

**ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN
BỘ MÔN: TỰ ĐỘNG HOÁ**

Giáo trình Kỹ thuật xung - số



***GV soạn: LÂM TĂNG ĐỨC
TRẦN ĐÌNH KHÔI QUỐC***

Đà Nẵng 06/2007

CHƯƠNG I

HỆ THỐNG SỐ VÀ MÃ SỐ

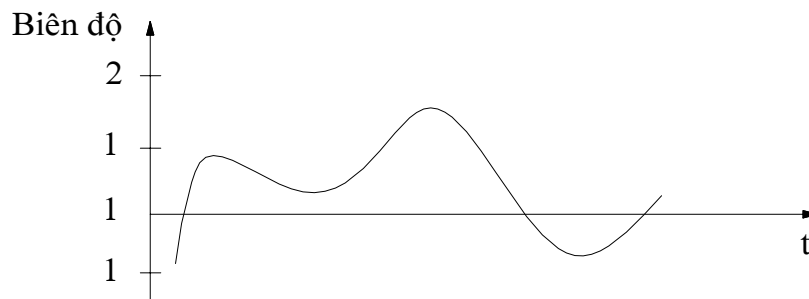
Chương này trình bày sự khác nhau giữa mạch tương tự và mạch số, các ưu điểm của mạch số. Mạch số còn gọi mạch logic, hoạt động ở hai trạng thái cách biệt rất rõ rệt, do đó số tự nhiên thích hợp với hệ thống số nhị phân chỉ dùng hai con số 0 và 1 thay vì mười con số 0 đến 9 như ở hệ thập phân quen thuộc. Để làm rõ thêm về hệ nhị phân, phép tính số học cơ bản và số có dấu, kể cả số bù 1 và bù 2 cũng được trình bày. Tiếp theo là hệ thập lục phân dùng các con số 0 đến 9 và một số chữ cái để diễn tả số lượng hay trạng thái từ 0 đến 15 thập phân. Nửa sau của chương là các mã số mà chính là mã BCD và mã ASCII thông dụng.

1.1 MẠCH TƯƠNG TỰ VÀ MẠCH SỐ

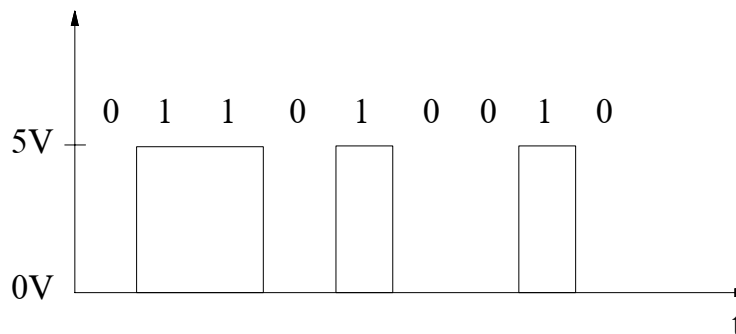
Tín hiệu là biến thiên của biên độ, mà thường là điện thế hay dòng điện, theo thời gian. Đường biểu diễn của tín hiệu là dạng sóng.

Mạch tương tự xử lý tín hiệu tương tự. Tín hiệu tương ứng với tiếng nói, tín hiệu tâm điện, tín hiệu tương ứng với sự biến thiên của nhiệt độ là vài ví dụ về tín hiệu tương tự (hình 1.1a). Tín hiệu tương tự có đặc tính:

- Thường do các hiện tượng tự nhiên phát sinh ra và được một cảm biến chuyển thành tín hiệu điện, ví dụ tiếng nói, hình ảnh.
- Liên tục về biên độ nghĩa là có bất cứ trị số nào trong khoảng biến thiên của nó, ví dụ 1V; 1,1V; 1,12V; 1,125V...
- Thường là liên tục về thời gian.



(a) tín hiệu tương tự



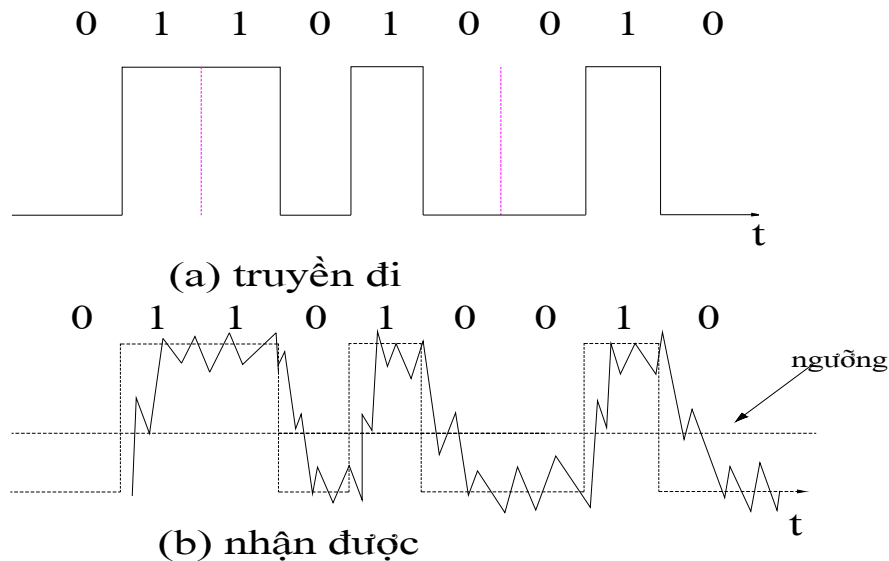
(b) tín hiệu số

Hình 1.1 Tín hiệu tương tự và số

Mặc khác mạch số xử lý tín hiệu số thường là tín hiệu nhị phân (gồm logic 0 và 1) có dạng sóng xung (hình 1.1b). với hai mức biên độ: mức cao (logic 1) ví dụ bằng 5V, và mức thấp (logic 0) ví dụ bằng 0V. Thời gian biến thiên giữa hai mức, gọi thời gian chuyển tiếp, là đột biến (vô cùng ngắn) nên tín hiệu số có thể xem như gián đoạn về biên độ. Thật ra mức cao và mức thấp có thể có các giá trị số khác với hình 1.1b.

Tín hiệu tương tự như ở hình 1.1a có thể được chuyển đổi thành tín hiệu số như ở hình 1.1b bởi mạch chuyển đổi tương tự sang số (Analog to Digital Converter - ADC). Dĩ nhiên tín hiệu số cũng được phát sinh bởi chính các mạch số (kể cả máy tính).

Ưu điểm của mạch số.



Hình 1.2 Ảnh hưởng của sự méo dạng và nhiễu

Mạch số có nhiều ưu điểm so với mạch tương tự khiến mạch số ngày càng phổ biến. Ở gần như mọi lĩnh vực từ đo lường, điều khiển đến tính toán, thông tin (điện thoại số thay thế điện thoại tương tự, truyền hình số sẽ thay thế truyền hình tương tự vv...). Tuy mạch điện tương tự và các thiết bị tương tự cũng có những đặc tính riêng khiến không bao giờ bị thay thế hoàn toàn bởi mạch số và các thiết bị số.

Sau đây là một số ưu điểm của mạch số:

➤ Khả năng chống nhiễu và sự méo dạng cao: Nhiễu là những tín hiệu lộn xộn do chính mạch điện tử tạo ra hay từ bên ngoài thâm nhập vào chồng lên tín hiệu đích thực biểu thị thông tin mà ta cần xử lý hay truyền đi. Ngoài ra tín hiệu truyền trong mạch điện tử và môi trường thông tin (dây điện cáp, sợi quang, không gian vv...) còn bị méo dạng. Kết quả là tín hiệu nhận được ở máy thu bị méo dạng và bị nhiễu (hình 1.2). Ở máy thu tín hiệu được so sánh với một ngưỡng (thời điểm so sánh là tại giữa

thời gian của xung biểu thị 1 hoặc 0) để xác định lại hai mức: nếu tín hiệu nhỏ hơn ngưỡng là mức thấp, nếu cao hơn ngưỡng là mức cao. Sau đó dạng xung vuông được tái tạo giống như ở đầu truyền. Như vậy, nhiễu và sự méo dạng, ngoại trừ khi trầm trọng, không ảnh hưởng lên kết quả.

➤ Tự phát hiện sai và sửa sai: khi nhiễu hay sự méo dạng trầm trọng thì kết quả nhận được bị sai. Điều hay là bằng cách mã hoá dữ liệu nhj phân một cách thích hợp người ta làm hệ thống mạch số có khả năng tự biết chỗ sai và tự sửa lại cho đúng.

➤ Lưu trữ và truy cập dễ dàng và nhanh chóng: Do tín hiệu số chỉ có hai mức nên việc lưu trữ ở các môi trường khác nhau (bộ nhớ bán dẫn ,băng từ, ...), và truy cập rất thuận tiện.

➤ Tính toán, lý luận nhanh chóng: Tính toán ý nói các phép tính cơ bản cộng trừ nhân chia và sự kết hợp các phép toán này để giải bài toán phức tạp hơn. Lý luận (logic) ý nói các phép so sánh, dịch chuyển, phân loại, xếp hạng vv...Máy tính là kết hợp các khả năng tính toán, lý luận và lưu trữ.

➤ Độ chính xác và độ phân giải cao: Trong việc đo đạc thời gian, tần số, điện thế vv... kỹ thuật số cho độ chính xác và độ phân giải cao hơn kỹ thuật tương tự.

➤ Thuận tiện cho công việc tích hợp: Mạch số dù là một máy tính , một mạng điện thoại số, vv... chủ yếu là do một số mạch cơ bản nhưng được lập đi lập lại hàng ngàn, hàng triệu lần tạo nên. Chính sự lập đi lập lại này rất thuận lợi cho việc chế tạo các mạch tích hợp (Intergrated Circuit □ IC). Thực tế đã có hàng ngàn các IC số khác nhau làm các chức năng từ giản đơn đến vô cùng tinh vi phức tạp.

➤ Dễ thiết kế, lắp ráp, sửa chữa: Do có rất nhiều mạch IC cho các chức năng khác nhau, mà các IC này gần như không cần các linh kiện thụ động hỗ trợ (khác với các IC tương tự bao giờ cũng cần nhiều là tụ điện, điện trở vv... chung quanh) và do điện thế ở các nơi trong mạch số chỉ hoặc ở mức cao hoặc mức thấp khiến sự thiết kế, lắp ráp và sửa chữa mạch số dễ dàng hơn mạch tương tự.

Nhược điểm của mạch số

Chỉ nói ưu điểm mà không nói nhược điểm là không công bằng. Mạch số có hai nhược điểm lớn. Trước tiên thế giới vật lý ta đang sống chủ yếu là thế giới tương tự: nhiệt độ tăng lên giảm xuống liên tục chứ không nhảy vọt, chiếc xe từ dừng đến lăn bánh rồi mới nhanh dần chỗ không đột biến, dạng sóng biểu thị tiếng nói biến thiên liên tục từ biên độ này sang biên độ khác vv... Do đó để có một lượng vật lý tự nhiên lọt được vào mạch số hay từ mạch số tác động trở lại thế giới tự nhiên phải có sự chuyển đổi. Kế đến, trong vài trường hợp mạch số có thể tốn kém hơn. Ví dụ hệ thống truyền hình số, bên cạnh nhiều ưu điểm, trước mắt sẽ tốn kém hơn nhiều so với hện thống truyền hình tươngt tự như hiện nay. Tuy nhiên trong hầu hết trường hợp mạch số mới có khả năng về tốc độ, độ chính xác, mức độ phức tạp cần thiết, ví dụ máy tính số, băng đèn quang báo.vv... Sự tiến triển nhanh chóng của công nghệ mạch tích hợp khiến cho mạch số càng ngày càng rẻ.

1.2 HỆ THỐNG SỐ NHỊ PHÂN.

Có nhiều hệ thống số. Thường dùng hàng ngày là hệ thống số thập phân hay gọi tắt hệ thống 10, dùng mười con số (digit) 0,1,2,3,..., 9. Khi số lượng lớn hơn 9 người ta dùng hay hay nhiều con số với quy ước về giá trị hạng khác nhau. Ví dụ để diễn tả số lượng bảy ngàn hai trăm năm mươi ba người ta viết 7253 theo quy ước như sau;

$$\begin{aligned} 7253_{10} &= 7000 + 200 + 50 + 3 \\ &= 7 \times 10^3 + 2 \times 10^2 + 5 \times 10^1 + 3 \times 10^0 \end{aligned}$$

Một ví dụ khác là để diễn tả số lượng bốn trăm bảy mươi lăm lẻ hai mươi tám ta viết 475,28 theo quy ước như sau:

$$\begin{aligned} 475,28 &= 400 + 70 + 5 + 0,2 + 0,08 \\ &= 4 \times 10^2 + 7 \times 10^1 + 5 \times 10^0 + 2 \times 10^{-1} + 8 \times 10^{-2}. \end{aligned}$$

Trong ngôn ngữ mạch logic và máy tính số lượng như 5; 202; 7252 là số nguyên còn số có phần lẻ (phần số) như 5,3; 202,2; 475,28 là số thực.

1.2.1.Cách gọi số nhị phân

Mạch điện tử rất khó biểu thị, xử lý và lưu trữ trực tiếp các số thập phân khác nhau, nhưng có thể hoạt động rất bảo đảm ở hai trạng thái cách biệt nhau ví dụ một công tắc điện có thể đóng (để dòng điện đi qua làm bóng đèn sáng) hay hở (để không có dòng điện đi qua, làm bóng đèn tắt), một transistor có thể ngưng dẫn hay dẫn mạnh, vv... Do đó người ta đã phát triển hệ thống số nhị phân, hay gọi tắt hệ 2, chỉ dùng hai con số 0 và 1. để diễn tả các số lượng khác nhau người ta dùng số có nhiều con số 0 và 1 với quy ước về giá trị tương tự như ở hệ thập phân nhưng bây giờ số nhân là 2^n thay vì 10^n (n là số nguyên dương hay âm). Ví dụ:

$$\begin{aligned} 1111111_2 &= 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\ &= 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255_{10} \end{aligned}$$

Một con số trong số nhị phân được gọi là một **bit** (viết tắt cho Binary) . Bit đầu (hàng đầu tận cùng bên trái) có giá trị cao nhất (ở ví dụ trên là 1×2^7) và được gọi MSB (Most Significant Bit - bit có nghĩa nhất) bit cuối (hàng tận cùng bên phải) có giá trị thấp nhất (ở ví dụ trên là 1×2^0) và được gọi LSB (Least Significant Bit - bit có nghĩa nhất) . Với số thập phân thì phải nói MSD (Most Significant Digit) và LSD (Least Significant Digit).

Khi một bit là 0 dù ở hàng nào cũng đều có trị giá là không (vì $0 \times 2^n = 0$). Ví dụ:

$$\begin{aligned} 10010101_2 &= 1 \times 2^7 + 0 + 0 + 1 + 2^2 + 0 + 1 \times 2^0 \\ &= 128 + 16 + 4 + 1 \\ &= 149_{10} \end{aligned}$$

Số nhị phân có 8 bit như ở hai ví dụ trên được gọi một **byte**, số nhị phân có 4 bit được gọi là một **nibble**. Một số nhị phân nói chung được gọi là một **word** (từ) nhưng thường được dùng để chỉ số có 16 bit. Còn số 32 bit gọi **doubleword**, 64 bit gọi **quaword**. Hoặc gọi 32 bit là word, 16 bit là **halfword**, 32 bit là doubleword.

Tưởng cũng cần biết cách gọi thứ tự các bit trong số nhị phân nhiều bit. Bit LSB (tận cùng bên phải) có thể được gọi bit 1 (bit thứ nhất) nên bit có nghĩa cao kế là bit 2 (bit thứ hai) , vv...Nhưng người ta gọi bit LSB là bit 0 (bit thứ không) nên bit có nghĩa cao kế là bit 1, vv... Ví dụ với số nhị phân 8 bit (một byte) thì sự gọi thứ tự các bit theo một trong hai cách sau:

	MSB				LSB			
Số nhị phân:	1	0	1	1	0	1	0	1
Thu tu bit :	8	7	6	5	4	3	2	1
Hoac :	7	6	5	4	3	2	1	1

Trong lãnh vực mạch số và máy tính rất thuận tiện nếu ta nhớ được sự chuyển đổi qua lại giữa số thập phân và nhị phân đối với các số thập phân 0 đến 15 như trình bày ở bảng 1.1.

Kế đến ta cũng cần biết một số lũy thừa nguyên của $2^{10} = 1024$ được gọi tắt là 1K (đọc K hay kilo), vậy trong ngôn ngữ nhị phân 1K là 1024 chứ không phải 1000 (một ngàn). Tiếp theo:

$$2^{11} = 2^1 \cdot 2^{10} = 2K$$

$$2^{12} = 2^2 \cdot 2^{10} = 4K$$

$$2^{20} = 2^{10} \cdot 2^{10} = 1K \cdot 1K = 1M \text{ (Mega)}$$

$$2^{24} = 2^4 \cdot 2^{20} = 4.1M = 4M$$

$$2^{30} = 2^{10} \cdot 2^{20} = 1K \cdot 1M = 1G \text{ (Gita)}$$

$$2^{32} = 2^2 \cdot 2^{30} = 4.G = 4G$$

Trong đó 1M (đọc Mega hay Meg) là $2^{20} = 1048576$ chứ không phải 1000000 (một triệu) và 1G (đọc Gita) là $2^{30} = 107374$ chứ không phải 1000000000 (một tỉ). Ngoài ra $2^{64} = 2^{32} \cdot 2^{32} = 16G^2 = 18.446.744073.709.551.616$.

Bảng 1.1 Thập Phân-Nhi Phân

Thập Phân	Nhi Phân
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000
17	10001
32	100000
64	1000000
128	10000000

Bảng 1.2: trị giá của 2^n

n	2^n	Viết tắt
0	1	
1	2	
2	4	
3	8	
4	16	
5	32	
6	64	
7	128	
8	256	
9	512	
10	1024	1K
11	2048	2K
12	4096	4K
13	8192	8K
14	16384	16K
15	32768	32K
16	65536	64K
20	1048576	1M
24	16777216	16M
30	1073741824	1G
32	4294967296	4G

1.2.2. Chuyển đổi thập phân sang nhị phân.

Thật ra ít khi ta cần sự chuyển đổi thập phân sang nhị phân ngoại trừ các số nhỏ như ghi ở bảng 1.1. Còn về nguyên tắc thì thực hiện phép chia hai bên liên tiếp để tính số dư. Ví dụ, số thập phân 26

Chia tiếp cho 2 : $0 \leftarrow 1 \leftarrow 3 \leftarrow 6 \leftarrow 13 \leftarrow 26$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

Số dư tương ứng : 1 1 0 1 0 LBS

Kết quả: $26_{10} = 11010$

Đầu tiên 26 chia 2 là 13 dư 0; kế đến 13 chia 2 là 6 dư 1; 6 chia 2 là 3 dư 0; 3 chia 2 là 1 dư 1; 1 chia 2 là 0 dư 1. Tập hợp các số dư theo thứ tự ngược lại là số nhị phân mong muốn.

Đối với số thập phân lớn, cách gọn hơn là tìm hiệu số liên tiếp của số thập phân với lũy thừa của 2 có giá trị thấp hơn nhưng gần số thập phân nhất. Ví dụ số thập phân 627:

$$\begin{aligned}
 \text{Hiệu số: } & 627 \quad 115 \quad 51 \quad 19 \quad 3 \quad 1 \\
 \text{Số trừ: } & 512 \quad 64 \quad 32 \quad 16 \quad 2 \quad 1 \\
 & (2^9) \quad (2^6) \quad (2^5) \quad (2^4) \quad (2^1) \quad (2^0) \\
 \text{Kết quả: } & 627_{10} = 2^9 + 2^6 + 2^5 + 2^4 + 2^1 + 2^0 \\
 & = 1001110011.
 \end{aligned}$$

Đầu tiên lũy thừa của 2 gần (nhưng nhỏ hơn) với 627 là $2^9 = 512$, hiệu số $627 - 512$ là 115. Kế đến lũy thừa của 2 gần với 115 nhất là $2^6 = 64$, hiệu số $115 - 64$ là 51, vv... Bảng 1.2 cho biết các lũy thừa nguyên dương của 2.

1.2.3.Số nhị phân biểu thị số có nghĩa lẻ.

Số có phần lẻ (số thực) là số có phần nguyên và phần phân số mà được viết ngăn cách nhau bởi dấu phẩy, gọi dấu thập phân ở hệ thập phân, ví dụ 725,475. Tương tự, ở hệ nhị phân người ta dùng dấu phẩy nhị phân để ngăn cách phần nguyên và phần phân số, ví dụ 1101,101. Cách viết số lẻ thập phân đã được trình bày ở trước, ví dụ khác là:

$$\begin{aligned}
 725,475 &= 7 \times 10^2 + 2 \times 10^1 + 5 \times 10^0 + 4 \times 10^{-1} + 7 \times 10^{-2} + 5 \times 10^{-3} \\
 &= 7 \times 100 + 2 \times 10 + 5 \times 1 + 4 \times 0,1 + 7 \times 0,01 + 5 \times 0,001 \\
 &= 700 + 20 + 5 + 0,4 + 0,07 + 0,005
 \end{aligned}$$

Tương tự, cách viết số lẻ ở hệ nhị phân có nghĩa như qua ví dụ sau:

$$\begin{aligned}
 1101,101 &= 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} \\
 &= 8 + 4 + 1 + 0,5 + 0,25 + 0,125 \\
 &= 13,625_{10}
 \end{aligned}$$

Theo quy ước quốc tế người ta dùng dấu chấm để ngăn cách phần nguyên và phần lẻ thay vì dấu phẩy.

1.2.4.Chuyển đổi số thập phân lẻ sang nhị phân.

Trước tiên xem sự chuyển đổi phần lẻ (phần phân số). Để chuyển đổi ta nhân phần lẻ của số thập phân với 2, phần nguyên nhận được, mà chỉ có thể là 1 hoặc 0 là bit MSB của phần lẻ của số nhị phân mong muốn. Tiếp theo nhân phần lẻ mới của số thập phân với 2 để tìm bit nhị phân kế và tiếp tục như vậy cho đến khi phần lẻ thập phân hết (trở thành 000...). Ví dụ số thập phân 0,6875:

$$\begin{aligned}
 0,6875 \times 2 &= 1,3750 \text{ bit nhị phân là } 2 \text{ (MSB)} \\
 0,3750 \times 2 &= 0,7500 \text{ bit nhị phân là } 0 \\
 0,7500 \times 2 &= 1,5000 \text{ bit nhị phân là } 1 \\
 0,5000 \times 2 &= 1,0000 \text{ bit nhị phân là } 1 \text{ (SLB)} \\
 \text{Kết quả: } & 0,6875_{10} = 0,1011
 \end{aligned}$$

Vì bit đầu tiên là bit MSB nên các bit càng về sau càng có nghĩa thấp tức càng có giá trị nhỏ nên trong trường hợp vài phép nhân 2 đầu không dẫn đến phần lẻ thập phân là 0 thì ta vẫn có thể dừng hay tiếp tục cho đến khi đủ số lẻ nhị phân cho sự chính xác cần thiết.

Khi số thập phân gồm phần nguyên và phần phân số ta chuyển đổi hai phần riêng biệt rồi kết hợp lại. Ví dụ:

$$\text{Đã biết: } 627_{10} = 1001110011$$

$$\text{Và: } 0,6875_{10} = 0,1011$$

$$\text{Nên: } 627,6875_{10} = 1001110011,1011.$$

1.3. TÍNH TOÁN SỐ HỌC VỚI SỐ NHỊ PHÂN.

Vấn đề tính toán số học với các số nhị phân và các mạch số thực hiện các phép tính sẽ được trình bày ở chương 10 sau này. Nhưng ngay bây giờ nên biết sơ lược về tính toán để hiểu thêm về số nhị phân.

1.3.1. Cộng và trừ số nhị phân.

Ta đã biết cộng hai số thập phân là cộng hàng đơn vị trước, nếu tổng nhỏ hơn 10 thì viết tổng, nếu tổng từ 10 trở lên thì viết hàng đơn vị và nhớ 1 cho lần cộng hàng kế trên. Việc cộng hai số nhị phân cũng tạo ra số nhớ. Trước tiên xem phép cộng hai số nhị phân 1 bit:

$$\begin{array}{r} \text{Số bị cộng:} \quad 0 \quad 0 \quad 1 \quad 1 \\ + \quad + \quad + \quad + \\ \text{Số cộng:} \quad \quad 0 \quad 1 \quad 1 \quad 0 \\ \hline \text{Tổng:} \quad \quad 0 \quad 1 \quad 1 \quad 10 \\ \quad \quad \quad \quad \quad \quad \uparrow \\ \quad \quad \quad \quad \quad \quad \text{Số nhớ (Carry)} \end{array}$$

Ở trường hợp cuối cùng, $1 + 1$ là 2 nhưng ở hệ nhị phân ta viết 0 và nhớ 1 cho hàng kế trên. Để ý là 10 có giá trị thập phân là 2. khi số nhị phân có nhiều bit ta thực hiện phép cộng ở bit có nghĩa ít nhất (LSB) trước và tiếp tục cho đến bit có nghĩa cao nhất (MSB). Ví dụ:

$$\begin{array}{r} \text{Số bị cộng:} \quad 1010 (= 10) \quad 1101 (= 13) \\ + \quad + \\ \text{Số cộng:} \quad \quad 1001 (= 9) \quad 1111 (= 15) \\ \hline \text{Tổng:} \quad \quad 10011 (= 19) \quad 11100 (= 28) \end{array}$$

Trong phép trừ nếu số bị trừ nhỏ hơn số trừ, cụ thể là khi 0 trừ đi 1, thì phải mượn 1 ở hàng cao kế mà là 2 ở hàng đang trừ và số mượn này phải trả lại cho hàng cao kế tương tự như hai phép trừ hai số thập phân. Trước tiên xem trường hợp trừ hai số bit:

$$\begin{array}{r} \text{Số bị trừ:} \quad 0 \quad 1 \quad 1 \quad 0 \\ - \quad - \quad - \quad - \\ \text{Số trừ:} \quad \quad 0 \quad 1 \quad 0 \quad 1 \\ \hline \text{Hiệu:} \quad \quad 0 \quad 0 \quad 1 \quad 11 \\ \quad \quad \quad \quad \quad \quad \uparrow \\ \quad \quad \quad \quad \quad \quad \text{Số mượn (borrow)} \end{array}$$

Để ý là 0 - 1 không phải là 11 mà là 1 với 1 là số mượn. Khi trừ hai số nhiều bit thì số mượn ở hàng nào phải được cộng vào với số trừ của hàng đó trước khi thực hiện việc trừ. Hãy tìm hiểu qua các ví dụ sau:

$$\begin{array}{r} \text{Số bị trừ:} \quad 1011 \quad 1011 \quad 1100 \\ - \quad - \quad - \end{array}$$

Số trừ:	<u>1001</u>	<u>0101</u>	<u>0111</u>
Hiệu:	0010	0110	0101

Dĩ nhiên có thể thử kết quả giống như ở phép trừ số thập phân đó là cộng hiệu với số trừ xem có bằng số bị trừ hay không.

1.3.2. Số nhị phân có dấu.

Nếu chỉ liên quan đến số dương kể cả số không thì số nhị phân và chuyển nhị phân sang thập phân là như đã biết trước. Các số nhị phân này là số không dấu, ý nói là các số chung chung và tự nhiên được hiểu là số dương. Trong tính toán số học người ta dùng dấu cộng (+) để chỉ số dương, dấu trừ (-) để chỉ số âm. Nhưng trong thế giới mạch logic (mạch số) kể cả máy tính mọi việc phải được biểu thị bởi logic 0 và logic 1, không gì khác. Do đó phải có cách để biểu thị số nhị phân có dấu, cách cơ bản là thêm 1 bit ở đầu (tận cùng bên trái) để chỉ dấu: bit 0 chỉ số dương, bit 1 chỉ số âm. Lúc bấy giờ có dấu gồm hai thành phần là **dấu** và bit đầu tiên và **độ lớn** chỉ trị số tuyệt đối của giá trị là các bit còn lại. Đây là cách biểu thị dấu - độ lớn của nhị phân. Ví dụ:

$$0 \ 10101 = +21$$

$$1 \ 10101 = -21$$

dấu độ lớn

dấu độ lớn.

Quy ước này dẫn đến hai biểu thị khác nhau cho không:

$$0 \ 00000 = +0$$

$$1 \ 00000 = -0$$

↑
dấu

↑
dấu

Dĩ nhiên để tránh nhầm lẫn giữa bit dấu và các bit độ lớn người ta phải quy định số bit độ lớn trước để thêm các số 0 ở trước cho đủ số bit quy định. Ví dụ quy định số có dấu là 8 bit trong đó một bit dấu và 7 bit độ lớn, thì để diễn tả +21 và -21 ta phải viết:

$$+21 = \underline{0} \ 0010101$$

$$-21 = \underline{1} \ 0010101$$

Bit dấu đã được gạch dưới để chỉ bit dấu, còn trong viết bình thường không có gạch dưới (bit dấu và các bit độ lớn được viết liên tục).

Cách biểu thị dấu - độ lớn cho số nhị phân có dấu như trên không cho phép thực hiện các phép tính vì kết quả thường sai. Ví dụ đối với số có dấu 5 bit:

01000 (+ 8)	01000 (+ 8)	11000 (- 8)
+	+	+
<u>01010 (+ 10)</u>	<u>10010 (- 2)</u>	<u>10110 (- 6)</u>
10010 (- 2)	11010 (- 10)	1 01110 (+ 14)
sai	sai	bỏ sai

Ở bài toán đầu nếu bit đầu được hiểu là bit MSB của độ lớn thay vì bit dấu thì kết quả đúng.

1.3.3.Số bù1.

Về phương diện mạch điện tử thì một ý đồ của sự biểu thị dấu - độ lớn là để biến phép trừ thành phép cộng, ví dụ như $8 - 2 = 8 + (-2)$, nhưng rất tiếc là ý đồ này không đạt được vì kết quả thường là sai (ở ví dụ trên là sai nhưng cũng có trường hợp đúng). Do đó, để có thể thực hiện các phép tính số học với số có dấu nhất là trong phạm vi mạch số (máy tính) người ta phải tìm các cách biểu thị khác cho số âm.

Số bù 9 của một số thập phân được định nghĩa qua ví dụ sau:

- Số bù 9 của 8 là $9 - 8 = 1$
- Số bù 9 của 7413 là $9999 - 7413 = 2586$

Tương tự, số bù 1 của một số nhị phân có n bit là hiệu số của 111...(n bit) và số nhị phân đó. Ví dụ:

$$\begin{array}{r} \text{Số bù 1 của 10 là:} \quad 11 \\ - \\ 10 \\ \hline 01 \\ \text{Số bù 1 của 101101 là:} \quad 111111 \\ - \\ 101101 \\ \hline 010010 \end{array}$$

Nhận xét là khi trừ ta chỉ gặp hai trường hợp đó là 1-1 mà là 0 hoặc 1-0 mà là 1. Do đó số bù 1 của một số nhị phân nhận được bằng cách đổi 0 thành 1 và 1 thành 0 (đảo) tức lấy bù từng bit. Ví dụ đối với số 4 bit:

Số bù 1 của 0010 (2) là: 1101

Số bù 1 của 1000 (8) là: 0111

Để thực hiện các phép toán số học người ta vẫn biểu thị số dương ở dạng dấu - độ lớn đã nói ở trước với bit dấu là 0 nhưng biểu thị số âm ở dạng dấu - bù 1 của độ lớn, dĩ nhiên bit dấu là 1. Ví dụ đối với số có dấu 5 bit gồm bit dấu MSB và 4 bit độ lớn:

$$\begin{array}{r} 2 = 0010 \\ + 2 = \underline{0} \quad 0010 \\ - 2 = \underline{1} \quad \text{bù 1 (0010)} = \underline{1} (1101) = \underline{1} \quad 1101 \end{array}$$

Để ý là bit dấu của số dương là 0 và bù của nó là 1 nên ta có thể nói số âm được biểu thị bởi số bù 1 của số dương tương ứng. Ví dụ đối với số 5 bit:

$$\begin{array}{r} - 2 = \text{bù 1 (+2)} = \text{bù 1 (0 0010)} = 1 \quad 1101 \\ - 8 = \text{bù 1 (+8)} = \text{bù 1 (0 1000)} = 1 \quad 0111 \end{array}$$

Ở hai dòng trên dấu = đầu tiên có nghĩa là " được biểu thị bởi "

Ý đồ là thực hiện phép trừ hai số nhị phân A - B bằng cách thực hiện phép cộng:

$$A - B = A + (-B) = A + \text{bù 1 (+B)}$$

Bây giờ xem phép trừ được tiến hành ra sao qua ví dụ:

$$8 - 2 = 8 + (-2) = 8 + \text{bù } 1(+2):$$

$$\underline{0} \ 1000 \quad (+8)$$

$$\underline{0} \ 1101 \quad (\text{bù } 1 \text{ của } +2)$$

$$0 \ \underline{0} \ 0101 \quad (+5)$$

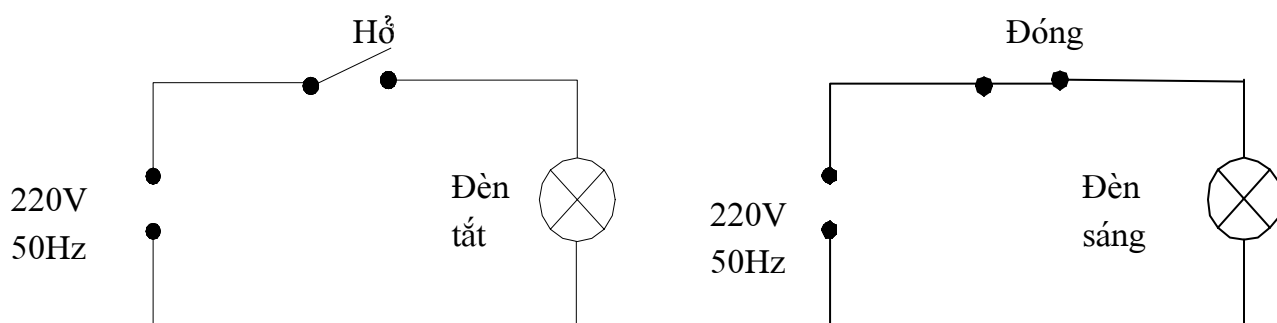
bỏ sai

CHƯƠNG II

Cổng Logic và đại số Boole

Mạch logic (hay mạch số) xử lý dữ liệu nhị phân. Người ta chỉ định nghĩa một số hàm logic cơ bản, mà về phương diện mạch gọi là các cổng cơ bản. Các hàm logic phức tạp hơn được thiết lập từ các hàm cơ bản này. Đại số Boole được dùng để diễn tả mạch logic theo đại số. Nó là công cụ toán học để phân tích, thiết kế mạch logic. Ở cuối chương là bản đồ Karnaugh giúp đơn giản biểu thức logic một cách có hệ thống.

2.1 TRẠNG THÁI LOGIC 1 VÀ 0

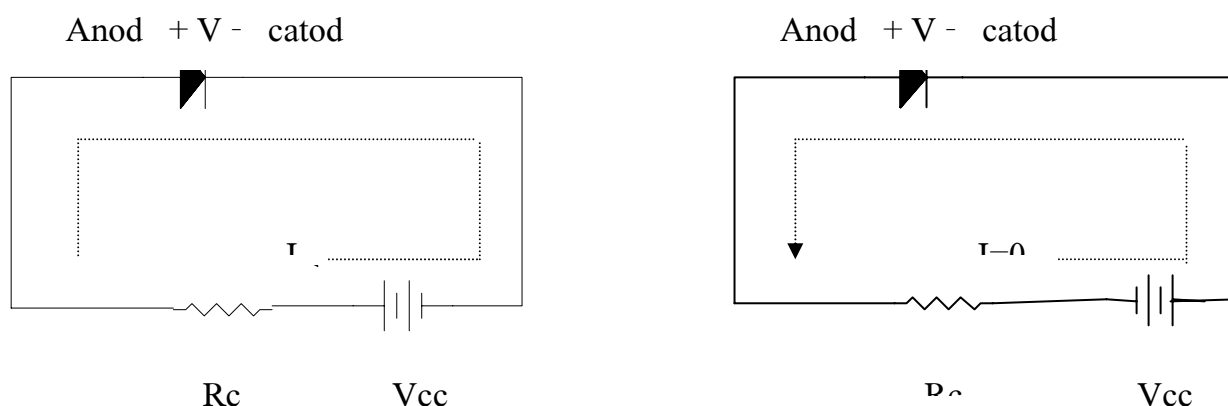


Hình 2.1: Hai trạng thái rõ rệt của công tắc điện

Một công tắc điện (bật điện) như dùng phổ biến trong nhà có hai trạng thái rõ rệt (hình 2.1):

- Hở : điện không qua nên đèn tắt.
- Đóng : điện qua làm đèn sáng.

Kế đến xem **diode bán dẫn** là một linh kiện điện tử có hai cực gọi là anod và catod (hình 2.2). Diode được phân cực thuận khi anod được nối đến cực dương của nguồn điện một chiều Vcc (có hiệu thế 1 volt trở lên) và catod đến cực âm của Vcc (hình 2.2a).

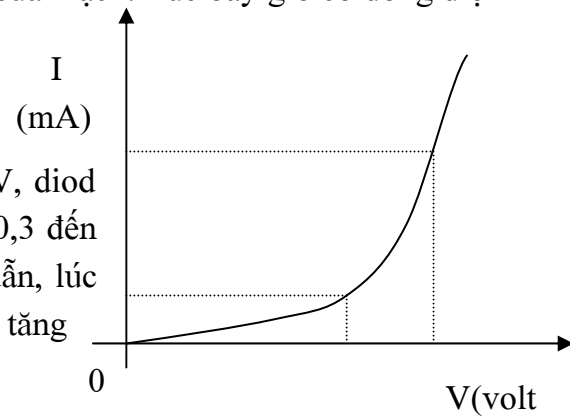


(a) Diod được phân cực thuận:
Diod dẫn điện

(b) Diod được phân cực ngược
diod không dẫn điện

Hình 2.2: Diod bán dẫn và sự phân cực

Điện trở R_C giới hạn dòng điện trong mạch và được gọi là tải của mạch. Lúc bấy giờ có dòng điện I chạy qua diod và một hiệu thế V xuất hiện ngang qua diod. Hiệu thế V thay đổi theo dòng I , nếu chất bán dẫn là silicium (ký hiệu Si) thì sự biến thiên của I theo V , gọi đặc tính I - V , là như hình 2.3. Ở điện thế ngưỡng V , xấp xỉ $0,6V$, diod xem như bắt đầu dẫn điện, lúc bấy giờ dòng điện là khoảng $0,3$ đến $0,5mA$ (miliampe). Ở ngưỡng V_T xấp xỉ $0,7V$ diod thực sự dẫn, lúc bấy giờ dòng điện là khoảng 3 đến $5 mA$, sau đó dòng điện tăng nhanh.



Khi diod không được phân cực (tức khi $V_{CC} = 0$) hay khi diod được phân cực nghịch (hình 2.2b), trong mạch không có dòng điện ($I = 0$) tức diod ngưng dẫn. Như vậy người ta có thể khống chế diod để nó hoạt động ở hai trạng thái khác nhau rõ rệt :

- Không phân cực (hay phân cực nghịch): diod ngưng dẫn.
- Phân cực thuận ở dòng trên vài mA : diod dẫn mạnh và hiệu thế ngang qua diod là khoảng $0,7$ volt trở lên (đến tối đa khoảng $0,85V$)

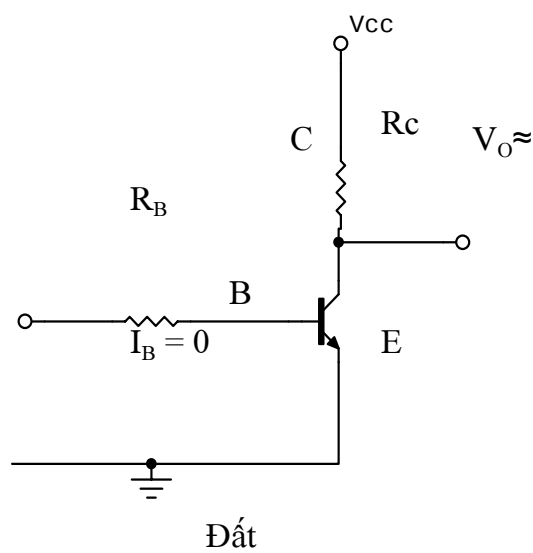
Hình 2.3: Đặc tính I - V tiêu biểu của

Bây giờ xem linh kiện điện tử quan trọng hơn : *của diod Si phân cực thuận*
Transistor lưỡng cực (Bipolar Junction Transistor - BJT) mà thường được gọi tắt là transistor. Transistor có ba cực gọi là cực nền (Base) B, cực thu (Collector) C, cực phát (Emitter) E (hình 2.4). Điện trở R_B để giới hạn dòng nền I_B , điện trở R_C nơi có dòng thu I_C chạy qua là tải ở ngõ ra. Ở transistor có hệ thức dòng điện cơ bản:

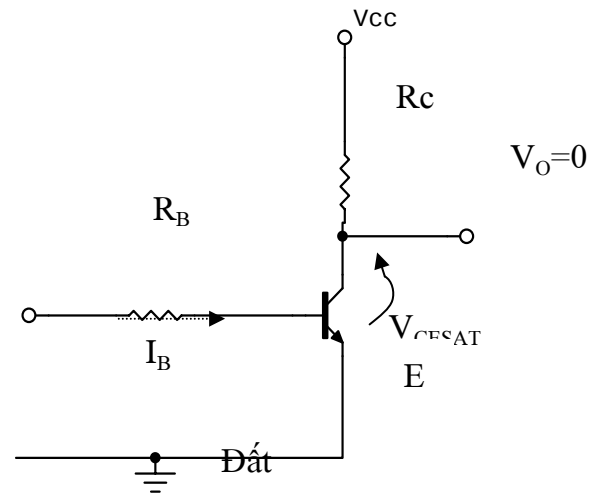
$$I_C = \beta I_B,$$

Trong đó β là **hệ số khuếch đại dòng** (β còn gọi h_{FE}), thường β là vài chục đến 200 . Khi điện thế ở ngõ vào (so với đất) $V_I = 0$ thì dòng nền $I_B = 0$ khiến dòng thu $I_C = 0$ và điện thế ở ngõ ra (so với đất) là:

$$V_O = V_{CC} - I_C R_C = V_{CC} - 0 = V_{CC}$$



(a) không phân cực: Transistor
ngưng dẫn điện thế ra $= V_{CC}$



(b) phân cực thuận mạch : Transistor
dẫn mạch, điện thế ra $\approx 0V$

Hình 2.4: Hai trạng thái của transistor lưỡng cực

Khi điện thế vào $V_I \approx V_{CC}$ thì dòng nền I_B lớn khiến transistor dẫn bão hoà (ý nói dẫn mạnh và dòng I_C không thể tăng hơn nữa dù I_B có tăng thêm), lúc bấy giờ

Thấy nếu đảo B thành $\bar{B}$ rồi HOẶC với A cho kết quả đúng (hình 2.18b và c). Dĩ nhiên cũng có thể đảo $\bar{A}$ thành A rồi HOẶC với B. Ở mục 2.7 ta sẽ biết cách thiết kế có tính toán học hơn.

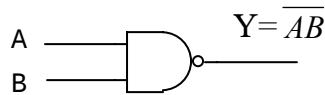
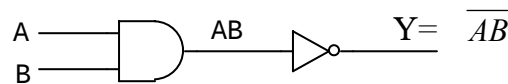
2.3 HÀM (VÀ CÔNG) LOGIC KHÔNG – VÀ (NAND), KHÔNG-HOẶC (NOR)

VÀ (AND) theo sau bởi KHÔNG (NOT) là KHÔNG – VÀ (NAND). Xem trường hợp có 2 biến số A và B. Ra ở cổng VÀ là AB nên ra cổng KHÔNG tiếp theo là đảo của AB tức là $\overline{AB}$ (hình 2.19):

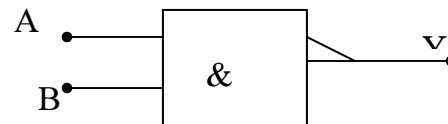
$$Y = \overline{AB}$$

Về kí hiệu thay vì dùng cổng KHÔNG người ta chỉ cần thêm vòng tròn phủ định nhỏ sau cổng VÀ, hình 2.19 cũng cho thấy kí hiệu thường mà rất phổ biến và kí hiệu IEEE/ANSI ít được dùng hơn.

Vào		Ra
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0



Ký hiệu thường

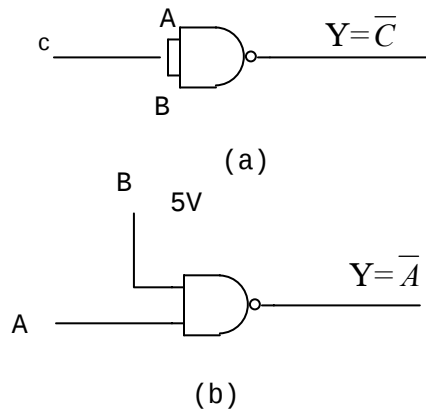


Hình 2.19: cổng KHÔNG – VÀ (NAND) và bảng sự thật

Dĩ nhiên cổng NAND cũng áp dụng trong trường hợp nhiều ngõ vào, ví dụ khi có 3 ngõ vào A,B,C thì ra là:

$$Y = \overline{ABC}$$

Hình 2.20 cho thấy cách tạo cổng NOT từ cổng NAND. Ở hình (a) các ngõ vào của NAND được nối chung tức là A=B. Bảng sự thật của NAND cho thấy khi A=B=0 thì ra là Y=1, Khi A=B=1 ra là Y=0 nên là cổng NOT. Ở hình (b) vì B được nối lên cao (B=1) nên bảng sự thật của NAND cho thấy khi A=0 (và B=1) ra là Y=1 và khi A=1 (và B=1) ra là Y=0, vậy là cổng NOT, về phương diện logic thì như trình bày, còn phương diện mạch thì hai trường hợp (a),(b) có khác nhau chút ít. Tuy nhiên tạm bỏ qua chi tiết tinh vi và xem cả 2 trường hợp đều thực hiện được cổng NOT như nhau.



Hình 2.20 cách tạo cổng NOT cổng NAND

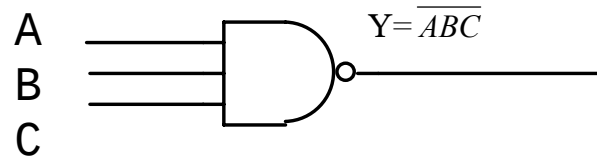
Ví dụ 2.3.1

lập bảng hàm logic (bảng sự thật) của cổng NAND ba ngõ vào

Giải

Gọi đầu vào A,B,C và đầu ra là Y. Trước tiên tạo tất cả các tổ hợp khả dĩ của A,B,C như đã biết, có 8 tổ hợp. Kế đến tạo logic ABC rồi lấy đảo, ở bảng sự thật, bước ABC được hiểu nhầm. Cách khác là nhớ quy luật của cổng NAND “ra là 0 khi tất cả các ngõ vào là 1 và ra là 1 ở trường hợp còn lại” để viết ngay kết quả ra.

Vào			Ra
A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0



Hình 2.21: ví dụ 2.3.1

Ví dụ 2.3.2

chúng tỏ khi một ngõ vào của cổng NAND ở thấp (logic 0) thì tín hiệu ở các ngõ vào còn lại không truyền qua cổng NAND được.

Giải:

Xem cổng NAND hai ngõ vào (hình 2.19). Khi ngõ vào B được giữ ở thấp (B=0) thì dù A=0 hay A=1 thì ra cũng là 1 có nghĩa sự thay đổi logic ở ngõ vào A không ảnh hưởng đến ngõ ra hay nói cách khác tín hiệu vào ở A không truyền qua cổng được (lúc bây giờ người ta nói cổng bị đóng). Khi cổng NAND có ba ngõ vào (hình 2.21) cũng vậy. Ví dụ khi A=0 thì ra Y=1 bất chấp trạng thái logic của B và C.

Ví dụ 2.3.3

Bằng cách lập bảng sự thật để tìm liên hệ giữa A+B và AB

Giải :

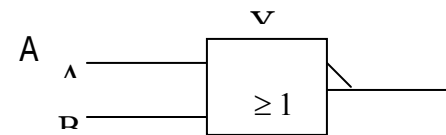
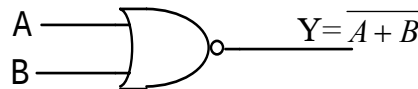
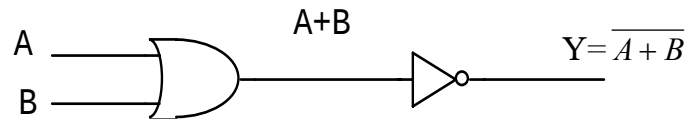
Ta dễ dàng lập bảng sự thật của $A+B$ và $\overline{AB}$ so sánh thấy $A+B$ và $\overline{AB}$ bằng nhau ở hai trường hợp giữa và khác nhau ở trường hợp đầu và cuối. Như vậy không có hệ thức logic nào giữa $A+B$ và $\overline{AB}$

Hàm (cổng) logic KHÔNG- HOẶC (NOR)

HOẶC (OR) theo sau bởi KHÔNG (NOT) là KHÔNG- HOẶC (NOR). Xem trường hợp hai ngõ vào là A,B ra ở cổng HOẶC là $A+B$ nên ra ở cổng KHÔNG tiếp theo là đảo của $A+B$ tức $\overline{A+B}$: $Y = \overline{A+B}$

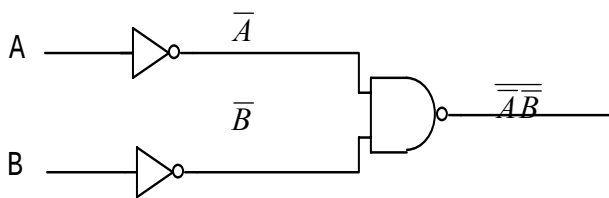
Về ký hiệu người ta thể cổng KHÔNG bằng vòng tròn phủ định nhỏ sau cổng HOẶC (xem hình 2.22). Cổng NOR cũng áp dụng cho trường hợp nhiều ngõ vào.

VÀO		RA
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0



Hình 2.22; cổng KHÔNG - HOẶC (NOR) và bảng sự thật

Ví dụ 2.3.4



Hình 2.23: ví dụ 2.2.4

Nếu đảo ngõ vào A và B rồi mới đưa đến cổng NAND thì mạch tương đương cổng gì ?

Hình 2.2.3: ví dụ 2.2.3

Giải

Lập bảng sự thật gồm cột A và B rồi lấy đảo ngành $\overline{A}$, $\overline{B}$. Thực hiện hàm NAND đối với $\overline{A}$, $\overline{B}$ ta lấy logic ra Y giống như ra cổng OR

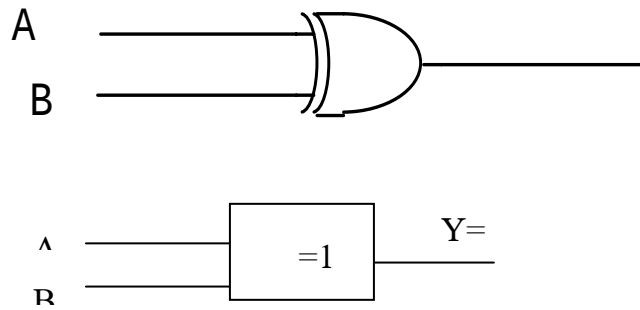
với vào là A, B. Vậy cổng tương đương là OR.

2.4 HÀM (VÀ CỔNG) LOGIC EX-OR VÀ EX-NOR

Hàm HOẶC ở trước được gọi HOẶC BAO GỒM (Inclusive OR), nó không đúng như nghĩa “hoặc” hàng ngày và nó không giải quyết được bài toán cộng nhị phân. Lý do là khi cả A và B là 1 thì $Y = 1$ thay vì là 0. Mặc dù HOẶC như vậy vẫn có nghĩa thực tế và vẫn được dùng nhưng người ta phải định nghĩa một HOẶC LOẠI TRỪ (Exclusive OR gọi tắt là EX-OR hay EXOR hay XOR) ý nói loại trừ trường hợp cuối nghĩa là lúc bấy giờ khi cả A và B là 1 thì $Y = 0$ (xem hình 2.2.4) ký hiệu là:

$$Y = A \oplus B$$

Biến (ngõ vào)		Hàm (ngõ ra)
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0



Hình 2.26: EX-NOR(hay XNOR)

Có thể biểu thị định nghĩa của XOR như cho ở bảng sự thật theo nhiều cách mà dẫn đến nhiều mạch khác nhau, nhưng dĩ nhiên là tương đương nhau. Trước tiên có hiểu là “Y=1 khi A=1 và A=0 hoặc A=0 và B=1” mà diễn tả thành biểu thức là như thấy ở hình 2.25. Hình này cũng cho thấy mạch logic, để ý là cổng NOT được thể bằng một vòng phủ định nhỏ. Một cách hiểu khác là “Y=1 khi HOẶC A=1, HOẶC B=1 VÀ KHÔNG phải A VÀ B đều bằng 1” mà diễn tả thành biểu thức là:

$$Y = (A+B) \cdot (\overline{AB})$$

KHÔNG

↓

↑ ↑ ↑

HOẶC VÀ VÀ

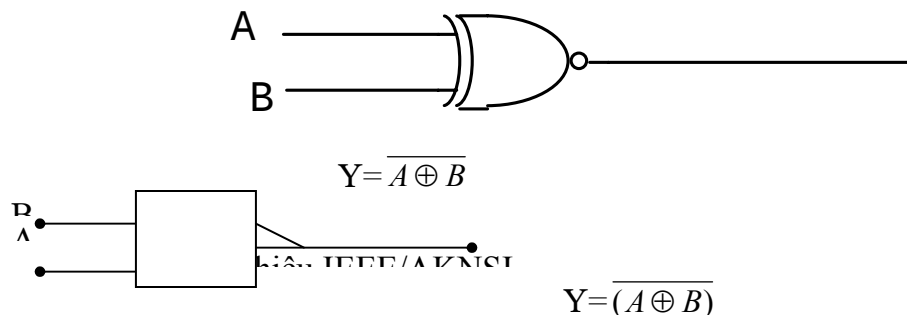
Từ biểu thức này có thể vẽ mạch thực hiện dễ dàng.

EX-OR theo sau bởi NOT là EX-NOR

Hoạt động logic của EX-NOR đảo lại so với EX-OR:

Ký hiệu thường

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1



Ví dụ 2.4.1

Bằng cách lập bảng sự thật nghiệm lại $Y=(A+B).(\overline{AB})$ chính là hàm EX-OR.

Giải:

Trước tiên lập các tổ hợp của A, B, Kế đến là logic $A+B$, AB , $\overline{AB}$, và sau cùng $(A+B).(\overline{AB})$. Bảng sự thật cho thấy logic sau cùng chính là EX-OR đối với A,B vậy đã chứng minh được.

A	B	$A+B$	AB	$\overline{AB}$	$(A+B).(\overline{AB})$
0	0	0	0	1	0
0	1	1	0	1	1
1	0	1	0	1	1
1	1	1	1	0	0

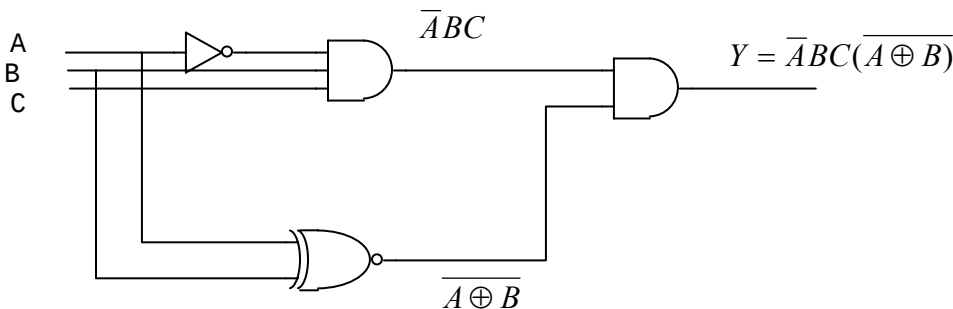
Ví dụ 2.4.2

Thực hiện mạch logic mô tả bởi hệ thức logic

$$Y = \overline{A}BC(\overline{A \oplus B})$$

Giải

Cách thực hiện là đảo A để có $\overline{A}$ rồi VÀ với B,C. Mặt khác cho A và B qua EX-NOR để có $\overline{A \oplus B}$. Sau cùng VÀ $\overline{A}BC$ với $\overline{A \oplus B}$.



Hình 2.27: ví dụ 2.4.2

Mạch tích hợp logic

Ở trước, cổng logic được trình bày gần như chỉ là các kí hiệu toán học hơn là một thực thể vật lý. Thực ra các cổng là các linh kiện điện tử, với một cấu trúc mạch cụ thể có các đặc tính kỹ thuật nhất định. Các cổng logic và các mạch logic nói chung được chế tạo ở dạng mạch tích hợp (IC) rất tiện lợi cho việc sử dụng.

Chương 3

Cở Sở Đại Số Logic

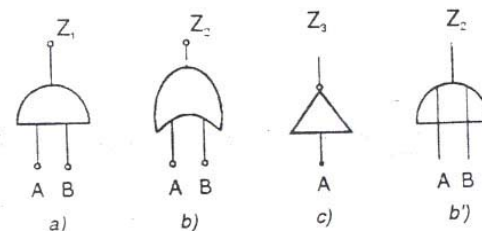
3.1> Khái niệm cơ bản, công thức và định lý:

Đại số logic do George Boole, nhà toán học nước Anh, sáng tạo vào giữa thế kỷ XIX - so với đại số thường đại số logic đơn giản hơn nhiều. Tuy đại số logic cũng dùng chữ biểu thị biến số nhưng biến số logic chỉ lấy giá trị rất đơn giản, 1 và 0, không có giá trị thứ ba nào nữa. Hơn nữa, 0 và 1 ở đại số logic không chỉ biểu thị số lượng to nhỏ cụ thể mà chủ yếu là để biểu thị hai trạng thái logic khác nhau. (ví dụ dùng 1 và 0 để biểu thị: đúng và sai; thật và giả; cao và thấp; có và không; mở và đóng.v.v...). Trong đại số logic có một quy tắc giống với đại số thường nhưng lại có một số quy tắc khác hoàn toàn khác với đại số thường, chúng ta cần lưu ý phân biệt trong quá trình học tập.

3.1.1> Phép toán logic và hàm logic cơ bản:

1/ Phép toán logic cơ bản

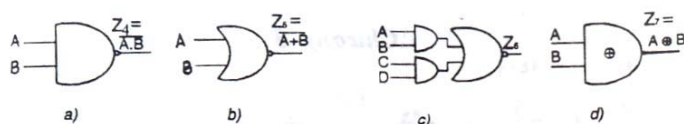
Như ta đã biết, quan hệ logic cơ bản nhất chỉ có 3 loại: Và, hoặc, phủ định. Vậy nên trong đại số logic cũng chỉ có tương ứng 3 phép toán logic cơ bản nhất là: nhân logic - và, cộng logic - hoặc, đảo logic - phủ định. Các mạch điện thực hiện 3 phép toán cơ



bản nhất, tương ứng là các cổng và (and); hoặc (or); đảo (not).

Hình 3-1-1. Ký hiệu logic của các cổng cơ bản

Ngoài 3 phép toán logic cơ bản nhất trên đây chúng ta còn thường xuyên gặp các phép toán logic sau: Và - phủ định, hoặc - phủ định, và - hoặc - phủ định, cộng với phép loại trừ... Mạch điện tương ứng thực hiện các phép toán trên, theo thứ tự các cổng: NAND, NOR, NORAND, XOR biểu thị trên hình 3-1-2



Hình 3-1-2. Ký hiệu logic các cổng logic thường dùng

Tương ứng:

$$\text{Hình 3-1-2a: cổng NAND} \quad Z_4 = \overline{A \cdot B} \quad (3-1-4)$$

$$\text{Hình 3-1-2b: cổng NOR} \quad Z_5 = \overline{A+B} \quad (3-1-5)$$

$$\text{Hình 3-1-2c: cổng NORAND} \quad Z_6 = \overline{A \cdot B + C \cdot D} \quad (3-1-6)$$

$$\text{Hình 3-1-2d: cổng XOR} \quad Z_7 = A \oplus B \quad (3-1-7)$$

2/ Biến logic và hàm logic:

Các công thức (3-1-1) + (3-1-7) là các biểu thức logic, trong đó A,B,C,D là các biến logic đầu vào, Z là biến logic đầu ra, dấu gạch trên biến logic biểu thị hàm logic đảo của biến đó. Công thức (3-1-1) biểu thị quan hệ VÀ giữa A với B, Z₁ là ham VÀ của các biến A và B. Công thức (3-1-2) biểu thị quan hệ HOẶC giữa A với B, Z₂ là hàm HOẶC của các biến A và B. Công thức (3-1-3) biểu thị Z₃ là hàm ĐẢO của biến A. Công thức (3-1-7) biểu thị quan hệ CỘNG VỚI PHÉP LOẠI TRỪ giữa A với B, Z₇ hàm XOR của các biến A và B.

Nói chung, sau khi đã xác định giá trị các biến đầu vào A, B, C... thì giá trị biến đầu ra Z cũng được xác định theo một cách đơn trị. Vậy ta gọi Z là hàm số logic của A, B, C..., và ta có thể viết:

$$Z = F(A, B, C, \dots)$$

Trong đại số logic, biến số và hàm số đều chỉ lấy hai giá trị; thường dùng 0 và 1 biểu thị. Điều đó có cở sở trong quan hệ nhân quả của các sự kiện. Mỗi biến số biểu thị một điều kiện để sự kiện có thể phát sinh. Điều kiện đó chỉ có thể có hay không. Hàm số biểu thị bản thân sự kiện đó phát sinh hay không. Số 0 và 1 biểu thị ký hiệu của hai khả năng đối lập nhau đó và trong đa số trường hợp, chúng không có ý nghĩa số lượng nữa.

3.1.2> Công thức và định lý:

1/ Quan hệ giữa các hằng số:

$$\text{Công thức 1:} \quad 0 \cdot 0 = 0 \quad (3-1-8)$$

$$\text{Công thức 1':} \quad 1 + 1 = 1 \quad (3-1-9)$$

$$\text{Công thức 2:} \quad 0 \cdot 1 = 0 \quad (3-1-10)$$

$$\text{Công thức 2':} \quad 1 + 0 = 1 \quad (3-1-11)$$

$$\text{Công thức 3:} \quad 1 \cdot 1 = 1 \quad (3-1-12)$$

$$\text{Công thức 3':} \quad \underline{0} + 0 = 0 \quad (3-1-13)$$

$$\text{Công thức 4:} \quad \underline{0} = 1 \quad (3-1-14)$$

$$\text{Công thức 4':} \quad \underline{1} = 0 \quad (3-1-15)$$

Những quan hệ trên đây giữa hai hàng số làm tiền đề của đại số logic. Nghĩa là, chúng là các quy tắc phép toán cơ bản đối với tư duy logic.

2/ Quan hệ giữa biến số và hằng số:

$$\text{Công thức 5:} \quad A \cdot 1 = A \quad (3-1-16)$$

$$\text{Công thức 5':} \quad A + 0 = A \quad (3-1-17)$$

$$\text{Công thức 6:} \quad A \cdot 0 = 0 \quad (3-1-18)$$

$$\text{Công thức 6':} \quad A + 1 = 1 \quad (3-1-19)$$

—

$$\text{Công thức 7:} \quad A \cdot A = 0 \quad (3-1-20)$$

—

$$\text{Công thức 7':} \quad A + A = 1 \quad (3-1-21)$$

3/ Các định lý tương tự đại số thường:

Luật giao hoán:

$$\text{Công thức 8:} \quad A \cdot B = B \cdot A \quad (3-1-22)$$

$$\text{Công thức 8':} \quad A + B = B + A \quad (3-1-23)$$

Luật kết hợp:

$$\text{Công thức 9:} \quad (A \cdot B) \cdot C = A \cdot (B \cdot C) \quad (3-1-24)$$

$$\text{Công thức 9':} \quad (A + B) + C = A + (B + C) \quad (3-1-25)$$

Luật phân phối:

$$\text{Công thức 10:} \quad A \cdot (B + C) = A \cdot B + A \cdot C \quad (3-1-26)$$

$$\text{Công thức 10':} \quad A \cdot BC = (A + B) \cdot (A + C) \quad (3-1-27)$$

4/ Các định lý đặc thù chỉ có trong đại số logic

Luật đồng nhất:

$$\text{Công thức 11:} \quad A \cdot A = A \quad (3-1-28)$$

$$\text{Công thức 11':} \quad A + A = A \quad (3-1-29)$$

Định lý De Morgan:

$$\text{Công thức 12:} \quad A \cdot B = \overline{\overline{A} + \overline{B}} \quad (3-1-30)$$

$$\text{Công thức 12':} \quad A + B = \overline{\overline{A} \cdot \overline{B}} \quad (3-1-31)$$

Luật hoàn nguyên

$$\text{Công thức 13} \quad A = A \quad (3-1-32)$$

Phương pháp chứng minh các công thức trên là lập bảng tất cả các giá trị có thể của các biến và tính tương ứng với vế phải, vế trái riêng rẽ. Nếu đẳng thức giữa hai vế tồn tại với tất cả các giá trị có thể thì công thức là đúng. Công thức 5 và công thức 13 rất dễ chứng minh. Dưới đây sẽ chứng minh làm mẫu các công thức 10' và công thức 12

Ví dụ 3-1-1. Chứng minh công thức 10'

$$A+B \times C = (A+B) \times (A+C)$$

Giải: lập bảng tất cả các giá trị có thể của biến và tính như sau:

Bảng 3-1-1:

A	B	C	B x C	A+B x C	A+B	A+C	(A+B) (A+C)
0	0	0	0	0		0	0
0	0	1	0	0		1	0
0	1	0	0	0		0	0
0	1	1	1	1		1	1
1	0	0	0	1		1	1
1	0	1	0	1		1	1
1	1	0	0	1		1	1
1	1	1	1	1		1	1

Tất cả các giá trị của 3 biến A,B,C tạo thành 8 tổ hợp. Bảng chân lí của hàm $A + B \times C$ trùng với bản chân lí của hàm $(A+B)(A+C)$. Vậy công thức $A + B \times C = (A+B) (A+C)$ đã được chứng minh.

5) 3 quy tắc về đẳng thức

a) Quy tắc thay thế

Trong bất kì đẳng thức logic nào, nếu thay thế một biến nào đó bằng một hàm số thì đẳng thức vẫn thiết lập.

Quy tắc này có ứng dụng rất lớn trong biến đổi công thức để tạo ra công thức mới từ một công thức đã biết, mở rộng phạm vi ứng dụng của công thức đã biết.

Ví dụ:

b) Quy tắc tìm đảo của một hàm số

Z là đảo của hàm số Z sẽ có được từ Z bằng cách đổi dấu “.” thành dấu “+” ; “+” thành dấu “.”; “0” thành “1”, “1” thành “0”, biến số thành đảo của biến số đó, đảo biến số thành nguyên biến số.

Ví dụ:

Khi tìm đảo của một hàm số, những gạch ngang nào (biểu thị phép toán đảo) ở trên nhiều biến thì vẫn giữ nguyên. Cũng cần chú ý thứ tự ưu tiên xử lý các kí hiệu: dấu móc, dấu nhân, dấu cộng. Ví dụ , theo thứ tự phép tính phải làm phép nhân AxB và CxD trước, sau mới tới phép cộng giữa chúng. Vậy thứ tự xử lý kí hiệu để tìm đảo sẽ dẫn tới kết quả

c) Quy tắc đối ngẫu

Hàm Z và hàm Z' gọi là đối ngẫu, khi các dấu “+” và “.”, với các giá trị “1” và “0” đổi chỗ cho nhau một cách tương ứng

Vì đối ngẫu là tương hỗ, nên nếu một đẳng thức đã tồn tại đối với biểu thức vế trái và biểu thức vế phải, thì đối ngẫu của vế trái và đối ngẫu của vế phải cũng là một đẳng thức.

Cần lưu ý thứ tự ưu tiên xử lý khi tìm biểu thức đối ngẫu

áp dụng quy tắc đối ngẫu có thể làm cho số công thức cần chứng minh giảm đi một nửa. Sau khi đã chứng minh hai biểu thức bằng nhau, căn cứ quy tắc đối ngẫu, các đối ngẫu của đẳng thức đã chứng minh cũng phải bằng nhau. Vậy nên, khi giới thiệu những công thức sau đây, chúng ta sẽ không đưa ra các công thức dạng đối ngẫu của chúng.

6) Một số công thức thường dùng

7) Những công thức XOR (phép cộng với sự loại trừ)

Định nghĩa phép XOR:

Hàm logic XOR = 1 khi các biến A,B lấy các giá trị khác nhau,

Và XOR = 0 khi các biến A, B lấy các giá trị bằng nhau.

Tên hàm XOR, vì vậy, mang ý nghĩa dị hoặc, hoặc tuyệt đối

Đảo của XOR là:

Hàm $AxB = 1$ khi các biến A,B lấy các giá trị bằng nhau

$AxB = 0$ khi các biến A,B lấy các giá trị khác nhau

AxB có tên hàm tương đương

1. Luật giao hoán: $A \oplus B = B \oplus A$

2. Luật kết hợp: $(A \oplus B) \oplus C = A \oplus (B \oplus C)$

3. Luật phân phối: $A(B \oplus C) = AxB \oplus AxC$

4. Các phép toán của biến và hằng số:

5. Luật đổi chỗ nhân quả

Nếu $A \oplus B = C$

Thì $A \oplus C = B$ và $B \oplus C = A$

Chứng minh:

Vì $A \oplus B = C$

$$\begin{aligned} \text{Nên } A \oplus B \oplus B &= C \oplus B \\ A \oplus 0 &= B \oplus C \\ A &= B \oplus C \end{aligned}$$

8. Định lý triển khai

3.2 Các phương pháp biểu thị hàm logic

Khi nghiên cứu và xử lý những vấn đề logic, ta có thể dùng những phương pháp khác nhau để biểu thị hàm logic tùy theo đặc điểm của hàm logic xét. Thường dùng 4 phương pháp. Đó là bảng chân lí, biểu thức logic, bảng Karnaugh và sơ đồ logic. Chúng ta không những cần nắm vững từng phương pháp, mà còn phải thành thạo chuyển đổi từ phương pháp này sang phương pháp khác.

3.2.1 Bảng chân lí

Bảng chân lí bằng miêu tả quan hệ giữa các giá trị của hàm số tương ứng với mọi giá trị có thể của biến số

1) Phương pháp liệt kê thành bảng chân lí

Mỗi biến đầu vào có thể lấy 2 giá trị 1 và 0, nếu có n biến đầu vào thì có 2^n tổ hợp các giá trị khác nhau của chúng. Để nhận được bảng chân lí, ta phải liệt kê tất cả các tổ hợp giá trị của biến đầu vào và giá trị xác định của hàm đầu ra tương ứng với từng tổ hợp đó.

Ví dụ 3-2-1: Hãy kê bảng chân lí của hàm số sau:

$$Z = AB + BC + CA$$

Giải: có 3 biến đầu vào, tức là có 8 tổ hợp các giá trị của chúng. Thay giá trị của mỗi tổ hợp vào hàm số và tính ra giá trị tương ứng, rồi liệt kê thành bảng 3-2-1 (Nói chung, để khỏi bỏ sót, để khỏi trùng lặp, thường sắp xếp thứ tự các giá trị biến vào theo tuần tự số đếm nhị phân).

Bảng 3-2-1:

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1

1	1	1	1
---	---	---	---

Ví dụ 3-2-2: một bóng đèn đường cần đóng, ngắt độc lập ở 4 nơi khác nhau. Hãy viết bản chân lí của hàm logic đó.

Giải: gọi A,B,C là chuyển mạch đóng ngắt ở 4 nơi, đóng điện thì các biến lấy giá trị 1, ngắt điện thì các biến lấy giá trị 0. Gọi Z là trạng thái đèn được điều khiển, đèn sáng $Z=1$, đèn tắt $Z=0$. Sau khi suy xét kĩ, ta kê được bảng chân lí 3-2-2

Bảng 3-2-2

A	B	C	D	Z	Thuyết minh
0	0	0	0	0	4 chuyển mạch đều ngắt, đèn tắt
0	0	0	1	1	Có 1 chuyển mạch đóng, đèn sáng
0	0	1	0	1	
0	1	0	0	1	
1	0	0	0	1	
0	0	1	1	0	Có 2 chuyển mạch đồng thời đóng, đèn tắt
0	1	0	1	0	
1	0	0	1	0	
1	0	1	0	0	
1	1	0	0	0	
0	1	1	1	1	Có 3 chuyển mạch đồng thời đúng, đèn sáng
1	0	1	1	1	
1	1	0	1	1	
1	1	1	0	1	
1	1	1	1	0	Cả 4 chuyển mạch đồng thời đóng, đèn tắt

Nếu phải giải quyết một vấn đề logic thực tế, đầu tiên ta hãy làm rõ đâu là đầu vào, đâu là đầu ra, dùng biến đại số biểu thị ; tiếp theo cần xác định quan hệ tương ứng của trạng thái đầu ra - đầu vào. Cuối cùng liệt kê bảng chân lí một cách chính xác.

2) Đặc điểm bảng chân lí

Bảng chân lí biểu thị hàm logic dưới dạng số, nó có các đặc điểm chủ yếu sau đây:

a- Rõ ràng, trực quan. Sau khi xác định giá trị biến đầu vào thì có thể tra bảng chân lí để biết giá trị tương ứng của hàm đầu ra. Vậy nên trong các sổ tay vi mạch số đều có bảng chân lí để giới thiệu chức năng logic của vi mạch

b- Để giải quyết một nhiệm vụ thực tế ở dạng vấn đề logic, thì bảng chân lí là tiện nhất. Vậy nên trong quá trình thiết kế logic của mạch số, việc đầu tiên là phân tích yêu cầu, kê ra bảng chân lí.

Nhược điểm chủ yếu của bảng chân lí là sẽ rối rắm nếu biến số khá nhiều, không thể dùng các công thức và định lí của đại số logic để tính toán.

Để đơn giản, đôi khi chỉ kê tổ hợp các giá trị đầu vào nào tương ứng hàm số lấy giá trị bằng 1. Những tổ hợp thực tế sử dụng không cần, hoặc làm cho hàm sẽ lấy giá trị 0 đều không cần kê ra.

3.2.2. Biểu thức hàm số

Biểu thức hàm số dạng đại số logic dùng các phép toán và, hoặc, đảo biểu thị quan hệ logic giữa các biến trong hàm.

1) Dạng chuẩn tắc tuyến (tổng các tích)

Chỉ cần chú ý đến tổ hợp giá trị các biến nào tương ứng hàm có giá trị 1 trong bảng chân lí. Trong tổ hợp đã chọn, giá trị 1 viết nguyên biến, giá trị 0 viết đảo biến, và kết quả viết được một số hạng dạng tích các biến tương ứng với tổ hợp xét — nếu đem cộng tất cả các số hạng như vậy, thì ta được dạng chuẩn tắc tuyến (Tổng các tích - ORAND) của hàm logic

Ví dụ 3-2-3: Hãy viết biểu thức từ bảng chân lí 3.2.3

Bảng 3-2-3:

A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Giải: hàm $Z = 1$ tương ứng 4 tổ hợp giá trị các biến

$ABC = 011, 101, 110, 111$. Các số hạng dạng tích các biến $\overline{A}BC, A\overline{B}C, ABC\overline{C}, ABC$.

Dạng chuẩn tắc tuyến của hàm số:

$$Z = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC \quad (3-2-1)$$

Kết quả này có chính xác không? Chúng ta có thể nghiệm lại

Biểu thức hàm số chuẩn tắc tuyến có tên gọi nhấn mạnh hình thức chuẩn của các số hạng dạng tích trong biểu thức. Chúng ta gọi số hạng chuẩn này là số hạng nhỏ nhất.

2) Số hạng nhỏ nhất

a) Định nghĩa

Số hạng nhỏ nhất là một khái niệm quan trọng trong đại số logic. Như ở ví dụ 3-2-3, Z là hàm của các biến A, B, C . 3 biến có 8 tổ hợp các giá trị khả dĩ: 000, 001, 010, 011, 100, 101, 110, 111. Tương ứng ta có 8 số hạng dạng tích là $\overline{A}\overline{B}\overline{C}, \overline{A}\overline{B}C, \overline{A}B\overline{C}, \overline{A}BC, A\overline{B}\overline{C}, A\overline{B}C, AB\overline{C}, ABC$. Đặc điểm chung của 8 số hạng này là:

- Đều có 3 thừa số:

Mỗi biến số xuất hiện chỉ 1 lần dưới dạng thừa số hoặc là nguyên biến hoặc là đảo biến

Vậy chúng ta gọi 8 số hạng dạng tích có đặc điểm trên là số hạng nhỏ nhất của các biến A, B, C .

Nói chung, đối với trường hợp n biến, số hạng dạng tích P có n thừa số; trong P mỗi biến đều xuất hiện một lần, và chỉ 1 lần mà thôi, hoặc dưới dạng nguyên biến, hoặc dưới dạng đảo biến; P được gọi là số hạng nhỏ nhất của n biến, n biến có tất cả 2^n số hạng nhỏ nhất. Vì mỗi biến đều có 2 trạng thái (nguyên biến và đảo biến), mà tất cả có n biến

b) Tính chất số hạng nhỏ nhất

Bảng 3-2-4: bảng chân lí toàn bộ số hạng nhỏ nhất của 3 biến số

A	B	C	$\overline{A}\overline{B}\overline{C}$	$\overline{A}\overline{B}C$	$\overline{A}B\overline{C}$	$\overline{A}BC$	$A\overline{B}\overline{C}$	$A\overline{B}C$	$AB\overline{C}$	ABC
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

Từ bảng 3-2-4, ta nhận thấy các tính chất sau của số hạng nhỏ nhất:

Mỗi số hạng nhỏ nhất tương ứng với một tổ hợp giá trị của biến để nó bằng 1, và chỉ có một tổ hợp mà thôi.

