CHƯƠNG 1: CƠ SỞ LÝ THUYẾT ĐIỀU KHIỂN LOGIC

1.1. KHÁI NIỆM LOGIC HAI TRẠNG THÁI

Trong cuộc sống hàng ngày, các sự vật hiện tượng thường biểu hiện ở hai mặt đối lập thông qua hai trạng thái đối lập rõ rệt của nó mà con người dễ dàng nhận thức được sự vật và hiện tượng bằng cách phân biệt hai trạng thái đó. Ví dụ như khi nói về giá cả và chất lượng hàng hoá ta thường có khái niệm đắt và rẻ hay tốt và xấu…

Trong kỹ thuật, đặc biệt trong kỹ thuật điện và điều khiển, ta thường có khái niệm về hai trạng thái: đóng và cắt, kín hay hở, làm việc hay không làm việc, có điện hay mất điện, …

Trong toán học, để lượng hoá hai trạng thái đối lập của sự vật hay hiện tượng người ta dùng hai giá trị: 0 và 1. Giá trị 0 hàm ý đặc trưng cho một trạng thái của sự vật hoặc hiện tượng thì giá trị 1 hàm ý đặc trưng cho trạng thái đối lập của sự vật hoặc hiện tượng đó. Ta gọi đó là các giá trị 0 và 1 logic.

Các nhà bác học đã xây dựng các cơ sở toán học để tính toán các hàm và biến chỉ lấy với hai giá trị 0 và 1 này, hàm và biến đó được gọi là hàm và biến logic, cơ sở toán học để tính toán các hàm và biến đó gọi là đại số logic. Đại số logic cũng có tên là đại số Boole vì lấy theo tên nhà toán học Boole, người có công đầu trong việc xây dựng nên công cụ đại số logic.

1.2. CÁC HÀM LOGIC CƠ BẢN VÀ TÍNH CHẤT

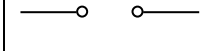
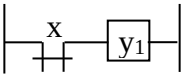
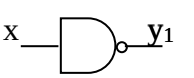
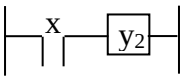
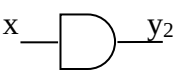
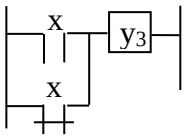
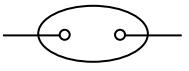
1.2.1. Hàm logic cơ bản

Một hàm $y = f(x_1, x_2, \ldots, x_n)$ với các biến $x_1, x_2, \ldots, x_n$ chỉ nhận hai giá trị: 0 hoặc 1 và hàm y cũng chỉ nhận hai giá trị : 0 hoặc 1, thì $x_1, x_2, \ldots, x_n$ được gọi là các biến logic và y là hàm logic.

- Hàm logic một biến: y = f(x)

Vì biến x sẽ nhận một trong hai giá trị 0 hoặc 1, nên hàm y có 4 khả năng hay thường gọi là 4 hàm y_0, y_1, y_2, y_3. Các khả năng và các ký hiệu mạch rơle và điện tử của hàm một biến cho trong bảng 1.1. Trong đó hai hàm y_0 và y_3 có giá trị luôn không đổi nên ta ít quan tâm, thường chỉ xét đến y_1 và y_2.

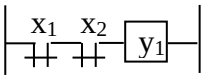
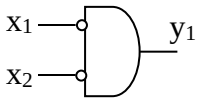
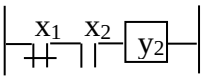
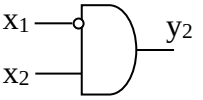
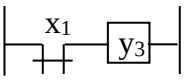
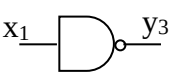
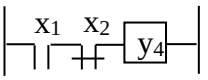
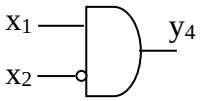
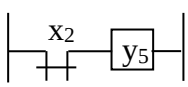
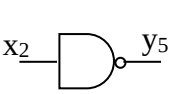
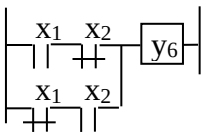
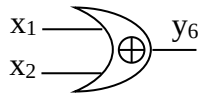
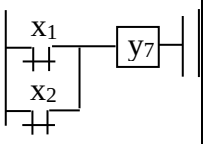
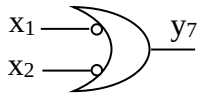
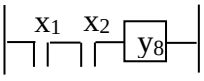
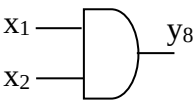
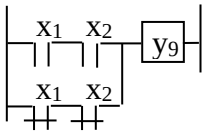
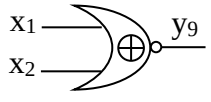
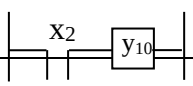
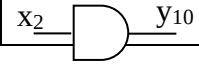
Bảng 1.1: *Hàm logic một biến y = f(x)*

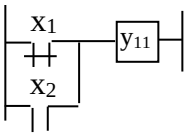
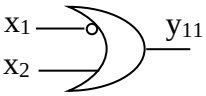
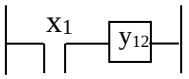
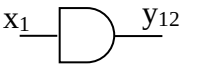
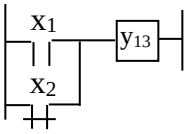
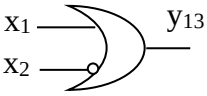
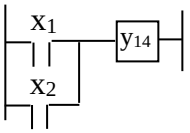
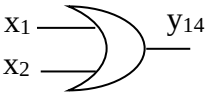
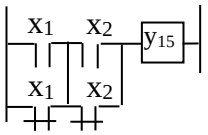
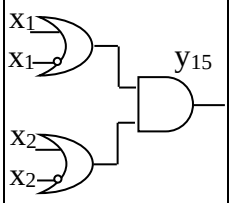
Tên hàm	Bảng chân lý			Thuật toán logic	Ký hiệu sơ đồ		Ghi chú
	x	0	1		Mạch rơle	Khối điện tử	
Hàm không	y_0	0	0	$y_0 = 0$ $y_0 = x\bar{x}$			Hàm luôn bằng không
Hàm đảo	y_1	1	0	$y_1 = \bar{x}$			
Hàm lặp	y_2	0	1	$y_2 = x$			
Hàm đơn vị	y_3	1	1	$y_3 = x + \bar{x}$			Hàm luôn bằng 1

- Hàm logic hai biến $y = f(x_1, x_2)$

Với hai biến logic x_1, x_2 mà mỗi biến có thể nhận giá trị 0 hoặc 1 ta có 16 tổ hợp logic tạo thành 16 hàm được biểu diễn ở bảng 1.2

Bảng 1.2: *Hàm logic hai biến $y = f(x_1, x_2)$*

Tên hàm	Bảng chân lý					Thuật toán logic	Ký hiệu sơ đồ		Ghi chú
	x_1	1	1	0	0		Mạch rơle	Khối điện tử	
	x_2	1	0	1	0				
Hàm không	y_0	0	0	0	0	$y_0 = x_1\bar{x}_1 + x_2\bar{x}_2$			Hàm luôn có giá trị bằng 0
Hàm Piec	y_1	0	0	0	1	$y_1 = \bar{x}_1 . \bar{x}_2$ $y_1 = \overline{x_1 + x_2}$			
Hàm cấm x_1	y_2	0	0	1	0	$y_2 = \bar{x}_1 x_2$			
Hàm đảo x_1	y_3	0	0	1	1	$y_3 = \bar{x}_1$			Chỉ phụ thuộc vào x_1
Hàm cấm x_2	y_4	0	1	0	0	$y_4 = x_1\bar{x}_2$			
Hàm đảo x_2	y_5	0	1	0	1	$y_0 = \bar{x}_2$			Chỉ phụ thuộc vào x_2
Hàm hoặc loại trừ	y_6	0	1	1	0	$y_6 = x_1\bar{x}_2 + \bar{x}_1 x_2$			Cộng modul
Hàm Cheffer	y_7	0	1	1	1	$y_7 = \bar{x}_1 + \bar{x}_2 = \overline{x_1 x_2}$			
Hàm Và	y_8	1	0	0	0	$y_8 = x_1.x_2$			
Hàm cùng dấu	y_9	1	0	0	1	$y_9 = x_1x_2 + \bar{x}_1\bar{x}_2$			
Hàm lặp	y_{10}	1	0	1	0	$y_{10} = x_2$			Chỉ phụ thuộc x_2

theo x_2									
Hàm kéo theo x_2	y_{11}	1	0	1	1	$y_{11} = \overline{x_1} + x_2$			
Hàm lặp theo x_1	y_{12}	1	0	1	0	$y_{12} = x_1$			Chỉ phụ thuộc x_1
Hàm kéo theo x_1	y_{13}	1	1	0	1	$y_{13} = x_1 + \overline{x_2}$			
Hàm hoặc	y_{14}	1	1	1	0	$y_{14} = x_1 + x_2$			
Hàm đơn vị	y_{15}	1	1	1	1				Hàm luôn bằng 1

Ta có nhận xét: Các hàm đối xứng qua trục nằm giữa y_7 và y_8, nghĩa là $y_0 = \overline{y_{15}}$, $y_1 = \overline{y_{14}}$,…

- Hàm logic n biến $y = f(x_1, x_2, \ldots, x_n)$

Với hàm logic n biến, mỗi biến nhận một trong hai giá trị 0 hoặc 1 nên ta có 2^n tổ hợp biến, mỗi tổ hợp biến lại nhận hai giá trị 0 hoặc 1, do vậy số hàm logic là 2^{2^n}. Với số biến bằng n = 1 ta có 4 khả năng tạo hàm, n = 2 có 16 còn với n = 3 sẽ có 256 khả năng tạo hàm, như vậy khi số biến nhiều thì số hàm có khả năng tạo thành rất lớn. Tuy nhiên tất cả các khả năng này đều được biểu hiện qua các khả năng tổng logic, tích logic và nghịch đảo logic của các biến.

Trong tất cả các hàm được tạo thành, ta đặc biệt chú ý đến loại hàm tổng chuẩn và hàm tích chuẩn. Hàm tổng chuẩn là hàm chứa tổng các tích mà mỗi tích có đủ tất cả các biến của hàm. Hàm tích chuẩn là hàm chứa tích các tổng mà mỗi tổng đều có đủ tất cả các biến của hàm.

1.2.2. Các tính chất và một số hệ thức cơ bản của đại số logic

Các tính chất của đại số logic được thể hiện ở 4 luật cơ bản là: luật hoán vị, luật kết hợp, luật phân phối và luật nghịch đảo (định lý De Morgan).

Các luật và một số hệ thức cơ bản là:

1. Định luật giao hoán đối với cộng và nhân logic

$$a + b = b + a$$

$$a \cdot b = b \cdot a$$

2. Định luật kết hợp đối với cộng và nhân logic

$$a + (b + c) = (a + b) + c$$
$$a . (b . c) = (a . b) . c$$

3. Định luật phân phối

$$a . (b + c) = a . b + a . c$$
$$a + b . c = (a + b) . (a + c)$$

4. Định luật nghịch đảo (De - Morgan)

$$\overline{a + b} = \bar{a} . \bar{b}$$
$$\overline{a.b} = \bar{a} + \bar{b}$$

5. Định luật phủ định hai lần

$$\bar{\bar{a}} = a$$

6. Định luật hấp thụ

$$a . (a + b) = a$$
$$a . (a + b) . (a + c) . ... . (a + z) = a$$

7. Luật dính

$$a . b + a . \bar{b} = a$$
$$(a + b) . (a + \bar{b}) = a$$

8. Qui tắc tính đối với các hằng số 0 và 1

$$\bar{0} = 1 \qquad \bar{1} = 0$$
$$a . 1 = a \qquad a . 0 = 0$$
$$a + 0 = a \qquad a + 1 = 1$$

9. Qui tắc tính đối với biến và phủ định của nó

$$a + \bar{a} = 1$$
$$a . \bar{a} = 0$$

10. Luật đồng nhất

$$a + a = a$$
$$a . a = a$$

11. Kết hợp luật phân phối và phép biến đổi đồng nhất

$$a + \bar{a} . b = a + b$$

Ta có thể kiểm tra định luật phân phối (3) bằng bảng sau (bảng 1.3):

Bảng 1.3: *Kiểm nghiệm luật phân phối*

a	b	c	a . (b + c)	a . b + a . c		a + b . c	(a + b) . (a + c)
0	0	0	0	0		0	0
0	0	1	0	0		0	0
0	1	0	0	0		0	0
0	1	1	0	0		1	1
1	0	0	0	0		1	1
1	0	1	1	1		1	1
1	1	0	1	1		1	1
1	1	1	1	1		1	1

1.3. CÁC PHƯƠNG PHÁP BIỂU DIỄN HÀM LOGIC

1.3.1. Phương pháp biểu diễn thành bảng

Với phương pháp này, các giá trị của hàm logic phụ thuộc vào các biến được biểu diễn thành một bảng. Nếu hàm có n biến thì bảng có n+1 cột (n cột cho biến và một cột cho hàm) và 2^n hàng tương ứng với 2^n tổ hợp của biến. Bảng này thường gọi là bảng chân lý.

Ví dụ: Cho một hàm 3 biến với giá trị hàm đã cho được biểu diễn thành bảng như bảng 1.4:

Bảng 1.4:

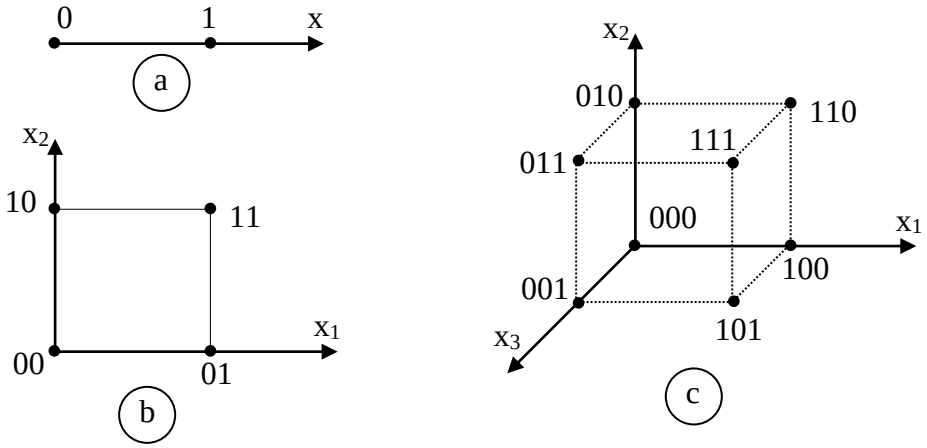
Giá trị thập phân (nhị phân) của tổ hợp biến	x_1	x_2	x_3	y
0 (000)	0	0	0	1
1 (001)	0	0	1	0
2 (010)	0	1	0	"x"
3 (011)	0	1	1	"x"
4 (100)	1	0	0	0
5 (101)	1	0	1	1
6 (110)	1	1	0	"x"
7 (111)	1	1	1	1

Ghi chú: Những chỗ đánh dấu "x" là những tổ hợp biến mà giá trị hàm không xác định (có thể là 0 hoặc 1)

Ưu điểm của phương này là dễ nhìn, ít nhầm lẫn. Nhược điểm là cồng kềnh, đặc biệt là khi số biến lớn.

1.3.2. Phương pháp hình học

Trong phương pháp biểu diễn này, miền xác định của hàm được biểu diễn trong không gian n chiều. Mỗi tổ hợp biến được biểu diễn bằng một điểm trong không gian đó. Hàm n biến tương ứng với không gian n chiều có 2^n điểm trong không gian đó, ứng với mỗi điểm sẽ có một giá trị của hàm. Hai điểm nằm trên cùng một trục chỉ khác nhau bởi sự thay đổi giá trị của một biến. Hình 1.1 là cách biểu diễn hàm logic 1, 2 và 3 biến.



Hình1.1: Biểu diễn hình học hàm logic
a - Hàm 1 biến; b - Hàm 2 biến; c - Hàm 3 biến

Nhược điểm của phương pháp này là khi số biến lớn sẽ rất phức tạp.

1.3.3. Phương pháp biểu thức đại số *(phương pháp giải tích)*

Người ta đã chứng minh được rằng, một hàm logic n biến bất kỳ bao giờ cũng có thể biểu diễn thành các hàm tổng chuẩn đầy đủ và tích chuẩn đầy đủ.

Cách viết hàm dưới dạng tổng chuẩn đầy đủ

- Chỉ quan tâm đến tổ hợp biến mà hàm có giá trị bằng 1. Số lần hàm bằng 1 sẽ chính là số tích của các tổ hợp biến, mỗi tích được gọi là một mintec, ký hiệu là m_i.
- Trong mỗi tích, các biến có giá trị bằng 1 được giữ nguyên, còn các biến có giá trị bằng 0 thì được lấy giá trị nghịch đảo.
- Hàm tổng chuẩn đầy đủ là tổng các tích đó.

Cách viết hàm dưới dạng tích chuẩn đầy đủ

- Chỉ quan tâm đến tổ hợp biến mà hàm có giá trị bằng 0. Số lần hàm bằng 0 sẽ chính là số tổng của các tổ hợp biến, mỗi tổng được gọi là một Maxtec, ký hiệu là M_i.
- Trong mỗi tổng, các biến có giá trị bằng 0 được giữ nguyên, còn các biến có giá trị bằng 1 thì được lấy giá trị nghịch đảo.
- Hàm tích chuẩn đầy đủ là tích các tổng đó.

Ví dụ: Xét hàm được biểu diễn ở bảng 1.4

- Dạng tổng chuẩn đầy đủ: Từ bảng 1.4, ta thấy có 3 tổ hợp biến hàm có giá trị bằng 1 đó là các tổ hợp 0, 5, 7. Hàm có dạng:

$$y = m_0+m_5+m_7 = \bar{x}_1.\bar{x}_2.\bar{x}_3 + x_1.\bar{x}_2.x_3 + x_1.x_2.x_3$$

- Dạng tích chuẩn đầy đủ: Từ bảng 1.4, ta thấy có 2 tổ hợp biến hàm có giá trị bằng 0 đó là các tổ hợp 1, và 4. Hàm có dạng:

$$y = M_1.M_4 = (x_1 + x_2 + \bar{x}_3).(\bar{x}_1 + x_2 + x_3)$$

Phương pháp này có ưu điểm là ngắn gọn, dễ sử dụng các luật của đại số logic để rút gọn hàm. Người ta có thể biểu diễn gọn các hàm trên:

Dạng tổng chuẩn đầy đủ:

$$y = \Sigma 0,5,7 \quad \text{với N} = 2,3,6$$

Dạng tích chuẩn đầy đủ:

$$y = \Pi 1,4 \quad \text{với N} = 2,3,6$$

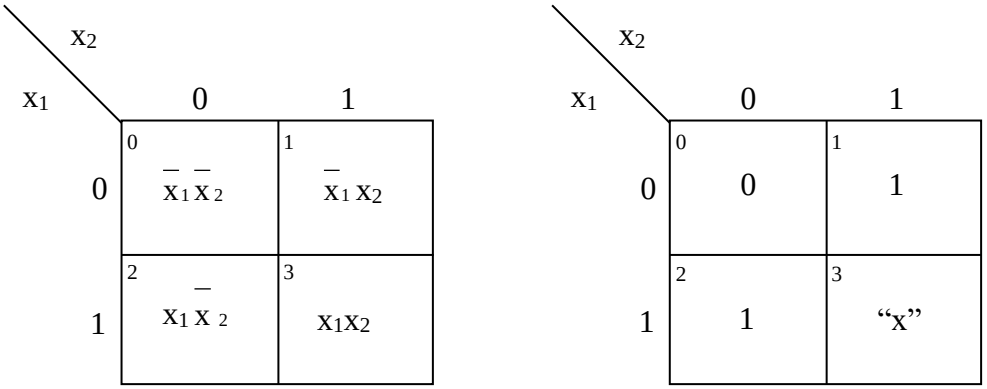
trong đó: N = 2,3,6 là các thứ tự tổ hợp biến mà hàm không xác định.

1.3.4. Phương pháp biểu diễn hàm logic bằng bảng Karnaugh *(Các nô)*

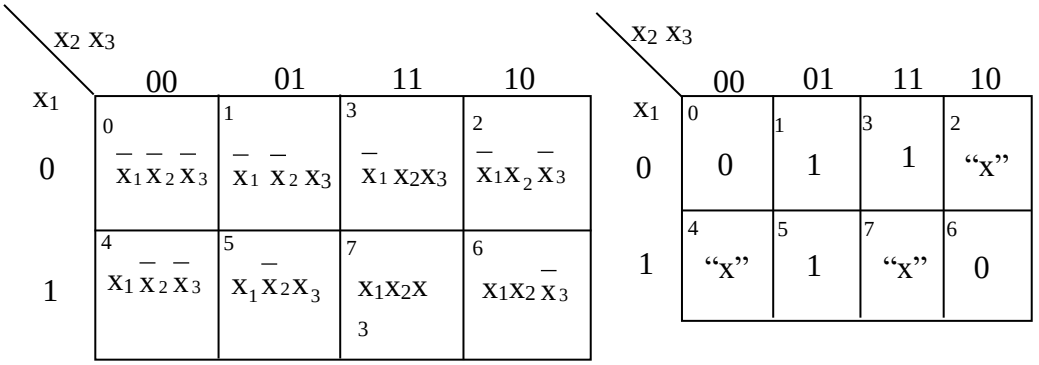
Nguyên tắc xây dựng bảng Karnaugh là:

- Để biểu diễn một hàm logic n biến, cần thành lập một bảng có 2^n ô; mỗi ô tương ứng với một tổ hợp biến. Đánh số thứ tự các ô trong bảng tương ứng với giá trị của tổ hợp biến.
- Các ô cạnh nhau hoặc đối xứng nhau chỉ cho phép khác nhau về giá trị của một biến.
- Trong các ô ghi giá trị của hàm tương ứng với giá trị của tổ hợp biến đó.

Ví dụ: Hình 1.2 là bảng Karnaugh của hàm 2 biến



Hình 1.2: Bảng Karnaugh cho hàm 2 biến; Ví dụ: y=Σ1,2 và N=3



Hình 1.3: Bảng Karnaugh cho hàm 3 biến; Ví dụ: y =Σ1,3,5 với N=2,4,7

x4 x1 x2 \ x3	00	01	11	10
00	$x_1+x_2+x_3+x_4$	$x_1+x_2+x_3+\bar{x}_4$	$x_1+x_2+\bar{x}_3+\bar{x}_4$	$x_1+x_2+\bar{x}_3+\bar{x}_4$
01	$x_1+\bar{x}_2+x_3+x_4$	$x_1+\bar{x}_2+x_3+\bar{x}_4$	$x_1+\bar{x}_2+\bar{x}_3+\bar{x}_4$	$x_1+\bar{x}_2+\bar{x}_3+x_4$
11	$\bar{x}_1+\bar{x}_2+x_3+x_4$	$\bar{x}_1+\bar{x}_2+x_3+\bar{x}_4$	$\bar{x}_1+\bar{x}_2+\bar{x}_3+\bar{x}_4$	$\bar{x}_1+\bar{x}_2+\bar{x}_3+x_4$
10	$\bar{x}_1+x_2+x_3+x_4$	$\bar{x}_1+x_2+x_3+\bar{x}_4$	$\bar{x}_1+x_2+\bar{x}_3+\bar{x}_4$	$\bar{x}_1+x_2+\bar{x}_3+x_4$

x4 x1 x2 \ x3	00	01	11	10
00		0	x	x
01			0	
11		0	x	
10			x	

Hình 1.4: Bảng Karnaugh của hàm 4 biến. Ví dụ y = Π1,7,13 và N=2,3,11,15

1.4. CÁC PHƯƠNG PHÁP TỐI THIỂU HOÁ HÀM LOGIC

Khi phân tích và tổng hợp mạch logic, ta phải quan tâm đến vấn đề tối thiểu hoá hàm logic để việc thực hiện mạch một cách kinh tế nhưng vẫn đảm bảo được các chức năng các yêu cầu. Thực chất của vấn đề tối thiểu hoá hàm logic là tìm dạng biểu diễn đại số đơn giản nhất của hàm và thường có hai nhóm phương pháp:

- Phương pháp biến đổi đại số
- Phương pháp dùng thuật toán

1.4.1. Phương pháp tối thiểu hoá hàm logic bằng biến đổi đại số

Việc rút gọn hàm thường dựa vào các luật và các hệ thức cơ bản của đại số logic

Ví dụ: Tối thiểu hoá hàm sau:

$y = \bar{a}.b + a.b + a.\bar{b} = (\bar{a}.b + a.b) + (a.b + a.\bar{b}) = b(\bar{a} + a) + a(b + \bar{b}) = a + b$

Do tính trực quan của phương pháp nên nhiều khi kết quả đưa ra vẫn không biết rõ là đã tối thiểu hay chưa, như vậy đây không phải là phương pháp chặt chẽ để cho phép tự động hoá quá trình tối thiểu hoá hàm logic.

1.4.2. Phương pháp tối thiểu hoá hàm logic theo thuật toán

Thường dùng nhất là các phương pháp: bảng Karnaugh và Quine Mc. Cluskey

1) Tối thiểu hoá hàm logic bằng phương pháp Quine Mc. Cluskey

a. Một số khái niệm và định nghĩa

+ Đỉnh: Đỉnh là một tích chứa đầy đủ các biến của hàm xuất phát, nếu hàm có n biến thì đỉnh là tích của n biến.

Đỉnh 1 là đỉnh mà hàm có giá trị bằng 1;

Đỉnh 0 là đỉnh mà hàm có giá trị bằng 0;

Đỉnh không xác định là đỉnh mà tại đó hàm có thể lấy một trong hai giá trị bằng 0 hoặc 1.

Ví dụ: Cho hàm $y = f(x_1, x_2, x_3)$ có L = 2,3,7 và N = 1,6 (L là các thứ tự tổ hợp biến mà hàm có giá trị bằng 1). Các đỉnh này có thể đánh dấu theo số ở hệ thập phân hay có thể theo số nhị phân như ở bảng 1.5.

Bảng 1.5

Tích	$\bar{x}_1.\bar{x}_2.x_3$	$\bar{x}_1.x_2.\bar{x}_3$	$\bar{x}_1.x_2.x_3$	$x_1.x_2.\bar{x}_3$	$x_1.x_2.x_3$
Số nhị phân	001	010	011	110	111
Số thập phân	1(x)	2	3	6(x)	7

+ Tích cực tiểu: Tích cực tiểu là tích có số biến là cực tiểu để hàm có giá trị bằng 1 hoặc có giá trị không xác định.

+ Tích quan trọng: Tích quan trọng là tích cực tiểu mà giá trị hàm chỉ duy nhất bằng 1 ở tích này.

b. Tối thiểu hoá hàm logic bằng phương pháp Quine Mc. Cluskey

Các bước tiến hành:

Quá trình tối thiểu hoá hàm logic bằng phương pháp Quine Mc. Cluskey được tiến hành theo các bước như trên hình 1.5.

Ví dụ minh hoạ: Cho hàm $y = f(x_1, x_2, x_3, x_4)$ với các đỉnh bằng 1 là L = 2, 3, 7, 12, 14, 15; và các đỉnh hàm không xác định là N = 6, 13 (bảng 1.6). Hãy tối thiểu hoá hàm bằng phương pháp Quine Mc. Cluskey

Cách làm:

Bước 1: Tìm các tích cực tiểu

Các công việc tiến hành như sau:

+) Lập bảng biểu diễn các giá trị hàm bằng 1 và các giá trị không xác định ứng với mã nhị phân của các biến (bảng 1.6a).

+) Sắp xếp các tổ hợp biến theo mã nhị phân theo thứ tự số các chữ số 1 tăng dần từ 0, 1, 2, 3, … . Như vậy ở đây ta có 4 tổ hợp: tổ hợp 1 (gồm các số chứa 1 chữ số 1), tổ hợp 2 (gồm các số chứa 2 chữ số 1), tổ hợp 3 (gồm các số chứa 3 chữ số 1), tổ hợp 4 (gồm các số chứa 4 chữ số 1) (bảng 1.6b).

+) So sánh mỗi tổ hợp thứ i với một tổ hợp thứ i+1, nếu hai tổ hợp chỉ khác nhau ở một cột thì kết hợp hai tổ hợp đó thành một tổ hợp mới, đồng thời thay cột số khác nhau của 2 tổ hợp cũ bằng một gạch ngang (-) và đánh dấu V vào hai tổ hợp cũ (bảng 1.6c). Về cơ sở toán học, ở đâyđể thu gọn các tổ hợp ta đã sử dụng tính chất:

$$a.b + a.\bar{b} = a$$

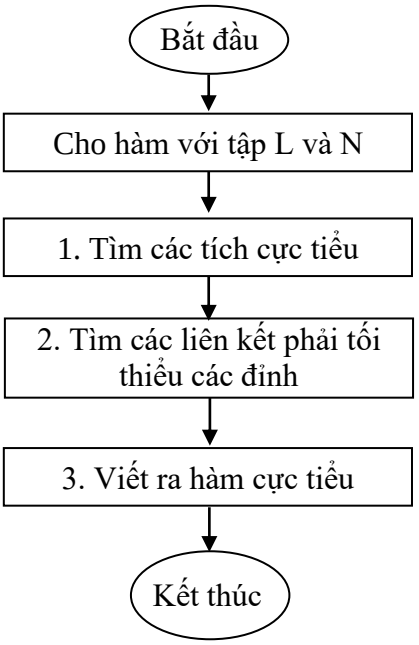
+) Tiếp tục công việc: Từ bảng 1.6c ta chọn ra các tổ hợp chỉ khác nhau 1 chữ số 1 và có cùng gạch ngang (-) trong một cột, nghĩa là có cùng biến vừa được giản ước ở bảng 1.6c, như vậy ta có bảng 1.8d.

Các tổ hợp tìm được ở bảng 1.6d là tổ hợp cuối cùng, không còn khả năng kết hợp nữa, đây chính là các tích cực tiểu của hàm f đã cho và được viết:

0 - 1 - (phủ các đỉnh 2, 3, 6, 7) : $\overline{x_1}\, x_3$

- 1 1 - (phủ các đỉnh 6, 7, 14, 15) : x_2x_3

1 1 - - (phủ các đỉnh 12, 13, 14, 15) : x_1x_2



Hình 1.5

Bảng 1.6

Bảng a		Bảng b			Bảng c		Bảng d	
Số thập phân	Số nhị phân ($x_1x_2x_3x_4$)	Số chữ số 1	Số thập phân	Số nhị phân ($x_1x_2x_3x_4$)	Liên kết	Số nhị phân ($x_1x_2x_3x_4$)	Liên kết	Số nhị phân ($x_1x_2x_3x_4$)
2	0010	1	2	0010V	2,3	001-V	2,3,6,7 2,6,3,7	0-1-
3	0011		3	0011V	2,6	0-10V	6,7,14,15 6,14,7,15	-11-

		2						
6	0110		6	0110V	3,7	0-11V	12,13,14,15	11--
12	1100		12	1100V	6,7	011-V	12,14,13,15	
7	0111		7	0111V	6,14	-110V		
13	1101	3	13	1101V	12,13	110-V		
14	1110		14	1110V	12,14	11-0V		
15	1111	4	15	1111V	7,15	-111V		
					13,15	11-1V		
					14,15	111-V		

Bước 2: Tìm các tích quan trọng

Việc tìm các tích quan trọng cũng được tiến hành theo trình tự nhiều bước nhỏ. Giả thiết có i bước nhỏ, với i = 0, 1, 2, 3, …, k

Gọi L_i là tập các đỉnh 1 đang xét ở bước thứ i, lúc này không quan tâm đến các đỉnh có giá trị không xác định nữa.

Z_i là tập các tích cực tiểu ở bước nhỏ thứ i.

E_i là tập các tích quan trọng ở bước nhỏ thứ i.

Trình tự công việc được tiến hành như sau:

+) Với i = 0

$L_0 = L = (2, 3, 7, 12, 14, 15)$

$Z_0 = Z = (\overline{x_1}\, x_3, x_2x_3, x_1x_2)$

Xác định các tích quan trọng E_0 từ các tập L_0 và Z_0 như sau:

Lập một bảng trong đó mỗi hàng ứng với một tích cực tiểu thuộc Z_0, mỗi cột ứng với một đỉnh thuộc L_0. Đánh dấu "x" vào các ô trong bảng ứng với tích cực tiểu bằng 1.

Xét từng cột , cột nào chỉ có một dấu "x" thì tích cực tiểu ứng với nó là tích quan trọng như ở bảng 1.7.

Bảng 1.7. $E_0 = (\overline{x_1}\, x_3, x_1x_2)$

Z_0 \ L_0	2	3	7	12	14	15
$\overline{x_1}\, x_3$	(x)	(x)	x			
x_2x_3			x		x	x
x_1x_2				(x)	x	x

+) Với i = 1

L_1: Tìm L_1 từ L_0 bằng cách loại khỏi L_0 các đỉnh 1 của E_0.

Z_1: Tìm Z_1 từ Z_0 bằng cách loại khỏi Z_0 các tích trong E_0 và các tích đã nằm trong hàng đã được chọn từ E_0 (đó là các tích không cần thiết).

Lập bảng tương tự như trên, từ bảng đó cũng bằng cách tương tự trên sẽ tìm được tích quan trọng E_1.

Công việc được tiếp tục cho đến khi hết các tích cực tiểu

$L_{i+1} = L_i$ - các đỉnh hàm có giá trị bằng 1 ứng với E_i

$Z_{i+1} = Z_i - E_i$

Lập bảng L_{i+1} và Z_{i+1} để tìm E_{i+1}. Lặp lại công việc cho đến khi $L_k = 0$.

Trong ví dụ trên thì $L_1 = 0$, do vậy ta có hàm đã được tối thiểu hoá là:

$$f = \overline{x}_1 x_3 + x_1 x_2$$

2/ Phương pháp dùng bảng Karnaugh

Phương pháp này được tiến hành theo các bước sau:

Bước 1: Biểu diễn hàm đã cho thành bảng Karnaugh.

Bước 2: Xác định các tích cực tiểu hoặc tổng cực tiểu.

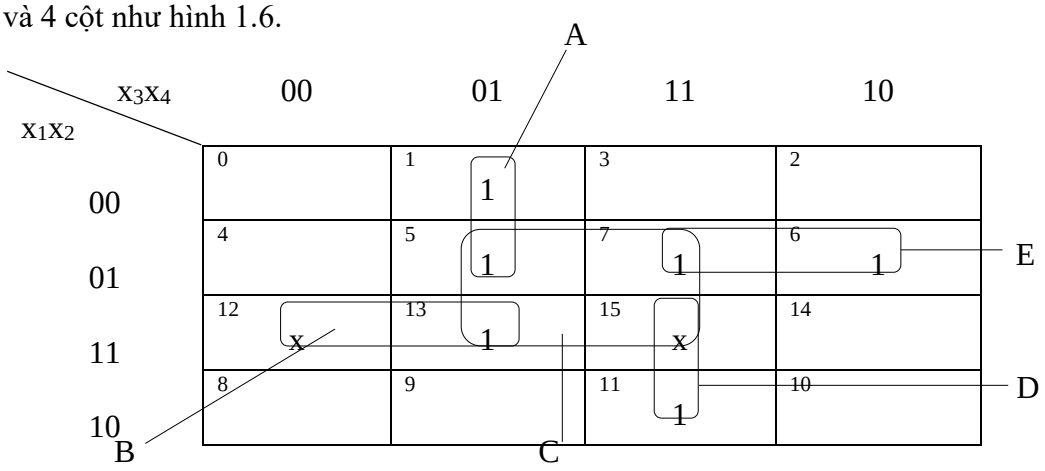
Bước 3: Tìm các liên kết phủ tối thiểu các ô "1" (nếu biểu diễn tối thiểu theo hàm tổng) hoặc các ô "0" (nếu biểu diễn theo hàm tích), sau đó viết hàm kết quả theo tổng hoặc tích.

Ví dụ 1: Hãy tối thiểu hàm logic sau đây theo hàm tổng:

$y = f(x_1, x_2, x_3, x_4) = \Sigma\ 1, 5, 6, 7, 11, 13;$ và $N = 12, 15;$

Giải:

Bước 1: Lập bảng Karnaugh. Vì hàm có 4 biến nên ta có thể lập bảng Karnaugh thành 4 hàng và 4 cột như hình 1.6.



Hình 1.6: Bảng Karnaugh của hàm $y = f(x_1, x_2, x_3, x_4)$

Quan sát bảng Karnaugh và chỉ xét các liên kết tối thiểu phủ hết các ô có kết quả hàm bằng 1 (lúc này không xét các ô có ký hiệu "x"), như vậy ta được kết quả tối thiểu của hàm là:

$$y = A + C + D + E = \overline{x}_1.\overline{x}_3.x_4 + x_2.x_4 + x_1.x_3.x_4 + \overline{x}_1.x_2.x_3$$

CHƯƠNG 2. HỆ ĐIỀU KHIỂN LOGIC

2.1. Hệ điều khiển logic tổ hợp

2.1.1 Khái niệm

Hệ điều khiển logic tổ hợp là hệ mà trạng thái đầu ra chỉ phụ thuộc vào tổ hợp các trạng thái đầu vào chứ không phụ thuộc vào trình tự tác động của các đầu vào. Theo quan điểm điều khiển thì hệ điều khiển logic tổ hợp là hệ hở, hệ không có phản hồi, nghĩa là trạng thái đóng mở của các phần tử trong hệ hoàn toàn không bị ảnh hưởng của trạng thái tín hiệu đầu ra.

Về mặt toán học, giả thiết một hệ điều khiển logic tổ hợp có n đầu vào với các x_i (i = 1-n) và m đầu ra với các y_j (j = 1-m), ta ký hiệu:

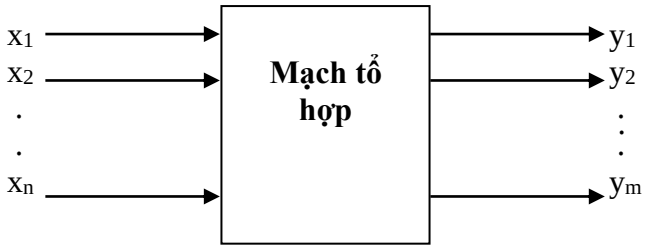
$X = \{ x_1, x_2, \ldots, x_n \}$ là tập các biến vào.

$Y = \{ y_1, y_2, \ldots, y_m \}$ là tập các biến ra.

thì mạch tổ hợp được biểu diễn bởi m phương trình đại số Boole như sau:

$y_j = f_j(x_1, x_2, \ldots, x_n)$ với j = 1-m.

Có thể biểu diễn mô hình toán học của mạch tổ hợp theo sơ đồ khối hình 2.1



Hình 2. 1 Hệ điều khiển logic tổ hợp

2.1.2. Thiết kế hệ logic tổ hợp

Nhiệm vụ chính ở đây là thiết kế được mạch tổ hợp thoả mãn yêu cầu kỹ thuật nhưng mạch phải tối giản. Bài toán thiết kế là bài toán phức tạp, vì ngoài các yêu cầu về chức năng logic, việc thiết kế mạch còn phụ thuộc vào việc sử dụng các phần tử, chẳng hạn như phần tử là loại rơ le-công tắc tơ, là các phần tử bán dẫn hay vi mạch chuẩn, … Với mỗi loại phần tử thì ngoài nguyên lý chung về mạch logic còn đòi hỏi phải bổ sung những nguyên tắc riêng lúc thiết kế hệ thống.

Thiết kế một mạch logic tổ hợp gồm các bước sau :

+ Phân tích yêu cầu công nghệ, xác định biến vào/ra
+ Mô tả hệ thống
+ Tìm hàm logic tối giản
+ Thực hiện mạch logic tổ hợp bằng việc sử dụng các rơ le, công tắc tơ (tổng hợp mạch rơ le), hoặc bằng các phần tử logic AND, OR, NAND, NOR đã chuẩn hoá đầu vào và đầu ra.

Ví dụ: Thiết kế hệ thống tự động cảnh báo trộm cho một ngôi nhà như sau: Khi có sự xâm nhập của kẻ trộm thì chuông cảnh báo và đèn cảnh báo được kích hoạt. Cảnh báo này được kích hoạt nếu kẻ xâm nhập được phát hiện bằng cảm biến gắn ở của sổ và một bộ phát hiện chuyển động. Cảm biến ở cửa sổ là loại thường đóng, khi cửa sổ vỡ do kẻ xâm nhập thì cảm biến bị ngắt. Cảm biến nhận biết chuyển động được thiết kế để khi một người được phát hiện thì ngõ ra sẽ ở mức "1". Ngoài ra còn có một công tắc để kích hoạt/ không kích hoạt cảnh báo.

Bước 1 : Phân tích yêu cầu công nghệ, xác định biến vào/ra

- Chuông và đèn được kích hoạt đồng thời nên chỉ là một đầu ra của hệ, đặt tên đầu ra là y(y=1: chuông kêu và đèn sáng, y=0 chuông không kêu và đèn tắt)
- Cảm biến gắn ở cửa sổ để phát hiện kính vỡ do đột nhập, là một đầu vào của hệ, đặt tên biến vào x_1(x_1=1 khi kính không vỡ, x_1=0 khi kính vỡ)
- Cảm biến phát hiện chuyển động khi đột nhập, là đầu vào thứ hai của hệ, đặt tên biến vào x_2(x_2=1 khi có chuyển động, x_2=0 khi không có chuyển động)
- Công tắc để kích hoạt hệ thống, là đầu vào thứ ba của hệ thống, đặt tên là x_3(x_3=1 khi kích hoạt hệ thống, x_3=0 khi không kích hoạt hệ thống)

Bước 2 : Mô tả hệ thống
- $y=f(x_3,x_2,x_1)$
- Dùng bảng Karnaugh để mô tả hệ

y x_2x_1 / x_3	00	01	11	10
0	0	0	0	0
1	1	0	1	1

Bước 3 : Tìm hàm logic tối giản

Dựa vào bảng trên có thể rút ra hàm tối giản

- $y=x_3x_2+x_3x'_1$

Bước 4 : Thực hiện

- Mạch rơ le
- Mạch logic số
- Lập trình

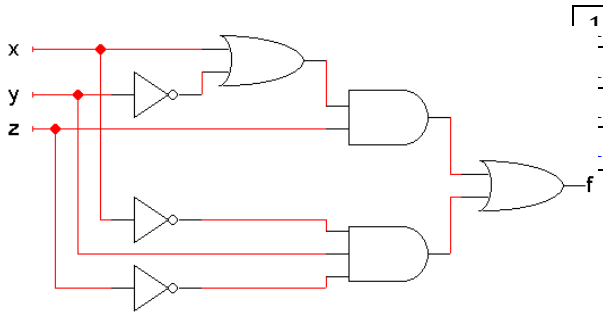
2.1.3. Phân tích hệ logic tổ hợp

- Phân tích là tìm ra nguyên lý hoạt động
– Mỗi một mạch điện thực hiện một hàm logic, mà được mô tả bởi một biểu thức logic hoặc bảng chân lý.
– Vì thế, mục đích là tìm ra biểu thức logic hoặc bảng chân lý cho mạch

Ví dụ

Việc đầu tiên là xác định các đầu vào và đầu ra của mạch: Đầu vào: x, y, z
Đầu ra: f

x	y	z	f
0	0	0	
0	0	1	
0	1	0	
0	1	1	



1	0	0	
	0	1	
	1	0	
	1	1	

Hình 2. 2. Mạch logic tổ hợp

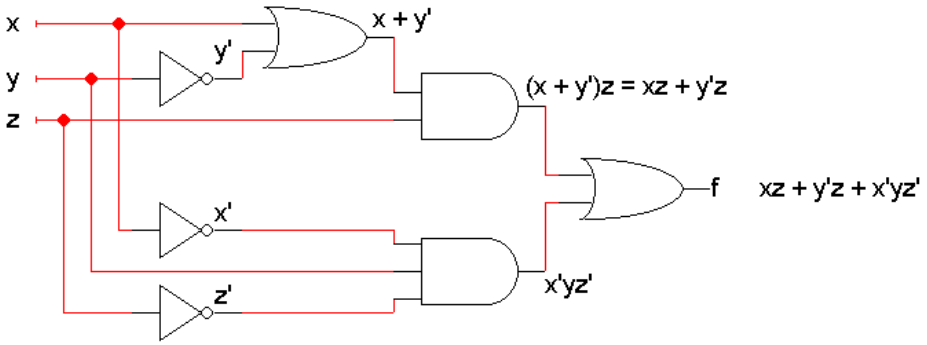
Cách 1:

- Xuất phát từ sơ đồ mạch
- Xác định biểu thức đầu ra của các cổng logic
- Các biểu thức trung gian được tổ hợp theo các cổng logic để tạo ra các biểu thức phức tạp.

Theo cách này có thể thực hiện các phép rút gọn bằng biến đổi đại số

- Lặp lại đến khi thu được hàm và biểu thức đầu ra

- Cách 1 có thể đưa ra cả bảng chân lý và biểu thức



Hình 2. 3 Phân tích mạch logic tổ hợp (cách 1)

Cách 2:

- Phân tích mạch theo cách 2 là quá trình gán từng giá trị cho các đầu vào, và ghi giá trị đầu ra tương ứng.

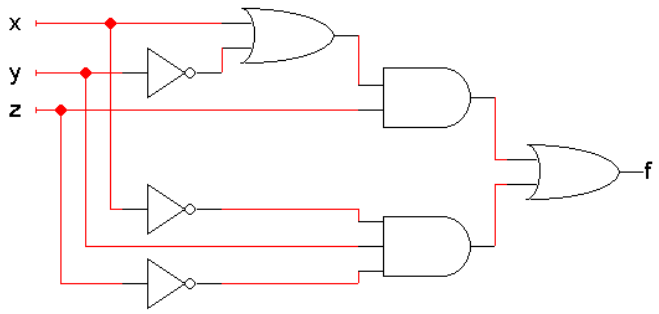
– với 'n' đầu vào có 2^n khả năng tổ hợp các đầu vào

– Từ các giá trị đầu vào, đầu ra các cổng được định giá trị để tạo ra bộ giá trị của đầu vào các cổng tiếp theo.

– Việc định giá trị được tiến hành liên tục đến khi giá trị đầu ra của các cổng chính đầu ra của mạch

- Cách phân tích mạch thứ 2 chỉ cho chúng ta bảng chân lý
- Một khi biết số lượng đầu vào và đầu ra, có thể liệt kê tổ hợp giá trị các đầu vào để đưa vào bảng chân lý

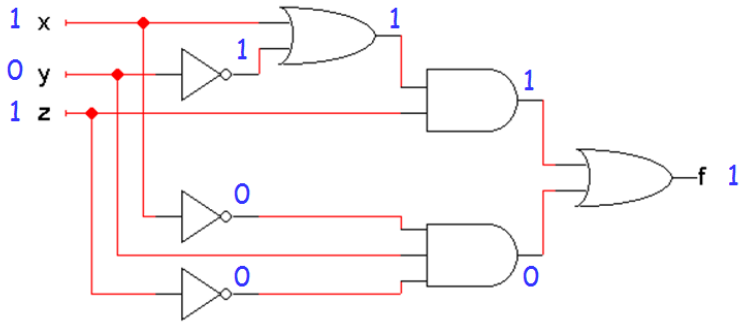
– Một mạch có n đầu vào nên cần bảng chân lý với 2^n dòng



x	y	z	f
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Hình 2. 4 Phân tích mạch logic tổ hợp (cách 2)

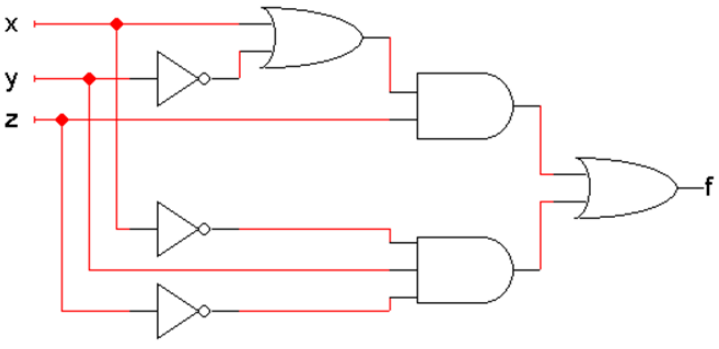
- Có mô phỏng mạch bằng tay để tìm giá trị đầu ra ứng với mỗi tổ hợp tín hiệu vào



x	y	z	f
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	1
1	1	0	
1	1	1	

Hình 2. 5 Mô phỏng mạch Logic tổ hợp với một tổ hợp tín hiệu vào/ra

- Làm tương tự như trên đối với các tổ hợp khác của các đầu vào sẽ có được bảng chân lý đầy đủ



x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Hình 2. 6 Mô phỏng mạch Logic tổ hợp với nhiều tổ hợp tín hiệu vào/ra

Cách 3:

- Nếu có được biểu thức logic, có thể dùng nó để dễ dàng xây dựng được bảng chân lý
- Ví dụ, khi mà chúng ta xác định được rằng mạch điện thực hiện hàm

f(x,y,z) = xz + y'z + x'yz', chúng ta có thể dùng nó để điền vào bảng chân lý:

x	y	z	xz	y'z	x'yz'	f
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	0	0	1	1
0	1	1	0	0	0	0
1	0	0	0	0	0	0
1	0	1	1	1	0	1
1	1	0	0	0	0	0
1	1	1	1	0	0	1

- Ngược lại thì cũng đúng: dễ dàng có được biểu thức nếu như có bảng chân lý
- Chuyển từ bảng chân lý sang dạng tổng các minterm

f(x,y,z) = x'y'z + x'yz' + xy'z + xyz

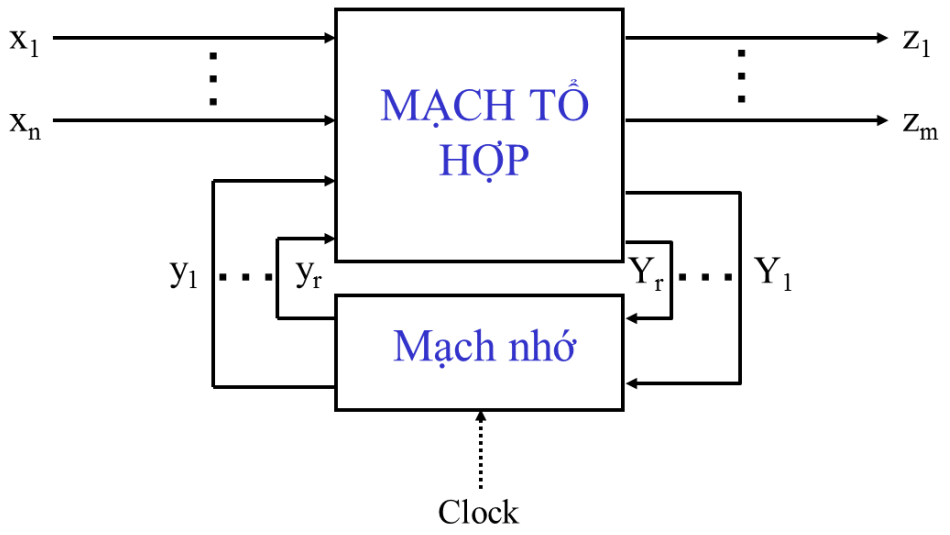
$= m_1 + m_2 + m_5 + m_7$

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

- Sau đó có thể đơn giản biểu thức dạng tổng các minterm nếu muốn — dùng K-map, đối với ví dụ
- Sau khi xác định được các đầu vào/ra , có thể dễ dàng được biểu thức hoặc bảng chân lý để miêu tả hoạt động của mạch
- Có thể dễ dàng chuyển giữa biểu thức và bảng chân lý

2.2. HỆ ĐIỀU KHIỂN LOGIC TRÌNH TỰ

2.2.1. Khái niệm hệ điều khiển logic trình tự



Hình 2. 7. Hệ điều khiển logic trình tự

Mạch trình tự hay mạch dãy (sequential circuits) là mạch mà trong đó trạng thái của đầu ra (tín hiệu ra) không những phụ thuộc tín hiệu vào mà còn phụ thuộc cả vào trình tự tác động của tín hiệu vào, nghĩa là có nhớ các trạng thái. Như vậy, về mặt thiết bị thì ở mạch trình tự không chỉ có các phần tử đóng mở mà còn có cả các phần tử nhớ.

Sơ đồ cấu trúc cơ bản của mạch trình tự như hình 3.1. Điểm đặc biệt ở đây là mạch có "phản hồi" thể hiện qua các biến trung gian (Y_1, Y_2 và y_1, y_2).

Hoạt động của mạch trình tự được thể hiện ở sự thay đổi của biến trung gian Y. Trong quá trình làm việc, do sự thay đổi của các biến vào X (x_1, x_2,…) sẽ dẫn đến thay đổi các biến ra Z (Z_1, Z_2, …) và cả biến trung gian Y(Y_1, Y_2,…). Sự thay đổi của Y(Y_1, Y_2,…) sẽ dẫn đến sự thay đổi biến y (y_1, y_2, …). Sự thay của các biến y (y_1, y_2, …) lại có thể dẫn đến sự thay đổi của các tín hiệu ra Z , kể cả Y, rồi sự thay đổi của Y lại dẫn đến sự thay đổi của y, …

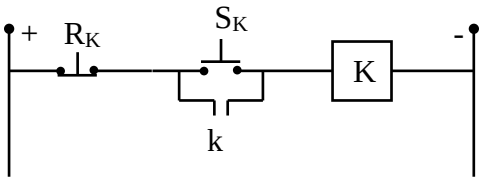
Mạch trình tự được chia thành 2 loại theo loại phần tử nhớ được sử dụng trong mạch: mạch trình tự đồng bộ và mạch trình tự không đồng bộ

2.2.2. Một số phần tử nhớ trong hệ điều khiển logic trình tự

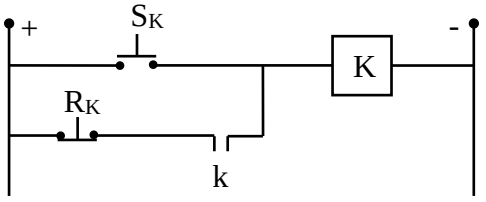
Như đã nói ở trên, tính đặc thù của mạch trình tự là có mạch điện tự duy trì hoặc phần tử nhớ, do vậy trong mục này sẽ giới thiệu tóm tắt một số phần tử nhớ.

a. Mạch duy trì

Mạch duy trì ưu tiên reset và ưu tiên set. K là cuộn dậy rơ le hoặc công tắc tơ, k là tiếp điểm.



Hình 2. 8 Mạch nhớ rơ le ưu tiên Reset



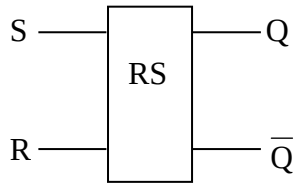
Hình 2. 9 Mạch nhớ rơ le ưu tiên Set

b. Các mạch lật

Các mạch lật FF (Flip-Flop) là các phần tử có khả năng nhớ một trong hai trạng thái: 0 hoặc 1. Để xây dựng các mạch số trình tự, ngoài các phần tử AND, OR, NAND, NOR, … thì còn cần phải có các phần tử nhớ là các mạch lật. Có nhiều loại mạch lật khác nhau, sau đây ta sẽ xét một loại mạch lật được sử dụng nhiều trong điều khiển logic trình tự.

Mạch lật RS có hai đầu vào điều khiển là S và R, có hai đầu ra là Q và $\overline{Q}$. Sơ đồ bố trí chân đầy đủ và bảng chân lý của mạch lật RS cho ở hình 3.4.

- Bảng đặc trưng



Q' \ RS	00	01	11	10
0	(0)	1	x	(0)
1	(1)	(1)	x	0

TT	S	R	Q'
1	0	0	Q
2	0	1	0
3	1	0	1
4	1	1	x

- Phương trình đặc trưng

Từ bảng Karnaugh ta có phương trình đặc trưng cho mạch RS:

$$Q(t+1)=S+\overline{R}Q(t)$$

Bảng đặc trưng và phương trình đặc trương được sử dụng trong phân tích hệ điều khiển logic trình tự.

- Bảng kích thích

Q(t)	Q(t+1)	S	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	x	0

Bảng kích thích được sử dụng để thiết kế hệ điều khiển logic trình tự.

3.3. Các phương pháp mô tả mạch logic trình tự

3.3.1. Phương pháp bảng chuyển trạng thái

Hoạt động của mạch trình tự có thể được mô tả bằng một bảng chuyển trạng thái. Đó là một bảng có nhiều hàng tương ứng với các trạng thái trong và nhiều cột tương ứng với các tổ hợp tín hiệu vào. Trong mỗi ô vuông ta đưa vào trạng thái đích của mạch, nghĩa là trạng thái tiếp theo mà mạch sẽ đạt tới khi nó ở trạng thái trong cho bởi hàng và vào cho bởi cột

Ví dụ : Một hệ thống bơm nước hoạt động như sau: bơm nước được mở khi nước ở dưới mức 1và vẫn mở cho đến khi nước tới mức 2, tại đó bơm đóng lại. Có 2 tín hiệu để báo mức nước là h và l

l=1 khi mực nước bằng hoặc trên mức 1

l=0 ở trường hợp khác

h=1 khi mực nước bằng hoặc trên mức 2

h=0 ở các trường hợp khác

Hệ có 4 trạng thái trong: S0 là trạng thái khi mức nưới dưới mức 1, p=1. Khi nước trên mức 1 hệ chuyển sang trạng thái S1, p=1. Khi mức nước bằng hoặc trên mức 2 thì hệ chuyển sang trạng thái S2, p=0. Khi mức nước giảm xuống dưới mức 2 thì hệ chuyển sang trạng thái S3, p=0. Ta xây dựng được như sau:

S' \ lh	00	01	11	10
S_0	(S0) P=1	X	S_2 P=x	S_1 P=x
S_1	S_0 P=x	X	S_2 P=x	(S1) P=1

(S2)

(S3)

S_2	S_0 P=x	X	 P=0	S_3 P=x
S_3	S_0 P=x	X	S_2 P=x	 P=0

Rút gọn bảng chuyển trạng thái

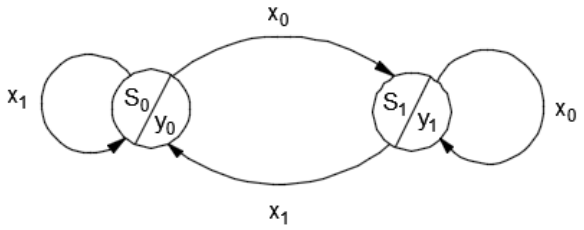
S' lh	00	01	11	10
S_{01}	(S_{01}) P=1	X	S_{23} P=x	(S_{01}) P=1
S_{23}	S_{01} P=x	X	(S_{23}) P=0	(S_{23}) P=0

3.3.2. Phương pháp đồ hình trạng thái

Đồ hình trạng thái là hình vẽ mô tả các trạng thái chuyển của một hệ logic trình tự, đồ hình gồm các đỉnh và các cung định hướng biểu thị sự chuyển biến trạng thái dưới sự tác động của biến vào. Có hai loại: đồ hình Moore và đồ hình Mealy.

- Đồ hình Moore

Đồ hình Moore gồm các đỉnh biểu diễn trạng thái và biến ra của hệ, còn các cung định hướng sẽ ghi biến vào (biến tác động).

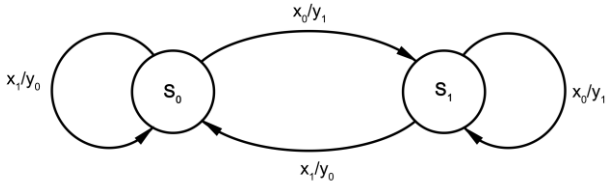


Hình 2. 10. Đồ hình Moore

Hệ logic trình tự có 2 trạng thái S_0, S_1 và 2 biến ra y_0, y_1. Biến ra y_0 chỉ phụ thuộc vào trạng trạng thái S_0, biến ra ra y_1 chỉ phụ thuộc vào trạng thái S_1. Sự tác động của biến vào x_0, hệ chuyển từ trạng thái S0 sang trạng thái S1. Sự tác động của biến vào x_1 hệ chuyển từ trạng thái S1 sang trạng thái S0.

- Đồ hình Mealy

Đồ hình Mealy gồm các đỉnh biểu diễn các trạng thái của hệ và các cung định hướng, trên các cung ghi biến vào và biến ra của hệ khi chịu sự tác động của biến vào.



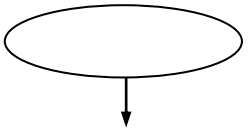
Hình 2. 11. Đồ hình Mealy

Hệ logic trình tự có 2 trạng thái S_0, S_1 và 2 biến ra y_0, y_1. Biến ra y_0 phụ thuộc vào trạng trạng thái S_1 và biến vào x_1, biến ra ra y_1 phụ thuộc vào trạng thái S_0 và biến vào x_0. Sự tác động của biến vào x_0, hệ chuyển từ trạng thái S0 sang trạng thái S1. Sự tác động của biến vào x_1 hệ chuyển từ trạng thái S1 sang trạng thái S0.

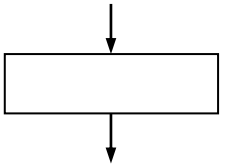
3.3.3. Phương pháp lưu đồ

Lưu đồ là cách mô tả hệ thống một cách suy luận trực quan. Các khối chính của lưu đồ :

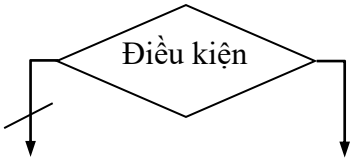
- Khối bắt đầu hoặc kết thúc



- Khối thực hiện



- Khối điều kiện



3.3.4. Phương pháp dùng GRAFCET

GRAFCET là từ viết tắt của tiếng Pháp "Graphe fonctionnel de commande étape transition", là một đồ hình chức năng mô tả các trạng thái làm việc của hệ thống và biểu diễn quá trình điều khiển với các trạng thái chuyển biến từ trạng thái này sang trạng thái khác, đó là một graphe định hướng và được xác định bởi các phần tử sau:

G = { E, T ,A, M}.

Trong đó:

- E = {E_1,E_2,…E_m} là một tập hữu hạn các trạng thái (giai đoạn) của hệ thống, được ký hiệu bằng các hình vuông. Mỗi trạng thái ứng với những tác động nào đó của phần điều khiển và trong một trạng thái các hành vi điều khiển là không thay đổi. Một trạng thái có thể là hoạt động hay không hoạt động. Điều khiển chính là thực hiện các mệnh đề logic chứa các biến vào và các biến ra để hệ thống có một trạng thái xác định trong hệ và đó cũng là một trạng thái của một grafcet. Ví dụ trạng thái E_j như ở hình 3.22 là sự phối hợp của biến P và M với M=E_k.a, trong đó E_k là biến đặc trưng cho sự hoạt động của trạng thái E_k, còn a là biến đầu vào của hệ.

