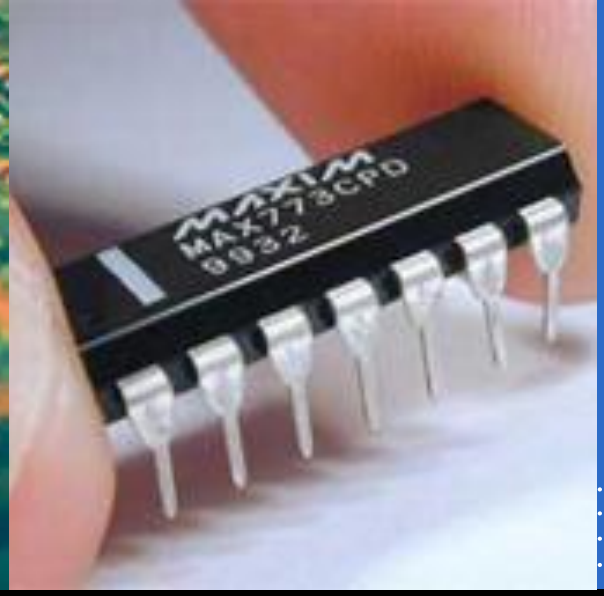
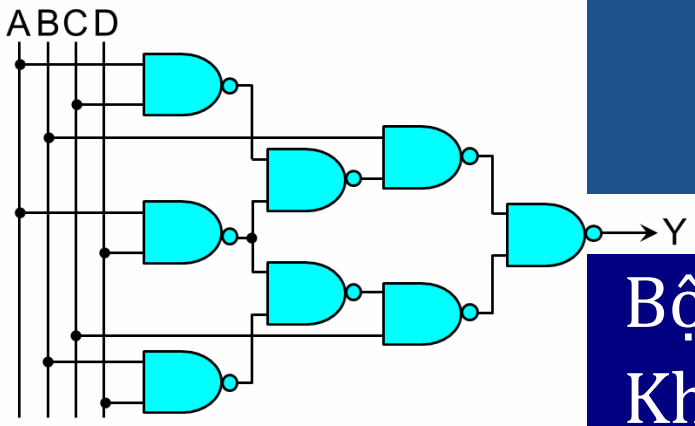
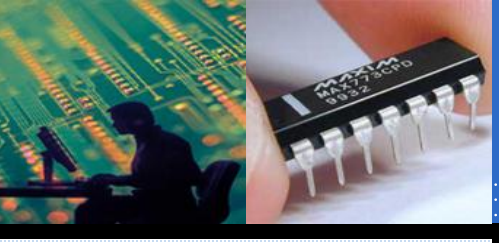


ĐIỆN TỬ SỐ - CHƯƠNG 4



TỔNG HỢP VÀ PHÂN TÍCH MẠCH TỔ HỢP

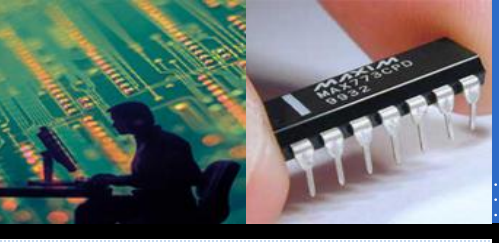
Bộ môn Kỹ thuật vi xử lý
Khoa Vô tuyến điện tử
Học viện Kỹ thuật quân sự



NỘI DUNG CHƯƠNG 4

MẠCH TỔ HỢP

1. KIẾN THỨC CHUNG
2. TỔNG HỢP MẠCH TỔ HỢP
3. PHÂN TÍCH MẠCH TỔ HỢP
4. CÁC MẠCH TỔ HỢP THÔNG DỤNG

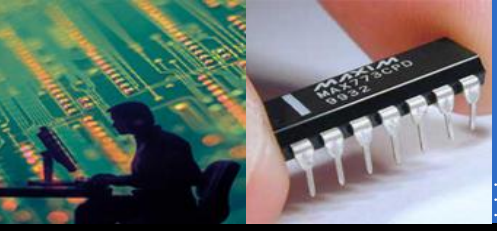


1. KIẾN THỨC CHUNG

1.1. Giới thiệu về mạch tổ hợp

1.2. Các phương trình mạch tổ hợp

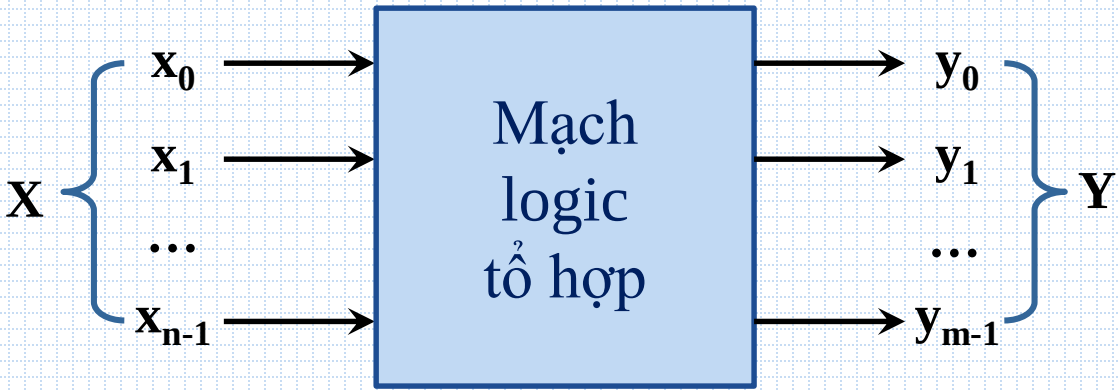
1.3. Các phương pháp biểu diễn



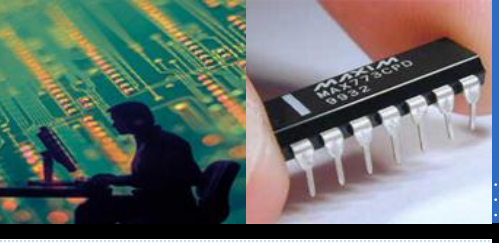
1.1. Giới thiệu về mạch tổ hợp

Mạch logic tổ hợp (mạch số tổ hợp, mạch tổ hợp) là một mạch số có tổ hợp tín hiệu ra tại một thời điểm chỉ phụ thuộc tổ hợp các tín hiệu vào tại cùng thời điểm đó

- **Sơ đồ khối mạch logic tổ hợp**

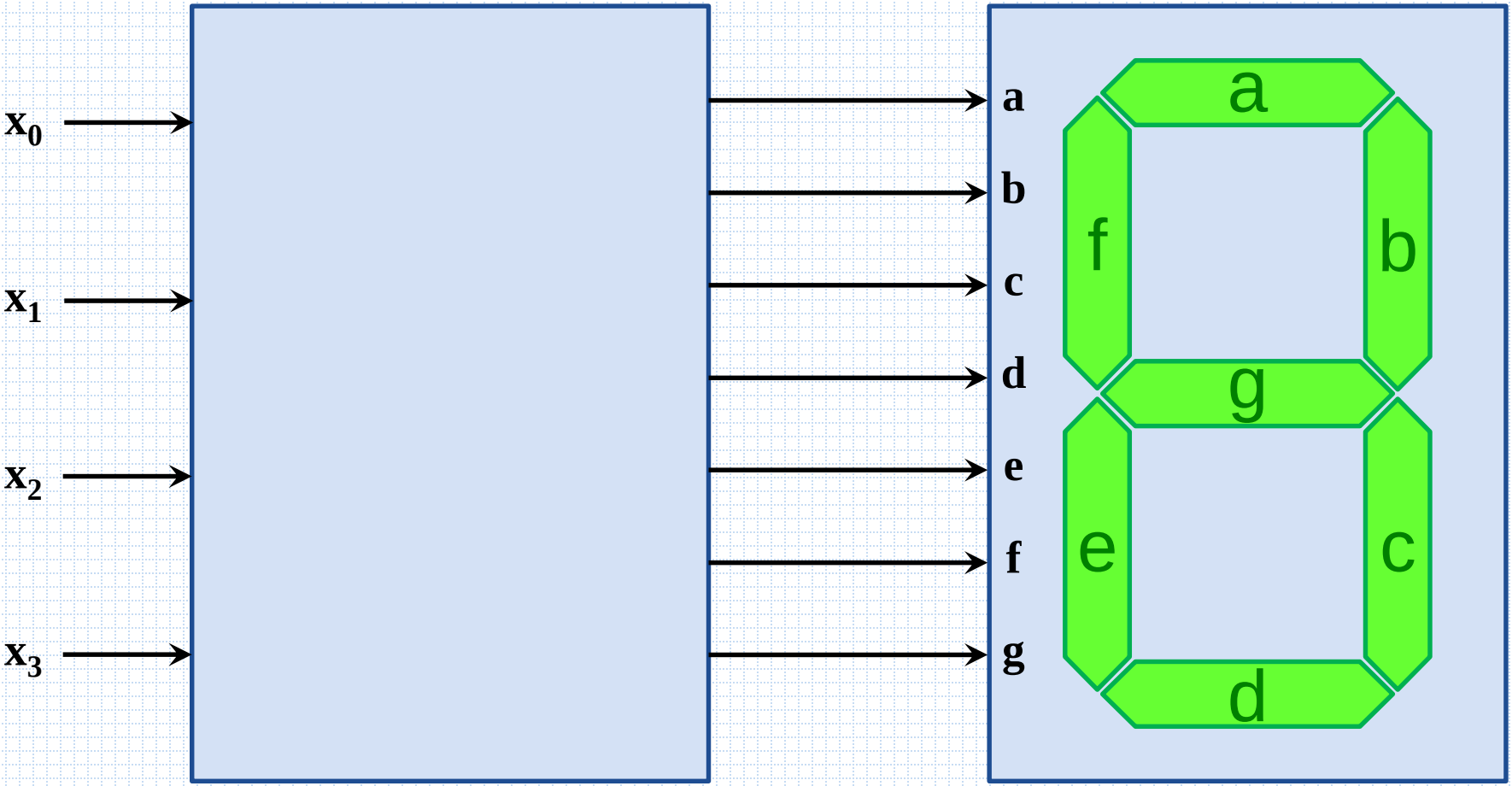


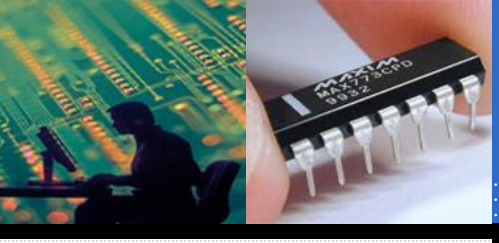
- **Đường truyền của mạch:** đường đi dài nhất (qua nhiều phần tử logic nhất) của tín hiệu từ đầu vào tới đầu ra
- **Cấp của mạch:** số lượng các phần tử logic trên đường truyền



1.1. Giới thiệu về mạch tổ hợp

❑ Ví dụ minh họa mạch logic tổ hợp



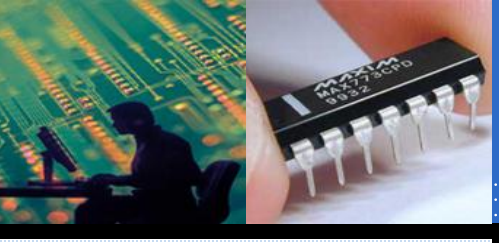


1. KIẾN THỨC CHUNG

1.1. Giới thiệu về mạch tổ hợp

1.2. Các phương trình mạch tổ hợp

1.3. Các phương pháp biểu diễn



1.2. Các phương trình mạch tổ hợp

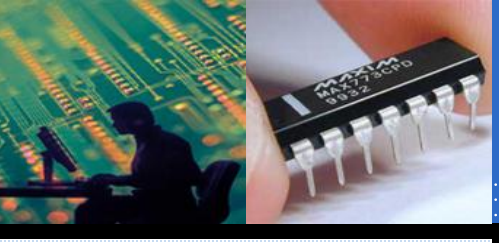
- **Phương trình đặc trưng**

$$Y(t) = F[X(t)]$$

$$Y = F[X]$$

- **Phương trình logic**

$$\begin{cases} y_0 = f_0(x_0, x_1, \ldots, x_{n-1}) \\ y_1 = f_1(x_0, x_1, \ldots, x_{n-1}) \\ \ldots \\ y_{m-1} = f_{m-1}(x_0, x_1, \ldots, x_{n-1}) \end{cases}$$

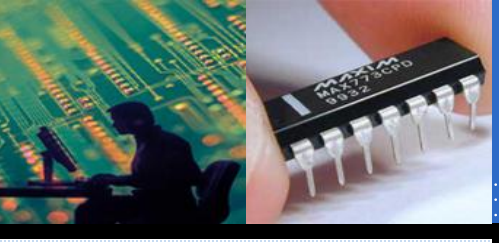


1. KIẾN THỨC CHUNG

1.1. Giới thiệu về mạch tổ hợp

1.2. Các phương trình mạch tổ hợp

1.3. Các phương pháp biểu diễn



1.3. Các phương pháp biểu diễn

- **Phương pháp bảng giá trị hàm**

  Thường được sử dụng để mô hình hóa cho các bài toán thực tế

- **Phương pháp bảng Karnaugh**

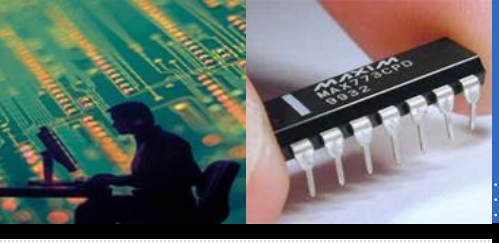
  Thường được sử dụng để đơn giản và tối thiểu hàm

- **Phương pháp đại số**

  Thường được sử dụng để đơn giản và tối thiểu hàm

- **Phương pháp sơ đồ mạch điện cổng**

  Thường được sử dụng để mô hình hóa cho các bài toán thực tế

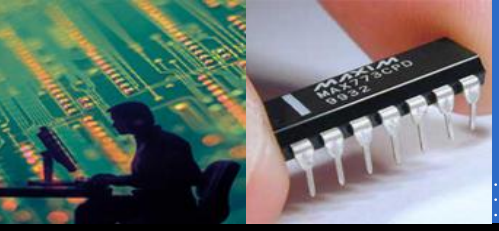


2. TỔNG HỢP MẠCH TỔ HỢP

2.1. Các bước tổng hợp mạch

2.2. Tổng hợp mạch một đầu ra

2.3. Tổng hợp mạch nhiều đầu ra



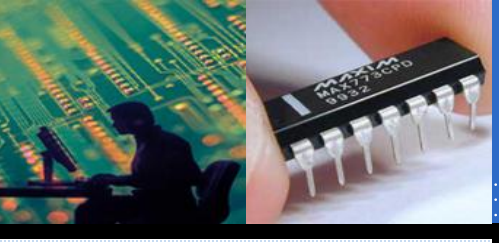
2.1. Các bước tổng hợp mạch

Mô tả quan hệ logic bằng các phương pháp biểu diễn (hệ) hàm logic

TỔNG HỢP MẠCH LOGIC TỔ HỢP

- Bước 1. Phân tích bài toán logic
- Bước 2. Mô tả quan hệ logic
- Bước 3. Tối thiểu hàm logic
- Bước 4. Biến đổi đại số logic
- Bước 5. Vẽ sơ đồ mạch logic

Biến đổi đại số logic biểu thức tối thiểu ở bước 3 để thu được biểu thức logic tối ưu tùy thuộc đặc điểm yêu cầu thực tế (ưu tiên về số cấp của mạch, ưu tiên về số cổng logic hay ưu tiên về loại cổng logic sử dụng…)

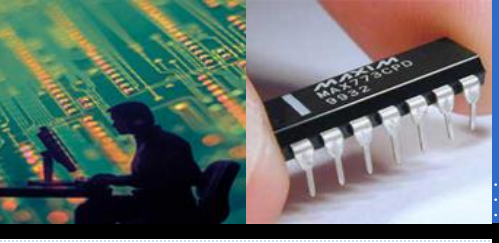


2. TỔNG HỢP MẠCH TỔ HỢP

2.1. Các bước tổng hợp mạch

2.2. Tổng hợp mạch một đầu ra

2.3. Tổng hợp mạch nhiều đầu ra



2.2. Tổng hợp mạch một đầu ra

Ví dụ 1: Tổng hợp mạch logic tổ hợp có bốn đầu vào chọn đa số bit 1

Bước 1 Mạch có 4 đầu vào ký hiệu A, B, C, D; 1 đầu ra ký hiệu Y
Đầu ra Y bằng 1 khi quá nửa số đầu vào A, B, C, D bằng 1

Bước 2 Xây dựng bảng giá trị hàm >> …

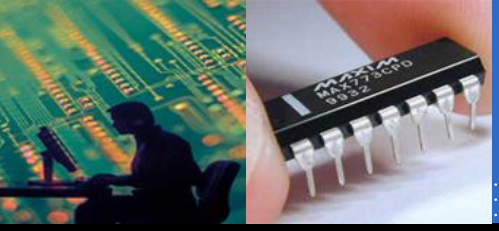
Bước 3 Tối thiểu hàm logic >> Sử dụng phương pháp bảng Karnaugh

