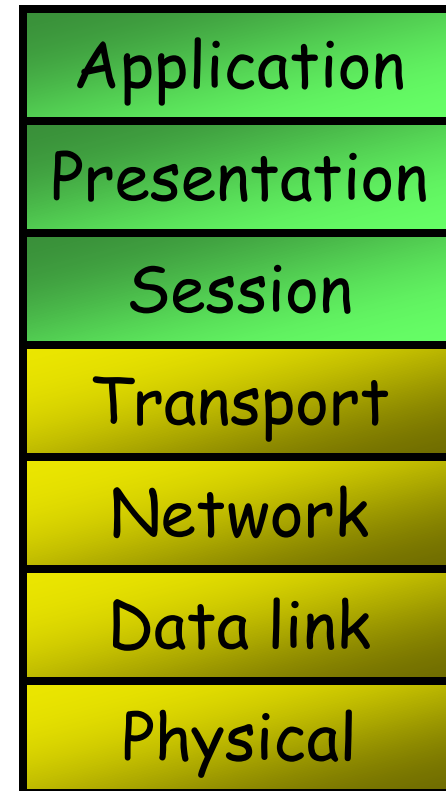




# Tầng Liên kết dữ liệu

# MỤC TIÊU

- điều khiển truy cập đường truyền
- Điều khiển liên kết

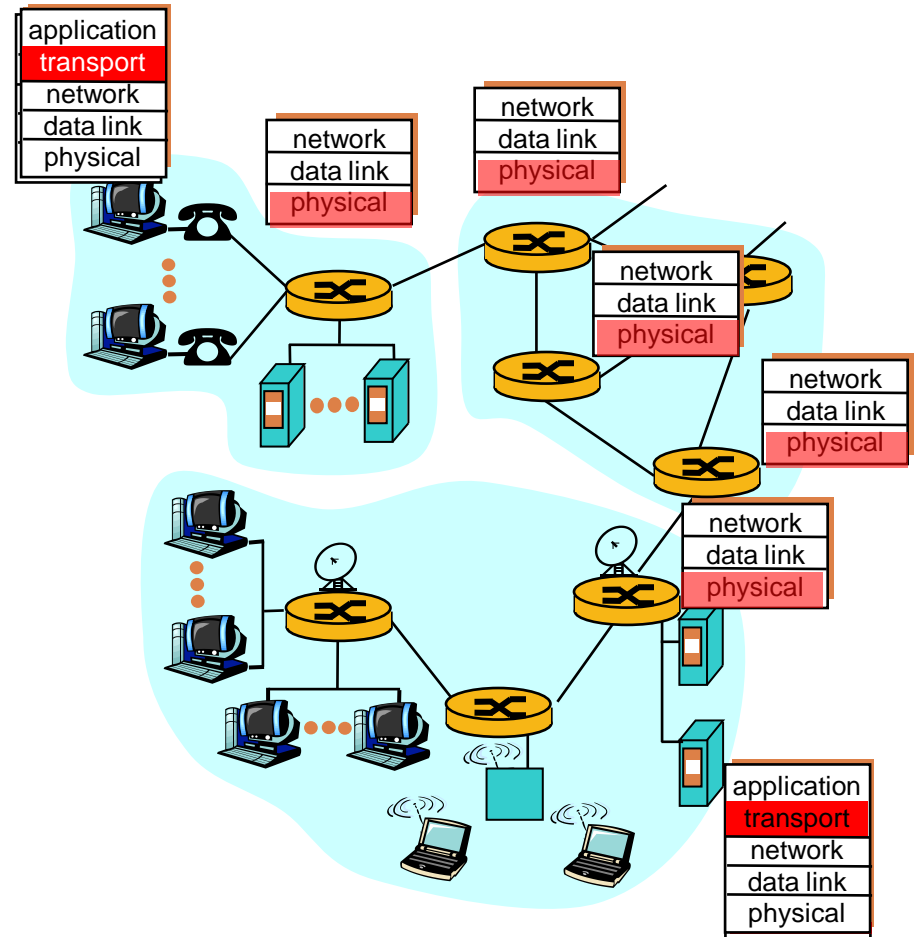


# NỘI DUNG

- Giới thiệu
- Kỹ thuật phát hiện và sửa lỗi
- Điều khiển truy cập đường truyền
- ARP
- Ethernet

# GIỚI THIỆU - 1

- Link: “kết nối/liên kết” giữa các nodes kề nhau
  - Wired
  - Wireless
- Data link layer: chuyển gói tin (frame) từ một node đến node kề qua 1 link
  - Mỗi link có thể dùng giao thức khác nhau để truyền tải frame



## GIỚI THIỆU - 2

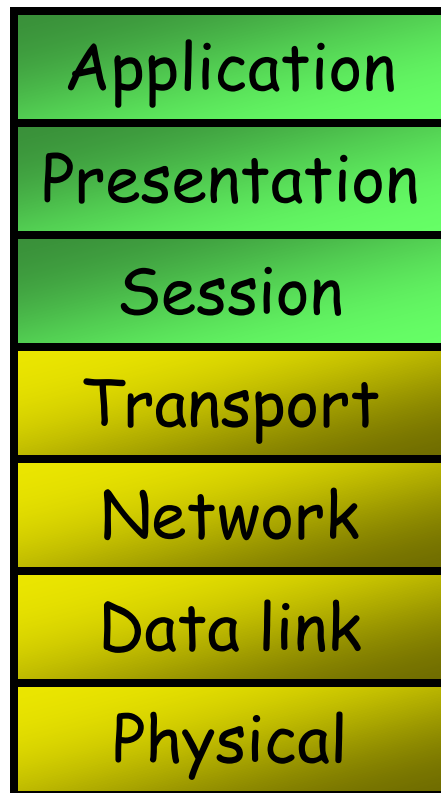
### ○ Tại nơi gửi:

- Nhận các packet từ tầng network → đóng gói thành các frame
- Truy cập đường truyền (nếu dùng đường truyền chung)

### ○ Tại nơi nhận:

- Nhận các frame dữ liệu từ tầng physical
- Kiểm tra lỗi
- Chuyển cho tầng network

# GIỚI THIỆU - 3



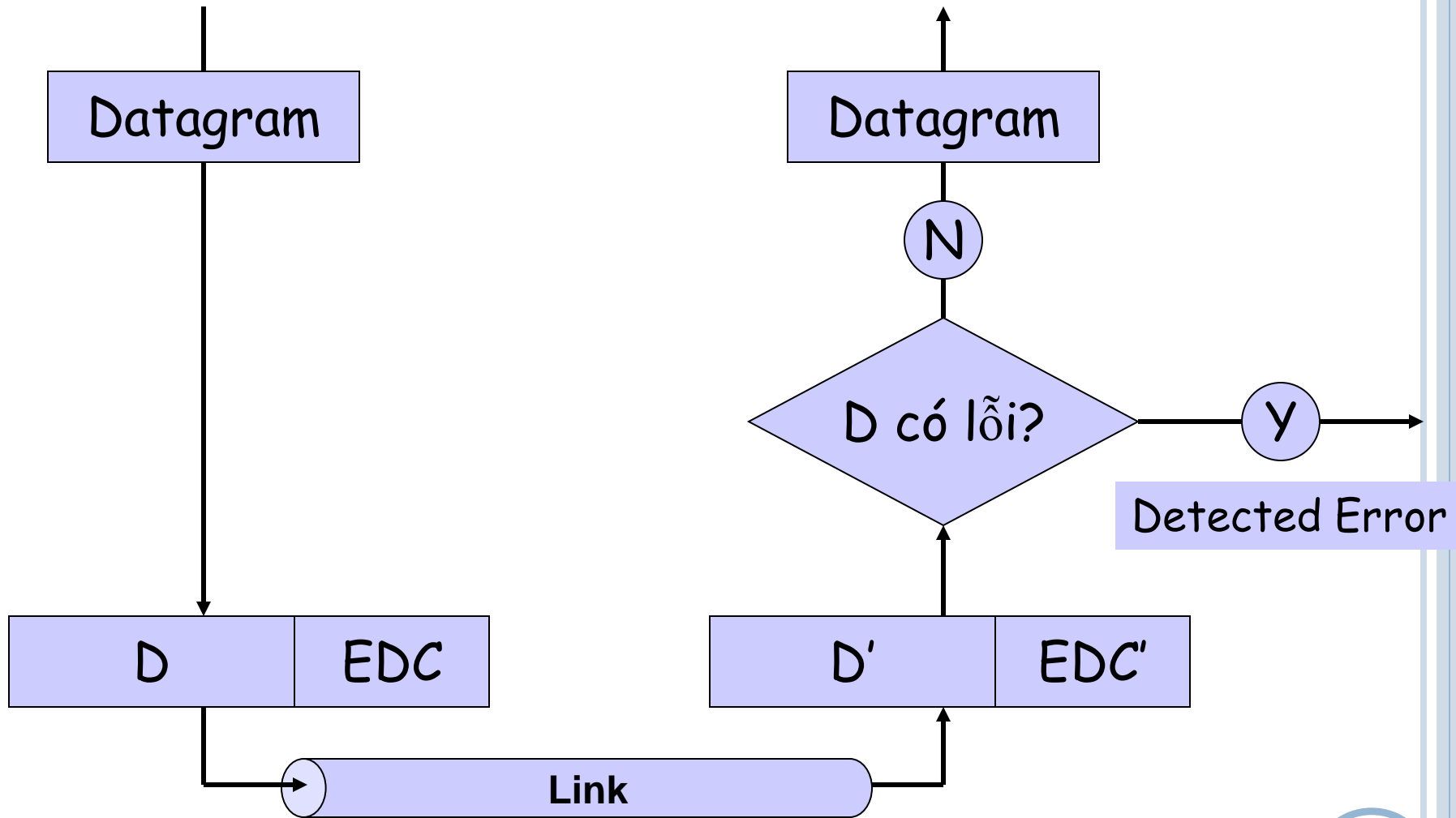
- LLC (Logical Link Control)
  - Điều khiển luồng
  - Kiểm tra lỗi
  - Báo nhận
- MAC (Media Access Control)
  - Truy cập đường truyền



# NỘI DUNG

- Giới thiệu
- Kỹ thuật phát hiện và sửa lỗi
- Điều khiển truy cập đường truyền
- ARP
- Ethernet

# KỸ THUẬT PHÁT HIỆN VÀ SỬA LỖI - 1



EDC= Error Detection and Correction

D = Data



# KỸ THUẬT PHÁT HIỆN VÀ SỬA LỖI - 2

- Các phương pháp:
  - Parity Check (bit chẵn lẻ)
  - Checksum
  - Cyclic Redundancy Check (CRC)

# PARITY CHECK

- Dùng thêm một số bit để đánh dấu tính chẵn lẻ
  - Dựa trên số bit 1 trong dữ liệu
  - Phân loại:
    - Even Parity: số bit 1 phải là một số chẵn
    - Odd Parity: số bit 1 phải là một số lẻ
- Các phương pháp:
  - Parity 1 chiều
  - Parity 2 chiều
  - Hamming code

# PARITY 1 CHIỀU - 1

- Số bit parity: 1 bit
- Chiều dài của dữ liệu cần gửi đi:  $d$  bit  
→ DL gửi đi sẽ có  $(d+1)$  bit
- Bên gửi:
  - Thêm 1 bit parity vào dữ liệu cần gửi đi
    - Mô hình chẵn (Even parity)
      - số bit 1 trong  $d+1$  bit là một số chẵn
    - Mô hình lẻ (Odd Parity)
      - số bit 1 trong  $d+1$  bit là một số lẻ

d bits	Parity bit	
0111000110101011	1	(mô hình chẵn)
	0	(mô hình lẻ)

# PARITY 1 CHIỀU - 2

## ○ Bên nhận:

- Nhận D' có  $(d+1)$  bits
- Đếm số bit 1 trong  $(d+1)$  bits = x
- Mô hình chẵn: nếu x lẻ  $\rightarrow$  error
- Mô hình lẻ: nếu x chẵn  $\rightarrow$  error