- T = {t_1, t_2…t_i } là tập hữu hạn các chuyển trạng thái được biểu hiện bằng gạch ngang "- ". Hàm Boole gắn với một chuyển trạng thái được gọi là "một tiếp nhận". Giữa hai trạng thái luôn luôn tồn tại một chuyển trạng thái. Chuyển trạng thái t_j ở hình 3.23 được thực hiện bởi tích logic $E_v.a.\bar{c}$, trong đó E_v là biến đặc trưng cho sự hoạt động của trạng thái E_v, còn a, v là biến đầu vào. việc chấp nhận chuyển t_j là $t_j = E_v.a.\bar{c}$.

Chuyển trạng thái t_j ở hình 3.24 được thực hiện bởi điều kiện logic E_k(↑a), trong đó E_k là biến biểu diễn hoạt động của trạng thái E_k, còn ↑a biểu diễn sự thay đổi từ 0→1 của biến vào a.

- A ={a_1,a_2…a_n} là tập các cung định hướng nối giữa một trạng thái với một chuyển hoặc giữa một chuyển với một trạng thái.

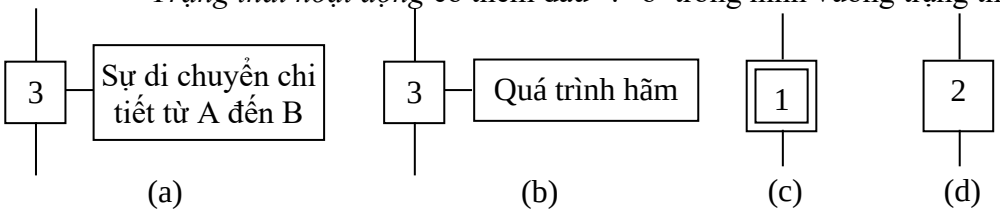
- M = {m_1,m_2…m_m} là tập các giá trị (0, 1). Nếu m_i =1 là hoạt động, m_i = 0 thì trạng thái i không hoạt động.

Grafcet cho một quá trình luôn là một đồ hình khép kín từ trạng thái đầu đến trạng thái cuối và từ trạng thái cuối đến trạng thái đầu.

3.3.5. Một số ký hiệu dùng trong grafcet

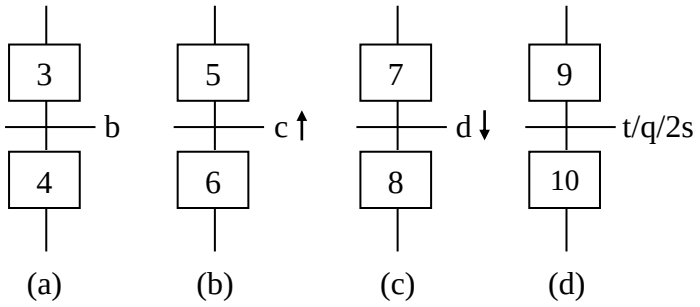
- Một trạng thái được biểu diễn bằng một hình vuông có đánh số. Gắn liền với biểu tượng trạng thái là một hình chữ nhật bên cạnh. Trong hình chữ nhật ghi hành vi của trạng thái đó.

- *Trạng thái khởi đầu* được thể hiện bằng hai hình vuông lồng vào nhau.
- *Trạng thái hoạt động* có thêm dấu "." ở trong hình vuông trạng thái



Hình 2. 12 a, b. ký hiệu trạng thái; c.trạng thái khởi đầu; d. trạng thái hoạt động

- Việc chuyển tiếp từ trạng thái này sang trạng thái khác được thực hiện khi các điều kiện chuyển tiếp được thoả mãn. Chẳng hạn việc chuyển tiếp giữa các trạng thái 3 và



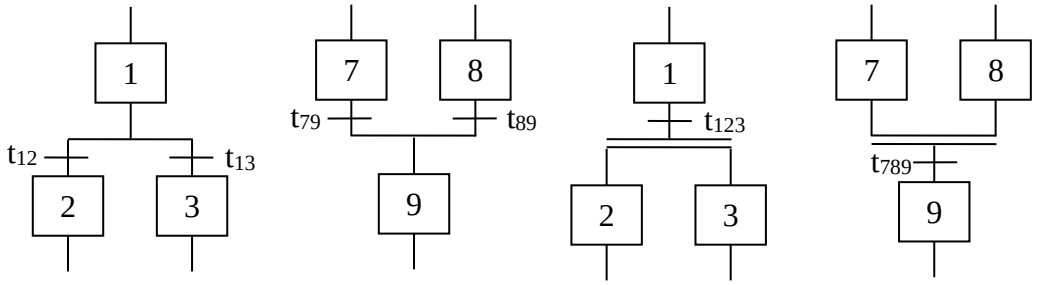
Hình 2. 13 Các trạng thái chuyển tiếp

4 (hình 2.13a) được thực hiện khi tác động lên biến b, còn chuyển tiếp giữa trạng thái 5 và 6 được thực hiện ở sườn tăng của biến c (hình 2.13b), ở hình 2.13c là tác động của sườn giảm biến d. Chuyển tiếp giữa trạng thái 9 và 10 (hình 2.13d) sẽ xảy ra sau 2s kể từ khi có tác động cuối cùng của trạng thái 9 được thực hiện.

- Các kí hiệu phân nhánh.

Hình 2.14a, 2.14b, 2.14c, 2.14d là các kí hiệu phân nhánh của grafcet.

ở hình 2.14a, khi trạng thái 1 đã hoạt động, nếu $t_{1,2}$ thoả mãn thì trạng thái 2 hoạt động; nếu chuyển trạng thái $t_{1,3}$ thoả mãn thì trạng thái 3 hoạt động (trạng thái OR).



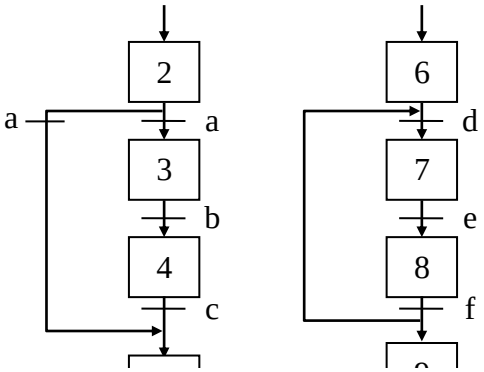
(a) OR (b) OR (c) AND (d) AND

Hình 2. 14 Ký hiệu phân nhánh của Grafcet

ở hình 2.14b nêu trạng thái 7 hoạt động và $t_{7,9}$ thoả mãn thì trạng thái 9 hoạt động, cũng như vậy nếu trạng thái 8 hoạt động và $t_{8,9}$ thoả mãn thì trạng thái 9 hoạt động (trạng thái OR)

ở hình 2.14c, nếu trạng thái 1 hoạt động và t_{123} thoả mãn thì trạng thái 2 và 3 đồng thời hoạt động (trạng thái AND).

ở hình 2.14d, nếu trạng thái 7 và 8 cùng hoạt động và t_{789} thoả mãn thì trạng thái 9 hoạt động (trạng thái AND).



Hình 2. 15 Ký hiệu bước nhảy của Grafcet

(a) (b)

Hình 2.15a biểu diễn grafcet cho phép thực hiện bước nhảy. khi điều kiện a được thoả mãn thì quá trình sẽ chuyển hoạt động từ trạng thái 2 sang trạng thái 5 và bỏ qua các trạng thái trung gian 3, 4: nếu điều kiện a không được thoả mãn các trạng thái chuyển tiếp theo trình tự bình thường ($2 \Rightarrow 3 \Rightarrow 4$).

ở hình 2.15b khi điều kiện f không được thoả mãn thì trạng thái 8 sẽ quay về trạng thái 7, nếu f thoả mãn thì trạng thái 8 mới chuyển sang trạng thái 9.

3.4. Thiết kế hệ điều khiển logic trình tự

Bài toán thiết kế hệ điều khiển logic trình tự là bài toán khó, hơn nữa từ một yêu cầu đề ra lại có nhiều cách giải quyết khác nhau. Do vậy, vấn đề chung ở đây là phải dựa vào một chỉ tiêu

tối ưu nào đó, đồng thời để tìm được lời giải tối ưu thì ngoài các suy luận toán học logic người thiết kế còn phải tận dụng các kinh nghiệm thực tế rất đa dạng và phong phú. Các phương pháp thiết kế

Phương pháp dùng bảng chuyển trạng thái:

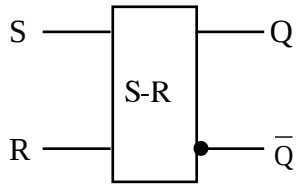
- Hệ có 2 trạng thái trong: S1 (y=1), S2(y=0), đầu ra y, 2 biến vào i_1 và i_2
- Bảng chuyển mô tả hệ như sau:

S' \ i_1i_2	00	01	11	10
S1	**S1** y=1	**X**	**S2** y=x	**S1** y=1
S2	**S1** y=x	**X**	**S2** y=0	**S2** y=0

Biến Q để mã hóa 2 trạng thái S1, S2. Q=0 ứng với S1, Q=1 ứng với S2.

S' \ i_1i_2 / Q	00	01	11	10
0	**0** y=1	**X**	**1** y=x	**0** y=1
1	**0** y=x	**X**	**1** y=0	**1** y=0

Chọn phần tử nhớ loại SR



Q(t)	Q(t+1)	S	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	x	0

SR \ i_1i_2 / Q	00	01	11	10
0	0x	**xx**	**10**	**0x**

1	**01**	**xx**	x0	**x0**

$S=i_2$

S	i_1i_2 00	01	11	10
Q				
0	0	**x**	**1**	**0**
1	**0**	**x**	x	**x**

$R=i'_1$

R	i_1i_2 00	01	11	10
Q				
0	x	**x**	**0**	**x**
1	**1**	**x**	0	**0**

$y=Q'$

y	i_1i_2 00	01	11	10
Q				
0	1	**x**	**x**	**1**
1	**x**	**x**	0	**0**

Phương pháp dùng Grafcet

Hàm vào kích phần tử

- Điều kiện tích cực bước n

$$S_n = Q_{n-1}.f_n$$

- Điều kiện không tích cực bước n

$$R_n = Q_{n+1}$$

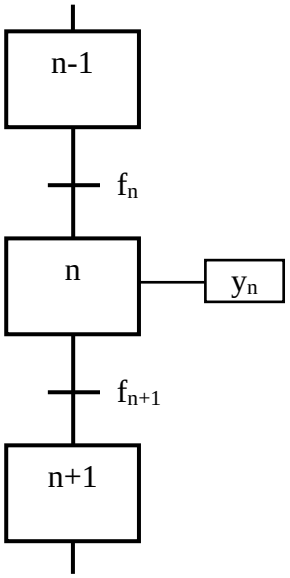
Hàm phần tử nhớ bước n

- Hàm ưu tiên Reset

$$Q_n = S_n + \overline{R}_n.Q_n = Q_{n-1}.f_n + \overline{Q}_{n+1}.Q_n$$

- Hàm ưu tiên Set

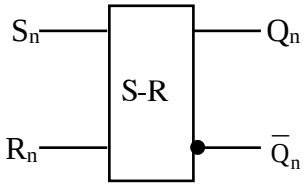
$$Q_n = (S_n + Q_n)\overline{R}_{n+1} = (Q_{n-1}.f_n + Q_n)\overline{Q}_{n+1}$$

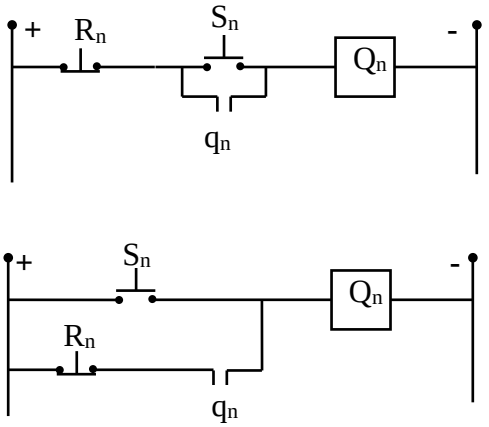


Hình 2. 16. Mô tả hệ bằng Grafcet

Hàm ra

$$y_n = Q_n$$





Hình 2. 17. Thực hiện hàm logic của các phần tử nhớ

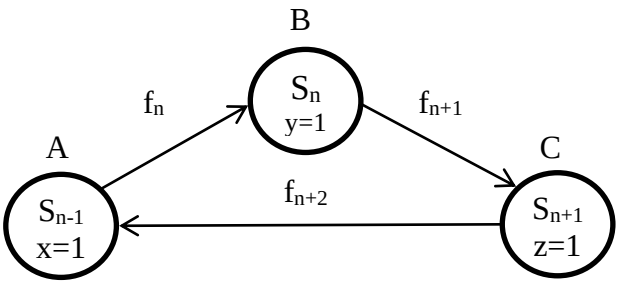
Phương pháp dùng đồ hình

❖ *Cách 1: Dùng mỗi biến trung gian mã hóa một trạng thái*

A=1, mã hóa trạng thái S_{n-1}

B=1, mã hóa trạng thái S_n

C=1, mã hóa trạng thái S_{n+1}



Hình 2. 18. Đồ hình Moore dùng mỗi biến mã hóa một trạng thái

Hàm vào kích phần tử nhớ

$S_B=A.f_n$

$R_B=C$

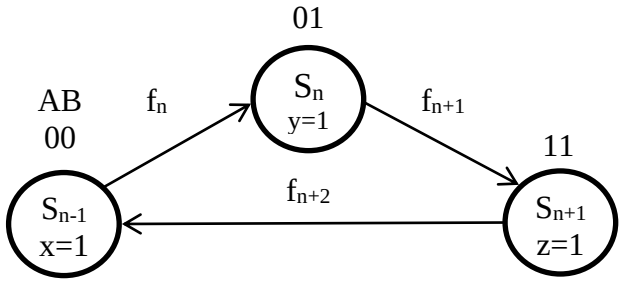
Hàm ra

Y=B

❖ *Cách 2: Dùng tổ hợp biến trung gian mã hóa trạng thái*

- n biến có thể mã hóa được 2^n trạng thái.

- Chuyển từ trạng thái này sang trạng thái khác, tổ hợp biến mã hóa chỉ khác nhau về giá trị của một biến, có thể thêm trạng thái trung gian để đảm bảo điều kiện này. Khi đó đầu ra của trạng thái trung gian là không xác định.



Hình 2. 19. Đồ hình Moore dùng tổ hợp biến để mã hóa trạng thái

Ví dụ tổng quát: dùng 2 biến A,B để mã hóa 3 trạng thái. AB=00 mã hóa trạng thái S_{n-1}, AB=01 mã hóa trạng thái S_n, AB=10 mã hóa trạng thái S_{n+1}, AB=11 mã hóa trạng thái trung gian. Các hàm vào ra có được như sau:

$S_A=A'B.f_{n+1}$

$R_A=AB'.f_{n+2}$

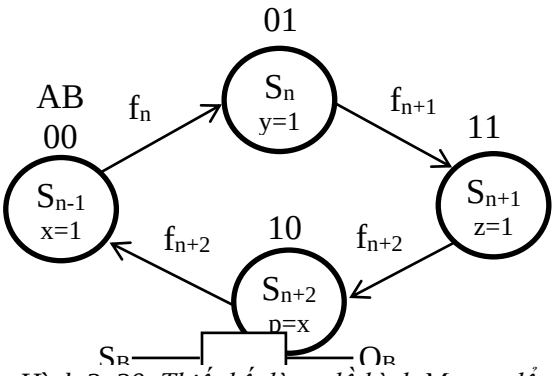
$S_B=A'B'.f_n$

$R_B=AB.f_{n+2}$

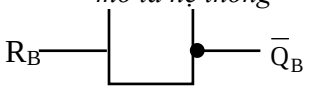
$x=A'B'$

$y=A'B$

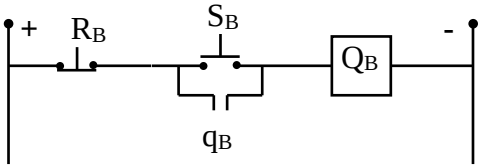
$z=AB$



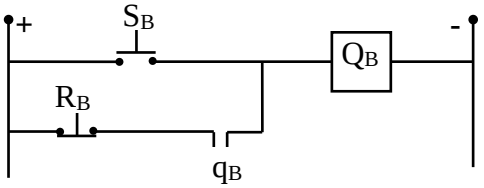
Hình 2. 20. Thiết kế dùng đồ hình Moore để mô tả hệ thống



Hình 2. 21. Mạch nhớ SR cho hàm kích



Hình 2. 22. Mạch nhớ dùng rơ le- Reset



Hình 2. 23. Mạch nhớ dùng rơ le-Set

Hàm phần tử nhớ bước n

- hàm ưu tiên Reset

$$Q_B=S_B+\overline{R}_B.Q_B$$

- hàm ưu tiên Set

$$Q_B=(S_B+Q_B)\overline{R}_B$$

Phương pháp dùng lưu đồ

Điều kiện tích cực bước n

$$S_n = Q_{n-1}.f_n$$

Điều kiện không tích cực bước n

$$R_n = Q_{n+1}$$

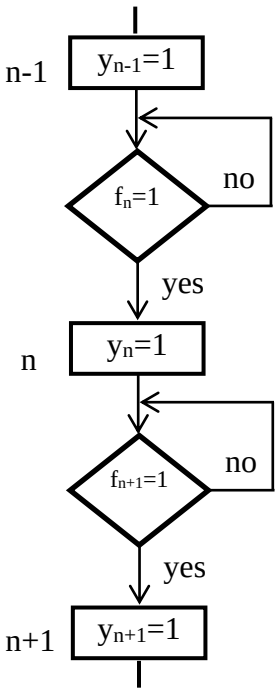
Hàm phần tử nhớ bước n

- hàm ưu tiên không tích cực

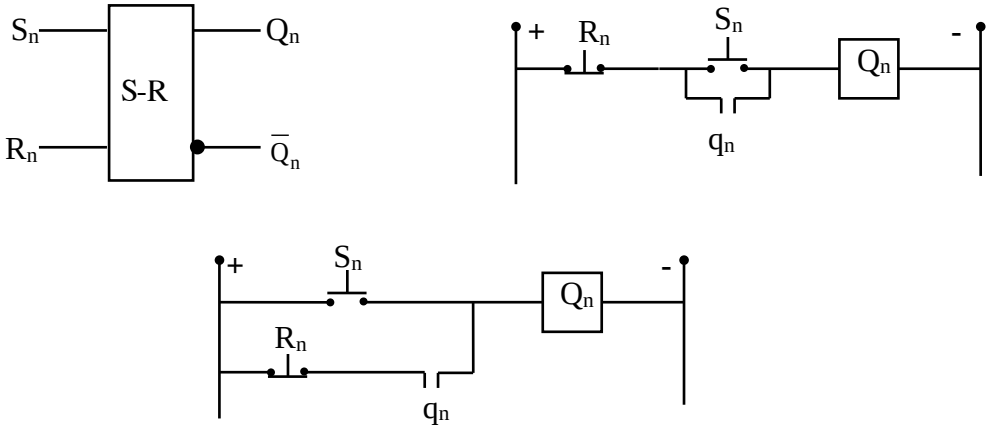
$$Q_n = S_n + \overline{R}_n.Q_n = Q_{n-1}.f_n + \overline{Q}_{n+1}.Q_n$$

- hàm ưu tiên tích cực

$$Q_n = (S_n + Q_n)\overline{R}_{n+1} = (Q_{n-1}.f_n + Q_n)\overline{Q}_{n+1}$$



Hình 2. 24. Lưu đồ thuật toán



Hình 2. 25. Thực hiện mạch nhớ với phương pháp thiết kế dùng lưu đồ

Trình tự chung trong thiết kế hệ điều khiển logic trình tự

Bước 1. Phân tích yêu cầu, chọn phương án thực hiện và xác định biến vào/ra

Bước 2. Mô tả hệ thống

Bước 3. Tìm hàm logic

Bước 4. Thực hiện

3.5 Ví dụ thiết kế dùng đồ hình trạng thái, dùng Grafcet và lưu đồ thuật toán

Thiết kế điều khiển tự động cửa một gara sử dụng các kỹ thuật. Hoạt động của bộ điều khiển cửa gara như sau:

- Một nút bấm trên gara và một nút bấm trên điều khiển từ xa

- Khi nút được bấm thì cửa sẽ đi lên hoặc đi xuống

- Nếu nút bấm được ấn một lần trong khi cửa đang chuyển động thì cửa sẽ dừng, ấn nút lần thứ hai cửa sẽ bắt đầu chuyển động trở lại và theo hướng ngược lại.

- Có khóa giới hạn trên và giới hạn dưới để dừng chuyển động của cửa.

- Tia sáng đạt ngang cửa phía dưới, khi tia sáng bị cắt trong khi cửa đang đóng thi cửa dừng lại và đảo chiều.

Thiết kế dùng đồ hình trạng thái

Bước 1: Phân tích yêu cầu, chọn phương án thực hiện và xác định biến vào/ra

Các biến vào:

- Nút ấn trong gara: G, G=1 khi ấn nút, G=0 khi không ấn nút(thường mở)

- Nút ấn trên điều khiển từ xa: R, R=1 khi ấn nút, R=0 khi không ấn (thường mở)

- Giới hạn mở: H, H=1 khi giới hạn chưa bị tác động, H=0 khi giới hạn bị tác động (thường kín)

- Giới hạn đóng: L, L=1 khi giới hạn chưa bị tác động, L=0 khi giới hạn bị tác động (thường kín)

- Cảm biến chùm tia sáng phát hiện vật cản dưới cửa: S, S=1 khi có vật cản, S=0 khi không có vật cản

- Thời gian trễ khi đóng hay mở cửa: T, T=1 khi đủ thời gian, T=0 khi chưa đủ thời gian

Các biến ra:

- Mở cửa: U, U=1 đang mở cửa, U=0 dừng mở cửa

- Đóng cửa: D, D=1 đang đón cửa, D=0 dừng đóng cửa

- Đèn chiếu sáng: X, X=1 đèn sáng, X=0 đèn tắt

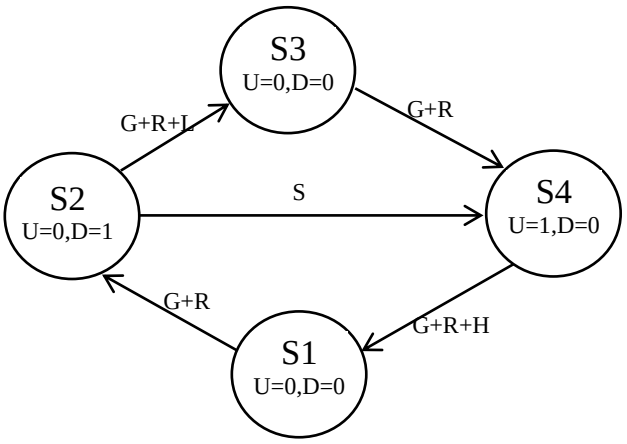
Các trạng thái

- Trạng thái cửa mở: S1

- Trạng thái cửa đang đóng: S2

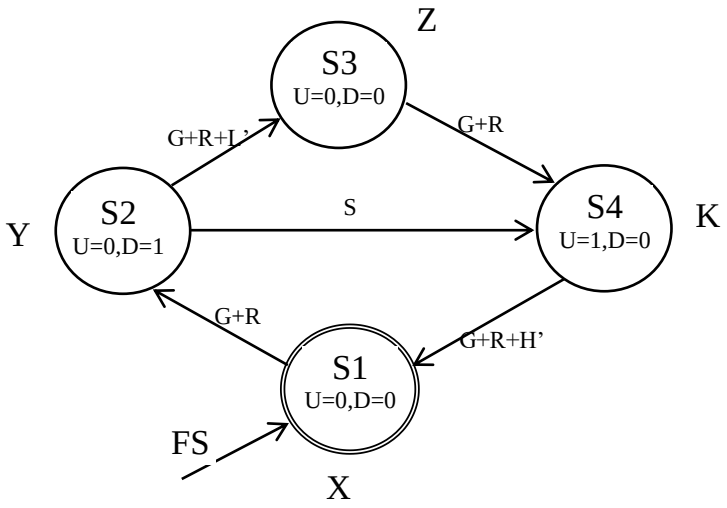
- Trạng thái cửa đóng: S3

- Trạng thái cửa đang mở: S4



Hình 2. 26. Mô tả dùng đồ hình

Bước 2: Mô tả hệ thống
- Dùng đồ hình



Hình 2. 27. Mã hóa các trạng thái

Bước 3: Tìm hàm logic

- Dùng mỗi biến mã hóa một trạng thái

X- S1, X=1 ở trạng thái S1, X=0 không ở trạng thái S1

Y- S2, Y=1 ở trạng thái S2, Y=0 không ở trạng thái S2

Z- S3, Z=1 ở trạng thái S3, Z=0 không ở trạng thái S3

K- S4, K=1 ở trạng thái S4, K=0 không ở trạng thái S4

+Hàm vào

S_X=K(G+R+H')+FS, R_X=Y

S_Y=X(G+R), R_Y=Z+K

S_Z=Y(G+R+L'),R_Z=K

S_K= Z(G+R)+YS, R_K=X

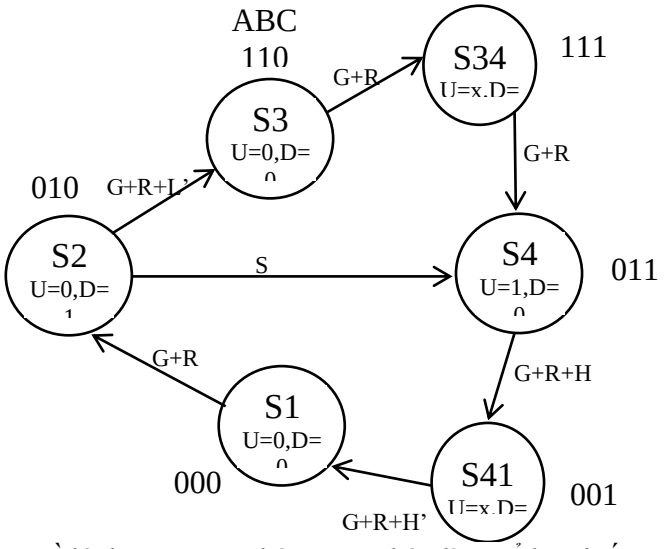
+ Hàm ra

D=Y; U=K

- Dùng tổ hợp biến để mã hóa trạng thái

Ở đây dùng 3 biến trung gian ABC để mã hóa. Để đảm bảo quy luật mã hóa, phải dùng đến 2 trạng thái trung gian S34 và S41.

Trạng thái	ABC
S1	000
S2	010
S3	110
S4	011
S34	111
S41	001



Hình 2. 28 Đồ hình Moore mã hóa trạng thái dùng tổ hợp biến

+ Hàm vào

S_A=A'BC'(G+R+L'),R_A=ABC(G+R)

S_B=A'B'C'(G+R), R_B=A'BC(G+R+H')

S_C=ABC'(G+R), R_C=A'B'C(G+R+H')

+ Hàm ra

D=A'BC'; U=A'BC

Thiết kế dùng Grafcet

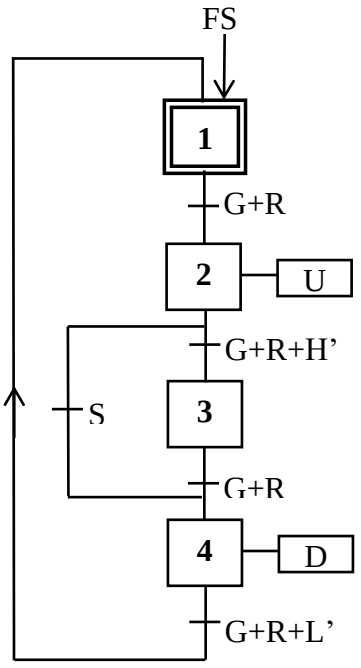
Phương pháp này chỉ khác phương pháp dùng đồ hình là ở bước 2 và bước 3

Bước 2: Mô tả hệ thống

Dùng Grafcet để mô tả hệ

Bước 3: Tìm hàm Logic

+ Hàm vào



Hình 2. 29 Mô tả hệ thống bằng Grafcet

$S_1=Q_4(G+R+H')+FS$, $R_1=Q_2$

$S_2=Q_1(G+R)$, $R_2=Q_3$

$S_3=Q_2(G+R+L')$,$R_3=Q_4$

$S_4= Q_3(G+R)+Q_2S$, $R_4=Q_1$

+ Hàm ra

$D=Q_4$; $U=Q_2$

Thiết kế dùng lưu đồ

Phương pháp này chỉ khác phương pháp dùng đồ hình là ở bước 2 và bước 3

Bước 2: Mô tả hệ thống

Dùng lưu đồ để mô tả hệ

Bước 3: Tìm hàm logic

+Hàm vào

$S_2=Q_1(G+R)$, $R_2=Q_3$

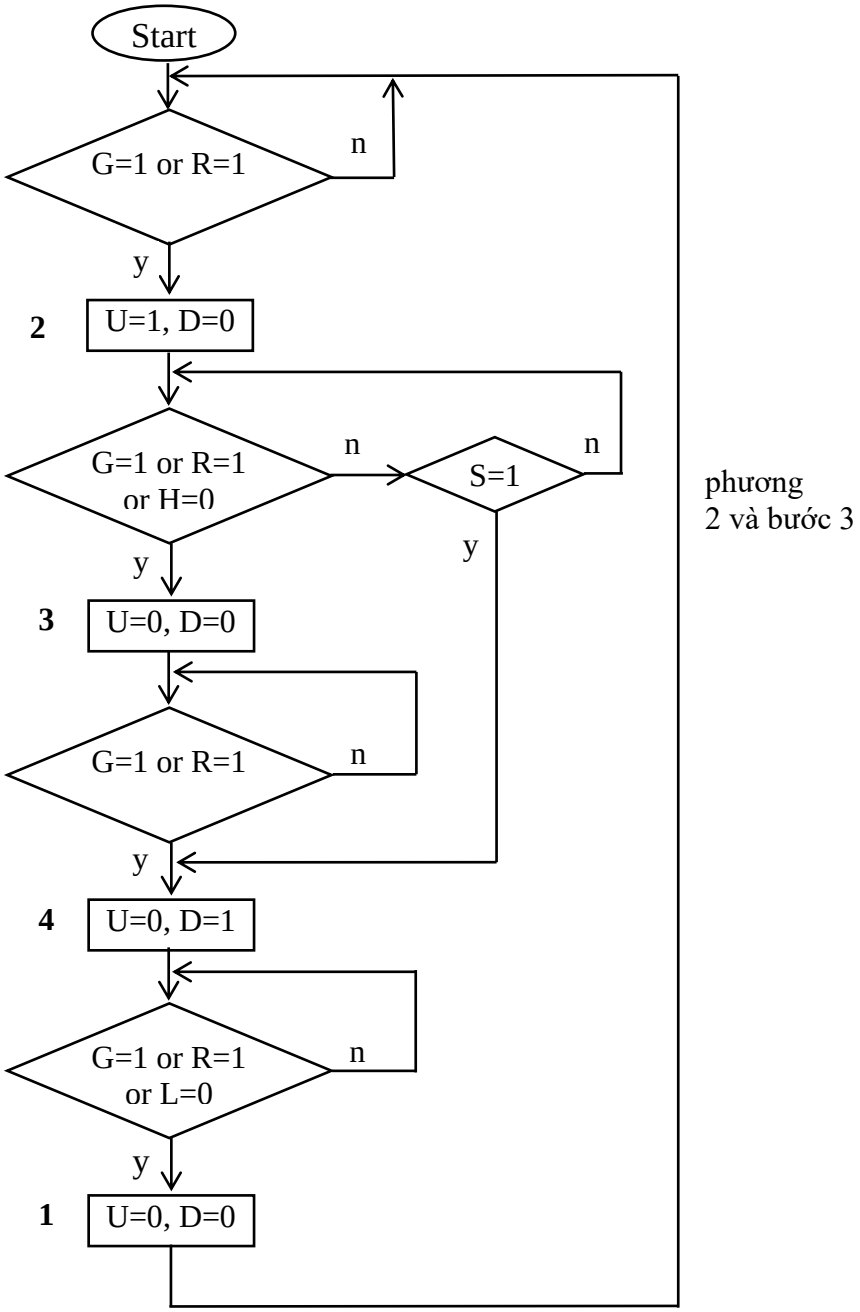
$S_3=Q_2(G+R+L')$,$R_3=Q_4$

$S_4= Q_3(G+R)+Q_2S$, $R_4=Q_1$

$S_1=Q_4(G+R+H')+FS$, $R_1=Q_2$

+ Hàm ra

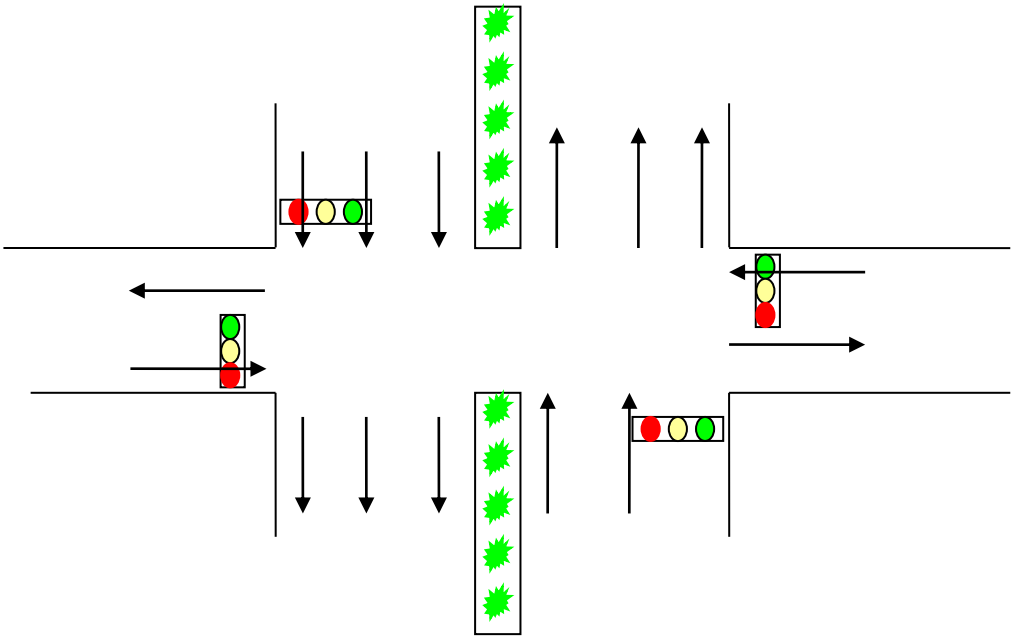
$D=Q_4$; $U=Q_2$



Hình 2. 30 Mô tả hệ thống bằng lưu đồ thuật toán

3.5 Ví dụ thiết kế hệ thống dùng bảng chuyển trạng thái

Thiết kế mạch điều khiển hệ thống đèn giao thông ở một ngã tư đường phố

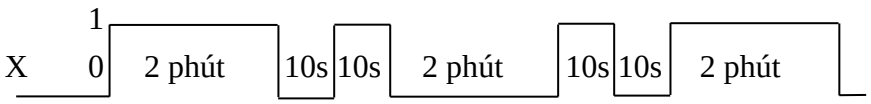


Hình 2. 31 Hệ thống đèn giao thống tại ngã tư

Trình tự các đèn sáng như sau:

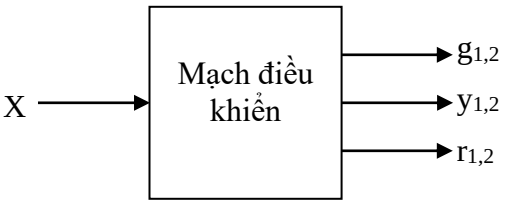
Đường 1	xanh	vàng	đỏ	đỏ	đỏ	đỏ
Đường 2	đỏ	đỏ	đỏ	xanh	vàng	đỏ
Thời gian	2 phút	10 s	10s	2 phút	10s	10s

Các đèn được điều khiển bởi thời gian của tín hiệu X như



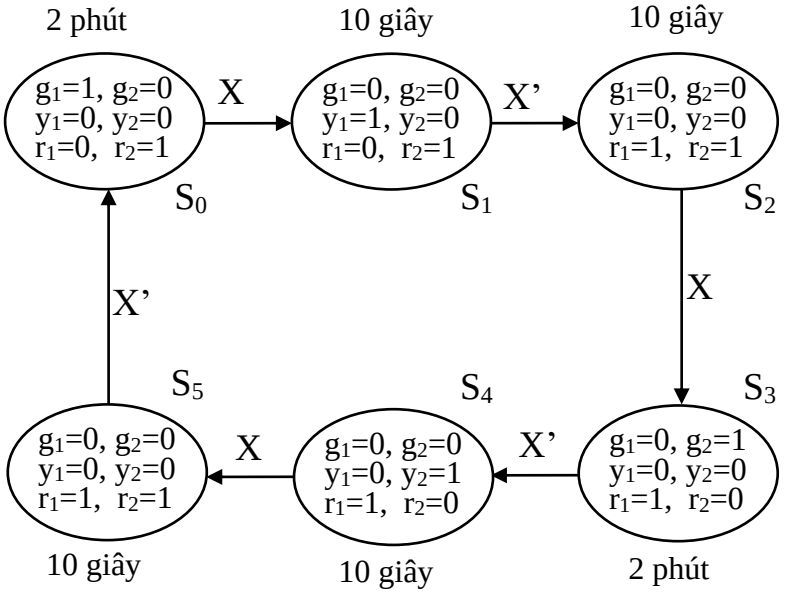
Bước 1. Phân tích yêu cầu, chọn phương án thực hiện và xác định biến vào/ra

Tín hiệu vào/ra



Bước 2. Mô tả hệ thống

Vì đầu ra chỉ phụ thuộc vào trạng thái của mạch nên dùng đồ hình Moore để mô tả hệ, biến vào là X, hệ có 6 trạng thái: S_0, S_1, S_2, S_3, S_4, S_5



Hình 2. 32 Mô tả hệ thống bằng đồ hình Moore

Trạng thái S_0 (g_1=1, y_1=0, r_1=0, g_2=0, y_2=0, r_2=1), đường 1 thông, đường 2 bị chặn, khoảng thời gian tồn tại trạng thái này là phút. Khi tín hiệu X trở về mức 0, mạch sẽ chuyển trạng thái S_1 (g_1=0, y_1=1, r_1=0, g_2=0, y_2=0, r_2=1), trạng thái S_1 tồn tại trong khoảng 10 giây cho đến khi tín hiệu X lên mức 1, mạch sẽ chuyển đến trạng thái S_2 (g_1=0, y_1=0, r_1=1, g_2=0, y_2=0, r_2=1), trong trạng thái 2 này cả hai trường hợp đều bị chặn, trạng thái S_2 này tồn tại trong khoảng 10 giây.

Khi tín hiệu X chuyển xuống mức 0, mạch sẽ chuyển đến trạng thái S_3, đường 2 thông, đường 1 bị chặn, trạng thái S_3 tồn tại trong 2 phút thì tín hiệu X chuyển lên mức 1, mạch sẽ chuyển đến trạng thái S_4 sau 10 giây mạch sẽ chuyển đến trạng thái S_5.

Như vậy quá trình từ S_0 ⟶ S_1 ⟶ S_2 là điều khiển đường 1, còn S_3 S_3 S_4 S_5 là điều khiển đường 2.

Khi tín hiệu X lên mức 1, ở trạng thái S_5 sẽ quay về trạng thái S_0 và tiếp tục bắt đầu một chu kì mới.

- Bảng trạng thái

Từ đồ hình trạng thái, vẽ được bảng trạng thái như sau:

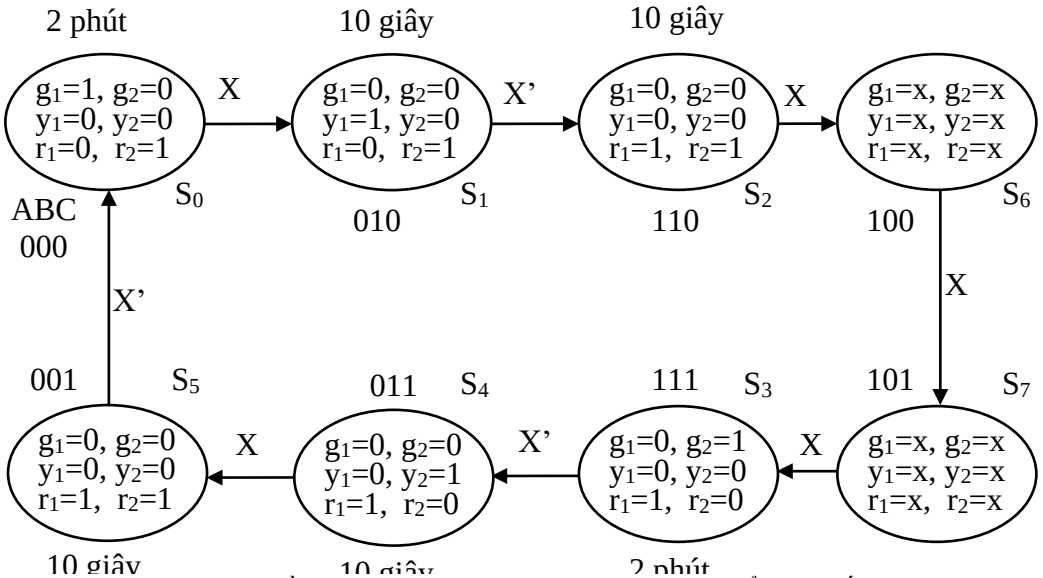
S	x=0	x=1	x=0	x=1
S_0	S_0	S_1	g_1=1, r_2 =1	y_1=1, r_2 =1
S_1	S_2	S_1	r_1=1, r_2 =1	y_1=1, r_2 =1

S_2	S_2	S_3	r_1=1, r_2 =1	r_1=1, g_2 =1
S_3	S_4	S_3	r_1=1, y_2 =1	r_1=1, g_2 =1
S_4	S_4	S_5	r_1=1, y_2 =1	r_1=1, r_2 =1
S_5	S_0	S_5	g_1=1, r_2 =1	r_1=1, r_2 =1

Bước 3. Tìm hàm logic

Xác định số lượng phần tử nhớ (bộ thời gian, mạch lật). Mã hoá các trạng thái trong.

Từ đồ hình trạng thái, thấy rằng hệ có 6 trạng thái nên để mã hoá cần đến 3 biến trung gian ABC. Như vậy còn 2 trạng thái S_6 và S_7 là không sử dụng.
Để đảm bảo điều kiện mã hóa, chuyển từ trạng thái này sang trạng thái khác chỉ khác nhau về giá trị của một biến, trạng thái S_6 và S_7 được sử dụng như đồ hình.



Hình 2. 33 Đồ hình Moore mã hóa trạng thái dùng tổ hợp biến

Xác định hàm kích thích các mạch lật và hàm tín hiệu ra.Từ đồ hình trạng thái ta có:

Hàm vào kích phần tử nhớ

$$S_A = B\bar{C}\bar{X};\ R_A = BC\bar{X}$$

$$S_B = \bar{A}\bar{C}X + ACX\ ;\ R_B = \bar{A}CX + A\bar{C}X$$

$$S_C = A\bar{B}\,X;\ R_C = \bar{A}\bar{B}\bar{X}$$

Hàm ra

$$g_1=\bar{A}\bar{B}\bar{C},\ y_1=\bar{A}B\bar{C},\ r_1=AB\bar{C}+ABC+\bar{A}BC+\bar{A}\bar{B}C=AB+\bar{A}C$$

$$g_2=ABC,\ y_2=\bar{A}BC,\ r_2=\bar{A}\bar{B}\bar{C}+\bar{A}B\bar{C}+AB\bar{C}+\bar{A}\bar{B}C=\bar{A}\bar{B}+\bar{B}\bar{C}$$

Trạng thái	Biến vào	Trạng thái	Biến ra

hiện tại	X	chuyển	g_1	y_1	r_1	g_2	y_2	r_2
S_0	0	S_0	1	0	0	0	0	1
S_0	1	S_1	x	x	x	x	x	x
S_1	0	S_2	x	x	x	x	x	x
S_1	1	S_1	0	1	0	0	0	1
S_2	0	S_2	0	0	1	0	0	1
S_2	1	S_3	x	x	x	x	x	x
S_3	0	S_4	x	x	x	x	x	x
S_3	1	S_3	0	0	1	1	0	0
S_4	0	S_4	0	0	1	0	1	0
S_4	1	S_5	x	x	x	x	x	x
S_5	0	S_0	x	x	x	x	x	x
S_5	1	S_5	0	0	1	0	0	1
S_6	0	x	x	x	x	x	x	x
S_6	1	x	x	x	x	x	x	x
S_7	0	x	x	x	x	x	x	x
S_7	1	x	x	x	x	x	x	x

Q(t)	Q(t+1)	**S**	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	x	0

TTHT	Biến vào	TTC	Đầu vào kích						Biến ra					
ABC	X	ABC	S_A	R_A	S_B	R_B	S_C	R_C	g_1	y_1	r_1	g_2	y_2	r_2
000	0	000	0	x	0	x	0	x	1	0	0	0	0	1
000	1	010	0	x	1	0	0	x	x	x	x	x	x	x
010	0	110	1	0	x	0	0	x	x	x	x	x	x	x
010	1	010	0	x	x	0	0	x	0	1	0	0	0	1
110	0	110	x	0	x	0	0	x	0	0	1	0	0	1
110	1	111	x	0	x	0	1	0	x	x	x	x	x	x
111	0	011	0	1	x	0	x	0	x	x	x	x	x	x
111	1	111	x	0	x	0	x	0	0	0	1	1	0	0
011	0	011	0	x	x	0	x	0	0	0	1	0	1	0
011	1	001	0	x	0	1	x	0	x	x	x	x	x	x
001	0	000	0	x	0	x	0	1	x	x	x	x	x	x
001	1	001	0	x	0	x	x	0	0	0	1	0	0	1
100	0	X	x	x	x	x	x	x	x	x	x	x	x	x
100	1	X	x	x	x	x	x	x	x	x	x	x	x	x

101	0	X	x	x	x	x	x	x	x	x	x	x	x	x
101	1	X	x	x	x	x	x	x	x	x	x	x	x	x

Vì đầu ra chỉ phụ thuộc vào trạng thái nên có được bảng mô tả hàm ra của hệ như sau:

g_1

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

y_1

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

r_1

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

g_2

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

y_2

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

r_2

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

S_A

AB \ CX	00	01	11	10
00	x	x	x	x
01	1	0	0	0
11	x	x	x	0
10	x	x	x	x

R_A

AB \ CX	00	01	11	10
00	x	x	x	x
01	0	x	x	x
11	0	0	0	1
10	0	0	0	0

S_B

AB \ CX	00	01	11	10
00	x	x	x	x
01	x	x	x	x

R_B

AB \ CX	00	01	11	10
00	x	x	x	x
01	0	0	0	0

11	**x**	**0**	**x**	**x**
10	**0**	**0**	**1**	**0**

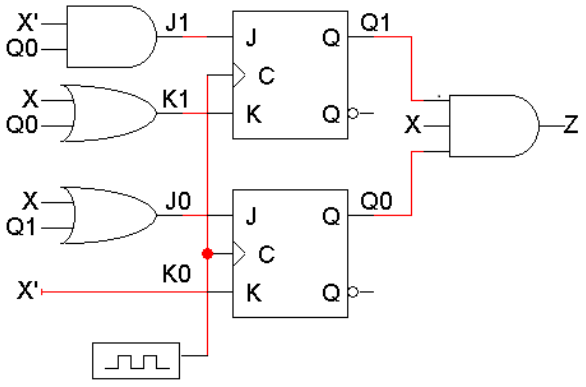
11	**0**	**1**	**0**	**0**
10	**x**	**x**	**0**	**1**

S_C CX / AB	**00**	**01**	**11**	**10**
00	**x**	**x**	**x**	**x**
01	**x**	**x**	**x**	**x**
11	**0**	**0**	**x**	**x**
10	**1**	**0**	**x**	**x**

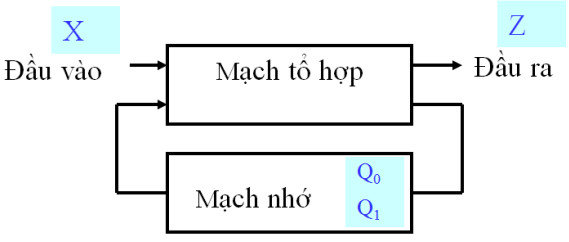
R_C CX / AB	**00**	**01**	**11**	**10**
00	**x**	**x**	**x**	**x**
01	**x**	**x**	**1**	**0**
11	**x**	**x**	**0**	**0**
10	**0**	**x**	**0**	**0**

3.6. Bài toán phân tích hệ logic trình tự

- Đây là mạch trình tự với 2 flip-flop JK. Có 1 đầu vào, X, và 1 đầu ra, Z.
- Giá trị của flip-flop (Q_1Q_0) tạo nên trạng thái, hoặc bộ nhớ, của mạch.
- Đầu ra của flip-flop cũng được được đưa về đầu vào các cổng logic bên trái.



Hình 2. 34 Mạch trình tự dùng Flip-flop J-K



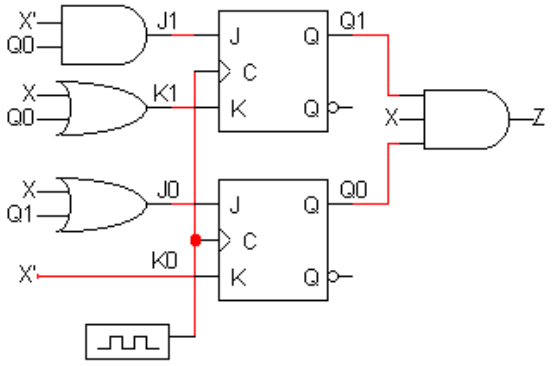
Hình 2. 35 Mạch trình tự

- Đối với mạch tổ hợp, chúng ta có thể tìm bảng chân lý, hoặc hàm logic để biểu diễn mối quan hệ đầu ra theo đầu vào.
- Bảng trạng thái của mạch trình tự tương tự bảng chân lý. Nó biểu diễn các đầu vào và trạng thái hiện tại ở bên trái, và các đầu ra và trạng thái tiếp theo ở bên phải.

– Đối với mạch trình tự, không những phụ thuộc và các đầu vào, mà còn phụ thuộc và trạng thái hiện tại của flip-flop.

– Hơn nữa để tìm các đầu ra, chúng ta cũng cần tìm trạng thái của flip-flop ở chu kì xung tiếp theo.

- Bảng các trạng thái cơ sở được trình bày như bên dưới.
- Mạch có một đầu vào X, một đầu ra Z, và 2 flip-flop Q_1Q_0.
  - Trạng thái hiện tại Q_1Q_0 và đầu vào sẽ xác định trạng thái tiếp theo và đầu ra.

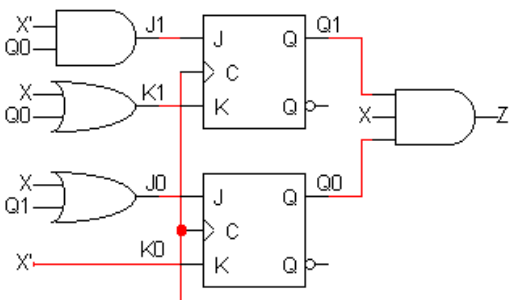


Hình 2. 36 Mạch trình tự dùng Flip-flop J-K

Trạng thái HT		đầu vào	Trạng thái chuyển		Đầu ra
Q_1	Q_0	X	Q_1	Q_0	Z
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

- Đầu ra phụ thuộc và trạng thái hiện tại – Q_0 và Q_1 – cũng như các đầu vào.
- Từ sơ đồ mạch, có thể thấy rằng

$$Z = Q_1Q_0X$$



Đầu ra ở thời điểm hiện tại

Hình 2. 37 Mạch trình tự dùng Flip-flop J-K

Trạng thái hiện tại		Đầu vào	Trạng thái chuyển		Đầu ra
Q_1	Q_0	X	Q_1	Q_0	Z
0	0	0			0
0	0	1			0
0	1	0			0
0	1	1			0
1	0	0			0
1	0	1			0
1	1	0			0
1	1	1			1

Việc tìm trạng thái tiếp theo có khó khăn hơn. Để làm điều này, chúng ta phải hiểu các flip-flop thay đổi như thế nào.

Bước 1: Tìm biểu thức logic cho các đầu vào flip-flop.

Bước 2: Dùng các biểu thức này để tìm giá trị thực của các đầu vào flip-flop với mỗi tổ hợp có thể của các trạng thái hiện tại và đầu vào.

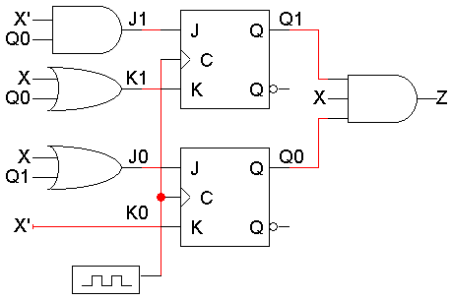
Bước 3: Dùng bảng đặc trưng hoặc phương trình đặc trưng của flip-flop để tìm trạng thái tiếp theo, trên cơ sở các giá trị đầu vào của flip-flop và các trạng thái hiện tại.

Bước 1: Biểu thức đầu vào Flip-flop

- Với ví dụ này, biểu thức đầu vào Flip-flop là:

  $J_1 = X' Q_0$; $K_1 = X + Q_0$

  $J_0 = X + Q_1$; $K_0 = X'$
- Mỗi Flip-flop JK có *2* đầu vào, J và K. (mỗi flip-flop D và T có 1 đầu vào.)



Hình 2. 38 Mạch trình tự dùng Flip-flop J-K

Bước 2: Giá trị đầu vào Flip-flop

- Với các biểu thức này, chúng ta có thể xây dựng bảng giá trị của J_1, K_1, J_1 và K_1 tương ứng với các tổ hợp khác nhau của trạng thái hiện tại Q_1Q_0 và đầu vào X.

$J_1 = X' Q_0$ $J_0 = X + Q_1$

$K_1 = X + Q_0$ $K_0 = X'$

S		Đầu vào	Đầu vào phần tử nhớ			
Q_1	Q_0	X	J_1	K_1	J_0	K_0
0	0	0	0	0	0	1
0	0	1	0	1	1	0
0	1	0	1	1	0	1
0	1	1	0	1	1	0
1	0	0	0	0	1	1
1	0	1	0	1	1	0
1	1	0	1	1	1	1
1	1	1	0	1	1	0

Bước 3: Tìm trạng thái tiếp theo

- Cuối cùng, dùng bảng đặc trưng hoặc phương trình đặc trưng của flip-flop JK để tìm trạng thái tiếp theo của mỗi flip-flop, trên cơ sở trạng và các đầu vào của flip-flop.
- Phương trình đặc trưng chung của flip-flop JK là:
  $Q(t+1) = K'Q(t) + JQ'(t)$
- Trong ví dụ này, chúng ta có 2 flip-flop JK , vì thế chúng ta phải áp dụng phương trình này cho mỗi flip-flop:

$Q_1(t+1) = K_1'Q_1(t) + J_1Q_1'(t)$

$Q_0(t+1) = K_0'Q_0(t) + J_0 Q_0'(t)$

- Chúng ta cũng có thể xác định trạng thái tiếp theo ứng với mỗi tổ hợp trạng thái hiện tại/đầu vào một cách trực tiếp từ bảng đặc trưng.

J	K	Q(t+1)	Hoạt động
0	0	Q(t)	Không thay đổi
0	1	0	Reset
1	0	1	Set
1	1	Q'(t)	Đầu ra đảo

- Trạng thái tiếp theo của Q_1 và Q_0, được tính toán theo các biểu thức:

$Q_1(t+1) = K_1'Q_1(t) + J_1 Q_1'(t)$

$Q_0(t+1) = K_0'Q_0(t) + J_0 Q_0'(t)$

S		Đầu vào	Đầu vào kích FF				S'	
Q_1	Q_0	X	J_1	K_1	J_0	K_0	Q_1	Q_0
0	0	0	0	0	0	1		
0	0	1	0	1	1	0		
0	1	0	1	1	0	1	1	
0	1	1	0	1	1	0		

1	0	0	0	0	1	1		
1	0	1	0	1	1	0		
1	1	0	1	1	1	1		
1	1	1	0	1	1	0		

- Dùng phương trình đặc trưng:

$Q_1(t+1) = K_1'Q_1(t) + J_1 Q_1'(t)$

$Q_0(t+1) = K_0'Q_0(t) + J_0 Q_0'(t)$

- Hoặc bảng đặc trưng

J	K	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	Q'(t)

S		Đầu vào	Đầu vào kích FF				S'	
Q_1	Q_0	X	J_1	K_1	J_0	K_0	Q_1	Q_0
0	0	0	0	0	0	1	0	
0	0	1	0	1	1	0		
0	1	0	1	1	0	1		
0	1	1	0	1	1	0		
1	0	0	0	0	1	1		
1	0	1	0	1	1	0		
1	1	0	1	1	1	1	0	
1	1	1	0	1	1	0		

- Cuối cùng, đây là các trạng thái tiếp theo của Q_1 và Q_0, dùng các phương trình sau:

$Q_1(t+1) = K_1'Q_1(t) + J_1 Q_1'(t)$

$Q_0(t+1) = K_0'Q_0(t) + J_0 Q_0'(t)$

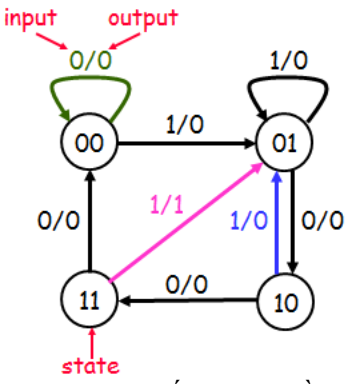
S		Đầu vào	Đầu vào kích FF				S'	
Q_1	Q_0	X	J_1	K_1	J_0	K_0	Q_1	Q_0
0	0	0	0	0	0	1	0	0
0	0	1	0	1	1	0	0	1
0	1	0	1	1	0	1	1	0
0	1	1	0	1	1	0	0	1
1	0	0	0	0	1	1	1	1
1	0	1	0	1	1	0	0	1
1	1	0	1	1	1	1	0	0
1	1	1	0	1	1	0	0	1

- Khởi đầu bảng với Present State và Inputs.
  - Present State và Inputs tạo ra FF Inputs.
  - Present State và FF Inputs tạo ra Next State, trên cơ sở bảng đặc trưng của flip-flop.
  - Present State và Inputs tạo ra Output.
- Chúng ta chỉ cần thực sự chú ý đến FF Inputs để tìm Next State.

S		Đầu vào	Đầu vào kích FF				S’		Đầu ra
Q_1	Q_0	X	J_1	K_1	J_0	K_0	Q_1	Q_0	Z
0	0	0	0	0	0	1	0	0	0
0	0	1	0	1	1	0	0	1	0
0	1	0	1	1	0	1	1	0	0
0	1	1	0	1	1	0	0	1	0
1	0	0	0	0	1	1	1	1	0
1	0	1	0	1	1	0	0	1	0
1	1	0	1	1	1	1	0	0	0
1	1	1	0	1	1	0	0	1	1

- Chúng ta cũng có thể biểu diễn một cách hình học bảng trạng thái bằng đồ hình trạng thái.
- Một đồ hình tướng ứng cho bảng trạng thái của ví dụ này được trình bày ở bên dưới.

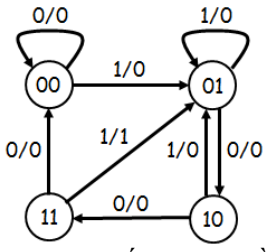
S		Đầu vào	S’		Đầu ra
Q_1	Q_0	X	Q_1	Q_0	Z
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	1	1



Hình 2. 39 Mô tả hệ thống dùng đồ hình Mealy

- Luôn luôn kiểm tra độ lớn của đồ hình trạng thái.
  - Nếu có *n* flip-flop, cần có 2^n đỉnh trong đồ hình.
  - Nếu có *m* đầu vào, khi đó mỗi đỉnh sẽ có 2^m mũi tên ra.
- Trong ví dụ này,
  - Chúng ta có 2 flip-flop, và vì thế có 4 trạng thái hoặc đỉnh.
  - Có một đầu vào, vì thế mỗi đỉnh có 2 mũi tên ra.

S		Đầu vào	S’		Đầu ra
Q_1	Q_0	X	Q_1	Q_0	Z
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	1	1



Hình 2. 40 Mô tả hệ thống dùng đồ hình Mealy

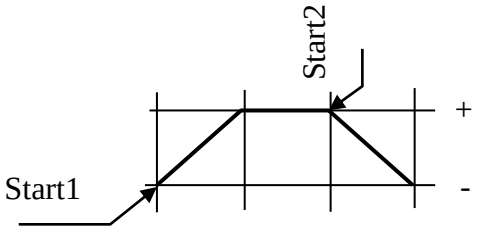
- Để phân tích mạch trình tự, cần phải:
  - Tìm biểu thức logic cho các đầu ra của mạch và các đầu vào của flip-flop.
  - Dùng các biểu thức này để điền vào các cột đầu vào và đầu ra trong bảng trạng thái.
  - Cuối cùng, dùng bảng đặc trưng hoặc phương trình đặc trưng của flip-flop để điền vào các cột trạng thái tiếptheo.
- Kết quả của phân tích mạch trình tự là một bảng trạng thái hoặc một đồ hình trạng thái mô tả mạch.

3.7. Phương pháp mô tả hệ thống dùng biểu đồ trạng thái

Với một hệ thống truyền động khí nén chúng ta hoàn toàn có thể sử dụng các phương pháp mô tả hệ thống như đã trình bày ở trên. Tuy nhiên, ở đây giới thiệu một phương pháp mô tả hệ thống đó là biểu đồ trạng thái hay còn gọi là sơ đồ hành trình bước chuyên dùng cho việc mô tả hệ truyền động khí nén.

