Chương 5: Tầng Liên kết dữ liệu

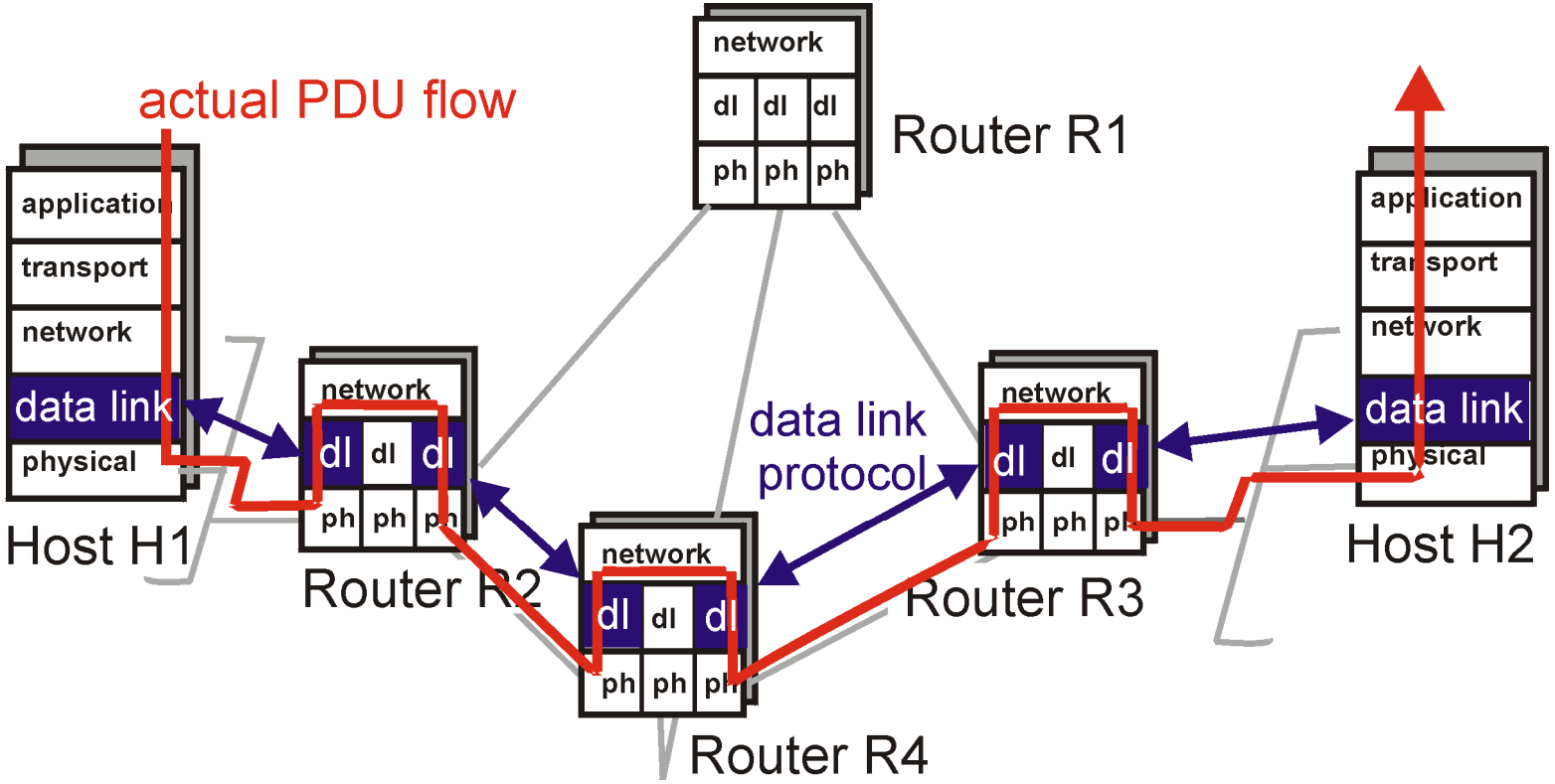
Mục tiêu:

- Các nguyên lý cung cấp dịch vụ của tầng Liên kết dữ liệu:
  - Phát hiện và Sửa lỗi
  - Chia sẻ kênh truyền dùng chung: đa truy cập
  - Địa chỉ tầng link
  - Truyền tin cậy, Điều khiển lưu lượng: ***đã học !***
- Cài đặt trên các công nghệ Liên kết dữ liệu khác nhau (rất nhiều)

Sẽ học gì:

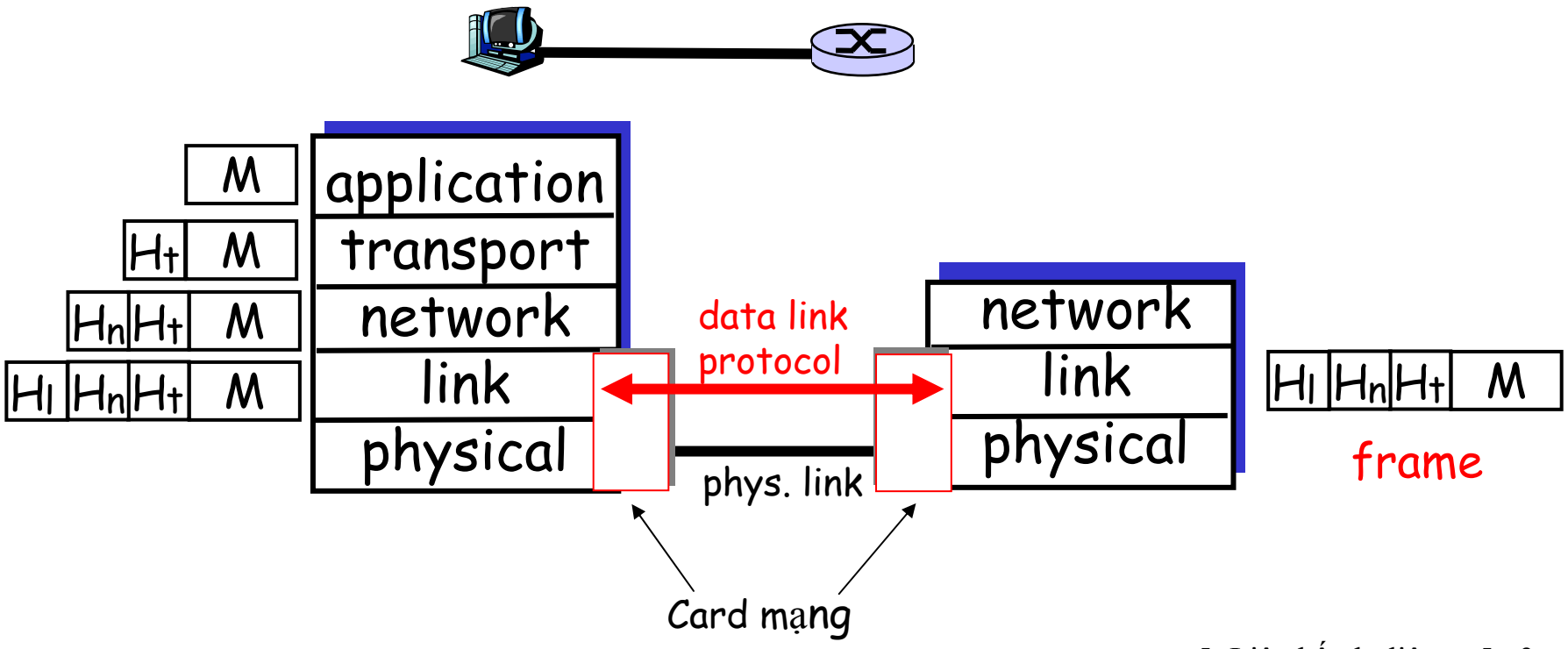
- Các dịch vụ ở tầng Liên kết dữ liệu
- Phát hiện, Sửa lỗi
- Đa truy cập và mạng LAN
- Địa chỉ ở tầng Liên kết dữ liệu và ARP
- Một vài công nghệ Liên kết dữ liệu cụ thể:
  - Ethernet
  - hubs, bridges, switches
  - IEEE 802.11 LANs
  - PPP

Liên kết dữ liệu: Vị trí trong Mô hình



Liên kết dữ liệu: Bối cảnh

- Hai thiết bị có **kết nối về mặt Vật lý**:
  - host-router, router-router, host-host
- Đơn vị trao đổi dữ liệu : ***frame***



Liên kết dữ liệu : Dịch vụ

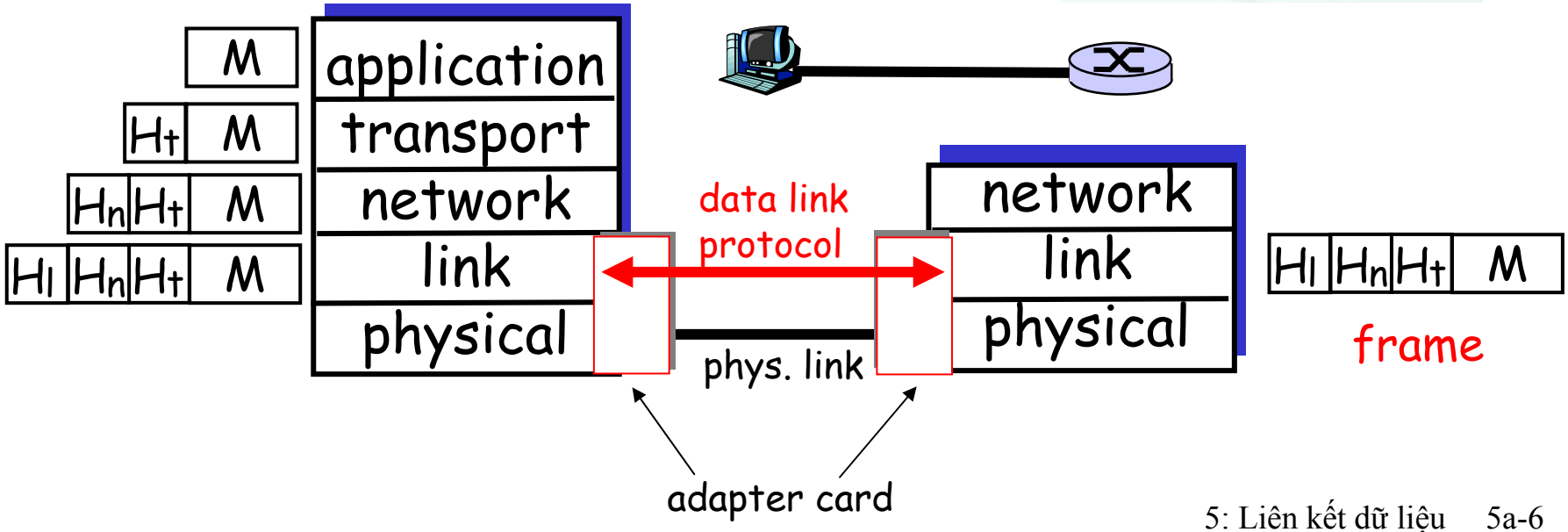
- **Tạo frame, Truy cập môi trường:**
  - Đặt các datagram trong các frame, bổ sung thêm header, trailer
  - Nếu môi trường truyền dùng chung, cài đặt chức năng đa truy cập
  - "địa chỉ Vật lý" trong tiêu đề của frame xác định Địa chỉ Gửi/ Nhận
    - Khác địa chỉ IP !
- **Truyền tin cậy giữa hai thiết bị có kết nối Vật lý trực tiếp:**
  - Nguyên lý đã được giải quyết (chương 3)!
  - Ít khi được dùng trên kênh truyền có tỷ lệ lỗi thấp (cáp quang, một số cáp đồng trục)
  - Đường truyền không dây: Tỷ lệ lỗi cao
    - Vấn đề: Tại sao đặt Tính tin cậy ở cả hai tầng ?

