University of Southern California

Viterbi School of Engineering

EE450

Computer Networks

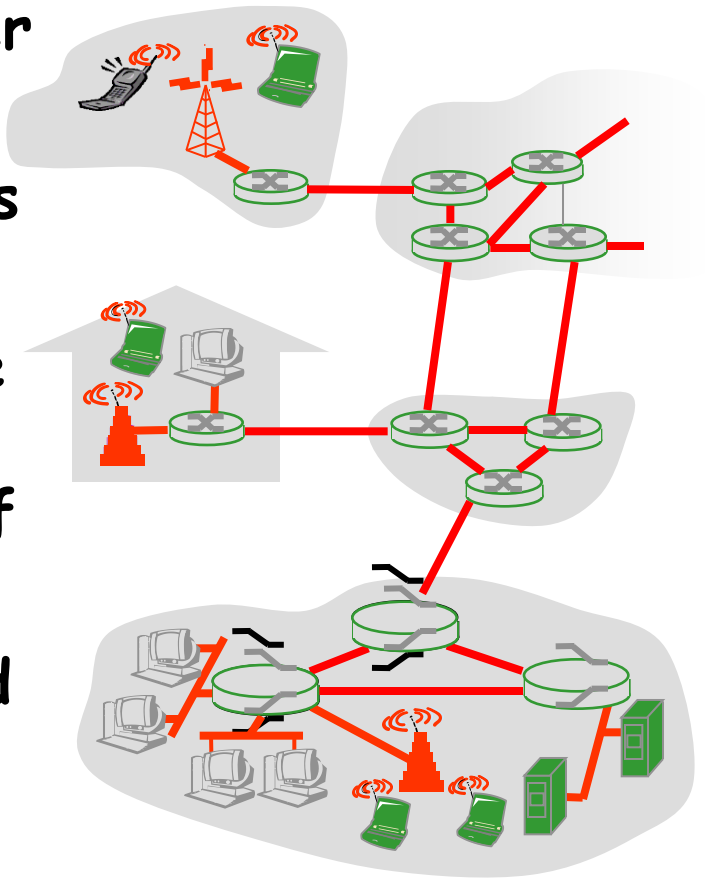
Data Link Layer

Shahin Nazarian

Spring 2013

Data Link Layer (DLL) – a Link to Link Protocol

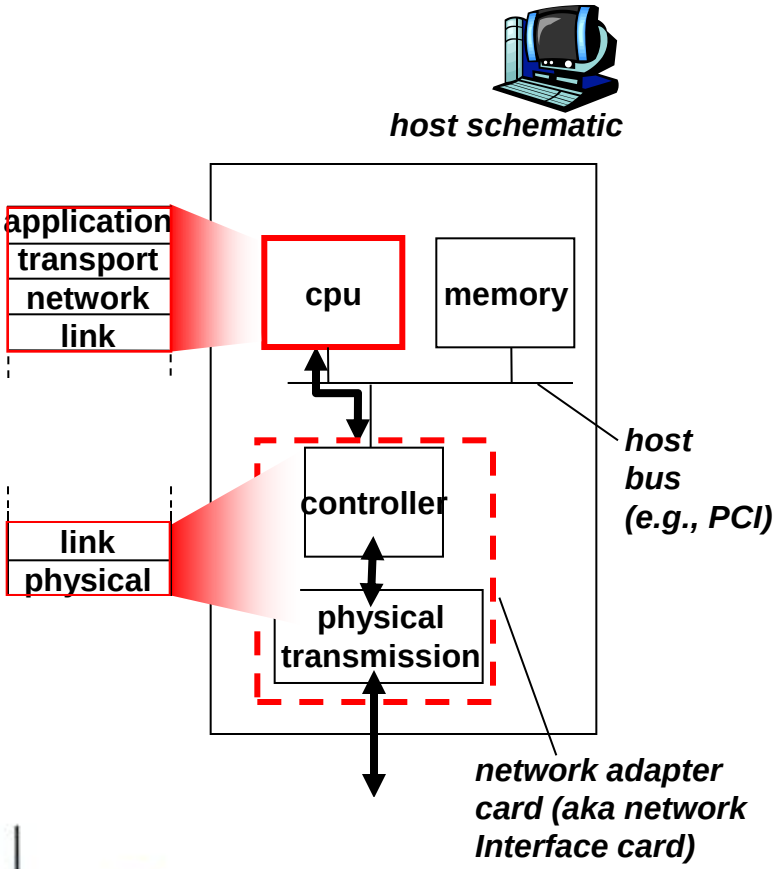
- Every layer provides a set of services and provides it to the layer above it
- DLL (Layer 2) is a link-to-link basis (not end-to-end basis) protocol
- Receiver side needs to synch frame (to know where it starts where it ends.) Receiver is the other side of the link
- Physical layer transmits the bit and makes sure no duplicate or missing bit exists, but DLL handles the errors (caused by noise or signal attenuation)



Where Is The Link Layer (LL) Implemented?

- LL is implemented in each and every node
- Much of LL controller's functionality is implemented in hardware
- LL is implemented in "adaptor" (aka *Network Interface Card* or NIC card)

Example: host link layer

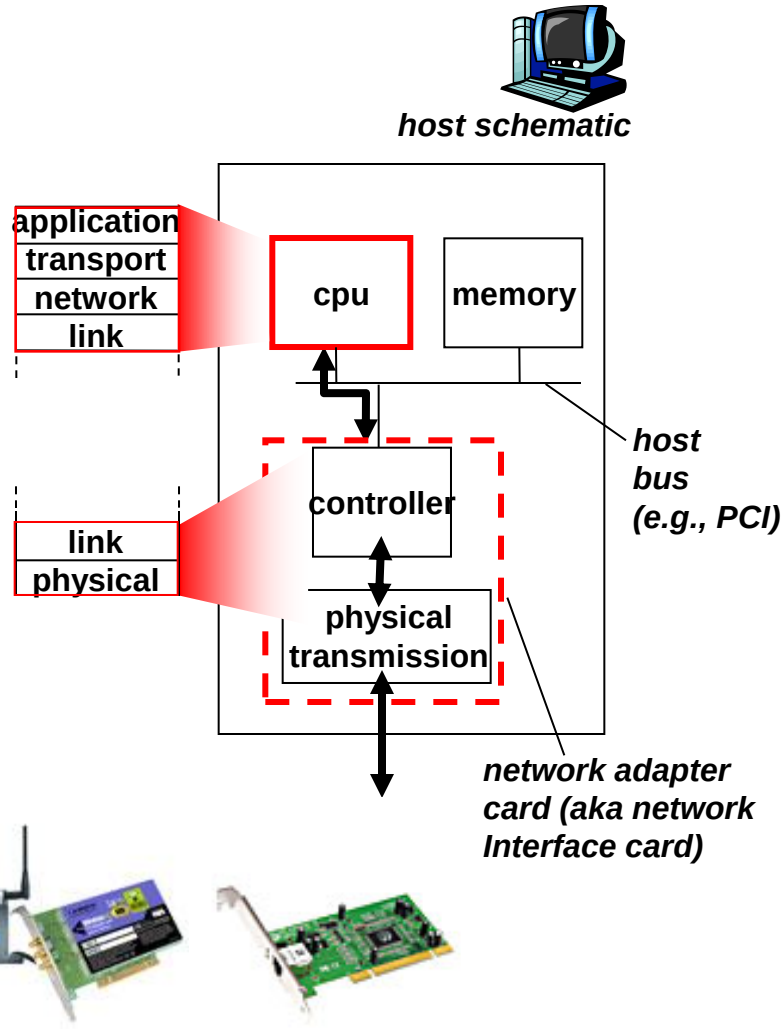


Where Is LL Implemented? (Cont.)

- E.g., Ethernet card (Intel's 8254x controller), PCM-CIA card (plug-in to PC's PCI card slot), 802.11 card (Atheros AR5006)
- Implements LL, Physical layer

- Attaches into node's system buses
- LL is a combination of hardware, software, and firmware

Example: host link layer



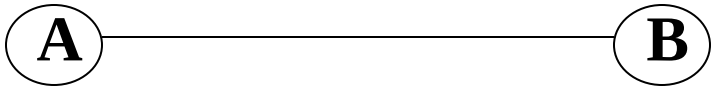
Some DLL Services

- Framing
  - Frame synchronization, adding header and trailer and encapsulating packet into frame
- Error detection, done by the receiver node, using parity checks, FCS (Frame Check Sequence,) etc.
- Flow control (to prevent buffer overflow at the receiver, i.e., to ensure the receiver is not overwhelmed by the sender)
  - A procedure that lets the receiver inform the sender to not overwhelm it by too many frames (so sender would slow down)
    - Stop & Wait
    - Sliding Window Protocol

Some DLL Services (Cont.)

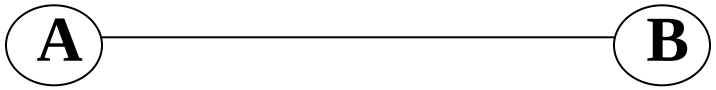
- **Error control**, done by receiver node, using one of the following (note that the transport layer performs error correction as well)
  - Receiver simply drops the frame [simplest approach]
    - In case error is detected, receiver will drop the frame, so automatically packet is dropped as well
  - **ARQ** (**Automatic Repeat reQuest**) protocols
    - Receiver will request from the sender (Note: not the end node, but the node on the other side of the link) to retransmit using ARQ
  - Error correction
    - Receiver is not only able to detect the error, but also to locate (i.e., correct) it
    - Most of the networking applications use dropping and ARQ instead of this technique

Point to Point DLC Protocols



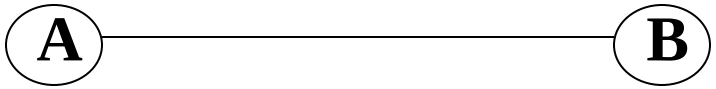
- One sender, one receiver, one link: easier than broadcast link:
  - No Media Access Control
  - No need for explicit MAC addressing
  - E.g., dialup link, ISDN line
- Popular point-to-point DLC protocols:
  - PPP (Point-to-Point Protocol)
  - HDLC (High-Level Data Link Control)
    - Data Link used to be considered "high layer" in protocol stack
    - HDLC is based on IBM's SDLC (Synch. DLC) protocol; and now can be used for both point-to-point and multipoint connections

PPP Data Link Control

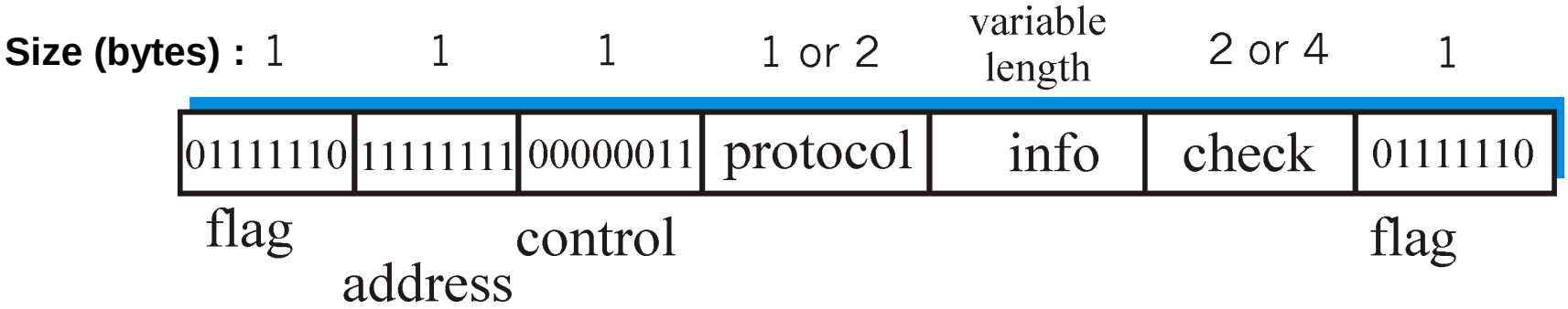


- Packet framing (encapsulating into frame) is performed
- Only error detection is performed, i.e., no error correction or error recovery
  - If a frame is corrupted it is silently dropped; the upper-layer protocol needs to take care of the problems
- No Flow control, i.e., the sender can send several frames one after another with no concern about overwhelming the receiver
  - Lack of error control and sequence numbering may cause a packet to be received out of order
- Error recovery, flow control, and data re-ordering are relegated to higher layers!

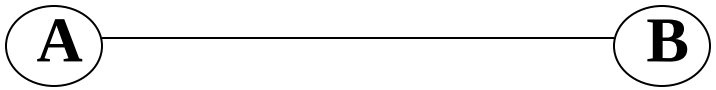
PPP Data Frame



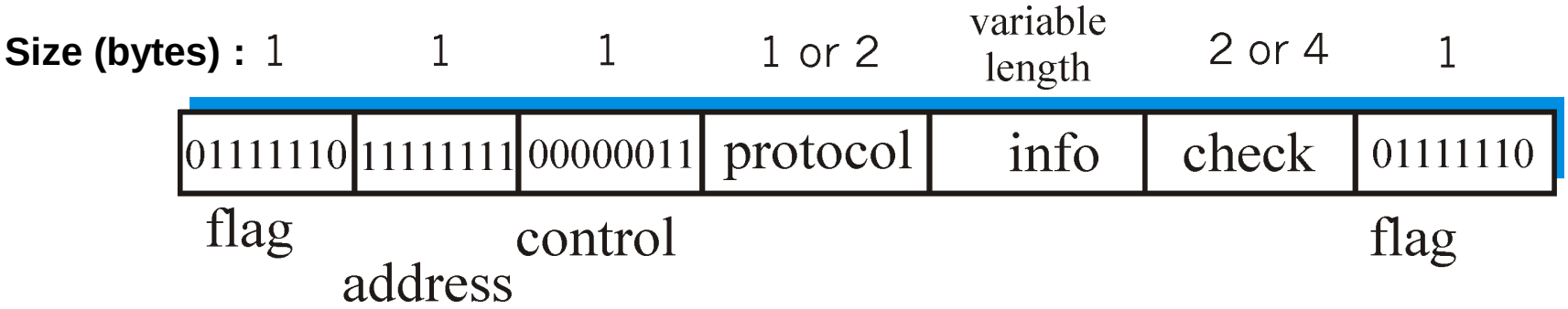
- **Flag:** delimiter, for framing (flag value is 01111110)
- **Address:** does nothing (the only possible value is 11111111)
- **Protocol:** upper layer protocol to which frame delivered, e.g., PPP-LCP (Link Control Protocol), IP, IPCP (Internet Protocol Control Protocol,) etc.
  - Upon receipt of a PPP frame, the PPP receiver will check the frame for correctness and then pass the encapsulated data on to the appropriate protocol



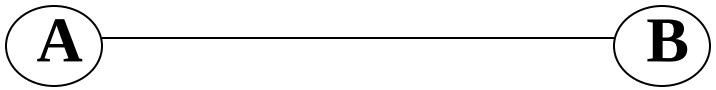
PPP Data Frame (Cont.)



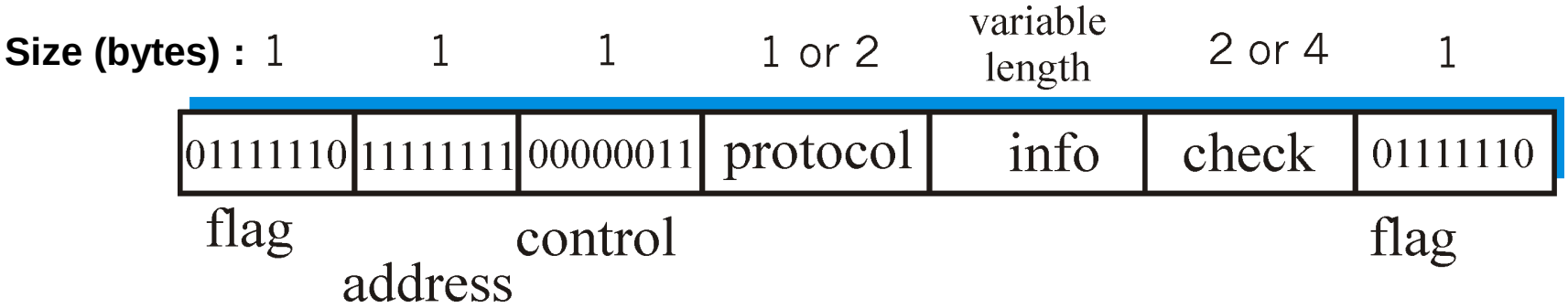
- **Control:** does nothing (the only possible value is 00000011)
- Address and control fields are saved for possible future use; currently PPP allows sender to not send them, saving 2 bytes of overhead
- **Check:** used to detect bit errors in a transmitted frame. It uses either a 2 or 4 byte HDLC-standard cyclic redundancy check (CRC)



PPP Data Frame (Cont.)



- **Info:** This part contains the encapsulated packet (data) that is being sent by an upper-layer protocol, e.g., IP over the PPP link
  - The default maximum length of the info field is 1500 bytes, although this can be changed when the link is first configured

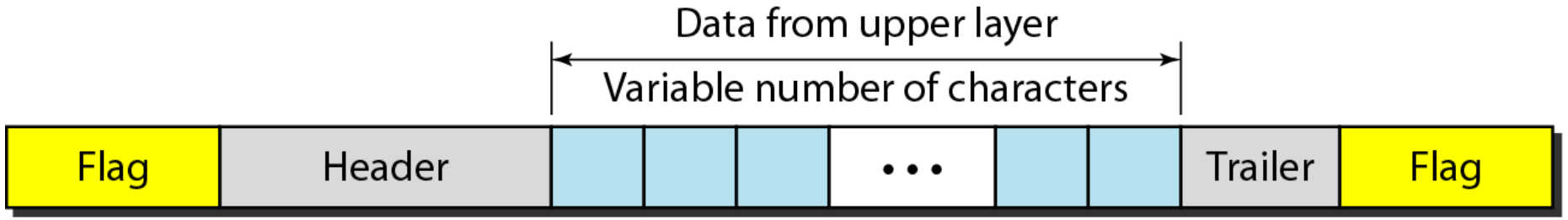


Framing

- Framing in DL layer separates a message from other messages
- If the whole message is packed into one frame, flow and error control become inefficient (e.g., a single-bit error, means retransmission of the whole message) therefore message is typically divided into smaller frames
- Fixed-size Framing: there is no need to define the boundaries of the frame; the size itself can be the delimiter. Ex: FR
- Variable-size Framing: need to define the end of the frame and beginning of the next using one of the two approaches: bit-oriented or byte-oriented

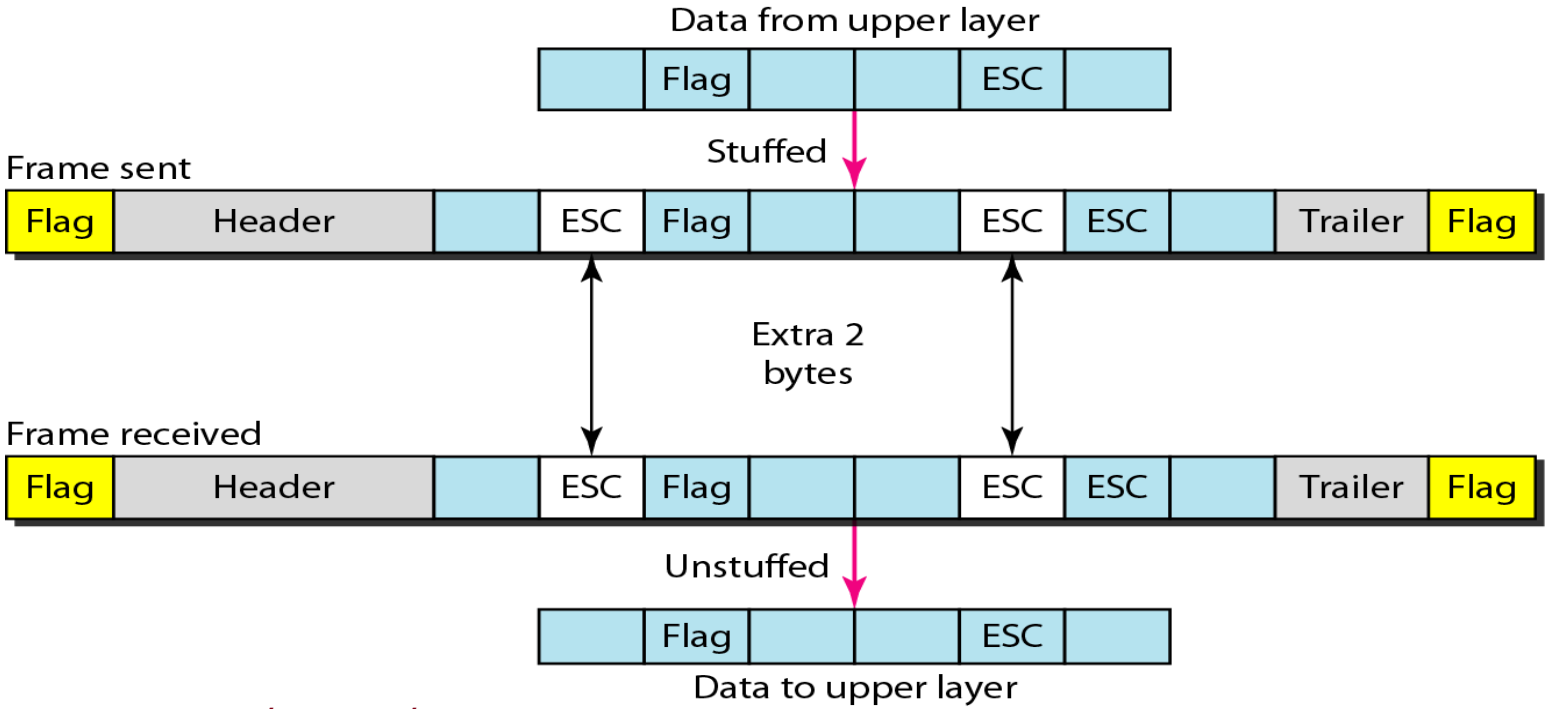
Byte-Oriented Protocols

- Data are 8-bit characters from a coding systems such as ASCII
- Header and trailer are multiples of 8 bits
- An 8-bit flag is added to the beginning and end of the frame to separate the frame



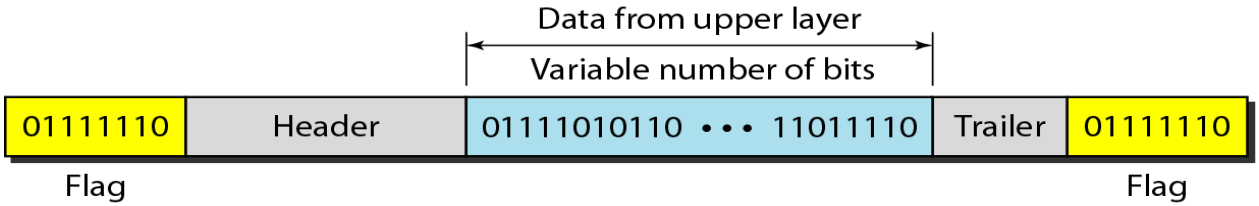
Byte-Oriented Protocols (Cont.)

- What if the frame itself contains that pattern?
- **Byte-stuffing**:
  - An escape character (ESC) is stuffed for each data part having the flag pattern
  - Frame will be unstuffed at the destination



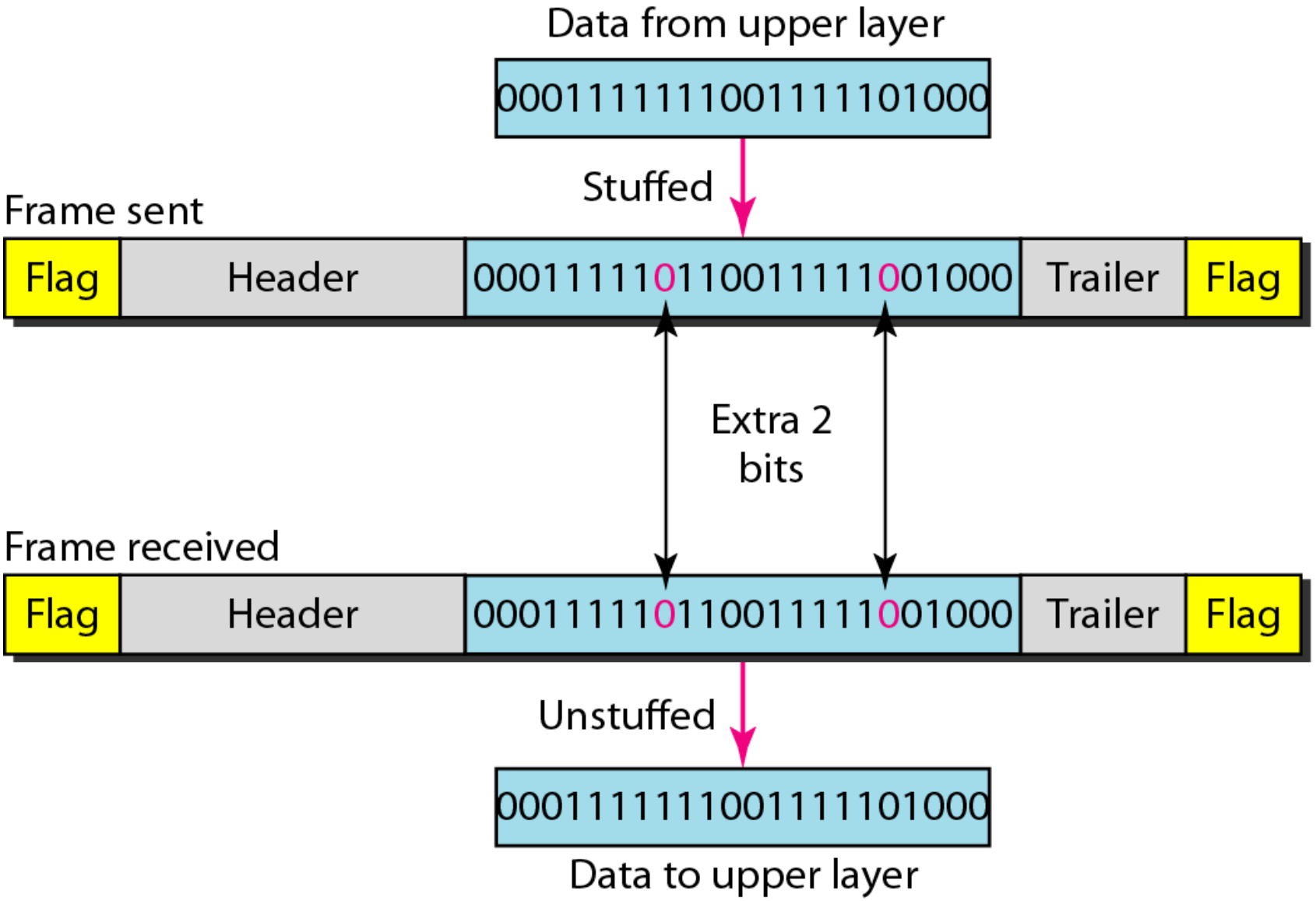
Bit-Oriented Protocols

- The data part of the frame is a sequence of bits. In addition to the header and the trailer, it needs a delimiter to separate one frame from another
  - Frame Synchronization (might be referred to as framing)
  - A special pattern, 01111110, as the delimiter is used to define the beginning and end for frame synchronization



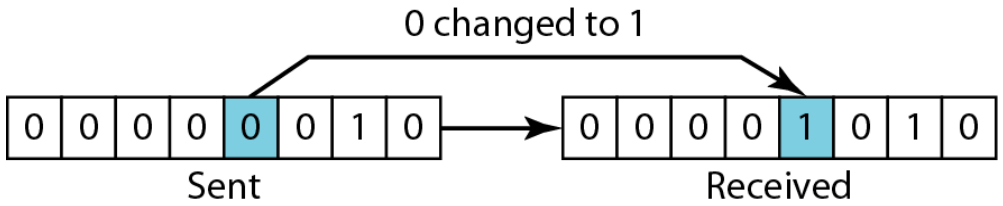
- Here, bit-stuffing is done to avoid confusion:
- Transmitter inserts an extra 0 after each consecutive five 1s *inside* the frame
- Receiver checks for five consecutive 1s
  - if next bit = 0, it is removed
  - if next two bits are 10, then flag is detected
  - If next two bits are 11, then frame has errors

Framing Example

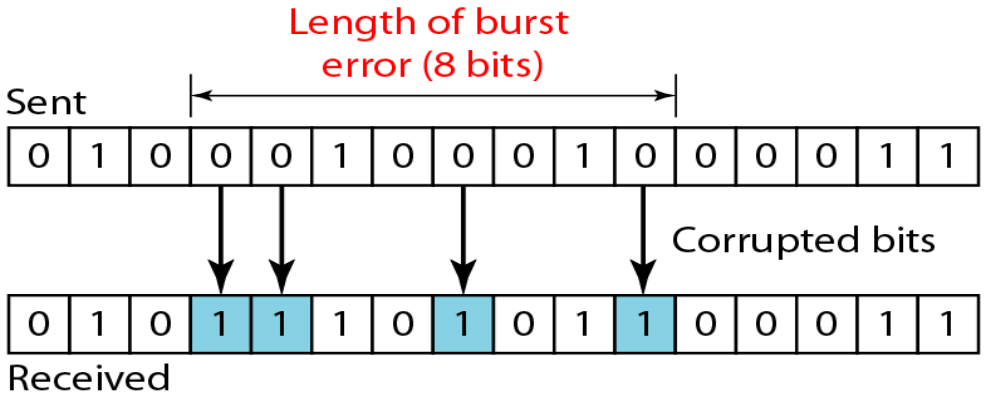


Error Detection

- **Single bit error**



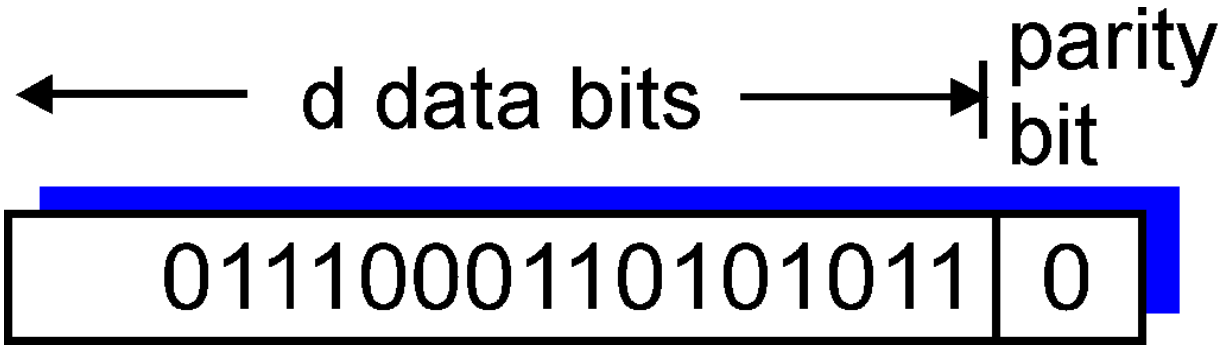
- **Burst error (i.e., 2 or more bits in the data unit have changed)**



- **To detect or correct errors, we need to send extra (redundant) bits with data in a frame, e.g.:**
  - **CRC or FCS (Frame Check Sequence) bits**
  - **Parity check bits: Odd (even) parity bit is chosen such that what is sent, i.e., data which is typically one character plus the parity bits together has odd (even) number of 1s in it**

Parity Checking – Single Bit Parity

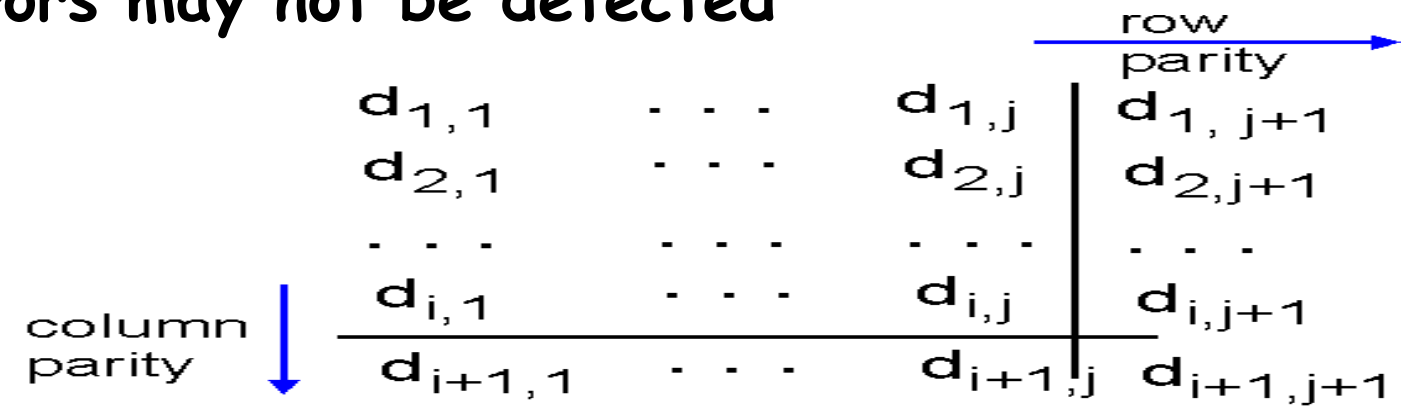
- Single Bit Parity detects odd number of errors

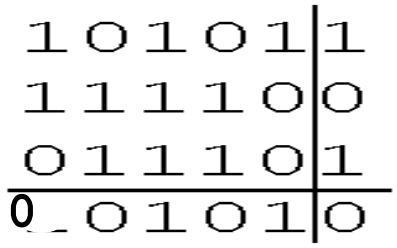


- Note: Efficiency in the above example is 16/17

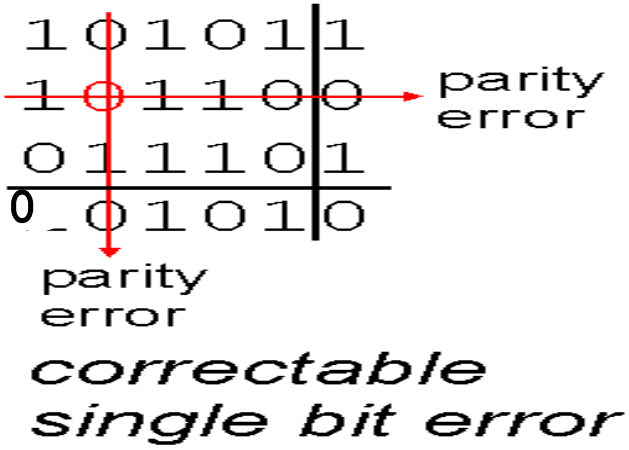
Parity Checking – 2D Bit Parity

- Two Dimensional Bit Parity detects *& corrects* 1 bit error, can detect up to 3 bit errors
- Four errors may not be detected



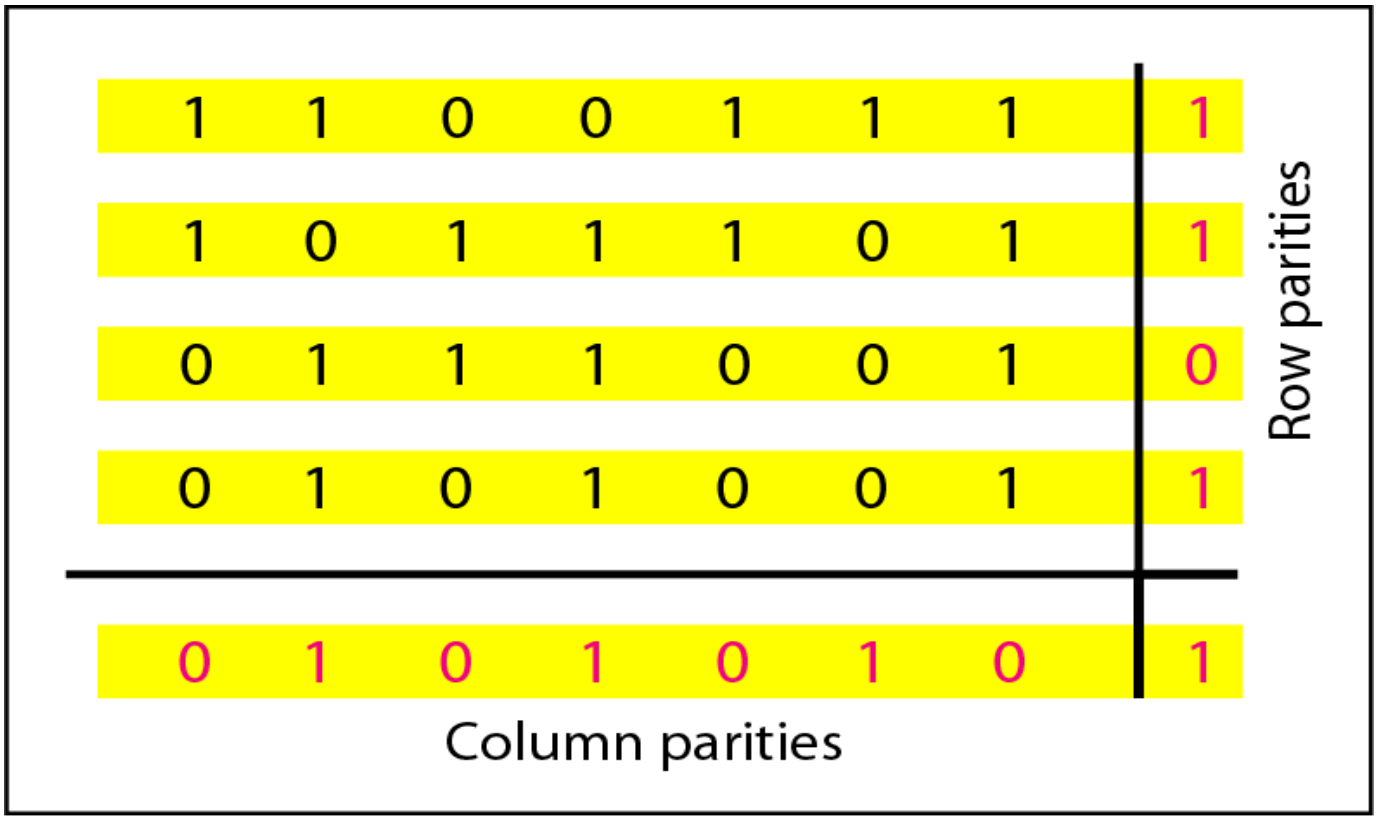


no errors



correctable single bit error

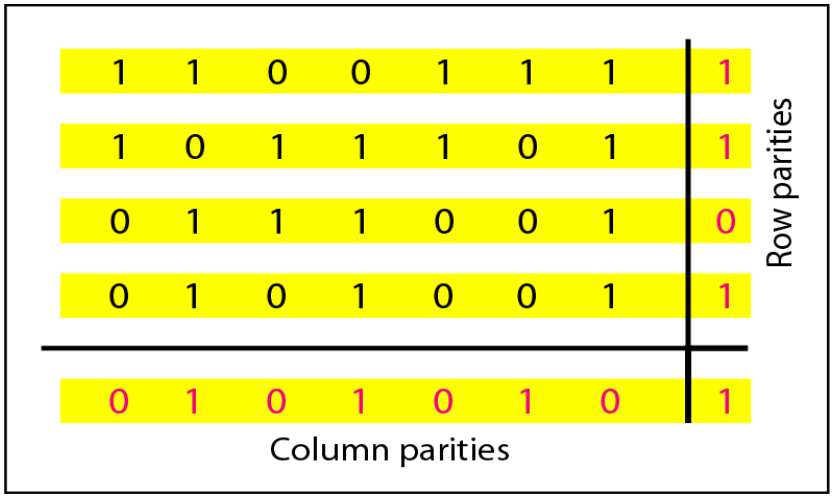
Parity Checking – 2D Bit Parity (Cont.)



