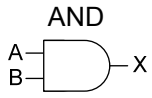
8 Thiết kế theo logic Bool & biểu đồ Karnaugh

8.1 Giới thiệu

Quá trình chuyển đổi một mục tiêu điều khiển thành một chương trình theo ngôn ngữ LAD, FBD hay STL yêu cầu phải thông qua một cấu trúc. Đại số BOOL là một trong các công cụ cần thiết để phân tích và thiết kế những hệ thống này.

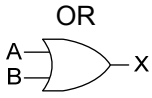
8.2 Đại số BOOL

Đại số BOOL được phát triển vào năm 1800 bởi một nhà toán học người Ai-len tên là James Bool. Nó cực kỳ hữu ích trong thiết kế các mạch số. Nó vẫn được sử dụng nhiều bởi các kỹ sư điện và tin học. Phương pháp thực hiện là mô hình hệ thống logic bằng các công thức riêng lẻ. Công thức có thể là sự kết hợp của các AND/OR đơn giản thành các dạng mới. Với cùng phương pháp này, người thiết kế mạch có thể ứng dụng cho lập trình ở LAD.



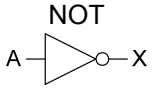
$X = A \cdot B$

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1



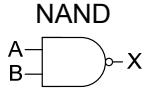
$X = A + B$

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1



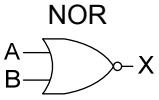
$X = \overline{A}$

A	X
0	1
1	0



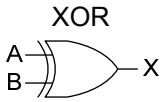
$X = \overline{A \cdot B}$

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1



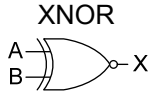
$X = \overline{A + B}$

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0



$X = A \oplus B$

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0



$X = \overline{A \oplus B}$

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

Hình 8.1: Các phép toán đại số bool với bảng sự thật và cổng logic

Công thức Boolean bao gồm nhiều biến và các hoạt động giống như các công thức đại số thông thường. Ba phép toán cơ bản là AND, OR và NOT, hoặc tổ hợp của các phép toán cơ bản là NAND, NOR, XOR, XNOR. Các phép toán với bảng sự thật được cho ở hình 4.1. Mỗi phép toán được trình bày bởi một công thức đơn giản với hai biến được sử dụng là A và B để tính giá trị X. Bảng sự thật là một phương pháp đơn giản để mô tả tất cả các tổ hợp có thể có là cho ngõ ra ở trạng thái "ON" hoặc "OFF" ("1" hoặc "0").

Chú ý: Cổng XOR thường được chuyển thành các cổng tương đương như sau:

$$X = A \oplus B = A \cdot \overline{B} + \overline{A} \cdot B$$

- **Các định lý của đại số Bool**

Tiên đề:

1. $A + A = 0$
2. $A \cdot 1 = A$
3. $A \cdot \overline{A} = 0$
4. $A + \overline{A} = 1$
5. $\overline{1} = 0$

Định lý:

1. $A + A = A$
2. $A \cdot A = A$
3. $A + 1 = 1$
4. $A \cdot 0 = 0$
5. $A + A \cdot B = A$
6. $A \cdot (A + B) = A$
7. $\overline{\overline{A}} = A$
8. $\overline{(A + B)} = \overline{A} \cdot \overline{B}$
9. $\overline{(A \cdot B)} = \overline{A} + \overline{B}$

} Định l ý DeMorgan's

10. $(A + B) + C = A + (B + C)$
11. $(A \cdot B) \cdot C = A \cdot (B \cdot C)$
12. $A + \overline{A} \cdot B = A + B$
13. $A \cdot (\overline{A} + B) = A \cdot B$
14. $A + B = B + A$
15. $A \cdot B = B \cdot A$
16. $A + (B \cdot C) = (A + B) \cdot (A + C)$
17. $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$

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

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

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

Ví dụ: Cho biểu thức $A=\overline{B}.(C.(\overline{D}+E+C)+\overline{F}.C)$

Biểu thức đại số A được đơn giản theo các bước như sau:

$$A=\overline{B}\cdot(C\cdot(\overline{D}+E+C)+\overline{F}\cdot C)$$

$$A=\overline{B}\cdot(\overline{D}\cdot C+E\cdot C+C\cdot C+\overline{F}\cdot C) \quad (1)$$

$$A=\overline{B}\cdot(\overline{D}\cdot C+E\cdot C+C+\overline{F}\cdot C) \quad (2)$$

$$A=\overline{B}\cdot C\cdot(\overline{D}+E+1+\overline{F}) \quad (3)$$

$$A=\overline{B}\cdot C\cdot(1) \quad (4)$$

$$A=\overline{B}\cdot C \quad (5)$$

Chú ý: Khi đơn giản các biểu thức đại số Bool, phép tóan OR có ưu tiên thấp nên chúng được thực hiện trước. Phép toán NOT có ưu tiên cao nhất, nên chúng được đơn giản sau. Cách thức thực hiện có thể minh họa cho việc đơn giản một biểu thức đại số như sau:

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

$$X=\overline{(A)}+\overline{(B\cdot C)}+A\cdot(B+\overline{C})$$ ← Các phép toán có ưu tiên cao được đặt trong ngoặc

$$X=\overline{(A)}\cdot\overline{(B\cdot C)}+A\cdot(B+\overline{C})$$ ← Ứng dụng định lý DeMorgan's

$$X=\overline{A}\cdot(\overline{B}+\overline{C})+A\cdot(B+\overline{C})$$ ← Ứng dụng tiếp định lý DeMorgan's

$$X=\overline{A}\cdot\overline{B}+\overline{A}\cdot\overline{C}+A\cdot B+A\cdot\overline{C}$$ ← Bỏ ngoặc

$$X=\overline{A}\cdot\overline{B}+(\overline{A}\cdot\overline{C}+A\cdot\overline{C})+A\cdot B$$ ← Chọn các số hạng có cùng thừa số, ở đây chỉ có NOT C

$$X=\overline{A}\cdot\overline{B}+\overline{C}\cdot(\overline{A}+A)+A\cdot B$$ ← Đặt thừa số chung

$$X=\overline{A}\cdot\overline{B}+\overline{C}+A\cdot B$$ ← Ứng dụng định lý để đơn giản

8.3 Thiết kế Logic

Các ý tưởng thiết kế có thể được chuyển đổi trực tiếp từ các biểu thức đại số Bool, hoặc bằng các phương pháp khác (ở các chương sau). Các biểu thức đại số Bool có thể được đơn giản hoặc sắp xếp lại và sau đó chuyển sang sơ đồ LAD hoặc FBD hay ở ngôn ngữ STL.

Nếu chúng ta mô tả một qui trình điều khiển bằng lời, thì chúng ta thường có thể chuyển trực tiếp nó thành biểu thức đại số Bool như ở hình 8.2

và hình 8.3. Trong ví dụ, việc mô tả quá trình được đưa ra trước. Trong các ứng dụng thực tế, điều này có được nhờ vào các bộ phận cơ của hệ thống. Trong nhiều trường hợp hệ thống chưa có, việc thực hiện sẽ là một bài tóan cho người thiết kế. Bước kế tiếp là xác định bộ điều khiển nên làm việc như thế nào. Trong trường hợp này, các câu lệnh được viết ra trước tiên, và sau đó chuyển đổi thành biểu thức đại số Bool. Biểu thức đại số Bool có thể được chuyển đổi theo dạng mong muốn. Công thức đầu tiên chứa một XOR, nó không thể biểu diễn dược ở dạng LAD, như vậy nên chuyển nó thành dạng các cổng tương đương sử dụng AND, OR và NOT.

Ví dụ 8.1: Điều khiển nhiệt độ lò nhiệt

Mô tả quá trình:

Một lò nhiệt có hai cửa có thể cấp nhiệt cho thỏi kim loại đúc ở mỗi cửa. Bộ phát nhiệt cung cấp đủ nhiệt cho hai thỏi kim loại đúc. Nhưng nếu chỉ có một thỏi kim lọai đúc thì nhiệt độ cung cấp trở nên quá nóng, để giảm nhiệt độ thì một quạt giải nhiệt cho lò sẽ được bật.

Mô tả điều khiển:

Nếu nhiệt độ quá cao và chỉ có một thỏi kim loại đúc ở một cửa thì bật quạt.

Giải

Bảng xác định input/output:

Ký hiệu	***Địa chỉ***	***Chú thích***
B1	I0.0	Cảm biến báo có thỏi kim loại đúc ở cửa 1
B2	I0.1	Cảm biến báo có thỏi kim loại đúc ở cửa 2
T	I0.2	Cảm biến báo quá nhiệt
F	Q0.0	Quạt giải nhiệt

Biểu thức đại số Bool:

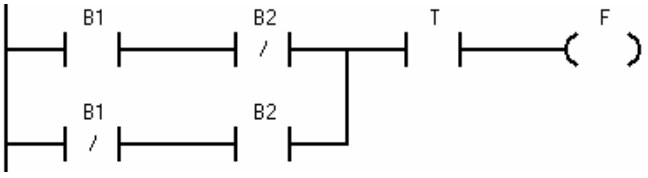
$$F = T \cdot (B_1 \oplus B_2) \qquad (1)$$

$$F = T \cdot (B1 \cdot \overline{B_2} + \overline{B_1} \cdot B_2) \qquad (2)$$

$$F = B_1 \cdot \overline{B_2} \cdot T + \overline{B_1} \cdot B_2 \cdot T \qquad (3)$$

Chương trình biểu diễn ở ngôn ngữ LAD, FBD và STL (đối với biểu thức 2):

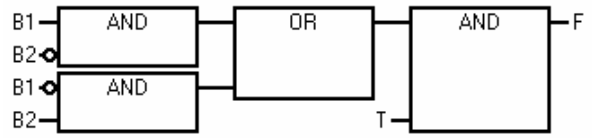
LAD



STL

```
LD   B1
AN   B2
LDN  B1
A    B2
```

FBD

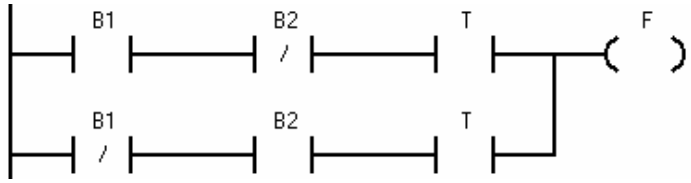



```
OLD
A     T
=     F
```

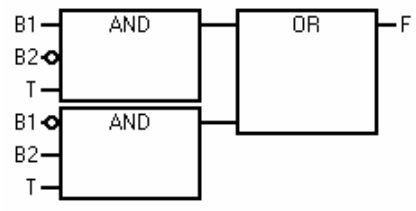
Hình 8.2: Biểu thức đại số Bool được thiết kế theo ngôn ngữ của PLC S7-200

Chương trình biểu diễn ở ngôn ngữ LAD, FBD và STL (đối với biểu thức 3):

LAD



FBD



STL

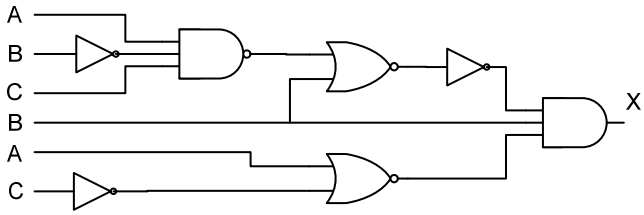
```
LD    B1
AN    B2
A     T
LDN   B1
A     B2
A     T
OLD
=     F
```

Hình 8.3: Biểu thức đại số Bool được thiết kế theo ngôn ngữ của PLC S7-200

Ví dụ 8.2: Hãy chuyển sơ đồ logic sau đây (hình 8.4) thành chương trình trong PLC ở ngôn ngữ LAD, FBD và STL:

Giải:

Nếu cứ giữ nguyên sơ đồ logic thì việc chuyển đổi chương trình ở LAD sẽ gặp nhiều khó khăn vì trong PLC không thể biểu diễn được cổng NAND và NOR. Vì vậy để đơn giản hơn, ta sử dụng phương pháp biến đổi sơ đồ thành biểu thức đại số Bool và sau đó đơn giản biểu thức này.



Hình 8.4: Sơ đồ logic

Sơ đồ trên được biểu diễn ở dạng biểu thức đại số Bool và sau đó được đơn giản.

$$X = \overline{\left(\overline{\overline{(A \cdot \overline{B} \cdot C)} + B}\right)} \cdot B \cdot \overline{(A + \overline{C})}$$

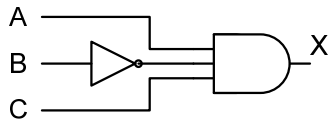
$$X = (\overline{A} + B + \overline{C} + B) \cdot B \cdot (\overline{A} \cdot C)$$

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

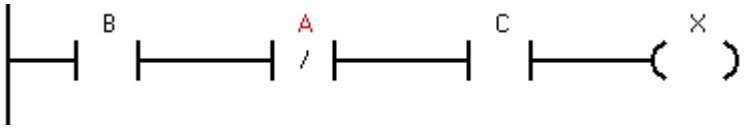
$$X = B \cdot \overline{A} \cdot C + B \cdot \overline{A} \cdot C + 0 + B \cdot \overline{A} \cdot C$$

$$X = B \cdot \overline{A} \cdot C$$

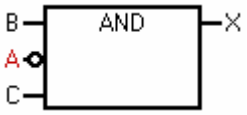
Từ biểu thức đã đơn giản ta được sơ đồ logic sau và biểu diễn ở LAD, FBD, STL (hình 8.5).



LAD



FBD



STL

```
LD   B
AN   A
A    C
=    X
```

Hình 8.5: Sơ đồ logic và chương trình trong PLC

Tóm lại, ta sẽ thu được các biểu thức đại số Bool từ việc mô tả yêu cầu công nghệ hoặc một sơ đồ mạch hoặc một sơ đồ LAD. Các biểu thức có thể được đơn giản bằng cách sử dụng các định lý của đại số Bool. Và sau đó từ biểu thức này ta có thể chuyển thành ngôn ngữ LAD, FBD hay STL trong PLC. Khi đơn giản các biểu thức đại số Bool ta cần chú ý một số quy tắc cơ bản sau:

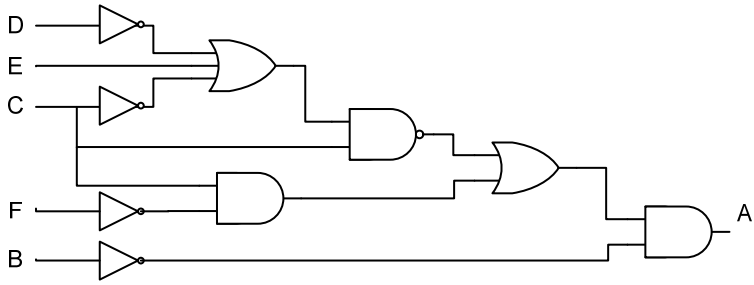
- Loại bỏ các cổng NOT không cần thiết. Thông thường có thể thực hiện bằng cách thay thế các cổng NAND và NOR bằng một biểu thức đơn giản hơn sử dụng định lý DeMorgan.
- Loại bỏ các công thức phức tạp như XOR.

Các qui tắc này có thể được mô tả như ví dụ sau:

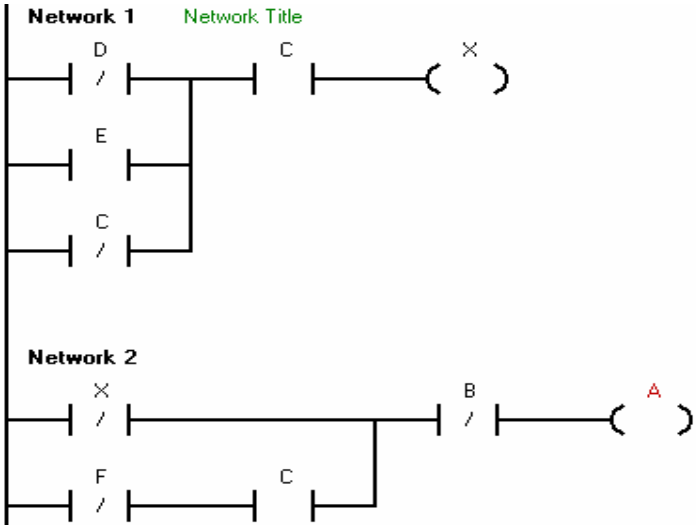
Ví dụ 8.3: Cho biểu thức điều khiển:

$$A = \overline{B} \cdot \overline{(C \cdot (\overline{D} + E + \overline{C})} + \overline{F} \cdot C$$

Biểu thức trên có thể được biểu diễn ở dạng sơ đồ mạch logic như sau:



Biểu diễn ở LAD:



Hình 8.6: Minh họa các qui tắc đơn giản khi chuyển đổi biểu thức đại số Bool sang LAD

8.3.1 Các kỹ thuật đại số Bool

Có một vài kỹ thuật chung được sử dụng khi đơn giản công thức. Các kỹ thuật này được biểu diễn ở hình 8.7.

$A + C\overline{A} = A + C$ Chứng minh: $A + C\overline{A}$

$$\Leftrightarrow (A + C)(A + \overline{A})$$
$$\Leftrightarrow (A + C)(1)$$
$$\Leftrightarrow A + C$$

$AB + A = A$ Chứng minh: $AB + A$

$\Leftrightarrow AB + A1$

$\Leftrightarrow A(B + 1)$

$\Leftrightarrow A(1)$

$\Leftrightarrow A$

$\overline{A + B + C} = \overline{A}\overline{B}\overline{C}$ Chứng minh: $\overline{A + B + C}$

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

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

$\Leftrightarrow (\overline{A}\overline{B})\overline{C}$

$\Leftrightarrow \overline{A}\overline{B}\overline{C}$

Hình 8.7: Các kỹ thuật đại số Bool

8.4 Các dạng logic chung

Khi biết một tập các dạng logic đơn giản sẽ cung cấp cho người thiết kế giải quyết các chiến lược điều khiển. Các dạng sau được cung cấp để sử dụng trực tiếp hoặc ý tưởng khi thiết kế.

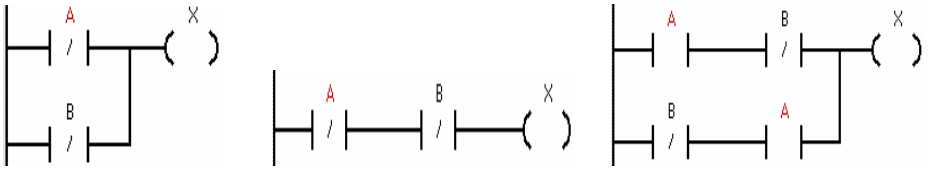
8.4.1 Dạng cổng phức

Tổng cộng có 16 loại cổng logic khác nhau có 2 ngõ vào. Dạng đơn giản nhất là AND và OR, các cổng khác là các cổng phức. Ba cổng phức thông dụng được thảo luận trước đây là NAND, NOR và XOR. Các cổng này có thể được biểu diễn thành dạng đơn giản hơn chỉ với các cổng AND và OR tương ứng ở sơ đồ LAD trong PLC biểu diễn ở hình 8.8.

NAND **NOR** **XOR**

$X = \overline{A \cdot B}$ $X = \overline{A + B}$ $X = A \oplus B$

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

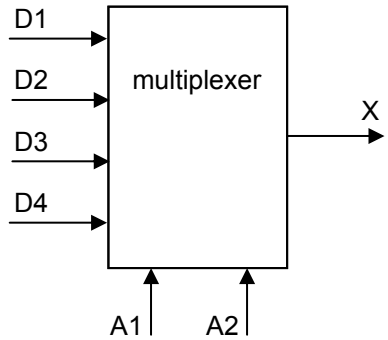


Hình 8.8: Chuyển đổi các chức năng logic phức

8.4.2 Multiplexers

Multiplexers là sự đa hợp các thiết bị được kết nối với một thiết bị đơn. Nó rất thông dụng trong các hệ thống điện thoại. Một *chuyển mạch* điện thoại được sử dụng để xác định điện thoại nào sẽ được kết nối.

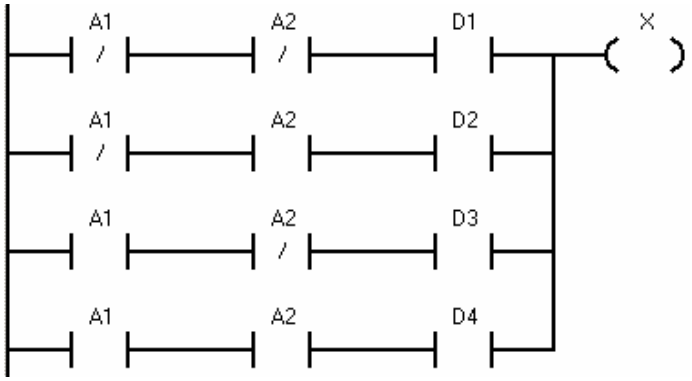
Hình 8.9 là một bộ multiplexer. Ngõ ra X sẽ được kết nối với một trong 4 ngõ vào D1, D2, D3 hoặc D4 tùy thuộc vào giá trị của các ngõ A1 và A2.



A1	A2	X
0	0	D1
0	1	D2
1	0	D3
1	1	D4

Hình 8.9: Một Multiplexer

Dạng multiplexer được biểu diễn ở LAD có thể trình diễn ở hình 8.10.



Hình 8.10: Một Multiplexer biểu diễn ở Ladder Logic

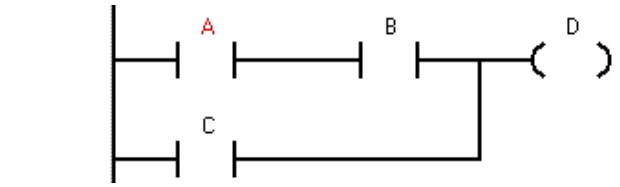
8.5 Một số ví dụ thiết kế đơn giản với đại số bool

Các trường hợp sau đây minh họa các vấn đề logic tổ hợp khác nhau và các giải pháp có thể thực hiện. Hãy đọc kỹ mô tả trước khi xem lời giải.

8.5.1 Các chức năng logic cơ bản

Yêu cầu 1: Viết một chương trình sao cho ngõ ra D ở mức logic "1" khi công tắc A và B đóng lại hoặc khi công tắc C được đóng.

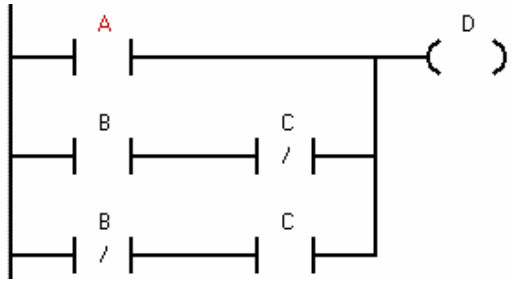
Giải quyết: $D = (A \cdot B) + C$



Hình 8.11: Chương trình được viết ở LAD

Yêu cầu 2: Viết một chương trình sao cho ngõ ra D ở mức logic "1" khi nút ấn A được ấn, hoặc chỉ B hoặc chỉ C được ấn.

Giải quyết: $D = A + (B \oplus C)$

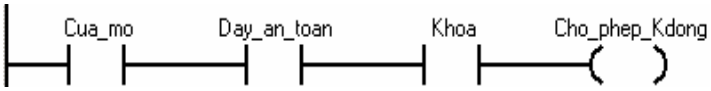


Hình 8.12: Chương trình được viết ở LAD

8.5.2 Hệ thống an toàn xe hơi

Yêu cầu: Viết chương trình ở LAD cho một hệ thống an toàn cửa xe hơi/dây an toàn chỗ ngồi. Khi cửa mở, hoặc dây an toàn chưa được thắt thì việc khoá khởi động không thể thực hiện được. Nếu tất cả được thực hiện thì khóa có thể khởi động được động cơ.

Giải quyết:



Hình 8.13: Chương trình hệ thống an toàn xe viết ở LAD

8.5.3 Quay phải/trái động cơ

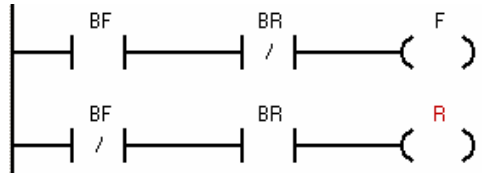
Yêu cầu: thiết kế một bộ điều khiển động cơ có một nút nhấn quay phải và một nút nhấn quay trái. Các ngõ ra quay phải và trái sẽ chỉ ở "1" khi một trong các nút nhấn được ấn. Khi cả hai nút nhấn được ấn thì động cơ không làm việc.

Giải quyết:

$F = BF \cdot \overline{BR}$

$R = \overline{BF} \cdot BR$

Ở đây:
F = Động cơ quay phải
R = Động cơ quay trái
BF = Nút nhấn quay phải
BR = Nút nhấn quay trái



Hình 8.14: Chương trình quay phải, trái viết ở LAD

8.5.4 Cảnh báo trộm

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

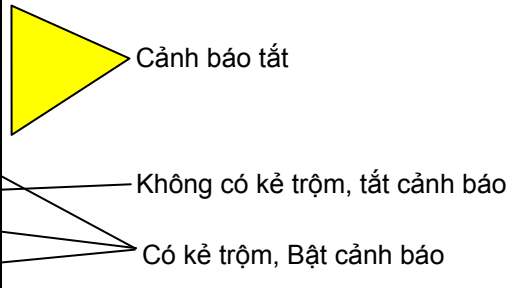
Ký hiệu	***Địa chỉ***	***Chú thích***
A	Q0.0	Đèn và cảnh báo, ON="1"
W	I0.0	Cảm biến cửa sổ/cửa chính, thường đóng
M	I0.1	Cảm biến chuyển động, thường hở
S	I0.2	Công tắc kích hoạt cảnh báo, ON="1"

Hoạt động cơ bản của cảnh báo có thể được mô tả theo qui tắc:

1. Nếu cảnh báo là "ON", kiểm tra cảm biến.
2. Nếu cảm biến cửa sổ/cửa chính bị ngắt, bật âm thanh cảnh báo và đèn báo sáng.

Bước kế tiếp là xác định công thức điều khiển. Trong trường hợp này có 3 ngõ vào khác nhau và 1 ngõ ra, bảng sự thật được trình bày ở hình 8.15.

Input			Output
S	M	W	A
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1



Hình 8.15: Bảng sự thật cảnh báo trộm

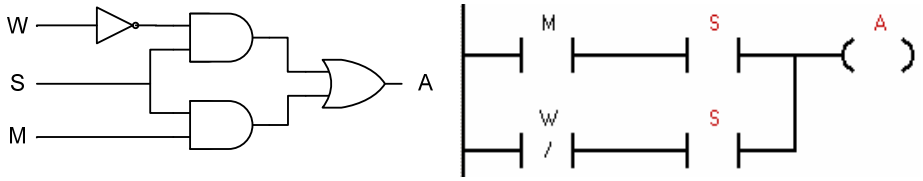
Biểu thức Boolean và đơn giản được cho ở hình 8.17 được viết từ bảng sự thật hình 8.16.

$$A = (S \cdot \overline{M} \cdot \overline{W}) + (S \cdot M \cdot \overline{W}) + (S \cdot M \cdot W)$$

$$A = S \cdot (\overline{M} \cdot \overline{W} + M \cdot \overline{W} + M \cdot W)$$

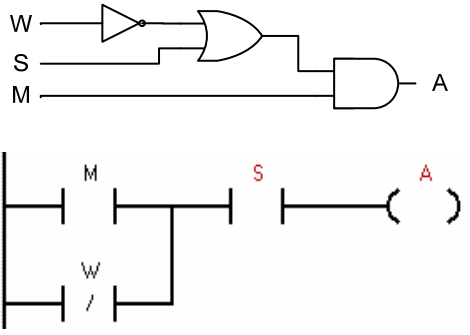
$$A = S \cdot ((\overline{M} \cdot \overline{W} + M \cdot \overline{W}) + (M \cdot \overline{W} + M \cdot W))$$

$$A = (S \cdot \overline{W}) + (S \cdot M) = S \cdot (\overline{W} + M)$$



Hình 8.16: Biểu thức Bool và được thực hiện với LAD

Công thức và mạch cho ở hình trên cũng có thể được đơn giản như hình 8.17.



Hình 8.17: Sơ đồ mạch theo biểu thức Bool đơn giản và được thực hiện với LAD

8.6 Biểu đồ Karnaugh

8.6.1 Giới thiệu

Bảng Karnaugh cho phép chúng ta chuyển đổi một bảng sự thật thành biểu thức Boolean đơn giản mà không sử dụng đại số Bool. Trong mục 8.5.4 của chương này có một ví dụ về cảnh báo trộm. Hình 8.18 là bảng sự thật của nó với một ngõ vào báo yên tĩnh được thêm vào.

Đã cho: A, W, M, S như trước đây, tức là:

Ký hiệu	*Địa chỉ*	*Chú thích*
A	Q0.0	Đèn và cảnh báo, ON=”1”
W	I0.0	Cảm biến cửa sổ/cửa chính, thường đóng

M	I0.1	Cảm biến chuyển động, thường hở
S	I0.2	Công tắc kích hoạt cảnh báo, ON="1"

Và:

Q = Báo yên tĩnh (0 = yên tĩnh)

Bước 1: Vẽ bảng sự thật

Bảng sự thật của mạch cảnh báo trộm nhưhình 8.18. Thay vì chuyển đổi trực tiếp bảng này thành biểu thức, thì ta đặt vào một bảng được chỉ ở hình 8.19. Dòng và cột được chọn từ các biến ngõ vào.

Việc quyết định các biến nào sử dụng cho các dòng hoặc các cột có thể tùy ý và các bảng sẽ trông khác nhau nhưng vẫn sẽ cho một kết quả giống nhau. Đối với các biến ở cả hai dòng và cột thì được sắp xếp theo thứ tự chỉ giá trị của bit sử dụng NOT. Trình tự không phải là nhị phân, nhưng được tổ chức để chỉ có một bit thay đổi tại một thời điểm. Như vậy trình tự của bit là 00, 01, 11, 10. Bước này rất quan trọng. Kế tiếp là đưa các giá trị là "1" trong bảng sự thật vào bảng Karnaugh. Giá trị "0" cũng có thể được đưa vào nhưng không cần thiết.

S	M	W	Q	A
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Hình 8.18: Bảng sự thật mạch cảnh báo trộm

Trong ví dụ, ba giá trị "1" từ bảng sự thật được đưa vào trong bảng.

Bước 2: Chia các biến vào.

Ở đây chọn SQ và MW

Bước 3: Vẽ bảng Karnaugh dựa vào các biến vào

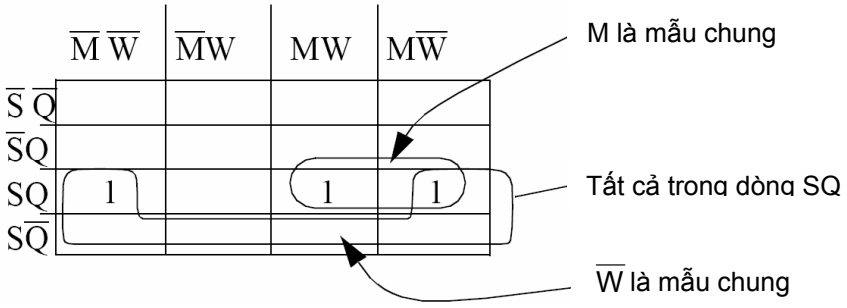
	$\overline{M}\overline{W}(=00)$	$\overline{M}W(=01)$	$MW(=11)$	$M\overline{W}(=10)$
$\overline{S}\overline{Q}(=00)$				
$\overline{S}Q(=01)$				
$SQ(=11)$	1		1	1
$S\overline{Q}(=10)$				

Hình 8.19: Bảng Karnaugh

Khi các bit được nhập vào bảng Karnaugh sẽ có một vài mẫu rõ ràng. Các mẫu tiêu biểu này có phần nào đối xứng. Hình 8.20 có hai mẫu được khoanh tròn. Trong trường hợp này, một mẫu có hai bit đứng kề nhau. Mẫu thứ hai thì khó nhìn thấy hơn vì các bit nằm ở bìa bên phải và trái của cột.

Sau đó các mẫu có thể được chuyển thành biểu thức Boolean. Để thực hiện trước tiên ta quan sát các mẫu đặt ở dòng thứ ba cho nên biểu thức sẽ được AND với SQ. Kế tiếp là tìm bit chung trong hai mẫu. Ta thấy trong mẫu một có M chung, mẫu 2 có $\overline{W}$ chung. Những cái này bây giờ có thể tổ hợp thành công thức. Cuối cùng công thức được chuyển thành sơ đồ LAD.

<u>Bước 4:</u> Tìm kiếm mẫu trong bảng

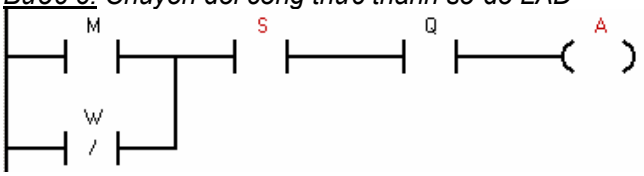


Hình 8.20: Khoanh mẫu

<u>Bước 5:</u> Viết thành công thức sử dụng các mẫu

$$A = S \cdot Q \cdot (M + \overline{W})$$

<u>Bước 6:</u> Chuyển đổi công thức thành sơ đồ LAD

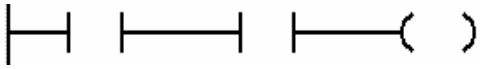


Hình 8.21: Chuyển đổi biểu thức thành sơ đồ LAD

Bảng Karnaugh là một phương pháp có thể được chọn để đơn giản biểu thức thay cho đại số Bool. Nó giúp cho người học dễ dàng hơn trong việc đơn giản các biểu thức. Ở ví dụ trên chỉ có 4 biến, như vậy chỉ có hai biến ở dòng và hai biến ở cột. Nếu có nhiều biến hơn vẫn có thể sử dụng. Ví dụ nếu có năm biến ngõ vào thì ta có thể sử dụng ba biến cho dòng hoặc cho cột với các mẫu là 000, 001, 011, 010, 110, 111, 101, 100. Nếu có nhiều hơn một ngõ ra, thì ta tạo bảng Karnaugh cho mỗi ngõ ra.

8.7 Câu hỏi và bài tập

BT 8.1: Cổng logic được biểu diễn ở ngôn ngữ LAD cho ở dưới đây là cổng AND hay OR?



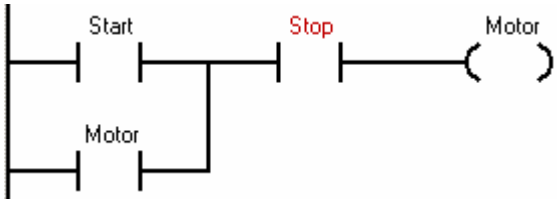
BT 8.2: Vẽ một sơ đồ hình thang với ngõ ra D là "1" khi công tắc A và công tắc B được đóng hoặc khi công tắc C được đóng.

BT 8.3: Vẽ một sơ đồ hình thang với ngõ ra D là "1" khi nút nhấn A được ấn hoặc B hoặc C được ấn.

BT 8.4:

a) Giải thích tại sao nút nhấn stop phải là thường đóng và nút nhấn start phải là thường hở.

b) Xem xét một trường hợp một ngõ vào PLC được nối với nút nhấn thường đóng làm nút nhấn stop. Tiếp điểm được sử dụng trong ngôn ngữ LAD là thường hở như được cho ở dưới. Tại sao cả hai là không giống nhau? (ví dụ cùng là NC hoặc NO)



BT 8.5:. Tạo một chương trình đơn giản ở ngôn ngữ LAD theo bảng sự thật được cho ở dưới với ngõ ra ở trạng thái "ON" khi các nút nhấn tương ứng được ấn.

INPUT	OUTPUT							
	A	B	C	D	E	F	G	H
Ngõ vào X ON	1	0	1	0	1	0	1	1
Ngõ vào Y ON	1	0	0	0	0	1	0	1
Ngõ vào Z ON	1	1	1	0	1	0	0	1

BT 8.6: Chuyển đổi biểu thức đại số Bool sau thành chương trình ở ngôn ngữ LAD đơn giản nhất có thể được.

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

BT 8.7: Đơn giản các biểu thức sau:

a) $A(B + AB)$ b) $\overline{A(\overline{B + AB})}$

c) $\overline{A}(B + AB)$ d) $\overline{\overline{A}(\overline{B + AB})}$

BT 8.8: Đơn giản các biểu thức sau:

a) $\overline{(A + B)} \cdot \overline{(A + \overline{B})}$

b) $ABCD + \overline{A}BCD + ABC\overline{D} + AB\overline{C}\overline{D}$

BT 8.9: Đơn giản biểu thức Boolean sau:

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

BT 8.10: Cho biểu thức Boolean

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

a) Vẽ sơ đồ mạch số

b) sơ đồ hình thang (không tối giản),

c) Đơn giản biểu thức.

BT 8.11: Đơn giản biểu thức đại số Boolean sau và viết chương trình ở ngôn ngữ LAD tương ứng.

$$Y = \overline{\overline{(AB\overline{C}D + AB\overline{C}\overline{D} + \overline{A}BCD + \overline{A}\overline{B}CD)} + D}$$

BT 8.12: Cho biểu thức đại số sau:

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

a) Viết thành sơ đồ logic khi chưa đơn giản biểu thức.

b) Đơn giản biểu thức.

c) Viết thành chương trình ở ngôn ngữ LAD theo biểu thức đã đơn giản.

BT 8.13: Cho bảng sự thật sau

a)Chỉ ra tổ hợp nào cho kết quả là 1.

b) Viết kết quả ở a) thành biểu thức đại số Bool.

c) Đơn giản biểu thức Bool ở b)

A	B	C	D	Kết quả
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

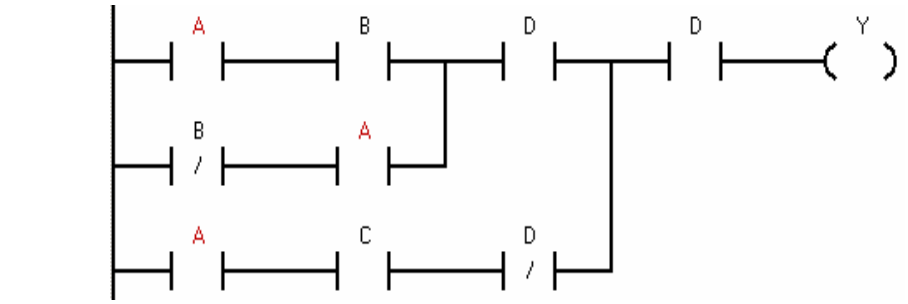
BT 8.14: Đơn giản biểu thức sau thành đơn giản nhất và viết thành chương trình ở ngôn ngữ LAD.

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

BT 8.15: Đơn giản biểu thức sau sử dụng đại số Bool và viết thành chương trình ở ngôn ngữ LAD tương ứng.

$$X = \overline{(A + B \cdot \overline{A}) + \overline{(C + D + E\overline{C})}}$$

BT 8.16: Chuyển đổi chương trình biểu diễn ở LAD sau thành biểu thức đại số. Sau đó đơn giản nó và chuyển lại ở ngôn ngữ LAD.

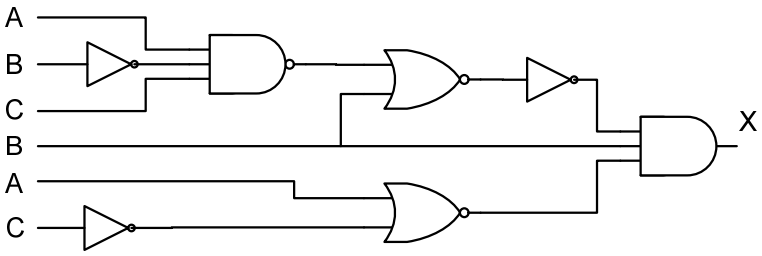


BT 8.17: Cho sơ đồ mạch logic như hình vẽ

a) Viết thành biểu thức ở mạch logic đã cho.

b) Đơn giản biểu thức này.

c) Vẽ lại sơ đồ mạch đơn giản hơn theo câu b).



BT 8.18: Cho một hệ thống được mô tả theo biểu thức sau:

$$X = A + (B \cdot (\overline{A} + C) + C) + A \cdot B \cdot (\overline{D} + \overline{E})$$

a) Đơn giản biểu thức sử dụng đại số Bool.

b) Thực hiện sơ đồ mạch số theo biểu thức ban đầu và biểu thức đã được đơn giản ở câu a).

c) Viết thành chương trình ở ngôn ngữ LAD theo biểu thức ban đầu và biểu thức đã được đơn giản ở câu a)

BT 8.19: Đơn giản biểu thức đã cho và sau đó viết thành chương trình ở ngôn ngữ LAD và sơ đồ mạch số theo biểu thức ban đầu và biểu thức đã đơn giản.

$$A + (\overline{B} + \overline{C} + \overline{D}) \cdot (B + \overline{C}) + A \cdot B \cdot (\overline{C} + \overline{D})$$

BT 8.20: Lập bảng Karnaugh theo bảng sự thật dưới đây.

A	B	C	D	Kết quả
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

BT 8.21: Sử dụng bảng Karnaugh để đơn giản bảng sự thật sau và viết thành chương trình ở ngôn ngữ LAD.

A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

BT 8.22: Viết ra biểu thức đơn giản nhất đối với bảng Karnaugh được cho dưới đây

	CD	$C\overline{D}$	$\overline{C}\overline{D}$	$\overline{C}D$
AB	1	0	0	1
$A\overline{B}$	0	0	0	0
$\overline{A}\overline{B}$	0	0	0	0
$\overline{A}B$	0	1	1	0

BT 8.23: Cho bảng sự thật ở hình BT 8.23 và viết thành chương trình PLC ở ngôn ngữ LAD với sự trợ giúp bằng kỹ thuật đơn giản biểu thức là bảng Karnaugh hay đại số Bool.

BT 8.24: Kiểm tra bảng sự thật ở hình BT 8.24 và viết thành chương trình PLC ở ngôn ngữ LAD sử dụng bảng Karnaugh.

BT 8.26: Cho bảng sự thật ở hình BT 8.25 với các ngõ vào A, B, C và D và ngõ ra X. Chuyển nó thành chương trình PLC ở LAD sử dụng bảng Karnaugh.

BT 8.25: Tìm biểu thức Boolean đơn giản nhất đối với bảng Karnaugh được cho ở hình BT 8.26 mà không sử dụng đại số Bool. Viết chương trình ở LAD.

A	B	C	D	X	Y
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	0	0
0	0	1	1	0	0
0	1	0	0	0	0
0	1	0	1	0	0
0	1	1	0	0	1
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	0	0
1	0	1	1	0	0
1	1	0	0	1	0
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	1	0	1

Hình BT 8.23

A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Hình BT 8.24

A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Hình BT 8.25

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

Hình BT 8.27

	$\bar{A}\bar{B}\bar{C}$	$\bar{A}\bar{B}C$	$\bar{A}B\bar{C}$	$\bar{A}BC$	$A\bar{B}\bar{C}$	ABC	$A\bar{B}C$	$A\bar{B}\bar{C}$
$\bar{D}\bar{E}$	1	1	0	1	0	0	0	0
$\bar{D}E$	1	1	0	0	0	0	0	0
DE	1	1	0	0	0	0	0	0
$D\bar{E}$	1	1	0	1	0	0	0	0

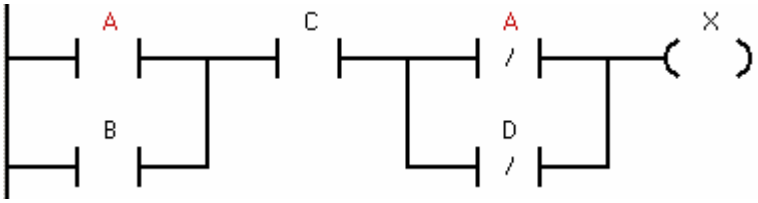
Hình BT 8.26

BT 8.27: Cho bảng sự thật như hình BT 8.27

a) Tìm biểu thức đại số Bool sử dụng bảng Karnaugh.

b) Vẽ sơ đồ LAD sử dụng bảng sự thật (không phải biểu thức Boolean).

BT 8.28: Chuyển đổi sơ đồ LAD sau thành bảng Karnaugh.



BT 8.29:

a) Xây dựng bảng sự thật cho các vấn đề sau đây:

i) Có 3 nút nhấn A, B, C.

ii) Ngõ ra là "1" nếu bất kỳ hai nút nhấn nào được ấn.

iii) Nếu C được ấn thì ngõ ra sẽ luôn luôn "1".

b) Viết thành biểu thức Bool.

c) Viết thành biểu thức Boolean sử dụng bảng Karnaugh.

BT 8.30: Viết ra biểu thức Boolean đơn giản nhất đối với bảng Karnaugh dưới đây

a) Bằng đồ thị.

b) Bằng đại số Boolean.

	$\overline{A}\,\overline{B}$	$A\overline{B}$	AB	$\overline{A}B$
CD	1			1
$C\overline{D}$		1	1	
$\overline{C}\,\overline{D}$				
$\overline{C}D$	1			1

BT 8.31: Xem xét biểu thức boolean sau:

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

a) Biểu thức Boolean này có thể được chuyển trực tiếp thành LAD. Giải thích nếu cần thiết, thực hiện bất kỳ các thay đổi được yêu cầu để có thể chuyển thành LAD.

b) Viết ra ở LAD, dựa vào kết quả ở bước a).

c) Đơn giản biểu thức sử dụng đại số Bool và viết ra LAD mới.

d) Viết bảng Karnaugh đối với biểu thức Boolean, và cho biết nó có thể được sử dụng để thu được biểu thức Bool đơn giản như thế nào.

9 Bộ định thời (Timer)

9.1 Giới thiệu

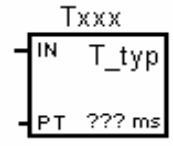
Bộ định thời được sử dụng trong các yêu cầu điều khiển cần trì hoãn về thời gian. Đây là phần tử chức năng cơ bản của các bộ PLC và rất thường được sử dụng trong các chương trình điều khiển. Chẳng hạn như một băng tải khi có tín hiệu hoạt động sẽ chạy trong 10s rồi dừng lại, một van khí nén cần có điện trong 5s, nguyên liệu cần trộn trong thời gian 10 phút…Các PLC S7-200 có 256 Timer có địa chỉ từ T0 đến T255, chia làm 3 loại (*xem thêm chương 4 Bộ điều khiển lập trình PLC S7-200*) :

+ Timer đóng mạch chậm TON (On-delay Timer).
+ Timer đóng mạch chậm có nhớ TONR (Retentive On-delay Timer).
+ Timer ngắt mạch chậm TOF (Off-delay Timer).

Khi sử dụng một timer chúng ta cần phải xác định các thông số sau:

- Loại timer (TON, TONR hay TOF)
- Độ phân giải của Timer. Có 3 độ phân giải là: 1ms, 10ms và 100ms
- Số của timer sẽ sử dụng, ví dụ T0, T37..cần tra bảng để biết loại timer sử dụng tương ứng với các số nào.
- Khai báo hằng số thời gian tương ứng với thời gian cần trì hoãn dựa vào độ phân giải của timer.
- Tín hiệu cho phép bắt đầu tính thời gian.

Ký hiệu chung của Timer S7-200 biểu diễn ở LAD như sau:



Với:

Txxx: Ký hiệu và số thứ tự của timer, ví dụ: T37

IN: Ngõ vào bit, cho phép timer hoạt động

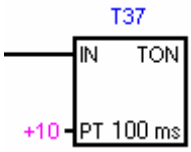
PT: Ngõ vào số Integer, hằng số thời gian.

T_typ: Cho biết loại Timer. Có thể là TON, TONR hay TOF

???ms: Báo độ phân giải của timer, tự động xuất hiện theo Txxx.

Thời gian trì hoãn = [PT] x [???ms].

Ví dụ ta có



Đây là loại On-delay timer, có tên gọi là T37, có độ phân giải là 100ms. Thời gian trì hoãn là : 10 x 100ms = 1s.

9.2 Timer đóng mạch chậm TON

Các Timer này được sử dụng khi có các yêu cầu trì hoãn một khoảng thời gian. Giá trị hiện hành của TON bị xóa khi ngõ vào IN ở logic "0".

On-Delay Timer (TON) thực hiện đếm thời gian khi ngõ vào IN ở mức logic "1". Khi giá trị hiện hành (Txxx) lớn hơn hoặc bằng thời gian đặt trước PT (preset time), thì Timer Bit ở logic "1". Giá trị hiện hành của TON bị xóa khi ngõ vào IN ở logic "0". Timer tiếp tục đếm dù đã đạt đến giá trị đặt PT, và dừng lại khi đếm đến giá trị max. 32767.

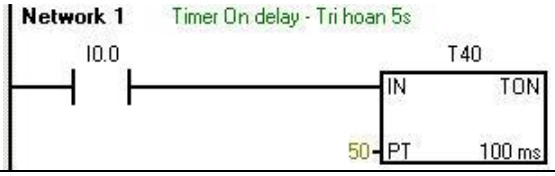
Để xóa timer, có thể sử dụng lệnh Reset (R). Lệnh Reset sẽ làm cho Timer Bit ở mức logic "0" và giá trị hiện hành của timer (Timer Current) =0.

Có 192 timer TON/TOF trong S7-200 được phân chia theo độ phân giải như ở bảng sau:

Số Timer	Độ phân giải	Thời gian trì hoãn tối đa
T32, T96	1ms	32,767s
T33 … T36, T97 … T100	10ms	327,67s
T37 … T63, T101 … T255	100ms	3276,7s

Chú ý: Vì TON và TOF sử dụng cùng số timer, nên không thể đặt cho cả hai có cùng số Timer. Ví dụ đã đặt TON là T37 thì không được đặt TOF là T37.

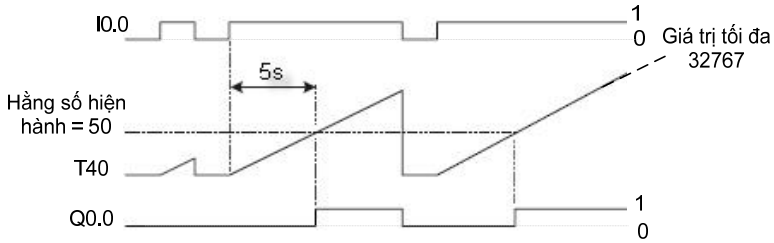
Ví dụ: Bật công tắc I0.0 (NO) thì sau 5s ngõ ra Q0.0 lên mức 1.



