

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG THÁI NGUYÊN
KHOA CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Bài giảng:

THUYẾT KẾ LOGIC
(Tài liệu hướng dẫn)

Thái Nguyên, năm 2012

M C L C

| | |
|--|-----------|
| M C L C | 1 |
| Ch  ng 1..... | 5 |
| C S L Y THUY T I U KHI N LOGIC..... | 5 |
| 1.1. Kh i ni m v logic hai tr ng th i..... | 5 |
| 1.2. C c h m v c c c t nh ch t c b n c a i s logic..... | 5 |
| 1.2.1. H m logic c b n..... | 5 |
| 1.2.2. C c t nh ch t v c m t s h th c c b n c a i s logic..... | 7 |
| 1.3. C c ph  ng ph p bi u di n h m logic..... | 8 |
| 1.3.1. Ph  ng ph p bi u di n th nh b ng | 8 |
| 1.3.2. Ph  ng ph p h nh h c | 9 |
| 1.3.3. Ph  ng ph p bi u th c i s (ph  ng ph p gi i t ch)..... | 9 |
| 1.3.4. Ph  ng ph p bi u di n h m logic b ng b ng Karnaugh..... | 10 |
| 1.4. C c ph  ng ph p t i thi u ho  h m logic | 10 |
| 1.4.1. Ph  ng ph p t i thi u ho  h m logic b ng bi n i i s | 10 |
| 1.4.2. Ph  ng ph p t i thi u ho  h m logic theo thu t to n..... | 11 |
| 2.1. Kh i ni m v c m  h nh to n h c h i u khi n logic t h p..... | 15 |
| 2.2. C ch ph n t ch h i u khi n logic t h p v c ng d ng | 15 |
| 2.3. Ph  ng ph p t ng h p m ch logic t h p..... | 17 |
| 2.4. Kh i ni m chung h i u khi n logic m ch tr nh t | 21 |
| 2.4.1. Gi i thi u v c m t s nh ngh a..... | 21 |
| 2.4.2. M t s ph n t nh trong logic tr nh t | 21 |
| 2.5.1. Ph  ng ph p b ng chuy n tr ng th i | 22 |
| 2.5.2. Ph  ng ph p h nh tr ng th i | 23 |
| 2.5.3. Ph  ng ph p l u | 24 |
| 2.6. T ng h p m ch tr nh t | 26 |
| 2.6.1. T ng h p theo ph  ng ph p b ng tr ng th i..... | 26 |
| 2.6.2. T ng h p theo ph  ng ph p h nh Mealy ho c Moore..... | 29 |
| 2.7. V i d v m ch tr nh t | 29 |
| Ch  ng 3..... | 30 |
| H I U KHI N LOGIC KH TR NH PLC | 30 |

| | |
|---|-----------|
| 3.1. Khái niệm chung | 30 |
| 3.2. Cấu tạo của PLC..... | 30 |
| 3.3. Sơ cấu trúc PLC..... | 31 |
| 3.4. Nguyên lý làm việc..... | 32 |
| 3.5. Ưu nhược điểm..... | 33 |
| 3.6. Ứng dụng: | 33 |
| 3.7. Trình tự thiết kế hệ thống logic ứng dụng PLC..... | 33 |
| 4.1. Sơ cấu trúc của PLC..... | 35 |
| 4.1.1. Cấu hình chung | 35 |
| 4.1.2. Mô tả các đèn báo và công tắc | 36 |
| 4.1.3. Cấu trúc bên trong | 37 |
| 4.1.4. Module mở rộng vào ra (Module mở rộng)..... | 38 |
| 4.1.5. Thiết bị hiển thị chương trình | 39 |
| 4.1.6. Cấu trúc chương trình..... | 40 |
| 4.2. Ngôn ngữ lập trình của S7-200..... | 41 |
| 4.2.1. Giới thiệu chung..... | 41 |
| 4.2.2. Bảng tóm tắt mã lệnh cơ bản của S7-200 | 42 |
| 4.2.3. Cú pháp lệnh của S7-200 | 45 |
| 4.3. Một số ví dụ ứng dụng S7-200..... | 56 |
| Chương 5..... | 62 |
| THIẾT BỊ UHKN LOGIC KHUYNH S7-300 | 62 |
| 5.1. Giới thiệu chung..... | 62 |
| 5.2. Các module của PLC S7-300 | 62 |
| 5.2.1. Module CPU | 63 |
| 5.2.2. Module mở rộng..... | 63 |
| 5.3. Kỹ thuật liên lạc và phân chia bên trong | 64 |
| 5.3.1- Kỹ thuật liên lạc: | 64 |
| 5.3.2. Cấu trúc bên trong của CPU | 64 |
| 5.4. Vòng quét chương trình..... | 65 |
| 5.5. Nhúng khối OB vào bộ nhớ..... | 65 |
| 5.6. Ngôn ngữ lập trình của S7-300..... | 67 |
| 5.6.1. Cấu trúc lệnh và trình tự thao tác | 67 |

| | |
|---|----|
| 5.6.2. Các l nh c b n..... | 67 |
| 5.6.3.Các l nh i u khi n ch ng trnh | 68 |
| 5.6.4 B th i gian (Timer)..... | 71 |
| 5.6.5. B m (Counter)..... | 75 |
| 5.6.6- K thu t s d ng con tr | 78 |

Chương 1

C S LÝ THUYẾT I U KHI N LOGIC

1.1. Khái niệm v logic hai tr ng thái

Trong cu c s ng hàng ngày, nhi u s v t hi n t ng th ng bi u hi n hai m t i l p thông qua hai tr ng thái i l p rõ r t. Ví d nh khi nói v giá c và ch t l ng hàng hoá ta th ng có khái ni m t và r hay t t và x uí

Trong k thu t i n và i u khi n, ta th ng có khái ni m v hai tr ng thái: óng và c t, kín hay h , làm vi c hay không làm vi c, có i n hay m t i n, í

Trong toán h c, l ng hoá hai tr ng thái i l p c a s v t hay hi n t ng ng i ta dùng hai giá tr : 0 và 1. Ta g i ó là các giá tr 0 và 1 logic.

Các nhà bác h c ã xây d ng c s toán h c tính toán các hàm và bi n ch l y v i hai giá tr 0 và 1 này, hàm và bi n ó c g i là hàm và bi n logic, c s toán h c tính toán các hàm và bi n ó g i là i s logic. i s logic còn có tên là i s Boole vì l y theo tên nhà toán h c Boole, ng i có công trong vi c xây d ng nên công c i s logic.

1.2. Các hàm và các tính ch t c b n c a i s logic



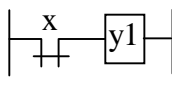
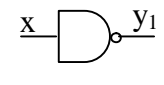
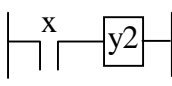
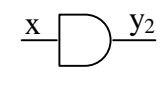
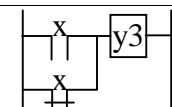
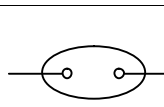
1.2.1. Hàm logic c b n

M t hàm $y = f(x_1, x_2, \dots, x_n)$ v i các bi n x_1, x_2, \dots, x_n ch nh n hai giá tr : 0 ho c 1 và hàm y c ng ch nh n hai giá tr : 0 ho c 1, thì x_1, x_2, \dots, x_n c g i là các bi n logic và y là hàm logic.

1.2.1.1. Hàm logic m t bi n: $y = f(x)$

Vì bi n x s nh n m t trong hai giá tr 0 ho c 1, nên hàm y có 4 kh n ng hay th ng g i là 4 hàm y_0, y_1, y_2, y_3

B ng 1.1 Hàm logic m t bi n $y = f(x)$

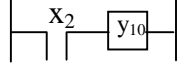
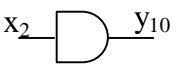
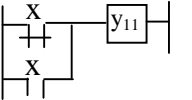
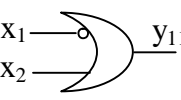
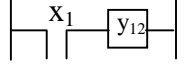
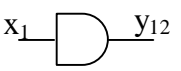
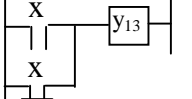
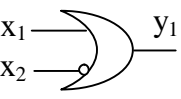
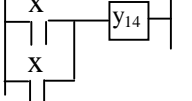
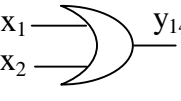
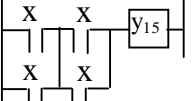
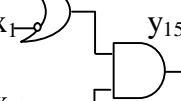
| <i>T n hàm</i> | <i>B ng chân lý</i> | | | <i>Thu t toán logic</i> | <i>Ký hi u s</i> | | <i>Ghi chú</i> |
|----------------|---------------------|----------|----------|----------------------------|---|---|---------------------|
| | <i>x</i> | <i>0</i> | <i>1</i> | | <i>M ch r le</i> | <i>Kh i i n t</i> | |
| Hàm không | y_0 | 0 | 0 | $y_0 = 0; y_0 = x \bar{x}$ |  |  | Hàm luôn b ng không |
| Hàm o | y_1 | 1 | 0 | $y_1 = \bar{x}$ |  |  | |
| Hàm l p | y_2 | 0 | 1 | $y_2 = x$ |  |  | |
| Hàm n v | y_3 | 1 | 1 | $y_3 = x + \bar{x}$ |  |  | Hàm luôn b ng 1 |

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

Vì 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 logic cơ bản như bảng 1.2

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

| Tên hàm | B ng chân lý | | | | | Thu t toán logic | Ký hi u s | | Ghi chú |
|------------------|--------------|---|---|---|---|---|-----------|------------|---------------------------|
| | x_1 | 1 | 1 | 0 | 0 | | M ch r le | Kh i i n t | |
| | 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 tr b ng 0 |
| Hàm Pic | y_1 | 0 | 0 | 0 | 1 | $y_1 = \bar{x}_1 \cdot \bar{x}_2$ $y_1 = \overline{x_1 + x_2}$ | | | |
| Hàm c m x_1 | y_2 | 0 | 0 | 1 | 0 | $y_2 = \bar{x}_1 x_2$ | | | |
| Hàm o x_1 | y_3 | 0 | 0 | 1 | 1 | $y_3 = \bar{x}_1$ | | | Ch ph thu c vào x_1 |
| Hàm c m x_2 | y_4 | 0 | 1 | 0 | 0 | $y_4 = x_1 \bar{x}_2$ | | | |
| Hàm o x_2 | y_5 | 0 | 1 | 0 | 1 | $y_5 = \bar{x}_2$ | | | Ch ph thu c vào x_2 |
| Hàm ho c lo i tr | y_6 | 0 | 1 | 1 | 0 | $y_6 = \bar{x}_1 \bar{x}_2 + x_1 x_2$ | | | C ng module |
| Hàm Cheffer | y_7 | 0 | 1 | 1 | 1 | $y_7 = \bar{x}_1 + \bar{x}_2 = x_1 x_2$ | | | |
| Hàm Và | y_8 | 1 | 0 | 0 | 0 | $y_8 = x_1 \cdot x_2$ | | | |
| Hàm cùng d u | y_9 | 1 | 0 | 0 | 1 | $y_9 = x_1 x_2 + \bar{x}_1 \bar{x}_2$ | | | |

| | | | | | | | | | |
|-----------------------|----------|---|---|---|---|----------------------------|---|--|----------------------|
| Hàm l p theo x_2 | y_{10} | 1 | 0 | 1 | 0 | $y_{10} = x_2$ |  |  | Ch ph thu c x_2 |
| Hàm kéo theo x_2 | y_{11} | 1 | 0 | 1 | 1 | $y_{11} = \bar{x}_1 + x_2$ |  |  | |
| Hàm l p theo x_1 | y_{12} | 1 | 0 | 1 | 0 | $y_{12} = x_1$ |  |  | Ch ph thu c x_1 |
| Hàm kéo theo x_1 | y_{13} | 1 | 1 | 0 | 1 | $y_{13} = x_1 + \bar{x}_2$ |  |  | |
| Hàm ho c | y_{14} | 1 | 1 | 1 | 0 | $y_{14} = x_1 + x_2$ |  |  | |
| Hàm v | y_{15} | 1 | 1 | 1 | 1 | |  |  | Hàm luôn b ng 1 |

Ta có nh n xét: Các hàm i x ng qua tr c n m gi a y_7 và y_8 , ngh a là $y_0 = \bar{y}_{15}$, $y_1 = \bar{y}_{14}$, í

1.2.1.3. Hàm logic n bi n $y = f(x_1, x_2, \bar{x}_1, \bar{x}_2, x_n)$

V i hàm logic n bi n, m i bi n nh n m t trong hai giá tr 0 ho c 1 nên ta có 2^n t h p bi n, m i t h p bi n l i nh n hai giá tr 0 ho c 1, do v y s hàm logic là 2^{2^n} . V i s bi n b ng n = 1 ta có 4 kh n ng t o hàm, n = 2 có 16 còn v i n = 3 s có 256 kh n ng t o hàm, nh v y khi s bi n nhi u thì s hàm có kh n ng t o thành r t l n. Tuy nhiên t t c các kh n ng này u c bi u hi n qua các kh n ng t ng logic, tích logic và ngh ch o logic c a các bi n. Trong t t c các hàm c t o thành, ta c bi t chú ý n lo i hàm t ng chu n và hàm tích chu n. Hàm t ng chu n là hàm ch a t ng các tích mà m i tích có t t c các bi n c a hàm. Hàm tích chu n là hàm ch a tích các t ng mà m i t ng u có t t c các bi n c a hàm.

1.2.2. Các tính ch t và m t s h th c c b n c a i s logic

1.2.2.1. nh lu t giao hoán i v i c ng và nhân logic

$$a + b = b + a \quad a.b = b.a$$

1.2.2.2. nh lu t k th p i v i c ng và nhân logic

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

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

1.2.2.3. *nh luật phân phối:* $a(b+c)=ab+ac$

1.2.2.4. *nh luật negation (De - Morgan)* $\overline{a+b}=\overline{a}.\overline{b}$ $\overline{a.b}=\overline{a}+\overline{b}$

1.2.2.5. *nh luật phủ định hai lần:* $\overline{\overline{a}}=a$

1.2.2.6. *Quy tắc tính giá trị các hằng số 0 và 1* $a.1=a$ $a.0=0$ $a+0=a$ $a+1=1$

1.2.2.7. *Quy tắc tính giá trị biến và phép nghịch đảo* $a.\overline{a}=0$ $a+\overline{a}=1$

1.2.2.8. *Luật đồng nhất* $a+a=a$ $a.a=a$

1.3. Các phép toán pháp biến đổi hàm logic

1.3.1. Phép toán pháp biến đổi thành bảng

Với phép toán pháp này, các giá trị của hàm logic phụ thuộc vào các biến của biến đổi 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 của biến đổi thành bảng như sau:

Bảng 1.3

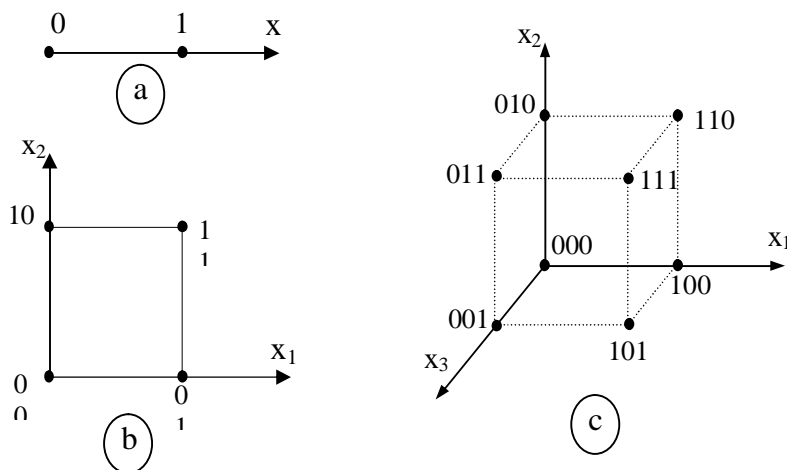
| Giá trị thập phân (nhân phân) của tổ hợp biến | x_1 | x_2 | x_3 | y |
|---|-------|-------|-------|-----|
| 0 (000) | 0 | 0 | 0 | 1 |
| 1 (001) | 0 | 0 | 1 | 0 |
| 2 (010) | 0 | 1 | 0 | 0x0 |
| 3 (011) | 0 | 1 | 1 | 0x0 |
| 4 (100) | 1 | 0 | 0 | 0 |
| 5 (101) | 1 | 0 | 1 | 1 |
| 6 (110) | 1 | 1 | 0 | 0x0 |
| 7 (111) | 1 | 1 | 1 | 1 |

Ghi chú: Dấu 0x0 là tổ hợp biến mà giá trị hàm không xác định (có thể là 0 hoặc 1)

Ưu điểm của phép toán pháp này là dễ nhìn, ít nhầm lẫn. Nhược điểm là cồng kềnh, số biến là khi số biến lớn.

1.3.2. Phương pháp biểu diễn 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 của 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 khác nhau biểu thị thay đổi giá trị của một biến. Hình 1.1 là cách biểu diễn hàm logic 1, 2 và 3 biến.



Hình 1.1. Biểu diễn hình học hàm logic
a - Hàm 1 biến; b - Hàm 2 biến; c - Hàm 3 biến

Như các hình vẽ phương pháp này là khi số biến lớn sẽ rất phức tạp.

1.3.3. Phương pháp biểu thức tích (phương pháp giản tích)

Ngay từ đầu chúng ta chứng minh rằng, một hàm logic n biến bất kỳ bao giờ cũng có thể biểu diễn thành các hàm tích đơn giản và tích đơn giản.

* Cách viết hàm dưới dạng tích đơn giản:

- Chú ý quan tâm đến tổ hợp biến mà hàm có giá trị bằng 1. Số lượng hàm bằng 1 sẽ chính là số tích của các tổ hợp biến, mà tích của chúng là minterm, ký hiệu là m_i .
- Trong minterm, các biến có giá trị bằng 1 sẽ giữ nguyên, còn các biến có giá trị bằng 0 thì lấy giá trị nghịch đảo.
- Hàm tích đơn giản là tích các minterm.

* Cách viết hàm dưới dạng tích đơn giản:

- Chú ý quan tâm đến tổ hợp biến mà hàm có giá trị bằng 0. Số lượng hàm bằng 0 sẽ chính là số tích của các tổ hợp biến, mà tích của chúng là maxterm, ký hiệu là M_i .
- Trong maxterm, các biến có giá trị bằng 0 sẽ giữ nguyên, còn các biến có giá trị bằng 1 thì lấy giá trị nghịch đảo.
- Hàm tích đơn giản là tích các maxterm.

1.3.4. Phương pháp biên độ n hàm logic boolean Karnaugh

Nguyên tắc xây dựng bảng Karnaugh là:

- Biên độ n m t hàm logic n biến, c n thành l p m t b ng có 2^n ô; m i ô t ng ng v i m t t h p bi n. ánh s th t các ô trong b ng t ng ng v i giá tr c a t h p bi n.

- Các ô c nh nhau ho c i x ng nhau ch cho phép khác nhau v giá tr c a m t bi n.

- Trong các ô ghi giá tr c a hàm t ng ng v i giá tr c a t h p bi n ó.

Ví d : Hình 1.2 là b ng Karnaugh c a hàm 2 bi n

| | | x_2 | |
|-------|---|----------------------------|----------------------|
| | | 0 | 1 |
| x_1 | 0 | 0 $\bar{x}_1 \bar{x}_2$ | 1 $\bar{x}_1 x_2$ |
| | 1 | 2 $x_1 \bar{x}_2$ | 3 $x_1 x_2$ |

| | | x_2 | |
|-------|---|-------|-----|
| | | 0 | 1 |
| x_1 | 0 | 0 | 1 |
| | 1 | 1 | õxõ |

Hình 1.2. B ng Karnaugh cho hàm 2 bi n; Ví d : $y = \Sigma 1,2$ và $N=3$

| | | $x_2 \ x_3$ | | | |
|-------|---|--------------------------------------|--------------------------------|--------------------------|--------------------------------|
| | | 00 | 01 | 11 | 10 |
| x_1 | 0 | 0 $\bar{x}_1 \bar{x}_2 \bar{x}_3$ | 1 $\bar{x}_1 \bar{x}_2 x_3$ | 3 $\bar{x}_1 x_2 x_3$ | 2 $\bar{x}_1 x_2 \bar{x}_3$ |
| | 1 | 4 $x_1 \bar{x}_2 \bar{x}_3$ | 5 $x_1 \bar{x}_2 x_3$ | 7 $x_1 x_2 x_3$ | 6 $x_1 x_2 \bar{x}_3$ |

| | | $x_2 \ x_3$ | | | |
|-------|---|-------------|----|-----|-----|
| | | 00 | 01 | 11 | 10 |
| x_1 | 0 | 0 | 1 | 1 | õxõ |
| | 1 | õxõ | 1 | õxõ | 0 |

Hình 1.3. B ng Karnaugh cho hàm 3 bi n; Ví d : $y = \Sigma 1,3,5$ v i $N=2,4,7$

1.4. Các phương pháp tối thiểu hóa hàm logic

1.4.1. Phương pháp tối thiểu hoá hàm logic boolean

Vì c rút g n hàm th ng d a vào các lu t và các h th c c b n c a i s logic

Ví d : T i thi u 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 tr c quan c a ph ng pháp nên nhi u khi k t qu a ra v n không bi t rõ là ã t i thi u hay ch a, nh v y ây không ph i là ph ng pháp ch t ch cho phép t ng hoá quá trình t i thi u hoá hàm logic.

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

Thuật toán dùng như là các phương pháp: bảng Karnaugh và Quine Mc. Cluskey

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

a. Một số khái niệm và định nghĩa

+ Định nghĩa: Một hàm tích chập của các biến của hàm xuất phát, nếu hàm có n biến thì hàm là tích chập n biến.

Hàm 1 là hàm mà hàm có giá trị bằng 1;

Hàm 0 là hàm mà hàm có giá trị bằng 0;

Hàm không xác định là hàm mà một số hàm có thể lấy một trong hai giá trị bằng 0 hoặc 1.

+ Tích chập tối thiểu: Tích chập tối thiểu là tích có số biến là các tối thiểu hàm có giá trị bằng 1 hoặc có giá trị không xác định.

+ Tích quan trọng: Tích quan trọng là tích chập tối thiểu mà giá trị hàm chỉ duy nhất bằng 1 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 tiến hành theo các bước như trên hình 1.2.

Ví dụ: Cho hàm $y = f(x_1, x_2, x_3, x_4)$ với các hàm bằng 1 là $L = 2, 3, 7, 12, 14, 15$; và các hàm không xác định là $N = 6, 13$ (bảng 1.4). Hãy tối thiểu hoá hàm bằng phương pháp Quine Mc. Cluskey

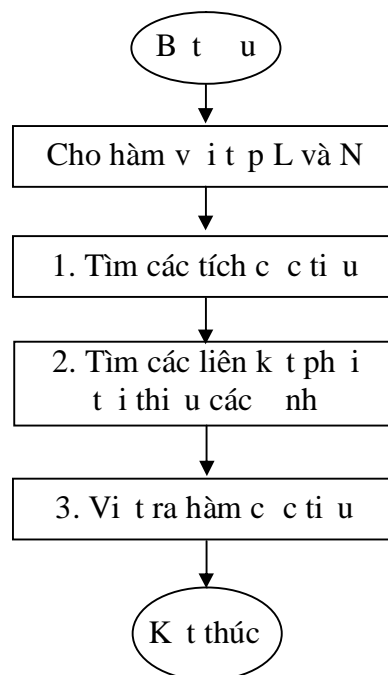
Cách làm:

Bước 1: Tìm các tích chập tối thiểu

Các công việc tiến hành như sau:

+) Lập bảng biến đổi các giá trị hàm bằng 1 và các giá trị không xác định về mã nhị phân của các biến (bảng 1.4a).

+) Sắp xếp các tập hợp biến theo mã nhị phân theo thứ tự các chữ số tăng dần từ 0, 1, 2, 3, ... Như vậy đây ta có 4 tập hợp: tập hợp 1 (gồm các số chẵn 1 chữ số), tập hợp 2 (gồm các số chẵn 2 chữ số), tập hợp 3 (gồm các số chẵn 3 chữ số), tập hợp 4 (gồm các số chẵn 4 chữ số) (bảng 1.4b).



Hình 1.4. Các bước tối thiểu hoá hàm logic theo phương pháp Quine

+) So sánh m i t h p th i v i m t t h p th i+1, n u hai t h p ch khác nhau m t c t thì k t h p hai t h p ó thành m t t h p m i, n g th i thay c t s khác nhau c a 2 t h p c b ng m t g ch ngang (-) vào hai t h p c (b ng 1.4 c)

+) Ti p t c công vi c: T b ng 1.4c ta ch n ra các t h p ch khác nhau l ch s 1 và có cùng g ch ngang (-) trong m t c t, ngh a là có cùng bi n v a c gi n c b ng 1.4c, nh v y ta có b ng 1.4d.