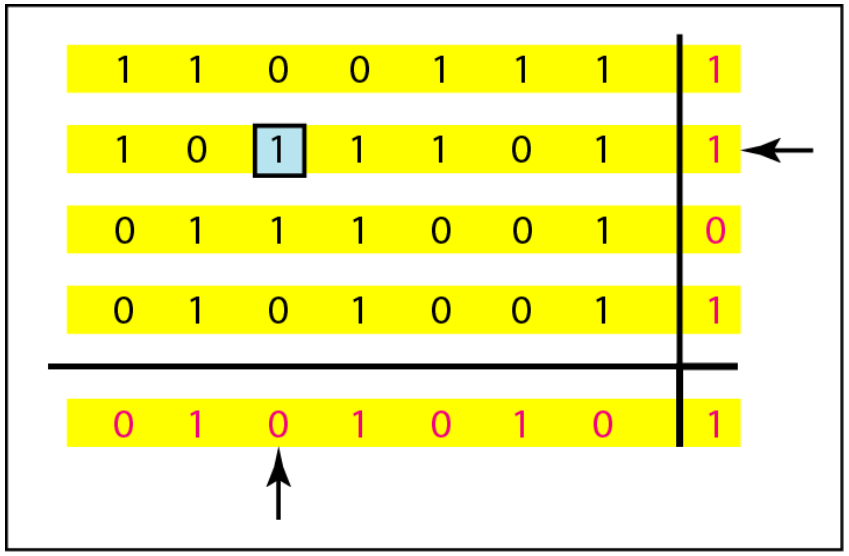
a. Design of row and column parities

- **Efficiency in the above example is 28/40**

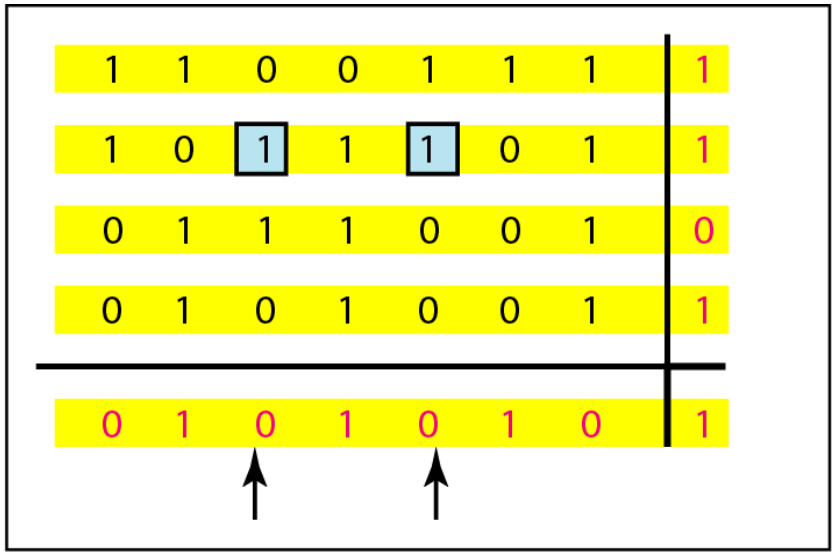
2D Bit Parity – Examples



a. Design of row and column parities

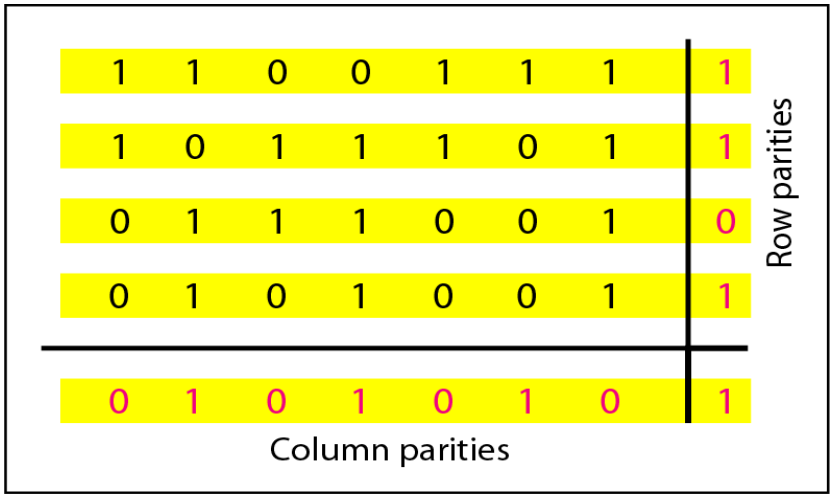


b. One error affects two parities

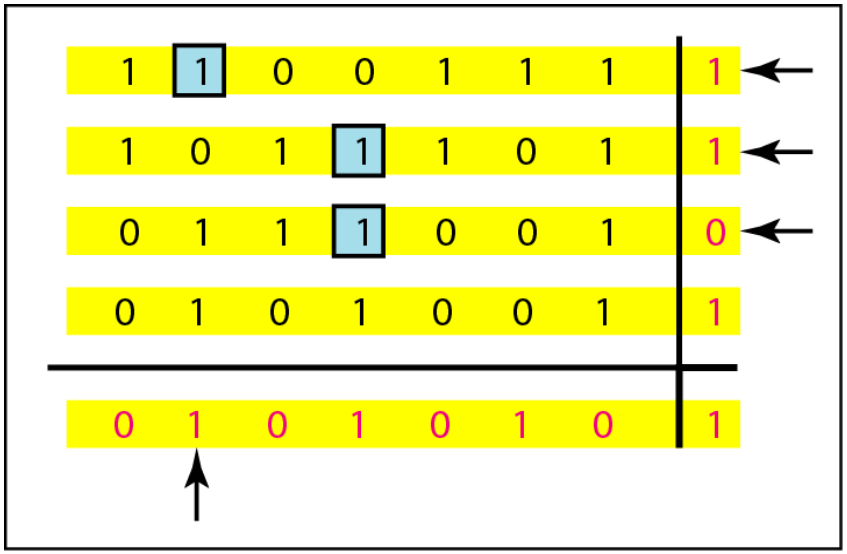


c. Two errors affect two parities

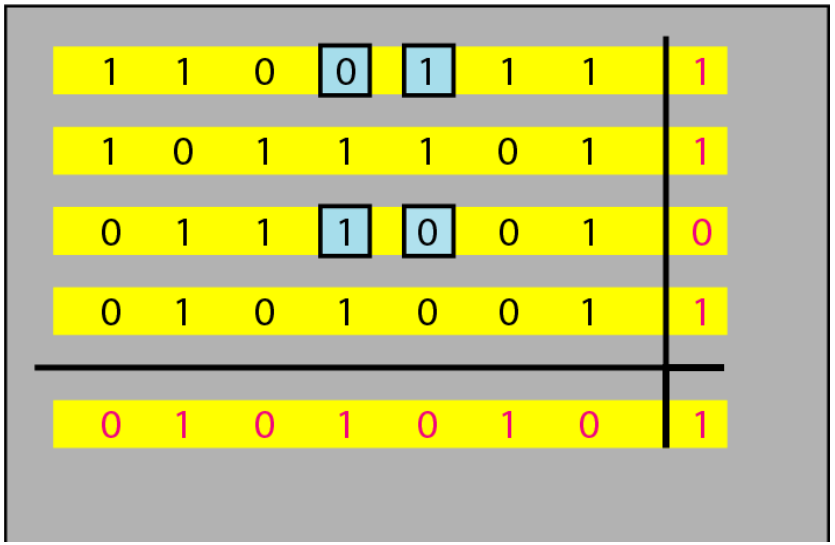
2D Bit Parity – Examples (Cont.)



a. Design of row and column parities



d. Three errors affect four parities

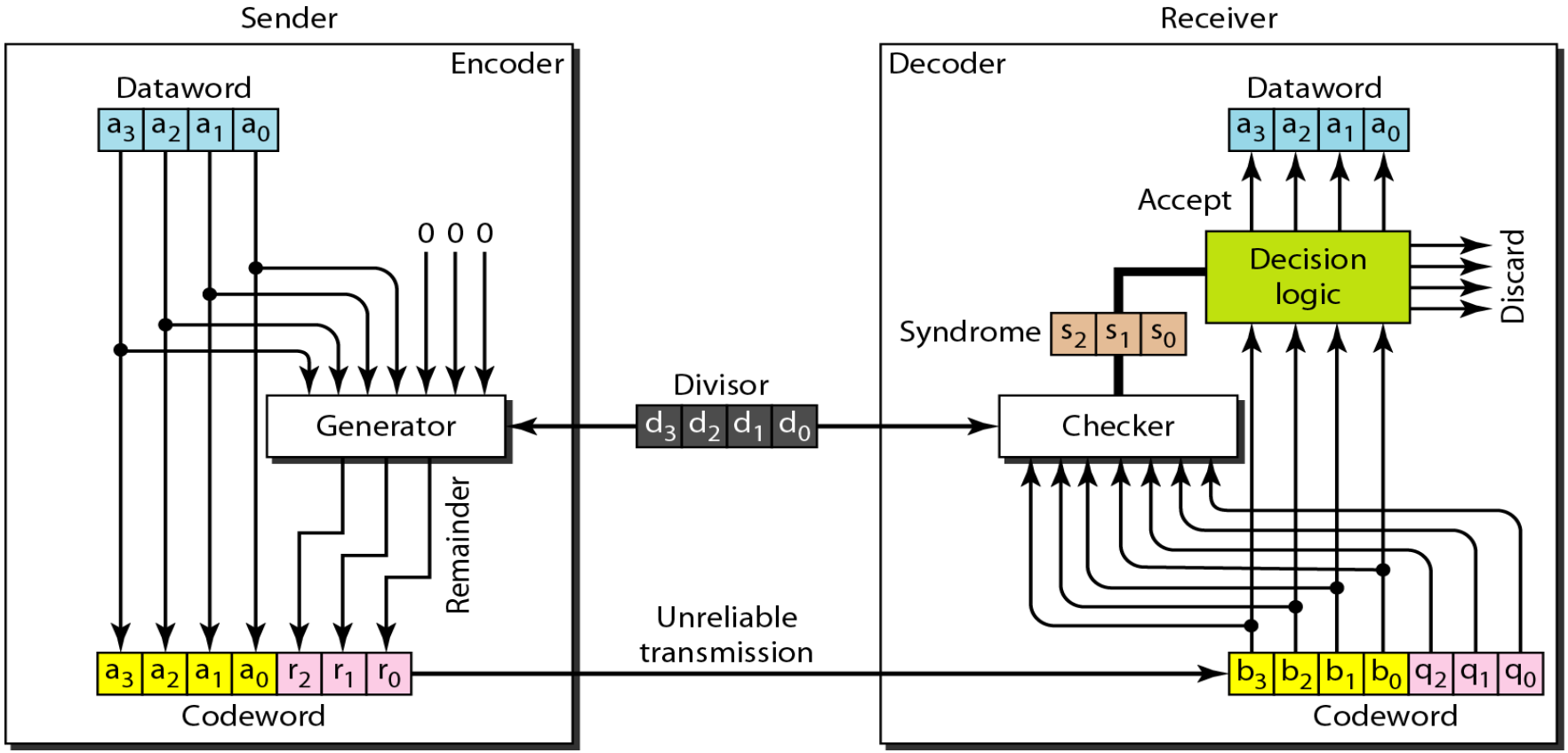


e. Four errors may not be detected

Cyclic Redundancy Check (CRC) or FCS

- View data word (i.e., data bits) as a binary number
- Choose r+1 bit pattern G (the divisor)
- Goal: choose r CRC bits, R, such that
  - Codeword, <D,R> is exactly divisible by G (modulo 2)
  - Receiver knows G, divides <D,R> by G. If non-zero remainder: error is detected!
  - Can detect all burst errors less than r+1 bits
- Widely used (Ethernet, 802.11 WiFi, ATM, etc.)
- All burst errors with $L \leq r$ will be detected
- (Optional to know) burst errors with length $L = r + 1$ will be detected with probability $1 - (0.5)^{r-1}$. Also burst errors with length $L > r + 1$ will be detected with probability $1 - (0.5)^{r}$

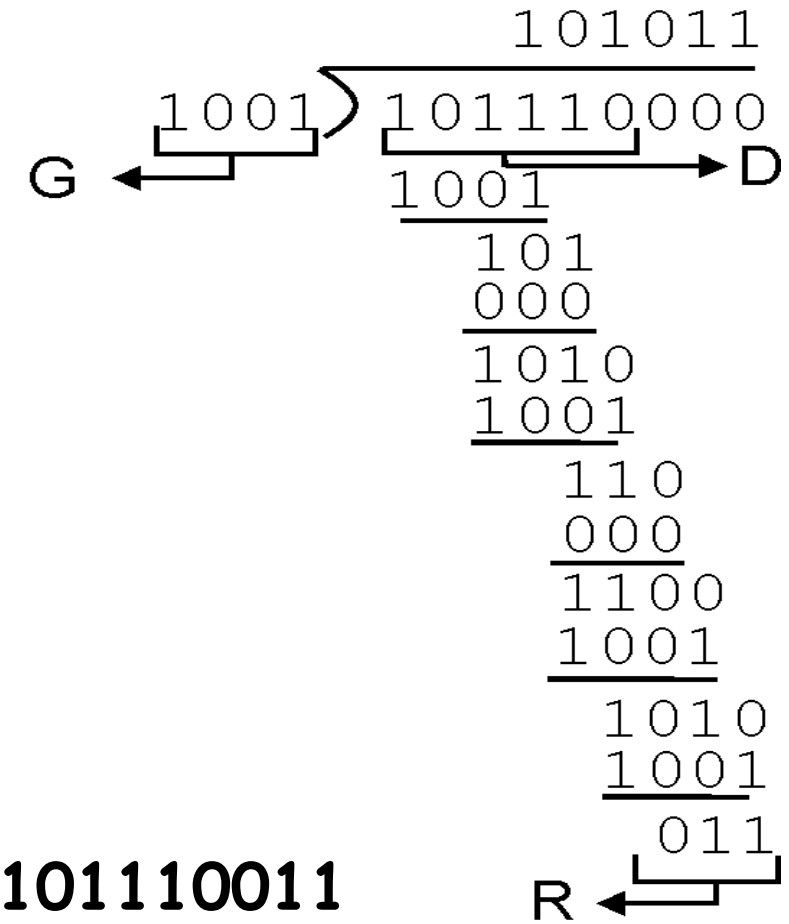
CRC (Cont.)



- What is the number of CRC (or FCS or check) bits if the generator is 65 bits? What is the efficiency for a frame of length 1500 bits? Compare it with single parity checking?
- There are 64 CRC bits and efficiency is 1500/1564. Efficiency is a lot higher than single parity bit checking

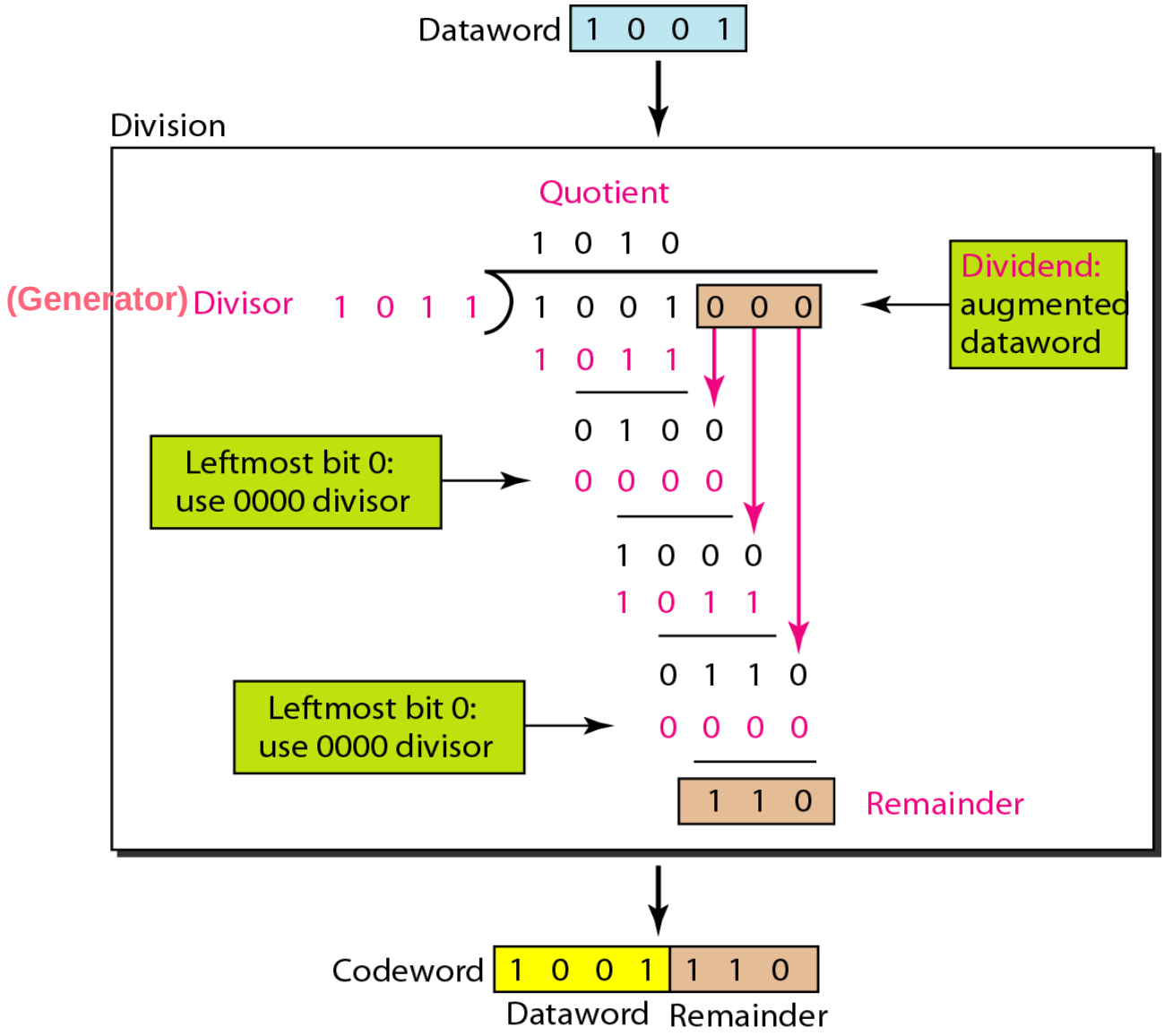
CRC Example 1

- Note that multiplication and division are the same as in base-2, except any required addition or subtraction is done without carries or borrows

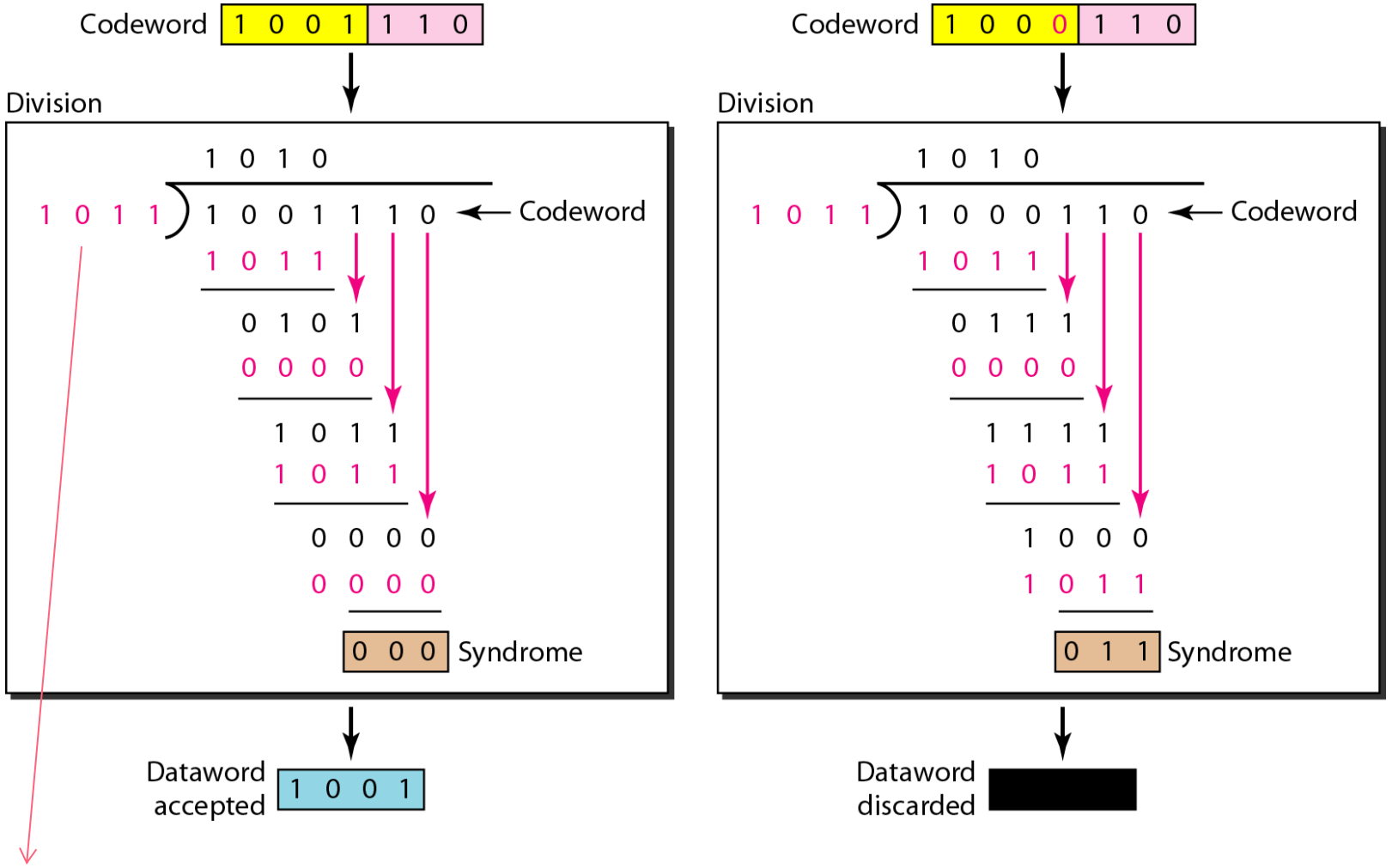


- Codeword: <D,R> = 101110011

CRC Example 2

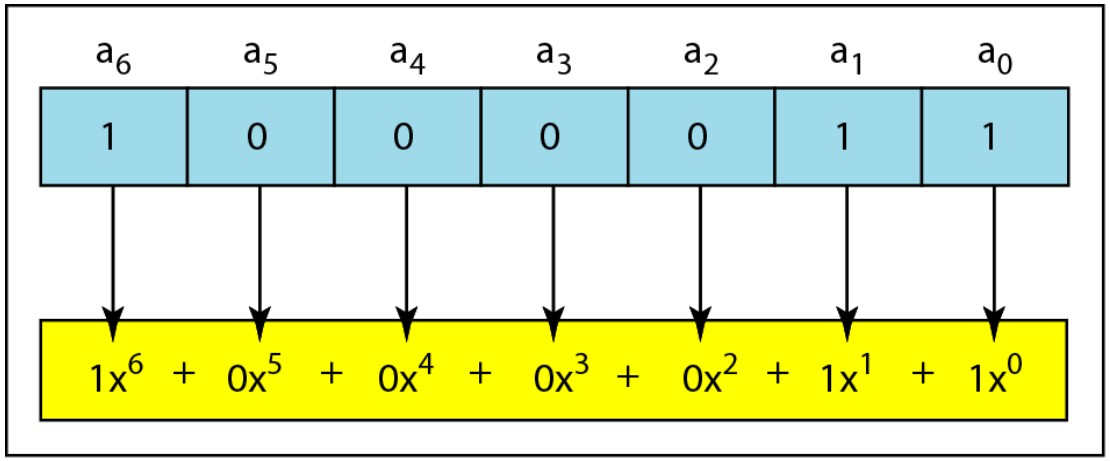


CRC Example 2 (Cont.)

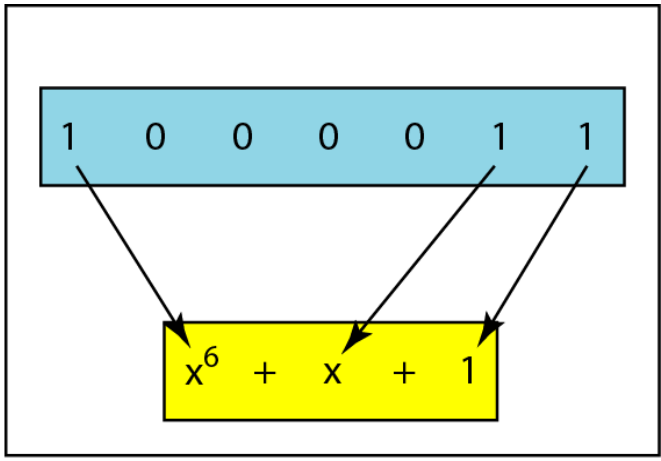


CRC Division Using Polynomials

- A polynomial representation can be alternatively used in CRC



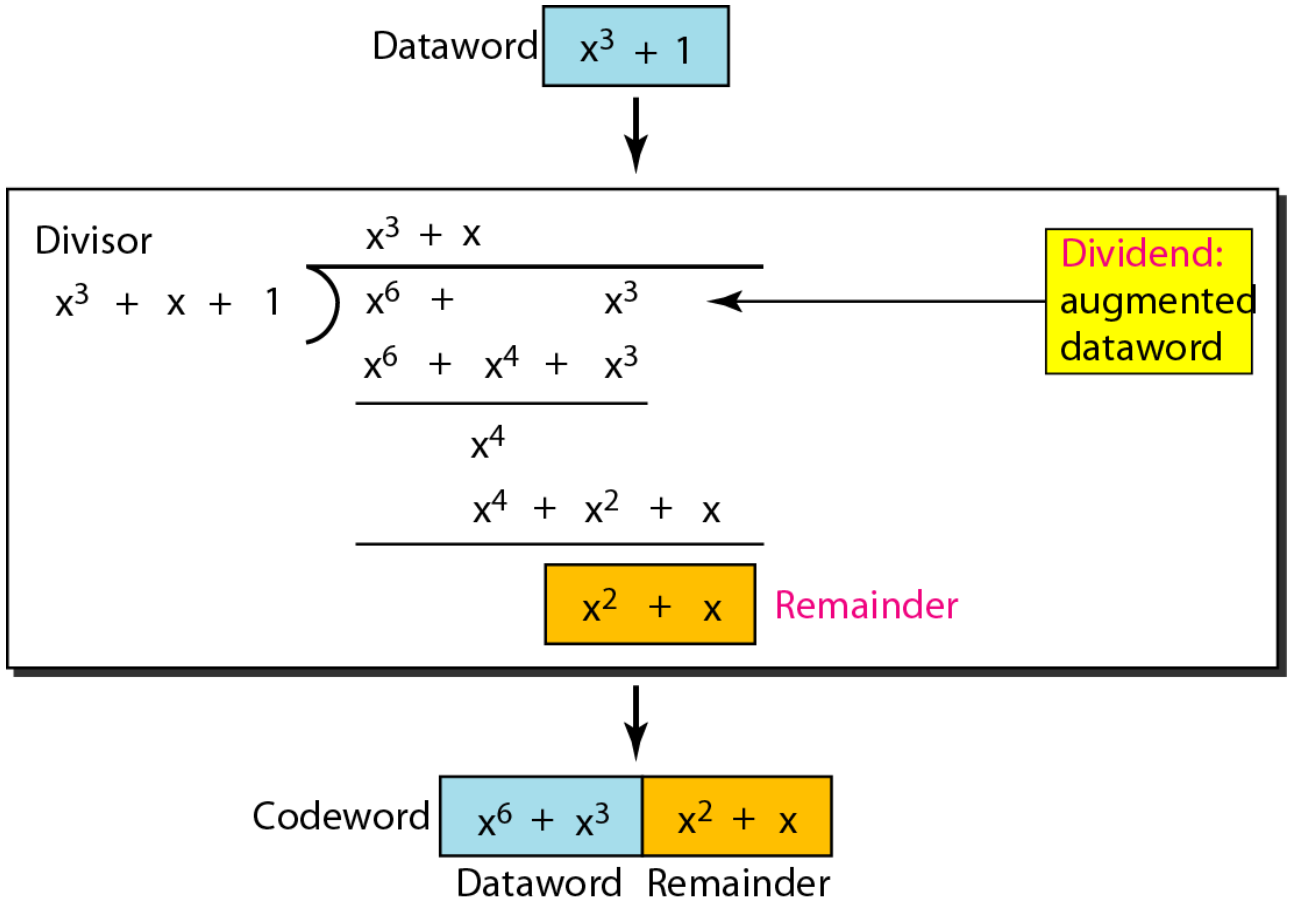
a. Binary pattern and polynomial



b. Short form

CRC Example 2 Using Polynomials

- **Dataword: 1001** **Generator: 1011**

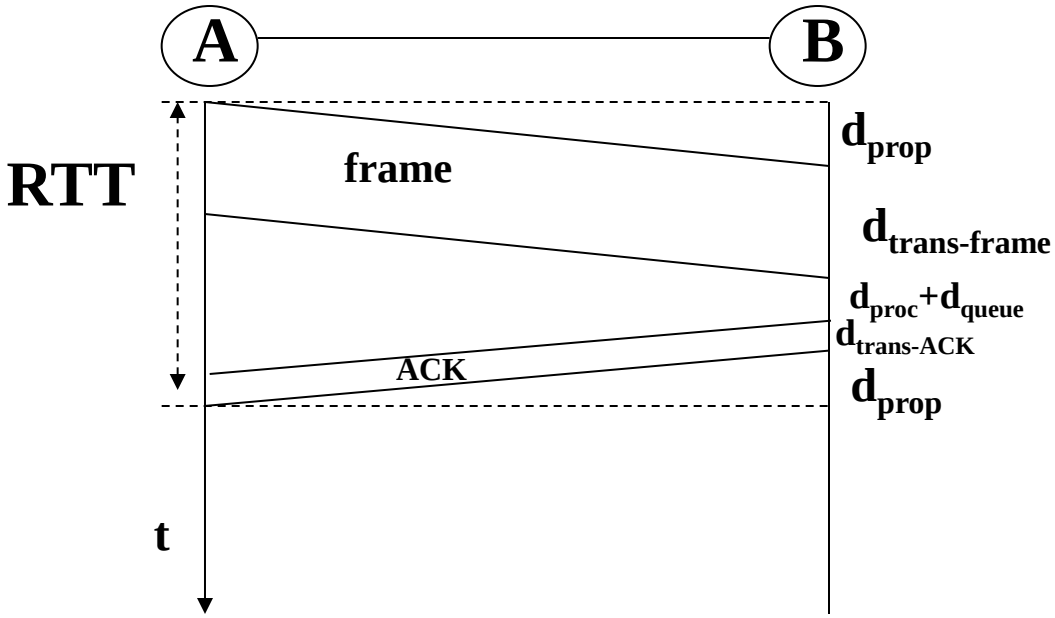


Latency, RTT, and Throughput

- B takes d_{proc-B} to do frame synchronization, frame error detection, etc:

$$RTT = d_{proc-A} + d_{queue-A} + d_{trans-frame} + d_{prop} + d_{proc-B} + d_{queue-B} + d_{trans-ACK} + d_{prop}$$

$$\cong d_{trans-frame} + 2d_{prop}$$



Throughput =

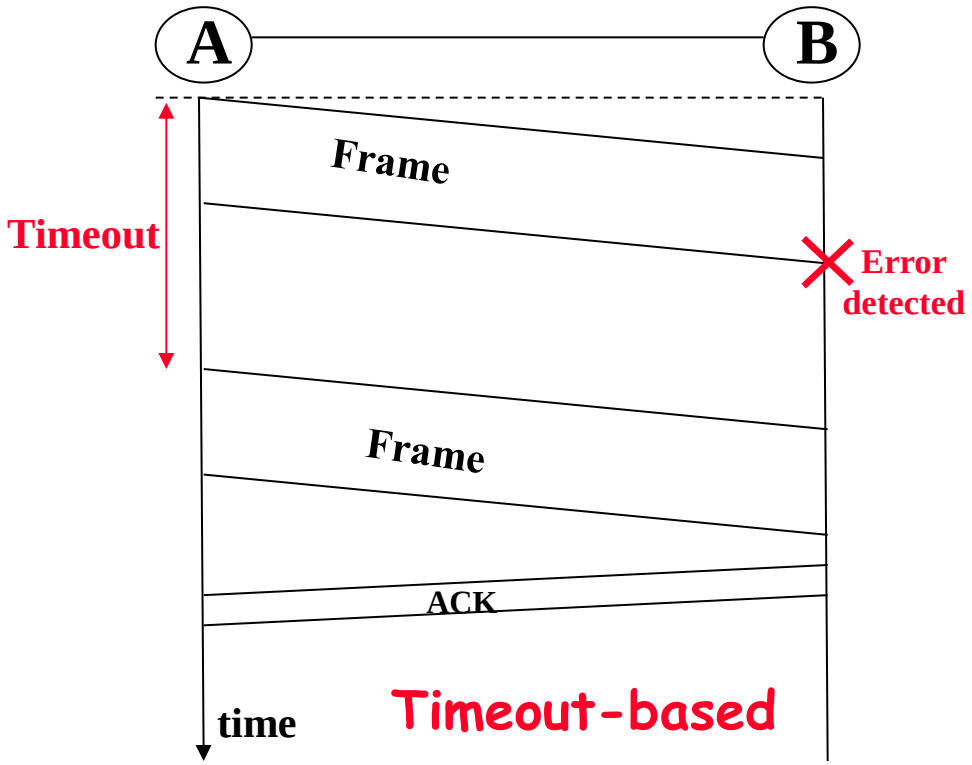
Frame Length/RTT

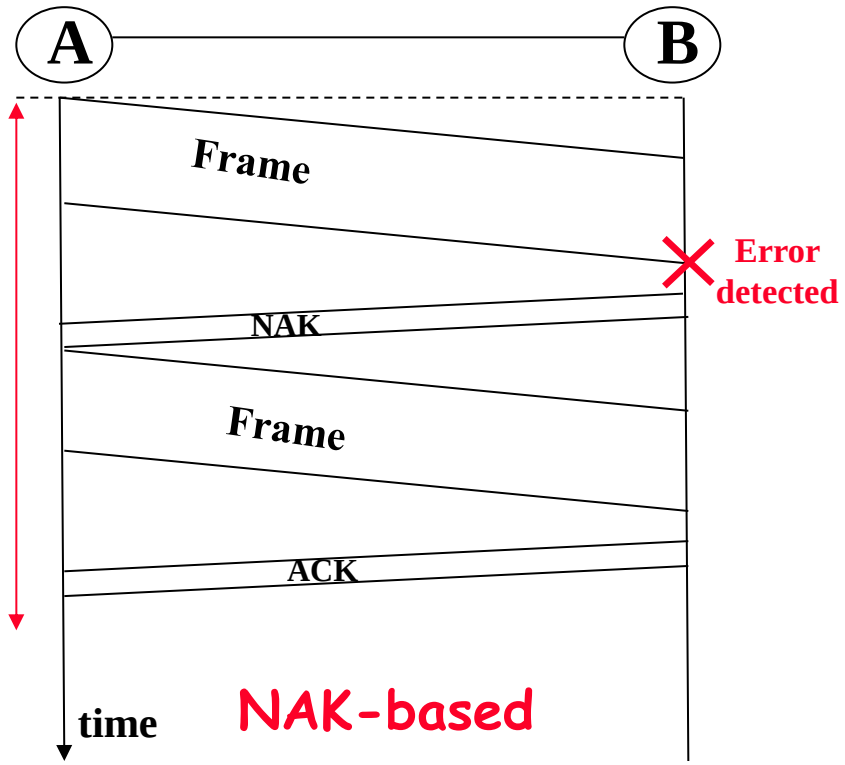
- Sender transmits the frame, but keeps the frame in its buffer and does not remove it until it receives an ACK from the receiver
- Due to frame error or ACK drop, RTT can be longer and throughput lower

Time Out vs Negative ACK (NAK)

- In a timeout-based-only protocol (i.e., no NAK) receiver does nothing after error detection, but sender starts a timer when transmitting the frame, and after a timeout and not hearing an ACK from the receiver, retransmits the frame, however if NAK is implemented in a protocol, the receiver will send a NAK after detecting the error
- Most DLL protocols work based on timeout rather than NAK
- In timeout-based, timer starts either at the beginning or end of transmission

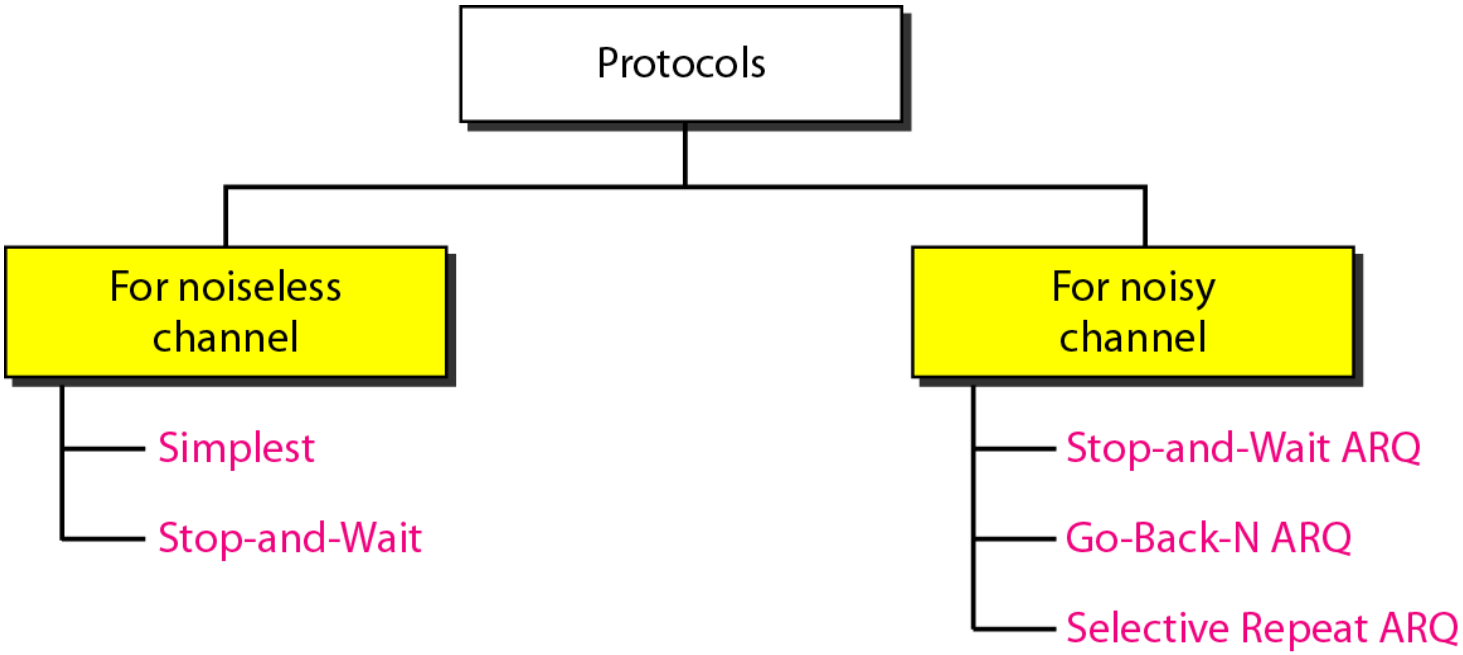
Time Out vs NAK (Cont.)





