

# **Tầng Liên Kết Dữ Liệu (Data Link Layer)**

Trình bày: TS Ngô Bá Hùng

# Mục đích

---

- Chương này nhằm giới thiệu những nội dung cơ bản sau:
  - Các chức năng cơ bản mà tầng liên kết dữ liệu đảm trách
  - Vai trò của khung trong vấn đề xử lý lỗi đường truyền và các phương pháp xác định khung
  - Giới thiệu các phương pháp phát hiện lỗi như Phương pháp kiểm tra chẵn lẻ, Phương pháp kiểm tra theo chiều dọc và Phương pháp kiểm tra phần dư tuần hoàn.
  - Giới thiệu các giao thức điều khiển lỗi cho phép theo dõi tình trạng lỗi của dữ liệu gửi đi
  - Giới thiệu các giao thức xử lý lỗi chỉ ra các cách giải quyết trường hợp dữ liệu truyền đi bị lỗi.

# Yêu cầu

---

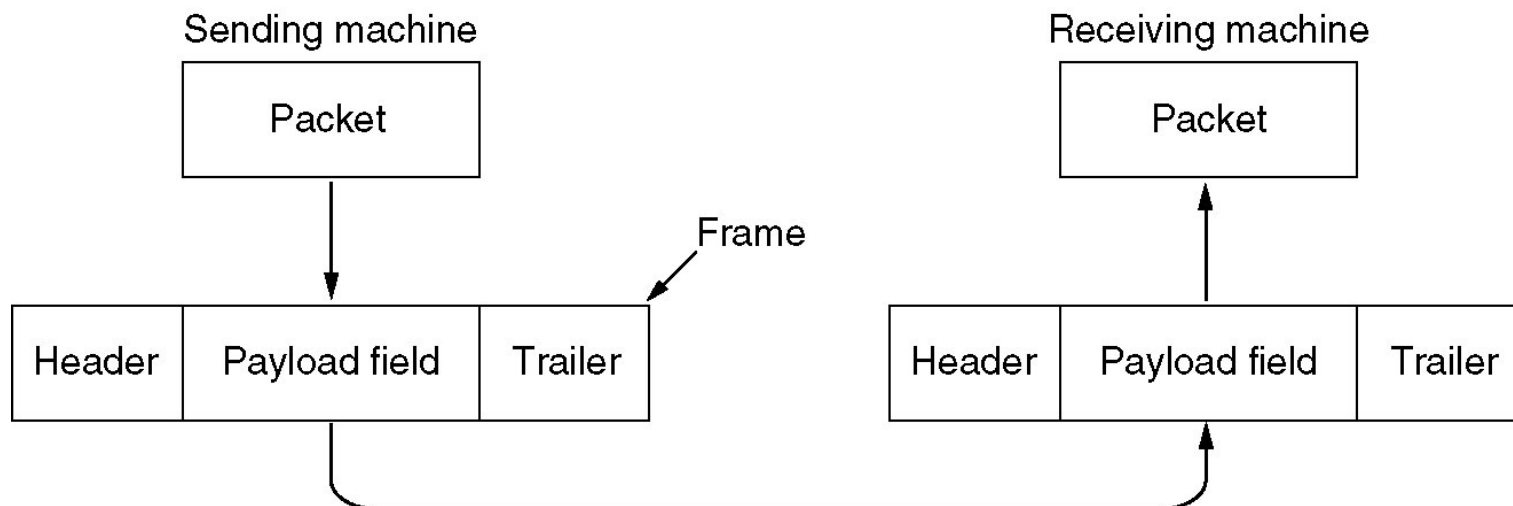
- Sau khi học xong chương này, người học phải có được những khả năng sau:
  - Biện luận được vai trò của tầng liên kết dữ liệu trong vấn đề xử lý lỗi dữ liệu truyền nhận
  - Trình bày được các phương pháp định khung đếm ký tự, phương pháp sử dụng byte là cờ và phương pháp sử dụng cờ đặc biệt
  - Phân biệt được sự khác nhau giữa các chức năng phát hiện lỗi, điều khiển lỗi và xử lý lỗi của tầng hai.
  - Cài đặt được cơ chế phát hiện lỗi theo các phương pháp kiểm tra chẵn lẻ, Phương pháp kiểm tra theo chiều dọc và Phương pháp kiểm tra phần dư tuần hoàn
  - Cài đặt được các giao thức điều khiển lỗi Dừng và chờ, giao thức cửa sổ trượt
  - Cài đặt được giao thức xử lý lỗi Go-Back-N và giao thức Selective Repeat
  - Trình bày được ý tưởng cơ bản của giao thức HDLC

## Chức năng của tầng liên kết dữ liệu

---

- Cung cấp một giao diện được định nghĩa chuẩn cho các dịch vụ cung cấp cho tầng mạng.
- Xử lý lỗi đường truyền.
- Điều khiển luồng dữ liệu nhờ đó bên truyền nhanh không làm tràn dữ liệu bên nhận chậm

# Chức năng của tầng liên kết dữ liệu



- Cung cấp các dịch vụ cho tầng mạng
- Truyền tải dữ liệu nhận được từ tầng mạng trên máy gửi đến tầng mạng trên máy nhận

# Chức năng của tầng liên kết dữ liệu

## Các dịch vụ cơ bản

---

- Dịch vụ không nối kết không báo nhận (unacknowledged connectionless service), thường được sử dụng trong mạng LAN.
- Dịch vụ không nối kết có báo nhận (acknowledged connectionless service), thường dùng cho mạng không dây.
- Dịch vụ nối kết định hướng có báo nhận (acknowledged connection-oriented service), thường dùng trong mạng WANs.

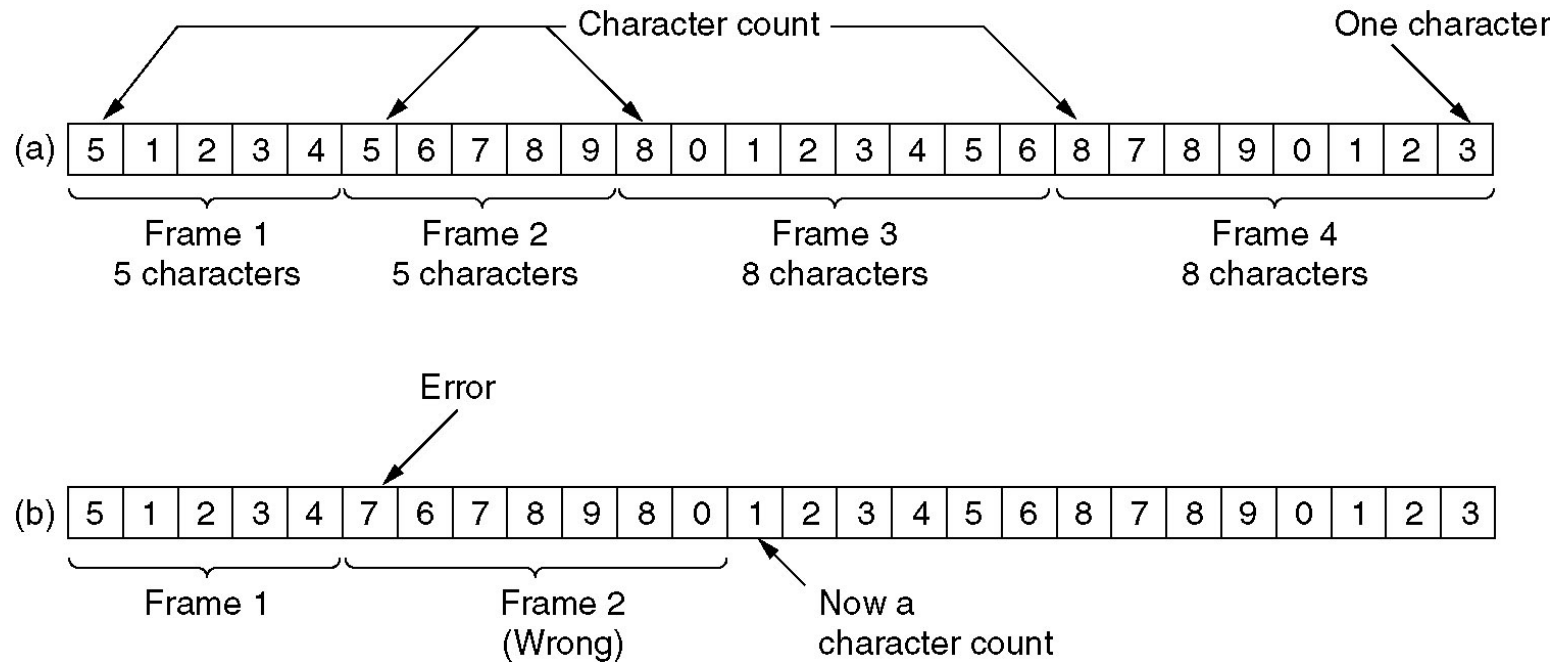
# Chức năng của tầng liên kết dữ liệu

## Định khung

---

- Qui định khuôn dạng của khung được sử dụng ở tầng Liên kết dữ liệu
- 3 phương pháp định khung phổ biến:
  - Đếm ký tự (Character count)
  - Sử dụng các bytes làm cờ hiệu và các bytes đệm (Flag byte with byte stuffing)
  - Sử dụng cờ bắt đầu và kết thúc khung cùng với các bit đệm (Starting and ending flags with bit stuffing)