Tích của hai số hạng nhỏ nhất bất kì luôn bằng 0

Tổng của tất cả các số hạng nhỏ nhất luôn bằng 1

c) Số hạng tối thiểu là phần tử cơ bản cấu trúc hàm logic

Một hàm logic bất kì đều có thể biểu thị dưới hình thức là tổng của các số hạng nhỏ nhất — dạng chuẩn tắc tuyển. Hơn nữa, hình thức đó là duy nhất, tức là, một hàm logic chỉ có một biểu thức duy nhất biểu thị nó dưới dạng tổng các số hạng tối thiểu. Không những có thể viết ra dạng chuẩn tắc tuyển của hàm logic trực tiếp từ bảng chân lí, mà còn có thể dùng các công thức và định lí của đại số logic, cũng có thể dùng cách khai triển và biến đổi để có dạng chuẩn tắc tuyển

Ví dụ 3-2-4: hãy viết dạng chuẩn tắc tuyển của hàm số $Z = AB + BC + CA$

Giải: $Z = AB + BC + CA$

Ví dụ 3-2-5: hãy viết dạng biểu thức số hạng tối thiểu của hàm

Giải:

d) Kí hiệu của số hạng nhỏ nhất

Để tiện viết, thường gán cho mỗi số hạng nhỏ nhất một kí hiệu. Phương pháp như sau: tổ hợp các giá trị biến số tương ứng với số hạng nhỏ nhất được xét, chuyển hình thức số nhị phân sang số thập phân, con số này là kí hiệu của số hạng nhỏ nhất xét — Ví dụ, trong các số hạng nhỏ nhất của các biến A, B, C thì $\overline{A}\overline{B}\overline{C}$ tương ứng tổ hợp giá trị 000, tức là 0_{10} , kí hiệu của $\overline{A}\overline{B}\overline{C}$ vì vậy là m_0 ; $\overline{A}\overline{B}C$ tương ứng tổ hợp giá trị 010, tức là 2_{10} kí hiệu $\overline{A}\overline{B}C$ là m_2

Tương tự $\overline{ABC} = m_1$; $\overline{ABC} = m_3$; $\overline{ABC} = m_4$; $\overline{ABC} = m_5$; $\overline{ABC} = m_6$; $ABC = m_7$

;

Hơn nữa, thường dùng kí hiệu biểu thị các số hạng nhỏ nhất của dạng chuẩn tắc tuyến; Ví dụ, trong ví dụ 3-2-4:

$Z = \overline{ABC} + \overline{ABC} + \overline{ABC} + ABC$ thường viết thành

$$Z = m_3 + m_5 + m_6 + m_7 = \sum(3,5,6,7)$$

Tương tự, trong ví dụ 3-2-5:

$$Z = \overline{ABC} + ABC = m_0 + m_7 = \sum(0,7)$$

3) Dạng chuẩn tắc tuyến của đảo hàm

Nếu lấy tổng các số hạng nhỏ nhất tương ứng với các tổ hợp giá trị các biến mà hàm lấy giá trị 0 trong bảng chân lí, thì ta có dạng chuẩn tắc tuyến của đảo hàm. Ví dụ, bảng chân lí 3-2-3 ta có: $\overline{Z} = \overline{ABC} + \overline{ABC} + \overline{ABC} + \overline{ABC}$

$\overline{Z}$ là đảo hàm của Z. Nếu ta lại lấy đảo lần nữa của $\overline{Z}$, và triển khai theo định lí triển khai, thì ta sẽ được dạng chuẩn tắc tuyến của Z mà ta đã có ở ví dụ 3-2-3:

4) Dạng chuẩn tắc hội (tích các tổng)

Dạng chuẩn tắc hội có thể nhận được bằng phương pháp sau:

Từ bảng chân lí tìm dạng chuẩn tắc tuyến của đảo hàm, sau đó dùng định lí De Morgan để tìm đảo của đảo hàm

Ví dụ, từ phần trên ta đã tìm được $\overline{Z} = \overline{ABC} + \overline{ABC} + \overline{ABC} + \overline{ABC}$

Các thừa số của hàm số dạng chuẩn tắc hội có tính chất sau:

Đều bao gồm tất cả các biến của hàm

Mỗi biến đều xuất hiện một lần và chỉ một lần trong dạng tổng của thừa số, hoặc là nguyên biến, hoặc là đảo biến.

Các thừa số có tính chất nêu trên được gọi là thừa số lớn nhất. Tích các thừa số lớn nhất là dạng chuẩn tắc hội của hàm số.

(3-2-2) là biểu thức của hàm Z dạng chuẩn tắc hội.

Nói chung, đối với trường hợp hàm n biến, thừa số lớn nhất là một tổng của n số hạng, mỗi số hạng là một biến, xuất hiện một lần dưới dạng nguyên biến hoặc đảo biến và chỉ xuất hiện một lần mà thôi, n biến có tương ứng 2^n thừa số lớn nhất. Bảng 3-2-5 là bảng chân lí của toàn bộ các thừa số lớn nhất tương ứng hàm 3 biến A, B, C

Bảng 3-2-5:

A	B	C	$A+B+C$	$A+B+\overline{C}$	$A+\overline{B}+C$	$A+\overline{B}+\overline{C}$	$\overline{A}+B+C$	$\overline{A}+B+\overline{C}$	$\overline{A}+\overline{B}+C$	$\overline{A}+\overline{B}+\overline{C}$
0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0

Nhận xét bảng 3-2-5, ta thấy thừa số lớn nhất có các tính chất sau;

Mỗi thừa số lớn nhất tương ứng với một tổ hợp giá trị của biến để nó bằng 0, và chỉ có một tổ hợp mà thôi.

Tổng của hai thừa số lớn nhất bất kì luôn luôn bằng 1

Tích của tất cả các thừa số lớn nhất luôn bằng 0

Cách kí hiệu các thừa số lớn nhất như sau: tổ hợp các giá trị biến số tương ứng với thừa số lớn nhất được xét chuyển hình thức số nhị phân sang số thập phân, con số này là kí hiệu của thừa số lớn nhất xét — Ví dụ, trong các thừa số lớn nhất của các biến A, B, C thì:

$A+B+C$	tương ứng tổ hợp	000,	chuyển thành 0_{10} ,	kí hiệu	M_0
$A+B+\overline{C}$	tương ứng tổ hợp	001,	chuyển thành 1_{10} ,	kí hiệu	M_1
$A+\overline{B}+C$	tương ứng tổ hợp	010,	chuyển thành 2,	kí hiệu	M_2
$A+\overline{B}+\overline{C}$	tương ứng tổ hợp	011,	chuyển thành 3,	kí hiệu	M_3
$\overline{A}+B+C$	tương ứng tổ hợp	100,	chuyển thành 4,	kí hiệu	M_4
$\overline{A}+B+\overline{C}$	tương ứng tổ hợp	101,	chuyển thành 5,	kí hiệu	M_5
$\overline{A}+\overline{B}+C$	tương ứng tổ hợp	110,	chuyển thành 6,	kí hiệu	M_6
$\overline{A}+\overline{B}+\overline{C}$	tương ứng tổ hợp	111,	chuyển thành 7,	kí hiệu	M_7

Cách viết ký hiệu rất thuận tiện. Chú ý rằng m_i và M_i là đảo của nhau:

$$m_i = \overline{M_i}$$

$$\text{Ví dụ: } m_0 = \overline{ABC} \quad M_0 = A + B + C$$

$$m_0 = \overline{M_0} = \overline{A+B+C} = \overline{ABC}$$

$$m_5 = \overline{ABC} \quad M_5 = \overline{A+B+\overline{C}}$$

$$m_5 = \overline{M}_5 = \overline{A+B+C} = \overline{A} \overline{B} \overline{C}$$

Thừa số lớn nhất cũng là phân tử cơ bản cấu trúc hàm logic. Biểu thức hàm số (3-2-2) có thể viết dưới dạng:

$$Z = M_0 M_1 M_2 M_4 = \square (0, 1, 2, 4)$$

Đặc điểm các biểu thức hàm số

Một hàm logic được biểu thị bằng biểu thức các phép toán Và, hoặc, đảo ... liên kết các biến số của nó với nhau. ưu điểm của phương pháp biểu thức hàm số là:

- Dùng các ký hiệu logic biểu thị quan hệ logic giữa các biến làm cho cách viết gọn và tiện, tính khái quát và trừu tượng rất cao.

- Rất tiện sử dụng các công thức và định lý của đại số logic để biến đổi, làm toán.

- Tiện cho việc dùng sơ đồ logic để thực hiện hàm số. Chỉ cần dùng các ký hiệu logic của mạch điện cổng tương ứng thay thế phép toán xét trong biểu thức hàm số, ta được một sơ đồ logic. Vấn đề này còn được giới thiệu cụ thể sau.

Nhược điểm chủ yếu của phương pháp biểu thức hàm số là khó xác định giá trị hàm ứng với giá trị biến một cách trực tiếp đối với các hàm số phức tạp (không trực quan như bảng chân lý).

3.2.3. Bảng Karnaugh:

Bảng Karnaugh là phương pháp hình vẽ biểu thị hàm logic, trong đó các giá trị hàm đầu ra tương ứng tổ hợp các biến đầu vào đều được biểu thị đầy đủ. Trên cơ sở bảng Karnaugh của các biến, điền các số hạng nhỏ nhất của hàm số vào các ô tương ứng thì ta có bảng Karnaugh của hàm.

1/ Bảng Karnaugh của biến logic:

a/ Hình 3-2-1 trình bày bảng Karnaugh 3 biến và 4 biến.

b/ Qui tắc vẽ bảng Karnaugh của biến

như sau:

- Bảng Karnaugh có dạng hình chữ nhật.

N biến có 2^n ô, mỗi ô tương ứng với một số hạng nhỏ nhất. Ví dụ hình 3-2-1, $n=3$ tương ứng bảng $2^3 = 8$ ô, $n = 4$ tương ứng bảng $2^4 = 16$ ô.

- Giá trị các biến được sắp xếp thứ tự theo mã vòng. (Nếu không sắp xếp thứ tự theo mã vòng thì không còn là bảng Karnaugh nữa).

Ví dụ: Sự sắp xếp của AB và CD đều là 00, 01, 11, 10 (hình 3-2-1).

A \ BC	00	01	11	10
	m ₀	m ₁	m ₃	m ₂
0				
1				

A \ BC	00	01	11	10
0				
1				

AB \ CD	00	01	11	10
00				
01				
11				
10				

AB \ CD	00	01	11	10
	0	1	3	2
00				
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

Mã vòng có thể suy ra từ mã số nhị phân như sau. Giả sử cho mã số nhị phân là B_3, B_2, B_1, B_0 , mã vòng tương ứng là $G_3, G_2,$

G_1, G_0 , thì có thể tính $G_i = B_{i+1} \oplus B_i$. Cụ thể, $G_0 = B_1 \oplus B_0$; $G_1 = B_2 \oplus B_1$; $G_2 = B_3 \oplus B_2$; $G_3 = B_4 \oplus B_3 = 0 \oplus B_3 = B_3$ ($B_4 = 0$). Hình 3-2-2 là bảng Karnaugh 5 biến và 6 biến.

Bảng 3-2-6 là mã vòng tương ứng với mã nhị phân (3 bit)

Hình 3-2-1

Bảng Karnaugh được xem như sơ đồ khối của các số hạng nhỏ nhất

AB \ CDE	000	001	011	010	110	111	101	100
	00	01	11	10	00	01	11	10
00								
01								
11								
10								

Hình 3-2-2 (a)

ABC \ DEF	000	001	011	010	110	111	101	100
	000	001	011	010	110	111	101	100
000								
001								
011								
010								
110								
111								
101								
100								

Hình 3-2-2 (b)

Hình 3-2-2 (a)

Hình 3-2-2 (b)

Hình 3-2-6

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

c/Đặc điểm bảng Karnaugh của biến:

-ưu điểm lớn nhất của bảng là làm nổi bật tính kế nhau của các số hạng nhỏ nhất. Các ô kế nhau bất kỳ trên bảng đều có các số hạng nhỏ nhất đều có tính kế nhau về logic. Sự sắp xếp giá trị các biến theo mã vòng bảo đảm đặc điểm quan trọng này.

Tính kế nhau bao gồm 3 tình huống sau: các ô kế nhau, các ô đầu cuối của hàng và cột, các ô đối xứng đều phải có một giá trị đối nhau của biến và chỉ một mà thôi. Đặc điểm này của bảng cho phép dễ dàng nhớ và phân biệt, kiểm tra, tính toán bằng bảng, nhất là khi có đến 5, 6 biến (xem hình 3-2-2). Như trên đã nói, nếu trong 2 số hạng nhỏ nhất có và chỉ có 1 biến lấy giá trị khác nhau, còn tất cả các biến khác đều lấy giá trị như nhau, thì hai số hạng nhỏ nhất đó có tính kế nhau về logic. Ví dụ, trong hình 3-2-1, m_0 có tính kế nhau về logic với m_1 , m_2 và m_4 .

Khi cộng các số hạng nhỏ nhất có tính kế nhau, thì biến đối nhau trong đó sẽ bị khử. Ví dụ $m_0 + m_1 = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C = \overline{A}\overline{B}(\overline{C} + C) = \overline{A}\overline{B}$; $\overline{A}\overline{B}$ là thừa số chung của m_0 và m_1 ; $m_0 + m_2 = \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} = \overline{A}\overline{C}$ khử mất $\overline{B}$ và B; $m_0 + m_4 = \overline{A}\overline{B}\overline{C} + \overline{A}BC = \overline{A}\overline{C}$ khử mất $\overline{B}$ và B.

-Nhược điểm chủ yếu của bảng Karnaugh: nếu số biến tăng thì độ phức tạp của bảng tăng nhanh. Ví dụ nếu số biến từ 7 trở đi thì hình vẽ quá phức tạp, hơn nữa rất khó xét đoán tính kế nhau về logic của các số hạng nhỏ nhất. Vì vậy, bảng Karnaugh chỉ thích hợp để biểu thị hàm logic có số biến từ 6 trở lại.

2/ Bảng Karnaugh của hàm logic

a/ Cách vẽ: có 3 trường hợp

Trường hợp 1: đã cho bảng chân lí của hàm.

Trên bảng Karnaugh của biến, điền giá trị 1 vào ô mà hàm lấy giá trị tương ứng tổ hợp giá trị các biến của ô xét, điền giá trị 0 vào ô mà hàm lấy giá trị 0 tương ứng tổ hợp giá trị các biến của ô xét.

Ví dụ 3-2-5:

Cho bảng chân lí 3-2-7 (hình dưới)

Hãy vẽ bảng Karnaugh của hàm Z

Giải:

- Đầu tiên vẽ bảng Karnaugh cho 4 biến A, B, C, D.

- Tiếp theo điền các giá trị của hàm Z vào các ô tương ứng phù hợp với bảng chân lí.

A	B	C	D	Z
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Bảng 3-2-7

- Kết quả : Hình 3-2-3

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

AB \ CD				
	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

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

Hình 3-2-3

Hình 3-2-4 a/ Các ô có ghi kí hiệu số hạng nhỏ nhất
b/ Các ô ghi giá trị 1 và 0 của hàm

Trường hợp 2; Đã cho biểu thức của hàm dưới dạng chuẩn tắc tuyến trên bảng Karnaugh của biến, điền giá trị 1 vào các ô tương ứng với từng số hạng nhỏ nhất có trong biểu thức, các ô khác đều điền vào giá trị 0.

Ví dụ 3-2-6

Hãy vẽ bảng Karnaugh của hàm logic

$$Z = \sum (0,3,5,6,9,10,12,15)$$

Giải:

-Vẽ bảng Karnaugh của hàm logic

-Điền giá trị

-Kết quả: hình 3-2-4

Trường hợp 3: cho biểu thức không chuẩn tắc của hàm.

-Biến đổi hàm đã cho thành dạng tổng các tích

-Trên bảng Karnaugh của biến, điền giá trị 1 vào tất cả các ô tương ứng số hạng nhỏ nhất bao hàm trong số hạng dạng tích nói trên, sau đó điền giá trị 0 vào các ô còn lại.

Ví dụ: 2-2-7: hãy vẽ bảng Karnaugh của hàm $Z = \overline{(A \oplus B)(C + D)}$

Giải:

-Biến hàm thành dạng tổng các tích:

$$Z = \overline{(A \oplus B)(C + D)} = \overline{A \oplus B} + \overline{C + D} = \overline{AB} + AB + \overline{CD}$$

-Xác định mỗi số hạng bao gồm những số hạng nhỏ nhất

nào:

$$\overline{AB} = m_0 + m_1 + m_2 + m_3$$

$$AB = m_{12} + m_{13} + m_{14} + m_{15}$$

$$\overline{CD} = m_0 + m_4 + m_8 + m_{12}$$

-Kết quả vẽ được như hình 3-2-5

(đối chiếu hình 3-2-4a)

b/ Từ bảng Karnaugh kê ra bảng chân lí và viết biểu thức:

Bảng chân lí hàm dạng chuẩn tắc tuyến và bảng Karnaugh đều là duy nhất biểu thị cho một hàm, chúng tất có quan hệ chuyển đổi nhau. Thực tế ở các phần trên đã chuyển đổi rồi.

Ưu điểm nổi bật nhất của bảng Karnaugh là tính kế nhau về logic của các số hạng nhỏ nhất của hàm biểu thị rõ rệt thành sự liên kết hình học của các ô trong bảng, do đó dễ dàng tối thiểu hoá hàm đã cho. Vấn đề này sẽ giảng chi tiết ở phần sau.

3.2.4. Sơ đồ logic

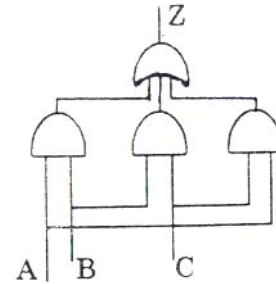
AB \ CD	CD			
	00	01	11	10
00	1	1	1	1
01	1	0	0	0
11	1	1	1	1
10	1	0	0	0

Hình 3-2-5

Trong mạch số, sau khi dùng các kí hiệu logic biểu thị một cấu trúc logic trên một bản vẽ, ta được sơ đồ logic. Sơ đồ logic cũng là một phương pháp biểu thị hàm logic, hơn nữa lại có ưu điểm nổi bật là rất tiếp cận thực tế. Các kí hiệu logic thông thường đều có cấu kiện điện tử cụ thể tương ứng, vậy nên thường gọi sơ đồ logic là sơ đồ mạch logic.

1) Cách vẽ sơ đồ logic của hàm logic

Như trên đã nói, ta dùng kí hiệu logic của mạch điện tử thay thế phép tính logic có trong biểu thức hàm logic thì được sơ đồ logic của hàm.



Hình 3-2-6

Ví dụ 3-2-8: cho hàm $Z = AB + BC + CA$

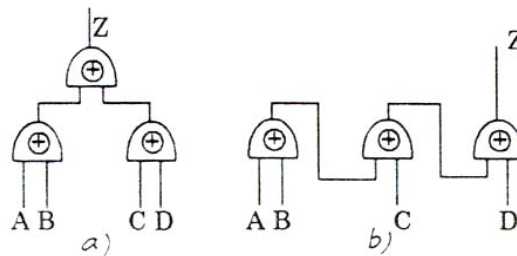
Hãy vẽ sơ đồ logic của Z.

Giải: Quan hệ nhân logic của các biến A và B, B và C, C và A được thực hiện bằng các cổng AND. Quan hệ, cộng logic của các số hạng AB, BC và CA được thực hiện bằng cổng OR. Kết quả: hình 3-2-6

Ví dụ 3-2-9: cho hàm $Z = A \oplus B \oplus C \oplus D$

Hãy vẽ sơ đồ logic của Z.

Giải: quan hệ cộng với phép loại trừ giữa các biến A, B, C, D được thực hiện qua cổng XOR — kết quả xem hình 3.2.7



Hình 3-2-7

- a) Cấu trúc hình tháp (trễ truyền đạt nhỏ);
b) Cấu trúc nối mắt xích (trễ truyền đạt lớn).

2) Cách xác định biểu thức từ sơ đồ logic

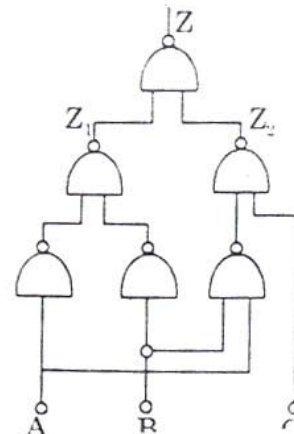
Trên sơ đồ logic, từ đầu vào đến đầu ra, viết biểu thức hàm đầu ra của từng cấp, cuối cùng được biểu thức hàm logic toàn sơ đồ.

Ví dụ 3-2-10: cho sơ đồ hình 3-2-8.

Hãy viết biểu thức hàm logic của sơ đồ

Giải: $Z_1 = \overline{AB}$

$Z_2 = \overline{ABC}$



$$Z = Z_1 Z_2 = \overline{\overline{ABAB}}C$$

3/ Đặc điểm của sơ đồ logic

Các kí hiệu logic trong sơ đồ logic có quan hệ phù hợp với cấu kiện điện tử trong thực tế, vậy sơ đồ logic tương đối tiếp cận thực tế công trình. Trong công tác, khi tìm hiểu chức năng logic của một hệ thống số nào đó hay thiết bị được điều khiển số nào đó, thường ta cần dùng sơ đồ logic, vì rằng sơ đồ logic có thể

Hình 3-2-8.

biểu thị rõ ràng chức năng logic từng tầng của các mạch

điện thực tế phức tạp. Trong việc chế tạo các thiết bị số, việc đầu tiên là thiết kế logic để vẽ ra sơ đồ logic, rồi chuyển từ sơ đồ logic thành mạch điện thực tế.

3.3 Phương pháp tối thiểu hoá hàm logic

Trực tiếp thiết kế sơ đồ mạch logic hàm số có được từ bảng chân lí thường là rất phức tạp. Nếu sau khi đã được thực hiện tối thiểu hoá hàm logic, nói chung việc thực hiện thuận tiện hơn, không những chỉ dùng số cấu kiện ít hơn, mà nâng cao được độ tin cậy. Dưới đây sẽ nói đến khái niệm tối thiểu hoá, tiếp theo sẽ giới thiệu 2 phương pháp thường dùng để tối thiểu hoá.

3.3.1 Khái niệm về tối thiểu hoá

1) Các loại biểu thức logic và sự thực hiện bằng mạch điện

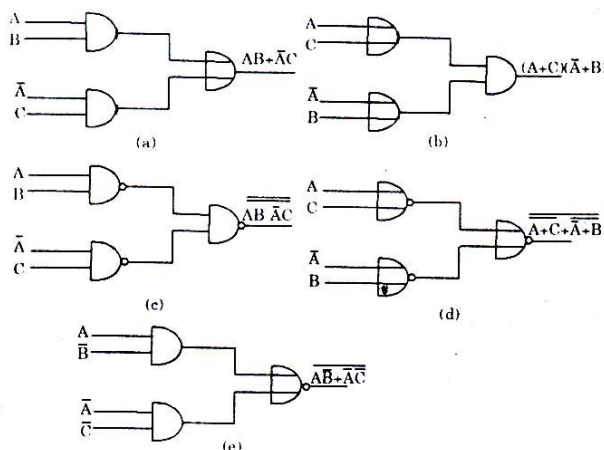
Ta đã biết, một hàm logic có thể có rất nhiều biểu thức khác nhau. Chúng ta có thể phân loại theo các hàm logic thành 5 loại căn cứ vào đặc điểm và quan hệ giữa các số hạng dạng tích trong hàm: OR-AND, AND-OR, NAND-NAND, NOR-NOR, NOR-AND. Ví dụ:

$Z = AB + \overline{A}C$	dạng biểu thức	OR-AND
$Z = (A + C)(\overline{A} + B)$		AND-OR
$Z = \overline{\overline{ABAB}}$		NAND-NAND
$Z = \overline{A + C} \overline{\overline{A} + B}$		NOR-NOR
$Z = \overline{AB + \overline{A}C}$		NOR-AND

Khi dùng các cổng logic thực hiện các hàm logic này, tiện nhất là: hai loại đầu dùng cổng AND và cổng OR, loại thứ ba dùng NAND, loại thứ tư dùng NOR, loại thứ năm

dùng NORNAND

giới
nói



Hình 3-3-1 dưới đây
giới thiệu mạch điện các hàm
trên:

Hình 3-3-1

Trên thực tế, khi chúng ta viết một hàm logic dưới một dạng nào đó, thì dạng có được không phải là duy nhất. Ví dụ, biểu thức OR-AND trong các ví dụ trên có thể viết thành:

$$Z = AB + \overline{A}C \quad (3-31a)$$

$$= AB + \overline{A}C + BC \quad (3-3-1b)$$

$$= ABC + AB\overline{C} + \overline{A}BC + \overline{A}\overline{B}C \quad (3-3-1c)$$

Dùng các cổng AND Và OR thực hiện (3.3.1a) ta có mạch đơn giản nhất. Nói chung, nếu biểu thức càng đơn giản thì mạch điện cũng càng đơn giản. Nhưng đối với các biểu thức dạng khác nhau thì tiêu chuẩn về sự đơn giản có khác nhau. Ta sẽ làm rõ điều này qua ví dụ về biểu thức OR-AND

2) Biểu thức OR-AND tối thiểu

a) Thế nào là tối thiểu

- Đầu tiên số các số hạng dạng tích phải là ít nhất
- Nếu điều kiện trên đã đảm bảo thì số biến của mỗi số hạng cũng phải là ít nhất.

$$\text{Ví dụ: } Z = A\overline{C} + B\overline{C} + \overline{A}B + \overline{A}C \quad (3-3-2a)$$

$$= A\overline{C} + B\overline{C} + \overline{A}C \quad (3-3-2b)$$

$\overline{A}B$ được khử bỏ theo quy tắc của công thức 17. Rõ ràng (3-3-2b) đơn giản hơn (3-3-2a) vì nó ít hơn một số hạng.

b) ý nghĩa việc tối thiểu hoá biểu thức OR-AND

Chúng ta tập trung nghiên cứu phương pháp tối thiểu hoá biểu thức OR-AND, vì chỉ cần có biểu thức OR-AND tối thiểu, ta sẽ dễ dàng có được các biểu thức dạng khác cũng tối thiểu. Có hai lý do: một là, một biểu thức logic bất kì đều dễ dàng triển khai thành biểu thức dạng OR-AND; Hai là, từ biểu thức dạng OR-AND tối thiểu, cũng dễ dàng có được các biểu thức tối thiểu dạng NAND-NAND, NORAND.V.V□

3.3.2 Phương pháp tối thiểu hoá bằng công thức

Dựa vào các công thức và định lý trong đại số logic để thực hiện tối thiểu hoá. Vì trong thực tế, biểu thức logic rất đa dạng, lại không có một cách nào hoàn chỉnh như một quy trình, nên việc đạt đến một biểu thức logic tối thiểu một cách nhanh nhất sẽ hoàn toàn phụ thuộc kinh nghiệm, hiểu biết và thành thạo của chúng ta.

Các ví dụ về tối thiểu hóa:

Ví dụ 3-3-1: Hãy tối thiểu hoá hàm $Z = A\bar{B}C + A\bar{B}\bar{C}$

Giải: $Z = A\bar{B}C + A\bar{B}\bar{C} = A\bar{B}(C + \bar{C}) = A\bar{B}$ (công thức 14)

Ví dụ 3-3-2: Hãy tối thiểu hóa hàm $Z = A(BC + \bar{B}\bar{C}) + A(\overline{BC + \bar{B}\bar{C}})$

Giải: $Z = A[(BC + \bar{B}\bar{C}) + (\overline{BC + \bar{B}\bar{C}})] = A$

Ví dụ: 3-3-3: Hãy tối thiểu hoá hàm $Z = A\bar{B} + A\bar{B}CD(E + F)$

Giải: $Z = A\bar{B} + A\bar{B}CD(E + F) = A\bar{B}$ (công thức 15)

Ví dụ: 3-3-4: Hãy tối thiểu hoá hàm $Z = AB + \bar{A}C + \bar{B}C$

Giải: $Z = AB + \bar{A}C + \bar{B}C = AB + (\bar{A} + \bar{B})C$
 $= AB + \overline{AB}C = AB + C$ (công thức 16)

Ví dụ 3-3-5: Hãy tối thiểu hoá hàm $Z = A\bar{B} + AC + ADE + \bar{C}D$

Giải: $Z = A\bar{B} + AC + \bar{C}D + ADE = A\bar{B} + AC + \bar{C}D$ (Hệ quả công thức 17)

Việc khử đi số hạng hay biến trong số hạng là do chúng được bao hàm trong các số hạng khác. Điều này khác hẳn đại số thường!

Ví dụ 3-3-6: Hãy tối thiểu hoá hàm $Z = AB + BC + \bar{B}C + \bar{A}B$

Giải $Z = A\bar{B} + B\bar{C} + \bar{B}C + \bar{A}B$
 $Z = (A\bar{B} + B\bar{C} + A\bar{C} + A\bar{C}) + (\bar{B}C + \bar{A}B)$
 $Z = (A\bar{B} + \bar{B}C + A\bar{C}) + (B\bar{C} + \bar{A}B + A\bar{C})$
 $Z = (\bar{B}A + \bar{B}C + A\bar{C}) + (B\bar{C} + \bar{A}B + A\bar{C})$
 $Z = (\bar{B}C + A\bar{C}) + (\bar{A}B + A\bar{C})$
 $Z = \bar{B}C + \bar{A}B + A\bar{C}$

Cách giải trên đây ứng với công thức 17, thêm vào và bớt đi rất linh hoạt. Chẳng hạn nếu thêm vào hay bớt đi AC , thì hàm đã cho ở ví dụ 3-3-6 có dạng tối thiểu hóa mới:

$Z = A\bar{B} + B\bar{C} + \bar{B}C + \bar{A}B = A\bar{B} + B\bar{C} + \bar{A}C$

Theo công thức 11', ta có thể chỉ viết 1 số hạng $A\bar{C}$ nhưng ngầm hiểu có thể gộp nó vào nhiều nhóm. Trong ví dụ này, thêm vào chỉ một, nhờ gộp vào hai nhóm nên bớt đi hai, kết quả là khử bớt 1 số hạng. Trong trường hợp tổng quát, ta thường ứng dụng nhiều công thức và định lý để có thể tối thiểu hoá một hàm phức tạp.

Ví dụ 3-3-7: Hãy tối thiểu hóa hàm

$Z = AD + \bar{A}\bar{D} + AB + \bar{A}C + BD + \bar{A}CEF + \bar{B}EF + DEFG$

Giải:

- Dùng công thức 14: $AD + A\bar{D} = A$

$$Z = A + AB + \bar{A}C + BC + \bar{A}CEF + \bar{B}EF + DEFG$$

- Dùng công thức 15: Khử bỏ AB, $\bar{A}CEF$

$$Z = A + \bar{A}C + BD + \bar{B}EF + DEFG$$

- Dùng công thức 16: Khử bỏ $\bar{A}$ trong số hạng $\bar{A}C$

$$Z = A + C + BD + \bar{B}EF + DEFG$$

- Dùng công thức 17: khử bỏ DEFG, vậy:

$$Z = A + C + BD + \bar{B}EF$$

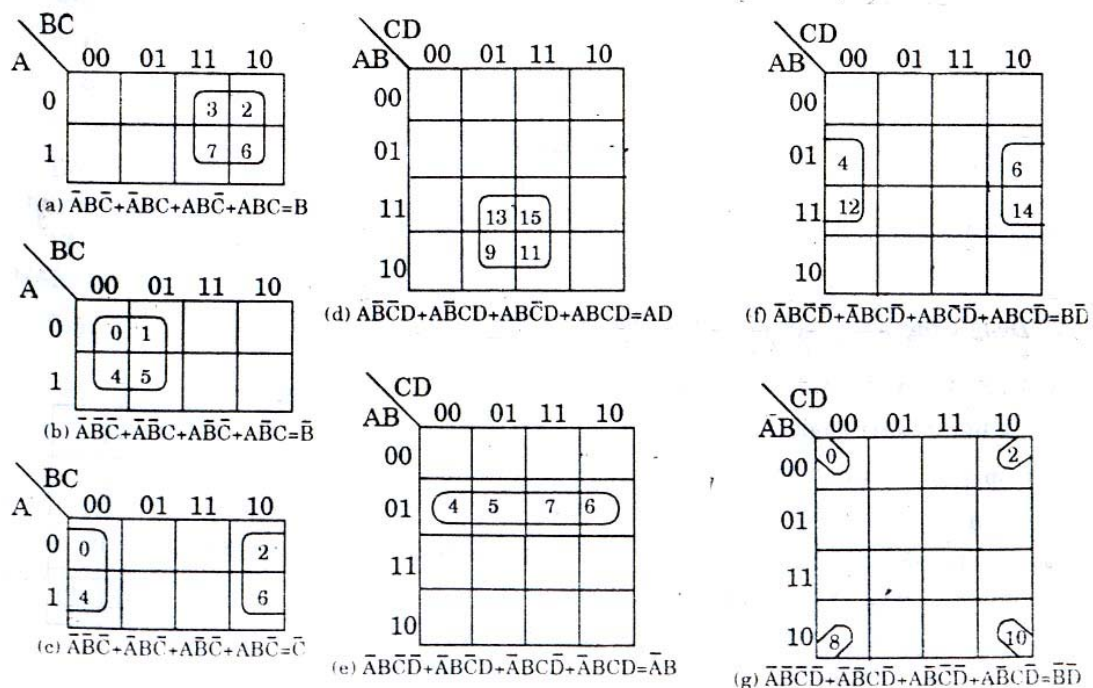
3.3.3 Phương pháp tối thiểu hóa bằng hình vẽ

Phương pháp này dùng bảng Karnaugh.

1) Quy luật gộp (dán) các số hạng nhỏ nhất trên bảng Karnaugh

Trên bảng Karnaugh của biến, tất cả các số hạng nhỏ nhất kề nhau đều có thể gộp với nhau, khi gộp lại thì có thể khử bỏ biến liên quan. Cụ thể, cứ 2 số hạng nhỏ nhất gộp lại thì khử bỏ được một biến, cứ 4 số hạng nhỏ nhất gộp lại (thành một số hạng) thì khử bỏ được 2 biến, cứ 8 số hạng nhỏ nhất gộp lại thì khử được 3 biến. Nói tổng quát, 2^n số hạng nhỏ nhất gộp lại (thành một số hạng) thì khử được n biến. Vì rằng 2^n số hạng nhỏ nhất cộng với nhau (gộp lại), trừ các thừa số chung rồi thì còn lại 2^n số hạng dạng tích, vừa đúng bằng toàn bộ số hạng nhỏ nhất của n biến cần khử bỏ. Ta đã biết tính chất của số hạng nhỏ nhất, tổng của chúng bằng 1.

Các hình 3-3-2, hình 3-3-3, hình 3-3-4 vẽ tương ứng các trường hợp có 2,4,8 số hạng nhỏ nhất được gộp.



Hình 3-3-2

		BC			
A	AB	00	01	11	10
	0			3	2
	1			7	6

(a) $\bar{A}\bar{B}\bar{C} + \bar{A}BC + A\bar{B}\bar{C} + ABC = B$

		BC			
A	AB	00	01	11	10
	0	0	1		
	1	4	5		

(b) $\bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C = \bar{B}$

		BC			
A	AB	00	01	11	10
	0	0			2
	1	4			6

(c) $\bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + AB\bar{C} = \bar{C}$

		CD			
AB	AB	00	01	11	10
	00				
	01				
	11			13	15
	10			9	11

(d) $\bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}CD + A\bar{B}\bar{C}D + A\bar{B}CD = \bar{A}\bar{B}$

		CD			
AB	AB	00	01	11	10
	00				
	01	4			6
	11	12			14
	10				

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

		CD			
AB	AB	00	01	11	10
	00				
	01	4	5	7	6
	11				
	10				

(e) $\bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D} = \bar{A}\bar{B}$

		CD			
AB	AB	00	01	11	10
	00	0			2
	01				
	11				
	10	8			10

(g) $\bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D} = \bar{A}\bar{B}$

Hình 3-3-3,

		CD			
AB	AB	00	01	11	10
	00				
	01	4	5	7	6
	11	12	13	15	14
	10				

(a) B

		CD			
AB	AB	00	01	11	10
	00	0	1	3	2
	01				
	11				
	10	8	9	11	10

(b) $\bar{B}$

		CD			
AB	AB	00	01	11	10
	00	0			2
	01	4			6
	11	12			14
	10	8			10

(c) $\bar{D}$

		CD			
AB	AB	00	01	11	10
	00	0	1		
	01	4	5		
	11	12	13		
	10	8	9		

(d) $\bar{C}$

Hình 3-3-4

2) Dùng bảng Karnugh tối thiểu hoá hàm logic

Nói chung, quy trình có 3 bước

- Vẽ bảng số hạng nhỏ nhất
- Gộp các số hạng nhỏ nhất
- Chọn số hạng viết biểu thức OR-AND tối thiểu

Ví dụ 3-3-8 : Dùng hình vẽ tối thiểu hoá hàm

$$Z = \Sigma (1,3,4,5,10,11,12,13)$$

Giải : - vẽ bảng Karnaugh của hàm Z : Vẽ bảng

Karnaugh của 4 biến A,B,C,D . Trên hình đánh dấu tất

cả các số

Hạng nhỏ nhất của hàm . Xem hình 3-3-5

- Gộp các số hạng nhỏ nhất .

Theo phương pháp đã giới thiệu trước đây , khoanh vòng các số hạng nhỏ

Nhất có thể gộp . Từ hình 3-3-5 , ta có :

$$\Sigma(4,5,12,13) = B\bar{C}$$

$$\Sigma(1,3) = \bar{A}\bar{B}D$$

$$\Sigma(10,11) = A\bar{B}C$$

$$\Sigma(1,5) = \bar{A}\bar{C}D$$

$$\Sigma(3,11) = \bar{B}CD$$

- Chọn số hạng viết biểu thức OR-AND tối thiểu .

Nguyên tắc chọn số hạng :

- Phải bao gồm các số hạng nhỏ nhất của hàm
- Số các số hạng được chọn phải là ít nhất
- Số thừa số của mỗi số hạng cũng phải là ít nhất .

Trong ví dụ này , có thể chọn $B\bar{C}$, $\bar{A}\bar{B}D$, $A\bar{B}C$

Vậy kết quả tối thiểu hoá , ta có:

$$Z = B\bar{C} + \bar{A}\bar{B}D + A\bar{B}C$$

CD \ AB	00	01	11	10
00		1	3	
01	4	5		
11	12	13		
10			11	10

Hình 3-3-5

3) Mấy vấn đề cần lưu ý

- Vòng gộp phải càng to càng tốt . Tương ứng số các số hạng nhỏ nhất được gộp lại càng nhiều ; do đó, sau khi gộp , số hạng càng ít thừa số .

- Mỗi vòng gộp bao gồm ít nhất một số hạng nhỏ nhất không có trong vòng khác . Vòng bao gồm các số hạng đều đã có trng vòng khác , thì vòng đó là thừa . Mặt khác, mỗi số hạng nhỏ nhất có thể được sử dụng nhiều lần (có mặt trong nhiều vòng khác nhau)

- Phải khoanh vòng sao cho toàn bộ số hạng nhỏ nhất của hàm số đều có các vòng , không sót . Các thừa số tương ứng của số hạng vòng gộp làm thành số hạng của hàm đã tối thiểu hoá .

- Trong một số trường hợp , có thể có nhiều cách khoanh vòng, nghĩa là có thể có nhiều vòng tối thiểu . Những hàm tối thiểu này cần được so sánh , kiểm tra để chọn ra đâu là hàm tối thiểu thực sự (tối thiểu của tối thiểu !)

- Khi gộp các số hạng nhỏ nhất , nghĩa là khi khoanh vòng , có 2 điều sau đây dễ quên : một là , phải nhớ rằng 4 ô ở 4 góc bảng Karnagh cũng có thể gộp với nhau (xem hình 3-3-3g); hai là, vẽ vòng lớn trước vòng bé sau, kiểm tra xem : mỗi vòng có ít nhất một số hạng nhỏ nhất không có trong vòng khác. Không lưu ý đến những vấn đề này , biểu thức hàm số đạt được không chắc là tối thiểu .

Ví dụ 3-3-9 : Dùng hình vẽ tối thiểu hoá hàm

$$Z = \Sigma(1,4,5,6,8,12,15)$$

Giải :

- Vẽ bảng Karnaugh của hàm Z

xem hình 3-3-6

- Gộp các số hạng nhỏ nhất.

Tuy vòng $m_4 + m_5 + m_{12} + m_{13}$, nên vòng này thừa.

-Biểu thức OR-AND tối thiểu:

$$Z = \overline{A} \overline{C} D + \overline{A} B \overline{D} + A \overline{C} \overline{D} + ABD$$

Ví dụ: 3-3-10: Dùng hình vẽ tối thiểu hóa hàm

$$Z = \overline{A \oplus C} - \overline{B(ACD + \overline{A} \overline{C} \overline{D})}$$

Giải:

-Biến đổi hàm Z thành dạng biểu thức OR — AND

$$Z = \overline{A \oplus C} \cdot \overline{B(ACD + \overline{A} \overline{C} \overline{D})}$$

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

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

- Vẽ bảng Karnaugh như hình 3-3-7

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

Hình 3-3-6.

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

Hình 3-3-7

- Gộp các số hạng nhỏ nhất.

Lưu ý các ô 4 góc bảng có thể gộp là $m_0 + m_2 + m_8, m_{10}$ (nếu gộp thành 2 vòng nhỏ m_0, m_2 và m_8, m_{10} thì vẫn chưa được tối thiểu).

-Viết hàm tối thiểu: $Z = \overline{A} \overline{C} + AC + \overline{B} \overline{D}$

4/ Dùng bảng Karnaugh tìm hàm OR-AND tối thiểu của hàm đảo

Nếu trên bảng Karnaugh của hàm số, ta gộp tất cả các số hạng nhỏ nhất ứng với giá trị 0 của hàm xét, thì ta được biểu thức OR-AND tối thiểu của hàm đảo $\overline{Z}$.

Ví dụ: 3-3-11: cho $Z = AB + BC + CA$. Dùng hình vẽ tìm ra biểu thức OR-AND tối thiểu của $\overline{Z}$.

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

Hình 3-3-8

Giải:

- Vẽ bảng Karnaugh của hàm Z

Hình 3-3-8

- Gộp các số hạng nhỏ nhất ứng với $Z = 0$

$$m_0 + m_1 = \overline{A} \overline{B}, m_0 + m_2 = \overline{A} \overline{C}$$

$$m_0 + m_4 = \overline{B} \overline{C}$$

- Viết biểu thức hàm OR-AND tối thiểu của Z:

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

3.3.4. Chuyển đổi biểu thức OR-AND tối thiểu thành biểu thức tối thiểu các dạng NAND-NAND, NORAND, NOR-NOR

Sau khi nghiên cứu xong phương pháp tối thiểu hóa biểu thức dạng OR-AND, chúng ta nghĩ ngay đến vấn đề: Làm thế nào từ biểu thức OR-AND tối thiểu để có được biểu thức tối thiểu các dạng khác.

1/ Tính hoàn hảo của phép tính NAND và NOR

Mọi phép toán trong đại số logic đều có thể quy về 3 phép toán cơ bản AND, OR, Đảo. Sử dụng các phép toán NAND và NOR cũng rất dễ dàng thực hiện 3 phép toán cơ bản trên. Ví dụ:

$$Z = A \cdot B = \overline{\overline{A} \cdot \overline{B}} = \overline{\overline{A} + \overline{B}}$$

$$Z = A + B = \overline{\overline{A} \cdot \overline{B}} = \overline{\overline{A} + \overline{B}}$$

$$Z = \overline{A} = \overline{A \cdot 1} = \overline{A + 0}$$

Vậy phép tính NAND và NOR là hoàn hảo, vạn năng. Cho nên, trong các mạch điện vi mạch số, các cổng NAND và NOR trở thành các phân tử cơ bản, điển hình.

2/ Biểu thức NAND — NAND

Vì trong công tác thực tế ta dùng nhiều cổng NAND, nên việc chuyển đổi biểu thức OR-AND đã tối thiểu hóa thành biểu thức NAND-NAND là công việc vô cùng quan trọng.

Biểu thức NAND-NAND tối thiểu phải là:

- Ký hiệu đảo ít nhất (không kể đến đảo biến, đảo biến coi như có sẵn)
- Số cổng NAND cần có cũng ít nhất.
- Số biến vào ít nhất (số đầu vào của cổng NAND ít nhất)

Đối với mạch điện 2 cấp cổng NAND, dùng phương pháp Đảo 2 lần thì có thể chuyển đổi biểu thức OR-AND tối thiểu thành biểu thức NAND-NAND tối thiểu.

Ví dụ 3-3-12: Hãy chuyển đổi hàm $Z = A\bar{B} + B\bar{C} + C\bar{A}$ thành biểu thức NAND-NAND tối thiểu.

Giải: áp dụng định lý De Morgan đến cấp số hạng (không đến cấp biến số):

$$\bar{Z} = \overline{A\bar{B} + B\bar{C} + C\bar{A}} = \overline{A\bar{B}} \cdot \overline{B\bar{C}} \cdot \overline{C\bar{A}}$$

Lấy đảo lần nữa

$$Z = \overline{\bar{Z}} = \overline{\overline{A\bar{B}} \cdot \overline{B\bar{C}} \cdot \overline{C\bar{A}}}$$

Mạch điện dùng cổng NAND thực hiện, xem hình 3-3-9

3/ Biểu thức NORAND tối thiểu:

Chúng ta cần biểu thức NORAND tối thiểu khi muốn dùng cổng NORAND để thực hiện hàm logic. Nói chung, cho phép xuất hiện ký hiệu Đảo kết hợp với NORAND.

Có hai phương pháp chuyển đổi từ biểu thức OR-AND tối thiểu thành biểu thức NORAND 2 cấp tối thiểu.

a/ Lấy Đảo 2 lần đối với hàm Z

Ví dụ:

$$Z = A\bar{B} + C\bar{D}$$

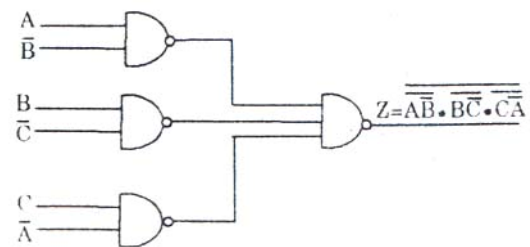
$$Z = \overline{\overline{A\bar{B} + C\bar{D}}}$$

Mạch điện thực hiện Z xem hình 3-3-10

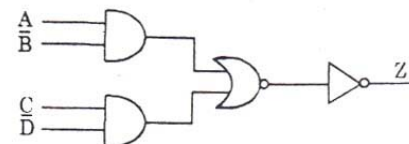
b/ Tìm biểu thức OR-AND tối thiểu của Z, sau đó lấy đảo lần nữa

$$\text{Ví dụ: } Z = AB + \bar{A}\bar{B}$$

Tìm biểu thức OR-AND tối thiểu của $\bar{Z} = A\bar{B} + \bar{A}B$



Hình 3-3-9



Hình 3-3-10

Lấy đảo lần nữa, ta có

$$Z = \overline{\overline{Z}} = \overline{AB + \overline{AB}}$$

Mạch điện thực hiện Z xem hình 3-3-11a (sơ đồ 3-3-11b là kết quả phương pháp lấy đảo 2 lần nói trên, rõ ràng phức tạp hơn)

4/ Biểu thức NOR-NOR tối thiểu

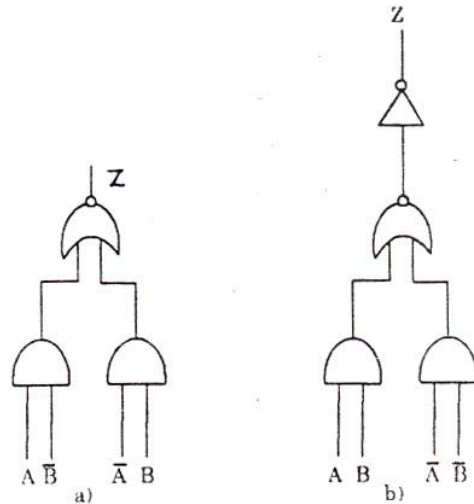
Biểu thức NOR-NOR tối thiểu được dùng khi chúng ta muốn thực hiện hàm logic bằng cổng NOR.

Biểu thức NOR-NOR tối thiểu là:

- Số cổng đảo ít nhất.
- Số đầu vào ít nhất

Cách thực hiện hàm NOR-NOR tối thiểu (2 cấp):

- Tìm biểu thức OR-AND tối thiểu của $\overline{Z}$
- áp dụng định lý De Morgan để có biểu thức tối thiểu dạng AND-OR
- Lấy đảo 2 lần nữa để có biểu thức NOR-NOR tối thiểu



Hình 3-3-11

Ví dụ 3-3-1: Hãy viết hàm $Z = AB + BC + CA$ dưới dạng biểu thức NOR-NOR tối thiểu.

Giải:

- Tìm $\overline{Z}$ dạng OR —AND:

Vẽ bảng Karnaugh của hàm Z và gộp các số hạng nhỏ nhất tương ứng $Z=0$ (hình 3-3-12)

$$\text{Ta được: } \overline{Z} = \overline{A} \overline{B} + \overline{B} \overline{C} + \overline{C} \overline{A}$$

- áp dụng định lý De Morgan để có dạng AND-OR:

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

$$= (A+B) (B+C) (C+A)$$

- Lấy đảo 2 lần để có dạng NOR-NOR:

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

- Mạch điện logic xem hình 3-3-13

A \ BC				
	00	01	11	10
0	0	0	1	0
1	0	1	1	1

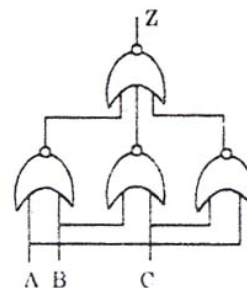
Hình 3-3-12

3.4 Sự tối thiểu hóa hàm logic ràng buộc

3.4.1. Khái niệm ràng buộc, điều kiện ràng buộc

1/ Ràng buộc, phân tử ràng buộc, điều kiện ràng buộc:

a/ Ràng buộc: Ràng buộc là khái niệm quan trọng nói về mối quan hệ qui định lẫn nhau giữa các biến dạng trong một hàm logic.



Hình 3-3-13

Ví dụ: 3-4-1: Ngày lễ quốc tế Phụ nữ 8-3, một đơn vị nọ tổ chức chiêu đãi phim, vé chỉ phát cho phụ nữ của đơn vị .
Hãy xem xét vấn đề logic đó.

Giải: Căn cứ để có thể liệt kê bảng chức năng 3-4-1

Bảng 3-4-1

Thuộc đơn vị không	Nam hay nữ	Có vé không	Được vào rạp không	Thuyết minh
Không	Nam	Không	Không	Không có trường hợp này
Không	Nam	Có		
Không	Nữ	Không	Không	
Không	Nữ	Có		
Có	Nam	Không	Không	
Có	Nam	Có		
Có	Nữ	Không	Không	
Có	Nữ	Có	Có	

Nếu dùng A, B, C biểu thị các biến logic tương ứng 3 cột đầu bảng 3-4-1, các biến lấy giá trị 0/1 tương ứng không /có thuộc đơn vị, nam/nữ, không/có vé, và nếu dùng Z biểu thị hàm logic $Z = 0/1$ tương ứng không/có được vào rạp xem phim, thì ta có bảng chân lí 3-4-2 sau:

Bảng 3-4-2 :

A	B	C	Z	Thuyết minh
0	0	0	0	
0	0	1	x	Không xảy ra
0	1	0	0	
0	1	1	x	Không xảy ra
1	0	0	0	
1	0	1	x	Không xảy ra
1	1	0	0	
1	1	1	1	

Bảng 3-4-2 cho biết rằng, các biến A, B, C chỉ lấy các giá trị có thể là 000, 010, 100, 110, 111 và không thể lấy các giá trị 001, 011, 101 vì vé xem phim chỉ phát cho phụ nữ của đơn vị. Vậy giữa các biến A, B, C có một quan hệ ràng buộc nhất định, hay là ta gọi chúng là một nhóm biến ràng buộc. Hàm logic ràng buộc là hàm có các biến ràng buộc.

b) Phần tử ràng buộc : Các số hạng nhỏ nhất có các tổ hợp giá trị không xảy ra, như trong ví dụ 3-4-1, là $\overline{A}\overline{B}C$, $\overline{A}BC$, $A\overline{B}C$. Những giá trị đó được gọi là số hạng ràng buộc.

Như ta đã biết về số hạng nhỏ nhất, mỗi số hạng nhỏ nhất có một tổ hợp giá trị các biến tương ứng để nó lấy giá trị 1, số hạng ràng buộc luôn bằng 0 vì tổ hợp giá trị các biến tương ứng để nó lấy giá trị 1 thuộc vào giá trị không xảy ra.

c) Điều kiện ràng buộc : Biểu thức logic cấu trúc bằng tổng các số hạng ràng buộc được gọi là điều kiện ràng buộc. Vì số hạng ràng buộc luôn bằng 0, nên tổng các số hạng ràng buộc cũng bằng 0, vậy điều kiện ràng buộc bằng 0.

2) Phương pháp biểu thị điều kiện ràng buộc

a) Trong bảng chân lí dùng dấu “x” biểu thị, xem ví dụ bảng 3-4-2 trên đây. Giá trị hàm số tương ứng tổ hợp giá trị biến của số hạng ràng buộc có dấu chéo “x”.

b) Trong biểu thức logic dùng đẳng thức điều kiện ràng buộc bằng 0 để biểu thị. Ví dụ, từ bảng 3-4-2 ta có :

$$\overline{A}\overline{B}C, \overline{A}BC, A\overline{B}C = 0 \text{ hay là } \Sigma (1,3,5) = 0$$

hay dưới dạng OR-AND tối thiểu :

$$\overline{A}C + \overline{B}C = 0$$

A \ BC	BC			
	00	01	11	10
0	0	x	x	0
1	0	x	1	0

Hình 3-4-1.

c) Trong bảng Karnaugh dùng dấu “x” biểu thị, ví dụ, tương ứng với bảng 3-4-2 là bảng Karnaugh hình 3-4-1.

3-4-2. Tối thiểu hoá hàm logic ràng buộc

Mấu chốt của vấn đề là dùng điều kiện ràng buộc như thế nào. Nói chung nếu tận dụng điều kiện ràng buộc thì có thể nhận được biểu thức tối thiểu hoá.

1) Ứng dụng điều kiện ràng buộc để tối thiểu hoá

a) Phương pháp công thức

Trong phương pháp công thức, tùy yêu cầu, có thể tùy ý cộng thêm hoặc khử bỏ số hạng ràng buộc. Số hạng ràng buộc bằng 0, nên thêm bớt 0 vào biểu thức logic không làm thay đổi giá trị biểu thức đó. Ví dụ, bảng 3-4-2, ta có : $Z = ABC$.

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

b) Phương pháp hình vẽ

Trong phương pháp hình vẽ, tùy yêu cầu, có thể tùy ý khoanh vòng qua số hạng ràng buộc. Vì số hạng ràng buộc bằng 0, nên sự gộp thêm nó không làm thay đổi giá trị hàm số. Ví dụ hình 3-4-1, có thể khoanh vòng 4 ô giữa, ta có :

$$Z = m_1 + m_3 + m_5 + m_4 = C$$

Phân tích bảng 3-4-1, 3-4-2, chúng ta sẽ rõ ý nghĩa thực tế của việc tối thiểu hoá dùng điều kiện ràng buộc. $Z = ABC$ nghĩa là người gác rạp chiếu phim phải kiểm tra vé, lại phải phân biệt giới tính và nơi công tác của người có vé : Phiền phức quá. $Z = C$ nghĩa là người gác rạp chiếu phim chỉ cần kiểm tra vé : Thật đơn giản. Tuy nhiên, điều kiện ràng buộc phải bảo đảm là vé chỉ phát cho phụ nữ của đơn vị mà thôi. Nếu không, người không được phép cũng vào xem phim. Nói một cách tổng quát, cần bảo đảm điều kiện ràng buộc khi tối thiểu hoá hàm logic ràng buộc. Nếu không, có thể sinh ra sai lầm.

2) Tối thiểu hoá hàm logic có biến loại trừ nhau

Trong một nhóm biến số, nếu chỉ có một biến lấy giá trị 1 thì giá trị các biến khác phải bằng 0. Đó là các loại trừ nhau.

Ví dụ 3-4-2 : Các biến A, B, C của hàm Z là loại trừ nhau.

Hãy kê bảng chân lí của Z và dùng phương pháp công thức, phương pháp hình vẽ để tìm biểu thức hàm số tối thiểu dạng ORAND.

Giải :

- Bảng 3-4-4 dưới đây căn cứ vào khái niệm các biến loại trừ nhau.

Bảng 3-4-4 : DẤU x BIỂU THỊ TRƯỜNG HỢP KHÔNG XẢY RA

A	B	C	Z
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	x
1	0	0	1
1	0	1	x
1	1	0	x
1	1	1	x

- Dùng công thức tối thiểu hoá

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

Điều kiện ràng buộc $\overline{A}BC + A\overline{B}C + ABC + \overline{A}\overline{B}\overline{C} = 0$

$$\begin{aligned}
Z &= \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + \overline{A}BC + A\overline{B}C + A\overline{B}\overline{C} + ABC \\
&= (\overline{A}\overline{B}C + \overline{A}B\overline{C}) + (\overline{A}B\overline{C} + A\overline{B}\overline{C}) + (A\overline{B}\overline{C} + ABC) + ABC \\
&= \overline{A}C + B\overline{C} + (A\overline{B} + ABC) \\
&= \overline{A}C + B\overline{C} + A(\overline{B} + BC) \\
&= \overline{A}C + B\overline{C} + A\overline{B} + AC \\
&= (\overline{A}C + AC) + B\overline{C} + A\overline{B} \\
&= C + B\overline{C} + A\overline{B} \\
&= A + B + C
\end{aligned}$$

- Dùng hình vẽ tối thiểu vẽ bảng Karnaugh như hình 3-4-2.

Gộp theo khoanh vòng

$$m_4 + m_5 + m_6 + m_7 = A$$

$$m_2 + m_3 + m_6 + m_7 = B$$

$$m_1 + m_3 + m_5 + m_7 = C$$

Hàm tối thiểu hoá : $Z = A + B + C$

Đối với hàm logic mà các biến loại trừ nhau, bảng chân lí thường có dạng đơn giản đặc biệt, ví dụ, từ bảng 3-4-4 ta có bảng 3-4-5.

Bảng 3-4-5

Biến	Hàm Z
A	1
B	1
C	1

Hàm logic có biểu thức dạng tổng các biến : $Z = A + B + C$

3) Ví dụ về tối thiểu hoá

Ví dụ 3-4-3 : hãy tối thiểu hoá hàm $Z = AC + \overline{A}\overline{B}C$

điều kiện ràng buộc $\overline{B}\overline{C} = 0$

Giải :

A \ BC	BC			
	00	01	11	10
0	0	1	x	1
1	1	x	x	x

Hình 3-4-2.

A \ BC	BC			
	00	01	11	10
0	x	1	0	0
1	x	1	1	0

Hình 3-4-3.

- Dùng công thức

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

$$Z = AC + \overline{B}$$

- Dùng hình vẽ (hình 3-4-3)

- Gộp khoang vòng

$$m_0 + m_1 + m_4 + m_5 = \overline{B}$$

$$m_5 + m_7 = AC$$

$$Z = \overline{B} + AC$$

Ví dụ 3-4-4 : Cho bảng chân lí của hàm logic ràng buộc (bảng 3-4-6). Hãy tìm biểu thức tối thiểu hoá của hàm dạng OR- AND và dạng AND-OR.

Giải : Từ bảng chân lí 3-4-6 vẽ bảng Karnaugh hình 3-4-4.

Bảng 3-4-6 :

A	B	C	D	Z
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	x
1	0	1	1	x
1	1	0	0	x
1	1	0	1	x
1	1	1	0	x
1	1	1	1	x

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

Hình 3-4-4.

- Tìm dạng OR-AND :

Khi khoanh vòng m_{10} , m_{11} , m_{12} , m_{13} , m_{14} , m_{15} được tận dụng xử lý gộp, ta có :

$$Z = \overline{B}\overline{D} + \overline{B}C + BD + A \quad (3-4-1)$$

- Tìm dạng AND-OR :

Nếu trên đây gộp các số hạng ràng buộc vào vòng các ô giá trị 1, thì ở đây ta chia số hạng ràng buộc thành hai nhóm :

m_{12} và m_{14} gộp vào các ô giá trị 0, còn m_{10} , m_{11} , m_{12} và m_{15} gộp vào các ô giá trị 1. Ta có : $\overline{Z} = \overline{A}\overline{B}\overline{C}\overline{D} + \overline{B}\overline{D}$ (3-4-2)

Lấy đảo lần nữa ta có :

$$\begin{aligned} Z &= \overline{\overline{Z}} = \overline{\overline{A}\overline{B}\overline{C}\overline{D} + \overline{B}\overline{D}} \\ &= \overline{\overline{A}\overline{B}\overline{C}\overline{D}} \cdot \overline{\overline{B}\overline{D}} \\ &= (A + B + C + \overline{D})(\overline{B} + D) \end{aligned}$$

Cần thuyết minh thêm điều sau đây ; hàm Z (3-4-1) và hàm đảo Z (3-4-2) không đơn giản là đảo của nhau, nếu quên đi điều kiện ràng buộc. Nói cách khác, nếu không có điều kiện ràng buộc thì (3-4-1) và (3-4-2) không thể là các biểu thức đảo của nhau. Xuất xứ của chúng là do trong quá trình xử lý các số hạng ràng buộc, ta gộp tất cả vào vòng các ô giá trị 1 để đi đến (3-4-1), nhưng lại chia chúng thành hai nhóm để đi đến (3-4-2). Muốn lấy đảo để chuyển đổi qua lại hai biểu thức hàm số (3-4-1) và (3-4-2), ta phải tận dụng điều kiện ràng buộc.

CHƯƠNG 4

MẠCH LOGIC TỔ HỢP

4.1 ĐẶC ĐIỂM CƠ BẢN VÀ PHƯƠNG PHÁP THIẾT KẾ CỦA MẠCH LOGIC TỔ HỢP

4.1.1. Phương pháp biểu thị và phân tích đặc điểm cơ bản và chức năng logic của mạch logic tổ hợp

Căn cứ vào đặc điểm và chức năng logic, chúng ta phân loại mạch số thành hai loại chính : mạch tổ hợp và mạch dãy (mạch dãy được trình bày trong chương sau).

1) Đặc điểm cơ bản của mạch tổ hợp

Trong mạch số, mạch tổ hợp là mạch mà trị số ổn định của tín hiệu đầu ra ở thời điểm bất kỳ chỉ phụ thuộc vào tổ hợp các giá trị tín hiệu đầu vào ở thời điểm đó. Trong mạch tổ hợp, trạng thái mạch điện trước thời điểm xét, tức trước khi có tác động của tín hiệu đầu vào, không ảnh hưởng đến tín hiệu đầu ra. Đặc điểm cấu trúc mạch tổ hợp là được cấu trúc nên từ các cổng logic. Vậy các mạch điện cổng ở chương 2 và các mạch logic đã gặp ở chương 3 đều là mạch tổ hợp.

2) Phương pháp biểu thị chức năng logic

Các phương pháp thường dùng để biểu thị chức năng logic của mạch tổ hợp là hàm số logic, bảng chân lí, sơ đồ logic, bảng Karnaugh, cũng có khi biểu thị bằng đồ thị thời gian dạng sóng.

Đối với vi mạch cỡ nhỏ (SSI) thường biểu thị bằng hàm logic. Đối với vi mạch cỡ vừa thường biểu thị bằng bảng chân lí, hay là bảng chức năng. Bảng chức năng dùng hình thức bảng kê, với mức logic cao (H) và mức logic thấp (L), để mô tả quan hệ logic giữa tín hiệu đầu ra với tín hiệu đầu vào của mạch điện xét. Chỉ cần thay giá trị logic cho trạng thái trong bảng chức năng, thì ta có bảng chân lí tương ứng.

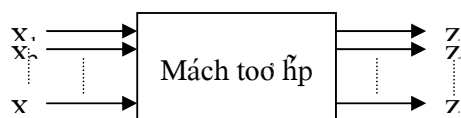
Như hình 4-1-1 cho biết, thường có nhiều tín hiệu đầu vào và nhiều tín hiệu đầu ra. Một cách tổng quát, hàm logic của tín hiệu đầu ra có thể viết dưới dạng :

$$z_1 = f_1(x_1, x_2, \dots, x_n)$$

$$z_2 = f_2(x_1, x_2, \dots, x_n)$$

...

$$z_m = f_m(x_1, x_2, \dots, x_n)$$



Hình 4-1-1

Sơ đồ logic khối mạch tổ hợp.

Cũng có thể viết dưới dạng đại lượng vectơ như sau :

$$Z = F(X)$$

3) Phương pháp phân tích chức năng logic

Chương 2 cũng đã đề cập. Bây giờ giới thiệu các bước phân tích, bắt đầu từ sơ đồ mạch logic đã cho, để cuối cùng tìm ra hàm logic hoặc bảng chân lí :

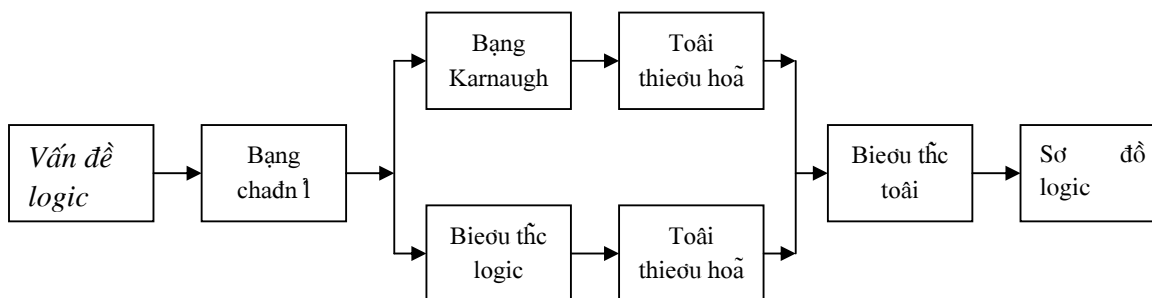
a) *Viết biểu thức* : tuân tự từ đầu vào đến đầu ra (hay cũng có thể ngược lại) viết ra biểu thức hàm logic của tín hiệu đầu ra.

b) *Rút gọn* : khi cần thiết thì rút gọn đến tối thiểu biểu thức trên đây bằng phương pháp đại số hay phương pháp hình vẽ.

c) *Kê bảng chân lí* : khi cần thiết thì tìm ra bảng chân lí bằng cách tiến hành tính toán các giá trị hàm logic tín hiệu đầu ra tương ứng với tổng hợp có thể của các giá trị tín hiệu đầu vào.

4.1.2. Phương pháp thiết kế logic mạch tổ hợp

Phương pháp thiết kế logic là các bước cơ bản tìm ra sơ đồ mạch điện logic từ yêu cầu nhiệm vụ logic đã cho.



Hình 4-1-2. Các bước thiết kế mạch logic tổ hợp

Hình 4-1-2 là quá trình thiết kế nói chung của mạch tổ hợp, trong đó bao gồm 4 bước chính :

1) Phân tích yêu cầu

Yêu cầu nhiệm vụ tl của vấn đề logic thực có thể là một đoạn văn, cũng có thể là một bài toán logic cụ thể. Nhiệm vụ phân tích là xác định cái nào là biến số đầu vào, cái nào là 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. Việc đó là một việc khó nhưng quan trọng trong quá trình thiết kế.

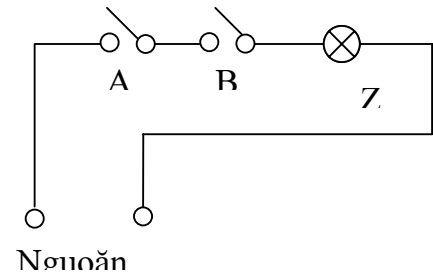
2) Kê bảng chân lí

Nói chung, đầu tiên, chúng ta liệt kê thành bảng về quan hệ tương ứng nhau giữa trạng thái tín hiệu đầu vào với trạng thái hàm số đầu ra. Đó là bảng kê yêu cầu chức năng logic, gọi tắt là bảng chức năng. Việc này có vẻ dễ và trực quan. Tiếp theo, ta thay giá trị logic cho trạng thái, tức là dùng các số 0, số 1 biểu thị các trạng thái tương ứng của đầu vào và đầu ra. Kết quả ta có bảng các giá trị thực logic, gọi tắt là là bảng chân lí. Đây chính là hình thức đại số của yêu cầu thiết kế.

Cần lưu ý rằng từ một bảng chức năng có thể được bảng chân lí khác nhau nếu thay giá trị logic khác nhau (tức là quan hệ logic giữa đầu ra với đầu vào cũng phụ thuộc việc thay giá trị). Ví dụ, sơ đồ mạch nguyên lí hình 4-1-3 dùng 2 chuyển mạch A, B mắc nối tiếp điều khiển bóng đèn Z.

Bảng 4-1-1 : BẢNG CHỨC NĂNG

Chuyển n mạch A	Chuyển n mạch B	Bóng đèn Z
Ngắt	Ngắt	Tắt
Ngắt	Ngắt	t
Nối	Nối	Tắt
Nối	Nối	t
		Tắt
		t
		Sáng



Hình 4-1-3. Mạch điện hai chuyển mạch nối tiếp.

Bảng chức năng 4-1-1 có được từ xem xét trực tiếp các khả năng có thể của mạch điện hình 4-1-3. Nếu thay thế giá trị logic theo 4 cách khác nhau thì từ bảng chân lí 4-1-2a, b, c, d ta được các biểu thức logic khác nhau.

Bảng 4-1-2 : BẢNG CHÂN LÍ TRONG 4 TÌNH HUỐNG THAY GIÁ TRỊ KHÁC NHAU

A	B	Z
0	0	0

A	B	Z
1	1	1

0	1	0
1	0	0
1	1	1

a) 0 biểu thị ngắt và tắt

1 biểu thị nối và sáng $Z = A.B$

tắt $Z = A + B$

A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

c) 0 biểu thị ngắt và sáng

1 biểu thị nối và tắt $Z = A.B$

$A + B$

1	0	1
0	1	1
0	0	0

b) 0 biểu thị nối và sáng

1 biểu thị ngắt và

A	B	Z
1	1	0
1	0	0
0	1	0
0	0	1

d) 0 biểu thị nối và tắt

1 biểu thị ngắt và sáng $Z =$

Từ bảng 4-1-2, ta thấy rằng chúng ta sẽ có mối quan hệ logic khác nhau nếu thay giá trị theo cách khác nhau. Chúng ta phải căn cứ vào giá trị thay thế trạng thái để xác định ý nghĩa cụ thể của 0 và 1 (tức là ý nghĩa thực tế của bảng chân lí).

Khi liệt kê chức năng hoặc bảng chân lí, có thể không liệt kê các tổ hợp trạng thái tín hiệu đầu vào nào không thể có thể có hay bị cấm. Những tổ hợp này cũng có thể được liệt kê, nhưng tại đầu ra, ở trạng thái tương ứng ta ghi một dấu chéo “x”, thường sử dụng các trạng thái đánh dấu chéo để tối thiểu hoá hàm logic.

3) Tiến hành tối thiểu hoá

Nếu số biến số tương đối ít thì có thể dùng phương pháp hình vẽ. Nếu số biến số tương đối nhiều, khi đó không tiện dùng phương pháp hình vẽ, thì dùng phương pháp đại số.

4) Vẽ sơ đồ logic

Kết quả việc tối thiểu hoá là biểu thức logic OR – AND. Căn cứ việc chọn lựa loại cổng logic cụ thể, cần biến đổi biểu thức logic đó thành dạng phù hợp. Ví dụ, nếu chọn dùng cổng NAND thì phải có biểu thức dạng AND, nếu chọn dùng cổng NOR thì

phải có biểu thức dạng NOR, nếu chọn dùng cổng NORAND thì phải có biểu thức dạng NORAND.

Cần lưu ý rằng, những bước thiết kế trên đây không phải là bắt buộc áp dụng máy móc, mà là nên được vận dụng linh hoạt theo tình huống cụ thể của thiết kế thực tế.

Trong hoạt động thực tiễn của loài người, các bài toán, và tương ứng các mạch điện của mạch logic tổ hợp là nhiều không kể hết. Các mạch tổ hợp hiện nay thường gặp trong hệ thống số là bộ mã hoá, bộ giải mã, bộ chọn kênh, bộ cộng, bộ kiểm tra chẵn lẻ (parity checker), ROM, v.v...

Dưới đây lần lượt giới thiệu một số mạch tổ hợp về cấu trúc mạch, nguyên lý công tác và đặc điểm của chúng, kết hợp soi sáng phương pháp thiết kế phân tích mạch tổ hợp.

4.2 BỘ MÃ HOÁ

4.2.1 Khái niệm mã hoá

Nói một cách khái quát, mã hoá là dùng văn tự, kí hiệu hay mã để biểu thị một đối tượng xác định. Có thể thấy biết bao ví dụ về mã hoá trong đời sống như đặt tên cho trẻ sơ sinh, mỗi vận động viên có một số thay thế trong thi đấu. Tên trẻ là văn tự, vận động viên đánh số theo hệ đếm thập phân. Văn tự và hệ đếm thập phân không tiện dùng cho mạch số. Mã hoá nhị phân là quá trình dùng mã nhị phân để biểu thị đối tượng xét đến (đối tượng này là tín hiệu). Mã nhị phân chỉ có hai số 0 và 1, dễ dàng biểu thị bằng trạng thái mạch điện, nên được dùng rộng rãi trong mạch số. Biểu thị số lượng nhiều thì tăng số bit (BINary digiT) trong cách viết số dựa theo vị trí. Mã nhị phân 1 bit có 2 trạng thái (0, 1) tương ứng 2 tín hiệu. Mã nhị phân 2 bit có 4 trạng thái (00, 01, 10, 11) tương ứng 4 tín hiệu. Tổng quát mà nói, mã nhị phân n bit có 2^n trạng thái, có thể biểu thị 2^n tín hiệu. Vậy, để tiến hành mã hoá n tín hiệu cần sử dụng n bit, theo công thức $2^n \geq N$.

Bộ mã hoá là mạch điện thực hiện thao tác mã hoá. Căn cứ vào yêu cầu và đặc điểm khác nhau của tín hiệu được mã hoá, chúng ta có các bộ mã hoá khác nhau, như bộ mã hoá nhị phân, bộ mã hoá nhị – thập phân, bộ mã hoá ưu tiên v.v...

4.2.2. Bộ mã hoá nhị phân

Bộ mã hoá nhị phân là mạch điện dùng n bit để mã hoá $N = 2^n$ tín hiệu. Chúng ta xét ví dụ dưới đây để thuyết minh nguyên lý công tác và quá trình thiết kế bộ mã hoá nhị phân.

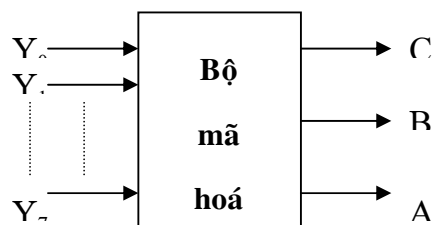
Ví dụ 4-2-1

Hãy thiết kế bộ mã hoá thực hiện mã hoá 8 tín hiệu $Y_0, Y_1, \dots, Y_7$ theo mã nhị phân.

Bài giải :

1. Phân tích yêu cầu.

Đối tượng được mã hoá là 8 tín hiệu đầu vào, tức là $Y_0, Y_1, \dots, Y_7$. Căn cứ vào công thức $N = 2^n = 8$ ta thấy đầu ra là mã nhị phân $n = 3$ bit, dùng A, B, C biểu thị. Xem hình 4-2-1.



Hình 4-2-1
Sơ đồ khối yêu cầu thiết kế

2. Kê bảng chân lí

Việc mã hoá được tiến hành chỉ với 1 tín hiệu đầu vào ở một thời điểm. Ở đầu vào không cho phép có đồng thời từ 2 tín hiệu trở lên là logic 1, tức là $Y_0, Y_1, \dots, Y_7$ không cùng nhau. Vậy quan hệ logic giữa đầu ra với đầu vào có thể biểu thị bằng bảng chân lí hay bảng mã hoá.

Bảng 4.2.1 :

	C	B	A
Y_0	0	0	0
Y_1	0	0	1
Y_2	0	1	0
Y_3	0	1	1
Y_4	1	0	0
Y_5	1	0	1
Y_6	1	1	0
Y_7	1	1	1

a) Phương án 1

	C	B	A
Y_0	0	0	0
Y_1	0	0	1
Y_2	0	1	1
Y_3	0	1	0
Y_4	1	1	0
Y_5	1	1	1
Y_6	1	0	1
Y_7	1	0	0

b) Phương án 2

Dùng mã nhị phân 3 bit biểu thị 8 tín hiệu đầu vào có thể có nhiều phương án khác nhau, Bảng 4.2.1 giới thiệu 2 phương án. Phương án 1 theo số đếm nhị phân. Phương án 2 là mã chu kì 3 bit.

3. Tối thiểu hoá

$Y_0 \div Y_7$ là không đồng thời. Chỉ cần lấy các biến nào làm cho hàm số đầu ra bằng 1 cộng lại thì ta có biểu thức tối thiểu hoá dạng OR - AND. Từ bảng 4.2.1.a (Phương án 1) ta có biểu thức hàm số đầu ra :

$$C = Y_4 + Y_5 + Y_6 + Y_7$$

$$B = Y_2 + Y_3 + Y_6 + Y_7$$

$$A = Y_1 + Y_3 + Y_5 + Y_7$$

Từ bảng 4.2.1.b (Phương án 2) ta có biểu thức hàm số đầu ra :

$$C = Y_4 + Y_5 + Y_6 + Y_7$$

$$B = Y_2 + Y_3 + Y_4 + Y_5$$

$$A = Y_1 + Y_2 + Y_5 + Y_6$$

4. Vẽ sơ đồ logic

Chọn dùng cổng NAND. Vậy cần biến đổi dạng AND – OR thành dạng NAND.

