Tai Lieu Mach to Hop

Kỹ Thuật Số (Trường Đại học Công nghiệp Hà Nội)

MỤC LỤC

HỆ LOGIC TỔ HỢP

4.1. KHÁI NIỆM CHUNG

Trong các thiết bị số, sự biến đổi và gia công tin tức được thực hiện nhờ các hệ thống mạch logic tổ hợp mà chúng có một số chức năng sau:

+ Trong các khâu biến đổi tin tức các hệ logic tổ hợp có thể thực hiện chức năng chuyển đổi từ hệ mã này sang hệ mã khác.

+ Trong các khâu gia công tin tức các hệ logic tổ hợp có thể thực hiện các chức năng như tạo hàm, giải mã, chọn kênh, phân kênh hoặc thực hiện các phép tính số học trên cơ sở các mã số khác nhau...

Trong thực tế các hệ logic tổ hợp thường được sử dụng phối hợp với các thiết bị tương tự để tạo ra các hệ nhiều chức năng.

Như vậy thế nào là hệ logic tổ hợp?

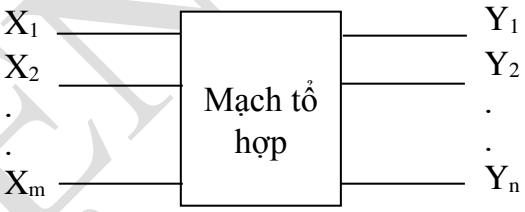
Định nghĩa: Hệ logic tổ hợp hay mạch tổ hợp là mạch mà tín hiệu ra chỉ phụ thuộc vào tín hiệu vào (hệ logic không nhớ).

$$Y_j = f(X_i)$$

$$(j = 1 \div n)$$

$$(i = 1 \div m)$$

Mô hình toán học tổng quát của mạch tổ hợp được minh hoạ trong hình 4.1.



Hình 4.1. *Mô hình tổng quát của mạch tổ hợp*

Mạch có m đầu vào và n đầu ra và được mô tả bởi hệ n phương trình đại số như sau:

$$Y_j = f_j(X_1, X_2, ..., X_m) \qquad \text{Với } j = 1 \div n$$

Hệ n phương trình mô tả hệ tổ hợp có thể là hệ phương trình xác định không đầy đủ, nghĩa là có ít nhất một trạng thái vào tại đó giá trị của hàm số không xác định, tương ứng với trạng thái vào không bao giờ xảy ra trong thực tế. Khi đó với trạng thái vào này ta có thể chọn giá trị tại hàm ra một cách tuỳ ý sao cho sơ đồ thực hiện là đơn giản nhất.

4.2. PHÂN TÍCH MẠCH LOGIC TỔ HỢP

Bài toán phân tích: Là bài toán từ sơ đồ logic cho trước viết hàm logic của các đầu ra theo các đầu vào và nếu cần thì còn phải chỉ ra dạng xung của tín hiệu ra tương ứng với tín hiệu vào, xác định giá trị tín hiệu ở từng thời điểm trên sơ đồ.

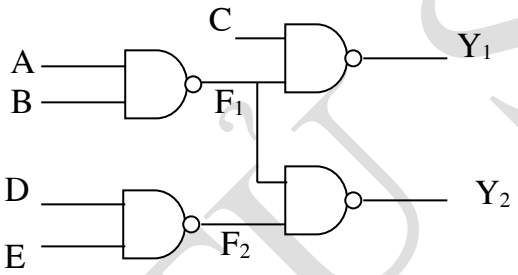
Các bước phân tích

Bước 1: Đặt các biến phụ vào mỗi đầu ra của các cổng logic.

Bước 2: Viết phương trình cho các bước phụ đó (viết lần lượt từ đầu vào cho đến đầu ra).

Bước 3: Ở biểu thức cuối cùng thay thế các giá trị tương ứng để rút ra được hàm logic cho các đầu ra của sơ đồ đã cho theo các biến logic đầu vào.

Ví dụ 1: Phân tích mạch tổ hợp ở hình 4.2.



Hình 4.2. *Mạch logic ví dụ 1*

Giải :

Cần xác định: $Y_1 = f_1 (A, B, C, D, E)$

$Y_2 = f_2 (A, B, C, D, E)$

+ Đặt các biến phụ và viết phương trình cho các biến phụ

$$F_1 = \overline{A.B}$$

$$F_2 = \overline{D.E}$$

$$Y_1 = \overline{F_1 . C}$$

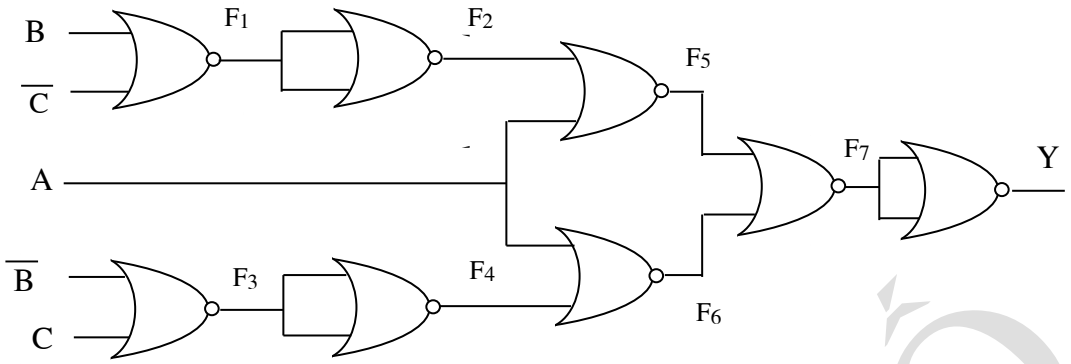
$$Y_2 = \overline{F_1.F_2}$$

+ Thay thế các giá trị tương ứng

$$Y_1 = \overline{F_1 . C} = \overline{\overline{A.B} . C} = AB + \overline{C}$$

$$Y_2 = \overline{F_1.F_2} = \overline{\overline{A.B} . \overline{D.E}} = AB + DE$$

Ví dụ 2: Phân tích mạch tổ hợp hình 4.3 và xác định khả năng có thể đơn giản hoá mạch (sử dụng các cổng NOR hai đầu vào).



Hình 4.3. *Mạch logic ví dụ 2*

Giải :

Cần xác định Y = f (A, B, C)

$F_1 = \overline{B + \overline{C}}$ $\quad F_4 = \overline{F_3}$

$F_2 = \overline{F_1}$ $\quad F_5 = \overline{F_2 + A}$

$F_3 = \overline{C + \overline{B}}$ $\quad F_6 = \overline{F_4 + A}$

$F_7 = \overline{F_5 + F_6}$

$Y = \overline{F_7}$

$= F_5 + F_6$

$= \overline{F_2 + A} + \overline{F_4 + A}$

$= \overline{B}.C.\overline{A} + B.\overline{C}.\overline{A}$

$= \overline{A}\,(\overline{B}.C + B.\overline{C})$

+ Đơn giản hoá mạch

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

$= \overline{\overline{A}\,(\overline{B}.C + B.\overline{C})}$

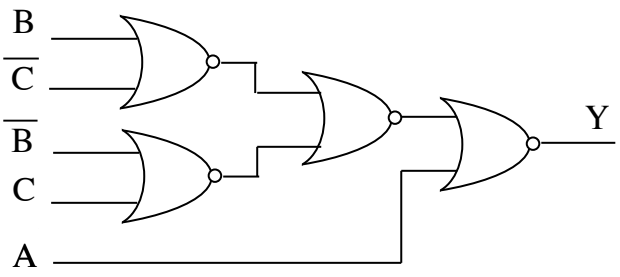
$= A + \overline{\overline{B}.C + B.\overline{C}}$

$= A + \overline{\overline{B}.C} + \overline{B.\overline{C}}$

$$= \overline{A + (B + \overline{C})(\overline{B} + C)}$$

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

Sơ đồ đơn giản hóa biểu diễn ở hình 4.4.



Hình 4.4. *Mạch logic ví dụ 2 rút gọn*

4.3. THIẾT KẾ MẠCH TỔ HỢP

4.3.1. Phương pháp thiết kế mạch tổ hợp

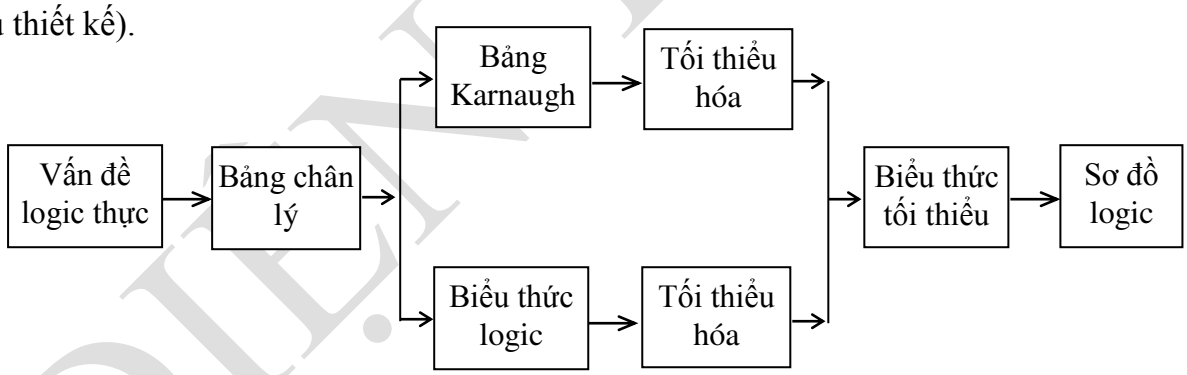
Phương pháp thiết kế mạch logic tổ hợp là các bước cơ bản để tìm ra sơ đồ mạch điện logic từ yêu cầu, nhiệm vụ logic đã cho (chức năng dạng sóng, tính năng kỹ thuật).