Y

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	0	1	1	1
10	0	0	1	0

Biểu thức logic

$Y = ABC + ACD + ABD + BCD$



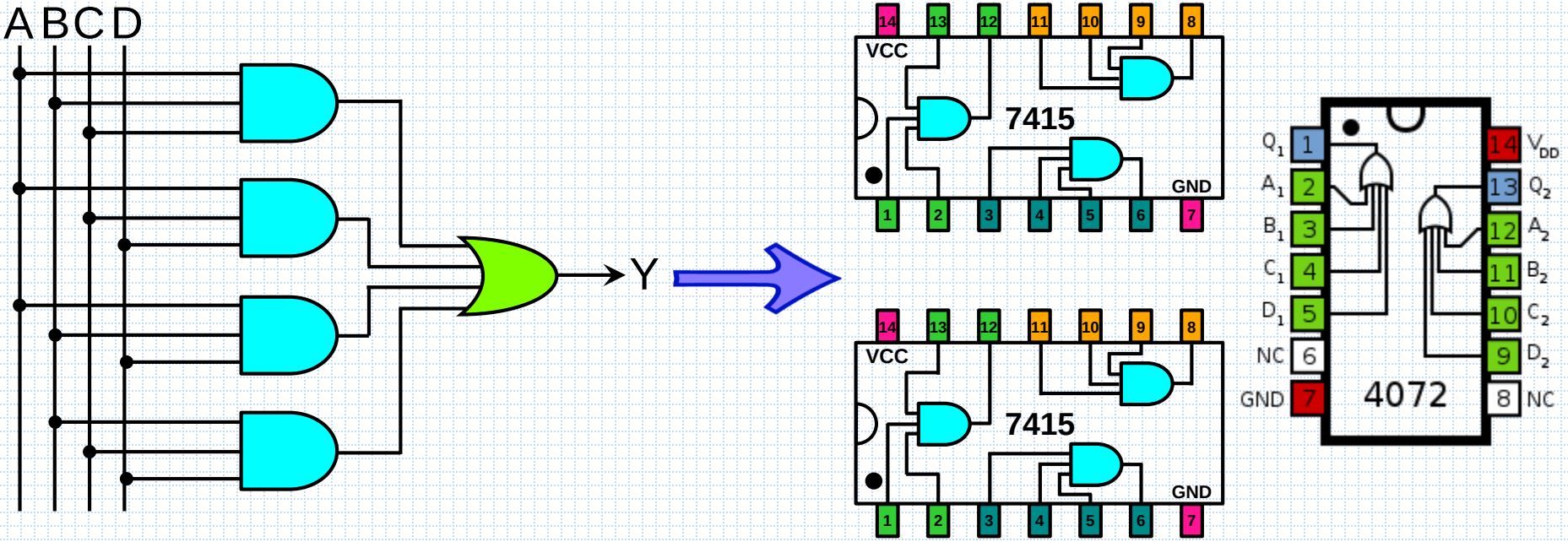
2.2. Tổng hợp mạch một đầu ra

Ví dụ 1: Tổng hợp mạch logic tổ hợp có bốn đầu vào chọn đa số bit 1

Bước 4 **Tr.h 1:** Các cổng logic không bị hạn chế về số đầu vào

$$Y = ABC + ACD + ABD + BCD \quad (1)$$

Bước 5



2.2. Tổng hợp mạch một đầu ra

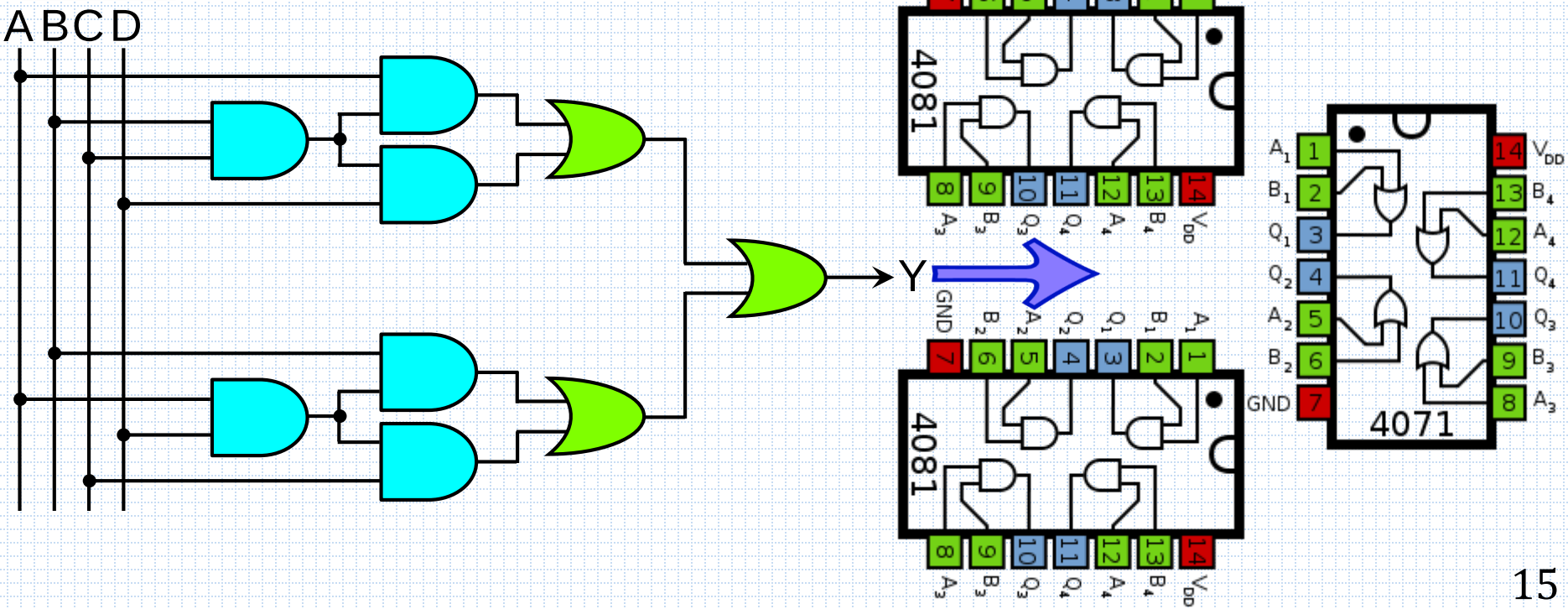
VD1: Tổng hợp mạch logic tổ hợp có bốn đầu vào chọn đa số bit 1

Bước 4 **Tr.h 2:** Chỉ sử dụng các cổng logic có hai đầu vào

(1) → $Y = ABC + ACD + ABD + BCD$

$Y = [A\cdot(B\cdot C) + (B\cdot C)\cdot D] + [(A\cdot D)\cdot B + (A\cdot D)\cdot C]$ (2)

Bước 5



2.2. Tổng hợp mạch một đầu ra

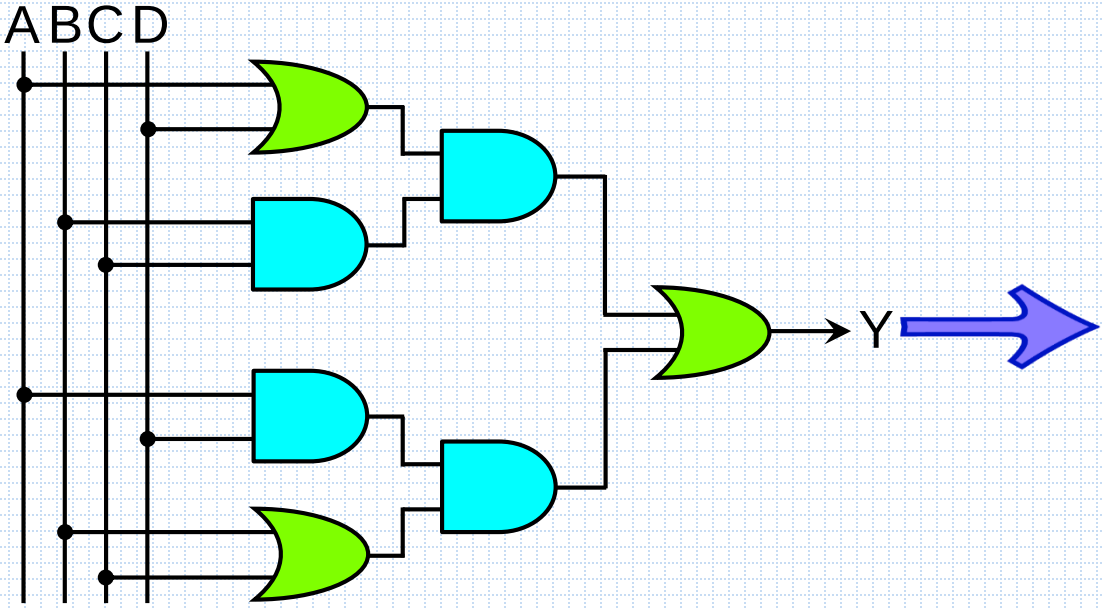
VD1: Tổng hợp mạch logic tổ hợp có bốn đầu vào chọn đa số bit 1

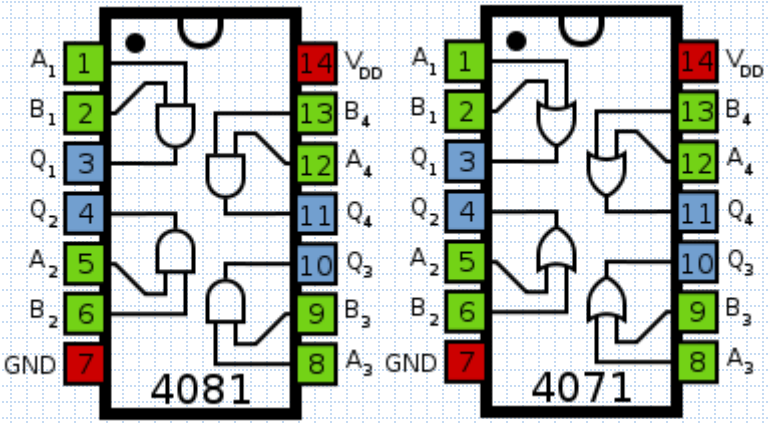
Bước 4 **Tr.h 3:** Chỉ sử dụng các cổng logic có hai đầu vào, tối ưu hóa

(2) →

$$Y = [A\cdot(B\cdot C) + (B\cdot C)\cdot D] + [(A\cdot D)\cdot B + (A\cdot D)\cdot C]$$
$$= (A + D)\cdot(B\cdot C) + (A\cdot D)\cdot(B + C) \quad (3)$$

Bước 5





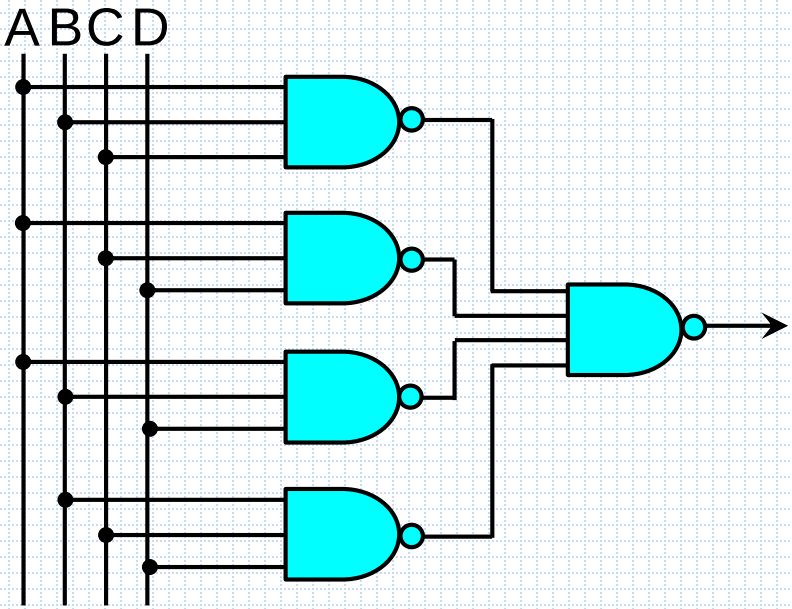
2.2. Tổng hợp mạch một đầu ra

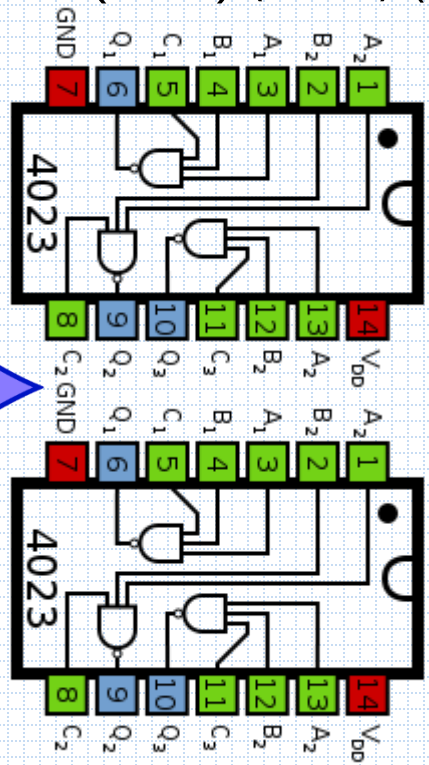
VD1: Tổng hợp mạch logic tổ hợp có bốn đầu vào chọn đa số bit 1

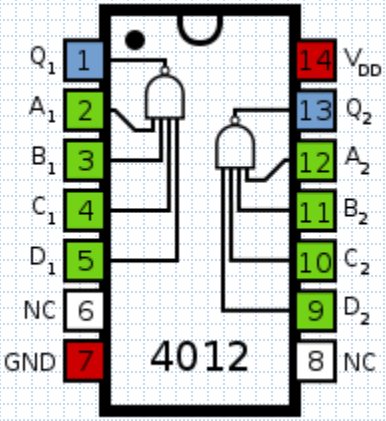
Bước 4 **Tr.h 4:** Chỉ sử dụng các cổng NAND (NOR)

(1) $Y = ABC + ACD + ABD + BCD$

Bước 5 $\overline{Y} = \overline{ABC + ACD + ABD + BCD} = \overline{(\overline{ABC})\cdot(\overline{ACD})\cdot(\overline{ABD})\cdot(\overline{BCD})}$ (4)







2.2. Tổng hợp mạch một đầu ra

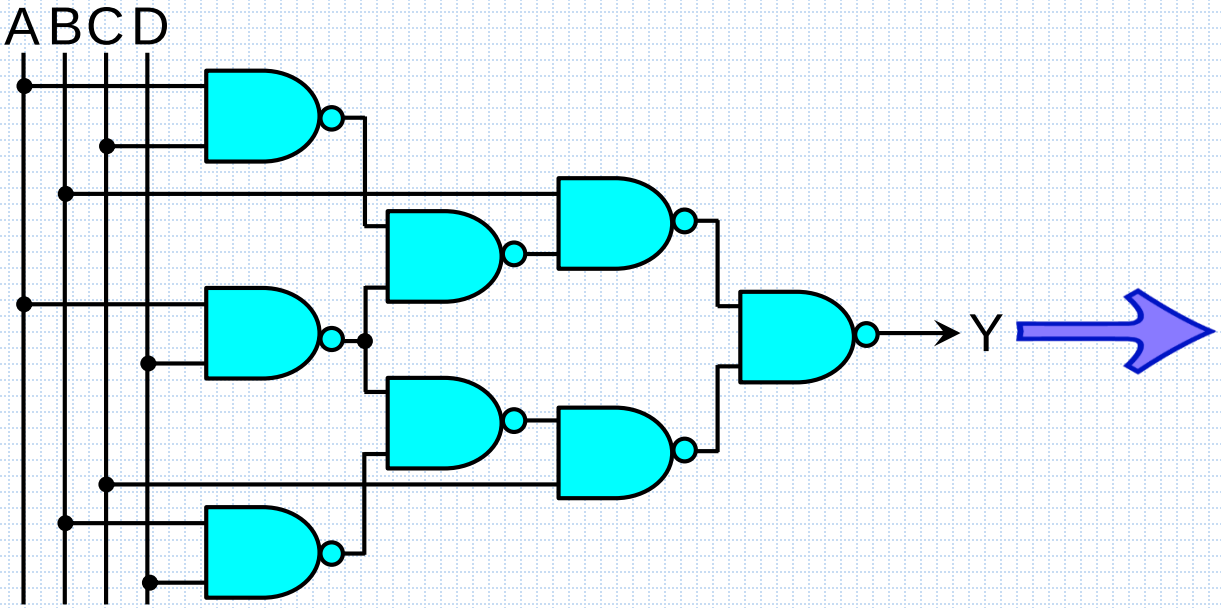
VD1: Tổng hợp mạch logic tổ hợp có bốn đầu vào chọn đa số bit 1

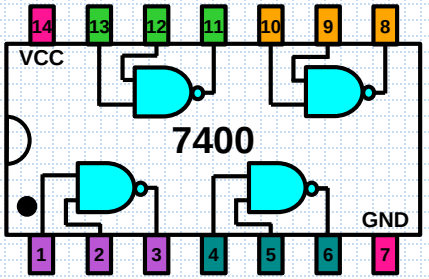
Bước 4 **Tr.h 5:** Chỉ SD các cổng NAND (NOR) 2 đầu vào, tối ưu hóa

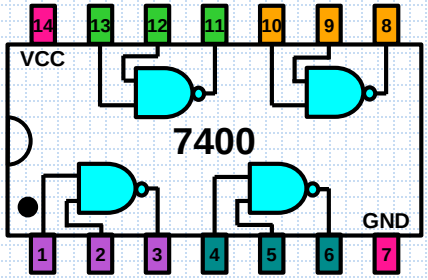
(1) → $Y = ABC + ACD + ABD + BCD = B(AC + AD) + C(AD + BD)$

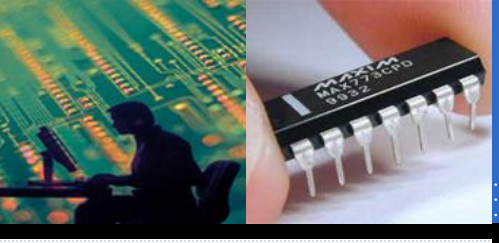
$$Y = \overline{B \cdot (\overline{AC} \cdot \overline{AD}) \cdot C \cdot (\overline{AD} \cdot \overline{BD})} \quad (5)$$

Bước 5







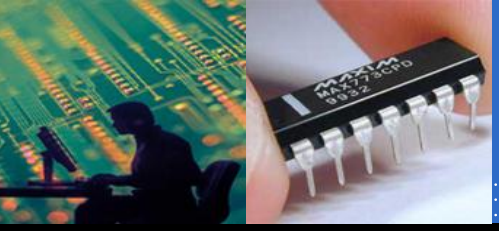


2. TỔNG HỢP MẠCH TỔ HỢP

2.1. Các bước tổng hợp mạch

2.2. Tổng hợp mạch một đầu ra

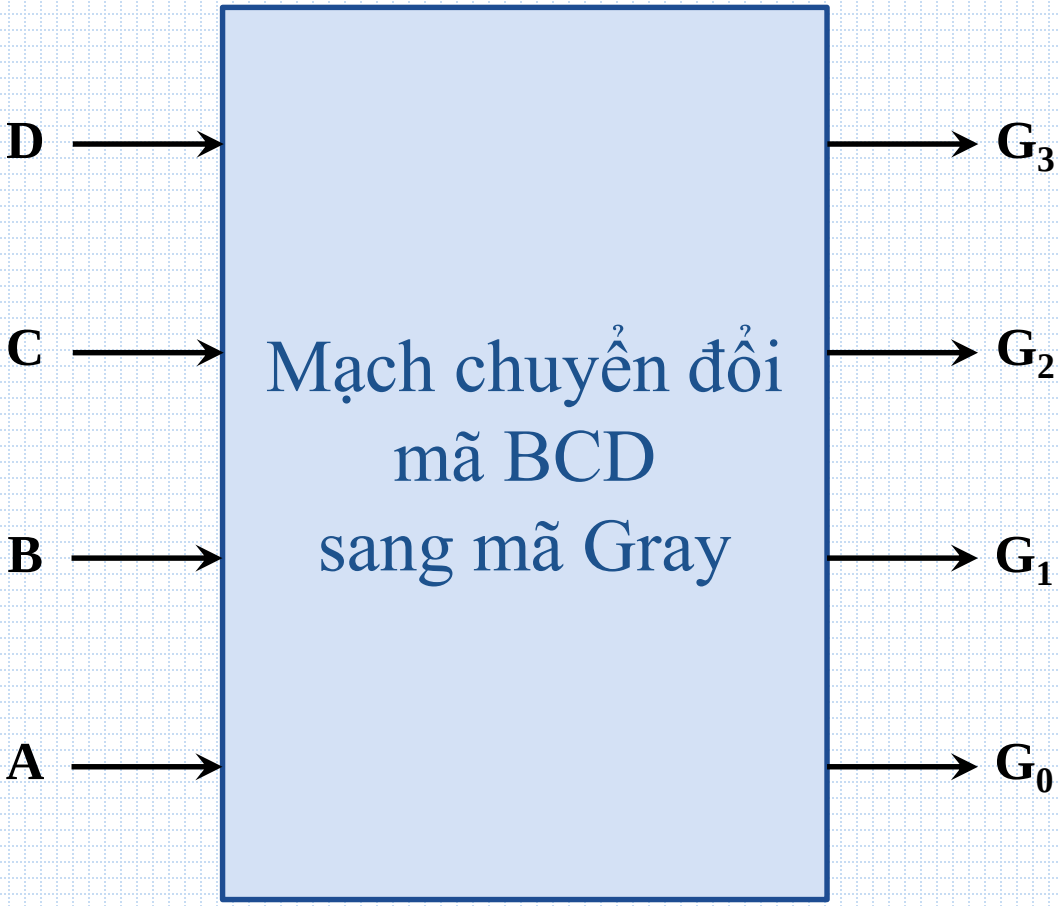
2.3. Tổng hợp mạch nhiều đầu ra



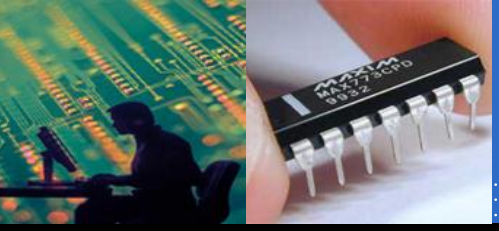
2.3. Tổng hợp mạch nhiều đầu ra

Ví dụ 2: Tổng hợp mạch logic tổ hợp chuyển đổi mã từ BCD sang Gray

Bước 1 ➢ **Phân tích bài toán logic**



- Mạch có 4 đầu vào A, B, C, D; 4 đầu ra G_3, G_2, G_1, G_0
- Ứng với mỗi tổ hợp 4 bit đầu vào mã BCD sẽ cho tương ứng một tổ hợp 4 bit đầu ra mã Gray