## ○ Ví dụ: nhận 0111000110101011

- Parity chẵn: sai
- Parity lẻ: đúng
  - Dữ liệu thật: 011100011010101

## ○ Đặc điểm:

- Phát hiện được lỗi khi số bit lỗi trong dữ liệu là số lẻ
- Không sửa được lỗi

# PARITY 2 CHIỀU - 1

- Dữ liệu gửi đi được biểu diễn thành ma trận  $N \times M$
- Số bit parity:  $(N + M + 1)$  bit
- Đặc điểm:
  - Phát hiện và sửa được 1 bit lỗi
- Bên gửi
  - Biểu diễn dữ liệu cần gửi đi thành ma trận  $N \times M$
  - Tính giá trị bit parity của từng dòng, từng cột

				row →
				parity
	$d_{1,1}$	$\dots$	$d_{1,j}$	$d_{1,j+1}$
	$d_{2,1}$	$\dots$	$d_{2,j}$	$d_{2,j+1}$
	$\dots$	$\dots$	$\dots$	$\dots$
	$d_{i,1}$	$\dots$	$d_{i,j}$	$d_{i,j+1}$
column ↓	$d_{i+1,1}$	$\dots$	$d_{i+1,j}$	$d_{i+1,j+1}$
parity				

# PARITY 2 CHIỀU - 2

- Ví dụ:

- Dùng parity chẵn
- $N = 3, M = 5$
- Dữ liệu cần gửi đi: 10101 11110 01110

10101	1
11110	0
01110	1
<hr/>	
00101	0

# PARITY 2 CHIỀU - 1

## ○ Bên nhận:

- Biểu diễn dữ liệu nhận thành ma trận  $(N+1) \times (M+1)$
- Kiểm tra tính đúng đắn của từng dòng/cột
- Đánh dấu các dòng/cột dữ liệu bị lỗi
- Bit lỗi: bit tại vị trí giao giữa dòng và cột bị lỗi

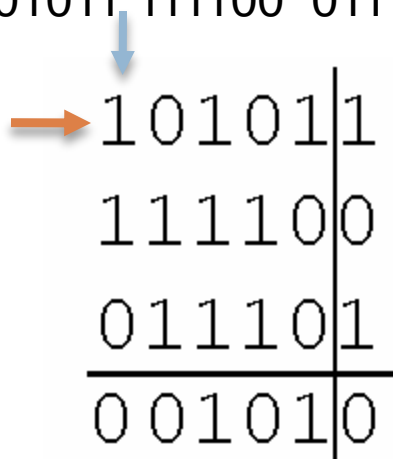
# PARITY 2 CHIỀU - 2

## ○ Ví dụ:

- Dùng parity chẵn
- $N = 3, M = 5$

Dữ liệu nhận:

101011 111100 011101 001010



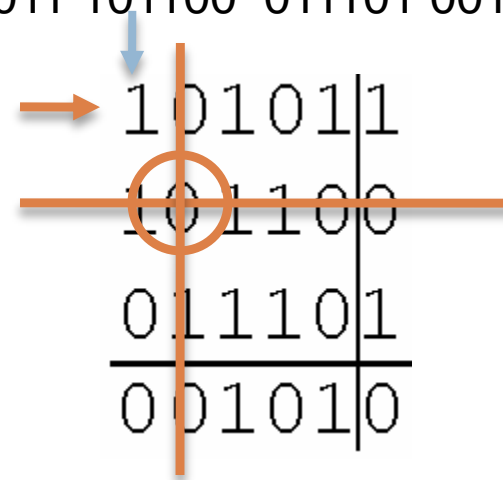
1	0	1	0	1	1
1	1	1	1	0	0
0	1	1	1	0	1
0	0	1	0	1	0

**Không có lỗi**

Dữ liệu thật: 10101 11110 01110

Dữ liệu nhận:

101011 101100 011101 001010



1	0	1	0	1	1
1	0	1	1	0	0
0	1	1	1	0	1
0	0	1	0	1	0

**Có lỗi**

Dữ liệu thật: 10101 11110 01110



# HAMMING CODE - 1

- Mỗi hamming code

- có M bit, đánh số từ 1 đến M
- Bit parity:  $\log_2 M$  bits, tại các vị trí lũy thừa của 2
- Dữ liệu thật được đặt tại các vị trí không là lũy thừa của 2
- VD:  $M = 7$ 
  - $\log_2 7 = 3$ : dùng 3 bits làm bit parity (1, 2, 4)
  - Có 4 vị trí có thể đặt dữ liệu (3, 5, 6, 7)

- Đặc điểm:

- sửa lỗi 1 bit
- nhận dạng được 2 bit lỗi
- Sửa lỗi nhanh hơn Parity code 2 chiều

# HAMMING CODE - 2

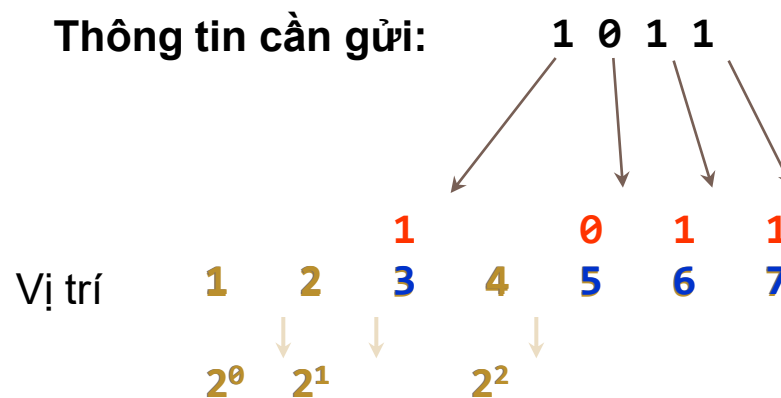
## ◦ Bên gửi:

- Chia dữ liệu cần gửi đi thành các khối dữ liệu (với số bit là số vị trí có thể đặt vào Hamming Code)
- Với mỗi khối dữ liệu → tạo 1 Hamming Code
  - Đặt các bit dữ liệu vào các vị trí không phải là lũy thừa của 2 trong Hamming Code
    - lưu ý: vị trí được đánh số từ 1 đến M
  - Tính check bits
  - Tính giá trị của các bit parity

# HAMMING CODE – 3

## ○ Ví dụ:

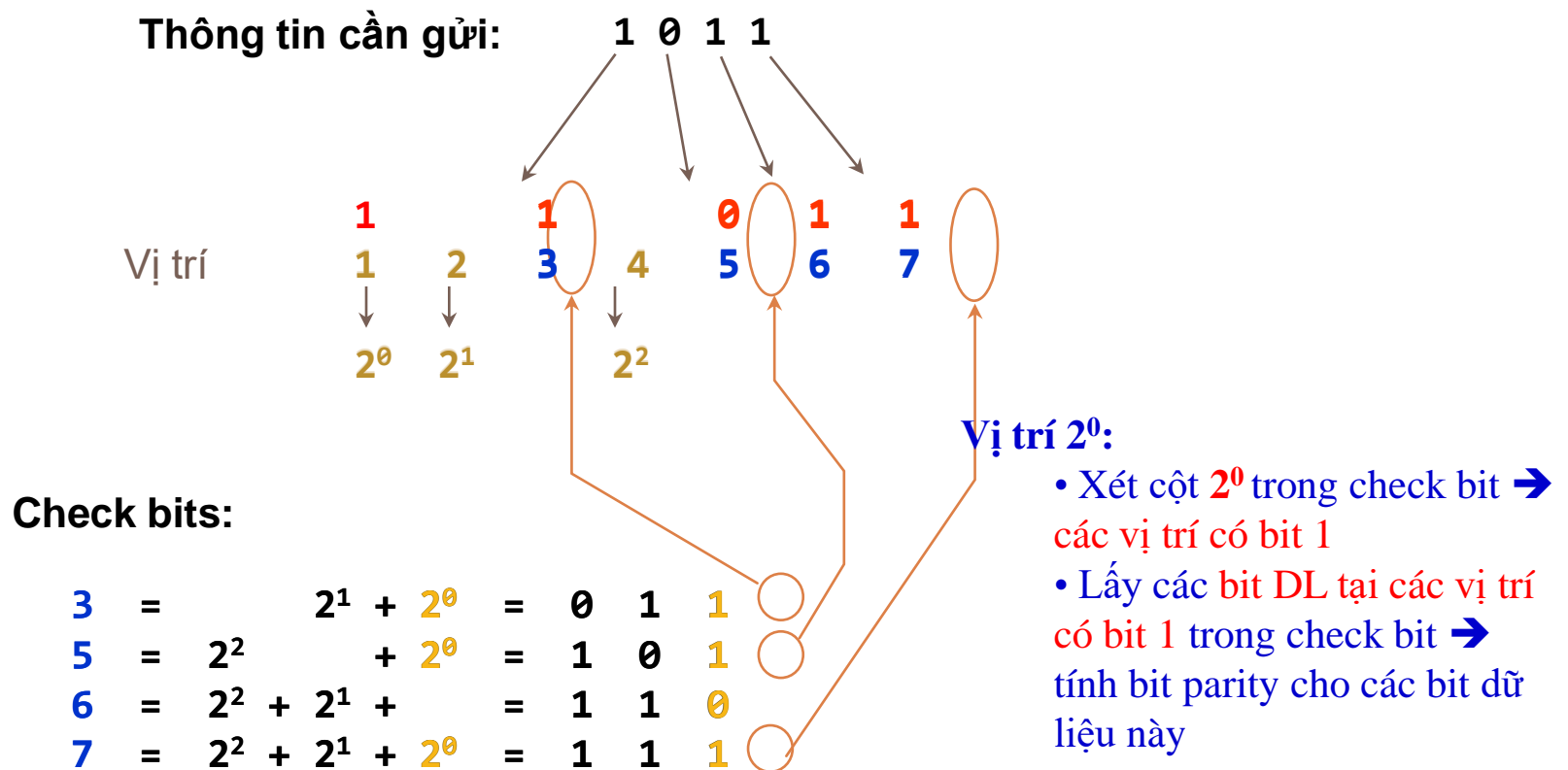
- $M = 7$
- Dùng parity lẻ
- Thông tin cần gửi: 1011



Tính check bits:

$$\begin{array}{rclcl} 3 & = & 2^1 + 2^0 & = & 0 \ 1 \ 1 \\ 5 & = & 2^2 + 2^0 & = & 1 \ 0 \ 1 \\ 6 & = & 2^2 + 2^1 + & = & 1 \ 1 \ 0 \\ 7 & = & 2^2 + 2^1 + 2^0 & = & 1 \ 1 \ 1 \end{array}$$

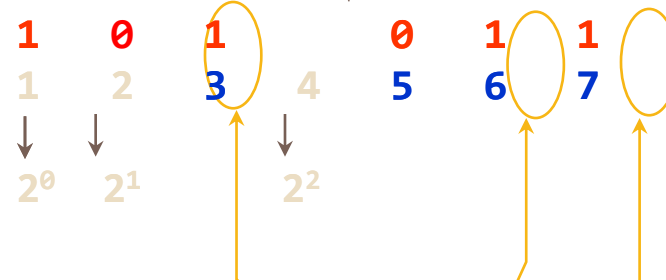
# HAMMING CODE - 4



# HAMMING CODE - 5

Thông tin cần gửi: 1 0 1 1

Vị trí



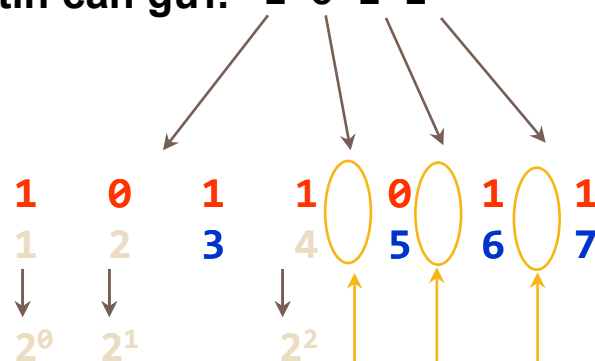
Check bits:

$$\begin{aligned}
 3 &= 2^1 + 2^0 = 0 \quad 1 \quad 1 \\
 5 &= 2^2 + 2^0 = 1 \quad 0 \quad 1 \\
 6 &= 2^2 + 2^1 + 2^0 = 1 \quad 1 \quad 0 \\
 7 &= 2^2 + 2^1 + 2^0 = 1 \quad 1 \quad 1
 \end{aligned}$$

