

Th.S Châu Chí Đức

*Kỹ thuật điều khiển*

# Lập trình PLC SIMATIC S7-200



Thành phố Hồ Chí Minh  
10-2008

# LỜI NÓI ĐẦU

Tự động hóa công nghiệp và dân dụng ngày càng phát triển. Bộ não trong các hệ thống tự động hóa là các bộ điều khiển lập trình. Việc học và tìm hiểu về các bộ điều khiển lập trình cũng như vận hành nó cho thật tốt đang là nhu cầu cấp thiết đối với học sinh, sinh viên các ngành kỹ thuật.

Hiện nay tài liệu để giảng dạy và tham khảo về kỹ thuật điều khiển lập trình còn khá hạn chế. Tài liệu "kỹ thuật điều khiển lập trình PLC Simatic S7-200", là quyển sách đầu tiên trong bộ sách về kỹ thuật điều khiển lập trình PLC họ SIMATIC S7, được biên soạn với mong muốn góp một phần nhỏ vào việc giảng dạy và tự học về kỹ thuật điều khiển lập trình của giáo viên, học sinh, sinh viên và đọc giả quan tâm về PLC họ SIMATIC S7-200 của công ty Siemens.

Tài liệu được chia thành 2 tập. Tập 1 bao gồm các phần cơ bản phù hợp với các bạn mới bắt đầu làm quen với PLC, tuy nhiên nó cũng có thể là tài liệu tham khảo cho các bạn đã có kiến thức cơ bản về PLC. Tập 2 là phần nâng cao tập trung về các vấn đề điều khiển số, truyền thông và màn hình điều khiển. Cấu trúc chung của các tập sách là ở mỗi chương trong các phần đều có ví dụ minh họa cho các mục, ngoài ra cuối mỗi chương có thêm một số câu hỏi và bài tập để đọc giả rèn luyện thêm.

Dù có một thời gian dài làm việc và giảng dạy về kỹ thuật điều khiển lập trình PLC họ SIMATIC, mạng truyền thông công nghiệp và truyền động của hãng Siemens cho rất nhiều đối tượng khác nhau cũng như đã rất cố gắng trong quá trình biên soạn nhưng tài liệu không tránh khỏi thiếu sót. Rất mong được sự góp ý chân thành của quý đọc giả để giúp tài liệu được hoàn thiện hơn. **Thư từ góp ý xin gửi về địa chỉ: [ccduc2006@gmail.com](mailto:ccduc2006@gmail.com).** Xin cảm ơn.

# **LỜI TÂM SƯ**

Tập 1 "kỹ thuật điều khiển lập trình PLC Simatic S7-200" đã được viết xong từ rất lâu. Nhưng vì nghĩ đến việc in ấn và phát hành quá nhiều khê, giá thành lại cao và phải chờ đợi thời gian rất lâu tập sách này mới đến tay bạn đọc, nên tác giả đã hoãn lại. Nghĩ rằng cung cấp cho đọc giả, các bạn học sinh, sinh viên và giáo viên thêm một tài liệu tham khảo để làm phong phú thêm kiến thức về tự động hóa là việc nên làm. Vì vậy tác giả chọn phương án phát hành qua mạng và truyền tay dưới dạng tập tin với phương châm "sách hữu ích thì mới có nhuận bút".

Các bạn thân mến!

Việc biên soạn tài liệu về kỹ thuật, nhất là kỹ thuật mới, đòi hỏi người biên soạn ngoài kinh nghiệm chuyên môn còn bỏ rất nhiều thời gian và công sức. Do đó sẽ là một niềm động viên vô cùng to lớn cho tác giả để tiếp tục hoàn thành tập 2, bộ sách về kỹ thuật điều khiển lập trình PLC SIMATIC S7-300/400, các tài liệu khác liên quan đến PLC họ SIMATIC, truyền thông công nghiệp, truyền động của hãng Siemens nếu được sự động viên từ tinh thần đến vật chất. Nếu thấy sách này giúp ích cho các bạn thì khi các bạn sở hữu nó (có được từ bất kỳ phương tiện nào) ở dạng tập tin hoặc được in ra ở dạng sách, xin vui lòng động viên tác giả bằng cách chuyển tiền vào **tài khoản số 49809449 cho CHÂU CHÍ ĐỨC, ngân hàng Thương mại Á Châu (ACB) chi nhánh Châu văn Liêm** với số tiền tùy theo ý của các bạn.

Nếu các bạn có những ý động viên khác xin gửi thông tin cho tác giả qua địa chỉ mail [ccduc2006@gmail.com](mailto:ccduc2006@gmail.com).

Cám ơn sự động viên của đọc giả.

# Mục lục

<b>1</b>	<b>Tổng quan về điều khiển .....</b>	<b>1</b>
1.1	Khái niệm chung về điều khiển .....	1
1.2	Cấu trúc một qui trình điều khiển .....	2
1.3	Các loại điều khiển .....	3
1.4	Hệ thống số .....	4
1.5	Các khái niệm xử lý thông tin .....	5
1.5.1	Bิต .....	5
1.5.2	Byte .....	5
1.5.3	Word .....	6
1.5.4	DoubleWord .....	6
<b>2</b>	<b>Bộ điều khiển lập trình PLC – Cấu trúc và phương thức hoạt động</b> .....	<b>7</b>
2.1	Giới thiệu .....	7
2.2	Sự khác nhau giữa hệ điều khiển bằng relay và hệ điều khiển bằng PLC .....	8
2.3	Cấu trúc của một PLC .....	11
2.4	Các khối của PLC .....	13
2.4.1	Khối nguồn cung cấp .....	13
2.4.2	Bộ nhớ chương trình .....	14
2.4.3	Khối trung tâm (CPU) .....	15
2.4.4	Khối vào .....	15
2.4.5	Khối ra .....	16
2.4.6	Các khối đặc biệt .....	16
2.5	Phương thức thực hiện chương trình trong PLC .....	16
<b>3</b>	<b>Cảm biến và cơ cấu chấp hành trong điều khiển logic</b> .....	<b>19</b>
3.1	Cảm biến .....	19
3.1.1	Giới thiệu .....	19
3.1.2	Nối dây cho cảm biến .....	19
3.1.2.1	Switch .....	20
3.1.2.2	Ngõ ra TTL .....	20
3.1.2.3	Ngõ ra Sinking/Sourcing .....	20
3.1.2.4	Ngõ ra Solid state relay .....	23
3.1.3	Phát hiện đối tượng .....	23
3.1.3.1	Chuyển mạch tiếp xúc .....	23
3.1.3.2	Reed Switches .....	23
3.1.3.3	Cảm biến quang (Optical Sensor) .....	23
3.1.3.4	Cảm biến điện dung (Capacitive Sensor) .....	25
3.1.3.5	Cảm biến điện cảm (Inductive Sensor) .....	26
3.1.3.6	Cảm biến siêu âm (Ultrasonic sensor) .....	28
3.1.3.7	Hiệu ứng Hall (Hall Effect) .....	28
3.1.3.8	Lưu lượng (Fluid Flow) .....	28
3.1.4	Tóm tắt .....	29
3.2	Cơ cấu chấp hành .....	29
3.2.1	Giới thiệu .....	29

3.2.2 Solenoid .....	29
3.2.3 Van điều khiển (VALVE) .....	30
3.2.4 Xy lanh (CYLINDER) .....	32
3.2.5 Động cơ .....	33
3.2.6 Các cơ cấu chấp hành khác.....	34
<b>4 Bộ điều khiển lập trình PLC Simatic S7-200 .....</b>	<b>35</b>
4.1 Cấu hình cứng .....	35
4.1.1 Khối xử lý trung tâm .....	35
4.1.2 Khối mở rộng .....	39
4.1.2.1 Digital module .....	39
4.1.2.2 Analog module .....	40
4.1.2.3 Intelligent module .....	41
4.1.2.4 Function module .....	41
4.2 Màn hình điều khiển .....	42
4.3 Các vùng nhớ .....	43
4.4 Qui ước địa chỉ trong PLC S7-200 .....	46
4.4.1 Truy xuất theo bit .....	46
4.4.2 Truy xuất theo byte (8 bit) .....	46
4.4.3 Truy xuất theo word (16 bit) .....	46
4.4.4 Truy xuất theo 2 word (Double word = 32 bit) .....	47
4.5 Xử lý chương trình .....	48
<b>5 Kết nối dây giữa PLC và thiết bị ngoại vi .....</b>	<b>51</b>
5.1 Kết nối dây giữa PLC và các thiết bị ngoại vi .....	51
5.1.1 Giới thiệu CPU 224 và cách kết nối với thiết bị ngoại vi ....	51
5.1.2 Kết nối với máy tính .....	52
5.1.3 Nối nguồn cung cấp cho CPU .....	54
5.1.4 Kết nối vào/ra số với ngoại vi .....	54
5.1.4.1 Kết nối các ngõ vào số với ngoại vi .....	55
5.1.4.2 Kết nối các ngõ ra số với ngoại vi .....	57
5.2 Kiểm tra việc kết nối dây bằng phần mềm .....	60
5.2.1 Status Chart .....	60
5.2.2 Giám sát và thay đổi biến với Status Chart .....	60
5.2.3 Cưỡng bức biến với Status Chart .....	62
5.2.4 Ứng dụng Status Chart trong việc kiểm tra kết nối dây trong S7-200 .....	63
5.3 Câu hỏi và bài tập .....	64
<b>6 Phần mềm Micro/WIN và ngôn ngữ lập trình .....</b>	<b>65</b>
6.1 Cài đặt phần mềm STEP 7-Micro/WIN .....	65
6.1.1 Yêu cầu hệ điều hành và phần cứng .....	65
6.1.2 Cài đặt phần mềm .....	65
6.2 Các phần tử cơ bản trong chương trình PLC S7-200 .....	66
6.2.1 Chương trình chính OB1 (main program) .....	66
6.2.2 Chương trình con SUB (subroutine) .....	66
6.2.3 Chương trình ngắt INT(interrupt routine) .....	67
6.2.4 Khối hệ thống (system block) .....	67

6.2.5	Khối dữ liệu (data block) .....	67
6.3	Ngôn ngữ lập trình .....	67
6.3.1	Dạng hình thang: LAD (Ladder logic) .....	68
6.3.2	Dạng khối chức năng: FBD (Function Block Diagram) .....	68
6.3.3	Dạng liệt kê lệnh: STL (StaTement List) .....	69
6.4	Soạn thảo chương trình với phần mềm STEP7-Micro/WIN V4.0 SP6 .....	69
6.4.1	Mở màn hình soạn thảo chương trình .....	69
6.4.1.1	Vùng soạn thảo chương trình .....	70
6.4.1.2	Cây lệnh .....	70
6.4.1.3	Thanh chức năng .....	70
6.4.2	Thanh công cụ (Toolbar) trong STEP7-Micro/WIN .....	75
6.4.3	Tạo một dự án STEP 7-Micro/WIN .....	77
6.4.3.1	Tạo dự án mới .....	77
6.4.3.2	Lưu dự án .....	77
6.4.3.3	Mở một dự án .....	78
6.4.4	Thư viện .....	78
6.4.5	Hệ thống trợ giúp trong STEP 7-Micro/WIN .....	79
6.4.6	Xóa bộ nhớ CPU .....	80
6.4.7	Mở một dự án đang tồn tại sẵn .....	80
6.4.8	Kết nối truyền thông S7-200 với thiết bị lập trình .....	81
6.4.9	Tải dự án từ PLC .....	82
6.4.9.1	Tải một khối hoặc ba khối .....	82
6.4.9.2	Tải vào một dự án mới hoặc dự án rỗng .....	82
6.4.9.3	Tải vào một dự án tồn tại .....	82
6.4.9.4	Thủ tục tải dự án từ PLC về thiết bị lập trình .....	82
6.4.10	Nạp (download) một dự án vào PLC .....	83
6.4.11	Thiết lập cấu hình chung cho phần mềm (menu option và customize) .....	85
6.4.11.1	Menu Option .....	85
6.4.11.2	Menu Custommize .....	86
6.4.12	Soạn thảo chương trình .....	88
<b>7</b>	<b>Các phép toán logic .....</b>	<b>95</b>
7.1	Ngăn xếp (logic stack) trong S7-200 .....	95
7.2	Các phép toán logic cơ bản .....	96
7.2.1	Phép toán AND .....	96
7.2.2	Phép toán OR .....	97
7.2.3	Tổ hợp các cổng AND và OR .....	98
7.2.3.1	AND trước OR .....	98
7.2.3.2	OR trước AND .....	98
7.2.4	Phép toán XOR .....	99
7.3	Xử lý các tiếp điểm, cảm biến được nối với ngõ vào PLC .....	100
7.4	Ví dụ ứng dụng các liên kết logic .....	102
7.4.1	Mạch tự duy trì ưu tiên mở máy .....	102
7.4.2	Mạch tự duy trì ưu tiên dừng máy .....	103
7.4.3	Điều khiển ON/OFF động cơ có chỉ báo .....	104
7.4.4	Điều khiển đảo chiều quay động cơ .....	106
7.5	Bit nhớ M (bit memory) .....	109

7.6	Các lệnh SET, RESET và mạch nhớ RS .....	111
7.6.1	Lệnh SET .....	111
7.6.2	Lệnh RESET (R) .....	112
7.6.3	Mạch nhớ R-S .....	112
7.6.3.1	Ưu tiên SET (khâu SR) .....	112
7.6.3.2	Ưu tiên RESET (khâu RS) .....	113
7.6.4	Các qui tắc khi sử dụng Set và Reset .....	114
7.6.5	Ví dụ ứng dụng mạch nhớ R-S .....	114
7.7	Các lệnh nhận biết cạnh tín hiệu và lệnh NOT .....	118
7.7.1	Lệnh NOT .....	118
7.7.1	Các lệnh nhận biết cạnh tín hiệu .....	118
7.8	Các Bit nhớ đặc biệt (Special Memory bits) .....	120
7.9	Câu hỏi và bài tập .....	121
<b>8</b>	<b>Thiết kế theo logic Bool &amp; biểu đồ Karnaugh .....</b>	<b>125</b>
8.1	Giới thiệu .....	125
8.2	Đại số BOOL .....	125
8.3	Thiết kế Logic .....	127
8.3.1	Các kỹ thuật đại số Bool .....	131
8.4	Các dạng logic chung .....	132
8.4.1	Dạng cồng phức .....	132
8.4.2	Multiplexers .....	132
8.5	Một số ví dụ thiết kế đơn giản với đại số bool .....	133
8.5.1	Các chức năng logic cơ bản .....	133
8.5.2	Hệ thống an toàn xe hơi .....	134
8.5.3	Quay phải/trái động cơ .....	134
8.5.4	Cảnh báo trộm .....	135
8.6	Biểu đồ Karnaugh .....	136
8.6.1	Giới thiệu .....	136
8.7	Câu hỏi và bài tập .....	139
<b>9</b>	<b>Bộ định thời (Timer)</b> .....	<b>147</b>
9.1	Giới thiệu .....	147
9.2	Timer đóng mạch chậm TON .....	148
9.3	Timer đóng mạch chậm có nhớ TONR .....	149
9.4	Timer mở mạch chậm TOF .....	150
9.5	Ứng dụng Timer .....	152
9.5.1	Tạo xung có tần số theo mong muốn .....	152
9.5.2	Tạo Timer xung và timer xung có nhớ .....	152
9.5.2.1	Timer xung (Pulse timer) .....	152
9.5.2.2	Timer xung có nhớ (Extended Pulse timer) .....	153
9.5.3	Đảo chiều quay động cơ có khống chế thời gian .....	154
9.5.4	Chiếu sáng Garage .....	155
9.5.5	Thiết bị rót chất lỏng vào thùng chứa .....	156
9.6	Câu hỏi và bài tập .....	161
<b>10</b>	<b>Bộ đếm (Counter)</b> .....	<b>170</b>
10.1	Giới thiệu .....	170
10.2	Bộ đếm lên CTU (Count Up) .....	171

10.3	Bộ đếm xuống CTD (Count Down) .....	172
10.4	Bộ đếm lên-xuống CTUD (Count Up/Down) .....	173
10.5	Ứng dụng bộ đếm .....	174
10.5.1	Đếm sản phẩm được đóng gói .....	174
10.5.2	Kiểm soát chốt cho Garage ngầm .....	175
10.6	Câu hỏi và bài tập .....	178
<b>11</b>	<b>Điều khiển trình tự .....</b>	<b>181</b>
11.1	Cấu trúc chung của một chương trình điều khiển .....	181
11.2	Điều khiển trình tự .....	182
11.2.1	Giới thiệu .....	182
11.2.2	Phương pháp lập trình điều khiển trình tự .....	184
11.3	Các thủ tục tổng quát để thiết kế bài toán trình tự .....	186
11.4	Cấu trúc của bài toán điều khiển trình tự .....	188
11.4.1	Chuỗi trình tự .....	188
11.4.2	Kiểu hoạt động .....	188
11.4.3	Các thông báo .....	190
11.4.4	Kích hoạt ngõ ra .....	190
11.5	Các ký hiệu .....	190
11.6	Bước trình tự .....	191
11.7	Các lệnh biểu diễn trong sơ đồ chức năng .....	193
11.8	Các chế độ hoạt động, cảnh báo và xuất lệnh .....	197
11.8.1	Bảng điều khiển .....	198
11.8.2	Các khâu chế độ hoạt động có cảnh báo .....	199
11.8.3	Hiển thị bước trình tự .....	201
11.8.4	Xuất lệnh .....	201
11.9	Các ví dụ ứng dụng .....	201
11.9.1	Máy phay đơn giản .....	201
11.9.2	Bảng chuyền đếm táo .....	205
11.10	Câu hỏi và bài tập .....	210
<b>12</b>	<b>An toàn trong PLC .....</b>	<b>218</b>
12.1	Khái niệm và mục đích .....	218
12.2	Hư hỏng ở PLC .....	218
12.3	Các quan điểm về kỹ thuật an toàn ở PLC .....	219
12.3.1	Các lỗi nguy hiểm và không nguy hiểm .....	219
12.3.2	Các cách giải quyết cho hoạt động an toàn của thiết bị điều khiển PLC .....	220
12.4	Bảo vệ các ngõ ra PLC .....	223
12.4.1	Bảo vệ ngõ ra dùng Transistor .....	224
12.4.2	Bảo vệ ngõ ra Röle có nguồn điều khiển DC .....	224
12.4.3	Bảo vệ ngõ ra Röle và ngõ ra AC có nguồn điều khiển AC .....	224
12.5	Câu hỏi và bài tập .....	225
<b>13</b>	<b>Chuyển điều khiển kết nối cứng sang điều khiển bằng PLC .....</b>	<b>226</b>
13.1	Kết nối ngõ vào/ ra của PLC từ một sơ đồ điều khiển có tiếp điểm .....	226
13.2	Chuyển đổi điều khiển từ contactor thành PLC .....	228

13.2.1	Điều khiển thiết bị bù công suất phản kháng .....	230
13.2.2	Thiết bị nghiền .....	237
13.3	Điều khiển khí nén .....	241
13.3.1	Máy uốn thanh kim loại .....	242
13.3.2	Máy dập miệng ống kim loại .....	246
13.4	Câu hỏi và bài tập .....	253
<b>14</b>	<b>Các phép toán cơ bản trong điều khiển số .....</b>	<b>257</b>
14.1	Các dạng số trong PLC .....	257
14.1.1	Kiểu dữ liệu Integer (INT) .....	257
14.1.2	Kiểu dữ liệu Double Integer (DINT) .....	258
14.1.3	Kiểu dữ liệu số thực (REAL) .....	259
14.1.4	Kiểu dữ liệu số BCD (Binary Coded Decimal) .....	260
14.2	Chức năng sao chép .....	261
14.2.1	Các lệnh sao chép, trao đổi nội dung .....	261
14.2.2	Các lệnh sao chép một mảng lớn dữ liệu .....	263
14.3	Phép toán so sánh .....	264
14.4	Phép toán số học .....	266
14.4.1	Cộng và trừ .....	266
14.4.2	Nhân và chia .....	267
14.4.3	Ví dụ phép toán số học .....	268
14.5	Tăng và giảm thanh ghi .....	269
14.6	Các phép toán logic số .....	271
14.6.1	Các logic số trong S7-200 .....	271
14.6.2	Ứng dụng .....	272
14.6.2.1	Che vị trí các bit .....	272
14.6.2.2	Chèn thêm bit .....	273
14.7	Chức năng dịch/quay thanh ghi .....	273
14.7.1	Chức năng dịch chuyển thanh ghi .....	273
14.7.1.1	Dịch trái .....	273
14.7.1.2	Dịch phải .....	274
14.7.2	Chức năng quay thanh ghi .....	275
14.7.2.1	Quay trái .....	276
14.7.2.2	Quay phải .....	277

# 1 Tổng quan về điều khiển

## 1.1 Khái niệm chung về điều khiển

Điều khiển có nhiệm vụ thực hiện các chức năng riêng của một máy móc hay thiết bị theo một trình tự hoạt động định trước phụ thuộc vào trạng thái của máy hay bộ phát tín hiệu.

Sự điều khiển được phân biệt theo các đặc điểm khác nhau:

### \* **Theo loại biểu diễn thông tin**

- **Điều khiển nhị phân:** Xử lý tín hiệu đầu vào nhị phân (tín hiệu 1-0) thành các tín hiệu ra nhị phân.

- **Điều khiển số:** Xử lý các thông tin số, có nghĩa các thông tin được biểu diễn dưới dạng số.

### \* **Theo loại xử lý tín hiệu**

- **Điều khiển liên kết:** Các trạng thái tín hiệu xác định của ngõ ra được điều khiển bởi các trạng thái tín hiệu của ngõ vào tuỳ thuộc vào các chức năng liên kết (AND, OR, NOT).

- **Điều khiển trình tự:** Điều khiển với trình tự theo từng bước, sự đóng mạch của một bước sau xảy ra phụ thuộc vào điều kiện đóng mạch tiếp theo. Điều kiện đóng mạch tiếp theo có thể phụ thuộc vào qui trình hay thời gian.

- **Điều khiển không đồng bộ:** Việc điều khiển được xử lý ở sự thay đổi trực tiếp của tín hiệu ngõ vào không cần tín hiệu xung phụ (điều khiển chậm).

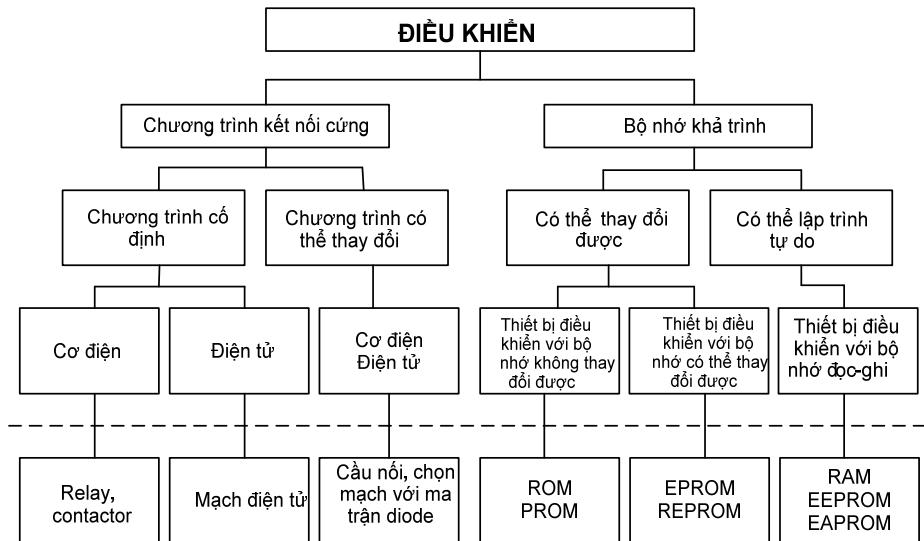
- **Điều khiển đồng bộ xung:** Việc điều khiển được xử lý ở các tín hiệu chỉ đồng bộ với một tín hiệu xung (điều khiển nhanh).

### \* **Theo loại thực hiện chương trình**

- **Điều khiển theo chương trình kết nối cứng:** Loại điều khiển này có thể được lập trình cố định, có nghĩa không thể thay đổi được ví dụ như lắp đặt dây nối cố định hay có thể thay đổi chương trình thông qua các đầu nối (ma trận diode).

- **Điều khiển khả trình:** Chức năng điều khiển được lưu giữ trong một bộ nhớ chương trình. Nếu sử dụng bộ nhớ đọc/ghi (RAM), thì có thể thay đổi chương trình mà không cần can thiệp đến phần cơ khí (điều khiển có thể lập trình tự do). Nếu ngược lại là một bộ nhớ chỉ đọc (ROM), thì chương trình có thể

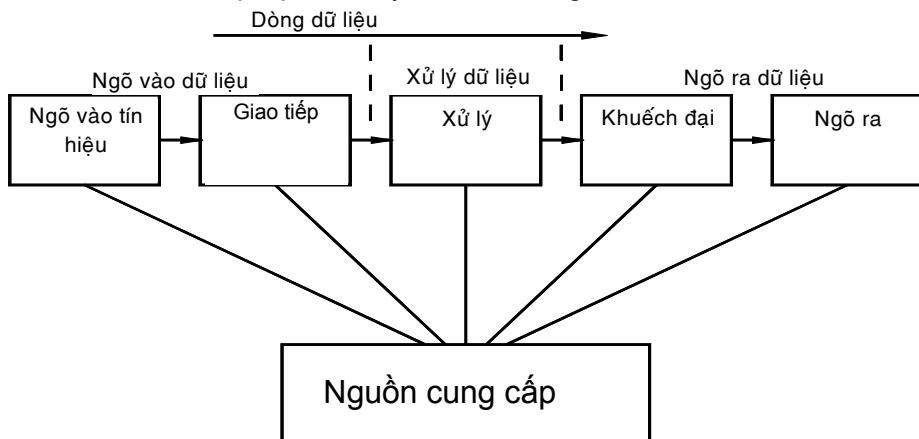
được thay đổi bằng cách thay đổi bộ nhớ (điều khiển có thể thay đổi chương trình).



Hình 1.1: Sơ đồ các loại điều khiển

## 1.2 Cấu trúc một qui trình điều khiển

Mỗi sự điều khiển được chia ra làm 3 bộ phận hợp thành: Ngõ vào dữ liệu (ngõ vào tín hiệu), Xử lý dữ liệu (xử lý tín hiệu cũng như các liên kết) và ngõ ra dữ liệu (ngõ ra tín hiệu). Dòng dữ liệu trong một sự điều khiển xảy ra từ đầu vào dữ liệu qua phần xử lý dữ liệu đến ngõ ra dữ liệu.



Hình 1.2: Cấu trúc chung của một qui trình điều khiển

+ **Ngõ vào tín hiệu**: Bao gồm các loại tín hiệu của các bộ phát tín hiệu như nút nhấn, công tắc hành trình, cảm biến điện dung, cảm biến điện cảm .v.v..

Tuỳ thuộc vào loại điều khiển, các tín hiệu có thể là nhị phân, số hay tín hiệu tương tự.

- + **Giao tiếp:** Phần này cần thiết, nếu tín hiệu của một hệ thống lạ cần phải được xử lý. Một bộ phận chuyển đổi từ tín hiệu ngõ vào thành tín hiệu phù hợp với mức của tín hiệu xử lý được đặt ở phần giao tiếp.
- + **Xử lý:** Toàn bộ các liên kết, trình tự thời gian, các chức năng nhớ, đếm .v.v.. được thực hiện trong phần này. Phần xử lý là phần chính của tất cả các hệ thống điều khiển. Các kỹ thuật điều khiển có tiếp điểm như khởi động từ phụ, relay thời gian, kỹ thuật điều khiển bằng mạch điện tử (như AND, OR, NOT ...) được PLC hay máy tính điều khiển quá trình tổng hợp tại đây.
- + **Khuếch đại:** Các tín hiệu từ phần xử lý có mức độ công suất bé được khuếch đại lớn lên nhiều lần ở đây để có thể điều khiển các khởi động từ, van từ hay các đối tượng điều khiển khác và các đèn báo.
- + **Ngõ ra:** Phần này được kết nối với đối tượng điều khiển mà có ảnh hưởng trực tiếp đến quá trình điều khiển (ví dụ: Khởi động từ, van từ, thyristor, v.v..)

### 1.3 Các loại điều khiển

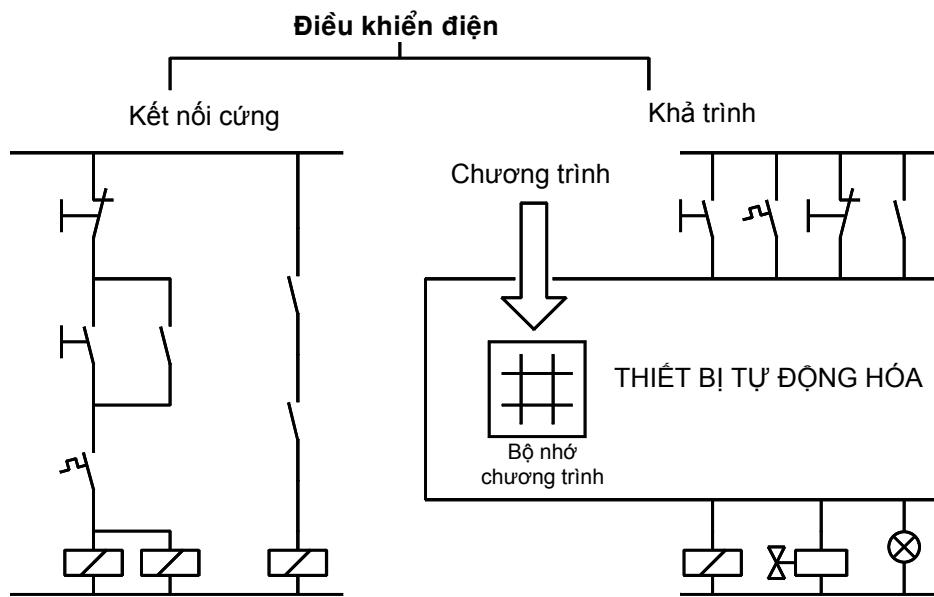
Trong kỹ thuật điều khiển cũng như tự động hóa, người ta chia ra làm hai loại điều khiển: điều khiển kết nối cứng và điều khiển khả trình.

#### \* Điều khiển kết nối cứng

Điều khiển kết nối cứng là loại điều khiển mà các chức năng của nó được đặt cố định (nối dây). Nếu muốn thay đổi chức năng điều đó có nghĩa là thay đổi kết nối dây. Điều khiển kết nối cứng có thể thực hiện với các tiếp điểm (Relay, khởi động từ, v.v.) hay điện tử (mạch điện tử).

#### \* Điều khiển khả trình (PLC)

Điều khiển khả trình là loại điều khiển mà chức năng của nó được đặt cố định thông qua một chương trình còn gọi là bộ nhớ chương trình. Sự điều khiển bao gồm một thiết bị điều khiển mà ở đó tất cả các bộ phát tín hiệu cần thiết và đối tượng điều khiển được kết nối cho một chức năng cụ thể. Nếu chức năng điều khiển cần được thay đổi, thì chỉ phải thay đổi chương trình bằng thiết bị lập trình ở đối tượng điều khiển tương ứng hay cắm một bộ nhớ chương trình đã lập trình khác vào trong điều khiển.



Hình 1.3: Hai loại điều khiển trong sản xuất

## 1.4 Hệ thống số

Trong xử lý các phần tử nhớ, các ngõ vào, các ngõ ra, thời gian, các ô nhớ v.v... bằng PLC thì hệ thập phân không được sử dụng mà là hệ thống số nhị phân (hệ hai trị).

### \* Hệ nhị phân

Hệ nhị phân chỉ có các số 0 và 1, có thể được đọc và biểu diễn giá trị dễ dàng trong kỹ thuật. Giá trị định vị của một số nhị phân là số mũ của hai. Độ lớn của số thông thường được biểu diễn ở dạng mã BCD (Binary-Code-Decimal). Đối với mỗi số Decimal được viết với số nhị phân 4 vị trí.

### \* Số thập lục phân (Hexadecimal)

Hệ thập lục phân có 16 ký hiệu khác nhau từ 0-9 và A-F. Giá trị định vị của một số thập lục phân số mũ của 16.

- **Hệ nhị phân:** Chữ số: 0,1

Giá trị định vị = Số mũ của cơ số 2

$$\begin{array}{cccc}
 2^3 & 2^2 & 2^1 & 2^0 \\
 8 & 4 & 2 & 1
 \end{array}$$

Ví dụ:

$$\begin{array}{ccccccc}
 & 1 & 1 & 0 & 1 \\
 & \swarrow & \searrow & \downarrow & \downarrow \\
 1 \cdot 2^3 & + & 1 \cdot 2^2 & + & 0 \cdot 2^1 & + & 1 \cdot 2^0 \\
 8 & + & 4 & + & 0 & + & 1 = 13_D
 \end{array}$$

- Hệ thập lục phân: chữ số: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E;F

Giá trị định vị = Số mũ của cơ số 16

$$\begin{array}{cccc}
 16^3 & & 16^2 & & 16^1 & & 16^0 \\
 & & 4096 & & 256 & & 16 & & 1
 \end{array}$$

Ví dụ:

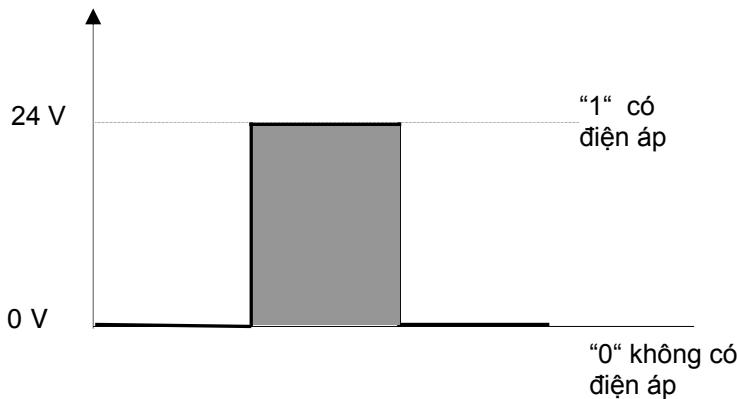
$$\begin{array}{ccccccc}
 & 2 & A & B \\
 & \swarrow & \searrow & \downarrow & \downarrow \\
 2 \cdot 16^2 & + & A \cdot 16^1 & + & B \cdot 16^0 \\
 512 & + & 160 & + & 11 = & & 683_D
 \end{array}$$

## 1.5 Các khái niệm xử lý thông tin

Trong PLC, hầu hết các khái niệm trong xử lý thông tin cũng như dữ liệu đều được sử dụng như Bit, Byte, Word và doubleword.

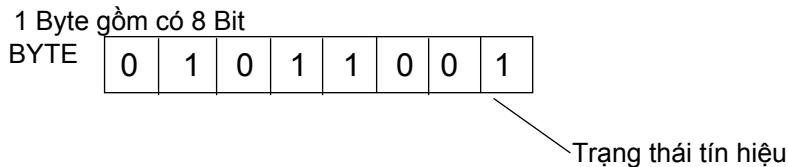
### 1.5.1 Bit

Bit là đơn vị thông tin nhị phân nhỏ nhất, có thể có giá trị 0 hoặc 1.



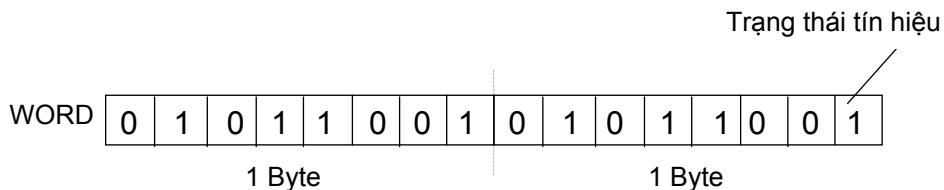
Hình 1.4: Một bit có thể có trạng thái tín hiệu “1” hoặc “0”

### 1.5.2 Byte



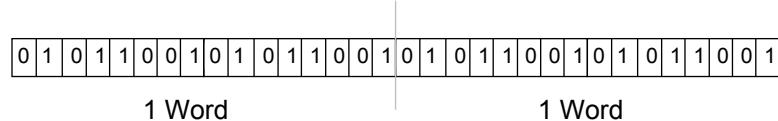
### 1.5.3 Word

1 Word gồm có 2 Byte hay 16 Bit. Với Word có thể biểu diễn ở các dạng: số nhị phân, ký tự hay câu lệnh điều khiển.

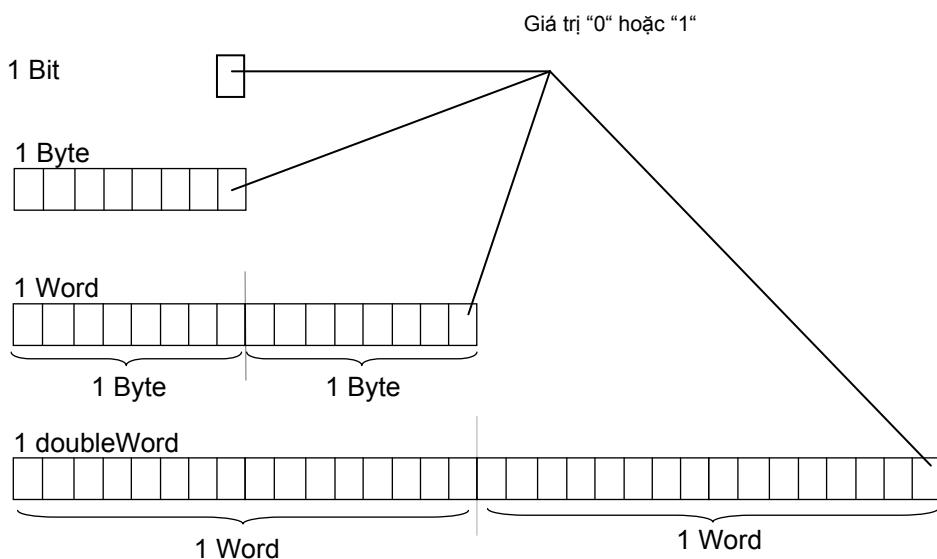


#### 1.5.4 DoubleWord

1 DoubleWord gồm có 4 Byte hay 32 Bit. Với DoubleWord có thể biểu diễn ở các dạng: số nhị phân, ký tự hay câu lệnh điều khiển.



## Tóm tắt:



## 2 Bộ điều khiển lập trình PLC – Cấu trúc và phương thức hoạt động

### 2.1 Giới thiệu

Các thành phần của kỹ thuật điều khiển điện và điện tử ngày càng đóng một vai trò vô cùng to lớn trong lĩnh vực tự động hóa ngày càng cao. Trong những năm gần đây, bên cạnh việc điều khiển bằng Relay và khởi động từ thì việc điều khiển có thể lập trình được càng phát triển với hệ thống đóng mạch điện tử và thực hiện lập trình bằng máy tính. Trong nhiều lĩnh vực, các loại điều khiển cũ đã được thay đổi bởi các bộ điều khiển có thể lập trình được, có thể gọi là các bộ điều khiển logic khả trình, viết tắt trong tiếng Anh là **PLC** (**Programmable Logic Controller**).

Sự khác biệt cơ bản giữa điều khiển logic khả trình (thay đổi được qui trình hoạt động) và điều khiển theo kết nối cứng (không thay đổi được qui trình hoạt động) là: *Sự kết nối dây không còn nữa, thay vào đó là chương trình.*

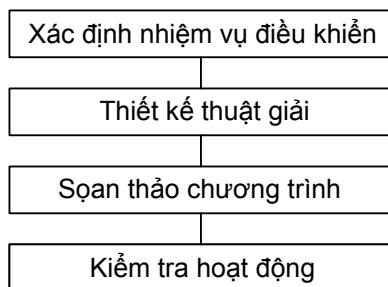
Có thể lập trình cho PLC nhờ vào các ngôn ngữ lập trình đơn giản. Đặc biệt đối với người sử dụng không cần nhớ vào các ngôn ngữ lập trình khó khăn, cũng có thể lập trình PLC được nhờ vào các liên kết logic cơ bản.

