BÀI 1-1

HỆ THỐNG SỐ ĐẾM

I- KHÁI NIỆM

1- Định nghiã

Hệ thống đếm là tập hợp tất cả các ký hiệu và quy tắc để biểu diễn các số.

2- Phân loại

a) Hệ thống đếm không có vị trí

Giá trị của mỗi chữ số không phụ thuộc vào vị trí của nó nằm trong con số biểu diễn.

Ví dụ: hệ đếm La mã, số XXIII = 23 đơn vị, số XXXIX= 39 đơn vị. Chữ số X luôn bằng 10 đơn vị, không phụ thuộc vào vị trí của nó. Hệ đếm này cồng kềnh nên ít dùng.

b) Hệ thống đếm có vị trí

Giá trị của mỗi chữ số phụ thuộc vào vị trí của nó nằm trong con số biểu diễn.

-Ví dụ: hệ đếm thập phân: Số 1234 có số 4 = 4 đơn vị; số 4321 có số 4 = 4.10^3 đơn vị. Chữ số 4 luôn phụ thuộc vào vị trí của nó. Hệ đếm này gọn, dễ biểu diễn nên được dùng phổ biến.

- Hệ thống đếm có vị trí, gồm: hệ thập phân, hệ nhị phân, hệ bát phân, ...

II- CÁC HỆ ĐẾM THEO VỊ TRÍ

Một số đếm N bất kỳ, có thể gồm cả phần nguyên (N_I) và phần phân (N_F):

$$N = N_I + N_F \quad (1)$$

Số N được biểu diễn dạng:

$$N_{(B)} = a_{n-1}\, a_{n-2} \ \dots \ a_1 a_0 \,,\, a_{-1} a_{-2} \ \dots \ a_{-m} \quad (2)$$

Trong đó:

N : số đếm bất kỳ của mọi hệ đếm
B : cơ số của hệ đếm (số ký tự phân biệt)
n : số chữ số trong phần nguyên
m : số chữ số trong phần phân
a_{n-1} : chữ số có nghĩa lớn nhất
a_{-m} : chữ số có nghĩa nhỏ nhất

- Tổng quát có thể biểu diễn số N theo cơ số B dới dạng đa thức sau:

$$N_{(B)} = a_{n-1} B_{n-1} + a_{n-2} B_{n-2} + \dots + a_1 B_1 + a_0 B_0 + a_{-1} B_{-1} + a_{-2} B_{-2} + \dots + a_{-m} B_{-m} \quad (3)$$

hoặc: $N_{(B)} = \Sigma a_i B_i \quad (4)$

Trong đó: $i = 0 \div (n-1)$, tương ứng với phần nguyên
$i = (-1) \div (-m)$, tương ứng với phần phân
và $0 \le a_i \le B - 1 \quad (i = 0, 1, \dots, n-1)$.

1- Hệ thập phân (Decimal)

- Cơ số B =10, dùng 10 chữ số từ (0÷9) để biểu diễn mọi số.
- Mỗi vị trí số bất kỳ đều có trọng số gấp 10 lần số có vị trí bên phải kề nó.
- Dạng tổng quát: $N_{(10)} = \Sigma a_i 10^i$ (5)
- Ví dụ: $143,75(10) = 1.10^2 + 4.10^1 + 3.10^0 + 7.10^{-1} + 5.10^{-2}$
- Hệ thập phân được dùng phổ biến trong đời sống sinh hoạt.

2- Hệ nhị phân (Binary)

- Cơ số B = 2, dùng 2 chữ số 0 và 1 để biểu diễn các số.
- Mỗi vị trí số (còn gọi là bít: Binary Digit) chỉ lấy gía trị 0 hoặc 1.
- Dạng tổng quát: $N_{(2)} = \Sigma a_i 2^i$ (6)

Ví dụ: $1011,11(2) = 1.2^3 + 0.2^2 + 1.2^1 + 1.2^0 + 1.2^{-1} + 1.2^{-2}$

- Hệ nhị phân được dùng rộng rãi trong các mạch số.

3- Hệ bát phân (Octal)

- Cơ số B = 8, dùng 8 chữ số từ (0÷7) để biểu diễn.
- Mỗi vị số có 8 mã số (a_i= 0÷7).
- Dạng tổng quát: $N_{(8)} = \Sigma a_i 8^i$ (7)

Ví dụ: $37,41_{(8)} = 3.8^1 + 7.8^0 + 4.8^{-1} + 1.8^{-2}$

- Hệ đếm cơ số 8 gọn hơn hệ nhị phân nên thường dùng nhiều trong kỹ thuật máy tính.

4- Hệ thập lục phân (Hex: Hexadecimal)

- Cơ số B = 16, mỗi vị số có 16 mã số: 0 ÷9, A (10), B (11), C (12), D (13), E (14), F (15).
- Dạng tổng quát: $N_{(16)} = \Sigma a=16^i$ (8)

Ví dụ: $2A,7F_{(16)} = 2.16^1 + 10.16^0 + 7.16^{-1} + 15.16^{-2}$

- Hệ đếm cơ số 16 gọn hơn hệ cơ số 2 và 8 nên được dùng phổ biến trong kỹ thuật máy tính.

III- CHUYỂN ĐỔI GIỮA CÁC HỆ ĐẾM

1- Hệ nhị phân và hệ thập phân

a) Từ hệ 2 sang hệ 10

- Viết số nhị phân dưới dạng đa thức (3), cộng tất cả các số hạng theo giá trị số thập phân. Kết quả cộng được sẽ là dạng thập phân của số nhị phân đã cho.

Ví dụ: $1011,01_{(2)} = 1.2^3 + 0.2^2 + 1.2^1 + 1.2^0 + 0.2^{-1} + 1.2^{-2}$

$= 8 + 0 + 2 + 1 + 0 + 0,25$

$= 11,25_{(10)}$ (9)

b) Từ hệ 10 sang hệ 2

- Phần nguyên

+ Lấy phần nguyên chia cho 2, ghi lại số dư vừa chia, được bít nhỏ nhất.

+ Lấy kết quả của phép chia ở trên, tiếp tục chia cho 2, ghi lại số d vừa chia, được bít tiếp theo.

+ Tiếp tục làm như trên cho đến khi kết quả nhỏ hơn 2 (không chia hết cho 2) thì dừng, được bít lớn nhất.

- Phần phân

+ Lấy phần phân nhân với 2. Nếu kết quả lớn hơn hoặc bằng1, ghi lại số 1. Nếu kết quả nhỏ hơn 1, ghi lại số 0, được bít lớn nhất.

+ Lấy phần phân của phép nhân ở trên, tiếp tục nhân với 2 và làm nh trên cho đến khi đạt độ chính xác cần thiết thì dừng lại. Ghi lại số nguyên, được bít nhỏ nhất.

Ví dụ: $13,6875_{(10)} = 1101,1011_{(2)}$

Phần nguyên: 1101

$13 : 2 = 6$, dư 1 $\Rightarrow$ được $a_0 = 1$

$6 : 2 = 3$, dư 0 $\Rightarrow$ được $a_1 = 0$

$3 : 2 = 1$, dư 1 $\Rightarrow$ được $a_2 = 1$

$1 : 2 = 0$, dư 1 $\Rightarrow$ được $a_3 = 1$

Phần phân: 1011

$0,6875 \times 2 = 1,375 = 1 + 0,375 \Rightarrow$ được $a_{-1} = 1$

$0,375 \times 2 = 0,75 = 0 + 0,75 \Rightarrow$ được $a_{-2} = 0$

$0,75 \times 2 = 1,5 = 1 + 0,5 \Rightarrow$ được $a_{-3} = 1$

$0,5 \times 2 = 1 = 1 + 0 \Rightarrow$ được $a_{-4} = 1$

2- Hệ nhị phân và hệ bát phân

a) Từ hệ 2 sang hệ 8

- Vì $2^3 = 8$, nên mỗi vị trí số trong hệ 8 tương ứng một nhóm 3 bít của hệ 2.
- Khi đổi: chia phần nguyên của hệ 2 thành từng nhóm 3 bít bắt đầu từ bít 2^0 và phần phân bắt đầu từ bít 2^{-1}.
- Dùng 8 chữ số của hệ 8 thay cho 8 chữ số tương ứng của nhóm 3 bít.

Ví dụ: $10110101,00111101_{(2)} = 265,172_{(8)}$

Ví dụ: 10110101,00111101(2) = 265,172(8)

Chia nhóm : 010 110 101 , 001 111 010 $_{(2)}$

Kết quả : 2 6 5 , 1 7 2 $_{(8)}$

b) Từ hệ 8 sang hệ 2

- Thay một chữ số trong hệ 8 bằng nhóm 3 bít của hệ 2.
- Ví dụ: 5 1 2 , 3 0 4 $_{(8)}$

= 101 001 010 , 011 000 100 $_{(2)}$

3- Hệ nhị phân và hệ thập lục phân

a) Từ hệ 2 sang hệ 16

- Vì 24 = 16, nên mỗi vị số của hệ 16 tương ứng với một nhóm 4 bít của hệ 2.
- Khi đổi: chia phần nguyên của hệ 2 thành từng nhóm 4 bít bắt đầu từ bít 2^0 và phần phân bắt đầu từ bít 2^{-1}.

Ví dụ: 10011011011,1001111(2) = 4DB,9E(16)

+ Chia nhóm: 0100 1101 1011 , 1001 1110 $_{(2)}$

+ Kết quả: 4 D B , 9 E $_{(16)}$

b) Từ hệ 16 sang hệ 2

- Thay một chữ số trong hệ 16 bằng nhóm 4 bít của hệ 2.
- Ví dụ: 7 F A , C 6 $_{(16)}$

Kết quả : 0111 1111 1010 , 1100 0110 $_{(2)}$

hoặc : 111 1111 1010 , 1100 011 $_{(2)}$

IV- BIỂU DIỄN SỐ TRONG CÁC HỆ ĐẾM

1- Về hệ đếm

2- Về dấu phẩy

3- Về dấu

BÀI TẬP

1- Đổi các số sau:

a) $1011010,101_{(2)}$ sang hệ số 10; 8 ; 16.

b) $678,42_{(10)}$ sang hệ 2; 8; 16.

c) $734,25_{(8)}$ sang hệ 2; 10; 16.

d) $8AFD,C4A_{(16)}$ sang hệ 2; 8; 10.

2- Làm tất cả các bài tập trong tài liệu.

BÀI 1-2: CÁC LOẠI M ·

I- KHÁI NIỆM

1- Đặt vấn đề

- Trong các hệ thống điện tử số (máy tính, điện thoại số..), dữ liệu đợc truyền hay xử lý ở dạng nhị phân gồm các bit 0 và 1. Vì vậy phải biến đổi các chữ cái, chữ số, ký tự đặc biệt,.. thành dạng nhị phân.

- Việc biến đổi các chữ cái, chữ số,.. (gọi chung là các phần tử mang tin) thành số nhị phân nhw trên gọi là quá trình mã hóa.

- Một số nhị phân n bit có thể biểu diễn cho 2^n phần tử tin khác nhau với giá trị thập phân từ: $0 \div 2^{n-1}$. Các số nhị phân (hay một nhóm) n bit đó gọi là mã (code) của phần tử tin tức.

2- Phân loại

Có nhiều loại mã đợc tạo ra để thực hiện các mục đích và nhiệm vụ khác nhau.Thường chia thành ba nhóm cơ bản:

+ Các loại mã dùng để mã hóa các ký tự số;
+ Các loại mã dùng để mã hóa các ký tự khác;
+ Các loại mã phát hiện và sửa sai;

Ta chỉ xét một số mã thông dụng

II- C¸c lo¹i m·

1- Mã số

- Thường dùng các loại mã: nhị phân; nhị - thập phân (BCD: Binariy Coded Decimal); thừa 3; Gray; 7 đoạn;...

- Ngoài ra còn chia thành hai loại: có trọng số (trọng số của các ký hiệu nhị phân phụ thuộc vào vị trí của chúng trong từ mã) và không có trọng số (trọng số của các ký hiệu nhị phân không phụ thuộc vào vị trí của chúng trong từ mã) .

- Bảng 1 là một số loại mã số điển hình.

a) Mã nhị phân

Dùng số nhị phân n bit để biểu diễn các số thập phân, ví dụ: số nhị phân 4 bit có các từ mã 0000 ÷ 1111 biểu diễn số thập phân từ 0 ÷ 15. Có trọng số sắp xếp từ thấp đến cao (tính từ phải sang trái) là: 8, 4, 2, 1.

b) Mã BCD (Binary Coded Decimal)

- Dùng từ mã nhị phân có độ dài 4 bit để mã hóa cho 10 chữ số thập phân.

- Tùy theo cách sử dụng 10 trên 16 tổ hợp mã nhị phân 4 bit mà ta có các loại mã BCD khác nhau.

- Một số mã BCD thường gặp:

+ BCD - Norman (NBCD) là mã BCD đơn trị và có trọng số (8,4,2,1) nên còn gọi là mã BCD 8421.

+ Trong kỹ thuật còn sử dụng mã BCD có trọng số khác, như: 2421, 5121, 5421, 7421,...

Số thập phân	Mã nhị phân	Mã BCD	Mã thừa 3	Mã Gray	Gray d 3	Mã 7 đoạn abcdefg
0	0000	0000	0011	0000	0010	1111110
1	0001	0001	0100	0001	0110	0110000
2	0010	0010	0101	0011	0111	1101101
3	0011	0011	0110	0010	0101	1111001
4	0100	0100	0111	0110	0100	0110011
5	0101	0101	1000	0111	1100	1011011
6	0110	0110	1001	0101	1101	1011111
7	0111	0111	1010	0100	1111	1110000
8	1000	1000	1011	1100	1110	1111111
9	1001	1001	1100	1101	1010	1111101
10	1010			1111	1011	
11	1011			1110	1001	
12	1100			1010	1000	
13	1101			1011	0000	
14	1110			1001	0001	
15	1111			1000	0011	

c) Mã dư 3 (thừa 3)

Được tạo ra từ mã nhị phân bằng cách cộng thêm 3 đơn vị (tức 0011) vào từ mã BCD 8421 tương ứng.

d) Mã Gray

+Là loại không có trọng số, các từ mã kề cận nhau chỉ khác nhau ở một biến số.

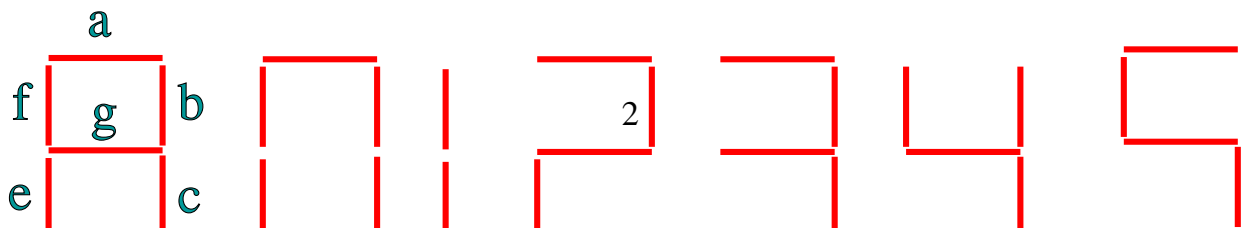
+ Được suy ra từ mã BCD8421, kể từ 0÷1 là giống nhau, từ 2 ÷9 ở BCD 8421 cứ số nào đứng bên phải số 1 khi sang Gray phải đổi sang 0 (ngược lại).

d) Mã Gray dư 3

Được tạo ra từ mã Gray bằng cách lệch đi 3 hàng. Vì vậy nó có đặc điểm giống nh mã Gray.

e) Mã 7 đoạn

- Độ dài từ mã là 7 bit, thường dùng để điều khiển 7 thanh phát sáng (điôt phát sáng: LED) còn gọi là đèn LED 7 đoạn.
- Đèn LED được cấu tạo tơng ứng với các số thập phân như hình 1.
- Mã 7 đoạn thường dùng trong kỹ thuật đo lường, để chỉ thị các phép đo.



Hình 1: Đèn LED 7 đoạn và các số tương ứng

2- Mã ký tự

Ngoài các loại mã chữ số còn các mã chữ cái và kí tự đặc biệt (các loại dấu, kí tự đồ họa ...). Hiện nay dùng phổ biến hai loại mã sau

a) Mã ASCII

+ Mã ASCII (American Standard Code for Information Interchange): mã trao đổi thông tin theo tiêu chuẩn Mỹ, dùng 8 bit để mã hóa cho một ký tự, trong đó 7 bit biểu thị các tin tức, bít thứ 8 là bit kiểm tra (parity) chẵn lẻ để phát hiện và sửa lỗi khi truyền tin.

+ Mã ASCII đƯợc dùng phổ biến trong kỹ thuật máy tính và các hệ thống thông tin số.

+ Ví dụ: mã ASCII cho chữ cái và một số kí tự đặc biệt như bảng 2.

Kí tự	Mã ASCII	Mã EBCDIC
A	100 0001	
B	100 0010	
C	100 0011	
D	100 0100	
0	011 0000	
1	011 0001	
2	011 0010	
$	010 0100	
*	010 1010	

b) Mã EBCDIC

- EBCDIC (Extended Binary Coded Decimal Interchange Code): mã trao đổi nhị thập phân mở rộng, dùng 8 bit để mã hóa cho một ký tự, do đó có nhiều biểu tượng và đặc trưng hơn mã ASCII.

- EBCDI được dùng phổ biến trong các hệ thống số có kích thước lớn.

3- Mã sửa sai

- Ngoài các bit mang thông tin còn có một số bit được thêm vào để phát hiện và sửa sai.

- Đơn giản nhất là mã chẵn lẻ, khi đó bit thêm vào gọi là bit chẵn lẻ (Parity bit).

- Mã sửa sai được dùng phổ biến trong kỹ thuật thông tin.

Mã ASCI và EBCDIC của một số ki hiệu, biểu tượng

Kí hiệu, biểu tượng	ASCII	EBCDIC
Spase	010 0000	0100 000
!	010 0001	0101 1010
#	010 0011	0111 1011
$	010 0100	0101 1011
&	010 0110	0101 0000
A	100 0001	1100 0001
J	100 1010	1101 0001
Q	101 0001	1101 1000
Z	101 1010	1110 1001

BÀI 1-3:

CÁC PHÉP TÍNH SỐ HỌC

I- CÁC PHÉP TÍNH TRONG HỆ ĐẾM HAI

1- Phép cộng

- Cộng hai số có một chữ số:

0 + 0 = 0

0 + 1 = 1 + 0 = 1

1 + 1 = 0, nhớ 1. Số nhớ được cộng chuyển tiếp sang số có giá trị cao hơn kế tiếp.

- Cộng 2 số có nhiều chữ số:

+ Cách tiến hành:

Ghi kết quả cộng không có nhớ ở bên dưới hai số hạng cần cộng với nhau (1).

ở dòng tiếp theo (2), ghi các số nhớ (đã được chuyển lên hàng trên theo bảng cộng).

Cộng kết quả (1) và (2) theo đúng thứ tự của các chữ số trên từng hàng ta có tổng cuối cùng.

2-Phép trừ

- Suy ra trực tiếp từ phép cộng:

0 - 0 = 0 (a)

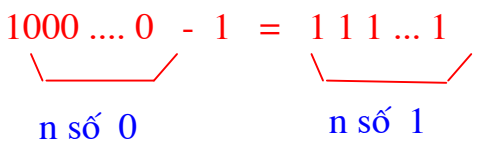
1 - 0 = 1 (b)

1 - 1 = 0 (c)

0 - 1 = 0 , nhớ 1 (d)

Để thực hiện (0-1), phải mượn 1 ở cột cao hơn, nên phải nhớ 1 để trừ tiếp vào cột kế sau.

Chú ý: nếu lấy một số có chữ số 1 đầu tiên và sau nó là n chữ số 0 trừ đi 1 thì kết quả là một số có n chữ số 1. Ví dụ:



- Trừ hai số nhiều hàng:

+ Nếu trừ số 0 cho số 1 thì phải mượn 1 từ hàng trước và tiến hành như biểu thức (d). Việc mượn số hoàn toàn tương tự như phép trừ ở số thập phân.

+ Ví dụ:

Số trừ	1 1 1 1 0 0 1 1
Số bị trừ	1 0 1 1 0 1 1 0
Kết quả	0 0 0 0 0 0 0 1

3- Phép nhân

- Nhân hai số 1 bit:

0 x 0 = 0

0 x 1 = 1 x 0 = 0

1 x 1 = 1

- Nhân hai số nhiều bit:

+ Cách tiến hành : thực hiện nhân liên tiếp như ở hệ thập phân.

+ Ví dụ:

- Trường hợp số nhân, số bị nhân hoặc cả hai đều có phần nguyên và phần thập phân thì cách nhân hoàn toàn tương tự nh ở hệ thập phân.

Ví dụ:

4-Phép chia

Phép chia trong hệ hai cũng thực hiện tương tự như phép chia trong hệ 10 và có hai trường hợp: số bị chia lớn hơn số chia và số bị chia nhỏ hơn số chia. .

a) Số bị chia lớn hơn số chia

- Ví dụ: chia số 11010111 cho số 10110 , kết quả cho phép lấy 2 số ở phần phân

b) Số bị chia nhỏ hơn số chia

II- CỘNG SỐ BCD

1- Khi tổng ≤ 9

- Ví dụ:

```
   0 1 0 1      (5)     Cộng bình thường
 + 0 1 0 0      (4)
 -------------
   1 0 0 1      (9)
```

2- Khi tổng > 9

- Ví dụ:

```
   0 1 0 1      (5)
 + 0 1 1 1      (7)
 -------------
   1 1 0 0      (12)    Không hợp lệ
 + 0 1 1 0      (6)     Cộng thêm 6 đơn vị
 -------------
 0 0 0 1 0 0 1 0  (12)  BCD đúng
```

III- CỘNG - TRỪ SỐ HEXA

1- Cộng số Hexa

- Nếu tổng ≤ 15, biểu diễn binh thường;

- Nếu tổng ≥ 16, trừ đi 16 và nhớ 1 đến vị trí kế tiếp.

2 - Trừ số Hexa (TL).

Lấy số bù 2 của số trừ rồi cộng với số bị trừ.

* Lấy số bù 2 của số hexa:

- PP1: đổi số hexa thành số nhị phân, rồi lấy số bù 2, sau đó đổi lại số hexa;

- PP2: lấy F trừ đi mỗi kí số, sau đó cộng thêm 1.

* Thực hiện phép trừ số hexa:

BÀI 2-1: CƠ SỞ ĐẠI SỐ LOGIC

I- KHÁI NIỆM

1- Đại số logic

- Trong thực tế luôn tồn tại hai khái niệm thống nhất nhng đối lập nhau. Ví dụ:

Đúng - Sai
Thật - Giả
Ngày - §ªm ... (1)

hoÆc hai tr¹ng th¸i lµm viÖc cña transistor:

Th«ng - T¾t (2)

C¸c quan hÖ (1) vµ (2) ®-îc gäi lµ quan hÖ logic.

- C«ng cô to¸n häc dïng ®Ó m« t¶ c¸c quan hÖ logic (1),(2) nªu trªn gäi lµ ®¹i sè logic hay ®¹i sè BOOLE (do George Boole ®Ò xíng vµo g÷a thÕ kû 19).
- §¹i sè logic chØ dïng hai ch÷ sè 0 vµ 1 ®Ó biÓu diÔn c¸c quan hÖ (1), (2).
- §¹i sè logic lµ c«ng cô to¸n häc ®Ó ph©n tÝch vµ thiÕt kÕ m¹ch sè.

II- BIẾN VÀ HÀM LOGIC

§Ó m« t¶ (diÔn gi¶i) c¸c qu¸ trxnh (tr¹ng th¸i) logic, ngêi ta dïng biÕn vµ hµm logic

1- Biến logic

+ Cho tËp B = 0, 1 , X_i gäi lµ biÕn logic nÕu X_i thuéc B.
+ KÝ hiÖu: A, B, C,... hoÆc X_1, X_2, X_3.
+ BiÕn logic chØ nhËn mét trong 2 gi¸ trþ 0 hoÆc 1.

2- Hàm logic

+ Mét hµm phô thuéc c¸c biÕn logic, gäi lµ hµm logic.
+ KÝ hiÖu: f(A, B, C,...) hoÆc f (X_1, X_2, X_3)
+ Hµm logic còng chØ nhËn mét trong 2 gi¸ trþ 0 hoÆc 1
+ Hµm ®¬n gi¶n: gåm 1 ®Õn 2 biÕn, gäi lµ hµm logic c¬ b¶n.
+ Hµn n biÕn: f (X_1, X_2, X_3,..., X_n), gäi lµ hµm phøc t¹p
+ Mäi hµm phøc t¹p ®Òu cã thÓ biÓu diÔn b»ng c¸c hµm ®¬n gi¶n.

Tãm l¹i : trong ®¹i sè logic thx biÕn vµ hµm logic ®Òu chØ lÊy gi¸ trþ 1 hoÆc 0.

