

v 2.1.3 5/April/2017

© 2012, Morgan-Kaufmann Pub. Co., Prof. Larry Peterson and Bruce Davie


Some texts and figures: © 2016 José María Foces Morán

# CH. 2

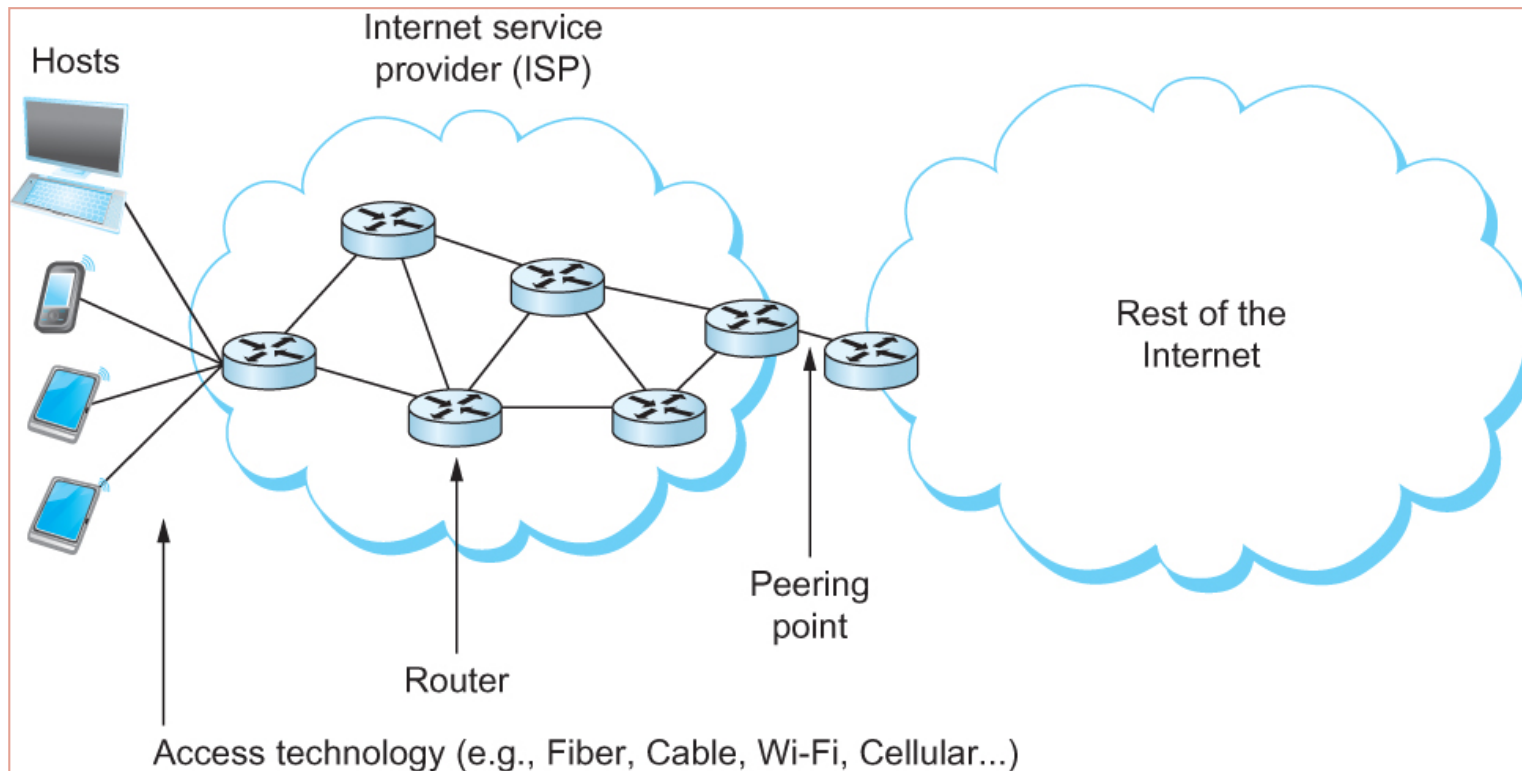
## GETTING CONNECTED

Lecture on how to connect nodes together and to the internet  
Computer Networks, Universidad de León, 2016

# Chapter Outline

- 
- Communication theory basics
  - Transmission media and communication links
  - Encoding: bits to signals
  - Framing: start ... stop
  - Error Detection
  - Reliable point-to-point communication
  - Multiple Access, CSMA Ethernet
  - Wireless Networks

# Connecting a host to a network

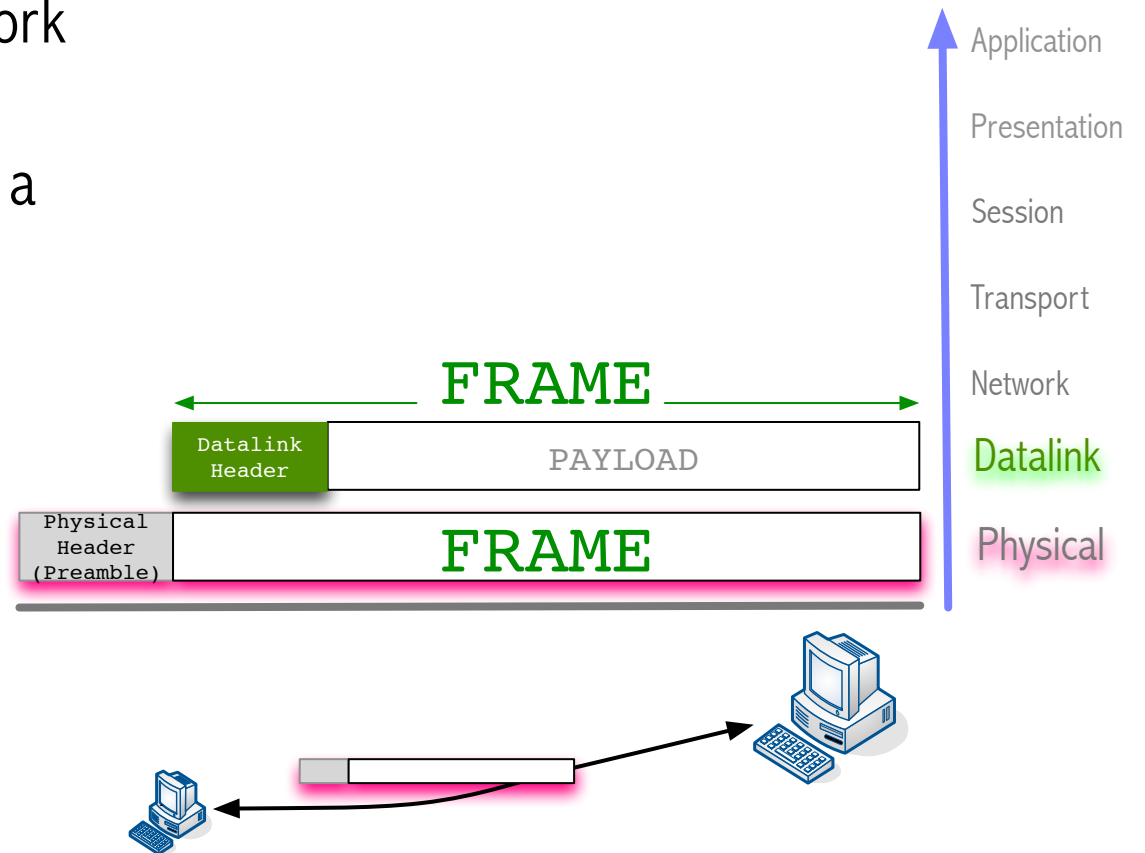


An end-user's view of the Internet

# Focus of this chapter

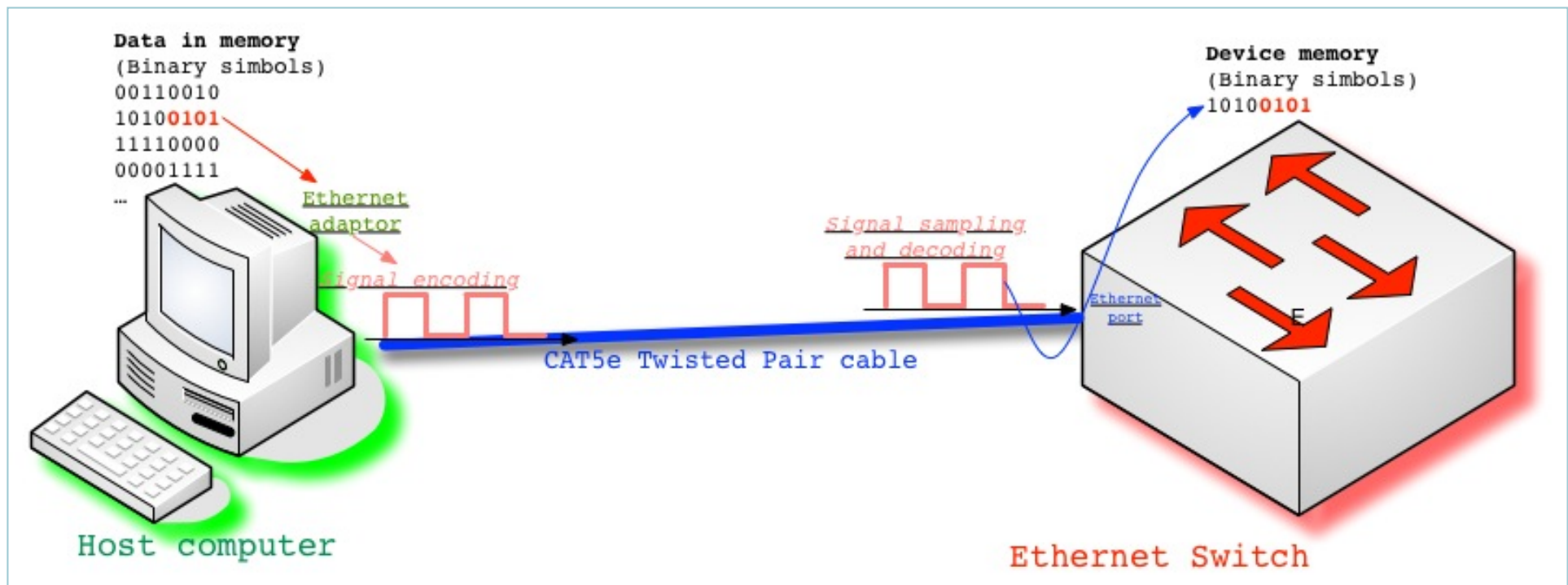
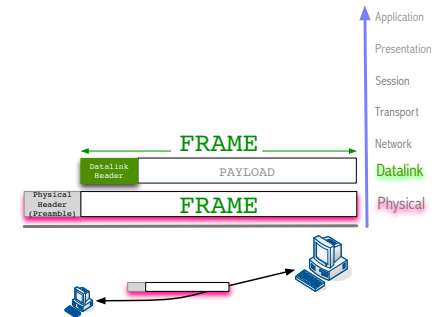
- Addressed problems:
  - ▣ How to connect two network nodes together?
  - ▣ How to connect a host to a network?

- The focus is on the OSI layers 1 and 2



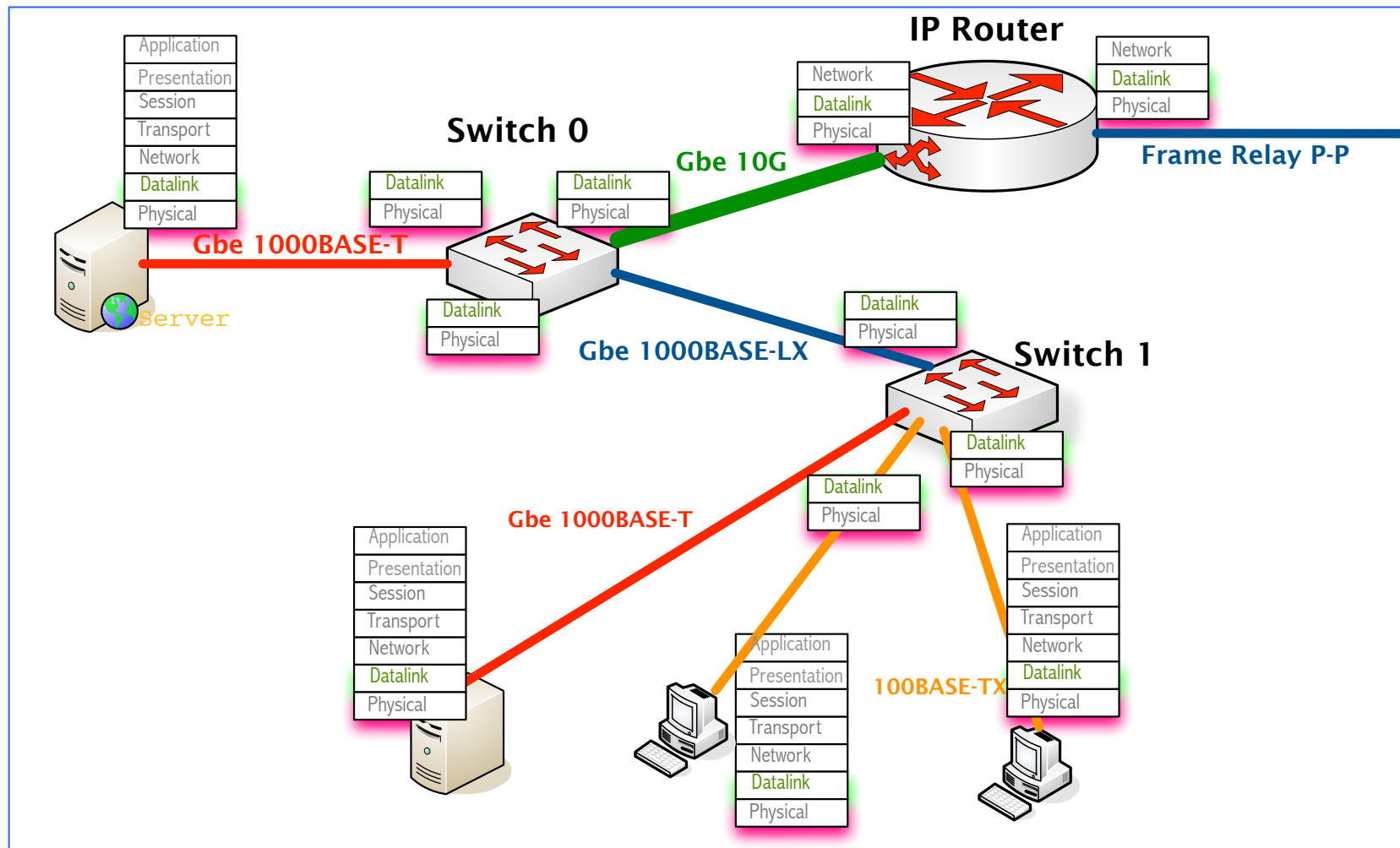
# Links and transmission media

- A link is made up of:
  - ▣ A transmission medium (CAT5e cable in this case)
  - ▣ Signal encoding technique
  - ▣ **Frame** format is the **Datalink** protocol data unit (PDU)



# Datalink protocols in the internetwork

- Each direct connection has its own datalink protocol

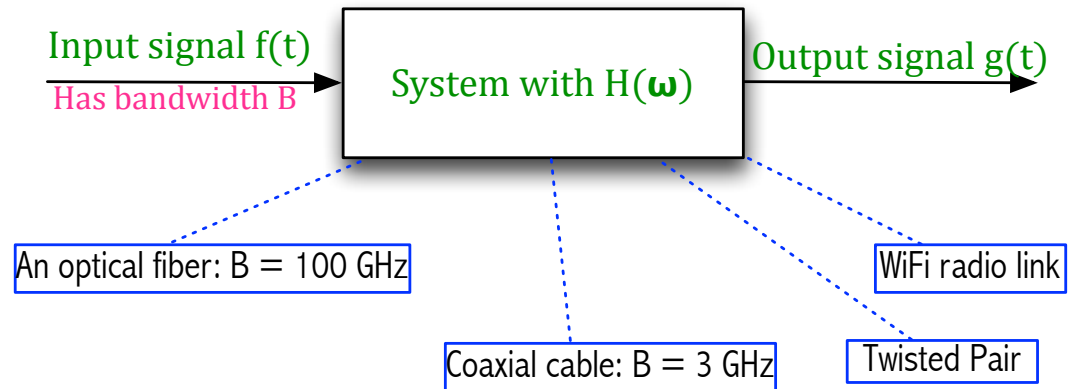
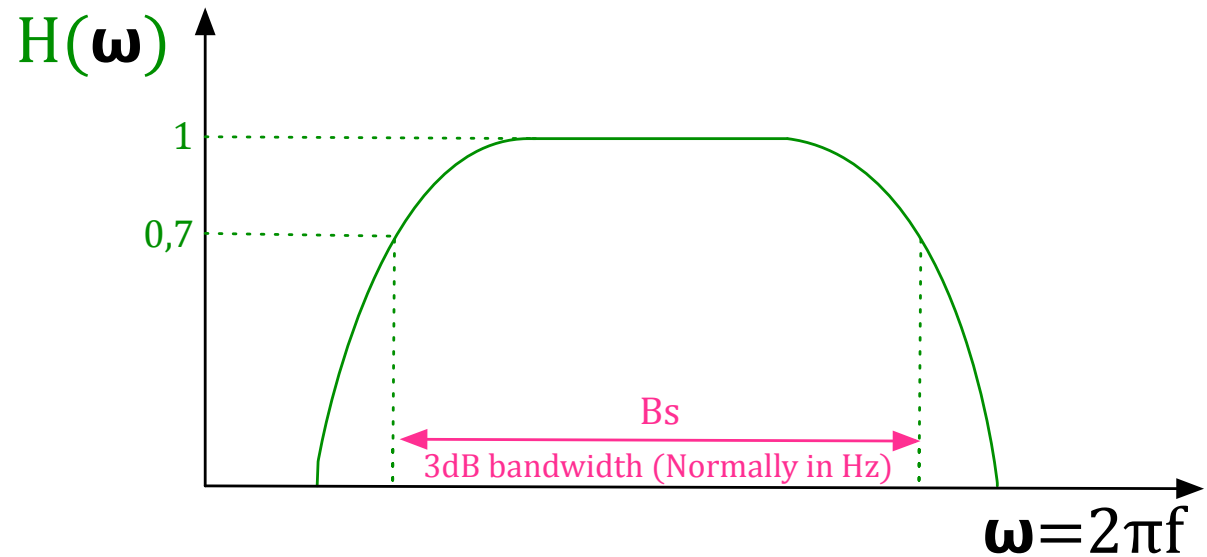




# Communication theory, intro

# If link bandwidth is limited

- How can we determine our data **signal's bandwidth**?
- So that the link properly transmits it



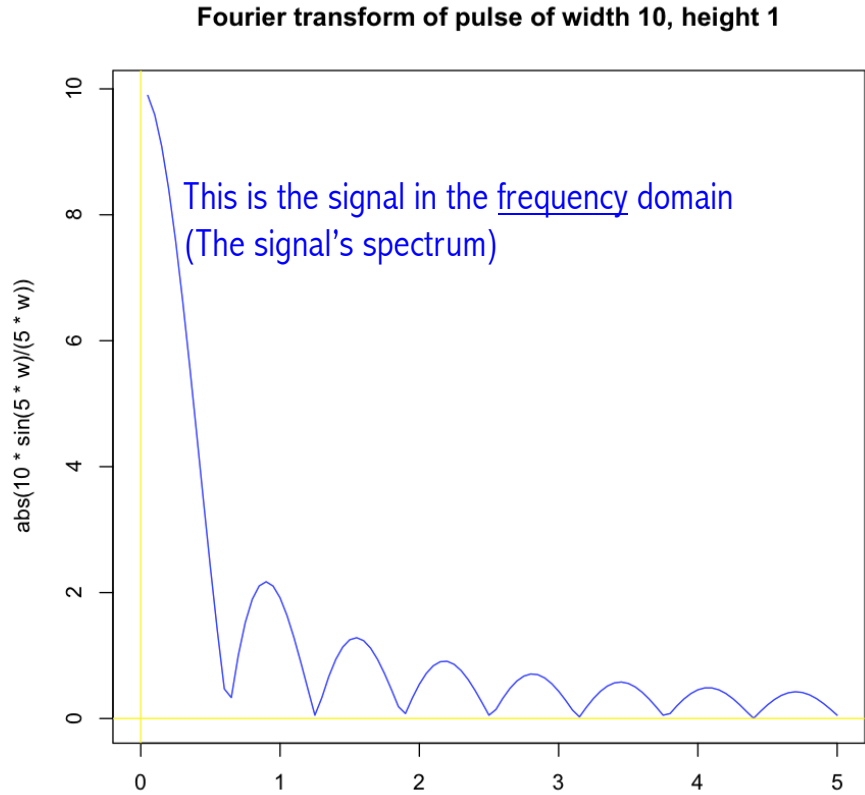
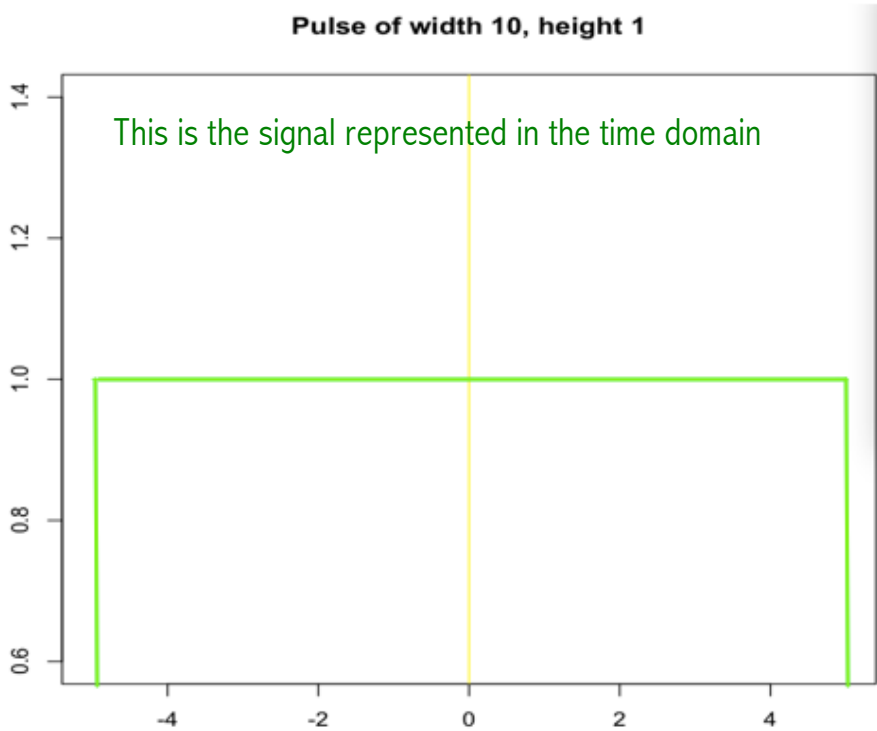


# Communication theory: Fourier transform

- Real signals are composed of an infinite number of sinusoidal signals whose frequency is a real number
  - ▣  $t$  is the real-valued, independent variable of time domain function  $f(t)$
- Fourier transform yields the frequency domain representation of a real signal
  - ▣  $\omega$  is the real-valued, independent variable of function  $F$ , the complex frequency
  - ▣ Natural frequencies  $f$  is such that:  $\omega = 2\pi f$

$$F(j\omega) = \int_{-\infty}^{+\infty} f(t) \cdot e^{-j\omega t} dt$$

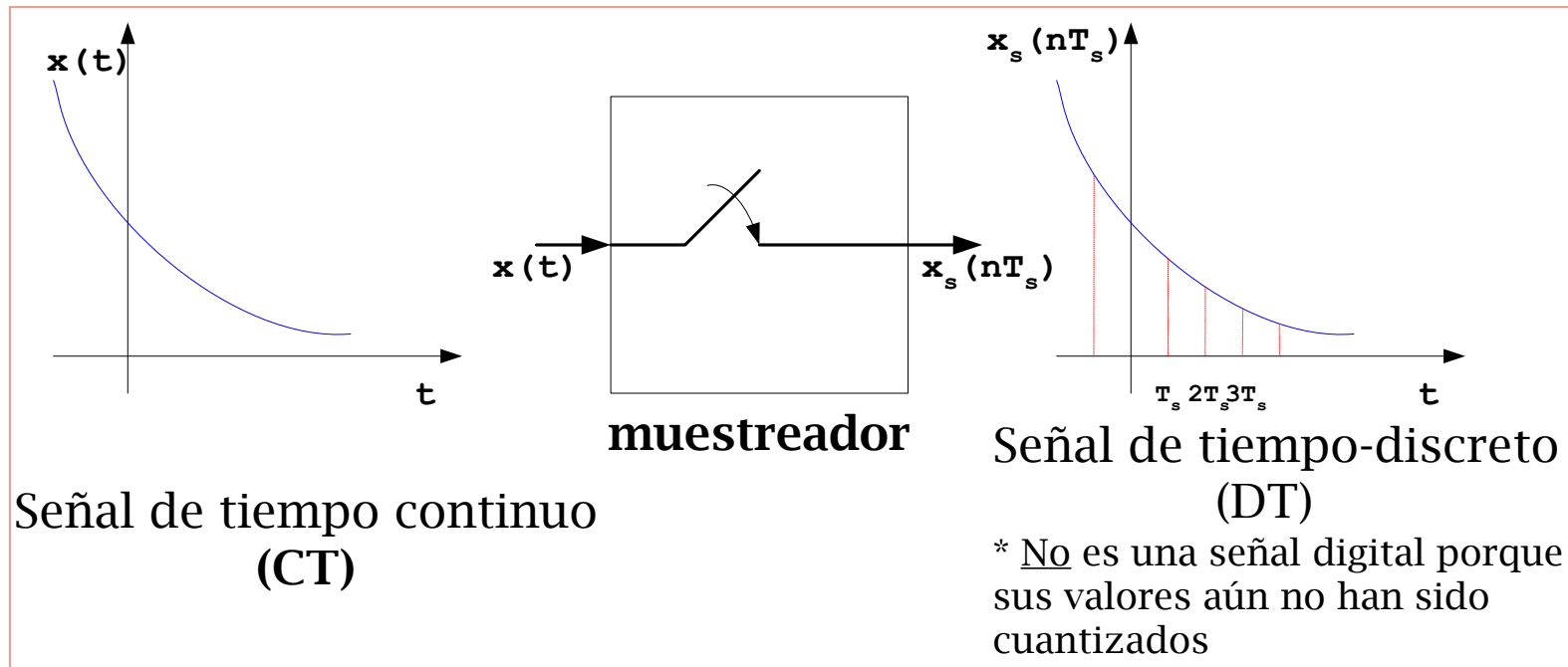
# Signal's bandwidth derived from its Fourier transform



$$F(j\omega) = \int_{-\infty}^{+\infty} f(t) \cdot e^{-j\omega t} dt$$

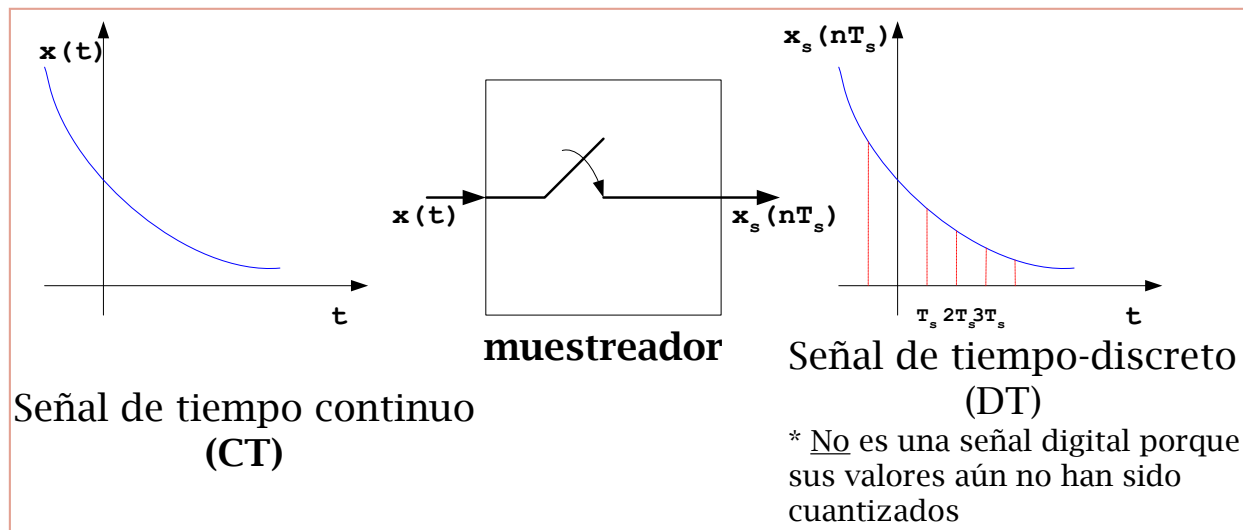
# Communication theory: Signal sampling

- Analog signals can be represented digitally
  1. First, they are sampled (in time)
  2. Then, each sample is quantized