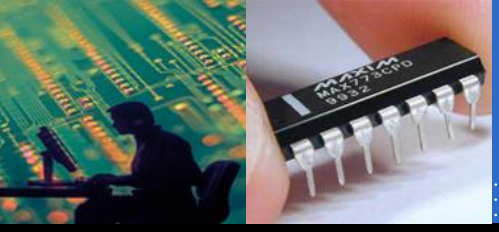


2.3. Tổng hợp mạch nhiều đầu ra

Ví dụ 2: Tổng hợp mạch logic tổ hợp chuyển đổi mã từ BCD sang Gray

Bước 2 ➢ **Mô tả quan hệ logic:** xây dựng bảng giá trị hàm

D	C	B	A	G_3	G_2	G_1	G_0	D	C	B	A	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0
0	0	0	1	0	0	0	1	1	0	0	1	1	1	0	1
0	0	1	0	0	0	1	1	1	0	1	0	x	x	x	x
0	0	1	1	0	0	1	0	1	0	1	1	x	x	x	x
0	1	0	0	0	1	1	0	1	1	0	0	x	x	x	x
0	1	0	1	0	1	1	1	1	1	0	1	x	x	x	x
0	1	1	0	0	1	0	1	1	1	1	0	x	x	x	x
0	1	1	1	0	1	0	0	1	1	1	1	x	x	x	x



2.3. Tổng hợp mạch nhiều đầu ra

Bước 3 ➢ **Tối thiểu hàm logic** >> Sử dụng phương pháp bảng Karnaugh

G_3

DC \ BA	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	x	x	x	x
10	1	1	x	x

$G_3 = D$

G_2

DC \ BA	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	x	x	x	x
10	1	1	x	x

$G_2 = C + D$

G_1

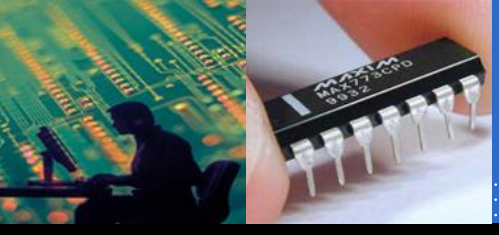
DC \ BA	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	x	x	x	x
10	0	0	x	x

$G_1 = C\overline{B} + \overline{C}B$

G_0

DC \ BA	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	x	x	x	x
10	0	1	x	x

$G_0 = B\overline{A} + \overline{B}A$



2.3. Tổng hợp mạch nhiều đầu ra

Bước 4 ➢ **Biến đổi đại số** tối ưu hệ hàm logic tối thiểu

$G_3 = D$ $G_2 = D + C$ $G_1 = C\overline{B} + \overline{C}B = C \oplus B$ $G_0 = B\overline{A} + \overline{B}A = B \oplus A$

❑ **Phương án 1**: Tối ưu số cổng nhưng chưa tối ưu số IC

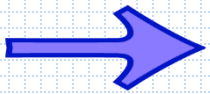
Next >>

❑ **Phương án 2**: Tối ưu số IC sử dụng $G_3 = D = D \oplus 0$

Next >>

G_2

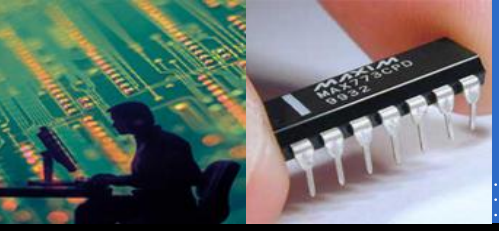
DC \ BA	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	x	x	x	x
10	1	1	x	x



G_2

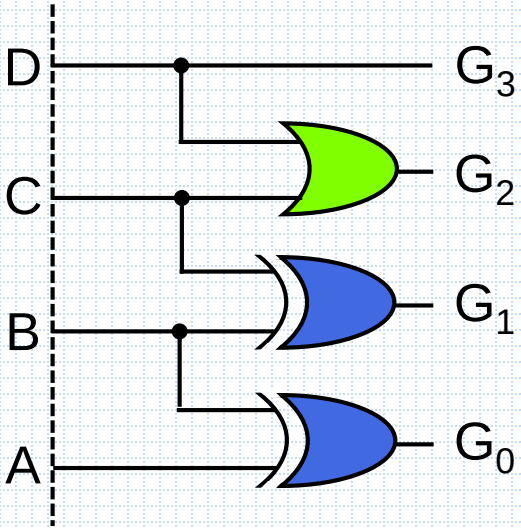
DC \ BA	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	0	0

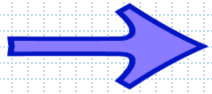
$G_2 = C\overline{D} + \overline{C}D = D \oplus C$

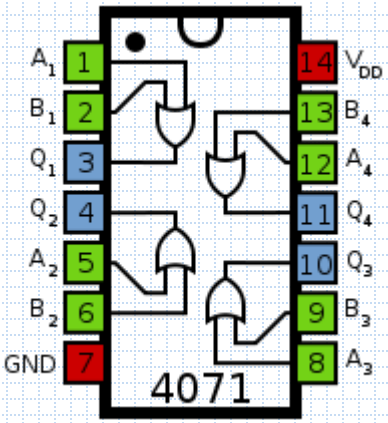


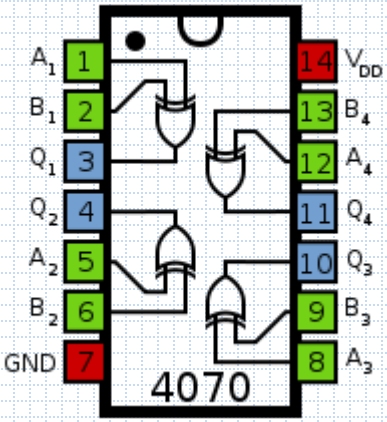
2.3. Tổng hợp mạch nhiều đầu ra

Bước 5 Vẽ sơ đồ mạch logic (Phương án 1)

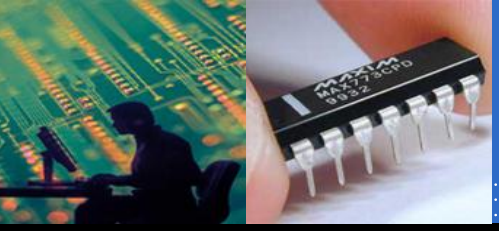






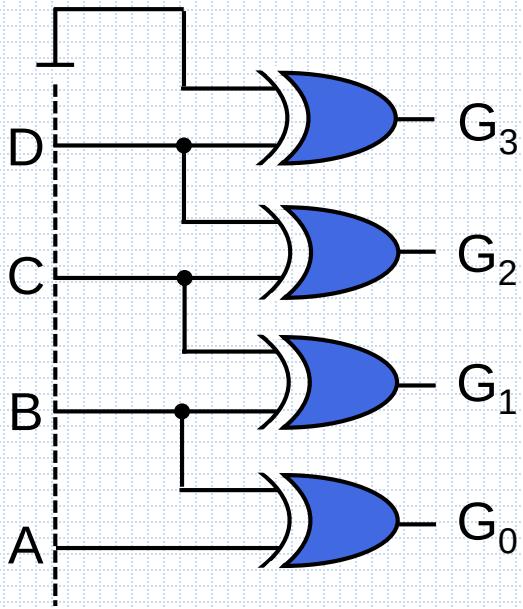


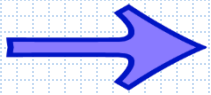
<< Back

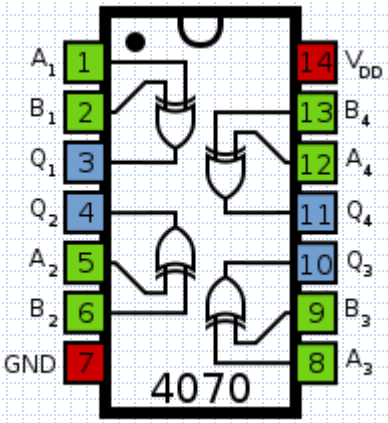


3.3. Tổng hợp mạch nhiều đầu ra

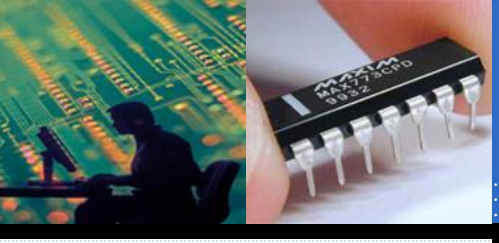
Bước 5 **Vẽ sơ đồ mạch logic** (Phương án 2)







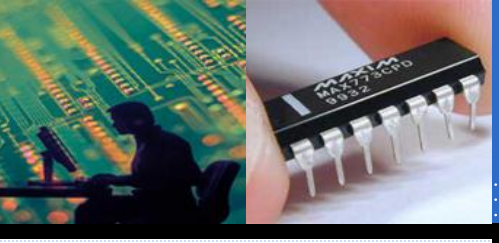
<< Back



3. PHÂN TÍCH MẠCH TỔ HỢP

3.1. Phân tích tĩnh mạch tổ hợp

3.2. Phân tích động mạch tổ hợp



3.1. Phân tích tĩnh mạch tổ hợp

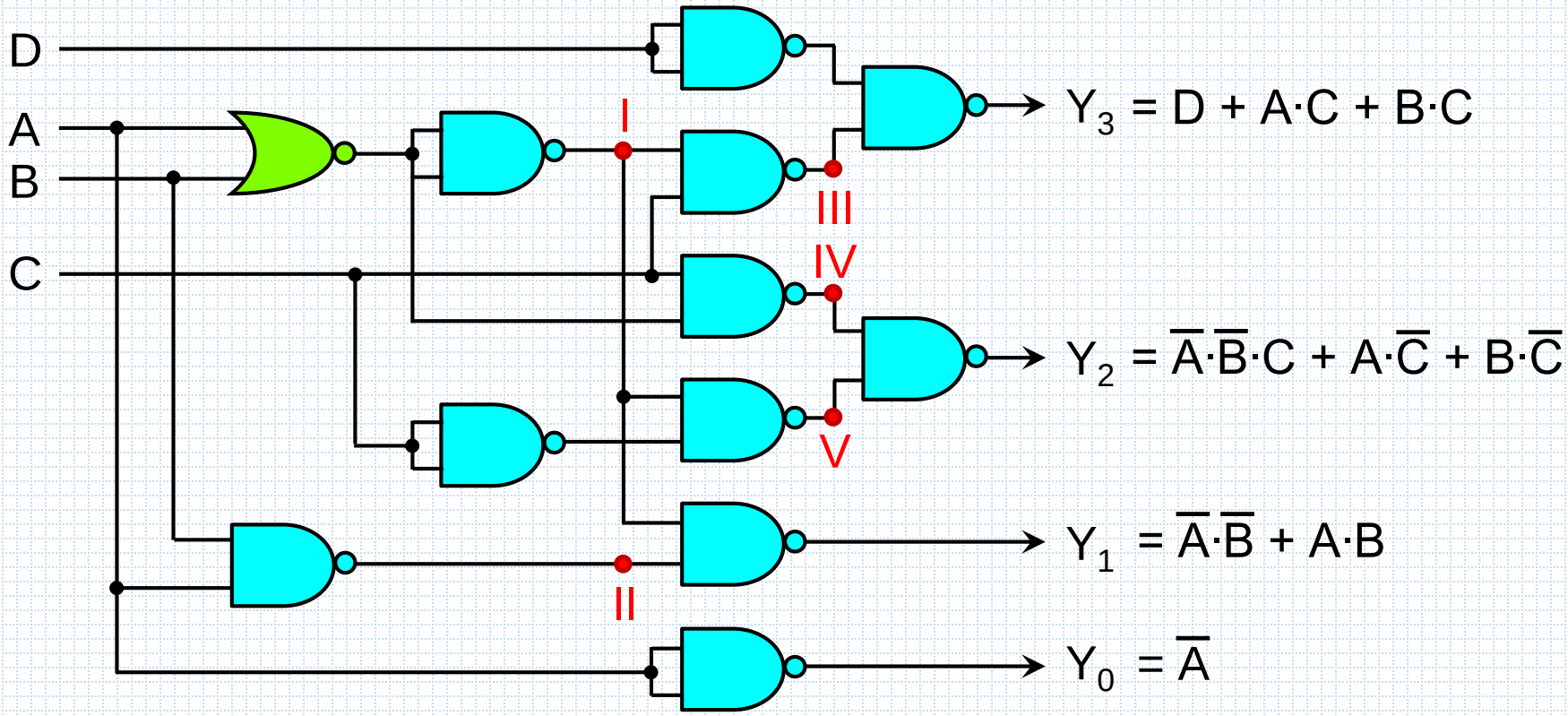
- **Các nhiệm vụ chính của bài toán phân tích tĩnh:** tìm mối quan hệ logic giữa tín hiệu ra và tín hiệu vào và xác định chức năng của mạch

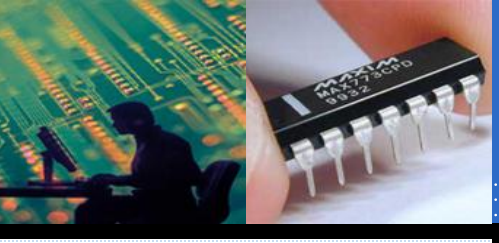
- **Các bước phân tích tĩnh**
  - **Bước 1:** Phân tích sơ đồ mạch logic
  - **Bước 2:** Tìm hệ hàm logic
  - **Bước 3:** Xây dựng bảng giá trị hàm
  - **Bước 4:** Xác định chức năng của mạch

3.1. Phân tích tĩnh mạch tổ hợp

- **Ví dụ:** Phân tích sơ đồ mạch logic cho trên hình sau

$I = \overline{A + B}$ $II = \overline{A \cdot B}$ $III = \overline{\overline{I} \cdot C}$ $IV = \overline{I \cdot C}$ $V = \overline{\overline{I} \cdot \overline{C}}$

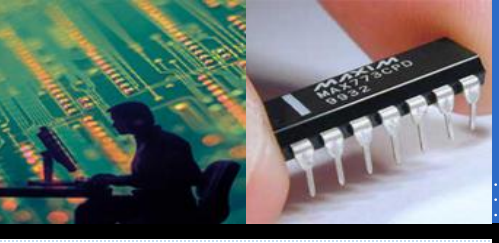




3.1. Phân tích tĩnh mạch tổ hợp

Bảng giá trị hàm

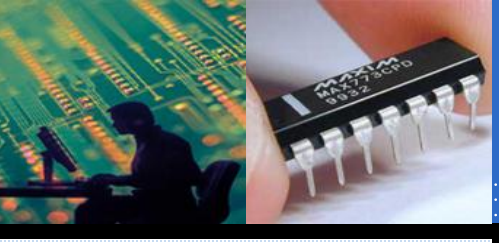
Đầu vào				Đầu ra				Đầu vào				Đầu ra			
D	C	B	A	Y_3	Y_2	Y_1	Y_0	D	C	B	A	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	1	1	1	0	0	0	1	0	1	1
0	0	0	1	0	1	0	0	1	0	0	1	1	1	0	0
0	0	1	0	0	1	0	1	1	0	1	0	1	1	0	1
0	0	1	1	0	1	1	0	1	0	1	1	1	1	1	0
0	1	0	0	0	1	1	1	1	1	0	0	1	1	1	1
0	1	0	1	1	0	0	0	1	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1	1	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0	1	1	1	1	1	0	1	0



3. PHÂN TÍCH MẠCH TỔ HỢP

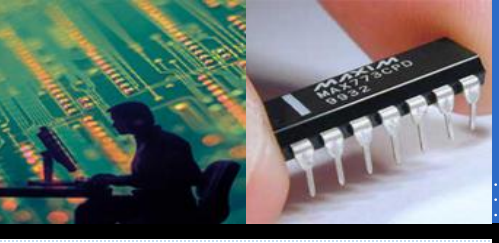
3.1. Phân tích tĩnh mạch tổ hợp

3.2. Phân tích động mạch tổ hợp



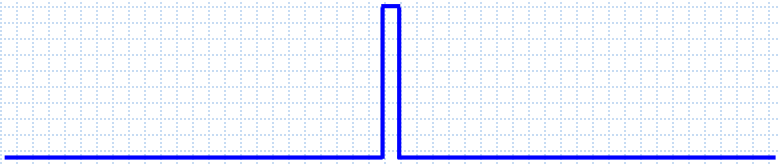
3.2. Phân tích động mạch tổ hợp

- **Bài toán phân tích động:** Phân tích hoạt động của mạch logic tổ hợp xem với các tổ hợp tín hiệu vào cho trước tín hiệu ra thu được có đúng như thiết kế hay không
- **Khái niệm về chạy đua trong mạch tổ hợp**
  - Chạy đua là hiện tượng xuất hiện xung nhiễu (Glitch) ở đầu ra của mạch tổ hợp hoặc trường hợp giá trị của các đầu ra thay đổi và ổn định không đồng thời khi đầu vào thay đổi với mạch có nhiều đầu ra
  - Glitch trong mạch có chạy đua có dạng các xung hình chữ nhật rất hẹp, xuất hiện trong khoảng thời gian chưa ổn định của tín hiệu ra
  - Mạch có khả năng xuất hiện Glitch ở đầu ra được gọi là mạch có chạy đua hay mạch có Hazard (nguy cơ)
  - Nguyên nhân cơ bản của chạy đua do độ trễ của tín hiệu đầu ra một cổng logic so với tín hiệu vào của chính cổng này