Khi thiết kế mạch dùng các phần tử rời rạc và các vi mạch cỡ nhỏ (SSI) thực hiện tuần tự theo các bước trên sơ đồ hình 4.5.

Quá trình thiết kế của mạch tổ hợp bao gồm 4 bước chính:

Bước 1: Phân tích yêu cầu.

Xác định được các biến số đầu vào, hàm số đầu ra và mối quan hệ logic giữa chúng với nhau (muốn phân tích đúng thì phải tìm hiểu xem xét một cách sâu sắc yêu cầu thiết kế).



Hình 4.5*. Sơ đồ khối các bước thiết kế mạch tổ hợp*

Bước 2: Lập bảng chân lý (bảng chức năng) nghĩa là liệt kê thành bảng về quan hệ tương ứng nhau giữa trạng thái tín hiệu đầu vào với trạng thái hàm số đầu ra. (Khi liệt kê bảng chức năng có thể không liệt kê các tổ hợp tín hiệu vào không thể có hay bị cấm, ở đầu ra trạng thái tương ứng được ghi dấu chéo (X), thường sử dụng các trạng thái này để tối thiểu hoá hàm logic).

Bước 3: Tối thiểu hoá. Dùng các phương pháp tối thiểu hoá khác nhau để đưa hàm về dạng đơn giản nhất.

Bước 4: Vẽ sơ đồ logic. Căn cứ việc chọn lựa loại cổng logic cụ thể, cần biến đổi biểu thức đó thành dạng phù hợp.

4.3.2. Thiết kế mạch hai tầng

Các mạch tổ hợp hai tầng có ưu nhược điểm sau:

- Ưu điểm:

  Có thể thực hiện được mọi hàm logic.

  Có tốc độ cao.

  Việc phân tích và thiết kế đơn giản.

- Nhược điểm:

  Trong một số trường hợp thiết kế không nhận được sơ đồ đơn giản nhất.

  Yêu cầu các phần tử có số đầu vào lớn (trong trường hợp số đầu vào của phần tử cho trước không thỏa mãn phải tăng số tầng của mạch lên).

* Cách thiết kế mạch hai tầng với các phần tử cho trước:

Giả thiết rằng : Các giá trị tín hiệu vào X_i và $\overline{X_i}$ có sẵn. Trên cùng một tầng chỉ sử dụng một loại phần tử (AND, OR, NAND, NOR), những phần tử này có số đầu vào không hạn chế.

Các phương pháp thiết kế mạch hai tầng được cho trong bảng 4.1.

Bảng 4.1. *Các phương pháp thiết kế mạch hai tầng*

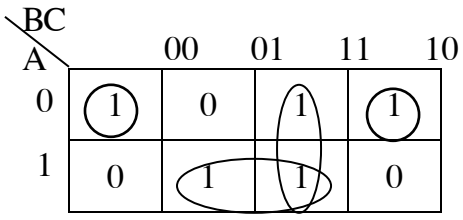
Tầng 2 / Tầng 1	AND	OR	NAND	NOR
AND	X	CTT	X	+ CTH + Demorgan
OR	CTH	X	+ CTT + Demorgan	X
NAND	+ CTH + Demorgan	X	+ CTT + Demorgan	X
NOR	X	+ CTT + Demorgan	X	+ CTH + Demorgan

Ví dụ: Cho hàm 3 biến f(A, B, C) = Σ 0, 2, 3, 5, 7

Hãy thiết kế hàm trên theo yêu cầu với các tầng cho trước.

Giải:

Tối thiểu hóa hàm f(A, B, C) (hình 4.6) ta thu được phương trình như sau:



Hình 4.6. *Tối thiểu ví dụ*

+ Theo CTT:

$f = \overline{A}\ \overline{C} + BC + AC$

+ Theo chuẩn tắc hội (CTH):

$f = (A + B + \overline{C})(\overline{A} + C)$

+ Tầng 1 dùng AND, tầng 2 dùng OR.

Từ bảng 4.1, ta thấy để thiết kế sử dụng AND - OR phương trình hàm f được viết ở dạng CTT.

$f = \overline{A}\ \overline{C} + BC + AC$

Từ phương trình này ta có thể dễ dàng vẽ sơ đồ mạch thực hiện.

+ Tầng 1 dùng OR, tầng 2 dùng AND.

Viết phương trình hàm f ở dạng CTH

$f = (A + B + \overline{C})(\overline{A} + C)$

+ Tầng 1 dùng NAND, tầng 2 dùng AND.

Viết phương trình hàm f ở dạng CTH, sau đó phủ định hai lần từng thành phần rồi áp dụng công thức Demorgan.

$$f = \overline{\overline{A + B + \overline{C}}}\ .\ \overline{\overline{\overline{A} + C}}$$

$$f = \overline{\overline{A}\ .\ \overline{B}\ .C}\ .\ \overline{A.\ \overline{C}}$$

+ Tầng 1 dùng NOR, tầng 2 dùng OR : Viết phương trình hàm f ở dạng CTT, phủ định hai lần từng thành phần rồi áp dụng Demorgan.

$f = \overline{A}\ \overline{C} + BC + AC$

$$f = \overline{\overline{\overline{A}\ \overline{C}}} + \overline{\overline{BC}} + \overline{\overline{AC}}$$

$$f = \overline{\overline{A + C}} + \overline{\overline{B} + \overline{C}} + \overline{\overline{A} + \overline{C}}$$

+ Tầng 1 dùng OR, tầng 2 dùng NAND : Viết phương trình hàm f ở dạng CTT phủ định hai lần hàm rồi áp dụng Demorgan.

$f = \overline{A}\ \overline{C} + BC + AC$

$$f = \overline{\overline{\overline{A}\ \overline{C}} + BC + AC}$$

$$f = \overline{\overline{\overline{A}\ \overline{C}}\ .\ \overline{BC}\ .\ \overline{AC}}$$

$$f = \overline{(A + C).(\overline{B} + \overline{C})(\overline{A} + \overline{C})}$$

+ Tầng 1 dùng NAND, tầng 2 dùng NAND: Viết phương trình hàm f ở dạng CTT, phủ định hai lần hàm f, rồi áp dụng Demorgan.

$$f = \overline{A}\ \overline{C} + BC + AC$$

$$f = \overline{\overline{\overline{A}\ \overline{C} + BC + AC}}$$

$$f = \overline{\overline{\overline{A}\ \overline{C}}\ .\ \overline{BC}\ .\ \overline{AC}}$$

+ Tầng 1 dùng AND, tầng 2 dùng NOR : Viết phương trình hàm f ở dạng CTH, phủ định hai lần hàm rồi áp dụng Demorgan.

$$f = (A + B + \overline{C})(\overline{A} + C)$$

$$f = \overline{\overline{(A + B + \overline{C})(\overline{A} + C)}} = \overline{A.\ \overline{C} + \overline{A}\ .\ \overline{B}\ .C}$$

+ Tầng 1 dùng NOR, tầng 2 dùng NOR: Viết phương trình hàm f ở dạng CTH, phủ định hai lần hàm f rồi áp dụng Demorgan.

$$f = (A + B + \overline{C})(\overline{A} + C)$$

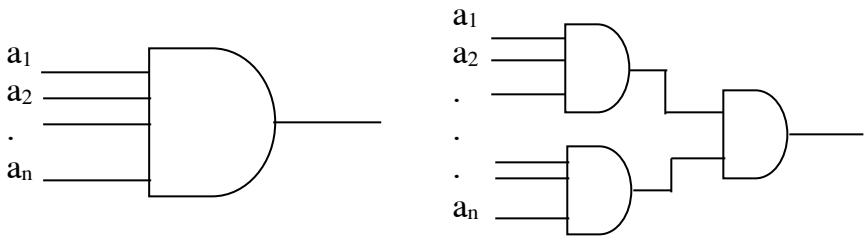
$$f = \overline{\overline{(A + B + \overline{C})(\overline{A} + C)}}$$

$$f = \overline{\overline{(A + B + \overline{C})}} + \overline{\overline{(\overline{A} + C)}}$$

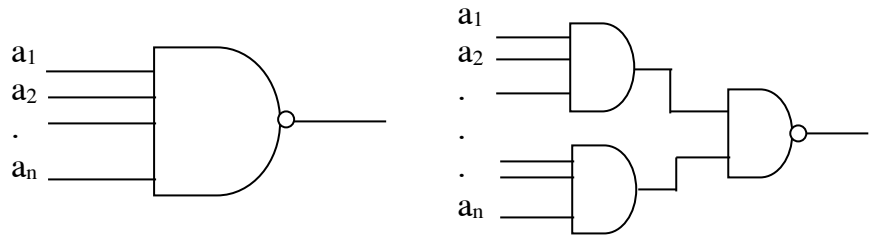
4.3.3. Thiết kế mạch nhiều tầng

Khi số đầu vào lớn hơn số đầu vào cho phép của các phần tử nhớ cho trước lúc đó phải tăng số tầng của mạch.