Dùng Timer T40, độ phân giải 100ms, hằng số thời gian 50. Thời gian trì hoãn = 50x100ms=5s

Tiếp điểm T40 đóng lại sau 5s.

Giản đồ thời gian:



Qua giản đồ trên ta nhận thấy để timer TON trì hoãn được hết thời gian đặt trước (ví dụ 5s) thì trạng thái tín hiệu tại ngõ vào IN cần được duy trì ở mức 1 trong suốt khoảng thời gian này. Nếu sau 5s mà ngõ vào IN vẫn duy trì ở mức 1 thì giá trị hằng số thời gian trong timer sẽ tiếp tục tăng cho tới khi đạt giá trị tối đa là 32767.

Để lấy TON, ta nhấp chuột vào dấu (+) ở biểu tượng Timers trong cây lệnh. Sau đó trỏ chuột vào TON giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập số Timer cho TON, điều kiện cho ngõ vào IN và giá trị ở PT theo mong muốn.

9.3 Timer đóng mạch chậm có nhớ TONR

Các Timer này được sử dụng khi cần tích lũy một số khoảng thời gian rời rạc. Giá trị hiện hành TONR chỉ có thể bị xóa bằng lệnh Reset (R).

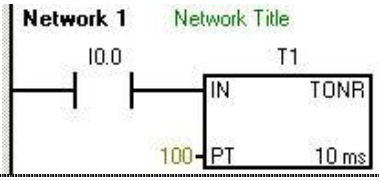
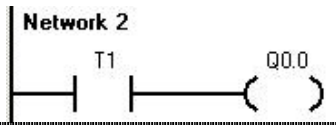
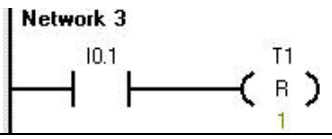
Timer đóng mạch chậm có nhớ TONR (Retentive On-Delay Timer) thực hiện đếm thời gian khi ngõ vào IN ở mức logic "1". Khi giá trị hiện hành (Txxx) lớn hơn hoặc bằng thời gian đặt trước PT (preset time), thì Timer Bit ở logic "1". Giá trị hiện hành của TONR được giữ lại khi ngõ vào IN ở logic "0". TONR được sử dụng để tích lũy thời gian cho nhiều chu kỳ ngõ vào IN ở mức "1". Timer này vẫn tiếp tục đếm sau khi đã đạt đến giá trị đặt trước và dừng lại ở giá trị max. 32767.

Để xóa giá trị hiện hành của TONR và Timer Bit, ta sử dụng lệnh Reset (R).

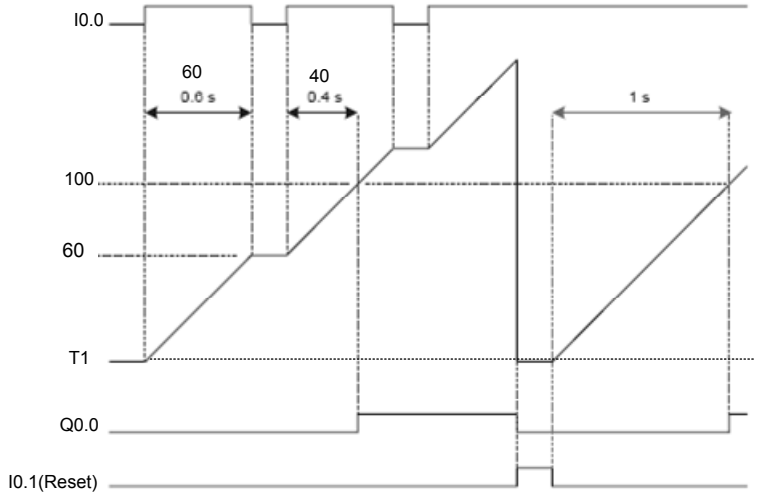
Có 64 timer TONR trong S7-200 được phân chia theo độ phân giải như ở bảng sau:

Số Timer	Độ phân giải	Thời gian trì hoãn tối đa
T0, T64	1 ms	32,767 s
T1 … T4, T65 …T68	10 ms	327,67 s
T5 … T31, T69 … T95	100 ms	3276,7 s

Ví dụ: Xét đoạn chương trình

	Tín hiệu I0.0 kích hoạt timer TONR T1 có độ phân giải 10ms (thời gian = 100 x 10ms = 1s)
	Sau 1 s ngõ ra Q0.0 lên mức 1
	Tín hiệu I0.1 Reset timer T1

Giản đồ thời gian:



Để lấy TONR, ta nhấp chuột vào dấu (+) ở biểu tượng Timers trong cây lệnh. Sau đó trỏ chuột vào TONR giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập số Timer cho TONR, điều kiện cho ngõ vào IN và giá trị ở PT theo mong muốn.

9.4 Timer mở mạch chậm TOF

Sử dụng timer này khi cần trì hoãn thêm một khoảng thời gian rồi mới tắt ngõ ra kể từ khi tín hiệu ngõ vào IN xuống "0". Timer TOF chỉ thực hiện đếm thời gian khi IN chuyển từ "1" xuống "0".

Khi ngõ vào IN của Off-Delay Timer (TOF) ở logic "1", thì Timer Bit ngay lập tức được đặt lên mức logic "1" và giá trị hiện hành được xóa về 0. Khi ngõ

vào IN xuống "0", thì timer đếm cho đến khi thời gian trôi qua đạt đến giá trị thời gian đặt trước. Khi đạt đến giá trị đặt trước, Timer Bit được đặt về "0" và giá trị hiện hành dừng đếm. Nếu ngõ vào IN ở "0" trong khoảng thời gian ngắn hơn giá trị đặt trước, thì Timer Bit giữ ở "1".

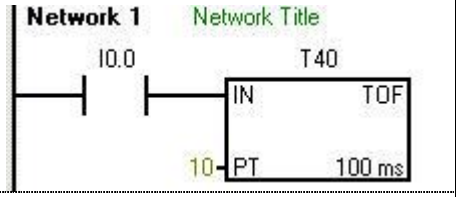
Để xóa timer, có thể sử dụng lệnh Reset (R). Lệnh Reset sẽ làm cho Timer Bit ở mức logic "0" và giá trị hiện hành của timer (Timer Current) =0.

Có 192 timer TON/TOF trong S7-200 được phân chia theo độ phân giải như ở bảng sau:

Số Timer	Độ phân giải	Thời gian trì hoãn tối đa
T32, T96	1ms	32,767s
T33 … T36, T97 … T100	10ms	327,67s
T37 … T63, T101 … T255	100ms	3276,7s

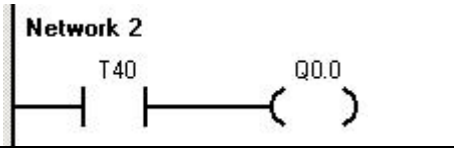
Chú ý: Vì TON và TOF sử dụng cùng số timer, nên không thể đặt cho cả hai có cùng số Timer. Ví dụ đã đặt TON là T37 thì không được đặt TOF là T37.

Ví dụ: Xét đoạn chương trình



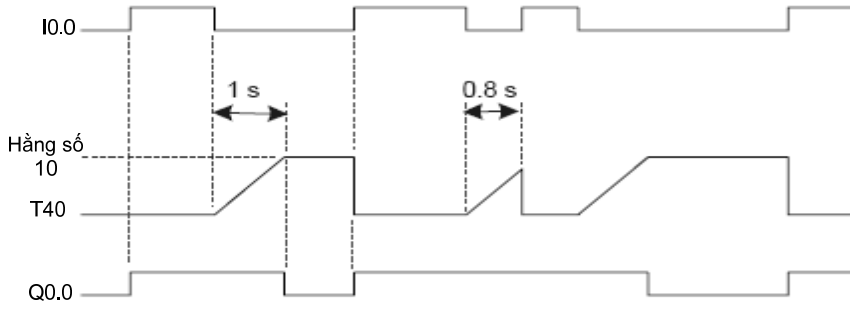
I0.0 chuyển trạng thái từ mức 1 xuống mức 0 sẽ kích hoạt timer off delay tính giờ.

Thời gian = 10 x 100ms = 1s



Sau 1s kể từ khi tín hiệu tại I0.0 chuyển từ 1 xuống 0 ngõ ra Q0.0 sẽ xuống mức 0

Giản đồ thời gian:

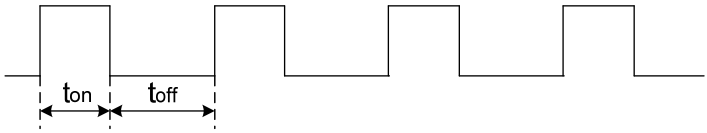


Để lấy TOF, ta nhấp chuột vào dấu (+) ở biểu tượng Timers trong cây lệnh. Sau đó trỏ chuột vào TOF giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập số Timer cho TOF, điều kiện cho ngõ vào IN và giá trị ở PT theo mong muốn.

9.5 Ứng dụng Timer

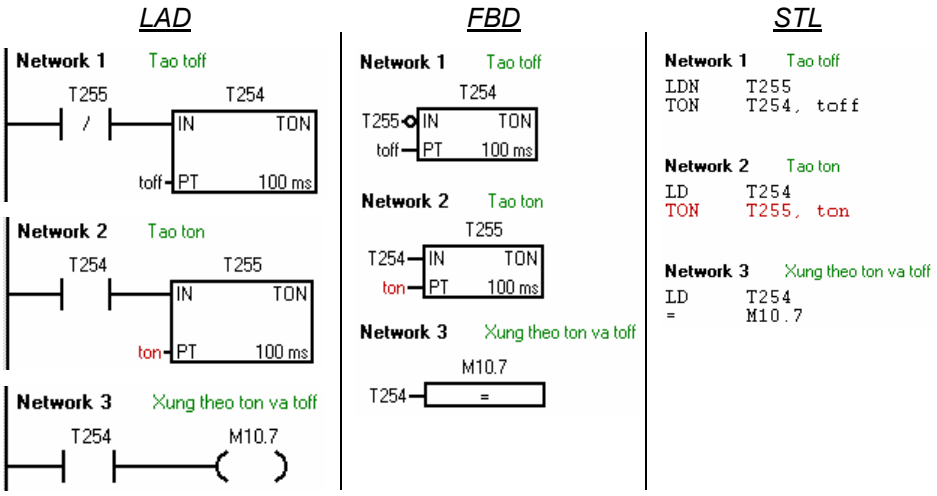
9.5.1 Tạo xung có tần số theo mong muốn

Viết chương trình tạo xung theo mong muốn để sử dụng vào các mục đích khác nhau theo giản đồ xung sau:



Để thực hiện, sử dụng 2 timer TON khóa chéo nhau. Tùy thuộc vào xung cần lấy có thời gian t_{on} và t_{off} là bao nhiêu mà ta có thể chọn số timer TON phù hợp. Trong ứng dụng này, chọn T254 và T255 làm timer tạo xung và thời gian thì tùy theo người sử dụng mong muốn cho vào giá trị t_{on} và t_{off} ở ngõ PT của timer (chú ý thời gian = [PT]x100ms). Xung được lưu ở bit M10.7.

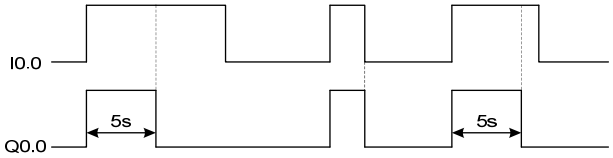
Chương trình:



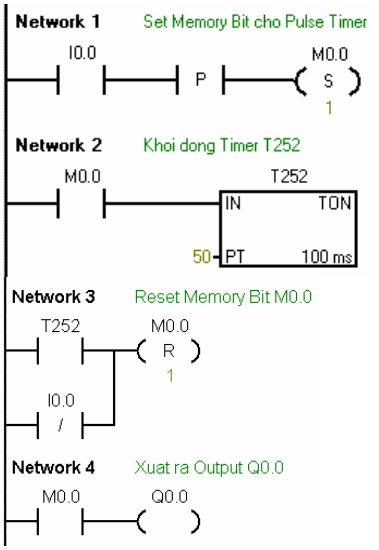
9.5.2 Tạo Timer xung và timer xung có nhớ

9.5.2.1 Timer xung (Pulse timer)

Timer xung sẽ cho ngõ ra là một xung khi tín hiệu vào ở mức logic "1" có thời gian lớn hơn hay bằng thời gian đặt ở timer xung. Để dễ hình dung xem giản đồ thời gian của chương trình tạo timer xung với ngõ ra timer là Q0.0, ngõ vào tín hiệu là I0.0, thời gian xung là 5s như sau:



LAD

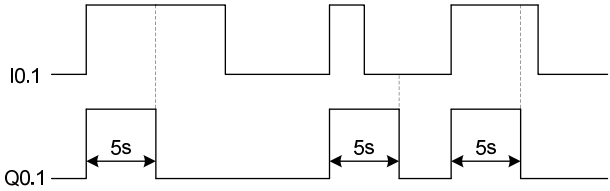


STL

```
Network 1    Set Memory Bit cho Pulse Timer
LD      I0.0        // load I0.0.
EU                  // Khi co canh len
                    // tai ngo vao I0.0
S       M0.0, 1     // set memory bit M0.0.

Network 2    Khoi dong Timer T252
LD      M0.0        // Load M0.0.
                    // Neu M0.0 duoc set,
TON     T252, 50    // khoi dong timer T252
                    // voi PT la 50 (5s)

Network 3    Reset Memory Bit M0.0
LD      T252    // Load timer T252.
                // Khi Preset Time = Current Time
ON      I0.0    // hoac neu I0.0 khong ="1" nua,
R       M0.0, 1 // thi reset M0.0.

Network 4    Xuat ra Output Q0.0
LD      M0.0    // Load M0.0.
                // Neu M0.0 duoc set,
=       Q0.0    // set ngo ra Q0.0.
```

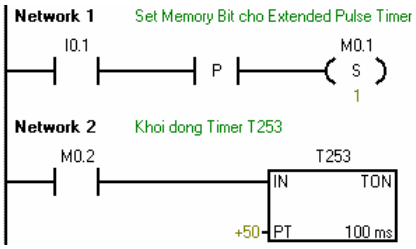
9.5.2.2 Timer xung có nhớ (Extended Pulse timer)

Timer xung sẽ cho ngõ ra là một xung khi có một xung tín hiệu vào. Để dễ hình dung xem giản đồ thời gian của chương trình tạo timer xung với ngõ ra timer là Q0.1, ngõ vào tín hiệu là I0.1, thời gian xung là 5s như sau:



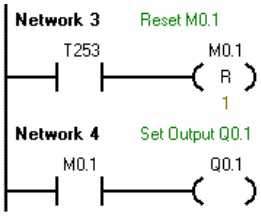
Chương trình:

LAD



STL

```
Network 1    Set Memory Bit cho Extended Pulse Timer
LD      I0.1        // Load I0.1.
EU                  // Khi co canh len o I0.1
S       M0.1, 1     // set M0.1.

Network 2    Khoi dong Timer T253
LD      M0.2        // Load M0.1.
                    // Neu M0.1 duoc set,
TON     T253, +50   // thi khoi dong timer T253
                    // voi PT=50 (5s)

Network 3    Reset M0.1
LD      T253        // Load timer T253.
                    // Khi Preset Time = Current Time,
R       M0.1, 1     // reset M0.1.

Network 4    Set Output Q0.1
LD      M0.1        // Load M0.1.
                    // Neu M0.1 duoc set,
=       Q0.1        // set output Q0.1.
```



9.5.3 Đảo chiều quay động cơ có khống chế thời gian

Mô tả hoạt động

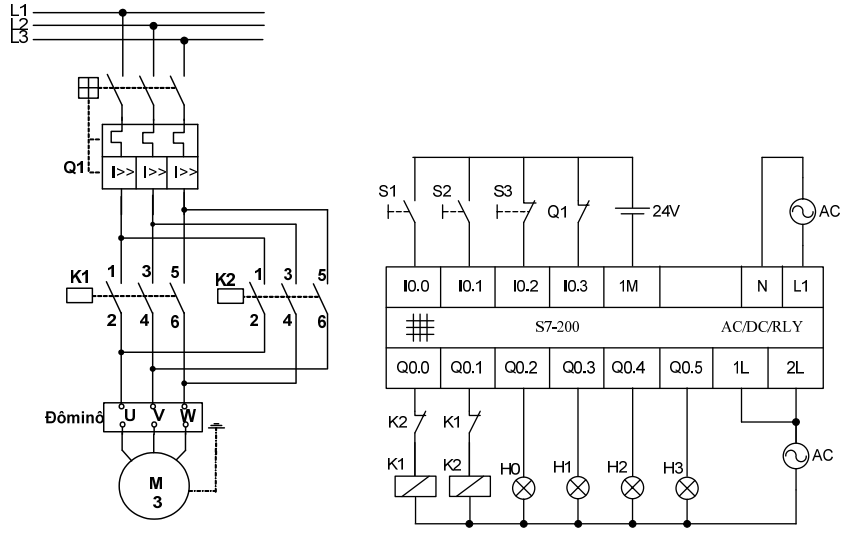
Một động cơ điện 3 pha có thể đảo chiều quay. Khi ấn nút nhấn quay phải "S1" (NO) thì động cơ quay phải, đèn "H1" sáng báo động cơ quay phải. Khi ấn nút nhấn quay trái "S2" (NO) thì động cơ quay trái, đèn "H2" sáng báo động cơ quay trái. Động cơ có thể dừng bất cứ lúc nào nếu ấn nút nhấn dừng "S3" (NC) hoặc xảy ra sự cố quá dòng làm cho tiếp điểm (NC) của thiết bị bảo vệ "Q1" (motor CB) tác động. Khi dừng thì đèn báo "H0" sáng.

Việc đảo chiều quay không thể thực hiện được sau khi nút dừng "S3" được ấn và chưa hết 5s chờ cho động cơ dừng hẳn. Đèn báo chờ đợi "H3" sẽ chớp tắt với tần số 1Hz trong thời gian chờ động cơ dừng hẳn.

Sơ đồ mạch động lực và nối dây với PLC:

Ở chương 7, ta đã sử dụng PLC S7-200 loại DC/DC/DC. Ở chương này để giúp bạn đọc làm quen với nhiều loại ngõ ra, S7-200 được sử dụng là loại AC/DC/RLY (Xem thêm chương 5).

Do ngõ ra của PLC là loại relay nên ta có thể nối trực tiếp ngõ ra với cuộn dây của contactor điều khiển động cơ, tuy nhiên cần chú ý đến mạch an toàn cho các ngõ ra.



Hình 9.1 Mạch động lực và nối dây vào/ra PLC AC/DC/Relay với ngoại vi

Bảng xác định vào/ra *(Bảng ký hiệu)*

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Nút nhấn quay phải, NO
S2	I0.1	Nút nhấn quay trái, NO
S3	I0.2	Nút nhấn dừng, NC
Q1	I0.3	Tiếp điểm motor CB bảo vệ quá tải, NC
K1	Q0.0	Contactor điều khiển quay phải
K2	Q0.1	Contactor điều khiển quay trái
H0	Q0.2	Đèn báo động cơ dừng
H1	Q0.3	Đèn báo động cơ quay phải
H2	Q0.4	Đèn báo động cơ quay trái
H3	Q0.5	Đèn báo chờ để đảo chiều

Phân tích:

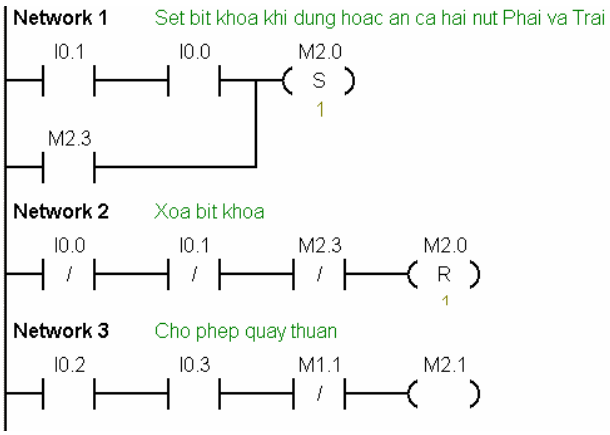
- Trong các bài toán điều khiển động cơ, ta cần phải chú ý xem, nếu có sự cố xảy ra với các nút nhấn có làm cho động cơ hoạt động không theo mong muốn hay không. Để đề phòng trường hợp này xảy ra, người lập trình phải tạo ra một khóa.

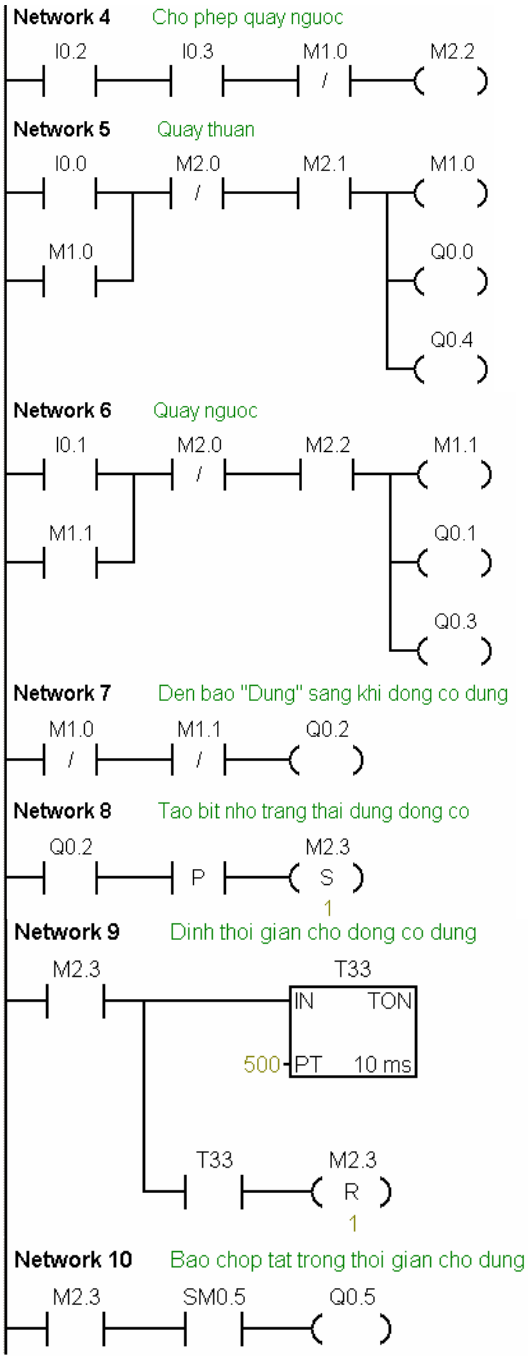
  Đối với mạch đảo chiều quay, có khống chế thời gian dừng (ở đây là 5s) thì khóa sẽ khống chế không cho động cơ khởi động không theo mong muốn cũng như sai chiều quay. Nếu khóa chưa được xóa về 0, thì không thể khởi động hay đảo chiều động cơ được. Trong bài toán này, khóa xóa về 0 khi cả 2 nút nhấn "S1" và "S2" không được tác động (ở trạng thái bình thường), hoặc thời gian chờ dừng đã hết. Khóa được chọn là M2.0

- Khi nút nhấn dừng "S3" được ấn, động cơ dừng và phải đợi trong thời gian 5s mới dừng hẳn, nên ta cần nhớ lại trạng thái này trong thời gian 5s để làm điều kiện SET cho khóa M2.0. Chọn memory bit M2.3.

- Để định thời 5s, sử dụng Timer TON. Chọn timer T33

Chương trình ở LAD:




Network 4
Cho phep quay nguoc
I0.2
I0.3
M1.0
/
M2.2
Network 5
Quay thuan
I0.0
M2.0
/
M2.1
M1.0
M1.0
Q0.0
Q0.4
Network 6
Quay nguoc
I0.1
M2.0
/
M2.2
M1.1
M1.1
Q0.1
Q0.3
Network 7
Den bao "Dung" sang khi dong co dung
M1.0
/
M1.1
/
Q0.2
Network 8
Tao bit nho trang thai dung dong co
Q0.2
P
M2.3
S
1
Network 9
Dinh thoi gian cho dong co dung
M2.3
T33
IN
TON
500
PT
10 ms
T33
M2.3
R
1
Network 10
Bao chop tat trong thoi gian cho dung
M2.3
SM0.5
Q0.5

9.5.4 Chiếu sáng Garage

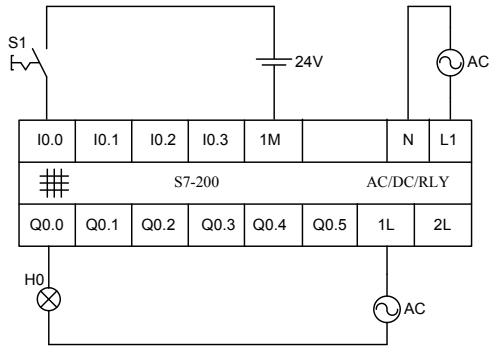
Mô tả hoạt động

Đèn trước cửa Garage không được tắt ngay lập tức khi ấn công tắc, mà nó vẫn còn sáng thêm một khoảng thời gian nữa (khoảng 1 phút) để cho người đi.

Bảng xác định vào/ra

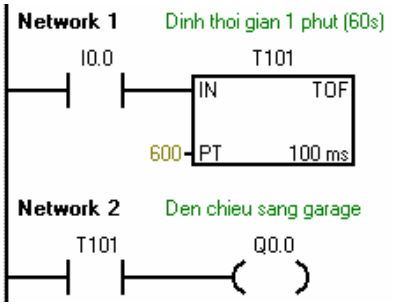
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Công tắc
H1	Q0.0	Đèn chiếu sáng Garage

Nối dây PLC:

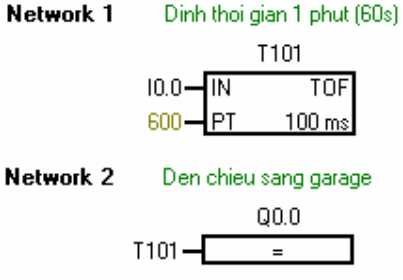


Chương trình

LAD



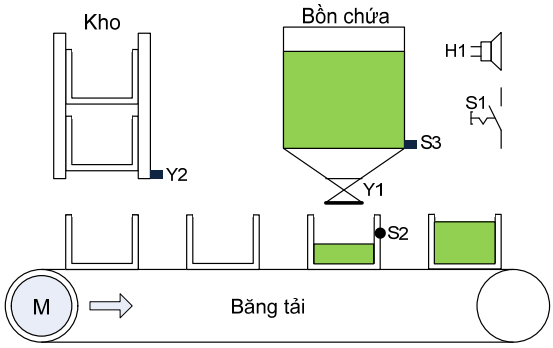
FBD



STL

9.5.5 Thiết bị rót chất lỏng vào thùng chứa

Sơ đồ công nghệ



Hình 9.2: Sơ đồ công nghệ thiết bị rót.

Mô tả hoạt động

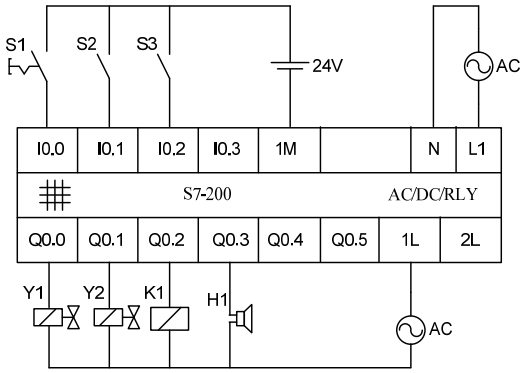
Khi bật công tắc "S1" thì thùng từ kho chứa thùng rỗng sẽ được đưa vào băng tải, và băng tải vận chuyển thùng hoạt động. Khi một thùng rỗng đến dưới bồn chứa (được nhận biết bởi cảm biến "S2") thì băng tải dừng. Van "Y1" mở rót chất lỏng trong bồn vào thùng. Sau thời gian 5s thì thùng chứa đầy. Van "Y1" đóng lại, một thùng rỗng sẽ được đưa vào băng tải và băng tải tiếp tục di chuyển cho đến khi nào thùng đến dưới bồn chứa thì dừng lại. Quá trình cứ lặp lại. Nếu chất lỏng trong bồn chứa hết thì còi "H1" sẽ báo với tần số 1Hz. Nếu thùng chứa trong kho hết thì băng tải cũng tự động dừng sau thời gian 15s kể từ thùng cuối cùng được rót đầy.

Chú ý: "Y2" là một solenoid được sử dụng để chặn thùng trong kho. Để thùng rớt vào băng tải chỉ cần solenoid có điện trong thời gian 100ms.

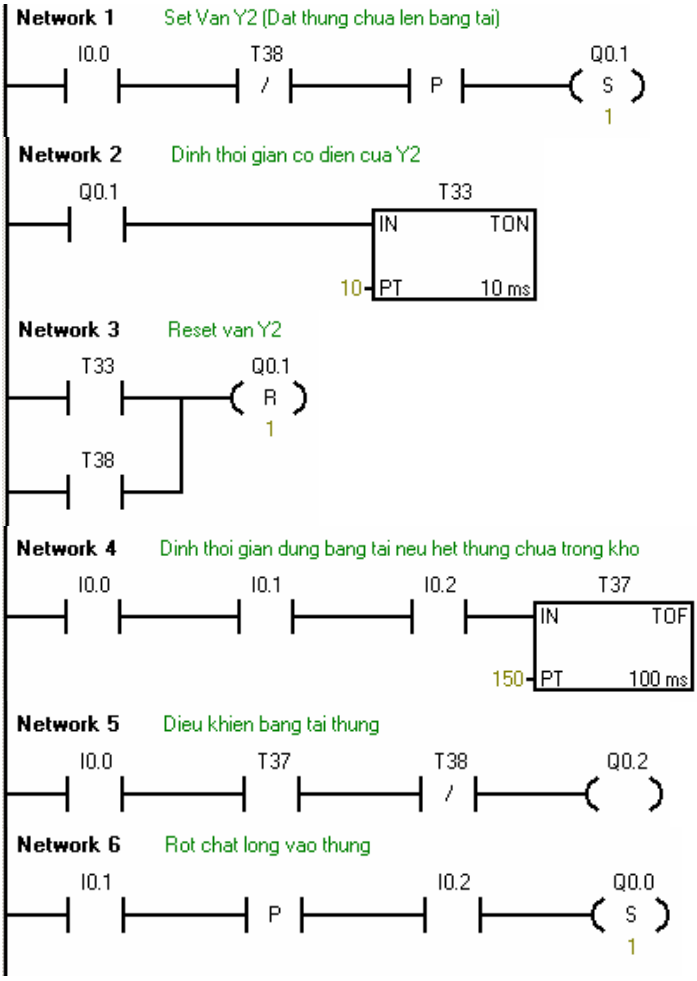
Bảng xác định vào/ra *(Bảng ký hiệu)*

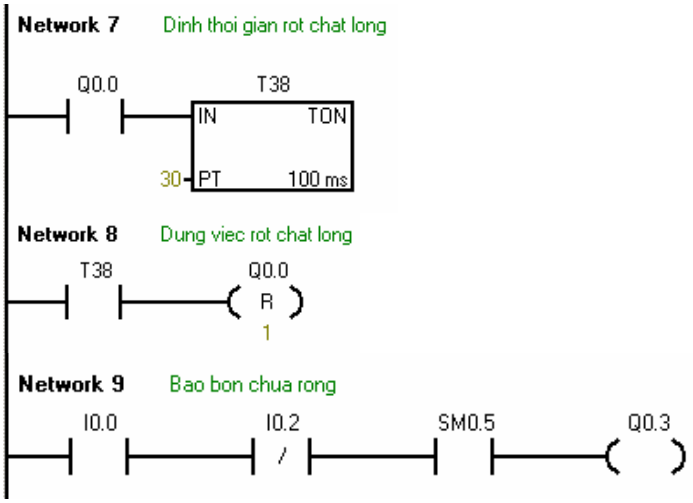
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Công tắc ON/OFF thiết bị rót
S2	I0.1	Cảm biến báo thùng đúng vị trí, (NO)
S3	I0.2	Cảm biến báo bồn rỗng, bồn rỗng ="0"
Y1	Q0.0	Van xả chất lỏng vào thùng chứa
Y2	Q0.1	Đặt thùng chứa lên băng tải
K1	Q0.2	Contactor điều khiển động cơ M kéo băng tải
H1	Q0.3	Còi báo bồn chứa rỗng

Sơ đồ nối dây với PLC



Chương trình ở LAD





Chương trình ở STL

Network 1 Set Van Y2 (Dat thung chua len bang tai)

```
LD      I0.0          // load I0.0.
AN      T38           // Qua trinh rot ket thuc
EU                    // Lay canh len T38
S       Q0.1, 1       // Mo van Y2
```

Network 2 Dinh thoi gian co dien cua Y2

```
LD      Q0.1          // Load Van Y2
                      // Neu Van Y2 duoc set,
TON     T33, 10       // khoi dong timer T33
                      // Thoi gian mo van Y2 la 100ms
```

Network 3 Reset van Y2

```
LD      T33           // Load timer T33.
                      // Neu thoi gian mo van da het
O       T38           // hoac bat dau qua trinh rot chat long,
R       Q0.1, 1       // Dong van Y2
```

Network 4 Dinh thoi gian dung bang tai neu het thung chua trong kho

```
LD      I0.0          // load I0.0
A       I0.1          // Neu thung khong co tai vi tri
A       I0.2          // hoac bon chua rong
TOF     T37, 150      // Dinh thoi gian dung bang tai
```

Network 5 Dieu khien bang tai thung

```
LD      I0.0          // Load I0.0
A       T37           // Neu Thoi gian dung bang tai chua het
AN      T38           // va qua trinh rot ket thuc
=       Q0.2          // khoi dong bang tai thung
```

Network 6 Rot chat long vao thung

```
LD      I0.1          // Neu thung dung vi tri
EU                    // Lay canh len
A       I0.2          // va bon chua con chat long
S       Q0.0, 1       // mo van Y1
```

```
Network 7    Dinh thoi gian rot chat long

LD      Q0.0          // Khi van Y1 mo
TON     T38, 30       // Dinh thoi gian rot chat long vao thung

Network 8    Dung viec rot chat long

LD      T38           // Khi du thoi gian rot
R       Q0.0, 1       // dong van Y1

Network 9    Bao bon chua rong

LD      I0.0          // load I0.0
AN      I0.2          // Bon chua rong
A       SM0.5         // lay xung 1Hz
=       Q0.3          // Coi bao dong voi tan so 1Hz
```

9.6 Câu hỏi và bài tập

BT9.1 Đèn hành lang hoặc đèn cầu thang có định thời.

Trên tường của các hành lang chung cư, trước mỗi cửa căn hộ có gắn một nút nhấn (giả sử hành lang có 6 căn hộ tương ứng 6 nút ấn từ S1 đến S6) . Khi tác động nút nhấn thì đèn chiếu sáng hành lang (gồm có 6 đèn H1 đến H6) sẽ sáng trong thời gian 1 phút rồi sau đó tự động tắt. Nếu trong thời gian 1 phút mà có một nút nhấn nào đó được ấn tiếp tục thì đèn sẽ sáng thêm 1 phút nữa kể từ lúc ấn sau cùng. Yêu cầu:

1. Lập bảng xác định vào/ra
2. Vẽ sơ đồ nối dây vào/ra và nguồn cấp cho PLC S7-200 AC/DC/RLY.
3. Viết chương trình và sau đó nạp vào PLC để kiểm tra.

BT9.2 Tạo OFF-delay Timer

Từ một ON-delay timer, hãy viết chương trình tạo OFF-delay timer theo sơ đồ ở mục 9.4.

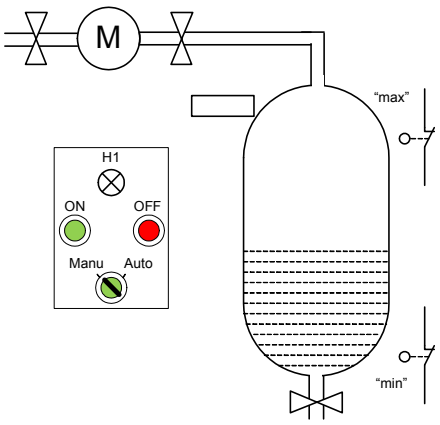
BT9.3 Điều khiển Đèn và Quạt hút

Trong một phòng vệ sinh có trang bị một đèn chiếu sáng và một quạt hút khí. Khi vào phòng, bật công tắc lên vị trí “ON” thì đèn sáng. Nếu ở trong phòng lâu hơn thời gian 3 phút thì quạt hút tự động hoạt động. Khi ra khỏi phòng bật công tắc về vị trí “OFF” thì đèn tắt. Nếu quạt hút đã hoạt động thì sau khi đèn tắt khoảng 5 phút nó mới tự động dừng. Yêu cầu:

1. Lập bảng xác định vào/ra
2. Vẽ sơ đồ nối dây PLC với ngoại vi

2. Viết chương trình điều khiển và nạp vào PLC để kiểm tra

BT9.4 Điều khiển bơm nước



Một bồn chứa nước được làm đầy bởi một bơm M. Bơm này có hai chế độ hoạt động:

** Chế độ tay:*

Đặt công tắc chọn chế độ "S1" ở vị trí "Manu". Đèn "H1" sáng báo chế độ "tay". Ở chế độ "tay", bơm chỉ có thể hoạt động nếu ấn nút nhấn S1 "ON" (NO). Bơm sẽ tự động tắt nếu ấn nút nhấn S2 "OFF" (NC) hoặc nước trong bồn đạt đến giá trị "max" (được phát hiện bởi cảm biến "S5").

Hình 9.3 Sơ đồ công nghệ điều khiển bơm

** Chế độ tự động:*

Khi đặt công tắc "S1" về vị trí "Auto", thì bơm nước hoạt động tự động. Nếu nước xuống dưới mức "min" (phát hiện bởi cảm biến "S4") thì bơm sẽ được đóng điện bởi contactor K1. Khi nước trong bồn lên đến vị trí "max" thì contactor mất điện và động cơ bơm nước dừng. Ở chế độ tự động thì đèn H1 tắt.

Nhằm loại trừ sự sóng sánh của mặt nước khi bơm làm cho cảm biến báo mực nước ở vị trí "max" không chính xác, thì động cơ bơm nước cần phải kéo dài thời gian hoạt động thêm 1s nữa rồi mới dừng hẳn cho cả hai trường hợp "Manual" và "Auto".

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Công tắc chọn chế độ, 0: Auto; 1: Manual
S2	I0.1	Nút nhấn mở máy bơm nước ở chế độ Manual, NO
S3	I0.2	Nút nhấn dừng bơm nước ở chế độ tay, NC
S4	I0.3	Cảm biến báo bồn nước ở min, NC
S5	I0.4	Cảm biến báo bồn nước ở max, NC
K1	Q0.0	Contactor điều khiển động cơ bơm nước
H1	Q0.1	Đèn báo chế độ Manual.

Yêu cầu:

1. Vẽ sơ đồ mạch động lực nối contactor với động cơ bơm nước 3pha
2. Lập bảng xác định vào/ra

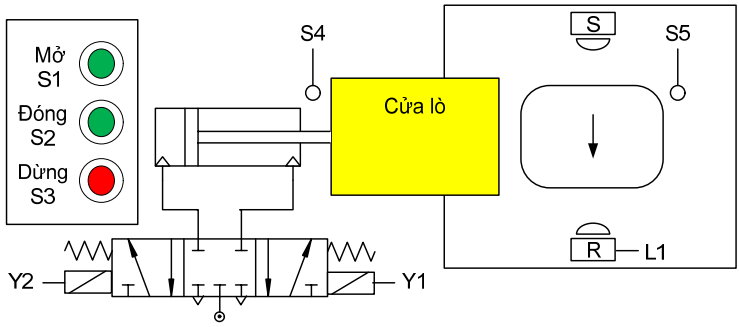
3. Vẽ sơ đồ nối dây PLC
4. Viết chương trình điều khiển và nạp vào PLC để kiểm tra.

BT9.5 Điều khiển cửa lò

Một cửa lò có chức năng "mở, đóng và ở vị trí bất kỳ" được điều khiển bởi một cylinder. Ở vị trí bình thường thì cửa lò được đóng.

- Khi tác động nút nhấn "S1" (NO) thì cửa lò mở ra và khi đến công tắc hành trình giới hạn mở cửa "S4" (NC) thì dừng lại.
- Nếu cửa đã mở ra ở vị trí giới hạn mở cửa "S4" thì sẽ tự động đóng lại sau thời gian 6s hoặc nút nhấn đóng cửa "S2" (NO) được ấn.
- Khi đến giới hạn cửa đóng "S5" (NC) thì việc đóng cửa kết thúc.
- Quá trình đóng cửa dừng ngay lập tức nếu cảm biến L1 (NO) bị tác động. Nhưng nếu cảm biến quang không bị tác động thì quá trình đóng cửa vẫn tiếp tục.
- Khi cửa lò đang dịch chuyển có thể dừng bằng cách ấn nút dừng "S3" (NC).

Sơ đồ công nghệ



Hình 9.4 Điều khiển cửa lò bằng khí nén với van 5/3.

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Nút nhấn mở cửa lò
S2	I0.1	Nút nhấn đóng cửa lò
S3	I0.2	Nút nhấn dừng, NC
S4	I0.3	Công tắc hành trình giới hạn mở cửa, NC
S5	I0.4	Công tắc hành trình giới hạn đóng cửa, NC
L1	I0.5	Cảm biến quang, NO
Y1	Q0.0	Van điều khiển cylnder đóng cửa
Y2	Q0.1	Van điều khiển cylnder mở cửa

Yêu cầu:

1. Vẽ sơ đồ nối dây với PLC
2. Viết chương trình và nạp vào PLC để kiểm tra.

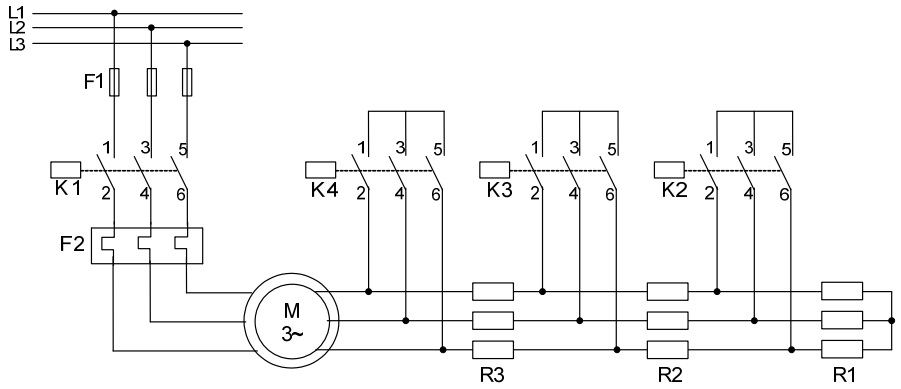
BT9.6 Điều khiển quá trình khởi động động cơ rotor dây quấn

Nhằm tránh dòng điện khởi động cao trong các động cơ rotor dây quấn có gắn thêm các điện trở phụ.

Khi tác động nút nhấn mở máy "S1" (NO), thì contactor K1 có điện. Các contactor K2, K3 và K4 bắt đầu đóng lần lượt cách nhau một khoảng thời gian là 5s. Khi contactor cuối cùng là K4 được đóng thì rotor được ngắn mạch và động cơ hoạt động ở chế độ định mức.

Khi tác động nút nhấn "S0" (NC) thì động cơ dừng.

Sơ đồ công nghệ



Hình 9.5: Điều khiển khởi động động cơ rotor dây quấn

Yêu cầu:

1. Lập bảng xác định vào/ra
2. Vẽ sơ đồ nối dây với PLC loại DC/DC/DC
3. Viết chương trình và nạp vào PLC để kiểm tra.

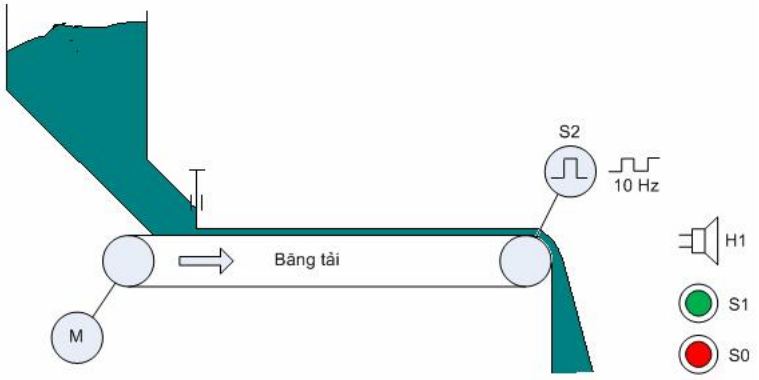
BT9.7 Giám sát hoạt động băng tải bằng cảm biến phát xung

Một băng tải được truyền động thông qua một động cơ. Khi băng tải hoạt động thì cảm biến giám sát băng tải "S2" phát xung có điện áp 24V với tần số 10Hz. Khi băng tải đứng yên thì "S2" phát ra tín hiệu "0".

Khi có lỗi xảy ra, ví dụ băng tải bị kẹt, tín hiệu giám sát không phát ra, ta cũng không biết là động cơ có tắt hay không. Trong trường hợp này, động cơ kéo băng tải phải dừng ngay lập tức và chuông báo băng tải bị lỗi "H1" vang với tần số 2Hz.

- Băng tải khởi động bằng nút nhấn "S1" (NO).
- Băng tải dừng bằng nút nhấn "S0" (NC).

Sơ đồ công nghệ



Hình 9.6: Giám sát hoạt động băng tải bằng cảm biến phát xung.

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S0	I0.0	Nút nhấn dừng, NC
S1	I0.1	Nút nhấn mở máy, NO
S2	I0.2	Cảm biến giám sát băng tải, xung
K1	Q0.0	Contactor điều khiển động cơ băng tải
H1	Q0.1	Đèn báo

Yêu cầu:

1. Vẽ sơ đồ nối dây với PLC loại DC/DC/DC
2. Viết chương trình và nạp vào PLC để kiểm tra.

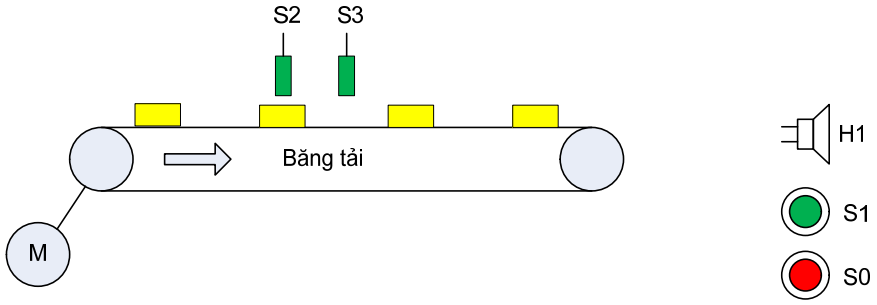
BT9.8 Giám sát hoạt động băng tải bằng thời gian

Một băng tải vận chuyển sản phẩm được truyền động thông qua một động cơ. Sản phẩm trên băng tải được nhận biết bởi hai cảm biến "S2" và "S3".

Thời gian tối đa để sản phẩm di chuyển từ "S2" đến "S3" là 3s. Nếu vượt quá thời gian này thì băng tải xem như bị lỗi. Khi bị lỗi thì động cơ kéo băng tải dừng ngay lập tức và một chuông báo phát ra với tần số 3Hz.

- Băng tải khởi động bằng nút nhấn "S1" (NO).
- Băng tải dừng bằng nút nhấn "S0" (NC).

Sơ đồ công nghệ



Hình 9.7: Giám sát hoạt động băng tải bằng thời gian.

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S0	I0.0	Nút nhấn dừng, NC
S1	I0.1	Nút nhấn mở máy, NO
S2	I0.2	Cảm biến giám sát sản phẩm 1, NO
S3	I0.3	Cảm biến giám sát sản phẩm 2, NO
K1	Q0.0	Contactor điều khiển động cơ băng tải
H1	Q0.1	Chuông báo

Yêu cầu:

1. Vẽ sơ đồ nối dây với PLC loại DC/DC/DC
2. Viết chương trình và nạp vào PLC để kiểm tra.

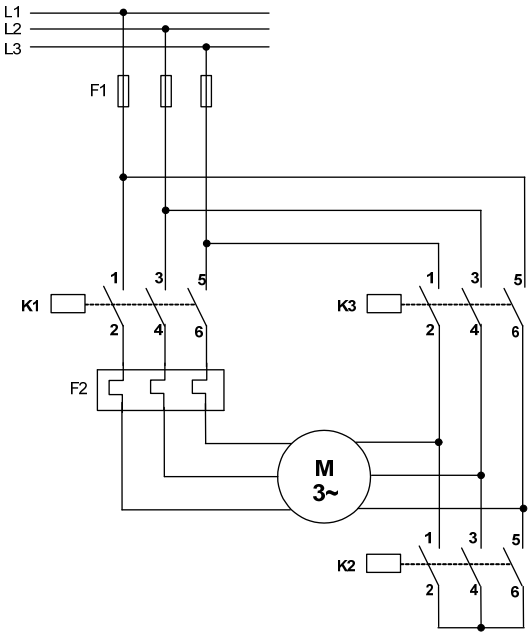
BT9.9 Khởi động Sao-tam giác

Thực hiện trình tự khởi động tự động sao-tam giác của một động cơ điện không đồng bộ 3 pha rotor lồng sóc với PLC theo sơ đồ hình 9.8.

Khi ấn nút nhấn “S1” (NO), thì động cơ hoạt động ở chế độ sao (K1 và K2 đóng). Và sau một thời gian đặt trước (giả sử 10s), thì tự động chuyển sang chế độ tam giác (K2 mất điện, K3 có điện).

Khi ấn nút “S0” (NC) thì động cơ dừng ngay lập tức. Trong trường hợp quá tải (được báo bởi tiếp điểm nhiệt F2) thì động cơ cũng dừng.

Sơ đồ mạch động lực



Hình 9.8: Mạch động lực khởi động sao-tam giác.