Liên kết dữ liệu : Dịch vụ (tiếp)

- **Điều khiển lưu lượng:**
  - Phù hợp Tốc độ Gửi và Nhận
- **Phát hiện lỗi:**
  - Lỗi do nhiễu.
  - Phía Nhận xác định được có lỗi:
    - Yêu cầu bên Gửi truyền lại hoặc loại bỏ Frame
- **Sửa lỗi:**
  - Phía Nhận xác định *và sửa* được các bit bị lỗi mà không yêu cầu truyền lại

Liên kết dữ liệu: Cài đặt ở đâu

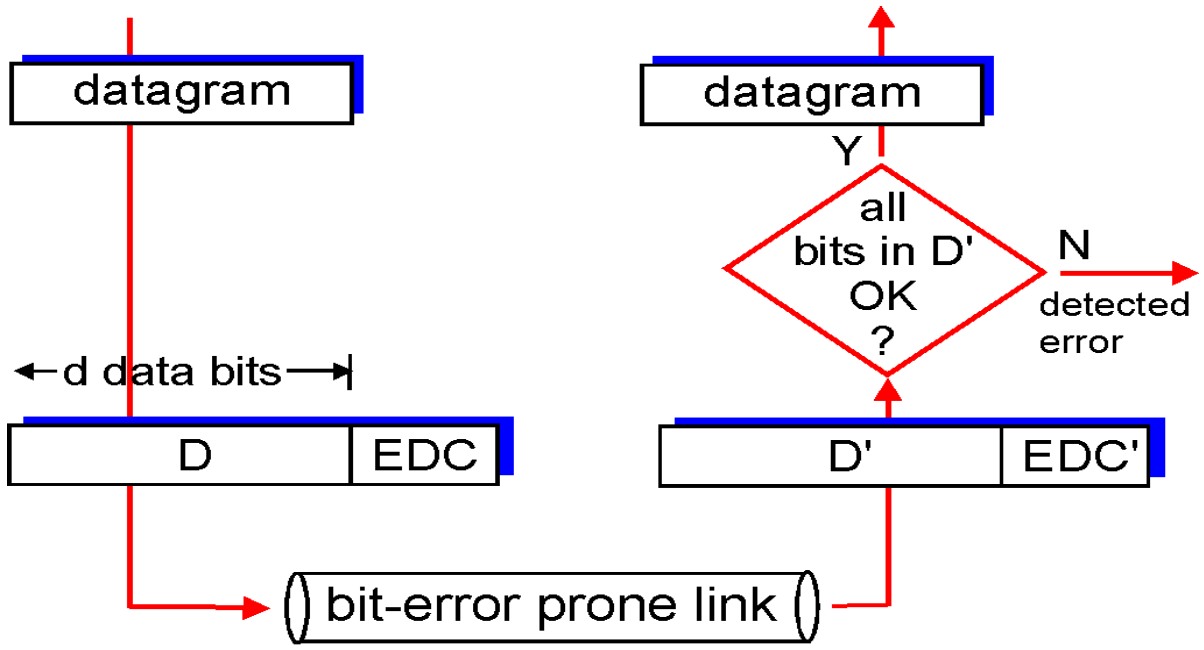
- Cài đặt trên các "adapter"
  - Ví dụ PCMCIA card, Ethernet card
  - Thường có: RAM, DSP chips, host bus interface, và link interface



Phát hiện Lỗi

EDC = Error Detection and Correction bit (Dư thừa)
D = Dữ liệu cần được bảo vệ (có thể thêm phần Tiêu đề)

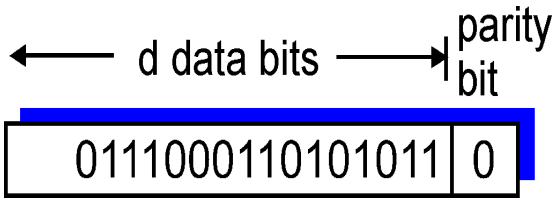
- Phát hiện lỗi Không hoàn toàn đáng tin cậy !
  - Giao thức có thể để "lọt" một số lỗi (hiếm khi)
  - Trường EDC lớn giúp Phát hiện và Sửa lỗi tốt hơn



Kiểm tra Chẵn lẻ

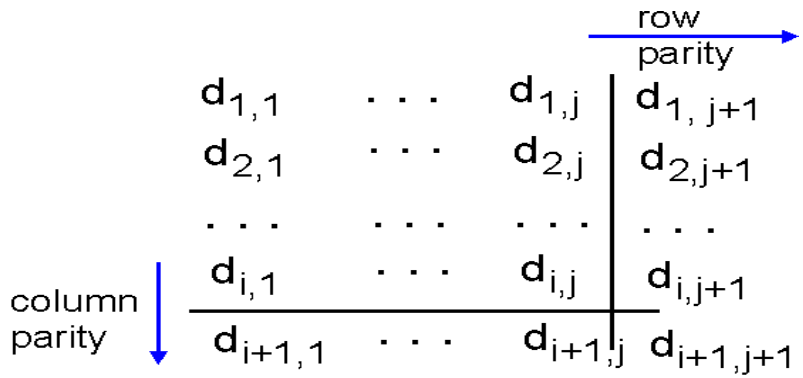
Một Bit Chẵn Lẻ:

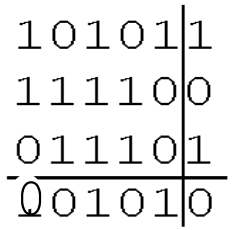
Phát hiện Một lỗi



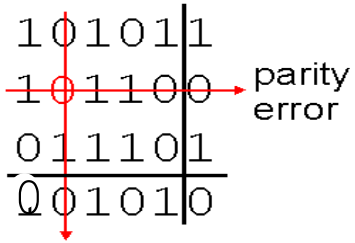
Bit Chẵn Lẻ hai chiều:

Phát hiện và Sửa được một Lỗi





no errors



correctable single bit error

Internet checksum

Mục tiêu: phát hiện "lỗi" (bit bị đổi) trong segment được truyền (chú ý: *chỉ* được sử dụng ở Tầng Giao vận)

Phía Gửi:

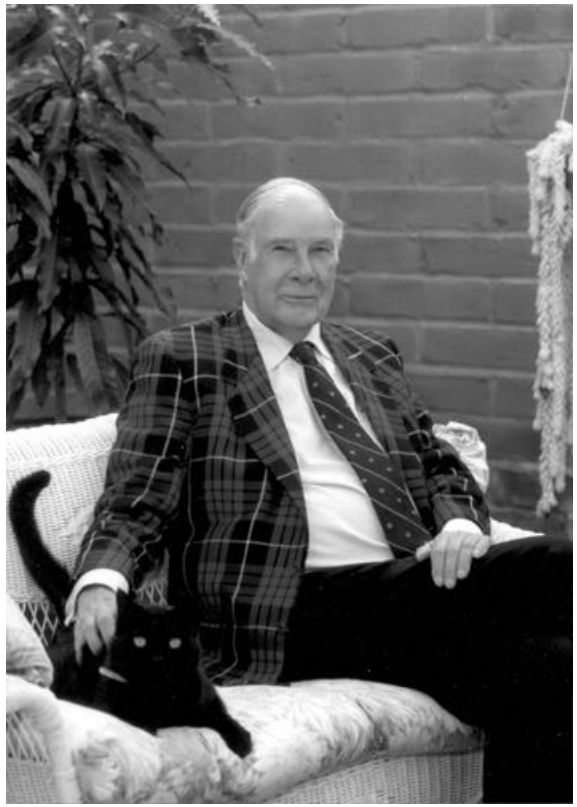
- Xem nội dung segment như các chuỗi số nguyên 16 bit.
- checksum: Tổng bù 1 của tất cả các từ
- Phía Gửi đặt giá trị checksum trong trường checksum của UDP

Phía Nhận:

- Tính checksum của segment nhận được
- Kiểm tra checksum vừa tính được với giá trị Trường checksum:
  - KHÔNG TRÙNG – Phát hiện có Lỗi
  - TRÙNG – Không phát hiện được Lỗi. ***Nhưng vẫn có thể có Lỗi?***

Mã sửa lỗi Hamming

- **Động lực** : Muốn có mã sửa lỗi cần ít dư thừa hơn kiểu mã ma trận chẵn lẻ hai chiều
- Mã Hamming : với log(M) bit dư thừa
  - Sửa tất cả lỗi một bit
  - Phát hiện các lỗi hai bit
- Đặt các bit chẵn lẻ kiểm tra xen kẽ