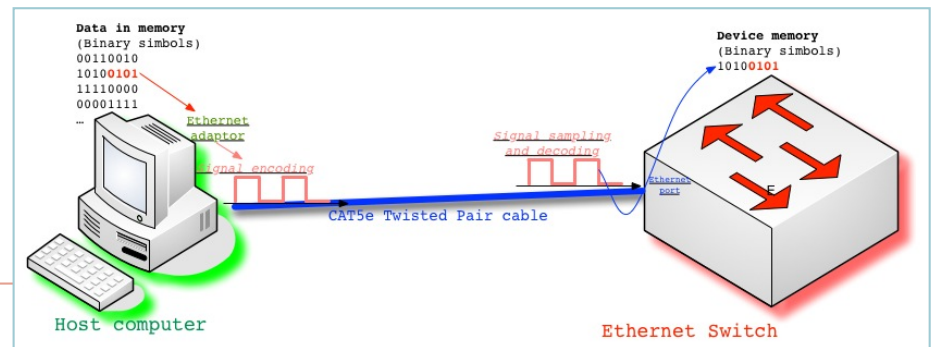
# Communication theory: Nyquist's criterion

- If we want to recover a real signal from its samples:
  - ▣ The sampling process must be carried out **at a speed twice the maximum significant frequency** in the spectrum of the real signal
  - ▣ Otherwise, **aliasing** will occur: new frequency components will appear which will create distortion




# Communication theory: Shannon-Hartley Theorem

- Establishes an upper bound to the capacity of a channel in bps
- If we try to send information through a channel at a higher rate, on the receiving side, the **probability of error in the estimation of the received symbols will be unbounded**
  - Where  $B = 3300 - 300 = 3000\text{Hz}$ ,  $S$  is the signal power,  $N$  the average noise.
  - The signal to noise ratio ( $S/N$ ) is measured in decibels is related to  $\text{dB} = 10 \times \log_{10}(S/N)$ . If there is 30dB of noise then  $S/N = 1000$ .
  - Now  $C = 3000 \times \log_2(1001) = 30\text{kbps}$ .
  - How can we get 56kbps?



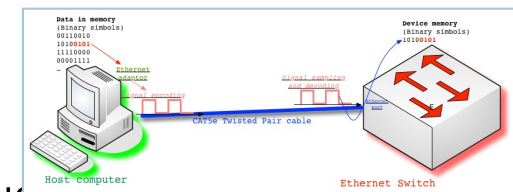
$$C = B \cdot \log_2 \left( 1 + \frac{s}{n} \right) \text{ bps}$$

# Links

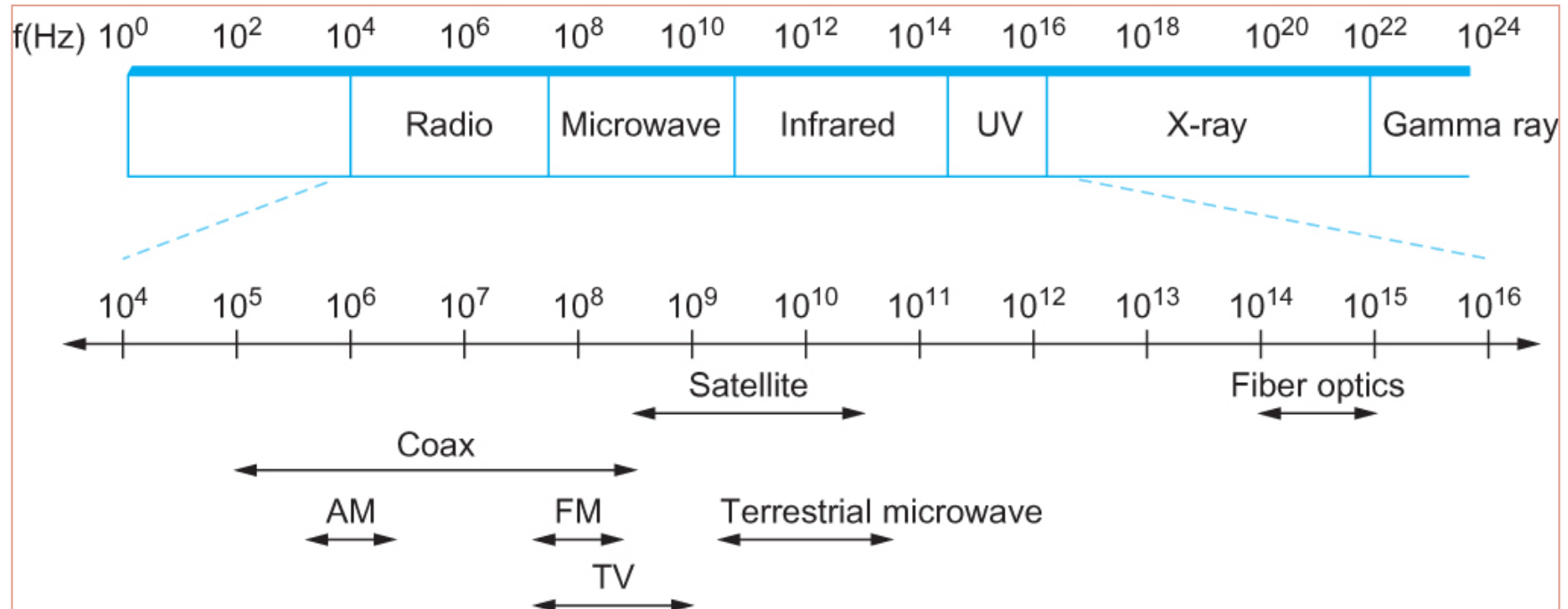
- 
- All practical links rely on some sort of **electromagnetic** radiation propagating through a medium or, in some cases, through free space
  - One way to characterize links, then, is by the medium they use
    - ▣ Typically **copper** wire in some form (as in Digital Subscriber Line (DSL) and coaxial cable),
    - ▣ **Optical fiber** (as in both commercial fiber-to-the home services and many long-distance links in the Internet's backbone), or
    - ▣ Air/**free** space (for wireless links)

# Links

- Another important link characteristic is the *frequency*
  - ▣ Measured in hertz, with which the electromagnetic waves oscillate
  - ▣ Electromagnetic waves propagate as the *electric* field generates a *magnetic* field that generates an electric field ...
  
- Distance between the adjacent pair of maxima or minima of an electromagnetic wave measured in meters is called *wavelength*:  $\lambda = v / f$ 
  - ▣ Speed of light divided by frequency gives the wavelength.
  - ▣ Frequency on a copper cable range from 300Hz to 3300Hz; Wavelength for 300Hz wave through copper is speed of light on a copper / frequency
  - ▣  $2/3 \times 3 \times 10^8 / 300 = 667 \times 10^3$  meters.
  
- Placing binary data on a signal is called *encoding*
  
- *Modulation* involves modifying the signals in terms of their frequency, amplitude, and phase
  - ▣ So that transmission over the physical medium is improved



# Links



Electromagnetic spectrum




# Links



<b>Service</b>	<b>Bandwidth (typical)</b>
Dial-up	28–56 kbps
ISDN	64–128 kbps
DSL	128 kbps–100 Mbps
CATV (cable TV)	1–40 Mbps
FTTH (fibre to the home)	50 Mbps–1 Gbps

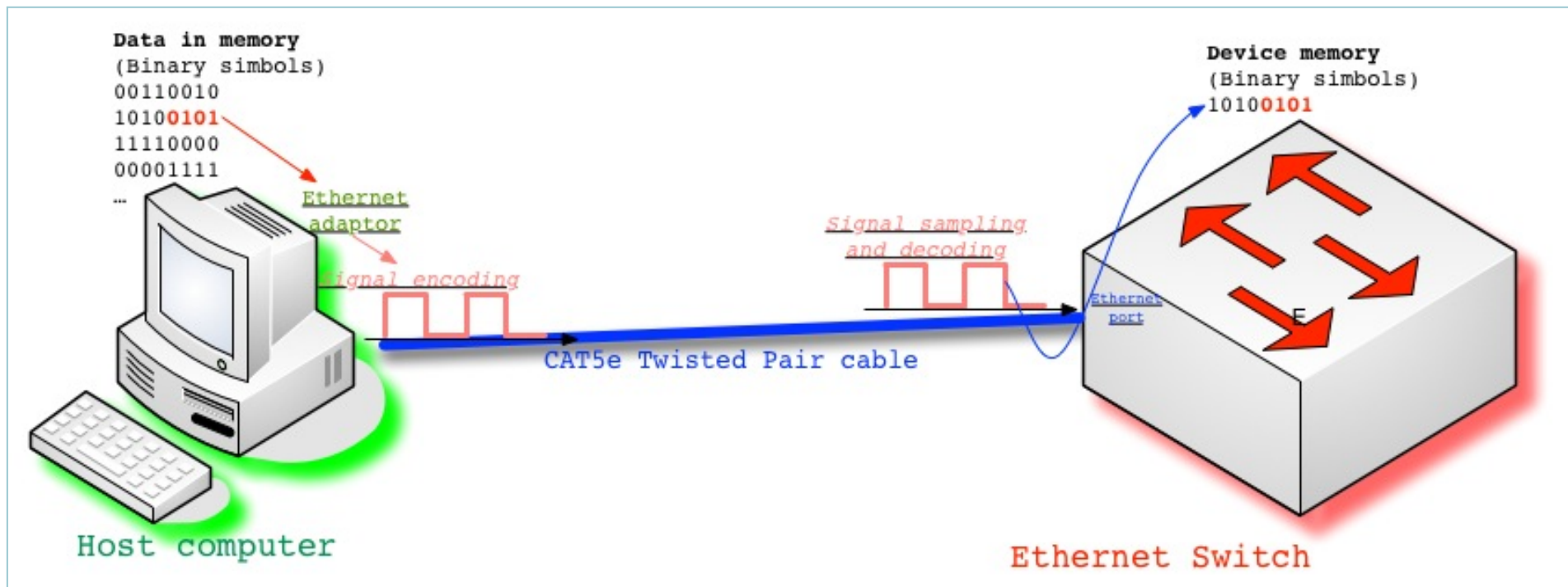
Common services available to connect your home



# Encoding

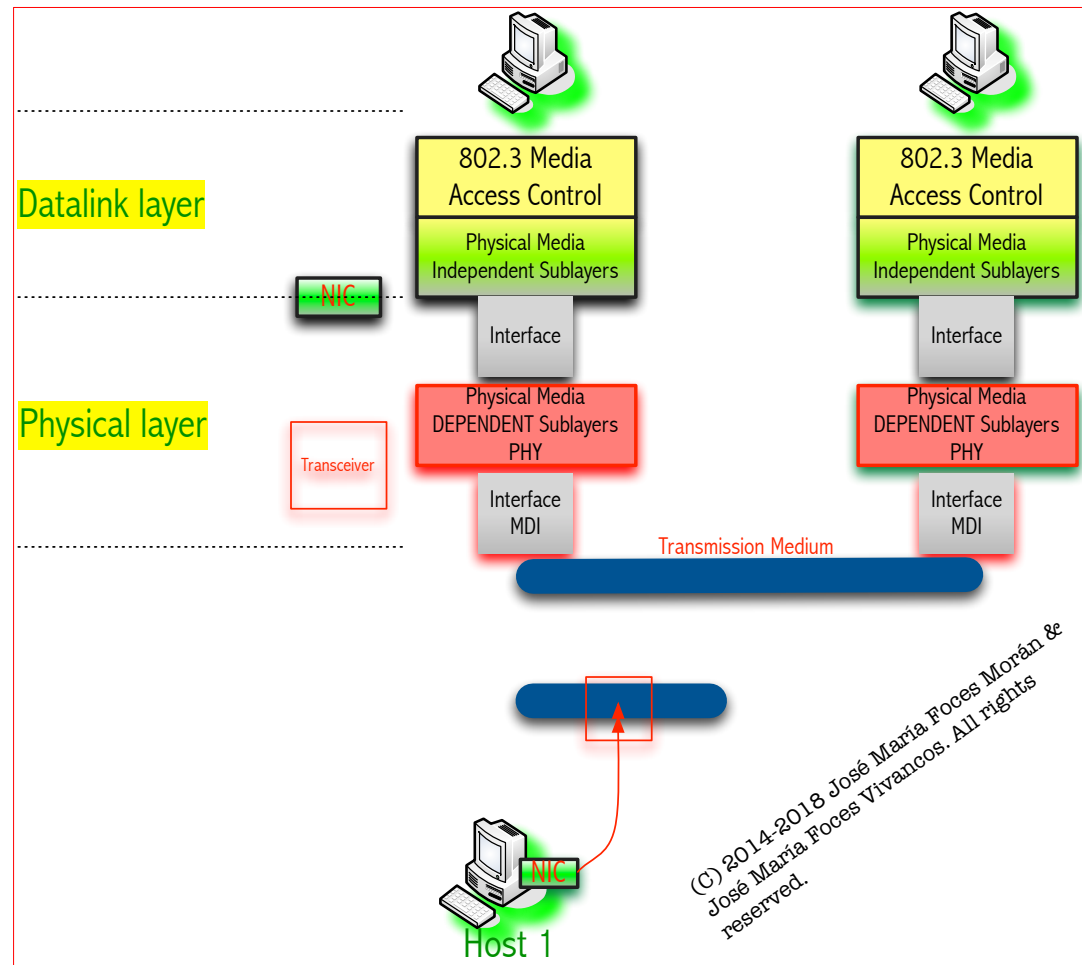
# Line encoding: concept

- The data to be transmitted is stored in memory
- Memory data must be translated into electromagnetic waveforms appropriate for the transmission media
  - ▣ There exist multiple ways of encoding a 0 and a 1: channel line encoding techniques
  - ▣ An essential waveform: pulses (PCM, Pulse Code Modulation)

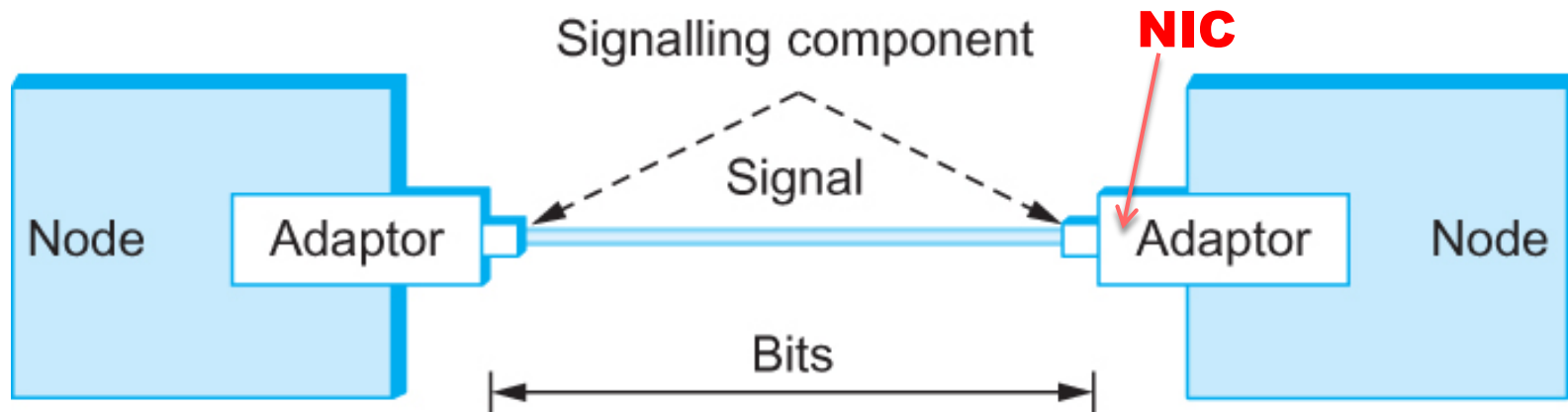


# Line encoding at the Physical Layer

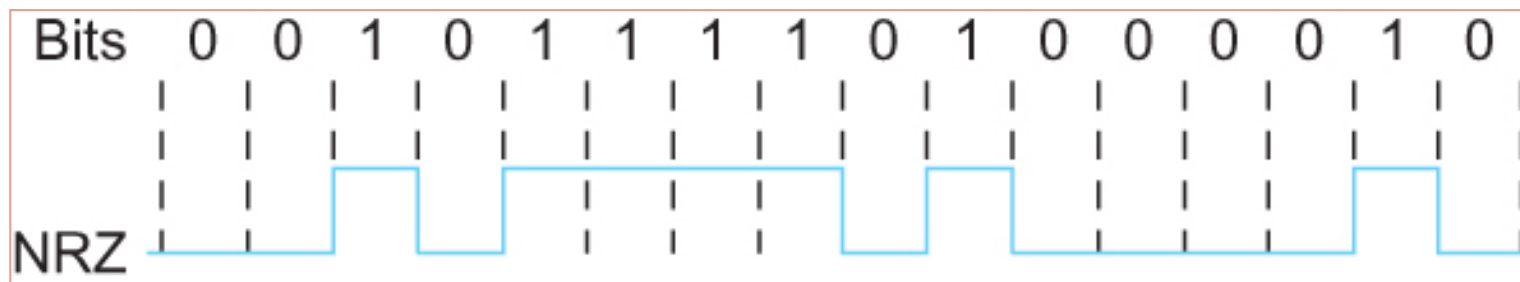
Line Encoding is performed by the Physical Layer module



# NRZ encoding (Non-Return to Zero)



Signals travel between signaling components; bits flow between adaptors

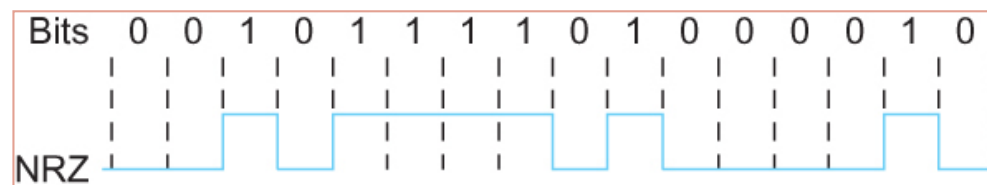


NRZ encoding of a bit stream

# Problems with NRZ

## □ Baseline wander

- The **receiver keeps an average** of the signals it has seen so far
- Uses this average to distinguish between a low level and a high level
- When a signal is significantly low than the average, it is 0, else it is 1
- Too many **consecutive 0's** and **1's** cause this average to shift, thereby making it more **difficult to differentiate between the levels**

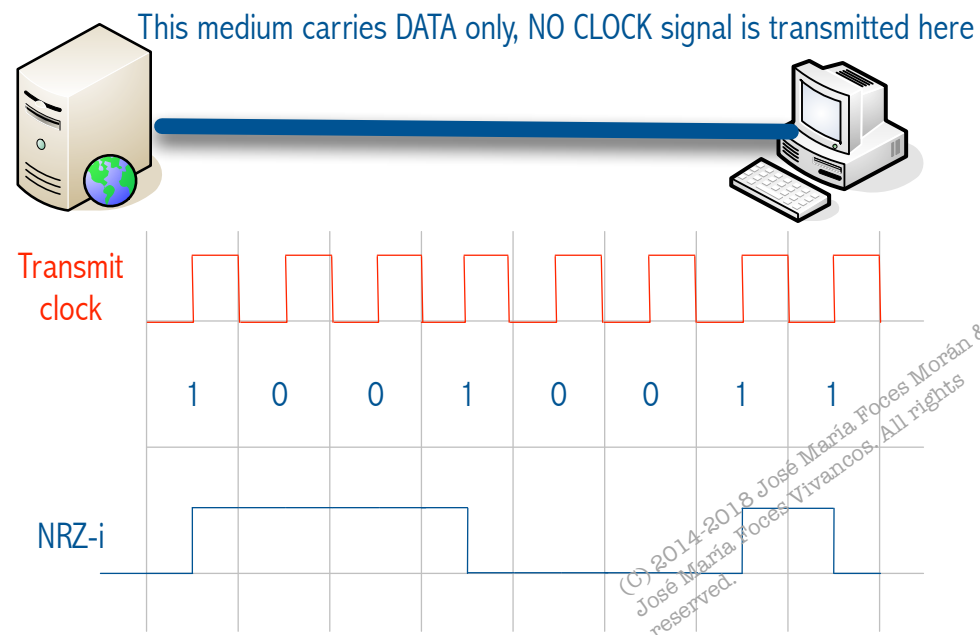


NRZ encoding of a bit stream

# Problems with NRZ

## □ Clock recovery

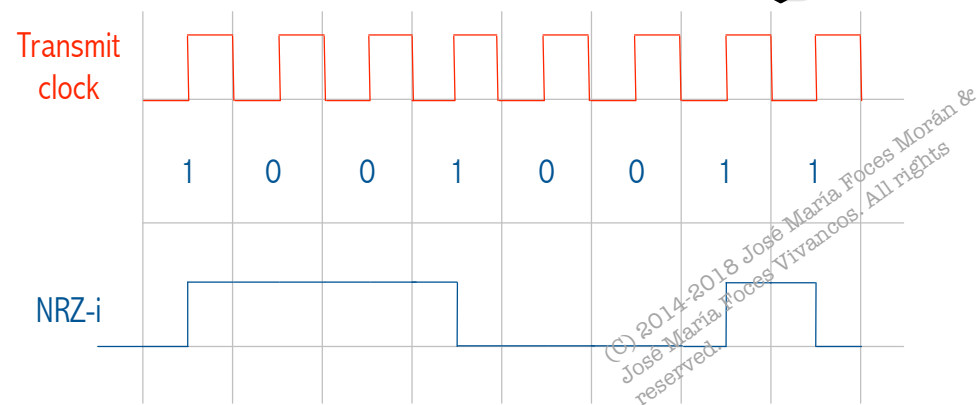
- ▣ The transmitter sends symbols (0/1) at some transmission speed determined by an internal clock signal
  - In data communications this clock signal is not sent from sender to receiver
  - Then, how does the receiver become aware of the used transmission speed?
- ▣ At every clock cycle, the sender transmits a bit
- ▣ The receiver must be able to derive the transmission speed
  - This entails frequent transitions from high to low or vice versa in the received data signal
  - This is known as clock recovery
  - Clock recovery yields a precise synchronization of sender and receiver



# NRZI: a partial solution to NRZ

## □ NRZI

- ▣ Non Return to Zero Inverted
- ▣ Sender makes a transition from the current signal to encode 1 and stay at the current signal to encode 0
- ▣ Solves for the consecutive 1's problem of NRZ





# Manchester: complete solution to NRZ

## □ Strategy:

- Merge the clock with signal by transmitting Ex-OR of the NRZ encoded data and the clock
- Clock is an internal signal that alternates from low to high, a low/high pair is considered as one clock cycle
- In Manchester encoding
  - 0: low → high transition
  - 1: high → low transition

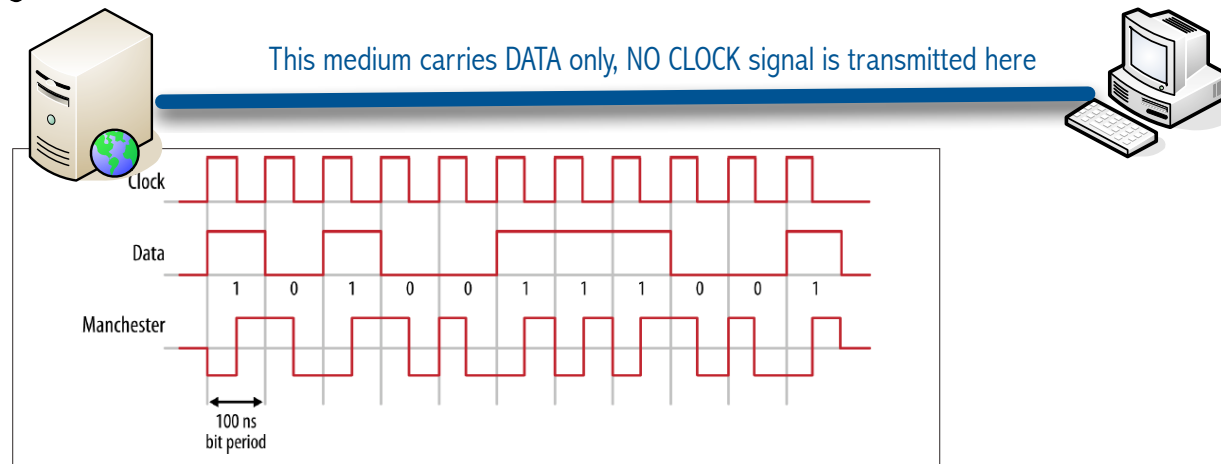


Figure 8-1. Manchester signals over 10BASE-T

# Manchester encoding problem

- Manchester doubles the rate of signal transitions present on the link
  - ▣ Which means the receiver has half of the time to detect each pulse of the signal
  - ▣ The rate at which the signal changes is called the link's **baud rate**
  - ▣ In Manchester the *bit rate is half the baud rate*

