🕮 CHƯƠNG 4: MẠCH TỔ HỢP

❋ MẠCH MÃ HÓA
❄ Mạch mã hóa 2^n đường sang n đường
❄ Mạch tạo mã BCD cho số thập phân
❋ MẠCH GIẢI MÃ
❄ Mạch giải mã n đường sang 2^n đường
❄ Mạch giải mã BCD sang 7 đoạn
❋ MẠCH ĐA HỢP VÀ GIẢI ĐA HỢP
❄ Khái niệm
❄ Mạch đa hợp
❄ Ứng dụng của mạch đa hợp
❄ Mạch giải đa hợp
❋ MẠCH SO SÁNH
❄ Mạch so sánh hai số một bit
❄ Mạch so sánh hai số nhiều bit
❋ MẠCH KIỂM / PHÁT CHẴN LẺ
❄ Mạch phát chẵn lẻ
❄ Mạch kiểm chẵn lẻ

Các mạch số được chia ra làm hai loại: Mạch tổ hợp và Mạch tuần tự.

- Mạch tổ hợp: Trạng thái ngã ra chỉ phụ thuộc vào tổ hợp các ngã vào khi tổ hợp này đã ổn định. Ngã ra Q của mạch tổ hợp là hàm logic của các biến ngã vào A, B, C . . ..

$$Q = f(A,B,C\ldots)$$

- Mạch tuần tự : Trạng thái ngã ra không những phụ thuộc vào tổ hợp các ngã vào mà còn phụ thuộc trạng thái ngã ra trước đó. Ta nói mạch tuần tự có tính nhớ. Ngã ra Q_+ của mạch tuần tự là hàm logic của các biến ngã vào A, B, C . . . . và ngã ra Q trước đó.

$$Q_+ = f(Q,A,B,C\ldots)$$

Chương này nghiên cứu một số mạch tổ hợp thông dụng thông qua việc thiết kế một số mạch đơn giản và khảo sát một số IC trên thực tế.

4.1. MẠCH MÃ HÓA

Mã hóa là gán các ký hiệu cho các đối tượng trong một tập hợp để thuận tiện cho việc thực hiện một yêu cầu cụ thể nào đó. Thí dụ mã BCD gán số nhị phân 4 bit cho từng số mã của số thập phân (từ 0 đến 9) để thuận tiện cho máy đọc một số có nhiều số mã; mã Gray dùng tiện lợi trong việc tối giản các hàm logic . . .. Mạch chuyển từ mã này sang mã khác gọi là mạch chuyển mã, cũng được xếp vào loại mạch mã hóa. Thí dụ mạch chuyển số nhị phân 4 bit sang số Gray là một mạch chuyển mã.

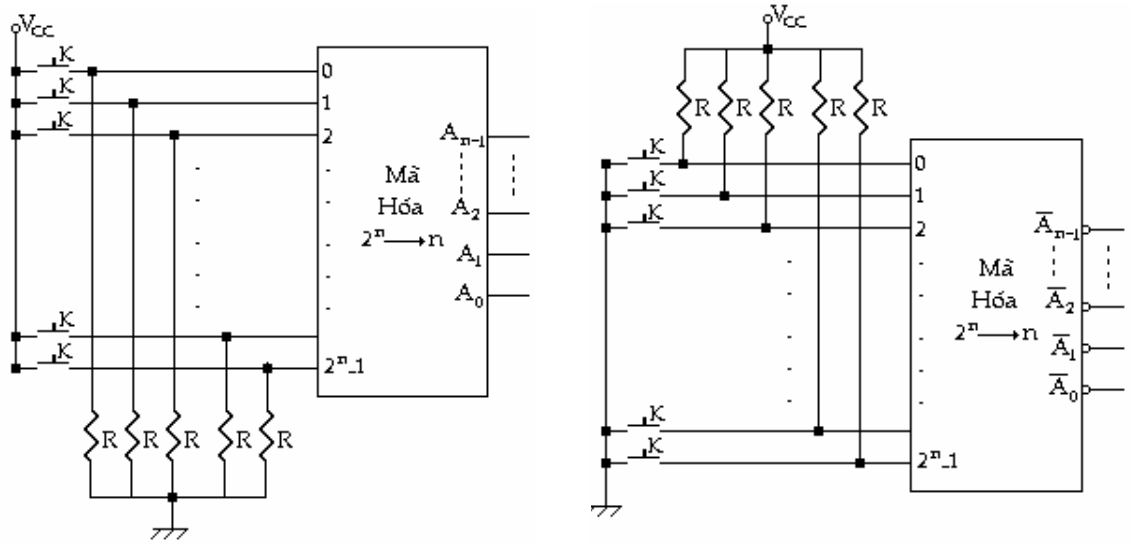
4.1.1 Mạch mã hóa 2^n đường sang n đường

Một số nhị phân n bit cho 2^n tổ hợp số khác nhau. Vậy ta có thể dùng số n bit để mã cho 2^n ngã vào khác nhau, khi có một ngã vào được chọn bằng cách đưa nó lên mức tác động, ở ngã ra sẽ chỉ báo số nhị phân tương ứng. Đó là mạch mã hóa 2^n đường sang n đường.

(H 4.1) là mô hình một mạch mã hóa 2^n đường sang n đường.

- (H 4.1a) là mạch có ngã vào và ra tác động cao : Khi các ngã vào đều ở mức thấp, mạch chưa hoạt động, các ngã ra đều ở mức thấp. Khi có một ngã vào được tác động bằng cách ấn khóa K tương ứng để đưa ngã vào đó lên mức cao, các ngã ra sẽ cho số nhị phân tương ứng.
- (H 4.1b) là mạch có ngã vào và ra tác động thấp. Hoạt động tương tự như mạch trên nhưng có mức tác động ngược lại. (trong mô hình (H 4.1b) ký hiệu dấu o ở ngã ra để chỉ mức tác động thấp, còn ở ngã vào không có dấu o vì là mạch thật)

Trong trường hợp ngã ra có mức tác động thấp, muốn đọc đúng số nhị phân ở ngã ra, ta phải đảo các bit để đọc.



(a) (b)

(H 4.1)

Dĩ nhiên, người ta cũng có thể thiết kế theo kiểu ngã vào tác động thấp và ngã ra tác động cao hay ngược lại. Trên thực tế, ta có thể có bất cứ loại ngã vào hay ra tác động theo bất cứ kiểu nào (mức cao hay thấp).

Ngoài ra, để tránh trường hợp mạch cho ra một mã sai khi người sử dụng vô tình (hay cố ý) tác động đồng thời vào hai hay nhiều ngã vào, người ta thiết kế các mạch mã hóa ưu tiên: là mạch chỉ cho ra một mã duy nhất có tính ưu tiên khi có nhiều ngã vào cùng được tác động.

4.1.1.1 Mã hóa ưu tiên 4 đường sang 2 đường

Thiết kế mạch mã hóa 4 đường sang 2 đường, ưu tiên cho mã có trị cao, ngã vào và ra tác động cao

Bảng sự thật và sơ đồ mạch (H 4.2)

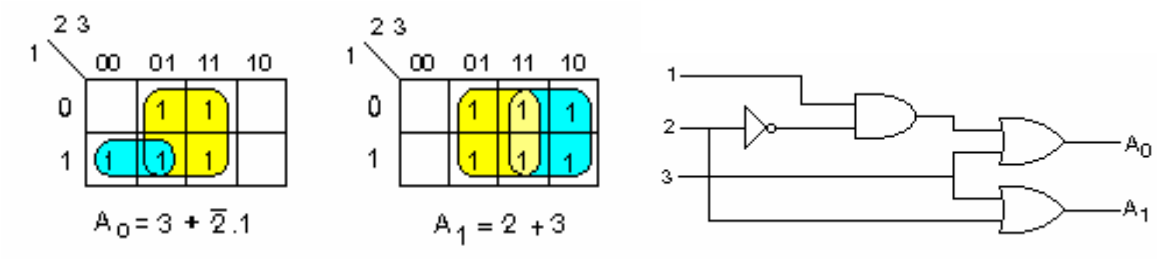
0	1	2	3	A_1	A_0
1	0	0	0	0	0

x	1	0	0	0	1
x	x	1	0	1	0
x	x	x	1	1	1

Bảng 4.1

Nhận thấy biến 0 trong bảng sự thật không ảnh hưởng đến kết quả nên ta chỉ vẽ bảng Karnaugh cho 3 biến 1, 2 và 3. Lưu ý là do trong bảng sự thật có các trường hợp bất chấp của biến nên ứng với một trị riêng của hàm ta có thể có đến 2 hoặc 4 số 1 trong bảng Karnaugh. Thí dụ với trị 1 của cả 2 hàm A_1 và A_0 ở dòng cuối cùng đưa đến 4 số 1 trong các ô 001, 011, 101 và 111 của 3 biến 123.

Từ bảng Karnaugh, ta có kết quả và mạch tương ứng. Trong mạch không có ngã vào 0, điều này được hiểu là mạch sẽ chỉ báo số 0 khi không tác động vào ngã vào nào.



(H 4.2)

4.1.1.2 Mã hóa 8 đường sang 3 đường

Chúng ta sẽ khảo sát một IC mã hóa 8 đường sang 3 đường.