Cụ thể như sau.

Phương án 1

$$C = \overline{\overline{Y_4 + Y_5 + Y_6 + Y_7}} = \overline{\overline{Y_4} \cdot \overline{Y_5} \cdot \overline{Y_6} \cdot \overline{Y_7}}$$

$$B = \overline{\overline{Y_2 + Y_3 + Y_6 + Y_7}} = \overline{\overline{Y_2} \cdot \overline{Y_3} \cdot \overline{Y_6} \cdot \overline{Y_7}}$$

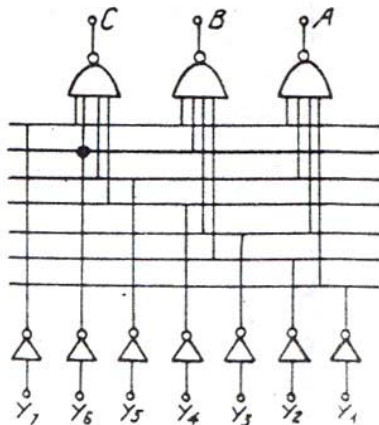
$$A = \overline{\overline{Y_1 + Y_3 + Y_5 + Y_7}} = \overline{\overline{Y_1} \cdot \overline{Y_3} \cdot \overline{Y_5} \cdot \overline{Y_7}}$$

Phương án 2

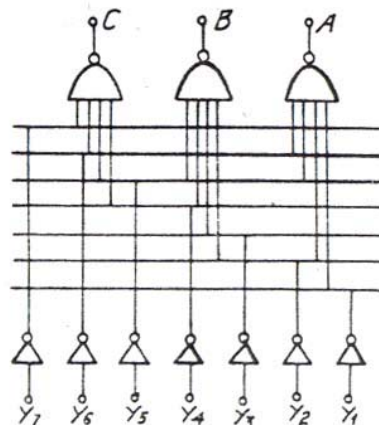
$$C = \overline{\overline{Y_4 + Y_5 + Y_6 + Y_7}} = \overline{\overline{Y_4} \cdot \overline{Y_5} \cdot \overline{Y_6} \cdot \overline{Y_7}}$$

$$B = \overline{\overline{Y_2 + Y_3 + Y_4 + Y_5}} = \overline{\overline{Y_2} \cdot \overline{Y_3} \cdot \overline{Y_4} \cdot \overline{Y_5}}$$

$$A = \overline{\overline{Y_1 + Y_2 + Y_5 + Y_6}} = \overline{\overline{Y_1} \cdot \overline{Y_2} \cdot \overline{Y_5} \cdot \overline{Y_6}}$$



a) Mạch phương án 1



b) Mạch phương án 2

Cần lưu ý rằng, trong sơ đồ logic hình 4.2.2, sự mã hoá của Y_0 được ngầm định : Khi $Y_1 \div Y_7$ đều bằng 0, trạng thái đầu ra của mạch (CBA = 000) là mã của Y_0 .

Trong bài toán phân tích, từ sơ đồ logic bộ mã hoá đã cho, ta phân tích mối quan hệ logic giữa đầu vào và đầu ra, tuần tự cho tín hiệu đầu vào đến bộ mã hoá, rồi căn cứ vào mức logic (0 hoặc 1) của các đầu ra mạch điện, tìm ra mã tương ứng, liệt kê thành bảng mã hoá.

4.2.3 Bộ mã hoá nhị – thập phân

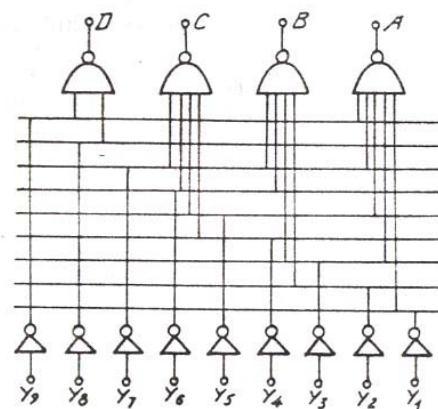
Bộ mã hoá nhị – thập phân là mạch điện chuyển mã hệ thập phân bao gồm 10 chữ số 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 thành mã hệ nhị phân.

Đầu vào là 10 chữ số, đầu ra là nhóm mã số nhị phân, gọi là mã nhị – thập phân (BCD – Binary – Coded – Decima). Căn cứ vào công thức $2^n \geq N = 10$, ta có $n = 4$. Mã nhị phân 4 bit có 16 trạng thái (từ mã). Chỉ cần chọn 10 từ mã tùy ý trong số đó là đủ để biểu thị 10 tín hiệu đầu vào. Vậy có rất nhiều phương án. Một số mã nhị – thập phân thường dùng xem giới thiệu ở 4.2.5. Cũng với lí do như đã biết ở ví dụ 4.2.1, 10 tín hiệu đầu vào tồn tại độc lập nhau.

Nguyên lí công tác và quá trình thiết kế của bộ mã hoá nhị – thập phân tương tự như bộ mã hoá nhị phân xét ở ví dụ 4.2.1. Dưới đây lấy mã 8421 là ví dụ để thuyết minh. Bảng 4.2.2 là bảng mã hoá BCD 8421 rất thường dùng

Bảng 4-2-2 : BẢNG MÃ HOÁ BCD 8421

Số thập phân	D	C	B	A
0(Y_0)	0	0	0	0
1(Y_1)	0	0	0	1
2(Y_2)	0	0	1	0
3(Y_3)	0	0	1	1
4(Y_4)	0	1	0	0
5(Y_5)	0	1	0	1
6(Y_6)	0	1	1	0
7(Y_7)	0	1	1	1
8(Y_8)	1	0	0	0
9(Y_9)	1	0	0	1



Hình 4-2-3
Sơ đồ logic bộ mã hoá (dùng mã 8421)

$$D = Y_8 + Y_9 = \overline{Y_8} \cdot \overline{Y_9}$$

$$C = Y_4 + Y_5 + Y_6 + Y_7$$

$$C = \overline{Y_4} \cdot \overline{Y_5} \cdot \overline{Y_6} \cdot \overline{Y_7}$$

$$B = Y_2 + Y_3 + Y_6 + Y_7 = \overline{Y_2} \cdot \overline{Y_3} \cdot \overline{Y_6} \cdot \overline{Y_7}$$

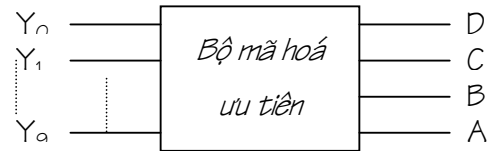
$$A = Y_1 + Y_3 + Y_5 + Y_7 + Y_9 = \overline{Y_1} \cdot \overline{Y_3} \cdot \overline{Y_5} \cdot \overline{Y_7} \cdot \overline{Y_9}$$

4.2.4 Bộ mã hoá ưu tiên

Trong các bộ mã hoá xét trên đây, tín hiệu đầu vào tồn tại độc lập, (không có tình huống có 2 tín hiệu trở lên đồng thời tác động). Bộ mã hoá ưu tiên 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, tất nhiên xuất phát từ yêu cầu thực tiễn. Bây giờ chúng ta sẽ xem xét nguyên lý công tác và quá trình thiết kế của bộ mã hoá ưu tiên qua ví dụ có 10 đầu vào, 4 đầu ra.

Ví dụ 4-2- :

Hãy thiết kế một mạch điện tổ hợp để mã hoá nhị phân đối với 10 tín hiệu đầu vào $Y_0, Y_1, \dots, Y_9$ sao cho mức độ ưu tiên từ cao nhất giảm dần theo chiều $Y_9, Y_8, \dots, Y_0$. Nếu có nhiều tín hiệu đồng thời xuất hiện ở đầu vào thì chỉ tín hiệu nào có mức ưu tiên cao nhất trong số đó mới được mã hoá. Giả thiết cả tín hiệu đầu vào, lẫn tín hiệu đầu ra đều tích cực ở mức thấp.



Bài giải :

1. Phân tích yêu cầu

Căn cứ công thức $2^n \geq N = 10$, vậy ta dùng mã nhị phân $n = 4$ bit.

Hình 4-2-4.

Sơ đồ khối bộ mã hoá ưu tiên.

Bảng 4-2-3 : BẢNG CHỨC NĂNG

Y ₉	Y ₈	Y ₇	Y ₆	Y ₅	Y ₄	Y ₃	Y ₂	Y ₁	Y ₀	D	C	B	A
H	H	H	H	H	H	H	H	H	L	H	H	H	H
H	H	H	H	H	H	H	H	L	X	H	H	H	L
H	H	H	H	H	H	H	L	X	X	H	H	L	H
H	H	H	H	H	H	L	X	X	X	H	H	L	L
H	H	H	H	H	L	X	X	X	X	H	L	H	H
H	H	H	H	L	X	X	X	X	X	H	L	H	L
H	H	H	L	X	X	X	X	X	X	H	L	L	H
H	H	L	X	X	X	X	X	X	X	H	L	L	L
H	L	X	X	X	X	X	X	X	X	L	H	H	H
L	X	X	X	X	X	X	X	X	X	L	H	H	L

Bảng 4-2-4 : BẢNG MÃ HOÁ ƯU TIÊN

Y ₉	Y ₈	Y ₇	Y ₆	Y ₅	Y ₄	Y ₃	Y ₂	Y ₁	Y ₀	D	C	B	A
1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	0	X	1	1	1	0
1	1	1	1	1	1	1	0	X	X	1	1	0	1
1	1	1	1	1	1	0	X	X	X	1	1	0	0
1	1	1	1	1	0	X	X	X	X	1	0	1	1
1	1	1	1	0	X	X	X	X	X	1	0	1	0
1	1	1	0	X	X	X	X	X	X	1	0	0	1
1	1	0	X	X	X	X	X	X	X	1	0	0	0
1	0	X	X	X	X	X	X	X	X	0	1	1	1
0	X	X	X	X	X		X	X	X	0	1	1	0

Theo đề bài, sự mã hoá thực hiện theo mức độ ưu tiên Y₉ đến Y₀, khi các tín hiệu cùng tác động thì các tín hiệu có mức ưu tiên thấp không tác dụng, nghĩa là bất kể mức logic của nó thế nào đều không ảnh hưởng đến đầu ra.

2. Kê bảng chân lí – Bảng mã hoá ưu tiên

Bảng chức năng 4.2.3 phản ánh yêu cầu thiết kế, mã hoá theo cấp ưu tiên, những biến số tương ứng mức ưu tiên thấp không tác dụng gì đến đầu ra có đánh dấu

chéo “x”. Mã nhị phân 4 bit đầu ra có rất nhiều phương án, trong bảng 4.2.4 là phương án mã 8421 với mức logic âm.

3. Tối thiểu hoá

Vì có quá nhiều biến số, nên dùng phương pháp đại số để tối thiểu hoá. Chúng ta sẽ dùng cổng NORAND trong sơ đồ mạch. Đầu tiên ta tìm biểu thức tối thiểu hoá của hg đầu ra dưới dạng OR - AND của hàm đảo, rồi lấy đảo để được dạng NORAND. Khi viết hg từ bảng mã hoá 4.2.4, một cách hình thức, ta không cần để ý những vị trí có dấu “x”. Kết quả ta được các hg đầu ra dưới dạng NORAND như sau :

$$\overline{D} = \overline{Y}_9 + Y_9 \cdot \overline{Y}_8 = \overline{Y}_9 + \overline{Y}_8$$

$$D = \overline{\overline{Y}_9 + \overline{Y}_8}$$

$$\begin{aligned}\overline{C} &= Y_9 Y_8 \overline{Y}_7 + Y_9 Y_8 Y_7 \overline{Y}_6 + Y_9 Y_8 Y_7 Y_6 \overline{Y}_5 + Y_9 Y_8 Y_7 Y_6 Y_5 \overline{Y}_4 \\ &= Y_9 Y_8 \overline{Y}_7 + Y_9 Y_8 \overline{Y}_6 + Y_9 Y_8 \overline{Y}_5 + Y_9 Y_8 \overline{Y}_4\end{aligned}$$

$$C = \overline{Y_9 Y_8 \overline{Y}_7 + Y_9 Y_8 \overline{Y}_6 + Y_9 Y_8 \overline{Y}_5 + Y_9 Y_8 \overline{Y}_4}$$

$$\begin{aligned}\overline{B} &= Y_9 Y_8 \overline{Y}_7 + Y_9 Y_8 Y_7 \overline{Y}_6 + Y_9 Y_8 Y_7 Y_6 Y_5 Y_4 \overline{Y}_3 + Y_9 Y_8 Y_7 Y_6 Y_5 Y_4 Y_3 \overline{Y}_2 \\ &= Y_9 Y_8 \overline{Y}_7 + Y_9 Y_8 \overline{Y}_6 + Y_9 Y_8 Y_5 Y_4 \overline{Y}_3 + Y_9 Y_8 Y_5 Y_4 \overline{Y}_2\end{aligned}$$

$$B = \overline{Y_9 Y_8 \overline{Y}_7 + Y_9 Y_8 \overline{Y}_6 + Y_9 Y_8 Y_5 Y_4 \overline{Y}_3 + Y_9 Y_8 Y_5 Y_4 \overline{Y}_2}$$

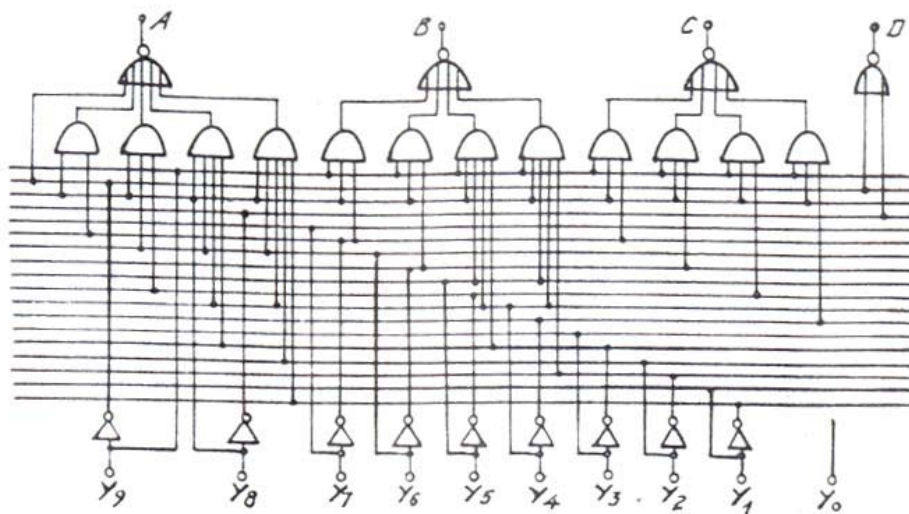
$$\overline{A} = \overline{Y}_9 + Y_9 Y_8 \overline{Y}_7 + Y_9 Y_8 Y_7 Y_6 \overline{Y}_5 + Y_9 Y_8 Y_7 Y_6 Y_5 Y_4 \overline{Y}_3 + Y_9 Y_8 Y_7 Y_6 Y_5 Y_4 Y_3 Y_2 \overline{Y}_1$$

$$A = \overline{\overline{Y}_9 + Y_9 Y_8 \overline{Y}_7 + Y_9 Y_8 Y_7 Y_6 \overline{Y}_5 + Y_9 Y_8 Y_7 Y_6 Y_5 Y_4 \overline{Y}_3 + Y_9 Y_8 Y_7 Y_6 Y_5 Y_4 Y_3 Y_2 \overline{Y}_1}$$

4. Vẽ sơ đồ logic

Hình 4-2-5 là sơ đồ logic thực hiện các hàm đầu ra D, C, B, A dưới dạng cổng NORAND.

Chúng ta có thể nghiệm chứng rằng sơ đồ hình 4-2-5 thỏa mãn yêu cầu thiết kế bằng cách phân tích logic, tìm ra quan hệ logic giữa đầu ra với đầu vào của mạch logic hình 4-2-5.

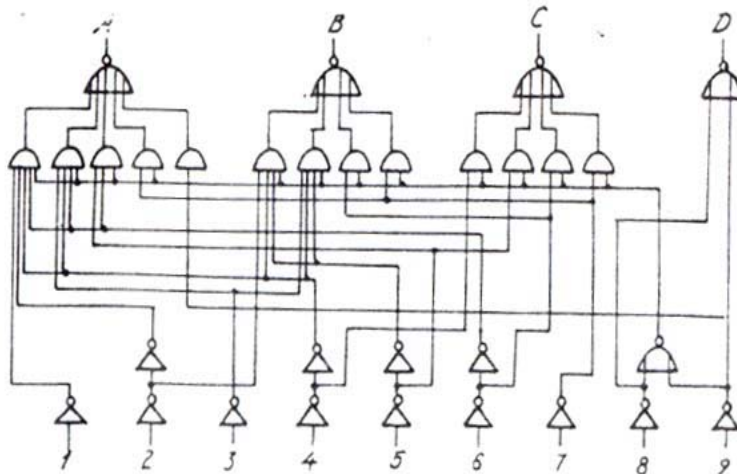


Hình 4-2-5 Bộ mã hoá ưu tiên

Chẳng hạn, nếu tất cả đầu vào đều tích cực (mức logic 0) thì DCBA = 0110, đây là mã tương ứng với đầu vào Y_9 (có mức ưu tiên cao nhất, cao hơn $Y_8 \div Y_0$). Nếu tất cả đầu vào không có tín hiệu (mức logic 1) thì DCBA = 1111, đây là mã (ngầm định) tương ứng với đầu vào Y_0 .

4.2.5. Vi mạch cỡ vừa (MSI) bộ mã hoá ưu tiên

Hình 4-2-6 là vi mạch cỡ vừa bộ mã hoá ưu tiên. Tuy rằng hình 4-2-5 và hình 4-2-6 có đôi chỗ khác nhau, nhưng chúng đều có một chức năng logic duy nhất. Đầu vào, đầu ra đều tích cực mức thấp. Trong hình 4-2-6, để đơn giản, các tín hiệu đầu vào được kí hiệu 0, 1, 2, ..., 9 (thay cho $Y_0, Y_1, Y_2, \dots, Y_9$). Căn cứ vào sơ đồ logic hình 4-2-6, ta viết ra biểu thức hàm đầu ra như sau :



Hình 4-2-6 Bộ mã hoá ưu tiên (MSI)

$$D = \overline{\overline{9+8}}$$

$$C = \overline{\overline{\overline{9+87} + \overline{9+86} + \overline{9+85} + \overline{9+84}}}$$

$$= \overline{\overline{9.8.7} + \overline{9.8.6} + \overline{9.8.5} + \overline{9.8.4}}$$

$$B = \overline{\overline{\overline{\overline{9+87} + \overline{9+86} + \overline{9+85.4.3} + \overline{9+85.4.2}}}}}$$

$$= \overline{\overline{9.8.7} + \overline{9.8.6} + \overline{9.8.5.4.3} + \overline{9.8.5.4.2}}$$

$$A = \overline{\overline{\overline{\overline{\overline{9+9+87} + \overline{9+865} + \overline{9+86.4.3} + \overline{9+86.4.2}}}}}}}$$

$$= \overline{\overline{9+9.8.7} + \overline{9.8.6.5} + \overline{9.8.6.4.3} + \overline{9.8.6.4.2.1}}$$

$$= \overline{\overline{9+8.7} + \overline{8.6.5} + \overline{8.6.4.3} + \overline{8.6.4.2.1}}$$

4.2.6. Một số mã thông dụng

1) Mã nhị – thập phân thông dụng

Xem bảng 4-2-5

Bảng 4-2-5 : MÃ NHỊ – THẬP PHÂN THÔNG DỤNG

Loại Số thập phân	8421	Dư 3	2421 (A)	2421 (B)	5211	Vòng dư 3	Dịch phải
0	0000	0011	0000	0000	0000	0010	00000
1	0001	0100	0001	0001	0001	0110	10000
2	0010	0101	0010	0010	0100	0111	11000
3	0011	0110	0011	0011	0101	0101	11100
4	0100	0111	0100	0100	0111	0100	11110
5	0101	1000	0101	1011	1000	1100	11111
6	0110	1001	0110	1100	1001	1101	01111
7	0111	1010	0111	1101	1100	1111	00111
8	1000	1011	1110	1110	1101	1110	00011
9	1001	1100	1111	1111	1111	1010	00001
Tổng số	8421		2421	2421	5211		

Cần lưu ý rằng, trong bảng 4.2.5, tuy mỗi vị trí (bit) chỉ có hai trạng thái 0 và 1, nhưng không nhất thiết phải có quan hệ về giá trị chuyển vị của số nhị phân (như 8421), mà nói chung thì 4 bit làm thành một từ mã để trở thành một kí hiệu hàm ý xác định nào đó. Cũng có thể nói chính xác hơn, bảng 4.2.5 giới thiệu các loại mã 4 bit hai trạng thái.

a) Mã 8421

Trong từ mã 8421, mỗi bit 1 đại biểu cho một trị số cố định. Nếu cộng tất cả trị số của các bit đại biểu trong từ mã, thì ta được trị số tương ứng của số thập phân mà từ mã đó đại biểu. Trong mã 8421 thì trọng số của mỗi bit là cố định, nên nó thuộc loại mã có trọng số.

b) Mã dư 3

Mã dư 3 được tạo thành từ mã nhị phân bằng cách cộng thêm 0011 (giá trị số 3 thập phân) vào từ mã nhị phân tương ứng.

c) Mã 2421

Hai loại mã 2421 (A và B) đều là mã có trọng số, nhưng sự tuần tự các bit của các từ mã khác số nhị phân (trọng số không đơn trị).

d) Mã 5211

Mã 5211 có tính chất tương tự mã 2421, đó là mã có trọng số không đơn trị.

e) mã vòng dư 3 (mã Gray dư 3)

Mã vòng (mã Gray) sẽ được giới thiệu ở bảng 4.2.6. mã vòng dư 3 không bắt đầu từ 0000 như mã vòng, mà lệch đi 3 hàng. (Từ mã 0010 là mã hoá số 3 của mã vòng, lại là mã hoá số 0 của mã dư 3. Từ mã 0110 là mã hoá số 4 của mã vòng, lại là mã hoá số 1 của mã dư 3 v.v...) Mã vòng dư 3 không phải là mã có trọng số.

g) Mã dịch phải

Mã dịch phải có đặc điểm là hai từ mã kế cận bất kì chỉ khác nhau 1 bit (trên bảng 4-2-5, ta thấy sự dịch sang phải của các bit) mã dịch phải không tận dụng tất cả các trạng thái của mã nhị phân 5 bit. Đây là nhược điểm loại mã này (phải thêm 1 bit so với các loại mã khác).

2) Mã Gray

Mã gray còn gọi là mã vòng, bảng 4-2-6 giới thiệu bảng mã gray 4 bit.

Bảng 4-2-6 : MÃ GRAY 4 BIT

Số thập phân	Mã Gray	Số thập phân	Mã Gray
0	0 0 0 0	8	1 1 0 0
1	0 0 0 1	9	1 1 0 1
2	0 0 1 1	10	1 1 1 1
3	0 0 1 0	11	1 1 1 0
4	0 1 1 0	12	1 0 1 0
5	0 1 1 1	13	1 0 1 1
6	0 1 0 1	14	1 0 0 1
7	0 1 0 0	15	1 0 0 0

Ta có thể thấy rõ đặc điểm mã Gray từ bảng 4-2-6 : sự biến đổi giá trị bit ở một vị trí xác định trong từ mã có tính chất tuần hoàn với chu kì cố định từ từ – mã này sang từ – mã khác. Chu kì của bit thứ nhất là 0110, chu kì của bit thứ hai là 00111100, chu kì của bit thứ ba là 0000111111110000 v.v...

Mã Gray không có trọng số. Ưu điểm chính của mã Gray là chỉ có 1 bit khác nhau trong 2 từ mã kế cận. Nhược điểm là thiếu trực quan.

3) Mã ISO (*International Standardization Organization*)

Cơ quan tiêu chuẩn hoá quốc tế ISO đưa ra mã nhị phân 8 bit, chủ yếu dùng để truyền tin. Mã này có 10 chữ số ($0 \div 9$), 26 chữ cái Anh, và 20 dấu, kí hiệu, tất cả là 56 kí tự. Xem bảng 4.2.7. Trong bảng, các bit được biểu thị bằng $b_7, b_6, \dots, b_1$. Bit thứ 8 là bit bù để biến từ mã bất kì thành từ mã có số bit 1 là số chẵn, mục đích phát hiện lỗi truyền tin.

Bảng 4-2-7 : MÃ ISO

$b_7b_6b_5$ Kí tự $b_4b_3b_2b_1$								
	000	001	010	011	100	101	110	111
0000	NUL		SP	0		P		
0001				1	A	Q		

0010				2	B	R		
0011				3	C	S		
0100			\$	4	D	T		
0101			%	5	E	U		
0110				6	F	V		
0111				7	G	U'		
1000	BS			8	H	X		
1001	HT	EM	)	9	I	Y		
1010	LF		*	:	J	Z		
1011			+		K			
1100			,		L			
1101	CR		-	=	M			
1110								4
1111			/		O			DEL

Đặc điểm các kí tự trong bảng 4-2-7 như sau : các chữ số đều có chung 3 bit $b_7b_6b_5 = 011$, các chữ cái có chung 2 bit $b_7b_6 = 10$; Các chữ số được chuyển thành mã ISO theo kiểu mã nhị phân của $b_4b_3b_2b_1$; thực tự các chữ cái A ÷ Z (từ 1 đến 26) cũng là mã nhị phân của $b_5b_4b_3b_2b_1$.

4) Mã ASCII (American National Standard Code for Information Interchange)

Bảng 4-2-8 : MÃ ASCII

$b_7b_6b_5$ Kí tự $b_4b_3b_2b_1$	000	001	010	011	100	101	110	111
	0000	0001	0010	0011	0100	0101	0110	0111
	NUL	DLE	SP	0	@	P	\	p
	SOH	DC1	!	1	A	Q	a	q
	STX	DC2	“	2	B	R	b	r

0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	\$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	,	7	G	U'	g	ư
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	?
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	↑	n	~
1111	SI	US	/	?	O	↓	o	DEL

Mã ASCII, viết tắt của cắct có nghĩa là mã chuẩn quốc gia Mĩ để trao đổi tin tức. Xem bảng 4.2.8. Trong đó. Mã các chữ số và chữ cái giống như mã ISO trên đây. Mã ASCII thường dùng cho thiết bị thông tin và máy tính. ASCII là mã 8 bit nhị phân, với 7 bit $b_1 \div b_7$ biểu thị các tin tức và bit thứ 8 là bit parity (kiểm tra chẵn lẻ dùng để phát hiện lỗi truyền tin). Bảng 4.2.9 giải thích các kí hiệu chữ.

Bảng 4.2.9 : Ý NGHĨA CÁC KÍ HIỆU CHỮ CỦA MÃ ASCII

Kí hiệu	Ý nghĩa	Kí hiệu	Ý nghĩa
NUL	Số không, không, vô hiệu	BS	Lùi một khoản kí tự
SOH	Bắt đầu của tiêu đề	HT	Kê bảng hướng ngang
STX	Bắt đầu của hành văn	LF	Chuyển dòng
ETX	Kết thúc của hành văn	VT	Kê bảng hướng dọc
EOT	Kết thúc truyền tin	FF	Điều khiển chạy giấy
ENQ	Hỏi	CR	Quay về đầu dòng

ACK	Thừa nhận	SO	Dịch ra (Shift out)
BEL	Chuông	SI	Dịch vào (Shift in)
DLE	Chuyển mã (Datalink escape)	EM	Hết giấy
DC1	Điều khiển thiết bị 1	SUB	Trừ
DC2	Điều khiển thiết bị 2	ESC	Chuyển mã
DC3	Điều khiển thiết bị 3	FS	Dấu phân cách (File separator)
DC4	Điều khiển thiết bị 4	GS	Dấu phân cách gói (Group separator)
NAK	Phủ định	RS	Dấu phân cách ghi (Record separator)
SYN	Đồng bộ	US	Dấu phân cách đơn vị (Unit separator)
ETB	Kết thúc truyền gói tin	SP	Khoảng trống kí tự
CAN	Hủy bỏ	DEL	Hủy bỏ

4.3. BỘ GIẢI MÃ

Khi mã hoá, mỗi từ mã nhị phân đều 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 phiên dịch hàm ý đã gán cho từ mã. Mạch điện thực hiện việc giải mã được gọi là bộ giải mã. Vậy bộ giải mã phiên dịch từ mã thành tín hiệu đầu ra, biểu thị tin tức vốn có. Tùy theo yêu cầu sử dụng tin tức được giải mã, tín hiệu đầu ra có thể là xung hay mức điện áp. Có nhiều loại bộ giải mã, nhưng chúng ta đều có nguyên lý công tác và phương pháp thiết kế tương tự nhau. Dưới đây ta sẽ thuyết minh cụ thể qua vài ví dụ.

4.3.1. Bộ giải mã nhị phân

Bộ giải mã nhị phân phiên dịch các từ mã nhị phân thành tín hiệu đầu ra.

Ví dụ 4.3.1



Hình 4.3-1 Sơ đồ mạch vận caàn

Hãy thiết kế bộ giải mã nhị phân 3 bit.

Bài giải :

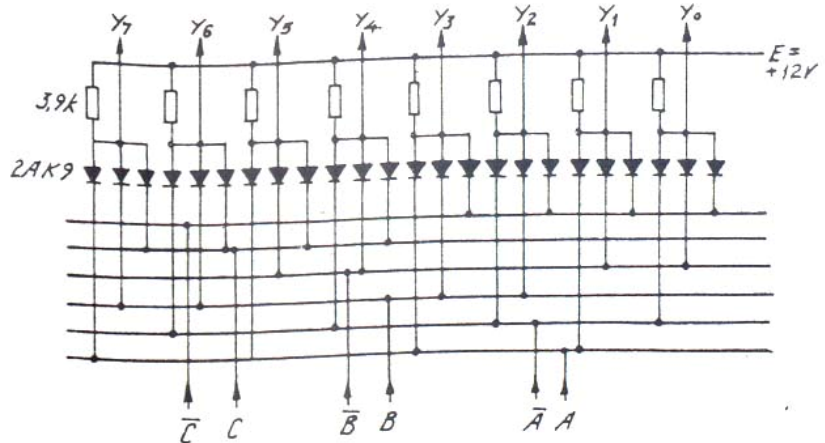
Bảng 4-3-1 : BẢNG CHÂN LÍ CỦA BỘ GIẢI MÃ

C	B	A	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

1. Phân tích yêu cầu. Đầu vào là nhóm từ mã nhị phân 3 bit. Đầu ra là 8 tín hiệu tương ứng các từ mã.

2. Kê bảng chân lí. Xem bảng 4-3-1. Cần lưu ý rằng $Y_0 \div Y_7$ là 8 tín hiệu độc lập nhau tương ứng với 8 từ mã (Mỗi Y_i là một hàm của các biến vào, bảng chân lí của nó là một cột tương ứng bên phải với 3 cột biến số bên trái. Tuy nhiên để tiện lợi, chúng ta gộp 8 bảng chân lí vào làm một).

3. Tối thiểu hoá.
Căn cứ vào bảng 4-3-1, ta có :



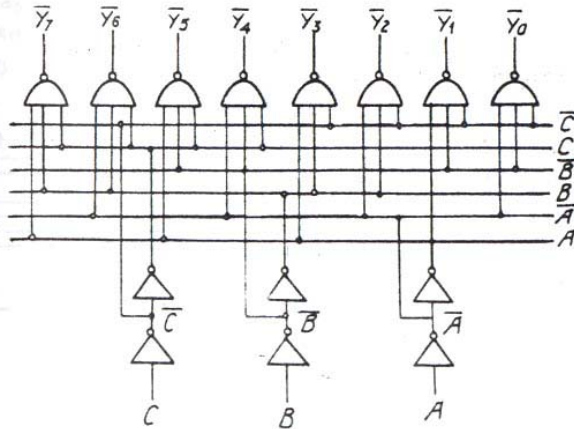
Hình 4-3-2 Bộ giải mã cổng AND dùng diốt

$$\begin{aligned} Y_0 &= \overline{C}\overline{B}\overline{A} & Y_1 &= \overline{C}\overline{B}A & Y_2 &= \overline{C}B\overline{A} & Y_3 &= \overline{C}BA \\ Y_4 &= C\overline{B}\overline{A} & Y_5 &= C\overline{B}A & Y_6 &= CB\overline{A} & Y_7 &= CBA \end{aligned}$$

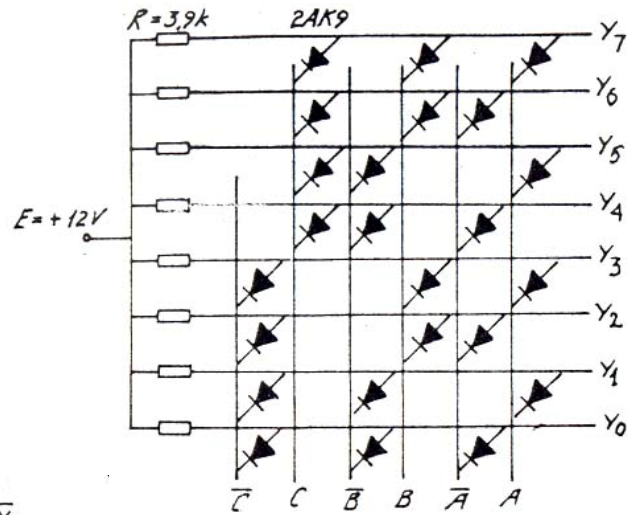
4. Vẽ sơ đồ logic

Các hàm logic trên đây có thể dùng cổng AND điôt thực hiện (hình 4-3-2, hình 4-3-3), hay cũng có thể dùng cổng NAND (TTL) thực hiện (hình 4-3-4).

Trong bộ giải mã nhị phân, nếu từ mã đầu vào có n bit thì sẽ có 2^n tín hiệu đầu ra tương ứng với mỗi từ mã. Bộ giải mã này có khi còn được gọi là bộ giải mã biến số, vì toàn bộ các từ mã của biến số đầu vào đều được giải mã.



Hình 4-3-4 Bộ giải mã dùng NAND

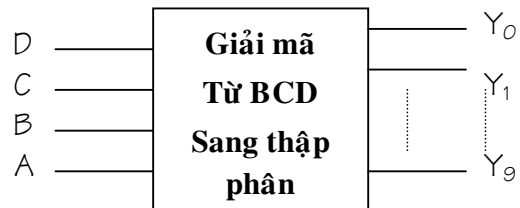


Hình 4-3-4 Bộ giải mã dùng NAND

4.3.2. Bộ giải mã (BCD) – thập phân

Bộ giải mã thực hiện chuyển đổi từ mã BCD thành 10 tín hiệu đầu ra tương ứng 10 chữ số của hệ thập phân (hình 4-3-5).

Xem bảng chức năng 4-3-2, trong đó



Hình 4-3-5. Sơ đồ mạch yêu cầu.

H là mức cao, L là mức thấp, tín hiệu ra tích cực ở mức logic thấp.

Bảng 4-3-2 : BẢNG CHỨC NĂNG CỦA BỘ GIẢI MÃ TỪ MÃ BCD SANG MÃ THẬP PHÂN

D	C	B	A	Y ₀	Y ₁	Y ₂	Y ₃	Y ₄	Y ₅	Y ₆	Y ₇	Y ₈	Y ₉
L	L	L	L	L	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	H	H	H	H	H	H
L	L	H	L	H	H	L	H	H	H	H	H	H	H
L	L	H	H	H	H	H	L	H	H	H	H	H	H
L	H	L	L	H	H	H	H	L	H	H	H	H	H
L	H	L	H	H	H	H	H	H	L	H	H	H	H
L	H	H	L	H	H	H	H	H	H	L	H	H	H
L	H	H	H	H	H	H	H	H	H	H	L	H	H
H	L	L	L	H	H	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

Bảng 4-3-3 là bảng chân lí dùng logic dương. Trong đó, phía trái là mã BCD 8421 đầu vào ; bên phải là đầu ra giải mã, tích cực ở logic 0, 6 từ mã 6 (6 trạng thái) từ 1010 đến 1111 không được dùng. Chúng không xuất hiện trong tình huống bình thường của bộ giải mã. Các trạng thái này được đánh dấu chéo “x” trong bảng 4-3-3 và trong các bảng Karnaugh (hình 4-3-6).

Tối thiểu hoá hàm logic đầu ra Y_i bằng phương pháp hình vẽ : bảng Karnaugh. Cụ thể, ta tìm hàm đảo Y₁ (tương ứng lấy giá trị 0), sau đó lấy đảo của hàm đảo đó, ta sẽ được hàm Y₁ cần tìm.

Bảng 4-2-4 : BẢNG CHÂN LÍ CỦA BỘ GIẢI MÃ TỪ MÃ BCD SANG MÃ THẬP PHÂN

D	C	B	A	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
			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	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1	1
1	0	0	0	1	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	1	0
1	0	1	0	x	x	x	x	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x	x	x	x	x
1	1	1	1	x	x	x	x	x	x	x	x	x	x	x

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

(a)

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

(b)

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

(c)

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

(d)

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

(e)

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

(f)

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

(g)

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

(h)

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

(i)

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

(j)

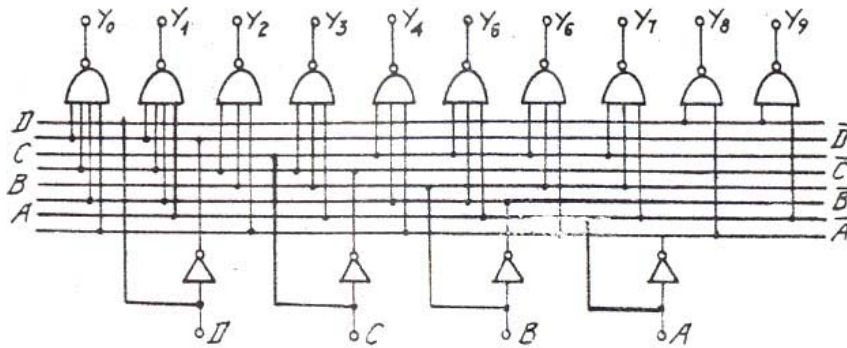
Hình 4-3-6.

Bảng Karnaugh của hàm $Y_9 \div$

Hàm logic tối thiểu dạng NAND của chúng là :

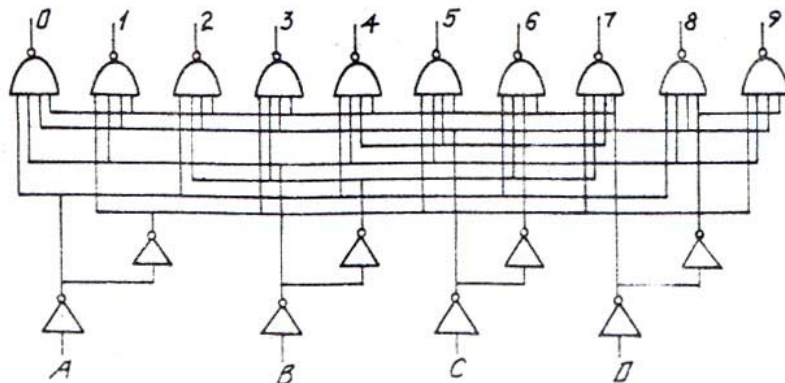
- | | |
|--|---|
| (a) $\overline{Y}_9 = DA$ | (b) $\overline{Y}_8 = D\overline{A}$ |
| $Y_9 = \overline{DA}$ | $Y_8 = \overline{D\overline{A}}$ |
| (c) $\overline{Y}_7 = CBA$ | (d) $\overline{Y}_6 = C\overline{B}\overline{A}$ |
| $Y_7 = \overline{CBA}$ | $Y_6 = \overline{C\overline{B}\overline{A}}$ |
| (e) $\overline{Y}_5 = C\overline{B}A$ | (f) $\overline{Y}_4 = C\overline{B}\overline{A}$ |
| $Y_5 = \overline{C\overline{B}A}$ | $Y_4 = \overline{C\overline{B}\overline{A}}$ |
| (g) $\overline{Y}_3 = \overline{C}BA$ | (h) $\overline{Y}_2 = \overline{C}\overline{B}A$ |
| $Y_3 = \overline{\overline{C}BA}$ | $Y_2 = \overline{\overline{C}\overline{B}A}$ |
| (i) $\overline{Y}_1 = \overline{D}\overline{C}\overline{B}A$ | (j) $\overline{Y}_0 = \overline{D}\overline{C}\overline{B}\overline{A}$ |
| $Y_1 = \overline{\overline{D}\overline{C}\overline{B}A}$ | $Y_0 = \overline{\overline{D}\overline{C}\overline{B}\overline{A}}$ |

Sơ đồ logic hình 4-3-7 bao gồm 10 cổng NAND và 4 cổng đảo



Hình 4-3-7 bao gồm 10 cổng NAND và 4 cổng đảo

Hình 4-3-8 là vi mạch MSI (đơn phiên) thực hiện giải mã từ BCD sang thập phân. IC này có 8 cổng đảo và 10 cổng NAND 4 đầu vào. Sự giải mã của IC là triệt để, hàm logic đầu ra xác định đơn trị. (Cần để ý rằng đối với mạch hình 4-3-7, do sử dụng các giá trị dấu chéo “X” của hàm logic trong tối thiểu hoá để xây dựng sơ đồ mạch, nên nếu đầu vào xuất hiện từ mã không được dùng, thì đầu ra có thể hướng ứng sai).



Hình 4-3-8 Vi mạch MSI giải mã BCD sang thập phân

Nhận xét sơ đồ hình 4-3-8, ta thấy : các cổng đảo là thành mạch đệm và bảo đảm phụ tải tiêu chuẩn đối với các tín hiệu vào A, B, C, D. (Sơ đồ hình 4-3-7 biểu thị phụ tải nặng, không đầu đối với nguồn tín hiệu vào).

4-3-3. Bộ giải mã của hiển thị kí tự

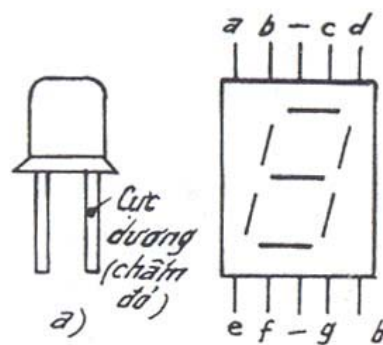
Trong hệ thống số, thường cần giải mã các kí tự đã mã hoá nhị phân thành các tín hiệu có thể hiển thị dưới dạng quen thuộc của con người. Vì phương thức làm việc của các linh kiện hiển thị khác nhau là khác nhau, nên trước hết ta hãy thuyết minh ngắn gọn về một số linh kiện hiển thị kí tự thông dụng.

1) Hai loại hiển thị số

a) Linh kiện hiển thị bán dẫn

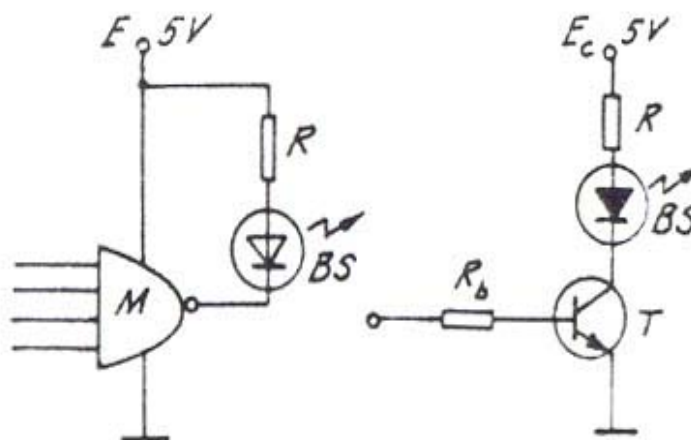
Nguyên lí :

Một số vật liệu bán dẫn đặc biệt như hợp chất GaAsP, khi làm thành chuyển tiếp PN, nếu có điện áp thuận đặt vào, thì có thể bức xạ quang, tức là biến điện năng thành quang năng. Sử dụng các chuyển tiếp PN bức xạ quang có thể chế tạo các linh kiện như điốt phát quang LED, đèn hiển thị 7 thanh hay đèn hiển thị ma trận. Ví dụ : hình 4-3-9, mạch kích sáng.



Hình 4-3-9 Hiển thị bán dẫn
a) LED b) đèn 7 thanh

Hình 4-3-10 giới thiệu mạch điện dùng để kích sáng thực hiện bằng cổng NAND (TTL) hay bằng tranzito : Trong hình, BS là LED (hay 1 thanh trong đèn 7 thanh). Cổng M thông hay T bão hoà sẽ làm BS sáng. R là điện trở hạn dòng. Điện áp công tác của BS cỡ 1,5 ÷ 3V, dòng điện công tác của BS cỡ trên dưới 10mA. Điều chỉnh R có thể làm thay đổi độ sáng của BS.



Hình 4-3-10 Mạch kích sáng

Đặc điểm cơ bản :

Quang phổ phát xạ của hiển thị bán dẫn phù hợp với cảm thụ thị giác, điện áp công tác thấp ($1,5 \div 3V$), thể tích nhỏ, tuổi thọ cao (hơn ngàn giờ làm việc), tốc độ hưởng ứng cao ($1 \div 100ns$), có nhiều màu.

Bảng 4-3-4. Giới thiệu tham số hiển thị số 7 thanh.

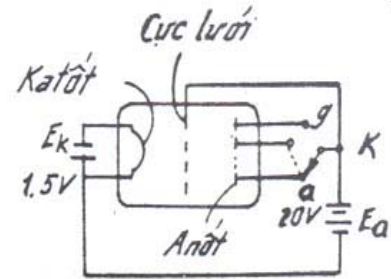
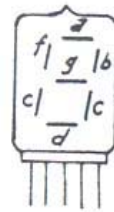
Bảng 4-3-4. THAM SỐ HIỂN THỊ SỐ 7 THANH.

Kí hiệu	Độ sáng (FL)	Điện áp công tác thuận (V)	Dòng điện dòng ngược (μA)	Điện áp ngược đánh thủng (V)	Dòng điện cực đại (mA)
BS201A	> 15 (7 thanh)	$1,5 \div 3V$	< 50	> 5	150 (7 thanh)
ES202A	> 15 (7 thanh)	$1,5 \div 3V$	< 50	> 5	150 (7 thanh)

b) Đèn hiện số 7 thanh chân không

Nguyên lí :

Hình 4-3-11 giới thiệu đèn hiển thị số chân không. Katốt phát xạ nhiệt điện tử, điện trường của cực lưới gia tốc điện tử để chúng đủ động năng đập vào anốt.



Hình 4-3-11 Đèn hiện số 7 thanh chân không

Trên bề mặt anốt có phủ

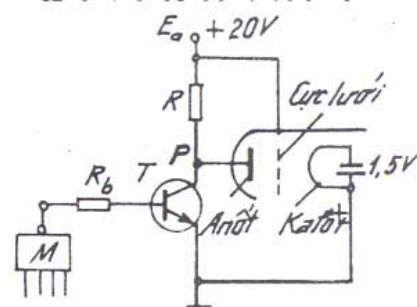
lớp huỳnh quang oxyt kẽm sẽ phát xạ quang màu lục. Các anốt có cấu trúc hiện số 7 thanh.

Chuyển mạch K điều khiển sự cấp điện áp cho anốt ; anốt nào được cấp điện áp thì sẽ phát sáng.

Mạch kích sáng :

Xem hình 4-3-12. R là điện trở hạn dòng giá trị của R xác định theo dòng colectơ cực đại I_{CM} của tranzito T. Khi cổng NANDM (TTL) đưa ra mức cao là T bão hoà, thì

$$I_c \cong \frac{E_a}{R} < I_{CM}$$



Hình 4-3-12 Mạch kích sáng Đèn hiển thị số chân không

Thay đổi R sẽ điều chỉnh độ sáng hiển thị. R_b cũng là điện trở hạn dòng. R_b xác định theo giá trị giới hạn I_{LM} của cổng NAND (TTL). Khi M ngắt, T bão hoà thì

$$I_B = (V_{OH} - V_{BE}) \times \frac{1}{R_b} < I_{LM}$$

Đặc điểm cơ bản :

Về ưu điểm : điện áp công tác tương đối thấp, dòng điệp nhỏ, hiển thị phù hợp thị giác, ổn định và tin cậy, tầm hiển thị khá lớn, tuổi thọ cao.

Về nhược điểm : cần nguồn điện công suất đáng kể cung cấp cho sợi đốt, không thuận tiện cho lắp ráp. Bảng 4-3-5 giới thiệu chỉ tiêu chủ yếu của một số đèn hiện số 7 thanh chân không thông dụng :

Bảng 4-3-5 : CHỈ TIÊU CHỦ YẾU ĐÈN HIỂN SỐ 7 THANH CHÂN KHÔNG

Kí hiệu	Điện áp anốt (V)	Dòng điện anốt (mA)	Điện áp lưới (V)	Dòng điện lưới (mA)	Điện áp sợi đốt (V)	Dòng điện sợi đốt (mA)
YS9-1	20	≤ 2	20	< 2	1,2	28
YS13-3	20	$< 1,5$	20	< 3	1,2	28
YS18-3	20	< 2	20	< 2	1,2	50
YS27-3	20	$\leq 2,5$	20	2,5	1,2	80

2) Bộ giải mã hiển thị

Chúng ta sẽ thuyết minh nguyên lí công tác và quá trình thiết kế của bộ giải mã hiển thị qua ví dụ bộ giải mã kích cho hiển thị 7 thanh LED.

Ví dụ 4-3-2 : hãy thiết kế bộ giải mã hiển thị kích chi hiển thị 7 thanh LED với tín hiệu đầu vào là mã BCD 8421.

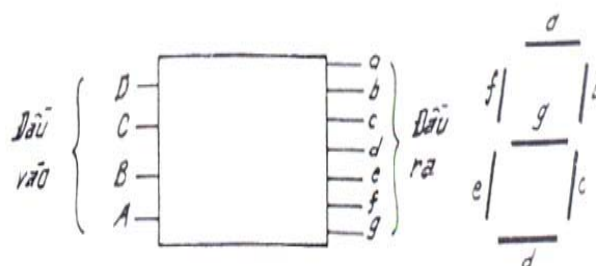
Bài giải :

Phân tích yêu cầu thiết kế :

Xem sơ đồ khối hình 4-3-

13.

Các đầu vào D, C, B, A là mã BCD 8421, trong đó 6 trạng thái 1010 ÷ 1111 không được sử dụng, đánh dấu chéo “X” để xử lí tối thiểu hoá. Tín hiệu đầu ra a, b, ..., g là



Hình 4-3-13 Bộ giải mã 7 thanh

để kích sáng LED tương ứng của hiển thị 7 thanh. Căn cứ mạch kích sáng hình 4-3-10, tín hiệu đầu ra bộ giải mã phải tích cực ở mức thấp (LED sáng khi tín hiệu kích mức thấp). Kê bảng chân lí :

Bảng 4-3-6 : BẢNG CHỨC NĂNG BỘ GIẢI MÃ

D	C	B	A		a	b	c	d	e	f	g	Số được hiển thị
L	L	L	L		L	L	L	L	L	L	H	0
L	L	L	H		H	L	L	H	H	H	H	1
L	L	H	L		L	L	H	L	L	H	L	2
L	L	H	H		L	L	L	L	H	H	L	3
L	H	L	L		H	L	L	H	H	L	L	4
L	H	L	H		L	H	L	L	H	L	L	5
L	H	H	L		L	H	L	L	L	L	L	6
L	H	H	H		L	L	L	H	H	H	H	7
H	L	L	L		L	L	L	L	L	L	L	8
H	L	L	H		L	L	L	L	H	L	L	9

Bảng 4-3-7 : BẢNG CHÂN LÍ BỘ GIẢI MÃ

D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	1	1	1	1
0	0	1	0	0	0	1	0	0	1	0
0	0	1	1	0	0	0	0	1	1	0
0	1	0	0	1	0	0	1	1	0	0
0	1	0	1	0	1	0	0	1	0	0
0	1	1	0	0	1	0	0	0	0	0
0	1	1	1	0	0	0	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	1	0	0

Bảng chức năng 4-3-6 được liệt kê từ kết quả phân tích yêu cầu thiết kế.

Các từ mã đầu vào của mã BCD 8421 quyết định số được hiển thị. Nhưng do cấu trúc không gian của các LED a,b, ..., g (hình 4-3-13) mà các giá trị tín hiệu đầu ra bộ giải mã được xác định sao cho :

- mức thấp L : LED sáng
- mức cao H : LED tắt
- các LED sáng hình thành số được hiển thị.

Bảng 4-3-7 là bảng chân lí tương ứng bảng 4-3-6.

Tối thiểu hoá

Dùng phương pháp hình vẽ. Chúng ta chọn dùng cổng NORAND trong sơ đồ. Do đó, đầu tiên ta tối thiểu hoá hàm đảo bằng dạng ORAND đối với các giá trị 0 của hàm đầu ra, sau đó lấy đảo thì được dạng NORAND đối với các giá trị 1 của hàm đầu ra.

Xem các bảng Karnaugh hình 4-3-14.

Ví dụ : xét LED thanh a.

Tối thiểu hoá dạng chuẩn tắc tuyển đối với các ô trong bảng Karnaugh có giá trị 0 để xác định hàm đảo :

$$\bar{a} = D + B + CA + \bar{C}\bar{A} \quad (\text{dạng ORAND})$$

$$\text{Lấy đảo : } a = \bar{a} = \overline{D + B + CA + \bar{C}\bar{A}} \quad (\text{dạng NORAND})$$

$$(b) \quad \bar{b} = \bar{C} + BA + B\bar{A}$$

$$b = \overline{\bar{C} + BA + B\bar{A}}$$

$$(c) \quad \bar{c} = C + \bar{B} + A$$

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

$$(d) \quad \bar{d} = D + \bar{C}\bar{B} + B\bar{A} + \bar{C}\bar{A} + C\bar{B}A$$

$$d = \overline{D + \bar{C}\bar{B} + B\bar{A} + \bar{C}\bar{A} + C\bar{B}A}$$

$$(e) \quad \bar{e} = \bar{C}\bar{A} + B\bar{A}$$

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

$$(f) \quad \bar{f} = D + \bar{C}\bar{B} + C\bar{A} + B\bar{A}$$

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

$$(g) \quad \bar{g} = D + C\bar{B} + \bar{C}B + B\bar{A}$$

$$g = \overline{D + C\bar{B} + \bar{C}B + B\bar{A}}$$

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

(a)

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

(b)

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

(c)

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

(d)

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

(e)

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

(f)

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

(g)

Hình 4-3-14.

Bảng Karnaugh và hàm logic đầu ra của các LED hiển thị a, b, c, d, e, f, g.

Vẽ sơ đồ logic :

Sơ đồ logic hình 4-3-15 xuất phát từ các hàm logic tối thiểu hoá bằng bảng

Karnaugh hình 4-3-14. Trong hình 4-3-15, tín hiệu ra tích cực ở mức thấp. Ví dụ,

DCBA = 1000, số được hiển thị là 8 với các LED abcdefg = 0000000 sáng.

DCBA = 0101, số được hiển thị là 5 với các LED acdfg sáng và be tắt.

Một

dạng kí

hiệu mạch

của cổng

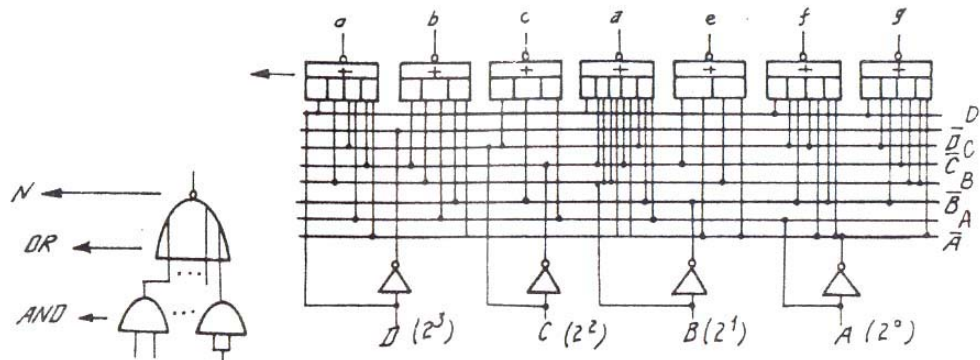
NORAND

(Về cổ

NORAND,

xem phần

3-3-4-3)



Hình 4-3-15 Bộ giải mã kích hiển thị LED 7 thanh

4.4. BỘ SO SÁNH

Trong các hệ thống số, đặc biệt là trong máy tính, thường thực hiện việc so sánh hai số, để biết số nào lớn hơn, hay chúng bằng nhau. 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 công tác theo kiểu nối tiếp hay kiểu song song. Ở chương này, ta sẽ xem xét nguyên lí công tác và quá trình thiết kế bộ so sánh song song qua Ví dụ bộ so sánh số nhị phân 4 bit $A = a_3a_2a_1a_0$ và $B = b_3b_2b_1b_0$

4.4.1. Bộ so sánh bằng nhau

Độ so sánh bằng nhau là mạch điện thực hiện chức năng logic xét hai số có bằng nhau hay không bằng nhau.

1) Bộ so sánh bằng nhau 1 bit

Xét hai bit a_i và b_i , gọi g_i là kết quả so sánh bằng nhau giữa a_i và b_i với $g_i = 1$ biểu thị $a_i = b_i$, $g_i = 0$ biểu thị $a_i \neq b_i$. Vậy ta có bảng chân lí 4-4-1.

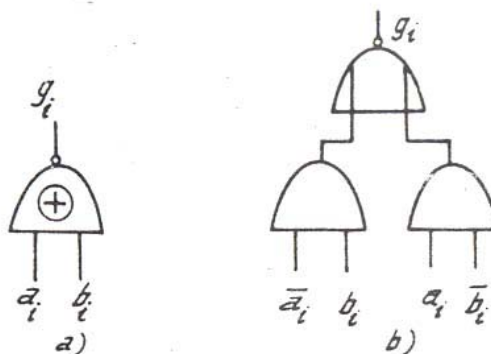
Bảng 4-4-1 : BẢNG CHÂN LÍ CỦA BỘ SO SÁNH 1 BIT

a_i	b_i	g_i	Thuyết minh
0	0	1	Bằng nhau
0	1	0	Khác nhau
1	0	0	Bằng nhau
1	1	1	Khác nhau

Từ bảng 4-4-1, ta có

$$g_i = \overline{a_i} \overline{b_i} + a_i b_i = \overline{a_i \oplus b_i} = \overline{a_i \oplus b_i}$$

Có thể dùng nhiều sơ đồ thực hiện hàm logic trên, hình 4-4-1a là cổng hàm tương đương, hình 4-4-1b là cổng NORAND
đều thực hiện chức năng so sánh bằng nhau.



Hình 4-4-1 Bộ so sánh 1 bit

2) Bộ so sánh bằng nhau 4 bit

Bộ so sánh nhị phân 4 bit $A = a_3a_2a_1a_0$ và $B = b_3b_2b_1b_0$, ta thấy rằng $A = B$ nếu $a_3 = b_3$, $a_2 = b_2$, $a_1 = b_1$, $a_0 = b_0$. Vậy nếu gọi g_i là các bộ so sánh 1 bit, chúng ta xây dựng được bảng chân lí 4-4-2.

Từ bảng 4-4-2, ta có

$$G = g_3 \cdot g_2 \cdot g_1 \cdot g_0$$

Với

$$g_3 = \overline{a_3 \oplus b_3}$$

$$g_2 = \overline{a_2 \oplus b_2}$$

$$g_1 = \overline{a_1 \oplus b_1}$$

$$g_0 = \overline{a_0 \oplus b_0}$$

Bảng 4-4-2 : BẢNG CHÂN LÍ BỘ SO SÁNH BẰNG NHAU 4 BIT

g_3	g_2	g_1	g_0	G
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
1	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

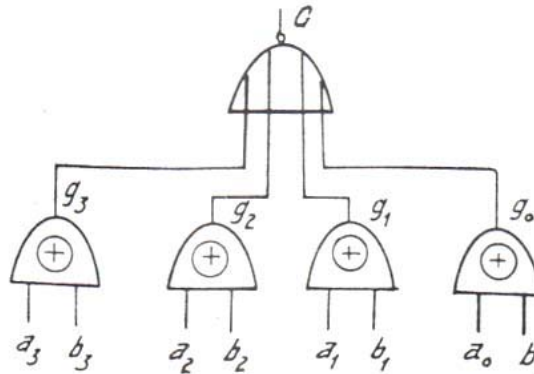
Vậy ta có :

$$G = g_3 \cdot g_2 \cdot g_1 \cdot g_0$$

$$G = a_3 \oplus b_3 a_2 \oplus b_2 a_1 \oplus b_1 a_0 \oplus b_0$$

$$G = (a_3 \oplus b_3) + (a_2 \oplus b_2) + (a_1 \oplus b_1) + (a_0 \oplus b_0)$$

Hình 4-4-2 Sơ đồ logic bộ so sánh bằng nhau 4 bit



4.4.2. Bộ so sánh

Bộ so sánh là mạch điện thực hiện chức năng logic xác định số nào lớn hơn trong hai số được so sánh.

1) Bộ so sánh 1 bit

Xét hai bit a_i và b_i , kết quả so sánh $l_i = 1$ biểu thị $a_i > b_i$, $m_i = 1$ biểu thị $a_i < b_i$.

Ta có bảng chân lí 4-4-3.

Bảng 4-4-1 : BẢNG CHÂN LÍ CỦA BỘ SO SÁNH 1 BIT

a_i	b_i	l_i	g_i	Thuyết minh
0	0	0	0	$a_i = b_i$
0	1	0	1	$a_i < b_i$
1	0	1	0	$a_i > b_i$
1	1	0	0	$a_i = b_i$

Từ bảng 4-4-3, ta có :

$$l_i = a_i \overline{b_i} \quad m_i = \overline{a_i} b_i$$

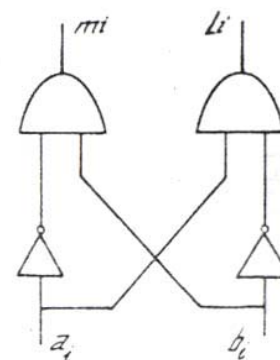
Căn cứ vào hàm logic trên đây, ta xây dựng sơ đồ logic hình 4-4-3.

Hình 4-4-3 Bộ so sánh 1 bit

2) Bộ so sánh 4 bit

a) Phương pháp so sánh hai số nhị phân nhiều

bit



Quá trình so sánh hai số nhị phân nhiều bit phải bắt đầu từ bit có trọng số cao nhất, cứ khi nào bit có trọng số cao nhất bằng nhau thì mới tiếp tục so sánh đến bit có trọng số thấp hơn liền kề. Ý nghĩa trọng số khiến việc so sánh quyết định bởi số có trọng số lớn. (Ví dụ quen thuộc : so sánh 901 và 899, so sánh số hàng trăm $9 > 8$, kết luận luôn $901 > 899$, không để ý đến số hàng chục và hàng đơn vị của chúng).

b) Bộ so sánh 4 bit

So sánh hai số nhị phân $A = a_3a_2a_1a_0$ và $B = b_3b_2b_1b_0$

Kết quả so sánh $G = 1$ nếu $A = B$; $g_i = 1$ nếu $a_i = b_i$

$L = 1$ nếu $A > B$; $l_i = 1$ nếu $a_i > b_i$

$M = 1$ nếu $A < B$; $m_i = 1$ nếu $a_i < b_i$

Bảng 4-4-4 : BẢNG CHÂN LÍ CỦA L

g_3	g_2	g_1	g_0	l_3	l_2	l_1	l_0	L	Thuyết minh
x	x	x	x	1	x	x	X	1	A > B
1	x	x	x	x	1	x	x	1	
1	1	x	x	x	x	1	x	1	
1	1	1	x	x	x	x	1	1	

Bảng 4-4-5 : BẢNG CHÂN LÍ CỦA M

g_3	g_2	g_1	g_0	m_3	m_2	m_1	m_0	M	Thuyết minh
x	x	x	x	1	x	x	x	1	A > B
1	x	x	x	x	1	x	x	1	
1	1	x	x	x	x	1	x	1	
1	1	1	x	x	x	x	1	1	

Khi liệt kê bảng 4-4-4 và 4-4-5, cần lưu ý các quan hệ phủ định nhau của các biến logic và một số giá trị của biến logic không ảnh hưởng gì đến hàm logic. Ví dụ : Xét bảng 4-4-4.

Hàng thứ nhất, $l_3 = 1$ chứng tỏ $a_3 > b_3$, $g_3 \neq 1$ ($l_3 = 1$ phủ định $g_3 = 1$). Vì a_3, b_3 có trọng số lớn nhất, nên $A > B$, $L = 1$ không phụ thuộc gì vào $g_2, g_1, g_0, l_2, l_1, l_0$ nên tương ứng với các dấu chéo “X”.

Hàng thứ hai, $g_3 = 1$ chứng tỏ $a_3 = b_3$, $l_3 \neq 1$. $l_2 = 1$ chứng tỏ $a_2 > b_2$, vì chúng có trọng số thứ hai, nên $A > B$, $L = 1$ không phụ thuộc gì vào $g_2, g_1, g_0, l_2, l_1, l_0$, tương ứng với các dấu chéo “X”.

Hàng thứ ba $g_3 = g_2 = 1$ chứng tỏ $a_3 = b_3$, $a_2 = b_2$, $l_3 \neq 1$, $l_2 \neq 1$, $l_1 = 1$ chứng tỏ $a_1 > b_1$, do đó $A > B$, $L = 1$, không phụ thuộc vào g_1, g_0, l_0 tương ứng với các dấu chéo “X”.

Hàng thứ tư $g_3 = g_2 = g_1 = 1$ ($a_3 = b_3$, $a_2 = b_2$, $l_3 \neq 1$, $l_2 \neq 1$, $l_1 \neq 1$) $l_0 = 1$ chứng tỏ $a_0 > b_0$, do đó $A > B$, $L = 1$, $g_0 \neq 1$. Tương ứng các giá trị $\neq 1$ đều đánh dấu chéo “X”.

Bảng 4-4-5 cũng được tìm hiểu và kê ra một cách tương tự.

Từ bảng 4-4-4 ta có : $L = l_3 + g_3.l_2 + g_3g_2l_1 + g_3g_2g_1l_0$.

Từ bảng 4-4-5 ta có : $M = m_3 + g_3m_2 + g_3g_2m_1 + g_3g_2g_1m_0$.

Sơ đồ logic hàm L, M thực hiện bằng cổng AND và OR trên hình 4-4-4.

g_3, g_2, g_1 là tín hiệu đầu ra.

Bộ so sánh bằng nhau 1 bit

$$g_3 = a_3 \oplus b_3$$

$$g_2 = a_2 \oplus b_2$$

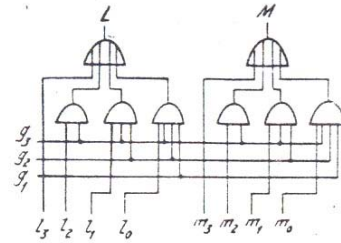
$$g_1 = a_1 \oplus b_1$$

l_3, l_2, l_1, l_0 và m_3, m_2, m_1, m_0 là các bộ

so sánh 1 bit.

$$l_3 = a_3 \overline{b_3}; l_2 = a_2 \overline{b_2}; l_1 = a_1 \overline{b_1}; l_0 = a_0 \overline{b_0}$$

$$m_3 = \overline{a_3} b_3; m_2 = \overline{a_2} b_2; m_1 = \overline{a_1} b_1; m_0 = \overline{a_0} b_0;$$



Hình 4-4-4 Bộ so sánh 4 bit.

4.4.4. ICMISI bộ so sánh 4 bit

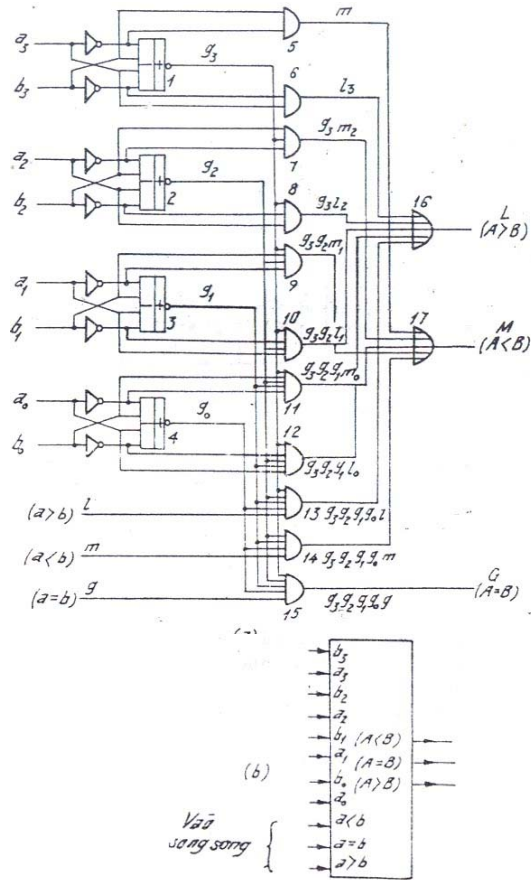
Hình 4-4-5 là vi mạch MSI bộ so sánh 4 bit.

Nhận xét sơ đồ hình 4-4-5, ta thấy rằng MSI này có sơ đồ mạch phù hợp với các sơ đồ bộ so sánh đã trình bày trên, nó có thêm 3 cổng AND và 3 đầu vào điều khiển $a > b$, $a < b$, $a = b$.

Các biểu thức hàm logic đầu ra L, M, G hoàn toàn không khác gì với các biểu thức đã dẫn ở trên. Bảng chân lý của bộ so sánh 4 bit hình 4-4-5 là bảng 4-4-6. Cho $l = m = 0$; $g = 1$, từ sơ đồ logic hình 4-4-5, theo từng cấp từ đầu ra, ta có thể viết các hàm đầu ra như sau :

$$\begin{aligned}
L &= l_3 + g_3 l_2 + g_3 g_2 l_1 + g_3 g_2 g_1 l_0 + g_3 g_2 g_1 l \\
&= l_3 + \overline{g_3 l_2 + g_3 g_2 l_1 + g_3 g_2 g_1 l_0 + 0} \\
&= \overline{a_3 b_3 + a_3 \oplus b_3 a_2 b_2 + a_3 \oplus b_3 a_2 \oplus b_2 a_1 b_1} \\
&\quad + \overline{a_3 \oplus b_3 a_2 \oplus b_2 a_1 \oplus b_1 a_0 b_0} \\
M &= m_3 + g_3 m_2 + g_3 g_2 m_1 + g_3 g_2 g_1 m_0 + g_3 g_2 g_1 m \\
&= m_3 + g_3 m_2 + g_3 g_2 m_1 + g_3 g_2 g_1 m_0 + 0
\end{aligned}$$

Hình 4-4-5 Bộ so sánh 4 bit (MSI)
a) Sơ đồ mạch điện
b) Kí hiệu



$$\begin{aligned}
&= \overline{a_3 b_3 + a_3 \oplus b_3 a_2 b_2 + a_3 \oplus b_3 a_2 \oplus b_2 a_1 b_1} \\
&\quad + \overline{a_3 \oplus b_3 a_2 \oplus b_2 a_1 \oplus b_1 a_0 b_0}
\end{aligned}$$

$$\begin{aligned}
G &= g_3 g_2 g_1 g_0 g \\
&= \overline{g_3 g_2 g_1 g_0} \\
&= \overline{a_3 \oplus b_3 a_2 \oplus b_2 a_1 \oplus b_1 a_0 \oplus b_0}
\end{aligned}$$

Bảng 4-4-6 : BẢN CHÂN LÝ BỘ SO SÁNH 4 BIT

Đầu vào so sánh				Đầu vào điều khiển			Đầu ra		
a_3b_3	a_2b_2	a_1b_1	a_0b_0	l $a > b$	m $a < b$	g $a = b$	L $A > B$	M $A < B$	G $A = B$
$a_3 > b_3$	X	X	X	X	X	X	1	0	0
$a_3 < b_3$	X	X	X	X	X	X	0	1	0
$a_3 = b_3$	$a_2 > b_2$	X	X	X	X	X	1	0	0
$a_3 = b_3$	$a_2 < b_2$	X	X	X	X	X	0	1	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 < b_1$	X	X	X	X	1	0	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	X	X	X	X	0	1	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 > b_0$	X	X	X	1	0	0
$a_3 = b_3$	$a_2 = b_2$	$a_1 = b_1$	$a_0 < b_0$	X	X	X	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	1	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

4.5. BỘ CỘNG

Thực hiện các phép tính là nhiệm vụ cơ bản của máy tính số. Trong máy tính, các phép cộng trừ nhân chia đều quy về phép tính cộng theo một cách nào đó, vậy phép cộng là phép tính cơ bản nhất của máy tính.

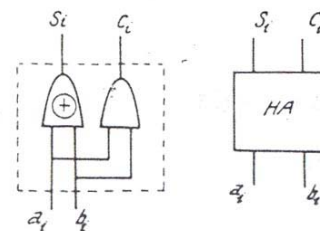
4.5.1. Bộ cộng nửa

Bộ cộng nửa là mạch điện thực hiện phép cộng 2 số 1 bit. Bảng 4-5-1 là bảng chân lý của phép cộng nửa, S_i là tổng a_i và b_i là các số hạng được cộng, c_i là số nhớ sang bit trọng số lớn hơn. Bảng chân lý 4-5-1 chỉ thực hiện phép tính đối với bản thân hai số a_i , b_i mà không kể đến số nhớ chuyển vị từ bit có trọng số bé hơn. Vậy mới có tên là cộng nửa. Từ bảng 4-5-1 ta có :

$$S_i = \overline{a_i}b_i + a_i\overline{b_i} = a_i \oplus b_i$$

$$c_i = a_ib_i$$

Hình 4-5-1 là sơ đồ logic và ký hiệu bộ cộng nửa.



Hình 4-5-1 : Bộ cộng nửa

Bảng 4-5-1 : BẢNG CHÂN LÝ BỘ CỘNG NỬA

a_i	b_i	S_i	C_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

4.5.2. Bộ cộng đủ :

Bộ cộng nửa không đáp ứng yêu cầu thực tế cộng hai số nhiều bit. Ví dụ cộng hai số $A = A_3a_2a_1a_0 = 1011$ và $B = b_3b_2b_1b_0 = 1110$

bit 3	bit 2	bit 1	bit 0	
1	0	1	1	A
1	1	1	0	B
1	1	0		Số nhớ chuyển lên
1	0	0	1	

Qua ví dụ ta thấy : việc cộng các bit 3, bit 2, bit 1 đều kèm theo số nhớ, nghĩa là có 3 số hạng tham gia, gồm 2 bit được cộng và số nhớ chuyển lên từ phép cộng bit trọng số bé liền kề. Phép cộng có nhớ được gọi là cộng đủ, mạch điện thực hiện tương ứng được gọi là bộ cộng đủ (Full Adder).

Bảng 4-5-2 là bảng chân lý của bộ cộng đủ, trong đó a_i , b_i là hai bit được cộng, c_{i-1} là số nhớ chuyển lên từ bit $i - 1$ (bé liền kề) s_i là tổng và c_i là số nhớ (phải chuyển lên phép cộng đủ của bit $i + 1$).

Tiến hành tối thiểu hóa bằng bảng Karnaugh, xem hình 4-5-2

Bảng 4-5-2 : BẢNG CHÂN LÝ CỦA BỘ CỘNG ĐỦ

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

**Hình 4-5-2: Bảng Karnaugh của
hàm logic đầu ra s_i, c_i**

		b_i, c_{i-1}			
a_i	0	00	01	11	10
	1	0	1	0	1
		a) S_i			

		b_i, c_{i-1}			
a_i	0	00	01	11	10
	1	0	0	1	0
		b) C_i			

hình 4-5-2a, ta có :

$$\begin{aligned}
 s_i &= \overline{a_i} \overline{b_i} c_{i-1} + \overline{a_i} b_i \overline{c_{i-1}} + a_i \overline{b_i} \overline{c_{i-1}} + a_i b_i c_{i-1} \\
 &= \overline{a_i} (\overline{b_i} c_{i-1} + b_i \overline{c_{i-1}}) + a_i (\overline{b_i} \overline{c_{i-1}} + b_i c_{i-1}) \\
 &= \overline{a_i} (b_i \oplus c_{i-1}) + a_i \overline{(b_i \oplus c_{i-1})} \\
 &= a_i \oplus b_i \oplus c_{i-1}
 \end{aligned}$$

Từ hình 4-5-2b, ta có :

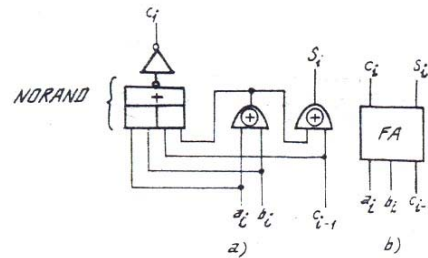
$$\begin{aligned}
 c_i &= \overline{a_i} b_i c_{i-1} + a_i \overline{b_i} c_{i-1} + a_i b_i \\
 &= (\overline{a_i} b_i + a_i \overline{b_i}) c_{i-1} + a_i b_i \\
 &= (a_i \oplus b_i) c_{i-1} + a_i b_i
 \end{aligned}$$

Hàm logic của c_i không ở dạng tối giản mà có $a_i \oplus b_i$ giống như trong hàm s_i , nhờ vậy sơ đồ chung lại đơn giản. Xem hình 4-5-3.

Hình 4-5-3 Bộ cộng đủ.

a) Sơ đồ đủ

b) Kí hiệu

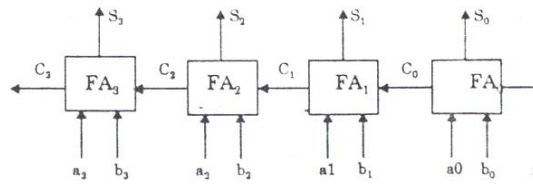


4.5.3. Bộ cộng có nhớ nối tiếp

Sau khi đã giới thiệu bộ cộng đủ 1 bit, bây giờ chúng ta xem xét bộ cộng số nhị phân nhiều bit. Có rất nhiều mạch điện khác nhau thực hiện việc này. Ta hãy xem xét cụ thể bộ cộng hai số nhị phân 4 bit, trong đó số liệu được cộng các bit đưa vào song song, số nhớ chuyển nối tiếp từ bit thấp nhất lên.

