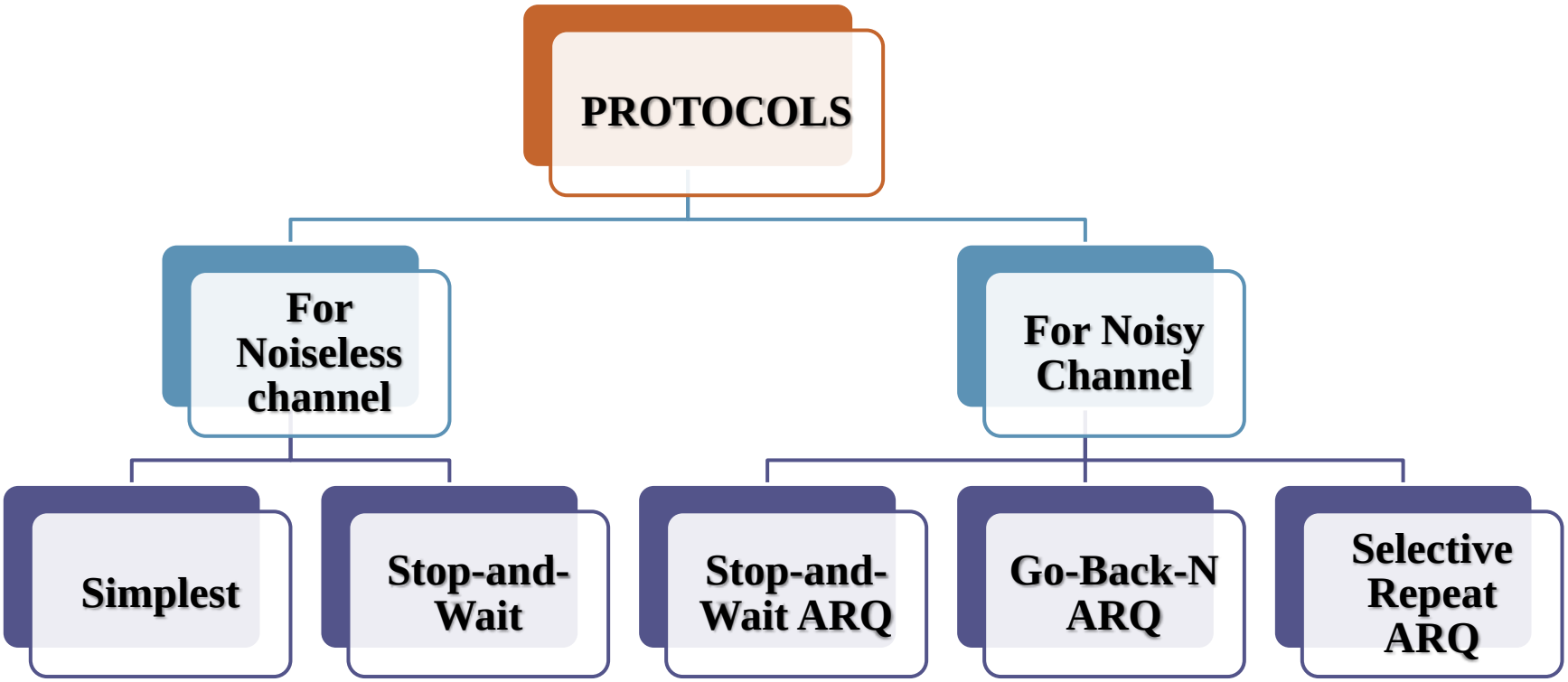
- **AUTOMATIC REPEAT REQUEST (ARQ)**

FLOW CONTROL

- Sender is running on a fast computer & receiver on a slow computer.

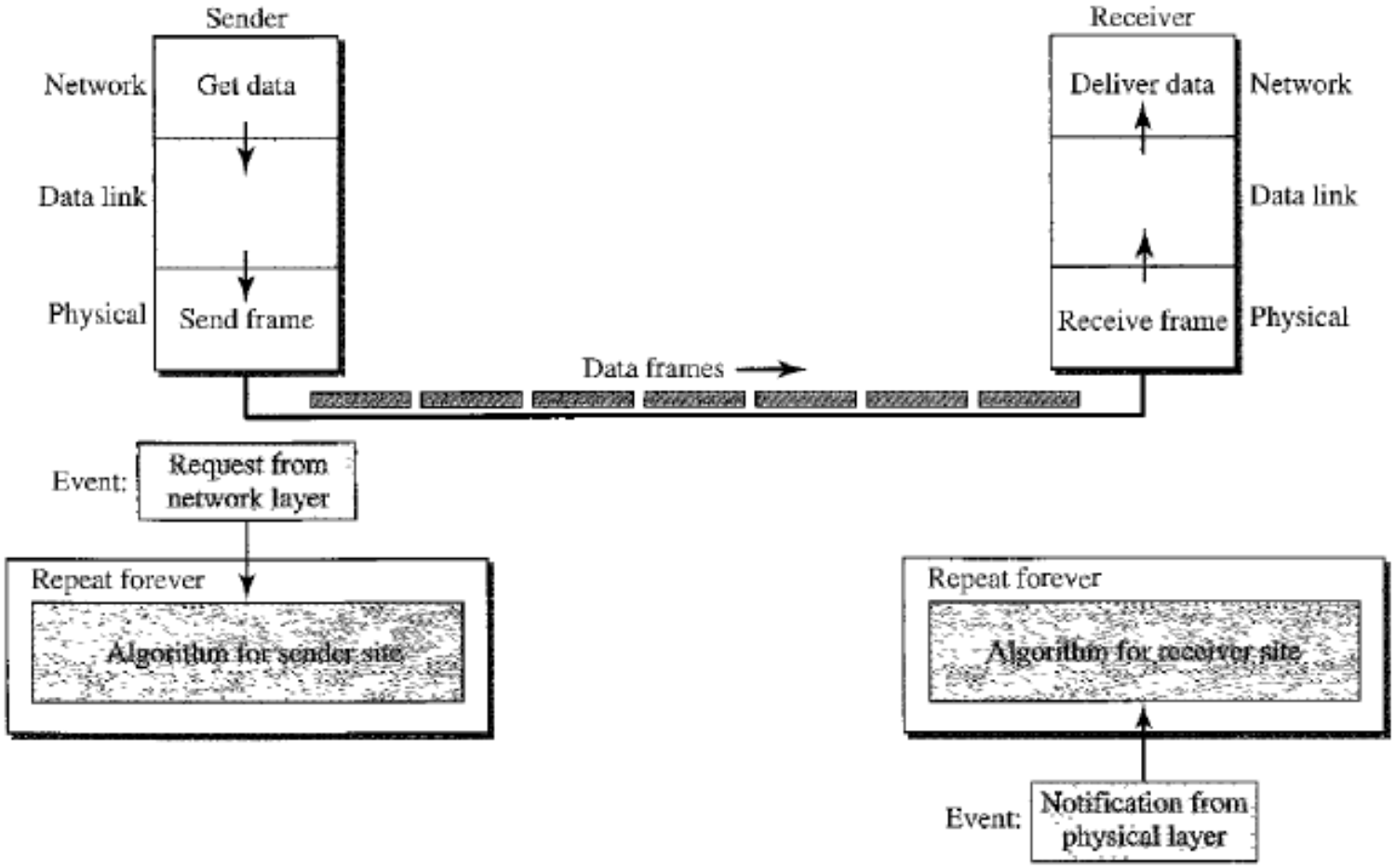
- If error free; some frames will be lost. Prevented by 2 methods:
  - **feedback-based flow control -** receiver sends back information to the sender giving it permission to send more data, or at least telling the sender how the receiver is doing.

  - **rate-based flow control -** protocol has a built-in mechanism that limits the rate at which senders may transmit data, without using feedback from the receiver.


PROTOCOLS
For Noiseless channel
For Noisy Channel
Simplest
Stop-and-Wait
Stop-and-Wait ARQ
Go-Back-N ARQ
Selective Repeat ARQ

NOISELESS CHANNELS

- Simplest Protocol
  - No flow or error control
  - Unidirectional protocol frames are travelling in only one direction
  - Receiver can immediately handle any frame it receives with a processing time that is small enough to negligible.
  - Data link layer at the receiver site receives the frame from its physical layer & delivers the data to its network layer after extracting data from frame.

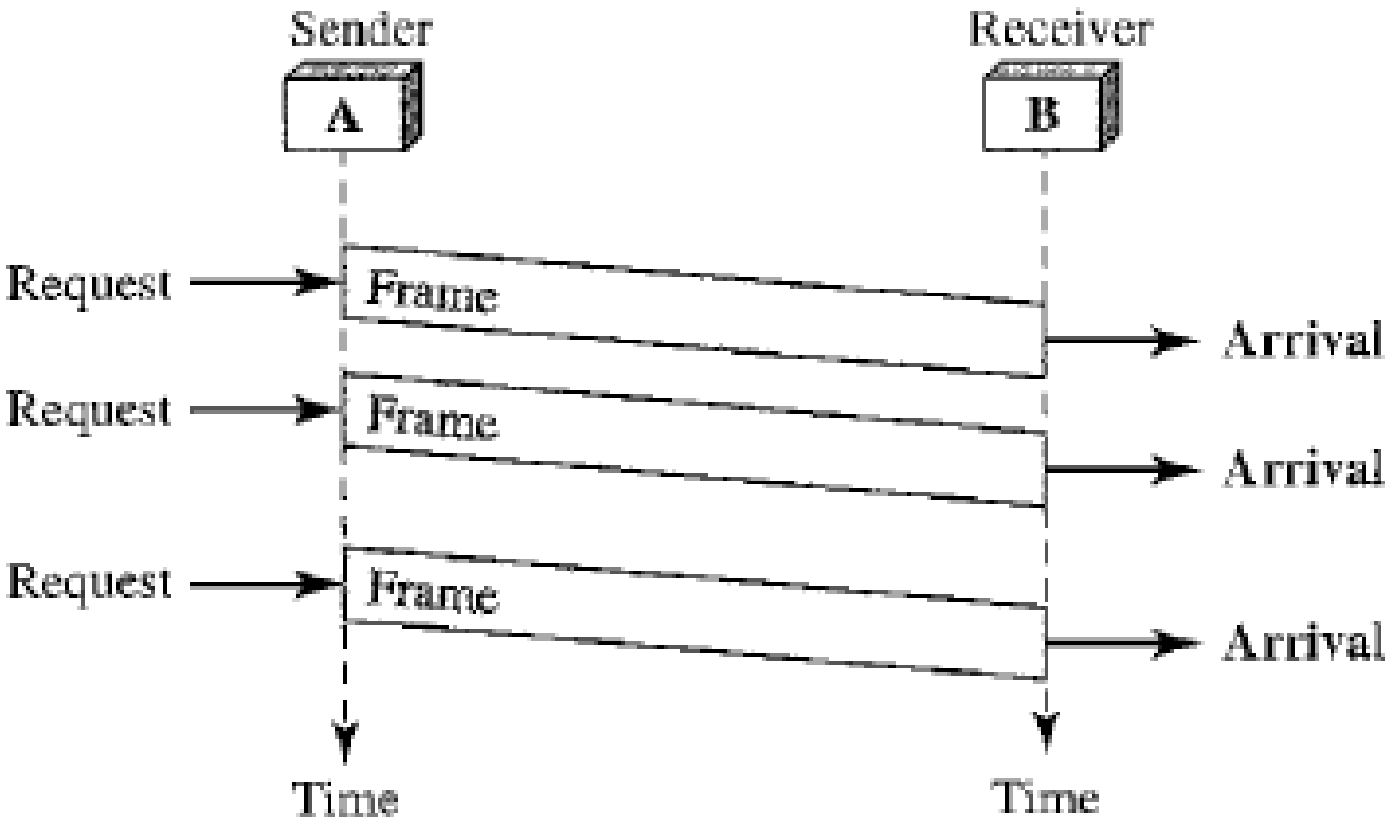

Sender
Receiver
Network
Get data
Deliver data
Network
Data link
Data link
Physical
Send frame
Receive frame
Physical
Data frames
Event:
Request from network layer
Repeat forever
Algorithm for sender site
Repeat forever
Algorithm for receiver site
Event:
Notification from physical layer

Algorithm 11.1 *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  }
```

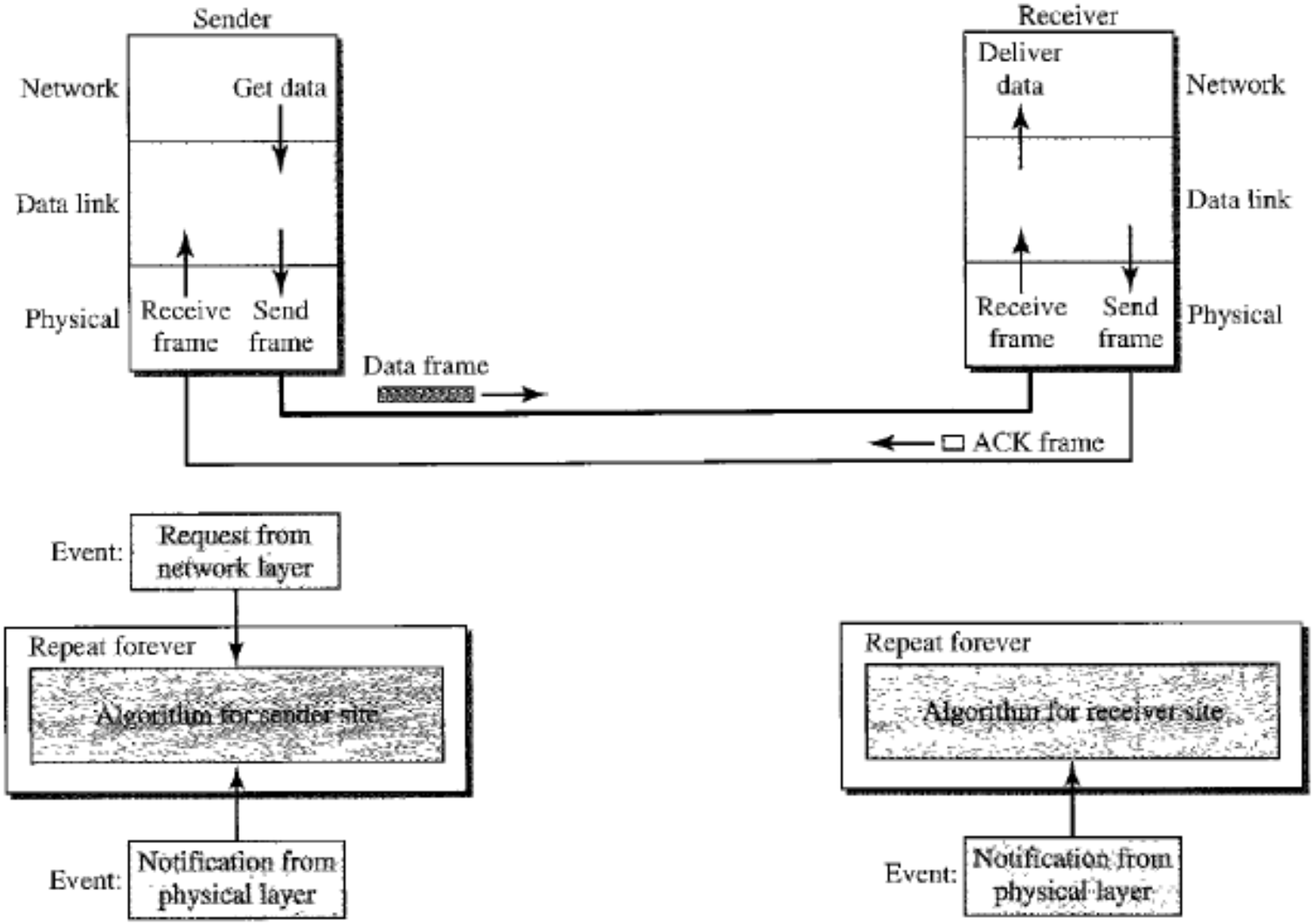
Algorithm 11.2 *Receiver-site algorithm for the simplest protocol*

```
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 }
```


Sender
Receiver
A
B
Request
Frame
Arrival
Request
Frame
Arrival
Request
Frame
Arrival
Time
Time

STOP-and-WAIT PROTOCOL

- When data arrives from many sources, receiver does not have enough storage.
- Either discarding frames or denial of service
- Need to tell sender to slow down→ feedback
- Stop-and-Wait Protocol, sender sends one frame, stops until it receives confirmation from receiver & the sends next frame.
- Add flow control to previous protocol

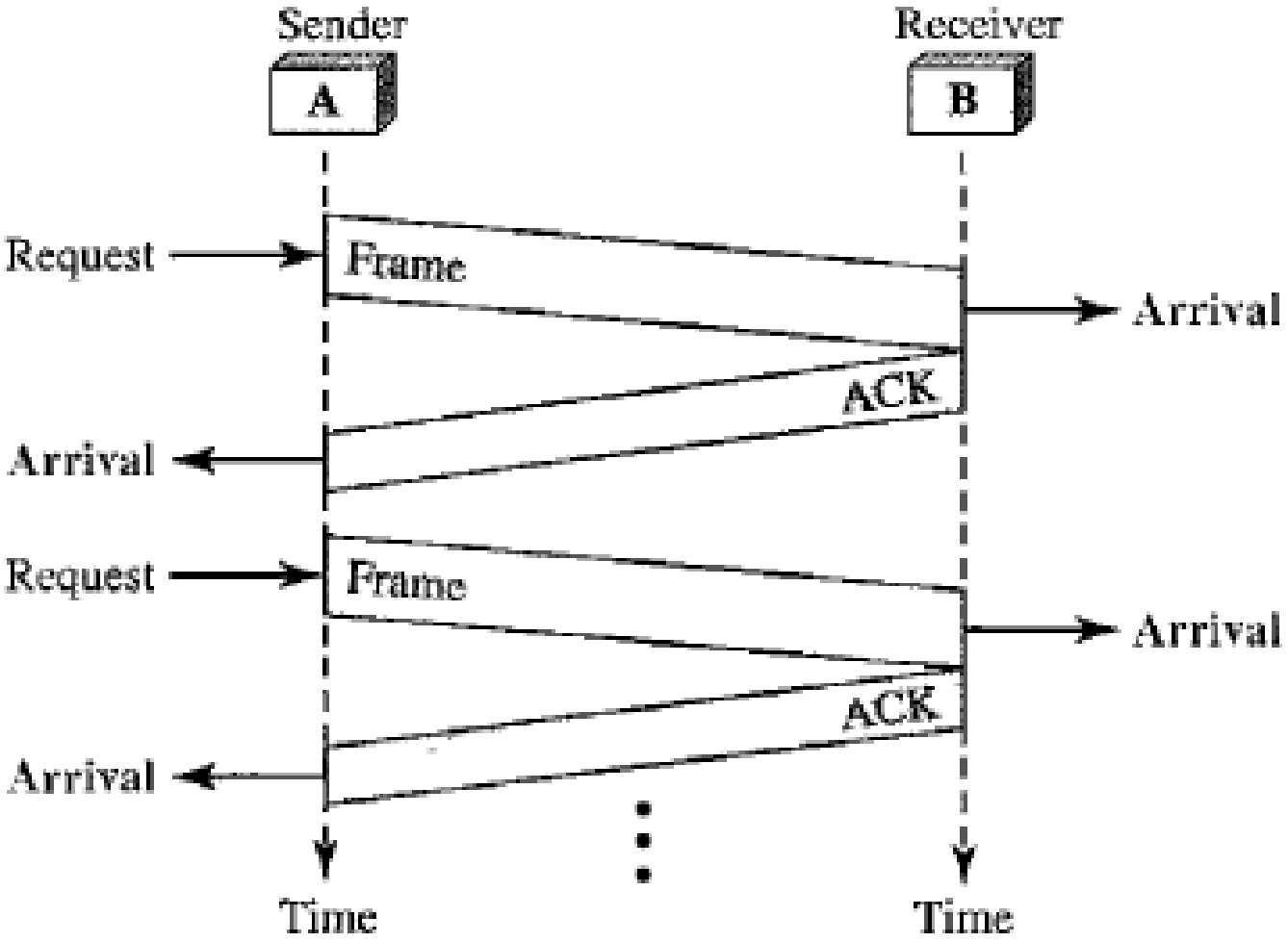

Sender
Receiver
Network
Get data
Deliver data
Network
Data link
Data link
Physical
Receive frame
Send frame
Receive frame
Send frame
Physical
Data frame
ACK frame
Event:
Request from network layer
Repeat forever
Algorithm for sender site
Event:
Notification from physical layer
Repeat forever
Algorithm for receiver site
Event:
Notification from physical layer

Algorithm 11.3 *Sender-site 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 }
```