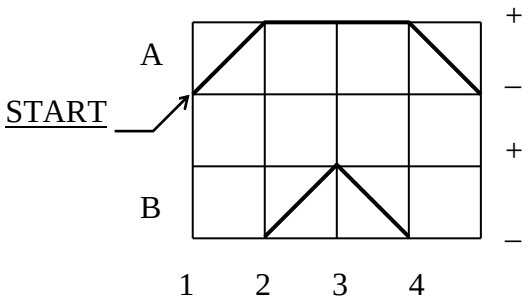
- Trong biểu đồ trạng thái người ta biểu diễn các phần tử trong mạch, mối liên hệ giữa các phần tử và trình tự chuyển mạch của các phần tử .
- Trục toạ độ thẳng đứng biểu diễn trạng thái (hành trình chuyển động, áp suất, thời gian, góc quay..) Trục toạ độ nằm ngang biểu diễn hành trình làm việc được chia thành các bước. Sự thay đổi trạng thái trong các bước được biểu diễn bằng nét liền đậm. Sự liên kết các tín hiệu biểu diễn bằng nét liền mảnh và chiều tác động biểu diễn bằng mũi tên.
- Trong mỗi cơ cấu chấp hành, nét liền mảnh nằm ngang phía trên (mang dấu +) biểu thị cho vị trí cơ cấu chấp hành ở phía ngoài (xylanh đi ra), và nét liền mảnh ở phía dưới (mang dấu -) biểu thị cơ cấu chấp hành ở phía trong. (xylanh đi vào)

Ví dụ 1 : Vẽ biểu đồ trạng thái : Nhấn nút nhấn 1 xylanh đi ra, nhấn nút nhấn 2 xylanh đi vào.



Hình 2.41. Biểu đồ trạng thái cho VD1

Ví dụ 2 : Nhấn nút nhấn xylanh A đi ra, cuối hành trình xylanh B đi ra, cuối hành trình xylanh B đi vào, và cuối cùng xylanh A đii vào, kết thúc một chu trình.

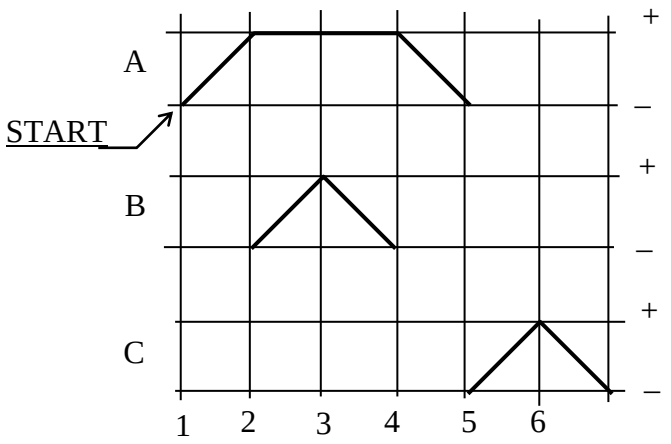


Hình 2.42. Biểu đồ trạng thái cho VD2

Ví dụ 3 : Một cơ cấu hoạt động như sau:

Một chi tiết cần khoan một lỗ khoan và được điều khiển bởi hệ thống điều khiển như sau: Các chi tiết được đặt trong một giá đỡ, nhấn một nút xy lanh tác động kép A đi ra đẩy chi tiết vào vị trí gia công đồng thời cũng kẹp chặt chi tiết, sau đó xy lanh tác động kép B được gắn với đầu khoan đi ra chậm để thực hiện công việc chuyển động chạy dao; cuối hành trình xy lanh B tự quay

về nhanh; sau đó xy lanh 1.0 quay về để tháo kẹp; cuối cùng xy lanh tác động đơn 3.0 sẽ đi ra đẩy chi tiết vừa thực hiện xong vào thùng đặt kế bên và quay về hoàn tất một chu trình.



Hình 2.43. Biểu đồ trạng thái cho VD3

Bài tập : Lập biểu đồ trạng thái cho các hệ thống khí nén sau :

1. Một cơ cấu hoạt động như sau:

Tấm thép X được uốn các góc 90 bằng hệ thống điều khiển như sau:

Tấm thép được đưa vào bằng tay, sau khi nhấn nút Start, xy lanh tác động đơn A kẹp tấm thép, xy lanh B đi ra uốn chi tiết góc 90 và lập tức quay trở về, xy lanh C đi ra uốn tiếp để hoàn tất, cuối cùng lần lượt xy lanh C và A quay trở về, chi tiết được lấy ra bằng tay.

2. Một cơ cấu máy hoạt động như sau:

Chi tiết cần khoan 2 lỗ giống nhau, được điều khiển bởi hệ thống sau:

Chi tiết được đựng trong giá đỡ; sau khi nhấn nút Start xy lanh tác động kép A đi ra đẩy chi tiết vào vị trí gia công, đồng thời chi tiết cũng được kẹp chặt, xy lanh B được gắn với đầu khoan đi ra để thực hiện chuyển động chạy dao, cuối hành trình tự quay trở về; sau đó xy lanh C đi ra để chuyển chi tiết đến vị trí thứ 2; lúc này xy lanh B lại đi ra để khoan lỗ thứ 2, cuối hành trìn xy lanh B quay về; cuối cùng xy lanh C rồi xy lanh A lần lượt đi vào hoàn tất 1 chu trình.

3.7.1. Phương pháp thiết kế mạch khí nén điều khiển theo chu trình

Đối với phương pháp điều khiển theo chu trình thì mạch điều khiển chỉ sử dụng có một nguồn duy nhất. Dựa vào sơ đồ hành trình bước, sau mỗi bước, cơ cấu chấp hành sẽ tác động vào một công tắc hành trình, tín hiệu này đưa tới mạch điều khiển tác động tiếp vào van điều khiển tương ứng để thay đổi trạng thái của cơ cấu chấp hành (tức là thực hiện bước tiếp theo) … cứ như vậy cho tới hết hành trình.

❖ **Trình tự thực hiện :**

- Từ sơ đồ hành trình bước ta xác định vị trí và số lượng các công tắc hành trình tương ứng. Đặt tên các công tắc hành trình, có thể kí hiệu là S_0 , S_1 , S_2 , S_3 …
- Vẽ các cơ cấu chấp hành (xylanh) , các van đảo chiều tương ứng (thường sử dụng các van 3/2 , 5/2 duy trì).

Lưu ý : khi sử dụng các van duy trì ta luôn qui ước vị trí ban đầu (trạng thái chưa hoạt động) là vị trí ô vuông bên phải.

- Vẽ tín hiệu vào (thường sử dụng nút nhấn 3/2 thường đóng)
- Vẽ tiếp các cong tắc hành trình tương ứng theo sơ đồ hành trình bước ta đã xác định ở trên.

- Kiểm tra
- Đánh số kí hiệu các phần tử và các công tắc hành trình theo quy ước.

Các ví dụ :

Ví dụ 1: Thiết kế hệ thống khí nén điều khiển theo chu trình.

Bài làm:

- Lập biểu đồ trạng thái:
- Xác định các tín hiêu điều khiển, vị trí và tác

dụng của các công tắc hành trình.

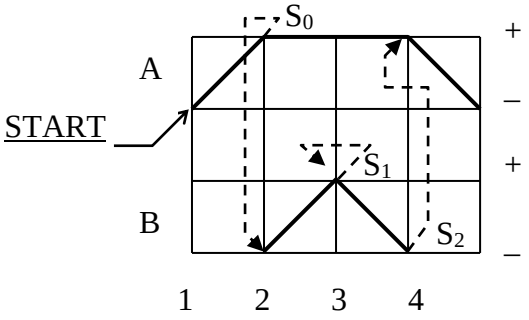
+ Nút nhấn Start → điều khiển xylanh A đi ra.

+ S_0 : ctht 1 chiều ra → nằm ngoài xylanh A

→ điều khiển B đi ra

+ S_1 : ctht 2 chiều → ngoài B → B đi vào

+S_2 : ctht 1 chiều vào → trong B → A đi vào



Hình 2.44. Biểu đồ trạng thái

- Vẽ sơ đồ mạch:

Để mạch hoạt động , tránh hiện tượng trùng tín hiệu thì vị trí các công tắc hành trình đặt như sau.

Ta biết hành trình của các xylanh là 100 mm. Vậy vị trí S_0 : 99 mm, S_1 : 100 mm, S_2 : 1 mm.

Ví dụ 2: Cho biểu đồ trạng thái, thiết kế mạch khí nén theo chu trình?

- Lập biểu đồ trạng thái
- Xác định các tín hiệu điều khiển :

+ Start → xylanh A đii ra

+ S_0 : ctht 1 chiều ra → xác định vị trí ngoài A

→ điều khiển B đii ra

+ S_1 : ctht 2 chiều → xác định vị trí ngoài B

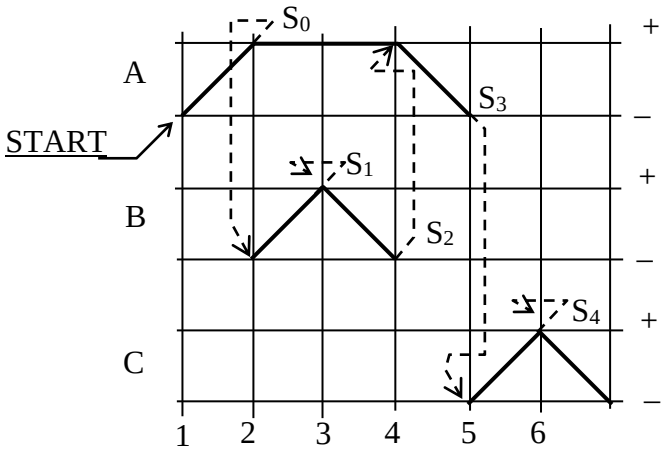
→ điều khiển B đi vào

+ S_2 : ctht 1 chiều vào → xác định vị trí trong B

→ điều khiển A đi vào

+ S_3 : ctht 1 chiều vào → xác định vị trí trong A → điều khiển C đi ra

+ S_4 : ctht 2 chều → xác định vị trí ngoài C → điều khiển C đi vào.



Hình 2.45. Biểu đồ trạng thái

3.7.2. Phương pháp thiết kế mạch khí nén điều khiển theo tầng

Nguyên tắc thiết kế mạch theo tầng là chia các bước thực hiện có cùng chức năng thành từng tầng riêng biệt, như vậy khi hoạt động thì nguồn cung caapscho hệ đảo tầng chỉ có ở tầng đang thực hiện các chuyển động, còn các tầng khác thì không có nguồn. phần tử cơ bản của điều khiển theo tầng là phần tử nhớ- van đảo chiều 4/2 hoặc 5/2. điều khiển theo tầng là bước hoàn thiện của điều khiển tùy động theo hành trình.

❖ **Nguyên lý điều khiển theo tầng.**

Trong mạch điều khiển theo tầng gồm có hai cụm

- cụm cơ cấu chấp hành : gồm các xylanh tạo ra các chuyển động, các van đảo chiều , các công tắc hành trình để chuyển đổi chuyển động của các xy lanh tương ứng.
- Cụm đảo tầng : Thực chất là các van 4/2 hoặc 5/2 duy trì.
- Giả sử biểu đồ trạng thái được chia làm n tầng:

- Đầu tiên nguồn ở cụm đảo tầng sẽ ở tầng thứ n (tầng cao nhất)
- Sau khi nhấn START nguồn sẽ chuyển đến tầng thứ 1, ở tầng này nguồn sẽ cung cấp cho các chuyển động trong tầng thứ 1, cuối tầng 1 sẽ tác động vào công tắc hành trình đảo tầng và nguồn sẽ chuyển lên tầng thứ 2, tương tự như tầng 1 nó sẽ cung cấp nguồn cho các chuyển động ở tầng 2 này. Tương tự cho đến khi nguồn chuyển đến tầng thứ n (tầng cao nhất)
- Lưu ý : Tại một thời điểm chỉ tầng đang hoạt động là có nguồn, các tầng còn lại không có nguồn. Khi nguồn chuyển sang tầng kế tiếp thì nguồn ở tầng trước đó phải bị xóa.

❖ **Nguyên tắc chia tầng**

Nếu ta ký hiệu các cơ cấu chấp hành bằng các mẫu tự A,B,C,D… và các chuyển động chạy ra được ký hiệu bởi dấu + và các chuyển động chạy vào mang dấu – thì : Trong một tầng có thể gồm nhiều mẫu tự khác nhau, nhưng một mẫu tự không được xuất hiện hai lần.

VD1 :

Dựa vào sơ đồ hành trình bước, ta có thể

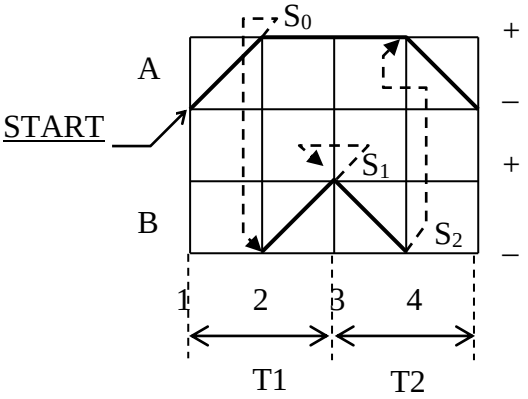
Chuyển đổi sang các mẫu tự sau:

A+ B+ B- A-

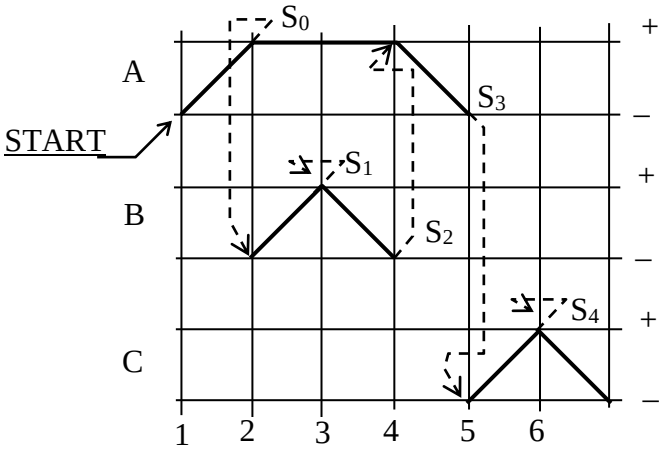
Theo nguyên tắc chia tầng ta thấy B+ B-

Không thể chung một tầng được, do đó mạch

Sẽ chia tầng từ đây. Ta có mạch 2 tầng



Hình 2.46. Biểu đồ trạng thái theo tầng



Hình 2.47. Biểu đồ trạng thái theo tầng

❖ **Biểu diễn hệ đảo tầng.**

Khi sơ đồ hành trình bước đã được chia ra làm n tầng, thì sẽ có (n-1) phần tử nhớ (n-1 van đảo tầng 5/2)

Ký hiệu e1 là tín hiệu vào tầng 1, T1 là tín hiệu ra tầng 1, tương tự e2 là tín hiệu váo tầng 2, T2 là tín hiệu ra tầng 2……..

Ta có mạch

Các bước thực hiện :

B1. lập biểu đồ trạng thái và chia tầng.

B2. Xác định các tín hệu điều khiển, các công tắc hành trình

B3. Vẽ sơ đồ mạch:

- Mạch động lực.
- Mạch đảo tầng.
- Mạch điều khiển.

- Lưu ý :

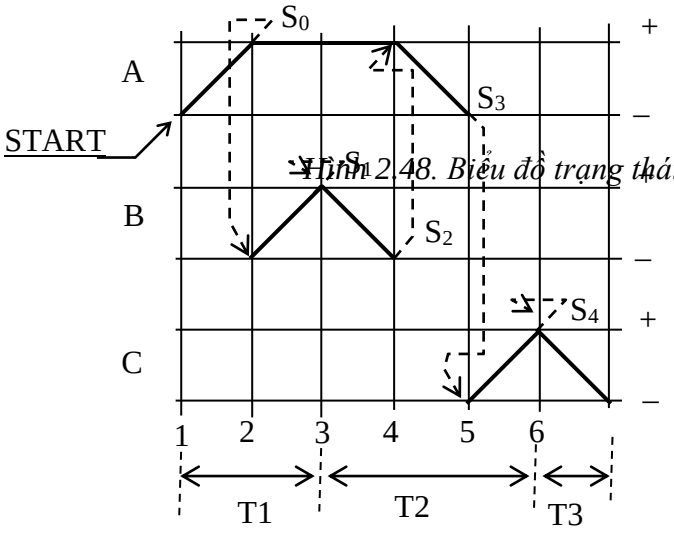
- Công tắc hành trình nào nằm giữa danh giới 2 tầng, sẽ là tín hiệu đảo tầng phía sau.
- Trong thiết kế theo tầng, tất cả các công tắc hành trình đều sử dụng CTHT tác động 2 chiều.
- Vị trí các CTHT là max, min.

Ví dụ 2: Cho biểu đồ trạng thái, thiết kế mạch khí nén theo tầng.

- Lập biểu đồ trạng thái

- Xác định các tín hiệu điều khiển :
+ Start → e1 → xylanh A đi ra
+ S_0 : → xác định vị trí ngoài A
→ điều khiển B đi ra
+ S_1 → e2 → xác định vị trí ngoài B
→ điều khiển B đi vào
+ S_2 → xác định vị trí trong B
→ điều khiển A đi vào
+ S_3 → xác định vị trí trong A → điều khiển C đi ra
+ S_4 → e3 → xác định vị trí ngoài C → điều khiển C đi vào.

- Vẽ sơ đồ mạch :

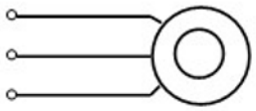
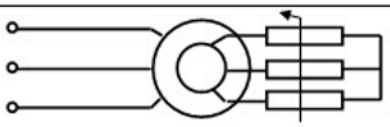
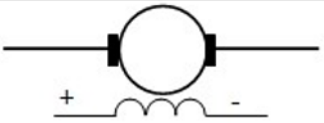


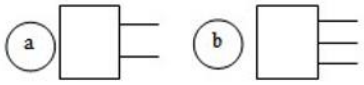
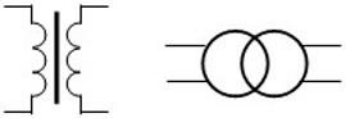
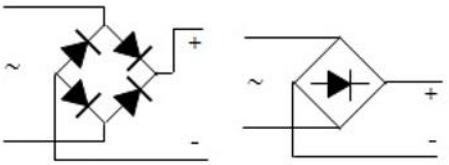
Hình 2.48. Biểu đồ trạng thái theo tầng

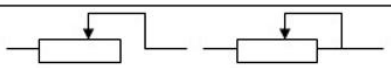
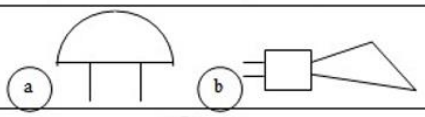
CHƯƠNG 3. GIỚI THIỆU MẠCH TỰ ĐỘNG KHỐNG CHẾ VÀ CÁC THIẾT BỊ NGOẠI VI TRONG HỆ THỐNG CN

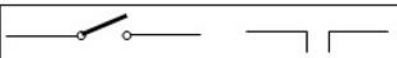
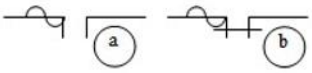
3.5.1 Kí hiệu các phần tử trong sơ đồ KCTĐ TĐĐ

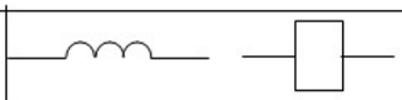
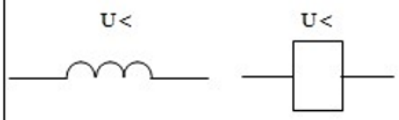
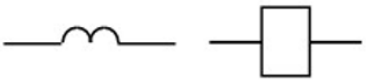
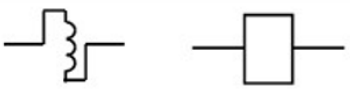
3.5.1.1 Ký hiệu các phần tử cơ bản:

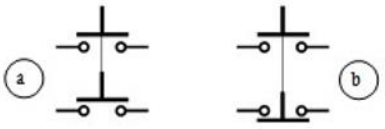
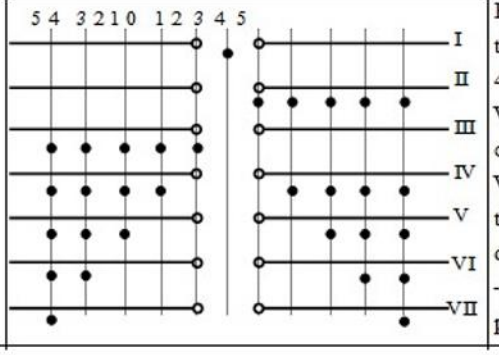
TT	KÝ HIỆU	TÊN GỌI, ĐẶC ĐIỂM
1		Động cơ điện xoay chiều không đồng bộ ba pha rotor lòng xóc (ngắn mạch)
2		Động cơ điện xoay chiều không đồng bộ ba pha rotor dây quấn
3		Động cơ điện một chiều kích từ độc lập

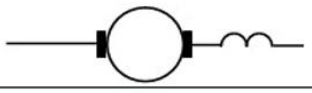
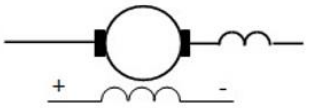
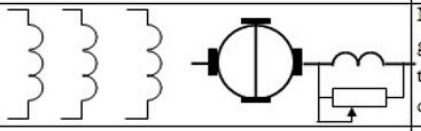
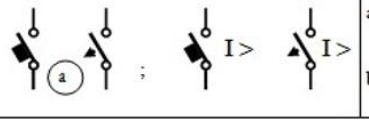
34		a/ Phanh hãm điện từ một pha b/ Phanh hãm điện từ ba pha
35		Máy biến áp một pha
36		Bộ chỉnh lưu hình cầu một pha không điều khiển (bằng 4 đi ốt)
37		Cuộn dây có lõi sắt

28		Tiếp điểm thường mở đóng - mở chậm: (Đóng chậm sau khi cuộn dây rơle có điện và mở chậm sau khi cuộn dây rơle mất điện một khoảng thời gian chỉnh định)
29		Tiếp điểm thường kín đóng - mở chậm:
30		Tiếp điểm thường kín có nút ấn phục hồi bằng tay của rơ le nhiệt và một số rơ le dòng điện
31		Biến trở
32		a/ Chuông điện b/ Còi điện
33		Đèn tín hiệu

20		Tiếp điểm thường mở (hở) của các rơle
21		Tiếp điểm thường kín (đóng) của các rơle
22		Tiếp điểm thường mở (a) và thường kín (b) của công tắc tơ
23		Tiếp điểm thường mở (a) và thường kín (b) có dập hồ quang của công tắc tơ
24		Tiếp điểm thường mở đóng chậm (đóng chậm sau khi cuộn dây rơle thời gian có điện một khoảng thời gian theo giá trị đã chỉnh định)
25		Tiếp điểm thường mở mở chậm
26		T/ điểm thường kín đóng chậm
27		Tiếp điểm thường kín mở chậm

14		Cuộn dây rơ le điện áp
15		Cuộn dây rơ le điện áp thấp (thiếu điện áp)
16		Cuộn dây rơ le dòng điện
17		Cuộn dây rơ le dòng điện cực đại
18		Phần tử đốt nóng của rơ le nhiệt
19		Cuộn dây công tắc tơ

10		Công tắc hành trình: a - Thường hở b - Thường kín
11		Nút ấn: a - Thường hở b - Thường kín
12		Nút ấn kép có liên động cơ khí: a/ 2 nút thường hở b/ 1 thường hở và 1 thường kín
13		Bộ khống chế chỉ huy khai triển theo mặt phẳng có 6 vị trí: 0, 1, 2, 3, 4, 5 và 7 tiếp điểm: I, II, III, IV, V, VI, VII. Tay gạt điều khiển có hai chiều: trái, phải. Ví dụ: - Khi tay điều khiển ở vị trí 0 thì tiếp điểm I kín, còn các tiếp điểm khác hở mạch. - Khi tay điều khiển ở vị trí 2 bên phải thì t/đ II và IV kín còn lại hở.

4		Động cơ điện một chiều kích từ nối tiếp
5		Động cơ điện một chiều kích từ hỗn hợp
6		Máy điện khuếch đại (MĐKĐ), còn gọi là máy điện khuếch đại từ trường ngang, máy điện khuếch đại có nhiều cuộn khống chế
7		Cầu dao 1 cực ; 2 cực ; 3 cực
8		Cầu chì
9		a/ Áp tô mát b/ Áp tô mát dòng cực đại

Bảng ký hiệu viết tắt tên các khí cụ điện và thiết bị điện:

KÝ HIỆU	TÊN GỌI- TÍNH NĂNG CỦA KHÍ CỤ, THIẾT BỊ ĐIỆN
A	Áp tô mát
BA	Máy biến áp
CC	Cầu chì
CD	Cầu dao
CKC	Cuộn khống chế
CKT	Cuộn kích từ
CK	Cuộn kháng
CL	Chỉnh lưu

D	Nút ấn dừng máy
Đ	Động cơ điện (nói chung)
ĐK	Động cơ điện xoay chiều không đồng bộ
F	Máy phát điện
FH	Phanh hãm điện từ
G	Công tắc tơ gia tốc
H	Công tắc tơ hãm
K	Công tắc tơ (nói chung)

KC	Bộ khống chế chỉ huy
KH	Công tắc hành trình
M	Nút ấn mở máy
MĐKĐ	Máy điện khuếch đại
NĐ	Nam châm điện
RA	Rơ le điện áp
RCB	Rơ le cưỡng bức kích thích
RG	Rơ le gia tốc
RH	Rơ le hãm
RI	Rơ le dòng điện
RN	Rơ le nhiệt
RTh	Rơ le thời gian
RTr	Rơ le trung gian
RTT	Rơ le thiếu từ trường
R_f r_f	Điện trở phụ

A. GIỚI THIỆU MẠCH TỰ ĐỘNG KHỐNG CHẾ

3.5.2 Một số nguyên tắc KCTĐ TĐĐ

3.5.2.1. Khái niệm chung

Thực **chất điều khiển tự động** là **đưa vào hoặc loại ra** khỏi hệ thống những phần tử, thiết bị nào đó (chẳng hạn: điện trở, điện kháng, điện dung, khâu hiệu chỉnh) để **thay đổi** một hoặc nhiều thông số đặc trưng hoặc **để giữ** một thông số nào đó không thay đổi khi có sự thay đổi ngẫu nhiên của thông số khác.

Để tự động điều khiển hoạt động của truyền động điện, hệ thống điều khiển phải có những cơ cấu, thiết bị **cảm nhận** được giá trị các thông số đặc trưng cho chế độ công tác của truyền động điện.

5.1. Khái niệm chung

- Nếu hệ thống điều khiển có tín hiệu phát ra từ phần tử nhận biết được thời gian của quá trình (từ một mốc thời gian nào đó) ta nói rằng hệ điều khiển theo nguyên tắc thời gian.
- Nếu hệ thống điều khiển có tín hiệu phát ra từ phần tử đo được tốc độ ta nói rằng hệ điều khiển theo nguyên tắc tốc độ.
- Nếu hệ thống điều khiển có tín hiệu phát ra từ phần tử cảm nhận được dòng điện ta nói rằng hệ điều khiển theo nguyên tắc dòng điện.
- Ngoài ra có thể điều khiển theo nhiệt độ, theo mô men, theo chiều công suất.

3.5.2.1 Khái niệm chung

- Quá trình điều khiển hệ thống truyền động điện có thể chia ra những quá trình sau:
  - Tự động điều khiển quá trình mở máy (khởi động),
  - Tự động điều khiển quá trình làm việc (duy trì một thông số nào đó theo một quy luật cho trước),
  - Tự động điều khiển quá trình hãm dừng máy.
- Một nhiệm vụ điều khiển đơn giản nhưng thường gặp là điều khiển quá trình mở máy và quá trình dừng máy các thống truyền động điện không thay đổi khi có sự thay đổi ngẫu nhiên của thông số khác.
- Khi mở máy các động cơ công suất trung bình và lớn người ta phải tiến hành hạn chế dòng khởi động nhờ các thiết bị như: điện trở, điện kháng, biến áp tự ngẫu. Quá trình khởi động xong ta phải loại trừ các thiết bị hạn chế đó ra.
  - Ví dụ: Sơ đồ lắp điện trở phụ vào mạch phần ứng động cơ một chiều kích thích độc lập hoặc vào mạch rô to động cơ không đồng bộ rô to dây quấn.

B. THIẾT BỊ NGOẠI VI

3.1 Cảm biến

3.1.1 Giới thiệu

Cảm biến (sensor) cho phép PLC phát hiện trạng thái của một quá trình. Các cảm biến logic chỉ có thể phát hiện trạng thái đúng hoặc sai. Các hiện tượng vật lý tiêu biểu cần được phát hiện là:

- Tiếp cận cảm: cho biết một đối tượng là kim loại có đến gần vị trí cần nhận biết chưa?
- Tiếp cận dung: cho biết một đối tượng là không kim loại có đến gần vị trí cần nhận biết chưa?
- Sự xuất hiện ánh sáng: Cho biết một đối tượng có làm ngắt chùm tia sáng hay ánh sáng phản xạ?
- Tiếp xúc cơ học: Đối tượng có chạm vào công tắc?

Giá thành của cảm biến ngày càng giảm thấp và trở nên thông dụng. Chúng có nhiều hình dáng khác nhau được sản xuất bởi nhiều công ty khác nhau như Siemens, Omron, Pepperl+Fuch,… Trong các ứng dụng, các cảm biến được kết nối với PLC của nhiều hãng khác nhau, nhưng mỗi cảm biến sẽ có các yêu cầu giao tiếp riêng. Phần này sẽ trình bày cách thức nối dây cho các cảm biến và một số tính chất cơ bản của nó.

3.1.2 Nối dây cảm biến

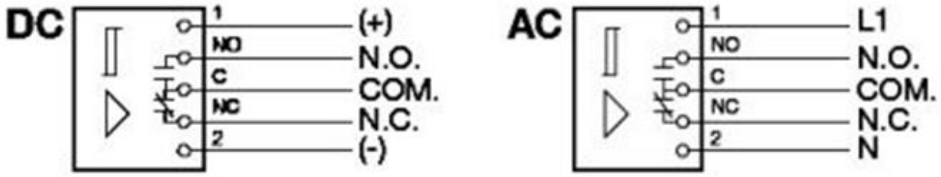
Khi một cảm biến phát hiện một sự thay đổi trạng thái logic thì nó phải truyền trạng thái thay đổi này đến PLC. Tiêu biểu là việc đóng hoặc ngắt dòng điện hay điện áp. Trong một vài trường hợp, ngõ ra của cảm biến sử dụng để đóng mạch trực tiếp cho tải mà không thông qua PLC. Các ngõ ra tiêu biểu của cảm biến là:

- *Sinking/Sourcing:* Đóng hoặc ngắt dòng điện
- *Switches:* Đóng hoặc ngắt điện áp

- *Solid State Relays:* Chuyển mạch AC
- *TTL (Transistor Transistor Logic):* Sử dụng điện áp 0V và 5V để chỉ thị mức logic.

3.1.2.1 Switch

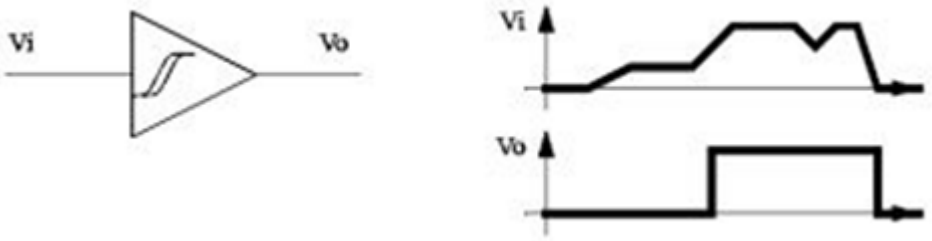
Một ví dụ đơn giản nhất của các ngõ ra cảm biến switch và relay được cho như hình 3.1.



Hình 3. 1: Cảm biến có ngõ ra là relay sử dụng nguồn DC và AC

3.1.2.2 Ngõ ra TTL

Ngõ ra TTL có hai mức điện áp: 0V tương ứng là mức thấp, 5V tương ứng mức cao. Điện áp thực tế có thể lớn hơn 0V hoặc nhỏ hơn 5V một chút vẫn có thể phát hiện đúng. Phương pháp này rất dễ bị nhiễu trong môi trường nhà máy cho nên nó chỉ được sử dụng khi cần thiết. Các ngõ ra TTL thường dùng trong các thiết bị điện tử và máy tính. Khi kết nối với các thiết bị khác thì một mạch Schmitt trigger thường được sử dụng để cải thiện tín hiệu (hình 3.2).



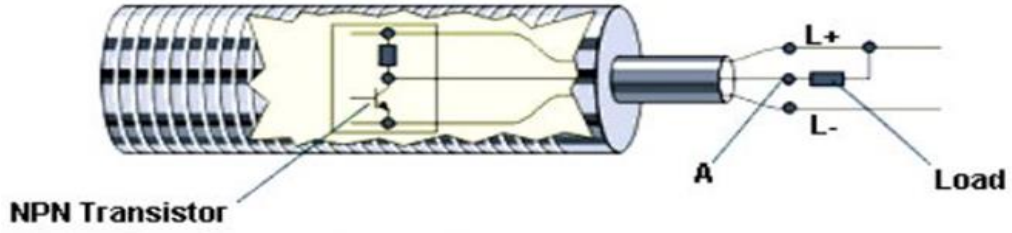
Hình 3. 2: Mạch Schmitt trigger

Mạch Schmitt trigger sẽ nhận điện áp ngõ vào giữa 0-5V và chuyển đổi nó thành 0V hoặc 5V. Nếu điện áp nằm trong khoảng 1.5-3.5V thì không chấp nhận. Nếu một cảm biến có ngõ ra TTL thì PLC phải sử dụng các ngõ vào là TTL để đọc các giá trị này. Nếu các cảm biến TTL được sử dụng cho các ứng dụng khác thì nên chú ý dòng ngõ ra cực đại của cảm biến (thường khoảng 20mA).

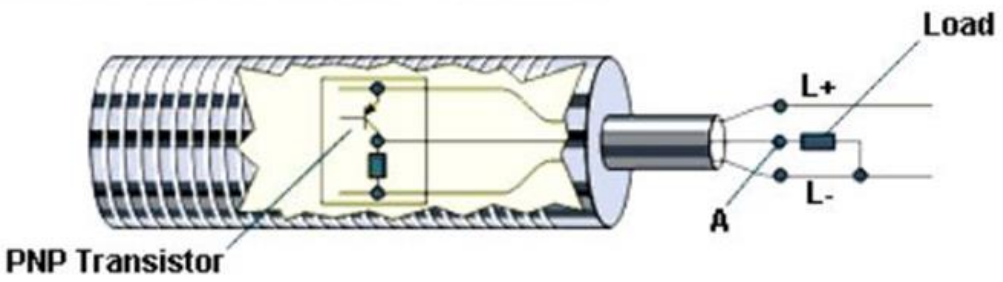
3.1.2.3 Ngõ ra Sinking/Sourcing

Các cảm biến có ngõ ra Sinking (rút dòng) cho phép dòng điện chạy vào cảm biến. Còn các cảm biến có ngõ ra sourcing (nguồn dòng) cho phép dòng điện chảy từ cảm biến ra đối tượng được kết nối. Ở hai ngõ ra này cần chú ý là dòng điện chứ không phải điện áp. Bằng cách sử dụng dòng điện thì nhiễu được loại trừ bớt.

Khi giải thích về vấn đề sinking hay sourcing thì ta nên quy các ngõ ra của cảm biến tác động như công tắc. Trong thực tế, các ngõ ra của cảm biến thường là một transistor chuyển mạch. Transistor PNP được sử dụng cho ngõ ra sourcing, và transistor NPN được sử dụng cho ngõ vào sinking. Khi giải thích các cảm biến này thì khái niệm "nguồn dòng" thường được dùng cho PNP, và "rút dòng" với NPN. Ví dụ cảm biến ngõ ra sinking được cho ở hình 3.3.

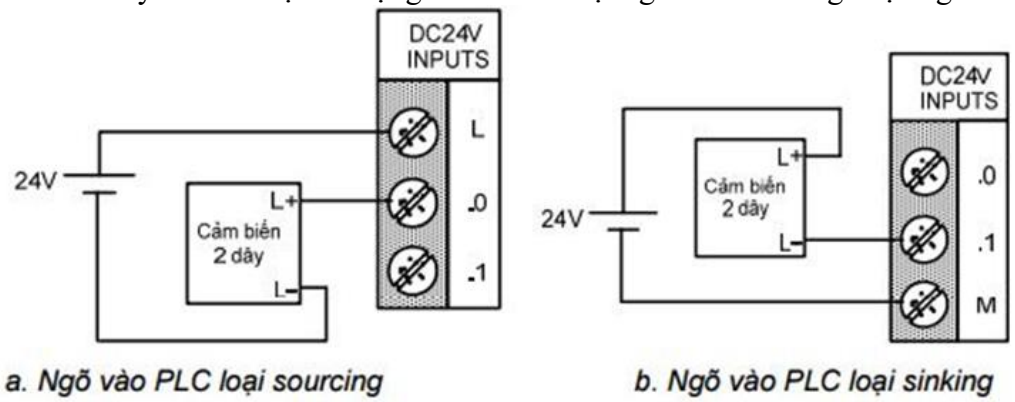


Hình 3. 3: Cảm biến NPN (cảm biến "rút dòng")



Hình 3. 4: Cảm biến PNP (cảm biến "sourcing")

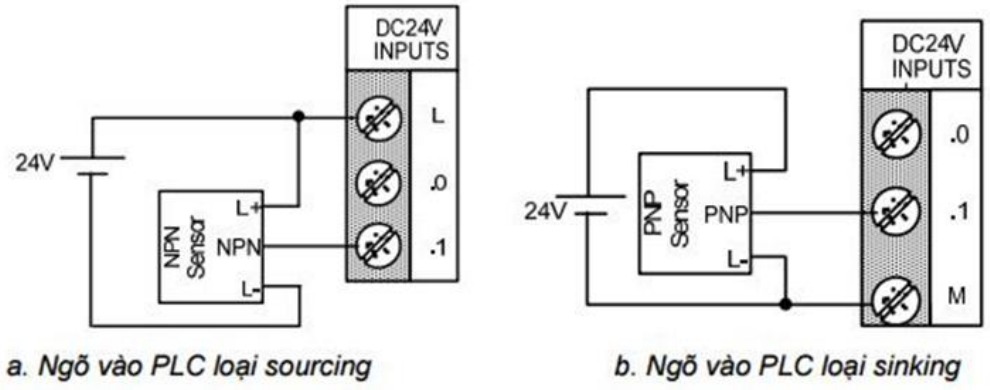
Để cảm biến hoạt động cần phải có nguồn cung cấp (chân L+ và L-). Khi cảm biến phát hiện đối tượng thì có điện áp tại cực B của transistor NPN, transistor chuyển sang trạng thái dẫn và cho phép dòng chảy vào cảm biến xuống mass (chân L-).
Khi không phát hiện đối tượng thì điện áp tại cực B của transistor ở mức thấp (0V), transistor không dẫn. Điều này có nghĩa ngõ ra NPN sẽ không có dòng vào/ra.
Các cảm biến "sourcing" thì ngược với các cảm biến "sinking". Nó sử dụng transistor PNP (hình 3.4). Khi cảm biến không được kích hoạt thì cực B của transistor ở giá trị L+, và transistor ở trạng thái ngưng dẫn. Khi cảm biến được kích hoạt thì cực B transistor sẽ được đặt ở 0V, và transistor cho phép dòng điện chảy từ cảm biến ra ngoài thiết bị được kết nối.
Hầu hết các cảm biến NPN/PNP có khả năng dòng đến vài ampere, và chúng có thể được sử dụng để nối trực tiếp với tải (luôn luôn kiểm tra sổ tay để biết chính xác dòng điện và điện áp định mức).
Chú ý: Cần phải nhớ kiểm tra dòng điện và điện áp định mức đối với các cảm biến. Khi nối dây các cảm biến cần chú ý đến các chân nguồn. Thường các
chân nguồn có ký hiệu là L+ và COM(chân chung), nhưng đôi khi không có chân COM mà có chân L-. Trong trường hợp này L- là chân chung.
Khi kết nối các cảm biến "sourcing" với các ngõ PLC, thì cần chú ý phải sử dụng các modul ngõ vào loại "sinking". Thông thường các ngõ vào PLC thường là loại "sinking".
Trong ứng dụng với PLC, để giảm lượng dây nối, thì các cảm biến hai dây thường được sử dụng. Ví dụ về sơ đồ nối dây các cảm biến sử dụng nguồn 24VDC với PLC được chỉ như hình 3.5. Cảm biến hai dây có thể được sử dụng cho cả hai loại ngõ vào sourcing hoặc ngõ vào sinking của PLC.



Hình 3. 5: Kết nối cảm biến 2 dây với ngõ vào PLC

Hầu hết các cảm biến hiện đại có cả hai ngõ ra PNP và NPN. Thông thường cảm biến loại PNP thường được sử dụng cho các ngõ vào PLC.

Trong các bản vẽ thì các chân của các cảm biến NPN và PNP có ký hiệu về màu sắc như sau: dây màu nâu là L+, dây màu xanh dương là L- và ngõ ra thì màu trắng đối với sinking và màu đen đối với sourcing.
Cần lưu ý là khi tiếp điểm trong cảm biến "sinking" đóng thì ngõ ra được nối với COM hoặc L-, tiếp điểm trong sourcing đóng thì ngõ ra nối với L+.



Hình 3. 6: Kết nối cảm biến NPN và PNP dây với ngõ vào PLC

3.1.2.4 Ngõ ra Solid staterelay

Các ngõ ra Solid state relays đóng mạch dòng điện AC. Các cảm biến này được sử dụng với tải lớn.

3.1.3 Phát hiện đối tượng

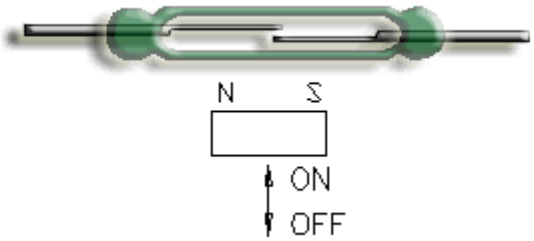
Có hai cách cơ bản để phát hiện đối tượng: tiếp xúc và tiếp cận (proximity).
Tiếp xúc có nghĩa là tiếp điểm cơ khí cần một lực tác động giữa cảm biến và đối tượng.
Tiếp cận để chỉ báo rằng một đối tượng đang ở gần nhưng không yêu cầu tiếp xúc.
Các phần sau đây sẽ minh họa các kiểu khác nhau của các cảm biến để phát hiện sự hiện diện của các đối tượng. Phần này không đi sâu vào các cảm biến mà chỉ mô tả các nguyên lý trong lĩnh vực ứng dụng.

3.1.3.1 Chuyển mạch tiếp xúc

Chuyển mạch tiếp xúc (contact switch) thường có hai dạng là thường hở (normally open) và thường đóng (normally closed). Vỏ của chúng được gia cố để có thể chịu được lực cơ tác động nhiều lần.

3.1.3.2 Switches

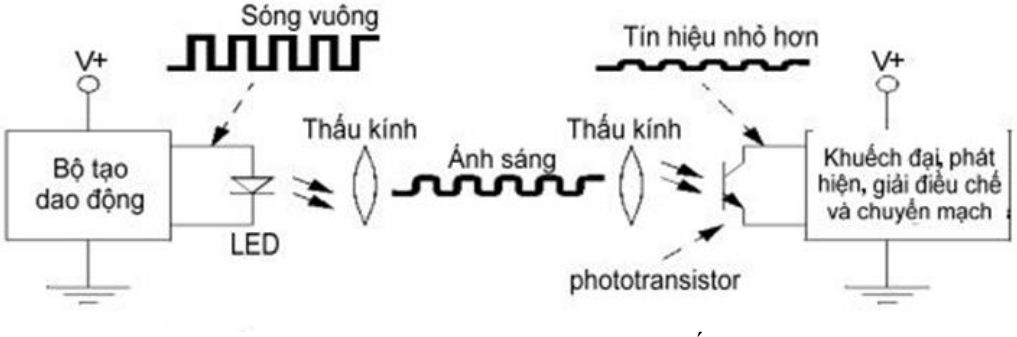
Reed switches thì rất giống relay, ngoại trừ một nam châm vĩnh cửu được sử dụng thay thế cuộn dây. Khi nam châm ở xa thì tiếp điểm mở, nhưng khi nam châm đến gần thì tiếp điểm đóng lại (hình 3.7). Các cảm biến này rẻ tiền và chúng thường được sử dụng cho các màn chắn và cửa an toàn.



Hình 3. 7 Read switch

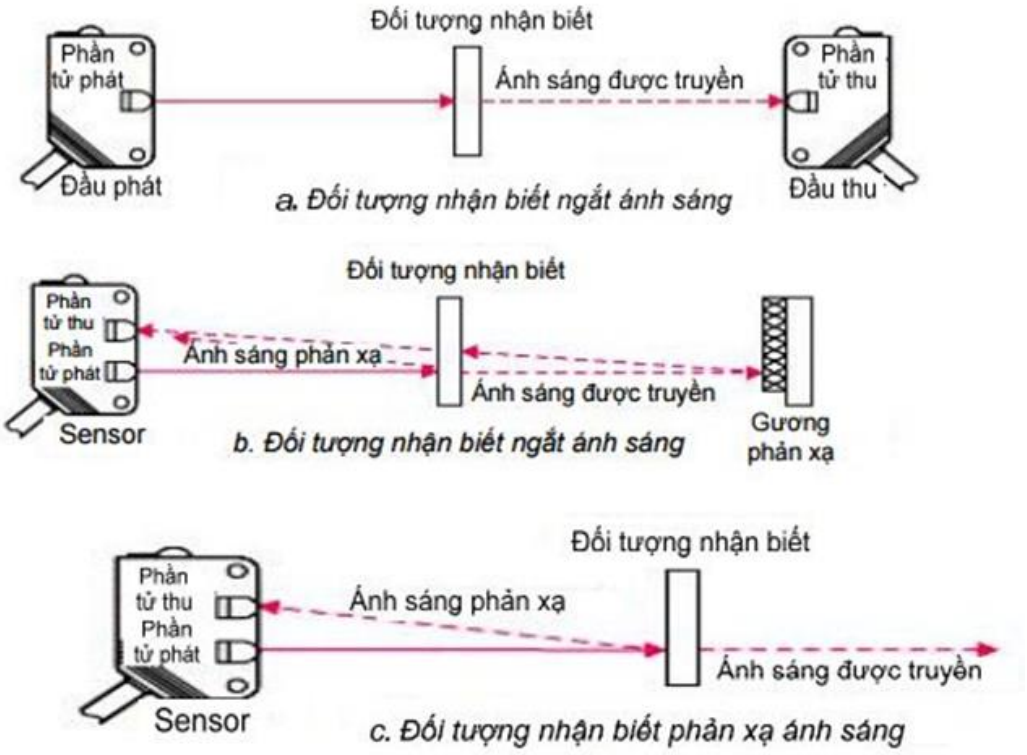
3.1.3.3 Cảm biến quang (Optical Sensor)

Cảm biến ánh sáng được sử dụng gần một thế kỷ qua. Nguyên thủy là tế bào quang được sử dụng cho các ứng dụng như đọc các track âm thanh trên các hình ảnh chuyển động. Nhưng các cảm biến quang hiện đại thì phức tạp hơn nhiều.
Các cảm biến quang yêu cầu có cả hai bộ phận là nguồn sáng (phát) và đầu thu (detector). Các đầu phát (emitter) sẽ phát ra các tia sáng trong vùng phổ nhìn thấy và không nhìn thấy được sử dụng LED và diode laser. Đầu thu có cấu tạo là các diode quang (photodidode) hoặc transistor quang (phototransistor). Đầu phát và đầu thu được đặt vào vị trí để đối tượng khi xuất hiện sẽ cắt ngang hoặc phản xạ lại tia sáng. Cảm biến quang đơn giản cho ở hình 3.8.



Hình 3. 8: Cảm biến quang

Trong hình, chùm sáng được tạo ra nằm ở bên trái, được hội tụ qua một thấu kính. Đối diện là đầu thu, chùm tia được hội tụ bằng một thấu kính thứ hai. Nếu chùm tia bị ngắt, thì đầu thu sẽ chỉ báo một đối tượng xuất hiện. Ánh sáng được tạo ra dưới dạng xung để cảm biến có thể lọc được ánh sáng bình thường trong phòng. Ánh sáng từ đầu phát được tắt và mở tại một tần số đặt. Khi đầu thu nhận ánh sáng, nó kiểm tra để đảm bảo chắc chắn rằng nó có cùng tần số. Nếu ánh sáng đang nhận được tại tần số đúng thì chùm tia không bị ngắt. Tần số dao động nằm trong phạm vi KHz. Ngoài ra với phương pháp tần số thì các cảm biến có thể được sử dụng với công suất thấp hơn và khoảng cách dài hơn. Đầu phát có thể bắt đầu từ một điểm trực tiếp tại đầu thu, đây còn gọi là chế độ tự phản xạ. Khi tia sáng bị ngắt, thì đối tượng được phát hiện. Cảm biến này cần hai bộ phận riêng (hình 3.9a). Sự xếp đặt này làm việc tốt với các đối tượng chắn sáng và phản xạ với đầu phát và đầu thu được tách riêng với khoảng cách lên đến cả trăm mét.



Hình 3. 9: Các loại cảm biến quang khác nhau

Đầu thu và đầu phát tách riêng làm tăng vấn đề về bảo trì và yêu cầu về sự thẳng hàng. Một giải pháp khác là đầu phát và đầu thu được đặt chung trên một vỏ. Nhưng điều này yêu cầu ánh sáng tự phản xạ trở về (hình 3.9b,c). Các cảm biến này chỉ tốt cho các đối tượng lớn với khoảng cách một vài met.

Trong hình, đầu phát phát một chùm tia sáng. Nếu ánh sáng bị dội trở về từ gương phản xạ thì hầu hết sẽ trở về đầu thu. Khi một đối tượng ngắt chùm tia giữa đầu phát và gương phản xạ thì chùm tia sẽ không tự phản xạ trở về đầu thu và cảm biến được tác động. Một vấn đề rủi ro cho các cảm biến này là các đối tượng tự phản xạ lại chùm tia sáng tốt. Để giải quyết thì sử dụng biện pháp phân cực ánh sáng tại đầu phát (bằng bộ lọc), và sau đó sau đó sử dụng một bộ lọc phân cực tại đầu thu.

3.1.3.4 Cảm biến điện dung (Capacitive Sensor)

Các cảm biến điện dung có thể phát hiện hầu hết các vật liệu với khoảng cách vài cm.

Công thức biểu diễn mối quan hệ điện dung:

$$C = \frac{\varepsilon . A}{d}$$

Với

C: Điện dung (Farads)

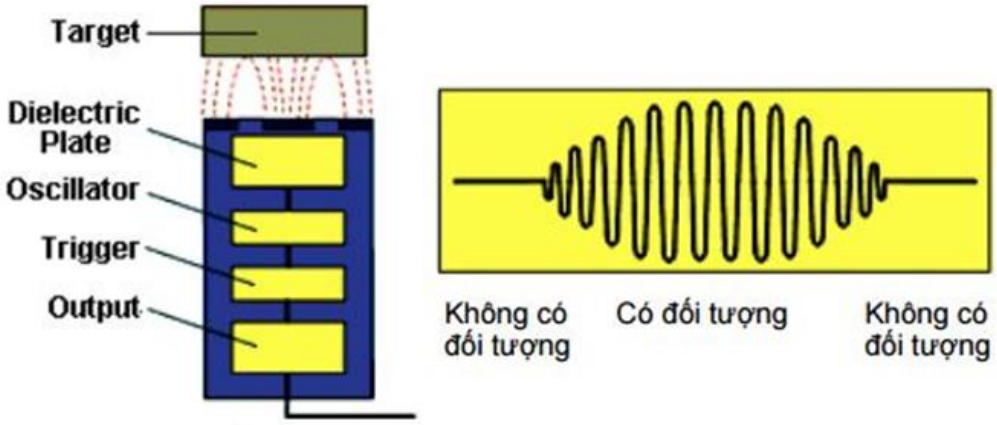
ε : hằng số điện môi

A: Diện tích bản cực

D: khoảng cách giữa các bản cực

Trong cảm biến, diện tích các bản cực và khoảng cách giữa chúng là cố định. Nhưng hằng số điện môi của không gian xung quanh chúng sẽ thay đổi khi các vật liệu được mang đến gần cảm biến. Minh họa ở hình 3.10.

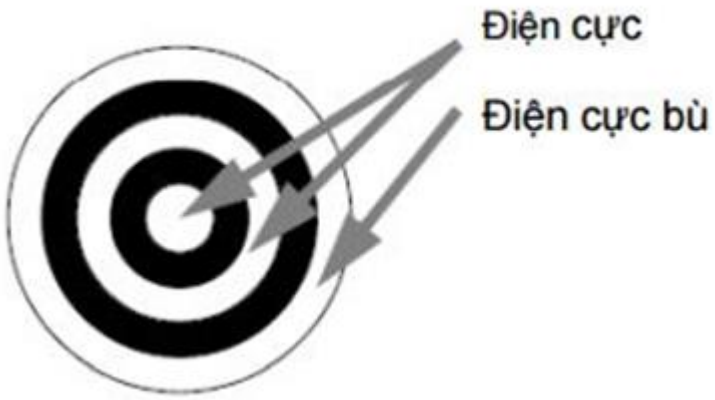
Bề mặt của cảm biến điện dung được hình thành bởi hai điện cực kim loại đồng tâm của một tụ điện. Khi một đối tượng đến gần bề mặt nhận biết nó đi vào vùng điện trường của các điện cực và thay đổi điện dung trong mạch dao động. Kết quả là bộ tạo dao động bắt đầu dao động. Mạch trigger đọc biên độ của bộ dao động và khi đạt đến mức xác định thì trạng thái ngõ ra sẽ thay đổi. Khi đối tượng rời khỏi cảm biến thì biên độ của bộ dao động giảm, cảm biến chuyển về trạng thái bình thường.



Hình 3. 10: Cảm biến điện dung

Các cảm biến này làm việc tốt đối với chất cách điện (như chất dẻo) có hằng số điện môi cao (làm tăng điện dung). Hằng số điện môi càng lớn thì khoảng cách hoạt động càng cao. Ví dụ khi hiệu chỉnh đúng thì chất lỏng trong thùng chứa có thể được phát hiện được dễ dàng. Tuy nhiên, chúng cũng làm việc tốt đối với kim loại.

Các cảm biến thường được chế tạo với các vòng (không phải bản cực) theo hình 3.11. Trong hình, hai vòng kim loại nằm bên trong là các điện cực của tụ điện, nhưng vòng ngoài thứ ba được thêm vào để bù sự thay đổi. Nếu không có vòng bù này thì cảm biến sẽ rất nhạy cảm với bụi bặm, dầu và các chất khác dính trên cảm biến.



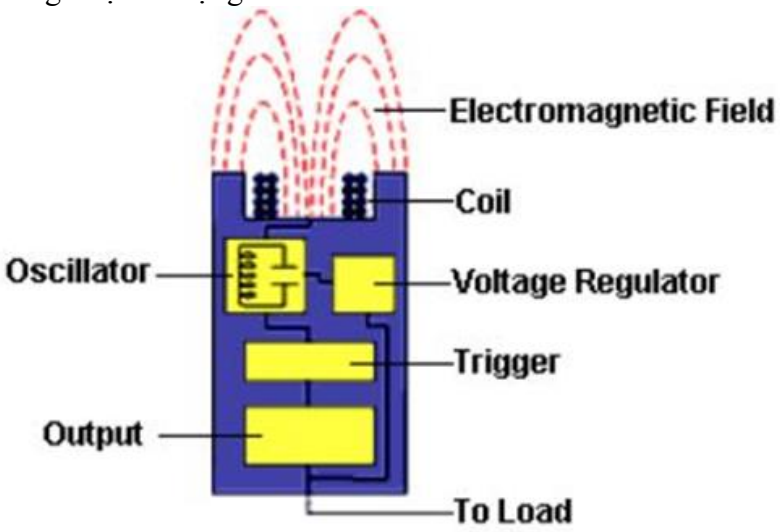
Hình 3. 11: Bề mặt nhận biết của cảm biến điện dung

Phạm vi và độ chính xác của các cảm biến được xác định bởi kích thước của chúng. Các cảm biến lớn có thể có đường kính vài centimeter. Cái nhỏ có đường kính nhỏ hơn một centimeter và có phạm vi nhỏ hơn nhưng chính xác hơn.

3.1.3.5 Cảm biến điện cảm (Inductive Sensor)

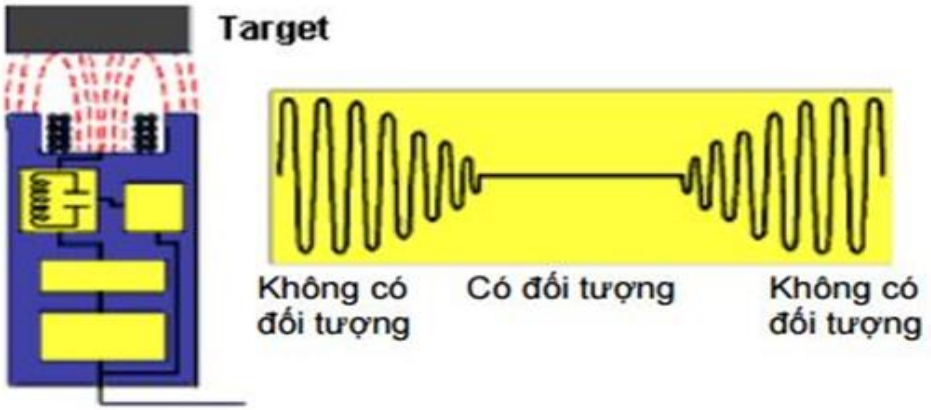
Các cảm biến điện cảm sử dụng dòng điện cảm ứng để phát hiện đối tượng là kim loại. Cảm biến điện cảm sử dụng một cuộn dây để tạo một từ trường tần số cao được cho ở hình 3.12.

Nếu có một đối tượng là kim loại đến gần làm thay đổi từ trường, thì sẽ có dòng chảy vào đối tượng. Dòng chảy này tạo ra một từ trường mới ngược với từ trường ban đầu. Kết quả là nó làm thay đổi độ tự cảm của cuộn dây trong cảm biến. Bằng cách đo độ tự cảm, cảm biến có thể xác định một đối tượng kim loại đến gần.
Các cảm biến này sẽ phát hiện bất kỳ kim loại nào, khi cần phát hiện các loại kim loại thì các cảm biến đa kim loại thường được sử dụng.



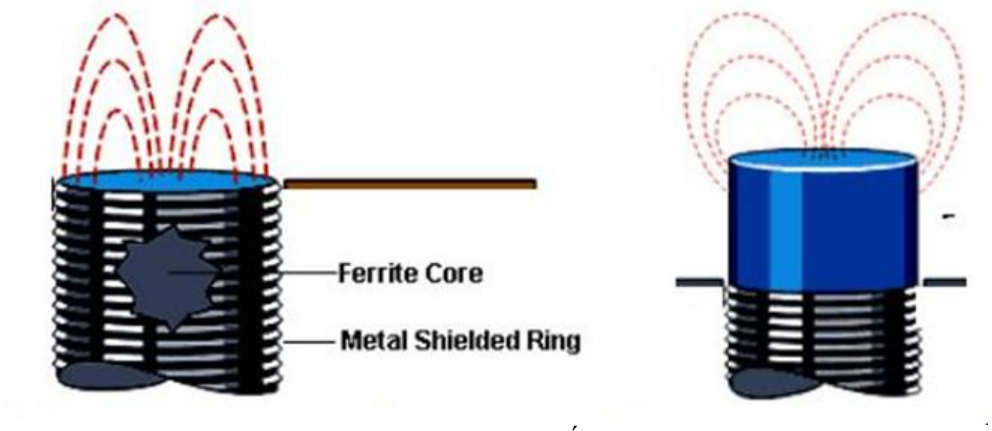
Hình 3. 12: Cảm biến tiếp cận điện cảm

Khi đối tượng kim loại đi vào vùng điện từ trường, thì dòng điện xoáy truyền vào đối tượng. Điều này làm tăng tải trong cảm biến, làm giảm biên độ của điện từ trường. Mạch trigger giám sát biên độ dao động khi đạt đến mức định trước thì nó chuyển đổi trạng thái ngõ ra của cảm biến. Khi đối tượng di chuyển khỏi cảm biến, thì biên độ dao động tăng lên. Khi đến giá trị định trước thì mạch trigger chuyển đổi trạng thái ngõ ra trở về điều kiện bình thường.



Hình 3. 13: Cảm biến tiếp cận điện cảm

Các cảm biến có thể phát hiện các đối tượng cách xa vài centimeter. Nhưng hướng của đối tượng có thể là bất kỳ như hình 3.14. Từ trường của các cảm biến không bọc bao phủ xung quanh đầu của cuộn dây lớn hơn. Bằng cách lắp thêm vỏ bọc kim loại thì từ trường sẽ nhỏ hơn, nhưng hướng của đối tượng nhận biết được cải thiện hơn.



Hình 3. 14: Cảm biến bọc và không bọc vỏ kim loại

3.1.3.6 Cảm biến siêu âm (Ultrasonic Sensor)

Cảm biến siêu âm phát ra âm thanh trên ngưỡng nghe bình thường 16kHz. Thời gian được yêu cầu để âm thanh di chuyển đến mục tiêu và phản hồi trở về tỷ lệ với khoảng cách mục tiêu. Có hai loại cảm biến là:

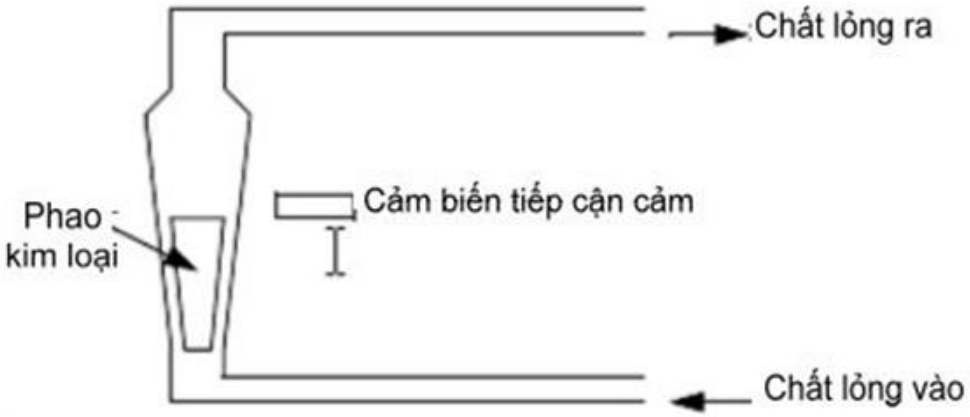
- Tĩnh điện (electrostatic): Sử dụng hiệu ứng điện dung. Phạm vi lớn và băng thông rộng hơn nhưng độ nhạy cao hơn với đối tượng ẩm ướt.
- Áp điện (piezoelectric): Dựa vào phần tử áp điện thạch anh.

Các cảm biến này có thể rất hiệu quả cho các ứng dụng như đo mức chất lỏng trong thùng chứa.