Hình 4-5-4 giới thiệu bộ cộng 4 bit được cấu trúc bằng 4 bộ cộng đủ 1 bit. Vì phải đợi số nhớ từ FA bit thấp hơn đưa lên, nên phép FA của bit cao thực hiện sau phép FA của bit thấp. Số nhớ chuyển nối tiếp từ FA bit thấp lên, từng bước một theo số bit.

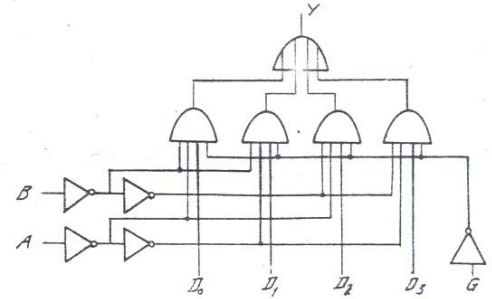
Ưu điểm của bộ cộng các bit song song có nhớ nối tiếp là : mạch điện đơn giản. Nhược điểm của nó là : tốc độ thấp. Để nâng cao tốc độ phép cộng, ta phải tìm cách rút ngắn thời gian thực hiện FA của bit cao, trong đó có cách số nhớ chuyển sớm tới FA của bit cao.



Hình 4-5-4 bộ cộng 4 bit.

4.6. BỘ CHỌN KÊNH

Bộ chọn kênh còn gọi là bộ dồn kênh (MUX) cũng gọi là bộ chọn dữ liệu (DATA SELECTOR). Bộ chọn kênh thực hiện dưới dạng một mạng các cổng NORAND (hay ORAND). Ứng dụng của nó rất rộng. Chức năng logic cơ bản của bộ chọn kênh là dưới sự điều khiển của tín hiệu chọn (n đầu vào điều khiển) thực hiện sự chọn ra một kênh (trong số 2^n kênh đầu vào) để thông tín hiệu kênh được chọn đến đầu ra (1 đầu ra).



Hình 4-6-1 : Bộ chọn kênh (MUX)

Hình 4-6-1 giới thiệu bộ chọn kênh có 4 đầu vào để chọn.

Trong hình, B và A là $n = 2$ đầu vào điều khiển, D_3, D_2, D_1, D_0 là $2^n = 4$, đầu vào dữ liệu được chọn, Y là đầu ra, còn G là đầu chọn chip (cho phép bộ chọn kênh làm việc).

Từ sơ đồ hình 4-6-1 ta viết hàm logic đầu ra :

$$\begin{aligned} Y &= \overline{G} \overline{B} \overline{A} D_0 + \overline{G} \overline{B} A D_1 + \overline{G} B \overline{A} D_2 + \overline{G} B A D_3 \\ &= \overline{G} (\overline{B} \overline{A} D_0 + \overline{B} A D_1 + B \overline{A} D_2 + B A D_3) \end{aligned} \quad (4-6-1)$$

6-1)

Khi $G = 1$ thì $Y = 0$, tức là bộ chọn kênh bị cấm, nó không làm việc.

$$\text{Khi } G = 0 \text{ thì } Y = \overline{B} \overline{A} D_0 + \overline{B} A D_1 + B \overline{A} D_2 + B A D_3 \quad (4-6-2)$$

6-2)

Kênh nào được chọn, phụ thuộc tín hiệu chọn (đầu vào điều khiển).

Nếu $BA = 00$, thì $Y = D_0$, nếu $BA = 01$, thì $Y = D_1$, nếu $BA = 10$ thì $Y = D_2$, nếu $BA = 11$ thì $Y = D_3$. Bảng 4-6-1 là bảng chức năng của bộ chọn kênh.

Bảng 4-6-1 : BẢNG CHỨC NĂNG BỘ CHỌN KÊNH

B	A	D ₀	D ₁	D ₂	D ₃	G	H
X	X	X	X	X	X	H	L
L	L	L	X	X	X	L	L
L	L	H	X	X	X	L	H
L	H	X	L	X	X	L	L
L	H	X	H	X	X	L	H
H	L	X	X	L	X	L	L
H	L	X	X	H	X	L	H
H	H	X	X	X	L	L	L
H	H	X	X	X	H	L	H

Có thể viết rút gọn bảng 4-6-1 thành dạng đơn giản như bảng 4-6-2

Bảng 4-6-2 : BẢNG CHỨC NĂNG RÚT GỌN CỦA BỘ CHỌN KÊNH

B	A	G	Y
X	X	H	L
L	L	L	D ₀
L	H	L	D ₁
H	L	L	D ₂
H	H	L	D ₃

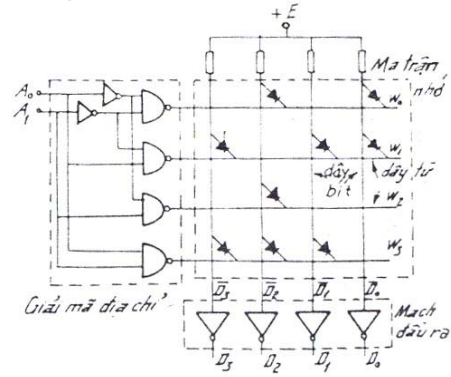
4.7. ROM (bộ nhớ chỉ đọc - Read Only Memory)

ROM còn gọi là bộ nhớ cố định. Vì dữ liệu lưu giữ trong ROM không dễ gì thay đổi, dữ liệu nói chung không thay đổi, chỉ đọc ra. Phần tử nhớ trong ROM rất đơn giản, do đó IC có độ tích hợp cao. Có nhiều chủng loại ROM. Căn cứ vào phần tử nhớ trong ROM, thì có 3 loại : ROM điốt, ROM tranzito lưỡng cực và ROM tranzito trường (MOS). Căn cứ cách viết dữ liệu vào ROM, cũng có 3 loại : ROM cố định (ROM mặt nạ - Maskable ROM), PROM và EPROM. ROM cố định khi chế tạo thì nhà máy nạp sẵn dữ liệu thông qua công đoạn mặt nạ, nội dung đó không thể thay đổi trong quá trình sử dụng. PROM (ROM có thể nạp chương trình - programable ROM) thì người sử dụng có thể nạp chương trình vào (bằng một thiết bị đặc biệt), nhưng dữ liệu chỉ được viết vào PROM một lần mà thôi, sau đó không thể thay đổi. Nội dung lưu giữ trong EPROM (erasable ROM) thì có thể thay đổi (viết mới) bằng một thiết bị đặc biệt. Nhưng do quá trình viết mới phức tạp, nên thông thường chỉ tiến hành đọc.

4.7.1. Bộ nhớ cố định chỉ đọc (ROM)

ROM cố định có 3 phần mạch : bộ giải mã địa chỉ, ma trận phần tử nhớ, và mạch điện đầu ra. Hình 4-7-1 giới thiệu mạch điện đơn giản nhất là ROM điốt.

Hình 4-7-1 ROM điốt.



Dùng phương pháp phân tích logic, ta viết được :

Hàm logic tín hiệu trên các dây từ :

$$W_3 = \overline{A_1 A_0} \quad W_2 = \overline{A_1} \overline{A_0} \quad W_1 = \overline{A_1} A_0 \quad W_0 = A_1 \overline{A_0}$$

Hàm logic tín hiệu đầu ra :

$$D_3 = \overline{W_3 W_1} = \overline{\overline{A_1 A_0} \cdot \overline{A_1} A_0} = A_1 A_0 + \overline{A_1} A_0 = A_0$$

$$D_2 = \overline{W_3 W_2 W_0} = \overline{\overline{A_1 A_0} \cdot \overline{A_1} \overline{A_0} \cdot A_1 \overline{A_0}} = A_1 A_0 + A_1 \overline{A_0} = \overline{A_1 A_0} = A_1 + \overline{A_0}$$

$$D_1 = \overline{W_3 W_1} = A_0$$

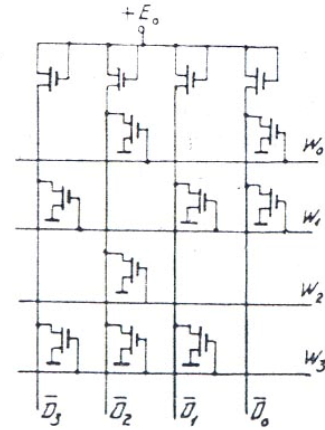
$$D_0 = \overline{W_1 W_0} = \overline{\overline{A_1} A_0 \cdot A_1 \overline{A_0}} = \overline{A_1 A_0} = \overline{A_1} \overline{A_0} = \overline{A_1}$$

Bảng 4-7-1 : BẢNG CHÂN LÝ CỦA ROM ĐIỐT

A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	1	0	1
0	1	1	0	1	1
1	0	0	1	0	0
1	1	1	1	1	0

Từ bảng chân lý, ta thấy sơ đồ hình 4-7-1 thực chất là một bộ chuyển mã từ mã $A_1 A_0$ sang mã $D_3 D_2 D_1 D_0$. Quá trình diễn ra như sau : bộ giải mã địa chỉ tiến hành giải mã $A_1 A_0$, ma trận 4 cổng AND điốt (đầu ra của AND điốt là dây bit, đầu vào của AND điốt là dây từ qua điốt nối dây bit tương ứng) và bộ đảo (mạch điện đầu ra) thực hiện mã hóa đối với tín hiệu đầu ra bộ giải mã. Cuối cùng ta có mã $D_3 D_2 D_1 D_0$.

Trong mạch điện hình 4-7-1, các từ mã của tín hiệu đầu vào và đầu ra có quan hệ đơn trị. Vì vậy, người ta thường xem từ mã đầu vào A_1A_0 là địa chỉ của từ mã $D_3D_2D_1D_0$ được lưu giữ trong ROM, còn bản thân $D_3D_2D_1D_0$ được xem là nội dung dữ liệu được lưu giữ ở địa chỉ tương ứng A_1A_0 . Trong ma trận phần tử nhớ, đường ngang để chọn từ gọi tắt là dây từ, đường dọc để chọn bit gọi tắt là dây bit. Chỗ giao nhau giữa dây từ dây bit được gọi là phần tử nhớ. Tại phần tử nhớ, nếu có điốt thì dữ liệu lưu giữ là 1, nếu không có điốt thì dữ liệu lưu giữ là 0. Nội dung dữ liệu lưu giữ trong ma trận nhớ không thể thay đổi sau khi hoàn thành chế tạo ra ROM, tức là chỉ đọc mà không viết được ...



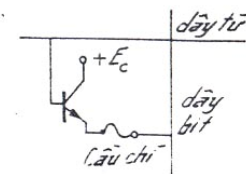
Hình 4-7-2 :
Ma trận nhớ
của ROM NMOS

Hình 4-7-2 là ma trận phần tử nhớ dùng tranzito trường MOS (có nội dung nhớ như mạch hình 4-7-1). Tại phần tử nhớ (nơi giao nhau giữa dây từ và dây bit) nếu có tranzito MOS thì nội dung lưu giữ là 1, nếu không có thì là 0. Chỉ lưu ý tín hiệu dây từ tích cực ở mức cao. (Cũng có thể dùng tranzito lưỡng cực để cấu trúc ma trận nhớ của ROM).

Khi chế tạo ma trận nhớ của ROM, căn cứ nội dung cần lưu giữ mà thiết kế mặt nạ. Điều này chỉ có lợi khi số lượng sản xuất lớn (vì thế tác mặt nạ và các công đoạn liên quan tới thời gian, giá thành cao). Trong trường hợp số lượng sản xuất không lớn, dùng PROM kinh tế hơn.

4.7.2. Bộ nhớ chỉ đọc có thể ghi hình tự (PROM)

Khi xuất xưởng, PROM có các phần tử nhớ đều là 1, người sử dụng căn cứ nội dung cần lưu trữ mà tùy ý chọn phần tử nhớ nào không thay đổi (1), phần tử nhớ nào phải thay đổi (0) để thực hiện thao tác viết vào bộ nhớ. Nhưng chỉ một lần viết mà thôi.



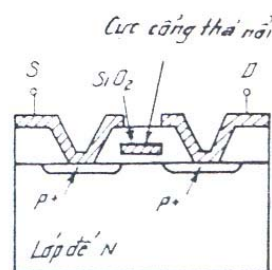
Hình 4-7-3 :
Phần tử nhớ PROM

Hình 4-7-3 giới thiệu một phần tử nhớ PROM, gồm có một tranzito lưỡng cực và một cầu chì (hợp kim Ni, Cr ...) Khi xuất xưởng, cầu chì của các phần tử nhớ đều thông. Người sử dụng muốn ghi bit 1 vào phần tử nhớ nào thì giữ nguyên cầu chì, muốn ghi bit 0 vào phần tử nhớ nào thì làm cầu chì đứt bằng dòng điện đủ lớn theo quy định. Cầu chì của phần tử nhớ đứt rồi thì không có cách gì nối lại như cũ được, tức là không thay đổi nội dung được nữa.

Ngoài hình thức cầu chì, người ta còn chế tạo PROM dùng điốt Schottky thay thế phương án cầu chì trên. Khi xuất xưởng, tất cả các điốt Schottky đều ngắt (ở trạng thái phân cực ngược), tương ứng bit 0. Để tạo ra bit 1, người sử dụng phải đặt điện áp ngược đủ lớn để điốt Schottky đánh thủng tạo thành chập cực thông mạch vĩnh viễn.

4.7.3. Bộ nhớ chỉ đọc có thể viết lại (EPROM)

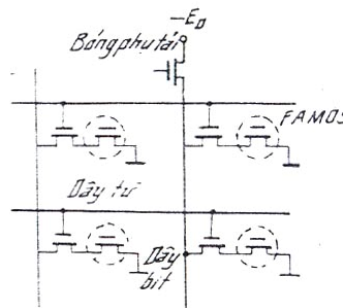
Thực tiễn luôn có nhu cầu sửa chữa, đổi mới một số dữ liệu nào đó trong ROM, do vậy người ta không thỏa mãn với ROM cố định và PROM. EPROM giải quyết và đáp ứng nhu cầu trên, nên có ứng dụng rộng rãi. Hiện nay EPROM sử dụng phần tử nhớ trên cơ sở tranzito FAMOS (Floating - gate Avalanche - injection Metal Oxide



Hình 4-7-4.
Kết cấu tranzito FAMOS

Semiconductor - Bán dẫn oxyt kim loại phun - cực cổng thác lũ thả nổi). Hình 4-7-4 trình bày kết cấu tranzito FAMOS.

Nó cơ bản là tranzito MOS cực cổng S_i kênh P. Điểm khác biệt của nó là cực cổng hoàn toàn bị SiO_2 cách li, nên ở trạng thái “bồng bênh” và có tên cực cổng thả nổi. Cực này vốn không mang điện, nên không có kênh dẫn giữa cực nguồn S và cực máng D, tranzito FAMOS ở trạng thái ngắt. Nhưng nếu đặt điện áp tương đối lớn (-30V chẳng hạn) giữa D và S, làm cho chuyển tiếp PN giữa cực máng D và đế bị đánh thủng thác lũ, điện tử trong vùng nghèo kiệt nhờ điện trường mạnh gia tốc lớn bay từ vùng P^+ ra ngoài. Do tốc độ cao, có một phần tử xuyên qua lớp oxyt tương đối mỏng để tới cực cổng thả nổi, chúng được tồn trữ ở điện tích ở cực cổng không có lối thoát đi, nên bảo tồn lâu dài tại cực cổng. (Ở $100^{\circ}C$, một năm suy giảm chưa tới 1%). Điều chỉnh biên độ và thời gian của điện áp giữa DS, thì có thể điều khiển lượng điện tử phun vào. Khi cực cổng đã có đủ nhiều điện tử (điện tích âm) thì có thể tạo ra kênh dẫn giữa D và S làm cho tranzito FAMOS trở thành thông. Nếu dùng tia cực tím hoặc tia X chiếu vào tranzito FAMOS làm trung hòa điện tích ở cực cổng, kênh dẫn biến mất, tranzito FAMOS trở lại trạng thái ngắt. Để tiện tiến hành việc khử bỏ kênh dẫn, khi bao gói, người ta để một cửa sổ thạch anh cho việc chiếu xạ khi cần.



Hình 4-7-5 : Ma trận nhớ EPROM

Hình 4-7-5 giới thiệu EPROM cấu trúc từ tranzito trường, mỗi phần tử nhớ gồm một tranzito MOS nối tiếp với một tranzito FAMOS. Dây từ điều

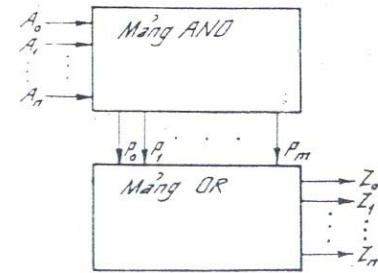
khởi cực cổng. Khi xuất xưởng, tất cả FAMOS đều ở trạng thái hở mạch. Người sử dụng căn cứ vào dữ liệu cần lưu giữ, đưa xung điện áp âm vào dây bit đã chọn, lại điều khiển dây từ cho MOS theo địa chỉ thông dẫn. FAMOS ở dây bit đã chọn sẽ đánh thủng thác lũ, phun điện tử vào cực cổng của nó.

Khi đọc ra, dây từ chọn hàng phần tử nhớ, FAMOS nào đã có điện tử phun vào cực cổng của nó sẽ dẫn điện, dây bit tương ứng sẽ có mức cao ; FAMOS nào không có điện tử phun vào cực cổng thì hở mạch, dây bit tương ứng sẽ có mức thấp.

Cần lưu ý rằng tranzito FAMOS có thể có những hình thức cấu trúc khác với cấu trúc được giới thiệu trên đây.

4.8. MẢNG LOGIC LẬP TRÌNH (PLA)

Hình 4-8-1 giới thiệu PLA cấu trúc từ một mảng cổng AND và một mảng cổng OR. Mảng cổng AND thực hiện phép nhân logic các biến số trong các tín hiệu đầu vào $A_0 \div A_n$. Tín hiệu đầu ra $P_0 \div P_m$ là tích (Hội) của các biến số đầu vào. Mảng cổng OR thực hiện phép cộng logic các tích $P_0 \div P_m$. Tín hiệu đầu ra của mảng OR (cũng là của PLA) là tổng (tuyển) của các tích đó. Tóm lại, ở đầu ra PLA ta có hàm logic dưới dạng chuẩn tắc tuyển. Người sử dụng PLA tùy ý thiết kế tổng của những tích nào, tích của những biến nào. Vậy cấu trúc hình 4-8-1 được gọi là mảng logic lập trình PLA.

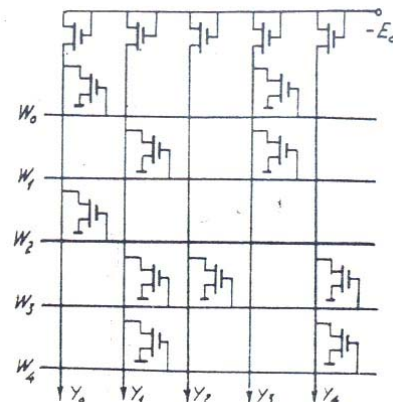


Hình 4-8-1 : Sơ đồ khối cấu trúc của PLA

Mọi người đều biết rằng một hàm logic bất kỳ để có thể biểu thị dưới dạng chuẩn tắc tuyển (ORAND). Vậy dùng PLA thực hiện hàm logic dạng chuẩn tắc tuyển là vô cùng thuận tiện. Vì chúng ta đưa biến logic đến đầu vào PLA, thiết kế các tích bằng mảng AND, sau đó thiết kế các tổng của các tích bằng mảng OR thì đầu ra PLA là hàm logic chuẩn tắc tuyển cần có.

Qua ví dụ tương đối đơn giản dưới đây, chúng ta sẽ thấy cụ thể nguyên lý công tác của PLA.

Hình 4-8-2 trình bày mảng có 5 cổng NAND PMOS. $W_0 + W_4$ là biến số đầu vào $Y_0 + Y_4$ là hàm số đầu ra. Đặc điểm công tác của

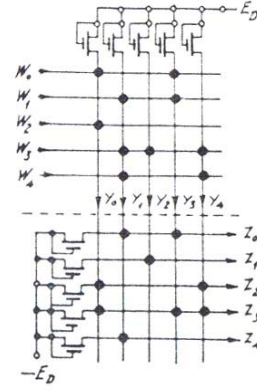


Hình 4-8-2 : Mảng cổng NAND PMOS

tranzito trường PMOS là : Thông dẫn khi điện áp mức thấp ở cực cổng, hở ngắt khi điện áp mức cao ở cực cổng, hở ngắt khi điện áp mức cao ở cực cổng. Vậy ta có bảng chức năng kê ở bảng 4-8-1.

Áp dụng logic dương (mức cao H biểu thị 1, mức thấp L biểu thị 0), ta có thể viết ra các hàm Y_i từ bảng 4-8-1 L:

$$\begin{cases} Y_0 = \overline{W_0 W_2} \\ Y_1 = \overline{W_1 W_3 W_4} \\ Y_2 = \overline{W_3} \\ Y_3 = \overline{W_0 W_1} \\ Y_4 = \overline{W_3 W_4} \end{cases} \quad (4-8-1)$$



Hình 4-8-3 PLA 2 cấp mạng cổng NAND.

Bảng 4-8-1 : BẢNG CHỨC NĂNG

W_2	Y_0	W_1	W_3	W_4	Y_1	W_3	Y_2
L	H	L	L	L	H	L	H
H	H	L	L	H	H	H	L
L	H	L	H	L	H		
H	L	L	H	H	H		
		H	L	L	H		
		H	L	H	H		
		H	H	L	H		
		H	H	H	L		

PLA hình 4-8-3 gồm 2 mạng cổng NAND PMOS. Để đơn giản hình vẽ, nét chấm đậm biểu thị tranzito PMOS tại chỗ giao nhau. Từ hình 4-8-3 ta có :

$$\begin{cases} Z_0 = \overline{Y_1 Y_3} \\ Z_1 = \overline{Y_2} \\ Z_2 = \overline{Y_0 Y_4} \\ Z_3 = \overline{Y_0 Y_3 Y_4} \\ Z_4 = \overline{Y_1} \end{cases} \quad (4-8-2)$$

Thay biểu thức (4-8-1) vào (4-8-2) áp dụng định lý Demoorgan ta có :

$$\begin{cases} Z_0 = W_1 W_3 W_4 + W_0 W_1 \\ Z_1 = W_3 \\ Z_2 = W_0 W_2 + W_3 W_4 \\ Z_3 = W_0 W_2 + W_0 W_1 + W_3 W_4 \\ Z_4 = W_1 W_3 W_4 \end{cases} \quad (4-8-3)$$

Tuy rằng hình 4-8-3 cấu trúc bằng hai mảng cổng NAND, nhưng kết quả của cách đấu nối thể hiện ở hàm đầu ra (4-8-3) chứng tỏ rằng sơ đồ hình 4-8-3 tương đương hình 4-8-1. Dùng sơ đồ cấu trúc hình 4-8-3, đưa tín hiệu đầu vào đến đầu vào mảng trên (W_i), căn cứ tích các biến đầu vào mà thiết kế vị trí tranzito MOS, rồi căn cứ vào tổng các tích mà thiết kế vị trí tranzito MOS trong mảng dưới; hiển nhiên đối với toàn bộ sơ đồ hình 4-8-3 (hai mảng) thì ở đầu ra ta có hàm logic dạng chuẩn tắc tuyển theo yêu cầu thiết kế.

Trong PLA được IC hóa, không chỉ có mảng OR - AND trên đây, mà còn có các bộ đảo cho tín hiệu đầu vào, và các cổng 3 trạng thái đệm ở đầu ra. Vậy trên 1 chip, ta có thể cấu trúc bất cứ mạng tổ hợp nào. Nếu thêm vào chip đó một số Flip Flop (sẽ giới thiệu sau) thì ta có thể cấu trúc được các mạch số nói chung (bao gồm mạch dãy). Cũng như ROM được sản xuất ra thành các loại ROM cố định, PROM, EPROM, PLA cũng được sản xuất ra thành các loại PLA cố định, PLA có thể nạp chương trình và PLA có thể viết mới.

Chúng ta có thể dễ ý thấy rằng kết cấu mạch điện mảng cổng trong PLA giống nhau với ma trận nhớ trong ROM. Chúng khác nhau phương thức công tác. Đầu vào PLA là các biến logic, có thể nhiều biến đồng thời tác động. Đầu vào ma trận nhớ trong ROM là tín hiệu giải mã địa chỉ, một lần chỉ tác động vào một dây từ. Giả sử cần thực hiện cùng một hàm logic phức tạp nào đó, thì phần tử nhớ của PLA ít hơn nhiều so với số phần tử nhớ của ROM tương đương. Vì vậy ứng dụng và sản xuất IC PLA ngày càng rộng rãi.

4.9. NGUY HIỂM CHẠY ĐUA TRONG CÁC MẠCH TỔ HỢP

Nguy hiểm chạy đua là hiện tượng trong mạch tổ hợp xuất ra tín hiệu giả, vốn là những xung nhiễu quá độ khi tín hiệu đầu vào chuyển đổi trạng thái. Nếu phụ tải của mạch xét nhạy cảm đối với xung nhiễu như Flip Flop chẳng hạn, thì cần tìm cách trừ bỏ nguy hiểm chạy đua.

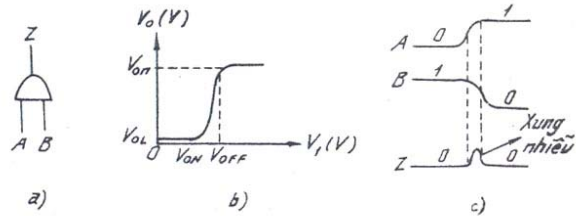
4.9.1. Nguyên nhân sinh ra hiện tượng nguy hiểm chạy đua

Trong mạch số, một cổng bất kỳ mà có hai tín hiệu đầu vào đồng thời chuyển đổi trạng thái theo hướng ngược nhau (ví dụ từ 01 sang 10) thì có thể sinh ra xung nhiễu ở đầu ra. Ta minh họa điều này bằng cổng AND (TTL) hình 4-9-1.

Vì chức năng của cổng AND, $Z = A \cdot B$. Nếu $AB = 01$ hay $AB = 10$ thì đáng lẽ Z phải luôn luôn là 0.

Hình 4-9-1 : Nguy hiểm chạy đua của cổng AND

- a) Cổng AND (TTL);
- b) Đặc tính truyền đạt điện áp của cổng AND;
- c) Xung nhiễu sinh ra do nguy hiểm chạy đua

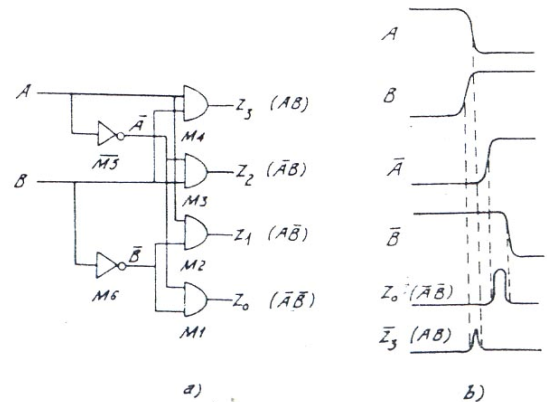


Trong quá trình chuyển đổi từ 01 sang 10 có thể sinh ra xung nhiễu vì :

1. Tín hiệu A, B không thể đột biến, sự chuyển đổi trạng thái điều trải qua thời gian quá độ.

2. Sự chuyển đổi trạng thái của tín hiệu A và B xảy ra có sớm muộn so với nhau (quãng đường truyền tín hiệu khác nhau, trễ truyền đạt của mạch điện không hoàn toàn bằng nhau).

Chẳng hạn tín hiệu A tăng lên mức khóa cổng V_{OFF} trước khi tín hiệu B giảm xuống đến mức mở cổng V_{ON} , khi đó ở đầu ra Z sẽ sinh ra xung nhiễu dương. Còn trong trường hợp tín hiệu B giảm xuống đến V_{ON} trước khi tín hiệu A tăng đến V_{OFF} thì lại không sinh ra xung nhiễu. Vậy mạch điện có nguy hiểm chạy đua không phải nhất định phải có xung nhiễu đầu ra. Tuy nhiên, ta không thể biết trước chính xác sự sai lệch gây ra sớm muộn nói trên. Vậy ý nghĩa hiện tượng nguy hiểm chạy đua là khả năng sinh ra xung nhiễu.



Hình 4-9-2 : Bộ giải mã nhị phân 2 bit.

- a) Sơ đồ logic
- b) Xung nhiễu do nguy hiểm chạy đua sinh ra

Hình 4-9-2 là một ví dụ về xung nhiễu sinh ra do nguy hiểm chạy đua. Giả sử tín hiệu đầu vào chuyển đổi trạng thái theo chiều mũi tên trên bảng 4-9-2.

Theo đồ thị thời gian hình 4-9-2b, tương ứng sai lệch thời gian quá độ tín hiệu A và B có thể sinh ra xung nhiễu đầu ra cổng AND M_4 , tương ứng sai lệch thời gian trễ truyền đạt của M_5 và M_6 có thể sinh ra xung nhiễu đầu ra cổng AND M_1 . Đó là những xung nhiễu sinh ra khi BA chuyển từ 01 sang 10 (đánh dấu trong bảng 4-9-2).

Bảng 4-9-2 : Bảng chân lý mạch điện - (hình 4-9-2)

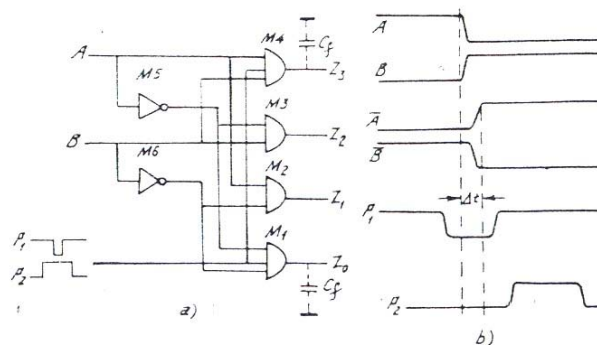
B	A	$\bar{B}$	$\bar{A}$	$\bar{B}A$	$B\bar{A}$	$\bar{B}\bar{A}$	BA
0	0	1	1	1	0	0	0
0	1	1	0	0	1	0	0
1	0	0	1	0	0	1	0
1	1	0	0	0	0	0	1

4.9.2. Phương pháp trừ bỏ nguy hiểm chạy đua

Có nhiều phương pháp để nhận biết một mạch tổ hợp có nguy hiểm chạy đua. Phương pháp trực quan hơn cả là kẻ bảng chân lý cho từng cấp mạch điện xét. Tìm xem cổng nào chịu tác động của tín hiệu đầu vào đồng thời 0 sang 1 và 1 sang 0. Phán đoán khả năng sinh ra xung nhiễu ở đầu ra của toàn mạch xét, tức là có nguy hiểm chạy đua hay không. Nếu phụ tải của mạch xét nhạy cảm với xung nhiễu thì cần tìm cách trừ bỏ nguy hiểm chạy đua. Dưới đây giới thiệu mấy phương pháp thông dụng.

1. Đưa vào xung khóa :

Trên hình 4-9-3 xung âm p_1 là xung khóa đưa vào trong thời gian quá độ để khóa cổng M_1 , M_4 . (Như đã trình bày trên M_1 và M_4 có nguy hiểm chạy đua). Xung p_1 phải đồng bộ với chuyển đổi trạng thái tín hiệu vào và không nhỏ hơn thời gian quá độ Δt .



Hình 4-9-3 : Các phương pháp trừ bỏ nguy hiểm chạy đua

2. Đưa vào xung mở :

Trên hình 4-9-3, xung dương p_2 là xung dùng để mở thông mạch, đưa vào sau thời gian quá độ, lúc mạch đạt đến trạng thái ổn định mới, để mở thông M_1 và M_4 . Tín hiệu đầu ra của M_1 và M_4 bây giờ có dạng xung bề rộng bằng xung mở p_2 . Ví dụ khi tín hiệu đầu ra $BA = 11$, Z_3 không tức thời chuyển lên mức cao, chỉ khi đã xuất hiện p_2 , Z_3 mới hình thành xung dương.

3. Mặc thêm tụ lọc :

Vì xung nhiễu do nguy hiểm chạy đua tạo ra là rất hẹp (hẹp hơn thời gian quá độ Δt) nên có thể mắc thêm một tụ điện lọc (điện dung không lớn) ở đầu ra để trừ bỏ. Đó là C_f trên hình 4-9-3. Đối với mạch TTL, C_f có điện dung vài trăm pF là đủ để suy giảm biên độ xung nhiễu đến mức không đáng kể.

4. Sửa đổi thiết kế logic :

Khi nguy hiểm chạy đua do một biến số chuyển đổi trạng thái gây ra, thì có thể dùng sửa đổi thiết kế logic để trừ bỏ. Xét ví dụ sau :

Giả sử hàm logic đã cho là :

$$Z = AB + \overline{AC}$$

Sơ đồ logic của hàm này là đường nét liền trên hình 4-9-4.

Khi $B = C = 1$ thì :

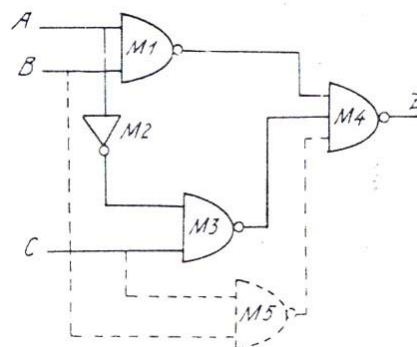
$$Z = AB + \overline{AC} = A.1 + \overline{A}.1$$

$$Z = A + \overline{A}$$

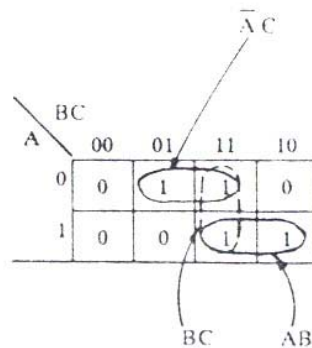
Hàm logic này chứng tỏ xuất hiện nguy hiểm chạy đua khi tín hiệu đầu vào A chuyển đổi trạng thái. Căn cứ công thức 17 chương 3 (3-1-36) ta có :

$$Z = AB + \overline{AC} = AB + \overline{AC} + BC$$

Tương ứng ta thêm cổng M_5 (đường nét đứt) trên sơ đồ hình 4-9-4, khi $B = C = 1$ thì đầu ra M_5 là mức thấp, khóa M_4 , không thể sinh ra xung nhiễu ở đầu ra M_4 nữa.



Hình 4-9-4 : Sửa đổi thiết kế logic để trừ bỏ nguy hiểm chạy đua.



Hình 4-9-5 : Bảng Karnaugh của hàm logic Z .

Hình 4-9-5 là bảng Karnaugh của hàm logic $Z = AB + \overline{AC}$. Ta có thể nhận biết nguy

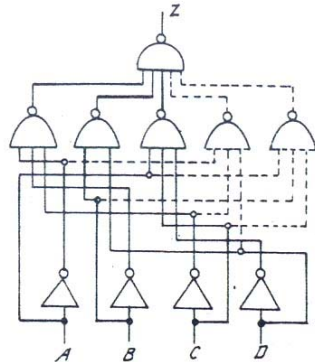
hiểm chạy đua khi xem xét bảng Karnaugh : Hai số hạng $\overline{AC}$ và AB liên kề. (ô $m_3 = \overline{A}BC$ liên kề ô $m_7 = ABC$). Ta cũng dựa vào bảng Karnaugh mà tìm cách trừ bỏ nguy hiểm chạy đua = thêm vào hàm logic một số hạng tương ứng hai ô liên kề m_3 và m_7 , số hạng BC (nét đứt trên hình 4-9-5), vậy $Z = AB + \overline{AC} + BC$.

Xét thêm ví dụ dưới đây :

Ví dụ 4-9-1 : xét xem mạch tổ hợp thực hiện hàm số : $Z = \overline{A}\overline{B}\overline{C} + BD + ACD$ có nguy hiểm chạy đua khi một biến số chuyển đổi trạng thái. Nếu có thì tìm cách trừ bỏ.

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

Hình 4-9-6 : bảng Karnaugh của hàm logic.



Hình 4-9-7 : Sơ đồ logic.

Bài giải :

Vẽ bản Karnaugh của hàm logic $Z = \overline{A}\overline{B}\overline{C} + BD + ACD$ như hình 4-9-6

Hai số hạng $\overline{A}\overline{B}\overline{C}$ và BD liên kề, hai số hạng ACD và BD cũng liên kề. Vậy có nguy hiểm chạy đua. (Khi $\overline{A} = \overline{C} = D = 1$ thì $Z = \overline{B} + B$, khi $A = B = C$ thì $Z = \overline{D} + D$)

Cách trừ bỏ nguy hiểm chạy đua là thêm vào hai số hạng tương ứng các ô liên kề (nét đứt trên hình 4-9-6) $\overline{A}\overline{C}\overline{D}$ và ABC . Hình 4-9-7 là sơ đồ logic của hàm :

$$Z = \overline{A}\overline{B}\overline{C} + BD + ACD + \overline{A}\overline{C}\overline{D} + ABC$$

sơ đồ tuy phức tạp thêm, nhưng đã trừ bỏ nguy hiểm chạy đua.

Chúng ta hãy so sánh 4 phương pháp trừ bỏ nguy hiểm chạy đua đã trình bày trên đây. Hai phương pháp đầu tương đối đơn giản, không làm tăng số linh kiện. Nhưng chúng bị hạn chế ở chỗ phải tìm được xung khóa hay xung mở. Những xung này có yêu cầu chính xác cao về bề rộng và thời gian. Mặc thêm tụ lọc có ưu điểm đơn giản, dễ làm, nhưng làm xấu dạng sóng đầu ra, điều đó là không thể chấp nhận trong một số tình huống nào đó. Phương pháp sửa đổi thiết kế logic khá là lý tưởng khi các IC có sẵn các cổng còn chưa dùng đến trong mạch.

CHƯƠNG 5

MẠCH FLIP - FLOP

5.1. MỘT SỐ CẤU TRÚC MẠCH THƯỜNG GẶP CỦA FLIP - FLOP

Tín hiệu số nhị phân là tín hiệu cơ bản trong mạch số FF là phân tử cơ bản lưu trữ (nhớ) tín hiệu nhị phân. Vì một bit tín hiệu nhị phân có thể nhận một trong hai giá trị 0, 1 nên FF tối thiểu cần có chức năng sau :

1. Có hai trạng thái ổn định, trạng thái 0 và trạng thái 1.
2. Có thể tiếp thu, lưu trữ và đưa ra tín hiệu vào.

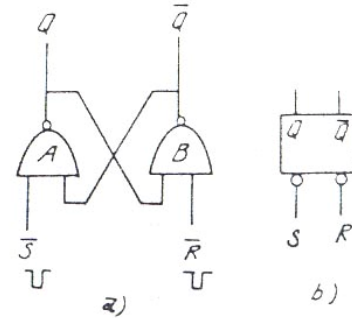
Đương nhiên, thực tiễn còn đề ra các yêu cầu khác. Tiết này trình bày vấn đề: Mạch điện như thế nào đảm bảo các chức năng nói trên.

5.1.1 . Flip Flop RS cơ bản

1. Cấu trúc mạch và ký hiệu

Hình 5-1-1 trình bày sơ đồ logic và ký hiệu logic của Flip Flop RS cơ bản. Mạch điện gồm hai cổng NAND nối ghép chéo. $\bar{R}, \bar{S}$ là các tín hiệu đầu vào, đầu

gạch ngang trên ký tự biểu thị tín hiệu hoạt động ở mức thấp (một khuyên tròn trong ký hiệu logic biểu thị điều đó). Nói cách khác, đầu vào ở mức thấp biểu thị có tín hiệu, đầu vào ở mức cao biểu thị không có tín hiệu. Q và $\bar{Q}$ biểu thị trạng thái của FF, đồng thời biểu thị đầu ra.



2. Nguyên lý làm việc

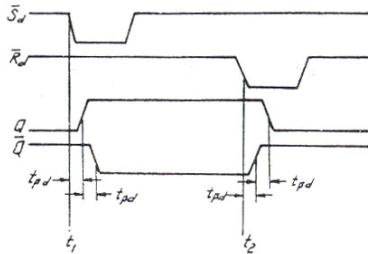
a) Hai trạng thái ổn định

Khi không có tín hiệu, tức là $\bar{R} = \bar{S} = 1$, mạch có hai trạng thái ổn định - Trạng thái 0 và trạng thái 1. Ở đây, chúng ta gọi $Q = 0$ và $\bar{Q} = 1$ là trạng thái 0, gọi $Q = 1$ và $\bar{Q} = 0$ là trạng thái 1. Ở trạng thái 0, vì $Q = 0$ hồi tiếp đến đầu vào cổng B làm B ngắt, duy trì $\bar{Q} = 1$. Mà $\bar{Q} = 1$ lại hồi tiếp đến đầu vào cổng A, cùng với $\bar{S} = 1$ làm cho cổng A thông, duy trì $Q = 0$. Do đó, mạch hoàn toàn tự động duy trì trạng thái 0. Tương tự, ở trạng thái 1, $Q = 1$ và $\bar{R} = 1$ làm cho cổng B thông, duy trì $\bar{Q} = 0$. Mà $\bar{Q} = 0$ lại làm cho cổng A ngắt, duy trì $Q = 1$. Vậy trạng thái 1 cũng được tự giữ.

b) Quá trình tiếp thu tín hiệu

Hình 5-1-1 Flip Flop RS cơ bản.
a) Sơ đồ logic b) Ký hiệu logic

Giả sử FF ở trạng thái 0, chúng ta đưa một xung âm vào đầu $\bar{S}$. Mạch điện sẽ chuyển biến nhanh sang trạng thái 1. Vì xung âm ở đầu vào $\bar{S}$, sau thời gian trễ truyền đạt 1 cấp cổng t_{pd} thì cổng A từ thông sang ngắt, đầu ra Q từ 0 sang 1. Lại sau thời gian trễ truyền đạt 1 cấp cổng t_{pd} nữa, thì cổng B từ ngắt sang thông, đầu ra $\bar{Q}$ từ 1



Hình 5-1-2.

Dạng sóng của Flip Flop RS cơ bản

sang 0. Vậy sau thời gian $2t_{pd}$ FF đã hoàn thành chuyển biến trạng thái từ 0 sang 1. Lúc này dù cho mất tín hiệu đầu vào, vì $\bar{Q} = 0$ đã hồi tiếp dẫn đến đầu vào cổng A, FF có thể tự động duy trì trạng thái 1, mà không trở lại trạng thái 0. Vì thế, xung âm đầu vào được gọi là xung kích.

Giả sử FF ở trạng thái 1, đưa xung âm vào đầu $\bar{R}$. Quá trình tương tự sẽ xảy ra, sau thời gian $2t_{pd}$, FF chuyển từ trạng thái 1 sang trạng thái 0. (Xem dạng sóng hình 5-1-2).

Vì tín hiệu ở đầu vào $\bar{S}$ có thể và chỉ có thể thiết lập FF ở trạng thái 1, tín hiệu đầu vào $\bar{R}$ có thể và chỉ có thể xóa FF ở trạng thái 0, nên thường gọi $\bar{S}$ là đầu vào đặt (set) và $\bar{R}$ là đầu vào xóa (Reset).

Nếu xem xét quá trình chuyển biến trạng thái tỉ mỉ hơn, chúng ta sẽ thấy rằng : vì hai mạch NAND nối ghép chéo nên không những FF có thể tự giữ khi không có tín hiệu, mà còn xuất hiện quá trình phản hồi dương trong mạch FF mỗi khi có xung kích tạo điều kiện để FF nhanh chóng hoàn thành chuyển trạng thái. Ví dụ, trong quá trình thiết lập 1, hễ $\bar{S}$ giảm đến mức mở cổng, mức đầu Q sẽ tăng lên, hồi tiếp đến đầu vào cổng B, làm cho cổng B chuyển từ ngắt sang thông, $\bar{Q}$ giảm mức, hồi tiếp đến đầu vào cổng A, lại càng làm cho cổng A ngắt sâu hơn, Q tăng mức hơn nữa, kết quả càng làm cho $\bar{Q}$ giảm mức thêm ... Cứ vậy, sóng dồn gió đập như bão áp, như thác lũ, làm cho cổng A ngắt rất nhanh, cổng B thông rất nhanh, FF chuyển trạng thái từ 0 sang 1 trong thời gian cực ngắn. Thông qua đầu Reset $\bar{R}$, cũng có quá trình phản hồi dương tương tự. Chính vì thế, dù cho sườn trước xung kích (xung âm) không dốc lắm, thì ở đầu ra của FF ta vẫn nhận được các xung có sườn rất dốc.

c) Không cho phép đồng thời đưa tín hiệu vào cả $\bar{R}$ và $\bar{S}$

Khi dùng loại FF này làm phần tử nhớ, không được phép đồng thời đưa tín hiệu kích vào cả hai đầu vào $\bar{R}$ và $\bar{S}$, tức là trạng thái $\bar{R} = \bar{S} = 0$ bị cấm. Do đặc tính mạch cổng NAND, khi $\bar{R} = \bar{S} = 0$ thì Q, $\bar{Q}$ đồng thời bằng 1, phần tử nhớ mà không phải là trạng thái 0, cũng không phải là trạng thái 1 như thế thì đầu còn là phần tử nhớ nữa ! mặt khác, khi $\bar{R}$ và $\bar{S}$ đồng thời từ 0 về 1 (bỏ tín hiệu) thì trạng thái của FF là bất

định, có thể là 0, cũng có thể là 1. Vì rằng những nhân tố quyết định trạng thái FF lúc này là không có cách nào biết trước chính xác được, chẳng hạn sự khác nhau rất nhỏ đặc tính động của hai cổng NAND hay tình huống nhiễu ở thời điểm xét. Đương nhiên, sau hai đầu vào $\bar{R}$ và $\bar{S}$ là không đồng thời, trạng thái FF có thể xác định sau khi bỏ tín hiệu.

d) Bảng chức năng và phương trình đặc trưng

Ta dùng ký hiệu Q^n biểu thị trạng thái FF trước khi tiếp thu tín hiệu, gọi là trạng thái hiện tại, dùng ký hiệu Q^{n+1} biểu thị trạng thái FF sau khi tiếp thu tín hiệu, gọi là trạng thái tiếp theo. Quan hệ logic giữa Q^{n+1} và Q^n , R, S biểu thị bằng bảng chức năng (bảng chân lý) mô tả sự chuyển đổi trạng thái xảy ra như bảng 5-1-1 dưới đây.

Bảng 5-1-1 : BẢNG CHỨC NĂNG CỦA FLIP FLOP RS CƠ BẢN

Q^n	R	S	Q^{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	x
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	x

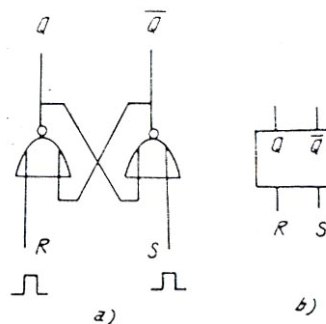
$\bar{R}\bar{S}$					
Q^n		00	01	11	10
		0	1	x	0
1		1	1	x	0

Hình 5-1-3 : Bảng Karnaugh của Q^{n+1}

Trong bảng hai trạng thái $Q^nRS = 011$, 111 là bị cấm, tương ứng đánh dấu chéo (x) ở cột Q^{n+1} , trong khi tối thiểu hóa có thể sử dụng.

Chúng ta có thể xem Q^n , R, S là các biến logic Q^{n+1} là hàm logic của các biến trên. Từ bảng 5-1-1 ta vẽ bảng Karnaugh của Q^{n+1} như hình 5-1-3.

Căn cứ vào bảng Karnaugh ta được phương trình đặc trưng sau của FF :



Hình 5-1-4 : FLip flop RS cơ bản dùng cổng NOR.

a) Sơ đồ logic b) Ký hiệu logic

$$\begin{cases} Q^{n+1} = S + \overline{R}Q^n \\ RS = 0 \text{ (ràng buộc từ trạng thái cấm)} \end{cases} \quad (5-1-1)$$

Bảng chức năng và phương trình đặc trưng

là phương pháp biểu diễn số học quan hệ logic giữa trạng thái hiện tại Q^n , các tín hiệu đầu vào r, S với trạng thái tiếp theo của Flip Flop RS cơ bản. Chúng miêu tả đầy đủ chức năng logic của nó.

3. Flip Flop RS cơ bản dùng cổng NOR

Xem hình 5-1-4, so sánh với hình 5-1-1 ta thấy có hai điểm khác biệt : vị trí R, S đảo và mức tích cực của tín hiệu cũng đảo (không có dấu gạch ngang). Các đầu vào R, S ở mức cao biểu thị có tín hiệu, ở mức thấp biểu thị không có tín hiệu. Căn cứ vào tính chất cổng NOR, chúng ta có thể tìm hiểu nguyên lý công tác và viết ra bảng chức năng, phương trình đặc trưng của mạch này.

Khi $R = S = 0$ thì $Q, \overline{Q}$ đồng thời bằng 0, đó là trạng thái cấm.

4. Đặc điểm cơ bản

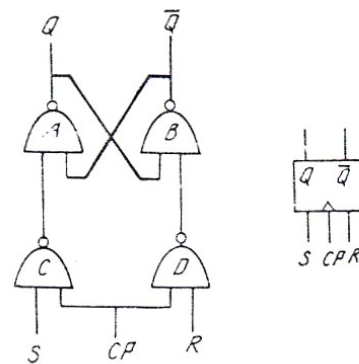
Ưu điểm : mạch đơn giản, có thể nhớ 1 bit là cơ sở để cấu trúc các FF hoàn hảo hơn.

Nhược điểm : tín hiệu trực tiếp điều khiển trạng thái đầu ra, ứng dụng bị hạn chế, tín hiệu vào ràng buộc lẫn nhau (không ở trạng thái cấm).

5.1.2. Flip Flop RS đồng bộ

1. Cấu trúc mạch và ký hiệu

Để khắc phục nhược điểm của loại Flip Flop RS cơ bản là trực tiếp điều khiển, người ta thêm vào hai cổng điều khiển và một tín hiệu điều khiển, nên tín hiệu đầu vào được truyền qua cổng điều khiển, xem hình 5-1-5. Các cổng A, B làm thành Flip Flop RS cơ bản, các cổng C, D là cổng điều khiển, CP là tín hiệu điều khiển, thường là xung đồng hồ hoặc xung mở chọn mạch. Trong ký hiệu logic, đầu CP có dấu Δ , tín hiệu này tích cực với sườn dương của xung.



Hình 5-1-5 Flip flop RS đồng bộ

a) Sơ đồ logic b) Ký hiệu logic

2. Nguyên lý làm việc

Khi $CP = 0$, các cổng C, D bị ngắt, FF bị cấm, duy trì trạng thái cũ. Khi $CP = 1$, các cổng C, D thông thì FF sẵn sàng (tiếp thu tín hiệu), nó tiếp thu tín hiệu đầu vào R, S . Dễ dàng thấy rằng tình huống công tác của mạch lúc này giống như Flip Flop

RS cơ bản. Nếu $R = 0$; $S = 1$ đầu ra cổng C ở mức thấp, FF lập ở trạng thái 1. Ngược lại, nếu $R = 1$, $S = 0$ đầu ra cổng D ở mức thấp, FF bị xóa về trạng thái 0. Nếu $R = S = 1$ thì các cổng C, D đều đưa ra mức thấp, dẫn đến Q và $\bar{Q}$ đều là mức cao, đó là trạng thái cấm. Có thể thấy rằng bảng chức năng và phương trình đặc trưng biểu thị quan hệ logic giữa Q^{n+1} với Q^n , R, S không khác gì của Flip Flop RS cơ bản, chẳng qua chúng chỉ đúng trong điều kiện $CP = 1$. Tức là các quan hệ logic ở bảng 5-1-1 đối với Flip Flop RS đồng bộ chỉ đúng khi nào xuất hiện xung đồng hồ ($CP = 1$).

3. Mạch chốt D

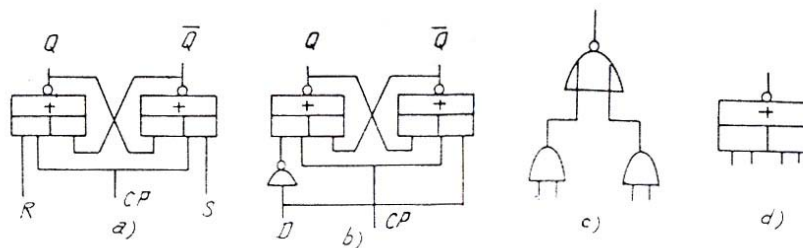
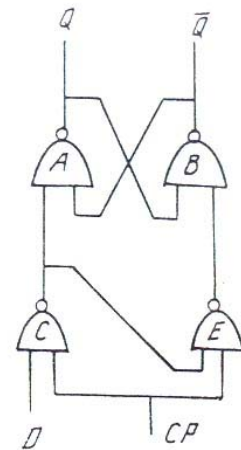
Hình 5-1-6 là sơ đồ logic mạch chốt D, nó được cấu tạo trên cơ sở mạch Flip Flop RS đồng bộ nhằm giải quyết vấn đề ràng buộc lẫn nhau của các tín hiệu đầu vào R, S. Đầu ra cổng C nối đến các đầu vào cổng A, E. Khi $CP = 0$, cổng C, E ngắt nên Flip Flop duy trì trạng thái cũ. Khi $CP = 1$ nếu $D = 0$ thì đầu ra C ở mức cao, đầu ra E ở mức thấp, Flip Flop ở trạng thái 0, nếu $D = 1$ thì đầu ra C ở mức thấp, đầu ra E ở mức cao, Flip Flop ở trạng thái 1. Vậy tức là D ở mức nào thì Q ở đúng mức ấy. Phương trình đặc trưng của mạch chốt Flip Flop D là :

$$Q^{n+1} = D \text{ với điều kiện } CP = 1 \quad (5-1-2)$$

4. Dùng cổng NORAND để cấu trúc Flip Flop RS đồng bộ và mạch chốt D. (Cổng NORAND xem mục 3-3-4-3)

Với điều kiện tiên quyết $CP = 1$, mạch trên hình 5-1-7a có bảng chức năng trùng hợp với bảng 5-1-1 và phương trình đặc trưng trùng hợp với phương trình 5-1-1, mạch trên hình 5-1-7b có phương trình đặc trưng trùng hợp với phương trình 5-1-2.

Hình 5-1-6 : Mạch chốt D



Hình 5-1-7 : Flip Flop cấu trúc từ NORAND

a) Flip Flop RS đồng bộ b) Mạch chốt D c) Cổng NORAND d) Kí hiệu rút gọn của

5. Đặc điểm cơ bản của Flip Flop RS đồng bộ

Ưu điểm : Điều khiển chọn mở mạch.
 Khi có xung đồng hồ CP = 1 thì Flip Flop tiếp thu tín hiệu vào, còn nếu CP = 0 thì Flip Flop bị cấm.

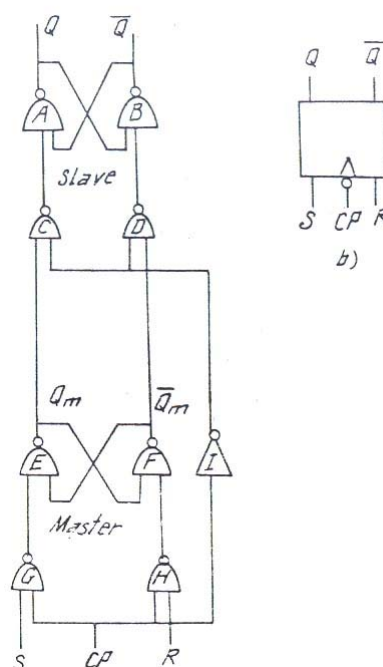
Nhược điểm : Trong thời gian CP = 1 tín hiệu vào vẫn trực tiếp điều khiển trạng thái đầu ra của FF, có mối ràng buộc R và S để tránh trạng thái cấm, tuy nhiên cấu trúc nối mạch của mạch chốt D giải quyết điều này.

5.1.3. Flip Flop RS master slave

Mạch này giải quyết triệt để vấn đề trực tiếp điều khiển, đó là nhược điểm của các loại FF trên.

1. Cấu trúc mạch và ký hiệu

Trên hình 5-1-8 có hai Flip Flop RS đồng bộ nối ghép dây chuyển với nhau, một là FF master, một là FF slave, xung đồng hồ cung cấp cho chúng là đảo nhau (qua mạch đảo I).



Hình 5-1-8 : Flip Flop RS master slave.

a) Sơ đồ logic

b) Ký hiệu logic

2. Nguyên lý làm việc

a) Khi CP = 0, Cổng G, H ngắt nên FF master ngắt. CP = 1, cổng C, D thông nên FF slave sẵn sàng, nó tiếp thu tín hiệu đầu ra master, do đó $Q = Q_m$, $\bar{Q} = \bar{Q}_m$

b) Sau đột biến sườn dương CP

CP = 1 master thông qua các cổng G, H tiếp nhận tín hiệu đầu vào. Vậy :

$$\begin{cases} Q_m^{n+1} = S + \bar{R}Q_m^n \\ RS = 0 \end{cases} \text{ với điều kiện CP=1}$$

$\bar{CP} = 0$ Slave bị ngắt, đầu ra Q, $\bar{Q}$ duy trì trạng thái cũ.

c) Khi sườn âm xung đồng hồ CP

CP đột biến xuống 0, master bị ngắt. CP đột biến lên 1, slave tiếp nhận tín hiệu đã được master ghi nhớ từ thời gian CP = 1. Nghĩa là slave chuyển đổi trạng thái . Vậy :

$$\begin{cases} Q^{n+1} = S + \bar{R}Q^n \\ RS = 0 \end{cases} \quad (5-1-3)$$

với điều kiện đã xuất hiện sườn âm xung đồng hồ CP.

Flip Flop RS master slave tuy rằng để master tiếp nhận tín hiệu đầu vào trong khoảng thời gian $CP = 1$, nhưng đầu ra lúc đó vẫn không chuyển đổi trạng thái, chỉ khi đã xuất hiện sườn âm xung đồng hồ CP thì đầu ra mới chuyển trạng thái. Người ta gọi sự kiện này là kích bằng sườn âm. FF có trạng thái đầu ra không chịu ảnh hưởng trực tiếp của các tín hiệu đầu vào R, S bất kỳ lúc nào. Vậy vấn đề trực tiếp điều khiển đã được giải quyết. Trên hình 5-1-8b đầu vào CP có dấu ô biểu thị rằng tính tích cực của tín hiệu CP là sườn âm của nó.

3. Đặc điểm cơ bản

Ưu điểm : Cấu trúc điều khiển master slave đã giải quyết vấn đề trực tiếp điều khiển, trong khi $CP = 1$ tiếp thu tín hiệu, sườn âm của CP kích chuyển trạng thái đầu ra.

Nhược điểm : Vẫn còn ràng buộc giữa R và S khi $CP = 1$.

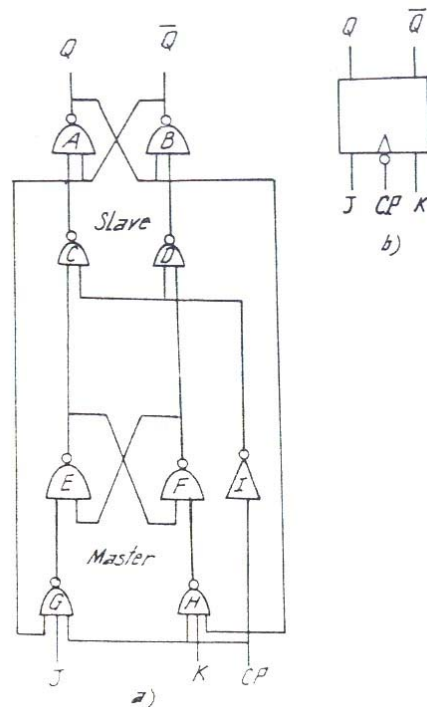
5.1.4. Flip Flop JK master slave

1. Cấu trúc mạch và ký hiệu

Loại Flip Flop RS master slave nói ở trên vẫn còn ràng buộc R và S, nguyên nhân chính là khi $R = S = 1$ đầu ra các cổng G, H đều ở mức thấp, dẫn đến tình huống không mong muốn là $Q_m = 1$ và $\overline{Q}_m = 1$. Cần chú ý một điều sau :

Xét mạch Flip Flop RS master slave khi $CP = 1$, Q và $\overline{Q}$ không đổi trạng thái và là đảo

Hình 5-1-9 : Flip Flop JK master slave.
a) Sơ đồ logic
b) Ký hiệu logic



của nhau. Chỉ cần đem mức các đầu ra Q và $\overline{Q}$ đưa đến đầu vào của G, H thì có thể khắc phục tình trạng cả Q và $\overline{Q}_m$ đều bằng 1, giải quyết vấn đề ràng buộc giữa tín hiệu đầu vào.

Để phân biệt với Flip Flop RS master slave, mạch cải tiến không dùng tên R, S nữa, mà lấy tên mới là J, K cho các đầu vào, và tên của mạch cải tiến là Flip Flop JK master slave, gọi tắt là Flip Flop JK (hình 5-1-9).

2. Nguyên lý làm việc

Theo sự trình bày trên đây về sự cải tiến của Flip Flop JK, ta thấy nguyên lý công tác của nó giống như của Flip Flop RS master slave, chỉ khác bởi sự tương đương sau của các tín hiệu đầu vào :

$$S = J\overline{Q}^n \quad (5-1-4)$$

$$R = KQ^n \quad (5-1-5)$$

Áp dụng công thức (5-1-3) ta có :

$$Q^{n+1} = S + \overline{R}Q^n = J\overline{Q}^n + \overline{K}Q^n = J\overline{Q}^n + \overline{K}Q^n \quad (5-1-6)$$

Với điều kiện đã xuất hiện sườn âm CP.

Công thức (5-1-6) là phương trình đặc trưng của Flip Flop JK nó phản ánh quan hệ logic giữa Q^{n+1} với Q^n , J, K. Nhờ $\overline{Q}^n$ và Q^n phản hồi về cổng điều khiển G, H mà J và K không còn ràng buộc lẫn nhau.

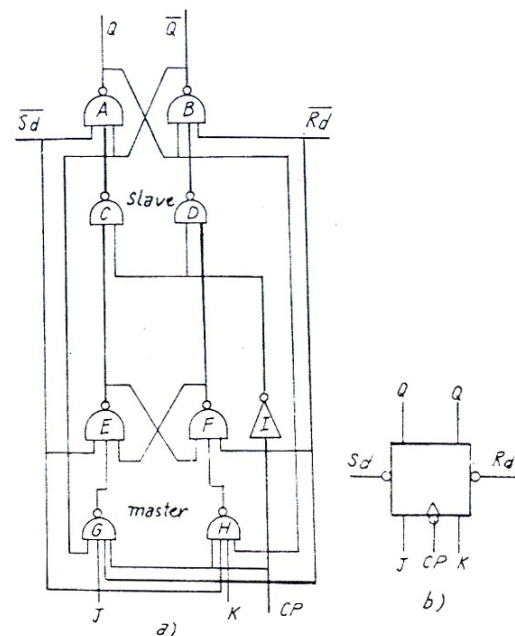
3. Tác dụng các đầu vào đi bộ $\overline{R}_d, \overline{S}_d$ (đi bộ : Asynchronous)

a) Các đầu vào đồng bộ

Bất kỳ lúc nào thì các đầu vào J, K chỉ tác động khi có sự điều khiển đồng bộ của xung đồng hồ CP ; vì vậy J, K là đầu vào đồng bộ . (Các đầu vào R, S của Flip Flop RS master slave đồng bộ cũng là các đầu vào đồng bộ).

b) Các đầu vào đi bộ

Để phần trình bày trên đơn giản, chưa vẽ các đầu vào đi bộ $\overline{R}_d, \overline{S}_d$ trên hình 5-1-9a. Có thể biết vị trí các đầu



Hình 5-1-10 : Flip Flop JK master slave.

a) Sơ đồ logic

b) Kí hiệu logic

vào di bộ này trên hình 5-1-10. Tác dụng của các đầu vào $\overline{R_d}, \overline{S_d}$ không chịu điều khiển đồng bộ của xung đồng hồ CP, vì vậy $\overline{R_d}, \overline{S_d}$ là các đầu vào di bộ. Như trên hình 5-1-10a chỉ rõ, $\overline{R_d}$ nối đến đầu vào B, F, G. Do đó xung âm đầu $\overline{R_d}$ không những xóa cả master và slave mà còn ngắt cổng G, trong thời gian CP = 1 không cho phép J = 1 thiết lập master ở trạng thái 1, điều đó bảo đảm trạng thái 0. Tương tự $\overline{S_d}$ nối đến đầu vào A, E, H. Xung âm đầu vào $\overline{S_d}$ bảo đảm trạng thái 1. Trên ký hiệu logic (hình 5-1-10b), khuyên tròn ở đầu vào S_d, R_d biểu thị tính tích cực của chúng là mức thấp.

4. Vấn đề một lần chuyển

Trong thời gian CP = 1 thì master chỉ chuyển đổi trạng thái một lần, hiện tượng này gọi là một lần chuyển. Như ta đã biết, các trạng thái đảo nhau của Q và $\overline{Q}$ phản hồi về H, G tương ứng làm cho một trong hai cổng (H hoặc G) bị ngắt, nếu một đầu vào có tín hiệu thì có thể vì chỉ chuyển đổi trạng thái một lần. Ví dụ khi $\overline{Q} = 0, Q = 1$, cổng G bị khóa, J không tác dụng, tín hiệu chỉ có thể từ đầu vào K, qua cổng H, xóa master về 0. Và một khi đã ở trạng thái 0 rồi, dù tín hiệu K biến đổi thế nào, master vẫn duy trì trạng thái 0. Ngược lại khi $\overline{Q} = 1, Q = 0$ thì H bị khóa, chỉ có tín hiệu J mới tác dụng, thông qua cổng G, đặt master lên 1, một khi đã ở trạng thái 1 rồi, master sẽ duy trì.

Vấn đề một lần chuyển không những hạn chế tác dụng của Flip Flop JK master slave mà còn làm giảm năng lực chống nhiễu của nó. Chẳng hạn, khi CP = 1 mà J, K đổi trạng thái nhiều lần thì cũng chỉ có một lần chuyển mà thôi. Mặt khác, thời điểm một lần chuyển trạng thái đó có thể rơi vào sườn dương, giữa xung, hay sườn âm của xung đồng hồ CP. Nếu như không biết chính xác quy luật tín hiệu J, K thì không có thể xác định trạng thái tiếp theo của Flip Flop. Vậy khi sử dụng Flip Flop JK master slave, thường đều yêu cầu trạng thái J, K duy trì không đổi trong thời gian CP = 1. Đã thế thì quá trình làm việc của Flip Flop nói gọn lại là : sườn dương tiếp thu, sườn âm chuyển. Nếu nhiều xếp chồng với J, K sao cho gây ra một lần chuyển sai trong khi CP = 1 thì tín hiệu nhiễu đó tiếp tục tác động đến hệ thống. Vậy vấn đề một lần chuyển là nguyên nhân làm giảm năng lực chống nhiễu của mạch này.

5. Flip Flop JK master slave trong một vỏ IC

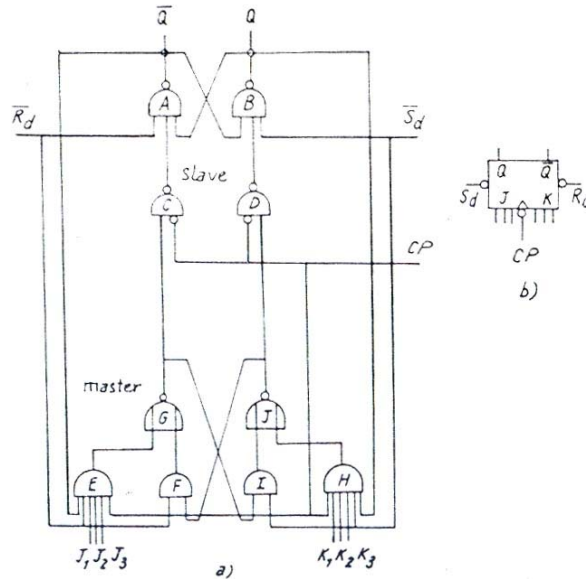
Hình 5-1-11a trình bày sơ đồ logic tương đương của Flip Flop JK master slave trong một vỏ IC. Master bao gồm hai NORAND, slave khá đơn giản, CP mức cao cấm, mức thấp cho phép. Nguyên lý công tác không khác mạch hình 5-1-10 nói trên.

6. Đặc điểm cơ bản của Flip Flop JK master slave

Ưu điểm : J và K không bị ràng buộc lẫn nhau, các IC của chúng được sản xuất nhiều, sử dụng rộng rãi, tính năng ưu việt.

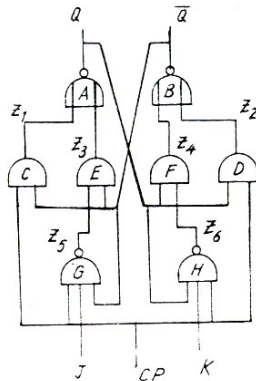
Nhược điểm : Vấn đề một lần chuyển, thường yêu cầu J, K duy trì không đổi trong thời gian CP = 1.

Hình 5-1-11 Flip Flop JK master slave trong vỏ IC
a) Sơ đồ logic tương đương
b) Kí hiệu logic



5.1.5. Flip Flop JK kích bằng sườn xung (Edge triggered FF)

1. Cấu trúc mạch



Hình 5-1-12 : Flip Flop JK kích bằng sườn xung.

Mạch trên hình 5-1-12 là Flip Flop JK kích bằng sườn xung, nó giải quyết vấn đề một lần chuyển của Flip Flop JK master slave. Khi CP = 0, CP = 1, hay khi sườn dương của CP các tín hiệu J, K đều không tác dụng. Chỉ trong thời gian sườn âm của CP thì Flip Flop mới chuyển trạng thái theo phương trình đặc trưng dưới đây :

$$Q^{n+1} = JQ^n + \bar{K}Q^n$$

2. Nguyên lý làm việc

a) Khi CP = 0, các cổng G, H bị khóa J, K không tác dụng, Flip Flop duy trì trạng thái cũ.

b) Khi CP = 1 các cổng C, D, G, H thông nhưng :

$$Z_1 = \bar{Q}^n Z_2 = Q^n$$

$$Z_3 = Z_5 \bar{Q}^n = \bar{J} \bar{Q}^n \cdot \bar{Q}^n = \bar{J} \bar{Q}^n$$

$$Z_4 = Z_6 Q^n = \bar{K} Q^n \cdot Q^n = \bar{K} Q^n$$

$$Q^{n+1} = \bar{Z}_1 + Z_3 = \bar{Q}^n + \bar{J} \bar{Q}^n = Q^n$$

$$\bar{Q}^{n+1} = \bar{Z}_2 + Z_4 = \bar{Q}^n + \bar{K} \bar{Q}^n = \bar{Q}^n$$

Vậy FF duy trì nguyên trạng, JK đều không tác dụng.

c) Khoảng thời gian sườn dương của CP, do tác dụng trễ của các cổng NAND G và H mà cổng C, D thông trước :

$$Z_1 = \overline{Q}^n \quad Z_2 = Q^n$$

$$\text{Tiếp sau mới có: } Z_3 = \overline{J.Q}^n ; \quad Z_4 = \overline{K.Q}^n$$

Vì vậy :

$$Q^{n+1} = \overline{Z_1 + Z_3} = \overline{\overline{Q}^n + \overline{J.Q}^n} = Q^n$$

$$\overline{Q}^{n+1} = \overline{Z_2 + Z_4} = \overline{Q^n + \overline{K.Q}^n} = \overline{Q}^n$$

J và K cũng không tác dụng.

d) Tình huống mạch trong khoảng sườn âm của CD khác hẳn. Do tác dụng trễ của các cổng NAND G và H mà C, D ngắt trước :

$$Z_1 = Z_2 = 0$$

mà đầu ra các cổng NAND duy trì $Z_5 = \overline{J.Q}^n, Z_6 = \overline{K.Q}^n$ trong khoảng thời gian t_{pd} nữa. Dễ dàng thấy rằng trong t_{pd} đó, các cổng NOR A và B, các cổng AND E và F cấu trúc như một Flip Flop RS cơ bản, với :

$$\overline{S} = Z_5 = \overline{J.Q}^n \quad \overline{R} = Z_6 = \overline{K.Q}^n$$

Căn cứ vào phương trình đặc trưng của Flip Flop RS cơ bản, ta có :

$$Q^{n+1} = S + \overline{R.Q}^n = \overline{J.Q}^n + \overline{K.Q}^n.Q^n = \overline{J.Q}^n + \overline{K.Q}^n$$

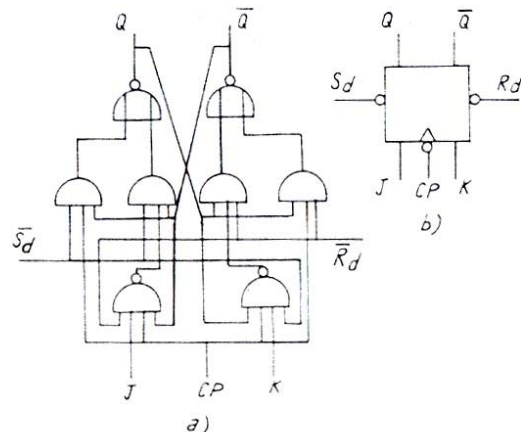
Do $Q^n, \overline{Q}^n$ phân biệt hồi tiếp đầu vào H và G nên J và K không bị ràng buộc lẫn nhau.

Vậy loại mạch điện này không những có tính năng tốt như Flip Flop JK master slave và có đặc điểm kích bằng sườn âm xung đồng hồ CP mà còn không có vấn đề một lần chuyển.

3. Đặc điểm cơ bản

Ưu điểm : Kích bằng sườn âm xung đồng hồ CP ; đây là loại FF tính năng ưu việt, chống nhiễu tốt và sử dụng linh hoạt.

Nhược điểm : Mạch điện dùng tác dụng trễ của các cổng NAND giải quyết vấn đề một lần chuyển, do đó yêu cầu công nghệ chế tạo phải tinh xảo để bảo đảm mạch điện công tác tin cậy.



Hình 5-1-13 Flip Flop JK kích bằng sườn xung.
a) Sơ đồ logic b) Ký hiệu logic

Hình 5-1-13 giới thiệu thêm các đầu vào định bộ $\overline{R_d}, \overline{S_d}$. Các đầu vào này tạo thuận lợi trong sử dụng. Xung âm đầu vào $\overline{S_d}$ xóa. Trong sơ đồ và ký hiệu logic của nó, dấu ngang và khuyên tròn biểu thị rằng tín hiệu tích cực ở mức thấp.

5.1.6. Flip Flop D (mạch chốt D cải tiến để giải quyết vấn đề điều khiển trực tiếp).

1. Cấu trúc mạch điện

Xem hình 5-1-14 thêm vào mạch chốt D hai cổng E và F.

2. Nguyên lý làm việc

a) Khi $CP = 0$

Các cổng C, D bị khóa, $Z_1 = Z_2 = 1$, Flip Flop cơ bản bao gồm các cổng A, B duy trì trạng thái cũ.

- Nếu $D = 1$ thì :

$$Z_4 = \overline{D.Z_2} = \overline{1.1} = 0$$

$$Z_3 = \overline{Z_1.Z_4} = \overline{1.0} = 1$$

CP trong vai trò tín hiệu đầu vào đối với cổng C thông, cổng D ngắt.

- Nếu $D = 0$ thì : $Z_4 = 1$ $Z_3 = 0$

CP trong vai trò tín hiệu đầu vào đối với cổng C ngắt, cổng D thông.

b) Thời gian sườn dương của CP.

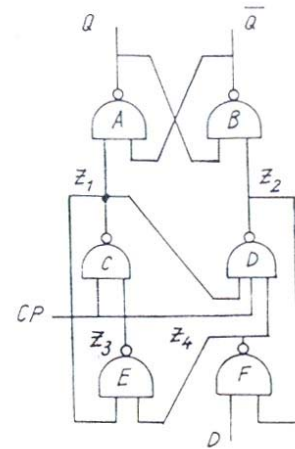
- Nếu $D = 1$ thì D bị ngắt, CP chỉ có thể thông qua cổng C mở, vậy $Z_1 = \overline{Z_3.CP} = \overline{1.1} = 0$.

$Z_1 = 0$ dẫn đến ba tác động sau : một là kích Flip Flop thiết lập 1, $Q = 1, \overline{Q} = 0$; hai là ngắt cổng D, ngăn trở Z_2 chuyển sang mức thấp, nghĩa là ngăn trở sự tạo ra tín hiệu kích chuyển Flip Flop về trạng thái 0 ; ba là Z_1 đưa đến đầu vào cổng E bảo đảm

Hình 5-1-14 Flip Flop D

$Z_3 = 1$, do đó duy trì $Z_1 = 0$ suốt thời gian $CP = 1$, tức là duy trì tín hiệu đặt Flip Flop ở trạng thái 1.

Đường nối từ đầu ra cổng C đến đầu vào cổng E gọi là đường duy trì trạng thái 1 của FF. Đường nối từ đầu ra cổng C đến đầu vào cổng D gọi là đường ngăn trở trạng thái 0 của FF. Một khi $Z_1 = 0$ đưa đến đầu vào các cổng D, E và sinh ra các tác động nói trên rồi thì dù tín hiệu D có thay đổi cũng không thể ảnh hưởng đến trạng thái 1 của FF.



Hình 5-1-14. Flip Flop D

- Nếu $D = 0$ thì cổng C bị ngắt. CP chỉ có thể thông qua cổng D mở, vậy $Z_2 = \overline{Z_1 Z_4 CP} = \overline{1.1.1} = 0$

$Z_2 = 0$ dẫn đến hai tác động sau : một là xóa FF về 0, $Q = 0$, $\overline{Q} = 1$; hai là làm cho cổng F bị ngắt, bảo đảm $Z_4 = 1$, do đó duy trì $Z_2 = 0$, tức là duy trì trạng thái 0 của FF. $Z_4 = 1$ còn duy trì Z_3 ở mức thấp, ngăn trở sự tạo ra tín hiệu $Z_1 = 0$ kích chuyển FF về trạng thái 1.