Trên thực tế khi chế tạo một IC, ngoài các ngã vào/ra để thực hiện chức năng chính của nó, người ta thường dự trù thêm các ngã vào và ra cho một số chức năng khác như cho phép, nối mạch để mở rộng hoạt động của IC.

IC 74148 là IC mã hóa ưu tiên 8 đường sang 3 đường, vào/ ra tác động thấp, có các ngã nối mạch để mở rộng mã hóa với số ngã vào nhiều hơn.

Dưới đây là bảng sự thật của IC 74148, trong đó E_i ngã vào nối mạch và cho phép, E_o là ngã ra nối mạch và G_s dùng để mở rộng cho số nhị phân ra.

Dựa vào bảng sự thật, ta thấy IC làm việc theo 10 trạng thái:

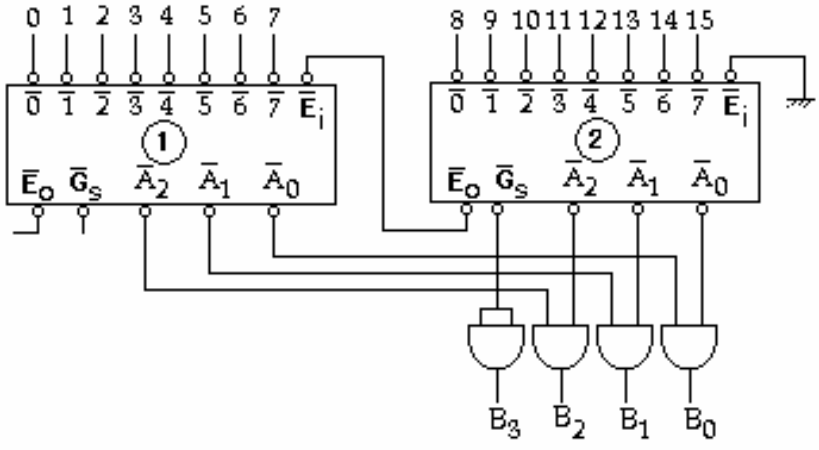
- Các trạng thái từ 0 đến 7: IC mã hóa cho ra số 3 bit
- Các trạng thái 8 và 9: dùng cho việc mở rộng, sẽ giải thích rõ hơn khi nối 2 IC để mở rộng mã hóa cho số 4 bit

Trạng thái	Ngã vào								Ngã ra				
	E_i	0	1	2	3	4	5	6	A_2	A_1	A_0	G_s	E_o
		7											
9	1	x	x	x	x	x	x	x	1	1	1	1	1
8	0	x							1	1	1	1	0
7	0	1	1	1	1	1	1	1	0	0	0	0	1
6	0	1							0	0	1	0	1
5	0	x	x	x	x	x	x	x	0	1	0	0	1
4	0	0							0	1	1	0	1

3	0	x	x	x	x	x	x	0	1	0	0	0	1
2	0	1							1	0	1	0	1
1	0	x	x	x	x	x	0	1	1	1	0	0	1
0	0	1							1	1	1	0	1
		x	x	x	x	0	1	1					
		1											
		x	x	x	0	1	1	1					
		1											
		x	x	0	1	1	1	1					
		1											
		x	0	1	1	1	1	1					
		1											
		0	1	1	1	1	1	1					
		1											

Bảng 4.2

(H 4.3) là cách nối 2 IC để thực hiện mã hóa 16 đường sang 4 đường



(H 4.3)

- IC2 có $E_i = 0$ nên hoạt động theo các trạng thái từ 0 đến 8, nghĩa là mã hóa từ 0 đến 7 cho các ngã ra $A_2A_1A_0$.

- IC1 có E_i nối với E_o của IC2 nên IC1 chỉ hoạt động khi tất cả ngã vào dữ liệu của IC2 lên mức 1 (IC2 hoạt động ở trạng thái 8)

* Để mã hóa các số từ 0 đến 7, cho các ngã vào 8 đến 15 (tức các ngã vào dữ liệu của IC2) lên mức 1, IC2 hoạt động ở trạng thái 8.

Lúc đó $E_{i1} = E_{o2} = 0$: kết quả là IC1 sẽ hoạt động ở trạng thái từ 0 đến 7, cho phép tạo mã các số từ 0 đến 7 (từ 111 đến 000) và IC2 hoạt động ở trạng thái 8 nên các ngã ra $(A_2A_1A_0)_2$= 111, đây là điều kiện mở các cổng AND để cho mã số ra là $B_2B_1B_0 = A_2A_1A_0$ của IC1, trong lúc đó $B_3 = G_{s2} = 1$, ta được kết quả từ 1111 đến 1000, tức từ 0 đến 7 (tác động thấp).

Thí dụ để mã số 4 , đưa ngã vào 4 xuống mức 0, các ngã vào từ 5 đến 15 lên mức 1, bất chấp các ngã vào từ 0 đến 3, mã số ra là $B_3B_2B_1B_0=G_{s2}B_2B_1B_0$=1011, tức số 4

* Để mã hóa các số từ 8 đến 15, cho IC2 hoạt động ở trạng thái từ 0 đến 7 (đưa ngã vào ứng với số muốn mã xuống thấp, các ngã vào cao hơn lên mức 1 và các ngã vào thấp hơn xuống mức 0), bất chấp các ngã vào dữ liệu của IC1 (cho IC1 hoạt động ở trạng thái 9), nên các ngã ra $(A_2A_1A_0)_1$=111, đây là điều kiện mở các cổng AND để cho mã số ra là $B_2B_1B_0$=

$A_2A_1A_0$ của IC2, , trong lúc đó $B_3 = G_{s2} = 0$, ta được kết quả từ 0111 đến 0000, tức từ 8 đến 15.

Thí dụ để mã số 14, đưa ngã vào 14 xuống mức 0, đưa ngã vào 15 lên mức 1, bất chấp các ngã vào từ 0 đến 13, mã số ra là $B_3B_2B_1B_0 = G_{s2}B_2B_1B_0 = 0001$, tức số 14

Muốn có ngã ra chỉ số nhị phân đúng với ngã vào được tác động mà không phải đảo các bit ta có thể thay các cổng AND bằng cổng NAND

4.1.2 Mạch tạo mã BCD cho số thập phân

Mạch gồm 10 ngã vào tượng trưng cho 10 số thập phân và 4 ngã ra là 4 bit của số BCD. Khi một ngã vào (tượng trưng cho một số thập phân) được tác động bằng cách đưa lên mức cao các ngã ra sẽ cho số BCD tương ứng

Bảng sự thật của mạch:

Trạng thái các ngã vào									Mã số ra		
9	8	7	6	5	4	3	2	1	0	A_3	A_2	A_1	A_0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	1

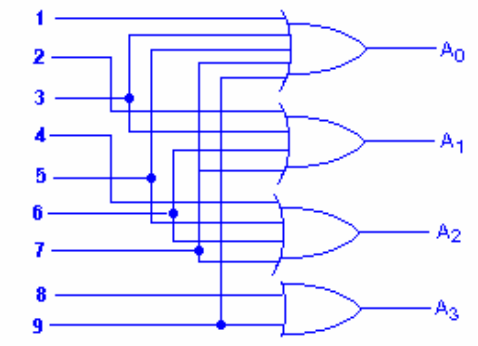
Bảng 4.3

Không cần bảng Karnaugh ta có thể viết ngay các hàm xác định các ngã ra:

$A_0 = 1 + 3 + 5 + 7 + 9$ $A_1 = 2 + 3 + 6 + 7$

$A_2 = 4 + 5 + 6 + 7$ $A_3 = 8 + 9$

Mạch cho ở (H 4.4)



(H 4.4)

Để tạo mã BCD ưu tiên cho số lớn, ta viết lại bảng sự thật và dùng phương pháp đại số để đơn giản các hàm xác định các ngã ra A_3 , A_2 , A_1 , A_0

Trạng thái các ngã vào								Mã số ra		
9	8	7	6	5	4	3	2	A_3	A_2	A_1 A_0
0	0	0	0	0	0	0	0	0	0	0 0
0	0	0	0	0	0	0	0	0	0	0 1
0	0	0	0	0	0	0	1	0	0	1 0
0	0	0	0	0	0	1	x	0	0	1 1
0	0	0	0	0	1	x	x	0	1	0 0
0	0	0	0	1	x	x	x	0	1	0 1
0	0	0	1	x	x	x	x	0	1	1 0
0	0	1	x	x	x	x	x	0	1	1 1
0	1	x	x	x	x	x	x	1	0	0 0
1	x	x	x	x	x	x	x	1	0	0 1

Bảng 4.4

$$A_3 = \bar{9}.8 + 9 = 9 + 8$$

$$A_2 = 7\bar{8}.\bar{9} + 6.7\bar{8}.\bar{9} + 5.6.7\bar{8}.\bar{9} + 4.5.6.7\bar{8}.\bar{9} = (7 + 6.\bar{7} + 5.6.\bar{7} + 4.5.6.7)\bar{8}.\bar{9}$$

$$A_2 = (7 + 6 + 5 + 4)\bar{8}.\bar{9} = (7 + 6 + 5 + 4)\overline{(8 + 9)}$$

$$A_1 = 7\bar{8}.\bar{9} + 6.7\bar{8}.\bar{9} + 3.\bar{4}.5.6.7\bar{8}.\bar{9} + 2.\bar{3}.\bar{4}.5.6.7\bar{8}.\bar{9} = (7 + 6.\bar{7} + 3.\bar{4}.5.6.\bar{7} + 2.\bar{3}.\bar{4}.5.6.7)\bar{8}.\bar{9}$$