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S0	I0.0	Nút nhấn dừng, NC
S1	I0.1	Nút nhấn mở máy, NO
F2	I0.2	Báo quá dòng, NC
K1	Q0.0	Contactor nguồn
K2	Q0.1	Contactor chạy sao
K3	Q0.2	Contactor chạy tam giác

Yêu cầu:

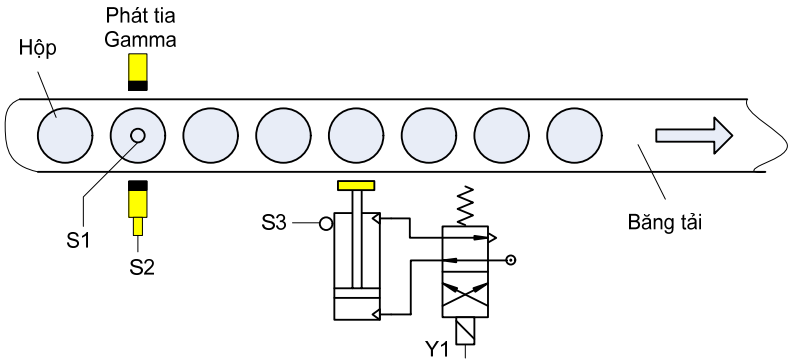
1. Vẽ sơ đồ nối dây với PLC loại AC/DC/RLY
2. Viết chương trình và nạp vào PLC để kiểm tra.

BT9.10 Kiểm tra chất lượng sản phẩm

Đồ hộp được vận chuyển trên một băng tải. Các hộp cách nhau một khoảng nhỏ. Các hộp đã được đóng nắp cần được kiểm tra tình trạng đổ đầy.

Việc kiểm tra chất lượng được thực hiện với một nguồn phát tia Gamma, đầu thu sẽ phát tín hiệu “1” nếu hộp không được đổ đầy. Việc đo được thực hiện xong nếu công tắc hành trình S1 bị tác động (phát ra tín hiệu “1”). Trường hợp hộp không được đổ đầy thì sau thời gian đo 2s, van Y1 điều khiển Cylinder đẩy hộp kém chất lượng ra ngoài.

Sơ đồ công nghệ



Hình 9.9: Kiểm tra chất lượng sản phẩm

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Công tắc hành trình, NO (tác động S1=1)
S2	I0.1	Nguồn tia Gama, không đầy S2=1
S3	I0.2	Cảm biến báo Cylinder đã đến cuối hành trình, NO
Y1	Q0.0	Van điều khiển Cylinder

Yêu cầu:

1. Vẽ sơ đồ nối dây với PLC loại AC/DC/RLY.
2. Viết chương trình và nạp vào PLC để kiểm tra.

BT9.11 Điều khiển đèn giao thông

Một giao lộ có lối đi dành cho người đi bộ và ô tô hoạt động ở hai chế độ ngày và đêm.

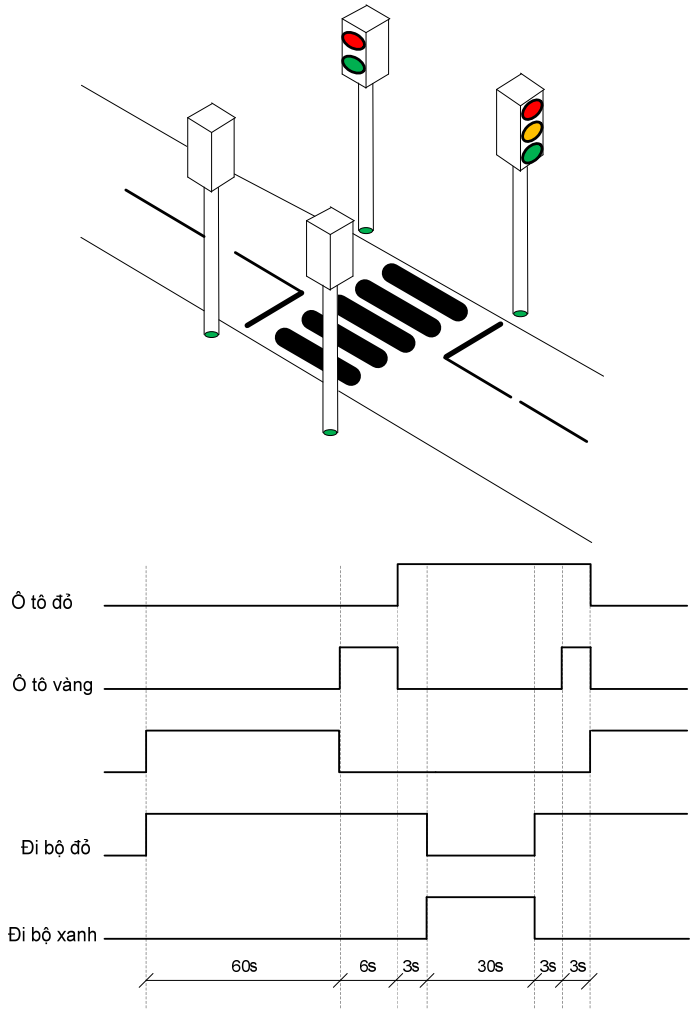
** Chế độ ngày*

Đèn hoạt động hoàn toàn tự động theo giản đồ thời gian hình 9.10. Chế độ ngày được chọn khi công tắc S1 ở logic "1".

** Chế độ đêm*

Khi đặt công tắc S1 ở logic "0" thì bộ điều khiển chuyển sang hoạt động ở chế độ đêm. Khi chuyển sang chế độ đêm thì chế độ ngày bị cắt ngay lập tức. Tất cả các đèn đều tắt, chỉ có đèn vàng ở đường dành cho ô tô chớp tắt với tần số 1Hz.

Sơ đồ công nghệ và giản đồ thời gian



Hình 9.10: Sơ đồ công nghệ đèn giao thông và giản đồ thời gian

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Công tắc chọn chế độ, 1: ngày; 0: đêm
H1	Q0.0	Ô tô đỏ
H2	Q0.1	Ô tô vàng
H3	Q0.2	Ô tô xanh
H4	Q0.3	Đi bộ đỏ
H5	Q0.4	Đi bộ xanh

10 Bộ đếm (Counter)

10.1 Giới thiệu

Trong nhiều trường hợp, việc kiểm tra một số lượng xác định phải thông qua tổng các xung. Có thể thực hiện đếm các xung này bằng các bộ đếm. Sử dụng bộ đếm có thể giải quyết được một số vấn đề sau:

- Đếm số lượng
- So sánh với một giá trị đặt trước ở các trường hợp bằng nhau, nhỏ hơn, lớn hơn.
- Kiểm tra sự khác biệt về số lượng.

Trong điều khiển vị trí thì việc sử dụng bộ đếm tốc độ cao là không thể thiếu. Phần điều khiển vị trí và bộ đếm tốc độ cao sẽ được trình bày chi tiết trong tập 2 của bộ sách này. Ở chương này chỉ đề cập đến các bộ đếm thông thường.

Bộ đếm cũng có thể sử dụng để thực hiện các nhiệm vụ như: Cộng các xung của bộ phát xung nhịp và dựa vào đó để gọi các giai đoạn điều khiển liên tiếp nhau. Hoặc các yêu cầu điều khiển theo chu kỳ lặp như điều khiển đèn giao thông.

Các PLC thường có 3 loại bộ đếm: bộ đếm lên, bộ đếm xuống, bộ đếm lên-xuống.

Có 256 bộ đếm ở S7-200 có địa chỉ từ C0 đến C255. Chúng cũng có 3 loại bộ đếm là:

+ Bộ đếm lên CTU (Up Counter).

+ Bộ đếm xuống CTD (Down Counter).

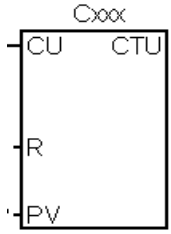
+ Bộ đếm lên-xuống (Up/Down Counter).

Khi sử dụng một counter chúng ta cần phải xác định các thông số sau:

- Loại counter (CTU, CTD hay CTUD)
- Số của counter sẽ sử dụng, không được gán cùng một số counter cho nhiều counter.
- Khai báo giá trị cần đếm cho counter.
- Tín hiệu xung cung cấp cho bộ đếm.
- Tín hiệu xóa bộ đếm.

10.2 Bộ đếm lên CTU (Count Up)

Bộ đếm CTU được biểu diễn ở LAD như sau:



Với:

Cxxx: Ký hiệu và số thứ tự của counter, ví dụ: C10.

CTU: Ký hiệu nhận biết bộ đếm lên

CU: Đếm lên. Ngõ vào bit,

R: Xóa bộ đếm về 0. Ngõ vào bit,

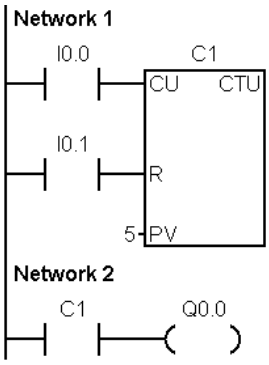
PV: Giá trị đặt trước cho bộ đếm. Biểu diễn ở số Integer.

Mỗi khi tín hiệu tại CU từ mức "0" lên "1" thì bộ đếm sẽ tăng giá trị hiện hành của nó lên 1 đơn vị. Khi giá trị hiện hành của bộ đếm (Cxxx) lớn hơn hoặc bằng giá trị đặt trước tại ngõ vào PV (Preset Value) thì ngõ ra bit của counter (counter bit) sẽ lên mức "1". Giá trị đếm lên tối đa là 32.767. Phạm vi của bộ đếm là C0 đến C255.

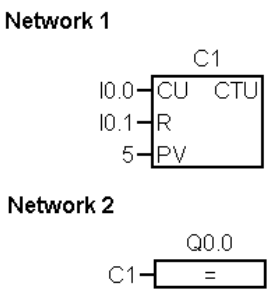
Bộ đếm sẽ bị xóa về 0 khi ngõ vào Reset (R) lên mức "1", hoặc khi sử dụng lệnh Reset để xóa bộ đếm.

Ví dụ: Cứ mỗi xung từ "0" chuyển lên "1" tại ngõ vào I0.0, bộ đếm sẽ tăng 1 đơn vị. Từ xung thứ 5 trở đi ngõ ra Q0.0 sẽ lên "1". Nếu có xung vào tại ngõ I0.1 thì ngõ ra Q0.0 xuống "0".

LAD



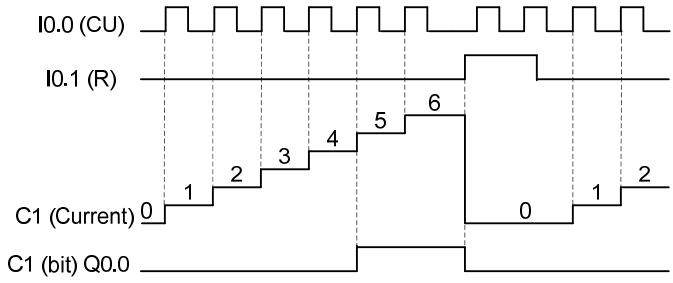
FBD



STL

```
Network 1
LD      I0.0
LD      I0.1
CTU     C1, 5
Network 2
LD      C1
=       Q0.0
```

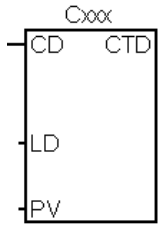
Giản đồ xung:



Để lấy counter CTU, trong cây lệnh bấm vào dấu (+) của biểu tượng Counters, sau đó chọn CTU, bấm và giữ chuột trái kéo thả vào vị trí mong muốn trong chương trình. Nhập các thông tin ở Cxxx, CU, R và PV.

10.3 Bộ đếm xuống CTD (Count Down)

Bộ đếm xuống CTD được biểu diễn ở LAD như sau:



Với:

Cxxx: Ký hiệu và số thứ tự của counter, ví dụ: C20.

CTD: Ký hiệu nhận biết bộ đếm xuống

CD: Ngõ vào đếm xuống. Ngõ vào bit,

LD: Nạp giá trị đặt trước cho bộ đếm xuống.Ngõ vào bit,

PV: Giá trị đặt trước cho bộ đếm. Biểu diễn ở số Integer.

Mỗi khi tín hiệu tại CD từ mức "0" lên "1" thì bộ đếm sẽ giảm giá trị hiện hành của nó xuống 1 đơn vị. Khi giá trị hiện hành của bộ đếm (Cxxx) bằng 0, thì Counter Bit Cxxx lên "1". Bộ đếm xóa Counter Bit Cxxx và nạp giá trị đặt trước ở PV khi ngõ vào LD (load) lên mức "1".

Bộ đếm sẽ dừng đếm khi giá trị hiện hành bằng 0 và counter bit Cxxx lên "1".

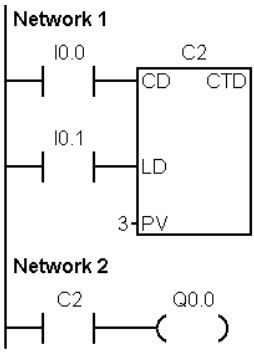
Phạm vi của bộ đếm là C0 đến C255.

Khi xóa bộ đếm bằng lệnh Reset, counter bit bị xóa và giá trị hiện hành được đặt về 0.

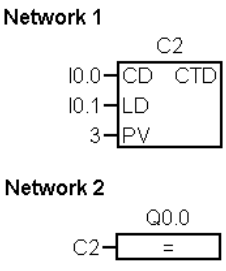
Để lấy counter CTD, trong cây lệnh bấm vào dấu (+) của biểu tượng Counters, sau đó chọn CTD, bấm và giữ chuột trái kéo thả vào vị trí mong muốn trong chương trình. Nhập các thông tin ở Cxxx, CD, LD và PV.

Ví dụ: Sử dụng bộ đếm xuống C2, giá trị hiện hành giảm từ 3 trở về 0. Với I0.1 ở logic "0" và mỗi lần I0.0 chuyển từ "0" lên "1" thì bộ đếm C2 giảm đi một đơn vị. Khi giá trị hiện hành trong bộ đếm C2 bằng 0 thì ngõ ra Q0.0 lên "1". Khi I0.1 ở "1" thì bộ đếm được đặt trước giá trị đếm là 3.

LAD



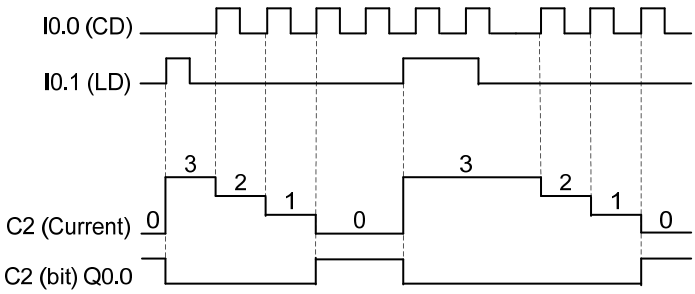
FBD



STL

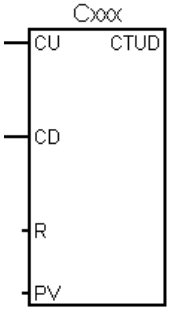
```
Network 1
LD      I0.0
LD      I0.1
CTD     C2, 3
Network 2
LD      C2
=       Q0.0
```

Giản đồ xung:



10.4 Bộ đếm lên-xuống CTUD (Count Up/Down)

Bộ đếm xuống CTUD được biểu diễn ở LAD như sau:



Với:

Cxxx: Ký hiệu và số thứ tự của counter, ví dụ: C0.

CTUD: Ký hiệu nhận biết bộ đếm lên-xuống

CU: Ngõ vào đếm lên. Ngõ vào bit

CD: Ngõ vào đếm xuống. Ngõ vào bit,

R: Xóa bộ đếm về 0.Ngõ vào bit,

PV: Giá trị đặt trước cho bộ đếm. Biểu diễn ở số Integer.

Lệnh đếm lên-xuống (CTUD) sẽ đếm lên mỗi khi ngõ vào đếm lên (CU) từ mức “0” lên “1”, và đếm xuống mỗi khi ngõ vào đếm xuống (CD) chuyển từ “0” lên “1”. Giá trị hiện hành Cxxx giữ giá trị hiện hành của bộ đếm. Giá trị đặt trước PV được so sánh với giá trị hiện hành mỗi khi thực hiện lệnh đếm.

Khi đạt đến giá trị max (32.767), thì ở cạnh lên kế tiếp tại ngõ vào đếm lên bộ đếm sẽ đặt về giá trị min (-32.768).

Khi đạt đến giá trị min (-32.768), thì ở cạnh lên kế tiếp tại ngõ vào đếm xuống bộ đếm sẽ đặt về giá trị max (32.767).

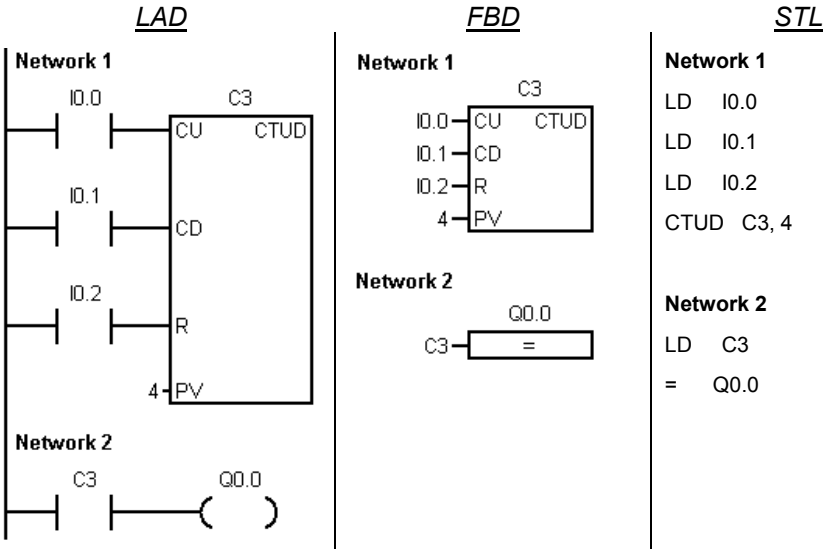
Khi giá trị hiện hành Cxxx lớn hơn hoặc bằng giá trị đặt trước PV, thì Counter Bit Cxxx lên “1”. Ngược lại Counter Bit Cxxx bằng “0”.

Phạm vi của bộ đếm là C0 đến C255.

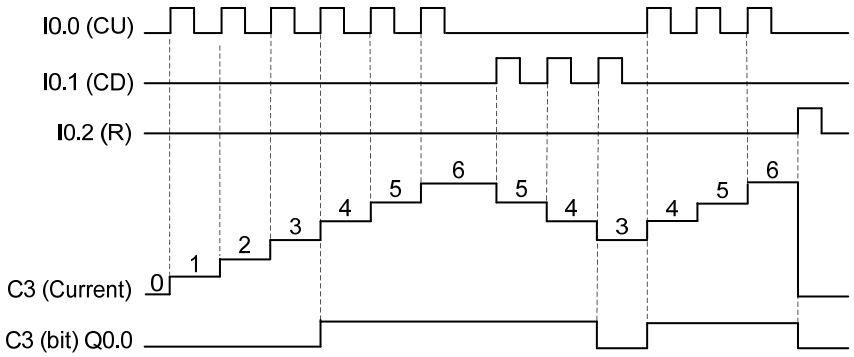
Bộ đếm sẽ bị xóa về 0 khi ngõ vào Reset (R) lên mức “1”, hoặc khi sử dụng lệnh Reset để xóa bộ đếm.

Để lấy counter CTUD, trong cây lệnh bấm vào dấu (+) của biểu tượng Counters, sau đó chọn CTUD, bấm và giữ chuột trái kéo thả vào vị trí mong muốn trong chương trình. Nhập các thông tin ở Cxxx, CU,CD, R và PV.

Ví dụ: Sử dụng bộ đếm xuống C3. Ngõ vào đếm lên nối với I0.0. Ngõ vào đếm xuống nối với I0.1. Xóa bộ đếm bằng I0.2. Khi bộ đếm có giá trị hiện hành >=4 thì ngõ ra Q0.0 lên "1".



Giản đồ xung:



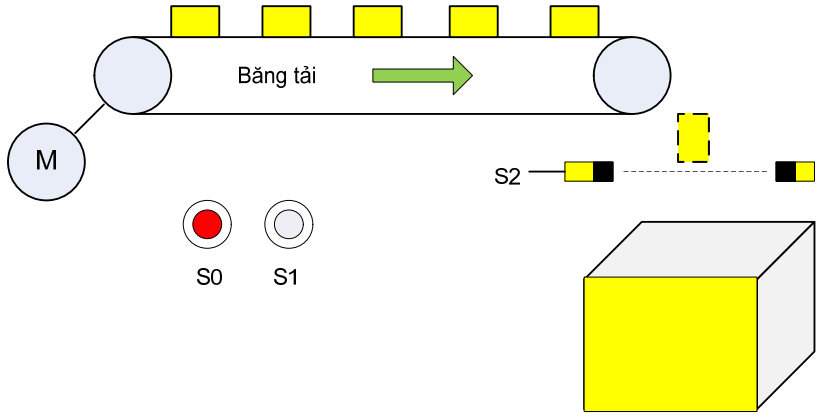
10.5 Ứng dụng bộ đếm

10.5.1 Đếm sản phẩm được đóng gói

Sản phẩm đã đóng gói được đưa vào một thùng chứa bằng một băng tải (kéo bởi động cơ M). Mỗi thùng chứa được 10 sản phẩm. Khi sản phẩm đã được đếm đủ thì băng tải dừng lại đề cho người vận hành đưa một thùng rỗng vào. Sau khi người vận hành ấn nút S1(NO) để tiếp tục thì băng tải hoạt động. Quá trình cứ lặp đi lặp lại cho đến khi nào ấn nút dừng S0 (NC).

Sản phẩm trước khi đưa vào thùng sẽ đi qua cảm biến quang S2 (NC).

Sơ đồ công nghệ:

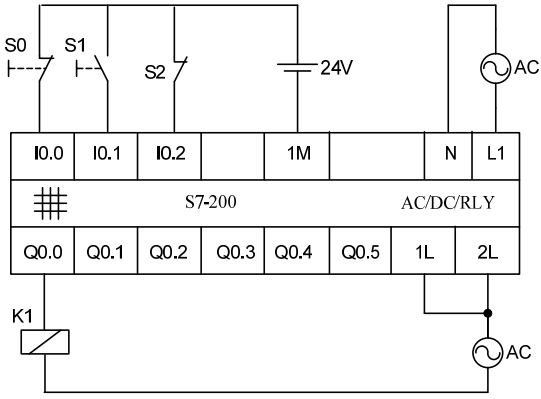


Hình 10.1: Đếm sản phẩm được đóng gói

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S0	I0.0	Nút nhấn dừng, NC
S1	I0.1	Nút nhấn khởi động băng tải, NO
S2	I0.2	Cảm biến nhận biết sản phẩm, NC
K1	Q0.0	Contactor điều khiển động cơ M

Nối dây với PLC



Phân tích

** Động cơ kéo băng tải:*

Điều kiện hoạt động: - Nút nhấn S1 (NO) được tác động

Điều kiện dừng: - Nút nhấn dừng S0 (NC) được tác động, hoặc

- Đếm đủ 10 sản phẩm (bộ đếm C1).

Nếu sử dụng Set, Reset:

Điều kiện Set động cơ M: K1= S1

Điều kện Reset động cơ M: K1= $\overline{S0} \vee C1$

Vì ưu tiên dừng máy nên sử dụng ưu tiên Reset. Ngoài ra khi đã đếm đủ 10 sản phẩm thì Counter Bit C1 luôn luôn =”1” nên ở ngõ R của khâu RS ta sử dụng cạnh lên đối với bit C1.

** Bộ đếm C1:*

Vì đếm đến 10 sản phẩm thì phát tín hiệu để động cơ dừng, nên ở đây sử dụng bộ đếm lên.

Điều kiện ngõ vào đếm lên CU: = $\overline{S2}$

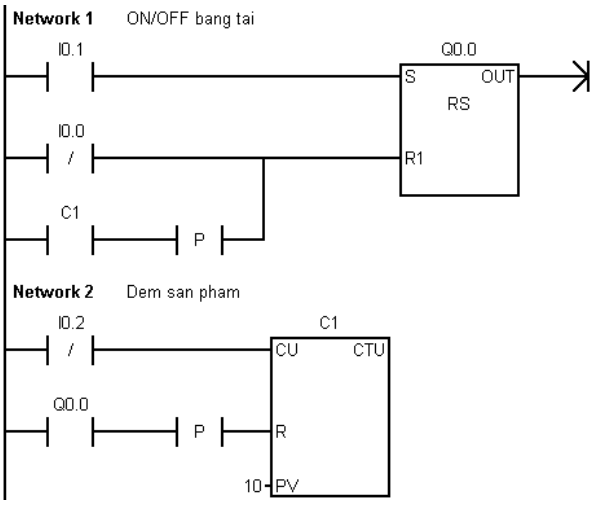
Giá trị đặt cho bộ đếm PV:= 10

Điều kiện xóa bộ đếm R:= cạnh lên K1

Chú ý: Vì chân Reset(R) của bộ đếm sẽ xóa bộ đếm về 0 theo mức logic nên ta phải sử dụng cạnh lên ở ngõ vào.

Chương trình

LAD



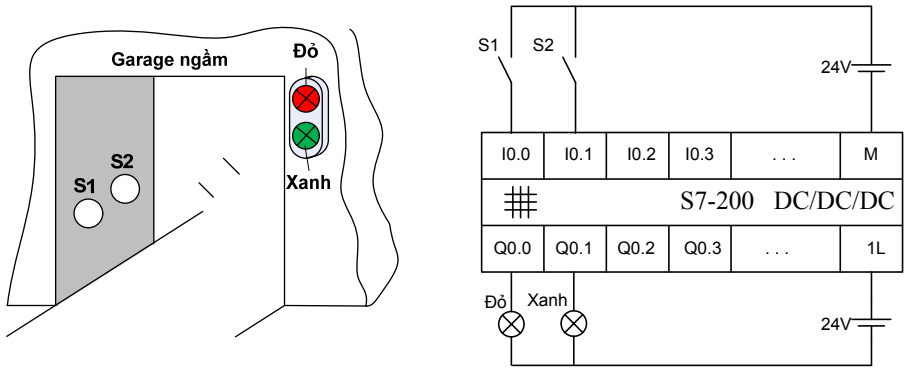
STL

```
Network 1  ON/OFF bang tai
LD    I0.1
LDN   I0.0
LD    C1
EU
OLD
NOT
LPS
A     Q0.0
=     Q0.0
LPP
ALD
O     Q0.0
=     Q0.0
Network 2  Dem san pham
LDN   I0.2
LD    Q0.0
EU
CTU   C1, 10
```

10.5.2 Kiểm soát chỗ cho Garage ngầm

Một Garage ngầm có 20 chỗ đậu xe. Ở ngõ vào có hai đèn báo: Đèn đỏ báo hiệu Garage đã hết chỗ, đèn xanh báo hiệu Garage còn chỗ trống. Đường vào và đường ra chỉ cho phép một xe chạy.

Sơ đồ công nghệ được cho ở hình 10.2. Hai cảm biến S1 và S2 được đặt gần nhau để nhận biết xe vào và ra.



Hình 10.2: Sơ đồ Ragare ngầm và sơ đồ nối dây PLC

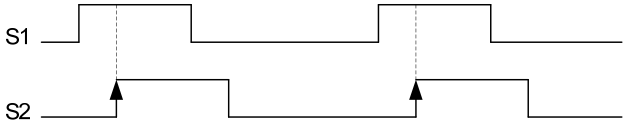
Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Cảm biến nhận biết xe vào/ra
S2	I0.1	Cảm biến nhận biết xe ra/vào
Đỏ	Q0.0	Đèn báo hết chỗ đậu xe
Xanh	Q0.1	Đèn báo còn chỗ đậu xe

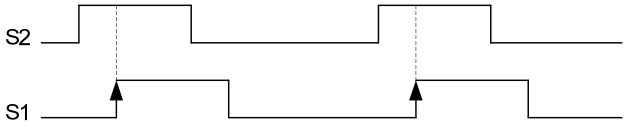
Phân tích

** Nhận biết xe vào/ra*

Vì Garage ngầm chỉ có một cửa ra vào cho một làn xe chạy, nên không thể lấy riêng lẻ một cảm biến để nhận biết xa vào và cảm biến còn lại để nhận biết xe ra vì sẽ có sự trùng lắp và không rõ ràng. Để giải quyết, kết hợp cả hai cảm biến này. Giản đồ xung cho xe vào rà ra Garage như sau:



Giản đồ thời gian xe vào



Giản đồ thời gian xe ra

Từ giản đồ thời gian ta nhận thấy:

Tín hiệu xe vào:= cạnh lên S2 AND mức logic “1” của S1

Tín hiệu xe ra:= cạnh lên S1 AND mức logic “1” của S2

** Bộ đếm*

Vì số lượng xe trong Ragage thay đổi khi có xe vào và ra, nên ở đây sử dụng bộ đếm lên và xuống. Ngoài ra, để đơn giản khi khởi động lại PLC thì bộ đếm xóa về 0, ta có thông tin cho các ngõ vào của bộ đếm như sau:

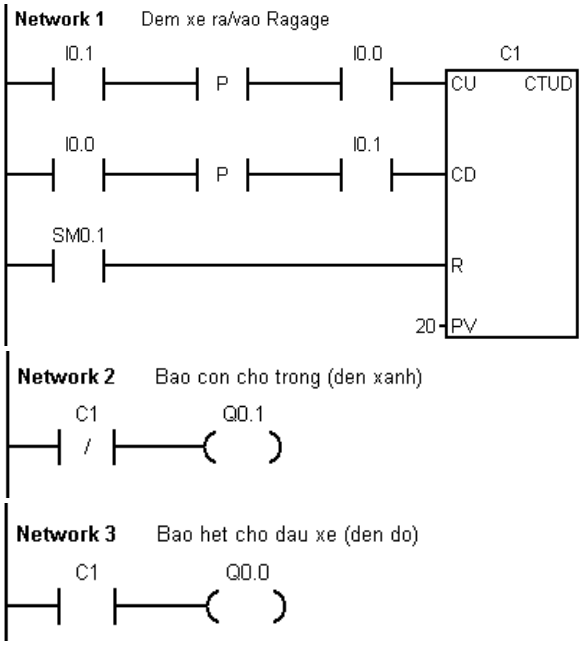
- Ngõ vào đếm lên CU:= Tín hiệu xe vào
- Ngõ vào đếm xuống:= Tín hiệu xe ra
- Ngõ vào giá trị đặt trước PV:= 20
- Ngõ vào xóa bộ đếm R:= SM0.1

* Đèn báo Garage còn chỗ trống (đèn xanh):= $\overline{C1}$

* Đèn báo Garage hết chỗ trống (đèn đỏ):= C1.

Chương trình

LAD



STL

Network 1 Dem xe ra/vao Ragage
LD I0.1
EU
A I0.0
LD I0.0
EU
A I0.1
LD SM0.1
CTUD C1, 20

Network 2 Bao con cho trong (den xanh)
LDN C1
= Q0.1

Network 3 Bao het cho dau xe (den do)
LD C1
= Q0.0

10.6 Câu hỏi và bài tập

BT10.6.1 Điều khiển bồn sấy

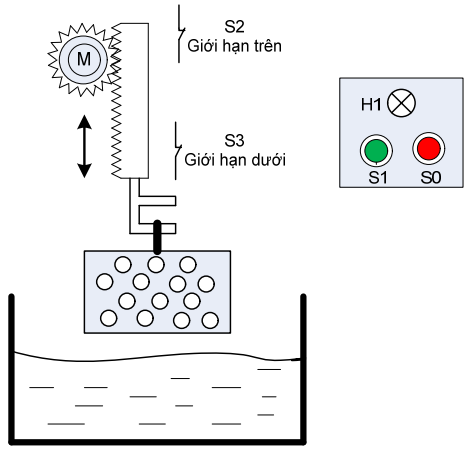
Một bồn sấy hoạt động như sau:

Khi ấn nút khởi động S1 (NO), thì bồn sấy quay phải 20s, tự động dừng lại 5s, sau đó quay trái 20s, tự động dừng lại 5s. Quá trình cứ lặp đi lặp lại cho đến khi ấn nút dừng S2 (NC) hoặc sau thời gian 20 chu kỳ lặp sẽ tự động dừng lại. Yêu cầu:

1. Lập bảng xác định vào ra (khi lập bảng chú ý liệt kê luôn các bit nhớ, bộ đếm, timer và ý nghĩa của chúng trong chương trình).
2. Lập bảng nối dây với PLC
3. Viết chương trình điều khiển và nạp vào PLC để kiểm tra.

BT10.6.2 Điều khiển bể ăn mòn

Một bể chứa dung dịch ăn mòn để ăn mòn phần đồng còn thừa trên tấm mạch in. Giỏ chứa các tấm mạch được treo vào một cần như hình 10.3. Khi ấn nút khởi động S1 (NO) thì cần hạ giỏ xuống đến giới hạn dưới S3 (NC) để đặt các tấm mạch in ngập trong dung dịch ăn mòn. Sau thời gian 15s thì cần nâng lên đến giới hạn trên của cần S2 (NC) thì tự động hạ xuống trở lại. Chu kỳ lặp lại được 6 lần thì tự động dừng hoặc có thể ấn nút dừng S0 (NC). Khi hệ thống đang hoạt động thì đèn báo H1 sáng.



Hình 10.3 Sơ đồ công nghệ bể ăn mòn

Bảng xác định vào/ra

Ký hiệu	Địa chỉ	Chú thích
S0	I0.0	Nút nhấn dừng, NC
S1	I0.1	Nút nhấn khởi động, NO
S2	I0.2	Công tắc hành trình giới hạn trên, NC
S3	I0.3	Công tắc hành trình giới hạn dưới, NC
K1	Q0.0	Contactor điều khiển động cơ kéo giỏ lên
K2	Q0.1	Contactor điều khiển động cơ hạ giỏ xuống
H1	Q0.2	Đèn báo hệ thống hoạt động

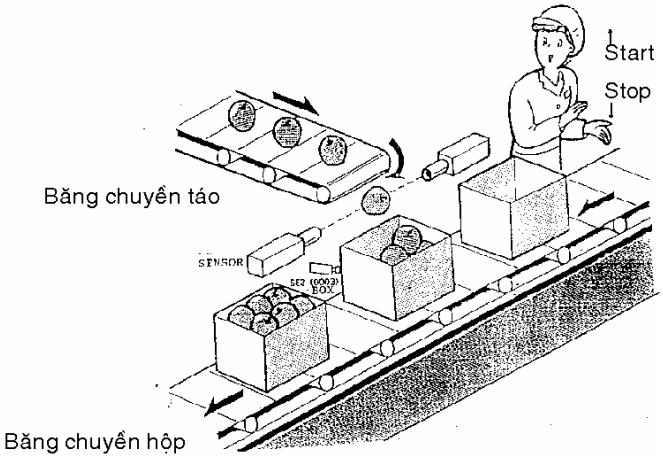
Yêu cầu:

1. Vẽ sơ đồ nối dây PLC
2. Viết chương trình điều khiển

BT10.6.3 Kiểm soát băng chuyền sản phẩm

Một hệ thống băng chuyền sản phẩm được cho theo sơ đồ công nghệ như hình vẽ 10.4.

Khi ấn nút "start" thì băng chuyền thùng hoạt động. Khi thùng đụng công tắc hành trình S3 (NO) thì băng chuyền thùng dừng lại, băng chuyền sản phẩm đã đóng gói bắt đầu chuyển động. Cảm biến S2(NC) được dùng để đếm số lượng sản phẩm. Khi đếm được 12 sản phẩm thì băng chuyền sản phẩm dừng và băng chuyền thùng lại bắt đầu chuyển động. Bộ đếm được đặt lại và quá trình vận hành lập lại cho đến khi ấn nút "stop" (NC).



Hình 10.4 Sơ đồ công nghệ băng chuyền sản phẩm

Bảng xác định vào/ra

Ký hiệu	*Địa chỉ*	*Chú thích*
Start	I0.0	Nút nhấn khởi động hệ thống, NO
Stop	I0.1	Nút nhấn dừng hệ thống, NC
S2	I0.2	Cảm biến đếm số lượng sản phẩm, NC
S3	I0.3	Công tắc hành trình nhận biết thùng, NO
K1	Q0.0	Contactor điều khiển động cơ băng chuyền thùng
K2	Q0.1	Contactor điều khiển động cơ băng chuyền sản phẩm

Yêu cầu:

1. Vẽ sơ đồ nối dây PLC
2. Viết chương trình điều khiển

11 Điều khiển trình tự

11.1 Cấu trúc chung của một chương trình điều khiển

Trong phần này đề cặp đến việc tổ chức và cấu trúc cho chương trình PLC, nghĩa là trong chương trình điều khiển gồm các phần có liên quan đến các vấn đề như các chế độ hoạt động, các chức năng cơ bản, trình tự xử lý, kích hoạt các ngõ ra, hiển thị trạng thái theo trình tự sau:

1. Bắt đầu chương trình
2. Các chế độ hoạt động và các chức năng cơ bản
    - Khởi tạo vị trí cơ bản.
    - Các điều kiện cho phép của ngõ ra.
    - Mạch logic điều khiển.
    - Kích hoạt các ngõ ra.
    - Xuất các chỉ thị, chỉ báo.
3. Kết thúc chương trình.

- ***Đoạn chương trình điều khiển chế độ hoạt dộng***

    - *Khởi tạo vị trí cơ bản*

Các thiết bị vật lý được điều khiển đều có vị trí cơ bản, ví dụ khi các cơ cấu tác động ở các trạng thái OFF và các công tắc hành trình ở vị trí hở. Tất cả các yếu tố này có thể được tổ hợp logic với nhau để báo hiệu và khởi tạo vị trí cơ bản, và được lập trình như là một bước trong chuỗi trình tự.

- *Đoạn chương trình chức năng khởi động hay dừng quá trình điều khiển.*

Hầu hết các điều khiển trong công nghiệp đều có nút khởi động (START) và nút dừng (STOP) mà có thể lập trình cho hành vi của chúng. Các nút này được lập trình bằng các tiếp điểm logic thực hiện khởi động hay dừng toàn bộ hoạt động điều khiển của PLC. Cũng có thể có một công tắc bằng tay để cho phép hay không cho phép các ngõ ra, dùng khi kiểm tra chương trình.

- ***Đoạn chương trình xử lý điều khiển***

Đây là phần chính của chương này, bao gồm việc thiết kế và lập trình các điều khiển dùng cơ chế trình tự hay logic tổ hợp. Các kết quả của sự tổ hợp logic trên thường không trực tiếp kích các cơ cấu chấp hành, mà thông qua các ô nhớ trung gian.

- ***Đoạn chương trình kích các ngõ ra***

Các tín hiệu ngõ ra dùng để kích cơ cấu tác động được khoá lẫn bởi các ô nhớ trung gian hình thành từ các đoạn chương trình xử lý điều khiển.

- ***Đoạn chương trình xuất các chỉ thị, chỉ báo***

Các trạng thái của quá trình hoạt động thường được biểu thị bằng đèn, chuông…để người vận hành máy có các quyết định thích hợp.

Việc lập trình theo cấu trúc như trên nhằm làm cho chương trình điều khiển có độ tin cậy cao hơn, dễ hiểu hơn, cho phép xác định lỗi nhanh chóng và rút ngắn được thời gian bảo trì, sửa chữa.

11.2 Điều khiển trình tự

11.2.1 Giới thiệu

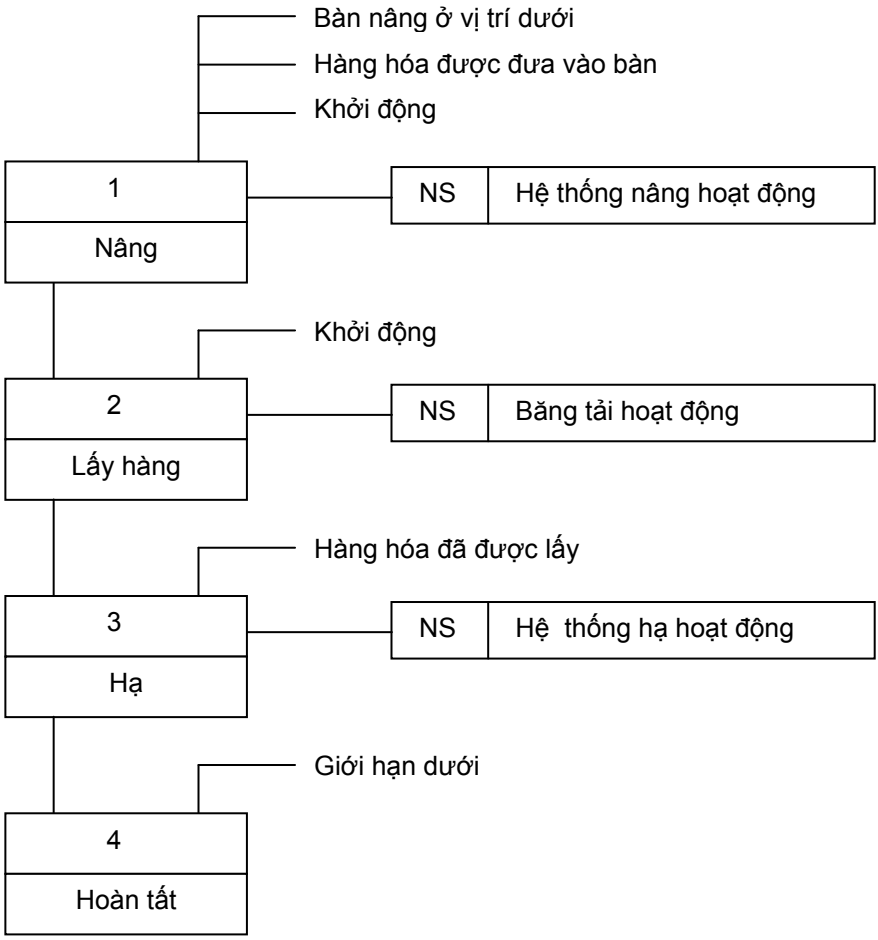
Trong công nghiệp, hầu hết các dự án điều khiển xảy ra một cách trình tự, khâu xử lý sau chậm hơn khâu xử lý trước một khoảng thời gian xác định. Ví dụ như quá trình chuyển động mới bắt đầu nếu như một quá trình khác được kết thúc.

Vấn đề này có thể được giải quyết bằng điều khiển liên kết, với việc kết nối cứng các điều kiện trong chương trình. Nhưng ở đây chỉ ra rằng từ một khuôn khổ điều khiển đã biết thì việc giải quyết vấn đề bằng điều khiển liên kết là rất khó đọc chương trình và việc tìm lỗi phải mất nhiều thời gian.

Nếu một dự án được thực hiện theo phương pháp điều khiển trình tự thì cấu trúc chương trình có thể nhận biết một cách dễ dàng và dự án có thể được biểu diễn bằng hình ảnh. Điều khiển trình tự giúp cho người đọc đọc chương trình một cách dễ dàng, chương trình điều khiển được trình bày theo cấu trúc, ưu điểm của nó là giúp cho việc lập trình, thay đổi và tìm lỗi các dự án một cách có hiệu quả.

Để dễ hiểu ta xét *Một hệ thống nâng hàng hoạt động như sau :*

Bàn nâng ở vị trí dưới và hàng hoá sẽ được đưa vào bàn nâng. Nếu nút khởi động được ấn thì bàn nâng được hệ thống nâng đưa lên cao, khi lên đến giới hạn trên thì hệ thống nâng ngừng lại và băng tải trên bàn nâng hoạt động kéo hàng hoá đưa sang bộ phận khác. Sau khi hàng hoá được lấy xong thì băng tải dừng, lúc này bàn sẽ được hạ xuống khi đến vị trí dưới thì dừng lại,và một quá trình mới lại bắt đầu. Từ yêu cầu công nghệ của hệ thống nâng hàng này ta có thể biểu diễn theo phương pháp điều khiển trình tự như ở hình 11.1.



Hình 11.1: Ví dụ hệ thống nâng hàng được biểu diễn theo sơ đồ chức năng trong điều khiển trình tự.

Ưu điểm của phương pháp điều khiển trình tự là:

- Thiết kế, lập trình nhanh và đơn giản.
- Cấu trúc chương trình rõ ràng.
- Thay đổi dễ dàng trình tự thực hiện.
- Nhận biết nhanh chóng các nguyên gây ra lỗi.
- Nhiều kiểu hoạt động khác nhau có thể thực hiện được.

Từ các ưu điểm này mà trong thực tế rất nhiều bài toán điều khiển được giải quyết bằng phương pháp điều khiển trình tự. Điều khiển trình tự có thể chia làm hai loại:

- Điều khiển trình tự theo thời gian .

- Điều khiển trình tự theo quá trình .

Điều khiển trình tự theo thời gian :

Ở điều khiển trình tự theo thời gian thì điều kiện chuyển tiếp chỉ phụ thuộc vào thời gian. Các khâu định thời, bộ đếm thời gian…để tạo ra điều kiện chuyển tiếp.

Điều khiển trình tự theo quá trình :

Ở điều khiển trình tự theo quá trình thì điều kiện chuyển tiếp phụ thuộc vào các tín hiệu của thiết bị được điều khiển. Các thông báo về từ các sự kiện của xử lý có thể là vị trí van các bộ giám sát hoạt động, lưu lượng áp suất, nhiệt độ, độ dẫn, độ nhờn …Trong nhiều trường hợp các thông báo về từ việc xử lý phải được biến đổi thành tín hiệu nhị phân .

Một dạng của điều khiển trình tự phụ thuộc vào quá trình xử lý của điều khiển theo hành trình, điều kiện chuyển tiếp của nó chỉ phụ thuộc vào các tín hiệu hành trình của thiết bị được điều khiển .

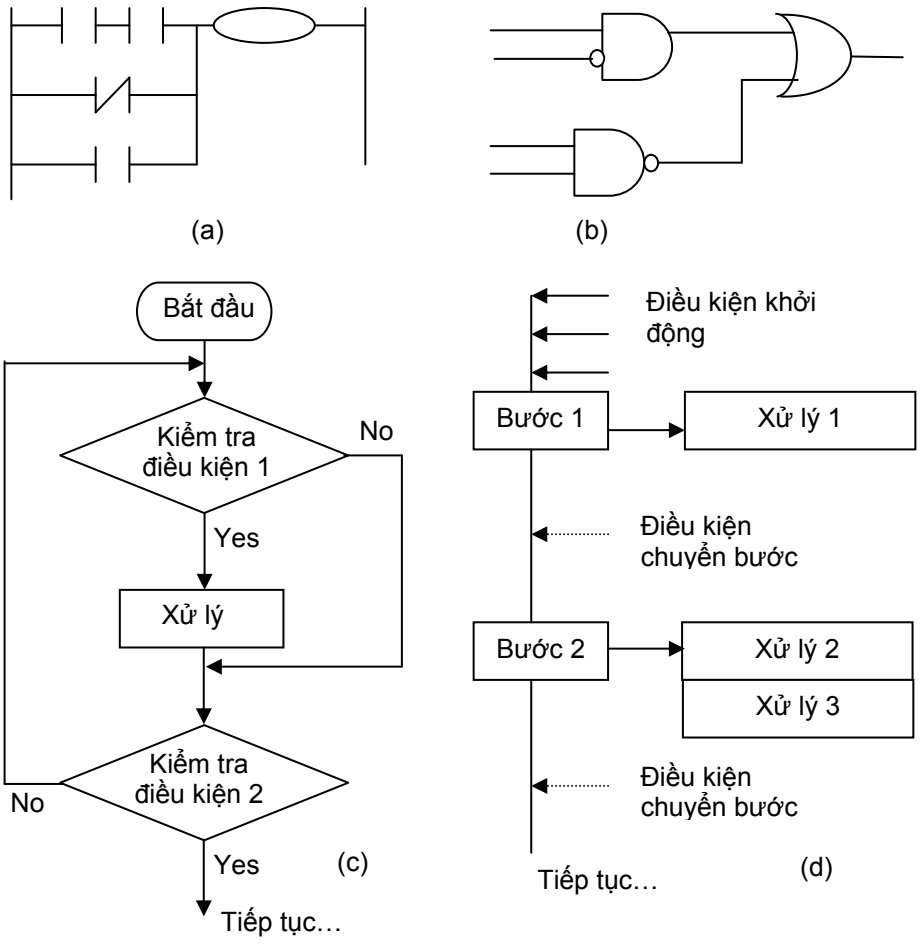
11.2.2 Phương pháp lập trình điều khiển trình tự

Các bước thiết kế chương trình trình tự cho PLC như sau :

- Quá trình điều khiển được diễn đạt bằng lời.
- Sự mô tả đó được chuyển sang dạng lưu đồ hay sơ đồ chức năng.
- Đến giai đoạn này, các điều kiện logic dễ dàng được xác định, sau đó được chuyển sang biểu thức boolean biểu diễn từng trạng thái của quá trình trình tự.
- Cuối cùng biểu thức boolean được chuyển đổi sang chương trình trong PLC.

Sự diễn đạt bằng lời hay ghi ra giấy mô tả quá trình điều khiển thường dài, khó theo dõi và không chính xác. Như đã đề cặp, toàn bộ quá trình điều khiển sẽ dễ hiểu hơn khi nó chia thành những đơn vị con (sub-units) hay xử lý con (sub- processor). Mỗi đơn vị con sau đó có thể được xây dựng theo dạng trình tự và khóa lẫn để thực hiện một chức năng nào đó theo yêu cầu. Cần có các phương pháp để mô tả hệ thống trình tự như trên sao cho rõ ràng và dễ theo dõi quá trình hoạt động.

Các phương pháp diễn đạt có thể tuỳ chọn: logic relay (relay logic diagram), cổng logic (logic schematics), lưu đồ (flowcharts) và sơ đồ chức năng (function charts) như hình 11.2. Các phương pháp này không thay thế cho bước diễn đạt bằng lời mà nó hỗ trợ rất nhiều cho bước này. Việc áp dụng phương pháp nào tuỳ thuộc chủ yếu vào kinh nghiệm về phương pháp đó. Người phân tích thiết kế hệ thống có kiến thức tốt về kỹ thuật số hay về máy tính thì thường dùng 3 phương pháp sau, còn phương pháp logic relay được dùng đối với những người quen với thiết kế mạch relay.



Hình 11.2 : Các phương pháp mô tả hệ thống điều khiển logic:

(a) logic relay; (b) cổng logic; (c) lưu đồ; (d) sơ đồ chức năng

- ***Phương pháp logic relay và cổng logic***

Cả hai phương pháp có liên hệ trực tiếp đến mạch vật lý, nên việc dùng PLC để thay thế hệ thống relay truyền thống là lý tưởng. Các phương pháp này thường dùng cho hệ thống điều khiển dùng tổ hợp các ngõ vào hay các hệ thống trình tự qui mô nhỏ vì sơ đồ biểu diễn cho trình tự qui mô lớn phức tạp và khó theo dõi.