III- CÁC CÔNG THỨC VÀ ĐỊNH LÝ TRONG ĐẠI SỐ LOGIC

1- Quan hệ giữa các hằng số

0 . 0 = 0	0 + 0 = 0	$\overline{0} = 1$
0 . 1 = 0	0 + 1 = 1	$\overline{1} = 0$
1 . 1 = 1	1 + 1 = 1	

2- Quan hệ giữa biến số và hằng số

X . 1 = X	X + 1 = 1
X . 0 = 0	X + 0 = X
X . $\overline{X}$ = 0	X + $\overline{X}$ = 1

3- Các công thức tương tự đại số thường

- LuËt giao ho¸n: $X_1 . X_2 = X_2 . X_1$
- LuËt kÕt hîp: $X_1 + X_2 = X_2 + X_1$

$(X_1 + X_2) + X_3 = X_1 + (X_2 + X_3)$

- LuËt ph©n phèi: $X_1 . (X_2 + X_3) = X_1 . X2 + X_1 . X_2$

$X_1 + X_2 . X_3 = (X_1 + X_2) . (X_1 + X_3)$

4- Các công thức đặc thù chỉ có trong đại số logic

- LuËt ®ång nhÊt: $X . X = X$ $X + X = X$
- §Þnh lÝ Demorgan: $\overline{X_1 . X_2} = \overline{X_1} + \overline{X_2}$

$\overline{X_1 + X_2} = \overline{X_1} . \overline{X_2}$

- LuËt hoµn nguyªn: $\overline{\overline{X}} = X$

5- Một số công thức thường dùng khác

$X_1 . X_2 + X_1 . \overline{X_2} = X_1$

$X_1 + X_1 \overline{X_2} = X_1$

$X_1 + \overline{X_1} X_2 = X_1 + X_2$

IV- CÁC PHÉP TOÁN VÀ CỔNG LOGIC

1- Các phép toán và cổng logic cơ bản

- Cã 3 quan hÖ logic c¬ b¶n: VÀ, HOẶC, phñ ®Þnh
- C¸c biÓu thøc to¸n häc m« t¶ c¸c quan hÖ logic nªu trªn gäi lµ c¸c phÐp to¸n logic c¬ b¶n.

T-¬ng ®ương lµ 3 phÐp to¸n logic c¬ b¶n: nh©n logic (VÀ) ; céng logic (HOẶC) ; ®¶o logic (PHỦ ĐỊNH).

- M¹ch ®iÖn t-¬ng ®ương thùc hiÖn c¸c phÐp to¸n logic c¬ b¶n gäi lµ cæng logic c¬ b¶n AND, OR, NOT.

a- Nhân logic (AND)

- BiÓu thøc: $y= x_1. x_2 . ... x_n$

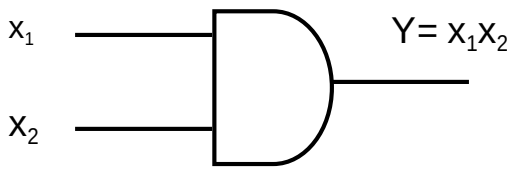
y= 1 khi mäi x_i= 1

y= 0 khi cã mét x_i= 0

- VÝ dô víi n=2
- B¶ng ch©n lý:

x_1	x_2	Y
0	0	0
0	1	0
1	0	0
1	1	1

- Ký hiÖu :



b- Cộng logic (OR: HOẶC)

- BiÓu thøc: $y= x_1 + x_2 + ...+ x_n$

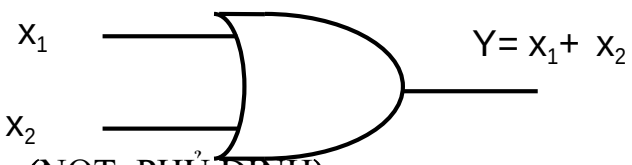
y= 1, khi cã mét x_i= 1

y= 0, khi mäi x_i= 0

- VÝ dô víi n=2
- B¶ng ch©n lý:

x_1	x_2	Y
0	0	0
0	1	1
1	0	1
1	1	1

- Ký hiÖu :



c- Đảo logic (NOT: PHỦ ĐỊNH)

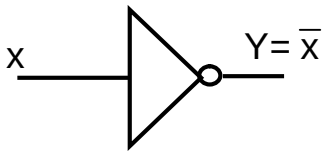
- Biểu thức: y = x

y= 0, khi chỉ một x= 1

y= 1, khi mọi x = 0

- Bảng chân lý:
- Ký hiệu :

x	y
0	1
1	0



2- Các phép toán và cổng logic khác

a) Nhân - phủ định (NAND)

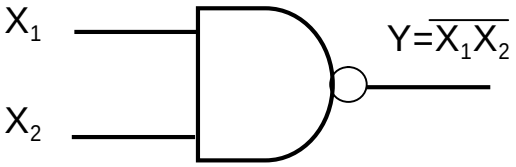
- Biểu thức: $y = \overline{x_1 x_2 ... x_n}$

y = 1 khi chỉ một $x_i = 0$

y = 0 khi mọi $x_i = 1$

- Ví dụ với n =2
- Bảng chân lý:
- Ký hiệu :

x_1	x_2	Y
0	0	1
0	1	1
1	0	1
1	1	0



b) Cộng - phủ định (NOR)

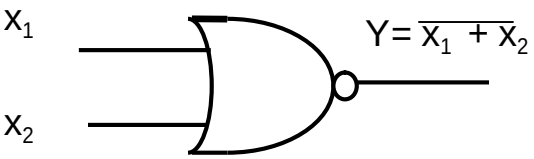
- Biểu thức: $y = \overline{x_1 + x_2 + ... + x_n}$

y = 1 khi chỉ mọi $x_i = 0$

y = 0 khi một $x_i = 1$

- Ví dụ với n =2
- Bảng chân lý:
- Ký hiệu :

x_1	x_2	Y
0	0	0
0	1	0
1	0	0
1	1	1



c) Cộng mô đun 2 (XOR)

- Biểu thức: $y = x_1 \oplus x_2 = x_1\overline{x}_2 + \overline{x}_1 x_2$

y= 1 khi $x_1 = x_2$

y= 0 khi $x_1 \neq x_2$

- Ví dụ : với n = 2

BÀI 2-2: PHƯƠNG PHÁP BIỂU DIỄN HÀM LOGIC

ĐẶT VẤN ĐỀ

Khi nghiªn cøu vµ xö lý nh÷ng vÊn ®Ò logic, ta cã thÓ dïng c¸c ph-¬ng ph¸p kh¸c nhau ®Ó biÓu diÔn hµm logic tïy theo ®Æc ®iÓm cña hµm logic cÇn xÐt.

Th-êng dïng c¸c ph-¬ng ph¸p: B¶ng ch©n lý; biÓu thøc logic; b¤a C¸c - n©u...

I- BẢNG CHÂN LÝ

M« t¶ quan hÖ gi÷a c¸c gi¸ trÞ cña hµm sè t-¬ng øng víi mäi gi¸ trÞ cã thÓ cã cña biÕn sè d-íi d¹ng b¶ng.

1- Cách lập bảng

X¸c ®Þnh sè biÕn vµ tæ hîp biÕn: mçi biÕn cã thÓ lÊy mét trong hai gi¸ trÞ 1 hoÆc 0, nÕu cã n biÕn th× sÏ cã 2n tæ hîp c¸c gi¸ trÞ kh¸c nhau cña chóng.

- LiÖt kª tÊt c¶ c¸c tæ hîp gi¸ trÞ cña biÕn (th-êng s¾p xÕp theo tuÇn tù sè ®Õm nhÞ ph©n).

- Thay gi¸ trÞ cña mçi tæ hîp biÕn vµo hµm sè vµ tÝnh ra gi¸ trÞ t-¬ng øng cña hµm, rải liÖt kª thµnh b¶ng.

-VÝ dô: LËp b¶ng ch©n lý cho hµm sè

$f(x_3,x_2,x_1) = x_1x_2 + x_2x_3 + x_3x_1$

+ Hµm cã 3 biÕn, nªn cã $2^3 = 8$ tæ hîp c¸c gi¸ trÞ cña biÕn

+ Thay gi¸ trÞ cña c¸c tæ hîp biÕn vµo hµm sè vµ tÝnh gi¸ trÞ cña hµm, ta cã b¶ng 1:

	X_3	X_2	X_1	Y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	x
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	x
7	1	1	1	1

Chó ý: øng víi tæ hîp biÕn nhng hµm kh«ng x¸c ®Þnh, ký hiÖu x, gäi lµ b¶ng khuyÕt. Khi ®ã cã thÓ chän tïy ý: x =1 hoÆc x = 0, ý nghÜa cña hµm kh«ng thay ®æi.

2- Đặc điểm

- ¦ u ®iÓm:

+ Trùc quan, khã nhÇm lÉn (trong c¸c sæ tay IC sè ®Òu cã b¶ng chøc n¨ng t-¬ng øng víi b¶ng ch©n lý ®Ó m« t¶ chøc n¨ng logic cña IC).

+ TiÖn lîi khi g¶i quyÕt mét nhiÖm vô thùc tÕ ë d¹ng vÊn ®Ò logic (trong thiÕt kÕ m¹ch sè th× ®Çu tiªn lµ kª ra b¶ng ch©n lý).

- Nh-îc ®iÓm: Cảng kÒnh, phøc t¹p khi sè biÕn lín. Kh«ng thÓ dïng c¸c c«ng thøc vµ ®Þnh lý cña ®¹i sè logic ®Ó tÝnh to¸n.

II- BIỂU THỨC LOGIC

Dïng c¸c phÐp to¸n AND, OR, NOT,...®Ó biÓu thÞ quan hÖ logic gi÷a c¸c biÕn trong hµm.

Cã hai d¹ng biÓu diÔn hµm n biÕn, ®ã lµ: chuÈn t¾c tuyÓn (CTT) vµ chuÈn t¾c héi (CTH).

1- Dạng CTT (tổng các tích)

- C¸ch lËp biÓu thøc:

+ Chỉ quan t©m ®Õn c¸c tæ hîp biÕn mµ hµm cã gi¸ trÞ b»ng 1. Sè lÇn hµm cã gi¸ trÞ 1 chÝnh lµ sè tÝch cña biÓu thøc.

+ Trong mçi tÝch, c¸c biÕn cã gi¸ trÞ 1 viÕt nguyªn biÕn, c¸c biÕn cã gi¸ trÞ 0 viÕt ®¶o biÕn.

+ BiÓu thøc cña hµm f b»ng tæng c¸c tÝch ®ã.

- VÝ dô: xÐt l¹i vÝ dô trong b¶ng 1.

+ Hµm f = 1 t¹i c¸c tæ hîp biÕn øng víi gi¸ trÞ thËp ph©n lµ 3, 5, 7 vµ c¸c tÝch lµ:

$m_3 = \overline{x}_3 x_2 x_1$

$m_5 = x_3 \overline{x}_2 x_1$

$m_7 = x_3 x_2 \overline{x}_1$

+ D¹ng CTT lµ : $f = \overline{x}_3 x_2 x_1 + x_3 \overline{x}_2 x_1 + x_3 x_2 \overline{x}_1$

- Khi cho gi¸ trÞ cña hµm logic díi d¹ng CTT, øng víi c¸c gi¸ trÞ f =1, gäi lµ c¸c Mintec (sè h¹ng nhá nhÊt), ký hiÖu m_i, víi i lµ sè thËp ph©n øng víi f_i =1.

Khi ®ã d¹ng CTT ®-îc viÕt lµ:

$f(x_3 x_2 x_1) = m_3 + m_5 + m_7 = \Sigma(3,5,7)$; N=2,6.

Trong ®ã: 3, 5, 7 lµ sè thËp ph©n cña c¸c tæ hîp biÕn øng víi f =1; N =2, 6 lµ sè thËp ph©n cña tæ hîp biÕn øng víi f = x (kh«ng x¸c ®Þnh).

2- Dạng CTH (tích các tổng)

- C¸ch lËp biÓu thøc:

+ Chỉ quan t©m ®Õn c¸c tæ hîp biÕn mµ hµm cã gi¸ trÞ b»ng 0. Sè lÇn hµm cã gi¸ trÞ 0 chÝnh lµ sè tæng cña biÓu thøc.

+ Trong mçi tæng, c¸c biÕn cã gi¸ trÞ 0 viÕt nguyªn biÕn, c¸c biÕn cã gi¸ trÞ 1 viÕt ®¶o biÕn.

+ BiÓu thøc cña hµm f b»ng tÝch c¸c tæng ®ã.

- VÝ dô: Ta lÊy l¹i vÝ dô trong b¶ng 1.

+ Hµm f = 0 t¹i c¸c tæ hîp biÕn øng víi gi¸ trÞ thËp ph©n lµ 0, 1, 4 vµ c¸c tæng ®-îc m« t¶:

$M_0 = x_3 + x_2 + x_1$

$M_1 = x_3 + x_2 + \overline{x}_1$

$M_4 = \overline{x}_3 + x_2 + x_1$

+ D¹ng CTH lµ : $f = (x_3 + x_2 + x_1)(x_3 + x_2 + \overline{x}_1)(\overline{x}_3 + x_2 + x_1)$

- §Ó ®¬n gi¶n khi cho gi¸ trÞ cña hµm logic d-íi d¹ng CTH, øng víi c¸c gi¸ trÞ f = 0, gäi ®ã lµ c¸c Maxtec (sè h¹ng lín nhÊt), ký hiÖu M_i, víi i lµ sè thËp ph©n øng víi $f_i = 0$.

Khi ®ã d¹ng CTH ®-îc viÕt lµ:

$$f(x_3x_2 x_1) = M_0 . M_1 . M_4 = \Pi(0,1,4); N=2,6.$$

Trong ®ã: $M_0 = x_3 + x_2 + x_1$

$M_1 = x_3 + x_2 + \overline{x}_1$; $M_4 = \overline{x}_3 + x_2 + x_1$

0, 1, 4 lµ gi¸ trÞ thËp ph©n cña c¸c tæ hîp biÕn mµ f = 0; N = 2, 6 lµ gi¸ trÞ thËp ph©n cña tæ hîp biÕn mµ f = x (kh«ng x¸c ®Þnh).

- ¦u ®iÓm:

+ C¸ch viÕt gän; tiÕn lîi; tÝnh kh¸i qu¸t cao (do biÓu diÔn trùc tiÕp quan hÖ logic gi÷a c¸c biÕn).

+ RÔ dµng sö dông c¸c c«ng thøc vµ ®Þnh lý cña ®¹i sè logic ®Ó biÕn ®æi, lµm to¸n.

+ TiÕn cho viÖc dïng s¬ ®å logic ®Ó thùc hiÖn hµm sè (chØ cÇn dïng c¸c ký hiÖu cña c¸c cæng logic t¬ng øng thay thÕ phÐp to¸n trong biÓu thøc hµm sè , ta sÏ cã mét s¬ ®å logic).

- Nh-îc: kh«ng trùc quan nh b¶ng ch©n lý (khã x¸c ®Þnh hµm øng víi gi¸ trÞ biÕn mét c¸ch trùc tiÕp ®èi víi c¸c hµm sè phøc t¹p).

III- BÌA CÁC-NÂU (KARNAUGH)

1- Cách xây dựng bìa Các-nâu (bìa K)

- Hµm cã n biÕn, bªa cã 2n « (t¬ng øng sÏ lµ sè hµng vµ cét), mçi « øng víi mét tæ hîp biÕn. C¸c « c¹nh nhau (hoÆc ®èi xøng nhau) chØ kh¸c nhau mét biÕn.

- Trªn c¸c cét vµ hµng (bªn ngoµi b¶ng) ghi c¸c tæ hîp gi¸ trÞ biÕn sao cho nh÷ng cét vµ hµng c¹nh nhau (hoÆc ®èi xøng nhau) chØ kh¸c nhau mét biÕn.

- Trong c¸c « ghi gi¸ trÞ cña hµm øng víi gi¸ trÞ tæ hîp biÕn t¹i « ®ã.

D¹ng CTT thÞ « cã f = 0 thêng ®Ó trèng.

D¹ng CTH thÞ « cã f = 1 thêng ®Ó trèng.

C¸c « mµ hµm kh«ng x¸c ®Þnh , ®¸nh dÊu X.

- VÝ dô: bªa K cña hµm 3 biÕn nh hªnh 1

+ D¹ng CTT: $f(x_3 x_2 x_1) = \Sigma(3.5.7)$; N = 2, 6 (hªnh 1a).

+ D¹ng CTT: $f(x_3 x_2 x_1) = \Pi(0,1,4)$; N = 2, 6 (hªnh 1b).

BÀI 2-3: PHƯƠNG PHÁP TỐI THIỂU HÀM LOGIC

I- KHÁI NIỆM

1- Quan hệ giữa biểu thức và mạch logic

- §Æc ®iÓm, quan hÖ gi÷a c¸c d¹ng tÝch vµ d¹ng tæng trong hµm logic th-êng thÓ hiÖn ë c¸c lo¹i sau: OR-AND; AND-OR; NAND-NAND; NOR-NOR; NOR-AND; ...

- VÝ dô:

$f = x_1x_2 + x_1x_3$ $\Rightarrow$ OR-AND

$f = (x_1 + x_3)(x_1 + x_2)$ $\Rightarrow$ AND-OR

$f = \overline{\overline{x_1x_2}\ \overline{x_1x_3}}$ $\Rightarrow$ NAND-NAND

$f = \overline{\overline{x_1 + x_3} + \overline{x_1 + x_2}}$ $\Rightarrow$ NOR-NOR

$f = \overline{x_1\overline{x_2} + \overline{x_1}x_3}$ $\Rightarrow$ NOR-AND

Sö dông c¸c cæng logic: NOT, AND, OR, NAND, NOR,... ®Ó thùc hiÖn c¸c hµm logic lµ thuËn lîi nhÊt.

- Thùc tÕ cho thÊy, mét hµm logic cã thÓ ®îc m« t¶ b»ng nhiÒu biÓu thøc logic kh¸c nhau.

VÝ dô:

$f = x_1x_2x_3 + x_1x_2x_3 + x_1x_2x_3 + x_1x_2x_3$ (1a)

$= x_1x_2 + x_1x_3 + x_2x_3$ (1b)

$= x_1x_2 + x_1x_3$ (1c)

$= .. ..$

NÕu dïng cæng AND vµ OR thùc hiÖn (1c), ta cã m¹ch ®¬n gi¶n nhÊt.

2- Tối thiểu hóa hàm logic

Tèi thiÓu hãa (rót gän) hµm logic lµ qu¸ trxnh ®i txm d¹ng biÓu diÔn ®¹i sè ®¬n gi¶n nhÊt cña hµm.

- Môc ®Ých:

Sè biÕn vµ sè c¸c sè h¹ng d¹ng tÝch (hoÆc tæng) cña hµm logic lµ nhá nhÊt nh-ng vÉn thùc hiÖn ®îc chøc n¨ng ®· ®Þnh (t¬ng ®¬ng v í i sè cÊu kiÖn ýt, ®é tin cËy cao, gi¸ thµnh rÎ h¬n,..).

- VÝ dô: BiÓu thøc (1b) ®¬n gi¶n h¬n (1a); biÓu thøc (1c) ®¬n gi¶n h¬n (1b).

II- CÁC PHƯƠNG PHÁP TỐI THIỂU HÓA HÀM LOGIC

1- Biến đổi đại số

- Dùa vµo c¸c ®Þnh lý, c«ng thøc, hÖ qu¶ vµ tÝnh chÊt cña ®¹i sè logic ®Ó tiÕn hµnh tèi thiÓu hãa.

1) $f = x_1x_2 + x_1x_2 + x_1x_2$ (6 cæng)

$= (x_1x_2 + x_1x_2) + (x_1x_2 + x_1x_2)$

$= x_2(x_1 + x_1) + x_1(x_2 + x_2)$

$f = x_2 + x_1$ (1 cæng)

2) $f = x1x2 + x1x3 + x2x3$

= x1x2 + (x1 + x2) x3
= x1x2 + x1 x2 x3 (§þnh lý Demorgan)
= x1x2 + x3 (c«ng thøc 28).

2- Dùng bìa Các-nâu

a) Nguyên tắc

- TiÕn hµnh rót gän theo 4 bwíc:

+ B-íc 1: BiÓu diÔn hµm ®· cho trªn bxa K theo d¹ng CTT (hoÆc CTH).

- TiÕn hµnh rót gän theo 4 b-íc:

+ B-íc 1: BiÓu diÔn hµm ®· cho trªn bxa K theo d¹ng CTT (hoÆc CTH).

+ B-c 2: gép 2^k « kÒ nhau (hoÆc ®èi xøng nhau) theo d¹ng CTT (hoÆc CTH). Cã thÓ kÕt hîp c¶ « kh«ng x¸c ®þnh x, víi k lµ tèi ®a ($0 \le k \le n$).

+ B-íc 3: TiÕn hµnh tèi thiÓu c¸c vßng ®· gép theo quy t¾c: nÕu biÕn nµo kh«ng thay ®æi gi¸ trþ thx gi÷ l¹i, ng-îc l¹i nÕu biÕn thay ®æi gi¸ trþ thx lo¹i. KÕt qu¶:

Gép 2« sÏ lo¹i ®-c 1 biÕn.

Gép 4« sÏ lo¹i ®-îc 2 biÕn.

Gép 2^k « sÏ lo¹i ®-îc k biÕn ($0 \le k \le n$).

- B-íc 4: ViÕt hµm ®· tèi thiÓu b»ng biÓu thøc.

Cã thÓ tiÕn hµnh tèi thiÓu theo d¹ng CTT hoÆc CTH.

@ Chó ý:

+ Vßng gép cµng lín cµng tèt, vx c¸c thõa sè nhËn ®-îc sau khi tèi thiÓu sÏ ýt nhÊt.

+ Vßng gép sau ph¶i cã ýt nhÊt mét sè h¹ng ch-a ®-îc gép ë vßng tr-íc ®ã.

+ Mét « cã thÓ tham gia nhiÒu lÇn gép.

+ 4 « ë 4 gãc cña bxa C¸c - n©u còng gép ®-îc víi nhau.

b) Dạng CTT

- Vý dô: Tèi thiÓu hãa hµm

$f(x_3x_2x_1x_0) = \Sigma 1, 5, 6, 7, 11, 13$; (N=12, 15)

b) Dạng CTH

- T-¬ng tù nh ë d¹ng CTT nhng tiÕn hµnh tèi thiÓu víi c¸c « 0 vµ x.

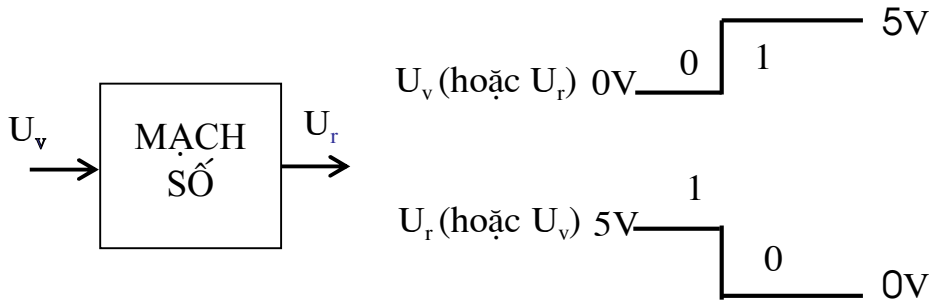
- Vý dô: Tèi thiÓu hãa hµm

$f(x_3x_2x_1x_0) = \Pi(3,5.6.7,12,13)$; (N= 0,2,15)

BÀI 3-1: NHỮNG VẤN ĐỀ CHUNG VỀ VI MẠCH SỐ

I- KHÁI NIỆM

- Mạch số (Digital Circuit) là mạch điện được thiết kế để tạo điện thế đầu ra (hoặc đầu vào) rơi vào các khoảng điện thế tương ứng với các mức logic 0 và 1 nh trên hình 1.



Hình 1: Sơ đồ khối mạch số và mức điện áp vào/ra

- Mach logic (Logic Circuit) là mạch số hoạt động tuân theo tập hợp quy tắc logic nhất định.
- Được chế tạo ở dạng IC, mỗi IC số có một chức năng xác định và được chế tạo theo từng công nghệ thích hợp.
- Hiện nay, các IC số thờng rất đa năng và có thể sử dụng linh hoạt trong nhiều thiết bị điện tử số khác nhau.
- Vi mạch số gồm:

+ Các mạch đơn giản: cổng logic cơ bản NOT, AND, OR, ...; mạch đếm; mạch dồn kênh, phân kênh;...

+ Các mạch phức tạp: bộ nhớ; mạch vào/ra dữ liệu; mạch điều khiển ngoại vi trong các máy vi tính; bộ vi xử lý; ...

- IC số được sử dụng rộng rãi trong nhiều lĩnh vực, nh: thông tin, đo lường, điều khiển,...
- Phân loại:

+ Theo bản chất của tín hiệu điện vào/ra: Vi mạch tương tự (Analog) và vi mạch số (Digital)

+ Theo mức độ tích hợp:

Vi mạch cỡ nhỏ: SSI (Small Scale Intergration).
Vi mạch cỡ vừa: MSI (Medium Scale Intergration).
Vi mạch cỡ lớn: LSI (Large Scale Intergration).
Vi mạch cực lớn: VLSI (Very Large Scale Intergration).