# HAMMING CODE - 6

Thông tin cần gửi: 1 0 1 1

Vị trí



Check bits:

$$\begin{aligned}
 3 &= 2^1 + 2^0 = 0 \ 1 \ 1 \\
 5 &= 2^2 + 2^0 = 1 \ 0 \ 1 \\
 6 &= 2^2 + 2^1 + 2^0 = 1 \ 1 \ 0 \\
 7 &= 2^2 + 2^1 + 2^0 = 1 \ 1 \ 1
 \end{aligned}$$

# HAMMING CODE - 7

- Dữ liệu cần gửi: 1011
- Dữ liệu gửi: 1011011

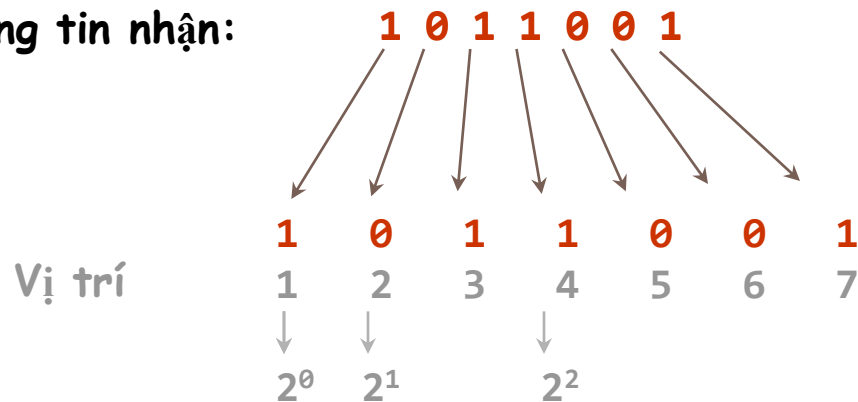
# HAMMING CODE - 8

- Bên nhận: với mỗi Hamming Code
  - Điền các bit Hamming Code nhận vào các vị trí từ 1 đến M
  - Tính check bit
  - Kiểm tra các bit parity
    - Nếu tại bit  $2^i$  phát hiện sai → đánh dấu Error, hệ số  $k_i = 1$
    - Ngược lại, đánh dấu No Error = 0, hệ số  $k_i = 0$
  - Vị trí bit lỗi:  $pos = \sum 2^i \cdot k_i$



# HAMMING CODE – 9

Thông tin nhận:



Tính check bits:

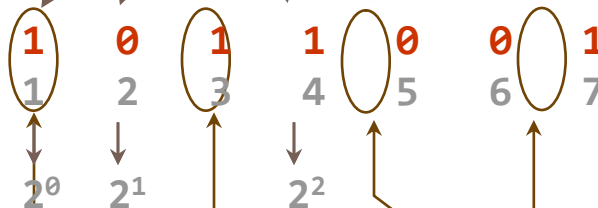
$$\begin{aligned} 3 &= 2^1 + 2^0 = 0 \ 1 \ 1 \\ 5 &= 2^2 + 2^0 = 1 \ 0 \ 1 \\ 6 &= 2^2 + 2^1 = 1 \ 1 \ 0 \\ 7 &= 2^2 + 2^1 + 2^0 = 1 \ 1 \ 1 \end{aligned}$$

# HAMMING CODE – 10

Thông tin nhận:

1 0 1 1 0 0 1

Vị trí



Tính check bits:

$$\begin{aligned}
 3 &= 2^1 + 2^0 = 0 \ 1 \ 1 \\
 4 &= 2^2 + 2^0 = 1 \ 0 \ 1 \\
 6 &= 2^2 + 2^1 = 1 \ 1 \ 0
 \end{aligned}$$

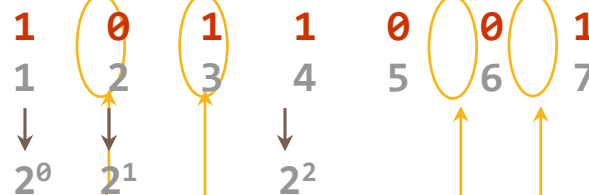
Odd parity: Không có lỗi

# HAMMING CODE – 11

Thông tin nhận:

1 0 1 1 0 0 1

Vị trí



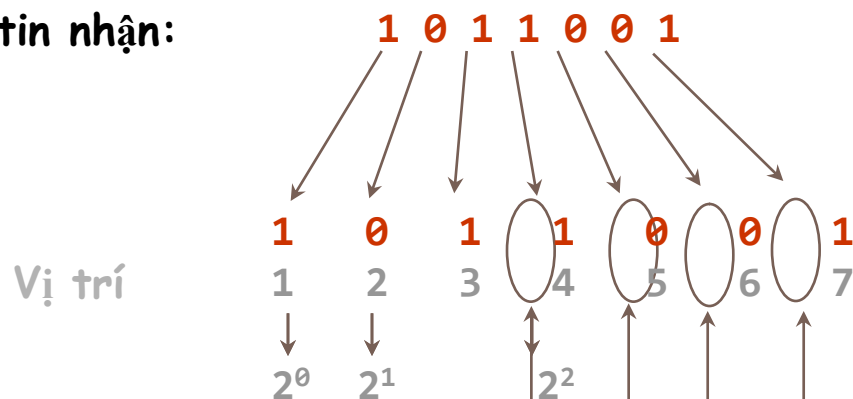
Tính check bits:

$$\begin{array}{rclcl} 3 & = & 2^1 + 2^0 & = & 0 \ 1 \ 1 \\ 5 & = & 2^2 + 2^0 & = & 1 \ 0 \ 1 \\ 6 & = & 2^2 + 2^1 & = & 1 \ 1 \ 0 \\ 7 & = & 2^2 + 2^1 + 2^0 & = & 1 \ 1 \ 1 \end{array}$$

Odd parity: LỖI

# HAMMING CODE – 12

Thông tin nhận:



Tính check bits:

$$\begin{array}{rcll} 3 & = & 2^1 + 2^0 & = 0 \ 1 \ 1 \\ 5 & = & 2^2 + 2^0 & = 1 \ 0 \ 1 \\ 6 & = & 2^2 + 2^1 & = 1 \ 1 \ 0 \\ 7 & = & 2^2 + 2^1 + 2^0 & = 1 \ 1 \ 1 \end{array}$$

Odd parity: LỖI



# CHECK SUM - 1

## ○ Bên gửi

- d bits trong DL gửi đi được xem như gồm N số k bits:  
 $x_1, x_2, \dots, x_N$
- Tính tổng  $X = x_1 + x_2 + \dots + x_N$
- Tính **bù 1** của X  $\rightarrow$  giá trị checksum

## ○ VD: Dữ liệu cần gửi: 1110 0110 0110 0110, k = 4

- 1110, 0110, 0110, 0110
- 0101, 0110, 0110
- ....
- Sum = 0010
- Checksum = 1101

$$\begin{array}{r} 1110 \\ 0110 \\ \hline 0100 \\ \text{ } \rightarrow 1 \\ \hline 0101 \end{array}$$

# CHECK SUM - 1

- Bên nhận:

- tính tổng cho tất cả giá trị nhận được (kể cả giá trị checksum).
- Nếu tất cả các bit là 1, thì dữ liệu nhận được là đúng; ngược lại: có lỗi xảy ra

- VD:

- nhận: 1110            0110    0110    0110    1101
  - Sum = 1111
  - ➔ đúng
- Nhận: 1010   0110    0110    0110    1101
  - Sum = 1011
  - ➔ sai

# NỘI DUNG

- Giới thiệu
- Kỹ thuật phát hiện và sửa lỗi
- Điều khiển truy cập đường truyền
- ARP
- Ethernet



# ĐIỀU KHIỂN TRUY CẬP ĐƯỜNG TRUYỀN - 1

## ◦ Loại liên kết (link)

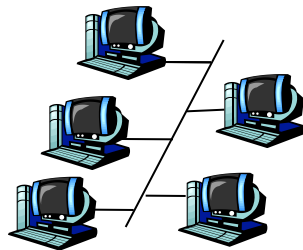
- Điểm đến điểm (Point-to-point)



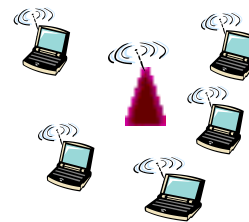
- Dialup

- Nối trực tiếp giữa: host - host, host – SW

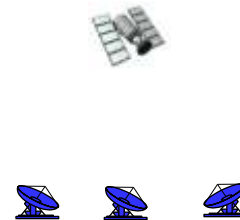
- Chia sẻ (Shared)



shared wire (e.g.,  
cabled Ethernet)



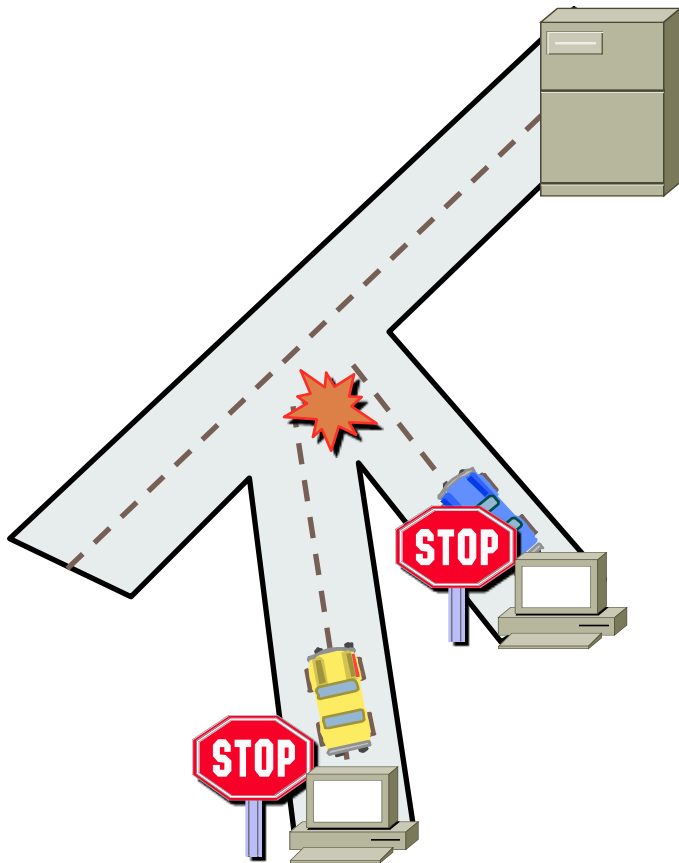
shared RF  
(e.g., 802.11 WiFi)



shared RF  
(satellite)

# ĐIỀU KHIỂN TRUY CẬP ĐƯỜNG TRUYỀN - 2

- Trong môi trường chia sẻ



## Hạn chế xảy ra collision

➔ Giao thức tầng Data link:  
Quyết định cơ chế để các  
node sử dụng môi trường  
chia sẻ

- khi nào được phép gửi DL xuống đường truyền
- Làm sao phát hiện xảy ra Collision
- ....