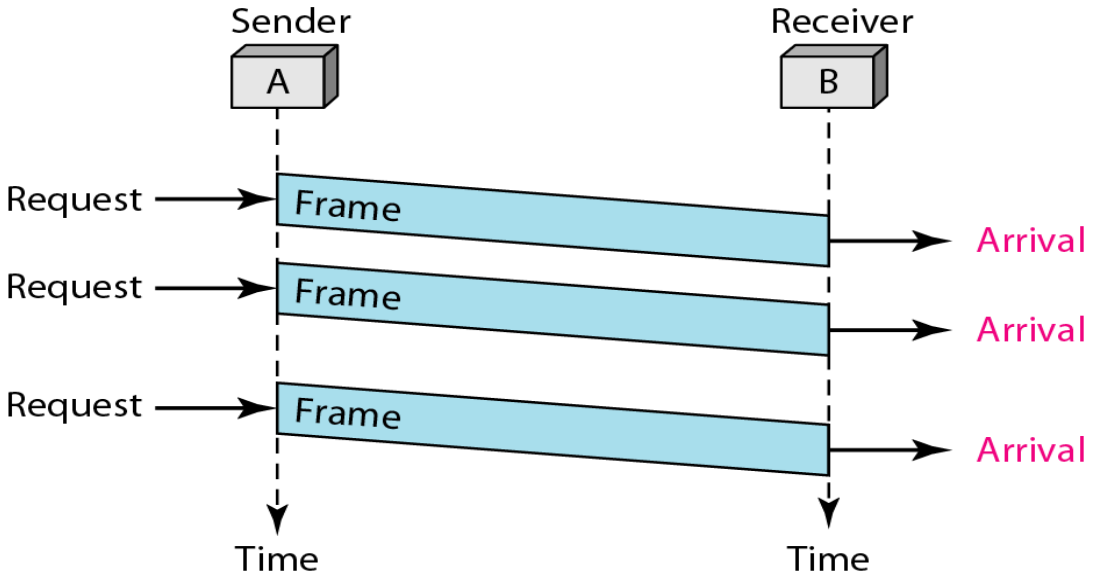
Framing, Error and Flow Control Protocols

- 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 languages



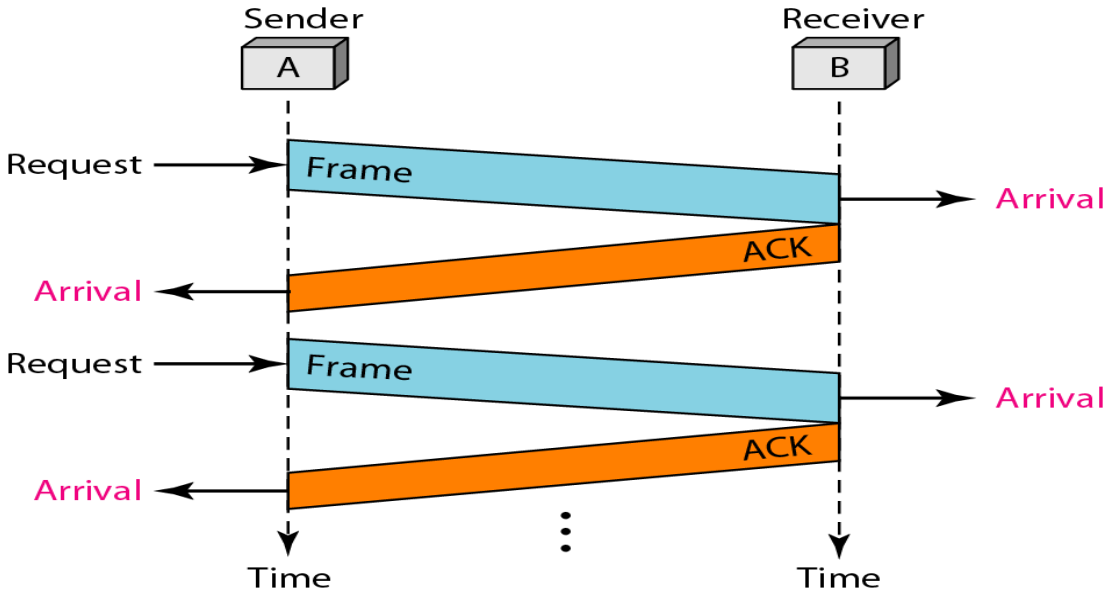
Noiseless Channels – Simplest Protocol

- A protocol that has no flow or error control
- Assumptions
  - No error exists
  - Receiver can immediately handle any frame it receives with negligible processing delay. DLL of the receiver removes the header from the frame and hands the data packet to its network layer which can also accept packet immediately. In other words the receiver can never be overwhelmed



Noiseless Channels – Stop & Wait

- If frames arrive at receiver faster than they can be processed (e.g., when data is transmitted from many senders,) they must be stored, but a receiver has typically not enough storage space
- Receiver needs to tell sender to slow down to avoid becoming overwhelmed
- Stop & Wait protocol is the Simplest protocol + flow control



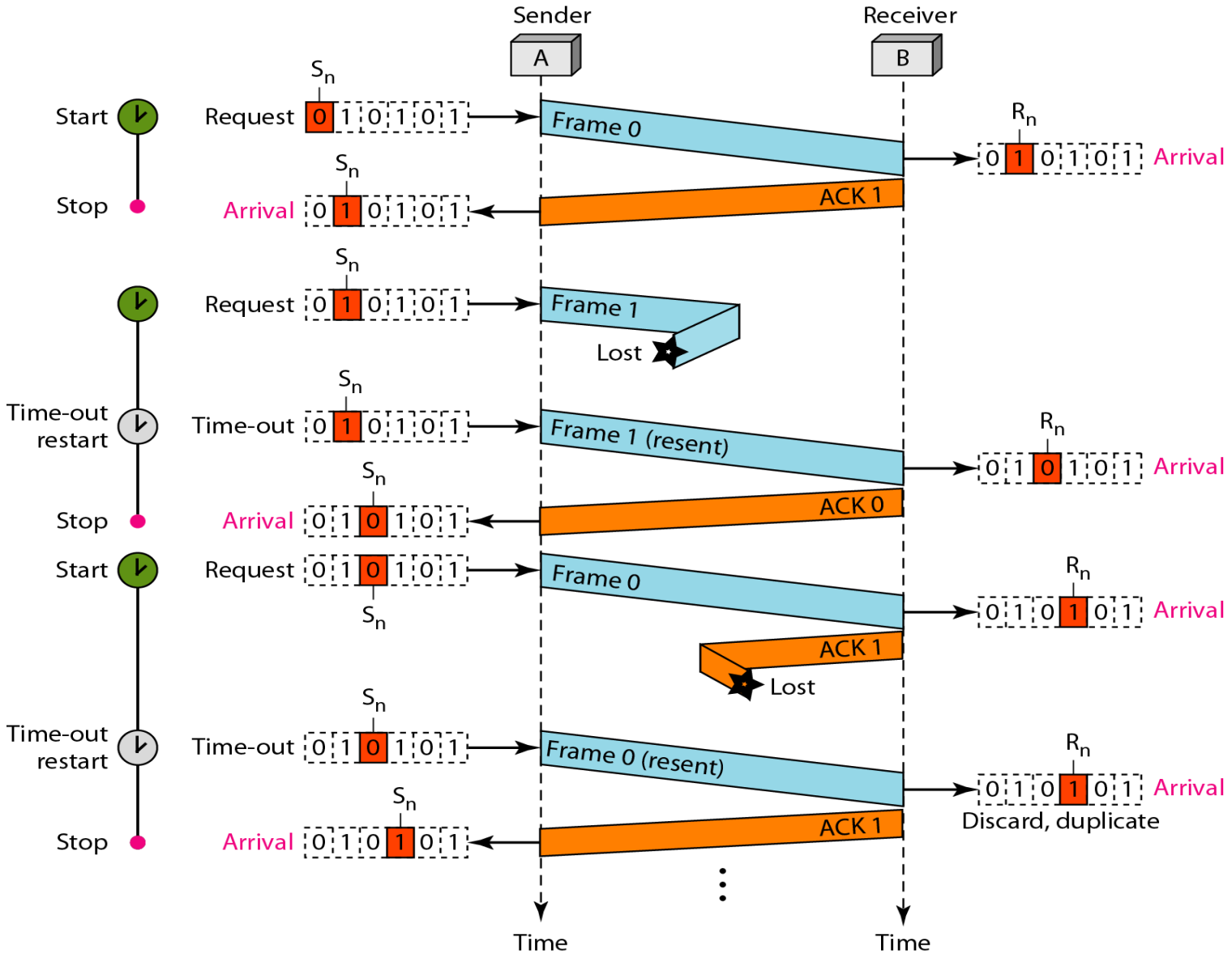
Noisy Channels – Stop & Wait ARQ

- Stop & Wait gave us an idea of how to add flow control, however noiseless channels are unreal, so instead of ignoring the error, we should add error control to our protocols
- In Stop & Wait ARQ (Automatic Repeat reQuest,) error control is done by keeping a copy of the sent frame and retransmitting the frame when the timer expires or when the acknowledgement gets damaged
- Similarly to Stop & Wait, the sender transmits only one frame at a time
- In Stop & Wait ARQ the send and sliding windows can move only one step at a time

Stop & Wait ARQ – Example

- Next page shows an example where Frame 0 is sent and acknowledged
- Frame 1 is lost and resent after the time-out
- The resent frame 1 is acknowledged and the timer stops
- Frame 0 is sent and acknowledged, but the acknowledgment is lost
- The sender has no idea if the frame or the acknowledgment is lost, so after the time-out, it resends frame 0, which is acknowledged

Stop & Wait ARQ – Example (Cont.)



Stop & Wait ARQ – Utilization Percentage

- Assume that, in a Stop & Wait ARQ system, the bandwidth of the line is 1 Mbps, and RTT=20ms
- Questions: What is the bandwidth-delay product? If the system data frames are 1000 bits in length, what is the utilization percentage of the link?
  - Bandwidth-delay product is $1 \times 10^6 \times 20 \times 10^{-3} =$ 20000 bits or 20 frames
  - The system can send 20,000 bits during RTT
  - However, in Stop & Wait ARQ, sender sends one frame i.e., 1000 bits

- Stop & Wait ARQ is very inefficient if the channel is thick and long

Stop & Wait ARQ – Utilization (Cont.)

- Question: In the previous example what is the utilization percentage of the link if we use a protocol that can send up to 15 frames before stopping and waiting for the ACK?
  - Bandwidth-delay product is still 20000 bits or 20 frames
  - The system can send up to 15 frames or 15000 bits during a round trip (RTT)
  - Of course if there are damaged frames, the utilization percentage is much less because frames have to be resent

Pipelining in Networking

- Stop & Wait ARQ protocol is not efficient (i.e., low resource utilization) because only one frame can be sent and then the sender needs to wait for an ACK, so there is a lot of idle time
- In networking and in other areas, a task is often begun before the previous task has ended. This is known as **pipelining**. There is no pipelining in Stop & Wait ARQ because the sender needs to wait for a frame to reach the destination and be acknowledged before it can transmit the next frame
- However pipelining does apply to our next two protocols (Go-Back-N and Selective Repeat) because several frames can be transmitted before sender receives news about previous frames
- Pipelining improves the efficiency of the transmission if the number of bits in transition is large enough compared with the bandwidth-delay product

Go-Back-N ARQ

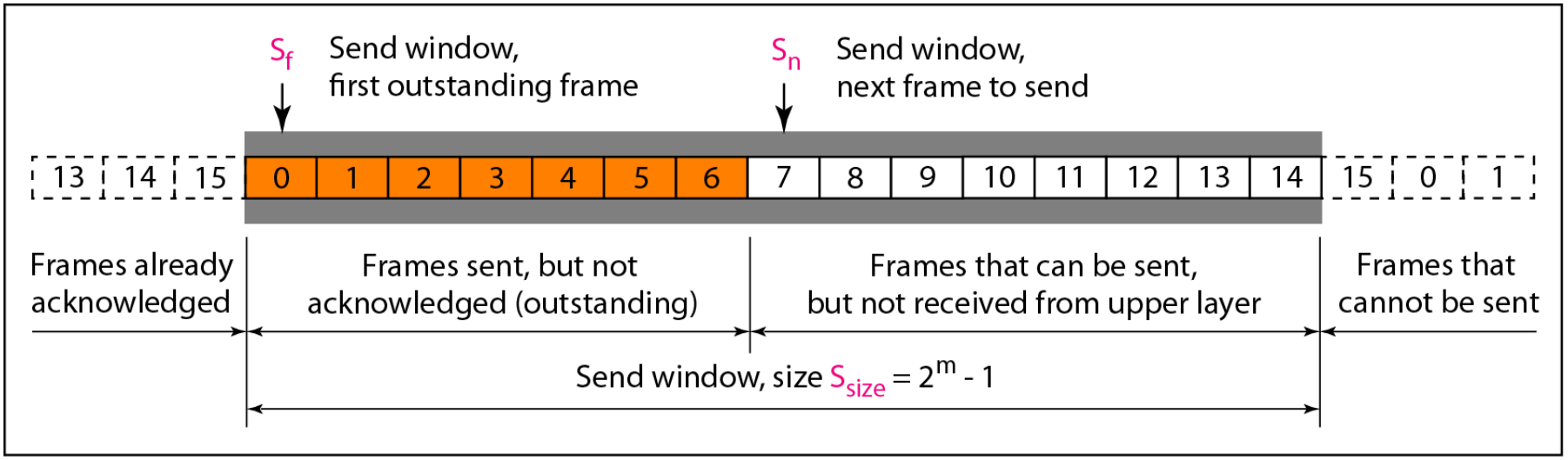
- A solution to Stop & Wait ARQ: to be able to let more than one **outstanding frame** (i.e., one that was sent but not acknowledged) to keep the channel busy while the sender is waiting for the acknowledgement
- **Go-Bank-N protocol** uses the **Sliding Window** protocol
- Sender can send more than one frame at a time, i.e., between receiving ACKs
- **The sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits**
- The send window is an abstract concept defining an abstraction, an imaginary box of **max size $2^m - 1$** with 3 variables: S_f (1st outstanding frame), S_n (next frame to send) and S_{size} (fixed to $2^m - 1$ in this protocol)

Go-Back-N ARQ (Cont.)

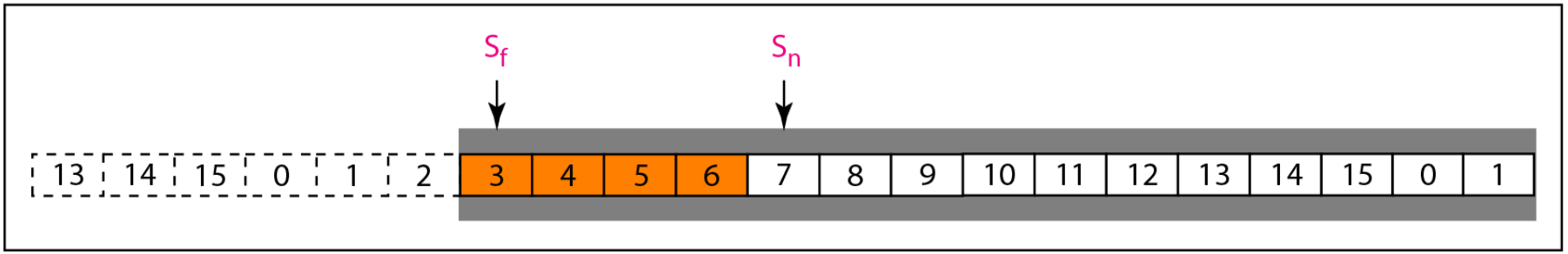
- The receive window is an abstract concept defining an imaginary box of size one with one single variable R_n. The window slides when a correct frame has arrived; sliding occurs one slot at a time
- The receiver sends an ACK if a frame has arrived with no errors detected. If a frame is damaged or received out of order, the receiver is silent and will discard all subsequent frames until it receives the one it is expecting
- The silence of the receiver causes the timer of the unacknowledged frame at the sender to expire. This in turn causes the sender to go back and resend all frames beginning with the one with the expired time
- The receiver does not have to acknowledge each frame received. Stop & Wait ARQ is a special case of Go-Back-N ARQ in which the size of the send window is 1

Go-Back-N ARQ – Send Window

- S_n holds the sequence number that will be assigned to the next frame to be sent
- Every time sender receives a valid ACK, send window will slide (S_f) by the number of frames that were acknowledged



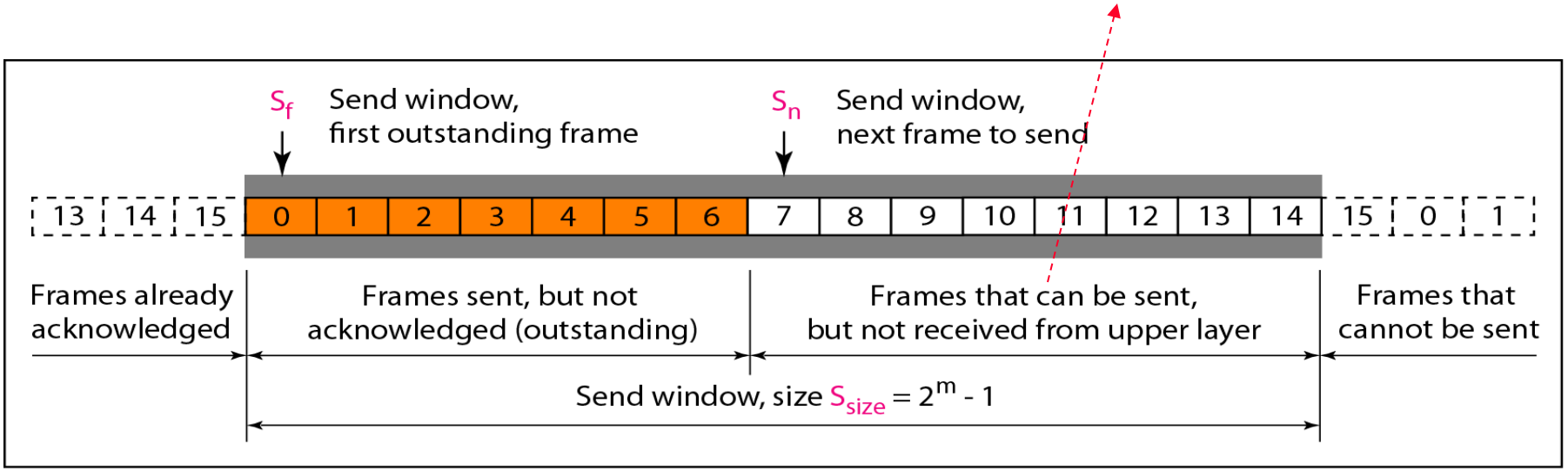
a. Send window before sliding



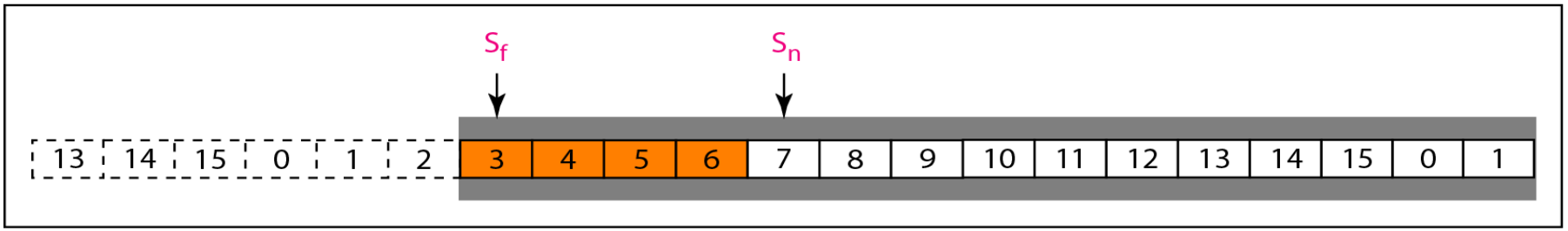
b. Send window after sliding

Go-Back-N ARQ – Send Window (Cont.)

- In this example, frames 0, 1, and 2 are acknowledged, so the window has slid 3 slots to the right. Now S_f is 3 because frame 3 is the first outstanding frame
- The sender gets the frames from the network layer



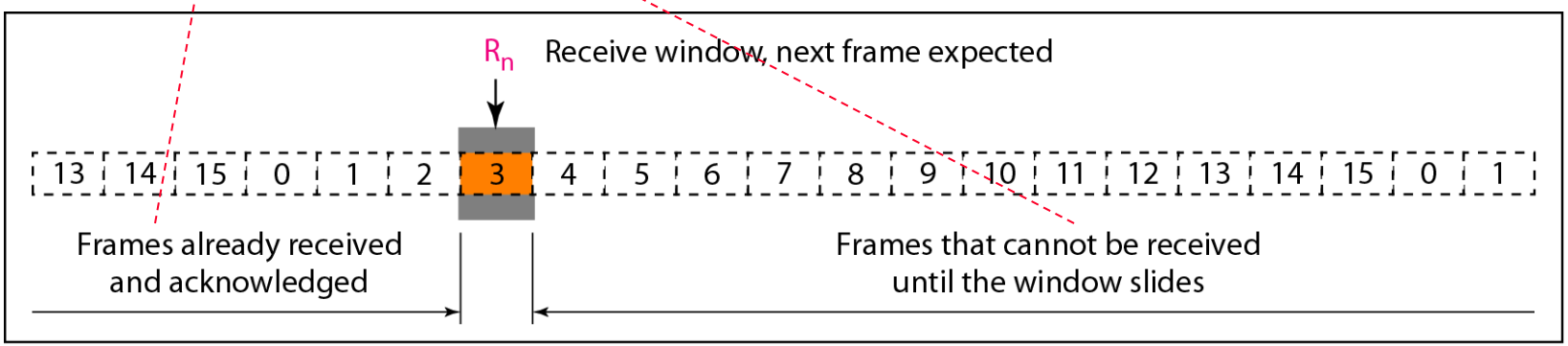
a. Send window before sliding



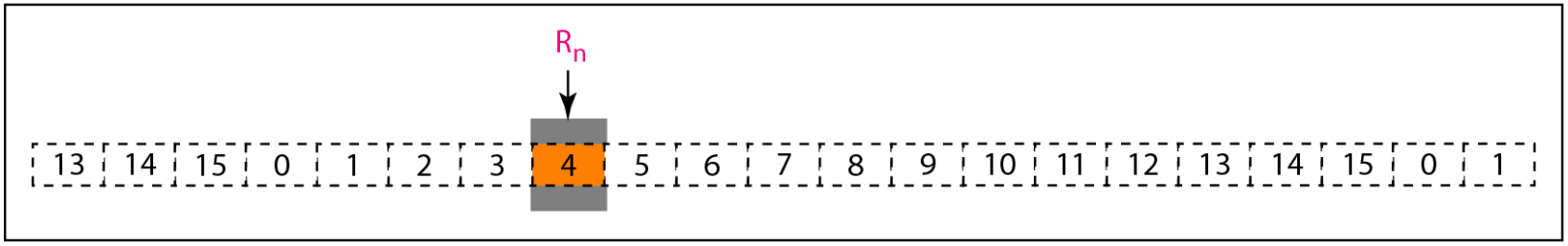
b. Send window after sliding

Go-Back-N ARQ – Receive Window

- Only one variable, R_n, is needed to define the receive window
- Any received frame with a sequence number in these two regions is discarded
- Only a frame with a sequence number matching the value of R_n is accepted and acknowledged



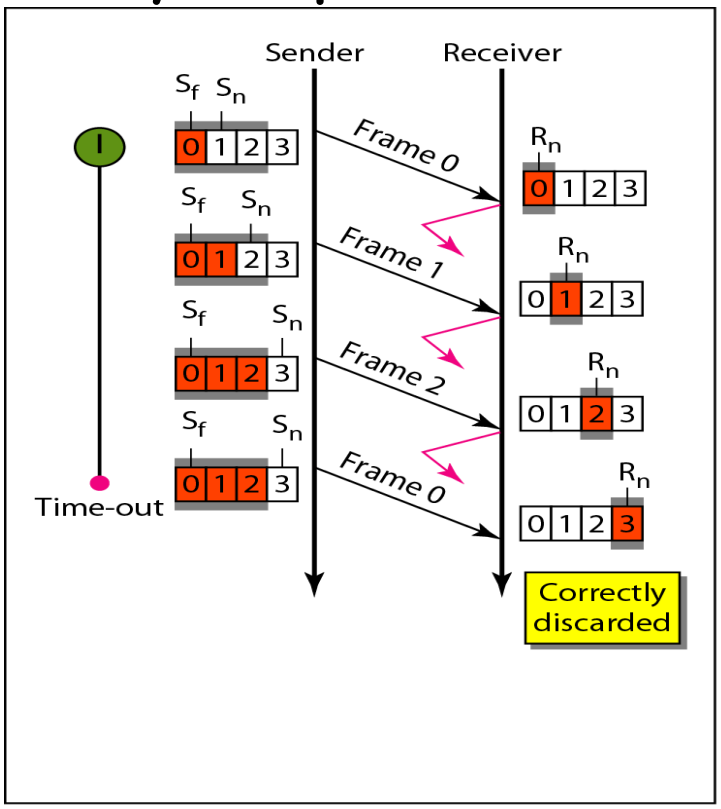
a. Receive window



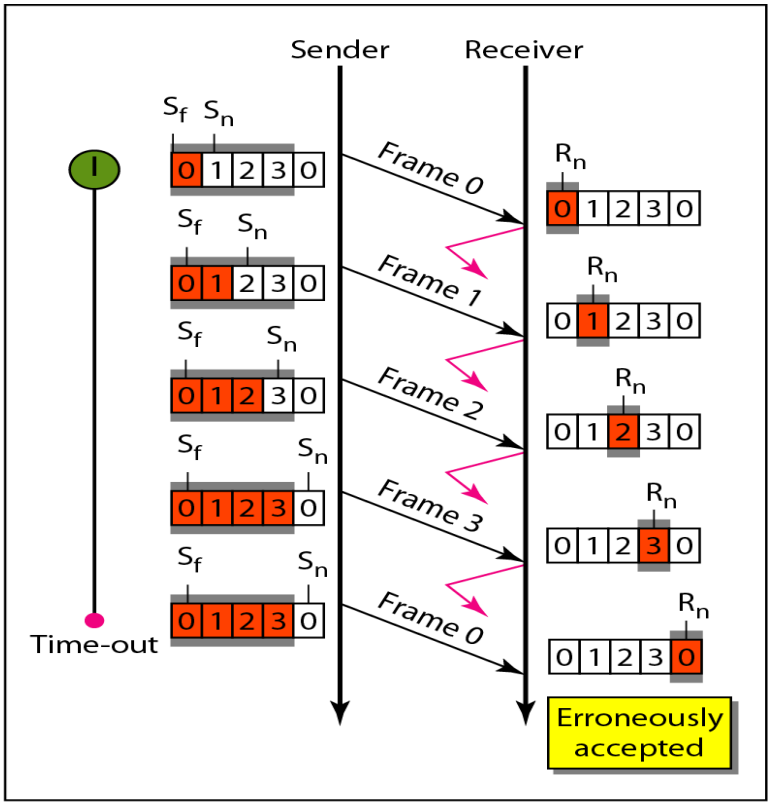
b. Window after sliding

Go-Back-N ARQ – Window Size

- Here m=2 and max send window size is supposed to be $2^2-1= 3$
- In this example the receiver is acknowledging after each frame however cumulative ACK is also possible
- With a wrong size of 4 (which is higher than S_{size}) receiver erroneously accepts the old Frame0 as the new Frame0



a. Window size < 2^m



b. Window size = 2^m

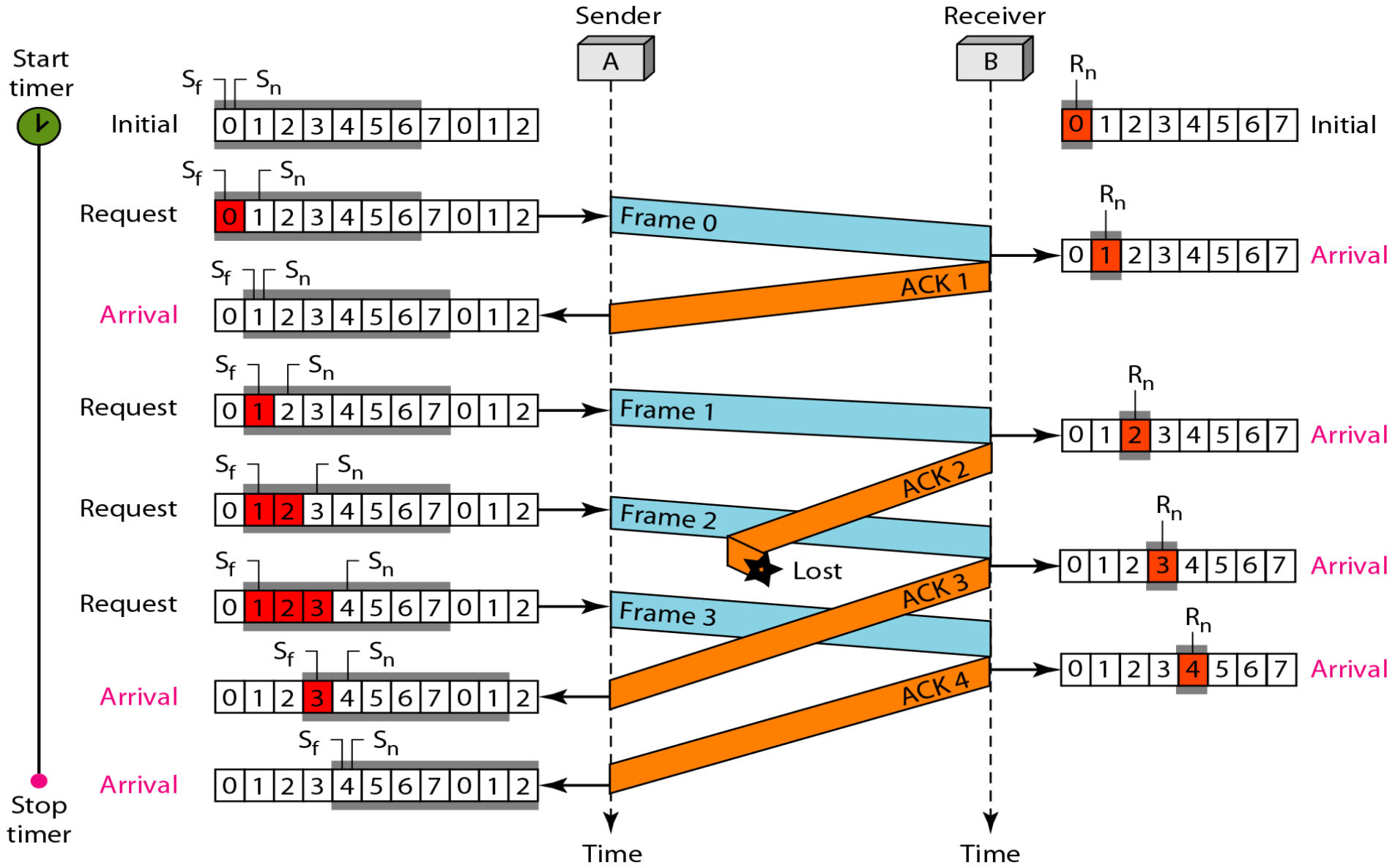
Go-Back-N ARQ – Timer

- Timer design
  - Although there can be a timer for each frame that is sent, we can use only one timer, because the timer for the 1st outstanding frame always expires first and we send all outstanding frames when that timer expires

Go-Back-N ARQ – Example 1

- Next page also shows an example of a case where the forward channel is reliable, but the reverse is not
- No data frames are lost, but some ACKs are delayed and one is lost
- The example also shows how cumulative acknowledgments can help if acknowledgments are delayed or lost
- After initialization, there are seven sender events
- Request events are triggered by data from the network layer; arrival events are triggered by acknowledgments from the physical layer
- There is no time-out event here because all outstanding frames are acknowledged before the timer expires
- Note that although ACK 2 is lost, ACK 3 serves as both ACK 2 and ACK 3

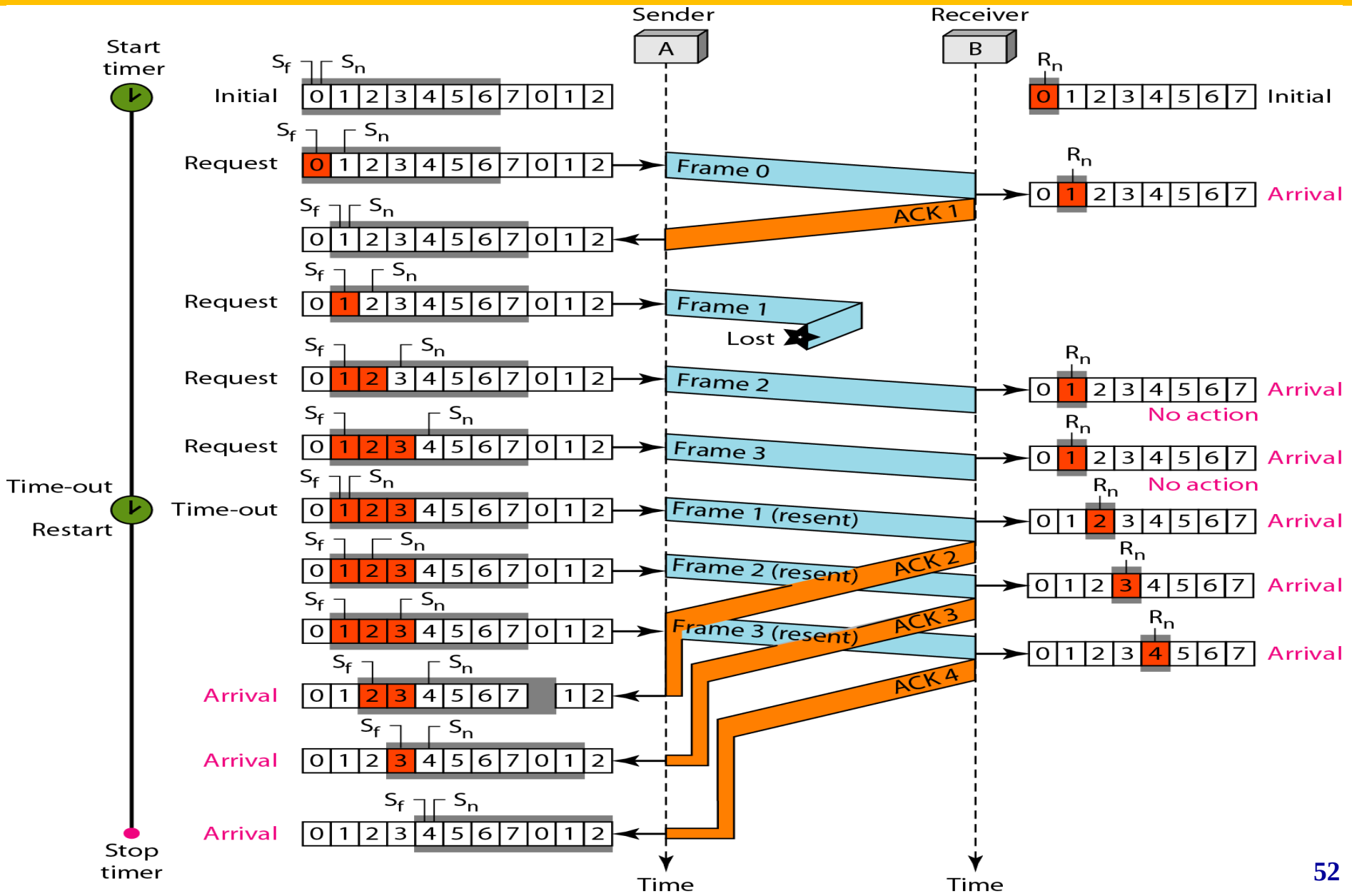
Go-Back-N ARQ – Example 1 (Cont.)



Go-Back-N ARQ – Example 2

- Next page shows what happens when a frame is lost. Frames 0, 1, 2, and 3 are sent. However, frame 1 is lost. The receiver receives frames 2 and 3, but they are discarded because they are received out of order. The sender receives no acknowledgment about frames 1, 2, or 3. Its timer finally expires. The sender sends all outstanding frames (1, 2, and 3) because it does not know what is wrong. Note that the resending of frames 1, 2, and 3 is the response to one single event. When the sender is responding to this event, it cannot accept the triggering of other events. This means that when ACK 2 arrives, the sender is still busy with sending frame 3
- The physical layer must wait until this event is completed and the data link layer goes back to its sleeping state. We have shown a vertical line to indicate the delay. It is the same story with ACK 3; but when ACK 3 arrives, the sender is busy responding to ACK 2. It happens again when ACK 4 arrives. Note that before the second timer expires, all outstanding frames have been sent and the timer is stopped

Go-Back-N ARQ – Example 2 (Cont.)

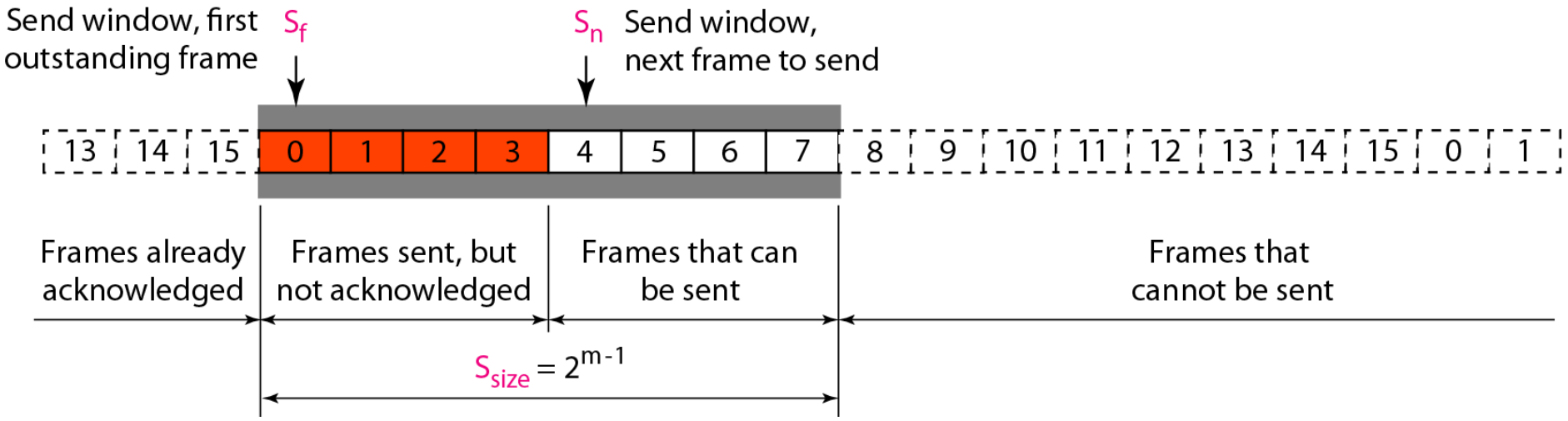


Selective Repeat ARQ

- Go-Bank-N ARQ simplifies the process at the receiver. The receiver keeps track of only one variable and there is no need to buffer out-of-order frames; they are simply discarded. This makes Go-Back-N very inefficient for a noisy link. In a noisy link a frame has a higher probability of damage, which means the need of resending of multiple frames; this resending uses up the bandwidth and slows down transmission. For noisy links, a more efficient mechanism is Selective Repeat ARQ which does not resend N frames when just one frame is damaged; it only sends the damaged frame
- Out-of-order frames are buffered by the receiver. Receiver only acknowledges the frames that are in order
- The process at the receiver is therefore more complex
- The size of the send window is much smaller than that in Go-Bank-N and is at most one-half of 2^m (i.e., 2^{m-1})

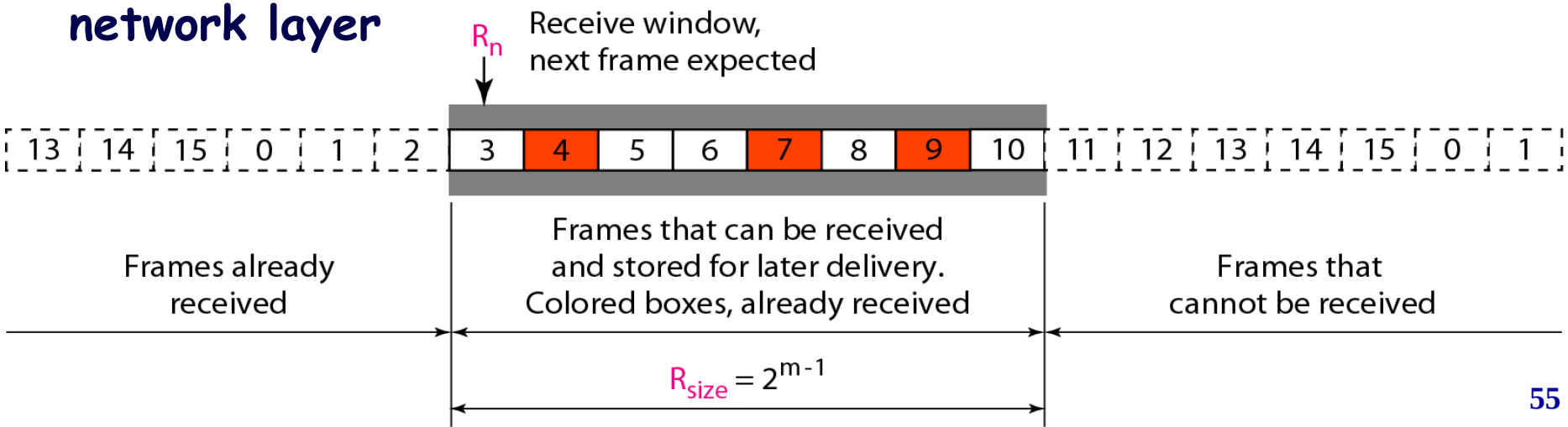
Selective Repeat ARQ – Send Window

- m=4, so the sequence numbers go from 0 to 15 (window size is 8 which is fine)
- Smaller size send window (compared to Go-Bank-N) means less efficiency in filling up the pipe, but the fact that there are fewer duplicate frames, can compensate for this



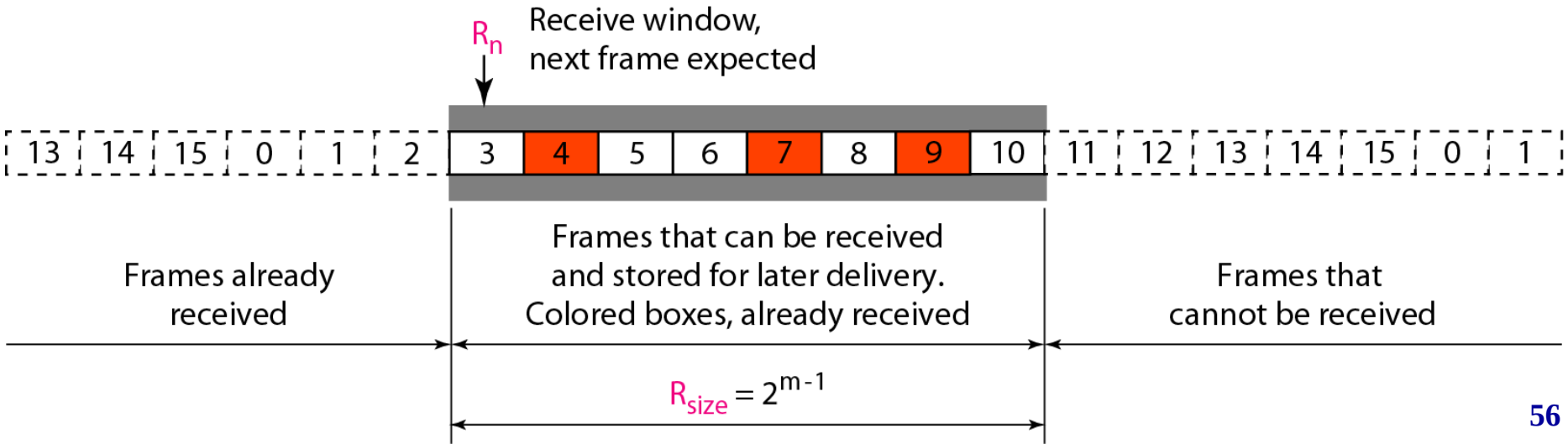
Selective Repeat ARQ – Receive Window

- The receive window is totally different from the one in Go-Back-N: main difference is that the receive window size is the same as that of the send window (i.e., $R_{size} = S_{size}$)
- This allows as many frames as the receive window size to arrive out of order and be kept until there is a set of in-order frames to be delivered to the network layer
- Because the size of the send and receive windows are the same, all the frames in the send window can arrive out of order and be stored until they can be delivered
- Note: receiver never delivers packets out of order to the network layer



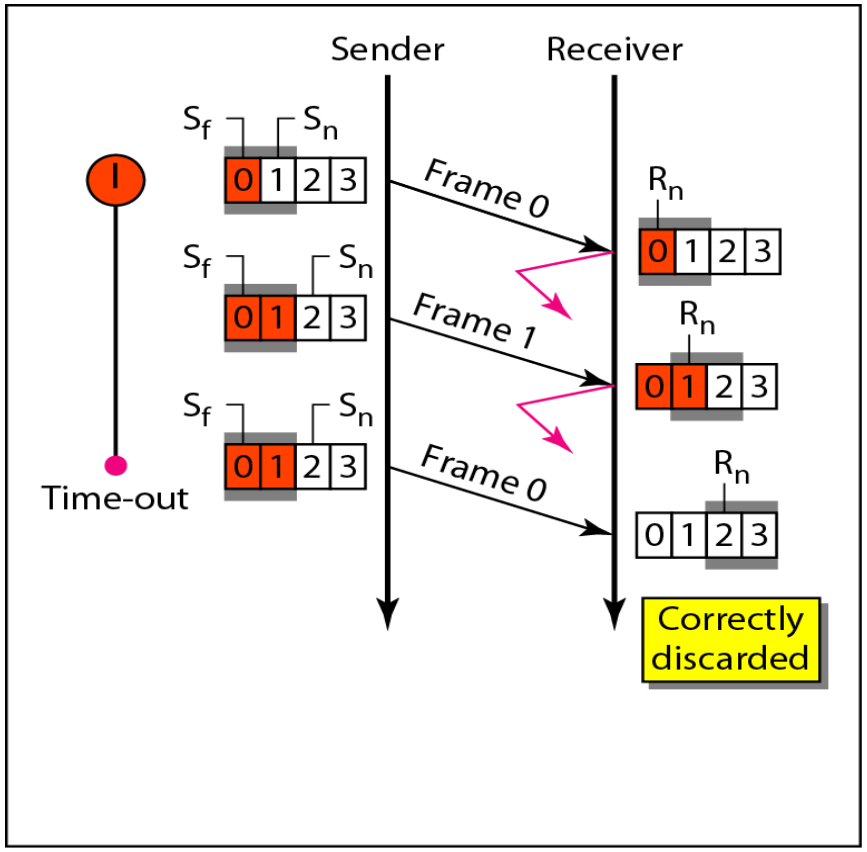
Selective Repeat ARQ – Receive Window (Cont.)