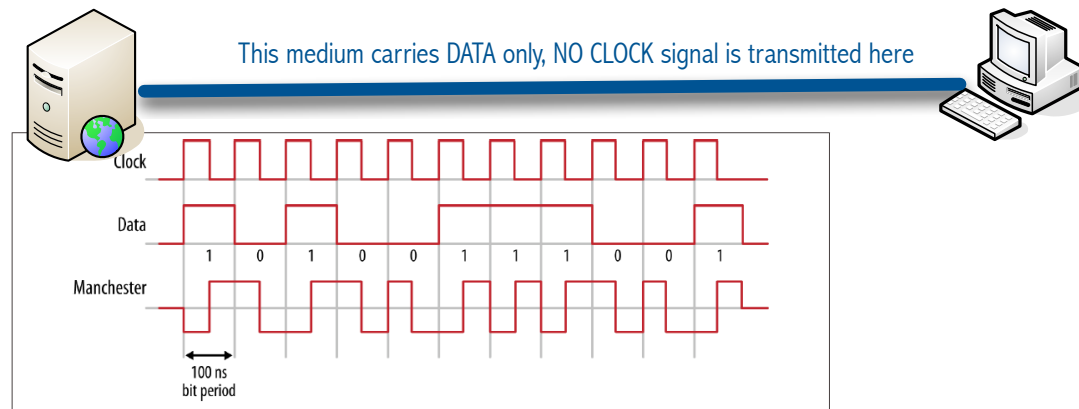
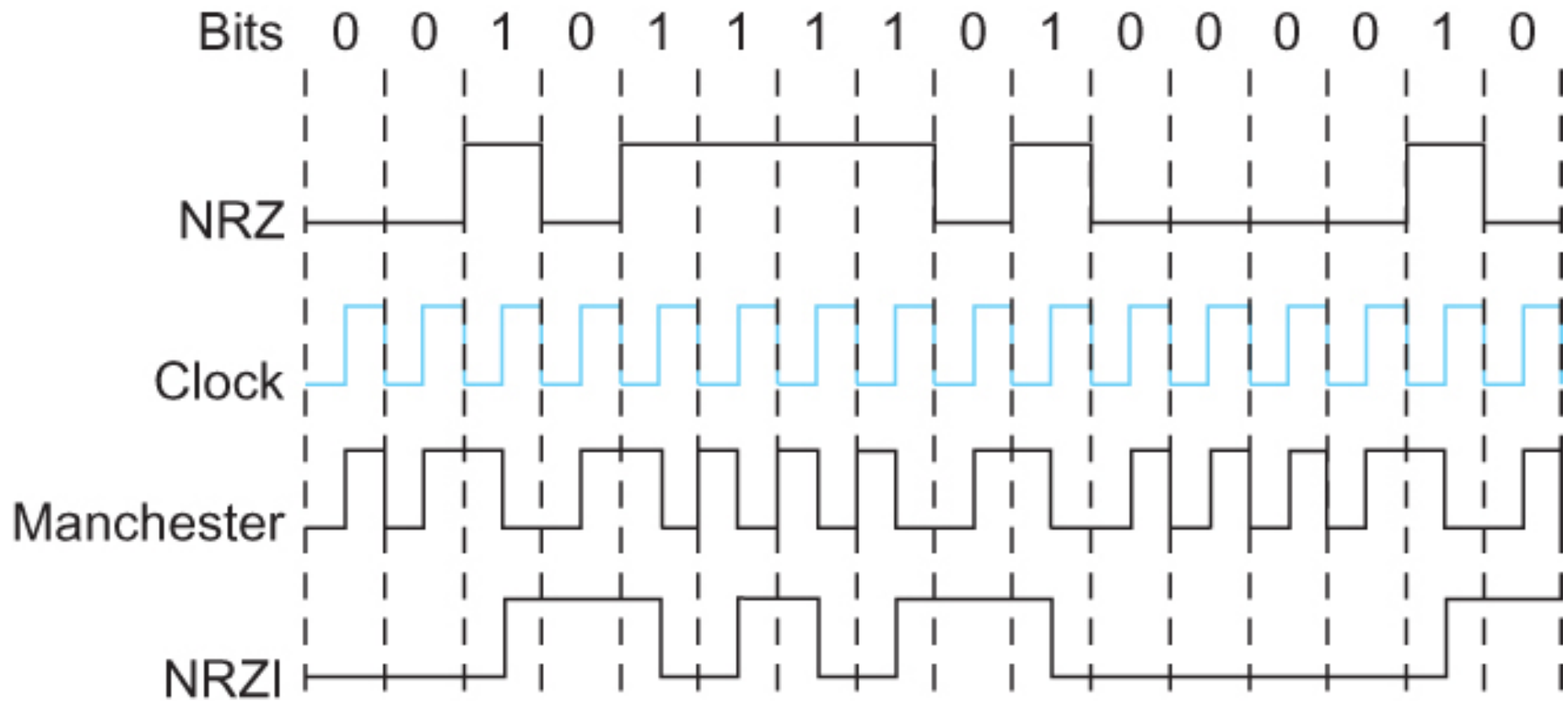


Figure 8-1. Manchester signals over 10BASE-T

# Summary of encoding



Different encoding techniques

# Channel Encoding: 4B/5B

- It's a **channel (block) encoding** technique
- Performed by the transmitter before transmission, the sender inserts extra bits into bit stream so as to break up the long sequences of 0's
  - ▣ Every group of 4-bits of actual data are encoded into a 5-bit code which is transmitted to the receiver
  - ▣ 5-bit codes are selected in such a way that each one has
    - no more than one leading 0 (zero)
    - no more than two trailing 0's
    - 01100 Ok, 00111 Not ok, 11000 not ok ...
  - ▣ No pair of 5-bit codes results in more than three consecutive 0's

# Source Channel Encoding: 4B/5B

## □ 4-bit data -> 5-bit code

16 4-bit combinations

0000 → 11110

0001 → 01001

0010 → 10100

...

1111 → 11101

16 left:

11111 – when the line is idle

00000 – when the line is dead

00100 – means halt

13 left

- 7 invalid
  - Not complying with 0..00
- 6 for various control signals (framing)

# Channel Encoding in the first place, then Line Encoding

- 4B/5B solves the problem of long sequences consecutive of 0's
- NRZI solves the problem of long sequences of consecutive 1's
  
- These encodings are used in tandem:
  - ▣ 4B/5B is applied first to the data to be transmitted
    - This is known as CHANNEL ENCODING
  - ▣ Then, the resulting bit stream is NRZI encoded
    - This is known as LINE ENCODING

(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.



# Framing

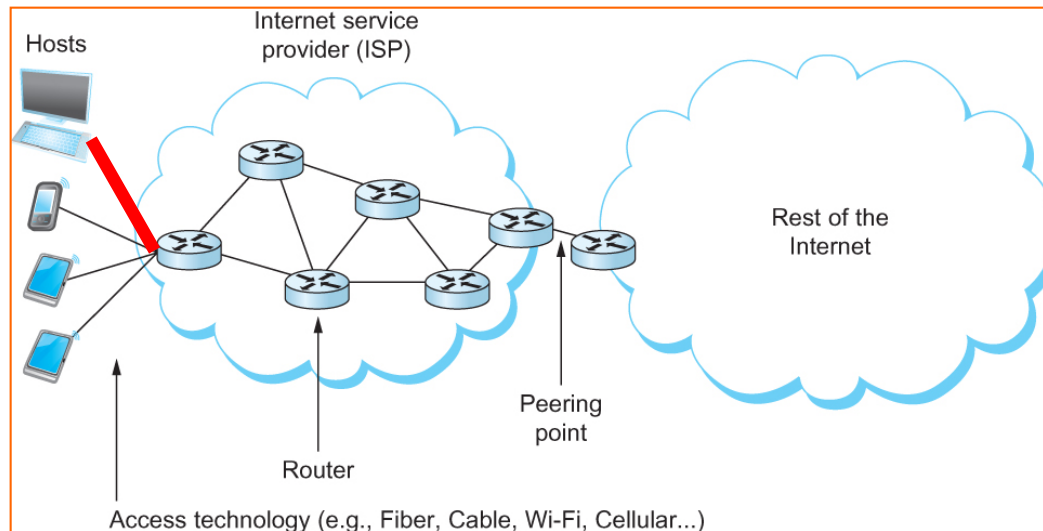
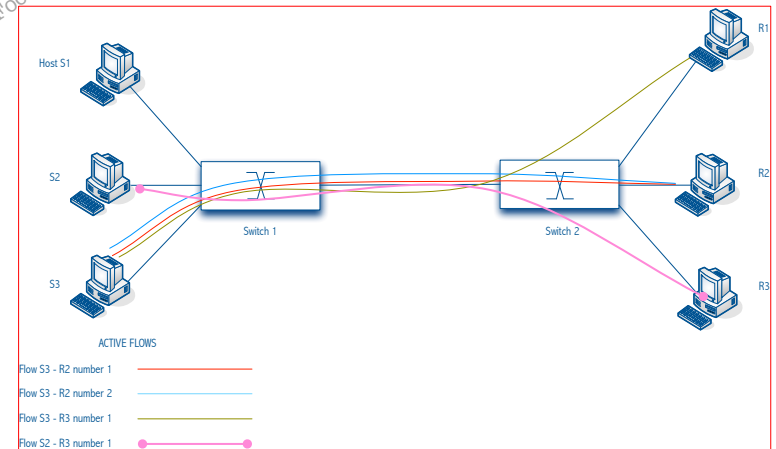


# Framing

Our focus in this chapter is on **Packet-switched networks**

- **Statistical multiplexing**
  - ▣ Information sent must be added its source and destination addresses
  - ▣ Info is not transferred in isolation
- Directly connected nodes exchange blocks of data known as **Frames**
  - *Example: Host --- Router*
  - **FRAME** = Payload + Layer-2 addressing data (Multiplexing data)
- Frames, not bit streams

(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.

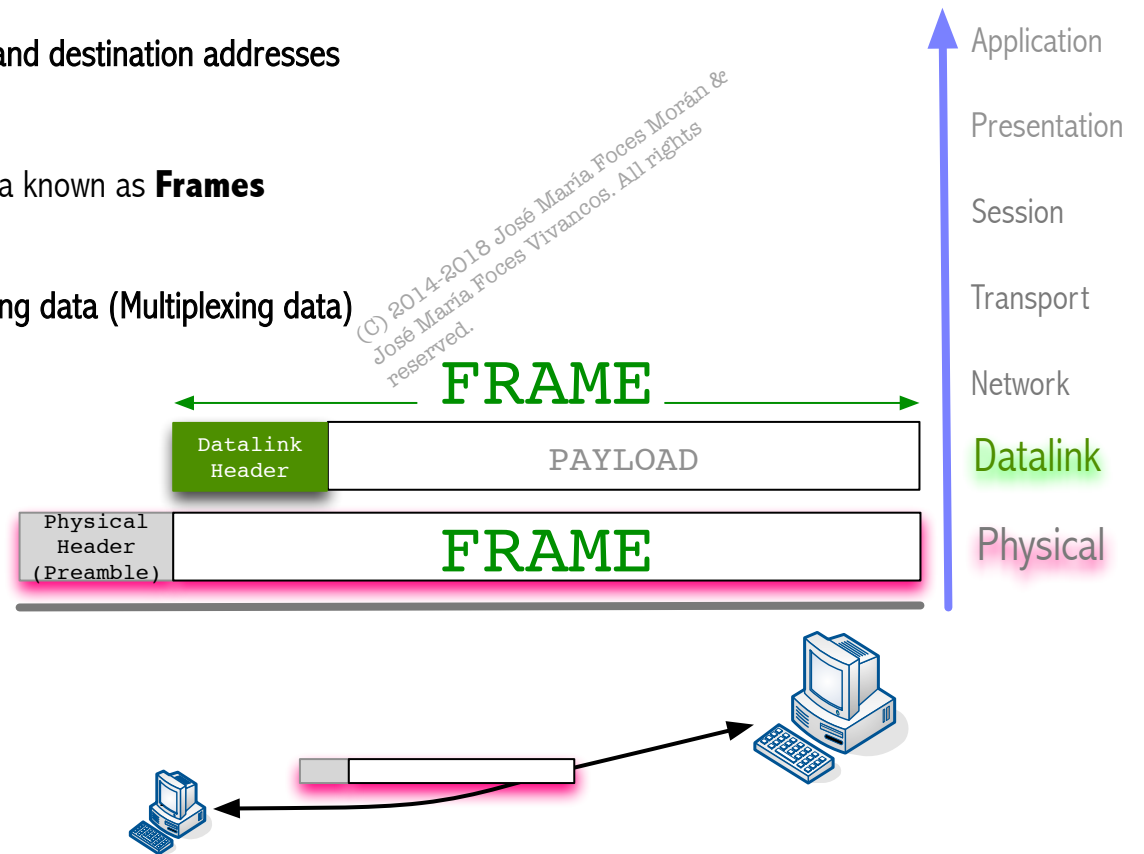




# Framing

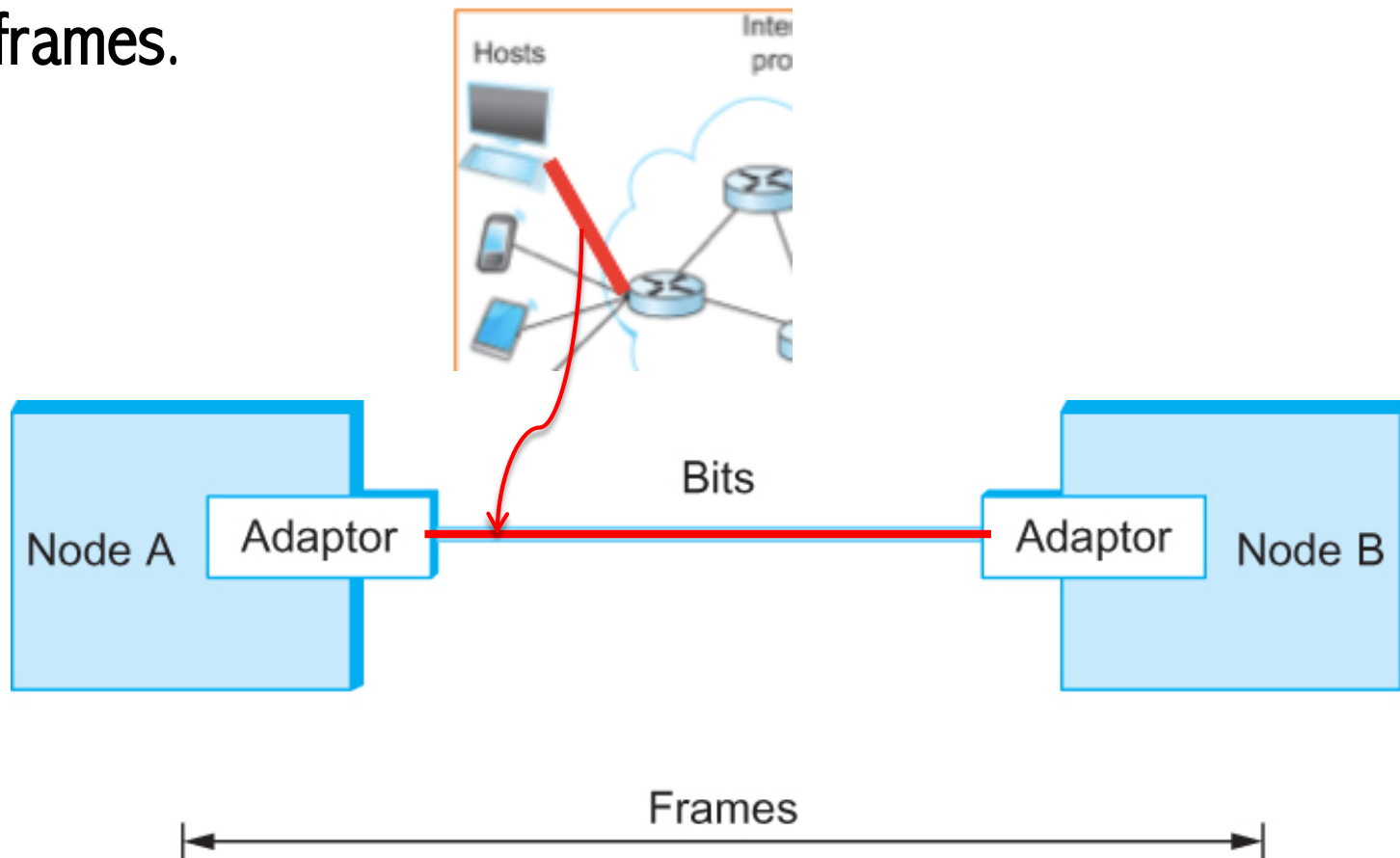
Our focus in this chapter is on **Packet-switched networks**

- **Statistical multiplexing**
  - Information sent must be added its source and destination addresses
  - Info is not transferred in isolation
- Directly connected nodes exchange blocks of data known as **Frames**
  - Example: Host --- Router
  - **FRAME** = Payload + Layer-2 addressing data (Multiplexing data)
- Frames, not bit streams



# Framing

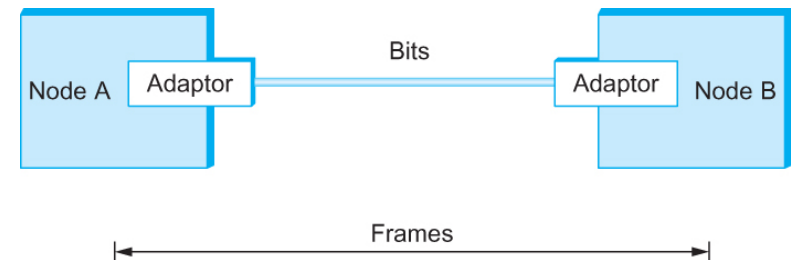
- It is the **network adaptor** that enables nodes to exchange frames.



*Bits flow between adaptors, frames between hosts*

# Framing

- Node A transmits a frame to node B
  - A sequence of bits being sent over the link.
- Adaptor on node B **collects** the sequence of bits
  - Deposits the **frame** in B's memory.
- Recognizing exactly what set of bits constitute a frame
  - Where the frame **begins** and **ends**
  - Central **challenge**



# Framing

- What set of bits constitute a frame?
  - ▣ Where the frame begins and ends
  - ▣ Central challenge faced by the adaptor
    - Three strategies:
      - **Byte-oriented** protocols: BISYNC, PPP, DDCMP
      - **Bit-oriented** protocols: HDLC, Ethernet
      - **Clock-based** protocols: SONET/SDH

# Framing

□ C strings: byte-oriented 😊

□ How is a C constant character string delimited?

```
char s[] = "Hello world!";
```

□ “ **sentinel** that marks the beginning

□ Next “ **sentinel** that marks the end

▣ ASCII Characters are stored in between the two delimiters

# Framing



- Byte-oriented protocols

- What is a **frame**? A collection of **bytes**, multiple of 8 bits

- Example protocols:

- BISYNC (Binary Synchronous Communication, BSC)

- Developed by IBM (late 1960)

- DDCMP (Digital Data Communication Protocol)

- Used in DECNet

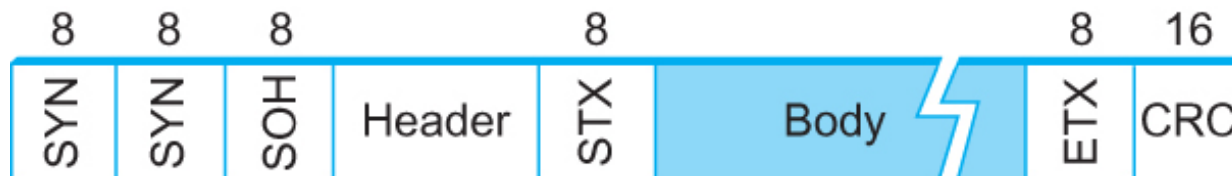
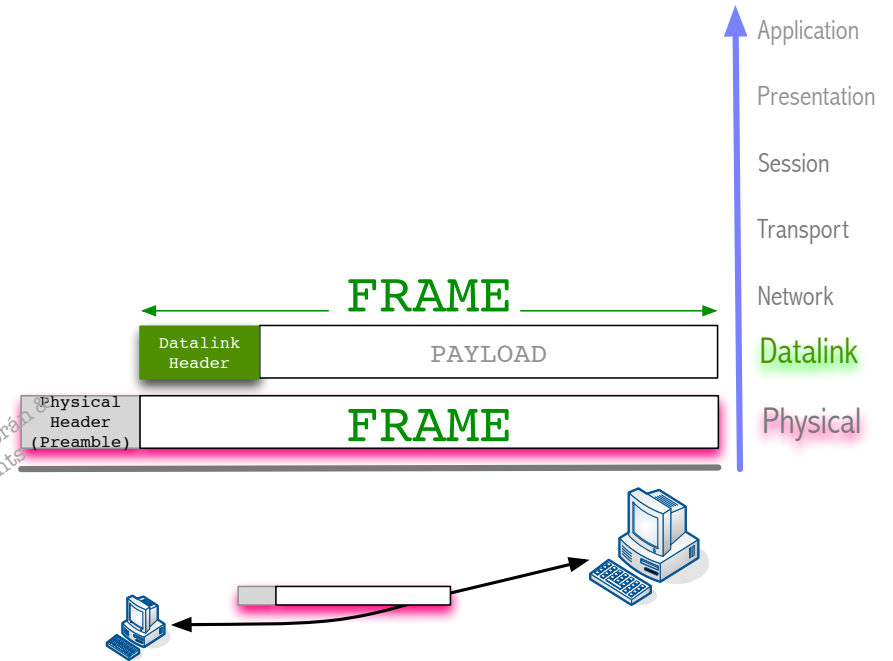
- PPP (Point to Point Protocol)

- IP packets over various media

# Framing

## BISYNC

- Byte-oriented
- Beginning of a frame:
  - ▣ Send a special ASCII/EBCDIC character named **SYN**
- Uses sentinels for signalling (They are ASCII control characters):
  - ▣ The start of the datalink header: **SOH (Start of Header)**
  - ▣ The start (**STX**) and end (**ETX**) of the layer-3 payload
- Frames transmitted beginning with leftmost field
- **Byte-transparency**
  - ▣ **What if the payload contains a *data* byte combination coincident with any of the sentinels?**
  - ▣ A special control character known as **DLE (Data Link Escape)** indicates that the next character is not to be understood as a sentinel but as pure literal data
  - ▣ Example:
    - ▣ We want to send the following ASCII character sequence as data: "[A][B][C][D][E][STX][F][G]"
    - ▣ The STX char is not to be understood as meaning "Start of TeXt" but its 8 bits mean only payload data
    - ▣ Then, we include a preceding [DLE] character meaning: "The next character is data, it is not the Bisync sentinel known as[STX]"
    - ▣ The transmitted sequence becomes:
      - ▣ "[A][B][C][D][E][DLE][STX][F][G]"
    - ▣ What if DLE itself appears in the layer-3 payload? Same as in the C language: Include an escaping DLE character that escapes the special meaning of the next character: [DLE][DLE]
- CRC: Error control



BISYNC Frame Format

# Framing

## PPP (Point to Point Protocol)

- **Byte-oriented** (A variant of HDLC-ABM protocol)
- Uses **sentinel** approach
- Over Internet links (ISDN/ADSL/ATM)
- **start of frame** character sentinel is denoted as **Flag**  
0 1 1 1 1 1 1 0
- Address (0xff), control (0x03): default numbers
- Protocol: A **demultiplexing key** (Example: IP / IPX)
- Payload: The data transported, size **negotiated** (1500 bytes)
- Checksum: error detection



PPP Frame Format



# Framing

## PPP (Point to Point Protocol)

- Payload: The transported data, size is **negotiated** (1500 bytes)
- Works in tandem with another two protocols
  - ▣ **Negotiate** parameters with:
    - ▣ LCP (Link Control Protocol): For testing and managing the link
    - ▣ NCP (Network Control Protocol): IP address, default router, etc



PPP Frame Format

# Framing

## Byte-counting approach

### □ DDCMP

- ▣ *count*: how many bytes are contained in the frame body
- ▣ *-the rest of fields are fixed-size*

### □ If *count* is corrupted

- ▣ Framing error



DDCMP Frame Format

# Framing

## Bit-oriented protocols

- HDLC : High Level Data Link Control
  - ▣ Beginning and Ending Sequence (Sentinel is known as FLAG)  
FLAG = 0 1 1 1 1 1 1 0
  - ▣ Transmits data in blocks of 1 bit
  - ▣ Any amount of bits, not necessarily a multiple of 8 bits(1 byte)