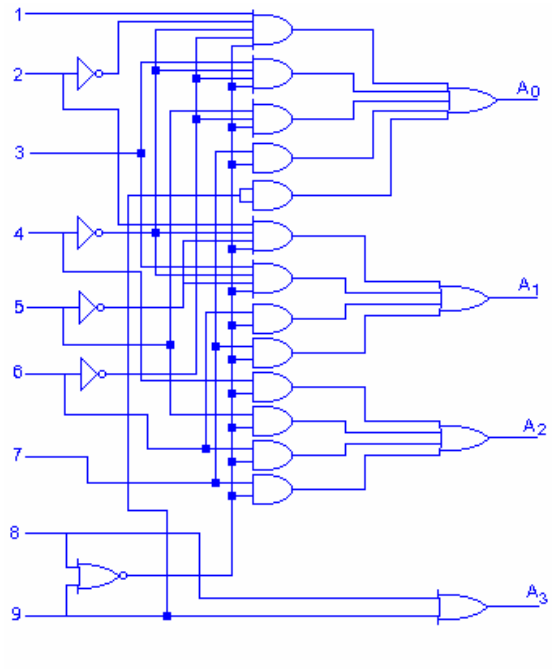
$$A_1 = (7 + 6 + 3.\bar{4}.\bar{5} + 2.\bar{3}.\bar{4}.5)\bar{8}.\bar{9} = (7 + 6 + 3.\bar{4}.\bar{5} + 2.\bar{4}.5)\overline{(8 + 9)}$$

$$A_0 = 9 + 7\bar{8}.\bar{9} + 5.\bar{6}.\bar{7}\bar{8}.\bar{9} + 3.\bar{4}.5.6.\bar{7}\bar{8}.\bar{9} + 1.\bar{2}.\bar{3}.\bar{4}.5.6.\bar{7}\bar{8}.\bar{9}$$

$$= 9 + (7 + 5.\bar{6}.\bar{7} + 3.\bar{4}.\bar{5}.\bar{6}.\bar{7} + 1.\bar{2}.\bar{3}.\bar{4}.\bar{5}.\bar{6}.\bar{7})\bar{8}.\bar{9}$$

$$A_0 = 9 + (7 + 5.\bar{6} + 3.\bar{4}.\bar{6} + 1.\bar{2}.\bar{4}.\bar{6})\bar{8}.\bar{9} = 9 + (7 + 5.\bar{6} + 3.\bar{4}.\bar{6} + 1.\bar{2}.\bar{4}.\bar{6})(\overline{8+9})$$

Mạch cho ở (H 4.5)



(H 4.5)

4.1.3 Mạch chuyển mã

Mạch chuyển từ một mã này sang một mã khác cũng thuộc loại mã hóa.

❁Mạch chuyển mã nhị phân sang Gray

Thử thiết kế mạch chuyển từ mã nhị phân sang mã Gray của số 4 bit.

Trước tiên viết bảng sự thật của số nhị phân và số Gray tương ứng. Các số nhị phân là các biến và các số Gray sẽ là hàm của các biến đó.

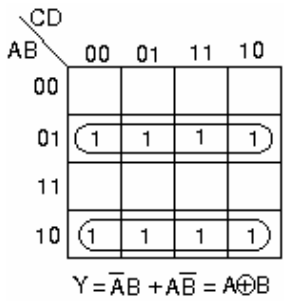
A	B	C	D		X	Y	Z	T
0	0	0	0	→	0	0	0	0
0	0	0	1	→	0	0	0	1
0	0	1	0	→	0	0	1	1
0	0	1	1	→	0	0	1	0
0	1	0	0	→	0	1	1	0
0	1	0	1	→	0	1	1	1
0	1	1	0	→	0	1	0	1
0	1	1	1	→	0	1	0	0
1	0	0	0	→	1	1	0	0
1	0	0	1	→	1	1	0	1
1	0	1	0	→	1	1	1	1
1	0	1	1	→	1	1	1	0
1	1	0	0	→	1	0	1	0
1	1	0	1	→	1	0	1	1
1	1	1	0	→	1	0	0	1
1	1	1	1	→	1	0	0	0
				→				

Bảng 4.5

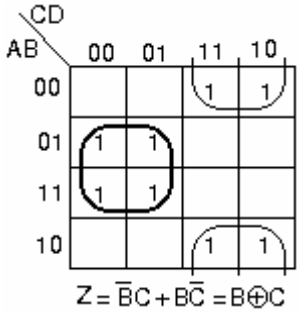
Dùng bảng Karnaugh để xác định X, Y, Z, T theo A, B, C, D

Quan sát bảng sự thật ta thấy ngay: X = A,

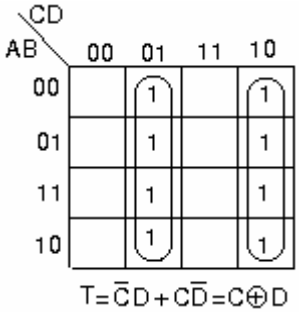
Vậy chỉ cần lập 3 bảng Karnaugh cho các biến Y, Z, T (H 4.6 a,b,c) và kết quả cho ở (H 4.6 d)



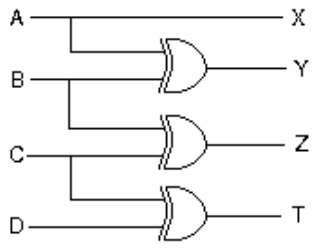
(a)



(b)



(c)



(H 4.6) (d)

4.2 . MẠCH GIẢI MÃ

4.2.1 Giải mã n đường sang 2^n đường

4.2.1.1 Giải mã 2 đường sang 4 đường:

Thiết kế mạch Giải mã 2 đường sang 4 đường có ngã vào cho phép (cũng được dùng để nối mạch)

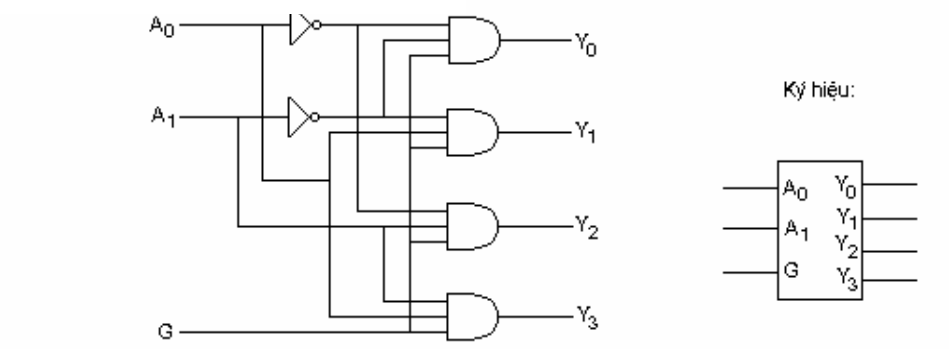
Để đơn giản, ta xét mạch giải mã 2 đường sang 4 đường có các ngã vào và ra đều tác động cao .

Bảng sự thật, các hàm ngã ra và sơ đồ mạch:

Vào			Ra			
G	A_1	A_0	Y_0	Y_1	Y_2	Y_3
0	**x**	**x**	**0**	**0**	**0**	**0**
1	**0**	**0**	**1**	**0**	**0**	**0**
1	**0**	**1**	**0**	**1**	**0**	**0**
1	**1**	**0**	**0**	**0**	**1**	**0**
1	**1**	**1**	**0**	**0**	**0**	**1**

00

$$Y_0 = G.\overline{A}_1\overline{A}_0$$
$$Y_1 = G.\overline{A}_1 A_0$$
$$Y_2 = G.A_1\overline{A}_0$$
$$Y_3 = G.A_1 A_0$$



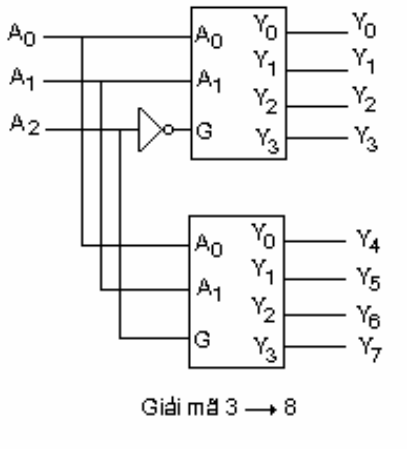
(H 4.7)

4.2.1.2 Giải mã 3 đường sang 8 đường

Dùng 2 mạch giải mã 2 đường sang 4 đường để thực hiện mạch giải mã 3 đường sang 8 đường (H 4.8)

Vào			Ra							
A_2	A_1	A_0	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Quan sát bảng sự thật ta thấy: Trong các tổ hợp số 3 bit có 2 nhóm trong đó các bit thấp A_1A_0 hoàn toàn giống nhau, một nhóm có bit $A_2 = 0$ và nhóm kia có $A_2 = 1$. Như vậy ta có thể dùng ngã vào G cho bit A_2 và mắc mạch như sau.