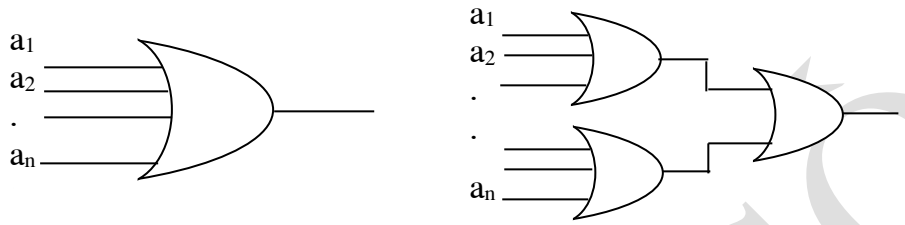
+ Thay thế mạch AND



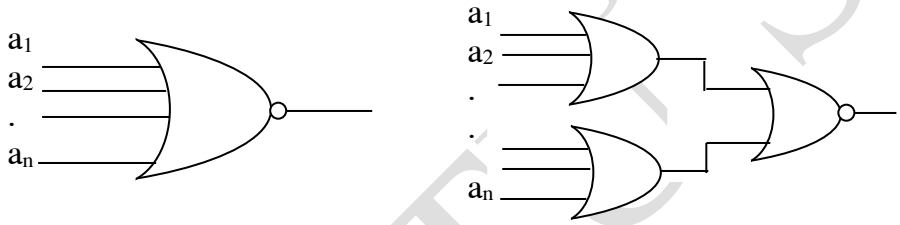
+ Thay thế mạch NAND



+ Thay thế mạch OR



+ Thay thế mạch NOR



Hình 4.7. *Thay thế các mạch nhiều đầu vào bởi các mạch cùng loại có số đầu vào ít hơn*

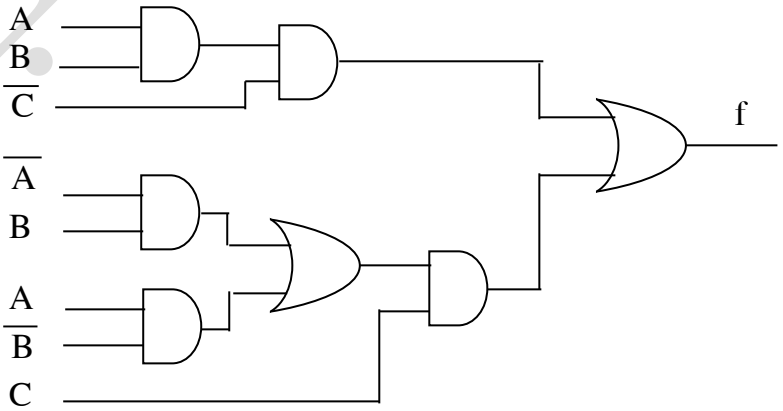
Ví dụ: Sử dụng các cổng AND, OR hai đầu vào thiết kế mạch thực hiện hàm sau:

$$f(A, B, C) = \Sigma\ 3, 5, 6$$

Giải: Ta có:

$$f(A, B, C) = \overline{A}\ BC + A\ \overline{B}\ C + AB\ \overline{C} = C\ (\ \overline{A}\ B + A\ \overline{B}\ ) + AB\ \overline{C}$$

Sơ đồ (hình 4.8)



Hình 4.8. *Sơ đồ mạch ví dụ*

4.4. MỘT SỐ MẠCH LOGIC TỔ HỢP THƯỜNG GẶP

4.4.1. Bộ cộng nhị phân

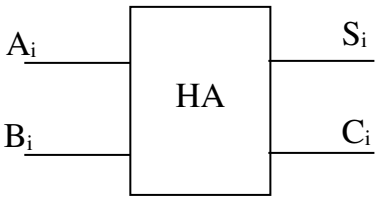
4.4.1.1. Bộ bán tổng (Half Adder)

- Sơ đồ kí hiệu:

  A_i, B_i là chữ số ở cột thứ i của A, B

  S_i là chữ số của tổng ở cột thứ i

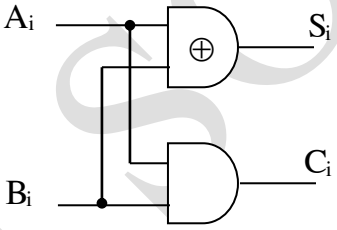
  C_i là số nhớ của phép cộng ở cột thứ i



***Hình 4.9**. Kí hiệu bộ bán tổng*

- Bảng trạng thái :

A_i	B_i	S_i	C_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



***Hình 4.10**. Sơ đồ logic bộ bán tổng*

$S_i = \overline{A_i}\ B_i + A_i\ \overline{B_i} = A_i \oplus B_i$

$C_i = A_i.B_i$

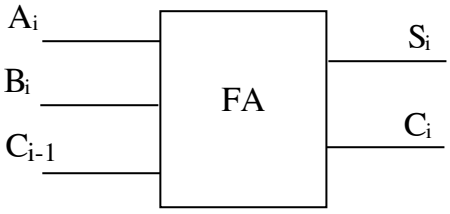
- Sơ đồ logic: gồm 1 XOR và 1 AND (hình 4.10)

4.4.1.2. Bộ tổng đầy đủ - Full Adder (bộ cộng nhị phân một cột số)

- Sơ đồ kí hiệu (hình 4.11)

C_{i-1}: là số nhớ của cột có trọng số nhỏ hơn bên cạnh đưa đến.

C_i: là số nhớ đưa đến cột có trọng số lớn hơn bên cạnh



***Hình 4.11**. Kí hiệu bộ tổng đầy đủ*

- Bảng trạng thái :

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

Ta có:

$$S_i = \overline{A_i}\ \overline{B_i}\ C_{i-1} + \overline{A_i}\ B_i\ \overline{C_{i-1}} + A_i\ \overline{B_i}\ \overline{C_{i-1}} + A_iB_iC_{i-1}$$

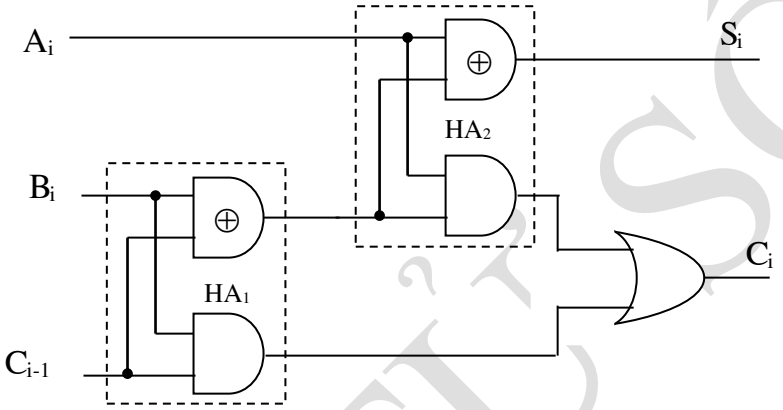
$$S_i = A_i \oplus B_i \oplus C_{i-1}$$

$$C_i = \overline{A_i}\ B_i\ C_{i-1} + A_i\ \overline{B_i}\ C_{i-1} + A_i\ B_i\ \overline{C_{i-1}} + A_iB_iC_{i-1}$$

$$C_i = B_iC_{i-1} + A_i\ B_i + A_iC_{i-1}$$ (tối thiểu theo AND - OR)

$$C_i = B_iC_{i-1} + A_i(B_i \oplus C_{i-1})$$ (thực hiện qua bộ HA)

- Sơ đồ logic thực hiện qua bộ HA (hình 4.12)



Hình 4.12. *Bộ tổng đầy đủ*

** Bộ cộng song song 4 bít (hình 4.13)*

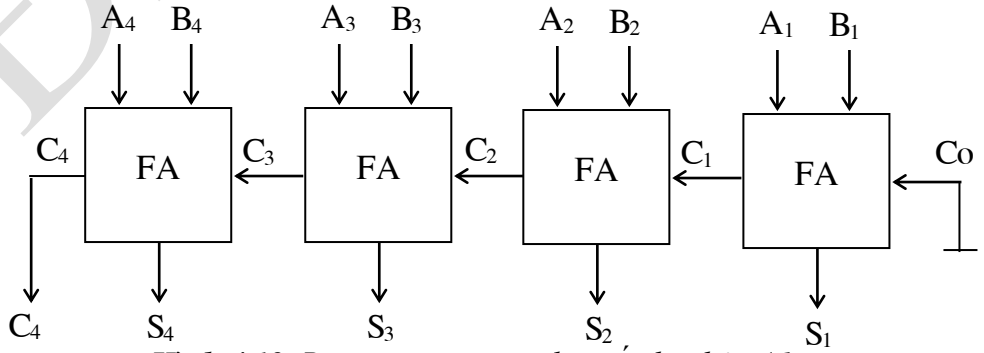
Có hai số nhị phân 4 bít A và B $A = A_4A_3A_2A_1$

$B = B_4B_3B_2B_1$

A_1, B_1 cột có trọng số nhỏ nhất (2^0)

A_4, B_4 cột có trọng số lớn nhất (2^3)

Ví dụ: Cộng hai số A = 1111
B = 1001
C = 11110
S = 10000

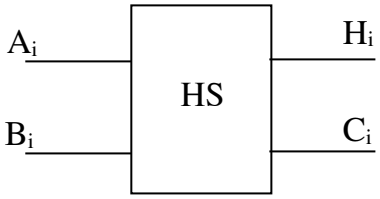


Hình 4.13. *Bộ cộng song song hai số nhị phân 4 bit*

Nhược điểm: Thời gian trễ lớn.