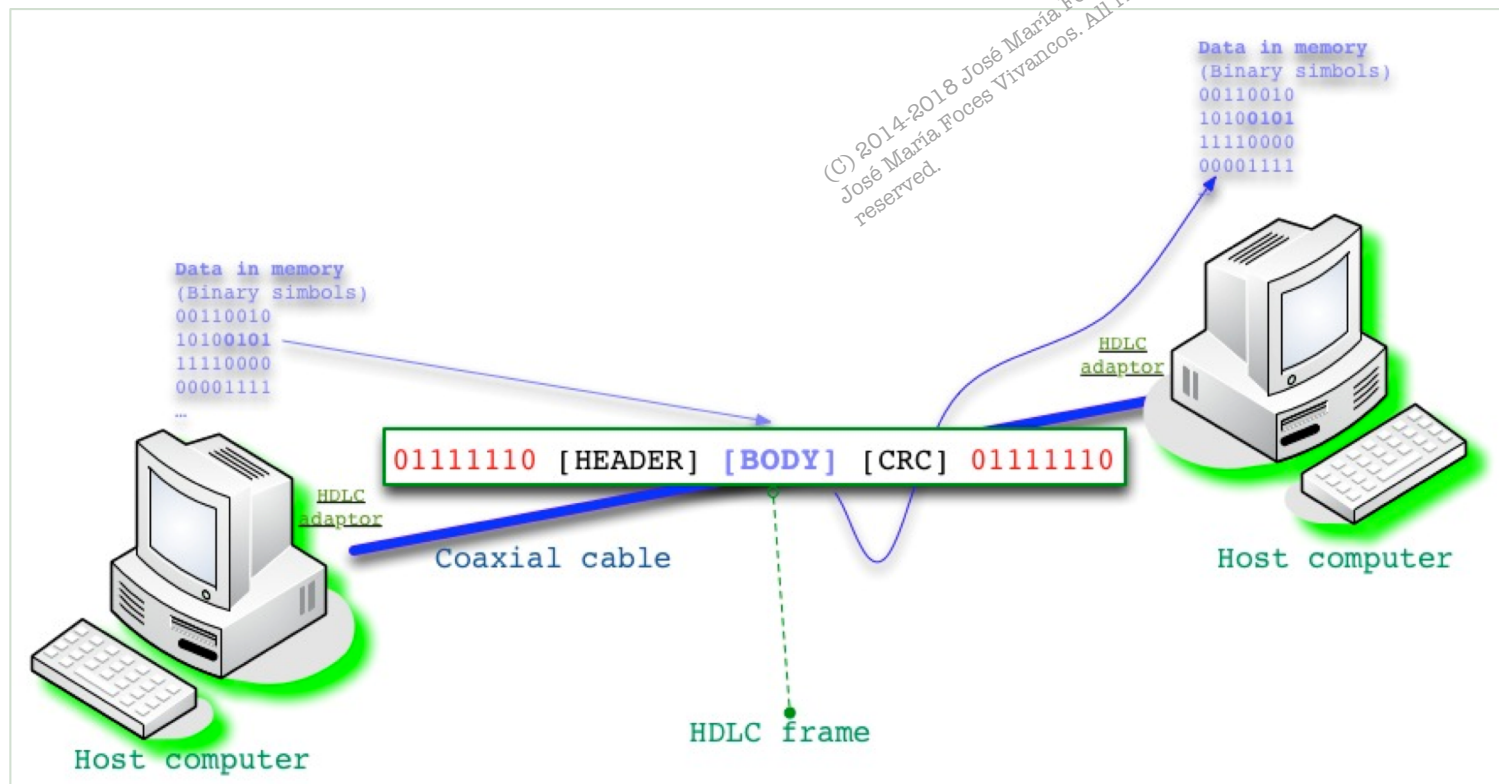


HDLC Frame Format

# Framing

## Bit-oriented protocols

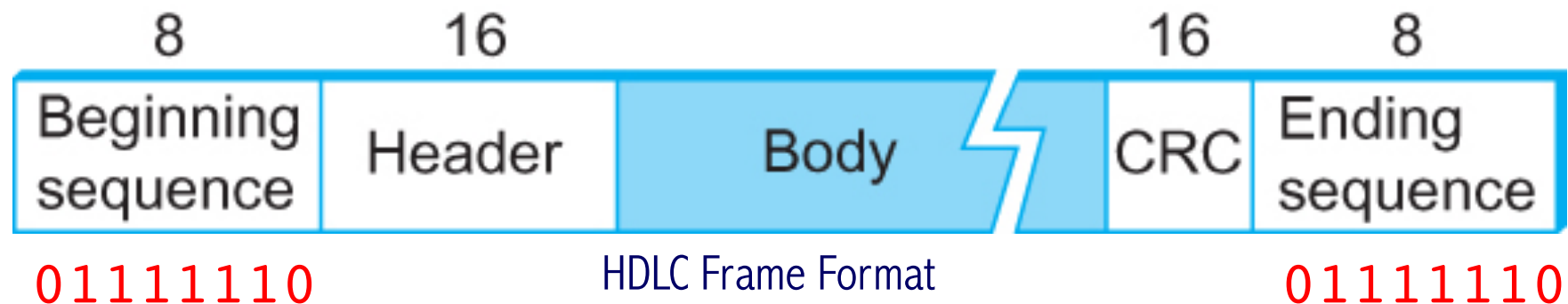
- HDLC : High Level Data Link Control
  - ▣ Beginning and Ending Sequence  
Flag = 0 1 1 1 1 1 0
  - ▣ Transmits data in blocks of 1 bit
  - ▣ Any amount of bits, not necessarily a multiple of 8 bits(1 byte)



# Framing

- HDLC protocol

- *Problem: What if the FLAG 0 1 1 1 1 1 1 0 is contained within the frame?*
- *Solution in HDLC: Bit stuffing*



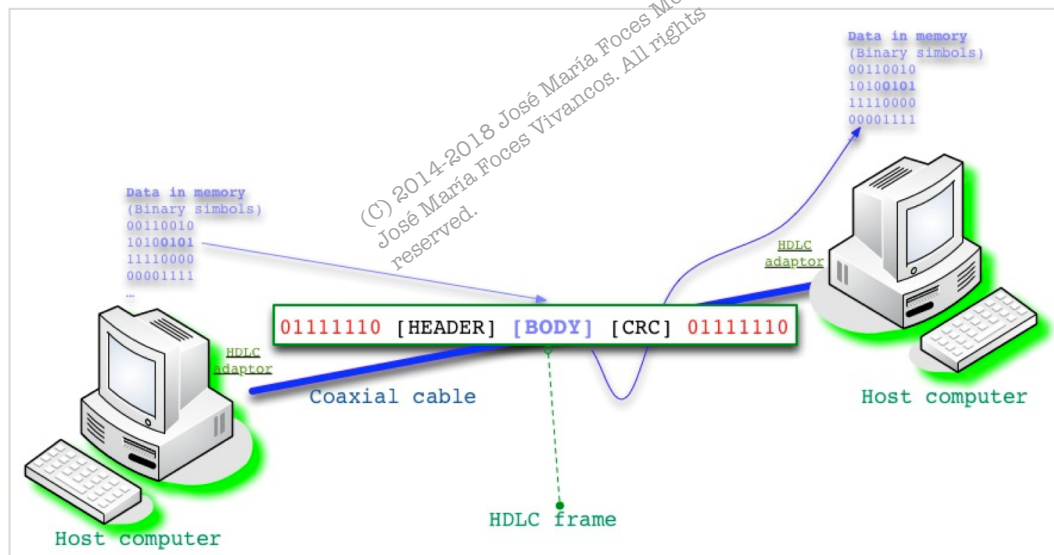
# Framing

HDLC, bit stuffing on the sending side

- Any time five consecutive 1's appear on the data in memory
- The sender inserts (*stuffs*) 0 before transmitting the next bit

□ 0111110<sub>10</sub>

□ (Not when the sender is trying to send the Flag 0111110)



# Framing

## HDLC, bit stuffing on the receiving side

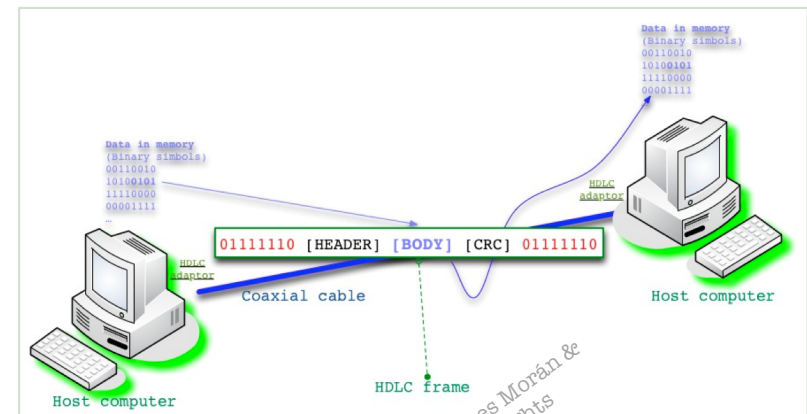
□ When 5 consecutive 1's are received 011111

□ If next bit 0: (01111100001010...)

- It's a stuffed 0, so discard it and keep receiving the ensuing bits (0001010...)

□ If next bit 1: (01111110001010...)

- Look at next bit (011111b)
- If 0: End of frame marker (0111110)
- If 1: Error has been introduced in the bitstream (0111111)



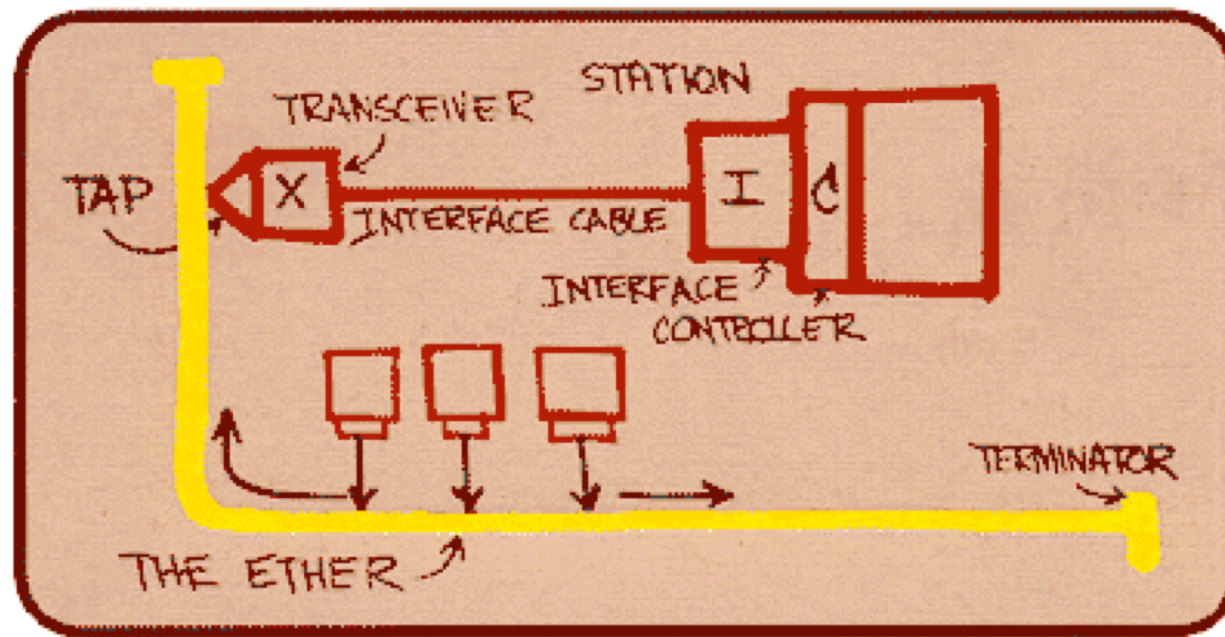
# Ethernet, intro

- A brief introduction to the original Ethernet, a shared medium local network technology with an access arbitration known as CSMA/CD
- Today the most prevalent form of Ethernet is the switched Ethernet which we will take up in chapter 3



# Ethernet

- Most **successful** local area networking technology of last 20 years.
- Developed in the mid-1970s by researchers at the Xerox Palo Alto Research Centers (PARC).

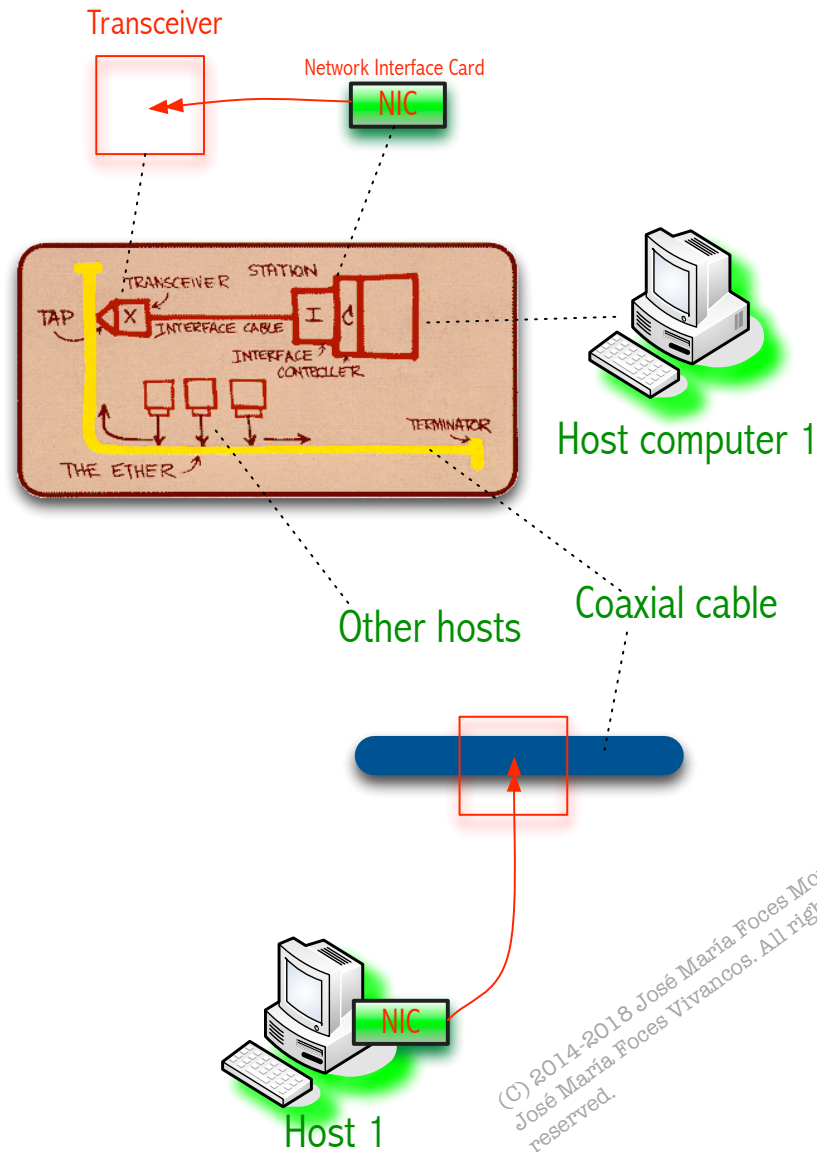


Original Ethernet drawing (© Bob Metcalfe)

# Ethernet

- Uses ALOHA (packet radio network) as the root protocol
  - ▣ Developed at the University of Hawaii to support communication across the Hawaiian Islands
  - ▣ For ALOHA the medium was atmosphere, for Ethernet the medium is a coax cable
  - ▣ Today's 802.11 (WiFi) protocols are based on the ideas developed in the Aloha network
- DEC and Intel joined Xerox to define a **10-Mbps Ethernet** standard in 1978
- This standard formed the basis for **IEEE standard 802.3**
  
- More recently **802.3** has been extended:
  - ▣ **100-Mbps**
    - Fast Ethernet
  - ▣ **1000-Mbps**
    - Gigabit Ethernet
  - ▣ **10G+**

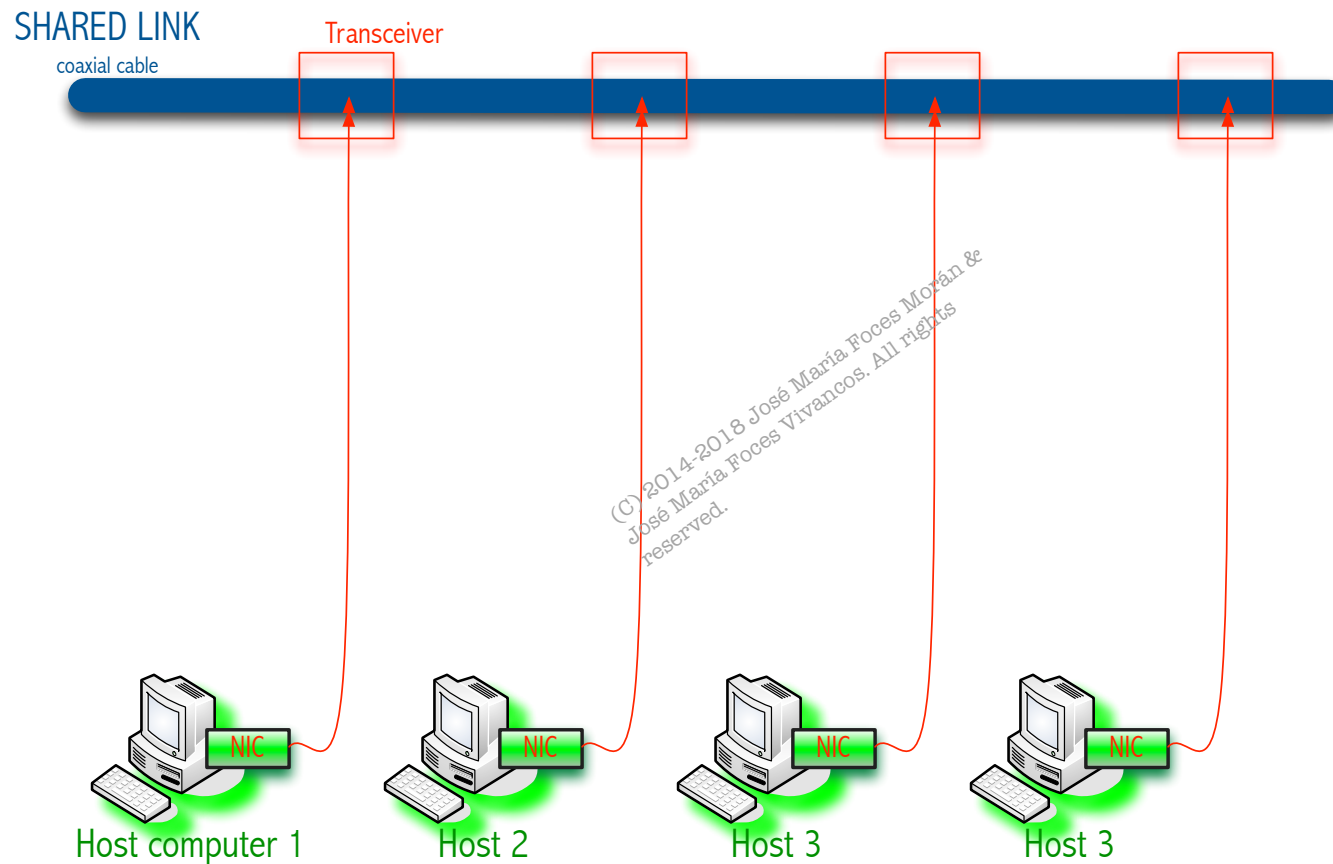
# Our Ethernet diagrams vs. Metcalf's



© 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.

# Ethernet

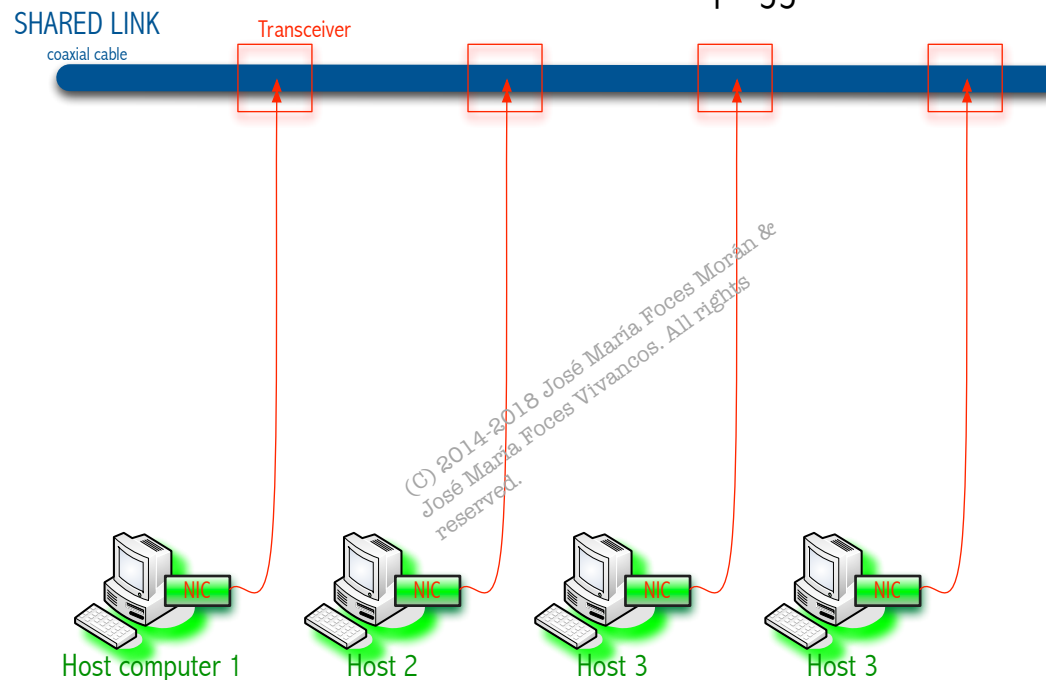
- An *Ethernet segment* is implemented on a coaxial cable of up to 500 m
- *Host* connects to an Ethernet segment by using a NIC (Network Interface Card)



# Ethernet

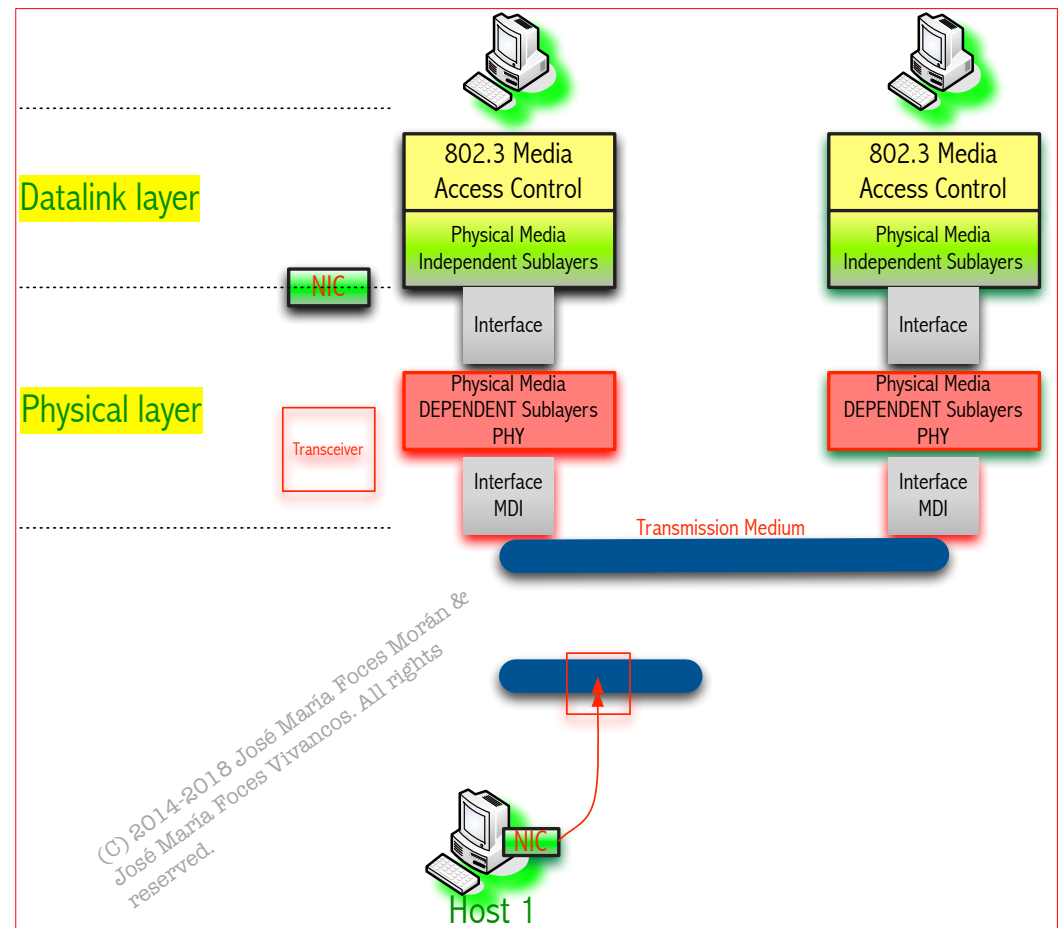
- **NIC** (Network Interface **Adaptor**)
  - ▣ **Datalink** protocol is implemented on NIC
  - ▣ NIC taps into the coaxial cable with a transceiver

- **Transceiver** (a small device directly attached to the tap)
  - ▣ Detects when the **line is idle** by applying *Carrier Sense = CS*
  - ▣ Drives signal when the host is transmitting
  - ▣ Receives incoming signal
  - ▣ Connected to an **Ethernet adaptor** which is plugged into the host.



# Ethernet

- **NIC** (Network Interface Adaptor)
  - ▣ Datalink protocol is implemented on NIC
  - ▣ NIC taps into the coaxial cable with a transceiver
- **Transceiver** (a small device directly attached to the tap)
  - ▣ Detects when the **line is idle**: **Carrier Sense = CS**
  - ▣ Drives signal when the host is transmitting
  - ▣ Receives incoming signal
  - ▣ Connected to an **Ethernet adaptor** which is plugged into the host.