+ Theo công nghệ chế tạo: Đơn cực và lưỡng cực

II- CÁC ĐẠI LƯỢNG VẬT LÝ MÔ TẢ DỮ LIỆU VÀO/RA

1- Biểu diễn bằng tín hiệu điện thế

2- Biểu diễn bằng tín hiệu xung

III- CÁC THÔNG SỐ KỸ THUẬT

1- Mức logic

2- Trễ truyền đạt

3- Công suất

4- Độ ổn định nhiễu

5- Khả năng mắc tải vào/ra

6- Các tham số dòng/ áp

7- Giới hạn nhiệt độ,…

IV- CÁC DẠNG VI MẠCH SỐ

1- VI MẠCH SỐ THEO CÔNG NGHỆ LƯỠNG CỰC

- Chế tạo dựa trên cơ sở transistor lưỡng cực nhng phức tạp hơn loại đơn cực.
- Dùng phổ biến hiện nay: họ logic lưỡng cực bão hòa và không bão hòa.

a- Họ logic lưỡng cực bão hòa

- RTL : (Resitor Transistor Logic : Điện trở- Tranzito- Logic).
- DTL : (Diode Transistor Logic : Điôt- Tranzito- Logic).
- TTL : (Transistor Transistor Logic : Tranzito- Tranzito- logic).

b- Họ logic lưỡng cực không bão hòa

- ECL : (Emittor Coupled Logic : Logic ghép emitơ chung).
- Schottky TTL : (TTL dùng điôt schottky), đây là họ tiêu chuẩn dùng để xét các họ khác về mức logic .

Mức 0: (0 ÷ 0,8)V.

Mức 1 : (2,4 ÷ 5)V.

Tiêu biểu là họ 74 xx ; 74Lxx ; 74Sxx ; 74LSxx ...

2- VI MẠCH SỐ THEO CÔNG NGHỆ ĐƠN CỰC

- Đặc điểm:
  + Chế tạo dựa trên các transistor trường.
  + Mật độ tích hợp cao, công suất tiêu thụ nhỏ.
  + Đơn giản, dễ chế tạo, giá thành rẻ.
- Tùy theo loại MOSFET mà vi mạch chế tạo theo công nghệ này được chia thành các họ sau: PMOS, NMOS, CMOS.

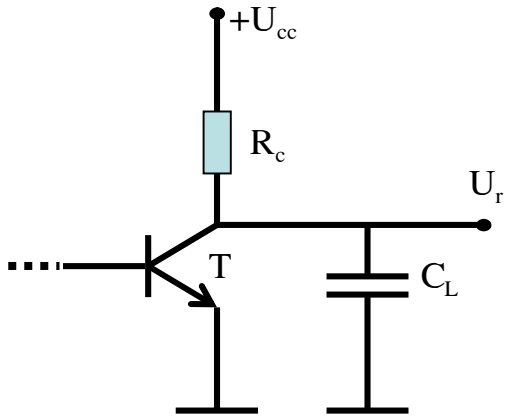
Tiêu biểu là họ 4000/14000; 74 HC/HCTxx; ...

BÀI 3-1: MẠCH ĐIỆN CÁC CỔNG LOGIC CƠ BẢN

I- CÁC KIỂU MẠCH RA CỦA VI MẠCH TTL

1- Mạch ra kéo lên thụ động (passlv-pull- up)

- Thuật ngữ kéo lên, còn gọi cung cấp dòng hoặc kéo xuống, còn gọi thu nhận dòng đề cập đến việc cung cấp dòng và thu nhận dòng của các transistor ở mạch ra các vi mạch số.



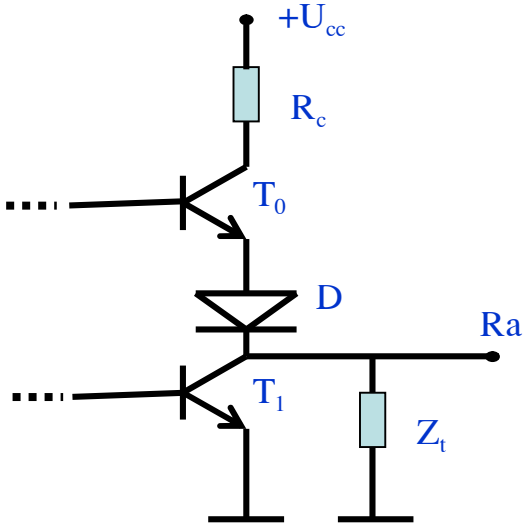
Hình 1.

+ $C_L = C_{lápráp} + C_{tải}$: tụ ký sinh

+ Nguyên lý: khi T tắt, Ura ở mức cao (H). Tụ C_L đợc nạp điện. Khi T thông bão hòa, tụ C_L phóng điện, Ura ở mức thấp (L).

2- Mạch ra kéo lên tích cực (Active- pull- up)

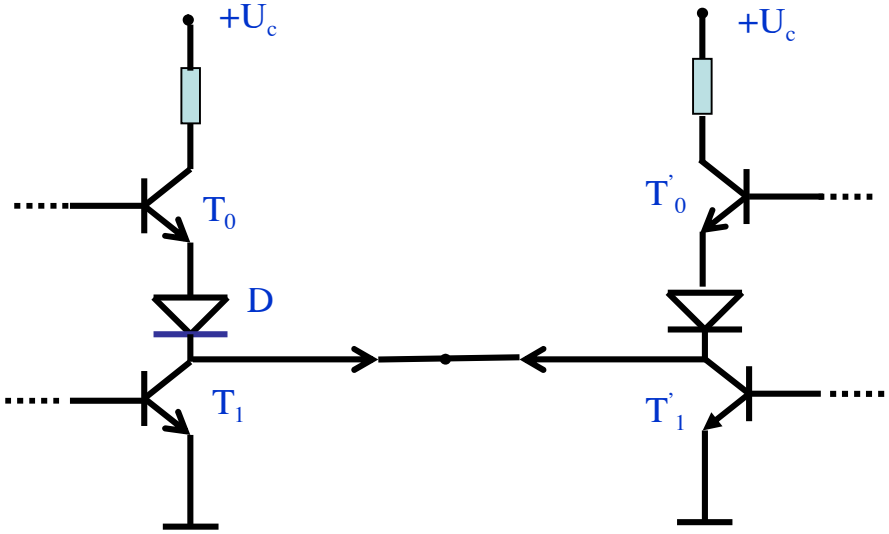
Mạch còn có tên mạch ra cột chạm (Totempole).



Hình 2.

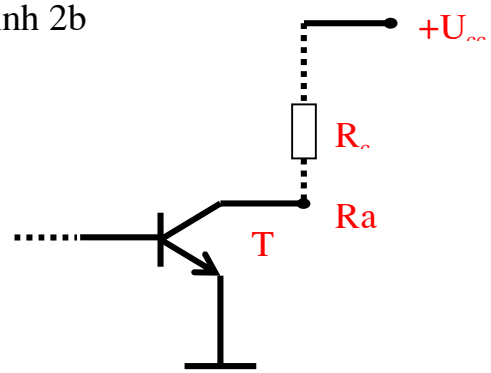
+ Mạch có : Zra = Z_T bão hòa , nên Zra rất nhỏ.

+ Nhược điểm: Dễ hỏng mạch khi nối chung các đầu ra. Ví dụ: hình 2b, mạch sẽ hỏng khi T_1 ,T'_0 cùng thông hoặc T'_1 , T_0 cùng thông (một đầu ra ở mức H, một đầu ra ở mức L).



Hình 2b

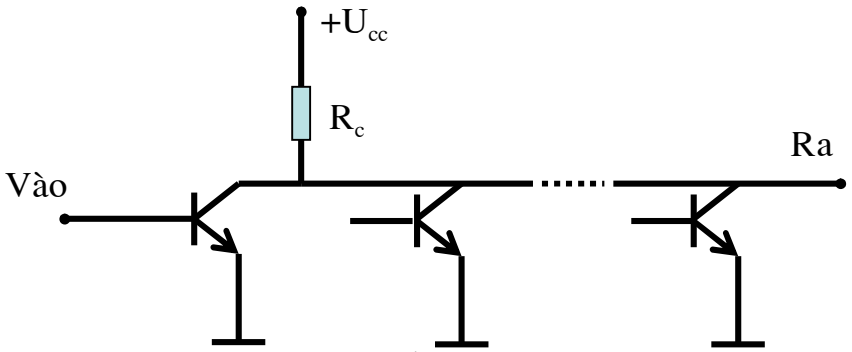
3- Mạch ra hở colecto (open collector)



Hình 3.

- Mạch có thể được sử dụng như mạch logic thông thường bằng cách mắc thêm Rc lên nguồn.

Gía trị của Rc (cỡ vài trăm Ω ÷ vài kΩ) phụ thuộc vào tải mắc ở đầu ra (có thể là TTL khác, LED,...).



Hình 4: Nối chung nhiều đầu ra mạch ra hở colector

Chương 3

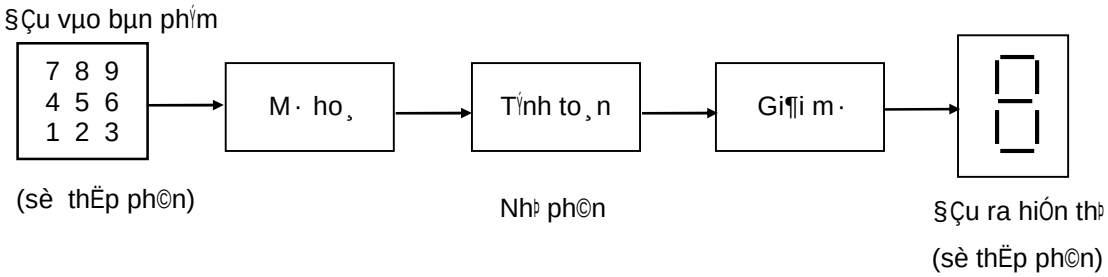
CÁC MẠCH TỔ HỢP THÔNG DỤNG

3.1 MẠCH CHUYỂN ĐỔI MÃ

I. Khái niệm

Trong các hệ thống điện tử số; các mạch chỉ có thể chế biến các bít 1 và 0 (ngôn ngữ máy - tín hiệu xung- tín hiệu số- hệ nhị phân). Tuy nhiên hệ nhị phân cũng có thể tồn tại dưới nhiều dạng khác nhau mà ta gọi là các dạng mã (đã đề cập ở chương 1).

Ví dụ: Mô hình làm việc của một máy tính loại đơn giản như hình 5-11.



Hình 5-11

Nguồn số liệu vào từ bàn phím là số thập phân (coi là mã thập phân). Quá trình tính toán là tín hiệu số (mã nhị phân). Sau khi tính toán xong phải đưa trở về tín hiệu (dạng mã) ban đầu thì con người mới hiểu được; do vậy cần có phần mạch trung gian để chuyển từ mã thập phân thành mã nhị phân gọi là mã hoá; sau đó lại phải chuyển kết quả từ mã nhị phân trở về mã thập phân gọi là giải mã.

Thông thường chuyển đổi từ kí hiệu (dạng mã) quen thuộc với con người sang một kí hiệu (dạng mã) không quen thuộc với con người bình thường thì gọi là mã hoá và quá trình chuyển đổi ngược lại gọi là giải mã.

Trên bàn phím của máy vi tính không chỉ có các số thập phân mà còn có các chữ cái, các kí hiệu, các dấu ..... khi tác động đều được chuyển thành tín hiệu số thông qua mạch mã hoá, sau đó thể hiện kết quả lên màn hình, máy in phải chuyển đổi ngược lại thông qua mạch giải mã.

Trong quá trình xử lý tin, lưu trữ, hiển thị .. còn có sự chuyển đổi qua lại dưới một số dạng mã như: BCD, thừa 3; Gray, Hexa; Octal..... tất cả các quá trình: mã hoá, giải mã, chuyển đổi mã ta có thể gọi tên chung là chuyển đổi mã. Suy cho cùng bất kỳ một dạng chữ viết , chữ số, kí hiệu, kí tự nào đó đều có thể coi là một dạng mã và sự chuyển đổi qua lại từ dạng nọ sang dạng kia đều được coi là chuyển đổi mã. Sau đây ta xét một số trường hợp cụ thể. ≠

II. Mã hoá

Khái niệm mã hoá (Encode) đã được đề cập ở trên; trong đó các kí hiệu được mã là các chữ cái; chữ số, kí hiệu, kí tự ... mã hoá có nhiệm vụ:

- Chuyển các kí hiệu đó thành các tín hiệu số (nhị phân). Sự mã hoá được thực hiện theo nguyên tắc $M \leq 2^N$.

Trong đó: M là số kí hiệu (tín hiệu) được mã.

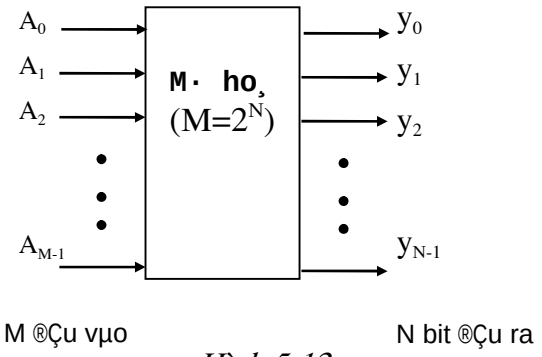
N là số bit nhị phân phải dùng để mã.

Thực tế có nhiều dạng mã hoá, ta xét một số dạng sau:

1. Mạch mã hoá nhị phân

Mạch mã hoá nhị phân là mạch điện, dùng N bit để mã cho M tín hiệu vào ($M = 2^N$). Mô hình tổng quát như hình 5-12.

Ở mỗi thời điểm chỉ có một đầu vào được mã (có mức tín hiệu tích cực) và tạo ra mã đầu ra N bít nhị phân tương đương.



Hình 5-12

Ví dụ: Cần mã 8 tín hiệu vào (M=8), thì số bit nhị phân đầu ra là 3 (N=3), bởi lẽ $M = 8 = 2^3 = 2^N$.

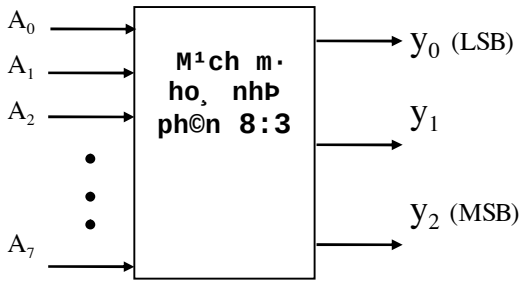
Sơ đồ khối mạch mã hoá nhị phân 8:3 như hình 5-13 và bảng trạng thái 5-6.

Căn cứ vào bảng 5-6 ta có các phương trình hàm ra sau:

$$y_0 = A_1 + A_3 + A_5 + A_7$$
$$y_1 = A_2 + A_3 + A_6 + A_7$$
$$y_2 = A_4 + A_5 + A_6 + A_7$$

Muốn cho sơ đồ logic dùng cổng NAND ta biến đổi dạng OR thành dạng NAND như sau:

$$y_0 = \overline{\overline{A_1 + A_3 + A_5 + A_7}} = \overline{\overline{A}_1 \cdot \overline{A}_3 \cdot \overline{A}_5 \cdot \overline{A}_7}$$
$$y_1 = \overline{\overline{A_2 + A_3 + A_6 + A_7}} = \overline{\overline{A}_2 \cdot \overline{A}_3 \cdot \overline{A}_6 \cdot \overline{A}_7}$$
$$y_2 = \overline{\overline{A_4 + A_5 + A_6 + A_7}} = \overline{\overline{A}_4 \cdot \overline{A}_5 \cdot \overline{A}_6 \cdot \overline{A}_7}$$

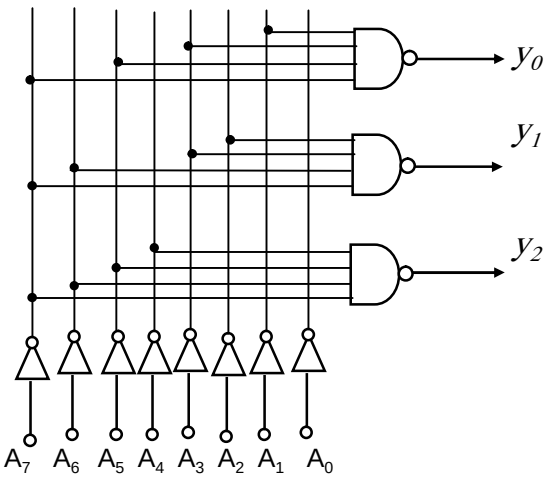


Hình 5-13

Bảng 5-6.

Đầu vào \ Đầu ra	y_2	y_1	y_0
A_0	0	0	0
A_1	0	0	1
A_2	0	1	0
A_3	0	1	1
A_4	1	0	0
A_5	1	0	1
A_6	1	1	0
A_7	1	1	1

Từ các phương trình dạng NAND, ta có sơ đồ logic như hình 5-14.

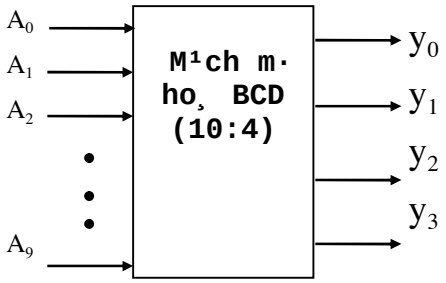


Hình 5-14

Lưu ý: A_0 thực tế không được nối với cổng logic vì các đầu ra của mạch mã hoá bình thường là 000 (tương ứng đầu A_0 được mã).

2. Mạch mã hoá thập phân - BCD

Mạch mã hoá nhị — thập phân (BCD-Binary-code-Decimal) là mạch điện chuyển mã hệ thập phân bao gồm 10 kí số :0,1,2,3,4,5,6,7,8,9 thành mã nhị phân như vậy M = 10 < 2^4 = 2^N, số bit dùng để mã là : N = 4, với 4 bit nhị phân sẽ có 16 tổ hợp nhị phân, nếu theo kiểu đếm tuần tự mạch sẽ dùng 10 tổ hợp đầu từ 0000 đến 1001 còn 6 tổ hợp cuối từ 1010 đến 1111 là thừa. Ta có sơ đồ khối như hình 5-15 và bảng trạng thái 5-7.



Hình 5-15

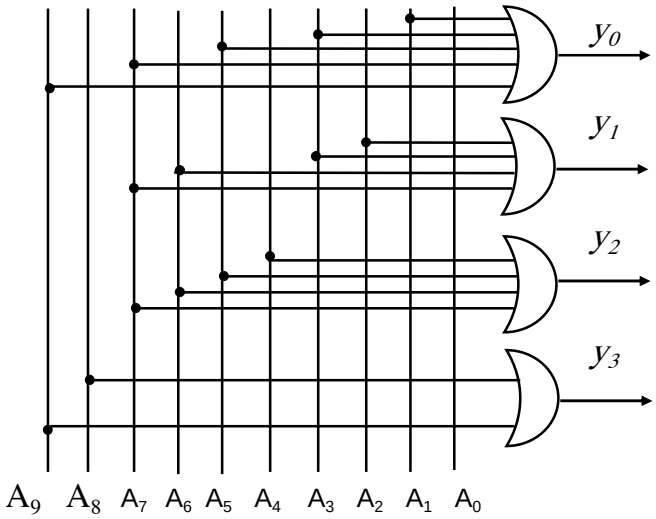
Căn cứ vào bảng 5-7, ta có các phương trình hàm ra sau:

$$y_0 = A_1+A_3+A_5+A_7+A_9$$
$$y_1 = A_2+A_3+A_6+A_7$$
$$y_2 = A_4+A_5+A_6+A_7$$
$$y_3 = A_8+A_9$$

Nếu để nguyên các phương trình ta có sơ đồ logic là các mạch OR, muốn chuyển dạng mạch NAND ta tiến hành phủ định 2 lần các phương trình trên sau đó dùng định luật Demon gan để chuyển thành NAND. Nếu để nguyên ta có sơ đồ logic hình 5-16.

Bảng 5-7

Đầu ra / Đầu vào	y_3	y_2	y_1	y_0
A_0	0	0	0	0
A_1	0	0	0	1
A_2	0	0	1	0
A_3	0	0	1	1
A_4	0	1	0	0
A_5	0	1	0	1
A_6	0	1	1	0
A_7	0	1	1	1
A_8	1	0	0	0
A_9	1	0	0	1



Hình 5-16

Nguyên lý làm việc của mạch mã hoá nhị — thập phân (BCD) cũng giống như mã hoá nhị phân. ở mỗi thời điểm chỉ có 1 đầu vào được mã, ở đầu ra sẽ có một tổ hợp nhị phân tương ứng; ví dụ khi có tín hiệu tích cực ở A_5 (tương ứng với số thập phân là 5) — tổ hợp mã nhị phân đầu ra là 0101. Bình thường mạch có tổ hợp nhị phân đầu ra là 0000 (tương ứng mã số 0).

3. Mạch mã hoá ưu tiên

Trong các mạch mã hoá đã xét ở trên, các tín hiệu đầu vào tồn tại độc lập (không có tình huống có 2 tín hiệu trở lên đồng thời tác động). Mạch mã hoá ưu tiên (Priority

Encoder) thì khác, có thể có nhiều tín hiệu đồng thời đưa đến, nhưng mạch điện chỉ tiến hành mã hoá tín hiệu đầu vào nào có cấp ưu tiên cao nhất ở thời điểm xét. Việc xác định cấp ưu tiên cho mỗi tín hiệu đầu vào là công việc của người thiết kế mạch.

Ta vẫn lấy ví dụ: mạch mã hoá nhị phân đối với 10 tín hiệu đầu vào từ A_0 , A_1, A_2 ... A_9 sao cho mức độ ưu tiên từ cao nhất đến thấp nhất theo chiều từ A_9 , A_8 ....A_1, A_0 (A_9 có mức ưu tiên cao nhất, A_0 có mức ưu tiên thấp nhất). Nếu có nhiều tín hiệu vào đồng thời xuất hiện ở đầu vào thì tín hiệu nào có mức ưu tiên cao nhất được mã hoá trước.

Giả thiết cả tín hiệu vào và tín hiệu ra có mức tích cực thấp. Mạch mã hoá này cũng có sơ đồ tương tự mạch mã hoá BCD đã đề cập ở trên, ta có bảng 5-8 cho mạch mã hoá này như sau:

Mức ưu tiên từ A_9 đến A_0, do đó những đầu vào có mức ưu tiên thấp không tác dụng gì đến đầu ra, đánh dấu "×" .

Bảng 5-8 là phương án mã BCD (8421) với mức logic âm.