Các t h p tìm c b ng 1.4d là t h p cu i cùng, không còn kh n ng k t h p n a, ây chính là các tích c c ti u c a hàm f ã cho và c vi t:

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

- 1 1 - (ph các nh 6, 7, 14, 15) : $x_2 x_3$

1 1 - - (ph các nh 12, 13, 14, 15): $x_1 x_2$

B ng 1.4

| B ng a | | B ng b | | | B ng c | | B ng d | |
|-------------|------------------------------------|----------|-------------|------------------------------------|----------|------------------------------------|------------------------|------------------------------------|
| S th p phân | S nh phân ($x_1 x_2 x_3 x_4$) | S ch s 1 | S th p phân | S nh phân ($x_1 x_2 x_3 x_4$) | Liên k t | S nh phân ($x_1 x_2 x_3 x_4$) | Liên k t | S nh phân ($x_1 x_2 x_3 x_4$) |
| 2 | 0010 | 1 | 2 | 0010 | 2,3 | 001- | 2,3,6,7 2,6,3,7 | 0-1- |
| 3 | 0011 | 2 | 3 | 0011 | 2,6 | 0-10 | 6,7,14,15 6,14,7,15 | -11- |
| 6 | 0110 | | 6 | 0110 | 3,7 | 0-11 | 12,13,14,15 | 11-- |
| 12 | 1100 | | 12 | 1100 | 6,7 | 011- | 12,14,13,15 | |
| 7 | 0111 | 3 | 7 | 0111 | 6,14 | -110 | | |
| 13 | 1101 | | 13 | 1101 | 12,13 | 110- | | |
| 14 | 1110 | | 14 | 1110 | 12,14 | 11-0 | | |
| 15 | 1111 | 4 | 15 | 1111 | 7,15 | -111 | | |
| | | | | | 13,15 | 11-1 | | |
| | | | | | 14,15 | 111- | | |

B c 2: Tìm các tích quan trọng

Vì c tìm các tích quan trọng c ng c tỉ n hành theo trình t nh u b c nh . Gi thi t có i b c nh , v i i = 0, 1, 2, 3, í , k

G i L_i là t p các nh l ang xét b c th i, lúc này không quan tâm n các nh có giá tr không xác nh n a.

Z_i là t p các tích c c tỉ u b c nh th i.

E_i là t p các tích quan trọng b c nh th i.

Trình t công vi c c tỉ n hành nh sau:

+) V i i = 0

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

$$Z_0 = Z = (\overline{x_1 x_3}, x_2 x_3, x_1 x_2)$$

Xác nh các tích quan trọng E_0 t các t p L_0 và Z_0 nh sau:

L p m t b ng trong ó m i hàng ng v i m t tích c c tỉ u thu c Z_0 , m i c t ng v i m t nh thu c L_0 . ánh d u õxõ vào các ô trong b ng ng v i tích c c tỉ u b ng 1.

Xét t ng c t, c t nào ch có m t d u õxõ thì tích c c tỉ u ng v i nó là tích quan trọng nh b ng 1.7.

B ng 1.5.

| $L_0 \backslash Z_0$ | 2 | 3 | 7 | 12 | 14 | 15 |
|----------------------|-----|-----|---|-----|----|----|
| $\overline{x_1 x_3}$ | (x) | (x) | x | | | |
| $x_2 x_3$ | | | x | | x | x |
| $x_1 x_2$ | | | | (x) | x | x |

+) V i i = 1

L_1 : Tìm L_1 t L_0 b ng cách lo i kh i L_0 các nh l c a E_0 .

Z_1 : Tìm Z_1 t Z_0 b ng cách lo i kh i Z_0 các tích trong E_0 và các tích ã n m trong hàng ã c ch n t E_0 (ó là các tích không c n thi t).

L p b ng t ng t nh trên, t b ng ó c ng b ng cách t ng t trên s tìm c tích quan trọng E_1 .

Công vi c c tỉ p t c cho n khi h t các tích c c tỉ u

$$L_{i+1} = L_i - E_i$$

Chương 2

HỆ THỐNG KHI N T H P VÀ LOGIC TRÌNH T

2.1. Khái niệm và mô hình toán học hệ thống logic trình t

Mạch logic trình t là mạch mà trạng thái đầu ra của mạch chỉ phụ thuộc vào trạng thái đầu vào chứ không phụ thuộc vào trình tự tác động của các đầu vào. Theo quan niệm này thì mạch trình t là mạch tĩnh, không có phần nhúng, nghĩa là trạng thái ổn định của các phần tử trong mạch hoàn toàn không phụ thuộc vào trạng thái tín hiệu đầu ra.

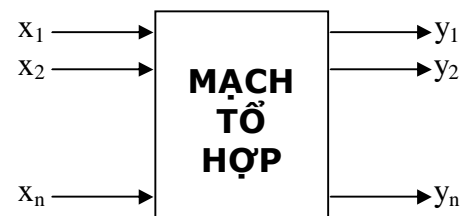
Với mô hình toán học, giả thiết mạch trình t có n đầu vào và m đầu ra với các x_i ($i = 1-n$) và y_j ($j = 1-m$), ta ký hiệu:

$X = \{x_1, x_2, \dots, x_n\}$ là tập các tín hiệu đầu vào.

$Y = \{y_1, y_2, \dots, y_m\}$ là tập các tín hiệu đầu ra.

Mạch trình t có thể biểu diễn bằng phương trình logic Boolean sau:

$$y_j = f_j(x_1, x_2, \dots, x_n) \quad \forall j = 1-m.$$



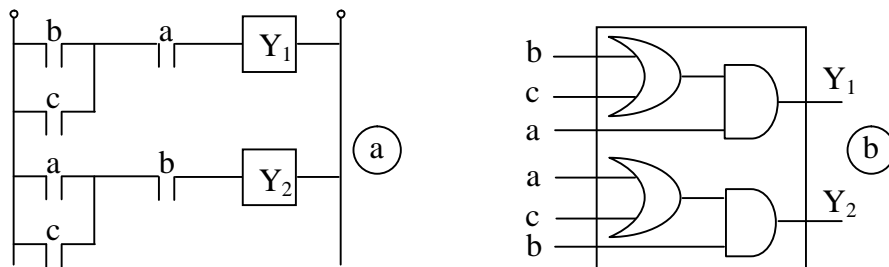
Hình 2.1: Mô hình toán học của mạch tổ hợp

Có thể biểu diễn mô hình toán học của mạch trình t theo sơ đồ như hình 2.1

2.2. Cách phân tích hệ thống logic trình t và ứng dụng

Bài toán phân tích có nhiệm vụ là tìm mạch trình t đã có, mô tả hoạt động của mạch, viết các hàm logic của các đầu ra theo các biến đầu vào và ngược lại có thể xét tính vi tính thi u hoá mạch.

Giả thiết có mạch logic trình t như hình 2.2, ta tiến hành phân tích mạch này.



Hình 2.2. Mạch trình t có 3 biến vào và 2 đầu ra
a. Ký hiệu theo mạch rơle; b. Ký hiệu theo mạch s

Vì phân tích mạch cần tiến hành theo các bước sau:

-Thăng kê số biến vào và ra, trên cơ sở đó lập bảng mô tả trạng thái của hệ thống.

Mạch hình 2.2 có 3 biến vào là a, b, c và 2 đầu ra là Y_1, Y_2 , bảng trạng thái của mạch biểu diễn như sau (bảng 2.1).

Bảng 2.1

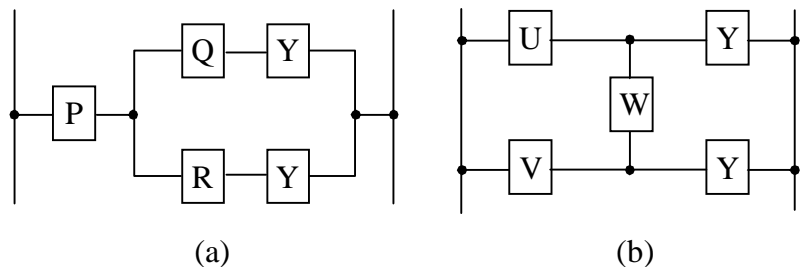
| a | b | c | Y_1 | Y_2 |
|---|---|---|-------|-------|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 |

- Viết các hàm logic mô tả quan hệ giữa tín hiệu đầu ra theo tín hiệu đầu vào. Sử dụng các phép toán logic cơ bản ta có thể viết các quan hệ này. Ví dụ hình 2.2, hoặc mô tả bởi bảng 2.1, ta có:

$$Y_1 = (b + c).a$$

$$Y_2 = (a + c).b$$

- Xét khả năng tối giản mạch: Giả sử thiết kế mạch như hình 2.2, ta có cấu trúc như hình 2.3a hoặc hình 2.3b



Hình 2.3

Với cấu trúc như hình 2.3a ta có:

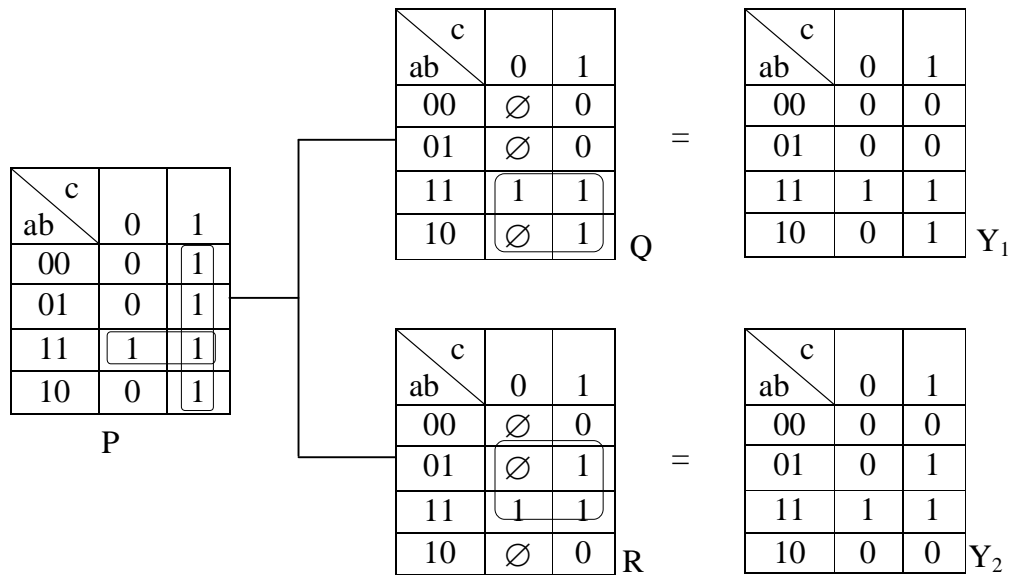
$$Y_1 = P.Q, \quad Y_2 = P.R$$

Với cấu trúc như hình 2.3b ta có:

$$Y_1 = U + V.W, \quad Y_2 = V + U.W$$

Với cấu trúc như hình 2.3a, mặt khi P, Q, R là tập hợp của 3 biến a, b, c, ta có bảng Karnaugh của P, Q, R và Y_1, Y_2 như hình 2.4. Đây, các giá trị của Y_1 và Y_2 có chép lại kết quả của bảng 2.1.

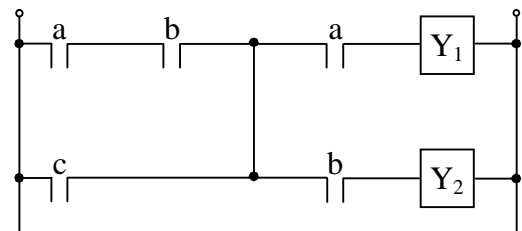
Các giá trị của P, Q, R có thể chia thành hai nhóm: một nhóm giá trị bất biến và một nhóm có thể nhận giá trị tùy ý. Vì rằng mạch P sản sinh tín hiệu P, nên tất cả giá trị đầu ra $Y_1 = 1$ thì P, Q bất biến phải bằng 1 với tất cả các tổ hợp a, b, c; ngược lại khi $Y_1 = 0$ thì chắc chắn P hoặc Q bằng 0 là. Khi tổ hợp abc = 100 nghĩa là $Y_1 = 0$, ta có thể chắc chắn P = 0, còn Q có thể bằng 0 hoặc 1. Với các ô trong bảng Karnaugh có giá trị $Y_2 = 1$ và $Y_1 = 0$ với điều kiện P = 1 thì bất biến Q phải bằng 0. Tóm lại suy ra: có 4 trong 8 ô của bảng Karnaugh của giá trị Q có giá trị bất biến và có 4 ô có giá trị tùy ý.



Hình 2.4.

Vì với $abc = 001$, chỉ khi $P = 1$ thì cùng một lúc Q và R phải bằng 0. Từ lập luận này ta điền các giá trị trong bảng Karnaugh hình 2.4. Với cách tính như hàm bảng Karnaugh như đã giới thiệu phần trước, ta có: $P = ab + c$, $Q = a$, $R = b$.

Với các biến P, Q, R và tìm c , ta vẽ các sơ đồ mạch logic như hình 2.5, sơ đồ hình 2.2 ta biết các tín hiệu vào. Trong thực tế, vì mạch rời rạc thì việc gán tín hiệu vào (mạch tích cực) rất có ý nghĩa, còn việc gán thì tín hiệu này thì ngược lại không đáng kể.



Hình 2.5

Vì phân tích theo cấu trúc hình 2.3b cũng xảy ra tương tự.

2.3. Phương pháp tổng hợp mạch logic tổ hợp

Với tổng hợp mạch tổ hợp thì kết quả là thiết kế mạch tổ hợp. Nhiệm vụ chính của đây là thiết kế các mạch tổ hợp thỏa mãn yêu cầu kỹ thuật nhằm mục đích gì. Bài toán tổng hợp là bài toán phức tạp, vì ngoài các yêu cầu về chức năng logic, việc tổng hợp mạch còn phải thu vào việc sử dụng các phần tử, chẳng hạn như phần tử là loại rời rạc-công tắc, là các phần tử bán dẫn hay vi mạch chuần, ... Việc thiết kế phần tử thì ngoài nguyên lý chung về mạch logic còn đòi hỏi phải bổ sung những nguyên tắc riêng lúc tổng hợp mạch.

Nguyên tắc chung khi tổng hợp mạch logic tổ hợp là:

+ T các yêu c u công ngh ta a ra c các hàm logic tho mãn các yêu c u ã cho.

+ Th c hi n t i thi u hoá các hàm logic ã thi t l p c, tìm ra các hàm t i gi n.

+ Th c hi n m ch logic t h p b ng vi c s d ng các r le, công t c t (t ng h p m ch r le), ho c b ng các ph n t logic AND, OR, NAND, NOR ã chu n hoá u vào và u ra.

Ví d 1: Hãy thi t k m ch logic t h p khi cho hàm logic 4 bi n (4 u vào):

$$Y = f(a,b,c,d) = \Sigma 2,4,5,7,8,13; \text{ và } N = 0,1,6,9,10,15.$$

Gi i:

1/ T i thi u hoá hàm ã cho, ây ta s d ng ph ng pháp Quine Mc.Cluskey. Ti n trình th c hi n c mô t theo b ng 2.2.

B ng 2.2

| S th p phân | S nh phân | | | | <i>Liên k t l n1</i> | <i>Liên k t l n2</i> | <i>K t qu</i> |
|-------------|-----------|---|---|---|----------------------|------------------------------|--------------------|
| | a | b | c | d | | | |
| 0 | 0 | 0 | 0 | 0 | 0,1 0,2 | 0, 1, 4, 5 A 0, 1, 8, 9 B | 0 - 0 - - 0 0 - |
| 1 | 0 | 0 | 0 | 1 | 0,4 | 0, 2, 4, 6 C | 0 - - 0 |
| 2 | 0 | 0 | 1 | 0 | 0,8 | 0, 2, 8, 10 D | - 0 - 0 |
| 4 | 0 | 1 | 0 | 0 | 1,5 | | |
| 8 | 1 | 0 | 0 | 0 | 1,9 | | |
| | | | | | 2,6 | | |
| 5 | 0 | 1 | 0 | 1 | 2,10 | 4, 5, 6, 7 E | 0 1 - - |
| 6 | 0 | 1 | 1 | 0 | 4,5 | | |
| 9 | 1 | 0 | 0 | 1 | 4,6 | 1, 5, 9, 13 F | - - 0 1 |
| 10 | 1 | 0 | 1 | 0 | 8,9 8,10 | | |
| 7 | 0 | 1 | 1 | 1 | 5,7 5,13 | 5, 7, 13, 15 G | - 1 - 1 |
| 13 | 1 | 1 | 0 | 1 | 6,7 9,13 | | |
| 15 | 1 | 1 | 1 | 1 | 7,15 13,15 | | |

2/ Tìm các tích c c tỉ u và tích quan tr ng: D a vào b ng 2.2 ta tìm c 7 tích c c tỉ u:

$$A = \bar{a}\bar{c}; B = \bar{b}\bar{c}; C = \bar{a}\bar{d}; D = \bar{b}\bar{d}; E = \bar{a}b; F = \bar{c}d; G = b.d.$$

T c các tích c c tỉ u ta l p b ng 2.3 tìm các tích quan tr ng

B ng 2.3. B ng các tích c c tỉ u

| | 2 | 4 | 5 | 7 | 8 | 13 |
|---|---|---|---|---|---|----|
| A | | x | x | | | |
| B | | | | | x | |
| C | x | x | | | | |
| D | x | | | | x | |
| E | | x | x | x | | |
| F | | | x | | | x |
| G | | | x | x | | x |

V i tr ng h p này ta th y, không có tích nào là tích quan tr ng, ta có th ch n m t s tích sao cho chúng v a bao các nh l (có nh xu t h i n m t s t h p). ây có th ch n G, B và C, ho c là G, D và A, ho c là G, D và C, ho c là G, D và E, ho c là D, E và F. T t c các kh năng này u dùng 6 tín hi u vào, vì r ng m i thành ph n u có 2 tín hi u (l y t 4 u vào a, b, c, d). gi s ta ch n t h p G, B và C thì hàm Y s là:

$$Y = b.d + \bar{b}\bar{c} + \bar{a}\bar{d} \quad (a)$$

S m ch r le ng v i tr ng h p ch n G, C và B nh hình 2.6a.

có th t c hàm n gi n ta có th xét v i t p bù c a t p L, t c là:

$$\bar{Y} = \bar{f}(a,b,c,d) = \bar{L}(3,11,12,14) + N(0,1,6,9,10,15)$$

(ây ta th c h i n tìm hàm t i gi n c a \bar{Y} , b ng ph ng pháp t ng t nh lúc tìm Y, nh ng các nh l bây gi c ch n là t p nh 0 c a hàm Y ã cho).

C ng áp d ng ph ng pháp Quine Mc.Cluskey. K t qu c hàm t i gi n:

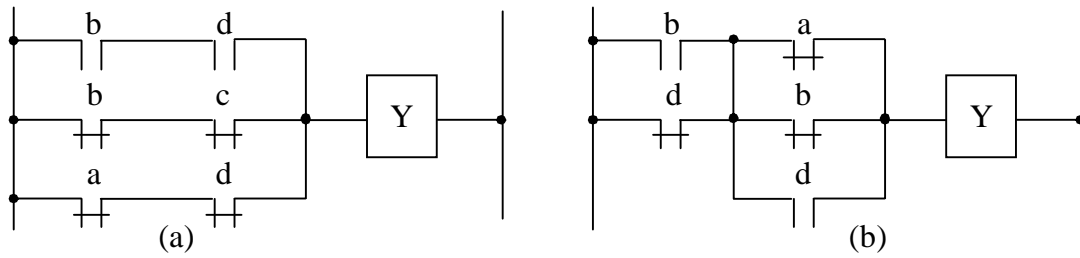
$$\bar{Y} = \bar{b}d + a.b.\bar{d}$$

S d ng các lu t c a i s logic ta tìm c Y:

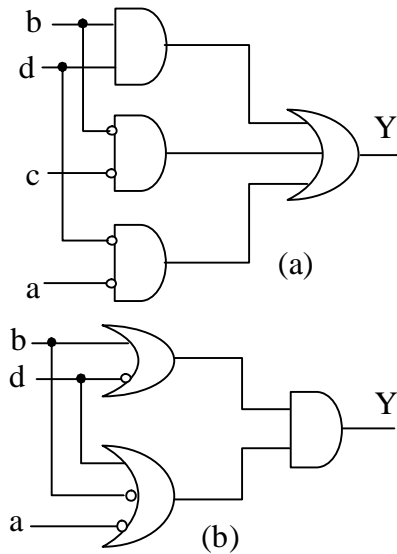
$$Y = \bar{\bar{Y}} = \overline{\bar{b}d + a.b.\bar{d}} = (b + \bar{d}).(\bar{a} + \bar{b} + d) \quad (b)$$

V i hàm t i gi n này ta có 5 tín hi u vào, s m ch r le c bi u di n nh hình 2.6b.

S các ph n t logic cho 2 tr ng h p (a) và (b) c cho trên hình 2.7.



Hình 2.6



Hình 2.7

| Các t h p | u vào | | | | F |
|--------------------|-------|---|---|---|---|
| | a | b | c | d | |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 2 | 0 | 0 | 1 | 0 | 0 |
| 3 | 0 | 0 | 1 | 1 | 0 |
| 4 | 0 | 1 | 0 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 | 0 |
| 6 | 0 | 1 | 1 | 0 | 0 |
| 7 | 0 | 1 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 0 |
| 10 | 1 | 0 | 1 | 0 | 1 |
| 11 | 1 | 0 | 1 | 1 | 1 |
| 12 | 1 | 1 | 0 | 0 | 1 |
| 13 | 1 | 1 | 0 | 1 | 1 |
| 14 | 1 | 1 | 1 | 0 | 1 |
| 15 | 1 | 1 | 1 | 1 | 1 |

→ $\bar{a}bcd$

→ $a\bar{b}\bar{c}d$

→ $a\bar{b}cd$

→ $ab\bar{c}d$

→ $ab\bar{c}d$

→ $abc\bar{d}$

→ $abcd$

Hình 2.8

Ví dụ 2: (hình 2.8)

M t t b o m t tài li u, có 4 chìa khoá v i 4 ng i gi : tr ng phòng (a), phó tr ng phòng (b) và 2 nhân viên (c & d). Cách m nh sau: a ch có th m khi có m t b ho c c. Còn b, c và d ch có th m khi có ít nh t 2 ng i khác. Tìm ph ng tr ình logic c a khoá (u ra F) theo các chìa khoá (abcd).

$$F = \bar{a}bcd + a\bar{b}\bar{c}d + a\bar{b}cd + ab\bar{c}\bar{d} + ab\bar{c}d + abc\bar{d} + abcd$$

V i: $\bar{a}bcd + a\bar{b}\bar{c}d = \bar{a}bc$; $a\bar{b}\bar{c}d + ab\bar{c}\bar{d} = abc$; $ab\bar{c}d + abc\bar{d} = abc$, ta có:

$$F = \bar{a}bcd + \bar{a}bc + abc + abc = \bar{a}bcd + ac + abc = c(a + \bar{a}bd) + abc$$

$$F = ac + bcd + abc = a(c + c) + bcd = a(c + b) + bcd$$

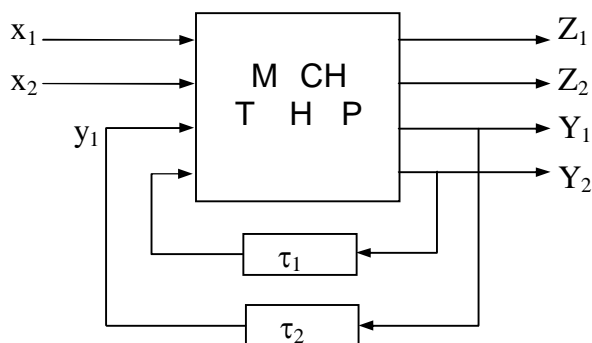
$$F = a(c + b) + bcd$$

2.4. Khái niệm chung về hệ thống logic mạch trình tự

2.4.1. Giới thiệu và mô tả sơ đồ nguyên lý

Mạch trình tự hay mạch dãy (sequential circuits) là mạch mà trong đó trạng thái của đầu ra (tín hiệu ra) không chỉ phụ thuộc tín hiệu vào mà còn phụ thuộc vào trình tự tác động của tín hiệu vào, nghĩa là có nhiều trạng thái. Như vậy, về mặt thời gian thì mạch trình tự không chỉ có các phần tử logic mà còn có các phần tử nhớ.

Sơ cấu trúc cơ bản của mạch trình tự như hình 3.1. Ở đây là mạch có ngõng hai đầu ra tín hiệu qua các biến nhị phân (Y_1, Y_2 và y_1, y_2).



Hình 2.9. Cấu trúc cơ bản của mạch

Hoạt động của mạch trình tự có thể hình dung thay thế các biến nhị phân Y . Trong quá trình làm việc, do sự thay thế các tín hiệu vào $X(x_1, x_2, \dots)$ sẽ dẫn đến thay thế các tín hiệu ra $Z(Z_1, Z_2, \dots)$ và các tín hiệu nhị phân $Y(Y_1, Y_2, \dots)$. Sự thay thế các $Y(Y_1, Y_2, \dots)$ sẽ dẫn đến sự thay thế các biến $y(y_1, y_2, \dots)$ sau thời gian (τ_1, τ_2, \dots). Sự thay thế các biến $y(y_1, y_2, \dots)$ lại có thể dẫn đến sự thay thế các tín hiệu ra Z , các Y , rồi sự thay thế các

Y lại dẫn đến sự thay thế các y . Quá trình này cứ diễn ra liên tục như vậy sẽ làm cho mạch ổn định, nghĩa là mạch không làm việc. Yêu cầu đặt ra là phải làm cho mạch ổn định, nghĩa là khi mạch trình tự có sự thay thế các tín hiệu vào sẽ chuyển từ một trạng thái ổn định này sang một trạng thái ổn định khác và trải qua một số giai đoạn trung gian không ổn định. Khái niệm ổn định và không ổn định này không chỉ liên quan đến toàn mạch mà còn liên quan đến từng phần tử.

Về mặt hình 2.9 thì:

- Mạch ổn định khi: $Y_1 = y_1$ và $Y_2 = y_2$;
- Mạch không ổn định khi: $Y_1 = \overline{y_1}$ và $Y_2 = \overline{y_2}$

2.4.2. Mô tả phần nhớ trong logic trình tự

Như đã nói trên, tính chất của mạch trình tự là có phần tử nhớ.

2.4.2.1. Về thời gian

2.4.2.2. Các mạch cơ bản

a. Mạch lật RS

b. Mạch lật D

2.5. Các phương pháp mô tả mạch logic trình tự

2.5.1. Ph ng pháp b ng chuy n tr ng thái

Ph ng pháp này mô t quá trình chuy n i tr ng thái d i hình th c b ng, trong b ng hình 2.10 bao g m:

⊕ Các c t c a b ng ghi các bi n vào và bi n ra:

Các tín hi u vào là các tín hi u i u khi n (α , β , γ , $\dot{\gamma}$), có th là tín hi u i u khi n c a ng i v n hành, tín hi u c a thi t b ch ng trình ho c các tín hi u phát ra t các thi t b công ngh .

| Tr ng thái | Tín hi u vào | | | | Tín hi u ra | |
|-------------|---------------|---------|----------|----------------|--------------|-------|
| | α | β | γ | $\dot{\gamma}$ | Y_1 | Y_2 |
| S_1 | | | | | | |
| S_2 | | | | | | |
| S_3 | | | | | | |

Hình 2.10

Các tín hi u ra (Y_1 , Y_2 , $\dot{\gamma}$) là tín hi u k t qu c a quá trình i u khi n và ghi c t u ra.