# Ethernet

Uses **CSMA/CD** distributed access scheme

- **CS: Carrier Sense**

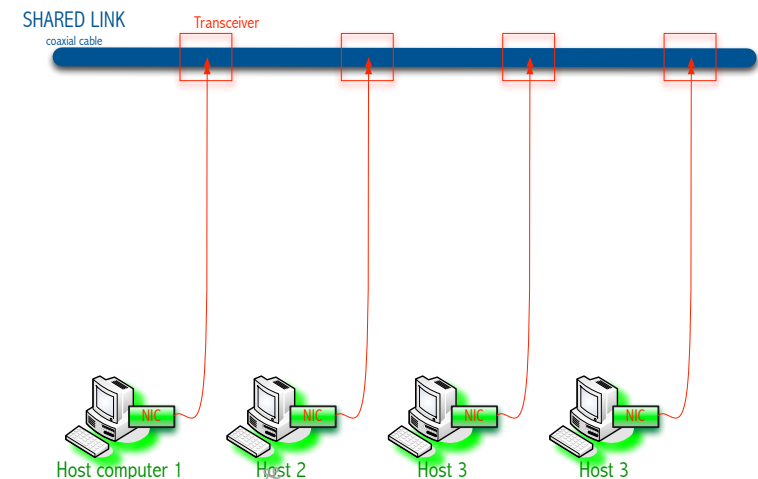
A computer can distinguish when the link is being used and when it is not (Idle)

- **MA: Multiple Access**

The link is shared among all the computers connected to it

- **CD: Collision Detection**

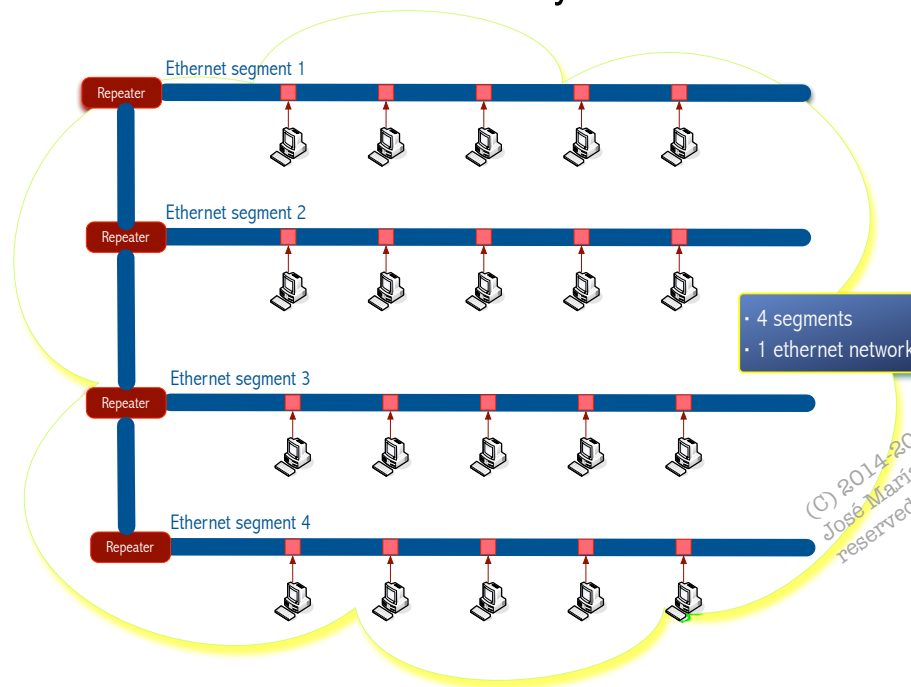
A transceiver ALWAYS listens on the medium as it transmits, therefore it can detect when its transmission is colliding with another frame being transmitted by another node



# Ethernet

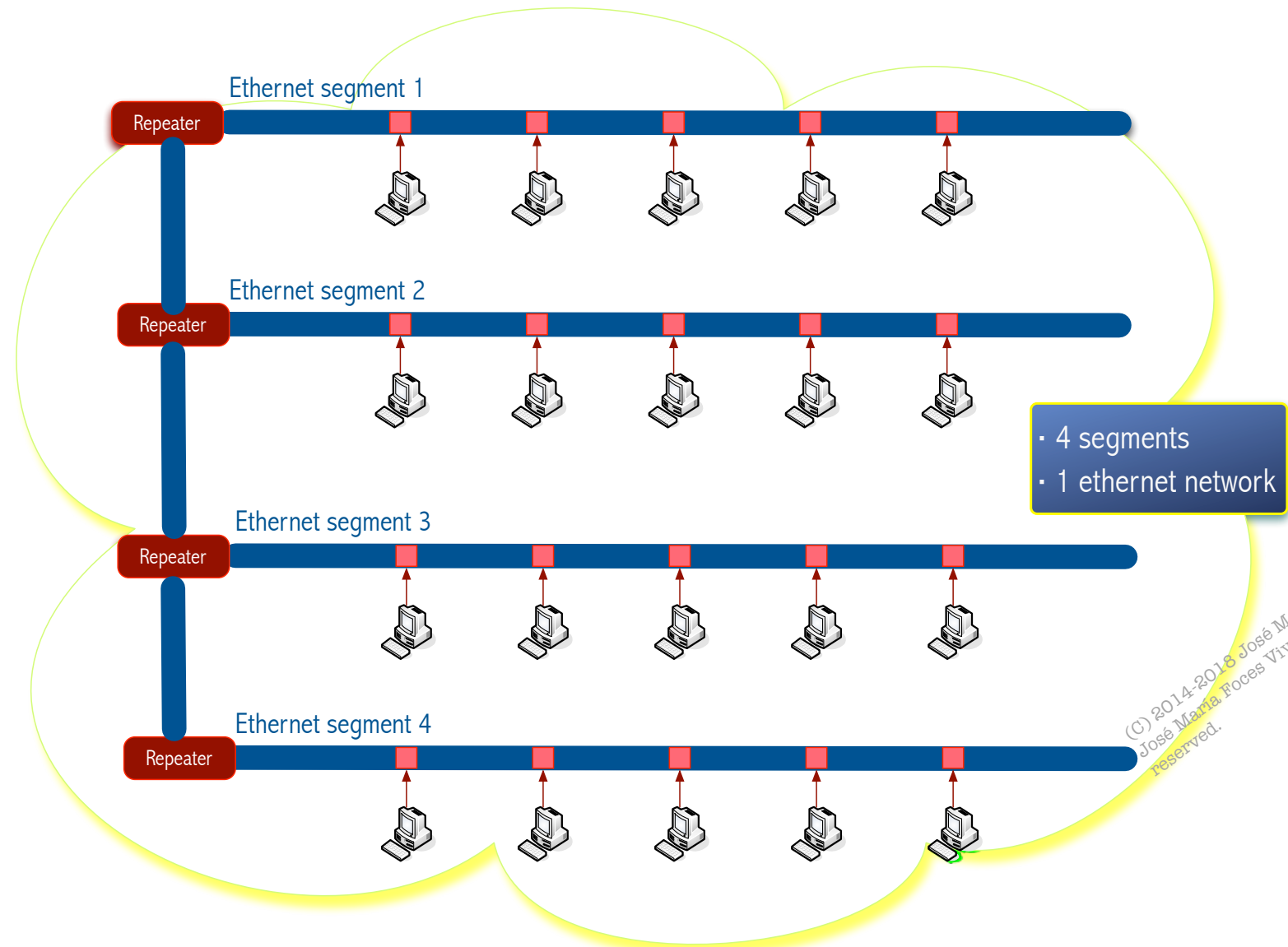
Multiple Ethernet **segments** can be joined together by *repeaters*.

- A *repeater* is a device that *forwards digital signals*.
- No more than four repeaters may be positioned between any pair of hosts.
- ▣ An Ethernet can have a max distance of only **2500 m**, **RTT = 51,2  $\mu$ s**





# Ethernet

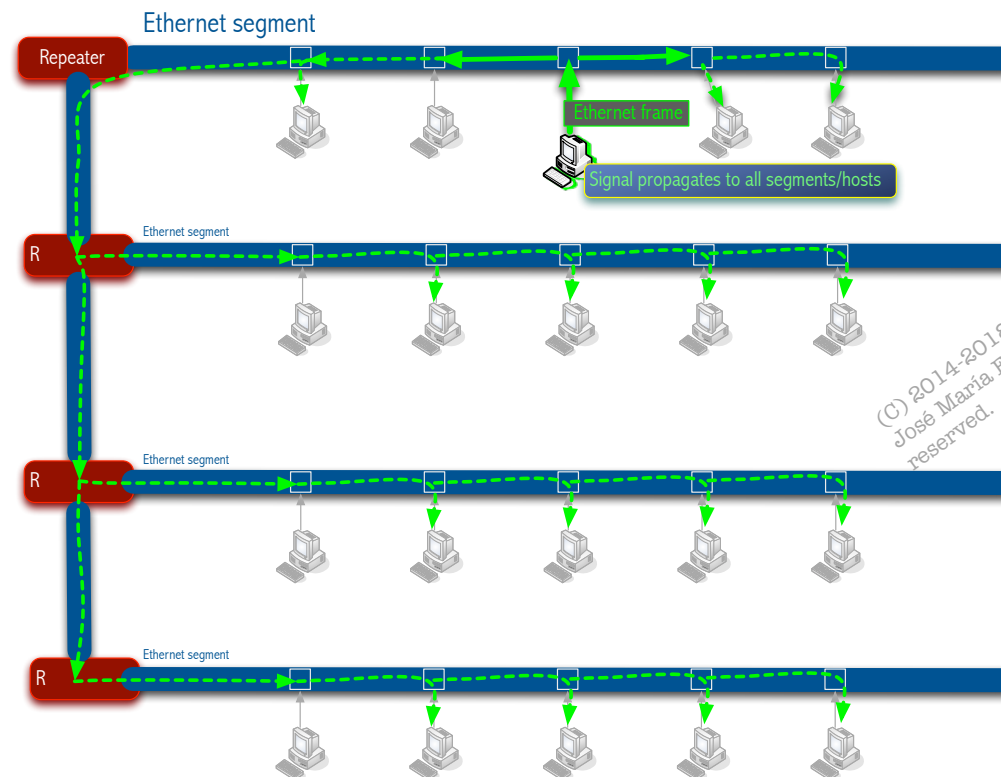


(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.

# Ethernet

Any **signal** placed on the Ethernet by a host is **broadcast** over the **entire network**

- ▣ Signal propagates in *both directions*
- ▣ **Repeaters** forward the signal **on all outgoing segments**
- ▣ Ethernet uses **Manchester** encoding scheme



(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.

# Other original Ethernet technologies

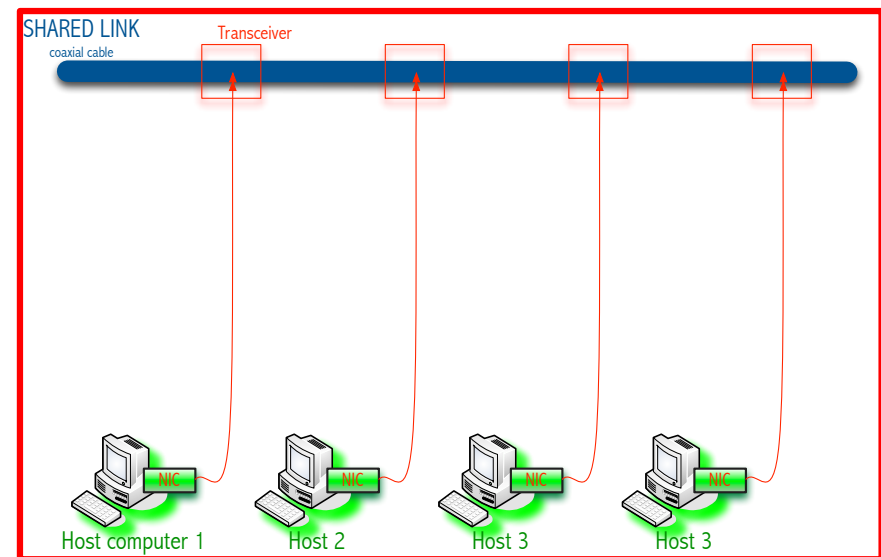
- Instead of using the thick coax cable, an Ethernet can be constructed from a **thinner cable**:

- ▣ **10Base2**

- ▣ The original was **10Base5**

- **10** means the network operates at *10 Mbps*
- **Base** means signals are pulses (Baseband transmission), no modulation (Broadband), encoding only
- **2** means that a given *segment* can be no longer than *200 m*
- **5** means that a given *segment* can be no longer than *500 m*

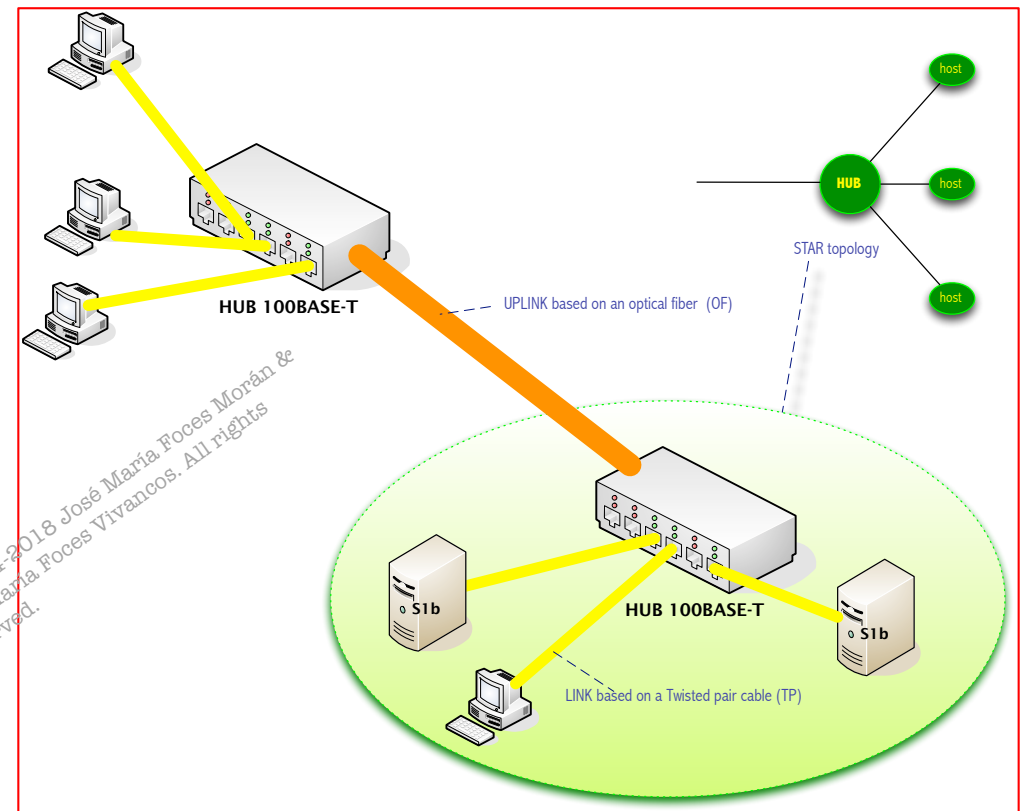
- Network topology in 10BASE5 and 10BASE2 is **Bus Topology**



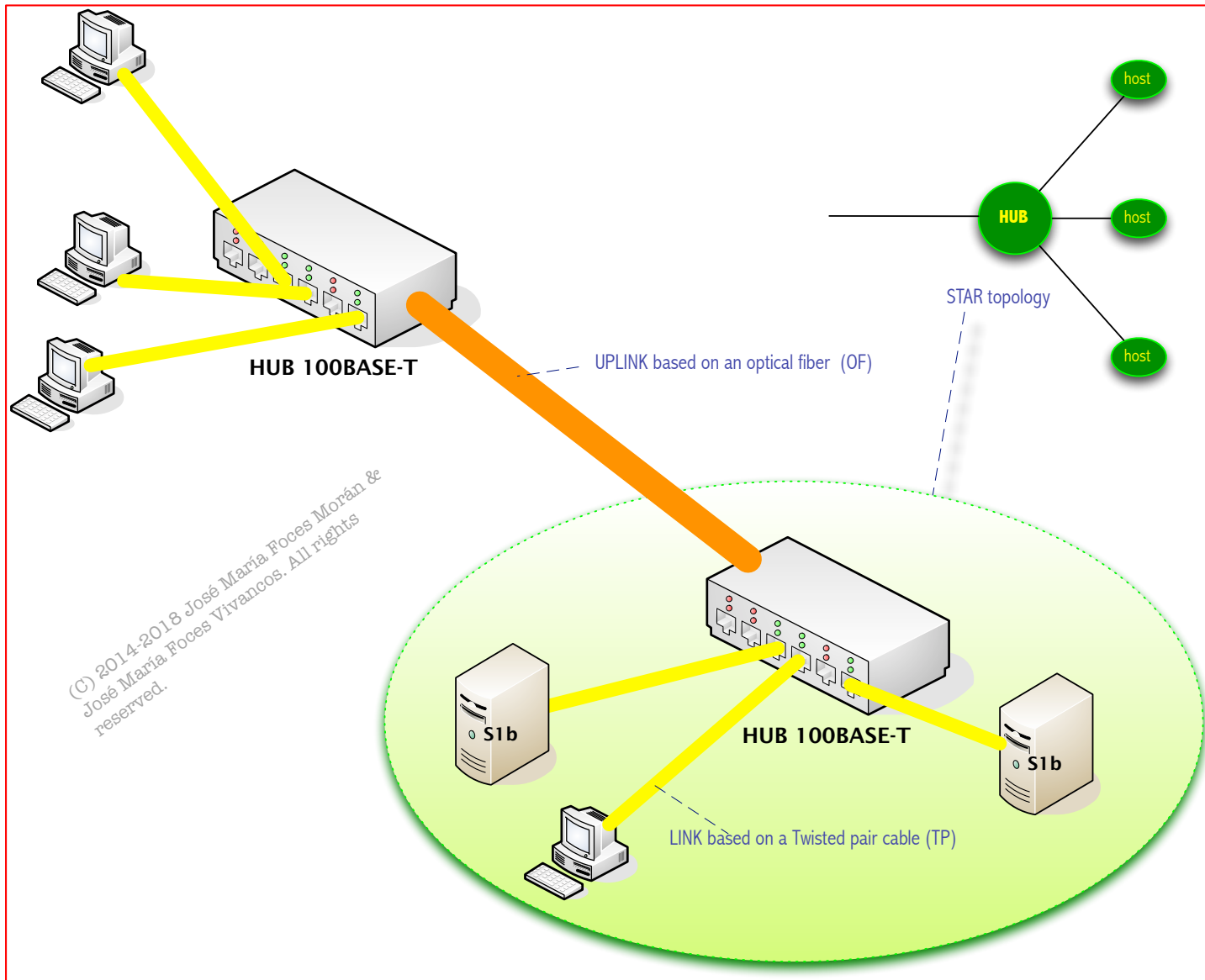
# More modern Ethernet technologies

- Another cable technology is **10BaseT**
  - ▣ T stands for **twisted** pair
  - ▣ Limited to **100 m** in length
- With 10BaseT, the common configuration is to have several point to point segments coming out of a *multiway repeater*, called *Hub*
  - ▣ *Concentrator*
  - ▣ *10 Mbps*
  - ▣ *Baseband transmission*
  - ▣ *T = Twisted pair cables*

- Network topology in 10BASET is **Star Topology**



# 10-BASE-T Hubs extending an Ethernet

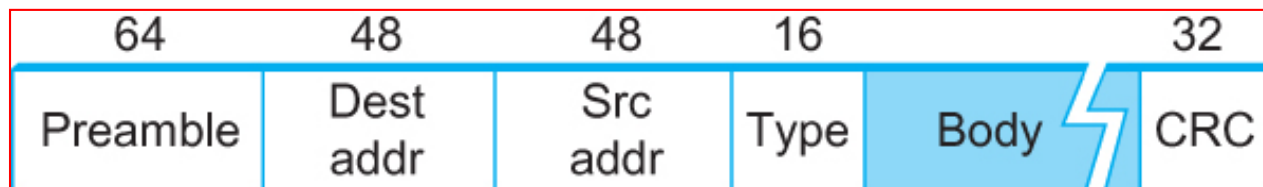


# Access Protocol for Ethernet

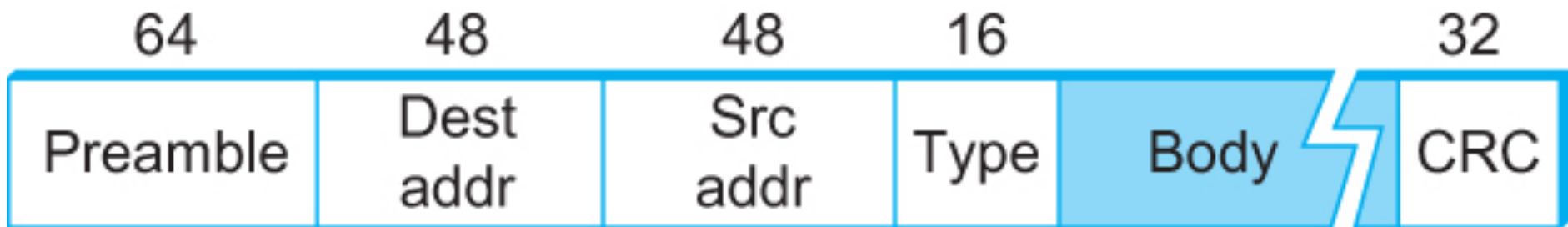
- Access protocol is called Ethernet's Media Access Control (MAC)
  - ▣ Remains CSMA/CD
  - ▣ It is implemented in Hardware on the **network adaptor**

- **Frame format**

- ▣ Preamble (64bit): allows the receiver to synchronize with the signal (sequence of alternating 64 0s and 1s ending in 11)
- ▣ Host and Destination Address (48bit each).
- ▣ Packet type (16bit): acts as demux key to identify the higher level protocol.
- ▣ Data (up to 1500 bytes)
  - ▣ Minimally a frame must contain at least 46 bytes of data (Padding if necessary)
  - ▣ Frame must be long enough to allow collision detection
- ▣ CRC (32bit)



# Ethernet Frame




Ethernet Frame Format

# Ethernet Addresses

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




# Ethernet Addresses

- 
- To ensure that every adaptor gets a **unique address**, each manufacturer of Ethernet devices is allocated a different prefix that must be prepended to the address on every adaptor they build
    - AMD has been assigned the 24bit prefix 8:0:20

# Ethernet Addresses

- Each frame transmitted on an Ethernet is received by every adaptor connected to that Ethernet
- *Each adaptor recognizes those frames addressed to its own address and passes only those frames on to the host*
- In addition to *unicast* address, an Ethernet address consisting of *all 1s* is treated as a *broadcast* address.
  - ▣ All adaptors pass frames addressed to the *broadcast* address up to the host
- Similarly, an address that has the first bit set to 1 but is not the *broadcast* address is called a *multicast* address
  - ▣ A given host can program its adaptor to accept some set of *multicast* addresses

# Ethernet Addresses

- 
- To summarize, an Ethernet adaptor *receives all frames*
  
  - *But, accepts only:*
    - ▣ Frames addressed to its own MAC address
    - ▣ Frames addressed to the broadcast MAC address
    - ▣ Frames addressed to a multicast address if it has been instructed

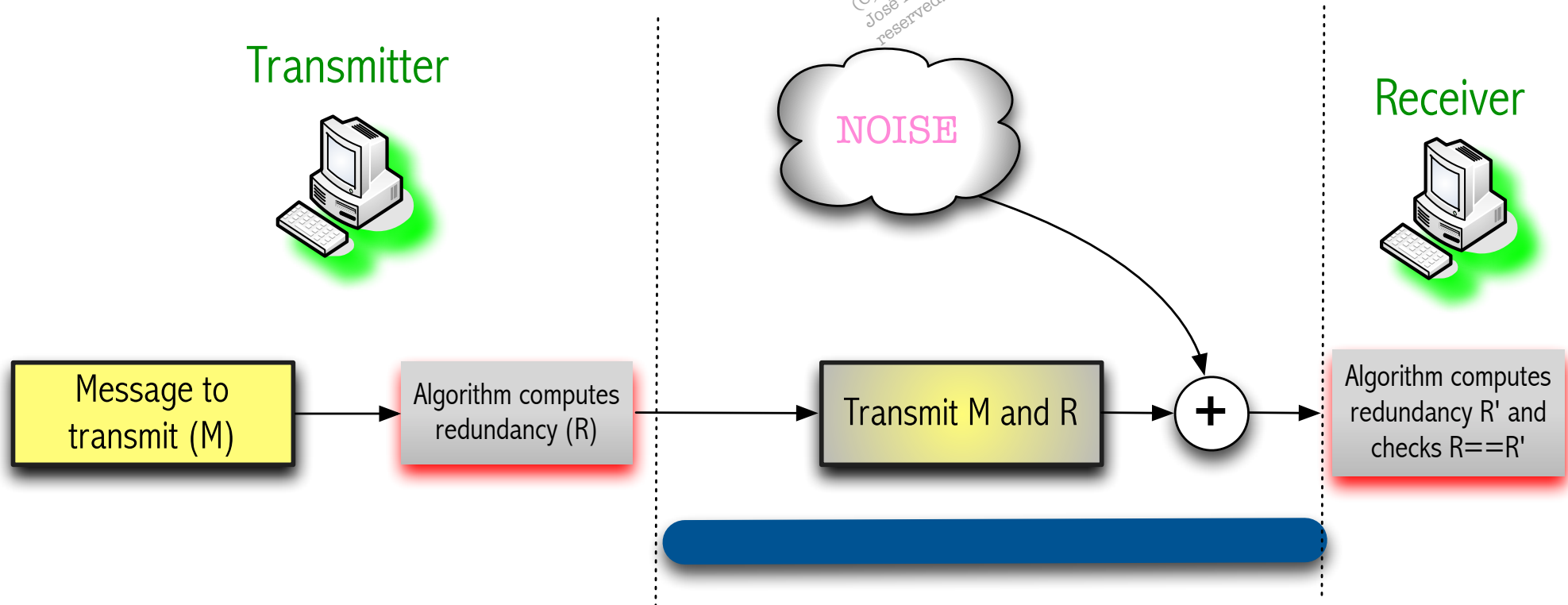


# Error control

# Error Control

- Bit errors are introduced into frames
  - ▣ Electrical interference and thermal noise

(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.



# Error Control

- - Error detection
  - Error correction
  
- If recipient detects an error, two options:
  1. Notify the sender
    - Retransmit the frame
    - If the probability of error is limited, the frame will be delivered without errors
  2. Receiver reconstructs message by using an error correcting code

# Error Detection



- Common techniques

- ▣ Other approaches

- Two Dimensional Parity (BISYNC)
    - Internet Checksum (IP)

- ▣ CRC (Cyclic Redundancy Check)

- Used in HDLC, DDCMP, CSMA/CD, Token Ring

# Error Detection



- Basic Idea of Error Detection

- To **add redundant information (REDUNDANCY)** to a frame that can be used to determine if errors have been introduced

- Naïve approach: Transmit two complete copies of the data?