Bảng 5-8:

A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	y_3	y_2	y_1	y_0
1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	0	×	1	1	1	0
1	1	1	1	1	1	1	0	×	×	1	1	0	1
1	1	1	1	1	1	0	×	×	×	1	1	0	0
1	1	1	1	1	0	×	×	×	×	1	0	1	1
1	1	1	1	0	×	×	×	×	×	1	0	1	0
1	1	1	0	×	×	×	×	×	×	1	0	0	1
1	1	0	×	×	×	×	×	×	×	1	0	0	0
1	0	×	×	×	×	×	×	×	×	0	1	1	1
0	×	×	×	×	×	×	×	×	×	0	1	1	0

Vì có nhiều biến số ta dùng phương pháp đại số để tối thiểu hoá. Căn cứ bảng 5-8, ta có các phương trình hàm ra như sau:

$$\overline{y}_3 = \overline{A}_9 + A_9 \cdot \overline{A}_8 = \overline{A}_9 + \overline{A}_8$$

Suy ra:

$$y_3 = \overline{\overline{A}_9 + \overline{A}_8}$$

$$\overline{y}_2 = A_9 A_8 \overline{A}_7 + A_9 A_8 A_7 \overline{A}_6 + A_9 A_8 A_7 A_6 \overline{A}_5 + A_9 A_8 A_7 A_6 A_5 \overline{A}_4$$

$$= A_9 A_8 \overline{A}_7 + A_9 A_8 \overline{A}_6 + A_9 A_8 \overline{A}_5 + A_9 A_8 \overline{A}_4$$

Suy ra:

$$y_2 = \overline{A_9 A_8 \overline{A}_7 + A_9 A_8 \overline{A}_6 + A_9 A_8 \overline{A}_5 + A_9 A_8 \overline{A}_4}$$

$$\overline{y}_1 = A_9 A_8 \overline{A}_7 + A_9 A_8 A_7 \overline{A}_6 + A_9 A_8 A_7 A_6 A_5 A_4 \overline{A}_3 + A_9 A_8 A_7 A_6 A_5 A_4 A_3 \overline{A}_2$$

$$\overline{y}_1 = A_9 A_8 \overline{A}_7 + A_9 A_8 \overline{A}_6 + A_9 A_8 A_5 A_4 \overline{A}_3 + A_9 A_8 A_5 A_4 \overline{A}_2$$

Suy ra:

$$y_1 = \overline{A_9 A_8 \overline{A}_7 + A_9 A_8 \overline{A}_6 + A_9 A_8 A_5 A_4 \overline{A}_3 + A_9 A_8 A_5 A_4 \overline{A}_2}$$

$$\overline{y}_0 = \overline{A}_9 + A_9\overline{A}_8 + A_9A_8\overline{A}_7 + A_9A_8A_7\overline{A}_6 + A_9A_8A_7A_6\overline{A}_5 + A_9A_8A_7A_6A_5\overline{A}_4 + A_9A_8A_7A_6A_5A_4\overline{A}_3 + A_9A_8A_7A_6A_5A_4A_3\overline{A}_2 + A_9A_8A_7A_6A_5A_4A_3A_2\overline{A}_1$$

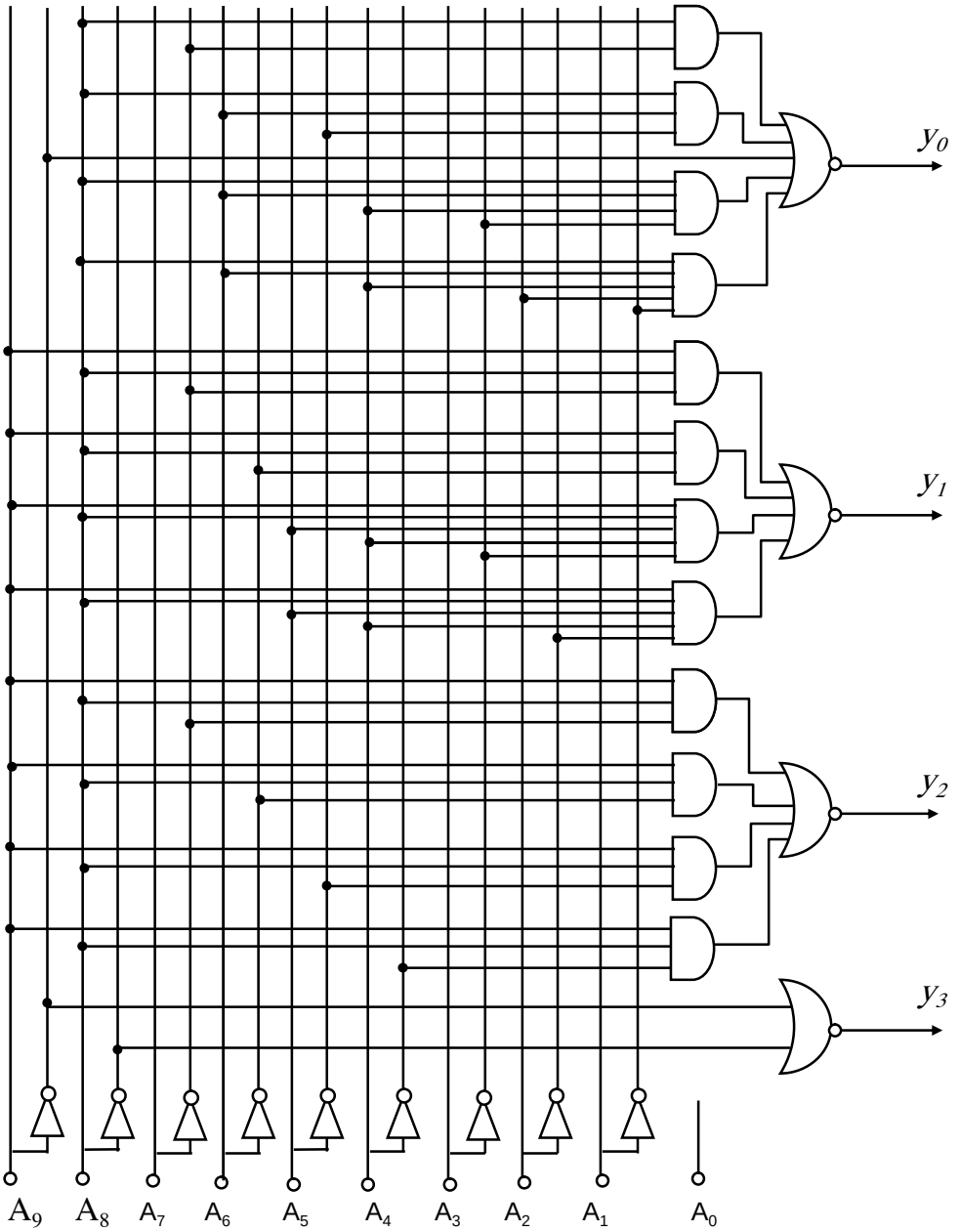
$$\overline{y}_0 = \overline{A}_9 + A_8\overline{A}_7 + A_8A_6\overline{A}_5 + A_8A_6A_4\overline{A}_3 + A_8A_6A_4A_2\overline{A}_1$$

Suy ra:

$$y_0 = \overline{\overline{A}_9 + A_8\overline{A}_7 + A_8A_6\overline{A}_5 + A_8A_6A_4\overline{A}_3 + A_8A_6A_4A_2\overline{A}_1}$$

Từ các phương trình: y_0, y_1, y_2, y_3 ta có sơ đồ logic như hình 5-17.

Giả sử tất cả các đầu vào đều có mức tích cực 0 , thì $y_3y_2y_1y_0 = 0110$, đây là mã tương ứng với đầu vào A_9(A_9: có mức ưu tiên cao nhất), tất cả các đầu vào có mức ưu tiên thấp hơn không có tác dụng đối với mạch mã hoá. Nếu tất cả các đầu vào có mức logíc 1 thì $y_3y_2y_1y_0 = 1111$ (đây là mã ngầm định — tương ứng với đầu vào A_0).



Hình 5-17

4. ứng dụng của bộ mã hoá

a) Một số vi mạch mã hoá thường gặp

IC: 74148; 74LS148; 74HC148 là các mạch mã hoá ưu tiên (8 đầu vào 3 đầu ra) mã hoá nhị phân .

IC: 74147; 74LS147 là mạch mã hoá ưu tiên BCD (10 đầu vào 4 đầu ra).

IC: 74 LS348- Mạch mã hoá ưu tiên, đầu ra 3 trạng thái.

IC: 74184 — chuyển đổi mã BCD thành nhị phân và nhị phân thành BCD.

IC: 74185: chuyển đổi BCD thành nhị phân và nhị phân thành BCD.

b) Ứng dụng

Ví dụ dùng IC 74147 làm bộ mã hoá chuyển mạch (SWitch Encoder) 10 chuyển mạch có thể là các chuyển mạch phím bấm trên máy tính bấm hiển thị 10 kí số từ 0÷9. 74LS147 là bộ mã hoá ưu tiên, nên nhiều phím bấm đồng thời nhưng sẽ tạo mã BCD cho phím có số thứ tự cao hơn trước.

Sơ đồ hình 5-18 là trường hợp thu nhận 3 kí số thập phân được nhập từ bàn phím theo thứ tự, mã hoá chúng thành mã BCD và lưu mã BCD vào 3 thanh ghi 12DFF ($Q_0 \div Q_{11}$), chuyên tiếp nhận lưu giữ mã BCD cho các kí số.

Ví dụ để nhập số 309.

1) Phím Clear — nhấn, xoá các FF ($Q_0 \div Q_{11}$ và X,Y) lập FFZ tại 1 sao cho bộ đếm vòng bắt đầu ở trạng thái 001 (X= Y = 0; Z = 1).

2) Thả phím Clear, phím số 3 được nhấn. các đầu ra 1100 của bộ mã hoá đảo thành 0011 là mã BCD cho 3 và được gửi đến đầu vào D của 3 thanh ghi đầu ra 4 bit.

3) Đầu ra OR lên cao (vì 2 đầu vào của nó đã ở mức cao) khởi động đầu ra Q = 1 của OS trong 20ms, sau 20ms; Q trở về thấp, đếm nhịp bộ đếm vòng đến trạng thái 100 (X lên cao, xung đầu ra X chuyển từ 0 lên 1) được đưa đến đầu vào CLK của các FF: $Q_8 \div Q_{11}$ sao cho đầu ra của bộ mã hoá truyền đến 4 FF này. Có nghĩa là $Q_{11} = 0$, $Q_{10} = 0$; $Q_9 = 1$, $Q_8 = 1$. Lưu ý 8FF từ $Q_0 \div Q_7$ không bị ảnh hưởng, vì đầu vào CLK của chúng không nhận được xung nhịp .

4) Phím số 3 được thả đầu ra cổng OR về thấp. Phím nhấn kế tiếp là "0" tạo mã BCD 0000 được đưa đến đầu vào của 3 thanh ghi.

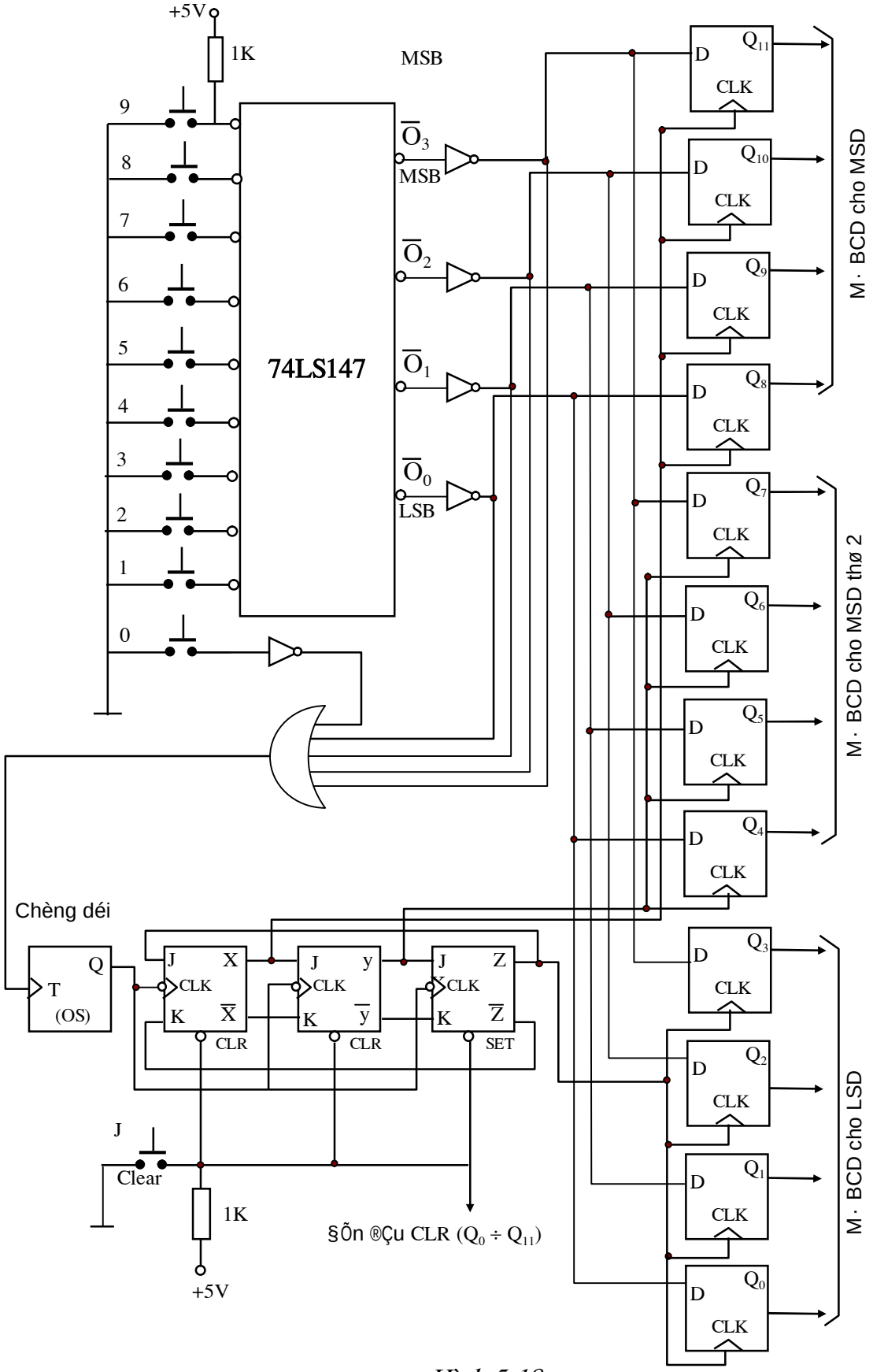
5) Đầu ra OR lên cao đáp lại phím O (lưu ý bộ đảo), khởi động OS trong 20ms. Sau 20ms bộ đếm vòng chuyển sang trạng thái 010 (Y lên cao) xung đầu ra Y chuyển từ 0 lên 1 được đưa đến đầu vào CLK của $Q_4 \div Q_7$ truyền mã 0000 đến 4 FF này lưu ý các FF $Q_0 \div Q_3$, $Q_8 \div Q_{11}$ không bị xung đầu ra Y tác động.

6) Phím "O" được thả, đầu ra OR trở về thấp. Phím 9 được nhấn tạo đầu ra BCD là 1001, chuyển đến 3 thanh ghi .

7) Đầu ra OR lên cao, khởi động OS, OS đếm nhịp bộ đếm đến trạng thái 001 (Z lên cao) xung đầu ra tại Z được đưa đến đầu vào CLK của các FF: $Q_0 \div Q_3$ truyền 1001 đến 4FF này, số FF còn lại không bị tác động.

8) Đến thời điểm này các thanh ghi lưu trữ chứa mã BCD 001100001001. Đây là mã BCD của số thập phân 309. Đầu ra các thanh ghi cấp cho bộ giải mã để hiển thị các kí số thập phân 309.

9) Đầu ra của các FF lưu trữ cũng được gửi tới các mạch khác trong hệ thống. Trong máy tính bấm chẳng hạn, các đầu ra này sẽ được gửi tới bộ phận số học để xử lý.


+5V
1K
MSB
9
8
7
6
5
4
3
2
1
0
74LS147
$\overline{O}_3$
MSB
$\overline{O}_2$
$\overline{O}_1$
$\overline{O}_0$
LSB
D
CLK
Q_{11}
Q_{10}
Q_9
Q_8
Q_7
Q_6
Q_5
Q_4
Q_3
Q_2
Q_1
Q_0
M· BCD cho MSD
M· BCD cho MSD thø 2
M· BCD cho LSD
Chèng déi
Q
T
(OS)
J
X
CLK
K
$\overline{X}$
CLR
y
$\overline{y}$
Z
$\overline{Z}$
SET
J
Clear
1K
+5V
§Õn ®Çu CLR ($Q_0 \div Q_{11}$)

Hình 5-18

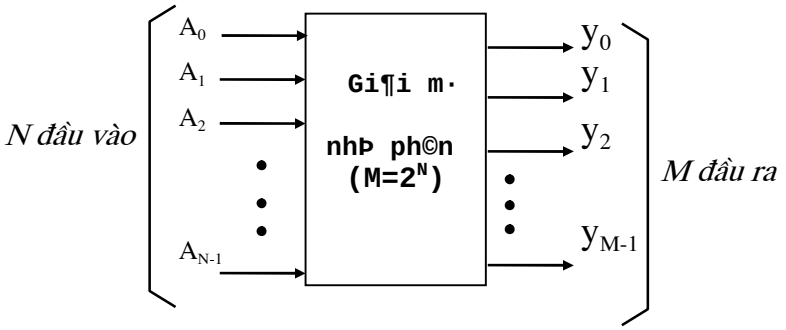
II. Giải mã

Mã hoá - mỗi từ mã nhị phân được gán một hàm ý xác định, tức là mỗi từ mã biểu thị một tin tức hoặc một đối tượng xác định. Giải mã là quá trình ngược lại — phiên dịch hàm ý đã gán cho từ mã (Decoder). Mạch giải mã phiên dịch từ mã thành tín hiệu đầu ra biểu thị tin tức vốn có, tín hiệu đầu ra có thể là xung hay mức điện áp.

Giả xử mạch giải mã có N đầu vào nhị phân, M đầu ra, vì mỗi đầu vào nhị phân có thể là "0" hoặc "1" nên có 2^N tổ hợp nhị phân đầu vào, và mạch giải mã cũng tuân thủ qui tắc: $M \le 2^N$ với mỗi tổ hợp nhị phân đầu vào chỉ có một đầu ra có mức tích cực. Mạch giải mã cũng có nhiều dạng như mã hoá, sau đây là một số mạch giải mã.

1. Mạch giải mã nhị phân

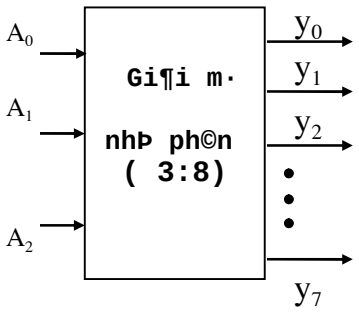
Mạch giải mã nhị phân thực hiện theo nguyên tắc: $M = 2^N$, nghĩa là số đầu ra (M) đúng bằng số tổ hợp nhị phân đầu vào (2^N). Mô hình giải mã nhị phân như hình 5-19.



Hình 5-19

Ví dụ: Mạch giải mã 3 đầu vào (3 bit)

N = 3, suy ra $M = 2^3 = 8$ đầu ra, ta có mô hình 5-20, bảng trạng thái 5-9.



Hình 5-20

Bảng 5-9

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

Căn cứ bảng 5-9, ta có các phương trình hàm ra sau:

$$y_0 = \overline{A}_2 \overline{A}_1 \overline{A}_0; \qquad y_4 = A_2 \overline{A}_1 \overline{A}_0$$

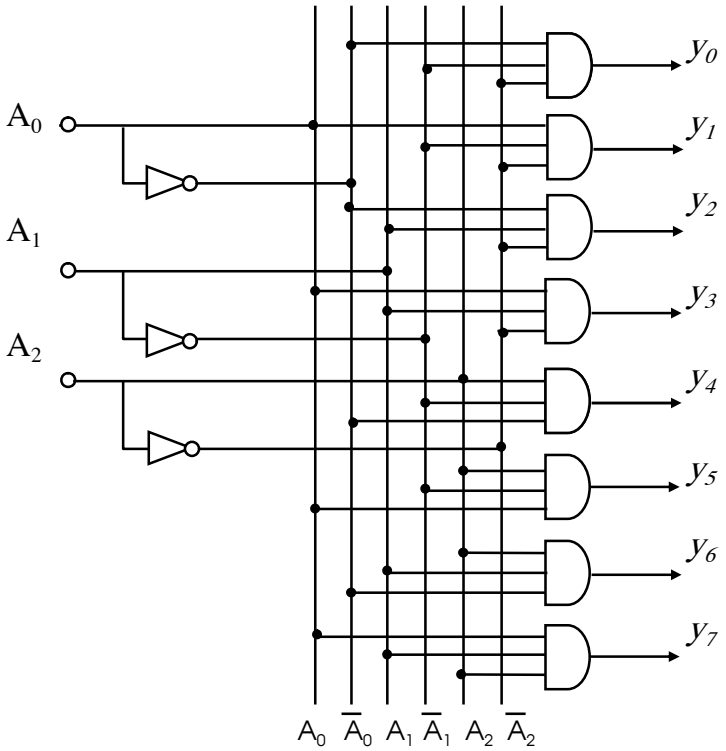
$$y_1 = \overline{A}_2 \overline{A}_1 A_0; \qquad y_5 = A_2 \overline{A}_1 A_0$$

$$y_2 = \overline{A}_2 A_1 \overline{A}_0; \qquad y_6 = A_2 A_1 \overline{A}_0$$

$$y_3 = \overline{A}_2 A_1 A_0; \qquad y_7 = A_2 A_1 A_0$$

Từ các phương trình ta có sơ đồ logic mạch giải mã nhị phân 3:8 như hình 5-21

Với sơ đồ hình 5-21, ở mỗi thời điểm chỉ có một đầu ra có mức tích cực ứng với tổ hợp nhị phân đầu vào, ví dụ với tổ hợp $A_2A_1A_0$ =001, đầu ra y_1 có mức tích cực 1, các đầu ra còn lại có mức tích cực 0, bình thường ở đầu y_0 có mức tích cực 1 (tương ứng $A_2A_1A_0$ = 000) sơ đồ cũng có thể dùng các cổng AND bằng điôt.



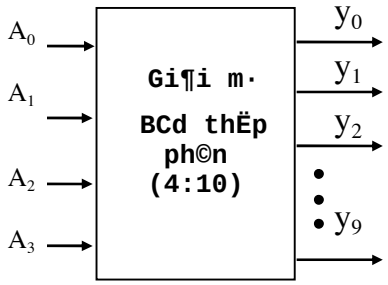
Hình 5-21

2. Mạch giải mã BCD — thập phân

Mạch giải mã BCD — thập phân thực hiện chuyển đổi từ mã BCD thành 10 chữ số của hệ thập phân. Như vậy mạch có 4 đầu vào nhị phân (N = 4) đáng lẽ có 16 tổ hợp nhị phân vào, nhưng do mã BCD (8421) chỉ dùng 10 tổ hợp từ (0000 ÷ 1001) còn 6 tổ hợp thừa (1010 ÷ 1111), có 10 đầu ra thập phân (từ $y_0 \div y_9$) đây là trường hợp:

$$M = 10 < 2^4 = 2^N$$

Mô hình 5-22; bảng trạng thái 5-10.



Hình 5-22

Bảng 5-10 với đầu vào là mã BCD (8421) dùng logic dương, đầu ra giải mã với mức tích cực 0; bảng dùng 10 tổ hợp đầu (từ 0000 ÷1001) còn 6 tổ hợp không dùng (1010 ÷ 1111), các trạng thái không dùng này coi là trạng thái không xác định, đánh dấu "×", do vậy khi sử dụng bìa các nâu để rút gọn các phương trình có thể gán cho "×" là 0 hoặc 1 theo điều kiện cụ thể. Để thiết lập phương trình cho các hàm ra dùng bìa các nâu (bảng 5-11).

Bảng 5-10:

A_3	A_2	A_1	A_0	y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0
1	0	1	0	×	×	×	×	×	×	×	×	×	×
1	0	1	1	×	×	×	×	×	×	×	×	×	×
1	1	0	0	×	×	×	×	×	×	×	×	×	×
1	1	0	1	×	×	×	×	×	×	×	×	×	×
1	1	1	0	×	×	×	×	×	×	×	×	×	×
1	1	1	1	×	×	×	×	×	×	×	×	×	×

Từ bìa các nâu ta có 10 phương trình cho 10 hàm ra như sau:

$\overline{y}_0 = \overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$ *hay* $y_0 = \overline{\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0}$

$\overline{y}_1 = \overline{A}_3\overline{A}_2\overline{A}_1 A_0$ *hay* $y_1 = \overline{\overline{A}_3\overline{A}_2\overline{A}_1 A_0}$

$\overline{y}_2 = \overline{A}_2 A_1\overline{A}_0$ *hay* $y_2 = \overline{\overline{A}_2 A_1\overline{A}_0}$

$\overline{y}_3 = \overline{A}_2 A_1 A_0$ *hay* $y_3 = \overline{\overline{A}_2 A_1 A_0}$

$\overline{y}_4 = A_2\overline{A}_1\overline{A}_0$ *hay* $y_4 = \overline{A_2\overline{A}_1\overline{A}_0}$

$\overline{y}_5 = A_2\overline{A}_1 A_0$ *hay* $y_5 = \overline{A_2\overline{A}_1 A_0}$

$\overline{y}_6 = A_2 A_1\overline{A}_0$ *hay* $y_6 = \overline{A_2 A_1\overline{A}_0}$

$\overline{y}_7 = A_2 A_1 A_0$ *hay* $y_7 = \overline{A_2 A_1 A_0}$

$\overline{y}_8 = A_3\overline{A}_0$ *hay* $y_8 = \overline{A_3\overline{A}_0}$

$\overline{y}_9 = A_3 A_0$ *hay* $y_9 = \overline{A_3 A_0}$

Bảng 5-11

y_0

A_3A_2 \ A_1A_0	00	01	11	10
00	(0)	1	1	1
01	1	1	1	1
11	×	×	×	×
10	1	1	×	×

y_1

A_3A_2 \ A_1A_0	00	01	11	10
00	1	(0)	1	1
01	1	1	1	1
11	×	×	×	×
10	1	1	×	×

y_2

A_3A_2 \ A_1A_0	00	01	11	10
00	1	1	1	(0)
01	1	1	1	1
11	×	×	×	×
10	1	1	×	(×)

y_6

A_3A_2 \ A_1A_0	00	01	11	10
00	1	1	1	1
01	1	1	1	(0)
11	×	×	×	×
10	1	1	×	×

y_7

A_3A_2 \ A_1A_0	00	01	11	10
00	1	1	1	1
01	1	1	(0)	1
11	×	×	×	×
10	1	1	×	×

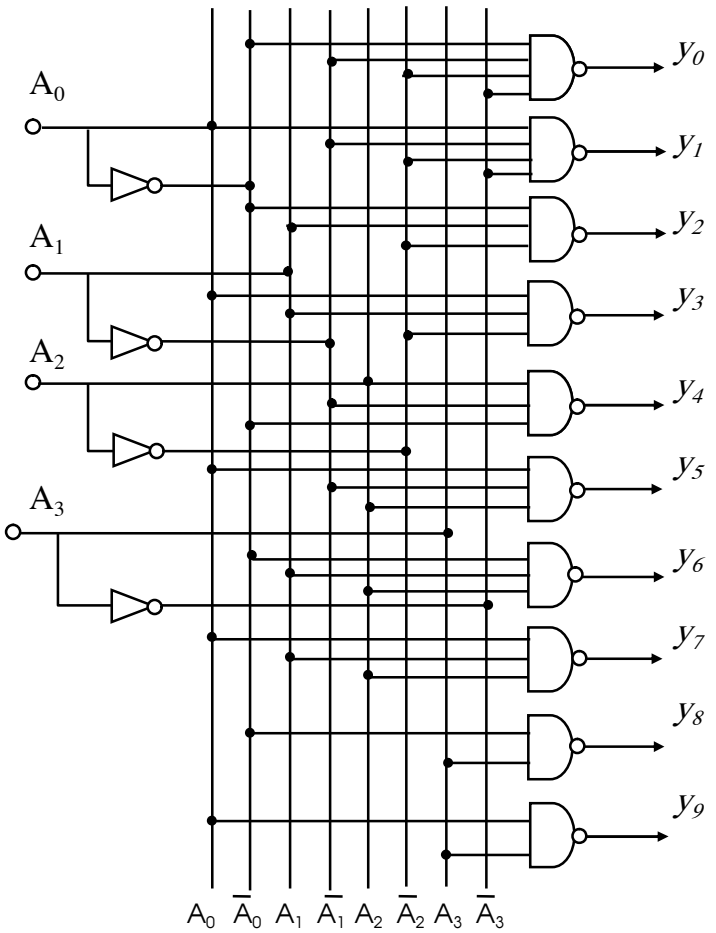
y_8

A_3A_2 \ A_1A_0	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	×	×	×	×
10	1	1	×	×

y_0	A_1A_0 00	01	11	10
A_3A_2 00	1	1	1	1
01	1	1	1	1
11	×	×	×	×
10	1	0	×	×

Từ các phương trình $y_0, y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9$ ta có sơ đồ logic mạch giải mã BCD thập phân như hình 5-23.