Algorithm 11.4 *Receiver-site 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  }
```


Sender
A
Receiver
B
Request
Frame
Arrival
ACK
Arrival
Request
Frame
Arrival
ACK
Arrival
Time
Time

NOISY : STOP-AND-WAIT AUTOMATIC REPEAT REQUEST

- adds a simple error control mechanism to the Stop-and –wait protocol.
- to detect & correct corrupted frames, need to add redundancy bits to our data frames.
- lost frames are difficult to handle than corrupted frames
- received frames → Correct one, or a duplicate, or a frame out of order.

- Solution : Number the frames
  - When receiver receives a frame that is out of order → frames was lost/duplicated

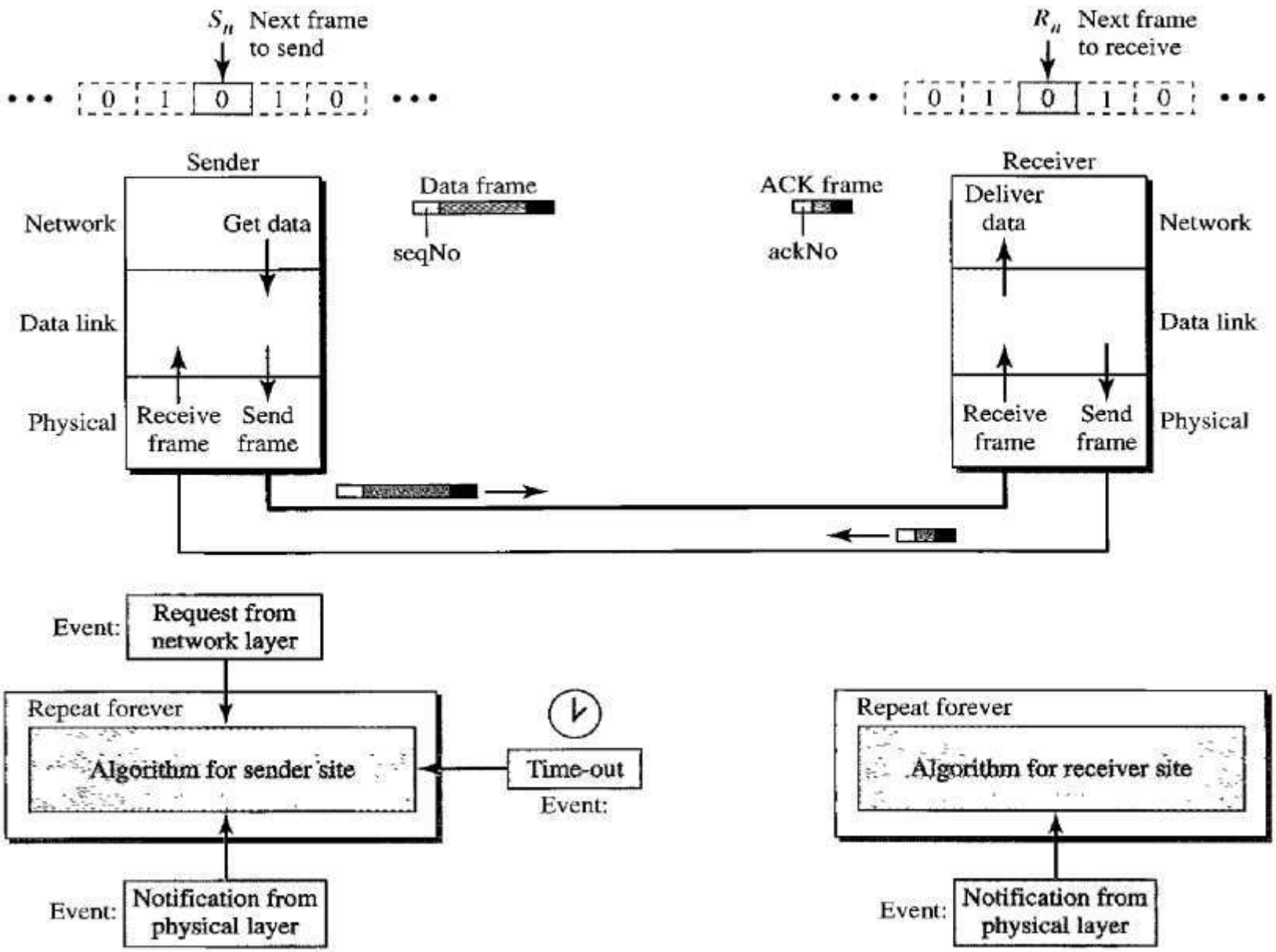
- If sender has to resend, a error/corruption
  - sender has to keep a copy of send frames
  - timer starts
  - if there is no ACK for sent frame, frame is resend.

- Error correction in Stop-and-wait ARQ is by keeping a copy of the sent frame & retransmitting the frame when the timer expires.

- Sequence Numbers
  - A field is added to the data frame to hold the sequence number of that frame
  - Sender has the frame numbered x, 3 things can happen:

1. frame → receiver; receiver ack's to sender; sender sends frame x+1
2. frame → receiver; receiver ack's to sender, but ack is lost; sender resends frame x after time out; frame x is duplicate.
3. frame never arrives at receiver, sender resends the frame x after the time out

- Acknowledgement Numbers
  - Ack numbers always announce the sequence number of the next frame expected by the receiver.
  - if frame 0 has arrived safe & sound, receiver sends an ack frame with ack 1, meaning frame 1 is expected next.
  - if frame 1 has arrived, receiver sends an ack frame with ack 0.

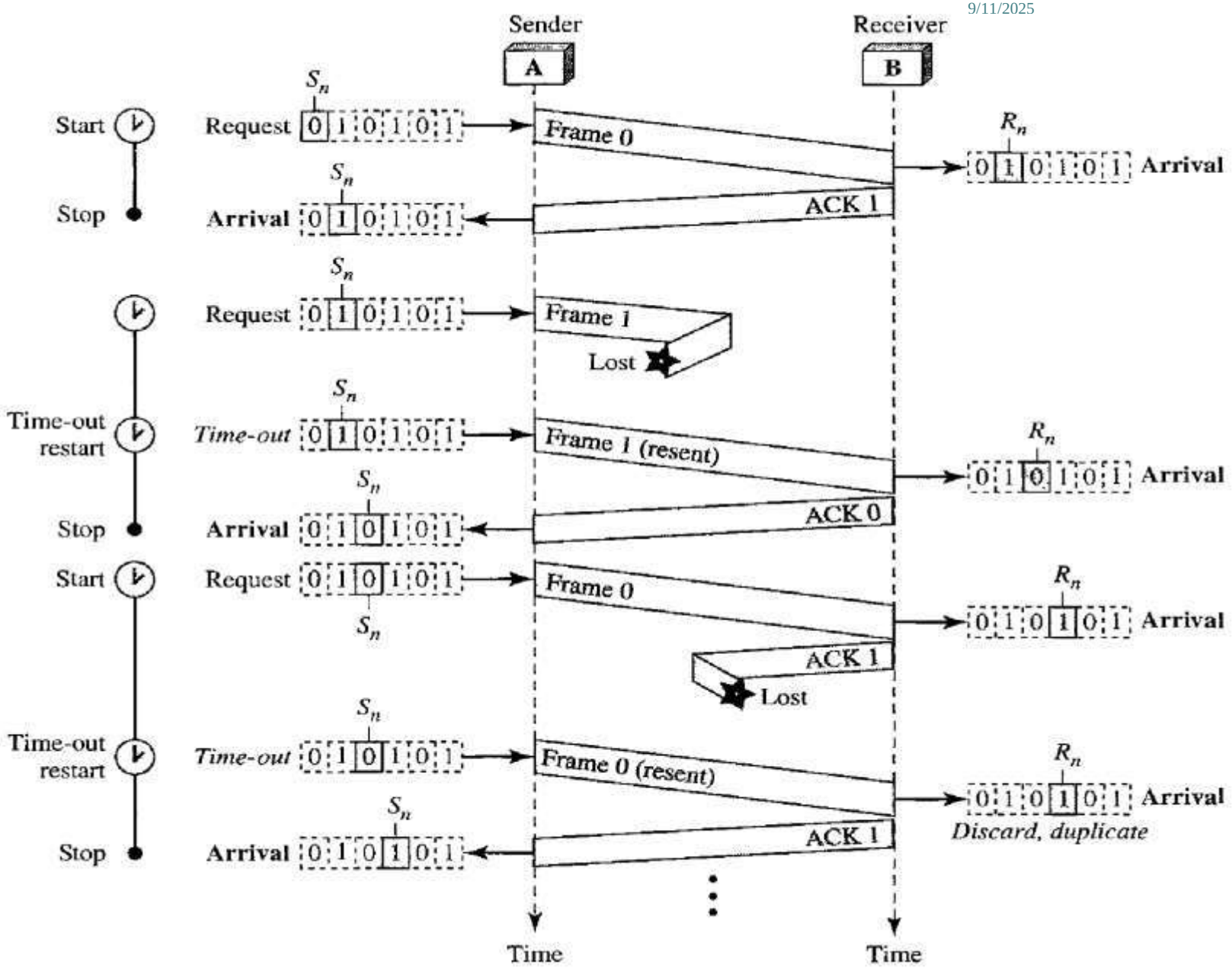

S_n Next frame to send
0 1 0 1 0
R_n Next frame to receive
0 1 0 1 0
Sender
Receiver
Data frame
seqNo
ACK frame
ackNo
Network
Get data
Deliver data
Network
Data link
Data link
Physical
Receive frame
Send frame
Receive frame
Send frame
Physical
Event:
Request from network layer
Repeat forever
Algorithm for sender site
Time-out
Event:
Event:
Notification from physical layer
Repeat forever
Algorithm for receiver site
Event:
Notification from physical layer

```
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
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  }
```

1. Rn = 0; //frame 0 expected to arrive first
2. While (true)
3. {
   WaitforEvent(); //sleep untill an event occurs

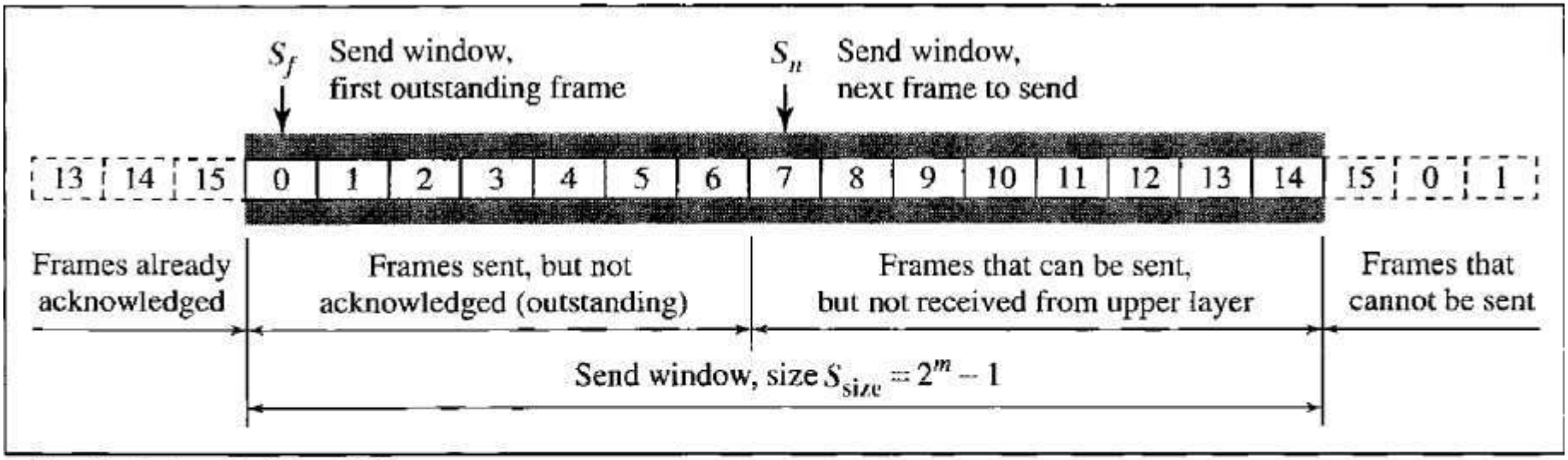
```
 5   if(Event(ArrivalNotification))  //Data frame arrives
 6   {
 7      ReceiveFrame();
 8      if(corrupted(frame));
 9         sleep();
10      if(seqNo == Rn)              //Valid data frame
11      {
12       ExtractData();
13        DeliverData();             //Deliver data
14        Rn = Rn + 1;
15      }
16       SendFrame(Rn);             //Send an ACK
17   }
18 }
```


Sender
Receiver
A
B
Start
Stop
Request
Arrival
Frame 0
ACK 1
Frame 1
Lost
Time-out
restart
Time-out
Frame 1 (resent)
ACK 0
Frame 0
ACK 1
Lost
Frame 0 (resent)
ACK 1
Discard, duplicate
Time
Time

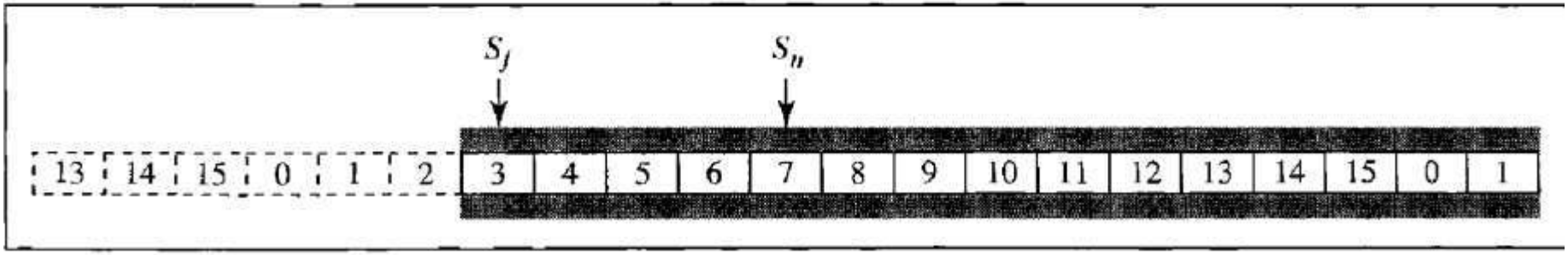
Go-Back-N Automatic Repeat Request

- To improve efficiency of transmission, multiple frames must be in transition, while waiting for an ack.
- Send several frames before receiving ack's; keep a copy of these frames, until ack arrive
- Sequence numbers → if header allows m bits, range is 2^m-1
- m is 4, 0 to 15 ie 0,1,2,3,…15,0,1,2,,…15,0,,,

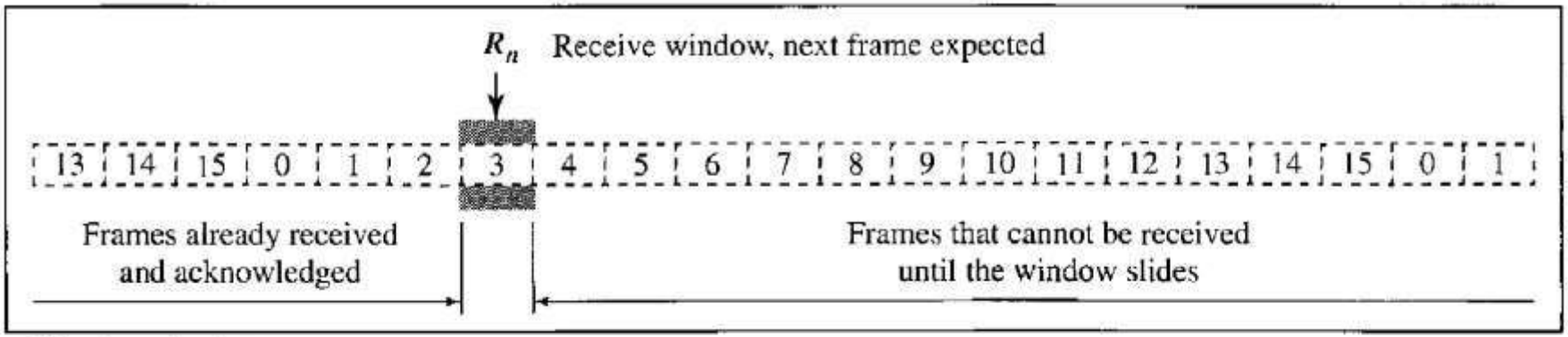
- Sliding Window
  - send sliding window
  - receiver sliding window
  - imaginary box covering sequence numbers of data frames which can be in transit



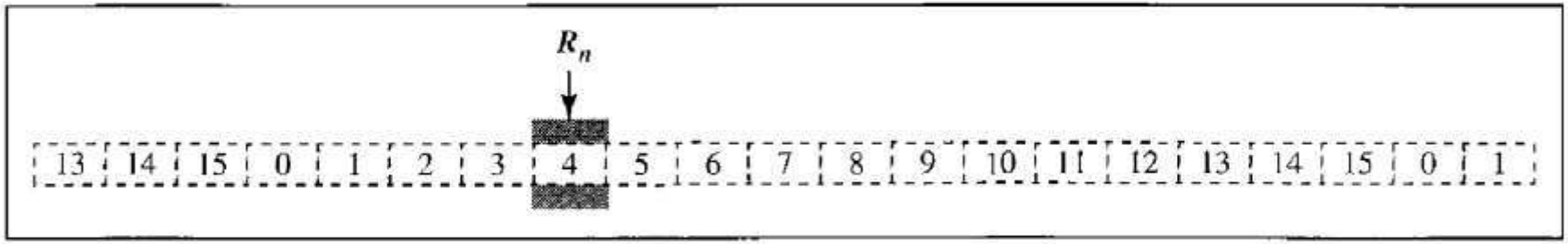
a. Send window before sliding



b. Send window after sliding

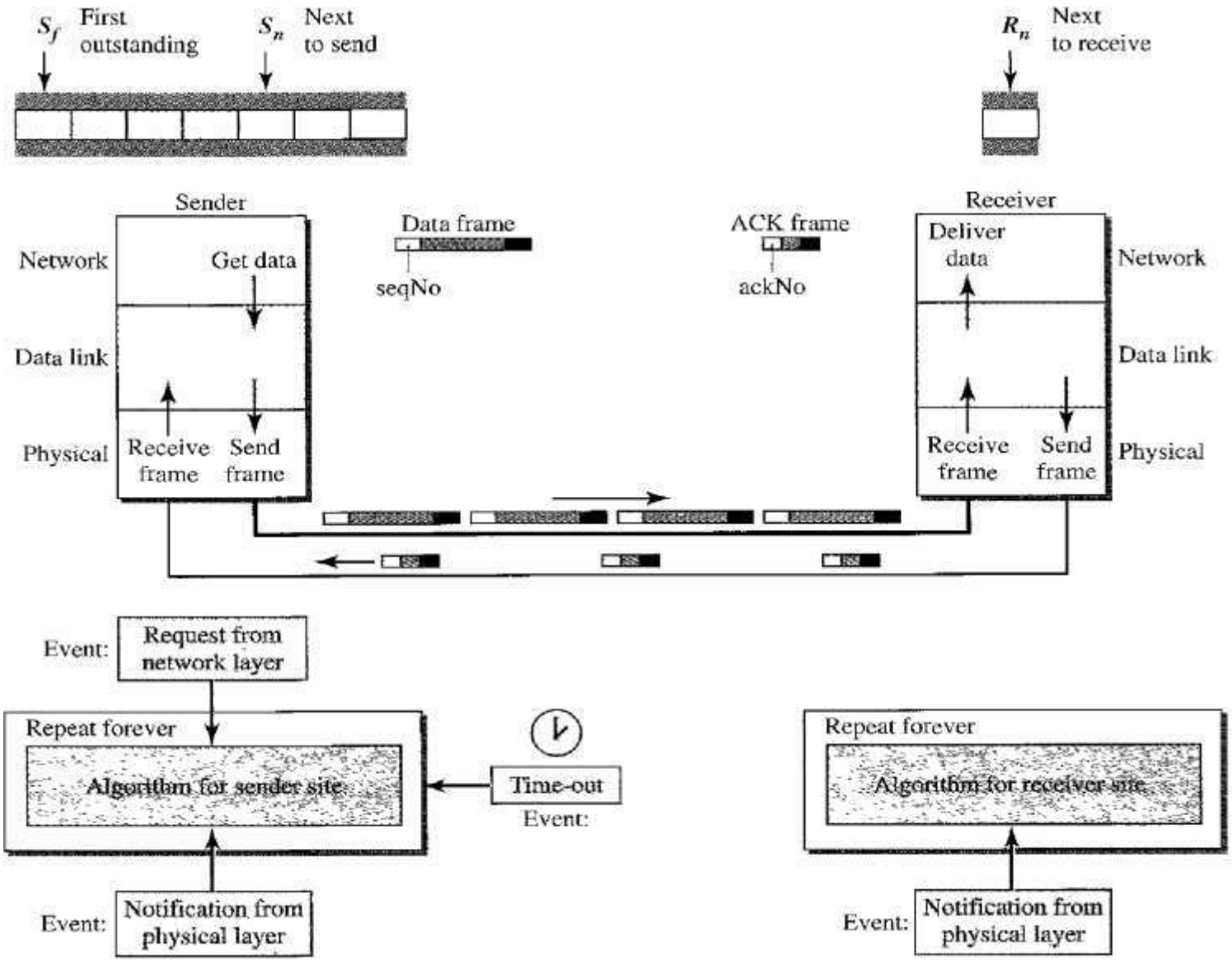


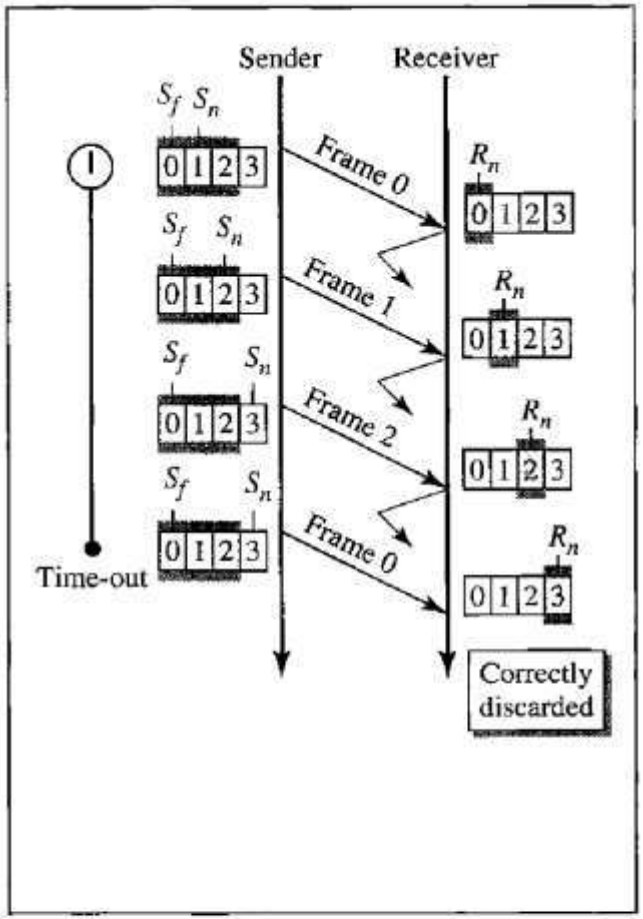
a. Receive window



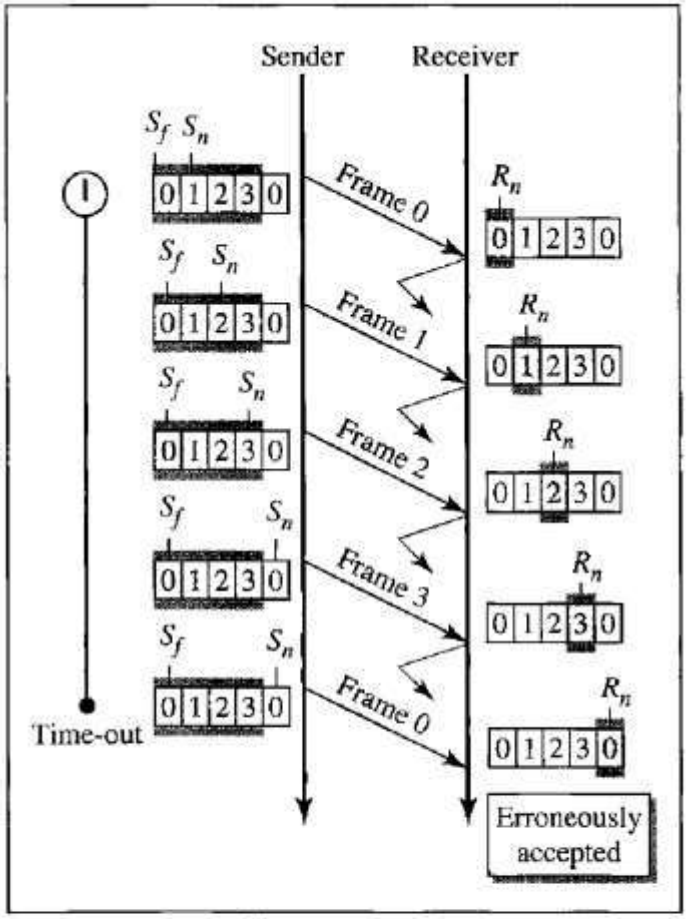
b. Window after sliding

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.


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



a. Window size $< 2^m$



b. Window size $= 2^m$

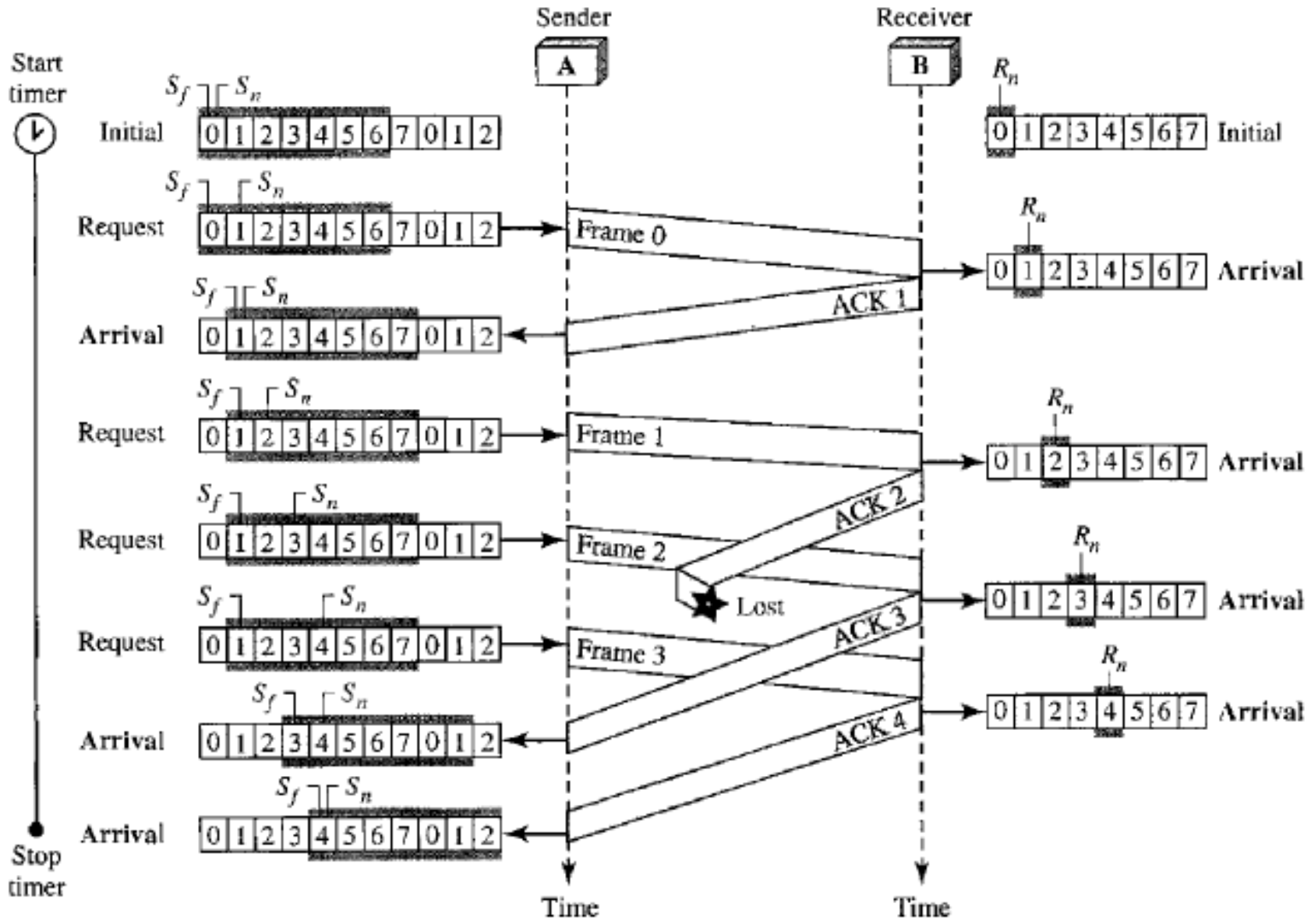
In Go-Back-*N* ARQ, the size of the send window must be less than 2^m; the size of the receiver window is always 1.

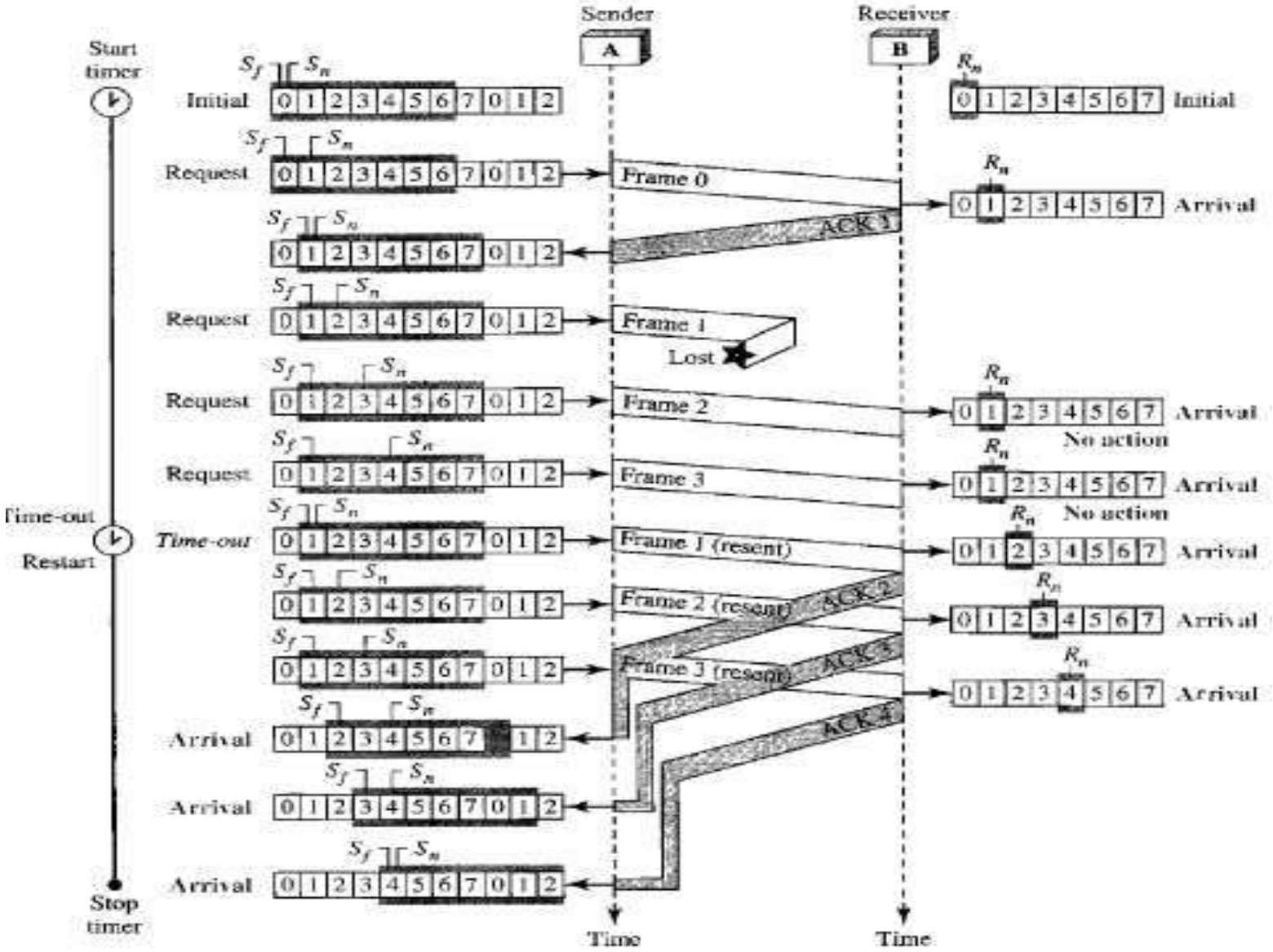
Algorithm 11.7 *Go-Back-N sender algorithm*