- Identical → No error
  - WRONG!
- Unequal → Error
- Extremely inefficient
  - n bit message, n bit redundant information

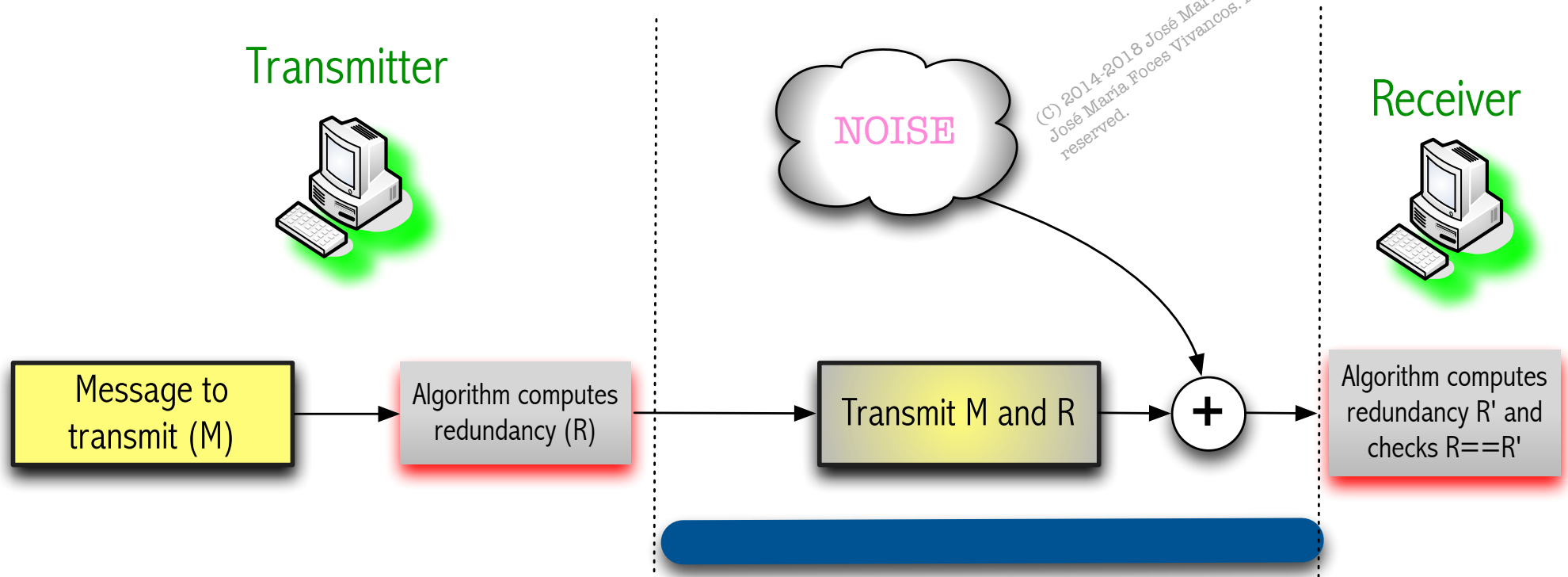
- Use strong error detection, instead

- k redundant bits, n bits message,  $k \ll n$
- Example: Ethernet CRC-32
  - frame can carry up to 12,000 bits of data
  - requires only 32 bits of redundancy




# Error Detection

- Extra bits are **redundant**
  - ▣ They add no new information to the message
  - ▣ Computed from the original message by applying *an algorithm* known by transmitter and receiver



# Horizontal parity

- 
- 1 bit of redundancy
    - ▣ Normally used for 7-bit data (ASCII characters, for example)
    - ▣ Add 1-bit parity, transmit the resulting 8 bits
      - Odd-parity sets the eighth bit to 1 if needed to give an odd number of 1s in the byte
        - 7-bit data: 0101010, Odd parity: 0, Send 01010100
        - 7-bit data: 0001010, Odd parity: 1, Send 00010101
      - Even parity sets the eighth bit to 1 if needed to give an even number of 1s in the byte
        - 7-bit data: 1111010, even parity: 1, Send 11110101
        - 7-bit data: 1111010, Odd parity: 0, Send 11110100

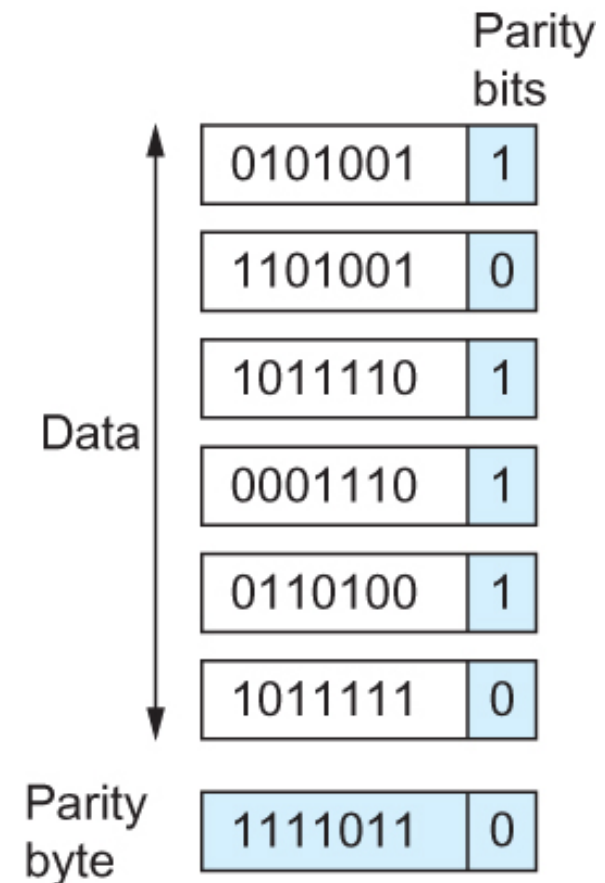
# Computing the even parity

Example:

- ▣ M = 7 bits of data
- ▣ R = 1 bit of redundancy (Even parity)
- ▣ Computation:
  - ▣ Parity bit = Data0 XOR Data1 XOR Data2 XOR Data 3 XOR Data4 XOR Data5 XOR Data6
  - ▣ DATA = 0110110
  - ▣ PARITY BIT = 0 xor 1 xor 1 xor 0 xor 1 xor 1 xor 0 = 0
  - ▣ Send 01101100

# Two-dimensional parity, an example


- Apply parity to each of the 7-bit bytes contained in the frame
- For every 6-BYTE block, compute its block-parity BYTE by computing the parity of each column of bits.
  - ▣ Two-dimensional parity catches all 1-, 2-, and 3-bit errors and most 4-bit errors
  - ▣ It's capable of correcting 1 error




# Internet Checksum Algorithm

- 
- Not used at the link level, used by UDP and TCP
- Transmitter
  - ▣ Add up all the words using 1-complement
  - ▣ Transmit the result (checksum) of that sum
- Receiver
  - ▣ Same calculation on the received data and compares the result with the received checksum
- If data | checksum gets corrupted
  - ▣ Results will not match
  - ▣ Receiver knows that an error occurred

# Internet Checksum Algorithm

- 
- Consider the data being checksummed as a sequence of 16-bit integers.
  - Add them together using 16-bit ones complement arithmetic (explained next slide) and then take the ones complement of the result.
  - That 16-bit number is the checksum

# Internet Checksum Algorithm

- 
- In ones complement arithmetic, a negative integer  $-x$  is represented as the complement of  $x$ ;
    - ▣ Each bit of  $x$  is inverted.
  - When adding numbers in ones complement arithmetic, a carryout from the most significant bit needs to be added to the result.

# Internet Checksum Algorithm

- - Consider, for example, the addition of  $-5$  and  $-3$  in ones complement arithmetic on 4-bit integers
    - $+5$  is  $0101$ , so  $-5$  is  $1010$ ;  $+3$  is  $0011$ , so  $-3$  is  $1100$
  - If we add  $1010$  and  $1100$  ignoring the carry, we get  $0110$
  - In ones complement arithmetic, the fact that this operation caused a carry from the most significant bit causes us to increment the result, giving  $0111$ , which is the ones complement representation of  $-8$  (obtained by inverting the bits in  $1000$ ), as we would expect



# Cyclic Redundancy Check (CRC)

- A few extra bits will **maximize** protection
- Given a (message), a bit string **110001** we can associate a polynomial on a single variable  $x$  for it
  - $M(x) = 1 \cdot x^5 + 1 \cdot x^4 + 0 \cdot x^3 + 0 \cdot x^2 + 0 \cdot x^1 + 1 \cdot x^0$   
 $= x^5 + x^4 + 1$
  - degree is 5, number of bits is 6
  - A  $k$ -bit frame has a maximum degree of  $k-1$
- Let  $M(x)$  be a message polynomial and
- Let  $C(x)$  be a generator polynomial

# Cyclic Redundancy Check (CRC)

- Assume  $M(x)/C(x)$  leaves a remainder of 0
- $M(x)$  is sent but, in fact,  $M'(x)$  is received
- The receiver will compute  $M'(x)/C(x)$ 
  - ▣ Remainder is not 0: **Error has been detected**
- Sender and receiver must use the same  $C(x)$ 
  - ▣ Examples: Ethernet (CRC-32)


# Cyclic Redundancy Check (CRC)




## □ Polynomial Arithmetic **Modulo 2**

- Any polynomial  $B(x)$  can be divided by a divisor polynomial  $C(x)$  if  $B(x)$  is of higher degree than  $C(x)$ .
- Any polynomial  $B(x)$  can be divided once by a divisor polynomial  $C(x)$  if  $B(x)$  is of the same degree as  $C(x)$ .
- The remainder obtained when  $B(x)$  is divided by  $C(x)$  is obtained by subtracting  $C(x)$  from  $B(x)$ .
- To subtract  $C(x)$  from  $B(x)$ , we simply perform the **exclusive-OR (XOR)** operation on each pair of matching coefficients.

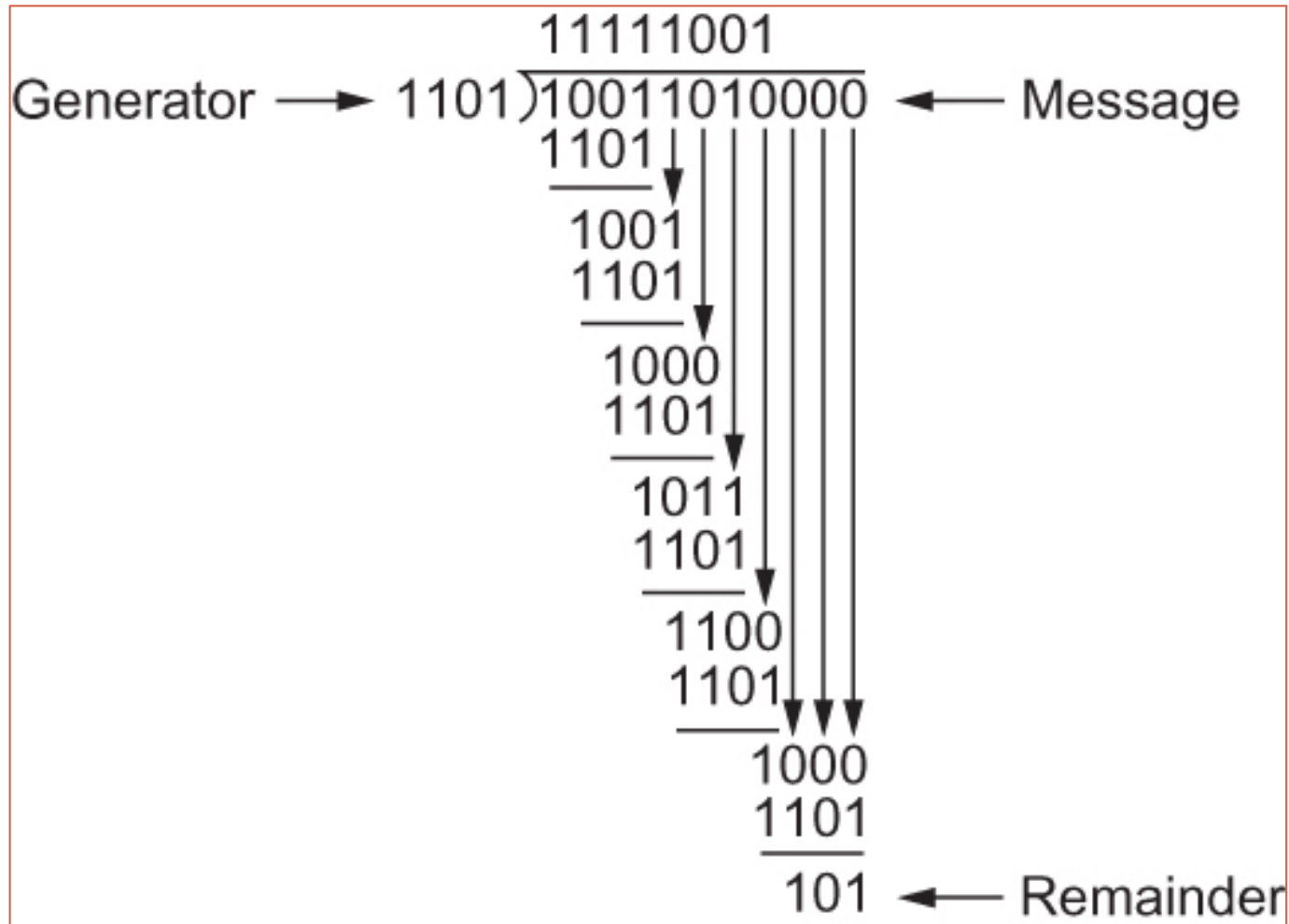
# Cyclic Redundancy Check (CRC)

- 
- Let  $M(x)$  be a frame with  $m$  bits and let the generator polynomial have less than  $m$  bits say equal to  $r$ .
  - Let  $r$  be the degree of  $C(x)$ . Append  $r$  zero bits to the low-order end of the frame, so it now contains  $m+r$  bits and corresponds to the polynomial  $x^r M(x)$ .

# Cyclic Redundancy Check (CRC)

- 
- Divide the bit string corresponding to  $xrM(x)$  by the bit string corresponding to  $C(x)$  using modulo 2 division.
  - Subtract the remainder (which is always  $r$  or fewer bits) from the string corresponding to  $xrM(x)$  using modulo 2 subtraction (addition and subtraction are the same in modulo 2).
  - The result is the checksummed frame to be transmitted. Call it polynomial  $M'(x)$ .

# Cyclic Redundancy Check (CRC)



CRC Calculation using Polynomial Long Division

# Cyclic Redundancy Check (CRC)



## □ Properties of Generator Polynomial

- ▣ Let  $P(x)$  represent what the sender sent and  $P(x) + E(x)$  is the received string. A 1 in  $E(x)$  represents that in the corresponding position in  $P(x)$  the message the bit is flipped.
- ▣ We know that  $P(x)/C(x)$  leaves a remainder of 0, but if  $E(x)/C(x)$  leaves a remainder of 0, then either  $E(x) = 0$  or  $C(x)$  is factor of  $E(x)$ .
- ▣ When  $C(x)$  is a factor of  $E(x)$  we have problem; errors go unnoticed.
- ▣ If there is a single bit error then  $E(x) = x^i$ , where  $i$  determines the bit in error. If  $C(x)$  contains two or more terms it will never divide  $E(x)$ , **so all single bit errors will be detected.**

# Cyclic Redundancy Check (CRC)

## □ **Properties** of Generator Polynomial

- ▣ In general, it is possible to prove that the following types of errors can be detected by a  $C(x)$  with the stated properties
  - **All single-bit errors**, as long as the  $x^k$  and  $x^0$  terms have nonzero coefficients.
  - **All double-bit errors**, as long as  $C(x)$  has a factor with at least three terms.
  - **Any odd number of errors**, as long as  $C(x)$  contains the factor  $(x+1)$ .
  - **Any “burst” error** (i.e., sequence of consecutive error bits) for which the length of the burst is less than  $k$  bits. (Most burst errors of larger than  $k$  bits can also be detected.)



# Cyclic Redundancy Check (CRC)

□ Six generator polynomials that have become international standards are:

□ CRC-8 =  $x^8+x^2+x+1$

□ CRC-10 =  $x^{10}+x^9+x^5+x^4+x+1$

□ CRC-12 =  $x^{12}+x^{11}+x^3+x^2+x+1$

□ CRC-16 =  $x^{16}+x^{15}+x^2+1$

□ CRC-CCITT =  $x^{16}+x^{12}+x^5+1$

□ CRC-32 =

$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

# Steps to compute the CRC by shift registers 1/4

(Example from pg 101/102 of P&D textbook)

## Complementary notes to CRC computing by shift registers

©2013, José María Foces Morán

1. Generator polynomial of CRC example in pg. 101 and 102 of P&D textbook.

$$C(x) = 1 + x^2 + x^3 = 1 \cdot x^0 + 0 \cdot x^1 + 1 \cdot x^2 + 1 \cdot x^3$$

2. One-bit shift registers

The order of  $C(x)$  is  $k = 3$ , the circuit will have  $k$  registers, each assigned to powers of  $x$  0 through  $k-1$ , in this case powers are: 0, 1 and 2 and the number of shift registers is 3

0

1

2

© 2014-2018 José María Foces Morán & José María Foces Vivanco. All rights reserved.

# Computing CRC by shift registers 2/4

(Example from pg 101/102 of P&D textbook)

### 3. XOR gate at the input of each register that does belong in $C(x)$

Since the term 1 ( $0 \times \text{power}$ ) does belong in  $C(x)$ , we add an XOR gate at the input of register 0



Since the term  $x^1$  ( $1 \times \text{power}$ ) does NOT belong in  $C(x)$ , we connect its input directly to the output of its former shift register ( 0 )



Since the term  $x^2$  ( $2 \times \text{power}$ ) does belong in  $C(x)$ , we add an XOR gate at the input of register 2

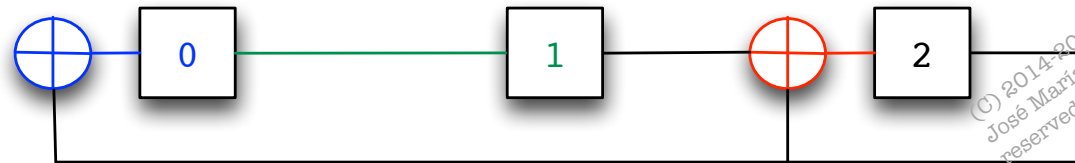


(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.

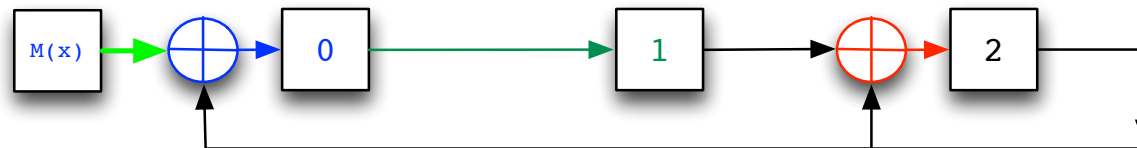
# Computing CRC by shift registers 3/4

(Example from pg 101/102 of P&D textbook)

- 4. Connect the output of the Most Significant Register (2) to all the XOR gates



- 5. Add one more shift register which will hold *each present value* from the message polynomial  $M(x)$ . The arrows represent the information flow of the circuit:



(C) 2014-2018 José María Foces Morán & José María Foces Vivanco. All rights reserved.

# Computing CRC by shift registers 4/4

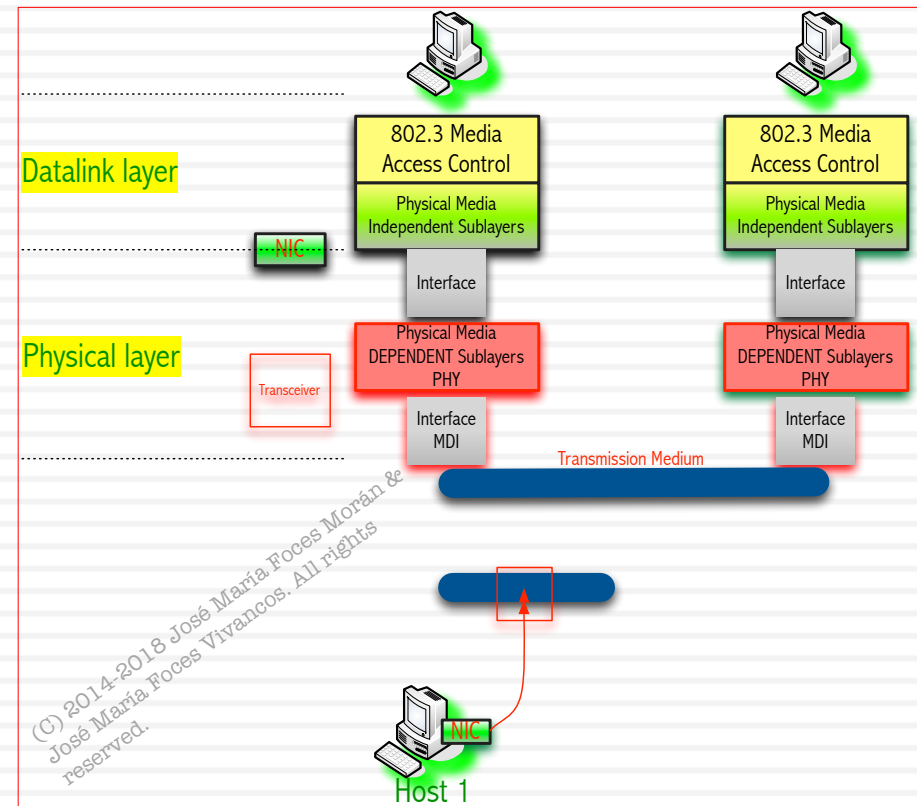
(Example from pg 101/102 of P&D textbook)

6. The CRC circuit is ready for computing the CRC. This type of circuit computes its next state by using its present state. Initially, the present state of the circuit will be represented by a 0 in each register, except for  $M(x)$  which will contain the bits from the message polynomial, starting with its Most Significant Bit – make sure you add k 0's to the least significant bits of  $M(x)$ , in this case 3. We will represent this computation in tabular form. We proceed by starting with the first row (initial state) and compute the next  $x^0$  bit (next row) according to the circuit and do the same for  $x^1$  and  $x^2$ . When a new row has been computed, we have a new present state and proceed by computing the next state(next row) using the explained procedure. The algorithm finishes when there remain no more bits in  $M(x)$ , at that state, bits  $x^0$ ,  $x^1$ ,  $x^2$  contain the **CRC**.

$M(x)$	$x^0$	$x^1$	$x^2$
1	0	0	0
0	1	0	0
0	0	1	0
1	0	0	1
1	0	0	1
0	0	0	1
1	1	0	1
0	0	1	1
0	1	0	0
0	0	1	0
0	0	0	1
EMPTY.	<b>CRC: 1</b>	<b>0</b>	<b>1</b>

(C) 2014-2018 José María Foces Morán & José María Foces Vivanco. All rights reserved.

# CSMA/CD Ethernet



# Ethernet

Uses CSMA/CD MAC (Media Access Control)

- CS: Carrier Sense

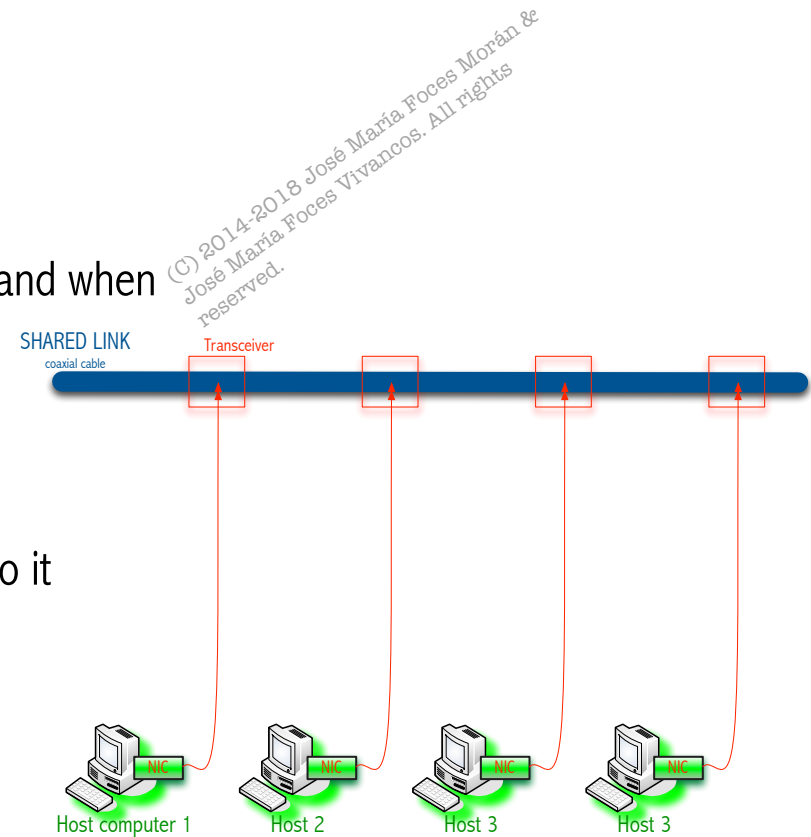
A computer can distinguish when the link is being used and when it is not

- MA: Multiple Access

The link is shared among all the computers connected to it

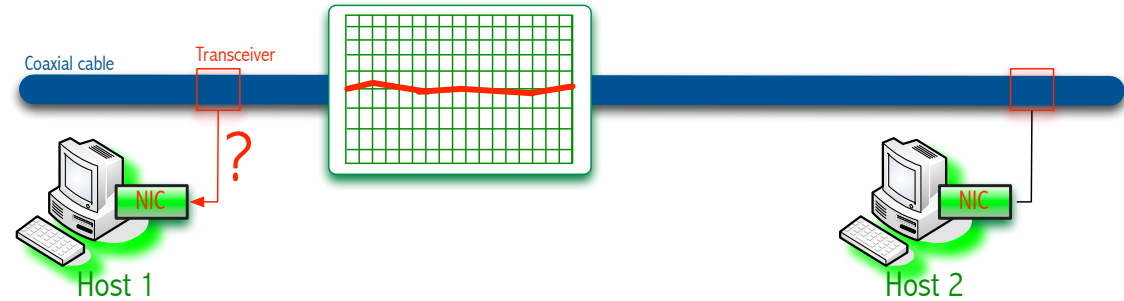
- CD: Collision Detection

A computer listens as it transmits, can detect when its transmission has collided with another frame being transmitted by another node

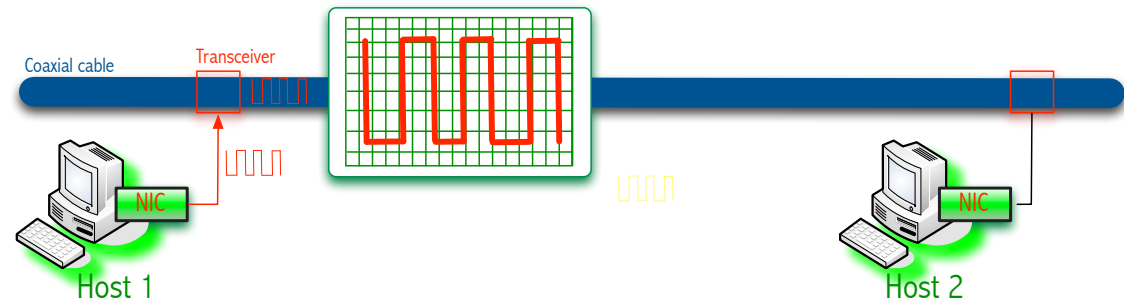


# Ethernet Transmitter Algorithm: No carrier

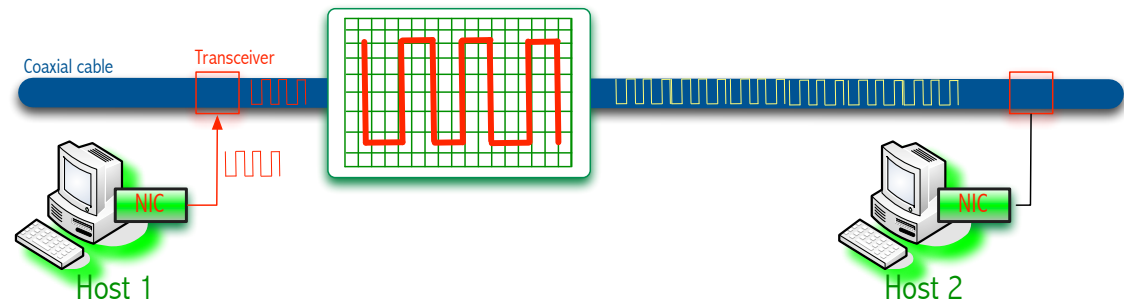
- NIC has a **new frame** to send
- Line is idle (**No carrier**)
  - ▣ Decides to transmit it *almost immediately*
  - ▣ Waits **IFG** seconds before starting transmission
  - ▣ IFG: Inter Frame Gap = 9,6  $\mu$ s



1. Host 1 senses the medium to establish whether or not CARRIER is present
2. Concludes that there is not CARRIER



3. Host 1 begins to transmit a new Ethernet frame

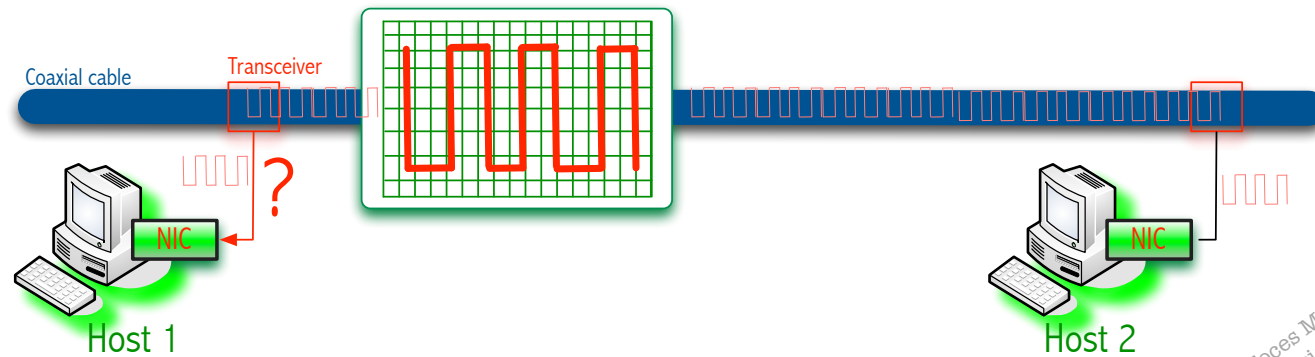


4. Host 1 continues transmitting the whole Ethernet frame, signal propagates



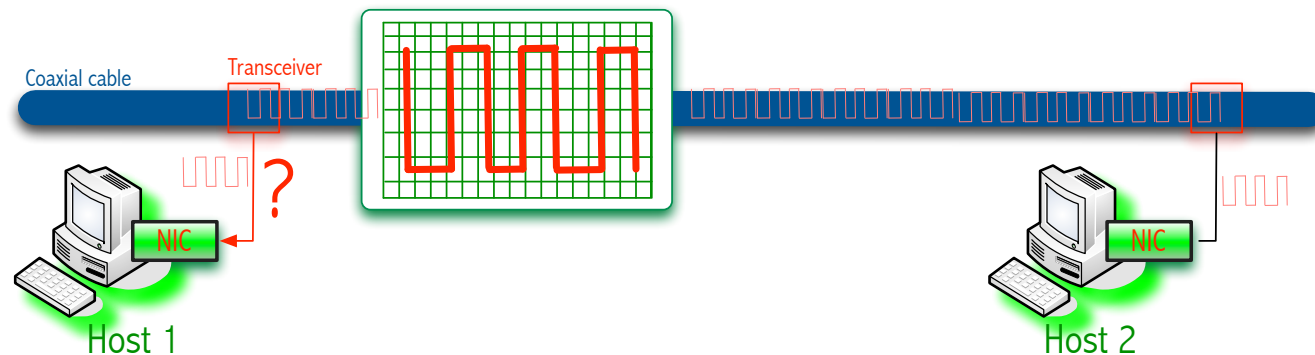
# Ethernet Transmitter Algorithm: Carrier found

- Ethernet is said to be **1-persistent protocol** because an adaptor with a frame to send attempts to transmit it with probability 1 after waiting an IFG time after it sees that the medium went idle



1. Host 2 is transmitting a frame and, so far, it propagated through host 1
2. Host 1 senses the medium to establish whether or not CARRIER is present
3. Concludes that there is CARRIER

(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.