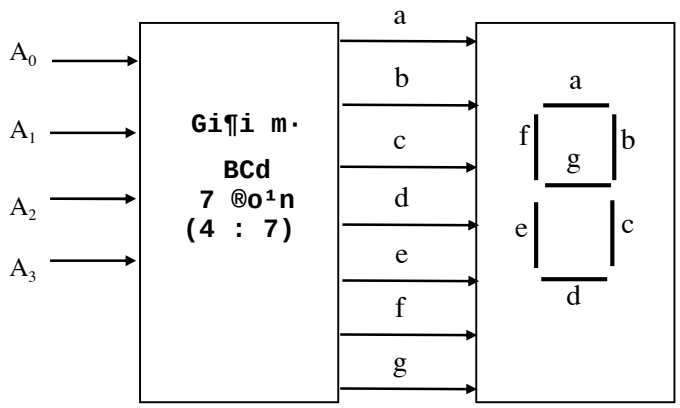
Hình 5-23 là mạch giải mã BCB thập phân dùng NAND. Tương tự như giải mã 3:8; ở mỗi thời điểm chỉ có 1 đầu ra có mức tích cực 0, ứng với tổ hợp nhị phân đầu vào. bình thường đầu y_0 có mức tích cực 0 (ứng với $A_3A_2A_1A_0 = 0000$), mỗi đầu ra thể hiện một số thập phân.



Hình 5-23

3. Mạch giải mã BCD - 7 đoạn

Giải mã BCD — 7 đoạn cũng thực hiện chuyển đổi mã BCD thành thập phân nhưng là 4 đầu vào 7 đầu ra. 7 đầu ra sẽ điều khiển 7 LED hoặc màn tinh thể lỏng để hiển thị số thập phân (7 đoạn). Mô hình 5-24, bảng trạng thái 5-12.



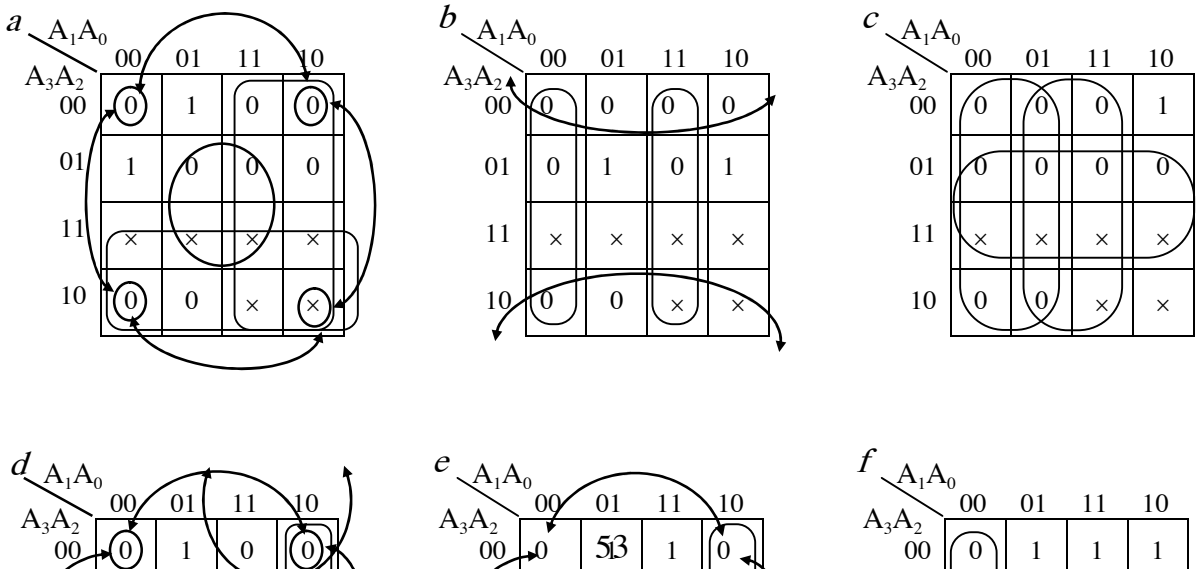
Hình 5-24

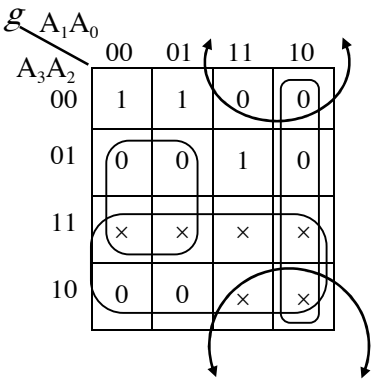
Bảng 5-12.

A_3	A_2	A_1	A_0	a	b	c	d	e	f	g	Số được hiển thị
0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	1	0	0	1	1	1	1	1
0	0	1	0	0	0	1	0	0	1	0	2
0	0	1	1	0	0	0	0	1	1	0	3
0	1	0	0	1	0	0	1	1	0	0	4
0	1	0	1	0	1	0	0	1	0	0	5
0	1	1	0	0	1	0	0	0	0	0	6
0	1	1	1	0	0	0	1	1	1	1	7
1	0	0	0	0	0	0	0	0	0	0	8
1	0	0	1	0	0	0	0	1	0	0	9

Sử dụng bìa các nâu để tối thiểu hoá và thiết lập các phương trình hàm ra a,b,c,d,e,f,g như bảng 5-13.

Bảng 5-13





Từ bảng 5-13 ta có phương trình hàm ra như sau:

$$\overline{a} = A_3 + A_1 + A_2A_0 + \overline{A}_2\overline{A}_0 \Rightarrow a = \overline{A_3 + A_1 + A_2A_0 + \overline{A}_2\overline{A}_0}$$

$$\overline{b} = \overline{A}_2 + A_1A_0 + \overline{A}_1\overline{A}_0 \Rightarrow b = \overline{\overline{A}_2 + A_1A_0 + \overline{A}_1\overline{A}_0}$$

$$\overline{c} = A_2 + \overline{A}_1 + A_0 \Rightarrow c = \overline{A_2 + \overline{A}_1 + A_0}$$

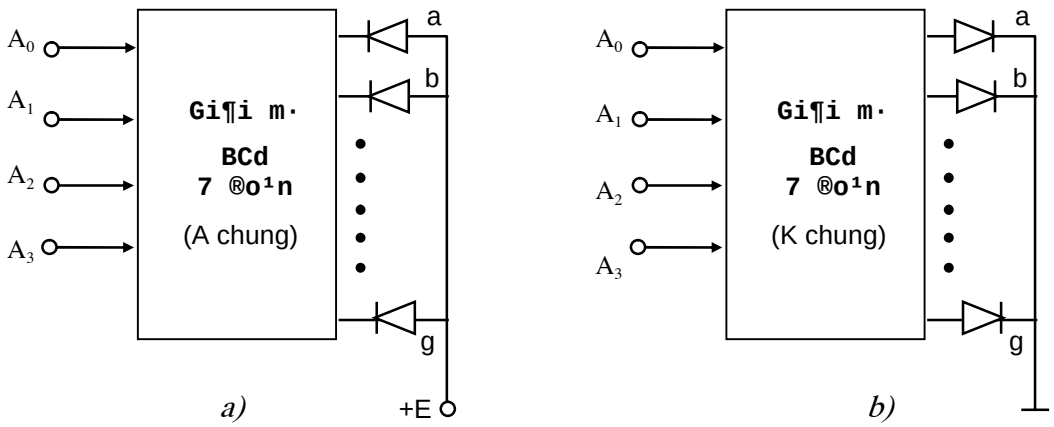
$$\overline{d} = A_3 + \overline{A}_2A_1 + A_1\overline{A}_0 + \overline{A}_2\overline{A}_0 + A_2\overline{A}_1A_0 \Rightarrow d = \overline{A_3 + \overline{A}_2A_1 + A_1\overline{A}_0 + \overline{A}_2\overline{A}_0 + A_2\overline{A}_1A_0}$$

$$\overline{e} = \overline{A}_2\overline{A}_0 + A_1\overline{A}_0 \Rightarrow e = \overline{\overline{A}_2\overline{A}_0 + A_1\overline{A}_0}$$

$$\overline{f} = A_3 + A_2\overline{A}_1 + A_2\overline{A}_0 + \overline{A}_1\overline{A}_0 \Rightarrow f = \overline{A_3 + A_2\overline{A}_1 + A_2\overline{A}_0 + \overline{A}_1\overline{A}_0}$$

$$\overline{g} = A_3 + A_2\overline{A}_1 + \overline{A}_2A_1 + A_1\overline{A}_0 \Rightarrow g = \overline{A_3 + A_2\overline{A}_1 + \overline{A}_2A_1 + A_1\overline{A}_0}$$

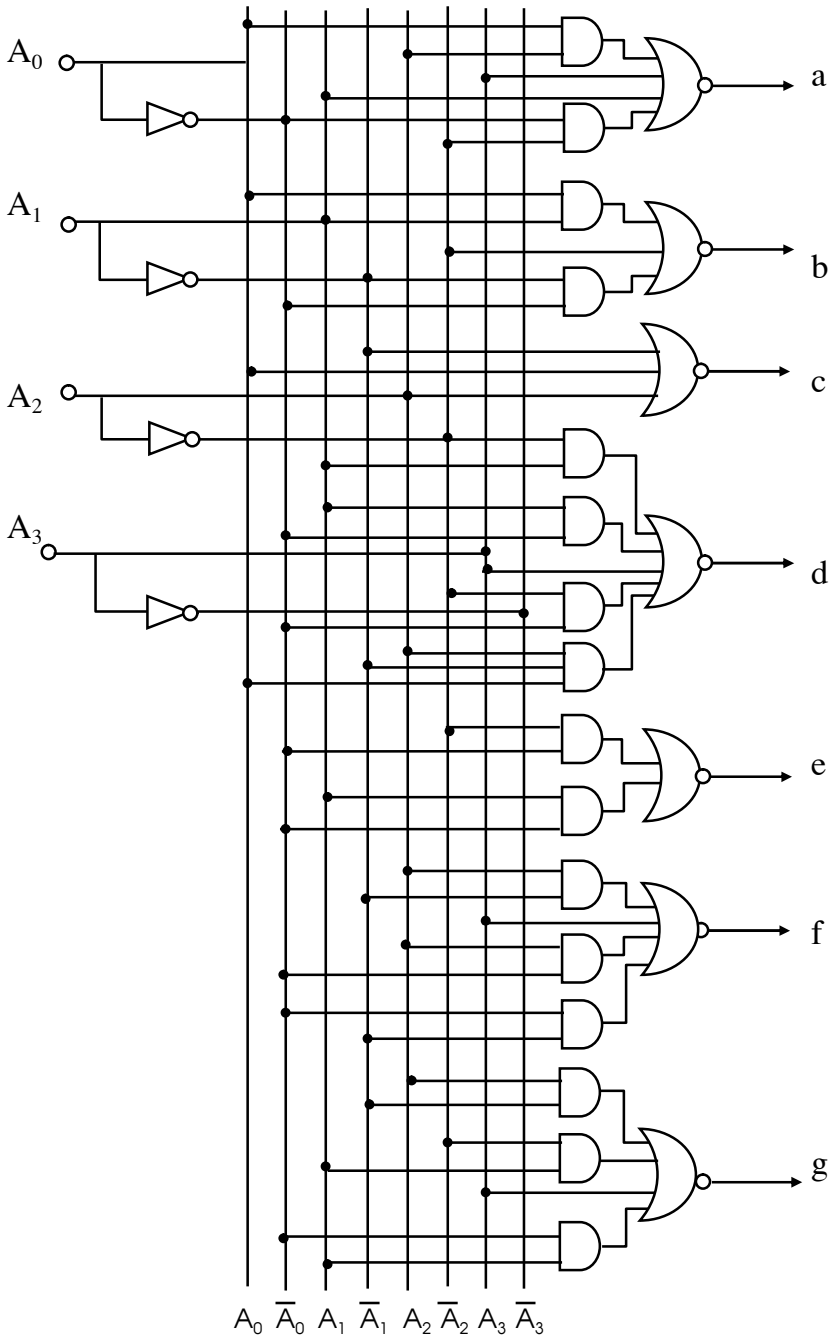
Bảng 5-12 với mức tích cực ra là 0, nghĩa là khi đầu ra đó có mức 0 thì LED tương ứng nối với đầu ra đó sẽ sáng. Theo kiểu này ra có sơ đồ giải mã Anôt chung như hình 5-25a. Còn nếu đầu ra là tích cực 1 ta có sơ đồ giải mã kiểu katốt chung như hình 5-25b.



Hình 5-25

Từ các phương trình *a,b,c,d,e,f,g* ta có sơ đồ logic như hình 5-26.

Mạch giải mã BCD — 7 đoạn cũng tuân theo nguyên tắc chung là ở mỗi thời điểm cũng chỉ giải mã cho đầu ra thể hiện một số thập phân. Tuy nhiên không phải là một đầu ra mà là sự kết hợp của 7 đầu ra: *a,b,c,d,e,f,g* để thể hiện một số thập phân. Bình thường đầu vào $A_3A_2A_1A_0 = 0000$, đầu ra $a = b = c = d = e = f = 0$ sẽ điều khiển phần hiển thị tương tứng sáng, đầu $g = 1$ điều khiển thanh *g* không sáng thể hiện số 0. Khi có tổ hợp $A_3A_2A_1A_0 = 0001$, đầu ra mạch giải mã có $b = c = 0$, điều khiển phần hiển thị thanh *b,c* sáng, các đầu ra còn lại mức 1 điều khiển các thanh *a,d,e,f,g* không sáng, thể hiện số 1... cứ như thế mạch giải mã cũng điều khiển để lần lượt hiển thị các số từ 0 đến 9.



Hình 5-26

4. Ứng dụng của bộ giải mã

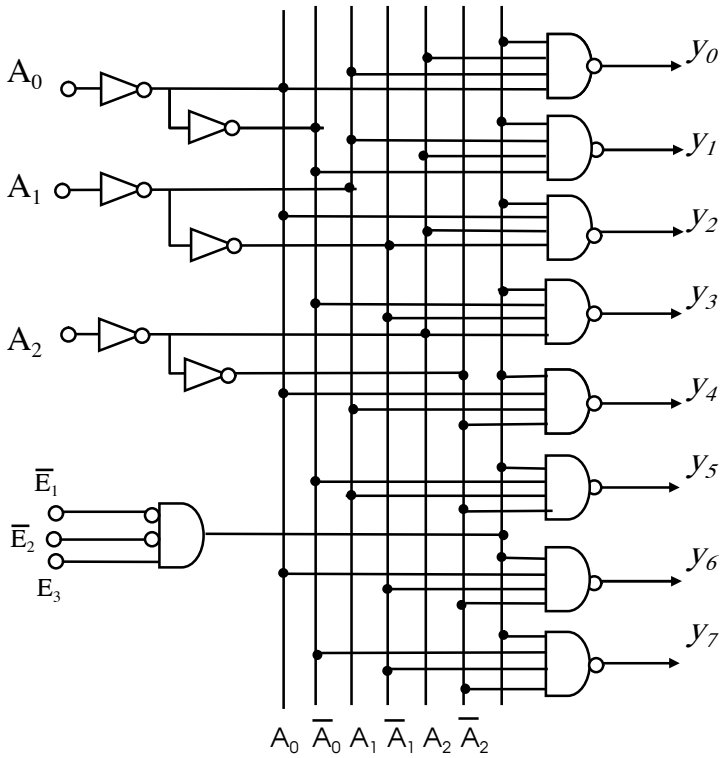
a) Một số vi mạch giải mã thường gặp

- IC: 74155; 74LS155 → giải mã nhị phân 2 sang 4.
- IC:74LS138; 74HC138 → giải mã nhị phân 3 sang 8.
- IC: 74154 → giải mã nhị phân 4 sang 16; có đầu ra hở.
- IC: 7441, 74141 → giải mã BCD — thập phân, đầu ra chịu điện áp cao (60V).
- 7442; 74LS 42 → giải mã BCD — thập phân .
- IC: 7445; 74145; 4LS145 → giải mã BCD — thập phân dòng lớn (80mA).
- IC:DM/7446A, 7447A, 74LS47, 5446A, 5447A → giải mã BCD — 7 đoạn đầu ra tích cực thấp.
- IC:DM/ 7448, 74LS48, 74LS49, 5448, 54LS48, 54LS49 → giải mã BCD — 7 đoạn đầu ra tích cực cao

b) Ứng dụng

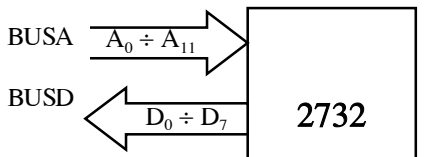
**Ví dụ :*

IC: 74LS138 — là một bộ giải mã nhị phân 3 sang 8; được sử dụng nhiều; sơ đồ logic như hình 5-27. Đầu ra mạch giải mã này mức tích cực thấp Các đầu vào $\overline{E}_1\overline{E}_2E_3$ là các đầu cho phép (Enble) — mạch giải mã sẽ hoạt động bình thường (cho phép) khi $\overline{E}_1 = \overline{E}_2 = 0$; $E_3 = 1$ chỉ cần 1 đầu vào E không có mức tích cực như qui định thì mạch sẽ không hoạt động (không có tín hiệu tích cực ở đầu ra như nguyên lý giải mã).



Hình 5-27

Trong các thiết bị số có nhiều thành phần mạch khác nhau để chọn mạch nào làm việc trước hay sau; người ta có thể sử dụng IC 74LS138 làm mạch chọn chip như hình 5-29.



Hình 5-29 là sơ đồ chọn các chip nhớ EPROM 2732 giả sử cần chọn 8IC 2732 ta dùng 1 IC 74 LS138. Mạch sử dụng 20 bit địa chỉ do vi xử lý phát ra, trong đó 12 bit ($A_0 \div A_{11}$) là các địa chỉ chọn ô nhớ của 2732; 3 bit tiếp theo dùng cho đầu vào giải mã 74LS138 (A_{12} A_{13}, A_{14}), các bit địa chỉ từ A_{15} đến A_{19} kết hợp cùng tín hiệu $IO/\overline{M}$ để điều khiển chọn IC74LS138 làm việc trên cơ sở đó để chọn các IC nhớ 2732. Để tiến hành chọn nhiều mạch (lớn hơn 8), ta cũng ghép nối các IC 74LS138 như hình 5-28.

IV. Các mạch chuyển mã khác

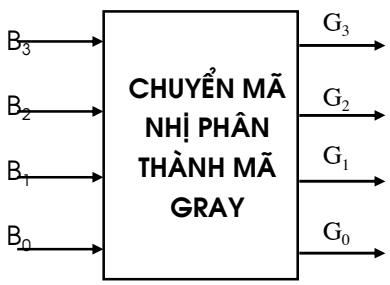
Trong các phần trên ta đã nghiên cứu 2 trường hợp chuyển mã là mã hoá và giải mã, ngoài ra còn có nhiều trường hợp chuyển mã khác như:

+ Mã nhị phân → mã Gray.
+ Mã ASCII → mã EBCDIC.
+ Mã BCD → mã nhị phân.
+ Mã thừa 3 → thừa 3 Gray.

Sau đây là một số trường hợp.

1. Mạch chuyển mã nhị phân mã Gray

Nếu các bit của mã nhị phân là $B_3B_2B_1B_0$ và các bit của mã Gray là $G_3G_2G_1G_0$ ta có sơ đồ khối hình 5-32, bảng trạng thái 5-14.



Hình 5-32

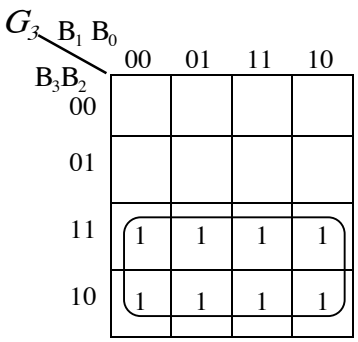
Bảng 5-14:

Mã nhị phân	Mã Gray

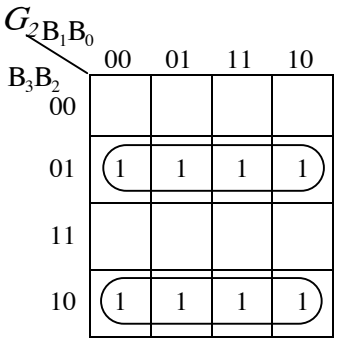
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Căn cứ vào bảng 5-14 ta có bìa các nâu (bảng 5-15).

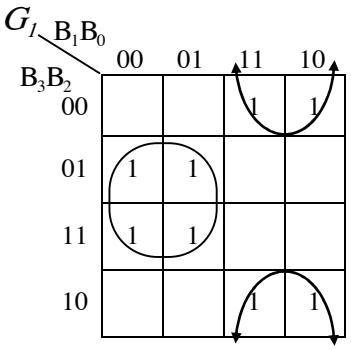
Bảng 5-15



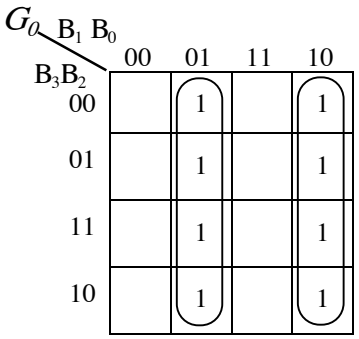
$G_3 = B_3$



$G_2 = \overline{B}_3 B_2 + B_3 \overline{B}_2 = B_2 \oplus B_3$

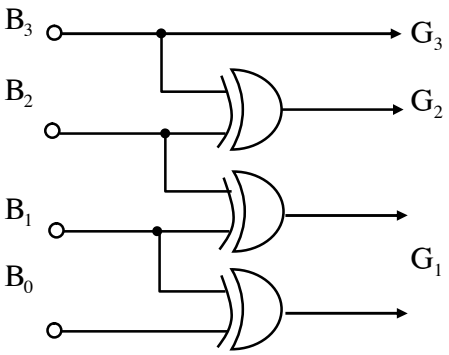


$G_1 = \overline{B}_2 B_1 + B_2 \overline{B}_1 = B_1 \oplus B_2$



$G_0 = \overline{B}_1 B_0 + B_1 \overline{B}_0 = B_0 \oplus B_1$

Từ các phương trình $G_3G_2G_1G_0$ ta có sơ đồ logic mạch chuyển mã nhị phân thành mã Gray như hình 5-33.