⊕ Các hàng c a b ng ghi các tr ng thái trong c a m ch (S_1 , S_2 , S_3 , $\dot{\gamma}$) (hình 2.11). S hàng c a b ng ch rõ s tr ng thái trong c n có c a h .

| Tr ng thái | Tín hi u vào | | | Tín hi u ra | |
|-----------------------|---------------------|---------------------|---------------------|--------------|-------|
| | α | β | γ | Y_1 | Y_2 |
| S_1 (t c th p) | $\textcircled{S_1}$ | S_2 | S_3 | 0 | 0 |
| S_2 (o chi u quay) | S_1 | $\textcircled{S_2}$ | | 1 | 0 |
| S_3 (ng ng máy) | | | $\textcircled{S_3}$ | 0 | 0 |

Hình 2.11

⊕ Các ô giao nhau c a c t bi n vào và các hàng tr ng thái s ghi tr ng thái c a m ch. N u tr ng thái m ch trùng v i tên hàng thì ó là tr ng thái ò n nh , n u tr ng thái m ch không trùng v i tên hàng thì ó là tr ng thái ò không n nh .

⊕ Các ô giao nhau c a c t tín hi u ra và các hàng tr ng thái s ghi giá tr tín hi u ra t ng ng.

b ng trên hình 3.3: α , β , γ là tín hi u vào, Y_1 , Y_2 là tín hi u ra. H có 3 tr ng thái: S_1 (làm vi c t c th p), S_2 (o chi u quay), S_3 (ng ng máy).

M i tr ng thái c a h có th di n t b ng ngôn ng và kèm theo m t con s g i tên tr ng thái ó. Ví d ta xét tr ng thái S_1 , lúc này máy ho t ng t c th p. N u lúc này cho bi n α tác ng thì máy v n làm vi c tr ng thái S_1 (tr ng thái S_1 là tr ng thái n nh), n u cho bi n β tác ng thì máy s chuy n sang tr ng thái S_2 (nh ng tr ng thái S_2 ghi hàng S_1 là không n nh - tr ng thái trung gian, m ch ang chu n b chuy n n tr ng thái n nh khác), n u cho bi n γ tác ng thì máy s chuy n sang tr ng thái S_3 (tr ng thái S_3 không n nh). Các bi n u ra Y_1 , Y_2 lúc này u b ng không. T ng t nh v y ta s lý gi i k t qu các hàng 2 và 3.

Khi b ng tr ng thái ch có 1 tín hi u ra thì có th không dùng c t tín hi u ra, các giá tr tín hi u ra c ghi luôn vào các ô tr ng thái chuy n (hình 2.12).

i u quan tr ng ây là ghi c y và úng các tr ng thái trong các ô c a b ng. Có hai cách th c hi n công vi c này:

- *Cách 1*: Trích hết dữ liệu bài toán, các hiệu bit và quá trình công nghệ, tổng ghi các trạng thái nên hiển nhiên có. Tiếp theo ghi các trạng thái chuyển rõ ràng (các trạng thái này có sự ghi trạng thái khác với tất cả các hàng - các trạng thái xuất phát), nếu trạng thái nào không bị tích hợp thì trạng thái bổ sung sau.

| Biến vào \ Trạng thái | α | β | γ |
|-----------------------|----------|---------|----------|
| S_1 | $S_2/1$ | $S_4/0$ | $S_3/0$ |
| S_2 | $S_4/1$ | $S_2/0$ | $S_4/1$ |
| S_3 | $S_1/1$ | $S_1/1$ | $S_1/1$ |
| S_4 | $S_3/1$ | $S_4/0$ | $S_2/0$ |
| S_5 | $S_5/0$ | $S_3/0$ | $S_4/0$ |

Hình 2.12

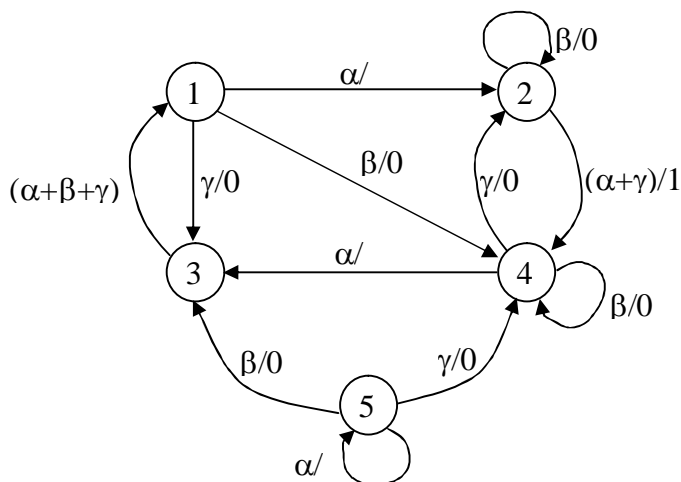
- *Cách 2*: Phân tích xem xét tổng thể trong trạng thái. Vì làm này là logic, chính xác và rõ ràng, tuy nhiên khi phân tích không thể quá chi lý để phân biệt giữa các ô có trạng thái gần nhau, do vậy rất khó in ý các ô.

2.5.2. Phương pháp hình trạng thái

Hình trạng thái là hình vẽ mô tả các trạng thái chuyển của mạch logic trình tự, hình vẽ các nhánh và các cung nhánh trên đó ghi các tín hiệu vào/ra và kết quả. Phương pháp này thường chỉ dùng cho hàm một đầu ra. Ta xét hai loại: hình Mealy và hình Moore.

2.5.2.1. hình Mealy

Hình Mealy (hình 3.5) gồm các nhánh biểu diễn các trạng thái trong mạch và các cung nhánh, trên các cung ghi biến tác động và kết quả hàm khi chu kỳ tác động của biến đó. Hình Mealy chính là chuyển biến trạng thái thành dạng đồ thị.



Hình 2.13. hình Mealy ứng với bảng trạng thái

2.5.2.2. hình Moore

Trong hình Moore, các nhánh là các trạng thái và giá trị trạng thái, còn các cung nhánh hiển thị ghi biến tác động.

T b ng tr ng thái (hình 2.12), ta có th l p h ình Moore theo các b c nh sau:

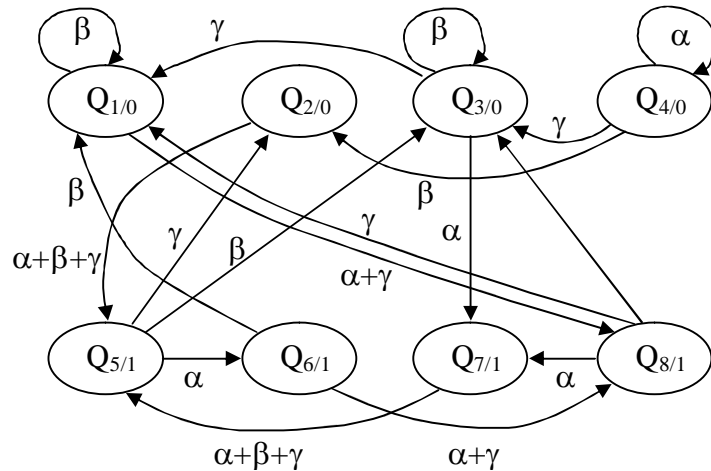
B c 1: T các ô ng v i c p tr ng thái và k t qu h ình 2.12, ta gán m t tr ng thái t ng ng Q cho h ình Moore. Ch ng h n ta gán $S_2/0 = Q_1$, $S_3/0 = Q_2$, $S_4/0 = Q_3$, $S_5/0 = Q_4$, $S_1/1 = Q_5$, $S_2/1 = Q_6$, $S_3/1 = Q_7$, $S_4/1 = Q_8$.

B c 2: Thành l p b ng chuy n i tr ng thái cho h ình Moore.

V i b ng tr ng thái hình 2.12 và cách gán nh b c 1, ta l p c b ng chuy n tr ng thái cho h ình Moore (hình 2.14).

| Tr ng thái | α | β | γ | Ra |
|--------------|----------|---------|----------|----|
| $Q_1(S_2/0)$ | Q_8 | Q_1 | Q_8 | 0 |
| $Q_2(S_3/0)$ | Q_5 | Q_5 | Q_5 | 0 |
| $Q_3(S_4/0)$ | Q_7 | Q_3 | Q_1 | 0 |
| $Q_4(S_5/0)$ | Q_4 | Q_2 | Q_3 | 0 |
| $Q_5(S_1/1)$ | Q_6 | Q_3 | Q_2 | 1 |
| $Q_6(S_2/1)$ | Q_8 | Q_1 | Q_8 | 1 |
| $Q_7(S_3/1)$ | Q_5 | Q_5 | Q_5 | 1 |
| $Q_8(S_4/1)$ | Q_7 | Q_3 | Q_1 | 1 |

Hình 2.14. B ng tr ng thái Moore ng v i b ng hình 3.4



Hình 2.15. h ình Moore

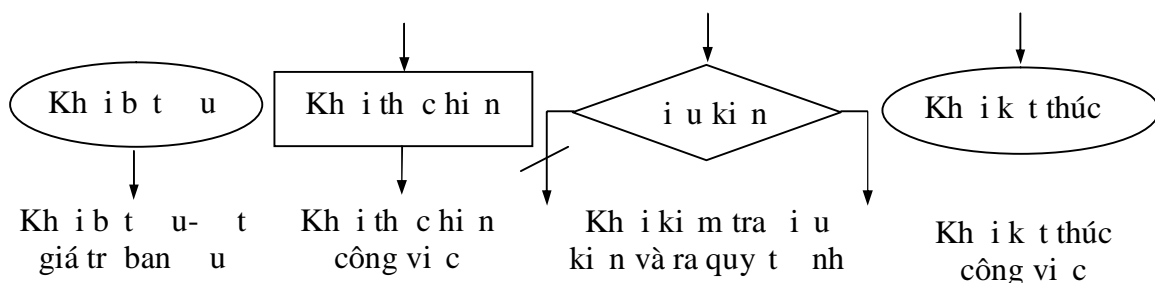
B c 3: D a vào b ng tr ng thái (hình 2.14), v c h ình Moore (hình 2.15).

T t nhiên n u có b ng tr ng thái c a h ình Moore, ta c ng d dàng thi t l p c b ng tr ng thái cho h ình Mealy b ng cách ghi thêm vào các ô chuy n tr ng thái c a b ng Moore các k t qu u ra t ng ng và b c t ra, sau ó tìm cách t i g i n b ng tr ng thái s nh n c b ng tr ng thái cho h ình Mealy.

Ta th y h ình Moore có s tr ng thái nhi u h n h ình Mealy, nh ng hàm ra c a h ình Moore n g i n h n c a h ình Mealy.

2.5.3. Ph ng pháp l u

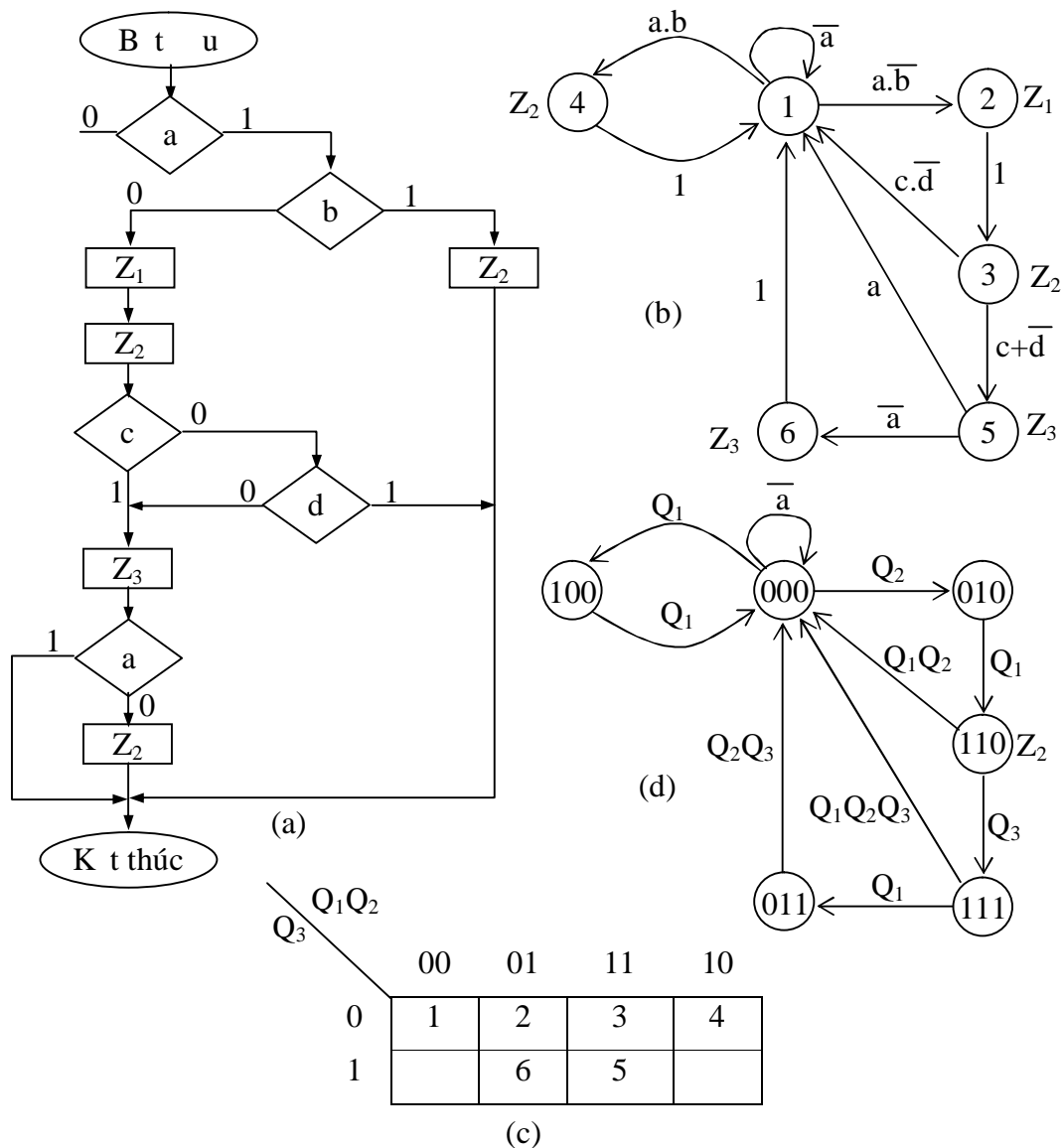
h ình thu t toán là cách mô t h th ng m t cách suy lu n tr c quan. Các kh i chính c a l u và các kh i c mô t h ình 2.16.



Hình 2.16: Ký hi u các kh i l u

Tập luật thu thập toán ta đã dùng chuyển thành hình trạng thái Mealy hoặc Moore và nó có thể thiết kế mạch trình tự.

Ví dụ, cho hình trạng thái hình 2.17a, hãy chuyển thành hình trạng thái Moore và viết phương trình mạch logic.



Hình 2.17. Ví dụ chuyển tập luật thành hình Moore

Việc chuyển tập luật thành hình Moore có thể chỉ ra qua các bước sau:

Bước 1: Kết hợp khi bắt đầu và khi kết thúc thành một trạng thái, như khi trạng thái kết thúc quay về trạng thái đầu.

Bước 2: Mỗi khi thiết bị chỉ ra là một trạng thái.

Bước 3: Xây dựng hình trạng thái Moore với các nhánh là các trạng thái, còn cung là các biến gây ra chuyển trạng thái.

B 4: Xây dựng hình nh phân v i các nh là các s nh phân h 2 và cung là các bi n mã hoá tr ng thái.

B 5: V i t ph ng trình m ch.

V i 5 b c nh trên ta có th phân tích quá trình th c hi n chuy n t l u ã cho hình 3.14a thành hình Moore nh hình 2.17b.

Tr ng thái 1: Là k t h p tr ng thái ã ban u và k t thúc ở lúc này có i u ki n u ra là a.

Tr ng thái 2: ng v i công vi c Z_1 , theo l u chuy n $1 \rightarrow 2$ c n $a\bar{b}$

Tr ng thái 3: ng v i công vi c Z_2 , theo l u chuy n $2 \rightarrow 3$ là liên t c.

Tr ng thái 4: ng v i công vi c Z_2 , theo nhánh l u $b=1$, chuy n $1 \rightarrow 4$ c n ab .

Tr ng thái 5: ng v i công vi c Z_3 , theo l u chuy n $3 \rightarrow 5$ c n $a + \bar{d}$.

Tr ng thái 6: ng v i công vi c Z_3 , theo nhánh l u $a=0$, chuy n $5 \rightarrow 6$ c n \bar{a}

Ngoài m t s tr ng thái có các cung quay v nh 1, t t c c th hi n trên hình 2.17b.

Ch n bi n u vào: Vì có 6 tr ng thái c n 3 bi n u vào Q_1, Q_2, Q_3 t h p các bi n $Q_1 Q_2 Q_3$ xác nh các tr ng thái nh hình 2.17c.

Gán giá tr nh phân cho hình Moore. Theo lu t chung, gi a hai tr ng thái k nhau ch c thay i m t bi n, còn vi c thay i tr ng thái v t c p các tr ng thái u ph i thay i c a t h p nhi u bi n (hình 2.17d).

V i t ph ng trình m ch. D a vào hình 2.17b và 2.17d. ta s vi t c hàm c a các bi n Q_1, Q_2, Q_3 và các tín hi u ra Z_1, Z_2, Z_3 m i tín hi u vào và tín hi u ra u là t h p c a t t c tr ng thái và các bi n kích thích trong l u hình 2.17a.

$$Q_1 = ab① + ② + \bar{c}d③ + ④ + ⑤$$

$$Q_2 = a\bar{b}① + \bar{c}d③ + a⑤ + ⑥$$

$$Q_3 = (c + \bar{d})③ + a⑤ + ⑥$$

$$Z_1 = ②; Z_2 = ③ + ④; Z_3 = ⑤ + ⑥.$$

2.6. T ng h p m ch trình t

Bài toán t ng h p m ch trình t là bài toán khó, h n n a t m t yêu c u ra l i có nhi u cách gi i quy t khác nhau. Do v y, v n chung ây là ph i d a vào m t ch tiêu t i u nào ó, ng th i tìm c l i gi i t i u thì ngoài các suy lu n toán h c logic ng i thi t k còn ph i t n d ng các kinh nghi m th c t r t a d ng và phong phú. Trong ph n này ta ch nêu m t s b c th c hi n chung và m t s ví d minh ho .

2.6.1. T ng h p theo ph ng pháp b ng tr ng thái

Trình t chung c a các b c nh sau:

1. Thành lập bảng chuyển trạng thái. Thắc mắc là vì cần tất các yêu cầu kỹ thuật thành ký hiệu cụ thể.
2. Thành lập bảng kích thích và bảng đầu ra.
3. Tìm hàm logic đầu ra và chuyển mạch.

Ta xét ví dụ: Hãy thiết kế mạch đếm chẵn lẻ (tín hiệu L) báo hiệu trạng thái làm việc không bình thường của mạch ghép hai chuyển mạch với yêu cầu sau: Nếu mạch không đúng theo trình tự chuyển mạch lần đầu tiên, chuyển mạch 2 sau và đúng theo trình tự chuyển mạch lần 2 đầu tiên, chuyển mạch 1 sau thì đèn L không sáng (làm việc bình thường). Nếu mạch không đúng hoặc sai trình tự trên thì đèn L sáng (báo làm việc không bình thường).

Cách làm:

Bảng 1: Thành lập bảng chuyển trạng thái.

Ta mã hoá trạng thái như sau:

X_1 - tín hiệu báo trạng thái của chuyển mạch 1.

X_2 - tín hiệu báo trạng thái của chuyển mạch 2.

Y - tín hiệu ra (tín hiệu kết quả của x_1, x_2).

Bảng chuyển trạng thái yêu cầu như hình 2.18. Trong bảng này các cột là các tổ hợp bit của tín hiệu vào x_1, x_2 , cột cuối cùng là Y , có 7 hàng tín hiệu 7 trạng thái của $(S_1 \div S_7)$.

| X_1X_2 | 00 | 01 | 11 | 10 | Y |
|------------|----|----|----|----|---|
| Trạng thái | | | | | |
| S_1 | ① | 4 | - | 2 | 0 |
| S_2 | 1 | - | 3 | ② | 0 |
| S_3 | - | 4 | ③ | 2 | 0 |
| S_4 | 5 | ④ | 6 | - | 1 |
| S_5 | ⑤ | 4 | - | 7 | 1 |
| S_6 | - | 4 | ⑥ | 7 | 1 |
| S_7 | 5 | - | 6 | ⑦ | 1 |

Hình 2.18

Thiết lập các bảng trạng thái (hình 2.18) ta thuận tiện xét tổ hợp bit của tín hiệu ra, có thể là:

Trạng thái S_1 (dòng 1): Tổ hợp bit vào $x_1x_2 = 00$, hệ thống chấp làm việc, tín hiệu ra $Y = 0$, đó là trạng thái bình thường.

Trạng thái S_2 (dòng 2): Lúc này $x_1x_2 = 10$, hệ thống làm việc vì chuyển mạch lần đầu tiên đúng yêu cầu, trạng thái bình thường và $Y = 0$.

Trạng thái S_3 (dòng 3): Từ trạng thái 2, chuyển sang $x_1x_2 = 11$ - đúng trình tự, đó là trạng thái bình thường và $Y = 0$.

Trạng thái S_4 (dòng 4): Từ trạng thái 3, chuyển sang $x_1x_2 = 01$ - sai trình tự, đó là trạng thái bình thường và $Y = 1$.

Trạng thái S_5 (dòng 5): Từ trạng thái 4, chuyển sang $x_1x_2 = 00$ - sai trình tự, đó là trạng thái bình thường và $Y = 1$.

Trạng thái S_6 (dòng 6): Từ trạng thái 5, hàm vi sai sai, chuyển sang trạng thái $x_1x_2 = 11$ - sai trình tự, là trạng thái nhàn và $Y = 1$.

Trạng thái S_7 (dòng 7): Từ trạng thái 6, hàm trạng thái sai, chuyển sang trạng thái $x_1x_2 = 10$ - vẫn trạng thái sai, là trạng thái nhàn và $Y = 1$.

Bảng lý giải tiếp theo ta tìm ra các trạng thái không nhàn và in vào các trạng thái vào bảng (hình 2.18).

Từ bảng trạng thái (hình 2.18) ta thấy hệ thống có 7 trạng thái khác nhau trong cùng một tổ hợp kết quả đầu ra cùng nhau, khi có cùng một đầu vào (trạng thái 1, 5 với $x_1x_2 = 00$, trạng thái 3, 6 với $x_1x_2 = 11$, trạng thái 2, 7 với $x_1x_2 = 10$). Phân loại các trạng thái mâu thuẫn đó, hệ thống phải đưa các biến nhị phân, đó chính là ý nghĩa của mạch logic trình tự.

Trước khi chọn các biến phân, ta tìm các rút gọn các hàng của bảng (hình 3.10). Nguyên tắc rút gọn là 2 hàng tiếp theo nhau thì rút gọn thành một hàng. Hai hàng coi là tiếp theo nhau khi có số trạng thái và kết quả đầu ra như nhau, hoặc có thể suy ra như nhau. Và như vậy từ bảng trạng thái hình 2.18 ta có thể rút gọn lại thành bảng trạng thái hình 2.19.

| x_1x_2 | 00 | 01 | 11 | 10 | Y |
|----------------------|----|----|----|----|---|
| Trạng thái | | | | | |
| S_1, S_2, S_3 | ① | 4 | ③ | ② | 0 |
| S_4, S_5, S_6, S_7 | ⑤ | ④ | ⑥ | ⑦ | 1 |

Hình 2.19

Bảng 2: Thành lập bảng kích thích và bảng tín hiệu đầu ra.

Với bảng chuyển trạng thái (hình 2.19), chỉ có 2 hàng, phân biệt 2 hàng chính của biến nhị phân. Ta chọn biến nhị phân đó là y.

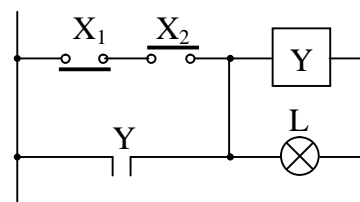
Với 3 biến x_1, x_2 và y, ta lập bảng trạng thái động bảng Karnaugh như hình 2.20, từ bảng trên hình 2.20 ta lập bảng kích thích (hình 2.21).

| x_1x_2 | 00 | 01 | 11 | 10 |
|----------|----|----|----|----|
| y | | | | |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |

Hình 2.20

| x_1x_2 | 00 | 01 | 11 | 10 |
|----------|----|----|----|----|
| y | | | | |
| 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

Hình 2.21



Hình 2.22

Bảng tín hiệu đầu ra lúc này quá ngắn gọn, không cần phải lập nữa, ta chọn luôn: $L = y$

Bảng 3: Viết phương trình hàm ra và vào.

Từ bảng hình 2.19 ta có: $Y = \overline{x_1}x_2 + y$

Từ 2 phương trình trên ta thiết kế mạch sơ đồ như hình 2.22.