# ĐIỀU KHIỂN TRUY CẬP ĐƯỜNG TRUYỀN - 3

- Các phương pháp:

- Phân chia kênh truyền (Channel partition protocols)
- Tranh chấp (Random access protocols)
- Luân phiên (Taking-turns protocols)

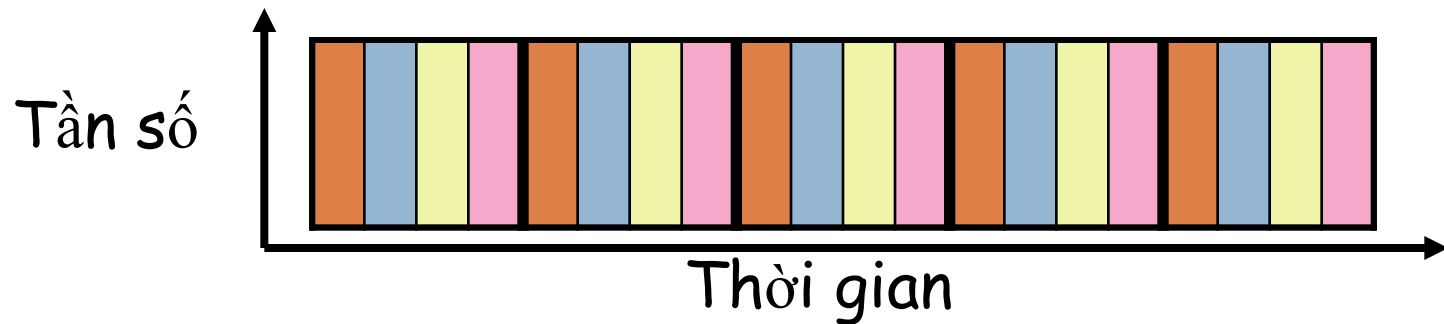
# PHÂN CHIA KÊNH TRUYỀN

- TDM (Time Division Multiplexing)
- FDM (Frequency Division Multiplexing)
- CDMA (Code Division Multiple Access)

# TDM

## ○ Ý tưởng:

- Chia kênh truyền thành các khe thời gian
  - Mỗi khe thời gian chia thành N khe nhỏ
  - Mỗi khe nhỏ dành cho 1 node trong mạng
- ➔ Mỗi node có băng thông:  $R/N$



# FDM

## ○ Ý tưởng:

- Chia kênh truyền thành N kênh truyền nhỏ
  - Mỗi kênh truyền dành cho 1 node
- ➔ Mỗi node có băng thông:  $R/N$

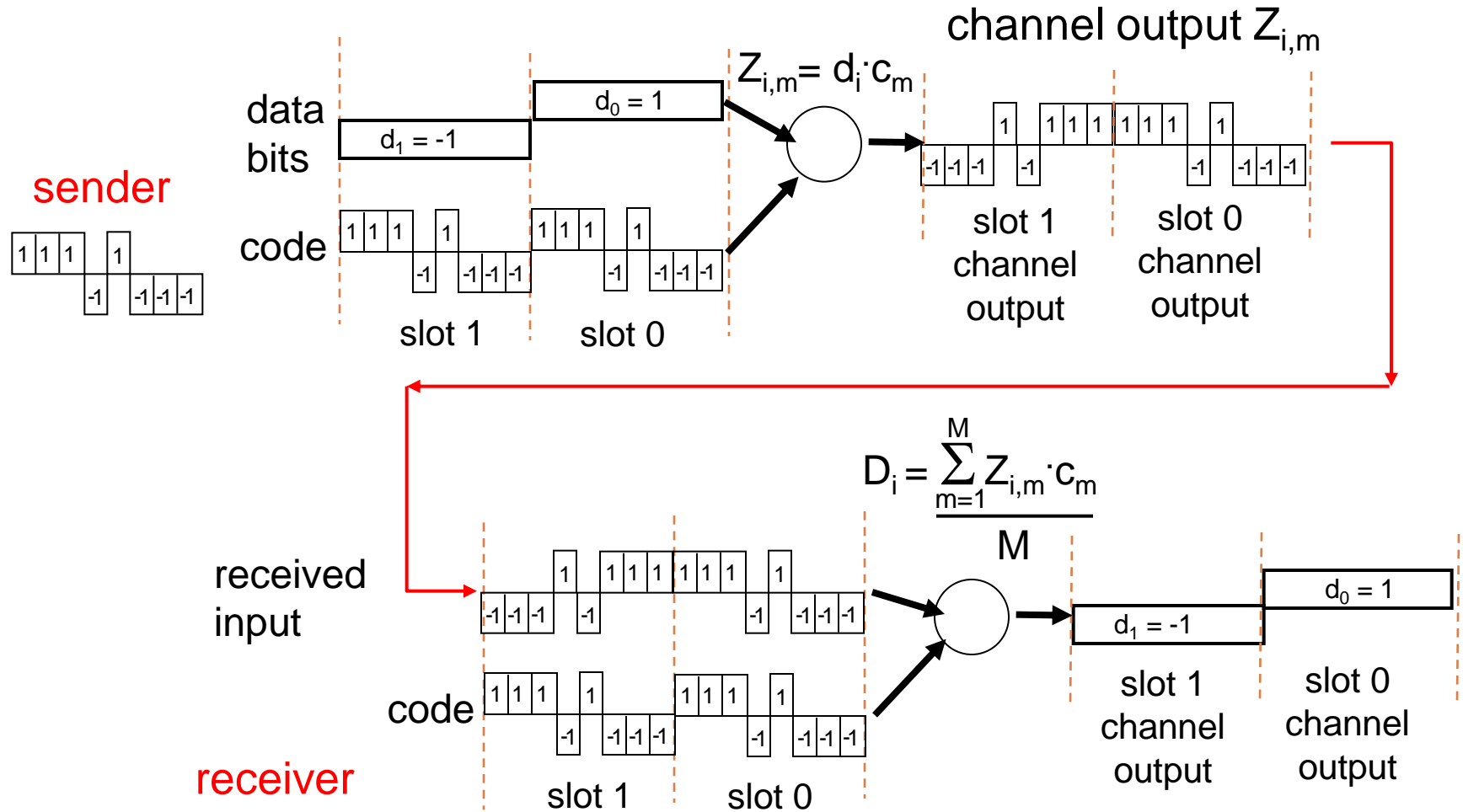


# CDMA - 1

- Ý tưởng:

- Mỗi node có 1 code riêng
- Bên gửi: mã hoá dữ liệu trước khi gửi bằng code của mình và bên nhận phải biết code của người gửi
- 1 bit DL được mã hoá thành M bits
- Kênh truyền: chia thành từng các khe thời gian, mỗi bit truyền trong 1 khe

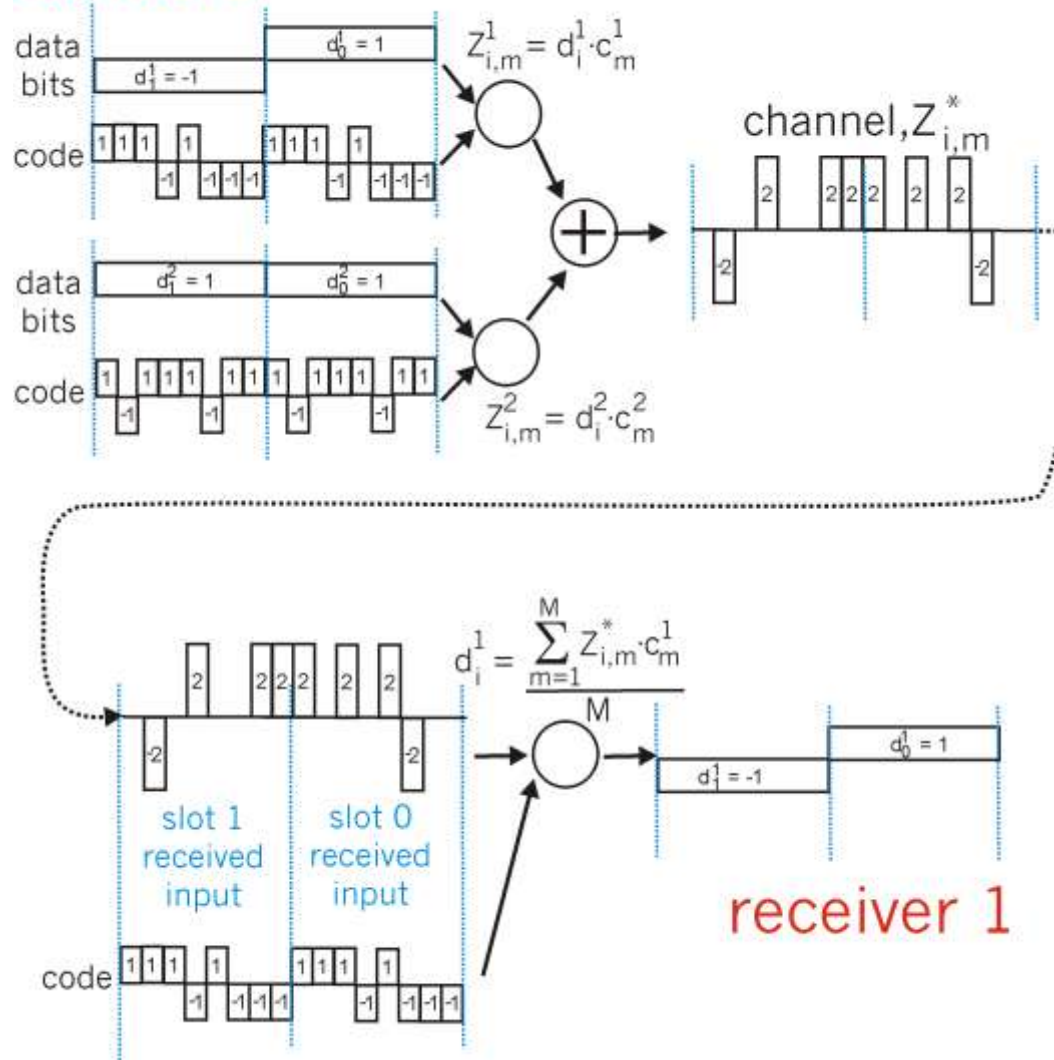
# CDMA - 2





# CDMA - 3

senders



# TRANH CHẤP

- Các node chiếm trọn băng thông khi truyền
- Lắng nghe đưng độ sau khi truyền
- Một số phương pháp:
  - ALOHA (Slotted, Pure)
  - CSMA (Carrier Sense Multiple Access)

# PURE ALOHA

- Mỗi node có thể bắt đầu truyền dữ liệu bất cứ khi nào node có nhu cầu
- Nếu phát hiện xung đột → chờ 1 khoảng thời gian rồi truyền lại

# SLOTTED ALOHA

- Giả thiết:
  - Các frame có kích thước tối đa là  $L$  bits
- Kênh truyền: chia thành các khe thời gian có kích thước  $L/R$  (s)
- Khi 1 node có nhu cầu truyền dữ liệu: phải chờ đến thời điểm bắt đầu của 1 khe mới được truyền
  - cần đồng bộ thời gian giữa các node
- Nếu đụng độ xảy ra: truyền lại với xác suất là  $p$

# CSMA - 1

- Lắng nghe đường truyền trước khi truyền:
  - Đường truyền rảnh: truyền dữ liệu
  - Đường truyền bận: chờ
- Lắng nghe đường truyền sau khi truyền
  - Nếu đụng độ xảy ra:
    - dừng truyền
    - đợi 1 khoảng thời gian và truyền lại

# CSMA - 2

## ○ Đánh giá:

- Các node có quyền ngang nhau
- Chi phí cao
- Tốc độ: chấp nhận được nếu số lượng node ít
- Không ấn định độ ưu tiên cho thiết bị đặc biệt

## ○ Cải tiến:

- CSMA/CD (Carrier Sense Multiple Access / Collision Detection)
- CSMA/CA (Carrier Sense Multiple Access / Collision Avoidance)

# CSMA/CD

## ○ Ý tưởng:

- Thiết bị lắng nghe đường truyền
- Nếu đường truyền rảnh, thiết bị truyền DL của mình lên đường truyền
- Sau khi truyền, lắng nghe đụng độ?
- Nếu có, thiết bị gửi tín hiệu cảnh báo các thiết bị khác
- Tạm dừng 1 khoảng thời gian ngẫu nhiên rồi gửi DL
- Nếu tiếp tục xảy ra đụng độ, tạm dừng khoảng thời gian gấp đôi.