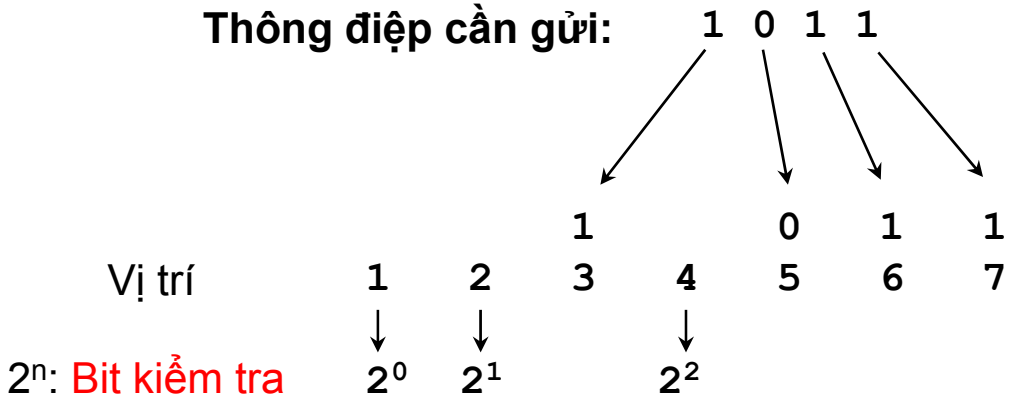


Richard W. Hamming

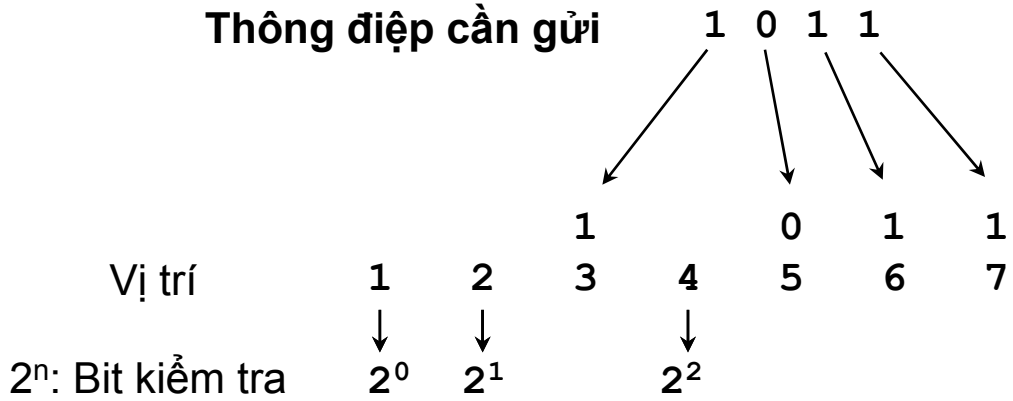
Xác định mã Hamming

- Thủ tục:
  - Đặt các bit dữ liệu thực sự (thông điệp) tại các vị trí Không phải là lũy thừa của hai.
  - Xây dựng bảng liệt kê biểu diễn nhị phân cho mỗi vị trí của bit dữ liệu
  - Tính giá trị các bit kiểm tra

Ví dụ về mã Hamming



Ví dụ về mã Hamming



Tính toán các bit kiểm tra:

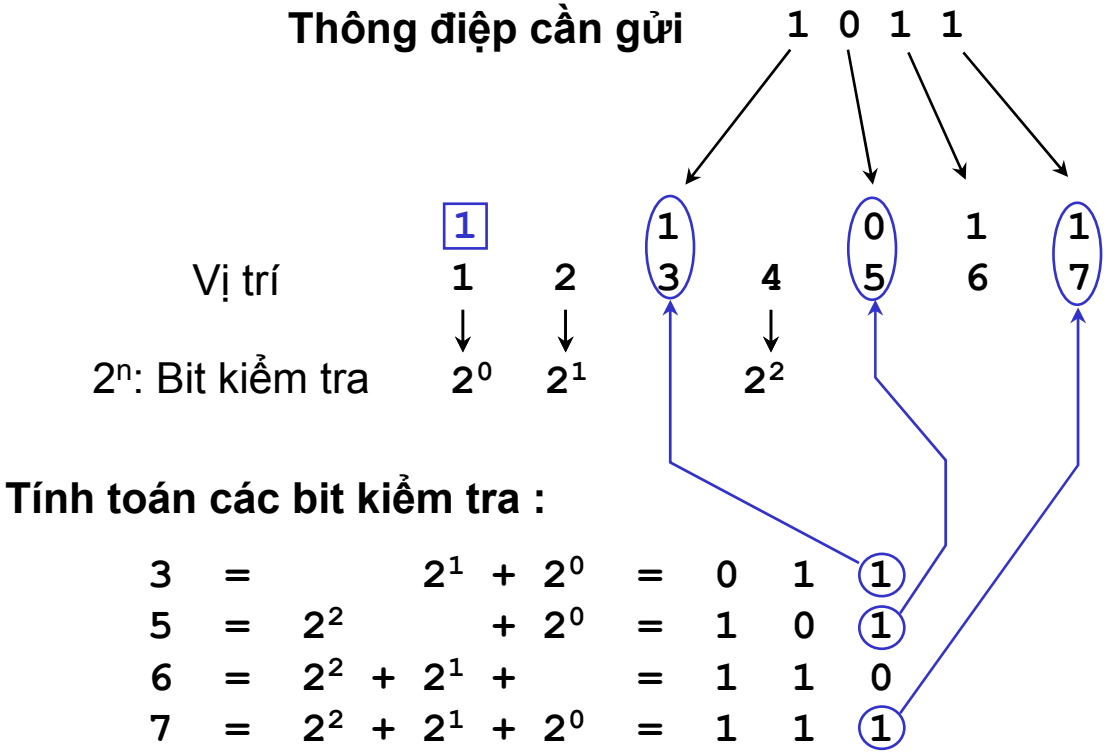
$3 = 2^1 + 2^0 = 0\ 1\ 1$

$5 = 2^2 + 2^0 = 1\ 0\ 1$

$6 = 2^2 + 2^1 = 1\ 1\ 0$

$7 = 2^2 + 2^1 + 2^0 = 1\ 1\ 1$

Ví dụ về mã Hamming



Bắt đầu từ vị trí 2^0 :

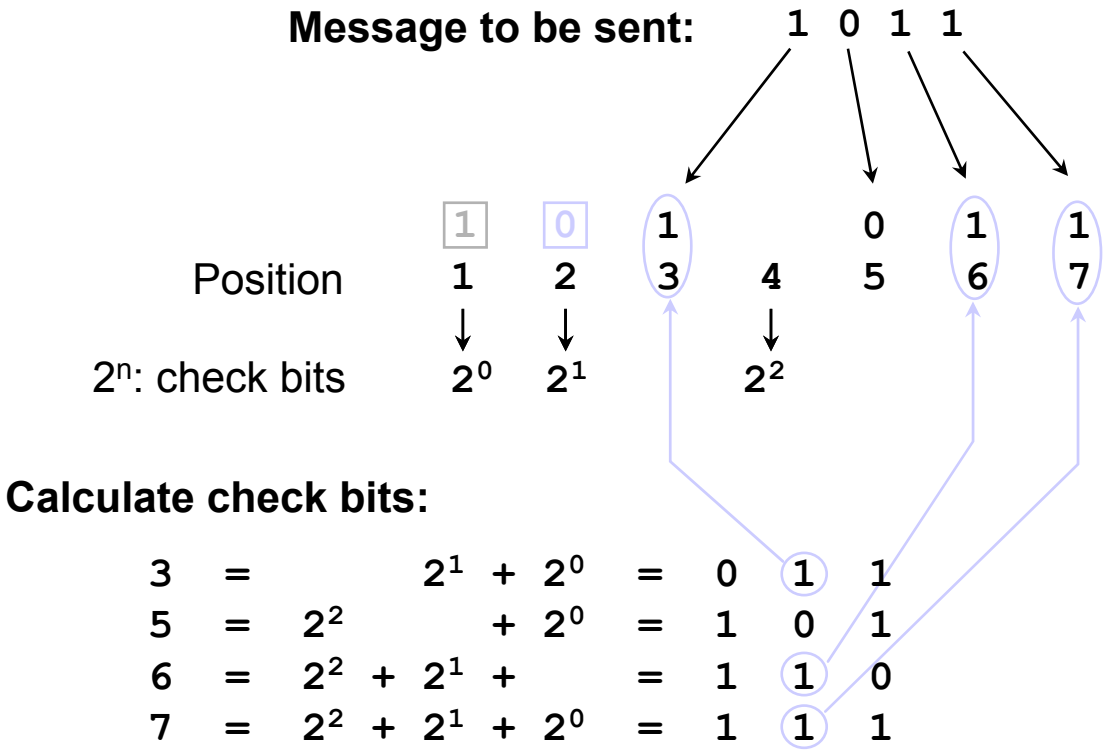
Kiểm tra tất cả các vị trí có giá trị 1 tại vị trí 2^0

Đếm số lượng số 1 trong các bit thông điệp tương ứng

Nếu CHẴN, đặt 1 ở vị trí bit kiểm tra 2^0 (Sử dụng bit chẵn lẻ lẻ)

Nếu LẺ, đặt số 0

Ví dụ về mã Hamming



Calculate check bits:

$3 = 2^1 + 2^0 = 0\ 1\ 1$

$5 = 2^2 + 2^0 = 1\ 0\ 1$

$6 = 2^2 + 2^1 = 1\ 1\ 0$

$7 = 2^2 + 2^1 + 2^0 = 1\ 1\ 1$

Bắt đầu từ vị trí 2^1 :

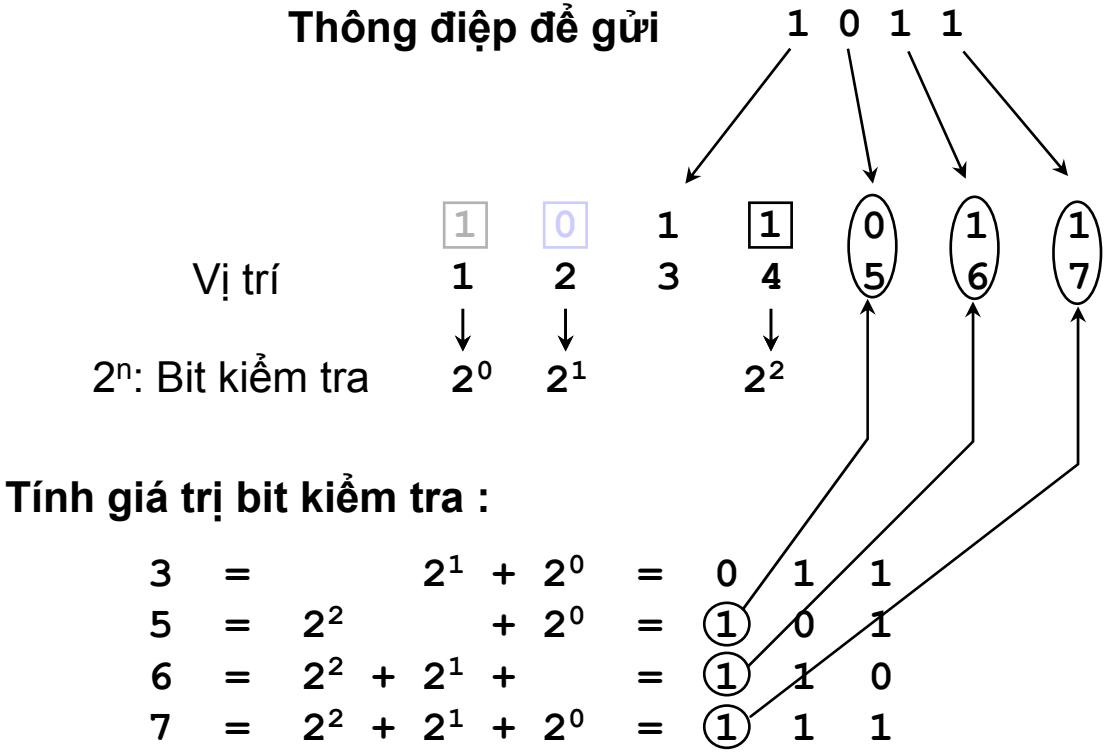
Kiểm tra tất cả các vị trí có giá trị 1 tại vị trí 2^1