Có thể thấy rằng đường nối từ đầu ra cổng D đến đầu vào cổng F vừa duy trì FF ở trạng thái 0, vừa ngăn trở FF chuyển về trạng thái 1. Chỉ cần $Z_2 = 0$ đã đến đầu vào cổng F thì dù biến hóa thế nào, tín hiệu D cũng không làm thay đổi trạng thái của FF được nữa.

Tóm lại :

$$Q^{n+1} = D \text{ với điều kiện đã xuất hiện sườn dương của CP} \quad (5-1-7)$$

Hơn nữa, khi đã có tác dụng duy trì nguyên trạng và ngăn trở chuyển đổi sang trạng thái khác trong suốt quá trình $CP = 1$ thì tín hiệu D không còn tác động đến mạch nữa. Vì vậy, đây là mạch điều khiển sườn trước và kích bằng sườn dương CP.

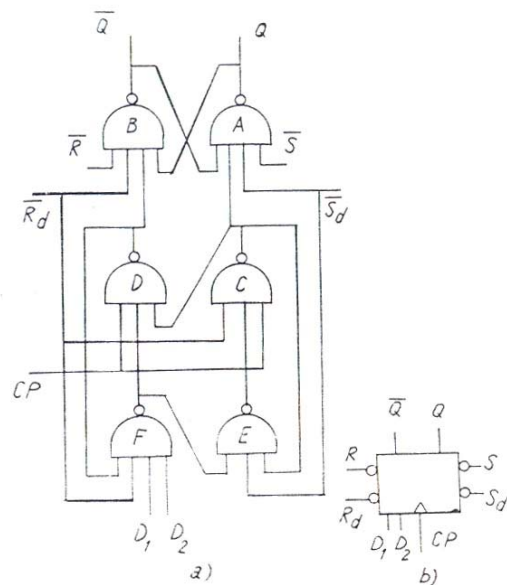
3. Tác dụng của các đầu vào dị bộ

$$\overline{R}, \overline{R}_d, \overline{S}, \overline{S}_d$$

Để sử dụng thêm thuận tiện, Flip Flop D còn có thêm các tín hiệu tích cực mức thấp ở các đầu vào Set ($\overline{S}, \overline{S}_d$) và Reset ($\overline{R}, \overline{R}_d$)

Hình 5-1-15 biểu thị sơ đồ logic tương đương và ký hiệu logic Flip Flop D trong vỏ IC.

$\overline{R}, \overline{S}$ chỉ dùng trong thời gian $CP = 0$, nếu không có thể phản tác dụng duy trì ngăn trở nói trên. Chẳng hạn khi $CP = 1$ và FF đang duy trì trạng thái 1. Nếu có xung âm đầu vào $\overline{R}$ thì có thể xuất hiện trạng thái cấm : Q và $\overline{Q}$ đồng thời mức cao. Hơn nữa, xung âm $\overline{R}$ kết thúc trước CP, do tác dụng duy trì trạng thái 1, FF vẫn ở trạng thái 1, còn nếu CP kết thúc trước $\overline{R}$ thì FF có thể xóa về 0. Sử dụng $\overline{S}$ để đặt FF vào trạng



Hình 5-1-15 Flip Flop D trong vỏ IC

a) Sơ đồ logic tương đương

b) Ký hiệu logic

thái 1 cũng gặp vấn đề tương tự, chỉ khác là nó phản tác dụng với việc duy trì trạng thái 0.

$\overline{R_d}$, $\overline{S_d}$ không bị trạng thái CP hạn chế. Ví dụ, xung âm ở đầu $\overline{R_d}$ làm FF về 0, nếu $CP = 1$ và FF đang duy trì trạng thái 1 thì $\overline{R_d} = 0$ không chỉ xóa FF về 0 mà còn thông qua đường nối đến cổng C, F bắt buộc mạch đang duy trì 1 ngăn trở 0 phải trở thành duy trì 0 ngăn trở 1. Cũng với lý do tương tự, xung âm đầu $\overline{S_d}$ cũng có thể làm cho FF chuyển sang trạng thái 1 một cách tin cậy.

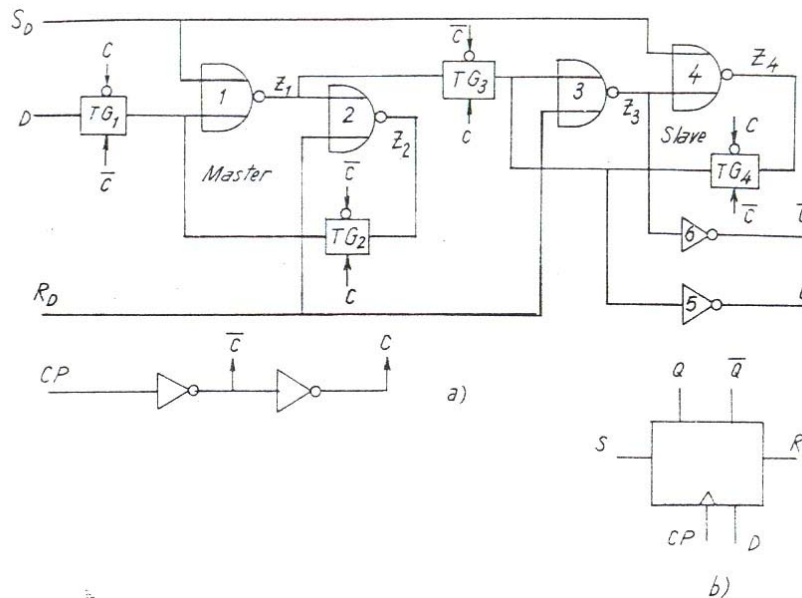
4. Đặc điểm cơ bản

Ưu điểm : Điều khiển sườn xung, kích với sườn dương CP, trong thời gian $CP = 1$ mạch tự giữ nguyên trạng.

Nhược điểm : Trong một số trường hợp sử dụng không tiện bằng Flip Flop JK.

5.1.7. Flip Flop CMOS

Các loại Flip Flop CMOS trong vỏ IS có đặc điểm tiêu hao năng lượng rất ít, năng lực chống nhiễu rất mạnh, phù hợp với nhiều điện thế nguồn khác nhau. Nhờ sử dụng các cổng chuyển mạch (xem mục 2-3-3) nên kết cấu mạch điện đơn giản.



Hình 5-1-16 Flip Flop D master slave CMOS.

a) Sơ đồ logic b) Kí hiệu logic

1. Flip Flop D họ CMOS

a) Cấu trúc mạch và ký hiệu

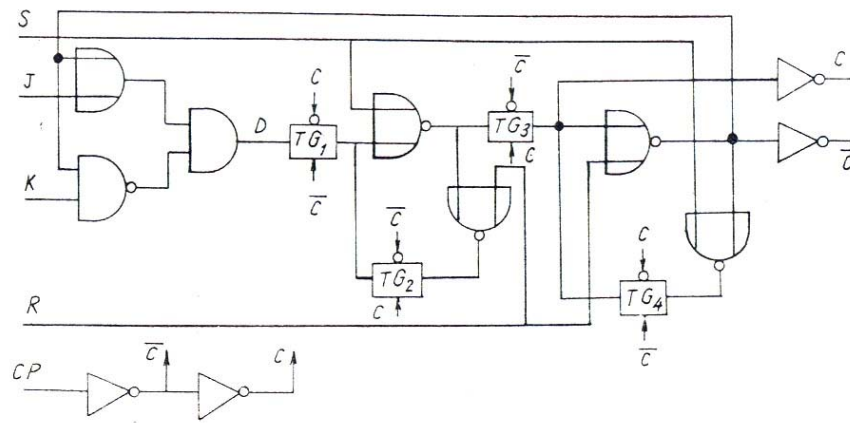
Xem hình 5-1-16 FF master bao gồm các cổng NOR 1,2 và cổng chuyển mạch TG_2 , FF slave bao gồm các cổng NOR 3, 4 và cổng chuyển mạch TG_4 . TG_1 là cổng điều khiển ở đầu vào. TG_3 là cổng điều khiển giữa master và slave. C và $\bar{C}$ là các tín hiệu đồng hồ đảo nhau. R_D , S_D là các đầu vào dệ bộ Reset và Set tích cực ở mức cao. D là tín hiệu vào. Q và $\bar{Q}$ là các đầu ra.

b) Nguyên lý làm việc

- Khi $CP = 0$, $\bar{C} = 1$, $C = 0$, TG_1 và TG_4 thông, TG_2 và TG_3 ngắt. Tín hiệu vào D thông qua TG_1 đến NOR 1, $Z_1 = \bar{D}$, $Z_2 = D$. Nhờ TG_4 thông, slave tự giữ.
- Khi $CP = 1$, $\bar{C} = 0$, $C = 1$, TG_1 và TG_4 ngắt, TG_2 và TG_3 thông. Đường tín hiệu vào bị ngắt. Master tự giữ nhờ đường hồi tiếp qua TG_2 Slave chuyển đổi trạng thái theo mức tín hiệu Z_1 , tức là đưa tín hiệu đầu vào D đã nhớ ra đầu ra, vậy FF D này được kích bằng sườn dương CP. Phương trình đặc trưng là :

$Q^{n+1} = D$ với điều kiện đã xuất hiện sườn dương CP (5-1-8).

2. FF JK CMOS



Hình 5-1-17 FF JK master slave CMOS

Hình 5-1-17 . sơ đồ này là cải tiến của mạch FFD trên đây.

Căn cứ sơ đồ logic, ta có :

$$D = (J + Q^n)KQ^n = JKQ^n + KQ^nQ^n$$

$$D = J\bar{K} + JQ^n + \bar{K}Q^n = JQ^n + \bar{K}Q^n$$

Thay vào (5-1-8), ta được :

$$Q^{n+1} = J\overline{Q}^n + \overline{K}Q^n \text{ với điều kiện đã xuất hiện sườn dương CP (5-1-9)}$$

Đây là phương trình đặc trưng của FF JK master slave CMOS.

Tác dụng tín hiệu đầu vào dị bộ R_D , S_D của hai mạch hình 5-1-16 và 5-1-17 là giống nhau.

5.2. PHÂN LOẠI FLIP FLOP THEO CHỨC NĂNG, SỰ CHUYỂN ĐỔI LẦN NHAU.

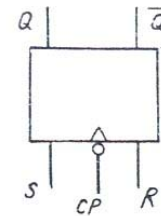
Các loại FF trình bày trong mục 5-1 trên đây : RS master slave, JK master slave, JK sườn xung, D đều có tín hiệu đồng hồ điều khiển ; vậy chúng được gọi là FF định thời theo xung đồng hồ CP. Căn cứ vào sự khác biệt tính năng logic dưới tác dụng điều khiển của CP, ta phân FF thành 5 loại : RS, D, T, T', JK. Các loại đó có thể chuyển đổi lẫn nhau theo phương pháp xác định.

5.2.1. Phân loại Flip Flop theo chức năng

1. Flip Flop RS

a) Định nghĩa

Flip Flop RS là mạch điện có chức năng thiết lập trạng thái 1 (Set) , trạng thái 0 (Reset) và duy trì (nhớ) các trạng thái đó căn cứ vào các tín hiệu đầu vào R, S và tín hiệu đồng hồ CP. Những điều đã trình bày trong tiết trên về RS là phù hợp với định nghĩa này. Ký hiệu logic của mạch trên hình 5-2-1, phương trình đặc trưng biểu thị chức năng logic của nó là :



Hình 5-2-1 Ký hiệu logic của Flip Flop RS.

$$\begin{cases} Q^{n+1} = S + \overline{R}Q^n \\ RS = 0 \\ \text{Với điều kiện đã xuất hiện sườn âm CP} \end{cases}$$

(5-2-1)

$R = 0, S = 1$	CP (sườn âm)	thì $Q^{n+1} = 1$
$R = 1, S = 0$	CP	thì $Q^{n+1} = 0$
$R = 0, S = 0$	CP	thì $Q^{n+1} = Q^n$ (duy trì)
$R = 1, S = 1$	trạng thái cấp	

b) Phương pháp biểu thị chức năng logic

Bảng chức năng là bảng tín hiệu đầu vào kích.

Bảng chức năng của Flip Flop RS 5-2-1 dùng hình thức bảng liệt kê các trạng thái logic để biểu thị chức năng logic của nó.

Bảng tín hiệu đầu vào kích của Flip Flop RS 5-2-2 có phần bên trái kê ra các yêu cầu chuyển đổi trạng thái của FF, và có phần bên phải kê ra các điều kiện tín hiệu đầu vào kích cần bảo đảm để đạt đến các yêu cầu tương ứng. Nếu các điều kiện được bảo đảm thì FF sẽ chuyển đổi trạng thái theo yêu cầu một khi xung đồng hồ cho phép. Ví dụ : Yêu cầu $Q^n \rightarrow Q^{n+1}$ kiểu duy trì $0 \rightarrow 0$ thì điều kiện cần là $S = 0$ (không phụ thuộc vào R, $R = x$). Nếu yêu cầu chuyển $0 \rightarrow 1$ thì điều kiện lại là $R = 0, S = 1$.

Bảng 5-2-1

Q^n	R	S	Q^{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	x
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	x

Bảng 5-2-2

$Q^n \rightarrow$	Q^{n+1}	R	S
0	0	x	0
0	1	0	1
1	0	1	0
1	1	0	x

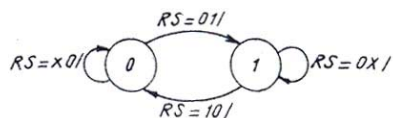
Như trên đã nói, các quan hệ logic của bảng 5-2-1 và 5-2-2 đều có chung một điều kiện : đã xuất hiện sườn âm CP, nếu điều kiện này không thỏa mãn, FF duy trì nguyên trạng.

Phương trình đặc trưng và phương trình kích.

Phương trình đặc trưng dùng hàm logic miêu tả quan hệ giữa Q^{n+1} với Q^n và các tín hiệu đầu vào, do đó nó rất tiện dùng công cụ toán logic (các công thức và định lý của đại số logic) (5-2-1) là phương trình đặc trưng của FF RS.

Phương trình kích dùng hàm logic của tín hiệu đầu vào kích biểu thị, phần sau sẽ trình bày rõ hơn.

c) Đồ hình trạng thái :



Hình 5-2-2 Đồ hình trạng thái của Flip Flop RS.

Hình 5-2-2 là đồ hình trạng thái của FFRS, nó biểu thị trực quan quy luật chuyển đổi trạng thái của FF dưới tác dụng định thời của CP.

Hai vòng tròn biểu thị hai trạng thái logic của FF. Mũi tên biểu thị hướng chuyển trạng thái. Bên cạnh mũi tên, ở trên gạch chéo là giá trị tín hiệu đầu vào kích - tức là điều kiện chuyển đổi trạng thái.

Hình 5-2-2 cho biết rằng :

Khi $Q^n = 0$	với $R = X, S = 0$	CP thì $Q^{n+1} = Q^n = 0$
	với $R = 0, S = 1$	CP thì $Q^{n+1} = 1$
Khi $Q^n = 1$	với $R = 0, S = X$	CP thì $Q^{n+1} = Q^n = 1$
	với $R = 1, S = 0$	CP thì $Q^{n+1} = 0$

d) Đồ thị thời gian dạng sóng

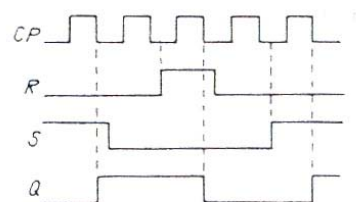
Đồ thị thời gian dạng sóng biểu thị trực quan quan hệ tương ứng nhau về mặt thời gian của các trạng thái FF, các tín hiệu đầu vào R, S và xung đồng hồ CP (hình 5-2-3).

Trên hình dạng sóng CP, R, S là đã biết. Để vẽ ra dạng sóng Q, ta cần chú ý :

- Nếu không cho trước thì có thể tùy ý giả định trạng thái ban đầu của Q.

- Căn cứ vào bảng chức năng, phương trình đặc trưng hoặc đồ hình trạng thái để xác định trạng thái Q tiếp theo.

- Sau khi xuất hiện sườn âm CP thì Q chuyển đổi trạng thái. Mọi lúc khác Q duy trì trạng thái cũ.



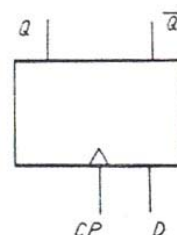
Hình 5-2-3 Đồ thị thời gian dạng sóng FFRS

Trên đây, 4 phương pháp biểu thị chức năng logic với Flip Flop (a, b, c, d) là liên quan mật thiết với nhau, có thể chuyển hóa lẫn nhau.

2. Flip Flop D

a) Định nghĩa

Flip Flop D là mạch điện có chức năng thiết lập trạng thái 0 theo tín hiệu đầu vào $D = 0$ và thiết lập trạng thái 1 theo tín hiệu đầu vào $D = 1$ trong điều kiện định thời của CP.



Hình 5-2-4 Kí hiệu logic FFD

Flip Flop D được giới thiệu ở tiết trên thỏa mãn định nghĩa này. Phương trình đặc trưng của FFD hình 5-2-4 là :

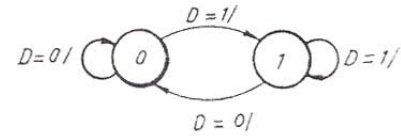
$$\begin{cases} Q^{n+1} = D \\ \text{với điều kiện đã xuất hiện sườn dương CP} \end{cases} \quad (5-2-2)$$

$D = 0$ CP thì $Q^{n+1} = 0$

$D = 1$ CP thì $Q^{n+1} = 1$

b) Bảng chức năng, bảng tín hiệu đầu vào kích, đồ hình trạng thái và đồ thị thời gian dạng sóng (bảng 5-2-3, 5-2-4 hình 5-2-5, 5-2-6).

Trạng thái đầu $Q = 0$, kích bằng sườn dương của CP. Dạng sóng CP và D là đã biết. Để vẽ dạng sóng Q, chú ý mức D khi CP.



Hình 5-2-5 Đồ hình trạng thái của FF D

Bảng 5-2-3 : BẢNG CHỨC NĂNG CỦA FF D

Q^n	D	Q^{n+1}
0	0	0
0	1	1
1	0	0
1	1	1

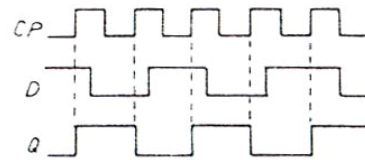
Bảng 5-2-4 : BẢNG ĐẦU VÀO KÍCH CỦA FF D

Q^n	$\rightarrow$	Q^{n+1}	Q^{n+1}
0	0	0	
0	1	1	
1	0	0	
1	1	1	

3. Flip Flop T

a) Định nghĩa :

Flip Flop T là mạch điện có chức năng duy trì và chuyển đổi trạng thái tùy thuộc tín hiệu đầu vào T trong điều kiện định thời của CP. Flip Flop JK giới thiệu ở tiết trên, nếu $J = K = T$ thì tạo thành FF. T ký hiệu như hình 5-2-7. Phương trình đặc trưng của FF T là :



Hình 5-2-6 Dạng sóng của FF D

$$\begin{cases} Q^{n+1} = J\bar{Q}^n + \bar{K}Q^n = T\bar{Q}^n + \bar{T}Q^n \\ = T \oplus Q^n \end{cases}$$

Vôùi ñiều kiện ñảo xuất hiện sôðòn âm CP

(5-2-3)

T = 0 CP thì $Q^{n+1} = Q^n$ duy trì nguyên trạng

T = 1 CP thì $Q^{n+1} = \bar{Q}^n$ chuyển đổi trạng thái.

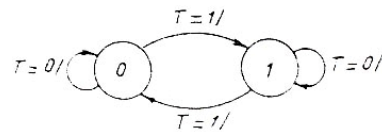
b) Bảng chức năng (bảng 5-2-5) bảng đầu vào kích (5-2-6) , đồ hình trạng thái (hình 5-2-8), đồ thị thời gian dạng sóng (hình 5-29) của Flip Flop T.

Bảng 5-2-5

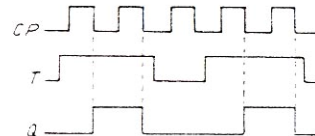
Q^n	T	Q^{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Bảng 5-2-6

$Q^n \rightarrow Q^{n+1}$	T
0 0	0
0 1	1
1 0	1
1 1	0



Hình 5-2-8



Hình 5-2-9

Trạng thái đầu của FF T là 0 và kích bằng sườn âm của CP.

4. Flip Flop T'

Flip Flop T' là mạch điện chỉ có chức năng chuyển đổi trạng thái trong điều kiện định thời của CP. FF T' là FF T mà T = 1 (T luôn luôn bằng 1)

Phương trình đặc trưng của Flip Flop T' là :

$$\begin{cases} Q^{n+1} = T \oplus Q^n = 1 \oplus Q^n = \bar{Q}^n \\ \text{Với điều kiện đã xuất hiện sườn âm CP} \end{cases}$$

Hình 5-2-10 : Đồ thị thời gian dạng sóng của Flip Flop T'

5. Flip Flop JK

a) Định nghĩa :

Flip Flop JK là mạch điện có chức năng thiết lập trạng thái 1, chuyển đổi trạng thái và duy trì trạng thái căn cứ vào các tín hiệu đồng hồ CP.

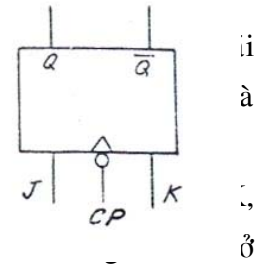
Trong kỹ thuật số thường yêu cầu FF có 4 chức năng nghĩa là FF JK rất vạn năng, rất linh hoạt. FFJK kích sườn master tiết 5-1 thỏa mãn định nghĩa này. Phương trình đặc trưng của FF trên hình 5-2-11 là :

$J = 0, K = 1$ CP thì $Q^{n+1} = 0$

$J = 1, K = 0$ CP thì $Q^{n+1} = 1$

$J = 1, K = 1$ CP thì $Q^{n+1} = \overline{Q}^n$ chuyển đổi

$J = 0, K = 0$ CP thì $Q^{n+1} = Q^n$ giữ nguyên trạng.



Hình 5-2-11.
Kí hiệu logic của FF JK

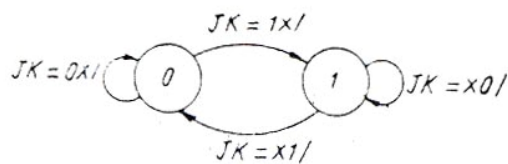
b) Bảng chức năng (5-2-7), bảng đầu vào kích (5-2-8), đồ hình trạng thái (hình 5-2-12), đồ thị thời gian dạng sóng (hình 5-2-13) của Flip Flop JK.

Bảng 5-2-7

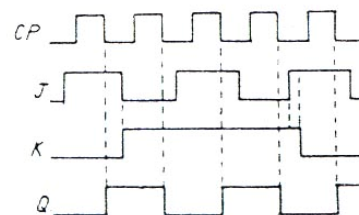
Q^n	J	K	Q^{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Bảng 5-2-8

$Q^n \rightarrow Q^{n+1}$	J	K
0 → 0	0	x
0 → 1	1	x
1 → 0	x	1
1 → 1	x	0



Hình 5-2-12



Hình 5-2-13.

Trên hình, trạng thái đầu của FF là 0, kích bằng sườn âm của CP. Trong ký hiệu logic của 5 loại FF trên, đầu CP có khuyên tròn chứng tỏ nó hoạt động với sườn âm, không có khuyên tròn chứng tỏ nó hoạt động với sườn dương. Khi vẽ đồ thị thời gian dạng sóng, ta phải đặc biệt chú ý quy ước này, chỉ khi đã xuất hiện sườn xung CP thì FF mới chuyển đổi trạng thái theo phương trình đặc trưng của nó, ở thời điểm khác FF giữ nguyên trạng thái.

5.2.2. Sự chuyển đổi lẫn nhau của các loại Flip Flop định thời theo CP

1. Phương pháp và ý nghĩa của sự chuyển đổi

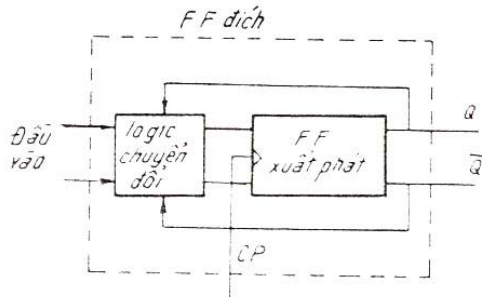
a) Ý nghĩa :

- Đa số FF trên thị trường là loại JK, D. Kỹ thuật số yêu cầu tất cả các loại FF. Nếu biết cách chuyển đổi thì có thể phát huy tác dụng của loại FF có sẵn.

- Phương pháp chuyển đổi có tính phổ biến, do đó giúp ích nhiều việc thiết kế mạch điện.

- Giúp đi sâu tìm hiểu chức năng logic của các loại FF.

b) Phương pháp :



Hình 5-2-14

Tư duy về chuyển đổi từ FF đích

bày viết, có thể dùng đại số logic xử lý, nhưng cần kỹ xảo nhất định ; trong phạm vi 5 loại FF cụ thể, chúng ta có thể nắm vững phương pháp này.

- Dùng sơ đồ : bảng chức năng, bảng đầu vào kích, bảng Karnaugh. Phương pháp này có phiền phức chút ít, nhưng trực quan, ít sai.

2. Flip Flop JK chuyển đổi thành Flip Flop D, T, RS

Phương trình đặc trưng của FF JK (Flip Flop xuất phát)

$$Q^{n+1} = J\bar{Q}^n + \bar{K}Q^n \quad (5-2-6)$$

a) JK → D

Phương trình đặc trưng của FF D (Flip Flop đích) :

$$Q^{n+1} = D \quad (5-2-7)$$

- Phương pháp chuyển đổi là những công việc cần làm để tìm logic chuyển đổi, để tìm phương trình hàm logic tín hiệu kích đối với FF xuất phát (hình 5-2-14).

- Dùng công thức : dùng các nghiệm phương trình đặc trưng để tìm logic chuyển đổi. Cách này tiện cho trình

Dùng công thức : chuyển dạng (5-2-7) sang dạng (5-2-6)

$$Q^{n+1} = D = D(\overline{Q}^n + Q^n) = D\overline{Q}^n + DQ^n$$

So sánh với (5-2-6) , ta có :

$$\begin{cases} J = K \\ K = \overline{D} \end{cases} \quad (5-2-8)$$

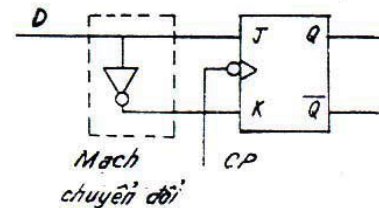
(5-2-8) là logic chuyển đổi cần tìm, cũng là phương trình đầu vào kích của FF JK. Dựa vào đó ta vẽ ra mạch điện hình 5-2-15.

Dùng sơ đồ :

Căn cứ vào bảng chức năng của FF D và bảng đầu vào kích của JK để liệt kê bảng sử dụng JK $\rightarrow$ D như sau :

Bảng 5-2-9 : BẢNG SỬ DỤNG JK $\rightarrow$ D

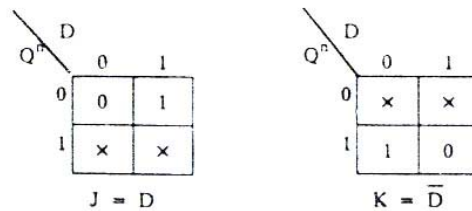
Q^n	D	Q^{n+1}	J	K
0	0	0	0	x
0	1	1	1	1
1	0	0	x	1
1	1	1	x	0



Hình 5-2-15
Mạch Flip Flop từ JK trở thành D

Bảng chức năng của FF D biểu thị yêu cầu chuyển đổi. Với Q^n , D đã xác định và có tác dụng định thời của CP thì Q^{n+1} cũng xác định theo bảng chức năng của FF D. Ở đây, Q^n và Q^{n+1} là trạng thái hiện tại và trạng thái tiếp theo của Flip Flop D (đích) và cũng là của FF JK (xuất phát). Vậy quan hệ tương ứng giữa Q^n và Q^{n+1} cũng phản ánh yêu cầu kích của FF JK, rồi căn cứ vào bảng đầu vào kích đó mà xác định giá trị tương ứng của J, K. Giá trị của Q^n và D quyết định giá trị của Q^{n+1} , do đó cũng quyết định giá trị của J, K. Vậy J, K và Q^{n+1} như nhau, đều là hàm số của Q^n và D. Quan hệ hàm số này dưới dạng bảng được gọi là bảng sử dụng.

Từ bảng sử dụng 5-2-9, ta vẽ bảng karnaugh và tìm được phương trình đầu vào kích, như hình 5-2-16. Kết quả của hai phương pháp (dùng công thức và dùng sơ đồ) trùng hợp.



Hình 5-2-16
Bảng Karnaugh của J,K

b) JK → T

Phương trình đặc trưng của FF T

$$Q^{n+1} = T\bar{Q}^n + \bar{T}Q^n$$

So sánh trực tiếp với (5-2-6) ta có phương trình kích của FF JK

$$\begin{cases} J = T \\ K = T \end{cases} \quad (5-2-9)$$

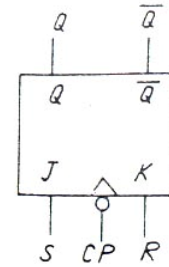
Mạch điện xem hình 5-2-7 ở phần trên. Cho T = 1 ta có FF T'

c) JK → RS

Phương trình đặc trưng của Flip Flop RS

$$\begin{cases} Q^{n+1} = S + \bar{R}Q^n \\ RS = 0 \end{cases}$$

Biến đổi :



Hình 5-2-17

Mạch Flip Flop từ J,K trở thành RS

$$\begin{aligned} Q^{n+1} &= RS + \bar{R}Q^n = S(\bar{Q}^n + Q^n) + \bar{R}Q^n \\ &= S\bar{Q}^n + SQ^n + \bar{R}Q^n = S\bar{Q}^n + \bar{R}Q^n + SQ^n(R + \bar{R}) \\ &= S\bar{Q}^n + \bar{R}Q^n + RSQ^n + \bar{R}SQ^n \\ &= S\bar{Q}^n + \bar{R}Q^n + RSQ^n = S\bar{Q}^n + \bar{R}Q^n \end{aligned}$$

. So sánh với phương trình đặc trưng của Flip Flop JK, ta có logic chuyển đổi :

$$\begin{cases} J = S \\ K = R \end{cases} \quad (5-2-10)$$

Sơ đồ logic : xem hình 5-2-17

3. Flip Flop D chuyển đổi thành

Flip Flop JK, RS, T, T'

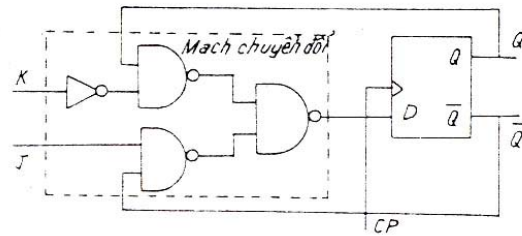
Phương trình đặc trưng của FF D

$$Q^{n+1} = D$$

$$D \rightarrow JK$$

$$JK : Q^{n+1} = J\bar{Q}^n + \bar{K}Q^n$$

$$\text{Đặt } D = J\bar{Q}^n + \bar{K}Q^n \quad (5-2-$$



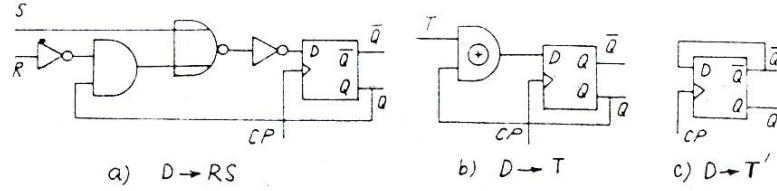
Hình 5-2-18

Mạch Flip Flop Từ D trở thành J,K

11)

Mạch điện hình 5-2-18

Hình 5-2-19 là mạch điện chuyển đổi theo logic chuyển đổi sau đây :



Hình 5-2-19
Mạch Flip Flop Từ D trở
thành RS,T,T'

$$D = S + \bar{R}Q^n \quad (5-2-12)$$

$$D = T\bar{Q}^n + \bar{T}Q^n = T \oplus Q^n \quad (5-2-13)$$

$$D = \bar{Q}^n \quad (5-2-14)$$

4. Flip Flop T chuyển đổi thành Flip Flop JK, D, RS

Phương trình đặc trưng

của Flip Flop T

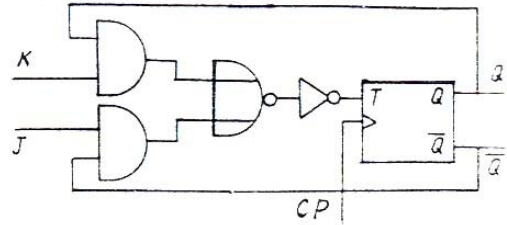
$$Q^{n+1} = T \oplus Q^n$$

$$T \rightarrow JK$$

$$JK : Q^{n+1} = J\bar{Q}^n + \bar{K}Q^n$$

Dùng các công thức của

hàm XOR (xem mục 3-1-2-7) ta biến đổi



Hình 5-2-20
Mạch Flip Flop Từ T trở
thành JK

$$\begin{aligned} T &= (J\bar{Q}^n + \bar{K}Q^n) \oplus Q^n \\ &= (J\bar{Q}^n + \bar{K}Q^n)\bar{Q}^n + \overline{(J\bar{Q}^n + \bar{K}Q^n)Q^n} \\ &= J\bar{Q}^n + \overline{J\bar{Q}^n} \cdot \overline{\bar{K}Q^n} \cdot \overline{Q^n} \\ &= J\bar{Q}^n + \overline{J\bar{Q}^n} \cdot KQ^n \\ &= J\bar{Q}^n + KQ^n \end{aligned} \quad (5-2-15)$$

Xem hình 5-5-20

JK Q ⁿ				
	00	01	11	10
0	0	0	1	1
1	0	1	1	0

Hình 5-2-21
Bảng Karnaugh của T

Bảng 5-2-10 : BẢNG SỬ DỤNG T → JK

Q^n	J	K	Q^{n+1}	T
0	0	0	0	0
0	0	1	0	0
0	1	0	1	1
0	1	1	1	1
1	0	0	1	0
1	0	1	0	1
1	1	0	1	0
1	1	1	0	1

Có thể đi đến kết quả trên theo phương pháp dùng sơ đồ như sau :

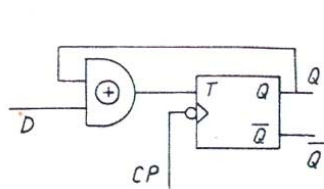
Xây dựng bảng sử dụng T → JK (bảng 5-2-10). Tìm logic chuyển đổi bằng bảng Karnaugh (hình 5-2-21).

Tương tự, ta tìm logic chuyển đổi T → D, T → RS như sau :

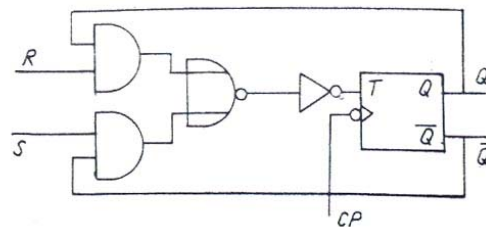
$$T = D \oplus Q^n \quad (5-2-16)$$

$$T = S\bar{Q}^n + RQ^n \quad (5-2-17)$$

Xem mạch điện chuyển đổi hình 5-2-22



a) T → D



a) T → RS

Hình 5-2-22

Mạch điện chuyển đổi Flip
Flop Từ D trở thành D.RS.

5 Flip Flop RS

chuyển đổi thành Flip Flop JK, D, T'

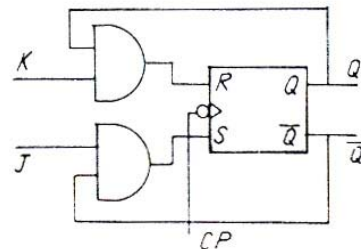
Phương trình đặc trưng của Flip

Flop RS :

$$\begin{cases} Q^{n+1} = S + \bar{R}Q^n \\ RS = 0 \end{cases}$$

RS → JK

$$JK : Q^{n+1} = J\bar{Q}^n + \bar{K}Q^n$$



Mạch Flip Flop Từ RS trở
thành JK.

So sánh , ta có :

$$\begin{cases} S = J\bar{Q}^n \\ R = K \end{cases}$$

Bảng 5-2-11 : BẢNG SỬ DỤNG RS $\rightarrow$ JK

Q^n	J	K	Q^{n+1}	R	S
0	0	0	0	x	0
0	0	1	0	x	0
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	1	0	x
1	0	1	0	1	0
1	1	0	1	0	x
1	1	1	0	1	0

Vì điều kiện ràng buộc RS = 0 nên ta phải kiểm tra. Khi J = K = 1 ; $Q^n = 0$ thì

$$\begin{cases} R = K = 1 \\ S = J\bar{Q}^n = 1 \end{cases}$$

Không thỏa mãn RS = 0 ta biến đổi lại :

$$\begin{aligned} Q^{n+1} &= J\bar{Q}^n + \bar{K}Q^n \\ &= J\bar{Q}^n + \overline{KQ^n}Q^n \end{aligned}$$

So sánh lại, ta có :

JK

Q^n

	00	01	11	10
0	x	x	1	1
1	0	1	1	0

(a)

JK

Q^n

	00	01	11	10
0	0	0	1	1
1	x	0	0	x

(b)

Hình 5-2-24

Bảng Karnaugh : a) R; b) S

$$\begin{cases} S = J\bar{Q}^n \\ R = KQ^n \end{cases}$$

(5-2-18)

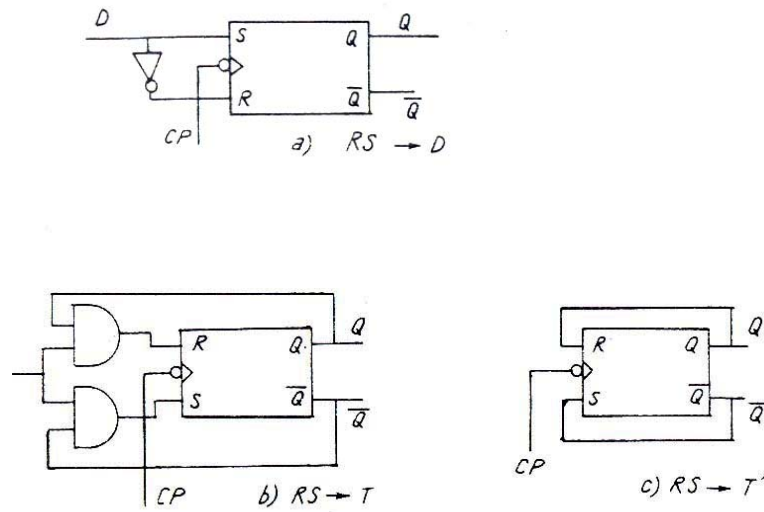
Hình 5-2-23 trên đây là mạch điện chuyển đổi RS $\rightarrow$ JK

(thỏa mãn điều kiện ràng buộc RS = 0).

Ta có thể nhận được cùng một kết quả như trên bằng phương pháp dùng sơ đồ dưới đây :

Xây dựng bảng sử dụng RS $\rightarrow$ JK (bảng 5-2-11). Tìm logic chuyển đổi bằng bảng Karnaugh (hình 5-2-24). Chú ý điều kiện ràng buộc RS = 0 khi xây dựng bảng 5-2-11.

Tương tự, ta tìm logic chuyển đổi :



Hình 5-2-25
Mạch điện chuyển đổi Flip Flop Từ RS
trở thành a) D; T;c) T'

$$RS \rightarrow D \begin{cases} R = \bar{D} \\ S = D \end{cases} \quad (5-2-19)$$

$$RS \rightarrow T \begin{cases} R = TQ^n \\ S = T\bar{Q}^n \end{cases} \quad (5-2-20)$$

$$RS \rightarrow T' \begin{cases} R = Q^n \\ S = \bar{Q}^n \end{cases} \quad (5-2-21)$$

5.3. ĐẶC TÍNH CÔNG TÁC XUNG VÀ CHỈ TIÊU CHỦ YẾU CỦA FLIP FLOP

5.3.1. Đặc tính công tác xung của Flip Flop

Muốn sử dụng chính xác FF, không những cần hiểu chức năng logic của FF, mà còn cần nắm vững đặc tính công tác xung của FF, tức là những yêu cầu mà FF đưa ra cho xung đồng hồ, tín hiệu đầu vào và sự phối hợp giữa chúng.

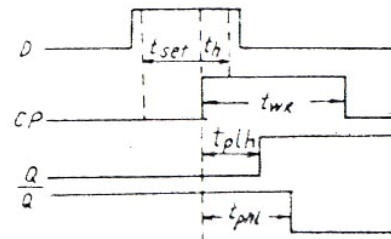
1. Đặc tính công tác xung của Flip Flop D

Xét mạch FF D trên hình 5-1-14 (mục 5-1-6) trước khi xuất hiện xung đồng hồ thì mạch điện ở trạng thái chuẩn bị. Lúc này, mức tín hiệu đầu vào D quyết định mức đầu ra của các cổng E, F. Khi xuất hiện sườn trước xung đồng hồ, trạng thái đầu ra của các cổng E, F. Khi xuất hiện sườn trước xung đồng hồ, trạng thái đầu ra của các cổng E, F thông qua các cổng C, D điều khiển FF chuyển trạng thái. Vậy mức đầu ra E, F phải đạt đến trạng thái ổn định trước lúc xuất hiện sườn trước xung đồng hồ. Nhưng từ lúc bắt đầu có tín hiệu ở đầu vào D đến lúc đầu ra các cổng E, F đã ổn định phải qua một khoảng thời gian, gọi là thời gian xác lập T_{set} của FF. Vậy tín hiệu đầu vào phải đến sớm khoảng t_{set} so với xung đồng hồ CP.

Từ hình 5-1-14 ta thấy rằng kể từ tín hiệu đến đầu vào D, thì đầu ra F ổn định phải sau thời gian trễ truyền đạt 1 cấp cổng t_{pd} , đầu ra E ổn định phải sau thời gian trễ truyền đạt 2 cấp cổng $2t_{pd}$, vậy thời gian xác lập của Flip Flop D là $t_{set} = 2t_{pd}$. Tiếp theo, từ khi xuất hiện sườn trước CP cho đến khi FF hoàn thành chuyển đổi, tức là đến khi trạng thái mới đã ổn định, là khoảng thời gian trễ truyền đạt của FF (t_{pd}). Giả sử trạng thái ban đầu của FF là 1, mức đầu ra cổng E là thấp, mức đầu ra cổng F là cao. Thế là dễ dàng thấy rằng, xung đầu ra cổng D chậm sau CP thời gian trễ truyền đạt 1 cấp cổng làm cho cổng B từ thông sang ngắt với thời gian trễ truyền đạt bản thân cổng B. Cổng A cũng chuyển từ ngắt sang thông với thời gian trễ truyền đạt bản thân cổng A. Vậy đầu ra $\bar{Q}$, chuyển trạng thái từ mức thấp lên cao có thời gian trễ t_{plh} tương đối ngắn hơn so với thời gian đầu ra Q chuyển trạng thái từ mức cao xuống thấp $t_{plh} = 2t_{pd}$, $t_{phl} = 3t_{pd}$).

Ngoài ra, để bảo đảm FF chuyển đổi tin cậy, tín hiệu đầu vào cần có thời gian tác dụng đủ dài. Khi D = 0, kể từ khi bắt đầu sườn dương của CP, phải sau t_{pd} thì cổng D mới đưa ra mức thấp ; mức logic này phản hồi đến đầu vào cổng F mới sinh ra tác dụng duy trì nguyên trạng ngăn trở chuyển đổi. Trước lúc đạt đến sự ổn định này, tín hiệu đầu vào không được phép thay đổi, nếu ngược lại, trạng thái mới của tín hiệu đầu vào có thể phá hoại sự chuyển đổi bình thường vốn có. Vậy sau khi xuất hiện sườn trước CP, tín hiệu đầu vào cần phải duy trì thêm một thời gian, gọi là thời gian duy trì t_h . Trong trường hợp D = 0 thì $t_h = t_{pd}$.

Căn cứ vào thời gian trễ đầu ra FF chuyển trạng thái t_{phl} và thời gian xác lập t_{set} ta có thể biết độ rộng cần thiết của xung đồng hồ. độ rộng (theo mức cao) t_{WH} của CP phải lớn hơn t_{phl} để phần mạch RS cơ bản trong FF D chuyển



Hình 5-3-1
Dạng sóng của FFD.
(theo mạch hình 5-1-14)

đổi tín hiệu. Độ rộng (theo mức thấp) t_{WL} của CP phải lớn hơn t_{set} để bảo đảm tín hiệu đầu vào đủ thời gian ổn định đầu ra cổng E, F trước khi xuất hiện sườn dương của CP. Vậy chu kỳ xung CP phải lớn hơn $t_{phl} + t_{set}$ tức là tần số cực đại của CP bị hạn chế nhỏ hơn $\frac{1}{t_{phl} + t_{set}}$. Ví dụ, $t_{ip} = 20ns$, $t_{phl} + t_{set} = 5t_{pd} = 100ns$, $f_{max} = 10MHz$.

Xem hình 5-3-1 biểu thị quan hệ thời gian vừa trình bày trên đây.

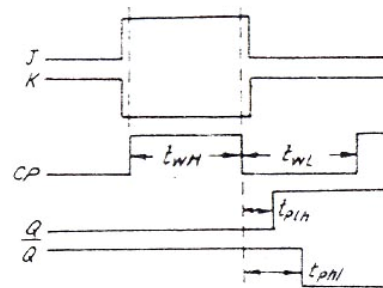
2. Đặc tính công tác xung của FF master slave

Xét mạch FF hình 5-1-11 (mục 5-1-4).

Khi xuất hiện sườn dương CP thì tín hiệu đầu vào J, K tác động đến master. Vì J, K, CP đồng thời nối đến các cổng E, H nên tín hiệu đầu vào chỉ cần xuất hiện không chậm hơn sự xuất hiện sườn dương CP, vậy $t_{set} = 0$.

Sau khi xuất hiện sườn dương CP.

Sự chuyển đổi của master chỉ hoàn thành sau thời gian trễ truyền đạt 2 cấp cổng NORAND. Nếu thời gian trễ truyền đạt của cổng NORAND bằng 1,4 lần của cổng NAND thì độ rộng (theo mức cao) t_{WH} của xung đồng hồ CP cần thiết: $t_{WH} \geq 2,8t_{pd}$.



Hình 5-3-2
Dạng sóng của FFJK masterslave
(theo mạch hình 5-1-11)

Sau khi xuất hiện sườn âm CP,

Slave chuyển đổi đầu vào master bị khóa, nên tín hiệu đầu vào J, K có thể không cần duy trì, tức là $t_h = 0$.

Kể từ khi bắt đầu sườn âm CP, cho đến khi ổn định trạng thái Q, $\bar{Q}$ là thời gian trễ truyền đạt. Vì mạch điện các cổng C, D rất đơn giản, ta có thể cho rằng thời gian trễ truyền đạt của chúng bằng nửa của cổng NAND. Vậy $t_{phl} = 1,5t_{pd}$, $t_{phl} = 2,5t_{pd}$. Rõ ràng yêu cầu độ rộng (theo mức thấp) t_{WL} của CP cần thiết là $t_{WL} \geq t_{phl}$.

Do đó, yêu cầu đối với xung đồng hồ :

$$T_{min} = 2,8t_{pd} + 2,5t_{pd} = 5,3t_{pd}$$

$$f_{max} = \frac{1}{5,3t_{pd}}$$

Hình 5-3-2 biểu thị quan hệ thời gian.

5.3.2. Các chỉ tiêu chủ yếu của vi mạch (IC) Flip Flop

1. Tham số tĩnh

Kết cấu mạch điện đầu vào, đầu ra của Flip Flop dưới dạng IC rất giống với các cổng NAND họ TTL. Những đặc tính đầu vào đầu ra cũng vậy. Nên cách định nghĩa và phương pháp đo lường các tham số chủ yếu của đặc tính đầu vào, đầu ra cũng cơ bản giống như của cổng NAND TTL. Các chỉ tiêu chủ yếu là :

Dòng điện nguồn I_E

Trong mạch điện cổng dòng điện nguồn khi mức đầu ra thấp và khi mức đầu ra cao khác nhau rất xa. Trong mạch điện Flip Flop bao gồm rất nhiều cổng, thường không đồng thời thông cả hoặc đồng thời ngắt cả, nên dù trạng thái mạch điện FF thay đổi thì dòng điện nguồn cũng không thay đổi đáng kể. Vậy thông thường chỉ đưa ra một giá trị dòng điện nguồn và quy định rằng tất cả các đầu vào đều phải treo khi đo lường dòng điện nguồn.

Dòng điện ngắn mạch đầu vào I_{IS}

Lần lượt nối đất các đầu vào, ta đo được dòng điện ngắn mạch đầu vào tương ứng. Có thể nhận thấy ở các hình 5-1-11 và 5-1-15 rằng mỗi một trong các đầu vào R, S, R_d , S_d , CP, D, J, K v.v... được nối với một số khác nhau tranzito nhiều emitơ. Do đó dòng điện ngắn mạch đầu vào của chúng cũng không bằng nhau. Số tranzito càng nhiều thì trị số dòng điện tương ứng càng lớn.

Dòng điện dò đầu vào I_{IH}

Là dòng điện chảy vào đầu vào xét khi đầu vào đó nối đến mức cao. Dòng điện I_{IH} cũng phụ thuộc vào số tranzito nhiều emitơ được nối đến đầu vào xét.

Mức tín hiệu đầu ra cao V_{OH} và thấp V_{OL}

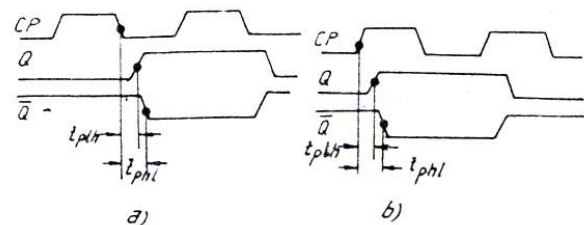
Khi FF ở trạng thái 1, đầu ra Q ở mức cao, $\bar{Q}$ ở mức thấp. Khi FF ở trạng thái 0, đầu ra Q ở mức thấp, $\bar{Q}$ ở mức cao. Vậy chỉ cần đo lường riêng mức đầu ra Q, $\bar{Q}$ khi FF ở trạng thái 1 và 0 là ta được V_{OH} và V_{OL} .

2. Tham số động

Có 2 tham số động thường dùng như sau :

Thời gian trễ truyền đạt trung bình t_{pd}

t_{pd} được định nghĩa là thời gian từ khi sườn xung đồng hồ tác động (ví dụ, sườn âm của CP đối với FF JK master slave, sườn dương của CP đối với FF D) đến khi trạng thái mới tạo ra ở đầu ra của FF đã ổn định (hình 5-3-3).



Hình 5-3-3 :

Thời gian trễ truyền đạt của IC Flip Flop

a) Kích với sườn âm

b) Kích với sườn dương

Thông thường thời gian trễ truyền đạt của đầu ra từ mức cao xuống mức thấp t_{phl} lớn hơn từ mức thấp lên mức cao t_{plh} . Trong các sổ tay IC, người ta chỉ cho biết giá trị trung bình t_{pd} .

$$t_{pd} = \frac{t_{plh} + t_{phl}}{2}$$

Khi FF nối thành T' thì tần số đồng hồ cao nhất cho phép f_{max} . Khi đo lường f_{max} , ta cần tiến hành với phụ tải định mức đã cho, vì kết quả sẽ phụ thuộc vào tình trạng phụ tải. Bảng 5-3-1 và 5-3-2 giới thiệu các chỉ tiêu chủ yếu của IC FF Z63 và IC FF D62.

Bảng 5-3-1 : CHỈ TIÊU CHỦ YẾU CỦA FF JK MASTER SLAVE IC Z63B

$T = 25^0$, $E_C = 5V$

Tham số		Ký hiệu	Đơn vị	Điều kiện đo	Chỉ tiêu
Dòng điện nguồn		I_E	mA	Đầu vào hở mạch Đầu ra không tải	≤ 15
Dòng điện ngắn mạch đầu vào	J, K	I_{IS}	mA	$V_I = 0$	$\leq 1,5$
	R, S, CP				$\leq 1,5$
Dòng điện dò đầu vào	J, K	I_{IH}	$\square A$	$V_I = 5V$	≤ 20
	R, S			Các đầu vào khác nối đất	≤ 60
	CP				≤ 80
Mức cao đầu ra	V_{OH}	V	$\square_L = 160 \square A$ $I_L = 12mA$ $I_L = 12mA$ $C_L = 15pF$		$3 \div 4$
Mức thấp đầu ra	V_{OL}	V			$\leq 0,35$
Tần số đồng hồ cao nhất	f_{max}	MHz			≤ 10

CHƯƠNG 6

MẠCH DÂY

6.1. ĐẠI CƯƠNG VỀ MẠCH DÂY

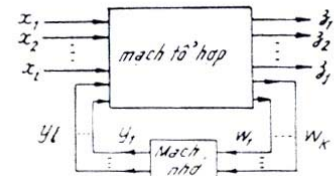
Căn cứ đặc điểm khác nhau về chức năng logic và cấu trúc mạch điện, mạch số được phân loại thành mạch tổ hợp (combinational Circuits) đã giới thiệu ở chương 4 và mạch dây (Sequential circuits) sẽ được trình bày trong chương này.

6.1.1. Đặc điểm và phương pháp miêu tả chức năng

1. Đặc điểm của mạch dây

Trong mạch số, một mạch điện được gọi là mạch dây nếu trạng thái đầu ra ổn định ở thời điểm xét bất kỳ không chỉ phụ thuộc vào trạng thái đầu vào thời điểm đó mà còn phụ thuộc cả vào trạng thái bản thân mạch điện ở thời điểm trước (trạng thái trong).

Mạch dây có đặc điểm nhất định phải bao gồm Flip Flop để nhớ trạng thái vốn có. Tư tưởng này thể hiện thành cấu trúc mạch như hình 6-1-1 gợi ý.



Hình 6-1-1
Sơ đồ khối mạch dây

2. Phương pháp miêu tả chức năng logic của mạch dây

Theo định nghĩa mạch dây trên đây, các Flip Flop đã được nghiên cứu ở chương 5 cũng là mạch dây, vì trạng thái đầu ra tiếp theo Q^{n+1} không chỉ phụ thuộc vào tín hiệu đầu vào mà còn phụ thuộc cả vào trạng thái (trong) Q^n vốn có. Phương pháp miêu tả chức năng logic của Flip Flop cũng thích hợp với mạch dây nói chung.

a) Phương trình logic

Xét hình 6-1-1, $X(x_1, x_2, \dots, x_i)$ là tín hiệu đầu vào ở thời điểm xét t_n , $Z(z_1, z_2, \dots, z_j)$ là tín hiệu đầu ra ở t_n , $W(w_1, w_2, \dots, w_k)$ là tín hiệu đầu vào mạch nhớ ở t_n (tức là tín hiệu kích đồng bộ của FF), $Y(y_1, y_2, \dots, y_e)$ là tín hiệu đầu ra mạch nhớ ở t_n (tức trạng thái hiện tại của FF). Quan hệ giữa các tín hiệu trên đây có thể biểu thị bằng các hàm logic :

$$Z(t_n) = F[X(t_n), Y(t_n)] \quad (6-1-1)$$

$$Y(t_{n+1}) = F[X(t_n), Y(t_n)] \quad (6-1-2)$$

$$W(t_n) = H[X(t_n), Y(t_n)] \quad (6-1-3)$$

t_n và t_{n+1} là hai thời điểm gần nhau. Vì $y_1, y_2, \dots, y_e$ là trạng thái của FF cấu trúc mạch nhớ, nên chúng được gọi là tín hiệu trạng thái, hay biến trạng thái, tương ứng hàm Y được gọi là vectơ trạng thái, (6-1-2) là phương trình trạng thái với $Y(t_{n+1})$ là

trạng thái tiếp theo, $Y(t_n)$ là trạng thái hiện tại. Tương tự (6-1-1) là phương trình đầu ra, (6-1-3) là phương trình kích. Nói riêng trường hợp Flip Flop, $X(t_n) = W(t_n)$, $Z(t_n) = Y(t_n)$. Vì vậy, chỉ riêng phương trình trạng thái (6-1-2) cũng đủ miêu tả chức năng logic của FF. Để phân biệt với mạch dây nối chung, ở chương 5, (6-1-2) của FF mang một tên chuyên biệt là phương trình đặc trưng như ta đã gọi.

b) Bảng trạng thái :

Bảng liệt kê mối quan hệ giữa $Z(t_n)$, $Y(t_{n+1})$ và $X(t_n)$, $Y(t_n)$ gọi là bảng trạng thái của mạch dây. (Riêng đối với FF, bảng trạng thái có tên riêng là bảng chức năng).

c) Đồ hình trạng thái :

Đồ hình trạng thái là hình vẽ phản ánh quy luật chuyển đổi trạng thái và tình hình các giá trị đầu vào, đầu ra tương ứng của mạch dây.

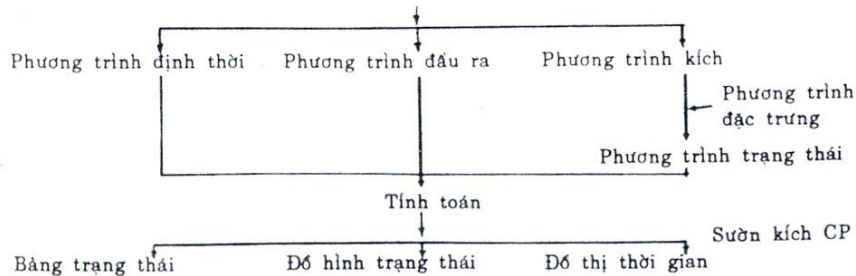
d) Đồ thị thời gian :

Đồ thị thời gian còn gọi là dạng sóng công tác. Nó biểu thị trực quan mối quan hệ tương ứng các giá trị tín hiệu đầu vào, tín hiệu đầu ra, trạng thái mạch điện về thời gian.

Như sau này chúng ta sẽ rõ, 4 phương pháp trình bày trên đây về bản chất đều phản ánh chức năng logic mạch dây theo các khía cạnh khác nhau, chúng liên quan và chuyển đổi lẫn nhau. Có thể tùy ý chọn dùng tùy theo tình hình cụ thể. Cũng cần lưu ý thêm rằng bảng Karnaugh có thể miêu tả chức năng logic mạch dây một cách tiện lợi.

6.1.2. Phương pháp cơ bản phân tích chức năng logic mạch dây

Nhiệm vụ phân tích là tìm ra bảng trạng thái, đồ hình trạng thái, đồ thị thời gian của mạch dây đã cho, rồi xác định đặc điểm công tác và chức năng logic của nó. Hình 6-1-2 là sơ đồ gợi ý về quá trình phân tích này.



Hình 6-1-2 Sơ đồ gợi ý quá trình phân tích mạch dây

Dưới đây đưa ra quy trình phân tích 4 bước :

1. Viết Phương trình :

Căn cứ vào mạch điện đã cho, viết phương trình định thời, phương trình đầu ra, phương trình kích, cũng tức là các công thức logic của tín hiệu định thời (đồng hồ) tín hiệu đầu ra và tín hiệu đầu vào.

2. Tìm phương trình trạng thái :

Thay phương trình kích vào phương trình đặc trưng của Flip Flop tương ứng, ta sẽ tìm được phương trình trạng thái của mạch điện, cũng tức là phương trình trạng thái tiếp theo của các Flip Flop. Vì rằng trạng thái mạch dãy bất kỳ đều nhờ các Flip Flop cấu trúc tạo ra mạch dãy đó mà có được khả năng nhớ.

3. Tính toán :

Đưa tất cả các tổ hợp có thể của trạng thái hiện tại và tín hiệu đầu vào phương trình trạng thái và phương trình đầu ra, rồi tiến hành tính toán, tìm ra trạng thái tiếp theo và tín hiệu đầu ra tương ứng. Ở đây có 4 điều chú ý :

- Điều kiện định thời tích cực của phương trình trạng thái.
- Trạng thái hiện tại của mạch điện, tức là tổ hợp các trạng thái hiện tại của FF cấu trúc nên mạch xét.
- Không bỏ sót một tổ hợp có thể nào của trạng thái hiện tại và tín hiệu đầu vào.
- Căn cứ vào giá trị ban đầu đã cho (hoặc tự cho) của trạng thái hiện tại và tín hiệu đầu vào mà tính toán lần lượt các trạng thái tiếp theo nhau.

4. Vẽ đồ hình trạng thái (hoặc bằng trạng thái, hoặc đồ thị thời gian). Xem xét kết quả tính toán, rồi vẽ đồ hình trạng thái. Ở đây cần chú ý 3 điều :

- Chuyển đổi trạng thái từ hiện tại đến tiếp theo, chứ không phải là từ hiện tại đến hiện tại, hoặc từ tiếp theo đến tiếp theo.
- Tín hiệu đầu ra là hàm số của trạng thái hiện tại, chứ không phải là hàm số của trạng thái tiếp theo.
- Vẽ đồ thị thời gian cần lưu ý rằng FF chỉ chuyển đổi trạng thái tương ứng với sườn kích của xung đồng hồ xuất hiện.

Quy trình 4 bước trên đây là chung, không bắt buộc phải tuân theo máy móc, mà nên vận dụng linh hoạt tùy tình huống cụ thể.

6.2. BỘ ĐẾM

6.2.1. Đặc điểm và phân loại bộ đếm

1. Đặc điểm cơ bản

Đếm là khả năng nhớ được số xung đầu vào ; mạch điện thực hiện thao tác đếm được gọi là bộ đếm.

Đếm là một thao tác cơ bản cực kỳ quan trọng. Vậy bộ đếm được sử dụng vô cùng rộng rãi, từ các thiết bị đo chỉ thị số đến các máy tính điện tử số loại lớn, bất kỳ hệ thống số hiện đại nào đều hiện diện bộ đếm.

2. Phân loại

Căn cứ vào sự khác biệt của tình huống chuyển đổi trạng thái của các Flip Flop trong bộ đếm, người ta phân thành hai loại lớn : bộ đếm đồng bộ và bộ đếm dị bộ. Trong bộ đếm đồng bộ, các Flip Flop đều chịu tác động điều khiển của một xung đồng hồ duy nhất, đó là xung đếm đầu vào. Vậy sự chuyển động trạng thái của chúng là đồng bộ. Bộ đếm dị bộ thì khác, có Flip Flop chịu tác động điều khiển trực tiếp của xung đếm đầu vào, nhưng cũng có FF chịu tác động điều khiển của xung đầu ra của FF khác (Có vai trò định thời của xung đồng hồ). Vậy sự chuyển đổi trạng thái của các FF không cùng lúc, tức là dị bộ.

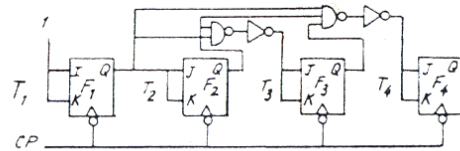
Căn cứ vào sự khác biệt về hệ số đếm của bộ đếm, người ta phân thành các loại : Bộ đếm nhị phân bộ đếm thập phân, bộ đếm N phân. Nếu gọi n là số chữ số vị trí trong mã số nhị phân, (cũng tức là số Flip Flop có trong bộ đếm), gọi N là số trạng thái tích cực (cũng tức là số trạng thái mã hóa đã được dùng khi lập mã) thì đối với bộ đếm nhị phân $N = 2^n$, đối với bộ đếm thập phân $N = 10$. Bộ đếm nhị phân và bộ đếm thập phân là trường hợp riêng của bộ đếm N phân. Ta thường gọi N là dung lượng bộ đếm hoặc độ dài đếm bộ đếm, hoặc hệ số đếm .

6.2.2. Bộ đếm đồng bộ

1. Bộ đếm nhị phân đồng bộ

Bộ đếm nhị phân đồng bộ nói chung cấu trúc bằng Flip Flop T.

a) Bộ đếm thuận nhị phân đồng bộ



Cấu trúc mạch :

(Hình 6-2-1) Bộ đếm nhị phân đồng bộ 4 chữ số

Hình 6-2-1 dưới đây là bộ đếm

thuận nhị phân đồng bộ 4 chữ số.

Bộ đếm cấu trúc bằng 4FFJK nối thành loại T và 4 cổng NAND, CP là xung đếm đầu vào ; chỉ có các đầu ra Q, $\bar{Q}$ của FF.

Nguyên lý làm việc :

Viết phương trình :

- Phương trình định thời $CP_1 = CP_2 = CP_3 = CP_4 = CP$ (6-2-1)

Xung đồng bộ của 4 FF đều là xung đếm đầu vào. Trong mạch dãy đồng bộ, các xung đồng hồ của các FF đều giống nhau, mỗi khi mạch điện chuyển đổi trạng thái, các điều kiện định thời nói chung đều đảm bảo, vậy nên phương trình định thời thường không cần viết ra.

- Phương trình đầu ra có thể không cần viết, vì không có tín hiệu đầu ra nào khác ngoài Q , $\overline{Q}$.

- Phương trình kích :

$$\begin{cases} T_1 = 1 \\ T_2 = Q_1^n \\ T_3 = Q_1^n Q_2^n \\ T_4 = Q_1^n Q_2^n Q_3^n \end{cases} \quad (6-2-2)$$

Tìm phương trình trạng thái :

Phương trình đặc trưng của Flip Flop T $Q^{n+1} = T\overline{Q}^n + \overline{T}Q^n$

Thay phương trình kích vào phương trình đặc trưng, ta có :

$$\begin{cases} Q_1^{n+1} = T_1 \overline{Q}_1^n + \overline{T}_1 Q_1^n = 1\overline{Q}_1^n + 1Q_1^n = \overline{Q}_1^n \\ Q_2^{n+1} = T_2 \overline{Q}_2^n + \overline{T}_2 Q_2^n = Q_1^n \overline{Q}_1^n + Q_1^n \overline{Q}_2^n = \overline{Q}_1^n Q_2^n \\ Q_3^{n+1} = T_3 \overline{Q}_3^n + \overline{T}_3 Q_3^n = Q_1^n Q_2^n \overline{Q}_3^n + Q_1^n Q_2^n Q_3^n \\ Q_4^{n+1} = T_4 \overline{Q}_4^n + \overline{T}_4 Q_4^n = Q_1^n Q_2^n Q_3^n \overline{Q}_4^n + Q_1^n Q_2^n Q_3^n Q_4^n \end{cases} \quad (6-2-2)$$

Tính toán :

Giả định các trạng thái $Q_4^n Q_3^n Q_2^n Q_1^n$ tuần tự, thay vào phương trình trạng thái (6-2-3) ta được bảng kết quả 6-2-1, đó là các trạng thái tiếp theo $Q_4^{n+1} Q_3^{n+1} Q_2^{n+1} Q_1^{n+1}$.

Bảng 6-2-1 : KẾT QUẢ TÍNH TOÁN

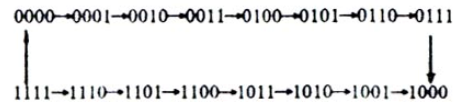
Q_4^n	Q_3^n	Q_2^n	Q_1^n	Q_4^{n+1}	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1

1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

Vẽ đồ hình trạng thái.

Từ bảng 6-2-1 ta vẽ ngay được đồ hình trạng thái, đó là hình 6-2-2.

Nhận xét hình 6-2-2 ta thấy mạch điện 6-2-1 làm việc của bộ đếm nhị phân đồng bộ 4 chữ số.



Hình 6-2-2 : Đồ hình trạng thái của bộ đếm

Đặc điểm :

Đặc điểm của bộ đếm đã nêu rõ trong tên “Bộ đếm nhị phân đồng bộ 4 chữ số” Dưới đây trình bày thêm về sự chuyển vị của bộ đếm.

Chuyển vị song song :

Mạch điện 6-2-1 thực hiện chuyển vị song song. Khái niệm chuyển vị đã dùng trong bộ cộng đủ, tổng của 1 với 1 là 0, với chuyển vị (nhớ) lên số có trọng số lớn hơn là 1. Từ góc độ phép cộng số 1 vào số có trọng số bé nhất của số nhị phân, đồng thời tuần tự chuyển vị lên số có trọng số lớn hơn, kích FF tương ứng chuyển đổi trạng thái. Phương thức chuyển vị song song của mạch hình 6-2-1 thể hiện ở chỗ tín hiệu chuyển vị (nhớ) từ đầu ra Q của một FF đưa đến đầu vào của FF trọng số lớn bất kỳ đều chỉ có trễ truyền đạt hai cấp cổng $2t_{pd}$. Xét thêm yêu cầu CP duy trì mức cao t_{WH} , yêu cầu thời gian chuyển của FF t_{phl} , thì chu kỳ đếm ngắn nhất có thể của bộ đếm (tức là khoảng thời gian cực tiểu giữa hai sườn âm xung đếm liên kế) :

$$T_{MIN} = 2t_{pd} + t_{phl} + t_{WH} \quad (6-2-4)$$

Vậy tần số cao nhất của bộ đếm là :

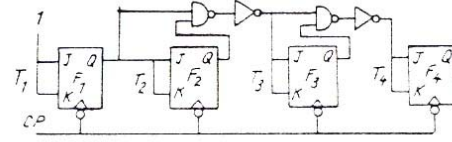
$$f_{MAX} = \frac{1}{T_{MIN}} \quad (6-2-5)$$

Nhược điểm của phương thức chuyển vị song song là phụ tải các FF không đều nhau, các FF trọng số càng bé thì phụ tải càng nặng, còn các FF trọng số càng lớn càng lăm đầu vào chuyển vị.

Chuyển vị nối tiếp

Xét mạch điện hình 6-2-3

Phương thức chuyển vị nối tiếp không có khuyết điểm nói trên của phương thức chuyển vị song song, tuy vậy, thời gian chuyển vị từ số trọng số bé nhất



Hình 6-2-3 Bộ đếm thuận đồng bộ chuyển vị nối tiếp

đến số trọng số lớn nhất lại kéo dài hơn.

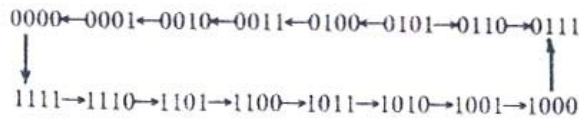
Giả sử có n trọng số, thời gian chuyển vị đó là thời gian tín hiệu chuyển vị đi qua $2(n-2)$ cổng để từ FF trọng số bé nhất đến FF trọng số lớn nhất. Do đó, tần số cực đại của bộ đếm chuyển vị nối tiếp tương đối thấp :

$$f_{\max} = \frac{1}{2(n-2)t_{pd} + t_{phl} + t_{wh}} \quad (6-2-6)$$

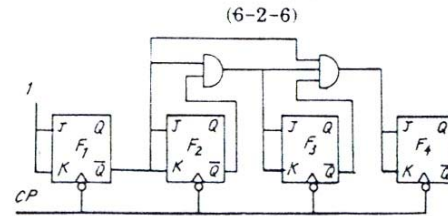
b) Bộ đếm nghịch nhị phân đồng bộ

Sơ đồ logic Hình 6-2-4

Tương tự như trên, dùng phương pháp phân tích căn bản để tìm đồ hình trạng thái, hình 6-2-5.



Hình 6-2-5 Đồ hình trạng thái của bộ đếm nghịch

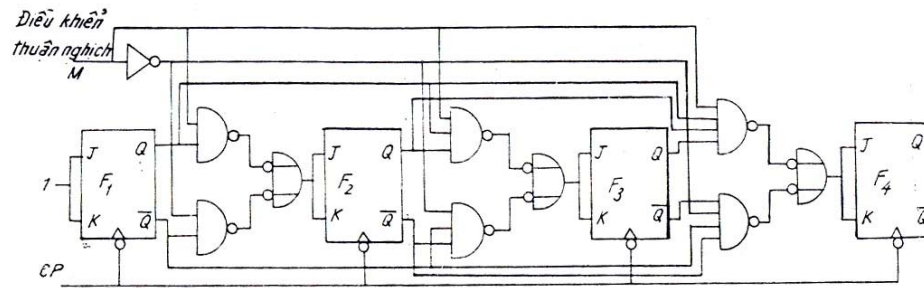


Hình 6-2-4 Bộ đếm nghịch nhị phân chuyển vị song song đồng bộ 4 chữ số.

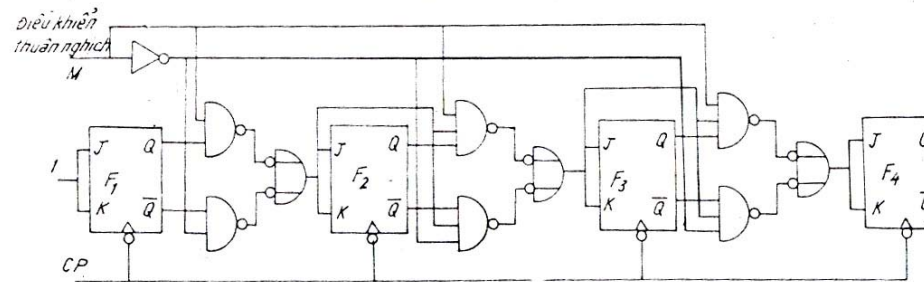
So sánh hình 6-2-4 với hình 6-2-1, ta thấy sự khác biệt của bộ đếm nghịch so với bộ đếm thuận là đầu ra $\bar{Q}$ (đảo) của FF cung cấp tín hiệu chuyển vị. Vậy từ bộ đếm thuận sẵn có, nếu ta tháo dây nối ở đầu Q , rồi đấu nối vào đầu $\bar{Q}$, ta sẽ có bộ đếm nghịch.

c) Bộ đếm thuận nghịch nhị phân đồng bộ

Kiểu mạch có đầu vào điều khiển đếm thuận, đếm nghịch.



Hình 6-2-6a) Chuyển vị song song.



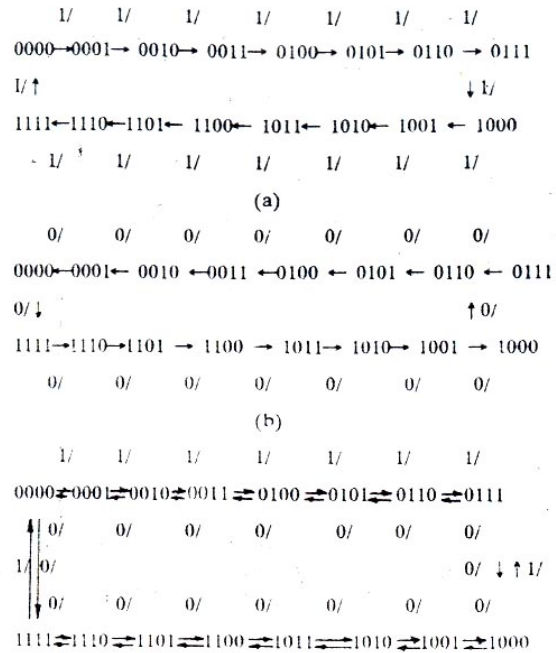
Hình 6-2-6 a) Chuyển vị song song

Hình 6-2-6 b) Chuyển vị nối tiếp

Hình 6-2-6 bộ đếm nghịch nhị phân đồng bộ có đầu vào điều khiển xét sơ đồ hình 6-2-6. Ta thấy bộ đếm gồm bộ đếm thuận và bộ đếm nghịch gộp lại với nhau, có thêm một số cổng điều khiển. Tín hiệu điều khiển đếm thuận hoặc đếm nghịch thông qua các cổng điều khiển để thực hiện sự điều khiển biến bộ đếm thành đếm thuận hay đếm nghịch. Sử dụng phương pháp phân tích logic, ta tìm đồ hình trạng thái như hình 6-2-7.

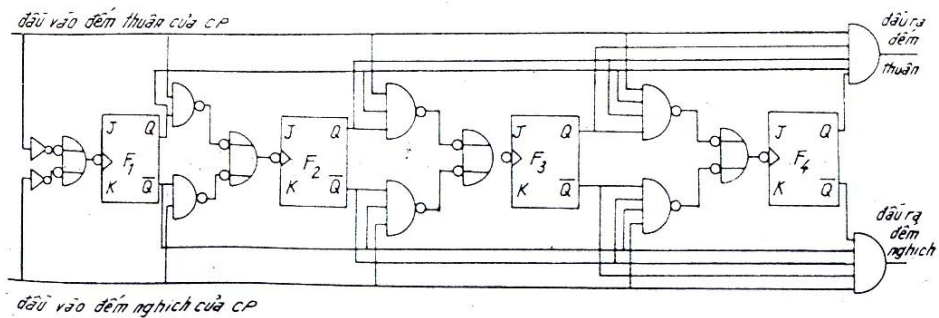
Bộ đếm trên đây, ngoài đầu vào đếm CP ra còn có đầu vào điều khiển đếm thuận hoặc đếm nghịch. Vậy nó được gọi là đếm thuận nghịch nhị phân đồng bộ kiểu có đầu vào điều khiển thuận hoặc nghịch.

Trong đồ hình trạng thái, con số trên dấu gạch chéo biểu thị trị số tín hiệu M điều khiển thuận / nghịch, tương ứng $M = 1/10$.

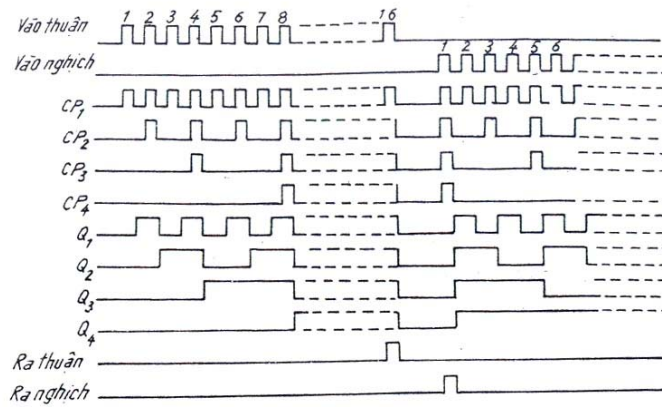


Hình 6-2-7 Đồ hình trạng thái của bộ đếm thuận nghịch

Kiểu mạch có 2 đầu vào xung đồng hồ



Hình 6-2-8 Bộ đếm thuận nghịch 2 đầu vào xung đồng hồ



Hình 6-2-9 Dạng sóng bộ đếm thuận nghịch 2 đầu vào xung đồng hồ.