# Phương pháp đếm ký tự (Character Count)

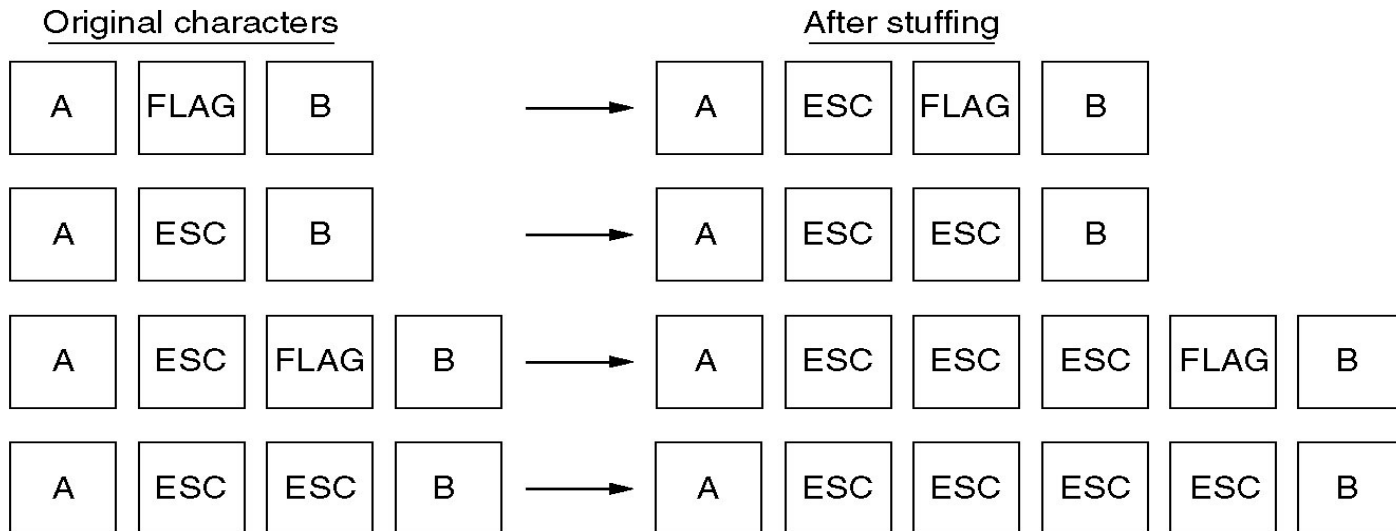




# Phương pháp sử dụng byte làm cờ và các byte độn (Flag byte with byte stuffing)



(a)



(b)

(a) Khung được đánh dấu bởi cờ hiệu,

(b) Dữ liệu có chứa cờ hiệu và byte ESC.

## Phương pháp sử dụng cờ bắt đầu và kết thúc khung cùng với các bit đệm (Starting and ending flags with bit stuffing)

- Sử dụng mẫu bit đặc biệt, 01111110, để làm cờ đánh dấu điểm bắt đầu và kết thúc khung

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(a) Dữ liệu gốc,

(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 0 1 0 0 1 0

(b) Dữ liệu chuyển lên đường truyền,

Stuffed bits

(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(c) Dữ liệu nhận sau khi loại bỏ các bit đệm.

# Chức năng của tầng liên kết dữ liệu

## Điều khiển lỗi (Error Control)

---

- Cách nào để đảm bảo rằng toàn bộ các khung đã được phân phát đến tầng mạng và được phân phát theo đúng trình tự chúng đã được gửi ?
  - Người nhận báo về tình trạng nhận khung:
    - Sử dụng Khung báo nhận (acknowledgement)
  - Tránh chờ vĩnh viễn:
    - Sử dụng bộ đếm thời gian (timer) + time-out
  - Trùng lặp gói tin nhận:
    - Gán số thứ tự cho khung

# Chức năng của tầng liên kết dữ liệu

## Điều khiển luồng (Flow Control)

---

- Giải quyết sự khác biệt về tốc độ truyền / nhận dữ liệu của bên truyền và bên nhận
- Hai tiếp cận:
  - Tiếp cận điều khiển luồng dựa trên phản hồi (feedback based flow control): Người nhận gửi thông tin về cho người gửi cho phép người gửi gửi thêm dữ liệu, cũng như báo với người gửi những gì mà người nhận đang làm.
  - Tiếp cận điều khiển luồng dựa trên tần số (rate based flow control): Trong giao thức truyền tin cài sẵn cơ chế giới hạn tần suất mà người gửi có thể truyền tin

# Vấn đề xử lý lỗi

# Vấn đề xử lý lỗi

---

- Bộ mã phát hiện lỗi là gì ?
- Những bộ mã phát hiện lỗi
  - Kiểm tra chẵn lẻ (Parity checks)
  - Kiểm tra thêm theo chiều dọc (Longitudinal redundancy check)
  - Kiểm tra phần dư tuần hoàn (Cyclic redundancy check)

# Lỗi trên đường truyền

---

- Bit 1 thành bit 0 và ngược lại

- Tỷ lệ lỗi

$\forall \tau = \text{Số bit bị lỗi} / \text{Tổng số bit được truyền}$

$\forall \tau : 10^{-5} \text{ đến } 10^{-8}$

- 88% : sai lệch một bit
- 10% : sai lệch 2 bit kề nhau

# Bộ mã phát hiện lỗi

- Bên cạnh các thông tin hữu ích cần truyền đi, ta thêm vào các thông tin điều khiển. Bên nhận thực hiện việc giải mã các thông tin điều khiển này để phân tích xem thông tin nhận được là chính xác hay có lỗi





# Bộ mã phát hiện lỗi

---

- Bộ mã sửa lỗi (Error-correcting codes):
  - Cho phép bên nhận có thể tính toán và suy ra được các thông tin bị lỗi (sửa dữ liệu bị lỗi)
- Bộ mã phát hiện lỗi (Error-detecting codes):
  - cho phép bên nhận phát hiện ra dữ liệu có lỗi hay không
  - Nếu có lỗi bên nhận sẽ yêu cầu bên gửi gửi lại thông tin
- Các hệ thống mạng ngày nay có xu hướng chọn bộ mã phát hiện lỗi.

# Phương pháp Kiểm tra chẵn lẻ (Parity Check)

---

- xxxxxxx: chuỗi bits dữ liệu cần truyền
- Thêm vào 1 bit chẵn-lẻ p
- Chuỗi bit truyền là: xxxxxxxp
- p được tính để đảm bảo:
  - Phương pháp kiểm tra chẵn: xxxxxxxp có một số chẵn các bit 1
  - Phương pháp kiểm tra lẻ: xxxxxxxp có một số lẻ các bit 1
- Bên nhận nhận được chuỗi xxxxxxxp:
  - Phương pháp kiểm tra chẵn:
    - Nếu có 1 số chẵn các bit 1: Dữ liệu xxxxxxx không có lỗi
    - Ngược lại là có lỗi
  - Phương pháp kiểm tra lẻ:
    - Nếu có 1 số lẻ các bit 1: Dữ liệu xxxxxxx không có lỗi
    - Ngược lại là có lỗi

# Phương pháp Kiểm tra chẵn lẻ (Parity Check)

---

- Ví dụ: Cần truyền ký tự  $G = 1110001$
- Sử dụng phương pháp kiểm tra chẵn:
  - $p=0$
  - Chuỗi truyền đi là:  $1110001\underline{0}$
- Bên nhận nhận được chuỗi:
  - $11100010$ : 4 bit 1  $\Rightarrow$  không có lỗi
  - $11\underline{0}00010$ : 3 bit 1  $\Rightarrow$  dữ liệu có lỗi
  - $11\underline{0}00\underline{1}10$ : 4 bit 1  $\Rightarrow$  không có lỗi ???

# Kiểm tra thêm theo chiều dọc (Longitudinal Redundancy Check or Checksum)

---

		Parity bits
		↓
	1 0 1 1 0 1 1	<u>1</u>
	1 1 0 1 0 1 1	<u>1</u>
Data bits →	0 0 1 1 1 0 1	<u>0</u>
	1 1 1 1 0 0 0	<u>0</u>
	1 0 0 0 1 0 1	<u>1</u>
	0 1 0 1 1 1 1	<u>1</u>
LRC bits →	<u>0</u> <u>1</u> <u>1</u> <u>1</u> <u>1</u> <u>1</u> <u>1</u>	<u>0</u>

# Kiểm tra phần dư tuần hoàn (Cyclic Redundancy Check)

---

- Một số phương pháp cài đặt khác nhau như:
  - Modulo 2,
  - Đa thức,
  - Thanh ghi dịch
  - Các cổng Exclusive-or

# Kiểm tra phần dư tuần hoàn

## Modulo 2

---

- Giả sử ta có:
  - M: Thông điệp k bit cần gửi sang bên nhận.
  - F : Chuỗi kiểm tra khung FCS gồm r bit là thông tin điều khiển được gửi theo M để giúp bên nhận có thể phát hiện được lỗi.
  - $T = MF$  là khung (k + r) bit, được hình thành bằng cách nối M và F lại với nhau. T sẽ được truyền sang bên nhận, với  $r < k$
- Với M (k bit) , P (r+1 bit), F (r bit), T (k+r bit), thủ tục tiến hành để xác định checksum F và tạo khung truyền như sau:
  - Nối r bit 0 vào cuối M, hay thực hiện phép nhân M với  $2^r$
  - Dùng phép chia modulo 2 chia chuỗi bit  $M \cdot 2^r$  cho P.
  - Phần dư của phép chia sẽ được cộng với  $M \cdot 2^r$  tạo thành khung T truyền đi.
  - Trong đó P được chọn dài hơn F một bit, và cả hai bit cao nhất và thấp nhất phải là 1
- Bên nhận thực hiện phép chia T cho P:
  - Chia hết: T không có lỗi, Dữ liệu M từ T – k bits trọng số cao
  - Chia không hết: T có lỗi