4.4.2. Bộ trừ nhị phân

4.4.2.1. Bộ bán trừ (Half Subtractor - HS)



***Hình 4.14**. Kí hiệu bộ bán trừ*

- Ký hiệu (hình 4.14)

  A_i, B_i là chữ số thứ i của số bị trừ và số trừ.

  H_i là chữ số của hiệu ở cột thứ i.

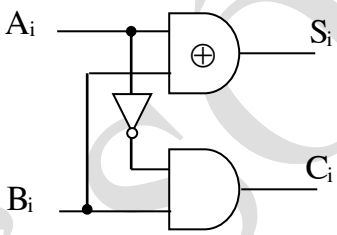
  C_i là số nhớ thứ i

- Bảng trạng thái :

A_i	B_i	H_i	C_i
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$H_i = \overline{A_i}\ B_i + A_i\ \overline{B_i} = A_i \oplus B_i$$

$$C_i = \overline{A_i}\ .B_i$$



***Hình 4.15**. Sơ đồ logic bộ bán trừ*

- Sơ đồ logic (hình 4.15).

*4.4.2.2. **Bộ trừ nhị phân** đầy đủ (Full Subtractor - FS)*

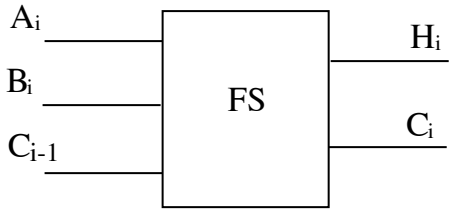
- Sơ đồ kí hiệu (hình 4.16).

  A_i, B_i là chữ số thứ i của số bị trừ và số trừ.

  C_{i-1}: là số nhớ của cột có trọng số nhỏ hơn bên cạnh đưa đến.

  H_i là chữ số của hiệu.

  C_i: là số nhớ đưa đến cột có trọng số lớn hơn bên cạnh.



***Hình 4.16**. Kí hiệu bộ trừ đầy đủ*

- Bảng trạng thái :

A_i	B_i	C_{i-1}	H_i	C_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Ta có:

$$H_i = \overline{A_i}\ \overline{B_i}\ C_{i-1} + \overline{A_i}\ B_i\ \overline{C_{i-1}} + A_i\ \overline{B_i}\ \overline{C_{i-1}} + A_iB_iC_{i-1}$$

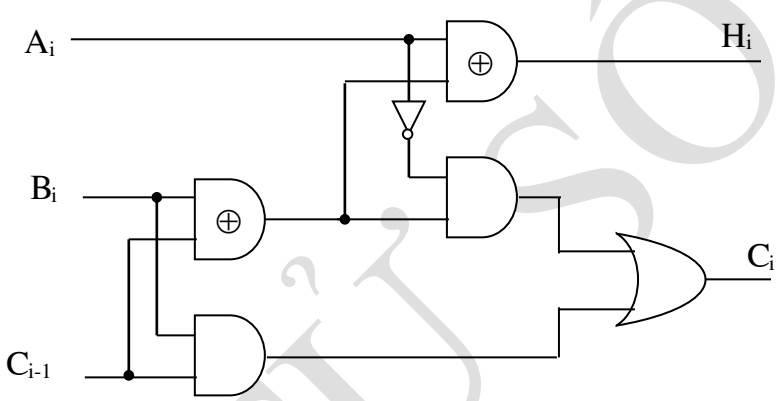
$$H_i = A_i \oplus B_i \oplus C_{i-1}$$

$$C_i = \overline{A_i}\ \overline{B_i}\ C_{i-1} + \overline{A_i}\ B_iC_{i-1} + \overline{A_i}\ B_i\ \overline{C_{i-1}} + A_iB_iC_{i-1}$$

$$C_i = B_iC_{i-1} + \overline{A_i}\ B_i + \overline{A_i}\ C_{i-1}$$ (phương trình tối thiểu theo AND - OR)

$$C_i = B_iC_{i-1} + \overline{A_i}\ (B_i \oplus C_{i-1})$$ (thực hiện qua bộ HS)

- Sơ đồ logic :



***Hình 4.17**. Bộ trừ đầy đủ*

4.4.3. Bộ so sánh (Comparator)

4.4.3.1. Bộ so sánh hai số nhị phân 1 bít

Trong nhiều trường hợp phải so sánh hai số nhị phân để chỉ ra được mối quan hệ giữa chúng A = B, A > B, A < B.

- Bảng trạng thái :

A_i	B_i	f_1 ($A_i = B_i$)	f_2 ($A_i < B_i$)	f_3 ($A_i > B_i$)
0	0	1	0	0
0	1	0	1	0
1	0	0	0	1
1	1	1	0	0

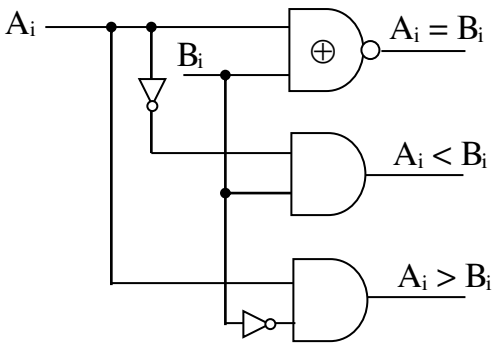
- Phương trình logic

$$f_1\ (A_i = B_i) = A_i \sim B_i = \overline{A_i \oplus B_i}$$

$$f_2\ (A_i < B_i) = \overline{A_i}\ B_i$$

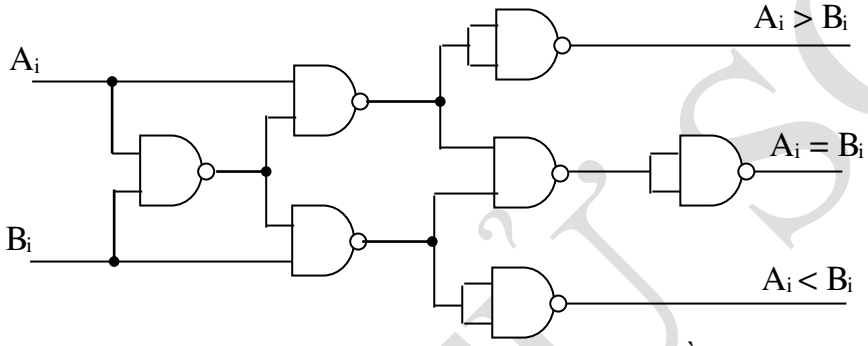
$$f_3\ (A_i > B_i) = A_i\ \overline{B_i}$$

- Sơ đồ logic:



***Hình 4.18**. Bộ so sánh nhị phân 1 bít*

Có thể xây dựng bộ so sánh từ các mạch NAND như sau:



***Hình 4.19.** Bộ so sánh nhị phân 1 bit sử dụng phần tử NAND*

4.4.3.2. Bộ so sánh n bit

Để xây dựng bộ so sánh hai số nhị phân n bit người ta có hai cách sau:

+ Xây dựng trực tiếp bộ so sánh n bít

+ Xây dựng gián tiếp qua các bộ so sánh 1 bit

$A = A_nA_{n-1}....A_1$

$B = B_nB_{n-1}....B_1$

Ví dụ so sánh hai số

$A = A_2A_1$

$B = B_2B_1$

Trước tiên ta so sánh hai cột có trọng số lớn nhất A_2, B_2

$A_2 > B_2 \Rightarrow A > B$

$A_2 = B_2 \Rightarrow$ So sánh tiếp

$A_2 < B_2 \Rightarrow A < B$

Khi $A_2 = B_2$ so sánh tiếp A_1, B_1

$A_1 > B_1 \Rightarrow A > B$

$A_1 = B_1 \Rightarrow A = B$

$A_1 < B_1 \Rightarrow A < B$

Ta có phương trình

$$f(A = B) = (A_2 = B_2)(A_1 = B_1)$$
$$= (A_2 \sim B_2)(A_1 \sim B_1)$$
$$f(A > B) = (A_2 > B_2) + (A_2 = B_2)(A_1 > B_1)$$
$$= A_2 \overline{B_2} + (A_2 \sim B_2) A_1 \overline{B_1}$$
$$f(A < B) = (A_2 < B_2) + (A_2 = B_2)(A_1 < B_1)$$
$$= \overline{A_2} B_2 + (A_2 \sim B_2) \overline{A_1} B_1$$

4.4.4. Mạch tạo và kiểm tra chẵn lẻ

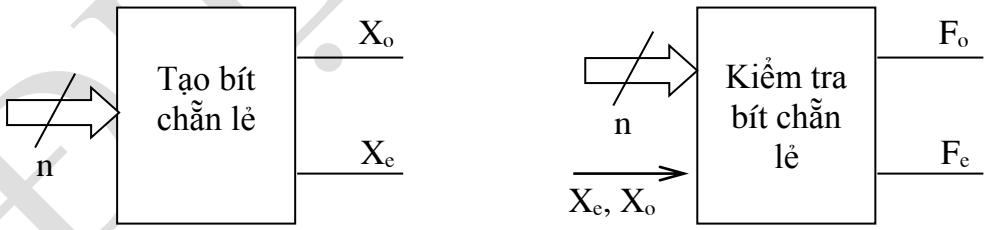
Trong thông tin khi truyền dữ liệu từ nơi này sang nơi khác cần phải xác định xem có lỗi truyền hay không, nếu có phải sửa lỗi đó. Có nhiều phương pháp mã hóa dữ liệu để có thể làm được điều này. Đơn giản nhất là chèn thêm 1 bít vào dữ liệu được truyền đi để sao cho chữ số 1 trong chuỗi dữ liệu luôn là chẵn hoặc luôn là lẻ.