```
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
```

```
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 }
```

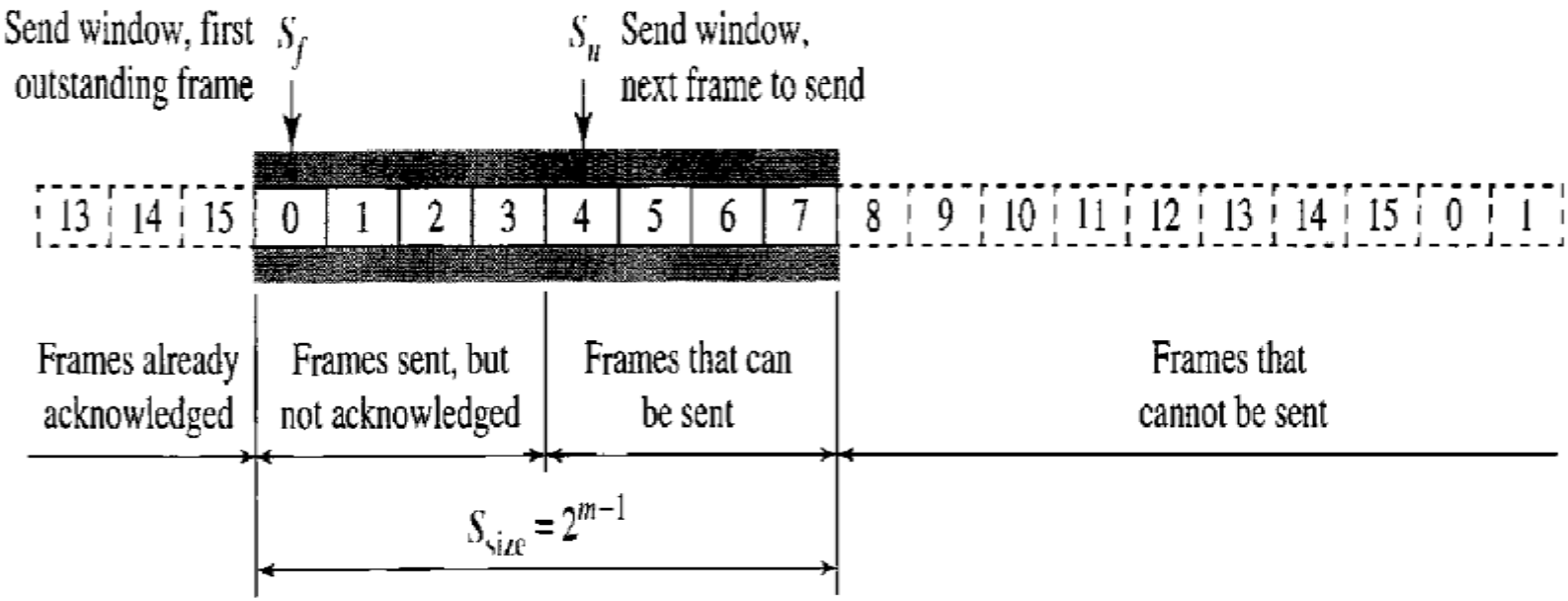
```
1  Rn = 0;
2
3  while (true)                          //Repeat forever
4  {
5    WaitForEvent();
6
7    if(Event(ArrivalNotification)) /Data frame arrives
8    {
9       Receive(Frame);
10      if(corrupted(Frame))
11           Sleep();
12      if(seqNo == Rn)                  //If expected frame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

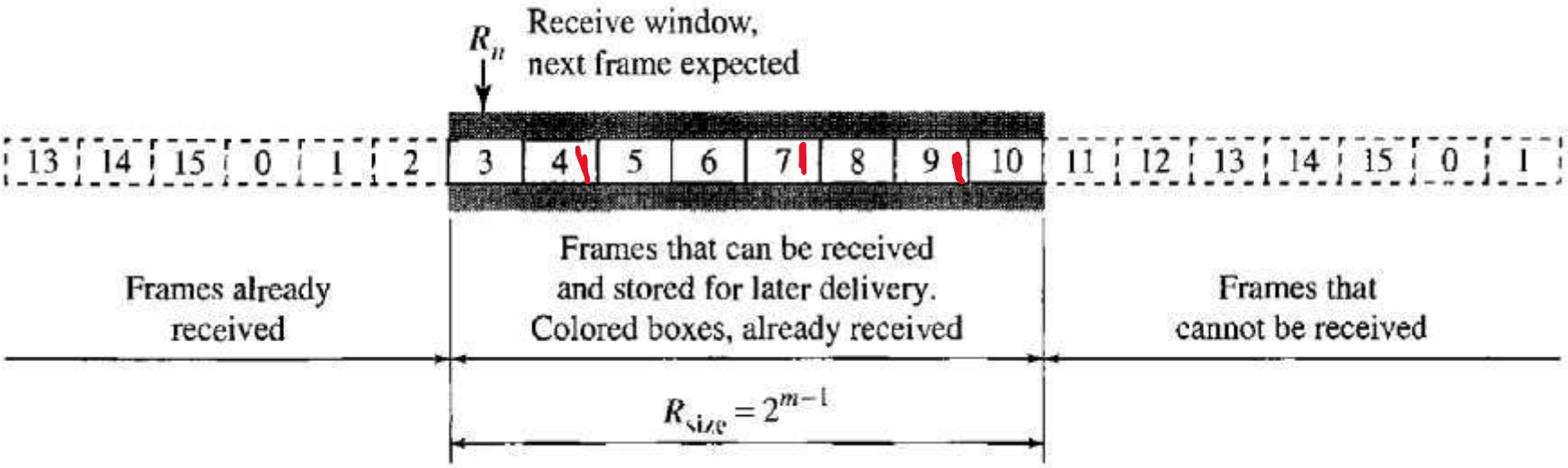

Sender
Receiver
A
B
Start timer
Stop timer
Initial
Request
Arrival
Frame 0
ACK 1
Frame 1
ACK 2
Frame 2
Lost
Frame 3
ACK 3
ACK 4
Time
Time

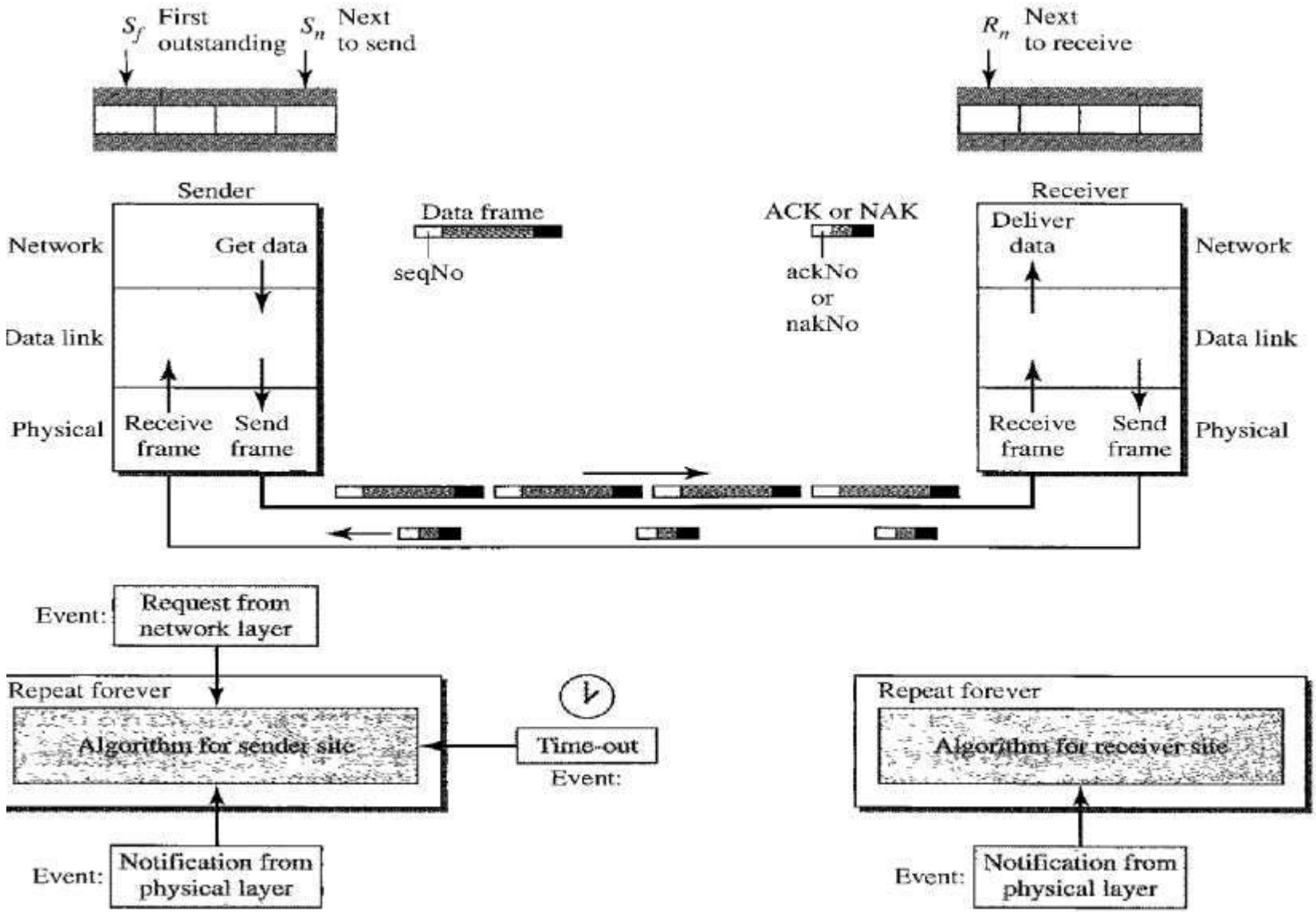

Sender
Receiver
A
B
Start timer
Initial
Request
Frame 0
ACK 1
Request
Frame 1
Lost
Request
Frame 2
No action
Request
Frame 3
No action
Time-out
Time-out
Restart
Frame 1 (resent)
Frame 2 (resent)
ACK 2
Frame 3 (resent)
ACK 3
ACK 4
Arrival
Arrival
Arrival
Stop timer
Time
Time
Initial
Arrival
Arrival
Arrival
Arrival

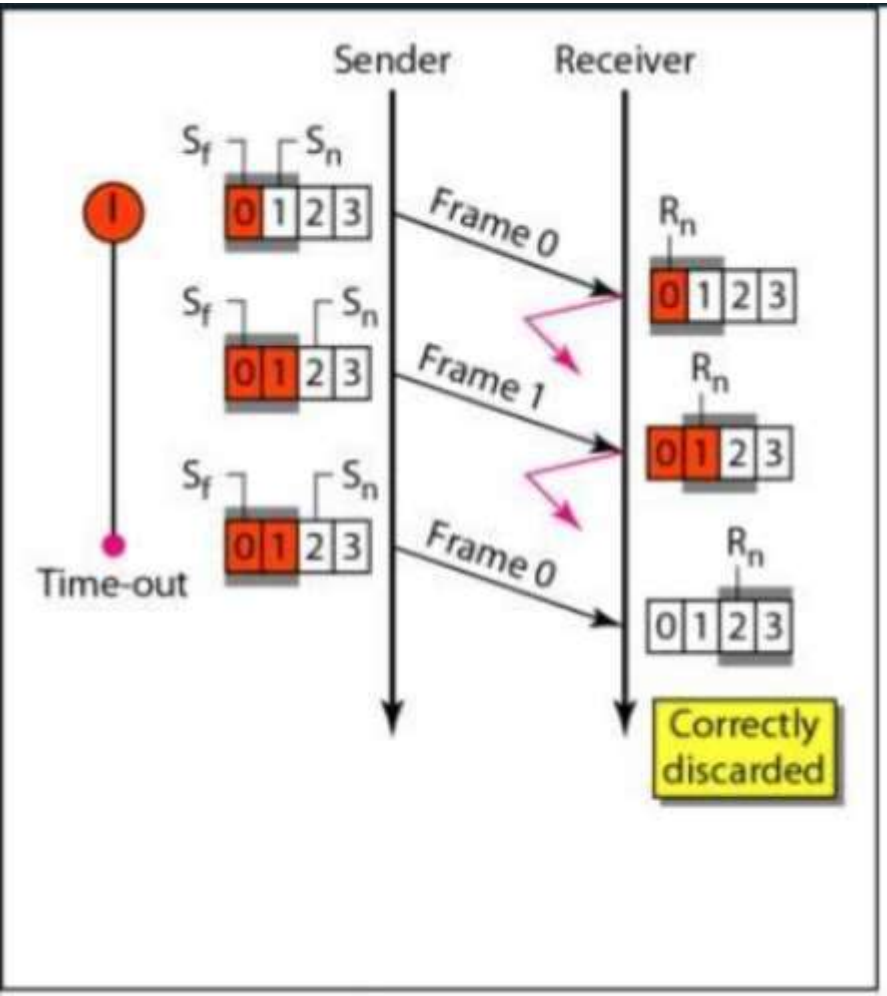
SELECTIVE REPEAT AUTOMATIC REPEAT REQUEST

- resending uses bandwidth & slows down the transmission.
- only damaged frame is resend
- Selective Repeat ARQ
- efficient for noisy links
- NAK

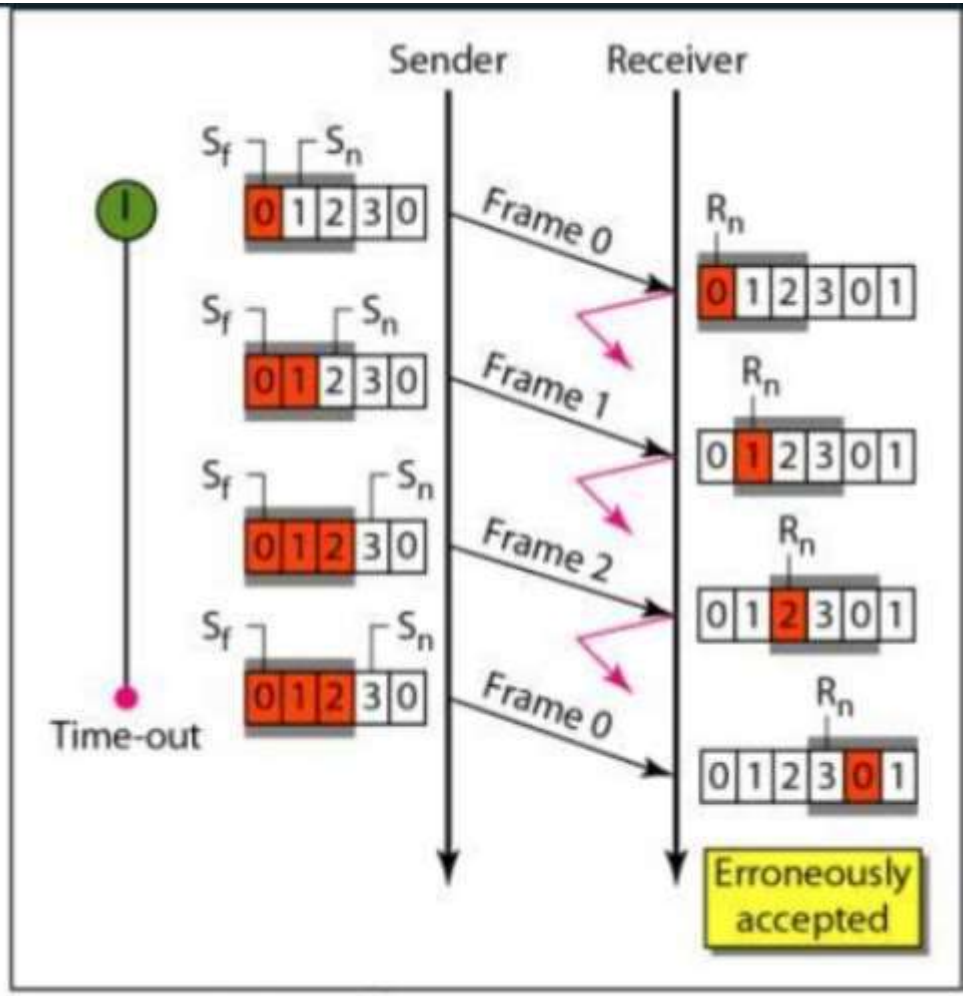

Send window, first outstanding frame
S_f
S_n
Send window, next frame to send
13 14 15 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 1
Frames already acknowledged
Frames sent, but not acknowledged
Frames that can be sent
Frames that cannot be sent
$S_{size} = 2^{m-1}$


R_n
Receive window,
next frame expected
13 14 15 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 0 1
Frames already
received
Frames that can be received
and stored for later delivery.
Colored boxes, already received
Frames that
cannot be received
$R_{size} = 2^{m-1}$