- ***Phương pháp biểu diễn theo lưu đồ***

Phương pháp này thường dùng khi thiết kế phần mềm cho máy tính, nhưng lại phổ biến để biểu diễn trình tự hoạt động của hệ thống điều khiển. Lưu đồ có quan hệ trực tiếp đến sự mô tả bằng lời hệ thống điều khiển, chỉ ra

từng điều kiện cần kiểm tra từng bước và các xử lý trong các bước đó theo chuỗi trình tự. Các xử lý trong lưu đồ được ghi trong 1 ô chữ nhật, trong khi các điều kiện được ghi vào ô hình thoi. Tuy nhiên, phương pháp này chiếm nhiều không gian khi biểu diễn hệ thống điều khiển lớn và trở nên nặng nề.

- ***Phương pháp sơ đồ chức năng***

Phương pháp này ngày càng trở nên phổ biến để biểu diễn các hoạt động trình tự, cho phép thể hiện chi tiết về các xử lý cũng như trình tự các hoạt động trong quá trình điều khiển. Với với cách dùng các ký hiệu gọn và cô đọng, phương pháp này có được ưu điểm của các phương pháp trên, việc biểu diễn bước tiến trình hoạt động mạch lạc và rõ ràng. Trong từng bước ta có thể ghi ra các điều kiện set và reset, điều kiện chuyển trạng thái và các tín hiệu điều khiển khác. Sơ đồ chức năng còn thể hiện đắc lực khi kiểm tra và thử hệ thống.

- ***Đại số Boolean***

Cho dù dùng phương pháp nào đi nữa, một khi các chức năng đã được đặc tả rõ ràng thì chúng phải được chuyển đổi sang dạng mà từ đó có thể chuyển thành chương trình PLC. Quá trình này được thực hiện bằng cách chuyển đổi các chức năng thành 1 chuỗi liên tiếp biểu thức boolean, và từ đó chuyển thành ngôn ngữ PLC. Một khi quen với kỹ thuật này, ta có thể dễ dàng chuyển đổi sự đặc tả chức năng thành biểu thức boolean bất kể là nó được đặc tả bằng phương pháp nào.

Ta cũng có thể đặc tả toàn bộ hệ thống điều khiển logic bằng biểu thức boolean, mặc dù việc dùng biểu thức Boolean thường kém hiệu quả về mặt thời gian thiết kế và không dễ hiểu đối với những người chưa có kinh nghiệm về các hệ thống điều khiển. Giải pháp dùng Boolean dù sao đi nữa cũng tiết kiệm được không gian biểu diễn trên giấy khi thiết kế.

Trong các phương pháp lập trình cho điều khiển trình tự trên thì phương pháp sơ đồ chức năng có ưu điểm hơn các phương pháp khác. Cho nên chương này chọn phương pháp sơ đồ chức năng để làm cơ sở chính cho việc thiết kế điều khiển trình tự.

11.3 Các thủ tục tổng quát để thiết kế bài toán trình tự

Trong bài toán điều khiển trình tự, để thực hiện một cách có hệ thống công việc điều khiển và tránh tối đa những thiếu sót, nhầm lẫn thì thủ tục để thiết kế bài toán trình tự bao gồm các bước như sau:

Bước 1: *Xây dựng sơ đồ phối hợp thao tác công nghệ của máy hoặc hệ thống thiết bị cần điều khiển.*

Đây là công việc có yêu cầu tương tự như khi bắt tay vào việc thiết kế một máy mới. Người thực hiện sẽ căn cứ vào yêu cầu hoạt động của máy để từ đó hình dung và phân tích ra một trình tự các thao tác thật chi tiết của các

khâu chấp hành hoặc từng bộ phận chấp hành của máy cũng như sự hoạt động giữa chúng.

Quá trình phân tích và thực hiện việc phối hợp các chuyển động hoặc các thao tác thường được thực hiện dưới dạng một sơ đồ phối hợp. Sơ đồ được thực hiện dưới dạng các dải hình chữ nhật đặt kế tiếp nhau. Mỗi dải tượng trưng cho diễn biến theo thời gian quá trình hoạt động của một khâu chấp hành hoặc một bộ phận chấp hành nhằm thực hiện một thao tác công nghệ nào đó.

Sơ đồ phối hợp các thao tác công nghệ cho phép người thiết kế hình dung toàn bộ quá trình hoạt động của máy hoặc của hệ thống thiết bị bao gồm trình tự các thao tác và thời điểm bắt đầu cũng như kết thúc thực hiện của từng thao tác. Sơ đồ phối hợp này sẽ là cơ sở cho việc soạn thảo chương trình điều khiển trên PLC cũng đồng thời là tài liệu gốc cho việc hiệu chỉnh sự làm việc máy hoặc hệ thống về sau.

Bước 2: *Lập sơ đồ khối điều khiển trình tự.*

Căn cứ vào sơ đồ phối hợp các hoạt động hoặc các thao tác của các bộ phận chấp hành trên máy thiết kế, người cán bộ kỹ thuật sẽ thực hiện một công việc tương tự tiếp theo là lập sơ đồ khối điều khiển trình tự (dạng lưu đồ (flowchart) hoặc sơ đồ chức năng (funtion-chart)). Công việc này là một bước tiếp cận hơn nữa của quá trình điều khiển. Tuỳ theo mức độ quen sử dụng cách biểu diễn nào mà người thiết kế sẽ lựa chọn các phương pháp biểu diễn quá trình điều khiển để mô tả chuỗi trình tự các thao tác công nghệ cũng như các tín hiệu điều khiển cho từng thao tác.

Bước 3: *Chuẩn bị phần cứng và mô tả các tham số vào/ra.*

Công việc lựa chọn các cơ cấu chấp hành như lựa chọn các loại động cơ, xylanh khí nén hoặc xylanh dầu ép, lựa chọn các loại van điều khiển,…, có liên quan mật thiết với quá trình điều khiển đã tổng hợp do nhiều yếu tố như đặc tính kỹ thuật của cơ cấu tác động có phù hợp với máy thiết kế hay không, kết cấu có phù hợp hay không, không gian có cho phép bố trí loại cơ cấu tác động đó hay không; và một yếu tố quan trọng có tính chất quyết định là thời gian và tốc độ đáp ứng của cơ cấu tác động được lựa chọn có phù hợp, thỏa mãn với yêu cầu phối hợp trên máy hay không.

Người thiết kế phải lựa chọn kỹ để tìm kiếm các cơ cấu tác động phù hợp nhất và mô tả đầy đủ các thông số kỹ thuật của cơ cấu tác động, chẳng hạn như các giá trị điện áp, dòng điện tác động vào động cơ điện hay tác động vào các van điện từ điều khiển các van khí nén. Các tín hiệu trên có liên quan mật thiết với các tín hiệu ngõ ra của PLC. Tương tự, các tín hiệu từ các cảm biến; phản ánh trạng thái của cơ cấu tác động, được đưa đến các ngõ vào của PLC.

Thông qua việc lựa chọn và mô tả các tham số vào/ ra này, người thiết kế sẽ cung cấp các số liệu cần thiết cho việc thiết kế các mạch giao tiếp giữa PLC với mạch công suất của các cơ cấu tác động, xác định số ngõ vào/ ra để lựa chọn PLC thích hợp.

Bước 4: *Lập trình.*

Với đầy đủ các dữ liệu được cung cấp từ các bước thực hiện ở trên, công việc tiếp theo của người lập trình là soạn thảo chương trình điều khiển cho PLC để thực hiện việc điều khiển máy hoặc hệ thống hoạt động đúng cho chu trình đã thiết kế. Tuỳ theo khả năng quen sử dụng loại ngôn ngữ lập trình trên PLC nào mà người lập trình sẽ chọn lựa để soạn thảo chương trình. Với các chương trình đơn giản, các phần mềm của các hãng cho phép biên dịch được chương trình được viết từ ngôn ngữ này sang ngôn ngữ khác.

Bước 5: *Chạy thử và hoàn chỉnh chương trình.*

Đây là công việc hết sức tự nhiên phải thực hiện sau khi lập trình. Việc chạy thử chương trình được thực hiện trong 2 chế độ:

Chế độ giả lập (chế độ offline): Cho chạy chương trình và theo dõi đáp ứng của các ngõ ra thông qua các đèn LED. Đèn LED ở ngõ ra cụ thể sẽ biểu thị cho tín hiệu xuất ở ngõ ra cho cơ cấu tác động và đáp ứng của chúng.

Chế độ thực (chế độ online): Sau khi đã chạy thử và điều chỉnh chương trình trong chế độ giả lập hoàn hảo. Chuyển chế độ hoạt động trên PLC và nối phần mạch giao tiếp với mạch công suất để điều khiển máy chạy trong chế độ thực. Trong chế độ này, với các đáp ứng thực của các cơ cấu tác động khi không tải và khi có tải sẽ giúp cho người lập trình hiệu chỉnh chương trình lần cuối trước khi đưa vào vận hành thực sự trong sản xuất.

11.4 Cấu trúc của bài toán điều khiển trình tự

Một bài toán điều khiển trình tự có thể chia làm 4 phần :

- Chuỗi trình tự
- Kiểu hoạt động
- Các thông báo
- Kích hoạt ngõ ra .

Mối liên hệ giữa các phần được biểu diễn theo sơ đồ hình 11.3.

11.4.1 Chuỗi trình tự

Hạt nhân của điều khiển trình tự là chuỗi trình tự. Chương trình điều khiển theo các bước đã biết được xử lý ở đây. Các bước trình tự riêng lẻ được kích hoạt phụ thuộc vào điều kiện chuyển tiếp.

11.4.2 Kiểu hoạt động

Điều kiện cho các chế độ hoạt động khác nhau được xử lý trong phần kiểu hoạt động. Các loại hoạt động sau thường được sử dụng trong kỹ thuật điều khiển .

a. Chế độ tự động:

Trong chế độ tự động, sau khi tín hiệu khởi động được kích hoạt thì trình tự điều khiển xảy ra ở các chuỗi trình tự hoàn toàn tự động không cần đến bảng điều khiển . Cơ cấu chấp hành sẽ được điều khiển theo chuỗi trình tự .

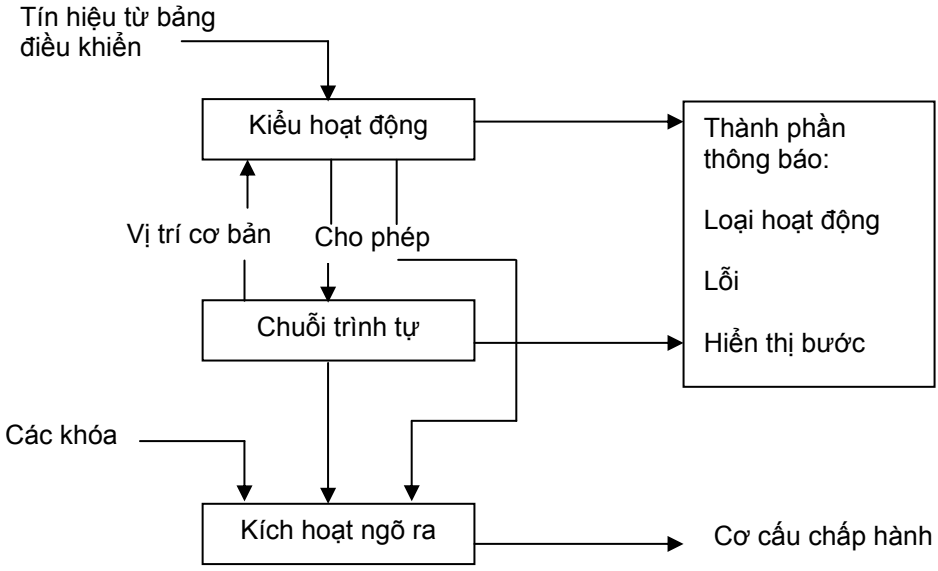
b. Chế độ tay hay hoạt động theo bước

Trong chế độ hoạt theo từng bước thì chuỗi trình tự được chuyển tiếp bằng tay .Ở chế độ này còn có thêm sự phân biệt : chuyển tiếp có điều kiện và chuyển tiếp không điều kiện. Chế độ làm việc này dùng để kiểm tra chương trình trong vận hành và xử lý lỗi .

c. Chế độ thiết bị

Trong chế độ này, từng cơ cấu chấp hành có thể được tác động bằng tay mà không phụ thuộc vào chương trình điều khiển. Các khóa an toàn vẫn có hiệu lực trong chế độ này.

Các chế độ làm việc khác nhau được điều khiển ở bảng điều khiển. Tùy theo chế độ hoạt động được điều chỉnh mà chuỗi trình tự xuất lệnh và phần thông báo tiếp nhận tín hiệu dưới dạng tín hiệu sẵn sàng, tín hiệu chuyển tiếp, tín hiệu khóa và tín hiệu hiển thị.



Hình 11.3: Cấu trúc của một bài toán điều khiển trình tự

Đối với mỗi chế độ hoạt động thường phải chú ý đến qui tắc an toàn.

Các qui tắc an toàn nhất có thể được tóm tắt sau đây :

- Các tình trạng nguy hiểm gây tai nạn cho người, máy móc cũng như vật liệu phải được tránh.
- Máy móc phải được ở trạng thái đứng yên (không hoạt động) khi nguồn có điện trở lại nếu xảy ra tình trạng mất điện.

- Các công tắc dừng khẩn cấp và các công tắc giới hạn an toàn phải luôn ở trạng thái sẵn sàng khi có sự cố. Bởi vậy các thiết bị bảo vệ này cần phải có tác dụng trực tiếp đến phần công suất của cơ cấu chấp hành.
- Trong trường hợp xảy ra sự cố đứt dây hay nối đất thì hệ thống không được phép tự khởi động cũng như không được phép hoạt động.

Các qui tắc chung này được thực hiện tùy theo mỗi nhiệm vụ điều khiển.

11.4.3 Các thông báo

Trong phần chương trình này, các thông báo cần thiết của điều khiển được đặt ở bảng điều khiển. Các thông báo điều khiển bao gồm chỉ thị chế độ hoạt động được đặt, chỉ thị số bước hiện hành và chỉ thị lỗi xảy ra.

11.4.4 Kích hoạt ngõ ra

Các lệnh thực hiện các bước đơn của chuỗi trình tự được kích hoạt trong phần chương trình xuất lệnh, đồng thời nó được liên kết với tín hiệu sẵn sàng của phần chế độ hoạt động và các tín hiệu khóa từ quá trình xử lý. Ở đây cần lưu ý đến các lệnh điều khiển bằng tay của cơ cấu chấp hành trong chế độ hoạt động thiết bị.

*** Đặc điểm của điều khiển trình tự:**

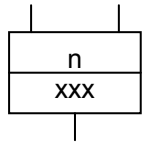
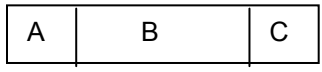
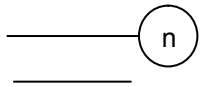
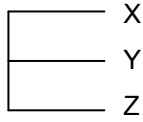
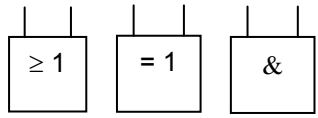
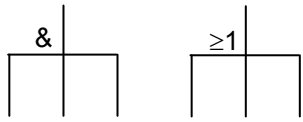
Các đặc điểm quan trọng nhất của điều khiển trình tự có thể kể ra như sau :

- Các bước trình tự được thực hiện kế tiếp nhau theo một trình tự xác định cho trước. Trình tự này chỉ có thể bị ảnh hưởng khi có tín hiệu “cho phép chuỗi trình tự ”và “reset chuỗi trình tự”.
- Khi có tín hiệu “cho phép chuỗi trình tự” và điều kiện chuyển tiếp được tác động thì bước sau được thực hiện.
- Việc đóng mạch cho bước kế tiếp phụ thuộc vào điều kiện chuyển tiếp được điều khiển từ quá trình hay thông qua các điều kiện thời gian. Khi bước sau được set thì bước trước đó phải bị reset.
- Các lỗi trong một chuỗi trình tự có thể được xác định và phân tích một cách nhanh chóng. Việc tìm lỗi giới hạn trong các bước được set và điều kiện chuyển tiếp của chúng, các lỗi được tìm ra ở đây.
- Khâu an toàn được thiết lập không phụ thuộc vào trình tự chương trình và tín hiệu của nó được liên kết với các khâu tương ứng của phần kích hoạt ngõ ra.

11.5 Các ký hiệu

Việc biểu diễn điều khiển trình tự được thực hiện theo sơ đồ khối. Nó biểu diễn vấn đề điều khiển cần giải quyết, không phụ thuộc vào cách thức

thực hiện của nó như chế độ hoạt động, sự lắp đặt dây dẫn cũng như vị trí lắp đặt. Sơ đồ khối bổ sung thêm cách mô tả hoạt động. Nhờ đó các yêu cầu cần thiết trong hoạt động và công nghệ được biểu diễn cô đọng, rỏ ràng. Như vậy sơ đồ khối cũng là một công cụ thích hợp diễn tả qui trình công nghệ giữa nhà sản xuất và người sử dụng. Dạng biểu diễn cho điều khiển trình tự được cho theo *bảng 11.1.*

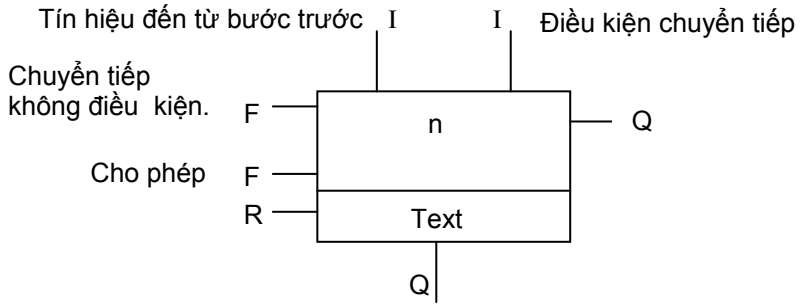
Ý nghĩa	*Ký hiệu*
Ký hiệu chung cho bước n : Bước thực hiện xxx: Tên bước thực hiện	
Lệnh: A : Loại lệnh. B : Tên gọi và tác dụng của các lệnh tới thiết bị được giải thích bằng chữ (ví dụ : băng tải dừng) C : Vị trí ngắt của lệnh.	
Đường dẫn tác dụng n : số kí hiệu của vị trí ngắt	
Tóm tắt của các đường dẫn tác dụng X,Y,Z : Tên các điều kiện được mô tả ngắn hay ở dạng chữ.	
Ký hiệu các cổng logic. ≥ 1: Cổng OR & : Cổng AND =1: Cổng XNOR	
Các rẻ nhánh &: AND ≥ 1: OR	

Bảng 11.1: Các ký hiệu

11.6 Bước trình tự

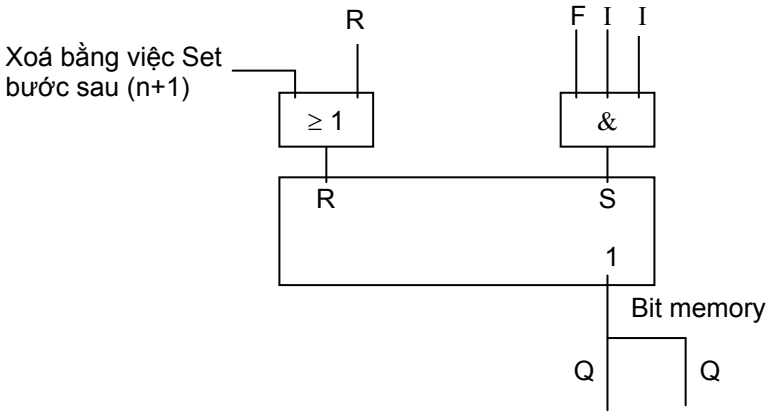
Một bước trình tự được cho như hình vẽ 11.4. Phần trên có kí hiệu “n” là số bước, phần dưới dùng để mô tả ngắn chức năng của bước. Bước “n” được

set nếu tất cả các ngõ vào “I” có giá trị logic “1”. Các ngõ ra “Q” ở bước được set có giá trị “1” và sẵn sàng để set cho bước tiếp theo (n+ 1). Bước sẽ bị reset nếu như bước sau (n+ 1) được set. Ngoài ra một bước có thể bị ảnh hưởng bởi tín hiệu reset “ R” và tín hiệu tự do “ F “.



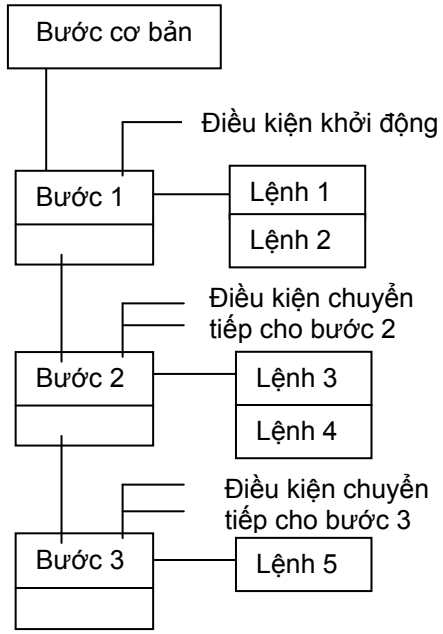
Hình 11.4: Ký hiệu của một bước với các ngõ vào và ra

Ví dụ sau là một chương trình biểu diễn một bước tương ứng trong điều khiển trình tự. Đây là trường hợp đơn giản nhất gồm có một khâu nhớ với cổng AND đặt ở ngõ “S”. Khâu trình tự này có thể bị Reset với liên kết OR thêm vào ở ngõ “R”.

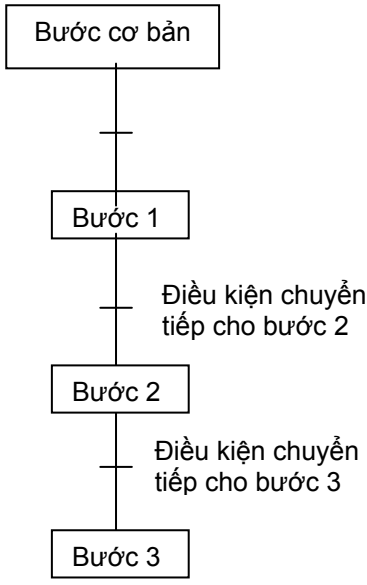


Trong thể hiện chương trình thì một bước được set tương ứng với một bit memory.

Cấu trúc của chuỗi tuần tự tương ứng trình tự các bước điều khiển của dự án. Có 2 phương pháp biểu diễn :



Sơ đồ biểu diễn theo DIN 40719



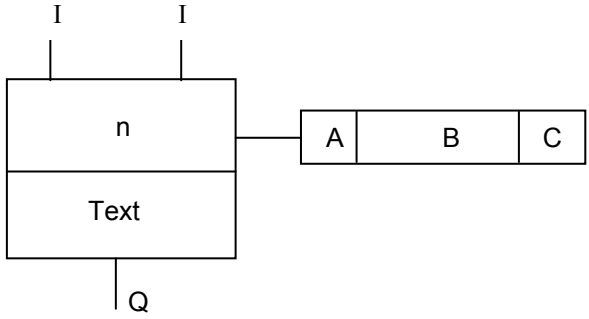
Sơ đồ biểu diễn theo IEC-SC65A

Hình 11.5: Các cách biểu diễn theo các chuẩn khác nhau

Ở hai phương pháp biểu diễn trên, chương này chỉ trình bày sơ đồ biểu diễn theo DIN 40719.

11.7 Các lệnh biểu diễn trong sơ đồ chức năng

Các lệnh cho ở ngõ ra của một bước ở phần kích hoạt ngõ ra của khâu điều chỉnh được điền vào dòng bên phải của hình chữ nhật của ký hiệu bước. Ký hiệu lệnh theo bước được ký hiệu như sau:



Vùng A: Cho biết loại lệnh.

Vùng B: Chỉ tác dụng của lệnh giải thích bằng chữ (ví dụ động cơ có điện, đèn H1 sáng . . .).

Vùng C: Ký hiệu vị trí ngắt của lệnh xuất. Nếu vị trí ngắt không tồn tại thì có thể bỏ vùng này.

Mỗi ký hiệu có thể sử dụng nhiều ngõ vào với các tác dụng khác nhau. Các tác dụng đặc biệt được ký hiệu thông qua chữ cái:

Ngõ vào cho phép: “F”.

Ngõ vào reset: “R”.

Ngõ vào cho các thông báo lại: “RC”.

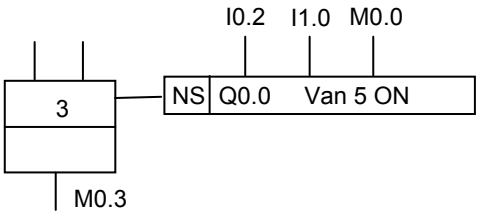
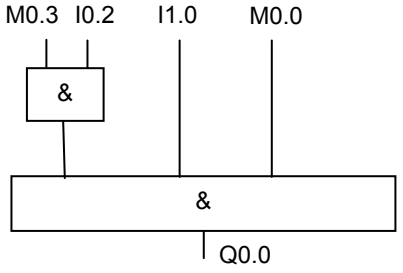
Một ký hiệu lệnh cũng được quyết định về các ngõ ra, hoặc được biểu diễn trực tiếp bằng đường dẫn tác dụng hoặc số lệnh của nó được điền vào vùng C. Các ngõ ra được ký hiệu “RC” dùng để thông báo lại từ khâu điều chỉnh.

Các loại lệnh sau có thể được điền vào vùng A:

Lệnh	*Ý nghĩa*
D	Lệnh trì hoãn thời gian
SD	Lệnh trì hoãn thời gian và được duy trì
NSD	Lệnh trì hoãn thời gian và không được duy trì
NS	Lệnh không được duy trì
R	Reset lại các phần tử đã bị set
S	Lệnh được duy trì
SH	Lệnh được duy trì trong trường hợp mất điện
T	Lệnh giới hạn thời gian
ST	Lệnh được duy trì và giới hạn thời gian

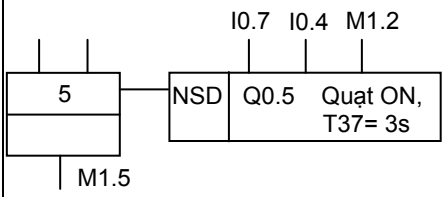
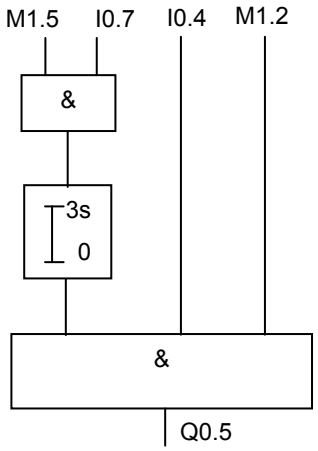
*** Lệnh NS** (không được duy trì)

Lệnh NS chỉ có tác dụng khi nào bước phụ thuộc được kích hoạt. Nếu bước sau được đóng mạch thì lệnh NS không còn tác dụng nữa.

Ví dụ lệnh	*Biểu diễn sơ đồ logic*
	

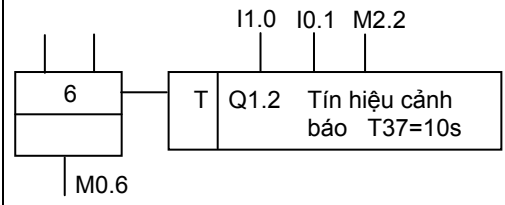
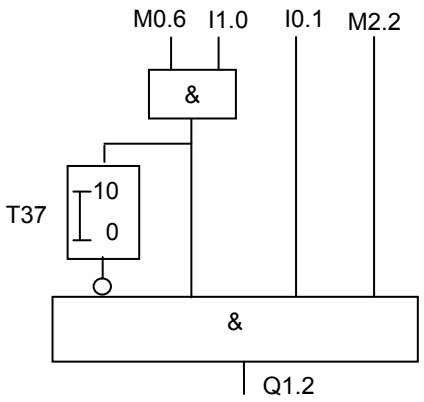
*** Lệnh NSD** (trì hoãn thời gian và không được duy trì)

Lệnh NSD tác dụng như lệnh NS, việc xuất lệnh xảy ra tùy thuộc vào quá trình của thời gian trì hoãn "t" được điều chỉnh trước.

Ví dụ lệnh	*Biểu diễn sơ đồ logic*
	

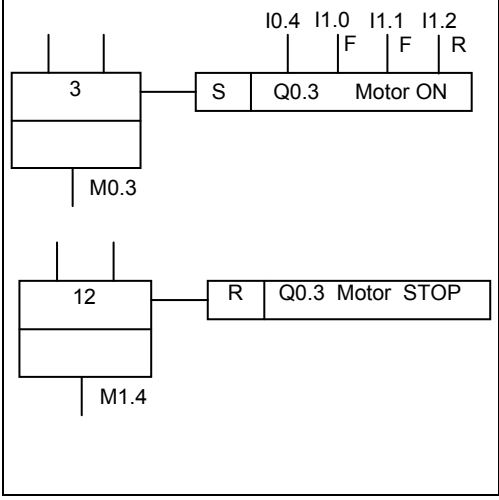
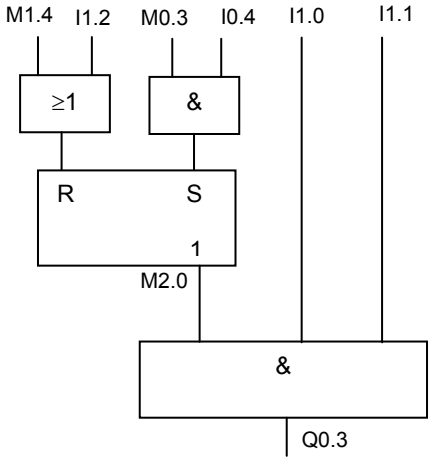
*** Lệnh T** (giới hạn thời gian)

Lệnh giới hạn thời gian bị xoá thông qua một bước. Nó đóng điện sau một thời gian xác định nếu bước còn tích cực. Nếu bước thoát khỏi trước thời gian định trước thì lệnh cũng mất tác dụng theo.

Ví dụ lệnh	*Biểu diễn sơ đồ logic*
	

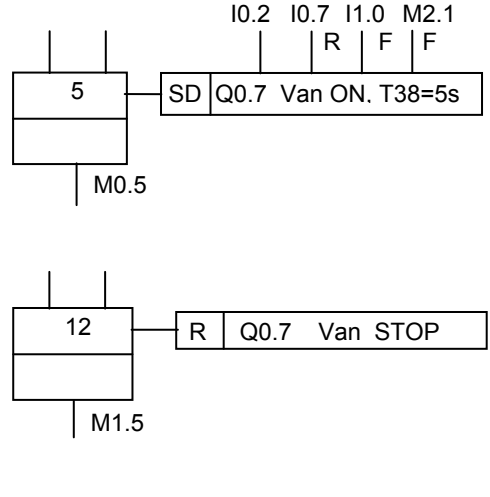
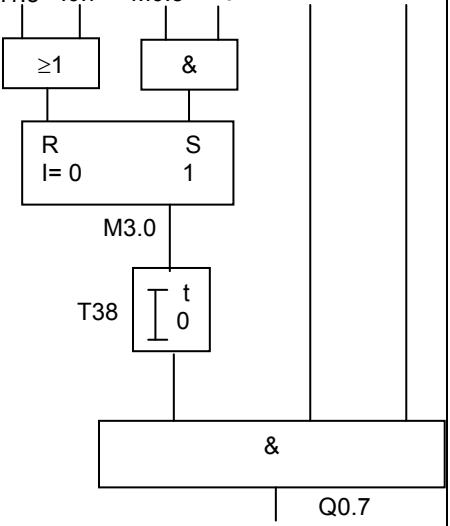
*** Lệnh S** (duy trì)

Lệnh duy trì được set trong một bước và giữ luôn sau đó nếu như bước không còn tác dụng nửa. Bởi vậy lệnh S phải được xóa bởi lệnh reset (R) ở một bước khác.

Ví dụ lệnh	Biểu diễn sơ đồ logic
	

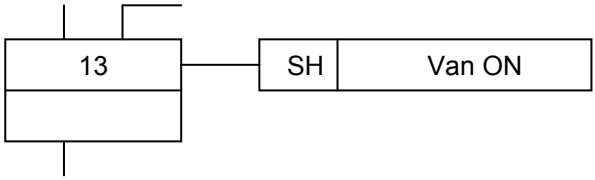
*** Lệnh SD** (trì hoãn thời gian và được duy trì)

Lệnh SD có tác dụng như lệnh S. Tuy nhiên ngõ ra có tác dụng sau quá trình thời gian trì hoãn "t" được điều chỉnh trước.

Ví dụ lệnh	Biểu diễn sơ đồ logic
	

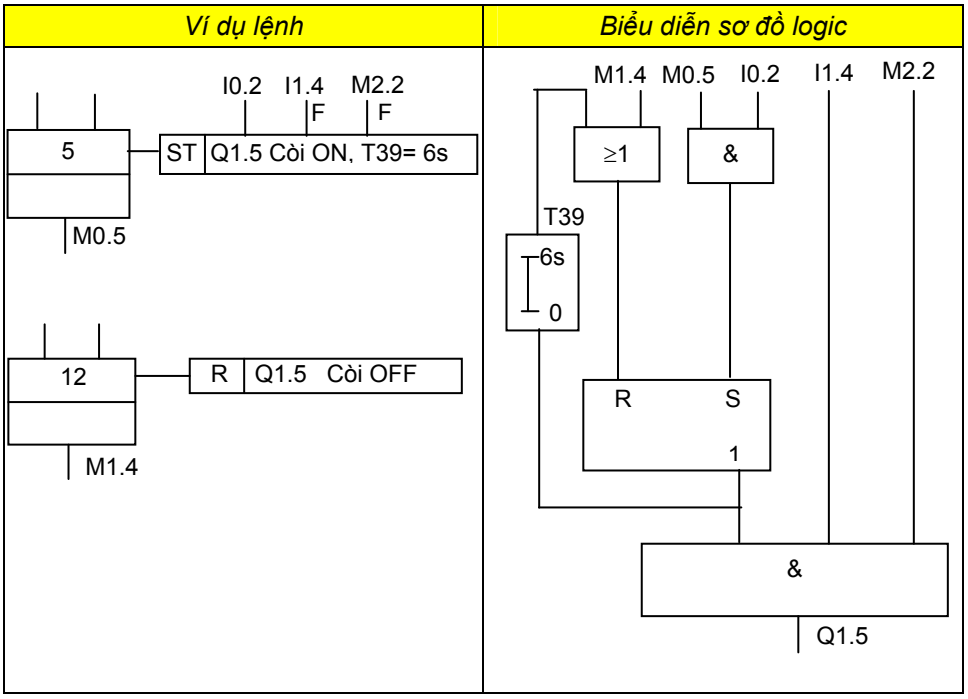
*** Lệnh SH** (duy trì trong trường hợp mất điện)

Lệnh SH có tác dụng như lệnh S nhưng sau đó lệnh được duy trì, nếu như vì một nguyên nhân nào đó điện áp cung cấp bị mất.



*** Lệnh ST** (duy trì và giới hạn thời gian)

Lệnh ST có tác dụng như lệnh S. Nó cũng còn được set nếu như bước phụ thuộc không còn được Set nữa và chỉ kéo dài trong một khoảng thời gian "t" được điều chỉnh trước.



11.8 Các chế độ hoạt động, cảnh báo và xuất lệnh

Tùy theo yêu cầu điều khiển mà người vận hành có thể đặt trạng thái hoạt động của thiết bị ở các trạng thái hoạt động khác nhau. Tùy theo chế độ làm việc được đặt mà chỉ cho tín hiệu ngõ ra ở các điều kiện xác định.

Một hệ thống điều khiển trình tự đầy đủ bên cạnh chuỗi trình tự còn bao gồm chế độ làm việc, cảnh báo và xuất lệnh.

Trong chương này chỉ trình bày chế độ hoạt động với các cảnh báo, hiển thị bước và xuất lệnh trong điều khiển trình tự, các chế độ hoạt động bao gồm:

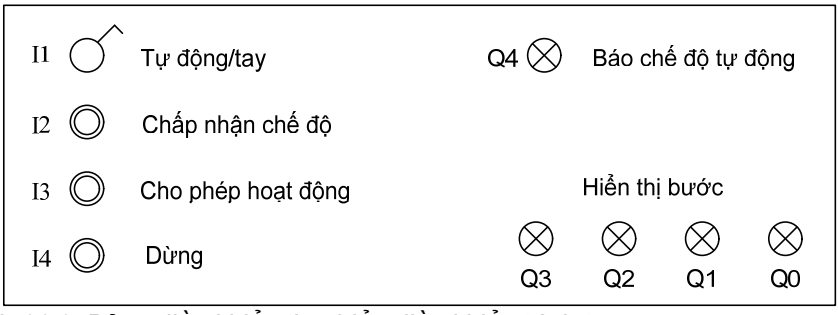
- Chế độ tự động

- Chế độ tay (chế độ bước đơn không có điều kiện)

11.8.1 Bảng điều khiển

Giao tiếp giữa người vận hành và hệ thống điều khiển là bảng điều khiển. Bảng điều khiển gồm có tất cả các công tắc chọn lựa chế độ, nút nhấn phục vụ theo yêu cầu của người điều khiển. Ngoài ra trên bảng điều khiển còn có các bộ chỉ thị để cảnh báo.

Bảng điều khiển được sử dụng trong chương này có dạng như sau:



Hình 11.6: Bảng điều khiển tiêu biểu điều khiển trình tự

Để tránh trùng các nút nhấn cũng như các đèn báo với các yêu cầu công nghệ đặt ra cho các bài toán điều khiển thì các nút nhấn và công tắc trên bảng điều khiển được ký hiệu là I1, I2, I3, I4 và các đèn báo là Q0 … Q4 với Q0..Q3 là bộ mã chỉ thị bước trình tự còn Q4 là báo chế độ tự động.

Nhiệm vụ của các nút nhấn, công tắc như sau:

Công tắc I1: **Tự động/tay**

Chọn chế độ hoạt động. Nếu I1 = "1" là chế độ tự động, I1 = "0" là chế độ tay.

Nút nhấn I2: **Chấp nhận chế độ**

Khi I1 = "1" (chế độ tự động) thì khi tác động I2 thì chuỗi trình tự được đặt về vị trí cơ bản (vị trí cơ bản) và ở lần tác động kế tiếp thì chế độ tự động được thực hiện. Nếu chuỗi trình tự đang sẵn sàng ở vị trí cơ bản thì chỉ cần tác động một lần I2 chế độ tự động được thực hiện.

Khi I1 = "0" (chế độ tay) mỗi lần tác động I2 sẽ đi đến bước kế tiếp trong chuỗi trình tự.

Nút nhấn I3: **Cho phép hoạt động**

Nút nhấn phải được tác động ở chế độ hoạt động theo bước đơn lẻ, để kích hoạt ngõ ra của mỗi bước.

Nút nhấn I4: **Dừng**

Kết thúc chế độ hoạt động tự động khi đến bước cuối cùng trong chuỗi trình tự.

11.8.2 Các khâu chế độ hoạt động có cảnh báo

Các chế độ hoạt động của điều khiển trình tự sẽ thực hiện xử lý tín hiệu từ bảng điều khiển và thiết bị cung cấp cho chuỗi trình tự các tín hiệu điều khiển được yêu cầu như:

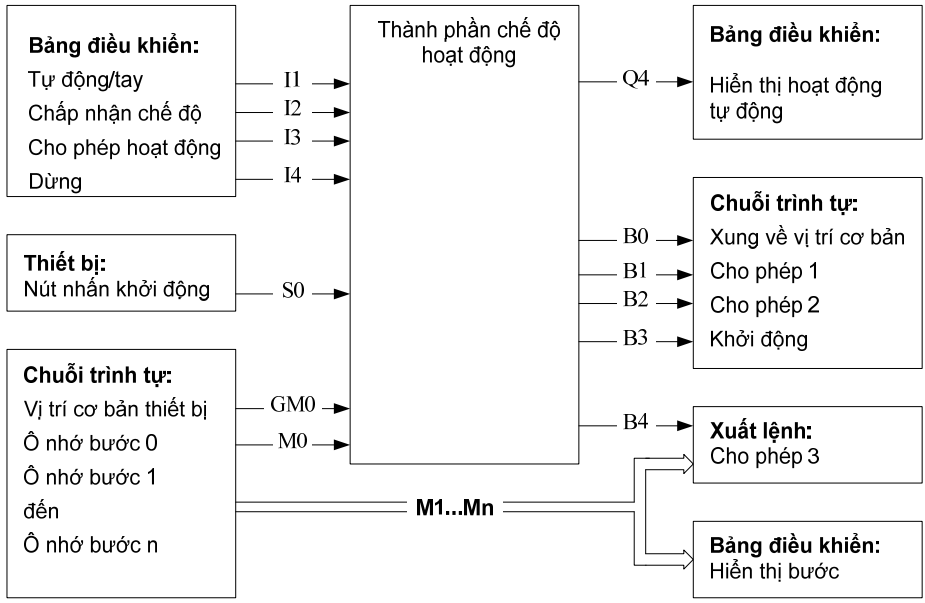
B0: Xung để trở về vị trí cơ bản của chuỗi trình tự

B1: Cho phép chuyển sang bước kế tiếp có điều kiện

B2: Cho phép chuyển sang bước kế tiếp không có điều kiện chuyển mạch

B3: Điều kiện khởi động chuỗi trình tự

Cấu trúc chương trình của các chế độ hoạt động với các tín hiệu vào và ra theo yêu cầu như sau:



Hình 11.7: Cấu trúc chương trình điều khiển trình tự theo các tín hiệu vào/ra

Ghi chú:

Tín hiệu cho phép 1 đối với chuyển mạch tiếp theo có điều kiện (tự động)

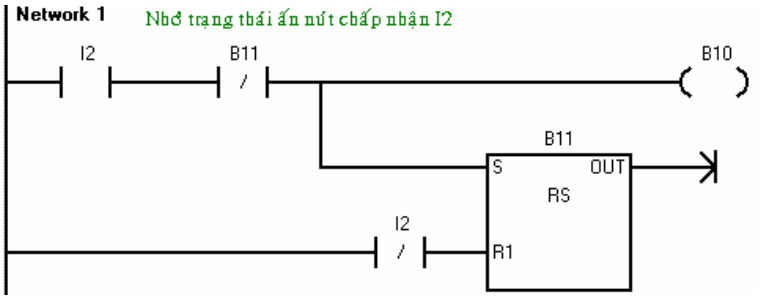
Tín hiệu cho phép 2 đối với chuyển mạch tiếp theo không điều kiện (tay)

Tín hiệu cho phép 3 đối với việc xuất lệnh

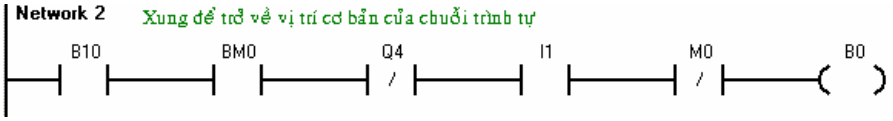
Dưới đây là các đoạn chương trình cho các khâu trong chế độ hoạt động với:

Các tín hiệu vào là các ngõ vào I1, I2, I3, I4, I0, GM0 và M0

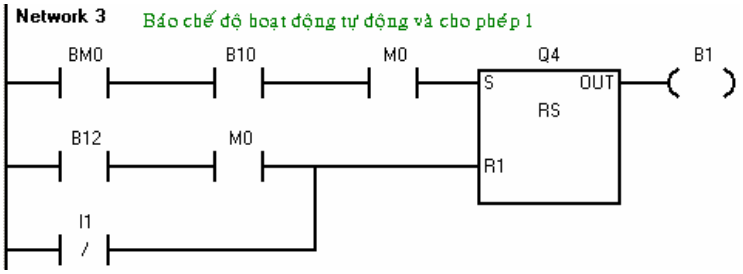
Các tín hiệu ra là Q4, B0, B1, B2, B3 và các ô nhớ phụ là B10, B11 và B12.



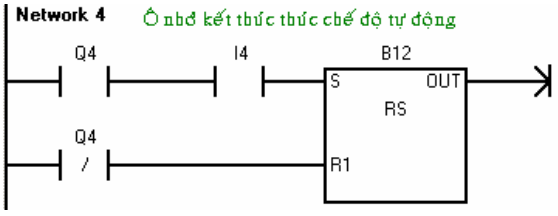
Tín hiệu B0:



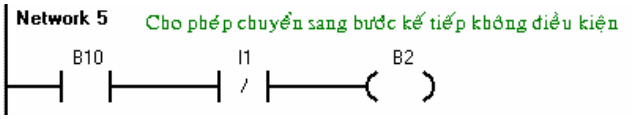
Tín hiệu Q4 và B1:



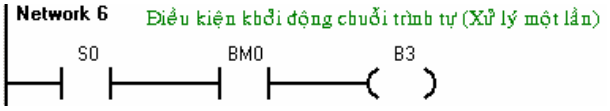
Tín hiệu B12:



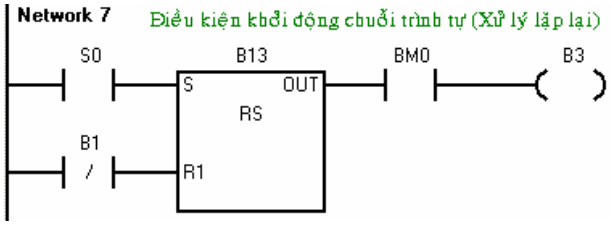
Tín hiệu B2: Cho phép chuyển mạch tiếp theo không điều kiện



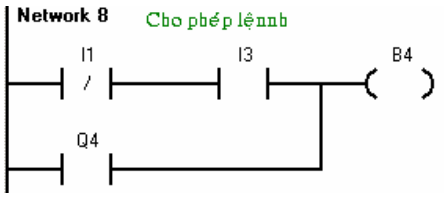
Điều kiện khởi động cho chuỗi trình tự (xử lý một lần)



Điều kiện khởi động cho chuỗi trình tự (xử lý lặp lại)



Cho phép lệnh:



Đoạn chương trình trên là chương trình tổng quát của các chế độ hoạt động với điều khiển trình tự. Tùy theo từng bài toán cụ thể mà ta sẽ gán cho các ngõ vào I1, I2, I3, I4, I0, Q4 các ngõ vào và ra tương ứng; GM0, M0, B0, B1, B2, B3, B10, B11 và B12 gán cho các ô nhớ M tương ứng.

11.8.3 Hiển thị bước trình tự

Tín hiệu hiển thị để cảnh báo trạng thái hoạt động của thiết bị được lập trình sẵn trong các khâu chế độ hoạt động.

Tín hiệu để cấp cho hiển thị bước là sự kết hợp của các ô nhớ của các bước.

11.8.4 Xuất lệnh

Trong phần xuất lệnh của điều khiển trình tự thì lệnh xuất được liên kết từ tín hiệu cho phép lệnh với ô nhớ bước trình tự.

11.9 Các ví dụ ứng dụng

Trong các ví dụ sẽ không trình bày phần kết nối dây với PLC nữa. Phần này yêu cầu bạn đọc tự thực hiện.

11.9.1 Máy phay đơn giản

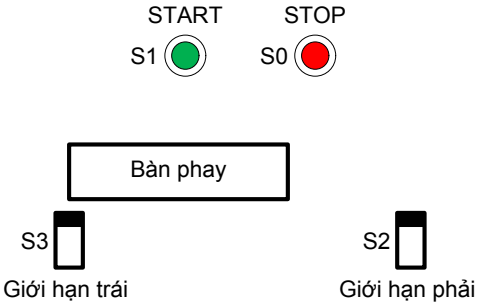
Mô tả hoạt động:

Khi ấn nút khởi động S1 thì bàn máy di chuyển về hướng phải. Khi bàn máy gặp công tắc hành trình S2 thì tự động quay ngược trở lại. Trong chiều chạy ngược, nếu bàn phay đụng công tắc hành trình S3 thì tự động đảo chiều. Quá trình cứ thế lặp đi lặp lại.

Khi ấn nút dừng S0 thì bàn phay tiếp tục quay cho hết chu kỳ và chỉ dừng lại khi trở về vị trí cơ bản (giới hạn trái).

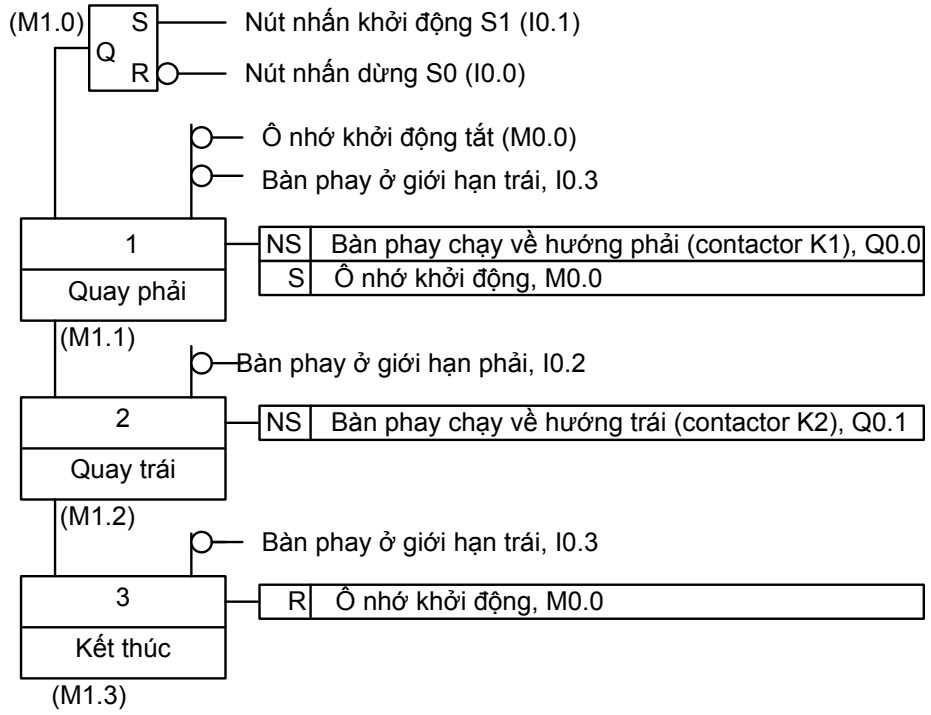
Thực hiện viết chương trình điều khiển máy phay này theo phương pháp trình tự.