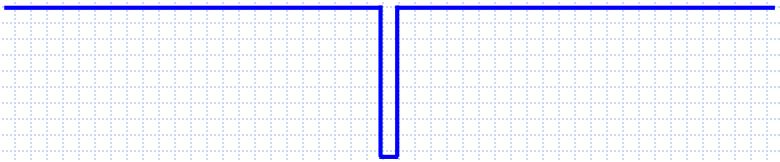


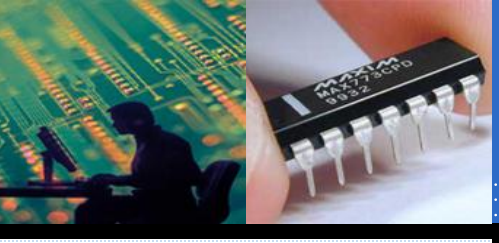
3.2.1. Hazard tĩnh

- **Hazard tĩnh là hiện tượng đầu ra thay đổi trạng thái một lần khác với giá trị lẽ ra phải có ở đầu ra trước khi giá trị này ổn định**
  - **Hazard tĩnh mức 0**: đầu ra lẽ ra phải giữ ổn định ở mức 0 nhưng xuất hiện Glitch lên 1 trong khoảng thời gian ngắn



  - **Hazard tĩnh mức 1**: đầu ra lẽ ra phải giữ ổn định ở mức 1 nhưng xuất hiện Glitch xuống 0 trong khoảng thời gian ngắn

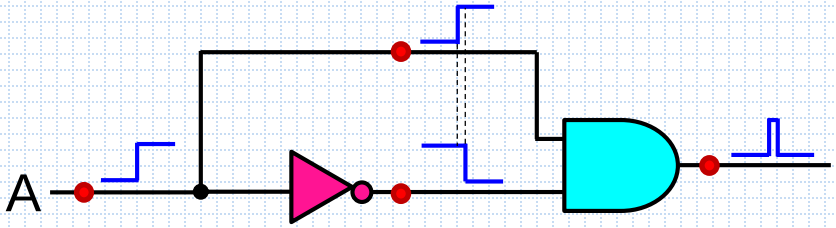




3.2.1. Hazard tĩnh

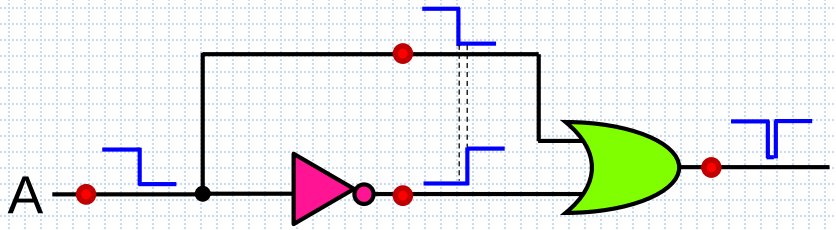
❑ Mạch căn bản có Hazard tĩnh

➢ Mạch căn bản có Hazard tĩnh 0

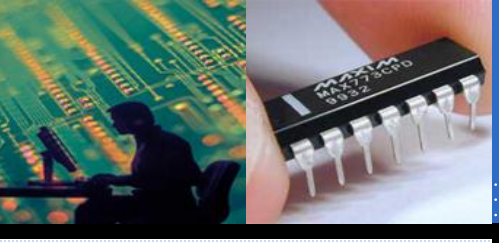


- Có hai đường truyền cho 1 biến vào
- Có một đường truyền đảo
- Hội tụ lại ở cổng AND

➢ Mạch căn bản có Hazard tĩnh 1



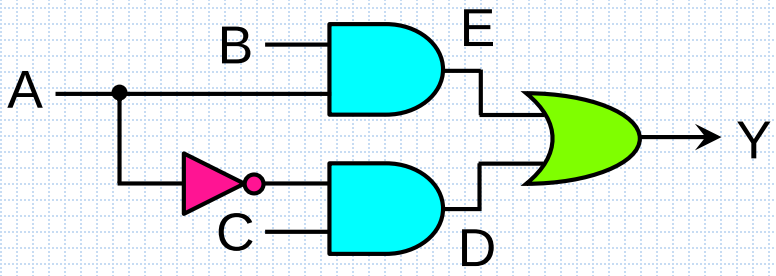
- Có hai đường truyền cho 1 biến vào
- Có một đường truyền đảo
- Hội tụ lại ở cổng OR

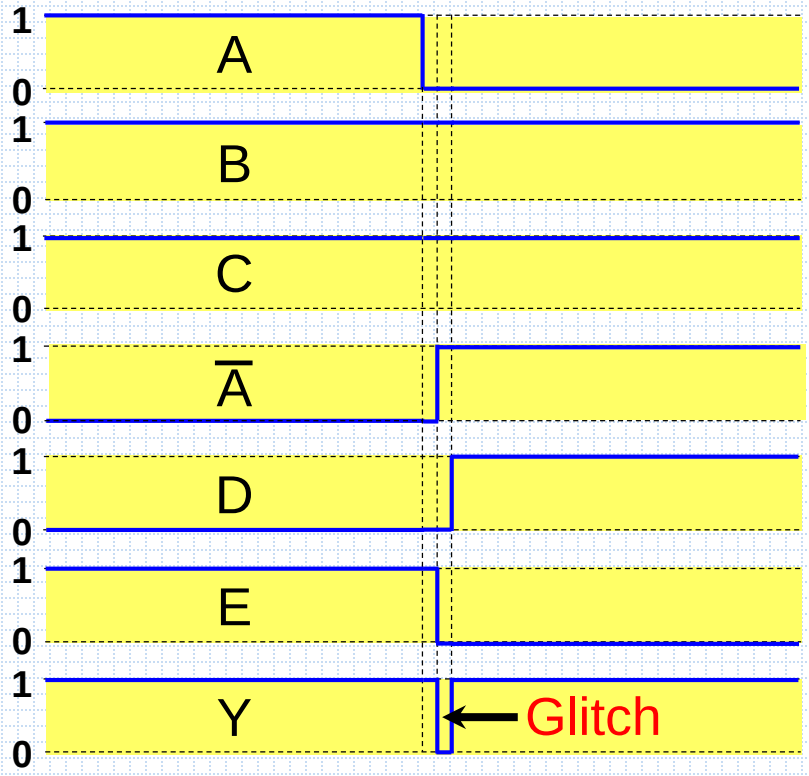


3.2.1. Hazard tĩnh

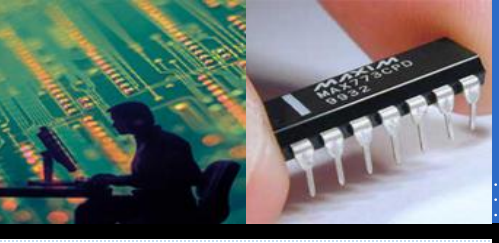
❑ Ví dụ về Hazard tĩnh

$$Y = A \cdot B + \overline{A} \cdot C$$



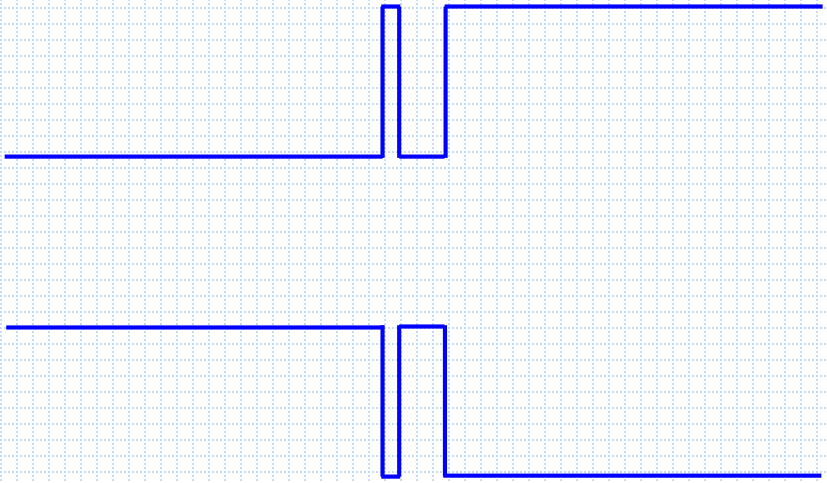


Khi hai tín hiệu vào ở một cổng bất kỳ đồng thời thay đổi trạng thái ngược nhau (từ 1 về 0 và từ 0 lên 1) thì có thể phát sinh Glitch ở đầu ra của nó



3.2.2. Hazard động

- **Hazard động** là hiện tượng đầu ra thay đổi trạng thái một vài lần khác với giá trị lẽ ra phải có ở đầu ra trước khi giá trị này ổn định

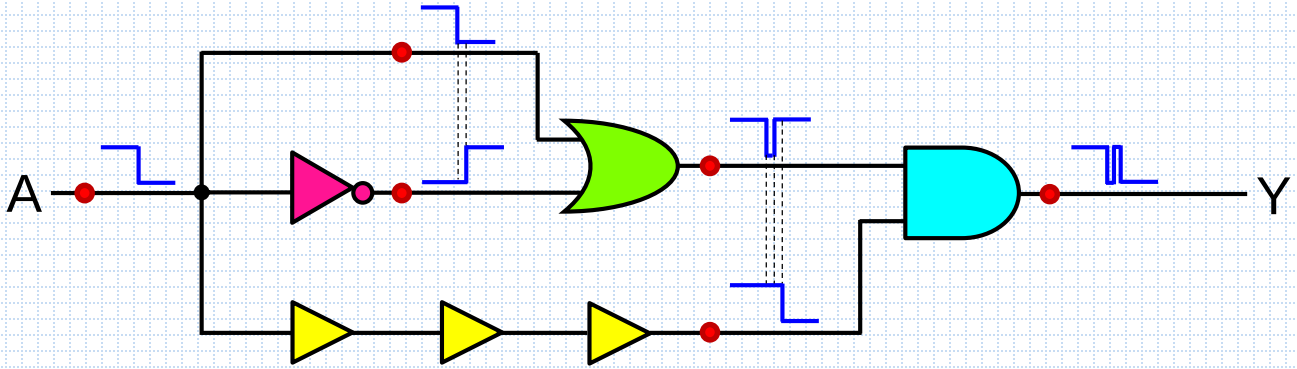


- Mạch căn bản có Hazard động là mạch luôn có 3 đường truyền song song, trong đó có 1 đường Hazard tĩnh
- Mạch không có Hazard tĩnh cũng sẽ không có Hazard động

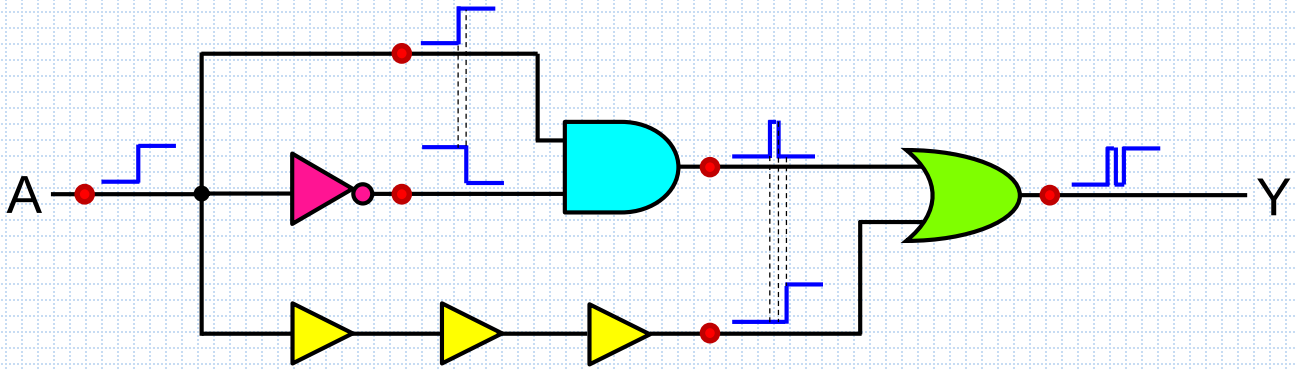
3.2.2. Hazard động

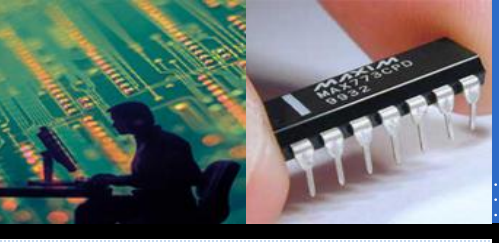
❑ Mạch căn bản có Hazard động

- ➢ Mạch căn bản có Hazard động với Hazard tĩnh 1 có sẵn



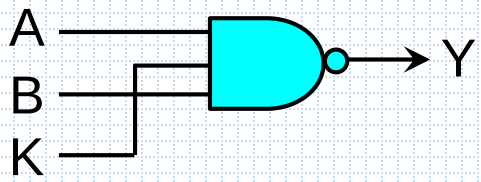
- ➢ Mạch căn bản có Hazard động với Hazard tĩnh 0 có sẵn

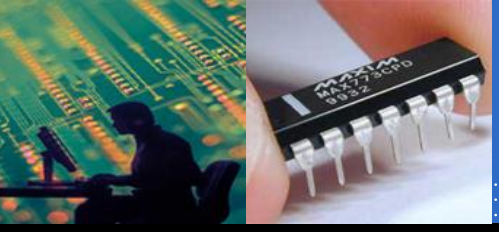




3.2.3. Các biện pháp khắc phục Hazard

- **Lựa chọn linh kiện có thời gian trễ nhỏ**
- **Dùng tụ lọc** có trị số từ vài pF đến vài trăm pF để giảm biên độ xung nhiễu đến mức không ảnh hưởng, tuy nhiên làm xấu sườn xung T/H
- **Dùng xung điều khiển làm xung khóa hay mở cổng**: xung khóa đưa vào trong thời gian quá độ, xung mở đưa vào sau để mở thông mạch

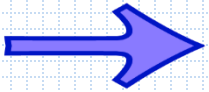




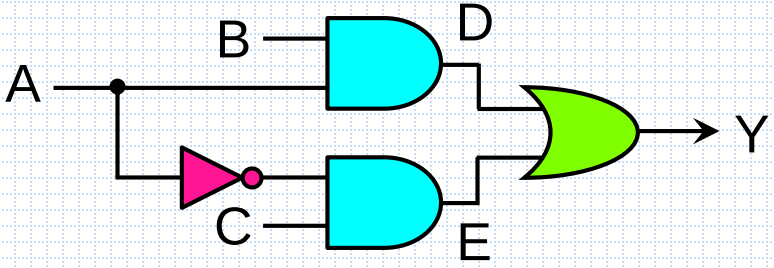
3.2.3. Các biện pháp khắc phục Hazard

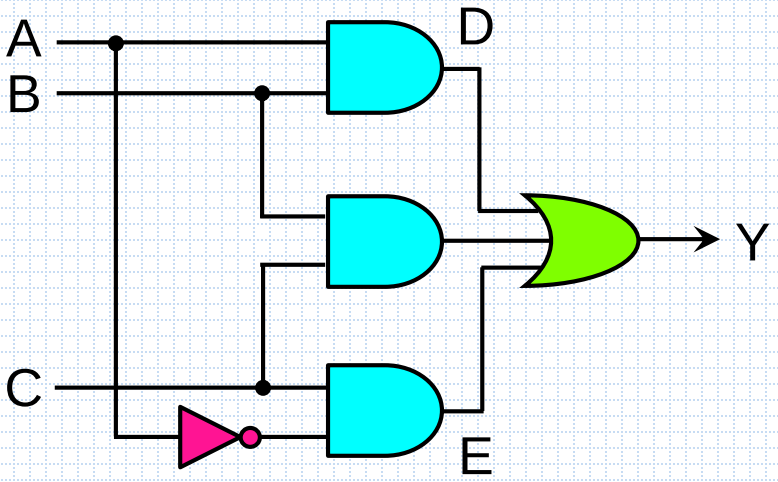
❑ Thay đổi thiết kế mạch che mặt nạ Hazard

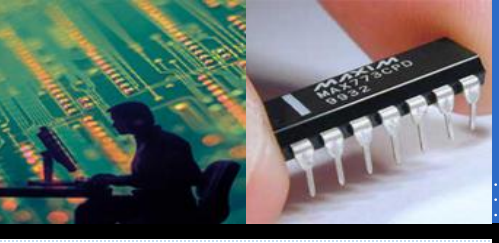
Y	BC 00	01	11	10
A = 0	0	1	1	0
A = 1	0	0	1	1



Y	BC 00	01	11	10
A = 0	0	1	1	0
A = 1	0	0	1	1





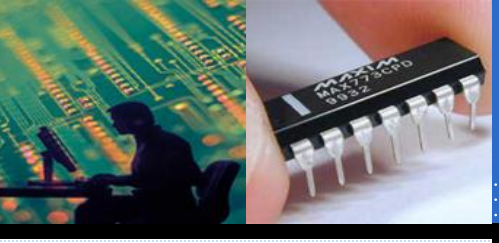


4. CÁC MẠCH TỔ HỢP THÔNG DỤNG

4.1. Bộ mã hóa/giải mã

4.2. Bộ dồn kênh/phân kênh

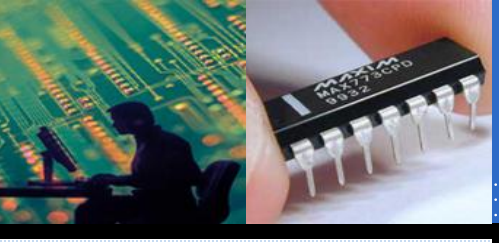
4.3. Mạch phép toán số học



4.1. Bộ mã hóa/giải mã

- **Bộ giải mã:** mạch logic dùng để chuyển đổi một bộ mã này sang bộ mã khác. Các bộ giải mã thông dụng là các bộ giải mã nhị phân, bộ giải mã BCD sang nhị phân và ngược lại, bộ giải mã BCD sang 7 đoạn

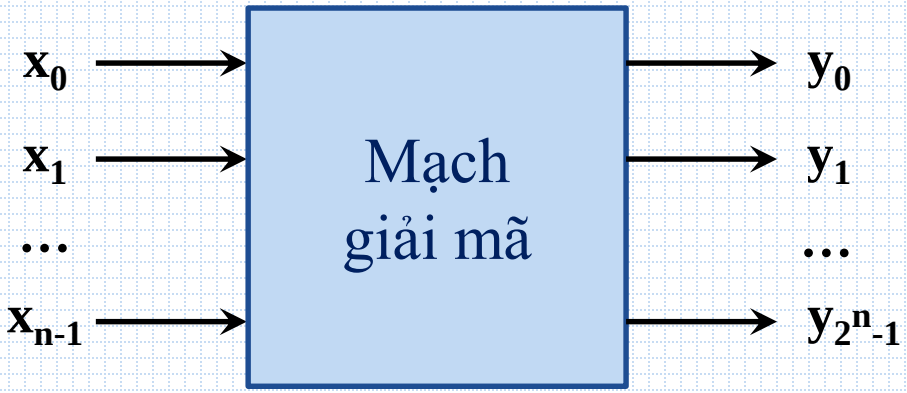
- **Bộ giải mã nhị phân** dùng để chuyển đổi mã nhị phân thành mã $1:2^n$. Mạch giải mã nhị phân được xây dựng trên nguyên lý: ứng với mỗi tổ hợp của tín hiệu vào sẽ cho tương ứng duy nhất một đầu ra có mức tích cực (bằng 1)