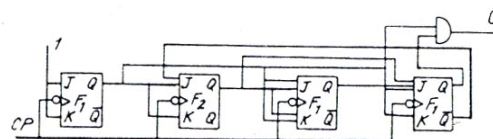
Xét sơ đồ 6-2-8. Mạch có hai đầu vào xung đếm (đồng hồ) . Khi đưa xung đồng hồ vào đầu đếm thuận, lúc đó đầu đếm nghịch ở mức thấp, chỉ có các cổng điều khiển nửa trên thông, các cổng điều khiển nửa dưới ngắt, mạch thực hiện đếm thuận. Khi đưa xung đồng hồ vào đầu đếm nghịch, lúc đó đầu đếm thuận ở mức thấp, chỉ có các cổng điều khiển nửa dưới thông, các cổng điều khiển nửa trên ngắt, mạch thực hiện đếm nghịch.

Khi đếm thuận, xung chuyển vị (đầu ra thuận) sinh ra vào lúc từ mã $Q_4Q_3Q_2Q_1 = 1111 (=15_{10})$ chuyển thành $Q_4Q_3Q_2Q_1 = 0000$. Khi đếm nghịch xung chuyển vị (đầu ra nghịch) sinh ra vào lúc từ mã $Q_4Q_3Q_2Q_1 = 0000$ chuyển thành $Q_4Q_3Q_2Q_1 = 1111 (=15_{10})$. Xem dạng sóng bộ đếm trên hình 6-2-9.

Nếu cần tăng dung lượng đếm thì có thể mắc dây chuyển (nối tiếp nhau) các mạch đếm hình 6-2-8 ; trong mạch mắc dây chuyển, đầu ra đếm thuận của bộ đếm phía trước nối vào đầu vào đếm thuận của bộ đếm sau liền kề, đầu ra đếm nghịch của bộ đếm phía trước nối vào đầu đếm nghịch của bộ đếm sau liền kề.

Trên cơ sở mạch điện hình 6-2-8, người ta thêm các đầu vào xóa, lập, vào số liệu ; người ta cũng dẫn tín hiệu đầu ra của Q của 4 Flip Flop ra ngoài, do đó tạo thành IC bộ đếm 4 bit, chẳng hạn T1193.

2. Bộ đếm thập phân



Hình 6-2-10 : Bộ đếm thuận thập phân đồng bộ

Bộ đếm thập phân là bộ đếm theo mã nhị - thập phân. Vì tương ứng với nhiều kiểu mã hóa của mã nhị - thập phân, nên cũng có nhiều kiểu bộ đếm thập phân khác nhau. Ở đây chỉ giới thiệu bộ đếm thập phân dùng mã nhị - thập phân 8421.

a) Bộ đếm thuận thập phân đồng bộ (Hình 6-2-10)

Cấu trúc mạch điện

Mạch điện cấu trúc bằng 4 Flip Flop JK và cổng chuyển vị (nhớ) đầu ra C, xung đếm đầu vào là CP.

Nguyên lý công tác

Viết phương trình

$$\begin{aligned} \text{Phương trình định thời} \quad & \text{CP}_1 = \text{CP}_2 = \text{CP}_3 = \text{CP}_4 = \text{CP} \\ & C = Q_4^n Q_1^n \end{aligned} \quad (6-2-7)$$

Phương trình kích

$$\begin{cases} J_1 = K_1 = 1 \\ J_2 = \overline{Q_4}^n Q_1^n & K_2 = Q_1^2 \\ J_3 = K_3 = Q_2^n Q_1^n \\ J_4 = Q_3^n Q_2^n Q_1^n K_4 = Q_1^n \end{cases} \quad (6-2-8)$$

(Đầu vào J của F₄ trên hình 6-2-10 vẽ 3 đường có thể hiểu rằng ở đó có cổng AND 3 đầu vào tương ứng).

Tìm phương trình trạng thái

$$\begin{cases} Q_1^{n+1} = J_1 \overline{Q_1}^n + \overline{K_1} Q_1^n = \overline{Q_1}^n \\ Q_2^{n+1} = J_2 \overline{Q_2}^n + \overline{K_2} Q_2^n = \overline{Q_4}^n Q_1^n \overline{Q_2}^n + \overline{Q_1}^n Q_2^n \\ Q_3^{n+1} = J_3 \overline{Q_3}^n + \overline{K_3} Q_3^n = Q_2^n Q_1^n \overline{Q_3}^n + \overline{Q_2}^n \overline{Q_1}^n Q_3^n \\ Q_4^{n+1} = J_4 \overline{Q_4}^n \overline{K_4} Q_4^n = Q_3^n Q_2^n Q_1^n \overline{Q_4}^n + \overline{Q_1}^n Q_4^n \end{cases} \quad (6-2-9)$$

Tính toán

Bắt đầu từ giá trị $Q_4^n Q_3^n Q_2^n Q_1^n = 0000$, thay vào phương trình trạng thái (6-2-9) và phương trình đầu ra (6-2-7), ta được kết quả ở bảng 6-2-2

Bảng 6-2-2 : KẾT QUẢ TÍNH TOÁN

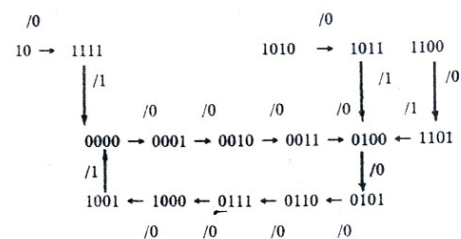
Q_4^n	Q_3^n	Q_2^n	Q_1^n	Q_4^{n+1}	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	C
0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	1	0	0
0	0	1	0	0	0	1	1	0
0	0	1	1	0	1	0	0	0
0	1	0	0	0	1	0	1	0

0	1	0	1	0	1	1	0	0
0	1	1	0	0	1	1	1	0
0	1	1	1	1	0	0	0	0
1	0	0	0	1	0	0	1	0
1	0	0	1	0	0	0	0	1
1	0	1	0	1	0	1	1	0
1	0	1	1	0	1	0	0	1
1	1	0	0	1	1	0	1	0
1	1	0	1	0	1	0	0	1
1	1	1	0	1	1	1	1	0
1	1	1	1	0	1	1	1	0
1	1	1	1	1	0	0	0	1

Đồ hình trạng thái

Căn cứ vào bảng 6-2-2, ta có quan hệ chuyển đổi trạng thái Q^n sang Q^{n+1} và giá trị đầu ra C, do đó vẽ được đồ hình trạng thái hình 6-2-11.

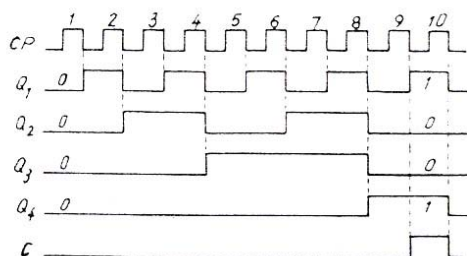
Cần lưu ý rằng, những trạng thái



Hình 6-2-11. Đồ hình trạng thái của bộ đếm thuận thập phân

liên tiếp nhau của mạch điện, một trạng thái ở thời điểm xét là Q^{n+1} đối với trạng thái trước liền

kề (ở phía trái bảng 6-2-2), đồng thời là Q^n đối với trạng thái sau liền kề (ở phía phải bảng 6-2-2), vậy nên trong hai hàng liên nhau của bảng 6-2-2, một trạng thái ở phía phải hàng trên sẽ đồng thời ở phía trái hàng dưới. Kết quả, khi vẽ đồ hình trạng thái ta phải không bỏ sót bất kỳ trạng thái khả dĩ nào để có hình 6-2-11.



Hình 6-2-12.

Dạng sóng bộ đếm thuận thập phân thập phân

Nhận xét hình 6-2-11 : Mạch điện ở hình 6-2-10 đúng là bộ đếm thập phân dùng mã 8421. Xem đồ thị dạng sóng hình 6-2-12.

Giả thiết bộ đếm hình 6-2-10 là bộ đếm hàng đơn vị của hệ thập phân. Hình 6-2-12 chứng tỏ rằng khi mạch điện chuyển đổi đến trạng thái 1001 (= 9₁₀), tín hiệu chuyển vị (nhớ) trở thành mức cao, nhưng thật ra không tác động ngay, mà đợi đến sườn âm

xung đồng hồ CP thứ 10 xuất hiện, thì tín hiệu C mới kích Flip Flop hàng chục của hệ thập phân chuyển đổi trạng thái, đồng thời bộ đếm hàng đơn vị trở về 0, tức mạch điện trở về trạng thái 0000. Tất nhiên Flip Flop hàng chục cũng được kích bằng sườn âm.

Trạng thái được sử dụng, trạng thái cấm, tự khởi động.

Như đã từng nói trước đây, những trạng thái từ mã được dùng khi mã hóa (biên mã) gọi là trạng thái được sử dụng, những trạng thái từ mã không được dùng khi mã hóa gọi là trạng thái cấm. Trong hình 6-2-11, các trạng thái từ mã 1010 ÷ 1111 là trạng thái cấm, vì chúng không được dùng khi mã hóa 8421.

Khi vì một nguyên nhân nào đó, chẳng hạn do nhiễu gây ra, mạch điện rơi vào trạng thái cấm, dưới tác động của xung đồng hồ CP mà mạch điện có thể trở lại trạng thái được sử dụng, ta nói mạch đã có thể tự khởi động.

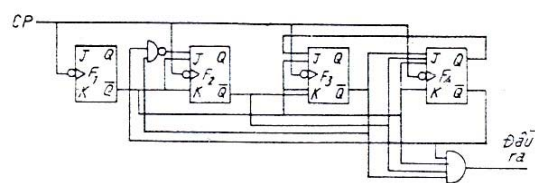
Dưới tác dụng của xung đếm đầu vào, bộ đếm vẫn công tác tuần hoàn. Ở tình huống bình thường, sự tuần hoàn đều đặn theo chu kỳ về trạng thái ban đầu mỗi chu kỳ gọi là sự tuần hoàn được sử dụng. Ngược lại, sự tuần hoàn trong trạng thái cấm gọi là sự tuần hoàn cấm. Chúng ta sẽ rõ thêm tình huống này khi trao đổi ở tiết 6-3 sau đây.

Hình 6-2-11 chứng tỏ rằng bộ đếm thuận thập phân theo mã 8421 nói ở trên là có thể tự khởi động.

b) Bộ đếm nghịch thập phân đồng bộ

Mạch điện hình 6-2-13

khá đơn giản, chúng ta dễ dàng phân tích theo phương pháp tương tự đã dùng ở trên.

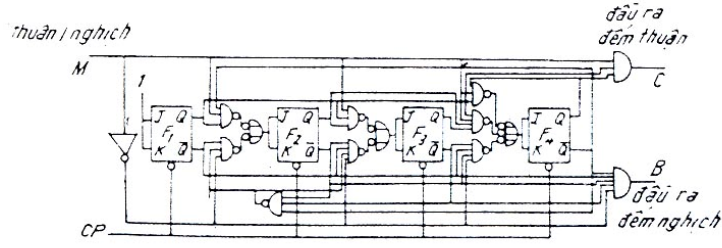


Hình 6-2-13 Bộ đếm nghịch thập phân đồng bộ

Có hai cách để cấu trúc bộ đếm thập phân đồng bộ nhiều số từ bộ đếm 1 số. Cách thứ nhất là nối dây chuyển các bộ đếm 1 số, đầu ra của bộ đếm trọng số bé nối với đầu vào bộ đếm trọng số lớn tiếp theo. Sự làm việc của bộ đếm 1 số là đồng bộ, nhưng sự làm việc giữa chúng là dị bộ.

Cách thứ hai là nối đầu ra của bộ đếm trọng số bé với tất cả các đầu vào đồng bộ của 4FF bộ đếm trọng số lớn tiếp theo, cũng nối đến đầu vào cổng chuyển vị đầu ra của bộ đếm này, còn các xung đồng hồ của các bộ đếm đều là xung đếm đầu vào CP. Trong cách thứ hai, không những từng bộ đếm 1 số là đồng bộ, mà sự làm việc của toàn bộ mạch nối ghép để đếm nhiều số cũng là đồng bộ.

c) Bộ đếm thuận nghịch thập phân đồng bộ :



Hình 6-2-14 Bộ đếm thuận nghịch thập phân đồng bộ kiểu điều khiển

Xét hình 6-2-14 . 4FFJK mắc thành FFT.

Khi tín hiệu thuận/nghịch $M = 1$, bộ đếm là thuận, $M = 0$ bộ đếm là nghịch.

$$\text{Phương trình định thời : } CP_1 = CP_2 = CP_3 = CP_4 = CP \quad (6-2-10)$$

$$\text{Phương trình đầu ra : } C = MQ_4^n Q_1^n \quad (6-2-11)$$

$$B = \overline{M} \overline{Q_4^n} \overline{Q_3^n} \overline{Q_2^n} \overline{Q_1^n} \quad (6-2-12)$$

Phương trình kích :

$$\left\{ \begin{array}{l} T_1 = 1 \\ T_2 = \overline{MQ_1^n Q_4^n} + \overline{MQ_1^n Q_2^n Q_3^n Q_4^n} \\ \quad = \overline{MQ_1^n Q_4^n} + \overline{MQ_1^n (Q_2^n + Q_3^n + Q_4^n)} \\ T_3 = \overline{MQ_1^n Q_2^n} + \overline{MQ_1^n Q_2^n Q_3^n Q_4^n} \\ \quad = \overline{MQ_1^n Q_2^n} + \overline{MQ_1^n Q_2^n (Q_3^n + Q_4^n)} \\ T_4 = \overline{MQ_1^n Q_4^n} + \overline{MQ_1^n Q_2^n Q_3^n} + \overline{MQ_1^n Q_2^n Q_3^n} \\ \quad = M(Q_1^n Q_4^n + Q_1^n Q_2^n Q_3^n) \overline{MQ_1^n Q_2^n Q_3^n} \end{array} \right. \quad (6-2-13)$$

2-13)

Với $M = 1$, từ (6-2-11), (6-2-12), (6-2-13) ta có :

$$C = Q_4^n Q_1^n \quad B = 0 \quad (6-2-14)$$

$$T_1 = 1 \quad T_2 = Q_1^n \overline{Q_4^n} \quad T_3 = Q_1^n Q_2^n \quad T_4 = Q_1^n Q_4^n + Q_1^n Q_2^n Q_3^n \quad (6-2-15)$$

Thay giá trị (6-2-15) vào phương trình đặc trưng của FFT là :

$$Q^{n+1} = T\overline{Q}^n + \overline{T}Q^n$$

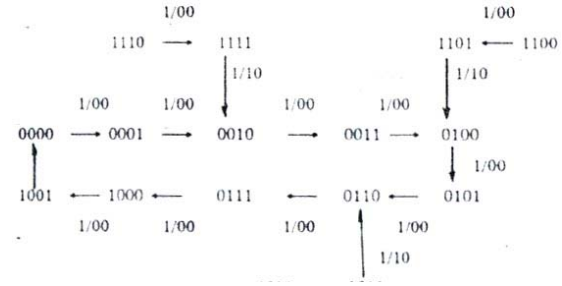
Ta có: $Q_1^{n+1} = \overline{Q}_1^n$ $Q_2^{n+1} = Q_1^n \overline{Q}_4^n \overline{Q}_2^n + \overline{Q_1^n \overline{Q}_4^n \overline{Q}_2^n} Q_2^n$

$$Q_3^{n+1} = Q_1^n Q_2^n \overline{Q}_3^n + \overline{Q_1^n Q_2^n \overline{Q}_3^n} Q_3^{n+1} = (Q_1^n Q_4^n + Q_1^n Q_2^n Q_3^n) \overline{Q}_4^n + \overline{Q_1^n Q_4^n + Q_1^n Q_2^n Q_3^n} Q_4^n \quad (6-2-16)$$

Giả định trạng thái hiện tại, tuần tự thay vào (6-12-14) và (6-12-16) để tính toán, kết quả ta có đồ hình trạng thái hình 6-2-15.

Có thể thấy rằng khi M = 1, mạch điện thực hiện đếm thuận thập phân theo mã 8421 và có thể tự khởi động.

Khi M = 0, từ (6-2-11) (6-2-12) (6-2-13) ta có :



Hình 6-2-15 : Đồ hình trạng thái khi M = 1

$$C = 0 \quad B = \overline{Q}_4^n \overline{Q}_3^n \overline{Q}_2^n \overline{Q}_1^n \quad (6-2-17)$$

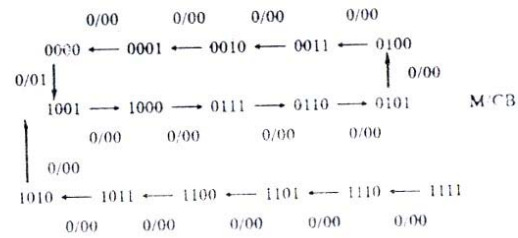
$$\begin{cases} T_1 = 1; T_2 = \overline{Q}_1^n (Q_2^n + Q_3^n + Q_4^n) \\ T_3 = \overline{Q}_1^n \overline{Q}_2^n (Q_3^n + Q_4^n); T_4 = \overline{Q}_1^n \overline{Q}_2^n \overline{Q}_3^n \end{cases} \quad (6-2-18)$$

Thay giá trị (6-2-18) vào phương trình đặc trưng của FFT, được phương trình trạng thái như sau :

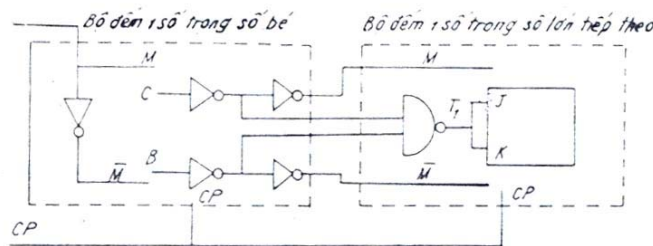
$$\begin{cases} Q_1^{n+1} = \overline{Q}_1^n \\ Q_2^{n+1} = \overline{Q}_1^n (Q_2^n + Q_3^n + Q_4^n) \overline{Q}_2^n + \overline{\overline{Q}_1^n (Q_2^n + Q_3^n + Q_4^n) \overline{Q}_2^n} Q_2^n \\ \quad = \overline{Q}_1^n \overline{Q}_2^n (Q_3^n + Q_4^n) + Q_1^n + Q_2^n \\ Q_3^{n+1} = \overline{Q}_1^n \overline{Q}_2^n (Q_3^n + Q_4^n) \overline{Q}_3^n + \overline{\overline{Q}_1^n \overline{Q}_2^n (Q_3^n + Q_4^n) \overline{Q}_3^n} Q_3^n \\ \quad = \overline{Q}_1^n \overline{Q}_2^n \overline{Q}_3^n Q_4^n + (Q_1^n + Q_2^n) Q_3^n \\ Q_4^{n+1} = \overline{Q}_1^n \overline{Q}_2^n \overline{Q}_3^n \overline{Q}_4^n + \overline{\overline{Q}_1^n \overline{Q}_2^n \overline{Q}_3^n \overline{Q}_4^n} Q_4^n \end{cases} \quad (6-2-19)$$

Giả định trạng thái hiện tại, tuần tự thay vào (6-2-17) và (6-2-19) để tính toán, kết quả ta có đồ hình trạng thái hình 6-2-16.

Có thể thấy rằng, khi M = 0, mạch điện thực hiện đếm nghịch thập phân theo mã 8421 và có thể tự khởi động.



(Hình 6-2-16) Đồ hình trạng khi $M = 0$



Hình 6-2-17 Cách ghép nối nhiều bộ đếm thuận nghịch thập phân đồng bộ 1 số thành nhiều số.

Phương pháp ghép nối nhiều bộ đếm thuận nghịch thập phân đồng bộ được trình bày ở hình 6-2-17.

Khi bộ đếm trọng số bé $C = 1$ thì ở bộ đếm trọng số lớn $M = 1$,

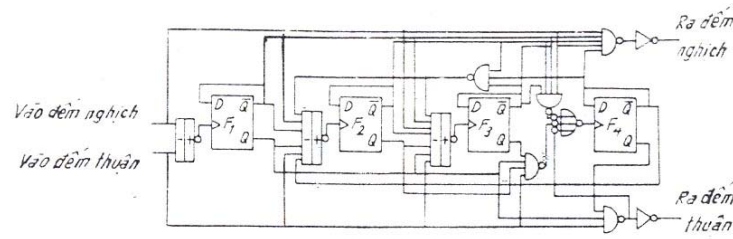
$T_1 = 1$, nó sẽ công tác chế độ đếm thuận.

Khi bộ đếm trọng số bé $B = 1$ thì ở bộ đếm trọng số lớn $\bar{M} = 1$, $T_1 = 1$, nó sẽ đếm nghịch.

Khi bộ đếm trọng số bé $C = B = 0$ thì ở bộ đếm trọng số lớn $M = \bar{M} = T_1 = 0$, và nó ngừng đếm.

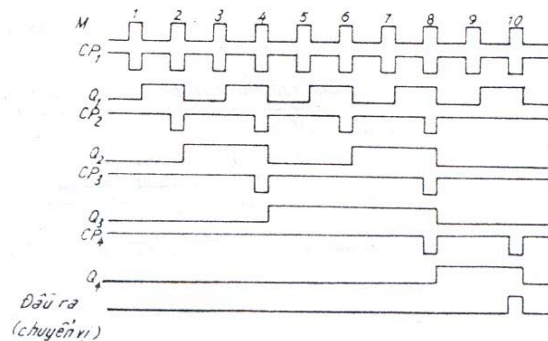
Xem lại hình 6-2-14, sơ đồ sử dụng Flip Flop T, xung CP liên tục đưa đến đồng thời tất cả đầu vào đồng hồ của các FF. Trạng thái đầu vào T quyết định sự chuyển đổi trạng thái của FF. $T = 1$ thì FF phải chuyển đổi, $T = 0$ thì FF phải không chuyển đổi.

Hình 6-2-18 dưới đây giới thiệu mạch điện bộ đếm thuận nghịch sử dụng Flip Flop T'.



Hình 6-2-18a Sơ đồ logic bộ đếm thuận nghịch thập phân đồng bộ kiểu 2 đầu vào đồng hồ.

Flip Flop T' sẽ chuyển đổi trạng thái mỗi khi xung đồng hồ đến. Xung CP là phải đi qua các cổng điều khiển mới đến được đầu vào đồng hồ của FF. Vậy FF muốn lật thì cổng phải mở, FF cần giữ nguyên trạng thái thì cổng phải đóng. Xung đếm thuận, xung đếm nghịch được đưa đến bằng hai đầu vào riêng biệt. Khi đưa vào xung đếm thuận thì bộ đếm thực hiện đếm thuận. Khi đưa vào đầu đếm nghịch thì bộ đếm đếm nghịch. Không cho phép hai đầu vào đồng thời có xung đếm. Khi ghép nối bộ đếm, chỉ cần nối đầu ra đếm thuận của bộ đếm 1 số trọng số bé với đầu vào đếm thuận của bộ đếm 1 số trọng số lớn hơn kế tiếp, tương tự cho đếm nghịch, nghĩa là nối dây chuyển.

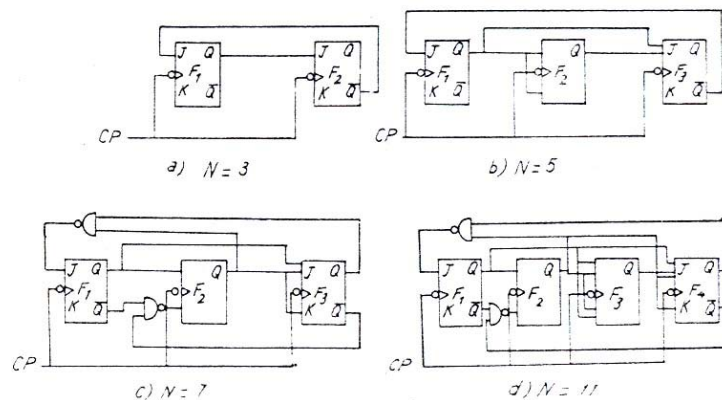


Hình 6-2-18b Dạng sóng bộ đếm hình 6-2-18a trường hợp đếm thuận

Sử dụng phương pháp phân tích logic, tham khảo phần 6-2-2-1c, tương tự, ta có thể tìm đồ hình trạng thái của mạch điện hình 6-2-18a. Còn dạng sóng thì xem hình 6-2-18b.

3. Bộ đến N phân đồng bộ

Sử dụng FF₁ ta có thể tùy ý xây dựng bộ đếm đồng bộ với hệ số đếm N bất kỳ (N phân) và dung lượng bất kỳ. Xem hình 6-2-19.



Hình 6-2-19 : Bộ đếm đồng bộ N phân

6.2.3. Bộ đếm dị bộ

1. Bộ đếm nhị phân dị bộ

Dem Flip Flop T' mắc dây chuyền với nhau thì được bộ đếm nhị phân dị bộ, kết cấu khá đơn giản.

a) Bộ đếm thuận nhị phân dị bộ

Cấu trúc mạch điện. Xem hình 6-2-20

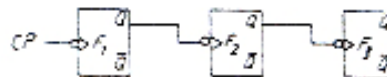
Nguyên lý làm việc :

Viết phương trình :

Phương trình định thời : $CP_1 = CP$,

$CP_2 = Q_1, CP_3 = Q_2$ (6-2-20)

Flip Flop T' sẽ lật (chuyển đổi) mỗi khi xung đồng hồ xuất hiện.



Hình 6-2-20 : Bộ đếm thuận nhị phân dị bộ

Phương trình trạng thái :

$$\begin{cases} Q_1^{n+1} = \overline{Q_1^n} \text{ với điều kiện xuất hiện sườn âm CP} \\ Q_2^{n+1} = \overline{Q_2^n} \text{ với điều kiện xuất hiện sườn âm } Q_1 \\ Q_3^{n+1} = \overline{Q_3^n} \text{ với điều kiện xuất hiện sườn âm } Q_2 \end{cases} \quad (6-2-21)$$

Trong (6-2-21) ghi rõ điều kiện định thời.

Tính toán

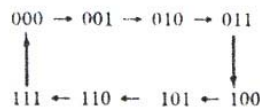
Giả thiết trạng thái ban đầu là $Q_3^n Q_2^n Q_1^n = 000$, tuần tự thay vào (6-2-21) để tính, ta được kết quả như bảng 6-2-3.

Bảng 6-2-3 : KẾT QUẢ TÍNH TOÁN

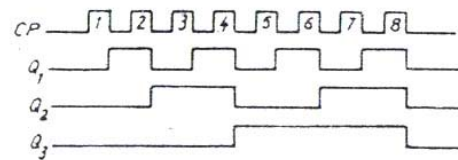
Q_3^n	Q_2^n	Q_1^n	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	Điều kiện sườn âm
0	0	0	0	0	1	CP_1
0	0	1	0	1	0	$CP_1 CP_2$
0	1	0	0	1	1	CP_1
0	1	1	1	0	0	$CP_1 CP_2 CP_3$
1	0	0	1	0	1	CP_1
1	0	1	1	1	0	$CP_1 CP_2$
1	1	0	1	1	1	CP_1
1	1	1	0	0	0	$CP_1 CP_2 CP_3$

Lưu ý khi tính đến các điều kiện định thời : Chỉ với điều kiện xuất hiện sườn xung kích thì FF mới lật đúng như phương trình quy định, nếu không thì FF duy trì nguyên trạng (không lật). Ví dụ, khi $Q_3^n Q_2^n Q_1^n = 000$ và với điều kiện sườn âm xung đến đầu vào xuất hiện, do $CP_L = CP$ nên F_1 đủ điều kiện định thời để thực hiện lật theo phương trình $Q_1^{n+1} = Q_1^n$ cụ thể $Q_1^n = 0$ nên $Q_1^{n+1} = 1$ mà $CP_2 = Q_1$ tuy rằng F_1 lật từ 0 sang 1, nhưng FF kích bằng sườn âm, nghĩa là chưa thỏa mãn điều kiện định thời, nên F_2 duy trì nguyên trạng, $Q_2^{n+1} = Q_2^n = 0$. F_3 cũng không lật.

Lại xét trạng thái $Q_3^n Q_2^n Q_1^n = 011$ với điều kiện xuất hiện sườn âm CP , F_1 lật trước, Q_1 xuất hiện sườn âm làm F_2 lật tiếp, Q_2 xuất hiện sườn âm làm F_3 lật sau cùng, trạng thái bộ đếm chuyển từ 011 thành 100. Trong bảng 6-2-3 có ghi chú điều kiện định thời được thỏa mãn khi bộ đếm chuyển trạng thái.



Hình 6-2-21 Đồ hình trạng thái bộ đếm



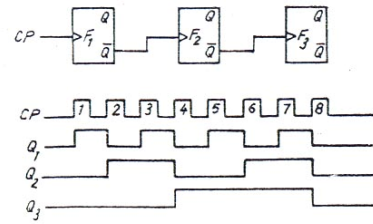
Hình 6-2-22 Dạng sóng bộ đếm.

Hình 6-2-22 là dạng sóng, ta thấy rõ đặc điểm kích bằng sườn âm của các FF bộ đếm.

Nhận xét các hình 6-2-21 và 6-2-22 : Mạch điện hình 6-2-20 là sơ đồ bộ đếm thuận nhị phân 3 số nhị bộ.

Bộ đếm thuận nhị bộ kích bằng sườn dương (hình 6-2-23)

Nguyên lý công tác của sơ đồ mạch hình 6-2-23 cũng giống sơ đồ mạch hình 6-2-10. Những Flip Flop T' được sử dụng trong sơ đồ đòi hỏi phải kích bằng sườn dương. Vậy đầu vào của FF trọng số lớn phải nối với đầu ra đảo, $\bar{Q}$ của FF



trọng số bé hơn kế tiếp.

$$CP_2 = \bar{Q}_1$$

$$\bar{Q}_2$$

Hình 6-2-23 Bộ đếm thuận nhị phân nhị bộ kích bằng sườn dương.

Chỉ cần chú ý đặc điểm điều kiện định thời kích bằng sườn dương thì ta dễ dàng hiểu được mạch đếm hình 6-2-23.

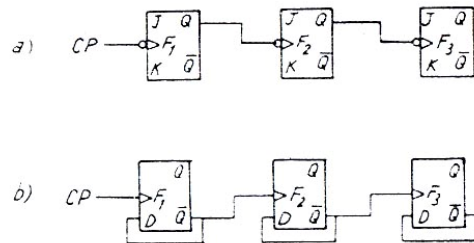
Hình 6-2-24 là sơ đồ mạch điện đếm thuận nhị phân nhị bộ 3 số.

Ở đây Flip Flop T' được cấu trúc từ Flip Flop JK và Flip Flop D.

Hình 6-2-24 Dùng Flip Flop JK, Flip Flop D.

a) dùng Flip Flop JK

b) dùng Flip Flop D



b) Bộ đếm nghịch nhị phân nhị bộ

Sơ đồ bộ đếm này trên hình 6-2-25 sử dụng Flip Flop T'.



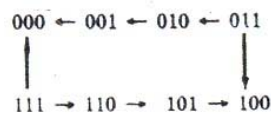
Hình 6-2-25 Kích bằng sườn âm, sườn dương

a) kích bằng sườn âm b) Kích bằng sườn dương

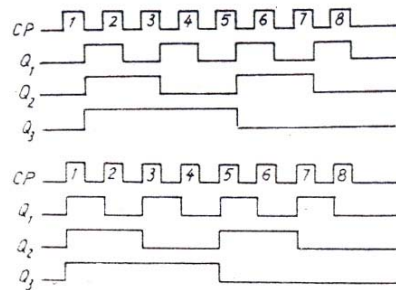
Khi không ngừng đưa xung đếm CP vào bộ đếm nghịch, tình huống chuyển đổi trạng thái của mạch điện như bảng 6-2-4, đồ hình trạng thái hình 6-2-26 và dạng sóng hình 6-2-27.

Bảng 6-2-4 : BẢNG CHỨC NĂNG

Q_3^n	Q_2^n	Q_1^n	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	Điều kiện sườn âm
0	0	0	1	1	1	$CP_1 CP_2 CP_3$
0	0	1	0	0	0	CP_1
0	1	0	0	0	1	$CP_1 CP_2$
0	1	1	0	1	0	CP_1
1	0	0	0	1	1	$CP_1 CP_2 CP_3$
1	0	1	1	0	0	CP_1
1	1	0	1	0	1	$CP_1 CP_2$
1	1	1	1	1	0	CP_1



Hình 6-2-26.
Đồ hình trạng thái



Hình 6-2-27

a) Kích bằng sườn âm

b) Kích bằng sườn dương

Hình 6-2-26 chứng tỏ rõ ràng rằng số trị của bộ đếm là giảm không ngừng theo chế độ đếm nghịch nhị phân.

c) Quy luật nối ghép các bộ đếm nhị phân dị bộ

Sự nối ghép giữa các bộ đếm nhị phân dị bộ khá đơn giản. Đầu ra Flip Flop trọng số bé nối vào đầu xung đồng hồ của Flip Flop trọng số lớn. Quy luật nối ghép xem bảng 6-2-5.

Bảng 6-2-5 : QUY LUẬT NỐI GHÉP CÁC BỘ ĐẾM NHỊ PHÂN DỊ BỘ

Quy luật nối ghép	Các kích của Flip Flop T'	
	Sườn dương	Sườn âm
Đếm thuận	$CP_i = \overline{Q}_{i-1}$	$CP_i = Q_{i-1}$
Đếm nghịch	$CP_i = Q_{i-1}$	$CP_i = \overline{Q}_{i-1}$

Trong bảng trên CP_i là xung đồng hồ của Flip Flop F_i , Q_{i-1} và $\overline{Q}_{i-1}$ là tín hiệu của Flip Flop F_{i-1} có trọng số bé hơn liền kề với F_i . Nếu dùng loại Flip Flop kích bằng sườn dương thì xung đồng hồ CP_i của F_i nối vào $\overline{Q}_{i-1}$ của F_{i-1} , nếu dùng loại Flip Flop kích bằng sườn âm thì CP_i nối vào Q_{i-1} để cấu trúc bộ đếm thuận. Còn để cấu trúc thành bộ đếm nghịch thì ngược lại.

d) Đặc điểm bộ đếm dị bộ nhị phân

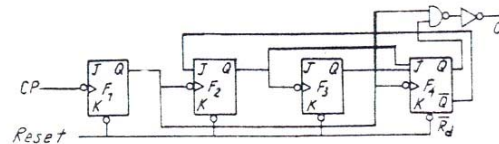
Ưu điểm : Cách nối ghép bộ đếm và cấu trúc đếm đều đơn giản.

Nhược điểm : Tần số công tác thấp, xung nhiễu quá độ từ trạng thái mã hóa này sang trạng thái mã hóa khác lớn.

2. Bộ đếm thập phân dị bộ

a) Bộ đếm thuận thập phân dị bộ

Cấu trúc mạch điện



Sơ đồ hình 6-2-28 bao gồm 4 Flip Flop JK và hai cổng . CP là xung đếm đầu vào, C là tín hiệu chuyển vị (nhớ) đưa đến bộ đếm trọng số lớn hơn.

Hình 6-2-28 Bộ đếm thuận thập phân dị bộ

Nguyên lý làm việc

Viết phương trình

Phương trình định thời : $CP_1 = CP$ $CP_2 = CP_4 = Q_1$ $CP_3 = Q_2$ (6-2-22)

Phương trình đầu ra : $C = Q_4^n Q_1^n$ (6-2-23)

Phương trình kích :

$$\begin{aligned} J_1 &= K_1 = 1 & J_2 &= \overline{Q}_4^n & K_2 &= 1 \\ J_3 &= K_3 = 1 & J_4 &= Q_3^n Q_2^n & K_4 &= 1 \end{aligned} \quad (6-2-24)$$

(Tạm quy ước rằng các đầu vào để trống là nối vào mức logic 1, đầu vào nào có nhiều đường nối coi như ở đó có mạch NAND).

Tìm phương trình trạng thái

Thay giá trị (6-2-24) vào phương trình đặc trưng của Flip Flop JK, ta có :

$$\begin{cases} Q_1^{n+1} = J_1 \bar{Q}_1^n + \bar{K}_1 Q_1^n = \bar{Q}_1^n & \text{với điều kiện xuất hiện sườn âm CP} \\ Q_2^{n+1} = J_2 \bar{Q}_2^n + \bar{K}_2 Q_2^n = \bar{Q}_4^n \bar{Q}_2^n & \text{với điều kiện xuất hiện sườn âm } Q_1 \\ Q_3^{n+1} = J_3 \bar{Q}_3^n + \bar{K}_3 Q_3^n = \bar{Q}_3^n & \text{với điều kiện xuất hiện sườn âm } Q_2 \\ Q_4^{n+1} = J_4 \bar{Q}_4^n + \bar{K}_4 Q_4^n = Q_3^n Q_2^n \bar{Q}_4^n & \text{với điều kiện xuất hiện sườn âm } Q_1 \end{cases} \quad (6-2-25)$$

Tiến hành tính toán

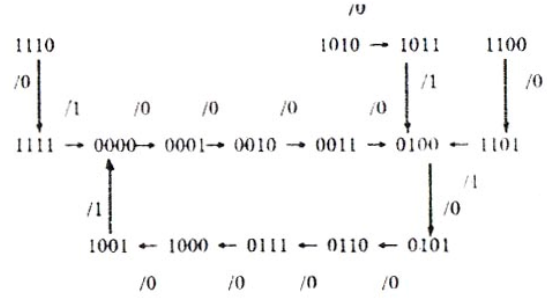
Giả định trạng thái, tuần tự thay vào (6-2-25) và (6-2-23) để tính ra kết quả như bảng 6-2-6 . Khi tính toán cần lưu ý điều kiện định thời . Chỉ khi sườn âm xung kích tương ứng xuất hiện thì trạng thái Flip Flop mới chuyển đến trạng thái kế tiếp theo phương trình.

Bảng 6-2-6 : BẢNG TRẠNG THÁI CỦA BỘ ĐẾM THUẬN THẬP PHÂN DỊ

BỘ									
Q_4^n	Q_3^n	Q_2^n	Q_1^n	Q_4^{n+1}	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}		
0	0	0	0	0	0	0	1	0	CP ₁
0	0	0	1	0	0	1	0	0	CP ₁ CP ₂ CP ₄
0	0	1	0	0	0	1	1	0	CP ₁
0	0	1	1	0	1	0	0	0	CP ₁ CP ₂ CP ₃ CP ₄
0	1	0	0	0	1	0	1	0	CP ₁
0	1	0	1	0	1	1	0	0	CP ₁ CP ₂ CP ₄
0	1	1	0	0	1	1	1	0	CP ₁
0	1	1	1	1	0	0	0	0	CP ₁ CP ₂ CP ₃ CP ₄
1	0	0	0	1	0	0	1	0	CP ₁
1	0	0	1	0	0	0	0	1	CP ₁ CP ₂ CP ₄
1	0	1	0	1	0	1	1	0	CP ₁
1	0	1	1	0	1	0	0	1	CP ₁ CP ₂ CP ₃ CP ₄
1	1	0	0	1	1	0	1	0	CP ₁
1	1	0	1	0	1	0	0	1	CP ₁ CP ₂ CP ₄
1	1	1	0	1	1	1	1	0	CP ₁
1	1	1	1	0	0	0	0	1	CP ₁ CP ₂ CP ₃ CP ₄

Vẽ đồ hình trạng thái

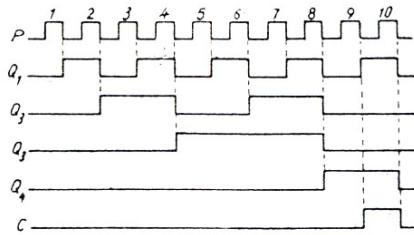
Căn cứ vào kết quả tính toán ở bảng 6-2-6 có thể vẽ đồ hình trạng thái như hình 6-2-29. Đồ hình này chứng tỏ rằng mạch điện có sơ đồ ở hình 6-2-28 là bộ đếm thuận thập phân



Hình 6-2-29.

dị bộ theo cách mã hóa 8421, hơn nữa mạch điện có thể tự khởi động. Xem dạng sóng hình 6-2-30.

Đồ hình trạng thái bộ đếm thuận thập phân dị bộ

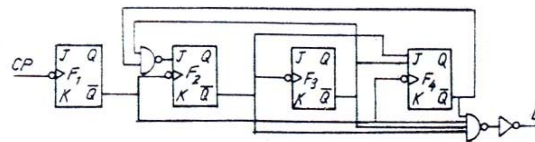


Hình 6-2-30.

Dạng sóng bộ đếm thuận thập phân dị bộ

b) Bộ đếm nghịch thập phân dị bộ (hình 6-2-31)

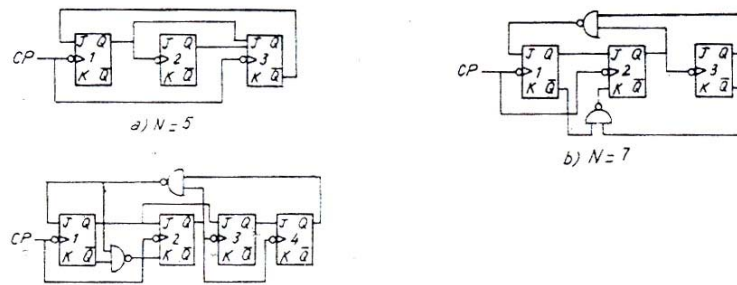
Phương pháp phân tích như trên.



Hình 6-2-31 Bộ đếm nghịch thập phân dị bộ

Về cách ghép nối các bộ đếm thập phân dị bộ 1 số thành bộ đếm nhiều chữ số : ghép nối dây chuyên, đầu ra của bộ đếm trọng số bé được nối vào đầu vào CP của bộ đếm trọng số lớn hơn kế tiếp.

c) Bộ đếm N phân dị bộ (hình 6-2-32)



Hình 6-2-32 Bộ đếm N phân dị bộ

d) Xóa và cài đặt ban đầu các bộ đếm dị bộ

Xóa bộ đếm là đưa tất cả Flip Flop của bộ đếm về trạng thái 0 trước khi bắt đầu công tác. Đối với Flip Flop D có mạch ổn định theo nguyên lý duy trì - ngăn trở thì cần chọn đầu vào R_d là mạch tín hiệu xóa Flip Flop về 0 (Đầu vào R bị trạng thái xung đồng hồ CP ảnh hưởng). Vì trong bộ đếm dị bộ, đầu vào CP của các Flip Flop có thể là 1, cũng có thể là 0.

Cài đặt ban đầu là làm cho bộ đếm trước khi bắt đầu công tác phải ở sẵn sàng trạng thái quy định bằng cách điều khiển đầu vào dị bộ. Cùng với một nguyên nhân đã nói trên, đối với Flip Flop D duy trì - ngăn trở thì cần chọn đầu vào S_d ; đầu vào S_d này không bị ảnh hưởng của CP (như đầu vào S). Đối với bộ đếm nghịch dị bộ, đầu tiên đặt tất cả Flip Flop về trạng thái 1, sau đó theo trạng thái quy định mà xóa những Flip Flop được chọn về 0, để đạt được mục đích là cài đặt ban đầu cho bộ đếm. Sở dĩ như vậy vì trong bộ đếm nghịch dị bộ, FF trọng số bé lật từ 0 sang 1 sẽ đưa ra tín hiệu chuyển vị (nhớ) cho FF trọng số lớn hơn, làm nó lật. Do đó thông qua đầu vào dị bộ, đầu tiên đặt tất cả FF ở trạng thái 0, sau đó mới chọn FF để lập 1. Nếu dùng các tín hiệu đảo nhau đồng thời đưa vào đầu R_d , S_d để cài đặt ban đầu thì không còn lo bị ảnh hưởng gì nữa.

6.2.4. Bộ đếm IC cỡ trung (MSI)

Các vi mạch MSI bộ đếm có nhiều chủng loại, chức năng mạnh, dùng tiện lợi.

1. Ví dụ phân tích :

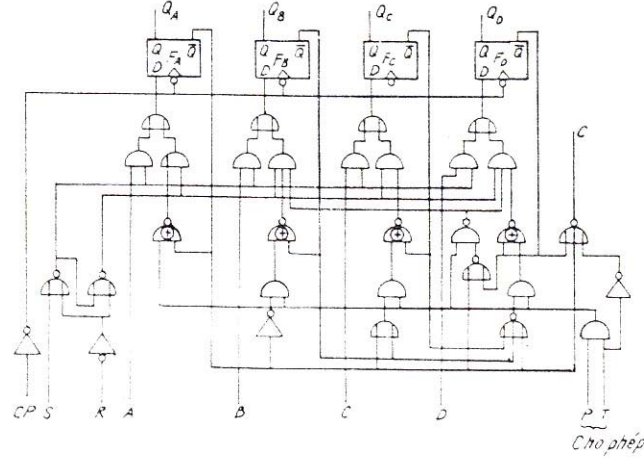
Hình 6-2-33 là bộ đếm thuận thập phân đồng bộ MSI.

Mạch điện bao gồm 4 Flip Flop D và nhiều cổng, 4 đầu vào dữ liệu A, B, C, D một đầu lập S, một đầu xóa R, hai đầu vào P, T bốn đầu ra Q_A , Q_B , Q_C , Q_D và đầu chuyển vị (nhớ)

a) Phương trình đầu ra và phương trình trạng thái

Phương trình đầu ra và phương trình trạng thái

$$\text{Phương trình đầu ra } C = \overline{Q_D^n} + \overline{Q_A^n} + \overline{T} = Q_D^n Q_A^n T \quad (6-2-26)$$



Hình 6-2-33. Bộ đếm MSI

Phương trình kích :

$$\begin{aligned} D_A &= \overline{\overline{R} + SA + \overline{R} + S + \overline{R}} \overline{PT} \oplus \overline{Q_A^n} \\ D_A &= \overline{R} \overline{S} A + \overline{R} S + \overline{R} \overline{PT} \oplus \overline{R_A^n} = \overline{R} \overline{S} A + \overline{R} S \overline{PT} + \overline{Q_A^n} \\ D_B &= \overline{\overline{R} + SB + \overline{R} + S + \overline{R}} \overline{PT} \overline{Q_A^n} \oplus \overline{Q_B^n} \overline{PT} \overline{Q_A^n} + \overline{Q_D^n} \\ D_B &= \overline{R} \overline{S} B + \overline{R} S \overline{PT} \overline{Q_A^n} \oplus \overline{Q_B^n} \overline{PT} \overline{Q_A^n} \overline{Q_D^n} \\ D_C &= \overline{\overline{R} + SC + \overline{R} + S + \overline{R}} \overline{PT} \overline{Q_A^n} + \overline{Q_B^n} \oplus \overline{Q_C^n} \\ D_C &= \overline{R} \overline{S} C + \overline{R} S \overline{PT} \overline{Q_A^n} + \overline{Q_B^n} \oplus \overline{Q_C^n} \\ D_D &= \overline{\overline{R} + SD + \overline{R} + S + \overline{R}} \overline{PT} \overline{Q_A^n} + \overline{Q_D^n} \overline{PT} \overline{Q_A^n} + \overline{Q_B^n} + \overline{Q_C^n} \oplus \overline{Q_D^n} \\ D_D &= \overline{R} \overline{S} D + \overline{R} S \overline{PT} \overline{Q_A^n} \overline{Q_D^n} \overline{PT} \overline{Q_A^n} \overline{Q_B^n} \overline{Q_C^n} \oplus \overline{Q_D^n} \end{aligned} \quad (6-2-27)$$

Phương trình trạng thái

Thay giá trị (6-2-27) vào phương trình đặc trưng của Flip Flop D, ta có :

$$\begin{cases} Q_A^{n+1} = R\bar{S}A + RSPT \oplus \bar{Q}_A^n \\ Q_B^{n+1} = R\bar{S}B + RSPTQ_A^n \oplus Q_B^n PTQ_A^n Q_D^n \\ Q_C^{n+1} = R\bar{S}C + RSPTQ_A^n Q_B^n \oplus Q_C^n \\ Q_D^{n+1} = R\bar{S}D + RSPTQ_A^n Q_D^n PTQ_A^n Q_B^n Q_C^n \oplus \bar{Q}_C^n \end{cases} \quad (6-2-28)$$

2-28)

b) 4 chế độ công tác :

Xóa :

$R = 0$, với sự xuất hiện sườn dương xung đồng hồ CP, bộ đếm bị xóa.

Thay giá trị $R = 0$ vào (6-2-28) , ta có :

$$Q_A^{n+1} = 0 \quad Q_B^{n+1} = 0 \quad Q_C^{n+1} = 0 \quad Q_D^{n+1} = 0$$

Cài đặt ban đầu.

$R = 1, S = 0$, với sự xuất hiện sườn dương CP, bộ đếm thiết lập số, tức là đưa dữ liệu đầu vào DCBA vào bộ đếm.

Thay giá trị $R = 1, S = 0$ vào (6-2-28), ta có :

$$Q_A^{n+1} = A \quad Q_B^{n+1} = B \quad Q_C^{n+1} = C \quad Q_D^{n+1} = D$$

Duy trì nguyên trạng (nhớ)

$R = S = 1, PT = 0$ với sự xuất hiện sườn dương CP, bộ đếm duy trì trạng thái vốn có, tức là cấm lật.

Thay giá trị $R = S = 1, PT = 0$ vào (6-2-28) , ta có :

$$Q_A^{n+1} = Q_A^n \quad Q_B^{n+1} = Q_B^n \quad Q_C^{n+1} = Q_C^n \quad Q_D^{n+1} = Q_D^n$$

Đếm :

$R = S = 1, P = T = 1$, bộ đếm thực hiện chức năng đếm.

Thay giá trị $R = S = P = T = 1$ vào (6-2-26), (6-2-28), ta có :

$$\begin{cases} Q_A^{n+1} = \bar{Q}_A^n \\ Q_B^{n+1} = Q_B^n \bar{Q}_A^n + \bar{Q}_D^n \bar{Q}_B^n Q_A^n \\ Q_C^{n+1} = \bar{Q}_C^n Q_B^n Q_A^n + Q_C^n \bar{Q}_B^n Q_A^n \\ Q_D^{n+1} = \bar{Q}_D^n Q_C^n Q_B^n Q_A^n + Q_D^n Q_A^n \end{cases} \quad (6-2-29)$$

$$C = Q_D^n Q_A^n \quad (6-2-30)$$

Nếu đặt $Q_A = Q_1, Q_B = Q_2, Q_C = Q_3, Q_D = Q_4$ thì các công thức (6-2-29), (6-2-30) hoàn toàn trùng hợp với các công thức (6-2-9) và (6-2-7) của bộ đếm.

Thuận tiện phân đồng bộ. Do đó có thể dùng chung bảng chức năng, để hình trạng thái (bảng (6-2-2) và hình (6-2-11)

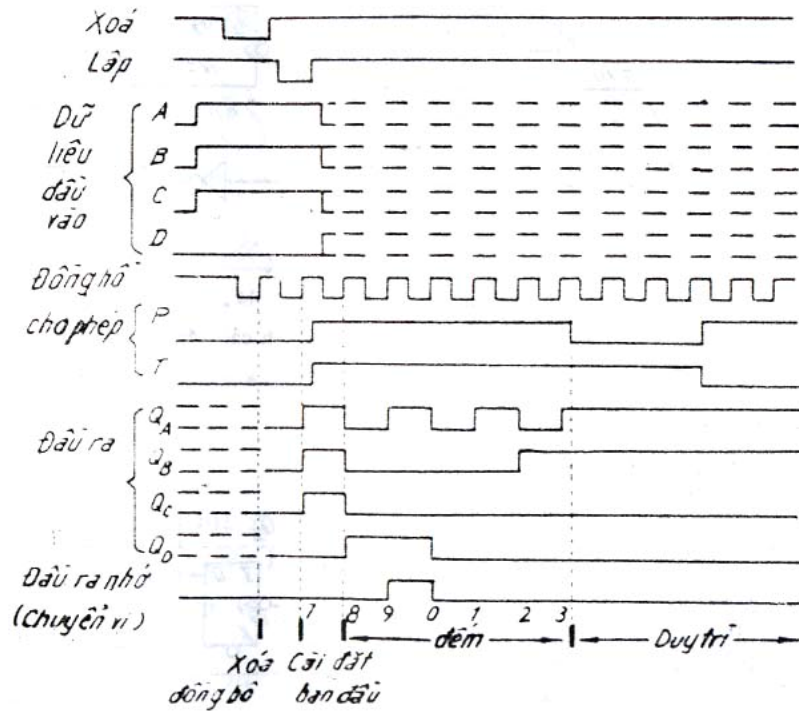
Qua trình bày trên đây, căn cứ phương trình đầu ra và phương trình trạng thái (6-2-26) và (6-2-28) có thể liệt kê thành bảng chức năng 6-27.

Bảng 6-2-7: BẢNG CHỨC NĂNG BỘ ĐẾM HÌNH 6-2-33

P	T	R	S	CP	Chức năng
H	H	H	H	↑	Đếm
x	x	H	L	↑	Cài đặt ban đầu
L	H	H	H	x	Duy trì nguyên trạng
x	L	H	H	x	Duy trì C = 0
x	x	L	x	↑	Xoá

H biểu thị mức cao, L biểu thị mức thấp, x biểu thị tùy ý, ↑ biểu thị sườn dương xung đồng hồ.

Hình 6-2-34 là dạng sóng của bộ đếm ; hình 6-2-34 phản ánh toàn diện tình huống công tác của mạch điện hình 6-2-33



Hình 6-2-34. Dạng sóng bộ đếm

2) Sử dụng bộ đếm IC cấu trúc bộ đếm N phân

Trên cơ sở các bộ đếm thuận thập phân (mã hoá 8421) và nhị phân dạng IC, bằng phương pháp phản hồi để xoá, chúng ta có thể cấu trúc bộ đếm N phân.

a) Các bước cơ bản

Giả sử S_0 biểu thị 0, S_1 biểu thị 1, ..., S_{N-1} biểu thị N-1, S_N biểu thị N. Phương pháp phản hồi để xoá có các bước sau:

- Viết ra mã nhị phân của S_N
- Tìm logic phản hồi để xoá – biểu thức của R_d
- Vẽ sơ đồ logic

b) Công thức logic phản hồi để xoá

Logic phản hồi để xoá là biểu thức hàm số của đầu xoá di bộ R_d của bộ đếm IC.

$$R_d = \bar{P} = \overline{\prod_{1-n} Q^1} \quad (6-2-31)$$

Trong công thức : P là mã của trạng thái S_N , $\prod_{1-n} Q^1$ là tích các giá trị Q của các FF trạng thái 1 tương ứng S_N . Trong mã trạng thái của S_N còn bao gồm $\prod_{1-n} Q^0$, đó là tích các giá trị $\bar{Q}$ của các FF trạng thái 0 tương ứng S_N . Nhưng trong bộ đếm N phân sử dụng logic phản hồi để xoá, vì các mã nhị phân S_{n+1} , S_{n+2} , ..., S_{2^n-1} không được dùng nên được lợi dụng để đơn giản hoá hàm logic (tối thiểu hoá), kết quả $\prod_{1-n} Q^0$ bị bỏ đi cả:

$$P = \prod_{1-n} Q^1 \prod_{1-n} Q^0 \prod_{1-n} Q^1 \quad (6-2-31)$$

Bây giờ dùng bảng Karnaugh 3 biến làm ví dụ để thuyết minh sự đúng đắn của (6-2-32).

N = 4 : $S_N = S_4 = 100$; $S_N = P_4 = Q_3^n$ $m_5 + m_7$ dùng để tối thiểu hoá

N = 5 : $S_N = S_5 = 110$; $P_N = P_5 = Q_3^n Q_1^n$ $m_6 + m_7$ dùng để tối thiểu hoá

N = 6 : $S_N = S_6 = 110$; $P_N = P_6 = Q_3^n Q_2^n$ m_7 dùng để tối thiểu hoá

N = 7 : $S_N = S_7 = 111$; $P_N = P_7 = Q_3^n Q_2^n Q_1^n$ không đơn giản hoá được

Ví dụ trên đây rất đơn giản nhưng minh hoạ (6-2-32) là chính xác

Nếu dùng bảng Karnaugh 4 biến, 5 biến cũng thấy (6-2-32) là đúng

c) Ví dụ:

Ví dụ 6-2-1 : Hãy xây dựng bộ đếm $N = 12$ bằng bộ đếm thuận nhị phân đồng bộ 4 số.

1- Viết mã nhị phân của S_N

$$N = 12 \quad S_N = S_{12} = 1100$$

2- Tìm logic phản hồi để xoá

$$R_d = \prod_{1-n} Q^1 = \overline{Q_4 Q_3}$$

3- Vẽ sơ đồ logic

Phương pháp này tồn tại hai vấn đề:

- Trạng thái quá độ S_N cực ngắn
- Độ tin cậy phản hồi xoá tương đối kém

Vấn đề trạng thái quá độ S_N

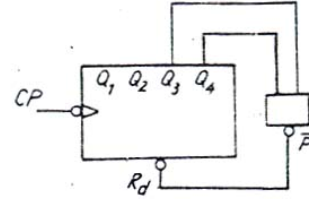
Theo tư tưởng trên đây, bộ đếm N bắt đầu đếm từ S_0 đến lúc đạt số đếm S_{N-1} nếu thêm một xung đếm thì phải lập tức bị xoá về 0. Thực tế mạch điện phản hồi để xoá không lập tức về 0. Đầu tiên mạch chuyển đổi đến trạng thái S_N , làm cho $R_d = \overline{P} = 0$. vì $R_d = 0$ nên bộ đếm reset, trạng thái S_N tiếp theo bị mất làm triệt tiêu tín hiệu $R_d = 0$. S_N tồn tại trong thời gian chừng xấp xỉ trễ truyền đạt 3 cấp cổng. Tuy trạng thái quá độ của S_N là cực ngắn, nhưng không thể thiếu; không có trạng thái quá độ đó thì không có cách nào tạo ra tín hiệu xoá về 0.

Vấn đề độ tin cậy phản hồi xoá tương đối kém

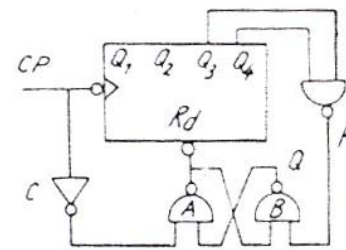
Vì đặc tính công tác xung và tình huống phụ tải của các Flip Flop bộ đếm là không giống nhau, lại luôn luôn chịu tác động của nhiễu dù nhiều dù ít, nên có thể xảy ra tình trạng có FF vẫn còn giữ nguyên trạng thái 1, trong khi đó có FF đã lật về 0.

Do $R_d = \overline{P}$, ta biết rằng chỉ cần một FF lật về 0 thì S_N mất, rồi tín hiệu $R_d = 0$ cũng mất. Một khi $R_d = 0$

mất thì FF nào không kịp lật sẽ không có cách nào lật về không nữa. Hình 6-2-36 là mạch điện giải quyết vấn đề này. Trong mạch cải tiến, Elip Flop RS kéo dài $R_d = 0$ một chút, do đó bảo đảm tín hiệu rất có thời gian tác dụng đủ dài để bộ đếm xoá về 0 chắc chắn.



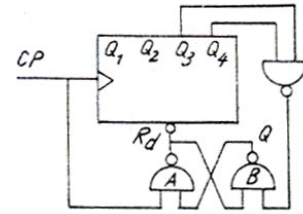
Hình 6-2-35. Bộ đếm $N = 12$



Hình 6-2-36. Mạch cải tiến của bộ đếm

Bình thường Flip Flop RS căn bản (do các cổng A, B cấu trúc nên) ở trạng thái 0 dưới tác dụng của CP, $R_d = \overline{Q} = 1$.

Khi bộ đếm đếm đến giá trị $S_{N-1} = S_{11} = 1011$, nếu xuất hiện thêm 1 xung đếm CP, vào khoảng sườn âm của nó bộ đếm lật từ S_{N-1} đến $S_N = S_{12} = 1100$, làm cho $\overline{P} = \overline{Q_4 Q_3} = 0$, Flip Flop RS căn bản lập 1, $Q = 1$, $\overline{Q} = R_d = 0$, bộ đếm reset về 0, tức là lật về $S_0 = 0000$. Chỉ khi nào bộ đếm bắt đầu đếm lại từ S_0 sườn dương xung CP qua cổng C được đảo pha rồi mới lật Flip Flop RS cơ bản về 0, $Q = 0$, $\overline{Q} = 1 = R_d$ (tín hiệu xoá về 0 bị triệt tiêu). Vậy thời gian $R_d = 0$ kéo dài thêm nhiều; nếu tỉ số bề rộng xung CP với chu kỳ của nó là 50% thì thời gian tương ứng $R_d = 0$ là một nửa chu kỳ xung CP.



Hình 6-2-37.

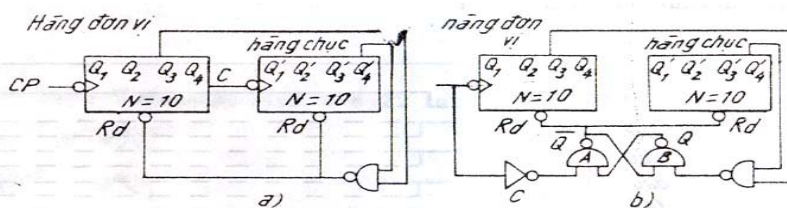
Bộ đếm nhị phân kích sườn dương

Nếu bộ đếm nhị phân cấu trúc bằng mạch kích bằng sườn dương xung đếm CP thì không cần cổng NOT nữa, mạch điện như hình 6-2-37.

Nếu không có yêu cầu đặc biệt cao về độ tin cậy, nói chung người ta vẫn hay dùng mạch điện đơn giản hình 6-2-35.

Sử dụng bộ đếm IC, cấu trúc bộ đếm N phân rất đơn giản tiện lợi. Cách làm này cũng khá kinh tế. Dùng 2 IC bộ đếm nhị phân 4 số có thể cấu trúc bộ đếm $N = 1 \text{ 5 } 256$. Dùng 2 IC bộ đếm thập phân có thể cấu trúc bộ đếm $N = 1 \text{ 5 } 100$. Hình 6-2-38 biểu thị bộ đếm $N = 84$

cấu trúc từ 2IC bộ đếm thập phân.



Hình 6-2-38. Bộ đếm $N = 84$.

a) Sơ đồ nguyên lý b) Mạch cải tiến

Sau đó là nạp số liệu : Dùng xung dương (gọi là xung tiếp nhận hay xung nạp số liệu) để mở thông các cổng 1 ➊ 4, đưa từ mã $D_4 D_3 D_2 D_1$ nạp vào bộ nhớ và lưu giữ lại.

Từ phương trình đặc trưng của Flip Flop RS cơ bản, sau khi xóa về 0:

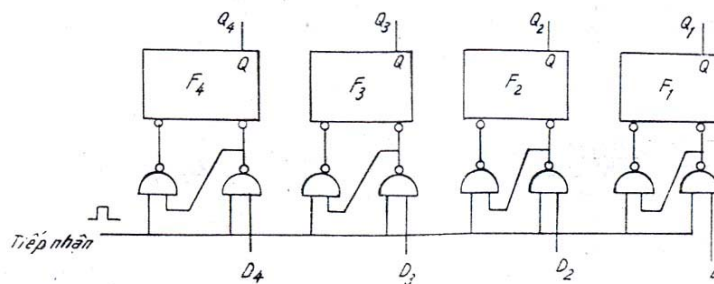
$$Q_4^n Q_3^n Q_2^n Q_1^n = 00000$$

Khi xung nạp xuất hiện :

$$Q_4^n + 1 Q_3^n + 1 Q_2^n + 1 Q_1^n + 1 = D_4 D_3 D_2 D_1$$

Vì cả quá trình công tác gồm 2 bước, nên được gọi là cách tiếp nhận hai nhịp

2) Cách tiếp nhận một nhịp (mạch điện 6-3-2)

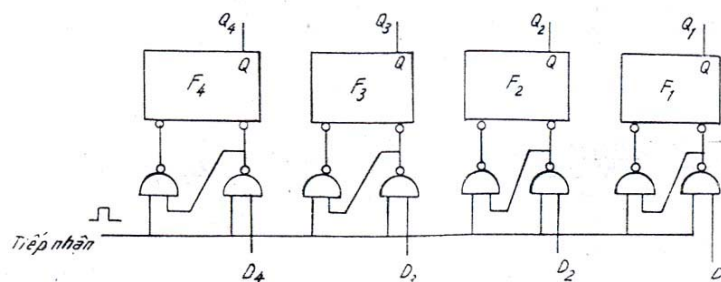


Hình 6-3-2. Cách tiếp nhận một nhịp.

Trong hình 6-3-2, $F_4 F_3 F_2 F_1$ là 4 Flip Flop RS cơ bản, $D_4 D_3 D_2 D_1$ là đầu vào số liệu, $Q_4 Q_3 Q_2 Q_1$ là đầu ra, ngoài ra còn có đầu vào điều khiển tiếp nhận. Nếu xem xét toàn bộ các FF và các cổng của nó một thể thì mạch điện 6-3-2 thực tế là do 4 mạch chốt D cấu trúc nên, với đầu vào điều khiển tiếp nhận là đầu vào xung đồng hồ của mạch chốt, tức là CP.

Phương trình đặc trưng của mạch chốt D là $Q^n + 1 = D$ với điều kiện xuất hiện CP.

Vậy trong mạch điện hình 6-3-2, khi xuất hiện xung tiếp nhận, Flip Flop chuyển đổi trạng thái



Hình 6-3-3. Bộ nhớ tiếp nhận một nhịp dùng Flip Flop D.

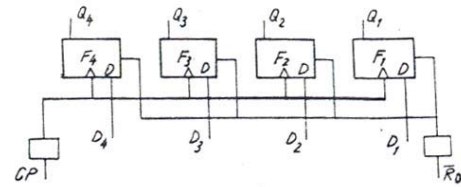
$$Q_4^n + 1 Q_3^n + 1 Q_2^n + 1 Q_1^n + 1 = D_4 D_3 D_2 D_1$$

tức là bộ nhớ tiếp nhận và lưu giữ tín hiệu nhị phân mã hoá. Mạch điện này hoàn thành chức năng bộ nhớ chỉ trong một bước, nên được gọi là mạch tiếp nhận một nhịp. Rõ ràng chúng ta có thể dùng các loại Flip Flop D, RS, JK để cấu trúc bộ nhớ cơ bản. Hình 6-3-3 giới thiệu bộ nhớ 4 bit dùng Flip Flop D

3) IC bộ nhớ cơ bản

Vì mạch bộ nhớ cơ bản bao gồm các Flip Flop và các đồng điều khiển liên quan được chế tạo trong một chip. Căn cứ vào sự khác biệt về cấu trúc, các vi mạch bộ nhớ phân thành bộ nhớ đơn (chỉ có 1 bộ nhớ trong 1 vỏ) và bộ nhớ ghép (một số bộ nhớ được đóng gói chung trong 1 vỏ).

Hình 6-3-4 là bộ nhớ 4 bit dùng Flip Flop D, ngoài ra còn hay gặp các bộ nhớ kép 2 x 5, 2 x 6 v.v... Các bộ nhớ do mạch chốt cấu trúc nên thường có mạch chốt cấu trúc nên thường có mạch chốt hai chiều 8 bit, mạch chốt 4 bit có



Hình 6-3-4. IC bộ nhớ 4 Flip Flop D

đầu vào xoá và mạch chốt kép 2 x 4. IC bộ nhớ ghép điển hình có bộ nhớ 4 x 4, 8 x 2 v.v...

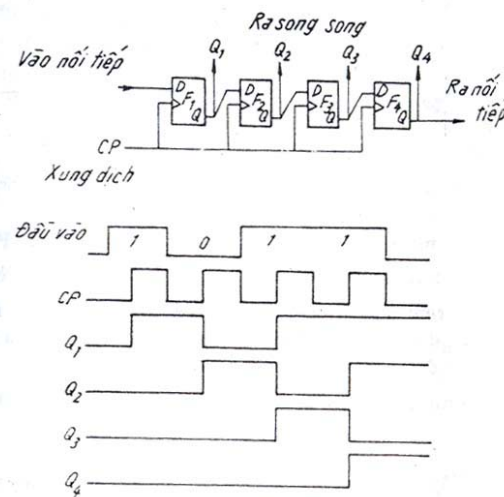
6.3.2. Bộ ghi dịch

Bộ ghi dịch ngoài chức năng lưu giữ ra, còn có chức năng dịch từng bit theo nhịp xung đồng hồ. Dữ liệu lưu giữ trong bộ ghi dịch dưới tác dụng của xung dịch (shist) có thể tuần tự dịch trái hay dịch phải.

1) Bộ ghi dịch một hướng

Hình 6-3-5 là bộ ghi dịch một hướng cấu trúc bằng Flip Flop D. Đầu ra Q của mỗi FF tuần tự nối đến đầu vào D của FF tiếp sau; chỉ có đầu vào D của FF thứ nhất tiếp nhận dữ liệu.

Cứ mỗi khi xuất hiện sườn dương xung đồng hồ thì dữ liệu mã hoá được dịch vào F₁, đồng thời trạng thái của mỗi FF cũng dịch đến FF tiếp theo. Giả sử từ mã đầu vào là 1011, dưới tác dụng của xung dịch, thì hình hướng dịch của từ mã trong bộ ghi dịch như bảng 6.3.1. Có thể thấy



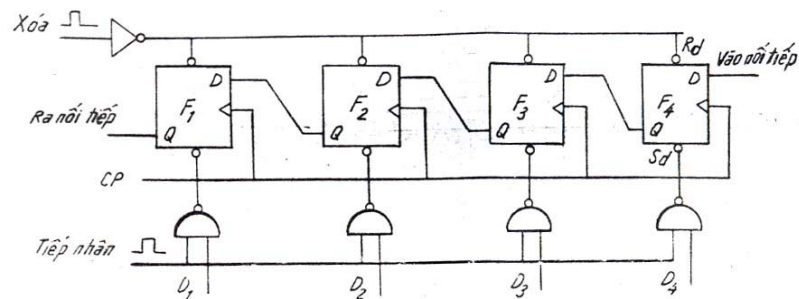
Hình 6-3-5. Bộ ghi dịch một hướng.

rằng, sau khi có 4 xung CP, từ mã 4 bit 1011 đã dịch vừa hết vào bộ ghi dịch. Lúc này, có thể lấy ra 4 bit song song của từ mã 1011 từ 4 đầu ra Q của các FF.

Bảng 6-3-1. TÌNH HUỐNG DỊCH TỪ MÃ RTONG BỘ GHI DỊCH

CP	Từ mã trong bộ ghi dịch			
Thứ tự	F1	F2	F3	F4
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	0	1	0
4	1	1	0	1

Flip Flop cuối cùng (đầu ra Q) có thể làm đầu ra nối tiếp của từ mã. Muốn thế chỉ cần qua thời gian 4 xung đồng hồ nữa thì 4 bit nối tiếp của từ mã được dịch ra ở đây. Đó là phương pháp đưa ra các bit nối tiếp. Vậy mạch điện hình 6-3-5 được gọi là bộ ghi dịch một hướng vào nối tiếp, ra nối tiếp / song song.



Hình 6-3-6. Bộ ghi dịch một hướng vào nối tiếp/ song song.

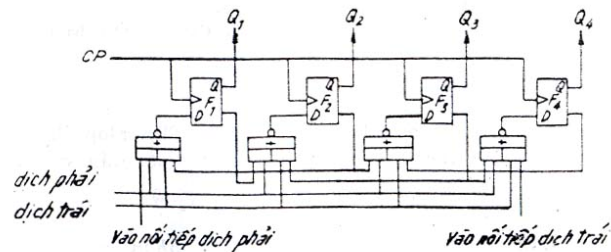
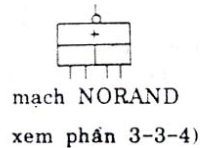
Đầu vào bộ ghi dịch cũng có thể dùng cách vào song song – Hình 6-3-6 là bộ ghi dịch vào nối tiếp / song song, ra nối tiếp.

Khi vào song song thì dùng cách tiếp nhận ra nối tiếp hai nhịp. Nhịp thứ nhất : xung xoá đưa đến đầu R_d xoá tất cả các FF, nhịp thứ hai : xung tiếp nhận đưa đến mở các cổng để tiếp nhận dữ liệu ở đầu S_d .

Trong một chip IC hiện nay, bộ ghi dịch có từ mã dài nhiều bit.

2) Bộ ghi dịch hai hướng

(ghi chú :

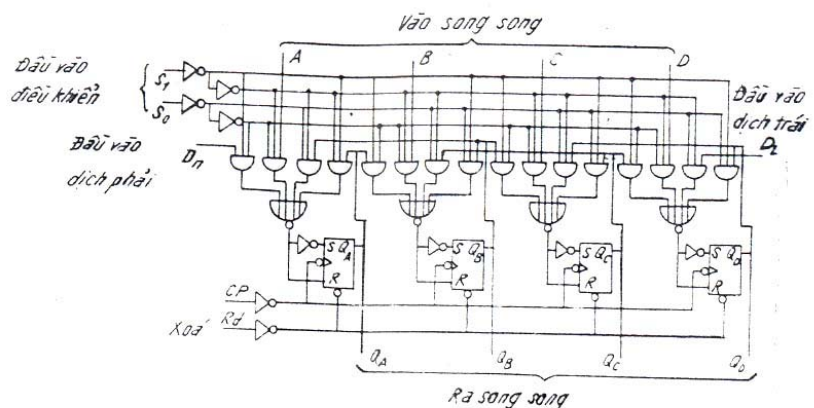


Hình 6-3-7. Bộ ghi dịch hai hướng.

Xét mạch hình 6-3-7. Dữ liệu có thể tuần tự dịch từ F_1 đến F_4 (dịch phải), cũng có thể dịch từ F_4 đến F_1 (dịch trái), vậy chúng ta gọi đây là bộ ghi dịch hai hướng.

Khi tín hiệu dịch phải bằng 1, cổng AND bên trái của mạch cấu NORAND mở thông đầu $\bar{Q}$ của FF bên trái thông qua NORAND (đã đảo mức logic) vào đầu vào D của FF bên phải (tương đương đầu Q của FF trái nối vào D của FF bên phải liền kề). Vậy khi xung dịch phải xuất hiện, dữ liệu được dịch từ trái sang phải. Ngược lại, khi tín hiệu dịch trái bằng 1, thì dữ liệu dịch từ phải sang trái. Đầu vào là tín hiệu đảo, qua mạch NORAND đảo pha, tín hiệu được lưu giữ trong bộ nhớ. Nếu tín hiệu đầu vào không đảo, thì tín hiệu được lưu giữ là đảo.

3) Vi mạch bộ ghi dịch MSI



Hình 6-3-8. Bộ ghi dịch MSI

Hình 6-3-8. Bộ ghi dịch MSI

Hình 6-3-8 là vi mạch bộ ghi dịch MSI 4 bit hai hướng tính năng mạch. Mạch điện dùng phần tử nhớ là Flip Flop RS master slave mắc thành FFD, có hai đầu D_R (dịch phải) và đầu vào dữ liệu nối tiếp D_I (dịch trái), một đầu vào xoá, một đầu vào đồng hồ CP, 4 đầu ra Q_A Q_B Q_C Q_D .

4 mạch NORAND làm thành bộ chọn lựa 4 đường thông. Các tín hiệu S_0 , S_1 điều khiển sự chọn lựa.

Phần tử nhớ là Flip Flop D, có phương trình đặc trưng:

$$Q^{n+1} = D$$

Khi $S_0, S_1 = 00$, từ sơ đồ logic, ta có:

$$Q_A^{n+1} = \overline{\overline{Q}}_A^n = Q_A^n, Q_B^{n+1} = \overline{\overline{Q}}_B^n = Q_B^n, Q_C^{n+1} = \overline{\overline{Q}}_C^n = Q_C^n; Q_D^{n+1} = \overline{\overline{Q}}_D^n = Q_D^n \quad (6-3-1)$$

Xung đồng hồ xuất hiện thì bộ nhớ giữ nguyên trạng thái.

Khi $S_0 S_1 = 01$, ta có :

$$Q_A^{n+1} = \overline{\overline{D}}_L = D_L, Q_C^{n+1} = \overline{\overline{Q}}_D^n = Q_D^n; Q_D^{n+1} = \overline{\overline{Q}}_C^n = Q_C^n, Q_A^{n+1} = \overline{\overline{Q}}_B^n = Q_B^n \quad (6-3-2)$$

Với tác dụng của xung đồng hồ, bộ nhớ công tác ở chế độ dịch trái tín hiệu vào nối tiếp.

Khi $S_0 S_1 = 11$, ta có

$$Q_D^{n+1} = \overline{\overline{D}} = D, Q_C^{n+1} = \overline{\overline{C}} = C; Q_B^{n+1} = \overline{\overline{B}} = B, Q_A^{n+1} = \overline{\overline{A}} = A \quad (6-3-3)$$

Khi xuất hiện xung đồng hồ, bộ nhớ tiếp nhận tín hiệu đầu vào song song.

Khi $S_0 S_1 = 10$, ta có:

$$Q_A^{n+1} = \overline{\overline{D}}_R = D_R, Q_B^{n+1} = \overline{\overline{Q}}_A^n = Q_A^n, Q_C^{n+1} = \overline{\overline{Q}}_B^n = Q_B^n, Q_D^{n+1} = \overline{\overline{Q}}_C^n = Q_C^n \quad (6-3-4)$$

Với tác dụng của xung đồng hồ, bộ nhớ công tác ở chế độ dịch phải tín hiệu vào nối tiếp.

Nếu ở đầu vào R_d có xung dương, thì bộ nhớ bị xoá về 0. bảng 6-3-2 tóm tắt chức năng bộ ghi dịch đã giới thiệu như trên.