## ○ Dùng trong mạng Ethernet

# CSMA/CA

## ○ Ý tưởng:

- Thiết bị lắng nghe đường truyền
- Nếu đường truyền rảnh, thiết bị gửi tín hiệu bắt đầu truyền tín hiệu RTS (request to send)
- Sau khi truyền xong, gửi tín hiệu báo xong CTS (clear to send)

## ○ Dùng trong mạng LocalTalk



# LUÂN PHIÊN

- Dùng thẻ bài (Token Passing)
- Lò chọn (Polling)

# TOKEN PASSING

- Ý tưởng:
  - Dùng 1 thẻ bài (token) di chuyển qua các node
  - Thiết bị muốn truyền DL thì phải chiếm được thẻ bài
- Đánh giá:
  - Thích hợp cho các mạng có tải nặng
  - Thiết lập được độ ưu tiên cho thiết bị đặc biệt
  - Chậm hơn CSMA trong mạng có tải nhẹ
  - Thiết bị mạng đắt tiền
- Dùng trong mạng Token Ring

# POLLING

## ○ Ý tưởng:

- Có 1 node đóng vai trò điều phối
- Node điều phối kiểm tra nhu cầu gửi DL của các node thứ cấp và xếp vào hàng đợi theo thứ tự và độ ưu tiên
- Thiết bị truyền DL khi đến lượt

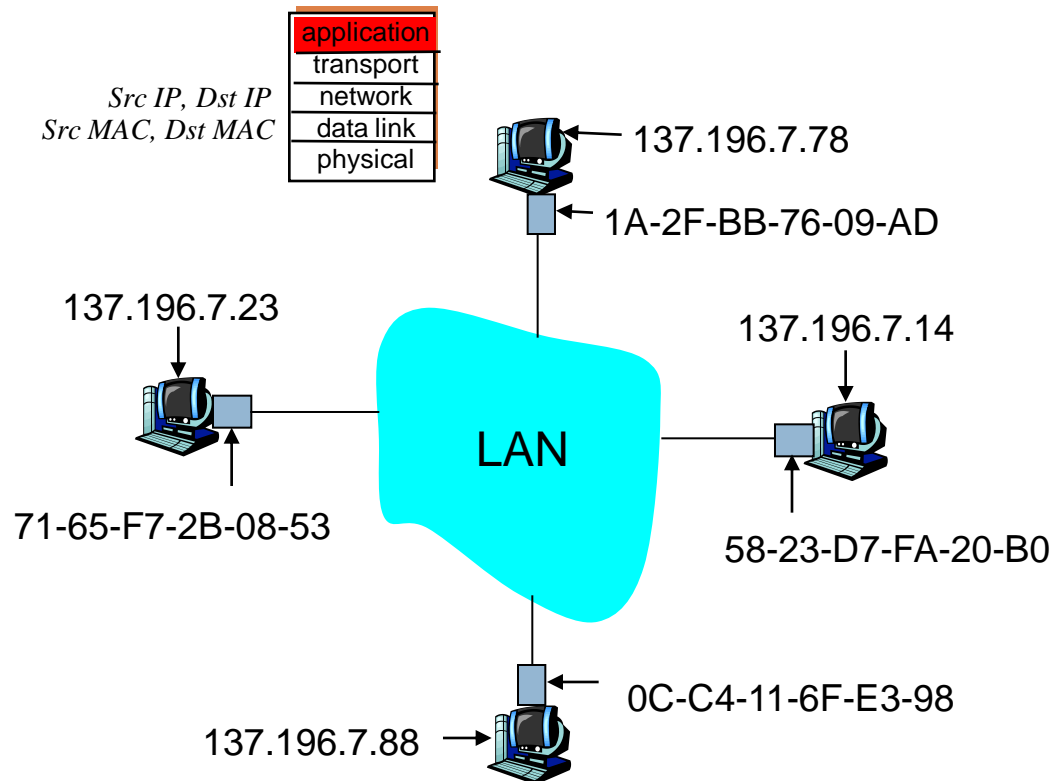
## ○ Đánh giá:

- Có thể thiết lập độ ưu tiên
- Tốn chi phí
- Việc truyền DL của 1 thiết bị tùy thuộc vào thiết bị dò chọn

# NỘI DUNG

- Giới thiệu
- Kỹ thuật phát hiện và sửa lỗi
- Điều khiển truy cập đường truyền
- ARP
- Ethernet

# ARP - 1

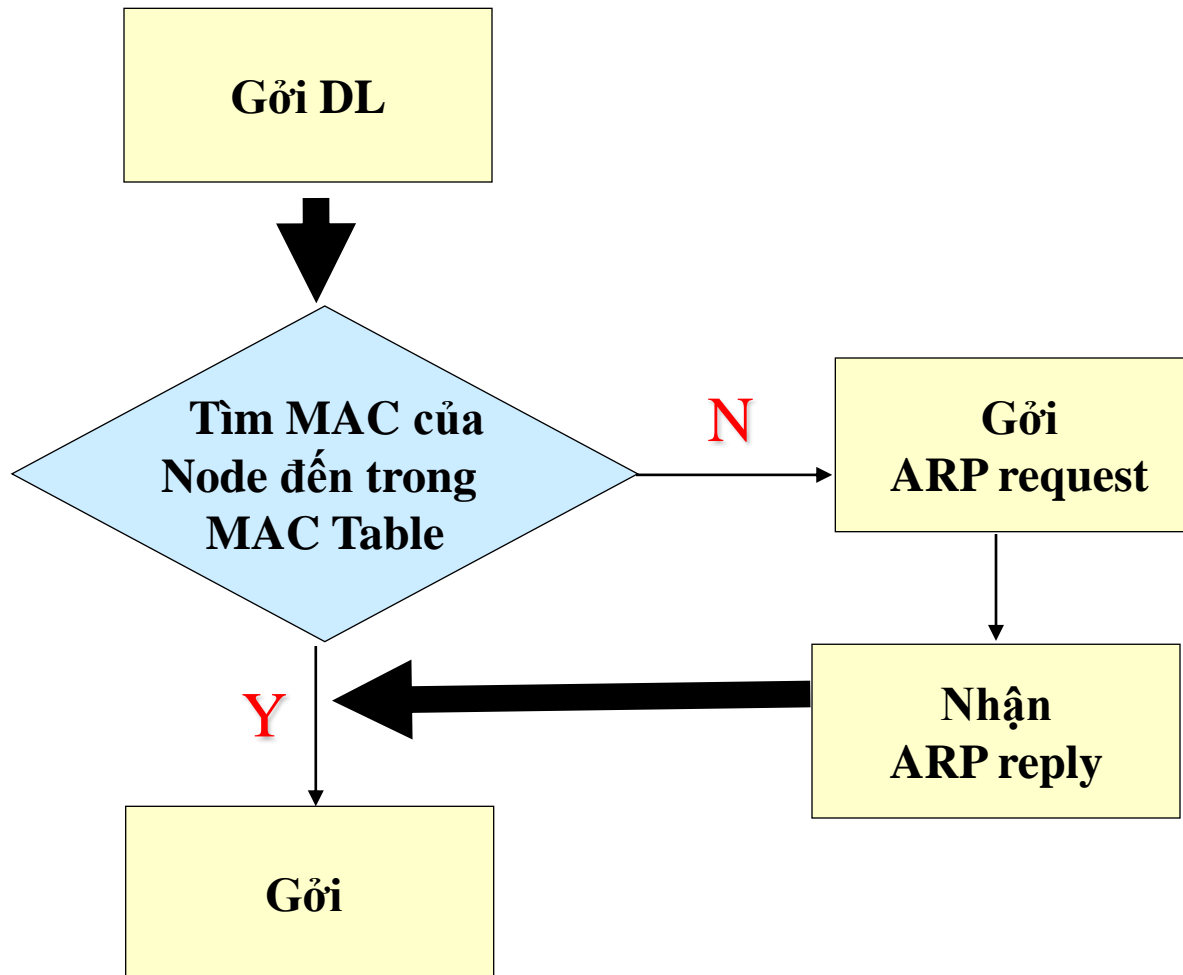


IP → ??? → MAC

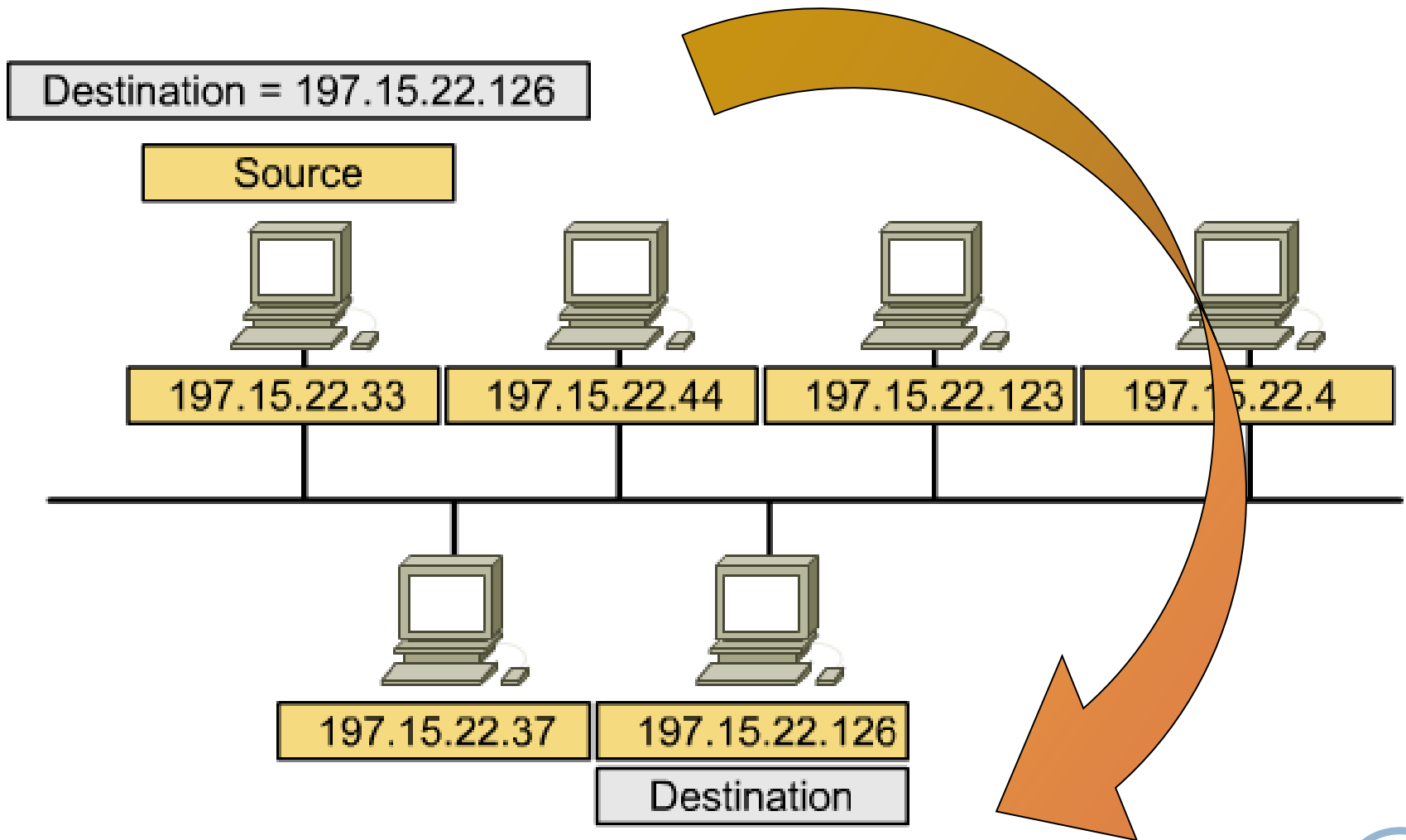
# ARP - 2

- ARP (Address Resolution Protocol)
  - Phân giải từ địa chỉ IP thành địa chỉ MAC
  - Chỉ phân giải trong cùng đường mạng
  - Sử dụng ARP table:
    - IP
    - MAC
    - TTL :thời gian sống của record
    - Lưu trong RAM