# Kiểm tra phần dư tuần hoàn

## Modulo 2

---

- Giả sử ta có:
  - $M = 1010001101$  (10 bit)
  - $P = 110101$  (6 bit)
  - FCS cần phải tính toán (5 bit)
- Lần lượt thực hiện các bước sau:
  - Tính  $M * 2^5 = 101000110100000$ .
  - Thực hiện phép chia modulo  $M * 2^5$  cho  $P$  ta được phần dư  $F = 01110$
  - Tạo khung gửi đi là  $T = M * 2^r + F = 101000110101110$

$$\begin{array}{r}
 \text{(P) } 1\ 1\ 0\ 1\ 0\ 1 \mid \begin{array}{r}
 1\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 1\ 0 \\
 \hline
 1\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 0 \\
 \hline
 1\ 1\ 0\ 1\ 0\ 1 \\
 \hline
 1\ 1\ 1\ 0\ 1\ 1 \\
 \hline
 1\ 1\ 0\ 1\ 0\ 1 \\
 \hline
 0\ 1\ 1\ 1\ 0\ 1 \\
 0\ 0\ 0\ 0\ 0\ 0 \\
 \hline
 1\ 1\ 1\ 0\ 1\ 0 \\
 1\ 1\ 0\ 1\ 0\ 1 \\
 \hline
 0\ 1\ 1\ 1\ 1\ 1 \\
 0\ 0\ 0\ 0\ 0\ 0 \\
 \hline
 1\ 1\ 1\ 1\ 1\ 0 \\
 1\ 1\ 0\ 1\ 0\ 1 \\
 \hline
 0\ 1\ 0\ 1\ 1\ 0 \\
 0\ 0\ 0\ 0\ 0\ 0 \\
 \hline
 1\ 0\ 1\ 1\ 0\ 0 \\
 1\ 1\ 0\ 1\ 0\ 1 \\
 \hline
 1\ 1\ 0\ 0\ 1\ 0 \\
 1\ 1\ 0\ 1\ 0\ 1 \\
 \hline
 0\ 0\ 1\ 1\ 1\ 0 \\
 0\ 0\ 0\ 0\ 0\ 0 \\
 \hline
 \mathbf{0\ 1\ 1\ 1\ 0 = F}
 \end{array}
 \end{array}
 \begin{array}{l}
 \text{(Q : Kết quả phép chia)} \\
 \text{(M*2^r)}
 \end{array}$$



# Kiểm tra phần dư tuần hoàn

## Phương pháp đa thức

---

- Giả sử ta có  $M=110011$  và  $P = 11001$ , khi đó  $M$  và  $P$  được biểu diễn lại bằng 2 đa thức sau:
  - $M(x) = x^5 + x^4 + x + 1$
  - $P(x) = x^4 + x^3 + 1$
- Quá trình tính CRC được mô tả dưới dạng các biểu thức sau:

$$\frac{X^n M(X)}{P(X)} = Q(X) + \frac{R(X)}{P(X)}$$

$$T(X) = X^n M(X) + R(X)$$

# Kiểm tra phần dư tuần hoàn

## Phương pháp đa thức

---

Các version thường được sử dụng của P là :

$$\text{CRC-12} = X^{12} + X^{11} + X^3 + X^2 + X + 1$$

$$\text{CRC-16} = X^{16} + X^{15} + X^2 + 1$$

$$\text{CRC-CCITT} = X^{16} + X^{12} + X^5 + 1$$

$$\begin{aligned} \text{CRC-32} = & X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} \\ & + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1 \end{aligned}$$

# Kiểm tra phần dư tuần hoàn

## Phương pháp đa thức

---

Ví dụ:

- Cho:  $M=1010001101$ ,  $P=110101$

- Ta có:  $r=5$

$$M(x) = x^9 + x^7 + x^3 + x^2 + 1$$

$$x^5 M(x) = x^{14} + x^{12} + x^8 + x^7 + x^5$$

$$P(x) = x^5 + x^4 + x^2 + 1$$

- Thực hiện phép toán:

$$\frac{x^5 M(x)}{P(x)} = Q(x) + \frac{F(x)}{P(x)}$$

$$\Rightarrow Q(x) = x^9 + x^8 + x^6 + x^4 + x^2 + x^1$$

$$F(x) = x^3 + x^2 + x^1 \Leftrightarrow \mathbf{01110}$$

$\Rightarrow$  Khung cần truyền đi là  $T = 1010001101\mathbf{01110}$

# **VẤN ĐỀ ĐIỀU KHIỂN (Error Control)**

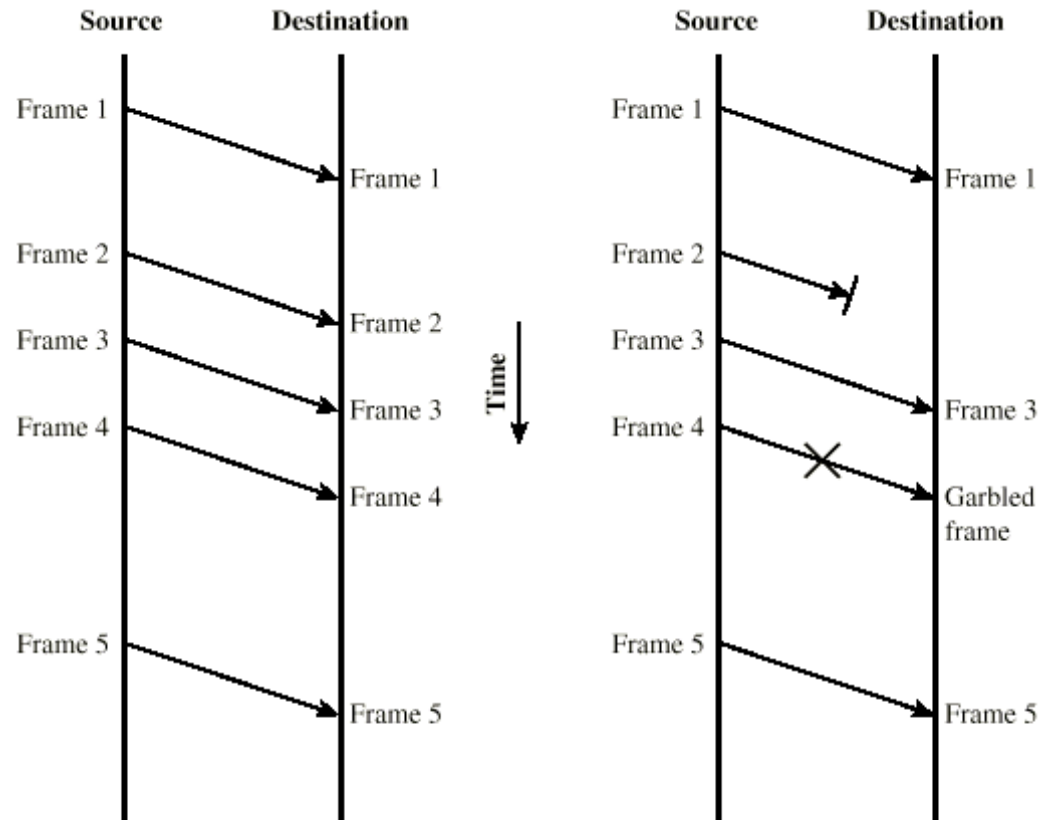
# Chức năng của tầng liên kết dữ liệu

## Điều khiển lỗi (Error Control)

---

- Cách nào để đảm bảo rằng toàn bộ các khung đã được phân phát đến tầng mạng và được phân phát theo đúng trình tự chúng đã được gửi ?
  - Người nhận báo về tình trạng nhận khung:
    - Sử dụng Khung báo nhận (acknowledgement)
  - Tránh chờ vĩnh viễn:
    - Sử dụng bộ đếm thời gian (timer) + time-out
  - Trùng lặp gói tin nhận:
    - Gán số thứ tự cho khung

# Điều khiển lỗi

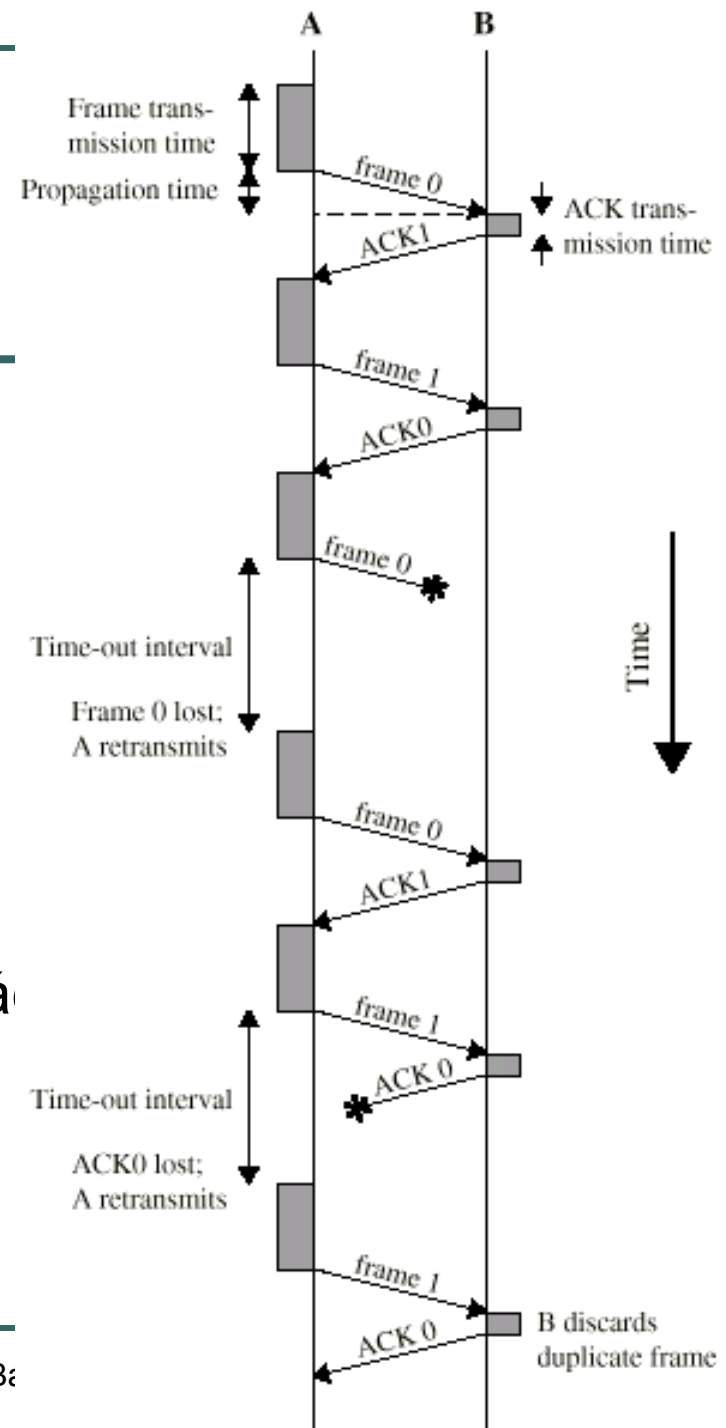


(a) Error-free transmission

(b) Transmission with losses and errors

# Stop and Wait - Diagram

- Người gửi không biết được khung có đến nơi nhận tốt hay không.
  - Giải pháp: Khung báo nhận.
- Các khung báo nhận có thể bị mất.
  - Giải pháp:
    - Timer.
    - Time-out
    - Gửi lại
- Bên nhận không phân biệt được các khung trùng lặp do bên gửi gửi lại.
  - Giải pháp: Mỗi khung sẽ có một số thứ tự



# Vấn đề truyền tải thông tin theo hai chiều (Duplex)

---

- Stop and Wait: truyền đơn công (Simplex)
- Mong muốn việc truyền tải thông tin theo chế độ song công (Duplex) để khai thác tối đa năng lực kênh truyền. Nguyên tắc thực hiện như sau:
  - Vẫn thực hiện việc truyền tải khung,
  - Phân loại khung: DATA, ACK, NACK
  - Sử dụng kỹ thuật **piggyback**.