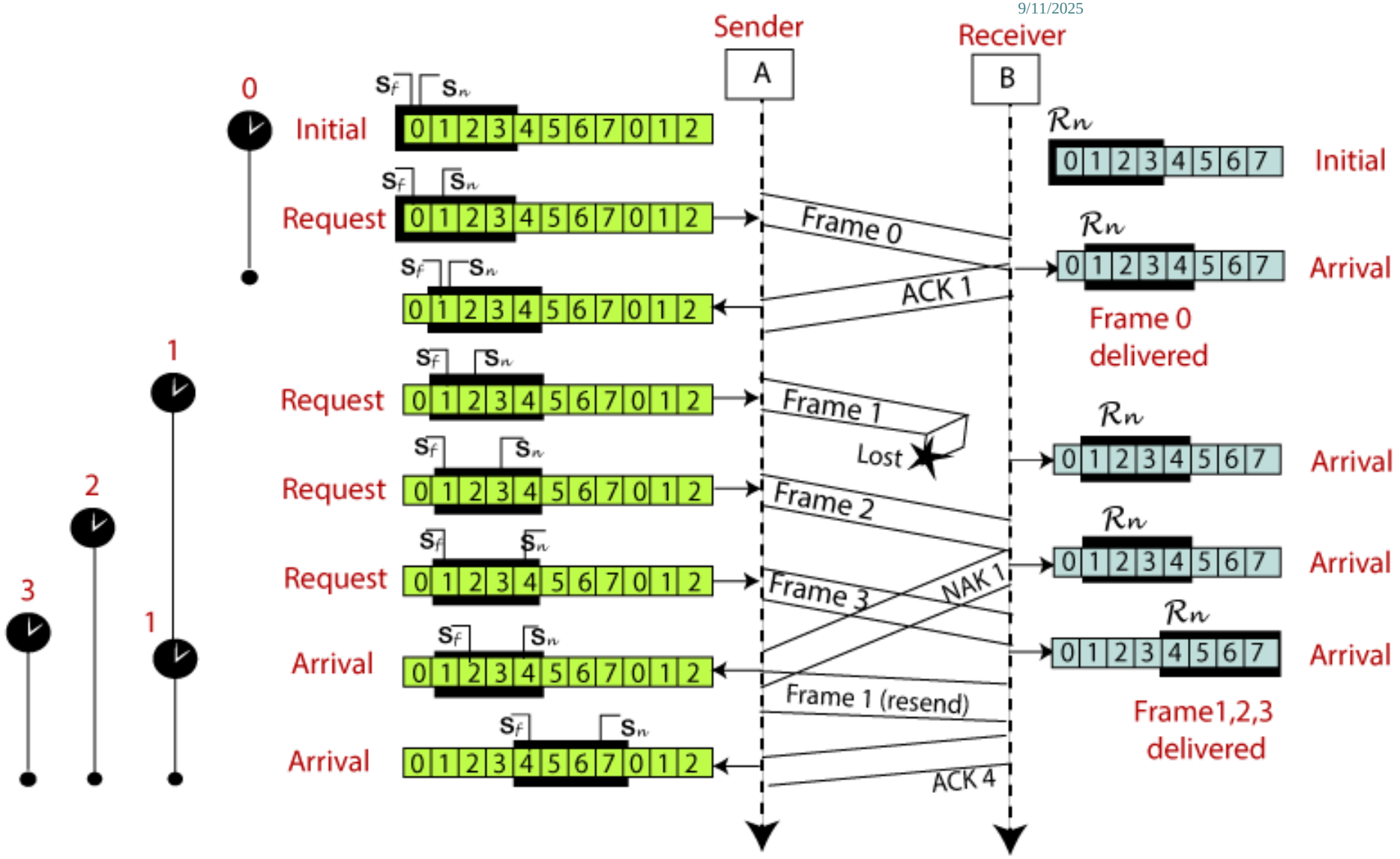

S_f First outstanding
S_n Next to send
R_n Next to receive
Sender
Receiver
Network
Get data
Data link
Physical
Receive frame
Send frame
Data frame
seqNo
ACK or NAK
ackNo or nakNo
Deliver data
Network
Data link
Physical
Receive frame
Send frame
Event:
Request from network layer
Repeat forever
Algorithm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



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

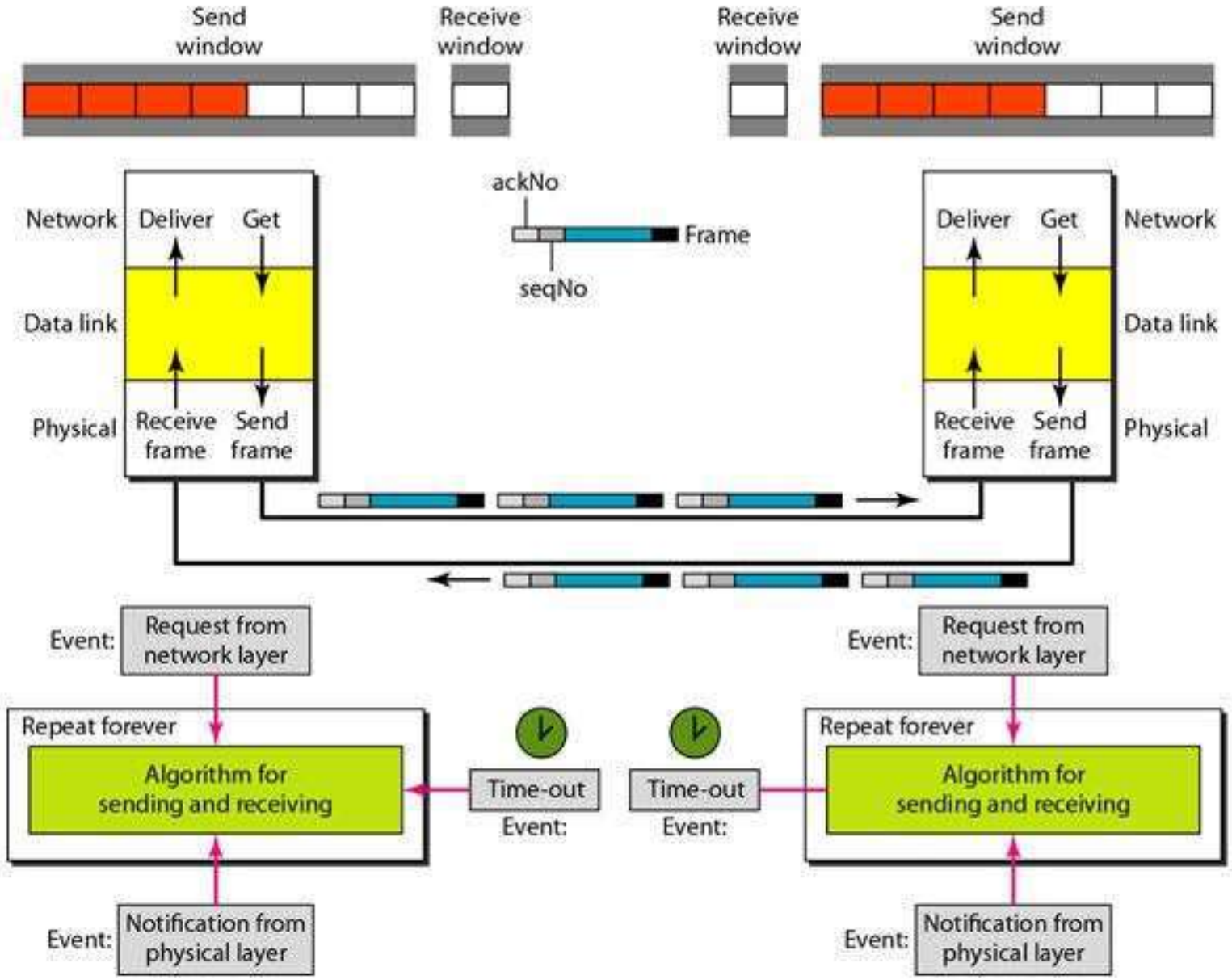


b. Window size > 2^{m-1}


Sender
Receiver
A
B
0
Initial
Request
Frame 0
ACK 1
Frame 0 delivered
1
Request
Frame 1
Lost
2
Request
Frame 2
3
Request
Frame 3
NAK 1
1
Arrival
Frame 1 (resend)
Frame1,2,3 delivered
Arrival
ACK 4
Initial
Arrival
Arrival
Arrival
Arrival

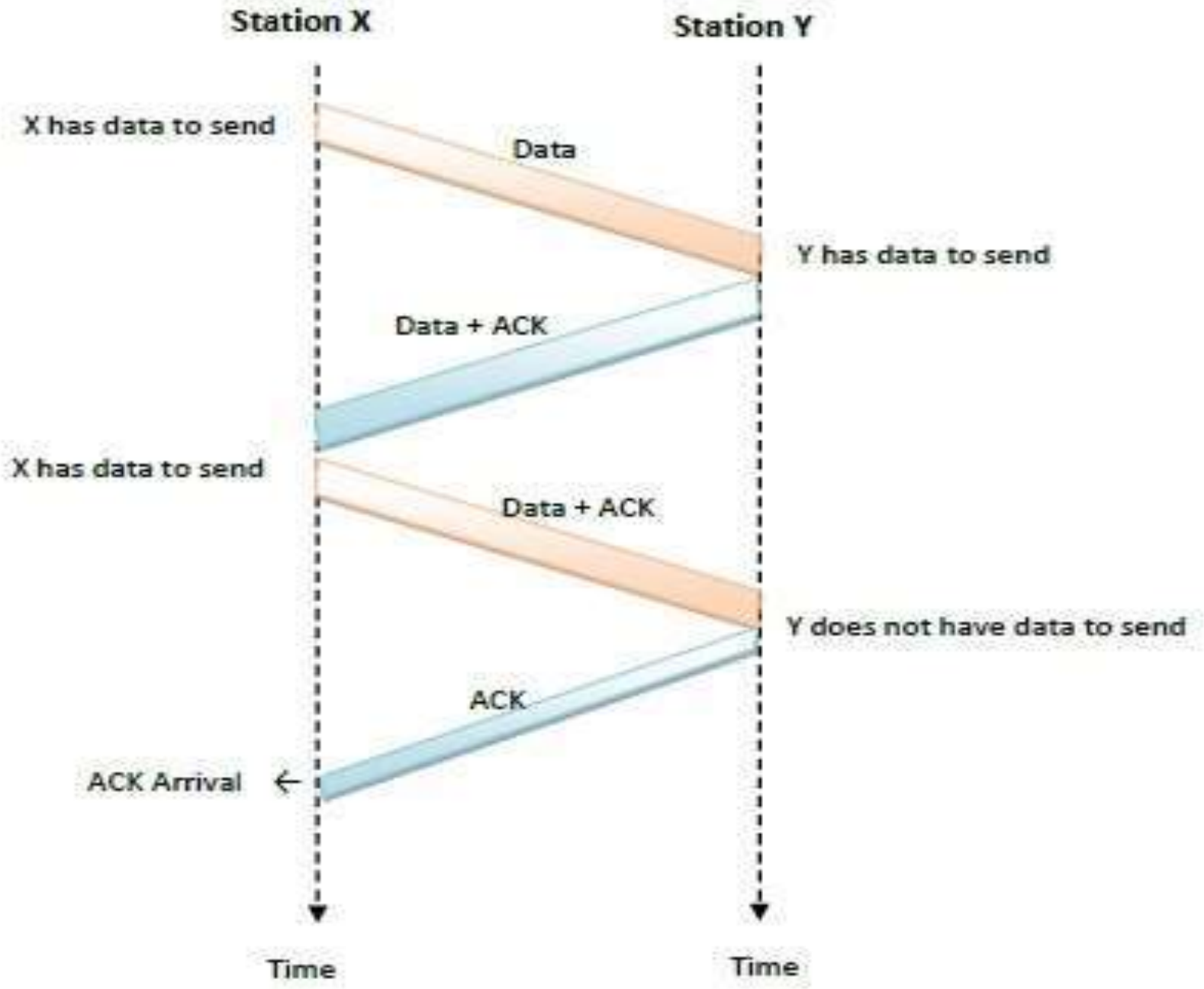
PIGGYBACKING

- Bidirectional Transmission
- data frames are flowing in both directions
  - control info also need to flow in both directions
  - A→B
    - frame is carrying data from A to B,
    - also carries control info about arrived/lost frames from B.


Send window
Receive window
Receive window
Send window
Network
Deliver
Get
Data link
Physical
Receive frame
Send frame
ackNo
Frame
seqNo
Deliver
Get
Network
Data link
Receive frame
Send frame
Physical
Event:
Request from network layer
Repeat forever
Algorithm for sending and receiving
Time-out
Event:
Time-out
Event:
Event:
Request from network layer
Repeat forever
Algorithm for sending and receiving
Event:
Notification from physical layer
Event:
Notification from physical layer

- each node now has two windows:
  - one send window and one receive window.
  - both also need to use a timer
  - both are involved in three types of events:
    - request, arrival, and time-out

- when a frame arrives, the site needs to handle control information as well as the frame itself
  - taken care in arrival event → complicated
  - request event uses only the send window at each site
  - both sites must use the same algorithm


Station X
Station Y
X has data to send
Data
Y has data to send
Data + ACK
X has data to send
Data + ACK
Y does not have data to send
ACK
ACK Arrival
Time
Time

DATA LINK PROTOCOLS WIDELY USED

- HIGH LEVEL DATA LINK PROTOCOL
- THE POINT-TO-POINT PROTOCOL
- Implements ARQ mechanisms

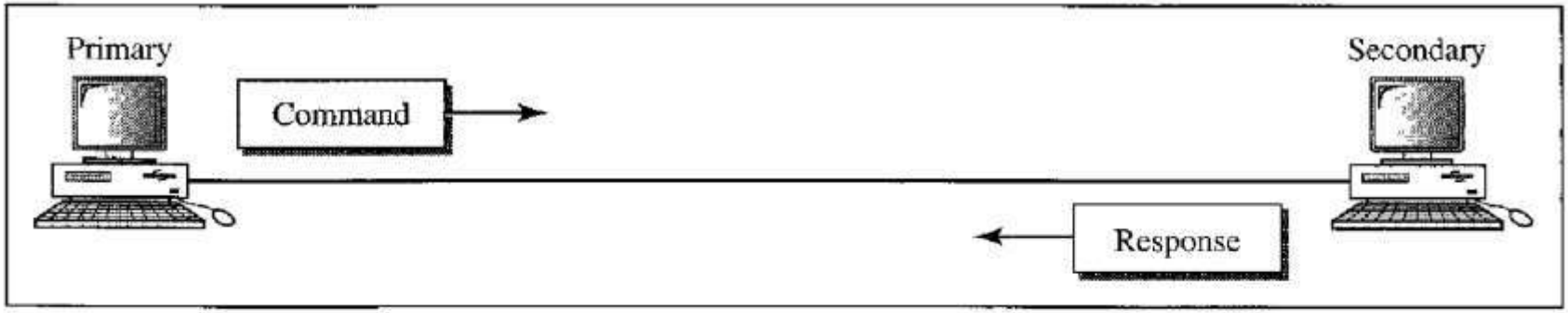
High-Level Data Link Control

- HDLC is bit oriented protocol, for communication over point-to-point & multipoint links.
- IBM mainframe world → Synchronous Data link control protocol (SDLC)
- Submitted to ANSI & ISO,
  - ANSI → ADCCP(Advanced Data Communication Control Procedure)
  - ISO → HDLC
  - CCITT → HDLC for LAP(Link Access Procedure) to LAPB

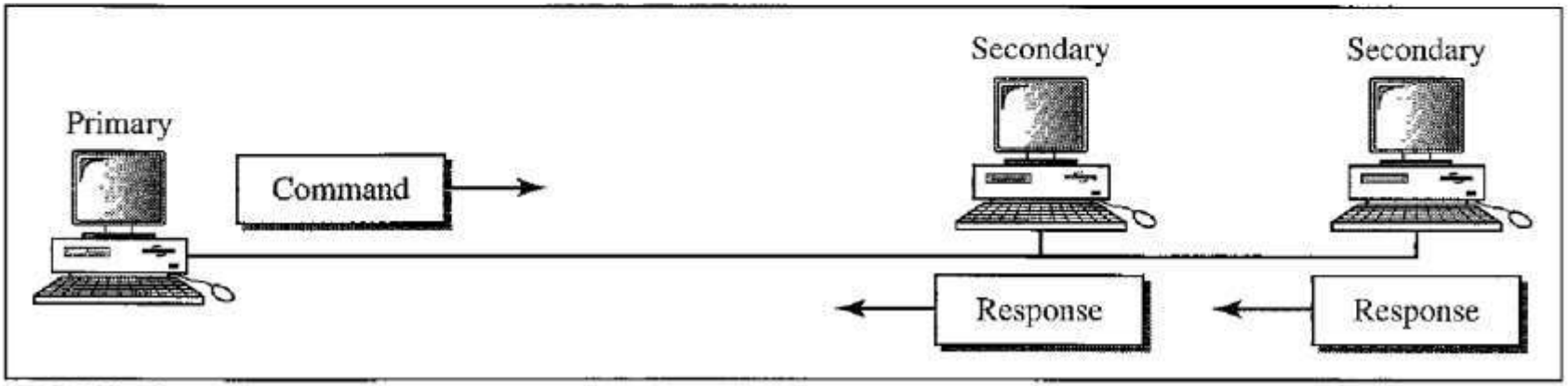
- Configuration & transfer modes
  - 2 transfer modes are used in different configuration:

1. Normal response mode (NRM)
2. Asynchronous balanced mode (ABM)

Normal Response Mode → Station configuration is Unbalanced

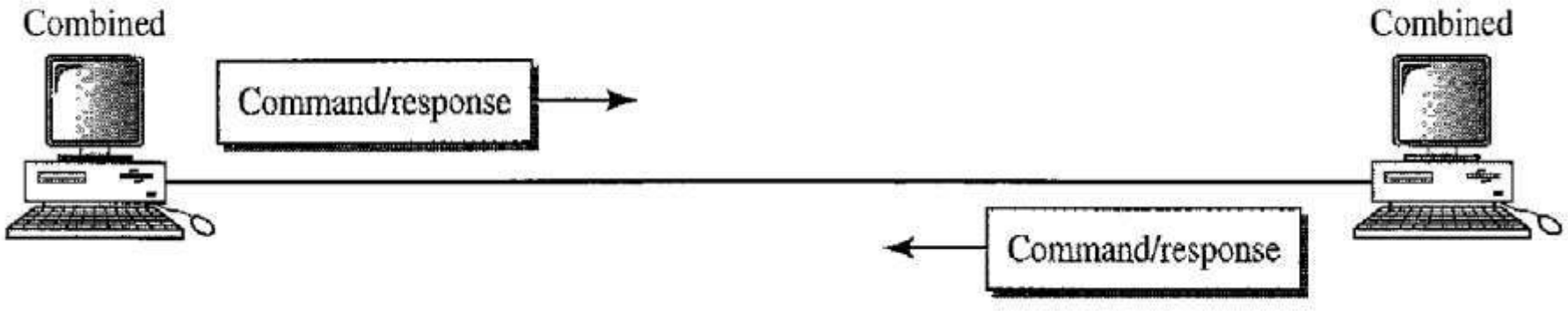


a. Point-to-point



b. Multipoint

Asynchronous Balanced Mode → Station configuration is balanced



Frames

01111110	Address	Control	Data	Checksum	01111110
8	8	8	>=0	16	8

- Address → Multiple terminals/ to identify one of the terminals
- Control → Sequence no's, Ack's etc
- Data → Any info, arbitrarily long
- Checksum → To detect errors, is a CRC
- Flag's → flags sequences are transmitted continuously

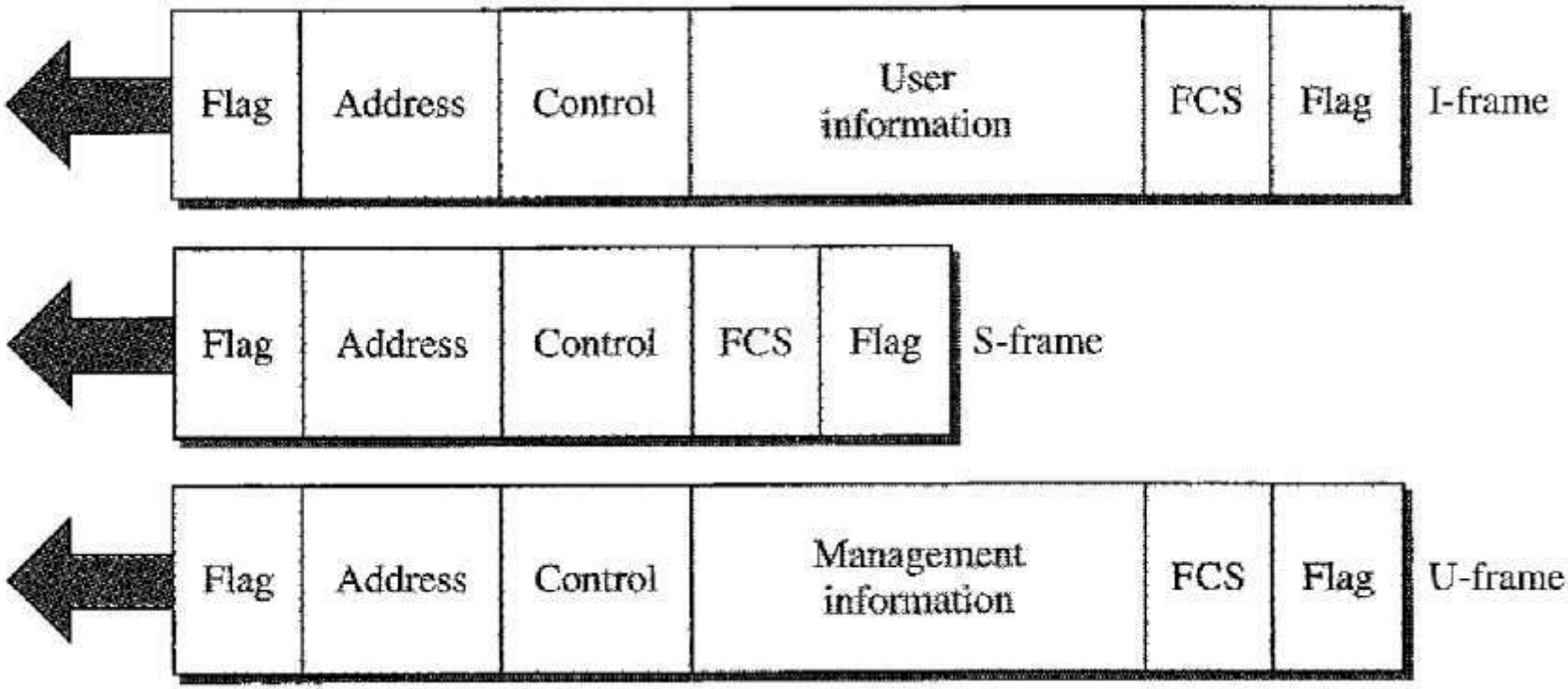
- Minimum, frame contain 3 fields & total of 32 bits excluding flags on either end

- 3 kinds of frames

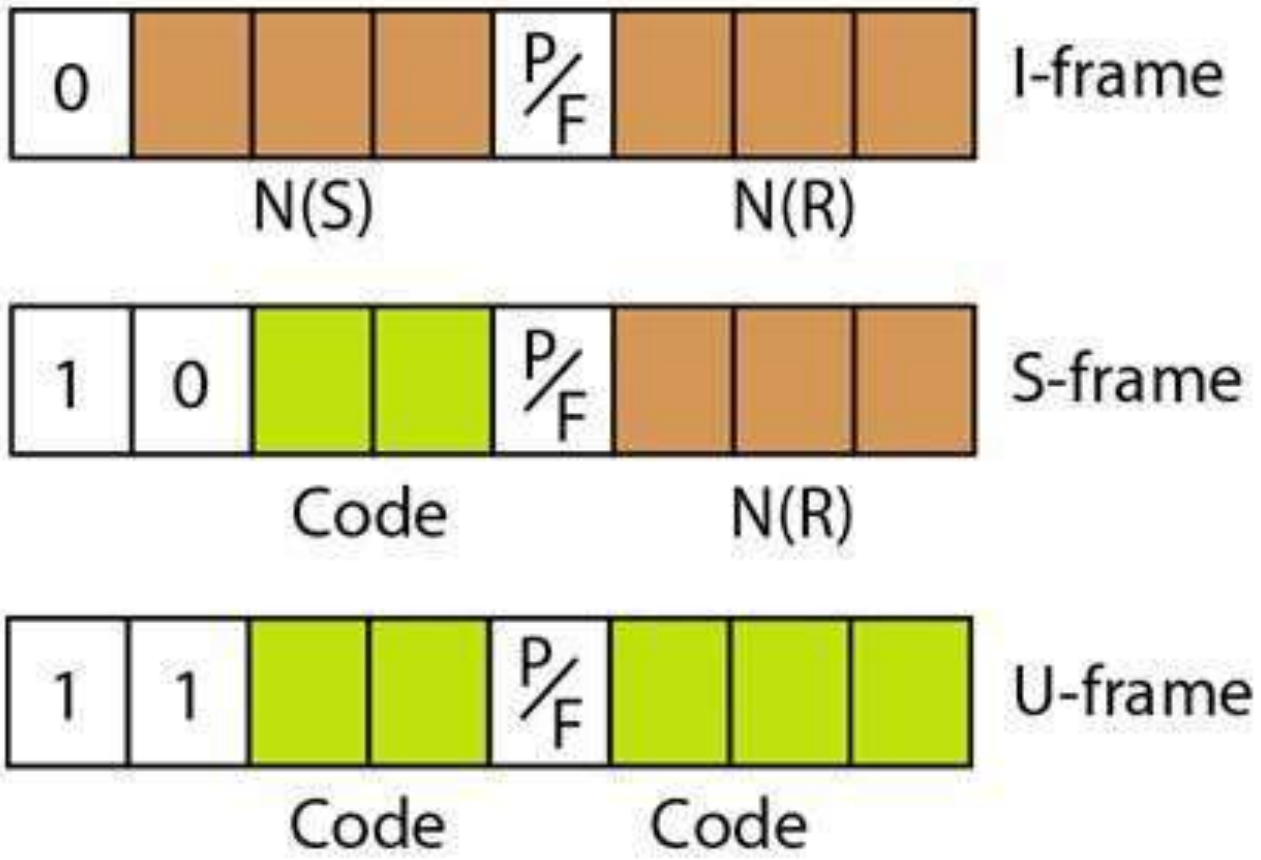
1. Information frames (I-frames)
   - used to transport user data & control info relating to user data.