Sơ đồ công nghệ:



Hình 11.8: Sơ đồ công nghệ máy phay đơn giản

Sơ đồ điều khiển theo trình tự:



Hình 11.9: Sơ đồ điều khiển theo trình tự máy phay đơn giản

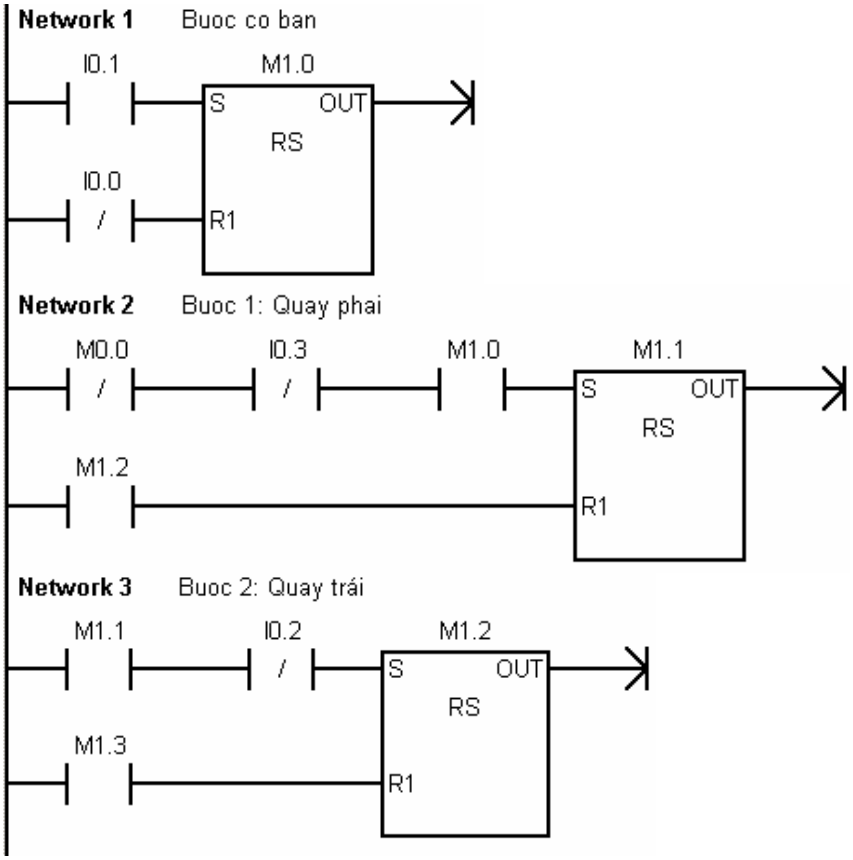
Bảng ký hiệu:

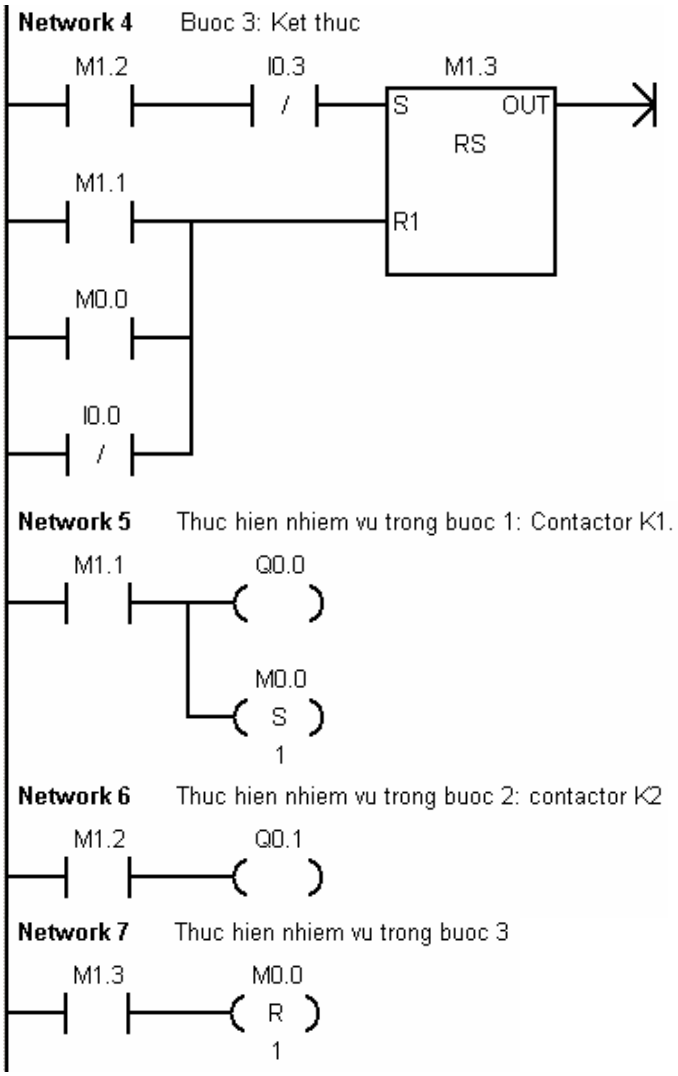
Ký hiệu	***Địa chỉ***	***Chú thích***
Các biến vào		
S0	I0.0	Nút nhấn dừng, NC

S1	I0.1	Nút nhấn khởi động
S2	I0.2	Công tắc hành trình báo giới hạn phải, NC
S3	I0.3	Công tắc hành trình báo giới hạn trái,NC
Các biến ra		
K1	Q0.0	Contactor điều khiển bàn phay chạy về hướng phải
K2	Q0.1	Contactor điều khiển bàn phay chạy về hướng trái

Chương trình

Biểu diễn ở LAD:





Biểu diễn ở STL:

Network 1 Buoc co ban
```
LD    I0.1
LDN   I0.0
NOT
LPS
A     M1.0
=     M1.0
LPP
ALD
O     M1.0
=     M1.0
```

Network 2 Buoc 1: Quay phai
```
LDN   M0.0
AN    I0.3
A     M1.0
LD    M1.2
NOT
LPS
A     M1.1
=     M1.1
LPP
ALD
O     M1.1
=     M1.1
```

Network 3 Buoc 2: Quay trái

```
LD    M1.1
AN    I0.2
LD    M1.3
NOT
LPS
A     M1.2
=     M1.2
LPP
ALD
O     M1.2
=     M1.2
```

Network 4 Buoc 3: Ket thuc

```
LD    M1.2
AN    I0.3
LD    M1.1
O     M0.0
ON    I0.0
NOT
LPS
A     M1.3
=     M1.3
LPP
ALD
O     M1.3
=     M1.3
```

Network 5 Thuc hien nhiem vu trong buoc 1: Contactor K1

```
LD    M1.1
=     Q0.0
S     M0.0, 1
```

Network 6 Thuc hien nhiem vu trong buoc 2: contactor K2

```
LD    M1.2
=     Q0.1
```

Network 7 Thuc hien nhiem vu trong buoc 3

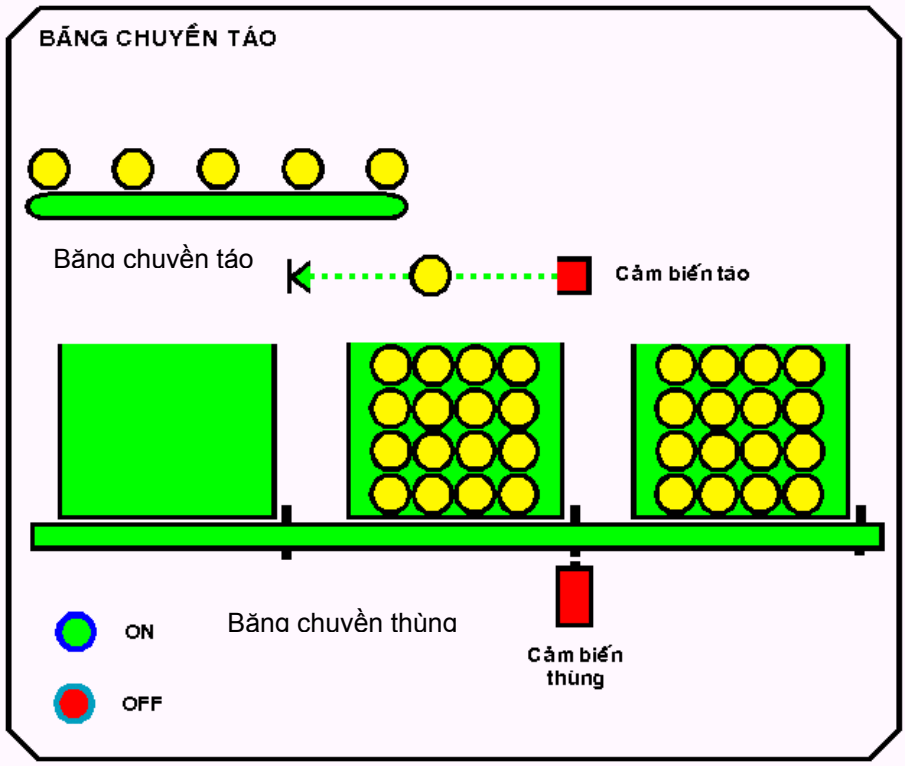
```
LD    M1.3
R     M0.0, 1
```

11.9.2 Băng chuyền đếm táo

Mô tả hoạt động:

Khi ấn nút khởi động ON thì băng chuyền thùng hoạt động. Khi thùng đến vị trí thì dừng lại và băng chuyền táo hoạt động. Nếu số lượng táo đếm được bằng 12 thì băng chuyền táo dừng. Băng chuyền chạy tiếp cho đến khi một thùng thứ hai đúng vị trí thì dừng lại. Quá trình được lặp đi lặp lại cho đến khi nào ấn nút OFF.

Sơ đồ công nghệ:

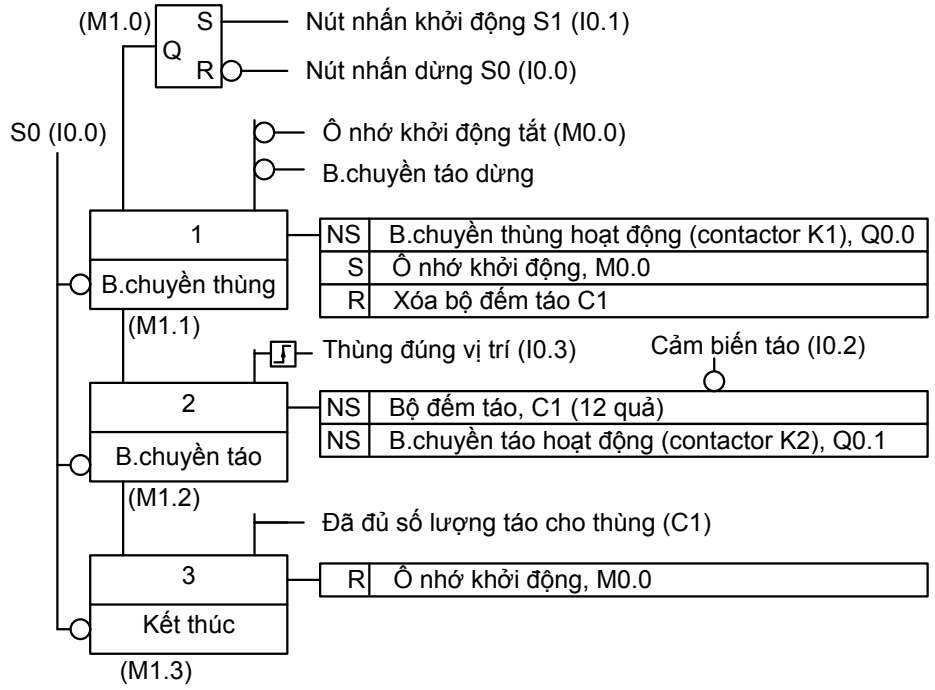


Hình 11.10: Sơ đồ công nghệ băng chuyền đếm táo

Bảng ký hiệu:

Ký hiệu	***Địa chỉ***	***Chú thích***
Các biến vào		
OFF	I0.0	Nút nhấn dừng, NC
ON	I0.1	Nút nhấn khởi động hệ thống
CB_tao	I0.2	Cảm biến táo, NC
CB_thung	I0.3	Cảm biến thùng đúng vị trí,NO
Các biến ra		
K1	Q0.0	Contactor điều khiển băng chuyền táo
K2	Q0.1	Contactor điều khiển băng chuyền thùng

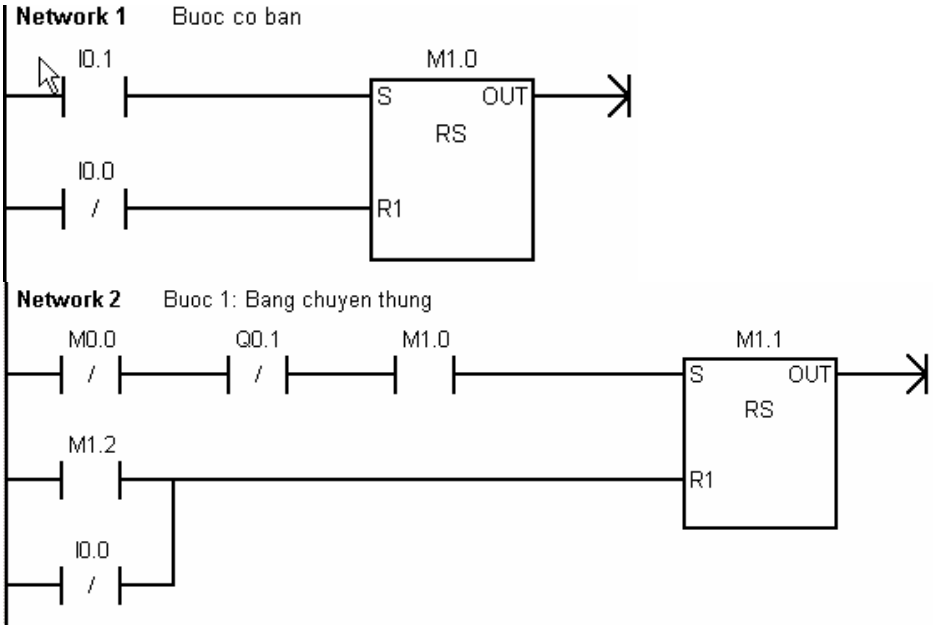
Sơ đồ điều khiển theo trình tự:

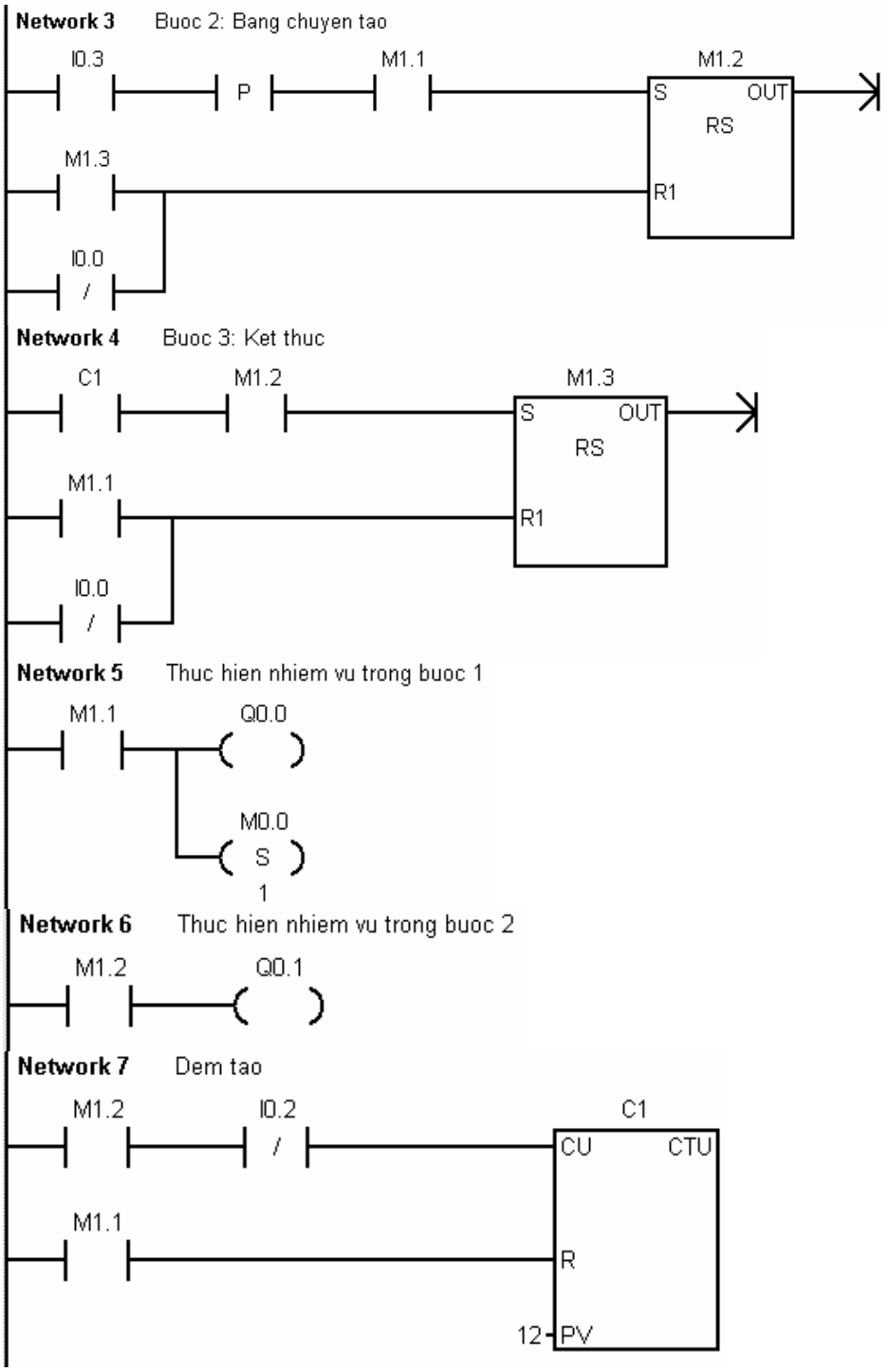


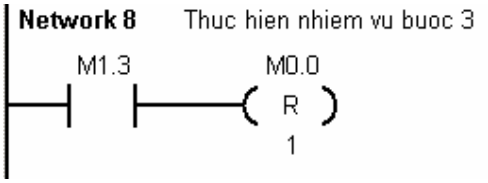
Hình 11.11: Sơ đồ điều khiển theo trình tự băng chuyền đếm táo

Chương trình

Biểu diễn ở LAD:




Network 3 Buoc 2: Bang chuyen tao
I0.3
P
M1.1
M1.2
S
OUT
RS
M1.3
R1
I0.0
/
Network 4 Buoc 3: Ket thuc
C1
M1.2
M1.3
S
OUT
RS
M1.1
R1
I0.0
/
Network 5 Thuc hien nhiem vu trong buoc 1
M1.1
Q0.0
M0.0
S
1
Network 6 Thuc hien nhiem vu trong buoc 2
M1.2
Q0.1
Network 7 Dem tao
M1.2
I0.2
/
C1
CU
CTU
M1.1
R
12
PV



Chương trình biểu diễn ở STL:

Network 1 Buoc co ban
```
LD    I0.1
LDN   I0.0
NOT
LPS
A     M1.0
=     M1.0
LPP
ALD
O     M1.0
=     M1.0
```

Network 2 Buoc 1: Bang chuyen thung
```
LDN   M0.0
AN    Q0.1
A     M1.0
LD    M1.2
ON    I0.0
NOT
LPS
A     M1.1
=     M1.1
LPP
ALD
O     M1.1
=     M1.1
```

Network 3 Buoc 2: Bang chuyen tao
```
LD    I0.3
EU
A     M1.1
LD    M1.3
ON    I0.0
NOT
LPS
A     M1.2
=     M1.2
LPP
ALD
O     M1.2
=     M1.2
```

Network 4 Buoc 3: Ket thuc
```
LD    C1
A     M1.2
LD    M1.1
ON    I0.0
NOT
LPS
A     M1.3
=     M1.3
LPP
ALD
O     M1.3
=     M1.3
```

Network 5 Thuc hien nhiem vu trong buoc 1
```
LD    M1.1
=     Q0.0
S     M0.0, 1
```

Network 6 Thuc hien nhiem vu trong buoc 2
```
LD    M1.2
=     Q0.1
```

Network 7 Dem tao
```
LD    M1.2
AN    I0.2
LD    M1.1
CTU   C1, 12
```

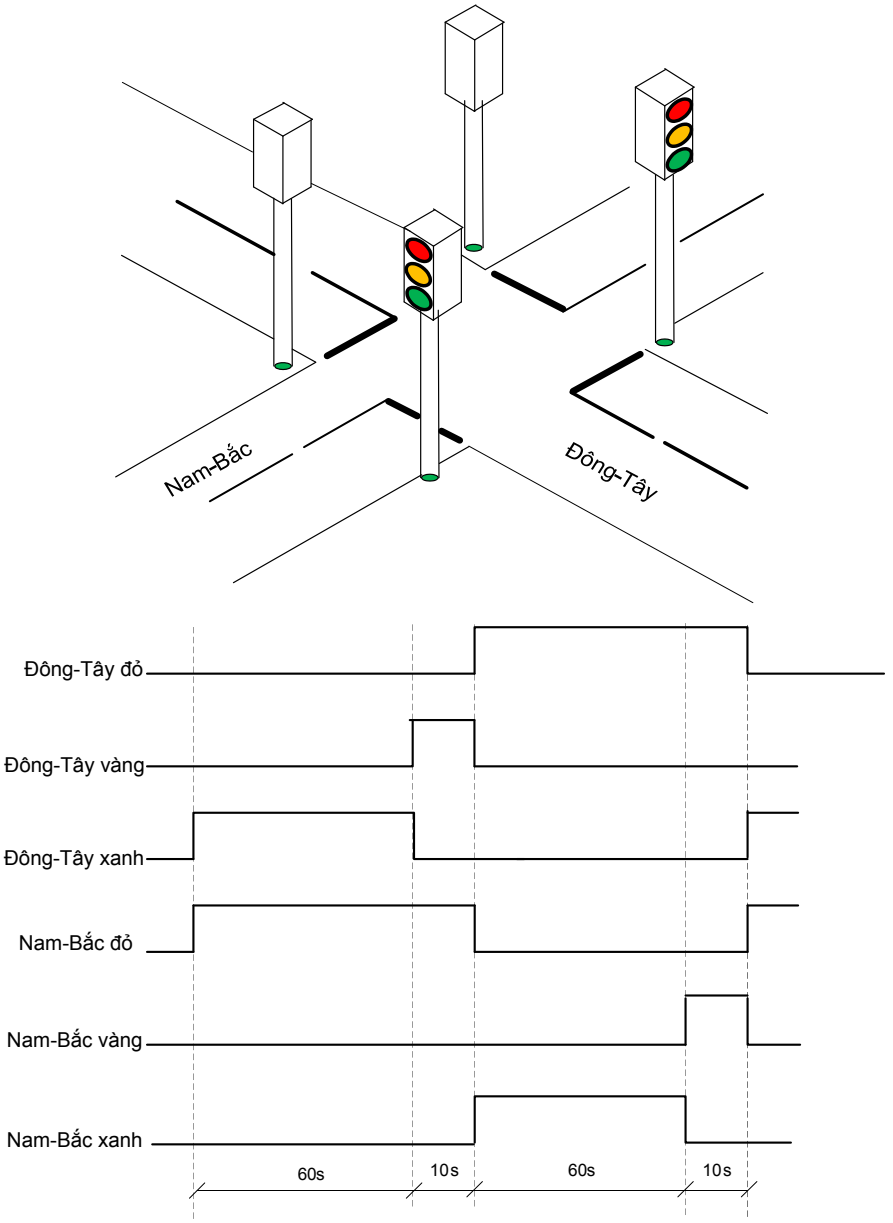
Network 8 Thuc hien nhiem vu buoc 3
```
LD    M1.3
R     M0.0, 1
```

11.10 Câu hỏi và bài tập

BT 11.1 Đèn giao thông

Một giao lộ hình ảnh và có chế độ làm việc như hình 11.12

Sơ đồ công nghệ và giản đồ thời gian



Hình 11.12: Sơ đồ công nghệ đèn giao thông và giản đồ thời gian

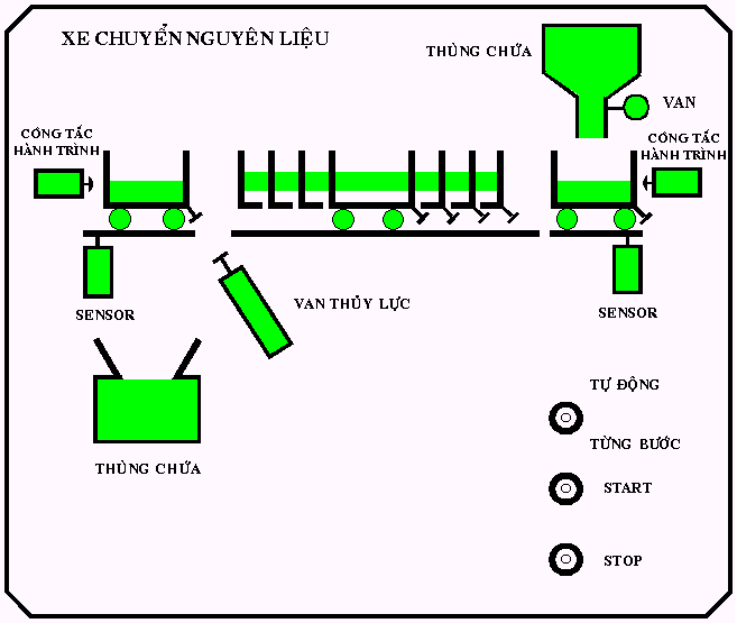
Bảng ký hiệu

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Công tắc hệ thống
H1	Q0.0	Đông-Tây đỏ
H2	Q0.1	Đông-Tây vàng
H3	Q0.2	Đông-Tây xanh
H4	Q0.3	Nam-Bắc đỏ
H5	Q0.4	Nam-Bắc vàng
H6	Q0.5	Nam-Bắc xanh

Khi bật công tắc S1 về vị trí "ON" thì hệ thống đèn giao thông hoạt động theo sơ đồ thời gian trên. Ở vị trí "OFF" thì toàn bộ hệ thống đèn tắt.

Hãy viết chương trình điều khiển theo phương pháp trình tự.

BT 11.2 Xe chuyển nguyên liệu



Hình 11.13: Sơ đồ công nghệ xe chuyển nguyên liệu

Bảng ký hiệu

Ký hiệu	***Địa chỉ***	***Chú thích***
Start	I0.0	Khởi động hệ thống, thường hở.
End 1	I0.1	Công tắc hành trình ở trạm xả, thường đóng
Fill 1	I0.2	Cảm biến báo xe rỗng, thường đóng.

End 2	I0.3	Công tắc hành trình trạm nạp, thường đóng.
Fill 2	I0.4	Cảm biến báo đầy, thường hở.
Stop	I0.5	Dừng, thường đóng.
Step	I0.6	Chế độ bước, thường hở.
Auto	I0.7	Chế độ tự động, thường hở.
Dir_A	Q0.0	Xe chạy về hướng A
Dir_B	Q0.1	Xe chạy về hướng B
Y1	Q0.2	Van xả nguyên liệu
Y2	Q0.3	Van thủy lực

Mô tả hoạt động

Xe vận chuyển nguyên liệu hoạt động như sau:

* Xe vận chuyển nguyên liệu có thể thực hiện qua công tắc chọn chế độ:
  - Chế độ tự động: I0.6
  - Chế độ bước: I0.7

* Vị trí cơ bản: Xe ở vị trí công tắc hành trình End 2 (I0.3 và xe chưa được làm đầy.

Chế độ tự động:

Khi xe ở vị trí cơ bản và công tắc chọn chế độ đặt ở chế độ tự động, khi nhấn nút khởi động (I0.0) thì van xả Y1 mở, vật liệu được đổ vào xe, cảm biến Fill 2 dùng để nhận biết xe đã được đổ đầy. Khi xe đầy thì van xả Y1 mất điện và xe chạy về hướng B sau thời gian ổn định 5s, xe dừng lại tại B (trạm nhận nguyên liệu) khi chạm công tắc hành trình S2. Xy lanh thủy lực của thiết bị xả được điều khiển và tấm chắn trên xe được mở vật liệu được rót vào bồn chứa. Khi xe xả hết vật liệu cảm biến S4 phát ra tín hiệu 1, pit tông thủy lực của thiết bị xả mất điện, tấm chắn trở về vị trí cũ, xe dừng 5 giây sau đó chạy về hướng A. Chu kỳ hoạt động được lặp lại.

Nếu trong chu kỳ hoạt động mà nút "dừng" được ấn thì quá trình vẫn tiếp tục cho đến khi xe trở về vị trí cơ bản (xe rỗng và ở trạm nhận nguyên liệu) và dừng hẳn.

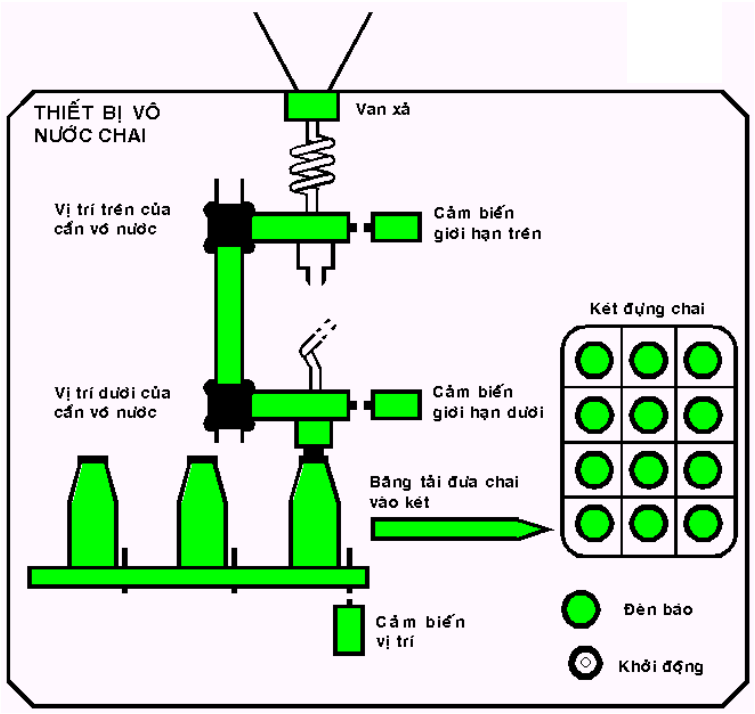
Chế độ bước:

Ở mỗi bước thực hiện phải thông qua nút nhấn "start".

Ví dụ : khi ấn "start" xe đúng vị trí van xả được mở, khi xe đầy thì S3 tác động, van xả đóng lại. Nếu tiếp tục ấn "start" thì xe chạy về hướng B.

Hãy viết chương trình điều khiển xe chuyển nguyên liệu này theo điều khiển trình tự.

BT 11.3 Thiết bị vô nước chai



Hình 11.14: Sơ đồ công nghệ thiết bị vô nước chai

Bảng ký hiệu

Ký hiệu	*Địa chỉ*	*Chú thích*
S1	I0.0	Giới hạn trên của cần vô nước, thường đóng
S2	I0.1	Giới hạn dưới của cần vô nước, thường đóng
S3	I0.2	Cảm biến vị trí chai, thường hở
S4	I0.3	Khởi động hệ thống, thường hở
S5	I0.4	Chai đúng vị trí trong két, thường hở
K1	Q0.0	Van xả nước
K2	Q0.1	Hạ cần vô nước xuống
K3	Q0.2	Nâng cần vô nước lên
K4	Q0.3	Băng tải vận chuyển chai rỗng
K5	Q0.4	Đèn báo két đầy

Mô tả

Thiết bị vô nước chai hoạt động như sau:

Trước khi vận hành thiết bị vô nước chai thì các chai rỗng phải được đặt lên băng tải. Nếu sau đó nút nhấn khởi động (I0.3) được tác động, thì băng tải sẽ vận chuyển chai rỗng với thời gian trì hoãn ban đầu là 1s. Băng tải dừng lại khi có một chai đến cảm biến vị trí (I0.2).

Bây giờ cần vô nước sẽ hạ từ trên xuống, khi đến giới hạn dưới (I0.1) thì dừng lại, sau đó 1s thì van xả sẽ được mở đổ nước vào chai, van xả sẽ đóng lại khi chai đầy thời gian làm đầy kéo dài khoảng 3s.

Sau khi van xả đóng lại 1s thì cần vô nước được nâng lên, đến giới hạn trên (I0.0) thì dừng lại. Sau đó 1s thì băng tải vận chuyển chai rỗng lại tiếp tục và quá trình cứ thế lặp lại.

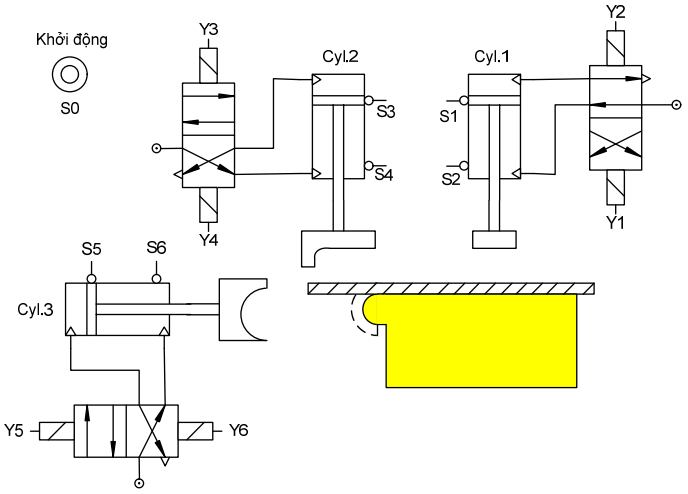
Chai đã đổ đầy nước được đưa sang băng tải đưa chai vào két khi băng tải chai rỗng hoạt động, khi chai đúng vị trí trong két thì có một tín hiệu phát ra (I0.4).

Quá trình được lặp đi lăp lại cho đến khi nào số lượng chai trong két đủ 12 thì đèn báo sáng lên và hệ thống dừng lại. Quá trình mới lại bắt đầu khi nút nhấn khởi động được tác động.

Hãy viết chương trình điều khiển sử dụng phương pháp trình tự.

BT 11.4 Máy uốn thanh kim loại

Sơ đồ công nghệ:



Hình 11.15: Sơ đồ công nghệ máy uốn thanh kim loại

Các thanh kim loại cần được uốn một đầu theo theo một khuôn cho trước (sơ đồ công nghệ). Qui trình hoạt động của máy như sau:

- Thanh kim loại cần uốn được đặt lên khuôn uốn
- Ấn nút khởi động S0 thì xy lanh Cyl.1 hạ xuống để giữ lấy thanh kim loại.

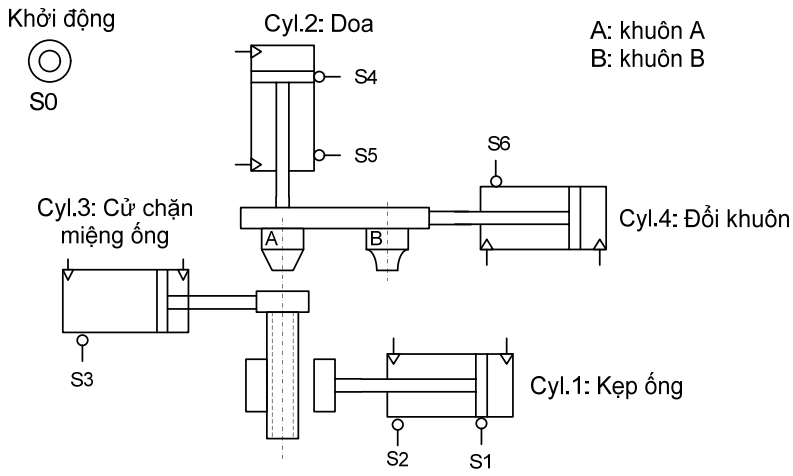
- Khi thanh kim loại được giữ chặt (nhận biết bởi công tắc hành trình S2) thì xy lanh Cyl.2 hạ xuống để uốn thanh kim loại vuông góc trước. Sau khi uốn xong thì tự động nâng lên nhờ công tắc hành trình S4.
- Khi xy lanh Cyl.2 trở về vị trí cơ bản (nhận biết bởi S3) thì xy lanh Cyl.3 được đẩy để uốn thanh kim loại ở giai đoạn uốn cuối theo định hình của khuôn uốn. Khi xy lanh Cyl.3 đến vị trí S6 thì tự động rút ngược về.
- Khi xy lanh Cyl.3 rút về đến vị trí cơ bản (nhận biết bởi S5) thì xy lanh Cyl.1 cũng rút về vị trí cơ bản của nó (nhận biết bởi S1). Lúc này thanh kim loại được tự do. Người sử dụng có thể lấy ra và đặt một thanh kim loại mới vào. Và một chu kỳ mới lại có thể bắt đầu.

Hãy viết chương trình điều khiển sử dụng phương pháp trình tự.

BT 11.5 Máy doa miệng ống kim loại

Ống kim loại cần được doa miệng theo một khuôn cho trước (sơ đồ công nghệ).

Sơ đồ công nghệ:



Hình 11.16: Sơ đồ công nghệ máy doa miệg ống kim loại.

Máy hoạt động như sau:

Người vận hành đặt ống kim loại cần doa miệng vào vị trí sao cho miệng ống phải chạm vào cử chặn miệng ống. Sau đó ấn nút nhấn S0, xy lanh Cyl.1 sẽ kẹp ống lại. khi ống đã được kẹp thì cử chặn miệng ống tự động rút về. Xy lanh Cyl.2 sẽ hạ xuống doa miệng ống theo khuôn A. thời gian doa khỏang 3s. Sau đó xy lanh Cyl.2 rút về và khuôn B được xylanh Cyl.4 đưa vào. Sau khi khuôn B được đưa vào thì xy lanh Cyl.2 hạ xuống để doa miệng ống theo khuôn B. Tương tự như khuôn A việc doa khoảng 3s. Sau đó xy lanh Cyl.2 trở về vị trí cơ bản của nó và xy lanh Cyl.4 cũng rút khuôn B về và đặt

khuôn A về vị trí sẵn sàng cho ống kim loại kế tiếp. Sau khi miệng ống đã được doa theo khuôn B xong thì xy lanh kẹp ống Cyl.1 co về thả ống kim loại khỏi hàm kẹp. Xy lanh Cyl.2 được đẩy trở về vị trí chặn miệng ống. Một chu kỳ mới lại có thể bắt đầu.

Hãy viết chương trình điều khiển sử dụng phương pháp trình tự.

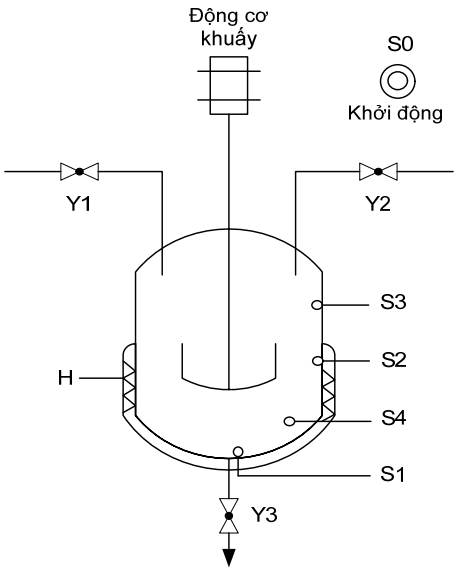
BT 11.6 Bồn trộn

Hai loại chất lỏng khác nhau được trộn và được nung nóng đến một nhiệt độ xác định theo sơ đồ công nghệ như hình vẽ.

Mô tả hoạt động:

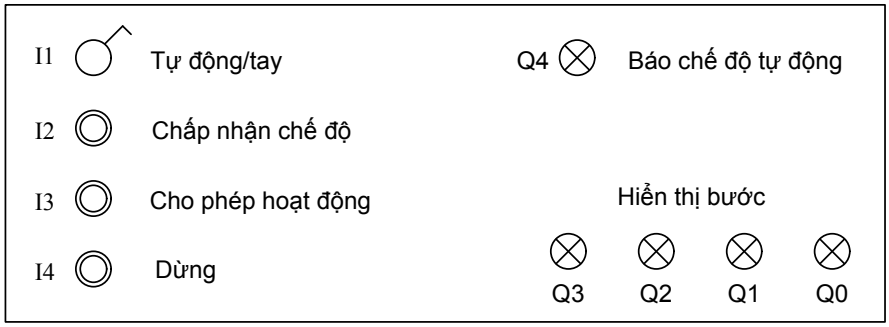
Sau khi nút nhấn S0 được tác động thì van Y1 mở cho chất lỏng A vào bồn đến công tắc giới hạn mức S2 thì đóng lại. Sau đó động cơ khuấy được cấp điện và van Y2 được mở. Khi công tắc giới hạn mức S3 tác động thì van Y2 đóng lại và điện trở nung H được cấp điện. Cảm biến nhiệt S4 thông báo nhiệt đã đạt đến nhiệt độ cho trước thì điện trở nung và động cơ khuấy mất điện và van Y3 được mở. Khi công tắc báo mức S1 thông báo rằng bồn đã xả hết thì van Y3 đóng lại và một quá trình mới được lặp lại nếu nút nhấn S0 được tác động.

Sơ đồ công nghệ:



Hình 11.17: Bồn trộn

Bảng điều khiển:



Bảng ký hiệu:

Ký hiệu	Địa chỉ	Chú thích
Các biến vào		
I1	I1.1	Công tắc tay/tự động
I2	I1.2	Chấp nhận chế độ
I3	I1.3	Cho phép hoạt động
I4	I1.4	Dừng
S0	I0.0	Nút nhấn khởi động
S1	I0.1	Công tắc hành trình báo mực chất lỏng 1 (bồn rỗng)
S2	I0.2	Công tắc hành trình báo mực chất lỏng 2
S3	I0.3	Công tắc hành trình báo mực chất lỏng 3
S4	I0.4	Cảm biến nhiệt độ
Các biến ra		
Q0	Q0.6	Chỉ thị bướ giá trị 1
Q1	Q0.7	Chỉ thị bước giá trị 2
Q2	Q1.0	Chỉ thị bước giá trị 4
Q4	Q1.1	Chỉ thị chế độ tự động
Y1	Q0.0	Van Y1, van mở Q0.0=”1”
Y2	Q0.1	Van Y2, van mở Q0.1=”1”
Y3	Q0.2	Van Y3, van mở Q0.2=”1”
H	Q0.3	Điện trở nung
M	Q0.4	Động cơ khuấy

Hãy viết chương trình điều khiển sử dụng phương pháp trình tự.

12 An toàn trong PLC

12.1 Khái niệm và mục đích

An toàn của một thiết bị điện không chỉ chú ý đối với PLC mà còn chú ý đến tổng thể các hoạt động bên ngoài máy móc và thiết bị. Sự an toàn của một trang bị điện phải được thực hiện không phụ thuộc vào loại điều khiển, ví dụ điều khiển bằng contactor hay PLC.

Khái niệm an toàn được hiểu theo nghĩa khả năng của một hệ thống có tác dụng trong một giới hạn cho trước trong một khoảng thời gian xác định mà không có nguy hiểm xảy ra. An toàn chỉ có thể đạt được trong khoảng giới hạn cho trước. Các giới hạn này thuộc về các điều kiện môi trường như:

- Nhiệt độ
- Độ ẩm
- Sự tác động cơ khí
- Bảo dưỡng đúng
- Sử dụng đúng
- Thời gian hoạt động

Mục đích của an toàn là:

- Không gây nguy hiểm đến tính mạng và sức khỏe con người
- Bảo đảm cho máy móc, thiết bị trước các sự cố đáng tiếc
- Bình thường trong các trường hợp lỗi

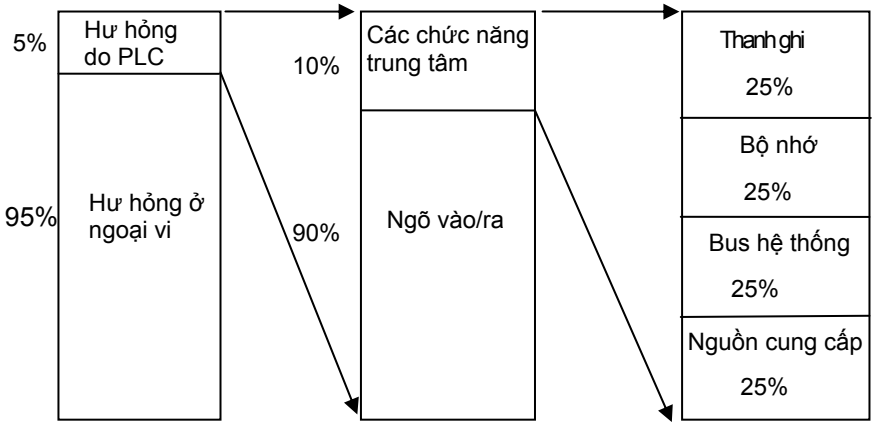
12.2 Hư hỏng ở PLC

Trong thực tế chỉ ra rằng 95% tất cả các hư hỏng là do thiết bị ngoại vi. Các hư hỏng có thể là:

- Đứt dây dẫn đến thiết bị hay khâu điều chỉnh
- Các hư hỏng ở cơ cấu chấp hành như nút nhấn, công tắc, công tắc hành trình.
- Hư hỏng ở khâu điều chỉnh.

Còn đối với hư hỏng do PLC gây ra thì vào khoảng 5%. Hầu hết là do các khối vào/ra, bộ xử lý trung tâm hay nguồn cung cấp.

Hư hỏng ở các thiết bị điều khiển được phân bố như sau:



Từ sơ đồ trên, ta có thể phán đoán được các lỗi xuất hiện ở đâu để tìm lỗi ở thiết bị ngoại vi hay ở PLC.

- Các lỗi ngoại vi có thể nhận biết, nếu:
  - Tất cả các ngõ vào/ra của PLC có LED hiển thị
  - Với sự giúp đỡ của thiết bị lập trình (đặt ở chế độ Online)
  - Nếu các thông báo lỗi có thể được thực hiện với phần mềm
- Các lỗi ở PLC có thể được nhận biết nếu các trạng thái bên trong hệ thống được chỉ thị với các LED báo trạng thái, ví dụ như:
  - Giám sát chương trình điều khiển, điều khiển chu kỳ
  - Kiểm tra nguồn cung cấp
  - Giám sát nhiệt độ
  - . . . .

Bên cạnh đó các lỗi cũng có thể được in ra ở dạng văn bản để dễ tìm lỗi.

12.3 Các quan điểm về kỹ thuật an toàn ở PLC

12.3.1 Các lỗi nguy hiểm và không nguy hiểm

Các lỗi có thể xuất hiện trong điều khiển ở một vị trí bất kỳ. Khi một lỗi xuất hiện, nó có thể là lỗi nguy hiểm hay không nguy hiểm tùy thuộc vào ảnh hưởng nào mà nó gây ra đối với trạng thái tín hiệu thực hiện

- *Các lỗi nguy hiểm được xem là nguy hiểm, nếu:*
  - Gây hại đến sự an toàn cho con người và máy móc, thiết bị
  - Các lỗi này cần phải được ngăn ngừa
  - Tác dụng của nó phải được ngăn ngừa đối với hoạt động an toàn của thiết bị.
- *Các lỗi không nguy hiểm, nếu:*

- Không tác hại đến sự an toàn
- Nó có thể được xử lý, ví dụ với các ngắt báo lỗi
- Cắt truyền động.

Các lỗi nguy hiểm và không nguy hiểm có thể xuất hiện là lỗi tích cực (tín hiệu "1" ở ngõ ra, đáng lẽ ra nó phải là "0") hoặc lỗi không tích cực (tín hiệu "0" ở ngõ ra, đáng lẽ ra nó phải là "1").

12.3.2 Các cách giải quyết cho hoạt động an toàn của thiết bị điều khiển PLC

Không có một giải pháp kỹ thuật an toàn nào có giá trị chung cho tất cả các vấn đề điều khiển, vì mỗi sự điều khiển có đặc điểm riêng, điều kiện công nghệ, trình tự hoạt động, qui luật và điều kiện môi trường. Từ đó, đối với mỗi thiết bị phải được quyết định lấy phương pháp kỹ thuật an toàn nào để tránh được các sự cố đáng tiếc cho người và máy móc. Hiện tại vẫn chưa có giải đáp thõa mãn về phần cứng và phần mềm cho vấn đề an toàn.

Các nhà chế tạo PLC đã đưa vào các chức năng an toàn của thiết bị điều khiển PLC. Chúng giúp cho người dùng tránh được tình trạng đứng máy của thiết bị tự động để thực hiện có chất lượng và hiệu quả cao.

Có thể tóm tắt các cách giải quyết cho hoạt động an toàn như sau:

- Cấu trúc PLC an toàn
- Thiết bị giám sát bên trong hệ thống của PLC (giám sát hoạt động chương trình (watch-dog), phương pháp đánh dấu kiểm tra).
- Thiết kế đúng (sự đóng mạch lại, dừng khẩn cấp, thời gian giám sát, dự phòng …)
- Lập trình an toàn khi đứt dây
- Các mạch an toàn cao
- Lắp mạch bảo vệ các ngõ ra

- *Các mạch an toàn cao*

Các mạch an toàn cao là các thiết bị điều khiển phụ được thực hiện ở ngõ ra của PLC cho chức năng an toàn. Các thiết bị điều khiển này đảm nhận chức năng an toàn riêng cho thiết bị điều khiển

- *Các "khóa"*

  Các khóa cần thiết để tránh các trạng thái đóng mạch không mong muốn. Có các loại "khóa" cứng khác nhau sau:

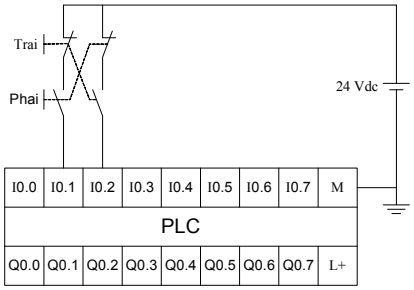
  **** Khóa 2 ngõ vào** (hình 12.10)*

  Trường hợp này chỉ sử dụng đối với các mạch điều khiển động cơ quay phải, trái dùng contactor. Còn trong PLC không bắt buộc.

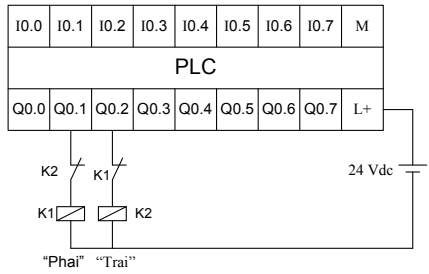
**** Khóa ngõ ra** (hình 12.11)*

Ở đây các ngõ ra được khóa chéo lẫn nhau sử dụng tiếp điểm thường đóng. Điều này tránh cho các contactor điều khiển động cơ quay phải và quay trái đóng cùng lúc.