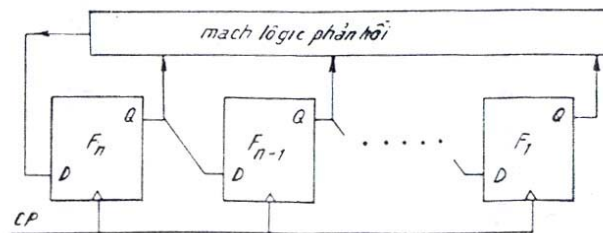
Bảng 6-3-2 BẢNG CHỨC NĂNG

Chức năng	R_d	S_0	S_1	CP
Xoá về 0	H	X	X	X
giữ nguyên trạng	L	L	L	↑
dịch trái	L	L	H	↑
vào dữ liệu song	L	H	H	↑
song dịch phải	L	H	H	↑

6.3.3. Ví dụ về ứng dụng bộ nhớ

Bộ nhớ, đặc biệt bộ ghi dịch, có ứng dụng rất rộng rãi : biến đổi từ mã nhị phân từ các bit nối tiếp thành các bit song song và ngược lại, dùng để cấu trúc bộ đếm kiểu ghi dịch rất tiện lợi v.v... Ở đây trình bày làm ví dụ về bộ đếm kiểu ghi dịch.

Nếu đem tín hiệu đầu ra của bộ ghi dịch phản hồi theo một cách xác định đến đầu vào nối tiếp D_n (đầu vào đồng bộ của Flip Flop D F_n) thì có thể cấu trúc thành bộ đếm cho nhiều mã đặc thù khác nhau. Hình 6-3-9 giới thiệu một cấu trúc như vậy. Dùng mạch ghi logic phản hồi khác nhau thì có được các bộ đếm khác nhau. Dưới đây trình bày mấy mạch điện thường dùng.



Hình 6-3-9. Bộ đếm kiểu ghi dịch

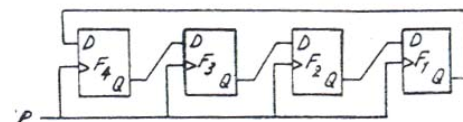
1) Bộ đếm vòng

a) Cấu trúc mạch điện

Lấy $D_n = Q_1$, tức là nối đầu ra Q_1 của F_1 vào đầu vào D_n của F_n . Vậy khi đó các Flip Flop tạo thành mạch vòng, nên có tên là

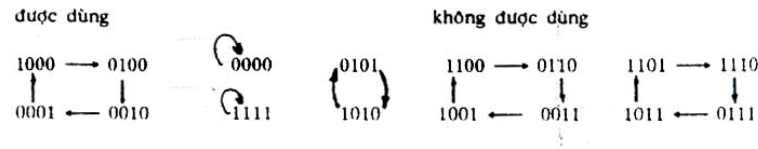
bộ đếm vòng ; thực chất là bộ ghi dịch tự tuần hoàn. Hình 6-3-10 là bộ đếm vòng $n = 4$.

b) Nguyên lý công tác



Hình 6-3-10. Bộ đếm vòng

Bằng phương pháp phân tích logic, ta có thể tìm được đồ hình trạng thái của bộ đếm vòng, như hình 6-3-11

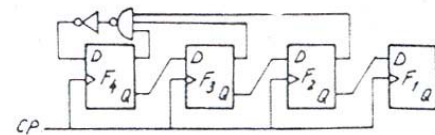


Hình 6-3-11. Đồ hình trạng thái của bộ đếm vòng

Nhận xét hình 6-3-11: khi tác dụng xung đếm đầu vào CP, mạch tuần hoàn dịch 1, cũng có thể tuần hoàn dịch 0. nếu chọn dùng tuần hoàn dịch 1 thì các trạng thái của bộ đếm là 1000, 0100, 0010, 0001. Khi công tác, đầu tiên nên dùng xung khởi động đưa bộ đếm vào trạng thái sử dụng, ví dụ 1000, sau đó hãy tác động CP vào.

c) Vấn đề tự khởi động

Đồ hình trạng thái cho chúng ta biết rằng bộ đếm loại này không thể tự khởi động giá như chịu tác động của nhiễu hay sự cố nguồn điện mà mạch rơi vào trạng thái không được dùng thì bộ đếm sẽ mãi mãi nằm trong vòng tuần hoàn không được dùng ; chỉ có khởi động lại thì mạch mới có thể trở về trạng thái được dùng.



Hình 6-3-12. Bộ đếm vòng 4 bit tự khởi động

Hình 6-3-12 là bộ đếm vòng 4 bit có thể tự khởi động từ sơ đồ logic ta có phương trình kích

$$\begin{aligned} D_4 &= \overline{Q}_4^n \overline{Q}_3^n \overline{Q}_2^n & D_3 &= Q_4^n \\ D_2 &= Q_3^n & D_1 &= Q_2^n \end{aligned} \quad (6-3-5)$$

Thay các giá trị trên vào phương trình đặc trưng của Flip Flop D, ta có :

$$\begin{aligned} Q_4^{n+1} &= \overline{Q}_4^n \overline{Q}_3^n \overline{Q}_2^n & Q_3^n &= Q_4^n \\ Q_2^{n+1} &= Q_3^n & Q_1^{n+1} &= Q_2^n \end{aligned} \quad (6-3-6)$$

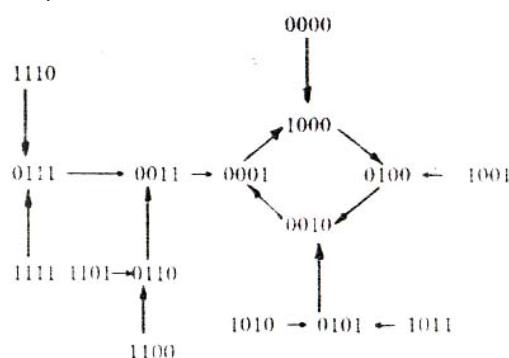
Giả định trạng thái đầu là 0000, tuần tự thay vào (6-3-6) tiến hành tính, ta được bảng 6-3-3

Bảng 6-3-3. BẢNG CHỨC NĂNG

Q_4^n	Q_3^n	Q_2^n	Q_1^n	Q_4^{n+1}	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}
0	0	0	0	1	0		0
0	0	0	1	1	0		0
0	0	1	0	0	0		1
0	0	1	1	0	0		1
0	1	0	0	0	0		0
0	1	0	1	0	0		0
0	1	1	0	0	0		1
0	1	1	1	0	0		1
1	0	0	0	0	1		0
1	0	0	1	0	1		0
1	0	1	0	0	1		1
1	0	1	1	0	1		1
1	1	0	0	0	1		0
1	1	0	1	0	1		0
1	1	1	0	0	1		1
1	1	1	1	0	1		1

Hình 6-1-13 là đồ hình trạng thái của mạch hình 6-3-12

Hình 6-3-13. Đồ hình trạng thái của bộ đếm vòng có thể tự khởi động.



d) Đặc điểm

ưu điểm của bộ đếm vòng : tất cả các Flip Flop bình thường chỉ chọn một Flip Flop ở trạng thái 1 (những Flip Flop còn lại ở trạng thái 0). Vậy có thể lấy đầu ra của bộ đếm vòng từ đầu Q của các Flip Flop mà không cần bộ giải mã. Khi liên tục có xung CP đầu vào, các đầu ra Q, Q của các Flip Flop sẽ cho ra xung vuông luân lưu. Vì thế mạch điện này còn gọi là bộ phân phối xung vòng.

Nhược điểm : hiệu suất sử dụng trạng thái là thấp, nhớ N số cần N Flip Flop (cần dùng nhiều Flip Flop).

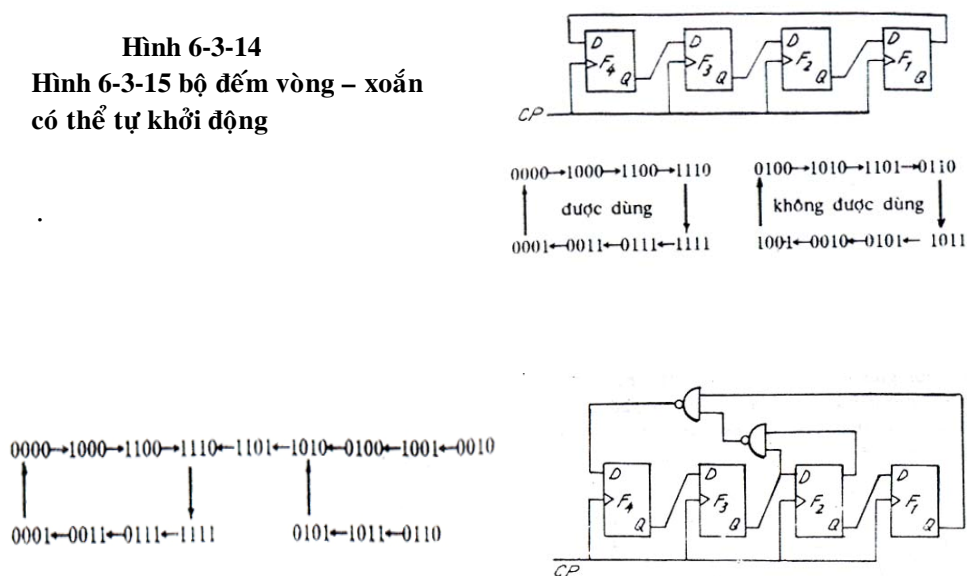
2) Bộ đếm vòng xoắn

Đặc điểm cấu trúc của bộ đếm vòng xoắn là

$$D_n = Q_1^n \quad (6-3-7)$$

Xem sơ đồ logic và đồ hình trạng thái của mạch đếm vòng – xoắn 4 bit hình 6-3-14 có 8 trạng thái được dùng. Có 8 trạng thái không được dùng. Mạch không thể tự khởi động. Trước khi đếm, cần thiết lập trạng thái 0000 cho bộ đếm vòng – xoắn.

Hình 6-3-14
Hình 6-3-15 bộ đếm vòng – xoắn
có thể tự khởi động



Hình 6-3-15. Bộ đếm vòng xoắn 4 bit có thể tự khởi động.

Đặc điểm của bộ đếm vòng – xoắn là chỉ có một Flip Flop duy nhất chuyển đổi trạng thái trong mỗi lần đếm; vậy không tồn tại nguy hiểm chấu đua khi giải mã, hơn nữa các cổng giải mã đều chỉ cần hai đầu vào. Nhược điểm của bộ đếm vòng – xoắn vẫn là chưa tận dụng hết các trạng thái bộ đếm. Trong bộ đếm n bit ($n \geq 3$) thì có $2^n - 2n$ trạng thái không được dùng.

2) Bộ đếm kiểu ghi dịch độ dài cực đại

Ý nói bộ đếm kiểu ghi dịch có độ dài đếm $N = 2^n - 1$. mạch logic phản hồi của bộ đếm này dùng cổng XOR. Bảng 6-3-4 là logic phản hồi khi $n = 3 \div 12$

Bảng 6-3-4. LOGIC PHẢN HỒI CỦA BỘ ĐẾM KIỂU GHI DỊCH ĐỘ DÀI CỰC ĐẠI

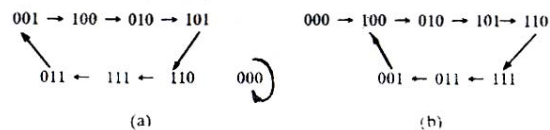
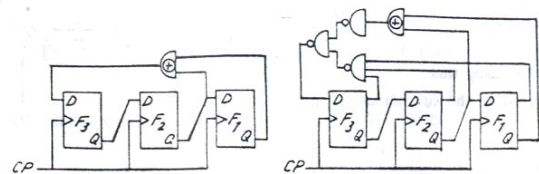
Bộ ghi dịch n bit	Logic phản hồi
3	$D_3 = Q_1 \oplus Q_2, Q_1 \oplus Q_3$

4	$D_4 = Q_1 \oplus Q_2, Q_1 \oplus Q_4$
5	$D_5 = Q_1 \oplus Q_3, Q_1 \oplus Q_4$
6	$D_6 = Q_1 \oplus Q_2, Q_1 \oplus Q_6$
7	$D_7 = Q_1 \oplus Q_2, Q_1 \oplus Q_7$
8	$D_8 = Q_1 \oplus Q_3, Q_4 \oplus Q_5, Q_1 \oplus Q_5 \oplus Q_6 \oplus Q_7$
9	$D_9 = Q_1 \oplus Q_5, Q_1 \oplus Q_6$
10	$D_{10} = Q_1 \oplus Q_4, Q_1 \oplus Q_8$
11	$D_{11} = Q_1 \oplus Q_3, Q_1 \oplus Q_{10}$
12	$D_{12} = Q_1 \oplus Q_2, Q_5 \oplus Q_7, Q_1 \oplus Q_7 \oplus Q_9 \oplus Q_{12}$

Hình 6-3-16 là sơ đồ logic và đồ hình trạng thái của bộ đếm kiểu ghi dịch độ dài cực đại 3 bit. Bất kể nhiều ít 0, logic cổng XOR là 0. Vậy trong bộ đếm này, trạng thái toàn 0 không được dùng, cấu thành vòng không được dùng.

Hình 6-3-16. Bộ đếm kiểu ghi dịch độ dài cực đại 3 bit.

- a) Không thể tự khởi động;
- b) Có thể tự khởi động.



Trong bộ đếm kiểu ghi dịch độ dài cực đại thì các trạng thái của bộ đếm tương đối được tận dụng, logic phản hồi rất đơn giản. Vậy nó kinh tế hơn bộ đếm nhị phân đồng bộ. Nhất là khi độ dài đếm khá dài thì ưu điểm này càng rõ rệt.

3) Thực hiện phương pháp tự khởi động của bộ đếm điều khiển ghi dịch\

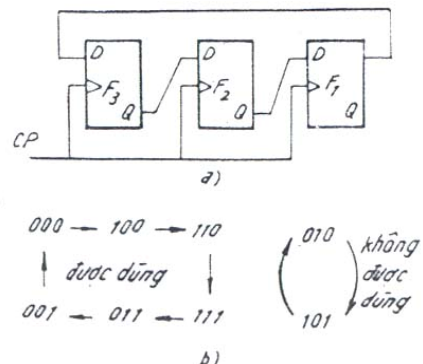
Các mạch cơ bản của bộ đếm : vòng, vòng – xoắn, kiểu độ dài cực đại v.v... đều không thể tự khởi động. Muốn giữ nguyên các trạng thái được dùng của các bộ đếm, mà mạch có thể tự khởi động, ta phải giải quyết vấn đề đó như dưới đây.

a) Các bước cơ bản:

-Vẽ ra mạch điện cơ bản và đồ hình trạng thái của nó.

Hình 6-3-17. Bộ đếm vòng – xoắn 3 bit.

- a) Sơ đồ logic;
- b) Đồ hình trạng thái.



- Sửa đổi quan hệ chuyển đổi trạng thái không được dùng, thực hiện tự khởi động của đồ hình trạng thái.

- Tìm logic phản hồi

b) Ví dụ : thiết kế bộ đếm vòng – xoắn 3 bit có thể tự khởi động.

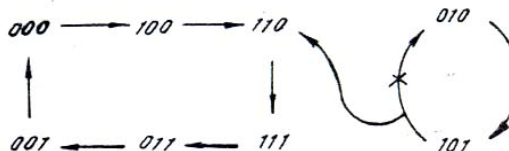
Vẽ ra mạch điện cơ bản và đồ hình trạng thái của nó

Xem hình 6-3-17

Sửa đổi vòng trạng thái không được dùng, thực hiện tự khởi động

Công việc sửa đổi là cắt đứt vòng trạng thái không được dùng, nối ghép trạng thái không được dùng tại chỗ đứt vào trạng thái được tương ứng để thực hiện tự khởi động. Xem hình 6-3-18.

Hình 6-3-18. Đồ hình trạng thái của mạch có thể tự khởi động.



Khi ghép nối trạng thái không được dùng cần chú ý rằng từ Flip Flop tương ứng bit cao nhất F_n đến Flip Flop tương ứng bit thấp nhất F_1 có quan hệ ghi dịch bit cố định; trạng thái hiện tại của bit cao hoàn toàn quyết định trạng thái kế tiếp của bit thấp; chúng ta chỉ sửa đổi được trạng thái kế tiếp của F_n (trong ví dụ này là F_3). Căn cứ vào vòng trạng thái kế tiếp của F_3 là 1. Nếu bộ đếm lạc vào vòng trạng thái không được dùng, nó chuyển đến 101, đợi xung CP nữa thì tự động theo cách ghép nối trạng thái của chúng ta, chuyển đến 110.

Tìm logic phản hồi.

Đầu tiên căn cứ quan hệ logic cần để tự khởi động, ta vẽ ra bảng Karnaugh của Q_n^{n+1} (trong ví dụ này là Q_3^{n+1}), rồi tìm hàm logic của Q_n^{n+1} .

Tiếp theo, ta so sánh với phương trình đặc trưng của Flip Flop F_n , tức là tìm ra logic phản hồi.

Từ hình 6-3-18 ta vẽ bảng Karnaugh hình 6-3-19

$Q_3^n \backslash Q_2^n Q_1^n$	00	01	11	10
0	1	0	0	x

Hình 6-3-19. Bảng Karnaugh của Q_3^{n+1}

Từ hình 6-3-19, ta có:

$$Q_3^{n+1} = \overline{Q_1^n} + Q_3^n \overline{Q_2^n} \quad (6-3-8)$$

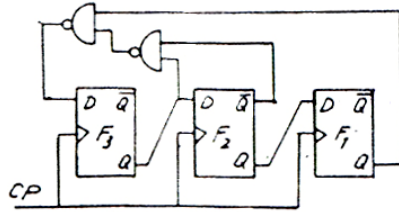
Phương trình đặc trưng của Flip Flop D là:

$$Q^{n+1} = D$$

$$\text{Vậy } D_3 = \overline{Q_1^n} + Q_3^n \overline{Q_2^n} \quad (6-3-9)$$

Vẽ sơ đồ logic

$$D_3 = \overline{\overline{Q_1^n} + Q_3^n \overline{Q_2^n}} = Q_1^n \overline{Q_3^n} \overline{Q_2^n} \quad \text{xem hình 6-3-20}$$



Hình 6-3-20. Bộ đếm vòng – xoắn có thể tự khởi động.

Cần nói rõ ràng, nếu mạch điện có nhiều trạng thái không được dùng, thì trong số đó có trạng thái có thể trực tiếp chuyển thành trạng thái được dùng, những trạng thái khác có thể qua một hay nhiều trạng thái không được dùng trung gian trước khi chuyển thành trạng thái được dùng. (Thể hiện trong đồ hình trạng thái tự khởi động).

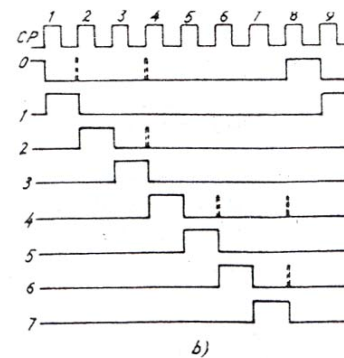
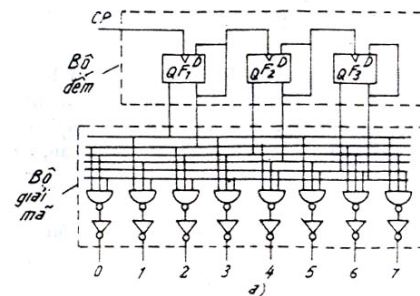
6.4. BỘ TẠO XUNG TUẦN TỰ

Trong các thiết bị điều khiển kỹ thuật số và máy tính số thường yêu cầu các phép toán và thao tác phải tiến hành tuần tự theo quy định trước một cách chính xác. Do đó phần điều khiển không chỉ tạo ra các tín hiệu điều khiển đó. Phương pháp hay dùng nhất là sử dụng bộ tạo xung tuần tự (bộ phát xung nhịp) để tạo ra các xung tuần tự có thứ tự thời gian trước sau; nhưng cung này làm chuẩn thời gian cho sự hiệp đồng của các bộ phận trong toàn máy.

Bộ tạo xung tuần tự thông thường bao gồm bộ đếm và bộ giải mã như hình 6-4-1a.

Xung đồng hồ thời gian chuẩn được đưa vào đầu vào bộ đếm. Nhờ bộ giải mã sẽ biến đổi các trạng thái bộ đếm thành xung tuần tự trên các đầu ra như hình 6-4-1b.

Bộ đếm hình 6-4-1a là dị bộ. Khi xuất hiện xung đồng hồ thì các FF không lật đồng thời mà có trước có sau so với nhau. Ngoài ra, trong mỗi lần chuyển đổi trạng thái, có thể có nhiều FF cùng lật, nên sinh ra hiện tượng nguy hiểm: chạy đua. Do đó,



Hình 6-4-1. Bộ tạo xung tuần tự.

a) Sơ đồ logic

b) Dạng sóng

tạo ra xung nhiều ở đầu ra bộ giải mã (của bộ tạo xung tuần tự); xung nhiều đôi khi được gọi là tạp âm quá độ. Ví dụ, như ở hình 6-4 1b: khi bộ đếm chuyển từ 001 thành 010, giả sử F_1 lật sang 0 trước F_2 lật thành 1, trong thời gian quá độ sẽ xuất hiện trạng thái 000 (ngắn ngủi).

Do đó, có xung hẹp sinh ra trên đầu ra “0”. Tương tự, khi từ 011 thành 100 thì cũng có thể xuất hiện xung nhiều quá độ trên đầu ra “0”. Các đầu ra khác cũng có thể xuất hiện xung nhiều quá độ. Chẳng hạn, trên đây “2” khi từ 101 thành 110...

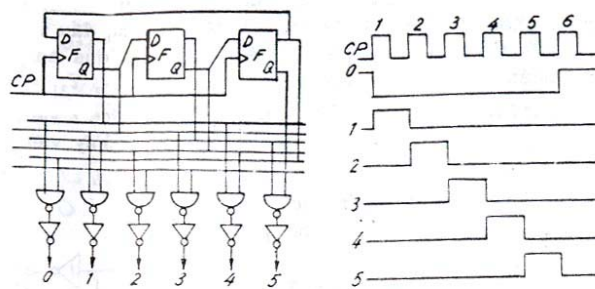
Có những phương pháp sau đây để trừ bỏ xung nhiều quá độ :

1) Dùng bộ đếm vòng. Cúng ta đã xem xét mạch điện bộ đếm vòng ở tiết 6-3-3. Vì đầu ra Q của mỗi Flip Flop đã là xung tuần tự, không cần bộ giải mã, nên tuy rằng có thể có nhiều FF lật khi bộ đếm chuyển đổi, vẫn không thể sinh ra xung nhiều quá độ.

Như ta đã, nhược điểm của mạch này là không kinh tế: cứ tăng thêm một đầu ra phải có thêm một Flip Flop .

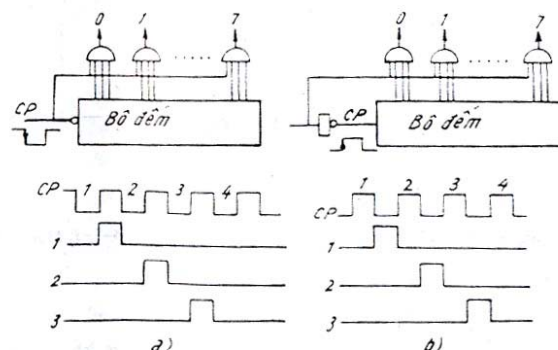
2) Dùng bộ đếm mỗi lần trị số đếm thay đổi chỉ có một Flip Flop lật. Thuộc về loại này là bộ đếm dùng mã Gray và bộ đếm vòng – xoắn (bộ đếm Johnson). Ví dụ: hình 6-4-2.

Hình 6-4-2. Bộ tạo xung tuần tự dùng bộ đếm vòng – xoắn
a) sơ đồ logic; b) Dạng sóng



3) Ứng dụng phương pháp trừ bỏ hiện tượng nguy hiểm “chạy đua” đã giới thiệu ở tiết 4.9.2. Ví dụ, dùng xung đếm đầu vào để khoá cổng giải mã.

Hình 6-4-3. Dùng xung đồng hồ để khoá cổng giải mã trừ bỏ “chạy đua”.
a) Kích bằng sườn âm;
b) Kích bằng sườn dương



Khi kích FF bằng sườn âm CP, dùng CP khoá cổng giải mã như hình 6-4-3a. Khi kích FF bằng sườn dương CP thì dùng $\overline{CP}$ khoá cổng giải mã như hình 6-4-3b. Xung đầu ra bây giờ tuy vẫn tuần tự theo thời gian, nhưng không còn nối sát nhau nữa.

Sử dụng bộ đếm đồng bộ có vẻ không sinh ra nhiều, vì trên lí thuyết cho là tất cả các FF đều lật đồng thời. Thực ra, các FF đều không thể hoàn toàn như nhau, phụ tải và dây nối lắp ráp của các FF cũng không hoàn toàn giống nhau, nghĩa là các FF không hoàn toàn đồng thời cùng lật, cũng tức là không thể triệt để trừ bỏ “chạy đua” được.

6.5. BỘ NHỚ RAM VÀ DỤNG CỤ GHÉP ĐIỆN TÍCH CCD

6.5.1. Bộ nhớ RAM (Random Access Memory – bộ nhớ đọc / viết)

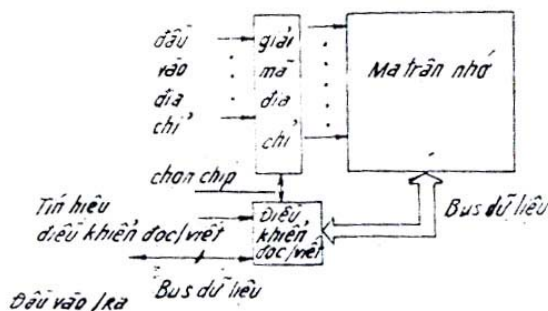
RAM là một phần không thể thiếu của máy tính điện tử số. Trước đây, RAM là bộ nhớ bằng xuyên từ. Hiện nay, RAM là bộ nhớ bán dẫn, ; RAM bán dẫn có ưu điểm tốc độ lớn, thể tích nhỏ, dung lượng lớn, tiết kiệm điện năng và độ tin cậy cao.

1) Kết cấu của RAM bán dẫn

Trong máy tính điện tử số, cả chương trình và số liệu đều biểu thị bằng số nhị phân – RAM bán dẫn là vi mạch (IC) cỡ lớn, bao gồm hàng trăm, nghìn phần tử nhớ để lưu giữ số nhị phân đó. Như ra đã biết, một phần tử nhớ có thể nhớ 1 số nhị phân (bit). Căn cứ vào số bit, RAM có 2 hình thức kết cấu nhiều từ 1 bit, mỗi tổ chức cơ sở của bộ nhớ chỉ nhớ 1 bit. Ví dụ, RAM 1024 x 1 là bộ nhớ có 1024 phần tử nhớ 1 bit. Trong kết cấu nhiều từ nhiều bit, mỗi tổ chức cơ sở của bộ nhớ lưu giữ nhiều bit. Ví dụ, RAM 256 x 4 là bộ nhớ có 256 phần tử nhớ 4 bit. Tích số của số phần tử cơ sở và số bit mỗi phần tử đó là dung lượng của bộ nhớ RAM. Ví dụ, RAM 16384 x 1 có dung lượng 16 Kbit (1 Kbit = 1024 bit). Nói chung RAM bán dẫn có cấu trúc nhiều từ 1 bit.

Hình 6-5-1 là sơ đồ khối điển hình của RAM, nó bao gồm các khối sau:

Hình 6-5-1. Kết cấu của RAM.



a) Bộ giải mã địa chỉ. Trong RAM có rất nhiều phần tử nhớ; để phân biệt, từng phần nhớ được gán một địa chỉ. Mỗi lần đọc hoặc viết chỉ có thể làm việc với một phần tử nhớ có địa chỉ đã cho. Hoặc là viết vào phần tử xét, hoặc là đọc ra nội dung đã viết vào trước đó, quá trình này gọi là truy nhập xuất bộ nhớ. Mã nhị phân biểu

thị địa chỉ cần truy nhập truy xuất được đưa vào bộ giải mã địa chỉ. Sau khi đã giải mã, ở đầu ra ta có tín hiệu tích cực một dây tương ứng để mở thông đường vào/ra cho tín hiệu dữ liệu được viết/đọc vào phần tử nhớ có địa chỉ đó.

b) Điều khiển đọc/viết. Đối với phần tử nhớ được chọn theo địa chỉ, sự đọc ra hay viết vào tùy thuộc sự điều khiển của tín hiệu điều khiển đọc/viết. Ví dụ, R/W, mức logic 1 là đọc, mức logic 0 là viết. Cũng có RAM có hai trường dây tín hiệu riêng để điều khiển đọc/viết.

c) Đầu vào/ra. Bằng các đầu vào/ra mà RAM trao đổi tin tức với bộ xử lý trung ương (CPU-Central Processing Unit). Các đường dây dữ liệu là hai chiều, do tín hiệu điều khiển đọc/viết mà khi đọc, nó là đầu ra; khi viết, nó là đầu vào. Số đầu vào/ra này phụ thuộc vào số bit trong một địa chỉ. Ví dụ, RAM 1024 x 1 mỗi địa chỉ là của một phần tử nhớ nên tương ứng chỉ có một đầu vào/ra. RAM 256 x 4 mỗi địa chỉ là của 1 nhóm 4 phần tử nhớ nên tương ứng có 4 đầu vào/ra. Cũng có RAM dây dữ liệu vào riêng, dây dữ liệu ra riêng. Đầu ra nói chung là mạch ra 3 trạng thái hay mạch hở cực góp (colectơ)

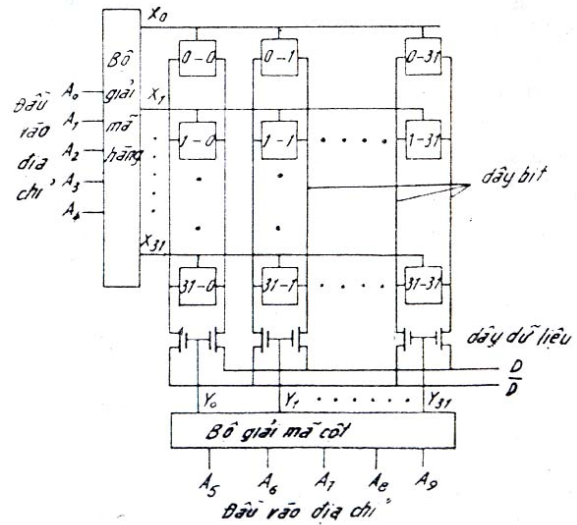
d) Điều khiển chọn chip. Do sự hạn chế về khả năng tích hợp RAM trên mỗi chip, bộ nhớ RAM của máy tính phải do nhiều chip RAM ghép nối logic với nhau. Khi CPU truy nhập truy xuất bộ nhớ theo địa chỉ thì địa chỉ đó thường tương ứng tín hiệu chọn chip dẫn vào chip RAM nào ở mức tích cực thì chip nhằm mục đích đó. Nếu tín hiệu chọn chip dẫn vào chip RAM nào ở mức tích cực thì chip đó được chọn. Tín hiệu đầu ra bộ giải mã của chip xét sẽ điều khiển mở thông đường vào/ra chip RAM khác không có tín hiệu chọn chip ở mức tích cực thì không được chọn, nghĩa là bị ngắt khỏi liên hệ với CPU)

e) Ma trận nhớ. Các phần tử nhớ của RAM thường được bố trí dạng ma trận, gọi là ma trận nhớ. Đầu ra bộ giải mã địa chỉ điều khiển sự nối/ ngắt

giữa phần tử nhớ trong ma trận với đầu vào/ra của bộ nhớ. Phần tử nhớ tương ứng với địa chỉ được giải mã được nối thông, những phần tử khác bị ngắt.

Hình 6-5-2 là ma trận nhớ và bộ giải mã địa chỉ của RAM thường được bố trí dạng ma trận, gọi là ma trận nhớ. Đầu ra bộ giải mã địa chỉ điều khiển sự nối/ngắt giữa phần tử nhớ trong ma trận với đầu vào/ra của bộ nhớ. Phần tử nhớ tương ứng với địa chỉ được giải mã được nối thông, những phần tử khác bị ngắt.

Hình 6-5-2 là ma trận nhớ và bộ giải mã địa chỉ của RAM 1024 x 1. Ma trận này có hình thức kết cấu nhiều từ 1 bit, cấu trúc thành ma trận 32 x 32, mỗi khối vuông nhỏ trong hình



Hình 6-5-2. Ma trận nhớ RAM 1024x 1 bit

vẽ là một phần tử nhớ; sự nối thông với mạch ngoài của phần tử nhớ do tín hiệu đầu ra bộ giải mã địa chỉ điều khiển. Có 10 dây địa chỉ, tương ứng 1024 địa chỉ của 1024 phần tử nhớ. Giải mã địa chỉ phân làm hai: bộ giải mã hàng và bộ giải mã cột. Các dây địa chỉ $A_0, \dots, A_4$ là đầu vào bộ giải mã hàng. Đầu ra bộ giải mã hàng là 32 dây $X_0, X_1, \dots, X_{31}$ là các dây chọn hàng; các dây địa chỉ $A_5, \dots, A_9$ là đầu vào bộ giải mã cột. Đầu ra bộ giải mã cột là 32 dây $Y_0, Y_1, \dots, Y_{31}$ là các dây chọn cột. Dây chọn hàng X_i điều khiển sự nối thông của mỗi phần tử nhớ trong hàng tương ứng của ma trận nhớ với dây bit. Dây chọn cột Y_j điều khiển sự nối thông của dây bit mỗi cột tương ứng với dây dữ liệu $D, \bar{D}$. Ví dụ, khi $X_0 = 1, Y_0 = 1$, 32 phần tử nhớ 0-0 ÷ 0-31 của hàng thứ nhất trong ma trận nhớ được nối thông với dây bit tương ứng. Trong 32 đôi dây bit thì chỉ có đôi dây được Y_0 điều khiển mới nối thông với dây dữ liệu. Vậy chỉ có mỗi phần tử nhớ 0-0 là được nối thông với dây dữ liệu. Phần tử nhớ 0-0 trên đây được đọc hay được viết, vấn đề này do tín hiệu dây đọc/viết điều khiển. Nếu là đọc thì tin tức lưu trữ trong phần tử nhớ 0-0. Tương tự, Khi $X_{31} = 1$ thì phần tử nhớ 31-31 được nối thông với dây dữ liệu, với điều khiển đọc/viết, CPU thông qua BÚ dữ liệu, đầu vào/ra, dây dữ liệu thực hiện đọc ra hay viết vào phần tử nhớ 31-31.

Hình 6-5-3 là mạch điện tương đối đơn giản dùng để điều kiện đọc/viết. Các cổng 1 ÷ 5 làm thành bộ phận điều khiển đọc/viết.

Khi $\overline{CS} = 0$; $R/W = 1$ thì $\overline{K_R} = 0$ $\overline{K_W} = 1$ thực hiện đọc.

Khi $\overline{CS} = 0$; $R/W = 0$ thì $\overline{K_R} = 1$ $\overline{K_W} = 1$ thực hiện viết.

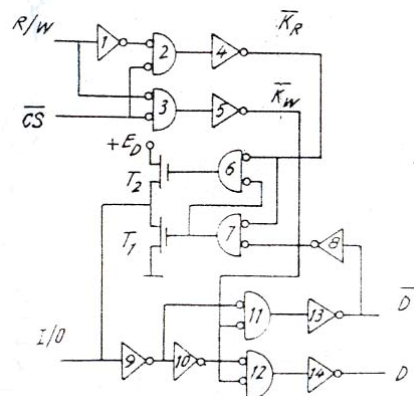
Các cổng 6,7 và bán dẫn MOS T_1, T_2 làm thành mạch đầu ra 3 trạng thái.

Khi $\overline{K_R} = 0$; $\overline{K_W} = 1$ thì cổng 11, 12 ngắt, cổng 6,7 sẵn sàng. Dữ liệu $\overline{D}$ qua cổng 8 đảo pha, qua mạch 3 trạng thái rồi ra đầu ra I/O : $I/O = \overline{D} = D$.

Vậy mạch đã thực hiện đọc.

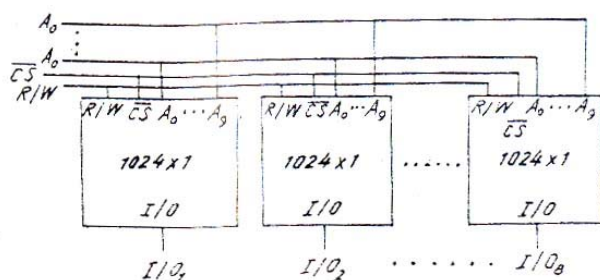
Khi $\overline{K_R} = 1$; $\overline{K_W} = 0$ thì các cổng 6,7 bị ngắt, đầu ra của chúng là mức thấp làm cho bán dẫn T_1, T_2 bị khoá, mạch ra 3 trạng thái thể hiện trở kháng cao. Trong khi đó các cổng 11, 12 sẵn sàng, đường vào cho tín hiệu dữ liệu được mở thông qua các cổng 9, 10, 11, 12, 13, 14 với $D = I/O$ và $\overline{D} = \overline{I/O}$, tức thực hiện viết dữ liệu vào bộ nhớ.

Khi $\overline{CS} = 1$; $\overline{K_R} = 1$; $\overline{K_W} = 1$, nghĩa là khi chip RAM không được chọn thì mạch 3 trạng thái thể hiện trở kháng cao và các cổng 11, 12, cũng bị ngắt, I/O bị ngắt khỏi $D, \overline{D}$, tức là không đọc, không viết: RAM đã giới thiệu ở hình 6-5-2 lưu giữ 1 bit ở 1 địa chỉ. Thực tế thường dùng lưu giữ 8 bit hay 16 bit ở địa chỉ. Vậy nên phải ghép nối song song các RAM nhiều từ 1 bit. Hình 6-5-4 dưới đây giới thiệu 8 chip RAM 1024 x 1 cấu trúc thành bộ nhớ RAM 1024 x 8. Trong hình này, các dây địa chỉ, dây đọc/viết, dây chọn chip đều nối song song, còn các dây I/O được dùng riêng rẽ.



Hình 6-5-3. Mạch điều khiển đọc/viết của RAM

Hình 6-5-4. Bộ nhớ RAM 1024 x 8 bit dùng 8 chip RAM 1024 x 1



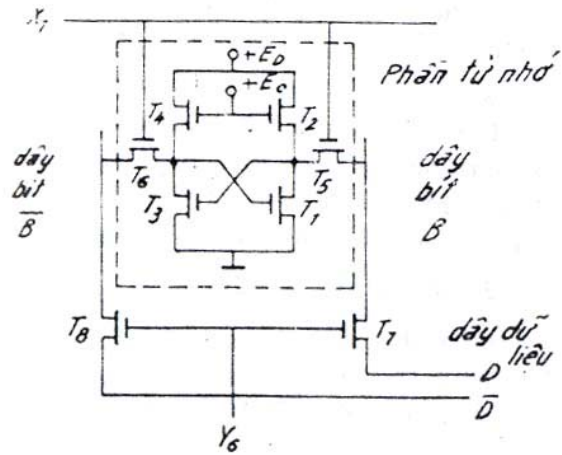
2) Phần tử nhớ RAM tĩnh

Phần tử nhớ là hạt nhân của bộ nhớ. Căn cứ sự khác biệt nguyên lí làm việc, chúng phân thành RAM tĩnh và RAM động. Căn cứ vào công nghệ sản xuất, chúng phân thành loại lưỡng cực và MOS. Vậy hình thức mạch điện rất đa dạng. Dưới đây trước tiên giới thiệu RAM tĩnh.

a) Phần tử nhớ NMOS 6

bóng bán dẫn

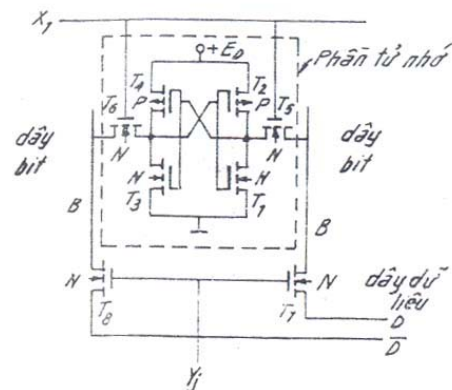
Mạch điện xem hình 6-5-5. T_1 và T_2 làm thành bộ đảo pha. T_3 và T_4 cũng làm thành một bộ đảo pha khác. Đầu vào đầu ra của hai bộ đảo pha nối chéo nhau tạo thành một Flip Flop RS cơ bản, nghĩa là cấu trúc nên phần tử nhớ. T_1 thông T_3 ngắt là trạng thái 0; T_3 thông và T_1 ngắt là trạng thái 1 – T_6 là các bóng bán dẫn điều khiển. Dây X_i điều khiển T_5, T_6 thông hoặc ngắt, do đó điều khiển sự nối thông đầu ra của Flip Flop với dây bit. Khi $X_i = 1$ thì T_5, T_6 thông, do đó FF thông với dây bit. Khi $X_i = 0$ thì T_5, T_6 ngắt, do đó FF bị ngắt khỏi dây bit. T_7, T_8 là các bóng bán dẫn điều khiển sự nối thông dây bit với dây dữ liệu bằng tín hiệu dây Y_j , nguyên lý làm việc tương tự T_5, T_6 . Không phải mỗi phần tử nhớ đều có riêng T_7, T_8 này; mà T_7, T_8 là dùng chung cho cả cột phần tử nhớ (xem hình 6-5-5). Vậy chỉ phần tử nào mà các giá trị X_c, Y_j của nó đều là 1 thì mới được nối thông với dây dữ liệu, nghĩa là mới có thể tiến hành đọc/viết đối với phần tử đó. Ta nói phần tử nhớ $i-j$ đã được chọn. Bất kì thời điểm nào thì cũng chỉ có mức 1. Nghĩa là khi truy nhập truy xuất, lúc nào cũng chỉ có một phần tử nhớ được chọn; các phần tử nhớ khác giữ nguyên trạng thái, tức lưu giữ không đổi nội dung đã ghi vào.



Hình 6-5-5. Phần tử nhớ RAM tĩnh 6 bóng bán dẫn MOS

b) Phần tử nhớ CMOS 6 bóng bán dẫn (xem hình 6-5-6). Cấu trúc và

nguyên lý công tác giống như mạch hình 6-5-5. $T_1 \div T_4$ làm thành Flip Flop RS cơ bản. T_7, T_6 là bóng điều khiển sự nối thông FF với dây bit. T_7, T_8 là bóng điều khiển sự nối thông dây bit với dây dữ liệu. Đặc điểm của mạch này là bóng bán dẫn phụ tải của bộ đảo pha loại MOS kênh P tăng cường. Phần tử nhớ CMOS chỉ cần tiêu tốn rất ít điện năng (cỡ μW). Vậy rất tiện dùng



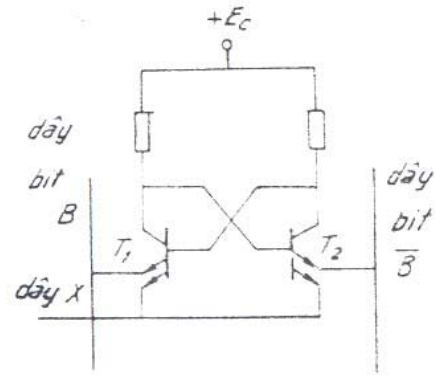
Hình 6-5-6. Phần tử nhớ RAM tĩnh 6 bóng CMOS

pin thay thế khi lưới điện bị sự cố nhằm giữ gìn thông tin đã ghi vào bộ nhớ.

c) Phần tử nhớ lưỡng cực (xem hình 6-5-7). Mạch gồm 2 tranzito nhiều emitter nối vào dây X, đôi emitter còn lại nối riêng rẽ vào dây bit B, $\overline{B}$.

Ở trạng thái lưu giữ số liệu, điện thế dây X cỡ 0,3V, điện thế dây bit cỡ 1,1V. Khi đó, dòng điện của tranzito thông sẽ chạy ra dây X, lớp tiếp giáp bán dẫn nối với dây bit phân cực nghịch làm hở mạch dây bit với FF.

Trạng thái phần tử nhớ có thể là 0 (T_1 thông, T_2 ngắt) hoặc có thể là 1 (T_2 thông, T_1 ngắt).



Hình 6-5-7. Phần tử nhớ lưỡng cực

Khi phần tử nhớ được chọn (để vào/ra dữ liệu), điện thế dây X lên cao đến cỡ 2,2V, điện thế dây bit bé hơn, nên dòng điện qua tranzito thông sẽ chạy ra dây bit.

Giả sử cần đọc ra, thì chỉ cần xác định dòng điện trên một dây bit có hay không. Chẳng hạn xét dây bit $\overline{B}$: trạng thái phần tử nhớ là 1, T_2 thông, dòng điện ra dây $\overline{B}$, qua bộ khuyết đại đọc ra điện áp mức 1; trạng thái phần tử nhớ là 0, T_2 ngắt, không có dòng điện ra dây $\overline{B}$, đầu ra bộ khuyết đại đọc không có tín hiệu (điện áp mức 0).

Giả sử cần viết vào giá trị 1, thì sau mạch đầu vào ta có $B = 1$, $\overline{B} = 0$, tương ứng T_1 ngắt, T_2 thông. Dòng điện của T_2 chạy ra dây bit $\overline{B}$. Sau khi dây X trở về mức thấp thì dòng điện của T_2 chuyển hướng chạy ra dây X, trạng thái phần tử nhớ không thay đổi, hoàn thành viết vào nó giá trị 1. Giả sử cần viết vào giá trị 0. Khi đó, $B = 0$, $\overline{B} = 1$ tương ứng T_1 thông, T_2 ngắt.

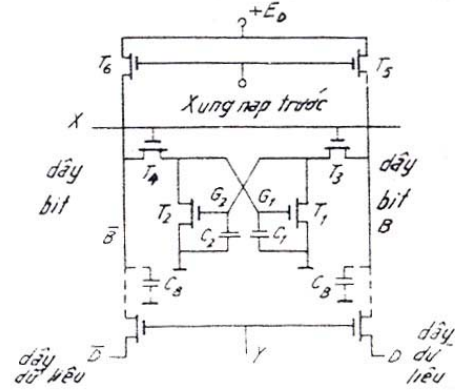
3) Phần tử nhớ RAM động

Phần tử nhớ động MOS sử dụng điện dung cực cửa (gate) của bóng bán dẫn MOS để nhớ tạm thời dữ liệu. Do có sự dò điện nên điện tích lưu giữ trên điện dung cực cửa bị suy giảm dần. Vậy để dữ liệu lưu giữ không bị mất đi, phải bù lượng điện tích bị dò mất bằng cách định kỳ nạp điện bổ sung cho điện dung cực cửa. Việc này được gọi là làm tươi dữ liệu bộ nhớ động.

Dưới đây giới thiệu mấy loại phần tử nhớ động thường gặp

a) Phần tử nhớ động 4 bóng (xem hình 6-5-8). T_1 và T_2 nối ghép chéo nhau. Dữ liệu (điện tích) lưu trữ trên C_1 và C_2 . Điện áp trên C_1 , C_2 điều khiển T_1 , T_2 thông hoặc ngắt. Khi C_1 nạp điện tích (điện áp trên C_1 lớn hơn điện áp cắt của T_1) và C_2

không có điện tích (điện áp trên C_2 nhỏ hơn điện áp cắt của T_2) thì T_1 thông, T_2 ngắt, tương ứng với trạng thái 0 của phần tử nhớ. Khi C_2 nạp điện tích, C_1 không có điện tích thì T_2 thông, T_1 ngắt, tương ứng với trạng thái 1 của phần tử nhớ T_3 và T_4 là những bóng điều khiển sự nối thông phần tử nhớ với dây bit. T_5 và T_6 là mạch điện nạp trước của dây bit, dùng chung cho tất cả các phần tử nhớ cùng cột trong ma trận nhớ. Khi bắt đầu truy nhập truy xuất bộ nhớ, trên cực cửa của T_5 và T_6 có xung



Hình 6-5-8. Phần tử nhớ động MOS 4 bóng

nạp trước nên T_5 , T_6 nối thông, do đó các dây bit B , $\bar{B}$ có mức điện áp cao vì thông đến nguồn E_D . Sau khi kết thúc xung nạp trước, T_5 và T_6 ngắt, dây bit cách li khỏi nguồn E_D . Nhưng do tác dụng của điện dung phân bố C_B và $C_{\bar{B}}$. Khi đó, mức điện áp cao của dây bit có thể duy trì thêm một khoảng thời gian nữa. Trong khoảng thời gian này, giả sử tiến hành đọc dữ liệu, dây X có mức cao, T_3 và T_4 thông. Giả sử phần tử nhớ có trạng thái 0 (T_1 thông, T_2 ngắt) thì G_1 có mức cao, G_2 có mức thấp. Lúc này C_B phóng điện qua T_1 và T_3 . Do đó dây bit B biến thành mức thấp. Do T_2 ngắt, dây bit $\bar{B}$ vẫn ở mức cao. Vậy dữ liệu nhớ trong phần tử đã được đọc ra dây bit B và $\bar{B}$. Nếu lúc này dây Y cũng ở mức cao thì tín hiệu sẽ đưa đến đầu ra của RAM qua dây dữ liệu D và $\bar{D}$.

Vậy mạch điện nạp trước của dây bit có tác dụng gì? Trong khoảng thời gian T_3 , T_4 thông, nếu dây bit không được nạp điện trước thì mức cao có được của dây $\bar{B}$ chỉ do C_1 phóng qua T_4 nạp vào $C_{\bar{B}}$. Nếu thế, điện tích trên C_1 bị suy giảm. $C_{\bar{B}}$ có giá trị thậm chí lớn hơn C_1 (vì có nhiều phần tử nối vào dây bit). Vậy có thể G_1 không giữ nguyên mức cao sau 1 lần đọc, tức là dữ liệu bị mất. Nhờ có mạch điện nạp trước, điện thế dây bit $\bar{B}$ còn cao hơn điện thế G_1 một ít. Vậy khi đọc dữ liệu, điện tích trên C_1 không những không hề bị suy giảm, mà còn được làm tươi nhờ sự nạp điện thêm cho C_1 qua T_4 .

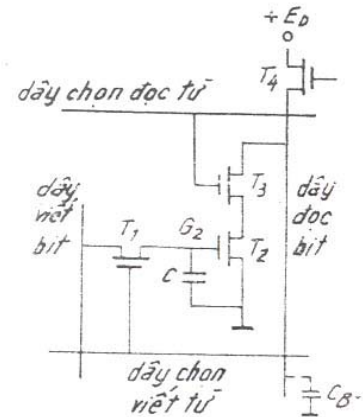
Khi tiến hành viết, đầu vào dữ liệu của RAM sẽ làm thay đổi trạng thái phần tử nhớ thông qua dây dữ liệu và dây bit, tức là đưa dữ liệu vào lưu giữ trong phần tử nhớ.

b) Phần tử nhớ động 3 bóng

Xem hình 6-5-9. Dữ liệu lưu giữ ở điện dung C của cực bóng T_2 . Các dây đọc/viết bit và dây chọn từ đều riêng biệt. Dây chọn đọc từ điều khiển bóng T_3 . Dây

chọn viết từ điều khiển bóng T_1 . T_4 là bóng bán dẫn mạch nạp trước dùng chung cho cột phần tử nhớ của ma trận.

Quá trình đọc diễn ra như sau. Đầu tiên dây đọc bit được nạp trước đến mức cao. Tiếp theo, dây chọn đọc từ lên mức cao làm cho T_3 thông. Giả sử C được nạp điện đến mức lớn hơn điện áp cắt của T_2 làm T_2 thông. C_B phóng điện qua T_3 , T_2 làm cho dây đọc bit ở mức thấp. Giả sử C không có điện tích, thì T_2 ngắt, C_B không có đường phóng điện, mức điện trên dây đọc bit qua bộ khuếch đại đọc ra đưa đến đầu ra bộ nhớ.



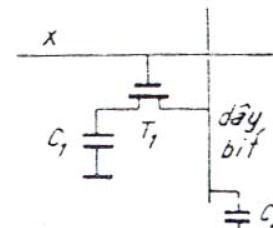
Hình 6-5-9. Phần tử nhớ động MOS

Khi tiến hành viết, dây chọn viết từ có mức cao, T_1 thông, dữ liệu viết vào được đưa đến dây viết bit điều khiển mức điện áp trên điện dung C qua T_1 . Vậy đã đưa dữ liệu vào lưu giữ trong phần tử nhớ.

Điện áp trên C khi đọc (tức mức ở G_2) ngược pha với mức điện dây đọc bit. Mức điện dây viết bit khi viết cùng cao cùng thấp như mức ở G_2 . Nếu tiến hành đọc một cách chu kì, đưa dữ liệu ở C ra dây đọc bit, qua bộ đảo pha đưa vào dây viết bit, rồi tiến hành viết lại vào phần tử nhớ. Đó là phương pháp làm tươi phần tử nhớ một cách chu kì.

c) Phần tử nhớ động 1 bóng

Xem hình 6-5-10. mạch này chỉ dùng một bóng bán dẫn MOS và một tụ điện. Dữ liệu lưu giữ ở điện dung C_1 . Bóng bán dẫn T điều khiển sự nối thông để đọc/viết; đưa dữ liệu từ dây bit vào phần tử nhớ hoặc đưa dữ liệu từ phần tử nhớ ra dây bit.



Hình 6-5-10. Phần tử nhớ động MOS

Khi viết, dây bit điều khiển mức điện trên C_1 qua T. Khi đọc, C_1 điều khiển mức điện trên C_B , tức đưa ra dữ liệu. Ví dụ, trên C_1 vốn mức V_1 , dây bit vốn $V_B = 0$ thì sau

$$\text{khi đọc, mức điện trên dây bit là } V_B = V_1 \times \frac{C_1}{C_1 + C_B}$$

Nhược điểm của mạch này là điện tích trên phần tử nhớ C_1 bị suy giảm mỗi lần đọc. Vậy nên cần làm tươi sau mỗi lần đọc. Ngoài ra, do nhiều phần tử nối vào dây bit nên giá trị điện dung C_B tương đối lớn, $C_B \gg C_1$. Dữ liệu đọc ra có mức điện bé; giá trị

mức logic 0 và mức logic 1 trên dây bit không khác xa nhau. Để xác định mức điện chênh lệch nhỏ trên dây bit cần phải dùng bộ khuếch đại đọc ra có độ nhạy cao.

Ba loại phần tử nhớ động trình bày trên dây đều có ưu nhược điểm. Mạch 4 bóng chiếm diện tích lớn trên chip nhưng không cần mạch làm tươi riêng, quá trình đọc ra đồng thời là quá trình làm tươi; vậy mạch phụ đơn giản. Mạch 3 bóng giảm diện tích một ít nhưng yêu cầu các dây chọn đọc/viết và dây dữ liệu đều riêng biệt, cần mạch ngoài điều khiển phản hồi để làm tươi; vậy dây nối với mạch ngoài tương đối nhiều hơn. Mạch 1 bóng đơn giản nhất. Nhưng yêu cầu Bộ khuếch đại đọc ra độ nhạy cao và yêu cầu làm tươi sau mỗi lần đọc ra. Vậy mạch điện phụ ngoài phức tạp. RAM tĩnh so với RAM động : Số lượng bán dẫn nhiều hơn, độ tích hợp thấp hơn, nhưng không nhất thiết định kì làm tươi, mạch phụ đơn giản, sử dụng tiện.

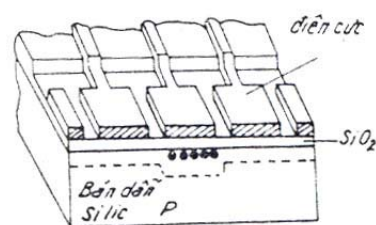
6.5.2. Dụng cụ ghép điện tích CCD (Charge Coupled Device)

Dụng cụ ghép điện tích CCD thuộc công nghệ MOS phát triển từ năm 1970. Ưu điểm của CCD là cấu trúc đơn giản, giá thành hạ, độ tích hợp cao, có thể chế tạo trên dây chuyền công nghệ MOS thông thường. CCD được quan tâm nhiều, hiện nay đã ứng dụng trong các bộ nhớ, xử lý tín hiệu và nhiếp ảnh. Nhược điểm của CCD là tốc độ chậm hơn bộ nhớ bán dẫn.

Sự khác biệt trong nguyên lí làm việc của CCD là ở chỗ nó dùng điện tích để biểu thị dữ liệu, chứ không phải là dòng hay áp. Về cơ bản, CCD là một bộ ghi dịch động, nên hiện nay dùng làm bộ nhớ nối tiếp.

1) Cấu trúc (xem hình 6-5-11)

Trên đế bán dẫn Silic loại P (hoặc loại N), người ta tạo ra một lớp mỏng cách điện SiO_2 cỡ 1200 Å. Trên lớp cách điện này người ta cho lắng đọng một loạt các điện cực 10 x 10 μm với giãn cách <3 μm . Mỗi điện cực và lớp đế bán dẫn với lớp cách điện ở giữa chúng tạo thành một tụ điện. Vậy CCD là một dãy các tụ điện MOS cấu trúc nên. Những tụ điện này là phần tử nhớ mà có hay không có điện tích biểu thị các mức logic 1 hay 0. Điện tích nạp vào có thể dịch chuyển từ tụ điện nọ sang tụ điện kia tương tự như sự chuyển dịch mức trong bộ ghi dịch ta đã biết.

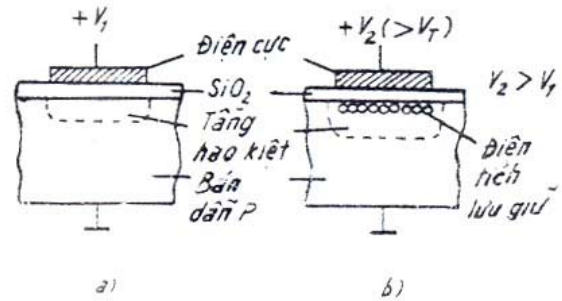


Hình 6-5-11. Sơ đồ mặt cắt của CCD.

2) Lưu giữ điện tích

Giả sử đưa điện áp dương $+V_1$ vào điện cực. Những phân tử lỗ trống (hạt dẫn đa số) trong lớp Silic bên dưới điện cực bị đẩy đi khỏi tạo thành một tầng hao kiệt (xem hình 6-5-12).

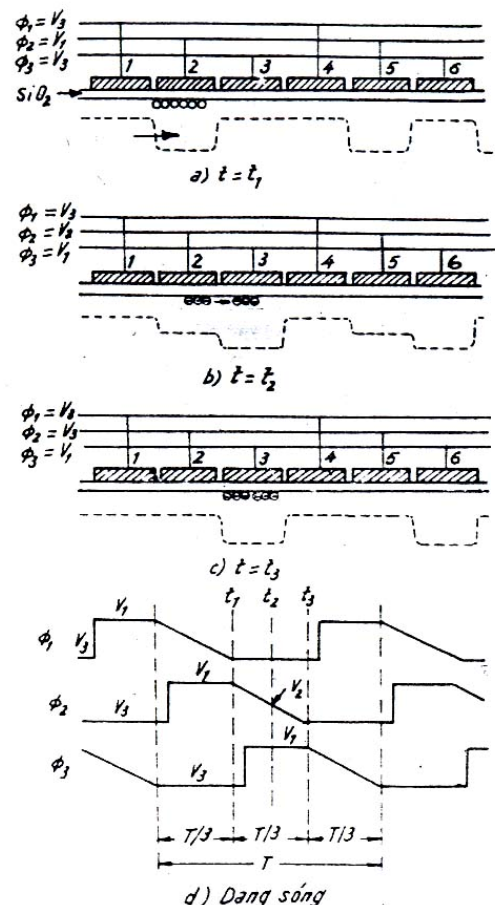
Độ dày của tầng hao kiệt tăng theo độ lớn của điện áp $+V_1$. Khi điện áp điện cực lớn hơn điện áp cắt của cấu trúc MOS, $+V_2 > V_T$, một rãnh loại N hình thành trên mặt đế Silic bên dưới điện cực đó.



Hình 6-5-12. CCD lưu giữ điện tích.

Trong bóng bán dẫn MOS, hai phía là cực nguồn và cực máng, giữa chúng các hạt dẫn đa số có thể khuếch tán về phía đế Silic, nên quá trình tạo ra rãnh N là khá dài hơn, cỡ nhiều giây đồng hồ. Trước khi rãnh N được tạo ra, nếu có điện tử từ bên ngoài nhập vào tầng hao kiệt thì nó sẽ khuếch tán về phía đế và tích lũy ở đó tạo thành điện tích lưu giữ (Xem hình 6-5-12b). Việc này xảy ra nhờ tác dụng điện trường. Nếu không có điện tử bên ngoài nhập vào thì không có điện tích lưu giữ tích tụ ở lớp đế dưới điện cực. Rõ ràng phương pháp lưu giữ điện tích chỉ có thể lưu giữ trong thời gian rất ngắn. Vì nếu thời gian đủ dài thì điện tử nhiệt tràn ngập lớp đế, bất kể có điện tích ngoài nhập vào hay không. Nói cách khác, tiên đề làm việc của CCD phải là lượng điện tử bên ngoài nhập vào lớn hơn nhiều số điện tử nhiệt. Khu vực bên dưới điện cực có thể lưu giữ điện tích được gọi là vùng bẫy điện thế.

3) Chuyển dịch điện tích bằng cách thay đổi thích hợp điện áp đưa vào điện cực, điện tích tích tụ hiệu lưu giữ trong vùng bẫy điện thế có thể chuyển dịch từ miền gần điện cực này đến miền gần điện cực khác, hình 6-5-13. Với 6 điện cực, điện cực 1,4 được cấp xung đồng hồ pha thứ nhất ϕ_1 , điện cực 3, 6 được cấp xung đồng hồ pha thứ ba ϕ_3 . Dạng sóng ba pha như



Hình 6-5-13. Sơ đồ 3 pha chuyển dịch điện tích của CCD.

hình 6-5-13d. Các pha lệch nhau $\frac{T}{3}$ với T là chu kỳ xung đồng hồ.

Hình 6-5-13a tương ứng thời điểm t_1 mà $\phi_2 = V_1$, $\phi_1 = \phi_3 = V_3$, lúc này tầng hao kiệt hình thành bên dưới tất cả các điện cực. Do giãn cách giữa các điện cực rất nhỏ nên tầng hao kiệt của các điện cực liên thông nhau. Trong tầng, điện thế càng gần đế càng thấp. Lúc này điện thế ϕ_2 cao nhất, nên tương ứng điện thế bề mặt đế Silic dưới điện cực 2,5 cũng cao nhất. Nếu điện tử nhập vào vùng bẫy điện thế 2 thì điện tử sẽ tích tụ ở vùng này và không chuyển dịch.

Hình 6-5-13b tương ứng thời điểm t_2 mà ϕ_3 lên mức V_1 , ϕ_2 giảm tới mức V_2 , ϕ_1 vẫn duy trì mức thấp nhất V_3 . Lúc này tầng hao kiệt của điện cực 3, 6 sâu nhất, điện thế bề mặt đế Silic dưới điện cực 3,6 cao nhất. Vậy những điện tử lúc t_1 ở bẫy điện thế 2, bây giờ dịch chuyển sang bẫy điện thế 3 (bên cạnh có điện thế cao hơn). Ở T_1 bẫy điện thế 5 giả thiết không có điện tử, nên bây giờ (t_2) cũng không có điện tử dịch chuyển sang bẫy điện thế 6.

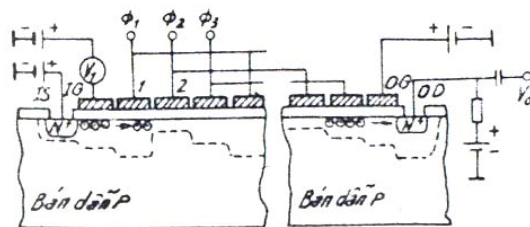
Hình 6-5-13c tương ứng thời điểm t_3 , khi mà việc chuyển dịch điện tích đã hoàn thành, điện tích đã dịch sang phải một điện cực và lưu giữ trong bẫy điện thế 3. Trong quá trình chuyển dịch đó, ϕ_1 liên tục duy trì mức điện thế thấp nhất để nhằm phòng ngừa điện tích dịch sang trái.

Từ t_1 đến t_3 là thời gian $\frac{1}{3}$ chu kỳ T, điện tích chuyển dịch một khoảng giữa hai điện cực cạnh nhau. Vậy sau một chu kỳ T thì điện tích chuyển dịch ba khoảng cách điện cực. Ba điện cực lân cận nhau cấu trúc thành 1 bit của bộ ghi dịch.

Nếu dùng hình dạng điện cực khác nhau và số pha đồng hồ khác nhau thì có thể tạo ra nhiều cách điều khiển điện tích chuyển dịch khác nhau.

4) đầu vào và đầu ra

mạch CCD thực tiễn phải có đầu vào để đưa dữ liệu vào và phải có đầu ra để lấy dữ liệu ra. Ở đầu vào CCD thì dòng (hoặc áp) phải biến đổi thành điện tích. Ở đầu ra của CCD thì điện tích phải biến đổi thành dòng (hoặc áp).



Hình 6-5-14. Mạch vào và mạch ra của CCD.

Hình 6-5-14 là ví dụ về mạch điện đầu vào, đầu ra của CCD.

Hai bên điện cực cơ bản có thêm các điện cực sau: cực nguồn vào IS, cực máng ra OD, cực cổng vào OG, cực cổng ra OG mà các cực cổng thực chất là điện cực chuyển dịch dây dẫn riêng. Khi công tác, trên IS và IG đều có thiên áp dương để hình thành tầng hao kiệt bên dưới nó. Tầng hao kiệt này liên thông với tầng hao kiệt bên dưới điện cực chuyển dịch như hình vẽ. Khi mà điện cực 1 có điện thế cao nhất, tầng hao kiệt của nó sâu hơn tầng hao kiệt của IS, IG. IS (N^+) là một nguồn điện tử nhưng điện thế IS và IG quyết định điện tử có nhập vào vùng dưới điện cực 1 hay không. Nếu $V_{IG} > V_{IS}$ thì ngừng nhập. Vậy IS, IG và điện cực 1 tương tự một bóng bán dẫn MOS. Điện áp tín hiệu (dữ liệu vào) đưa nối tiếp vào IG hay vào IS điều khiển quan hệ điện thế tương đối của IS và IG, nghĩa là điều khiển nhập hay ngừng nhập. Sau khi kết thúc quá trình điều khiển nhập dữ liệu vào này, IG trở về mức điện thế thấp nhất làm cách li nguồn điện tử khỏi vùng bẫy điện thế bên dưới điện cực 1. Bây giờ có thể tiến hành điều khiển chuyển dịch điện tích.

Trong mạch ra, trên OG và OD đều có thiên áp dương, và có 1 điện trở nối tiếp vào mạch vòng của OD. Sau khi điện tích đã chuyển dịch đến điện cực chuyển dịch cuối cùng, nếu điện thế trên nó thấp hơn điện thế OG thì điện tích đi qua lớp đế Silic dưới OG để nhập vào khu vực máng Ra N^+ , sinh ra dòng điện trong mạch OD, tạo ra điện áp tín hiệu ra trên điện trở. Tác dụng của OG là giảm nhỏ ghép điện dung giữa điện cực chuyển dịch cuối cùng và OD. Nhờ vậy giảm nhỏ ảnh hưởng của xung đồng hồ đối với tín hiệu ra.

Phương pháp nhập điện tích trên đây đây gọi là phương pháp điện. Nếu dùng phần tử bán dẫn nhạy quang thì có thể nhập điện tích nhờ điện tử kích quang. Dựa vào nguyên lý nhập phương pháp quang có thể chế tạo máy ảnh chất rắn – ghép điện tích.

6.6. THIẾT KẾ MẠCH DÂY

Thiết kế là tổng hợp mạch điện, ngược với quá trình phân tích. Xuất phát từ yêu cầu đề ra, mục đích thiết kế là mạch logic thoả mãn yêu cầu đó. Từ việc phân tích mạch dây, ta biết rằng có thể dễ dàng tìm phương trình định thời, phương trình kích, phương trình ra và vẽ sơ đồ logic. Trong tiết này ta sẽ tập trung tìm hiểu công việc thiết kế bộ đếm, đồng thời xem xét nhưng nét chính công việc thiết kế mạch dây nói chung. Bộ đếm là một mạch dây tương đối đơn giản nhưng rất điển hình. Phương pháp thiết kế bộ đếm cũng rất điển hình. Vả lại mục đích tìm hiểu công việc thiết kế không phải để giải quyết vấn đề thiết kế bất kì mạch dây phức tạp nào. Trên cơ sở nắm vững việc thiết kế bộ đếm, ta sẽ càng hiểu sâu toàn bộ nội dung chương này và sẽ có năng lực phân tích giải quyết những vấn đề mạch dây.

Với tình hình sản xuất đại trà các IC số trung, đại quy mô hiện nay, với quan điểm người dùng, nhiệm vụ thiết kế có đổi khác.

6.6.1. Thiết kế bộ đếm đồng bộ

Dưới đây giới thiệu 2 phương pháp thiết kế bộ đếm đồng bộ (có thể dùng những phương pháp khác).

1) Phương pháp thứ nhất

a) Các bước cơ bản

- Phân tích yêu cầu thiết kế, xây dựng đồ hình trạng thái ban đầu.
- Xác định số lượng và chủng loại Flip Flop, chọn lựa sự mã hoá trạng thái.
- Tìm phương trình trạng thái và phương trình ra, kiểm tra khả năng tự khởi động.

động.

- Tìm phương trình kích

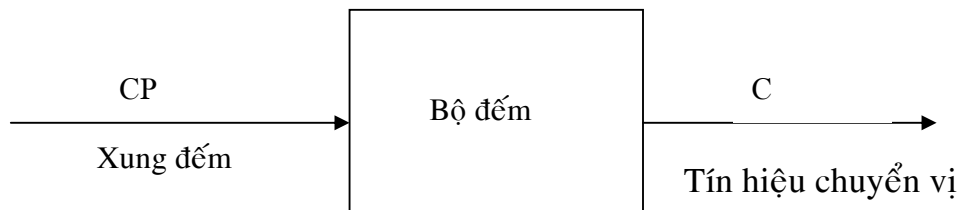
- Vẽ sơ đồ logic

b) Ví dụ thiết kế

Ví dụ 6-6-1: hãy thiết kế bộ đếm thuận thập phân đồng bộ.

Bài giải

Phân tích yêu cầu thiết kế, xây dựng đồ hình trạng thái ban đầu

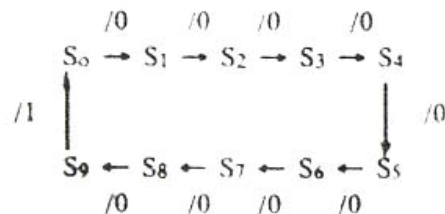


Hình 6-6-1. Mô hình yêu cầu của bộ đếm

Hình 6-6-1. Mô hình yêu cầu của bộ đếm.

Bộ đếm cần có mười trạng thái, $N = 10$, biểu thị bằng $S_0, S_1, \dots, S_9$. Căn cứ quy luật đếm thuận thập phân, ta vẽ ra đồ hình trạng thái ban đầu như hình 6-6-2 dưới đây.

Đồ hình trạng thái phản ánh toàn diện yêu cầu thiết kế. Trong hình 6-6-2, S_0 thay mặt số 0, S_1 thay mặt số 1, ..., S_9 thay mặt số 9. Dưới tác động của xung đếm đưa vào, trạng thái mạch điện phải chuyển đổi trạng thái tuần tự theo luật đếm thuận. Tương ứng trạng thái S_9 thì $C=1$, tương ứng các trạng thái khác thì $C = 0$. Khi bộ đếm từ S_9 chuyển đổi sang S_0 thì bộ đếm



Hình 6-6-2. Đồ hình trạng thái ban đầu của bộ đếm thập phân đồng bộ

xóa về 0, tín hiệu chuyển vị (nhớ) kích lật bộ đếm trọng số lớn hơn.

Xác định số lượng và chủng loại Flip Flop, chọn lựa mã hoá trạng thái.

Vì $2^n \geq N = 10$

Vậy $n = 4$, chọn Flip Flop JK

4 FF có tất cả 16 trạng thái, để biểu thị $S_0 + S_9$ chỉ cần chọn 10 trạng thái, nên có nhiều phương án mã hoá. Chúng ta hãy chọn phương án mã hoá thông dụng nhất: mã 8421 ($Q_4Q_3Q_2Q_1$)

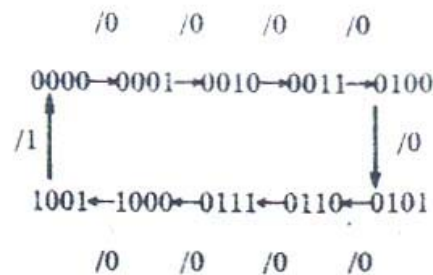
$S_0 = 0000, S_1 = 0001, S_2 = 0010, S_3 = 0011, S_4 = 0100,$

$S_5 = 0101, S_6 = 0110, S_7 = 0111, S_8 = 1000, S_9 = 1001.$

Tìm phương trình trạng thái, phương trình ra, kiểm tra tự khởi động

Tìm phương trình trạng thái

Trạng thái kế tiếp và đầu ra của bộ đếm đều là hàm số của trạng thái xét, mà trạng thái xét và trạng thái kế tiếp của bộ đếm biểu thị bằng trạng thái hiện tại và kế tiếp của các Flip Flop cấu trúc nên bộ đếm. Do vậy, căn cứ đồ hình trạng thái hình 6-6-3 ta vẽ ra bảng Karnaugh của đầu ra và trạng thái kế tiếp bộ đếm. Rồi từ đó tìm được phương trình trạng thái và phương trình ra của mạch điện.



Hình 6-6-3. Đồ hình trạng thái (sau khi chọn cách mã hoá) của bộ đếm

6 trạng thái 1010 ÷ 1111 không được dùng, sẽ không xuất hiện trong khi bộ đếm làm việc bình thường, có thể giúp việc tối thiểu hoá.

Phương trình đặc trưng của Flip Flop JK là

$$Q^{n+1} = JQ^n + \overline{K}Q^n \quad (6-6-1)$$

Kết quả tối thiểu hoá phải được viết dưới dạng tương tự (6-6-1). Ví dụ biểu thức hàm số trạng thái kế tiếp đó ta tìm phương trình kích.

Trong hình 6-6-4, trạng thái kế tiếp của bộ đếm được điền vào các ô, đó cũng là

$Q_4^1 Q_3^1$ \ $Q_2^1 Q_1^1$	00	01	11	10
00	0001	0010	0100	0011
01	0101	0110	1000	0111
11	x	x	x	x
10	1001	0000	x	x

Hình 6-6-4. Bảng Karnaugh trạng thái kế tiếp của bộ đếm

trạng thái kế tiếp đó ta tìm phương trình kích.

Trong hình 6-6-4, trạng thái kế tiếp của bộ đếm được điền vào các ô, đó cũng là trạng thái kế tiếp của các FF cấu trúc bộ đếm.

Tương ứng sắp xếp trong mỗi ô là $Q_4^{n+1} Q_3^{n+1} Q_2^{n+1} Q_1^{n+1}$. Từ đó ta có thể tách riêng thành bảng Karnaugh trạng thái kế tiếp của mỗi FF. Rồi dùng phương pháp đồ hình để tìm ra phương trình trạng thái. Nhưng chỉ cần chú ý đặc điểm sắp xếp không thay đổi của $Q_4^{n+1} Q_3^{n+1} Q_2^{n+1} Q_1^{n+1}$, ta có thể tối thiểu hoá luôn.

$$\begin{cases} Q_4^{n+1} = Q_3^n Q_2^n Q_1^n \bar{Q}_4^n + \bar{Q}_1^n Q_4^n \\ Q_n^{n+1} = Q_2^n Q_1^n \bar{Q}_3^n + (\bar{Q}_2^n + \bar{Q}_1^n) Q_3^n = Q_2^n Q_1^n \bar{Q}_3^n + \overline{Q_2^n Q_3^n} Q_3^n \\ Q_2^{n+1} = \bar{Q}_4^n Q_1^n \bar{Q}_2^n + \bar{Q}_1^n Q_2^n \\ Q_1^{n+1} = \bar{Q}_1^n \end{cases} \quad (6-6-2)$$

Tìm phương trình ra

Căn cứ đồ hình trạng thái hình 6-6-3 ta được bảng Karnaugh của hàm số đầu ra C như hình 6-6-5.

Kết quả tối thiểu hoá là:

$$C = Q_4^n Q_1^n \quad (6-6-3)$$

Kiểm tra khả năng tự khởi động của bộ đếm

6 trạng thái không được dùng 1011 ÷ 1111 có

$Q_4^n Q_3^n$	$Q_2^n Q_1^n$			
	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	×	×	×	×
10	0	1	×	×

Hình 6-6-5. Bảng Karnaugh của C.

khả năng tạo ra vòng tuần hoàn không được dùng,

làm cho bộ đếm không tự khởi động. Sau khi đã tìm

phương trình trạng thái và phương trình ra, bây giờ ta cần phân tích tình huống chuyển đổi của các trạng thái không được dùng. Nếu dưới tác dụng của xung đếm đầu vào mà bộ đếm không thể trở về trạng thái được dùng, tức là không thể tự khởi động, thì ta phải tìm mọi cách giải quyết. Chẳng hạn, ta chọn lại mã hoá trạng thái, hoặc sửa đổi trạng thái kế tiếp của các trạng thái không được dùng (như cách tự khởi động của bộ đếm kiểu ghi dịch), hoặc sử dụng đầu vào di bộ cưỡng chế bộ đếm về trạng thái được dùng v.v...

Đưa các trạng thái không được dùng vào (6-6-3), (6-6-3) tiến hành tính toán, ta được bảng 6-6-1 sau:

Bảng 6-6-1: TÌNH HUỐNG CHUYỂN ĐỔI TRẠNG THÁI KHÔNG ĐƯỢC DỪNG.