- Bít chẵn: Nếu bít thêm vào có giá trị sao cho chữ số 1 trong chuỗi dữ liệu là một số chẵn (Even)
- Bít lẻ: Nếu bít thêm vào có giá trị sao cho chữ số 1 trong chuỗi dữ liệu là một số lẻ (Odd).

Để thực hiện việc truyền dữ liệu theo kiểu đưa thêm bít chẵn lẻ vào chuỗi dữ liệu cần phải:

+ Xây dựng sơ đồ tạo được bít chẵn lẻ thêm vào n bít của dữ liệu.

+ Xây dựng sơ đồ kiểm tra được hệ là hệ chẵn hay hệ lẻ với (n + 1) bít ở đầu vào (n bít dữ liệu, một bít chẵn lẻ).



***Hình 4.20**. Sơ đồ khối mạch tạo và mạch kiểm tra bit chẵn lẻ*

4.4.4.1. Mạch tạo bít chẵn lẻ (xét với n = 3)

Gọi 3 bít của dữ liệu là d_1, d_2, d_3; X_e, X_o là hai bit chẵn lẻ thêm vào dữ liệu

X_e là giá trị bít thêm vào để hệ là hệ chẵn;

X_o là giá trị bít thêm vào để hệ là hệ lẻ.

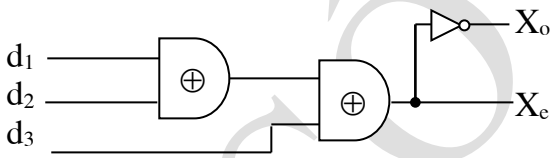
Bảng trạng thái :

d_3	d_2	d_1	Xe	Xo
0	0	0	0	1
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	0

Ta thấy $X_o = \overline{X_e}$

$X_e = d_3 \oplus d_2 \oplus d_1$

$X_o = \overline{d_3 \oplus d_2 \oplus d_1}$



***Hình 4.21**. Mạch tạo bit chẵn lẻ*

Sơ đồ logic (hình 4.21): 2 cổng XOR.

4.4.4.2. Mạch kiểm tra chẵn lẻ

Xét mạch với 3 bít dữ liệu d_3, d_2, d_1: mạch có 4 đầu vào : 3 đầu vào dữ liệu và 1 đầu vào X, hai đầu ra F_e và F_o F_e = 1 nếu hệ là hệ chẵn

F_o = 1 nếu hệ là hệ lẻ

Bảng trạng thái:

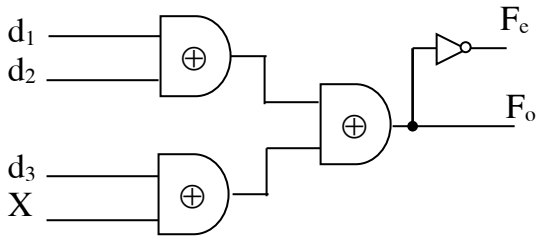
d_3	d_2	d_1	X	F_e	F_o
0	0	0	0	1	0
0	0	0	1	0	1
0	0	1	0	0	1
0	0	1	1	1	0
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	0
0	1	1	1	0	1
1	0	0	0	0	1
1	0	0	1	1	0
1	0	1	0	1	0
1	0	1	1	0	1
1	1	0	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1
1	1	1	1	1	0

Ta luôn có $F_e = \overline{F_o}$

$F_o = d_3 \oplus d_2 \oplus d_1 \oplus X$

$F_e = \overline{d_3 \oplus d_2 \oplus d_1 \oplus X}$

Sơ đồ logic (hình 4.22)



***Hình 4.22**. Mạch kiểm tra bit chẵn lẻ*

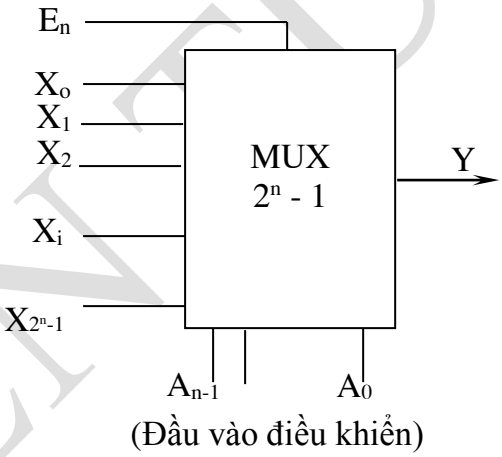
4.4.5. Bộ ghép kênh và bộ tách kênh (MUX và DEMUX)

4.4.5.1. Bộ ghép kênh (MUX)

Định nghĩa: Bộ ghép kênh là mạch có 2^n đầu vào biến, n đầu vào điều khiển, một đầu vào chọn mạch và một đầu ra. Tuỳ theo giá trị của n đầu vào điều khiển mà đầu ra sẽ bằng một trong những giá trị ở đầu vào biến.

Nếu giá trị thập phân ứng với tổ hợp nhị phân của n đầu vào điều khiển bằng j thì $Y = X_j$.

Sơ đồ khối (hình 4.23):



***Hình 4.23.** Sơ đồ khối của MUX 2^n - 1*

Phương trình tín hiệu ra của MUX 2^n - 1

$$Y = X_o(\overline{A_{n-1}}\ \overline{A_{n-2}}\ \ldots\ \overline{A_0}) + X_1(\overline{A_{n-1}}\ \overline{A_{n-2}}\ \ldots\ \overline{A_1}\ A_0) + \ldots +$$

$$+ X_{2^n-1}(A_{n-1}A_{n-2}\ldots A_1A_0)$$

Ví dụ: Tạo bộ MUX 4 - 1

Mạch có: 4 đầu vào biến: X_3, X_2, X_1, X_0

2 đầu vào điều khiển A_1, A_0

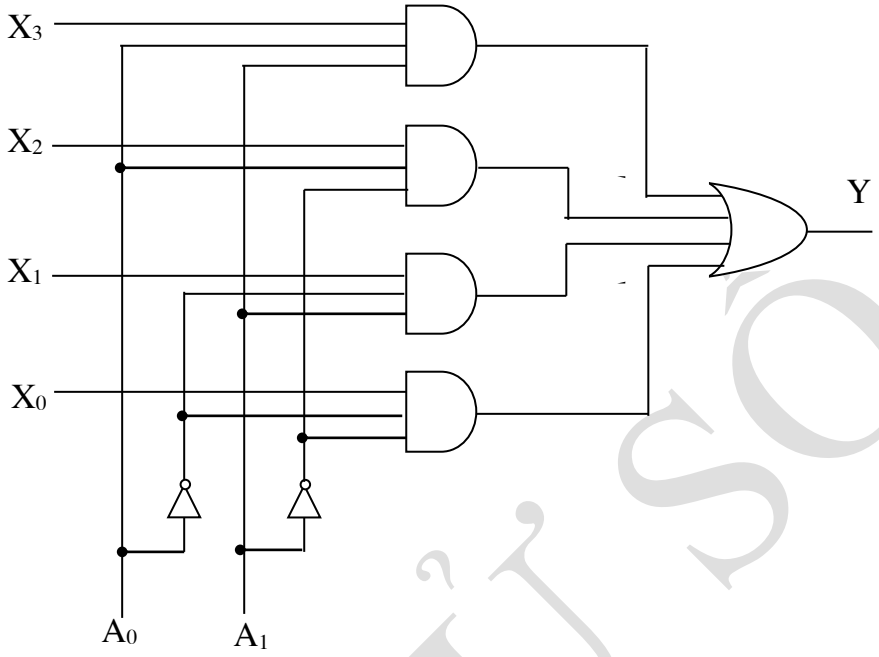
1 đầu ra Y

A_1	A_0	Y
0	0	X_0
0	1	X_1
1	0	X_2
1	1	X_3

$$Y = X_0\ \overline{A_1}\ \overline{A_0} + X_1\ \overline{A_1}\ A_0 + X_2\ A_1\ \overline{A_0} + X_3\ A_1A_0$$

Sơ đồ mạch (hình 4.24).