3.1.3.7 Hiệu ứng Hall (Hall Effect)

Các công tắc hiệu ứng Hall cơ bản là các transistor có thể chuyển mạch bởi từ trường. Các ứng dụng của chúng thì rất giống với reed switch, nhưng vì chúng chỉ là chất bán dẫn nên chúng phù hợp với các chuyển động. Các máy móc tự động hóa thường sử dụng chúng để thực hiện khởi động và phát hiện vị trí dừng.

3.1.3.8 Lưu lượng (Fluid Flow)



Hình 3. 15: xác định lưu lượng dòng chảy với cảm biến tiếp cận cảm

Chúng ta có thể thay thế các cảm biến phức tạp bằng các cảm biến đơn giản. Hình 3.15 cho thấy một phao kim loại trong một kênh hình nón. Tốc độ dòng chảy tăng áp lực đẩy phao lên trên. Dạng hình nón của phao đảm bảo vị trí của chất lỏng tỷ lệ với tốc độ dòng chảy. Một cảm biến tiếp cận điện cảm có thể được định vị để nó phát hiện khi phao đạt đến độ cao nào đó, và hệ thống đạt đến tốc độ dòng chảy đã định.

3.1.4 Tóm tắt

- Cảm biến Sourcing cho phép dòng điện chảy từ cực L+ của nguồn.

- Cảm biến Sinking cho phép dòng điện chảy từ cực L- của nguồn..
- Cảm biến quang có thể sử dụng chùm tia phản xạ, đầu phát và đầu thu và ánh sáng phản xạ để phát hiện đối tượng.
- Cảm biến điện dung có thể phát hiện kim loại và các vật liệu khác.
- Cảm biến điện cảm phát hiện được kim loại.
- Cảm biến hiệu ứng Hall và reed switch có thể phát hiện được nam châm.
- Cảm biến siêu âm sử dụng sóng âm để phát hiện các phần tử cách xa nhiều meter.

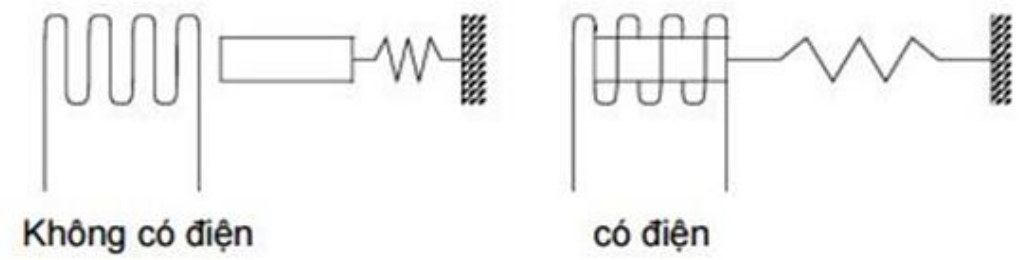
3.2 Cơ cấu chấp hành

3.2.1 Giới thiệu

Cơ cấu chấp hành của đầu ra số PLC được sử dụng để biến đổi trạng thái logic (ON-OFF) đầu ra thành trạng thái đóng cắt có công suất lớn phù hợp với tải.

3.2.2 Cuộn dây nam châm điện(Solenoid)

Cuộn dây nam châm điện là cơ cấu chấp hành thông dụng nhất. Nguyên lý hoạt động cơ bản là sự di chuyển lõi sắt trong cuộn dây (hình 3.16). Bình thường lõi sắt được giữ bên ngoài cuộn dây. Khi cuộn dây được cấp điện, cuộn dây sinh ra từ trường hút lõi sắt và kéo nó vào trung tâm của cuộn dây. Ứng dụng quan trọng nhất của solenoid là điều khiển các van khí nén, thủy lực và khóa cửa xe.

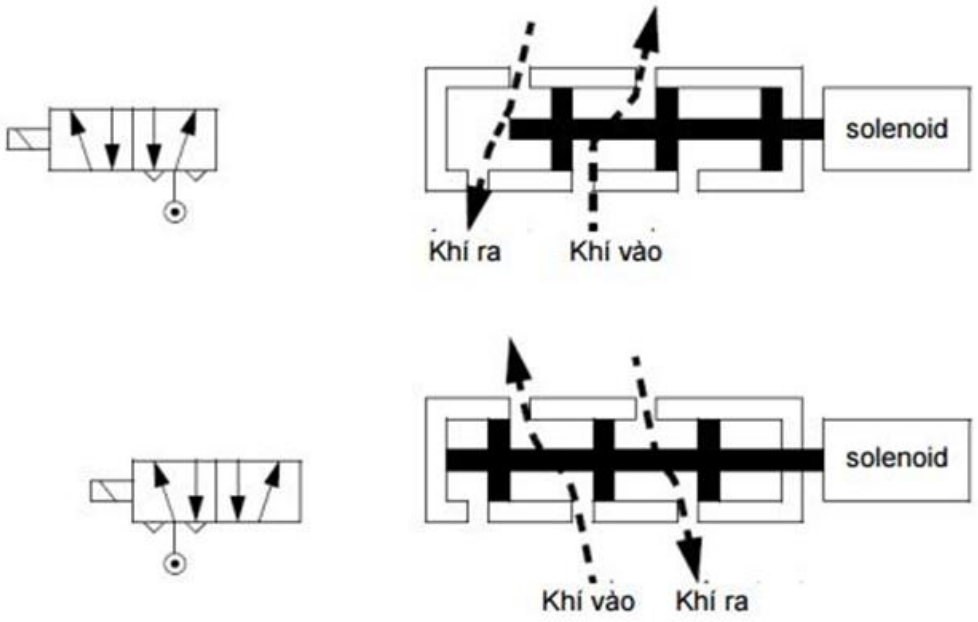


Hình 3. 16: Solenoid

Cần chú ý là các cuộn cảm có thể tạo ra điện áp đỉnh nhọn và có thể cần các bộ giảm sốc. Mặc dù vậy hầu hết trong các ứng dụng công nghiệp có điện áp thấp và dòng điện định mức, chúng có thể được kết nối trực tiếp với các ngõ ra của PLC. Hầu hết các solenoid công nghiệp sử dụng nguồn cung cấp 24Vdc và dòng định mức một vài trăm mA.

3.2.3 Van điều khiển hướng (VALVE)

Dòng chất lỏng và khí có thể được điều khiển bằng các van điều khiển solenoid. Ví dụ van điều khiển solenoid được cho ở hình 3.17.



Hình 3. 17: Một solenoid điều khiển van 5 cửa 2 vị trí

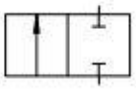
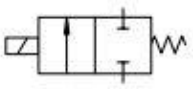
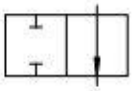
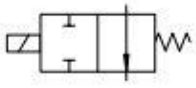
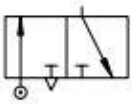
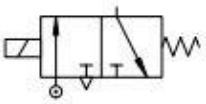
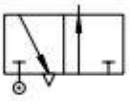
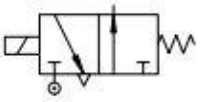
Hình 3. 18

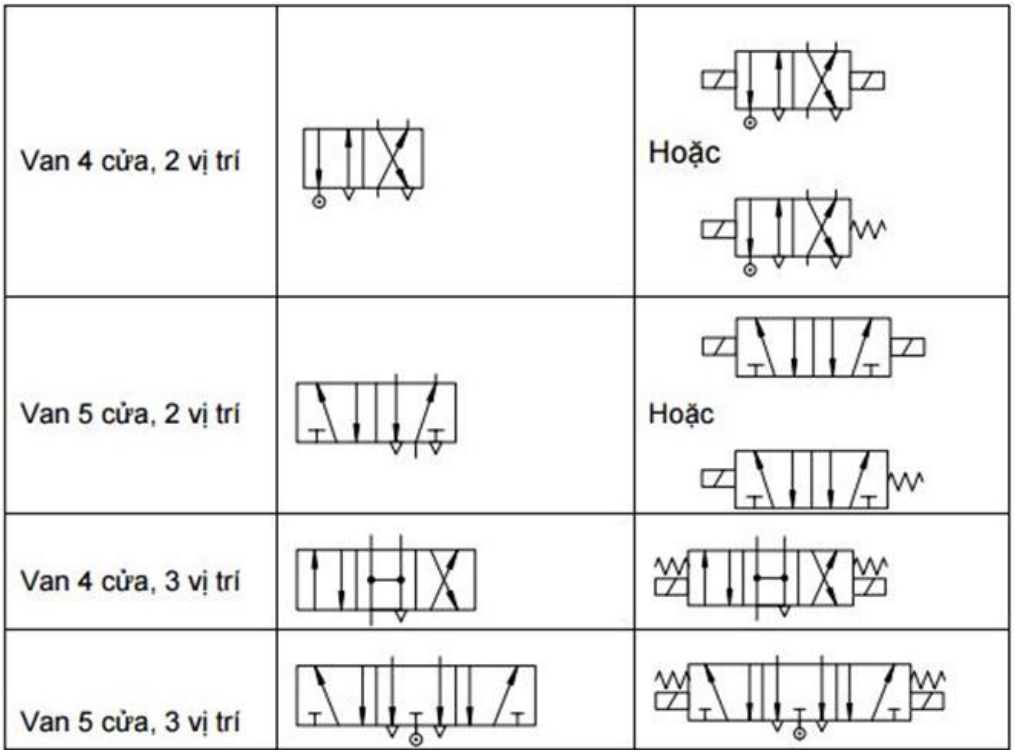
Các loại van được liệt kê dưới đây. Theo tiêu chuẩn, thuật ngữ 'n-cửa' (n-cửa) để chỉ định số lượng kết nối các ngõ vào và ra của van. Trong một vài trường hợp có cửa để xả khí ra. Việc thiết kế thường đóng/thường mở cho biết điều kiện van khi mất nguồn cấp.

- *Van 2 cửa, 2 vị trí thường đóng (van 2/2):* Các van này có 1 cửa vào và một cửa ra. Khi mất nguồn cung cấp thì ở vị trí thường đóng. Khi có nguồn cung cấp, thì van mở cho phép dòng khí hay chất lỏng chảy qua. Các van này được sử dụng để cho phép dòng chảy.
- *Van 2 cửa, 2 vị trí thường mở (van 2/2):* Các van này có một cửa vào và một cửa ra. Khi mất nguồn thì mở cho phép dòng chảy. Khi có nguồn, van đóng. Các van này được sử dụng để ngắt dòng chảy.
- *Van 3 cửa, 2 vị trí thường đóng (van 3/2):* Các van này có cửa vào, cửa ra và cửa xả khí. Khi mất nguồn thì cửa ra được nối với cửa xả khí. Khi có nguồn thì cửa vào được nối với cửa ra. Các van này được sử dụng cho các cylinder tác động đơn.

- *Van 3 cửa, 2 vị trí thường mở (van 3/2):* Các van này có cửa vào, cửa ra và cửa xả khí. Khi mất nguồn thì cửa vào được nối với cửa ra. Khi có nguồn thì van nối cửa ra với cửa xả khí. Các van này được sử dụng cho các cylinder tác động đơn.
- *Van 3 cửa, 2 vị trí đa năng (van 3/2):* Các van này có 3 cửa. Một trong các cửa hoạt động như là cửa vào hoặc cửa ra, và được nối đến một trong hai cửa khác khi mất nguồn hoặc có nguồn. Các van này có thể được sử dụng để làm chuyển hướng dòng chảy, hoặc chọn nguồn qua lại.
- *Van 4 cửa, 2 vị trí (van 4/2):* Các van này có 4 cửa, 1 vào, 2 ra và 1 cửa xả khí. Khi có nguồn van nối các cửa vào với các cửa ra và ngược lại. Các van này được sử dụng với các cylinder tác động kép.
- *Van 5 cửa, 2 vị trí (van 5/2):* Các van này có 5 cửa, 1 vào, 2 ra và 2 cửa xả khí.
- *Van 4 cửa, 3 vị trí (van 4/3):* Các van này có 4 cửa, 1 vào, 2 ra và 1 xả. Ở trạng thái bình thường (không có nguồn năng lượng) thì các cửa vào/ra đều bị chặn. Van này được sử dụng để điều khiển vị trí các cylinder.
- *Van 5 cửa, 3 vị trí (van 5/3):* Van này có 5 cửa, 1 vào, 2 ra và 2 cửa xả. Tương tự như van 4/3, van này được sử dụng để điều khiển vị trí các cylinder.

Ký hiệu của các van được cho ở hình 3.18. Khi sử dụng trong các bản vẽ thì vẽ ở trạng thái không được cấp nguồn năng lượng. Mũi tên chỉ đường dẫn dòng chảy đến các vị trí khác. Biểu tượng tam giác nhỏ để chỉ cửa xả khí.

Loại van	**Ký hiệu**	
	Điều khiển bằng khí nén	**Điều khiển bằng solenoid**
Van 2 cửa, 2 vị trí	 Thường đóng	 Thường đóng
	 Thường mở	 Thường mở
Van 3 cửa, 2 vị trí	 Thường đóng	 Thường đóng
	 Thường mở	 Thường mở



Hình 3. 19: Kí hiệu các van điều khiển bằng khí và solenoid

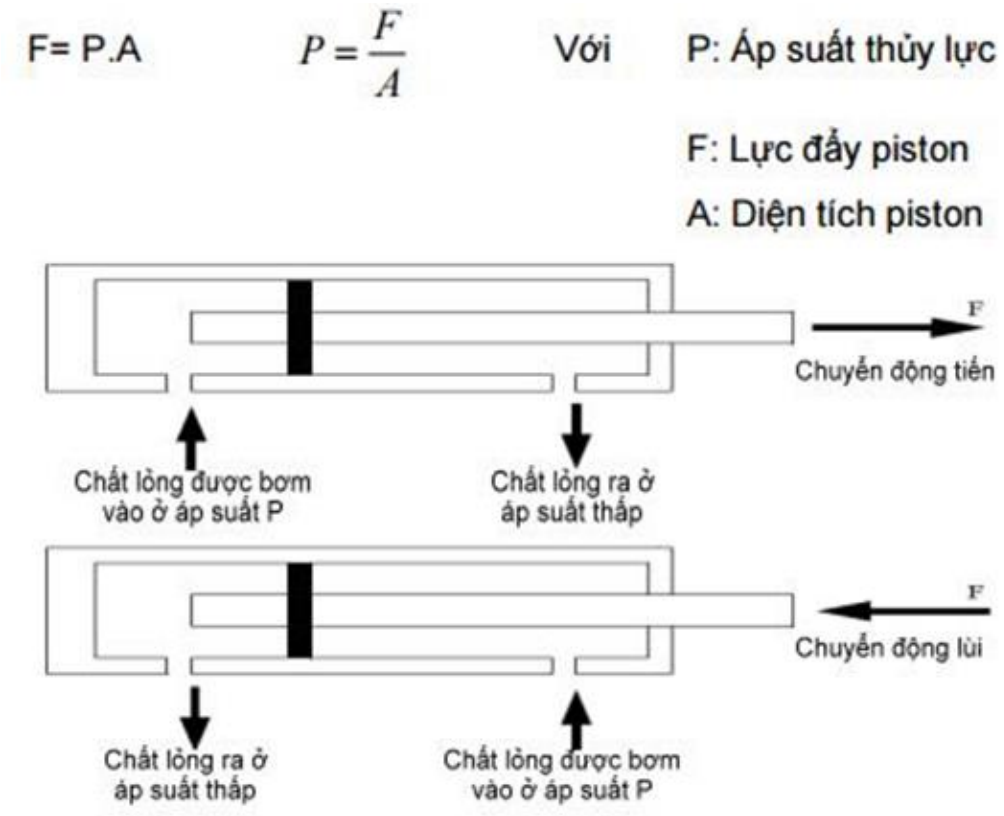
Khi chọn lựa van, cần chú ý một số chi tiết sau:

- Kích thước ống: Cửa vào và ra theo tiêu chuẩn NPT (national pipe thread).
- Tốc độ dòng chảy: Tốc độ dòng chảy cực đại thường được cung cấp cho các van thủy lực.
- Áp suất hoạt động: Áp suất hoạt động cực đại phải được chỉ báo. Một vài van có yêu cầu áp suất tối thiểu để hoạt động.
- Nguồn điện: Các cuộn dây solenoid yêu cầu được cung cấp một điện áp và dòng điện cố định (AC hoặc DC).
- Thời gian đáp ứng: Đây là thời gian để van đóng/mở hoàn toàn. Thời gian tiêu biểu cho các van nằm trong phạm vi từ 5ms đến 150ms.
- Vỏ bọc: Vỏ bọc cho các van được xếp theo loại:

Loại 1 hoặc 2: Sử dụng trong nhà, yêu cầu bảo vệ chống nước. Loại 3: Sử dụng ngoài trời, chống bụi bặm và mưa gió.
Loại 3R hoặc 3S hoặc 4: Chống nước và bụi. Loại 4X: Chống nước, bụi và sự ăn mòn.

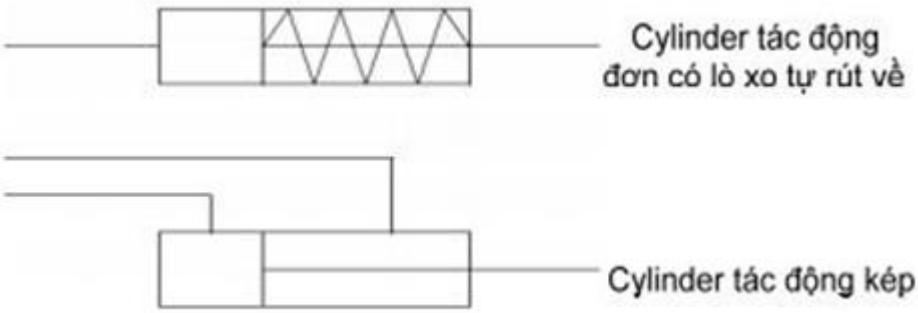
3.2.4 Xy lanh(CYLINDER)

Cylinder sử dụng áp lực khí hoặc chất lỏng để tạo lực/chuyển động tuyến tính (hình 3.19). Trong hình, dòng chất lỏng được bơm vào một phía của cylinder làm dịch chuyển piston về phía còn lại. Chất lỏng ở phía này được thoát tự do. Lực tác dụng lên cylinder tỷ lệ với diện tích bề mặt của piston.
Công thức tính lực:



Hình 3. 20: Mặt cắt của một cylinder thủy lực

Cylinder tác động đơn yêu cầu cung cấp lực khi duỗi ra và sử dụng lò xo để co về. Còn cylinder tác động kép thì cung cấp lực ở cả hai phía.



Hình 3. 21: cylinder tác động đơn và cylinder tác động kép

Các cylinder từ thường được sử dụng trong điều khiển khí nén. Trên đầu của piston có một mảnh nam châm. Khi nó di chuyển đến vị trí giới hạn thì các công tắc reed switch sẽ phát hiện ra.

3.2.5 Rơ le

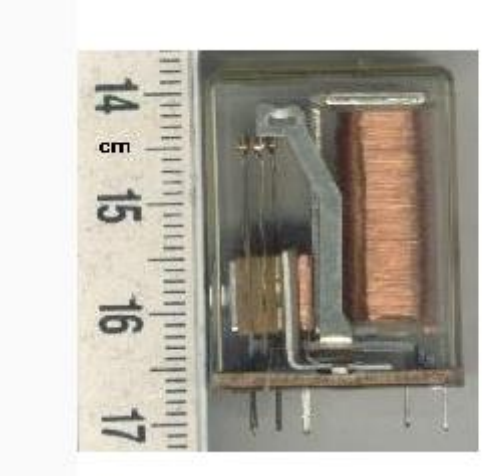
Rơ le là một trong các linh kiện điện tử thụ động rất hay thấy khi chúng ta gặp các vấn đề liên quan đến công suất cần sự ổn định cao. Rơ le được định nghĩa là một công tắc chạy bằng điện. Khác với những công tắc khác cần đến sự tác động của con người thì rơ le được kích hoạt bằng điện thay vì dùng tay người. Nhiều rơ le sử dụng một nam châm điện để vận hành cơ khí công tắc, những nguyên lí vận hành khác cũng được sử dụng chẳng hạn như rơ le trạng thái rắn.

Như ta đã biết rơ le cơ khí hiện nay khi đưa vào sử dụng bộc lộ một số hạn chế nổi bật có thể kể đến là việc phát ra tiếng ồn rất lớn và còn có những hiện tượng như tóe tia lửa điện khi các tiếp điểm đóng cắt. Để khắc phục yếu điểm trên rơ le bán dẫn ra đời với đầy đủ tính chất của rơ le và loại bỏ được những phiền toái trên. Điểm khác biệt rõ nét nhất của SSR so với rơ le thông thường là nó không có "bộ phận chuyển động" (moving part).
Rơ le bán dẫn có tên tiếng Anh là Solid State Relay và thường được viết tắt thành SSR.

Rơ le bán dẫn trọng thực tế có rất nhiều loại khác nhau từ hình dáng cho đến màu sắc nhưng cho dù nó có biến đổi như nào đi nữa thì nó vẫn có một cấu trúc chung và nguyên lí hoạt động.

- Rơ le điện từ

Rơ le chuyển trường từ của cuộn dây thành lực cơ học để mở hoặc đóng cơ khí một hoặc nhiều tiếp điểm điện.



Hình 3. 22 Hình ảnh minh họa Rơle (Relay)

Dạng cách điện này và khả năng chuyển mạch nhiều nhóm tiếp điểm bằng một tín hiệu điều khiển của rơ le, là những chức năng vô cùng hữu ích trong tự động hóa nhà máy.

Rơ le thông dụng với rất nhiều dạng, là một thiết bị độc lập có giá thành nói chung thấp, thực hiện nhiều chức năng hữu ích trong cuộc sống hàng ngày của chúng ta cũng như trong nhà máy.

Trong thế giới Tự động hóa của nhà máy, các thiết bị điều khiển công nghiệp như PLC, bộ hẹn giờ, bộ đếm và thiết bị kiểm soát nhiệt độ vận hành ở điện áp và dòng điện tương đối thấp. Thông thường nhỏ hơn 25 V. Tuy nhiên, khi tín hiệu đầu ra của các thiết bị điều khiển này được kết nỗi với thiết bị trong nhà máy, thường cần mức điện (điện áp và dòng điện) lớn hơn. Khi một thiết bị điều khiển xuất ra lệnh 24 V để bật động cơ 220 VAC, thông tin điện áp thấp này được chuyển tiếp đến một thiết bị có khả năng chuyển mạch lượng điện lớn hơn này theo lệnh.

Rơ le cơ điện có thể thực hiện chức năng này.

Hình 3. 23: Hình minh họa chức năng của Relay

Rơ le cơ điện thực hiện rất nhiều chức năng. Một số chức năng bao gồm:

- Cách ly các mạch điều khiển khỏi mạch tải hoặc mạch được cấp điện AC khỏi mạch được cấp điện DC.
- Chuyển mạch nhiều dòng điện hoặc điện áp sang các tải khác nhau sử dụng một tín hiệu điều khiển.
- Giám sát các hệ thống an toàn công nghiệp và ngắt điện cho máy móc nếu đảm bảo độ an toàn.
- Sử dụng một vài rơ le để cung cấp các chức năng logic đơn giản như 'AND,' 'NOT' hoặc 'OR' cho điều khiển tuần tự hoặc khóa liên động an toàn.

Hình 3. 24: Hình minh họa ứng dụng của Rơle (Relay) trong nhà máy

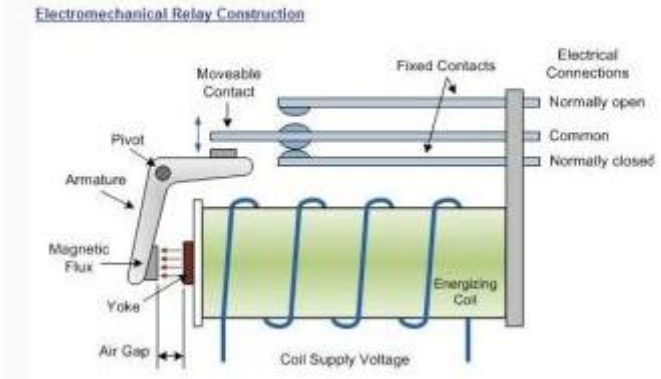
Dạng phổ biến nhất của rơ le cơ điện gồm một cuộn dây điện được cuốn trên một lõi sắt từ. Bộ phận này có cả một phần tĩnh được gọi là Ách từ (Yoke) và một phần động được gọi là Phần ứng (Armature). Phần ứng được liên kết cơ học với một tiếp điểm động.

Khi cuộn dây được cấp điện, từ trường được tạo ra xung quanh cuộn và được lõi tập trung lại Nam châm điện này hút phần ứng động để mở hoặc đóng trực tiếp các tiếp điểm điện.

Khi rơ le bị ngắt điện (TẮT) từ trường biến mất và phần ứng, được lò xo phản hồi hỗ trợ, đưa tiếp điểm trở lại vị trí "bình thường" của nó.

Khi vận hành, có 5 bước cơ bản xảy ra khi rơ le cơ điện được cấp điện và bị ngắt điện:

- Điện được cung cấp cho cuộn dây tạo ra từ trường.
- Từ trường được chuyên thành lực cơ học bằng cách hút phần ứng.
- Phần ứng động đóng/mở một hoặc nhiều tiếp điểm điện .
- Các tiếp điểm cho phép chuyển mạch điện sang tải như động cơ, bóng đèn,v.v..
- Sau khi điện áp cuộn bị loại bỏ, từ trường biến mất, các tiếp điểm tách ra và trở về vị trí "bình thường" của chúng.
- Các tiếp điểm có thể thường đóng hoặc thường mở.

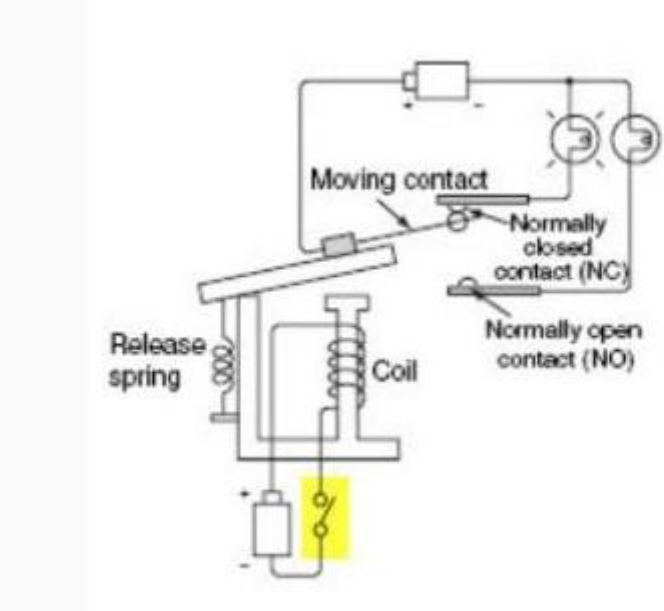


Hình 3. 25: Hình minh họa cầu tạo tiếp điểm của một Rơle Relay điển hình

Cài chốt và không cài chốt

Rơ le có thể là Cài chốt (Lưỡng ổn) hoặc Không chốt (Đơn ổn).

Phần mô tả hoạt động cơ bản của rơ le mà chúng tôi đưa ra dựa trên rơ le không chốt hay rơ le đơn ổn. Rơ le không chốt sẽ trở về trạng thái bình thường của nó khi bị ngắt điện trong khi rơ le chốt thì không.



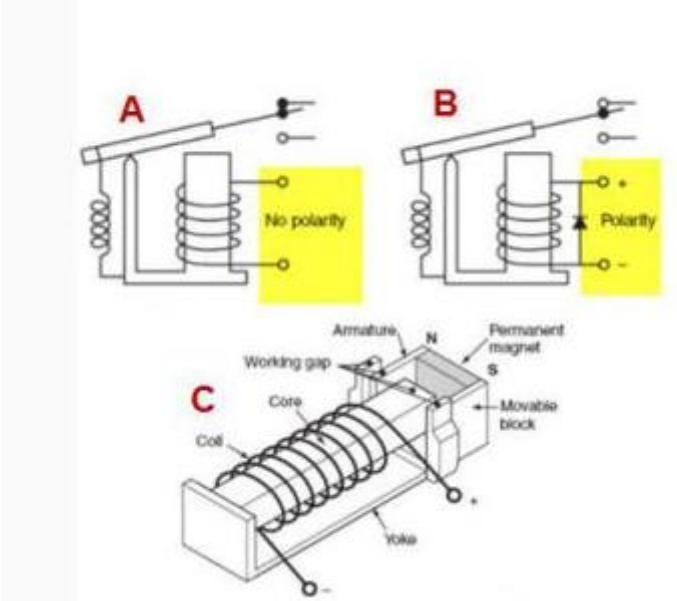
Hình 3. 26: Hình minh họa Rơle Relay có cài chốt và không có cài chốt

Cuộn dây Rơ le

Thiết kế của cuộn dây rơ le quyết định thông số kỹ thuật đầu vào của rơ le.

Rơ le được phân loại theo điện áp yêu câu để cấp điện cho cuộn dây của chúng. Nếu không đủ điện, từ trường của cuộn dây sẽ quá yếu để di chuyển tiếp điểm. Nếu quá điện đầu vào, cuộn dây có thể bị hỏng.

Một số rơ le yêu cầu Dòng điện Một chiều được cấp đúng cực và một số rơ le có thể sử dụng Dòng điện Xoay chiều hoặc Dòng điện Một chiều không phụ thuộc cực kết nối.



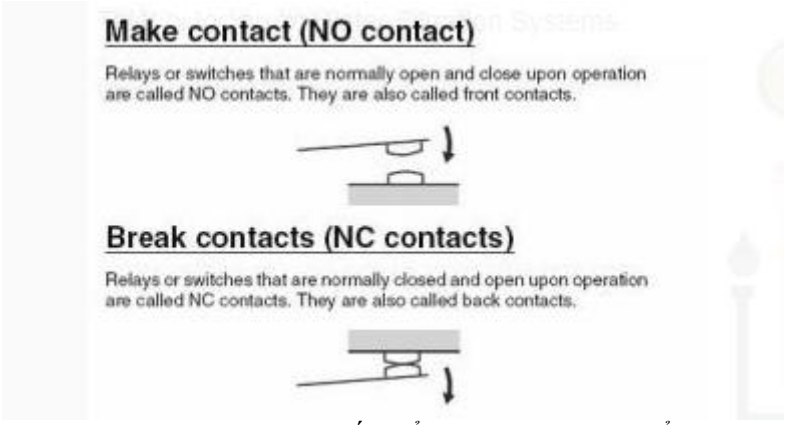
Hình 3. 27: Hình minh họa Rơle (Relay) với cầu trúc cuộn dây khác nhau

- Rơ le A sẽ hoạt động với Dòng điện Một chiều hoặc Dòng điện Xoay chiều và không yêu cầu đi dây đúng cực để rơ le hoạt động.
- Rơ le B có một đi-ốt phát sáng để biểu thị khi cuộn dây được cấp điện. Đi-ốt phát sáng được phân cực, bởi vậy mặc dù cuộn dây rơ le không có gì đặc biệt, sự có mặt của đi-ốt phát sáng yêu cầu nguồn điện đúng cực để đi-ốt hoạt động.
- Rơ le C sử dụng một nam châm vĩnh cửu để hỗ trợ lực điện từ. Cực của cuộn dây tại khe hở hoạt động là Bắc hoặc Nam tùy thuộc vào sự phân cực của cuộn dây. Vì các cực giống nhau thì đẩy và cực khác nhau thì hút, sự phân cực của cuộn dây là cần thiết để tạo ra lực thích hợp để rơ le hoạt động.

Tiếp điểm Rơ le

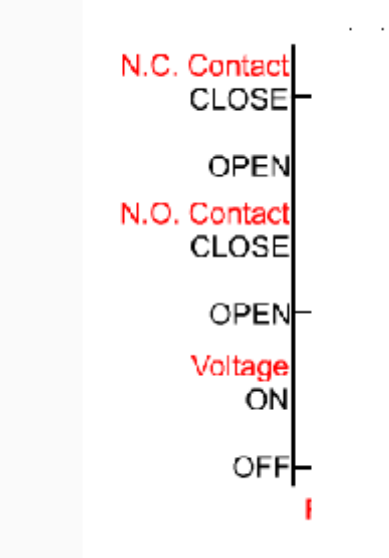
Tiếp điểm là một bộ phận quan trọng của thiết kế rơ le. Tiếp điểm rơ le chuyển mạch dòng điện và điện áp cho các loại thiết bị điện

hoặc "tải" khác nhau. Thiết kế tiếp điểm và vật liệu làm tiếp điểm là hai vấn đề trọng yếu đối với hiệu suất của rơ le.



Hình 3. 28: Hình minh họa tiếp điểm của Rơle Relay điển hình

Hoạt động của tiếp điểm nhìn có vẻ đơn giản. Chúng mở và đóng!



Hình 3. 29

Tuy nhiên, thời gian hoạt động này lại vô cùng quan trọng đối với nhiều ứng dụng và các kỹ sư không những phải dựa vào dữ liệu của nhà sản xuất mà còn phải dựa vào tính nhất quán mà mọi rơ le sẽ đạt các thông số kỹ thuật này.

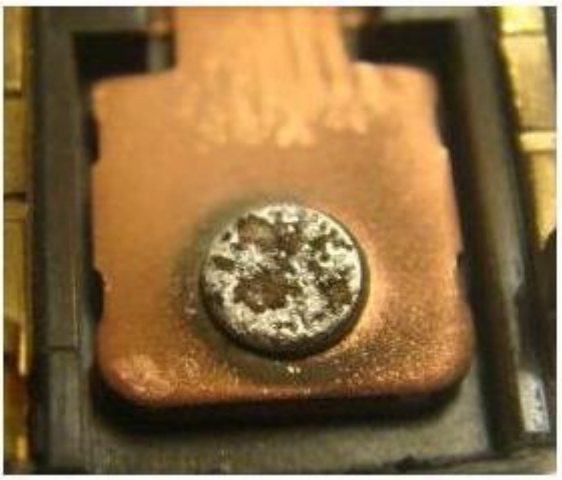
Thời gian vận hành là thời gian từ khi cấp điện cho cuộn đây đến khi tiếp điểm đầu tiên đóng, không bao gồm sự nảy tiếp điểm.

Thời gian nhả là thời gian cần đề tiếp điểm mở sau khi đã ngừng cấp điện cho cuộn dây rơ le.

Các thông số kỹ thuật khác bao gồm Điện áp Phải Vận hành và Điện áp Phải Nhả. Những thông số kỹ thuật này xác định điện áp tối thiểu cần đề đóng tiếp điểm và điện áp tối đa mà tại đó tiếp điểm sẽ mở.

Sự ăn mòn tiếp điểm

Tiếp điểm rơ le là bộ phận hoạt động nhiều nhất của rơ le và dễ bị mòn. Mỗi lần tiếp điểm của rơ le được đóng hoặc mở, có sự mài mòn nhất định ảnh hưởng đến hiệu suất của rơ le cơ điện theo thời gian.



Hình 3. 30:Hình minh họa tiếp điểm của một Relay bị ăn mòn

Mòn tiếp điểm là do:

Mài mòn cơ học

Tiếp điểm rơ le mở và đóng bằng một lực cơ học. Mặc dù lực này rất nhỏ, theo thời gian, bề mặt tiếp điểm có thể mòn đi do tiếp xúc và cọ xát lặp đi lặp lại. Mài mòn cơ học có thể chịu ảnh hưởng của:

- Vật liệu làm tiếp điểm chống mài mòn
- Vật liệu phủ bề mặt tiếp điểm
- Hình dạng tiếp điểm phù hợp
- Biên dạng chuyển động của tiếp điểm

Dòng điện mạnh

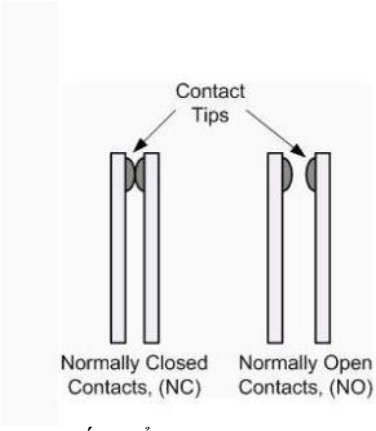
Contact Ratings

Item	2-pole	
	Resistive load (cosΦ = 1)	Inductive load (cosΦ = 0.4, L/R = 7 ms)
Rated load	5A, 250 VAC 5A, 30 VDC	2A, 250 VAC 2 A, 30 VDC
Carry current	10 A (see note)	
Max. switching voltage	250 VAC 125 VDC	
Max. switching current	10 A	
Max. switching power	2,500 VA 300 W	1,250 VA 300 W
Failure rate	5 VDC, 1 mA	

Hình 3. 31: Hình minh họa tiếp điểm của Relay bị ăn mòn do dòng điện quá mạnh

Tiếp điểm rơ le phải chịu tất cả dòng điện cần để vận hành tải. Diện tích của bề mặt tiếp xúc, điện trở suất của vật liệu làm tiếp điểm và loại tải, tất cả đều ảnh hưởng đến lượng nhiệt được tạo ra trong tiếp điểm khi tải được BẬT hoặc TẮT.

Phóng hồ quang điện



Hình 3. 32: Hình minh họa tiếp điểm của Relay bị ăn mòn do bị phóng hồ quang

Tiếp điểm rơ le là những mảnh kim loại dẫn điện tiếp xúc với nhau để tạo ra một mạch giống như công tắc.

Khi các tiếp điểm mở, điện trở giữa các tiếp điểm rất cao và không có dòng điện nào giữa chúng. Khi các tiếp điểm được đóng, điện trở tiếp điểm rất thấp.

Tất cả các tiếp điểm rơ le đều có một lượng "điện trở tiếp điểm" nhất định khi chúng được đóng và được gọi là "Điện trở khi đóng mạch". Với rơ le mới, điện trở tiếp điểm này sẽ rất thấp, thông thường nhỏ hơn $0{,}2\,\Omega$ do các đầu mới và sạch.

Sự ô-xi hoá và ăn mòn

Tiếp điểm rơ le có thể tiếp xúc với không khí và có thể bị ô xy hóa và ăn mòn.

Ví dụ: phóng hồ quang tại tiếp điểm rơ le có thể, trong điều kiện ẩm, kết hợp khí ni tơ và ô xy trong không khí tạo thành axit nitric (HNO3) ăn mòn kim loại. Trong một số môi trường công nghiệp, lưu huỳnh và clo có thể tấn công vật làm liệu tiếp điểm. Tác động tổng thể của việc hư hại bề mặt tiếp điểm là làm tăng điện trở và giảm tính dẫn điện qua tiếp điểm rơ le. Ngay cả khi được lưu trữ dài ngày mà không vận hành, điện trở của tiếp điểm có thể tăng tùy thuộc vào điều kiện môi trường.

Contact Tip Material	Characteristics
Ag (fine silver)	Electrical and thermal conductivity are the highest of all metals, exhibits low contact resistance, is inexpensive and widely used. Contacts tarnish through sulphur influence.
AgCu (silver copper)	"Hard silver", better wear resistance and less tendency to weld, but slightly higher contact resistance.
AgSnIn (silver tin indium)	Better resistance to welding than cadmium which is not RoHS compliant. Good for high inrush currents and offers long life.
AgW (silver tungsten)	Hardness and melting point are high, arc resistance is excellent. Not a precious metal. High contact pressure is required. Contact resistance is relatively high, and resistance to corrosion is poor.
AgNi (silver nickel)	Equals the electrical conductivity of silver, excellent arc resistance.
AgPd (silver palladium)	Low contact wear, greater hardness. Expensive.
platinum, gold and silver alloys	Excellent corrosion resistance, used mainly for low-current circuits.

Hình 3. 33: Bảng vật liệu làm tiếp điểm cho Relay

Vật liệu làm tiếp điểm thích hợp tăng cường hiệu suất của Rơ le cơ điện.

Lý tưởng là tiếp điểm rơ le sẽ có độ dẫn điện cao, bề mặt rất sạch mà không bị ôxi hóa, độ chống mài mòn cao và diện tích bề mặt mang điện hiệu quả. Để đáp ứng được những yêu câu này, rơ le sử dụng nhiều loại vật liệu làm tiếp điểm, chẳng hạn như những loại được liệt kê ở đây. Như bạn có thể thấy, các kim loại quý thường được sử dụng đơn lẻ hoặc kết hợp với các vật liệu khác để nâng cao hiệu suất của tiếp điểm.

Một số tiếp điểm được thiết kế có hình cầu và phần ứng kết nối với tiếp điểm được phép vượt quá vị trí. Điều này dẫn đến việc lau tiếp điểm hoặc làm sạch bề mặt của chính nó và của tiếp điểm liên hợp! Hoạt động cơ học này giúp duy trì điện trở tiếp xúc thấp theo thời gian.

Sự nảy tiếp điểm

Sự nảy tiếp điểm (còn được gọi là rung) là một đặc điểm phổ biến của cả công tắc và rơ le cơ điện Tiếp điểm rơ le thường được làm bằng kim loại đàn hồi bị ép tiếp xúc bằng bộ tác động. Khi các tiếp điểm đập vào nhau, lực xung và lực đàn hồi của chúng tác động với nhau tạo ra sự nảy (lập bập). Kết quả là một dòng điện có xung nhanh thay vì sự chuyển tiếp trơn tru từ không có điện đến toàn bộ dòng điện. Ảnh hưởng thường không quan trọng trong mạch điện, nhưng gây ra vấn đề trong một số mạch tương tự và logic phản ứng đủ nhanh để dịch sai xung bật-tắt là một luồng dữ liệu.

Cấu hình tiếp điểm

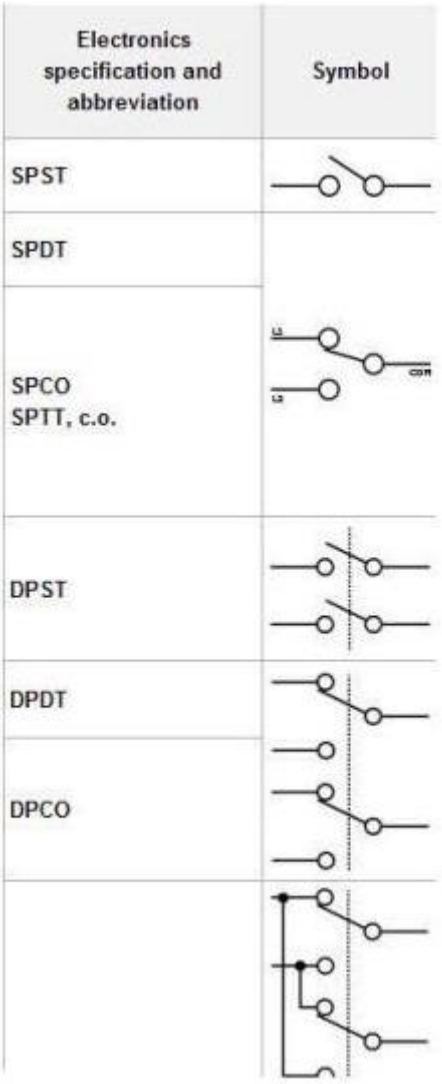
Vì rơ le là công tắc, thuật ngữ áp dụng cho công tắc cũng được áp dụng cho rơ le. Một rơ le sẽ chuyển mạch một hoặc nhiều cực, mỗi cực của những tiếp điểm này có thê được đóng mở khi cấp điện cho cuộn bằng một trong ba cách:

Tiếp điểm thường mở (NO) kết nối mạch khi rơ le được cấp điện: mạch bị ngắt kết nối khi rơ le bị ngắt điện. Tiếp điểm này còn được gọi là tiếp điểm Kiểu A hoặc tiếp điểm "đóng".

Tiếp điểm thường đóng (NC) ngắt kết nối mạch khi rơ le được cấp điện; mạch được kết nối khi rơ le bị ngắt điện. Tiếp điểm này còn được gọi là tiếp điểm Kiểu B hoặc tiếp điểm “ngắt”.

Tiếp điểm chuyển đổi (CO) hoặc hai tiếp điểm (DT) kiểm soát hai mạch: một tiếp điểm thường mở và một tiếp điểm thường đóng có một cực chung (xem ảnh). Tiếp điểm này còn được gọi là tiếp điểm Kiểu C hoặc tiếp điểm “chuyển mạch" (“ngắt rồi đóng"). Nếu loại tiếp điểm này sử dụng chức năng “ đóng rồi ngắt “ thì được gọi là tiếp điểm Kiểu D contact.

- Tiếp điểm: SPST, SPDT, SPCO, DPST, DPDT, DPCO



Hình 3. 34: Hình minh họa các loại câu hình của tiếp điểm Relay

Tải của Rơ le

Một "tải" là một thiết bị được cấp điện bởi mạch điện. Việc chọn rơ le và hiệu suất của rơ le qua thời gian chịu ảnh hưởng của loại tải được kết nối với rơ le.

Tải thường được phân loại là yêu cầu năng lượng Dòng điện Xoay chiều hoặc Dòng điện Một chiều và được phân loại chi tiết hơn theo bản chất chủ yếu là Điện, Điện dung hoặc Cảm ứng. Mỗi loại tải có những nhu cầu độc nhất với các tiếp điểm rơ le. Chọn sai loại rơ le hoặc vận hành rơ le vượt ra ngoài thông số kỹ thuật cho một loại tải cho trước, có thể làm giảm đáng kể tuổi thọ của rơ le.

Dưới đây là các loại tải

Tải điện trở

Một Tải điện trở là một tải mà dòng điện ổn định theo thời gian (mạch Dòng điện Một chiều) và theo pha với điện áp (mạch Dòng điện Xoay chiều).

Một tải thuần điện trở cần cùng một lượng điện để BẬT cũng như lượng điện để duy trì hoạt động.

Đối với thiết bị được cấp điện AC (Dòng điện Xoay chiều), dòng điện và điện áp luôn tăng và giảm củng nhau, theo bước hoặc "theo pha”

Các ví dụ về tải điện trở bao gồm lò nướng bánh bằng điện, máy pha cà phê hoặc thiết bị nhiệt bằng điện trở công nghiệp.

Hình 3. 35: Hình minh họa ký hiệu tải điện trở của Relay

Tải cảm ứng

Tải cảm ứng là tải khiến dòng điện trễ ngoài thay đỗi về điện áp và đối với mạch Dòng điện Xoay chiều là lệch pha.

Dòng điện trong tải cảm ứng thường đi qua một số dạng cuộn dây. Thuộc tính điện của cuộn dây là kháng lại thay đổi về dòng điện. Tải cảm ứng có thể cần nhiều điện đề bắt đầu hoạt động hơn là duy trì hoạt động.

Các ví dụ về tải cảm ứng bao gồm tủ lạnh, quạt điện, động cơ công nghiệp, solenoid và thậm chí là các cuộn dây rơ le khác!

Tải cảm ứng rất nặng cho tiếp điểm rơ le! Sức phản Điện động hoặc “phản cảm ứng" phổ biến khi chuyển mạch các tải cảm ứng và gây ra sự phóng hồ quang tiếp điểm đáng kể.

Hình 3. 36: Hình minh họa về ký hiệu tải cảm cứng của Relay

Điều gì gây ra Sức phản Điện động?

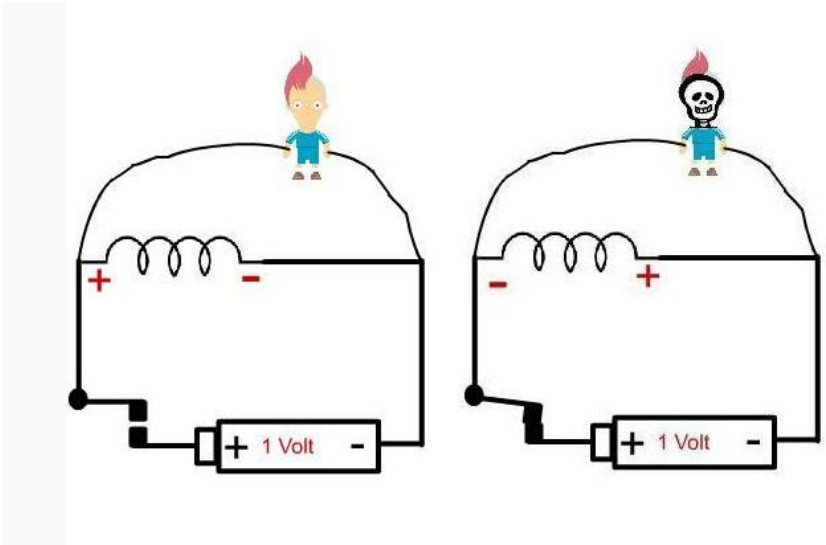
Khi cuộn dây được cấp điện. giống như cuộn dây rơ le, trường điện từ được tạo ra xung quanh cuộn dây và năng lượng được “lưu trữ” trong trường điện từ này.

Khi cuộn dây bị ngắt điện, trường điện từ này biến mất, cắt qua các vòng của cuộn dây và tạo ra điện áp trong cuộn dây của cực đối lập.

Vì trường biến mất này đi qua rất nhiều vòng dây, mỗi vòng lại thêm điện áp vào tổng. Đối với cuộn dây có nhiều vòng, điện áp này có thê hơn 1000 vôn!

Khi rơ le chuyển mạch các tải cảm ứng, sức phản điện động sẽ tạo ra sự mài mòn tiếp điểm đáng kể vì điện áp cao hơn này nhảy vọt qua khoảng cách tiếp điểm khi tiếp điểm mở.

Các vật liệu làm tiếp điểm đặc biệt và kỹ thuật triệt hỗ quang được sử dụng đề bảo vệ các tiếp điểm rơ le chuyên mạch tải cảm ứng.



Hình 3. 37

Tải Điện dung

Một Tải điện dung là tải khiến dòng điện dẫn đầu các thay đôi về điện áp và đối với mạch Dòng điện Xoay chiều là lệch pha.

Tải điện dung gây ra dòng điện lớn trong vài mili giây đầu tiên sau khi được BẬT. Dòng điện kích từ lớn này là một đặc điểm phổ biến của tải điện dung.

Các ví dụ về tải điện dung bao gồm các bộ nguồn có lọc tốt, tụ điện khởi động động cơ, tụ trữ năng lượng, v.v.

Hình 3. 38: Hình minh họa ký hiệu tải điện dung của Relay

Các thiết bị điện thường có đặc điểm của tất cả ba loại tải này! Tuy nhiên, thông thường một đặc điểm, Điện trở, Cảm ứng hoặc Điện dung sẽ vượt trội hơn. Tiếp điểm rơ le sẽ chủ yếu cảm nhận được sự thay đổi điện áp và dòng điện theo loại tải chủ yếu này khi chúng mở và đóng.

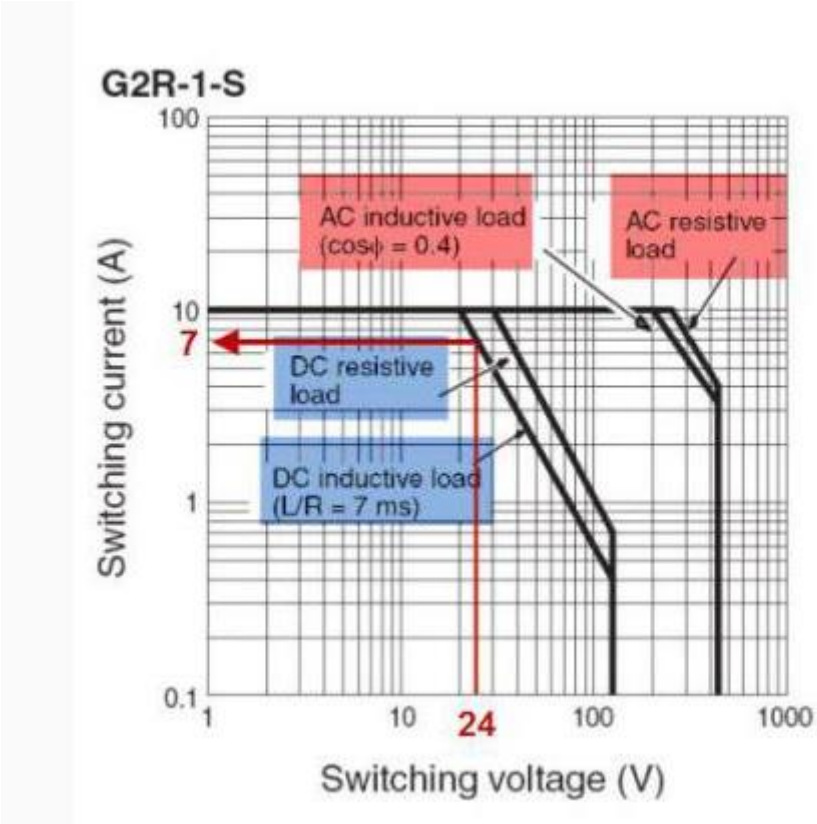
Bộ phận bảo vệ tiếp điểm

Trong những năm qua, các kỹ sư đã phát minh ra rất nhiều cách để giúp bảo vệ tiếp điểm khỏi sự tăng đột biến dòng điện hoặc sự phóng hồ quang ngoài việc chọn vật liệu làm tiếp điểm phù hợp.

Trong mạch lân cận, năng lượng sẽ gây ra hồ quang được hấp thụ bởi mạch RC cùng với tiếp điểm rơ le. Điều này giúp giảm thiểu hồ quang và sự hư hại cho tiếp điểm.

Phải xem xét các chỉ tiêu thiết kế điện nhất định khi chọn điện trở và bộ tụ điện để tối ưu hóa sự bảo vệ.

Điện áp và dòng điện



Hình 3. 39: Hình minh họa điện áp và dòng điện qua Relay

Mức dòng điện và điện áp mà rơ le có thể chuyển mạch an toàn được biểu thị cả trong bảng và trong biểu đồ.

Biểu đồ Chuyển mạch Tối đa này dành cho rơ le G2R-1-S biểu thị giới hạn của tiếp điểm cho cả Dòng điện Xoay chiều và Dòng điện Một chiều và tải điện trở cùng với tải cảm ứng.

Mặc dù công suất chuyển mạch dòng điện của rơ le này là 10A cho cả tải điện trở và cảm ứng ở Dòng điện Một chiều 10V, chúng ta muốn kiểm tra công suất dòng điện tại Dòng điện Một chiều 24V.

Theo đường màu đỏ từ 24V lên tới đường tải Cảm ứng Dòng điện Một chiều và chuyển sang trái từ điểm giao cắt, chúng ta thấy khả năng tiếp điểm được giảm xuống 7A tại mức điện áp cao hơn này.

Bạn phải luôn tham khảo bản dữ liệu sản phẩm để biết chi tiết về việc cài đặt, sử dụng và các giới hạn điện phù hợp của rơ le để đảm bảo an toàn và tuổi thọ hoạt động cao.

Độ bền của Rơ le

Độ bền của rơ le liên quan đến cả loại tải và dòng điện được các tiếp điểm của nó chuyển mạch.

Tại đây, chúng ta có thể xem độ bền của rơ le đó tăng đối với:

Tải điện trở

Chuyển mạch điện áp thấp hơn

Chuyển mạch dòng điện thắp hơn

Ví dụ khi chuyển mạch tải điện trở Dòng điện 4A, 30VDC độ bền dự kiến là 350.000 lần vận hành.

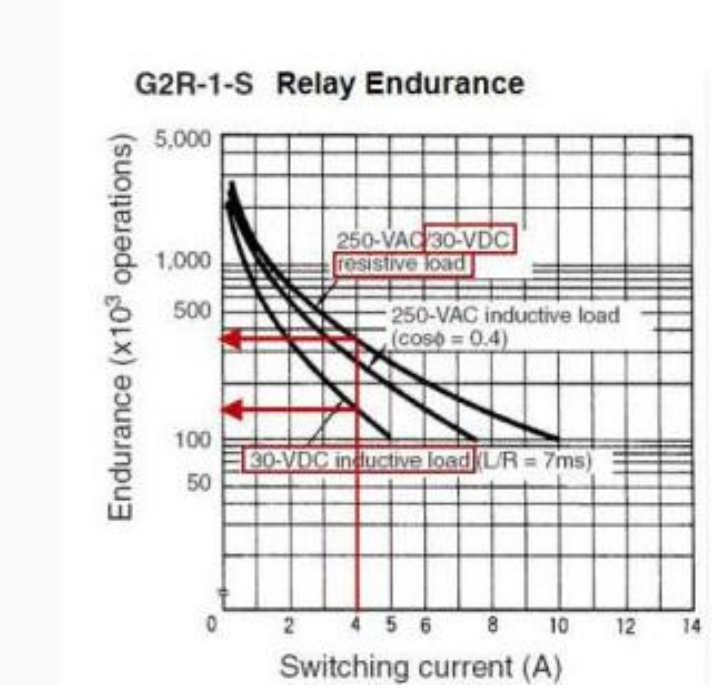
Chúng ta cũng có thê thấy rằng độ bên của rơ le giảm đối với:

Tải cảm ứng

Điện áp cao hơn

Dòng điện mạnh hơn

Khi cùng một Dòng điện Một chiều 4A, 30VDC vận hành một tải cảm ứng, số lần vận hành dự kiến giảm hơn 50% còn 150.000



Hình 3. 40: Hình minh họa Relay endurance

- Rơ le phi tiếp điểm(rơ le bán dẫn)

SSR còn được gọi bằng cái tên relay bán dẫn SSR. Đây là một loại relay vô cùng phổ biến. Vậy relay SSR có nguyên lý hoạt động ra sao, cấu tạo thế nào và cách thức sử dụng.

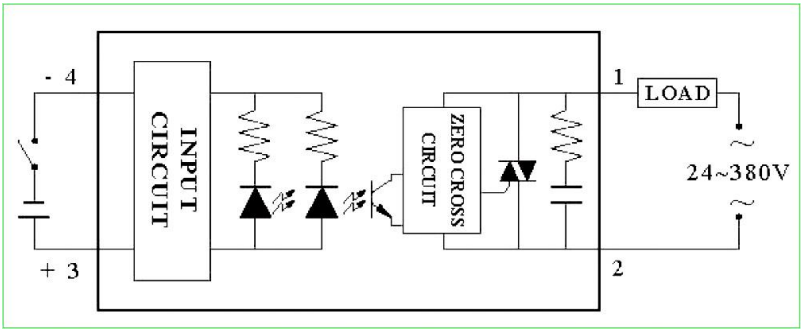
Hình 3. 41: Hình minh họa Rơle Relay SSR (Solid State Relay)

SSR là một cụm từ viết tắt của Solid state relay. Đây là một loại relay trạng thái rắn được sử dụng rất phổ biến. SSR là rơ le chuyển mạch không yêu cầu sử dụng bất kỳ bộ phận cơ khí nào. Điều này thường mang lại cho chúng lợi thế về tuổi thọ dài hơn so với rơ le điện cơ thông thường, và mặc dù rơ le trạng thái rắn có cường độ nhanh hơn so với rơ le điện cơ, nhưng chúng có một số quy định thiết kế.

Rơ le trạng thái rắn đã gây bão trên toàn thế giới, tạo ra một cuộc cách mạng phân phối điện trong mọi ngành công nghiệp từ tự động hóa nông nghiệp đến hàng không vũ trụ.

Rơ le trạng thái rắn có những ưu điểm như sau so với rơ le điện từ: độ tin cậy cao, không tiếp xúc, không có tia lửa, tuổi thọ cao, tốc độ chuyển mạch nhanh, khả năng chống nhiễu mạnh và kích thước nhỏ. Nó đã được sử dụng rộng rãi trong các ứng dụng rộng rãi như máy CNC, hệ thống điều khiển từ xa và các thiết bị tự động hóa công nghiệp, công nghiệp hóa chất, thiết bị y tế, hệ thống an ninh, v.v.

Nguyên lý hoạt động của SSR



Hình 3. 42: Nguyên lý hoạt động của SSR

Bạn có thể tự hỏi - loại công tắc nào cho phép tín hiệu điều khiển cung cấp năng lượng cho hàng trăm am pe? Vẻ đẹp thực sự của rơ le SSR so với rơ le điện cơ cuối cùng nằm ở sự khác biệt giữa các cơ chế chuyển mạch. Rơ le trạng thái rắn sử dụng cái mà ngành công nghiệp gọi là bộ cách ly quang hoặc bộ ghép quang.

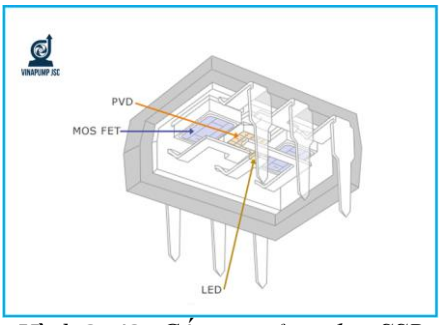
Nói theo cách nói của con người, điều đó có nghĩa là một bộ tách ánh sáng của người. Đó là quyền - công tắc bên trong rơ le trạng thái rắn chỉ là một chùm ánh sáng! Nói chung, có một đèn LED công suất rất thấp chiếu một chùm ánh sáng vào photodiode, điều này gần như ngay lập tức cho phép năng lượng được truyền qua nó - hoặc bật chế độ khởi động.

Bộ cách ly quang là rất quan trọng trong rơ le trạng thái rắn vì chúng là những gì tách rời hai hoặc nhiều mạch của rơ le. Vì rơ le sử dụng tín hiệu điện áp nhỏ để điều khiển tín hiệu điện áp rất lớn, điều cực kỳ quan trọng là phải tách các tín hiệu này.

Vẻ đẹp và đặc điểm cách mạng của các chất cô lập quang là không có bộ phận chuyển động. Ví dụ, trong các rơ le điện cơ, sự phân tách mạch này được thực hiện bởi một từ trường điện, cũng là thứ được sử dụng để hoàn thành cuối cùng của mạch tải lớn.

Trong rơ le trạng thái rắn, photodiode là thứ làm cho kết nối trong mạch tải hoàn thành. Vì vậy, những gì trên thế giới là một photodiode? Nó là một bóng bán dẫn rất chuyên dụng sử dụng các photon để cung cấp năng lượng cho cổng, chứ không phải là tín hiệu điện thông thường. Làm thế nào trên thế giới làm việc đó? Nó sử dụng một ngã ba silicon P-N chuyên dụng cao.

Cấu tạo của relay SSR



Hình 3. 43: Cấu tạo của relay SSR

Rơ le SSR thường được thiết kế như một công tắc bật tắt đơn giản với đầu cực nguồn và đầu cực tải chuyển đổi khi tín hiệu điều khiển bên ngoài được chuyển đến rơ le qua một đầu cực khác. Khi điều này xảy ra, việc chuyển đổi xảy ra rất nhanh và tải được cấp nguồn, thường là bằng bóng bán dẫn công suất aMOSFET.