2. Supervisory frames (S-frames)
   - used only to support control info.

3. Unnumbered frames (U-frames)
   - reserved for system management.


Flag
Address
Control
User information
FCS
Flag
I-frame
Flag
Address
Control
FCS
Flag
S-frame
Flag
Address
Control
Management information
FCS
Flag
U-frame

Control field of I-frame, S-frame, U-frame



- I-frame control field

  - 1st bit → defines type,if 0, its I-frame
  - next 3 bits, N(S),define sequence no of the frame
  - last 3 bits, N(R), ack no, when piggybacking is used.
  - P/F, if 1, can mean poll/final

    - Poll → frame is sent by primary station to secondary
    - Final → frame is sent by secondary to a primary

- S-frame control field
  - 1st 2 bits are 10, its S-frame
  - last 3 bits, N(R), ACK or NAK depending on type of frame
  - 2 bits, Code → type of S-frame, 4 types
    - Receive Ready (RR) 00
    - Receive not Ready (RNR) 10
    - Reject (REJ) 01
    - Selective Reject (SREJ) 11

U- frame control field

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

POINT TO POINT PROTOCOL

- SERVICES
  1. defines the format of the frame to be exchanged between devices
  2. how 2 devices can negotiate the establishment of the link & the exchange of data.
  3. how network layer data are encapsulated in data link frame
  4. how 2 devices can authenticate each other
  5. provides multiple layer services supporting a variety of network layer protocols
  6. provides connections over multiple links
  7. provides network address configuration.

3 features:

1. A framing method that unambiguously delineates the end of one frame and the start of the next one. The frame format also handles error detection.
2. A link control protocol for bringing lines up, testing them, negotiating options, and bringing them down again gracefully when they are no longer needed. This protocol is called **LCP (Link Control Protocol)**.
3. A way to negotiate network-layer options in a way that is independent of the network layer protocol to be used. The method chosen is to have a different **NCP (Network Control Protocol)** for each network layer supported.

- No flow control
- Lack of error control & Sequence numbering may cause packet to be received out of order.
- Does not provide a sophisticated addressing mechanism in a multi point configuration.

- Frames
  - Byte-oriented protocol

•Address field → always set to binary value 11111111, all stations are to accept the frame.

• Control → PPP does not provide reliable transmission using seq,ack's.

• Protocol field → 0 bit are network layer protocols, 1 bit are used to negotiate other protocols.

• Payload field → default length 1500 byte

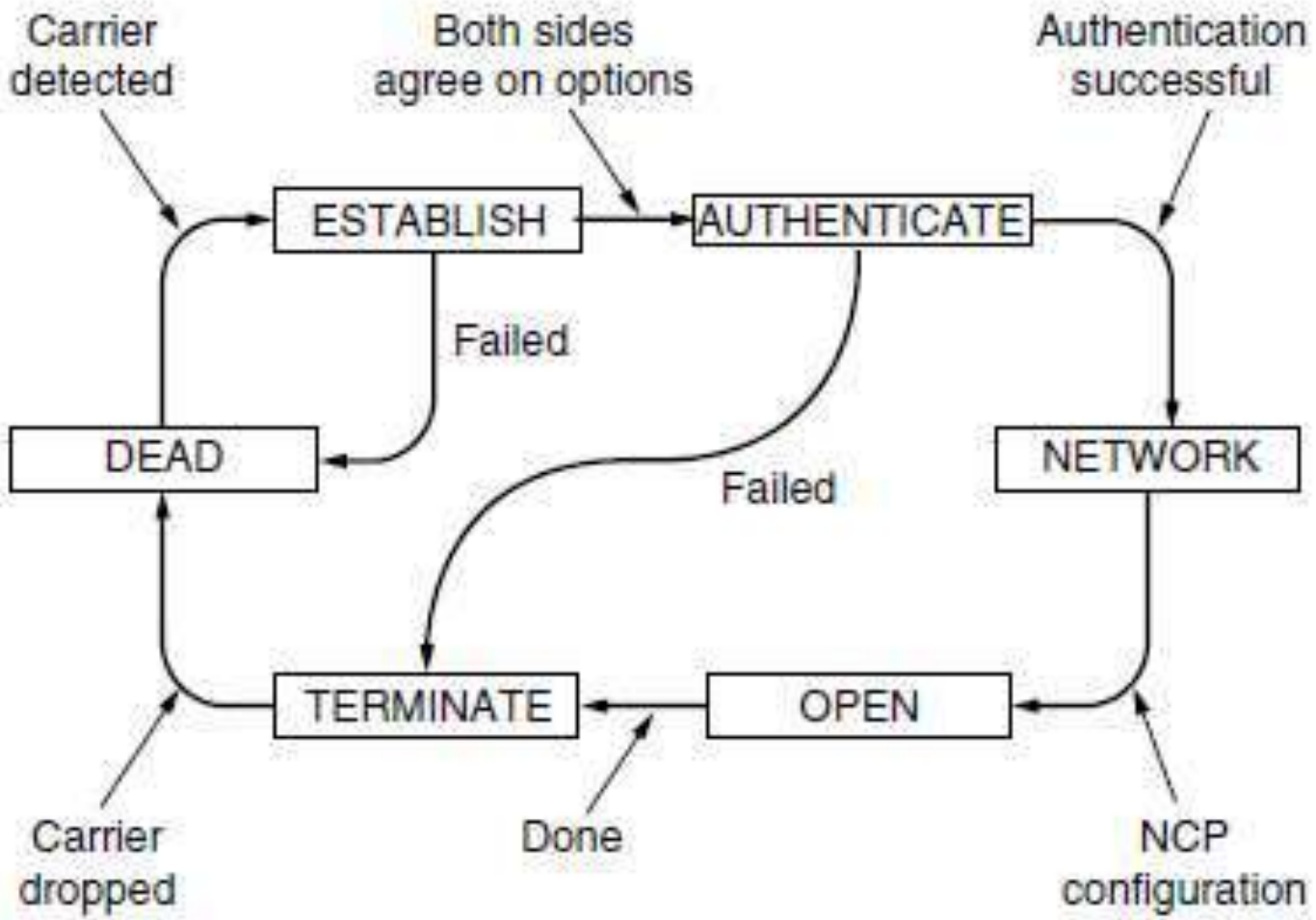
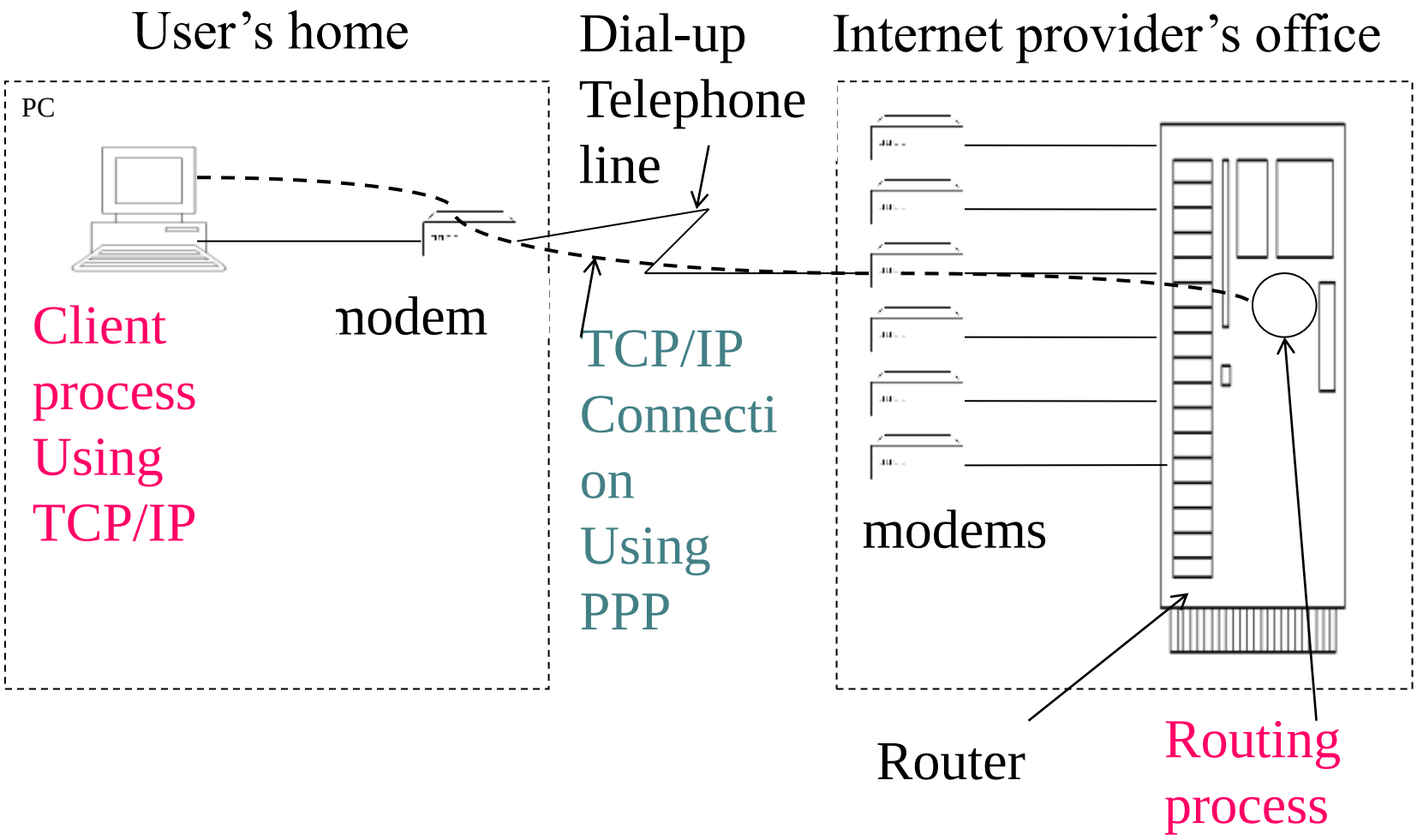


Figure 3-25. State diagram for bringing a PPP link up and down.

DLL in the Internet



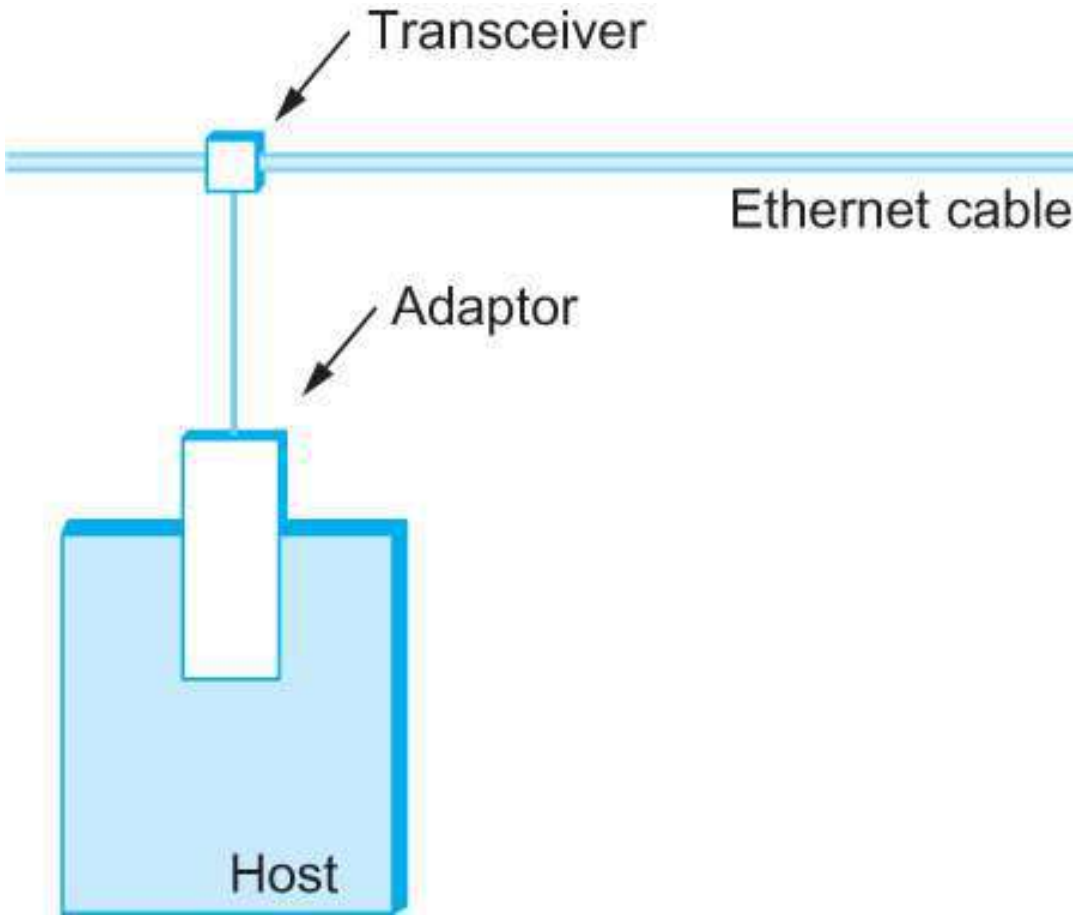
ETHERNET (802.3)

- Most successful LAN technology for last 20 years
- Developed by researchers at Xerox Palo Alto Research Center (PARC)
- Working example of CSMA/CD LAN technology
- Roots in an early packet radio network, called Aloha
  - how to mediate access to a shared medium fairly & efficiently
  - Aloha → medium was the atmosphere
  - Ethernet → medium is a coax cable
  - Core idea →an algorithm that controls when each node can transmit

- DEC & Intel Corporation joined Xerox to define a 10-Mbps Ethernet standard in 1978
- Basis for IEEE standard 802.3
- Extended to include a 100-Mbps version called Fast Ethernet, & a 1000-Mbps version called Gigabit Ethernet
- 10 Mbps Ethernet

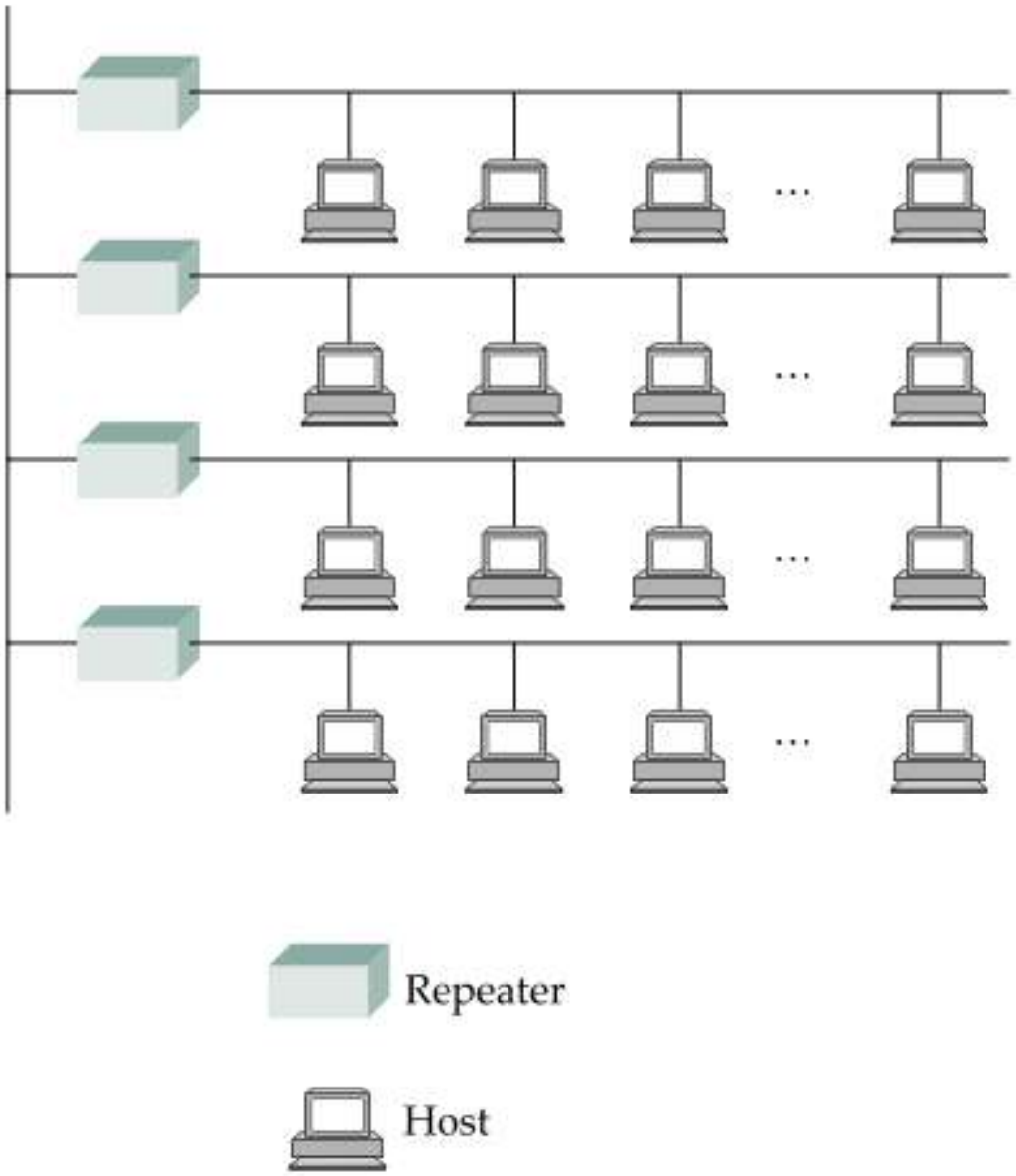
1. Physical Properties

- An Ethernet segment is implemented on a co-axial cable of upto 500m
- Hosts connect to an Ethernet segment by tapping into it.
- A transceiver is a small device directly attached to the tap
  - detects when the line is idle and drives signal when the host is transmitting.
  - transceiver also receives incoming signal.
  - transceiver is connected to an Ethernet adaptor which is plugged into the host.
  - protocol is implemented on the adaptor.



Ethernet transceiver and adaptor

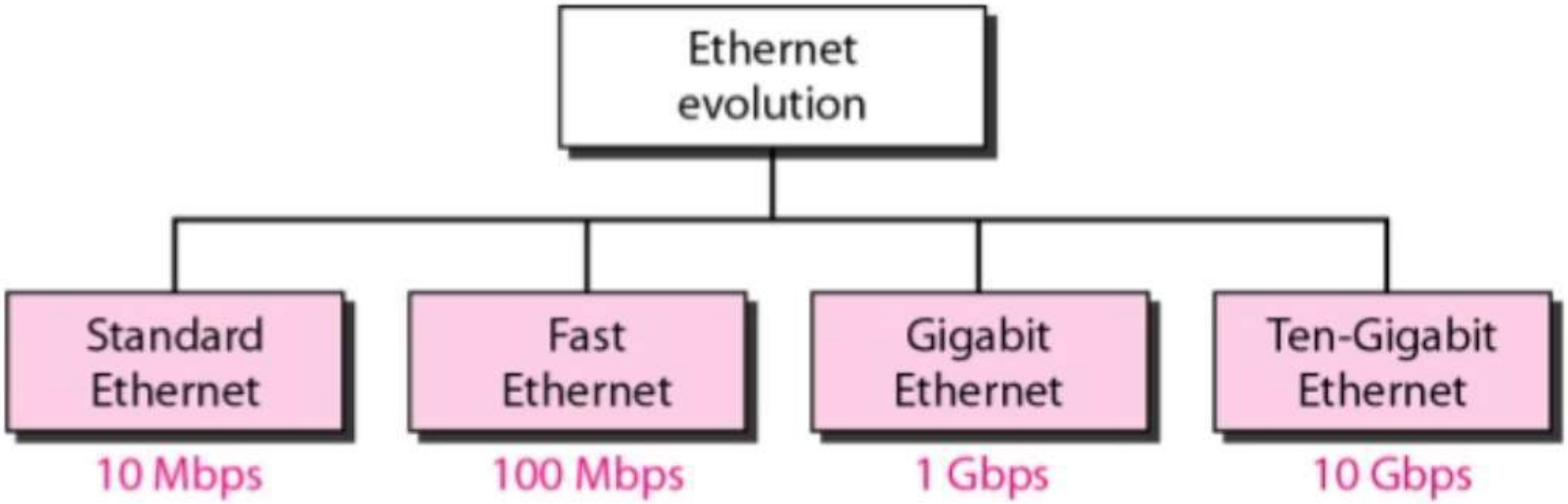
- Multiple Ethernet segments can be joined together by repeaters.
- A **repeater** is a device that forwards digital signals.
- No more than four repeaters may be positioned between any pair of hosts.
  - An Ethernet has a total reach of only 2500 m.