Đếm số lượng số 1 trong các bit thông điệp tương ứng

Nếu CHẴN, đặt 1 ở vị trí bit kiểm tra 2^1 (Sử dụng bit chẵn lẻ lẻ)

Nếu LẺ, đặt số 0

Ví dụ về mã Hamming



Bắt đầu từ vị trí 2^2 :

Kiểm tra tất cả các vị trí có giá trị 1 tại vị trí 2^2

Đếm số lượng số 1 trong các bit thông điệp tương ứng

Nếu CHẴN, đặt 1 ở vị trí bit kiểm tra 2^2 (Sử dụng bit chẵn lẻ lẻ)

Nếu LẺ, đặt số 0

Ví dụ về mã Hamming

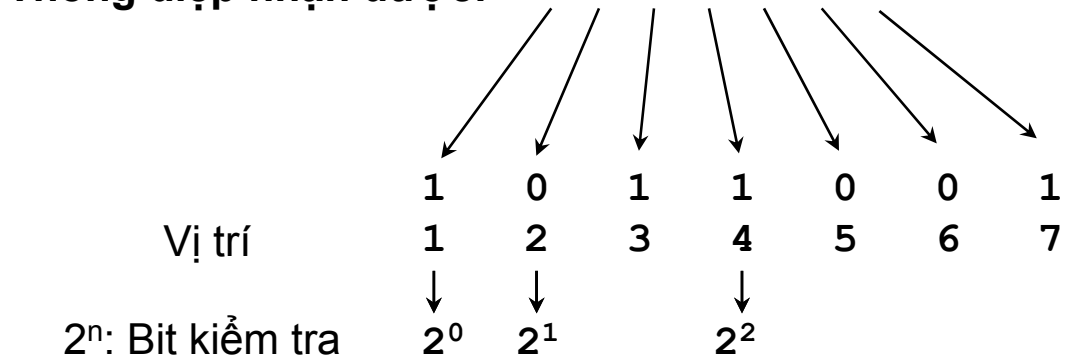
Thông điệp ban đầu = 1011

Thông điệp gửi đi = 1011011

Vậy làm thế nào để có thể xác định vị trí bit bị lỗi trong thông điệp bằng cách sử dụng mã Hamming ?

Sử dụng mã Hamming để sửa lỗi 1 bit

Thông điệp nhận được: 1 0 1 1 0 0 1



Tính các bit kiểm tra:

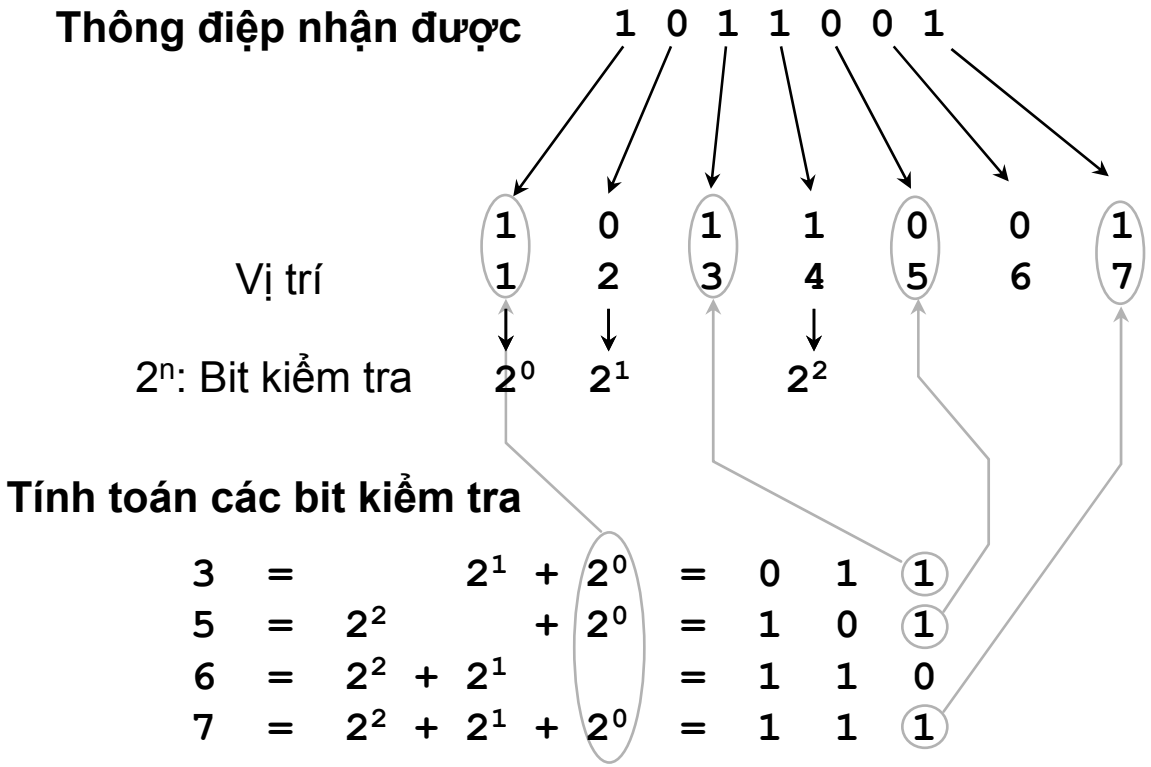
$3 = 2^1 + 2^0 = 0\ 1\ 1$

$5 = 2^2 + 2^0 = 1\ 0\ 1$

$6 = 2^2 + 2^1 = 1\ 1\ 0$

$7 = 2^2 + 2^1 + 2^0 = 1\ 1\ 1$

Sử dụng mã Hamming để Sửa lỗi 1 bit



Chẵn lẻ lẻ: Không có lỗi ở các bit 1, 3, 5, 7

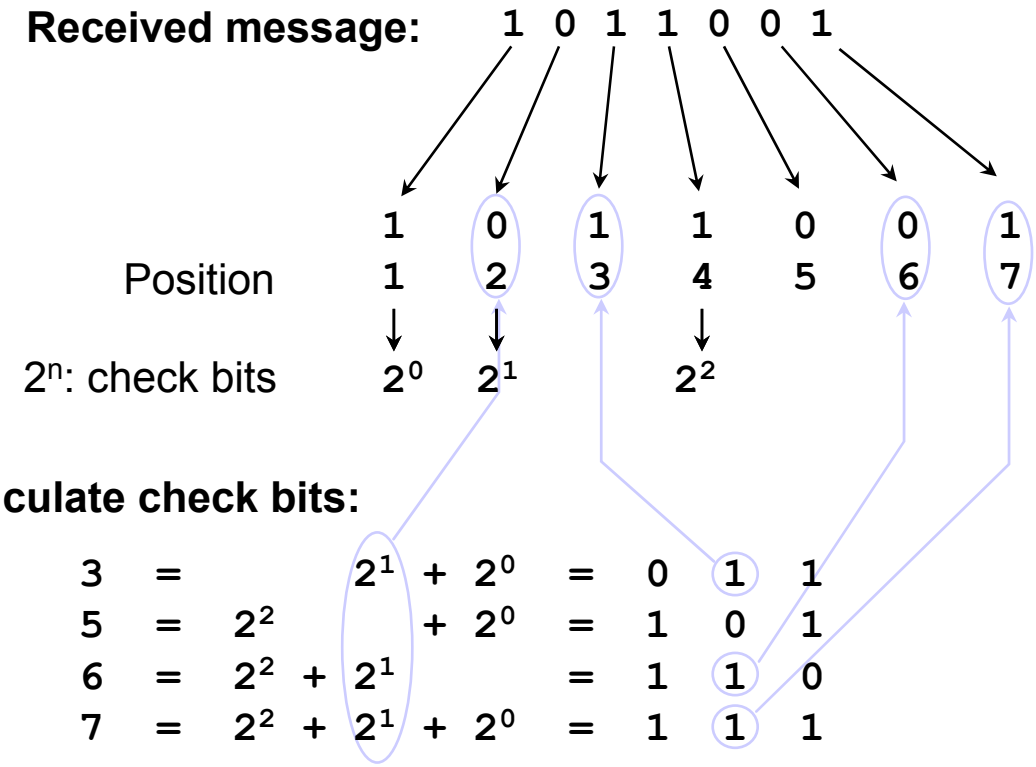
Bắt đầu từ vị trí 2^0:

Kiểm tra tất cả các vị trí có giá trị 1 tại 2^0

Đếm số số 1 trong tất cả các bit tương ứng của thông điệp và vị trí 2^0 và Giá trị chẵn lẻ của số đếm này.

Nếu số đếm này là một số chẵn, chắc chắn có một trong bốn bit được kiểm tra bị lỗi.

Sử dụng mã Hamming để Sửa lỗi 1 bit



Calculate check bits:

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

Chẵn lẻ Chẵn: LỖI trong các bit 2, 3, 6 hoặc 7!

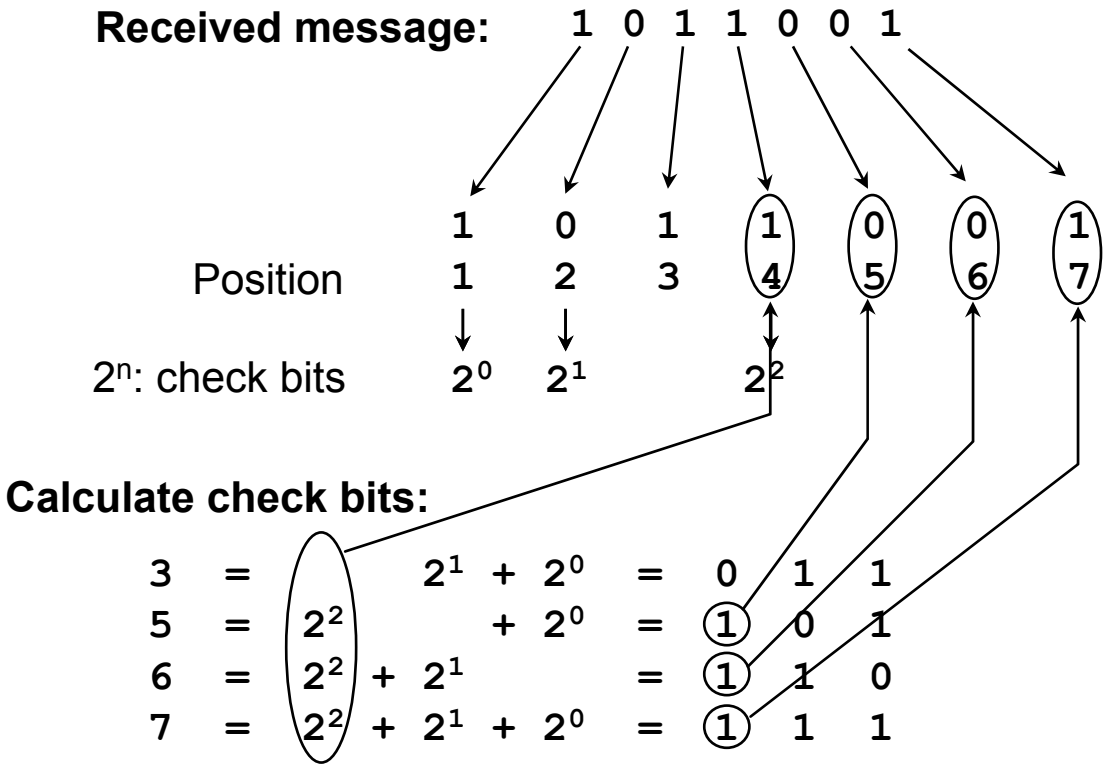
Tiếp tục với vị trí 2^1 :

Kiểm tra tất cả các vị trí có giá trị 1

Đếm số số 1 trong tất cả các bit tương ứng của thông điệp và vị trí 2^1 và Giá trị chẵn lẻ của số đếm này.