Loại khoá này ở PLC là loại khóa được chỉ định bắt buộc, vì hiện tượng dính tiếp điểm của contactor và lỗi lập trình gây ra.

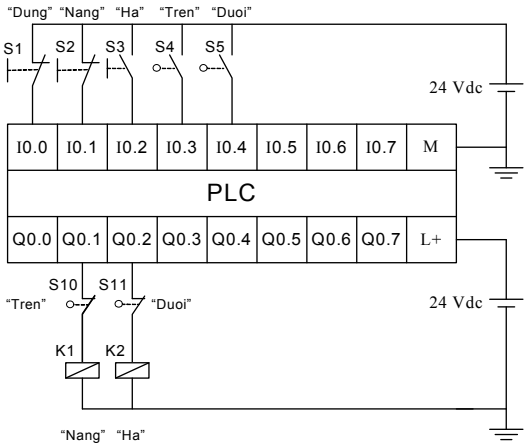


Hình 12.14: Khóa 2 ngõ vào



Hình 12.15: Khóa 2 ngõ ra

**** Khóa do nhấn 2 tay cùng lúc***



Trong khóa này cần phải lập trình sao cho việc tác động nút nhấn trong một thời gian xác định (ví dụ 0,2s).

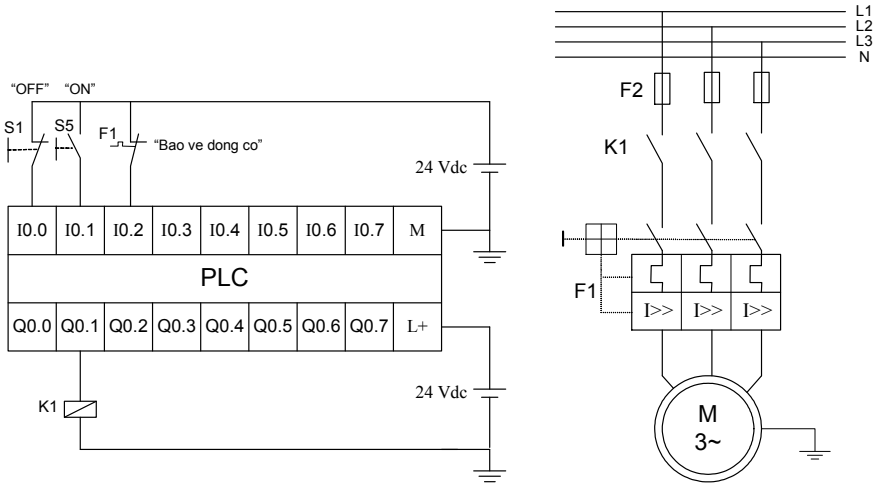
**** Công tắc giới hạn an toàn***

Ở một thiết bị nâng, nếu công tắc hành trình bị hư hỏng thì sẽ có nguy hiểm xảy ra, vì vậy cần phải có các công tắc hành trình an toàn và đèn báo tiếp điểm bị hư hỏng.

Hình 12.16: Sử dụng công tắc giới hạn an toàn

- ***Công tắc bảo vệ động cơ***

Công tắc bảo vệ động cơ là một công tắc 3 cực bảo vệ quá tải cho động cơ. Chúng được lắp đặt trực tiếp vào mạch điện chính của động cơ được điều khiển. Tín hiệu hồi tiếp về của công tắc bảo vệ động cơ được nối vào ngõ vào của PLC.



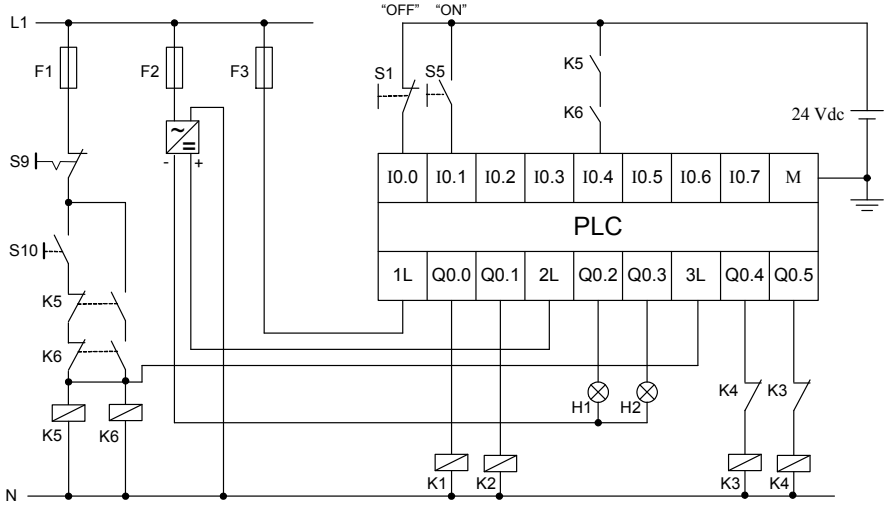
Hình 12.17: Sử dụng công tắc bảo vệ động cơ trong hệ thống điều khiển bằng PLC

- ***Công tắc dừng khẩn cấp***

  Công tắc dừng cấp phải được tách ra khỏi khâu truyền động và thiết bị điều chỉnh. Thông qua tác dụng của nó có thể tránh được sự nguy hiểm cho người và thiết bị.

  Tất cả các thiết bị cảnh báo không được phép tắt khi có sự tác động bởi nút dừng khẩn cấp. Chúng giúp cho biết trạng thái sự cố xảy ra.

  Hình vẽ dưới đây ví dụ một mạch *"DỪNG KHẨN CẤP"*.



Hình 12.18: Ví dụ mạch "DỪNG KHẨN CẤP" trong hệ thống điều khiển bằng PLC

Các contactor K1, K2 là các khâu không nguy hiểm vì vậy không cần thiết phải cắt mạch bằng nút dừng khẩn cấp S9. Các đèn H1, H2 là các thiết bị cảnh báo. Các contactor K3, K4 dùng để điều khiển các động cơ, đây là khâu nguy hiểm nên nhất thiết phải bị cắt điện nếu nút dừng khẩn cấp S9 được ấn. Khi nút ***dừng khẩn cấp S9*** được tác động thì các contactor K5, K6 mất điện, các tiếp điểm K5, K6 được nối với ngõ vào I0.4 (dùng cho dừng khẩn cấp) sẽ trở về trạng thái bình thường (thường hở), thông qua chương trình K3 và K4 sẽ bị mất điện.

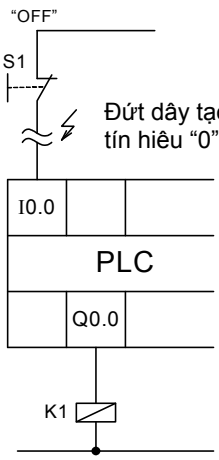
- ***Lập trình an toàn khi đứt dây***

  Lập trình an toàn khi đứt dây có nghĩa là khi đứt dây ở một tín hiệu ngõ vào thì cũng không có nguy hiểm xảy ra. Ví dụ trong hình 3.15 là trường hợp đứt dây sẽ không xảy ra sự cố nguy hiểm.

  Sự đứt dây có thể gây ra tác dụng nguy hiểm, nếu tín hiệu "0" ngăn cản sự cắt truyền động, đóng mạch truyền động hoặc ngăn cản các cảnh báo nguy hiểm. Ngược lại sự đứt dây có thể không gây nguy hiểm, tín hiệu "0" cắt truyền động, ngăn cản sự đóng mạch truyền động và đóng các cảnh báo nguy hiểm, mặc dù không có nguy hiểm tồn tại.

Từ sự suy đoán này có thể đưa ra các yêu cầu sau cho các tín hiệu ngõ vào:

- Bộ phát tín hiệu để truyền động phải có tín hiệu "1" khi tác động nó (vd: tiếp điểm thường hở).
- Bộ phát tín hiệu để cắt truyền động khi tác động phải có tín hiệu "0" (vd: tiếp điểm thường đóng).



Hình 12.19: Sự cố đứt dây

- Bộ phát tín hiệu để cảnh báo nguy hiểm, khi tác động hay biểu thị nguy hiểm phải có tín hiệu "0" ở ngõ vào PLC

Nếu một bộ phát tín hiệu trong điều khiển thi hành nhiều chức năng thì cần phải được xem xét, chức năng nào cần được thực hiện trước cũng như chức năng nào biểu diễn sự quan trọng ở kỹ thuật an toàn. Ở đây phải đặt ra câu hỏi: Sự điều khiển xảy ra như thế nào khi đứt dây?

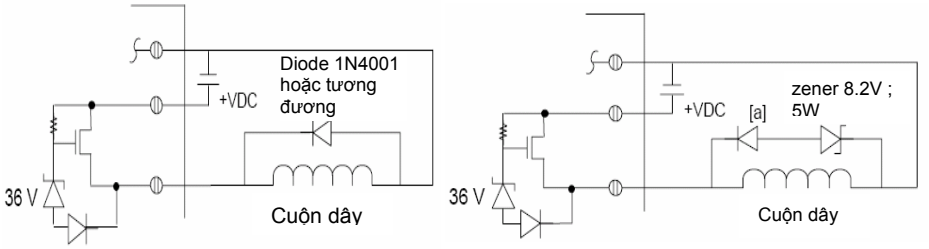
Với sự xem xét có tính nguyên tắc này cho phép thiết bị điều khiển từ chương trình thực hiện an toàn ở các bước tiếp theo. Nếu các yêu cầu an toàn được đặt cao hơn, thì lỗi nguy hiểm phải được nhận biết thông qua các biện pháp phụ và ngăn cản các tác dụng của nó.

12.4 Bảo vệ các ngõ ra PLC

Trường hợp các ngõ ra của PLC nối với các cuộn kháng thì cần phải bảo vệ cho chúng để tránh hiện tượng quá áp khi ngõ ra mất điện. Tùy theo ngõ ra được thiết kế cho ứng dụng mà có thể sử dụng các linh kiện thích hợp để bảo vệ.

12.4.1 Bảo vệ ngõ ra dùng Transistor

Ngõ ra S7-200 DCTransistor có diode zenner để bảo vệ cho nó. Việc lắp thêm một diode bên ngoài cũng giúp cho việc bảo vệ ngõ ra khi tải mắc với cuộn cảm để tránh quá áp trên các diode nội. Có hai cách lắp các mạch bảo vệ như hình 12.20 và 12.21 (trích từ sổ tay S7-200). Trong trường hợp này cũng có thể sử dụng mạch bảo vệ dùng diode hoặc diode kết hợp với zenner nhưng điện áp U_Z của Zenner phải lấy đến 36V.

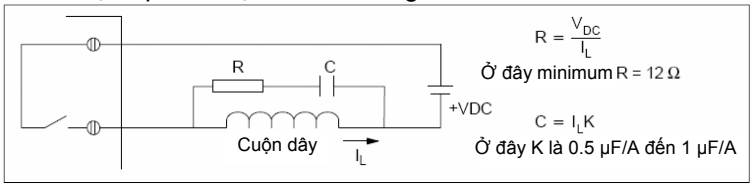


12.4.2 Bảo vệ ngõ ra Rơle có nguồn điều khiển DC

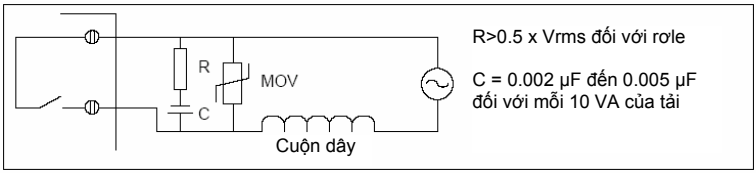
Trong trường hợp này người ta thường sử dụng mạng điện trở/tụ điện và điện áp điều khiển có thể đến 30VDC.

12.4.3 Bảo vệ ngõ ra Rơle và ngõ ra AC có nguồn điều khiển AC

Khi sử dụng rơle hoặc ngõ ra AC để đóng cắt tải 115V/220 VAC, thì có thể bảo vệ bằng điện trở/tụ điện hoặc cũng có thể sử dụng Varistor để giới hạn điện áp đỉnh nhưng chú ý rằng điện áp làm việc của Varistor ít nhất phải lớn hơn 20% điện áp làm việc bình thường.



Hình 12.18: Mạch bảo vệ dùng điện trở/tụ điện cho ngõ ra relay có nguồn điều khiển DC



Hình 12.19: Mạch bảo vệ ngõ ra relay có nguồn điều khiển AC.

12.5 Câu hỏi và bài tập

BT 12.1: Hãy giải thích tại sao nút nhấn dừng phải là thường đóng và nút nhấn khởi động phải là thường hở?

BT 12.2: Hãy cho biết điều gì xảy ra nếu một nút nhấn thường đóng được sử dụng để mở máy trong một hệ thống khi dây nối với nút nhấn bị đứt? Và điều gì xảy ra cho một hệ thống có nút nhấn thường hở được sử dụng làm nút nhấn dừng khi dây nối với nút nhấn bị đứt?

BT 12.3: Hãy vẽ sơ đồ nối dây cho PLC có các ngõ vào được nối với một cảm biến PNP và một cảm biến NPN. Các ngõ ra được nối với hai đèn báo công suất nhỏ 24VDC, hai relay 24VDC để điều khiển hai contactor tương ứng. Trong mạch có gắn hệ thống dừng khẩn cấp.

BT 12.4: Hãy vẽ sơ đồ điện và sơ đồ khí nén cho một hệ thống điều khiển bằng PLC. Hệ thống bao gồm các linh kiện được liệt kê dưới đây. Trong mạch có gắn hệ thống dừng khẩn cấp.

- Một động cơ 3 pha/50 HP
- Một cảm biến NPN
- Một nút nhấn thường hở (NO)
- Một công tắc hành trình thường đóng (NC)
- Hai đèn báo công suất thấp 24VDC
- Một van có 2 cuộn dây 24VDC.

13 Chuyển điều khiển kết nối cứng sang điều khiển bằng PLC.

13.1 Kết nối ngõ vào/ ra của PLC từ một sơ đồ điều khiển có tiếp điểm

Trong nhiều trường hợp, cần cải tạo một hệ thống điều khiển với relay và contactor thành hệ thống điều khiển với PLC. Một câu hỏi đặt ra là chúng ta cần giữ lại những phần nào trong hệ thống điều khiển, còn phần nào sẽ loại bỏ đi?

Để dễ dàng trong việc chuyển đổi, có thể áp dụng phương pháp sau để chuyển đổi từ một hệ thống điều khiển cũ sang điều khiển với PLC:

- **Về phần cứng:**
  - Xác định các bộ tạo tín hiệu (ví dụ: nút nhấn, công tắc, cảm biến . . .) cần thiết nhất trong hệ thống điều khiển, mỗi bộ tạo tín hiệu tùy theo loại tạo ra tín hiệu nào nên được kết nối với một ngõ vào của PLC tương ứng, ví dụ nếu bộ tạo ra tín hiệu nhị phân thì được kết nối với các ngõ vào số, còn bộ tạo ra tín hiệu tương tự thì kết nối với ngõ vào tương tự (ngõ vào analog). Còn các bộ tạo tín hiệu còn lại nếu không cần thiết thì có thể bỏ đi và sẽ được thực hiện bằng chương trình trong PLC.
  - Tương tự xác định các cơ cấu chấp hành (đối tượng điều khiển) cần thiết nhất, thông thường các đối tượng này là các đèn báo, contactor chính, van từ, .v.v.. Tuỳ theo loại mà mỗi đối tượng điều khiển có thể kết nối trực tiếp hoặc gián tiếp với các ngõ ra tương ứng, mỗi một đối tượng điều khiển cần một ngõ ra. Nếu các đối tượng điều khiển cần dòng điều khiển lớn thì yêu cầu phải sử dụng rơ le trung gian. Ví dụ như các contactor chính điều khiển các động cơ công suất lớn thì ngõ ra của PLC sẽ được nối với một rơ le trung gian và thông qua tiếp điểm của rơ le trung gian để điều khiển các contactor này. Còn các đối tượng điều khiển không tác động trực tiếp đến quá trình điều khiển mà chỉ đóng vai trò trung gian hỗ trợ cho quá trình điều khiển như rơ le trung gian thì có thể loại bỏ và được thay thế bằng một ô nhớ nào đó trong chương trình của PLC.
  - Sau khi đã xác định được số lượng các ngõ vào, ngõ ra cần thiết và hệ thống điện cung cấp cho phần điều khiển thì tiến hành đến việc lựa chọn loại PLC phù hợp.

- Thiết lập bảng xác định các ngõ vào/ra với các ngoại vi tương ứng và chú ý ghi chú lại càng chi tiết càng tốt.
- Thực hiện việc nối dây các ngõ vào, ngõ ra của PLC với các bộ tạo tín hiệu điều khiển và đối tượng điều khiển. Trong quá trình nối dây cần lưu ý đến các nguyên tắc an toàn trong hệ thống điều khiển (xem mục 4.3).
- Tất cả việc kết nối dây trong hệ thống điều khiển trước đây sẽ được biến đổi thành chương trình trong PLC.

- **Về phần mềm:**

  Việc viết chương trình có thể thực hiện theo hai cách:

  Cách 1: Tùy theo yêu cầu công nghệ mà có thể thiết lập giải thuật điều khiển và viết chương trình theo giải thuật điều khiển này.

  Cách 2: Vẫn duy trì hoạt động của hệ thống như cũ, hay nối khác đi là không cần thiết phải lập lại giải thuật điều khiển vì tất cả đã được thiết kế trong sơ đồ điều khiển cứng trước đây mà chỉ cần biến đổi sơ đồ điều khiển này thành chương trình trong PLC. Cách này tương đối dễ dàng và có thể không bị lỗi khi lập trình.

  Trong phần này trình bày phương pháp chuyển đổi theo cách 2 theo các bước như sau:

  - Thực hiện viết chương trình lần lượt cho mỗi đối tượng điều khiển, mỗi đối tượng điều khiển được viết ở một đoạn chương trình và có ghi chú cụ thể để dễ dàng sữa lỗi.
  - Chỉ có các điều kiện cần thiết nhất cho đối tượng điều khiển mới được viết vào đoạn chương trình điều khiển nó.
  - Nếu một số đối tượng điều khiển có cùng chung một nhóm điều kiện, thì nhóm điều kiện này nên được được viết riêng ở một đoạn chương trình và cất kết quả vào một ô nhớ trong PLC. Nếu đối tượng điều khiển nào cần nhóm điều kiện này thì chỉ cần lấy kết quả được chứa trong ô nhớ. Điều này giúp cho cấu trúc chương trình mạch lạc và việc đọc chương trình trở nên dễ dàng hơn.
  - Các đối tượng điều khiển không cần thiết (ví dụ contactor trung gian) sẽ được thay thế bằng một ô nhớ trong PLC. Nếu các đối tượng điều khiển nào cần đến tiếp điểm của rơ le trung gian này thì chỉ cần thay thế bằng tiếp điểm của ô nhớ.
  - Tùy theo hệ thống điều khiển có phức tạp hay không mà có thể phân chia thành nhiều khối chương trình để dễ dàng trong quá trình quản lý.

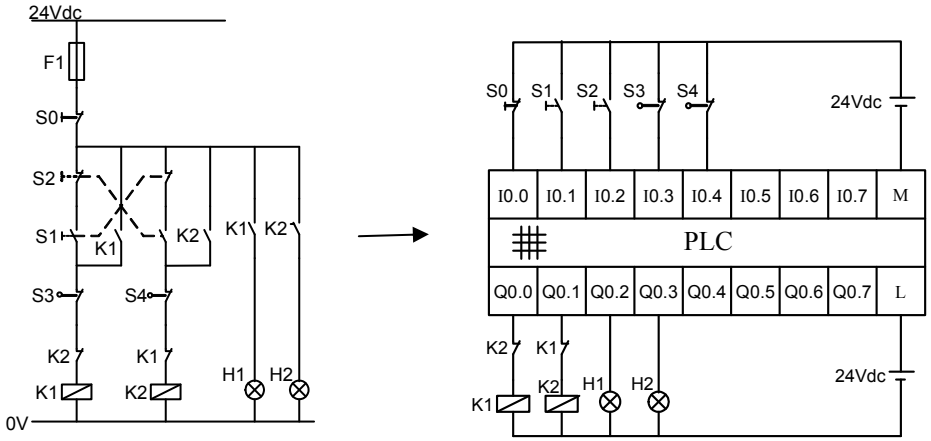
Hình 13.1 là một ví dụ về việc chuyển đổi một sơ đồ điều khiển cửa ra vào cơ quan bằng contactor thành hệ thống điều khiển với PLC (chỉ dừng lại ở việc chuyển đổi kết nối dây, còn chương trình thực hiện ở các chương sau).

Dựa vào các bước trên, ta nhận thấy các nút nhấn, contactor cần thiết dược giữ lại như trong bảng xác định kết nối vào/ra với ngoại vi và PLC được chọn ở đây là loại CPU 224 DC/DC/relay. Do contactor K1 và K2 không được

phép có điện đồng thời nên theo quan điểm an toàn cần phải khóa chéo hai contactor này lại với nhau.

Bảng xác định kết nối vào/ra với ngoại vi

Ký hiệu	*Địa chỉ*	*Chú thích*
S0	I0.0	Nút nhấn dừng, thường đóng
S1	I0.1	Nút nhấn mở cửa, thường hở
S2	I0.2	Nút nhấn đóng cửa, thường hở
S3	I0.3	Công tắc hành trình giới hạn cửa mở, thường đóng
S4	I0.4	Công tắc hành trình giới hạn cửa đóng, thường đóng
K1	Q0.0	Cuộn dây contactor K1, điều khiển mở cửa
K2	Q0.1	Cuộn dây contactor K2, điều khiển đóng cửa
H1	Q0.2	Đèn báo cửa đang mở
H2	Q0.3	Đèn báo cửa đang đóng



a) b)

Hình 13.1: Kết nối ngõ vào/ ra của PLC từ một sơ đồ điều khiển có tiếp điểm

13.2 Chuyển đổi điều khiển từ contactor thành PLC

Contactor là một chuyển mạch bằng điện. Tùy theo loại và phạm vi ứng dụng mà nó được phân thành 2 loại là contactor chính và contactor phụ.

Contactor chính là contactor chịu tải, nó được sử dụng để đóng, cắt điện cho tải như động cơ, thiết bị chiếu sáng, thiết bị nung, van từ, thắng v.v… Trong ứng dụng với điều khiển bằng PLC thì contactor chính là thiết bị không thể thiếu.

Cotactor phụ chỉ được sử dụng để tăng thêm tiếp điểm trong mạch điều khiển. Chính vì thế trong việc điều khiển với PLC thì các contactor phụ được thay thế bằng các ô nhớ (bit Memory) trong chương trình PLC.

Các bộ định thời (timer) như đóng mạch chậm hoặc mở mạch chậm trong mạch điều khiển với relay và contactor sẽ không cần thiết trong điều khiển với PLC, chúng sẽ được thay thế bằng các timer tương ứng trong chương trình PLC.

Trong việc chuyển đổi, các bộ tạo ra tín hiệu như nút nhấn, công tắc, công tắc hành trình, cảm biến v.v… thật sự cần thiết sẽ được giữ lại. Còn những tiếp điểm không cần thiết sẽ được xử lý thông qua chương trình.

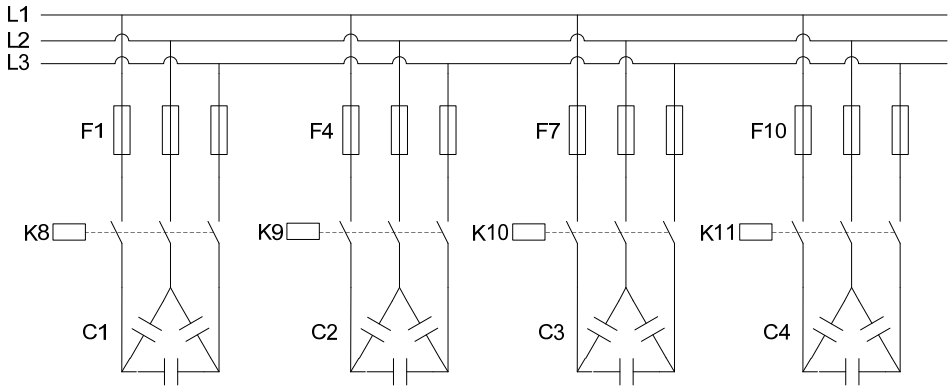
Việc thực hiện chuyển đổi từ điều khiển bằng contactor thành PLC có thể xem chương 4 (kết nối dây PLC với ngoại vi). Ngoài ra cần chú ý thêm một số điểm sau:

- Các tiếp điểm được nối song song tương ứng là các cổng OR trong chương trình PLC
- Các tiếp điểm được nối nối tiếp tương ứng là các cổng AND.
- Về phương diện an toàn tránh sự cố do đứt dây thì các nút nhấn mở máy phải là thường hở (loại NO (Normal Opened)). Các nút nhấn dừng máy phải là thường đóng (loại NC (Normal Closed)).
- Mỗi nút nhấn, công tắc, cảm biến v.v… tùy theo nhiệm vụ có thể nối với một ngõ vào (điều này có nghĩa là không nhất thiết một bộ tạo ra tín hiệu nhị phân phải nối với một ngõ vào số).
- Mỗi một ngõ ra của PLC sẽ được kết nối với một đối tượng điều khiển như đèn báo, cuộn dây relay, cuộn dây contactor. Tuy nhiên cần phải chú ý đến phương diện an toàn và điện áp điều khiển. Nếu điện áp cuộn dây relay, đèn báo hoặc cuộn dây contactor khác với điện áp của các ngõ ra thì bắt buộc phải sử dụng relay làm thiết bị trung gian.
- Hệ điều hành trong PLC hoàn toàn không biết đâu là tiếp điểm thường đóng đâu là tiếp điểm thường hở mà chỉ biết ngõ vào PLC có điện áp (mức logic “1”) hay không có điện áp (mức logic “0”). Cho nên khi viết chương trình cần đặc biệt chú ý đến vấn đề này (xem lại kỹ chương 7 phép toán nhị phân).
- Khi sử dụng với các lệnh S và R trong chương trình PLC cần *chú ý các qui tắc* sau:
    - *Các điều kiện làm cho đối tượng điều khiển ở mức tích cực (logic “1”) được sử dụng với lệnh S.*
    - *Các điều kiện làm cho đối tượng điều khiển ở mức không tích cực (logic “0”) được sử dụng với lệnh R.*
    - *Khi viết lệnh S cho một đối tượng điều khiển thì nhất thiết (tùy theo yêu cầu công nghệ) phải có một lệnh R cho đối tượng điều khiển đó.*

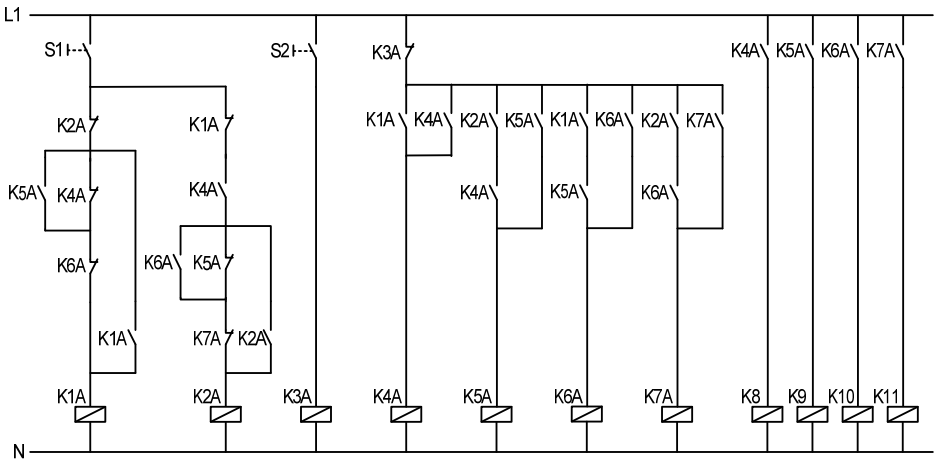
- *Nếu lệnh S được viết trước lệnh R thì kết quả thu được sẽ là kết quả của lệnh R nếu cả hai điều kiện cho S và R cùng ở mức logic "1" nghĩa là đối tượng điều khiển ở mức logic "0".*
- *Nếu lệnh R được viết trước lệnh S thì kết quả thu được sẽ là kết quả của lệnh S nếu cả hai điều kiện cho S và R cùng ở mức logic "1" nghĩa là đối tượng điều khiển ở mức logic "1".*
- *Khi đã viết chương trình với lệnh S thì không được sử dụng tiếp điểm tự duy trì (loại bỏ tiếp điểm tự duy trì).*
- *Tùy theo công nghệ khi sử dụng các điều kiện cho lệnh R thì ở trạng thái bình thường các điều kiện này phải có mức logic "0".*

13.2.1 Điều khiển thiết bị bù công suất phản kháng

Sơ đồ mạch động lực và điều khiển



Hình 13.1: Mạch động lực của thiết bị đóng tụ bù.



Hình 13.2: Sơ đồ mạch điều khiển bằng contactor thiết bị đóng tụ bù

Mô tả:

Tùy theo yêu cầu mà các tụ bù công suất phản kháng C1, C2, C3, C4 sẽ được đóng vào lưới điện. Cứ mỗi lần ấn nút nhấn S1 thì một bộ tụ bù được đóng vào lưới điện. Để cắt tụ bù ra khỏi lưới thì ấn nút nhấn S2.

Thực hiện với PLC:

Phân tích:

Trong mạch điều khiển sử dụng 2 nút nhấn S1 và S2, đây là các nút nhấn cần thiết để đóng và cắt tụ bù cho nên cần phải giữ lại. Như vậy để thực hiện điều khiển bằng PLC ta sử dụng 2 ngõ vào số để kết nối với 2 nút nhấn này.

Trong sơ đồ mạch điều khiển trên gồm có 4 contactor chính K8, K9, K10, K11. Đây là các thiết bị không thể thiếu và bắt buộc phải giữ lại để đóng cắt tụ với lưới điện. Để điều khiển 4 contactor này ta sẽ dùng 4 ngõ ra của PLC.

Chú ý: Để đơn giản và không lặp lại những mô tả như trong chương 7, các bài tập này được sử dụng với CPU 224 AC/DC/Relay.

Để điều khiển 4 contactor chính theo nhiệm vụ đặt ra cần đến 7 contactor phụ K1A, K2A, K3A, K4A, K5A, K6A, K7A. Các contactor phụ này là các thiết bị hỗ trợ trong điều khiển bằng contactor vì vậy không cần thiết phải giữ lại. Nó sẽ được thay thế bằng các ô nhớ trong PLC.

Đối với mạch này, người thiết kế có thể sử dụng hai cách lập trình

Cách 1: Chuyển thành chương trình theo như sơ đồ điều khiển đã trình bày

Cách 2: Theo yêu cầu công nghệ đặt ra

Để rõ ràng, ta sẽ thực hiện theo 2 cách

Cách 1: theo sơ đồ mạch điều khiển contactor có sẵn

Để tiện lợi trong quá trình chuyển đổi ta nên lập một bảng ký hiệu để kết nối giữa PLC và các thiết bị ngoại vi cũng như các qui đổi tương ứng.

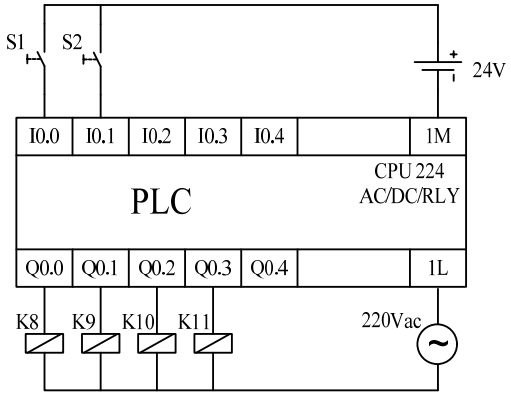
Khi lập bảng ký hiệu nên ghi chú đầy đủ thông tin để dễ dàng trong quá trình viết chương trình.

Bảng ký hiệu

Ký hiệu	***Địa chỉ (PLC)***	***Chú thích***
Biến ngõ vào		
S1	I0.0	Nút nhấn đóng tụ bù vào lưới điện, thường hở
S2	I0.1	Nút nhấn cắt tụ bù khỏi lưới điện, thường hở
Biến ngõ ra		
K8	Q0.0	Contactor chính K8, đóng tụ bù C1
K9	Q0.1	Contactor chính K9, đóng tụ bù C2

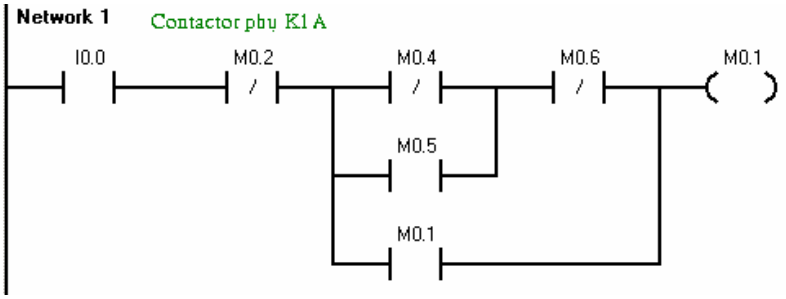
K10	Q0.2	Contactor chính K10, đóng tụ bù C3
K11	Q0.3	Contactor chính K11, đóng tụ bù C4
Biến trung gian		
K1A	M0.1	Contactor phụ K1A
K2A	M0.2	Contactor phụ K2A
K3A	M0.3	Contactor phụ K3A
K4A	M0.4	Contactor phụ K4A
K5A	M0.5	Contactor phụ K5A
K6A	M0.6	Contactor phụ K6A
K7A	M0.7	Contactor phụ K7A

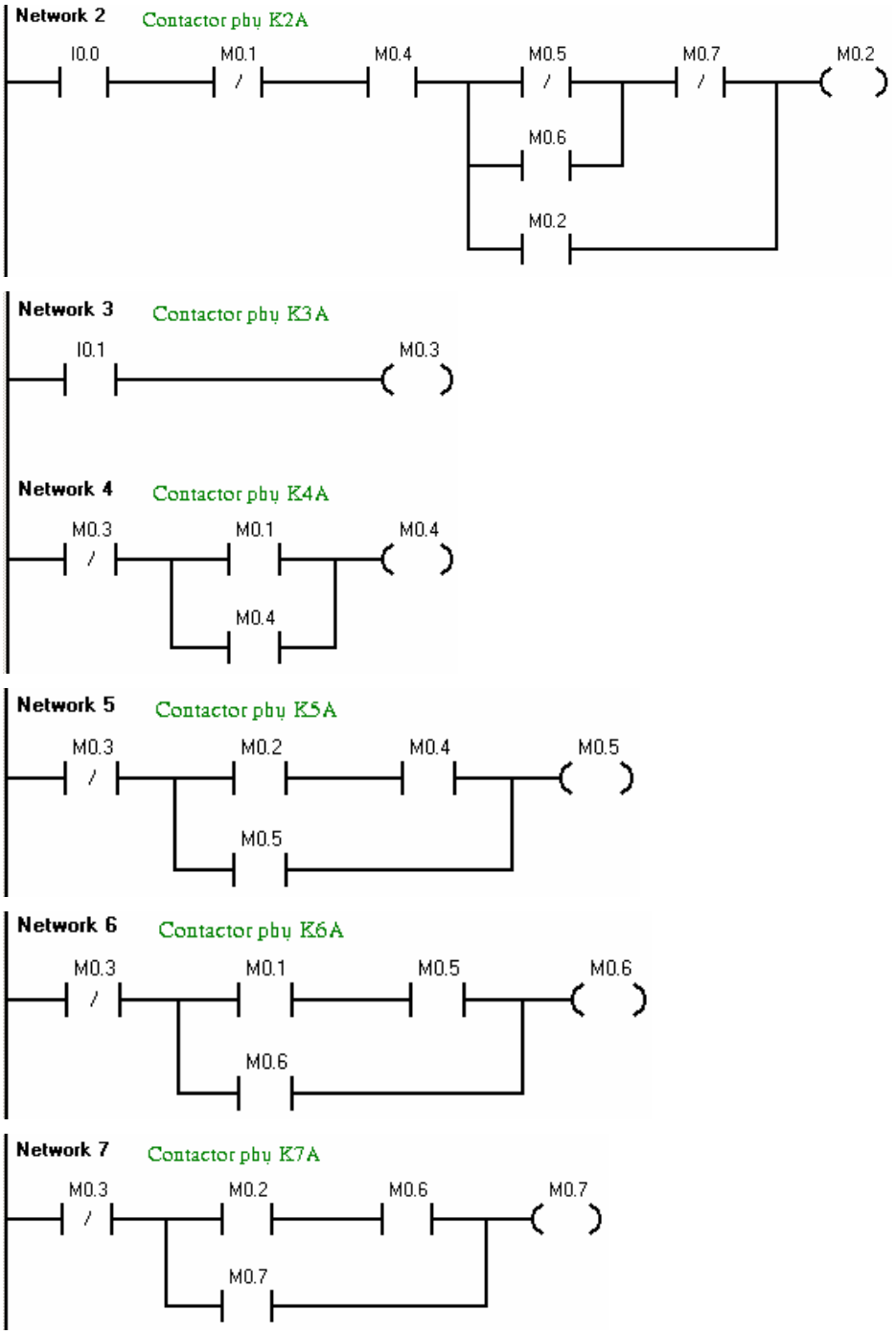
Kết nối dây với PLC:

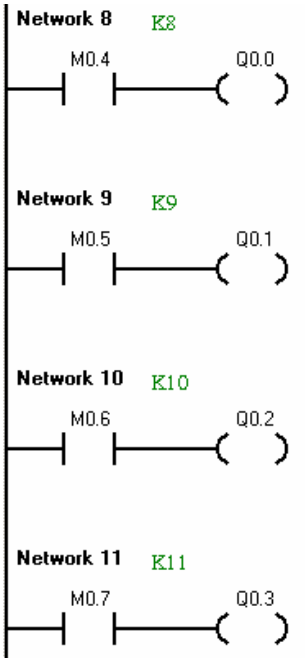


Hình 13.3: Nối dây các ngoại vi với ngõ vào ra PLC khi điều khiển bằng PLC

Chương trình PLC ở LAD:




Network 2
Contactor phụ K2A
I0.0
M0.1
M0.4
M0.5
M0.7
M0.2
M0.6
M0.2
Network 3
Contactor phụ K3A
I0.1
M0.3
Network 4
Contactor phụ K4A
M0.3
M0.1
M0.4
M0.4
Network 5
Contactor phụ K5A
M0.3
M0.2
M0.4
M0.5
M0.5
Network 6
Contactor phụ K6A
M0.3
M0.1
M0.5
M0.6
M0.6
Network 7
Contactor phụ K7A
M0.3
M0.2
M0.6
M0.7
M0.7



Chương trình PLC ở STL:

Network 1 Contactor phụ K1A

```
LD      I0.0
AN      M0.2
LDN     M0.4
O       M0.5
AN      M0.6
O       M0.1
ALD
=       M0.1
```

Network 2 Contactor phụ K2A

```
LD      I0.0
AN      M0.1
A       M0.4
LDN     M0.5
O       M0.6
AN      M0.7
O       M0.2
ALD
=       M0.2
```

Network 3 Contactor phụ K3A

```
LD      I0.1
=       M0.3
```

Network 4 Contactor phụ K4A

```
LDN     M0.3
LD      M0.1
O       M0.4
ALD
=       M0.4
```

Network 5 Contactor phụ K5A

```
LDN     M0.3
LD      M0.2
A       M0.4
O       M0.5
ALD
=       M0.5
```

```
Network 6    Contactor phụ K6A
LDN     M0.3
LD      M0.1
A       M0.5
O       M0.6
ALD
=       M0.6

Network 7    Contactor phụ K7A
LDN     M0.3
LD      M0.2
A       M0.6
O       M0.7
ALD
=       M0.7
```

```
Network 8    K8
LD      M0.4
=       Q0.0

Network 9    K9
LD      M0.5
=       Q0.1

Network 10   K10
LD      M0.6
=       Q0.2

Network 11   K11
LD      M0.7
=       Q0.3
```

Cách 2: Theo yêu cầu công nghệ

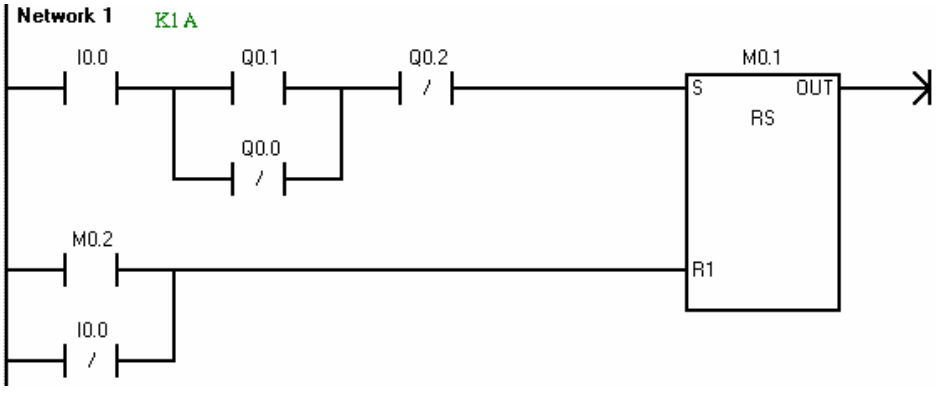
Theo cách thức điều khiển đặt ra, cứ mỗi lần tác động S1 thì một contactor chính được đóng điện, tác động S2 thì cắt điện tòan bộ.

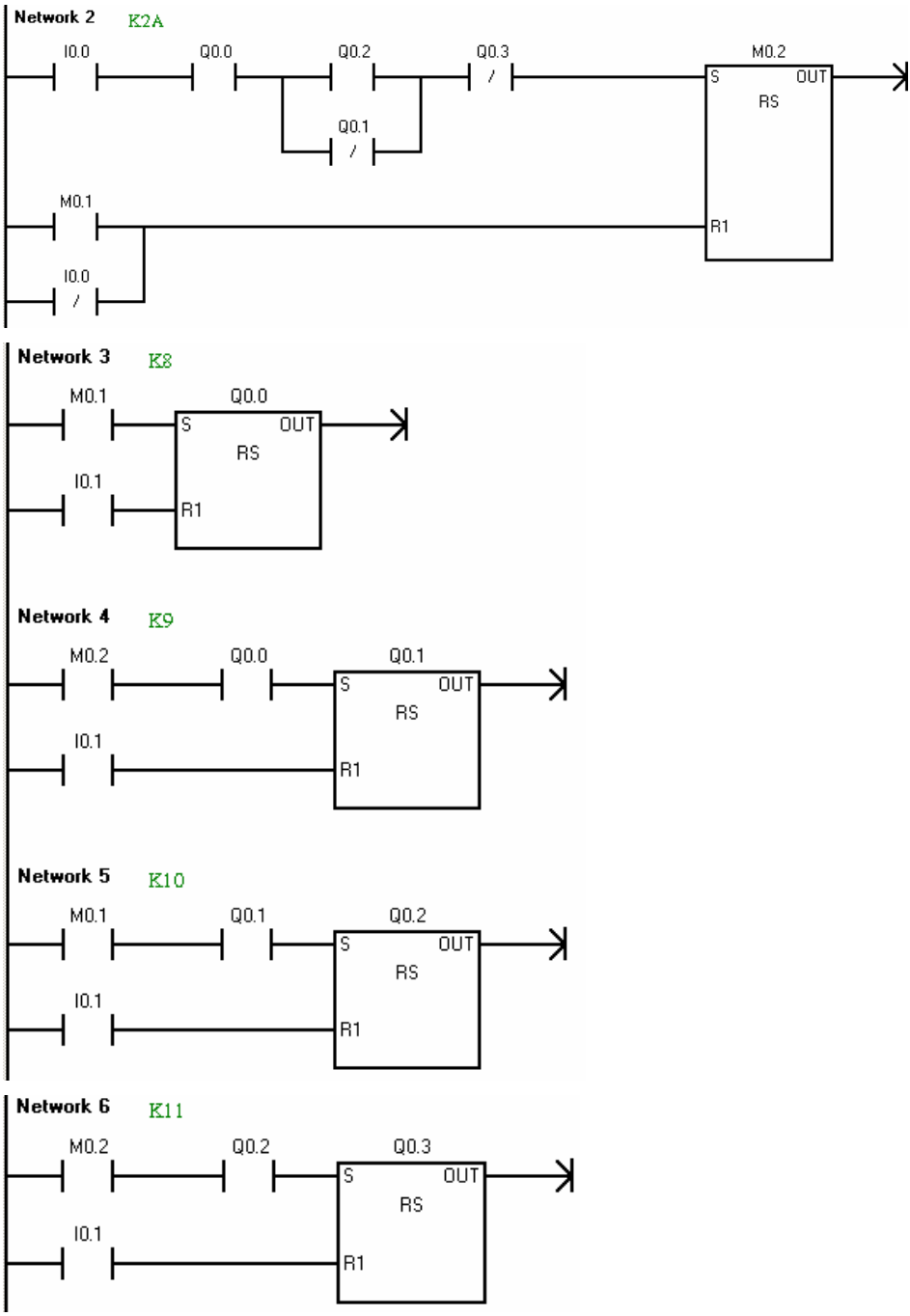
Mục đích của việc thêm các contactor phụ là để tăng thêm số lượng tiếp điểm. Nếu thực hiện bằng chương trình ta có thể đưa trực tiếp ra các ngõ ra từ Q0.0 đến Q0.3 mà không cần phải qua các ô nhớ M0.4 đến M0.7. M0.3 cũng có thể loại bỏ, thay thế trực tiếp bằng nút nhấn S2 (I0.1).

Từ việc phân tích mạch điều khiển, ta có thể làm cho chương trình được đơn giản hơn. Ngoài ra ta thay thế luôn mạch tự duy trì bằng một khâu SR.

Chương trình bây giờ rất đơn giản như sau:

Chương trình được viết ở LAD:





Chương trình viết ở STL:

Network 1 K1A

```
LD      I0.0
LD      Q0.1
ON      Q0.0
ALD
AN      Q0.2
LD      M0.2
ON      I0.0
NOT
LPS
A       M0.1
=       M0.1
LPP
ALD
O       M0.1
=       M0.1
```

Network 2 K2A

```
LD      I0.0
A       Q0.0
LD      Q0.2
ON      Q0.1
ALD
AN      Q0.3
LD      M0.1
ON      I0.0
NOT
LPS
A       M0.2
=       M0.2
LPP
ALD
O       M0.2
=       M0.2
```

Network 3 K8

```
LD      M0.1
LD      I0.1
NOT
LPS
A       Q0.0
=       Q0.0
LPP
ALD
O       Q0.0
=       Q0.0
```

Network 4 K9

```
LD      M0.2
A       Q0.0
LD      I0.1
NOT
LPS
A       Q0.1
=       Q0.1
LPP
ALD
O       Q0.1
=       Q0.1
```

Network 5 K10

```
LD      M0.1
A       Q0.1
LD      I0.1
NOT
LPS
A       Q0.2
=       Q0.2
LPP
ALD
O       Q0.2
=       Q0.2
```

Network 6 K11

```
LD      M0.2
A       Q0.2
LD      I0.1
NOT
LPS
A       Q0.3
=       Q0.3
LPP
ALD
O       Q0.3
=       Q0.3
```

13.2.2 Thiết bị nghiền

Phần này trình bày một khâu trong hệ thống điều khiển sản xuất gồm là vận chuyển vật liệu nghiền. Vật liệu nghiền từ cối nghiền sẽ được băng tải vận chuyển vào một xe đặt dưới băng tải.

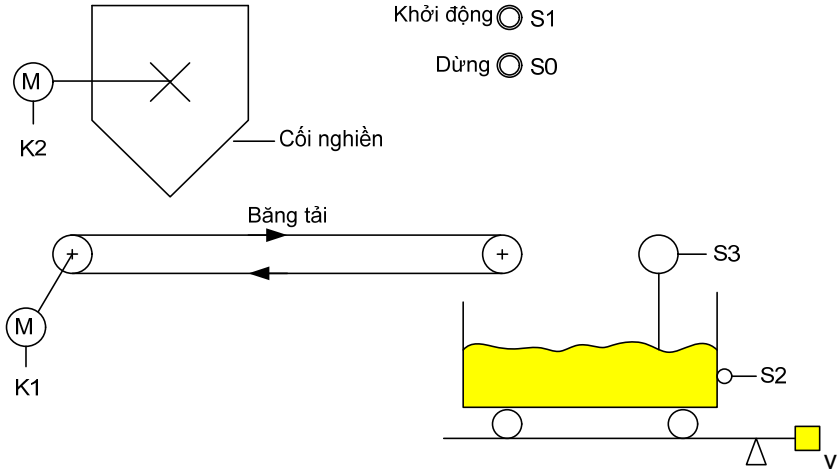
Quá trình vận chuyển vật liệu đã được nghiền được khởi động nếu xe đã vào vị trí vận chuyển và nút nhấn S1 được ấn. Để đảm bảo an toàn thì

trước tiên băng tải phải hoạt động trước 2 giây sau đó mới đóng điện cho cối nghiền.

Khi xe đầy (được báo bởi cảm biến cân) thì cối nghiền ngay lập tức bị ngắt điện. Băng tải còn tiếp tục vận chuyển cho hết vật liệu trên băng tải xuống xe với thời gian là 3 giây.

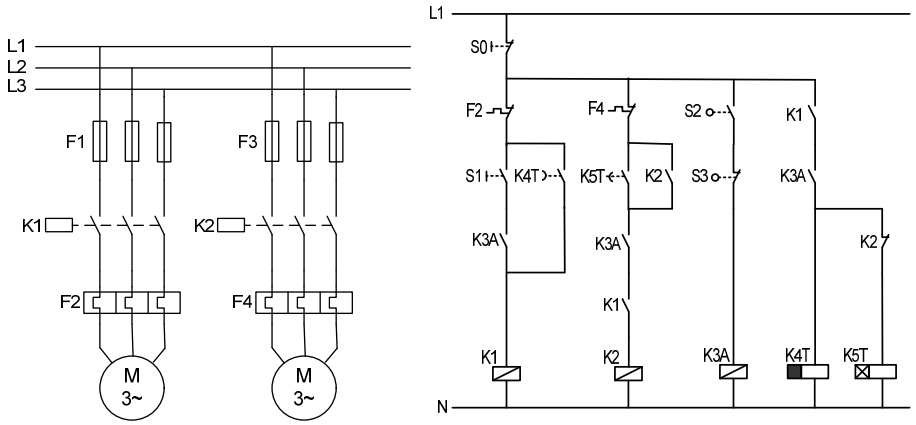
Trong quá trình hoạt động có thể dừng bằng nút nhấn S0.