# ARP – CƠ CHẾ HOẠT ĐỘNG

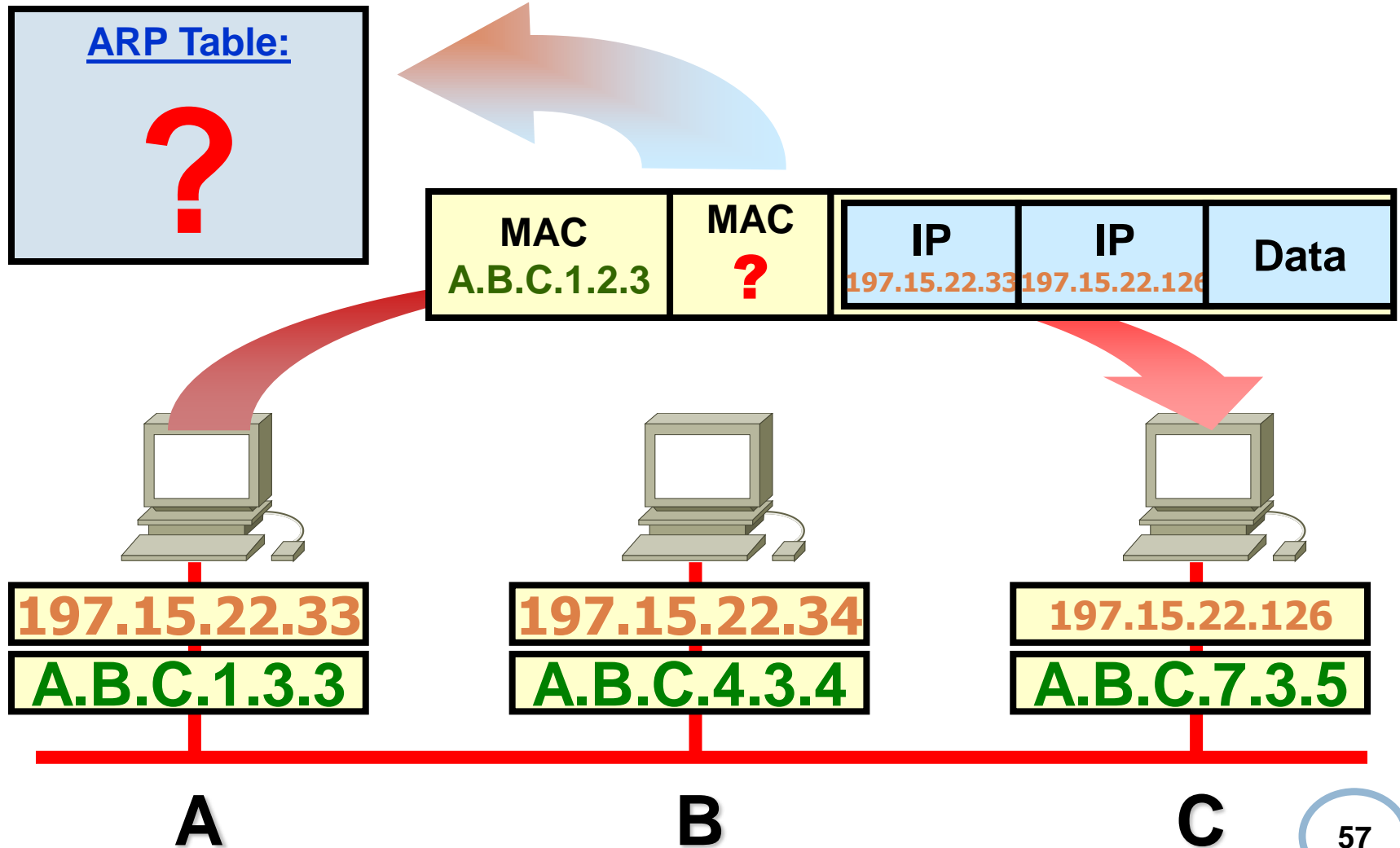


# ARP – MINH HỌA - 1

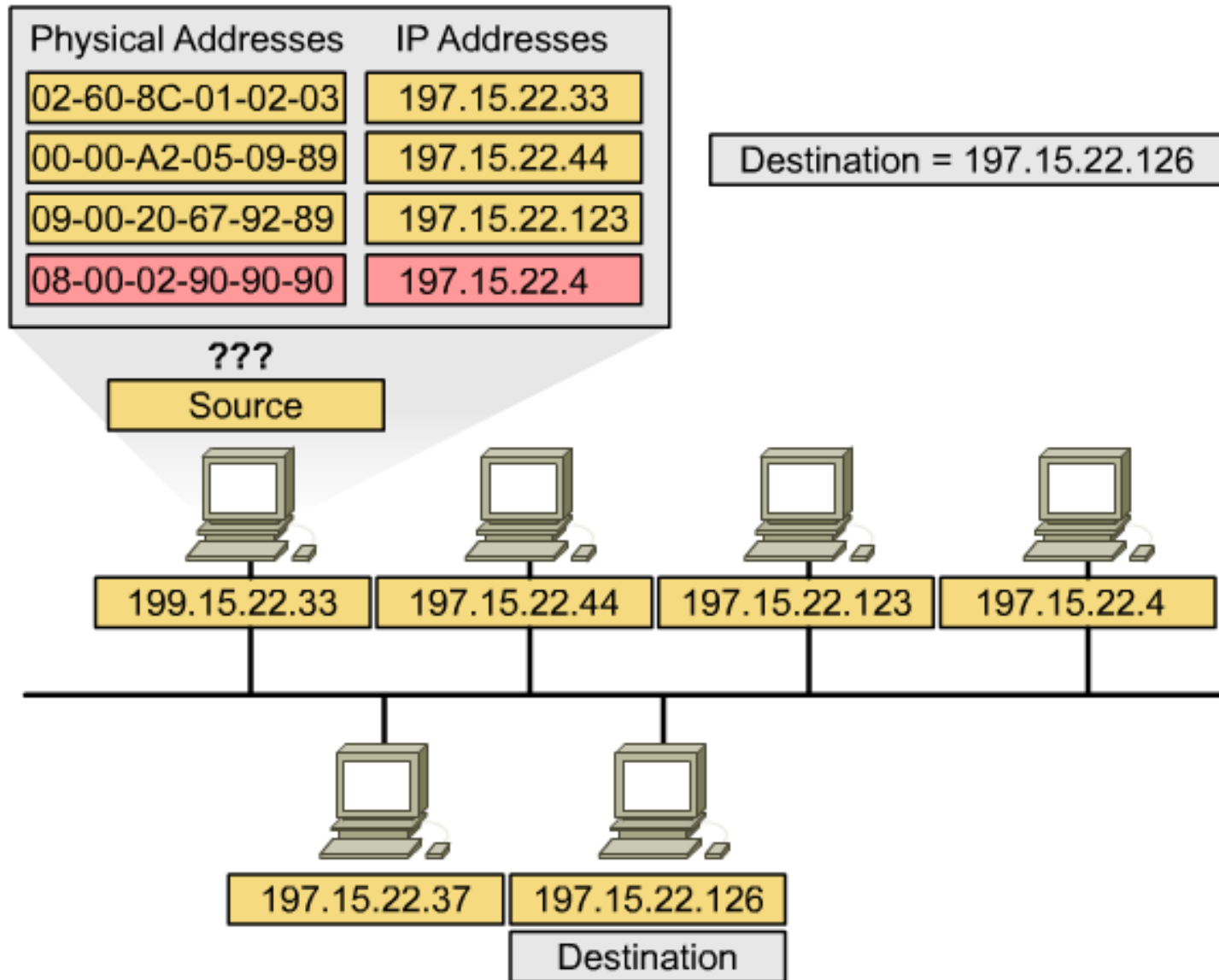




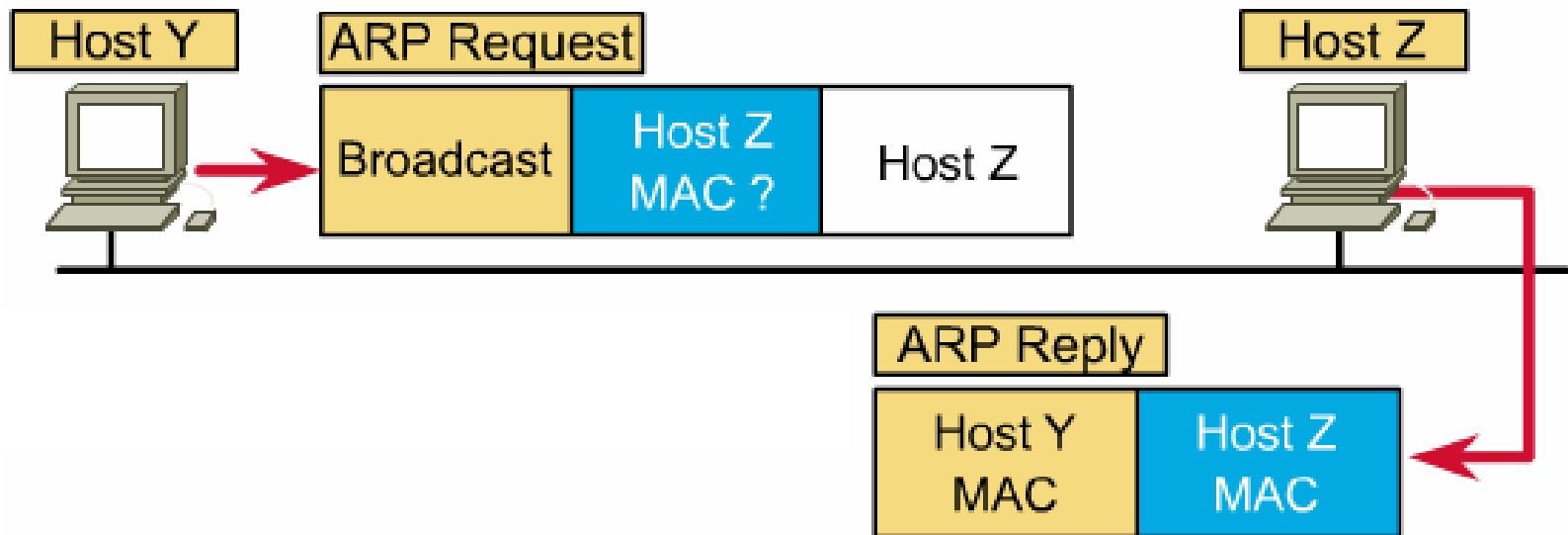
# ARP – MINH HỌA - 2



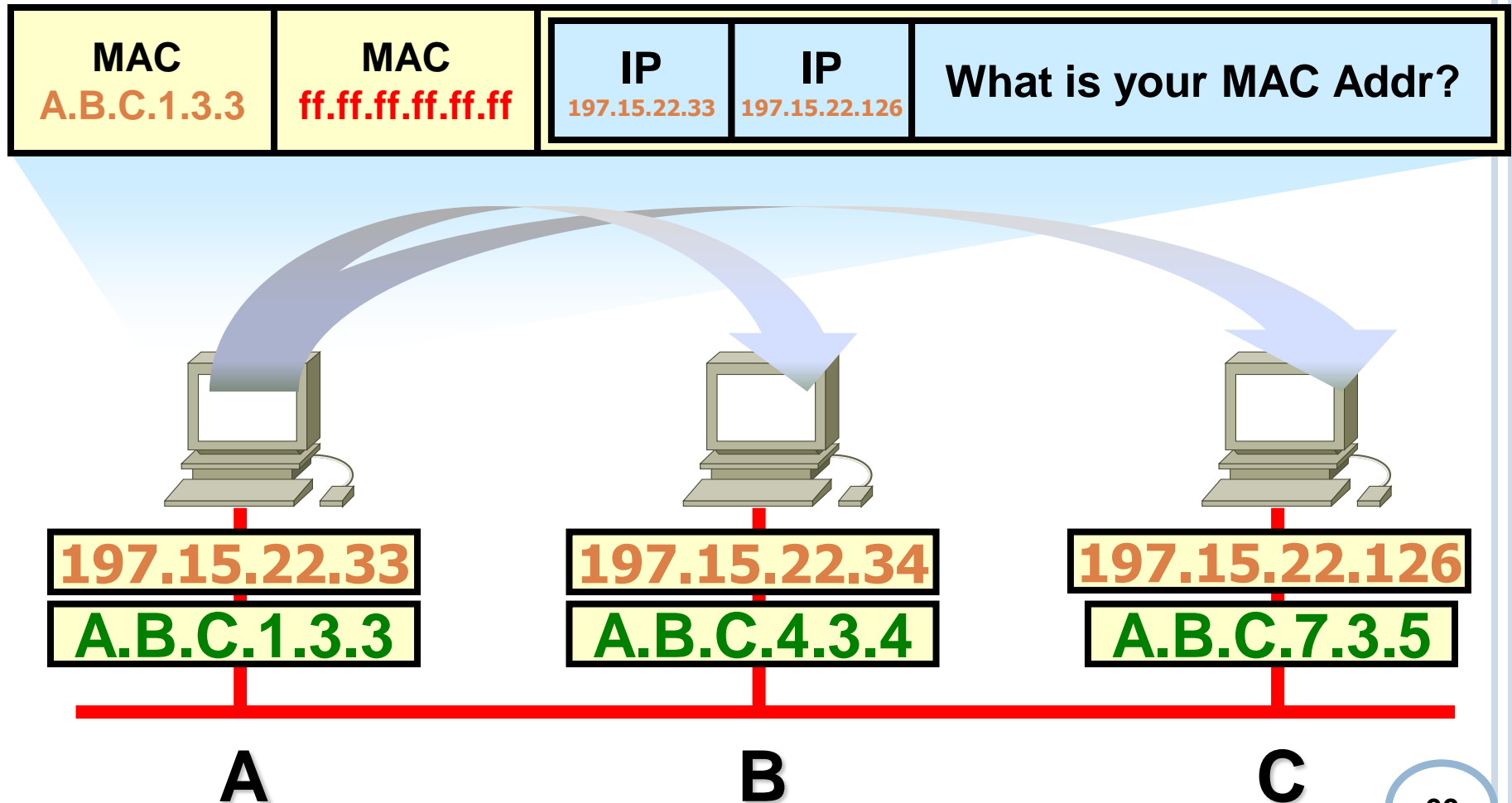
# ARP – MINH HỌA - 3



# ARP – MINH HỌA - 4

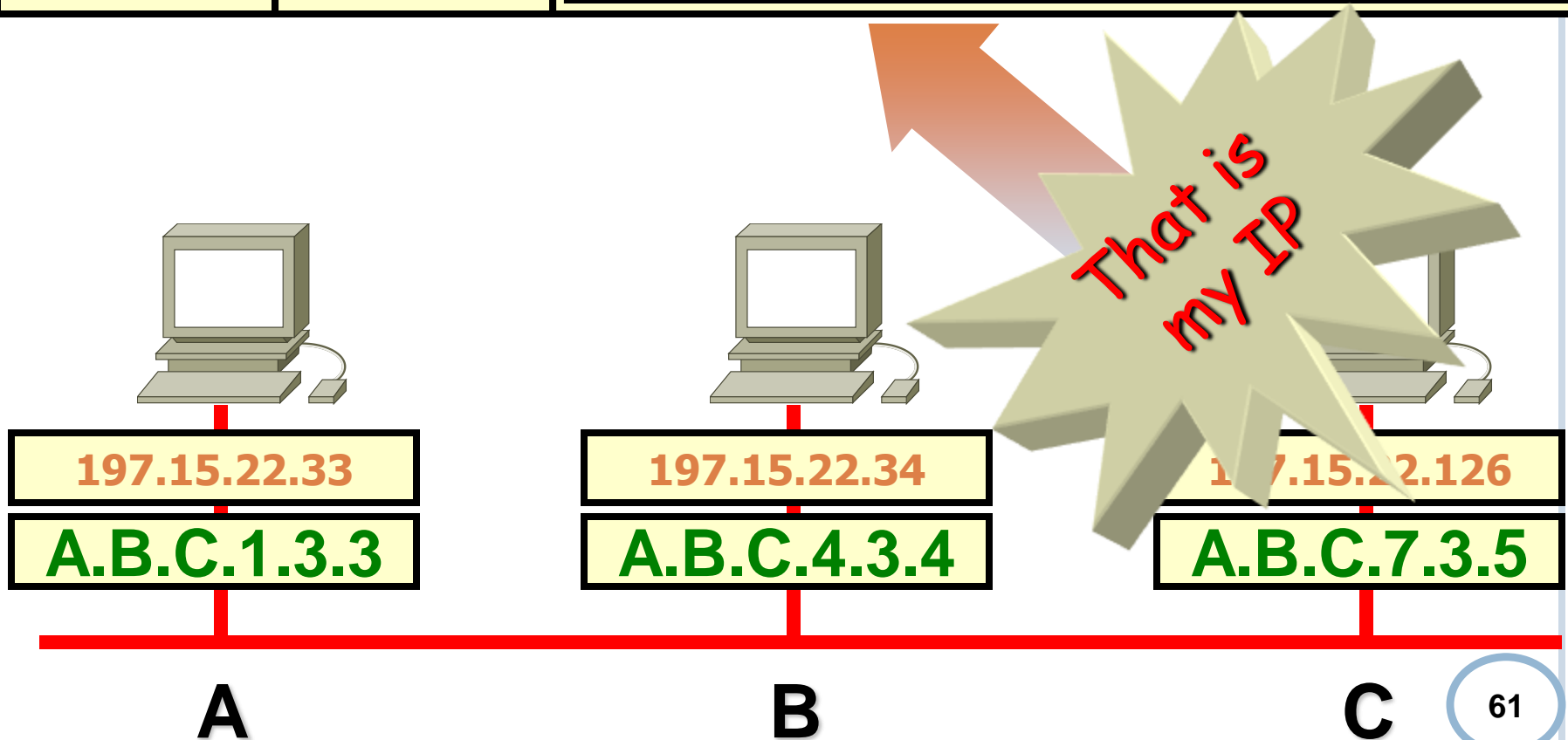


# ARP – REQUEST

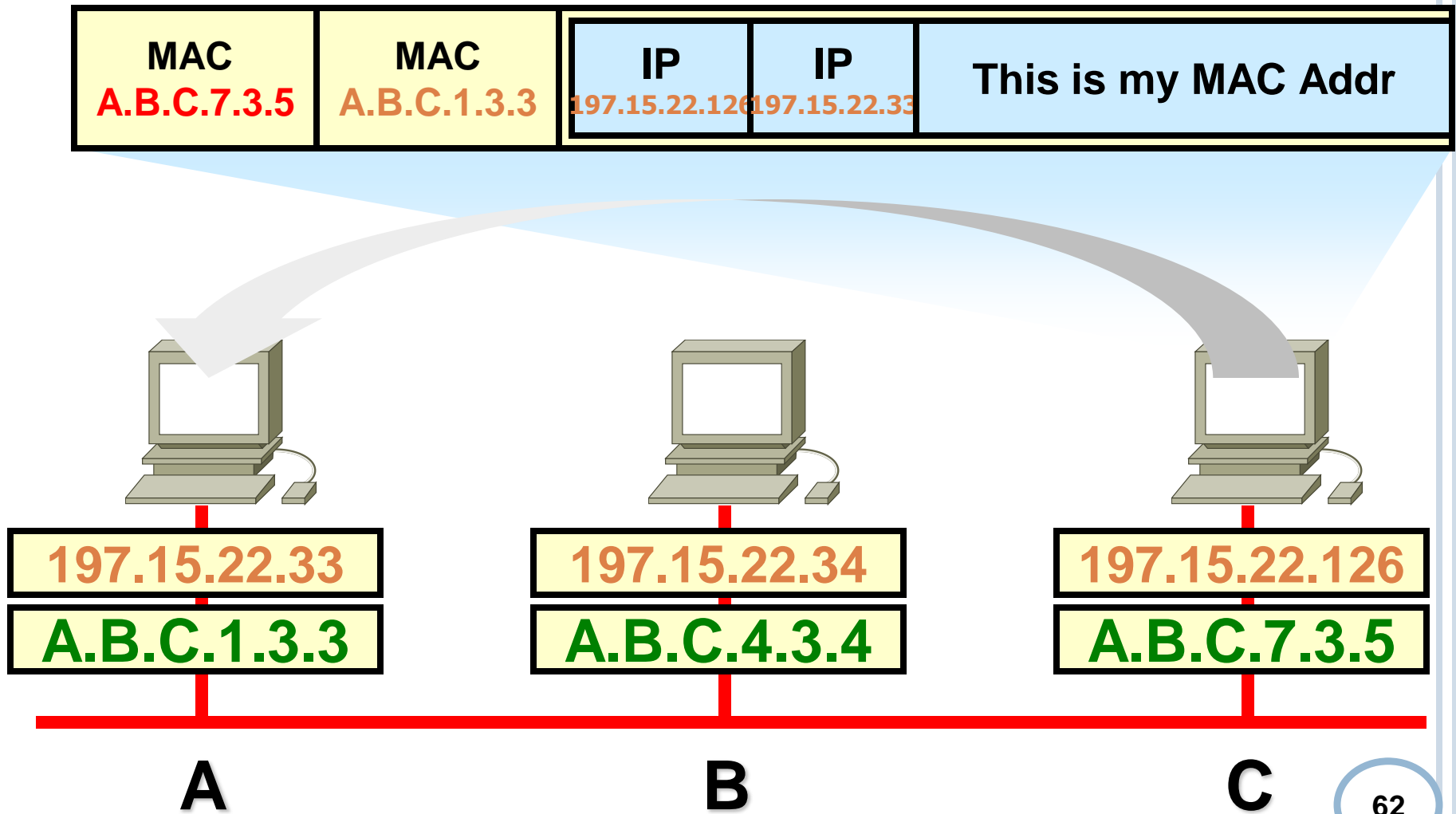