Hình 5-33

Bằng cách tương tự muốn chuyển ngược từ mã Gray sang mã nhị phân ta coi các bit của mã gray là biến vào, các bit của mã nhị phân là hàm ra, từ đó ta có các phương trình hàm ra sau: $B_3 = G_3$

$$B_2 = G_2 \oplus G_3$$
$$B_1 = G_1 \oplus G_2$$
$$B_0 = G_0 \oplus G_1$$

2. Mạch chuyển mã thừa 3 — thừa 3 gray

Hai loại mã này có bảng trạng thái 5-16; cũng tiến hành bìa các nâu để rút gọn ta có các phương trình hàm ra:

$$G_3^*, G_2^* \; G_1^*, G_0^*$$
$$G_3^* = B_3^*$$
$$G_2^* = B_2^* \oplus B_3^*$$
$$G_1^* = B_1^* \oplus B_2^*$$
$$G_0^* = B_0^* \oplus B_1^*$$

Sơ đồ logic hình 5-34a.

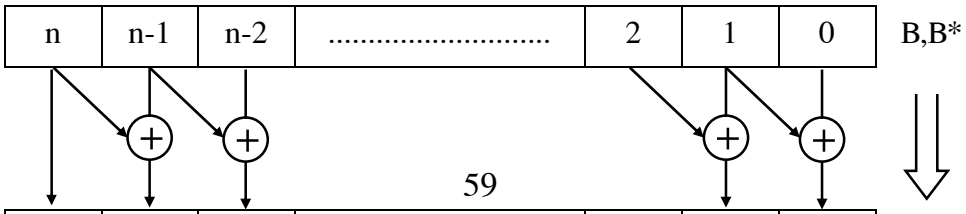
Bảng 5- 16

Mã thừa 3				Mã thừa 3 gray			
B_3^*	B_2^*	B_1^*	B_0^*	G_3^*	G_2^*	G_1^*	G_0^*
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
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1

Khi thực hiện chuyển mã ngược từ mã thừa 3 gray sang thừa 3, ta có các phương trình hàm ra $B_3^*, B_2^* \; B_1^*, B_0^*$

$$B_3^* = G_3^*$$
$$B_2^* = G_2^* \oplus G_3^*$$
$$B_1^* = G_1^* \oplus G_2^* \oplus G_3^*$$
$$B_0^* = G_0^* \oplus G_1^* \oplus G_2^* \oplus G_3^*$$

Sơ đồ logic hình 5-34b.



3.2 MẠCH DỒN KÊNH, MẠCH PHÂN KÊNH

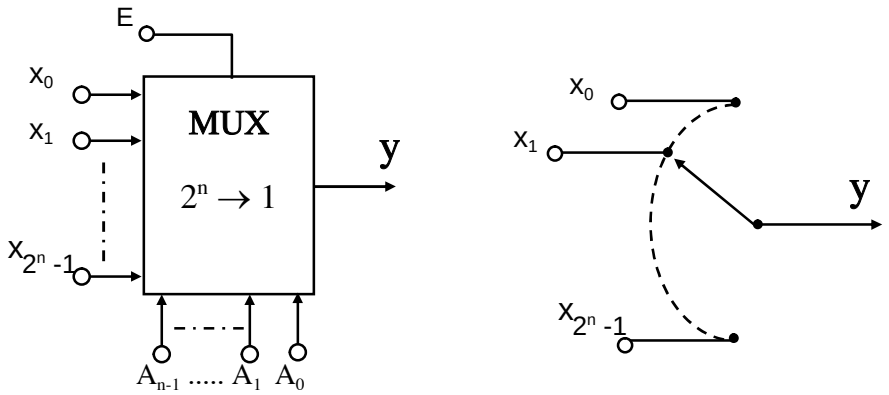
I. Mạch dồn kênh

1. Khái niệm

+Dồn kênh (Multiplexer —MUX), hay chọn kênh (Selector), có khả năng tại một thời điểm chọn một trong các tín hiệu vào để chuyển đến một đầu ra.

+ Bộ dồn kênh số (Digital Multiplexer converter) là một thiết bị số làm nhiệm vụ tại một thời điểm chọn 1 trong 2^n tín hiệu đầu vào để chuyển đến một đầu ra duy nhất theo sự điều khiển của tín hiệu địa chỉ.

Mô hình dồn kênh số, hình 5-37.



Hình 5-37

$x_0, x_1, ... \ x_{2^n-1}$: Các đầu vào tín hiệu số (dữ liệu) có 2^n tín hiệu vào.

$A_0, A_1,...A_{n-1}$: Các đầu vào điều khiển (địa chỉ) có *n* địa chỉ.

E (Enble): đầu vào chọn mạch (chọn chip).

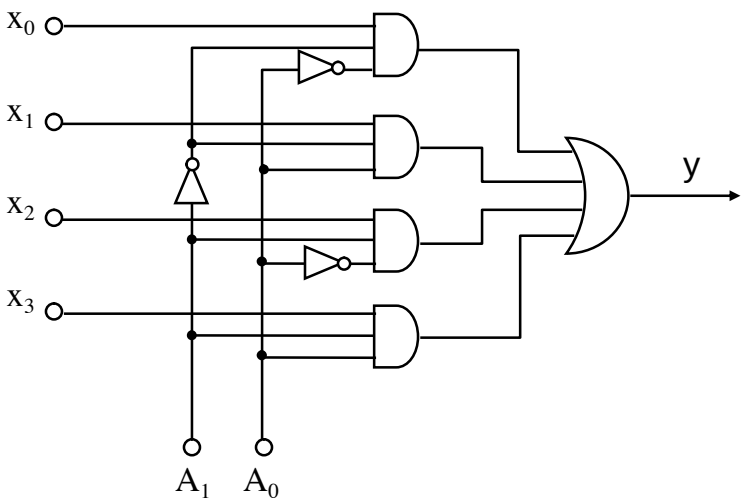
Y : đầu ra.

+Dồn kênh được xem như một đảo mạch (chuyển mạch điện tử) nhiều vị trí.

+ Phương trình hàm ra y của mạch có dạng:

$$y = x_0\left(\overline{A}_{n-1}\overline{A}_{n-2}\cdots\overline{A}_1\overline{A}_0\right) + x_1\left(\overline{A}_{n-1}\overline{A}_{n-2}\cdots\overline{A}_1 A_0\right) + \cdots + x_{2^n-1}\left(A_{n-1}A_{n-2}\cdots A_1 A_0\right)$$

2. Mạch dồn kênh 4 đầu vào



4 đầu vào: x_0, x_1, x_2, x_3 *Hình 5-38*

Cần có 2 biến (2 bit) địa chỉ: A_1, A_0 tạo thành 4 tổ hợp địa chỉ: $\overline{A}_1\overline{A}_0, \overline{A}_1 A_0, A_1\overline{A}_0$, A_1A_0 phương trình hàm ra y như sau:

$$y = x_0 \overline{A}_1 \overline{A}_0 + x_1 \overline{A}_1 A_0 + x_2 A_1 \overline{A}_0 + x_3 A_1 A_0$$

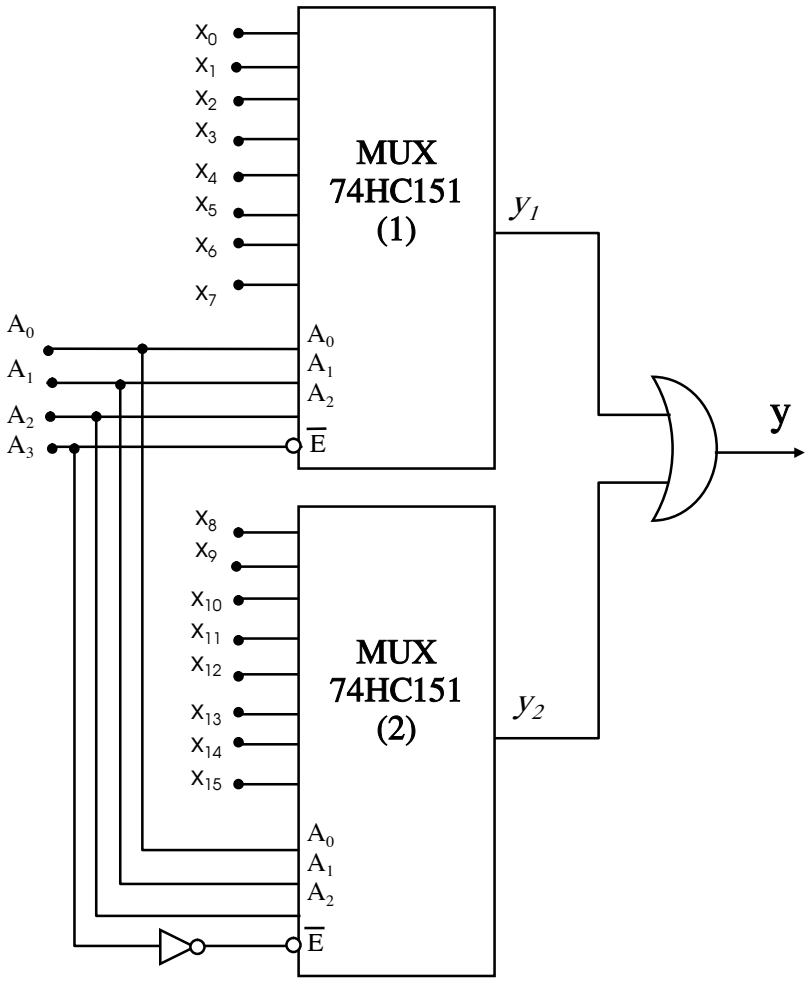
Căn cứ phương trình, có sơ đồ logic hình 5-38.

Ở mỗi thời điểm chỉ có tín hiệu của 1 đầu vào (1 kênh) được chọn lựa để đưa đến đầu ra y. Ví dụ với địa chỉ : $\overline{A}_1 \overline{A}_0 = 00$, tín hiệu x_0 được đưa đến y : $\overline{A}_1 A_0 = 01$ tín hiệu x_1 được đưa đến đầu ra y_1, ...

3. Ứng dụng của bộ dồn kênh

a) Một số IC dồn kênh thường gặp

- IC TTL 74157; 74158: gồm 4 bộ dồn kênh 2:1.
- IC 74151, 74LS151; 74HC151: dồn kênh 8:1. Khi cần tiếp nhận số đầu vào nhiều hơn ta có thể ghép nhiều IC 74151, chẳng hạn cần 16 đầu vào ta ghép 2IC theo sơ đồ hình 5-39.



Hình 5-39

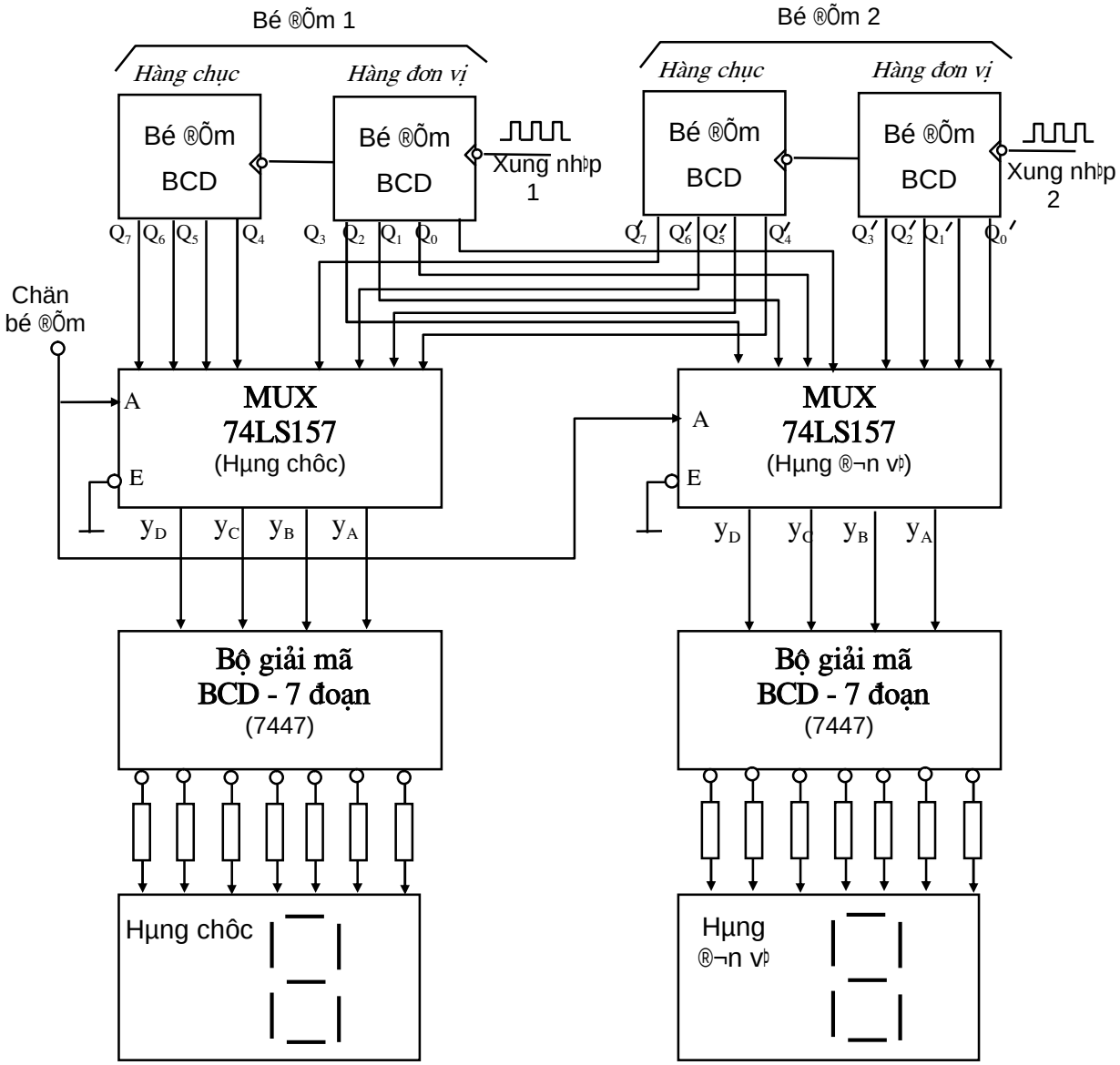
Đầu địa chỉ A_3 nối với $\overline{E}$ là đầu khống chế chọn chip. Nếu tổ hợp địa chỉ $A_3A_2A_1A_0$ thực hiện đếm tuần tự (A_0: bit bé nhất; A_3 bít lớn nhất) ở 8 tổ hợp đầu khi $A_3 = 0 = \overline{E}$; IC_1 sẽ làm việc tiếp nhận các đầu vào x_0 đến x_7 lần lượt đưa đến đầu ra y_1 rồi y (lúc này IC_2 không làm việc) 8 tổ hợp sau, khi $A_3 = 1 = \overline{E}$ qua mạch NOT, IC_2 làm việc (IC_1 không làm việc) tiếp nhận các đầu vào x_8 đến x_{15} đưa đến đầu ra y_2 rồi y.

b) Ứng dụng

Ví dụ 1: Định tuyến dữ liệu

Có nhiều dữ liệu được định hướng tới một đích duy nhất, sử dụng MUX cho phép chọn dữ liệu vào (định tuyến đầu vào) hướng tới đích, các dữ liệu không được chọn sẽ bị ngăn lại (cấm) không tới đích được. Ví dụ có 2 luồng dữ liệu ở dạng mã BCD 8 bit thể hiện kết quả đếm số xung tại các đầu ra song song của 2 bộ đếm xung, cần chọn một trong hai luồng BCD (8421) này tới đích là một khối giải mã BCD thành 7 vạch và hiển thị kết quả trên hai màn hình LED tương ứng số thập phân hàng chục và hàng đơn vị.

Sơ đồ 5-40 sử dụng MUX 74LS157 loại dồn kênh 2:1, mỗi IC 74157 có 4 mạch dồn kênh 2 đầu vào, sử dụng 2IC sẽ có 8 mạch dồn kênh 2:1.



Hình 5-40

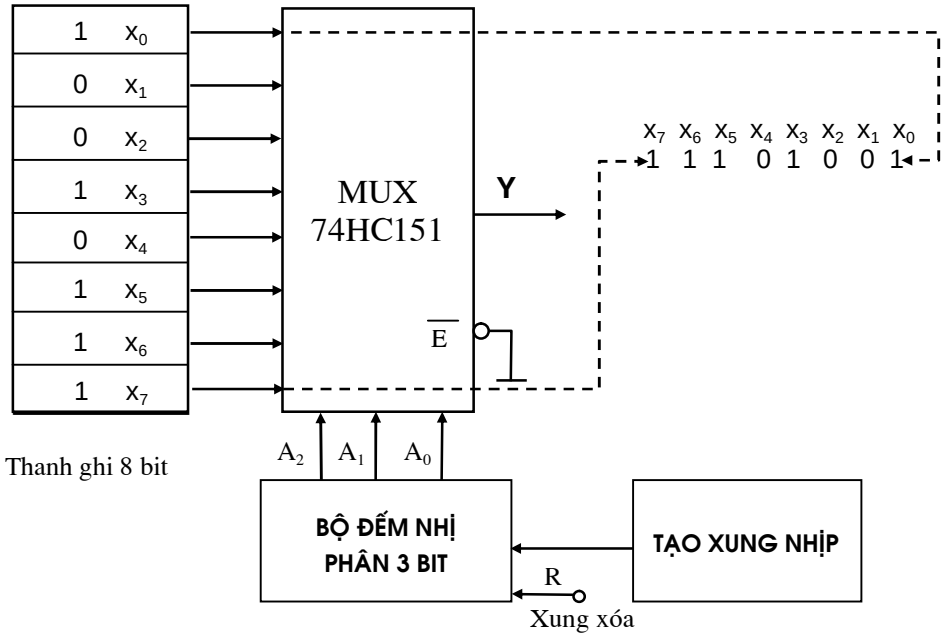
Sơ đồ cho phép bộ hiển thị sẽ báo kết quả của bộ đếm 1 hay bộ đếm 2 tuỳ theo A. Khi A = 1 bộ đếm 1 được chọn, dữ liệu từ bộ đếm 1 được đưa đến giải mã hiển thị kết quả. Khi A = 0 bộ đếm 2 được chọn dữ liệu từ bộ đếm 2 được đưa đến giải mã hiển thị.

Như vậy MUX để phân chia thời gian tới đích của các luồng dữ liệu sẽ giúp cho người thiết kế giảm được các chi phí về linh kiện, tăng độ tin cậy khi vận hành, giảm được tiêu hao năng lượng. Sơ đồ này thường gặp ở đồng hồ hiện số, mạch chịu trách nhiệm theo dõi giây, phút, giờ, ngày, tháng, xác lập báo thức...

Ví dụ 2: Chuyển dữ liệu từ song song thành nối tiếp

Một luồng dữ liệu song song (xuất hiện đồng thời) thành nối tiếp (xuất hiện tuần tự) để thực hiện dùng MUX như hình 5-41.

Một luồng dữ liệu số song song (mọi bit xuất hiện đồng thời) có ưu thế tốc độ xử lý nhanh nhưng khi truyền ở cự ly xa, sẽ tốn nhiều đường truyền, nên luồng song song đó được chuyển thành nối tiếp. Ví dụ luồng song song $x_7x_6x_5x_4x_3x_2x_1x_0$ = 11101001 sẽ chuyển thành nối tiếp; đầu tiên tín hiệu của x_0 được đưa đến đầu ra y, tiếp đến x_1, tiếp đến x_2, ... cho đến x_7 sau cùng, sau đó quá trình lặp lại....



Hình 5-41

II. Mạch phân kênh

1. Khái niệm

+ Phân kênh (demultiplexer — DeMUX), hoạt động ngược với dồn kênh- có khả năng tại một thời điểm tách được một tín hiệu từ trong nhiều tín hiệu ở một đầu vào cho từng đầu ra riêng biệt.

+ Bộ phân kênh số (Digital Demultiplexer converter) là một thiết bị số làm nhiệm vụ: tại một thời điểm tách ra một tín hiệu, từ 2^n tín hiệu ở một đầu vào cho đầu ra tương ứng theo sự điều khiển của địa chỉ.

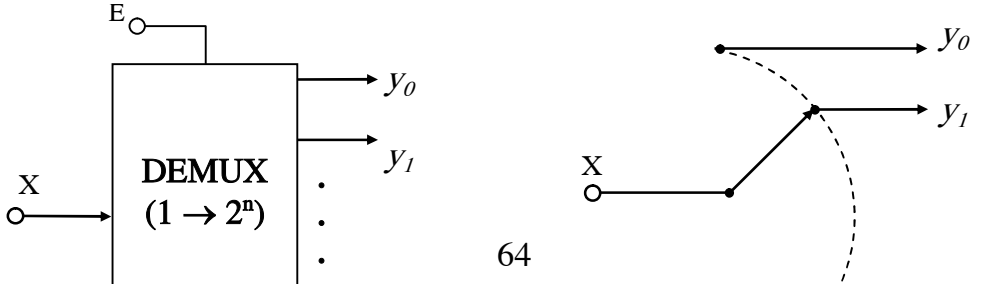
Mô hình phân kênh số (hình 5-42).

X: đầu vào (chứa 2^n tín hiệu).

A_0A_1, ... A_{n-1}: các đầu vào điều khiển (địa chỉ) có *n* đầu địa chỉ.

E (Enble): đầu vào chọn mạch (chọn chíp).

$y_0, y_1, \cdots, y_{2^n-1}$: đầu ra (có 2^n đầu ra).



+ Dồn kênh cũng được xem như một đảo mạch (chuyển mạch điện tử) nhiều vị trí.

Phương trình các hàm ra:

$$y_0 = X \cdot \overline{A}_{n-1} \cdots \overline{A}_1 \overline{A}_0$$

$$y_1 = X \cdot \overline{A}_{n-1} \cdots \overline{A}_1 A_0$$

.....

$$y_{2^n-1} = X \cdot A_{n-1} \cdots A_1 A_0$$

2. Mạch phân kênh 8 đầu ra

Mô hình của mạch có 1 đầu vào, 8 đầu ra. Để địa chỉ hoá cho 8 đầu ra (8 kênh) phải dùng 3 bit địa chỉ: A_2, A_1, A_0 tạo thành 8 tổ hợp địa chỉ:

$\overline{A}_2 \overline{A}_1 \overline{A}_0$, $\overline{A}_2 \overline{A}_1 A_0$, $\overline{A}_2 A_1 \overline{A}_0$, $\overline{A}_2 A_1 A_0$, $A_2 \overline{A}_1 \overline{A}_0$, $A_2 \overline{A}_1 A_0$, $A_2 A_1 \overline{A}_0$, $A_2A_1A_0$ để địa chỉ cho 8 đầu ra phương trình của các hàm ra:

$$y_0 = X \cdot \overline{A}_2 \overline{A}_1 \overline{A}_0$$

$$y_1 = X \cdot \overline{A}_2 \overline{A}_1 A_0$$

$$y_2 = X \cdot \overline{A}_2 A_1 \overline{A}_0$$

$$y_3 = X \cdot \overline{A}_2 A_1 A_0$$

$$y_4 = X \cdot A_2 \overline{A}_1 \overline{A}_0$$

$$y_5 = X \cdot A_2 \overline{A}_1 A_0$$

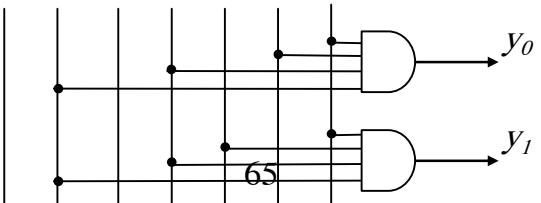
$$y_6 = X \cdot A_2 A_1 \overline{A}_0$$

$$y_7 = X \cdot A_2 A_1 A_0$$

Căn cứ vào phương trình, có sơ đồ logic hình 5-43 nguyên lý phân kênh l*à: ở mỗi thời điểm chỉ có tín hiệu của một kênh (trong số 2^n kênh chứa trong X) được đưa đến đầu ra (đầu thu) tương ứng theo sự điều khiển của địa chỉ.*

Ví dụ: với địa chỉ $\overline{A}_2 \overline{A}_1 \overline{A}_0$ = 000 tín hiệu x_0 (chứa trong X) được đưa đến đầu ra y_0, địa chỉ $\overline{A}_2 \overline{A}_1 A_0$ = 001, tín hiệu x_1 (chứa trong X) được đưa đến đầu ra y_1, ...

Mạch phân kênh ở hình 5-43 giống mạch giải mã 3:8 nếu ta chuyển các đầu địa chỉ (chọn lựa) A_2, A_1, A_0 của phân kênh thành đầu vào nhị phân x_0, x_1, x_2 cần giải mã, còn đầu vào X của phân kênh thành đầu chọn lựa E của giải mã thì phân kênh có thể chuyển thành giải mã. Vì lẽ đó các IC phân kênh sẽ kiêm luôn cả giải mã khi sử dụng với các mục đích khác nhau chỉ cần thay đổi cách sử dụng các đầu vào là được.



3. Ứng dụng của bộ phân kênh

a) Một số IC phân kênh thường gặp

- IC giải mã/phân kênh TTL 74139 gồm 2 bộ giải mã 2:4.
- IC giải mã/phân kênh 74138 chứa một bộ giải mã 3:8.
- IC giải mã/phân kênh 74154 chứa một bộ giải mã 4;16.