Như vậy thiết bị PLC làm nhiệm vụ thay thế phần mạch điện điều khiển trong khâu xử lý số liệu. Nhiệm vụ của sơ đồ mạch điều khiển sẽ được xác định bởi một số hữu hạn các bước thực hiện xác định gọi là **chương trình**. Chương trình này mô tả các bước thực hiện gọi một tiến trình điều khiển, tiến trình này được lưu vào bộ nhớ nên được gọi là **điều khiển lập trình nhớ** hay **điều khiển khả trình**. Trên cơ sở khác nhau ở khâu xử lý số liệu có thể biểu diễn hai hệ điều khiển như sau:

**Các bước thiết lập hệ  
điều khiển bằng relay điện**



**Các bước thiết lập hệ  
điều khiển bằng PLC**



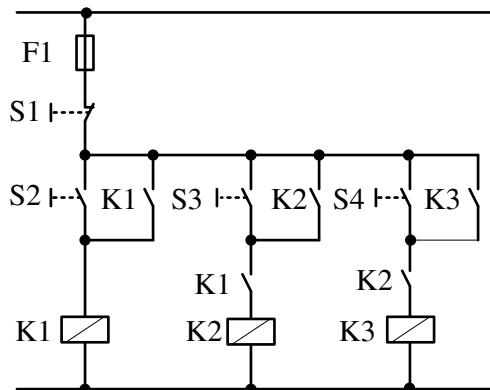
Khi thay đổi nhiệm vụ điều khiển thì người ta thay đổi mạch điều khiển: Lắp lại mạch, thay đổi các phần tử mới ở hệ điều khiển bằng relay điện. Trong khi đó khi thay đổi nhiệm vụ điều khiển ở hệ điều khiển logic khả trình (PLC) thì người ta chỉ thay đổi chương trình soạn thảo.

## 2.2 Sự khác nhau giữa hệ điều khiển bằng relay và hệ điều khiển bằng PLC

Sự khác nhau giữa hệ điều khiển bằng relay và hệ điều khiển bằng PLC có thể minh họa một cách cụ thể như sau:

Điều khiển hệ thống của 3 máy bơm qua 3 khởi động từ K1, K2, K3. Trình tự điều khiển như sau: Các khởi động từ chỉ được phép thực hiện tuần tự, nghĩa là K1 đóng trước, tiếp theo K2 đóng và cuối cùng K3 mới đóng.

Để thực hiện nhiệm vụ theo yêu cầu trên mạch điều khiển được thiết kế như sau:



Hình 2.1: Mạch điều khiển trình tự 3 máy bơm

Khởi động từ K2 sẽ đóng khi công tắc S3 đóng với điều kiện là khởi động từ K1 đã đóng trước đó. Phương thức điều khiển như vậy được gọi là điều khiển trình tự. Tiến trình điều khiển này được thực hiện một cách cưỡng bức.

- Bốn nút nhấn S1, S2, S3, S4: Các phần tử nhập tín hiệu.
- Các tiếp điểm K1, K2, K3 và các mối nối liên kết là các phần tử xử lý.
- Các khởi động từ K1, K2, K3 là kết quả xử lý.

Nếu thay đổi mạch điện điều khiển ở phần xử lý bằng hệ PLC ta có thể biểu diễn hệ thống như sau:

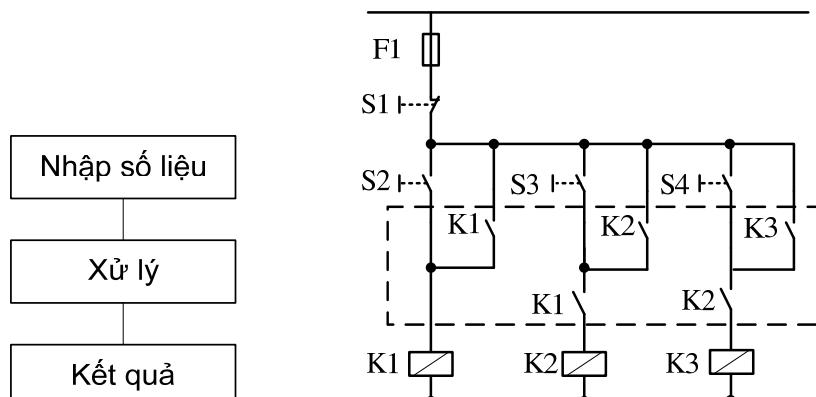
- *Phần tử vào*: Các nút nhấn S1, S2, S3, S4 vẫn giữ nguyên.
- *Phần tử ra*: Ba khởi động từ K1, K2, K3, để đóng và mở ba máy bơm vẫn giữ nguyên.
- *Phần tử xử lý*: Được thay thế bằng PLC.

Sơ đồ kết nối với PLC được cho như ở hình 2.3. Trình tự đóng mở theo yêu cầu đề ra sẽ được lập trình, chương trình sẽ được nạp vào bộ nhớ.

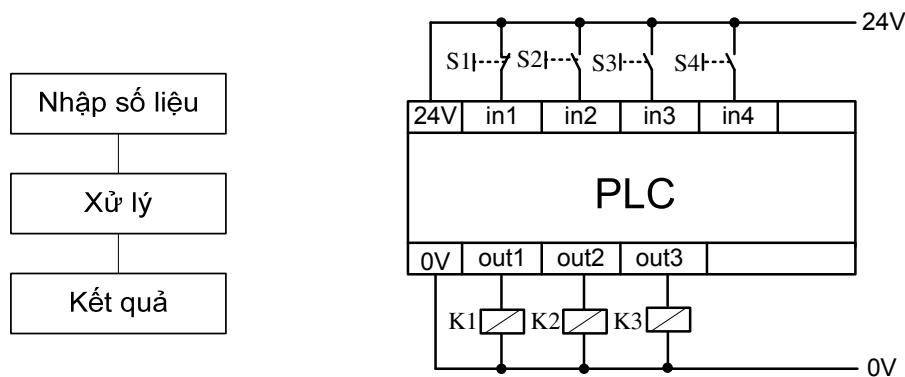
Bây giờ giả thiết rằng nhiệm vụ điều khiển sẽ thay đổi. Hệ thống ba máy bơm vẫn giữ nguyên, nhưng trình tự được thực hiện như sau: chỉ đóng được hai trong ba máy bơm hoặc mỗi máy bơm có thể hoạt động một cách độc lập. Như vậy theo yêu cầu mới đổi với hệ thống điều khiển bằng relay điện phải thiết kế lại mạch điều khiển, sơ đồ lắp ráp phải thực hiện lại hoàn toàn mới. Sơ đồ mạch điều khiển biểu diễn như hình 2.4.

Như vậy mạch điều khiển sẽ thay đổi rất nhiều nhưng phần tử đưa tín hiệu vào và ra vẫn giữ nguyên, chi phí cho nhiệm vụ mới sẽ cao hơn.

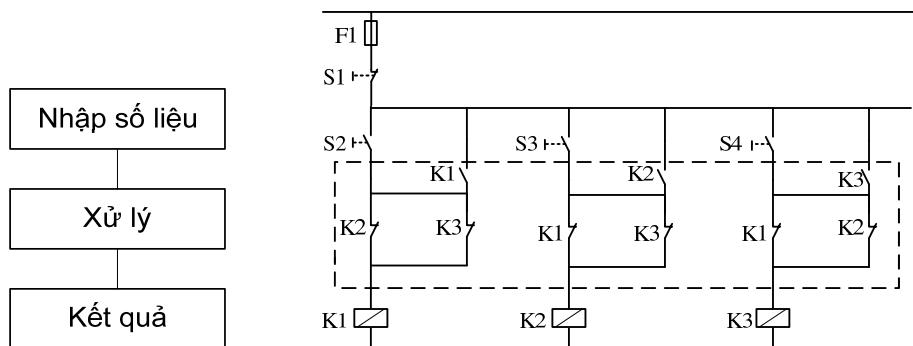
Nếu ta thay đổi hệ điều khiển trên bằng hệ điều khiển lập trình PLC, khi nhiệm vụ điều khiển thay đổi thì thực hiện sẽ nhanh hơn và đơn giản hơn bằng cách thay đổi lại chương trình.



Hình 2.2: Sơ đồ mạch được chuyển thành chương trình trong PLC



Hình 2.3: Sơ đồ kết nối với PLC



Hình 2.4: Sơ đồ mạch điều khiển 3 động cơ đã được thay đổi

Hệ điều khiển lập trình PLC có những ưu điểm sau:

- Thích ứng với những nhiệm vụ điều khiển khác nhau.
- Khả năng thay đổi đơn giản trong quá trình đưa thiết bị vào sử dụng.
- Tiết kiệm không gian lắp đặt.
- Tiết kiệm thời gian trong quá trình mở rộng và phát triển nhiệm vụ điều khiển bằng cách copy các chương trình.
- Các thiết bị điều khiển theo chuẩn.
- Không cần các tiếp điểm.
- V.v...

Hệ thống điều khiển lập trình PLC được sử rộng rất rộng rãi trong các ngành khác nhau:

- Điều khiển thang máy.
- Điều khiển các quá trình sản xuất khác nhau: sản suất bia, sản xuất xi măng v.v ....

- Hệ thống rửa ô tô tự động.
- Thiết bị khai thác .
- Thiết bị đóng gói bao bì, tự động mạ và tráng kẽm v.v ...
- Thiết bị sấy.
- ...

### 2.3 Cấu trúc của một PLC

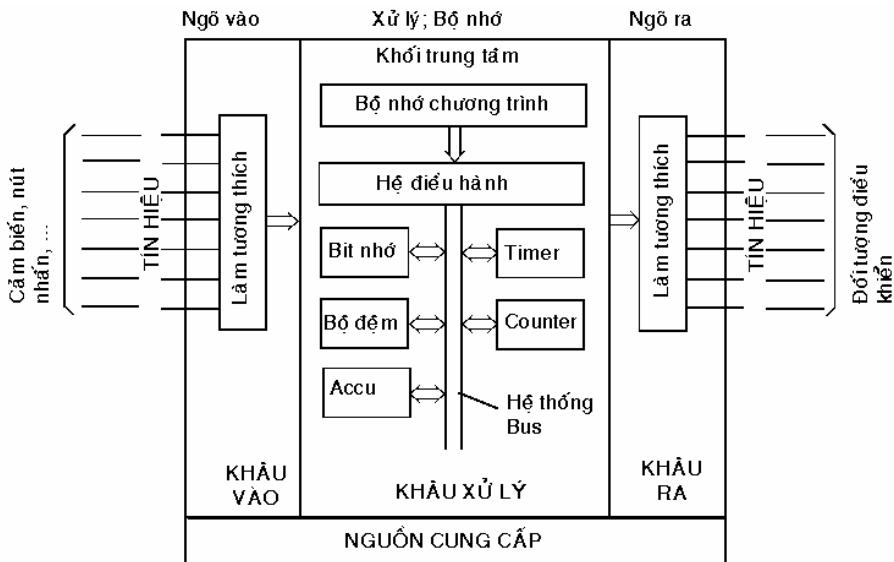
Các bộ điều khiển PLC được sản xuất theo dòng sản phẩm. Khi mới xuất xưởng, chúng chưa có một chương trình cho một ứng dụng nào cả. Tất cả các cổng logic cơ bản, chức năng nhớ, timer, counter .v.v... được nhà chế tạo tích hợp trong chúng và được kết nối với nhau bằng chương trình được viết bởi người dùng cho một nhiệm vụ điều khiển cụ thể nào đó. Bộ điều khiển PLC có nhiều loại khác nhau và được phân biệt với nhau qua các thành phần sau:

- Các ngõ vào và ra
- Dung lượng nhớ
- Bộ đếm (counter)
- Bộ định thời (timer)
- Bit nhớ
- Các chức năng đặc biệt
- Tốc độ xử lý
- Loại xử lý chương trình.
- Khả năng truyền thông.

Các bộ điều khiển lớn thì các thành phần trên được lắp thành các modul riêng. Đối với các bộ điều khiển nhỏ, chúng được tích hợp trong bộ điều khiển. Các bộ điều khiển nhỏ này có số lượng ngõ vào/ra cho trước cố định.

Bộ điều khiển được cung cấp tín hiệu bởi các tín hiệu từ các cảm biến ở ngõ vào của nó. Tín hiệu này được xử lý tiếp tục thông qua chương trình điều khiển đặt trong bộ nhớ chương trình. Kết quả xử lý được đưa ra ngõ ra để đến đối tượng điều khiển hay khâu điều khiển ở dạng tín hiệu.

Cấu trúc của một PLC có thể được mô tả như hình vẽ sau:



Hình 2.5: Cấu trúc chung của bộ điều khiển lập trình PLC

#### \* **Bộ nhớ chương trình**

Bộ nhớ chương trình trong PLC là một bộ nhớ điện tử đặc biệt có thể đọc được. Nếu sử dụng bộ nhớ đọc-ghi được (RAM), thì nội dung của nó luôn luôn được thay đổi ví dụ như trong trường hợp vận hành điều khiển. Trong trường hợp điện áp nguồn bị mất thì nội dung trong RAM có thể vẫn được giữ lại nếu như có sử dụng Pin dự phòng.

Nếu chương trình điều khiển làm việc ổn định, hợp lý, nó có thể được nạp vào một bộ nhớ cố định, ví dụ như EPROM, EEPROM. Nội dung chương trình ở EPROM có thể bị xóa bằng tia cực tím.

#### \* **Hệ điều hành**

Sau khi bật nguồn cung cấp cho bộ điều khiển, hệ điều hành của nó sẽ đặt các counter, timer, dữ liệu và bit nhớ với thuộc tính non-retentive (không được nhớ bởi Pin dự phòng) cũng như ACCU về 0.

Để xử lý chương trình, hệ điều hành đọc từng dòng chương trình từ đầu đến cuối. Tương ứng hệ điều hành thực hiện chương trình theo các câu lệnh.

#### \* **Bit nhớ (Bit memory)**

Các bit memory là các phần tử nhớ, mà hệ điều hành ghi nhớ trạng thái tín hiệu.

#### \* **Bộ đếm (Process Image)**

Bộ đếm là một vùng nhớ, mà hệ điều hành ghi nhớ các trạng thái tín hiệu ở các ngõ vào ra nhị phân.

### \* Accumulator

Accumulator là một bộ nhớ trung gian mà qua nó timer hay counter được nạp vào hay thực hiện các phép toán số học.

### \* Counter, Timer

Timer và counter cũng là các vùng nhớ, hệ điều hành ghi nhớ các giá trị đếm trong nó.

### \* Hệ thống Bus

Bộ nhớ chương trình, hệ điều hành và các modul ngoại vi (các ngõ vào và ngõ ra) được kết nối với PLC thông qua Bus nối. Một Bus bao gồm các dây dẫn mà các dữ liệu được trao đổi. Hệ điều hành tổ chức việc truyền dữ liệu trên các dây dẫn này.

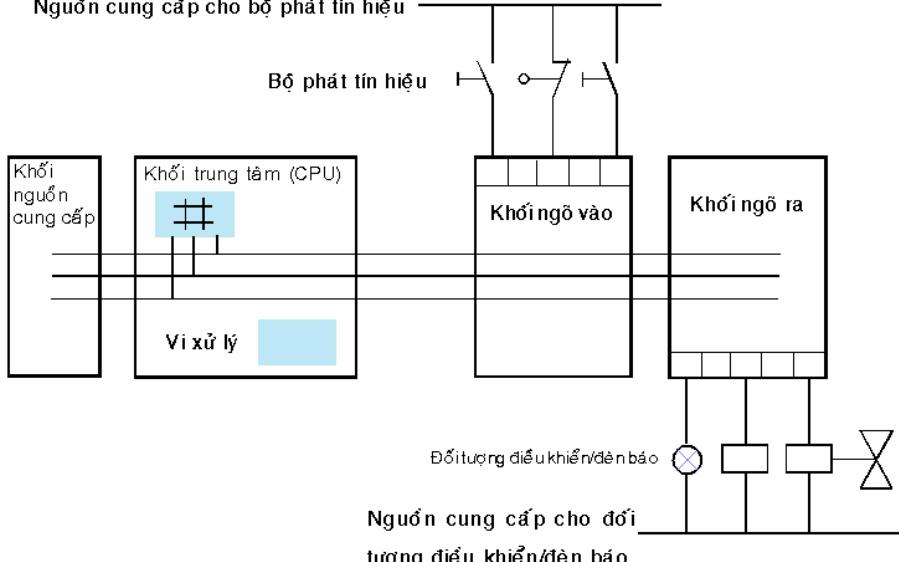
## 2.4 Các khối của PLC

Các khối khác nhau của một PLC được cho như hình 2.6.

### 2.4.1 Khối nguồn cung cấp

Khối nguồn có nhiệm vụ biến đổi điện áp lưới (110V hay 220V ) thành điện áp thấp hơn cung cấp cho các khối của thiết bị tự động. Điện áp này là 24VDC. Các điện áp cho cảm biến, thiết bị điều chỉnh và các đèn báo nằm trong khoảng (24...220V) có thể được cung cấp thêm từ các nguồn phụ ví dụ như biến áp.

Nguồn cung cấp cho bộ phát tín hiệu



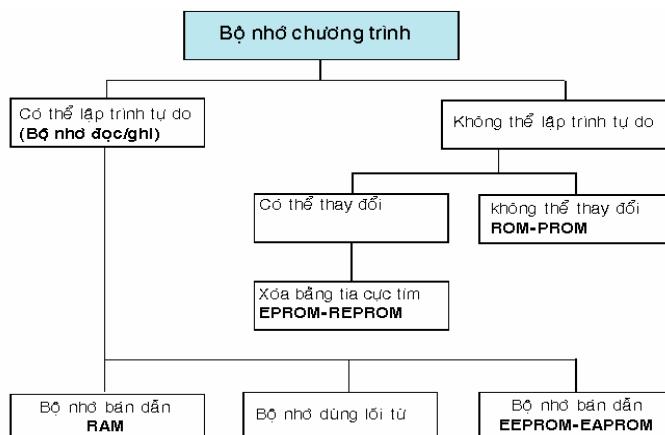
Hình 2.6:Các khối trong một PLC

## 2.4.2 Bộ nhớ chương trình

Các phần tử nhớ là các linh kiện mà thông tin có thể được lưu trữ (được nhớ) trong nó ở dạng tín hiệu nhị phân. Trong PLC các bộ nhớ bán dẫn được sử dụng làm bộ nhớ chương trình. Một bộ nhớ bao gồm 512, 1024, 2048 . . . phần tử nhớ, các phần tử nhớ này sắp đặt theo các địa chỉ từ 0 tới 511, 1023 hoặc 2047 . . . Thông thường số lượng của các phần tử nhớ trong một bộ nhớ cho biết dung lượng của nó là bao nhiêu kilobyte (1kB = 1024 byte). Trong mỗi ô nhớ có thể mô tả một câu lệnh điều khiển nhờ thiết bị lập trình. Mỗi phần tử nhị phân của một ô nhớ có thể có trạng thái tín hiệu "0" hoặc "1". Sơ đồ của một bộ nhớ chương trình được cho như hình 2.7.

### \* Bộ nhớ đọc-ghi RAM (random-access memory)

Bộ nhớ ghi-đọc có 1 số lượng các ô nhớ xác định. Mỗi ô nhớ có 1 dung lượng nhớ cố định và nó chỉ tiếp nhận 1 lượng thông tin nhất định. Các ô nhớ được ký hiệu bằng các địa chỉ riêng của nó. Bộ nhớ này chứa các chương trình còn sửa đổi hoặc các dữ liệu, kết quả tạm thời trong quá trình tính toán, lập trình. Đặc điểm của loại này là dữ liệu sẽ mất đi khi hệ thống mất điện. RAM được hình dung như một tủ chứa có nhiều ngăn kéo. Mỗi ngăn kéo được đánh số một địa chỉ và người ta có thể cất vào hoặc lấy các dữ liệu ra.



Hình 2.7: Sơ đồ một bộ nhớ chương trình

### \* Bộ nhớ cố định ROM (read-only memory)

Bộ nhớ cố định (ROM) chứa các thông tin không có khả năng xóa được và không thể thay đổi được. Các thông tin này do các nhà sản xuất viết ra và không thể thay đổi được. Chương trình trong bộ nhớ ROM có nhiệm vụ sau:

- Điều khiển và kiểm tra các chức năng hoạt động của CPU. Được gọi là hệ điều hành.
- Dịch ngôn ngữ lập trình thành ngôn ngữ máy.

Một ROM có thể so sánh với một quyển sách. Trong đó nó chứa các thông tin cố định, không thể thay đổi được và ta chỉ đọc các thông tin đó mà thôi. Đặc điểm của loại này là dữ liệu vẫn tồn tại khi mất điện.

#### \* ***EPROM (eraseable read-only memory)***

EPROM là một bộ nhớ cố định có thể lập trình và xóa được. Nội dung của EPROM có thể xóa bằng tia cực tím và có thể lập trình lại.

#### \* ***EEPROM (electrically erasable read-only memory)***

EEPROM là bộ nhớ cố định có thể lập trình và xóa bằng điện. Mỗi ô nhớ trong EEPROM cho phép lập trình và xóa bằng điện.

### 2.4.3 Khối trung tâm (CPU)

Khối CPU là loại khối có chứa bộ vi xử lý, hệ điều hành, bộ nhớ, các bộ thời gian, bộ đếm, cổng truyền thông ... và có thể còn có một vài cổng vào ra số. Các cổng vào ra số có trên CPU được gọi là cổng vào/ra onboard.

### 2.4.4 Khối vào

Các ngõ vào của khối này sẽ được kết nối với các bộ chuyển đổi tín hiệu và biến đổi các tín hiệu này thành tín hiệu phù hợp với tín hiệu xử lý của CPU. Dựa vào loại tín hiệu vào sẽ có các khối ngõ vào tương ứng. Gồm có hai loại khối vào cơ bản sau:

- ***Khối vào số (DI: Digital Input):***

Các ngõ vào của khối này được kết nối với các bộ chuyển đổi tạo ra tín hiệu nhị phân như nút nhấn, công tắc, cảm biến tạo tín hiệu nhị phân .v.v... Do tín hiệu tại ngõ vào có thể có mức logic tương ứng với các điện áp khác nhau, do đó khi sử dụng cần phải chú ý đến điện áp cần thiết cung cấp cho khối vào phải phù hợp với điện áp tương ứng mà bộ chuyển đổi tín hiệu nhị phân tạo ra.

Ví dụ: Các nút nhấn, công tắc được nối với nguồn 24VDC thì yêu cầu phải sử dụng khối vào có nguồn cung cấp cho nó là 24VDC.

- ***Khối vào tương tự (AI: Analog Input):***

Khối này có nhiệm vụ biến đổi tín hiệu tương tự (hay còn gọi là tín hiệu analog) thành tín hiệu số. Các ngõ vào của khối này được kết nối với các bộ chuyển đổi tạo ra tín hiệu analog như cảm biến nhiệt độ (Thermocouple), cảm biến lưu lượng, ngõ ra analog của biến tần .v.v... Khi sử dụng các khối vào analog cần phải chú ý đến loại tín hiệu analog được tạo ra từ các bộ chuyển đổi (cảm biến)

Ví dụ: Các cảm biến tạo ra tín hiệu analog là dòng điện (4..20 mA) thì phải sử dụng ngõ vào analog là loại nhận tín hiệu dòng điện (4..20 mA). Nếu cảm biến tạo ra tín hiệu analog là điện áp (0..5V) thì phải sử dụng ngõ vào analog nhận tín hiệu là điện áp (0..5V).

### 2.4.5 Khối ra

Khối này có nhiệm vụ khuếch đại các tín hiệu sau xử lý của CPU (được gởi đến vùng đệm ra) cung cấp cho đối tượng điều khiển là cuộn dây, đèn báo, van từ .v.v.. Tùy thuộc vào đối tượng điều khiển nhận tín hiệu dạng nào mà sẽ có các khối ra tương ứng. Gồm có hai loại khối ra tiêu biểu:

- ***Khối ra số (DO: Digital Output):***

Các ngõ ra của khối này được kết nối với các đối tượng điều khiển nhận tín hiệu nhị phân như đèn báo, cuộn dây relay .v.v... Vì đối tượng điều khiển nhận tín hiệu nhị phân sử dụng nhiều cấp điện áp khác nhau nên khi sử dụng các khối ra số cần phải chú ý đến điện áp cung cấp cho nó có phù hợp với điện áp cung cấp cho đối tượng điều khiển hay không. Theo loại điện áp sử dụng, ngõ ra số được phân thành hai loại:

- ***Điện áp một chiều (DC: Direct Current):*** Gồm có hai loại ngõ ra là Transistor và relay. Thông thường trong công nghiệp điện áp một chiều được sử dụng là 24V.
- ***Điện áp xoay chiều (AC: Alternative Current):*** Gồm có hai loại ngõ ra là relay và TRIAC.

- ***Khối ra tương tự (AO: Analog Output):***

Khối này có nhiệm vụ biến đổi tín hiệu số được gởi từ CPU đến đối tượng điều khiển thành tín hiệu tương tự. Các ngõ ra của khối này được kết nối với các đối tượng điều khiển nhận tín hiệu tương tự như ngõ vào analog của biến tần, van tỷ lệ, .v.v... Khi sử dụng các ngõ ra tương tự cần chú ý đến loại tín hiệu tương tự cung cấp cho đối tượng điều khiển có phù hợp với tín hiệu tương tự mà đối tượng điều khiển cần nhận hay không.

**Ví dụ:** Ngõ vào analog của biến tần nhận tín hiệu là điện áp (0..10V) thì nhất thiết phải sử dụng ngõ ra tương tự tạo ra tín hiệu analog là điện áp (0..10V).

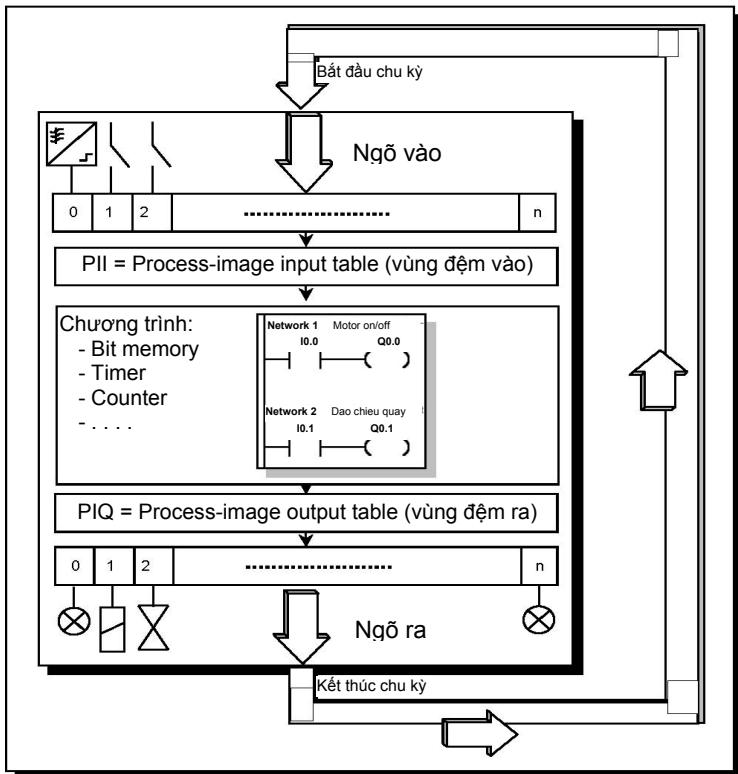
### 2.4.6 Các khối đặc biệt

Ngoài ra còn có một số khối khác đảm nhận các chức năng đặc biệt như xử lý truyền thông, thực hiện các chức năng đặc biệt như: điều khiển vị trí, điều khiển vòng kín, đếm tốc độ cao .v.v...

Tùy thuộc vào từng loại PLC mà các khối trên có thể ở các dạng module riêng hoặc được tích hợp chung trong khối xử lý trung tâm (CPU).

## 2.5 Phương thức thực hiện chương trình trong PLC

Hình vẽ minh họa việc xử lý chương trình trong CPU được cho như hình 2.8



Hình 2.8: Chu kỳ quét trong PLC

PLC thực hiện chương trình theo chu trình lắp. Mỗi vòng lắp được gọi là vòng quét (scan). Mỗi vòng quét được bắt đầu bằng giai đoạn chuyển dữ liệu từ các cổng vào số tới vùng bộ đệm ảo ngõ vào (I), tiếp theo là giai đoạn thực hiện chương trình. Trong từng dòng quét, chương trình được thực hiện từ lệnh đầu tiên đến lệnh kết thúc. Sau giai đoạn thực hiện chương trình là giai đoạn chuyển các nội dung của bộ đệm ảo ngõ ra (Q) tới các cổng ra số. Vòng quét được kết thúc bằng giai đoạn truyền thông nội bộ và kiểm tra lỗi.

Thời gian cần thiết để PLC thực hiện được một vòng quét gọi là thời gian vòng quét (Scan time). Thời gian vòng quét không cố định, tức là không phải vòng quét nào cũng được thực hiện trong một khoảng thời gian như nhau. Có vòng quét thực hiện lâu, có vòng quét thực hiện nhanh tùy thuộc vào số lệnh trong chương trình được thực hiện, vào khối lượng dữ liệu truyền thông ... trong vòng quét đó.

Như vậy giữa việc đọc dữ liệu từ đối tượng để xử lý, tính toán và việc gửi tín hiệu điều khiển tới đối tượng có một khoảng thời gian trễ đúng bằng thời gian vòng quét. Nói cách khác, thời gian vòng quét quyết định tính thời gian thực của chương trình điều khiển trong PLC. Thời gian quét càng ngắn, tính thời gian thực của chương trình càng cao.

Tại thời điểm thực hiện lệnh vào/ra, thông thường lệnh không làm việc trực tiếp với cổng vào/ra mà chỉ thông qua bộ đệm ảo của cổng trong vùng

nhớ tham số. Việc truyền thông giữa bộ đệm ảo với ngoại vi do hệ điều hành CPU quản lý. Ở một số module CPU, khi gấp lệnh vào/ra ngay lập tức, hệ thống sẽ cho dừng mọi công việc khác, ngay cả chương trình xử lý ngắn, để thực hiện lệnh trực tiếp với cổng vào/ra.

### 3 Cảm biến và cơ cấu chấp hành trong điều khiển logic.



Chương này nhằm giúp cho bạn đọc tìm hiểu sơ lược về một số các thiết bị ngoại vi sẽ được kết nối với các ngõ vào ra số của PLC và một số ký hiệu về các thiết bị ngoại vi.

#### 3.1 Cảm biến

##### 3.1.1 Giới thiệu

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

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

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

##### 3.1.2 Nối dây cho cảm biến

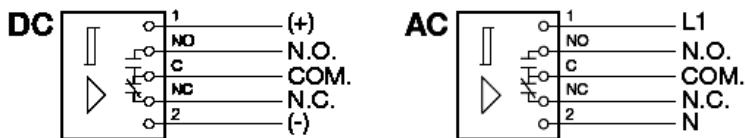
Khi một cảm biến phát hiện một sự thay đổi trạng thái logic thì nó phải truyền trạng thái thay đổi này đến PLC. Tiêu biểu là việc đóng hoặc ngắt dòng điện hay điện áp. Trong một vài trường hợp, ngõ ra của cảm biến sử dụng để đóng mạch trực tiếp cho tải mà không thông qua PLC. Các ngõ ra tiêu biểu của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

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

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

### 3.1.2.1 Switch

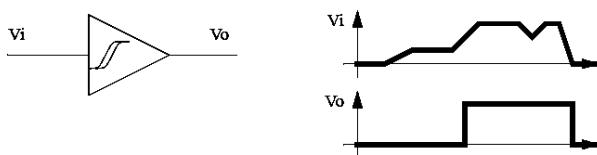
Một ví dụ đơn giản nhất của các ngõ ra cảm biến switch và relay được cho như hình 3.1.



Hình 3.1: Cảm biến có ngõ ra là relay sử dụng nguồn DC và AC .

### 3.1.2.2 Ngõ ra TTL

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



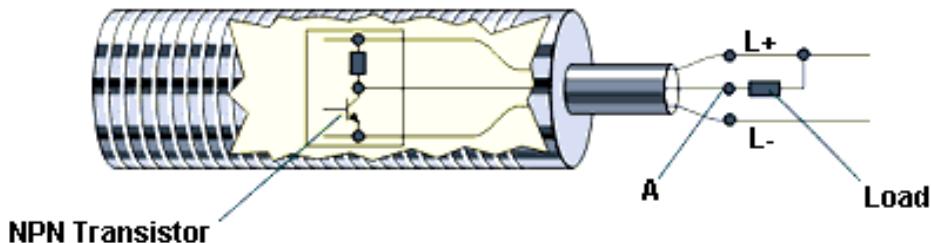
Hình 3.2: Mạch Schmitt trigger

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

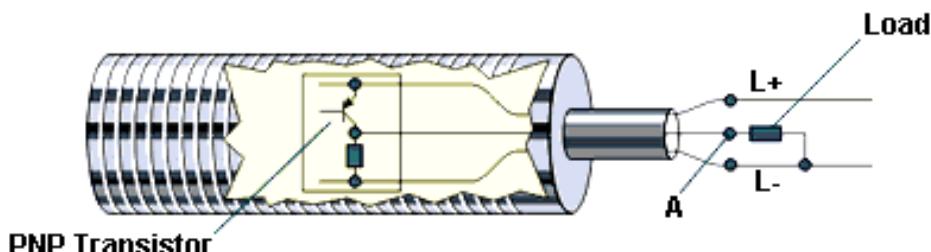
### 3.1.2.3 Ngõ ra Sinking/Sourcing

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

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



Hình 3.3: Cảm biến NPN (cảm biến “rút dòng”).



Hình 3.4: Cảm biến PNP (cảm biến “sourcing”)

Để cảm biến hoạt động cần phải có nguồn cung cấp (chân L+ và L-). Khi cảm biến phát hiện đối tượng thì có điện áp tại cực B của transistor NPN, transistor chuyển sang trạng thái dẫn và cho phép dòng chảy vào cảm biến xuống mass (chân L-).

Khi không phát hiện đối tượng thì điện áp tại cực B của transistor ở mức thấp (0V), transistor không dẫn. Điều này có nghĩa ngõ ra NPN sẽ không có dòng vào/ra.

Các cảm biến “sourcing” thì ngược với các cảm biến “sinking”. Nó sử dụng transistor PNP (hình 3.4). Khi cảm biến không được kích hoạt thì cực B của transistor ở giá trị L+, và transistor ở trạng thái ngưng dẫn. Khi cảm biến được kích hoạt thì cực B transistor sẽ được đặt ở 0V, và transistor cho phép dòng điện chảy từ cảm biến ra ngoài thiết bị được kết nối.

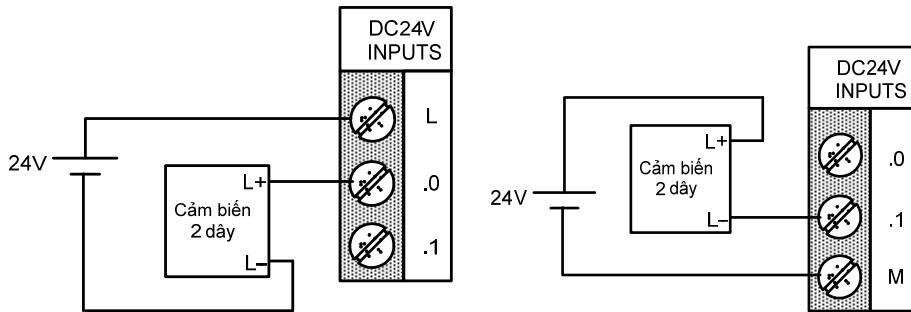
Hầu hết các cảm biến NPN/PNP có khả năng dòng đến vài ampere, và chúng có thể được sử dụng để nối trực tiếp với tải (luôn luôn kiểm tra tay để biết chính xác dòng điện và điện áp định mức).

**Chú ý:** Cần phải nhớ kiểm tra dòng điện và điện áp định mức đối với các cảm biến. Khi nối dây các cảm biến cần chú ý đến các chân nguồn. Thường các

chân nguồn có ký hiệu là L+ và COM(chân chung), nhưng đôi khi không có chân COM mà có chân L-. Trong trường hợp này L- là chân chung.

Khi kết nối các cảm biến “sourcing” với các ngõ PLC, thì cần chú ý phải sử dụng các modul ngõ vào loại “sinking”. Thông thường các ngõ vào PLC thường là loại “sinking”.

Trong ứng dụng với PLC, để giảm lượng dây nối, thì các cảm biến hai dây thường được sử dụng. Ví dụ về sơ đồ nối dây các cảm biến sử dụng nguồn 24VDC với PLC được chỉ như hình 3.5. Cảm biến hai dây có thể được sử dụng cho cả hai loại ngõ vào sourcing hoặc ngõ vào sinking của PLC.



a. Ngõ vào PLC loại sourcing

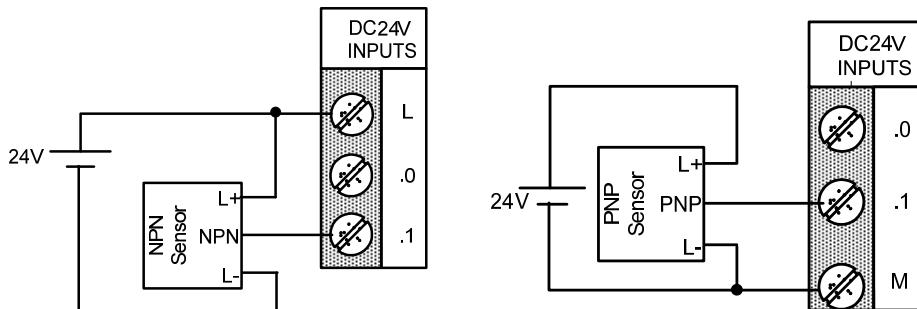
b. Ngõ vào PLC loại sinking

Hình 3.5: Kết nối cảm biến 2 dây với ngõ vào PLC.

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

Trong các bản vẽ thì các chân của các cảm biến NPN và PNP có ký hiệu về màu sắc như sau: dây màu nâu là L+, dây màu xanh dương là L- và ngõ ra thì màu trắng đối với sinking và màu đen đối với sourcing.

Cần lưu ý là khi tiếp điểm trong cảm biến “sinking” đóng thì ngõ ra được nối với COM hoặc L-, tiếp điểm trong sourcing đóng thì ngõ ra nối với L+.



a. Ngõ vào PLC loại sourcing

b. Ngõ vào PLC loại sinking

Hình 3.6: Kết nối cảm biến NPN và PNP dây với ngõ vào PLC.

### 3.1.2.4 Ngõ ra Solid state relay

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

### 3.1.3 Phát hiện đối tượng

Có hai cách cơ bản để phát hiện đối tượng: tiếp xúc và tiếp cận (proximity).

Tiết xúc có nghĩa là tiếp điểm cơ khí cần một lực tác động giữa cảm biến và đối tượng.

Tiết cận để chỉ báo rằng một đối tượng đang ở gần nhưng không yêu cầu tiếp xúc.

Các phần sau đây sẽ minh họa các kiểu khác nhau của các cảm biến để phát hiện sự hiện diện của các đối tượng. Phần này không đi sâu vào các cảm biến mà chỉ mô tả các nguyên lý trong lĩnh vực ứng dụng.

#### 3.1.3.1 Chuyển mạch tiếp xúc

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

#### 3.1.3.2 Reed Switches

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



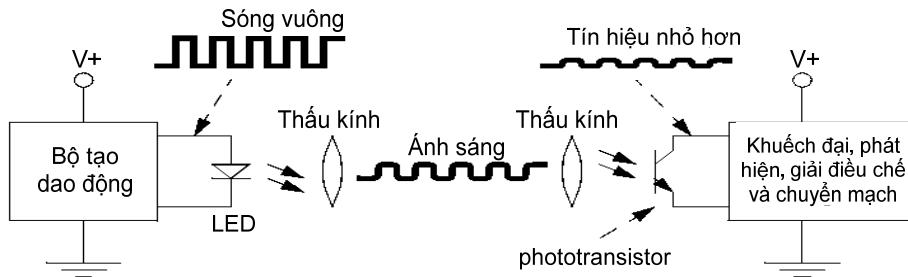
Hình 3.7: Reed switch

#### 3.1.3.3 Cảm biến quang (Optical Sensor)

Cảm biến ánh sáng được sử dụng gần một thẻ kỷ qua. Nguyên thủy là tê bào quang được sử dụng cho các ứng dụng như đọc các track âm thanh trên các hình ảnh chuyển động. Nhưng các cảm biến quang hiện đại thì phức tạp hơn nhiều.

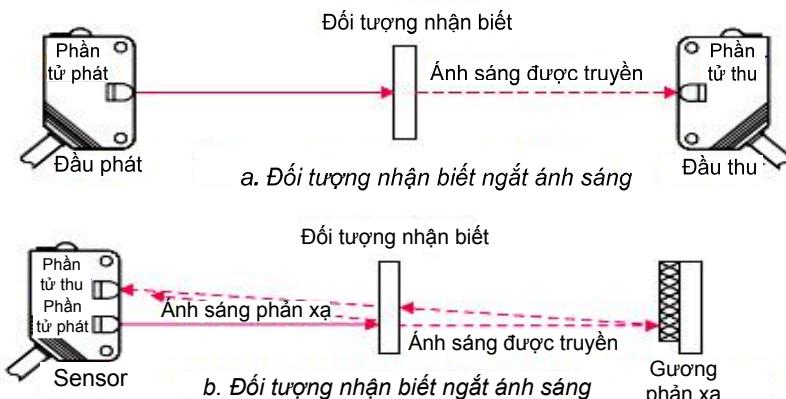
Các cảm biến quang yêu cầu có cả hai bộ phận là nguồn sáng (phát) và đầu thu (detector). Các đầu phát (emitter) sẽ phát ra các tia sáng trong vùng phổ nhìn thấy và không nhìn thấy được sử dụng LED và diode laser. Đầu thu có cấu tạo là các diode quang (photodiode) hoặc transistor quang (phototransistor). Đầu phát và đầu thu được đặt vào vị trí để đối tượng khi

xuất hiện sẽ cắt ngang hoặc phản xạ lại tia sáng. Cảm biến quang đơn giản cho ở hình 3.8.