Rơ le có thể được thiết kế và sử dụng trong khả năng chuyển đổi AC hoặc DC, nhưng cấu hình bên trong phải được sửa đổi để hoạt động cho cả hai trường hợp. Rơ le DC có thể hoạt động với một MOSFET duy nhất, với nguồn và cống được kết nối với nguồn và tải của mạch chính và tín hiệu điều khiển được gắn vào cổng thông qua.

Tín hiệu điều khiển có thể có công suất rất thấp, cho phép rơ le (và mạch tải lớn) được điều khiển bởi một thứ nhỏ như Arduino. Rơ le trạng thái rắn có thể có nhiều bóng bán dẫn được xếp song song để cho phép tiềm năng dòng điện cao hơn, có thể được đánh giá vào 100 ampe.

Công tắc AC yêu cầu ít nhất hai bóng bán dẫn vì một MOSFET không thể ức chế dòng điện theo cả hai hướng khi rơ le ở trạng thái tắt. Hai bóng bán dẫn, với các nguồn được kết nối, được sử dụng để chặn dòng điện khi tắt và sau đó truyền nguồn khi tín hiệu điều khiển được bật trong rơ le.

Ưu điểm nổi bật :

- Thiết kế nhỏ gọn, dễ dàng lắp đặt vào tủ điện vị trí hẹp.
- Độ bền và tuổi thọ cao.
- Khả năng đóng ngắt rơ le luôn tục, độ ổn định cao.
- Khi đóng ngắt không phát ra âm thanh và tia lửa điện.
- Tín hiệu đầu vào đa dạng.

Nhược điểm :

- Cần tản nhiệt tốt cho relay bán dẫn khi làm việc với tải lớn
- Tín hiệu đầu vào đa dạng. Yêu cầu kỹ thuật phải có hiểu biết về sản phẩm trước khi lắp đặt.
- Có thể xảy ra hiện tượng dò điện và chết chập

Ứng dụng của SSR.

Relay bán dẫn là thiết bị không thể thiếu trong việc gia nhiệt các máy ép nhựa. Vì các điện trở gia nhiệt dùng cho máy ép nhựa có công suất khá lớn. Dòng tải vài chục đến hàng trăm Ampe (A) nguồn dùng 220/380V.

SSR bán dẫn thường dùng trong các nhà máy nhựa, lò hơi dùng điện trở gia nhiệt, máy sản xuất liên quan đến nhựa cần gia nhiệt các loại.

3.2.5 Động cơ

Động cơ là cơ cấu chấp hành thông thường, nhưng đối với ứng dụng cho điều khiển nhị phân thì đặc điểm của nó không quan trọng. Điều khiển logic tiêu biểu của các động cơ là đóng cắt điện cho nó. Các động cơ có dòng điện nhỏ có thể đấu trực tiếp vào các ngõ ra của PLC, còn đối với các động cơ công suất lớn thì sử dụng relay hay contactor hoặc bộ khởi động động cơ. Các động cơ sẽ được khảo sát chi tiết hơn ở *chương các cảm biến và cơ cấu chấp hành analog (tập 2)*.

3.2.6 Các cơ cấu chấp hành khác

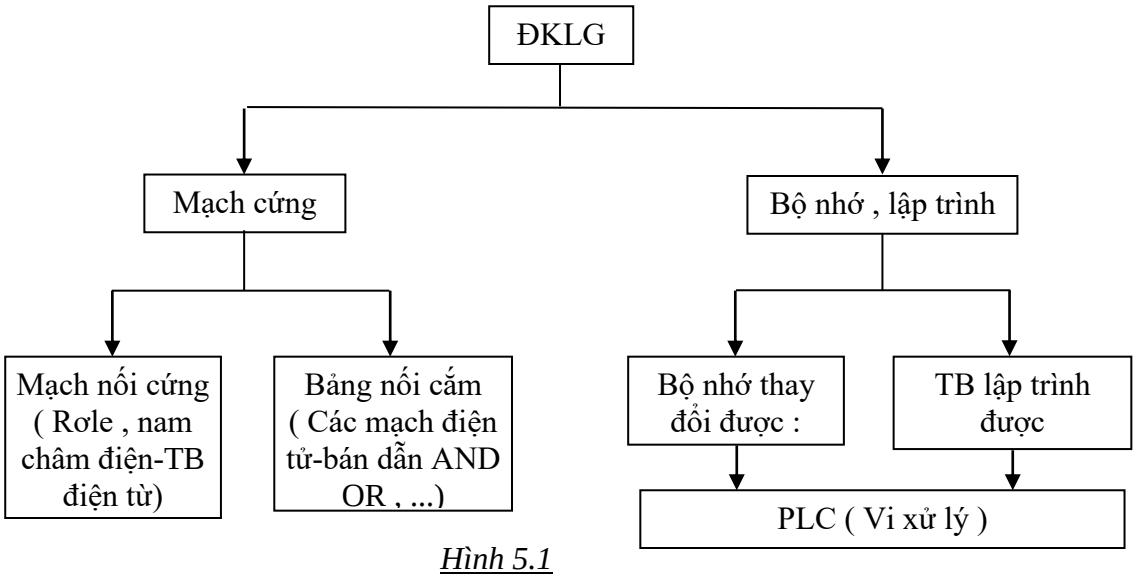
Ngoài các cơ cấu chấp hành kể trên còn có nhiều loại cơ cấu chấp hành khác nhau trong điều khiển logic. Một số cơ cấu chấp hành thường được sử dụng relay và contactor.
Ngoài ra có một số cơ cấu chấp hành khác:

- *Lò nhiệt:* Thường được điều khiển bằng relay, đóng và cắt điện để giữ nhiệt độ nằm trong một phạm vi nào đó.
- *Đèn báo:* Đèn báo được sử dụng cho hầu hết các máy móc để chỉ báo trạng thái máy và cung cấp thông tin cho người vận hành. Hầu hết các đèn báo có dòng điện thấp và được kết nối trực tiếp đến PLC.
- *Còi/chuông báo:* Còi hay chuông báo có thể được sử dụng cho các máy móc không được giám sát hoặc đang bị nguy hiểm. Chúng thường được nối trực tiếp với các ngõ ra của PLC.

CHƯƠNG 4: KHÁI NIỆM CHUNG VỀ PLC

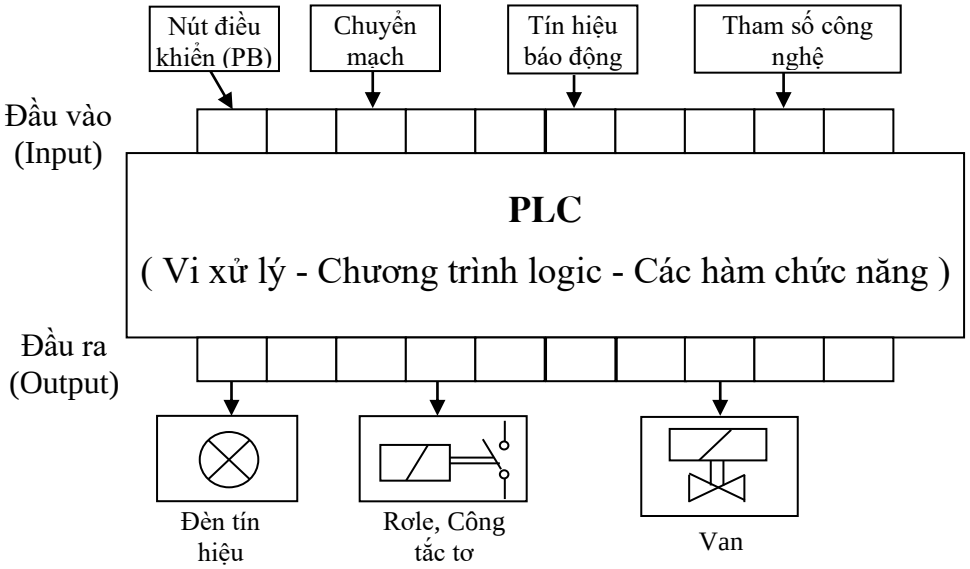
4.1. PHÂN LOẠI CÁC THIẾT BỊ ĐIỀU KHIỂN LOGIC

Các hệ thống điều khiển logic có thể được phân theo sơ đồ sau (hình 5.1) :



Hình 5.1

5.2. SƠ ĐỒ CẤU TRÚC PLC



Hình 5-2

Sơ đồ cấu trúc một PLC có dạng như hình 5.2. Phần cơ bản của PLC là hệ vi xử lý với một bộ xử lý trung tâm (CPU), cùng với các bộ nhớ, các thiết bị ghép nối vào ra, biên dịch chương trình điều khiển. Bên ngoài PLC có một bộ các đầu vào (input) và một bộ các đầu ra (output) để ghép nối với các thiết bị điều khiển, nhận các thông tin điều khiển và cho ra các lệnh điều khiển hệ thống. Các tín hiệu vào, ra của PLC ngoài dạng tín hiệu lôgic (tín hiệu số - digital) thì trong các PLC hiện nay thường có cả tín hiệu vào và ra dạng tương tự (analog). Các tín hiệu vào dạng digital

của PLC có thể tín hiệu từ các nút ấn điều khiển; tín hiệu từ các công tắc hành trình, cực hạn; tín hiệu từ các thiết bị báo động, v.v... . Các tín hiệu tương tự đưa đến đầu vào PLC có thể là các tín hiệu về các tham số công nghệ; v.v... Tín hiệu ra của PLC thường được dùng để khống chế các rơ le và cũng có thể là một tín hiệu tương tự để điều khiển một hệ thống liên tục nào đó.

4.3. CẤU HÌNH VÀO / RA (I/O) CỦA PLC

4.3.1. Giới thiệu chung

Cấu hình vào ra của một PLC là số các đầu vào và đầu ra của một PLC; cách bố trí các đầu vào; đầu ra; số các bộ timers (bộ thời gian); số các bộ counters (bộ đếm); dạng tín hiệu ra: điện áp hay dòng điện và mức tương ứng của chúng; các mođul khai triển (modul mở rộng).

Ví dụ với PLC S7-200 của hãng Siemens có dạng như sau:

Trong PLC lại có thể bố trí nhiều loại CPU, tương ứng với mỗi loại CPU có số lượng đầu vào và đầu ra khác nhau. Trong PLC S7-200 có 2 loại CPU là: CPU212 và CPU214. Các đầu vào dạng giắc cắm có công tắc mô phỏng hoá, các đầu ra dạng các đầu cốt .

	CPU 212	CPU 214
Số đầu vào số (DI)	8	14
Số đầu ra số (DO)	6	10
Số timers	64	128
Độ phân giải	1ms :2 ; 10 ms : 8 100ms : 54	1ms :4 ; 10 ms : 16 100ms : 108
Số counters	64	128
Mức điện áp vào (mức logic 1)	24 V / DC và 120 V / AC	24 V / DC và 120 V / AC
Mức tín hiệu ra (mức logic 1)	24 ÷ 230 V / AC Relay 2A	24 ÷ 230 V / AC Relay 2A
Thời gian không mất dữ liệu bộ nhớ khi mất nguồn nuôi	50 giờ	190 giờ
Số modul mở rộng nhiều nhất	2	7
Tổng số cổng vào/ra cực đại	64/64	64/64

Các modul mở rộng có thể lắp vào PLC S7-200

EM 221 có 8 DI 24 V / DC và 8 DI 120 V / AC

EM 222 có 8 DO 24 V / DC , 8 DO 24 ÷ 230 V / AC và 8 DO Relay

EM 235 có 3 AI (12 bit) và 1 AO (12 bit)

<u>Các đèn báo trên PLC S7-200 , CPU 214 :</u>

SF (đèn đỏ): Đèn đỏ SF báo hiệu hệ thống bị hỏng. SF sáng khi PLC có hỏng hóc

RUN (đèn xanh): Đèn xanh RUN chỉ thị PLC đang làm việcvà thực hiện chương trình đã được nạp vào bên trong PLC.

STOP (đèn vàng): Đèn vàng STOP sáng chỉ thị PLC đang ở chế độ dừng. Dừng thực hiện chương trình.

Ix.x (đèn xanh): Đèn xanh ở cổng vào chỉ thị trạng thái tức thời của mỗi cổng vào. Đèn ở cổng vào Ix.x sáng báo mức logic ở cổng là mức 1, đèn tắt chỉ thị mức logic ở cổng là mức 0.

Qx.x (đèn xanh): Đèn xanh ở cổng vào chỉ thị trạng thái tức thời của mỗi cổng ra. Đèn ở cổng ra Qx.x sáng báo mức logic ở cổng là mức 1, đèn tắt chỉ thị mức logic ở cổng là mức 0.

PLC S7-200 sử dụng cổng truyền thông nối tiếp RS-485 với phích nối 9 chân để ghép nối với máy lập trình (PPI) hoặc PLC khác.

Để ghép nối PLC S7-200 với máy lập trình thuộc họ PG7xx có thể sử dụng cáp nối thẳng qua MPI . Cáp đó đi kèm với máy lập trình.

Để ghép nối PLC S7-200 với máy tính PC qua cổng RS-232 cần có cáp nối PC/PPI với bộ chuyển đổi RS-232/RS-485.

Công tắc chọn chế độ làm việc của PLC S7-200 :

Công tắc chọn chế độ làm việc nằm ở phía trên, bên cạnh các đầu ra của PLC, có ba vị trí tương ứng với 3 chế độ của PLC :

RUN cho phép PLC thực hiện chương trình đã nạp vào trong bộ nhớ. PLC sẽ rời khỏi chế độ RUN và chuyển sang chế độ STOP nếu trong máy có sự cố hoặc trong chương trình gặp lệnh STOP.

STOP cưỡng bức PLC dừng công việc thực hiện chương trình đang chạy và chuyển sang chế độ nghỉ (STOP). Ở chế độ STOP PLC cho phép hiệu chỉnh lại chương trình hoặc nạp một chương trình mới .

TERM cho phép máy lập trình tự quyết định chế độ làm việc cho PLC hoặc ở chế độ RUN hoặc ở chế độ STOP.

Chỉnh định tương tự :

Điều chỉnh tương tự (1 bộ trong CPU 212 và 2 bộ trong CPU 214) cho phép điều chỉnh phép điều chỉnh các biến cần phải thay đổi và sử dụng trong chương trình. Núm điều chỉnh analog được lắp đặt dưới nắp đậy bên cạnh các cổng ra . Thiết bị chỉnh định có thể quay 270^0 .

Pin và nguồn nuôi :

Nguồn nuôi dùng để ghi hoặc nạp một chương trình mới và cung cấp điện năng cho PLC hoạt động. Trong PLC còn bố trí nguồn pin, pin được sử dụng để lưu giữ tín hiệu khi mất hoặc cắt nguồn nuôi. Pin sẽ tự động chuyển sang trạng thái tích cực nếu năng lượng tích luỹ trong tụ nhớ bị cạn và nó thay thế vào vị trí đó để dữ liệu trong bộ nhớ không bị mất .

4.4. ĐỊA CHỈ VÀO RA

Mỗi một đầu vào hoặc đầu ra của PLC đều được gán cho một địa chỉ nhất định để phân biệt với nhau. Địa chỉ vào ra thường gồm hai phần: phần chữ và phần số - phần chữ để phân biệt đầu vào với đầu ra và với các biến trung gian khác; phần số để phân biệt các cổng cùng loại với nhau.

4.4.1. Địa chỉ vào ra của các PLC hãng Siemens (Đức)

Địa chỉ vào ra các PLC hãng Siemens của Đức qui ước phần chữ là:

- Vào số – I

- Ra số - Q

Phần số trong địa chỉ vào/ra số được phân thành nhóm, mỗi nhóm 8 bit, đánh số từ 0 đến 7. Cụ thể đầu vào tín hiệu số của PLC S7-200 : I0.0, I0.1, ... , I7.7 và đầu ra tín hiệu số là: Q0.0, Q0.1, ... , Q7.7.

- Vào tương tự - AIW
- Ra tương tự - AQW

Phần số trong địa chỉ vào/ra tương tự lấy các giá trị 0, 2, 4, 6, 8, 10, 12, ...

Ví dụ có modul mở rộng là modul 2 có 3 đầu vào tương tự và một đầu ra tương tự thì địa chỉ của chúng sẽ là: AIW0, AIW2, AIW4 và AQW0.

Chú ý: Người ta ngầm định mỗi modul mở rộng tương tự có 4 cổng cho nên modul mở rộng tương tự thứ nhất phần số sẽ là 0 , 2 , 4 , 6 nên modul mở rộng tương tự tiếp theo địa chỉ phần số phải là từ 8 ÷ 14 , ...

4.4.2. Địa chỉ vào ra của PLC hãng Hitachi (Nhật bản)

Địa chỉ vào ra các PLC hãng Hitachi và một số hãng khác của Nhật qui ước phần chữ là:

- Vào số - X
- Ra số - Y

Phần số trong địa chỉ vào/ra số gồm ba con số viết theo hệ cơ 8. Ví dụ :

Đầu vào: X000 ÷ X007, X010 ÷ X017, ...

Đầu ra : Y030 ÷ Y037, ... , Y430 ÷ Y437, Y530 ÷ Y537, ...

4.4.3. Địa chỉ vào ra của PLC hãng General Electrics (GE - Mỹ)

Địa chỉ vào ra các PLC hãng General Electrics (Mỹ) qui ước phần chữ là:

- Vào số : %I
- Ra số : %Q

Phần số trong địa chỉ vào/ra số được biểu diễn bởi 4 con số theo hệ thập phân. Cụ thể đầu vào tín hiệu số của PLC loại này là: %I0000, %I0001, ... , %I0010, ..., và đầu ra tín hiệu số là: %Q0000, %Q0001, ..., %Q0010, ...

Dưới đây là ví dụ về địa chỉ vào/ra trên PLC S7-200 và cách đặt địa chỉ cho các modul mở rộng trên CPU 214 :

CPU 214		Modul 0 (4 vào số/ 4 ra số)	Modul 1 (8 vào số)	Modul 2 (3 vào analog/ 1 ra analog)	Modul 3 (8 ra số)	Modul 4 (3 vào analog/ 1 ra analog)
I0.0	Q0.0	I2.0	I3.0	AIW0	Q3.0	AIW8
I0.1	Q0.1	I2.1	I3.1	AIW2	Q3.1	AIW10
I0.2	Q0.2	I2.2	I3.2	AIW4	Q3.2	AIW12
I0.3	Q0.3	I2.3	I3.3		Q3.3	
I0.4	Q0.4		I3.4	AQW0	Q3.4	AQW4
I0.5	Q0.5	Q2.0	I3.5		Q3.5	

I0.6	Q0.6	Q2.1	I3.6		Q3.6	
I0.7	Q0.7	Q2.2	I3.7		Q3.7	
I1.0	Q1.0	Q2.3				
I1.1	Q1.1					
I1.2						
I1.3						
I1.4						
I1.5						

4.5. ĐỊA CHỈ CÁC BIẾN TRUNG GIAN VÀ BIẾN KHÁC

Tuỳ theo từng hãng sản xuất mà trong PLC có các biến số trung gian , thực chất đây là các ô nhớ làm nhiệm vụ lưu giữ các kết quả của các phép xử lý logic ở các giai đoạn trung gian. Trong PLC hãng Siemens có biến nhớ ký hiệu bằng chữ M kèm theo phần số tương tự như các biến vào, ra. Trong PLC của Nhật còn có thêm biến trạng thái ký hiệu bằng chữ S (State), địa chỉ các biến trạng thái là S600 ÷ S647 gồm 40 trạng thái .

Các bộ thời gian có địa chỉ gồm phần chữ là T, với S7-200 thì địa chỉ các timers là T0 ÷ T63 với CPU 212 và T0 ÷ T127 với CPU 214.

Các bộ đếm (counters) trong S7-200 có địa chỉ phần chữ là C, cũng với phần số tương tự các timers: C0 ÷ C63 với CPU 212 và C0 ÷ C127 với CPU 214.

CHƯƠNG 5: LẬP TRÌNH CHO PLC S7-200

5.1. CẤU TRÚC CHƯƠNG TRÌNH CỦA S7-200

Có thể lập trình cho PLC S7-200 bằng cách sử dụng một trong những phần mềm sau:

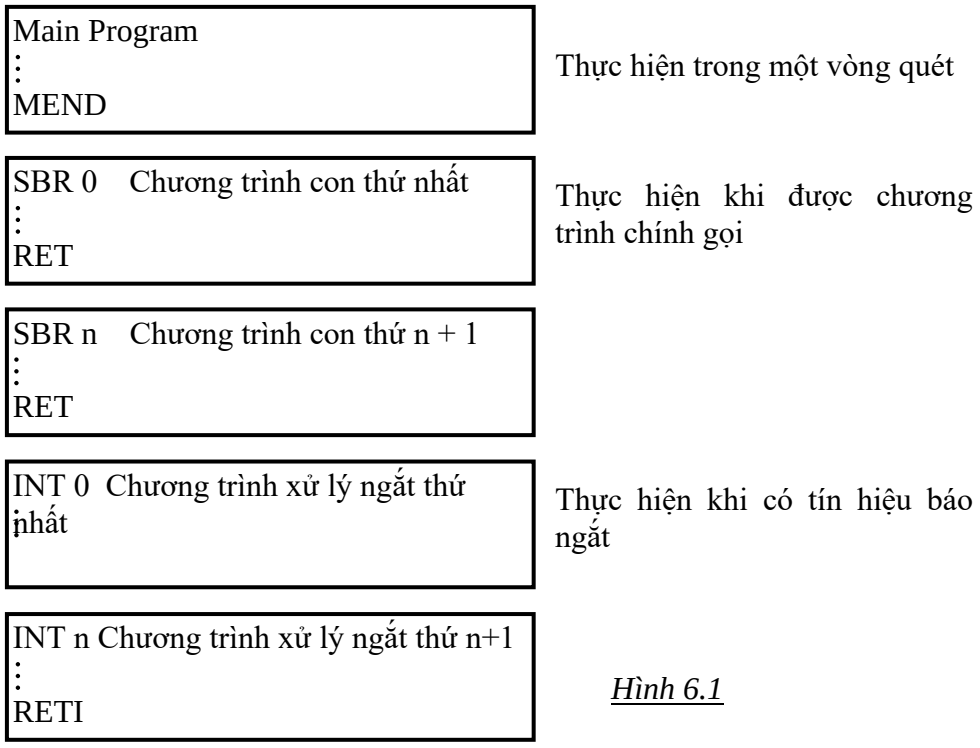
- STEP7 - Micro/DOS
- STEP7 - Micro/WIN

Những phần mềm này có thể cài đặt trên các máy lập trình họ PG7xx và các máy tính cá nhân (PC) .

Các chương trình cho S7-200 phải có cấu trúc bao gồm chương trình chính (main program) và sau đó đến các chương trình con và các chương trình xử lý ngắt (nếu có) như sau :

- Chương trình chính được kết thúc bằng lệnh kết thúc chương trình (MEND) .
- Chương trình con là một bộ phận của chương trình. Các chương trình con phải được viết sau lệnh kết thúc chương trình chính, tức là sau lệnh MEND.
- Các chương trình xử lý ngắt là một bộ phận của chương trình. Nếu cần sử dụng các chương trình xử lý ngắt thì chúng cũng được viết sau lệnh kết thúc chương trình chính MEND.

Các chương trình con được nhóm lại thành một nhóm ngaysau chương trình chính. Sau đó đến chương trình xử lý ngắt. Với kết cấu như vậy, chương trình được rõ ràng và thuận tiện hơn trong việc đọc chương trình sau này. Có thể tự do trộn lẫn các chương trình con với chương trình xử lý ngắt đằng sau chương trình chính.



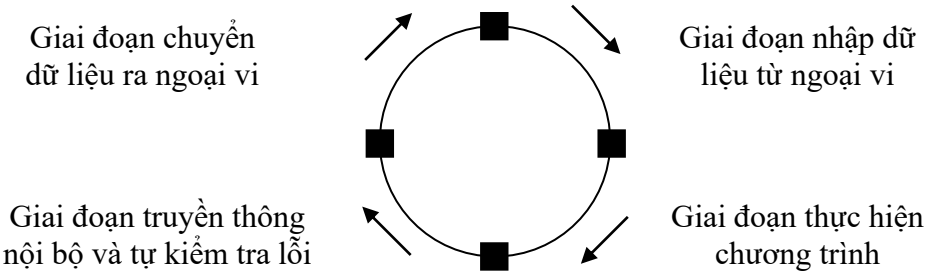
Hình 6.1

5.2. NGÔN NGỮ LẬP TRÌNH CỦA S7-200

5.2.1. Phương pháp lập trình

S7-200 biểu diễn một mạch logic cứng bằng một một chương trình. Chương trình bao gồm một tập dãy các lệnh lập trình. S7-200 thực hiện chương trình bắt đầu từ lệnh lập trình đầu tiên và kết thúc ở lệnh lập trình cuối cùng. Một vòng như vậy được gọi là *vòng quét (scan).*

Một vòng quét (scan cycle) được bắt đầu bằng việc đọc các trạng thái của đầu vào và sau đó thực hiện chương trình. Vòng quét kết thúc bằng việc thay đổi trạng thái đầu ra. Trước khi thực hiện vòng quét tiếp theo PLC thực hiện các nhiệm vụ bên trong và nhiệm vụ truyền thông. Chu trình thực hiện chương trình là một chu trình lặp (*hình 6-2*)



Hình 6.2: Thực hiện chương trình theo vòng quét trong S7-200

Cách lập trình cho S7-200 nói riêng và cho các PLC của hãng Siemens nói chung dựa trên ba phương pháp cơ bản: Phương pháp hình thang (*Ladder Logic* viết tắt là LAD), phương pháp liệt kê lệnh (*Statement List* viết tắt là STL) và phương pháp sơ đồ các khối chức năng (FBD). Nếu chương trình lập theo FBD thì thiết bị lập trình có thể chuyển sang dạng LAD hoặc STL tương ứng. Ngược lại không phải mọi chương trình viết ở dạng STL hoặc LAD cũng có thể chuyển sang dạng FBD. Nếu chương trình lập theo LAD thì thiết bị lập trình có thể chuyển sang dạng STL tương ứng, nhưng không phải mọi chương trình viết ở dạng STL cũng có thể chuyển sang dạng LAD. Trong phần này chúng ta nghiên cứu hai phương pháp lập trình là LAD và STL, chúng đều có sẵn trong ngôn ngữ lập trình STEP7-Micro/DOS và STEP7-Micro/WIN, còn lập trình kiểu FBD chỉ có trong STEP7-Micro/WIN. Để dễ dàng làm quen với với các thành phần cơ bản của LAD và STL ta cần nắm các định nghĩa cơ bản sau đây:

Định nghĩa về LAD: LAD là một ngôn ngữ lập trình bằng đồ hoạ. Những thành phần cơ bản dùng trong LAD tương ứng với các thành phần của bảng điều khiển bằng rơle. Trong chương trình LAD các phần tử cơ bản dùng để biểu diễn các lệnh logic như sau :

- *Tiếp điểm*: là biểu tượng (symbol) mô tả các tiếp điểm của rơle. Các tiếp điểm đó có thể là thường mở (–| |–) hoặc thường đóng (–| / |–) .

- *Cuộn dây (coil)*: là biểu tượng (–( )–) mô tả cuộn dây rơle

- *Hộp (box)*: là biểu tượng mô tả các hàm khác nhau, nó làm việc khi có dòng điện chạy đến hộp. Những dạng hàm thường được biểu diễn bằng hộp là các bộ thời gian (*Timer*), các bộ đếm (*Counter*) và các hàm toán học. Cuộn dây và hộp phải được mắc đúng chiều dòng điện .

- **Mạng LAD**: là đường nối các phần tử thành một mạch hoàn thiện, đi từ đường nguồn bên trái sang đường nguồn bên phải. Đường nguồn bên trái là dây nóng, đường nguồn bên phải là dây

trung hoà (neutral) hay là đường trở về nguồn cung cấp (Đường nguồn bên phải thường không được thể hiện khi dùng chương trình tiện dụng STEP7-Micro/DOS hoặc STEP7-Micro/WIN). Dòng điện chạy từ trái qua các tiếp điểm đóng đến các cuộn dây hoặc các hộp trở về bên phải nguồn.

Định nghĩa về STL: Phương pháp liệt kê lệnh (STL) là phương pháp thể hiện chương trình dưới dạng tập hợp các câu lệnh. Mỗi câu lệnh trong chương trình, kể cả những lệnh hình thức biểu diễn một chức năng của PLC.

Định nghĩa về ngăn xếp logic (logic stack):

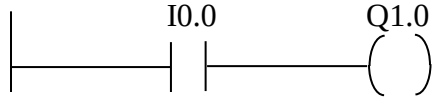
S0	Stack 0-bit đầu tiên hay bit trên cùng của ngăn xếp
S1	Stack 1-bit thứ hai của ngăn xếp
S2	Stack 2-bit thứ ba của ngăn xếp
S3	Stack 3-bit thứ tư của ngăn xếp
S4	Stack 4-bit thứ năm của ngăn xếp
S5	Stack 5-bit thứ sáu của ngăn xếp
S6	Stack 6-bit thứ bảy của ngăn xếp
S7	Stack 7-bit thứ tám của ngăn xếp
S8	Stack 8-bit thứ chín của ngăn xếp

Hình 6.3

Để tạo ra một chương trình dạng STL, người lập trình cần phải hiểu rõ phương thức sử dụng 9 bit ngăn xếp logic của S7-200. ***Ngăn xếp logic là một khối gồm 9 bit chồng lên nhau*** . Tất cả các thuật toán liên quan đến ngăn xếp đều chỉ làm việc với bit đầu tiên hoặc với bit đầu và bit thứ hai của ngăn xếp. Giá trị logic mới đều có thể được gửi (hoặc được nối thêm) vào ngăn xếp. Khi phối hợp hai bit đầu tiên của ngăn xếp, thì ngăn xếp sẽ được kéo lên một bit. Ngăn xếp và tên của từng bit trong ngăn xếp được biểu diễn trên hình 6.3 .

Ví dụ về ladder logic và statement list :

Hình 6.4 mô tả việc thực hiện lệnh LD (viết tắt từ Load trong tiếng Anh) đưa giá trị logic của tiếp điểm I0.0 vào trong ngăn xếp theo cách biểu diễn của LAD và STL:

LAD	STL
	LD I0.0 = Q1.0

Hình 6.4

5.2.2- Bảng tóm tắt một số lệnh cơ bản của S7-200

Hệ lệnh của S7-200 được chia làm 3 nhóm :

- Các lệnh mà khi thực hiện thì làm việc độc lập không phụ thuộc vào giá trị logic của ngăn xếp.

- Các lệnh chỉ thực hiện được khi bit đầu tiên của ngăn xếp có giá trị logic bằng 1.
- Các nhãn lệnh đánh dấu vị trí trong tập lệnh .

Trong các bảng lệnh còn mô tả sự thay đổi tương ứng của nội dung ngăn xếp khi lệnh được thực hiện .

Cả hai phương pháp LAD và STL đều sử dụng ký hiệu I để chỉ việc thực hiện *tức thời* (immediately), tức là giá trị được chỉ dẫn trong lệnh vừa được chuyển vào thanh ghi ảo vừa đồng thời được chuyển đến tiếp điểm chỉ dẫn trong lệnh ngay khi lệnh được thực hiện chứ không phải chờ tới giai đoạn trao đổi với ngoại vi của vòng quét (xem hình 6-2). Điều đó khác với lệnh không tức thời là giá trị được chỉ định trong lệnh chỉ được chuyển vào thanh ghi ảo khi thực hiện lệnh.

Bảng 5.1: Một số lệnh của S7-200 thuộc nhóm lệnh thực hiện vô điều kiện

Tên lệnh	***Mô tả***
= n	Giá trị của bit đầu tiên ngăn xếp được sao chép sang điểm n chỉ dẫn trong lệnh.
=I n	Giá trị của bit đầu tiên ngăn xếp được sao chép trực tiếp sang điểm n chỉ dẫn trong lệnh ngay khi lệnh được thực hiện.
A n	Thực hiện toán tử **và** (AND) giữa giá trị logic của bit đầu tiên ngăn xếp với giá trị logic của điểm n chỉ dẫn trong lệnh. Kết quả được ghi lại vào bit đầu tiên của ngăn xếp.
ALD	Thực hiện toán tử **và** (AND) giữa giá trị logic của bit đầu tiên ngăn xếp với giá trị logic của bit thứ hai ngăn xếp. Kết quả được ghi lại vào bit đầu tiên ngăn xếp. Các giá trị còn lại trong ngăn xếp được kéo lên một bit.
AN n	Thực hiện toán tử **và** (AND) giữa giá trị logic của bit đầu tiên ngăn xếp với giá trị logic nghịch đảo của điểm n chỉ dẫn trong lệnh. Kết quả được ghi lại vào bit đầu tiên của ngăn xếp.
CTU Cxx,PV	Khởi động bộ đếm tiến theo sườn lên của tín hiệu vào. Bộ đếm được đặt lại trạng thái ban đầu (*reset*) nếu đầu vào R của bộ đếm được kích (có mức logic 1).
CTUD Cxx,PV	Khởi động bộ đếm tiến theo sườn lên của tín hiệu đầu vào thứ nhất và đếm lùi theo sườn lên của tín hiệu đầu vào thứ hai. Bộ đếm được reset lại nếu đầu vào R của bộ đếm được kích (có mức logic 1).
ED	Đặt giá trị logic 1 vào bit đầu tiên của ngăn xếp khi xuất hiện sườn xuống của tín hiệu.
EU	Đặt giá trị logic 1 vào bit đầu tiên của ngăn xếp khi xuất hiện sườn lên của tín hiệu.
LD n	Nạp giá trị logic của điểm n chỉ dẫn trong lệnh vào bit đầu tiên của ngăn xếp. Các giá trong ngăn xếp được đẩy xuống một bit.
LDN n	Nạp giá trị logic nghịch đảo của điểm n chỉ dẫn trong lệnh vào bit đầu tiên của ngăn xếp. Các giá trong ngăn xếp được đẩy xuống một bit.

LDW<= n1, n2	Bit đầu tiên trong ngăn xếp nhận giá trị logic 1 nếu nội dung hai từ n1 và n2 thoả mãn n1 ≤n2.
LDW = n1, n2	Bit đầu tiên trong ngăn xếp nhận giá trị logic 1 nếu nội dung hai từ n1 và n2 thoả mãn n1 = n2.
LDW>= n1, n2	Bit đầu tiên trong ngăn xếp nhận giá trị logic 1 nếu nội dung hai từ n1 và n2 thoả mãn n1 ≥ n2.
LPP	Kéo nội dung ngăn xếp lên một bit. Giá trị mới của bit trên là giá trị cũ của bit dưới, độ sâu ngăn xếp giảm đi một bit (giá trị của bit đầu tiên bị đẩy ra khỏi ngăn xếp - xoá).
LPS	Sao chép giá trị của bit đầu tiên vào bit thứ hai của ngăn xếp. Các giá trị còn lại từ bit thứ hai trở đi được đẩy xuống 1 bit.
LRD	Sao chép giá trị của bit thứ hai vào bit đầu tiên của ngăn xếp. Các giá trị còn lại từ bit thứ hai trở đi được giữ nguyên vị trí.
MEND	Kết thúc phần chương trình chính trong một vòng quét.
NOT	Đảo giá trị logic của bit đầu tiên ngăn xếp.
O n	Thực hiện toán tử **hoặc** (OR) giữa giá trị logic của bit đầu tiên ngăn xếp với giá trị logic của điểm n chỉ dẫn trong lệnh. Kết quả được ghi lại vào bit đầu tiên của ngăn xếp.
OI n	Thực hiện tức thời toán tử **hoặc** (OR) giữa giá trị logic của bit đầu tiên ngăn xếp với giá trị logic của điểm n chỉ dẫn trong lệnh. Kết quả được ghi lại vào bit đầu tiên của ngăn xếp.
OLD	Thực hiện toán tử **hoặc** (OR) giữa giá trị logic của bit đầu tiên ngăn xếp với giá trị logic của bit thứ hai ngăn xếp. Kết quả được ghi lại vào bit đầu tiên của ngăn xếp. Các giá trị còn lại trong ngăn xếp được kéo lên một bit.
ON n	Thực hiện toán tử **hoặc** (OR) giữa giá trị logic của bit đầu tiên ngăn xếp với giá trị logic nghịch đảo của điểm n chỉ dẫn trong lệnh. Kết quả được ghi lại vào bit đầu tiên của ngăn xếp .
RET	Lệnh thoát khỏi chương trình con, trả điều khiển về chương trình chính.
RETI	Lệnh thoát chương trình xử lý ngắt, trả điều khiển về chương trình chính.

Bảng 5.2: Một số lệnh trong nhóm lệnh có điều kiện (chỉ thực hiện khi bit đầu tiên ngăn xếp có giá trị logic 1):

Tên lệnh	***Mô tả***
+D IN1, IN2	Thực hiện phép cộng hai số nguyên kiểu từ kép IN1 và IN2 . Kết quả được ghi lại vào IN2.
+I IN1, IN2	Thực hiện phép cộng hai số nguyên kiểu từ IN1 và IN2. Kết quả được ghi lại vào IN2.
-D IN1, IN2	Thực hiện phép trừ hai số nguyên kiểu từ kép IN1 và IN2. Kết quả

	được ghi lại vào IN2.
-I IN1, IN2	Thực hiện phép trừ hai số nguyên kiểu từ IN1 và IN2. Kết quả được ghi lại vào IN2.
+R IN1, IN2	Thực hiện phép cộng hai số thực (32 bit) IN1 và IN2. Kết quả được ghi lại vào IN2.
-R IN1, IN2	Thực hiện phép trừ hai số thực (32 bit) IN1 và IN2. Kết quả được ghi lại vào IN2.
*R IN1 , IN2	Thực hiện phép nhân hai số thực (32 bit) IN1 và IN2. Kết quả được ghi lại vào IN2.
/ R IN1, IN2	Thực hiện phép cộng hai số thực (32 bit) IN1 và IN2. Kết quả được ghi lại vào IN2.
ANDD IN1, IN2	Thực hiện toán tử logic AND giữa các giá trị kiểu từ kép IN1 và IN2. Kết quả được ghi lại vào IN2.
ANDW IN1, IN2	Thực hiện toán tử logic AND giữa các giá trị kiểu từ IN1 và IN2. Kết quả được ghi lại vào IN2.
CALL n	Gọi chương trình con được đánh nhãn n.
CRET	Kết thúc một chương trình con và trả điều khiển về chương trình đã gọi nó.
CRETI	Kết thúc một chương trình xử lý ngắt và trả điều khiển về chương trình chính.
MOVB IN, OUT	Sao giá trị của byte IN sang byte OUT.
MOVD IN, OUT	Sao giá trị của từ kép IN sang từ kép OUT.
MOVR IN, OUT	Sao số thực IN sang OUT.
MOVW IN, OUT	Sao giá trị của từ IN sang từ OUT.
ORD IN1, IN2	Thực hiện toán tử OR cho hai từ kép IN1 và IN2, kết quả được ghi lại vào IN2.
ORW IN1, IN2	Thực hiện toán tử OR cho hai từ IN1 và IN2, kết quả được ghi lại vào IN2.
PLS x	Đưa bộ phát xung nhanh đã được định nghĩa trong bộ nhớ đặc biệt vào trạng thái tích cực. Xung ra được đưa ra cổng Q0.x.
RLD IN, n	Quay tròn từ kép IN sang trái n bit.
RLW IN, n	Quay tròn từ IN sang trái n bit.
RRD IN, n	Quay tròn từ kép IN sang phải n bit.
RRW IN, n	Quay tròn từ IN sang phải n bit.
SLD IN, n	Dịch từ kép IN sang trái n bit
SLW IN, n	Dịch từ IN sang trái n bit
SQRT IN, OUT	Lấy căn bậc hai của một số thực 32 bit IN và ghi kết quả vào OUT (32 bit)
SRD IN , n	Dịch từ kép IN sang phải n bit.
SRW IN , n	Dịch từ IN sang phải n bit.

STOP	Dừng "mềm" chương trình.
SWAP IN	Đổi chỗ hai bit đầu tiên và cuối cùng của byte IN cho nhau.

Bảng 6.3 : Các lệnh đặt nhãn (label)

Tên lệnh	***Mô tả***
INT n	Khai báo nhãn n cho chương trình xử lý ngắt
LBL xx	Đặt nhãn "xx" trong chương trình, định hướng cho lệnh nhảy JMP
NEXT	Lệnh kết thúc vòng lặp FOR ... NEXT
NOP	Lệnh rỗng (no operation)
SBR n	Khai báo nhãn n cho chương trình con

5.2.3. Cú pháp hệ lệnh của S7-200

Mặc dù S7-200 có một khối lượng lớn các lệnh để thực hiện các thuật toán của đại số *Boolean* song chỉ có một vài các kiểu lệnh khác nhau. Sau đây chúng ta sẽ mô tả chi tiết về cách sử dụng, chức năng và tác động của chúng vào nội dung ngăn xếp. Trong đó giá trị của ngăn xếp trước khi thực hiện lệnh được ký hiệu bằng c0 đến c8 (viết tắt chữ "cũ"), ký hiệu "m" được dùng để cho nội dung ngăn xếp sau khi thực hiện lệnh . Trong các ví dụ sau, dấu $\wedge$ dùng cho toán tử AND, dấu $\vee$ dùng cho toán tử OR và ký hiệu ~ dùng cho toán tử NOT.

6.2.3.1. Toán hạng và giới hạn cho phép

Bảng 6.4:

Phương pháp truy nhập	Giới hạn cho phép của toán hạng			
	CPU212		CPU214	
Truy nhập bit (địa chỉ byte.chỉ số bit)	V I Q M SM T C	(0.0 đến 1023.7) (0.0 đến 7.7) (0.0 đến 7.7) (0.0 đến 15.7) (0.0 đến 45.7) (0 đến 63) (0 đến 63)	V I Q M SM T C	(0.0 đến 4095.7) (0.0 đến 7.7) (0.0 đến 7.7) (0.0 đến 31.7) (0.0 đến 85.7) (0 đến 127) (0 đến 127)
Truy nhập byte	VB IB MB SMB AC Hằng số	(0 đến 1022) (0 đến 7) (0 đến 15) (0 đến 45) (0 đến 3)	VB IB MB SMB AC Hằng số	(0 đến 4095) (0 đến 7) (0 đến 31) (0 đến 85) (0 đến 3)
Truy nhập từ đơn (địa chỉ byte	VW T C	(0 đến 1022) (0 đến 63) (0 đến 63)	VW T C	(0 đến 4094) (0 đến 127) (0 đến 127)

cao)	IW (0 đến 6) QW (0 đến 6) MW (0 đến 14) SMW (0 đến 44) AC (0 đến 3) AIW (0 đến 30) AQW (0 đến 30) Hằng số	IW (0 đến 6) QW (0 đến 6) MW (0 đến 30) SMW (0 đến 84) AC (0 đến 3) AIW (0 đến 30) AQW (0 đến 30) Hằng số
Truy nhập từ kép (địa chỉ byte cao)	VD (0 đến 1022) ID (0 đến 4) QD (0 đến 4) MD (0 đến 12) SMD (0 đến 42) AC (0 đến 3) HC (0) Hằng số	VD (0 đến 4092) ID (0 đến 4) QD (0 đến 4) MD (0 đến 28) SMD (0 đến 82) AC (0 đến 3) HC (0 đến 2) Hằng số

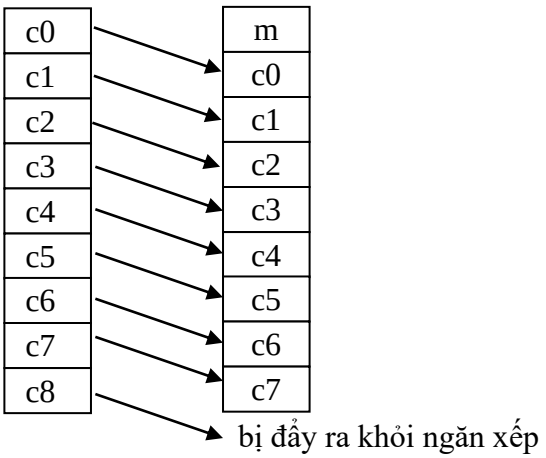
5.2.3.2. Lệnh vào / ra

Load (LD) và **Load Not (LDN)**

Lệnh LD nạp giá trị logic của một tiếp điểm vào bit đầu tiên của ngăn xếp (hình 6.5), các giá trị cũ còn lại trong ngăn xếp bị đẩy lùi xuống một bit.

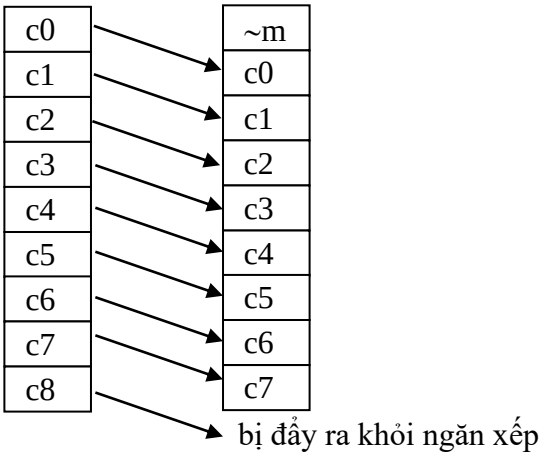
Lệnh LDN nạp giá trị logic nghịch đảo của một tiếp điểm vào trong bit đầu tiên của ngăn xếp (hình 6.6), các giá trị cũ còn lại trong ngăn xếp bị đẩy lùi xuống một bit.

Lệnh **LD** :



Hình 6.5: Trạng thái ngăn xếp trước và sau khi thực hiện lệnh LD

Lệnh **LDN** :



Hình 6-6 : Trạng thái ngăn xếp trước và sau khi thực hiện lệnh LDN

Các dạng khác nhau của lệnh LD, LDN cho LAD

LAD	Mô tả	Toán hạng
n ⊣ ⊢	Tiếp điểm thường mở sẽ được đóng nếu n = 1	n: I, Q, M, SM, T, C, V (bit)
n ⊣/⊢	Tiếp điểm thường đóng sẽ được mở nếu n = 1	

Các dạng khác nhau của lệnh LD , LDN cho STL

Lệnh	Mô tả	Toán hạng
LD n	Tiếp điểm thường mở sẽ được đóng nếu n = 1	n: I, Q, M, SM, T, C, V (bit)
LDN n	Tiếp điểm thường đóng sẽ được mở nếu n = 1	

OUTPUT (=) Lệnh sao chép nội dung của bit đầu tiên trong ngăn xếp vào bit (điểm) được chỉ dẫn trong lệnh. Nội dung của ngăn xếp không bị thay đổi.

Mô tả lệnh "=" bằng LAD như sau:

LAD	Mô tả	Toán hạng
n ( )	Cuộn dây đầu ra ở trạng thái kích thích, có dòng điều khiển đi qua nếu n = 1	n: I, Q, M, SM, T, C, V (bit)

Mô tả lệnh "=" bằng STL như sau :

Lệnh	Mô tả	Toán hạng

= n	Cuộn dây đầu ra ở trạng thái kích thích, có dòng điều khiển đi qua nếu n = 1	n: I, Q, M, SM, T, C, V (bit)

5.2.3.3. Lệnh ghi / xoá giá trị cho tiếp điểm

SET (S)

RESET (R) Đây là các lệnh dùng để đóng các điểm (tiếp điểm) đã được thiết kế. Trong LAD, logic điều khiển dòng điện đóng hoặc ngắt các cuộn dây đầu ra. Khi dòng điều khiển đến các cuộn dây thì các cuộn dây đóng hoặc mở các tiếp điểm (hoặc một dãy các tiếp điểm).

Trong STL, thì đây là lệnh truyền trạng thái bit đầu tiên của ngăn xếp đến các điểm thiết kế. Nếu bit này bằng 1, các lệnh S và R sẽ đóng ngắt một tiếp điểm hoặc một dãy các tiếp điểm (giới hạn từ 1 đến 255). Nội dung của ngăn xếp không bị thay đổi.

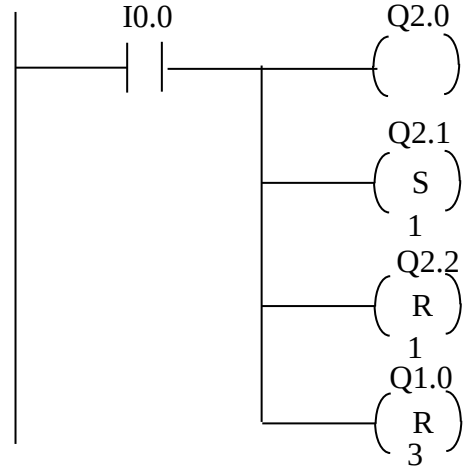
Mô tả lệnh S và R bằng LAD như sau:

LAD	Mô tả	Toán hạng
S_BIT S n	Đóng một mảng gồm n các tiếp điểm kể từ S_BIT	S_BIT : I, Q, M, SM, T, C, V (bit) n: IB, QB, MB, SMB, VB (byte)
S_BIT R n	Ngắt một mảng gồm n các tiếp điểm kể từ S_BIT. Nếu S_BIT lại chỉ vào Timer hoặc Counter thì lệnh sẽ xoá bit đầu ra của Timer/Counter đó.	

Mô tả lệnh S và R bằng STL như sau:

Lệnh	Mô tả	Toán hạng
S S_BIT , n	Đặt giá trị logic 1 cho một mảng gồm n bit kể từ địa chỉ S_BIT.	S_BIT : I, Q, M, SM, T, C, V (bit)
R S_BIT , n	Xoá một mảng gồm n bit kể từ địa chỉ S_BIT. Nếu S_BIT lại chỉ vào Timer hoặc Counter thì lệnh sẽ xoá bit đầu ra của Timer/Counter đó.	n: IB, QB, MB, SMB, VB (byte)

Ví dụ : Mô tả việc thực hiện lệnh S và R trong LAD và STL:

LAD	***STL***
	LD **I0.0** = Q2.0 S Q2.1 , 1 R Q2.2 , 1 R Q1.0 , 3

5.2.3.4. Các lệnh logic đại số Boolean

Các lệnh tiếp điểm đại số Boolean cho phép tạo lập được các mạch logic (không có nhớ). Trong LAD các lệnh này được biểu diễn thông qua cấu trúc mạch (mắc nối tiếp hay song song các tiếp điểm thường đóng và thường mở) Trong STL các lệnh được viết tắt như sau A (AND), O (OR), AN (AND NOT), ON (OR NOT). Giá trị ngăn xếp thay đổi phụ thuộc vào từng lệnh.

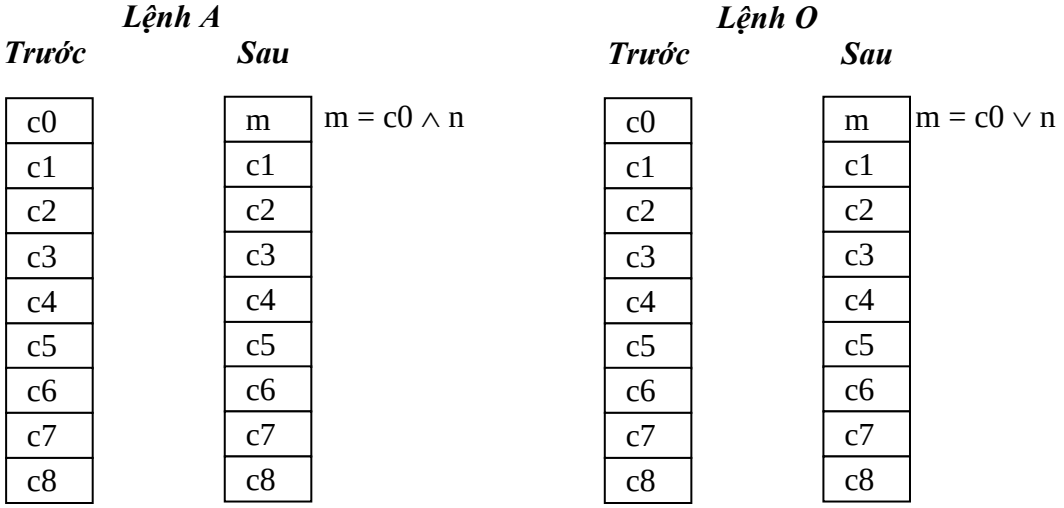
Lệnh	Mô tả	Toán hạng
A n O n	Lệnh thực hiện toán tử $\wedge$ (AND) và $\vee$ (OR) giữa giá trị logic của tiếp điểm n và giá trị bit đầu tiên của ngăn xếp . Kết quả lại được ghi vào bit đầu tiên của ngăn xếp	n :I, Q, M, (bit) SM, T, C, V
AN n ON n	Lệnh thực hiện toán tử $\wedge$ (AND) và $\vee$ (OR) giữa giá trị logic nghịch đảo của tiếp điểm n và giá trị bit đầu tiên của ngăn xếp . Kết quả lại được ghi vào bit đầu tiên của ngăn xếp	

Ngoài những lệnh làm việc trực tiếp với tiếp điểm, S7-200 còn có 5 lệnh đặc biệt biểu diễn các phép tính đại số Boolean cho các bit trong ngăn xếp , được gọi là các lệnh ***stack logic***. Đó là các lệnh ALD (And load), OLD (Or load), LPS (Logic push), LRD (Logic read), LPP (Logic pop). Lệnh ***stack logic*** được dùng để tổ hợp, sao chụp hoặc xoá các mệnh đề logic. Bảng sau tóm tắt các lệnh ***stack logic*** trong STL:

Lệnh	Mô tả	Toán hạng
ALD	Lệnh tổ hợp giá tri logic của bit đầu tiên và bit thứ hai của ngăn xếp bằng phép tính $\wedge$ (AND). Kết quả được ghi lại vào bít đầu tiên của ngăn xếp, giá trị còn lại của ngăn xếp được kéo lên môt bit.	Không có
OLD	Lệnh tổ hợp giá tri logic của bit đầu tiên và bit thứ hai của ngăn xếp bằng phép tính $\vee$ (OR). Kết quả được ghi lại vào bít đầu tiên	Không có

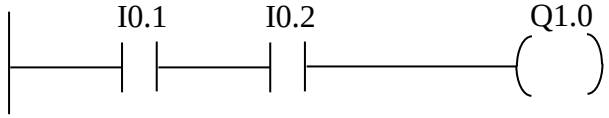
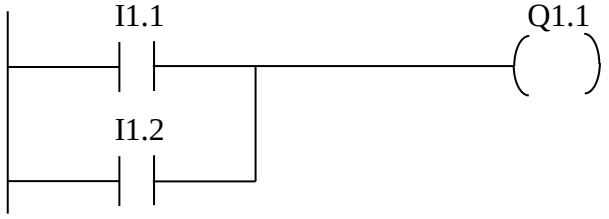
	của ngăn xếp, giá trị còn lại của ngăn xếp được kéo lên môt bit.	
LPS	Lệnh Logic Push thực hiện sao chụp giá trị logic của bit đầu tiên ngăn xếp vào bit thứ hai. Giá trị còn lại trong ngăn xếp bị đẩy xuống 1 bit. Bit cuối cùng bị đẩy ra khỏi ngăn xếp.	Không có
LRD	Lệnh thực hiện sao chụp giá trị logic của bit thứ hai ngăn xếp vào bit đầu tiên. Giá trị còn lại trong ngăn xếp được giữ nguyên vị trí.	Không có
LPP	Lệnh kéo ngăn xếp lên 1 bit. Giá trị của bit sau được chuyển cho bit trước. Giá trị cũ của bit đầu tiên bị đẩy ra ngoài .	Không có

A và **O :** Tác động của lệnh **A** và **O** vào ngăn xếp :

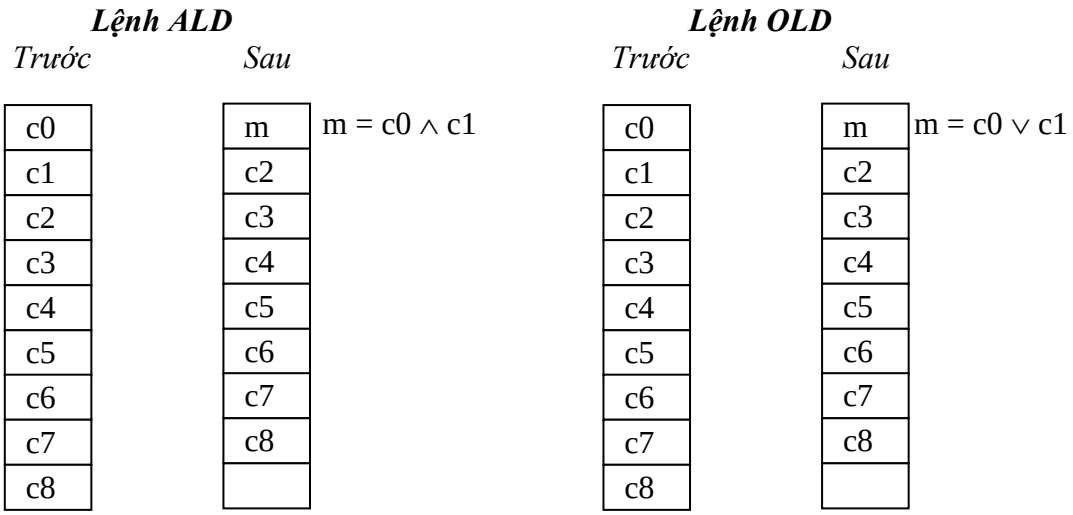


Hình 6-6 : Trạng thái ngăn xếp trước và sau khi thực hiện lệnh A và O

Mô tả lệnh A và O ở dạng LAD và STL:

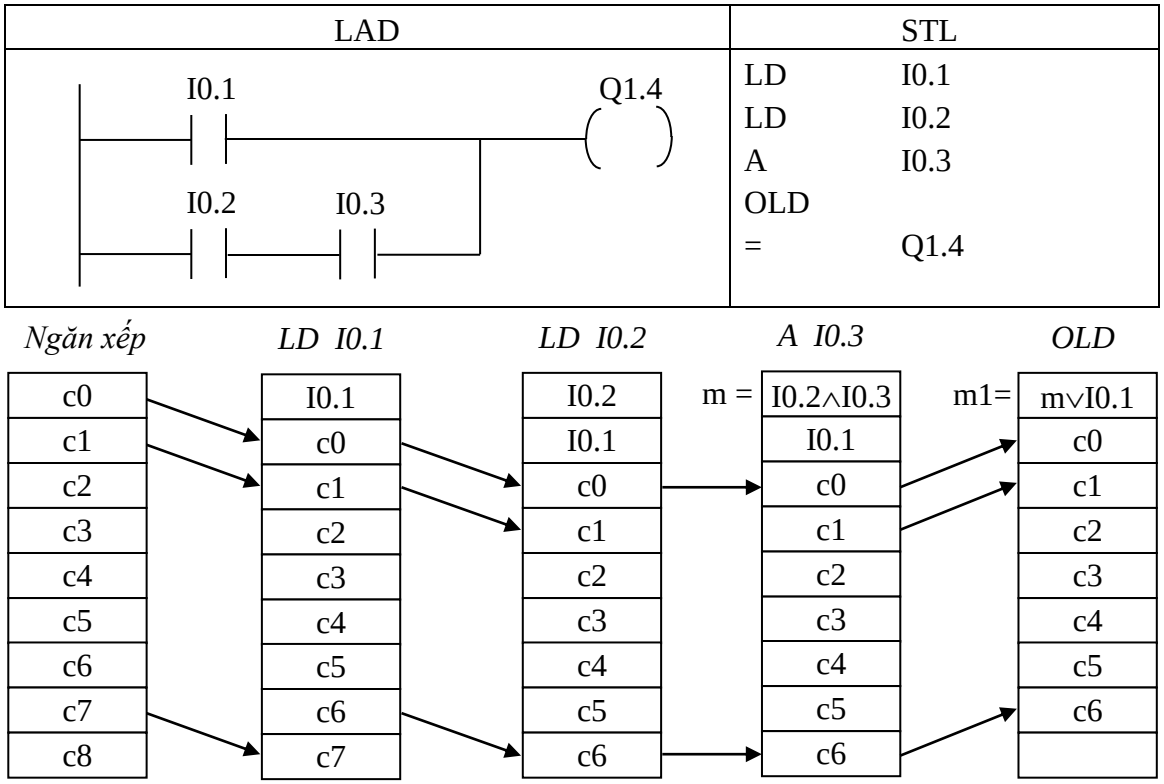
LAD	STL
	LD I0.1 A I0.2 = Q1.0
	LD I1.1 O I1.2 = Q1.1

ALD và **OLD :** Tác động của lệnh **ALD** và **OLD** vào ngăn xếp:



Hình 6.7: Trạng thái ngăn xếp trước và sau khi thực hiện lệnh ALD và OLD

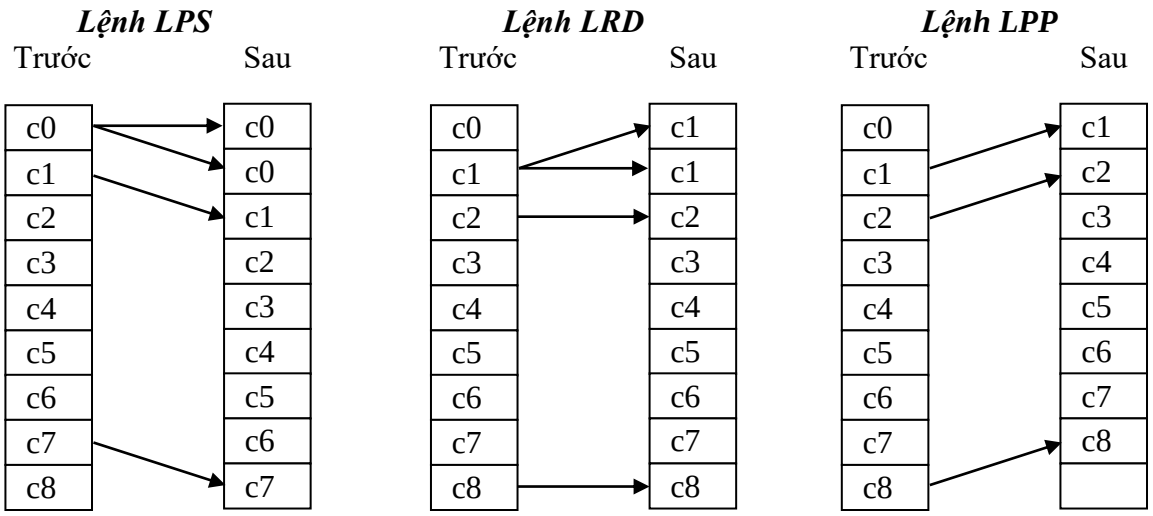
Ví dụ : Mô tả lệnh OLD ở dạng LAD và STL:



Hình 6-8 : Trạng thái ngăn xếp trước và sau khi thực hiện các lệnh ở bảng trên

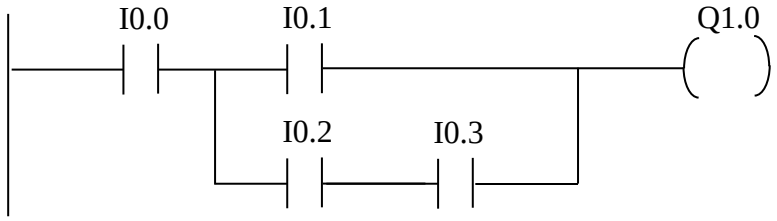
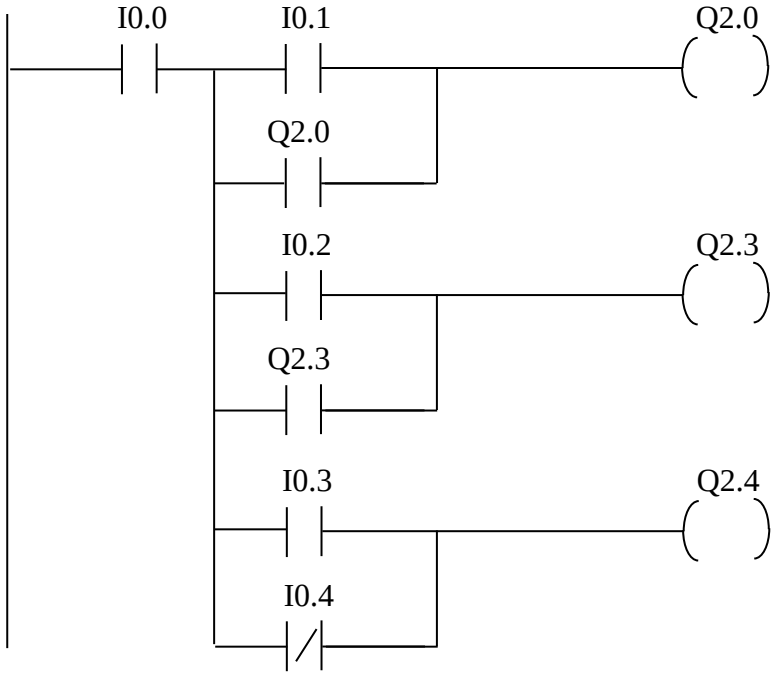
LPS, LRD, LPP

Các lệnh này thực hiện sao chép, thay đổi nội dung bit đầu tiên hoặc bit thứ hai của ngăn xếp và được mô tả như sau:



Hình 6.9: Trạng thái ngăn xếp trước và sau khi thực hiện lệnh các lệnh LPS, LRD, LPP

Ví dụ: Các lệnh Logic stack ở dạng LAD và STL

LAD	STL
	LD I0.0 LD I0.1 LD I0.2 A I0.3 OLD ALD = Q1.0
	LD I0.0 LPS LD I0.1 O Q2.0 ALD = Q2.0 LRD LD I0.2 O Q2.3 ALD = Q2.3 LPP LD I0.3 ON I0.4 ALD = Q2.4

6.2.3.5. Các lệnh so sánh và di chuyển nội dung ô nhớ:

STL	Mô tả	Toán hạng
LDW>= n1 n2 AW>= n1 n2 OW>= n1 n2	Lệnh thực hiện phép tính logic Load, And hoặc Or giữa giá trị 1 với nội dung của đỉnh ngăn xếp khi nội dung 2 từ n1, n2 thoả mãn n1 ≥ n2.	n1, n2 (từ) : VW, T, C, IW, QW, MW, SMW ...
MOVW IN OUT	Lệnh sao chép nội dung từ đơn IN sang từ đơn OUT.	IN, OUT (từ đơn): VW, T, C, IW, QW

6.2.3.6. Các lệnh làm việc với Timer:

Timer là bộ tạo thời gian trễ giữa tín hiệu vào và tín hiệu ra nên trong điều khiển thường được gọi là khâu trễ. Nếu ký hiệu tín hiệu (logic) vào là *x(t)* và thời gian trễ là τ thì tín hiệu đầu ra của Timer là *x(t-τ)*. Trong S7-200 có hai loại Timer khác nhau:

- Timer tạo thời gian trễ không có nhớ (*On-Delay Timer*), ký hiệu là TON.
- Timer tạo thời gian trễ có nhớ (*Retentive On-Delay Timer*), ký hiệu là TONR.