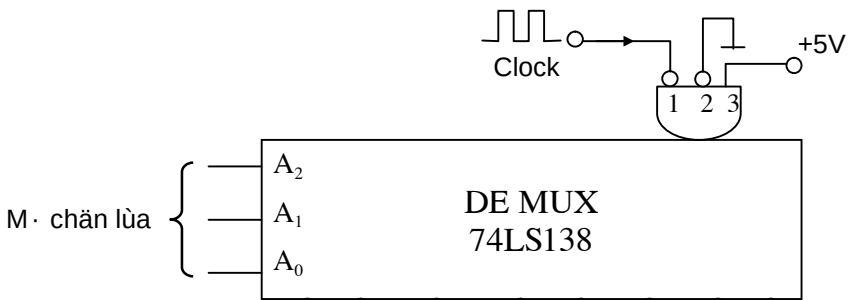
b) Ứng dụng

Ví dụ 1: Phân kênh bằng xung nhịp

Sử dụng Demux 74LS138 như hình 5-44.

74LS138 sử dụng với chức năng DEMUX. Dưới sự điều khiển của tín hiệu chọn lựa ($A_2A_1A_0$) tín hiệu xung nhịp được định tuyến đến đích dự kiến.

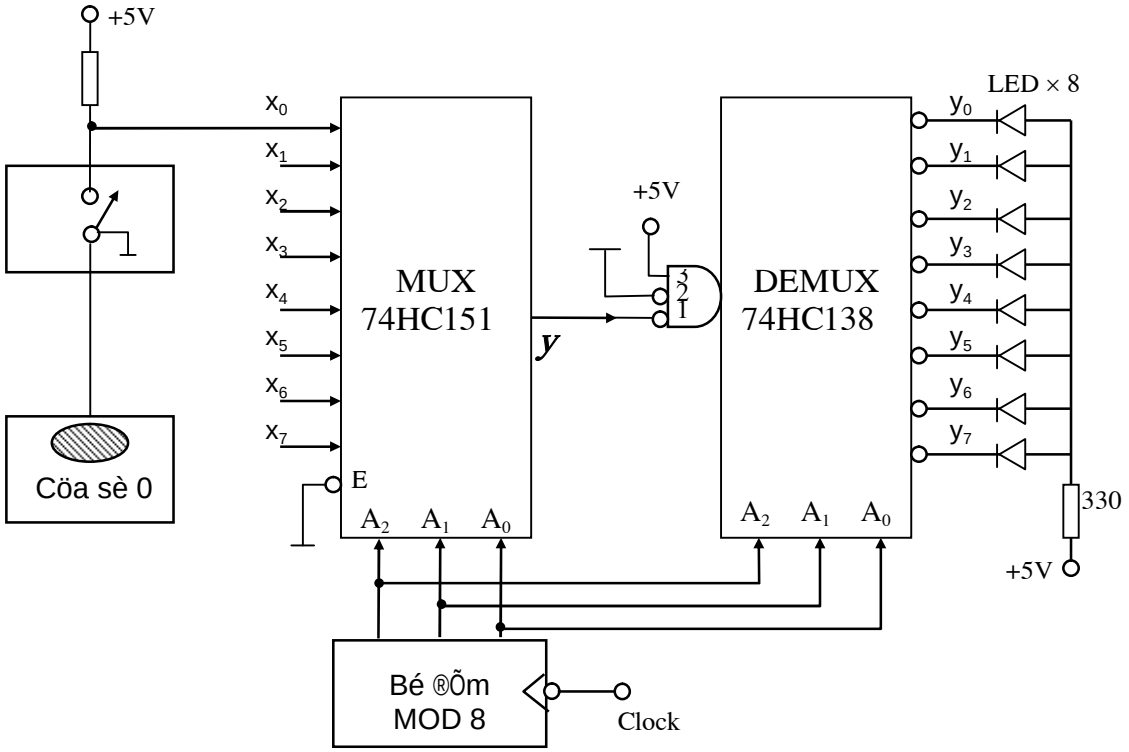
Ví dụ với $A_2A_1A_0 = 000$, tín hiệu xung nhịp được đưa đến đầu ra y_0 để đến bộ đếm; $A_2A_1A_0 = 001$, tín hiệu xung nhịp được đưa đến đầu ra y_1 để đến thanh ghi dịch.



Ví dụ 2: Hệ thống giám sát an ninh

Ví dụ cần giám sát an ninh trong một nhà máy, nơi giám sát là một trạng thái đóng/mở của nhiều cửa vào. Mỗi cửa điều khiển trạng thái của một công tắc và hiển thị trạng thái này ra LED gắn trên bảng theo dõi đặt ở phòng bảo vệ. Phương pháp đầu tiên là truyền một tín hiệu riêng biệt từ công tắc gắn ở mỗi cửa đến LED trên bảng theo dõi. Làm vậy phải cần đến nhiều dây nối, nếu xa càng phức tạp.

Phương pháp thứ 2 hiệu quả hơn sử dụng MUX và DEMUX như hình 5-45.



Hình 5-45

Sơ đồ điều khiển được 8 cửa (có thể mở rộng số cửa tuỳ ý). Nguyên lý làm việc như sau: Công tắc ở 8 cửa là đầu vào dữ liệu được đưa đến MUX, sinh mức cao khi cửa mở, mức thấp lúc cửa đóng. Bộ đếm MOD 8 cung cấp tín hiệu cho mạch đếm làm tín hiệu địa chỉ (chọn lựa cho cả MUX và DEMUX). Mỗi đầu ra của DEMUX được nối với một LED, LED sáng khi đầu ra DEMUX mức thấp. Tín hiệu địa chỉ sẽ lần lượt thay đổi từ 000 đến 111, tại mỗi số đếm trạng thái công tắc ở cửa có cùng số hiệu sẽ bị MUX nghịch đảo đưa đến đầu ra *y,* từ đó đưa đến DEMUX và truyền đến đầu ra cùng số hiệu.

Ví dụ bộ đếm đang đếm đến 110, lúc đó giả sử cửa sổ 6 đóng — mức thấp ở x_6 sẽ truyền đến MUX, bị đảo để sinh mức cao tại y truyền đến DEMUX đến đầu $\bar{y}_6$ làm cho LED_6 tắt, cho biết cửa số 6 đang đóng. Nếu có người mở cửa số 6, mức thấp xuất hiện tại y và $\bar{y}_6$ làm cho LED_6 sáng — báo cửa số 6 mở, lúc này các LED khác đều tắt vì ở thời điểm này chỉ có đầu $\bar{y}_6$ có mức tích cực thấp (0). Như vậy mạch đếm sẽ điều khiển quét một vòng lần lượt hiển thị trạng thái của 8 cửa, sau đó quét tiếp vòng 2 và vòng 3 ... nếu cả 8 cửa đều đóng thì không có LED nào sáng. Các cửa mở, 1 lần quét, một lần chớp sáng LED, ta có thể điều chỉnh tốc độ chớp tắt bằng cách thay đổi tần số xung nhịp cho bộ đếm.

Số đường dây sử dụng chỉ còn 4 (đáng lẽ 8) một đường cho đầu ra y nối từ MUX đến DEMUX và 3 đường cho địa chỉ $A_2A_1A_0$, tiết kiệm được 4 đường.

Ví dụ 3: Hệ thống truyền dữ liệu đồng bộ

Hình 5-46 là sơ đồ logic của một hệ thống truyền dữ liệu đồng bộ có nhiệm vụ truyền lần lượt 4 từ dữ liệu 4 bit từ trạm phát đến trạm thu ở xa. ë phía phát, dữ liệu được lưu ở các thanh ghi A,B,C,D các thanh ghi dịch quay vòng bằng đầu vào Shift (xung nhịp) chung. Mỗi thanh ghi sẽ dịch phải khi có sườn dương của xung dịch từ cổng 2. LSB của từng thanh ghi được nối sát với MUX

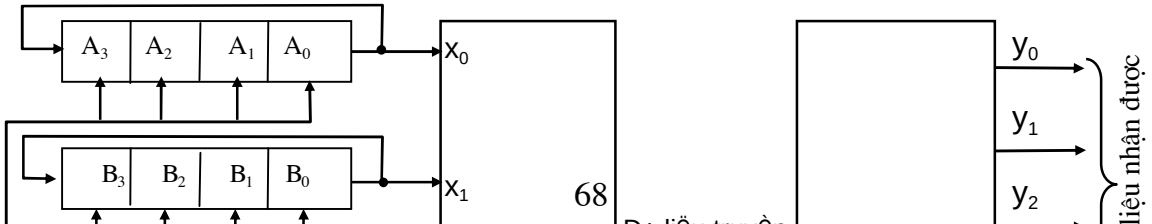
Hai bộ đếm MOD 4 điều khiển hoạt động truyền nội dung của thanh ghi dữ liệu đến đầu ra y của MUX. Bộ đếm từ (Word counter), chọn dữ liệu (từ thanh ghi) để chuyển đến Y; khi bộ đếm này đếm từ 00 đến 11, dữ liệu ở mỗi thanh ghi sẽ lần lượt xuất hiện tại y. Bộ đếm bit (bit counter) bảo đảm 4 bit dữ liệu từ mỗi thanh ghi sẽ được truyền qua MUX, trước khi chuyển sang thanh ghi kế tiếp. Bộ đếm bit đếm lên một số ứng với từng xung dịch, sau 4 xung dịch nó quay về 00. ë sườn xuống (sườn âm) của xung tại đầu Q_1 của bộ đếm bit, làm cho bộ đếm từ đếm lên số đếm kế tiếp để chọn thanh ghi truyền dữ liệu kế tiếp. Theo cách này nội dung mỗi thanh ghi dữ liệu sẽ được truyền đến Y, mỗi lần 1 bit, bắt đầu bằng thanh ghi A ($A_1A_0 = 00$), lần lượt qua từng thanh ghi, khi bộ đếm từ đếm lên một số, sau 4 xung dịch một. Tín hiệu ở Y chứa 16 bit dữ liệu nối tiếp (dữ liệu dồn kênh phân thời gian), có 4 tập hợp dữ liệu khác nhau xuất hiện ở cùng một đầu ra tại những thời điểm khác nhau.

Phía đầu thu là DEMUX (1:4), tiếp nhận tín hiệu từ y của MUX (phía phát) chuyển đến và phân kênh, nó tách thành 4 tập hợp dữ liệu khác nhau, phân phối cho 4 đầu ra của DEMUX.

- Dữ liệu từ thanh ghi A sẽ đến đầu ra y_0.
- Dữ liệu từ thanh ghi B sẽ đến đầu ra y_1.
- Dữ liệu từ thanh ghi C sẽ đến đầu ra y_2.
- Dữ liệu từ thanh ghi D sẽ đến đầu ra y_3.

Dữ liệu từ phát đến thu được gửi mỗi lần 1 từ trên một thanh ghi qua đường truyền nối tiếp.

Bộ đếm MOD_4 ở phía phát và thu có cùng tính năng. Bộ đếm từ quyết định đầu ra nào của bộ phân kênh sẽ nhận dữ liệu, còn bộ đếm bit cho phép 4 dữ liệu truyền đến mỗi đầu ra trước khi đẩy bộ đếm từ lên trạng thái kế tiếp.



Hình 5-46

Để dữ liệu chuyển từ phát đến thu hợp lý (đồng bộ) phải có phương tiện đồng bộ để chọn đầu ra của bộ dồn kênh (phía phát) với hoạt động chọn đầu ra của phân kênh (phía thu) ta xét ví dụ cụ thể dữ liệu ở các thanh ghi như sau:

[A] = 0110 [C] = 1011

[B] = 1001 [D] = 0100

Quá trình thể hiện ở dạng sóng hình 5-47.

1) FFW, M trong mạch phát và FFN ở mạch thu thường xuống mức thấp. Mức thấp từ M và N sẽ giữ cả 2 tập hợp bộ đếm ở trạng thái 0. Mức thấp tại W ngăn không cho xung nhịp qua AND_1.

2) Trước t_0 cả hai bộ đếm từ đầu đều ở trạng thái 00, thanh ghi A (phía phát) và đầu ra y_0 (phía thu) được chọn.

3) Tại t_0 xung truyền định W=1 làm cho AND_1 mở, xung nhịp đi qua đưa đến mạch phát và thu.

4) Tại t_1 xung nhịp đầu tiên qua AND_1 có sườn âm làm cho (M,N) chuyển sang mức cao (1) đồng thời cổng AND_2, AND_3 mở chuyển xung nhịp cho các mạch đếm phát và thu.

5) Tại t_1, t_2, t_3, t_4 xung dịch có sườn dương sẽ làm cho A_0, A_1, A_2, A_3, chuyển vào MUX, đến y vào DEMUX ra đầu y_0. Các xung dịch này được cả 2 bộ đếm bit (phát, thu) cùng đếm.

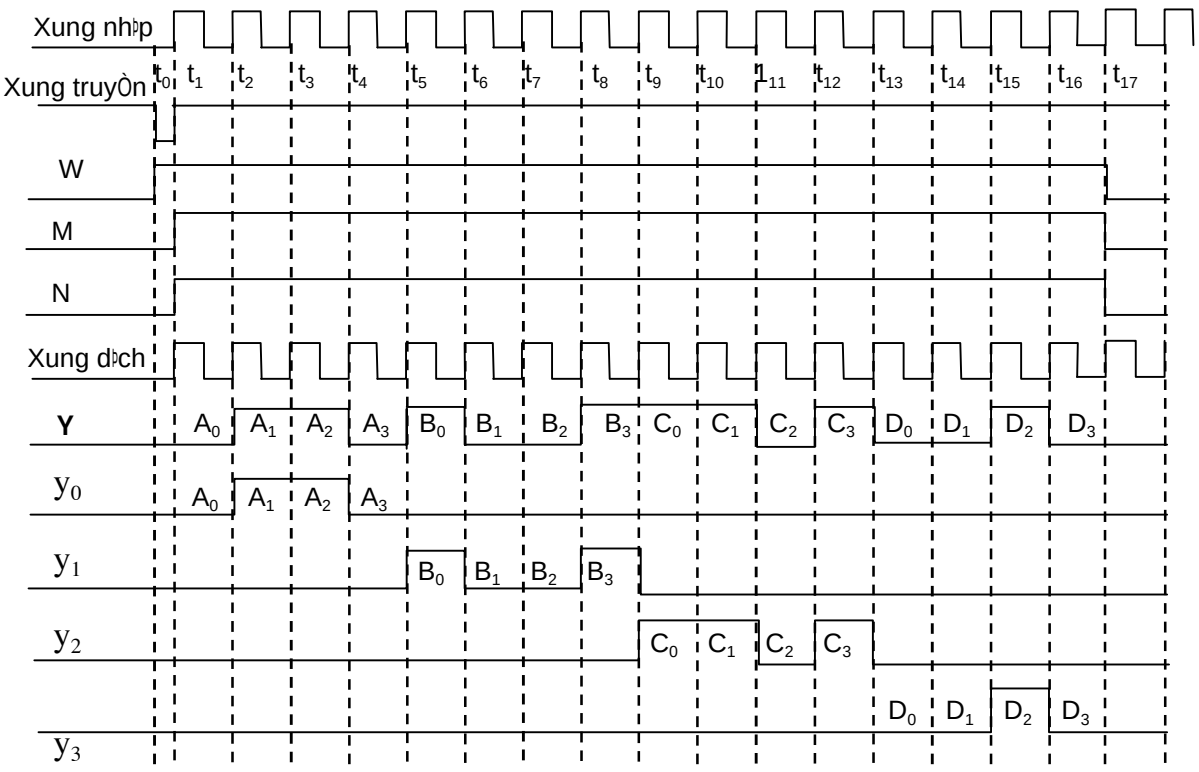
6) Sau t_4 tất cả các thanh ghi trở về ban đầu, trả bộ đếm bit về 00, tại sườn xuống (sườn âm) của xung nhịp thứ 4, Q_1 sẽ kích sang bộ đếm từ mạch phát và mạch thu từ 00 lên 01 để chọn tiếp đầu vào x_1 (phía phát) , đầu y_1 (phía thu). Do vậy mức cao tại B_0 sẽ chuyển đến MUX.

7) Xung dịch tại t_5, t_6, t_7, t_8 sẽ dịch $B_0B_1B_2B_3$ vào MUX và ra y_1, bộ đếm bit lại trở về 00 tăng bộ đếm từ lên 10 để chọn x_2, y_2 đặt mức cao từ C_0 đến MUX.

8) Xung dịch t_9, t_{10}, t_{11}, t_{12}, dịch C_0, C_1, C_2, C_3 vào MUX, ra y_2, bộ đếm bit trở về đầu vòng đếm (00), tăng bộ đếm từ lên 11, chọn x_3 và y_3 đặt mức thấp từ D_0 đến MUX.

9) Xung dịch $t_{13}, t_{14}, t_{15}, t_{16}$ sẽ dịch $D_0D_1D_2D_3$ vào MUX, qua DEMUX ra y_3, bộ đếm bit về lại đầu vòng đếm (00) tăng bộ đếm từ lên 00. Với sườn âm tại Q_1 của bộ đếm từ sẽ khởi động mạch đơn ổn (OS) tương ứng để sinh xung xoá hẹp FFW, M,N. Với 3FF ở mức thấp, mọi xung nhịp và xung dịch đều bị cấm, tất cả bộ đếm duy trì trạng thái 0.

10) Mạch quay về trạng thái ban đầu, không dữ liệu nào được truyền thêm cho đến khi xuất hiện xung truyền kế tiếp....



Hình 5-47

3.3 MẠCH SO SÁNH

I. Khái niệm

Trong các hệ thống số đặc biệt là trong máy tính thường thực hiện việc so sánh (Comparator) hai số, để biết số nào lớn hơn hay chúng bằng nhau (A>B; A<B; hay A = B). Hai số cần so sánh có thể là các số nhị phân, cũng có thể là các kí tự đã mã hoá nhị phân. Bộ so sánh có thể thiết lập theo kiểu nối tiếp hay song song. Trước tiên ta xét bộ so sánh 2 số nhị phân 1 bit, sau đó là các bộ so sánh 2 số nhiều bit.

II. So sánh 2 số nhị phân 1 bit.

Giả sử 2 số nhị phân A và B đều là 1 bit, quá trình so sánh có thể xảy ra 1 trong 3 khả năng: A = B; A< B; A > B; ứng với trường hợp nào thì đầu ra y đó sẽ có mức 1. Sự so sánh thể hiện ở bảng 5-1.

Căn cứ vào bảng 5-1, ta có phương trình sau:

$$y_1 = \overline{A}\cdot\overline{B} + AB = \overline{A \oplus B}$$

$$y_2 = \overline{A}B$$

$$y_3 = A\overline{B}$$

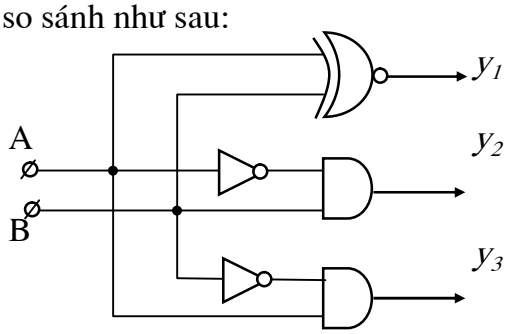
Từ các phương trình y_1, y_2, y_3 ta có sơ đồ lô gic sau (hình 5-1)

Bảng 5-1

A	B	y_1(A=B)	y_2(A<B)	y_3 (AB)
0	0	1	0	0
0	1	0	1	0
1	0	0	0	1
1	1	1	0	0

Nếu chỉ cần so sánh đơn thuần A = B, ta cần dùng mạch XOR hoặc XNOR (dùng XOR đầu ra bằng 0 khi A = B, dùng XNOR đầu ra bằng 1 khi A=B).

Căn cứ kết quả đầu ra ta biết được A = B hay A ≠B, với sơ đồ hình 5-1 (dùng XNOR và mạch AND) ta có kết quả so sánh như sau:



Hình 5-1

- Khi A = B = 0 (hoặc A = B = 1), đầu ra y_1 = 1.
- Khi A = 0; B = 1 đầu ra $y_2 = 1$ (thể hiện A <B).
- Khi A = 1; B = 0 đầu ra $y_3 = 1$ thể hiện A > B)

Ở mỗi thời điểm chỉ có 1 đầu ra có mức tích cực 1 thể hiện kết quả so sánh.

III. So sánh 2 số nhiều bit

Giả sử có 2 số nhị phân (*n bit)* A và B.

$$A = a_n\, a_{n-1} \ldots\ldots\ a_1.$$

$$B = b_n\, b_{n-1} \ldots\ldots\ b_1$$

Trong đó: a_n; b_n *:* bit có trọng số lớn nhất.

$a_1 b_1$: bit có trọng số nhỏ nhất.

Quá trình so sánh hai số nhị phân nhiều bít phải bắt đầu từ bít có trọng số lớn nhất, chỉ khi nào bít có trọng số lớn nhất bằng nhau thì mới tiếp tục so sánh đến bít có trọng số thấp hơn liền kề; quá trình đó xảy ra cho đến bit có trọng số bé nhất; bởi vì kết quả so sánh hai số A và B được quyết định theo các cặp bít có trọng số lớn nhất

cho đến các cặp bít có trọng số bé nhất (ở đây ta đang xét hệ đếm 2 là hệ đếm có trọng số phụ thuộc vị trí).

Quá trình so sánh cũng xảy ra 3 trường hợp.

A = B; A<B; A > B.

Mô hình như hình 5-2.

Hình 5-2

Để xây dựng sơ đồ logic mạch so sánh có thể dùng 2 cách:

- Xây dựng trực tiếp từ các hàm: y_1 *(A = B)*
  y_2, *(A<B)*
  y_3 *(A>B)*

Phương pháp này thực chất là xây dựng 1 hệ 3 hàm logic, mỗi hàm có 2^n biến.

- Xây dựng sơ đồ gián tiếp qua các bộ so sánh 1 bit, sơ đồ được ghép nối từ các bộ so sánh 1 bit.

1. Xây dựng mạch so sánh trực tiếp từ các hàm

Ví dụ: xây dựng mạch so sánh 2 số A,B là các số 2 bit:

$$A = a_2a_1; \quad B = b_2b_1;$$

Tổng hợp 2 số 2 bít có 16 khả năng của tổ hợp biến (2 số 2 bít thành 4 bit).

a) Trường hợp A = B

Ta có sự so sánh như bảng 5-2 (có 4 trường hợp bằng nhau của 2 số A,B).

Cả 4 trường hợp cho 2 bít của mỗi số đều phải có $a_2 = b_2$; $a_1 = b_1$ mới có được A = B và đầu ra y_1 trong cả 4 trường hợp đều bằng 1.

Từ bảng 5-2 ta có phương trình y_1 như sau:

$$y_1 = \bar{a}_2\bar{b}_2 \cdot \bar{a}_1\bar{b}_1 + \bar{a}_2\bar{b}_2\, a_1b_1 + a_2b_2\bar{a}_1\bar{b}_1 + a_2b_2a_1b_1$$

$$= \bar{a}_2\bar{b}_2\left(\bar{a}_1\bar{b}_1 + a_1b_1\right) + a_2b_2\left(\bar{a}_1\bar{b}_1 + a_1b_1\right) = \left(\bar{a}_1\bar{b}_1 + a_1b_1\right)\cdot\left(\bar{a}_2\bar{b}_2 + a_2b_2\right)$$

$$y_1 = \overline{\left(a_1 \oplus b_1\right)}\cdot\overline{\left(a_2 \oplus b_2\right)}$$

b) Trường hợp A > B

Trường hợp A > B sẽ có 6 khả năng trong tổng số 16 khả năng, ta có bảng 5-3.