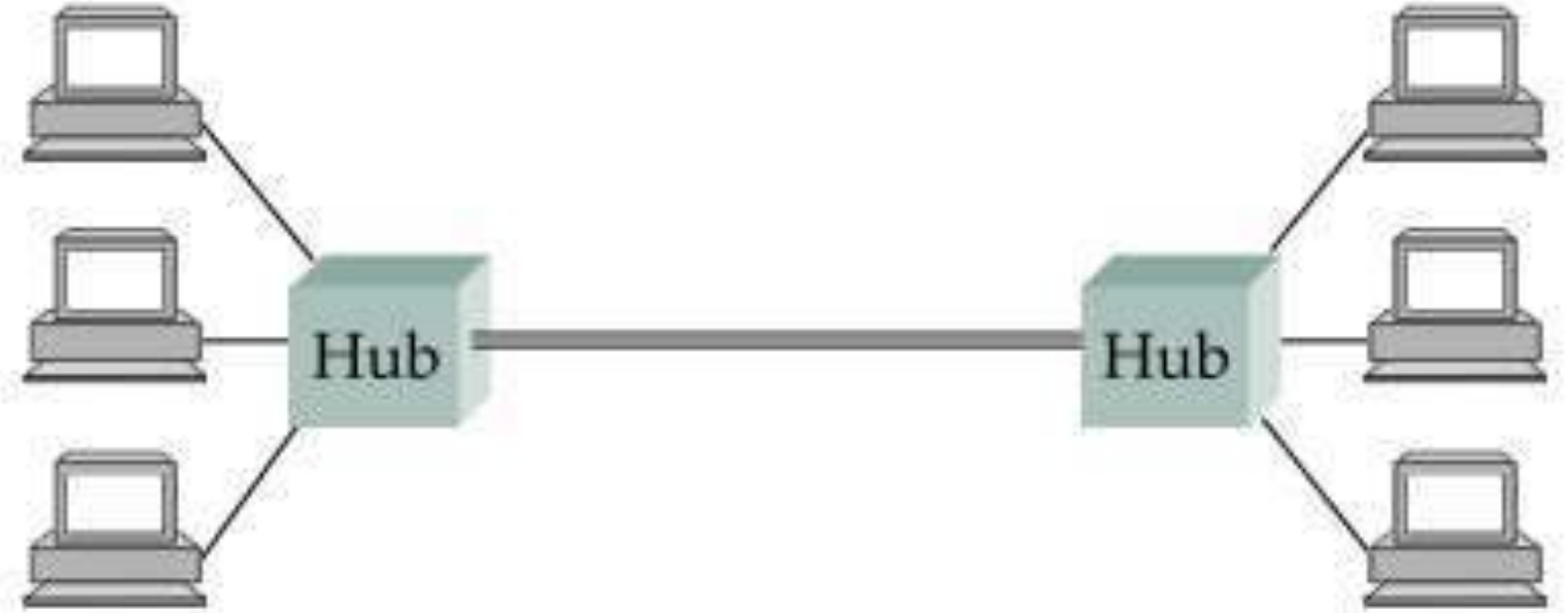


Ethernet Repeater

- Any signal placed on the Ethernet by a host is broadcast over the entire network
  - Signal is propagated in both directions.
  - Repeaters forward the signal on all outgoing segments.
  - Terminators attached to the end of each segment absorb the signal.
- Ethernet uses Manchester encoding scheme

- **New Technologies in Ethernet**
  - Instead of using coax cable, an Ethernet can be constructed from a thinner cable, **10Base2**(the original was 10Base5)
    - 10 means the network operates at **10 Mbps**
    - Base means the cable is used in a **base band system**
    - 2means that a given segment can be **no longer than 200 m**
  - **10BaseT**
    - T stands for **twisted pair**
    - Limited to **100 m in length**
    - With 10BaseT, the common configuration is to have **several point to point segments** coming out of a multi-way repeater, called **Hub**


Ethernet evolution
Standard Ethernet
10 Mbps
Fast Ethernet
100 Mbps
Gigabit Ethernet
1 Gbps
Ten-Gigabit Ethernet
10 Gbps

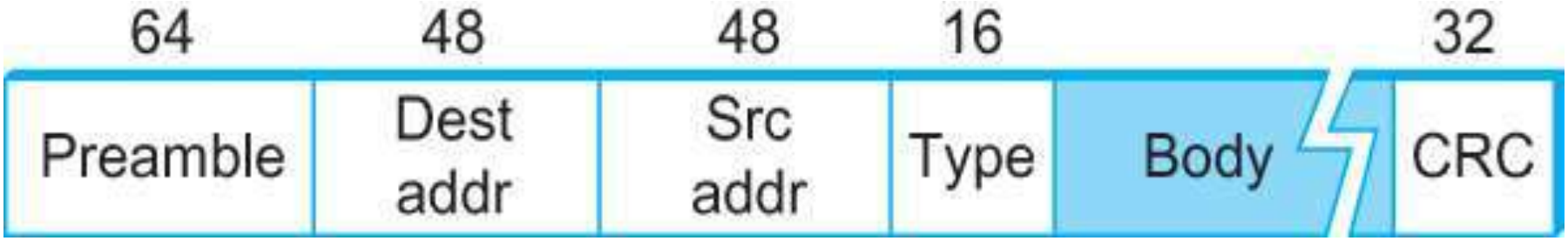


Ethernet hub

2. ACCESS PROTOCOL

- The algorithm that controls access to the shared internet link is commonly called Ethernet's Media Access Control (MAC).
  - implemented in Hardware on the network adaptor.
- **Frame format**
  - **Preamble (64bit)**: allows the receiver to synchronize with the signal (sequence of alternating 0s and 1s).
  - **Source and Destination Address (48bit each)**.
  - **Packet type (16bit)**: acts as demux key to identify the higher level protocol.
  - **Data (up to 1500 bytes)**
    - Minimally a frame must contain at least 46 bytes of data.
    - Frame must be long enough to detect collision.
  - **CRC (32bit)**

ETHERNET FRAME FORMAT



ETHERNET ADDRESS

- Each host on an Ethernet has a unique Ethernet Address.
- The address belongs to the adaptor, not the host.
  - It is usually burnt into ROM.
- Ethernet addresses are typically printed in a human readable format
  - sequence of six numbers separated by colons.
  - Each number corresponds to 1 byte of the 6 byte address and is given by a pair of hexadecimal digits, one for each of the 4-bit nibbles in the byte
  - Leading 0s are dropped.
  - Eg→ **8:0:2b:e4:b1:2** is
    - **00001000 00000000 00101011 11100100 10110001 00000010**

- To ensure that every adaptor gets a unique address, each manufacturer of Ethernet devices is allocated a different **prefix**
  - AMD has been assigned the 24bit prefix 8:0:20
- Each **frame transmitted** is received by **every adaptor** connected to that Ethernet.
- Each **adaptor** recognizes those frames addressed to its **address** and passes only those **frames on to the host**.
- In addition, to **unicast** address, an Ethernet address consisting of all **1's** is treated as a **broadcast** address.
  - All adaptors pass frames addressed to the *broadcast* address up to the host.

- An address that has the first bit set **to 1** but is not the broadcast address is called **a multicast address**.
  - A given host can program its adaptor to accept some set of multicast addresses.

- To summarize, an Ethernet adaptor receives all frames and accepts
  - Frames addressed to its own address
  - Frames addressed to the broadcast address
  - Frames addressed to a multicast addressed if it has been instructed

Transmitter Algorithm

- When the adaptor has a frame to send and the line is idle, it transmits the frame immediately.
  - The upper bound of 1500 bytes in the message means that the adaptor can occupy the line for a fixed length of time.
- When the adaptor has a frame to send and the line is busy, it waits for the line to go idle and then transmits immediately.
- The Ethernet is said to be 1-persistent protocol because an adaptor with a frame to send transmits with probability 1 whenever a busy line goes idle.

- Since there is **no centralized control** it is possible for two (or more) adaptors to begin transmitting at the same time,
  - Either because both found the line to be idle,
  - Or, both had been waiting for a busy line to become idle.

- When this happens, the **two (or more) frames** are said to be **collide** on the network

- Since Ethernet **supports collision detection**, each sender is able to determine that a collision is in progress

- At the moment an adaptor detects that its frame is colliding with another, it first makes sure to transmit a 32-bit jamming sequence and then stops transmission.
  - Thus, a transmitter will minimally send 96 bits in the case of collision
    - 64-bit preamble + 32-bit jamming sequence (**Runt frame**)

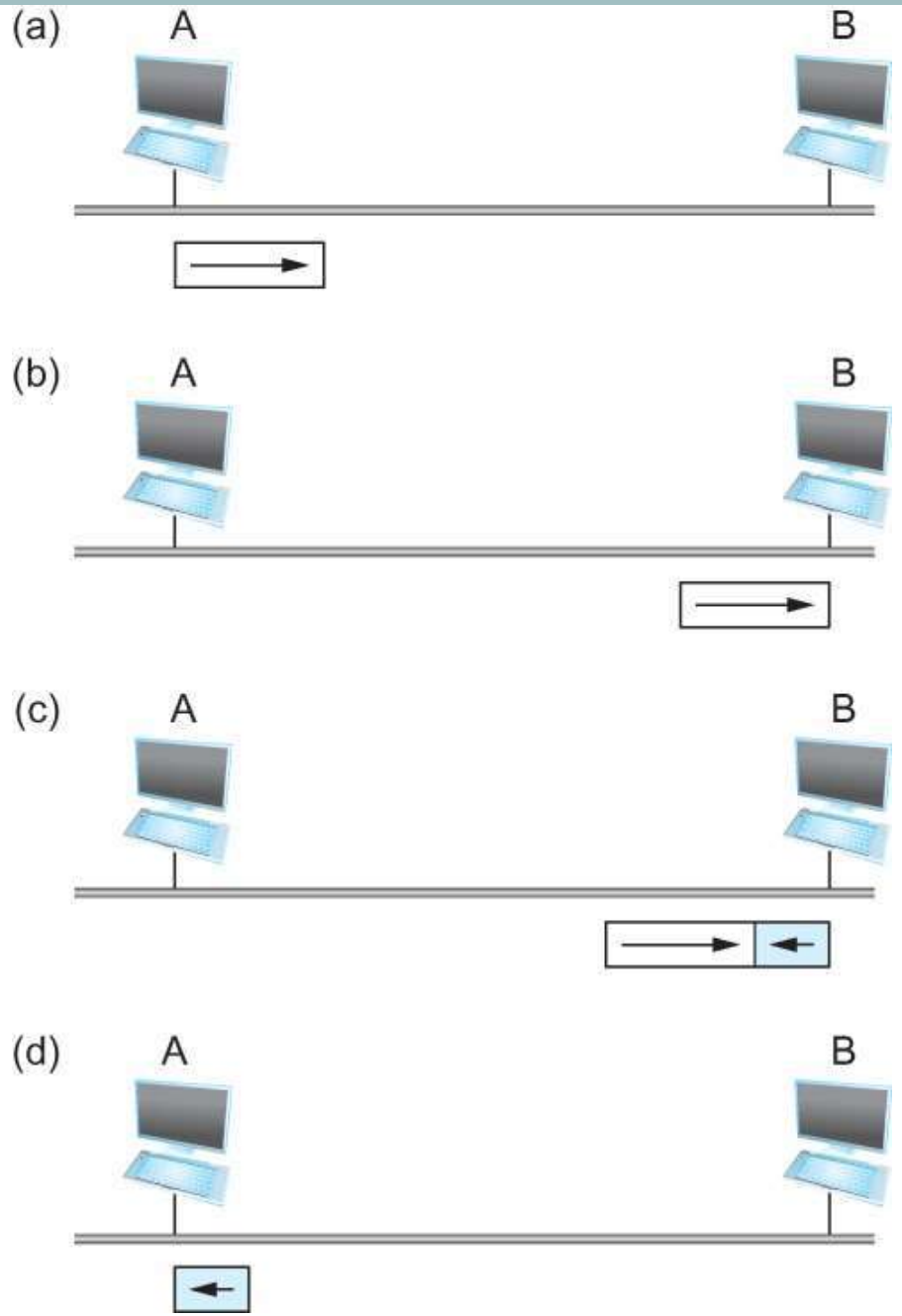
- If hosts were far, they would have had to transmit longer, and thus send more bits, before detecting the collision

- Worst case scenario → when the two hosts are at **opposite ends of the Ethernet**

- To know for sure that the frame its just sent did not collide with another frame, the transmitter may need to send as many as 512 bits.
  - Every Ethernet frame must be at least **512 bits** (64 bytes) long.
    - **14 bytes of header + 46 bytes of data + 4 bytes of CRC**

- Why 512 bits…
  - Why is its length limited to 2500 m?

1. **A** begins transmitting a frame at time ***t***
2. ***d*** denotes the one link latency
3. The first bit of **A**'s frame arrives at **B** at time ***t + d***
4. Suppose an instant before **host A**'s frame arrives, **host B** begins to transmit its own frame
5. **B's frame** will immediately collide with **A's frame** and this collision will be detected by **host B**
6. **Host B** will send the **32-bit jamming sequence**
7. Host **A** will not know that the collision occurred until **B's frame** reaches it, which will happen at ***t + 2 * d***
8. Host A must continue to transmit until this time in order to detect the collision
    1. Host A must transmit for *2 * d* to be sure that it detects all possible collisions



Worst-case scenario:

a. A sends a frame at time t;

b. A's frame arrives at B at time t + d;

c. B begins transmitting at time t + d and collides with A's frame;

d. B's runt (32-bit) frame arrives at A at time t + 2d

- Ethernet is 2500 m long, and there may be up to four repeaters between any two hosts, the round trip delay has been determined to be 51.2 μs
  - Which on 10 Mbps Ethernet corresponds to 512 bits

- The other way to look at this situation,
  - We need to limit the Ethernet's maximum latency to a fairly small value (51.2 μs) for the access algorithm to work
    - Hence the maximum length for the Ethernet is on the order of 2500 m.

- Once an adaptor has detected a collision, and stopped its transmission, it waits a certain amount of time and tries again.

- Each time the adaptor tries to transmit but fails, it doubles the amount of time it waits before trying again.

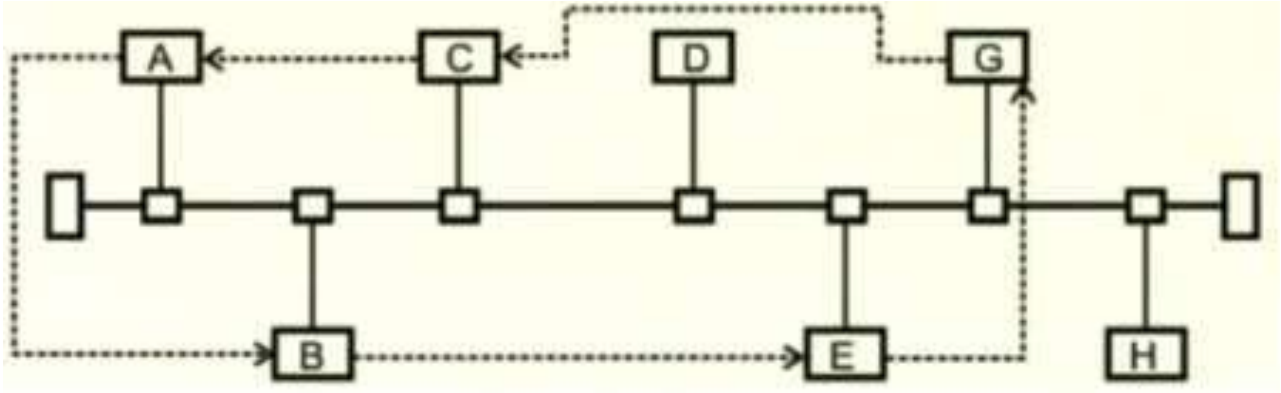
- This strategy of doubling the delay interval between each retransmission attempt is known as **Exponential Backoff**.

- The adaptor first delays either 0 or 51.2 μs, selected at random.
- If this effort fails, it then waits 0, 51.2, 102.4, 153.6 μs (selected randomly) before trying again;
  - This is $k * 51.2$ for $k = 0, 1, 2, 3$
- After the third collision,
  - it waits $k * 51.2$ for $k = 0 \ldots 2^3 - 1$ (again selected at random).
- In general, the algorithm randomly selects a k between 0 and $2^n - 1$ and waits for $k * 51.2$ μs, where n is the number of collisions experienced so far.

- Three LAN standards
  - **802.3** based on the CSMA/CD
  - **802.4** token bus
  - **802.5** token ring

802.4 token bus

- Maximum delay is deterministic



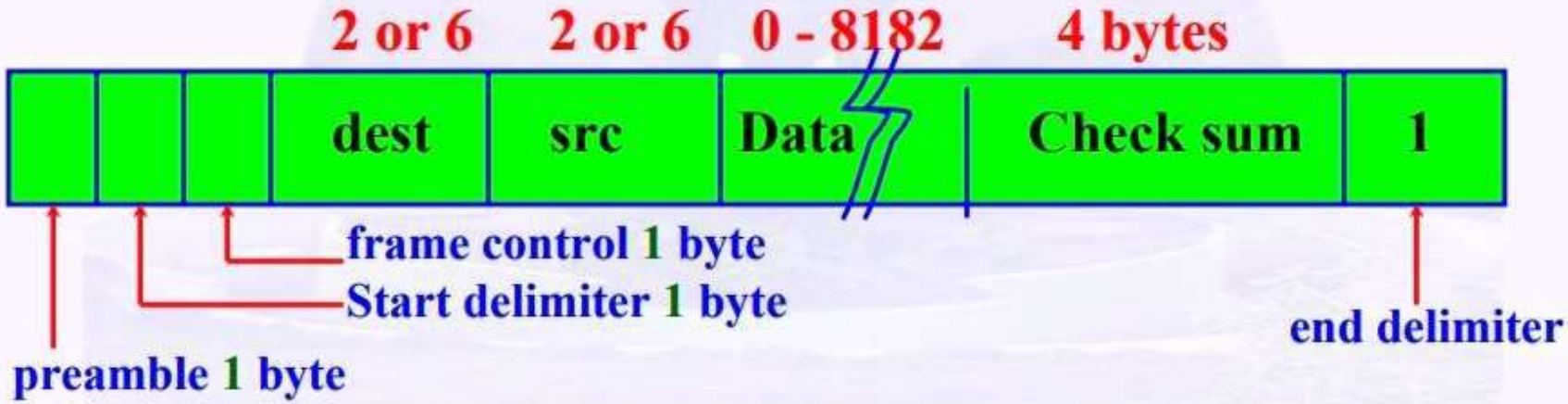
• Token bus is a physical bus that operates as a logical ring using tokens.

• Combines features of 802.3 & 8.2.5

- Token ring requires that stations takes turns sending data
- Each station may transmit only during its turn, & may send only during its turn
- The mechanism that coordinates this rotation is called token passing

- priority classes 0, 2, 4 and 6
  - 0 as the lowest
  - 6 as the highest priority.
  - if a particular station has frames of highest priority it will first send those frames and then other station will send the lower priority station

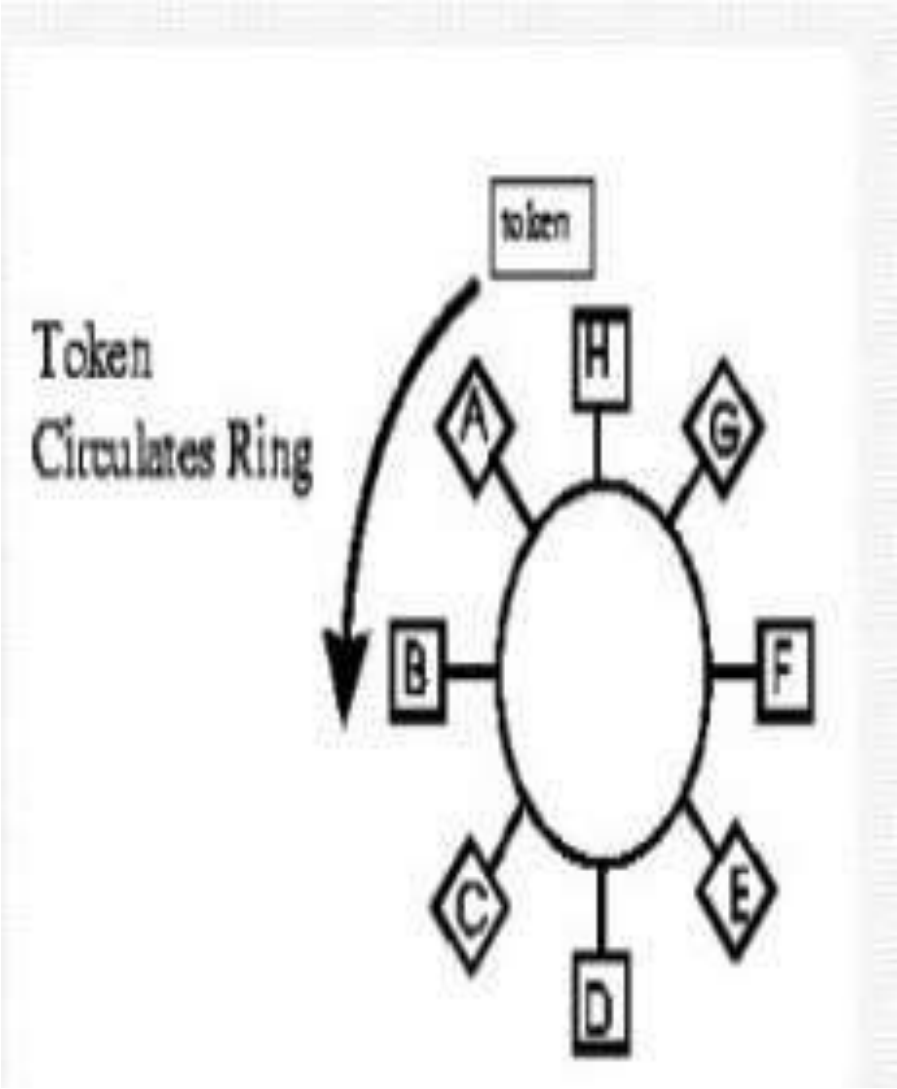
Token Bus Frame Format



Token Bus Control Frames		
Frame control field	Name	Meaning
00000000	Claim_token	Claim token during ring initialisation
00000001	Solicit_successor_1	Allow stations to enter the ring
00000010	Solicit_successor_2	Allow stations to enter the ring
00000011	Who_follows	Recover from lost token
00000100	Resolve_contention	Used when multiple stations want to enter the ring
00001000	Token	Pass the token
00001100	Set_successor	Allow stations to leave the ring

802.5 Token Ring

- Shared media network
- A ring consists of a set of nodes connected in a ring.
- Data always flows in particular direction around the ring, with each node receiving frame from its upstream neighbor & then forwarding to its downstream neighbor.
- Early forms of ring network were all token rings

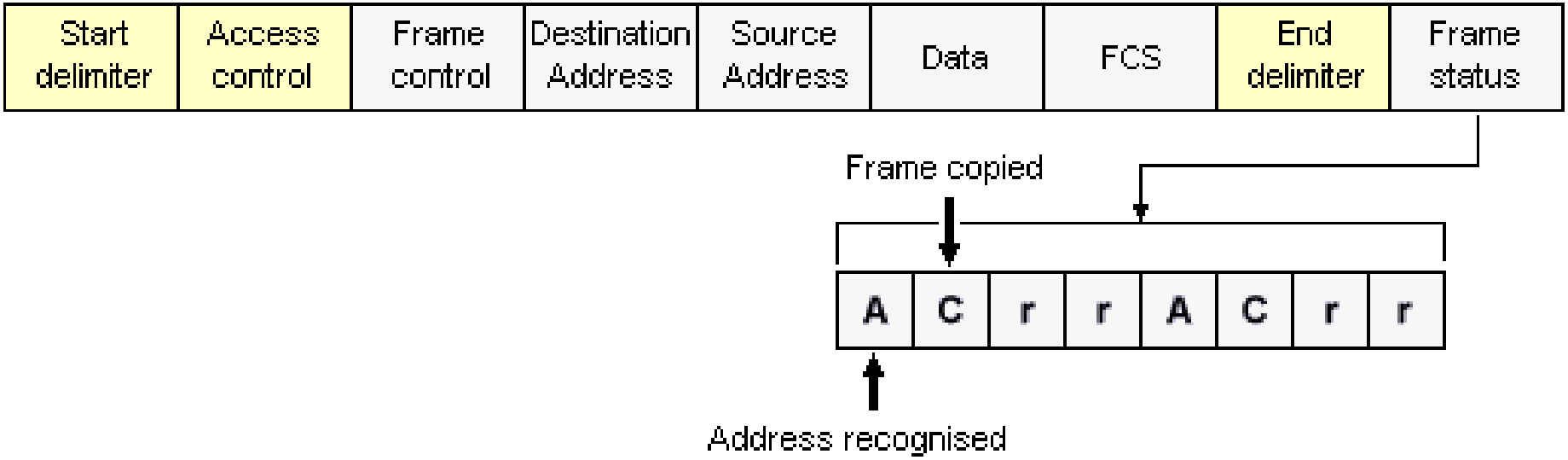

Token
Circulates Ring
token
A
H
G
B
F
C
E
D

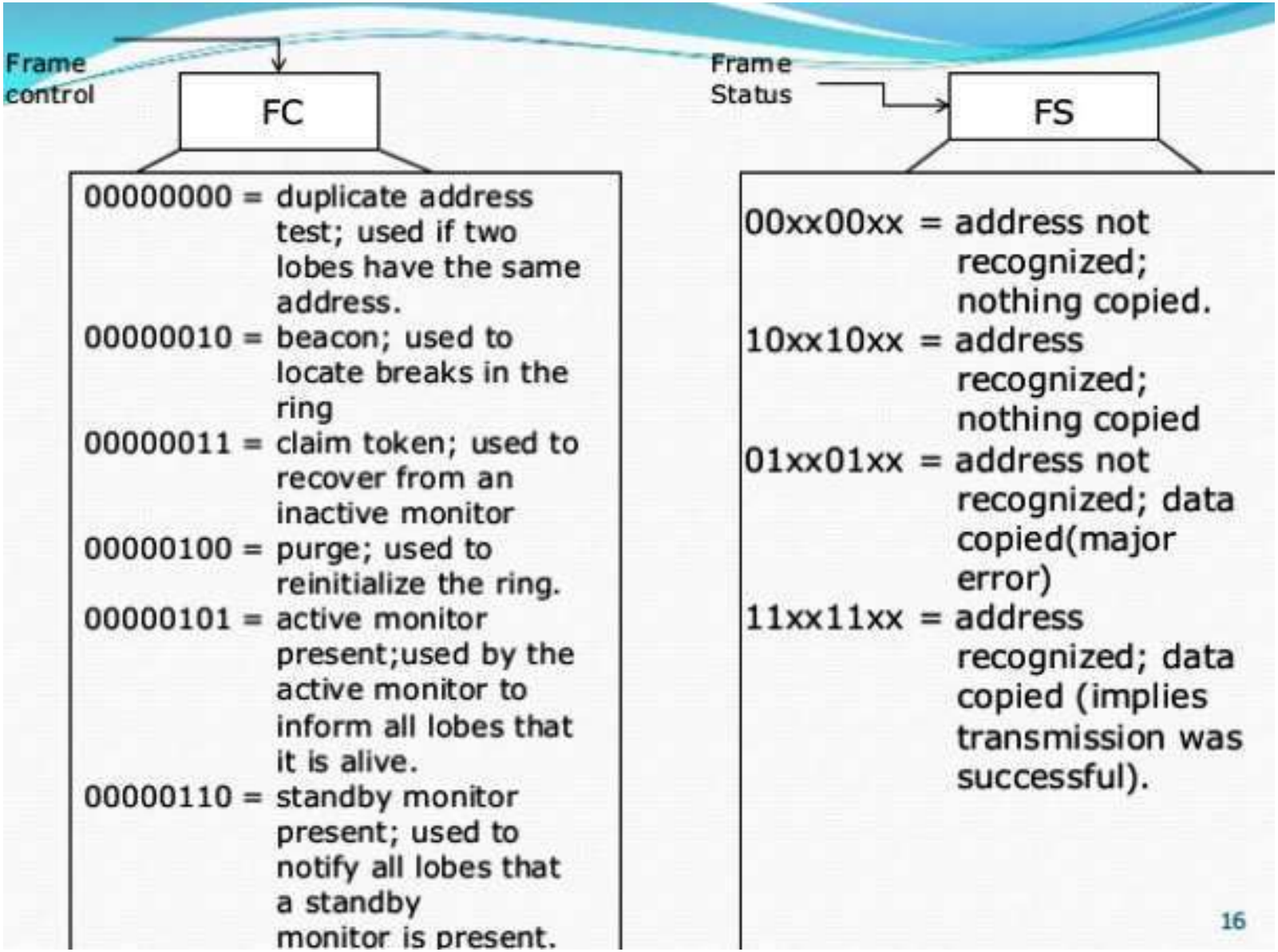
- token – special sequence of bits, circulates around the ring
- When a node that has a frame to transmit sees the token, it takes the token off the ring & inserts its frame into the ring.