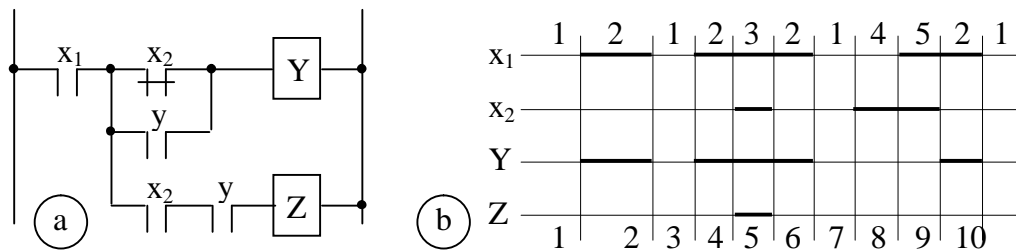
2.6.2. Thiết kế theo phương pháp hình Mealy hoặc Moore

Việc thiết kế các mạch trình tự bằng hình Mealy hoặc Moore có tiến hành theo các bước như sau:

- Bước 1: Xác định hình trạng thái
- Bước 2: Xác định số lượng phần tử (bộ nhớ, mạch logic). Mã hóa các trạng thái trong.
- Bước 3: Xác định hàm kích thích các mạch logic và hàm tín hiệu ra.

2.7. Ví dụ về mạch trình tự

Ví dụ 1: Giải thích có mạch trình tự có biểu diễn như hình 2.23a



Hình 2.23. Sơ đồ mạch trình tự và biểu đồ sóng

Để mô tả hoạt động của mạch, đây ta sẽ dùng biểu đồ sóng (hình 2.23b). Trên biểu đồ, chiều ngang biểu thị thời gian, chiều dọc thể hiện tín hiệu các đầu vào và đầu ra của mạch, nét đậm biểu thị tín hiệu có giá trị 1, còn nét mảnh biểu thị tín hiệu có giá trị 0. Từ biểu đồ ta thấy rằng, trạng thái $Z = 1$ chỉ xảy ra khi theo trình tự $x_1 = 1$, tiếp theo $x_2 = 1$. Nếu cho $x_2 = 1$ trước, sau đó cho $x_1 = 1$ thì cả Y và Z đều không thể bằng 1. Đây là tính chất phân biệt lâu dài của Y và Z , đó là:

$$Y.Z = 00, 10, 11.$$

Để mô tả hoạt động của mạch logic ta có thể dùng các phương pháp khác nhau bằng chuyển trạng thái, hình trạng thái, í

Chương 3

H I U KHI N LOGIC KH TRÌNH PLC

3.1. Khái niệm chung

PLC viết tắt của Programmable Logic Controller là thiết bị u khi n logic lập trình c (kh trình) cho phép thực hiện linh hoạt các thuật toán u khi n logic thông qua ngôn ngữ lập trình

PLC cấu thành từ hai phần chính:

+ Phần cứng: cấu tạo nên bởi vi xử lý, ROM, RAM

+ Phần mềm: Thực chất là ngôn ngữ để chuyển ngôn ngữ lập trình sang ngôn ngữ mà thiết bị u khi n logic có thể thực hiện được. Nó có nhiệm vụ chuyển các thiết bị u khi n logic thành các tín hiệu và thực hiện các phép tính logic theo công nghệ có thể thực hiện trên máy tính PC hoặc trên máy lập trình PG sau đó chuyển chương trình vào bộ nhớ RAM.

3.2. Cấu tạo PLC

Vì PLC là một hệ thống vi xử lý có chức năng chuyên dùng để thực hiện các nhiệm vụ logic, nó có nhiệm vụ chuyển các tín hiệu từ các thiết bị u khi n logic thành các tín hiệu mà thiết bị u khi n logic có thể thực hiện được. Nó có nhiệm vụ chuyển các tín hiệu từ các thiết bị u khi n logic thành các tín hiệu mà thiết bị u khi n logic có thể thực hiện được. Nó có nhiệm vụ chuyển các tín hiệu từ các thiết bị u khi n logic thành các tín hiệu mà thiết bị u khi n logic có thể thực hiện được.

3.2.1. Khái niệm u hành: Chương trình u hành

Khái niệm này dùng để chỉ chương trình u hành thực hiện và phân chia các bộ phận của chương trình u hành thành các phần khác nhau: vùng nhớ chương trình u hành, vùng nhớ biến trung gian, vùng nhớ cho tín hiệu vào và tín hiệu ra của chương trình giám sát kiểm tra hệ thống. Khái niệm này thường được dùng để chỉ bộ nhớ ROM.

3.2.2. Khái niệm chương trình:

Khái niệm này dùng để chỉ toàn bộ chương trình u hành của PLC và khái niệm này cũng bao gồm bộ nhớ RAM, chương trình được ghi vào hoàn toàn chương trình do người sử dụng, tạo ra và công nghệ. Trong quá trình thực hiện trong PLC thì chương trình này cũng gọi là lập trình câu lệnh đầu tiên của nó, khi đó nó cho vi xử lý thực hiện các phép toán logic để xử lý tín hiệu vào và gửi tín hiệu ra.

3.2.3. Khái niệm nh vào ra:

Đây là một phần của bộ nhớ RAM nó cũng là một phần của bộ nhớ RAM, nó làm việc theo nguyên tắc: Khi có tín hiệu tín hiệu vào thì khi bộ nhớ nh vào sẽ ghi giá trị của tín hiệu vào và khi có tín hiệu tín hiệu ra thì khi bộ nhớ nh ra sẽ gửi giá trị của tín hiệu ra.

qu x lý u ra không c a th ng ra u ra mà c ghi k t qu l i b ng nh ra và ch t i khi c l nh chuy n t i u ra thì tín hi u này m i c a ra ngoài.

3.2.4. C a (c ng) truy n thông:

C a này dùng truy n thông tin gi a PLC v i các thi t b bên ngoài nh : máy l p trình, máy tính cá nhân ho c n i m ng trong h PLC thông tin c truy n theo ki u n i ti p và quá trình truy n c chu n hoá qua cáp ghép n i RS232, RS485.

3.2.5. Kh i s h c:

Trong PLC ngoài vi c x lý các phép tính logic còn có thêm kh n ng x lý các phép tính s h c ho c so sánh tín hi u t ng t bi n i t ó t o nên c các hàm dùng trong i u khi n quá trình ch ng h n nh PID. Mu n v y trong PLC có 2 thanh ghi tích lu ký hi u ACCU1, ACCU2. ây là hai thanh ghi m i thanh 16 bit chia làm 2 byte: byte cao và byte th p. Quá trình th c hi n các phép tính s h c ho c so sánh c th c hi n trên hai thanh ghi này theo nguyên t c d li u u tiên s c chuy n vào ACCU1 khi ó d li u c trên ACCU1 c y sang ACCU2 và th c hi n các phép tính (+), (-), (x), (:) ho c so sánh c th c hi n qua câu l nh, k t qu phép tính c ghi l i trên ACCU1

3.2.6. Kh i vi x lý:

Làm nhi m v c ch ng trình trong kh i ch ng trình và ch ng trình ch d n làm gì thì vi x lý s i u ch nh các kh i khác làm theo ch c n ng. Ch ng h n: Lúc nào thì ghi d li u vào và x lý d li u này theo thu t toán nào và khi nào thì chuy n ra ngoài...

3.2.7. BUS:

Trong PLC nh ng thông tin c n ghép n i nh gi a kh i i u khi n trung tâm CPU, c s v i các kh i bên ngoài m r ng ho c gi a PLC v i b nh c ng EPROM c ng nh gi a vi x lý v i các b nh ROM, RAM c th c hi n b ng các dây n i, ó là c c u các h th ng bus. Ng i ta phân bi t h th ng bus thành 3 nhóm ch c n ng:

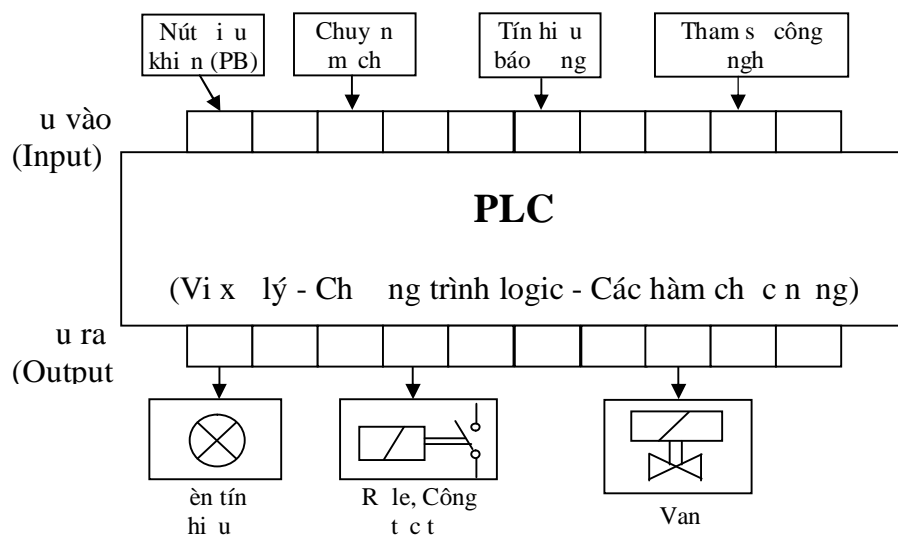
- + Bus s h i u: Tín hi u truy n trên ó theo 2 chi u
- + Bus a ch : Tín hi u ch truy n theo 1 chi u t CPU t i (ho c t thi t b i u khi n tr c ti p - DMAC), b nh ho c c a vào ra.
- + Bus các tín hi u i u khi n: G m m t s là tín hi u g i t CPU ra còn m t s l i là tín hi u g i t ngoài vào CPU.

3.3. S c u trúc PLC

Ph n c b n c a PLC là h vi x lý v i m t b x lý trung tâm (CPU), cùng v i các b nh , các thi t b ghép n i vào ra, biên d ch ch ng trình i u khi n. Bên ngoài PLC có m t b các u vào (input) và m t b các u ra (output) ghép n i v i các thi t b i u khi n, nh n các thông tin i u khi n và cho ra các l nh i u khi n h th ng. Các tín hi u vào, ra c a PLC ngoài d ng tín hi u lôgic (tín hi u s - digital) thì trong các PLC hi n nay th ng

có các tín hiệu vào và ra dạng tín hiệu (analog). Các tín hiệu vào dạng digital của PLC có thể là tín hiệu từ các nút nhấn, tín hiệu từ các công tắc hành trình, các cảm biến; tín hiệu từ các thiết bị báo động, v.v... Các tín hiệu từ tín hiệu analog vào PLC có thể là các tín hiệu từ các cảm biến công nghệ, v.v... Tín hiệu ra của PLC thường dùng để điều khiển các rơle và có thể là một tín hiệu từ tín hiệu nhấn khi cần thiết liên tục nào đó.

Sơ cấu trúc một PLC có dạng như hình 3.1



Hình 3.1

3.4. Nguyên lý làm việc

PLC làm việc theo nguyên tắc các chu kỳ lặp, mỗi chu kỳ lặp gọi là một vòng quét. Mỗi vòng quét có một lần nhận dữ liệu vào và đưa kết quả ra bên ngoài, khi hết vòng quét thì thời gian chuyển sang vòng quét tiếp theo và cứ tiếp tục như vậy.

Trong một vòng quét thì chia thành 4 bước:

- + Bước 1: Nhận dữ liệu vào ghi lại bộ nhớ vào.
- + Bước 2: Các chương trình nhấn trên các dữ liệu vào đã có (các hàm) xử lý theo chương trình các kết quả ghi lại bộ nhớ ra.
- + Bước 3: Thời gian truyền thông trong PLC hoặc các PLC với nhau cũng như thông tin qua lại với máy tính, thời gian kiểm tra liên lạc.
- + Bước 4: Gửi kết quả ra bên ngoài để thực hiện bên ngoài.

Như vậy thì thời gian một chu kỳ quét một lần một khoảng thời gian, thời gian này càng nhỏ càng tốt, như vậy thì thu được các xử lý các máy móc và thời gian các thiết bị do nhà chế tạo. Vì vậy thì chương trình nhấn nên lập sao cho càng ngắn càng tốt.

3.5. *Ưu nhược điểm*

3.5.1. *Ưu điểm*:

+ Với môi trường khi cần cấu lập bộ PLC có ưu điểm nổi bật là hoàn toàn chỉ cần kỹ thuật viên lập trình phần mềm mà không cần phải thay đổi phần cứng vì tất cả các thiết bị đều được sử dụng trong hệ thống qua chương trình phần mềm và chúng được ghép nối với nhau thông qua phần mềm, chính vì vậy làm cho quá trình cài đặt thay đổi hoàn toàn và sửa chữa rất dễ dàng, chỉ cần thay đổi phần mềm mà không cần thay đổi phần cứng. Do đó ưu điểm là tất cả mà các hệ thống logic truyền thống không có được.

+ Với hệ thống dùng PLC vẫn có thể tiến hành hiệu chỉnh sửa chữa nhanh chóng và dễ dàng vì cấu trúc là dạng mô-đun chuyên công nghệ không biến dạng, vì vậy nó cho phép hiệu chỉnh được tất cả các thông số hiệu chỉnh là tất cả.

3.5.2. *Nhược điểm*:

+ Do hệ thống dùng PLC rất thu gọn nên vì các hiệu chỉnh thay đổi thông số công nghệ thay đổi hàm hiệu chỉnh do đó người sử dụng phải am hiểu và khi thay đổi chương trình không đúng thì dễ làm rơi lỗi dây chuyền vì vậy gây khó khăn cho người quản lý.

+ Do công suất đầu ra của PLC nhỏ, thường chỉ có $I \leq 500 \text{ mA}$ vì vậy phải ghép với thiết bị bên ngoài có công suất lớn thì người sử dụng các thiết bị bên ngoài trung gian như các relay hoặc công tắc.

+ Do điện áp sử dụng trên PLC thấp vì vậy mô-đun an toàn phải có biện pháp cách ly với điện áp lưới.

+ Thiết bị lập trình chương trình hệ thống bắt buộc phải có máy tính hoặc máy lập trình đi kèm với các cấp chuẩn hóa.

3.6. *Ứng dụng*:

Với PLC có các ứng dụng như sau:

+ Với các dây chuyền sản xuất công nghệ chế biến và bảo vệ thực phẩm, công nghệ cán kéo thép, công nghệ sản xuất xi măng và các dây chuyền sản xuất công nghệ khác. Trong các hệ thống dùng các PLC có truyền động hệ thống.

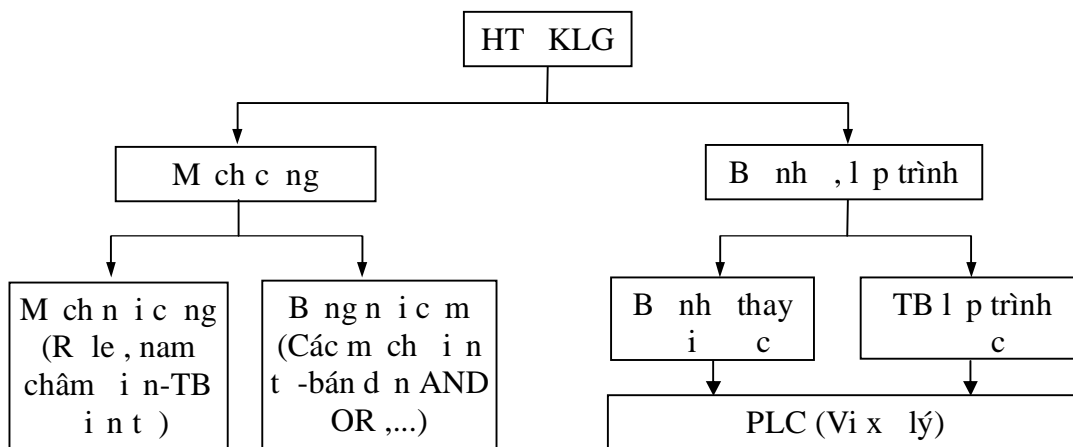
+ Trên thiết bị của máy tự động dùng PLC thiết bị nhận thông tin theo công nghệ và quy trình các chế độ làm việc một cách có hiệu quả công nghệ có mặt tự động nhận phát hiện phay, bào gọt... và các hệ thống quá trình khi ngừng công suất.

3.7. *Trình tự thiết kế hệ thống logic ứng dụng PLC*

Tổ chức môi trường hệ thống logic dùng PLC ta phải qua các bước:

- + Xu t phát t yêu c u công ngh ta ph i tính toán thi t k ra hàm i u khi n logic.
- + Xu t phát t công ngh và hàm logic ã có ta ph i xác nh c các sensor t o nên các bi n u vào.
- + T các sensor ã ch n ta ti n hành chu n hoá tín hi u n u c n (tín hi u sensor ch a phù h p v i tín hi u c a vào PLC)
- + Ti n hành l p trình và ch n lo i PLC áp ng cho h . Sau khi ch ng trình vào PLC ta ph i ti n hành ch y th b ng cách t o tín hi u gi .
- + Khi có ch ng trình i u khi n thì chúng ta s n i các sensor và các i t ng vào PLC, ti n hành ch y th trên thi t b th c, trong quá trình ch y th ta s hi u ch nh tính các thông s công ngh .
- + Ghi ch ng trình ã c hi u ch nh vào EPROM (v i lo i PLC có c a c m EPROM l u ch ng trình)

Các h th ng i u khi n logic có th c phân theo s sau (hình 3.2):



Hình 3.2

Chương 4

LẬP TRÌNH CHO PLC S7 - 200

4.1. Sơ cấu trúc của PLC

4.1.1. Cấu hình chung

S7-200 là thiết bị xử lý logic khi trình của hãng Siemens (CHLB Đức), có cấu trúc theo kiểu module và có các module mở rộng. Các module này được sử dụng cho nhiều ứng dụng lập trình khác nhau. S7-200 được cấu tạo theo các module trong đó module chính là module chứa CPU có địa chỉ vào ra, còn các module mở rộng có thể là tăng tốc độ xử lý và địa chỉ hoàn toàn phụ thuộc vào kiểu module và vị trí của module trong hệ vì vậy không ghi địa chỉ.

Thành phần cơ bản của S7 - 200 là khi vi xử lý CPU 212 hoặc CPU 214. Về hình thức bên ngoài, sẽ khác nhau của hai loại CPU này như nhận biết được như sau vào/ra nguồn cung cấp.

- CPU 212 có 8 cổng vào, 6 cổng ra và có khả năng mở rộng thêm bằng 2 module mở rộng.

- CPU 214 có 14 cổng vào và 10 cổng ra và có khả năng mở rộng thêm bằng 7 module mở rộng.

S7-200 có nhiều loại module mở rộng khác nhau.

CPU 212 bao gồm

- + 512 từ nhớ là 1KB lưu chương trình thu nhỏ nên nhớ/ghi nhớ và không bị mất dữ liệu khi có giao diện với EPROM.

- + Có 8 cổng vào logic và 6 cổng ra logic.

- + Có thể ghép nối thêm 2 module mở rộng số cổng vào/ra, bao gồm cả module tăng tốc (analog).

- + Tổng số cổng logic vào/ra tối đa là 64 cổng vào và 64 cổng ra.

- + Có 64 bộ đếm thời gian

- + Có 64 bộ đếm, chia làm hai loại: loại bộ đếm đếm lên và loại bộ đếm đếm xuống.

- + 368 bit nhớ bit, sử dụng làm các bit trạng thái hoặc các bit đặt/reset.

- + Có các chế độ ngừng và xử lý tín hiệu ngừng khác nhau bao gồm ngừng truyền thông, ngừng theo số lên hoặc số xuống, ngừng theo thời gian và ngừng báo hiệu cao/bộ đếm tối đa cao (2 KHz).

+ B ản nh không b m t d li u trong kho ng th i gian 50h khi PLC b m t ngu n nu i.

CPU 214 bao g m

+ 2048 t n (4KB) l u ch ng tr nh thu c m i n b nh c/ghi c và không b m t d li u nh có giao di n v i EEPROM.

+ Có 14 c ng vào và 10 c ng ra logic

+ Có 7 module m r ng th m c ng vào/ra bao g m c module analog.

+ T ng s c ng vào ra c i là 64 c ng vào và 64 c ng ra.

+ Có 128 b t o th i gian tr .

+ Có 128 b m chia làm hai lo i ch m t i n và v a m t i n v a m l i.

+ Có 688 b t nh c b i t dùng thông báo tr ng th i và t ch làm vi c.

+ Các ch ng t và x lý ng t g m ng t truy n thông, ng t theo s n lên ho c xu ng, ng t theo th i gian, ng t c a b m t c cao và ng t truy n xung.

+ Có 3 b m t c cao v i nh p 2 KHz và 7 KHz.

+ Có 2 b phát xung nhanh cho d y xung ki u PTO ho c ki u PWM

+ Có 2 b i u ch nh t ng t

+Toàn b vùng nh không b m t gi li u trong kho ng th i gian 190h khi PLC b m t ngu n nu i.

4.1.2. Mô t các ền báo và công t c

- Trên S7-200 ền báo có các lo i:

+ ền (SF): ền này sáng khi PLC có h ng h c.

+ ền xanh (RUN): ền này ch nh PLC ang ch làm vi c và th c hi n ch ng tr nh c n p trong máy.

+ ền vàng (STOP) ền này ch nh r ng PLC ang ch d ng. D ng ch ng tr nh ang th c hi n l i (Có th do ng i v n hành cho ngh ho c trong ch ng tr nh g p l nh STOP ho c trong PLC có h ng h c).

+ ền xanh I x.x: ền xanh c ng vào ch nh tr ng th i t c th i c a c ng Ix.x ($x.x = 0.0 \div 1.5$). ền này báo hi u tr ng th i c a tín hi u theo giá tr logic c a c ng.

+ ền xanh Qy.y: ền này báo hi u tr ng th i t c th i c a c ng Qy.y ($y.y = 0.0 \div 1.1$), ền này báo hi u tr ng th i c a tín hi u theo giá tr logic c ng.

- Công t c ch n ch làm vi c cho PLC:

Công tắc chọn chế độ làm việc nằm phía trên, bên cạnh các công tắc S7 - 200 có 3 vị trí cho phép chọn các chế độ làm việc khác nhau cho PLC

+ RUN: Cho phép PLC thực hiện chương trình trong bộ nhớ. PLC S7 - 200 sẵn sàng khi chuyển từ chế độ STOP sang chế độ RUN ngay khi công tắc chuyển sang RUN. Do đó nên quan sát trạng thái thực tế của PLC theo đèn báo.

+ STOP: Cho phép PLC dừng công việc thực hiện chương trình đang chạy và chuyển sang chế độ STOP. Khi chuyển sang chế độ STOP PLC cho phép khởi động lại chương trình hoặc nạp mới chương trình mới.

+ TERM: Cho phép máy lập trình tải chương trình mới vào bộ nhớ của PLC hoặc xóa chương trình hiện tại.

- Chế độ chọn chế độ: Nút chọn chế độ công tắc nằm bên cạnh các công tắc. Khi nhấn nút, công tắc quay 270°.

4.1.3. Cấu trúc bộ nhớ

- Phân chia bộ nhớ: Bộ nhớ của S7 - 200 được chia thành 4 vùng vật lý có nhiệm vụ duy trì dữ liệu trong bộ nhớ khi mất nguồn. Bộ nhớ của S7 - 200 có tính năng cao, tốc độ ghi và đọc trong toàn bộ vùng lưu trữ các phần mềm của bộ nhớ ký hiệu bằng SM (special memory) có thể truy cập được.

+ Vùng chương trình: Là bộ nhớ dùng để lưu trữ các lệnh chương trình khi cần có thể chuyển vào bộ nhớ lập trình, bộ nhớ EPROM và trong quá trình làm việc thực hiện chương trình yêu cầu.

+ Vùng tham số: Là bộ nhớ dùng để lưu trữ các tham số: thời gian, địa chỉ... các thông tin này có thể cài đặt do nhà chế tạo hoặc do người sử dụng khi mua mã khóa chương trình.

+ Vùng dữ liệu: Vùng này dùng để lưu trữ các dữ liệu của chương trình bao gồm: các kết quả của các phép tính, hằng số... cần nhớ trong chương trình, bộ nhớ truy cập ngẫu nhiên... Vùng dữ liệu là bộ nhớ ngẫu nhiên, nó có thể truy cập theo từng bit, từng byte, từng từ, từng cặp và dùng để lưu trữ dữ liệu cho các thuật toán, các hàm truy cập, lập trình, các hàm dịch chuyển, xoay vòng thanh ghi, con trỏ... ..

Vùng dữ liệu lại được chia ra thành nhiều miền khác nhau vì các công dụng khác nhau. Chúng ta ký hiệu bằng các chữ cái Latinh để riêng cho công dụng của chúng như sau:

V (Variable memory): Vùng nhớ trung gian.

I (Input image register): Vùng nhớ đầu vào.

O (Output image register): Vùng nhớ đầu ra.

M (Internal memory bits): Vùng nhớ lưu giữ.

SM (Special memory): Vùng nhớ đặc biệt.

truy nhập vào vùng này ta phải ghi địa chỉ theo nguyên tắc:

+ Truy nhập theo bit: (Tên miền) + (Địa chỉ byte) + (.) + chỉ số bit.

Ví dụ: V150.4: chỉ số bit 4 của byte 150 thuộc miền V

+ Truy nhập theo byte: (Tên miền) + (B) + (Địa chỉ của byte trong miền)

Ví dụ: VB150 chỉ số byte 150 thuộc miền V

+ Truy nhập theo từ: (Tên miền) + (W) + (Địa chỉ byte cao cấp nhất trong miền)

Ví dụ: VW150: chỉ số từ gồm hai byte 150 và 151 thuộc miền V, trong đó byte 150 có vai trò là byte cao trong từ.

| | | | | | | | | | | | | | | | | |
|-------|------------------|----|----|----|----|----|---|---|-------------------|---|---|---|---|---|---|---|
| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| VW150 | VB150 (byte cao) | | | | | | | | VB151 (byte thấp) | | | | | | | |

+ Truy nhập theo từ kép: (Tên miền) + (D) + (Địa chỉ byte cao cấp nhất trong miền)

VD: VD150: chỉ số từ kép gồm 4 byte 150, 151, 152, 153 thuộc miền V, trong đó byte 150 có vai trò là byte cao và byte 153 là byte thấp trong từ kép.