Bảng 5-2

A		B		y_1
a_2	a_1	b_2	b_1	(A=B)
0	0	0	0	1
0	1	0	1	1
1	0	1	0	1
1	1	1	1	1

Bảng 5-3

A		B		y_3
a_2	a_1	b_2	b_1	(A>B)
0	1	0	0	1
1	0	0	0	1
1	0	0	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1

Từ bảng 5-3 ta có phương trình y_3 như sau:

$$y_3 = \bar{a}_2\,\bar{b}_2\,a_1\,\bar{b}_1 + a_2\,\bar{b}_2\,\bar{a}_1\,\bar{b}_1 + a_2\,\bar{b}_2\,\bar{a}_1\,b_1 + a_2\,\bar{b}_2\,a_1\,\bar{b}_1 + a_2\,\bar{b}_2\,a_1\,b_1 + a_2\,b_2\,a_1\,\bar{b}_1$$

Nhóm số hạng đầu, cuối và 4 số hạng giữa ta có:

$$y_3 = a_1\,\bar{b}_1\left(\bar{a}_2\,\bar{b}_2 + a_2\,b_2\right) + a_2\,\bar{b}_2\left(\bar{a}_1\,\bar{b}_1 + a_1\,b_1 + \bar{a}_1\,b_1 + a_1\,\bar{b}_1\right)$$

$$= a_1\,\bar{b}_1\left(\overline{a_2 \oplus b_2}\right) + a_2\,\bar{b}_2\left[\left(\overline{a_1 \oplus b_1}\right) + \left(a_1 \oplus b_1\right)\right]$$

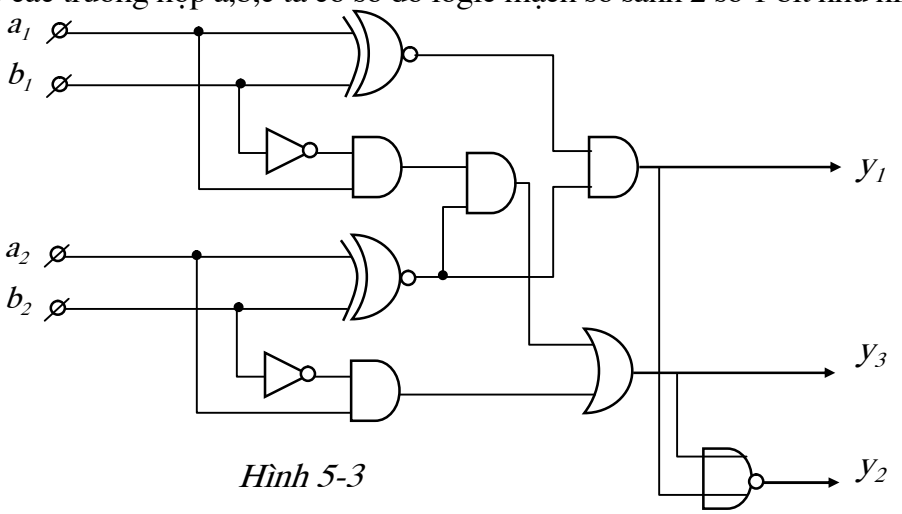
Vì $\left(\overline{a_1 \oplus b_1}\right) + \left(a_1 \oplus b_1\right) = 1$

Cuối cùng: $y_3 = a_1\,\bar{b}_1\left(\overline{a_2 \oplus b_2}\right) + a_2\,\bar{b}_2$

c) Trường hợp A < B

Mạch so sánh là mạch logic, nếu đã có A = B hoặc A > B thì sẽ không có A < B; vậy A < B, nếu không phải là A ≥ B hay : $y_{2_{(A<B)}} = \overline{y_1}.\overline{y_2} = \overline{y_1 + y_2}$

Kết hợp các trường hợp a,b,c ta có sơ đồ logic mạch so sánh 2 số 1 bit như hình 5-3.



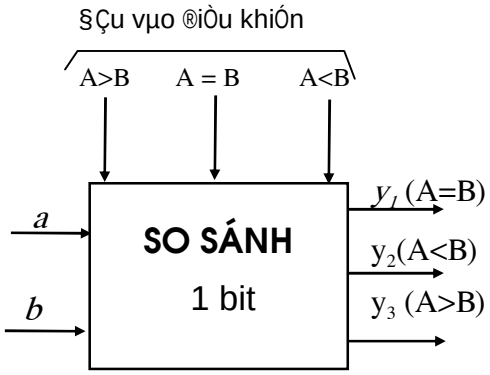
Hình 5-3

Bằng cách tương tự ta có thể xây dựng các mạch so sánh 2 số nhiều bít trực tiếp từ các hàm.

2. Xây dựng mạch so sánh nhiều bit từ các mạch so sánh 1 bit

**Mạch so sánh 1 bit*

Mạch so sánh 1 bit ngoài các đầu vào tiếp nhận các bít so sánh *a,b* và các đầu ra thể hiện: A=B; A < B; A>B; còn có các đầu vào điều khiển : A>B; A=B; A<B;. Mô hình so sánh 1 bit (hình 5-4) và bảng trạng thái 5-4.



Hình 5-4

Bảng 5-4:

Đầu vào điều khiển			Đầu vào		Đầu ra		
A=B	A<B	A>B	a	b	A=B	A<B	A>B
0	1	0	×	×	0	1	0
0	0	1	×	×	0	0	1
1	0	0	1	0	0	0	1
1	0	0	0	1	0	1	0
1	0	0	0	0	1	0	0
1	0	0	1	1	1	0	0

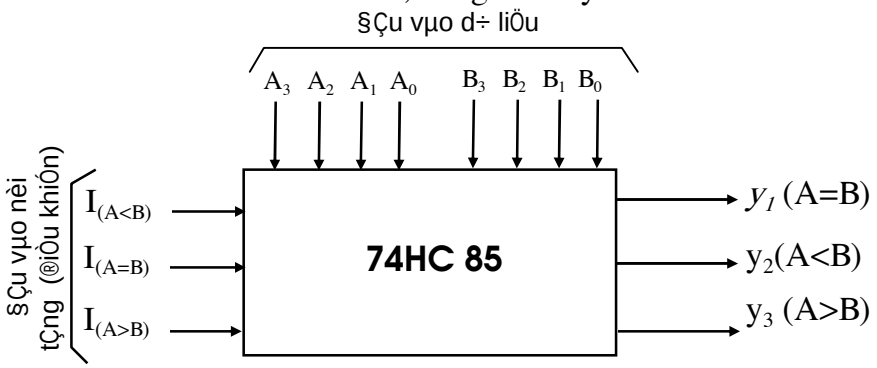
Bảng 5-4 cho thấy: khi đầu vào *a,b* chưa có tín hiệu vào để so sánh (2 hàng đầu) thì tín hiệu ra phụ thuộc tín hiệu điều khiển, khi có tín hiệu vào a,b (4 hàng cuối) lúc này đầu vào điều khiển được ghim: đầu A>B và A<B mức 0 đầu A = B mức 1; tín hiệu ra phụ thuộc tín hiệu vào ở *a,b.*

Trên cơ sở mạch so sánh 1 bit, người ta có thể ghép các so sánh 1 bit theo 2 dạng nối tiếp và song song để tạo thành các mạch so sánh nhiều bit.

IV. Vi mạch so sánh và ứng dụng

Một IC so sánh 4 bit không dấu tương đối phổ biến là 74HC85 và cùng xê ri với nó còn có 7485; 74LS85.

74HC85 có sơ đồ khối hình 5-7; bảng chân lý 5-5.



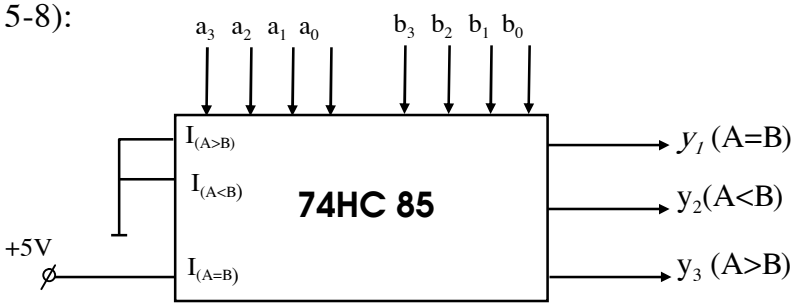
Hình 5-7

Bảng 5-5:

Đầu vào so sánh				Đầu vào nối tầng			Đầu ra		
a_3b_3	a_2b_2	a_1b_1	a_0b_0	$I_{(A=B)}$	$I_{(A<B)}$	$I_{(A>B)}$	$y_{1(A=B)}$	$y_{2(A<B)}$	$y_{3(A>B)}$
$a_3 > b_3$	×	×	×	×	×	×	0	0	1
$a_3 < b_3$	×	×	×	×	×	×	0	1	0
$a_3 = b_3$	$a_2 > b_2$	×	×	×	×	×	0	0	1
$a_3 = b_3$	$a_2 < b_2$	×	×	×	×	×	0	1	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 > b_1$	×	×	×	×	0	0	1
$a_3 = b_3$	$a_2 = b_2$	$a_1 < b_1$	×	×	×	×	0	1	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 > b_0$	×	×	×	0	0	1
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 < b_0$	×	×	×	0	1	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 = b_0$	0	0	1	0	0	1
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 = b_0$	0	1	0	0	1	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 = b_0$	1	0	0	1	0	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 = b_0$	0	0	0	0	1	1
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 = b_0$	0	1	1	0	0	0

+ Đầu vào dữ liệu: $a_3a_2a_1a_0$ — tiếp nhận số A.
  $b_3b_2b_1b_0$ — tiếp nhận số B.
+ Đầu ra: $y_{1(A=B)}$ mức 1 khi A = B.
  $y_{2(A<B)}$ mức 1 khi A < B.
  $y_{3(A=>B)}$ mức 1 khi A > B.

+ Đầu nối tầng (hình 5-8):



Hình 5-8

Với cách nối $I_{(A>B)}$; $I_{(A<B)}$ và $I_{(A=B)}$ như vậy làm cho các đầu điều khiển:
A > B và A<B mức 0; còn đầu A = B mức 1.

3.4 MẠCH TÍNH TOÁN

I. Đặt vấn đề

Trong kỹ thuật số, kỹ thuật vi xử lý và máy tính, tính toán để xử lý thông tin là một vấn đề lớn. Các phép tính được phân thành hai dạng: Các phép tính số học và các phép tính logic. Trong chương 1 đã đề cập nó với tư cách toán học. Với bài này ta sẽ xét các mạch logic để thực hiện các phép toán đó.

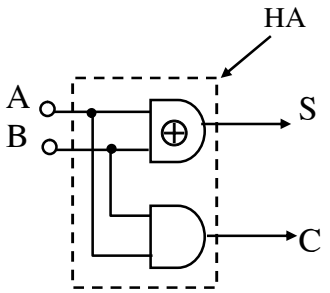
II. Mạch số học cơ bản

1. Mạch bán tổng

Khi thực hiện cộng các số nhị phân, bao giờ cũng cộng từ cột có trọng số nhỏ nhất cho đến cột có trọng số lớn nhất, đồng thời phép cộng thường có nhớ từ cột có trọng số thấp lên cột có trọng số cao liền kề. Ví dụ cộng hai số nhị phân 1 bit, ta có bảng cộng như hình 7- 38a.

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

a)



b)

Hình 7- 38

Căn cứ vào bảng 7-38a ta có ngay kết quả:

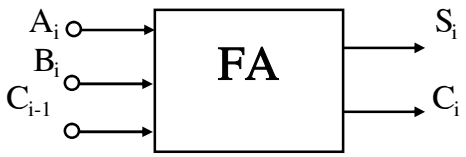
$S = A \oplus B$.
$C = A.B$

A, B là các số nhị phân 1 bit.
S : Là kết quả tổng.
C: Là số nhớ.

Sơ đồ logic như hình 7- 38b. Mạch này ta gọi mạch bán tổng (HA-Half Adder). Sở dĩ gọi là bán tổng vì mạch chưa có đầu vào để tiếp nhận số nhớ từ trước chuyển sang.

2. Mạch tổng đầy đủ

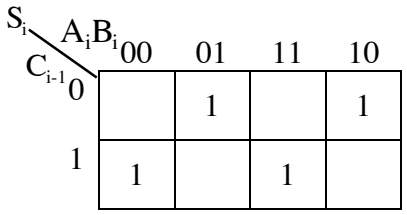
Mạch này có mô hình như hình 7-39.



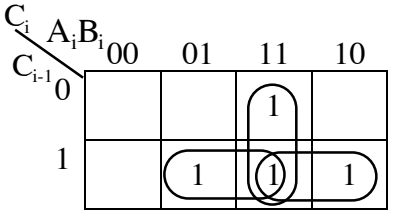
a)

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

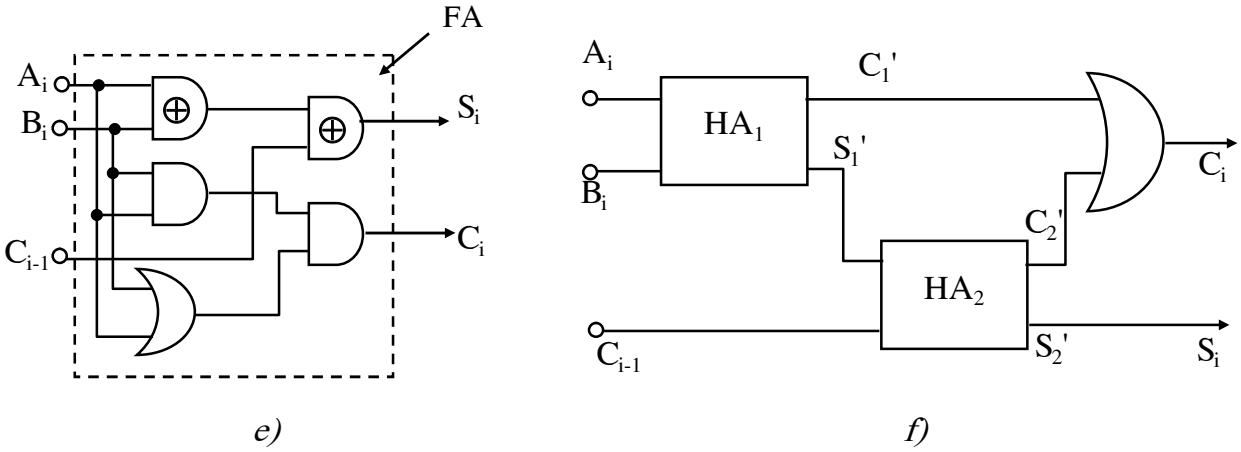
b)



c)



d)



e) *f)*

Hình 7- 39

Mạch phải có thêm đầu C_{i-1} để tiếp nhận số nhớ từ cột có trọng số thấp hơn liền kề chuyển đến. Bảng cộng như hình 7-39b, để lập phương trình cho S_i và C_i , dùng bảng Karnaugh như hình 7-39c,d, rút gọn có kết quả như sau:

$$\begin{aligned} C_i &= A_iB_i + C_{i-1}A_i + C_{i-1}B_i \\ &= A_iB_i + C_{i-1}(A_i+B_i) \\ S_i &= \overline{C}_{i-1}\left(\overline{A}_iB_i + A_i\overline{B}_i\right) + C_{i-1}\left(\overline{A}_i\overline{B}_i + A_iB_i\right) \\ &= \overline{C}_{i-1}\left(A_i \oplus B_i\right) + C_{i-1}\left(\overline{A_i \oplus B_i}\right) \\ &= A_i \oplus B_i \oplus C_{i-1} \end{aligned}$$

Từ các phương trình S_i và C_i , sơ đồ logic thực hiện như hình 7-39e. Và sơ đồ này gọi là bộ cộng đầy đủ (FA-Full-Adder).

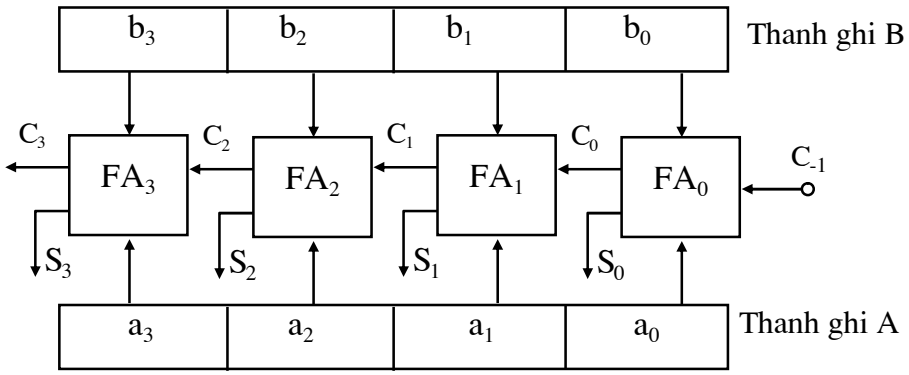
Bộ Tổng đầy đủ có thể được ghép từ hai bộ bán tổng như hình 7-39f, bạn đọc tự chứng minh sơ đồ này, cách làm này thường được áp dụng trong thực tế.

III. Bộ cộng song song

1. Bộ cộng song song 4 bit

Để thực hiện cộng hai số nhiều bit theo phương pháp song song, ta sử dung dụng các mạch cộng đầy đủ 1 bit (FA)

Ví dụ để cộng hai số 4 bit, sử dụng 4 FA ghép theo sơ đồ như hình 7-40.



Hình 7-40

Giả sử số: $A = a_3a_2a_1a_0$

$B = b_3b_2b_1b_0$.

Với a_0 , b_0 là cột có trọng số bé nhất, a_3, b_3 là cột có trọng số lớn nhất. Hai số được chứa trong 2 thanh ghi A và B. Cặp có trọng số bé nhất a_0b_0 được đưa vào FA_0, cặp a_1b_1 vào FA_1, cặp a_2b_2 vào FA_2, cặp a_3b_3 được đưa vào FA_3, 4 cặp nhị phân được đưa vào 4FA một cách đồng thời. Kết quả tổng cho 4FA được đưa ra các đầu $S_0S_1S_2S_3$, kết quả này

có thể được chứa trên một thanh ghi khác, cũng có thể sử dụng luôn thanh ghi A làm thanh tổng. Số nhớ được chuyển liên tiếp - đầu C_{-1} để tiếp nhận số nhớ từ lần cộng trước chuyển tới, sau đó số nhớ được chuyển tiếp một cách liên tiếp qua FA_0; FA_1, FA_2,FA_3, đầu C_3 để đưa số nhớ đến lần cộng tiếp sau.

Ví dụ:

A	=		1	1	1	1
B	=		1	0	0	1
C	=	1	1	1	1	
		C_3	C_2	C_1	C_0	C_{-1}
Σ	=	1	1	0	0	0

Với sơ đồ này các cặp bit được đưa đồng thời vào 4FA sẽ nhanh hơn phương pháp đưa vào tuần tự. Tuy nhiên số nhớ lại thực hiện theo kiểu tuần tự, tốc độ sẽ chậm do truyền và thực hiện cộng số nhớ. Để khắc phục người ta dùng thêm mạch logic kiểm tra trước bit nhớ của nhóm bit thấp $a_1 a_0$ và $b_1 b_0$ liệu có tồn tại số nhớ C_1 hay không ? nếu có thì số nhớ này xuất hiện sớm hơn bằng con đường truyền khác với thời gian chậm trễ nhỏ hơn nhiều khi đi theo con đường truyền $FA_0 \Rightarrow FA_1 \Rightarrow C_2$.

2. Bộ cộng nhớ nhanh (Fast Carry)

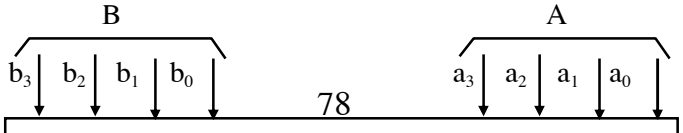
Để khắc phục sự chậm trễ do việc truyền số nhớ một cách tuần tự, người ta sử dụng bộ cộng nhớ nhanh . Với cách thực hiện mạch nhằm khắc phục sự truyền chậm trễ của số nhớ.

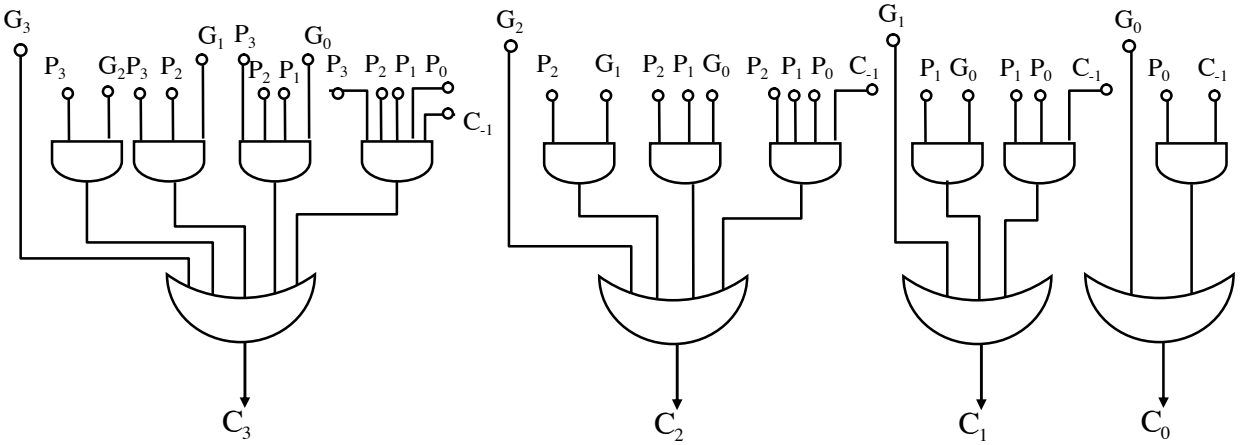
Ta đã xét bộ cộng FA một cột số, dựa vào bảng 7-39b, tiến hành tối thiểu hóa C_i theo một phương án khác, ta có bảng bên:

C_i \ A_iB_i	00	01	11	10
C_{i-1} = 0			1	
C_{i-1} = 1		(1)	1	(1)

Ta có:

$$\left.\begin{aligned} C_i &= A_i B_i + C_{i-1}\left(\overline{A}_i B_i + A_i \overline{B}_i\right) \\ &= A_i B_i + C_{i-1}\left(A_i \oplus B_{i\,i}\right) \end{aligned}\right\} \qquad (7\text{-}1)$$





Đã có : $S_i = A_i \oplus B_i \oplus C_{i-1}$

Đặt:
$$\left.\begin{aligned} A_i \oplus B_i &= p_i \\ A_i B_i &= G_i \end{aligned}\right\} \qquad (7\text{-}2)$$

Thay (7-2) vào (7-1) ta có:

$C_i = G_i + p_i C_{i-1}$

$S_i = p_i \oplus C_{i-1}$

Đối với bộ cộng 4 bit ta có:

$C_0 = p_0 C_{-1} + G_0$; $\qquad C_1 = p_1 C_0 + G_1$

$C_2 = p_2 C_1 + G_2$

$C_3 = p_3 C_2 + G_3$

Thay lần lượt :C_0 vào C_1, C_1 vào C_2, C_2 vào C_3, ta được:

$$\left.\begin{aligned} C_1 &= p_1 p_0 C_{-1} + p_1 G_0 + G_1 \\ C_2 &= p_2 p_1 p_0 C_{-1} + p_2 p_1 G_0 + p_2 G_1 + G_2 \\ C_3 &= p_3 p_2 p_1 p_0 C_{-1} + p_3 p_2 p_1 G_0 + p_3 p_2 G_1 + p_3 G_2 + G_3 \end{aligned}\right\} \qquad (7\text{-}3)$$

Các chữ số của tổng sẽ có giá trị sau:

$$\left.\begin{aligned} S_0 &= p_0 \oplus C_{-1} \\ S_1 &= p_1 \oplus C_0 \\ S_2 &= p_2 \oplus C_1 \\ S_3 &= p_3 \oplus C_2 \end{aligned}\right\} \qquad (7\text{-}4)$$

Dùng các hệ phương trình (7-3) và (7-4) để xây dựng bộ cộng song song nhớ nhanh gồm 3 khối như sơ đồ hình 7-41a.

Sơ đồ logic chi tiết của các khối này thể hiện ở hình 7-41b,c,,d.

CÂU HỎI VÀ BÀI TẬP

Câu 1: Khi cần so sánh 2 số nhị phân 16 bit, phải dùng mấy IC7485. Hãy mô tả và thực hiện ghép các IC đã chọn để thực hiện. Hãy áp dụng để so sánh 2 số nhị phân sau:

A = 1001 0010 1101 1110

B = 1001 0010 1011 0101

Câu 2: Phân biệt các trường hợp mã hoá

+ Mã hoá nhị phân.
+ Mã hoá ưu tiên.
+ Mã hoá nhị — thập phân (BCD)

Câu 3: Bộ mã hoá và bộ giải mã khác nhau như thế nào?

Câu 4: Phân biệt các trường hợp giải mã

+ Giải mã nhị phân.
+ Giải mã BCD — thập phân.
+ Giải mã BCD — 7 đoạn.

Câu 5: Hãy thiết kế bộ giải mã, nhị phân 4 bit?

Câu 6: Bộ dồn kênh có nhiệm vụ gì ? chức năng của đầu chọn lựa (địa chỉ) của nó.

- Có 4 dữ liệu, mỗi dữ liệu 8 bit, cần cho qua MUX phải làm thế nào để tại những thời điểm nhất định tại cổng ra của MUX sẽ xuất hiện chỉ một trong 4 dữ liệu này. Cấu trúc sẽ gồm bao nhiêu đầu vào dữ liệu? bao nhiêu đầu ra và bao nhiêu đầu vào chọn (địa chỉ).

Câu 7: Hãy giải thích điểm khác biệt giữa MUX và DEMUX mạch của DEMUX giống hệt mạch giải mã "đúng hay sai.

- Câu 8:

+ Bộ giải mã 7442 không có đầu vào cho phép E, sử dụng nó như một bộ giải mã 1:8 được không? (nếu không dùng $\overline{y}_8, \overline{y}_9$) có thể dùng đầu dữ liệu D làm đầu cho phép được không?

+ Khi dùng 7442 làm DEMUX 1:8 thì phải sử dụng các đầu vào dữ liệu và đầu vào chọn như thế nào?