# ARP - CHECKING

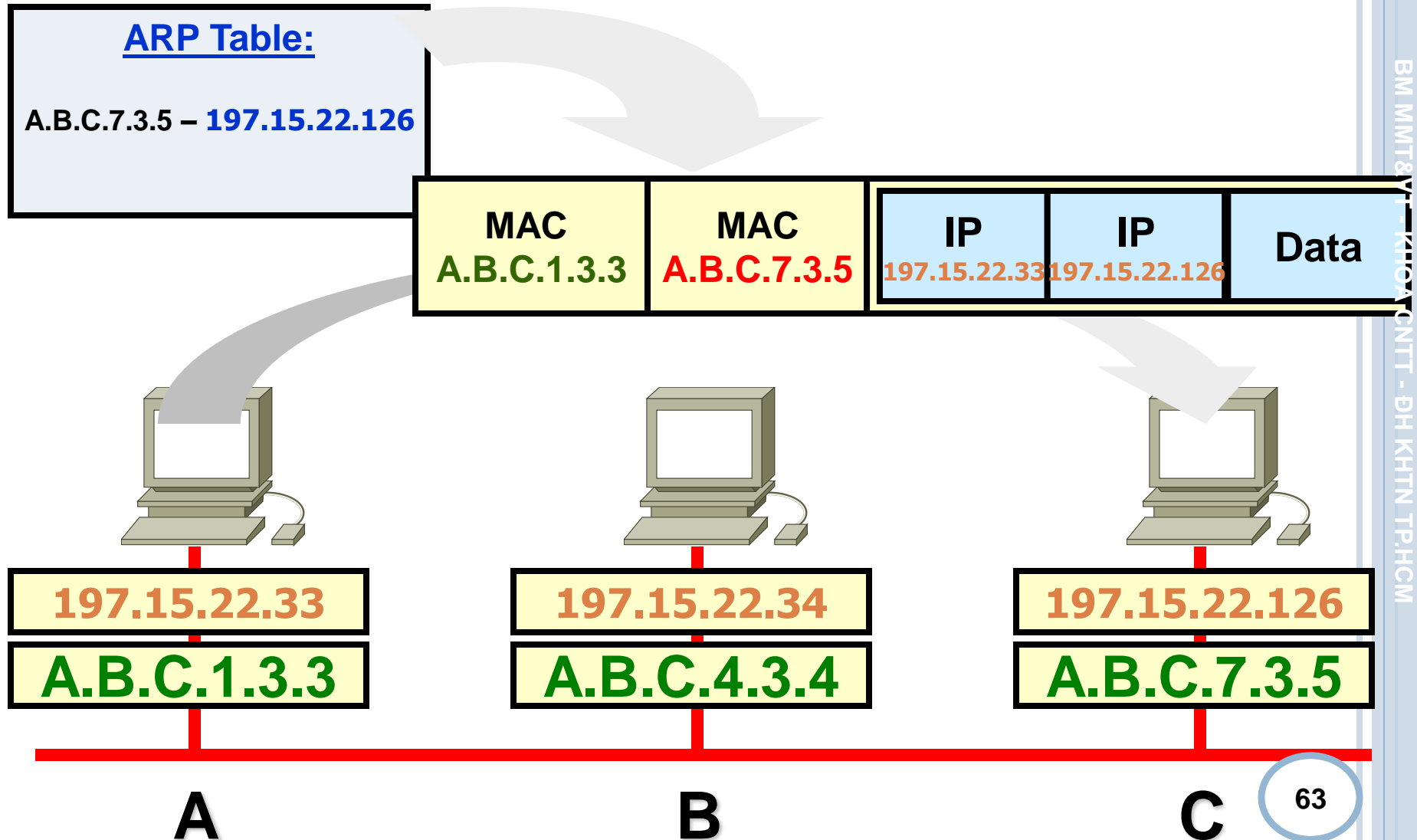
MAC A.B.C.1.3.3	MAC ff.ff.ff.ff.ff.ff	IP 197.15.22.33	IP 197.15.22.126	What is your MAC Addr?
--------------------	--------------------------	--------------------	---------------------	------------------------



# ARP - REPLY



# ARP - CACHING



# NỘI DUNG

- Giới thiệu
- Kỹ thuật phát hiện và sửa lỗi
- Điều khiển truy cập đường truyền
- ARP
- Ethernet