+ Vùng I/O: Là vùng nhớ đặc biệt ký hiệu và sử dụng riêng cho các bộ nhớ, các rơ le thời gian, các bộ đếm và tín hiệu vào ra từ và từ thanh ghi. Tên gọi và ký hiệu ta có các vùng và các chức năng như sau:

. C + n: Vùng nhớ dành riêng cho các bộ đếm có địa chỉ n. VD: C0, C63..

. AIW + chỉ số byte cao: Vùng nhớ dành cho các bộ đếm đếm vào liên tục

. AQW + chỉ số byte thấp: Vùng nhớ dành cho các bộ đếm đếm ra liên tục.

. T + n: Vùng nhớ dành riêng cho các rơ le thời gian có chỉ số là n.

. AC + chỉ số thanh ghi: Các thanh ghi dành riêng cho vùng này (AC0 - AC3)

. HSC + chỉ số thanh ghi: Thanh ghi dành riêng cho các bộ đếm tốc độ cao (HSC0 - HSC2)

4.1.4. Mở rộng ngõ vào ra (Module mở rộng)

Mở rộng khả năng I/O khi cần ta phải mở rộng ngõ vào ra của PLC bằng cách ghép nối thêm vào nó các module mở rộng về phía bên phải của CPU làm thành một mô-đun xích. Địa chỉ của các vị trí của module được xác định bằng ký hiệu vào/ra và vị trí của module trong mô-đun xích, bao gồm các module có cùng ký hiệu. Ví dụ như một module đếm ra không thể gán địa chỉ của một module đếm vào, cũng như một module từ tính không thể có địa chỉ của một module số và ngược lại.

CPU 212 có thể mở rộng thêm 2 module và CPU 214 có thể mở rộng 4 module

VD: Cách kết nối cho các module mở rộng trên CPU 214

| CPU 214 | | Module | Module | Module | Module 8 ra |
|---------|------|-------------|--------|-------------------------|-------------|
| | | 4 vào /4 ra | 8 vào | 3 vào analog/1ra analog | |
| I0.0 | Q0.0 | I2.0 | I3.0 | AIW0 | Q3.0 |
| I0.1 | Q0.1 | I2.1 | I3.1 | AIW2 | Q3.1 |
| I0.2 | Q0.2 | I2.2 | I3.2 | AIW4 | Q3.2 |
| I0.3 | Q0.3 | I2.3 | I3.3 | | Q3.3 |
| I0.4 | Q0.4 | | I3.4 | AQW0 | Q3.4 |
| 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 | | | | | |

4.1.5. Thắc mắc khi thực hiện chương trình

Với S7 - 200 việc thực hiện chương trình cũng tương tự như các PLC khác đó là thực hiện các bước theo chu trình lập, mỗi vòng lập công việc là một vòng quét và làm việc theo cách lập trình vì vậy thực hiện chương trình thì gian trễ và xử lý phải như hình thì gian trễ vòng quét và một vòng quét được chia làm 4 giai đoạn:

+ Giai đoạn 1: Nhận thông tin từ các cửa vào và nhận tín hiệu vào.

+ Giai đoạn 2: Thực hiện trong chương trình điều kiện và lấy thông tin có sẵn bên ngoài vào, xử lý theo câu lệnh và chuyển kết quả ra bên ngoài. Trong quá trình này câu lệnh được thực hiện từ câu lệnh đầu tiên đến câu lệnh cuối cùng và khi gặp lệnh kết thúc (MEND) thì quay lại và chờ lệnh tiếp theo.

+ Giai đoạn 3: Truy cập thông tin và kiểm tra lại.

+ Giai đoạn 4: Chuyển kết quả đã có bên ngoài để gửi tín hiệu vào bên ngoài.

4.1.6. Cấu trúc chương trình

Trong quá trình lập trình i u khi n v i các ch ng tr ãnh n gi n ta t ãn hành l p tr ãnh t u dd n cu i ch có m t ch ng tr ãnh, nh ng v i các h th ng ph c t p n u l p tr ãnh nh trên r t ph c t p. n gi n ta t ãn hành l p tr ãnh cho t ng m ng công ngh và công tr ãnh c a c dây chuy n s c liên k t b i r t nhi u m ng v i nhau, ó g i là l p tr ãnh có c u tr ãnh.

Ch ng tr ãnh i u khi n c a các m ng n u không ph thu c i u khi n ch ph thu c vào ch ng tr ãnh g i là ch ng tr ãnh con còn n u các m ng v a ph thu c vào ch ng tr ãnh v a ph thu c vào quá tr ãnh i u khi n c g i là ch ng tr ãnh ng t.

Các ch ng tr ãnh cho S7 - 200 ph i có c u tr ãnh bao g m ch ng tr ãnh chính và sau ó n các ch ng tr ãnh con và các ch ng tr ãnh ng t c ch ra sau ây:

+ Ch ng tr ãnh chính ph i c l p tr u c và k t thúc b ng l nh MEND

+ Sau ch ng tr ãnh chính vi t các ch ng tr ãnh con và k t thúc ch ng tr ãnh con ph i có l nh RET. Khi g p l nh RET thì vi x lý quay v ch ng tr ãnh chính và các ch ng tr ãnh con ph i c ký hi u và có a ch SBR.

+ Ch ng tr ãnh ng t c vi t sau ch ng tr ãnh con khi có l nh ng t trong ch ng tr ãnh chính (i u khi n ng t) thì ch ng tr ãnh s vào ch ng tr ãnh ng t khi làm vi c xong ch ng tr ãnh ng t g p l nh RETI thì nó t ng quay v ch ng tr ãnh chính t i i m ng t và m i ch ng tr ãnh ng t u ph i có l nh (INT + a ch)

Main Program

í

MEND

Th c hi n trong m t vòng quét

SBR 0 Ch ng tr ãnh con th nh t

í

RET

SBR n Ch ng tr ãnh con th n + 1

í

RET

INT 0 Ch ̣ng tṛnh x ̣ lý ng ̣ t th ̣ nh ̣ t
 í
 RETI

Th ̣ c hi ̣ n khi có tín hi ̣ u báo ng ̣ t

INT n Ch ̣ng tṛnh x ̣ lý ng ̣ t th ̣ n + 1
 í
 RETI

4.2 Ngôn ng ̣ l ̣ p tṛnh c ̣ a S7-200

4.2.1. Gi ̣ i thi ̣ u chung

Ph ̣ ng pháp li ̣ t kê l ̣ nh (*Statement List* vi ̣ t t ̣ t là STL) và ph ̣ ng pháp s ̣ các kh ̣ i ch ̣ c n ̣ ng (FBD). N ̣ u ch ̣ng tṛnh l ̣ p theo FBD thì thi ̣ t b ̣ l ̣ p tṛnh có th ̣ chuy ̣ n sang d ̣ ng LAD ho ̣ c STL t ̣ ng ̣ ng. Ng ̣ c l ̣ i không ph ̣ i m ̣ i ch ̣ng tṛnh vi ̣ t d ̣ ng STL ho ̣ c LAD c ̣ ng có th ̣ chuy ̣ n sang d ̣ ng FBD. N ̣ u ch ̣ng tṛnh l ̣ p theo LAD thì thi ̣ t b ̣ l ̣ p tṛnh có th ̣ chuy ̣ n sang d ̣ ng STL t ̣ ng ̣ ng, nh ̣ ng không ph ̣ i m ̣ i ch ̣ng tṛnh vi ̣ t d ̣ ng STL c ̣ ng có th ̣ chuy ̣ n sang d ̣ ng LAD. Trong ph ̣ n này chúng ta nghiên c ̣ u hai ph ̣ ng pháp l ̣ p tṛnh là LAD và STL, chúng ̣ u có s ̣ n trong ngôn ng ̣ l ̣ p tṛnh STEP7-Micro/DOS và STEP7-Micro/WIN, còn l ̣ p tṛnh ki ̣ u FBD ch ̣ có trong STEP7-Micro/WIN. đ ̣ d ̣ ng làm quen v ̣ i v ̣ i các thành ph ̣ n c ̣ b n c ̣ a LAD và STL ta c ̣ n n ̣ m các nh ̣ ngh ̣ a c ̣ b n sau ̣ y:

nh ̣ ngh ̣ a v ̣ LAD: LAD là m ̣ t ngôn ng ̣ l ̣ p tṛnh b ̣ ng ̣ ho ̣ . Nh ̣ ng thành ph ̣ n c ̣ b n dùng trong LAD t ̣ ng ̣ ng v ̣ i các thành ph ̣ n c ̣ a b ̣ ng ̣ i u khi n b ̣ ng r ̣ le. Trong ch ̣ng tṛnh LAD các ph ̣ n t ̣ c ̣ b n dùng ̣ bi ̣ u di ̣ n các l ̣ nh logic nh ̣ sau:

- *Ti ̣ p ̣ i m*: là bi ̣ u t ̣ ng (symbol) mô t ̣ các ti ̣ p ̣ i m c ̣ a r ̣ le. Các ti ̣ p ̣ i m ó có th ̣ là th ̣ ng m ̣ (ó|ó) ho ̣ c th ̣ ng óng (ó/|ó).

- *Cu ̣ n d ̣ y (coil)*: là bi ̣ u t ̣ ng (ó()ó) mô t ̣ cu ̣ n d ̣ y r ̣ le

- *H ̣ p (box)*: là bi ̣ u t ̣ ng mô t ̣ các hàm khác nhau, nó làm vi ̣ c khi có dòng i ̣ n ch ̣ y ̣ n h ̣ p. Nh ̣ ng d ̣ ng hàm th ̣ ng ̣ c bi ̣ u di ̣ n b ̣ ng h ̣ p là các b ̣ th i gian (*Timer*), các b ̣ m (*Counter*) và các hàm toán h ̣ c. Cu ̣ n d ̣ y và h ̣ p ph ̣ i c ̣ m c ̣ úng chi ̣ u dòng i ̣ n.

- *M ̣ ng LAD*: là ̣ ng n ̣ i các ph ̣ n t ̣ thành m ̣ t m ̣ ch hoàn thi ̣ n, i ̣ t ̣ ng ngu ̣ n bên trái sang ̣ ng ngu ̣ n bên ph ̣ i. ̣ ng ngu ̣ n bên trái là d ̣ y nóng, ̣ ng ngu ̣ n bên ph ̣ i là d ̣ y trung hoà (neutral) hay là ̣ ng tr ̣ v ̣ ngu ̣ n cung c ̣ p (̣ ng ngu ̣ n bên ph ̣ i th ̣ ng không c ̣ th ̣ hi ̣ n khi dùng ch ̣ng tṛnh ti ̣ n d ̣ ng STEP7-

Micro/DOS hoặc STEP7-Micro/WIN). Dòng i n ch y t trái qua các t i p i m óng n các c u n dây hoặc các h p tr v bên ph i ngu n.

nh ngh a v STL: Ph ãng pháp l i t kê l ãnh (STL) là ph ãng pháp th h i n ch ãng trình d i d ãng t p h p các câu l ãnh. M i câu l ãnh trong ch ãng trình, k c ãnh ãng l ãnh h ãnh th c b i u d i n m t ch c n ãng c a PLC.

nh ngh a v ãng n x p logic (logic stack):

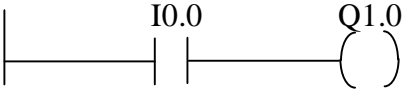
| | |
|----|--|
| S0 | Stack 0-bit u tiên hay bit trên cùng c a ãng n x p |
| S1 | Stack 1-bit th hai c a ãng n x p |
| S2 | Stack 2-bit th ba c a ãng n x p |
| S3 | Stack 3-bit th t c a ãng n x p |
| S4 | Stack 4-bit th n m c a ãng n x p |
| S5 | Stack 5-bit th sáu c a ãng n x p |
| S6 | Stack 6-bit th b y c a ãng n x p |
| S7 | Stack 7-bit th tám c a ãng n x p |
| S8 | Stack 8-bit th chín c a ãng n x p |

Hình 4.1

t o r a m t ch ãng trình d ãng STL, ãng i l p trình c n ph i h i u rõ ph ãng th c s d ãng 9 bit ãng n x p logic c a S7-200. **Ng ãng n x p logic là m t kh i g m 9 bit ch ãng lên nhau**. T t c các thu t toán liên quan ãng ãng n x p u ch làm v i c v i bit u tiên hoặc v i bit u và bit th hai c a ãng n x p. Giá tr logic m i u có th c g i (hoặc c n i thêm) vào ãng n x p. Khi ph i h p hai bit u tiên c a ãng n x p, thì ãng n x p s c kéo lên m t bit. ãng ãng n x p và tên c a t ãng bit trong ãng n x p c b i u d i n trên hình 4.1.

V i d v ladder logic và statement list:

Hình 4.4 mô t v i c th c h i n l ãnh LD (v i t t t t Load trong t i ãng Anh) a giá tr logic c a t i p i m I0.0 vào trong ãng n x p theo cách b i u d i n c a LAD và STL:

| LAD | STL |
|---|-------------------|
|  | LD I0.0 = Q1.0 |

Hình 4.2

4.2.2.B ãng tóm t t m t s l ãnh c b n c a S7-200

H l ãnh c a S7-200 c chia làm 3 nhóm:

- Các l ãnh mà khi th c h i n thì làm v i c c l p không ph thu c vào giá tr logic c a ãng n x p.

- Các lệnh chuyển các bit ưu tiên sang ng x p có giá trị logic bằng 1.
- Các nhãn lệnh ảnh hưởng vị trí trong tập lệnh.

Trong các bảng lệnh còn một số thay đổi đáng kể so với bảng lệnh cũ khi lệnh chuyển các bit.

C hai phương pháp LAD và STL sử dụng ký hiệu I để chỉ vị trí các bit chuyển (immediately), tức là giá trị được chuyển vào thanh ghi ngay lập tức (immediately), tức là giá trị được chuyển vào thanh ghi ngay lập tức khi lệnh được thực hiện. Các bit này không phải chờ đợi để được chuyển vào thanh ghi khi thanh ghi đang bận. Điều này khác với lệnh chuyển các bit từ thanh ghi này sang thanh ghi khác (LDR) khi thanh ghi đang bận thì lệnh sẽ chờ đợi.

Bảng 4.1. Một số lệnh của S7-200 thuộc nhóm lệnh chuyển các bit

| Tên lệnh | Mô tả |
|-------------|---|
| = n | Giá trị của bit ưu tiên ng x p được sao chép sang bit n ch d n trong l nh. |
| = I n | Giá trị của bit ưu tiên ng x p được sao chép trực tiếp sang bit n ch d n trong l nh ngay khi l nh được thực hiện. |
| AND n | Thực hiện toán tử AND giữa giá trị logic của bit ưu tiên ng x p và giá trị logic của bit n ch d n trong l nh. Kết quả ghi lại vào bit ưu tiên c a ng n x p. |
| ALD | Thực hiện toán tử AND giữa giá trị logic của bit ưu tiên ng x p và giá trị logic của bit hai ng n x p. Kết quả ghi lại vào bit ưu tiên ng n x p. Các giá trị còn lại trong ng n x p được kéo lên một bit. |
| AND n | Thực hiện toán tử AND giữa giá trị logic của bit ưu tiên ng n x p và giá trị logic của bit n ch d n trong l nh. Kết quả ghi lại vào bit ưu tiên c a ng n x p. |
| CTU Cxx,PV | Khởi động bộ đếm lên theo số lần của tín hiệu vào. Bộ đếm có thể đặt lại trạng thái ban đầu (reset) nếu vào R của bộ đếm kích (có mức logic 1). |
| CTUD Cxx,PV | Khởi động bộ đếm lên theo số lần của tín hiệu vào và bộ đếm lùi theo số lần của tín hiệu vào hai. Bộ đếm có thể đặt lại nếu vào R của bộ đếm kích (có mức logic 1). |
| LD n | Nạp giá trị logic của bit n ch d n trong l nh vào bit ưu tiên c a ng n x p. Các giá trị trong ng n x p được xóa xuống một bit. |

| | |
|--------------|---|
| LDN n | N p giá tr logic ngh ch o c a i m n ch d n trong l nh vào bit u tiên c a ng n x p. Các giá tr trong ng n x p c y xu ng m t bit. |
| LDW<=n1, n2 | Bit u tiên trong ng n x p nh n giá tr logic 1 n u n i dung hai t n1 và n2 tho m n n1 ≤ n2. |
| LDW = n1, n2 | Bit u tiên trong ng n x p nh n giá tr logic 1 n u n i dung hai t n1 và n2 tho m n n1 = n2. |
| LDW>=n1, n2 | Bit u tiên trong ng n x p nh n giá tr logic 1 n u n i dung hai t n1 và n2 tho m n n1 ≥ n2. |
| LPP | Kéo n i dung ng n x p lên m t bit. Giá tr m i c a bit trên là giá tr c c a bit d i, s u ng n x p g i m i m t bit (giá tr c a bit u tiên b y ra kh i ng n x p - xoá). |
| LRD | Sao chép giá tr c a bit th hai vào bit u tiên c a ng n x p. Các giá tr còn l i t bit th hai tr i c g i nguyên v trí. |
| MEND | K t thúc ph n ch ng trình chính trong m t vòng quét. |
| NOT | o giá tr logic c a bit u tiên ng n x p. |
| O n | Th c hi n toán t ho c (OR) g i a giá tr logic c a bit u tiên ng n x p v i giá tr logic c a i m n ch d n trong l nh. K t qu c ghi l i vào bit u tiên c a ng n x p. |
| OI n | Th c hi n t c th i toán t ho c (OR) g i a giá tr logic c a bit u tiên ng n x p v i giá tr logic c a i m n ch d n trong l nh. K t qu c ghi l i vào bit u tiên c a ng n x p. |
| OLD | Th c hi n toán t ho c (OR) g i a giá tr logic c a bit u tiên ng n x p v i giá tr logic c a bit th hai ng n x p. K t qu c ghi l i vào bit u tiên c a ng n x p. Các giá tr còn l i trong ng n x p c kéo lên m t bit. |
| ON n | Th c hi n toán t ho c (OR) g i a giá tr logic c a bit u tiên ng n x p v i giá tr logic ngh ch o c a i m n ch d n trong l nh. K t qu c ghi l i vào bit u tiên c a ng n x p. |
| RET | L nh thoát kh i ch ng trình con và tr i u khi n v ch ng trình chính ã g i nó. |
| RETI | L nh thoát kh i ch ng trình x lý ng t (<i>interrupt</i>) và tr i u khi n v ch ng trình chính. |

4.2.3. Cú pháp hình ảnh của S7-200

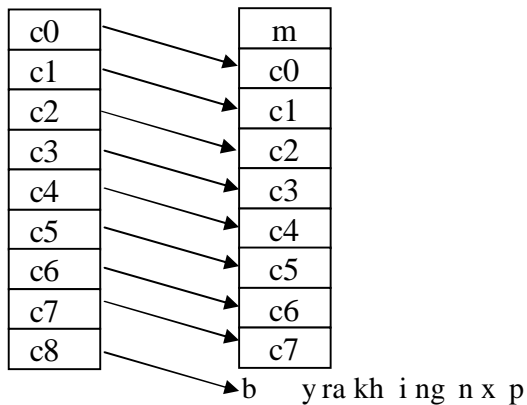
Mạch dù S7-200 có một khối riêng biệt các lệnh thể hiện các thuật toán của các Boolean song song có một vài các kiểu lệnh khác nhau

4.2.3.1- Lệnh vào/ra

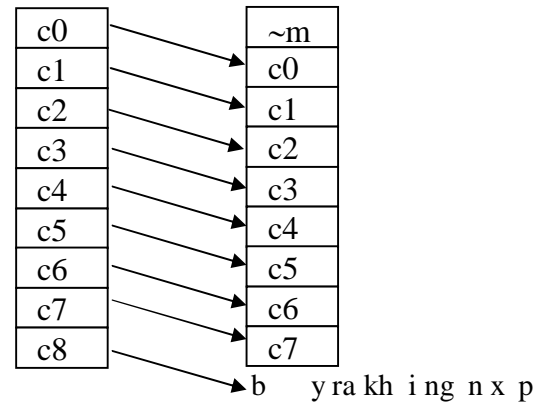
* Load (LD) và Load Not (LDN)

Lệnh LD nạp giá trị logic của một tiếp điểm vào bit ưu tiên của ng xếp (hình 4.3), các giá trị còn lại trong ng xếp bị xóa đi.

Lệnh LDN nạp giá trị logic nghịch của một tiếp điểm vào trong bit ưu tiên của ng xếp (hình 4.4), các giá trị còn lại trong ng xếp bị xóa đi.



Hình 4.3. Trạng thái ng xếp trước và sau khi thực hiện lệnh LD



Hình 4.4. Trạng thái ng xếp trước và sau khi thực hiện lệnh LDN

Các dạng khác nhau của lệnh LD, LDN cho LAD

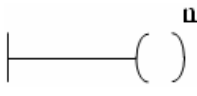
| LAD | Mô tả | Toán học |
|-----|---|----------------------------------|
| | Tiếp điểm thường mở cổng n u n = 1 | n: I, Q, M, SM, T, C, V (bit) |
| | Tiếp điểm thường đóng cổng n u n = 1 | |

Các dạng khác nhau của lệnh LD, LDN cho STL

| Lệnh | Mô tả | Toán học |
|-------|---|----------------------------------|
| LD n | Tiếp điểm thường mở cổng n u n = 1 | n: I, Q, M, SM, T, C, V (bit) |
| LDN n | Tiếp điểm thường đóng cổng n u n = 1 | |

* **OUTPUT (=)** : Lệnh sao chép nội dung của bit ưu tiên trong ngõ vào bit (i m) về kênh trong lệnh. Nội dung của ngõ vào không thay đổi.

Mô tả lệnh Õ=ö b ng LAD nh sau:

| LAD | Mô tả | Toán học |
|---|--|-------------------------------|
|  | Cu n dây u ra tr ng thái kích thích, có dòng i u khi n i qua n u n = 1 | n: I, Q, M, SM, T, C, V (bit) |

Mô tả lệnh Õ=ö b ng STL nh sau:

| Lệnh | Mô tả | Toán học |
|------|--|-------------------------------|
| = n | Cu n dây u ra tr ng thái kích thích, có dòng i u khi n i qua n u n = 1 | n: I, Q, M, SM, T, C, V (bit) |

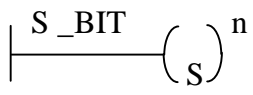
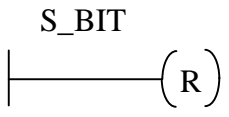
4.3.3.2. Lệnh ghi / xoá giá trị cho từng bit

SET (S) / RESET (R)

Đây là các lệnh dùng để đặt các bit (từng bit) về trạng thái 1. Trong LAD, logic i u khi n dòng i n óng ho c ng t các cu n dây u ra. Khi dòng i u khi n n các cu n dây thì các cu n dây óng ho c m các t i p i m (ho c m t d ã y các t i p i m).

Trong STL, thì đây là lệnh truyền trạng thái bit ưu tiên của ngõ vào các bit về từng bit. Nếu bit này bằng 1, các lệnh S và R sẽ đặt trạng thái từng bit về 1 (giới hạn từ 1-255). Nội dung của ngõ vào không thay đổi.