- If the receiver does not want the sender to overwhelm it with frames, the receiver delays the ACK, so that the send window does not open (Note: Once send window closes, sender has to stop and wait for acknowledgement from receiver)
- Sender does not know that the receiver is delaying ACK, so it will time out and resend those frames again, however receiver knows they are duplicate and discards them

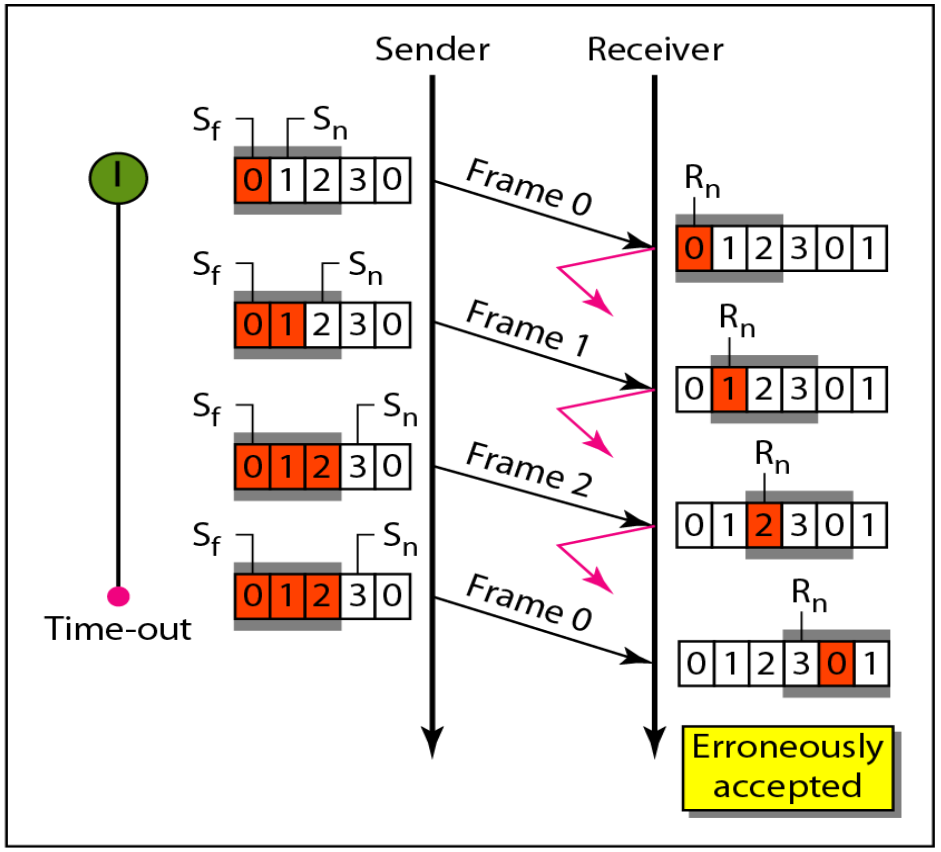


Selective Repeat ARQ – Window Size

- Here the receiver is acknowledging after each frame (although in general it can be cumulative
- With wrong window size of 3, receiver erroneously accepts the old Frame 0 as the new Frame0



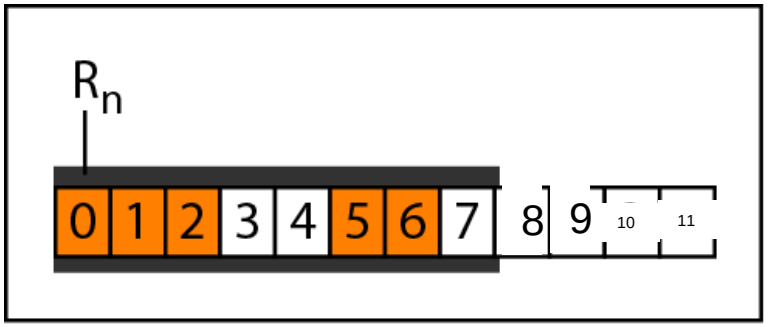
a. Window size = 2^{m-1}



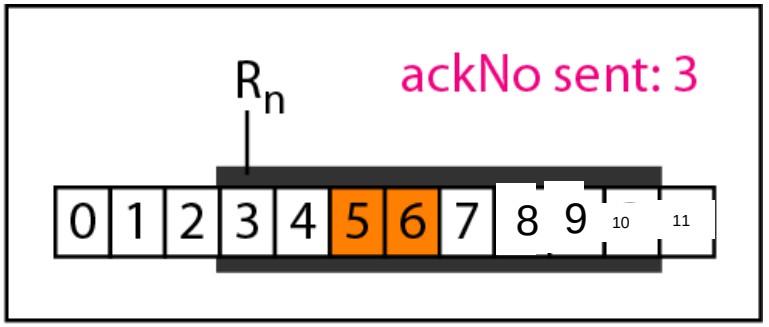
b. Window size > 2^{m-1}

Selective Repeat ARQ – Data Delivery

- If receiver receives a frame and no error is detected and the sequence number is in the window, the receiver stores the frame and marks the slot
- If contiguous frames, starting from R_n have been marked, their data is delivered to the network layer and the window is slid



a. Before delivery



b. After delivery

Selective Repeat ARQ –Example

- This example is similar to Example 2 of Go-Back-N in which frame 1 is lost. We show how Selective Repeat behaves in this case
- One main difference is the number of timers. Here, each frame sent or resent needs a timer, which means that the timers need to be numbered (0, 1, 2, and 3)
- The timer for frame 0 starts at the first request, but stops when the ACK for this frame arrives
- The timer for frame 1 starts at the second request, restarts when a NAK arrives, and finally stops when the last ACK arrives
- The other two timers start when the corresponding frames are sent and stop at the last arrival event
- At the receiver site we need to distinguish between the acceptance of a frame and its delivery to the network layer
  - At the second arrival, frame 2 arrives and is stored and marked, but it cannot be delivered because frame 1 is missing

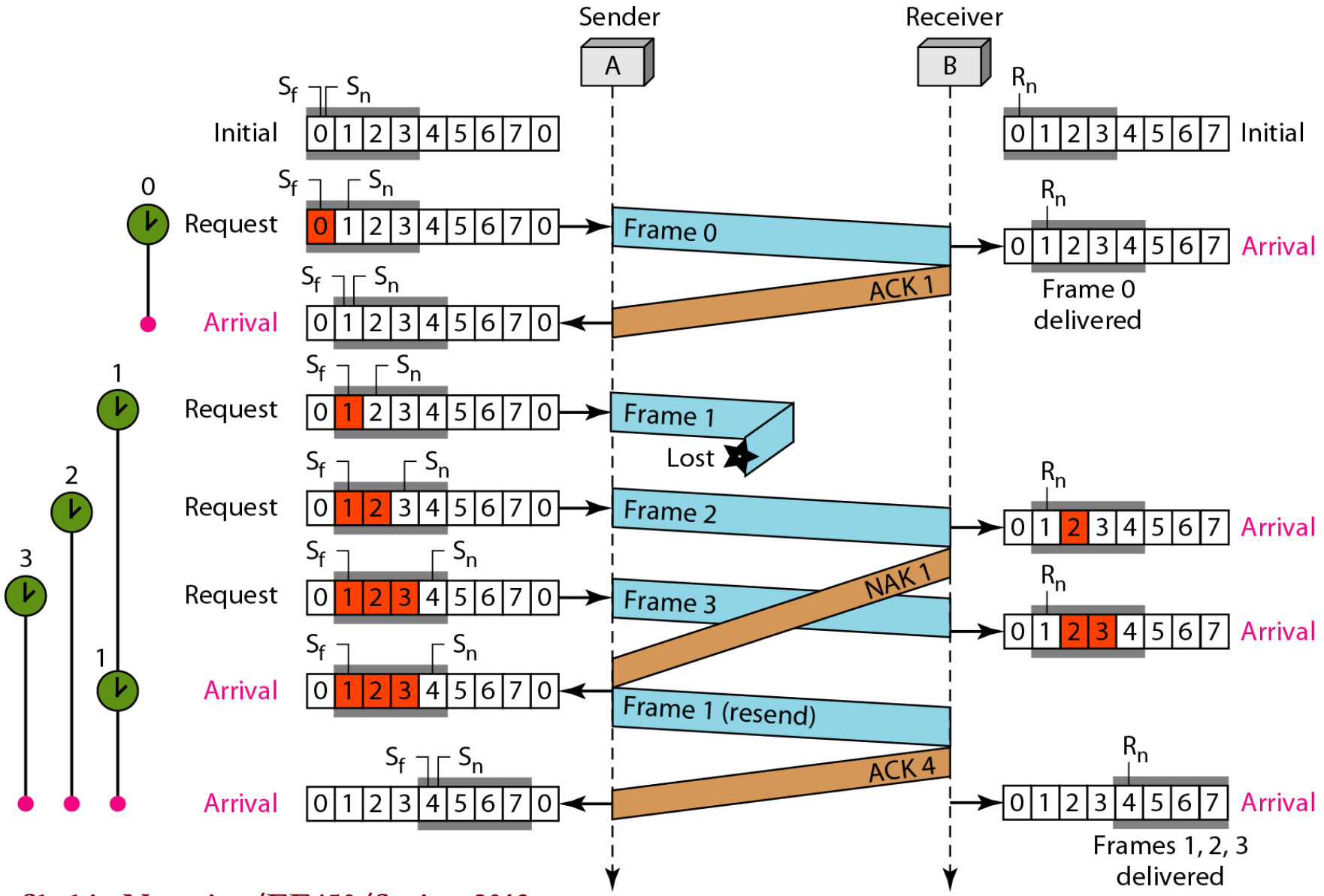
Selective Repeat ARQ –Example (Cont.)

- At the next arrival, frame 3 arrives and is marked and stored, but still none of the frames can be delivered
- Only at the last arrival, when finally a copy of frame 1 arrives, can frames 1, 2, and 3 be delivered to the Network layer
- There are two conditions for the delivery of frames to the network layer: First, a set of consecutive frames must have arrived. Second, the set starts from the beginning of the window
- The next point is about the ACKs. Notice that only two ACKs are sent here
- The first one acknowledges only the first frame; the second one acknowledges three frames
- In Selective Repeat, ACKs are sent when data are delivered to the network layer

Selective Repeat ARQ –Example (Cont.)

- If the data belonging to n frames are delivered in one shot, only one ACK is sent for all of them
- Another important point is that a NAK is sent after the second arrival, but not after the third, although both situations look the same
  - The reason is that the protocol does not want to crowd the network with unnecessary NAKs and unnecessary resent frames
- The second NAK would still be NAK1 to inform the sender to resend frame 1 again; this has already been done
- The first NAK sent is remembered and is not sent again until the frame slides
- A NAK is sent once for each window position and defines the first slot in the window

Selective Repeat ARQ –Example (Cont.)



Piggybacking

- The protocols we discussed so far are all unidirectional: The data frames flow in only one direction whereas the control ACK and NAK frames travel in the opposite direction

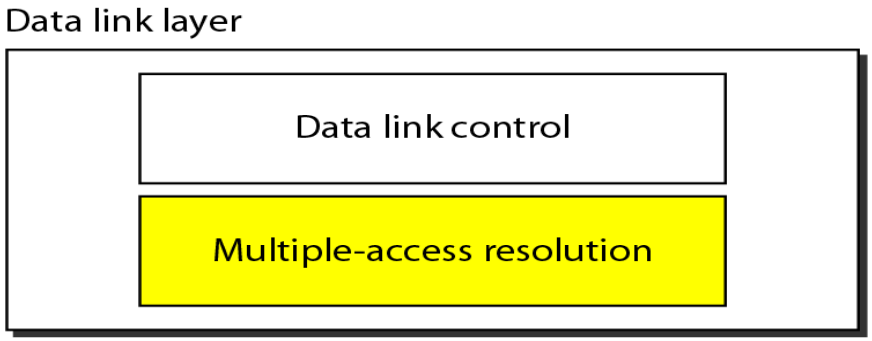
- In practice, data frames flow in both directions, and so do the control frames

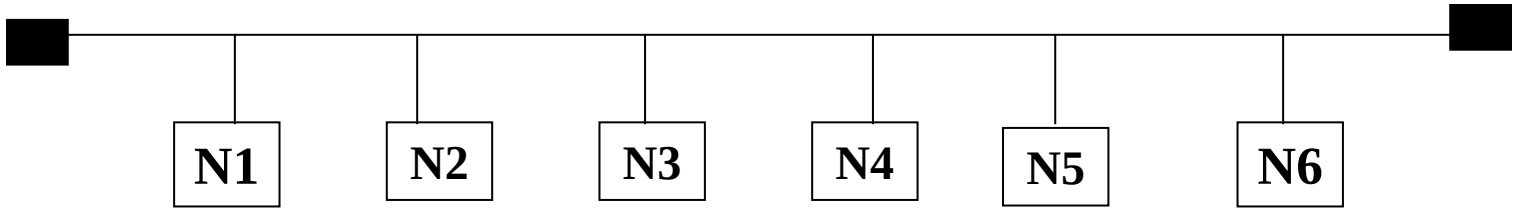
- **Piggybacking** is used to improve the efficiency of the bidirectional protocols

- In piggybacking it is important that both sides use the same algorithm

Medium Access Control (MAC) Protocols

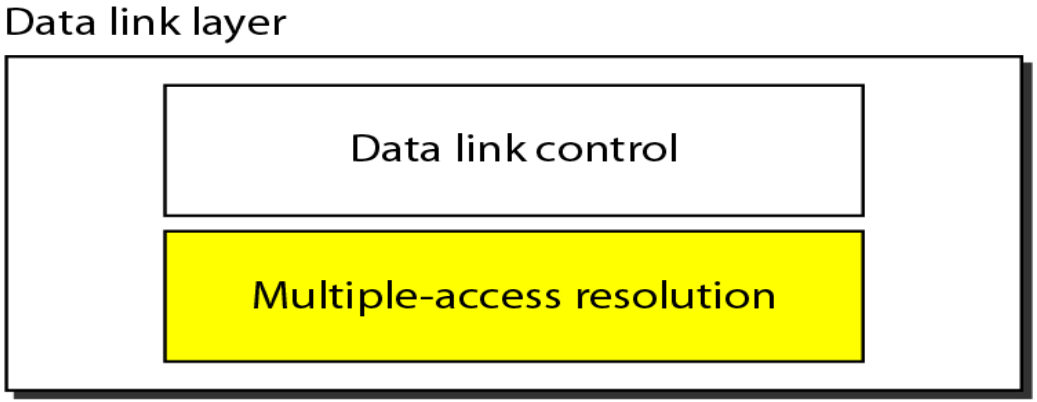
- So far our protocols assumed point-to-point connection (i.e., dedicated link or channel) btn the sender and the receiver
- DSL is an examples of point-to-point connection for which collision cannot happen
- To account for multi-layer connections, the DLL has two sublayers
- If the channel is dedicated we do not need the lower sublayer

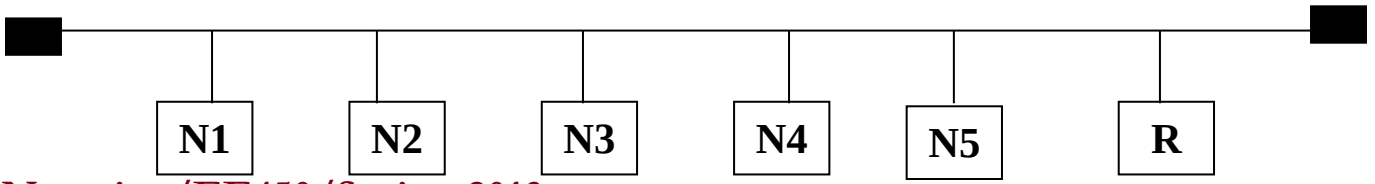




Medium Access Control (MAC)

- Examples of multi-point connection (shared access) are Ethernet (LAN) and cable access. If two or more nodes transmit simultaneously collision occurs
- We need a MAC procedure to avoid or detect collision
- Every node in multi-point connection (including the immediate router) needs to implement the MAC procedure



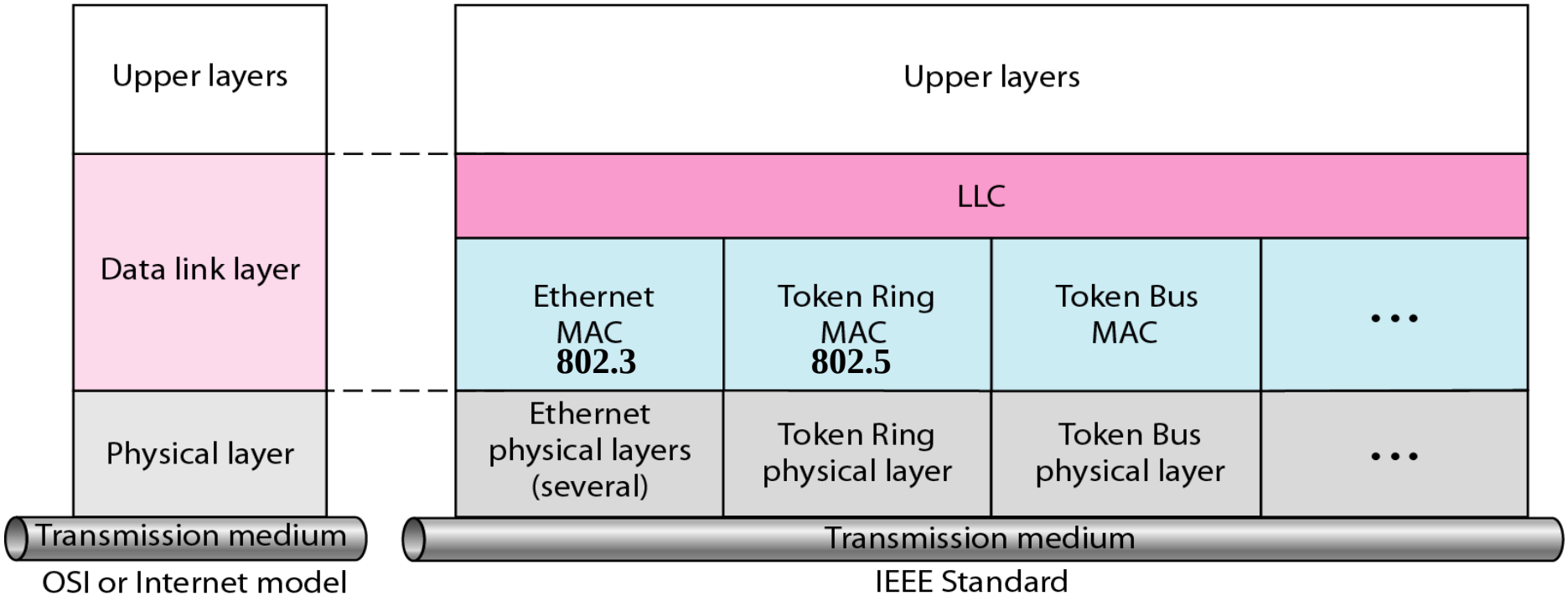


IEEE Standards for LANs (IEEE802)

- LAN is a computer network that is designed for a limited geographic area such as a building or a campus
- LAN became popular when the TCP protocols had already been written based on point-to-point connection, therefore TCP protocols only included the data link control tasks we looked (e.g., framing, error detection, error control and flow control)
- In 1985 IEEE started the 802 project to standardize Physical layer and Data link layer for LANs
- Project 802 does not seek to replace any part of OSI or the Internet model, instead it is a way of specifying functions of Physical layer and Data Link layer of major LAN protocols
- IEEE802 subdivides the Data Link layer into two sublayers, **Logical Link Control** (**LLC**) [as the data link control sublayer] and **Media Access Control** (**MAC**) [as Multiple access resolution sublayer] and also has several Physical layer standards for different LAN protocols

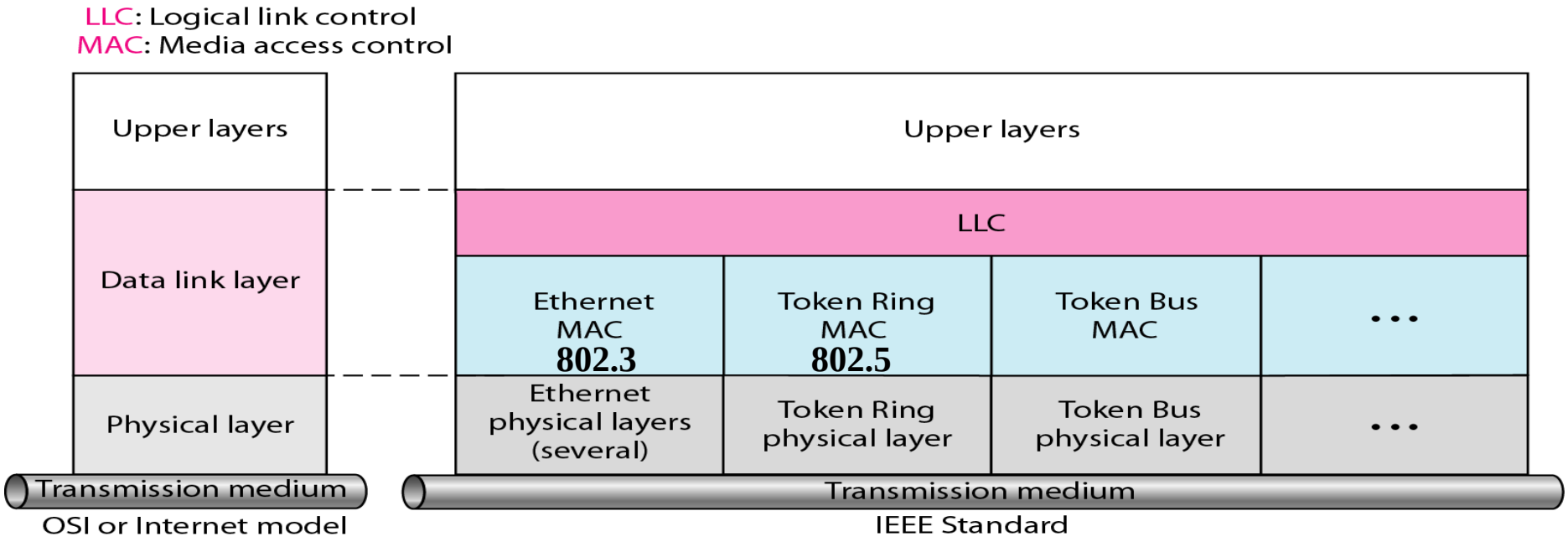
IEEE802 (Cont.)

LLC: Logical link control
MAC: Media access control



IEEE802 (Cont.)

- The former data link control tasks (framing, error detection and control and flow control) are collected into sublayer LLC
- LLC is like a common piece that provides one single data link control protocol for all IEEE LANs. Therefore LLC is different from MAC sublayer that provides different protocols for different LANs
- LLC hides the details of the MAC protocols from Network layer



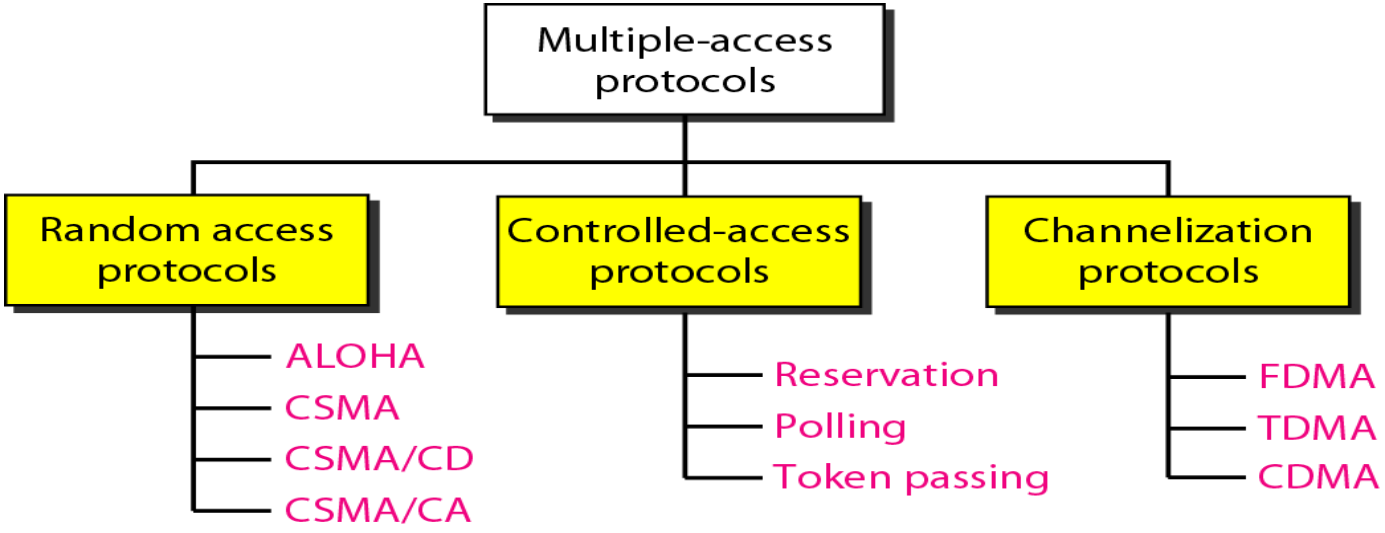
IEEE802 (Cont.)

- IEEE802 has created the MAC sublayer to define the specific access method for each LAN, e.g., it defines CDMA/CD as the MAC method for Ethernet LANs and token ring passing method for Token Ring LANs

- Physical layer is dependent on the implementation and type of physical media used. IEEE defines specifications for each LAN implementation, e.g., although there is only one MAC sublayer for Standard Ethernet, there are different physical layer specifications for each Ethernet implementations

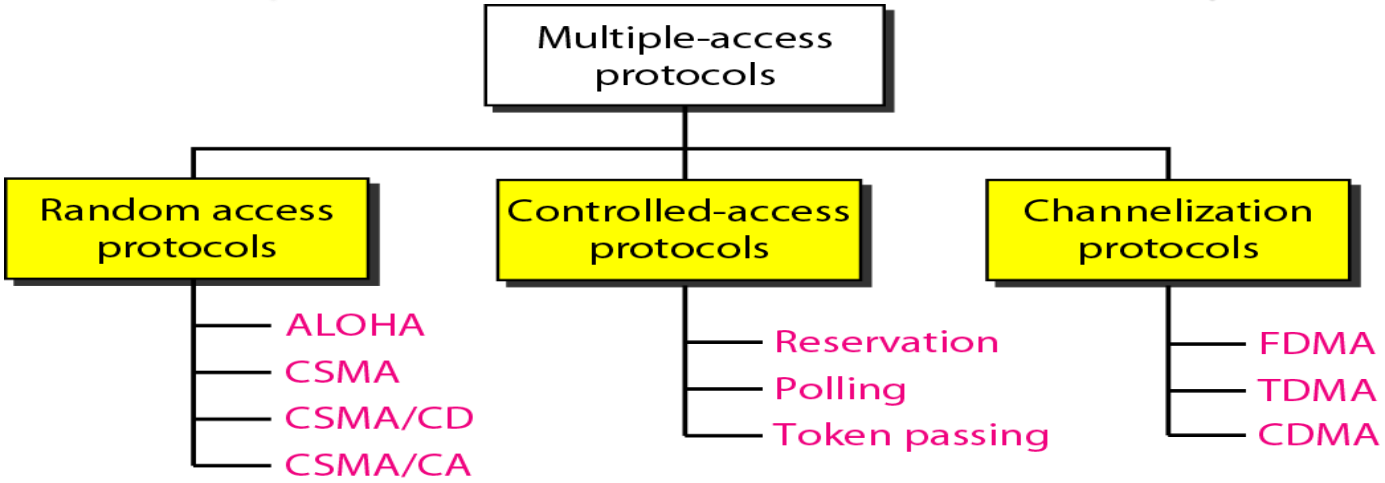
Multiple Access Protocols - Channelization

- **Channelization** is a multiple-access method in which the available bandwidth of a link is shared in frequency, time, or through code, between different stations
- No collision will occur in channelization protocols (channel partitioning) because each user has its own channel and users can transmit simultaneously without worrying about collision, however this may be an expensive protocol for the service provider (waste of resources) if the chance that users transmit simultaneously is very low



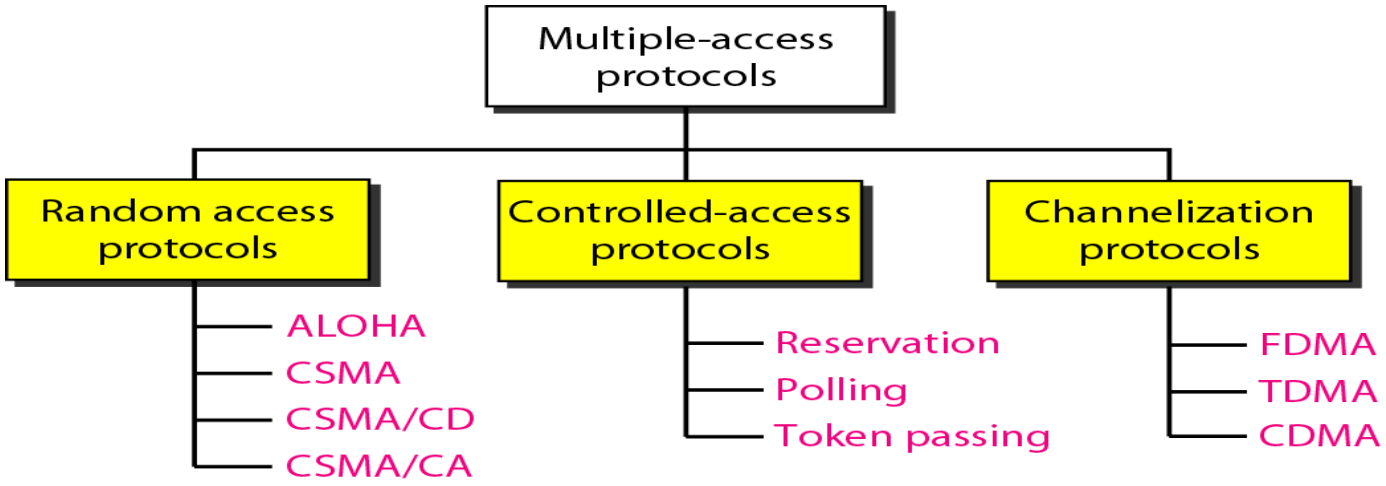
Multiple Access Protocols – Random Access

- In random access or contention methods, no station is superior to another station and none is assigned the control over another
- No station permits, or does not permit, another station to send
- At each instance, a station that has data to send uses a procedure defined by the protocol to make a decision on whether or not to send
- There is no scheduled time for a station to transmit. Transmission is random among the stations
- No rules specify which station should send next. Stations compete with one another to access the medium (that's the reason such protocols are also called contention protocols)



Multiple Access Protocols – Controlled Access

- In **controlled access**, the stations consult one another to find which station has the right to send
- A station cannot send unless it has been authorized by other stations
- Controlled access protocol is in between the two extreme protocols of random access and channelization
  - We will discuss only token passing



Random Access – ALOHA

- Was developed at the University of Hawaii in early 70s for wireless LAN, however it can be used on any shared medium
- A station transmits whenever it has data to transmit
- ALOHA applies the simplest access method: Just do it!
- The procedure is referred to as only MA (Multiple Access)
- If more than one frames are transmitted, they interfere with each other (collide) and are lost. Note that in general, the coexistence of even one bit from frame 1 and another bit from frame 2, means collision for frames 1 and 2
- ALOHA relies on acknowledgement
- If ACK is not received from receiver to the sender within timeout, then the sender picks random backoff time (to avoid repeated collision)
- The sender station retransmits frame after backoff time

Random Access – CSMA

- Carrier Sense Multiple Access

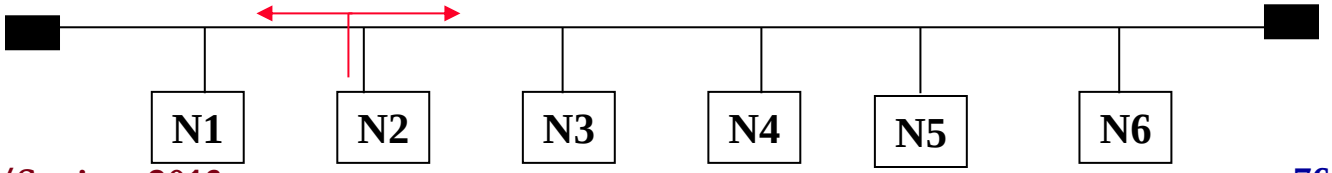
- CSMA later evolved into two parallel methods **CSMA/CA** (CSMA/**Collision Avoidance**) and **CSMA/CD** (CSMA/**Collision Detection**)
- CSMA/CA tries to avoid the collision and CSMA/CD tells the sender what to do when a collision is detected

Random Access – Ethernet (802.3)

- Ethernet is a proprietary protocol that was originally developed in 1976 by Xerox, then Intel and DEC (Digital Equipment Corporation) joined. Later on, IEEE modified it into 802.3, therefore any manufacturer can develop Ethernet as long as it adheres with IEEE802.3
- NIC card has layer 1 and 2, and so Ethernet as a layer 2 protocol has its own specific NIC called the Ethernet card
- The access method for Ethernet is referred to **CSMA/CD**
  - Consider a bus-based Ethernet. Assume a node (a station) has a frame to transmit; this happens in layer 2 (DL layer or MAC layer) of the node, which means layer 2 has a frame to transmit

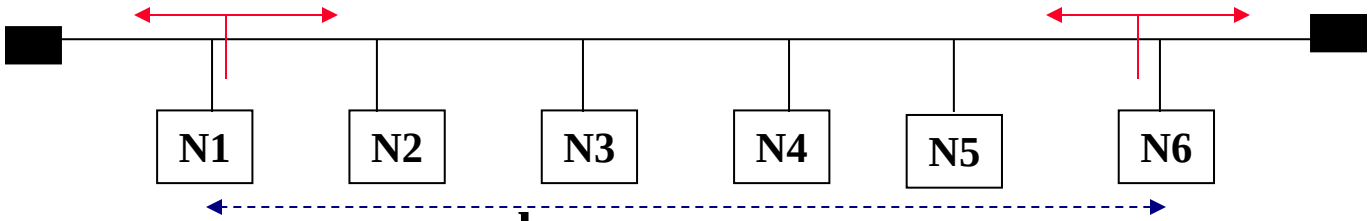
CSMA/CD and Ethernet (Cont.)

- The physical layer sensor will sense the activity if another station is transmitting. Layer 2 will wait as long as the physical layer of the node is sensing some activity
- As soon as the physical layer finds the medium idle, it will inform layer 2. Layer 2 then gives the frame to the physical layer to put in the channel to travel. Remember that frame from physical layer's point of view is unstructured set of bits. The NIC of all other nodes and the end buses will receive the data, bit by bit. Bus end will absorb the bits
- Question: Sensing helps to avoid collision, but how could it guarantee a collision-free channel?



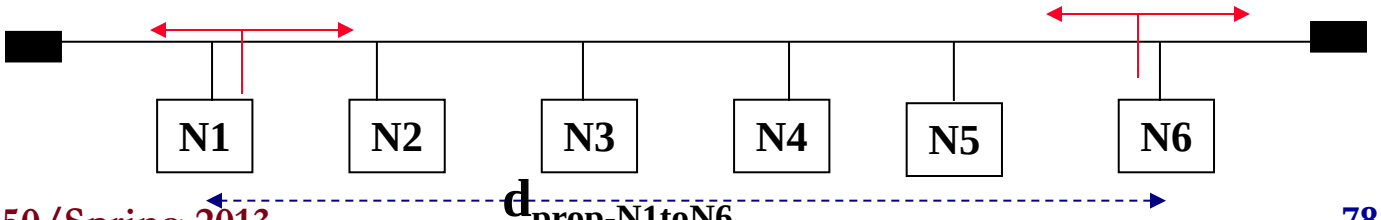
CSMA/CD and Ethernet (Cont.)

- What if the unstructured set of transmitted bits by N2 have not yet propagated to (arrived at) a certain node, say N5, then if N5 has a frame to transmit; its physical layer finds the channel idle and hence N5 will transmit its frame as well, and then… Bboomm! Collision!
- One worst case scenario is N1 as the transmitter and N6 as the last node to sense activity on the channel, because it sees the highest propagation delay from N1 to itself. This means it is more likely for N6 than others, to sense the channel idle, while in fact N1 has already sent the frame into the channel. If N6 transmits right at the time N1 also transmits or latest after $d_{prop-N1toN6}$ a collision occurs. Collision results in much higher signal voltages



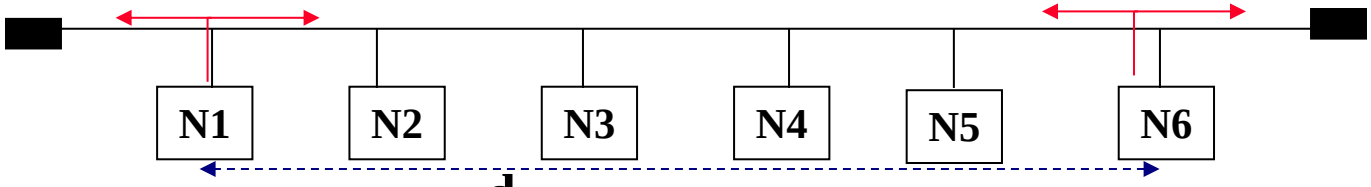
CSMA/CD and Ethernet (Cont.)

- Meanwhile N1 has not yet known the collision so it will keep sending the bits that are part of the frame, it is trying to transmit. Transmission on top of collision makes things even worse
- Imagine I am driving 120MPH down south I5 freeway, and suddenly there is this truck moving towards me in the opposite direction! I sense too late! my car collides with it; in less than a second the frame is lost ☺
- The cars behind me will keep colliding, until they can sense the collision soon enough and finally stop (stop transmitting)



CSMA/CD and Ethernet (Cont.)

- Back to our Ethernet collision, the collided signal will travel and finally arrive at N1 (after RTT,) so N1's NIC sensor senses it and sends a signal to its layer 2. Layer 2 then stops sending the rest of the bits and retransmits after a backoff time which is random
- Transmission is typically baseband. Radio-based MAC is also possible but has a different modem (Ethernet radio modem)
- Layer 2 of the transmitter node will keep a copy of the frame while it's being transmitted, because in case of collision, it has to resend the whole frame



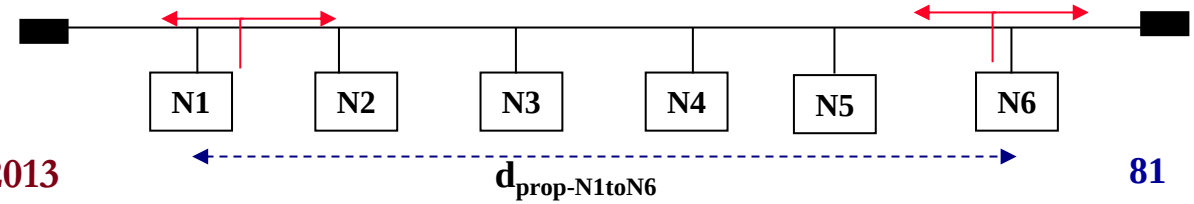
CSMA/CD and Ethernet (Cont.)

- To have a random backoff time, a random number generator is embedded into the NIC of every station
- A transmitter may experience collision after collision!
- Note: More collisions mean higher latency for the frame delivery and lower throughput
- It is possible in Ethernet (at least in theory) that a frame will always face collision after collision, so it will never be received by the destination!
  - Higher number of nodes increases the chance
  - That is the reason that a limit on the number of nodes on the bus is considered especially if it is known that the traffic will be heavy

CSMA/CD and Ethernet (Cont.)