Hình 3.8: Cảm biến quang cơ bản

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





Hình 13.9: Các loại cảm biến quang khác nhau

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

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

### 3.1.3.4 Cảm biến điện dung (Capacitive Sensor)

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

Công thức biểu diễn mối quan hệ điện dung:

$$C = \frac{\epsilon \cdot A}{d} \quad \text{với} \quad C: \text{Điện dung (Farads)}$$

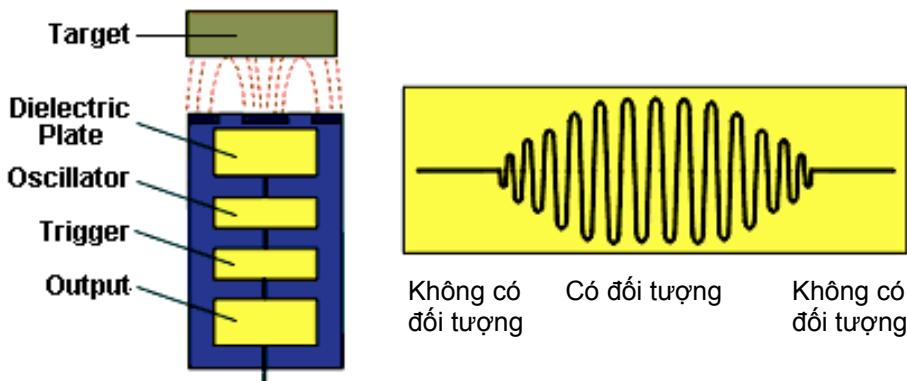
$\epsilon$ : Hằng số điện môi

A: Diện tích bản cực

D: Khoảng cách giữa các bản cực.

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

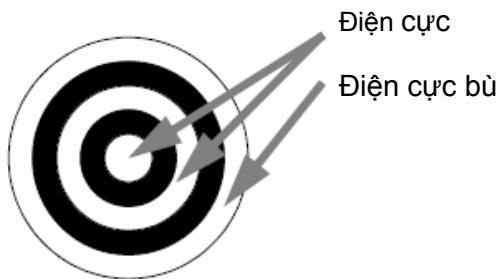
Bề mặt của cảm biến điện dung được hình thành bởi hai điện cực kim loại đồng tâm của một tụ điện. Khi một đối tượng đến gần bề mặt nhận biết nó đi vào vùng điện trường của các điện cực và thay đổi điện dung trong mạch dao động. Kết quả là bộ tạo dao động bắt đầu dao động. Mạch trigger đọc biên độ của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



Hình 3.10: Cảm biến điện dung

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

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



Hình 3.11: Bề mặt nhận biết của cảm biến điện dung

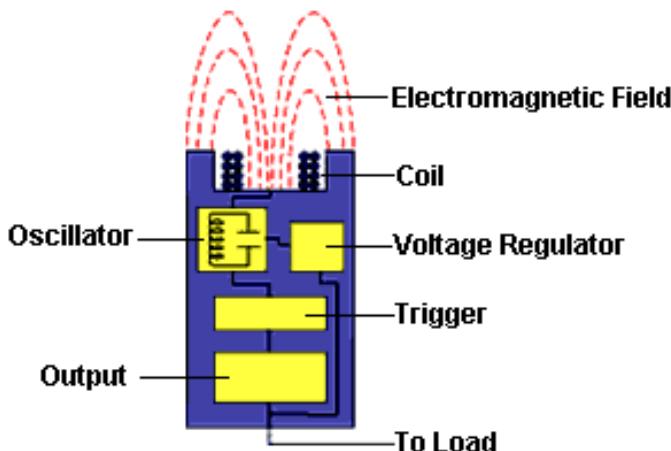
Phạm vi và độ chính xác của các cảm biến được xác định bởi kích thước của chúng. Các cảm biến lớn có thể có đường kính vài centimeter. Cái nhỏ có đường kính nhỏ hơn một centimeter và có phạm vi nhỏ hơn nhưng chính xác hơn.

### 3.1.3.5 Cảm biến điện cảm (Inductive Sensor)

Các cảm biến điện cảm sử dụng dòng điện cảm ứng để phát hiện đối tượng là kim loại. Cảm biến điện cảm sử dụng một cuộn dây để tạo một từ trường tần số cao được cho ở hình 3.12. Nếu có một đối tượng là kim loại đến gần làm thay đổi từ trường, thì sẽ có dòng chảy vào đối tượng. Dòng chảy này tạo ra một từ trường mới ngược với từ trường ban đầu. Kết quả là nó làm thay

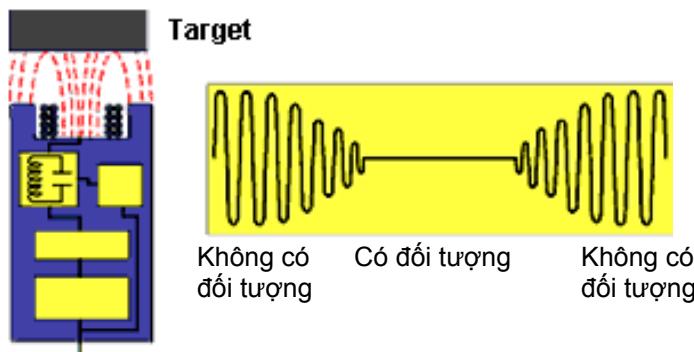
đổi độ tự cảm của cuộn dây trong cảm biến. Bằng cách đo độ tự cảm, cảm biến có thể xác định một đối tượng kim loại đến gần.

Các cảm biến này sẽ phát hiện bất kỳ kim loại nào, khi cần phát hiện các loại kim loại thì các cảm biến đa kim loại thường được sử dụng.



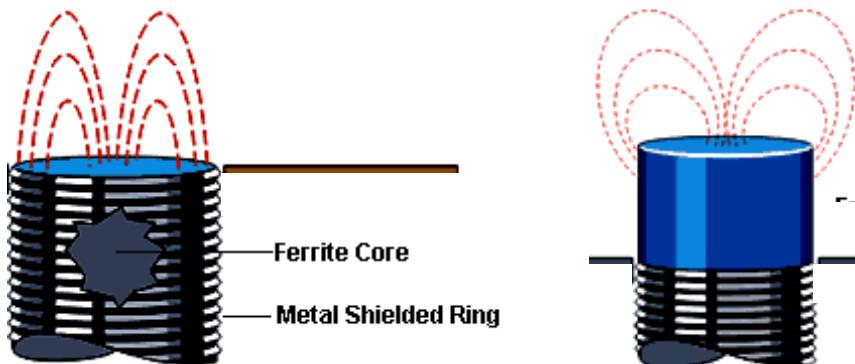
Hình 3.12: Cảm biến tiếp cận điện cảm

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



Hình 3.13: Cảm biến tiếp cận điện cảm

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



Hình 3.14: Cảm biến bọc và không bọc vỏ kim loại

### 3.1.3.6 Cảm biến siêu âm (Ultrasonic sensor)

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

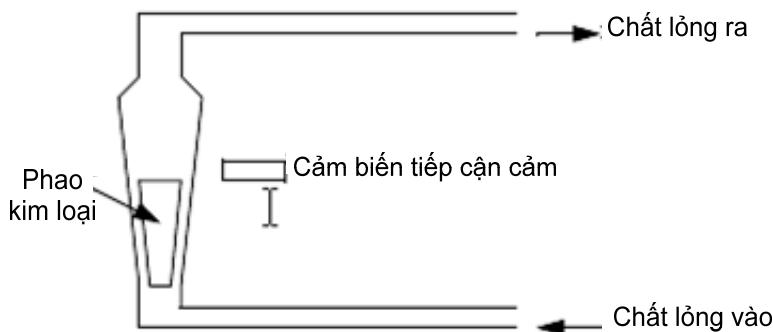
- Tĩnh điện (electrostatic): Sử dụng hiệu ứng điện dung. Phạm vi lớn và băng thông rộng hơn nhưng độ nhạy cao hơn với đối tượng ẩm ướt.
- Áp điện (piezoelectric): Dựa vào phần tử áp điện thạch anh.

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

### 3.1.3.7 Hiệu ứng Hall (Hall Effect)

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

### 3.1.3.8 Lưu lượng (Fluid Flow)



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

### 3.1.4 Tóm tắt

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

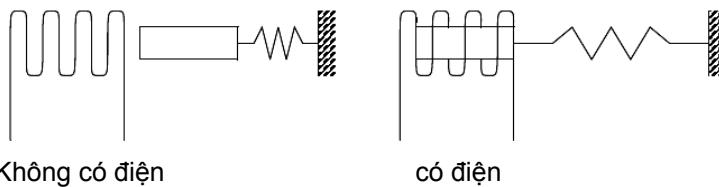
## 3.2 Cơ cấu chấp hành

### 3.2.1 Giới thiệu

Cơ cấu chấp hành được sử dụng để biến đổi năng lượng điện thành chuyển động cơ học.

### 3.2.2 Solenoid

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

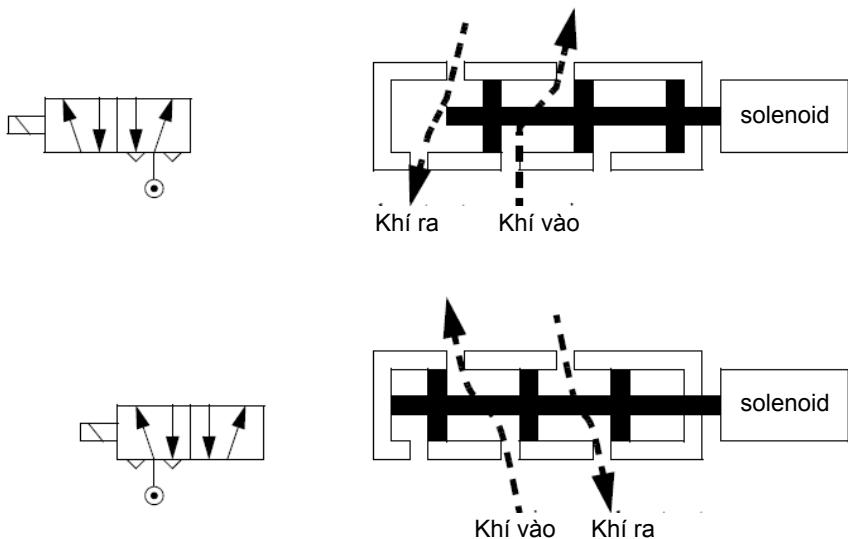


Hình 3.16: Solenoid

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

### 3.2.3 Van điều khiển (VALVE)

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



Hình 3.17: Một solenoid điều khiển van 5 cửa 2 vị trí

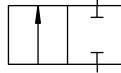
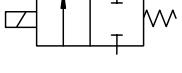
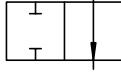
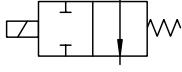
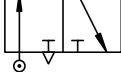
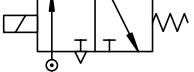
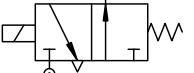
Các loại van được liệt kê dưới đây. Theo tiêu chuẩn, thuật ngữ ‘n-cửa’ (n-cửa) để chỉ định số lượng kết nối các ngõ vào và ra của van. Trong một vài trường hợp có cửa để xả khí ra. Việc thiết kế thường đóng/thường mở cho biết điều kiện van khi mất nguồn cấp.

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

được sử dụng để làm chuyển hướng dòng chảy, hoặc chọn nguồn qua lại.

- Van 4 cửa, 2 vị trí (van 4/2):** Các van này có 4 cửa, 1 vào, 2 ra và 1 cửa xả khí. Khi có nguồn van nối các cửa vào với các cửa ra và ngược lại. Các van này được sử dụng với các cylinder tác động kép.
- Van 5 cửa, 2 vị trí (van 5/2):** Các van này có 5 cửa, 1 vào, 2 ra và 2 cửa xả khí.
- Van 4 cửa, 3 vị trí (van 4/3):** Các van này có 4 cửa, 1 vào, 2 ra và 1 xả. Ở trạng thái bình thường (không có nguồn năng lượng) thì các cửa vào/ra đều bị chặn. Van này được sử dụng để điều khiển vị trí các cylinder.
- Van 5 cửa, 3 vị trí (van 5/3):** Van này có 5 cửa, 1 vào, 2 ra và 2 cửa xả. Tương tự như van 4/3, van này được sử dụng để điều khiển vị trí các cylinder.

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

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

Van 4 cửa, 2 vị trí		
Van 5 cửa, 2 vị trí		
Van 4 cửa, 3 vị trí		
Van 5 cửa, 3 vị trí		

Hình 3.18 Ký hiệu các van điều khiển bằng khí và solenoid

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

- Kích thước ống: Cửa vào và ra theo tiêu chuẩn NPT (national pipe thread).
  - Tốc độ dòng chảy: Tốc độ dòng chảy cực đại thường được cung cấp cho các van thủy lực.
  - Áp suất hoạt động: Áp suất hoạt động cực đại phải được chỉ báo. Một vài van có yêu cầu áp suất tối thiểu để hoạt động.
  - Nguồn điện: Các cuộn dây solenoid yêu cầu được cung cấp một điện áp và dòng điện cố định (AC hoặc DC).
  - Thời gian đáp ứng: Đây là thời gian để van đóng/mở hoàn toàn. Thời gian tiêu biểu cho các van nằm trong phạm vi từ 5ms đến 150ms.
  - Vỏ bọc: Vỏ bọc cho các van được xếp theo loại:
    - Loại 1 hoặc 2: Sử dụng trong nhà, yêu cầu bảo vệ chống nước.
    - Loại 3: Sử dụng ngoài trời, chống bụi bẩn và mưa gió.
    - Loại 3R hoặc 3S hoặc 4: Chống nước và bụi.
    - Loại 4X: Chống nước, bụi và sự ăn mòn.

### 3.2.4 Xy lanh (CYLINDER)

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

Công thức tính lực:

$$F = P \cdot A$$

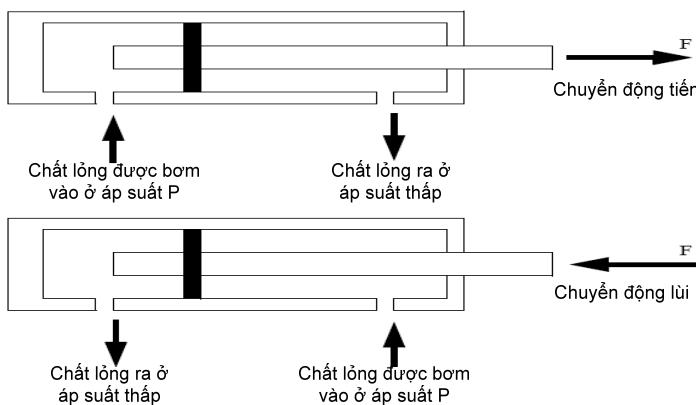
$$P = \frac{F}{A}$$

Với

P: Áp suất thủy lực

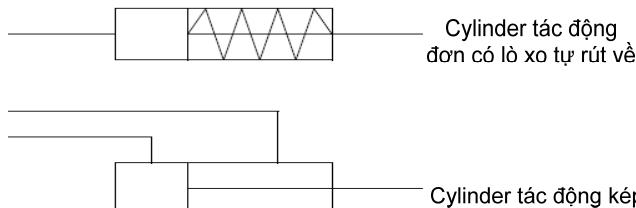
F: Lực đẩy piston

A: Diện tích piston



Hình 3.19 Mặt cắt của một cylinder thủy lực

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



Hình 3.20 cylinder tác động đơn và cylinder tác động kép

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

### 3.2.5 Động cơ

Động cơ là cơ cấu chấp hành thông thường, nhưng đối với ứng dụng cho điều khiển nhị phân thì đặc điểm của nó không quan trọng. Điều khiển logic tiêu biểu của các động cơ là đóng/cắt điện cho nó. Các động cơ có dòng

điện nhỏ có thể đấu trực tiếp vào các ngõ ra của PLC, còn đối với các động cơ công suất lớn thì sử dụng relay hay contactor hoặc bộ khởi động động cơ. Các động cơ sẽ được khảo sát chi tiết hơn ở *chương các cảm biến và cơ cấu chấp hành analog (tập 2)*.

### 3.2.6 Các cơ cấu chấp hành khác

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

Ngoài ra có một số cơ cấu chấp hành khác:

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

## 4 Bộ điều khiển lập trình PLC Simatic S7-200

### 4.1 Cấu hình cứng

#### 4.1.1 Khối xử lý trung tâm

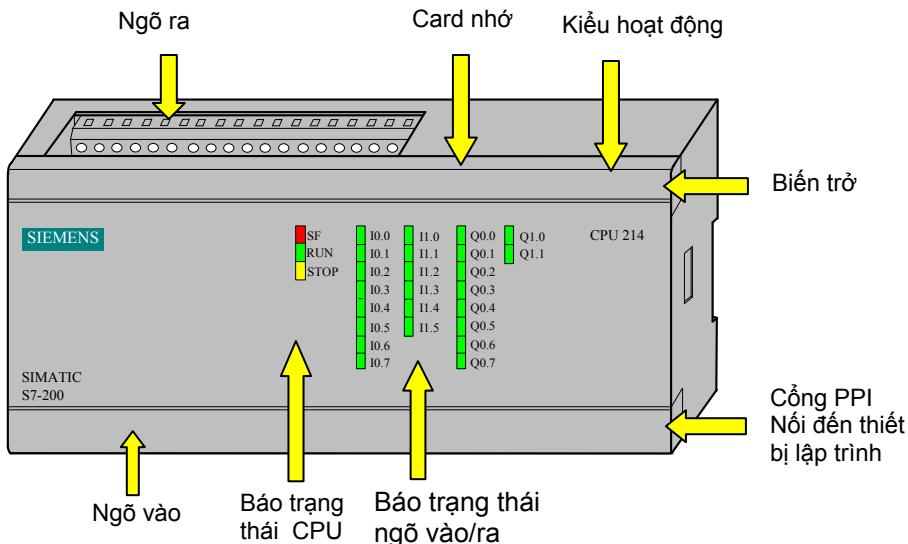
PLC S7-200 là thiết bị điều khiển lập trình loại nhỏ (micro PLC) của hãng Siemens (CHLB Đức) có cấu trúc theo kiểu *modul* và có các modul mở rộng. Thành phần cơ bản của S7 - 200 là khối xử lý trung tâm (CPU: Central Processing Unit) bao gồm hai chủng loại: CPU 21x và CPU 22x. Mỗi chủng loại có nhiều CPU. Loại CPU 21x ngày nay không còn sản xuất nữa, tuy nhiên hiện vẫn còn sử dụng rất nhiều trong các trường học và trong sản xuất. Tiêu biểu cho loại này là CPU 214. CPU 214 có các đặc tính như sau:

- Bộ nhớ chương trình (chứa trong EEPROM): 4096 Byte (4 kByte)
- Bộ nhớ dữ liệu (Vùng nhớ V): 4096 Byte (trong đó 512 Byte chứa trong EEPROM)
- Số lượng ngõ vào: 14 , và
- Số lượng ngõ ra: 10 ngõ ra digital tích hợp trong CPU
- Số module mở rộng: 7 gồm cả module analog
- Số lượng vào/ra số cực đại: 64
- Số lượng Timer : 128 Timer chia làm 3 loại theo độ phân giải khác nhau: 4 Timer 1ms, 16 Timer 10 ms và 108 Timer có độ phân giải 100ms.
- Số lượng Counter: 128 bộ đếm chia làm hai loại: 96 Counter Up và 32 Counter Up/Down.
- Bit memory (Vùng nhớ M): 256 bit
- Special memory (SM) : 688 bit dùng để thông báo trạng thái và đặt chế độ làm việc.
- Có phép tính số học

- Bộ đếm tốc độ cao (High-speed counters): 2 counter 2 KHz và 1 counter 7 KHz
- Ngõ vào analog tích hợp sẵn (biến trờ): 2.
- Các chế độ ngắt và xử lý ngắt gồm: ngắt truyền thông, ngắt theo sườn lên hoặc xuống, ngắt thời gian, ngắt của bộ đếm tốc độ cao và ngắt truyền xung.

Toàn bộ vùng nhớ không bị mất dữ liệu trong khoảng thời gian 190 giờ khi PLC bị mất nguồn nuôi.

Sơ đồ bề mặt của bộ điều khiển logic khả năng S7-200 CPU 214 được cho như hình 4.1.



Hình 4.1: Bộ điều khiển lập trình S7-200 CPU 214

#### \* Mô tả các đèn báo trên CPU 214:

- **SF (Đèn đỏ):** Đèn đỏ SF báo hiệu hệ thống bị lỗi. Đèn SF sáng lên khi PLC có lỗi.
- **RUN (Đèn xanh):** Cho biết PLC đang ở chế độ làm việc và thực hiện chương trình được nạp vào trong bộ nhớ chương trình của PLC.
- **STOP (Đèn vàng):** Đèn vàng STOP chỉ định PLC đang ở chế độ dừng. Dừng chương trình đang thực hiện lại.
- **I x.x (Đèn xanh):** Đèn xanh ở cổng vào chỉ định trạng thái tức thời của cổng ( $x.x = 0.0 - 1.5$ ). Đèn này báo hiệu trạng thái của tín hiệu theo giá trị logic của cổng.

- **Qy.y (Đèn xanh):** Đèn xanh ở cổng ra chỉ định trạng thái tức thời của cổng ( $y.y = 0.0 - 1.1$ ). Đèn này báo hiệu trạng thái của tín hiệu theo giá trị logic của cổng.

Hiện nay, CPU 22x với nhiều tính năng vượt trội đã thay thế loại CPU 21x và hiện đang được sử dụng rất nhiều. Tiêu biểu cho loại này là CPU 224. Thông tin về CPU 22x được cho như bảng 4.1 và hình dáng CPU 224 ở hình 4.2.

<b>Đặc điểm</b>	<b>CPU 221</b>	<b>CPU 222</b>	<b>CPU 224</b>	<b>CPU 224XP</b>	<b>CPU 226</b>
I/O trên CPU Digital Analog	6DI/4DO -	8DI/6DO -	14DI/10DO -	14DI/10DO 2AI/1AO	24DI/16DO -
Số module mở rộng max.	0	2	7	7	7
Bộ nhớ chương trình	4KB	4KB	8KB	12KB	16KB
Bộ nhớ dữ liệu	2KB	2KB	8KB	10KB	10KB
Thời gian xử lý	0,37 µs	0,37 µs	0,37 µs	0,37 µs	0,37 µs
Memory bits/counters/timers	256/256/256	256/256/256	256/256/256	256/256/256	256/256/256
High-speed counters	4 x 30 kHz	4 x 30 kHz	6 x 30 kHz	4 x 30 kHz 2x 200 kHz	6 x 30 kHz
Real-time clock	card	card	Tích hợp	Tích hợp	Tích hợp
Ngõ ra xung	2 x 20 kHz	2 x 20 kHz	2 x 20 kHz	2 x 100 kHz	2 x 20 kHz
Cổng giao tiếp	1x RS-485	1x RS-485	1x RS-485	2x RS-485	2x RS-485
Biến trở analog trên CPU	1	1	2	2	2

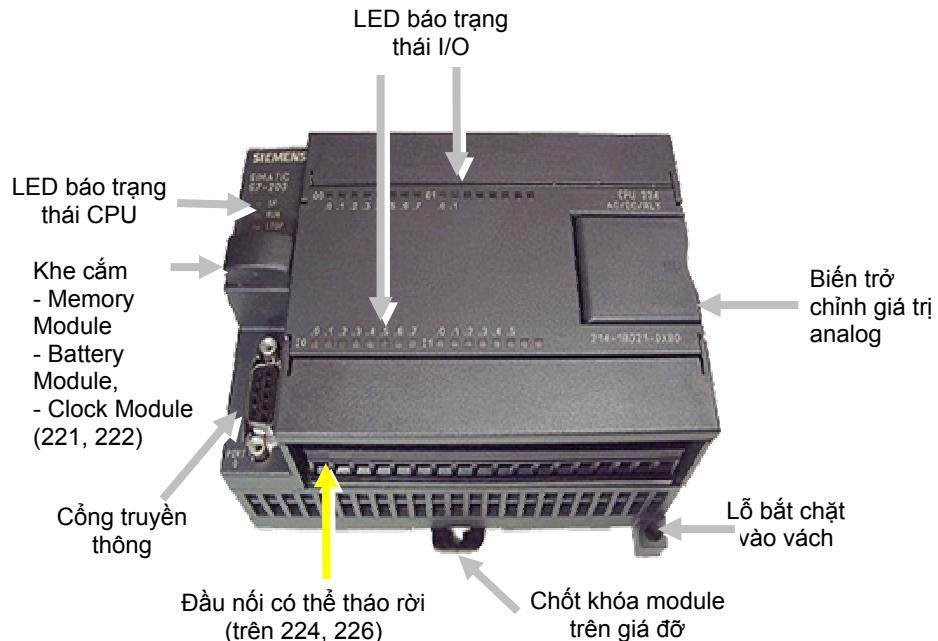
Bảng 4.1: Bảng dữ liệu về CPU họ 22x

#### \* Chọn chế độ làm việc cho PLC

Công tắc chọn chế độ làm việc nằm ở phía trên, có ba vị trí cho phép chọn các chế độ làm việc khác nhau của PLC:

- **RUN:** Cho phép PLC thực hiện chương trình trong bộ nhớ. PLC S7-200 sẽ rời khỏi chế độ RUN và chuyển sang chế độ STOP nếu trong máy có sự cố, hoặc trong chương trình gặp lệnh STOP.

- **STOP:** Cưỡng bức PLC dừng chương trình đang chạy và chuyển sang chế độ STOP. Ở chế độ STOP, PLC cho phép hiệu chỉnh, nạp, xóa một chương trình.
- **TERM:** Cho phép người dùng từ máy tính quyết định chọn một trong hai chế độ làm việc cho PLC hoặc RUN hoặc STOP.



Hình 4.2: Bộ điều khiển lập trình CPU 224

#### \* Cổng truyền thông

S7-200 sử dụng cổng truyền thông nối tiếp RS485 với phích nối 9 chân để phục vụ cho việc ghép nối với thiết bị lập trình hoặc với các trạm PLC khác. Tốc độ truyền cho máy lập trình kiểu PPI là 9600 baud. Tốc độ truyền cung cấp của PLC theo kiểu tự do là từ 300 baud đến 38400 baud.

Để ghép nối S7-200 với máy lập trình PG720 (hãng Siemens) hoặc với các loại máy lập trình thuộc họ PG7xx có thể sử dụng một cáp nối thẳng qua MPI. Cáp đó đi kèm theo máy lập trình.

Ghép nối S7-200 với máy tính PC qua cổng RS-232 cần có cáp nối PC/PPI với bộ chuyển đổi RS232/RS485, và qua cổng USB ta có cáp USB/PPI.

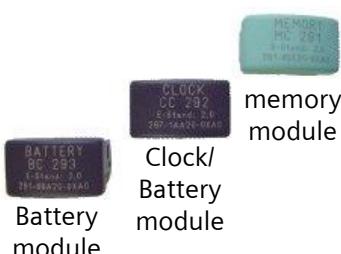
#### \* Card nhớ, pin, clock (CPU 221, CPU222)

S7-200 cung cấp nhiều biện pháp đảm bảo cho chương trình người dùng, dữ liệu chương trình và cấu hình dữ liệu được duy trì sau:

Một tụ điện với điện dung lớn cho phép nuôi bộ nhớ RAM sau khi bị mất nguồn điện cung cấp. Tùy theo loại CPU mà thời gian lưu trữ có thể kéo dài nhiều ngày. Chẳng hạn ở CPU 224 là khoảng 100 giờ.

Vùng nhớ EEPROM cho phép lưu chương trình, các vùng nhớ được người dùng chọn chứa vào EEPROM và cấu hình dữ liệu.

Cho phép gắn thêm Pin để nuôi RAM và cho phép kéo dài thời gian lưu trữ dữ liệu, có thể lên đến 200 ngày kể từ khi mất nguồn điện. Nguồn của Pin sẽ được lấy sau khi tụ điện đã xả hết.



Hình 4.3: Hình dáng các module

- **Card nhớ:** Được sử dụng để lưu trữ chương trình. Chương trình chứa trong card nhớ bao gồm: program block, data block, system block, công thức (recipes), dữ liệu đo (data logs), và các giá trị cưỡng bức (force values).
- **Card pin:** Dùng để mở rộng thời gian lưu trữ các dữ liệu có trong bộ nhớ. Nguồn pin được tự động chuyển sang khi tụ trong PLC cạn. Pin có thể sử dụng đến 200 ngày.

- **Card Clock / Battery module:** đồng hồ thời gian thực (Real-time clock) cho CPU 221, 222 và nguồn pin để nuôi đồng hồ và lưu dữ liệu. Thời gian sử dụng đến 200 ngày.

#### \* Biến trở chỉnh giá trị analog:

Hai biến trở này được sử dụng như hai ngõ vào analog cho phép điều chỉnh các biến cần phải thay đổi và sử dụng trong chương trình.

### 4.1.2 Khối mở rộng

Trên các CPU đã tích hợp sẵn một số các ngõ vào và ngõ ra số, chẳng hạn như CPU 224 DC/DC/DC có sẵn 16 ngõ vào và 14 ngõ ra. Tuy nhiên trong thực tế, xuất phát từ yêu cầu điều khiển như: cần nhiều hơn số ngõ vào/ra có sẵn, có sử dụng tín hiệu analog hay có các yêu cầu về truyền thông, nối mạng các PLC... mà ta phải gắn thêm vào CPU các khối mở rộng (Expansion module) có các chức năng khác nhau (bảng 4.2).

#### 4.1.2.1 Digital module

Các module số gắn thêm vào khối CPU để mở rộng số lượng các ngõ vào/ra số.

- **Khối ngõ vào số DI (Digital Input):** Siemens sản xuất các khối ngõ vào số như: DI8 x 24VDC, DI8 x AC120/230V, DI16 x 24VDC.
- **Khối ngõ ra số (Digital Output):** Các ngõ ra này được chia ra làm 3 loại là ngõ ra DC, ngõ ra AC và ngõ ra relay. Điện áp ngõ ra có thể là 24Vdc hoặc 230Vac tùy loại, với số lượng ngõ ra có thể là 4 hoặc 8.

Ngoài ra còn có sự kết hợp các ngõ vào và ra số trên cùng một module.

#### 4.1.2.2 Analog module

Ngoại trừ CPU 224XP có tích hợp sẵn 2 ngõ vào và 1 ngõ ra analog (2AI/1AO) để kết nối với ngoại vi nhận và phát tín hiệu analog, thì hầu hết các CPU khác của họ S7-200 đều không có tích hợp sẵn. Vì vậy khi điều khiển với tín hiệu analog thì yêu cầu người sử dụng phải gắn thêm các khối analog.

- Khối ngõ vào tương tự AI (Analog Input):** Tín hiệu analog ngõ vào có thể là tín hiệu điện áp hoặc dòng điện. Tùy thuộc vào tín hiệu analog cần đọc là loại nào mà người sử dụng có thể cài đặt cho phù hợp bằng các công tắc được gắn trên module (*Chi tiết xem chương xử lý tín hiệu analog*).

Hiện có các khối ngõ vào: 4AI, 8AI. Đối với tín hiệu analog được tạo ra bởi thermocoupe (cặp nhiệt) và RTD thì sử dụng các module đo nhiệt tương ứng (bảng 4.2).

- Khối ngõ ra tương tự AO (Analog Output):** Tín hiệu tương tự này có thể là điện áp hoặc dòng điện tùy theo người dùng cài đặt. Tín hiệu ra là điện áp nằm trong khoảng  $\pm 10\text{Vdc}$  tương ứng với giá trị số từ -32000 tới +32000 và tín hiệu dòng điện nằm trong khoảng từ 0 - 20mA tương ứng với giá trị số từ 0 tới +32000.

Ngoài các khối trên còn có các khối có sự kết hợp cả 2 loại tín hiệu vào và ra analog trên cùng một khối.

Các khối mở rộng	Loại			
<b>Digital module</b>				
Input	8 x DC In	8 x AC In	16 x DC In	
Output	4 x DC Out	4 x Relay	8 x Relay	
	8 x DC Out	8 x AC Out		
Tổ hợp	4 x DC In/ 4 x DC Out	8 x DC In/ 8 x DC Out	16 x DC In/ 16x DC Out	32 x DC In/ 32x DC Out
	4 x DC In/ 4 x Relay	8 x DC In/ 8 x Relay	16 x DC In/ 16x DC Out	32 x DC In/ 32x Relay
<b>Analog module</b>				
Input	4 x Analog In	8 x Analog In	4xThermocouple In	
	2 x RTD In	2 x RTD In		
Output	2 x Analog Out	4 x Analog Out		
Tổ hợp	4 x Analog In 4 x Analog Out			
<b>Intelligent module</b>				
	Position	Modem	PROFIBUS-DP	
	Ethernet	Ethenet IT		
<b>Các module khác</b>				
	AS-Interface	SIWAREX MS		

Bảng 4.2: Các loại khối mở rộng

#### 4.1.2.3 Intelligent module

Các PLC S7-200 có thể nối vào các loại mạng khác nhau để tăng cường khả năng mở rộng, truyền thông với các thiết bị khác trong hệ thống tự động hóa.

- *Master trong mạng AS-Interface:* Giao tiếp AS-i (Actuator Sensor Interface) hay giao tiếp actuator/sensor là hệ thống kết nối cho cấp quá trình thấp nhất trong hệ thống tự động hóa nhằm tối ưu hóa việc kết nối cảm biến và cơ cấu chấp hành với thiết bị tự động hóa. Với module CP243-2 cho phép kết nối mạng AS-Interface vào PLC S7-200 và đóng vai trò là master.
- *Kết nối vào mạng PROFIBUS-DP:* Các PLC S7-200 có thể kết nối vào mạng Profibus hoạt động như một DP Slave nhờ vào khối mở rộng EM277. Việc sử dụng EM277 cho phép PLC S7-200 có thể kết nối truyền thông với các thiết bị trong mạng Profibus như: PLC S7-300, S7-400, màn hình điều khiển...
- *Kết nối vào mạng Ethernet:* Để có thể kết nối S7-200 vào mạng Industrial Ethernet thì cần có khối CP 243-1. Đây là khối truyền thông cho phép các PLC S7-200 có thể được cấu hình, lập trình, chẩn đoán từ xa qua Ethernet nhờ phần mềm STEP 7 Micro/win. Giúp cho các CPU S7-200 có thể giao tiếp với các S7-200 khác, S7-300 hay S7-400 qua Ethernet. Các CPU có thể sử dụng là họ CPU 22X. Có thể thực hiện cấu hình cho các CPU vào mạng Ethernet nhờ vào *Wizard (Menu Tools → Ethernet wizard)*.
- *Internet Technology:* Khối mở rộng CP 243-1 IT cho phép các CPU S7-200 có thể thực hiện các giám sát hay thay đổi qua trình duyệt Web từ một PC có nối mạng. Các thông báo chẩn đoán có thể gửi qua email từ một hệ thống. Sử dụng các chức năng IT cho phép trao đổi các tập tin dữ liệu với các máy tính hay các hệ thống điều khiển khác. Mỗi một khối CP 243-1IT chỉ nên kết nối cho 2 CPU S7-200.
- *Modem module:* Cho phép kết nối trực tiếp S7-200 vào đường dây điện thoại, và cung cấp truyền thông giữa S7-200 và Step 7- micro/Win.

Với công cụ *Modem Expansion wizard* cho phép thiết lập một modem ở xa hoặc kết nối S7-200 với một thiết bị ở xa qua modem.

Khả năng truyền thông của S7-200 được cho như hình 4.4.

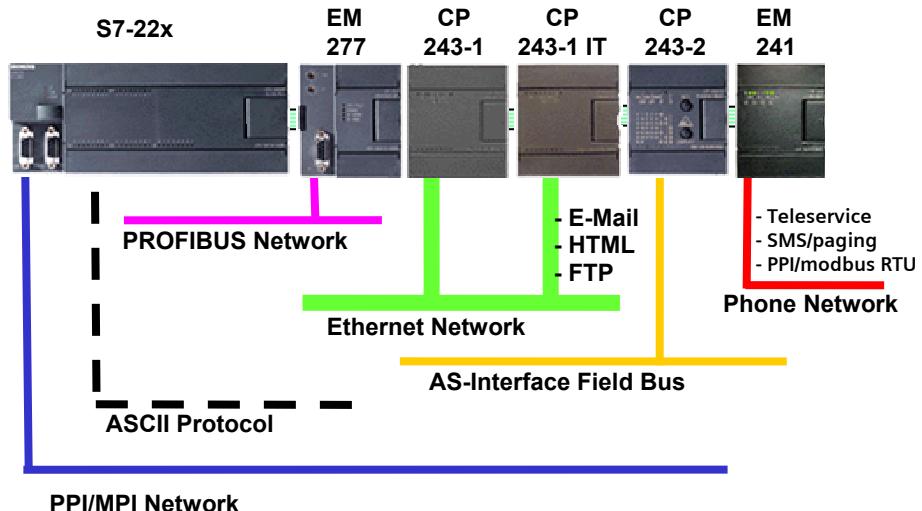
#### 4.1.2.4 Function module

Là các khối chức năng thực hiện các chức năng đặc biệt như điều khiển vị trí (position module), cân (SIWREX MS).

- *Position module:* Module vị trí được sử dụng để điều khiển tốc độ và vị trí của động cơ bước (stepper motor) hoặc động cơ servo (servo motor). Với công cụ Position Control wizard trong phần mềm STEP 7--Micro/WIN

để thiết lập cấu hình cho module điều khiển vị trí. Module điều khiển vị trí thường được sử dụng là EM253.

SIWAREX MS: Là module cân đa năng và linh hoạt, nó được sử dụng với các hệ thống cân hoặc đo lực sử dụng PLC S7-200.

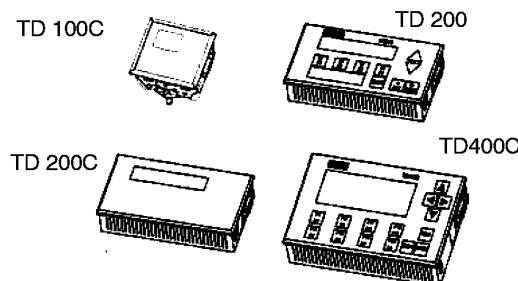


Hình 4.4: Khả năng truyền thông của PLC S7-200

## 4.2 Màn hình điều khiển

Trong các yêu cầu điều khiển có giám sát thì đối với các PLC S7-200 chúng ta có thể gắn thêm các màn hình để điều khiển và giám sát. Hiện có các loại là: màn hình hiển thị dòng văn bản (Text Display), màn hình điều khiển bằng bàn phím (Operator panel) và màn hình cảm ứng (Touch Panel).

\* **Bảng điều khiển hiển thị dòng văn bản (Text Display):** Các màn hình này có giá thành thấp, cho phép người vận hành máy có thể xem, giám sát bằng các dòng văn bản và thay đổi các thông số hay chế độ hoạt động của hệ thống điều khiển bằng các phím trên bảng điều khiển. Gồm có các loại là TD100C, TD200C, TD 200, TD400C (hình 4.5).



Hình 4.5: Bảng điều khiển hiển thị dòng văn bản

Các bảng điều khiển này có thể được thiết lập các thông báo và nút nhấn điều khiển dễ dàng bằng công cụ **Text Display wizard** (menu lệnh Tools > Text Display Wizard) trong STEP 7-Micro/WIN.

\* **Operator Panel và Touch Panel:** Các màn hình được ứng dụng điều khiển và giám sát các máy móc, thiết bị nhỏ. Thời gian thiết lập cấu hình và vận hành nhanh với phần mềm WinCC flexible. Gồm có các loại: OP 73micro, TP 177micro (màn hình này thay thế các màn hình trước TP 070/TP 170micro) (hình 4.6).



Hình 4.6: Màn hình OP 73micro và TP 177micro.

### 4.3 Các vùng nhớ

Bộ nhớ của các PLC S7-200 được chia ra làm các vùng nhớ như bảng 4.3.

\* **Vùng nhớ đệm ngõ vào số I:**

CPU sẽ đọc trạng thái tín hiệu của tất cả các ngõ vào số ở đầu mỗi chu kỳ quét, sau đó sẽ chứa các giá trị này vào vùng nhớ đệm ngõ vào. Có thể truy cập vùng nhớ này theo bit, Byte, Word hay Doubleword.