Mô tả lệnh S và R b ng LAD nh sau:

| LAD | Mô tả | Toán học |
|---|---|-----------------------------------|
|  | Đặt trạng thái của các bit i m k t S_BIT | S_BIT: I, Q, M, SM, T, C, V (bit) |
|  | Ng t m t m ng g m n các t i p i m k t S_BIT. Nếu S_BIT l i ch vào Timer ho c Counter thì l nh s xoá bit u ra c a Timer/Counter ó. | n: IB, QB, MB, SMB, VB (byte) |

Mô tả lệnh S và R b ng STL nh sau:

| Lệnh | Mô tả | Toán học |
|------------|--|-----------------------------------|
| S S_BIT, n | Đặt giá trị logic 1 cho m t m ng g m n bit k t a ch S_BIT. | S_BIT: I, Q, M, SM, T, C, V (bit) |

| | | |
|------------|---|-------------------------------|
| R S_BIT, n | Xoá m t m ng g m n bit k t a ch S_BIT. N u S_BIT l i ch vào Timer ho c Counter thì l nh s xoá bit u ra c a Timer/Counter ó. | n: IB, QB, MB, SMB, VB (byte) |
|------------|---|-------------------------------|

Ví d : Mô t vi c th c hi n l nh 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 |

4.23.3- Các l nh logic i s Boolean

Các l nh ti p i m i s Boolean cho phép t o l p c các m ch logic (không có nh). Trong LAD các l nh này c bi u di n thông qua c u trúc m ch (m c n i ti p hay song song các ti p i m th ng óng và th ng m) Trong STL các l nh c vi t t nh sau A (AND), O (OR), AN (AND NOT), ON (OR NOT). Giá tr ng n x p thay i ph thu c vào t ng l nh.

| L nh | Mô t | Toán h ng |
|--------------|---|-------------------------------|
| A n O n | L nh th c hi n toán t \wedge (AND) và \vee (OR) gi a giá tr logic c a ti p i m n và giá tr bit u tiên c a ng n x p. K t qu l i c ghi vào bit u tiên c a ng n x p | n: I, Q, M, (bit) SM, T, C, V |
| AN n ON n | L nh th c hi n toán t \wedge (AND) và \vee (OR) gi a giá tr logic ngh ch o c a ti p i m n và giá tr bit u tiên c a ng n x p. K t qu l i c ghi vào bit u tiên c a ng n x p | |

Ngoài nh ng l nh làm vi c tr c ti p v i ti p i m, S7-200 còn có 5 l nh c bi t bi u di n các phép tính i s Boolean cho các bit trong ng n x p, c g i là các l nh **stack logic**. ó là các l nh ALD (And load), OLD (Or load), LPS (Logic push), LRD (Logic read), LPP (Logic pop). L nh **stack logic** c dùng t h p, sao ch p ho c xoá các m nh logic. B ng sau tóm t t các l nh **stack logic** trong STL:

| L nh | Mô t | Toán h ng |
|------|--|-----------|
| ALD | L nh t h p giá trị logic c a bit u tiên và bit th hai c a ng n x p b ng phép tính \wedge (AND). K t qu c ghi l i vào bít u tiên c a ng n x p, giá tr còn l i c a ng n x p c kéo lên m t bit. | Không có |
| OLD | L nh t h p giá trị logic c a bit u tiên và bit th hai c a ng n x p b ng phép tính \vee (OR). K t qu c ghi l i vào bít u tiên c a ng n x p, giá tr còn l i c a ng n x p c kéo lên m t bit. | Không có |
| LPS | L nh Logic Push th c hi n sao ch p giá tr logic c a bit u tiên ng n x p vào bit th hai. Giá tr còn l i trong ng n x p b y xu ng 1 bit. Bit cu i cùng b y ra kh i ng n x p. | Không có |
| LRD | L nh th c hi n sao ch p giá tr logic c a bit th hai ng n x p vào bit u tiên. Giá tr còn l i trong ng n x p c gi nguyên v trí. | Không có |
| LPP | L nh kéo ng n x p lên 1 bit. Giá tr c a bit sau c chuy n cho bit tr c. Giá tr c c a bit u tiên b y ra ngoài. | Không có |

A và O: Tác ng c a l nh **A** và **O** vào ng n x p:

L nh A

Tr c

| |
|----|
| c0 |
| c1 |
| c2 |
| c3 |
| c4 |
| c5 |
| c6 |
| c7 |
| c8 |

Sau

| |
|----|
| m |
| c1 |
| c2 |
| c3 |
| c4 |
| c5 |
| c6 |
| c7 |
| c8 |

$$m = c0 \wedge n$$

L nh O

Tr c

| |
|----|
| c0 |
| c1 |
| c2 |
| c3 |
| c4 |
| c5 |
| c6 |
| c7 |
| c8 |

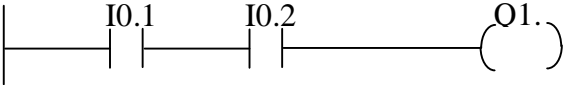
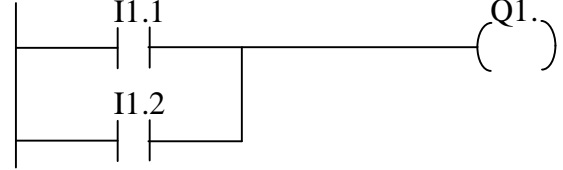
Sau

| |
|----|
| m |
| c1 |
| c2 |
| c3 |
| c4 |
| c5 |
| c6 |
| c7 |
| c8 |

$$m = c0 \vee n$$

Hình 4.5. Tr ng thái ng n x p tr c và sau khi th c hi n l nh A và O

Mô t l nh A và O d ng 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 dụng của lệnh **ALD** và **OLD** vào ng n x p:

L nh ALD

Tr c

| |
|----|
| c0 |
| c1 |
| c2 |
| c3 |
| c4 |
| c5 |
| c6 |
| c7 |
| c8 |

| |
|----|
| m |
| c2 |
| c3 |
| c4 |
| c5 |
| c6 |
| c7 |
| c8 |
| |

$$m = c0 \wedge c1$$

L nh OLD

Tr c

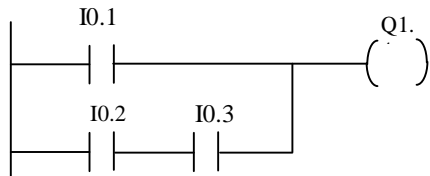
| |
|----|
| c0 |
| c1 |
| c2 |
| c3 |
| c4 |
| c5 |
| c6 |
| c7 |
| c8 |

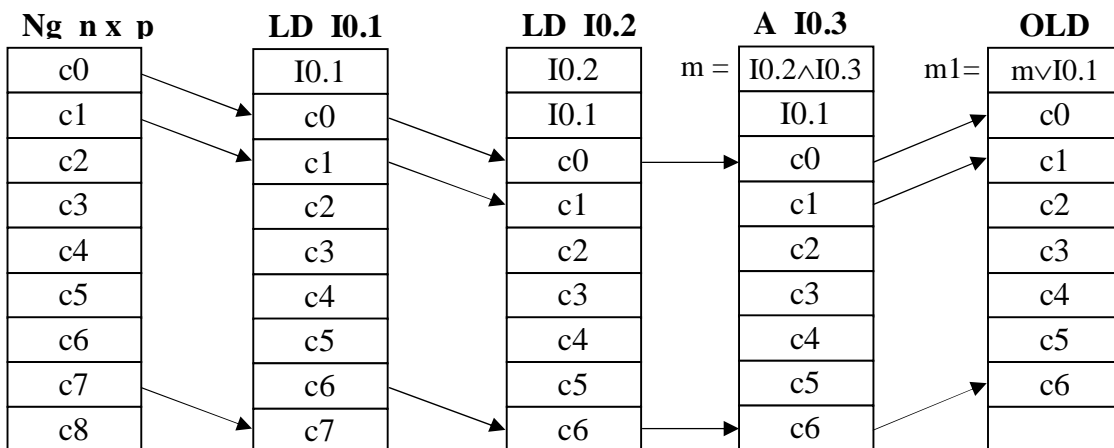
| |
|----|
| m |
| c2 |
| c3 |
| c4 |
| c5 |
| c6 |
| c7 |
| c8 |
| |

$$m = c0 \vee c1$$

Hình 4.6. Tr ng thái ng n x p tr c và sau khi th c hi nh ALD và OLD

Ví d : Mô t l nh OLD d ng LAD và STL:

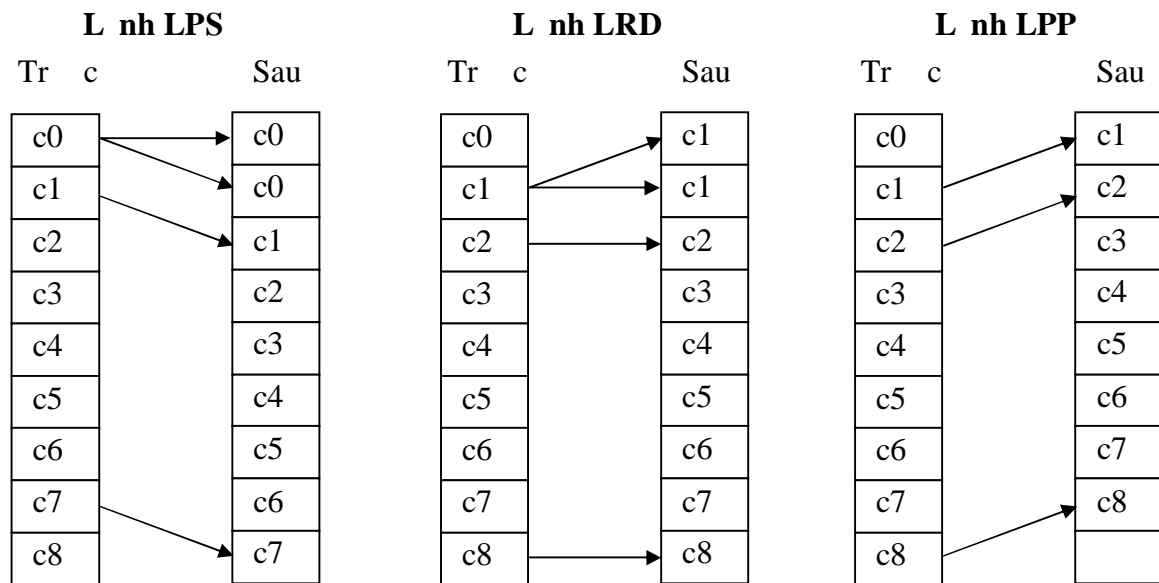
| LAD | STL |
|---|---|
|  | LD I0.1 LD I0.2 A I0.3 OLD = Q1.4 |



Hình 4.7. Tr ng thái ng n x p tr c và sau khi th c hi nh các l nh b ng trên

LPS, LRD, LPP

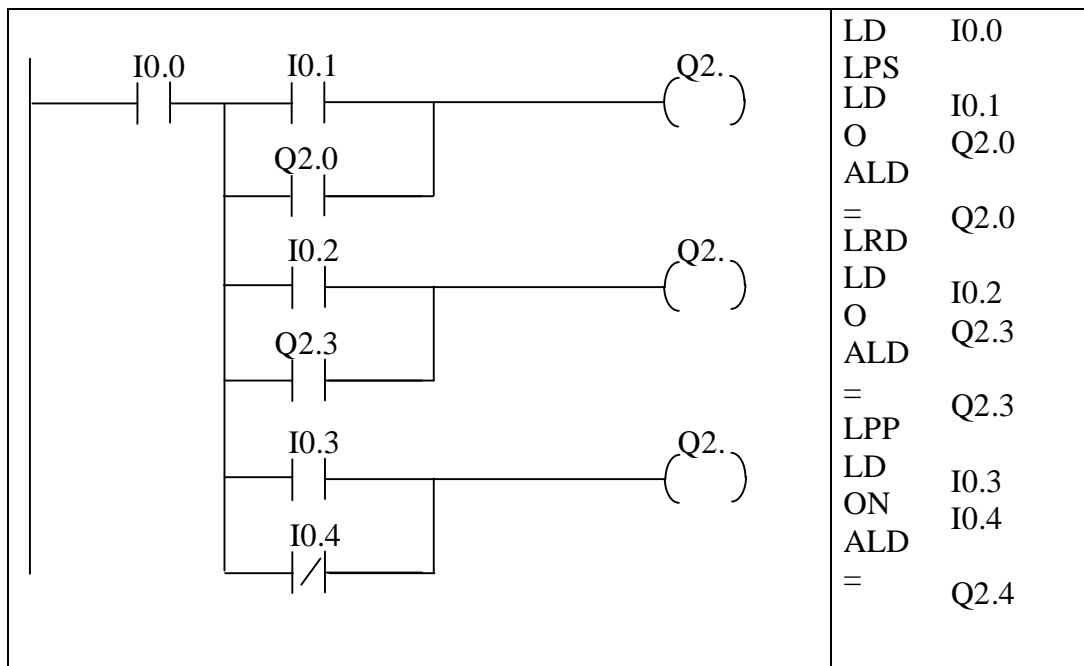
Các lệnh này thực hiện sao chép, thay đổi nội dung bit ưu tiên hoặc bit thanh ghi của ngõ vào và các mô tả như sau:



Hình 4.8. Trạng thái ngõ vào trước và sau khi thực hiện các lệnh LPS, LRD, LPP

Ví dụ: Các lệnh Logic stack dùng LAD và STL:

| LAD | STL |
|-----|--|
| | <pre>LD I0.0 LD I0.1 LD I0.2 A I0.3 OLD ALD = Q1.0</pre> |



4.2.3.4. Các lệnh so sánh và di chuyển n i dung ô nh :

| STL | Mô t | Toán h ng |
|---|--|--|
| LDW>= n1 n2 AW>= n1 n2 OW>= n1 n2 | L nh th c hi n phép tính logic Load, And ho c Or gi a giá tr 1 v i n i dung c a nh ng n x p khi n i dung 2 t n1, n2 tho m n n1 ≥ n2. | n1, n2 (t): VW, T, C, IW, QW, MW, SMW.. |
| MOVW IN OUT | L nh sao chép n i dung t n IN sang t n OUT. | IN, OUT (t n): VW, T, C, IW, QW |

4.2.3.5. Các lệnh làm vi c v i Timer:

Timer là b t o th i gian tr gi a tín hi u vào và tín hi u ra nên trong i u khi n th ng c g i là khâu tr . N u ký hi u tín hi u (logic) vào là $x(t)$ và th i gian tr là τ thì tín hi u u ra c a Timer là $x(t-\tau)$. Trong S7-200 có hai lo i Timer khác nhau:

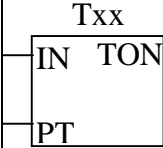
- Timer t o th i gian tr không có nh (*On-Delay Timer*), ký hi u là TON.
- Timer t o th i gian tr có nh (*Retentive On-Delay Timer*), ký hi u là TONR.

Hai Timer này phân bi t nhau b i ph n ng c a chúng i v i tín hi u vào. C hai lo i u b t u t o th i gian tr t th i i m có s n lên c a tín hi u vào. Nh ng TON s t Reset khi u vào có m c logic 0, còn TONR thì không t Reset khi m t tín hi u vào. TON

cùng một thời gian trong một khoảng thời gian, còn với TONR thì thời gian có thể khác nhau. Trong phần này ta chỉ nghiên cứu loại Timer TON.

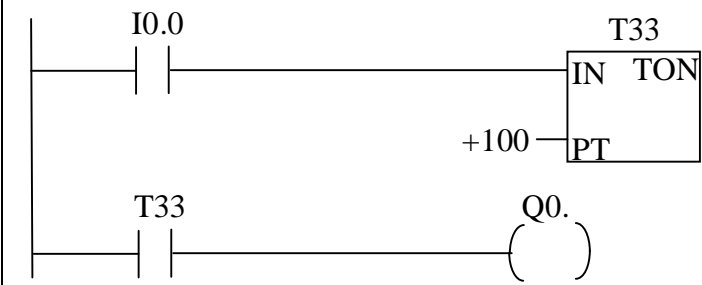
| Lệnh | phân giải | Giá trị cực đại | CPU 212 | CPU 214 |
|------|-----------|-----------------|-----------|------------------------|
| TON | 1 ms | 32,767 s | T32 | T32, T96 |
| | 10 ms | 327,67 s | T33 ÷ T36 | T33 ÷ T36, T97 ÷ T100 |
| | 100 ms | 3276,7 s | T37 ÷ T63 | T37 ÷ T63, T101 ÷ T127 |
| TONR | 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:

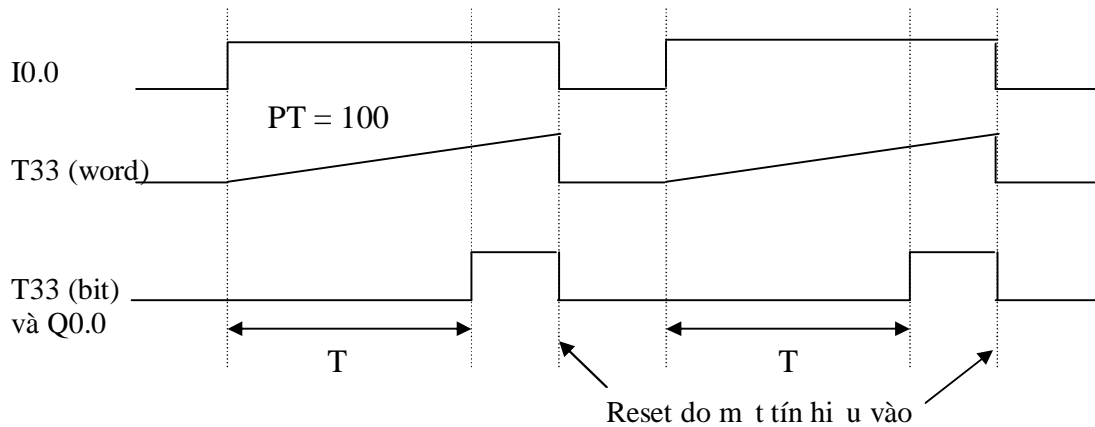
| LAD | STL | Mô tả | Toán học |
|---|----------------|---|--|
|  | TON Txx, +n | Khai báo Timer số hi u xx khi u TON thời gian tr tính t khi u vào IN cực kích (có mức 1). Nếu giá trị mức thời gian h n ho c b ng giá trị tr c PT thì T-bit có giá trị logic b ng 1. Có thể Reset Timer khi u TON b ng 1 nh Reset ho c b ng giá trị logic 0 u vào IN. | Txx(word): CPU 212: 32÷63 CPU 214: 32÷63 và 96÷127 PT(word): VW,T,C,IW,... n = 1÷32767 (s nguyên) |

Thời gian $T = PT \times$ phân giải, ví dụ với T33 có phân giải là 10 ms và $PT=100$ thì thời gian $T = 10 \times 100 = 1000 \text{ ms} = 1 \text{ s}$.

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

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

Giãn thời gian t_{ng} ng:



Hình 4.9

4.2.3.6. Các lệnh làm việc với Counter:

Counter là bộ đếm thời gian đếm xung. S7-200 có hai loại: bộ đếm thời gian (CTU) và bộ đếm thời gian/lùi (CTUD).

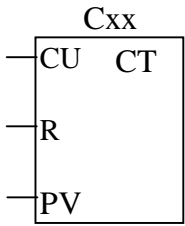
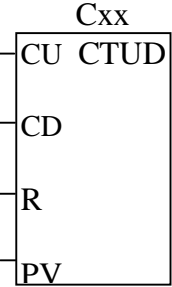
Bộ đếm thời gian đếm xung vào, tức là mỗi lần thay đổi trạng thái logic từ 0 lên 1 sẽ đếm xung. Số xung đếm, được ghi vào thanh ghi 2 byte của bộ đếm, gọi là thanh ghi C-word.

Nội dung của C-word, gọi là giá trị thực tế của bộ đếm, luôn được so sánh với giá trị đặt trước của bộ đếm ký hiệu là PV. Khi giá trị đếm thực tế bằng hoặc lớn hơn giá trị đặt trước thì bộ đếm báo ra ngoài bằng cách đặt giá trị logic 1 vào bit của nó, gọi là C-bit. Trạng thái 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 vào khi xóa thời gian. Khi phát ban đầu (reset) cho bộ đếm, ký hiệu bằng chữ cái R trong LAD, hay quy định là trạng thái bit ưu tiên của ngõ ra trong STL. Bộ đếm sẽ reset khi tín hiệu xóa này có mức 1 hoặc khi lệnh R (reset) được thực hiện với C-bit. Khi bộ đếm reset thì C-word và C-bit đều về giá trị 0.

Bộ đếm thời gian/lùi CTUD thực hiện đếm khi gặp xung vào đếm, ký hiệu là CU trong LAD hoặc bit thứ 3 ngõ ra trong STL, và đếm lùi khi gặp xung vào đếm lùi, ký hiệu là CD trong LAD hoặc bit thứ 2 ngõ ra trong STL. Việc xóa 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 học |
|---|--------------|--|--|
|  | CTU Cxx, +n | Khai báo bộ đếm tăng theo số n lên các tín hiệu vào cổng CU số hiệu xx khi bộ CTU. Khi giá trị đếm đạt tới C-word của Cxx lần hiện hoặc bộ giá trị trừ của PV, C-bit (Cxx) có giá trị logic bằng 1. Bộ đếm được reset khi 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 (s nguyên) |
|  | CTUD Cxx, +n | Khai báo bộ đếm tăng/lùi, đếm tăng theo số n lên các tín hiệu vào cổng CU và đếm lùi theo số n lên các tín hiệu vào cổng CD. Khi giá trị đếm đạt tới C-word của Cxx lần hiện hoặc bộ giá trị trừ của PV, C-bit (Cxx) có giá trị logic bằng 1. Bộ đếm được reset khi vào R có giá trị logic 1. Bộ đếm ngừng đếm tăng 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 (s nguyên) |

Ký hiệu Cxx của bộ đếm tăng hoặc giảm là cách hình thức của C-word và của C-bit. Mặc dù cùng cách hình thức, song C-word và C-bit vẫn được phân biệt với nhau như khi sử dụng làm vị trí ký hiệu hay ký hiệu bit (bit). Ví dụ :

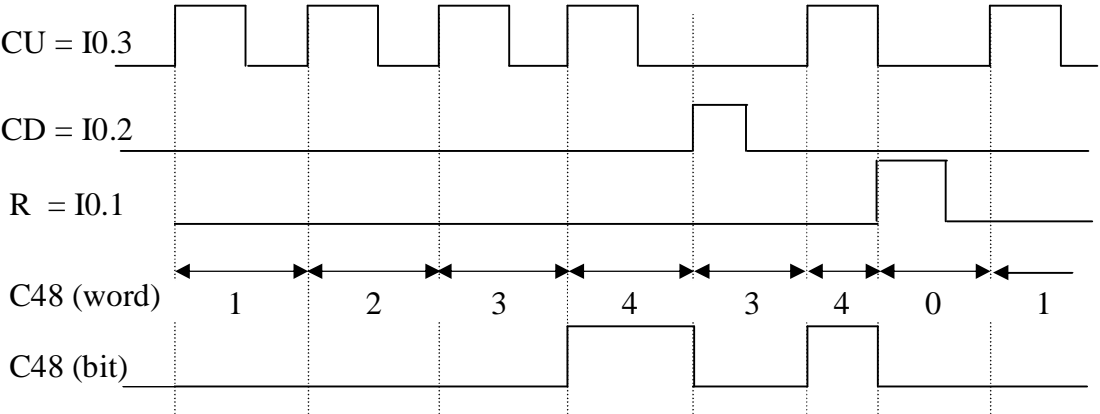
LD C48 // Lệnh làm vị trí C-bit của bộ đếm C48.

LDW>= C48 // Lệnh làm vị trí 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 |
|-----|--|
| | <pre> Network 1 I0.3 LD I0.2 LD I0.1 LD C48, +4 CTUD Network 2 C48 LD Q1.0 = </pre> |

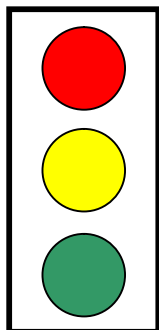
Gi n th i gian t ng ng:



Hình 4.10

4.3. M t s ví d ng d ng S7-200

4.3.1- Ch ng tr ình ì u khi n òn ng



Hình 4-11

Qui nh các tì p ì m ì u khi n òn tr c chính:

- Xanh: Q0.0 v ì th ì gian 10 s
- Vàng: Q0.1 v ì th ì gian 1 s
- : Q0.2 v ì th ì gian 7 s

Qui nh các tì p ì m ì u khi n òn tr c ph :

- Xanh: Q0.5 v ì th ì gian 7 s
- Vàng: Q0.6 v ì th ì gian 1 s
- : Q0.7 v ì th ì gian 10 s

Yêu c u c a bài toán là kh ng ch t ng h th ng òn m t ngã t v ì tr c chính và ph : òn xanh tr c chính và òn tr c ph cùng sáng trong 10 s, tì p sau òn vàng c hai tr c ng cùng sáng trong 1 s, tì p n a òn tr c chính và òn xanh tr c ph cùng sáng trong 7 s và tì p sau òn vàng c hai tr c ng l ì cùng sáng trong 1 s - k t thúc 1 chu k và h th ng t ng ho t ng l p l ì. l p ch ng tr ình kh ng ch chúng ta có nhi u cách khác nhau. Sau ây chúng ta s tì n hành l p tr ình theo m t cách:

S d ng 3 Timer k ì u TON là T37, T38, T39, T40 u có phân gi ì 100 ms kh ng ch th ì gian, ch ng tr ình d ng STL nh sau:

NETWORK1

LDN I0.1

LD I0.0

O M0.0

ALD

= M0.0

NETWORK2

LD M0.0

AN T40

TON T37 , +100

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

| | | | |
|----------|------|----------|-----------|
| NETWORK3 | = | Q0.5 | |
| LD | M0.0 | | |
| AN | T37 | NETWORK8 | |
| AN | Q0.1 | LD | M0.0 |
| AN | Q0.2 | A | T39 |
| = | Q0.0 | TON | T40 , +10 |
| = | Q0.7 | | |

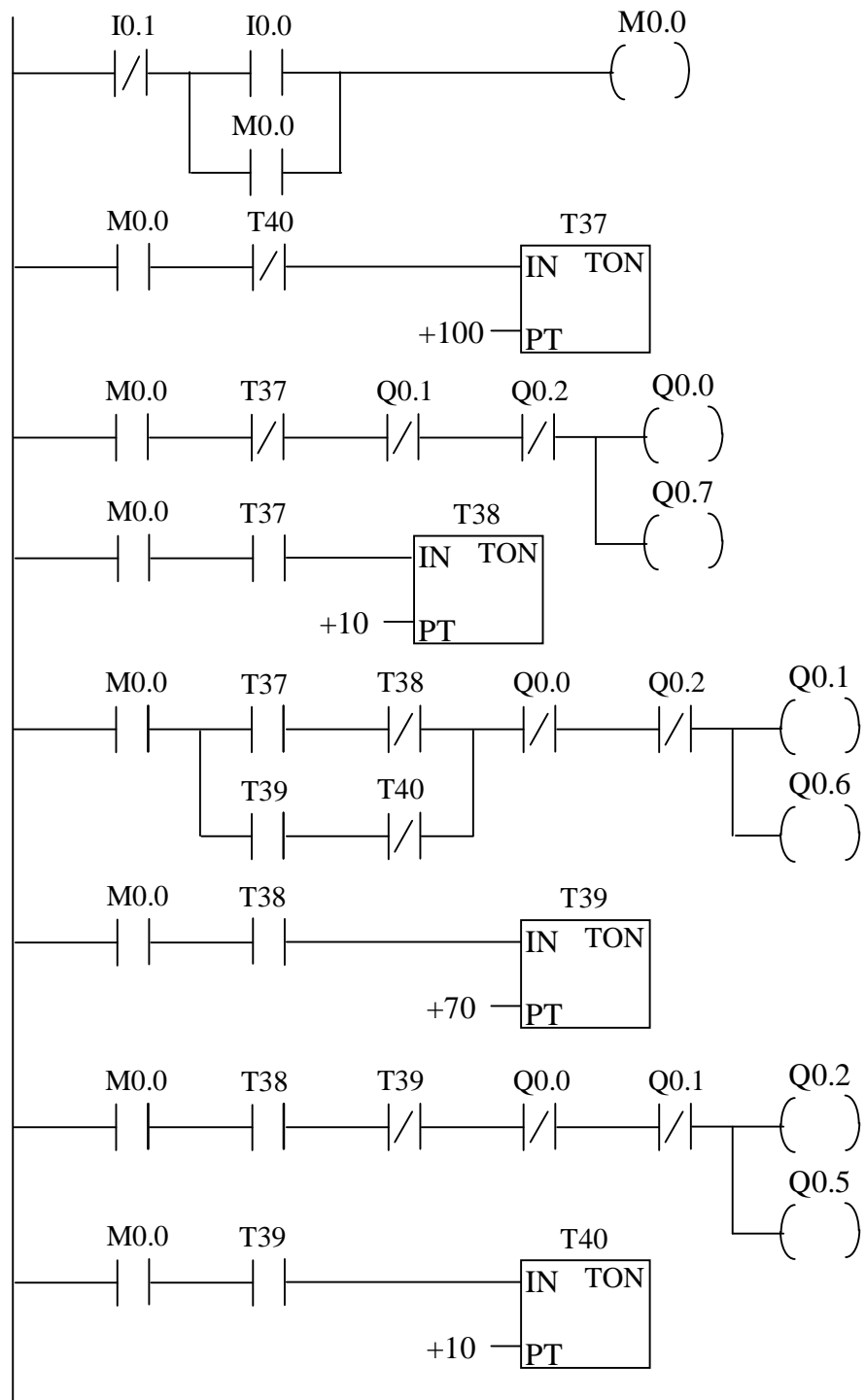
NETWORK4

| | |
|-----|-----------|
| 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 |

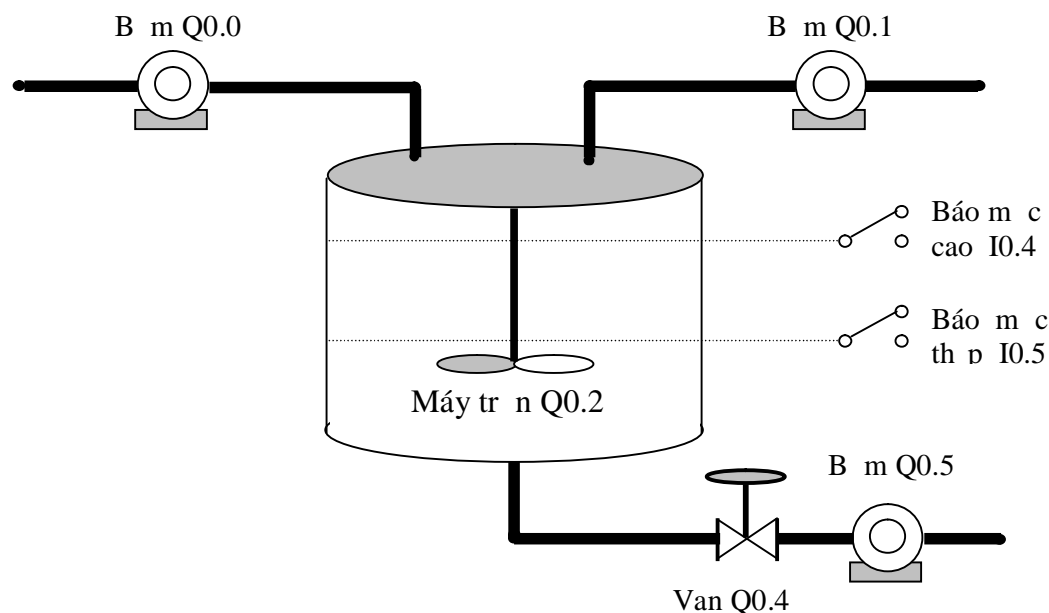
Chương trình viết trong LAD



Hình 4.12. S LAD của bài toán đèn

4.3.2. Chương trình i u khi n máy tr n s n

Hình 4.13 là s m t bình tr n t o các màu s n khác nhau. Trong s cho th y có hai ng ng a hai lo i s màu khác nhau làm c s cho vì c t o màu s n mong mu n.