4.1. Bộ mã hóa/giải mã

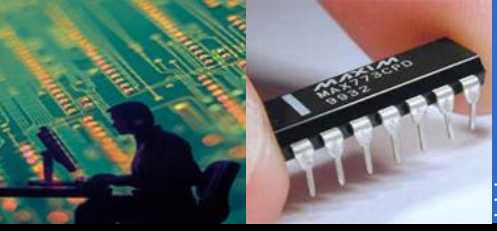
❑ Bộ giải mã nhị phân

➢ Sơ đồ mạch



➢ Hệ phương trình logic

$$\begin{cases} y_0 = \overline{x}_0\,\overline{x}_1 \dots \overline{x}_{n-2}\,\overline{x}_{n-1} \\ y_1 = x_0\,\overline{x}_1 \dots \overline{x}_{n-2}\,\overline{x}_{n-1} \\ \dots \\ y_{2^n-1} = x_0\,x_1 \dots x_{n-2}\,x_{n-1} \end{cases}$$

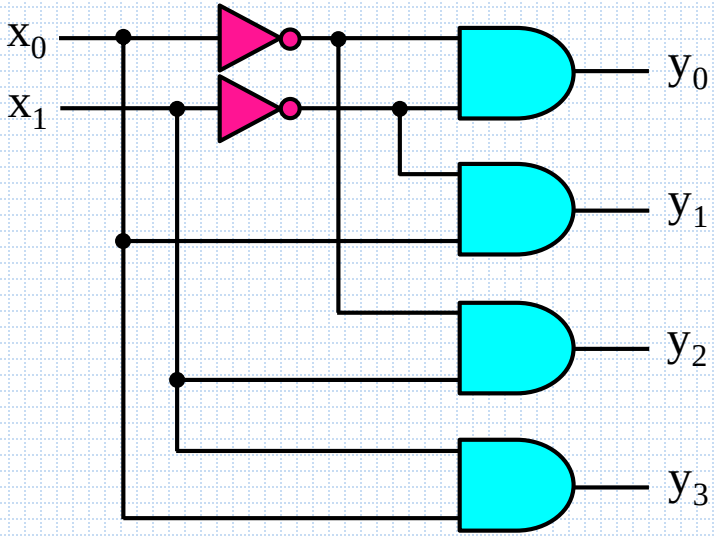


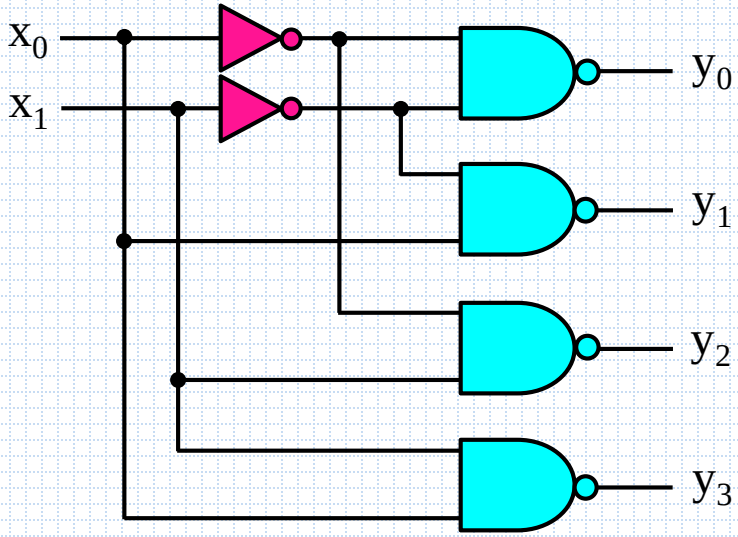
4.1. Bộ mã hóa/giải mã

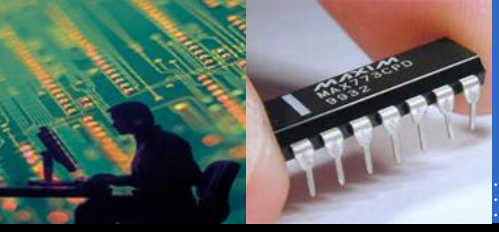
- **VD1: Bộ giải mã nhị phân có 2 đầu vào 4 đầu ra**
  - **Hệ phương trình logic** (mức tích cực cao và mức tích cực thấp)

$$\begin{cases} y_0 = \overline{x_0}\cdot\overline{x_1} \\ y_1 = x_0\cdot\overline{x_1} \\ y_2 = \overline{x_0}\cdot x_1 \\ y_3 = x_0\cdot x_1 \end{cases} \qquad \begin{cases} \overline{y_0} = \overline{\overline{x_0}\cdot\overline{x_1}} \\ \overline{y_1} = \overline{x_0\cdot\overline{x_1}} \\ \overline{y_2} = \overline{\overline{x_0}\cdot x_1} \\ \overline{y_3} = \overline{x_0\cdot x_1} \end{cases}$$

  - **Sơ đồ mạch logic** (mức tích cực cao và mức tích cực thấp)





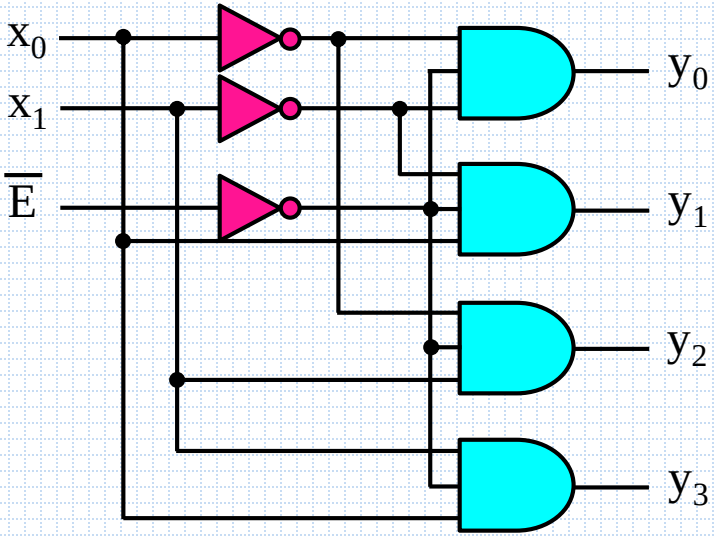


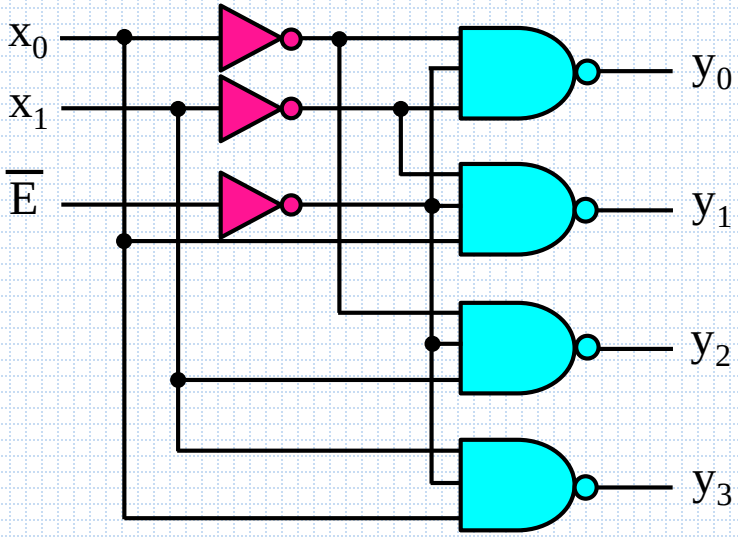
4.1. Bộ mã hóa/giải mã

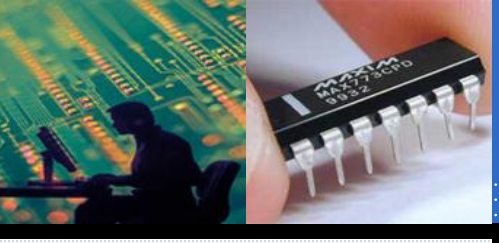
- **VD1: Bộ giải mã nhị phân có 2 đầu vào 4 đầu ra**
  - **Hệ phương trình logic** (Thêm đầu vào cho phép E – Enable)

$$\begin{cases} y_0 = \overline{x_0}\cdot\overline{x_1}\cdot\overline{E} \\ y_1 = x_0\cdot\overline{x_1}\cdot\overline{E} \\ y_2 = \overline{x_0}\cdot x_1\cdot\overline{E} \\ y_3 = x_0\cdot x_1\cdot\overline{E} \end{cases} \qquad \begin{cases} \overline{y_0} = \overline{\overline{x_0}\cdot\overline{x_1}\cdot\overline{E}} \\ \overline{y_1} = \overline{x_0\cdot\overline{x_1}\cdot\overline{E}} \\ \overline{y_2} = \overline{\overline{x_0}\cdot x_1\cdot\overline{E}} \\ \overline{y_3} = \overline{x_0\cdot x_1\cdot\overline{E}} \end{cases}$$

  - **Sơ đồ mạch logic** (mức tích cực cao và mức tích cực thấp)





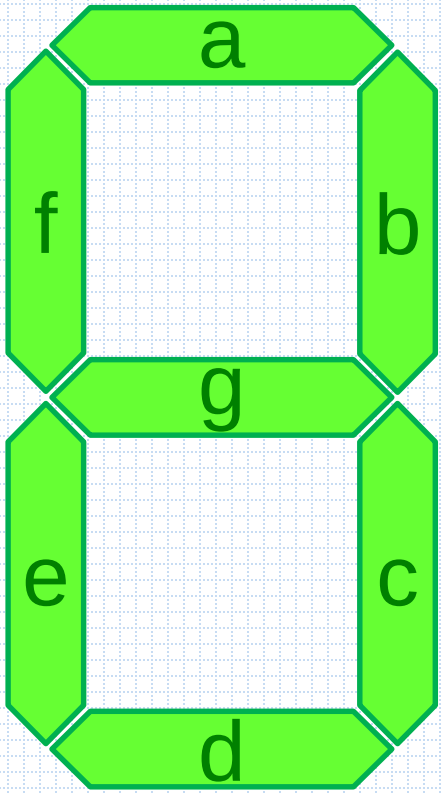


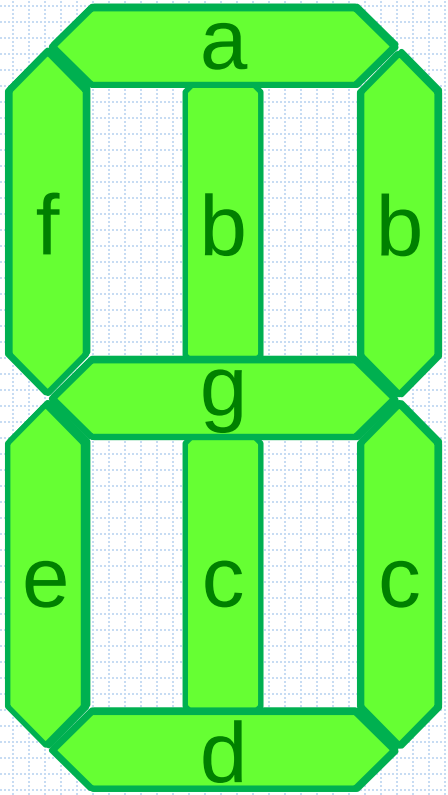
4.1. Bộ mã hóa/giải mã

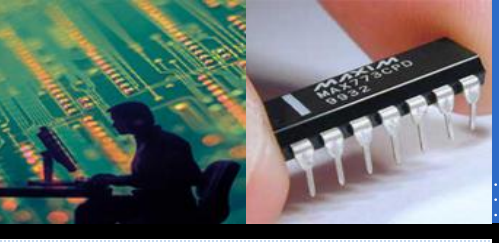
❑ Bộ giải mã BCD – 7 đoạn

➢ Mã 7 đoạn

0	1	2	3	4	5	6	7	8	9



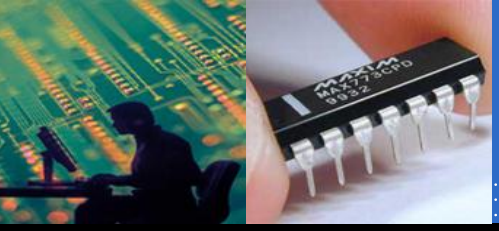




4.1. Bộ mã hóa/giải mã

➢ Bảng giá trị hàm

A_3	A_2	A_1	A_0	X	A	B	C	D	E	F	G
0	0	0	0	**0**	1	1	1	1	1	1	0
0	0	0	1	**1**	0	1	1	0	0	0	0
0	0	1	0	**2**	1	1	0	1	1	0	1
0	0	1	1	**3**	1	1	1	1	0	0	1
0	1	0	0	**4**	0	1	1	0	0	1	1
0	1	0	1	**5**	1	0	1	1	0	1	1
0	1	1	0	**6**	1	0	1	1	1	1	1
0	1	1	1	**7**	1	1	1	0	0	0	0
1	0	0	0	**8**	1	1	1	1	1	1	1
1	0	0	1	**9**	1	1	1	1	0	1	1



4.1. Bộ mã hóa/giải mã

➢ Bảng Karnaugh và biểu thức logic

A

A1, A0

A3, A2 \ A1 A0	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	x	x	x	x
10	1	1	x	x

$\overline{A2}\,\overline{A0} + A1 + A2\,A0 + A3$

B

A1, A0

A3, A2 \ A1 A0	00	01	11	10
00	1	1	1	1
01	1	0	1	0
11	x	x	x	x
10	1	1	x	x

$\overline{A2} + \overline{A1}\,\overline{A0} + A1\,A0$

C

A1, A0

A3, A2 \ A1 A0	00	01	11	10
00	1	1	1	0
01	1	1	1	1
11	x	x	x	x
10	1	1	x	x

$\overline{A1} + A0 + A2$

D

A1, A0

A3, A2 \ A1 A0	00	01	11	10
00	1	0	1	1
01	0	1	0	1
11	x	x	x	x
10	1	1	x	x

$\overline{A2}\,\overline{A0} + \overline{A2}\,A1 + A1\,\overline{A0} + A2\,\overline{A1}\,A0 + A3$

G

A1, A0

A3, A2 \ A1 A0	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	x	x	x	x
10	1	1	x	x

$\overline{A2}\,A1 + A1\,\overline{A0} + A2\,\overline{A1} + A3$

F

A1, A0

A3, A2 \ A1 A0	00	01	11	10
00	1	0	0	0
01	1	1	0	1
11	x	x	x	x
10	1	1	x	x

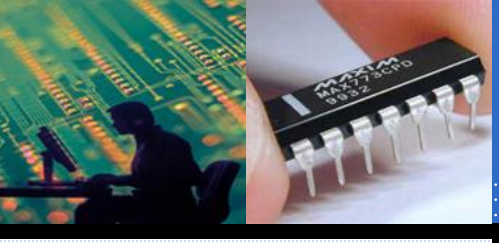
$\overline{A1}\,\overline{A0} + A2\,\overline{A1} + A2\,\overline{A0} + A3$

E

A1, A0

A3, A2 \ A1 A0	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	x	x	x	x
10	1	0	x	x

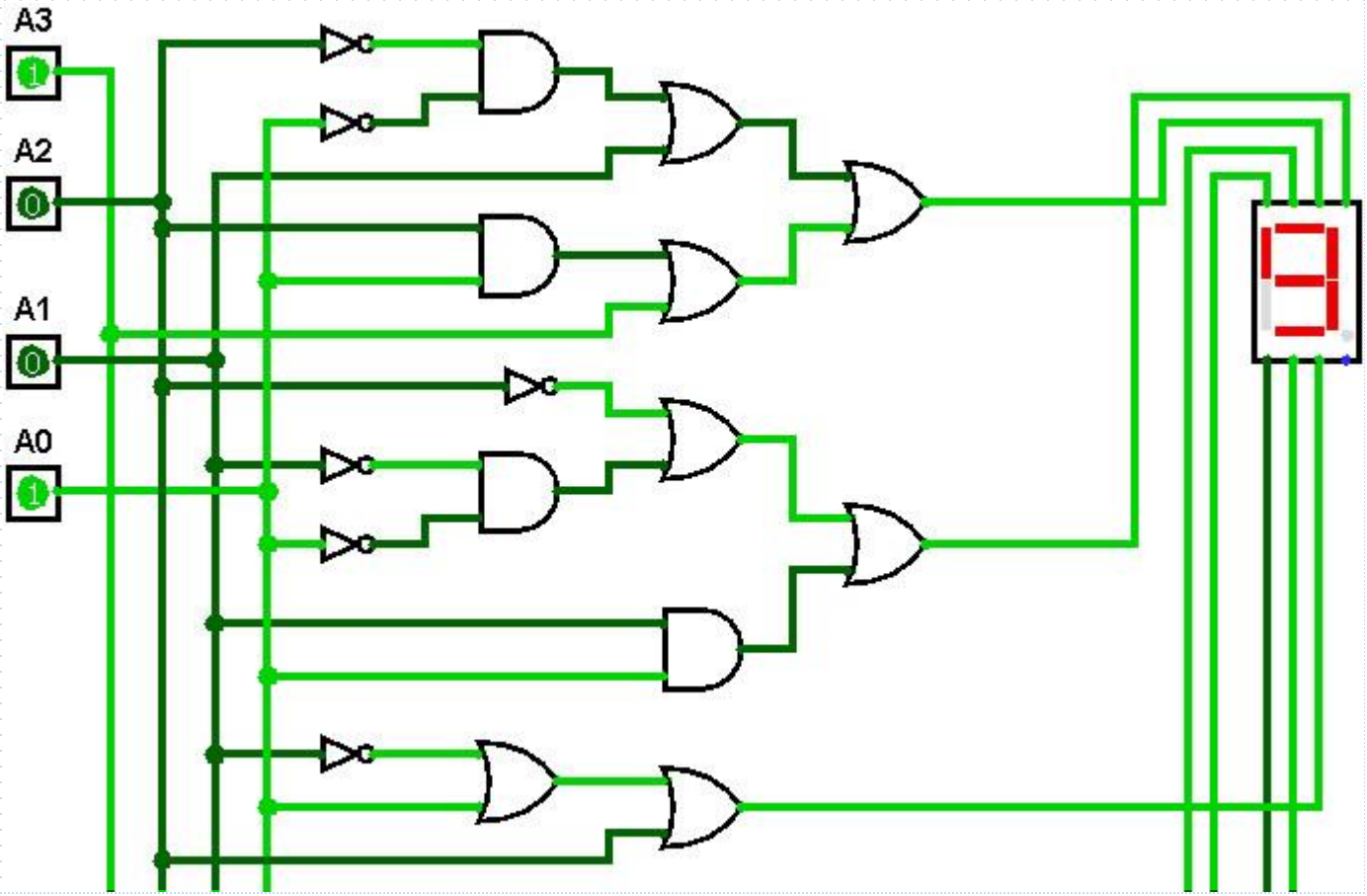
$\overline{A2}\,\overline{A0} + A1\,\overline{A0}$