Nếu số đếm này là một số chẵn, chắc chắn có một trong bốn bit được kiểm tra bị lỗi.

Sử dụng mã Hamming để Sửa lỗi 1 bit



Chẵn lẻ Chẵn: LỖI trong bit 4, 5, 6 hoặc 7!

Tiếp tục với vị trí 2^2 :

Kiểm tra tất cả các vị trí có giá trị 1

Đếm số số 1 trong tất cả các bit tương ứng của thông điệp và vị trí 2^2 và Giá trị chẵn lẻ của số đếm này.

Nếu số đếm này là một số chẵn, chắc chắn có một trong bốn bit được kiểm tra bị lỗi.

Xác định Vị trí Lỗi

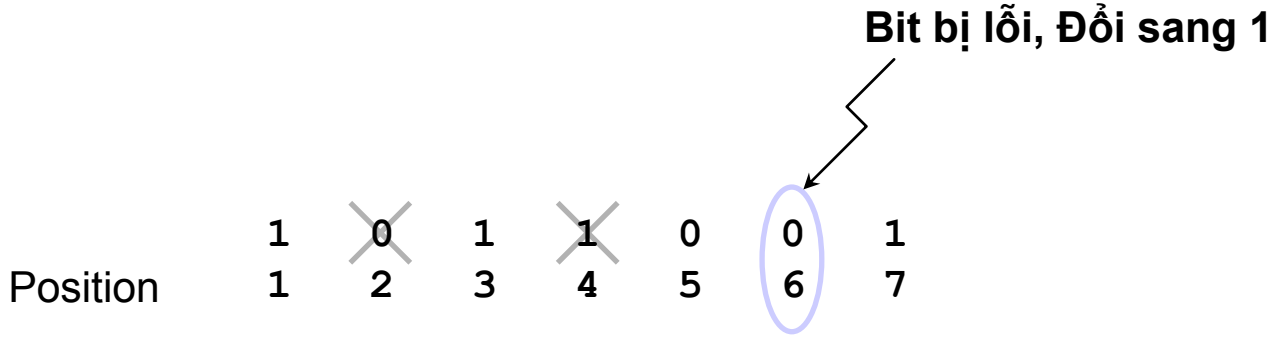
	1	0	1	1	0	0	1
Vị trí	1	2	3	4	5	6	7

Xác định Vị trí Lỗi

	1	0	1	1	0	0	1
Position	1	2	3	4	5	6	7

Không có lỗi tại các vị trí 1, 3, 5, 7

Xác định Vị trí Lỗi



LỖI ở các bit 2, 3, 6 hoặc 7

LỖI ở các bit 4, 5, 6 hoặc 7

Lỗi phải ở trong bit 6 vì các bit 3, 5, 7 đều đúng và tất cả các thông tin còn lại đều khẳng định bit 6 bị lỗi

Xác định Vị trí Lỗi

Giải pháp đơn giản hơn so với slide trước

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

E E NE

↓ ↓ ↓

1 1 0 = 6

E = Cột bị lỗi

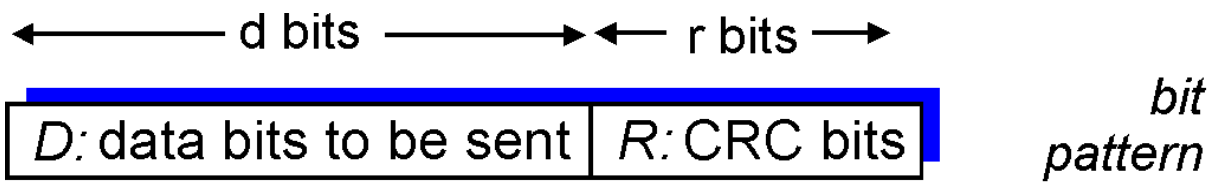
NE = Cột không có lỗi

Mã Hamming – Kết thúc

- Mã Hamming có thể Phát hiện và Sửa được Lỗi một bit
- Nếu có nhiều hơn một bit bị lỗi, mã Hamming không thể sửa được
- Giống bit chẵn lẻ, mã Hamming chỉ có hiệu quả khi thông điệp ngắn

Tổng Kiểm tra: Cyclic Redundancy Check

- Xem các bit dữ liệu (**D**), như một số nhị phân
- Thống nhất một nhóm r+1 bit mẫu **G** (bộ sinh)
- Mục tiêu: Chọn r bit CRC là **R**, sao cho
  - <D,R> chia hết cho G (modulo 2)
  - Phía Nhận cũng xác định được G, chia <D,R> cho G. Nếu dư khác 0 : Phát hiện được Lỗi!
  - Phát hiện được các cụm Lỗi lớn hơn (r+1) bit
- Được sử dụng nhiều trong (ATM, HDCL)



$D * 2^r$ XOR R

mathematical formula

Ví dụ về CRC

Muốn:

$D \cdot 2^r \text{ XOR } R = nG$

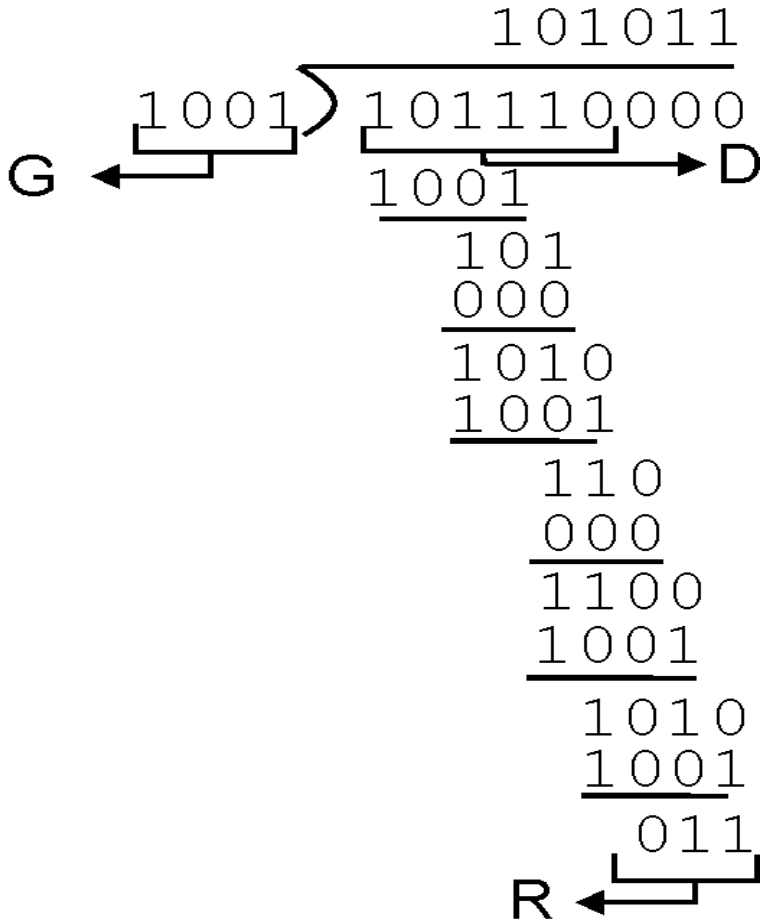
Tương đương với:

$D \cdot 2^r = nG \text{ XOR } R$

Tương đương với:

nếu chia $D \cdot 2^r$ cho G, muốn dư là R

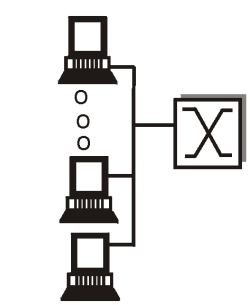
$$R = \text{Dư } \left[\frac{D \cdot 2^r}{G}\right]$$



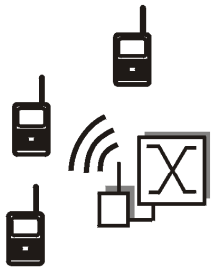
Đa truy cập đường truyền và các Giao thức

Ba kiểu "đường truyền":

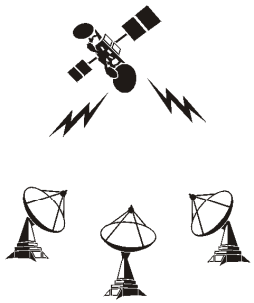
- Điểm nối Điểm (trên một dây truyền, ví dụ PPP, SLIP)
- Quảng bá (dây hay môi trường dùng chung; ví dụ Ethernet, Wavelan,...)



shared wire
(e.g. Ethernet)



shared wireless
(e.g. Wavelan)



satellite

cocktail party

- Chuyển (ví dụ switched Ethernet, ATM)

Giao thức Đa truy cập

- Chia sẻ kênh truyền duy nhất dùng chung
- Nếu có hai cuộc truyền diễn ra đồng thời: xung đột
  - Tại một thời điểm chỉ có một nút có thể truyền Thành công
- *Giao thức Đa truy cập:*
  - Thuật toán phân tán xác định cách thức các trạm chia sẻ kênh truyền, tức là được truyền khi nào.
  - Thông tin về Kênh truyền được lấy từ chính Kênh truyền !
  - Đối với giao thức Đa truy cập:
    - Đồng bộ hay Không đồng bộ
    - Thông tin cần thiết về các trạm khác
    - Khả năng (ví dụ mức độ có lỗi của kênh truyền)
    - Hiệu quả sử dụng

Các Giao thức Đa truy cập

- Chú ý: Loài người thường xuyên sử dụng Giao thức Đa truy cập
- Các bạn có thể đưa ra các giao thức Đa truy cập trong xã hội loài người
  - Giao thức 1:
  - Giao thức 2:
  - Giao thức 3:
  - Giao thức 4:

Đa truy cập : Phân loại

Chia ra ba nhóm chính :