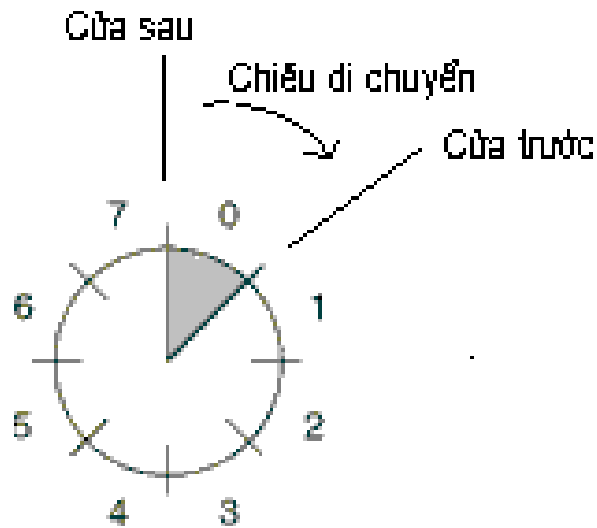


# Giao thức cửa sổ trượt (Sliding windows)

---

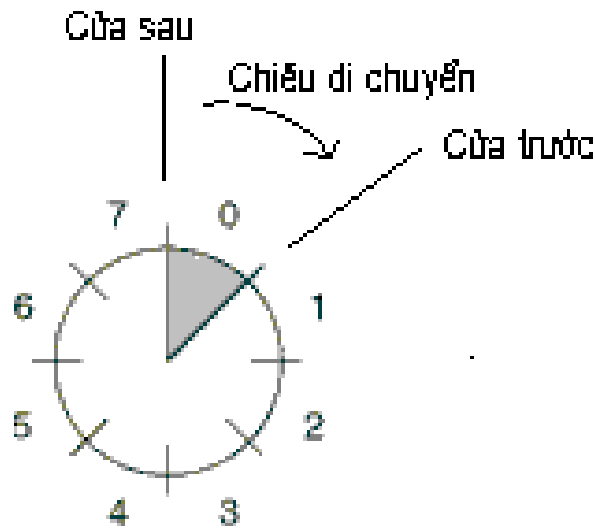
- Thay vì chỉ truyền đi một khung tại một thời điểm (simplex), giao thức cửa sổ trượt cho phép bên gửi có thể gửi đi nhiều khung.
- Cửa sổ gửi (Sending Windows): Bên gửi theo dõi các khung mà nó được phép gửi đi và các khung mà nó đang chờ báo nhận
- Cửa sổ nhận (Receiving Windows): Bên nhận theo dõi các khung mà nó được phép nhận

# Cửa sổ trượt (Sliding Windows)



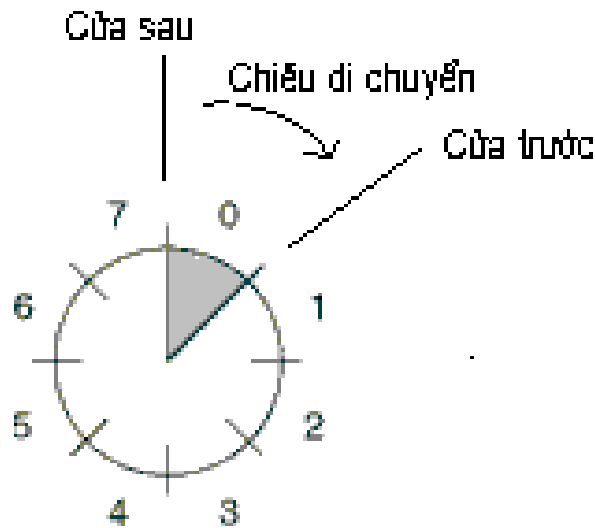
- Cửa sổ gồm có **cửa trước** và **cửa sau** cùng di chuyển theo một chiều.
- Kích thước của cửa sổ là chiều dài của cung giới hạn từ cửa sau đến cửa trước.
- Kích thước của cửa sổ có thể thay đổi:
  - Khi cửa trước di chuyển, cửa sổ được mở rộng ra.
  - Khi cửa sau di chuyển, kích thước của cửa sổ bị thu hẹp lại và nó làm cho cửa sổ thay đổi vị trí, trượt / quay quanh một tâm của vòng

# Cửa sổ trượt (Sliding Windows)



- Kích thước nhỏ nhất của cửa sổ là 0
- Kích thước tối đa của cửa sổ là  $n-1$
- $k$  bit để đánh số thứ tự khung  $[0 - (2^k - 1)] \Rightarrow$  Khi đó cửa sổ trượt sẽ được chia thành  $2^k$  vị trí tương ứng với  $2^k$  khung.
- Đối với cửa sổ gởi, các vị trí nằm trong cửa sổ trượt biểu hiện số thứ tự của các khung mà bên gởi đang chờ bên nhận báo nhận. Phần bên ngoài cửa sổ là các khung có thể gởi tiếp. Tuy nhiên phải đảm bảo rằng, cửa sổ gởi không được vượt quá kích thước tối đa của cửa sổ.

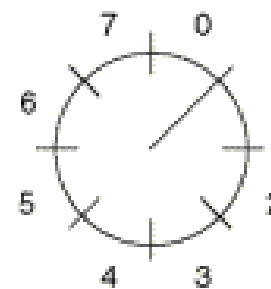
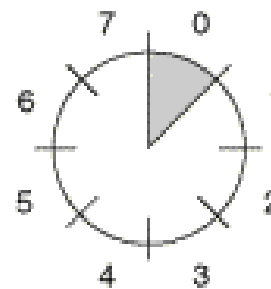
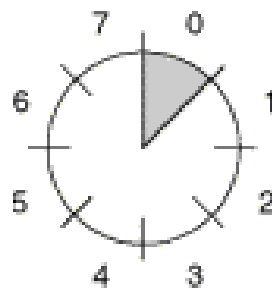
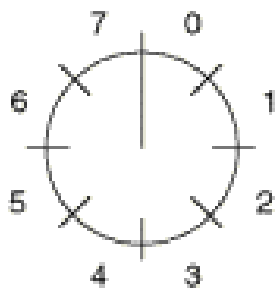
# Cửa sổ trượt (Sliding Windows)



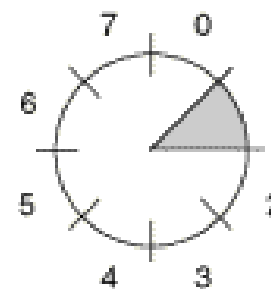
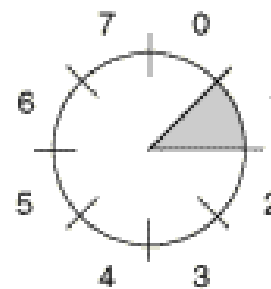
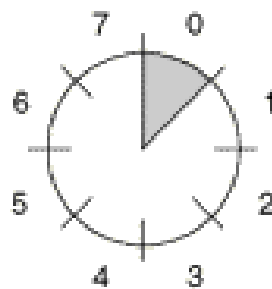
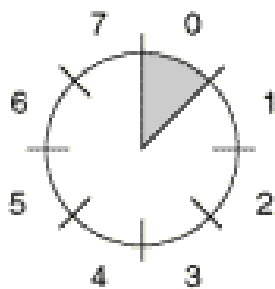
- Đối với bên nhận, các vị trí nằm trong cửa sổ biểu hiện số thứ tự các khung mà nó đang sẵn sàng chờ nhận.
- Kích thước tối đa của cửa sổ biểu thị dung lượng bộ nhớ đệm của bên nhận có thể lưu tạm thời các gói tin nhận được trước khi xử lý chúng.
- Giả sử bên nhận có một vùng bộ nhớ đệm có khả năng lưu trữ 4 khung nhận được. Khi đó, kích thước tối đa của cửa sổ sẽ là 4

# Hoạt động của ổ trượt

Sender



Receiver



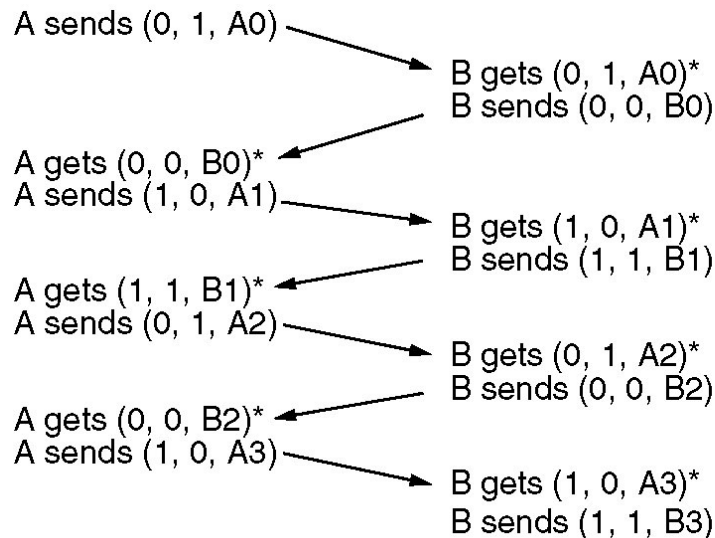
(a)