Thực chất của MUX là bộ chuyển mạch điện tử dùng các tín hiệu điều khiển ($A_{n-1}A_{n-2}...A_1A_0$) để điều khiển sự nối mạch của đầu ra với một trong số 2^n đầu vào.



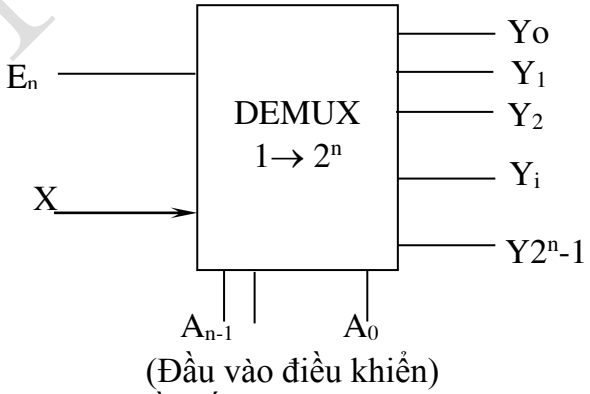
***Hình 4.24**. Sơ đồ logic MUX 4 - 1*

4.4.5.2. Bộ tách kênh (DEMUX)

Định nghĩa: Bộ tách kênh là mạch có một đầu vào biến, n đầu vào điều khiển, một đầu vào chọn mạch và 2^n đầu ra. Tuỳ theo giá trị của n đầu vào điều khiển mà một trong các đầu ra sẽ bằng giá trị của đầu vào biến.

Nếu n đầu vào điều khiển là $A_{n-1}A_{n-2}...A_0$ thì Yi = X khi $(A_{n-1}A_{n-2}...A_0)_2 = (i)_{10}$.

Sơ đồ khối (hình 4.25).



***Hình 4.25**. Sơ đồ khối DEMUX 1 - 2^n*

Hệ phương trình các đầu ra:

$$Y_0 = X(\overline{A_{n-1}}\ \overline{A_{n-2}}\ ...\ \overline{A_0})$$

$$Y_1 = X(\overline{A_{n-1}}\ \overline{A_{n-2}}\ ...\ \overline{A_1}\ A_0)$$

...

$$Y_{2^n-1} = X(A_{n-1}A_{n-2}...A_1A_0)$$

Ví dụ : Tạo DEMUX 1 - 4

Mạch có: 1 đầu vào biến X ; 4 đầu ra: Y_3, Y_2, Y_1, Y_0 ;

2 đầu vào điều khiển A_1, A_0

Bảng trạng thái.

A_1	A_0	Y_0	Y_1	Y_2	Y_3
0	0	X	0	0	0
0	1	0	X	0	0
1	0	0	0	X	0
1	1	0	0	0	X

Hệ phương trình đầu ra:

$$Y_0 = X\ \overline{A_1}\ \overline{A_0}$$

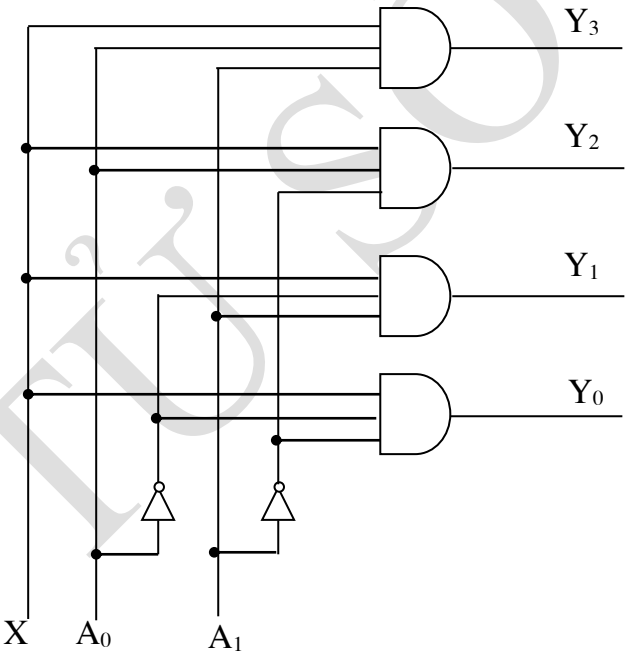
$$Y_1 = X\ \overline{A_1}\ A_0$$

$$Y_2 = X\ A_1\ \overline{A_0}$$

$$Y_3 = X\ A_1 A_0$$

Sơ đồ mạch (hình 4.26).

Thực chất của DEMUX là bộ chuyển mạch điện tử dùng các tín hiệu điều khiển ($A_{n-1}A_{n-2}. . .A_1A_0$) để điều khiển sự nối mạch của đầu vào với một trong số 2^n đầu ra.



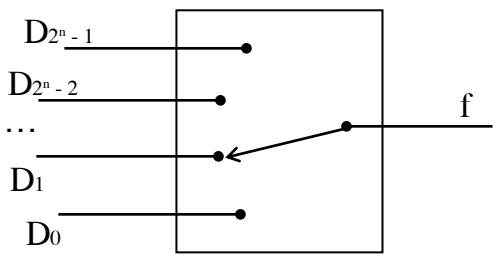
Hình 4.26. *Sơ đồ logic DEMUX 1 - 4*

4.4.5.3. Một số ứng dụng của MUX và DEMUX

a) Sử dụng MUX làm bộ chọn dữ liệu (hay chuyển mạch điện tử)

Dữ liệu vào được chọn để đưa tới đầu ra phụ thuộc vào tín hiệu điều khiển hay tín hiệu chọn. Điều này được mô tả trong hình 4.27.

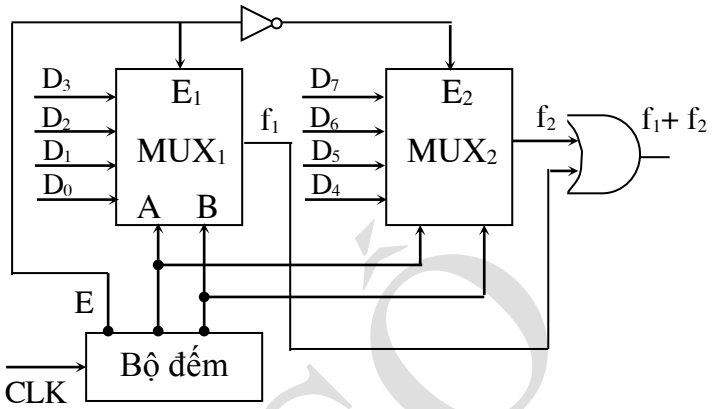
Số các đầu vào dữ liệu có thể tăng lên nhờ sử dụng MUX có nhiều đầu vào hoặc bằng cách mắc tổ hợp nhiều MUX có số đầu vào nhỏ.



Hình 4.27. *Bộ chọn dữ liệu dùng MUX 2^n - 1*

Ví dụ: Xây dựng MUX 8 đầu vào từ các MUX 4 đầu vào, việc điều khiển hoạt động của MUX do một bộ đếm nhị phân 3 đầu ra thực hiện. Giá trị của bộ đếm được cho trong bảng.

E	A	B	E_1	E_2
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0



Hình 4.28. *Xây dựng MUX 8 đầu vào từ các MUX 4 đầu vào và các mạch phụ cận.*

b) Tạo dãy tín hiệu nhị phân tuần hoàn

Muốn tạo dãy tín hiệu nhị phân tuần hoàn ta chỉ cần nối các đầu vào của MUX sử dụng với các mức logic nhất định. Việc thực hiện đưa tín hiệu ra nhờ một bộ đếm nhị phân, mà các đầu ra của bộ đếm nhị phân này được đưa vào các đầu vào điều khiển của MUX đó.

c) Tạo hàm logic tổ hợp

Một bộ MUX 2^n - 1 có thể dùng để tạo hàm logic bất kỳ có n + 1 biến vào, trong đó n biến sẽ đưa vào n đầu điều khiển, còn một biến cùng với các hằng số 0 và 1 được đưa vào 2^n đầu vào còn lại tuỳ thuộc giá trị của hàm số.

Ví dụ: Sử dụng MUX 4 $\Rightarrow$ 1, tạo hàm 3 biến:

$$f = \overline{A}\ \overline{B}\ C + \overline{A}\ B\ C + \overline{A}\ B\ \overline{C} + A\ B\ C$$

Giải:

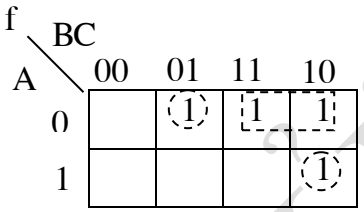
Ta sẽ đưa 2 biến vào đến các đầu vào điều khiển của MUX 4 - 1. Có 3 khả năng chọn hai biến này trong 3 biến của hàm f, các khả năng đó là:

- Hai biến điều khiển A và B.
- Hai biến điều khiển B và C.
- Hai biến điều khiển A và C.