\* **Vùng nhớ đệm ngõ ra số Q:**

Trong quá trình xử lý chương trình CPU sẽ lưu các giá trị xử lý thuộc vùng nhớ ngõ ra vào đây. Tại cuối mỗi vòng quét CPU sẽ sao chép nội dung vùng nhớ đệm này và chuyển ra các ngõ ra vật lý. Có thể truy cập vùng nhớ này theo bit, Byte, Word hay Doubleword.

\* **Vùng nhớ biến V:**

Sử dụng vùng nhớ V để lưu trữ các kết quả phép toán trung gian có được do các xử lý logic của chương trình. Cũng có thể sử dụng vùng nhớ để lưu trữ các dữ liệu khác liên quan đến chương trình hay nhiệm vụ điều khiển. Có thể truy cập vùng nhớ này theo bit, Byte, Word hay Doubleword.

\* **Vùng nhớ M:**

Có thể coi vùng nhớ M như là các relay điều khiển trong chương trình để lưu trữ trạng thái trung gian của một phép toán hay các thông tin điều khiển khác. Có thể truy cập vùng nhớ này theo bit, Byte, Word hay Doubleword.

\* **Vùng nhớ bộ định thời T:**

S7-200 cung cấp vùng nhớ riêng cho các bộ định thời, các bộ định thời được sử dụng cho các yêu cầu điều khiển cần trì hoãn thời gian. Giá trị thời gian sẽ được đếm tăng dần theo 3 độ phân giải là 1ms, 10ms và 100ms.

Mô tả	CPU 221	CPU 222	CPU 224	CPU 224XP	CPU226
Kích thước chương trình người dùng	4 KB	4 KB	8 KB	12 KB	16 KB
Kích thước dữ liệu	2 KB	2 KB	8 KB	10 KB	10 KB
Vùng đếm vào số	I0.0 ... I15.7	I0.0 ... I15.7	I0.0 ... I15.7	I0.0 ... I15.7	I0.0 ... I15.7
Vùng đếm ra số	Q0.0 ... Q15.7	Q0.0 ... Q15.7	Q0.0 ... Q15.7	Q0.0 ... Q15.7	Q0.0 ... Q15.7
Ngõ vào analog	AIW0 .. AIW30	AIW0 .. AIW30	AIW0 .. AIW62	AIW0 .. AIW62	AIW0 .. AIW62
Ngõ ra analog	AQW0...AQW30	AQW0...AQW30	AQW0...AQW62	AQW0...AQW62	AQW0...AQW62
Vùng nhớ biến (V)	VB0...VB2047	VB0...VB2047	VB0...VB8191	VB0...VB10239	VB0...VB10239
Vùng nhớ cục bộ (L)	LB0...LB63	LB0...LB63	LB0...LB63	LB0...LB63	LB0...LB63
Vùng nhớ bit (M)	M0.0...M31.7	M0.0...M31.7	M0.0...M31.7	M0.0...M31.7	M0.0...M31.7
Vùng nhớ đặc biệt Chỉ đọc (SM)	SM0.0...SM179.7 SM0.0...SM29.7	SM0.0...SM299.7 SM0.0...SM29.7	SM0.0...SM549.7 SM0.0...SM29.7	SM0.0...SM549.7 SM0.0...SM29.7	SM0.0...SM549.7 SM0.0...SM29.7
Timer Retentive on-delay	256 (T0...T255) 1ms T0, T64 10ms T1...T4, và T65...T68 100ms T5...T31, và T69...T95	256 (T0...T255) T0, T64 T1...T4, và T65...T68 T5...T31, và T69...T95			
On/Off delay	1ms T32, T96 10ms T33 ... T36, và T97 ... T100 100ms T37 ... T63, và T101 ... T255	T32, T96 T33 ... T36, và T97 ... T100 T37 ... T63, và T101 ... T255	T32, T96 T33 ... T36, và T97 ... T100 T37 ... T63, và T101 ... T255	T32, T96 T33 ... T36, và T97 ... T100 T37 ... T63, và T101 ... T255	T32, T96 T33 ... T36, và T97 ... T100 T37 ... T63, và T101 ... T255
Counter	C0 ... C255	C0 ... C255	C0 ... C255	C0 ... C255	C0 ... C255
Bộ đếm tốc độ cao	HC0 ... HC5	HC0 ... HC5	HC0 ... HC5	HC0 ... HC5	HC0 ... HC5
Bit điều khiển trình tự (S)	S0.0 ... S31.7	S0.0 ... S31.7	S0.0 ... S31.7	S0.0 ... S31.7	S0.0 ... S31.7
Thanh ghi Accu	AC0 ... AC3	AC0 ... AC3	AC0 ... AC3	AC0 ... AC3	AC0 ... AC3
Jumps/Labels	0 ... 255	0 ... 255	0 ... 255	0 ... 255	0 ... 255
Call/Subroutine	0 ... 63	0 ... 63	0 ... 63	0 ... 63	0 ... 127
Interrupt routines	0 ... 127	0 ... 127	0 ... 127	0 ... 127	0 ... 127
Ô nhớ sườn xung (positive/negative)	256	256	256	256	256
PID loops	0 ... 7	0 ... 7	0 ... 7	0 ... 7	0 ... 7
Port	Port 0	Port 0	Port 0	Port 0, Port 1	Port 0, Port 1

Bảng 4.3: Các vùng nhớ và đặc điểm của CPU S7-200.

\* Vùng nhớ bộ đếm C:

Có 3 loại bộ đếm là bộ đếm lên, bộ đếm xuống và bộ đếm lên-xuống. Các bộ đếm sẽ tăng hoặc giảm giá trị hiện hành khi tín hiệu tại ngõ vào thay đổi trạng thái từ mức thấp lên mức cao.

\* *Vùng nhớ bộ đếm tốc độ cao HC (High speed Counter):*

Các bộ đếm tốc độ cao được sử dụng để đếm các sự kiện tốc độ cao độc lập với vòng quét của CPU. Giá trị đếm là số nguyên 32 bit có dấu. Để truy xuất giá trị đếm của các bộ đếm tốc độ cao cần xác định địa chỉ của bộ đếm tốc độ cao, sử dụng vùng nhớ HC và số của bộ đếm, ví dụ HC0. Giá trị đếm hiện hành của các bộ đếm tốc độ cao là các giá trị chỉ đọc và truy xuất theo double word.

\* *Các thanh ghi AC (Accumulators):*

Các thanh ghi AC là các phần tử đọc/ghi mà có thể được dùng để truy xuất giống như bộ nhớ. Chẳng hạn, có thể sử dụng các thanh ghi để truy xuất các thông số từ các chương trình con (Subroutine) và lưu trữ các giá trị trung gian để sử dụng cho tính toán. Các CPU S7-200 có 4 thanh ghi là AC0, AC1, AC2 và AC3. Chúng ta có thể truy xuất dữ liệu trong các thanh ghi này theo Byte, Word, và Doubleword.

\* *Vùng nhớ đặc biệt SM (Special Memory):*

Các bit SM là các phần tử cho phép truyền thông tin giữa CPU và chương trình người dùng. Có thể sử dụng các bit này để chọn lựa và điều khiển một số chức năng đặc biệt của CPU, chẳng hạn như bit lên mức 1 trong vòng quét đầu tiên, các bit phát ra các xung có tần số 1Hz...Chúng ta truy xuất vùng nhớ SM theo bit, byte, word, doubleword.

\* *Vùng nhớ cục bộ L (Local Memory Area):*

Vùng nhớ này có độ lớn 64 Byte, trong đó 60 byte có thể được dùng như vùng nhớ cục bộ hay chuyển các thông số tới các chương trình con, 4 byte cuối dùng cho hệ thống. Vùng nhớ này tương tự như vùng nhớ biến V chỉ khác ở chỗ các biến vùng nhớ V cho phép sử dụng ở tất cả các khối chương trình còn vùng nhớ L chỉ có tác dụng trong phạm vi soạn thảo của một khối chương trình mà thôi. Vị trí biến thuộc vùng nhớ L trong chương trình chính thì không thể sử dụng ở chương trình con và ngược lại.

\* *Vùng nhớ ngõ vào tương tự AI (Analog Inputs):*

Các PLC S7-200 chuyển giá trị một tương tự (chẳng hạn điện áp hay nhiệt độ) thành giá trị số và chứa vào một vùng nhớ 16 bit. Bởi vì các giá trị tương tự chiếm một vùng nhớ word nên chúng luôn luôn có các giá trị word chẵn, chẳng hạn như AIW0, AIW2, AIW4..và là các giá trị chỉ đọc.

\* *Vùng nhớ ngõ ra tương tự AQ (Analog Outputs):*

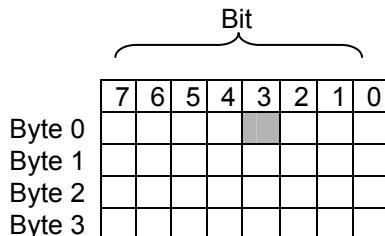
Các PLC S7-200 chuyển một giá trị số 16 bit sang giá trị điện áp hoặc dòng điện, tương ứng với giá trị số (digital). Giống như các ngõ vào tương tự chúng ta chỉ có thể truy xuất các ngõ ra tương tự theo word. Và là các giá trị word chẵn, chẳng hạn như AQW0, AQW2, AQW4.

## 4.4 Qui ước địa chỉ trong PLC S7-200

### 4.4.1 Truy xuất theo bit

Để truy xuất địa chỉ theo dạng Bit chúng ta xác định vùng nhớ, địa chỉ của Byte và địa chỉ của Bit.

Ví dụ:



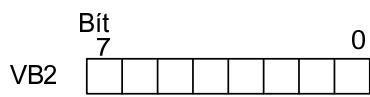
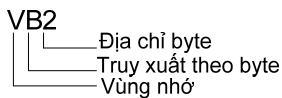
Hình 4.7: Vùng nhớ ngõ vào /

Trong hình 4.7 là bản đồ vùng nhớ của bộ đệm dữ liệu ngõ vào I (Process Image Input). Bản đồ của các vùng nhớ khác cũng có cấu trúc tương tự như vậy. Bit thấp nhất là bit 0 nằm bên phải và bit cao nhất là bit 7 nằm bên trái. Do đó chúng ta hoàn toàn có thể khai báo tương tự như ví dụ trên, chẳng hạn như: Q1.0, V5.2, M0.1... Dung lượng của các vùng nhớ phụ thuộc vào loại CPU mà chúng ta sử dụng.

### 4.4.2 Truy xuất theo byte (8 bit)

Khi truy xuất dữ liệu theo byte, chúng ta xác định vùng nhớ, và thứ tự của byte cần truy xuất.

Ví dụ:

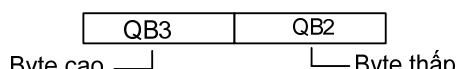
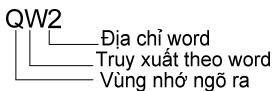


Tương tự như ví dụ ta khai báo cho các vùng nhớ khác, chẳng hạn như IB3, MB2, QB5..

### 4.4.3 Truy xuất theo word (16 bit)

Đối với truy xuất vùng nhớ theo dạng word chúng ta cũng cần xác định vùng nhớ cần truy xuất, khai báo dạng word và địa chỉ của word trong vùng nhớ. Mỗi một vùng nhớ dạng word sẽ gồm 2 byte và được gọi là byte thấp và byte cao.

Ví dụ:



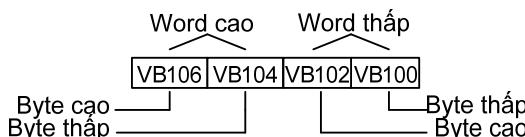
Chú ý:

- Đối với tín hiệu tương tự (Analog) thì chúng ta chỉ có một dạng truy xuất duy nhất là truy xuất theo word. Điều này là do mỗi tín hiệu tương tự sẽ ứng với một giá trị số nguyên 16 bit. Ví dụ: AIW0, AIW2, AQW0...
- Khi truy xuất địa chỉ theo word thì hai word liền kề nhau bắt buộc cách nhau 2 byte. Ví dụ ta cần chứa 2 dữ liệu dạng số integer vào vùng biến V, thì dữ liệu thứ nhất giả sử chứa vào VW20 thì word kế tiếp lưu dữ liệu thứ hai là VW22.

#### 4.4.4 Truy xuất theo 2 word (Double word = 32 bit)

Khi truy xuất vùng nhớ 32 bit, tương ứng với 4 byte. Trong đó gồm có word thấp, word cao và byte thấp, byte cao.

Ví dụ: VD100



Bảng tóm tắt việc truy xuất các vùng nhớ theo bit, byte, word và double word được cho ở bảng 4.4.

Cách truy xuất	CPU 221	CPU 222	CPU 224	CPU 224XP	CPU 226
Truy xuất Bit (byte.bit)	I 0 ... 15.7 Q 0 ... 15.7 V 0 ... 2047.7 M 0 ... 31.7 SM 0 ... 165.7 S 0 ... 31.7 T 0 ... 255 C 0 ... 255 L 0 ... 63.7	0.0 ... 15.7 0.0 ... 15.7 0.0 ... 2047.7 0.0 ... 31.7 0.0 ... 299.7 0.0 ... 31.7 0 ... 255 0 ... 255 0.0 ... 63.7	0.0 ... 15.7 0.0 ... 15.7 0.0 ... 8191.7 0.0 ... 31.7 0.0 ... 549.7 0.0 ... 31.7 0 ... 255 0 ... 255 0.0 ... 63.7	0.0 ... 15.7 0.0 ... 15.7 0.0 ... 10239.7 0.0 ... 31.7 0.0 ... 549.7 0.0 ... 31.7 0 ... 255 0 ... 255 0.0 ... 63.7	0.0 ... 15.7 0.0 ... 15.7 0.0 ... 10239.7 0.0 ... 31.7 0.0 ... 549.7 0.0 ... 31.7 0 ... 255 0 ... 255 0.0 ... 63.7
Truy xuất Byte	IB 0 ... 15 QB 0 ... 15 VB 0 ... 2047 MB 0 ... 31 SMB 0 ... 165 SB 0 ... 31 LB 0 ... 63 AC 0 ... 3 KB (Constant)	0 ... 15 0 ... 15 0 ... 2047 0 ... 31 0 ... 299 0 ... 31 0 ... 63 0 ... 3 KB (Constant)	0 ... 15 0 ... 15 0 ... 8191 0 ... 31 0 ... 549 0 ... 31 0 ... 63 0 ... 3 KB (Constant)	0 ... 15 0 ... 15 0 ... 10239 0 ... 31 0 ... 549 0 ... 31 0 ... 63 0 ... 255 KB (Constant)	0 ... 15 0 ... 15 0 ... 10239 0 ... 31 0 ... 549 0 ... 31 0 ... 63 0 ... 255 KB (Constant)
Truy xuất Word	IW 0 ... 14 QW 0 ... 14 VW 0 ... 2046 MW 0 ... 30 SMW 0 ... 164 SW 0 ... 30 T 0 ... 255 C 0 ... 255 LW 0 ... 62 AC 0 ... 3 AIW 0 ... 30 AQW 0 ... 30 KW (Constant)	0 ... 14 0 ... 14 0 ... 2046 0 ... 30 0 ... 298 0 ... 30 0 ... 255 0 ... 255 0 ... 62 0 ... 3 0 ... 30 0 ... 30 0 ... 12 0 ... 12 0 ... 2044	0 ... 14 0 ... 14 0 ... 8190 0 ... 30 0 ... 548 0 ... 30 0 ... 255 0 ... 255 0 ... 62 0 ... 3 0 ... 30 0 ... 30 0 ... 12 0 ... 12 0 ... 2044	0 ... 14 0 ... 14 0 ... 10238 0 ... 30 0 ... 548 0 ... 30 0 ... 255 0 ... 255 0 ... 62 0 ... 3 0 ... 30 0 ... 30 0 ... 12 0 ... 12 0 ... 8188	0 ... 14 0 ... 14 0 ... 10238 0 ... 30 0 ... 548 0 ... 30 0 ... 255 0 ... 255 0 ... 62 0 ... 3 0 ... 30 0 ... 30 0 ... 12 0 ... 12 0 ... 10236
T. xuất Double word	ID 0 ... 12 QD 0 ... 12 VD 0 ... 2044	0 ... 12 0 ... 12 0 ... 2044	0 ... 12 0 ... 12 0 ... 8188	0 ... 12 0 ... 12 0 ... 10236	0 ... 12 0 ... 12 0 ... 10236

MD	0 ... 28	0 ... 28	0 ... 28	0 ... 28	0 ... 28
SMD	0 ... 162	0 ... 296	0 ... 546	0 ... 546	0 ... 546
SD	0 ... 28	0 ... 28	0 ... 28	0 ... 28	0 ... 28
LD	0 ... 60	0 ... 60	0 ... 60	0 ... 60	0 ... 60
AC	0 ... 3	0 ... 3	0 ... 3	0 ... 3	0 ... 3
HC	0 ... 5	0 ... 5	0 ... 5	0 ... 5	0 ... 5
KD (Constant)					

**Bảng 4.4: Truy xuất các vùng nhớ theo địa chỉ bit, byte, word, double word.**

Tóm lại, về cơ bản chúng ta có bốn dạng truy xuất dữ liệu như trên. Trong mỗi yêu cầu điều khiển cụ thể chúng ta sẽ chọn truy xuất theo dạng nào.

- Kiểm tra trạng thái của các tín hiệu được tạo ra từ các ngoại vi nối với ngõ vào số như nút nhấn, cảm biến, công tắc hành trình... thì sẽ chọn truy xuất là bit, trong trường hợp này thì chọn địa chỉ ngõ vào tương ứng được kết nối ví dụ như I0.0, I0.5, I1.1...
- Xuất tín hiệu ra các cơ cấu chấp hành nhận tín hiệu nhị phân như relay, đèn báo, van từ ... thì sẽ chọn truy xuất là bit, trong trường hợp này thì chọn địa chỉ ngõ ra tương ứng được kết nối ví dụ như Q0.0, Q0.2, Q1.0...
- Nhận tín hiệu từ các cảm biến tạo ra tín hiệu analog như cảm biến nhiệt độ, áp suất, độ ẩm ... thì sử dụng địa chỉ word, ví dụ: AIW0, AIW2, AIW4...
- Xuất tín hiệu analog ra các cơ cấu chấp hành nhận tín hiệu analog như ngõ vào analog biến tần, van tỉ lệ ... thì sử dụng địa chỉ word, ví dụ: AQW0, AQW2, AQW4...
- Trong quá trình thực hiện chương trình cần lưu trữ thông tin ở dạng số 16 bit như đếm số sản phẩm (số nguyên 16 bit) thì truy cập địa chỉ word, còn ở dạng 32 bit như nhiệt độ, áp suất (số thực) thì truy cập địa chỉ double word...

## 4.5 Xử lý chương trình

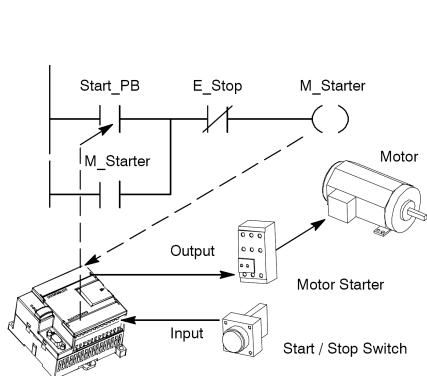
S7-200 thực hiện đọc và ghi dữ liệu theo logic điều khiển trong chương trình liên tục theo chu kỳ.

Hoạt động của S7-200 rất đơn giản:

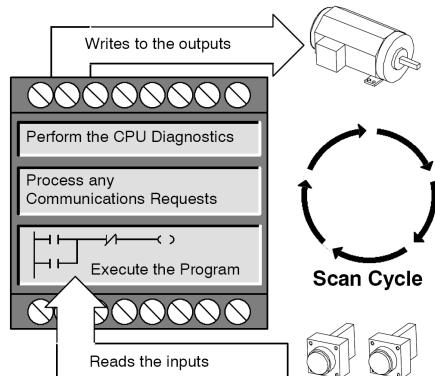
- Đọc trạng thái các ngõ vào
- S7-200 sử dụng các ngõ vào này để thực hiện logic điều khiển theo chương trình được lưu trữ trong nó. Dữ liệu luôn được cập nhật khi chương trình được thực hiện.
- Xuất dữ liệu ra ngõ ra.

Hình 4.8 là một sơ đồ đơn giản chỉ mối quan hệ giữa sơ đồ điện và PLC S7-200. Các nút nhấn khởi động/dừng động cơ được kết nối với ngõ vào. Trạng thái của các ngõ vào tùy thuộc vào nút nhấn. Các trạng thái của ngõ vào sẽ quyết định trạng thái của ngõ ra. Ngõ ra được kết nối với contactor.

Tùy theo trạng thái của ngõ ra mà contactor có điện hay mất điện và tương ứng động cơ sẽ hoạt động hay dừng.



Hình 4.8: Điều khiển ngõ vào và ra



Hình 4.9: Chu kỳ quét S7-200

#### \* Chu kỳ quét trong S7-200

S7-200 thực hiện một loạt các nhiệm vụ theo chu kỳ. Việc thực hiện các nhiệm vụ theo chu kỳ được gọi là chu kỳ quét (scan cycle). Hình 4.9 là ví dụ một chu kỳ quét. S7-200 thực hiện các nhiệm vụ sau trong một chu kỳ quét:

- **Đọc ngõ vào:** S7-200 sao chép trạng thái của các ngõ vào vật lý vào bộ đệm ngõ vào.

*Digital inputs: Mỗi chu kỳ quét bắt đầu bằng cách đọc giá trị hiện hành các ngõ vào số và sau đó ghi các giá trị này vào vùng đệm ngõ vào.*

*Analog inputs: S7-200 không cập nhật các ngõ vào analog từ các module mở rộng nếu là chu kỳ quét bình thường trừ khi có kích hoạt khâu lọc các ngõ vào analog (xem chương xử lý tín hiệu analog). Bộ lọc analog được cung cấp cho phép ta có một tín hiệu ổn định hơn. Có thể cho phép bộ analog ở mỗi điểm ngõ vào analog. Khi một ngõ vào analog được kích hoạt ở bộ lọc, S7-200 cập nhật ngõ vào analog mỗi lần trong chu kỳ quét và lưu trữ giá trị lọc. Giá trị lọc được cung cấp mỗi khi truy cập ngõ vào analog. Khi bộ lọc analog không được kích hoạt, S7-200 đọc giá trị ngõ vào analog từ module mở rộng mỗi lần chương trình truy xuất ngõ vào analog.*

- **Thực hiện theo logic điều khiển trong chương trình:** S7-200 thực hiện các lệnh trong chương trình và lưu giá trị vào vùng nhớ.

*Khi thực hiện chu kỳ quét, S7-200 thi hành từ lệnh đầu tiên cho đến lệnh cuối cùng. Các lệnh truy cập I/O tức thì cho phép ta truy xuất ngay lập tức các ngõ vào và ngõ ra khi thực hiện chương trình cũng như chương trình ngắt (interrupt routine).*

*Nếu có sử dụng các ngắt trong chương trình (chương trình ngắt được gọi bởi các yêu cầu ngắt) thì nó không được thực hiện ở chu kỳ quét*

bình thường. Nó được thực hiện khi có sự kiện ngắn (có thể xảy ra tại bất kỳ thời điểm nào trong chu kỳ quét).

- **Xử lý bất kỳ yêu cầu truyền thông nào:** S7-200 thi hành bất kỳ nhiệm vụ được yêu cầu cho truyền thông.

*Trong giai đoạn xử lý thông tin của chu kỳ quét, S7-200 xử lý bất kỳ thông tin nào nhận được từ cổng truyền thông hoặc từ các module truyền thông (intelligent I/O module).*

- **Thực hiện tự chẩn đoán CPU:** S7-200 tự kiểm tra để đảm bảo phần firmware, bộ nhớ chương trình, và bất kỳ các module mở rộng nào cũng đang làm việc đúng.

*Trong giai đoạn này, S7-200 kiểm tra cho hoạt động thích hợp của CPU và trạng thái của bất kỳ module mở rộng nào.*

- **Xuất ra ngõ ra:** Các giá trị được lưu trong vùng đệm ngõ ra sẽ được xuất ra các ngõ ra vật lý.

*Tại cuối mỗi chu kỳ, S7-200 xuất các giá trị được lưu trong bộ đệm ngõ ra đến các ngõ ra số. (Các ngõ ra analog thì được cập nhật ngay lập tức, không phụ thuộc vào chu kỳ quét).*

Việc thực hiện chương trình còn tùy thuộc vào S7-200 đang ở chế độ STOP hay chế độ RUN. Ở chế độ RUN thì chương trình được thực hiện; còn ở chế độ STOP thì chương trình không được thực hiện.

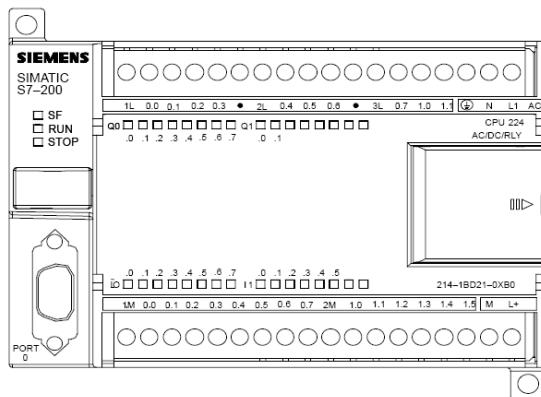
# 5 KẾT NỐI DÂY GIỮA PLC VÀ THIẾT BỊ NGOẠI VI

## 5.1 Kết nối dây giữa PLC và các thiết bị ngoại vi

Việc kết nối dây giữa PLC với ngoại vi rất quan trọng. Nó quyết định đến việc PLC có thể giao tiếp được với thiết bị lập trình (máy tính) cũng như hệ thống điều khiển có thể hoạt động đúng theo yêu cầu được thiết kế hay không. Ngoài ra việc nối dây còn liên quan đến an toàn cho PLC cũng như hệ thống điều khiển.

### 5.1.1 Giới thiệu CPU 224 và cách kết nối với thiết bị ngoại vi

Sơ đồ bề mặt của bộ điều khiển lập trình S7-200 CPU 224 được cho như hình 5.1.



Hình 5.1: Bộ điều khiển lập trình S7-200 CPU 224

Để cho bộ điều khiển lập trình này hoạt động được thì người sử dụng phải kết nối PLC với nguồn cung cấp và các ngõ vào ra của nó với thiết bị ngoại vi. Muốn nạp chương trình vào CPU, người sử dụng phải soạn thảo chương trình bằng các thiết bị lập trình hoặc máy tính với phần mềm tương ứng cho loại PLC đang sử dụng và có thể nạp trực tiếp vào CPU hoặc copy chương trình vào card nhớ để cắm vào rãnh cắm card nhớ trên CPU của PLC. Thông thường khi lập trình cũng như khi kiểm tra hoạt động của PLC thì người lập trình thường kết nối trực tiếp thiết bị lập trình hoặc máy tính cá nhân

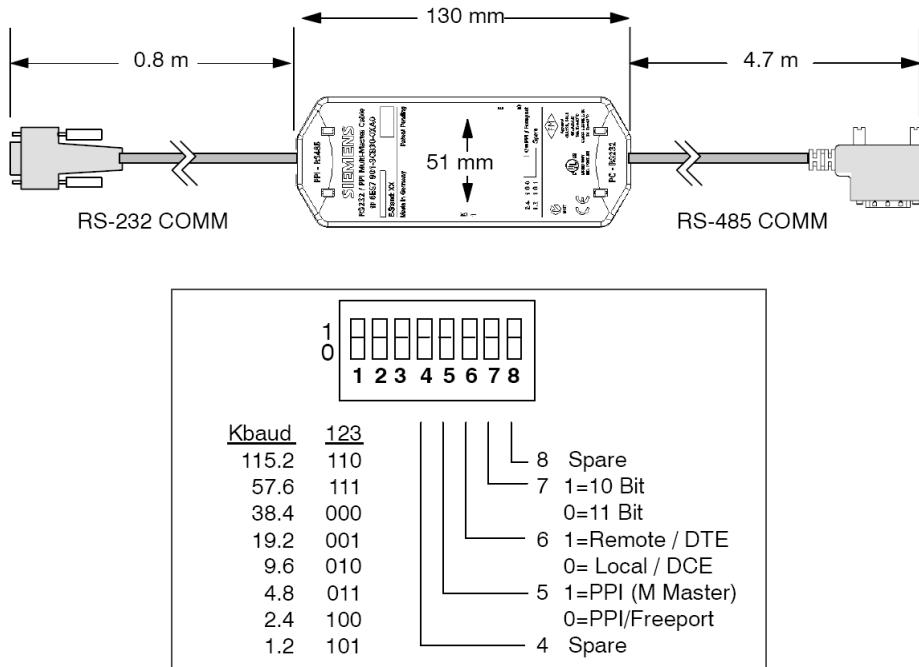
với PLC. Như vậy, để hệ thống điều khiển khiển bằng PLC hoạt động cũng như lập trình cho nó, cần phải kết nối PLC với máy tính cũng như các ngõ vào ra với ngoại vi.

### 5.1.2 Kết nối với máy tính

Đối với các thiết bị lập trình của hãng Siemens có các cổng giao tiếp PPI thì có thể kết nối trực tiếp với PLC thông qua một sợi cáp. Tuy nhiên đối với máy tính cá nhân cần thiết phải có cáp chuyển đổi PC/PPI. Có 2 loại cáp chuyển đổi là cáp RS-232/PPI Multi-Master và cáp USB/PPI Multi-Master.

#### \* Cáp RS-232/PPI multi-master:

Hình dáng của cáp và công tắc chọn chế độ truyền được cho ở hình 5.2.



Hình 5.2: Hình dáng cáp RS-232/PPI và các chuyển mạch trên cáp.

Tùy theo tốc độ truyền giữa máy tính và CPU mà các công tắc 1,2,3 được để ở vị trí thích hợp. Thông thường đối với CPU 22x thì tốc độ truyền thường đặt là 9,6 Kbaud (tức công tắc 123 được đặt theo thứ tự là 010).

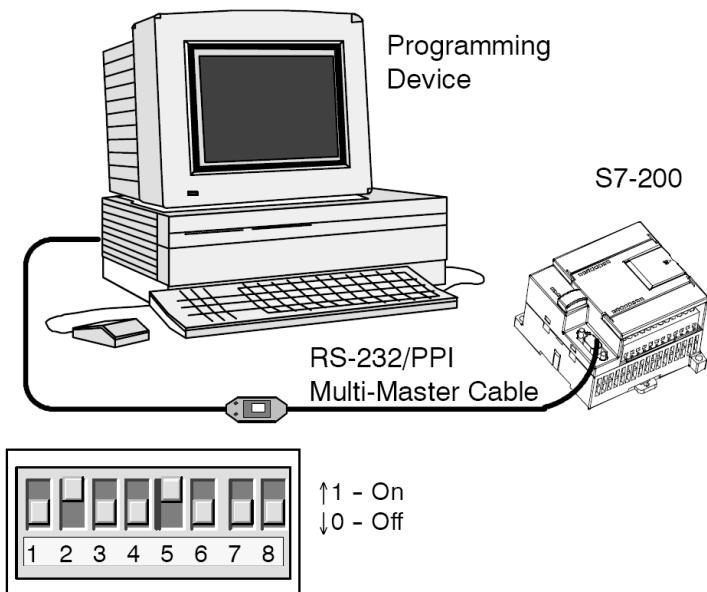
Tùy theo truyền thông là 10 Bit hay 11 Bit mà công tắc 7 được đặt ở vị trí thích hợp. Khi kết nối bình thường với máy tính thì công tắc 7 chọn ở chế độ truyền thông 11 Bit (công tắc 7 đặt ở vị trí 0).

Công tắc 6 ở cáp RS-232/PPI Multi-Master được sử dụng để kết nối port truyền thông RS-232 của một modem với S7-200 CPU. Khi kết nối bình thường với máy tính thì công tắc 6 được đặt ở vị trí data Communications Equipment (DCE) (công tắc 6 ở vị trí 0). Khi kết nối cáp PC/PPI với một

modem thì port RS-232 của cáp PC/PPI được đặt ở vị trí Data Terminal Equipment (DTE) (công tắc 6 ở vị trí 1).

Công tắc 5 được sử dụng để đặt cáp RS-232/PPI Multi-Master thay thế cáp PC/PPI hoặc hoạt động ở chế độ Freeport thì đặt ở chế độ PPI/Freeport (công tắc 5 ở vị trí 0). Nếu kết nối bình thường là PPI (master) với phần mềm STEP 7 Micro/WIN 3.2 SP4 hoặc cao hơn thì đặt ở chế độ PPI (công tắc 5 ở vị trí 1).

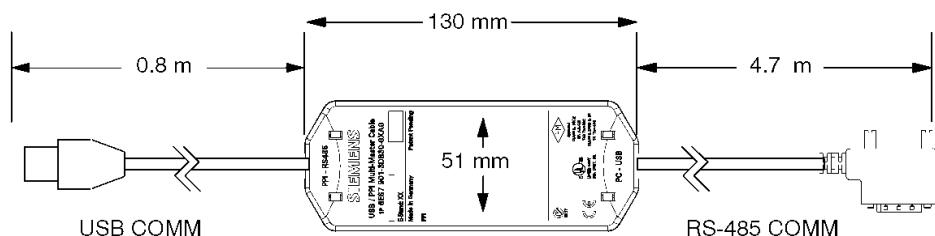
Sơ đồ nối cáp RS-232/PPI Multi-Master giữa máy tính và CPU S7-200 với tốc độ truyền 9,6 Kbaud được cho như hình 5.3.



Hình 5.3: Kết nối máy tính với CPU S7-200 RS-232/PPI Multi-Master

#### \* Cáp USB/PPI multi-master:

Hình dáng của cáp được cho ở hình 5.4.



Hình 5.4: Hình dáng cáp USB/PPI.

Cách thức kết nối cáp USB/PPI Multi-Master cũng tương tự như cáp RS-232/PPI Multi-Master. Để sử dụng cáp này, phần mềm cần phải là STEP 7-

Micro/WIN 3.2 Service Pack 4 (hoặc cao hơn). Cáp chỉ có thể được sử dụng với loại CPU22x hoặc sau này. Cáp USB không được hỗ trợ truyền thông Freeport và download cấu hình màn TP070 từ phần mềm TP Designer.

### 5.1.3 Nối nguồn cung cấp cho CPU

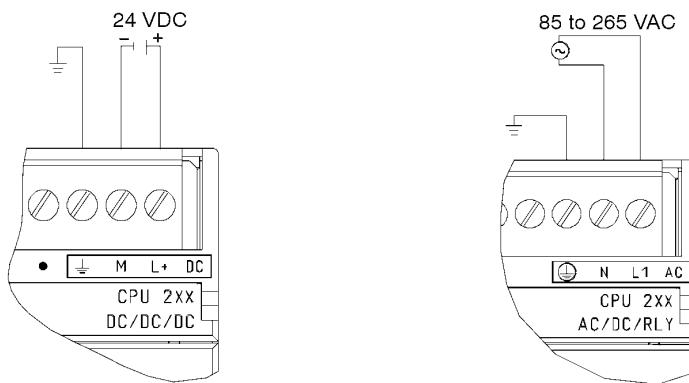
Tùy theo loại và họ PLC mà các CPU có thể là khối riêng hoặc có đặt sẵn các ngõ vào và ra cũng như một số chức năng đặc biệt khác. Hầu hết các PLC họ S7-200 được nhà sản xuất lắp đặt các khâu vào, khâu ra và CPU trong cùng một vỏ hộp. Nhưng nguồn cung cấp cho các khâu này hoàn toàn độc lập nhau. Nguồn cung cấp cho CPU của họ S7-200 có thể là:

Xoay chiều: 20...29 VAC , f = 47...63 Hz;

85...264 VAC, f = 47...63 Hz

Một chiều: 20,4 ... 28,8 VDC

Hình 5.5 a,b là sơ đồ nối dây nguồn cung cấp cho CPU



a. Cáp nguồn cho CPU 2xx loại DC/DC/DC;      b. Cáp nguồn cho CPU 2xx loại AC/DC/RLY

Hình 5.5: Nối nguồn cung cấp cho CPU

Để có thể nhận biết việc cấp nguồn cho CPU, khối vào, khối ra số ta căn cứ vào các chữ số đi kèm theo CPU. Các mã số kèm theo CPU 2xx có thể có như sau:

- CPU 2xx DC/DC/DC: Nguồn cấp cho CPU là DC, nguồn cho ngõ vào là DC, nguồn cấp cho ngõ ra là DC.
- CPU 2xx AC/DC/Relay: Nguồn cấp cho CPU là AC, nguồn cho ngõ vào là DC, ngõ ra là Relay có thể cấp nguồn là DC hoặc AC.

### 5.1.4 Kết nối vào/ra số với ngoại vi

Các ngõ vào, ra của PLC cần thiết để điều khiển và giám sát quá trình điều khiển. Các ngõ vào và ra có thể được phân thành 2 loại cơ bản: số (Digital) và tương tự (analog). Hầu hết các ứng dụng sử dụng các ngõ vào/ra số. Trong bài này chỉ đề cập đến việc kết nối các ngõ vào/ra số với ngoại vi, còn đối với *ngõ vào/ra tương tự sẽ trình bày ở chương “xử lý tín hiệu analog”*.

Đối với bộ điều khiển lập trình họ S7-200, hãng Siemens đã đưa ra rất nhiều loại CPU với điện áp cung cấp cho các ngõ vào ra khác nhau. Tùy thuộc từng loại CPU mà ta có thể nối dây khác nhau. Việc thực hiện nối dây cho CPU có thể tra cứu sổ tay kèm theo của hãng sản xuất.

#### 5.1.4.1 Kết nối các ngõ vào số với ngoại vi

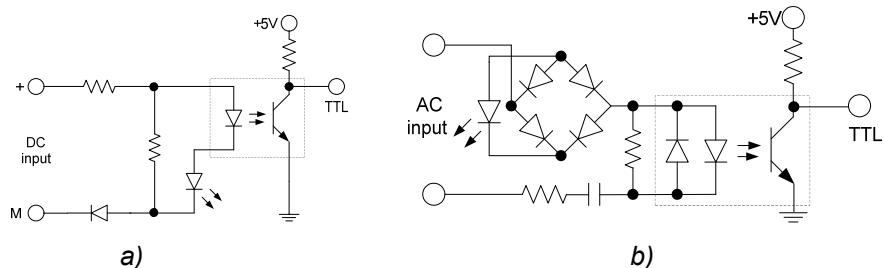
Các ngõ vào số của PLC có thể được chế tạo là một khối riêng, hoặc kết hợp với các ngõ ra chung trong một khối hoặc được tích hợp trên khối CPU. Trong trường hợp nào cũng vậy, các ngõ vào cũng phải được cung cấp nguồn riêng với cấp điện áp tùy thuộc vào loại ngõ vào. Cần lưu ý trong một khối ngõ vào cũng như các ngõ vào được tích hợp sẵn trên CPU có thể có các nhóm được cung cấp nguồn độc lập nhau. Vì vậy cần lưu ý khi cấp nguồn cho các nhóm này. Nguồn cung cấp cho các khối vào của họ S7-200 có thể là:

Xoay chiều: 15...35 VAC, f = 47...63 Hz; dòng cần thiết nhỏ nhất 4mA

79...135 VAC, f = 47...63 Hz; dòng cần thiết nhỏ nhất 4mA

Một chiều: 15 ... 30 VDC; dòng cần thiết nhỏ nhất 4mA

Sơ đồ mạch điện bên trong của một số ngõ vào được cho như hình 5.6a,b.



Hình 5.6:  
a) Mạch điện của 1 ngõ vào số sử dụng nguồn cung cấp DC  
b) Mạch điện của 1 ngõ vào số sử dụng nguồn cung cấp AC

Tùy theo yêu cầu mà có thể quyết định sử dụng loại ngõ vào nào.

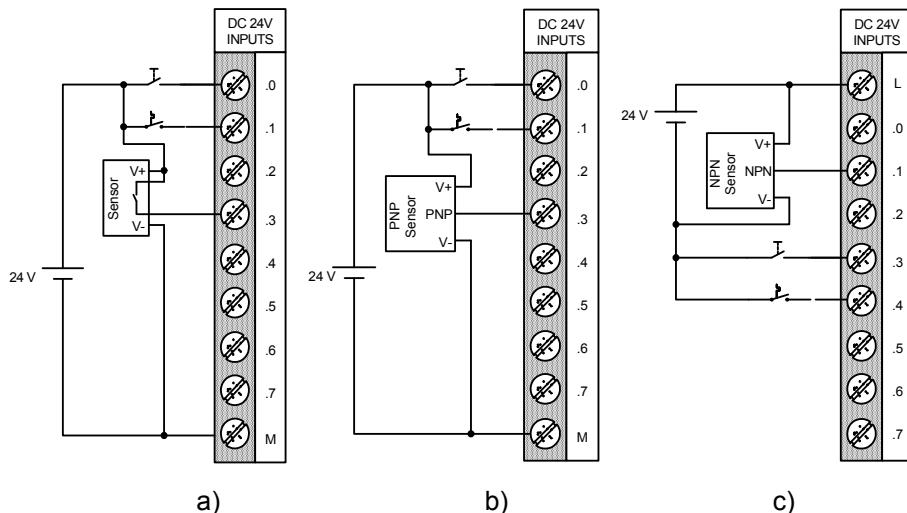
- + Ngõ vào DC:
  - Điện áp DC thường thấp do đó an toàn hơn.
  - Đáp ứng ngõ vào DC rất nhanh.
  - Điện áp DC có thể được kết nối với nhiều phần tử trong hệ thống điện.