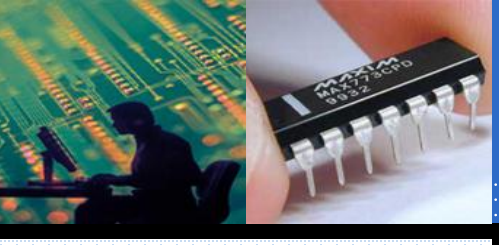


4.1. Bộ mã hóa/giải mã

- **Mạch logic**

0 1 2 3 4 5 6 7 8 9



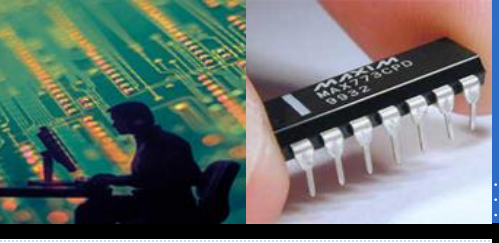


4. CÁC MẠCH TỔ HỢP THÔNG DỤNG

4.1. Bộ mã hóa/giải mã

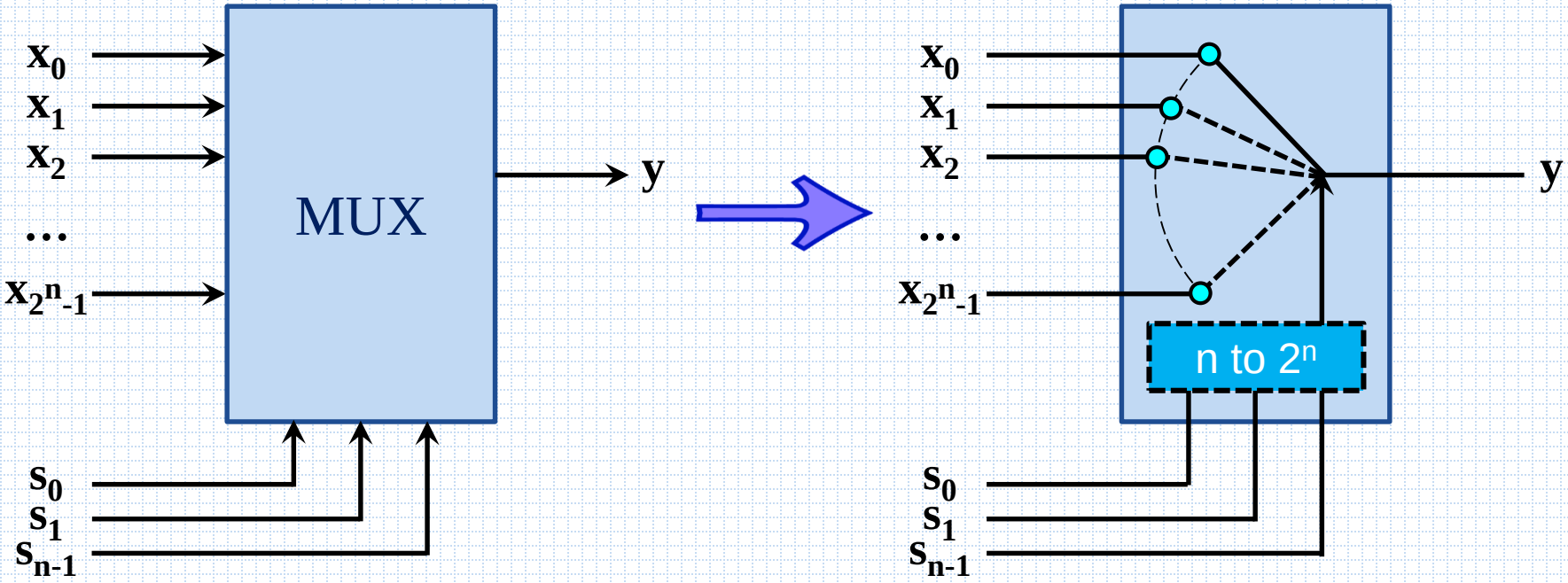
4.2. Bộ dồn kênh/phân kênh

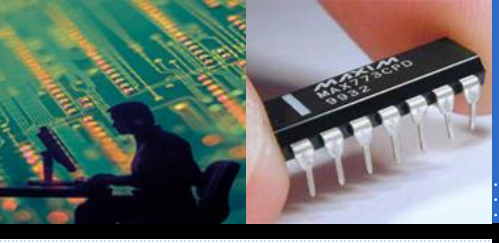
4.3. Mạch phép toán số học



4.2. Bộ dồn kênh/phân kênh

- **Bộ ghép kênh (Bộ dồn kênh, Multiplexer – MUX)**: mạch logic có chức năng chọn lần lượt 1 trong N đầu vào tín hiệu để đưa đến một đầu ra duy nhất. MUX gồm 2^n đầu vào, một đầu ra, ngoài ra còn có n tín hiệu điều khiển (hay còn gọi là địa chỉ). Có thể xem MUX như một chuyển mạch điện tử





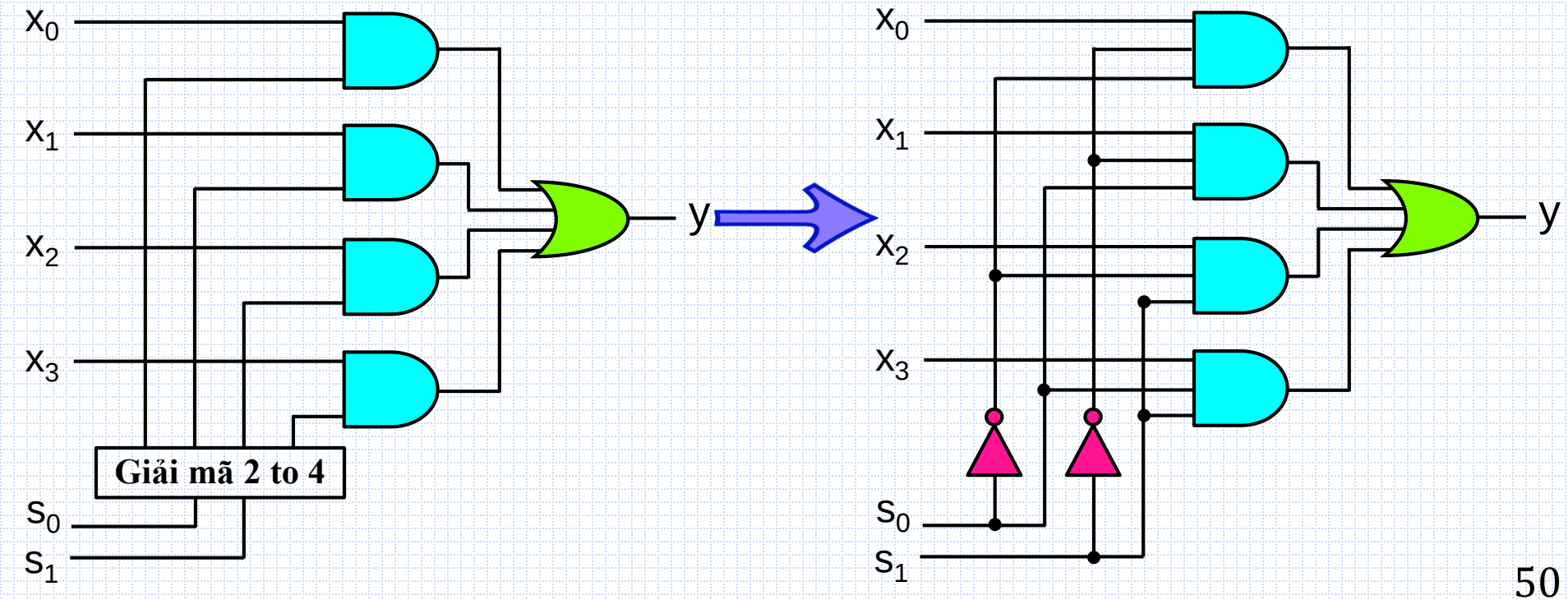
4.2. Bộ dồn kênh/phân kênh

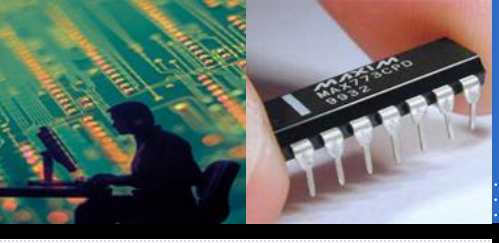
- **Phương trình hàm logic tổng quát của MUX**

$$y = x_0(\overline{s_0}\cdot\overline{s_1}\ldots\overline{s_{n-1}}) + x_1(s_0\cdot\overline{s_1}\ldots\overline{s_{n-1}}) + \ldots + x_{2^n-1}(s_0\cdot s_1\ldots s_{n-1})$$

- **VD: MUX có 4 đầu vào dữ liệu và 2 đầu vào địa chỉ**

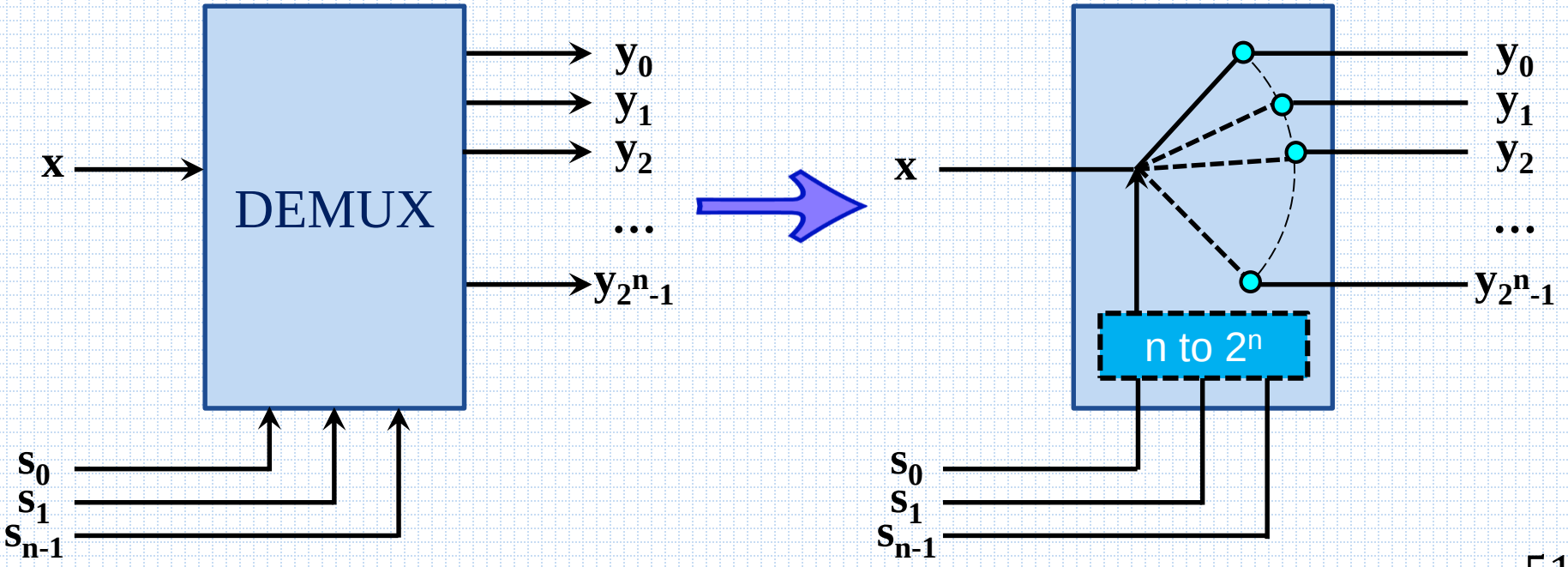
$$y = x_0(\overline{s_0}\cdot\overline{s_1}) + x_1(s_0\cdot\overline{s_1}) + x_2(\overline{s_0}\cdot s_1) + x_3(s_0\cdot s_1)$$

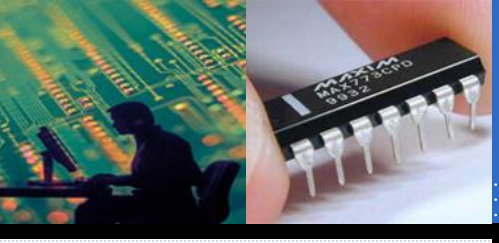




4.2. Bộ dồn kênh/phân kênh

- **Bộ tách kênh (Bộ phân kênh, Demultiplexer – DEMUX)**: mạch logic có chức năng ngược với bộ ghép kênh. Tùy thuộc vào tổ hợp tín hiệu điều khiển (tín hiệu địa chỉ) lần lượt các tín hiệu từ đầu vào x được đưa tới các đầu ra tương ứng. Có thể xem DEMUX như một chuyển mạch điện tử theo chiều ngược





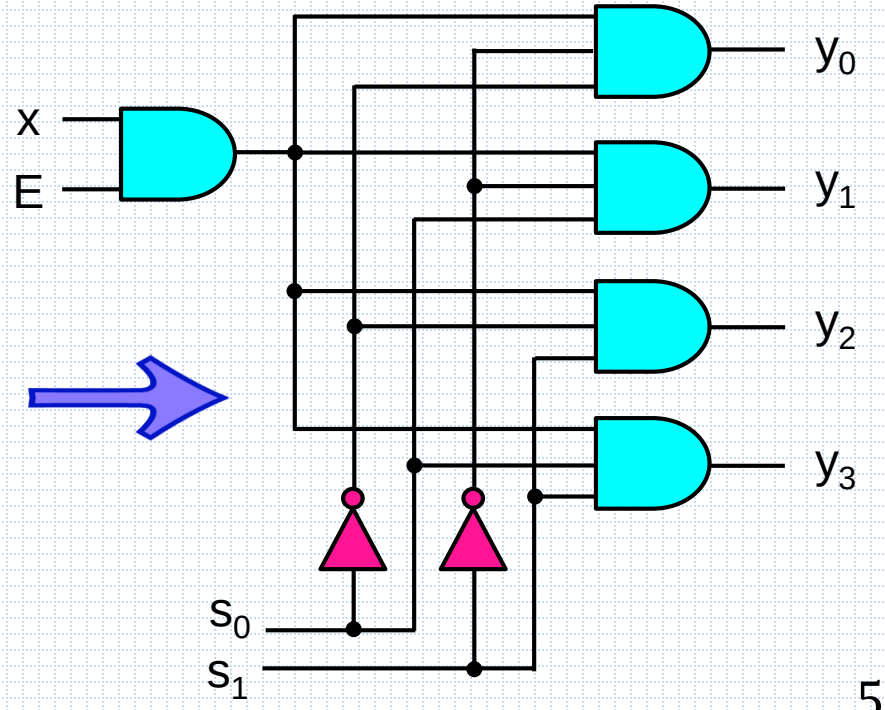
4.2. Bộ dồn kênh/phân kênh

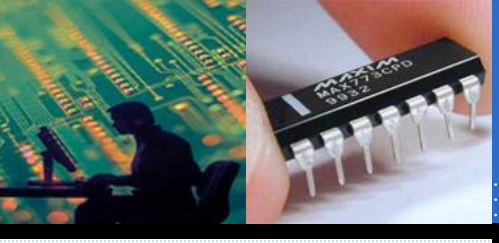
- **Phương trình hàm logic tổng quát của DEMUX**

$$\begin{cases} y_0 = x(\overline{s_0}\cdot\overline{s_1}\ldots\overline{s_{n-1}}) \\ y_1 = x(s_0\cdot\overline{s_1}\ldots\overline{s_{n-1}}) \\ \ldots \\ y_{2^n-1} = x(s_0\cdot s_1\ldots s_{n-1}) \end{cases}$$

- **VD: DEMUX có 4 đầu ra dữ liệu và 2 đầu vào địa chỉ**

$$\begin{cases} y_0 = x(\overline{s_0}\cdot\overline{s_1}) \\ y_1 = x(s_0\cdot\overline{s_1}) \\ y_2 = x(\overline{s_0}\cdot s_1) \\ y_3 = x(s_0\cdot s_1) \end{cases} \qquad \begin{cases} y_0 = x(\overline{s_0}\cdot\overline{s_1})\cdot E \\ y_1 = x(s_0\cdot\overline{s_1})\cdot E \\ y_2 = x(\overline{s_0}\cdot s_1)\cdot E \\ y_3 = x(s_0\cdot s_1)\cdot E \end{cases}$$



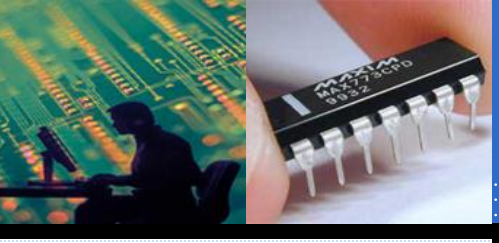


4. CÁC MẠCH TỔ HỢP THÔNG DỤNG

4.1. Bộ mã hóa/giải mã

4.2. Bộ dồn kênh/phân kênh

4.3. Mạch phép toán số học



4.3.1. Bộ cộng nhị phân

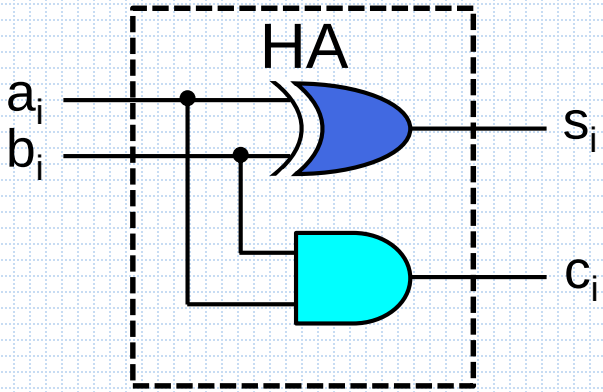
- **Mạch cộng 1 bit không nhớ - Mạch cộng bán phần (HA – Half Adder):** Thực hiện phép toán cộng 2 số nhị phân 1 bit a_i và b_i, không có bit nhớ từ các bước tính toán trước đó. Kết quả cho ra bit tổng s_i và bit nhớ c_i

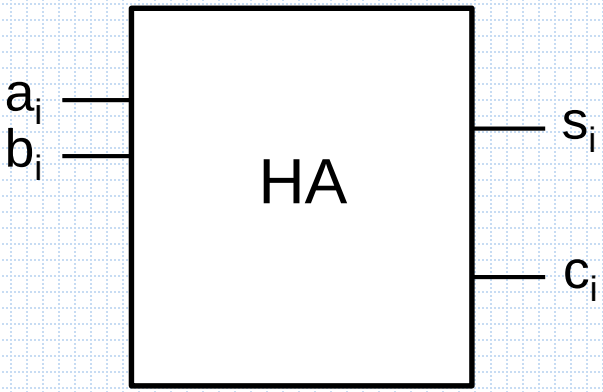
  - **Bảng giá trị hàm và biểu thức logic của mạch cộng bán phần**

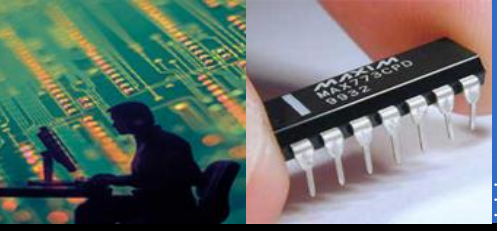
a_i	b_i	s_i	c_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$\begin{cases} s_i = a_i \oplus b_i \\ c_i = a_i \cdot b_i \end{cases}$$