(b)

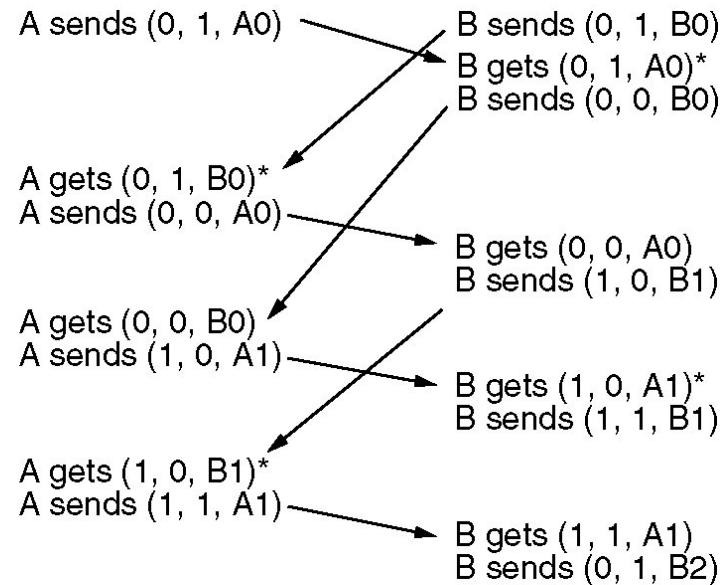
(c)

(d)

# Ví dụ giao thức cửa sổ trượt với kích thước là 1



(a)



(b)

Time

(a): Việc gửi nhận diễn ra bình thường theo đúng tuần tự

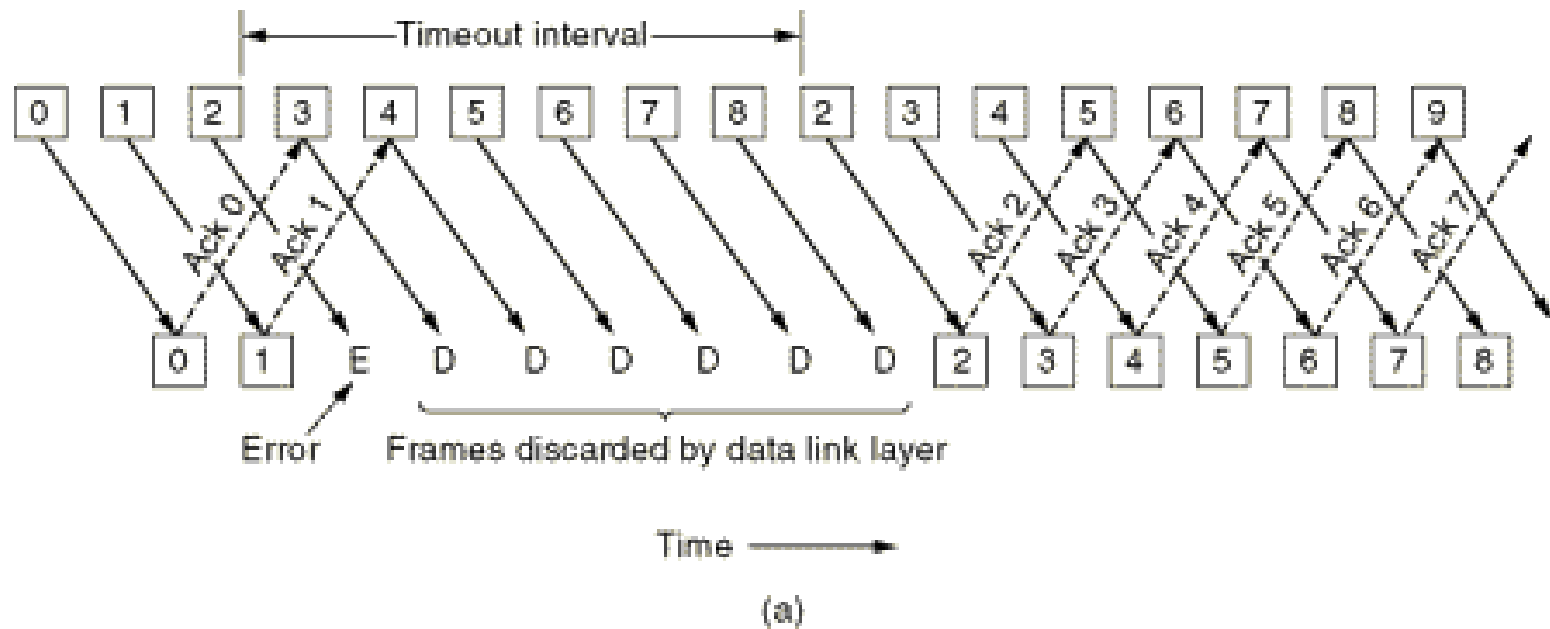
(b): Việc gửi nhận diễn ra theo một trình tự bất kỳ

## Giao thức Go-Back-N

---

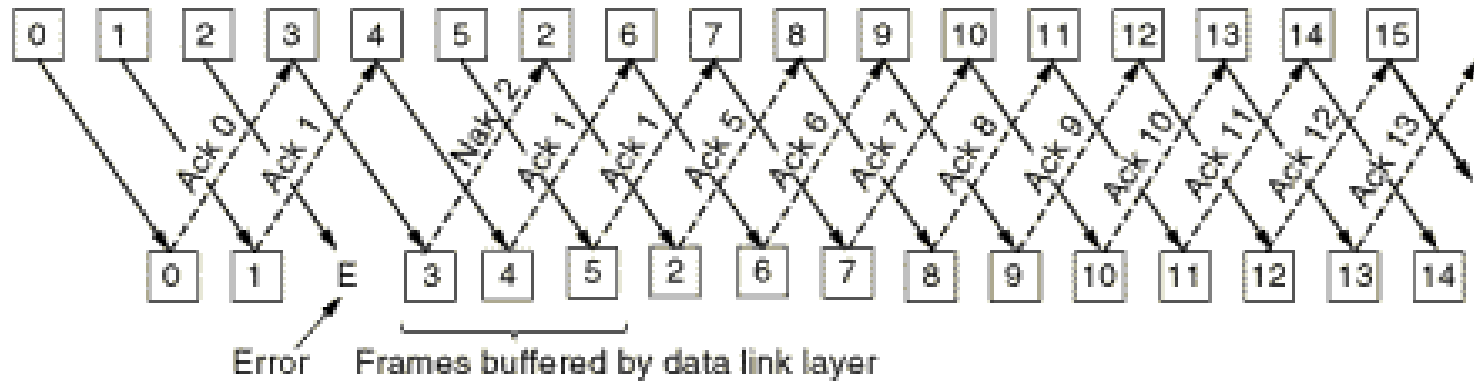
- Khi một khung bị lỗi. Bên nhận bỏ qua khung. Vì không một báo nhận nào gửi về cho bên gửi nên sự kiện quá thời gian xảy ra, bên gửi phải gửi lại khung bị lỗi và toàn bộ các khung phía sau nó.

# Giao thức Go-Back-N





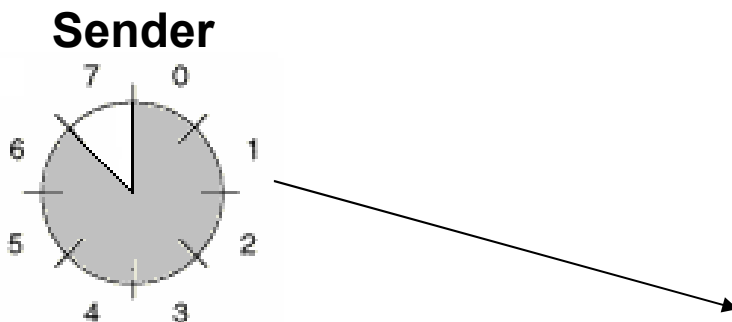
# Giao thức Selective Repeat



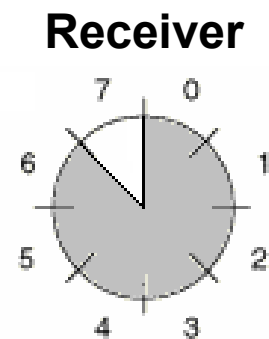
(b)

# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



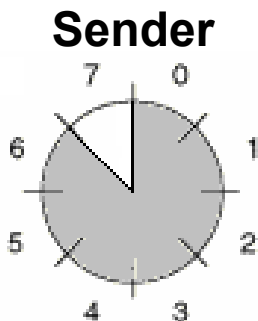
*Đã gửi và chờ bảo nhận  
các khung 0,1,2,3,4,5,6*



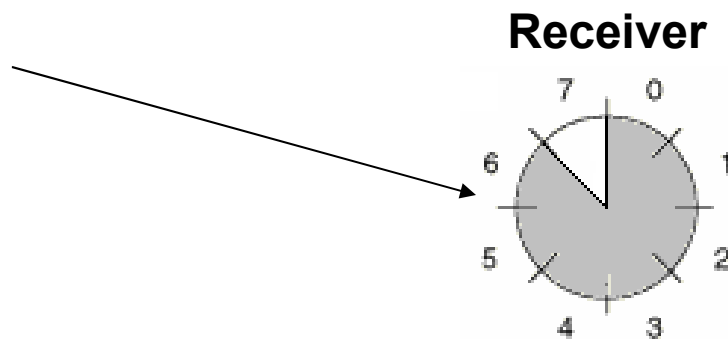
*Đang sẵn sàng chờ nhận  
các khung 0,1,2,3,4,5,6*

# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



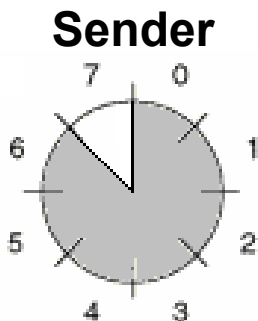
*Đã gửi và chờ bảo nhận  
các khung 0,1,2,3,4,5,6*