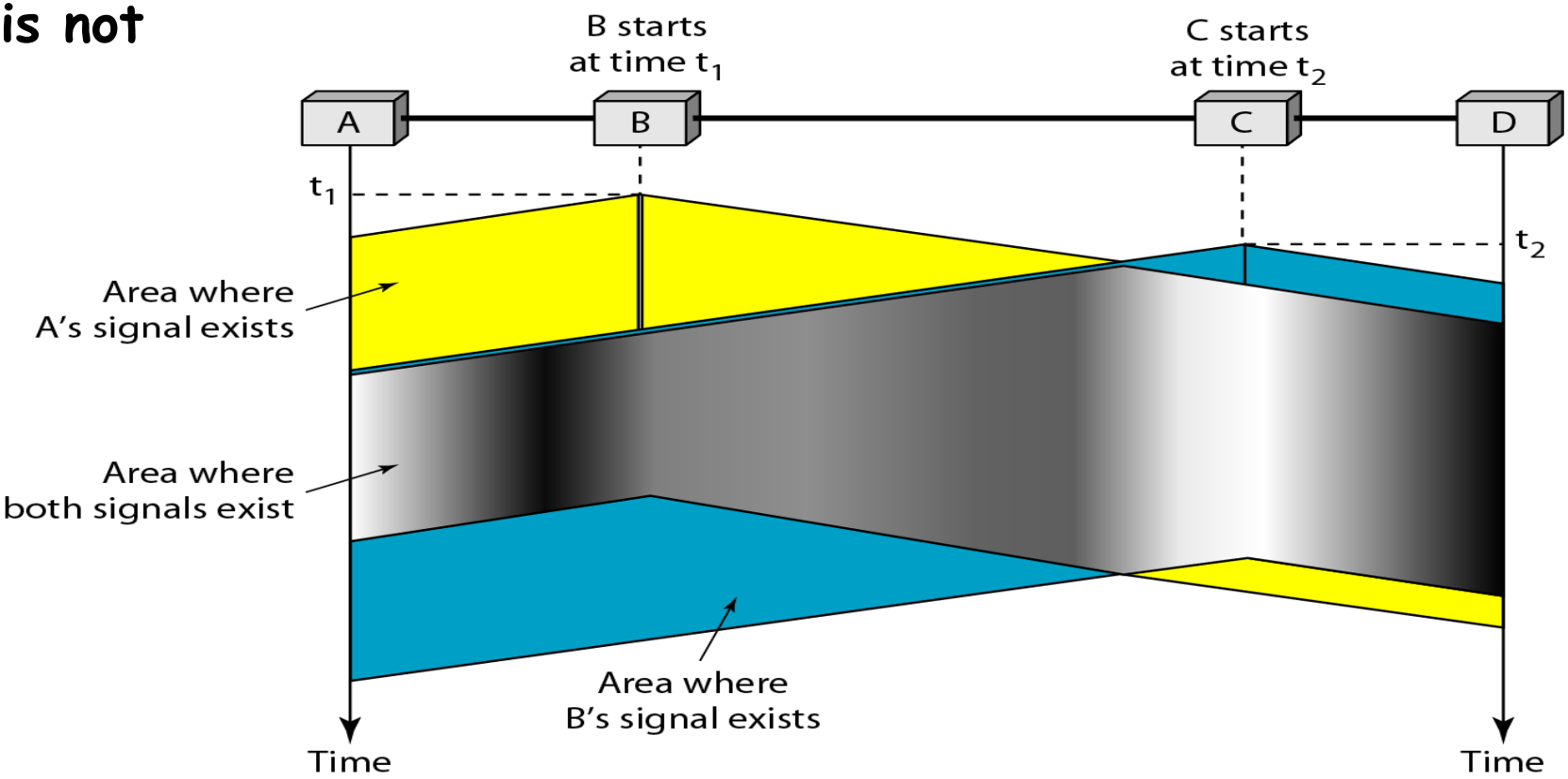
- In protocol design, we are interested in worst case scenarios to see how effective the protocols are and also prepare ourselves for the most terrible situation. Remember that the worse case scenario is when N1 and N6 transmit
- Question: N1 transmits and the bits will collide with the transmitted bits by N6. Assume the transmission delay of the frame, $d_{trans-frame}$ is smaller than $RTT=2d_{prop}$. Therefore N1 will be done transmitting before it finally senses the collision in this case. Will N1 be able to tell if the collision happened to its frame? No, N1 won't be able to tell whether the collision involved its frame or its frame was safely transmitted, and the collision involved other frames

$$d_{trans-frame} \geq RTT=2d_{prop}$$



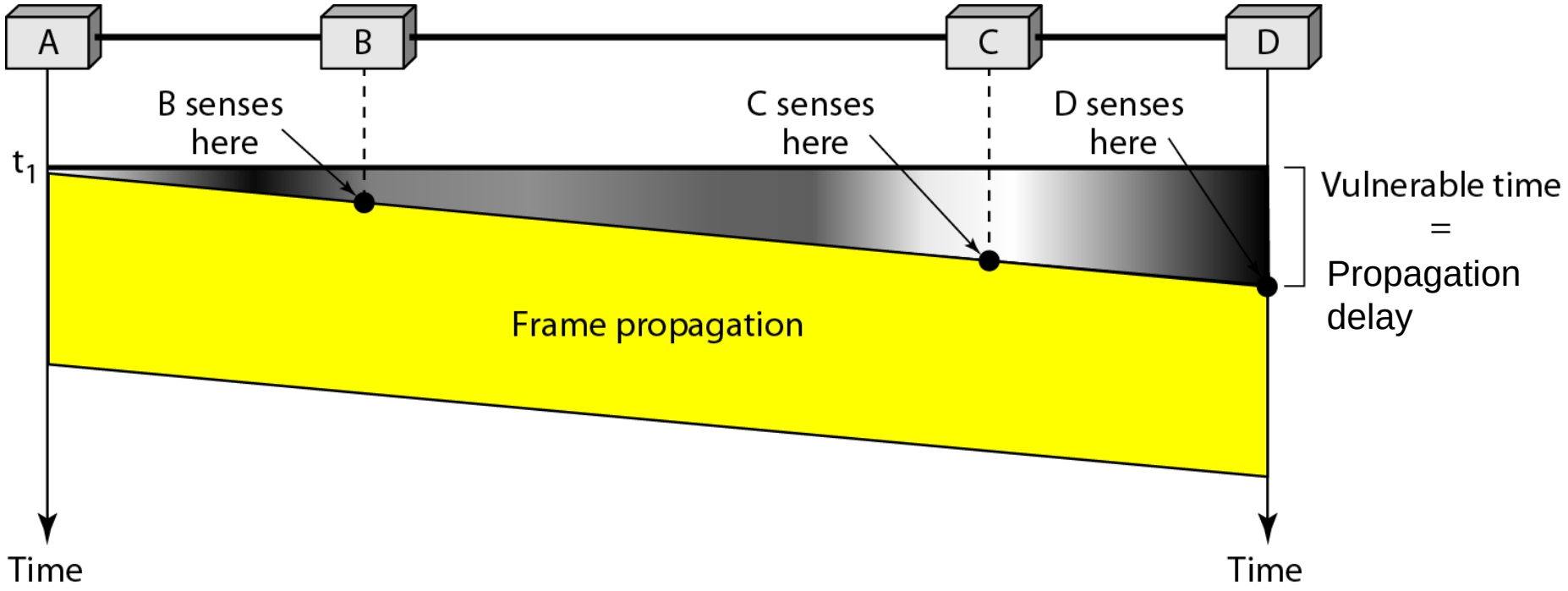
CSMA – Space/time Model of the Collision

- This space/time model shows that the possibility of collision exists; so it cannot be completely eliminated
- When a station sends a frame, it still takes time for the 1st bit to reach every station and for every station to sense it, therefore a station may sense and find the medium idle, when it is not



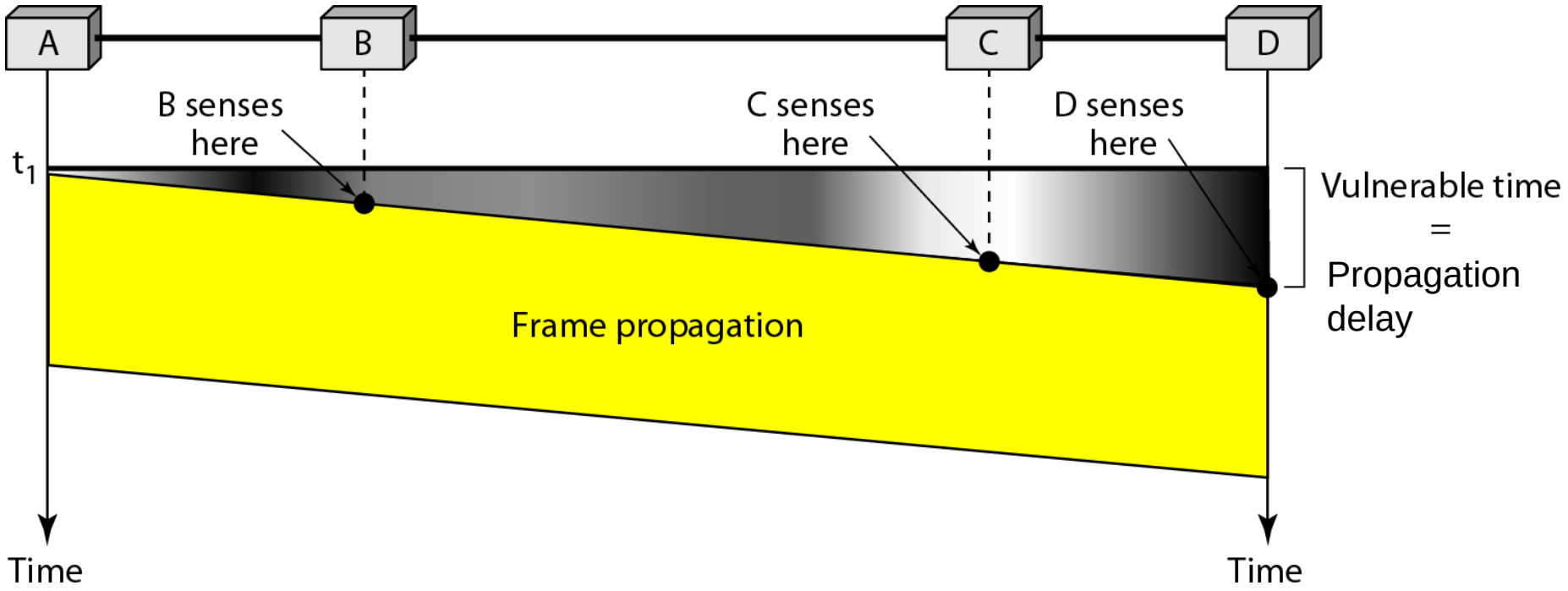
Vulnerable Time in CSMA

- The **vulnerable time** for CSMA (i.e., the time for which there is a possibility of collision) is d_{propr}
- When a station sends a frame and any other station tries to send a frame during this time, a collision will result
- But if the first bit of the frame reaches the end of the medium, every station will already have heard the bit and will refrain from sending



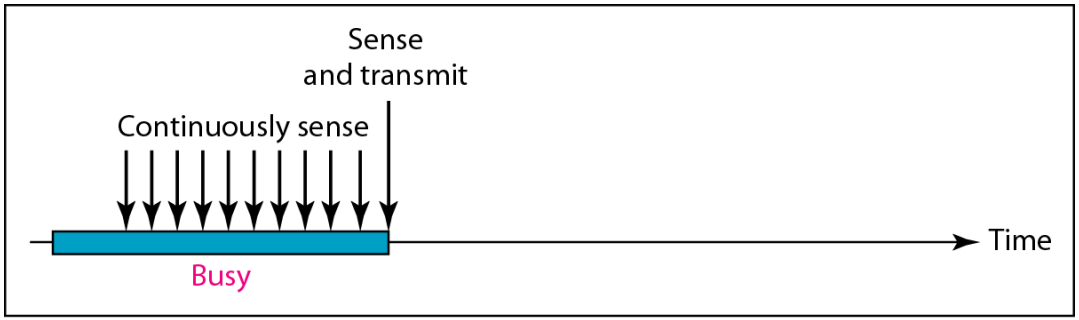
CSMA/CD and Ethernet (Cont.)

- We should make sure $d_{trans-frame} \geq RTT = 2d_{prop}$
- Therefore frame length cannot be too small
- $d_{trans-frame} \geq RTT$ will give a lower bound on the length of the frame. Furthermore, frame length has an upper bound as well to make the protocol fair for all nodes

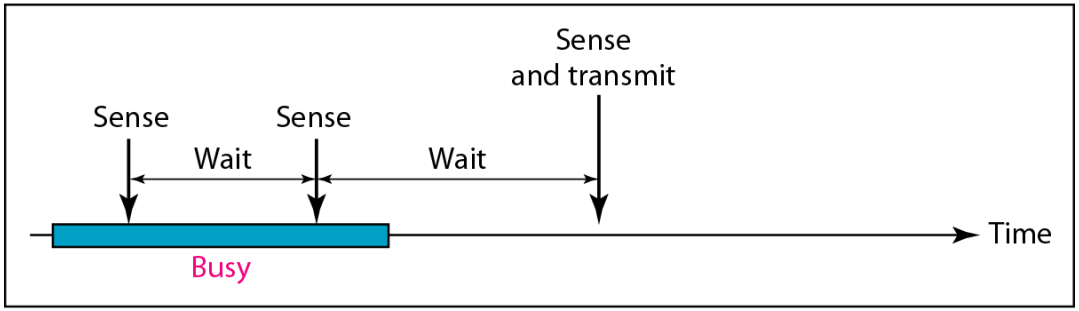


CSMA – Persistent Methods

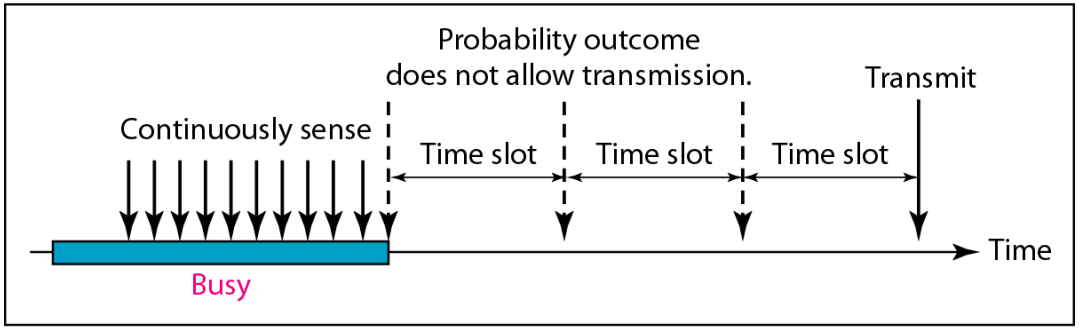
- Question: What should a station do if the channel is busy? What should a station do if the channel is idle?
- Three methods are devised:
  - a. 1-persistent (aka persistent)
  - b. Non-persistent
  - c. P-persistent



a. 1-persistent



b. Nonpersistent



c. p-persistent

CSMA – Persistent Methods (Cont.)

- In 1-Persistent the station senses the media continuously; it keeps sensing while the channel is busy and transmits as soon as channel is idle

- In non-persistent if the station senses the media as busy, it waits a random period of time and senses again, if busy again, it waits for a longer period of time
  - This reduces the chance of collision
  - However, it reduces the throughput as the media might remain idle while stations have frames to transmit

CSMA – Persistent Methods (Cont.)

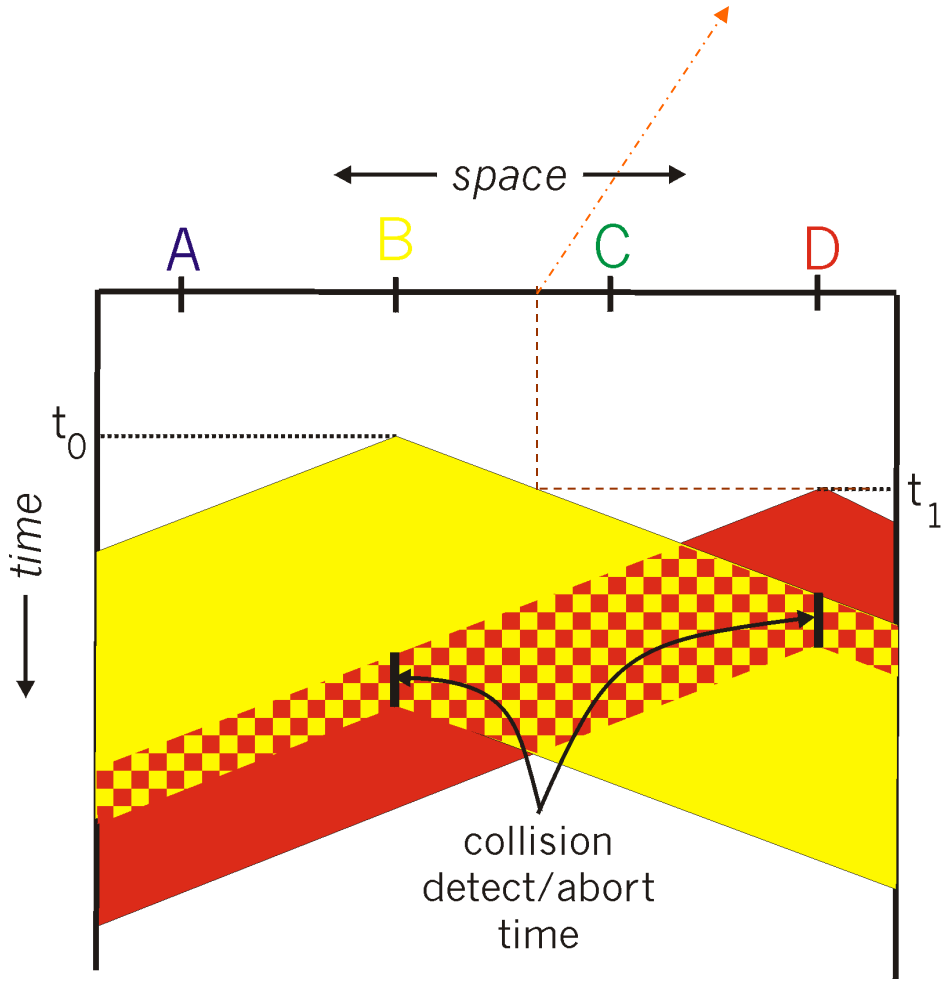
- P-persistent (probabilistic) is not popular. It is used if the channel has time slots with a slot duration equal to or greater than the maximum d_{prop}
- The stations keeps sensing and after it finds the media to be idle:

1. With probability p it will transmit
2. With probability 1-p the station waits for the beginning of the next time slot and checks the line again
   - If media is idle, it goes to step 1
   - If the line is busy, it acts as though a collision has occurred and uses the backoff procedure

- P-persistent tries to combine the advantages of the other two methods (reduces the chance of collison while improving throughput)

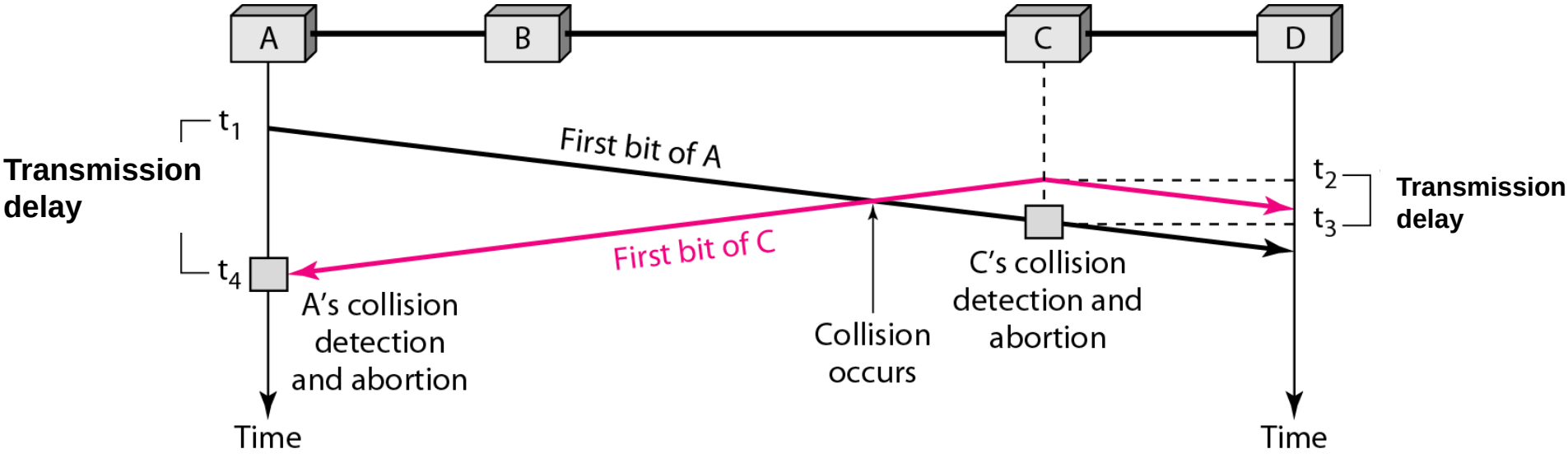
CSMA/CD Collision detection

First bit of B arrives at this point
When D starts transmitting



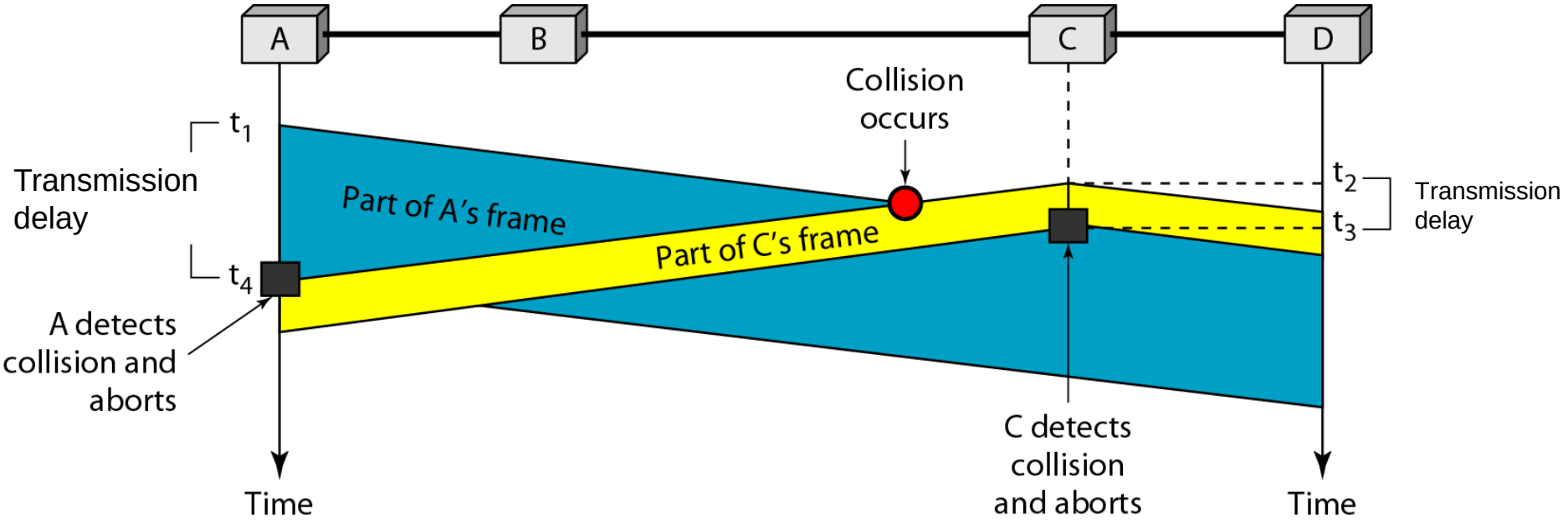
Collision of the First Bit in CSMA/CD

- When collision occurs, the collided signal propagates both ways



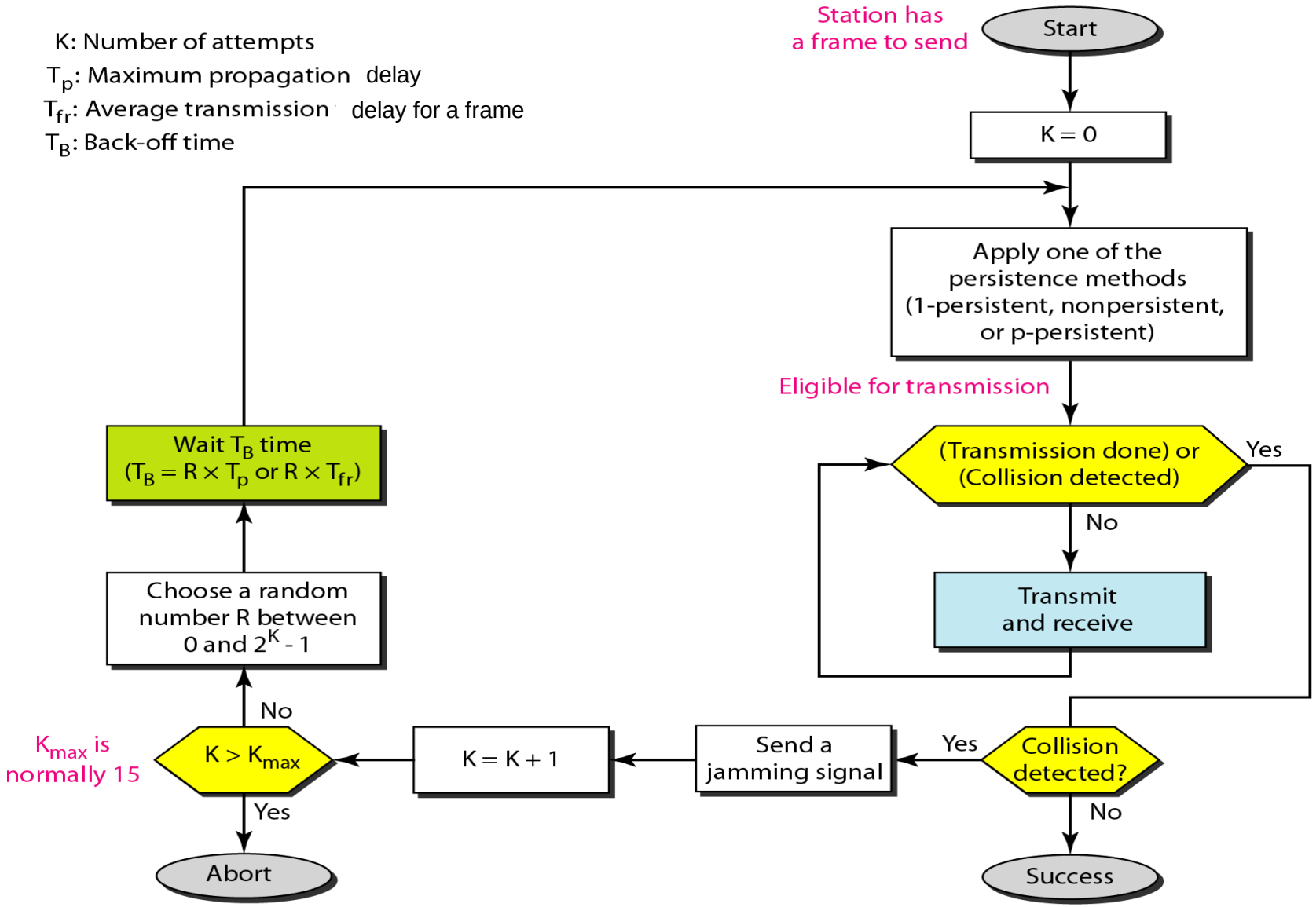
Collision and Abortion in CSMA/CD

- In this example, station C is closer to the collision's point of occurrence, so it detects the collision earlier than A



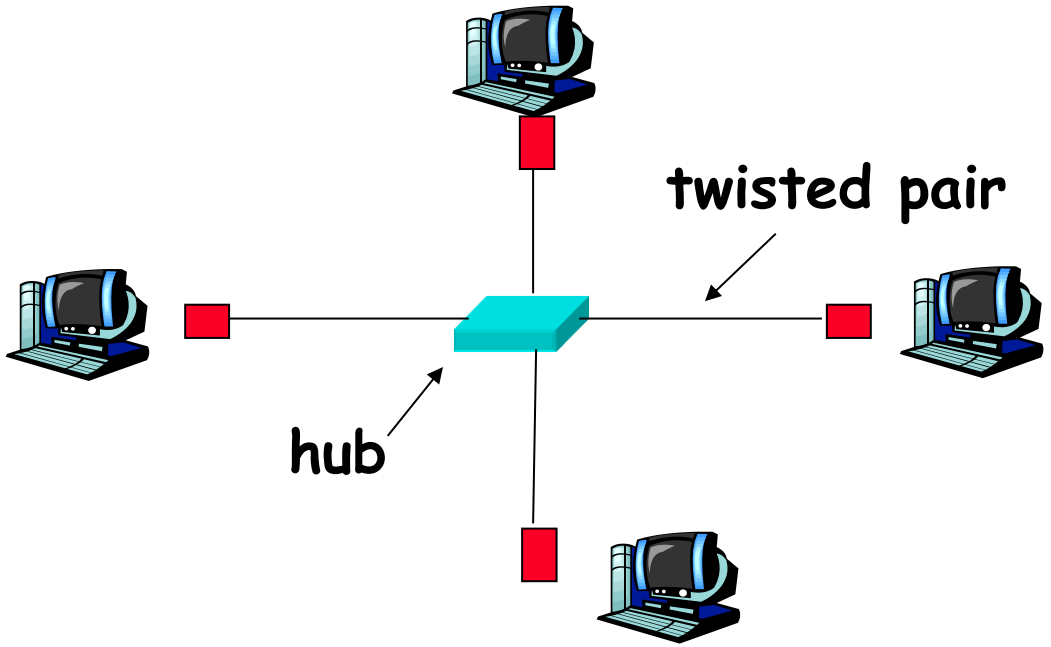
- **Jam signal**: Any node that detects a collision sends a jam signal, alarming other nodes about the collision. This is to ensure that all nodes will effectively know a collision has happened. The node sending the jam signal may not be necessarily one of the nodes transmitting

Flow diagram for CSMA/CD



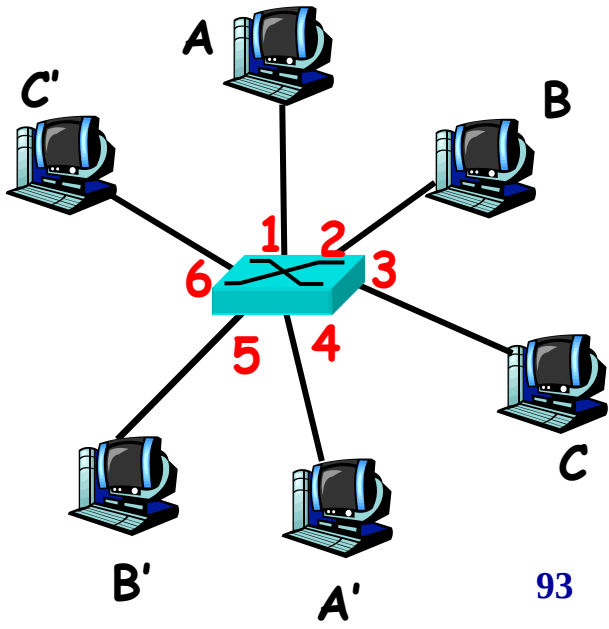
Hub-based Ethernet

- Hub is a physical layer device: whenever it receives a bit from one of its nodes it sends a copy of that bit to all other nodes, with the same rate
- If a hub receives multiple bits from two or more different nodes at the same time a collision occurs
- It then sends a Collision Detect signal

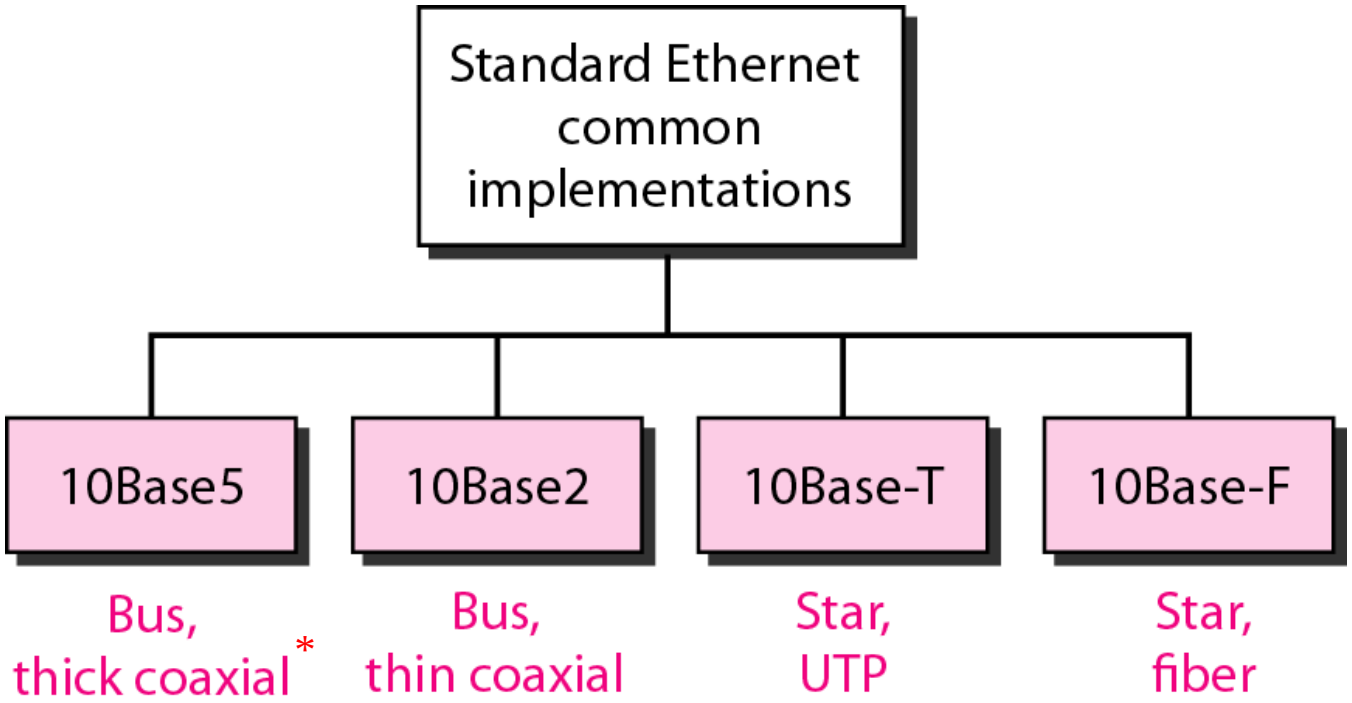


Switch-based Ethernet

- In the early 2000s Ethernet standard replaced hub with a switch (a layer 2 device) which is not only collisionless but also a store-and-forward packet switch, but unlike routers which have three layers, an Ethernet switch has only two layers
- **Switching**: A-to-A' and B-to-B' can be simultaneous, without any collision. This is not possible with hub



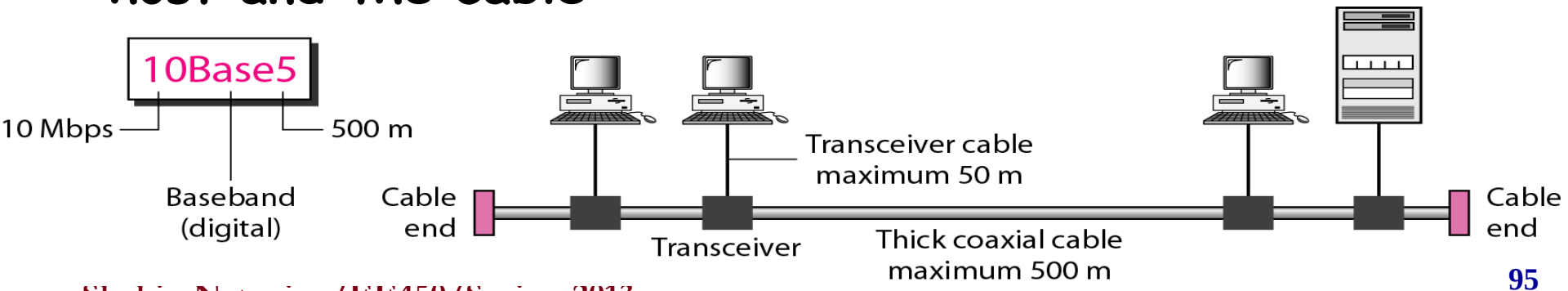
Categories of Standard Ethernet



* Thick implies that the cable has protection against interference, noise, etc. Thin does not

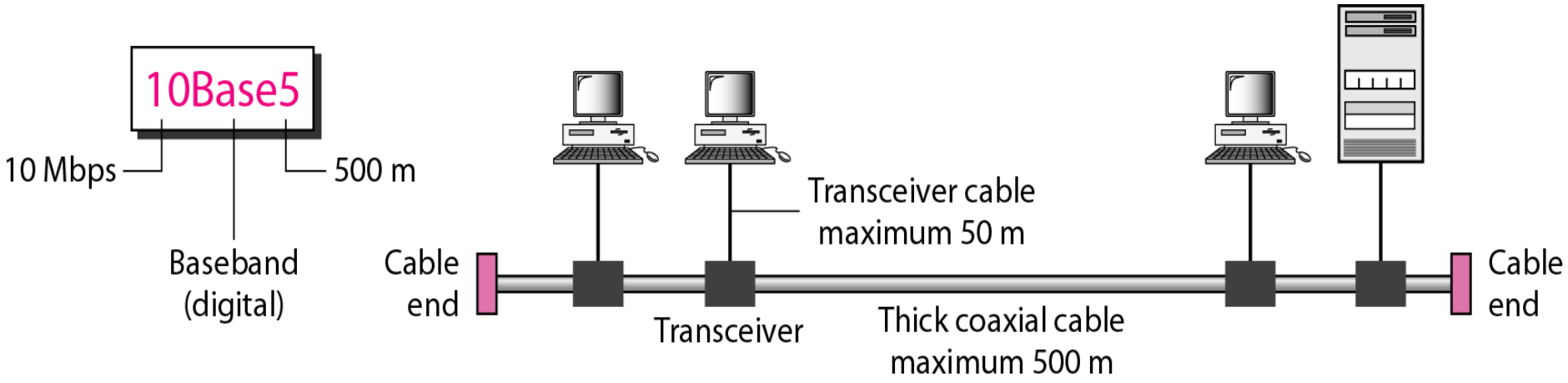
10Base5 Implementation

- NIC in general embeds the Physical and DL layers, however here the Physical layer (i.e., the transceiver) is brought out of the NIC card and put on the cable. This is because the cable is thick, and it is not desirable to bring it directly to the end systems (hosts.) Note: length is less than 500m to avoid degradations
- Instead of connecting stations directly to the cable, the physical layer of each station is brought out of NIC and put on the cable as the interface between host and the cable



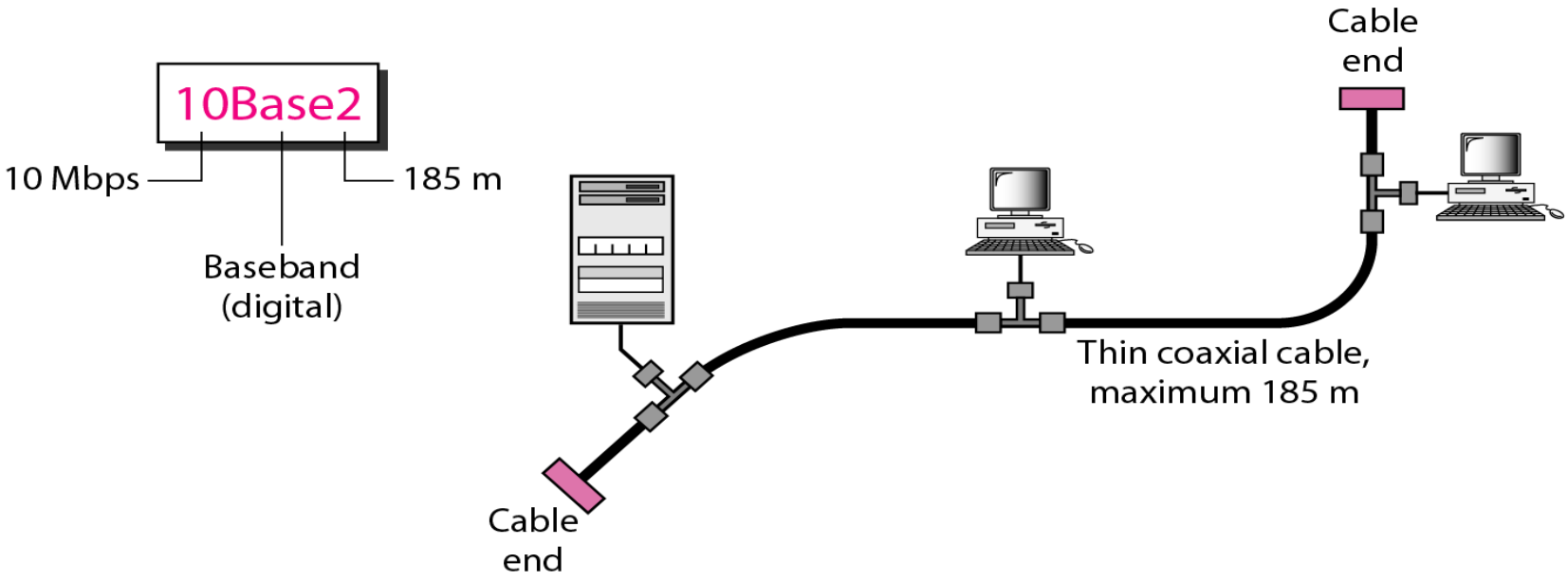
10Base5 Implementation (Cont.)