- + Ngõ vào AC:
- Ngõ vào AC yêu cầu cần phải có thời gian. Ví dụ đối với điện áp có tần số 50 Hz phải yêu cầu thời gian đến 1/50 giây mới nhận biết được.
  - Tín hiệu AC ít bị nhiễu hơn tín hiệu DC, vì vậy chúng thích hợp với khoảng cách lớn và môi trường nhiễu (từ).
  - Nguồn AC kinh tế hơn.
  - Tín hiệu AC thường được sử dụng trong các thiết bị tự động hiện hữu.

Đối với các ngõ vào số, khi kết nối với ngoại vi, ngoại trừ các trường hợp đặc biệt thì thông thường mỗi một ngõ vào được kết nối với một bộ tạo tín hiệu nhị phân như: nút nhấn, công tắc, cảm biến tiếp cận .... Hình 5.7a,b,c minh họa cách kết nối dây các ngõ vào PLC với các bộ tạo tín hiệu nhị phân khác nhau.

Cần lưu ý đến các loại cảm biến khi kết nối với các ngõ vào PLC (xem lại chương 3: cảm biến và cơ cấu chấp hành trong điều khiển logic).

Trong ví dụ hình 5.7a có 3 ngõ vào, một là nút nhấn thường mở, hai là tiếp điểm của relay nhiệt, và ba là cảm biến tiếp cận với ngõ ra là relay. Cả ba bộ tạo tín hiệu này được cung cấp bởi một nguồn 24VDC. Khi tiếp điểm mở hoặc cảm biến phát tín hiệu “0” thì không có điện áp tại các ngõ vào. Nếu các tiếp điểm đóng lại hoặc cảm biến phát tín hiệu “1” thì ngõ vào được cấp điện.



Hình 5.7: Kết nối ngõ vào với ngoại vi.

- Nút nhấn và cảm biến có ngõ ra là relay nối với ngõ vào loại sinking.
- Nút nhấn và cảm biến loại PNP nối với ngõ vào loại sinking.
- Nút nhấn và cảm biến loại NPN nối với ngõ vào loại sourcing.

Đối với các ngõ vào ra của CPU 214 DC/DC/DC, CPU 224 AC/DC/Relay theo số tay được kết nối như hình 5.10 và hình 5.11.

### 5.1.4.2 Kết nối các ngõ ra số với ngoại vi

Các ngõ ra của PLC có thể được chế tạo là một khối riêng, hoặc kết hợp với các ngõ ra chung trong một khối hoặc được tích hợp trên khối CPU. Trong trường hợp nào cũng vậy, các ngõ ra cũng phải được cung cấp nguồn riêng với cáp điện áp tùy thuộc vào loại ngõ ra. Cần lưu ý trong một khối ra cũng như các ngõ ra được tích hợp sẵn trên CPU có thể có các nhóm được cung cấp nguồn độc lập nhau. Vì vậy cần lưu ý khi cấp nguồn cho các nhóm này. Nguồn cung cấp cho các khối ra của họ S7-200 có thể là:

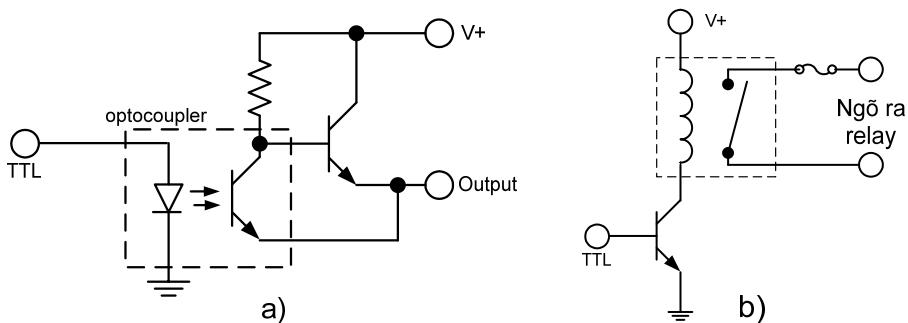
Xoay chiều: 20...264 VAC, f = 47...63 Hz;

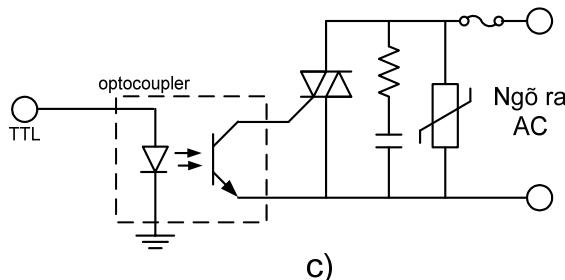
Một chiều: 5...30 VDC đối với ngõ ra rơ le; 20.4 ... 28.8 VDC đối với ngõ ra transistor;

Các khối ra tiêu chuẩn của PLC thường có 8 đến 32 ngõ ra theo cùng loại và có dòng định mức khác nhau. Ngõ ra có thể là rơ le, transistor hoặc triac. Rơ le là ngõ ra linh hoạt nhất. Chúng có thể là ngõ ra AC và DC. Tuy nhiên đáp ứng của ngõ ra rơ le chậm, giá thành cao và bị hư hỏng sau vài triệu lần đóng cắt. Còn ngõ ra transistor thì chỉ sử dụng với nguồn cung cấp là DC và ngõ ra triac thì chỉ sử dụng được với nguồn AC. Tuy nhiên đáp ứng của các ngõ ra này nhanh hơn.

Sơ đồ mạch điện bên trong của các ngõ ra được cho như hình 5.8.

Cần chú ý khi thiết kế hệ thống có cả hai loại ngõ ra AC và DC. Nếu nguồn AC nối vào ngõ ra DC là transistor, thì chỉ có bán kỵ dương của chu kỳ điện áp được sử dụng và do đó điện áp ra sẽ bị giảm. Nếu nguồn DC được nối với ngõ ra AC là triac thì khi có tín hiệu cho ngõ ra, nó sẽ luôn luôn có điện cho dù có điều khiển tắt bằng PLC.





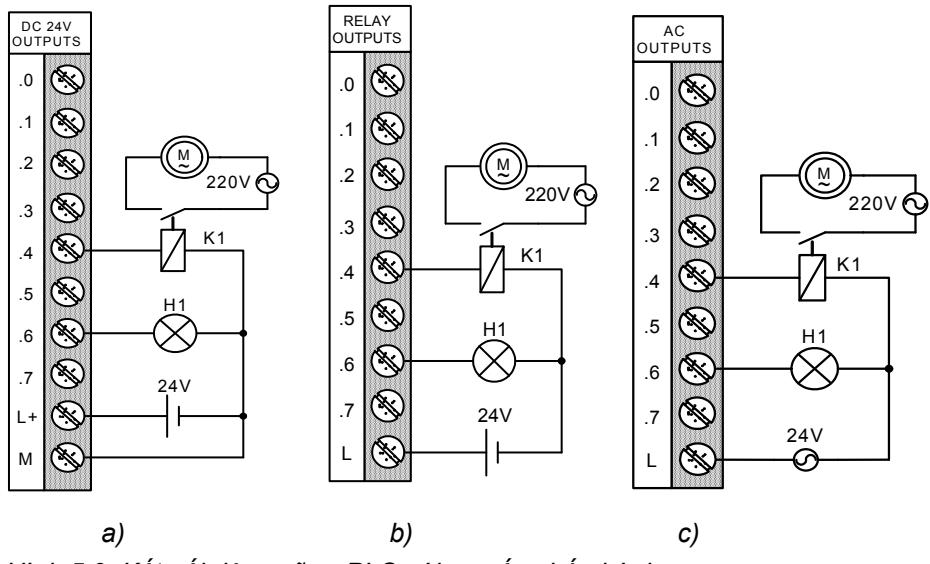
Hình 5.8: Mạch điện bên trong của các loại ngõ ra khác nhau.

a) Ngõ ra transistor ; b) Ngõ ra relay ; c) Ngõ ra triac

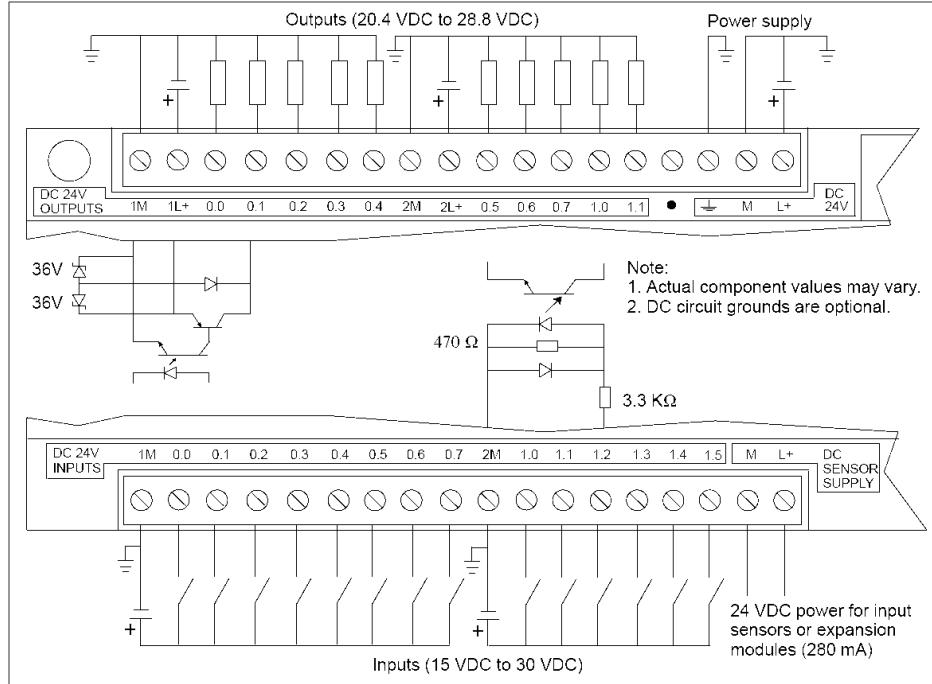
Đối với các ngõ ra số, khi kết nối với ngoại vi, ngoại trừ các trường hợp đặc biệt thì thông thường mỗi một ngõ ra được kết nối với một đối tượng điều khiển nhận tín hiệu nhị phân như: đèn báo, cuộn dây rơ le, chuông báo . . . . Hình 5.9 minh họa cách kết nối dây các ngõ ra PLC với các cơ cầu chìa hành. Hình 5.9a là một ví dụ cho các khối ra sử dụng 24Vdc với mass chung. Tiêu biểu cho loại này là ngõ ra transistor. Trong ví dụ này các ngõ ra được kết nối với tải công suất nhỏ là đèn báo và cuộn dây relay. Quan sát mạch kết nối này, đèn báo sử dụng nguồn cung cấp là 24Vdc. Nếu ngõ ra .6 ở mức logic “1” (24Vdc) thì dòng sẽ chảy từ ngõ ra .6 qua đèn H1 và xuống Mass (M), đèn sáng. Nếu ngõ ra ở mức logic “0” (0V), thì đèn H1 tắt. Nếu ngõ ra .4 ở mức logic “1” thì cuộn dây rơ le có điện, làm tiếp điểm của nó đóng lại cung cấp điện 220 Vac cho động cơ.

Hình 5.9b là một ví dụ ngõ ra relay sử dụng nguồn cung cấp là 24 Vdc, và hình 5.9c là ví dụ ngõ ra triac sử dụng nguồn xoay chiều 24 Vac.

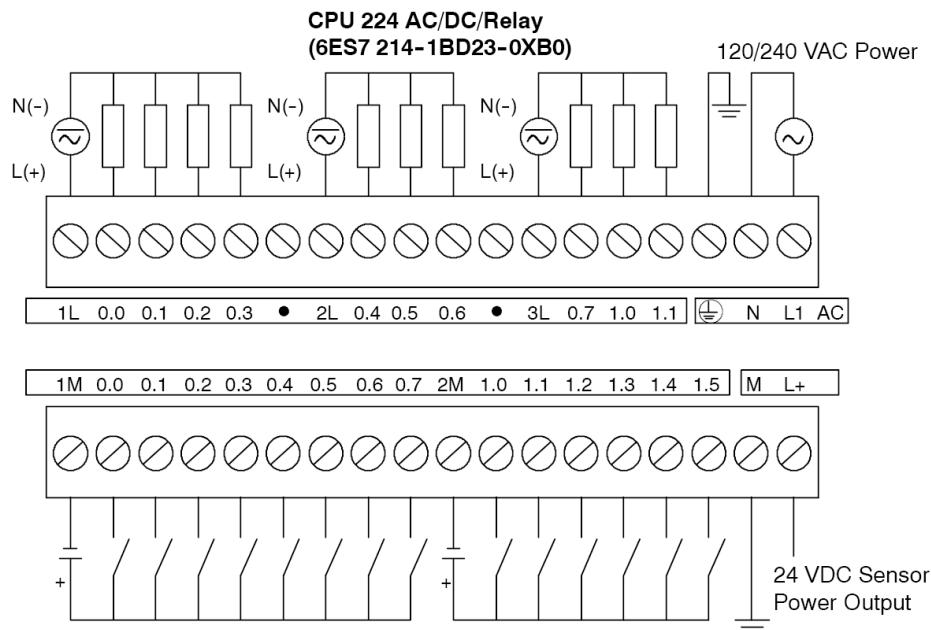
Một chú ý quan trọng khi kết nối các ngõ ra cần tra cứu sổ tay khối ngõ ra hiện có để có được thông tin chính xác tránh được những sự cố đáng tiếc xảy ra. Hình 5.10 là ví dụ của CPU 214 với nguồn cung cấp DC, ngõ vào DC và ngõ ra DC được kết nối dây với ngoại vi ( trích từ sổ tay S7-200 Programmable Controller System Manual). Ta nhận thấy mỗi một nhóm ngõ vào cũng như một nhóm ngõ ra và CPU được cung cấp nguồn riêng là 24 Vdc. Ngoài ra trên khối CPU còn có nguồn phụ 24 Vdc (đến 280 mA) có thể được sử dụng để cung cấp cho các cảm biến hoặc khối mở rộng.



Hình 5.9: Kết nối dây ngõ ra PLC với cơ cấu chấp hành



Hình 5.10: Sơ đồ nối dây CPU 214 DC/DC/DC với nguồn và ngoại vi



Hình 5.11: Sơ đồ nối dây CPU 224 AC/DC/Relay với nguồn và ngoại vi

## 5.2 Kiểm tra việc kết nối dây bằng phần mềm

Một công việc quan trọng cho người lắp đặt và vận hành là biết được các kết nối của các ngõ vào/ra với ngoại vi có đúng hay không trước khi nạp chương trình điều khiển vào CPU. Hoặc khi một hệ thống đang hoạt động bình thường nhưng một sự cố hư hỏng xảy ra thì các phần ngoại vi nào bị hư và phát hiện nó bằng cách nào. Các phần mềm cho các bộ điều khiển bằng PLC thường có trang bị thêm công cụ để kiểm tra việc kết nối dây ngõ vào/ra với ngoại vi. Trong phần mềm Step 7 Micro/Win (phần mềm lập trình cho họ S7-200) có trang bị thêm phần này đó là mục **Status Chart**.

Để sử dụng phần mềm tốt hơn hãy xem thêm chương “**Phần mềm STEP 7-Micro/Win và ngôn ngữ lập trình**”.

### 5.2.1 Status Chart

Chúng ta có thể sử dụng Status Chart để đọc, ghi hoặc cưỡng bức các biến trong chương trình theo mong muốn. Để có thể mở Status Chart, ta nhấp

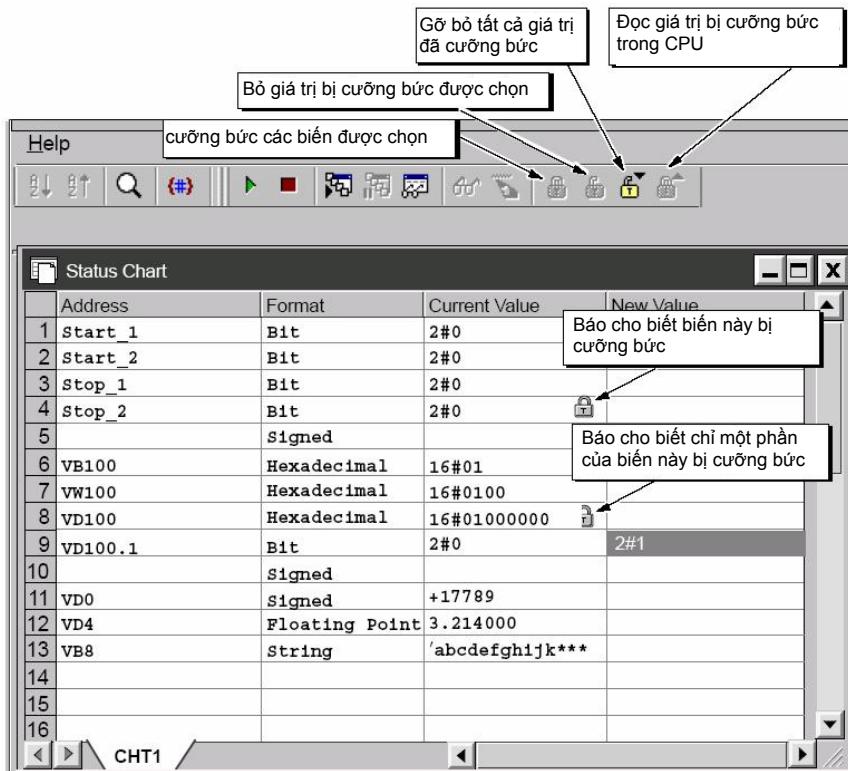


đúp chuột vào biểu tượng Status Chart trong cửa sổ Navigation Bar trên màn hình Step 7-Micro/Win32 hoặc vào mục View → Component → Status Chart.

### 5.2.2 Giám sát và thay đổi biến với Status Chart

Hình 5.9 chỉ một ví dụ về cách sử dụng Status Chart. Để đọc hay ghi các biến chúng ta thực hiện theo các bước sau:

- Bước 1:** Ở ô đầu tiên trong cột Address ta nhập vào địa chỉ hay tên ký hiệu của một biến trong chương trình ứng dụng mà muốn giám sát hoặc điều khiển, sau đó ấn ENTER. Lặp lại bước này cho tất cả các biến được thêm vào biểu đồ.
- Bước 2:** Nếu biến là 1 Bit (ví dụ:I, Q, hoặc M), thì kiểu biến đặt ở cột Format là bit. Nếu biến là một byte, word, hay double word thì chọn ở cột Format và nhấp đúp chuột để tìm kiểu biến mong muốn.
- Bước 3:** Để xem giá trị hiện hành của các biến trong PLC trong biểu đồ, hãy nhấp chuột vào biểu tượng  hoặc chọn *Debug → Chart Status*. Để chụp được một giá trị của các biến tại thời điểm nhấp chuột sử dụng *Debug → Single Read* hoặc nhấp chuột vào biểu tượng .
- Bước 4:** Để dừng việc giám sát thì nhấp chuột vào biểu tượng  hoặc chọn *Debug → Chart Status*.
- Bước 5:** Để thay đổi giá trị của một biến hoặc nhiều biến, hãy nhập giá trị mới vào cột “New Value” cho các biến mong muốn và nhấp chuột vào biểu tượng  hoặc chọn *Debug → Write All* để ghi tất cả các giá trị này vào các biến tương ứng trong CPU.



Hình 5.13: Ví dụ về status chart

### 5.2.3 Cưỡng bức biến với Status Chart

Trong một số trường hợp cần thiết phải ép buộc một ngõ vào hoặc một ngõ ra hoặc bất kỳ một biến nào đó trong chương trình theo một giá trị mong muốn cho phù hợp với hoàn cảnh hoạt động hiện tại của hệ thống hoặc để kiểm tra các lỗi xảy ra trong hệ thống điều khiển, ta có thể sử dụng công cụ cưỡng bức biến (Force).

Để cưỡng bức biến trong Status Chart với một giá trị xác định, thực hiện các bước sau:

**Bước 1:** Chọn một ô trong cột Address, vào địa chỉ hay tên của biến cần cưỡng bức.

**Bước 2:** Nếu biến là 1 Bit (ví dụ:I0.0, Q0.1), thì kiểu biến ở cột Format luôn luôn là bit. Nếu biến là một byte, word, hay double word thì chọn ở cột Format và nhấp đúp chuột để tìm kiểu biến mong muốn.

**Bước 3:** Để cưỡng bức biến với giá trị hiện hành, trước tiên hãy đọc giá trị hiện hành trong PLC bằng cách nhấp chuột vào biểu tượng hoặc chọn Debug → Chart Status.

Nhấp hoặc cuộn ô chứa giá trị hiện hành muốn cưỡng bức. Nhấp chuột vào biểu tượng hoặc chọn *Debug → Force* ở trên vị trí giá trị hiện hành để cưỡng bức biến giá trị đó.

**Bước 4:** Để cưỡng bức một giá trị mới cho một biến, nhấp giá trị vào cột “New Value” và nhấp chuột vào biểu tượng hoặc chọn *Debug → Force*.

**Bước 5:** Để xem giá trị hiện hành của tất cả các biến bị cưỡng bức, kích chuột vào biểu tượng *Read All Forced* hoặc chọn *Debug → Read All Forced*.

**Bước 6:** Để cho tất cả các biến trở lại trạng thái bình thường, hãy kích chuột vào biểu tượng *Unforce All* hoặc chọn *Debug → Unforce All*. Muốn gỡ bỏ cưỡng bức một biến, hãy chọn biến mong muốn và nhấp chuột vào biểu tượng hoặc chọn *Debug → Unforce*.

#### 5.2.4 Ứng dụng Status Chart trong việc kiểm tra kết nối dây trong S7-200

Sau khi kết nối dây ngoại vi với các ngõ vào/ra của PLC, việc kế tiếp là kiểm tra lại kết nối dây này để phát hiện ra các lỗi kết nối. Một công cụ hữu hiệu là sử dụng Status Chart. Lưu ý khi kiểm tra kết nối dây:

- **Đối với ngõ vào:**

- Các ngõ vào nào được nối với các tiếp điểm thường đóng hay tín hiệu có mức logic “1” thì các ngõ vào có điện áp và đèn báo trạng thái các ngõ vào sáng. Khi quan sát trong status chart, ta sẽ nhận thấy các giá trị này có mức logic “1”.
- Việc kiểm tra các ngõ vào nên thực hiện lần lượt cho từng ngõ vào theo bảng kết nối dây vào/ra với ngoại vi. Có nghĩa là mỗi lần ta chỉ thay đổi trạng thái của một bộ tạo tín hiệu (nút nhấn, cảm biến,...) và quan sát trạng thái của ngõ vào được kết nối với nó trong status chart.
- Ghi chép lại các kết nối bị sai và sửa chữa.

- **Đối với ngõ ra:**

- Ở trạng thái bình thường khi chưa có chương trình thì tất cả các ngõ ra của PLC đều ở mức logic “0” (không có điện áp) và đèn báo trạng thái các ngõ ra đều tắt.
- Việc kiểm tra nối dây ngõ ra nên thực hiện lần lượt từng ngõ ra theo bảng kết nối dây bằng cách cho ngõ ra muốn kiểm tra lên mức logic “1” trong status chart và quan sát trạng thái của ngoại vi được kết nối tương ứng. Nếu ngoại vi tương ứng có điện chứng tỏ nó được kết nối đúng còn ngược lại kết nối sai.

- Ghi chép lại các kết nối sai và sửa chữa.

### 5.3 Câu hỏi và bài tập

**BT 5.1:** Ngõ vào của PLC có thể đóng điện cho cuộn dây rơ le để điều khiển một động cơ được không? Các khối vào và khối ra đóng vai trò gì trong việc giao tiếp giữa PLC và thiết bị ngoại vi?

**BT 5.2:** Các khối mở rộng ngõ vào/ra có lợi ích gì?

**BT 5.3:** Điều gì xảy ra nếu một ngõ ra AC được cấp nguồn DC?

**BT 5.4:** Một khối vào/ra mở rộng của PLC họ S7-200 loại EM223 gồm có 8 ngõ vào DC/8 ngõ ra rơle. Các ngõ vào được nối với 4 nút nhấn, 2 ngõ ra được nối với một rơle trung gian sử dụng nguồn 24VDC dùng để đóng mạch cho một contactor 220VAC để điều khiển động cơ 3 pha 220V/380V. 2 ngõ ra được nối với 2 đèn báo 220VAC để báo chiều quay của động cơ. 2 ngõ ra được sử dụng cho các van khí nén 24 VDC. Hãy vẽ sơ đồ nối dây các ngõ vào và ra này với ngoại vi theo yêu cầu.

**BT 5.5:** Hãy thiết kế một dự án được điều khiển bằng PLC. Trước khi đặt hàng, cần phải phác thảo việc nối dây cơ bản và chọn lựa các loại PLC hoặc khối vào/ra có các ngõ vào/ra tương ứng. Các thiết bị được sử dụng để nối với các ngõ vào gồm có: 2 công tắc hành trình, 1 nút nhấn thường hở, 1 nút nhấn thường đóng và một tiếp điểm nhiệt. Ngõ ra sẽ điều khiển một van solenoid 24VDC, một đèn báo 110VAC và một động cơ 220VAC/50HP. Hãy lựa chọn loại PLC hoặc một khối vào/ra phù hợp và kết nối dây theo yêu cầu đặt ra.

**BT 5.6:** Hãy phác thảo sơ đồ nối dây cho các ngõ ra PLC theo yêu cầu được liệt kê dưới đây:

- Một van khí nén có 2 cuộn dây solenoid
- Một đèn báo 24VDC
- Một đèn báo 120 VAC
- Một động cơ công suất thấp 12 VDC.

## 6 Phần mềm Micro/Win và ngôn ngữ lập trình

### 6.1 Cài đặt phần mềm STEP 7-Micro/WIN

STEP 7-Micro/WIN là một phần mềm lập trình cho họ PLC S7-200. Hiện phiên bản đang được sử dụng là STEP 7-Micro/Win V4.0 Service Pack 6.

#### 6.1.1 Yêu cầu hệ điều hành và phần cứng

Máy tính cá nhân PC, muốn cài đặt được phần mềm STEP 7-micro/WIN phải thỏa mãn những yêu cầu sau đây:

- Microsoft Windows 2000 Service Pack 3 hoặc cao hơn, Windows XP Home, hoặc Windows XP Professional.
- Có ít nhất 350 MB ổ đĩa cứng còn trống
- Sử dụng chế độ cài đặt font chữ nhỏ độ phân giải màn hình tối thiểu là 1024x768 pixels.

Nếu chưa có cáp để kết nối máy tính với PLC S7-200 thì ta vẫn có thể soạn thảo chương trình ở chế độ offline và kiểm tra hoạt động của chương trình với một phần mềm mô phỏng.

Để truyền thông với S7-200, ta cần một trong các phần cứng sau:

- PC/PPI Cable kết nối CPU S7-200 với PC qua cổng USB
- PC/PPI Cable kết nối CPU S7-200 với PC qua cổng RS232 (COM1 hoặc COM2)
- CP card (Communications processor) và cáp MPI (multipoint interface).
- EM241 modem
- CP243-1 hoặc CP243-1 IT Ethernet

#### 6.1.2 Cài đặt phần mềm

Thực hiện theo các bước sau:

1. Đóng tất cả các ứng dụng
2. Chèn đĩa CD STEP 7-Micro/Win vào ổ đĩa CD-Rom. Chương trình sẽ được tự động cài đặt. Ta cũng có thể khởi động chương trình cài đặt bằng cách nhấp đúp chuột vào file “Setup.exe” trên CD.

3. Sau đó sẽ nhận được dần dần từng bước các chỉ dẫn thao tác tiếp theo trên màn hình và hoàn thành công việc cài đặt.
4. Khi cài đặt xong, hộp thoại “set PG/PC Interface” tự động xuất hiện. Kích “Cancel” để kết thúc.
5. Ta cần khởi động lại máy để hoàn tất việc cài đặt.

Sau khi đã cài đặt xong có thể bắt đầu soạn thảo chương trình nhờ phần mềm STEP 7-Micro/WIN bằng cách nhấp đúp chuột vào biểu tượng STEP 7 MicroWIN trên màn hình.

**Chú ý:** Khi cài đặt phiên bản STEP 7-Micro/WIN V4.0 Service Pack 6 thì trước tiên ta cần phải uninstall phiên bản cũ và sau đó mới cài đặt được phiên bản này. Sau khi download ta nhấp đúp chuột vào file STEP7-MicroWIN\_V40\_SP6.exe và thực hiện theo các bước sau:

*Bước 1: Uninstall phiên bản STEP 7-Micro/WIN V4.0 bằng công cụ “control panel” trong Window (menu Start → settings → control panel → add or remove program).*

*Bước 2: Khởi động lại máy tính*

*Bước 3: Cài đặt STEP 7-Micro/WIN V4.0 Service Pack (SP6) bằng cách nhấp đúp chuột vào file STEP7-MicroWIN\_V40\_SP6.exe.*

## 6.2 Các phần tử cơ bản trong chương trình PLC S7-200

Các phần tử cơ bản trong một chương trình PLC S7-200 là:

1. Chương trình chính (main program)
2. Chương trình con (subroutine)
3. Chương trình ngắn (interrupt routine)
4. Khối hệ thống (system block)
5. Khối dữ liệu (data block)

### 6.2.1 Chương trình chính OB1 (main program)

Đây là phần khung của chương trình, chứa các lệnh điều khiển chương trình ứng dụng. Với một số chương trình điều khiển nhỏ, đơn giản chúng ta có thể viết tắt cả các lệnh trong khối này. Chương trình ứng dụng được xử lý bắt đầu từ chương trình chính, các lệnh được xử lý lần lượt từ trên xuống dưới và chỉ một lần ở mỗi vòng quét. Trong S7-200 chương trình được chứa trong khối OB1.

### 6.2.2 Chương trình con SUB (subroutine)

Các lệnh viết trong chương trình con chỉ có thể được xử lý khi chương trình con được gọi (Call) từ chương trình chính, từ một chương trình con khác hoặc từ một chương trình ngắn. Sử dụng chương trình con khi chúng ta muốn

phân chia nhiệm vụ điều khiển. Mỗi một chương trình con viết cho một nhiệm vụ nhỏ hoặc khi có các yêu cầu điều khiển tương tự nhau (ví dụ: điều khiển băng tải 1, điều khiển băng tải 2...) thì chúng ta chỉ cần tạo ra chương trình con một lần và có thể gọi ra nhiều lần từ chương trình chính.

Sử dụng chương trình con có một số ưu điểm sau:

- Chương trình điều khiển được chia theo nhiệm vụ điều khiển nên có cấu trúc rõ ràng, rất thuận tiện cho việc chỉnh sửa hay kiểm tra chương trình.
- Giảm thời gian vòng quét của chương trình. CPU không phải liên tục xử lý tất cả các lệnh của chương trình mà chỉ xử lý chương trình con khi có lệnh gọi tương ứng.
- Chương trình con cho phép giảm công việc soạn thảo khi có các yêu cầu điều khiển tương tự nhau.

(Bạn đọc xem phần ví dụ và cách sử dụng chương trình con ở chương “phép toán nhị phân”).

### 6.2.3 Chương trình ngắt INT(interrupt routine)

Chương trình ngắt được thiết kế để sử dụng cho một sự kiện ngắt được định nghĩa trước. Bất cứ khi nào sự kiện ngắt xác định xảy ra, thì S7-200 thực hiện chương trình ngắt.

Chương trình ngắt không được gọi bởi chương trình chính mà theo sự kiện ngắt xảy ra. Chương trình ngắt sẽ chỉ được xử lý mỗi khi sự kiện ngắt xảy ra.

(Phần chương trình ngắt sẽ được trình bày chi tiết ở tập 2).

### 6.2.4 Khối hệ thống (system block)

System block cho phép ta cấu hình các tùy chọn phần cứng khác nhau cho S7-200.

### 6.2.5 Khối dữ liệu (data block)

Data block lưu trữ các giá trị biến khác nhau (vùng nhớ V) được sử dụng trong chương trình. Giá trị ban đầu của các dữ liệu có thể nhập vào trong khối dữ liệu.

(Phần khối dữ liệu sẽ được trình bày chi tiết ở tập 2).

## 6.3 Ngôn ngữ lập trình

Để có thể soạn thảo chương trình cho các PLC S7-200, chúng ta dùng phần mềm Step7 MicroWin. Và cũng giống như PLC của các hãng khác, chúng ta có 3 dạng soạn thảo thông dụng là dạng LAD, FBD và STL. Việc chọn dạng soạn thảo nào để viết chương trình điều khiển là do người dùng tùy chọn.

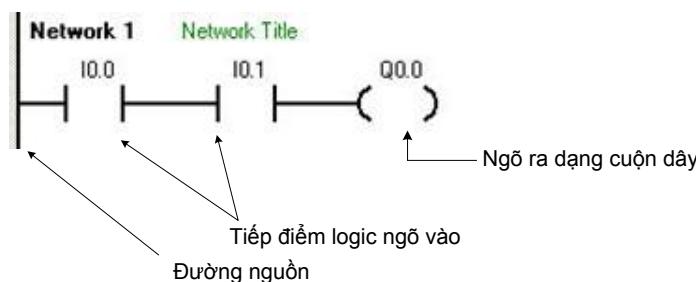
### 6.3.1 Dạng hình thang : LAD (Ladder logic)

Ở dạng soạn thảo này chương trình được hiển thị gần giống như sơ đồ nối dây một mạch trang bị điện dùng các relay và contactor. Chúng ta xem như có một dòng điện từ một nguồn điện chạy qua một chuỗi các tiếp điểm logic ngõ vào từ trái qua phải để tới ngõ ra. Chương trình điều khiển được chia ra làm nhiều Network, mỗi một Network thực hiện một nhiệm vụ nhỏ và cụ thể. Các Network được xử lý lần lượt từ trên xuống dưới và từ trái sang phải.

Các phần tử chủ yếu dùng trong dạng soạn thảo này là:

- Tiếp điểm không đảo:
- Tiếp điểm đảo:
- Ngõ ra (hoặc trạng thái nội của biến):
- Các hộp chức năng (Box): các chức năng được biểu diễn ở dạng hộp như các phép toán số học, định thời, bộ đếm...

Ví dụ:



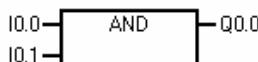
Dạng soạn thảo này có một số ưu điểm:

- Dễ dàng cho những người mới bắt đầu lập trình
- Biểu diễn dạng đồ họa dễ hiểu và thông dụng
- Luôn luôn có thể chuyển từ dạng STL sang LAD

### 6.3.2 Dạng khối chức năng : FBD (Function Block Diagram)

Dạng soạn thảo FBD hiển thị chương trình ở dạng đồ họa tương tự như sơ đồ các cổng logic. FBD không sử dụng khái niệm đường nguồn cung cấp trái và phải; do đó khái niệm “dòng điện” không được sử dụng. Thay vào đó là logic “1”. Không có tiếp điểm và cuộn dây như ở dạng LAD, nhưng có các cổng logic và các hộp chức năng. Các cổng logic như AND, OR, XOR... sẽ tương ứng với các tiếp điểm logic nối tiếp hay song song...

Ví dụ:

**Network 1 Network Title**

Đầu ra của các cổng logic hay hộp chức năng có thể được sử dụng để nối tiếp với đầu vào của các cổng logic hay các hộp chức năng khác. Với dạng soạn thảo này có một số điểm chính sau:

- Biểu diễn ở dạng đồ họa các cổng chức năng giúp chúng ta dễ đọc hiểu theo trình tự điều khiển.
- Luôn có thể chuyển từ hiển thị dạng FBD sang STL.

**6.3.3 Dạng liệt kê lệnh : STL (Statement List)**

Đây là dạng soạn thảo chương trình dạng tập hợp các câu lệnh. Người dùng phải nhập các câu lệnh từ bàn phím, giữa lệnh và toán hạng (toán hạng có thể là địa chỉ, dữ liệu) có khoảng trắng và mỗi lệnh chiếm một hàng. Ở dạng soạn thảo này sẽ có một số chức năng mà ở dạng soạn thảo LAD hay FBD không có.

Ví dụ:

<b>Network 1</b>	<b>Network Title</b>
LD        I0.0	// Doc trang thai I0.0
A        I0.1	// Thuc hien AND voi I0.1
=        Q0.0	// Xuat ket qua ra Q0.0

Dạng soạn thảo này có một số điểm chính:

- Là dạng soạn thảo phù hợp cho những người có kinh nghiệm lập trình PLC.
- STL cho phép giải quyết một số vấn đề mà đôi khi khó khăn khi dùng LAD hoặc FBD.
- Luôn luôn có thể chuyển từ dạng LAD hay FBD về dạng STL nhưng khi chuyển ngược lại từ STL sang LAD hay FBD sẽ có một số phần tử chương trình không chuyển được.

**6.4 Soạn thảo chương trình với phần mềm STEP7-Micro/Win V4.0 SP6****6.4.1 Mở màn hình soạn thảo chương trình**

Để mở STEP 7--Micro/WIN, nhấp đúp chuột vào biểu tượng STEP 7-

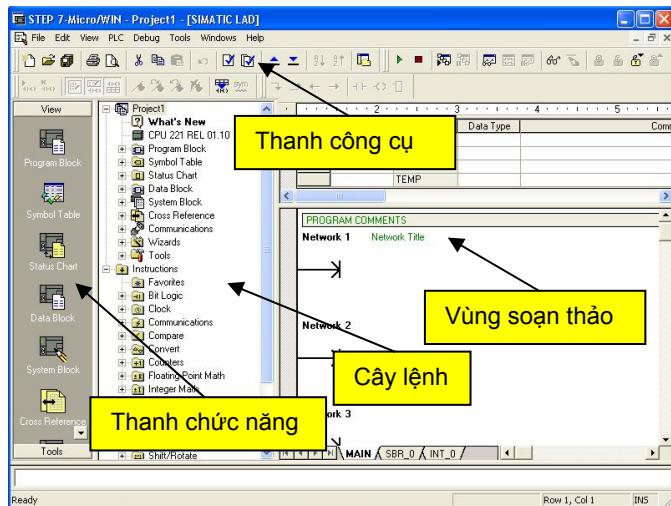
 Micro/WIN trên màn hình desktop, hoặc chọn Start > SIMATIC > STEP 7 MicroWIN V4.0. Giao diện màn hình có dạng (hình 6.1).

### 6.4.1.1 Vùng soạn thảo chương trình

Vùng soạn thảo chương trình chứa chương trình và bảng khai báo biến cục bộ của khối chương trình đang được mở. Chương trình con (viết tắt là SUB) và chương trình ngắn (viết tắt là INT) xuất hiện ở cuối cửa sổ soạn thảo chương trình. Tùy thuộc vào việc nhấp chuột ở mục nào mà cửa sổ màn hình soạn thảo chương trình tương ứng sẽ được mở.

### 6.4.1.2 Cây lệnh

Cây lệnh hiển thị tất cả các đối tượng của dự án và các lệnh để viết chương trình điều khiển. Có thể sử dụng phương pháp “drag and drop” (kéo và thả) từng lệnh riêng từ cửa sổ cây lệnh vào chương trình, hay nhấp đúp chuột vào một lệnh mà muốn chèn nó vào vị trí con trỏ ở màn hình soạn thảo chương trình.



Hình 6.1: Màn hình soạn thảo chương trình STEP 7-Micro/Win

### 6.4.1.3 Thanh chức năng

Thanh chức năng chứa một hóm các biểu tượng để truy cập các đặc điểm chương trình khác nhau của STEP 7--Micro/WIN.



\* **Program Block:** Program Block

Nhấp đúp chuột vào biểu tượng này để mở ra cửa sổ soạn thảo các chương trình ứng dụng (OB1, SUB hoặc INT)



\* **Symbol Table:** Symbol Table

Bảng ký hiệu (Symbol table) cho phép người dùng mô tả các địa chỉ sử dụng trong chương trình dưới dạng các tên gọi gợi nhớ. Điều này giúp cho

việc đọc hiểu chương trình dễ dàng và khi viết chương trình ít bị sai sót do sử dụng trùng địa chỉ.

		Symbol	Address	Comment
1		START	I0.0	nút nhấn khởi động
2		M1	Q0.0	Dong co 1
3				

Tên gọi nhớ  Địa chỉ tuyệt đối  Chú thích 



\* Status Chart:

Bảng trạng thái (Status chart) cho phép người dùng giám sát trạng thái các ngõ vào và thay đổi trạng thái từng ngõ ra. Sử dụng bảng trạng thái để kiểm tra nối dây phần cứng và xem nội dung các vùng nhớ.