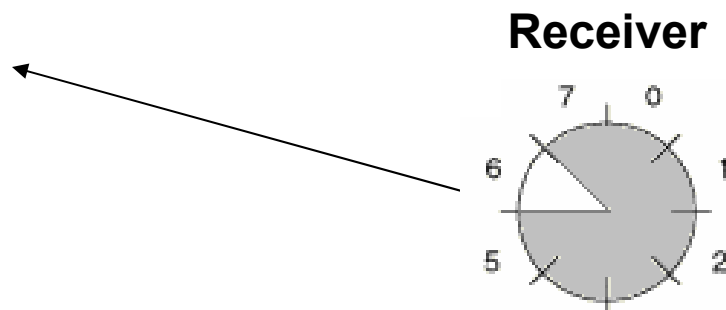
*Nhận các 0,1,2,3,4,5,6,  
Kiểm tra lỗi*

# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



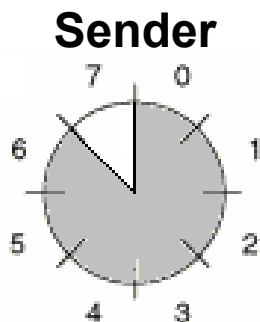
*Đã gửi và chờ bảo nhận các khung 0,1,2,3,4,5,6*



*Khung 0,1,2,3,4,5,6 không có lỗi,  
Gửi báo nhận cho các khung 0,1,2,3,4,5,6  
Sẵn sàng chờ nhận các khung 7,0,1,2,3,4,5*

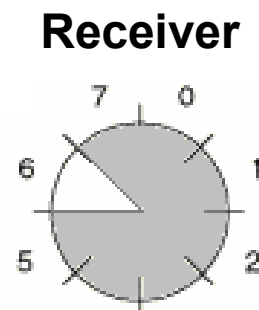
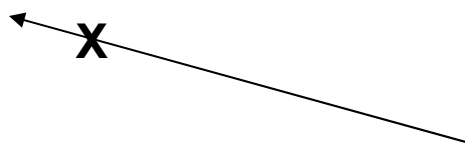
# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



*Đã gửi và chờ báo nhận các khung 0,1,2,3,4,5,6*

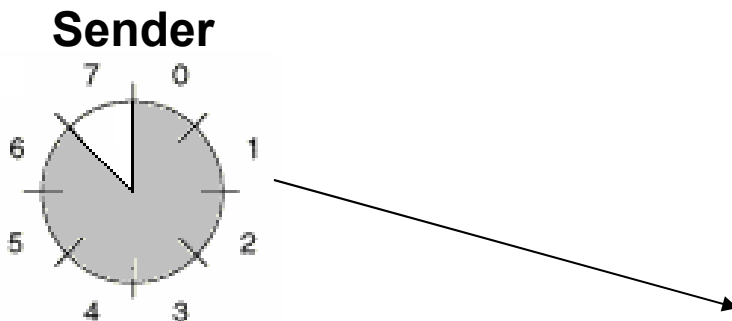
*Đường truyền bị lỗi  
Khung báo nhận không đến nơi*



*Khung 0,1,2,3,4,5,6 không có lỗi,  
Gửi báo nhận cho các khung 0,1,2,3,4,5,6  
Sẵn sàng chờ nhận các khung 7,0,1,2,3,4,5*

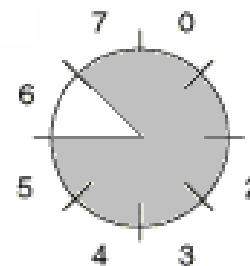
# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



*Quá thời hạn  
Gởi lại khung 0  
Chờ bảo nhận  
các khung 0,1,2,3,4,5,6*

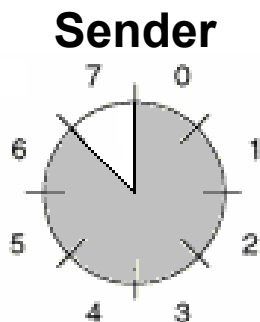
**Receiver**



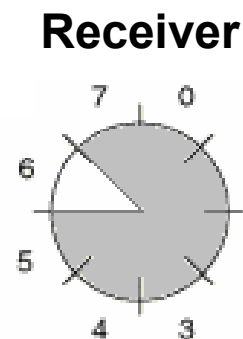
*Đang sẵn sàng chờ nhận  
các khung 7,0,1,2,3,4,5*

# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



*Quá thời hạn  
Gởi lại khung 0  
Chờ bảo nhận  
các khung 0,1,2,3,4,5,6*



*Khung 0 đến nơi,  
Là khung đang chờ nhận, đưa lên tầng  
mạng => **tầng mạng nhận 2 lần khung 0***

# Xác định kích thước cửa sổ trượt

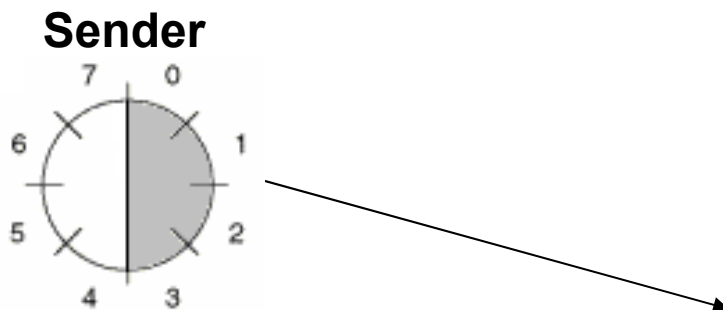
---

- Phải đảm bảo rằng cửa sổ nhận mới không đè chồng lên cửa sổ trước đó.
- Kích thước tối đa của cửa sổ nhận bằng một nửa khoảng đánh số thứ tự của khung.
  - Ví dụ:
    - Nếu dùng 3 bit để đánh số thứ tự khung từ 0 đến 7 thì kích thước tối đa của cửa sổ nhận là  $(7-0+1)/2 = 4$ .
    - Nếu dùng 4 bit để đánh số thứ tự khung từ 0 đến 15 thì kích thước tối đa của cửa sổ nhận là  $(15-0+1)/2 = 8$ .

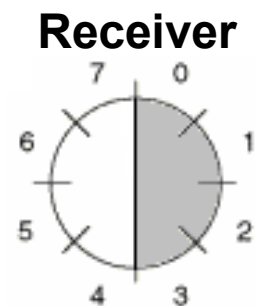


# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 4



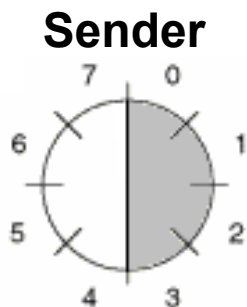
*Đã gửi và chờ báo nhận  
các khung 0,1,2,3,*



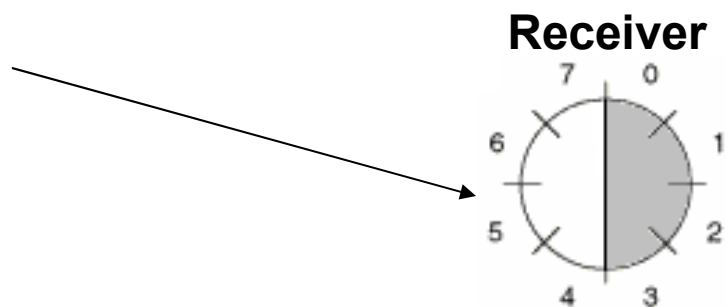
*Đang sẵn sàng chờ nhận  
các khung 0,1,2,3*

# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



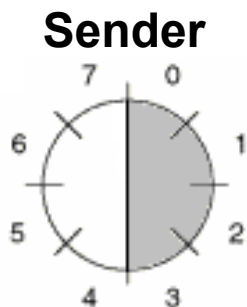
*Đã gửi và chờ báo nhận  
các khung 0,1,2,3*



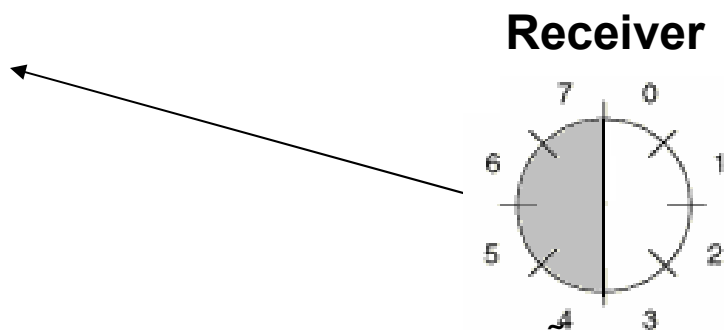
*Nhận các 0,1,2,3  
Kiểm tra lỗi*

# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



*Đã gửi và chờ báo nhận  
các khung 0,1,2,3*

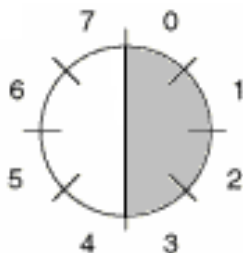


*Khung 0,1,2,3 không có lỗi,  
Gửi báo nhận cho các khung 0,1,2,3  
Sẵn sàng chờ nhận các khung 4,5,6,7*

# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7

**Sender**

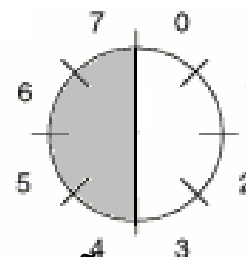


*Đã gửi và chờ báo nhận các khung 0,1,2,3*

*Đường truyền bị lỗi  
Khung báo nhận không đến nơi*

**X**

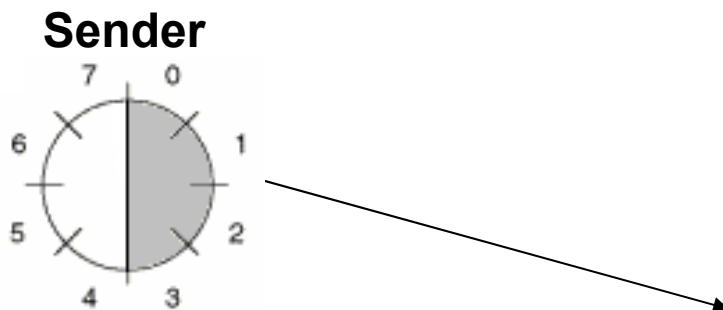
**Receiver**



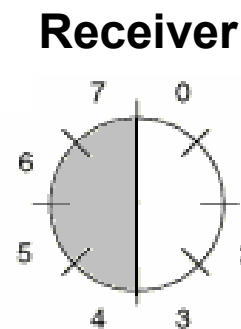
*Khung 0,1,2,3 không có lỗi,  
Gửi báo nhận cho các khung 0,1,2,3  
Sẵn sàng chờ nhận các khung 4,5,6,7*