Q_4^n	Q_3^n	Q_2^n	Q_1^n	Q_1^{n+1}	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	C
1	0	1	0	1	0	1	1	0
1	0	1	1	0	1	0	0	1
1	1	0	0	1	1	0	1	0
1	1	0	1	0	1	0	0	1
1	1	1	0	1	1	1	1	0
1	1	1	1	0	0	0	0	1

Từ bảng ta thấy các tình huống có thể xảy ra

$\begin{matrix} /0 & /1 \\ 1010 \rightarrow 1011 \rightarrow 0100 \end{matrix}$
 $\begin{matrix} /0 & /1 \\ 1100 \rightarrow 1101 \rightarrow 0100 \end{matrix}$
 $\begin{matrix} /0 & /1 \\ 1110 \rightarrow 1111 \rightarrow 0000 \end{matrix}$

Đều khởi động được.

Đồng nhất các hệ số của (6-6-2) và (6-6-1) ta có phương trình kích là:

$$\left\{ \begin{array}{l} J_1 = K_1 = 1 \\ J_2 = \overline{Q_4^n} Q_1^n \\ J_3 = K_3 = Q_2^n Q_1^n \\ J_4 = Q_3^n Q_2^n Q_1^n \end{array} \right. \quad \left\{ \begin{array}{l} K_2 = Q_1^n \\ K_4 = Q_1^n \end{array} \right. \quad (6-6-4)$$

Vẽ sơ đồ logic:

Căn cứ vào đặc điểm bộ đếm đồng bộ – xung đếm đầu vào là xung đồng hồ của các FF, dựa vào phương trình kích (6-6-4) và phương trình ra (6-6-3), ta có thể vẽ được sơ đồ logic như hình 6-2-10.

Nếu việc chọn lựa mã hoá trạng thái theo các mã khác thì ta có thể đi đến bộ đếm thuận thập phân đồng bộ khác tương ứng (mã dư 3, mã Gray dư 3...)

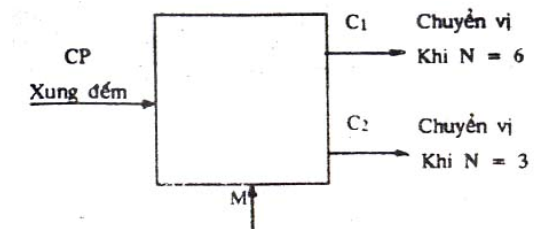
Ví dụ 6-6-2: hãy thiết kế bộ đếm thuận đồng bộ điều khiển được, với $M=0$ thì $N=6$, với $M=1$ thì $N=3$.

Bài giải:

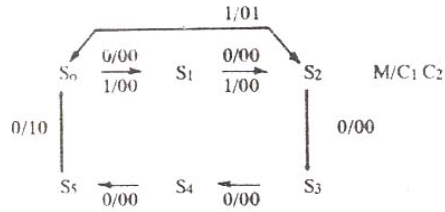
Phân tích yêu cầu, xây dựng đồ hình trạng thái ban đầu

Khi $M=0$ thì $N=6$

Khi $M=1$ thì $N=3$



Hình 6-6-6. Mô hình yêu cầu của bộ đếm điều khiển



Hình 6-6-7. Đồ hình trạng thái ban đầu

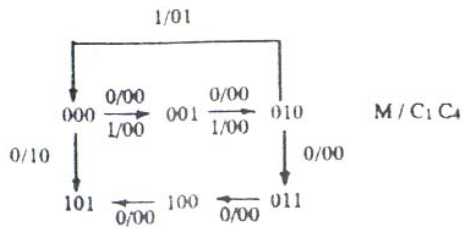
Xác định số lượng và loại FF, chọn lựa mã trạng thái

$$2^n \geq N = 6$$

Vậy $n = 3$, chọn Flip Flop JK

Với mã hoá $Q_3 Q_2 Q_1$, chọn

$$S_0 = 000, S_1 = 001, S_2 = 011, S_4 = 100, S_5 = 101$$



Hình 6-6-8.

Đồ hình trạng thái (sau khi mã hoá).

Tìm phương trình trạng thái, phương trình ra, kiểm tra tự khởi động

$Q_2^n Q_1^n$		$Q_3^n Q_2^n$			
		00	01	11	10
$M Q_3^n$	00	001	010	100	011
	01	101	000	×	×
	11	×	×	×	×
	10	001	010	×	000

Hình 6-6-9. Bảng Karnaugh trạng thái kế tiếp của bộ đếm.

Từ hình 6-6-9, tối thiểu hoá xong, ta có:

$$\begin{cases} Q_3^{n+1} = Q_2^n Q_1^n \bar{Q}_3^n + \bar{Q}_1^n Q_3^n \\ Q_2^{n+1} = \bar{Q}_3^n Q_1^n \bar{Q}_2^n + \bar{M} Q_1^n Q_2^n \\ Q_1^{n+1} = \bar{Q}_2^n \bar{Q}_1^n + \bar{M} \bar{Q}_1^n = (\bar{Q}_2^n + \bar{M}) \bar{Q}_1^n = \bar{Q}_2^n M \bar{Q}_1^n \end{cases} \quad (6-6-5)$$

Phương trình ra:

MQ ₃ \ Q ₂ ⁿ Q ₁ ⁿ				
	00	01	11	10
00	0	0	0	0
01	0	1	×	×
11	×	×	×	0
10	0	0	×	0

a)

MQ ₃ \ Q ₂ ⁿ Q ₁ ⁿ				
	00	01	11	10
00	0	0	0	0
01	0	0	×	×
11	×	×	×	×
10	0	0	×	1

b)

Hình 6-6-10. bảng Karnaugh đầu ra.

a) của chuyển vị C₁

b) của chuyển vị C₂.

Sau khi tối thiểu hoá bảng Karnaugh hình 6-6-10, ta được:

$$C_1 = Q_3^n Q_1^n \quad (6-6-6)$$

$$C_2 = MQ_2^n \quad (6-6-7)$$

Về tình huống chuyển đổi các trạng thái không được dùng, xem bảng 6-6-2 và bảng 6-6-3

Bảng 6-6-2: TÌNH HUỐNG TRẠNG THÁI KHÔNG ĐƯỢC DÙNG KHI M=0

Q ₃ ⁿ	Q ₂ ⁿ	Q ₁ ⁿ	Q ₃ ⁿ⁺¹	Q ₂ ⁿ⁺¹	Q ₁ ⁿ⁺¹	C ₁	C ₂
1	1	0	1	1	1	0	0
1	1	1	0	0	0	1	0

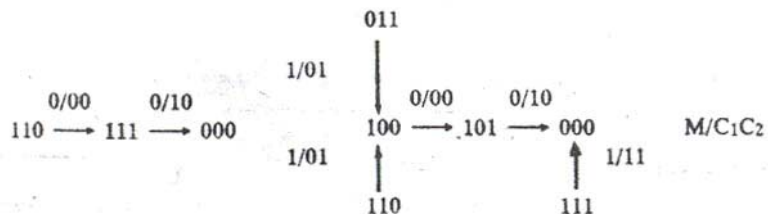
Bảng 6-6-3: TÌNH HUỐNG TRẠNG THÁI KHÔNG ĐƯỢC DÙNG KHI M = 1

Q ₃ ⁿ	Q ₂ ⁿ	Q ₁ ⁿ	Q ₃ ⁿ⁺¹	Q ₂ ⁿ⁺¹	Q ₁ ⁿ⁺¹	C ₁	C ₂
0	1	1	1	0	0	0	1
1	0	0	1	0	1	0	0
1	0	1	0	0	0	1	0
1	1	0	1	0	0	0	1
1	1	1	0	0	0	1	1

Hình 6-6-11. Tình huống chuyển đổi trạng thái không được dùng

a) Khi M = 0;

b) b) Khi M = 1.



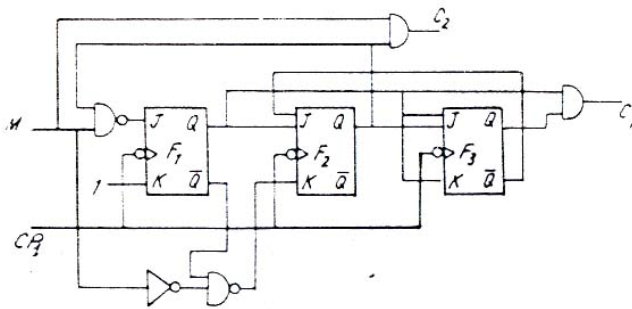
Hình 6-6-11 chứng tỏ rằng mạch có khả năng tự khởi động

Tìm phương trình kích

Đồng nhất các hệ số của (6-6-5) và (6-6-1) ta có:

$$\begin{cases} J_2 = Q_2^n Q_1^n & K_3 = Q_1^n \\ J_2 = \overline{Q_3}^n Q_1^n & K_2 = \overline{M} \overline{Q_1}^n \\ J_1 = Q_2^n M & K_1 = 1 \end{cases} \quad (6-6-8)$$

Vẽ sơ đồ logic



Hình 6-6-12. Bộ đếm điều khiển được.

Đặc điểm của phương pháp thiết kế trình bày trên đây là sau khi chọn lựa mã hoá trạng thái thì dùng bảng Karnaugh tìm phương trình trạng thái, rồi kết hợp với phương trình đặc trưng của FF mà tìm ra phương trình kích, để cuối cùng hoàn thành thiết kế bộ đếm đồng bộ mong muốn.

2) Phương pháp thứ hai

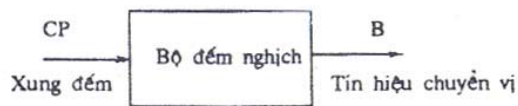
a) Các bước cơ bản

- Phân tích yêu cầu thiết kế, xây dựng đồ hình trạng thái ban đầu.
- Xác định số lượng và chủng loại FF, chọn lựa sự mã hoá trạng thái
- Kê ra bảng sử dụng
- Tìm phương trình kích và phương trình ra
- Vẽ sơ đồ logic
- Kiểm tra khả năng tự khởi động

b) Ví dụ thiết kế

Ví dụ 6-6-3: hãy thiết kế bộ đếm nghịch thập phân đồng bộ

Bài giải:



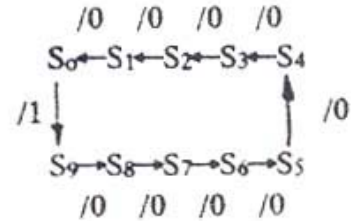
Hình 6-6-13. Mô hình yêu cầu của bộ đếm nghịch

Phân tích yêu cầu thiết kế, xây dựng đồ hình trạng thái ban đầu

Bộ đếm có $N = 10$ tương ứng các trạng thái bộ đếm là $S_0, S_1, \dots, S_9$.

Căn cứ quy luật đếm nghịch, ta vẽ được đồ hình trạng thái ban đầu như hình 6-6-14

Hình 6-6-14. Đồ hình trạng thái ban đầu của bộ đếm nghịch



Xác định số lượng và chủng loại FF, chọn lựa mã hoá trạng thái

Vì $2^n \geq N = 10$

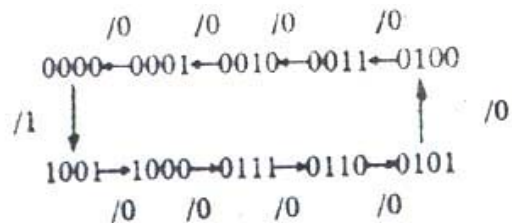
Vậy $n = 4$. Chọn Flip Flop JK.

Dùng mã 8421

$S_0 = 0000, S_1 = 0001, S_2 = 0010, S_3 = 0011, S_4 = 0100$

$S_5 = 0101, S_6 = 0110, S_7 = 0111, S_8 = 1000, S_9 = 1001$

Hình 6-6-15. Đồ hình trạng thái bộ đếm nghịch (sau khi mã hoá)



Kê ra bảng sử dụng

Căn cứ đồ hình trạng thái kê ra bảng trạng thái bộ đếm, từ yêu cầu chuyển đổi trạng thái của bảng sử dụng như dưới đây (bảng 6-6-4)

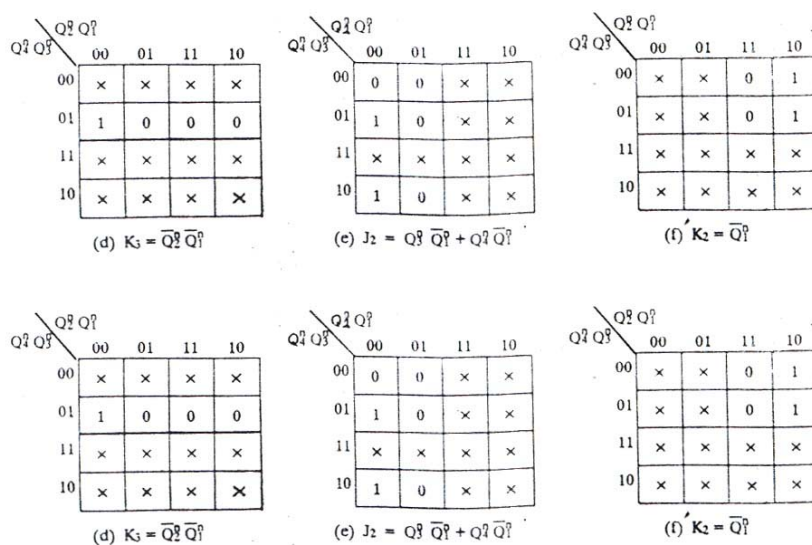
Bảng 6-6-4: BẢNG SỬ DỤNG CỦA BỘ ĐẾM NGHỊCH

Q_4^n	Q_3^n	Q_2^n	Q_1^n	Q_4^{n+1}	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	B	$J_4 K_4$	$J_3 K_3$	$J_2 K_2$	$J_1 K_1$
0	0	0	1	1	0	0	1	1	1 x	0 x	0 x	1 x
0	0	0	0	0	0	0	0	0	0 x	0 x	0 x	x 1
0	0	1	1	0	0	0	1	0	0 x	0 x	x 1	1 x
0	0	1	0	0	0	1	0	0	0 x	0 x	x 0	x 1
0	1	0	1	0	0	1	1	0	0 x	x 1	1 x	1 x
0	1	0	0	0	1	0	0	0	0 x	x 0	0 x	x 1
0	1	1	1	0	1	0	1	0	0 x	x 0	x 1	1 x
0	1	1	0	0	1	1	0	0	0 x	x 0	x 0	x 1
1	0	0	1	0	1	1	1	0	x 1	1 x	1 x	1 x
1	0	0	0	1	0	0	0	0	x 0	0 x	0 x	x 1

Khi xác định yêu cầu kích (từ yêu cầu chuyển đổi trạng thái) cần sử dụng bảng đầu vào kích của FF; nếu nắm vững kĩ năng thì có thể sử dụng phương trình đặc trưng.

Tìm phương trình kích và phương trình ra

Dựa vào quan hệ logic đã biết trong bảng sử dụng vẽ bảng Karnaugh. Từ đó tìm phương trình kích và phương trình ra. Xem hình 6-6-16 và 6-6-17.



Hình 6-6-16. Bảng Karnaugh của đầu vào kích
(Từ bảng sử dụng 6-6-4, ta trực tiếp rút ra ngay $J_1 = K_1 = 1$)

Hình 6-6-17. Bảng Karnaugh của chuyển vị B.

$Q_4^2 \ Q_3^2$		$Q_2^2 \ Q_1^2$			
		00	01	11	10
00		1	0	0	0
01		0	0	0	0
11		x	x	x	x
10		0	0	x	x

$B = \overline{Q_4^2} \overline{Q_3^2} \overline{Q_2^2} \overline{Q_1^2}$

Vẽ sơ đồ logic

Kiểm tra khả năng tự khởi động

Sử dụng cách phân tích logic để vẽ đồ hình trạng thái, từ đó thấy được mạch có tự khởi động hay không.

Đặc điểm của phương pháp thiết kế thứ hai này là sau khi đã chọn lựa mã hoá trạng thái, căn cứ đồ hình trạng thái kê ra bảng sử dụng rồi dùng bảng Karnaugh tìm phương trình kích. So sánh với phương pháp thứ nhất ta thấy chỗ mạnh hơn là dùng bảng sử dụng nên không cần giải phương trình logic, dùng bảng Karnaugh để đi đến phương trình kích một cách dễ dàng; chỗ yếu hơn là có rắc rối, dễ nhầm khi xác định yêu cầu kích, nhất là bộ đếm nhiều số.

Hai phương pháp được giới thiệu trên là rất cơ bản, phổ dụng, có thể áp dụng nguyên tắc của nó vào thiết kế mạch dây nối chung.

6.6.2. Thiết kế bộ đếm dị bộ

Dưới đây chọn ra một phương pháp cơ bản trong nhiều phương pháp có thể thực hiện thiết kế bộ đếm dị bộ. Nói chung mạch điện bộ đếm dị bộ đơn giản hơn mạch điện bộ đếm đồng bộ.

1) Các bước cơ bản

- Phân tích yêu cầu thiết kế, xác định đồ hình trạng thái ban đầu
- Xác định số lượng và loại hình FF, chọn lựa mã hoá trạng thái
- Vẽ đồ thị dạng sóng, chọn xung đồng hồ
- Tìm phương trình trạng thái, phương trình ra, kiểm tra tự khởi động
- Tìm phương trình kích
- Vẽ sơ đồ logic

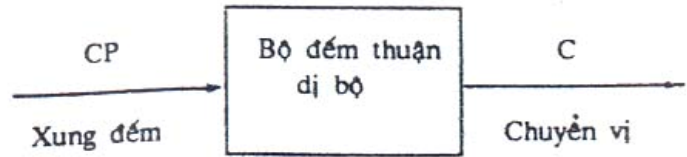
2) Ví dụ thiết kế

Ví dụ 6-6-4: hãy thiết kế bộ đếm thuận thập phân dị bộ

Bài giải:

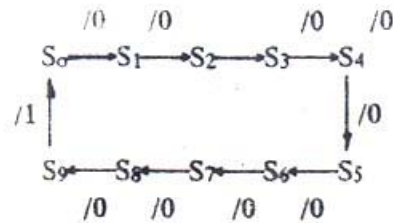
Phân tích yêu cầu thiết kế, xây dựng đồ thị hình trạng thái ban đầu

$N = 10$ sử dụng 10 trạng thái $S_0, S_1, \dots, S_9$



Hình 6-6-18. Bộ đếm thuận thập phân dị bộ.

a) Mô hình bộ đếm yêu cầu



b) Đồ hình trạng thái ban đầu

Chọn lựa mã hoá trạng thái

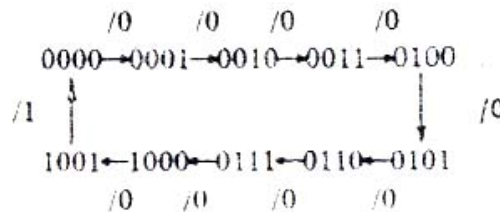
Vì $2^n \geq N = 10$ vậy $n = 4$, chọn Flip Flop D. Chọn mã 8421, $Q_4 Q_3 Q_2 Q_1$

$S_0 = 0000$, $S_1 = 0001$, $S_2 = 0010$, $S_3 = 0011$, $S_4 = 0100$

$S_5 = 0101$, $S_6 = 0110$, $S_7 = 0111$, $S_8 = 1000$, $S_9 = 1001$

Hình 6-6-19

Đồ hình trạng thái (sau mã hoá).



Chọn xung đồng hồ

Vẽ dạng sóng như hình 6-6-20

Khi vẽ dạng sóng cần lưu ý 2

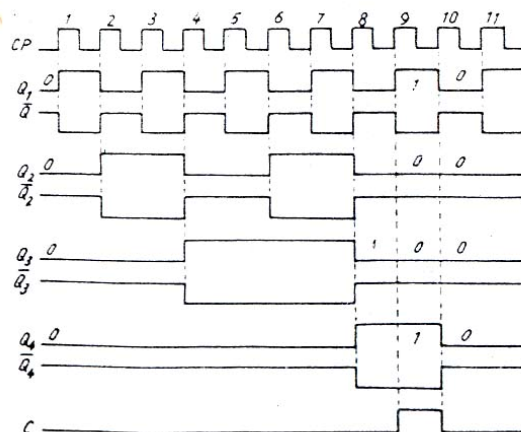
điểm:

- Quy luật chuyển đổi trạng thái của mỗi FF do đồ hình trạng thái quyết định. Thời điểm lật tương ứng sườn kích của xung đồng hồ.

- Vẽ số xung đồng hồ CP phải xấp xỉ lớn hơn N.

Mục đích việc lưu ý thứ hai

là để phản ánh toàn bộ tình huống làm



Hình 6-6-20. Dạng sóng bộ đếm thuận

việc bình thường của bộ đếm, đạt đến đầy đủ yêu cầu thiết kế:

Từ chức năng logic của FF ta biết rằng phương trình đặc trưng có điều kiện cần là xuất hiện xung đồng hồ. Trong đồ thị dạng sóng, chỗ nào yêu cầu FF lật thì phải cung cấp xung đồng hồ với sườn kích thích hợp. Vậy căn cứ vào hình 6-6-20 và hình 6-6-19, ta có thể chọn

$$\begin{cases} CP_1 = CP \\ CP_2 = \overline{Q_1} \\ CP_3 = \overline{Q_2} \\ CP_4 = \overline{Q_1} \end{cases} \quad (6-6-9)$$

CP_1, CP_2, CP_3, CP_4 là xung đồng hồ tương ứng của các Flip Flop F_1, F_2, F_3, F_4 .

Khi thoả mãn yêu cầu chuyển đổi trạng thái rồi thì số xung đồng hồ càng ít càng tốt. Ví dụ, đối với F_3 chẳng hạn, xung đồng hồ của nó có thể là CP , hoặc $\overline{Q_1}$, hoặc $\overline{Q_2}$ đều được. Nhưng so sánh chúng với nhau, thì nên chọn $\overline{Q_2}, \overline{Q_3}$. Vì khi Q_4 lật từ 1 về 0, yêu cầu phải có sườn dương kích F_4 . Vậy có thể dùng $\overline{Q_1}$ và CP , nhưng ta chọn $\overline{Q_1}$ vì số lần biến đổi ít hơn.

Dưới đây giải thích thêm luận điểm trên: “Dưới tiền đề thoả mãn yêu cầu chuyển đổi trạng thái thì số xung đồng hồ càng ít càng tốt”. Nếu xung đồng hồ càng ít thì yêu cầu kích (điều khiển đầu vào đồng hồ) càng đơn giản. Giả sử không có xung đồng hồ thì không cần điều khiển, FF duy trì không đổi nguyên trạng. Giả sử cần có xung đồng hồ mỗi khi cần chuyển đổi trạng thái thì xác định phương trình kích kiểu Flip Flop T’ là được. Giả sử cần có xung đồng hồ cả lúc không cần chuyển đổi trạng thái thì phải thêm điều kiện điều khiển đầu vào đồng bộ để bảo đảm bộ đếm rõ ràng càng phức tạp.

Tìm phương trình trạng thái, phương trình ra, kiểm tra tự khởi động

a) Tìm phương trình trạng thái – phương trình trạng thái kế tiếp của mỗi FF. Hình 6-6-21 là bảng Karnaugh của trạng thái kế tiếp bộ đếm. Cần lưu ý rằng khi xét trạng thái kế tiếp của bất kì

Flip Flop nào, ngoài những trạng thái không được dùng, cả các trạng

$Q_1^i Q_2^i$		$Q_3^i Q_4^i$			
		00	01	11	10
00	00	0001	0010	0100	0011
01	00	0101	0110	1000	0111
11	00	×	×	×	×
10	00	1001	0000	×	×

Hình 6-6-21. Bảng Karnaugh của trạng thái thiết kế tiếp của bộ đếm

thái không kèm kích của xung đồng hồ, chúng đều được dùng để tối thiểu hoá. Ví dụ, đối với trạng thái kế tiếp Q_4^{n+1} của F_4 , ngoài những trạng thái không được dùng $S_{10} \div S_{15} = 1010 \div 1111$, thì những trạng thái nào không thoả mãn điều kiện kích

$$CP_4 = \sqrt{Q_1}$$

đều được dùng để tối thiểu hóa. Nếu Q_1 bất biến hoặc lật từ 0 sang 1 thì $\overline{Q_1}$ bất biến hoặc lật từ 1 sang 0 đều không thoả mãn điều kiện kích. Xem xét hình 6-6-19, ta thấy S_0, S_2, S_4, S_8 là các trạng thái không kèm kích (sườn dương) của CP_4 . Xem hình 6-6-22a. Hình 6-6-22b là bảng Karnaugh của Q_2^{n+1} cũng được giải thích tương tự (lưu ý $CP_2 = CP_4$)

Hình 6-6-22. Bảng Karnaugh của trạng thái kế tiếp.

		$Q_2^n Q_1^n$			
		00	01	11	10
$Q_3^n Q_2^n$	00	×	0	0	×
	01	×	0	1	×
	11	×	×	×	×
	10	×	0	×	×

(a) Q_4^{n+1}

		$Q_2^n Q_1^n$			
		00	01	11	10
$Q_3^n Q_2^n$	00	×	1	0	×
	01	×	1	0	×
	11	×	×	×	×
	10	×	0	×	×

(b) Q_2^{n+1}

Tối thiểu hoá bảng Karnaugh hình 6-6-22- ta có

$$Q_4^{n+1} = Q_3^n Q_2^n$$

$$Q_2^{n+1} = \overline{Q_4^n} \overline{Q_2^n} \quad (6-6-10)$$

Đối với Q_3^{n+1} của F_3 , vì $CP_3 = \overline{Q_2}$ (6-6-9), đối chiếu đồ hình trạng thái hình

6-6-19, những trạng thái dùng xử lý tối thiểu hoá là:

Không được dùng: $S_{10} \div S_{15}$

$\overline{Q_2}$ bất biến: $S_0, S_2, S_4, S_6, S_8, S_9$

$\overline{Q_2}$ từ 1 lật sang 0 (Q_2 từ 0 sang 1): S_1, S_5

Hình 6-6-23. bảng Karnaugh của trạng thái kế tiếp.

		$Q_2^n Q_1^n$			
		00	01	11	10
$Q_3^n Q_2^n$	00	×	×	1	×
	01	×	×	0	×
	11	×	×	×	×
	10	×	×	×	×

(a) Q_3^{n+1}

		$Q_2^n Q_1^n$			
		00	01	11	10
$Q_3^n Q_2^n$	00	1	0	0	1
	01	1	0	0	1
	11	×	×	×	×
	10	1	0	×	×

(b) Q_1^{n+1}

Tối thiểu hóa bảng Karnaugh hình 6-6-23, ta được :

$$\begin{aligned} Q_3^{n+1} &= \overline{Q_3^n} \\ Q_1^{n+1} &= \overline{Q_1^n} \end{aligned} \quad (6-6-11)$$

Ta sẽ giải thích việc dùng những trạng thái không kèm kích xung đồng hồ để tối thiểu hóa như sau :

Căn cứ đặc điểm chức năng logic của Flip Flop, chúng ta biết rằng trạng thái kế tiếp là hàm số của đầu vào và hiện trạng. Trạng thái kế tiếp của bộ đếm biểu thị bằng trạng thái kế tiếp của các Flip Flop cấu trúc bộ đếm đó. Vậy trạng thái kế tiếp của mỗi FF cũng là hàm số của đầu vào và hiện trạng bộ đếm. Khi bộ đếm chuyển đổi dưới tác động tín hiệu đầu vào, giả sử Flip Flop F_i nào đó không có kích của xung đồng hồ thì trạng thái F_i bất biến ($Q_i^{n+1} = Q_i^n$). Xét ví dụ hình 6-6-21, hiện trạng $Q_4^n Q_3^n Q_2^n Q_1^n = 0000$, khi CP đến, trạng thái kế tiếp $Q_4^{n+1} Q_3^{n+1} Q_2^{n+1} Q_1^{n+1} = 0001$, Q_1 từ 0 sang 1, $\overline{Q_1}$ từ 1 sang 0. Vì Flip Flop D kích bằng sườn dương được dùng trong mạch này, mà $CP_4 = CP_2 = \overline{Q_1}$, nên F_4, F_2 không có kích, S_0 được dùng tối thiểu hóa.

b) Tìm phương trình ra

Hình 6 - 6- 24 : Bảng Karnaugh của chuyển vị C.

Tối thiểu hóa, ta có :

$$C = Q_4^n Q_1^n \quad (6-6-12)$$

c) Kiểm tra khả năng tự khởi động :

Sử dụng phương trình trạng thái và phương

trình ra để phân tích tình huống chuyển đổi trạng thái không được dùng.

Hình 6-6-25

Tình huống chuyển đổi trạng thái không được dùng.

Bảng 6-6-5

TÌNH HUỐNG CHUYỂN ĐỔI CỦA TRẠNG THÁI KHÔNG ĐƯỢC DÙNG

Q_4^n	Q_3^n	Q_2^n	Q_1^n	Q_4^{n+1}	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	C	
1	0	1	0	1	0	1	1	0	CP_1
1	0	1	1	0	1	0	0	1	$CP_1 CP_2 CP_3 CP_4$
1	1	0	0	1	1	0	1	0	CP_1
1	1	0	1	0	1	0	0	1	$CP_1 CP_2 CP_4$
1	1	1	0	1	1	1	1	0	CP_1
1	1	1	1	1	0	0	0	1	$CP_1 CP_2 CP_3 CP_4$

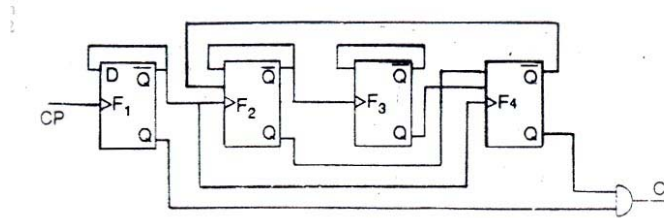
Xem xét hình 6-6-25 và bảng 6-6-5 ta thấy bộ đếm đã thiết kế có thể tự khởi động.

Tìm phương trình kích :

Đồng nhất phương trình trạng thái bộ đếm (6-6-10) và (6-6-11) với phương trình đặc trưng của Flip Flop D cấu trúc nên bộ đếm ta có :

$$\begin{cases} D_1 = \overline{Q_1^n} \\ D_2 = \overline{Q_4^n} \overline{Q_2^n} \\ D_3 = \overline{Q_3^n} \\ D_4 = Q_3^n Q_2^n \end{cases} \quad (6-6-13)$$

Vẽ sơ đồ logic

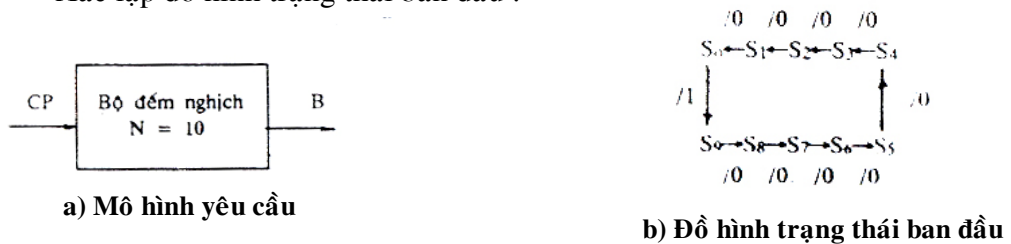


Hình 6-6-26 : Bộ đếm thuận thập phân đi bộ.

Ví dụ 6-6-5 : Hãy thiết kế bộ đếm nghịch thập phân đi bộ.

Bài giải :

Xác lập đồ hình trạng thái ban đầu :

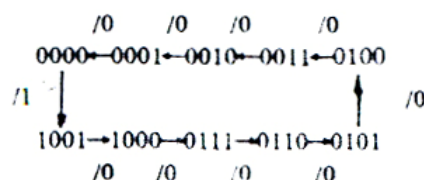


(Hình 6-6-27) : Bộ đếm nghịch thập phân

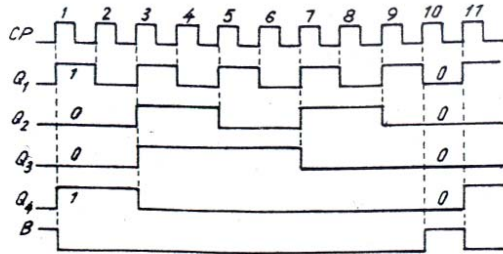
Xác định số lượng chủng loại FF, chọn lựa mã hóa trạng thái.

Vì $2^n \geq N = 10$

Vậy $n = 4$, chọn Flip Flop D ; chọn mã 8421



Hình 6-6-28 : Đồ hình trạng thái bộ đếm nghịch (sau khi mã hóa).



Hình 6-6-29 : Dạng sóng bộ đếm nghịch.

Chọn xung đồng hồ :

$$\begin{cases} CP_1 = CP \\ CP_2 = Q_1 \\ CP_3 = Q_2 \\ CP_4 = Q_1 \end{cases}$$

(6-6-14)

Tìm phương trình trạng thái

Hình 6-6-30 : Bảng Karnaugh trạng thái kế tiếp bộ đếm nghịch.

$Q_2^i Q_1^i$		$Q_2^i Q_1^i$			
		00	01	11	10
$Q_2^i Q_1^i$	00	1001	0000	0010	0001
	01	0011	0100	0110	0101
	11	x	x	x	x
	10	0111	1000	x	x

$Q_2^i Q_1^i$		$Q_2^i Q_1^i$			
		00	01	11	10
$Q_2^i Q_1^i$	00	0	x	x	0
	01	0	x	x	0
	11	x	x	x	x
	10	0	1	x	x

(a) Q_2^{i+1}

$Q_2^i Q_1^i$		$Q_2^i Q_1^i$			
		00	01	11	10
$Q_2^i Q_1^i$	00	x	x	x	x
	01	0	x	x	x
	11	x	x	x	x
	10	1	x	0	0

(b) Q_1^{i+1}

$Q_2^i Q_1^i$		$Q_2^i Q_1^i$			
		00	01	11	10
$Q_2^i Q_1^i$	00	0	x	x	0
	01	1	x	x	0
	11	x	x	x	x
	10	1	x	x	x

(c) Q_0^{i+1}

Hình 6-6-31 : Bảng Karnaugh trạng thái kế tiếp của các FF.

$Q_2^i Q_1^i$		$Q_2^i Q_1^i$			
		00	01	11	10
$Q_2^i Q_1^i$	00	1	0	0	1
	01	1	0	0	1
	11	x	x	x	x
	10	1	0	x	x

Tối thiểu hóa bảng Karnaugh hình 6-6-31 :

$$\begin{cases} Q_4^{n+1} = \overline{Q_4^n} \overline{Q_3^n} \overline{Q_2^n} \\ Q_3^{n+1} = \overline{Q_3^n} \\ Q_2^{n+1} = Q_4^n + Q_3^n \overline{Q_2^n} \\ Q_1^{n+1} = \overline{Q_1^n} \end{cases} \quad (6-6-15)$$

Tìm phương trình ra :

Tối thiểu hóa bảng Karnaugh hình 6-6-

32 ta có :

$$B = \overline{Q_4^n} \overline{Q_3^n} \overline{Q_2^n} \overline{Q_1^n} \quad (6-6-16)$$

Kiểm tra khả năng tự khởi động

$Q_2^n Q_1^n$	$Q_4^n Q_3^n$			
	00	01	11	10
00	1	0	0	0
01	0	0	0	0
11	x	x	x	x
10	0	0	x	x

Hình 6-6-32 : Bảng Karnaugh của chuyển vị B.

Bảng 6-6-6 : TÌNH HUỐNG CHUYỂN ĐỔI TRẠNG THÁI KHÔNG ĐƯỢC DÙNG

Q_4^n	Q_3^n	Q_2^n	Q_1^n	Q_4^{n+1}	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}	B	
1	0	1	0	0	0	1	1	0	CP ₁ CP ₂ CP ₄
1	0	1	1	1	0	1	0	0	CP ₁
1	1	0	0	0	0	1	1	0	CP ₁ CP ₂ CP ₃ CP ₄
1	1	0	1	1	1	0	0	0	CP ₁
1	1	1	0	0	1	1	1	0	CP ₁ CP ₂ CP ₄
1	1	1	1	1	1	1	0	0	CP ₁

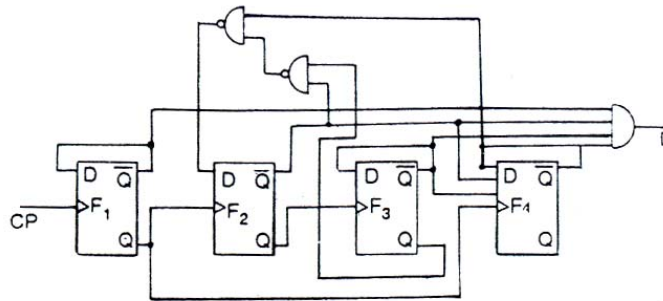
(Hình 6-6-33) : Tình huống chuyển đổi trạng thái không được dùng.

Từ bảng 6-6-6 và hình 6-6-33 ta thấy bộ đếm tự khởi động được.

Tìm ra phương trình kích :

$$\begin{cases} D_4 = \overline{Q_4^n} \overline{Q_3^n} \overline{Q_2^n} \\ D_3 = \overline{Q_3^n} \\ D_2 = Q_4^n + Q_3^n \overline{Q_2^n} \\ D_1 = \overline{Q_1^n} \end{cases} \quad (6-6-17)$$

Vẽ sơ đồ logic.



Hình 6-6-34 : Bộ đếm nghịch thập phân dị bộ.

Phương pháp thiết kế trên đây có thể áp dụng cho các mạch dãy đệ bộ nói chung. Chỉ cần chú ý tính đặc thù trong việc chọn xung đồng hồ. Ví dụ, trong tình huống tín hiệu đầu vào biến đổi ngẫu nhiên thì rất khó vẽ ra đồ thị dạng sóng ; do đó việc chọn xung đồng hồ cho các FF của bộ đếm theo dạng sóng là vô cùng phức tạp. Tuy nhiên, phương pháp thiết kế mạch dãy đệ bộ không gặp khó khăn còn chưa tìm ra.

6.6.3. Thiết kế mạch dây

Thiết kế mạch dây nói chung so với thiết kế bộ đếm trên đây có ba vấn đề đặc biệt như sau :

1. Xây dựng đồ hình trạng thái tương đối khó khăn. Thông thường yêu cầu thiết kế là ngôn ngữ thông thường, có thể là một vấn đề logic cụ thể của mạch dây. Trong vấn đề đó hàm chứa dữ liệu nào đó cần nhớ, biểu thị bằng cách nào đó thành trạng thái mạch dây, tồn tại một quan hệ logic nào đó giữa hiện trạng, đầu vào, đầu ra và trạng thái kế tiếp. Từ đó làm thế nào xây dựng đồ hình trạng thái là công việc phân tích cụ thể đối với từng vấn đề cụ thể, vì chưa có quy trình gì. Chỉ riêng đối với bộ đếm thì vấn đề này mới rõ ràng thế.

2. Cần tối thiểu hóa trạng thái. Từ yêu cầu thiết kế xây dựng đồ hình trạng thái thường có nhiều trạng thái thừa. Trạng thái mạch dây được Flip Flop nhớ và thể hiện. Vậy trạng thái càng nhiều thì số FF cần dùng càng nhiều. Nếu thiết kế đảm bảo yêu cầu đề ra, thì mong muốn trạng thái càng ít càng tốt. Chỉ riêng đối với bộ đếm thì mới rõ ràng đến thế : bộ đếm N có N trạng thái ; không có vấn đề tối thiểu hóa trạng thái đối với bộ đếm.

3. Việc chọn lựa mã hóa trạng thái cũng không dễ như đối với bộ đếm. Quan hệ giữa các trạng thái trong mạch dây tương đối phức tạp. Mà việc chọn lựa mã hóa trạng thái liên quan đến tính phức tạp của phương trình trạng thái và phương trình ra, ảnh

hưởng tính kinh tế của phương án khả thi. Cho nên thường nghiên cứu, cân nhắc, so sánh, làm lại để có thiết kế tối ưu. Hiện tại chưa có quy trình về việc chọn lựa mã hóa trạng thái. Chỉ riêng đối với bộ đếm thì quan hệ giữa các trạng thái tương đối đơn giản, có mấy loại mã đếm thường dùng, phương án mã hóa trạng thái dễ chọn.

Trên quan điểm của người sử dụng các IC mạch dây, biết phương pháp chung là đủ. Dưới đây ta xét một ví dụ thiết kế mạch dây.

Ví dụ 6-6-6 : Thiết kế bộ giám sát dữ liệu nối tiếp. Yêu cầu đối với bộ giám sát là : Nếu số bit 1 liên tục nhau ≥ 3 , thì đầu ra là 1 (các trường hợp khác của đầu vào chỉ có giá trị 0 ở đầu ra).

Bài giải :

a) Phân tích yêu cầu thiết kế, xây dựng đồ hình trạng thái ban đầu. Gọi số bit 1 liên tục nhau ở đầu vào mạch là m.

m = 0 ứng với trạng thái S_0

m = 1 ứng với trạng thái S_1

m = 2 ứng với trạng thái S_2

m ≥ 3 ứng với trạng thái S_3

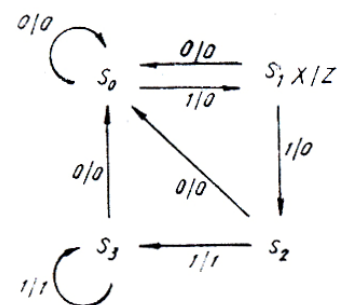
Vậy mạch cần 4 trạng thái khác nhau. Căn cứ yêu cầu thiết kế, vẽ ra đồ hình trạng thái ban đầu như hình 6-6-35. Trong hình ký hiệu X/Z với X là đầu vào, Z là đầu ra.

Hình 6-6-35 cho thấy rằng, khi mạch điện ở trạng thái ban đầu S_0 , giả sử đầu vào có bit 1 thì đầu ra là 0 và trạng thái kế tiếp là S_1 ; Giả sử đầu vào có bit 0 tiếp theo thì đầu ra là 0 và trạng thái kế tiếp là S_0 . Nếu hiện trạng bộ giám sát là S_1 , giả sử đầu vào có bit 1 thì đầu ra là 0 và trạng thái kế tiếp là S_2 ; Giả sử đầu vào có bit 0 thì đầu ra là 0 và trạng thái kế tiếp là S_0 (tức về trạng thái ban đầu).

Nếu hiện trạng bộ giám sát là S_2 , giả sử đầu vào có bit 1, tức là đã đủ 3 bit 1 liên nhau thì đầu ra là 1 và trạng thái kế tiếp là S_3 ; Giả sử đầu vào có bit 0 thì đầu ra là 0 và trạng thái kế tiếp là S_0 .

b) Tiến hành tối thiểu hóa trạng thái

Gộp trạng thái tương đương. Trạng thái tương đương là những trạng thái hiện tại nào trong điều kiện đầu vào hiện tại như nhau thì có cùng giá trị logic đầu ra và có cùng trạng thái kế tiếp. Vậy trạng thái tương đương lập lại nhau nên có thể gộp.



Hình 6-6-35 : Đồ hình trạng thái ban đầu của bộ giám sát.

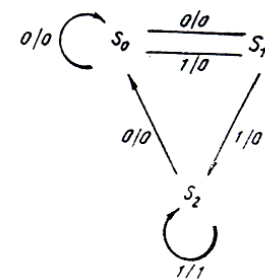
Để xem xét có các trạng thái tương đương nhau không, từ hình 6-6-35 ta xây dựng bảng trạng thái 6-6-7.

Bảng 6-6-7 : CÁC TRẠNG THÁI BAN ĐẦU CỦA BỘ GIÁM SÁT

Trạng thái hiện tại	Trạng thái kế tiếp/ đầu ra	
	Đầu vào hiện tại X	
	X = 0	X = 1
S_0	$S_0/0$	$S_1/0$
S_1	$S_0/0$	$S_2/0$
S_2	$S_0/0$	$S_3/1$
S_3	$S_0/0$	$S_3/1$

Nhận xét bảng 6-6-7, ta thấy các trạng thái S_2, S_3 là tương đương.

Sau khi tiến hành tối thiểu hóa trạng thái, ta được đồ hình trạng thái hình 6-6-36.



Hình 6-6-36 : Đồ hình trạng thái sau tối thiểu hóa

c) Xác định số lượng chủng loại FF, chọn lựa mã hóa trạng thái.

$$\text{Vì } 2^n \geq N = 3$$

Vậy $n = 2$. Chọn Flip Flop D.

2 FF có 4 trạng thái 00, 01, 10, 11 - Trong số nhiều cách mã hóa khả dĩ, chúng ta chọn :

$$S_0 = 00$$

$$S_1 = 01$$

$$S_2 = 11$$

Về sau ta phải xét xem mạch điện có tự khởi động không và đã đơn giản nhất chưa.

d) Tìm phương trình trạng thái và phương trình ra.

Hình 6-6-37 : Bảng Karnaugh của trạng thái kế tiếp/ đầu ra.

		$Q_2 Q_1$			
		00	01	11	10
X	0	00/0	00/0	00/0	x
	1	01/0	11/0	11/1	x

Trạng thái 10 không được dùng, trong quá trình tối thiểu hóa cần lưu ý xử lý để đạt kết quả như sau :

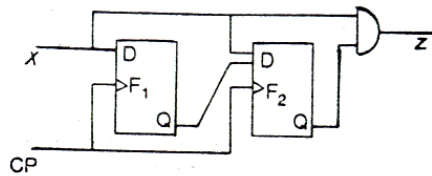
$$\begin{cases} Q_2^{n+1} = XQ_1^n \\ Q_1^{n+1} = X \end{cases} \quad (6-6-18)$$

$$Z = XQ_2^n \quad (6-6-19)$$

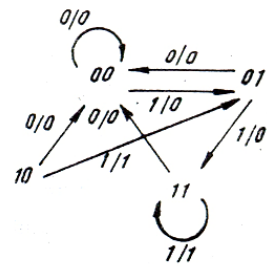
e) Tìm phương trình kích

$$\begin{cases} D_2 = XQ_1^n \\ D_1 = X \end{cases} \quad (6-6-20)$$

g) Vẽ sơ đồ logic



Hình 6-6-38 : Bộ giám sát dữ liệu kết nối.



Hình 6-6-39 : Đồ hình trạng thái bộ giám sát.

Hình 6-6-39 là đồ hình trạng thái vẽ từ sơ đồ logic hình 6-6-38 . Ta nhận thấy mạch tự khởi động được và khá đơn giản.

PHỤ LỤC II

HỆ ĐẾM VÀ CHUYỂN ĐỔI

I. CÁC HỆ ĐẾM THƯỜNG DÙNG

1. Hệ thập phân

Hệ thập phân là hệ đếm thường dùng nhất trong sinh hoạt và công tác của chúng ta. Trong hệ này, có 10 chữ số $0 \div 9$ để mã hóa số 0 và 9 số tự nhiên đầu tiên. Vậy cơ số hệ thập phân là 10. Từ số lớn hơn 9, nhờ cách ghi số theo vị trí, trong đó số có vị trí bất kỳ có trọng số gấp mười lần số có vị trí bên phải kề nó ; ta có thể dùng 10 chữ số để biểu diễn mọi con số. Ví dụ :

$$143,75 = 1 \times 10^2 + 4 \times 10^1 + 3 \times 10^0 + 7 \times 10^{-1} + 5 \times 10^{-2}$$

Một số dương S bất kỳ trong hệ thập phân có thể khai triển thành :

$$S = \sum k_i 10^i \quad (\text{II} - 1)$$

i là thứ tự vị trí của số tính từ dấu phân cách nguyên - phân ($i = 0$) k_i là hệ số của số có vị số (thứ tự vị trí của số) i , $k_0 = 0 \div 9$, nếu phần nguyên có n chữ số thì tương ứng $i = n - 1 \div 0$. Nếu phần phân có m chữ số thì tương ứng $i = -1 \div -m$. Dùng N thay cho cơ số 10, biểu thức (II - 1) có dạng tổng quát cho mọi hệ đếm .

$$S = \sum k_i N^i \quad (\text{II} - 2)$$

2. Hệ nhị phân

Hệ nhị phân được dùng rộng rãi nhất trong mạch số. Trong hệ nhị phân, mỗi vị số (bit) chỉ có hai khả năng lấy giá trị : 1 và 0. Có số đếm của hệ nhị phân $N = 2$.

Triển khai số nhị phân bất kỳ theo dạng (II - 2), ta có :

$$S = \sum k_i 2^i \quad (\text{II} - 3)$$

với $k_i = 0, 1$

Ví dụ :

$$101,11 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$$

3. Hệ đếm cơ số 8

Trong hệ này, mỗi vị số có 8 mã số là $0 \div 7$, cơ số là 8.

Dạng tổng quát của số đếm hệ cơ số 8 là :

$$S = \sum k_i 8^i \quad (\text{II} - 4)$$

với $k_i = 0, 1, 2, 3, 4, 5, 6, 7$

Ví dụ :

$$37,41 = 3 \times 8^1 + 7 \times 8^0 + 4 \times 8^{-1} + 1 \times 8^{-2}$$

Cùng một con số, dạng biểu thị trong hệ đếm cơ số 8 gọn hơn dạng biểu thị trong hệ đếm nhị phân. Như sau này sẽ rõ, sự chuyển đổi lẫn nhau của hai hệ này lại cực dễ dàng, nên trong các sách viết về trình tự máy tính hay dùng hệ đếm cơ số 8.

4. Hệ đếm cơ số 16

Trong hệ này, mỗi vị số có 16 mã số. đó là 0,1,2,3,4,5,6,7,8,9, A(10), B(11), C(12), D(13), E(14), F(15).

Dạng tổng quát của số đếm hệ cơ số 16 là :

$$S = \sum k_i 16^i$$

với $k_i = 0 \div F$

Ví dụ :

$$\begin{aligned} 2a, 7F &= 2 \times 16^1 + A \times 16^0 + 7 \times 16^{-1} + F \times 16^{-2} \\ &= 2 \times 16 + 10 \times 1 + \frac{7}{16} + \frac{15}{16^2} \\ &= 42,4961 \end{aligned}$$

Hiện nay trong máy vi tính, đa số dùng từ mã nhị phân 8 bit, 16 bit. Những từ mã này có thể biểu thị gọn rõ bằng số trong hệ đếm 16 với 2, 4 vị số tương ứng. Vậy trong các sách viết về trình tự máy tính càng hay dùng hệ đếm cơ số 16. Sự chuyển đổi lẫn nhau của hệ nhị phân và hệ đếm cơ số 16 cùng cực kỳ dễ dàng. Nên ứng dụng hệ đếm cơ số 16 còn rộng rãi hơn hệ đếm cơ số 8.

II. CHUYỂN ĐỔI LẦN NHAU GIỮA CÁC HỆ ĐẾM

1. Hệ nhị phân và hệ thập phân

Từ nhị phân sang thập phân : viết số nhị phân dưới dạng (II - 3) triển khai : cộng tất cả các số hạng theo giá trị số thập phân, tổng số là dạng thập phân của số nhị phân đã cho. Ví dụ :

$$\begin{aligned} 1011,01_{(2)} &= 1 \times 3^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} \\ &= 8 + 0 + 2 + 1 + 0 + 0,25 \\ &= 11,25_{(10)} \end{aligned}$$

Lưu ý : chúng ta dùng chữ số trong ngoặc viết thấp để chỉ cơ số hệ đếm của con số xét.

Từ thập phân sang nhị phân :

a) Phần nguyên

Trong các đẳng thức dưới đây, vế phải là số nhị phân, vế trái là số thập phân

$$\begin{aligned} S_{(10)} &= k_n 2^n + k_{n-1} 2^{n-1} + \dots + k_1 2^1 + k_0 \\ &= 2(k_n 2^{n-1} + k_{n-1} 2^{n-2} + \dots + k_1) + k_0 \end{aligned}$$

vì $k_i = 0,1$ đồng phân với số 0, 1 trong hệ thập phân, nên ta có :

$$\begin{aligned}\frac{S_{(10)} - k_0}{2} &= k_n 2^{n-1} + k_{n-1} 2^{n-2} + \dots + k_1 \\ &= 2(k_n 2^{n-2} + k_{n-1} 2^{n-3} + \dots + k_2) + k_1\end{aligned}$$

Nhận xét các biểu thức trên, ta thấy :

bit đầu tiên của số nhị phân là k_0 bằng số dư khi chia $S_{(10)}$ cho 2.

bit đầu tiên của số nhị phân là k_1 bằng số dư khi chia thương số của phép chia trước cho 2.

Tương tự như vậy để tìm toàn bộ các bit của số nhị phân. Ví dụ :

173	2	dư 1	k_0	Vậy $173_{(10)} = 10101101_{(2)}$
86	2	dư 0	k_1	
43	2	dư 1	k_2	
21	2	dư 1	k_3	
10	2	dư 0	k_4	
5	2	dư 1	k_5	
2	2	dư 0	k_6	
1	2	dư 1	k_7	
0				

b) Phần phân :

Trong các đẳng thức dưới đây, vế phải là số nhị phân, vế trái là số thập phân :

$$S_{(10)} = k_1 2^{-1} + k_2 2^{-2} + k_3 2^{-3} + \dots + k_m 2^{-m}$$

Nhân 2 vế với 2, ta có :

$$2S_{(10)} - k_1 = k_2 2^{-1} + k_3 2^{-2} + \dots + k_m 2^{-m+1}$$

Nếu tiếp tục nhân 2 vế với 2 ta lại được k_2 là phần nguyên của vế phải (của tích số lần thứ 2) :

$$2[2S_{(10)} - k_1] = k_2 + (k_3 2^{-1} + \dots + k_m 2^{-m+2})$$

Tương tự như vậy, ta tìm được toàn bộ các bit của số nhị phân. Ví dụ :

Quá trình chuyển đổi từ $0,8128_{(10)}$ sang $0,1101_{(2)}$ như sau :

$$0,8128 \times 2 = 1,6256 = 0,6256 + 1 \quad (k_1)$$

$$0,6256 \times 2 = 1,2512 = 0,2512 + 1 \quad (k_2)$$

$$0,2512 \times 2 = 0,5024 = 0,5024 + 0 \quad (k_3)$$

$$0,5024 \times 2 = 1,0048 = 1 + 0 \quad (k_4)$$

Quá trình kết thúc vì phần phân của tích số bằng 0.

2. Hệ nhị phân và hệ đếm cơ số 8

Từ nhị phân sang hệ 8

Vì $2^3 = 8$, mỗi vị số của hệ 8 tương ứng với một nhóm 3 bit của số nhị phân, bắt đầu từ bit 2^0 . Muốn chuyển đổi từ nhị phân sang hệ 8, đầu tiên ta phân nhóm 3 bit, sau đó dùng 8 chữ số của hệ 8 thay cho 8 mã số tương ứng của 3 bit.

Đối với phần phân, chia nhóm 3 bit bắt đầu từ 2^{-1} .

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

Quá trình như sau :

Chia nhóm	010	110	101,	001	111	010
Chuyển mã	2	6	5,	1	7	2

Từ hệ 8 sang nhị phân :

Thay một chữ số trong số hệ 8 bằng nhóm 3 bit nhị phân

Ví dụ :

	5	1	2,	3	0	4 ₍₈₎
=	101	001	010	011	000	100 ₍₂₎

3. Hệ nhị phân và hệ đếm cơ số 16

Từ nhị phân sang hệ 16.

Vì $2^4 = 16$, mỗi vị số của số hệ 16 tương ứng với một nhóm 4 bit của số nhị phân, bắt đầu từ bit 2^0 . Muốn chuyển đổi từ nhị phân sang hệ 16, đầu tiên ta phân nhóm 4 bit, sau đó dùng 16 chữ số của hệ 16 thay cho 16 mã số tương ứng của 4 bit.

Đối với phần phân, chia nhóm 4 bit bắt đầu từ 2^{-1} .

Ví dụ : 0101, 1110, 1011, 0010₍₂₎ = 5E, B2₍₁₆₎

Quá trình như sau :

Chia nhóm	0101	1110,	1011	0010
Chuyển mã	5	E,	B	2

Từ hệ 16 sang nhị phân :

Thay một chữ số trong số hệ 16 bằng nhóm 4 bit nhị phân.

Ví dụ :

	8	F	A	C	6 ₍₁₆₎
=	1000	1111	1010	1100	0110

PHỤ LỤC III

PHƯƠNG PHÁP BIỂU THỊ SỐ NHỊ PHÂN TRONG MÁY TÍNH

1. Số trong máy tính và giá trị thực

Trong mạch số, giá trị 0 và 1 của 1 bit đều biểu thị bằng 2 trạng thái của một phần logic cơ bản (mức điện cao, thấp của cổng hay của Flip Flop). Vậy dấu âm, dương của số được biểu thị bằng cách nào. Rõ ràng, dấu cũng biểu thị bằng hai trạng thái logic, tức là dùng 0 hay 1 biểu thị. Phương pháp đơn giản nhất là, cho thêm 1 bit dấu đằng trước : giá trị 0 của bit dấu biểu thị số dương, giá trị 1 của bit dấu biểu thị số âm. Ví dụ đối với số nhỏ hơn 1 về giá trị tuyệt đối, số +0, 1011 trong máy tính được biểu thị là 0,1011, và số -0,1011 trong máy tính được biểu thị là 1,1011. Tức là dùng giá trị 0 và 1 của phần nguyên biểu thị dấu dương và âm của số nhị phân nhỏ hơn 1 về giá trị tuyệt đối. Trong ví dụ trên, +0, 1011 và -0,1011 là giá trị thực của số 0,1011 và 1,1011 tương ứng trong máy tính.

Dưới đây giới thiệu 3 dạng thường gặp của số trong máy tính : mã gốc, mã bù và mã đảo.

Trong máy tính nhỏ và thiết bị điều khiển số nói chung đều áp dụng phép toán dấu phẩy cố định. Mã gốc, mã bù và mã đảo được giới thiệu tiếp theo đây là tương ứng với phép toán dấu phẩy cố định. Khi đó, giả định rằng trong toàn bộ phép toán, giá trị tuyệt đối của mọi số đều nhỏ hơn 1. Dấu phẩy đặt trước bit cao nhất (2^{-1}). Nếu như giá trị tuyệt đối của số thực tế lớn hơn 1 thì phép toán vẫn tiến hành như quy ước trên nhờ quá trình chuyển đổi thuận nghịch tỉ lệ giá trị.

2. Mã gốc

Trong mã gốc, bit dấu đặt trước dấu phẩy của phân số (số có giá trị tuyệt đối nhỏ hơn 1). Số dương tương ứng giá trị 0 của bit dấu. Số âm tương ứng giá trị 1 của bit dấu. Vậy số dương có mã gốc trùng với giá trị thực, số âm có mã gốc bằng giá trị thực cộng 1, tức là :

$$[x]_{\text{mã gốc}} = \begin{cases} x & x \geq 0 \\ 1 + |x| & x < 0 \end{cases}$$

Vậy số 0 có hai hình thức mã gốc :

$$[0]_{\text{mã gốc}} = \begin{cases} 0,0000 \\ 1,0000 \end{cases}$$

Mã gốc có ưu điểm là trực quan, đơn giản, rất tiện trong phép nhân. Tích số có phần giá trị tuyệt đối và phần dấu. Giá trị tuyệt đối của tích số bằng tích các giá trị tuyệt đối của thừa số. Dấu của tích theo quy tắc logic : các thừa số cùng dấu

thì tích số dương, các thừa số khác dấu thì tích số âm. Quy tắc logic này rất dễ thực hiện.

Nhưng chúng ta sẽ gặp khó khăn nếu làm phép cộng trừ với số mã gốc. Ví dụ: khi cộng đại số, đầu tiên phải xét dấu các số hạng, sau đó mới quyết định về cộng trừ các giá trị tuyệt đối. Khi trừ, còn phải so sánh giá trị tuyệt đối, từ đó mới định hướng lấy gì trừ cho gì, cuối cùng xác định dấu của hiệu số. Mỗi một bước như vậy đều bắt máy tính phải thao tác, vậy tất sẽ làm cho máy tính phức tạp lên và thời gian làm tính kéo dài.

Để tránh khó khăn trên trong phép cộng trừ, hiện nay máy tính dùng rộng rãi mã bù trong phép cộng trừ.

3. Mã bù

Để dễ dàng về mã bù, xin độc giả hãy xét một ví dụ liên quan trong cuộc sống đời thường sau đây. Giả sử lúc 06h00 sáng bạn phát hiện đồng hồ bị chết với kim đồng hồ ở vị trí 11h00. Bạn có 2 cách để chỉnh kim đồng hồ từ vị trí 11h00 đến vị trí 06h00 : cách thứ nhất đưa kim ngược chiều 5 khoảng giờ ($11-5=6$), cách thứ hai đưa kim thuận chiều 7 khoảng giờ ($11 + 7 = 18$) ; vì mặt đồng hồ có 12 khoảng giờ thôi, cứ vượt 12 thì lại bắt đầu từ 0, nên đưa kim thuận chiều 7 khoảng giờ thì kim cũng đến vị trí 06h00. Điều đó chứng tỏ rằng, nếu xét riêng một vị số (mà không đếm chuyển vị từ vị số bên phải liền kề sang) thì $11-5$ và $11+7$ trong hệ đếm cơ số 12 có cùng một kết quả là 6. Tổng của $5 + 7 = 12$. Ta gọi 7 là mã bù của 5 trong hệ 12. $7 = 12 - 5$.

Tương tự, trong phép trừ nhị phân, việc trừ đi một số nhị phân trở thành việc cộng mã bù trong hệ nhị phân của số đó.

Ví dụ : $a = 0,1011$; $b = -0,1001$ thì :

$$\begin{array}{r} 0,1011 \\ -0,1001 \\ \hline a + b = 0,0010 \end{array}$$

Chúng ta đi đến cùng kết quả trên bằng cách : tìm mã bù trong hệ nhị phân (cơ số 2) của b là :

$$[b]_{\text{mã bù}} = 10 - 0,1001 = 1,0111$$

Thực hiện cộng với mã bù, bỏ chuyển vị :

$$\begin{array}{r} 0,1011 \\ 1,0111 \text{ bỏ chuyển vị} \\ \hline a + [b]_{\text{mã bù}} = 10,0010 \longrightarrow 0,0010 \end{array}$$

Chúng ta viết biểu thức tổng quát mã bù 2 của số x nhị phân :

$$[x]_{\text{mã bù}} = \begin{cases} x & \text{nếu } x \geq 0 \\ 2 + x & \text{nếu } x \leq 0 \end{cases}$$

Ví dụ :

$x = -0,1011$	$[x]_{\text{mã bù}} = 10 - 0,1011 = 1,0101$
$x = -0,0001$	$[x]_{\text{mã bù}} = 10 - 0,0001 = 1,1111$
$x = -0,0000$	$[x]_{\text{mã bù}} = 10 - 0,0000 = 0,0000$

4. Mã đảo

Mã đảo của số nhị phân x là chính nó khi $x \geq 0$:

$$[x]_{\text{mã đảo}} = x$$

Còn khi $x \leq 0$ thì giá trị của mỗi bit phải đảo từ 0 sang 1 và từ 1 sang 0.

Ví dụ :

$$x = -0,0110 \quad [x]_{\text{mã đảo}} = 1,1001$$

$$x = -0,0001 \quad [x]_{\text{mã đảo}} = 1,1110$$

Xét đặc tính mã đảo qua ví dụ sau :

$$x = -0,0110 \quad [x]_{\text{mã đảo}} = 1,1001$$

$$-x + [x]_{\text{mã đảo}} = 0,0110 + 1,1001 + 1,1111$$

$$-x + [x]_{\text{mã đảo}} + 0,0001 = 10,0000$$

$$10 + x = [x]_{\text{mã đảo}} + 0,0001.$$

Vậy, một số nhị phân âm bất kỳ, nếu lấy giá trị tuyệt đối của nó cộng với mã đảo của chính nó, ta luôn được 1, 1111.

Nếu đem tổng số đó (giá trị tất cả các bit đều là 1) cộng với 1 ở bit cuối (tức cộng thêm 2^{-n}) thì kết quả luôn bằng 2.

Do đặc tính này, chúng ta ứng dụng mã đảo để tìm mã bù của một số nhị phân.

Dưới dạng số thập phân, ta có :

$$2 + x = [x]_{\text{mã đảo}} + 2^{-n}$$

$$\text{mà } 2 + x = [x]_{\text{mã bù}} \quad (\text{theo định nghĩa, khi } x \leq 0)$$

$$\text{Vậy : } [x]_{\text{mã bù}} = [x]_{\text{mã đảo}} + 2^{-n}$$

với n là số bit sau dấu phẩy của số nhị phân.

Ví dụ :

$$x = -0,1011 \quad [x]_{\text{mã bù}} = 1,0100 + 0,0001 = 1,0101$$

$$x = -0,0001 \quad [x]_{\text{mã bù}} = 1,1110 + 0,0001 = 1,1111$$

$$x = -0,0000 \quad [x]_{\text{mã bù}} = 1,1111 + 0,0001 = 0,0000$$

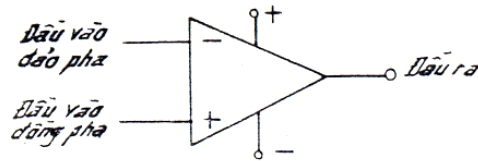
Vậy áp dụng phương pháp này, khi tìm mã bù của số nhị phân âm, ta không phải làm phép trừ như trên (theo định nghĩa).

PHỤ LỤC IV

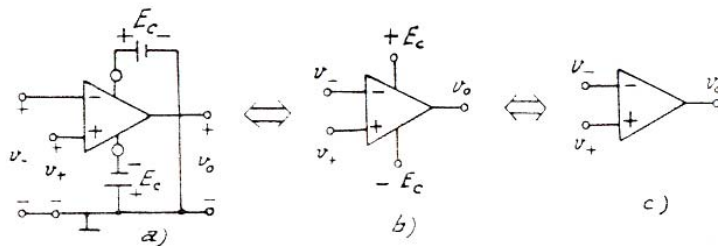
BỘ KHUYẾCH ĐẠI THUẬT TOÁN

Bộ khuếch đại thuật toán là phần quan trọng trong DAC và ADC. Với điện trở đầu vào vô cùng lớn, điện trở đầu ra vô cùng bé, hệ số khuếch đại điện áp cực lớn, kèm theo mạch phản hồi thích hợp thì bộ khuếch đại thuật toán dùng để xử lý tín hiệu vừa chính xác, lại tin cậy.

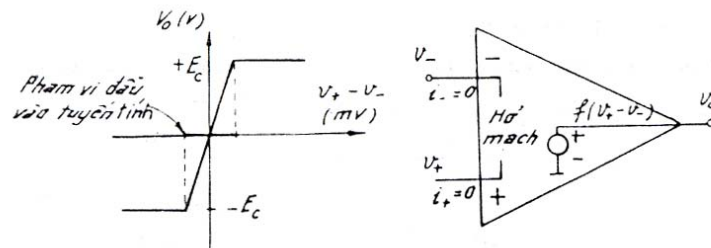
8.1.1. Ký hiệu và đặc tính của bộ khuếch đại thuật toán



Trên ký hiệu bộ khuếch đại thuật toán, dấu - ký hiệu đầu vào đảo pha (gọi tắt là đầu đảo), dấu + ký hiệu đầu vào đồng pha (gọi tắt là đầu thuận). Ngoài ra, còn có đầu ra và các đầu nối với nguồn một chiều.



Tuy sự cần thiết, chúng ta sẽ vẽ đầy đủ như sơ đồ a hay đơn giản như sơ đồ b, hoặc chỉ lưu ý đến tín hiệu như sơ đồ c.



Để xét đặc tính của bộ khuếch đại thuật toán, chúng ta hãy giả thiết lý tưởng hóa nó (thực tế giả thiết này không đưa tới sai số đáng kể nào). Bộ khuếch đại thuật toán lý tưởng có điện trở đầu vào vô cùng lớn, điện trở đầu ra bằng 0.

Trong phạm vi tuyến tính :

$$|v_0| < |E_c| \quad (IV-1)$$

$$v_0 = A_0(v_+ - v_-) \quad (IV-2)$$

$$|v_+ - v_-| = \left| \frac{v_0}{A_0} \right| < \left| \frac{E_c}{A_0} \right| \quad (IV-3)$$

A_0 là hệ số khuếch đại điện áp trong khu vực tuyến tính, số trị điển hình của $A_0 = 10^4 \div 10^6$. Giả sử $E_c = 10 \div 15V$, $A_0 = 10^6$. Ta thấy điện áp đầu vào trong khu vực tuyến tính chỉ cỡ vài $\square V$. Một cách gần đúng, đặc tính truyền đạt điện áp lý tưởng hóa của bộ khuếch đại thuật toán $A_0 = \infty$ như sau :

Kết luận về đặc tính bộ

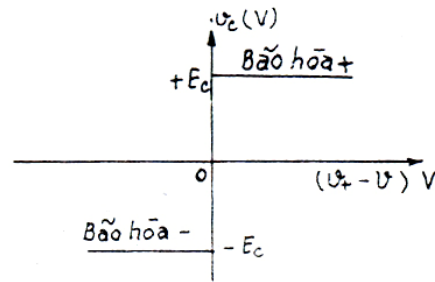
khuếch đại thuật toán như sau :

1. Dòng điện đầu vào bằng 0 :

$$i_+ = i_- = 0$$

2. $v_+ = v_-$

$$|v_+ - v_-| = \left| \frac{v_0}{\infty} \right| = 0$$



Nếu bộ khuếch đại thuật toán

công tác trong khu vực bão hòa thì $v_+ \neq v_-$.

Khi $v_+ > v_-$ ta có bão hòa +, $v_0 = +E_c$. Khi $v_+ < v_-$, ta có bão hòa -, $v_0 = -E_c$. Dòng điện đầu vào khi này cũng vẫn bằng 0.

Trở lại đặc tính truyền đạt với phạm vi đầu vào tuyến tính hữu hạn, ta thấy : v_0 tăng theo v_+ , v_0 đồng pha với v_+ , v_0 giảm khi v_- tăng, v_0 ngược pha với v_- .

8.1.2. Các sơ đồ khuếch đại thuật toán

1. Bộ khuếch đại đảo

Bộ khuếch đại thuật toán mắc thành bộ khuếch đại đảo như sau : đầu thuận nối đất.

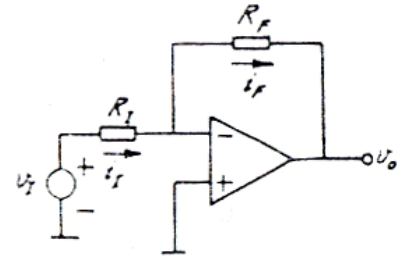
Điện áp đầu vào v_I đưa vào đầu đảo qua R_I , điện áp ra v_0 phản hồi đến đầu đảo qua R_F .

Vì $v_+ = v_- = 0$, nên :

$$i_I = \frac{v_I}{R_I} \quad (IV-4)$$

$$i_F = i_I \quad (IV-5)$$

$$v_0 = -i_F R_F = -i_I R_F = -\frac{R_F}{R_I} v_I \quad (IV-6)$$



Biểu thức (IV - 6) chứng tỏ khi V_1 tăng thì v_0 giảm, v_1 giảm thì v_0 tăng. Hệ số khuếch đại của bộ khuếch đại đảo bằng $A_v = -\frac{R_F}{R_1}$ không phụ thuộc bản thân bộ

khuếch đại thuật toán, A_v chỉ phụ thuộc thông số R_F, R_1 của mạch ngoài.

2. Bộ khuếch đại thuận

Bộ khuếch đại thuật toán mắc thành bộ khuếch đại thuận như sau :
điện áp đầu vào v_1 đưa tới đầu thuận qua R_1 ,
điện áp đầu ra v_0 qua phân áp R_1, R_2 đưa tới đầu đảo.

$$\text{vì } v_+ = v_-$$

(IV - 7)

$$\text{và } i_+ = i_- = 0, \text{ nên}$$

$$v_- = \frac{v_0}{R_1 + R_2} R_2 = v_1$$

(IV - 8)

từ đó, ta có :L

$$v_0 = \frac{R_1 + R_2}{R_2} v_1 = A_v v_1 \quad (\text{IV - 9})$$

Biểu thức (IV - 9) chứng tỏ khi v_1 tăng thì v_0 cũng tăng, v_1 giảm thì v_0 giảm theo. Hệ số khuếch đại của bộ khuếch đại thuận là :

$$A_v = \frac{R_1 + R_2}{R_2}$$

(IV - 10)

A_v cũng chỉ phụ thuộc thông số mạch ngoài (R_1, R_2)

3. Bộ khuếch đại lặp

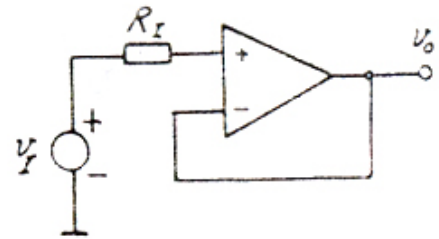
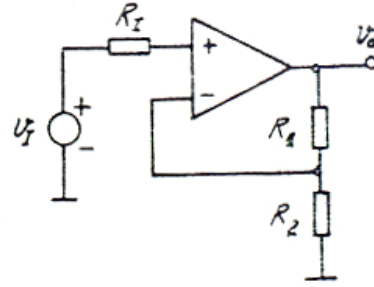
Bộ khuếch đại lặp là bộ khuếch đại thuận đặc biệt : đầu ra nối vào đầu đảo. Vậy $v_0 = v_1$. (IV - 11)

Bộ khuếch đại lặp thường dùng để kích tải (yêu cầu dòng đáng kể).

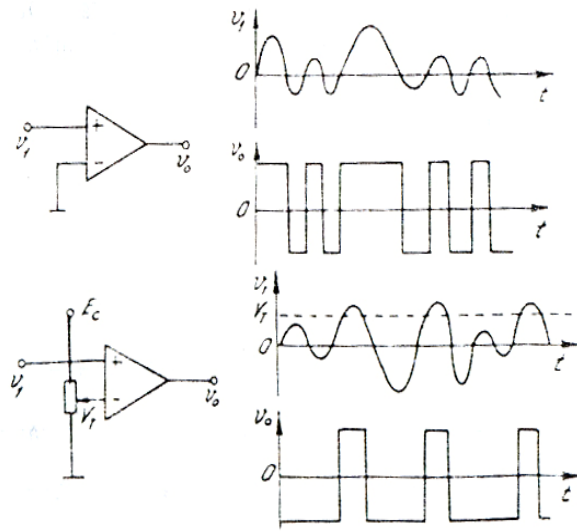
8.1.3. Bộ so sánh và Trigơ Smit

1. Bộ so sánh

Hình bên là bộ so sánh 0, vì $A_0 = \infty$ nên khi $v_+ = v_1 > v_- = 0$ thì $v_0 = +E_c$.
khi $v_+ = v_1 < v_- = 0$ thì $v_0 = -E_c$. Bộ so sánh có thể làm nhiệm vụ giám sát 0.



Hình bên là bộ so sánh ngưỡng V_T . Vì $A_0 = \infty$ nên khi $v_I > v_T$ thì $v_0 = +E_c$, khi $v_I < v_T$ thì $v_0 = -E_c$. Điều chỉnh chiết áp làm thay đổi ngưỡng v_T .



2. Trigon Smit

Hình bên là sơ đồ Trigon Smit dùng bộ khuếch đại thuật toán.

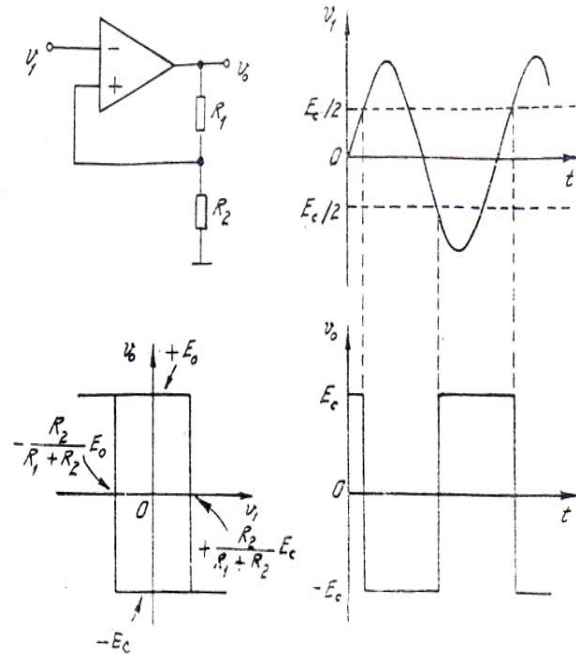
Vì $i_+ = i_- = 0$ nên :

$$v_+ = \frac{R_2}{R_1 + R_2} v_0 \quad (\text{IV-12})$$

Khi v_I tương đối âm, bộ khuếch đại công tác ở vùng bão hòa +, $v_0 = +E_c$.

Vậy mức ngưỡng trên là :

$$V_{T+} = +\frac{R_2}{R_1 + R_2} E_c, v_0 = +E_c \text{ khi } v_I < V_{T+}$$



Khi v_I tăng đến ngưỡng V_{T+} thì v_O đột biến từ $+E_c$ sang $-E_c$, bộ khuếch đại làm việc ở vùng bão hòa -, tương ứng với mức ngưỡng dưới $V_{T-} = -\frac{R_2}{R_1 + R_2} E_c$, tiếp theo nếu $v_I > V_{T-}$ thì $v_O = -E_c$.

Đến khi ngưỡng nào v_I giảm đến ngưỡng dưới V_{T-} thì lại đột biến mới. Hiệu các điện áp ngưỡng

$$\Delta V = V_{T+} - V_{T-} = \frac{R_2}{R_1 + R_2} E_c - \left(-\frac{R_2}{R_1 + R_2} E_c \right) = \frac{2R_2}{R_1 + R_2} E_c \quad (\text{IV - 13})$$

Thay đổi giá trị R_1, R_2 thì điều chỉnh được ΔV .

Nếu $R_1 = R_2$ thì $\Delta V = E_c$ (IV - 14)

Sự phân tích phạm vi công tác của Trigơ Smit, tóm lại như sau :

Trạng thái v_O	Phạm vi v_I
$+E_c$	$v_I < \frac{R_2}{R_1 + R_2} E_c$
$-E_c$	$v_I > -\frac{R_2}{R_1 + R_2} E_c$

So với Trigơ Smit đã giới thiệu ở chương 7, Trigơ Smit cấu trúc từ khuếch đại thuật toán càng ưu việt, vì vậy nó được ứng dụng rất rộng rãi.