(H 4.8)

Khi A_2=G=0, IC1 giải mã cho 1 trong 4 ngã ra thấp và khi A_2=G=1, IC2 giải mã cho 1 trong 4 ngã ra cao

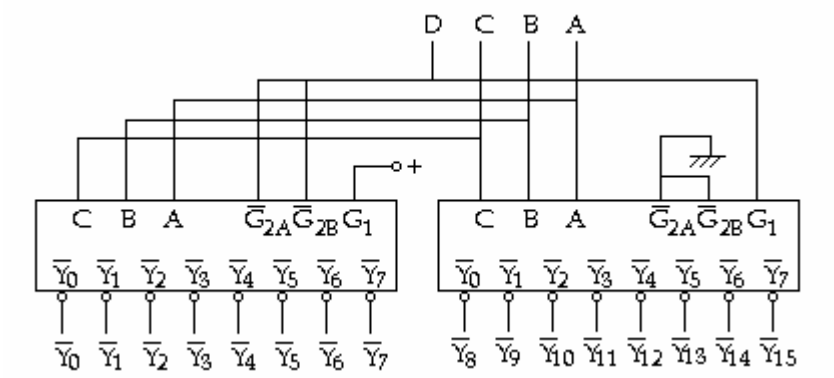
Trên thị trường hiện có các loại IC giải mã như:

- 74139 là IC chứa 2 mạch giải mã 2 đường sang 4 đường, có ngã vào tác động cao, các ngã ra tác động thấp, ngã vào cho phép tác động thấp.
- 74138 là IC giải mã 3 đường sang 8 đường có ngã vào tác động cao, các ngã ra tác động thấp, hai ngã vào cho phép G_{2A} và G_{2B} tác động thấp, G_1 tác động cao.
- 74154 là IC giải mã 4 đường sang 16 đường có ngã vào tác động cao, các ngã ra tác động thấp, 2 ngã vào cho phép E_1 và E_2 tác động thấp

Dưới đây là bảng sự thật của IC 74138 và cách nối 2 IC để mở rộng mạch giải mã lên 4 đường sang 16 đường (H 4.9)

Vào					Ra							
Cho phép		Dữ liệu										
G_1	G_2	C	B	A	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
x	H	x	x	x	H	H	H	H	H	H	H	H
L	x	x	x	x	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

Ghi chú $G_2 = G_{2A}+G_{2B}$, H = 1, L =0, x: bất chấp



(H 4.9)

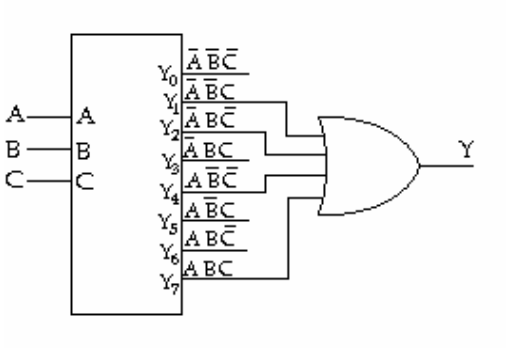
Một ứng dụng quan trọng của mạch giải mã là dùng giải mã địa chỉ cho bộ nhớ bán dẫn.

Ngoài ra, mạch giải mã kết hợp với một cổng OR có thể tạo được hàm logic.

Thí dụ, thiết kế mạch tạo hàm $Y=f(A,B,C)=\overline{A}BC+A\overline{B}\overline{C}+A\overline{B}C+ABC$

Với hàm 3 biến, ta dùng mạch giải mã 3 đường sang 8 đường. 8 ngã ra mạch giải mã tương ứng với 8 tổ hợp biến của 3 biến, các ngã ra tương ứng với các tổ hợp biến có trong hàm sẽ lên mức 1. Với một hàm đã viết dưới dạng tổng chuẩn, ta chỉ cần dùng một cổng OR có số ngã vào bằng với số tổ hợp biến trong hàm nối vào các ngã ra tương ứng của mạch giải mã để cộng các tổ hợp biến có trong hàm lại ta sẽ được hàm cần tạo.

Như vậy, mạch tạo hàm trên có dạng (H 4.10)



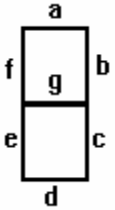
(H 4.10)

Dĩ nhiên, với những hàm chưa phải dạng tổng chuẩn, chúng ta phải chuẩn hóa. Và nếu bài toán có yêu cầu ta phải thực hiện việc đổi cổng, bằng cách dùng định lý De Morgan.

4.2.2 Giải mã BCD sang 7 đọan

4.2.2.1 Đèn 7 đọan

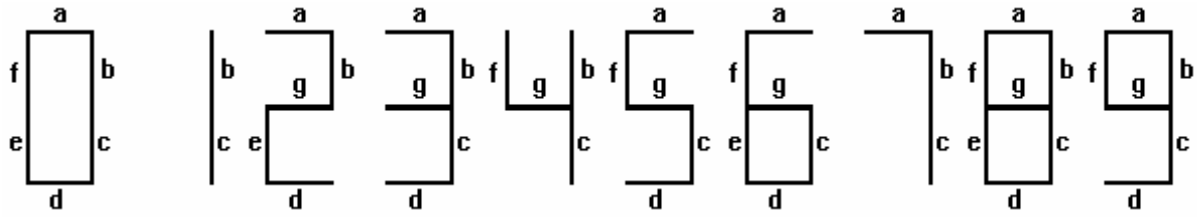
Đây là loại đèn dùng hiển thị các số từ 0 đến 9, đèn gồm 7 đọan a, b, c, d, e, f, g, bên dưới mỗi đọan là một led (đèn nhỏ) hoặc một nhóm led mắc song song (đèn lớn). Qui ước các đọan cho bởi (H 4.11).



(H 4.11)

Khi một tổ hợp các đọan cháy sáng sẽ tạo được một con số thập phân từ 0 - 9.

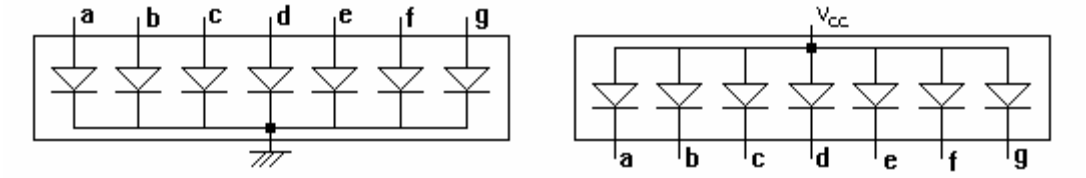
(H 4.12) cho thấy các đoạn nào cháy để thể hiện các số từ 0 đến 9



0 1 2 3 4 5 6 7 8 9

(H 4.12)

Đèn 7 đoạn cũng hiển thị được một số chữ cái và một số ký hiệu đặc biệt.
Có hai loại đèn 7 đoạn:
- Loại catod chung (H 4.13a), dùng cho mạch giải mã có ngã ra tác động cao.
- Loại anod chung (H 4.13b), dùng cho mạch giải mã có ngã ra tác động thấp.



(a) (H 4.13) (b)

4.2.2.2 Mạch giải mã BCD sang 7 đoạn :

Mạch có 4 ngã vào cho số BCD và 7 ngã ra thích ứng với các ngã vào a, b, c, d, e, f, g của led 7 đoạn, sao cho các đoạn cháy sáng tạo được số thập phân đúng với mã BCD ở ngã vào.

Bảng sự thật của mạch giải mã 7 đoạn, có ngã ra tác động thấp:

Số	Ngã vào				Ngã ra						
TP	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	1	1	0	0	1	1	1	1
2	0	0	1	0	0	0	1	0	0	1	0
3	0	0	1	1	0	0	0	0	1	1	0
4	0	1	0	0	1	0	0	1	1	0	0
5	0	1	0	1	0	1	0	0	1	0	0
6	0	1	1	0	0	1	0	0	0	0	0
7	0	1	1	1	0	0	0	1	1	1	1
8	1	0	0	0	0	0	0	0	0	0	0
9	1	0	0	1	0	0	0	0	1	0	0

Bảng 4.6

Dùng Bảng Karnaugh hoặc có thể đơn giản trực tiếp với các hàm chứa ít tổ hợp, ta có kết quả:

$$a = \overline{D}\,\overline{B}(C\overline{A} + \overline{C}A)$$
$$b = C\overline{B}A + CB\overline{A}$$
$$c = \overline{D}\,\overline{C}B\overline{A}$$
$$d = \overline{D}\,\overline{C}\,\overline{B}A + C\overline{B}\,\overline{A} + CBA$$
$$e = A + C\overline{B}$$
$$f = \overline{C}B + BA + \overline{D}\,\overline{C}A$$
$$g = \overline{D}\,\overline{C}\,\overline{B} + CBA$$