	Address	Format	Current Value	New Value
1	I0.0	Bit		
2	Q0.0	Bit		2#1
3	Vw2	Unsigned		1200
4		Signed		

Trong đó:

- + Cột Address: Cho phép nhập địa chỉ các biến hay vùng nhớ
- + Cột Format: Cho phép chọn dạng dữ liệu của địa chỉ
- + Cột Current Value: Hiển thị giá trị hiện hành của địa chỉ
- + Cột New Value: Cho phép thay đổi trạng thái ngõ ra hay nội dung vùng nhớ



\* Data Block:

Sử dụng Data Block như một vùng nhớ để đặt trước dữ liệu cho các biến thuộc vùng nhớ V. Có thể tạo ra các Data block khác nhau và đặt tên theo dữ liệu chương trình. Ví dụ:



Cửa sổ soạn thảo dữ liệu:

```

//Du lieu ban chua 1 (Tank 1)
//
VD100 500.0          // So lit maximum
VD104                  // so lit hien hanh
VW106                  // chua ma loi
  
```





**\* System Block :**

Đây là khối chức năng hệ thống, khi mở System Block chúng ta có thể cài đặt các chức năng như:

- **Communication ports:** Chọn các thông số truyền thông với thiết bị khác như máy tính hay CPU khác.
- **Retentive Ranges:** Chọn các vùng nhớ và địa chỉ sẽ có thuộc tính retentive
- **Output Tables:** Cho phép thiết lập cấu hình trạng thái ON và OFF của mỗi ngõ ra số khi CPU chuyển từ trạng thái Run sang Stop.
- **Input filter:** Cho phép chọn thời gian trễ cho một vài ngõ vào hoặc tắt cả ngõ vào số (từ 0.2ms đến 12.8 ms). Mục đích là giúp chống nhiễu ở việc nối dây ngõ vào.
- **Pulse Catch Bits:** Cho phép thiết lập một ngõ vào để bắt lấy sự chuyển đổi trạng thái tín hiệu rất nhanh. Ngay khi có chuyển đổi, giá trị ngõ vào sẽ được chốt cho đến khi được đọc bởi chu kỳ quét của PLC.
- **Background Time:** Cho phép thiết lập lượng thời gian PLC sẽ dành cho các hoạt động nền trong chế độ RUN. Đặc điểm này được sử dụng chủ yếu để điều khiển ảnh hưởng của chu kỳ quét khi xử lý trạng thái và trong hoạt động soạn thảo runtime.
- **EM Configuration:** Các module intelligent và địa chỉ cấu hình tương ứng được định nghĩa trong dự án. Thường thì STEP 7-Micro/WIN wizard đặt các địa chỉ này.
- **Configure LED:** LED SF/DIAG (System Fault/Diagnostic) có thể được chọn sáng khi thực hiện chức năng cưỡng bức (Force) hoặc xảy ra lỗi vào/ra (I/O).
- **Increase Memory:** Tăng bộ nhớ chương trình bằng cách không cho soạn thảo ở chế độ RUN. Đối với bộ nhớ Dữ liệu thì không thể.
- **Password:** Cho phép đặt mật khẩu để bảo vệ chương trình. Có 4 cấp để người dùng tùy chọn theo bảng sau:

Mô tả chức năng	Level 1	Level 2	Level 3	Level 4
Đọc và ghi dữ liệu				
Start, Stop, khởi động CPU	Cho phép truy cập			
Đọc và ghi đồng hồ thời gian (time-of-day)				

Clock)			
Upload chương trình, dữ liệu, cấu hình CPU			Không bao giờ cho phép
Download chương trình, data block hoặc system block			Yêu cầu password (không bao giờ cho phép với system Block)
Soạn thảo ở Runtime			Không bao giờ cho phép
Xóa chương trình, data block hoặc system block	Yêu cầu password	Yêu cầu password	Yêu cầu password (không bao giờ cho phép với system Block)
Copy chương trình, data block hoặc system block vào card nhớ			Yêu cầu password
Cưỡng bức dữ liệu trong status chart			
Ghi ngõ ra ở trạng thái stop			
Xóa tốc độ quét trong PLC information			
So sánh dự án			Không bao giờ cho phép



**\* Cross Reference:**

Bảng tham chiếu cho biết những địa chỉ vùng nhớ nào (Byte, bit, word hay DWord, timer, counter...) đã sử dụng và vị trí (location) trong chương trình cũng như chức năng của chúng.

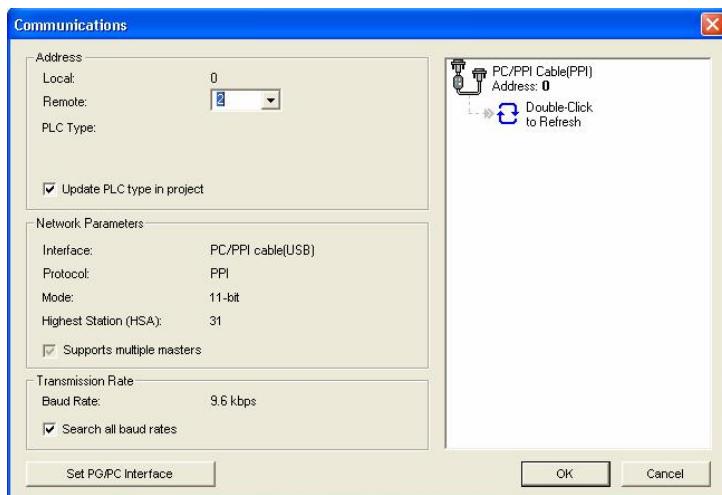
Một ví dụ bảng cross reference được cho ở hình 6.2. Tại cột Element, nhấp đúp vào địa chỉ nào thì trình soạn thảo sẽ mở cửa sổ chương trình có chứa địa chỉ tương ứng. Việc này giúp cho chúng ta dễ dàng kiểm tra hay thay đổi địa chỉ khi có nhu cầu.

	Element	Block	Location	Context
1	I0.0	MAIN (OB1)	Network 1	-II-
2	I0.0	MAIN (OB1)	Network 2	-II-
3	I0.2	MAIN (OB1)	Network 1	-II-
4	Q0.0	MAIN (OB1)	Network 2	-()
5	Q0.1	SBR_0 (SBR0)	Network 1	-()
6	C1	MAIN (OB1)	Network 1	CTU
7	T40	SBR_0 (SBR0)	Network 1	-II-

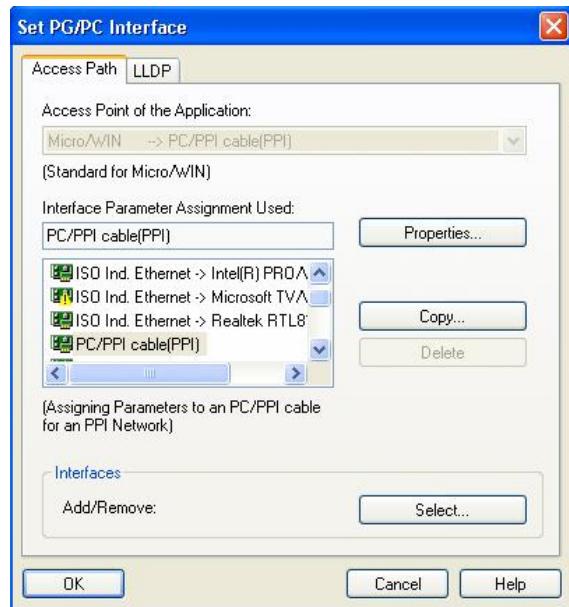
Hình 6.2: Ví dụ bảng cross reference.

**Communication:** Communications và Set PG/PC Interface

Các biểu tượng này khi kích hoạt sẽ mở ra hộp thoại cho phép chúng ta cài đặt các giao tiếp với máy tính như: chọn cổng giao tiếp, địa chỉ CPU, tốc độ truyền. Đây là *bước cần thực hiện* khi bắt đầu giao tiếp giữa PLC với máy tính.



Hình 6.3: Cửa sổ Communications



Hình 6.4: Cửa sổ Set PG/PC Interface.

#### 6.4.2 Thanh công cụ (Toolbar) trong STEP7-Micro/WIN

Trong phần mềm có đặt sẵn nhiều công cụ giúp người lập trình dễ dàng trong việc sử dụng. Các công cụ có ý nghĩa như sau:

- New Project (File menu): Khởi động một dự án mới
- Open Project (File menu): Mở một dự án tồn tại
- Save Project (File menu): Lưu dự án
- Print (File menu): In chương trình và tài liệu dự án
- Print Preview (File menu): Xem trước khi in
- Cut (Edit menu): Cắt phần chọn và đưa vào clipboard
- Copy (Edit menu): Copy phần được chọn vào clipboard
- Paste (Edit menu): Dán nội dung clipboard vào cửa sổ được kích hoạt
- Undo (Edit menu): Khôi phục lại phần bị xóa trước

-  Compile (PLC menu): Biên dịch cửa sổ được kích hoạt (Program Block hoặc Data Block).
-  Compile All (PLC menu): Biên dịch tất cả các phần tử dự án (Program Block, Data Block, and System Block)
-  Upload (File menu): Lấy (Upload) các phần tử dự án từ PLC vào màn hình soạn thảo chương trình
-  Download (File menu): Nạp (download) các phần tử dự án từ STEP7-MicroWin vào PLC.
-  Option (Tools menu): Truy cập menu Options
-  RUN (PLC menu): Đặt PLC ở chế độ RUN
-  STOP (PLC menu): Đặt PLC ở chế độ STOP
-  Program Status (Debug menu): ON/OFF trạng thái chương trình trong PLC.
-  Pause Program Status (Debug menu): Dừng ON/OFF trạng thái chương trình trong PLC.
-  Chart Status (Debug menu): ON/OFF hiển thị trạng thái dữ liệu trong bảng Status chart.
-  Trend View (View menu): ON/OFF xem trạng thái dữ liệu trong PLC ở dạng đồ thị
-  Pause Trend View: Dừng việc vẽ đồ thị dữ liệu
-  Single Read (Debug menu): Sử dụng Single Read để cập nhật một lần tất cả các giá trị trong bảng Status Chart.
-  Write All (Debug menu): Ghi tất cả các giá trị ở cột New Value trong bảng Status Chart vào PLC.
-  Force (Debug menu): Cưỡng bức dữ liệu PLC
-  Unforce For (Debug menu): Gỡ bỏ cưỡng bức dữ liệu PLC

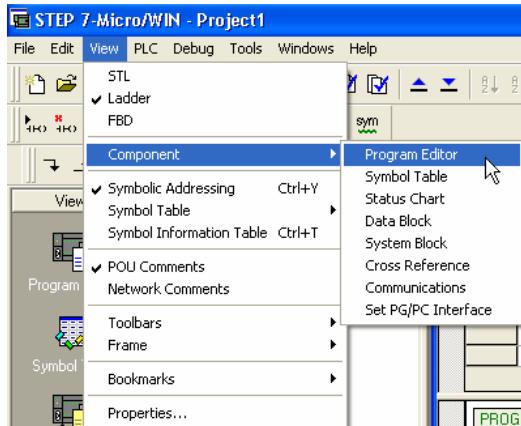


Unforce All (Debug menu): Gỡ bỏ tất cả các cường bức trong bảng Status Chart.



Read All Forced (Debug menu): Đọc tất cả các giá trị cường bức trong Status Chart.

### 6.4.3 Tạo một dự án STEP 7-Micro/WIN



Hình 6.5: Đường dẫn vào màn hình soạn thảo chương trình.

chương trình (hình 6.5).

Cũng trong menu View, ta có thể chọn ngôn ngữ lập trình là STL, Ladder hay FBD theo mong muốn.

#### 6.4.3.1 Tạo dự án mới

Để tạo một dự án mới trong STEP 7-Micro/Win, chọn menu **File > New** hoặc biểu tượng trong toolbar để mở hộp thoại "New" cho phép tạo mới một dự án (project).

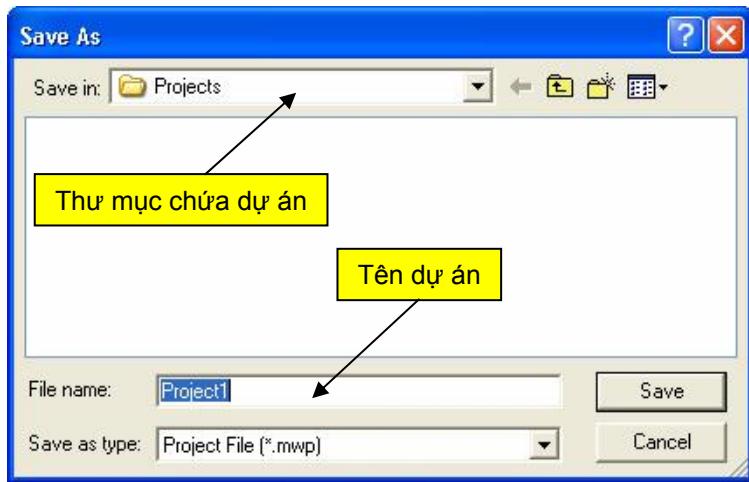
Trong thanh chức năng, bấm vào biểu tượng hoặc vào menu **View > Component > Program Editor** để mở màn hình soạn thảo

Để soạn thảo bảng ký hiệu cho các địa chỉ ta bấm vào biểu tượng trong thanh chức năng, hoặc vào menu **View > Component > symbol Table**. Sau đó có thể đặt ký hiệu cho các địa chỉ như trình bày ở mục 6.4.1.3. Phần chi tiết sẽ được trình bày trong chương phép toán nhị phân.

#### 6.4.3.2 Lưu dự án

Để lưu dự án, nhấp chuột vào biểu tượng , hoặc vào menu **File > Save**. Cửa sổ màn hình xuất hiện như hình 6.6. Chọn thư mục cần chứa dự án, đặt tên dự án và nhấp chuột vào thẻ **Save** để lưu dự án

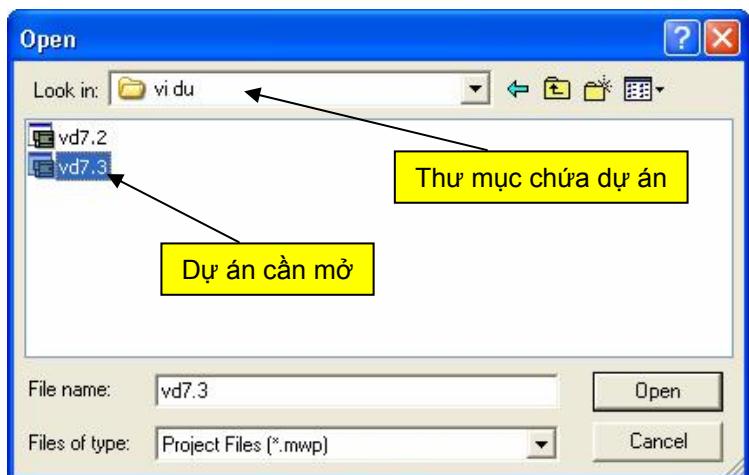




Hình 6.6: Cửa sổ màn hình lưu dự án

#### 6.4.3.3 Mở một dự án

Để mở một dự án đang có sẵn, nhấp chuột vào biểu tượng , hoặc vào menu **File > Open**. Cửa sổ màn hình xuất hiện như hình 6.7. Chọn thư mục chứa chương trình cần mở, chọn tên dự án và sau đó nhấp chuột vào thẻ **Open**.



Hình 6.7: Cửa sổ màn hình chứa dự án cần mở

#### 6.4.4 Thư viện

Thư viện (Libraries) được sử dụng để lưu trữ các khối chương trình con có truyền tham số được sử dụng để lập trình. Các khối có thể copy vào trong

một thư viện từ một dự án có sẵn hoặc chúng có thể được tạo ra trực tiếp trong thư viện độc lập với các dự án.

Khi cài đặt STEP 7-Micro/WIN thì các khối chưa được cài đặt vào trong thư viện. Để cài đặt thư viện chuẩn có thể download thư viện S7-200 từ trang [www.siemens.com](http://www.siemens.com) hoặc sử dụng đĩa phần mềm **STEP 7--Micro/WIN Add-on: STEP 7--Micro/WIN 32 Instruction Library, V1.1 (CD-ROM)**.

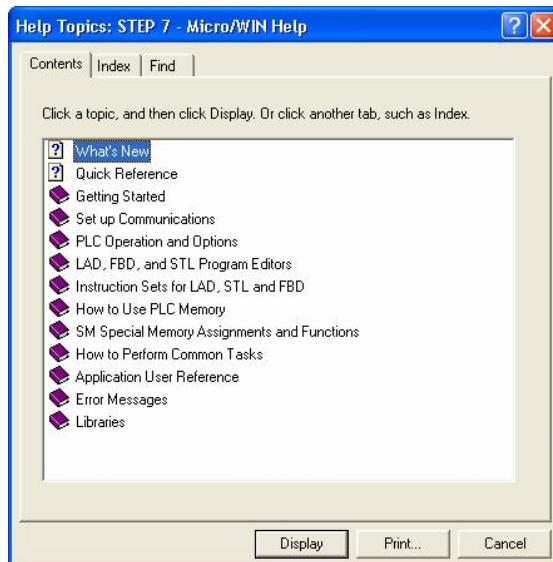
Có thể chèn thêm hoặc xóa bỏ bớt các khối chương trình trong thư viện sử dụng **File > Add/Remove Libraries** và sau đó chọn thẻ Add để chọn khối chương trình thư viện mong muốn đưa vào thư viện.

Để mở thư viện, vào **Cây Lệnh** chọn mục **Libraries**, chọn các khối chương trình cần sử dụng. Việc tạo thêm các khối chương trình con truyền tham số được sử dụng để làm thư viện có thể được tạo ra từ **File > Create Library** và chọn chương trình con cần làm thư viện.

#### 6.4.5 Hệ thống trợ giúp trong STEP 7-Micro/WIN

Trường hợp gặp khó khăn trong lập trình cũng như cần tìm hiểu rõ hơn về một thông tin nào đó trong phần mềm ta có thể sử dụng công cụ trợ giúp. Có nhiều cách khác nhau để mở trợ giúp:

- Sử dụng menu **Help > Contents and Index** để kích hoạt trợ giúp chung.
- Sử dụng phím F1 để trợ giúp theo ngữ cảnh với đối tượng được chọn.



Hình 6.8: Màn hình trợ giúp

- Thẻ Content:** Hiển thị danh sách các chủ đề trợ giúp

- **Thẻ Index:** Cho phép truy cập thông tin trợ giúp bằng việc hiển thị danh sách các thuật ngữ theo thứ tự alphabe.
- **Thẻ Find:** Cho phép tìm kiếm các từ cụ thể và thuật ngữ trong chủ đề trợ giúp.

Khi nhấp chuột vào các từ được nổi lên có màu xanh và gạch chân (hotwords) sẽ xuất hiện các trợ giúp chi tiết hơn.

#### 6.4.6 Xóa bộ nhớ CPU

Khi xóa PLC thì PLC phải đặt ở chế độ STOP và reset PLC theo chuẩn nhà máy, ngoại trừ địa chỉ PLC, tốc độ truyền, và đồng hồ thời gian (time-of-date clock). Để xóa chương trình trong PLC thực hiện như sau:

1. Chọn **PLC > Clear...** thì hộp thoại Clear xuất hiện
2. Chọn tất cả các mục chấp nhận bằng cách nhấp OK.
3. Nếu đã có password trong bộ nhớ PLC thì hộp thoại yêu cầu password xuất hiện. Để xóa password thì nhập **CLEARPLC** vào hộp thoại và tiếp tục hoạt động xóa tất cả.

#### 6.4.7 Mở một dự án đang tồn tại sẵn

Mở một dự án tồn tại (tập tin có phần mở rộng .mwp) hay thành phần của dự án và bắt đầu một phần soạn thảo mới bằng cách sử dụng các phương pháp sau:

1. Nhấp chuột vào biểu tượng Open Project .
2. Chọn menu lệnh **File > Open**.
3. Ấn tổ hợp phím Ctrl+O
4. Mở Windows Explorer và nhấp đúp chuột và tập tin có phần mở rộng .mwp.
5. Mở một thành phần dự án bằng cách nhấp chuột phải vào các ghi chú trong cây lệnh (Instruction Tree). Chọn Open để mở.

Để mở các dự án được tạo với các phiên bản trước của STEP 7-Micro/WIN hay STEP 7-Micro/DOS thì nhấp chuột vào Open  hay chọn File>Open và chọn tập tin mong muốn.

#### **Chú ý:**

- Dự án đã tạo bằng các phiên bản trước của STEP 7-Micro/WIN hay STEP 7-Micro/DOS có thể chứa một hay nhiều cấu trúc logic mà STEP 7-Micro/WIN, Version 3.0 và cao hơn không hỗ trợ. Để mở được dự án, ta phải sử dụng phiên bản cũ đã tạo dự án và lưu lại dự án theo thủ tục sau:

1. Chuyển màn hình soạn thảo sang STL.

2. Tắt địa chỉ theo ký hiệu.

3. Lưu tập tin dự án.

- Chương trình đã tạo với STEP 7-Micro/WIN V3.1 SP1 sử dụng lệnh AND có ngõ vào đơn ở FBD, và được lưu để xem ở FBD, thì không thể mở được với STEP 7-Micro/WIN V3.1. Để mở các dự án này với STEP 7-Micro/WIN V3.1, dự án trước tiên nên được chuyển sang để xem ở STL và lưu lại ở dạng này.

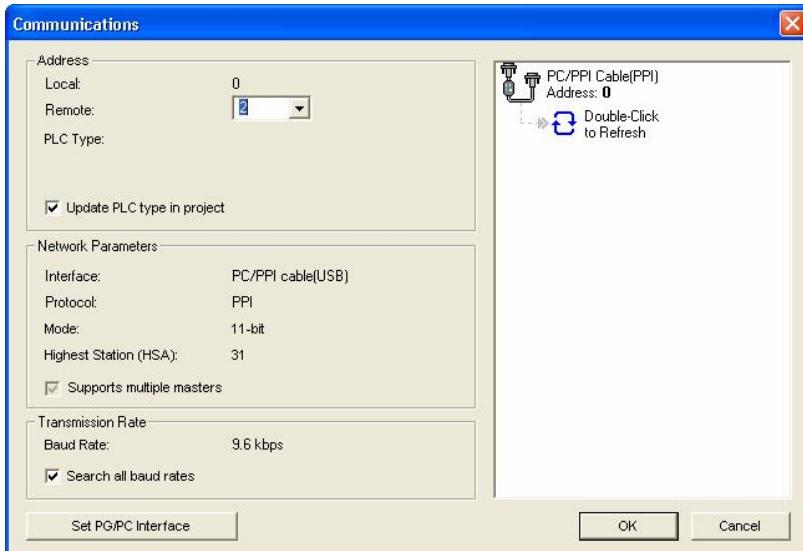
- Không thể sử dụng lệnh Open để mở một dự án trong PLC; Các tập tin dự án chỉ có thể mở được nếu nó được lưu trữ trên PC hoặc PG (thiết bị lập trình)

- Với phần mềm STEP-7 Micro/WIN mỗi lần mở chỉ được một dự án. Vì vậy muốn mở 2 dự án tại cùng một thời điểm thì phải chạy hai lần STEP-7 Micro/WIN. Khi mở hai dự án, ta có thể copy các phần tử chương trình lẫn nhau.

#### 6.4.8 Kết nối truyền thông S7-200 với thiết bị lập trình

Để kết nối truyền thông S7-200 với thiết bị lập trình thì cần phải có cáp kết nối (xem chương 4). Việc kết nối truyền thông thực hiện theo các bước sau:

- Nhấp chuột vào biểu tượng communication  trong thanh chức năng hay vào **View > Component > Communications**.



Hình 6.9: Màn hình thiết lập truyền thông

- Kiểm tra xem địa chỉ của cáp PC/PPI trong hộp thoại có được đặt là 0 chưa? Thường mặc định là 0.

3. Kiểm tra tham số mạng (Network Parameters) và tốc độ truyền (Transmission Rate) có đúng chưa. Nếu chưa đúng thì nhấp chuột vào thẻ **Set PG/PC Interface** để thiết lập lại giao tiếp giữa PC và PLC.
4. Nhấp đúp chuột vào biểu tượng to Refresh để tìm trạm S7-200 và một biểu tượng CPU cho trạm S7-200 được kết nối sẽ được hiển thị (ví dụ biểu tượng **CPU 224 REL100B3** **Address: 2**).
5. Chọn S7-200 và nhấp OK. Nếu STEP 7--Micro/WIN không tìm ra CPU S7-200, kiểm tra việc đặt chỉnh các tham số truyền thông và lặp lại bước này.
6. Sau khi đã thiết lập truyền thông với S7-200, ta có thể sẵn sàng tạo và download chương trình vào CPU.

#### 6.4.9 Tải dự án từ PLC

Có thể sử dụng biểu tượng trên toolbar hoặc menu File để tải (upload) chương trình từ PLC về máy tính khi sử dụng phần mềm STEP 7-Micro/WIN. Cần lưu ý là PLC đã được kết nối truyền thông với thiết bị lập trình.

##### 6.4.9.1 Tải một khối hoặc ba khối

Có thể tải khối chương trình (OB1, chương trình con, chương trình ngắn), System Block, và Data Block hay chọn lựa một trong ba khối này từ PLC về máy tính. Chương trình trong PLC không chứa các địa chỉ ký hiệu hay thông tin status chart. Do đó, ta không thể tải một bảng Symbol Table hay Status Chart.

##### 6.4.9.2 Tải vào một dự án mới hoặc dự án rỗng

Để tải chương trình về máy tính thì một cách không làm ảnh hưởng đến các chương trình đang mở là đóng nó lại và tạo một dự án mới, vì dự án mới là rỗng nên không thể vô tình phá hủy dữ liệu. Đây là cách thức an toàn để lấy khối chương trình, system block hoặc thông tin data block. Nếu muốn lấy sử dụng bảng ký hiệu (symbol table) hoặc status chart đã được tạo cho dự án này, thì có thể mở dự án cũ ở màn hình STEP 7-Micro/WIN khác và copy các thông tin này vào dự án được upload về.

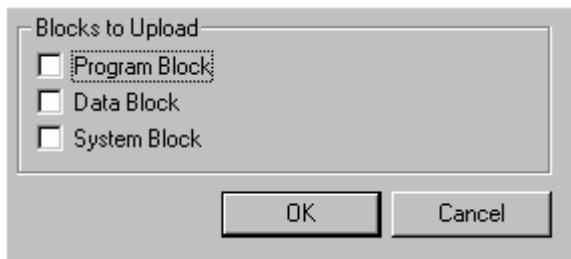
##### 6.4.9.3 Tải vào một dự án tồn tại

Đây là một cách để viết đè tất cả các phần của chương trình hiện hành bằng chương trình đã được nạp vào PLC trước đó.

##### 6.4.9.4 Thủ tục tải dự án từ PLC về thiết bị lập trình

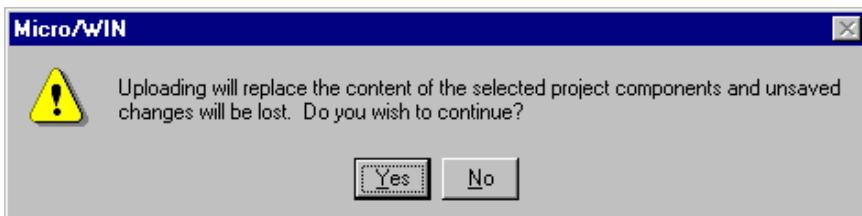
Để thực hiện tải, thực hiện các bước sau:

- Trong STEP 7-Micro/WIN mở một dự án để giữ các khối sẽ được upload từ PLC.
  - Nếu muốn upload vào một dự án rỗng, chọn **File > New** hoặc sử dụng biểu tượng New Project  trên toolbar.
  - Nếu muốn upload vào một dự án tồn tại, chọn **File > Open** hoặc sử dụng biểu tượng Open Project  trên toolbar.
- Chọn **File > Upload** hoặc sử dụng biểu tượng Upload  trên toolbar để khởi động quá trình upload.
- Hộp thoại Upload xuất hiện để yêu cầu chọn các khối: program block, data block, and system block. Hãy chọn các khối muốn Upload, và sau đó nhập OK.



Hình 6.10: Hộp thoại Upload

- STEP 7-Micro/WIN hiển thị chú ý sau:



Hình 6.11: Chú ý khi upload từ PLC về thiết bị lập trình

Nhấn Yes để chấp nhận việc upload.

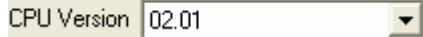
STEP 7-Micro/WIN hiển thị một thông báo khi upload các khối thành công từ PLC về thiết bị lập trình hoặc máy tính PC.

#### 6.4.10 Nạp (download) một dự án vào PLC

Khi cho phép kết nối truyền thông giữa PC và PLC, ta có thể download chương trình vào PLC. Cần lưu ý rằng khi download một program block, data block hay system block vào PLC thì nội dung của các khối được download vào sẽ viết đè lên các khối hiện hành trong PLC. Các bước thực hiện như sau:

- Trước khi download vào PLC, cần phải kiểm tra xem PLC đã ở chế độ Stop chưa thông qua đèn báo STOP trên PLC. Nếu công tắc chọn chế độ trên PLC đặt ở vị trí TERM thì ta có thể chọn PLC ở chế độ RUN hoặc STOP từ máy lập trình. Nếu PLC không ở chế độ STOP, thì nhấp chuột vào biểu tượng STOP  trong toolbar hoặc chọn **PLC > STOP**.

Trong trường hợp không dùng phần mềm thì chuyển công tắc chọn chế độ cho PLC về vị trí STOP.

- Nhấp chuột vào biểu tượng download  trong toolbar hoặc chọn **File > Download**. Hộp Download xuất hiện.
- Chọn các khối cần download. Thông thường là chọn tất cả.
- Nhấp OK để bắt đầu quá trình download.
- Nếu download thành công, thì một hộp thoại hiển thị thông báo: *Download Successful*. Tiếp tục đến bước 12.
- Nếu loại PLC được chọn cho chương trình trong STEP 7/Micro/WIN không phù hợp với PLC thực tế, thì một hộp thoại xuất hiện với thông báo:  
*"The PLC type selected for the project does not match the remote PLC type. Continue Download?"*.
  - Đặt lại loại PLC cho phù hợp, chọn **No** để dừng tiến trình download.
  - Chọn **PLC > Type...** để vào hộp thoại chọn loại PLC.
  - Có thể chọn đúng loại PLC theo danh sách trong mục **PLC Type**  **CPU 224**  **CPU Version** **02.01** của hộp thoại. Hoặc nhấp chuột vào thẻ **Read PLC** để STEP 7-Micro/WIN tự động tìm đúng loại PLC đang kết nối.
  - Nhấp **OK** để chấp nhận loại PLC và đóng hộp thoại.
  - Khởi động lại quá trình download bằng cách nhấp chuột vào biểu tượng download  trong toolbar hay chọn **File > Download**.
  - Ngay khi download thành công, ta phải chuyển PLC từ STOP sang RUN trước khi PLC có thể thực hiện chương trình. Nhấp chuột vào biểu tượng RUN  trong toolbar hay chọn **PLC > RUN** để chuyển

PLC sang chế độ RUN khi công tắc chọn chế độ cho PLC để ở vị trí TERM.

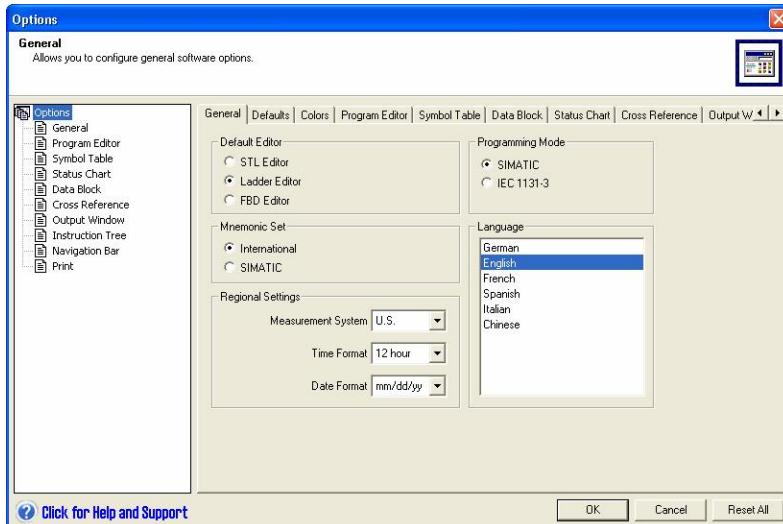
Trường hợp sử dụng công tắc thì chuyển từ vị trí STOP sang RUN.

### 6.4.11 Thiết lập cấu hình chung cho phần mềm (menu option và customize)

#### 6.4.11.1 Menu Option

Có thể định nghĩa một đường dẫn mặc định đến một thư mục tập tin xác định để mở và lưu các dự án STEP 7-Micro/WIN. Ta sử dụng menu lệnh Tools > Options.

Ngoài ra, để truy cập trực tiếp Option cho từng thành phần trong cây lệnh (Instruction tree) thì trỏ chuột vào thành phần mong muốn và nhấp chuột phải, sau đó chọn mục **option**.



Hình 6.12: Cửa sổ Options

#### \* General Options

- **Thẻ General:** Chọn thẻ này để lựa chọn Program Editor, Mnemonic Set, Programming Mode, Language, và Regional Settings(Measurement System, Time Format, and Date Format) mặc định.

- **Thẻ Defaults:** Chọn thẻ này để đặt vị trí tập tin và loại PLC mặc định cho các dự án mới. Ta cũng có thể chọn để thêm System Symbol Table cho tất cả các dự án mới.

- **Thẻ Colors:** Chọn thẻ này để gán Font và Color cho các cửa sổ khác nhau.

#### \* Program Editor Options

- *Thẻ Program Editor:* Chọn thẻ này để định kích thước, hiển thị và font của cửa sổ soạn thảo chương trình. Chọn trạng thái hiển thị bên trong hay bên ngoài lệnh. Cấu hình địa chỉ theo ký hiệu. Ta cũng có thể chọn để cho phép soạn thảo toán tử sau khi đặt một lệnh và định dạng tự động bất kỳ mã lệnh STL được nhập vào.

- *Thẻ STL Status:* Chọn thẻ này để tùy biến cách thức mà Program Status được trình diễn ở STL. Ta có thể thay đổi các đặt chỉnh sau: Watch Values, Operands, Logic Stack, Instruction Status Bits.

#### \* Other Options

- *Thẻ Symbol Table:* Chọn thẻ này để thiết lập kiểu font, kiểu dáng và kích thước của bảng ký hiệu (symbol table). Ta có thể chọn để hiển thị các ký hiệu trùng nhau, không được sử dụng.

- *Thẻ Status Chart:* Chọn thẻ này để thiết lập kiểu font, kiểu dáng và kích thước của status chart. Cũng có thể thiết lập việc định địa chỉ theo ký hiệu.

- *Thẻ Data Block:* Chọn thẻ này để thiết lập kiểu font, kiểu dáng và kích thước và độ rộng của data block.

- *Thẻ Cross Reference:* Chọn thẻ này để thiết lập kiểu font, kiểu dáng và kích thước của bảng cross reference. Cũng có thể thiết lập việc định địa chỉ theo ký hiệu.

- *Thẻ Output Window:* Chọn thẻ này để thiết lập kiểu font, kiểu dáng và kích thước của output window.

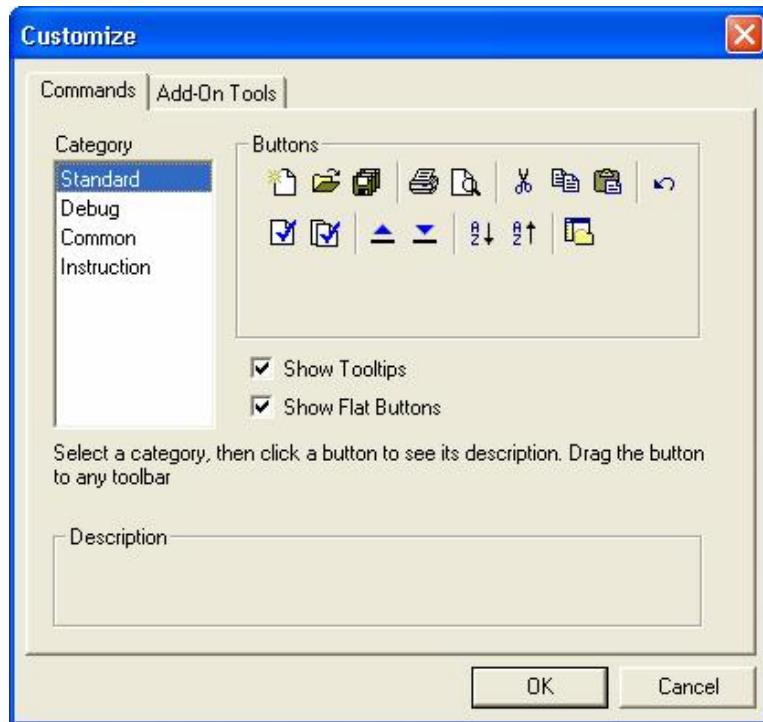
- *Thẻ Instruction Tree:* Chọn thẻ này để thiết lập kiểu font, kiểu dáng và kích thước của Instruction Tree (cây lệnh). Ta cũng có thể chọn để cho phép tự động xếp lại của instruction tree.

- *Thẻ Navigation Bar:* Chọn thẻ này để thiết lập kiểu font, kiểu dáng và kích thước của navigation bar.

- *Thẻ Print:* Chọn thẻ này để thiết lập kiểu font, kiểu dáng và kích thước của các dự án muốn in.

#### 6.4.11.2 Menu Custommize

Menu custommize cho phép ta thay đổi sự xuất hiện nội dung trong toolbar và thêm vào các công cụ được sử dụng thường xuyên vào menu Tools.



Hình 6.13: Cửa sổ customize.

Chọn menu lệnh **Tools > Customize** để thiết lập các lựa chọn sau:

- **Thẻ Commands:** Cho phép thay đổi sự xuất hiện các nội dung của toolbars.
- **Thẻ Add-On Tools:** Cho phép thêm vào các công cụ được sử dụng thường xuyên vào menu Tools.

#### \* **Thay đổi sự xuất hiện:**

- Chọn **Show Tooltips** nếu muốn các nút nhấp hiển thị các thông tin về nó khi con trỏ chuột dừng trên nút nhấp.
- Chọn **Show Flat Buttons** nếu muốn các nút nhấp xuất hiện ở dạng phẳng thay vì xuất hiện ở dạng 3-D.

#### \* **Di chuyển một nút nhấp:**

- Chọn một toolbar từ hộp danh sách Category để hiển thị các nút nhấp của toolbar đó. Để di chuyển một nút nhấp từ toolbar mặc định sang toolbar khác, thì chọn tên của toolbar chứa nút nhấp cần di chuyển từ hộp danh sách Category. Kéo nút nút nhấp mong muốn trong vùng nút nhấp ra vùng toolbar để thêm nó vào toolbar.

- Để loại bỏ một nút nhấp trên toolbar, kéo nút nhấp trên toolbar và bỏ vào vùng nút nhấp của hộp thoại Customize.

\* *Thẻ Add-On Tools:* Thêm một công cụ vào menu Tools.

Đặc điểm này được dự định để tiết kiệm thời gian đối với các công cụ được sử dụng thường xuyên. Để thêm một công cụ, nhấp vào thẻ Add-On Tools, nhấp vào nút  Add, và điền vào các vùng ở dưới:

Bất kỳ lệnh được yêu cầu được bắt đầu và kết thúc bởi dấu ngoặc kép khi nhập vào vùng command(ví dụ: "xxx xxx").

- *Menu Text:* Chọn một tên để nhận dạng công cụ trên menu Tools.

- *Command:* Cung cấp tên tập tin của chương trình công cụ hay bat. file.

- *Arguments:* Cung cấp các chủ đề dòng lệnh đã sử dụng bởi tập tin \*.exe.

- *Initial Directory:* Cung cấp đường dẫn thư mục đang mở cho công cụ.

Sử dụng nút  ... để tìm các tập tin và thư mục.

Khi thêm vào một công cụ thành công, trong menu Tools xuất hiện công cụ đã thêm.