- Question: in case of collision is it the NIC card that is sensing it?
  - Not here, because physical layer is responsible for sensing the medium and in this case, physical layer is brought out of the NIC. Transceiver as the physical layer is responsible for sensing collisions



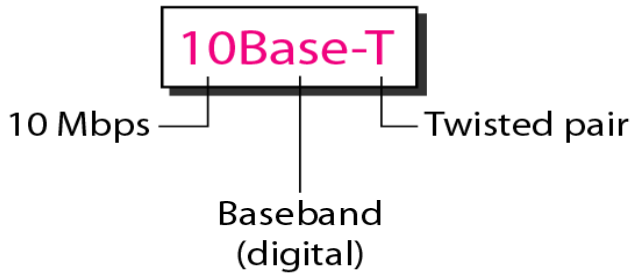
10Base2 Implementation

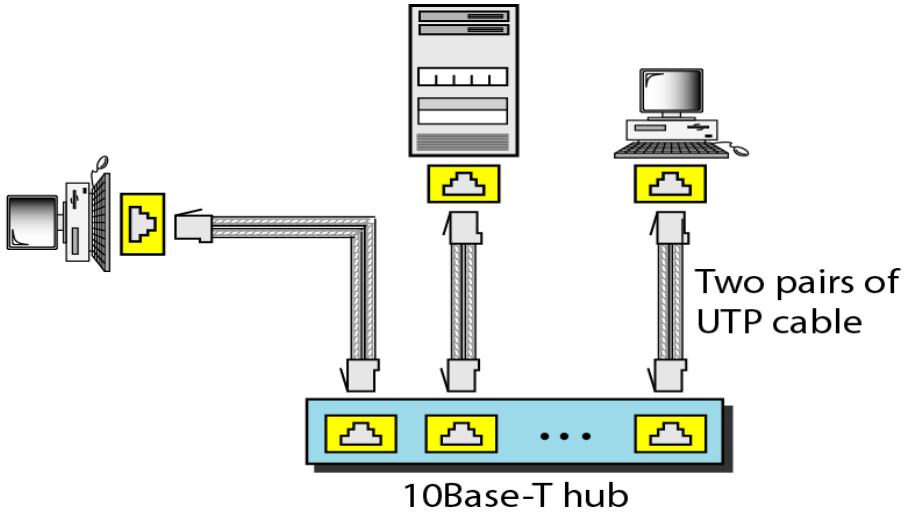
- A T-type connection is used and in this case, since the cable is thin, the NICs are normal, i.e., the physical layer is not separated from the NIC
- The length of thin cable should be a lot smaller than thick cables, because of its high level of attenuation



10Base-T Implementation

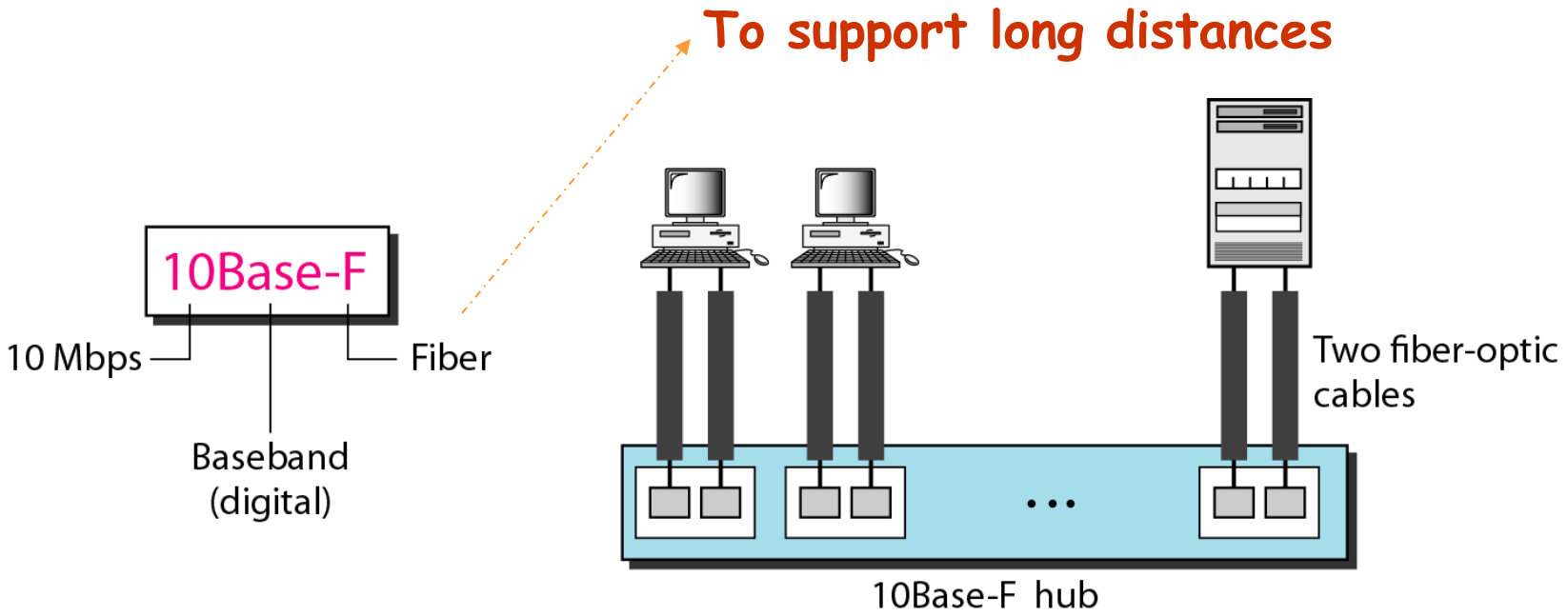
- 10Base-T uses a physical star topology and the stations are connected to a hub via two pairs of twisted cable
- Two pairs of twisted cable create two paths btn hub and station: one is for sending and one for receiving
- Any collision here happens in the hub. Compared to 10Base5 and 10Base2 we can see that the hub actually replaces the coaxial cable as far as a collision is concerned. Maximum twisted cable length is 100m to minimize the effect of attenuation





10Base-F Implementation

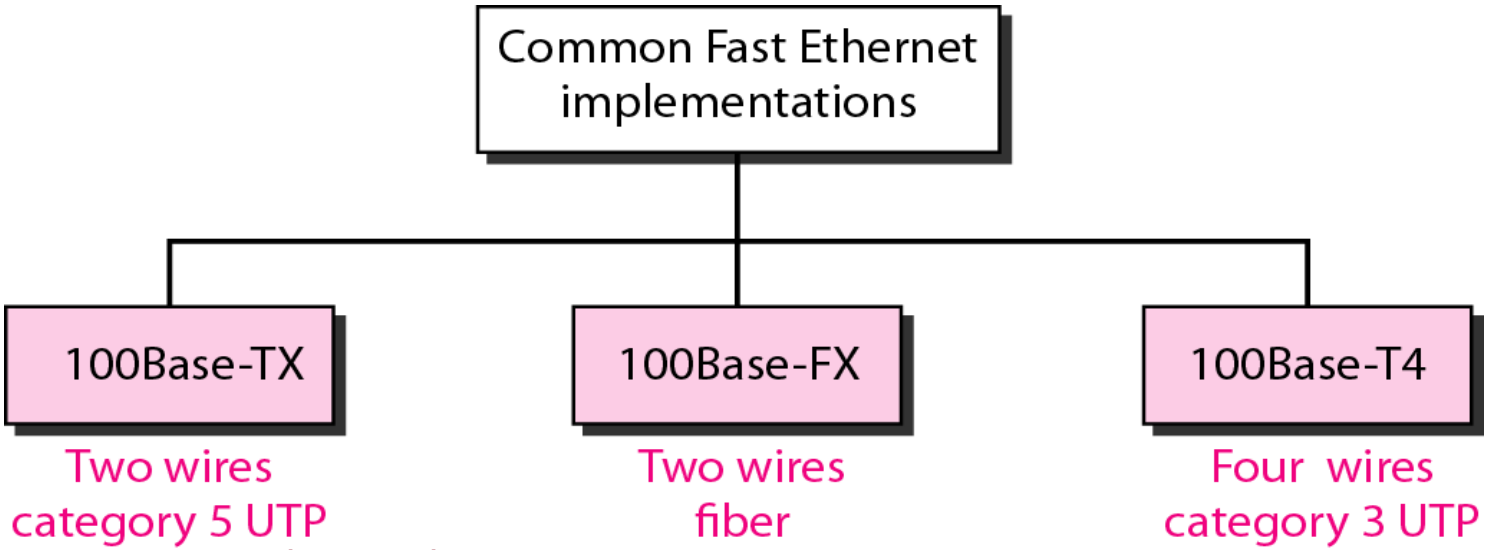
- There are several types of optical fiber 10Mbps Ethernet, however the most common is called 10Base-F or Fiber Ethernet
- It uses a star topology with a hub. Stations are connected to the hub using two fiber-optic cables



(Optional) Fast Ethernet Implementations

Characteristics	*100Base-TX*	*100Base-FX*	*100Base-T4*
Media	Cat 5 UTP or STP	Fiber	Cat 4 UTP
Number of wires	2	2	4
Maximum length	100 m	100 m	100 m
Block encoding	4B/5B	4B/5B	
Line encoding	MLT-3*	NRZ-I	8B/6T

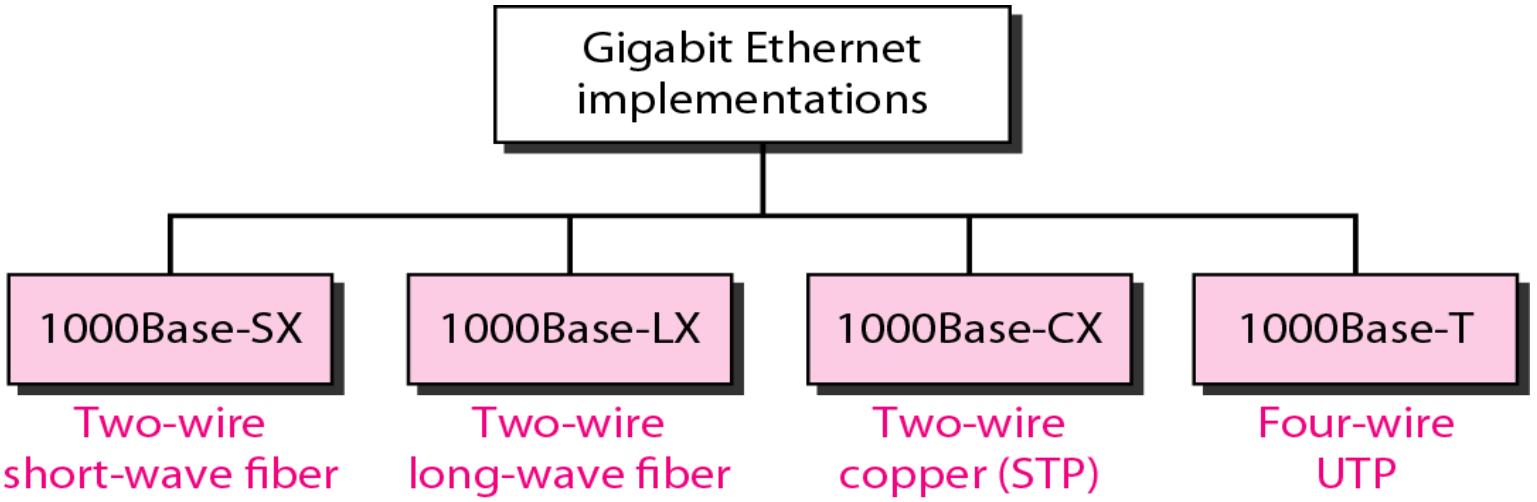
* MLT-3 uses 3 voltage levels: http://en.wikipedia.org/wiki/MLT-3_encoding



(Optional) Gigabit Ethernet Implementations

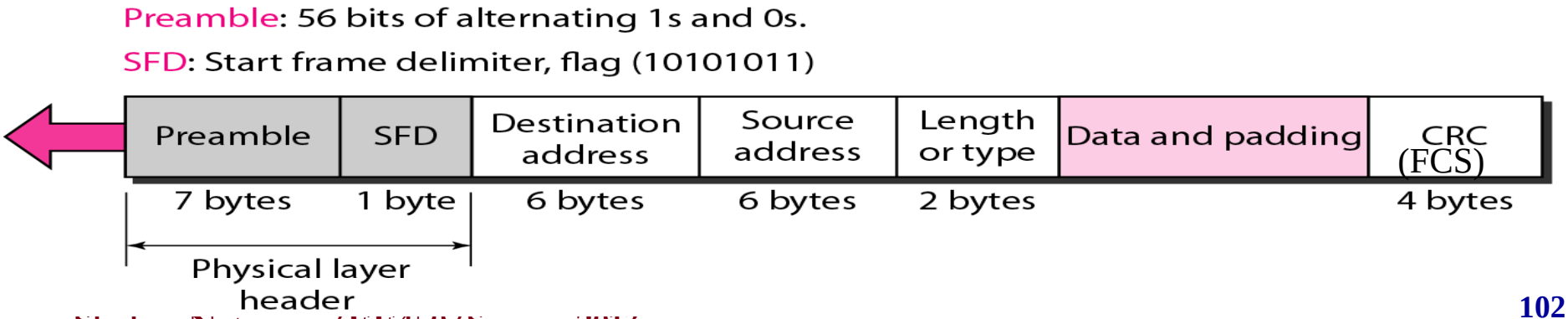
- **Ethernet has proved to be a very scalable technology, i.e., the data rate can be increased without increasing the complexity of the protocol**

Characteristics	*1000Base-SX*	*1000Base-LX*	*1000Base-CX*	*1000Base-T*
Media	Fiber short-wave	Fiber long-wave	STP	Cat 5 UTP
Number of wires	2	2	2	4
Maximum length	550 m	5000 m	25 m	100 m
Block encoding	8B/10B	8B/10B	8B/10B	
Line encoding	NRZ	NRZ	NRZ	4D-PAM5



IEEE 802.3 MAC Frame

- **Preamble**: 56 bits of alternating 0s and 1s that alerts the receiving system to the coming frame and enables it to synchronize its input timing. 56-bit pattern allows the stations to miss some bits at the beginning of the frame. Preamble is added at the physical layer and is not formally part of the frame
- **SFD** signals the beginning of the frame. SFD warns the station or stations that this is the last chance for synchronization! The last 2 bits are 11 and alerts the receiver that the next field is the destination address

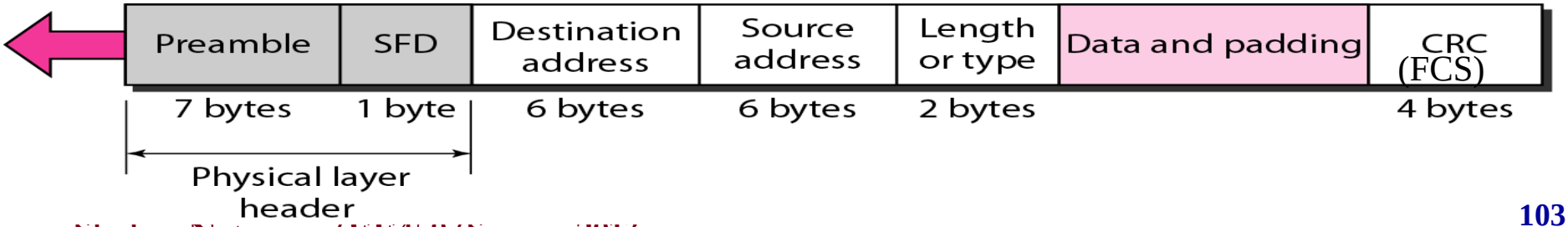


IEEE 802.3 MAC Frame (Cont.)

- Length or type: The original Ethernet used this field as the type field to define the upper layer protocol using the MAC frame. IEEE standard uses it as the length field to define the number of bytes in the data field. Both uses are common today
- The main difference between this frame and a PPP frame is that this one needs two addresses, whereas the PPP frame has only one address (which in fact is not even needed)

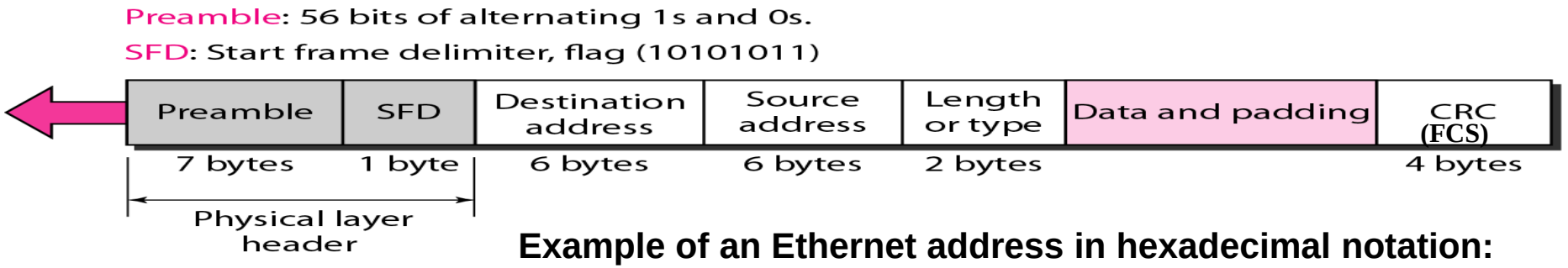
Preamble: 56 bits of alternating 1s and 0s.

SFD: Start frame delimiter, flag (10101011)



IEEE 802.3 MAC Frame (Cont.)

- Also in the frame of point to point protocols a field can be added to keep the sequence bits (for sequence number of the frames) however here the frame does not need that field. E.g., the "Control" field can be used in the PPP frame for error and flow control, MAC frame does not have the control field
- 802.3 MAC frame data needs to be at least 46 bytes. Padding in data is to make sure the frame is not too short. Data & padding includes IP addresses, but to layer 2's eyes they mean nothing more than pure data



Example of an Ethernet address in hexadecimal notation:

06 : 01 : 02 : 01 : 2C : 4B

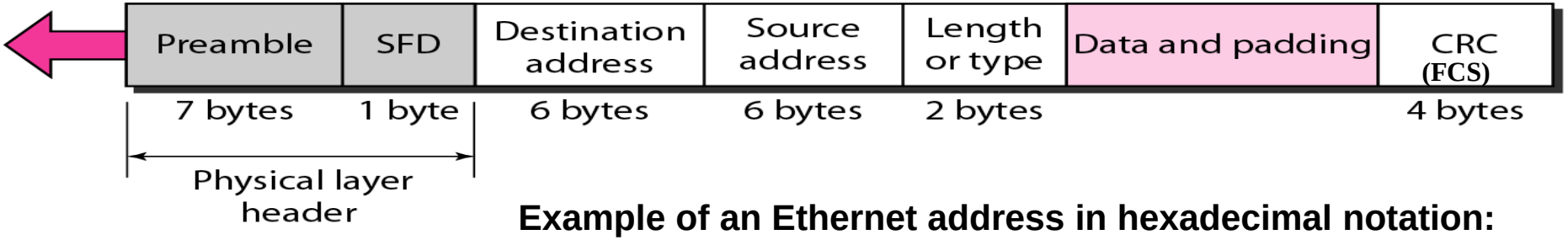
6 bytes = 12 hex digits = 48 bits

IEEE 802.3 MAC Frame (Cont.)

- There is no ending flag here, because length of the frame is given
- The maximum size of data and padding (aka the **Maximum Transmission Unit** or **MTU**) is 1500 bytes. Typically this is the part that represents the frame length, meaning when we mention the frame length we may not take into account the flags, and preamble, etc.
- It is Network layer's responsibility to make sure the message is fragmented small enough to be maximum 1500 bytes

Preamble: 56 bits of alternating 1s and 0s.

SFD: Start frame delimiter, flag (10101011)



Example of an Ethernet address in hexadecimal notation:

06 : 01 : 02 : 01 : 2C : 4B

6 bytes = 12 hex digits = 48 bits

(Optional) Frame Transmission Example

- Show how the address 47:20:1B:2E:08:EE is sent out on line
- Solution
- The address is sent left-to-right, byte by byte; for each byte, it is sent right-to-left, bit by bit, as shown below:

IEEE 802.3 MAC Frame (Cont.)

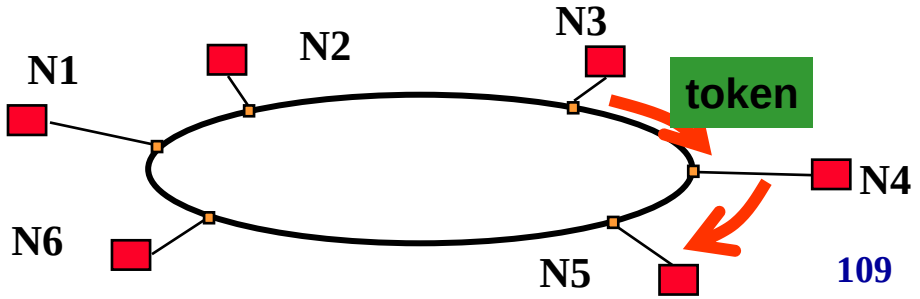
- Reminder: MAC addresses are globally unique, even if they just need to be locally unique (unique in LAN)
- IEEE assigns the MAC addresses: the 1st 3 bytes, represent the vendor (http://standards.ieee.org/regauth/oui/oui.txt) and the next 24 bits are assigned by the vendor
- Destination MAC address used in the frame could be unicast, broadcast (FF:FF:FF:FF:FF:FF) or multicast

IEEE 802.3 MAC Frame (Cont.)

```
Windows IP Configuration
        Host Name . . . . . . . . . . . . : OHE340-PC03
        Primary Dns Suffix  . . . . . . . :
        Node Type . . . . . . . . . . . . : Unknown
        IP Routing Enabled. . . . . . . . : No
        WINS Proxy Enabled. . . . . . . . : No
        DNS Suffix Search List. . . . . . : usc.edu
Ethernet adapter Local Area Connection:
        Connection-specific DNS Suffix  . :
        Description . . . . . . . . . . . : Intel(R) PRO/100 VE Network Connection
        Physical Address. . . . . . . . . : 00-03-24-42-3E-39
        Dhcp Enabled. . . . . . . . . . . : Yes
        Autoconfiguration Enabled . . . . : Yes
        IP Address. . . . . . . . . . . . : 190.118.0.2
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 190.118.1.1
        DHCP Server . . . . . . . . . . . : 190.118.1.1
        DNS Servers . . . . . . . . . . . : 190.118.1.1
        Lease Obtained. . . . . . . . . . : Wednesday, July 01, 2009 6:48:44 PM
        Lease Expires . . . . . . . . . . : Thursday, July 02, 2009 6:48:44 PM
```

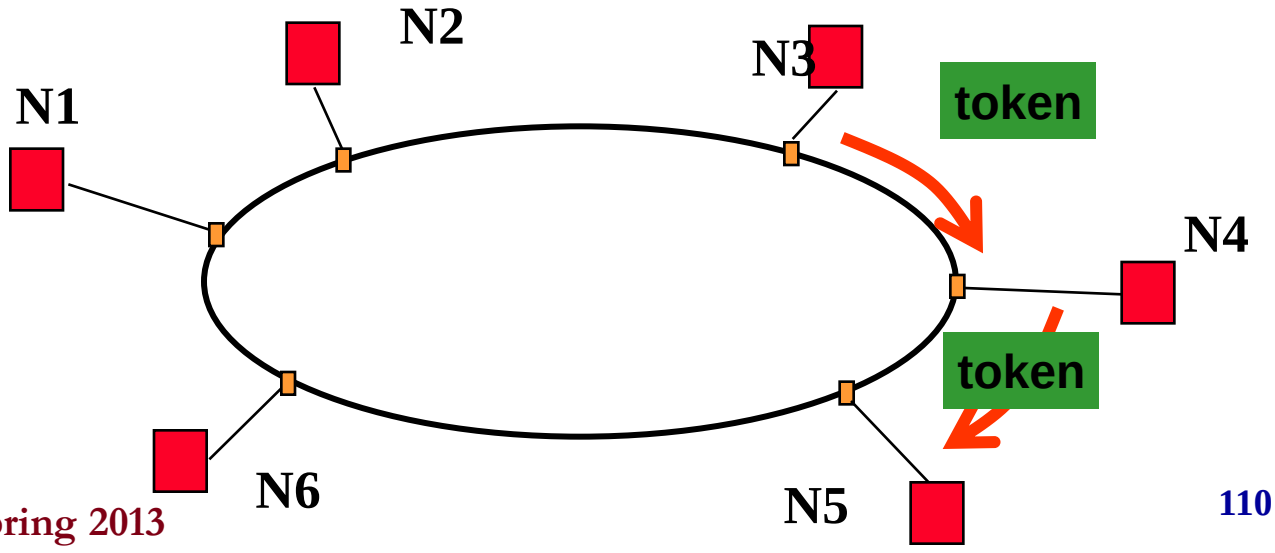
Controlled Access – Token Ring (802._)

- IBM popularized the use of token ring LANs in the mid 1980s to compete with Ethernet and also to remove the randomness of Ethernet (theoretically a frame in Ethernet may never reach destination due to collisions) by using round robin (taking turn.) Token ring also matched closer, the IBM structure of master mainframe and slave terminals more than Ethernet
- IEEE later modified and standardized it as 802.5
- Token Ring is not as popular as Ethernet in the US
- Among the nodes connected to the ring, one of them is the monitor. Here let's assume N1 is the monitor. Monitor issues a free token (of size 3 bytes)
- FDDI (Fiber Distributed Data Interface) has dual ring (Optional: http://www.cisco.com/en/US/docs/internetworking/technology/handbook/FDDI.html Not part of EE450



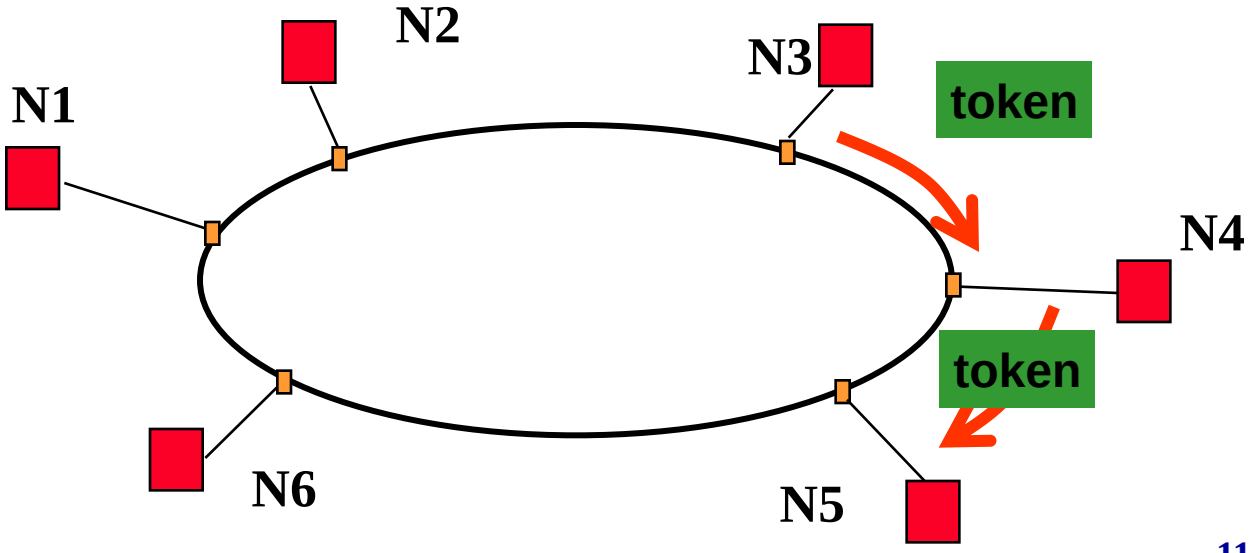
Controlled Access – Token Ring (802.5)

- Ring topology is unidirectional. Token is the node's ticket to transmit
- If N1 does not have any frame to transmit, it passes the token to N2, and similarly N2 would pass the token to N3 if it does not have any frame to transmit and so on & so forth
- Token will keep rotating if no node wants to transmit, and this is a waste of resources



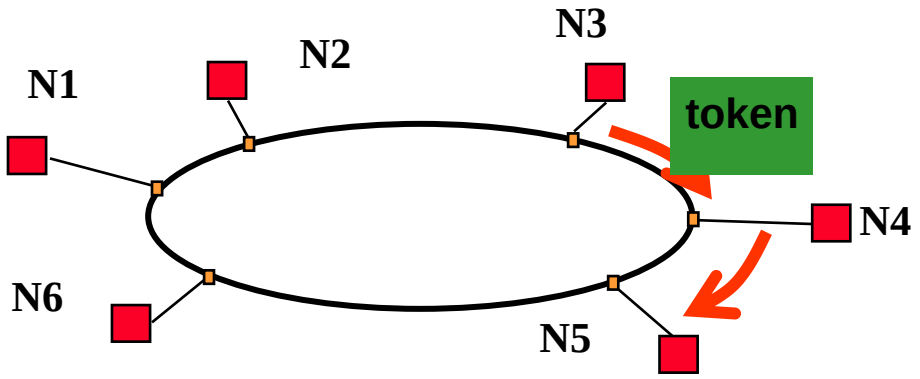
Controlled Access – Token Ring (Cont.)

- Free token and busy token
- N3 checks the data frame (destination MAC address) to see whether the frame is destined to it, in this case it's not, so N3 will not process the frame, instead N3 relays the frame and it will arrive at N4
- N4 may have its own frame to transmit, but finding the token busy, it refrains from transmission. Things are similar for N5



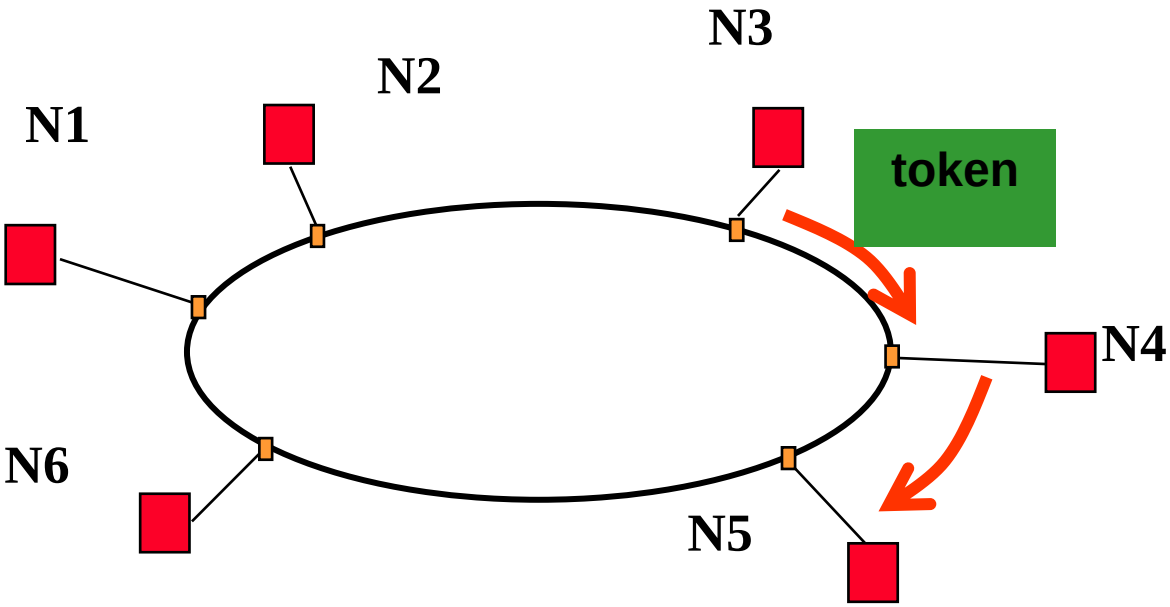
Controlled Access – Token Ring (Cont.)

- However in case of N6, it finds out that the frame is destined to it, so it processes the frame
- N6 then passes the payload (to be packet) to its network layer
- Note that the frame is busy, i.e., its token is busy. N1 then relays the frame into the ring and N2 receives its own frame that it had transmitted
- N2 knows only one frame can be transmitted at a time. It checks the source MAC address and so finds out it is the same frame it had just transmitted



Controlled Access – Token Ring (Cont.)

- N2 then removes the frame and also issues a free token to the ring
- Monitor issues the free token, but later whichever node that changed it to busy, is responsible to issue a free token



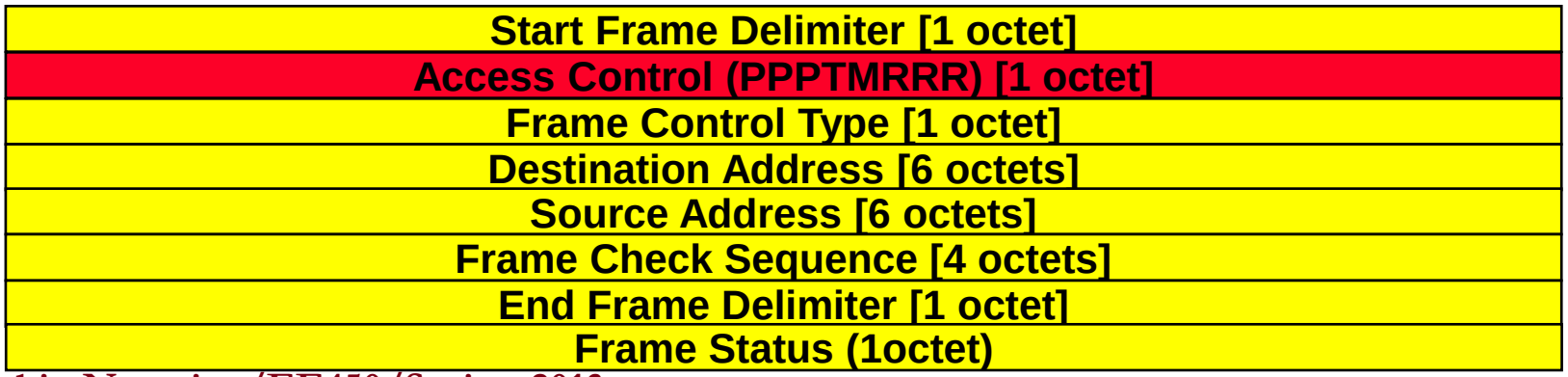
Token Ring Frame

- Collision in Token Ring
- To be fair, a node has a certain time to use the token (**Token Hodling Time** (**THT**) defined by the network administrator.) If the timer expires and the node has more frames to transmit, it needs to release the token and wait. Therefore, N2 cannot keep removing its transmitted frame and instead of issuing a free token, attach a new frame to the busy token endlessly
- **T**=1 is the full frame (busy token) **T**=0 is a token (free token)
- **M** is set to 0 by the sending station, to 1 by the monitor station

Start Frame Delimiter [1 octet]
Access Control (PPPTMRRR) [1 octet]
Frame Control Type [1 octet]
Destination Address [6 octets]
Source Address [6 octets]
Frame Check Sequence [4 octets]
End Frame Delimiter [1 octet]
Frame Status (1octet)

Token Ring Frame (Cont.)

- The purpose of the M bit is that monitor does not want to see that frame again. This means it wants to make sure N2 removes that frame:
  - When N2 sends the frame, M=0, T=1 (i.e., busy token.) When the frame is received by the monitor, N1, it sets M=1, (T still 1) then when N2 receives its frame back, it should remove the frame and issue a free token by setting T=0 and also it resets M, i.e., M=0

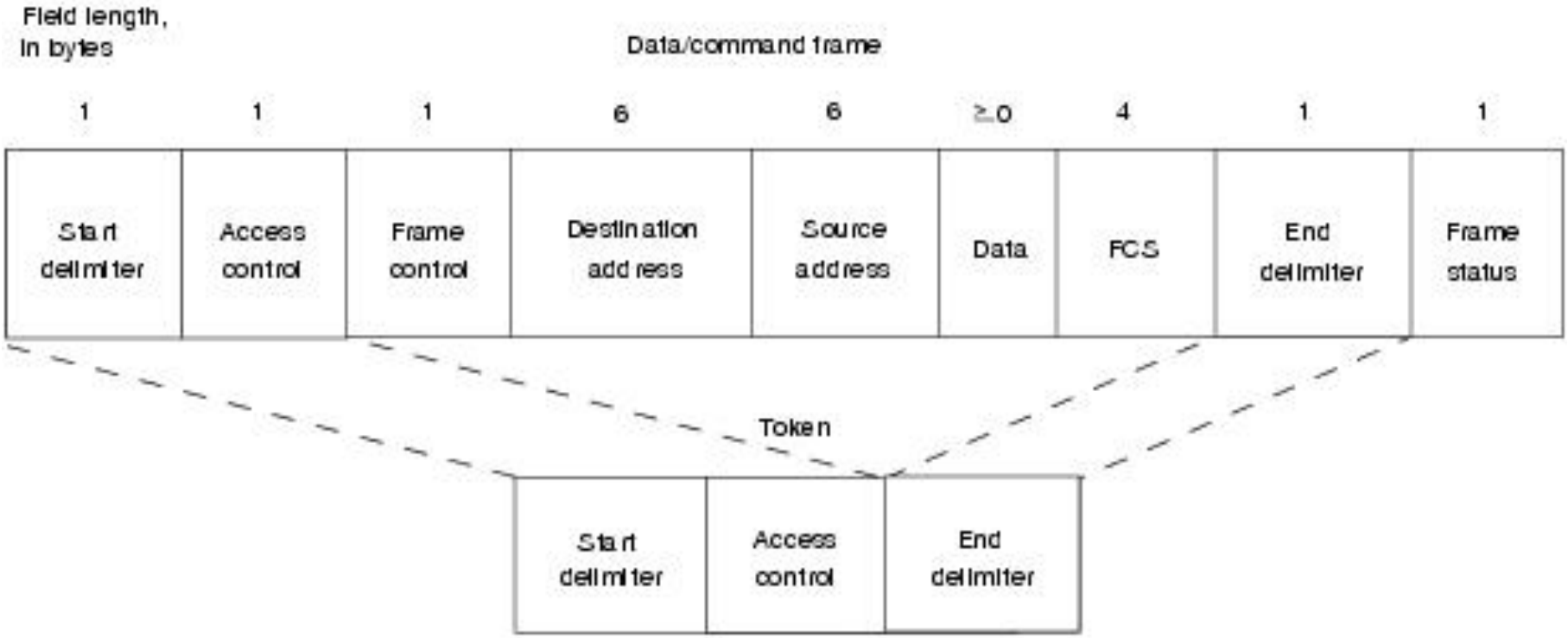


Token Ring Frame (Cont.)

- **Frame Check Sequence** is identical to 802.3 FCS field
- Addresses are 48 bits long; same as those in 802.3
- **Frame Status** is for reporting specific errors; a one byte field used as a primitive acknowledgement scheme on whether the frame was recognized and copied by its intended receiver. **Frame Control Type** specifies the type of frame (data or supervisory) [not our concern]
- Note: The 802.5 frame format is not compatible with that in Ethernet (802.3) This means a Token Ring NIC cannot be used in Ethernet, vice versa

Start Frame Delimiter [1 octet]
Access Control (PPPTMRRR) [1 octet]
Frame Control Type [1 octet]
Destination Address [6 octets]
Source Address [6 octets]
Frame Check Sequence [4 octets]
End Frame Delimiter [1 octet]
Frame Status (1octet)

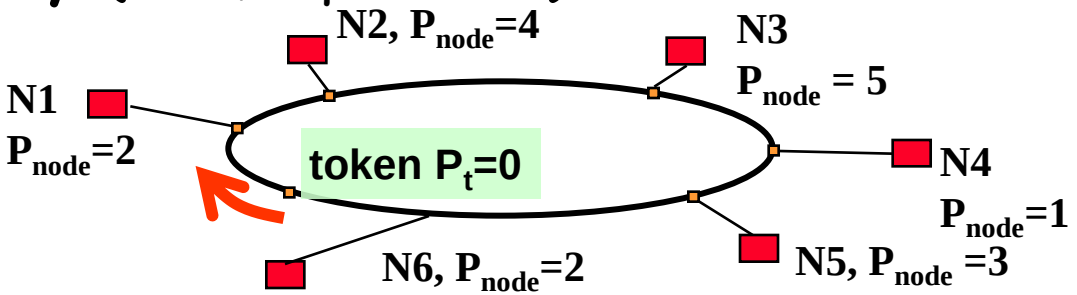
Token Ring Frame (Cont.)