Hai loại Timer này phân biệt nhau bởi phản ứng của chúng đối với tín hiệu vào. Cả hai loại đều bắt đầu tạo thời gian trễ từ thời điểm có sườn lên của tín hiệu vào. Nhưng TON sẽ tự Reset khi đầu vào có mức logic 0, còn TONR thì không tự Reset khi mất tín hiệu vào. TON được dùng để tạo thời gian trễ trong một khoảng thời gian, còn với TONR thời gian trễ được tạo ra trong nhiều khoảng khác nhau. Trong phần này ta chỉ nghiên cứu loại Timer TON.

Lệnh	Độ phân giải	Giá trị cực đại	CPU 212	CPU 214
TON	1 ms	32,767 s	T32	T32 , T96
	10 ms	327,67 s	T33 ÷ T36	T33 ÷ T36 , T97 ÷ T100
	100 ms	3276,7 s	T37 ÷ T63	T37 ÷ T63 , T101 ÷ T127
TONR	1 ms	32,767 s	T0	T0 , T64
	10 ms	327,67 s	T1 ÷ T4	T1 ÷ T4 , T65 ÷ T68
	100 ms	3276,7 s	T5 ÷ T31	T5 ÷ T31 , T69 ÷ T95

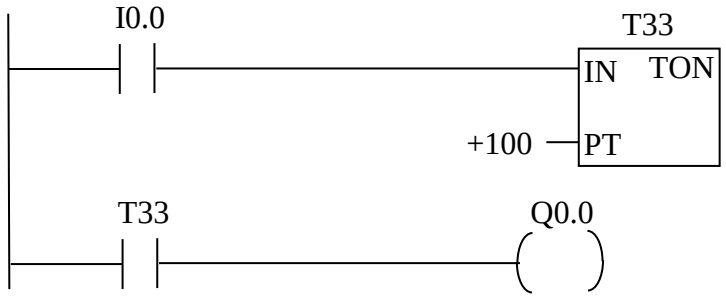
Cú pháp khai báo Timer trong LAD và STL như sau:

LAD	STL	Mô tả	Toán hạng
Txx IN TON PT	TON Txx, +n	Khai báo Timer số hiệu xx kiểu TON để tạo thời gian trễ tính từ khi đầu vào IN được kích (có mức 1). Nếu như giá trị đếm tức thời lớn hơn hoặc bằng giá trị đặt trước PT thì T-bit có giá trị logic bằng 1. Có thể Reset Timer kiểu TON bằng lệnh R hoặc bằng giá	Txx(word): CPU 212 : 32÷63 CPU 214 : 32÷63 và 96÷127 PT(word): VW,T,C,IW,... n = 1÷32767 (số nguyên)

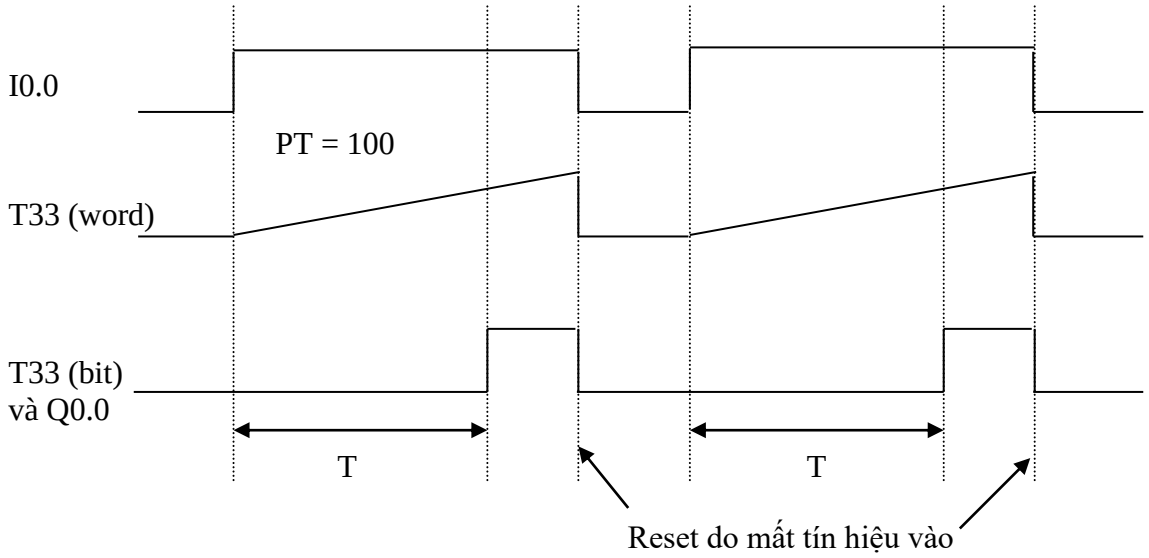
		trị logic 0 ở đầu vào IN .	

Thời gian trễ T = PT × độ phân giải, ví dụ với T33 có độ phân giải là 10 ms và PT=100 thì thời gian trễ T = 10 × 100 = 1000 ms = 1 s.

Sau đây là một ví dụ về sử dụng Timer kiểu TON:

LAD	STL
	Network 1 LD I0.0 TON T33, +100 Network 2 LD T33 = Q0.0

Giản đồ thời gian tương ứng:



Hình 6.10

6.2.3.7. Các lệnh làm việc với Counter:

Counter là bộ đếm thực hiện chức năng đếm sườn lên của xung. S7-200 có hai loại bộ đếm: bộ đếm tiến (CTU) và bộ đếm tiến/lùi (CTUD).

Bộ đếm tiến đếm số sườn lên của xung vào, tức là đếm số lần thay đổi trạng thái logic từ 0 lên 1 của tín hiệu. Số sườn xung đếm được, được ghi vào thanh ghi 2 byte của bộ đếm, gọi là thanh ghi C-word.

Nội dung của C-word, được gọi là *giá trị tức thời* của bộ đếm, luôn được so sánh với giá trị đặt trước của bộ đếm ký hiệu là PV. Khi giá trị đếm tức thời bằng hoặc lớn hơn giá trị đặt trước thì

bộ đếm báo ra ngoài bằng cách đặt giá trị logic 1 vào bit đặc biệt của nó, được gọi là C-bit. Trường hợp giá trị đếm còn nhỏ hơn giá trị đặt trước thì C-bit có giá trị logic 0.

Khác với các Timer, các Counter đều có chân nối với tín hiệu điều khiển xoá để thực hiện đặt lại chế độ khởi phát ban đầu (*rsset)* cho bộ đếm, được ký hiệu bằng chữ cái R trong LAD, hay được qui định là trạng thái bit đầu tiên của ngăn xếp trong STL. Bộ đếm được reset khi tín hiệu xoá này có mức 1 hoặc khi lệnh R (reset) được thực hiện với C-bit. Khi bộ đếm reset thì cả C-word và C-bit đều nhận giá trị 0.

Bộ đếm tiến/lùi CTUD thực hiện đếm tiến khi gặp sườn lên của xung vào cổng đếm tiến, ký hiệu là CU trong LAD hoặc bit thứ 3 ngăn xếp trong STL, và đếm lùi khi gặp sườn lên của xung vào cổng đếm lùi, ký hiệu là CD trong LAD hoặc bit thứ 2 ngăn xếp trong STL.Việc xoá bộ đếm CTUD cũng có hai cách tương tự như bộ đếm CTU.

Cú pháp khai báo Counter trong LAD và STL như sau:

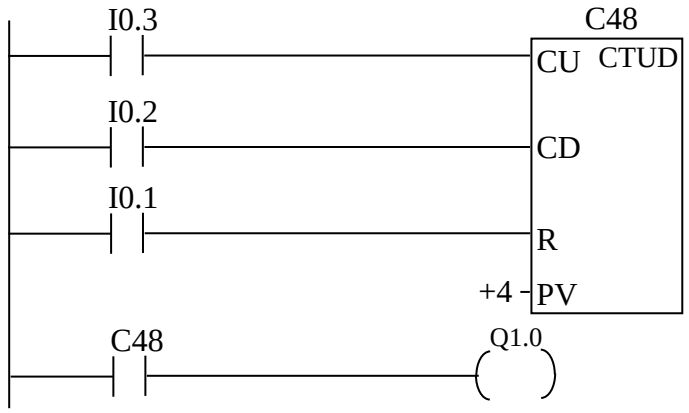
LAD	STL	Mô tả	Toán hạng
Cxx CU CTU R PV	CTU Cxx, +n	Khai báo bộ đếm tiến theo sườn lên của tín hiệu vào cổng CU số hiệu xx kiểu CTU. Khi giá trị đếm tức thời C-word của Cxx lớn hơn hoặc bằng giá trị đặt trước PV, C-bit (Cxx) có giá trị logic bằng 1. Bộ đếm được reset khi đầu vào R có giá trị logic 1. Bộ đếm ngừng đếm khi C-word Cxx đạt giá trị cực đại 32767	Cxx(word): CPU 212 : 0÷47 CPU 214 : 0÷47 và 80÷127 PV(word): VW,T,C,IW,... n = 1÷32767 (số nguyên)
Cxx CU CTUD CD R PV	CTUD Cxx, +n	Khai báo bộ đếm tiến/lùi, đếm tiến theo sườn lên của tín hiệu đến CU và đếm lùi theo sườn lên của tín hiệu đến CD. Khi giá trị tức thời C-word của Cxx lớn hơn hoặc bằng giá trị đặt trước PV, C-bit (Cxx) có giá trị logic bằng 1. Bộ đếm được reset khi đầu vào R có giá trị logic 1. Bộ đếm ngừng đếm tiến khi C-word Cxx đạt giá trị cực đại 32767 và ngừng đếm lùi khi C-word Cxx đạt giá trị cực tiểu là -32767.	Cxx(word): CPU 212: 48÷63 CPU 214: 48÷79 PV(word): VW,T,C,IW,... n = 1÷32767 (số nguyên)

Ký hiệu Cxx của bộ đếm đồng thời cũng là địa chỉ hình thức của C-word và của C-bit. Mặc dù cùng địa chỉ hình thức, song C-word và C-bit vẫn được phân biệt với nhau nhờ kiểu lệnh sử dụng làm việc với kiểu từ hay kiểu tiếp điểm (bit). Ví dụ:

```
LD          C48          // Lệnh làm việc với C-bit của bộ đếm C48.
```

LDW>= C48 // Lệnh làm việc với C-word của bộ đếm C48.

Sau đây là một ví dụ về việc sử dụng Counter loại CTUD trong LAD và trong STL:

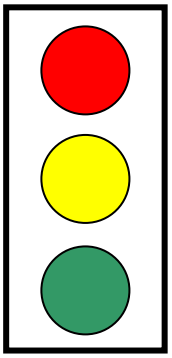
LAD	STL
	Network 1 LD I0.3 LD I0.2 LD I0.1 CTUD C48, +4 Network 2 LD C48 = Q1.0

Một số bit nhớ đặc biệt thường sử dụng khi lập các chương trình đơn giản

Ô nhớ	***Mô tả***
SM0.0	Luôn có giá trị logic bằng 1
SM0.1	Có giá trị logic bằng 1 ở vòng quét đầu tiên
SM0.2	Bit báo dữ liệu bị thất lạc (0-Dữ liệu còn đủ; 1-Dữ liệu bị thất lạc)

5.3. MỘT SỐ VÍ DỤ ỨNG DỤNG S7-200

5.3.1. Chương trình điều khiển đèn đường



Hình 6-12

Qui định các tiếp điểm điều khiển đèn trục chính:

- Xanh: Q0.0 với thời gian 10 s
- Vàng: Q0.1 với thời gian 1 s
- Đỏ: Q0.2 với thời gian 7 s

Qui định các tiếp điểm điều khiển đèn trục phụ:

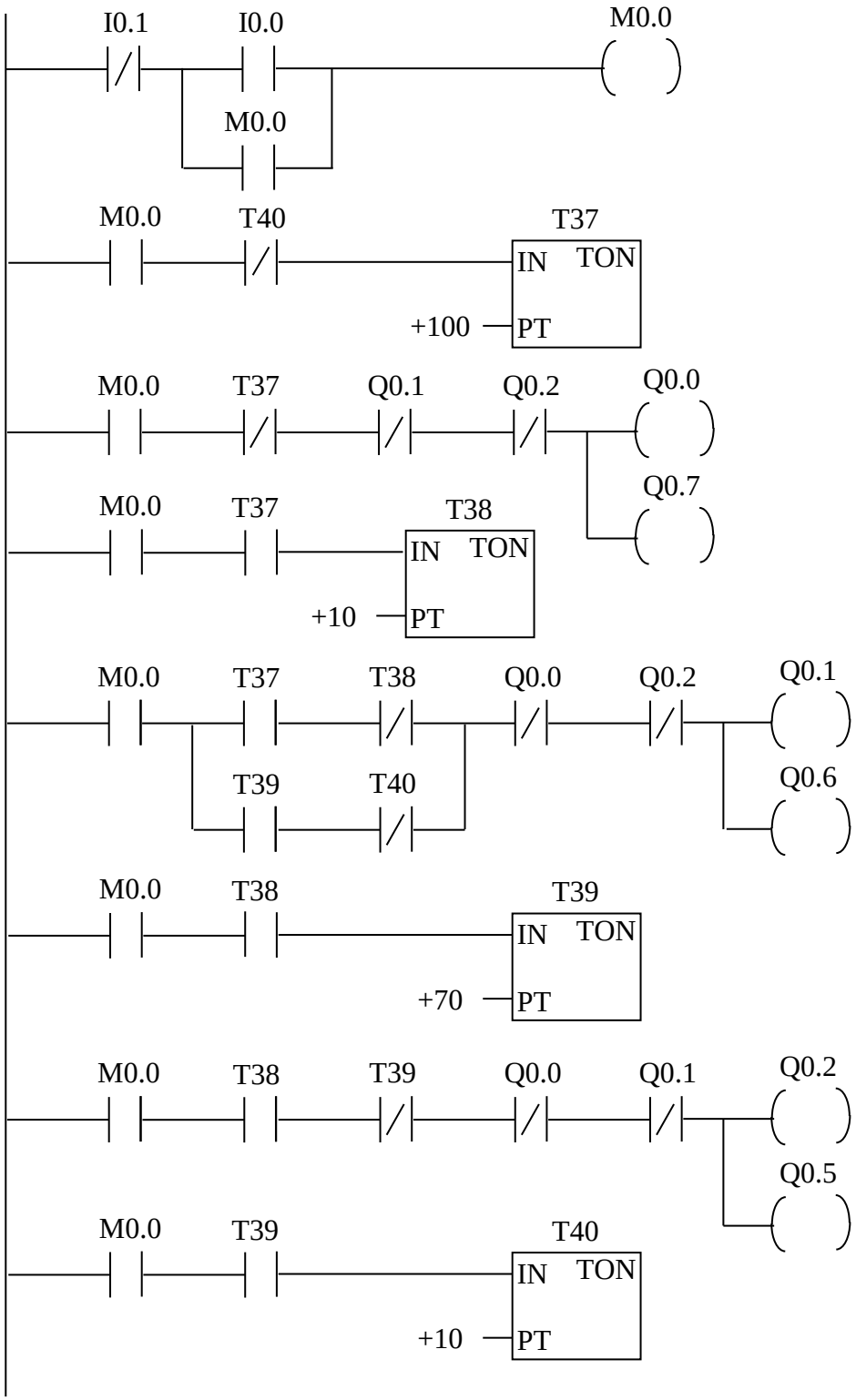
- Xanh: Q0.5 với thời gian 7 s
- Vàng: Q0.6 với thời gian 1 s
- Đỏ: Q0.7 với thời gian 10 s

Yêu cầu của bài toán là khống chế tự động hệ thống đèn ở một ngã tư với trục chính và phụ: Đèn xanh trục chính và đèn đỏ trục phụ cùng sáng trong 10 s, tiếp sau đèn vàng cả hai trục đường cùng sáng trong 1 s, tiếp nữa đèn đỏ trục chính và đèn xanh trục phụ cùng sáng trong 7 s và tiếp sau đèn vàng cả hai trục đường lại cùng sáng trong 1 s - kết thúc 1 chu kỳ và hệ thống tự động hoạt động lặp lại. Để lập chương trình khống chế chúng ta có nhiều cách khác nhau. Sau đây chúng ta sẽ tiến hành lập trình theo một cách:

Sử dụng 3 Timer kiểu TON là T37, T38, T39,T40 đều có độ phân giải 100 ms để khống chế thời gian, chương trình dạng STL như sau :

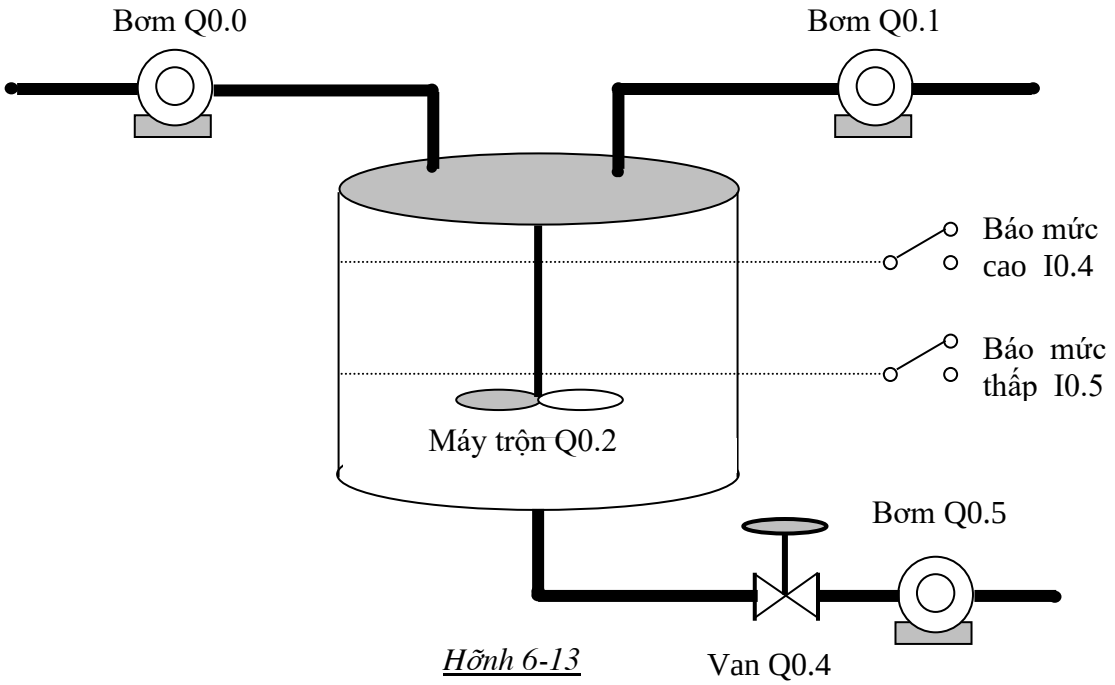
```
 NETWORK1
LDN     I0.1
LD      I0.0
O       M0.0
ALD
=       M0.0

 NETWORK2
LD      M0.0
AN      T40
TON     T37 ,     +100

 NETWORK3
LD      M0.0
AN      T37
AN      Q0.1
AN      Q0.2
=       Q0.0
=       Q0.7

 NETWORK4
LD      M0.0
A       T37
TON     T38 ,     +10

 NETWORK5
LD      M0.0
LD      T37
AN      T38
LD      T39
AN      T40
 OLD
 ALD
AN      Q0.0
AN      Q0.2
=       Q0.1
=       Q0.6

 NETWORK6
LD      M0.0
A       T38
TON     T39 ,     +70

NETWORK7
LD      M0.0
A       T38
AN      T39
AN      Q0.0
AN      Q0.1
=       Q0.2
=       Q0.5

NETWORK8
LD      M0.0
A       T39
TON     T40 ,     +10
```

Chương trình viết trong LAD



5.3.2. Chương trình điều khiển máy trộn sơn

Hình 6.13 là sơ đồ một bình trộn để tạo các màu sơn khác nhau. Trong sơ đồ cho thấy có hai đường ống để đưa hai loại sơ màu khác nhau làm cơ sở cho việc tạo màu sơn mong muốn.



Hỗnh 6-13

Để khống chế các quá trình ta sử dụng hai cảm biến để báo mức trong bình: Cảm biến mức cao (I0.4) và cảm biến mức thấp (I0.5). Thiết bị trộn được điều khiển bởi động cơ trộn (Q0.2). Hai bơm dùng để bơm hai loại sơn màu khác nhau vào bình trộn (Q0.0) và (Q0.1). Điều khiển mở van (Q0.4). Bơm dùng để tháo sản phẩm ra khỏi bình (Q0.5).

Quá trình làm việc của thiết bị có thể mô tả: Khi ấn nút khởi động (I0.0) thì hệ thống bắt đầu làm việc với công việc đầu tiên là bơm hai loại sơn vào bình. Khi đã sơn có trong bình thì I0.5 có mức 0, nhưng chưa đầy bình thì I0.4 đang có mức 0. Khi lượng sơn đủ (đạt mức cao) thì I0.4 có mức 1, cho tín hiệu khởi động động cơ trộn (Q0.2) và bộ Timer T37 dùng để khống chế thời gian trộn. Khi đạt đến thời gian chỉnh định của T37 thì động cơ trộn sẽ được cắt điện và đóng điện mở van (Q0.4) và bơm tháo sản phẩm ra (Q0.5). Khi toàn bộ sơn thành phẩm được lấy hết khỏi bình thì I0.5 có mức 1, cho tín hiệu khởi động bộ đếm tiến C30 và reset bộ timer T37 và hệ thống lại tiếp tục lặp lại quá trình (chu kỳ mới). Khi đạt số chu kỳ làm việc cần thiết (đặt trước bằng C30) thì hệ thống tự động dừng. Trong quá trình làm việc có thể dừng hệ thống nếu cần nhờ nút ấn dừng I0.1. Để xóa giá trị bộ đếm ta sử dụng nút ấn với địa chỉ là I0.7.

<u>Chương trình viết trong STL như sau :</u>

```
NETWORK 1
LDN     I0.1
LD      I0.0
O       Q0.0
ALD
=       M0.0

NETWORK 2
LD      M0.0
AN      I0.4
AN      C30
AN      T37
=       Q0.0
=       Q0.1

NETWORK 3
LD      M0.0
A       I0.4
S       M0.1 ,      1

NETWORK 4
LD      M0.0
A       M0.1
TON     T37 ,       +100

NETWORK 5
LD      M0.0
A       I0.0
AN      T37
=       Q0.2

NETWORK 6
LD      M0.0
A       T37
AN      I0.5
=       Q0.4
=       Q0.5

NETWORK 7
LD      M0.0
A       T37
LD      I0.7
CTU     C30 ,       +10

NETWORK 8
LD      M0.0
A       I0.5
R       M0.1 ,      1
```

CHƯƠNG 6: TỔNG QUAN VỀ PLC VÀ CẤU TRÚC HỌ PHẦN CỨNG PLCS7-300 CỦA HÃNG SIEMENS

1.1. GIỚI THIỆU CHUNG VỀ PLC

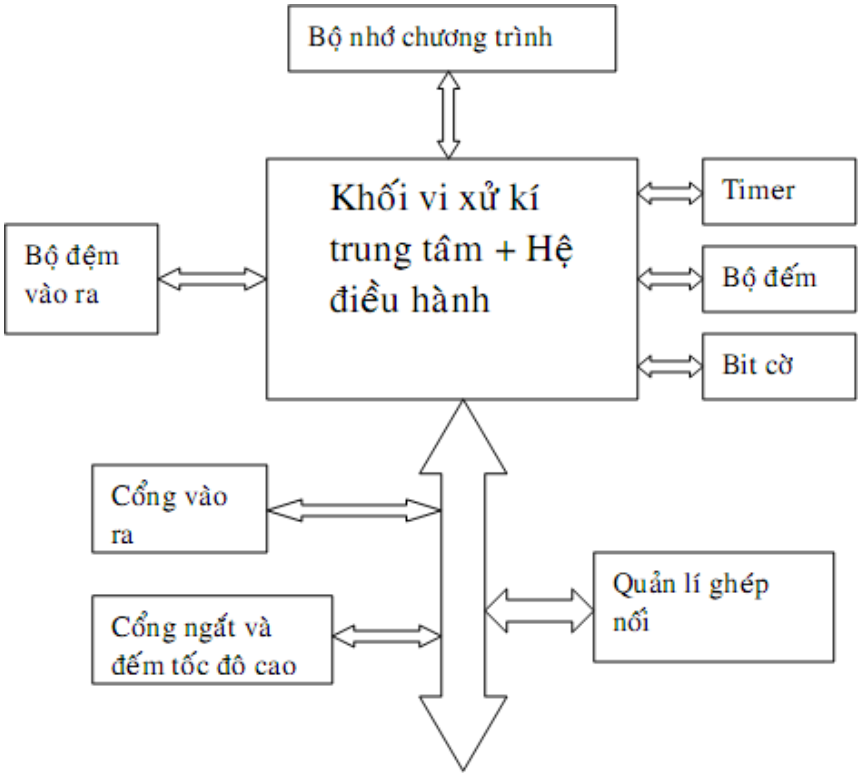
1.1.1. Mở đầu

Sự phát triển kỹ thuật điều khiển tự động hiện đại và công nghệ điều khiển logic khả trình dựa trên cơ sở phát triển của tin học mà cụ thể là sự phát của kỹ thuật máy tính.

Kỹ thuật điều khiển logic khả trình PLC (Programmable Logic Control) được phát triển từ những năm 1968 - 1970. Trong giai đoạn đầu các thiết bị khả trình yêu cầu người sử dụng phải có kỹ thuật điện tử, phải có trình độ cao. Ngày này các thiết bị PLC đã phát triển mạnh mẽ và có mức độ phổ cập cao.

PLC (Programmable Logic Control): Thiết bị điều khiển logic khả trình PLC. Là loại thiết bị cho phép điều khiển linh hoạt các thuật toán điều khiển số thông qua một ngôn ngữ lập trình, thay cho việc phải thể hiện mạch toán đó trên mạch số. Như vậy với chương trình điều khiển trong mình, PLC trở thành bộđiều khiển nhỏ gọn, dễ thay đổi thuật toán và đặc biệt dễ trao đổi thông tin với môi trường xung quanh (với các PLC khác hay với máy tính).

Để có thể thực hiện một chương trình điều khiên, PLC phải có tính năng như một máy tính. Nghĩa là phải có một bộ vi xử lý trung tâm (CPU), một hệ điều hành, một bộ nhớ chương trình để lưu chương trình cũng như dữ liệu và tất nhiên phải có các cổng vào ra để giao tiếp với các thiết bị bên ngoài. Bên cạnhđó, nhằm phục vụ các bài toán điều khiển số, PLC phải có các khối hàm chứcnăng như Timer, Counter, và các hàm chức năng đặc biệt khác.



Hình 1.1. Sơ đồ khối của PLC

Các PLC tương tự máy tính, nhưng máy tính được tối ưu hóa cho các nhiệm vụ tính toán và hiển thị còn PLC được chuyên biệt cho các nhiệm vụ điều khiển và môi trường công nghiệp. Vì vậy các PLC được thiết kế:

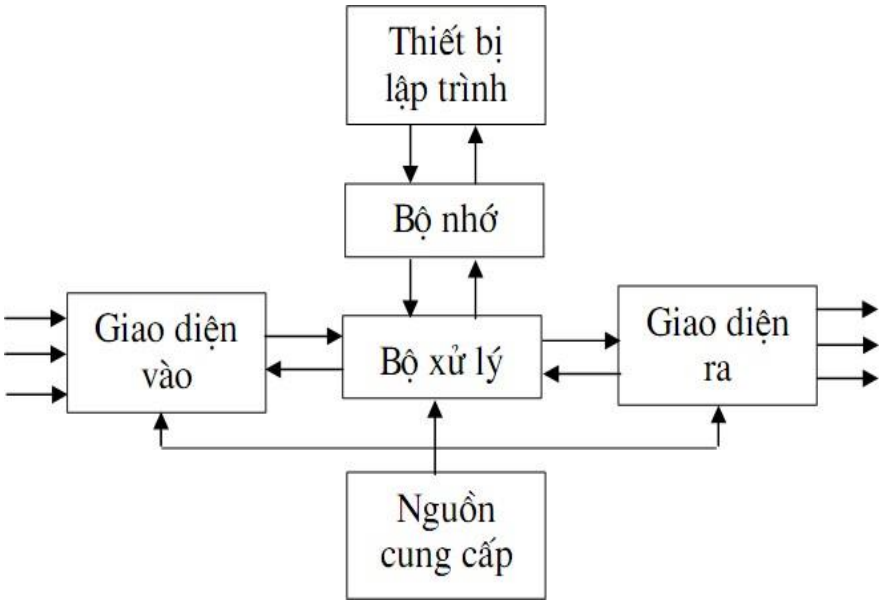
- Để chịu được các rung động, nhiệt độ, độ ẩm, bụi bẩn và tiếng ồn.
- Có sẵn giao diện cho các thiết bị vào ra.
- Được lập trình dễ dàng với ngôn ngữ lập trình dễ hiểu, chủ yếu giải quyết các phép toán logic và chuyển mạch.

Về cơ bản chức năng của bộ điều khiển logic PLC cũng giống chức năngcủa bộ điều khiển thiết kế trên cơ sở rơle công tắc tơ hay trên cơ sở các khối điện tử đó là:

- Thu thập các tín hiệu vào và các tín hiệu phản hồi từ cảm biến.
- Liên kết, ghép nối các tín hiệu theo yêu cầu điều khiển và thực hiện đóng mở các mạch phù hợp với công nghệ.
- Tính toán và soạn thảo các lệnh điều khiển đến các địa chỉ thích hợp.

1.1.2. Các thành phần cơ bản của một bộ PLC

Hệ thống PLC thông dụng có năm bộ phận cơ bản gồm: Bộ xử lý, bộ nhớ,bộ nguồn, giao diện vào ra và các thiết bị lập trình. Sơ đồ hệ thống như sau:



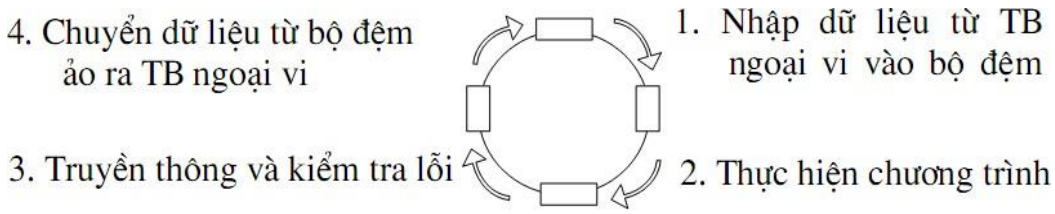
Hình 1.2. Sơ đồ hệ thống của PLC

a. Bộ xử lý:

Bộ xử lý còn gọi là bộ xử lý trung tâm (CPU) là linh kiện chứa bộ vi xử lý. Bộ xử lý nhận các tín hiệu vào và thực hiện các hoạt động điều khiển theo chương trình được lưu trong bộ nhớ của CPU, truyền các quyết định dưới dạng tín hiệu hoạt động đến các thiết bị ra.

Nguyên lý làm việc của bộ xử lý tiến hành theo từng bước tuần tự. Đầu tiên các thông tin lưu trữ trong bộ nhớ chương trình được gọi tên tuần tự và được kiểm soát bởi bộ đếm chương trình. Bộ xử lý liên kết các tín hiệu và đưa kết quả

ra đầu ra. Chu kỳ thời gian này gọi là thời gian quét (scan). Thời gian vòng quét phụ thuộc vào tầm vóc bộ nhớ, tốc độ của CPU. Chu kỳ một vòng quét có hình như hình 1.3:



Hình 1.3. Chu kỳ một vòng quét

Sự thao tác tuần tự của chương trình dẫn đến một thời gian trễ trong khibộ đếm của chương trình đi qua một chu kỳ đầy đủ, sau đó lại bắt đầu lại từ đầu.

Để đánh giá thời gian trễ người ta đo thời gian quét của một chương trình dài 1 Kbyte và coi đó là chỉ tiêu để so sánh các PLC. Với nhiều loại thiết bị thời gian trễ này có thể tới 20ms hoặc hơn. Nếu thời gian trễ gây trở ngại cho quá trình điều khiển thì phải dùng các biện pháp đặc biệt, chẳng hạn như lặp lại những lần gọi quan trọng trong thời gian một lần quét, hoặc là điều khiển các thông tin chuyển giao để bỏ bớt đi những lần gọi sit quan trọng khi thời gian quét dài tới mức không thể chấp nhận được. Nếu các biện pháp trên không thỏa mãnthì phải dùng PLC có thời gian quét ngắn hơn.

b. Bộ nguồn:

Bộ nguồn có nhiệm vụ chuyển đổi điện áp AC thành điện áp thấp cho bộvi xử lý (thường là 5VDC) và cho các mạch điện cho các module còn lại (thường là 24V).

c. Thiết bị lập trình:

Thiết bị lập trình được sử dụng để lập các chương trình điều khiển cần thiết sau đó được chuyển cho PLC. Thiết bị lập triofnh có thể là thiết bị lập trình chuyên dụng, có thể là thiết bị cầm tay gọn nhẹ, có thể là phần mềm được cài đặt trên máy tính cá nhân.

d. Bộ nhớ:

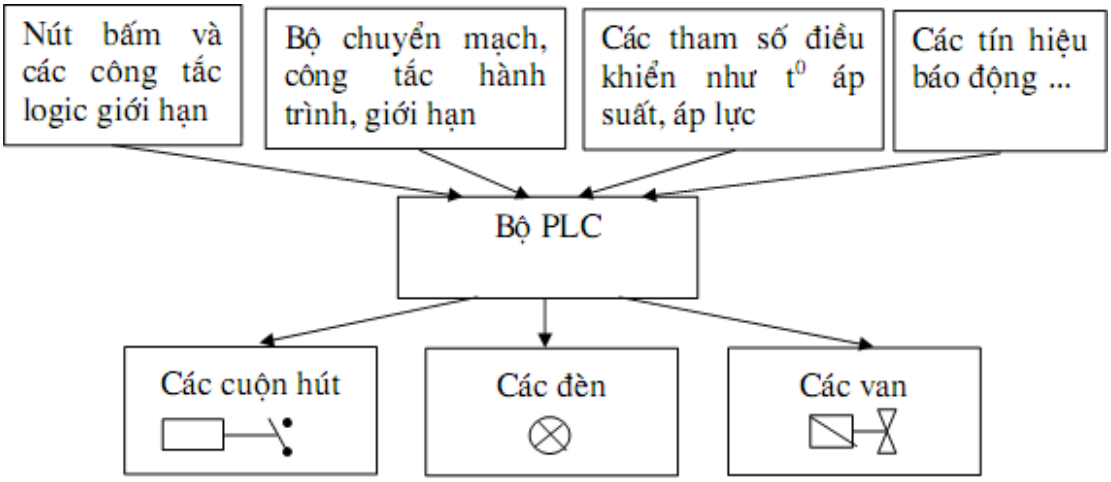
Bộ nhớ là nơi lưu trữ chương trình sử dụng cho các hoạt động điều khiển. Các dạng bộ nhớ có thể là RAM, ROM, EPROM. Người ta luôn chế tạo nguồn dự phòng cho RAM để duy trì chương trình trong trường hợp mất điện nguồn, thời gian duy trì tùy thuộc vào

từng PLC cụ thể. Bộ nhớ cũng có thể được chếtạo thành module cho phép dễ dàng thích nghi với các chức năng điều khiển kíchcỡ khác nhau, khi cần mở rộng có thể cắm thêm.

e. Giao diện vào/ra:

Giao diện vào là nơi bộ xử lý nhận thông tin từ các thiết bị ngoại vi và truyền thông tin đến các thiết bị bên ngoài. Tín hiệu vào có thể từ các công tắc,bộ cảm biến nhiệt độ, các tế bào quang điện,… Tín hiệu ra có thể cung cấp cho các cuộn dây công tắc tơ, các rơle, các van điện từ, các động cơ nhỏ,… Tín hiệu vào/ra có thể là các tín hiệu rời rạc, tín hiệu liên tục, tín hiệ logic,… Các tín hiệu vào/ra có thể biểu hiện như sau:

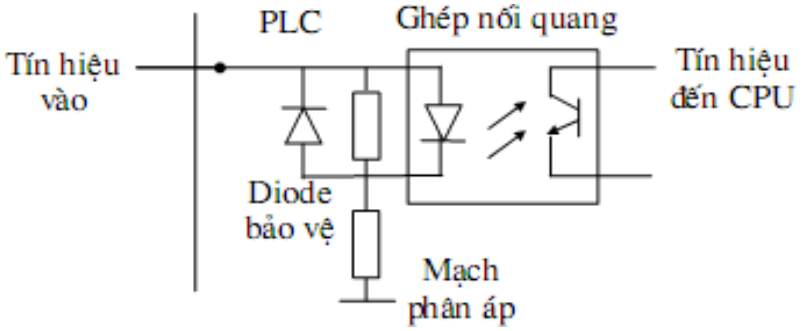
Mỗi điểm vào/ra có một địa chỉ



Hình 1.4. Giao diện vào ra của PLC

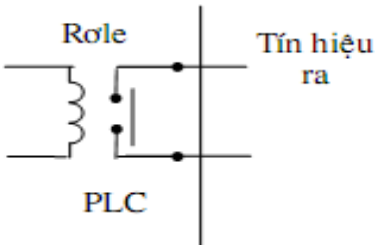
Các kênh vào ra đã có chức năng cách ly và điều hóa tín hiệu sao cho các bộ cảm biến và các bộ tác động có thể nối trực tiếp với chúng mà không cần thêm mạch điện khác.

Tín hiệu vào thường được ghép cách điện (cách ly) nhờ linh kiện quang như hình 1.5. Dải tín hiệu nhận vào cho các PLC cỡ lớn có thể là 5V, 24V, 110V, 220V. Các PLC cỡ nhỏ chỉ nhập tín hiệu 24V.

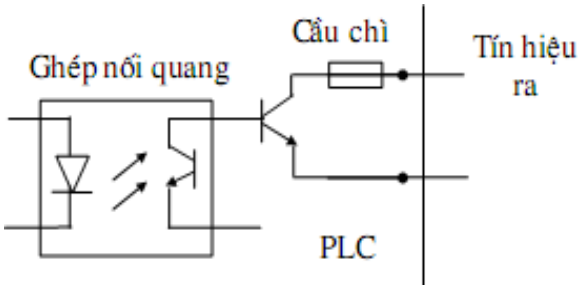


Hình 1.5. Mạch cách ly tín hiệu vào

Tín hiệu ra cũng được ghép cách ly, tín hiệu ra cũng được cách ly kiểu rơle như hình 1.6 hay cách ly kiểu quang như hình 1.7. Tín hiệu ra có thể là tín hiệu chuyển mạch 24V, 100mA; 110V, 1A một chiều; thậm chí 240V, 1A xoay chiều tùy loại PLC. Tuy nhiên, với PLC cỡ lớn dải tín hiệu ra có thể thay đổi bằng cách lựa chọn các module ra thích hợp.



Hình 1.6. Mạch cách ly tín hiệu ra kiểu rơ le



Hình 1.7. Mạch cách ly tín hiệu ra kiểu quang

1.1.3. Đánh giá ưu nhược điểm của PLC

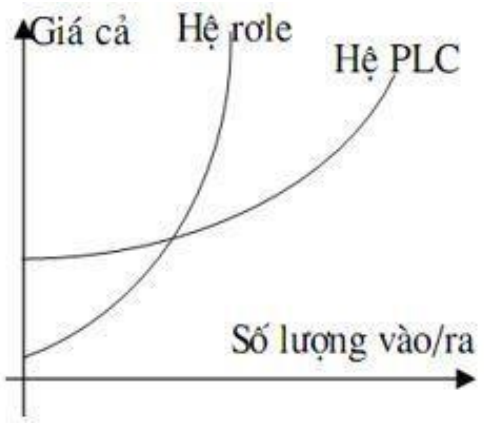
Trước đây, bộ PLC thường rất đắt, khả năng hoạt động bị hạn chế và quy trình lập trình phức tạp. Vì những lý do đó mà PLC chỉ được dùng trong những nhà máy và các thiết bị đặc biệt. Ngày nay, do giá thành hạ kèm theo tăng khả năng của PLC dẫn đến PLC ngày càng được áp dụng rộng cho các thiết bị máy móc. Các bộ PLC đơn khối với 24 kênh đầu vào và 26 kênh đầu ra thích hợp vớicác máy tiêu chuẩn đơn, các trang thiết bị liên hợp. Còn các bộ PLC với nhiều khả năng ứng dụng và lựa chọn được dùng cho những nhiệm vụ phức tạp hơn. Có thể kể ra các ưu điểm của PLC như sau:

- Chuẩn bị vào hoạt động nhanh: Thiết kế kiểu module cho phép thích nghi nhanh với mọi chức năng điều khiển. Khi đã được lắp ghép thì PLC sẵn sàng làm việc ngay. Ngoài ra nó còn được sử dụng lại cho các ứng dụng khác dễ dàng.
- Độ tin cậy cao: Các linh kiện điện tử có tuổi thọ dài hơn các thiết bị cơ điện. Độ tin cậy của PLC ngày càng tăng, bảo dưỡng định kỳ thường không cần thiết còn với mạch rơle công tắc tơ thì việc bảo dưỡng định kỳ là cần thiết.
- Dễ dàng thay đổi chương trình: Việc thay đổi chương trình được tiến hành đơn giản. Để sửa đổi hệ thống điều khiển và các quy tắc điều khiển đang được sử dụng, người vận hành chỉ cần nhập tập lệnh khác,

gần như không cần mắc nối lại dây. Nhờ đó hệ thống rất linh hoạt và hiệu quả.

- Đánh giá nhu cầu đơn giản: Khi biết các đầu vào và đầu ra thì có thể đánh giá được kích cỡ yêu cầu của bộ nhớ hay độ dài chương trình. Do đó có thể dễ dàng và nhanh chóng lựa chọn PLC phù hợp với các yêu cầu công nghệ đặt ra.
- Khả năng tái tạo: Nếu dùng PLC với quy cách kỹ thuật giống nhau thì chi phí lao động sẽ giảm thấp hơn nhiều so với bộ điều khiển rơle. Đó là do giảm phần lớn lao động lắp ráp.
- Tiết kiệm không gian: PLC đòi hỏi ít không gian hơn so với bộ điều khiển rơle tương đương.

- Có tính chất nhiều chức năng: PLC có ưu điểm chính là có thể sử dụng cùng một thiết bị điều khiển cơ bản cho nhiều hệ thống điều khiển. Người ta thường dùng PLC cho các quá trình tự động linh hoạt vì dễ dàng thuận tiện trong tính toán, so sánh các giá trị tương quan, thay đổi chương trình và thay đổi thông số.
- Về giá trị kinh tế: Khi xét về giá trị kinh tế của PLC ta phải đề cập đến số lượng đầu vào và đầu ra. Quan hệ về giá thành với số lượng đầu vào và đầu ra có dạng như hình 1.8. Như vậy, nếu số lượng đầu vào/ra quá ít thì hệ rơle ra kinh tế hơn, nhưng khi số lượng đầu vào/ra tăng lên thì hệ PLC kinh tế hơn hẳn.



Hình 1.8. Quan hệ giữa số lượng vào/ra và giá thành

Có thể so sánh hệ điều khiển rơle và hệ điều khiển PLC như sau:

- Hệ rơle:
  - Nhiều bộ phận đã được chuẩn hóa.
  - Ít nhạy cảm với nhiễu..
  - Kinh tế với các hệ thống nhỏ.
  - Thời gian lắp đặt lâu.
  - Thay đổi khó khăn.
  - Kích thước lớn.
  - Cần bảo quản thường xuyên.
  - Khó theo dõi và kiểm tra các hệ thống lớn, phức tạp.
- Hệ PLC:
  - Thay đổi dễ dàng.
  - Lắp đặt đơn giản.
  - Thay đổi nhanh quy trình điều khiển.
  - Kích thước nhỏ.
  - Có thể nối với mạng máy tính.
  - Giá thành cao.
  - Bộ thiết bị lập trình thường đắt, sử dụng ít.

1.1.4. Ứng dụng của hệ thống sử dụng PLC

Từ các ưu điểm trên, hiện nay PLC đã được ứng dụng trng rất nhiều lĩnhvực khác nhau trong công nghiệp như:

- Hệ thống nâng vận chuyển.
- Dây chuyền đóng gói.
- Các robot lắp ráp sản phẩm.
- Điều khiển bơm.
- Dây chuyền xử lý hóa học.
- Công nghệ sản xuất giấy.
- Dây chuyền sản xuất thủy tinh.
- Sản xuất xi măng.
- Công nghệ chế biến sản phẩm.
- Điều khiển hệ thống đèn giao thông.
- Quản lý tự động bãi đỗ xe.
- Hệ thống may công nghiệp.
- Điều khiển thang máy.

1.2. GIỚI THIỆU VỀ BỘ ĐIỀU KHIỂN S7-300

1.2.1. Giới thiệu chung

Từ khi ngành công nghiệp sản xuất bắt đầu phát triển, để điều khiển một dây chuyền, một thiết bị máy móc công nghiệp nào,… Người ta thường thực hiện kết nối các linh kiện điều khiển riêng lẻ (rơle, timer, contactor,…) lại vớinhau tùy theo mức độ yêu cầu thành một hệ thống điện điều khiển đáp ứng nhucầu mà bài toán công nghệ đặt ra.

Công việc này diễn ra khác phức tạp trong thi công vì phải thao tác chủ yếu trong việc đấu nối, lắp đặt mất khác nhiều thời gian mà hiệu quả lại không cao vì một thiết bị có thể cần được lấy tín hiệu nhiều lần mà số lượng lại rất hạn

chế, bởi vậy lượng vật tư là rất nhiều đặc biệt trong quá trình sửa chữa bảo trì, hay cần thay đổi quy trình sản xuất gặp rất nhiều khó khăn và mất rất nhiều thời gian trong việc tìm kiếm hư hỏng và đi lại dây bởi vậy năng suất lao động giảmđi rõ rệt.

Với những nhược điểm trên các nhà khoa học, nhà nghiên cứu đã nỗ lựcđể tìm ra một giải pháp điều khiển tối ưu nhất đáp ứng mong mỏi của ngành công nghiệp hiện đại đó là tự động hóa quá trình sản xuất làm giảm sức lao động, giúp người lao động không phải làm việc ở những khu vực nguy hiểm, độc hại,… mà năng suất lao động lại tăng cao gấp nhiều lần.

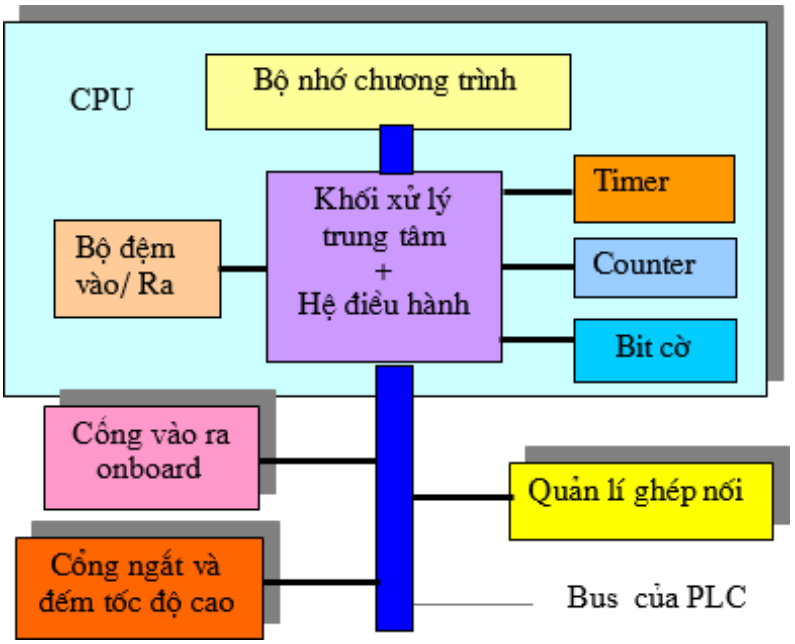
Một hệ thống điều khiển ưu việt mà chúng ta phải chọn để điều khiển cho ngành công nghiệp hiện đại cần phải hội tụ đủ các yếu tố sau: Tính tự động cao, kích thước và khối lượng nhỏ gọn, giá thành hạ, dễ thi công, sửa chữa, chất lượng làm việc ổn định linh hoạt,…

Từ đó hệ thống điều khiển có thể lập trình được PLC (Programmable Logic Control) ra đời đầu tiên năm 1968 (Công ty General Motors – Mỹ). Tuy nhiên hệ thống này còn khá đơn giản và cồng kềnh, người sử dụng gặp nhiều khó khăn trong việc vận hành hệ thống, vì vậy qua nhiều năm cải tiến và phát triển không ngừng khắc phục những nhược điểm còn tồn tại để có được bộ điều khiển PLC như ngày nay, đã giải quyết được các vấn đề nêu trên với các ưu việt như sau:

- Là bộ điều khiển số nhỏ gọn, dễ thay đổi thuật toán điều khiển.
- Có khả năng mở rộng các module vào ra khi cần thiết.
- Ngôn ngữ lập trình dễ hiểu thích hợp với nhiều đối tượng lập trình.
- Có khả năng truyền thông đó là trao đổi thông tin với môi trường xung quanh như với máy tính, các PLC khác, các thiết bị giám sát, điều khiển,...

- Có khả năng chống nhiễu với độ tin cậy cao và có rất nhiều ưu điểm khác nữa.

Hiện nay trên thế giới đang song hành có nhiều hãng PLC khác nhau cùng phát triển như hãng Omron, Mitsubishi, Hitachi, ABB, Siemens,... và có nhiều hãng khác nữa nhưng chúng đều có chung nguyên ly cơ bản chỉ có vài điểm khác biệt với từng mặt mạnh riêng của từng ngành mà người sử dụng sẽ quyết định nên dùng hãng PLC nào cho thích hợp với mình mà thôi. Chúng ta đi sâu vào tìm hiểu chi tiết loại PLC S7-300 của hãng Siemens sản xuất đang được sử dụng khápho biến hiện nay.



Hình 1.9. Miêu tả nguyên lý chung về cấu trúc PLC

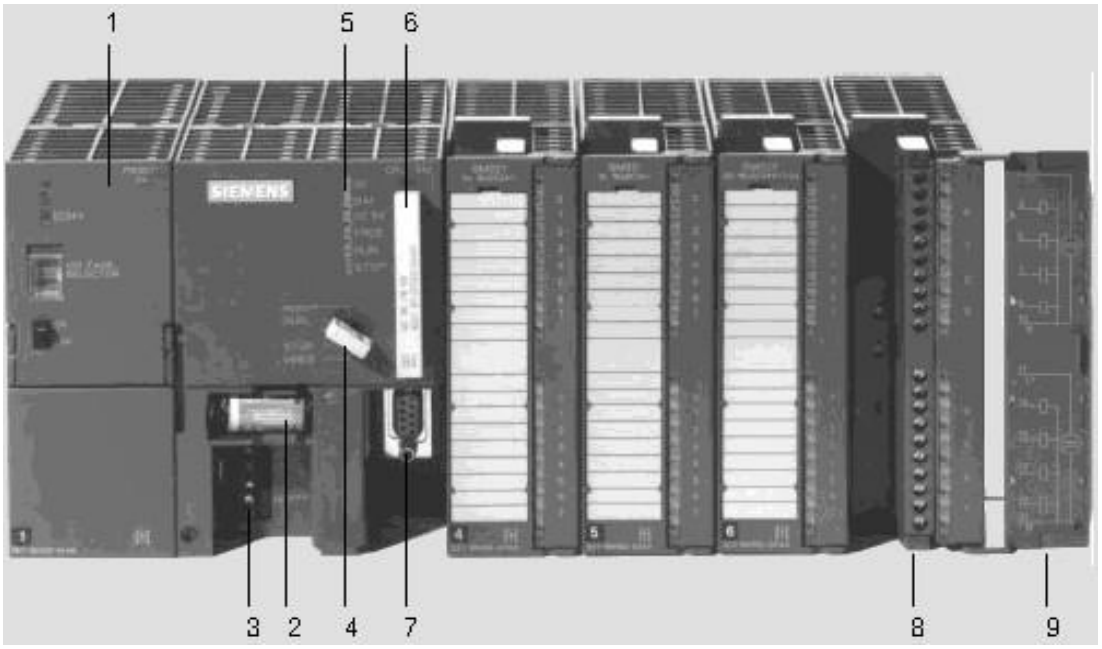
Để thực hiện được một chương trình điều khiển thì PLC cũng phải có chức năng như là một chiếc máy tính nghĩa là phải có bộ vi xử lý (CPU), một hệ điều hành, bộ nhớ để lưu chương trình điều khiển, dữ liệu và có các cổng vào/ra để còn trao đổi thông tin với môi trường bên ngoài. Ngoài ra để thực hiện các bài

toán điều khiển số thì PLC còn có các bộ Timer, Counter và các loại hàm chuyêndụng khác nữa,... Đã tạo thành một bộ điều khiển rất linh hoạt.

1.2.2. Các module của PLC S7-300

Trong quá trình các ứng dụng thực tế thì với mỗi bài toán điều khiển đặt ra là hoàn toàn khác nhau bởi vậy việc lựa chọn chủng loại các thiết bị phần cứng là cũng khác nhau, sao cho phù hợp với yêu cầu mà không gây lãng phí tiền của.

Vì vậy việc chọn lựa các CPU và các thiết bị vào ra là không giống nhau. Bởi vậy PLC đã được chia nhỏ ra thành các module riêng lẻ để cho PLC khôngbị cứng hóa về cấu hình. Số các module được sử dụng nhiều hay ít tùy thuộc từng yêu cầu của bài toán đặt ra nhưng tối thiểu phải có module nguồn nuôi, module CPU còn các module còn lại là các module truyền nhận tín hiệu với môi trường bên ngoài, ngoài ra còn có các module có chức năng chuyên dụng như PID, điều khiển mờ, điều khiển động cơ bước, các module phục vụ cho các chức năng truyền thông,... Tất cả các module kể trên được gắn trên một thanh Rack.



Hình 1.10. Miêu tả về cấu hình PLC S7-300

Trong đó:

1. Là nguồn nuôi cho PLC.
2. Là pin lưu trữ (cho CPU 313 trở lên).
3. Đầu nối 24VDC.
4. Công tắc chọn chế độ làm việc.
5. Đèn LED báo trạng thái và báo lỗi.
6. Card nhớ (cho CPU 313 trở lên).
7. Cổng truyền thông (RS485) kết nối với thiết bị lập trình.
8. Vị trí đấu nối với các thiết bị điều khiển bên ngoài.
9. Nắp đậy bảo vệ trong khi làm việc.

1.2.2.1. Module CPU

Module CPU loại module có chứa bộ vi xử lý, hệ điều hành, bộ nhớ, cácbộ thời gian, bộ đếm, cổng truyền thông (RS485),... Và có thể còn có một vài cổng vào ra số. Các cổng vào ra số có trên module CPU được gọi là các cổng vào ra Onboard.

Trong họ PLC S7-300 có nhiều loại module CPU khác nhau, được đặt tên theo bộ vi xử lý có trong nó như module CPU 312, module CPU 314, module CPU 315,...

Hình 1.11. Miêu tả hình dáng của hai CPU 314 và CPU 314IFM

Những module này cùng sử dụng một bộ vi xử lý nhưng khác nhau về cổng vào/ra Onboard cũng như các khối hàm đặc biệt được tích hợp sẵn trong

thư viện của hệ điều hành phục vụ việc sử dụng các cổng vào/ra Onboard này được phân biệt với nhau trong tên gọi bằng cụm từ chữ cái IFM (Intergrated Funtion Module). Ví dụ như CPU 312IFM, CPU 314IFM,...

Ngoài ra còn có các loại module CPU với hai cổng truyền thông, trong đó cổng truyền thông thứ hai có chức năng chính là phục vụ việc nối mạng phân tán. Các loại module CPU này được phân biệt với các loại CPU khác bằng cụm từ DP (Distributed Port). Ví dụ như CPU 315DP.

1.2.2.2. Module nguồn

Module PS (Power Supply). Module nguồn nuôi có 3 loại với các thông số đó là 2A, 5A, 10A.

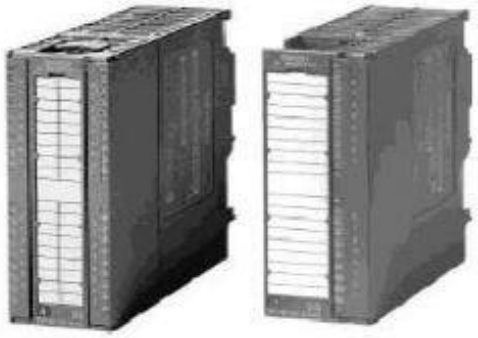
Ví dụ: PS 307-2A, PS 307-5A, PS 307-10A.

Hình 1.12. Miêu tả hình dáng module nguồn nuôi PS307

1.2.2.3. Module mở rộng

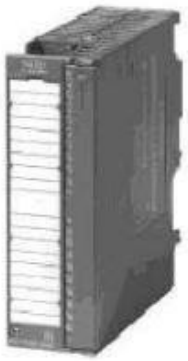
Module SM (Signal Module). Module mở rộng cổng tín hiệu vào/ra bao gồm:

- DI (Digital Input): Module mở rộng các cổng vào số. Số các cổng vào số mở rộng có thể là 8, 16 hoặc 32 tùy thuộc từng loại module.



Hình 1.13. Miêu tả hình dáng module SM321 DI 32 point 24VDC

- DO (Digital Output): Module mở rộng các cổng ra số. Số các cổng ra số mở rộng có thể là 8, 16 hoặc 32 tùy thuộc từng loại module.
- DI/DO (Digital Input/Digital Output): Module mở rộng các cổng vào/ra số. Số các cổng vào/ra số mở rộng có thể là 8 vào/8 ra hoặc 16 vào/16 ra tùy thuộc từng loại module.
- AI (Analog Input): Module mở rộng các cổng vào tương tự. Về bản chất chúng là những bộ chuyển đổi tương tự sang số 12 bit (AD), tức là mỗi tín hiệu tương tự được chuyển thành một tín hiệu số (nguyên) có độ dài 12 bit. Số các cổng vào tương tự có thể là 2, 4 hoặc 8 tùy thuộc từng loại module.



Hình 1.14. Miêu tả hình dáng module SM332 AI 8x12 bit

- AO (Analog Output): Module mở rộng các cổng ra tương tự. Chúng thực chất là những bộ chuyển đổi tín hiệu số sang tương tự (DA). Số các cổng ra tương tự có thể là 2, 4 hoặc 8 tùy thuộc từng loại module.
- AI/AO (Analog Input/Analog Output): Module mở rộng các cổng vào/ra tương tự. Số các cổng vào/ra tương tự có thể là 2, 4 tùy thuộc vào từng loại module.