Từ các kết quả ta có thể vẽ mạch giải mã 7 đoạn dùng các cổng logic.
Hai IC thông dụng dùng để giải mã BCD sang 7 đoạn là:
- CD 4511 (loại CMOS, ngã ra tác động cao và có đệm)
- 7447 (loại TTL, ngã ra tác động thấp, cực thu để hở)

Chúng ta khảo sát một IC giải mã BCD sang 7 đoạn : IC 7447
Bảng sự thật của 7447:

Vào								Ra						
Sô / Hàm	LT	RBI	D	C	B	A	BI (1) $\overline{RBO}$	a	b	c	d	e	f	g
0	1	1	0	0	0	0	1	0	0	0	0	0	0	1
1	1	x	0	0	0	1	1	1	0	0	1	1	1	1
2	1	x	0	0	1	0	1	0	0	1	0	0	1	0
3	1	x	0	0	1	1	1	0	0	0	0	1	1	0
4	1	x	0	1	0	0	1	1	0	0	1	1	0	0
5	1	x	0	1	0	1	1	0	1	0	0	1	0	0
6	1	x	0	1	1	0	1	1	1	0	0	0	0	0
7	1	x	0	1	1	1	1	0	0	0	1	1	1	1
8	1	x	1	0	0	0	1	0	0	0	0	0	0	0
9	1	x	1	0	0	1	1	0	0	0	1	1	0	0
10	1	x	1	0	1	0	1	1	1	1	0	0	1	0
11	1	x	1	0	1	1	1	1	1	0	0	1	1	0
12	1	x	1	1	0	0	1	1	0	1	1	1	0	0
13	1	x	1	1	0	1	1	0	1	1	0	1	0	0
14	1	x	1	1	1	0	1	1	1	1	0	0	0	0
15	1	x	1	1	1	1	1	1	1	1	1	1	1	1
(2)	x	x	x	x	x	x	0	1	1	1	1	1	1	1
(3)	1	0	0	0	0	0	0	1	1	1	1	1	1	1
(4)	0	x	x	x	x	x	1	0	0	0	0	0	0	0

Ghi chú:

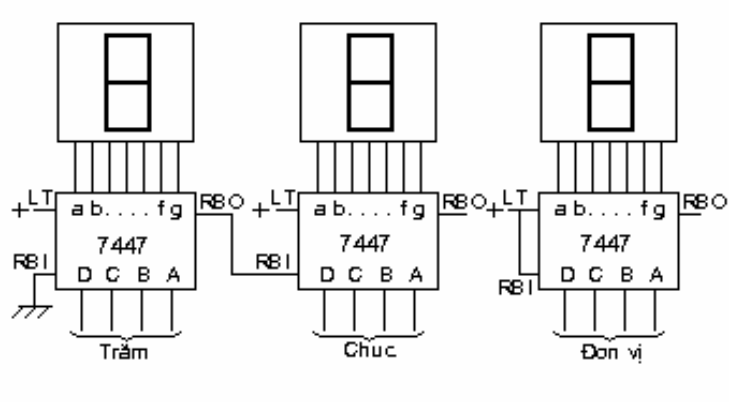
1. BI/RBO được nối theo kiểu điểm AND bên trong IC và được dùng như ngã vào xóa (Blanking Input, BI) và/hoặc ngã ra xóa dợn sóng (Ripple Blanking Output, RBO). Ngã vào BI phải được để hở hay giữ ở mức cao khi cần thực hiện giải mã cho số ra. Ngã vào xóa dợn sóng (Ripple Blanking Input, RBI) phải để hở hay ở mức cao khi muốn đọc số 0.

2. Khi đưa ngã vào BI xuống thấp, ngã ra lên 1 (không tác động) bất chấp các ngã vào còn lại. Ta nói IC làm việc dưới điều kiện bị ép buộc và đây là trường hợp duy nhất BI giữ vai trò ngã vào.

3. Khi ngã vào RBI ở mức 0 và A=B=C=D=0, tất cả các ngã ra kể cả RBO đều xuống 0. Ta nói IC làm việc dưới điều kiện đáp ứng.

4. Khi BI/RBO để hở hay được giữ ở mức 1 và ngã vào thử đèn (Lamp test, LT) xuống 0, tất cả các led đều cháy (ngã ra xuống 0).

Dựa vào bảng sự thật và các ghi chú 7447 là IC giải mã BCD sang 7 đoạn có đầy đủ các chức năng khác như : thử đèn, xóa số 0 khi nó không có nghĩa. Ta có thể hiểu rõ hơn chức năng này với thí dụ mạch hiển thị một kết quả có 3 chữ số sau đây: (H 4.14)



(H 4.14)

Vận hành của mạch có thể giải thích như sau:

- IC hàng đơn vị có ngã vào RBI đưa lên mức cao nên đèn số 0 hàng đơn vị luôn luôn được hiển thị (dòng 0 trong bảng sự thật), điều này là cần thiết để xác nhận rằng mạch vẫn chạy và kết quả giải mã là số 0.
- IC hàng chục có ngã vào RBI nối với ngã ra RBO của IC hàng trăm nên số 0 hàng chục chỉ được hiển thị khi số hàng trăm khác 0 (RBO=1) (dòng 0 đến 15).
- IC hàng trăm có ngã vào RBI đưa xuống mức thấp nên số 0 hàng trăm luôn luôn tắt (dòng ghi chú 3).

4.2.2.3 Hiển thị 7 đoạn bằng tinh thể lỏng (liquid crystal displays, LCD)

LCD gồm 7 đoạn như led thường và có chung một cực nền (backplane). Khi có tín hiệu xoay chiều biên độ khoảng 3 - 15 V_{RMS} và tần số khoảng 25 - 60 Hz áp giữa một đoạn và cực nền, thì đoạn đó được tác động và sáng lên.

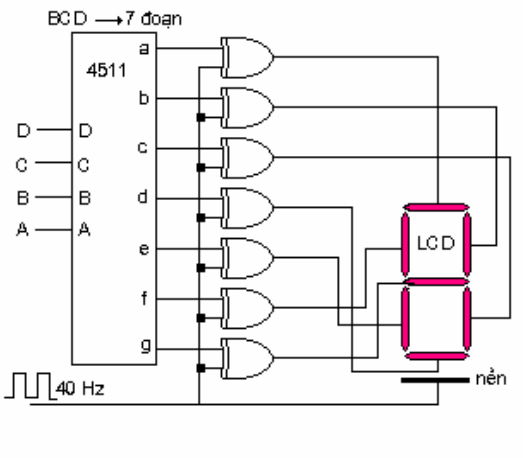
Trên thực tế người ta tạo hai tín hiệu nghịch pha giữa nền và một đoạn để tác động cho đoạn đó cháy.

Để hiểu được cách vận chuyển ta có thể dùng IC 4511 kết hợp với các cổng EX-OR để thúc LCD (H 4.15). Các ngã ra của IC 4511 (Giải mã BCD sang 7 đoạn, tác động cao) nối vào các ngã vào của các cổng EX-OR, ngã vào còn lại nối với tín hiệu hình vuông tần số khoảng 40 Hz (tần số thấp có thể gây ra nhấp nháy), tín hiệu này đồng thời được đưa vào nền. Khi một ngã ra mạch giải mã lên cao, ngã ra cổng EX-OR cho một tín hiệu đảo pha với tín hiệu ở nền, đoạn tương ứng xem như nhận được tín hiệu có biên độ gấp đôi và sẽ sáng lên. Với các ngã ra mạch giải mã ở mức thấp,
ngã ra cổng EX-OR cho một tín hiệu cùng pha với tín hiệu ở nền nên đoạn tương ứng không sáng.

Người ta thường dùng IC CMOS để thúc LCD vì hai lý do:

- CMOS tiêu thụ năng lượng rất thấp phù hợp với việc dùng pin cho các thiết bị dùng LCD.

- Mức thấp của CMOS đạt trị 0 và tín hiệu thúc LCD sẽ không chứa thành phần một chiều, tuổi thọ LCD được kéo dài. (Mức thấp của TTL khoảng 0,4 V, thành phần DC này làm giảm tuổi thọ của LCD).



(H 4.15)

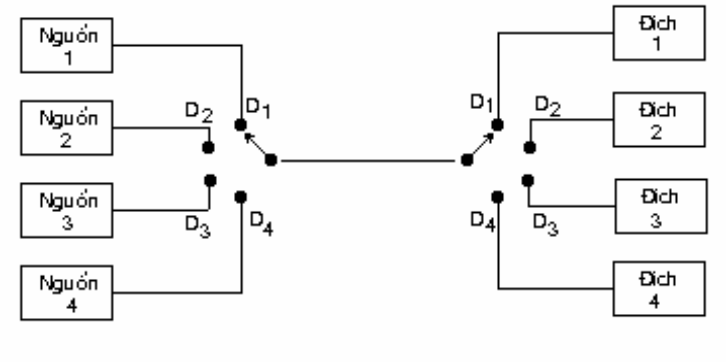
4.3 MẠCH ĐA HỢP VÀ MẠCH GIẢI ĐA HỢP

4.3.1.Khái niệm

Trong truyền dữ liệu, để tiết kiệm đường truyền, người ta dùng một đường dây để truyền nhiều kênh dữ liệu, như vậy phải thực hiện viêc chọn nguồn dữ liệu nào trong các nguồn khác nhau để truyền.