Sau khi đã chọn biến điều khiển cần xác định các giá trị được đưa vào đầu vào MUX để thực hiện hàm f ban đầu. Có nhiều cách để xác định giá trị này, cách dùng bảng Karnaugh được tiến hành theo các bước sau:

- n biến điều khiển được phân thành 2^n vùng khác nhau trên bảng Karnaugh đánh dấu là D_0, D_1..., D_{2^n-1} (vùng D_i ứng với giá trị thập phân của n biến điều khiển là i).
- Điền giá trị hàm số cho trước vào bảng Karnaugh.
- Tối thiểu hoá hàm đã cho trong từng vùng D_i, gọi hàm số này là D_i. D_i chính là giá trị đầu vào tại D_i của MUX.
- Nếu D_i là các hàm một biến, hoặc hằng 0, hằng 1 thì bài toán đã giải xong. Trong trường hợp ngược lại phải tiếp tục dùng MUX hoặc dùng các cổng logic để thực hiện các hàm D_i để đáp ứng các yêu cầu.

Trường hợp hai biến điều khiển A, B ta có bảng Karnaugh như sau:



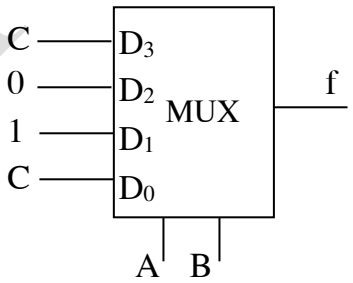
Hình 4.29. *Tìm hàm logic tổ hợp theo MUX*

Giá trị của các hàm đưa vào các đầu vào của MUX 4 - 1 là:

$D_0 = C$ $D_1 = 1$

$D_2 = 0$ $D_3 = C$

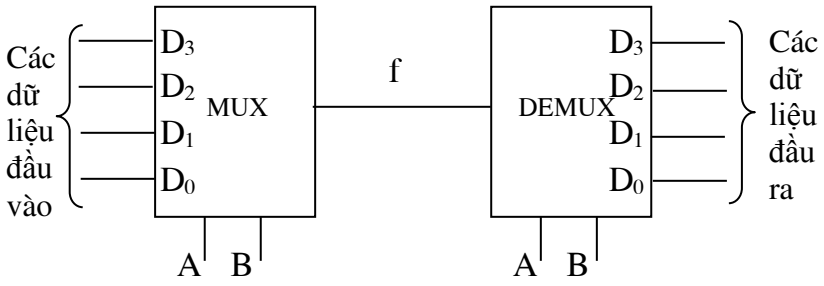
Sơ đồ thực hiện hàm với hai biến điều khiển A, B (hình 4.30).



Hình 4.30. *Tạo hàm logic tổ hợp theo MUX*

d) Đo lường điều khiển kết hợp với truyền dữ liệu

MUX có nhiều đầu vào dữ liệu, mỗi đầu vào có thể thu nhận một dữ liệu (như sự đóng, mở của một công tắc, đầu ra của một cảm biến…). Các dữ liệu vào ở các đầu vào được tích hợp lại thành dữ liệu nối tiếp để truyền đi xa. Ở đầu kia mạch DEMUX sẽ tách các dữ liệu để dùng vào việc theo dõi hay điều khiển.

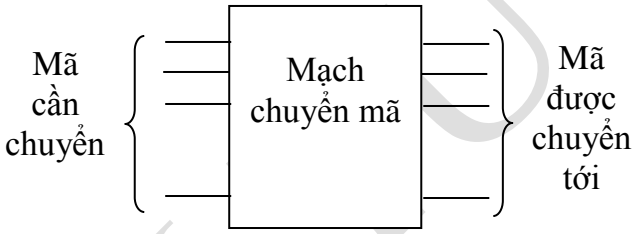


Hình 4.31. *Hệ thống truyền dữ liệu nối tiếp*

4.4.6. Mạch mã hóa và giải mã

- Mạch mã hoá có nhiệm vụ tạo ra các mã số, ở mỗi thời điểm chỉ có một đầu vào tích cực.
- Mạch giải mã có nhiệm vụ biến đổi mã nhận được thành mã ban đầu (ngược với mạch mã hoá).

Để thiết kế các mạch chuyển mã ta làm theo các bước sau:



Hình 4.32. *Sơ đồ khối mạch chuyển mã*

Bước 1: Lập bảng giá trị của mạch.

Bước 2: Tối thiểu hoá các hàm ra.

Bước 3: Vẽ sơ đồ mạch thực hiện.

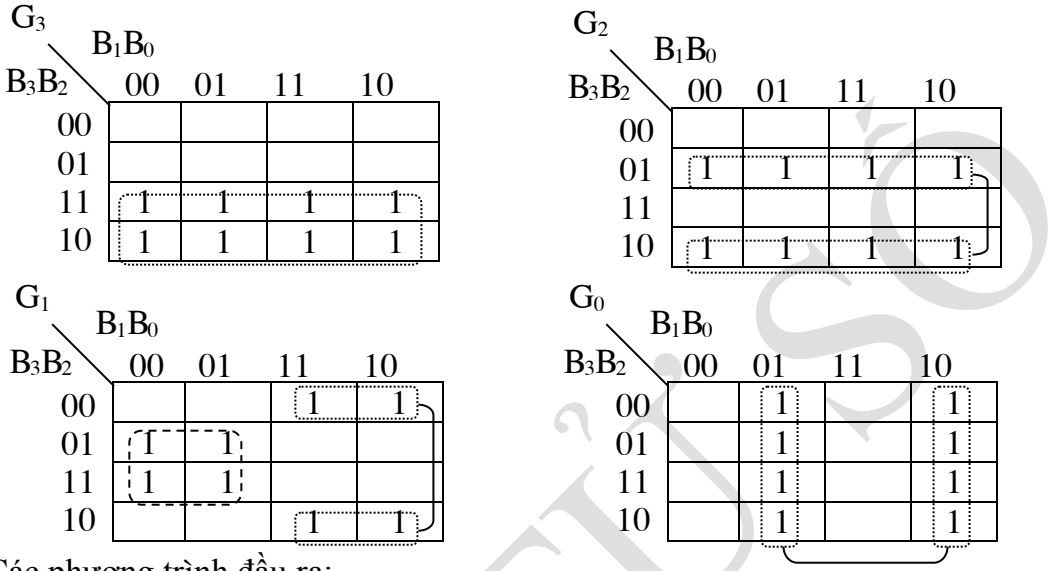
Ví dụ 1: Chuyển mã nhị phân thành mã Gray

+ Lập bảng trạng thái: có 4 đầu vào (B_3, B_2, B_1, B_0) và 4 đầu ra (G_3, G_2, G_1, G_0)

Mã nhị phân (đầu vào)				Mã Gray (đầu ra)			
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

Biểu diễn các hàm đầu ra trên bìa Karnaugh và tối thiểu hoá hàm:



Các phương trình đầu ra:

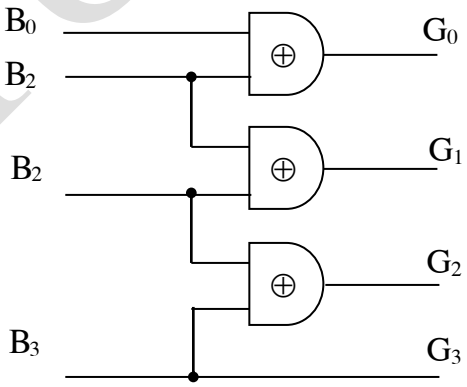
$G_3 = B_3$

$G_2 = \overline{B_3}\ B_2 + B_3\ \overline{B_2} = B_3 \oplus B_2$

$G_1 = \overline{B_1}\ B_2 + B_1\ \overline{B_2} = B_1 \oplus B_2$

$G_0 = \overline{B_1}\ B_0 + B_1\ \overline{B_0} = B_1 \oplus B_0$

Sơ đồ logic (hình 4.33).



Hình 4.33. *Sơ đồ mạch chuyển mã nhị phân sang mã Gray*

Ví dụ 2: Chuyển mã Gray thành mã nhị phân.

Thiết kế tương tự như ví dụ 1 ta có kết quả như sau:

$B_3 = G_3$

$B_2 = G_3 \oplus G_2$

$B_1 = G_3 \oplus G_2 \oplus G_1$

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

Sử dụng cách thiết kế tương tự các ví dụ trên ta có các sơ đồ chuyển mã khác.

4.5. CÂU HỎI VÀ BÀI TẬP CHƯƠNG 4

1. Hãy nêu các bước thiết kế mạch logic tổ hợp.

2. Có thể từ sự mô tả bài toán thiết kế viết ngay biểu thức logic mà không cần lập bảng chân lý không?

3. Thiết kế mạch thực hiện hàm logic f = Σ 2, 4, 7

Chỉ sử dụng các cổng AND, OR hai đầu vào?

Chỉ sử dụng cổng NAND hai đầu vào?

Chỉ sử dụng cổng NOR hai đầu vào?

4. Một mạch logic có hai đầu vào tín hiệu A và B, đầu vào điều khiển C và một đầu ra Y. Hãy thiết kế mạch logic trên chỉ sử dụng cổng NAND, sao cho khi C ở mức logic cao thì A ra ở Y, khi C ở mức logic thấp thì B ra ở Y.

5. Thiết kế mạch so sánh hai số nhị phân 2 bit A và B, sao cho đầu ra

Y_1 = 1 khi A bằng B

Y_2 = 1 Khi A khác B.