Hình 4.13

kh ng ch các quá trình ta s d ng hai c m bi n báo m c trong bình: C m bi n m c cao (I0.4) và c m bi n m c th p (I0.5). Thi t b tr n c i u khi n b i ng c tr n (Q0.2). Hai b m dùng b m hai lo i s n màu khác nhau vào bình tr n (Q0.0) và (Q0.1). i u khi n m van (Q0.4). B m dùng tháo s n ph m ra kh i bình (Q0.5).

Quá trình làm vi c c a thi t b có th mô t : Khi n nút kh i ng (I0.0) thì h th ng b t u làm vi c v i công vi c u tiên là b m hai lo i s n vào bình. Khi ã s n có trong bình thì I0.5 có m c 0, nh ng ch a y bình thì I0.4 ang có m c 0. Khi l ng s n (t m c cao) thì I0.4 có m c 1, cho tín hi u kh i ng ng c tr n (Q0.2) và b Timer T37 dùng kh ng ch th i gian tr n. Khi t n th i gian ch nh nh c a T37 thì ng c tr n s c c t i n và óng i n m van (Q0.4) và b m tháo s n ph m ra (Q0.5). Khi toàn b s n thành ph m c l y h t kh i bình thì I0.5 có m c 1, cho tín hi u kh i ng b m t i n C30 và reset b timer T37 và h th ng l i ti p t c l p l i quá trình (chu k m i). Khi t s chu k làm vi c c n thi t (t tr c b ng C30) thì h th ng t ng d ng. Trong quá trình làm vi c có th d ng h th ng n u c n nh nút n d ng I0.1. xóa giá tr b m ta s d ng nút n v i a ch là I0.7.

Chương trình vi t trong STL nh sau:

NETWORK 1

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

NETWORK 5

LD M0.0
A I0.0
AN T37
= Q0.2

NETWORK 2

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

NETWORK 6

LD M0.0
A T37
AN I0.5
= Q0.4
= Q0.5

NETWORK 3

LD M0.0
A I0.4
S M0.1, 1

NETWORK 7

LD M0.0
A T37
LD I0.7
CTU C30, +10

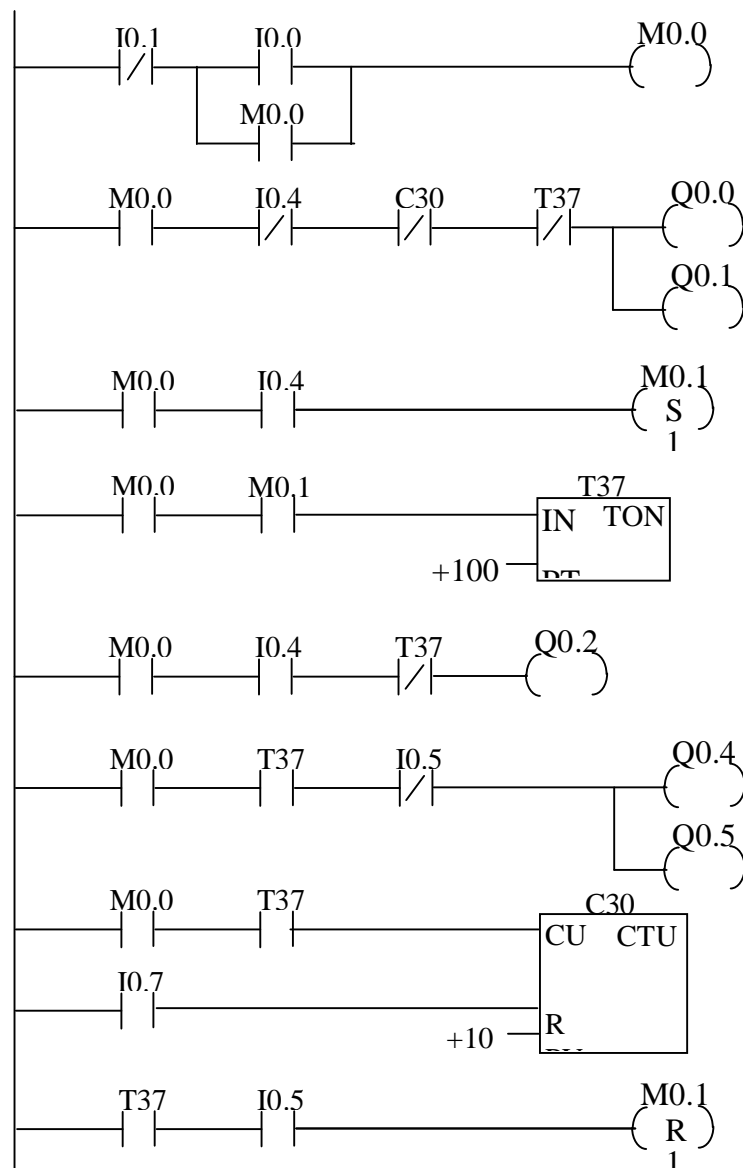
NETWORK 4

LD M0.0
A M0.1
TON T37, +100

NETWORK 8

LD T37
A I0.5
R M0.1, 1

Chứng trình vi tính dạng LAD như sau :



Hình 4.14. S LAD của bài toán trên

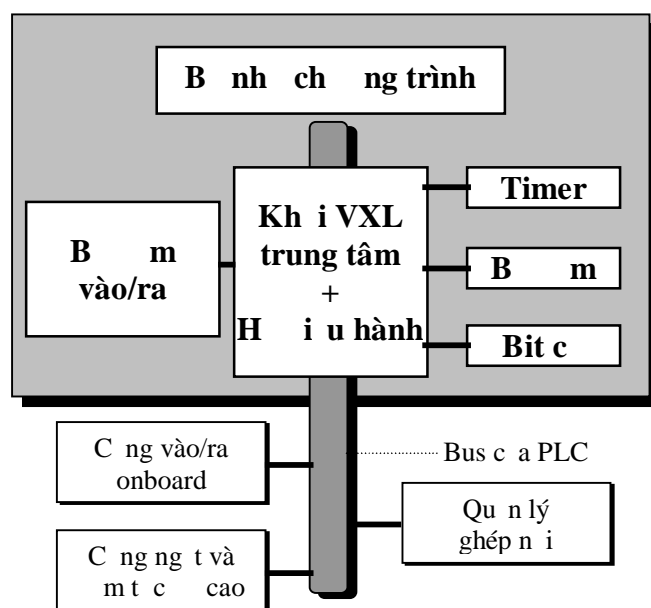
Chương 5

THIẾT BỊ U KHI N LOGIC KH TRÌNH S7-300

5.1. Giới thiệu chung

Thiết bị u khi n logic kh trình (*Programmable Logic Control*), viết tắt thành PLC, là loại thiết bị cho phép thực hiện linh hoạt các thuật toán u khi n s thông qua một ngôn ngữ lập trình, thay cho việc lập trình thủ công trước đây. Như vậy, với một số chương trình u khi n trong thiết bị, PLC trở thành một thiết bị u khi n s nhg n, để thay đổi thuật toán và các bộ phận trao đổi thông tin với môi trường xung quanh (với các PLC khác hoặc với máy tính). Toàn bộ chương trình u khi n sẽ lưu trữ trong bộ nhớ của PLC để điều khiển các khối chương trình (khối OB, FC hoặc FB) và thực hiện lập trình theo chu kỳ của vòng quét (scan).

Có thể thể hiện cấu trúc chung của thiết bị u khi n, tức nhiên PLC phải có tính năng như một máy tính, nghĩa là phải có một bộ vi xử lý (CPU), một thiết bị u hành, bộ nhớ lưu trữ chương trình u khi n, đầu vào và đầu ra tức nhiên là phải có các cổng vào/ra giao tiếp với các thiết bị u khi n và trao đổi thông tin với môi trường xung quanh [13]. Bên cạnh đó, nhằm phục vụ bài toán u khi n s, PLC còn cần phải có thêm các khối chức năng đặc biệt khác như bộ đếm (Counter), bộ thời gian (Timer), và nhiều khối hàm chuyên dùng (hình 5.1).



Hình 5.1. Nguyên lý chung về cấu trúc của một thiết bị u khi n logic kh trình (PLC)

5.2. Các module của PLC S7-300

Thông thường, tính năng của một module được thể hiện trong các thiết bị u khi n có sẵn tín hiệu vào, đầu ra và các chức năng logic khác nhau mà các thiết bị u khi n PLC cần thiết không thể thực hiện được. Chúng ta chia nhỏ thành các module để sử dụng linh hoạt tùy theo từng bài toán, song tất cả đều bao gồm một module chính là module CPU. Các module còn lại là những module nhận/truyền tín hiệu với các thiết bị u khi n, các module chức năng chuyên dùng như PID, u khi n ng c í chúng ta gọi chung là các module mở rộng. Tất cả các module đều được lắp trên một thanh ray.

5.2.1. Modul CPU

Tổng thể S7-200. Tuy nhiên trong hệ PLC S7-300 có nhiều loại module khác nhau và chúng thường được đặt tên theo bố cục lý có trong nó như module CPU312, module CPU314, module CPU315.

5.2.2. Module mở rộng

Các module mở rộng được chia thành 5 loại chính:

PS (Power Supply): module nguồn nuôi: có 3 loại 2A, 5A, 10A.

SM (Signal module): module mở rộng tín hiệu vào/ra bao gồm:

DI (Digital Input): module mở rộng các tín hiệu vào số. Số các tín hiệu vào số mở rộng có thể là 8, 16 hoặc 32 tùy thuộc vào tổng số module.

DO (Digital Output): module mở rộng các tín hiệu ra số. Số các tín hiệu ra số mở rộng có thể là 8, 16 hoặc 32 tùy thuộc ra tổng số module.

DI/DO (Digital Input/Digital Output): module mở rộng các tín hiệu vào/ra số. Số các tín hiệu vào/ra số mở rộng có thể là 8 vào/8 ra, 16 vào/16 ra tùy thuộc ra tổng số module.

AI (Analog Input): module mở rộng các tín hiệu vào tương tự. AO (Analog Output): module mở rộng các tín hiệu ra tương tự. Về bản chất chúng chính là mạch chuyển đổi tín hiệu số 12 bit (AD). Tức là tín hiệu tương tự được chuyển thành tín hiệu số (nguyên) có độ dài 12 bit. Số các tín hiệu ra tương tự có thể là 2, 4 tùy tổng số module.

AI/AO: module mở rộng các tín hiệu vào/ra tương tự. Số các tín hiệu vào/ra tương tự có thể là 4 vào/2 ra hoặc 4 vào/4 ra tùy tổng số module.

IM (Interface module): module ghép nối. Đây là loại module chuyên dùng có nhiệm vụ nối từng nhóm các module mở rộng lại với nhau thành một khối và quản lý chung bởi 1 module CPU.

FM (Function module): module có chức năng riêng biệt. Như module đếm, module xử lý xung nhịp, module xử lý xung nhịp servo, module PID, module xử lý xung nhịp kín.

CP (Communication module): module phục vụ truyền thông trong mạng giữa các PLC với nhau hoặc giữa PLC với máy tính.

5.3. Kiểu dữ liệu và phân chia bộ nhớ

5.3.1- Kiểu dữ liệu:

Một chương trình đang chạy trong S7-300 có thể sử dụng các kiểu dữ liệu sau:

BOOL: vùng chứa 1 mảng bit có giá trị 0 hoặc 1. Đây là kiểu dữ liệu cho biến hai trạng thái.

Byte: vùng chứa dữ liệu nguyên đơn lẻ trong khoảng từ 0÷255 hoặc là mã ASCII của một ký tự.

Word: gồm 2 byte dữ liệu nguyên đơn lẻ từ 0 ÷ 65535.

INT: vùng chứa 2 byte dữ liệu nguyên đơn lẻ trong khoảng -32768 ÷ 32767.

DINT: gồm 4 byte, dữ liệu nguyên đơn lẻ từ -2147483648 ÷ 2147483647

REAL: gồm 4 byte, dữ liệu thực đơn lẻ.

S5T (hay **S5TIME**): khoảng thời gian, tính theo giây/phút/giây/mili giây.

DATE: dữ liệu giá trị thời gian tính theo năm/tháng/ngày

CHAR: dữ liệu một hoặc nhiều ký tự (nhiều nhất là 4 ký tự)

5.3.2. Cấu trúc bộ nhớ của CPU

Bộ nhớ của S7-300 được chia thành 3 vùng chính:

Vùng chứa chương trình đang chạy

Vùng nhớ chứa chương trình được chia thành 3 miền:

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 trao đổi dữ liệu với chương trình gọi nó.

FB (Function block): miền chứa chương trình con, được tổ chức thành hàm và có khả năng trao đổi dữ liệu với các module khác.

Vùng chứa tham số của hệ thống và chương trình đang chạy: được chia thành 7 miền khác nhau:

I (Process image input): miền chứa các dữ liệu được nạp vào.

Q (Process image output): miền chứa các dữ liệu được nạp ra.

M: miền các biến.

T: miền thời gian.

C: mini nh ph c v b m.

PI: mini a ch c ng vào c a các module t ng t .

PQ: mini a ch c ng ra c a các module t ng t .

Vùng ch a các kh i d li u: c chia thành 2 lo i:

DB (Data block): mini ch a các d li u c t ch c thành kh i. Kích th c c ng nh s l ng kh i do ng i s d ng quy nh phù h p v i t ng bài toán i u khi n.

L (Local data block): mini d li u a ph ng, c các kh i ch ng trình OB, FC, FB t ch c và s d ng cho các bi n nháp t c th i và trao i d li u c a bi n hình th c v i nh ng kh i ch ng trình ã g i nó.

5.4. Vòng quét ch ng trình

T ng t nh PLC S7-200.

N u s d ng các kh i ch ng trình c bi t có ch ng t, ví d nh kh i OB40, OB80, í , ch ng trình c a các kh i ó s c th c hi n trong vòng quét khi xu t hi n tín hi u báo ng t cùng ch ng lo i. Các kh i ch ng trình này có th c th c hi n t i m i i m trong vòng quét ch không b gò ép là ph i trong giai o n th c hi n ch ng trình. Ch ng h n n u m t tín hi u báo ng t xu t hi n khi PLC ang giai o n truy n thông và ki m tra n i b , PLC s t m d ng công vi c truy n thông, ki m tra, th c hi n kh i ch ng trình t ng ng v i tín hi u báo ng t ó. V i hình th c x lý tín hi u ng t nh v y, th i gian vòng quét s càng l n khi càng có nhi u tín hi u ng t xu t hi n trong vòng quét. Do ó, nâng cao tính th i gian cho ch ng trình i u khi n, tuy t i không nên vi t ch ng trình x lý ng t quá dài ho c quá l m d ng vi c s d ng ch ng t trong ch ng trình i u khi n.

5.5. Nh ng kh i OB c bi t

Trong khi kh i OB1 c th c hi n u n t ng vòng quét trong giai o n th c hi n ch ng trình (giai o n 2) thì các kh i OB khác ch th c hi n khi xu t hi n tín hi u báo ng t t ng ng, nói cách khác ch ng trình vi t cho các kh i OB này chính là ch ng trình x lý tín hi u ng t (event). Chúng bao g m:

1. OB10 (Time of Day Interrupt): ch ng trình trong kh i OB10 s c th c hi n khi giá tr c a ng h th i gian th c n m trong m t kho ng th i gian ã c quy nh. OB10 có th c g i m t l n, nhi u l n cách u nhau t ng phút, t ng gi , t ng ngày, í V i c quy nh kho ng th i gian hay s l n g i OB10 c th c hi n nh ch ng trình h th ng SFC28 ho c trong b ng tham s c a module CPU nh ph n m m STEP 7.

2. OB20 (Time Delay Interrupt): chương trình trong khối OB20 sẽ thực hiện sau một khoảng thời gian trễ trước khi gọi chương trình hàm SFC32 theo thời gian trễ.
3. OB35 (Cyclic Interrupt): chương trình trong OB35 sẽ thực hiện cách đều nhau một khoảng thời gian cố định. Mặc định, khoảng thời gian này sẽ là 100ms, song ta có thể thay đổi nó trong bảng tham số của module CPU như phần mềm STEP 7.
4. OB40 (Hardware Interrupt): chương trình trong OB40 sẽ xuất hiện một tín hiệu báo ngừng tạm thời vì đưa vào module CPU thông qua các cổng vào/ra số onboard cục bộ, hoặc thông qua các module SM, CP, FM.
5. OB80 (Cycle Time Fault): chương trình trong khối OB80 sẽ thực hiện khi thời gian quét (scan time) vượt quá khoảng thời gian cố định đã quy định hoặc khi có một tín hiệu ngừng tạm thời từ khối OB nào đó mà khối này chưa kích thức lại ngay lập tức. Mặc định, scan time cố định là 150ms, nhưng có thể thay đổi nó thông qua bảng tham số của module CPU như phần mềm STEP 7.
6. OB81 (Power Supply Fault): module CPU sẽ gọi chương trình trong khối OB81 khi phát hiện thấy có lỗi về nguồn nuôi.
7. OB82 (Diagnostic Interrupt): chương trình trong OB82 sẽ gọi khi CPU phát hiện có sự cố tại các module vào/ra mô-đun. Các module mô-đun này phải là những module có khả năng tự kiểm tra mình (diagnostic capabilities).
8. OB85 (Not Load Fault): CPU sẽ gọi khối OB85 khi phát hiện thấy chương trình đang có sự dừng chờ ngừng tạm thời ngừng chương trình xử lý tín hiệu tạm thời không có trong khối OB tạm ngừng.
9. OB87 (Communication Fault): khối OB87 sẽ gọi khi CPU phát hiện thấy lỗi trong truyền thông, ví dụ như không có tín hiệu truyền lại tức thì.
10. OB100 (Start Up Information): khối OB100 sẽ thực hiện một lần khi CPU chuyển trạng thái từ STOP (dừng) sang RUN (chạy).
11. OB101 (chỉ có với S7-400): khối OB101 sẽ thực hiện một lần khi công tắc nguồn của CPU chuyển trạng thái từ OFF sang ON.
12. OB121 (Synchronous error): khối OB121 sẽ thực hiện khi CPU phát hiện thấy lỗi logic trong chương trình như sai kiểu dữ liệu hoặc lỗi truy cập khối DB, FC, FB không có trong bộ nhớ của CPU.
13. OB122 (Synchronous error): khối OB122 sẽ thực hiện khi CPU phát hiện thấy lỗi truy cập module trong chương trình.

5.6. Ngôn ngữ lập trình của S7-300

Các loại PLC nói chung thường có nhiều ngôn ngữ lập trình nhằm phục vụ các ứng dụng khác nhau. PLC S7-300 có ba ngôn ngữ lập trình cơ bản. Đó là:

- Ngôn ngữ liệt kê nhỏ, ký hiệu là STL (Statement list). Đây là 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 chỉ một mệnh đề và đều có cấu trúc chung ở tiền nhỏ + toán hạng.

- Ngôn ngữ hình thang, ký hiệu là LAD (Ladder logic). Đây là dạng ngôn ngữ khá thích hợp với những người quen với kỹ thuật khi làm logic.

- Ngôn ngữ hình khối, ký hiệu là FBD (Function block diagram). Đây cũng là ngôn ngữ khá dành cho người có thói quen với kỹ thuật khi làm.

5.6.1. Cấu trúc lệnh và trạng thái kết quả

Để thực hiện trong S7-300 gồm hai phần: phần chính và phần phụ.

Ví dụ: PIW304 hoặc M300.4

- Phần chính chứa vị trí và kích thước của ô nhớ.

Chú ý: Một chương trình viết trên STL (tùy thuộc vào từng ngôn ngữ lập trình) có thể gồm nhiều Network. Mỗi một Network chứa một hoặc nhiều chương trình phục vụ một công việc nhất định. Mỗi một Network, thanh ghi trạng thái nhận giá trị 0. Chỉ sau lệnh đầu tiên của Network, các bit trạng thái mới thay đổi theo kết quả phép tính.

5.6.2. Các lệnh cơ bản

5.6.2.1. Lệnh gán

Cú pháp = <Toán hạng>

5.6.2.2. Lệnh thực hiện phép tính AND

Cú pháp A <toán hạng>

Toán hạng là dữ liệu kiểu BOOL hoặc các bit I, Q, M, L, D, T, C.

5.6.2.3. Lệnh thực hiện phép tính AND với giá trị nghịch đảo

Cú pháp AN <toán hạng>

Toán hạng là dữ liệu kiểu BOOL hoặc các bit I, Q, M, L, D, T, C.

5.6.2.4. Lệnh thực hiện phép tính OR

Cú pháp O <toán hạng>

Toán hạng là dữ liệu kiểu BOOL hoặc các bit I, Q, M, L, D, T, C.

5.6.2.5. Lệnh thực hiện phép tính OR với giá trị nghịch đảo

Cú pháp ON <toán học>

Toán học là dữ liệu kiểu BOOL hoặc các bit I, Q, M, L, D, T, C.

5.6.2.6. *Lệnh thể hiện phép tính \wedge và giá trị m t bit u th c*

Cú pháp A(

5.6.2.7. *Lệnh thể hiện phép tính \wedge và giá trị ngh ch o c a m t bit u th c*

Cú pháp AN(

Lệnh không có toán học.

5.6.2.8. *Lệnh thể hiện phép tính \vee và giá trị m t bit u th c*

Cú pháp O(

Lệnh không có toán học.

5.6.2.9. *Lệnh thể hiện phép tính \vee và giá trị ngh ch o c a m t bit u th c*

Cú pháp ON(

Lệnh không có toán học.

5.6.2.10. *Lệnh giá trị c a RLO*

Cú pháp NOT

Lệnh không có toán học và có tác động o n i dung c a RLO

5.6.2.11. *Lệnh gán có i u kiện giá trị logic 1 vào ô nh*

Cú pháp S <toán học>

Toán học là các bit I, Q, M, L, D.

5.6.2.12. *Lệnh gán có i u kiện giá trị logic 0 vào ô nh*

Cú pháp R <toán học>

Toán học là các bit I, Q, M, L, D.

5.6.3. Các lệnh i u kiện ch ng trình

5.6.3.1. *Nhóm lệnh k t thúc ch ng trình*

S7-300 có hai lệnh k t thúc ch ng trình là BEC và BEU.

1) Lệnh k t thúc vô i u kiện

Cú pháp BEU

Lệnh không có toán học và thể hiện vị trí k t thúc ch ng trình trong khi m t cách vô i u kiện.