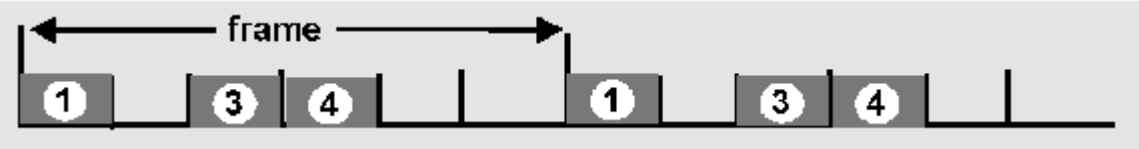
- Phân chia Kênh truyền
  - Chia kênh truyền thành các "phần" nhỏ hơn (theo thời gian, tần số)
  - Mỗi phần được cấp phát riêng cho một nút
- Truy cập Ngẫu nhiên
  - Cho phép xung đột
  - "khắc phục" từ xung đột
- **"Lấy lượt"**
  - Phối hợp chặt chẽ với nhau để đảm bảo không có xung đột

Mục tiêu: Hiệu quả, Công bằng, Đơn giản, Phân tán

Giao thức phân chia kênh truyền: TDMA

TDMA: Time Division Multiple Access

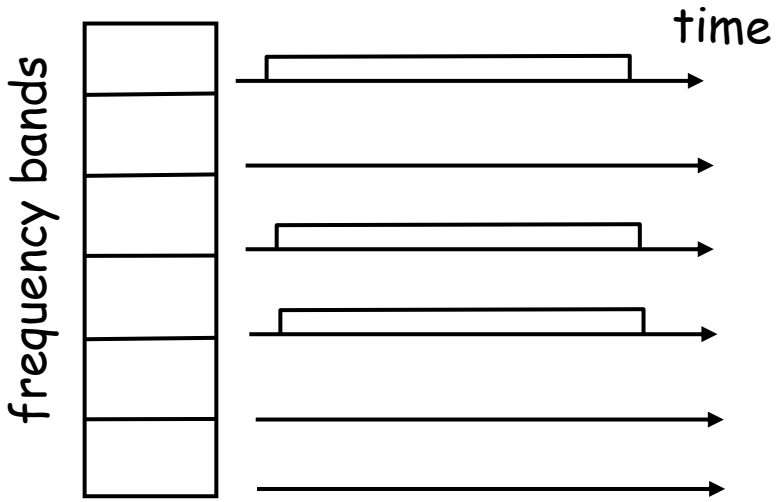
- Truy cập kênh truyền theo từng “vòng"
- Trong mỗi vòng, mỗi trạm sẽ được cấp phát một slot có kích thước cố định (đủ để truyền đi một gói tin)
- Các slot rỗi không được sử dụng
- Ví dụ: 6 trạm LAN là 1,3,4 có dữ liệu, slots 2,5,6 rỗi



Giao thức phân chia kênh truyền: FDMA

FDMA: Frequency Division Multiple Access

- Phổ của kênh truyền được chia thành các dải tần số
- Mỗi trạm được cấp phát một dải băng tần cố định
- Các băng tần rỗi không được sử dụng
- Ví dụ: 6 trạm LAN là 1,3,4 có dữ liệu, slots 2,5,6 rỗi

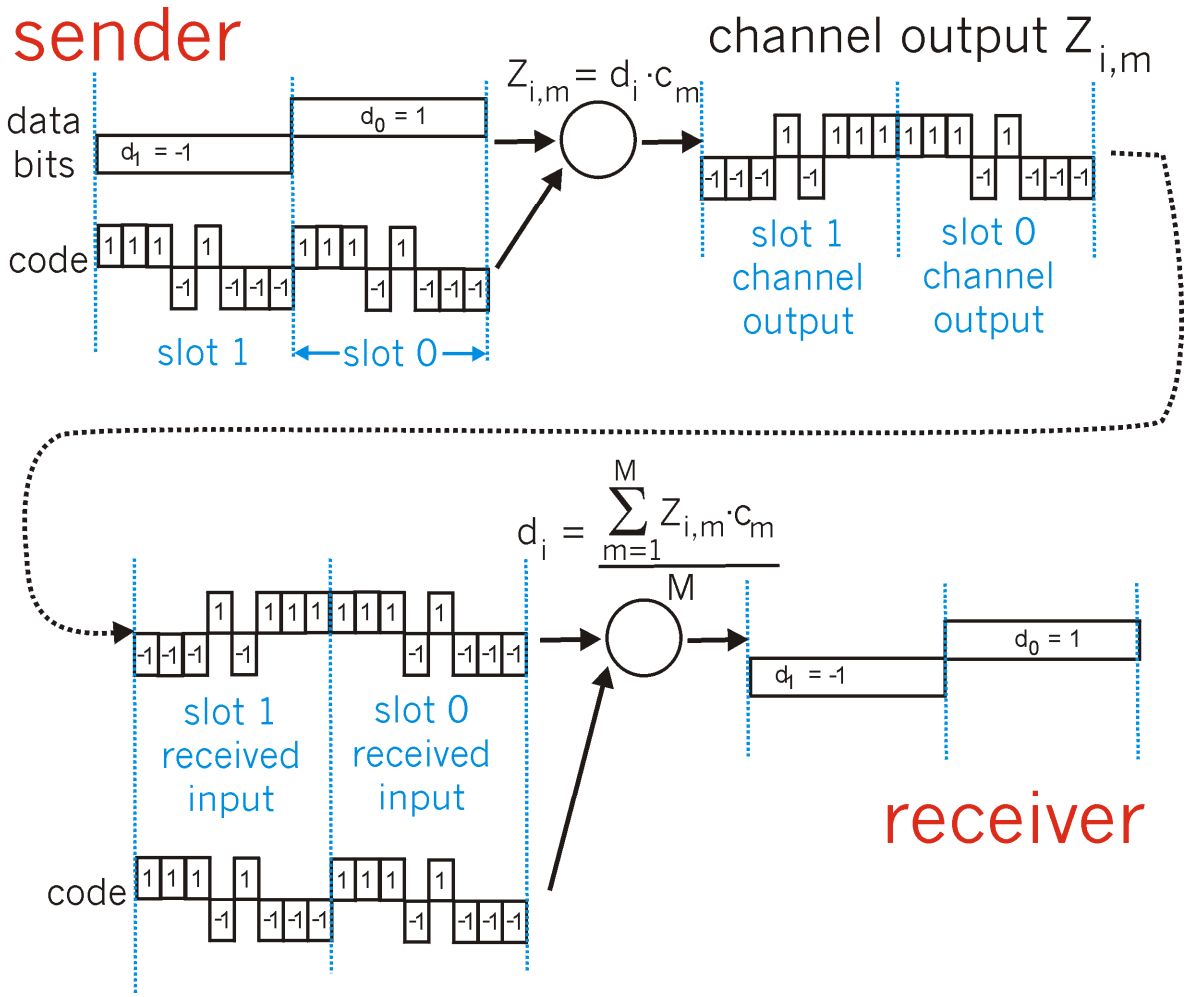


Giao thức phân chia kênh truyền: CDMA

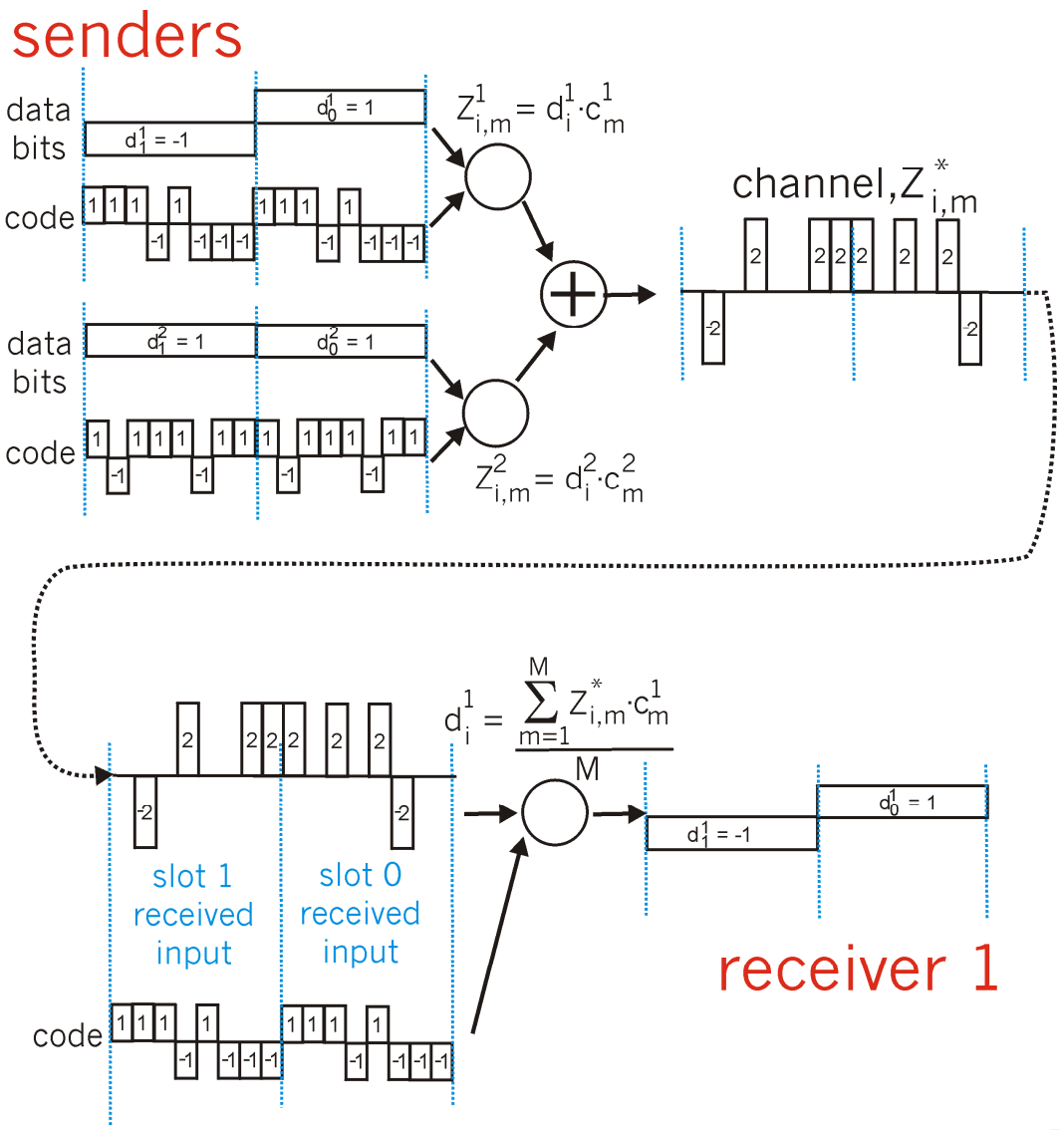
CDMA (Code Division Multiple Access)

- Một "mã" duy nhất được gán cho mỗi người sử dụng (phân chia theo mã)
- Chủ yếu sử dụng trong kênh truyền Quảng bá không dây (cellular, satellite,etc)
- Tất cả người sử dụng dùng chung một tần số, nhưng sử dụng các "mã" riêng để mã hóa dữ liệu
- *Tín hiệu được mã hóa* = (Dữ liệu gốc) X (mã)
- *Giải mã:* Tích của Dữ liệu gốc với Mã
- Cho phép nhiều người dùng có thể truyền đồng thời