Start Frame Delimiter [1 octet]
Access Control (PPPTMRRR) [1 octet]
Frame Control Type [1 octet]
Destination Address [6 octets]
Source Address [6 octets]
Frame Check Sequence [4 octets]
End Frame Delimiter [1 octet]
Frame Status (1octet)

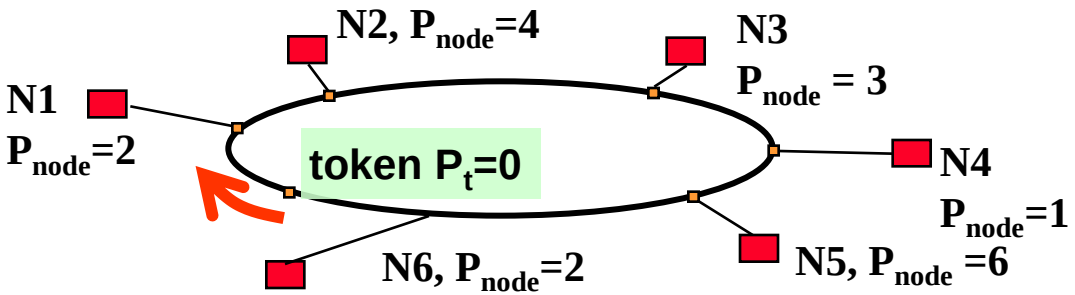
Token Ring – Priority Scheme

- Location affects the chance of getting a free token. In our example N2 would always have a better chance than N6. Priority Scheme addresses this issue. It is an optional scheme, meaning a Token Ring many not need to implement it
- It assigns priority for free token by PPP in the frame. There are 8 levels of priority
- A node can grab a free token only if its priority, P_{node}, is higher than the priority of the free token, P_{token}
- P_{token} is set by the monitor when it is releasing the free token. Since it does not know the priority of nodes, it sets P_{token} to a low priority (i.e., P_t=PPP=0)



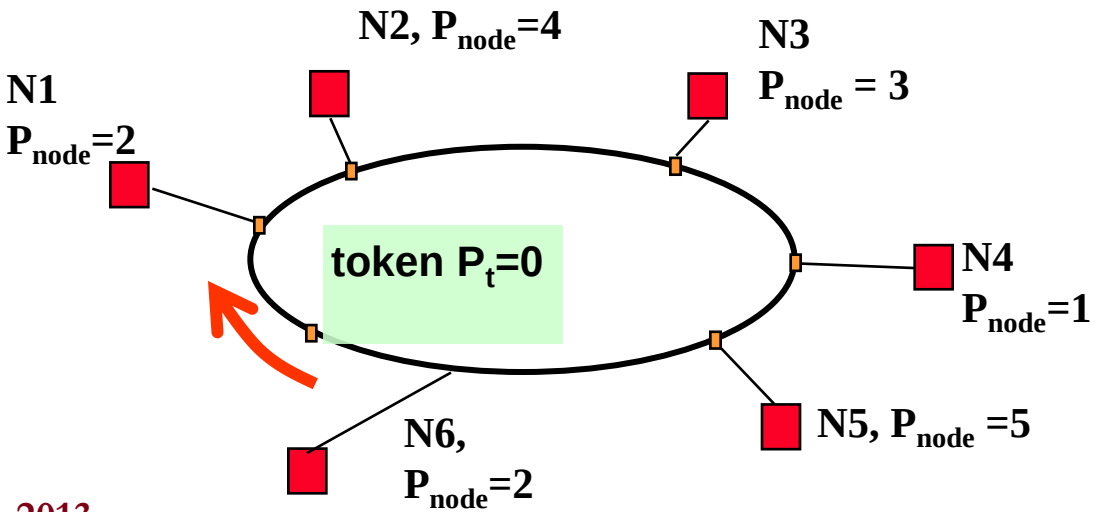
Token Ring – Priority Scheme

- RRR is also originally 000. N1 can grab the token if it wants to transmit, but it does not have a frame to transmit, it relays the token. In our example N2 grabbed the token and transmitted a frame. When the frame gets to N3, if N3 has a frame to transmit, it cannot grab the token, even if P_{node} of N3 is higher than PPP=0, however it can reserve it by setting RRR = 011
- Then finally N2 receives back the frame it just transmitted; it issues a new token, it takes the content of the reservation RRR and moves it to priority PPP
- N3 cannot grab the token, if PPP=110



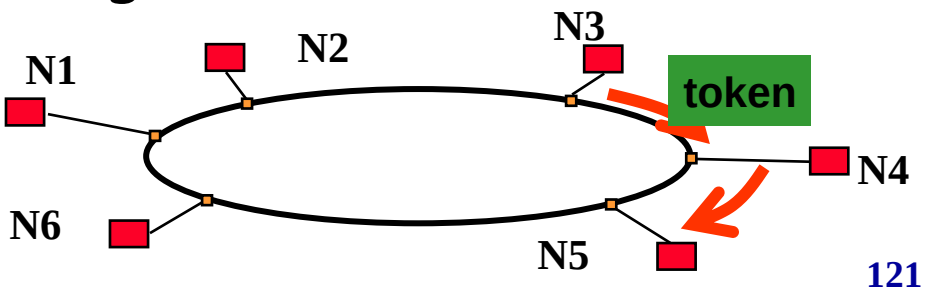
Token Ring – Priority Scheme

- Note that monitor needs to monitor the token that it is circulating and make sure it does not stay at high priority for a long period of time to make it fair for low priority nodes
- Also note that compared to Ethernet, the Token Ring uses a controlled access procedure (and not random access which is used in Ethernet,) and it does not need to have a protocol similar to Ethernet's CSMA/CD



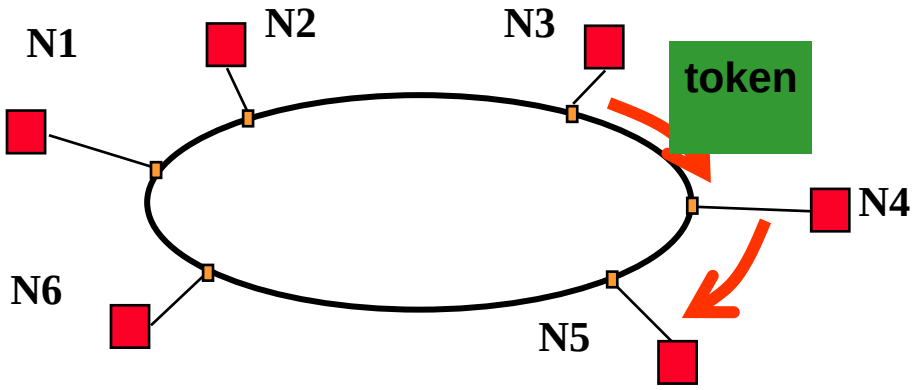
Token Ring – Delay and Throughput

- What is the maximum amount of time a node needs to wait to be able to transmit a frame?
  - In channelization each node knows when it is going to get the channel and transmit
  - In random access (e.g., CSMA/CD) because of the randomness and the number of collisions the wait time may theoretically be infinity as the node may never be able to transmit
  - Token Ring stands in between: based on the THT (Token Holding Time) each node can come up with an upper bound Token as the maximum amount of time it needs to wait to be able to grab the token and transmit



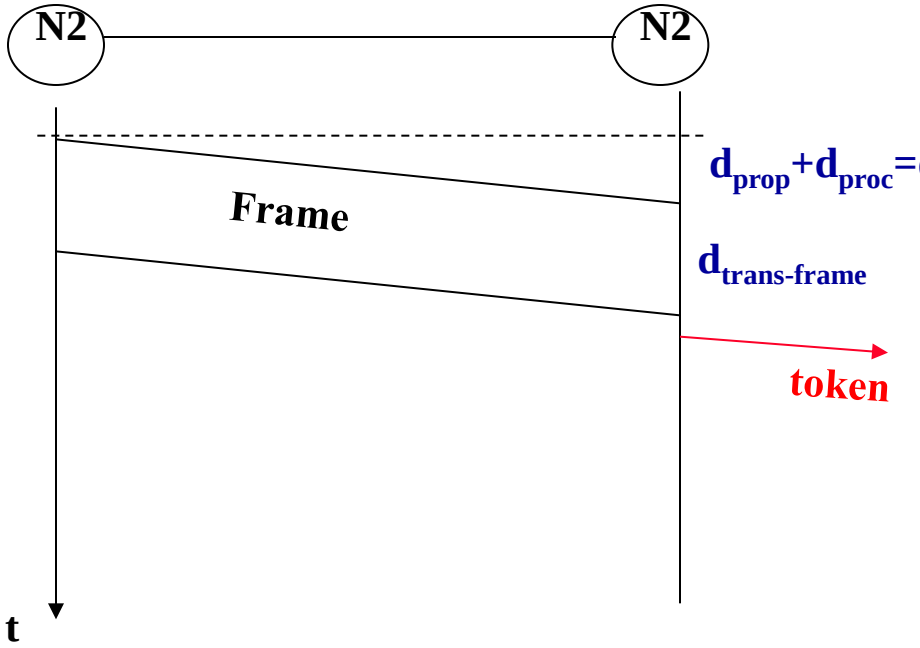
Token Ring – Delay and Throughput (Cont.)

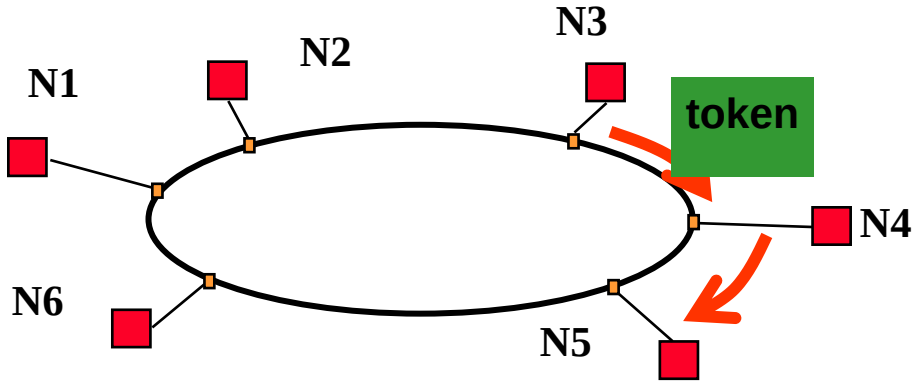
- Example: In the following each node introduces a 3 bit delay in average. Assuming the bit rate, R is 1Mbps what is the processing delay of each node?
  - A 3 bit delay means 3 times as long as the bit duration which is 1/R=1μsec. Therefore each node has a d_{proc} = 3μsec
- Question: The propagation delay of each segment (link) in the following Token Ring is 1μsec. What is the total propagation delay? d_{prop} = 6× 1μsec = 6 μsec



Token Ring – Delay and Throughput (Cont.)

- Model I: delayed release of token is used
- For throughput: consider the delay as the time a node transmits a frame to the time another node transmits a new frame. The best case is that next node transmits the new one





Best case

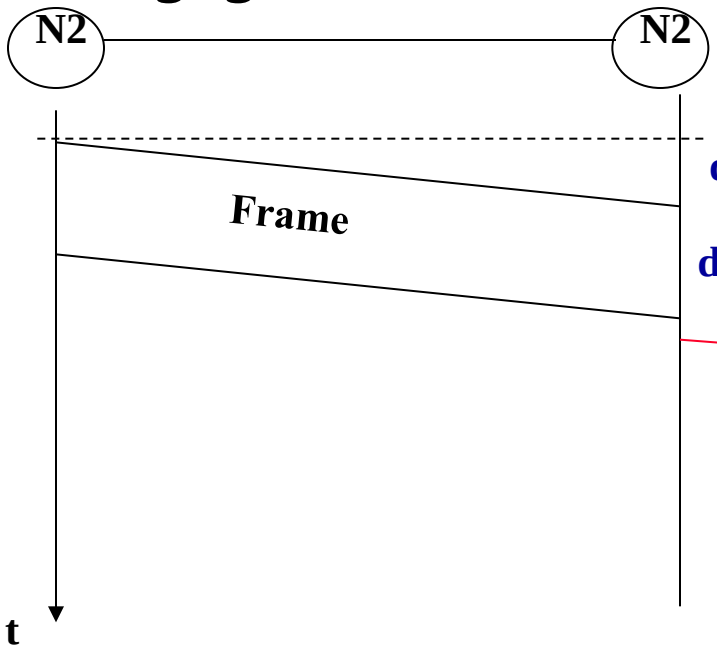
Throughput = Frame length/$\{d_R + d_{trans-frame} + d_{trans-token} + d_R/N\}$

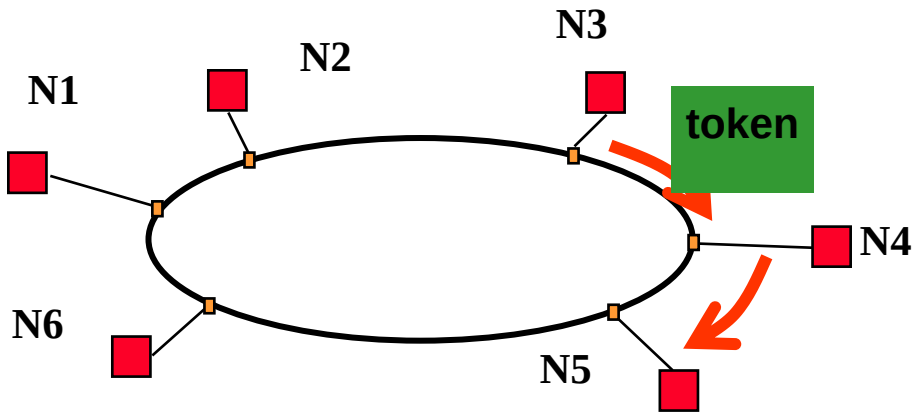
Token Ring – Delay and Throughput (Cont.)

- Example: What is the throughput if the next 3 node gets the free token and transmits a new frame?

Throughput =
Frame length/{d_R + $d_{trans\text{-}frame}$ + $d_{trans\text{-}token}$+ $3d_R/N$}

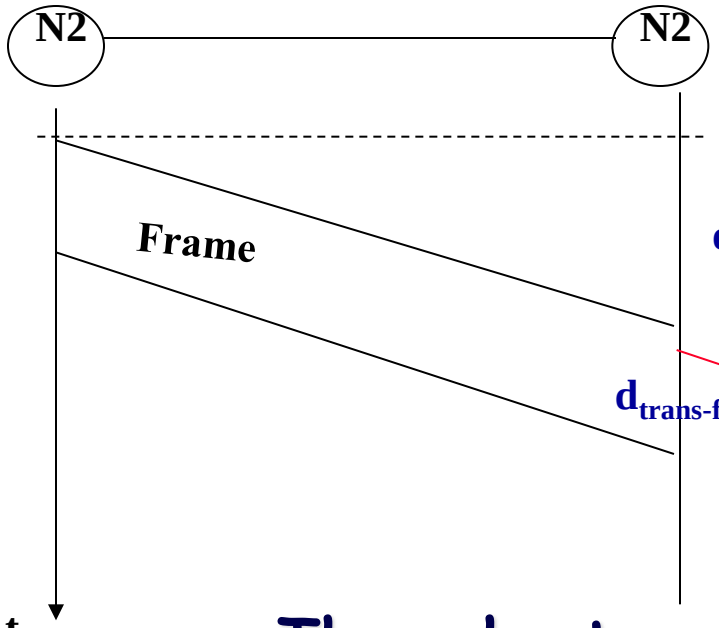
- Note: $d_{trans\text{-}token}$ is negligible. Also d_R/N can be negligible because N is typically large

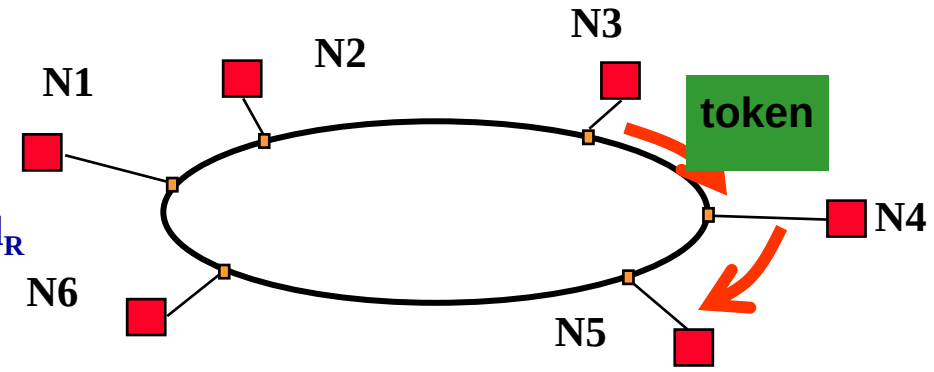




Token Ring – Delay and Throughput (Cont.)

- Model II: delayed token release but faster than Model I
- Note that a node may not have finished transmitting its frame, when it receives the first bit of that frame back from the ring and in this case this delay model won't be applicable, because the token is not free yet, so the node cannot issue a free token yet

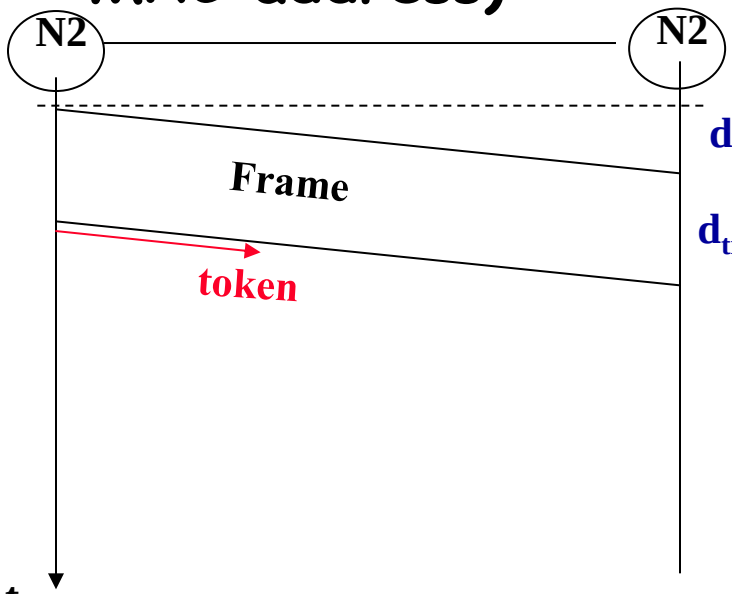


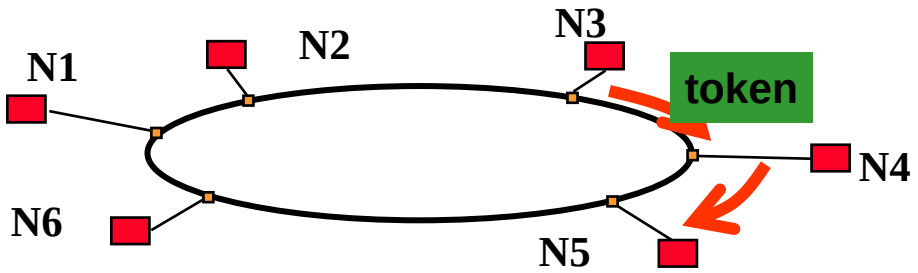


Throughput = Frame length/{d_R+ $d_{trans-token}$+d_R/N}

Token Ring – Delay and Throughput (Cont.)

- Model III: Early token release
- N2 releases the token early, therefore N3 can grab the new free token and use it and still no collision would happen because one frame following another
- Also there will be only one free token. N2 will receive the 1st frame and remove it (would recognize it from source MAC address)



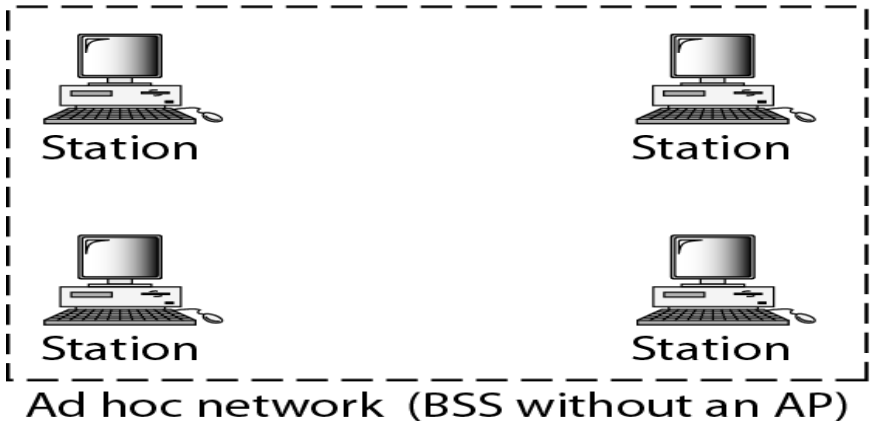


Throughput = Frame length/{$d_{trans-frame}$ + $d_{trans-token}$ +d_R/N} ≈ R

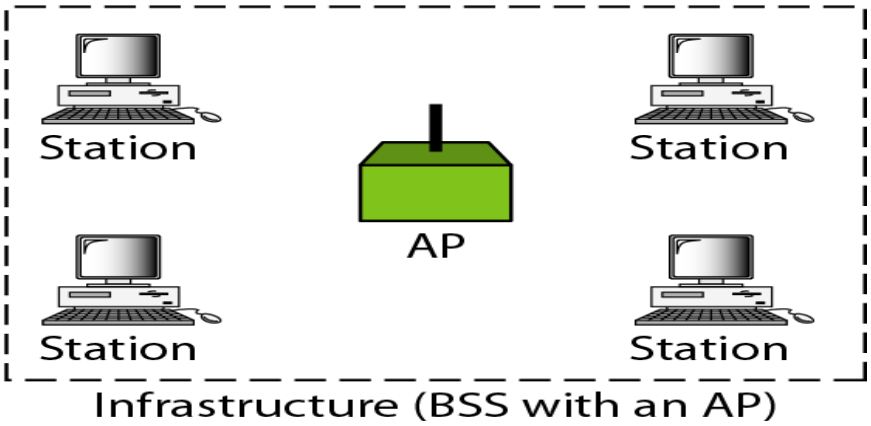
Wireless LANs

- A BSS without an AP is called an **ad hoc** network. Bluetooth (802.15) is an example. There is no structure and no wiring in ad hoc and access to Internet is not possible
- a BSS with an AP is called an **infrastructure** network also known as **WiFi** (802.11 a/b/g/n). In infrastructure all communications go through AP which is connected to a backbone infrastructure. The backbone can be cable modem connected to Ethernet or a DSL modem. Access to Internet is possible through the backbone infrastructure

BSS: Basic service set
AP: Access point



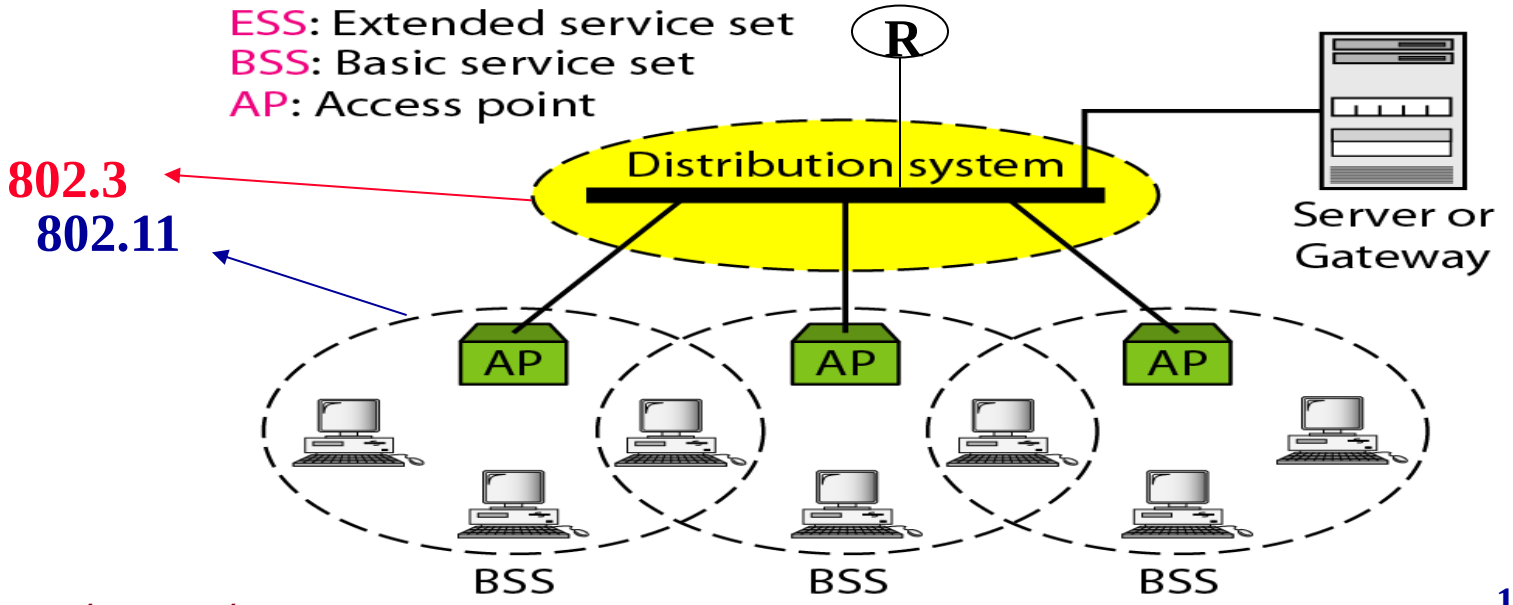
Ad hoc network (BSS without an AP)



Infrastructure (BSS with an AP)

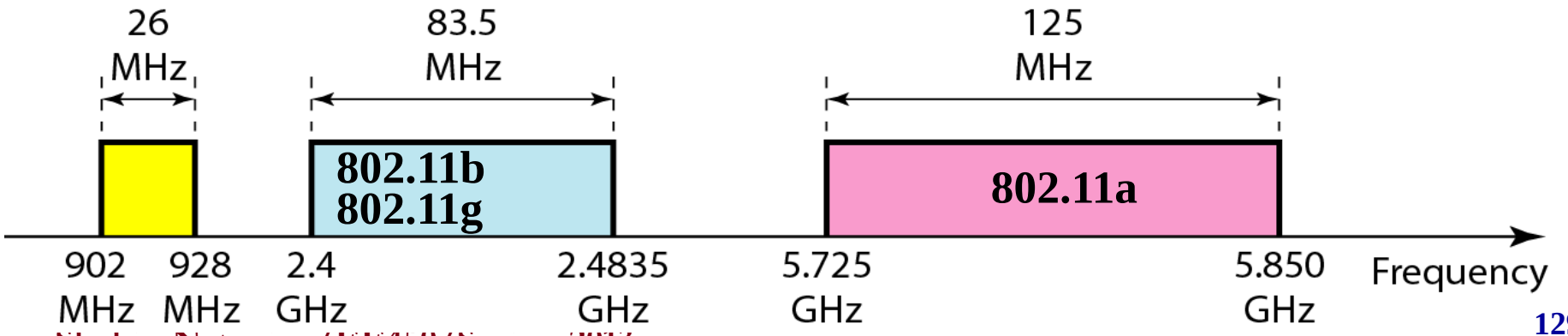
Extended Service Set (ESS)

- NICs in the PCs are of wireless (802.11) type, whereas the AP has Ethernet NIC as well in here
- The term **wireless** is more reasonable than **mobile** for 802.11 as it is used inside a building and mobility is hence very limited
- A **BSS** in 802.11 resembles a **cell** in a cellular network. As the (caller) user moves from one cell to another cell, system needs to hand over the connections from one AP to another AP (one base station to another base station)



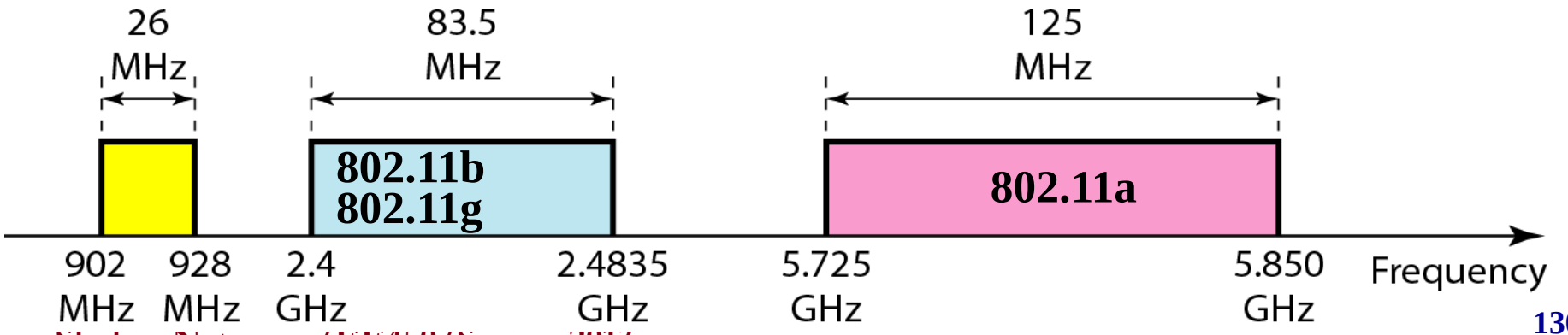
Industrial, Scientific, and Medical (ISM) bands

- Radio (RF) signal is used to carry information from the node to AP
- RF is analog and the node generates digital data, therefore a modem is required to do D/A and D/A conversion
- An antenna is also needed to convert the analog electric signal to electromagnetic radiation
- FCC (Federal Communications Commission) assigns bands for wireless/cordless operations



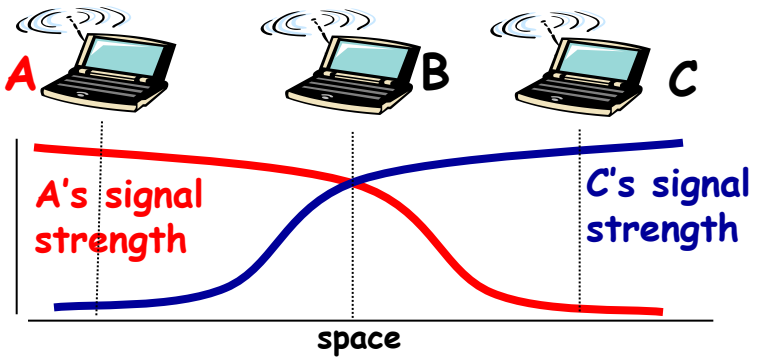
Industrial, Scientific, and Medical (ISM) bands

- Wireless and cordless bands are unlicensed
- The bands (the medium, air) are shared, therefore protocols are required to avoid interference
- The main difference between 802.11a, b, and g is related to the modulation techniques



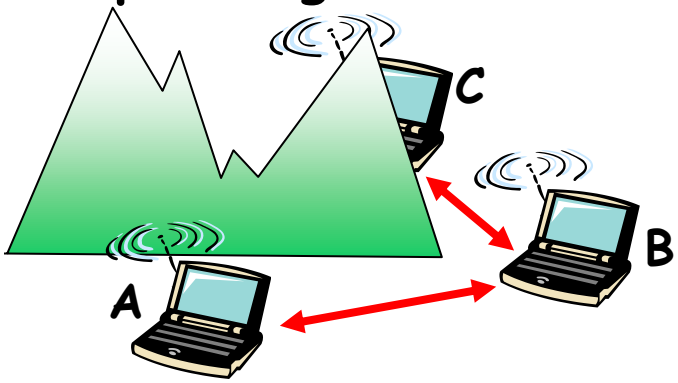
Wireless Link Characteristics – Differences with Wired Link

- **Decreased signal strength**: radio signal attenuates as it propagates through matter (path loss)
  - When signal is propagated using RF (Radio Frequency) in the air, signal strength will decrease as signal source moves away from AP (or base station.) This is referred to as **signal fading**
- **Interference from other sources**: standardized wireless network frequencies (e.g., 2.4 GHz) shared by other devices (e.g., phone); devices (motors) interfere as well
  - Open air is the shared resource (medium) and wireless bands are unlicensed, so it can be used by any device that operates on those bands, this means signals will interfere with each other. Even other devices, such as elevators, microwaves, etc. can result in **interference**



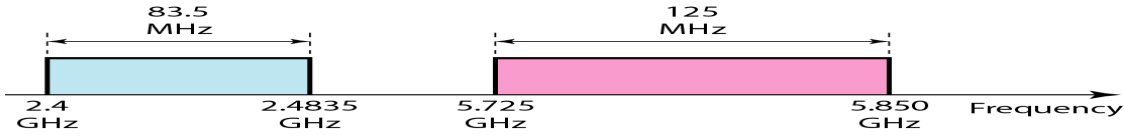
Wireless Link Characteristics – Differences with Wired Link (Cont.)

- **Multipath propagation**: radio signal reflects off objects and ground, arriving at destination at slightly different times
  - Signals may travel in **multiple paths** to the receiver, reflecting from ground, buildings, etc. Since propagation delay of the signals may be different, they may become destructive and can weaken/cancel each other. In case of IEEE802.11 the signal may travel indoors reflecting from walls, ceiling, etc. and result in multi-path signals
- **Hidden Terminal**:
  - B, A hear each other
  - B, C hear each other
  - A, C can not hear each other
  - This means A and C unaware of their interference at B



…. make communication across (even a point to point) wireless link much more "difficult"

802.11 – Channel Association

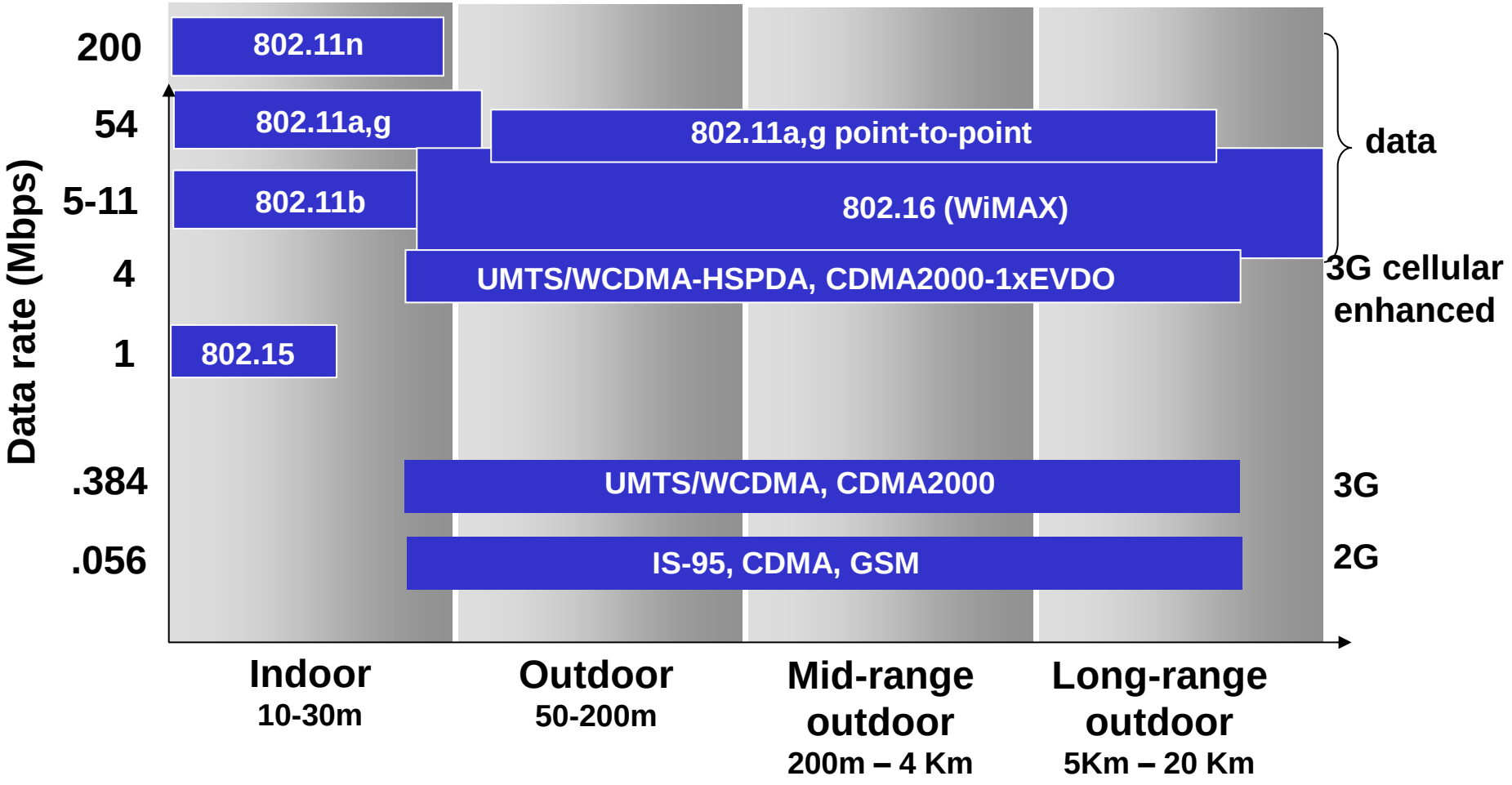


- 802.11b: 2.4GHz-2.485GHz (i.e., about 85MHz of bandwidth) spectrum divided into 11 channels at different frequencies
  - Each channel is about 10MHz so there is overlap btn channels
  - Every BSS needs only one frequency and network administrator chooses frequency for the AP
  - Interference possible: channel can be same as that chosen by neighboring AP
- Each host must ***associate*** with an AP
  - Scans channels, listening for ***bacon frames*** ☺ containing AP's name (**SSID** or **Service Set Identification**) and MAC address
  - Selects AP to associate with
  - May perform **authentication**
  - Will typically run DHCP to get IP address in AP's subnet

802.11 – Channel Association (Cont.)

- Question: why is coordination not allowed in 802.11 networks?
  - Because then every time we want to setup a WLAN, we need to check with all our neighbors and see whether they have WLAN
- Instead of coordination, every network administrator chooses one of the channels. It is possible that two neighboring APs use the same channel and then interference will happen
- The farther the APs are from each other, the lower is the interference
- If two neighboring channels choose different channels, then the interference is lower than when they choose the same channel
- A wireless NIC may be close to several APs, then it needs to make a decision which AP it wants to associate with
- How does the NIC know where the APs are?
  - To know the APs and where they are and choose from, the NIC first needs to know the APs' frequencies

(Optional) Characteristics of Selected Wireless Link Standards



(Optional) IEEE 802.11n

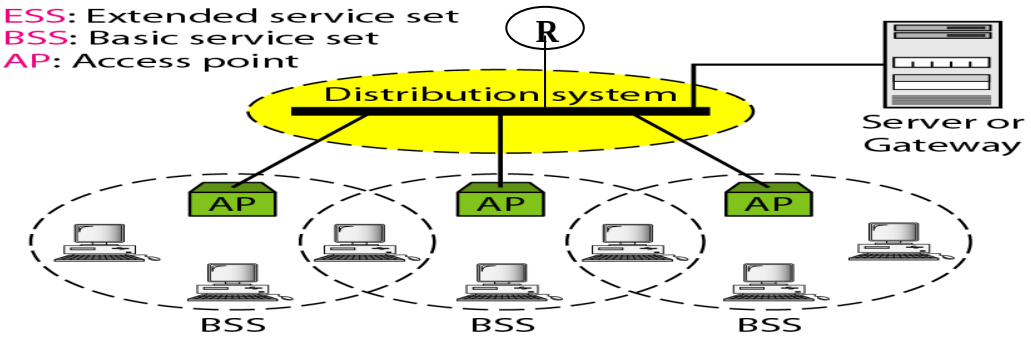
- 802.n is a recent amendment to IEEE standard 802.11 (a and g)
- It improves 802.11 by adding **multiple-input multiple-output** (**MIMO**) antennas, i.e., two or more antennas on the sender side and two or more antennas on the receiver side
- The main improvement is the significant increase in the maximum raw data rate, e.g., from 54Mbit/s to 600Mbit/s with the use of four spatial streams at a channel width of 40MHz
- The IEEE has approved the amendment and it was published in October 2009

802.11 – Bacon Frame :)

- Each AP broadcasts it periodically at a frequency chosen by the admin
- The beacon frame informs the wireless NICs that there is an AP, it also informs them about AP's MAC address, its SSID which is like a serial number for the AP
- The modem of the NIC (of the wireless station) scans all 11 channels and picks the one that is strongest (i.e., the closest AP)
- It receives the beacon frame signal from that AP, demodulate it and gets the AP's MAC address and SSID of the AP
- Using the same frequency of the chosen AP (the frequency of the beacon frame) the wireless station then sends a unicast signal to the AP to inform the AP, that it is a wireless station that wants to associate with that AP, and also gives its MAC address
- AP then allows connection if authentication is not required, but otherwise it goes through an authentication process before allowing the wireless station to associate with it

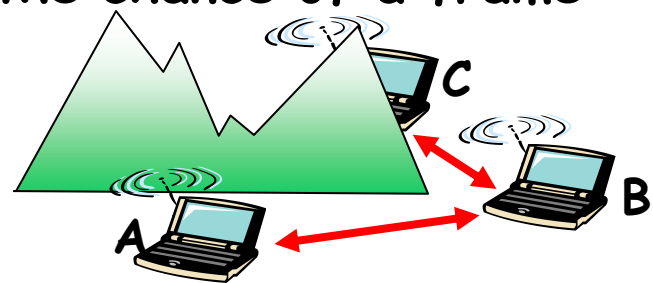
802.11 – Handoff

- Once the wireless station (its NIC) associates with the AP it needs to have an IP address (to put in the header of its packets) to connect to the network
- It gets an IP address from the DHCP server (e.g., connected to the backbone distribution system)
- While using the 802.11 service through an AP, if the wireless station moves away from an AP and gets closer to another AP it receives a stronger beacon signal from the closer AP. However if the closer AP uses a different frequency, wireless station needs to handoff, i.e., switch its frequency from the original AP to the new one (similarly to what a cell phone does while moving away from one BS and getting closer to another. During handoff disconnection may occur because switching may not be a fast process)



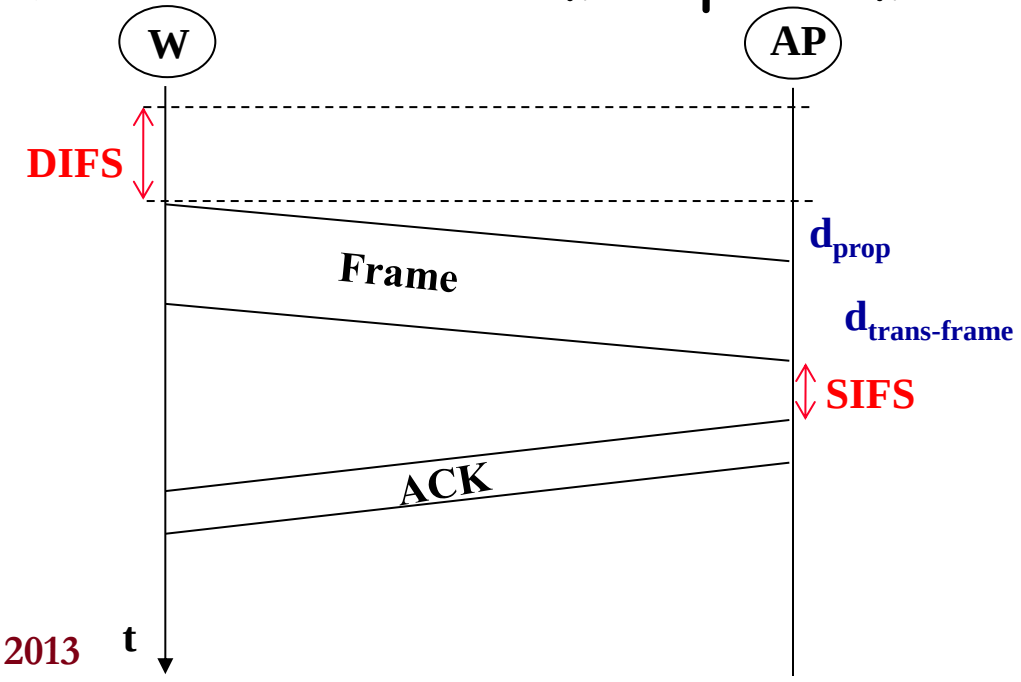
802.11 MAC Procedure is CSMA/CA

- Ethernet uses CSMA/CD, however collision detection does not work for 802.11 because if a collision happens close to B, C and A cannot hear it: the collision propagates back towards A and C, but it will fade
- We need a protocol to try its best to avoid the collision
- 802.11 uses **CSMA/CA**. A different protocol means the wireless card (802.11 NIC) is not compatible with the Ethernet card (802.3) [also not compatible with Token Ring's, i.e., 802.5 NIC]
- In wireless, data is bursty, meaning that data is transmitted for a short period of time and then for a long time there is no transmission. Also BSS has a few number of work stations, it won't be a big network.
- Still there is a chance a frame won't ever be transmitted. The randomness of CSMA/CA tries to increase the chance of a frame going through



802.11 MAC Protocol – CSMA/CA

- MAC procedure is started after the station associates with AP
- CSMA/CA is used. CSMA implies that sensing (using an electronic sensor) is performed before using the medium, however CA means the procedure tries its best to avoid collision
- Case I: Idle medium: the wireless NIC has a frame to transmit. It senses medium as idle
- Important: Even though the medium is sensed as idle, the channel may in fact be busy due to the hidden terminal problem



CSMA/CA – Idle Medium (Cont.)