Mạch đa hợp hay còn gọi là mạch chọn dữ liệu sẽ làm công việc này.

Ở nơi thu, dữ liệu nhận được phải được chuyển tới các đích khác nhau, ta cần mạch phân bố dữ liệu hay giải đa hợp (H 4.16).



(H 4.16)

4.3.2 Mạch đa hợp

Còn được gọi là mạch chọn dữ liệu, gồm 2^n ngã vào dữ liệu, n ngã vào địa chỉ (hay điều khiển) và một ngã ra. Khi có một địa chỉ được tác động dữ liệu ở ngã vào tương ứng với địa chỉ đó sẽ được chọn.

- Thiết kế mạch đa hợp 4→1

Mạch có 4 ngã vào dữ liệu D_0 . . . . D_3, 2 ngã vào điều khiển AB và ngã ra Y

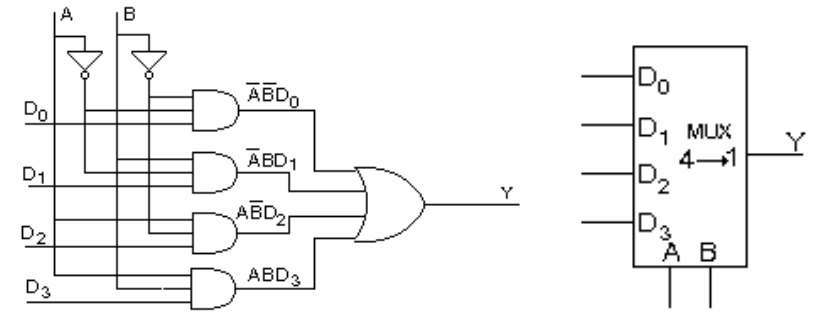
Bảng sự thật:

A	B	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

Tư bảng sự thật ta có hàm Y như sau:

$$Y = \overline{A}.\overline{B}D_0 + \overline{A}BD_1 + A\overline{B}D_2 + ABD_3$$

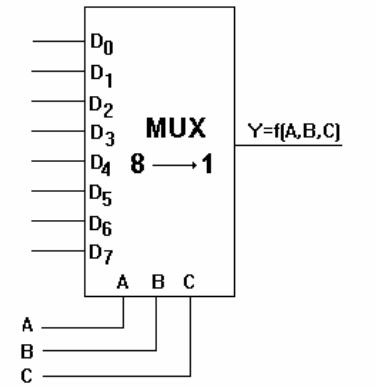
Và mạch có dạng (H 4.17)



(H 4.17)

Nếu chịu khó quan sát ta sẽ thấy mạch đa hợp 4→1 có thể được thiết kế từ mạch giải mã 2 đường sang 4 đường trong đó ngã vào cho phép G đã được tách riêng ra để làm ngã vào dữ liệu (D_0 . . . . D_3) và ngã vào dữ liệu của mạch giải mã đã trở thành ngã vào điều khiển của mạch đa hợp (A, B)

(H 4.18) là ký hiệu một mạch đa hợp với 8 ngã vào dữ liệu, 3 ngã vào điều khiển và 1 ngã ra, ta gọi là đa hợp 8 → 1.



Bảng sự thật:

A	B	C	Y
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

(H 4.18)

Một đa hợp 8 → 1 có ngã ra Y quan hệ với các ngã vào dữ liệu và điều khiển theo hàm :

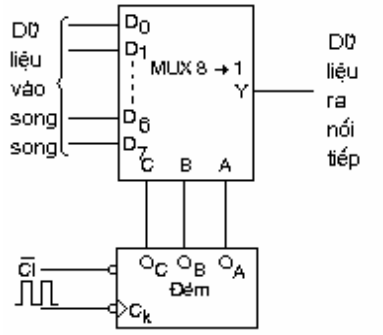
$$Y = \overline{A}.\overline{B}.\overline{C}D_0 + \overline{A}.\overline{B}.CD_1 + \overline{A}B\overline{C}D_2 + \overline{A}BCD_3 + A\overline{B}.\overline{C}D_4 + A\overline{B}CD_5 + AB\overline{C}D_6 + ABCD_7$$

4.3.3 Ứng dung mạch đa hợp

Ngoài chức năng chọn dữ liệu mạch đa hợp còn được dùng để:

4.3.3.1 Biến chuỗi dữ liệu song song thành nối tiếp:

Một mạch đa hợp kết hợp với một mạch đếm sẽ biến chuỗi dữ liệu song song ở ngã vào thành chuỗi dữ liệu nối tiếp ở ngã ra (H 4.19)



(H 4.19)

4.3.3.2 Tạo chuỗi xung tuần hoàn :

Nếu cho dữ liệu vào tuần hoàn, dữ liệu ra nối tiếp cũng tuần hoàn, như vậy chỉ cần đặt trước các ngã vào thay đổi theo một chu kỳ nào đó ta sẽ được chuỗi xung tuần hoàn ở ngã ra.

4.3.3.3 Tạo hàm:

❄ Một đa hợp $2^n \rightarrow 1$ có thể tạo hàm n biến bằng cách cho các biến vào ngã vào điều khiển và cho trị riêng của hàm vào các ngã vào dữ liệu.

Thí dụ: Để tạo hàm 3 biến bằng đa hợp 8→1 ta viết lại biểu thức của đa hợp

$$Y = \overline{A}.\overline{B}.\overline{C}D_0 + \overline{A}.\overline{B}.CD_1 + \overline{A}B\overline{C}D_2 + \overline{A}BCD_3 + A\overline{B}.\overline{C}D_4 + A\overline{B}CD_5 + AB\overline{C}D_6 + ABCD_7$$

So sánh với biểu thức của hàm viết dưới dạng triển khai theo định lý Shanon thứ nhất

$$f(A,B,C) = \overline{A}.\overline{B}.\overline{C}f(0,0,0) + \overline{A}.\overline{B}.Cf(0,0,1) + \overline{A}B\overline{C}f(0,1,0) + \overline{A}BCf(0,1,1) + A\overline{B}.\overline{C}f(1,0,0)$$
$$+ A\overline{B}Cf(1,0,1) + AB\overline{C}f(1,1,0) + ABCf(1,1,1)$$

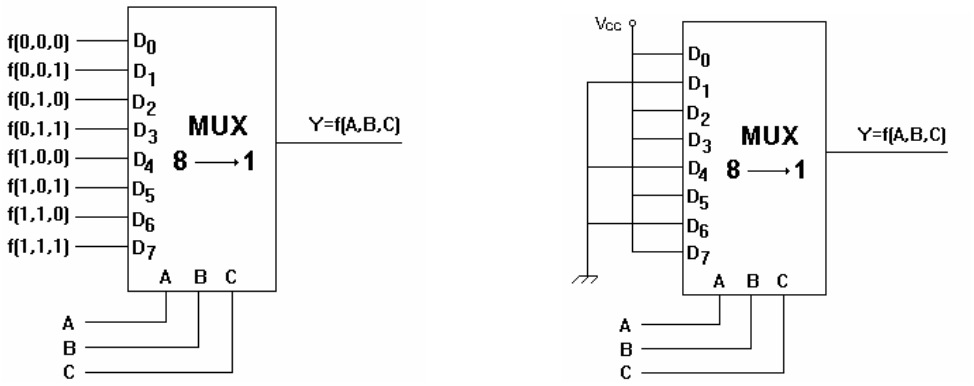
Ta được kết quả:

$D_0 = f(0,0,0)$; $D_1 = f(0,0,1)$, ... $D_6 = f(1,1,0)$ và $D_7 = f(1,1,1)$

Thí dụ: Tạo hàm:

$$Y = f(A, B, C) = \overline{A}.\overline{B}.\overline{C} + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}C + ABC$$

Ta thấy $D_0=D_2=D_3=D_5=D_7=1$ nên các ngã vào này được nối lên nguồn, các ngã vào còn lại $D_1=D_4=D_6=0$ nên được đưa xuống mass (H 4.20).



(H 4.20)

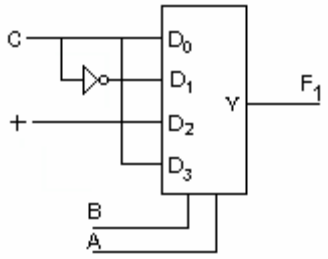
❄ Một đa hợp $2^n \rightarrow 1$ kết hợp với một cổng NOT có thể tạo hàm (n+1) biến. *Thí dụ* : Tạo hàm $F_1 = A\overline{B} + \overline{A}B\overline{C} + \overline{B}C + AC$ dùng đa hợp $4 \rightarrow 1$ và cổng NOT

Giải

Đa hợp 4 sang 1 thực hiện hàm: $Y = \overline{A}\,\overline{B}D_0 + \overline{A}BD_1 + A\overline{B}D_2 + ABD_3$

Chuẩn hóa hàm F_1 : $F_1 = \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}C + A\overline{B}\,\overline{C} + ABC$

Để $Y = F_1$ ta phải có: $D_0 = C; D_1 = \overline{C}; D_2 = 1; D_3 = C$



(H 4.21)

Trên thực tế, ta có đủ các loại mạch đa hợp từ $2 \rightarrow 1$ (IC 74157), $4 \rightarrow 1$ (IC 74153), $8 \rightarrow 1$ (IC 74151) và $16 \rightarrow 1$ (74150) . . . .