CDMA : Mã hóa và Giải mã



CDMA: Hai phía Gửi xen kẽ nhau

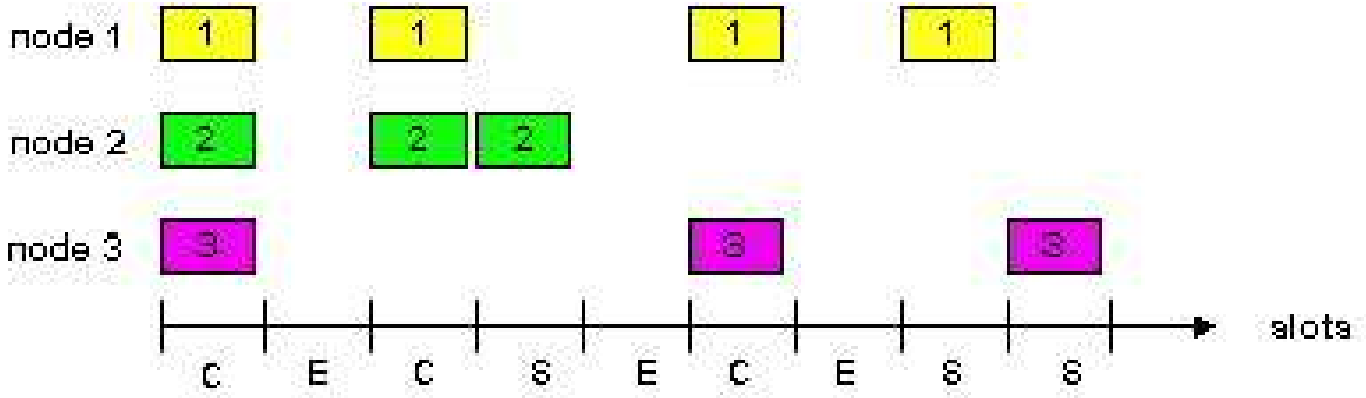


Giao thức Truy cập Ngẫu nhiên

- Khi nút có dữ liệu để truyền đi
  - Truyền với tốc độ tối đa R.
  - Không có sự phối hợp trước với các nút
- Nhiều hơn hai nút cần truyền -> "xung đột",
- Giao thức đa truy cập ngẫu nhiên cần phải:
  - Làm thế nào để xác định có xung đột
  - Khắc phục xung đột như thế nào (ví dụ truyền lại sau một khoảng thời gian)
- Ví dụ các Giao thức Đa truy cập ngẫu nhiên:
  - ALOHA chia khe
  - ALOHA
  - CSMA và CSMA/CD

ALOHA chia khe

- Thời gian được chia thành các slot có kích thước bằng nhau (thời gian đủ để truyền đi một gói tin)
- Nếu có dữ liệu cần gửi: Nút truyền tại đầu slot
- Nếu xung đột: truyền lại gói tin trong slot sau với xác suất truyền là p.



Thành công (S), Xung đột (C), Rỗi (E)

Hiệu suất của ALOHA chia khe

Q: Tỷ lệ truyền thành công của slot là bao nhiêu?

A: Giả sử có N trạm có Dữ liệu cần truyền

- Mỗi trạm truyền tại đầu slot với xác suất *p*
- Xác suất truyền thành công *S* được xác định:

Một nút truyền thành công: $S = p\,(1-p)^{(N-1)}$

Bởi N nút

S = *Xác suất (chỉ một nút truyền)*

$= N\,p\,(1-p)^{(N-1)}$

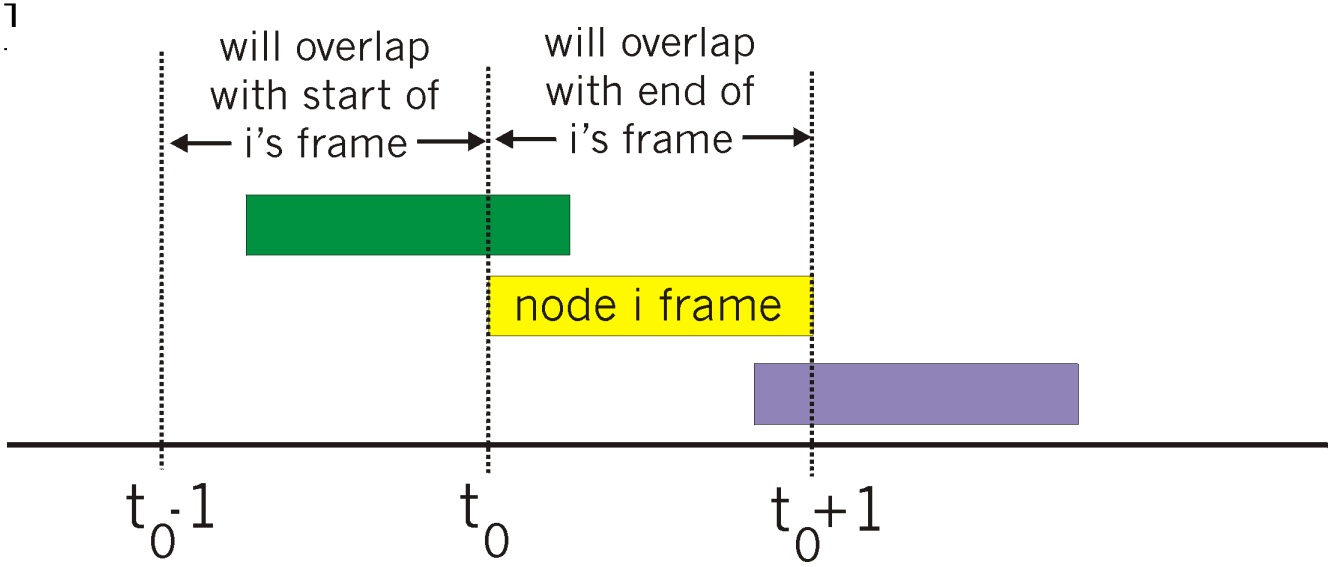
…

$= 1/e = .37$ *khi N tiến ra vô cùng*

Tốt nhất: **Kênh chỉ được sử dụng hữu ích trong 37% Thời gian**

ALOHA thuần túy (Không chia khe)

- Đơn giản hơn, Không cần đồng bộ
- Cũng phải truyền lại gói tin:
  - Gửi không cần phải tại đầu slot
- Xác suất xung đột tăng:
  - Gói tin gửi lúc t0 xung đột với gói tin gửi trong khoảng [t0-1, t0+1]



ALOHA thuần túy (tiếp)

P(một nút truyền thành công) = P(nút truyền) .

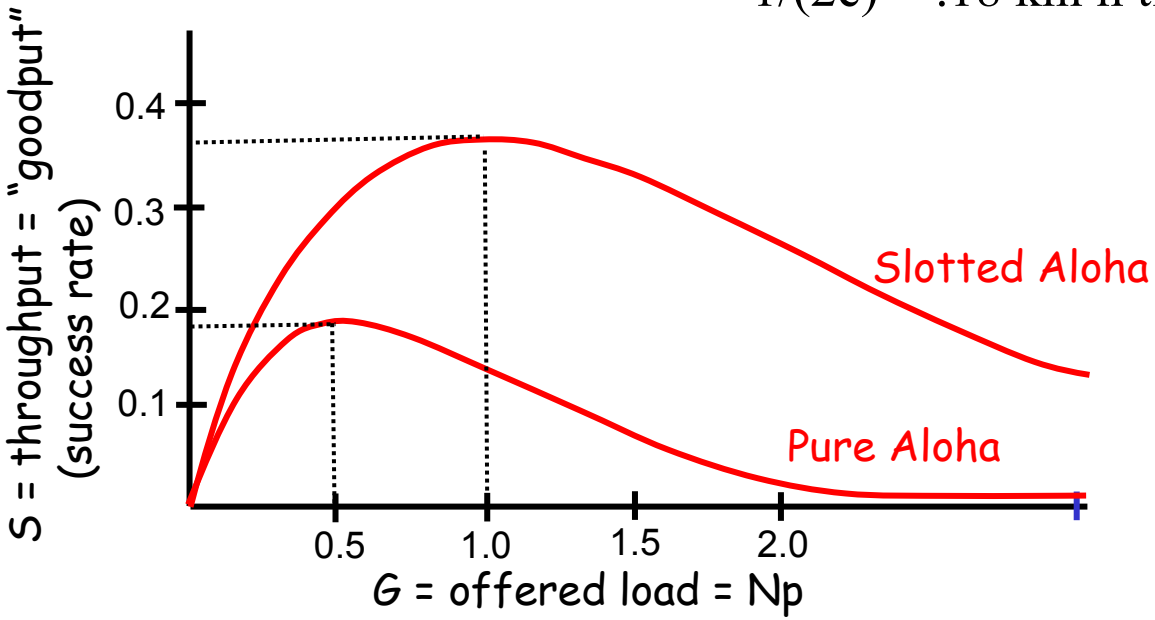
P(không có nút nào truyền trong [p0-1,p0] .

P(không có nút nào truyền trong [p0-1,p0]

= p · (1-p) · (1-p)

P(Bất kỳ nút nào trong N nút truyền thành công) = N p . (1-p) . (1-p)

= 1/(2e) = .18 khi n tiến ra vô cùng



Giao thức hạn chế đáng kể thông lượng Kênh truyền!

CSMA: Carrier Sense Multiple Access

CSMA: Nghe trước khi nói:

- Nếu kênh truyền rỗi : truyền toàn bộ gói tin
- Nếu kênh truyền bận : trì hoãn việc truyền
  - CSMA Kiên trì: thử lại ngay lập tức việc lắng nghe kênh truyền với xác suất p (có thể diễn ra ngay lập tức)
  - CSMA không kiên trì: Thử lại sau một khoảng thời gian ngẫu nhiên
- Xã hội loài người : Không "nhảy vào miệng người khác" !