#### 6.4.12 Soạn thảo chương trình

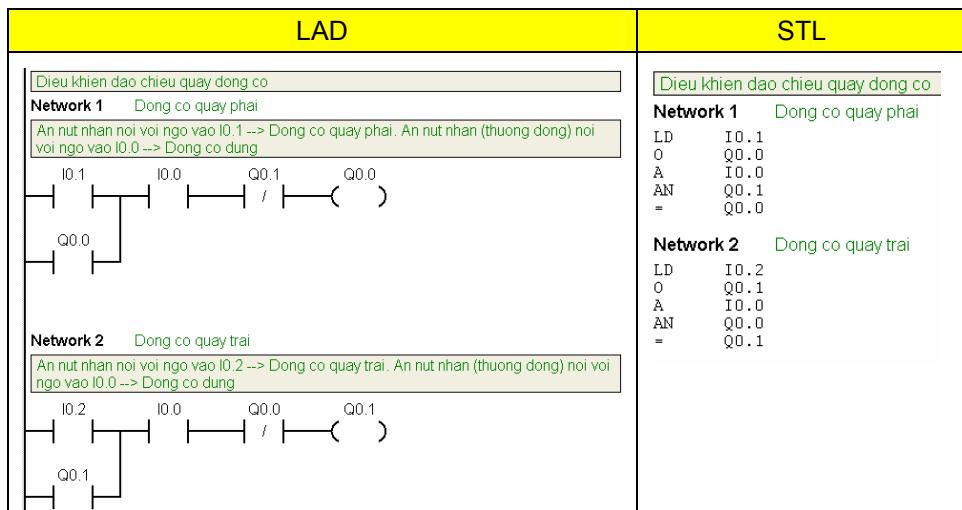
Trước khi soạn thảo chương trình, các bước sau đây cần phải hoàn thành:

- Kết nối giữa PLC và máy tính
- Kết nối dây đúng các ngõ vào và ra với ngoại vi

Trường hợp không có PLC, thì ta chỉ có thể soạn thảo chương trình và lưu trữ lại. Còn nếu muốn kiểm tra thì cần phải có phần mềm mô phỏng S7-200. Các bước để soạn thảo một dự án mới:

1. Mở màn hình soạn thảo chương trình
2. Nhập bảng ký hiệu
3. Nhập chương trình
4. Lưu chương trình
5. Download chương trình vào CPU.
6. Đặt CPU ở chế độ RUN.
7. Tìm lỗi và chỉnh sửa chương trình.

Để hiểu được phần mềm STEP 7-Micro/WIN dễ dàng, chúng ta nên viết một ví dụ đơn giản được cho ở hình 6.14 và bảng thiết lập vào/ra cho ở bảng 6.1. Do mới bắt đầu, ta nên viết chương trình ở dạng LAD, rồi sau đó có thể xem ở dạng FBD hay STL.



Hình 6.14: Ví dụ để soạn thảo một chương trình mới

Ký hiệu	Địa chỉ	Chú thích
S_Stop	I0.0	Nút nhấn dừng động cơ, thường đóng (NC)
S_Right	I0.1	Nút nhấn động cơ quay phải, thường hở (NO)
S_Left	I0.2	Nút nhấn động cơ quay trái, thường hở (NO)
K1	Q0.0	Contactor cấp điện để động cơ quay phải
K2	Q0.1	Contactor cấp điện để động cơ quay trái

Bảng 6.1: Bảng xác định kết nối dây vào/ra với ngoại vi

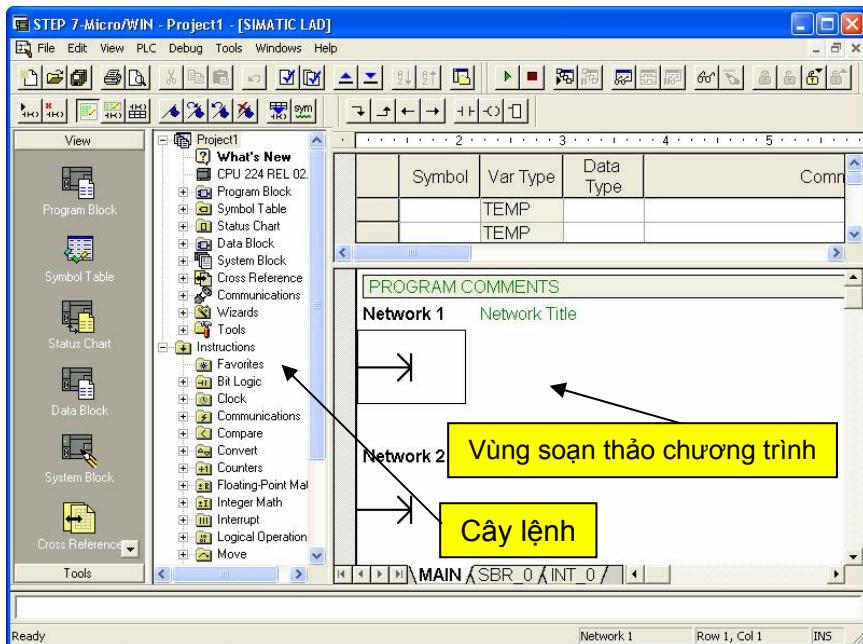
### Các bước thực hiện:

#### Bước 1: Mở màn hình soạn thảo chương trình



Nhấp chuột vào biểu tượng Program Block để mở màn hình soạn thảo chương trình (hình 6.15). Chú ý cửa sổ cây lệnh (instruction tree) và vùng soạn thảo chương trình. Sử dụng cây lệnh để chèn các lệnh được biểu diễn ở dạng LAD vào các networks của màn hình soạn thảo chương trình bằng cách kéo và thả các lệnh từ cây lệnh vào các networks.

Để có thể nhập đầy đủ các chú thích (comment), thì cần hiển thị các chú thích trong màn hình soạn thảo chương trình. Vào **View > POU Comment** để hiển thị dòng chú thích tiêu đề chương trình và **View > Network comments** để hiển thị dòng chú thích của từng network.



Hình 6.16: Màn hình soạn thảo chương trình

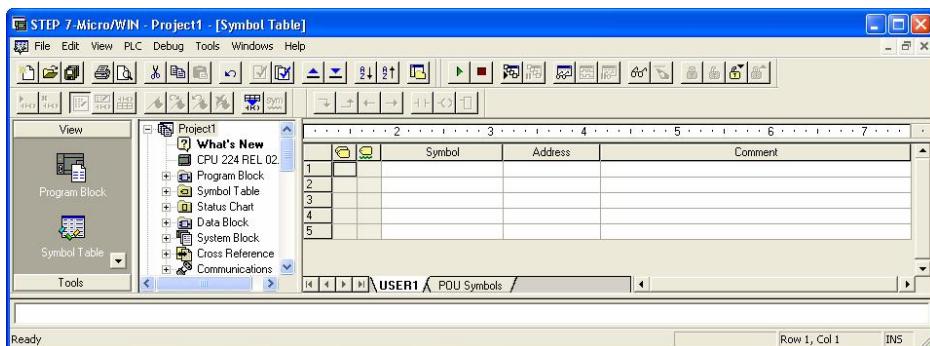
## Bước 2: Nhập bảng ký hiệu



Nhấp chuột vào biểu tượng Symbol Table để mở màn hình soạn thảo bảng ký hiệu (hình 6.17).

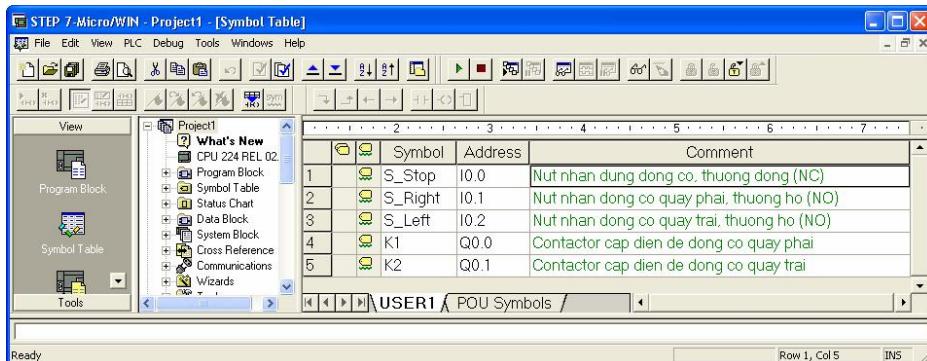
Nhập các thông tin (chữ không dấu) ở bảng 6.1 vào bảng Symbol Table. Với:

- Cột **ký hiệu** tương ứng với cột *Symbol*.
- Cột **địa chỉ** tương ứng với cột *Address*.
- Cột **chú thích** tương ứng với cột *comment*.



Hình 6.17: Màn hình soạn thảo bảng ký hiệu

Sau khi nhập xong, ta có bảng ký hiệu như hình 6.18.



Hình 6.18: Bảng ký hiệu các phần tử trong chương trình

Trong quá trình lập trình có thể phát sinh thêm các địa chỉ mới. Khi phát sinh thêm địa chỉ mới, ta nên bổ sung địa chỉ đó vào trong bảng ký hiệu để dễ dàng cho quá trình tìm và xử lý lỗi sau này.

### Bước 3: Nhập chương trình



Nhấp chuột vào biểu tượng Program Block để mở lại màn hình soạn thảo chương trình (hình 6.15).

- Nhập Network 1: Dong co quay phai**

Khi ấn nút nhấn S\_Right (I0.1), thì tiếp điểm I0.1 đóng, nút nhấn S\_Stop là thường đóng nên ngõ vào I0.0 luôn luôn có điện hay tiếp điểm I0.0 cũng đóng, và bình thường ngõ ra Q0.1 cũng không có điện (0) nên tiếp điểm này cũng đóng. Kết hợp 3 tiếp điểm này sẽ có dòng điện cung cấp cho cuộn dây Q0.0 (nối với K1). Contactor K1 có điện đóng tiếp điểm động lực của nó để cấp nguồn cho động cơ quay phải. Tiếp điểm Q0.0 (song song I0.1) đóng duy trì dòng cung cấp cho Q0.0 khi nút nhấn S\_Right hở ra.

Nhập các dòng chú thích như đã cho trong hình 6.14.

Nhập các tiếp điểm như sau:

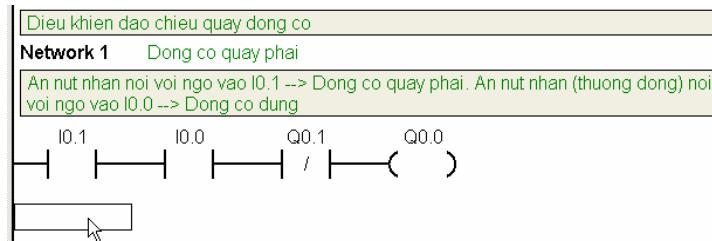
- Nhấp đúp chuột vào hình tượng Bit Logic hoặc nhấp chuột vào dấu cộng (+) ở cửa sổ cây lệnh để hiển thị các lệnh trong bit logic.
- Chọn tiếp điểm Normally Open .
- Giữ chuột trái và kéo tiếp điểm vào network đầu tiên.
- Nhấp chuột vào “???” trên tiếp điểm và nhập vào địa chỉ: I0.1 và sau đó nhấn phím Enter.
- Tương tự từ bước 2 đến bước 4 nhập địa chỉ I0.0
- Chọn tiếp điểm Normally Closed . và sau đó nhập vào địa chỉ Q0.1

7. Chọn cuộn dây Output và nhập vào ở “???” địa chỉ Q0.0

**Chú ý:** khi gõ các địa chỉ I0.0, I0.1, Q0.0, Q0.1 có thể ta sẽ nhận được kết quả là các địa chỉ theo ký hiệu. Để hiện lại các địa chỉ tuyệt đối ta bỏ kích hoạt **View > Symbolic Addressing**.

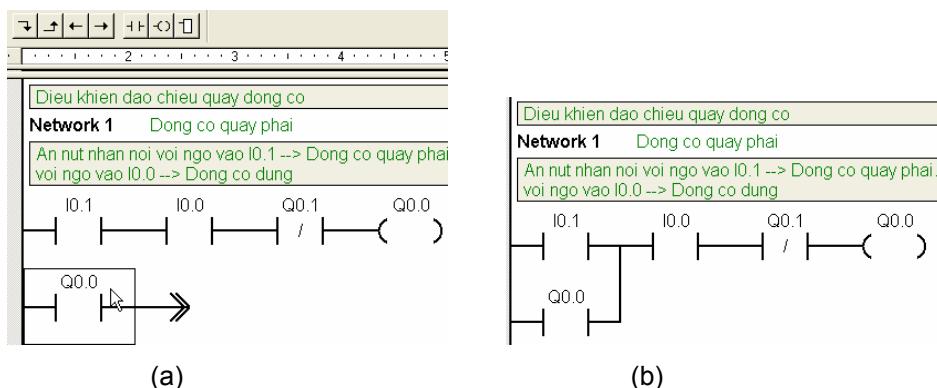
### Rẽ nhánh Network 1.

1. Tương chọn tiếp điểm Normally Open giữ chuột trái và kéo tiếp điểm vào vị trí con trỏ chuột (hình 6.18) và đặt tên Q0.0.



Hình 6.18: Rẽ nhánh network

2. Để con trỏ chuột ở vị trí như hình 6.19a và nhấp chuột vào biểu tượng để kết thúc (hình 6.19b).



Hình 6.19: Rẽ nhánh network

- Nhập network 2: Dong co quay trai
- Tương tự như network 1.

### Bước 4: Lưu chương trình

Sau khi nhập hai network lệnh, ta đã nhập xong chương trình. Khi lưu chương trình, ta tạo một dự án bao gồm loại CPU S7-200 và các tham số khác. Để lưu một dự án, thực hiện như sau:

1. Chọn **File > Save As**
2. Nhập vào tên của dự án trong hộp thoại Save As

3. Nhấp OK để lưu dự án.

#### **Bước 5: Download chương trình vào CPU**

Sau khi lưu dự án, ta có thể download chương trình vào S7-200.

Mỗi dự án được liên kết với một loại CPU (CPU 221, CPU 222, CPU 224, CPU 224XP, hoặc CPU 226). Nếu kiểu dự án không phù hợp với CPU đang kết nối, thì STEP 7--Micro/WIN báo lỗi không tương thích và các đường dẫn để ta tiếp tục công việc. Nếu điều này xảy ra, chọn “Continue Download”.

Thực hiện download chương trình như sau:

1. Nhấp chuột vào biểu tượng Download  trên toolbar hoặc chọn **File > Download** để download chương trình.
2. Nhấp OK để download các phần tử chương trình vào S7-200. Nếu S7-200 ở chế độ RUN, một hộp thoại xuất hiện yêu cầu bạn đặt S7-200 ở chế độ STOP. Nhấp chuột vào Yes để đặt S7-200 ở chế độ STOP.

#### **Bước 6: Đặt S7-200 ở chế độ RUN**

Đối với phần mềm STEP 7-Micro/WIN để đặt CPU S7-200 vào chế độ RUN, thì công tắc chọn chế độ của S7-200 phải được đặt ở vị trí **TERM** hoặc **RUN**. Khi đặt S7-200 ở chế độ RUN, thì S7-200 thực hiện chương trình:

1. Nhấp chuột vào biểu tượng RUN  trên toolbar hoặc chọn **PLC > RUN**.
2. Nhấp OK chuyển chế độ hoạt động của S7-200.
3. Khi S7-200 đi vào chế độ RUN thì đèn RUN trên PLC sáng.

#### **Bước 7: Tìm lỗi và chỉnh sửa chương trình**

Sau khi CPU đã ở chế độ RUN, ta có thể kiểm tra lại chương trình bằng cách ấn các nút nhấn S\_Right, S\_Stop, S\_Left và quan sát các đèn LED Q0.0 và Q0.1.

Nếu ấn nút nhấn S\_Right, thì đèn LED Q0.0 sáng.

Ấn nút S\_Stop, thì đèn LED Q0.0 tắt.

Ấn nút S\_Left, thì đèn Q0.1 sáng.

Ấn nút S\_Stop, thì đèn LED Q0.1 tắt.

Nếu việc kiểm tra không đạt được kết quả như mô tả, thì có thể giám sát chương trình bằng cách chọn **Debug > Program Status** hoặc nhấp chuột vào

biểu tượng . Dựa vào trạng thái của các tiếp điểm và các cuộn dây trong chương trình mà có thể tìm ra các lỗi và chỉnh sửa cho phù hợp với yêu cầu công nghệ.

Để dừng chương trình, đặt S7-200 về chế độ STOP bằng cách nhấp chuột vào biểu tượng STOP  hoặc chọn **PLC > STOP**.

## 7 Các phép toán logic

### 7.1 Ngăn xếp (logic stack) trong S7-200

Trong các CPU S7-200 có một ngăn xếp gồm 9 bit, chúng được sử dụng cho các câu lệnh mà dữ liệu là dạng bit. Khi viết chương trình dạng STL thì người lập trình cần hiểu rõ về phương thức hoạt động của các bit trong ngăn xếp. Ngăn xếp logic là một khối gồm 9 bit chồng lên nhau. Tất cả các thuật toán liên quan đến ngăn xếp đều chỉ làm việc với bit đầu tiên hoặc với bit đầu và bit thứ hai của ngăn xếp. Giá trị logic mới đều có thể được gửi (hoặc được nối thêm) vào ngăn xếp. Khi phối hợp hai bit đầu tiên của ngăn xếp, thì ngăn xếp sẽ được kéo lên một bit. Ngăn xếp và tên của từng bit trong ngăn xếp được biểu diễn dưới đây:

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

Trong 9 Stack, thì Stack 0 là ngăn xếp quan trọng nhất. Giá trị logic của nó sẽ là kết quả của phép toán logic. Hay nói khác đi, sau một phép toán logic nhị phân thì kết quả của phép toán sẽ được lưu ở Stack 0. Nếu giá trị logic ở Stack 0 có giá trị là “0” thì kết quả thu được là “0”, tương tự nếu có giá trị là “1” thì kết quả thu được là “1”.

Ngoài ra giá trị logic “1” của Stack 0 còn là điều kiện bắt buộc cho việc thi hành đối với một số lệnh.

## 7.2 Các phép toán logic cơ bản

Trong phần này trình bày các phép toán đối với dữ liệu là bit. Trước tiên là phần lý thuyết sau đó tới ví dụ và chương trình. CPU sử dụng trong các ví dụ là loại DC/DC/DC (nguồn cung cấp cho ngõ vào, ra và CPU là 24Vdc).

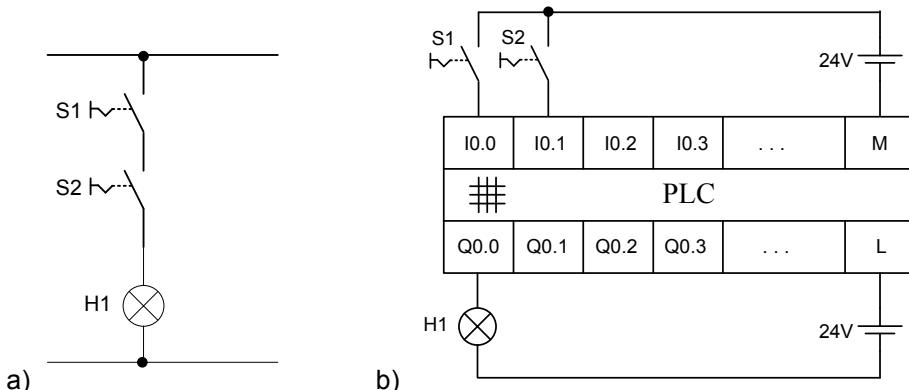
Vì phần soạn thảo chương trình đã được trình bày ở *chương 6*, nên trong phần này không trình bày lại. Bạn đọc có thể xem *mục 6.4.12 của chương 6* để thực hiện cho các ví dụ ở chương này và các chương tiếp theo.

Chương này chủ yếu trình bày về các phép toán liên quan đến bit hay còn gọi là phép toán nhị phân. Vì vậy khi viết chương trình, ta chỉ lấy các phần tử trong bit logic ( Bit Logic) của cây lệnh.

### 7.2.1 Phép toán AND

Phép toán AND được sử dụng khi có yêu cầu điều khiển là trạng thái của 2 hay nhiều tín hiệu *đồng thời* xảy ra thì sẽ thực hiện một nhiệm vụ điều khiển nào đó.

**Ví dụ 7.1:** Đèn H1 sẽ sáng nếu đồng thời cả 2 công tắc S1 và S2 ở trạng thái đóng mạch. Đèn tắt khi 1 trong 2 công tắc hở mạch.



Hình 7.1 Liên kết AND: a) Sơ đồ mạch điện, b) Nối dây với ngõ vào/ra PLC

- Lập bảng ký hiệu mô tả tên và địa chỉ của biến (soạn thảo bằng cách mở mục *Symbol Table* trong phần mềm soạn thảo):

	Symbol	Address	Comment
1		I0.0	Cong tac
2		I0.1	Cong tac
3		Q0.0	Den bao

Hình 7.2 Bảng ký hiệu

+ Chương trình:

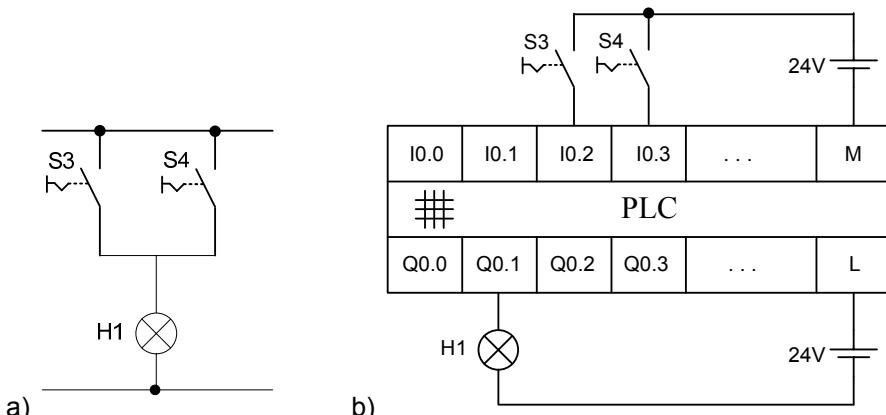
LAD	FBD	STL
		<pre> LD      I0.0 A      I0.1 =      Q0.0     </pre>

Hình 7.3 Chương trình được biểu diễn ở 3 dạng LAD, FBD và STL.

### 7.2.2 Phép toán OR

Phép toán OR sẽ được sử dụng khi trạng thái của một trong hai (hoặc nhiều) tín hiệu thỏa mãn điều kiện của yêu cầu cầu điều khiển thì sẽ thực hiện một nhiệm vụ điều khiển nào đó.

**Ví dụ 7.2:** Có 2 công tắc S3 và S4 đều là thường mở. Hãy viết chương trình sao cho nếu một trong 2 công tắc đóng lại thì đèn H2 sẽ sáng. Đèn tắt khi cả 2 công tắc đều mở.



Hình 7.4 Liên kết OR: a) Sơ đồ mạch điện, b) Nối dây với ngõ vào/ra PLC,

	Symbol	Address	Comment
1	S3	I0.2	Cong tac
2	S4	I0.3	Cong tac
3	H2	Q0.1	Den bao

LAD	FBD	STL
		<pre> LD      I0.2 O      I0.3 =      Q0.1     </pre>

Hình 7.5 Bảng ký hiệu và chương trình liên kết OR

### 7.2.3 Tổ hợp các cỗng AND và OR

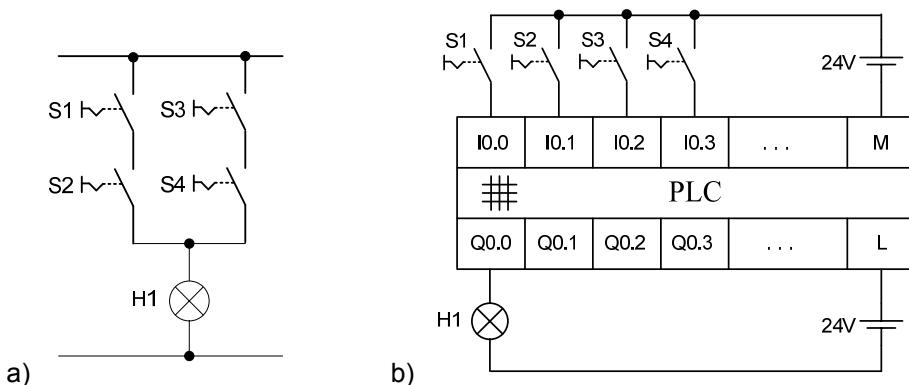
Trong thực tế, các đối tượng điều khiển phụ thuộc vào một tổ hợp các liên kết logic AND và OR. Tùy theo liên kết nào đứng trước mà sẽ có các lệnh ở STL khác nhau.

#### 7.2.3.1 AND trước OR

Để thực hiện phép OR hai liên kết AND lại với nhau thì trong chương trình viết ở dạng STL phải sử dụng thêm lệnh **OLD**.

**Ví dụ 7.3:**

	Symbol	Address	Comment
1	S1	I0.0	Cong tac
2	S2	I0.1	Cong tac
3	S3	I0.2	Cong tac
4	S4	I0.3	Cong tac
5	H1	Q0.0	Den bao



c) chương trình

LAD	FBD	STL												
		<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 33%;">LD</td> <td style="width: 33%;">I0.0</td> </tr> <tr> <td>A</td> <td>I0.1</td> </tr> <tr> <td>LD</td> <td>I0.2</td> </tr> <tr> <td>A</td> <td>I0.3</td> </tr> <tr> <td>OLD</td> <td></td> </tr> <tr> <td>=</td> <td>Q0.0</td> </tr> </table>	LD	I0.0	A	I0.1	LD	I0.2	A	I0.3	OLD		=	Q0.0
LD	I0.0													
A	I0.1													
LD	I0.2													
A	I0.3													
OLD														
=	Q0.0													

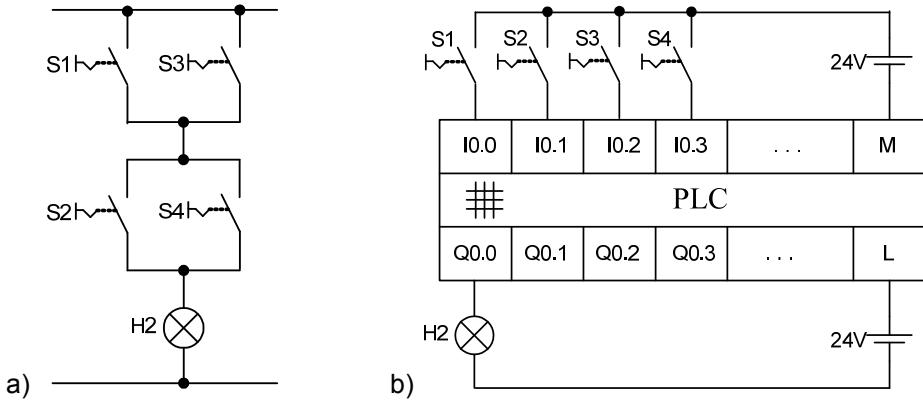
Hình 7.6 AND trước OR: a) Mạch điện, b) Nối dây với PLC, c) Chương trình

#### 7.2.3.2 OR trước AND

Để thực hiện phép AND hai liên kết OR lại với nhau thì trong chương trình viết ở dạng STL phải sử dụng thêm lệnh **ALD**.

**Ví dụ 7.4:**

	Symbol	Address	Comment
1	S1	I0.0	Cong tac
2	S2	I0.1	Cong tac
3	S3	I0.2	Cong tac
4	S4	I0.3	Cong tac
5	H2	Q0.0	Den bao



c) Chương trình

LAD	FBD	STL
		<pre> LD   I0.0 0   I0.2 LD   I0.1 0   I0.3 ALD =       Q0.0   </pre>

Hình 7.7 OR trước AND: a) Mạch điện, b) Nối dây với PLC, c) Chương trình

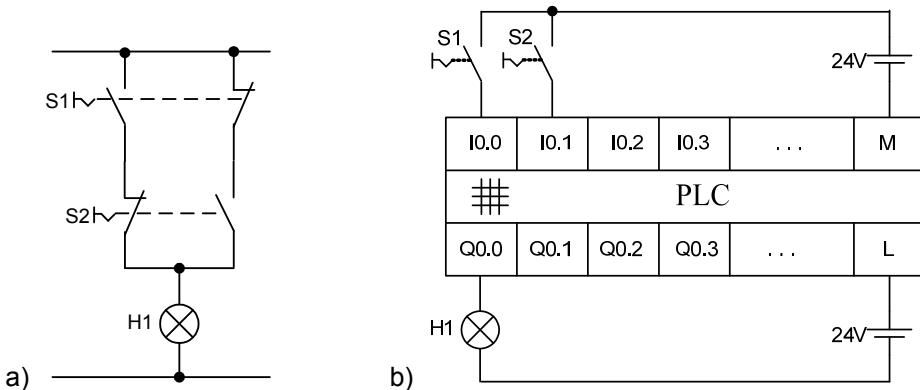
**7.2.4 Phép toán XOR**

Phép toán XOR được sử dụng khi có 2 tín hiệu mà nếu chúng có cùng trạng thái thì ngõ ra sẽ xuống mức 0 còn nếu 2 tín hiệu này khác trạng thái thì ngõ ra sẽ lên mức 1.

**Ví dụ 7.5:** Ở sơ đồ hình 7.8a, mỗi một nút nhấn được gắn 2 tiếp điểm (1NO và 1NC), khi tác động nút nhấn thì cả 2 tiếp điểm này tác động theo. Đèn sáng nếu tác động chỉ một trong hai công tắc S1 hoặc S2.

**Bảng ký hiệu**

	Symbol	Address	Comment
1	S1	I0.0	Cong tac
2	S2	I0.1	Cong tac
3	H1	Q0.0	Den bao



Hình 7.8 Liên kết XOR a) Sơ đồ mạch điện, b) Kết nối với PLC

LAD	FBD	STL
		<pre> LD   I0.0 ON  I0.0 LDN I0.1 O   I0.1 ALD =       Q0.0     </pre>

Hình 7.9 Chương trình liên kết XOR

### 7.3 Xử lý các tiếp điểm, cảm biến được nối với ngõ vào PLC

Một vấn đề quan trọng đối với người mới làm quen với chương trình PLC là việc xác định đúng trạng thái các loại tiếp điểm được viết ở LAD. Đặc biệt là các tiếp điểm ngõ vào.

Các cảm biến, công tắc hoặc nút nhấn thường có hai dạng là *thường đóng (NC)*, hoặc *thường mở (NO)*. Vì các ngõ vào số được nối với các đối tượng này nên các tiếp điểm trong chương trình, tùy theo trường hợp, cũng sẽ có dạng tương ứng. Tuy nhiên, để dễ dàng phân biệt ta không nên gọi các tiếp điểm trong chương trình là thường đóng hoặc thường mở. Qui ước đặt tên cho các tiếp điểm trong chương trình như sau:

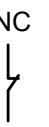
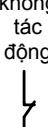
- Tiếp điểm : Được gọi là *tiếp điểm không đảo trạng thái tín hiệu*.
- Tiếp điểm : Được gọi là *tiếp điểm đảo trạng thái tín hiệu*.

Để rõ hơn trạng thái các tiếp điểm được nối với ngõ vào số và kết quả xử lý chương trình trong PLC, ta xem bảng 7.1.

Từ bảng này, ta có một số nhận xét như sau:

1. *Ngõ vào có logic “1” khi ngõ vào có điện áp.*

2. Nếu ngõ vào được nối với tiếp điểm thường đóng (NC), thì ngõ vào ở trạng thái bình thường luôn có điện (đèn LED báo ngõ vào tương ứng sáng). Nó chỉ bị mất điện nếu tiếp điểm NC bị tác động.
3. Nếu ngõ vào được nối với tiếp điểm thường mở (NO), thì ngõ vào ở trạng thái bình thường không có điện (đèn LED báo ngõ vào tương ứng tắt). Nó chỉ có điện khi tác động tiếp điểm NO.
4. Nếu sử dụng tiếp điểm không đảo trạng thái tín hiệu  , thì kết quả xử lý trong chương trình có **cùng trạng thái logic** với ngõ vào.
5. Nếu sử dụng tiếp điểm đảo trạng thái tín hiệu  , thì kết quả xử lý trong chương trình có **trạng thái logic ngược** với ngõ vào.
6. Không được thay tùy tiện tiếp điểm thường mở (NO) bằng tiếp điểm  trong chương trình, cũng như tiếp điểm thường đóng (NC) bằng tiếp điểm  . Mà phải chú ý đến yêu cầu công nghệ đặt ra.

Bộ tạo tín hiệu nhị phân			Thực hiện trong chương trình PLC				
Cảm biến, nút nhấn là một ...	Cảm biến, nút nhấn bị ...	Điện áp tại ngõ vào PLC	Trạng thái tín hiệu tại ngõ vào	Kiểm tra cho trạng thái tín hiệu "1"	Kiểm tra cho trạng thái tín hiệu "0"		
			Ký hiệu/lệnh	Kết quả kiểm tra	Ký hiệu/lệnh	Kết quả kiểm tra	
NO 	tác động 	có	1	LAD:  "tiếp điểm không đảo"	1	LAD:  "tiếp điểm đảo"	0
	không tác động 	không	0	FBD:  AND	0	FBD:  AND	1
NC 	tác động 	không	0	STL: LD Ix.y	0	STL: LDN Ix.y	1
	không tác động 	có	1		1		0

Bảng 7.1 Trạng thái các tiếp điểm và xử lý trong chương trình PLC

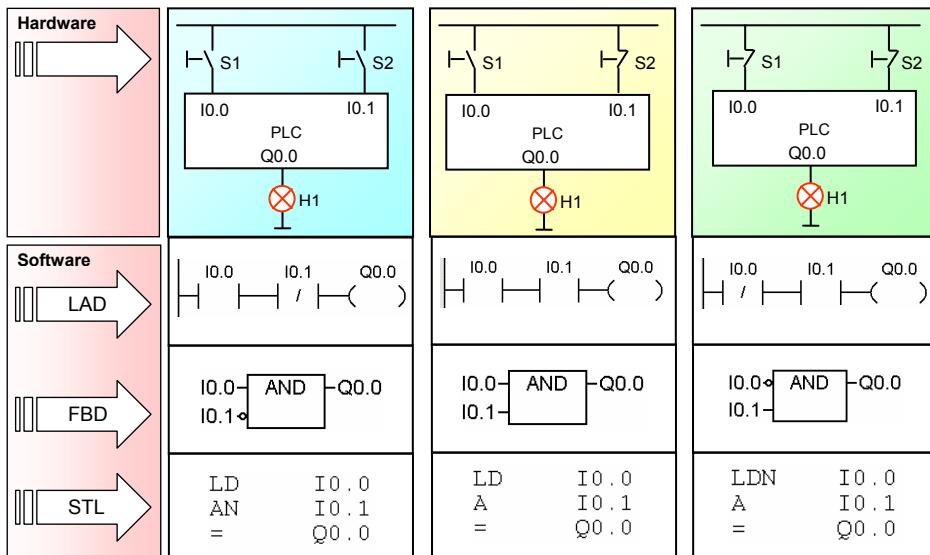
Ví dụ sau đây sẽ làm sáng tỏ hơn về việc xử lý các tiếp điểm nối với ngõ vào.

**Ví dụ 7.6:** Trong 3 mạch dưới đây (hình 7.10), đèn H1 sẽ sáng khi ấn nút nhấn S1 và không ấn nút nhấn S2.

Từ ví dụ ta nhận thấy dù ngõ vào được nối với loại nút nhấn nào cũng vẫn có thể lập chương trình để thỏa mãn được yêu cầu đặt ra. Tuy nhiên việc sử dụng các tiếp điểm thường mở hoặc thường đóng trong quá trình điều khiển phụ thuộc vào các qui tắc an toàn.

Các tiếp điểm thường đóng luôn luôn được sử dụng cho công tắc hành trình và công tắc an toàn, để khống chế sự nguy hiểm nếu dây điện bị đứt trong mạch điện cảm biến.

Các tiếp điểm thường đóng cũng được dùng để tắt máy vì lý do tương tự như trên.



Hình 7.10: Ví dụ xử lý các loại tiếp điểm.

## 7.4 Ví dụ ứng dụng các liên kết logic

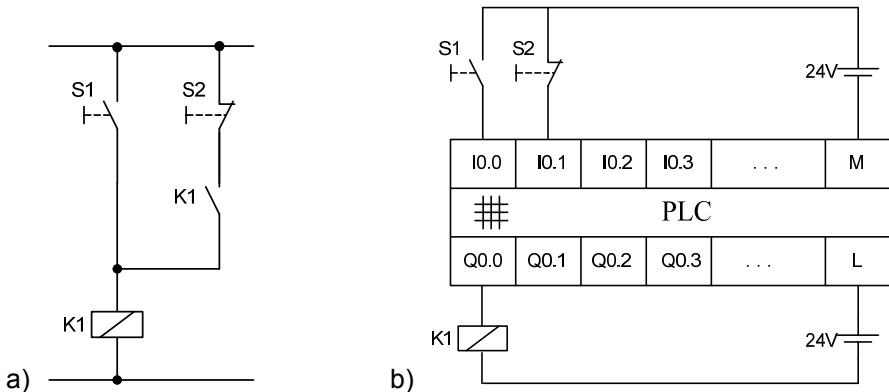
Phần này trình bày một số ví dụ ứng dụng nhỏ sử dụng các liên kết logic. Ở một số ví dụ có trình bày mạch điều khiển thông thường với kiểu nối dây khi không dùng PLC để chúng ta thấy sự giống nhau và khác nhau giữa 2 kiểu điều khiển.

### 7.4.1 Mạch tự duy trì ưu tiên mở máy

Mạch điều khiển dùng contactor có chức năng nhớ là mạch tự duy trì.

Trong trường hợp nếu cả hai nút nhấn mở máy S1 và dừng S2 cùng tác động mà contactor có điện thì là mạch tự duy trì ưu tiên mở máy.

Bảng ký hiệu		
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Nút nhấn mở máy, thường hở (NO)
S2	I0.1	Nút nhấn dừng máy, thường đóng (NC)
K1	Q0.0	Contactor



Hình 7.11 Mạch ưu tiên mở máy: a) mạch điều khiển, b) nối dây PLC

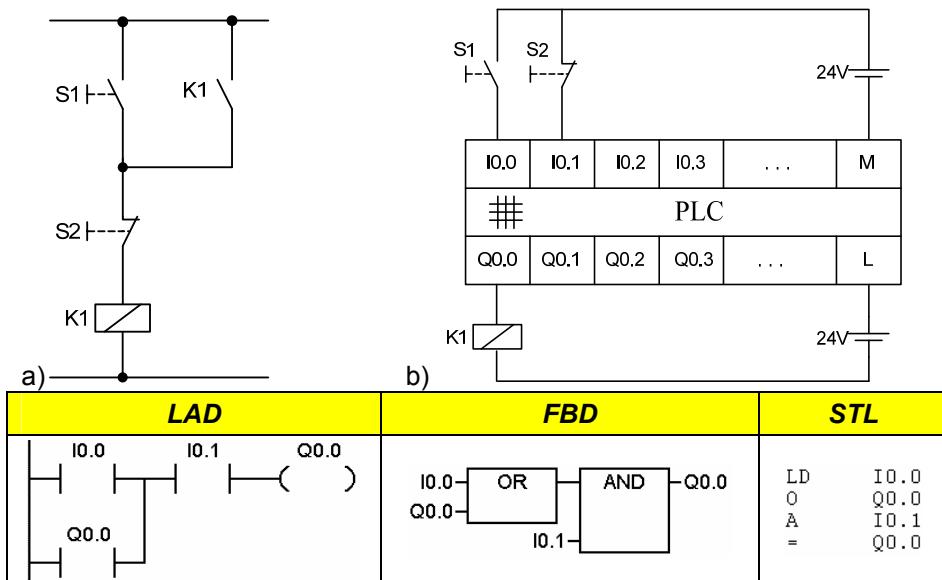
LAD	FBD	STL
		<pre> LD   I0.0 LD   I0.1 A    Q0.0 OLD  =       Q0.0   </pre>

Hình 7.12 Chương trình mạch tự duy trì ưu tiên mở máy:

#### 7.4.2 Mạch tự duy trì ưu tiên dùng máy

Trong trường hợp nếu cả hai nút nhấn mở máy S1 và dừng S2 cùng tác động mà contactor không có điện thì là mạch tự duy trì ưu tiên dừng máy.

Bảng ký hiệu		
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Nút nhấn mở máy, thường hở (NO)
S2	I0.1	Nút nhấn dừng máy, thường đóng (NC)
K1	Q0.0	Contactor



Hình 7.13 Mạch ưu tiên dùng máy:

a) mạch điều khiển, b) nối dây PLC và chương trình

#### 7.4.3 Điều khiển ON/OFF động cơ có chỉ báo

Một động cơ điện 3 pha được điều khiển bằng một PLC S7-200. Khi nhấn nút S2 (thường hở) thì động cơ sẽ chạy. Khi nhấn nút S1 (thường đóng) thì động cơ sẽ dừng lại. Các chế độ hoạt động chạy và dừng được báo bằng 2 đèn báo H1 và H2.