1.2.2.4. Module ghép nối

Module IM (Interface Module): Module ghép nối. Đây là loại module chuyên dụng có nhiệm vụ nối từng nhóm các module mở rộng lại với nhau thành một khối và được quản lý chung bởi một module CPU. Các module mở rộng được gá trên một thanh rack. Trên mỗi rack có thể gá được tối đa 8 module mở rộng (Không kể module CPU và module nguồn nuôi). Một module CPU S7-300 có thể làm việc trực tiếp được với nhiều nhất 4 racks và các rack này phải đượcnối với nhau bằng module IM. Các module này ở các rack mở rộng có thể cần được cung cấp nguồn cho hệ thống rack đó ngoài ra tùy thuộc vào từng loại module IM mà có thể cho phép được mở rộng tối đa đến 4 racks ví dụ IM360 chỉ cho mở rộng tối đa là với 1 module.

Hình 1.15. Miêu tả hình dáng module IM361

Module FM (Funtion Module): Module có chức năng điều khiển riêng, vídụ như module điều khiển động cơ bước, module điều khiển động cơ servo, module PID, module điều khiển vòng kín,...

Module CP (Communication Module): Module phục vụ truyền thông trong mạng giữa các PLC với nhau hoặc giữa PLC với máy tính.

1.2.3. Kiểu dữ liệu và phân chia bộ nhớ

1.2.3.1. Kiểu dữ liệu

Trong một chương trình có thể có các kiểu dữ liệu sau:

BOOL: Với dung lượng 1 bit và có giá trị là 0 hay 1. Đây là kiểu dữ liệucó biến 2 trị.

BYTE: Gồm 8 bit, có giá trị nguyên dương từ 0 đến 225. Hoặc mã ASCIIcủa một ký tự.

WORD: Gồm 2 byte, có giá trị nguyên dương từ 0 đến 65535.

INT: Có dung lượng 2 byte, dùng để biểu diễn số nguyên từ -32768 đến32767.

DINT: Gồm 4 byte, biểu diễn số nguyên từ -2147463846 đến 2147483647.

REAL: Gồm 4 byte, biểu diễn số thực dấu phẩy động.

S5T: Khoảng thời gian, được tính theo giờ/phút/giây/mili giây.TOD: Biểu diễn giá trị thời gian tính theo giờ/phút/giây.

DATE: Biểu diễn giá trị thời gian tính theo năm/tháng/ngày. CHAR: Biểu diễn một hoặc nhiều ký tự (nhiều nhất là 4 ký tự).

1.2.3.2. Phân chia bộ nhớ

Bộ nhớ trong PLC S7-300 có 3 vùng nhớ cơ bản sau:

- Vùng chứa chương trình ứng dụng:
  - OB (Organisation Block): Miền chứa chương trình tổ chức.
  - FC (Function): Miền chứa chương trình con được tổ chức thành hàm có biến hình thức để trao đổi dữ liệu với chương trình đã gọi nó.
  - FB (Function Block): Miền chứa chương trình con được tổ chức thành hàm có khả năng trao đổi dữ liệu với bất cứ một khối chương trình nào khác, các dữ liệu này được xây dựng thành một khối dữ liệu riêng (DB – Data Block).
- Vùng chứa tham số của hệ điều hành và các chương trình ứng dụng. Được chia thành 7 miền khác nhau bao gồm:
  - I (Process Input Image): Miền bộ đệm các dữ liệu cổng vào số. Trước khi bắt đầu thực hiện chương trình, PLC sẽ đọc giá trị logic của tất cả các đầu vào và cất giữ chúng trong vùng nhớ I. Thông thường chương trình ứng dụng không đọc trực tiếp trạng thái logic của cổng vào số mà chỉ lấy dữ liệu của cổng vào từ bộ đệm I.
  - Q (Process Output Image): Miền bộ đệm các dữ liệu cổng ra số. Kết thúc giai đoạn thực hiện chương trình, PLC sẽ chuyển giá trị logic của bộ đệm Q tới các cổng ra số. Thông thường chương trình không trực tiếp gán giá trị tới tận cổng ra mà chỉ chuyển chúng vào bộ đệm Q.

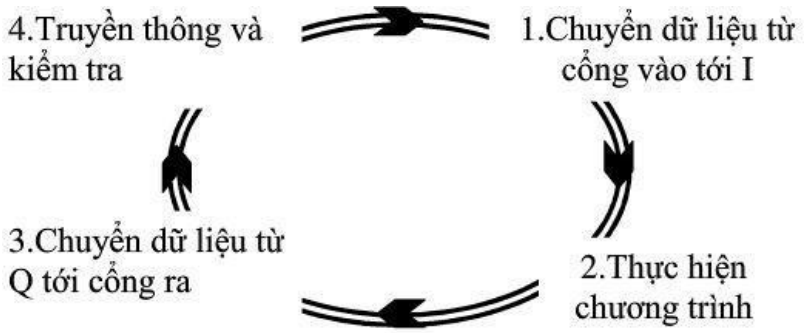
- M: Miền các biến cờ. Chương trình ứng dụng sử dụng vùng nhớ này để lưu trữ các tham số cần thiết và có thể truy nhập nó theo bit (M), byte (MB), từ (MW), từ kép (MD).
- T (Timer): Miền nhớ phục vụ bộ định thời bao gồm việc lưu trữ các giá trị thời gian đặt trước (PV-Preset Value), giá trị đếm thời gian tức thời (CV-Current Value) cũng như giá trị logic đầu ra của bộ thời gian.
- C (Counter): Miền nhớ phục vụ bộ đếm bao gồm việc lưu trữ giá trị đặt trước (PV-Preset Value), giá trị đếm tức thời (CV-Current Value) và giá trị logic của bộ đếm.
- PI (I/O External Input): Miền địa chỉ cổng vào của các module tương tự. Các giá trị tương tự tại cổng vào của module tương tự sẽ được module đọc và chuyển tự động theo những địa chỉ.
- PQ (I/O External Output): Miền địa chỉ cổng ra của các module tương tự. Các giá trị tương tự tại cổng ra của module tương tự sẽ được module đọc và chuyển tự động theo những địa chỉ.

- Vùng chứa các khối dữ liệu. Được chia làm hai loại:
  - DB (Data Block): Miền chứa các dữ liệu được tổ chức thành khối. Kích thước cũng như số lượng khối do người sử dụng quy định, phù hợp với từng bài toán điều khiển. Chương trình có thể truy cập miền này theo từng bit (DBX), byte (DBB), từ (DBW) hoặc từ kép (DBD).
  - L (Local Data Block): Miền dữ liệu địa phương, được các khối chương trình OB, FC, FB tổ chức và sử dụng cho các biện pháp tức thời và trao đổi dữ liệu của biến hình thức với những khối chương trình đã gọi nó. Nội dung của một số dữ liệu trong miền này sẽ bị

xóa khi kết thúc chương trình tương ứng trong OB, FC, FB. Miền này có thể truy nhập từ chương trình theo bit (L), byte (LB), từ (LW) hoặc từ kép (LD).

1.2.4. Vòng quét chương trình PLC S7-300

PLC thực hiện chương trình theo một chu trình lặp được gọi là vòng quét (scan). Một vòng lặp được gọi là một vòng quét. Có thể chia một chu trình thực hiện của S7-300 ra làm 4 giai đoạn. Giai đoạn một là giai đoạn đọc dữ liệu từ các cổng vào, các dữ liệu này sẽ được lưu trữ trên vùng đệm các đầu vào. Tiếp theo là giai đoạn thực hiện chương trình, trong từng vòng quét chương trình lần lượt thực hiện tuần tự từ lệnh đầu tiên và kết thúc ở lệnh cuối cùng tiếp đến là giaiđoạn chuyển nội dung các bộ đệm ảo tới cổng ra. Giai đoạn cuối cùng là giai đoạn truyền thông nội bộ và kiểm tra lỗi. Đến đây một vòng quét được hoàn thành và một vòng quét mới tiếp tục tạo nên một chu trình lặp vô hạn.



Hình 1.16. Miêu tả một vòng quét chương trình của S7-300

Một điểm cần chú ý là tại thời điểm thực hiện lệnh vào/ra thông thường các lệnh không làm việc trực tiếp với các cổng vào/ra mà chỉ thông qua bộ đệmảo của cổng trong vùng nhớ tham số. Chỉ khi gặp lệnh yêu cầu truy xuất các đầu vào/ra ngay lập tức thì hệ thống sẽ cho dừng các công việc khác, ngay cả chương trình xử lý ngắt để thực hiện lệnh này một cách trực tiếp với các cổng vào/ra.

Các chương trình con xử lý ngắt chỉ được thực hiện trong vòng quét khi xuất tín hiệu báo ngắt và có thể xảy ra bất cứ thời điểm nào trong vòng quét.

Bộ đệm I và Q không liên quan đến các cổng vào/ra tương tự nên các lệnh truy nhập tương tự được thực hiện trực tiếp với cổng vật lý chứ không qua bộ đệm.

Thời gian cần thiết để PLC thực hiện được một vòng quét gọi là thời gian vòng quét (Scan Time). Thời gian vòng quét không cố định, tức là không phải vòng quét nà cũng được thực hiện theo một khoảng thời gian như nhau. Các vòng quét nhanh, chậm phụ thuộc vào số lệnh trong chương trình được thực hiện, vào khối lượng dự x liệu được truyền thông... trong vòng quét đó.

Như vậy giữ việc đọc dữ liệu từ đối tượng để xử lý, tính toán và việc gửi tín hiệu điều khiển đến đối tượng đó có một khoảng thời gian trễ đúng bằng thời gian vòng quét. Thời gian vòng quét càng ngắn, tính thời gian thực của chương trình càng cao.

Nếu sử dụng các khối chương trình đặc biệt có chế độ ngắt, ví dụ như là OB40, OB80,... Chương trình của các khối đó sẽ được thực hiện trong vòng quétkhi xuất hiện tín hiệu báo ngắt cùng chủng loại. Nếu một tín hiệu báo ngắt xuất hiện khi PLC đang trong giai đoạn truyền thông và kiểm tra nội bộ, PLC sẽ dừng công việc truyền thông, kiểm tra để thực hiện khối chương trình tương ứng với tín hiệu báo ngắt đó. Với hình thức tín hiệu xử lý ngắt như vậy, thời gian củavòng quét càng lớn khi càng có nhiều tín hiệu ngắt xuất hiện trong vòng quét.

Do đó, để nâng cao tính thời gian thực của chương trình điều khiển, tuyệt đối không nên viết chương trình xử lý ngắt quá dài hoặc quá lạm dụng việc sử dụng chế độ ngắt trong chương trình điều khiển.

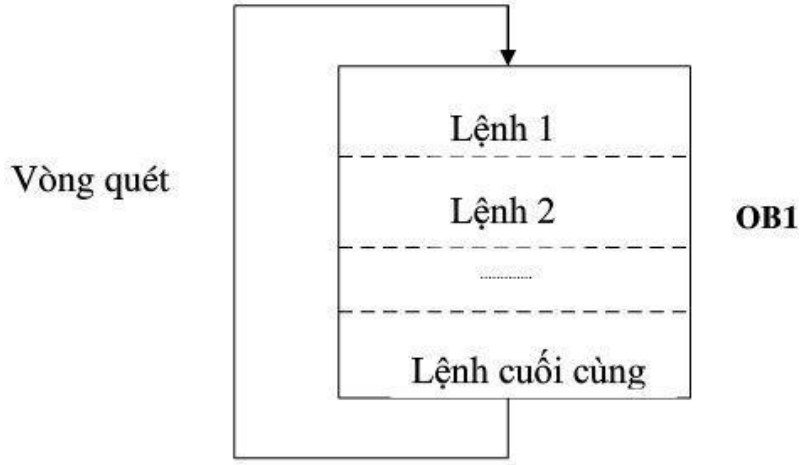
1.2.5. Cấu trúc chương trình của PLC S7-300

Các chương trình điều khiển PLC S7-300 được viết theo một trong haidạng sau: chương trình tuyến tính và chương trình có cấu trúc.

1.2.5.1. Lập trình tuyến tính

Toàn bộ chương trình điều khiển nằm trong một khối trong bộ nhớ. Loại hình cầu trúc tuyến tính này phù hợp với những bài toán tự động nhỏ, không phức tạp. Khối được chọn phải là khối OB1, là khối mà CPU luôn quét và thực hiện các lệnh trong nó thường xuyên, từ lệnh đầu tiên đến lệnh cuối cùng vàquay lại từ lệnh đầu tiên.

Hình 1.17. Miêu tả cách thức lập trình tuyến tính



1.2.5.2. Lập trình có cấu trúc

Trong PLC Siemens S7-300 chương trình được chia thành từng khối nhỏ mà có thể lập trình được với từng nhiệm vụ riêng. Loại hình cấu trúc này phùhợp với những bài toán điều khiển nhiều nhiệm vụ và phức tạp. PLC S7-300 có 4 loại khối cơ bản:

- Khối tổ chức OB (Oganization Block): Khối tổ chức và quản lý chương trình điều khiển.
- Khối hàm FC (Function): Khối chương trình với những chức năng riêng giống như một chương trình con hoặc một hàm.
- Khôi hàm chức năng FB (Function Block): Là một khối FC đặc biệt có khả năng trao đổi dữ liệu với các khối chương trình khác. Các dữ liệu

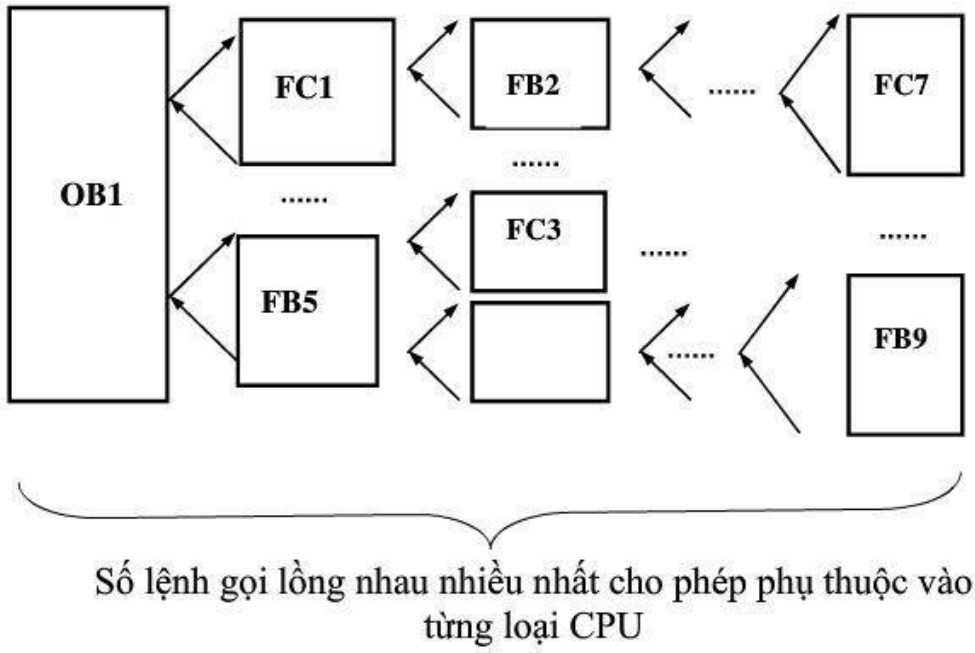
này phải được tổ chức thành khối dữ liệu riêng có tên gọi là Data Block(DB).

- Khối dữ liệu DB (Data Block): Khối chứa các dữ liệu cần thiết để thực hiện chương trình, các tham số khối do ta tự đặt. Khối dữ liệu dùng để chứa các dữ liệu của chương trình. Có hai loại DB: Shared DB (thang ghi DB) và Instance DB (thanh ghi DI).
- Khối Share DB (DB): Là khối dữ liệu có thể được truy cập bởi tất cả các khối trong chương trình đó.
- Khối Instance DB (DI): Là khối dữ liệu được gán cho một khối hàm duy nhất, dùng để chứa dữ liệu của khối hàm này.
- Khối SFC (System Function): Là các hàm được tích hợp trong hệ điều hành của CPU, các hàm này có thể được gọi bởi chương trình khi cần. Ngươi lập trình không thể tạo ra các SFC. Hàm được lập trình trước và tích hợp sẵn trong CPU S7. Ta có thể gọi SFC từ chương trình, vì những SFC là một phần của hệ điêu hành, ta không cần phải nạp chúng vào như một phần của chương trình.
- Khối SFB (System Function Block): Chức năng tương tư như SFC nhưng SFB cần DB tình huống như vậy FB vậy. Ta phải tải DB này xuống CPU như phần của chương trình.
- Khối SDB (System Data Block): Vùng nhớ của chương trình được tạo bởi các ứng dụng STEP7 khác nhau để chứa dữ liệu cần để điều hành PLC. Thí dụ: ứng dụng “S7 Configuration” cất dữ liệu cấu hình và các tham số làm việc khác trong các SDB, và ứng dụng “Communication Configuration” tạo tạo các SDB mà cất dữ liệu thông tin toàn cục được chia sẻ giữa các CPU khác nhau.

Chương trình trong lập trình có cấu trúc là các khối được liên kết lại với nhau bằng các lệnh gọi khối, chuyển khối. Xem như những phần chương trình trong các khối như là các chương trình con.

Trong S7-300 cho phép gọi chương trình con lồng nhau, tức là chương trình con này gọi từ một chương trình con khác và từ chương trình con được gọi gọi lại gọi đến chương trình con thứ 3... Số các lệnh gọi lồng nhau phụ thuộc vào từng chủng loại module CPU khác nhau mà ta đang sử dụng. Ví dụ như đối với module CPU 314 thì số lệnh gọi lồng nhau nhiều nhất có thể cho phép là 8. Nếusố lần gọi lồng nhau mà vượt quá con số giới hạn cho phép, PLC sẽ chuyển sangchế độ Stop và đặt cờ báo lỗi.

Hình 1.18. Miêu tả cách thức lập trình có cấu trúc



1.2.6. Các khối OB đặc biệt

Trong khi khối OB1 được thực hiện đều đặn ở từng vòng quét thì các khốiOB khác chỉ được thực hiện khi xuất hiện tín hiệu ngắt tương ứng, nói cách khác chương trình viết trong các khối này là các chương trình xử lý ngắt.

Các khối này gồm có:

- OB10 (Time of Day Interput): Ngắt thời gian trong ngày, bắt đầu chạy ở thời điểm (được lập trình nhất định) đặc biệt.
- OB20 (Time Delay Interput): Ngắt trì hoãn, chương trình trong khối này được thực hiện sau một khoảng thời gian delay cố định.
- OB35 (Cyclic Interput): Ngắt tuần hoàn, lặp lại sau khoảng thời gian cách đều nhau được định trước (1ms đến 1 phút).
- OB40 (Hardware Interput): Ngắt cứng, chạy khi phát hiện có lỗi trong module ngoại vi.
- OB80 (Cycle Time Fault): Lỗi thời gian chu trình, thực hiện khi thời gian vòng quét vượt quá thời gian cực đại đã định,
- OB81 (Power Supply Fault): Thực hiện khi CPU phát hiện thấy có lỗi nguồn nuôi.
- OB82 (Diagnostic Interput): Chương trình trong khối này được gọi khi CPU phát hiện có sự cố từ module I/O mở rộng.
- OB85 (Not Load Fault): Được gọi khi CPU thấy chương trình ứng dụng có sử dụng chế độ ngắt nhưng chương trình xử lý tín hiệu ngắt lại không có trong khối OB tương ứng.
- OB87 (Communication Fault): Thực hiện khi có lỗi truyền thông.
- OB100 (Start Up Information): Thực hiện một lần khi CPU chuyển trạng thái từ STOP sang RUN.

- OB101 (Cold Start Up Information - chỉ có ở CPU S7-400): Thực hiện một lần khi công tắc nguồn của CPU chuyển trạng thái từ OFF sang ON.
- OB121 (Synchronous Error): Được gọi khi có lỗi logic trong chương trình.
- OB122 (Synchronous Error): Được gọi khi có lỗi module trong chương trình.

1.2.7. Ngôn ngữ lập trình của PLC S7-300

Các loại PLC nói chung có nhiều loại ngôn ngữ lập trình nhằm phục vụ các đối tượng sử dụng khác nhau. PLC S7-300 có 3 ngôn ngữ lập trình cơ bản đólà:

- Ngôn ngữ STL (Statement List).
- Ngôn ngữ FBD (Function Block Diagram).
- Ngôn ngữ LAD (Ladder Diagram).

Ngôn ngữ STL (Statement List): Ngôn ngữ "liệt kê lệnh" dạng ngôn ngữ lập trình thông thường của máy tính, một chương trình được ghép bởi nhiều câu lệnh theo một thuật toán nhất định, mỗi lệnh chiếm một hàng và có cấu trúcchung "tên lệnh + toán hạng".

Ngôn ngữ FBD (Function Block Diagram): Ngôn ngữ "hình khối" là ngôn ngữ đồ họa cho những người quen thiết kế mạch điều khiển số.

Ngôn ngữ LAD (Ladder Diagram): Đây là ngôn ngữ lập trình "hình thang", dạng ngôn ngữ đồ họa thích hợp cho những người quen thiết kế mạch điều khiển logic.

Nhưng có một điểm cần lưu ý đó là một chương trình viết trên ngôn ngữ STL thì có thể được chuyển thành dạng ngôn ngữ LAD, FBD nhưng ngược lạithì chưa chắc vì trong tập lệnh của STL thì trong 2 ngôn ngữ trên chưa hẳn đã

có. Vì ngôn ngữ STL là ngôn ngữ có tính đa dạng nhất sau đây xin giới thiệu chitiết hơn về các lệnh trong ngôn ngữ này.

- Các lệnh cơ bản trong STL

Các lệnh về logic tiếp điểm, bao gồm:

=	Lệnh gán.
A	Lệnh thực hiện phép AND.
AN	Lệnh thực hiện phép AND NOT.O
	Lệnh thực hiện phép OR.
ON	Lệnh thực hiện phép OR NOT.
A(	Lệnh thực hiện phép AND với biểu thức.
AN(	Lệnh thực hiện phép AND NOT với biểu thức.O(
	Lệnh thực hiện phép OR với biểu thức.
OR(	Lệnh thực hiện phép OR NOT với biểu thức.X
	Lệnh thực hiện phép EXCLUSIVE OR.
XN	Lệnh thực hiện phép EXCLUSIVE OR NOT.
X(	Lệnh thực hiện phép EXCLUSIVE OR với biểu thức.
XN(	Lệnh thực hiện phép EXCLUSIVE OR NOT với biểu thức.SET
	Lệnh thực hiện phép ghi giá trị 1 vào RLO.
CLR	Lệnh thực hiện phép ghi giá trị 0 vào RLO.NOT
	Lệnh đảo giá trị của RLO.
S	Lệnh ghi giá trị 1 vào toán hạng khi mà trước đó RLO=1. R
	Lệnh ghi giá trị 0 vào toán hạng khi mà trước đó RLO=1. FP
	Lệnh phát hiện sườn lên.
FN	Lệnh phát hiện sườn xuống.
SAVE	Lệnh chuyển nội dung của RLO với bit trạng thái BR.

Các lệnh về thanh ghi ACCU. Có 2 thanh ghi được ký hiệu là ACCU1 và ACCU2. Hai thanh ghi này cùng có kích thước 32 bit, mọi phép tính toántrên số thực, số nguyên, các phép tính logic với mảng nhiều bit,… Đều được thực hiện trên 2 thanh ghi trạng thái này. Các tập lệnh trong 2 thanh ghi này có nhiều lệnh khác nhau gồm những lệnh như:

- Các lệnh đọc ghi và chuyển nội dung thanh ghi ACCU:

L Lệnh đọc giá trị chỉ định trong toán hạng vào thanh ghi ACCU1 và giá trị cũ của ACCU1 sẽ được chuyển tới thanh ghi ACCU2.

T Lệnh cất nội dung ACCU1 vào ô nhớ.

POP Lệnh chuyển nội dung của ACCU2 vào ACCU1. PUSP Lệnh chuyển nội dung của ACCU1 vào ACCU2. TAK Lệnh đảo nội dung của ACCU2 vào ACCU1.

CAW Lệnh đảo nội dùng 2 byte của từ thấp trong ACCU1.CAD Lệnh đảo nội dung các byte trong ACCU1.

INVI Lệnh đảo giá trị các bit trong từ thấp ACCU1.

INVD Lệnh đảo giá trị các bit trong ACCU1.

- Các lệnh logic thực hiện trên thanh ghi ACCU:

AW Lệnh thực hiện phép tính AND giữa các bit trong từ thấpcủa 2 thanh ghi ACCU1 và ACCU2 với nhau.

AD Lệnh thực hiện phép tính AND giữa các bit trong 2 thanhghi ACCU1 và ACCU2 với nhau.

OW Lệnh thực hiện phép tính OR giữa các bit trong từ thấp của2 thanh ghi ACCU1 và ACCU2 với nhau.

OD Lệnh thực hiện phép tính OR giữa các bit trong 2 thanh ghi ACCU1 và ACCU2 với nhau.

XOW Lệnh thực hiện phép tính XOR giữa các bit trong từ thấp

của 2 thanh ghi ACCU1 và ACCU2 với nhau.

XOD Lệnh thực hiện phép tính XOR giữa các bit trong 2 thanhghi ACCU1 và ACCU2 với nhau.

- Các lệnh tăng giảm nội dung thanh ghi ACCU:

INC Lệnh tăng giá trị của byte thấp của từ thấp thanh ghi ACCU1lên 1 đơn vị.

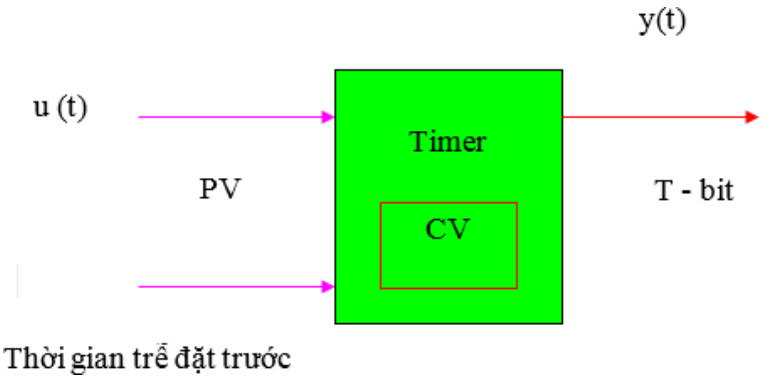
DEC Lệnh giảm giá trị của byte thấp của từ thấp thanh ghi ACCU1xuống 1 đơn vị.

1.2.8. Bộ thời gian (Timer)

1.2.8.1. Nguyên tắc làm việc của bộ thời gian

Bộ thời gian (Timer) hay còn gọi là bộ tạo thời gian trễ theo mong muốn khi có tín hiệu đầu vào cấp cho bộ Timer. Tín hiệu này được tính từ khi có sườnlên ở tính hiệu đầu vào u(t) chuyển từ trạng thái 0 lên 1, được gọi là thời gian điểm kích Timer.

Hình 1.19. Miêu tả tín hiệu vào ra của bộ thời gian



phần:

Thời gian trễ được khai báo với Timer bằng một giá trị 16 bit gồm 2 thành

Độ phân giải với đơn vị là ms. Timer S7-300 có 4 loại độ phân giải khác nhau là 10ms, 100ms, 1s vào 10s.

Một số nguyên (BCD) trong khoảng 0 đến 999, gọi là PV (giá trị đặt trước cho Timer).

Vật thời gian trễ = Độ phân giải * PV.

Ngay tại thời điểm kích Timer giá trị PV (giá trị đặt) được chuyển vào thanh ghi 16 bit của Timer T-Work (Gọi là thanh ghi CV thanh ghi biểu diễn giátrị tức thời). Timer sẽ ghi nhớ khoảng thời gian trôi qua kể từ khi được kích bằng cách giảm dần một cách tương ứng nội dung thanh ghi CV. Nếu nội dung thanh ghi CV trở về không thì Timer đã đạt được thời gian trễ mong muốn và điều nàysẽ được thông báo ra bên ngoài bằng cách thay đổi trạng thái tín hiệu đầu ra y(t).

Nhưng việc thông báo ra bên ngoài cũng còn phụ thuộc vào từng loạiTimer khác nhau.

Bên cạnh sườn lên của tín hiệu đầu vào u(t). Timer còn có thể được kíchbởi sườn lên của tín hiệu chủ động kích có tên là tín hiệu enable.

Và nếu như tại thời điểm có sườn lên của tín hiệu enable, tín hiệu u(t) cógiá trị bằng 1.

Từng loại Timer được đánh số thứ tự từ 0 đến 255 tùy thuộc vào từng loại CPU. Một Timer đang làm việc có thể được đưa về trạng thái chờ khởi động ban đầu nhờ tín hiệu Reset, khi có tín hiếu xóa thì Timer cũng ngừng làm việc luôn. Đồng nghĩa với các giá trị của T-Word và T-Bit cũng đồng thời được xóa về 0 lúc đó giá trị tức thời CV và tín hiệu đầu ra cũng là 0 luôn.

1.2.8.2. Khai báo sử dụng

Việc khai báo làm việc của bộ Timer bao gồm các bước sau:

- Khai báo tín hiệu enable nếu muốn sử dụng tín hiệu chủ động kích.
- Khai báo tín hiệu đầu vào u(t).
- Khai báo thời gian trễ mong muốn.
- Khai báo loại Timer được sử dụng (SD, SS, SP, SE, SF).

- Khai báo tín hiệu xóa Timer nếu muốn sử dụng chế độ Reset chủ động.

Trong các khai báo trên thì các bước 2, 3, 4 là bắt buộc phải có. S7-300 có 5 loại Timer được khai báo bằng các lệnh:

- Timer SD (On Delay Timer): Trễ theo sườn lên không nhớ.
- Timer SS (Retentive On Delay Timer): Trễ theo sườn lên có nhớ.
- Timer SP (Pulse Timer): Timer tạo xung không có nhớ.
- Timer SE (Extended Pulse Timer): Timer tạo xung có nhớ.
- Timer SF (Off Delay): Timer trễ theo sường xuống.

1.2.9. Bộ đếm (Counter)

1.2.9.1. Nguyên tắc làm việc của bộ đếm

Counter là bộ đếm thực hiện chức năng đếm sường xung của các tín hiệu đầu vào. S7-300 có tối đa 256 Counter, ký hiệu Cx trong đó x là số nguyên trong khoảng từ 0 đến 255. Những bộ đếm của S7-300 đều có thể đồng thời đếm tiến theo sườn lên của một tín hiệu vào thứ nhất, ký hiệu là CU (Count Up) và đếmlùi theo sườn lên của một tín hiệu vào thứ hai, ký hiệu là CD (Count Down). Bộ đếm còn có thể được đếm bàng tín hiệu chủ động kích enable khi mà tín hiệu chủ động kích có tín hiệu đồng thời tín hiệu vào CU hoặc CD thì bộ đếm sẽ thực hiện tín hiệu đếm tương ứng.

Số sườn xung đếm được ghi vào thanh ghi 2 byte của bộ đếm, gọi là thanh ghi C-Word. Nội dung của C-Word được gọi là giá trị đếm tức thời của bộ đếmvà ký hiệu bằng CV (Current Value). Bộ đếm báo trạng thái của C-Word rangoài qua chân C-Bit của nó. Nếu CV#0 thì C-Bit có giá trị bằng 1. Ngược lại khi CV=0 thì C-Bit có giá trị bằng 0. CV luôn là giá trị không âm bộ đếm sẽ không đếm lùi khi mà giá trị CV=0.

Khác với Timer giá trị đặt trước PV (Preset Value) của bộ đém chỉ được chuyển vào C-Word tại thời điểm xuất hiện sườn lên của tín hiệu đặt (Set-S).

Bộ đếm có thể được xóa chủ động bằng tín hiệu xóa (Reset-R). Khi bộđếm được xóa cả C-Word và C-Bit đều nhận giá trị 0.

1.2.9.2. Khai báo sử dụng

Bộ đếm trong S7-300 có 2 loại đó là đếm tiến (CU) và đếm lùi (CD) cácbước khai báo sử dụng một bộ đếm Counter bao gồm các bước sau:

- Khai báo tín hiệu enable nếu muốn sử dụng tín hiệu chủ động kích hoạt.
- Khai báo tín hiệu đầu vào CU được sử dụng để đếm tiến.
- Khai báo tín hiệu đầu vào CD được sử dụng để đếm lùi.
- Khai báo tín hiệu (Set) và giá trị đặt trước (PV).
- Khai báo tín hiệu xóa (Reset).

Trong đó ít nhất bước 2 hoặc bước 4 phải được thực hiện.Ngoài ra còn có lệnh về đọc nội dung thanh ghi C-Word.

L <Tên Counter> // Đọc giá trị đếm tức thời dạng nhị phân vào thanh ghiACCU1.

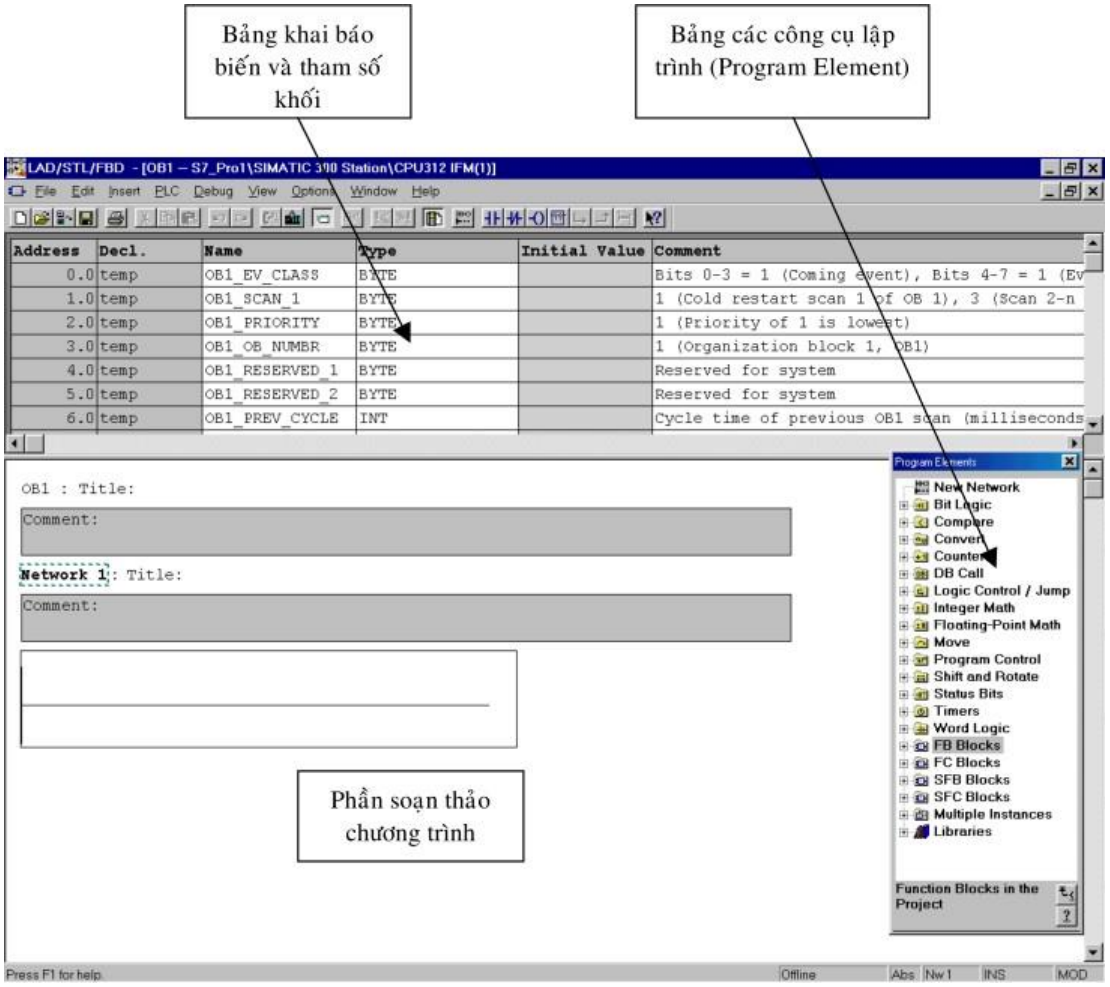
LC <Tên Counter> // Đọc giá trị đếm tức thời dạng BCD vào thanh ghiACCU1.

1.3. PHẦN MỀM LẬP TRÌNH

1.3.1. Khai báo phần cứng

Ta phải xây dựng cấu hình phần cứng khi tạo một project. Dữ liệu về cấuhình sẽ được truyền đến PLC sau đó.

1.3.2. Cấu trúc cửa sổ lập trình



Hình 1.20. Cấu trúc cửa sổ lập trình

Bảng khai báo phụ thuộc khối. Dùng để khai báo biến và tham số khối.

Phần soạn thảo chứa một chương trình, nó chia thành từng Network. Cácthông số nhập được kiểm tra lỗi cú pháp.

Nội dung cửa sổ "Program Element" tuỳ thuộc ngôn ngữ lập trình đã lựachọn. Có thể nhấn đúp vào phần tử lập trình cần thiết trong danh sách để chèn

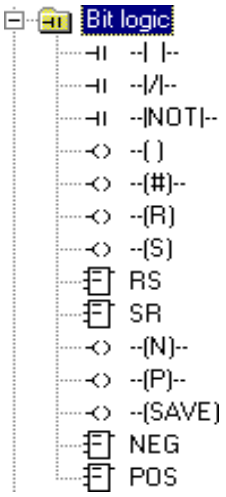
chúng vào danh sách. Cũng có thể chèn các phần tử cần thiết bằng cách nhấn vànhả chuột.

Các thanh công cụ thường sử dụng:

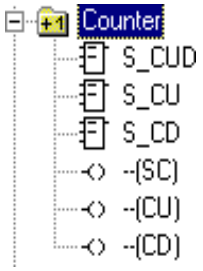
- New (File Menu): Tạo mới.
- Open (File Menu): Mở file.
- Cut (Edit Menu): Cắt.
- Paste (Edit Menu): Dán.
- Copy (Edit Menu): Sao chép.
- Download (PLC Menu): Tải xuống.
- Network (Insert): Chèn network mới.
- Program Elements (Insert): Mở cửa sổ các phần tử lập trình.
- Clear/Reset (PLC): Xóa chương trình hiện thời trong PLC.
- LAD, STL, FBD (View): Hiển thị dạng ngôn ngữ yêu cầu.

Các phần tử lập trình thường dùng (cửa sổ Program Elements):

- Các lệnh logic tiếp điểm:

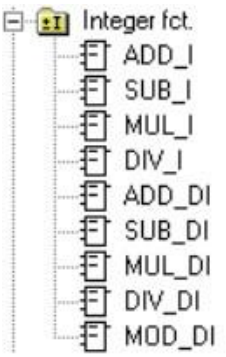


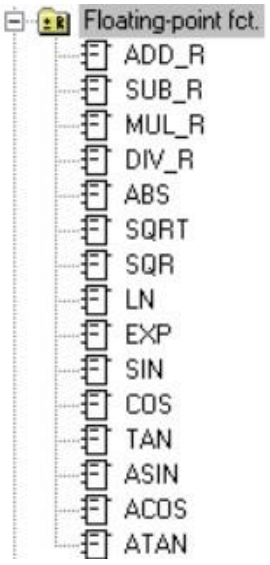
- Các loại counter:



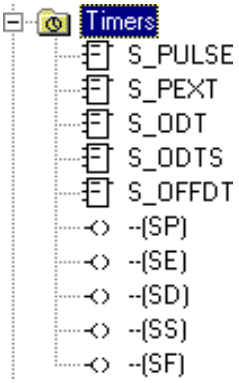
- Các lệnh toán học:

Số nguyên: Số thực:

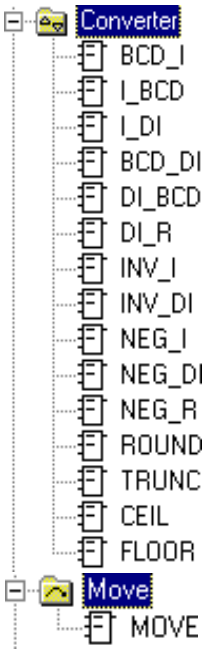




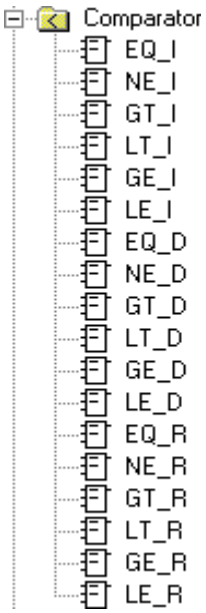
- Các loại Timer:



- Các lệnh chuyển đổi dữ liệu:



- Các lệnh so sánh:



CHƯƠNG 7: GIỚI THIỆU PLC S7-1200

1. Giới thiệu chung về PLC S7-1200

1.1. Khái niệm chung PLC s7-1200

Năm 2009, Siemens ra dòng sản phẩm S7-1200 dùng để thay thế dần cho S7-200. So với S7-200 thì S7-1200 có những tính năng nổi trội:

-S7-1200 là một dòng của bộ điều khiển logic lập trình (PLC) có thể kiểm soát nhiều ứng dụng tự động hóa. Thiết kế nhỏ gọn, chi phí thấp, và một tập lệnh mạnh làm cho chúng ta có những giải pháp hoàn hảo hơn cho ứng dụng sử dụng với S7-1200

-S7-1200 bao gồm một microprocessor, một nguồn cung cấp được tích hợp sẵn, các đầu vào/ra (DI/DO).

-Một số tính năng bảo mật giúp bảo vệ quyền truy cập vào cả CPU và chương trình điều khiển:

+Tất cả các CPU đều cung cấp bảo vệ bằng password chống truy cập vào PLC

+Tính năng "know-how protection" để bảo vệ các block đặc biệt của mình

-S7-1200 cung cấp một cổng PROFINET, hỗ trợ chuẩn Ethernet và TCP/IP. Ngoài ra bạn có thể dùng các module truyền thong mở rộng kết nối bằng RS485 hoặc RS232.

-Phần mềm dùng để lập trình cho S7-1200 là Step7 Basic. Step7 Basic hỗ trợ ba ngôn ngữ lập trình là FBD, LAD và SCL. Phần mềm này được tích hợp trong TIA Portal 11 của Siemens.

-Vậy để làm một dự án với S7-1200 chỉ cần cài TIA Portal vì phần mềm này đã bao gồm cả môi trường lập trình cho PLC và thiết kế giao diện HMI

1.2. Các module trong hệ PLC S7-1200

1.2.1. Giới thiệu về các module CPU

Các module CPU khác nhau có hình dạng, chức năng, tốc độ xử lý lệnh, bộ nhớ chương trình khác nhau….

PLC S7-1200 có các loại sau:

	SIMATIC S7-1200		Product Description
CPU	CPU 1211C	1211 CPU AC/DC/Rly	Compact CPU 1211C, 25 KB integral PROGRAM / DATA MEMORY , 1MB loading memory; execution times for boolean operations: 0.1 µs; integral I/Os: 6 digital inputs, 4 digital outputs, 2 analog inputs; expandable with up to 3 communication modules and 1 signal board; digital inputs as HSC with 100kHz, 24 DC digital outputs can be used as PTO or PWM with 100kHz
		1211 CPU DC/DC/DC	
		1211 CPU DC/DC/Rly	
	CPU 1212C	1212-CPU AC/DC/Rly	Compact CPU 1212C, 25 KB integral PROGRAM / DATA MEMORY, 1MB loading memory; execution times for boolean operations: 0.1 µs; integral I/Os: 8 digital inputs, 6 digital outputs, 2 analog inputs; expandable with up to 3 communication modules, 2 signal modules and 1 signal board; digital inputs can be used as HSC with 100kHz and 24 DC digital outputs as PTO or PWM with 100kHz
		1212-CPU DC/DC/DC	
		1212-CPU DC/DC/Rly	
	CPU 1214C	1214 CPU AC/DC/Rly	Compact CPU 1214C, 50 KB integral PROGRAM / DATA MEMORY, 2MB loading memory;
		1214 CPU DC/DC/DC	operations: 0.1 µs; integral I/Os: 14 digital inputs, 10 digital outputs, 2 analog inputs; expandable with up to 3 communication modules, 8 signal modules and 1 signal board; digital inputs can be used as HSC with 100kHz and 24 DC Digital outputs as PTO or PWM with 100kHz
		1214 CPU DC/DC/Rly	

1.2.2. Sign board của PLC SIMATIC S7-1200

Sign board: SB1223 DC/DC

-Digital inputs / outputs

-DI 2 x 24 VDC 0.5A

-DO 2x24 VDC 0.5A

Sign boards : SB1232AQ

- Ngõ ra analog

-AO 1 x 12bit

-+/- 10VDC, 0 – 20mA

Signal Boards Digital / Analog	SB 1223	2 x 24VDC inputs / 2 x 24VDC outputs	2 inputs, DC 24V, IEC type 1, current sinking; 2 transistor outputs DC 24V, 0.5 A, 5 W; can be used as additional HSC with up to 30 kHz
	SB 1232	1 analog outputs	1 analog output, ±10 V with 12 bits or 0 to 20 mA with 11 bits

Cards ứng dụng:

-CPU tín hiệu để thích ứng với các ứng dụng

-Thêm điểm của kỹ thuật số I/O hoặc tương tự với CPU như các yêu cầu ứng dụng

-Kích thước của CPU sẽ không thay đổi

SIMATIC MEMORY CARD	2 MB	MEMORY CARD for S7-1200 CPU, 2 MB
SIMATIC MEMORY CARD	24 MB	MEMORY CARD for S7-1200 CPU, 24 MB

1.2.3. Module xuất nhập tín hiệu số

Signal Modules Digital	SM 1222	8 x relay outputss	8 relay outputs, DC 5 to 30V / AC 5 to 250V, 2 A, 30 W DC / 200 W AC
	SM 1222	8 x 24V DC outputs	8 transistor outputs, DC 24V, 0.5 A, 5 W
	SM 1223	8 x 24V DC inputs / 8 x relay outputs	8 inputs, DC 24V, IEC type 1, current sinking; 8 relay outputs, DC 5 to 30V / AC 5 to 250V, 2 A, 30 W DC / 200 W AC
	SM 1223	8 x 24V DC inputs / 8 x 24V DC outputs	8 inputs, DC 24V, IEC type 1, current sinking; 8 transistor outputs, DC 24V, 0.5 A, 5 W
	SM 1221	8 x 24V DC inputs	8 inputs, DC 24V, IEC type 1, current sinking
	SM 1222	16 x relay outputs	16 relay outputs, DC 5 to 30V / AC 5 to 250V, 2 A, 30 W DC / 200 W AC
	SM 1222	16 x 24V DC outputs	16 transistor outputs, DC 24V, 0.5 A, 5 W
	SM 1223	16 x 24V DC inputs / 16 x relay output	16 inputs, DC 24V, IEC type 1, current sinking; 16 relay outputs, DC 5 to 30V / AC 5 to 250V, 2 A, 30 W DC / 200 W AC
	SM 1223	16 x 24V DC inputs / 16 x 24VDC outputs	16 inputs, DC 24V, IEC type 1, current sinking; 16 Transistor outputs, DC 24V, 0,5 A, 5 W
	SM 1221	16 x 24V DC	16 inputs, DC 24V, IEC type 1,

1.2.4. Module xuất nhập tín hiệu tương tự

Signal Modules Analog	SM 1234	4 x analog inputs / 2 x analog outputs	4 analog inputs, ±10 V, ±5 V, ±2.5 V, or 0 to 20 mA, 12 bits + sign; 2 analog outputs, ±10 V at 14 bits or 0 to 20 mA at 13 bits
	SM 1231	4 x analog inputs	4 analog inputs ±10 V, ±5 V, ±2.5 V, or 0 to 20 mA 12 bits + sign
	SM 1232	2 x analog outputs	2 analog outputs, ±10 V at 14 bits or 0 to 20 mA at 13 bits

1.2.5. Module truyền thông

Communication Modules	CM 1241	RS 485	RS 485 point-to-point communication module
	CM 1241	RS 232	RS 232 point-to-point communication module

2. Làm việc với phần mềm Tia Portal

2.1. Giới thiệu SIMATIC STEP 7 Basic – tích hợp lập trình PLC và HMI

Step 7 basic hệ thống kỹ thuật đồng bộ đảm bảo hoạt động liên tục hoàn hảo. Một hệ thống kỹ thuật mới

Thông minh và trực quan cấu hình phần cứng kỹ thuật và cấu hình mạng, lập trình, chẩn đoán và nhiều hơn nữa.

Lợi ích với người dùng:

-Trực quan : dễ dàng để tìm hiểu và dễ dàng để hoạt động

-Hiệu quả : tốc độ về kỹ thuật

-Chức năng bảo vệ : Kiến trúc phần mềm tạo thành một cơ sở ổn định cho sự đổi mới trong tương lai.

2.2. Kết nối qua giao thức TCP/IP

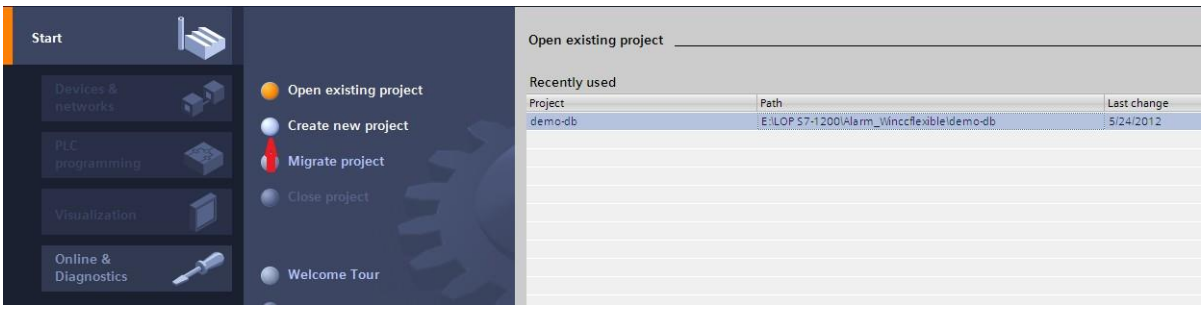
-Để lập trình SIMATIC S7-1200 từ PC hay Laptop cần một kết nối TCP/IP

-Để PC và SIMATIC S7-1200 có thể giao tiếp với nhau, điều quan trọng là các địa chỉ IP của cả hai thiết bị phải phù hợp với nhau

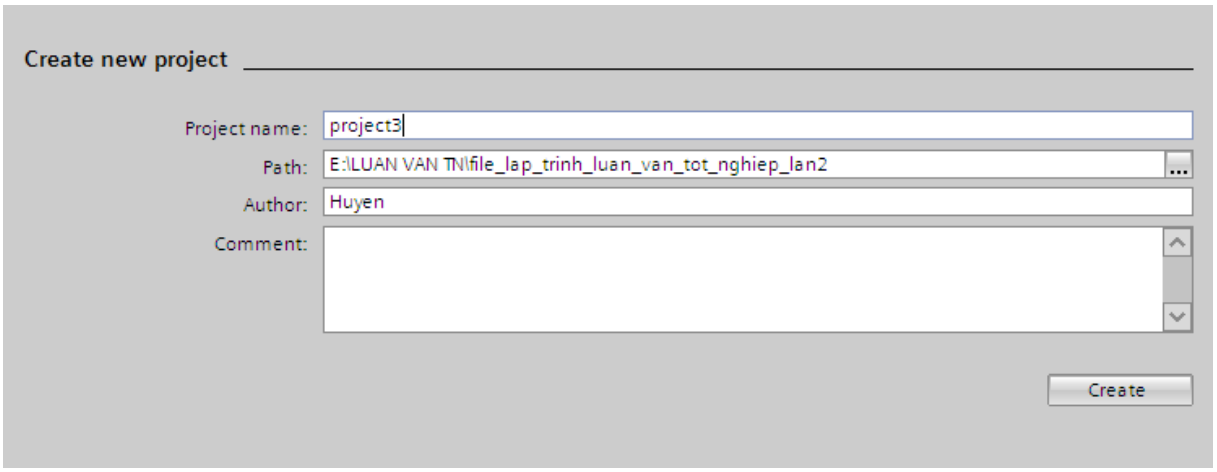
2.3. Cách tạo một Project

Bước 1: từ màn hình desktop nhấp đúp chọn biểu tượng Tia Portal V11

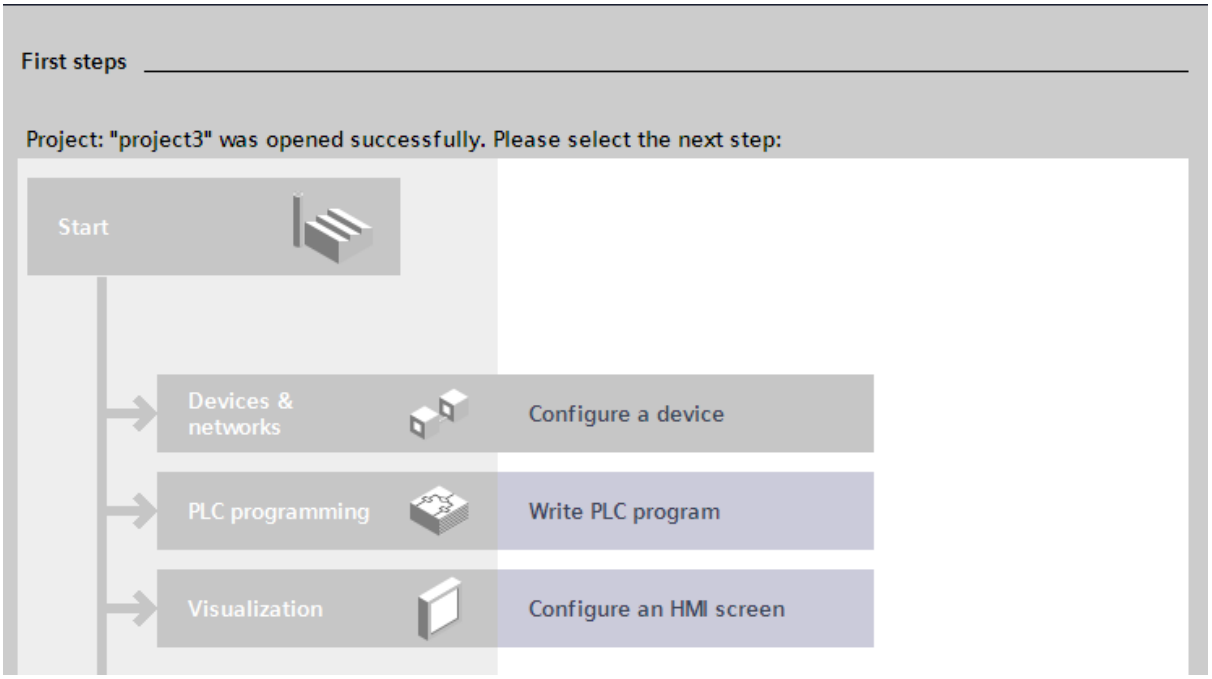
Bước 2 : Click chuột vào Create new project để tạo dự án.



Bước 3 : Nhập tên dự án vào Project name sau đó nhấn create

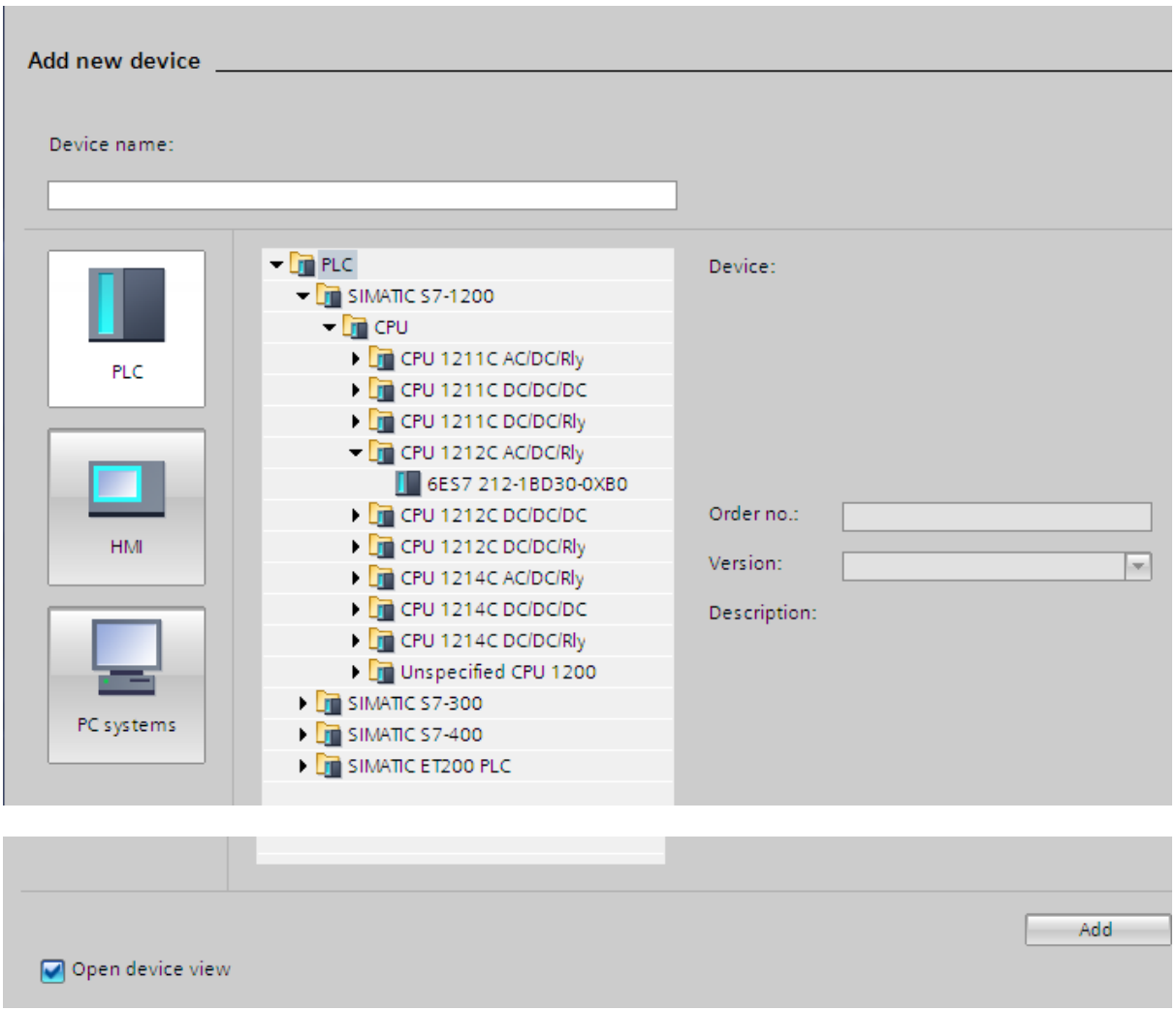


Bước 4 : Chọn configure a device

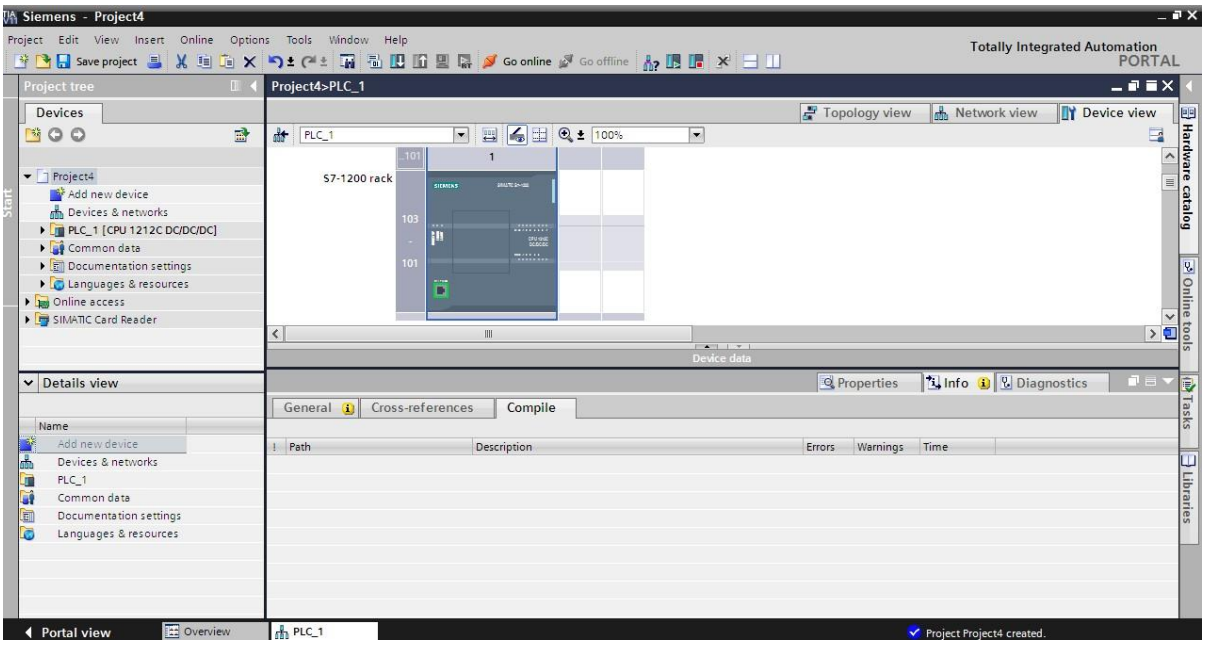


Bước 5 : Chọn add new device

Bước 6 : Chọn loại CPU PLC sau đó chọn add



Bước 7 : Project mới được hiện ra



2.4. TAG của PLC / TAG local

Tag của PLC

-Phạm vi ứng dụng : giá trị Tag có thể được sử dụng mọi khối chức năng trong PLC

-Ứng dụng : binary I/O, Bits of memory

-Định nghĩa vùng : Bảng tag của PLC

-Miêu tả : Tag PLC được đại diện bằng dấu ngoặc kép Tag

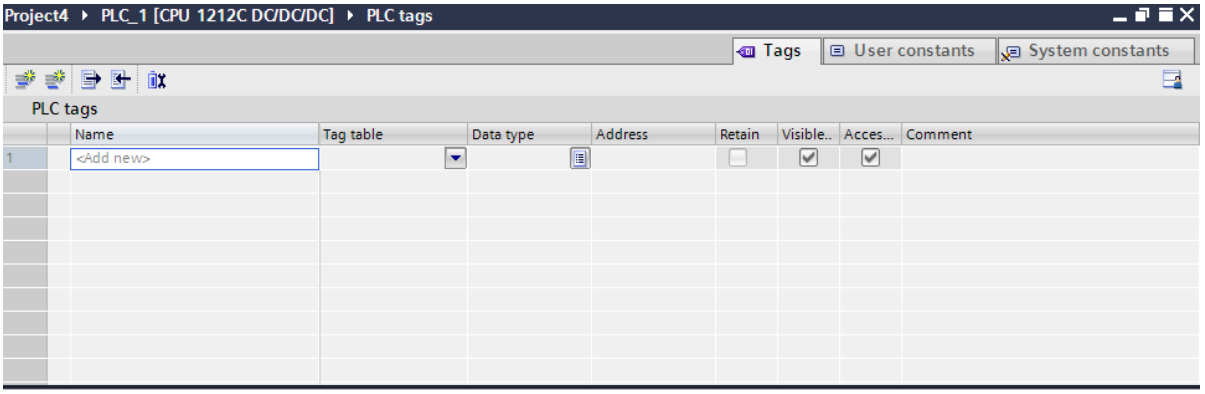
Local

-Phạm vi ứng dụng : giá trị chỉ được ứng dụng trong khối được khai báo, mô tả tương tự có thể được sử dụng trong các khối khác nhau cho các mục đích khác nhau.