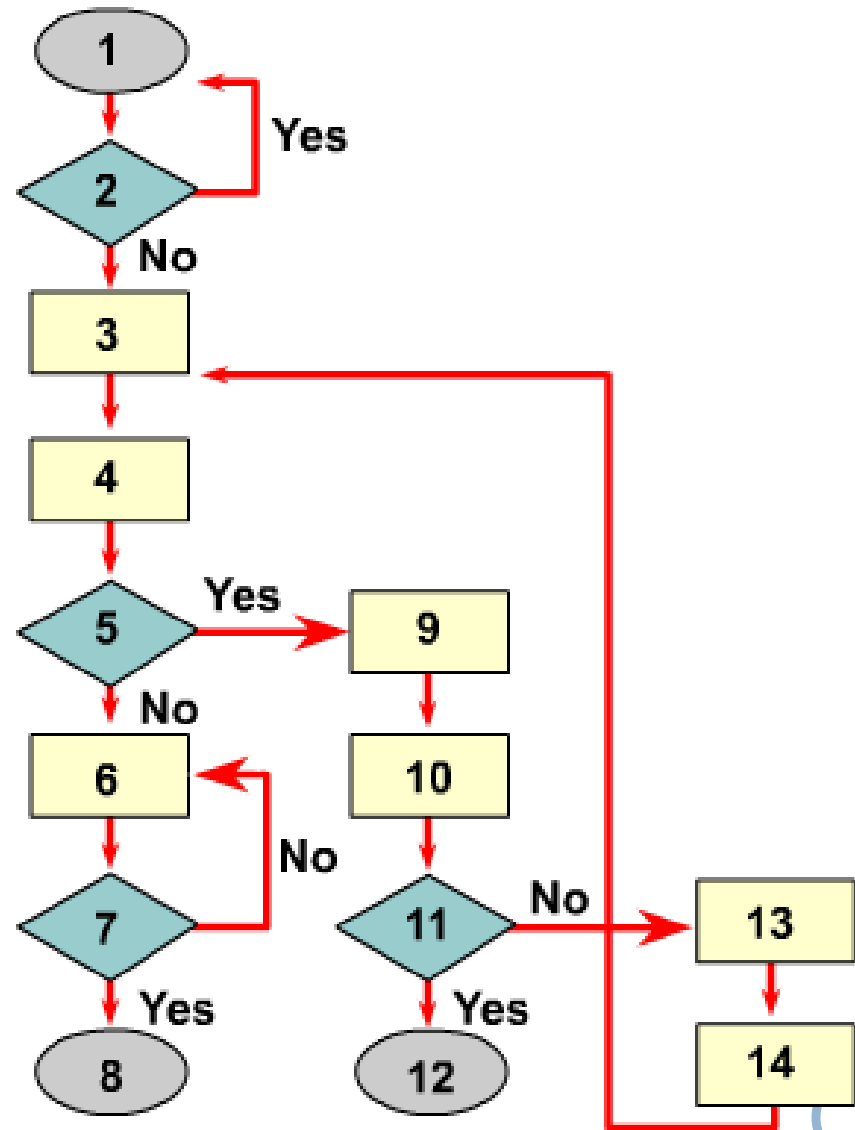


# ETHERNET - 1

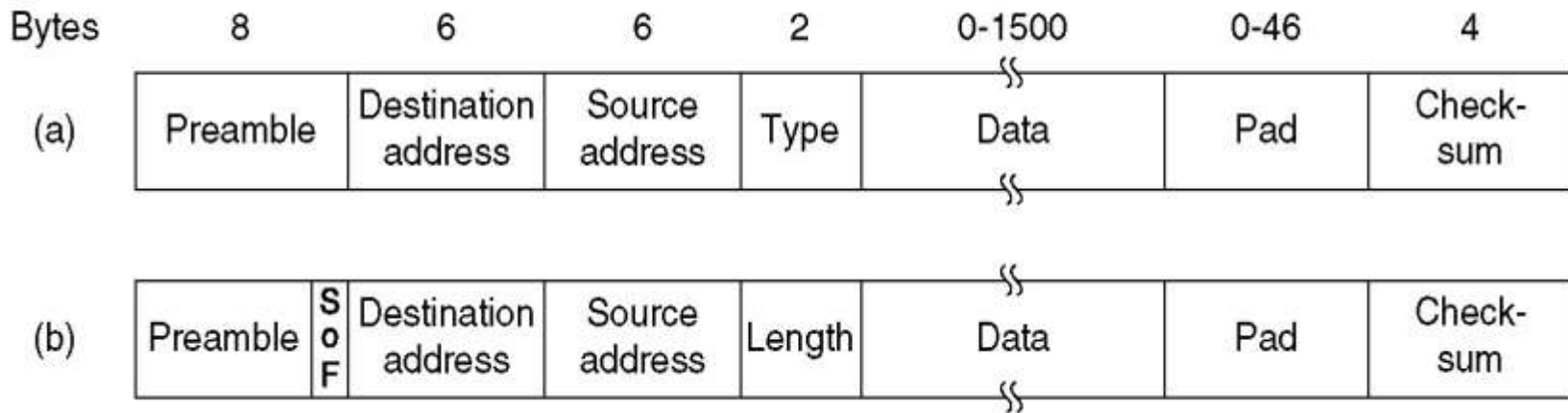
- Là 1 kỹ thuật (technology) mạng LAN có dây
  - Là 1 kỹ thuật mạng LAN đầu tiên
  - Chuẩn 802.3
  - Hoạt động tầng Data Link và Physical
  - Tốc độ: 10 Mbps – 10 Gbps
  - Đồ hình mạng:
    - Bus
    - Star
  - Giao thức tầng MAC: CSMA/CD
  - Đơn giản và rẻ hơn mạng Token Ring LAN, ATM

# CSMA/CD – QUÁ TRÌNH TRUYỀN DỮ LIỆU

1. Host wants to transmit
2. Is carrier sensed?
3. Assemble frame
4. Start transmitting
5. Is a collision detected?
6. Keep transmitting
7. Is the transmission done?
8. Transmission completed
9. Broadcast jam signal
10. Attempts = Attempts + 1
11. Attempts > Too many?
12. Too many collisions; abort transmission
13. Algorithm calculates backoff
14. Wait for t microseconds



# ETHERNET – CẤU TRÚC FRAME



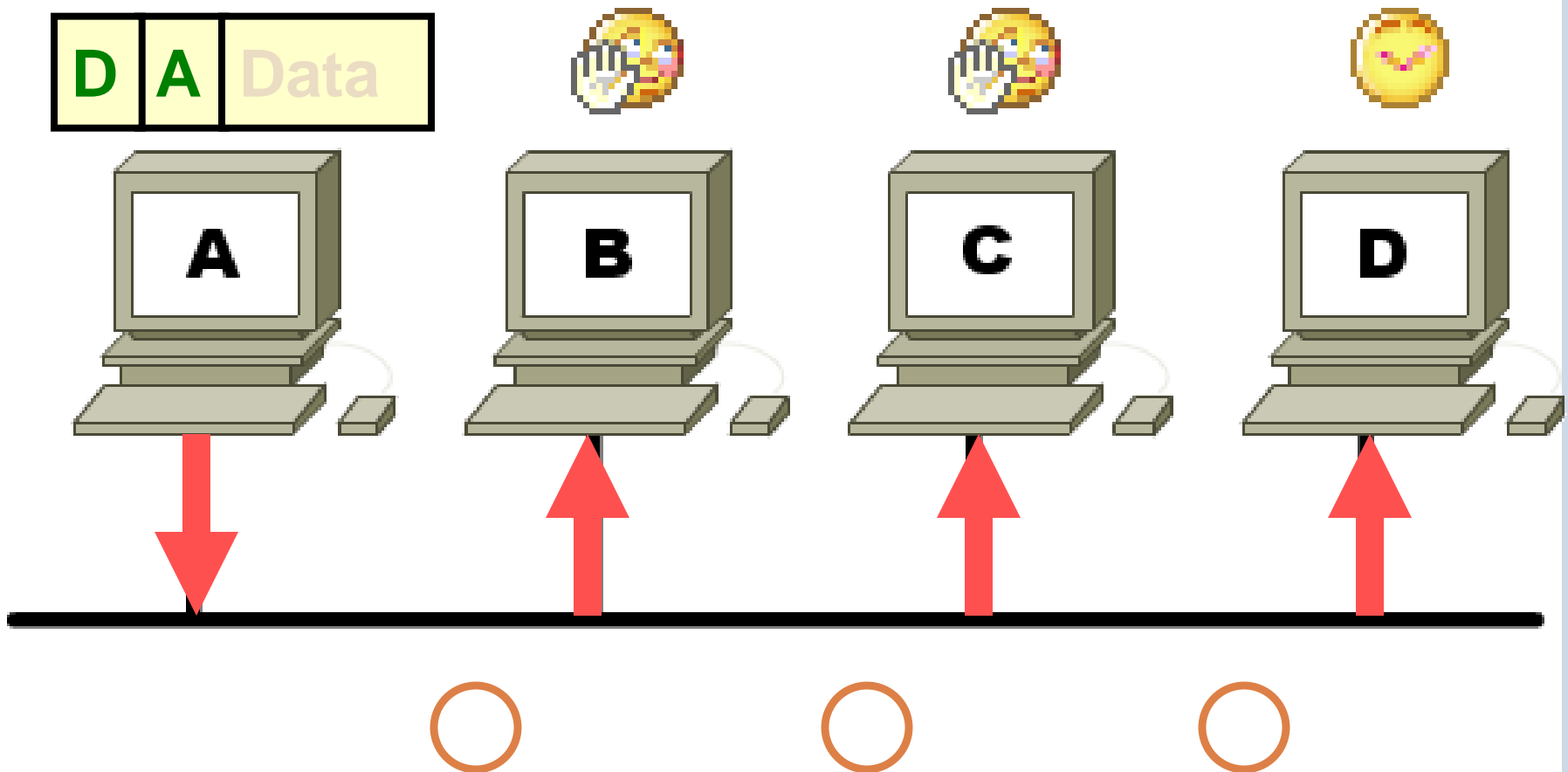
**a) earlier Ethernet frames - b) 802.3 frames**

- Preamble (8 bytes)
  - Đồng bộ đồng hồ bên gửi và bên nhận (10101010)
  - Start of Frame (SOF): báo hiệu bắt đầu frame (10101011)
- Dest. Addr (6 bytes)
  - địa chỉ MAC của card mạng nhận gói tin tiếp theo
- Src. Addr (6 bytes)
  - địa chỉ MAC của card mạng gửi gói tin
- Type (2 bytes)
  - Giao thức sử dụng ở tầng trên
- CRC: dùng để kiểm tra lỗi

# ETHERNET – TRƯỜNG TYPE

<b>EtherType</b>	<b>Protocol</b>
0x0800	Internet Protocol, Version 4 (IPv4)
0x0806	Address Resolution Protocol (ARP)
0x8035	Reverse Address Resolution Protocol (RARP)
0x809b	AppleTalk (Ethertalk)
0x80f3	AppleTalk Address Resolution Protocol (AARP)
0x8100	IEEE 802.1Q-tagged frame
0x8137	Novell IPX (alt)
0x8138	Novell
0x86DD	Internet Protocol, Version 6 (IPv6)
0x8847	MPLS unicast
0x8848	MPLS multicast

# ETHERNET – MINH HOẠ



# ETHERNET – CÁC CÔNG NGHỆ MẠNG

- 10Base2
- 10Base5
- 10BaseT
- 100BaseTX
- 100BaseFX
- Gigabit Ethernet



# ETHERNET – CHUẨN 10MBPS

Standard	Topology	Medium	Maximum cable length	Transport
10BASE5	Bus	Thick coaxial cable	500m	Half-duplex
10BASE2	Bus	Thin coaxial cable	185m	Half-duplex
10BASE-T	Star	CAT3 UTP	100m	Half or Full-duplex

# ETHERNET – CHUẨN 100MBPS

Standard	Medium	Maximum cable length
100BASE-TX	CAT5 UTP	100m
100BASE-FX	Multi-mode fibre (MMF) 62.5/125	412m



# ETHERNET – CHUẨN GIGABIT

Standard	Medium	Maximum cable length
1000BASE-SX	Fiber optics	550 m
1000BASE-LX	Fiber optics	5000 m
1000BASE-CX	STP	25 m
1000BASE-T	Cat 5 UTP	100 m

# TÀI LIỆU THAM KHẢO

- Slide của J.F Kurose and K.W. Ross về Computer Networking: A Top Down Approach
- Slide CCNA, version 3.0, Cisco

# CDMA - 2