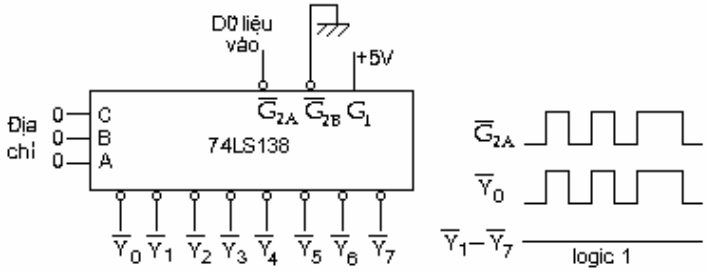
Ngoài ra, để chọn dữ liệu là các nguồn tín hiệu tương tự, ta cũng có các đa hợp tương tự với tên gọi khóa tương tự (analog switch), được chế tạo theo công nghệ MOS như IC 4051 (8 kênh) IC 4053 (2 kênh). . . . Cũng có loại khóa sử dụng được cho cả tín hiệu tương tự và số (bilateral switches) như IC 4016, IC 4066,. . mà sinh viên có thể tìm hiểu, sử dụng dễ dàng khi có bảng tra kỹ thuật.

4.3.4 Mạch giải đa hợp

Mạch giải đa hợp thực chất là mạch giải mã trong đó ngã vào cho phép trở thành ngã vào dữ liệu và ngã vào của tổ hợp số nhị phân trở thành ngã vào địa chỉ.

Trên thị trường, người ta chế tạo mạch giải mã và giải đa hợp chung trong một IC, tùy theo điều kiện mà sử dụng. Thí dụ IC 74138 là IC Giải mã 3 sang 8 đường đồng thời là mạch giải đa hợp $1 \rightarrow 8$.

Khi sử dụng IC 74138 làm mạch giải đa hợp, người ta dùng một ngã vào cho phép làm ngã vào dữ liệu và các ngã vào số nhị phân làm ngã vào địa chỉ. (H 4.22a) là IC 74138 dùng giải đa hợp với ngã vào dữ liệu là $\overline{G_{2A}}$. (H 4.22b) là dạng dữ liệu vào $\overline{G_{2A}}$ và ra ở $\overline{Y_0}$ (vì CBA=000), các ngã ra khác ($\overline{Y_1} - \overline{Y_7}$) ở mức cao.



(a) (H 4.22) (b)

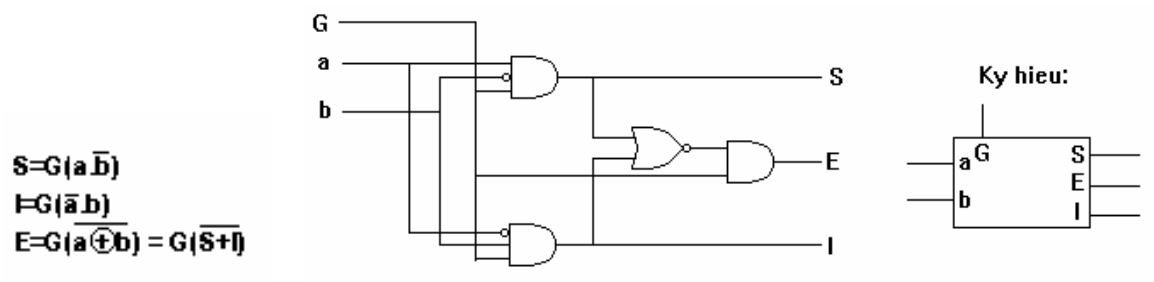
4.4 MẠCH SO SÁNH

4.4.1 Mạch so sánh 2 số 1 bit

Trước tiên ta thiết kế mạch so sánh hai số 1 bit.
Bảng sự thật của mạch so sánh một bit có ngã vào cho phép (nối mạch) G :

G	a	b	S (a>b)	I (a<b)	E (a=b)
0	x	x	0	0	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	0	0	1

Bảng 4.7



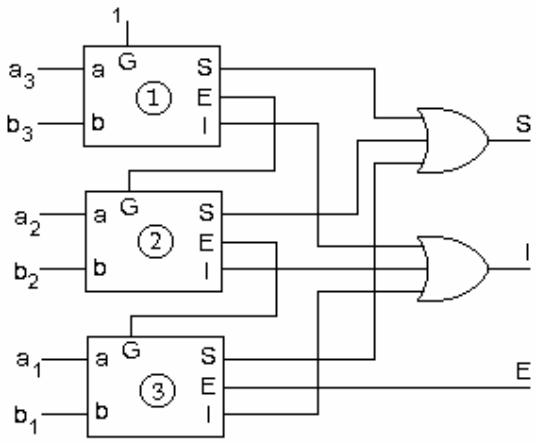
(H 4.23)

Từ mạch so sánh 1 bit ta có thể mở rộng để so sánh nhiều bit.

4.4.2 Mạch so sánh 2 số nhiều bit

Để so sánh 2 số nhiều bit, trước tiên người ta so sánh 2 bit cao nhất (MSB), kết quả lớn hoặc nhỏ hơn do 2 bit này quyết định, nếu 2 bit MSB bằng nhau người ta so sánh 2 bit có trọng số thấp hơn tiếp theo và kết quả được quyết định theo cách tương tự như ở 2 bit MSB. . . . . Sự so sánh được lặp lại cho đến bit LSB để được kết cuối cùng.

Dưới đây là sơ đồ mạch so sánh 3 bit (H 4.24).



(H 4.24)

- IC 1 so sánh 2 bit cao (a_3 & b_3) nên ngã vào cho phép được đưa lên mức cao, nếu kết quả bằng nhau, ngã ra E của nó lên cao, cho phép IC 2 so sánh, nếu kết quả lại bằng nhau, ngã ra E của IC 2 lên cao cho phép IC 3 so sánh, kết quả bằng nhau cuối cùng chỉ bởi ngã ra E của IC 3.

- Các ngã vào cổng OR nhận tín hiệu từ các ngã ra S (hoặc I) sẽ cho kết quả lớn hơn (hoặc nhỏ hơn) tùy vào kết quả so sánh ở bất cứ bit nào. Thật vậy khi có một kết quả lớn hơn (hoặc nhỏ hơn) thì S (hoặc I) ở một IC lên cao, các ngã ra E và I (hoặc S) của các IC khác bằng 0, đây là điều kiện mở cổng OR để cho kết quả so sánh xuất hiện ở một trong các cổng OR này.

Trên thị trường có sẵn loại IC so sánh 4 bit 7485 có ngã nối mạch để mở rộng việc so sánh cho số nhiều bit hơn.

Bảng sự thật của IC 7485

Trạng thái	Ngã A_3,B_3	vào A_2,B_2	so A_1,B_1	sánh A_0,B_0	Vào A'>B'	nối A'<B'	mạch A'=B'	ra A>B	A<B	A=B
1	$A_3>B_3$	x	x	x	x	x	x	1	0	0
2	$A_3<$	x	x	x	x	x	x	0	1	0
3	B_3	$A_2>B_2$	x	x	x	x	x	1	0	0
4	$A_3=$	$A_2<$	x	x	x	x	x	0	1	0
5	B_3	B_2	$A_1>B_1$	x	x	x	x	1	0	0
6	$A_3=$	$A_2=$	$A_1<$	x	x	x	x	0	1	0
7	B_3	B_2	B_1	$A_0>B_0$	x	x	x	1	0	0
8	$A_3=$	$A_2=$	$A_1=$	$A_0<$	x	x	x	0	1	0
9	B_3	B_2	B_1	B_0	0	0	1	0	0	1
10	$A_3=$	$A_2=$	$A_1=$	$A_0=$	1	0	0	1	0	0
11	B_3	B_2	B_1	B_0	0	1	0	0	1	0
	$A_3=$	$A_2=$	$A_1=$	$A_0=$						
	B_3	B_2	B_1	B_0						
	$A_3=$	$A_2=$	$A_1=$	$A_0=$						
	B_3	B_2	B_1	B_0						
	$A_3=$	$A_2=$	$A_1=$							
	B_3	B_2	B_1							
	$A_3=$	$A_2=$								
	B_3	B_2								
	$A_3=$									
	B_3									

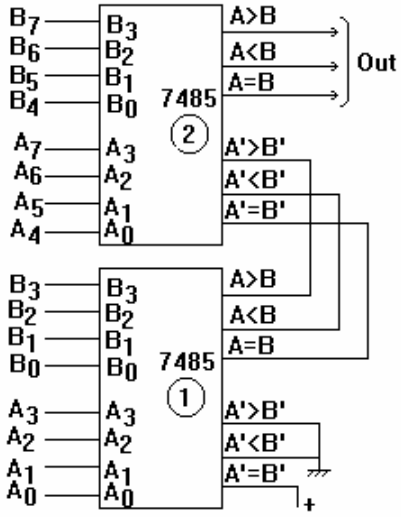
Bảng 4.8

Dựa vào bảng sự thật, ta thấy:

- Khi dùng IC 7485 để so sánh 2 số 4 bit ta phải giữ ngã vào nối mạch A'=B' ở mức cao, hai ngã vào nối mạch còn lại ở mức thấp, như vậy IC mới thể hiện được kết quả của trạng thái 9.