# Xác định kích thước cửa sổ trượt

- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7



*Quá thời hạn  
Gởi lại khung 0  
Chờ bảo nhận  
các khung 0,1,2,3*

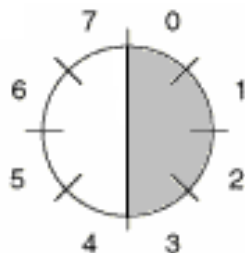


*Đang sẵn sàng chờ nhận  
các khung 4,5,6,7*

# Xác định kích thước cửa sổ trượt

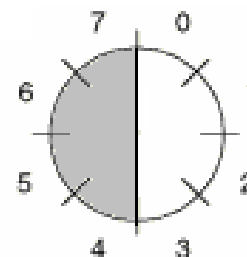
- Xét cửa sổ trượt sử dụng 3 bits để đánh chỉ số khung => Kích thước cửa sổ là 7

**Sender**



*Quá thời hạn  
Gởi lại khung 0  
Chờ bảo nhận  
các khung 0,1,2,3*

**Receiver**



*Khung 0 đến nơi, Là khung đã nhận  
=> **không đưa lên tầng mạng***

## Kích thước vùng đệm dữ liệu (buffer)

---

- Số lượng buffer chỉ cần bằng kích thước tối đa của cửa sổ nhận, không cần thiết phải bằng số lượng khung.
- Ví dụ: Nếu dùng 3 bit để đánh số thứ tự khung từ 0 đến 7 thì kích thước tối đa của cửa sổ nhận là  $(7-0+1)/2 = 4$  và số lượng buffer cần thiết cũng là 4.

## Thời điểm gửi báo nhận

---

- Piggy-back: Gói báo nhận vào khung dữ liệu của bên nhận
- Bên nhận không còn dữ liệu để gửi đi ?
  - Mỗi lần khung đến khởi động một timer
  - Time – out mà bên nhận không có dữ liệu để gửi => Gửi một khung báo nhận riêng



# **GIAO THỨC HDLC**

## **(High Level Data Link Control)**

# Các loại trạm (HDLC Station Types)

---

- **Primary station**
  - Điều khiển đường nối kết
  - Khung gởi đi là các lệnh
  - Duy trì nhiều nối kết luận lý đến các secondary station
- **Secondary station**
  - Chịu sự điều khiển của primary station
  - Các khung gởi đi là các trả lời
- **Combined station**
  - Có đặc tính của cả Primary station và Secondary station
  - Có thể gởi lệnh và trả lời

# Các cấu hình đường nối kết (HDLC Link Configurations)

---

- Không cân bằng (Unbalanced)
  - Một Primary station và một hoặc nhiều secondary stations
  - Hỗ trợ full duplex và half duplex
- Cân bằng (Balanced)
  - Gồm hai combined stations
  - Hỗ trợ full duplex và half duplex

# Các chế độ truyền tải (HDLC Transfer Modes )

---

- Normal Response Mode (NRM)
- Asynchronous Balanced Mode (ABM)
- Asynchronous Response Mode (ARM)

# Các chế độ truyền tải (HDLC Transfer Modes )

---

- Normal Response Mode (NRM)
  - Cấu hình không cân bằng
  - Primary khởi động cuộc truyền tải tới secondary
  - Secondary chỉ có thể truyền dữ liệu dưới dạng các trả lời cho các yêu cầu của primary
  - Được sử dụng trên các loại cáp nhiều sợi
  - Máy tính đóng vai trò primary
  - Terminals đóng vai trò secondary

# Các chế độ truyền tải (HDLC Transfer Modes )

---

- **Asynchronous Balanced Mode (ABM)**
  - Cấu hình cân bằng
  - Các trạm đều có thể khởi động một cuộc truyền tải mà không cần có phép
  - Được sử dụng rộng rãi

## HDLC Transfer Modes (3)

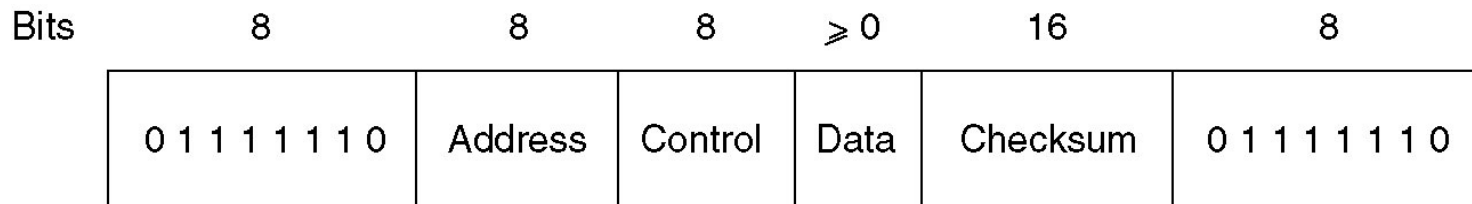
---

- Asynchronous Response Mode (ARM)
  - Cấu hình không cân bằng
  - Secondary có thể khởi động một cuộc truyền tải mà không cần xin phép từ primary
  - Primary đảm bảo về đường truyền
  - Ít được dùng

# Cấu trúc khung

---

- Truyền tải đồng bộ (Synchronous transmission)
- Tất cả các cuộc truyền tải đều sử dụng khung
- Một dạng khung cho tất cả các loại dữ liệu và điều khiển





Bits

8

8

8

 $\geq 0$ 

16

8

0 1 1 1 1 1 1 0	Address	Control	Data	Checksum	0 1 1 1 1 1 1 0
-----------------	---------	---------	------	----------	-----------------

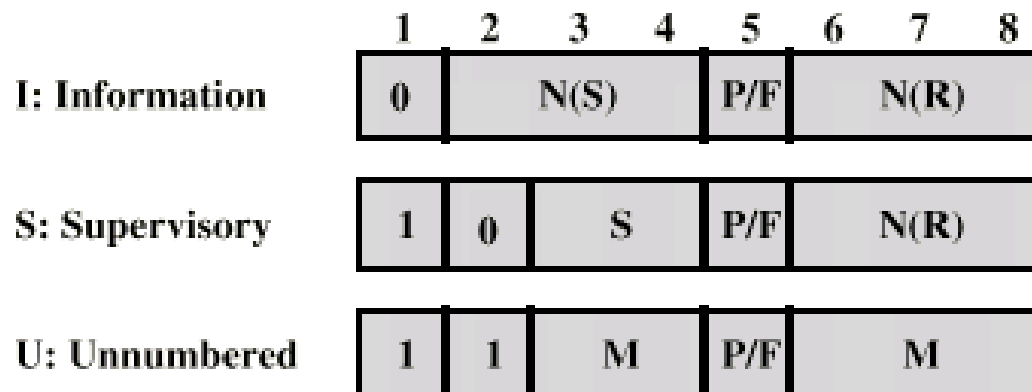
## Cấu trúc khung

- **Flag (8 bit):** 01111110 , Sử dụng kỹ thuật bít độn
- **Address (8 bit):** Vùng ghi địa chỉ để xác định máy phụ được phép truyền hay nhận khung.
- **Control (8bit):** Được dùng để xác định loại khung:
  - Thông tin (Information),
  - Điều khiển (Supervisory )
  - Không đánh số (Unnumbered).
- **Information(128-1024 bytes):** Vùng chứa dữ liệu cần truyền.
- **FCS (Frame Check Sequence- 8 bit)**
  - CRC-CCITT =  $X^{16} + X^{12} + X^5 + 1$

# Control Field

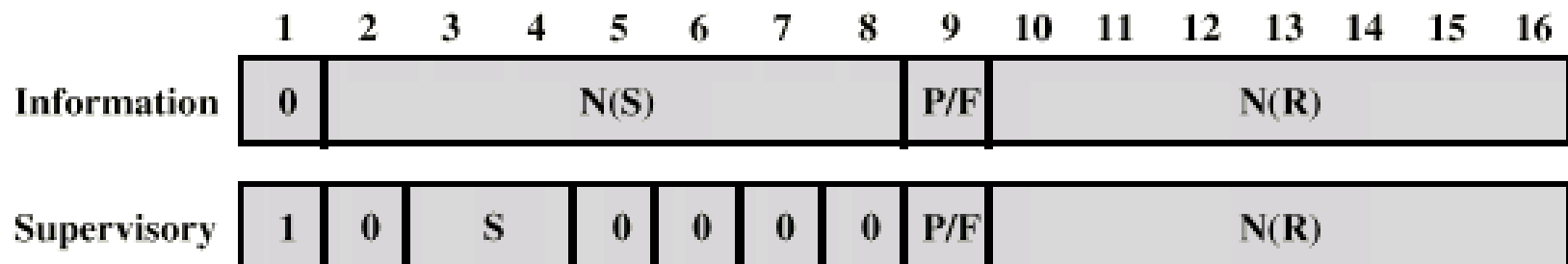
---

- Khác nhau tùy thuộc vào kiểu khung
  - Information :
    - Khung chứa dữ liệu được truyền đi
    - Đồng thời chứa thông tin báo nhận (piggy-back)
  - Supervisory: Khung báo nhận khi không còn dữ liệu để gửi ngược lại
  - Unnumbered: Dùng để điều khiển nối kết



**N(S)** = Send sequence number  
**N(R)** = Receive sequence number  
**S** = Supervisory function bits  
**M** = Unnumbered function bits  
**P/F** = Poll/final bit