Sơ đồ công nghệ:



Hình 13.4: Sơ đồ công nghệ thiết bị nghiền

Sơ đồ mạch động lực và điều khiển bằng contactor:



Hình 13.5: Mạch động lực và điều khiển bằng contactor của thiết bị nghiền.

Contactor chính K1 điều khiển động cơ M1 của băng tải, contactor chính K2 điều khiển động cơ M2 của cối nghiền.

Phân tích:

Trong mạch điều khiển sử dụng các nút nhấn S0, S1, công tắc hành trình S2, tín hiệu báo xe đầy S3. Đây là các tín hiệu điều khiển không thể loại bỏ. Cần phải có 4 ngõ vào cho các tín hiệu này. Ngoài ra còn có tín hiệu bảo vệ quá dòng động cơ là F2 và F4 cũng cần được nối với các ngõ vào. Một điều cần chú ý là các nút nhấn, công tắc hành trình, tiếp điểm bảo vệ quá dòng là các khâu cơ khí cho nên không thể thay đổi được mà phải sử dụng lại (nghĩa là giữ nguyên tính nguyên thủy của nó). Nên khi chuyển thành chương trình thì vẫn đảm bảo hoạt động đúng theo yêu cầu công nghệ mà sơ đồ mạch điều khiển bằng contator thể hiện và không có sự thay đổi nào với các bộ phát tín hiệu này.

Các contactor chính K1 và K2 cần phải có 2 ngõ ra để điều khiển

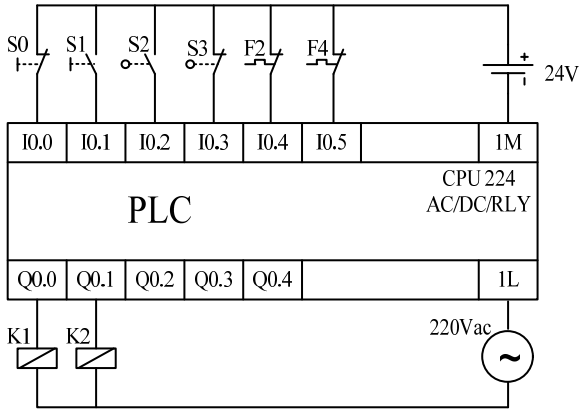
Contactor phụ K3A được thay thế bằng một ô nhớ.

Các bộ định thời K4T được thay thế bằng một timer OFF delay, K5T được thay thế bằng một timer ON delay.

Bảng ký hiệu

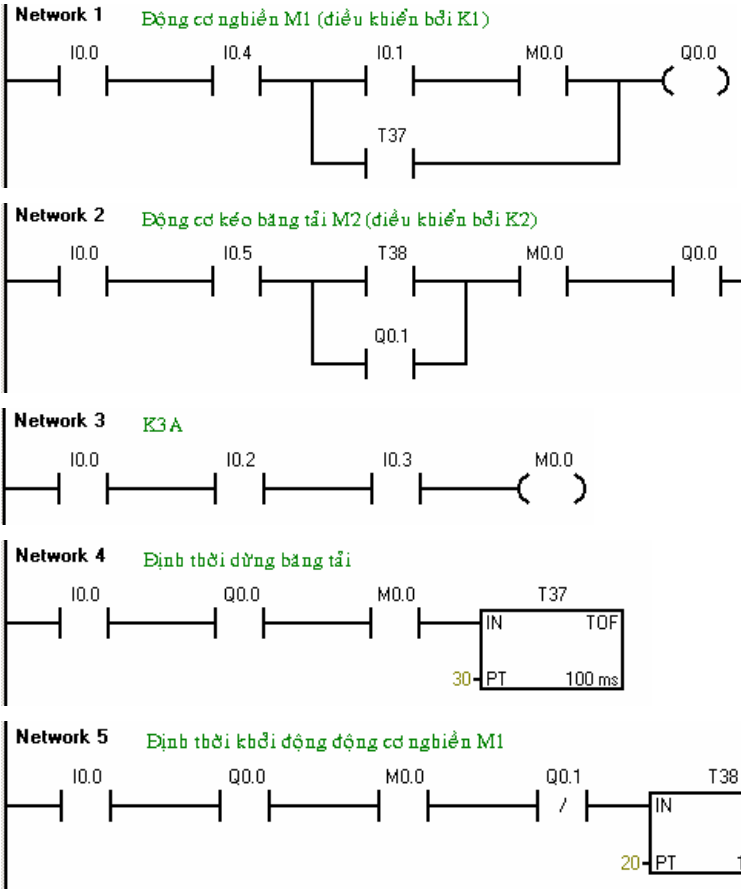
Ký hiệu	***Địa chỉ (PLC)***	***Chú thích***
Biến ngõ vào		
S0	I0.0	Nút nhấn dừng, thường đóng (NC)
S1	I0.1	Nút nhấn khởi động hệ thống, thường hở (NO)
S2	I0.2	Công tắc hành trình, báo xe đúng vị trí (NO)
S3	I0.3	Tín hiệu báo xe đầy, thường đóng (NC)
F2	I0.4	Tiếp điểm bảo vệ quá dòng M1, (NC)
F4	I0.5	Tiếp điểm bảo vệ quá dòng M2, (NC)
Biến ngõ ra		
K1	Q0.0	Contactor chính K1, điều khiển đ.cơ nghiền M1
K2	Q0.1	Contactor chính K2, điều khiển đ.cơ băng tải M2
Biến trung gian		
K3A	M0.0	Contactor phụ K3A
Bộ định thời		
K4T	T37	OFF delay timer, định thời dừng băng tải, 3s
K5T	T38	ON ddelay timer, định thời khởi động M1, 2s

Kết nối dây với PLC:



Hình 13.6: Sơ đồ nối dây ngoại vi với ngõ vào ra của PLC

Chương trình PLC ở LAD:



Chương trình PLC ở STL:

```
Network 1   Động cơ nghiền M1 (điều khiển bởi K1)
LD      I0.0
A       I0.4
LD      I0.1
A       M0.0
O       T37
ALD
=       Q0.0

Network 2   Động cơ kéo băng tải M2 (điều khiển bởi K2)
LD      I0.0
A       I0.5
LD      T38
O       Q0.1
ALD
A       M0.0
A       Q0.0
=       Q0.1

Network 3   K3A
LD      I0.0
A       I0.2
A       I0.3
=       M0.0

Network 4   Định thời dừng băng tải
LD      I0.0
A       Q0.0
A       M0.0
TOF     T37, 30

Network 5   Định thời khởi động động cơ nghiền
LD      I0.0
A       Q0.0
A       M0.0
AN      Q0.1
TON     T38, 20
```

13.3 Điều khiển khí nén

Trong kỹ thuật điều khiển bằng khí nén, người ta phân biệt các phần tử điều khiển sau:

- *Khâu tín hiệu:* Phát ra tín hiệu khi phần tử điều khiển đạt đến một giá trị xác định đối với các đại lượng vật lý.
- *Khâu điều khiển:* Phản ứng lại theo các tín hiệu đơn và có ảnh hưởng đến trạng thái của khâu điều chỉnh.
- *Khâu điều chỉnh:* Điều khiển dòng năng lượng sinh công và thay đổi trạng thái của các phần tử làm việc.

Nếu thực hiện thay thế mạch điều khiển khí nén bằng chương trình điều khiển PLC, thì khâu điều chỉnh điều khiển cho các phần tử làm việc bây giờ điện từ. Dù các van xung điện từ hay van điện từ sử dụng lò xo được sử dụng, thì nó còn phụ thuộc vào yêu cầu công nghệ và an toàn. Khi chuyển đổi thành chương trình PLC thì các khâu này cần giữ lại.

Van xung trong kỹ thuật điều khiển khí nén có hai ngõ vào điều khiển và có đặc tính nhớ. Theo cách thức hoạt động có thể so sánh nó với khâu nhớ RS. Việc chuyển đổi thật sự đơn giản nếu ta thay tất cả van xung bằng khâu nhớ RS. Ngõ vào điều khiển của khâu điều chỉnh SET của van tương ứng với điều kiện cho set, và ngõ vào còn lại tương ứng với reset của khâu RS.

Van xung sử dụng 2 cuộn dây từ. Để điều khiển, một cuộn dây sẽ sử dụng ngõ ra không đảo của khâu nhớ RS. Còn cuộn dây thứ hai ta sử dụng ngõ ra đảo của khâu nhớ RS.

Tùy theo yêu cầu công nghệ mà mạch điều khiển khí nén đảm nhận, mà ta có thể sử dụng hướng điều khiển cho các van tương ứng. Sau khi tất cả đã được xác định, mạch điều khiển khí nén có thể được chuyển đổi trực tiếp thành chương trình ở LAD.

Một số qui tắc cần chú ý:

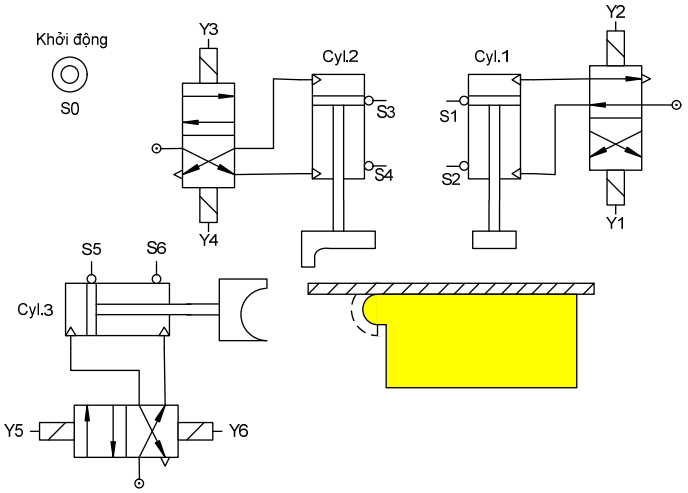
- Khâu điều chỉnh của xylanh làm việc được thay thế bằng van điện từ.
- Tất cả các van xung được thay thế bằng khâu nhớ RS.
- Xác định được tính logic của mạch.
- Chuyển đổi mạch thành chương trình PLC.

13.3.1 Máy uốn thanh kim loại

Các thanh kim loại cần được uốn một đầu theo theo một khuôn cho trước (sơ đồ công nghệ). Qui trình hoạt động của máy như sau:

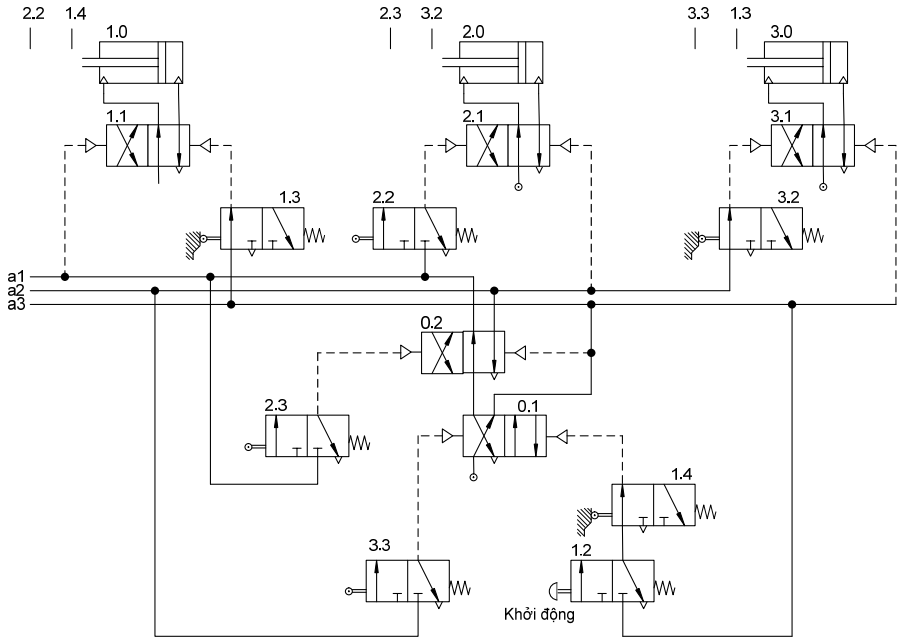
- Thanh kim loại cần uốn được đặt lên khuôn uốn
- Ấn nút khởi động S0 thì xy lanh Cyl.1 hạ xuống để giữ lấy thanh kim loại.
- Khi thanh kim loại được giữ chặt (nhận biết bởi công tắc hành trình S2) thì xy lanh Cyl.2 hạ xuống để uốn thanh kim loại vuông góc trước. Sau khi uốn xong thì tự động nâng lên nhờ công tắc hành trình S4.
- Khi xy lanh Cyl.2 trở về vị trí cơ bản (nhận biết bởi S3) thì xy lanh Cyl.3 được đẩy để uốn thanh kim loại ở giai đoạn uốn cuối theo định hình của khuôn uốn. Khi xy lanh Cyl.3 đến vị trí S6 thì tự động rút ngược về.
- Khi xy lanh Cyl.3 rút về đến vị trí cơ bản (nhận biết bởi S5) thì xy lanh Cyl.1 cũng rút về vị trí cơ bản của nó (nhận biết bởi S1). Lúc này thanh kim loại được tự do. Người sử dụng có thể lấy ra và đặt một thanh kim loại mới vào. Và một chu kỳ mới lại có thể bắt đầu.

Sơ đồ công nghệ:



Hình 13.7: Sơ đồ công nghệ máy uốn thanh kim loại

Sơ đồ mạch điều khiển bằng khí nén:



Hình 13.8: Sơ đồ mạch điều khiển bằng khí nén.

Phân tích:

Từ sơ đồ điều khiển bằng khí nén ta nhận thấy các van xung chính trong mạch là 1.1, 2.1 và 3.1. Khi chuyển sang điều khiển bằng chương trình nhất thiết ta phải thay các van này bằng các van xung điện từ có đặc tính nhớ. Mỗi van xung điện từ có 2 cuộn dây. Vì vậy cần phải có 2 ngõ ra số để điều khiển mỗi van. Tổng cộng ta cần có 6 ngõ ra để điều khiển 3 van này. Để thực hiện điều khiển bằng chương trình PLC, các van xung được thay thế bởi các khâu RS, các ngõ ra của các khâu nhớ có thể được sử dụng để điều khiển trực tiếp các van xung điện từ thay thế Y1, Y3, và Y5 cũng như Y2, Y4 và Y6 (sơ đồ côngnghệ).

Hai van xung 0.1 và 0.2 là hai van hỗ trợ trong mạch điều khiển bằng khí. Hai van này không phải là các van chính. Vì vậy khi chuyển thành chương trình nó sẽ được thay thế bằng các ô nhớ. Van 0.1 là M0.0, và van 0.2 là M0.1.

Theo sơ đồ mạch điều khiển, ta có:

a1= $M0.0 \,\&\, \overline{M0.1}$

a2 = M0.0 & M0.1

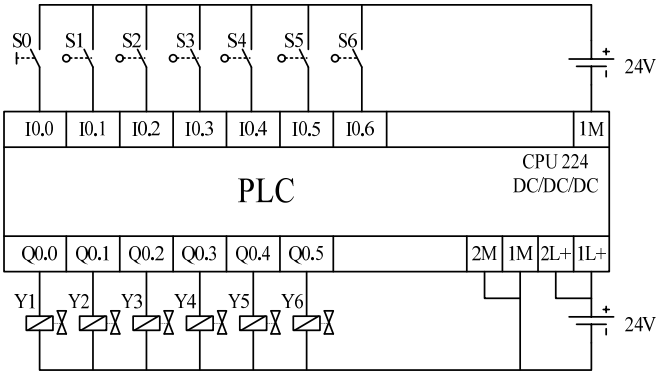
a3= $\overline{M0.0}$

Mỗi vị trí của các xy lanh đều được xác định bởi các công tắc hành trình (CTHT). Xy lanh Cyl.1 nhận biết bởi S1 và S2, xy lanh Cyl.2 nhận biết bởi S3 và S4, xy lanh Cyl.3 nhận biết bởi S5 và S6. Các công tắc hành trình này không thể thiếu trong điều khiển. Ngoài ra để khởi động còn có nút nhấn S0. Như vậy cần đến 7 ngõ vào số.

Bảng ký hiệu

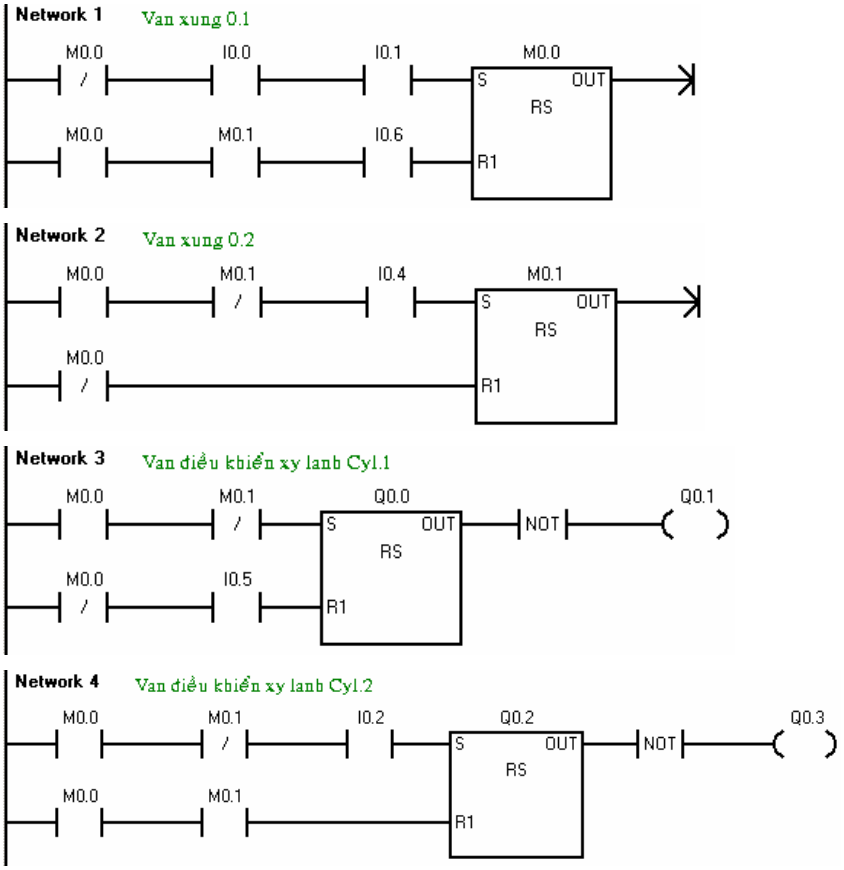
Ký hiệu	***Địa chỉ (PLC)***	***Chú thích***
Biến ngõ vào		
S0	I0.0	Nút nhấn khởi động, thường hở
S1	I0.1	CTHT nhận biết vị trí cơ bản xy lanh Cyl.1
S2	I0.2	CTHT nhận biết vị trí giữ thanh kim loại của xy lanh Cyl.1
S3	I0.3	CTHT nhận biết vị trí cơ bản xy lanh Cyl.2
S4	I0.4	CTHT nhận biết vị trí uốn của xy lanh Cyl.2
S5	I0.5	CTHT nhận biết vị trí cơ bản xy lanh Cyl.3
S6	I0.6	CTHT nhận biết vị trí uốn của xy lanh Cyl.3
Biến ngõ ra		
Y1	Q0.0	Điều khiển xy lanh Cyl.1 để giữ thanh kim loại
Y2	Q0.1	Đưa xy lanh Cyl.1 về vị trí cơ bản
Y3	Q0.2	Điều khiển xy lanh Cyl.2 uốn vuông góc
Y4	Q0.3	Đưa xy lanh Cyl.1 về vị trí cơ bản
Y5	Q0.4	Điều khiển xy lanh Cyl.3 uốn theo khuôn
Y6	Q0.5	Đưa xy lanh Cyl.1 về vị trí cơ bản
Biến trung gian		
Van 0.1	M0.0	Van 0.1
Van 0.2	M0.1	Van 0.2

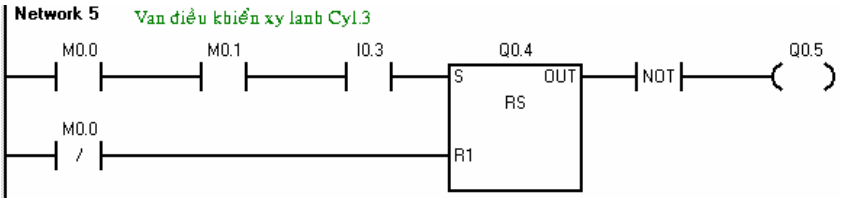
Kết nối dây với PLC:



Hình 13.9: Sơ đồ nối dây ngoại vi với ngõ vào ra của PLC

Chương trình PLC ở LAD:





Chương trình được viết ở STL:

Network 1 Van xung 0.1

```
LDN     M0.0
A       I0.0
A       I0.1
LD      M0.0
A       M0.1
A       I0.6
NOT
LPS
A       M0.0
=       M0.0
LPP
ALD
O       M0.0
=       M0.0
```

Network 2 Van xung 0.2

```
LD      M0.0
AN      M0.1
A       I0.4
LDN     M0.0
NOT
LPS
A       M0.1
=       M0.1
LPP
ALD
O       M0.1
=       M0.1
```

Network 3 Van điều khiển xy lanh Cyl.1

```
LD      M0.0
AN      M0.1
LDN     M0.0
A       I0.5
NOT
LPS
A       Q0.0
=       Q0.0
LPP
ALD
O       Q0.0
=       Q0.0
NOT
=       Q0.1
```

Network 4 Van điều khiển xy lanh Cyl.2

```
LD      M0.0
AN      M0.1
A       I0.2
LD      M0.0
A       M0.1
NOT
LPS
A       Q0.2
=       Q0.2
LPP
ALD
O       Q0.2
=       Q0.2
NOT
=       Q0.3
```

Network 5 Van điều khiển xy lanh Cyl.3

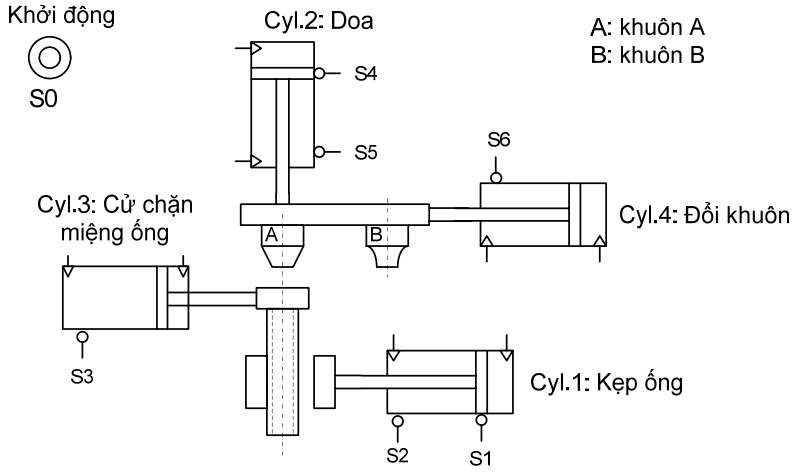
```
LD      M0.0
A       M0.1
A       I0.3
LDN     M0.0
NOT
LPS
A       Q0.4
=       Q0.4
LPP
ALD
O       Q0.4
=       Q0.4
NOT
=       Q0.5
```

13.3.2 Máy doa miệng ống kim loại

Ống kim loại cần được doa miệng theo một khuôn cho trước (sơ đồ công nghệ). Máy hoạt động như sau:

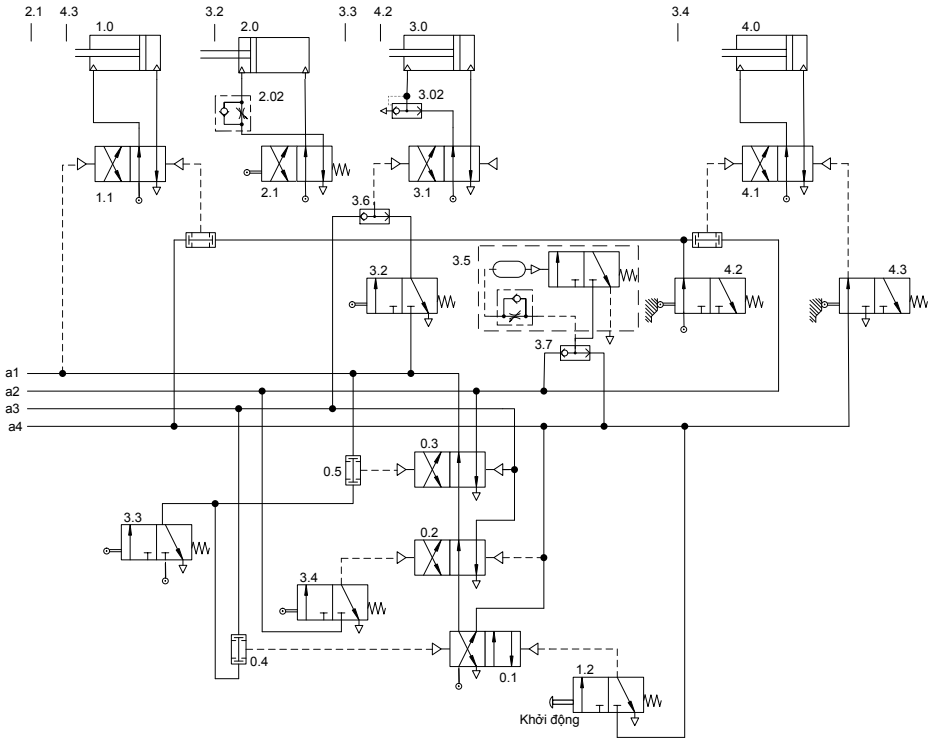
Người vận hành đặt ống kim loại cần doa miệng vào vị trí sao cho miệng ống phải chạm vào cử chặn miệng ống. Sau đó ấn nút nhấn S0, xy lanh Cyl.1 sẽ kẹp ống lại. khi ống đã được kẹp thì cử chặn miệng ống tự động rút về. Xy lanh Cyl.2 sẽ hạ xuống doa miệng ống theo khuôn A. thời gian doa khỏang 3s. Sau đó xy lanh Cyl.2 rút về và khuôn B được xylanh Cyl.4 đưa vào. Sau khi khuôn B được đưa vào thì xy lanh Cyl.2 hạ xuống để doa miệng ống theo khuôn B. Tương tự như khuôn A việc doa khoảng 3s. Sau đó xy lanh Cyl.2 trở về vị trí cơ bản của nó và xy lanh Cyl.4 cũng rút khuôn B về và đặt khuôn A về vị trí sẵn sàng cho ống kim loại kế tiếp. Sau khi miệng ống đã được doa theo khuôn B xong thì xy lanh kẹp ống Cyl.1 co về thả ống kim loại khỏi hàm kẹp. Xy lanh Cyl.2 được đẩy trở về vị trí chặn miệng ống. Một chu kỳ mới lại có thể bắt đầu.

Sơ đồ công nghệ:



Hình 13.10: Sơ đồ công nghệ máy doa miệg ống kim loại.

Sơ đồ mạch điều khiển khí nén:



Hình 13.11: Mạch điều khiển bằng khí nén máy doa miệng ống kim loại.

Phân tích:

Từ sơ đồ điều khiển bằng khí nén ta nhận thấy các van xung chính trong mạch là 1.1, 3.1 và 4.1 sẽ được thay thế bằng các van xung điện từ, và trong chương trình PLC sẽ sử dụng các khau RS. Để điều khiển các van này ta cần 2 ngõ ra

Van 2.1 trong sơ đồ được thay thế bằng van điện từ có lò xo hồi phục vị trí. Để điều khiển van này ta dùng một ngõ ra.

Ba van xung 0.1, 0.2 và 0.3 là các van hỗ trợ trong mạch điều khiển bằng khí. Nó được thay thế bằng các ô nhớ. Van 0.1 là M0.0, van 0.2 là M0.1, và van 0.3 là M0.2.

Theo sơ đồ điều khiển thì:

a1 = $M0.0 \,\&\, \overline{M0.1} \,\&\, \overline{M0.2}$

a2 = $M0.0 \,\&\, \overline{M0.1} \,\&\, M0.2$

a3 = M0.0 & M0.1

a4 = $\overline{M0.0}$

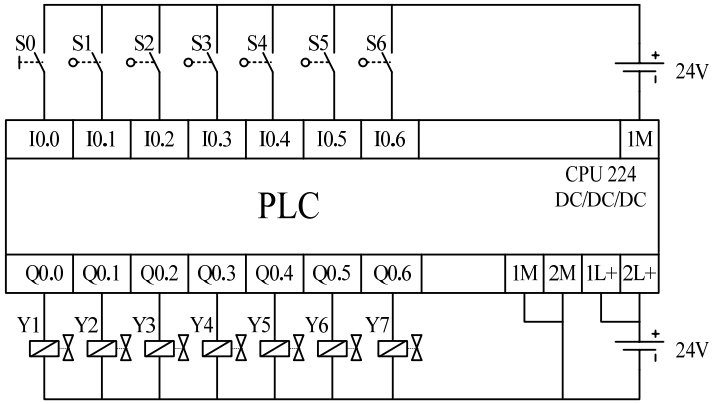
Khâu điều chỉnh trễ 3.5 được thay thế bằng một timer.

Theo sơ đồ công nghệ ta cần đến 6 CTHT và một nút nhấn khởi động từ S0 đến S6 . Như vậy cần đến 7 ngõ vào số.

Bảng ký hiệu

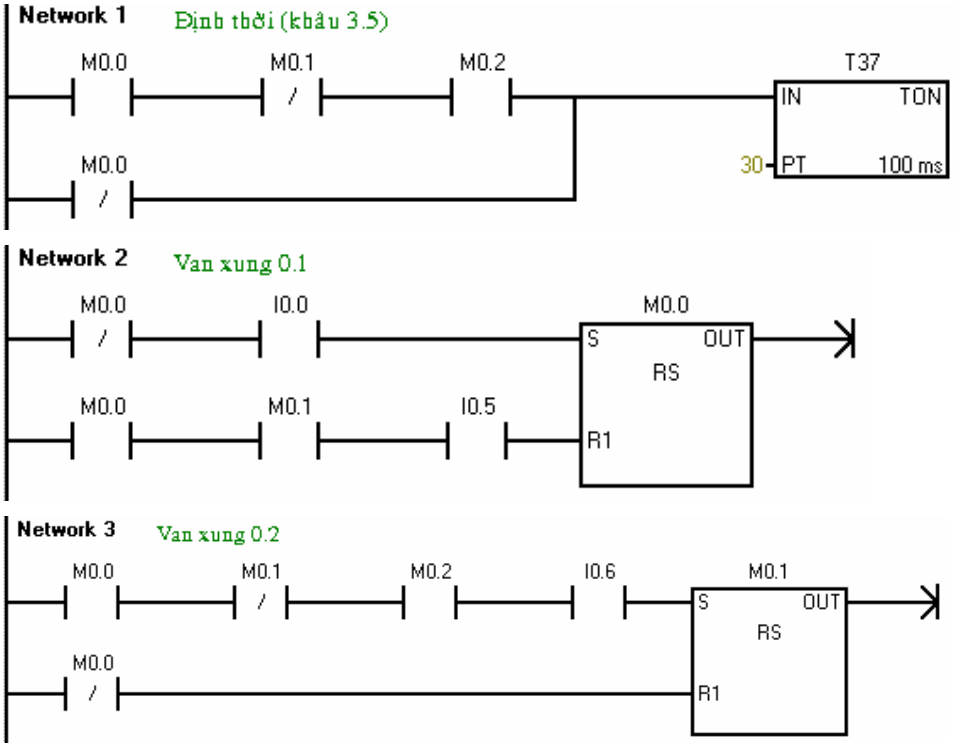
Ký hiệu	***Địa chỉ (PLC)***	***Chú thích***
Biến ngõ vào		
S0	I0.0	Nút nhấn khởi động, thường hở
S1	I0.1	CTHT nhận biết vị trí cơ bản xy lanh Cyl.1
S2	I0.2	CTHT nhận biết vị trí giữ ống kim loại của xy lanh Cyl.1
S3	I0.3	CTHT nhận biết vị trí rút về của xy lanh Cyl.2
S4	I0.4	CTHT nhận biết vị trí rút về của xy lanh Cyl.3
S5	I0.5	CTHT nhận biết vị trí doa của xy lanh Cyl.3
S6	I0.6	CTHT nhận biết vị trí đẩy của xy lanh Cyl.4
Biến ngõ ra		
Y1	Q0.0	Đẩy xy lanh Cyl.1
Y2	Q0.1	Rút xy lanh Cyl.1 về
Y3	Q0.2	Rút xy lanh Cyl.2 về
Y4	Q0.3	Đẩy xy lanh Cyl.3
Y5	Q0.4	Rút xy lanh Cyl.3 về
Y6	Q0.5	Đẩy xy lanh Cyl.4
Y7	Q0.6	Rút xy lanh Cyl.4 về
Biến trung gian		
Van 0.1	M0.0	Van 0.1
Van 0.2	M0.1	Van 0.2
Van 0.3	M0.1	Van 0.3
Bộ định thời		
Delay 3.5	T37	ON delay timer, định thời doa, 3s

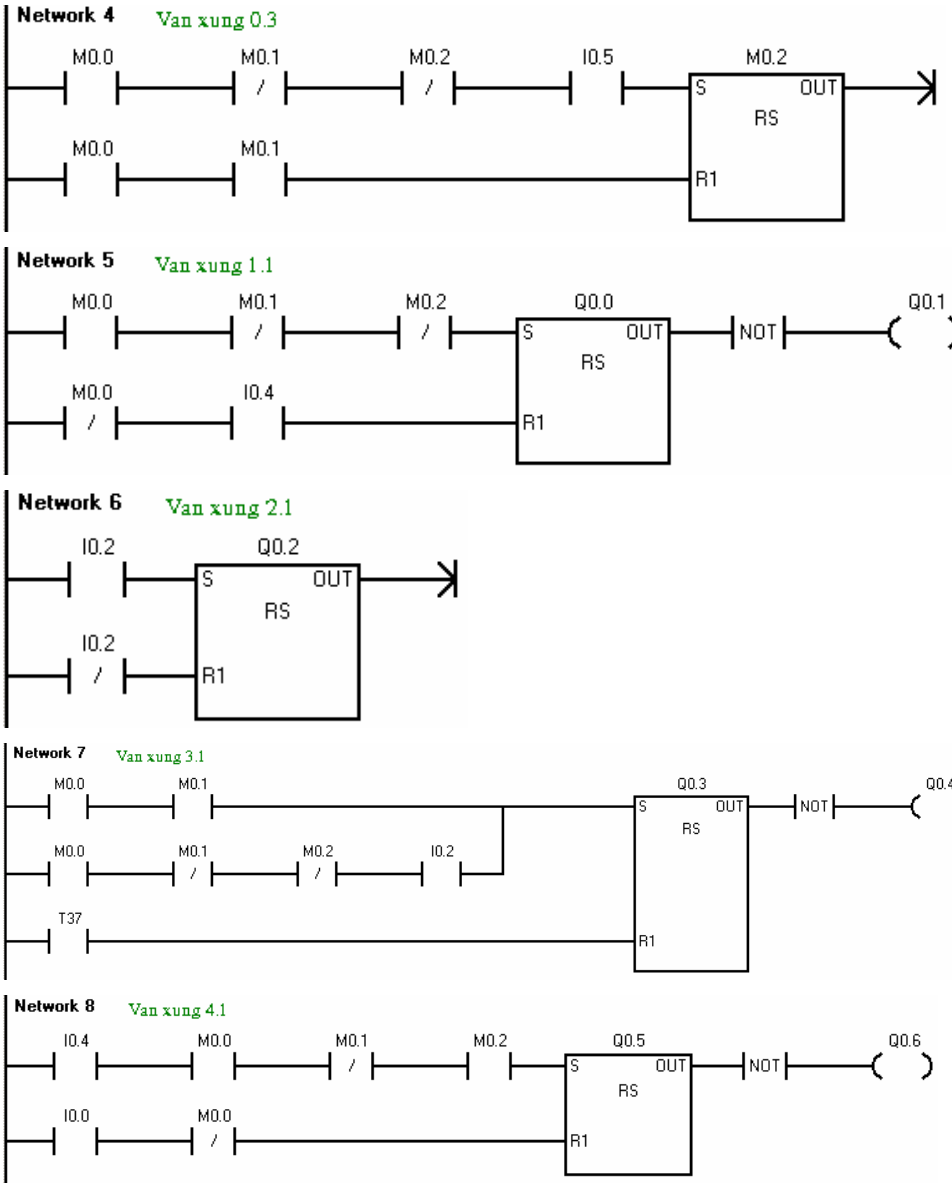
Kết nối dây với PLC:



Hình 13.12: Sơ đồ nối dây ngoại vi với ngõ vào ra của PLC

Chương trình viết ở LAD:





Chương trình viết ở STL:

Network 1 Định thời (khâu 3.5)

```
LD      M0.0
AN      M0.1
A       M0.2
ON      M0.0
TON     T37, 30
```

Network 2 Van xung 0.1

```
LDN     M0.0
A       I0.0
LD      M0.0
A       M0.1
A       I0.5
NOT
LPS
A       M0.0
=       M0.0
LPP
ALD
O       M0.0
=       M0.0
```

Network 3 Van xung 0.2

```
LD      M0.0
AN      M0.1
A       M0.2
A       I0.6
LDN     M0.0
NOT
LPS
A       M0.1
=       M0.1
LPP
ALD
O       M0.1
=       M0.1
```

Network 4 Van xung 0.3

```
LD      M0.0
AN      M0.1
AN      M0.2
A       I0.5
LD      M0.0
A       M0.1
NOT
LPS
A       M0.2
=       M0.2
LPP
ALD
O       M0.2
=       M0.2
```

Network 5 Van xung 1.1

```
LD      M0.0
AN      M0.1
AN      M0.2
LDN     M0.0
A       I0.4
NOT
LPS
A       Q0.0
=       Q0.0
LPP
ALD
O       Q0.0
=       Q0.0
NOT
=       Q0.1
```

Network 6 Van xung 2.1

```
LD      I0.2
LDN     I0.2
NOT
LPS
A       Q0.2
=       Q0.2
LPP
ALD
O       Q0.2
=       Q0.2
```

Network 7 Van xung 3.1

```
LD      M0.0
A       M0.1
LD      M0.0
AN      M0.1
AN      M0.2
A       I0.2
OLD
LD      T37
NOT
LPS
A       Q0.3
=       Q0.3
LPP
ALD
O       Q0.3
=       Q0.3
NOT
=       Q0.4
```

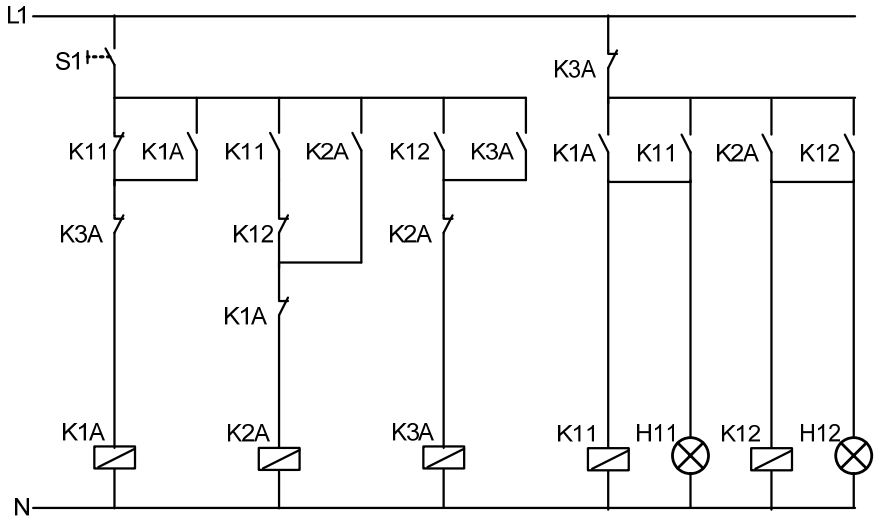
```
Network 8      Van xung 4.1
LD      I0.4
A       M0.0
AN      M0.1
A       M0.2
LD      I0.0
AN      M0.0
NOT
LPS
A       Q0.5
=       Q0.5
LPP
ALD
O       Q0.5
=       Q0.5
NOT
=       Q0.6
```

13.4 Câu hỏi và bài tập

BT 13.1 Điều khiển lò nhiệt bằng nút nhấn

Hai lò nhiệt cần điều khiển bằng một nút nhấn. Ở lần nhấn đầu tiên, thì lò nhiên thứ nhất hoạt động. Ở lần nhấn thứ hai thì lò nhiệt thứ hai được đưa vào hoạt động. Và ở lần nhấn thứ ba thì cả hai lò nhiệt cùng tắt. Các lò nhiệt được cung cấp điện thông qua các contactor K11 và K12. Ngoài ra các đèn tín hiệu H11 và H12 dùng để báo lò nhiệt tương ứng đang hoạt động.

Mạch điều khiển:



Hãy chuyển sang điều khiển sử dụng PLC theo các yêu cầu sau:

1. Thiết lập bảng ký hiệu.
2. Vẽ sơ đồ kết nối dây với PLC
3. Viết chương trình điều khiển theo hai cách:
    a. Sơ đồ kết nối dây cứng
    b. Theo yêu cầu công nghệ

BT 13.2 Điều khiển đèn quảng cáo

Đèn quảng cáo cần được điều khiển như sau:

Đóng công tắc S1.

Sau 10s đèn E1 sáng

Sau 20s đèn E2 sáng

Sau 30s đèn E3 sáng

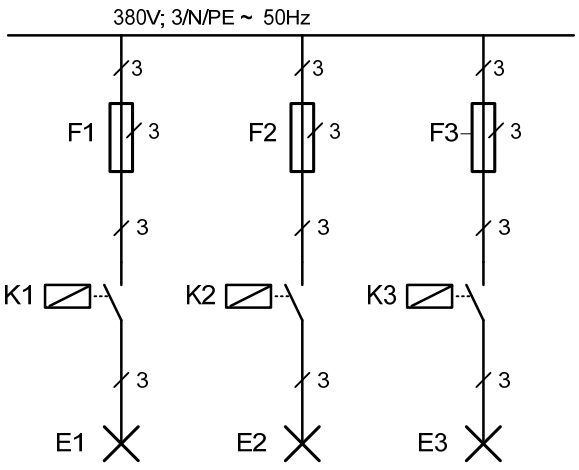
Sau 40s tất cả các đèn đều tắt

Sau đó bắt đầu tự động lại chu kỳ mới

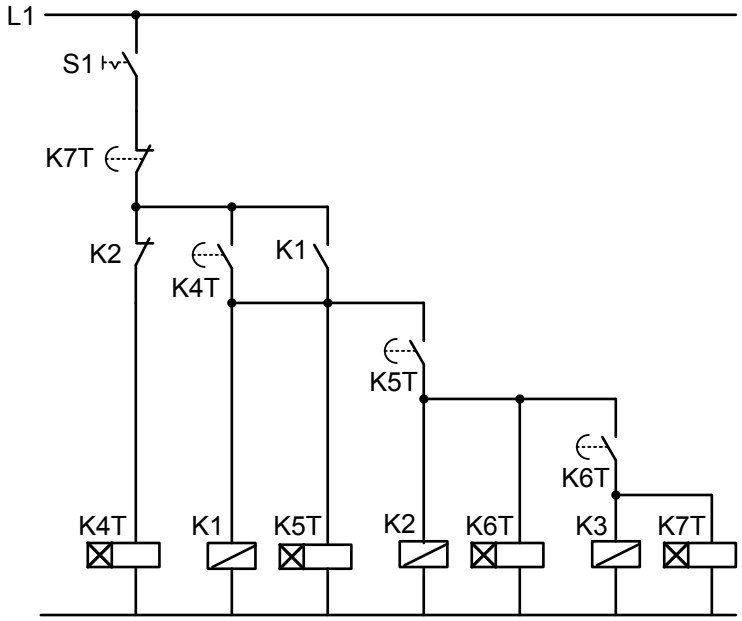
Hãy chuyển sang điều khiển sử dụng PLC theo các yêu cầu sau:

1. Thiết lập bảng ký hiệu.
2. Vẽ sơ đồ kết nối dây với PLC
3. Viết chương trình điều khiển theo hai cách:
    a. Sơ đồ kết nối dây cứng
    b. Theo yêu cầu công nghệ

Sơ đồ mạch động lực:



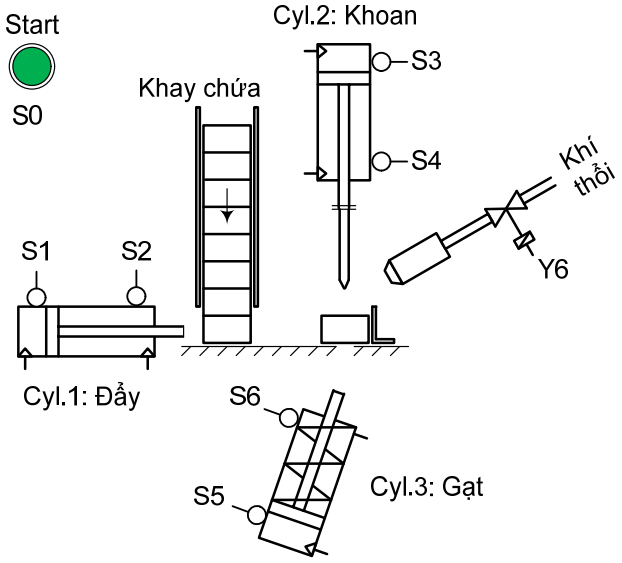
Sơ đồ mạch điều khiển:



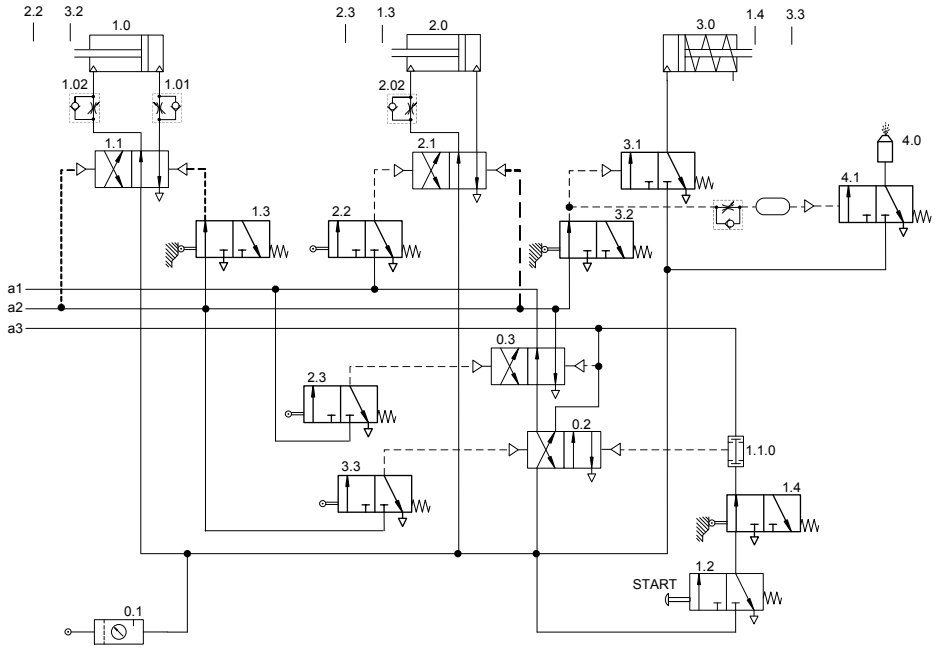
BT 13.3 Máy khoan

Một mẫu gỗ cần được khoan một lỗ ở giữa. Sơ đồ công nghệ để khoan mẫu gỗ được cho như hình vẽ.

Sơ đồ công nghệ:



Sơ đồ điều khiển bằng khí nén:



Hãy chuyển sang điều khiển sử dụng PLC theo các yêu cầu sau:

1. Thiết lập bảng ký hiệu.
2. Vẽ sơ đồ kết nối dây với PLC
3. Viết chương trình điều khiển theo hai cách:
    a. Sơ đồ kết nối dây cứng
    b. Theo yêu cầu công nghệ

14 Các phép toán cơ bản trong điều khiển số

Các hệ thống điều khiển logic trong thực tế xử lý với các dữ liệu nhị phân. Đặc điểm của các máy tính điều khiển hiện nay là xử lý dữ liệu, chất lượng điều khiển, v.v... ngày càng tăng với bộ xử lý dữ liệu số sử dụng PLC.

Các biến quá trình số có thể được tìm thấy trong tất cả lĩnh vực của điều vòng hở như trong các thiết bị được kết nối cho hoạt động quá trình và giám sát hoặc trong điều khiển của các thiết bị trường. Mục đích của giám sát quá trình là cung cấp thông tin về máy móc hoặc hệ thống hoạt động nhanh chóng, ngắn gọn và rõ ràng theo từng phút, cũng như sự đúng lúc để can thiệp, điều khiển và tác động đến quá trình.

Trong hầu hết các điều khiển đơn giản trước đây, các thiết bị vào ra như màn hiển thị 7-đoạn và các nút nhấn xoay số được sử dụng để hiển thị và nhập giá trị số. Ngày nay các thiết bị thao tác và giám sát „thông minh“ thường được kết nối với PLC.

Ngày nay các thiết bị xử lý, thu thập dữ liệu và điều khiển quá trình được cung cấp trực tiếp với các biến số thông qua hệ thống bus trường. Việc kết nối các thiết bị trường, như biến tần hay hệ thống cân, sử dụng các module vào ra analog càng ngày càng không được sử dụng nữa.

Tuỳ thuộc vào kiểu thiết bị được kết nối, nhiều dạng số khác nhau để mã hóa dữ liệu được sử dụng để truyền dữ liệu giữa thiết bị và PLC, cũng như để lưu trữ và xử lý dữ liệu trong PLC.