CSMA : Xung đột

Vẫn có thể có Xung đột :

Độ trễ lan toả nghĩa là hai nút có thể không nghe được cuộc truyền của bên kia

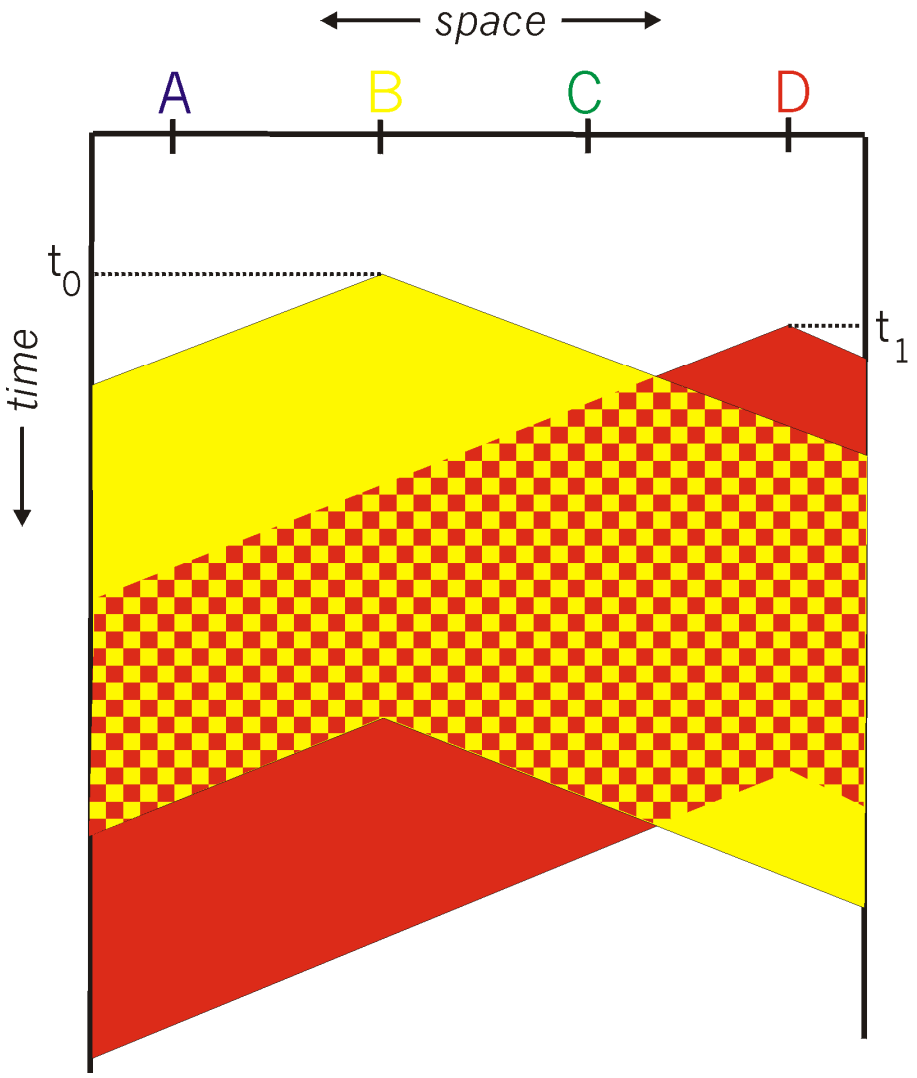
Xung đột:

Thời gian truyền gói tin bị lãng phí

Chú ý:

Khoảng cách cũng như Vận tốc lan tỏa ảnh hưởng lớn đến xác suất xảy ra xung đột.

Cần tính Khoảng cách giữa các nút



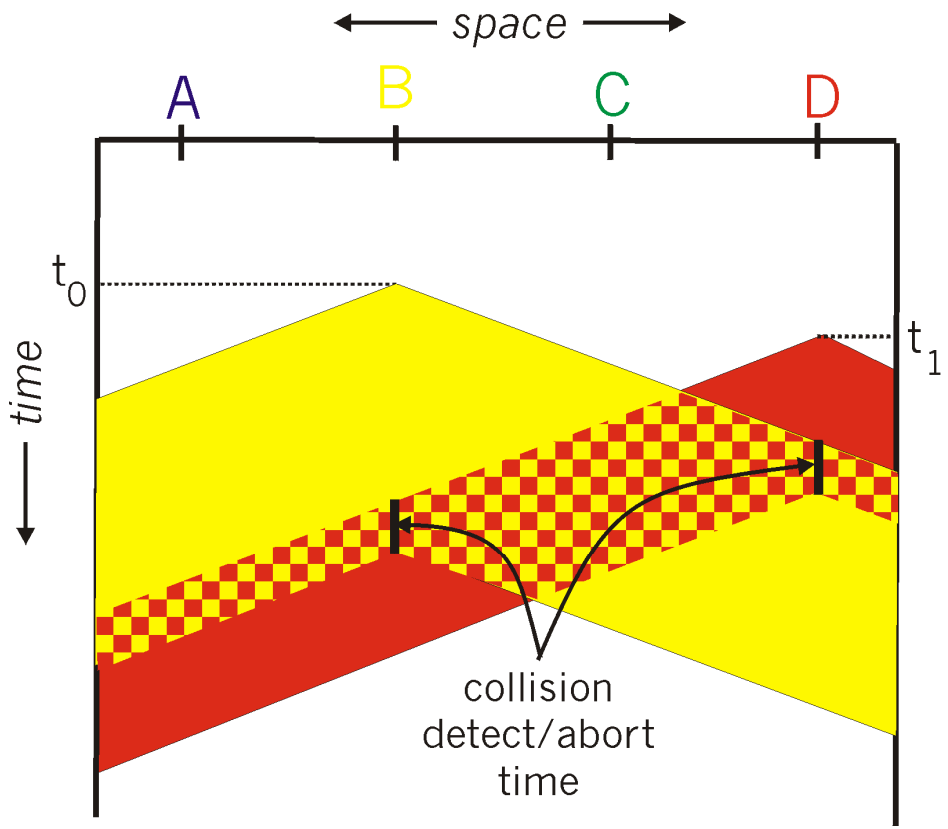
CSMA/CD (Collision Detection)

CSMA/CD: cảm nhận sóng mang, giống như trong CSMA

- Xung đột *bị phát hiện* trong thời gian ngắn
- Loại bỏ các cuộc truyền xung đột => Không lãng phí
- Truyền lại : Kiên trì hoặc Không kiên trì

- Phát hiện Xung đột:
  - Dễ trong môi trường Hữu tuyến: đo bước sóng, so sánh tín hiệu truyền đi và tín hiệu nhận được
  - Khó trong môi trường Vô tuyến: Phía nhận tự dưng bị treo
- Con người: Hội thoại một cách Lịch sự

CSMA/CD collision detection



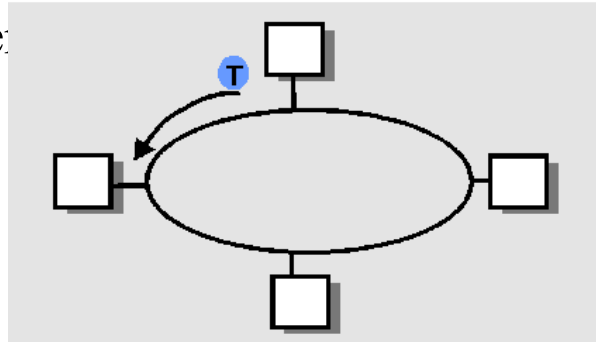
Các Giao thức "Lần lượt"

Hỏi vòng:

- Nút master "mời" các nút slave truyền theo lượt
- Các thông điệp Request to Send, Clear to Send
- Vấn đề:
  - Chi phí phụ trội
  - Độ trễ
  - Sụp đổ khi nút master lỗi

Chuyển Thẻ bài :

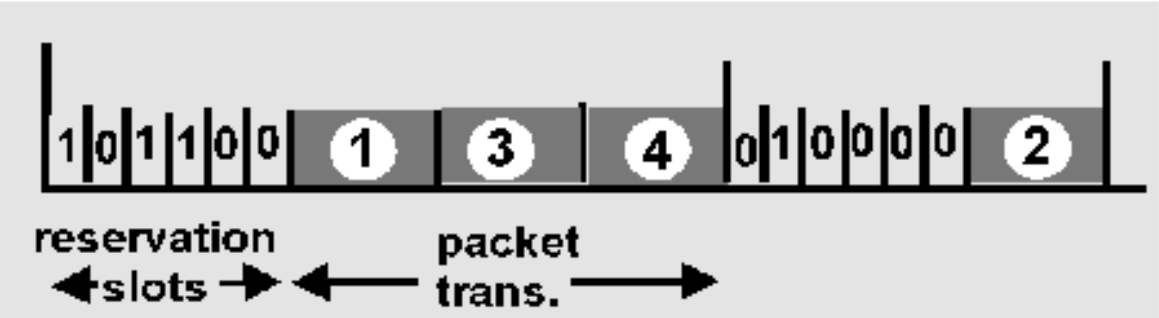
- **Thẻ bài** (thông điệp điều khiển) chuyển từ nút này sang nút kế tiếp.
- Thông điệp Thẻ bài
- Vấn đề:
  - Chi phí phụ trội
  - Độ trễ
  - Sụp đổ khi mất thẻ bài



Các Giao thức dựa trên việc đặt chỗ trước

Thăm dò Phân tán:

- Thời gian được chia thành các slot
- Bắt đầu bằng N slot đặt chỗ
  - Thời gian mỗi slot đặt chỗ bằng độ trễ lan tỏa
  - Trạm cần truyền dữ liệu : đặt chỗ
  - Tất cả các trạm đều quan sát được slot đặt chỗ
- Sau các slot đặt chỗ, các nút truyền theo thứ tự



Giao thức Đếm nhị phân

- Trong thời gian có tranh chấp,
  - Mỗi máy tính gửi Quảng bá địa chỉ cả mình, lần lượt từng bit một, bắt đầu từ bit cao nhất
  - Các bit được truyền đồng thời sẽ được OR với nhau

- Quy tắc Trọng tài tranh chấp:
  - Nếu máy tính gửi đi bit 0 nhưng kết quả phép toán OR là 1 thì sẽ tự động dừng lại và không tiếp tục gửi các bit địa chỉ nữa
  - Nút nào có thể truyền đi tất cả các bit địa chỉ sẽ được quyền truy cập kênh truyền

Giao thức Đếm nhị phân

Địa chỉ Host	Lần lượt từng bit			
	0	1	2	3
0 0 1 0	0	-	-	-
0 1 0 1	0	-	-	-
1 0 0 1	1	0	0	-
1 0 1 0	1	0	1	0

Giao thức Đếm nhị phân : Tính công bằng

- Trạm có địa chỉ lớn nhất luôn "chiến thắng".
- Ưu điểm : Khi muốn cài đặt dựa trên Độ ưu tiên
- Nhược điểm : Khi muốn phân chia kênh truyền công bằng
- Giải pháp:
  - Thay đổi địa chỉ của nút sau mỗi lần truyền thành công
  - Các nút có địa chỉ nhỏ hơn A được cộng thêm 1
  - A nhận địa chỉ 0

Các Giao thức "Lần lượt"

Phân chia Kênh truyền:

- Chỉ Hiệu quả cao khi tải lớn
- Không hiệu quả khi tải thấp: có độ trễ, băng thông chỉ là 1/N kể cả khi chỉ có một nút truyền!

Truy cập Ngẫu nhiên

- Hiệu quả khi tải thấp: Nút duy nhất có thể tận dụng toàn bộ kênh truyền
- Tải cao: Quá nhiều Xung đột

Truy cập Lần lượt

Có vẻ tốt hơn cả hai!