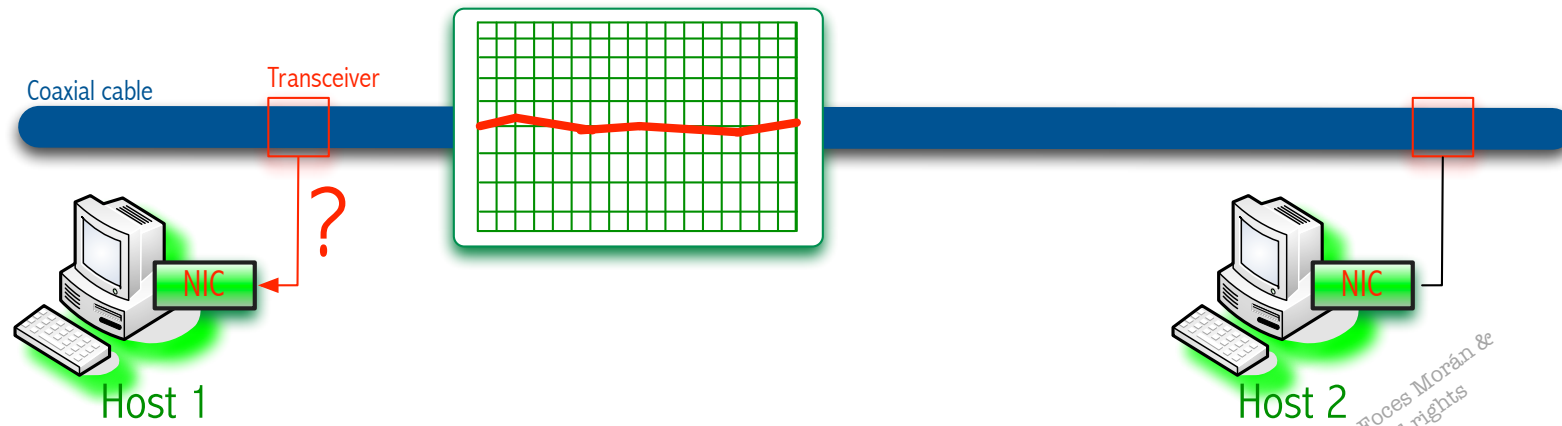


4. Host 1 defers transmission until the medium is idle (no carrier) + an **IFG time (InterFrame Gap = 9,6 μs)**

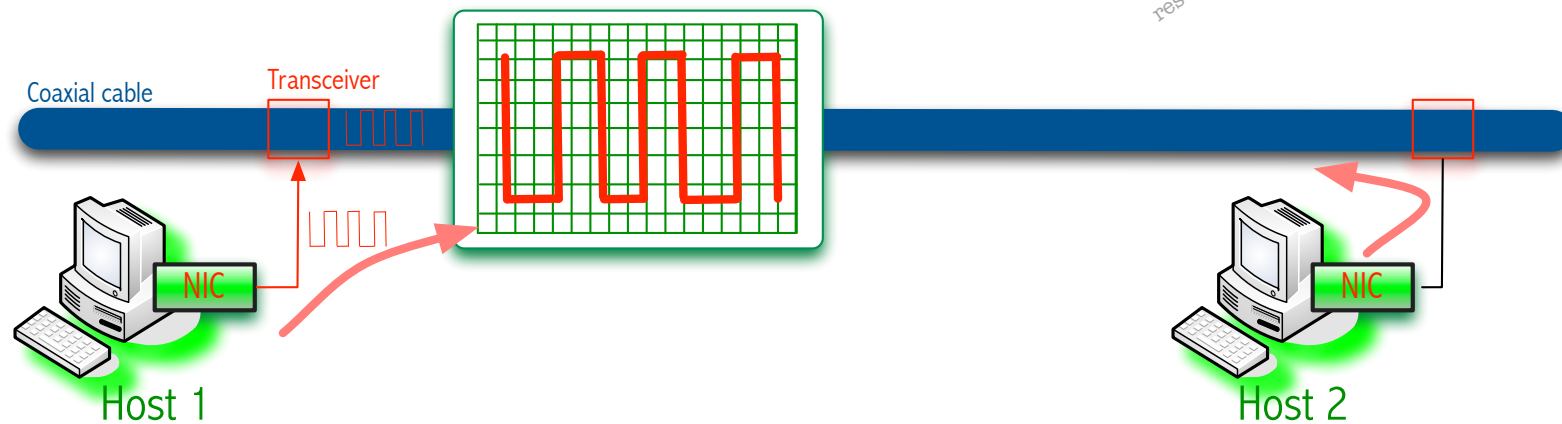
# Ethernet Transmitter Algorithm: Collisions

- Since there is **no centralized control** it is possible for two adaptors to
  - ▣ Begin transmitting at the same time, is it possible? Yes, it is!
    - Either because both found the line to be idle at their physical positions along the cable
    - Or, both had been waiting for a busy line to become idle after an IFG time (9,6 μs)
- When this happens, the two frames are said to ***collide*** on the network
  - ▣ **A collision has occurred**
  - ▣ ***Collisions may involve any number of nodes (>1, obviously)***

# Ethernet Transmitter Algorithm: Collisions



1. Host 1 senses the medium to establish whether or not CARRIER is present
2. Concludes that there is not CARRIER

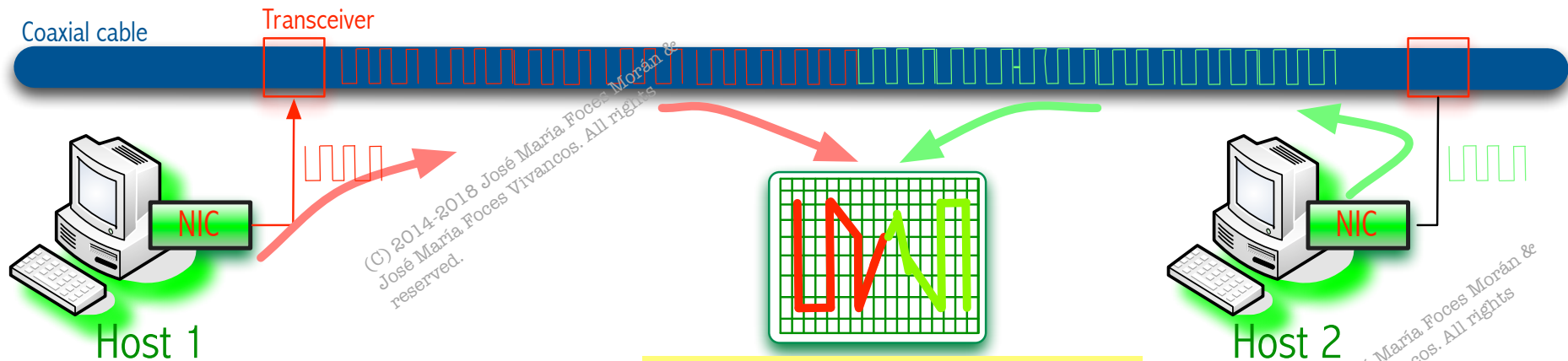


3. Host 1 begins to transmit a new Ethernet frame

4. Host 2 sees no CARRIER either and begins to transmit its new Ethernet frame

(C) 2014-2018 José María Foces Morán & José María Foces Vivanco. All rights reserved.

# Ethernet Transmitter Algorithm: Collisions



(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.

(C) 2014-2018 José María Foces Morán & José María Foces Vivancos. All rights reserved.

4. The two wavefronts collide and the collided signal begins to propagate to the right and to the left

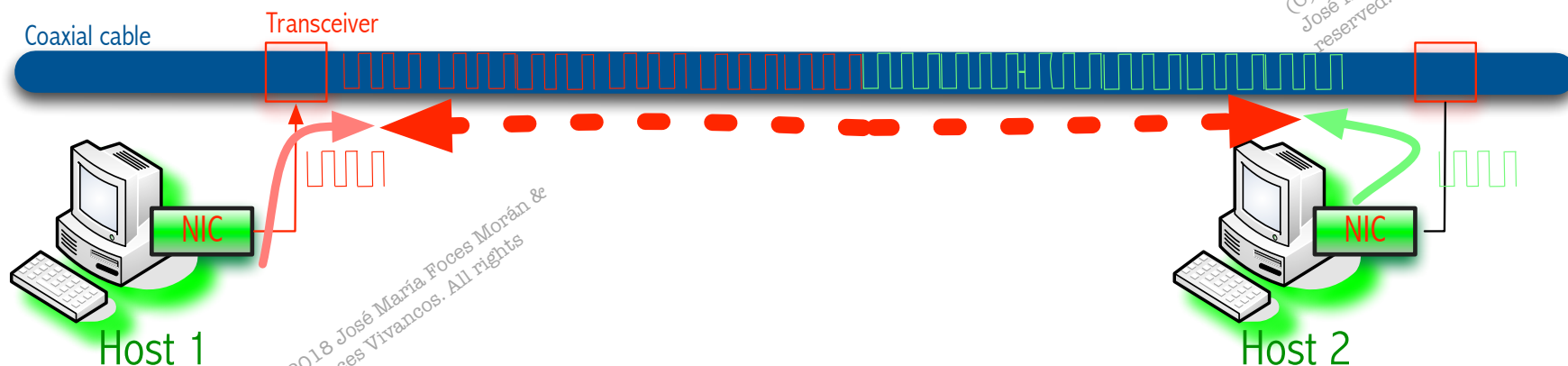
4'. Host 1 continues transmitting the whole Ethernet frame, bit by bit, each time it transmits a new bit, it reads it to make sure it is correct and to make sure that no collision has occurred. From host 1 standpoint, the collision has not occurred yet

4'. Host 2: same as Host 1, keeps transmitting unaware yet that a collision has occurred

5. Collided data signal keeps propagating as the hosts keep transmitting

# Ethernet Transmitter Algorithm: Collisions


- Since Ethernet supports collision detection (CD), each sender is able to determine that a collision is in progress.
- The moment an adaptor detects that its frame is colliding with another, it first makes sure to transmit a 32-bit jamming sequence and then stops transmission.
  - ▣ Thus, a transmitter will minimally send 96 bits in the case of collision
    - 64-bit preamble + 32-bit jamming sequence



6. When the collided signal arrives at host 2, for example, it interferes with the data transmitted and the host notices this because the data read is not coincident with the data written anymore.

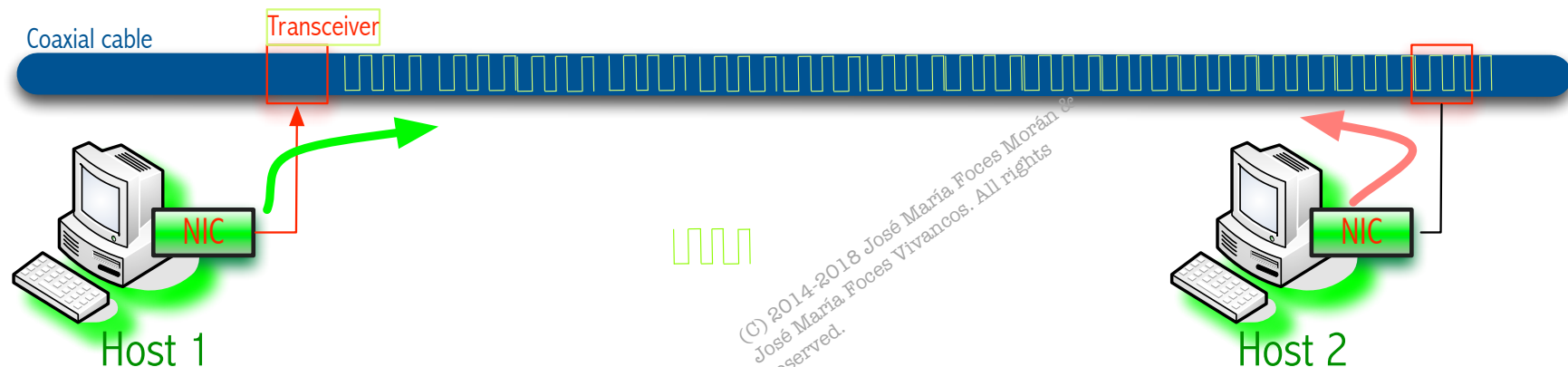
7. Host 2 sends a 32-bit jamming sequence so that the rest of nodes are informed that a collision occurred.

# Ethernet Transmitter Algorithm: Collisions

- 
- One way that an adaptor will send only 96 bit (called a *runt frame*) is if the two hosts are close to each other.
  - Had they been farther apart,
    - ▣ They would have had to transmit longer, and thus send more bits, before detecting the collision.

# Ethernet Transmitter Algorithm: Collisions

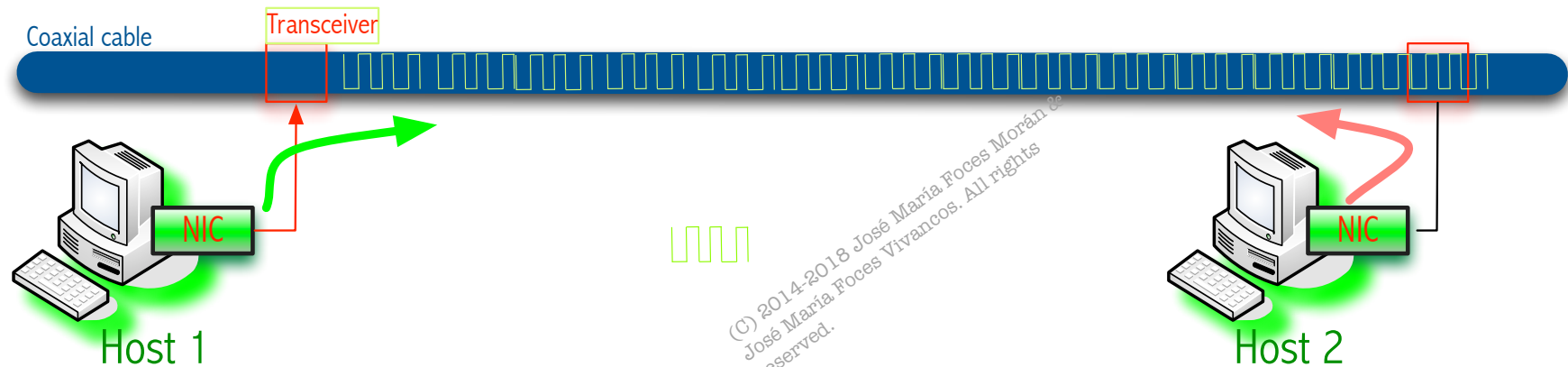
- The worst case scenario happens when the two hosts are at opposite ends of the Ethernet
- To know for sure that the frame it has just sent did not collide with another frame, the transmitter has to send at least 512 bits
  - ▣ CONCLUSION: Ethernet frames must be at least 512 bits (64 bytes) long
    - 14 bytes of header + 46 bytes of data + 4 bytes of CRC
    - If the application is sending less than 46 bytes of data, the transmitter will include padding 0's



Host 1 must transmit at least 512 bits to make sure that it itself detects a collision even with the host located at the farthest end of the network, otherwise it will miss that a collision occurred

# Ethernet Transmitter Algorithm: Collisions

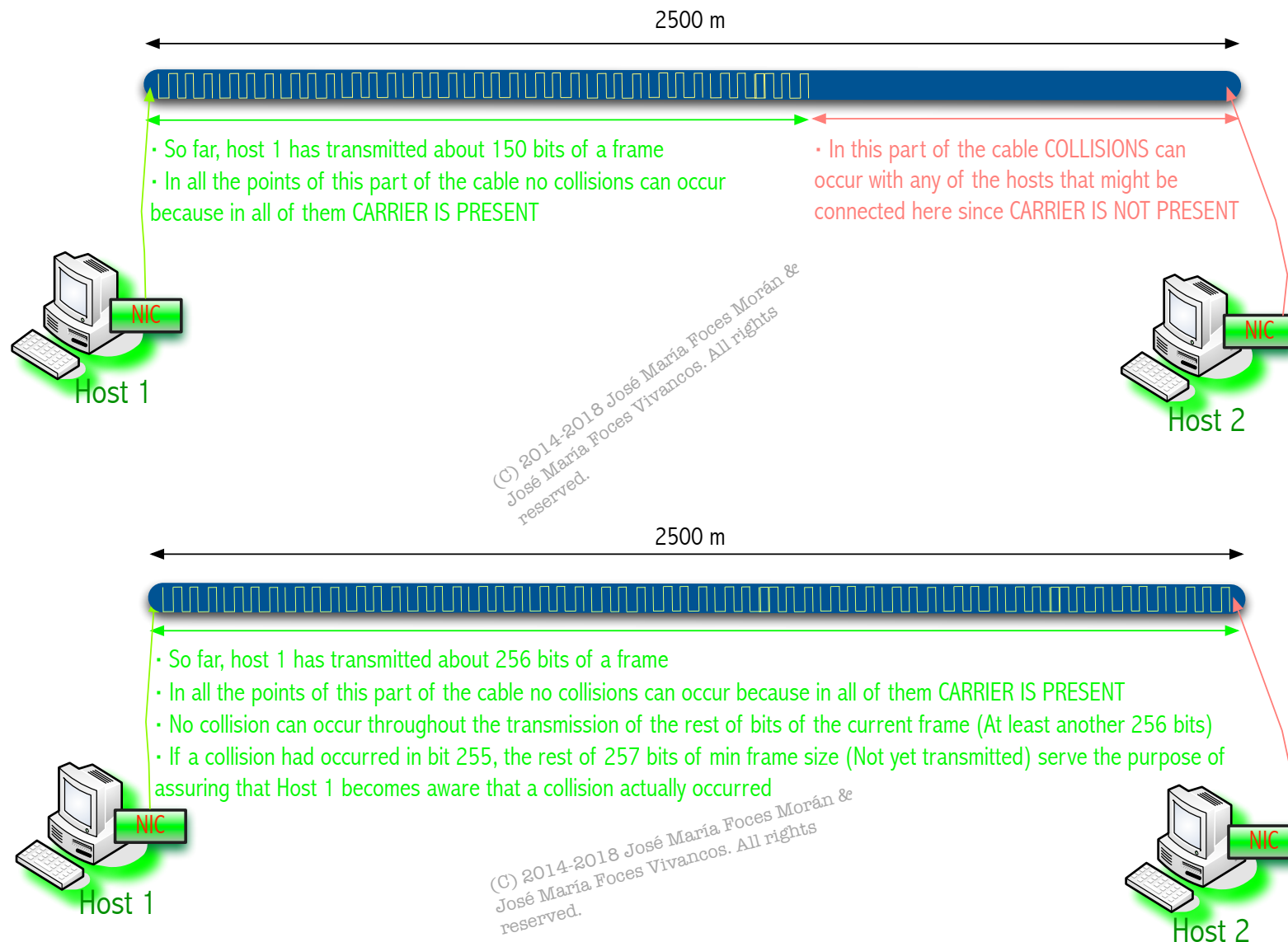
- Why is 512 bits Ethernet's minimum frame size?
  - ▣ Why is its maximum length limited to 2500 m?
  - ▣ Recall speed is 10Mbps and  $RTT=51,2 \mu s$ , therefore  $V_{prop} \cong 2/3 \cdot c$
- Only the first 256 bits (maximum) of the 512 are vulnerable to collisions with any other ethernet host connected to the cable
- In the worst case, the whole 512 bits are necessary for the transmitting host to become aware that a collision took place –with any of the first 256 bits



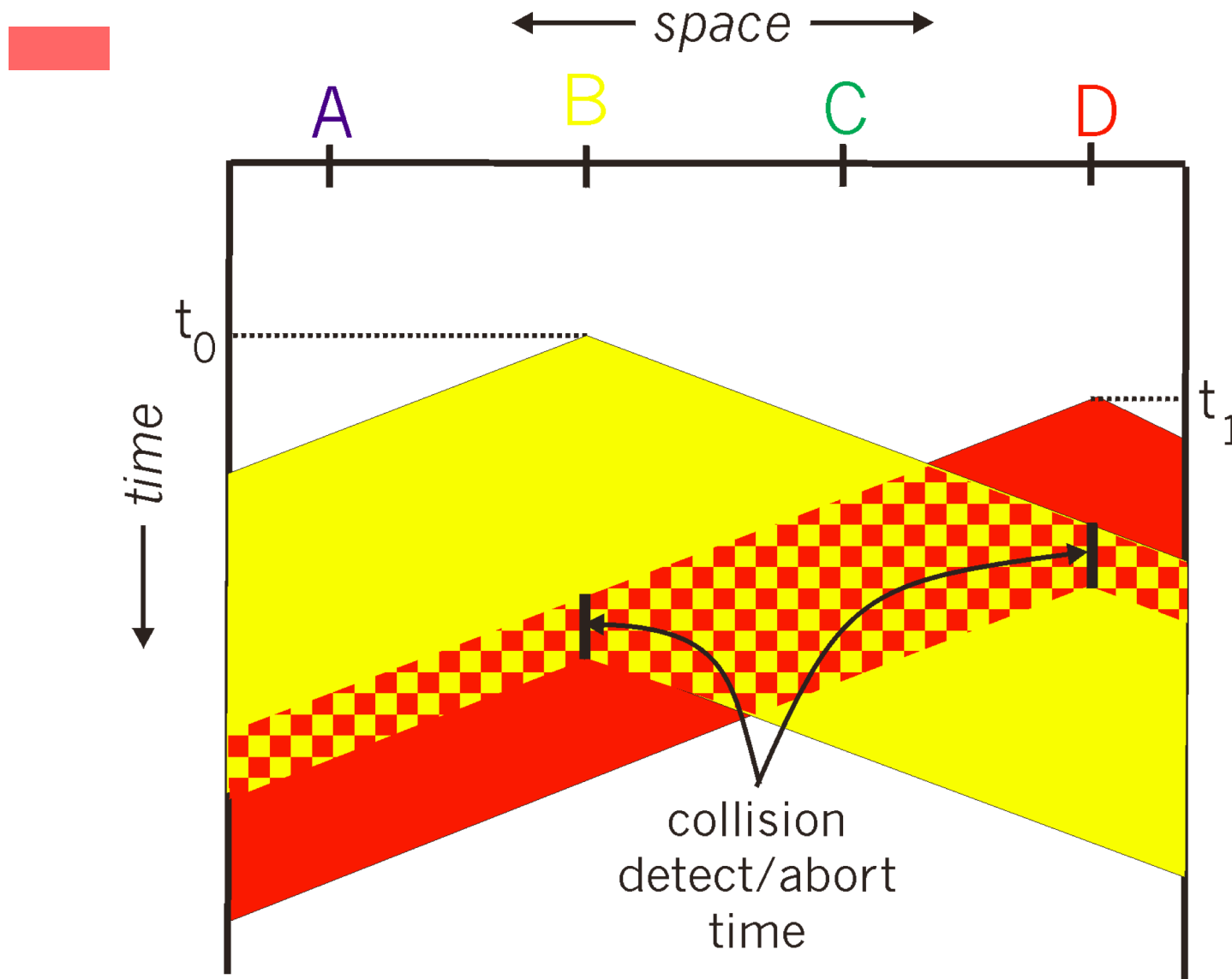
Host 1 must transmit at least 512 bits to make sure that it itself detects a collision even with the host located at the farthest end of the network, otherwise it will miss that a collision occurred



# Ethernet Transmitter Algorithm: Collisions

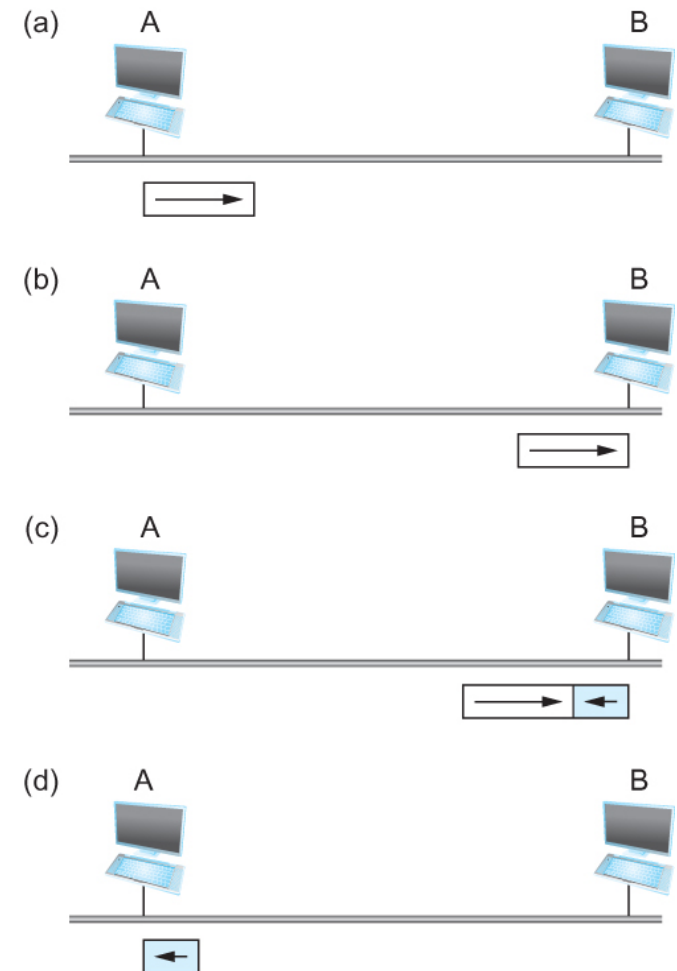


# Ethernet Transmitter Algorithm: Collisions



# Ethernet Transmitter Algorithm: Collisions

- A begins transmitting a frame at time  $t$
- $d$  denotes the one link latency.  $RTT = 51,2 \mu s$ ,  $d = RTT/2$
- The first bit of A's frame arrives at B at time  $t + d$
- Suppose an instant before host A's frame arrives, host B begins to transmit its own frame
- B's frame will immediately collide with A's frame and this collision will be detected by host B
- Host B will send the 32-bit jamming sequence
- Host A will not know that the collision occurred until B's frame reaches it, which will happen at  $t + 2 * d$
- Host A must continue to transmit until this time in order to detect the collision
  - ▣ Host A must transmit for  $2 * d = RTT$  to be sure that it detects all possible collisions



Worst-case scenario: (a) A sends a frame at time  $t$ ; (b) A's frame arrives at B at time  $t + d$ ; (c) B begins transmitting at time  $t + d$  and collides with A's frame; (d) B's runt (32-bit) frame arrives at A at time  $t + 2d$ .