-Ứng dụng : tham số của khối, dữ liệu static của khối, dữ liệu tạm thời

-Định nghĩa vùng : khối giao diện

-Miêu tả : Tag được đại diện bằng dấu # Sử dụng Tag trong hoạt động



-Layout : bảng tag PLC chứa các định nghĩa của các Tag và các hằng số có giá trị trong CPU. Một bảng tag của PLC được tự động tạo ra cho mỗi CPU được sử dụng trong project.

-Colum : mô tả biểu tượng có thể nhấp vào để di chuyển vào hệ thống hoặc có thể kéo nhả như một lệnh chương trình

-Name : chỉ được khai báo và sử dụng một lần trên CPU

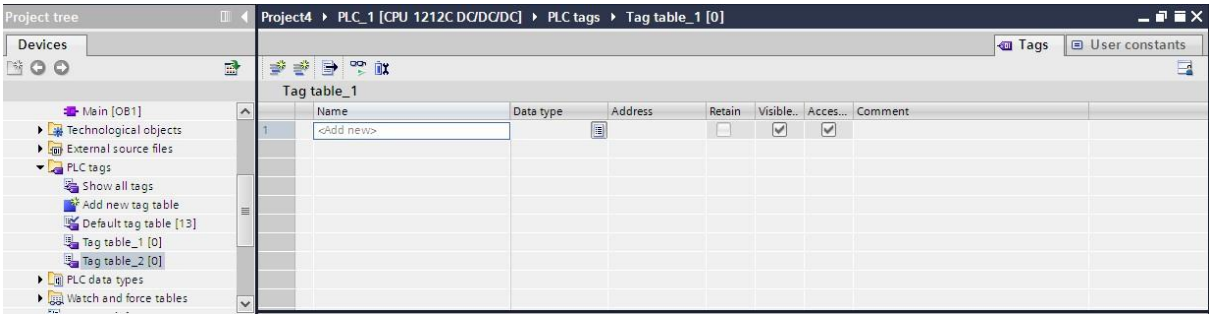
-Data type : kiểu dữ liệu chỉ định cho các tag

-Address : địa chỉ của tag

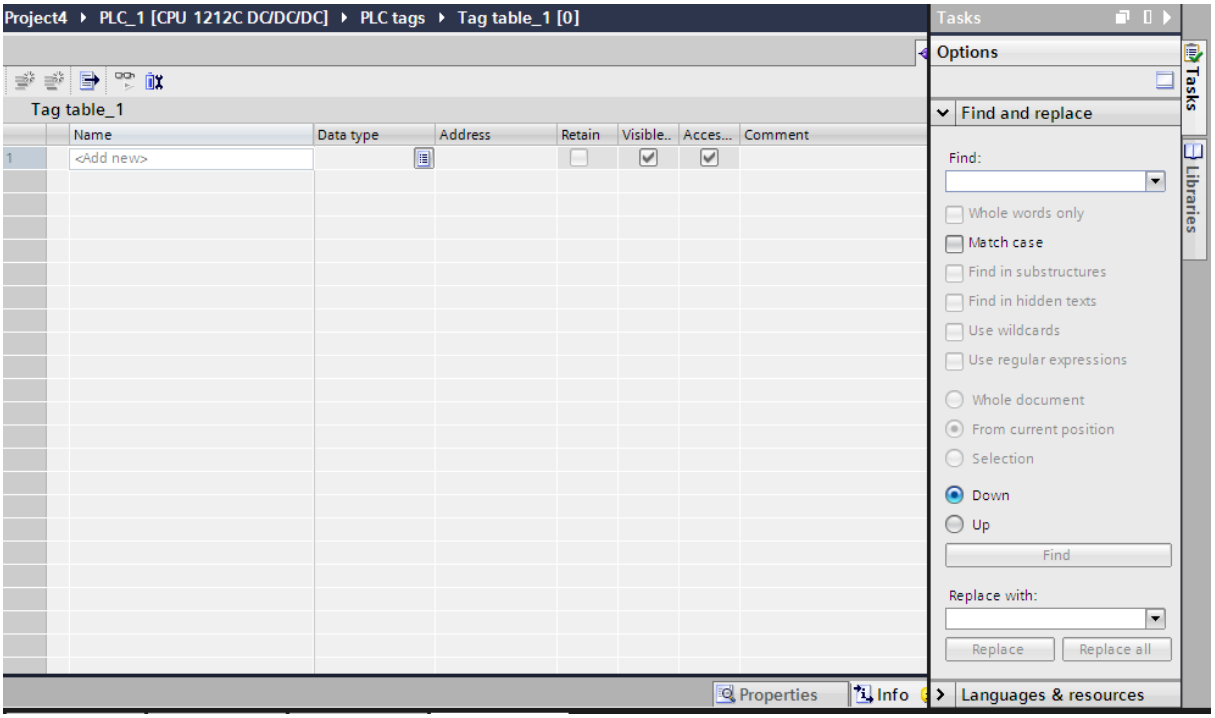
-Retain : khai báo của tag sẽ được lưu trữ lại

-Comment : comment miêu tả của tag

Nhóm tag : tạo nhóm tag bằng cách chọn add new tag table



Tìm và thay thế tag PLC



Ngoài ra còn có một số chức năng sau:

-Lỗi tag

-Giám sát tag của plc

-Hiện / ẩn biểu tượng

-Đổi tên tag : Rename tag

-Đổi tên địa chỉ tag : Rewire tag

-Copy tag từ thư viện Global

3. Làm việc với một trạm PLC

3.1. Quy định địa chỉ IP cho module CPU

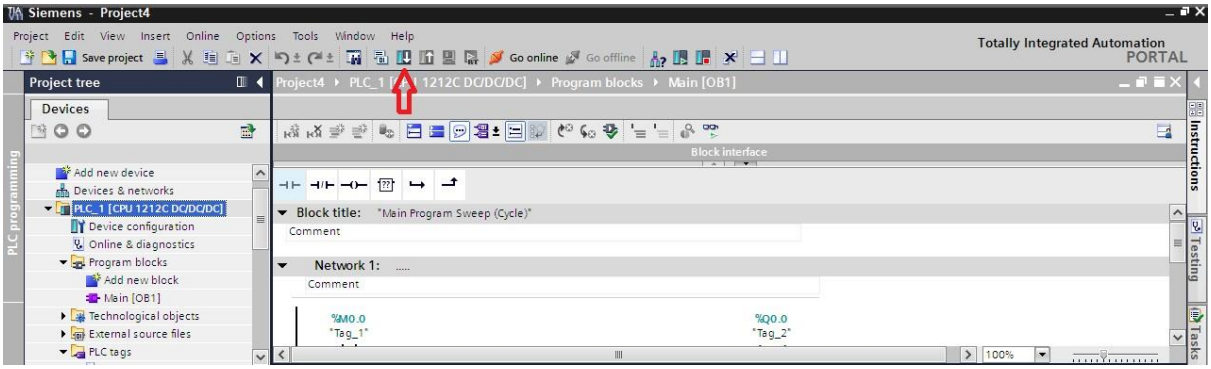
IP TOOL có thể thay đổi IP address của PLC S7-1200 bằng 1 trong 2 cách. Phương pháp thích hợp được tự động xác định bởi trạng thái của địa chỉ IP đó :

-Gán một địa chỉ IP ban đầu : Nếu PLC S7-1200 không có địa chỉ IP, IP TOOL sử dụng các chức năng thiết lập chính để cấp phát một địa chỉ IP ban đầu cho PLC S7- 1200.

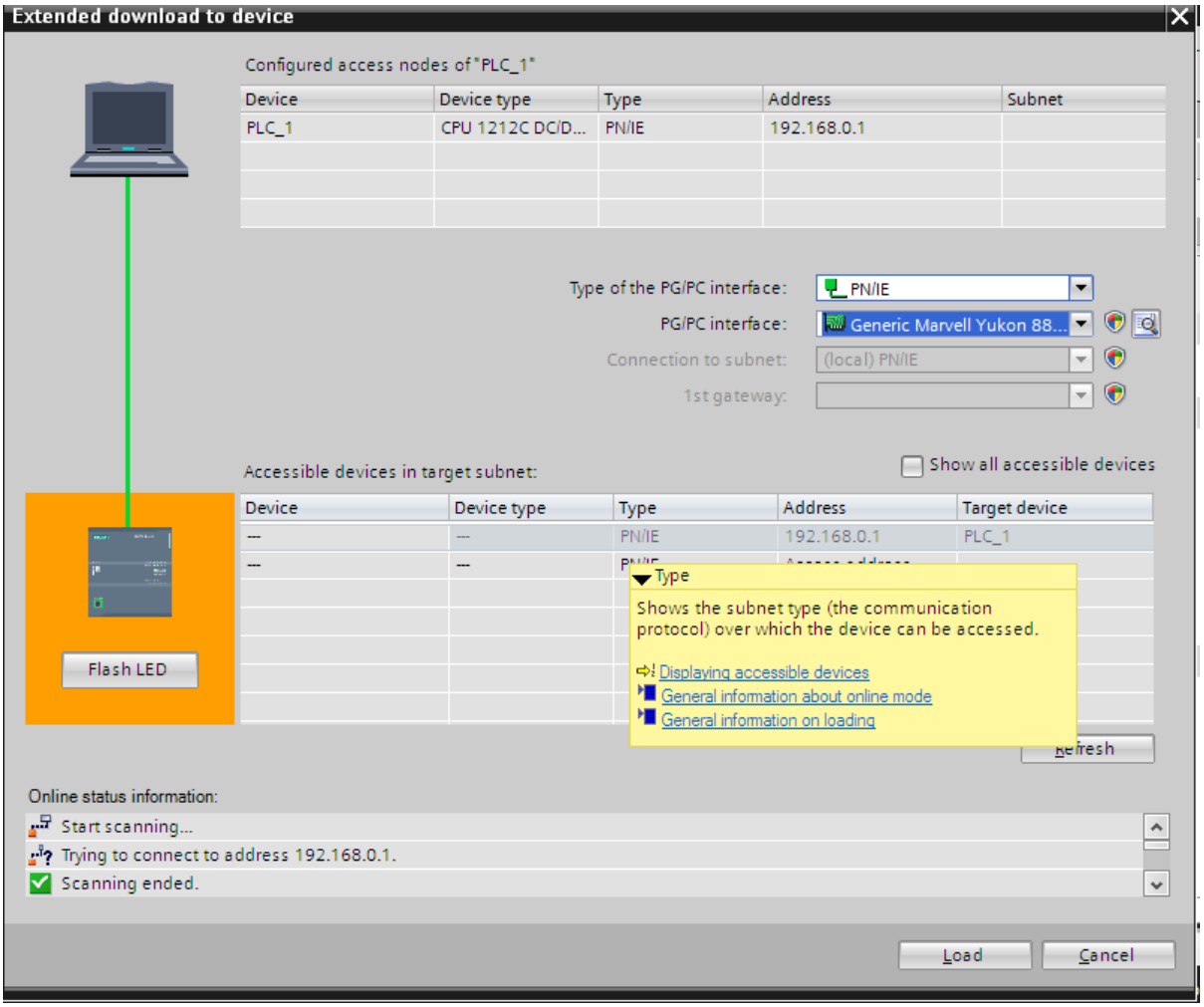
-Thay đổi địa chỉ IP : nếu địa chỉ IP đã tồn tại, công cụ IP TOOL sẽ sửa đổi cấu hình phần cứng (HW config) của PLC S7-1200.

3.2. Đổ chương trình xuống CPU

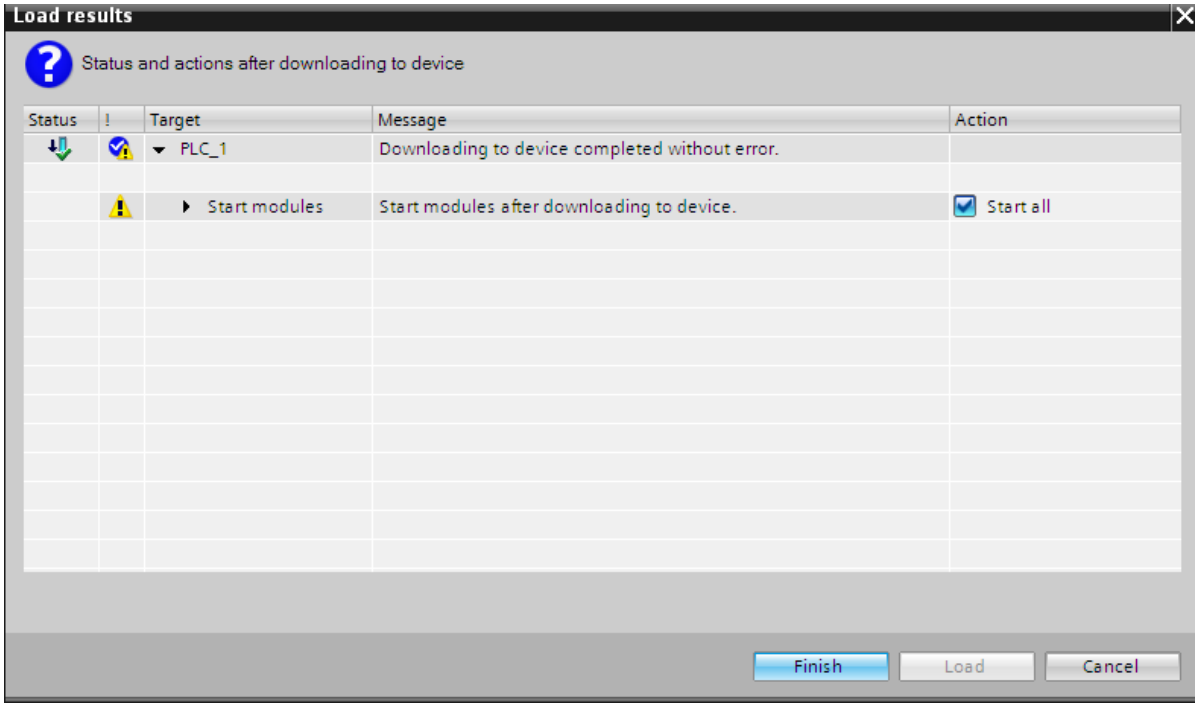
Đổ từ màn hình soạn thảo chương trình bằng cách kích vào biểu tượng download trên thanh công cụ của màn hình



Chọn cấu hình Type of the PG/PC interface và PG/PC interface như hình dưới sau đó nhấn chọn load

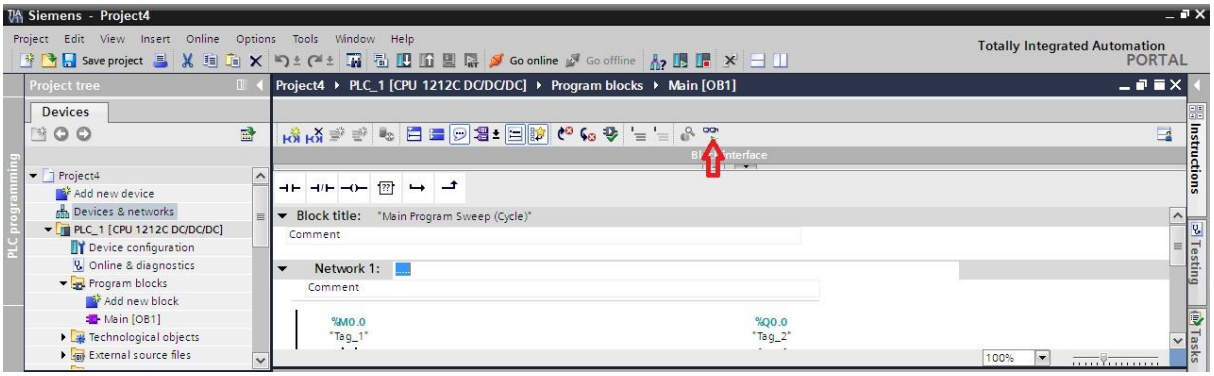


Chọn start all như hình vẽ và nhấn finish

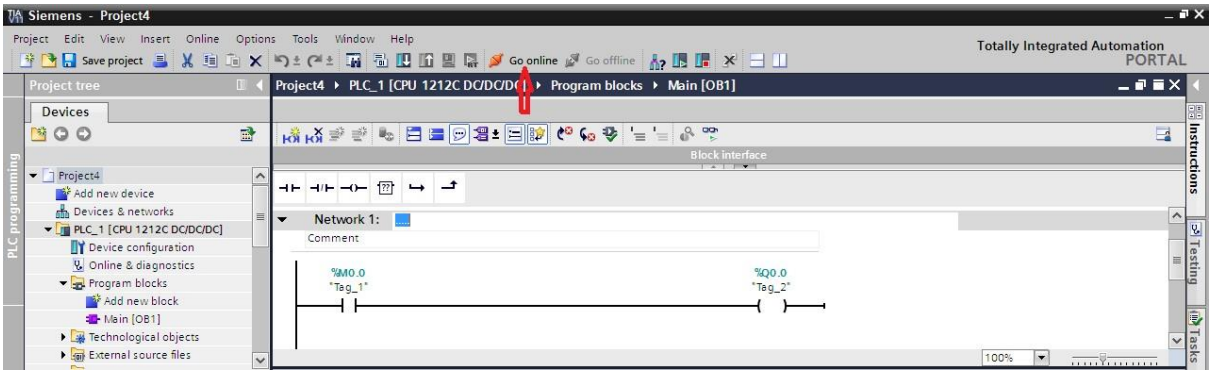


3.3. Giám sát và thực hiện chương trình

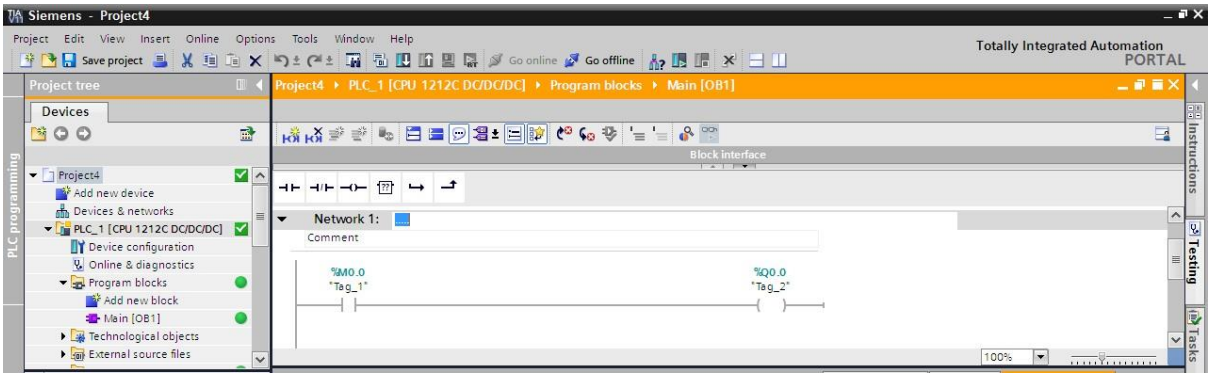
Để giám sát chương trình trên màn hình soạn thảo kích chọn Monitor trên thanh công cụ.



Hoặc cách 2 làm như hình dưới



Sau khi chọn monitor chương trình soạn thảo xuất hiện như sau:



4. Kỹ thuật lập trình

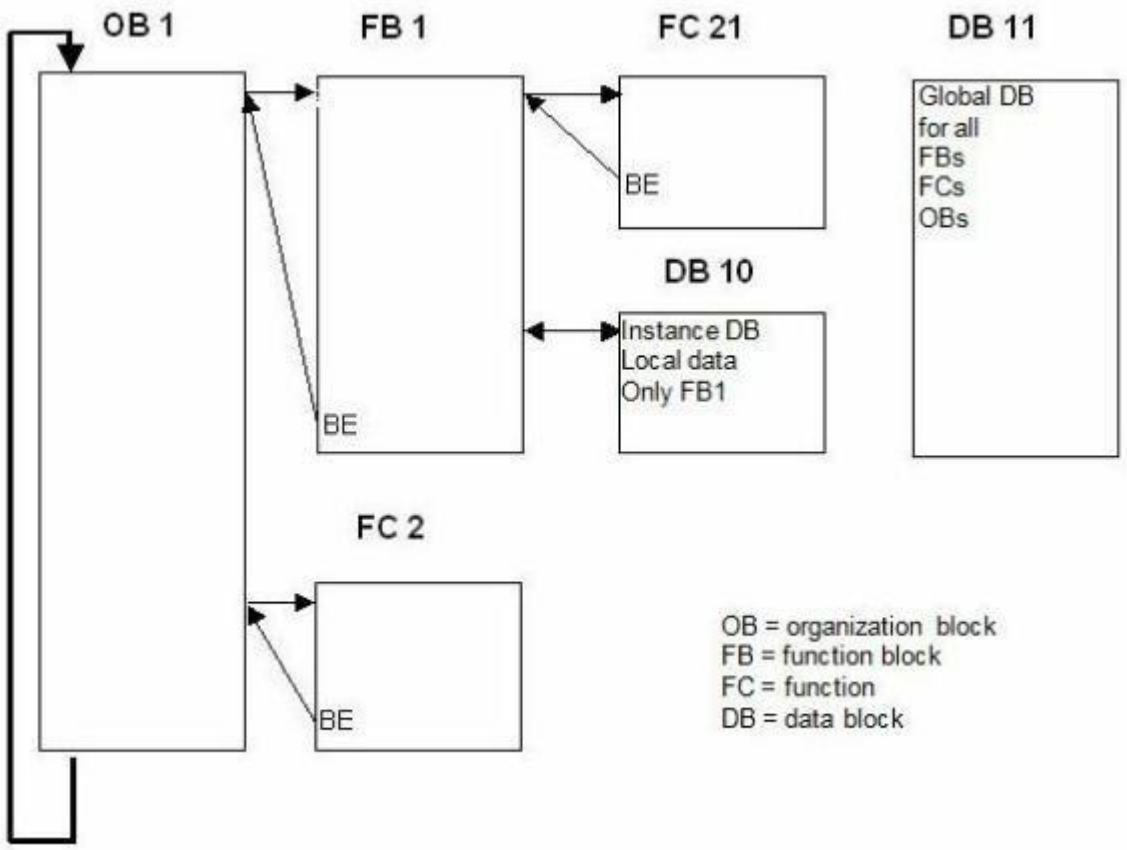
4.1. Vòng quét chương trình

PLC thực hiện chương trình theo chu trình lặp. Mỗi vòng lặp được gọi là vòng quét. Mỗi vòng quét được bắt đầu bằng giai đoạn chuyển dữ liệu từ các cổng vào số tới vùng bộ đệm ảo I, tiếp theo là giai đoạn thực hiện chương trình. Trong từng vòng quét chương trình được thực hiện từ lệnh đầu tiên đến lệnh kết thúc của khối OB1.

Sau giai đoạn thực hiện chương trình là giai đoạn chuyển các nội dụng của bọ đệm ảo Q tới các cổng ra số. Vòng quét kết thúc bằng giai đoạn truyền thông nội bộ và kiểm tra lỗi.

Chú ý rằng bộ đệm I và Q không liên quan tới các cổng vào / ra tương tự nên các lệnh truy nhập cổng tương tự được thực hiện trực tiếp với cổng vật lý chứ không thông qua bộ đệm.

4.2. Cấu trúc lập trình



4.2.1. Khối tổ chức OB – OGANIZATION BLOCKS

-Organization blocks (OBs) : là giao diện giữa hoạt động hệ thống và chương trình người dùng. Chúng được gọi ra bởi hệ thống hoạt động, và điều khiển theo quá trình:

+Xử lý chương trình theo quá trình

+Báo động – kiểm soát xử lý chương trình

+Xử lý lỗi

-Startup oB, Cycle OB, Timing Error OB và Diagnosis OB : có thể chèn và lập trình các khối này trong các project. Không cần phải gán các thông số cho chúng và cũng không cần gọi chúng trong chương trình chính.

-Process Alarm OB và Time Interrupt OB : Các khối OB này phải được tham số hóa khi đưa vào chương trình. Ngoài ra, quá trình báo động OB có thể được gán cho một sự kiện tại thời gian thực hiện bằng cách sủ dụng các lệnh ATTACH, hoặc tách biệt với lệnh DETACH.

-Time Delay Interrupt OB : OB ngắt thời gian trễ có thể được đưa vào dự án và lập trình. Ngoài ra, chúng phải được gọi trong chương trình với lệnh SRT_DINT, tham số là không cần thiết

-Start Information : Khi một số OB được bắt đầu, hệ điều hành đọc ra thông tin được thẩm định trong chương trình người dùng, điều này rất hữu ích cho việc chẩn đoán lỗi, cho dù thông tin được đọc ra được cung cấp trong các mô tả của các khối OB

4.2.2. Hàm chức năng – FUNCTION

-Funtions (FCs) là các khối mã không cần bộ nhớ. Dữ liệu của các biến tạm thời bị mất sau khi FC được xử lý. Các khối dữ liệu toàn cầu có thể được sử dụng để lưu trữ dữ liệu FC.

-Functions có thể được sử dụng với mục đích

+Trả lại giá trị cho hàm chức năng được gọi

+Thực hiện công nghệ chức năng, ví dụ : điều khiển riêng với các hoạt động nhị phân

+Ngoài ra, FC có thể được gọi nhiều lần tại các thời điểm khác nhau trong một chương trình. Điều này tạo điều kiện cho lập trình chức năng lập đi lặp lại phức tạp.

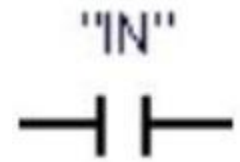
-FB (function block) : đối với mỗi lần gọi, FB cần một khu vực nhớ. Khi một FB được gọi, một Data Block (DB) được gán với instance DB. Dữ liệu trong Instance DB sau đó truy cập vào các biến của FB. Các khu vực bộ nhớ khác nhau đã được gán cho một FB nếu nó được gọi ra nhiều lần.

-DB (data block) : DB thường để cung cấp bộ nhớ cho các biến dữ liệu . Có hai loại của khối dữ liệu DB : Global DBs nơi mà tất cả các OB, FB và FC có thể đọc được dữ liệu lưu trữ, hoặc có thể tự mình ghi dữ liệu vào DB, và instance DB được gán cho một FB nhất định.

5. Giới thiệu các tập lệnh

5.1. Bit logic (tập lệnh tiếp điểm)

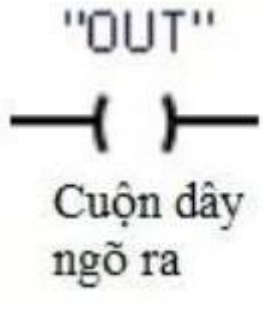
1)tiếp điểm thường hở

L A D		Tiếp điểm thường hở sẽ đóng khi giá trị của bit có địa chỉ là n bằng 1 Toán hạng n: I, Q, M, L, D

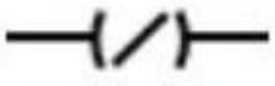
2)tiếp điểm thường đóng

L A D		Tiếp điểm thường đóng sẽ đóng khi giá trị của bit có địa chỉ n là 0 Toán hạng n: I, Q, M, L, D

3)lệnh OUT

L A D		Giá trị của bit có địa chỉ là n sẽ bằng 1 khi đầu vào của lệnh này bằng 1 và ngược lại Toán hạng n : Q, M, L, D Chỉ sử dụng một lệnh out cho 1 địa chỉ

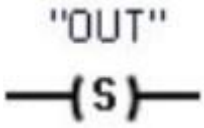
4)Lệnh OUT đảo

L A D		Giá trị của bit có địa chỉ là n sẽ bằng 1 khi đầu vào của lệnh này bằng 0 và ngược lại Toán hạng n : Q, M, L, D Chỉ sử dụng một lệnh out not cho 1 địa chỉ

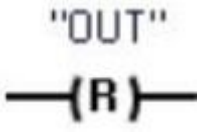
5)Lệnh logic NOT

L A D		Lệnh đảo trạng thái ngõ vào / ra

6) Lệnh SET

L A D		Giá trị của các bit có địa chỉ là n sẽ bằng 1 khi đầu vào của lệnh này bằng 1 Khi đầu vào của lệnh bằng 0 thì bit này vẫn giữ nguyên trạng thái. Toán hạng n: Q, M, L, D

7) lệnh Reset

L A D		Giá trị của các bit có địa chỉ là n sẽ bằng 0 khi đầu vào của lệnh này bằng 1. Khi đầu vào của lệnh bằng 0 thì các bit này vẫn giữ nguyên trạng thái. Toán hạng n: Q, M, L, D

8) Lệnh set nhiều bit

L A D		Giá trị của các bit có địa chỉ đầu tiên là OUT sẽ bằng 1 khi đầu vào của lệnh này bằng 1 Khi đầu vào của lệnh bằng 0 thì các bit này vẫn giữ nguyên trạng thái. Trong đó số bit là giá trị của n Toán hạng OUT: Q, M, L, D n : là hằng số

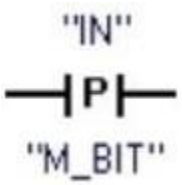
9) lệnh reset nhiều bit

L A D		Giá trị của các bit có địa chỉ đầu tiên là OUT sẽ bằng 0 khi đầu vào của lệnh này bằng 1 Khi đầu vào của lệnh bằng 0 thì các bit này vẫn giữ nguyên trạng thái. Trong đó số bit là giá trị của n Toán hạng OUT: Q, M, L, D n : là hằng số

10) Tiếp điểm phát hiện xung cạnh lên dạng 1

L A D		Tiếp điểm phát hiện cạnh lên sẽ phát ra một xung khi đầu vào tiếp điểm P có sự chuyển đổi từ mức thấp lên mức cao Trạng thái của tín hiệu được lưu lại vào "M_BIT" Độ rộng của xung này bằng thời gian của một chu kì quét.

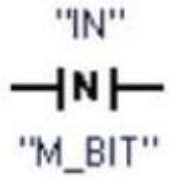
11) Tiếp điểm phát hiện xung cạnh lên dạng 2

L A D		Thay đổi trạng thái tín hiệu phía trước không ảnh hưởng đến "IN" Phát hiện sự thay đổi trạng thái của 1 tín hiệu "IN" từ 0 lên 1 Trạng thái của tín hiệu IN được lưu lại vào "M_BIT" Độ rộng của xung này bằng thời gian của một chu kì quét.

12) Tiếp điểm phát hiện xung cạnh xuống dạng 1

L A D		Tiếp điểm phát hiện cạnh xuống sẽ phát ra một xung khi đầu vào tiếp điểm này có sự chuyển đổi từ mức cao xuống mức thấp Trạng thái của tín hiệu được lưu lại vào "M_BIT" Độ rộng của xung này bằng thời gian của một chu kì quét.

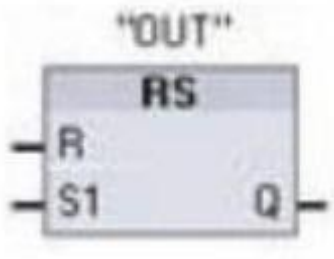
13) tiếp điểm phát hiện xung cạnh xuống dạng 2

L A D		Thay đổi trạng thái tín hiệu phía trước không ảnh hưởng đến "IN" Phát hiện sự thay đổi trạng thái của 1 tín hiệu "IN" từ 1 xuống 0 Trạng thái của tín hiệu IN được lưu lại vào "M_BIT" Độ rộng của xung này bằng thời gian của một chu kì quét.

14) lệnh SR fliplop

L A D		Mạch chốt RS ưu tiên Reset

15) lệnh RS fliplop

L A D		Mạch chốt RS ưu tiên Set

5.2. Sử dụng bộ Timer

Sử dụng lệnh Timer để tạo một chương trình trễ định thời. Số lượng của Timer phụ thuộc vào người sử dụng và số lượng vùng nhớ của CPU. Mỗi timer sử dụng 16 byte IEC_Timer dữ liệu kiểu cấu trúc DB. Step 7 tự động tạo khối DB khi lấy khối Timer

Kích thước và tầm của kiểu dữ liệu Time là 32 bit, lưu trữ như là dữ liệu Dint : T#-14d_20h_31m_23s_648ms đến T#24d_20h_31m_23s_647ms hay là - 2.147.483.648 ms đến 2.147.483.647 ms.

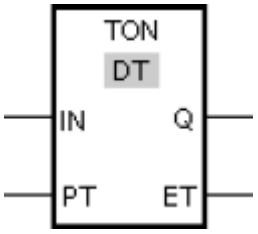
1) Timer tạo xung - TP

L A D		Timer TP tạo một chuỗi xung với độ rộng xung đặt trước. Thay đổi PT, IN không ảnh hưởng khi Timer đang chạy. Khi đầu vào IN được tác động vào timer sẽ tạo ra một xung có độ rộng bằng thời gian đặt PT

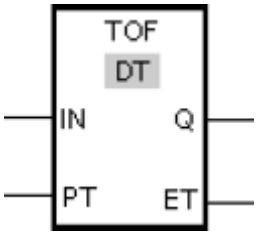
2) Timer trễ sườn lên có nhớ - Timer TONR

L A D		Thay đổi PT không ảnh hưởng khi Timer đang vận hành, chỉ ảnh hưởng khi timer đếm lại Khi ngõ vào IN chuyển sang "FALSE" khi vận hành thì timer sẽ dừng nhưng không đặt lại bộ định thì. Khi chân IN "TRUE" trở lại thì Timer bắt đầu tính thời gian từ giá trị thời gian đã tích lũy.

3) timer trễ không nhớ - TON

L A D		Khi ngõ vào IN ngừng tác động thì reset và dừng hoạt động Timer. Thay đổi PT khi Timer vận hành không có ảnh hưởng gì

4) timer trễ sườn xuống – TOF

L A D		Khi ngõ vào IN ngừng tác động thì reset và dừng hoạt động Timer. Thay đổi PT khi Timer vận hành không có ảnh hưởng gì

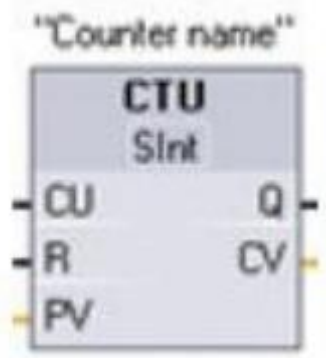
5.3. Sử dụng bộ Counter

Lệnh Counter được dùng để đếm các sự kiện ở ngoài hay các sự kiện quá trình ở trong PLC. Mỗi Counter sử dụng cấu trúc lưu trữ của khối dữ liệu DB để làm dữ liệu của Counter. Step 7 tự động tạo khối DB khi lấy lệnh.

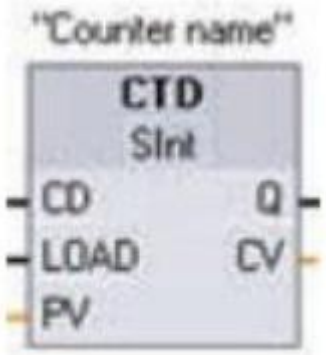
Tầm giá trị đếm phụ thuộc vào kiểu dữ liệu mà bạn chọn lựa. Nếu giá trị đếm là một số Interger không dấu, có thể đếm xuống tới 0 hoặc đếm lên tới tầm giới hạn. Nếu

giá trị đếm là một số interder có dấu, có thể đếm tới giá trị âm giới hạn hoặc đếm lên tới một số dương giới hạn.

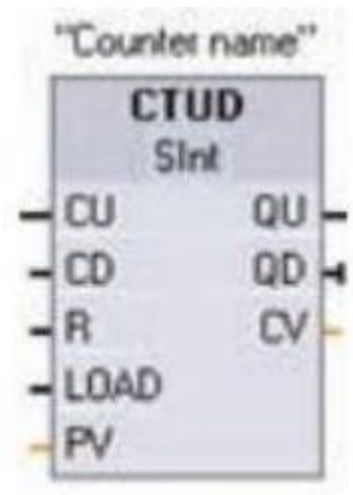
1) Counter đếm lên - CTU

L A D		Giá trị bộ đếm CV được tăng lên 1 khi tín hiệu ngõ vào CU chuyên từ 0 lên 1. Ngõ ra Q được tác động lên 1 khi CV>=PV. Nếu trạng thái R = Reset được tác động thì bộ đếm CV = 0.

2) Counter đếm xuống – CTD

L A D		Giá trị bộ đếm được giảm 1 khi tín hiệu ngõ vào CD chuyển từ 0 lên 1. Ngõ ra Q được tác động lên 1 khi CV <=0. Nếu trạng thái LOAD được tác động thì CV = PV.

3) Counter đếm lên xuống – CTUD

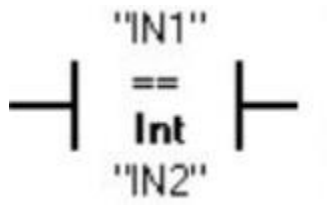
L A D		Giá trị bộ đếm CV được tăng lên 1 khi tín hiệu ngõ vào CU chuyển từ 0 lên 1. Ngõ ra QU được tác động lên 1 khi CV >=PV. Nếu trạng thái R = Reset được tác động thì bộ đếm CV = 0. Giá trị bộ đếm CV được giảm 1 khi tín hiệu ngõ vào CD chuyển từ 0 lên 1. Ngõ ra QD được tác động lên 1 khi CV <=0. Nếu trạng thái Load được tác động thì CV = PV.

5.4. So sánh

1)Lệnh so sánh

So sánh 2 kiểu dữ liệu giống nhau, nếu lệnh so sánh thỏa thì ngõ ra sẽ là mức 1 = TRUE

Kiểu dữ liệu so sánh là : SInt, Int, Dint, USInt, UDInt, Real, LReal, String, Char, Time, DTL, Constant.

L A D		Lệnh so sánh dùng để so sánh hai giá trị IN1 và IN2 bao gồm IN1 = IN2, IN1 >= IN2, IN1 <= IN2, IN1 < IN2, IN1 > IN2 hoặc IN1 <> IN2 So sánh 2 kiểu dữ liệu giống nhau, nếu lệnh so sánh thỏa thì ngõ ra sẽ là mức 1 = TRUE (tác động mức cao) và ngược lại Kiểu dữ liệu so sánh là : SInt, Int, Dint, USInt, UDInt, Real, LReal, String, Char, Time, DTL, Constant.

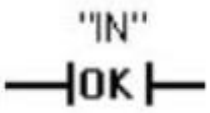
2)Lệnh trong khoảng In – range

L A D		Tham số : MIN, VAL, MAX Kiểu dữ liệu so sánh : SInt, Int, Dint, USInt, UInt, UDInt, Real, LReal, Constant So sánh 2 kiểu dữ liệu giống nhau, nếu so sánh MIN<=VAL<=MAX thỏa thì tác động mức cao và ngược lại

3)Lệnh ngoài khoảng out-of-range

L A D		Tham số : MIN, VAL, MAX Kiểu dữ liệu so sánh : SInt, Int, Dint, USInt, UInt, UDInt, Real, LReal, Constant So sánh 2 kiểu dữ liệu giống nhau, nếu so sánh MIN > VAL hoặc MAX < VAL thỏa thì tác động mức cao và ngược lại

4) Lệnh OK

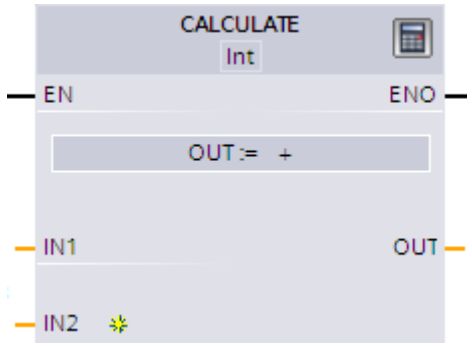
L A D		Tham số : IN Kiểu dữ liệu : Real, LReal Lệnh OK kiểm tra tính hợp lệ của toán tử

5) Lệnh NOT OK

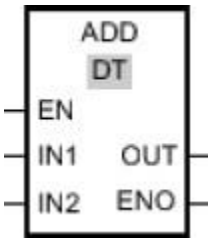
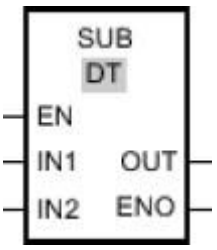
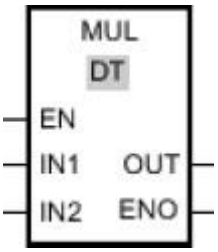
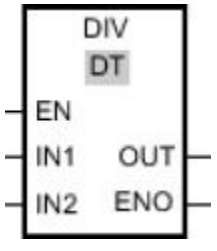
L A D		Tham số : IN Kiểu dữ liệu : Real, LReal Lệnh NOT_OK kiểm tra tính không hợp lệ của toán tử

5.5. Toán học

1) Lệnh tính toán

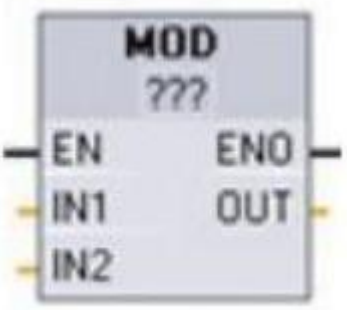
L A D		Công dụng : thực hiện phép toán từ các giá trị ngõ vào IN1, IN2, IN(n) theo công thức OUT=…(+,-,*,/) rồi xuất kết quả ra ngõ ra OUT. Các thông số ngõ vào dùng trong khối phải chung định dạng

2) Lệnh cộng, trừ, nhân, chia

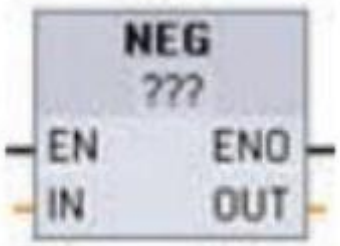
L A D	   	Lệnh cộng ADD : OUT = IN1 + IN2 Lệnh trừ SUB : OUT = IN1 – IN2 Lệnh nhân MUL : OUT = IN1*IN2 Lệnh chia DIV : OUT = IN1/IN2 Tham số IN1, IN2 phải cùng kiểu dữ liệu : SInt, Int, Dint, USInt, UInt, UDInt, Real, LReal, Constant Tham số OUT có kiểu dữ liệu : SInt, Int, Dint, USInt, UInt, UDInt, Real, LReal Tham số ENO = 1 nếu không có lỗi xảy ra trong quá trình thực thi. Ngược lại ENO = 0 khi có lỗi, một số lỗi xảy ra khi thực hiện lệnh này : -Kết quả toán học nằm ngoài phạm vi của kiểu dữ liệu. -Chia cho 0 (IN2 = 0) -Real/LReal : Nếu một trong những giá trị đầu vào là NaN sau đó được trả về NaN. -ADD Real/LReal : Nếu cả hai giá trị IN là INF có dấu khác nhau, đây là một khai báo không hợp lệ và

		được trả về NaN -SUB Real/LReal : Nếu cả hai giá trị IN là INF cùng dấu, đây là một khai báo không hợp lệ và được trả về NaN -MUL Real/LReal : Nếu một trong 2 giá trị là 0 hoặc là INF, đây là khai báo không hợp lệ và được trả về NaN. -DIV Real/LReal : Nếu cả hai giá trị IN bằng không hoặc INF, đây là khai báo không hợp lệ và được trả về NaN.

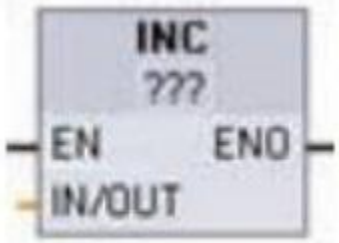
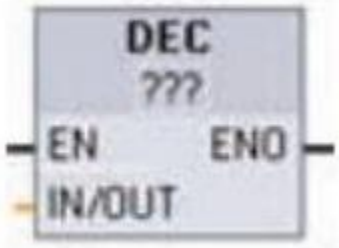
3) Lệnh lấy phần dư

L A D		Lệnh Modulo sẽ lấy phần dư của phép toán. Giá trị ngõ vào IN1 chia cho IN2 và giá trị phần dư sẽ được lưu vào OUT Tham số: EN : Bool ENO : Bool IN1 : SINT, INT, DINT, USINT, UINT, UDINT IN2 : SINT, INT, DINT, USINT, UINT, UDINT OUT : SINT, INT, DINT, USINT, UINT, UDINT

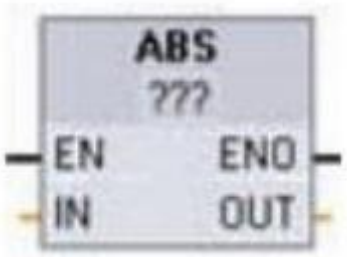
4) Lệnh phủ định

L A D		Lệnh NEG đảo ngược dấu hiệu số học của giá trị ở trong tham số và lưu trữ các kết quả trong tham số OUT Tham số : EN : Bool – cho phép ngõ vào ENO: Bool – cho phép ngõ ra -ENO = 1 : không có lỗi -ENO = 0: kết quả giá trị nằm ngoài tầm giá trị của kiểu dữ liệu IN : toán tử đầu vào SInt, INt, Dint, Real, LReal, Constant OUT : toán tử đầu ra Sint, Int, Dint, Real, LReal

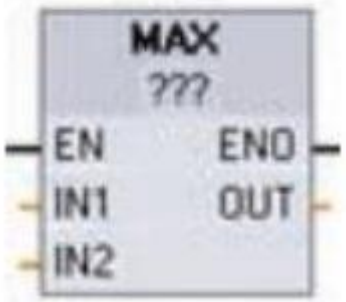
5) Lệnh tăng, giảm

L A D	 	Tăng / giảm giá trị kiểu số Interger lên / xuống một đơn vị Tham số : EN : cho phép ngõ vào IN/OUT : toán tử ngõ vào và ra ENO : cho phép ngõ ra -ENO = 1 : không có lỗi -ENO = 0: kết quả nằm ngoài tầm giá trị của kiểu dữ liệu

6) Lệnh giá trị tuyệt đối

L A D		Tinh giá trị tuyệt đối của một số nguyên hoặc số thực của tham số IN và lưu trữ kết quả vào tham số OUT Tham số : EN : cho phép ngõ vào IN : Toán tử ngõ vào OUT : Toán tử ngõ ra ENO : Cho phép ngõ ra

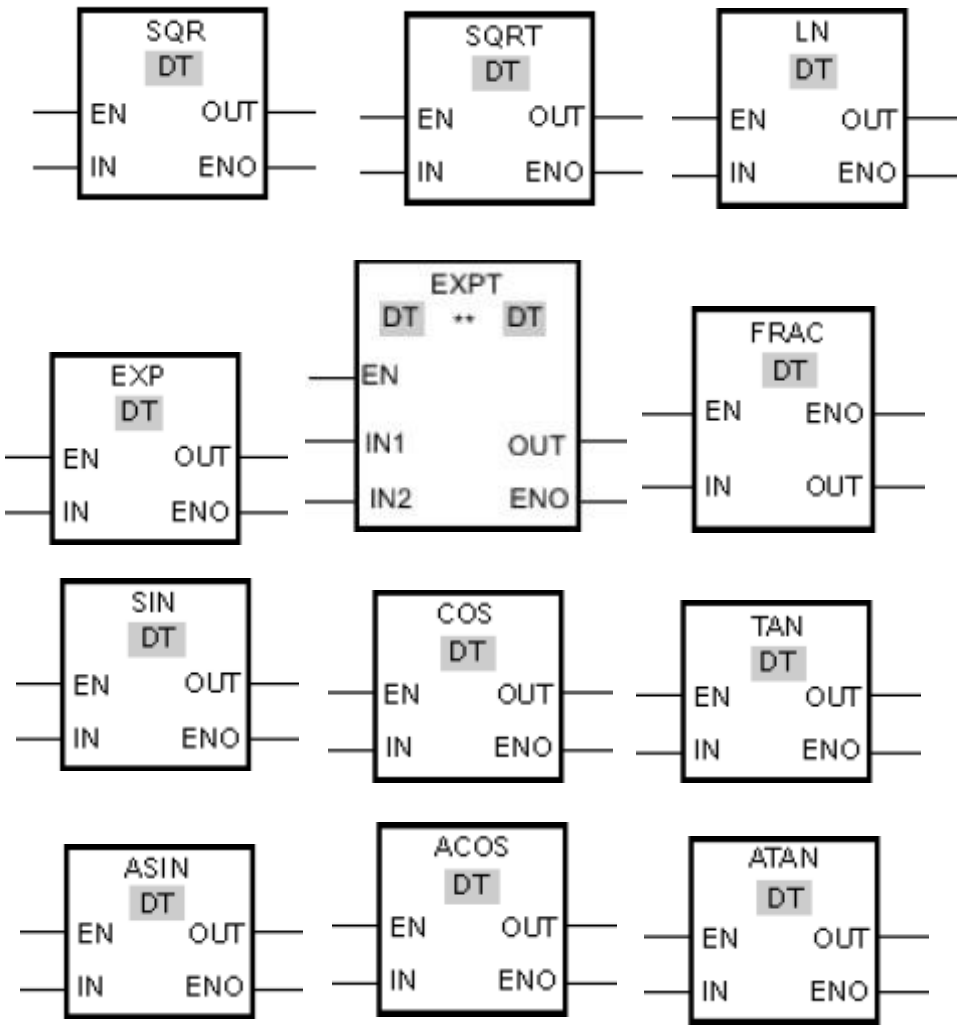
7) Lệnh giá trị nhỏ nhất và lớn nhất

L A D	 	Lệnh MIN/MAX so sánh các giá trị đầu vào và trả lại giá trị nhỏ nhất/ lớn nhất ở đầu ra Tham số : EN : cho phép ngõ vào IN : Toán tử đầu vào, có thể lên tới 32 đầu vào OUT : Toán tử ngõ ra ENO : cho phép ngõ ra

8) Lệnh giới hạn

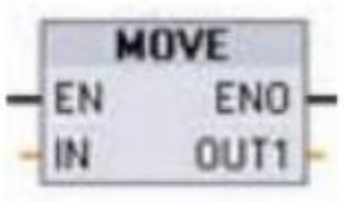
L A D		Công dụng : Giới hạn giá trị của ngõ vào IN trong khoảng của ngõ vào MIN và MAX. Nếu giá trị của IN đáp ứng MIN < IN <MAX thì giá trị của IN được copy vào giá trị của OUT. Còn nếu giá trị của IN < MIN thì giá trị cảu MIN được copy vào OUT, và nếu giá trị của IN > MAX thì giá trị của MAX được copy vào OUT Lệnh chỉ được thực hiện khi tín hiệu ngõ vào là 1 tại ngõ vào EN, Nếu lệnh được thực hiện mà không có lỗi xảy ra thì tại ngõ ra ENO cũng có giá trị bằng 1. Ngõ ra ENO có trạng thái 0 nếu 1 trong số các điều kiện sau đây không thỏa mãn : -Ngõ vào EN có tín hiệu “0” -Các thông số nhập vào không đúng định dạng -Các toán hạng không đúng giá trị -Giá trị Min lớn hơn giá trị Max

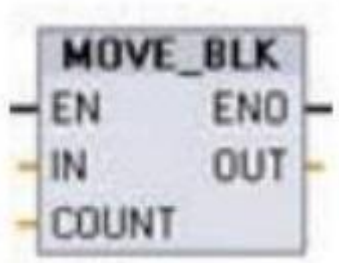
9) Lệnh toán học số thực dấu chấm động



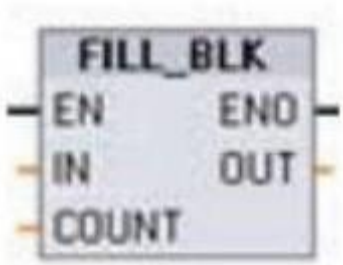
5.6. Di chuyển MOVE

1) Lệnh MOVE

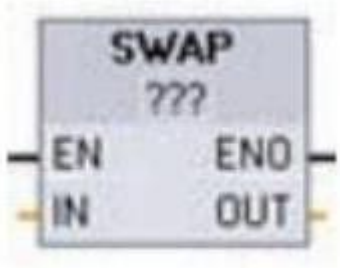
L A D		Lệnh Move di chuyển nội dung ngõ vào IN đến ngõ ra OUT mà không làm thay đổi giá trị ngõ IN Tham số: EN : cho phép ngõ vào ENO : cho phép ngõ ra IN : nguồn giá trị đến OUT1: Nơi chuyển đến

		Lệnh Move_BLK sao chép các nội dung của một vùng nhớ IN đến một bộ nhớ xác định khác. Số lượng các giá trị được sao chép được quy định trong COUNT. Hoạt động sao chép theo hướng tăng dần các địa chỉ Tham số: EN : cho phép ngõ vào ENO : cho phép ngõ ra IN : nguồn giá trị đến COUNT : số giá trị sao chép OUT1: Nơi chuyển đến

2) Lệnh làm đầy FILL

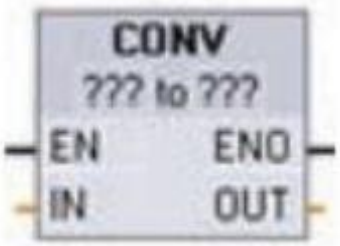
L A D		Công dụng : dùng để lấp đầy một vùng nhớ với nội dung tại một vùng nhớ khác. Lệnh Fill block di chuyển nội dung của một vùng nhớ tới một vùng nhớ xác định. Hành động vận chuyển các biến sao chép theo hướng tăng dần

3) Lệnh đảo Swap

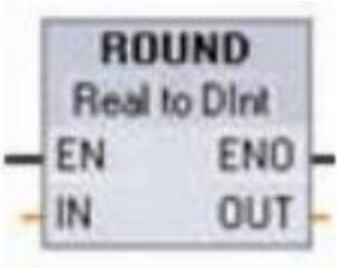
L A D		Công dụng : Đổi thứ tự của 2 byte hay 4 byte thành phần của một Word hay một Dword. Nó không làm đổi thứ tự của các bit trong mỗi byte

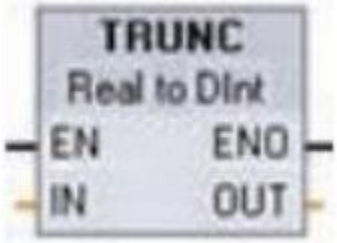
5.7. Chuyển đổi

1)Lệnh CONV

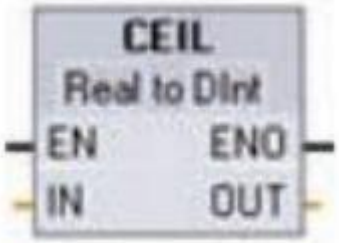
L A D		Công dụng : chuyển đổi từ kiểu dữ liệu này sang kiểu dữ liệu khác Tham số : IN : giá trị ngõ vào OUT : giá trị sau khi chuyển đổi

2) Lệnh làm tròn ROUND và cắt bỏ TRUNCATE

L A D		Lệnh ROUND : Chuyển đổi số thực thành sô Interger. Các phần phân số của sô thực được làm tròn đến số nguyên gần nhất. Nếu số thực nằm ở giữa 2 số nguyên thì số thực này được làm tròn thành số nguyên chẵn. Ví dụ ROUND (10.5) = 10, ROUND (11.5) = 12. Lệnh TRUNC : chuyển đổi số thực thành số interger. Phần phân số của số thực bị cắt bỏ

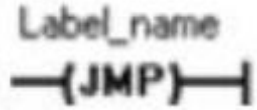
		

3) Lệnh CEILING và FLOOR

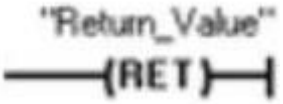
L A D	 	Lệnh CEIL : chuyển đổi số thực thành số nguyên nhỏ nhất lớn hơn hay bằng số thực đó Lệnh FLOOR : Chuyển đổi số thực thành số nguyên nhỏ hơn hya bằng số thực đó.

5.8. Lệnh điều khiển chương trinh

1)Lệnh nhảy JUMP và nhãn LABEL

L A D		Công dụng : Dừng chương trình đang chạy và tiếp tục trên một network khác, network này được xác định bởi 1 jump label.

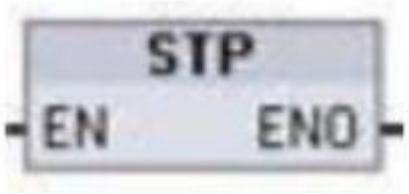
2) Lệnh điều khiển thực thi RET

L A D		Công dụng : Để dừng việc thực thi trong một khối hàm và chỉ đươc tiếp tục sau khi có lệnh gọi khối hàm đó.

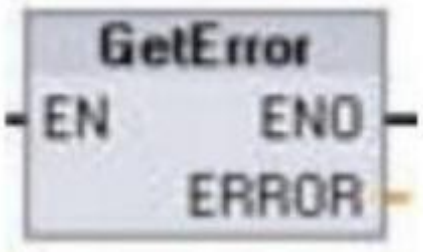
3) Lệnh Re – trigger giám sát quét chu kỳ

L A D		Công dụng : Khởi động lại việc giám sát chu kỳ của CPU. Thời gian giám sát được cấu hình trong phần cứng. Việc khởi động lại thời gian giám sát chu kỳ để ngăn chặn lỗi.

4) Lệnh ngừng quét chu kỳ

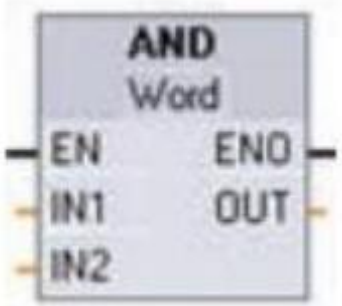
L A D		Công dụng : Đặt PLC về chế độ STOP, do vậy ngừng việc thực hiện chương trình

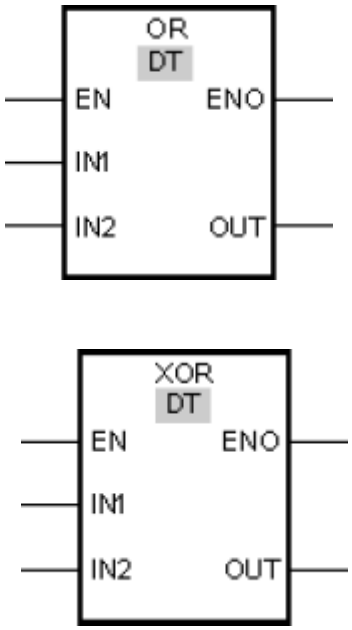
5) Lệnh lấy lỗi GET ERROR

L A D		Công dụng : Truy vấn các lỗi xãy ra trong một khối

5.9. Toán tử word logic

1) Lệnh AND, OR và XOR

L A D		Công dụng : -Lệnh AND kết hợp các giá trị ngõ vào IN1 và IN2 theo các bit tương ứng theo phép AND logic, xuất kết quả tại OUT - Lệnh OR kết hợp các giá trị ngõ vào IN1 và IN2 theo các bit tương ứng theo phép OR logic, xuất kết quả tại OUT - Lệnh XOR kết hợp các giá trị ngõ vào IN1 và IN2 theo các bit tương ứng theo phép XOR logic,

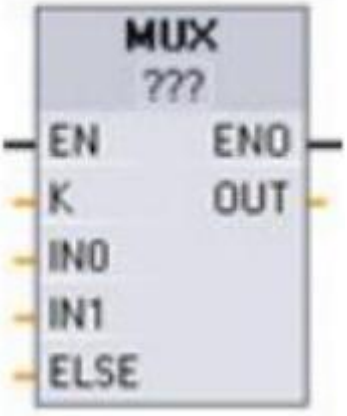
		xuất kết quả tại OUT

2) Lệnh đảo INVERT

L A D		Công dụng : Đảo bit tín hiệu tại ngõ vào IN. Giá trị của những bit lấy bù sẽ được gửi tối ngõ ra

3) Lệnh SELECT, MULTIPLEX và DEMULTIPLEX

L A D		Công dụng : -Lệnh SEL : Dựa vào tín hiệu ngõ vào G, lệnh SEL lựa chọn ngõ vào IN1 hoặc IN0 và di chuyển nội dung của nó vào ngõ ra OUT. +Nếu G = 0 □ OUT = IN0

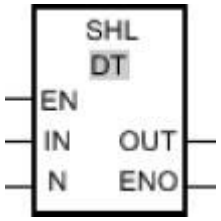
	 	+Nếu G = 1 □ OUT = IN1 -Lệnh MUX : Sao chép nội dung của một ngõ vào xác định tới ngõ ra OUT. Nếu giá trị của tham số K lớn hơn số ngõ vào hiện hữu thì nội dung của tham số ELSE sẽ được sao chép tới ngõ ra OUT

5.10. Dịch chuyển và xoay vòng

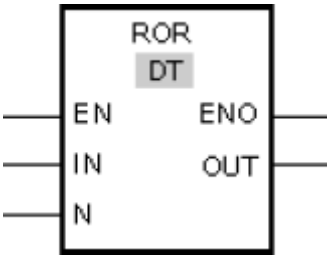
1)Lệnh dịch phải

L A D		Công dụng : Dịch chuyển nội dung của toán hạng tại ngõ vào IN theo từng bit sang bên phải và truy xuất kết quả ra ngõ ra OUT. Thông số N để xác định số bit dịch chuyển

2)Lệnh dịch trái

L A D		Công dụng : Dịch chuyển nội dung của toán hạng tại ngõ vào IN theo từng bit sang bên trái và truy xuất kết quả ra ngõ ra OUT. Thông số N để xác định số bit dịch chuyển

3) Lệnh quay phải

L A D		Công dụng : Xoay nội dung của một toán hạng tại ngõ vào IN theo từng bit về hướng bên phải và truy xuất tại ngõ ra OUT Thông số N xác định số bit dịch chuyển

4) Lệnh quay trái

L A D		Công dụng : Xoay nội dung của một toán hạng tại ngõ vào IN theo từng bit về hướng bên trái và truy xuất tại ngõ ra OUT Thông số N xác định số bit dịch chuyển