6. Thiết kế mạch logic có 3 đầu vào A, B, C, đầu ra lên cao chỉ khi nào đa số đầu vào ở mức cao?

7. Thiết kế mạch tổ hợp thực hiện việc kiểm tra lẻ, biết mạch có 3 đầu vào, một đầu ra. Chức năng của mạch như sau: nếu các tín hiệu lối vào có số lẻ ở mức cao thì đầu ra có mức cao, ngược lại đầu ra có mức thấp.

8. Thiết kế mạch giải mã từ mã nhị phân 4 bit sang mã thập phân.

9. Thiết kế mạch giải mã NBCD sang mã 7 vạch.

10. Thiết kế bộ hợp kênh 8-1.

11. Thiết kế bộ phân kênh 1-8, trong đó có một đầu vào điều khiển hệ thống.

12. Mạch điều khiển máy photo có 4 đầu vào và một đầu ra, các đầu vào nối đến các công tắc nằm dọc theo đường di chuyển của giấy. Bình thường công tắc mở và các đầu vào A, B, C, D nhận giá trị cao, khi giấy chạy qua một công tắc thì nó đóng và đầu vào tương ứng xuống thấp. Hai công tắc nối đến A, D không bao giờ cùng đóng một lúc (nghĩa là giấy ngắn hơn khoảng cách tới hai công tắc này). Thiết kế mạch điều khiển trên biết đầu ra cao khi có hai hoặc ba công tắc đóng cùng một lúc.

a. Thiết kế mạch sử dụng NAND.

b. Thiết kế mạch sử dụng 1 bộ MUX 8-1 và các cổng logic cơ bản nếu cần.

13. Thiết kế mạch chuyển mã nhị phân sang mã d- 3 (4 bít).

a. Vẽ sơ đồ mạch sử dụng cổng logic cơ bản.

b. Vẽ sơ đồ mạch sử dụng MUX 8-1

14. Thiết kế mạch chuyển mã mã d- 3 sang nhị phân (4 bít).

a. Vẽ sơ đồ mạch sử dụng cổng logic cơ bản.

b. Vẽ sơ đồ mạch sử dụng MUX 8-1

15. Thiết kế mạch chuyển mã Gray sang mã nhị phân (4 bít).

a. Vẽ sơ đồ mạch sử dụng cổng logic cơ bản.

b. Vẽ sơ đồ mạch sử dụng MUX 8-1

16. Thiết kế mạch chuyển mã nhị phân sang mã Gray (4 bít).

a. Vẽ sơ đồ mạch sử dụng cổng logic co bản.

b. Vẽ sơ đồ mạch sử dụng MUX 8-1

17. Cho một hệ tổ hợp hoạt động theo bảng sau:

E	X_1	X_0	Y_0	Y_1	Y_2	Y_3
1	X	X	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	1	0	0
0	1	0	0	0	1	0
0	1	1	0	0	0	1

a. Thiết kế hệ tổ hợp này dùng cổng logic bất kỳ.

b. Dùng hệ tổ hợp đã thiết kế ở câu a (vẽ ở dạng sơ đồ khối) và các cổng logic thực hiện hàm $F(A,B,C) = \sum(4,6)$

18. Thiết kế mạch giải mã 2421 thành mã thập phân

a. Thực hiện bằng các cổng logic.

b. Thực hiện bằng mạch giải mã (decoder) 4-16 có đầu ra tích cực mức 1.

19. Một mạch tổ hợp có 5 đầu vào A, B, C, D, E và một đầu ra Y. Đầu vào là một từ mã thuộc bộ mã như sau

E	D	C	B	A
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	1	1
1	0	0	0	0
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

a. Thiết kế mạch tổ hợp dùng cổng AND-OR sao cho Y=1 khi đầu vào là một từ mã đúng và Y=0 khi đầu vào là một từ mã sai.

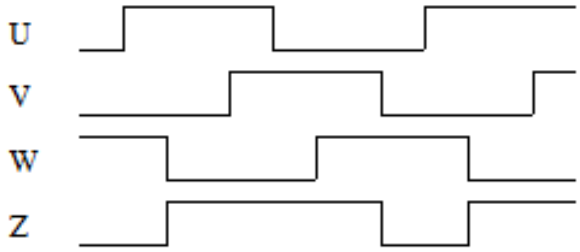
b. Thực hiện lại câu a chỉ dùng toàn cổng NAND.

20. Cho một hệ tổ hợp hoạt động theo bảng sau

E	X_1	X_0	Y_0	Y_1	Y_2	Y_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

a. Thiết kế hệ tổ hợp này dùng cổng NOT và NAND 3 đầu vào.

b. Dùng hệ tổ hợp đó thiết kế ở câu a (vẽ ở dạng sơ đồ khối) và một cổng AND 2 đầu vào để thực hiện một hệ tổ hợp hoạt động theo giản đồ xung như sau (với U, V, W là các đầu vào; Z là đầu ra).



***Hình 4.34**. Tín hiệu vào và ra bài 20*

21. Thực hiện mạch FA trên cơ sở mạch Mux 4-1

22. Lập bảng chân lý của mạch ghép kênh Mux 16-1. Sau đó, thực hiện mạch ghép kênh 16-1 trên cơ sở mạch ghép kênh 4-1.

23. Thiết kế mạch trừ hai số một bit, trong đó V là biến điều khiển, C_{i-1} là số mượn đầu vào, C_i là số mượn đầu ra. Khi V = 0 thì mạch thực hiện D = A - B, khi V = 1 thì thực hiện D = B - A

24. Thiết kế mạch trừ hai số 3 bit A và B với biến điều khiển V, dựa trên cơ sở mạch trừ hai số một bit ở bài 23.

26. Thiết kế mạch trừ hai số 3 bit A và B sao cho kết quả luôn luôn dương.

27. Thiết kế mạch cộng/trừ hai số nhị phân 4 bit X và Y dùng vi mạch 7483 (mạch cộng 4 bit) và các cổng logic (nếu cần). Mạch có tín hiệu điều khiển là v, khi v = 0 mạch thực hiện X+Y, khi v = 1 mạch thực hiện X - Y.

28. Chỉ sử dụng mạch FA, hãy thiết kế hệ tổ hợp có bảng chân lý sau:

x_1	x_0	y_0	y_1	y_2	y_3
0	0	0	1	0	0
0	1	1	0	1	0
1	0	1	0	1	0
1	1	0	1	1	1

29. Cho hàm F với 4 biến vào. Hàm có giá trị bằng 1 nếu số lượng biến vào có giá trị bằng 1 nhiều hơn hoặc bằng số lượng biến có giá trị bằng 0. Ngược lại, hàm có giá trị bằng 0.

a. Hãy biểu diễn hàm trên bìa Karnaugh.

b. Rút gọn hàm và vẽ mạch thực hiện dùng cổng NAND.

30. Sử dụng các mạch ghép kênh 8-1 và mạch ghép kênh 4-1 để thiết kế mạch ghép kênh 32-1.

31. Cho F là một hàm 4 biến A, B, C, D. Hàm F = 1 nếu giá trị thập phân tương ứng với các tổ hợp biến của hàm chia hết cho 3 hoặc 5, ngược lại F=0.

a. Lập bảng chân lý cho hàm F.

b. Thực hiện hàm F bằng mạch ghép kênh 16-1.

c. Thực hiện hàm F bằng mạch ghép kênh 8-1 và các cổng logic (nếu cần).

d. Thực hiện hàm F bằng mạch ghép kênh 4-1 và các cổng logic (nếu cần).

e. Hãy biểu diễn hàm F trên bìa Karnaugh

f. Hãy rút gọn F và thực hiện F chỉ dùng các mạch HA.

32. Cho hàm $F(A,B,C) = AB + BC + AC$. Hãy thiết kế mạch thực hiện hàm F chỉ sử dụng:

a. Một vi mạch 74138 (decoder 3-8, đầu ra tích cực thấp) và một cổng có tối đa 4 đầu vào.

b. Một vi mạch 74153 (mux 4-1, có đầu cho phép tích cực thấp).

c. Hai mạch HA và một cổng OR.

33. Sử dụng ba mạch ghép kênh 2-1 để thực hiện một mạch ghép kênh 4-1. Không dùng thêm cổng logic cơ bản.

TÀI LIỆU THAM KHẢO

1. Nguyễn Thuý Vân, *Kỹ thuật số*, Nhà xuất bản khoa học kỹ thuật - 1999

2. Nguyễn Hữu Phương, *Mạch số*, Nhà xuất bản thống kê - 2001

3. Đặng Văn Chuyết, *Điện tử số*, Nhà xuất bản giáo dục - 2000

4. Vũ Đức Thọ (dịch), *Cơ sở kỹ thuật điện tử số*, Nhà xuất bản giáo dục - 2001

5. Huỳnh Đắc Thắng, *Kỹ thuật số thực hành*, Nhà xuất bản giáo dục - 2001

6. Nguyễn Xuân Quỳnh, *Lý thuyết mạch logic và kỹ thuật số*, Nhà xuất bản đại học

7. Donald G.Fink, Donald Christiansen, *Sổ tay kỹ sư điện tử*, Nhà xuất bản khoa học và kỹ thuật - 2002