  - **Sơ đồ logic và sơ đồ khối mạch cộng bán phần**







4.3.1. Bộ cộng nhị phân

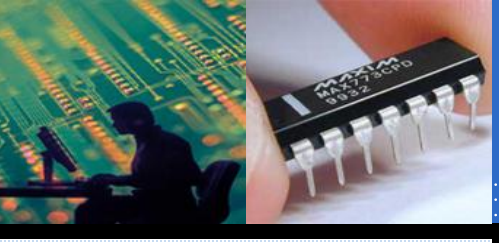
- **Mạch cộng một bit có nhớ - Mạch cộng toàn phần (FA – Full Adder):**
  Thực hiện phép toán cộng hai số nhị phân 1 bit a_i và b_i có tính tới bit nhớ c_{i-1} từ các bước tính toán trước đó. Kết quả cho ra bit tổng s_i và bit nhớ c_i

  - **Bảng giá trị hàm và biểu thức logic của mạch cộng bán phần**

$$\begin{cases} s_i = \overline{a_i}\cdot\overline{b_i}\cdot c_{i-1} + \overline{a_i}\cdot b_i\cdot\overline{c_{i-1}} + a_i\cdot\overline{b_i}\cdot\overline{c_{i-1}} + a_i\cdot b_i\cdot c_{i-1} \\ c_i = \overline{a_i}\cdot b_i\cdot c_{i-1} + a_i\cdot\overline{b_i}\cdot c_{i-1} + a_i\cdot b_i\cdot\overline{c_{i-1}} + a_i\cdot b_i\cdot c_{i-1} \end{cases}$$

$$\begin{cases} s_i = (a_i \oplus b_i) \oplus c_{i-1} = (b_i \oplus c_{i-1}) \oplus a_i \\ c_i = (a_i \oplus b_i)\cdot c_{i-1} + a_i\cdot b_i = (b_i \oplus c_{i-1})\cdot a_i + b_i\cdot c_{i-1} \end{cases}$$

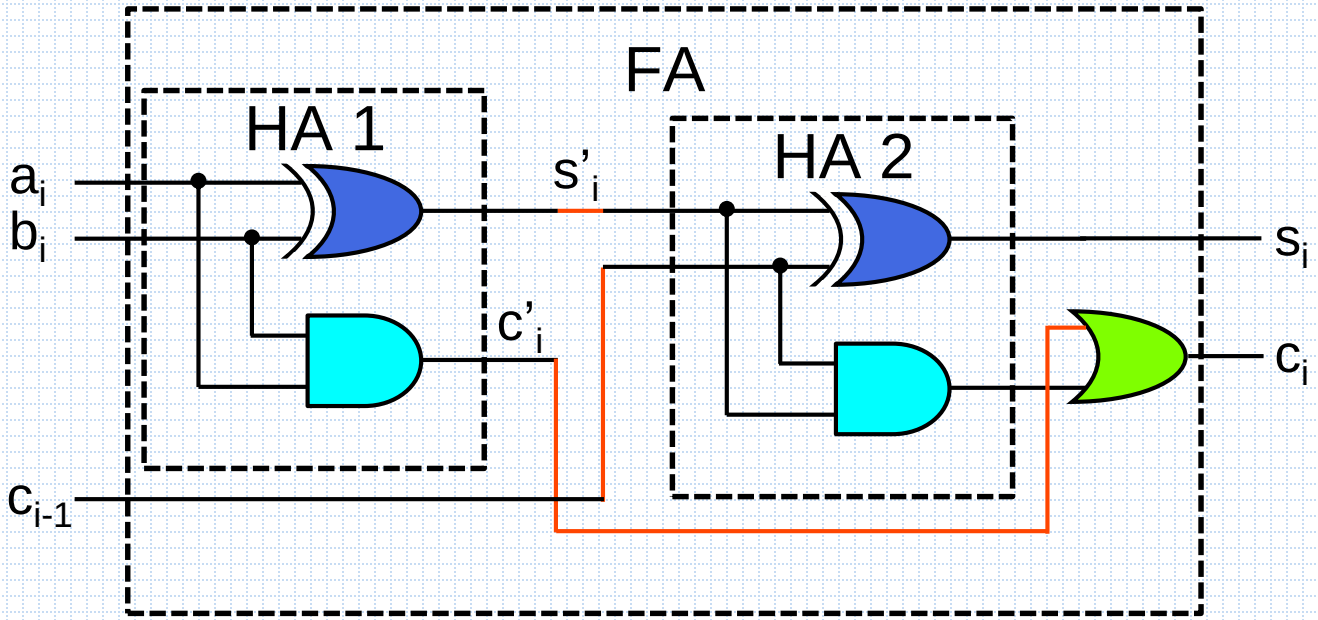
a_i	b_i	c_{i-1}	s_i	c_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

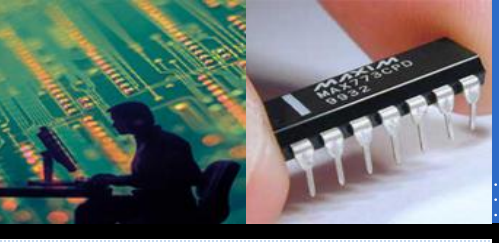


4.3.1. Bộ cộng nhị phân

➢ Sơ đồ logic và sơ đồ khối tương đương mạch cộng toàn phần

$$\begin{cases} s_i = (a_i \oplus b_i) \oplus c_{i-1} = s'_i \oplus c_{i-1} \\ c_i = (a_i \oplus b_i) \cdot c_{i-1} + a_i \cdot b_i = s'_i \cdot c_{i-1} + c'_i \end{cases}$$

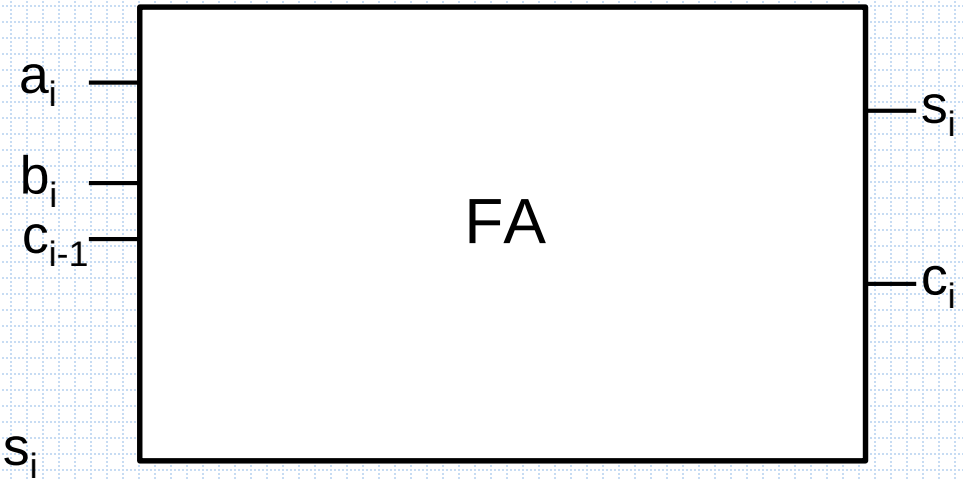


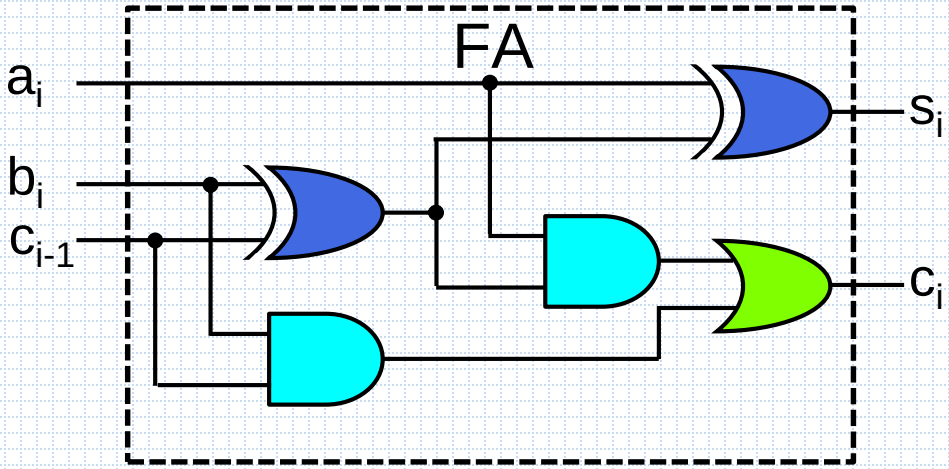


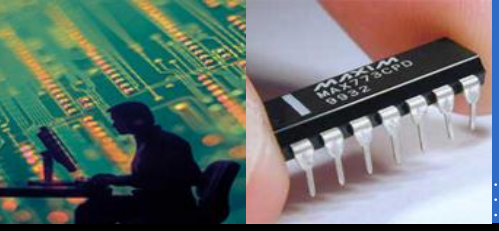
4.3.1. Bộ cộng nhị phân

- **Sơ đồ logic và sơ đồ khối mạch cộng toàn phần (Phương án 2)**

$$\begin{cases} s_i = (b_i \oplus c_{i-1}) \oplus a_i \\ c_i = (b_i \oplus c_{i-1}) \cdot a_i + b_i \cdot c_{i-1} \end{cases}$$

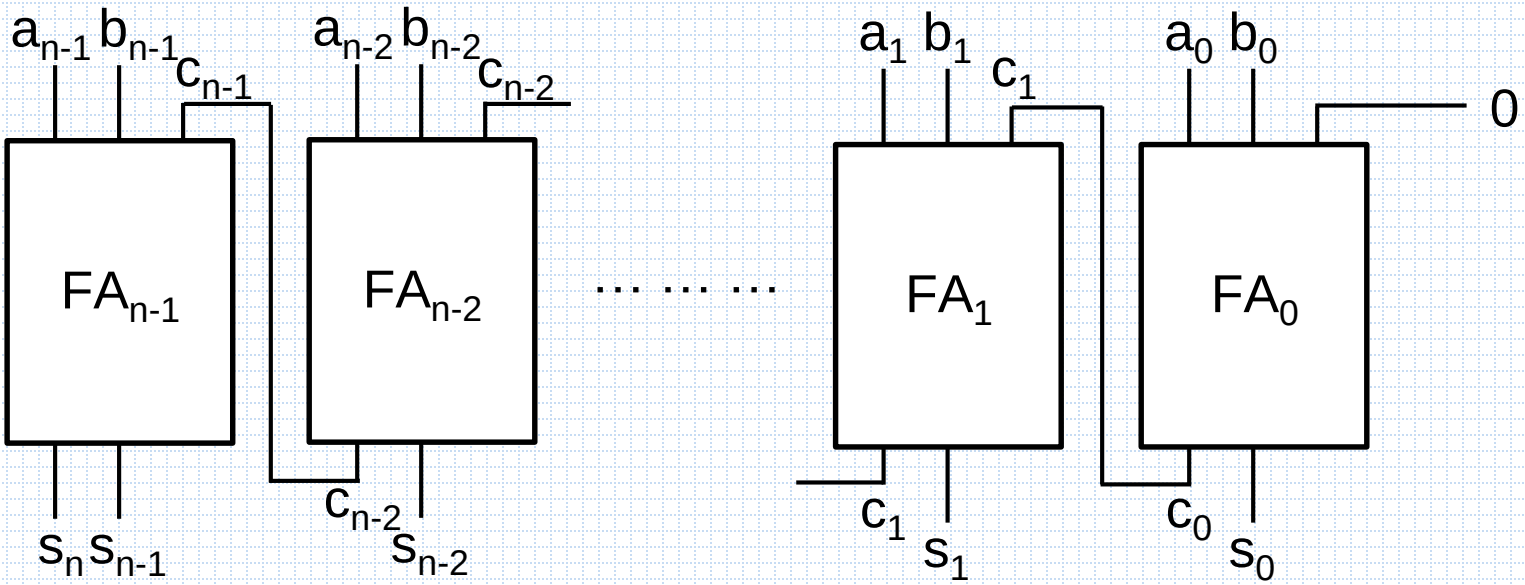


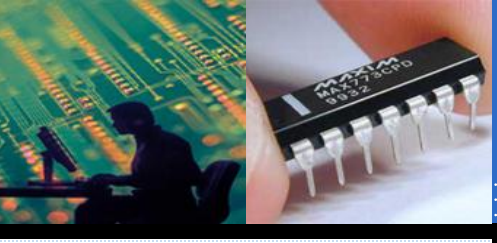




4.3.1. Bộ cộng nhị phân

- **Bộ cộng hai số nhị phân nhiều bit**
  - **Bộ cộng nhiều bit xây dựng từ các mạch cộng FA song song**





4.3.1. Bộ cộng nhị phân

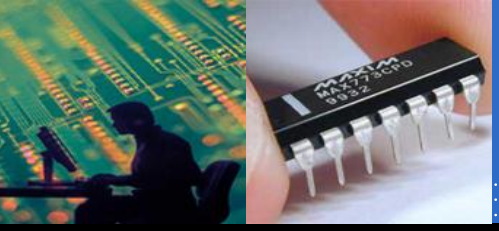
❑ Bộ cộng hai số nhị phân nhiều bit

➢ Bộ cộng nhiều bit xây dựng từ mạch cộng thấy trước nhớ – Carry Look-ahead Adder (CLA)

$$\begin{cases} c_0 = a_0 \cdot b_0 \\ c_1 = a_1 \cdot b_1 + (a_1 \oplus b_1) \cdot c_0 = a_1 \cdot b_1 + (a_1 \oplus b_1) \cdot (a_0 \cdot b_0) \\ c_2 = a_2 \cdot b_2 + (a_2 \oplus b_2) \cdot c_1 = a_2 \cdot b_2 + (a_2 \oplus b_2) \cdot [a_1 \cdot b_1 + (a_1 \oplus b_1) \cdot (a_0 \cdot b_0)] = \\ \quad = a_2 \cdot b_2 + (a_2 \oplus b_2) \cdot (a_1 \cdot b_1) + (a_2 \oplus b_2) \cdot (a_1 \oplus b_1) \cdot (a_0 \cdot b_0) \end{cases}$$

Đặt $g_i = a_i \cdot b_i$ và $p_i = (a_i \oplus b_i)$

$$\Rightarrow \begin{cases} c_0 = a_0 \cdot b_0 = g_0 \\ c_1 = g_1 + p_1 \cdot c_0 = g_1 + p_1 \cdot g_0 \\ c_2 = g_2 + p_2 \cdot c_1 = g_2 + p_2 \cdot g_1 + p_2 \cdot p_1 \cdot g_0 \end{cases}$$

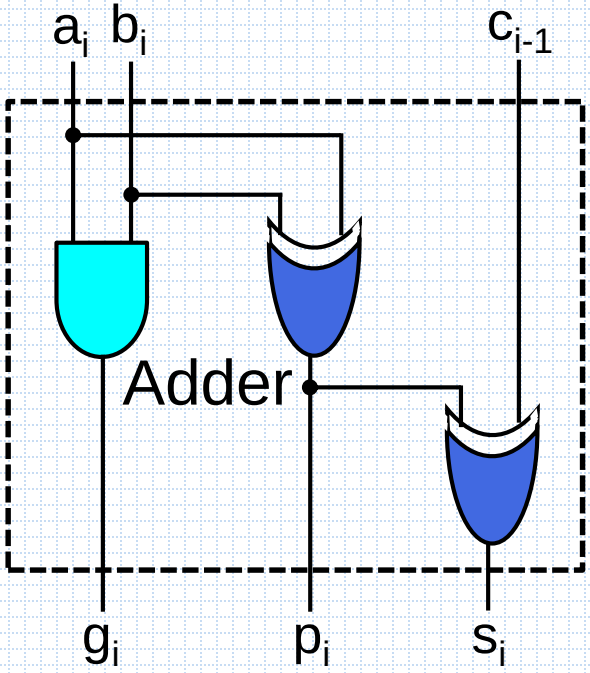


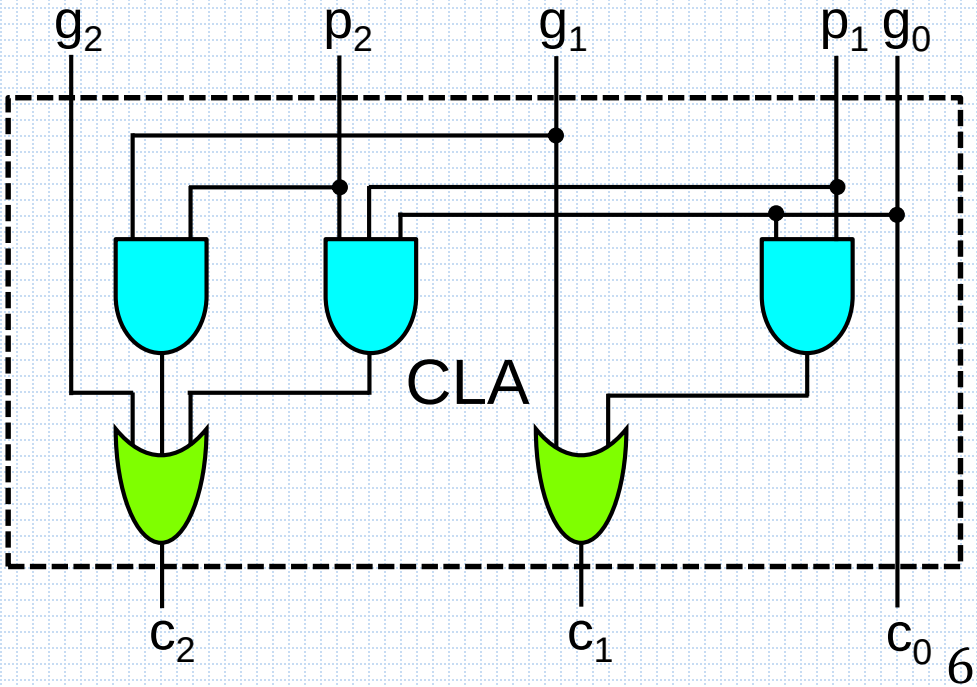
4.3.1. Bộ cộng nhị phân

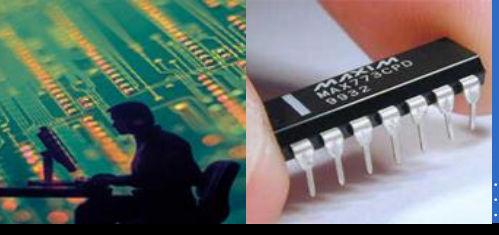
- **Bộ cộng hai số nhị phân nhiều bit**
  - **Bộ cộng nhiều bit xây dựng từ mạch cộng thấy trước nhớ – Carry Look-ahead Adder (CLA)**

$$s_i = (a_i \oplus b_i) \oplus c_{i-1} = p_i \oplus c_{i-1}$$

$$\begin{cases} c_0 = g_0 \\ c_1 = g_1 + p_1 \cdot g_0 \\ c_2 = g_2 + p_2 \cdot g_1 + p_2 \cdot p_1 \cdot g_0 \end{cases}$$