14.1 Các dạng số trong PLC

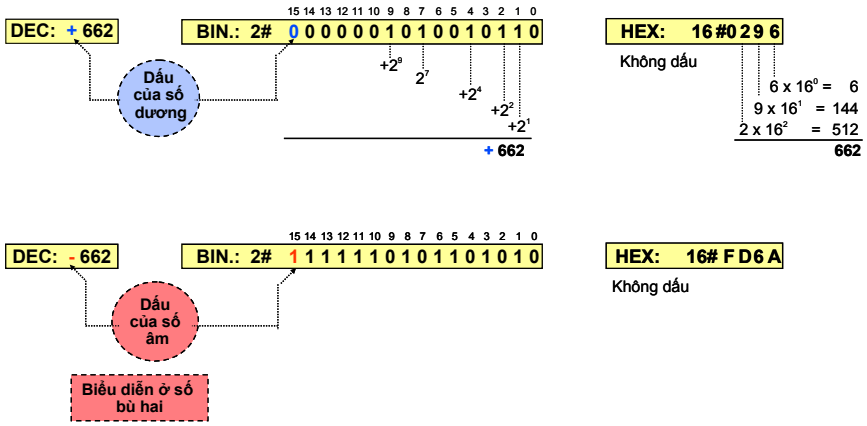
14.1.1 Kiểu dữ liệu Integer (INT)

Giá trị kiểu dữ liệu *Integer* hoàn toàn là giá trị số không có dấu chấm thập phân. S7-200 lưu trữ giá trị dữ liệu kiểu *Integer* có dấu ở mã 16 bit. Phạm vi của số integer là -32768 đến +32767.

STEP 7 sử dụng dạng hiển thị *Decimal* (không phải BCD) để xác định các hằng số của kiểu dữ liệu *Integer*. Nó cũng được mô tả ở dạng có dấu và không dấu. Theo nguyên lý thì có thể sử dụng các giá trị integer hằng số biểu

diễn ở dạng *Binary* và *Hexadecimal*, nhưng vì không rõ ràng, nên chúng không còn phù hợp nữa. Vì lý do này, cú pháp của STEP7 chỉ cung cấp giá trị của integer biểu diễn ở decimal.

Ví dụ: Biểu diễn số +662 và -662



Hình 14.1: Biểu diễn số integer

Trong hệ thống máy tính số, tất cả các giá trị được lưu trữ ở dạng mã binary. Chỉ các số 0 và 1 được sử dụng trong hệ thống số nhị phân. Cơ số 2 của hệ thống số này là kết quả từ số của các số có giá trị. Giá trị của mỗi vị trí của số nhị phân là kết quả của lũy thừa của cơ số 2. Nó được biểu diễn ở dạng 2#.... .

Giá trị số âm là sự biểu diễn các số nhị phân ở dạng bù hai. Trong dạng biểu diễn này, bit có trọng số lớn nhất (most significant bit) (bit số 15 cho kiểu dữ liệu Integer) có giá trị - 2^{15}. Vì giá trị này lớn hơn tổng của tất cả các giá trị còn lại, nên bit này được làm bit thông tin dấu. Nếu bit = 0, thì giá trị dương; nếu bit = 1, thì giá trị là âm. Việc chuyển đổi giữa các số nhị phân thành số decimal được thực hiện bằng cách cộng các giá trị của các vị trí có bit = 1. (xem ví dụ).

Hệ thống số hexadecimal cung cấp 16 chữ số khác nhau (0 đến 9 và A đến F). Đây là hệ thống số theo cơ số 16. Do đó, giá trị mỗi vị trí của số hexadecimal có kết quả từ lũy thừa của cơ số 16.

Các số Hexadecimal được xác định với dạng 16#. Các chữ số A đến F biểu diễn theo giá trị số decimal 10 đến 15. Giá trị 15 là giá trị cuối cùng có thể được mã hóa nhị phân của 4 bit không dấu. 4 bit nhị phân tạo thành một số của số hexadecimal.

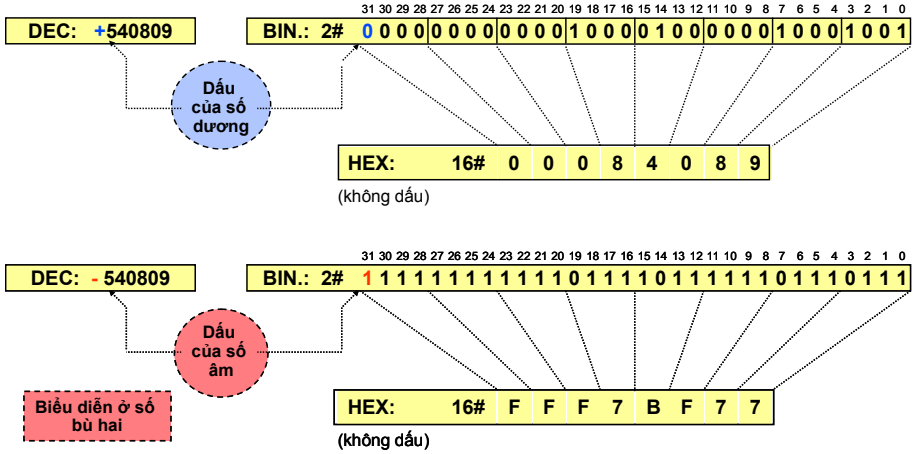
Hằng số trong dạng số Hexadecimal không được sử dụng cho các giá trị số integer.

14.1.2 Kiểu dữ liệu Double Integer (DINT)

S7-200 lưu giá trị kiểu dữ liệu *Double Integer* với mã 32 bit có dấu. Phạm vi giá trị kiểu double Integer từ -2147483648 đến +2147483647.

S7-200 sử dụng số decimal (không phải BCD) để xác định một hằng số kiểu dữ liệu *Double Integer*.

Ví dụ: Biểu diễn số +540809 và – 540809



Hình 14.2: Biểu diễn số double integer

14.1.3 Kiểu dữ liệu số thực (REAL)

Các kiểu dữ liệu INT và DINT được mô tả trước được sử dụng để lưu toàn bộ các giá trị số có dấu. Do đó, chỉ có các phép toán được cung cấp các giá trị số nguyên mới có thể thực hiện được.

Trong trường hợp các biến là analog như điện áp, dòng điện, và nhiệt độ thì các giá trị thực trở nên cần thiết. Để trình diễn các giá trị thập phân, các số nhị phân phải được định nghĩa là giá trị của nó nhỏ hơn 1 (lũy thừa của cơ số 2 với số mũ âm).

Để biểu diễn số thực S7-200 sử dụng double word (32 bit).Trong mã nhị phân của số thực, một phần của các chữ số nhị phân sử dụng cho phần thập phân, phần còn lại là để biểu diễn số mũ và dấu của số thực.

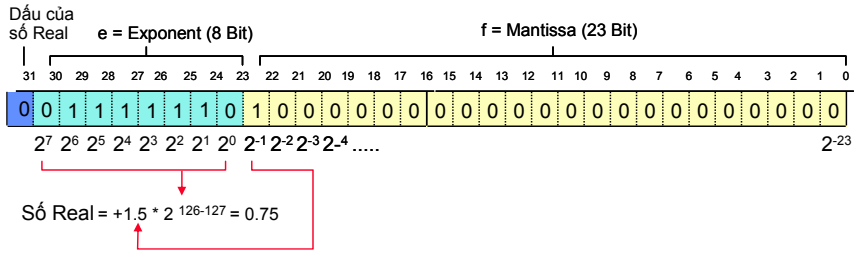
Phạm vi biểu diễn của số thực từ $-1.175495 \cdot 10^{-38}$ đến $3.402823 \cdot 10^{+38}$

Khi sử dụng các giá trị của số thực, ta không cần phải xác định định dạng của nó. Khi nhập vào một hằng số là số thực thì ta bắt buộc phải nhập có thành phần thập phân cho dù phần thập là số 0, ví dụ 20.0.

Số thực được sử dụng để „xử lý giá trị analog„. Ưu điểm lớn của số thực là các phép toán được sử dụng với nó. Các phép toán này bao gồm: cộng, trừ, nhân, chia cũng như các lệnh sin, cos, exp, ln, v.v..., được sử dụng chính trong các thuật giải điều khiển vòng kín (closed-loop control algorithms).

Dang tổng quát của số Real = (dấu) • (1.f) • (2^{e-127}) với f: phần thập phân.

Ví dụ: Biểu diễn số 0.75



Hình 14.2: Biểu diễn số real

14.1.4 Kiểu dữ liệu số BCD (Binary Coded Decimal)

Trước đây, để liệt kê và mô tả các số nguyên được thực hiện đơn giản với các nút nhấn số dạng xoay vòng và bộ chỉ thị số. Các nút nhấn số và hiển thị số này được kết nối với các module vào và ra số của PLC.

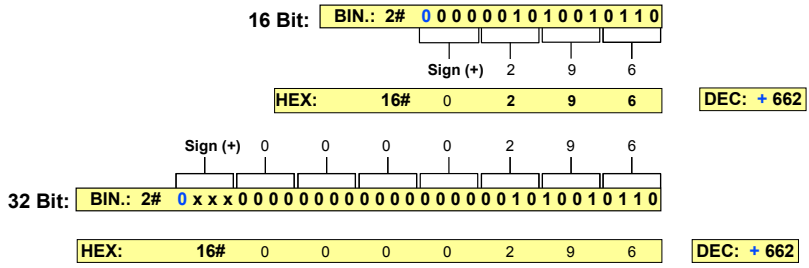
Mỗi chữ số của số decimal được mã hóa ở bốn bit. Vì chữ số cao nhất của decimmal là 9 nên bốn bit được sử dụng và có mã nhị phân tương ứng cho các chữ số decimal như sau:

Số Decimal	*BCD Code*	*Số Decimal*	*BCD Code*
0	0000	6	0110
1	0001	7	0111
2	0010	8	1000
3	0011	9	1001
4	0100	10 ... 15	không có
5	0101		

Để các số âm cũng có thể được xác định bằng nút nhấn số xoay vòng mã BCD, thì S7-200 mã hóa dấu trong bit có trọng số cao nhất (most significant bit). Bit dấu = 0 để chỉ số dương. Bit dấu = 1 chỉ thị số âm. S7-200 chấp nhận các số BCD mã 16-bit (dấu + 3 digits) và mã 32-bit (dấu + 7 digits). Phạm vi biểu diễn của số BCD 16 bit từ - 999 đến + 999, phạm vi biểu diễn của số BCD 32 bit từ -9999999 đến + 9999999.

Không có định dạng dữ liệu cho việc xác định các giá trị theo mã BCD trong S7-200. Tuy nhiên ta có thể xác định số decimal với mã BCD được cho ở số HEX. Mã nhị phân của số HEX và số decimal mã BCD thì giống nhau.

Ví dụ: Biểu diễn số 662 ở BCD 16 bit và BCD 32 bit



Hình 14.4: Biểu diễn số BCD 16 bit và BCD 32 bit

14.2 Chức năng sao chép

Với chức năng sao chép, nội dung của một vùng này sẽ được sao chép đến một vùng khác trong bộ nhớ. Việc trao đổi hay sao chép nội dung có thể thực hiện với một byte, một word, một double word hay một giá trị số hoặc một mảng lớn dữ liệu từ vùng này sang vùng khác trong bộ nhớ.

14.2.1 Các lệnh sao chép, trao đổi nội dung

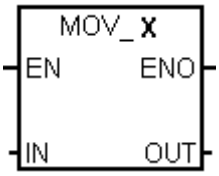
Để sao chép các dữ liệu kiểu byte, word, double word kể cả số thực (real) từ nơi này đến nơi khác ta sử dụng lệnh Move.

Trong một số trường hợp cần tráo đổi nội dung của một byte (byte thấp và byte cao) trong một word ta sử dụng lệnh Swap.

Cú pháp của các lệnh ở STL như sau:

- ***Lệnh MOVB IN,OUT:*** Lệnh Move Byte (MOVB) thực hiện sao chép nội dung của byte IN sang byte OUT.
- ***Lệnh MOVW IN,OUT:*** Lệnh Move Word (MOVW) thực hiện sao chép nội dung của word IN sang word OUT
- ***Lệnh MOVD IN,OUT:*** Lệnh Move Double Word (MOVD) thực hiện sao chép nội dung của double word IN sang double word OUT.
- ***Lệnh MOVR IN,OUT:*** Lệnh Move Real (MOVR) thực hiện sao chép nội dung của một số thực IN sang số thực OUT.
- ***Lệnh SWAP IN:*** Lệnh Swap Byte (Swap) thực hiện tráo đổi nội dung của byte thấp và byte cao trong word IN.

Cú pháp của các lệnh MOVE ở LAD và FBD có cấu trúc chung như sau:



Với:

* **X:** Có thể là B (Byte), W (Word), D (Double word) hoặc R(Real).

* IN: Dữ liệu cần sao chép, có thể là byte, word, double word hoặc real tùy theo **X** là B, W, D hay R.

* OUT: Vị trí của nơi cần sao chép đến, có thể là byte, word, double word hoặc real tùy theo **X** là B, W, D hay R.

* EN: Là ngõ vào bit. Cho phép thực hiện lệnh được viết ở LAD hoặc FBD.

  Trường hợp không cần thiết có điều kiện ở ngõ vào EN thì phải sử dụng SM0.0.

* ENO: Ngõ ra bit. Cho phép kết nối song song hoặc nối tiếp với các hộp khác. Nếu phép toán xử lý không có lỗi thì EN=ENO.

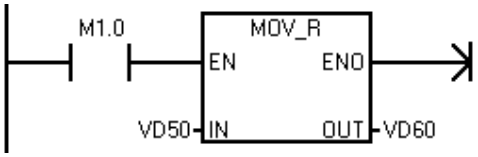
Để lấy lệnh MOV ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Move trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là:

MOVB : sao chép Byte
MOVD : sao chép double Word
MOVW : sao chép Word
MOVR : sao chép số thực

giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0, Byte có nội dung cần sao chép đặt ở ngõ IN và byte chứa đựng thông tin sao chép chứa ở OUT.

Ví dụ: Copy ô nhớ số thực ở VD50 vào ô nhớ số thực VD60 khi M1.0 tích cực. Chương trình được viết như sau:

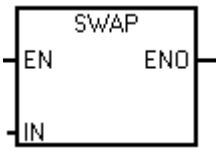
LAD:



STL:

```
LD      M1.0
MOVR    VD50, VD60
```

**** Cú pháp dùng lệnh SWAP trong LAD như sau:***

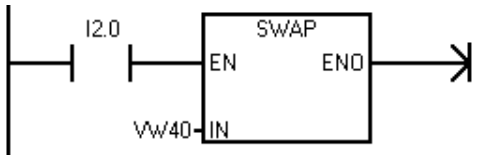
LAD	Toán hạng
	***IN (Word):*** VW, IW, QW, MW, SW, SMW, T, C, LW, AC, *VD, *AC, *LD

Để lấy lệnh SWAP ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Move trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là: SWAP, giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit

nhớ SM0.0, word cần tráo đổi nội dung giữa byte thấp và byte cao đặt ở ngõ IN.

Ví dụ: Ô nhớ VW40 có giá trị được biểu diễn ở số Hex là CAFE. Giá trị này sẽ được đảo lại thành FECA khi ngõ vào I2.0 được kích hoạt. Chương trình được viết như sau:

LAD:



STL:

```
LD      I2.0
SWAP    VW40
```

14.2.2 Các lệnh sao chép một mảng lớn dữ liệu

Để sao chép một mảng lớn dữ liệu từ nơi này đến nơi khác ta sử dụng lệnh Block Move. Lệnh sao chép một mảng lớn cho phép thực hiện với Byte, Word và Double Word.

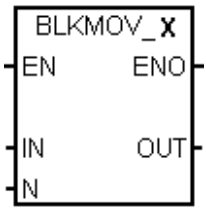
Cú pháp của các lệnh ở STL như sau:

**** Lệnh BMB IN,OUT,N***: Lệnh Block Move Byte (BMB) sao chép nội dung của một mảng Byte. Số lượng byte được sao chép xác định bởi N có kiểu byte. Do đó có thể sao chép tối đa là 255 byte. Byte đầu tiên của mảng được xác định ở ngõ IN (kiểu byte). Nơi đến được xác định với byte đầu tiên của mảng ở ngõ OUT.

**** Lệnh BMW IN,OUT,N***: Tương tự như lệnh BMB, lệnh Block Move Word (BMW) sao chép nội dung của một mảng word. Số lượng word được sao chép xác định bởi N có kiểu byte. Do đó có thể sao chép tối đa là 255 word. Word đầu tiên của mảng được xác định ở ngõ IN (kiểu word). Nơi đến được xác định với word đầu tiên của mảng ở ngõ OUT.

**** Lệnh BMD IN,OUT,N***: Tương tự như lệnh BMB, lệnh Block Move Double Word (BMD) sao chép nội dung của một mảng Double Word. Số lượng Double word được sao chép xác định bởi N có kiểu byte. Do đó có thể sao chép tối đa là 255 Double word. Double Word đầu tiên của mảng được xác định ở ngõ IN (kiểu Double word). Nơi đến được xác định với Double word đầu tiên của mảng ở ngõ OUT.

Cú pháp của các lệnh ở LAD và FBD có cấu trúc tổng quát như sau:



Với:

* **X**: Có thể là B (Byte), W (Word), D (Double word).

* IN: Vị trí đầu tiên của mảng dữ liệu cần sao chép, có thể là Byte, Word hoặc double Word tùy theo **X**.

* OUT: Vị trí đầu tiên của mảng dữ liệu cần lưu trữ thông tin sao chép. có thể là Byte, Word hoặc double Word tùy theo **X**.

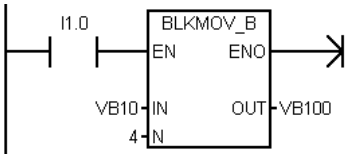
* N: Số lượng Byte, Word, Double word được sao chép, có giá trị từ 0 đến 255.

* EN, ENO: tương tự như ở lệnh MOVE.

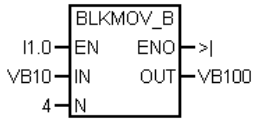
Để lấy lệnh BLKMOV ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Move trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là: BLKMOV_B (sao chép mảng Byte), BLKMOV_W (sao chép mảng Word), BLKMOV_D (sao chép mảng double Word), giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0; Byte, word hoặc double word (tùy theo lệnh) đầu tiên của mảng cần sao chép đặt ở ngõ IN và số lượng tương ứng được đặt vào chân N.

Ví dụ: Khi kích hoạt I1.0 thì nội dung của một mảng gồm 4 byte bắt đầu từ Byte VB10 sẽ được copy sang vùng nhớ gồm có 4 byte khác có byte đầu tiên là VB100. Chương trình được viết như dưới đây:

LAD:



FBD:



STL:

```
LD    I1.0
BMB   VB10, VB100, 4
```

Giả thiết nội dung của mảng cần sao chép là:

Byte	*VB10*	*VB11*	*VB12*	*VB13*
Nội dung	20	21	22	23

Kết quả thu dược sau lệnh: BMB VB10, VB100, 4 là:

Byte	*VB100*	*VB101*	*VB102*	*VB103*
Nội dung	20	21	22	23

14.3 Phép toán so sánh

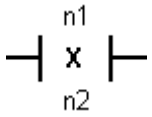
Với chức năng so sánh, giá trị của hai toán hạng của cùng kiểu dữ liệu sẽ được so sánh với nhau. Kết quả của so sánh là một giá trị logic, nếu đúng theo chức năng so sánh thì kết quả logic là “1”, còn nếu sai kết quả logic là

"0". Tùy thuộc vào loại CPU của họ S7-200 mà có thể có ít hoặc nhiều chức năng so sánh.Các chức năng so sánh đối CPU 22x có thể là:

Toán hạng 1 (IN1)	*Chức năng so sánh*	*Toán hạng 2 (IN2)*
Dữ liệu có thể là: ***Byte, Int,DInt, Real***	**> : Lớn hơn** **>= : Lớn hơn hoặc bằng** **== : Bằng nhau** **<> : Không bằng nhau (khác nhau)** **<= : Bé hơn hoặc bằng** **< : Bé hơn**	**Dữ liệu có thể là:** ***Byte, Int,DInt, Real***

Khi so sánh giá trị Byte (B) thì không cần phải để ý đến dấu của toán hạng, ngược lại khi so sánh là các số Int (I), Dint (D), Real (R) thì phải chú ý đến dấu của tóan hạng.

Cú pháp tổng quát cho phép tóan so sánh ở LAD là:



Với:

X: là phép so sánh. Nó có thể là:

+ So sánh byte: >B, >=B, ==B, <>B, <B, <=B
+ So sánh số Int: >I, >=I, ==I, <>I, <I, <=I
+ So sánh số Dint: >D, >=D, ==D, <>D, <D, <=D
+ So sánh số Real: >R, >=R, ==R, <>R, <R, <=R
+ n1: Giá trị cần được so sánh (giá trị chưa biết).
+ n2: Giá trị so sánh (giá trị đã biết).

Đối với ngôn ngữ LAD và FBD, khi kết quả so sánh là đúng, thì lệnh so sánh sẽ đặt tiếp điểm (LAD) hoặc ngõ ra (FBD) ở trạng thái "ON".

Đối với ngôn ngữ STL, khi kết quả so sánh là đúng thì lệnh so sánh Load, AND, hoặc OR giá trị 1 với giá trị ở đỉnh của ngăn xếp.

Để lấy các ***lệnh so sánh*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Compare trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy, giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập giá trị chưa biết theo lệnh cần so sánh (byte, word, double word) vào vị trí các dấu chấm hỏi nằm trên lệnh. Nhập giá trị đã biết (thường là các con số) hoặc giá trị được chứa trong các ô nhớ byte, word, double word vào vị trí các dấu chấm hỏi nằm dưới lệnh.

Ví dụ 14.2: *Giới hạn giá trị.*

Viết một chương trình thực hiện nhiệm vụ sau: Nếu giá trị ở MW20 nằm trong phạm vi (50;500) thì sẽ cho phép xuất giá trị ra ở ngõ ra MW22. Nếu giá trị ở MW20 lớn hơn giá trị 500 thì ngõ ra số MW22 là giá trị 500 và đèn báo giá trị max sáng. Nếu giá trị ở MW20 nhỏ hơn giá trị 50 thì ngõ ra số MW22 là giá trị 50 và đèn báo giá trị min sáng. Chú ý các ngõ vào ra số là Int.

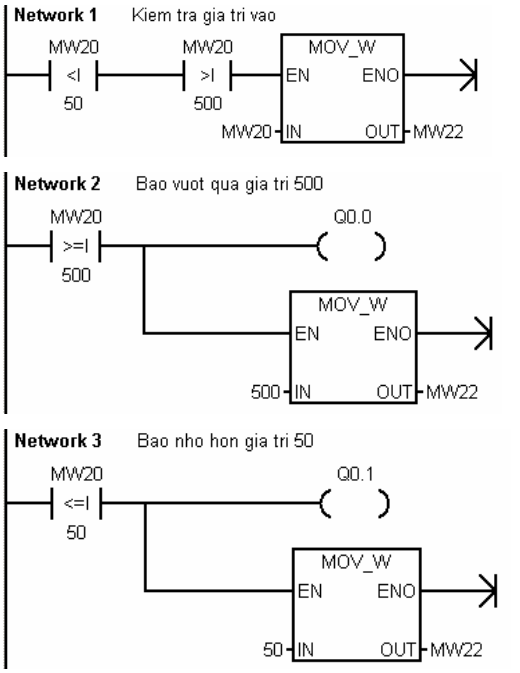
Giải:

Bảng ký hiệu

Ký hiệu	Địa chỉ	Chú thích
GT_sosanh	MW20	Giá trị số cần biết có vượt ngoài phạm vi (50;500)
GT_dung	MW22	Giá trị nằm trong phạm vi cho phép
Bao_max	Q0.0	Đèn báo giá trị lớn hơn 500
Bao_min	Q0.1	Đèn báo giá trị nhỏ hơn 50

Chương trình:

LAD



STL

```
Network 1  Kiem tra gia tri vao
LDW<   MW20, 50
AW>    MW20, 500
MOVW   MW20, MW22

Network 2  Bao vuot qua gia tri 500
LDW>=  MW20, 500
=      Q0.0
MOVW   500, MW22

Network 3  Bao nho hon gia tri 50
LDW<=  MW20, 50
=      Q0.1
MOVW   50, MW22
```

14.4 Phép toán số học

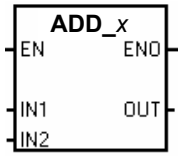
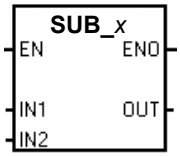
Ở nhiều nhiệm vụ đếm như đếm sản phẩm, đếm số vòng quay, đếm xung .v.v... thì kết quả đếm phải được giám sát. Bên cạnh các phép tóan so sánh đã biết cần phải có thêm các phép toán số học như cộng, trừ, nhân, chia. Còn các phép toán khác như sin, cos, tan, PID .... sẽ được khảo sát ở *tập 2* của bộ sách *kỹ thuật điều khiển lập trình PLC SIMATIC S7-200*.

14.4.1. Cộng và trừ

Các phép tóan cộng và trừ có thể thực hiện được đối với các số Integer (16 bit), Double integer (32 bit) và số thực (32 bit). Tùy thuộc vào phép tóan là cộng hoặc trừ dạng số nào mà kết quả thu được sẽ ở dạng số đó.

Khi có lỗi do tràn hoặc giá trị không hợp lệ thì bit SM1.1 được set lên mức logic „1“.

Cú pháp lệnh biểu diễn cho phép toán cộng và trừ như sau:

Phép toán cộng	***Phép toán trừ***	***Chú thích***
Biểu diễn ở LAD:  *Thực hiện:* IN1 + IN2 = OUT	Biểu diễn ở LAD:  *Thực hiện:* IN1 - IN2 = OUT	* *x*: có thể là I (Integer), DI (Double integer), R(Real). * EN = “1”: cho phép cộng hoặc trừ. * ENO = “0”: khi có lỗi. * IN1, IN2, OUT: các ngõ vào ra dạng số có cùng kiểu dữ liệu với x.
Biểu diễn ở STL: +I IN1, OUT +D IN1, OUT +R IN1, OUT *Thực hiện:* IN1 + OUT = OUT	Biểu diễn ở STL: -I IN1, OUT -D IN1, OUT -R IN1, OUT *Thực hiện:* OUT – IN1 = OUT	

Để lấy lệnh ***cộng hoặc trừ số nguyên*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Integer Math trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là: ADD_I (cộng số Integer), ADD_DI (cộng số DInt), SUB_I (trừ số Integer), hoặc SUB_DI (trừ số DInt), giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến của phép toán tương ứng vào các ngõ IN1 và IN2. Nhập biến chứa kết quả ở ngõ OUT.

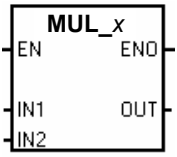
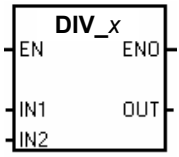
Để lấy lệnh ***cộng hoặc trừ số thực (real)*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Floating-Point Math trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là: ADD_R (cộng số real), SUB_R (trừ số real), giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến của phép toán tương ứng vào các ngõ IN1 và IN2. Nhập biến chứa kết quả ở ngõ OUT.

14.4.2. Nhân và chia

Các phép tóan nhân và chia có thể thực hiện được đối với các số Integer (16 bit), Double integer (32 bit) và số thực (32 bit). Tùy thuộc vào phép tóan là nhân hoặc chia dạng số nào mà kết quả thu được sẽ ở dạng số đó.

Khi có lỗi do tràn hoặc giá trị không hợp lệ thì bit SM1.1 được set lên mức logic „1“. Nếu kết quả là zero thì SM1.0 =”1”, kết quả âm thì SM1.2 =”1”, và SM1.3 =”1” nếu chia cho 0.

Cú pháp lệnh biểu diễn cho phép toán nhân và chia như sau:

Phép toán nhân	***Phép toán chia***	***Chú thích***
Biểu diễn ở LAD:  *Thực hiện:* IN1 * IN2 = OUT	Biểu diễn ở LAD:  *Thực hiện:* IN1 / IN2 = OUT	* *x*: có thể là I (Integer), DI (Double integer), R(Real). * EN = “1”: cho phép nhân hoặc chia. * ENO = “0”: khi có lỗi. * IN1, IN2, OUT: các ngõ vào ra dạng số có cùng kiểu dữ liệu với x.
Biểu diễn ở STL: *I IN1, OUT *D IN1, OUT *R IN1, OUT *Thực hiện:* IN1 * OUT = OUT	Biểu diễn ở STL: /I IN1, OUT /D IN1, OUT /R IN1, OUT *Thực hiện:* OUT / IN1 = OUT	

Để lấy lệnh ***nhân hoặc chia số nguyên*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Integer Math trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là: MUL_I (nhân số Integer), MUL_DI (nhân số DInt), DIV_I (chia số Integer), hoặc DIV_DI (chia số DInt), giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến của phép toán tương ứng vào các ngõ IN1 và IN2. Nhập biến chứa kết quả ở ngõ OUT.

Để lấy lệnh ***nhân hoặc chia số thực (real)*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Floating-Point Math trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là: MUL_R (nhân số real), DIV_R (chia số real), giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến của phép toán tương ứng vào các ngõ IN1 và IN2. Nhập biến chứa kết quả ở ngõ OUT.

14.4.3. Ví dụ phép toán số học

Ví dụ 14.3: Đếm sản phẩm

Sản phẩm trên một băng tải được nhận biết bởi cảm biến S1. Tổng số lượng sản phẩm đếm được chứa trong MD20. Cứ 10 sản phẩm sẽ được đóng thành một thùng và số lượng thùng được chứa trong MD24. Số lượng sản phẩm có thể bị xóa bằng nút nhấn S2.

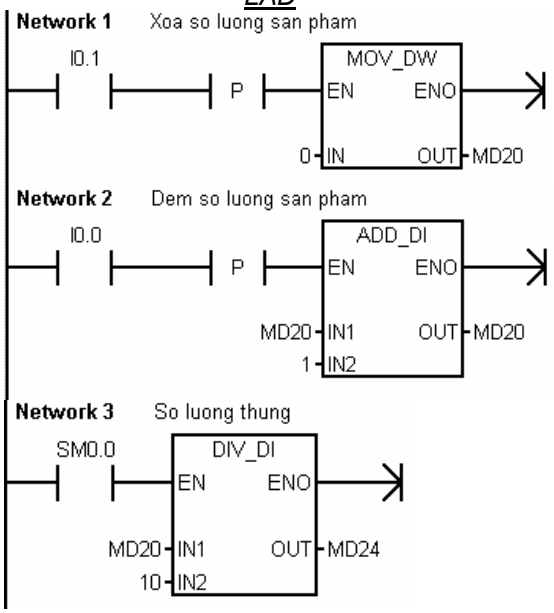
Giải

Bảng ký hiệu

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Cảm biến nhận biết sản phẩm
S2	I0.1	Nút nhấn xóa số lượng sản phẩm
So_SP	MD20	Giá trị sản phẩm đếm được
So_Thung	MD24	Số lượng thùng

Chương trình:

LAD



STL

Network 1 Xoa so luong san pham

```
LDW<   LD    I0.1
EU
MOVD   0, MD20
```

Network 2 Dem so luong san pham

```
LD     I0.0
EU
+D     1, MD20
```

Network 3 So luong thung

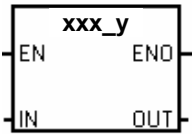
```
LD     SM0.0
MOVD   MD20, MD24
/D     10, MD24
```

14.5 Tăng và giảm thanh ghi

Tăng và giảm là một hình thức khác của quá trình đếm. Lệnh tăng hoặc giảm cộng 1 với ngõ vào hoặc lấy ngõ vào trừ 1 và kết quả được đưa ra ngõ ra.

Lệnh tăng hoặc giảm thực hiện được với byte, word và double word.

Biểu diễn tổng quát ở LAD:



với xxx_y có thể là:

- INC_B (tăng byte), INC_W (tăng word), INC_DW (tăng double word).
- DEC_B (giảm byte), DEC_W (giảm word), DEC_DW (giảm double word).

Ý nghĩa:

* Tăng: IN + 1 = OUT

* Giảm: IN -1 = OUT

Biểu diễn ở STL:

	Lệnh tăng:	*Lệnh giảm:*
Byte:	**INCB OUT**	**DECB OUT**
Word:	**INCW OUT**	**DECW OUT**
Double word:	**INCD OUT**	**DECD OUT**
Ý nghĩa:	**OUT + 1 = OUT**	**OUT -1 = OUT**

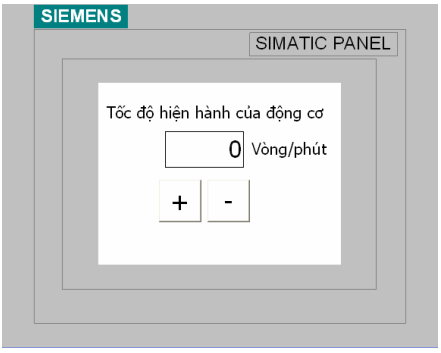
Để lấy lệnh ***tăng hoặc giảm thanh ghi*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Integer Math trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là:

INC_B: tăng byte
INC_W: tăng word
INC_DW: tăng double word

DEC_B: giảm byte
DEC_W: giảm word
DEC_DW: giảm double word

giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến cần tăng hoặc giảm ngõ IN. Nhập biến chứa kết quả ở ngõ OUT. (thông thường ngõ vào và ra có chung một biến).

Ví dụ 14.4*:* Hãy viết một chương trình con cho khâu tăng giảm tốc độ động cơ trên màn hình điều khiển TP170micro để khi ấn phím (+) thì tốc độ động cơ tăng dần lên, còn khi ấn phím (-) thì tốc độ động cơ giảm dần xuống.



Giải

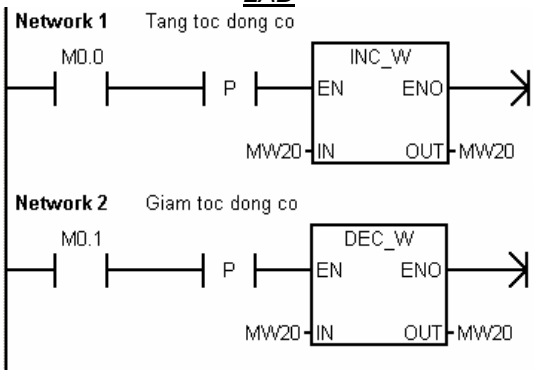
Nhằm giúp cho bạn đọc dễ hiểu, cứ mỗi lần ấn một phím (+) hoặc phím (-) thì tốc độ động cơ tăng hoặc giảm đi một vòng quay. Ở đây có thể có nhiều phương pháp nhưng chỉ giới hạn kiến thức cơ bản trong quyển sách, còn các kiến thức nâng cao xin bạn đọc tập 2.

Bảng ký hiệu

Ký hiệu	Địa chỉ	Chú thích
Phim +	M0.0	Tăng tốc động cơ
Phim -	M0.1	Giảm tốc động cơ
TD_Dongco	MW20	Biến tốc độ động cơ

Chương trình:

LAD



STL

Network 1 Tang toc dong co

LD M0.0

EU

INCW MW20

Network 2 Giam toc dong co

LD M0.1

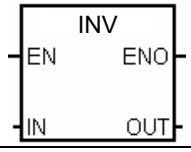
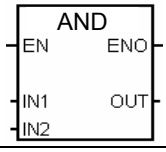
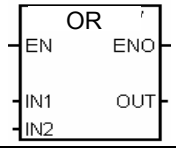
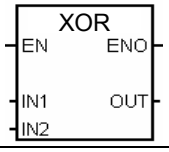
EU

DECW MW20

14.6. Các phép tóan logic số

14.6.1 Các logic số trong S7-200

Phép toán logic số sẽ thực hiện theo từng bit của hai tóan hạng số tương ứng hay một tóan hạng số với một hằng số. Các phép logic số có thể liệt kê ở bảng sau:

Phép toán:	INV	AND	OR	XOR
Ví dụ:	IN: ....1001 OUT: 0110	IN1: ....1010 IN2: ....1100 OUT:...1000	IN1:....1010 IN2: ....1100 OUT:...1110	IN1:....1010 IN2: ....1100 OUT:...0110
Biểu diễn:				

Các lệnh logic số là:

** Lệnh đảo byte (INVB), đảo word (INVW), đảo double word (INVD) sẽ đảo các bit ở ngõ vào IN và kết quả được đưa ra ngõ OUT.*

** Lệnh AND Byte (ANDB), AND Word (ANDW), và AND Double Word (ANDD) thực hiện AND các bit tương ứng của hai giá trị ngõ vào IN1 và IN2 và kết quả được đưa ra OUT.*
** Lệnh OR Byte (ORB), OR Word (ORW), và OR Double Word (ORD) thực hiện OR các bit tương ứng của hai giá trị ngõ vào IN1 và IN2 và kết quả được đưa ra OUT.*
** Lệnh XOR Byte (XORB), XOR Word (XORW), và XOR Double Word (XORD) thực hiện XOR các bit tương ứng của hai giá trị ngõ vào IN1 và IN2 và kết quả được đưa ra OUT.*

Để lấy các phép toán ***logic số*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Logical Operations trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là:

INV_B :	đảo byte	WOR_B :	OR byte
INV_W :	đảo word	WOR_W :	OR word
INV_DW :	đảo dword	WOR_DW :	OR double word
WAND_B :	AND byte	WXOR_B :	XOR byte
WAND_W :	AND word	WXOR_W :	XOR word
WAND_DW :	AND double word	WXOR_DW :	XOR double word

giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến cần tăng hoặc giảm ngõ IN. Nhập biến chứa kết quả ở ngõ OUT. (thông thường ngõ vào và ra có chung một biến).

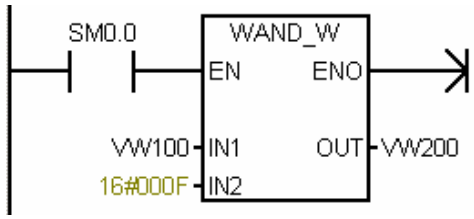
14.6.2. Ứng dụng

14.6.2.1 Che vị trí các bit

Một ứng dụng của phép tóan AND số là che vị trí bit. Để làm ẩn đi những vị trí bit không cần thiết hoặc không muốn xuất hiện thì ta sử dụng mặt nạ, ở những vị trí bit cần thiết ta cho giá trị "1" và làm ẩn những bit không cần thiết bằng cách cho bit tương ứng giá trị „0". Ví dụ ta cần lấy 4 bit cuối cùng của VW100 thì ta sẽ OR VW100 với mặt nạ sau: 0000 0000 0000 1111 và kết quả được chứa vào VW200.

Chương trình:

LAD



STL

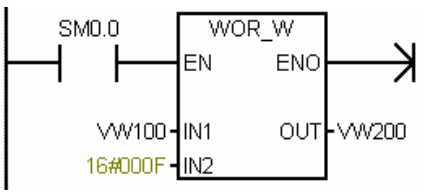
```
LD      SM0.0
MOVW    VW100, VW200
ANDW    16#000F, VW200
```

14.6.2.2 Chèn thêm bit

Một ứng dụng của phép tóan OR số là chèn bit. Muốn cho bit nào trong thanh ghi lên mức "1" thì ta sẽ OR ở bit tương ứng với giá trị 1. Ví dụ ta muốn 4 bit cuối của VW100 có giá trị „1" thì ta sẽ OR nó với giá trị sau: 0000 0000 0000 1111.

Chương trình:

LAD



STL

```
LD      SM0.0
MOVW    VW100, VW200
ORW     16#000F, VW200
```

14.7 Chức năng dịch/quay thanh ghi

14.7.1 Chức năng dịch chuyển thanh ghi

Với chức năng dịch chuyển thanh ghi, các bit của biến sẽ được dịch về bên phải hay bên trái theo một giá trị xác định. Tùy theo việc dịch chuyển thanh ghi là 1 Byte, 1 word hay 1 double word mà giá trị dịch có thể tối đa là 8, 16 hay 32.

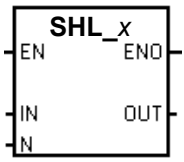
Nếu có thực hiện phép toán dịch (khác 0) thì nội dung của bit sau cùng thoát ra khỏi thanh ghi được chứa trong ô nhớ SM1.1. Còn nếu sau khi thực hiện phép dịch mà kết quả thu được của các thanh ghi là 0 thì ô nhớ SM1.0 được hệ điều hành đặt giá trị là 1.

Trong PLC họ S7-200, ngoài ngôn ngữ được biểu diễn theo chuẩn IEC 1131-3, còn có ngôn ngữ được biểu diễn theo chuẩn của hãng sản xuất (Siemens). Các lệnh dịch chuyển thanh ghi được cho như sau:

14.7.1.1 Dịch trái

Ở phép toán dịch trái, cho phép dịch byte, word và double word.

Cú pháp chung biểu diễn ở LAD là:



Với:

* *x*: Có thể là B (Byte), W (Word), DW (Double word).

* IN: Thanh ghi cần dịch trái có thể Byte, Word hoặc Double word.

* OUT: Nơi lưu trữ giá trị sau khi dịch trái. có thể Byte, Word hoặc Double word.

* N: Số lượng bit cần dịch trái. Tùy theo dịch byte, word hay double word mà N có giá trị max là 8, 16, 32.

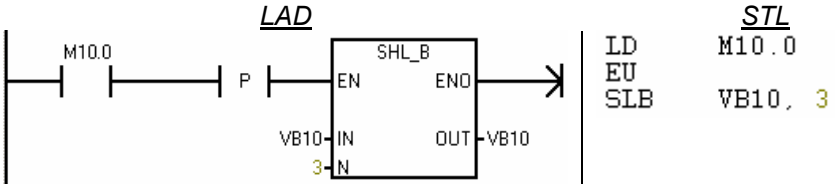
* EN, ENO: Xem mục 14.2.1

Cú pháp chung biểu diễn ở STL là:

* Dịch trái byte: SLB OUT, N
* Dịch trái word: SLW OUT, N
* Dịch trái double word: SLD OUT, N

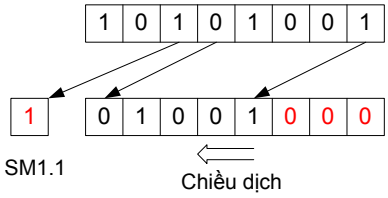
Chú ý: *Ở STL, thì kết quả sau phép dịch trái sẽ được chứa vào chính thanh ghi cần dịch.*

Ví dụ: Khi bit M10.0 từ "0"→ "1" thì yêu cầu dịch trái byte VB10 đi 3 vị trí, kết quả chứa vào VB10.



VB10 trước khi dịch:

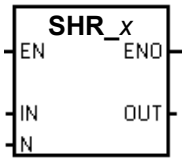
VB10 sau khi dịch 3 vị trí:



14.7.1.2 Dịch phải

Ở phép toán dịch phải, cho phép dịch byte, word và double word.

Cú pháp chung biểu diễn ở LAD là:



Với:

* *x*: Có thể là B (Byte), W (Word), DW (Double word).
* IN: Thanh ghi cần dịch phải có thể Byte, Word hoặc Double word.
* OUT: Nơi lưu trữ giá trị sau khi dịch phải. có thể Byte, Word hoặc Double word.
* N: Số lượng bit cần dịch phải. Tùy theo dịch byte, word hay double word mà N có giá trị max là 8, 16, 32.

* EN, ENO: Xem mục 14.2.1

Cú pháp chung biểu diễn ở STL là:

* Dịch phải byte: SRB OUT, N

* Dịch phải word: SRW OUT, N

* Dịch phải double word: SRD OUT, N

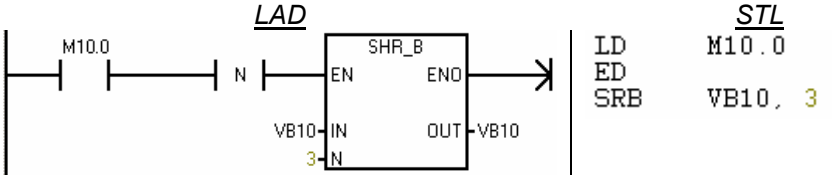
Chú ý: *Ở STL, thì kết quả sau phép dịch phải sẽ được chứa vào chính thanh ghi cần dịch.*

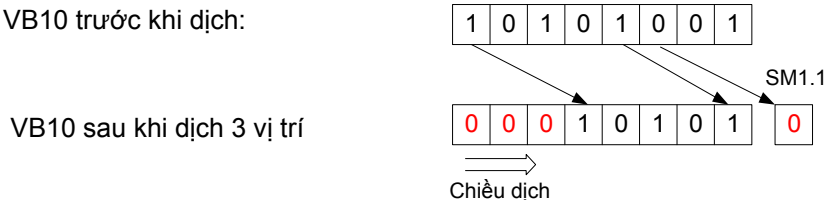
Để lấy các phép toán ***dịch thanh ghi*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Shift/Rotate trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là:

SHL_B : dịch trái byte
SHL_W : dịch trái word
SHL_DW : dịch trái double word
SHR_B : dịch phải byte
SHR_W : dịch phải word
SHR_DW : dịch phải double word

giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến cần dịch ở ngõ IN. Số bit cần dịch ở ngõ N. Nhập biến chứa kết quả ở ngõ OUT. (thông thường ngõ vào và ra có chung một biến).

Ví dụ: Khi bit M10.0 từ "1"→ "0" thì yêu cầu dịch trái byte VB10 đi 3 vị trí, kết quả chứa vào VB10.





14.7.2 Chức năng quay thanh ghi

Với chức năng quay thanh ghi, các bit của biến (byte, word, double word) sẽ được đẩy vòng tròn sang phải hay sang trái theo một giá trị xác định. Tại mỗi một lần quay, giá trị logic của bit bị đẩy ra khỏi đầu này cũng là giá trị logic được đưa vào đầu kia của biến.

Lệnh quay sẽ không thực hiện được nếu như số đếm lần quay có giá trị bằng 0 hay là bằng bội số của 8 đối với quay byte, 16 đối với word hay 32 đối với double word.

Đối với các giá trị khác của số đếm lần quay lớn hơn 8 (đối với byte), lớn hơn 16 (đối với word) hoặc 32 (đối với double word), thì lệnh sẽ thực hiện với số đếm lần quay mới bằng phần dư của số lần quay cũ chia cho 8, 16 hoặc chia cho 32.

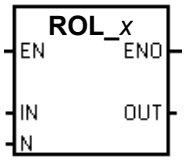
Nếu có thực hiện phép toán quay (khác 0) thì nội dung của bit sau cùng thoát ra khỏi thanh ghi được chứa vào ô nhớ SM1.1. Còn nếu sau khi thực hiện phép quay mà kết quả thu được của các thanh ghi là 0 thì ô nhớ SM1.0 được hệ điều hành đặt giá trị là 1.

Trong PLC họ S7-200, ngoài ngôn ngữ được biểu diễn theo chuẩn IEC 1131-3, còn có ngôn ngữ được biểu diễn theo chuẩn của hãng sản xuất (Siemens). Các lệnh quay thanh ghi được cho như sau:

14.7.2.1 Quay trái

Ở phép toán quay trái, cho phép quay byte, word và double word.

Cú pháp chung biểu diễn ỏ LAD là:



Với:

* *x*: Có thể là B (Byte), W (Word), DW (Double word).

* IN: Thanh ghi cần quay trái có thể Byte, Word hoặc Double word.

* OUT: Nơi lưu trữ giá trị sau khi quay trái. có thể Byte, Word hoặc Double word.

* N: Số lượng bit cần quay trái.

* EN, ENO: Xem mục 14.2.1

Cú pháp chung biểu diễn ỏ STL là:

* Quay trái byte: RLB OUT, N

* Quay trái word: RLW OUT, N

* Quay trái double word: RLD OUT, N

Chú ý: *Ở STL, thì kết quả sau phép quay trái sẽ được chứa vào chính thanh ghi cần quay.*

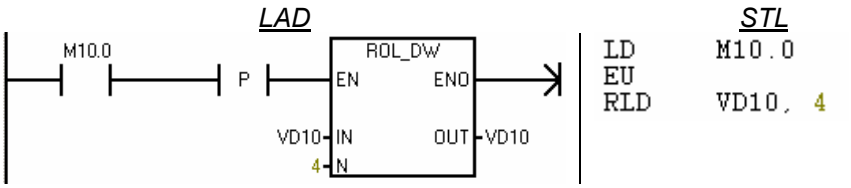
Để lấy các phép toán ***quay thanh ghi*** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Shift/Rotate trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là:

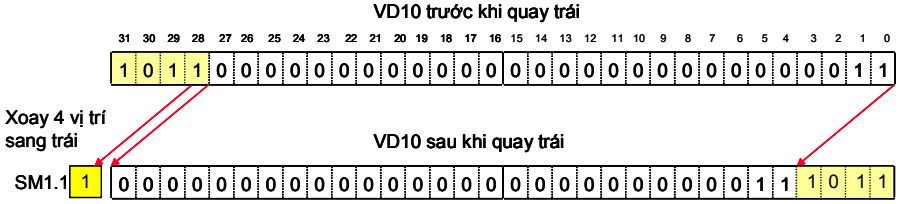
ROL_B: Quay trái byte
ROL_W: Quay trái word
ROL_DW: Quay trái double word

ROR_B: Quay phải byte
ROR_W: Quay phải word
ROR_DW: Quay phải double word

giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến cần quay ở ngõ IN. Số bit cần quay ở ngõ N. Nhập biến chứa kết quả ở ngõ OUT. (thông thường ngõ vào và ra có chung một biến).

Ví dụ: Khi bit M10.0 từ "0"→ "1" thì yêu cầu quay trái byte VD10 đi 4 vị trí, kết quả chứa vào VD10.

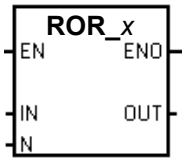




14.7.2.2 Quay phải

Tương tự như ở phép toán quay trái, ở phép toán quay phải cho phép quay byte, word và double word.

Cú pháp chung biểu diễn ở LAD là:



Với:

* *x*: Có thể là B (Byte), W (Word), DW (Double word).

* IN: Thanh ghi cần quay phải có thể Byte, Word hoặc Double word.

* OUT: Nơi lưu trữ giá trị sau khi quay phải. có thể Byte, Word hoặc Double word.

* N: Số lượng bit cần quay phải.

* EN, ENO: Xem mục 14.2.1

Cú pháp chung biểu diễn ở STL là:

* Quay phải byte: RLB OUT, N

* Quay phải word: RLW OUT, N

* Quay phải double word: RLD OUT, N

Chú ý: Ở STL, thì kết quả sau phép quay phải sẽ được chứa vào chính thanh ghi cần quay.