2) Lệnh kết thúc có điều kiện

Cú pháp BEC

Lệnh không có toán hạng và thính hiệu vì kết thúc chương trình trong khi nếu RLO có giá trị 1.

5.5.3.2- Nhóm lệnh nhánh theo bit trạng thái

Lệnh nhánh theo bit trạng thái là loại lệnh thính hiệu để chuyển nhánh qua một hoặc nhiều chương trình tiếp theo khác để ảnh hưởng đến việc điều kiện kiểm tra trong thanh ghi trạng thái tiếp theo. Nếu lệnh này tiếp tục cùng một khi chương trình tiếp theo. Không thể chuyển từ chương trình này sang một khi chương trình khác, ví dụ không thể chuyển từ FC1 sang FC10.

1) R nhánh khi BR = 1

Cú pháp JBI <nhãn >

2) R nhánh khi BR = 0

Cú pháp JNBI <nhãn >

Lệnh thay thế thanh ghi trạng thái ghi nhớ là JBI:

3) R nhánh khi RLO = 1

Cú pháp JC <nhãn >

4) R nhánh khi RLO = 0

Cú pháp JCN <nhãn >

Lệnh thay thế thanh ghi trạng thái ghi nhớ là JC:

5) R nhánh khi CC1 = 0 và CC0 = 1

Cú pháp JM <nhãn >

Lệnh này không làm thay đổi nội dung thanh ghi trạng thái. Nó chỉ sử dụng nhánh nếu phép tính trước có kết quả âm.

6) R nhánh khi CC1 = 1 và CC0 = 0

Cú pháp JP <nhãn >

Lệnh này không làm thay đổi nội dung thanh ghi trạng thái. Nó chỉ sử dụng nhánh nếu phép tính trước có kết quả dương.

7) R nhánh khi CC1 = CC0 = 0

Cú pháp JZ <nhãn >

Lệnh này không làm thay đổi nội dung thanh ghi trạng thái. Nó chỉ sử dụng nhánh nếu phép tính trên có kết quả bằng 0.

8) R nhánh khi $CC1 \neq CC0$

Cú pháp JN <nhãn >

Lệnh này không làm thay đổi nội dung thanh ghi trạng thái. Nó chỉ sử dụng nhánh nếu phép tính trên có kết quả khác 0.

9) R nhánh khi $CC1 = CC0 = 0$ hoặc $CC1 = 0$ và $CC0 = 1$

Cú pháp JMZ <nhãn >

Lệnh này không làm thay đổi nội dung thanh ghi trạng thái. Nó chỉ sử dụng nhánh nếu phép tính trên có kết quả là mts không đúng (âm hoặc bằng 0).

10) R nhánh khi $CC1 = CC0 = 0$ hoặc $CC1 = 1$ và $CC0 = 0$

Cú pháp JMZ <nhãn >

Lệnh này không làm thay đổi nội dung thanh ghi trạng thái. Nó chỉ sử dụng nhánh nếu phép tính trên có kết quả là mts không âm (dương hoặc bằng 0).

11) R nhánh vô điều kiện

Cú pháp JU <nhãn >

Lệnh này không làm thay đổi nội dung thanh ghi trạng thái và chỉ thị vô điều kiện, không phụ thuộc bất kỳ bit trạng thái nào.

5.6.3.3. Lệnh xoay vòng (LOOP)

Cú pháp LOOP <nhãn >

Khi gặp lệnh LOOP, CPU của S7-300 sẽ kiểm tra nội dung của thanh ghi ACCU1 để xem có bằng 0 hay không. Nếu kết quả khác 0, CPU sẽ thực hiện bước tiếp theo của chương trình và đánh dấu bit nhảy. Ngược lại thì CPU sẽ thực hiện lại từ đầu.

Lệnh này không làm thay đổi nội dung thanh ghi trạng thái.

5.6.3.4- Lệnh rẽ nhánh theo danh mục (JUMP LIST)

Cú pháp JL <nhãn >

Lệnh thực hiện một loạt rẽ nhánh tùy theo nội dung của ACCU1. Danh mục các nhánh được xếp ngay sau lệnh JL để định nghĩa vô điều kiện và vị trí tiếp theo của chương trình theo nội dung của ACCU1.

Số các nhánh rẽ nhánh tối đa có thể là 255. Toán hạng <nhãn> trong lệnh chỉ định vị trí bắt đầu của danh mục các nhánh rẽ.

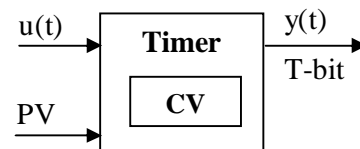
Lưu ý rằng nhánh theo danh mục không làm thay đổi nội dung thanh ghi trạng thái và có tác động gì như bình thường do case của Access hay dBASE.

5.6.4 B. Thời gian (Timer)

5.6.4.1. Nguyên tắc làm việc

Bộ thời gian (Timer), là bộ đo thời gian trông mong muốn ghi giá trị logic vào vào $u(t)$ và giá trị logic ra $y(t)$.

S7-300 có 5 loại Timer khác nhau. Tất cả 5 loại Timer này cùng bộ đo thời gian trông mong muốn ghi giá trị logic vào vào $u(t)$ và giá trị logic ra $y(t)$ khi giá trị logic vào vào $u(t)$ chuyển trạng thái logic từ 0 lên 1, nghĩa là thời điểm Timer kích.

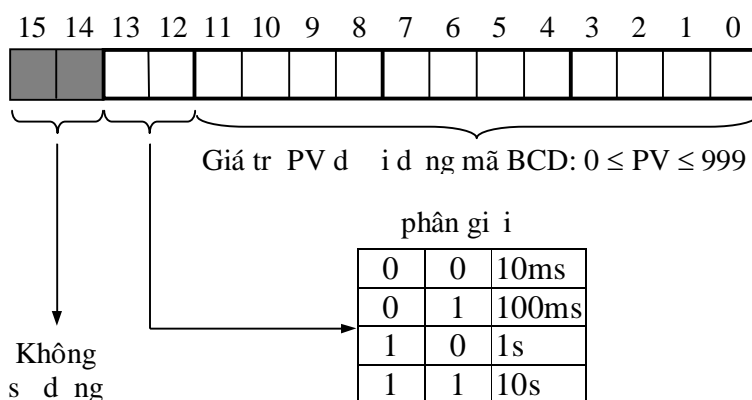


Hình 5.1. Mô tả nguyên lý làm việc của Timer

Thời gian trông mong muốn khai báo với Timer bằng một giá trị 16 bits (hình 2.8) bao gồm hai thành phần:

- phân ghi với n là ms. Timer của S7-300 có 4 loại phân ghi khác nhau là 10ms, 100ms và 10s.

- Mã số nguyên (BCD) trong khoảng 0÷999 nghĩa là PV (Preset value-giá trị đặt trước).



Hình 5.2. Cấu hình giá trị thời gian đặt trước khi khai báo với timer

Như vậy thời gian trông mong muốn chính là tích $\tau = \text{phân ghi} \times PV$

Ngay tại thời điểm kích Timer, giá trị PV được chuyển vào thanh ghi 16 bits của Timer T-Word (gọi là thanh ghi CV, vì tất cả của Current value-giá trị hiện tại). Timer sẽ ghi nhớ khoảng thời gian trôi qua kể từ khi kích bằng cách giảm dần một cách từng bước nội dung thanh ghi CV. Nếu nội dung thanh ghi CV trở về bằng 0 thì Timer sẽ chuyển nội dung mong muốn τ và từ đây sẽ báo ra ngoài bằng cách chuyển trạng thái giá trị logic ra $y(t)$. Vì thông báo ra ngoài bằng cách chuyển trạng thái giá trị logic ra $y(t)$ như thế nào còn phụ thuộc vào loại Timer nào được sử dụng.

Bên cạnh số lên của giá trị logic vào vào $u(t)$. Timer còn có thể kích bằng số lên của giá trị logic kích chế có tên là giá trị logic enable như tại thời điểm có số lên của giá trị logic enable, giá trị logic vào vào $u(t)$ có giá trị logic 1.

Tổng loại Timer được ánh xạ từ 0 đến (tùy thuộc tổng loại CPU) 255. Mã Timer có tên là Tx, trong đó x là số Timer ($0 \leq x \leq 255$). Ký hiệu Tx cũng có thể là địa chỉ

hình thức của thanh ghi CV (T-Word) và của u ra (T-bit) của Timer đó. Tuy chúng có cùng các hình thức, song T-Word và T-bit vẫn được phân biệt với nhau như khi u l nh s đ ng v i toán h ng Tx. Khi dùng l nh làm vi c v i t , Tx c hi u là a ch c a T-Word, ng c l i khi s đ ng l nh làm vi c v i t i p i m, Tx c hi u là a ch c a T-bit.

M t Timer đang trong ch làm vi c (sau khi c kích) có th c a l i v tr ng thái ch kh i ng ban u, t c là ch s n lên t i p theo c a tín hi u u vào. Công vi c này g i là reset Timer đó. Tín hi u reset Timer c g i là tín hi u xóa và khi tín hi u xóa có giá tr b ng 1 Timer s không làm vi c. T i th i i m xu t hi n s n lên c a tín hi u xóa, T-Word và T-bit c a nó ng th i c xóa v 0, t c là thanh ghi m t c th i CV c t v 0 và tín hi u u ra c ng có tr ng thái logic b ng 0.

5.6.4.2. Khai báo s đ ng

V i c khai báo s đ ng m t Timer g m các b c :

Khai báo tín hi u enable n u mu n s đ ng tín hi u ch ng kích.

Khai báo tín hi u u vào u(t).

Khai báo th i gian tr mong mu n.

Khai báo lo i Timer c s đ ng (SD, SS, SP, SE, SF).

Khai báo tín hi u xóa Timer n u mu n s đ ng ch reset ch ng.

Trong t t c 5 b c trên, các b c 2, 3, 4 là b t bu c.

(1) Khai báo tín hi u enable (ch ng kích)

Cú pháp A < a ch bit>

FR <Tên Timer>

Toán h ng th nh t ò a ch bit ò xác nh tín hi u s c s đ ng làm tín hi u ch ng kích cho Timer có tên cho trong toán h ng th 2.

(2) Khai báo tín hi u u vào

Cú pháp A < a ch bit>

ò a ch bit ò trong toán h ng xác nh tín hi u u vào u(t) cho Timer.

(3) Khai báo th i gian tr mong mu n

Cú pháp L <h ng s >

õ h ng s ò trong toán h ng xác nh tín hi u giá tr th i gian tr τ t tr c cho Timer. H ng s này có hai d ng.

- S5T#gi H_phútM_miligiâyMS. ây là d ng d li u th i gian th c.

- Dùng m t s nguyên 16 bits.

(4) Khai báo lo i Timer

S7-300 có 5 lo i Timer c khai báo b ng các l nh

- SD: Tr theo s n lên không có nh .
- SS: Tr theo s n lên có nh .
- SP: T o xung không có nh .
- SE: T o xung có nh .
- SF : Tr theo s n xu ng.

a) Tr theo s n lên không có nh (On delay Timer)

Cú pháp A <Tên Timer>

Th i gian gi tr c b t u khi có s n lên c a tín hi u u vào (ho c khi có s n lên c a tín hi u enable ng th i c a tín hi u vào b ng 1), t c là ngay th i i m ó giá tr PV c chuy n vào thanh ghi T-Word (CV). Trong kho ng th i gian T-bit có giá tr 0. Khi h t th i gian tr T-bit có giá tr b ng 1. Nh v y, T-bit có giá tr 1 khi T-Word=0.

Kho ng th i gian tr chính là kho ng th i gian gi a th i i m xu t hi n s n lên c a tín hi u u vào và s n lên c a T-bit.

Khi tín hi u vào b ng 0, T-bit và T-Word cùng nh n giá tr 0.

b) Tr theo s n lên có nh

Cú phápA <Tên Timer>

Th i gian tr c b t u tính t khi xu t hi n s n lên c a tín hi u u vào (ho c khi có s n lên c a tín hi u enable ng th i tín hi u vào b ng 1), t c là ngay th i i m ó giá tr PV c chuy n vào thanh ghi T-Word (CV). Khi h t th i gian tr , t c là khi T-Word=0, T-bit có giá tr b ng 1.

Kho ng th i gian tr chính là kho ng th i gian gi a th i i m xu t hi n s n lên c a tín hi u u vào và s n lên c a T-bit.

V i b Timer có nh , th i gian tr v n c tính cho dù lúc ó tín hi u u vào ã v 0.

c) Timer t o xung không có nh (Pulse Timer)

Cú phápSP <Tên Timer>

Th i gian tr c b t u tính t khi xu t hi n s n lên c a tín hi u u vào (ho c khi có s n lên c a tín hi u enable ng th i tín hi u vào b ng 1), t c là ngay th i i m

ó giá trị PV được chuyển vào thanh ghi T-Word (CV). Trong khoảng thời gian tr, t c là khi T-Word \neq 0, T-bit có giá trị bằng 1. Ngoài khoảng thời gian tr T-bit có giá trị bằng 0.

Nếu chưa hết thời gian tr mà tín hiệu u vào v 0 thì T-bit và T-Word cũng về giá trị 0.

d) Timer tạo xung có nh (Extended Pulse Timer)

Cú phápSE <Tên Timer>

Thời gian gì được bắt đầu khi xuất hiện sườn lên của tín hiệu u vào (hoặc khi có sườn lên của tín hiệu u enable trong thời gian tín hiệu u vào bằng 1), t c là ngay thời điểm có giá trị PV được chuyển vào thanh ghi T-Word (CV).

Trong khoảng thời gian tr, t c là khi T-Word \neq 0, T-bit có giá trị bằng 1. Ngoài khoảng thời gian tr T-bit có giá trị bằng 0.

Nếu chưa hết thời gian tr mà tín hiệu u vào v 0 thì thời gian tr vẫn được tính tiếp t c, t c là T-bit và T-Word không về 0 theo tín hiệu u vào.

e) Timer trễ theo sườn xuống (Off Delay Timer)

Cú phápSF <Tên Timer>

Thời gian tr được tính bắt đầu khi có sườn xuống của tín hiệu u vào, t c là thời điểm xuất hiện sườn xuống của tín hiệu u vào, giá trị PV được chuyển vào thanh ghi T-Word (CV).

Trong khoảng thời gian gì sườn lên của tín hiệu u vào hoặc T-Word \neq 0, T-bit có giá trị bằng 1. Ngoài ra khoảng thời gian T-bit có giá trị bằng 0.

(5) Khai báo tín hiệu xóa (reset)

Cú pháp A < a ch bit>

R <Tên Timer>

Toán học nh t ã a ch bit ã xác định tín hiệu u s được dùng làm tín hiệu u ch ng xóa cho Timer có tên trong toán học nh t hai.

Khi tín hiệu xóa=1, T-Word (thanh ghi CV) và T-bit cùng về giá trị 0. Nếu tín hiệu xóa về 0, Timer sẽ kích lại.

5.6.4.3. *c n i dung thanh ghi T-Word (CV)*

Nội dung thanh ghi T-Word là CV có thể được nạp vào ACCU1 theo 2 cách:

1) nạp số m t c th i (không có phân ghi i)

Cú pháp L <Tên Timer>

Toán học nh t tên Timer mà thanh ghi T-Word của nó sẽ được nạp vào ACCU1.

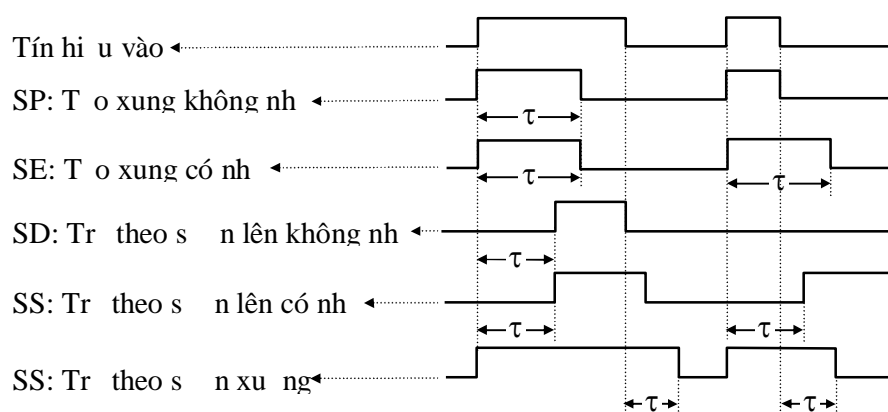
Giá trị τ là một số nguyên dương xác định số mặt cắt (không có thập nguyên), τ là chu kỳ của xung thời gian khi Timer được kích, và phân giải.

2) Thời gian trễ mặt cắt

Toán học tên Timer mà thanh ghi T-Word của nó được nạp vào ACCU1. Giá trị được nạp hai phần: một số BCD xác định số mặt cắt (không có thập nguyên) và phân giải.

Tổng kết

Hình dưới đây liệt kê các loại Timer của S7-300 cho tiện việc tra cứu sử dụng, trong đó τ là thời gian trễ mặt cắt.



5.6.5. Bộ đếm (Counter)

5.6.5.1. Nguyên tắc làm việc

Counter là bộ đếm thể hiện chức năng đếm xung của các tín hiệu vào. S7-300 có tới 256 Counter (phụ thuộc CPU), ký hiệu bằng Cx, trong đó x là số nguyên trong khoảng 0÷255. Nhờ bộ đếm của S7-300 có thể đếm theo số lần tăng của một tín hiệu vào nhất định, ký hiệu là CU (count up) và đếm lùi theo số lần tăng của tín hiệu vào thì hai ký hiệu là CD (count down).

Thông thường bộ đếm các số lần tăng của tín hiệu CU và CD, song cũng có thể đếm ngược đếm các tín hiệu của chúng bằng cách sử dụng thêm tín hiệu enable (kích hoạt). Nếu có tín hiệu enable, bộ đếm sẽ đếm khi xuất hiện số lần tăng của tín hiệu enable ngay thì tức là CU có một tín hiệu là 1. Tương tự bộ đếm lùi khi có số lần tăng của tín hiệu enable và tức thì tức là CD có một tín hiệu là 1.

Số xung đếm được, được ghi vào thanh ghi 2 byte của bộ đếm, gọi là thanh ghi C-Word. Nội dung của T-Word cũng là giá trị mặt cắt của bộ đếm và ký hiệu bằng CV (Current Value). Bộ đếm báo trạng thái của C-Word ra ngoài qua chân C-bit của nó. Nếu CV≠0, C-bit có giá trị 1. Ngược lại khi CV=0, C-bit nhận giá trị logic 0. CV luôn là một giá trị không âm. Bộ đếm sẽ không đếm lùi khi CV=0.

Khác với Timer, giá trị đặt trước PV (preset value) của bộ đếm chuyển vào C-Word thì khi đếm xong thì sẽ lên cao tín hiệu set (set-S)

Bộ đếm có thể xóa bằng cách đưa tín hiệu xóa (reset). Khi bộ đếm xóa, C-Word và C-bit đều như giá trị 0.

5.6.5.2. Khai báo sơ đồ

Việc khai báo sơ đồ đếm bao gồm các bước:

- Khai báo tín hiệu enable của module đếm tín hiệu bằng kích thước.
- Khai báo tín hiệu vào CU của module.
- Khai báo tín hiệu vào CD của module.
- Khai báo tín hiệu set (set) và giá trị đặt trước (PV).
- Khai báo tín hiệu xóa (reset).

Trong đó ít nhất phải có một trong hai bước 2 hoặc 3 để thể hiện.

(1). Khai báo tín hiệu kích thước (enable).

Cú pháp A < a ch bit>

FR <Tên Counter>

Toán học thể hiện tổng các bit xác định tín hiệu sẽ đếm làm tín hiệu kích thước cho bộ đếm có tên cho trong toán học thể hiện hai. Tên của bộ đếm có dạng Cx với $0 \leq x \leq 255$

(2). Khai báo tín hiệu của module theo sơ đồ.

Cú pháp A < a ch bit>

CU <Tên Counter>

Toán học thể hiện tổng các bit xác định tín hiệu mà sẽ lên cao nó của bộ đếm và tên cho trong toán học thể hiện hai module. Tên của bộ đếm có dạng Cx với $0 \leq x \leq 255$. Mỗi khi đếm xong thì sẽ lên cao tín hiệu, bộ đếm sẽ ngừng nội dung thanh ghi C-Word (CV) lên 1 lần. Lưu ý CU tác động vào thanh ghi trạng thái gì ngược lại như FR.

(3). Khai báo tín hiệu của module theo sơ đồ.

Cú pháp A < a ch bit>

CD <Tên Counter>

Toán học thể hiện tổng các bit xác định tín hiệu mà sẽ lên cao nó của bộ đếm và tên cho trong toán học thể hiện hai module. Tên của bộ đếm có dạng Cx với $0 \leq x \leq 255$.

M i khi xu t hi n s n lên c a tín hi u, b m s gi m n i dung thanh ghi C-Word (CV) i b t l n v n u CV>0. Trong tr ng h p CV ã b ng 0 thì n i dung C-Word không b thay i. L nh CD tác ng vào thanh ghi tr ng thái gi ng nh l nh FR.

(4). Khai báo tín hi u t (set) giá tr t tr c (PV)

Cú pháp A < a ch bit>

L C#<h ng s >

S <Tên Counter>

Toán h ng th nh t õ a ch bitõ xác nh tín hi u mà m i khi xu t hi n s n lên c a nó, h ng s PV cho trong l nh th hai đ i đ ng BCD s c chuy n vào thanh ghi C-Word c a b m có tên trong toán h ng c a l nh th 3.

Tên c a b m có đ ng Cx v i $0 \leq x \leq 255$.

(5). Khai báo tín hi u xóa (reset)

Cú pháp A < a ch bit>

R <Tên Counter>

Toán h ng th nh t õ a ch bitõ xác nh tín hi u mà m i khi xu t hi n s n lên c a nó, thanh ghi C-Word c a b m có tên trong toán h ng c a l nh th 2 s c xóa v 0.

Tên c a b m có đ ng Cx v i $0 \leq x \leq 255$.

(6). c n i dung thanh ghi C-Word

N i dung c a thanh ghi C-Word là CV, c ng gi ng nh Timer, có th c c vào ACCU1 theo 2 cách:

a) c s m t c th i đ ng binary

Cú pháp L <Tên Counter>

Toán h ng là tên b m mà thanh ghi C-Word c a nó s c c vào ACCU1. Giá tr c c là m t s nguyên đ ng xác nh s m t c th i.

Tên c a b m có đ ng Cx v i $0 \leq x \leq 255$.

b) c s m t c th i đ ng BCD

Cú pháp LC <Tên Counter>

Toán h ng là tên b m mà thanh ghi C-Word c a nó s c c vào ACCU1. Giá tr c c là s BCD.

Tên c a b m có đ ng Cx v i $0 \leq x \leq 255$.

5.6.6- Kỹ thuật sử dụng con trỏ

Con trỏ (Pointer) là một công cụ mạnh mẽ, rất cần dùng trong các chương trình điều khiển. Vì các sử dụng con trỏ cho phép truy cập gián tiếp tới dữ liệu trong bộ nhớ. Ta hãy xét lệnh nạp dữ liệu ô nhớ MW0 vào ACCU1 làm ví dụ:

LC MW0 // nạp giá trị của ô nhớ MW0 vào thanh ghi ACCU1

Lệnh này là lệnh truy cập trực tiếp ô nhớ MW0 vì địa chỉ của ô nhớ đó là MW0 đã cho trực tiếp trong lệnh điều khiển toán học. Như vậy, có thể hình dung ra lệnh nạp dữ liệu ô nhớ MW0 mà địa chỉ ô nhớ đó không cho trực tiếp trong lệnh sẽ là lệnh truy cập gián tiếp.

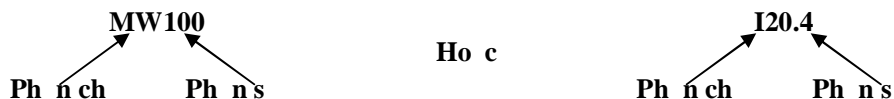
Trong lệnh truy cập gián tiếp, địa chỉ ô nhớ được truy cập sẽ là nội dung của một ô nhớ khác mà ta gọi là con trỏ. Ví dụ vì truy cập trực tiếp ô nhớ MW0 trên tổng cộng với lệnh truy cập gián tiếp như con trỏ MD10 như sau:

L 0

T MD10

L MW[10] // nạp giá trị của ô nhớ có địa chỉ cho trong MD10

Như phần trước đã mô tả, địa chỉ ô nhớ trong S7-300 gồm 2 phần: phần chỉ và phần số. Ví dụ:



Trong đó phần chỉ chỉ vị trí trong vùng, kích thước của ô nhớ và phần số chỉ địa chỉ của byte hoặc bit trong vùng nhớ đã xác định. Tổng cộng với cách biểu diễn địa chỉ như vậy mà con trỏ cũng có 2 dạng:

- Chỉ địa chỉ phần số. Đây là kiểu con trỏ địa chỉ xác định vị trí ô nhớ trong vùng.
- Chỉ địa chỉ phần số và chỉ. Đây là con trỏ toàn cục xác định vị trí ô nhớ trong bộ nhớ.

5.6.6.1 Sử dụng MW hoặc bit kép MD làm con trỏ

Ta có thể sử dụng một ô nhớ thuộc vùng nhớ M có kích thước là từ (MW) hoặc bit kép (MD) làm con trỏ. Trong những trường hợp như vậy, con trỏ MW hoặc MD chỉ có thể là con trỏ địa chỉ (chỉ địa chỉ phần số của địa chỉ).

Do phần số của địa chỉ có hai dạng thể hiện (địa chỉ byte, ví dụ: 20, 22, 100, ...).

Hay địa chỉ bit, ví dụ: 20.0, 22.2, 100.5, ...) nên con trỏ địa chỉ phần số MW, MD cũng có hai hình thái.

- Con trỏ địa chỉ chỉ vị trí byte trong vùng và con trỏ địa chỉ chỉ vị trí bit trong vùng.