(c) 8-bit control field format



(d) 16-bit control field format

## Poll/Final Bit

---

- Được sử dụng tùy thuộc vào ngữ cảnh
- Nếu là khung lệnh
  - Có ý nghĩa là Poll
  - Giá trị 1 để yêu cầu bên kia trả lời
- Nếu là khung trả lời
  - Có ý nghĩa là Final
  - Giá trị 1 để biểu thị rằng nó kết thúc việc gửi

# Supervisory function bits

---

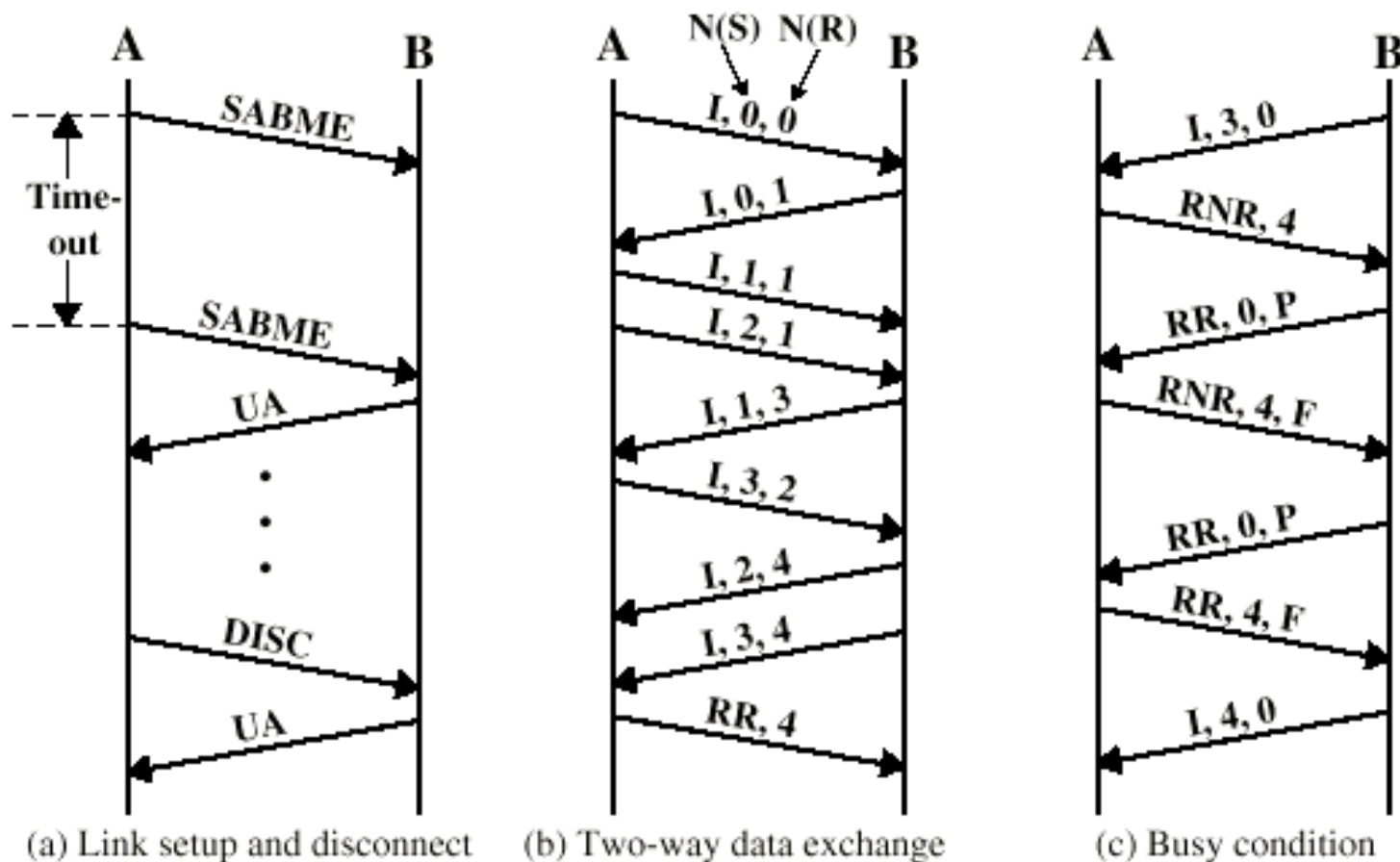
SS=00	RR (Receive Ready), là khung báo nhận, thông báo sẵn sàng nhận dữ liệu, đã nhận tốt đến khung Next-1, và đang đợi nhận khung Next. Được dùng đến khi không còn dữ liệu gửi từ chiều ngược lại để vừa làm báo nhận (figgyback)
SS=01	REJ (Reject): đây là một khung báo không nhận (negative acknowledge), yêu cầu gửi lại các khung, từ khung Next.
SS=10	RNR (Receive Not Ready): thông báo không sẵn sàng nhận tin, đã nhận đến đến khung thứ Next-1, chưa sẵn sàng nhận khung Next
SS=11	SREJ (Selective Reject): yêu cầu gửi lại một khung có số thứ tự là Next

# Unnumbered Function Bits

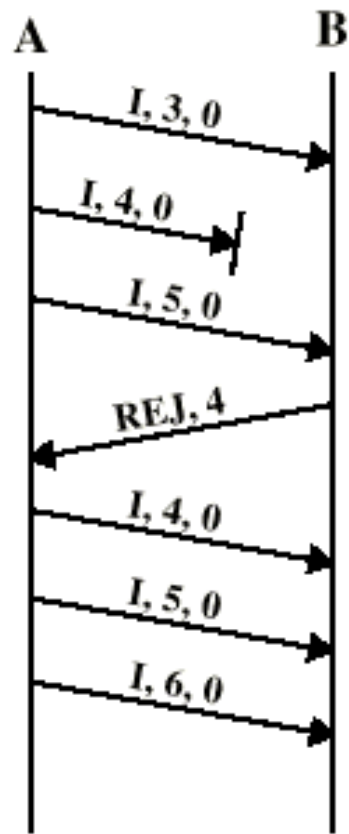
---

1111P100	Lệnh này dùng để thiết lập chế độ truyền tải SABM (Set Asynchronous Balanced Mode).
1100P001	Lệnh này dùng để thiết lập chế độ truyền tải SNRM (Set Normal Response Mode).
1111P000	Lệnh này dùng để thiết lập chế độ truyền tải SARM (Set Asynchronous Response Mode).
1100P010	Lệnh này để yêu cầu xóa nối kết DISC (Disconnect).
1100F110	UA (Unnumbered Acknowledgment). Được dùng bởi các trạm phụ để báo với trạm chính rằng nó đã nhận và chấp nhận các lệnh loại U ở trên.
1100F001	CMDR/FRMR (Command Reject/Frame Reject). Được dùng bởi trạm phụ để báo rằng nó không chấp nhận một lệnh mà nó đã nhận chính xác.

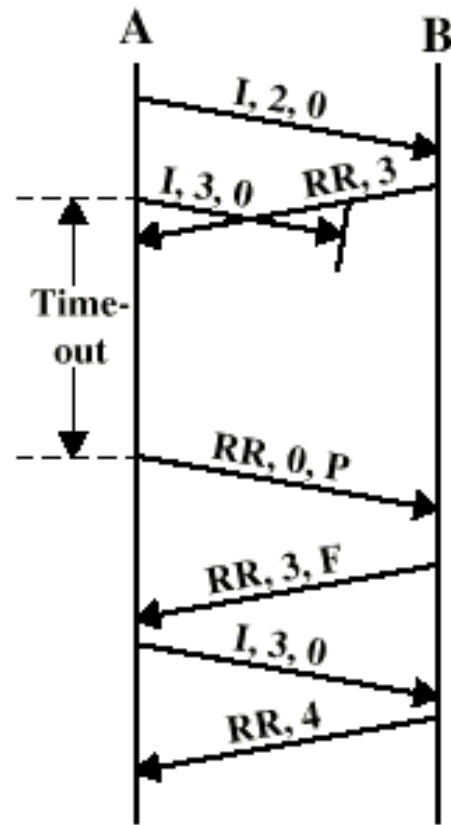
# Một số kịch bản



# Một số kịch bản



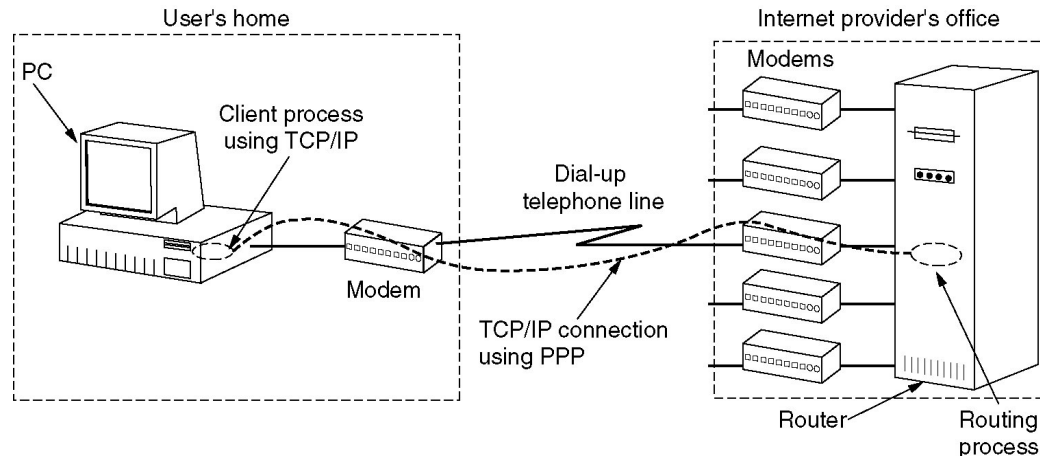
(d) Reject recovery



(e) Timeout recovery



# Giao thức Điểm nối điểm (PPP- Point-to-Point Protocol)



- Cho phép truyền tải thông tin giữa các router trên mạng hay để cho phép nối các máy tính người dùng vào mạng của nhà cung cấp dịch vụ Internet (ISP)
- Giao thức điều khiển đường truyền LCP ( Link Control Protocol).
- Giao thức thương lượng về các tùy chọn tầng mạng NCP (Network Control Protocol)