- Each node simply forwards the frame, destination node saves a copy & forwarding onto the next node in the ring

- On the way back to sender, this node strips its frame off the ring & reinserts token.

- Nodes are serviced in Round Robin fashion
- Any link/node failure would render the whole network useless
  - Addressed by connecting each station into the ring using an electromechanical relay.
  - Several relays are usually packed into a single box , MULTI-STATION ACCESS UNITS.


Start delimiter
Access control
Frame control
Destination Address
Source Address
Data
FCS
End delimiter
Frame status
Frame copied
A C r r A C r r
Address recognised


Frame control
FC
00000000 = duplicate address test; used if two lobes have the same address.
00000010 = beacon; used to locate breaks in the ring
00000011 = claim token; used to recover from an inactive monitor
00000100 = purge; used to reinitialize the ring.
00000101 = active monitor present;used by the active monitor to inform all lobes that it is alive.
00000110 = standby monitor present; used to notify all lobes that a standby monitor is present.
Frame Status
FS
00xx00xx = address not recognized; nothing copied.
10xx10xx = address recognized; nothing copied
01xx01xx = address not recognized; data copied(major error)
11xx11xx = address recognized; data copied (implies transmission was successful).

- Starting delimiter and ending delimiter mark the beginning & ending of the frame.
- Access control consist of token bit, monitor bit, priority bit.
- Destination address & source address fields gives the address.
- Checksum field is used to detect transmission errors.

Comparison of 802.3, 802.4 and 802.5

Function	CSMA/CD	Token bus	Token ring
Access determination	Contention	Token	Token
Packet length restriction	64 bytes (Greater than 2XTprop)	None	None
Priority	Not supported	Supported	Supported
Sensitivity to work load	Most sensitive	Sensitive	Least sensitive
Principle advantage	Simplicity, wide installed base	Regulated/fair access	Regulated/fair access
Principle disadvantage	Nondeterministic delay	Complexity	Complexity

Wireless

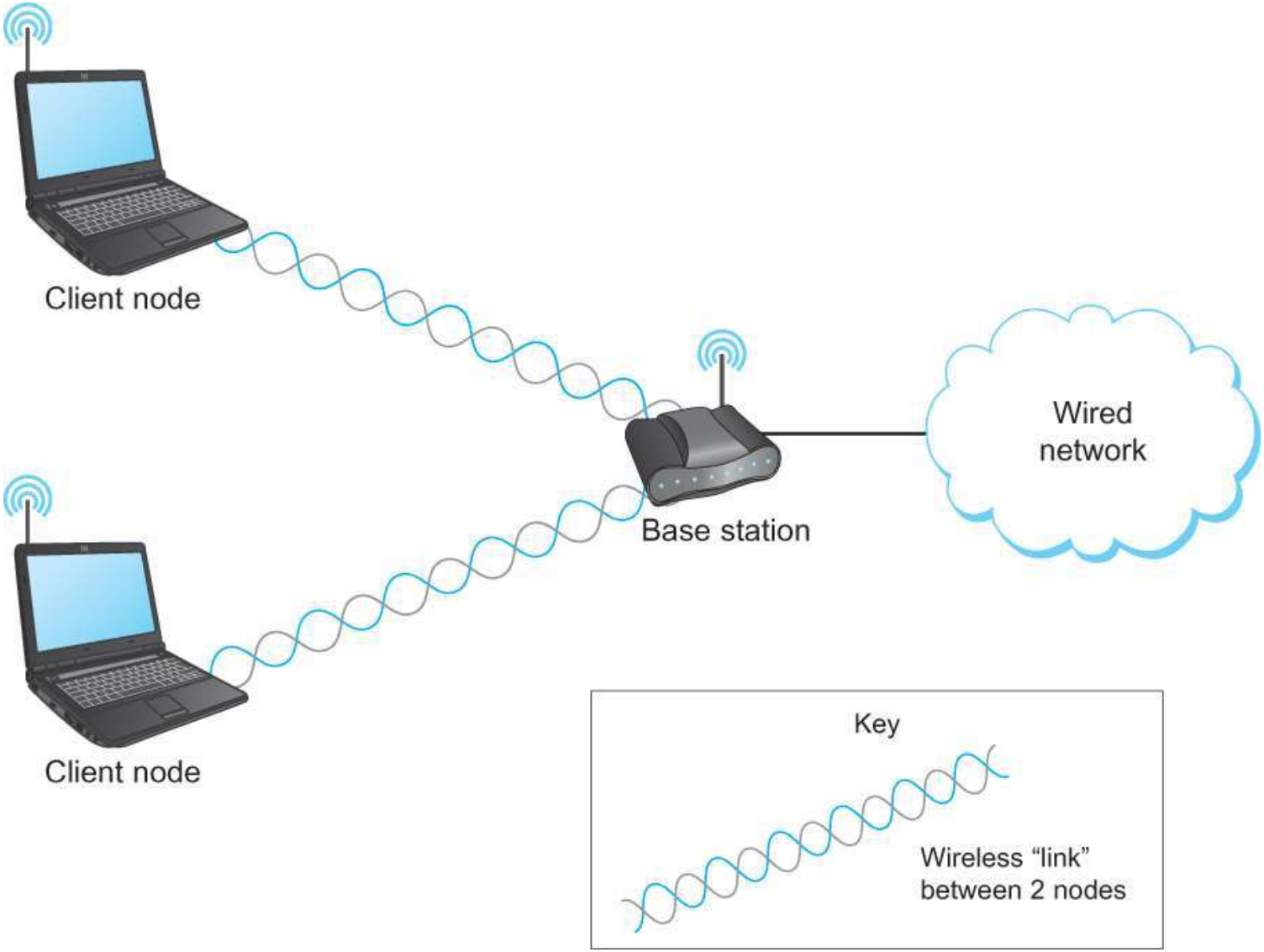
- Wireless links transmit electromagnetic signal
  - Radio, microwave, infrared
- Wireless links all share the same “wire” (so to speak)
  - The challenge is to share it efficiently without unduly interfering with each other
  - Sharing is accomplished by dividing the “wire” along the dimensions of frequency and space
- Exclusive use of a particular frequency in a particular geographic area may be allocated to an individual entity such as a corporation

- Wireless technologies differ in a variety of dimensions
  - How much bandwidth they provide
  - How far apart the communication nodes can be
- Four prominent wireless technologies
  - Bluetooth
  - Wi-Fi (more formally known as 802.11)
  - WiMAX (802.16)
  - 3G cellular wireless

	Bluetooth (802.15.1)	Wi-Fi (802.11)	3G Cellular
Typical link length	10 m	100 m	Tens of kilometers
Typical data rate	2 Mbps (shared)	54 Mbps (shared)	Hundreds of kbps (per connection)
Typical use	Link a peripheral to a computer	Link a computer to a wired base	Link a mobile phone to a wired tower
Wired technology analogy	USB	Ethernet	DSL

Overview of leading wireless technologies

- Mostly widely used wireless links today are usually asymmetric
  - Two end-points are usually different kinds of nodes
    - One end-point usually has no mobility, but has wired connection to the Internet (known as **base station**)
    - The node at the other end of the link is often mobile

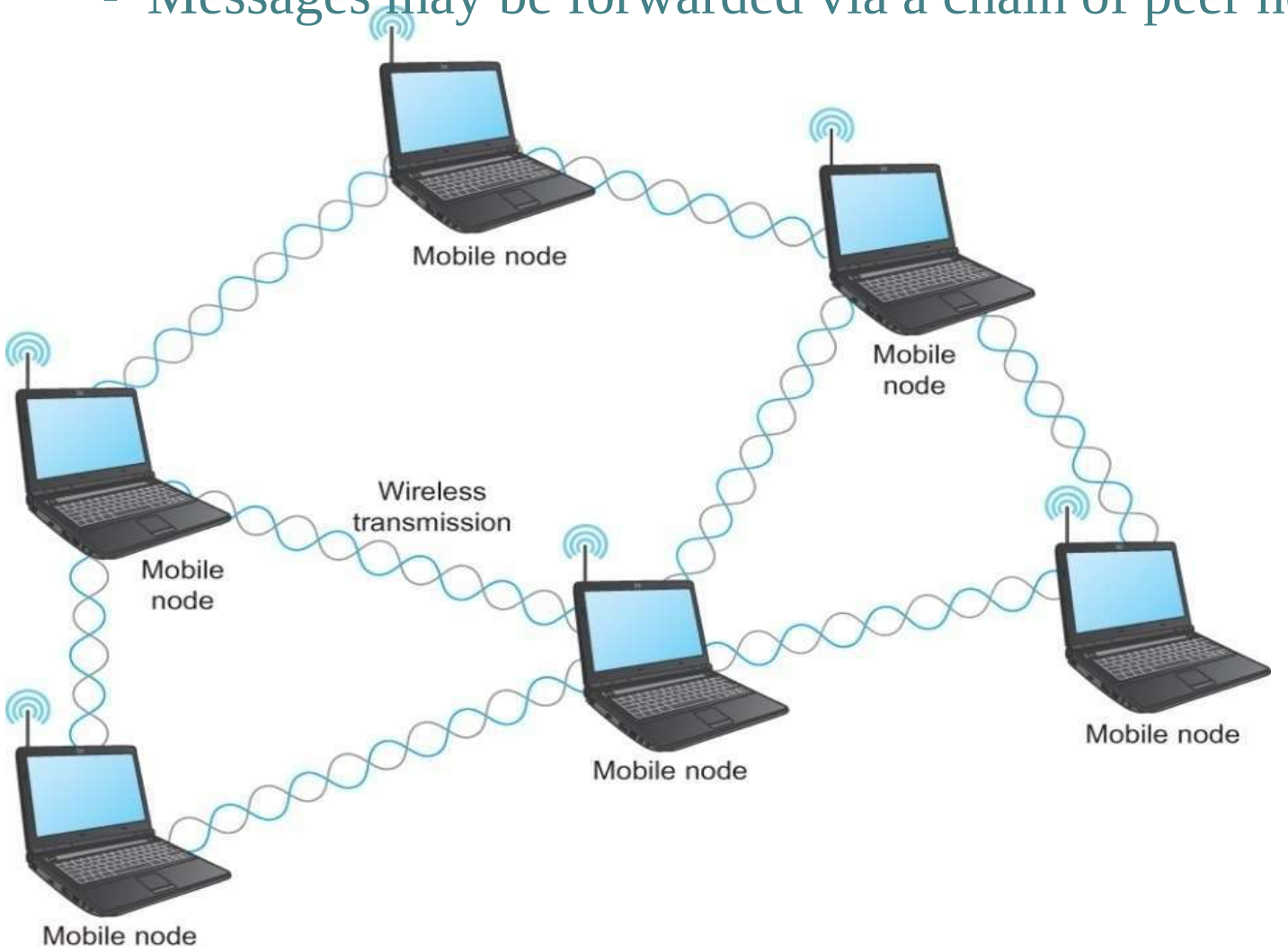


A wireless network using a base station

- Wireless communication supports point-to-multipoint communication
- Communication between non-base (client) nodes is routed via the base station
- Three levels of mobility for clients

1. No mobility: the receiver must be in a fix location to receive a directional transmission from the base station (initial version of **WiMAX**)
2. Mobility is within the range of a base (**Bluetooth**)
3. Mobility between bases (**Cell phones and Wi-Fi**)

- Mesh or Ad-hoc network
  - Nodes are peers
  - Messages may be forwarded via a chain of peer nodes



A wireless ad-hoc or mesh network

IEEE 802.11 (Wi-Fi)

- use in a **limited geographical area** (homes, office buildings, campuses)
  - Primary challenge is to mediate access to a shared communication medium – in this case, signals propagating through space
  - 802.11 is a set of IEEE standards that govern wireless networking transmission methods

- 802.11 supports additional features
  - Power management
  - security mechanisms

- Original 802.11 standard defined two radio-based physical layer standard
  - One using the frequency hopping
    - Over 79 1-MHz-wide frequency bandwidths
  - Second using direct sequence
    - Using 11-bit chipping sequence
  - Both standards run in the 2.4-GHz and provide up to 2 Mbps

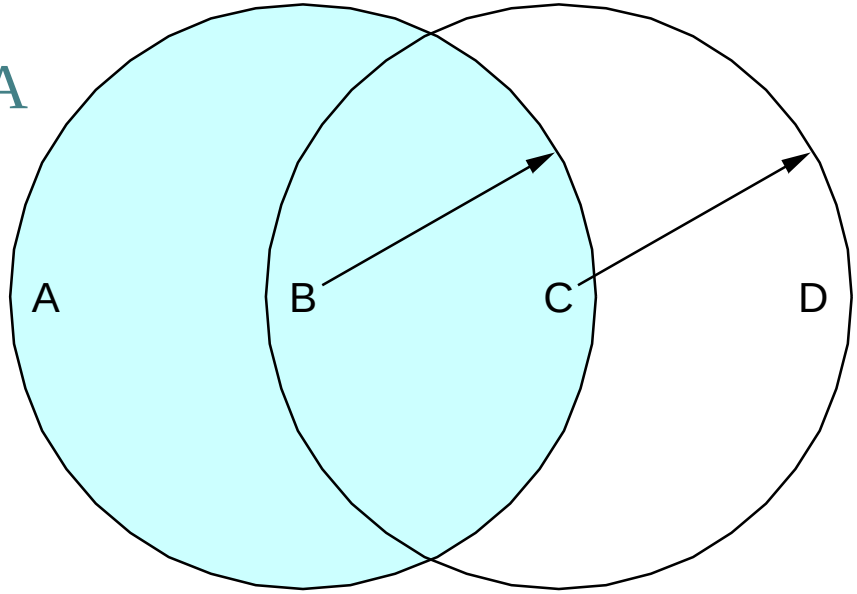
- Then physical layer standard **802.11b** was added
  - Using a variant of direct sequence 802.11b provides up to 11 Mbps
  - Uses license-exempt 2.4-GHz band
- Then came **802.11a** which delivers up to 54 Mbps
  - 802.11a runs on license-exempt 5-GHz band
  - Variant of FDM – Orthogonal frequency division multiplexing (OFDM)
- Most recent standard is **802.11g** which is backward compatible with 802.11b
  - Uses 2.4 GHz band, OFDM and delivers up to 54 Mbps

- **802.11n**, is a wireless-networking standard that uses multiple antennas to increase data rates.
  - MIMO, "multiple input and multiple output“
  - 54 Mbit/s to 600 Mbit/s
  - 802.11n extends the coexistence management to protect its transmissions from legacy devices, which include 802.11g, 802.11b and 802.11a.

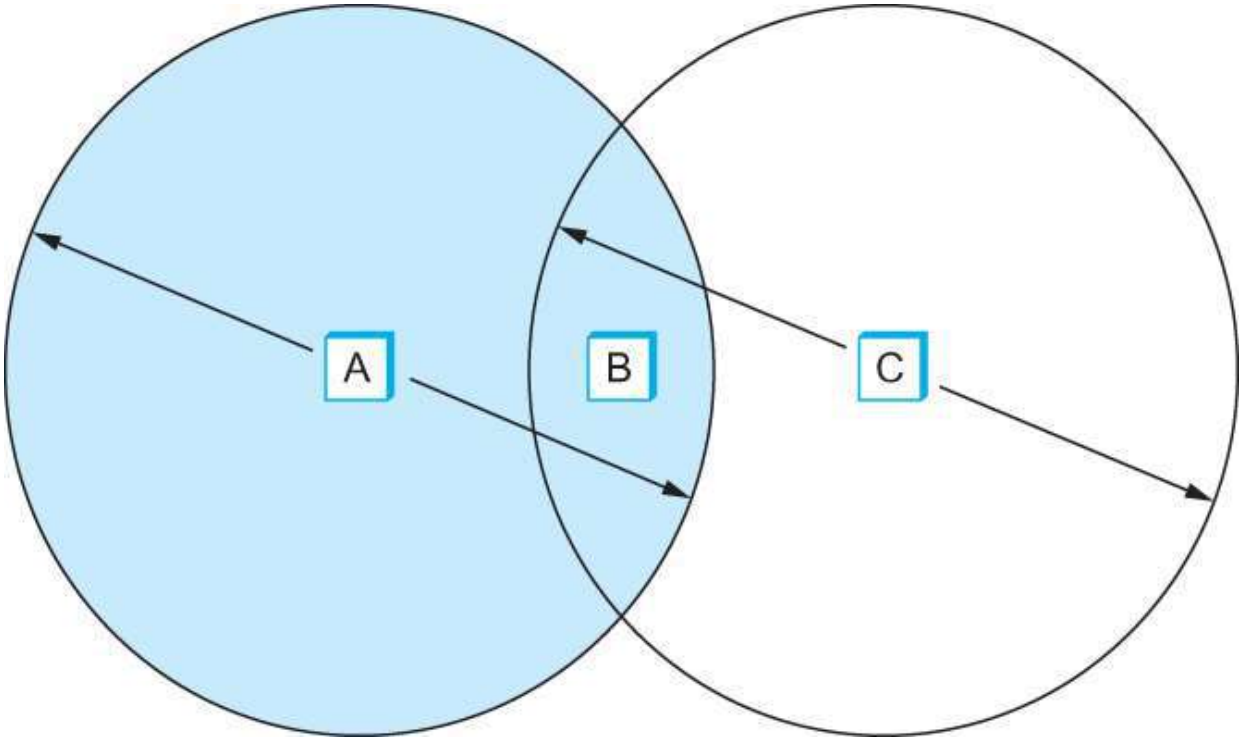
Wi-Fi– Collision Avoidance

- Consider the situation where each of 4 nodes is able to send and receive signals that reach just the nodes to its immediate left and right
  - For example, B can exchange frames with A and C, but it cannot reach D
  - C can reach B and D but not A

Eg of a wireless network



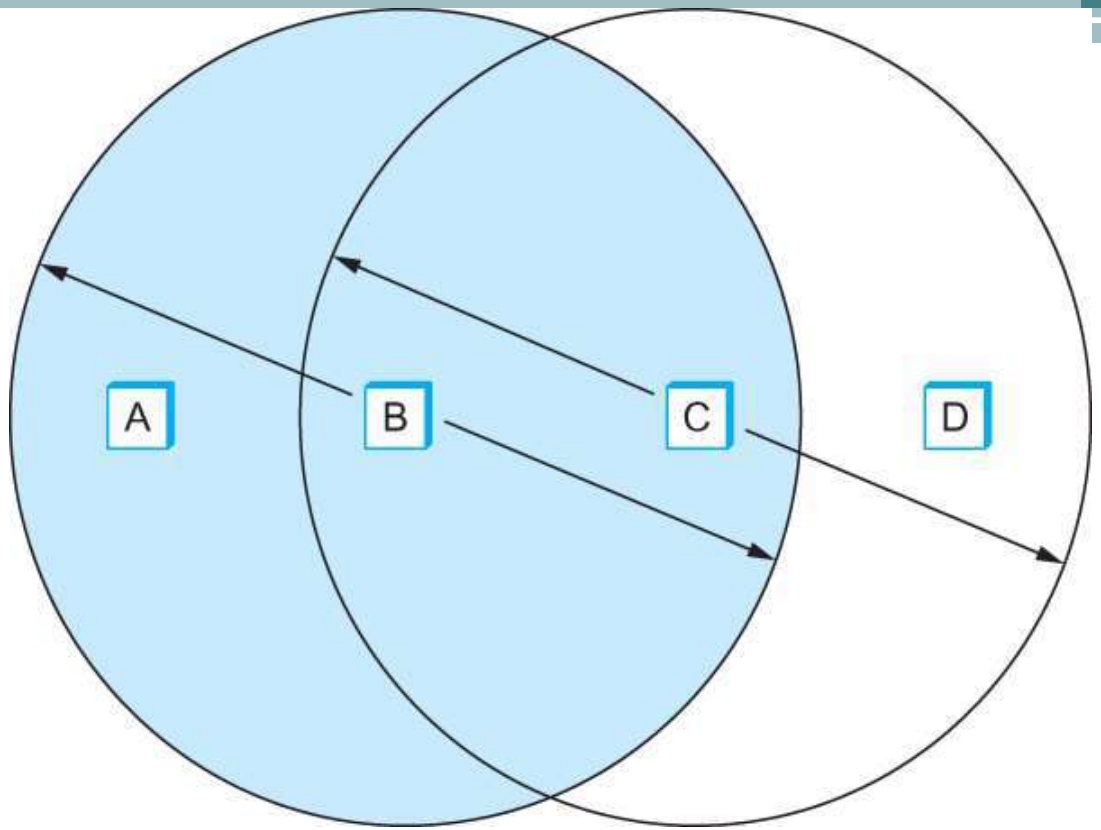
- Suppose both A and C want to communicate with B and so they each send it a frame.
  - A and C are unaware of each other since their signals do not carry that far
  - These two frames collide with each other at B
    - But unlike an Ethernet, neither A nor C is aware of this collision
  - A and C are said to ***hidden nodes*** with respect to each other



- The “Hidden Node” Problem.
- Although A and C are hidden from each other, their signals can collide at B.
- B’s reach is not shown.