- W waits a certain period of time, DIFS (DCF InterFrame Space) after it senses an idle channel [DCF≡Distributed Coordination Function]
- This is different from CSMA/CD 1-persistent that sender transmits if it senses idle channel
- During DIFS, W keeps sensing the channel and if the channel remains idle, it transmits right after DIFS
- Due to hidden terminal problem channel may indeed be busy and a collision will then occur (the question is how do we detect it?!)

Answer: as mentioned earlier, CSMA/CA does not detect collision!

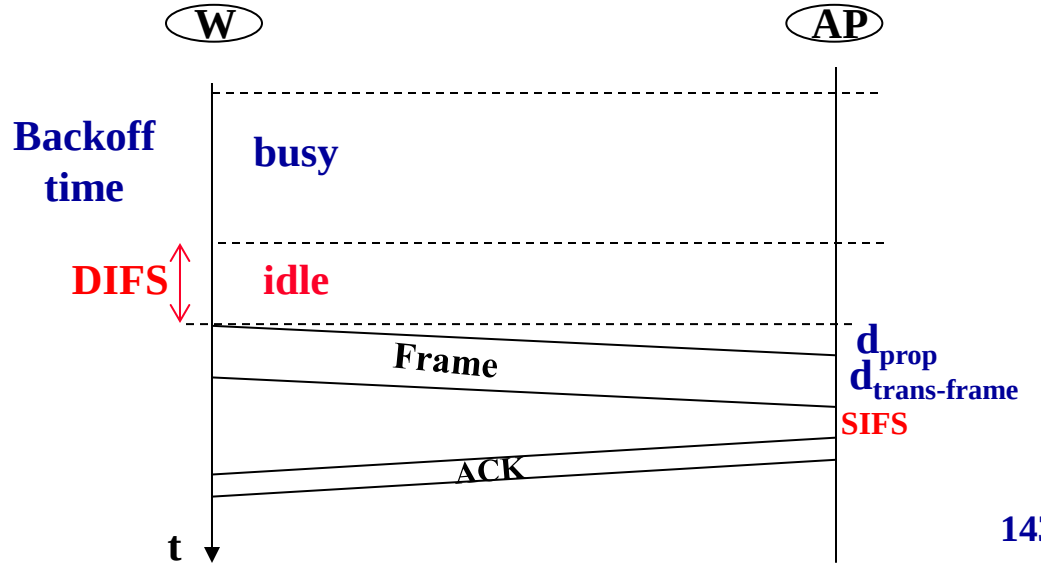
- Note that AP can detect a collision, because AP can see all the wireless stations
- How does the station know that its transmitted frame was received with no collision/no error detected? Acknowledgement is used in CSMA/CA. Frame sequencing is also used (similarly to point-to-point protocols)
- Note that 802.3 has neither acknowledgement nor frame sequencing

CSMA/CA – Idle Medium (Cont.)

- AP waits for a short period of time, **SIFS** (**Short InterFrame Space**) after it receives the frame
- During SIFS, AP keeps sensing the medium to check whether some other station is transmitting and if not, it sends the ACK
- SIFS is shofter than DIFS, because ACK should have higher priority; if AP waits longer, it is more likely that someone else transmits and then ACK gets delayed, and this in turn will result in W's timeout and W has to retransmit the frame needlessly (this would be waste of resources; AP has received the frame with no error already. Therefore it is reasonable to consider SIFS short to let AP be ahead of wireless stations)
- Questions: A BSS is found to have a high chance of not having hidden terminal. Does it make sense to implement CSMA/CD instead of CSMA/CA?
  - No, what if things change and we see hidden terminal problem frequently!

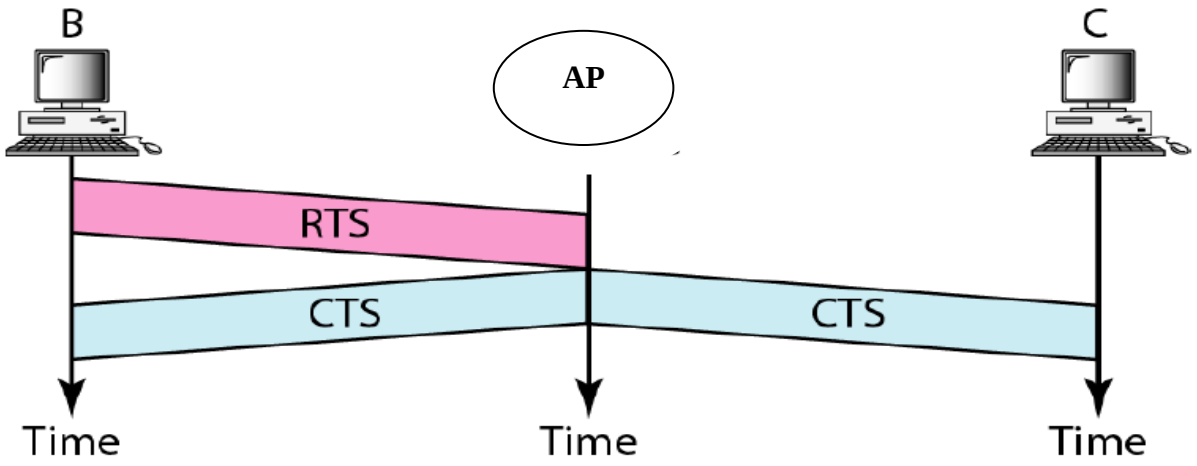
CSMA/CA – Busy Medium

- Case II: Busy medium: W has a frame to transmit. It senses the medium and finds it busy, so someone else (either AP or another station) must be transmitting. W refrains from transmitting for a random period of time (similar the 802.3 in case of collision) and keeps its counter frozen
- As soon as W senses the channel to be idle, it (does not transmit yet,) decrements the counter. If the channel again becomes busy, it freezes the counter. Counter is decremented only during idle channel. As soon as counter becomes 0, W transmits its frame



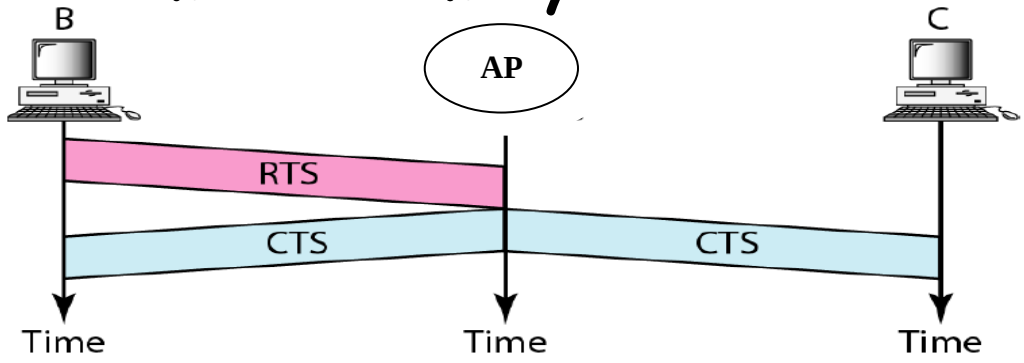
CSMA/CA – RTS-CTS Mechanism

- CSMA/CA is a very cautious protocol. A lot of wait time is considered to avoid collision
- If a collision happens AP sends a NAK, then wireless station waits a longer backoff time
- Cases I and II (idle and busy channel) are required to be implemented. However there is a mechanism in CSMA/CA to address hidden terminal problem, which is not necessitated by the protocol to be implemented. It is called **RTS-CTS**



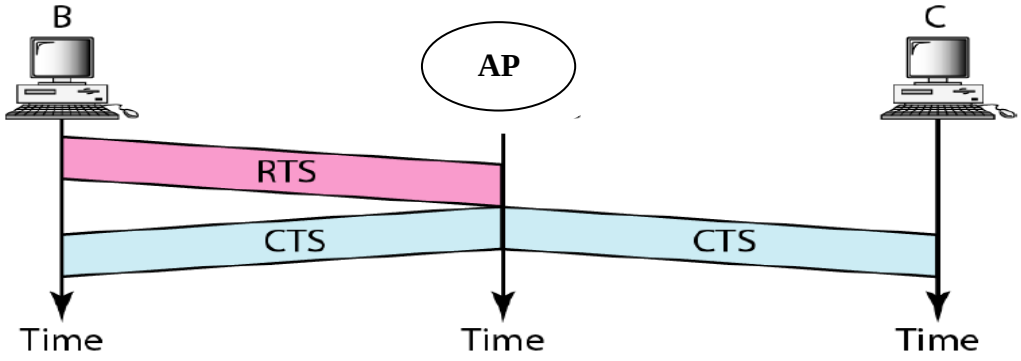
802.11 RTS-CTS Mechanism

- Remember AP sees all the stations. B and C are hidden from each other. B wants to transmit a frame, so it sends a short control frame, called **RTS** asking AP to send a frame and also asking for a certain amount of time **duration** or **reservation** denoted by T_B
- AP then broadcasts a **CTS** frame, telling all stations that B will transmit for the next T_B and telling them to go to hibernate (sleep.) After receiving CTS, B will send its frame and no collision will occur because every other station is asleep. This means CSMA/CA avoids collision using this mechanism by reservation



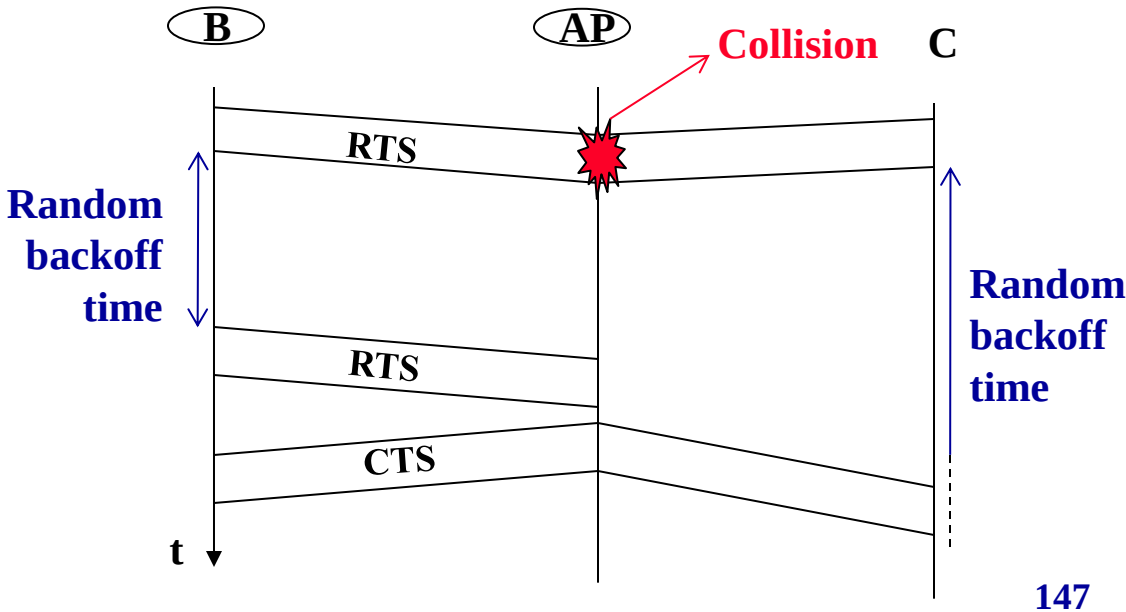
RTS-CTS Mechanism (Cont.)

- If C wants to transmit, if will defer that to after it wakes up
- Note: Hibernation is considered in this mechanism to let stations conserve their power (esp their transceiver) by setting a timer to wake them up after T_B
- Power consumption is a major issue in wireless and mobile systems, e.g., in cellular network. Cell phones are designed to go to minimal power consumption just enough to be able to detect BS transmission to them



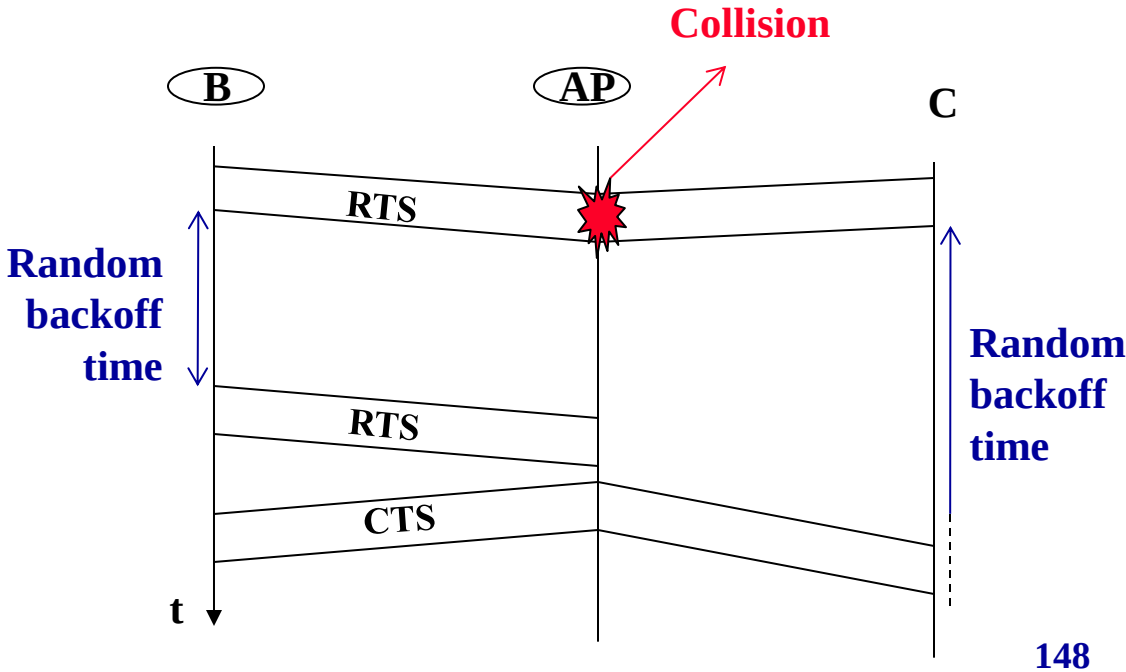
RTS-CTS Mechanism (Cont.)

- Collision can still happen during RTS, which means a short control frame is lost, but it is good is that no data is lost
- After T_B passes, stations will wake up and then the whole story will go on based on the rules described
- Note: In its RTS, B will specify the amount of time it needs to transmit (denoted here by T_B)
- B, should include the time it needs to transmit RTS itself, the time AP needs to process RTS and the time B needs to receive ACK (CTS)



RTS-CTS Mechanism (Cont.)

- Question: How does collision occur during RTS?
  - Assume B and C want to transmit and they send RTS both at the same time. A collision will take place and B or C won't hear from AP, so they find out a collsion must have occurred. They wait a random back off time



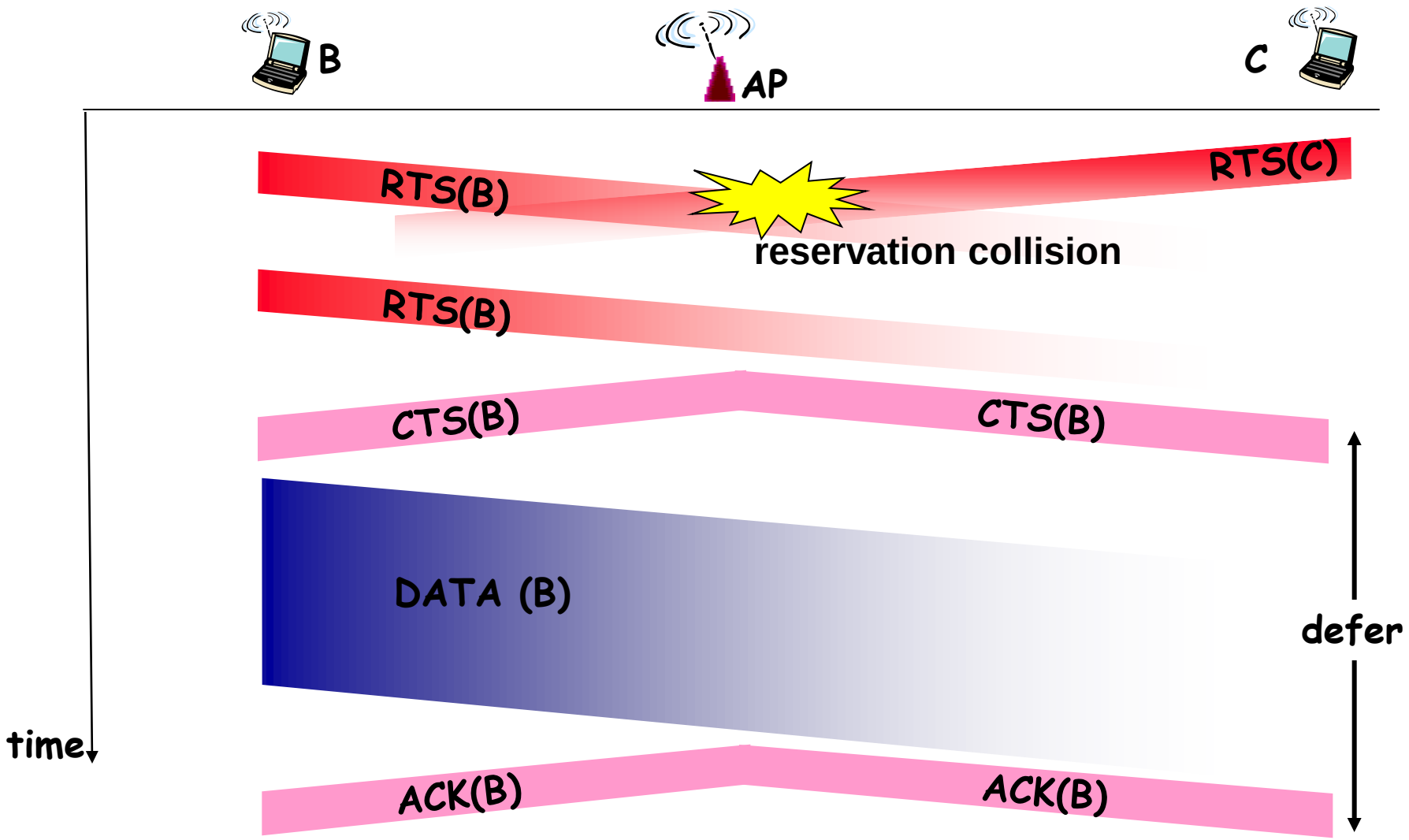
CTS-RTS Summary

Idea: Allow sender to "reserve" channel rather than random access of data frames: avoid collisions of long data frames

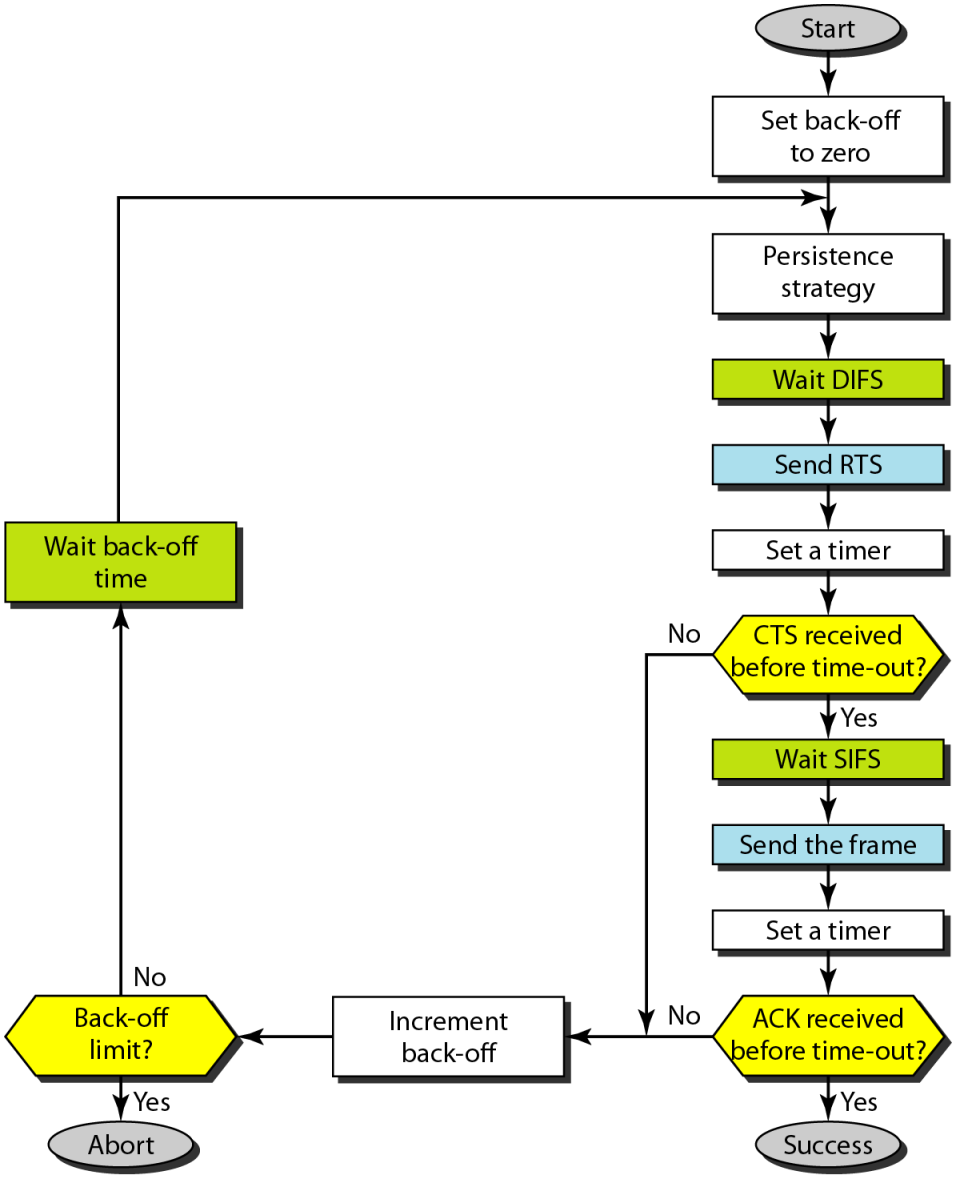
- Sender first transmits *small* request-to-send (RTS) packets to BS using CSMA
  - RTSs may still collide with each other (but they're short)
- BS broadcasts clear-to-send CTS in response to RTS
- CTS is heard by all nodes
  - Sender transmits data frame
  - Other stations defer transmissions

Avoid data frame collisions completely using small reservation packets!

Collision Avoidance: RTS-CTS Exchange

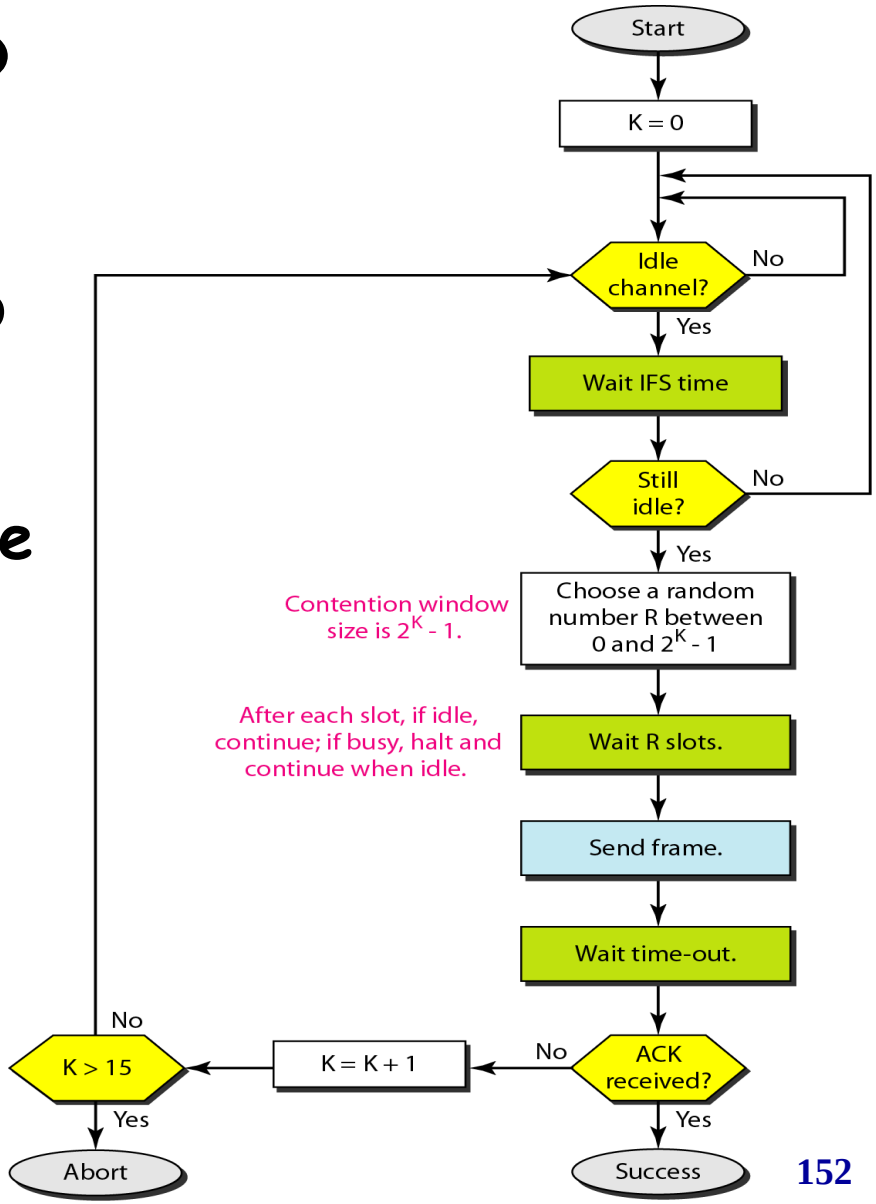


CSMA/CA – RTS/CTS Flowchart

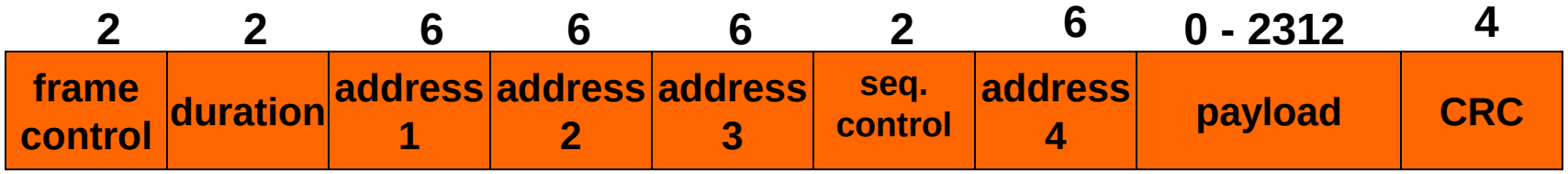


CSMA/CA Flowchart

- Compare this with CSMA/CD flowchart [for Ethernet] in the current slide unit
- K is incremented similarly to what we saw for the case CSMA/CD to increase the number of trials and also the backoff time (R slot wait time)



802.11 Frame



Address 1: MAC address of wireless host or AP to receive this frame

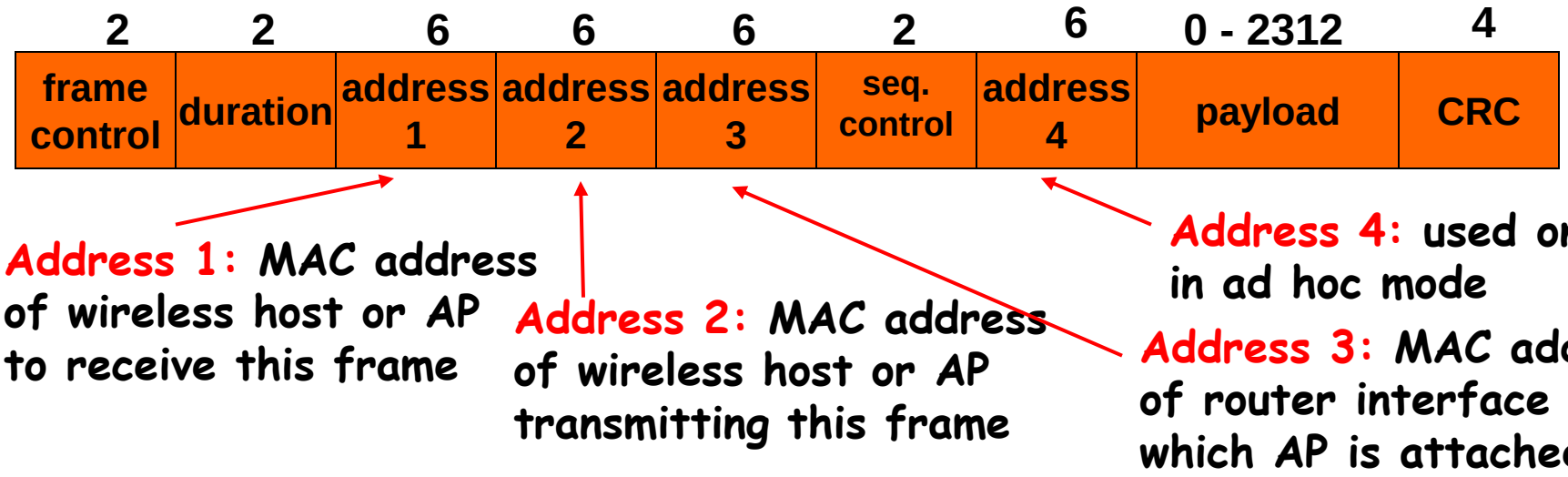
Address 2: MAC address of wireless host or AP transmitting this frame

Address 3: MAC address of router interface to which AP is attached

Address 4: used only in ad hoc mode

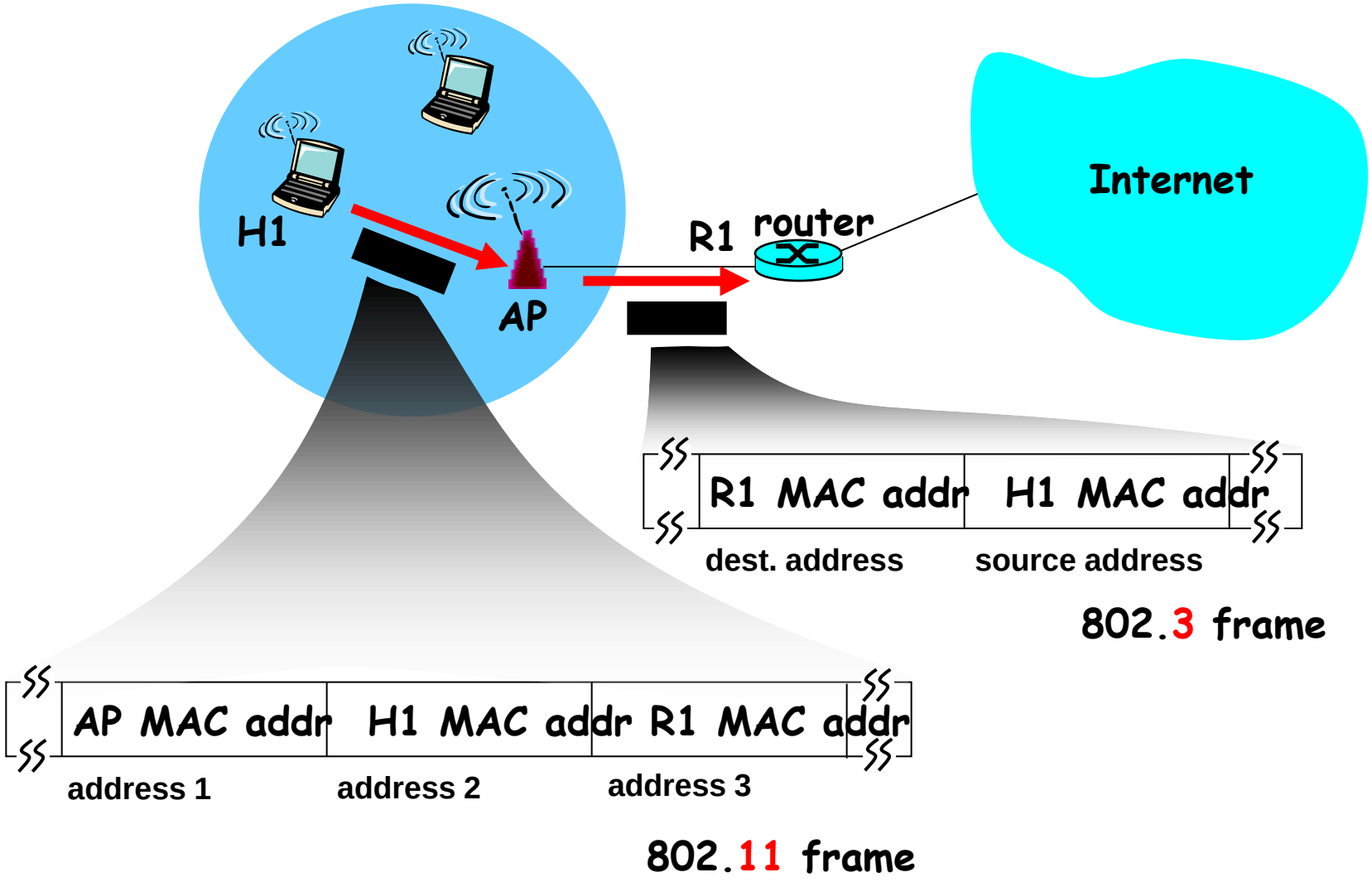
- Payload is for the network layer (it is the packet from layer 3)
- CRC (or FCS) is for the same error checking algorithms we saw before
- Sequencing is required for CSMA/CA. therefore a sequence control is added to acknowledge a particular frame

802.11 Frame (Cont.)



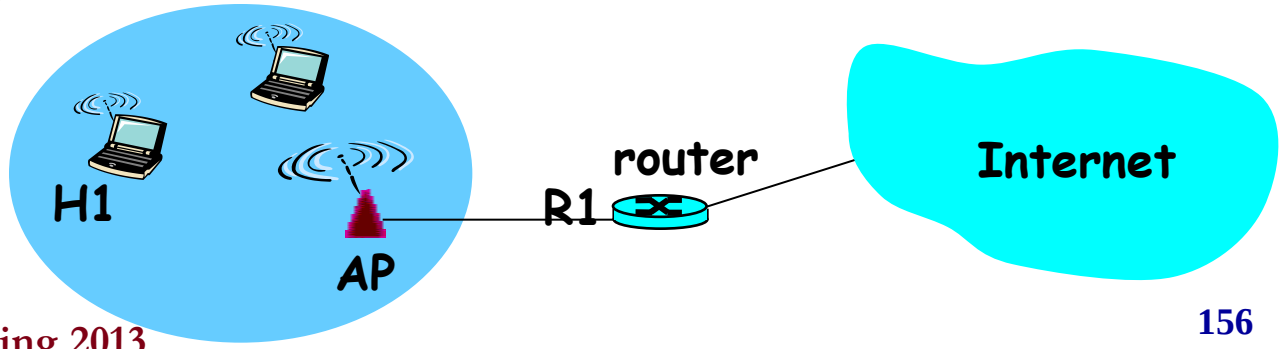
- 802.11 is different from PPP, Ethernet and Token Ring frames, therefore their corresponding NICs are not compatible
- Duration is the reservation amount
- Frame control has some other functionalities in the frame (not our concern)

802.11 Frame – Addressing



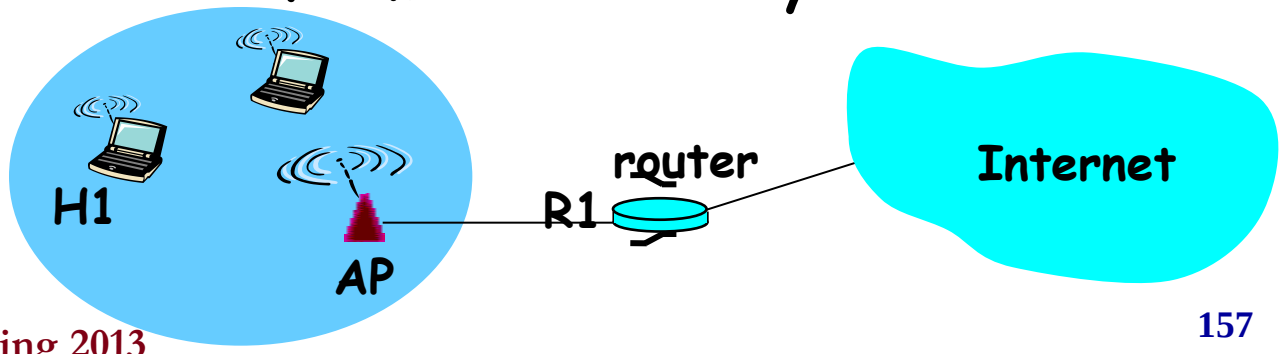
802.11 – Addressing (Cont.)

- All host communications go through the AP. If H1 wants to communicate with another station in its BSS, it has address1: destination host's MAC address, address2: H1's MAC address, however H1 does not put the MAC of the router in address3
- If H1, wants to communicate with a node on the Internet it needs the help of the router, R1 and uses address3. Host communicates with the AP not R1. R1 does not have an 802.11 NIC and does not understand wireless
- AP recognizes that address3 has the MAC address of R1, so AP knows that the ultimate destination of that frame is R1, not AP
- AP needs to change the 802.11 format to a frame R1 understands



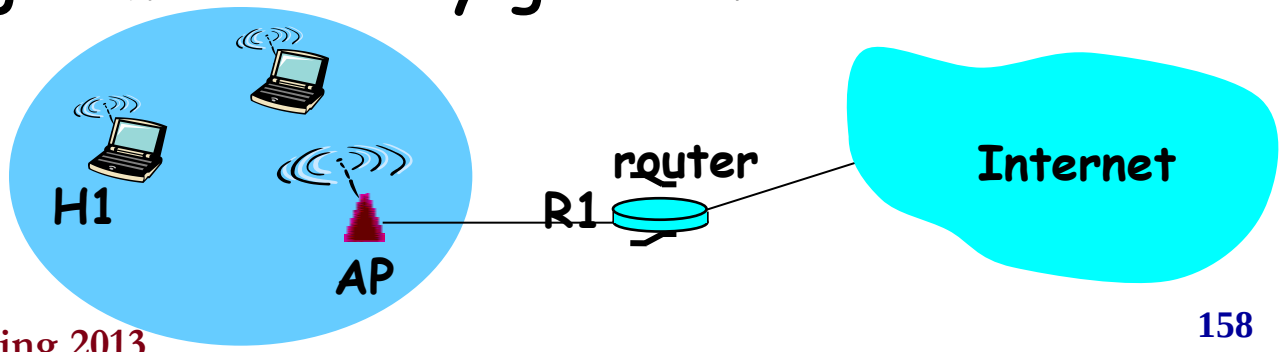
802.11 – Addressing (Cont.)

- Assume the LAN connecting R1 to AP is Ethernet, so R1 understands 802.3 MAC frame. Address 3 allows AP to determine the appropriate destination MAC address. AP creates an 802.3 MAC frame, by using H1 MAC address as the source MAC address. This makes R1 think that H1 sent the frame even though AP is the sender of the 802.3 frame. We say AP is **transparent** to router, R1. This way R1 thinks it is directly connected to the nodes H1, H2, etc.
- The only nodes that know about AP are the wireless stations (hosts.) Router does not know about AP existence
- Therefore in the 802.3 network (including R1) there is nothing indicating that the frame was sent by the AP



802.11 – Addressing (Cont.)

- Question: How does H1 know the MAC address of R1? Through ARP, but in order to use ARP, it needs the IP address of router (that it obtains either manually, or through DHCP)
- R1 will forward the frame properly to the Internet till it gets to the server. Server responds and the reply frame is received by R1
- Router creates an 802.3 frame with H1's MAC address as the destination address and R1's MAC address as the source MAC address and sends it to the shared network believing it will directly get to H1



802.11 – Addressing (Cont.)

- Note again that R1 thinks it is directly connected to all BSS hosts, H1, H2, etc.
- AP intercepts, when it figures that the destination MAC address is H1's. It will create an 802.11 frame which has address 1 as H1's MAC address, address 2 as AP's MAC address and address 3 as R1's MAC address. AP gets R1's MAC address from the 802.3 frame