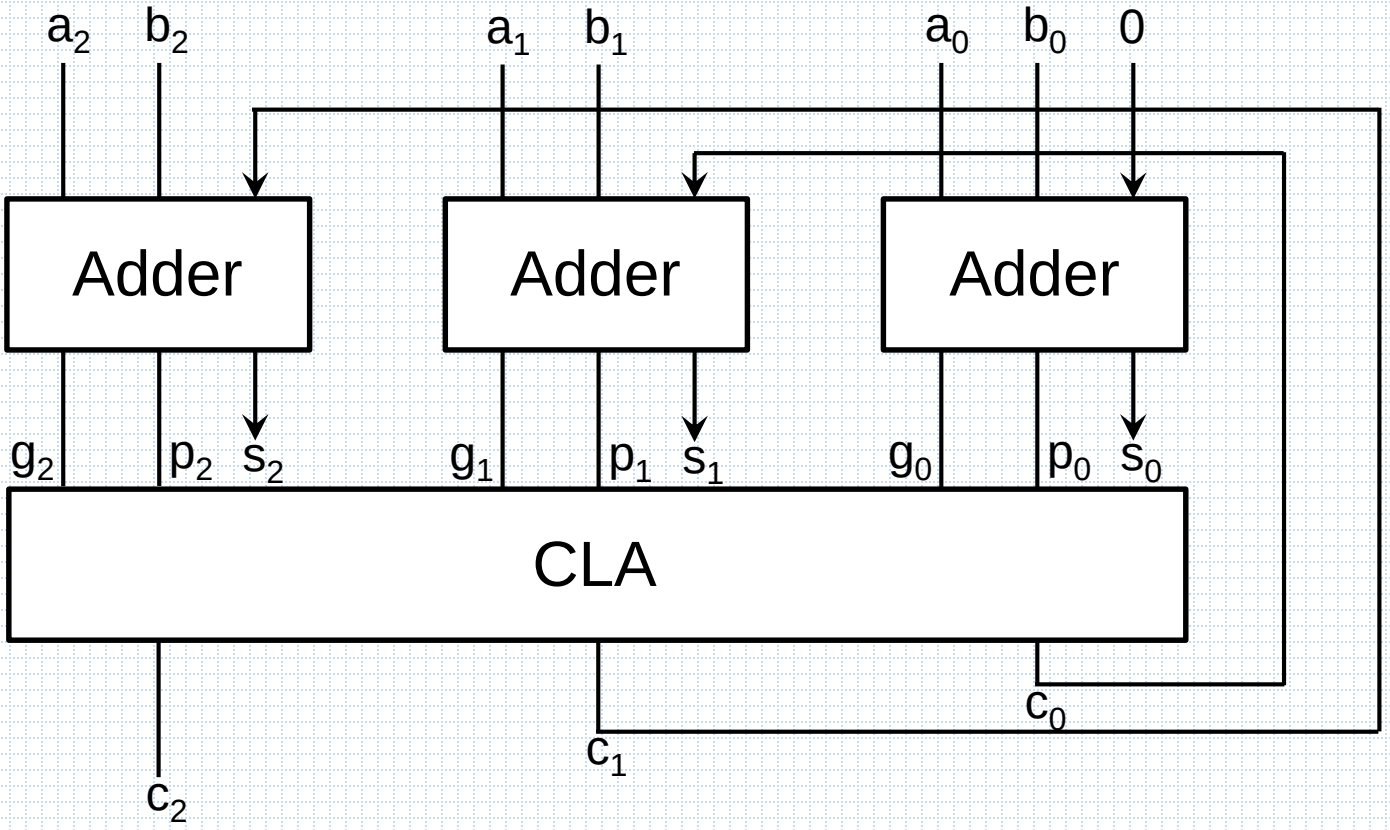


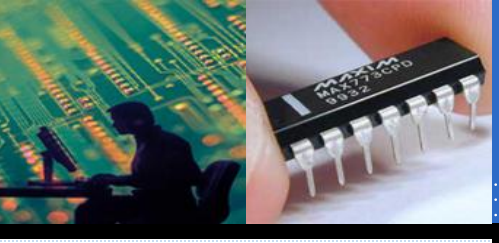




4.3.1. Bộ cộng nhị phân

- **Bộ cộng hai số nhị phân nhiều bit**
  - **Bộ cộng nhiều bit xây dựng từ mạch cộng thấy trước nhớ – Carry Look-ahead Adder (CLA)**





4.3.2. Bộ trừ nhị phân

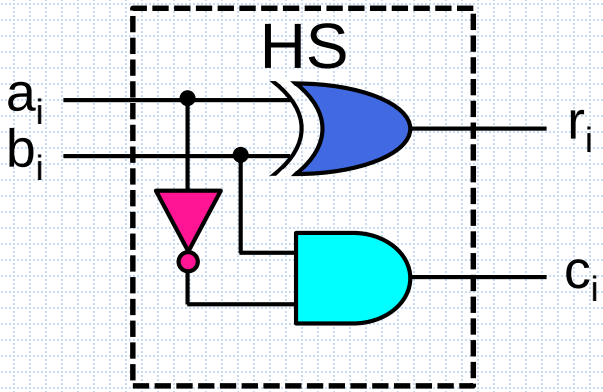
- **Mạch trừ 1 bit không nhớ - Mạch trừ bán phần (HS – Half Subtractor):** Thực hiện phép toán trừ hai số nhị phân 1 bit a_i và b_i, không có bit nhớ từ các bước tính toán trước đó. Kết quả cho ra bit hiệu r_i và bit nhớ c_i

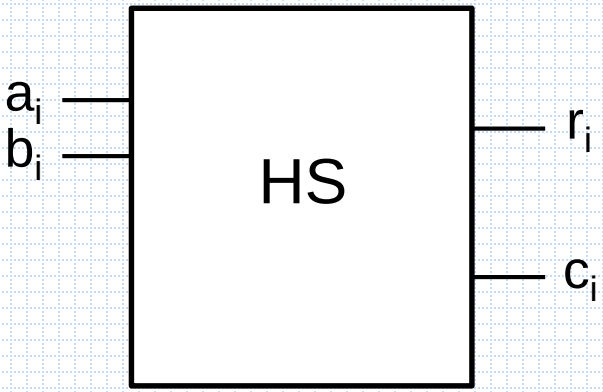
  - **Bảng giá trị hàm và biểu thức logic của mạch trừ bán phần**

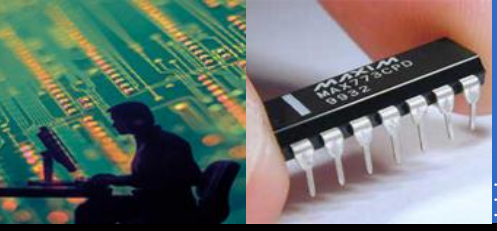
a_i	b_i	r_i	c_i
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$\begin{cases} r_i = a_i \oplus b_i \\ c_i = \overline{a}_i \cdot b_i \end{cases}$$

  - **Sơ đồ logic và sơ đồ khối mạch trừ bán phần**







4.3.2. Bộ trừ nhị phân

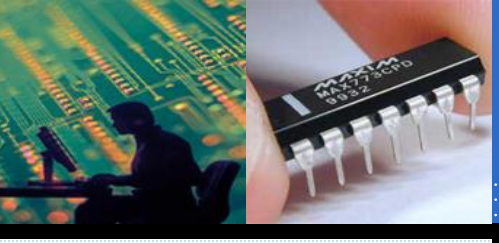
- **Mạch trừ một bit có nhớ - Mạch trừ toàn phần (FA – Full Subtractor):**
  Thực hiện phép toán trừ 2 số nhị phân 1 bit a_i và b_i có tính tới bit nhớ c_{i-1} từ các bước tính toán trước đó. Kết quả cho ra bit hiệu r_i và bit nhớ c_i

➢ **Bảng giá trị hàm và biểu thức logic của mạch trừ toàn phần**

$$\begin{cases} r_i = \overline{a_i}\cdot\overline{b_i}\cdot c_{i-1} + \overline{a_i}\cdot b_i\cdot\overline{c_{i-1}} + a_i\cdot\overline{b_i}\cdot\overline{c_{i-1}} + a_i\cdot b_i\cdot c_{i-1} \\ c_i = \overline{a_i}\cdot\overline{b_i}\cdot c_{i-1} + \overline{a_i}\cdot b_i\cdot\overline{c_{i-1}} + \overline{a_i}\cdot b_i\cdot c_{i-1} + a_i\cdot b_i\cdot c_{i-1} \end{cases}$$

$$\begin{cases} r_i = a_i \oplus b_i \oplus c_{i-1} \\ c_i = \overline{(a_i \oplus b_i)}\cdot c_{i-1} + \overline{a_i}\cdot b_i \end{cases}$$

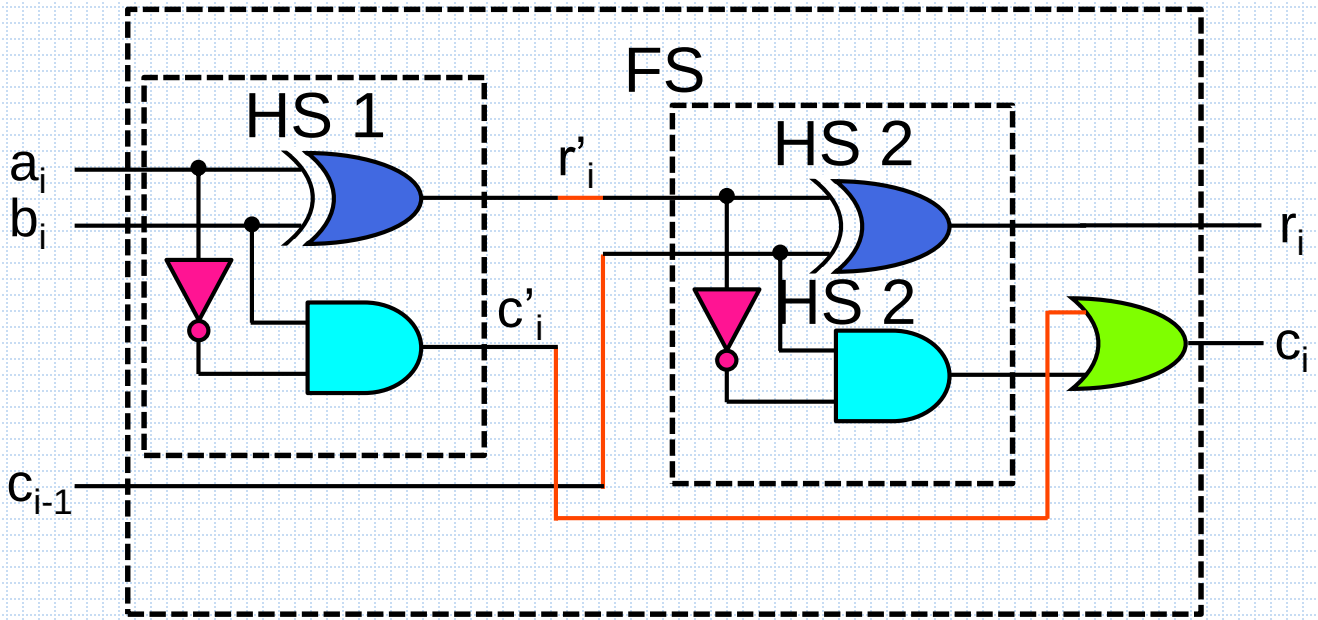
a_i	b_i	c_{i-1}	r_i	c_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

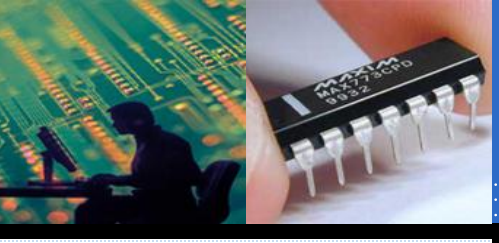


4.3.2. Bộ trừ nhị phân

➢ **Sơ đồ logic và sơ đồ khối tương đương mạch cộng toàn phần**

$$\begin{cases} r_i = a_i \oplus b_i \oplus c_{i-1} \\ c_i = (\overline{a_i \oplus b_i}) \cdot c_{i-1} + \overline{a}_i \cdot b_i \end{cases}$$





4.3.3. Bộ so sánh nhị phân

- **Mạch so sánh hai bit đầy đủ - FC (Full Comparator):** Thực hiện phép toán so sánh hai số nhị phân 1 bit a_i và b_i. Đầu ra của mạch có ba tín hiệu để thông báo kết quả so sánh
  - $a_i > b_i <=> (M_i, L_i, E_i) = (1, 0, 0)$
  - $a_i < b_i <=> (M_i, L_i, E_i) = (0, 1, 0)$
  - $a_i = b_i <=> (M_i, L_i, E_i) = (0, 0, 1)$

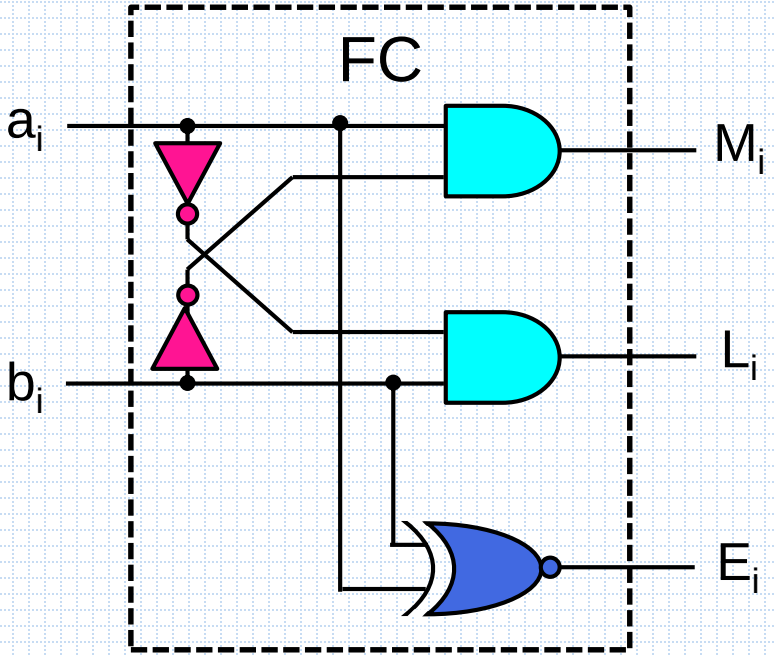
➢ **Bảng giá trị hàm**

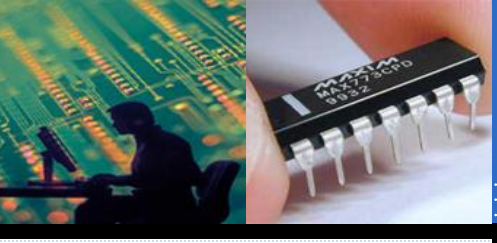
a_i	b_i	M_i	L_i	E_i
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

➢ **Biểu thức đại số**

$$\begin{cases} M_i = a_i \cdot \overline{b_i} \\ L_i = \overline{a_i} \cdot b_i \\ E_i = \overline{a_i \oplus b_i} \end{cases}$$

➢ **Sơ đồ logic**

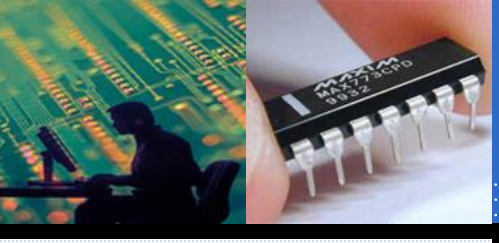




4.3.3. Bộ so sánh nhị phân

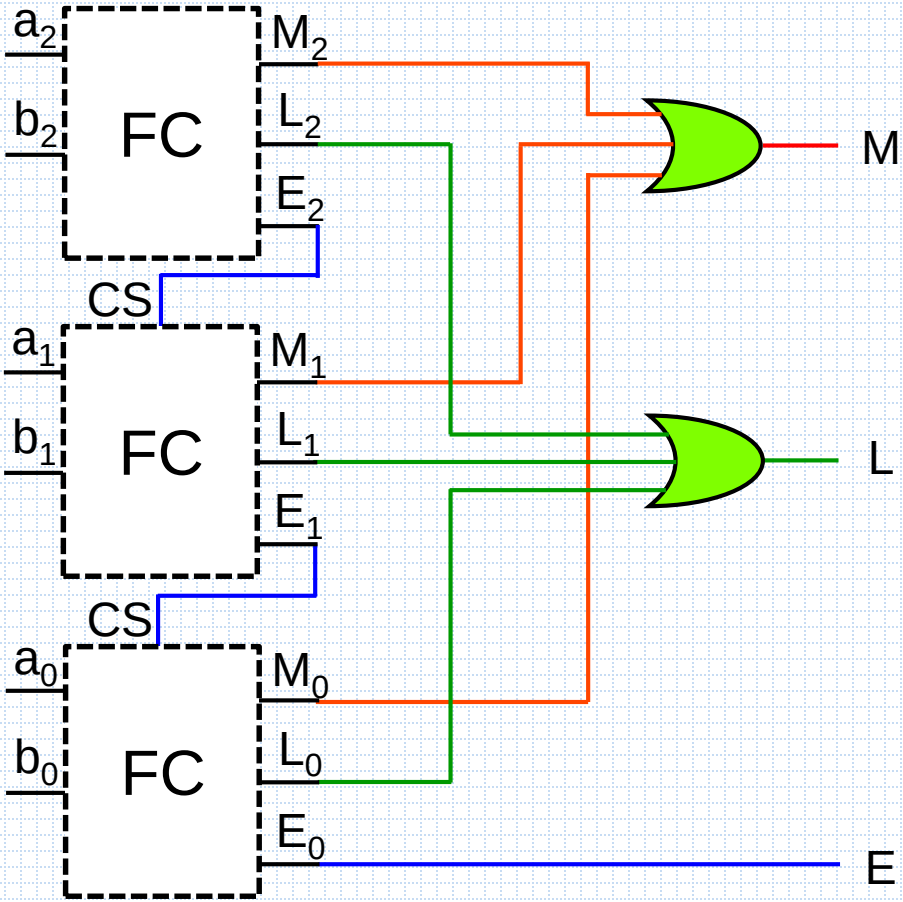
- **Mạch so sánh đầy đủ hai số nhị phân:** Thực hiện phép toán so sánh hai số nhị phân A và B. Đầu ra của mạch có ba tín hiệu để thông báo kết quả so sánh.
  - **Cấu tạo mạch:** Gồm các bộ so sánh 2 bit đầy đủ FC ghép lại, sử dụng thêm tín hiệu chọn chip CS – Chip Select. Nếu CS bằng 0 tất cả các đầu ra của FC tương ứng bằng 0, nếu CS bằng 1 FC tương ứng hoạt động bình thường
  - **Biểu thức đại số các đầu ra FC**

$$\begin{cases} M_i = CS\cdot(a_i\cdot\overline{b_i}) \\ L_i = CS\cdot(\overline{a_i}\cdot b_i) \\ E_i = CS\cdot\overline{(a_i \oplus b_i)} \end{cases}$$



4.3.3. Bộ so sánh nhị phân

- **Mạch so sánh đầy đủ hai số nhị phân**
  - **Sơ đồ logic**



Kết thúc chương 4