# Ethernet Transmitter Algorithm

- 
- Consider that a maximally configured Ethernet is 2500 m long, and there may be up to four repeaters between any two hosts, the round trip delay has been determined to be  $R_{tt} = 51.2 \mu s$ 
  - ▣ Which, on 10 Mbps Ethernet, corresponds to 512 bits
- The other way to look at this situation,
  - ▣ We need to limit the Ethernet's maximum RTT to a fairly small value (51.2  $\mu s$ ) for the access algorithm to work fine
    - Hence the maximum length for the Ethernet is on the order of 2500 m.

# Ethernet Transmitter Algorithm

- Once an adaptor has detected a **collision**, and stopped its transmission, it **waits a certain amount of time** and tries again
- Each time the adaptor tries to transmit but fails, it doubles the amount of time it waits before trying again
- This strategy of doubling the delay interval between each retransmission attempt is known as ***Exponential Backoff***

# Exponential Backoff: Defer next transmission attempt after a collision

- When two hosts have collided, how can we avoid further collisions to occur which involve those two hosts?
- We'll have both hosts choose a random number to determine how much time they will defer the new transmission attempt, but, what algorithm can be applied? Ethernet uses *Exponential Backoff Algorithm*
- Assume  $k$  is the number of collisions undergone by a host adapter when attempting to transmit a specific frame
  - ▣  $k = 1$  The *adaptor flips 1 coin* and gets heads (0) or tails (1) = {0, 1}. Name the result  $r$ , then:
    - Time to defer transmission =  $r * 51,2 \mu\text{s}$
    - 0 -> 0  $\mu\text{s}$
    - 1 -> 51,2  $\mu\text{s}$
  - ▣  $k = 2$  The *adaptor flips 2 coins* and gets one of {00, 01, 10, 11} = {0, 1, 2, 3}
    - Time to defer transmission =  $r * 51,2 \mu\text{s}$
    - 0 -> 0  $\mu\text{s}$
    - 1 -> 1 x 51,2  $\mu\text{s}$
    - 2 -> 2 x 51,2  $\mu\text{s}$
    - 3 -> 3 x 51,2  $\mu\text{s}$
  - ▣ In general, the transmitter will flip  $k$  coins, thereby obtaining one of {0, 1, 2, ...  $2^{k-1}$ } and calculating the time to defer next transmission attempt as  $r * 51,2 \mu\text{s}$
- Flipping the coin, in a computer system, consists of generating random numbers

# Experience with Ethernet


- Ethernets work best under lightly loaded conditions.
  - ▣ Under heavy loads, too much of the network's capacity is wasted by collisions.
- Most Ethernets are used in a conservative way.
  - ▣ Have fewer than 200 hosts connected to them which is far fewer than the maximum of 1024.
- Most Ethernets are far shorter than 2500m with a round-trip delay of closer to 5  $\mu$ s than 51.2  $\mu$ s.
- Ethernets are easy to administer and maintain.
  - ▣ There are no switches that can fail and no routing and configuration tables that have to be kept up-to-date.
  - ▣ It is easy to add a new host to the network.
  - ▣ It is inexpensive.
    - Cable is cheap, and only other cost is the network adaptor on each host.




# Reliable transmission



# Reliable Transmission

- 
- CRC is used to detect errors
    - Corrupt frames must be **discarded**
  - What is a **Reliable Link Protocol**?
    - Guaranteed delivery
    - Integrity
    - In order, etc
    - From **point-to-point**
  - Two fundamental **mechanisms**
    - **Acknowledgements**
    - **Timeouts**

# Reliable Transmission

- 
- An *acknowledgement* (**ACK**)
    - A small *control frame* that a protocol sends back to its peer saying that it has received the earlier frame
    - A *control frame* is a frame with header only (*no data*)
  - The receipt of an *acknowledgement* indicates to the sender of the original frame that its frame was *successfully delivered*

# Reliable Transmission

- If the sender does **not** receive an **acknowledgment** after a reasonable amount of **time**, then it **retransmits** the original frame
- The action of waiting a reasonable amount of time is called a **timeout**.

- Automatic Repeat reQuest (ARQ)
  - Acknowledgements
  - Timeouts

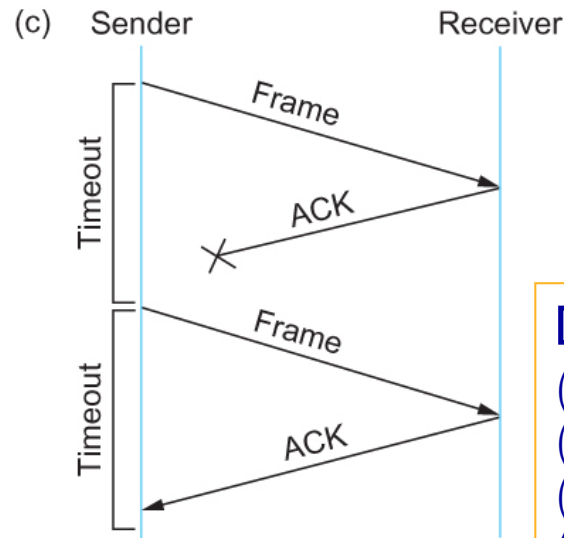
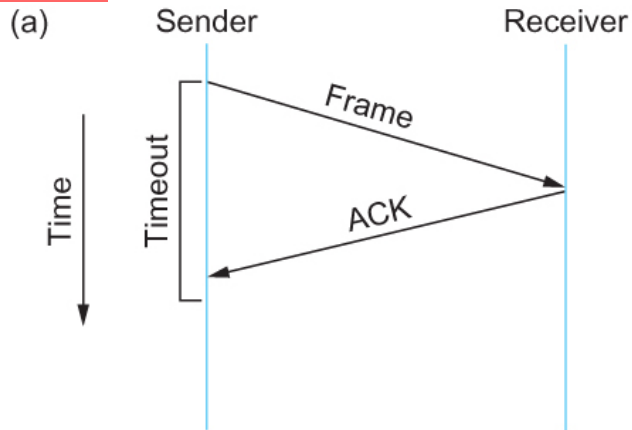
# Stop and Wait Protocol



This abstract protocol uses ARQ

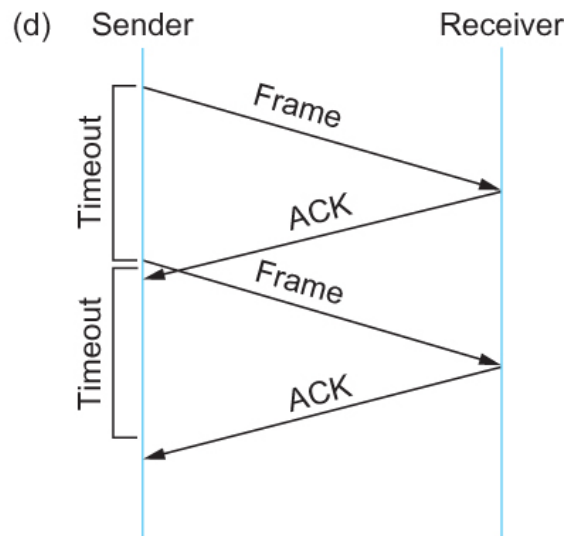
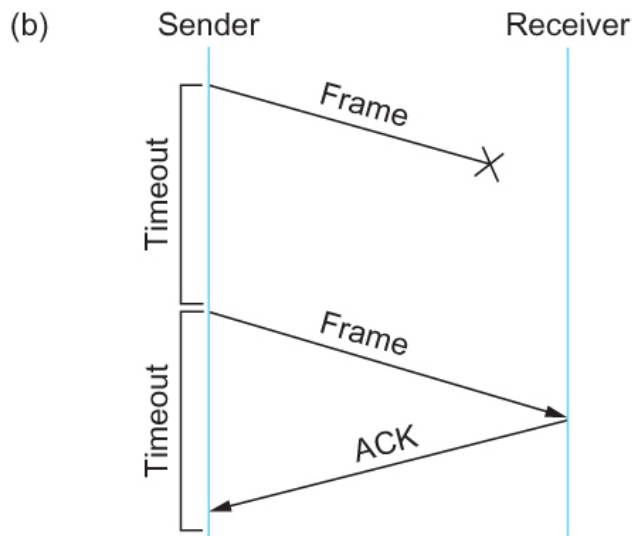
1. **Sender** transmits one frame
2. Waits for an acknowledgement (ACK)
3. If the ACK does **not arrive** after a certain time (timeout) **retransmits** the original frame

# Stop and Wait Protocol



## Different scenarios in *stop-and-wait*

- (a) The ACK is received before the timer expires
- (b) Original frame is lost
- (c) The ACK is lost
- (d) The timeout fires too soon



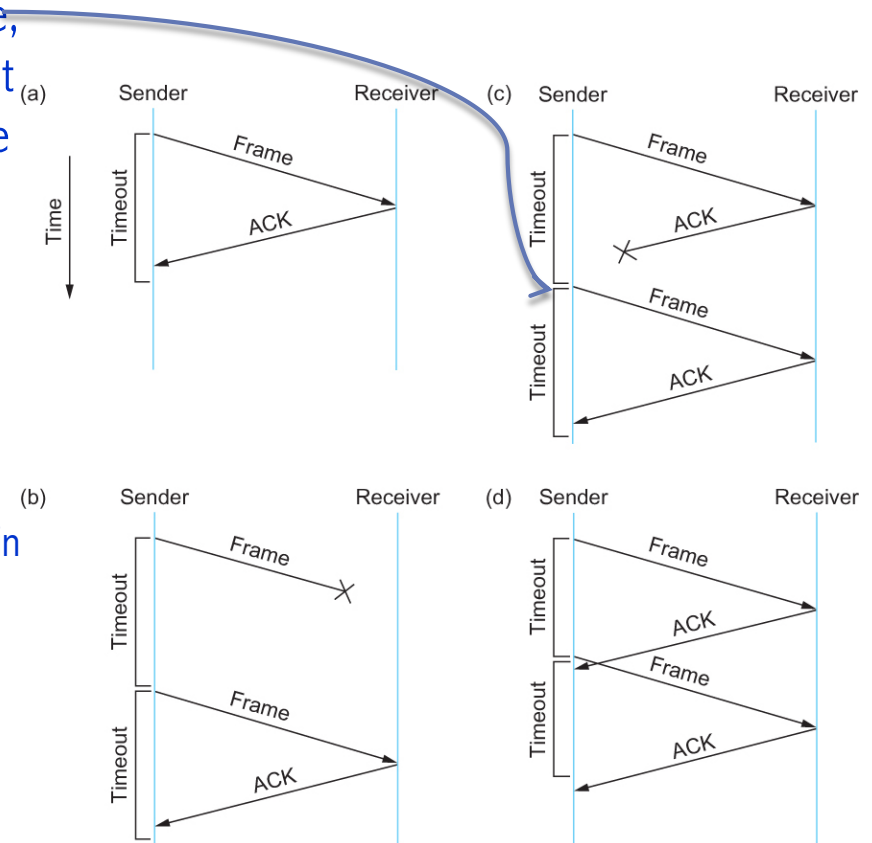
# Stop and Wait Protocol

## ■ ACK is lost or delayed (c) and (d)

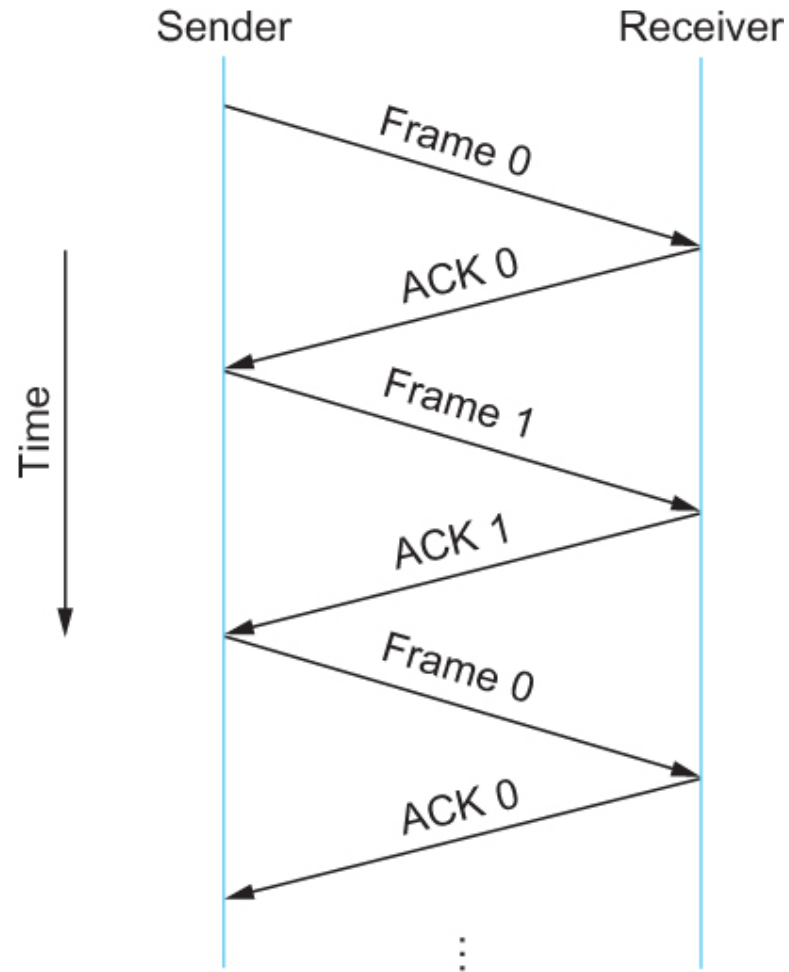
- The sender times out and retransmits the original frame, but the *receiver will think that it is the next frame* since it has correctly received and acknowledged the first frame
- As a result: duplicate copies of frames will be delivered

## ■ How to solve this

- Use 1 bit sequence number (0 or 1)
- When the sender retransmits frame 0, the receiver can determine that it is seeing a second copy of frame 0 rather than the first copy of frame 1 and therefore can ignore it (the receiver still acknowledges it, in case the first acknowledgement was lost)



# Stop and Wait Protocol



Timeline for stop-and-wait with 1-bit sequence number

# Stop and Wait Protocol

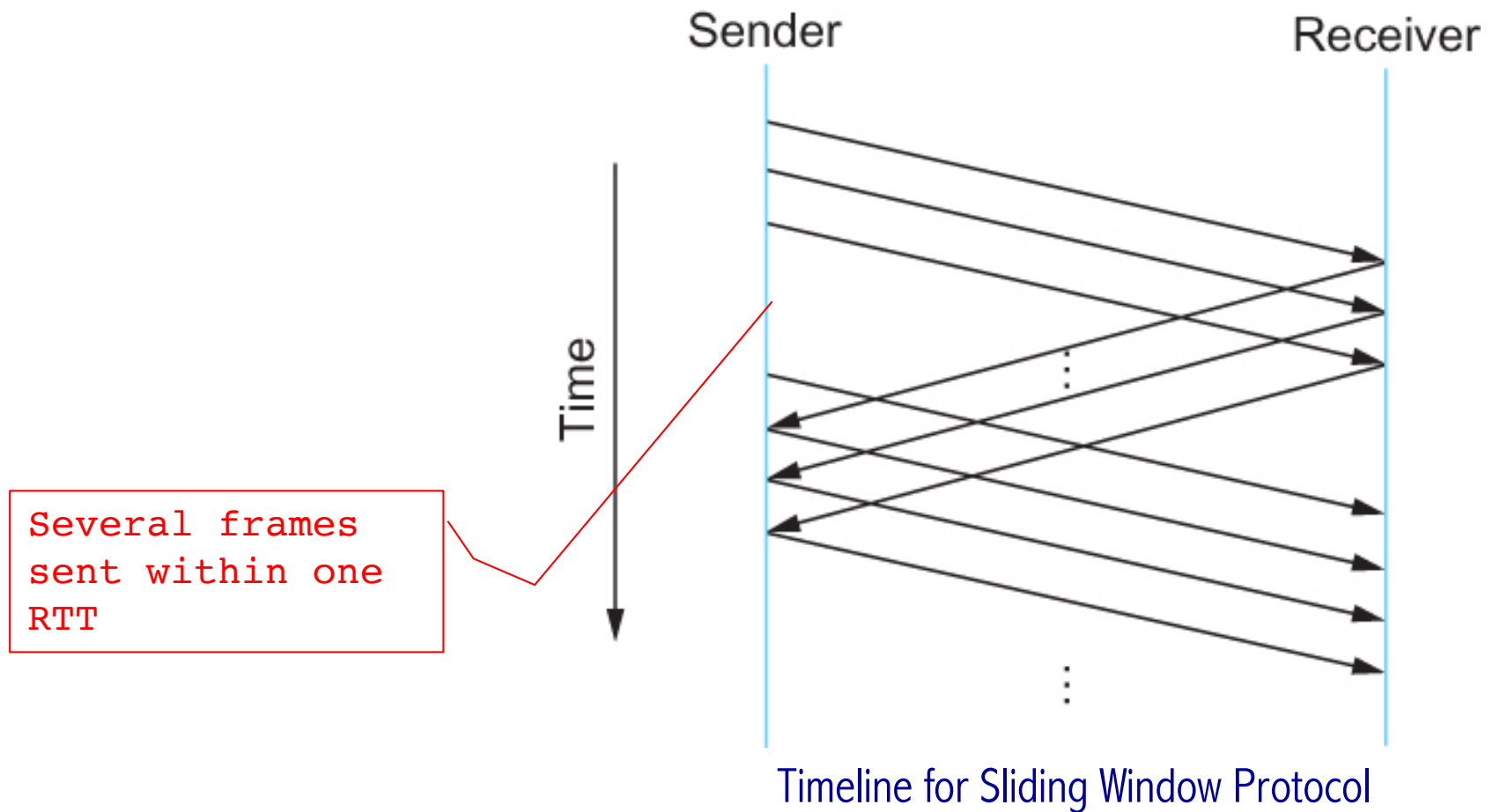
Delay x bandwidth: low use of network resources

- The sender has **only one outstanding frame** on the link at a time
  - ▣ This may be far below the link's capacity
- Consider a 1.5 Mbps link with a 45 ms RTT
  - ▣ The link has a delay × bandwidth product of 67.5 Kb or approximately 8 KB
  - ▣ Since the sender can send only one frame per RTT and assuming a frame size of 1 KB
  - ▣ Maximum Sending rate
    - Bits per frame ÷ Time per frame =  $1024 \times 8 \div 0.045 = 182 \text{ Kbps}$   
Or about **one-eighth of the link's capacity**
  - ▣ **To use the link fully**, then sender should transmit up to eight frames before having to wait for an acknowledgement



# Sliding Window Protocol

One solution is the Sliding Window Protocol



# Sliding Window Protocol

- Sender assigns a sequence number denoted as **SeqNum** to each frame.
  - ▣ Assume SeqNum can grow infinitely large

- Sender maintains three variables

- ▣ Sending Window Size (**SWS**)

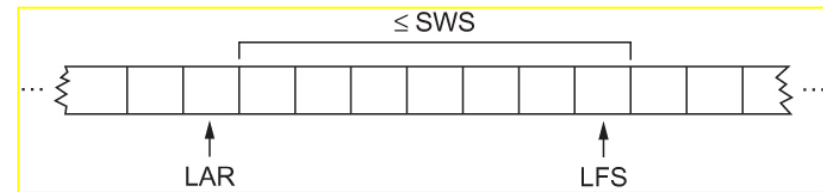
- ▣ Upper bound on the number of outstanding (unacknowledged) frames that the sender can transmit

- ▣ Last Acknowledgement Received (**LAR**)

- ▣ Sequence number of the last acknowledgement received

- ▣ Last Frame Sent (**LFS**)

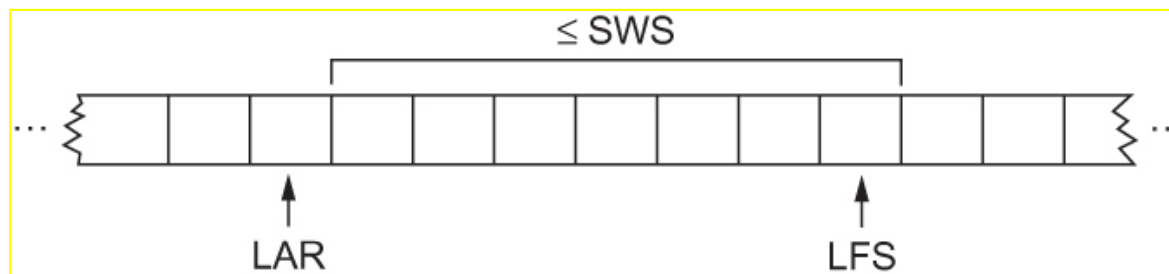
- ▣ Sequence number of the last frame sent



# Sliding Window Protocol

- Sender also maintains the following invariant

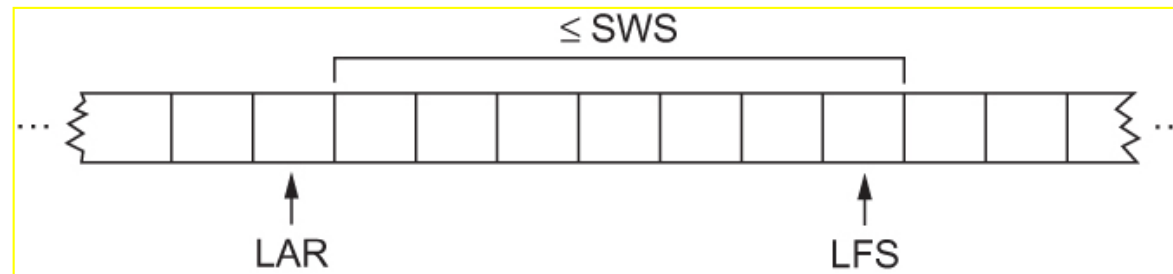
$$LFS - LAR \leq SWS$$



Sliding Window on Sender

# Sliding Window Protocol

- When an acknowledgement arrives
  - ▣ the sender moves LAR to right, thereby allowing the sender to transmit another frame
- Also the sender associates a timer with each frame it transmits
  - ▣ It retransmits the frame if the timer expires before the ACK is received
- Note that the sender has to be willing to buffer up to SWS frames
  - ▣ WHY?



Sliding Window on Sender

# Issues with Sliding Window Protocol



- Serves three different roles

- ▣ **Reliable**

- Errors, loss, delays, etc.: retransmissions

- ▣ **Preserve the order**

- Each frame has a sequence number
    - The receiver makes sure that it does not pass a frame up to the next higher-level protocol until it has already passed up all frames with a smaller sequence number

- ▣ **Flow control**

- Receiver is able to throttle the sender
      - Keeps the sender from overrunning the receiver
        - From transmitting more data than the receiver is able to process

# Issues with Sliding Window Protocol

- Negative Acknowledgement (NAK)
  - ▣ Receiver sends NAK for frame 6 when frame 7 arrive (in the previous example)
    - However this is unnecessary since sender's timeout mechanism will be sufficient to catch the situation
  
- Additional Acknowledgement
  - ▣ Receiver sends additional ACK for frame 5 when frame 7 arrives
    - Sender uses duplicate ACK as a clue for frame loss
  
- Selective Acknowledgement
  - ▣ Receiver will acknowledge exactly those frames it has received, rather than the highest number frames
    - Receiver will acknowledge frames 7 and 8
    - Sender knows frame 6 is lost
    - Sender can keep the pipe full (additional complexity)