Wi-fi Collision Avoidance

- Another problem called ***exposed node*** problem occurs :
  - Suppose B is sending to A. Node C is aware of this communication because it hears B's transmission.
  - It would be a mistake for C to conclude that it cannot transmit to anyone just because it can hear B's transmission.
  - Suppose C wants to transmit to node D.
  - This is not a problem since C's transmission to D will not interfere with A's ability to receive from B.



- Exposed Node Problem.
- Although B and C are exposed to each other's signals, there is no interference if B transmits to A while C transmits to D.
- A and D's reaches are not shown.

- 802.11 addresses these two problems with an algorithm called Multiple Access with Collision Avoidance (**MACA**).
- Key Idea
  - Sender and receiver exchange control frames with each other before the sender actually transmits any data.
  - This exchange informs all nearby nodes that a transmission is about to begin
  - Sender transmits a ***Request to Send*** (RTS) frame to the receiver.
    - The RTS frame includes a field that indicates how long the sender wants to hold the medium
      - Length of the data frame to be transmitted
  - Receiver replies with a ***Clear to Send*** (**CTS**) frame
    - This frame echoes this length field back to the sender

- Any node that sees the CTS frame knows that
  - it is close to the receiver, therefore
  - cannot transmit for the period of time it takes to send a frame of the specified length
- Any node that sees the RTS frame but not the CTS frame
  - is not close enough to the receiver to interfere with it
  - so is free to transmit

- Using ACK in MACA
  - Proposed in : MACA for Wireless LANs
- Receiver sends an **MACAW** ACK to the sender after successfully receiving a frame
- All nodes must wait for this ACK before trying to transmit
- If two or more nodes detect an idle link and try to transmit an RTS frame at the same time
  - Their RTS frame will collide with each other

- 802.11 does not support collision detection
  - So the senders realize the collision has happened when they do not receive the CTS frame after a period of time
  - In this case, they each wait a random amount of time before trying again.
  - The amount of time a given node delays is defined by the same **exponential backoff algorithm** used on the Ethernet.

Distribute Coordination Function (DCF)

- Uses CSMA/CA (*CSMA* with *C*ollision *A*voidance).
  - Uses one of two modes of operation:
    - *virtual carrier sensing*
    - physical carrier sensing

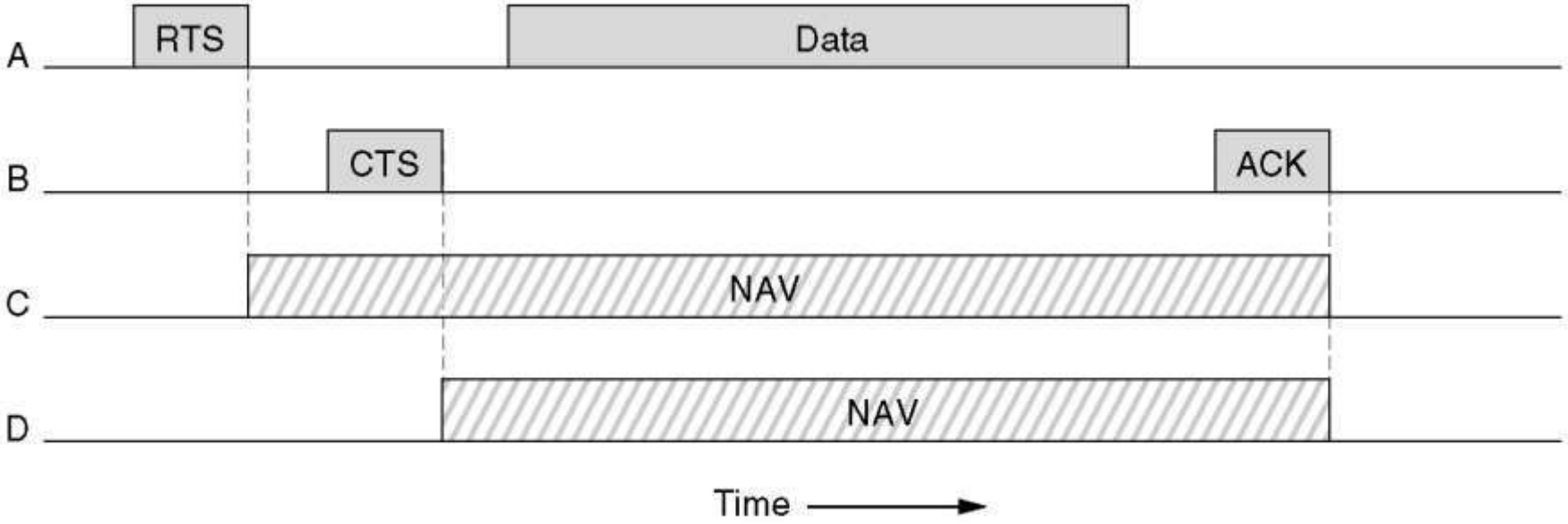
The two methods are supported:

1. MACAW (Multiple Access with Collision Avoidance for Wireless) with virtual carrier sensing

2. 1-persistent physical carrier sensing

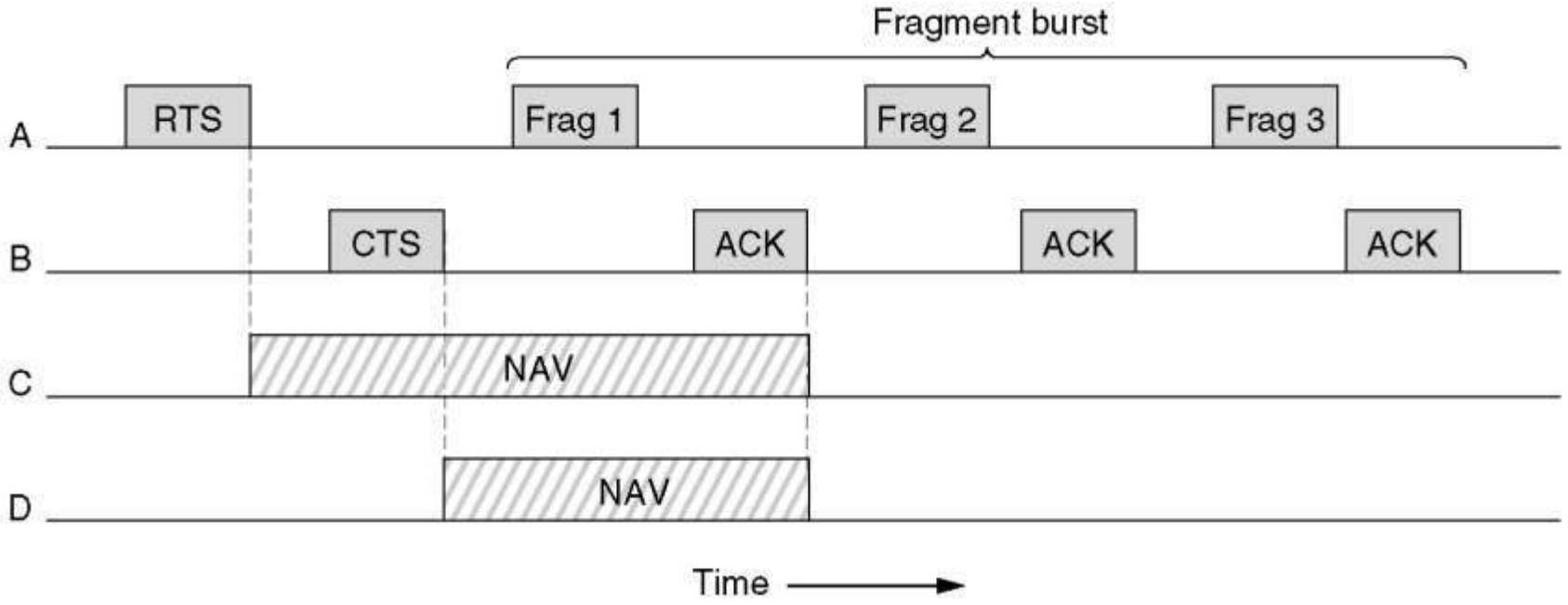
MACA protocol solved hidden and exposed terminal problems

Virtual Channel Sensing in CSMA/CA



The use of virtual channel sensing using CSMA/CA

- C (in range of A) receives the RTS and based on information in RTS creates a virtual channel busy NAV(Network Allocation Vector)
- D (in range of B) receives the CTS and creates a shorter NAV



- High wireless error rates ➔ long packets have less probability of being successfully transmitted.
- Solution: MAC layer fragmentation with stop-and-wait protocol on the fragments.

1-Persistent Physical Carrier Sensing

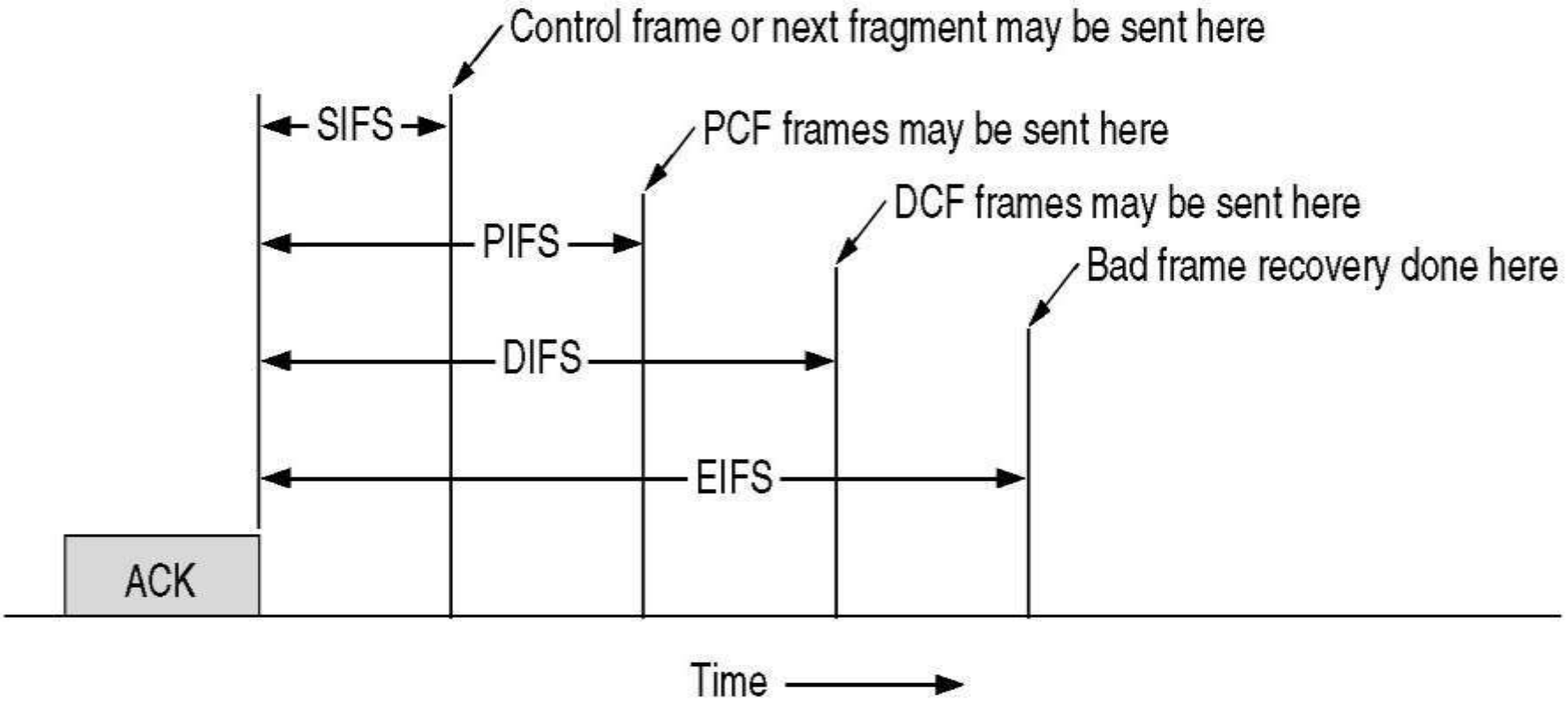
- The station senses the channel when it wants to send.
- If idle, the station transmits.
  - A station does not sense the channel while transmitting.
- If the channel is busy, the station defers until idle and then transmits (1-persistent).
- Upon collision, wait a random time using binary exponential backoff (BEB).

Point Coordinated Function (PCF)

- PCF uses a base station to poll other stations to see if they have frames to send
- No collisions occur
- Base station sends *beacon frame* periodically
- Base station can tell another station to *sleep* to save on batteries and base stations holds frames for sleeping station.

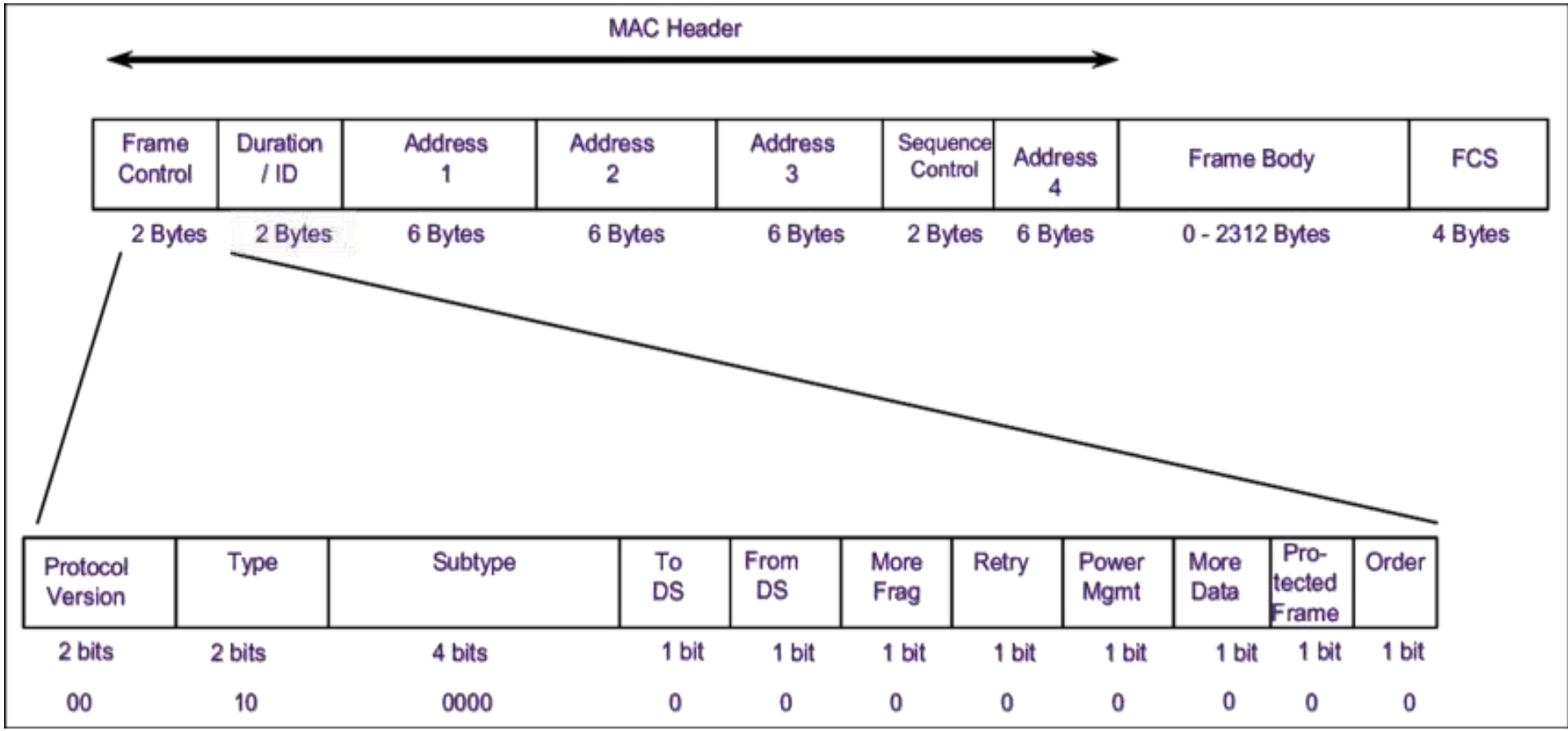
DCF and PCF Co-Existence

- Distributed and centralized control can co-exist using InterFrame Spacing.
- SIFS (Short IFS) is the time waited between packets in an ongoing dialog (RTS,CTS,data, ACK, next frame)
- PIFS (PCF IFS)-when no SIFS response, base station can issue beacon or poll.
- DIFS (DCF IFS)-when no PIFS, any station can attempt to acquire the channel.
- EIFS (Extended IFS)-lowest priority interval used to report bad or unknown frame.


Control frame or next fragment may be sent here
SIFS
PCF frames may be sent here
DCF frames may be sent here
PIFS
Bad frame recovery done here
DIFS
EIFS
ACK
Time

- To deal with this mobility and partial connectivity,
  - 802.11 defines additional structures on a set of nodes
  - Instead of all nodes being created equal,
    - some nodes are allowed to roam
    - some are connected to a wired network infrastructure
      - they are called **Access Points (AP)** and they are connected to each other by a so-called **distribution system**

802.11 data frame



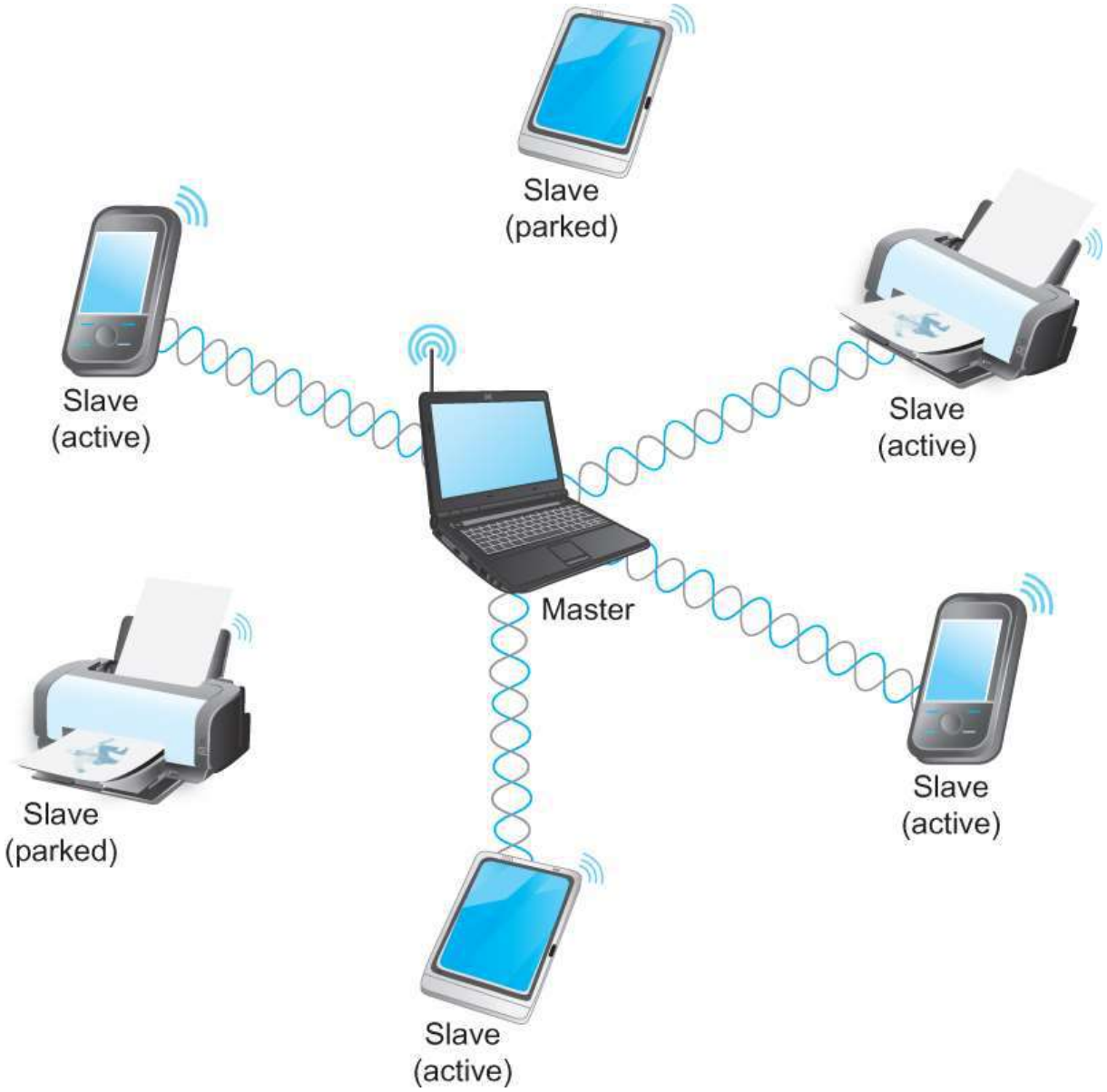
Bluetooth

- Used for very short range communication between mobile phones, PDAs, notebook computers and other personal or peripheral devices

- license-exempt band at 2.45 GHz, a range of only 10 m
- Communication devices typically belong to one individual or group
  - Sometimes categorized as Personal Area Network (PAN)
- Version 2.0 provides speeds up to 2.1 Mbps
- Power consumption is low

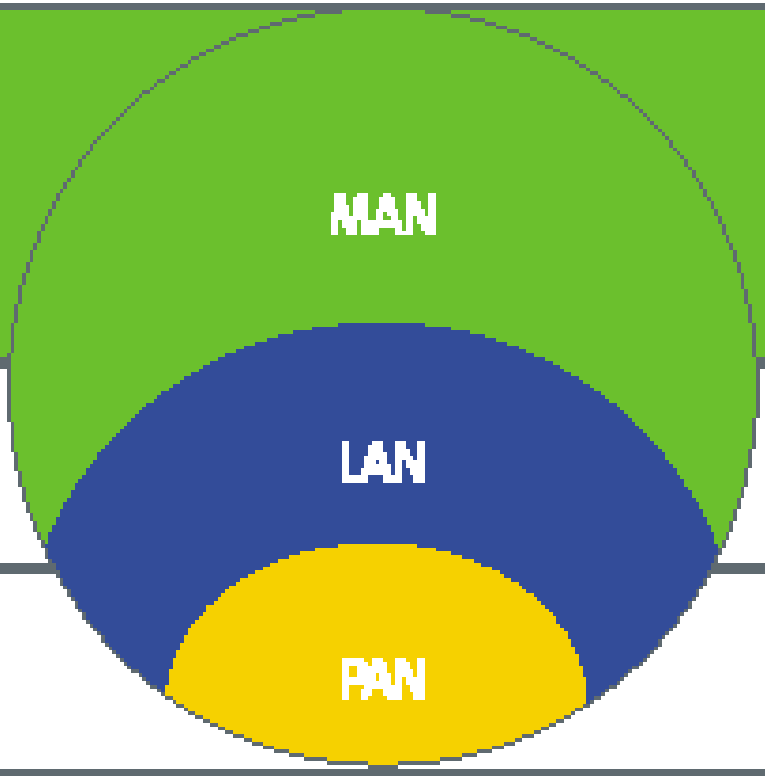
- Bluetooth is specified by an industry consortium called the Bluetooth Special Interest Group
- It specifies an entire suite of protocols, going beyond the link layer to define application protocols, which it calls **profiles**, for a range of applications
  - There is a profile for synchronizing a PDA with personal computer
  - Another profile gives a mobile computer access to a wired LAN

- The basic Bluetooth network configuration is called a **piconet**
  - Consists of a master device and up to seven slave devices
  - Any communication is between the master and a slave
  - The slaves do not communicate directly with each other
  - A slave can be **parked**:
    - set to an inactive
    - low-power state



A Bluetooth Piconet

WiMAX	802.16/HiperMAN
WiFi	802.11
Bluetooth	802.15



Wireless standards and their networking environments

- Hiperman 2-to 11-GHz
- WiBro 2.3 GHz