- Khi so sánh 2 số nhiều bit hơn ta phải dùng nhiều IC 7485 và nối ngã ra của IC so sánh bit thấp vào ngã vào nối mạch tương ứng của các IC so sánh các bit cao hơn và IC so sánh các bit thấp nhất có ngã vào nối mạch được mắc như khi dùng riêng lẻ. Để đọc được kết quả so sánh ta phải quan tâm tới các trạng thái 9, 10 và 11 trong bảng sự thật.

(H 4.25) cho ta cách mắc 2 IC 7485 để so sánh 2 số nhị phân 8 bit:



(H 4.25)

Thí dụ :

a. So sánh hai số A7 . . . .A0 = 10101111 và B7 . . . . B0 = 10110001

IC 2 so sánh các bit cao A_7 . . .A_4 = 1010 và B_7 . . .B_4 =1011 có A_7= B_7 , A_6= B_6 , A_5= B_5 và A_4<B_4 cho ngã ra A<B = 1 bất chấp trạng thái của các ngã vào nối mạch (trạng thái 8). Điều này có nghĩa là khi IC so sánh bit cao thấy có kết quả khác nhau giữa 2 số bit cao thì không quan tâm tới kết quả của bit thấp.

b. So sánh hai số A_7 . . . .A_0 = 10101111 và B_7 . . . . B_0 = 10101001

Trong trường hợp này kết quả hai số bit cao bằng nhau nên IC 2 nhìn vào ngã vào nối mạch để xem kết quả so sánh của IC1 (so sánh bit thấp), $A_3A_2A_1A_0$ =1111>$B_3B_2B_1B_0$ = 1001 nên ngã ra A>B = 1 để chỉ kết quả so sánh của 2 số 8 bit (trạng thái 10).

4.5 MẠCH KIỂM / PHÁT CHẴN LẺ

Do yêu cầu kiểm sai trong truyền dữ liệu, người ta có phương pháp kiểm tra chẵn lẻ. Trong phương pháp này, ngoài các bit dữ liệu, người ta thêm vào 1 bit kiểm tra sao cho tổng số bit 1 kể cả bit kiểm tra là số chẵn (KT chẵn) hoặc lẻ (KT lẻ)

1	0	1	1	0	0	1	1

← Bit chẵn lẻ thêm vào (KT lẻ)

1	1	0	0	1	0	1	0

← Bit chẵn lẻ thêm vào (KT chẵn)

Ở nơi thu, mạch kiểm tra chẵn lẻ sẽ kiểm tra lại số số 1 có trên tất cả các bit để biết dòng dữ liệu nhận được đúng hay sai.

Với phương pháp này máy thu sẽ có kết luận đúng khi số bit lỗi là số lẻ. Như vậy phương pháp chỉ cho kết quả đúng với xác suất 50%, tuy nhiên vì xác suất để một lỗi xảy ra là rất nhỏ nên phương pháp vẫn được sử dụng phổ biến trong một số hệ truyền thông.

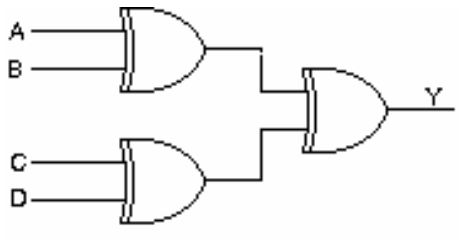
4.5.1 Mạch phát chẵn lẻ (Parity Generator, PG)

Ta sẽ xét trường hợp mạch có 4 bit dữ liệu.

Mạch có 4 ngã vào dữ liệu A, B, C, D và 1 ngã vào chọn chẵn lẻ S

- Giai đoạn 1: Thiết kế mạch ghi nhận số bit 1 là chẵn hay lẻ
Giả sử ta muốn có mạch báo kết quả Y=1 khi số bit 1 là lẻ và Y=0 khi ngược lại.
Lợi dụng tính chất của cổng EX-OR có ngã ra =1 khi số số 1 ở ngã vào là lẻ. Với 4 ngã vào, ta dùng 3 cổng EX-OR để thực hiện mạch ghi nhận này:

$$Y = (A \oplus B) \oplus (C \oplus D)$$



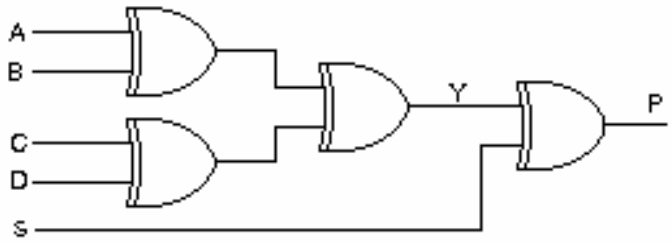
(H 4.26)

- Giai đoạn 2: Thiết kế phần mạch tạo bit chẵn lẻ P theo sự điều khiển của ngã vào S
Giả sử ta muốn có
Tổng số bit 1 của A, B, C, D, P là lẻ khi S = 1 và chẵn khi S = 0

S	Số bít 1 của ABCD	Y	P
0	Lẻ	1	1
0	Chẵn	0	0
1	Lẻ	1	0
1	Chẵn	0	1

Bảng 4.9

Bảng 4.9 cho kết quả: $P = S \oplus Y$
Vậy mạch có dạng



(H 4.27)

4.5.2 Mạch kiểm chẵn lẻ (Parity checker, PC)

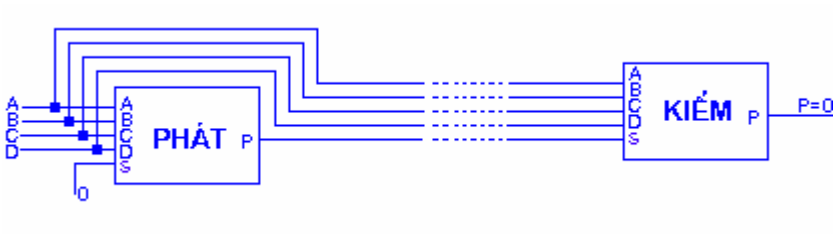
Nếu ta xem mạch phát ở (H 4.27) như là mạch có 5 ngã vào thì ngã ra P quan hệ với số lượng bit 1 ở các ngã vào đó có thể được suy ra từ bảng 4.9

Số bít 1 của ABCDS	P
Lẻ	1
Chẵn	0

Bảng 4.9

Như vậy, ta có thể dùng mạch phát ở trên để làm mạch kiểm tra chẵn lẻ.

Tóm lại, một hệ thống gồm mạch phát và kiểm chẵn lẽ được mắc như (H 4.28) Khi ngã vào S của mạch phát đưa xuống mức 0, nếu bản tin nhận đúng thì ngã ra P ở mạch kiểm cũng xuống 0.



(H 4.28)

Trên thị trường có các IC kiểm/phát chẵn lẻ như 74180 (9 bit) 74280 (9 bit), loại CMOS có 40101 (9 bit), 4531 (13 bit).

BÀI TẬP

1. Thiết kế mạch mã hóa 32 đường sang 5 đường dùng IC 74148 và cổng logic.

2. Thiết kế mạch giải mã 4 đường sang 16 đường từ mạch giải mã 2 đường sang 4 đường có ngã vào cho phép.

3. Thiết kế mạch so sánh 4 bit từ mạch so sánh 1 bit

4. Thiết kế mạch chuyển từ mã Gray sang mã nhị phân

5. Thiết kế mạch chuyển từ mã BCD sang mã Excess-3 của các số từ 0 đến 9.
(Mã Excess-3 của 1 số có được từ trị nhị phân tương ứng cộng thêm 3, thí dụ mã số 0 là 0011, mã số 9 là 1100)

6. Dùng một mạch giải mã 3 sang 8 đường, 2 cổng NAND 3 ngã vào và 1 cổng AND 2 ngã vào thực hiện các hàm sau:

$F_1 = \Sigma(1,2,3)$; $F_2 = \Sigma(4,5,7)$; $F_3 = \Sigma(1,2,3,4,5,7)$

7. Cài đặt các hàm sau dùng bộ dồn kênh (multiplexer) $4 \rightarrow 1$ (Dùng thêm cổng logic nếu cần)

$$F_1 = A\bar{B} + \bar{A}B\bar{C} + \bar{B}C + AC$$

$$F_2 = A \oplus (\bar{B}C)$$

$$F_3 = \prod(1,3,6)$$

8. Thiết kế mạch MUX $4 \rightarrow 1$ từ các MUX $2 \rightarrow 1$

9. Dùng 2 MUX $2 \rightarrow 1$ để thực hiện 1 MUX $3 \rightarrow 1$ như sau:

AB = 00 chọn C
AB = 01 chọn D
AB =1X chọn E (Trường hợp này B không xac định).

10. Thực hiện hàm Z= AB +BC + CA

- Giải mã 3 sang 8 đường (dùng thêm cổng logic nếu cần).
- Đa hợp $4 \rightarrow 1$ (dùng thêm cổng logic nếu cần).
- Hai mạch cộng bán phần và một cổng OR.