Các thiết bị động lực gồm có:

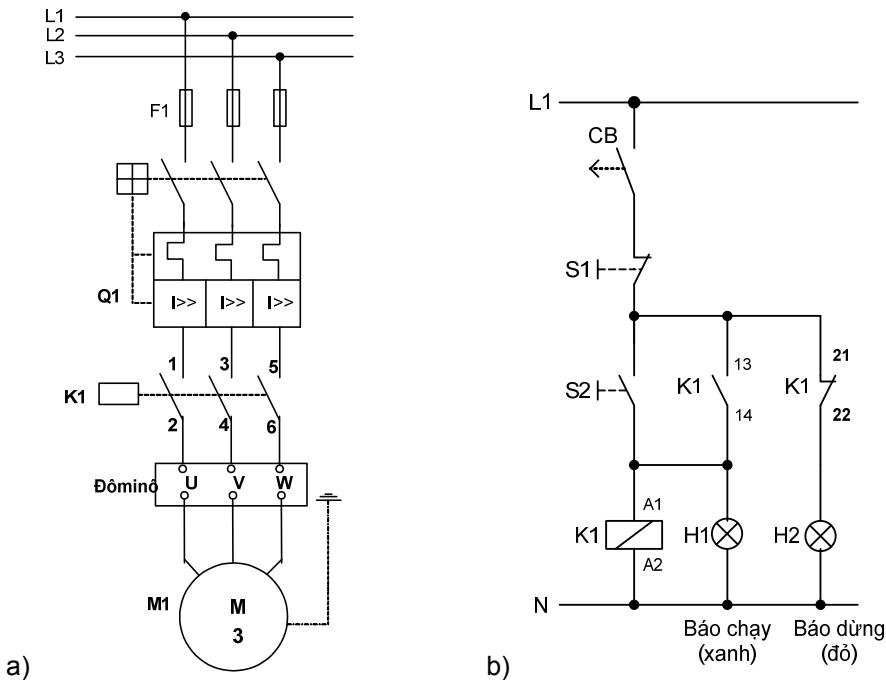
- Cầu chì 3 pha F1
- CB bảo vệ động cơ (Motor CB) Q1
- Contactor K1

Khi điều khiển dùng PLC thì mạch động lực vẫn giữ nguyên. Phần mạch điều khiển được biến đổi thành chương trình. Cần chú ý rằng các thiết bị điện như nút nhấn, CB, đèn báo đều giữ nguyên không thay đổi.

Nếu ta sử dụng PLC S7-200 loại DC/DC/DC thì ngõ ra của PLC cần phải kết nối với một relay trung gian K11 sử dụng nguồn 24Vdc. Relay này được dùng để đóng điện cho cuộn dây contactor K1 (hình 7.15). Riêng các đèn báo ta có thể thay thế bằng loại 24Vdc nhằm tiết kiệm relay trung gian.

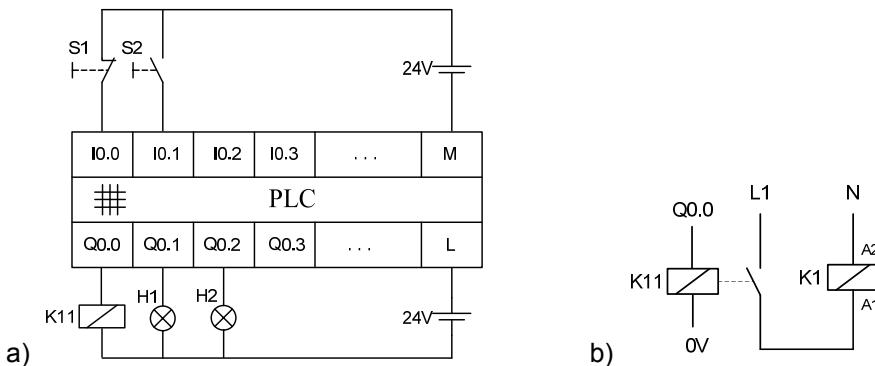
**Chú ý:** Cũng có thể sử dụng loại CPU DC/DC/RLY, thì ngõ ra của nó có thể kết nối trực tiếp với cuộn dây K1. (xem thêm chương 5 về nối dây PLC với ngoại vi).

Bảng ký hiệu		
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Nút nhấn dừng máy, thường đóng (NC)
S2	I0.1	Nút nhấn mở máy, thường hở (NO)
K11	Q0.0	Relay trung gian
H1	Q0.1	Đèn báo động cơ hoạt động
H2	Q0.2	Đèn báo động cơ dừng



Hình 7.14 Mạch ON/OFF động cơ dùng contactor.

a) Mạch động lực; b) Mạch điều khiển



Hình 7.15: a) Sơ đồ nối dây PLC

b) Nối relay trung gian với contactor

+ Chương trình:

Biểu diễn ở STL:

**Network 1** ON/OFF doing control

LD	I0.0
LD	I0.1
O	Q0.0
ALD	
=	Q0.0

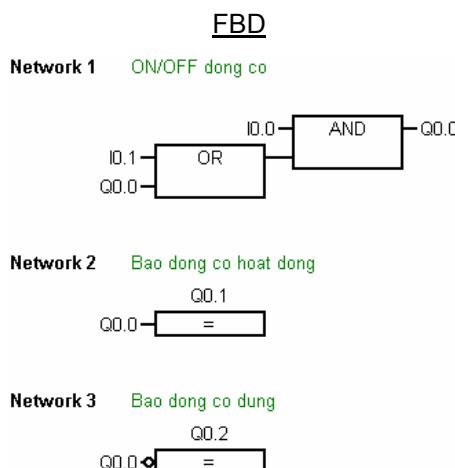
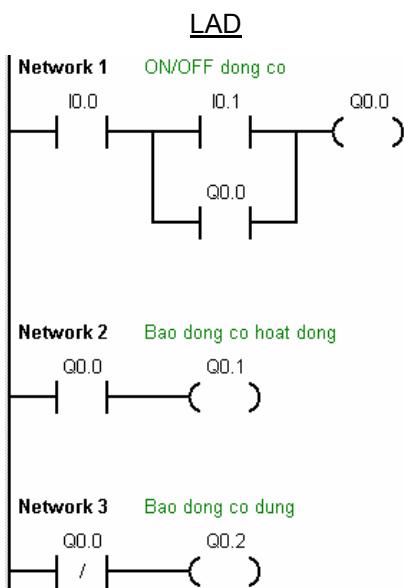
## **Network 2** Bao dong co hoat dong

LD = Q0.0  
= Q0.1

### **Network 3 Bao dong co dung**

LDN = Q0 . 0

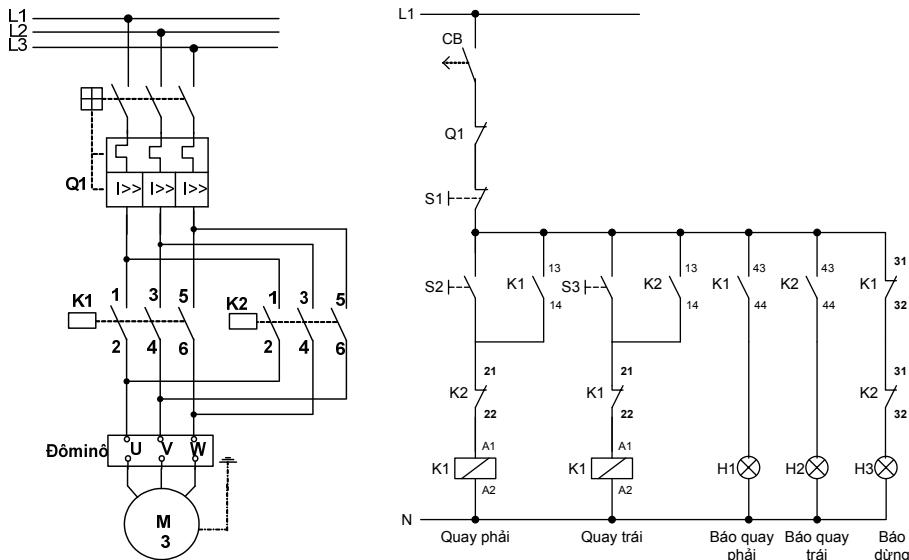
#### Biểu diễn ở LAD và FBD:



#### 7.4.4 Điều khiển đảo chiều quay động cơ

Một động cơ điện 3 pha cần được điều khiển đảo chiều. Khi án S1 (thường hở) thì động cơ sẽ quay phải và đèn H1 sáng báo động cơ đang quay phải. Khi nhấn nút S2 (thường hở) thì động cơ quay trái và đèn H2 sáng báo động cơ đang quay trái. Động cơ có thể dừng bất cứ lúc nào nếu án nút dừng S3 (thường đóng) hoặc động cơ xảy ra sự cố quá dòng làm cho tiếp điểm của thiết bị bảo vệ Q1 tác động (tiếp điểm 13, 14 của Motor CB). Khi động cơ dừng đèn báo H3 sáng.

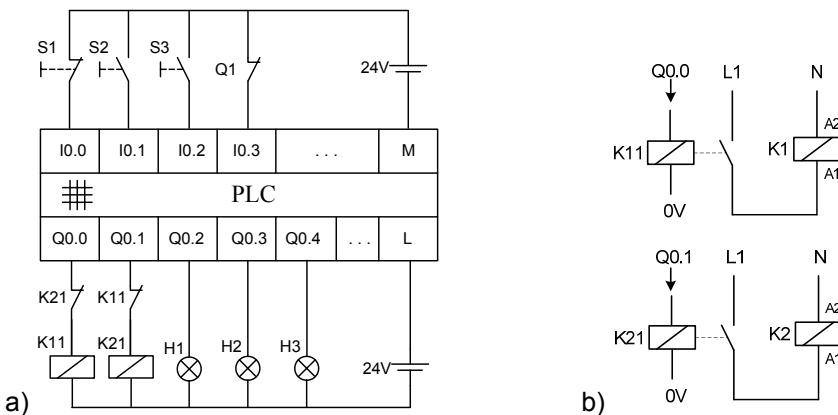
Tương tự như mục 7.4.3, ta sử dụng PLC S7-200 loại DC/DC/DC, ngõ ra của PLC điều khiển quay phải kết nối với relay trung gian K11, ngõ ra của PLC điều khiển quay trái kết nối với relay trung gian K21 sử dụng nguồn 24Vdc. Các relay này được dùng để đóng điện cho cuộn dây contactor K1 và K2 (hình 7.17). Riêng các đèn báo ta có thể thay thế bằng loại 24Vdc nhằm tiết kiệm relay trung gian.



Hình 7.16 Mạch động lực và điều khiển đảo chiều quay động cơ dùng contactor

Bảng ký hiệu

Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Nút nhấn dừng máy, thường đóng (NC)
S2	I0.1	Nút nhấn quay phải, thường hở (NO)
S3	I0.2	Nút nhấn quay trái, thường hở (NO)
Q1	I0.3	Tiếp điểm báo quá dòng, thường đóng (NC)
K11	Q0.0	Relay trung gian điều khiển quay phải
K21	Q0.1	Relay trung gian điều khiển quay trái
H1	Q0.2	Đèn báo động cơ quay phải
H2	Q0.3	Đèn báo động cơ quay trái
H3	Q0.4	Đèn báo động cơ dừng

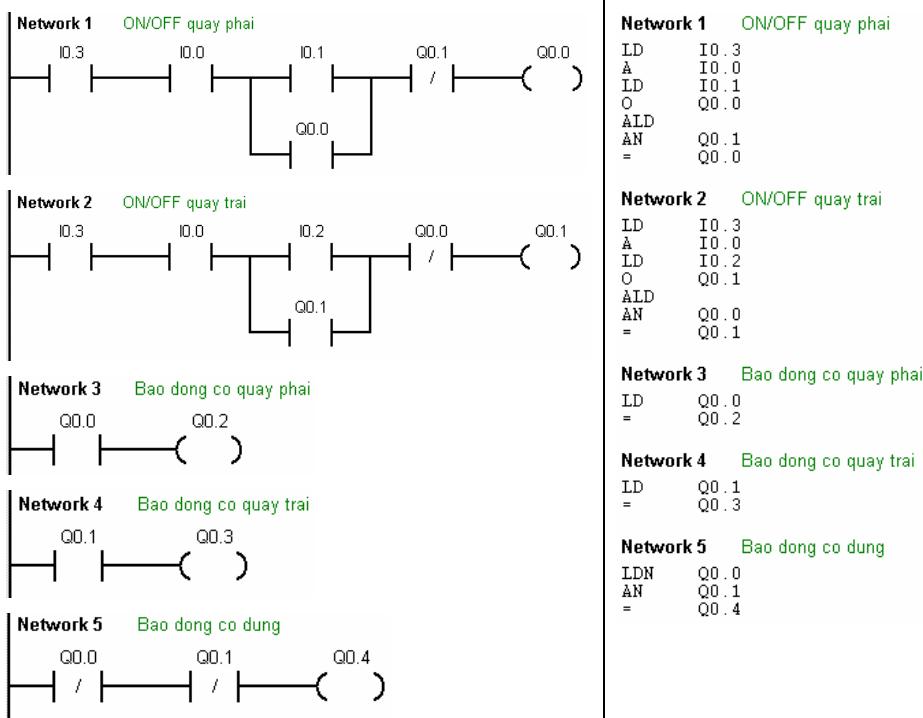


Hình 7.17 a) Sơ đồ nối dây PLC; b) Nối relay với contactor

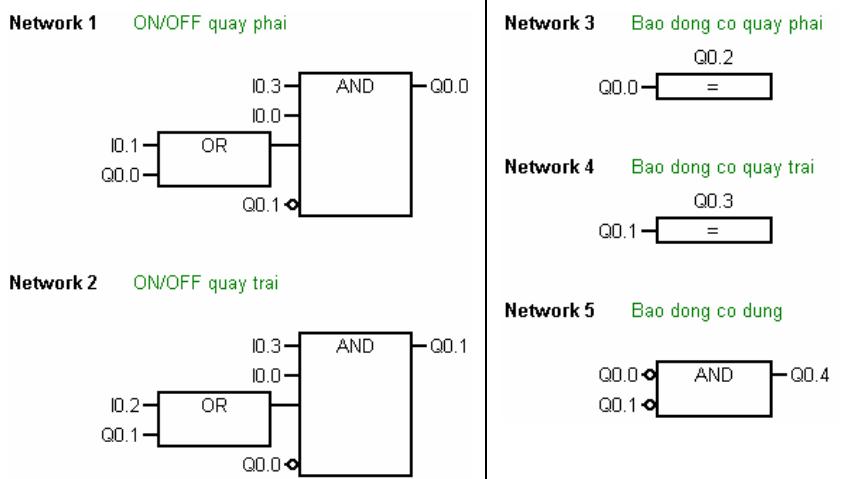
**Chú ý:** Trong các điều khiển có đảo chiều quay thì tại các ngõ ra PLC điều khiển 2 chiều quay của động cơ ta cần phải nối thêm 2 tiếp điểm thường đóng khóa chéo nhau của 2 contactor (hoặc relay) để đảm bảo an toàn.

### Chương trình PLC:

Biểu diễn ở LAD và STL:

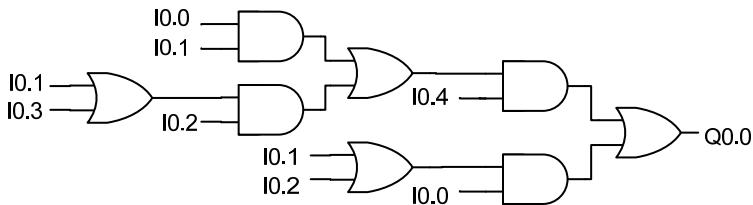


Biểu diễn ở FBD:

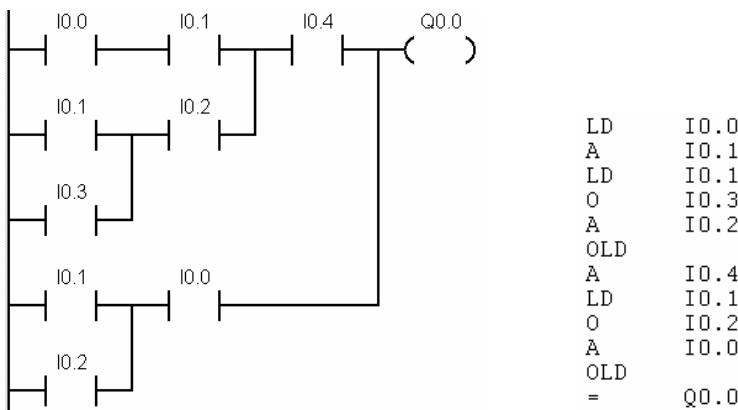


## 7.5 Bit nhớ M (bit memory)

Trong thiết kế các chương trình điều khiển, ta có thể có một số lượng lớn các logic được liên kết với nhau. Ví dụ như mạch sau:



Chương trình được viết ở LAD và STL:



Với các liên kết logic như thế này thì việc tìm lỗi rất khó khăn. Để dễ dàng hơn trong lập trình và tìm lỗi, thì các kết quả trung gian sẽ được lưu vào một ô nhớ. Trong S7-200 thì các ô nhớ này là bit memory (M).

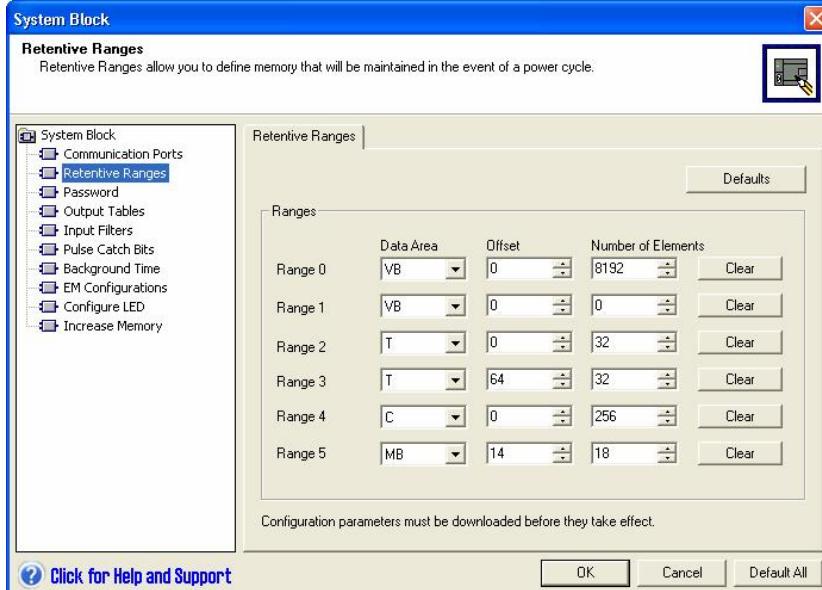
Trong S7-200 có 32 byte nhớ M (từ M0.0 đến M31.7). Chúng được xem như là các ngõ ra trung gian. Khi mất nguồn cấp thì nội dung được nhớ trong các bit nhớ M có thể bị mất hoặc vẫn còn giữ lại tùy thuộc vào việc đặt thuộc tính cho vùng nhớ này là retentive (nhớ lâu dài) hay non-retentive (không nhớ lâu dài).

\* *Bit memory có thuộc tính Retentive*: Các bit có thuộc tính này đều giữ lại giá trị của nó khi nguồn cung cấp bị mất. Nghĩa là nếu trước khi bị mất điện, ô nhớ M có giá trị nào thì nó vẫn giữ nguyên giá trị đó khi PLC bị mất điện. Các ô nhớ được ứng dụng để nhớ các trạng thái hoạt động của máy móc hay thiết bị trước khi bị mất điện. Ở lần khởi động kế tiếp thì các máy móc hay thiết bị có thể tiếp tục làm việc tại vị trí trước lúc mất điện. Vùng retentive được thiết

lập bằng cách nhấp chuột vào biểu tượng system Block hoặc vào menu **View > Component > System Block**. Chọn mục **Retentive Ranges**. Nếu chọn thẻ **defaults** thì tất cả các vùng nhớ có thuộc tính retentive đều theo

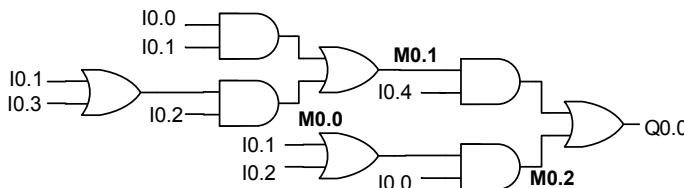
chuẩn của nhà sản xuất. Đối với vùng nhớ M thì bắt đầu từ byte MB14 đến MB31. Tuy nhiên chúng ta vẫn có thể đặt lại theo ý muốn (hình 7.18).

\* Bit memory có thuộc tính non-retentive: Giá trị các bit này bị xóa khi PLC mất nguồn cung cấp. Theo chuẩn nhà sản xuất thì ta có MB0 đến MB13 ở thuộc tính non-retentive.



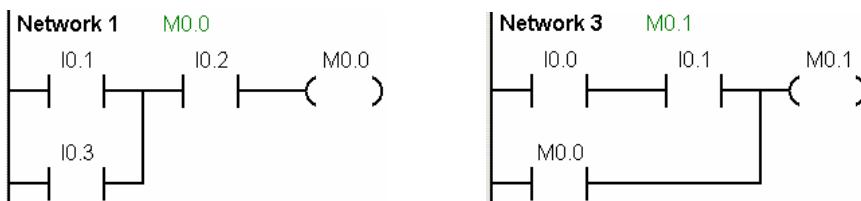
Hình 7.18: Màn hình thiết lập retentive memory.

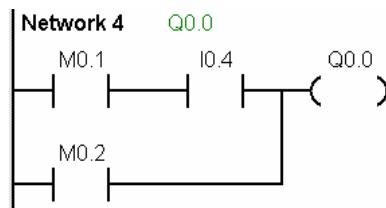
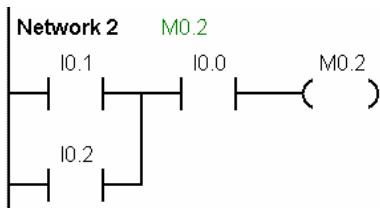
Khi sử dụng bit memory (M), ta có thể làm cho chương trình dễ đọc hơn. Sơ đồ mạch như hình 7.19.



Hình 7.19: Mạch logic được làm cho dễ đọc hơn với bit memory.

Chương trình ở LAD và STL như sau:



STL:**Network 1 M0.0**

```

LD      I0.1
O
A
=
M0.0

```

**Network 3 M0.1**

```

LD      I0.0
A
O
=
M0.1

```

**Network 2 M0.2**

```

LD      I0.1
O
A
=
M0.2

```

**Network 4 Q0.0**

```

LD      M0.1
A
O
=
Q0.0

```

## 7.6 Các lệnh SET, RESET và mạch nhớ RS

### 7.6.1 Lệnh SET

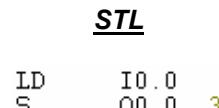
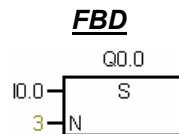
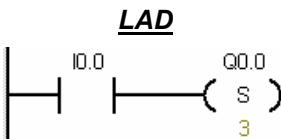
Lệnh SET (S) là lệnh thông dụng rất thường được sử dụng và lệnh này đều có trong hầu hết các PLC. Lệnh Set sẽ đặt trạng thái của một hoặc nhiều bit (thuộc vùng nhớ V, M, Q, T, C, SM, L) có địa chỉ liên tục lên mức 1 và duy trì ở trạng thái này cho đến khi bị xóa bằng một lệnh khác. Chúng ta có thể Set một lần tối đa tới 255 bit. Lệnh SET chỉ được thực hiện khi Stack 0 có giá trị logic “1”.

Cú pháp ở STL: S S\_Bit, n

và ở LAD:

Với *S\_Bit* là bit đầu tiên của vùng nhớ cần đặt lên mức logic “1”.và *n* là số lượng bit bắt đầu từ *S\_Bit*.

**Ví dụ:** Khi tín hiệu tại I0.0 lên mức 1 thì sẽ set 3 bit từ Q0.0 đến Q0.2. Chương trình ở 3 dạng như sau:



Khi tín hiệu tại I0.0 xuống mức 0 thì 3 ngõ ra Q0.0, Q0.1, Q0.2 vẫn duy trì mức 1.

## 7.6.2 Lệnh RESET (R)

Lệnh Reset (R) đặt trạng thái của một hoặc nhiều bit có địa chỉ liên tục xuống mức 0. Tương tự như lệnh Set chúng ta có thể Reset tối 255 bit nhớ thuộc các vùng nhớ V, M, Q, T, C, SM, L. Lệnh RESET chỉ được thực hiện khi Stack 0 có giá trị logic “1”.



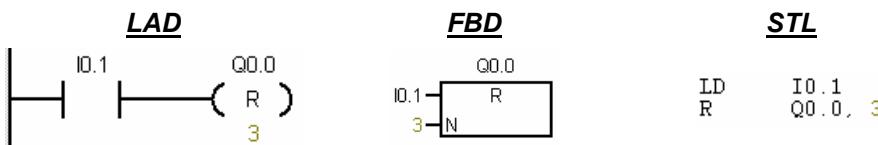
Cú pháp ở STL:  $R \ S\_Bit, n$

và ở LAD:

Với  $S\_Bit$  là bit đầu tiên của vùng nhớ cần đặt xuống mức logic “0”.

và  $n$  là số lượng bit bắt đầu từ  $S\_Bit$ .

**Ví dụ:** Khi tín hiệu tại I0.1 lên mức 1 thì sẽ reset 3 bit từ Q0.0 đến Q0.2 về logic “0”. Chương trình ở 3 dạng như sau:



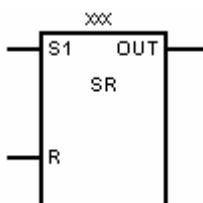
## 7.6.3 Mạch nhớ R-S

Mạch nhớ là mạch có hai trạng thái ổn định và thông qua tín hiệu ngõ vào mà trạng thái của nó thay đổi. Đối với mạch điều khiển dùng relay và contactor ta có mạch tự duy trì. Còn trong PLC có khâu R-S (viết tắt của Reset và Set).

Mạch nhớ R-S là rất cần thiết trong kỹ thuật điều khiển. Nó được xem là một chức năng cơ bản trong hầu hết các loại PLC và được chia thành hai loại là: *Ưu tiên SET* và *Ưu tiên RESET*.

### 7.6.3.1 Ưu tiên SET (khâu SR)

Biểu diễn ở LAD:



Với:

xxx: Địa chỉ cần điều khiển

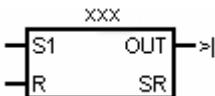
S1: Ngõ vào Set. Ký hiệu ưu tiên Set.

R: Ngõ vào Reset.

OUT: Ngõ ra, có thể nối với một địa chỉ dạng bit

SR: Ký hiệu gọi nhớ khâu SR

và FBD:



Nếu cả hai điều kiện cho S và R lên mức logic “1” thì ngõ ra OUT là “1”.

Bảng sự thật

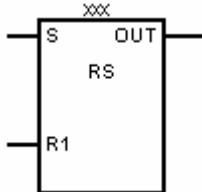
S1	R	OUT
0	0	Trạng thái trước
0	1	0
1	0	1
1	1	1

Để lấy khâu SR, ta nhấp chuột vào dấu cộng của Bit Logic trong cây lệnh, chọn phần tử SR và kéo thả vào network mong muốn.

Khâu SR tương đương với mạch tự duy trì ưu tiên mở máy trong điều khiển dùng contactor.

### 7.6.3.2 Ưu tiên RESET (khâu RS)

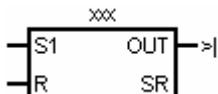
Biểu diễn ở LAD:



Với:

- xxx: Địa chỉ cần điều khiển
- S: Ngõ vào Set.
- R1: Ngõ vào Reset. Ký hiệu ưu tiên ReSet.
- OUT: Ngõ ra, có thể nối với một địa chỉ dạng bit
- RS: Ký hiệu gợi nhớ khâu RS

và FBD:



Nếu cả hai điều kiện cho S và R lên mức logic “1” thì ngõ ra OUT là “0”.

Bảng sự thật

S1	R	OUT
0	0	Trạng thái trước
0	1	0
1	0	1
1	1	0

Để lấy khâu RS, ta nhấp chuột vào dấu cộng của Bit Logic trong cây lệnh, chọn phần tử RS và kéo thả vào network mong muốn.

Khâu RS tương đương với mạch tự duy trì ưu tiên dừng máy trong điều khiển dùng contactor.

### 7.6.4 Các qui tắc khi sử dụng Set và Reset

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”.

### 7.6.5 Ví dụ ứng dụng mạch nhớ R-S

#### Ví dụ 7.7 : Mạch ưu tiên mở máy.

Yêu cầu của mạch ưu tiên mở máy như ở mục 7.4.1, tuy nhiên cần phải sử dụng mạch nhớ R-S khi lập trình.

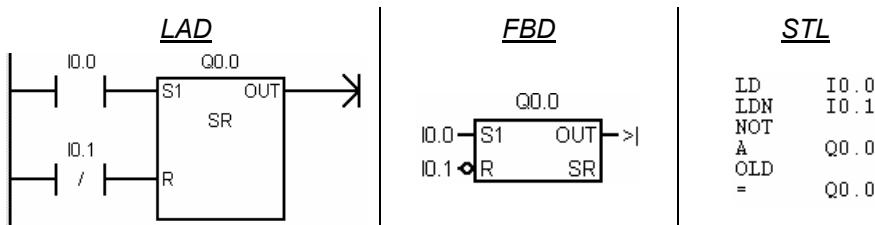
*Để tránh lập lại ta sử dụng lại bảng ký hiệu và sơ đồ nối dây PLC ở mục 7.4.1*

*Phân tích:* Theo yêu cầu của mạch ta có các nhận xét sau:

1. Điều kiện để cho contactor K1 có điện là nút nhấn S1 được ấn → nút nhấn S1 được sử dụng với lệnh S.
2. Điều kiện để cho contactor K1 mất điện là nút nhấn S2 được ấn → nút nhấn S2 được sử dụng với lệnh R.
3. Khi cả hai nút nhấn S1 và S2 cùng ấn thì contactor có điện → sử dụng mạch nhớ **ưu tiên SET (khâu SR)**.
4. Trạng thái bình thường của nút nhấn S1 là thường hở (logic “0” tại ngõ vào I0.0) nên khi lập trình sử dụng tiếp điểm không đảo trạng thái tín hiệu (tiếp điểm   ). Còn S2 là thường đóng (logic “1”)

tại ngõ vào I0.1) nên khi lập trình sử dụng tiếp điểm đảo trạng thái tín hiệu (tiếp điểm  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$  /  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$ ).

Chương trình được viết như sau:



### Ví dụ 7.8 : Mạch ưu tiên dừng máy.

Yêu cầu của mạch ưu tiên dừng máy như ở mục 7.4.2, tuy nhiên cần phải sử dụng mạch nhớ R-S khi lập trình.

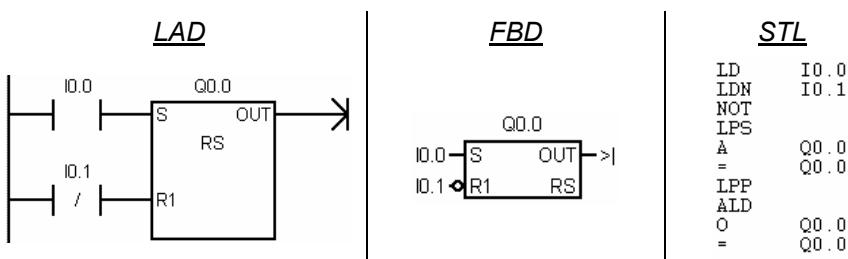
*Để tránh lập lại ta sử dụng lại bảng ký hiệu và sơ đồ nối dây PLC ở mục 7.4.2*

*Phân tích:* Theo yêu cầu của mạch ta có các nhận xét sau:

1. Điều kiện để cho contactor K1 có điện là nút nhấn S1 được ấn → nút nhấn S1 được sử dụng với lệnh S.
2. Điều kiện để cho contactor K1 mất điện là nút nhấn S2 được ấn → nút nhấn S2 được sử dụng với lệnh R.
3. Khi cả hai nút nhấn S1 và S2 cùng ấn thì contactor mất điện → sử dụng mạch nhớ ưu tiên **RESET (khâu RS)**.
4. Trạng thái bình thường của nút nhấn S1 là thường hở (logic “0” tại ngõ vào I0.0) nên khi lập trình sử dụng tiếp điểm không đảo trạng thái tín hiệu (tiếp điểm  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$  /  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$ ). Còn S2 là thường đóng (logic “1” tại ngõ vào I0.1) nên khi lập trình sử dụng tiếp điểm đảo trạng thái tín hiệu (tiếp điểm  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$  /  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$ ).

tín hiệu (tiếp điểm  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$  /  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$ ). Còn S2 là thường đóng (logic “1” tại ngõ vào I0.1) nên khi lập trình sử dụng tiếp điểm đảo trạng thái tín hiệu (tiếp điểm  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$  /  $\begin{smallmatrix} / \\ \text{---} \end{smallmatrix}$ ).

Chương trình được viết như sau:



### Ví dụ 7.9 : Mạch đảo chiều quay động cơ.

Để đơn giản và dễ hiểu, ví dụ này lấy lại yêu cầu công nghệ của mạch điều khiển đảo chiều quay ở mục 7.4.4. Tuy nhiên cần phải sử dụng mạch nhớ R-S khi lập trình.

*Để tránh lập lại ta sử dụng lại bảng ký hiệu và sơ đồ nối dây PLC ở mục 7.4.4.*

*Phân tích:* Theo yêu cầu công nghệ ta có các nhận xét sau:

#### 1. Đối với contactor K1 (được đóng điện gián tiếp bởi K11).

- *Điều kiện Set (làm cho K1 có điện):* Nút nhấn S2 được ấn. Tuy nhiên vì lý do an toàn K2 mất điện mới được phép mở máy nên phải kết hợp thêm điều kiện K2 mất điện.

$$\text{Set K1} = \text{S2} \wedge \overline{\text{K2}}$$

- *Điều kiện Reset (làm cho K1 mất điện):* Có 2 khả năng là hoặc nút nhấn dừng S1 được ấn hoặc tiếp điểm bảo vệ quá dòng Q1 tác động.

$$\text{Reset K1} = \overline{\text{S1}} \vee \overline{\text{Q1}}$$

- *Vì lý do an toàn, K1 bị mất điện nếu điều kiện SET và RESET cho nó cùng ở logic “1” → sử dụng khâu SR.*

#### 2. Đối với contactor K2 (được đóng điện gián tiếp bởi K21)

- *Điều kiện Set:* Nút nhấn S3 được ấn. Tuy nhiên vì lý do an toàn K1 mất điện mới được phép mở máy nên phải kết hợp thêm điều kiện K1 mất điện.

$$\text{Set K2} = \text{S3} \wedge \overline{\text{K1}}$$

- *Điều kiện Reset:* Có 2 khả năng là hoặc nút nhấn dừng S1 được ấn hoặc tiếp điểm bảo vệ quá dòng Q1 tác động.

$$\text{Reset K2} = \overline{\text{S1}} \vee \overline{\text{Q1}}$$

- *Vì lý do an toàn, K2 bị mất điện nếu điều kiện SET và RESET cho nó cùng ở logic “1” → sử dụng khâu SR.*

#### 3. Đối với đèn báo H1.

- *Đèn sáng khi K1 có điện và tắt khi K1 mất điện*  

$$\text{H1} = \text{K1}$$

#### 4. Đối với đèn báo H2

- *Đèn sáng khi K2 có điện và tắt khi K2 mất điện.*  

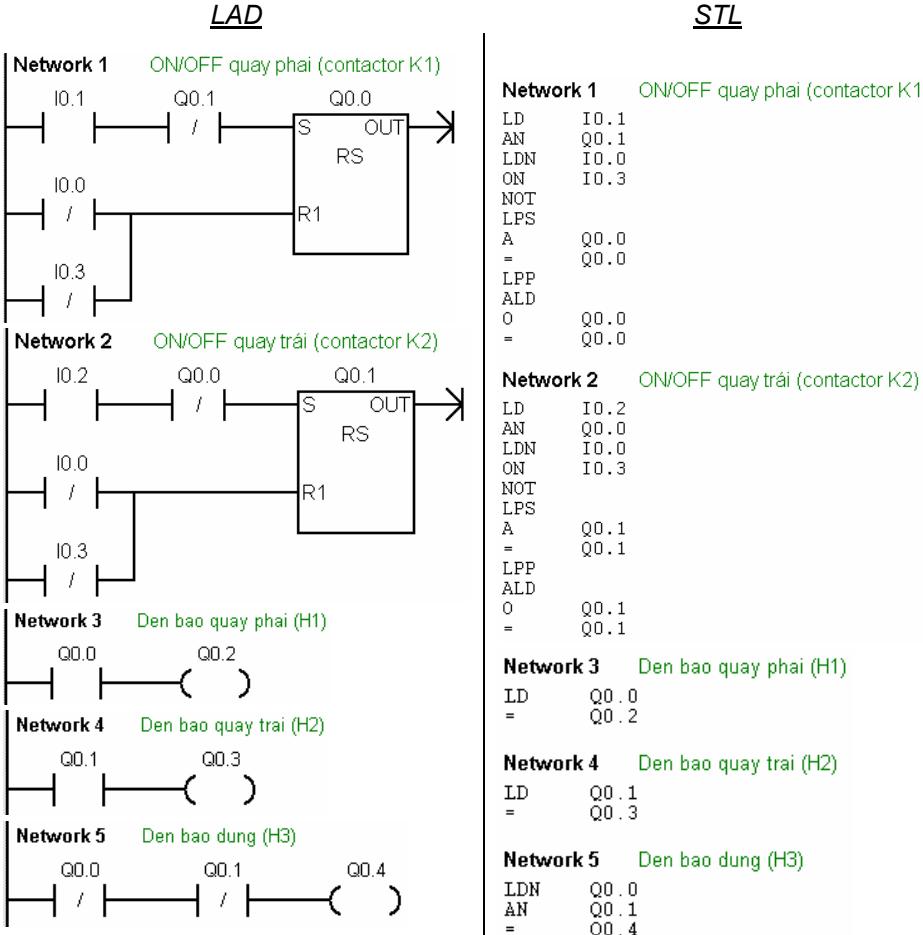
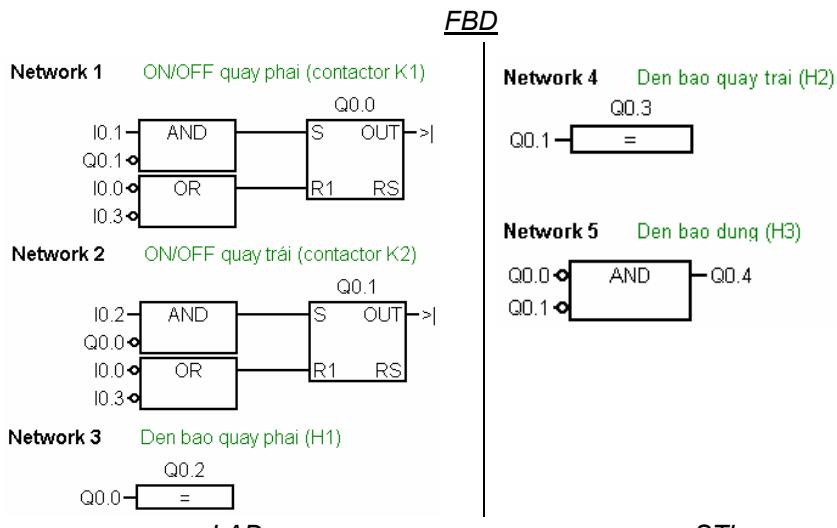
$$\text{H2} = \text{K2}$$

#### 5. Đối với đèn báo H3

- *Đèn sáng khi cả K1 và K2 mất điện.*

$$\text{H3} = \overline{\text{K1}} \wedge \overline{\text{K2}}$$

Theo các phân tích ta viết được chương trình như sau:



## 7.7 Các lệnh nhận biết cạnh tín hiệu và lệnh NOT

Các lệnh nhận biết cạnh tín hiệu và lệnh NOT thực hiện các thuật toán đặc biệt trên bit đầu tiên của ngăn xếp (Stack 0).

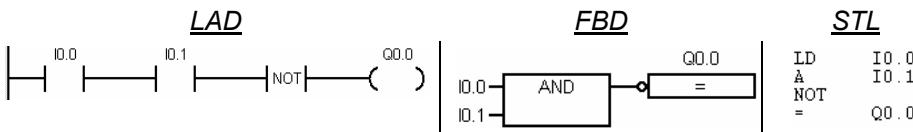
### 7.7.1 Lệnh NOT

Lệnh NOT đảo giá trị của bit đầu tiên trong ngăn xếp (Stack 0). Nếu sau một phép toán nhị phân mà sử dụng lệnh NOT thì kết quả sẽ bị đảo lại. Nghĩa là nếu kết quả phép toán nhị phân làm cho Stack 0 có giá trị logic “1” thì lệnh NOT sẽ cho kết quả là “0”, và ngược lại.

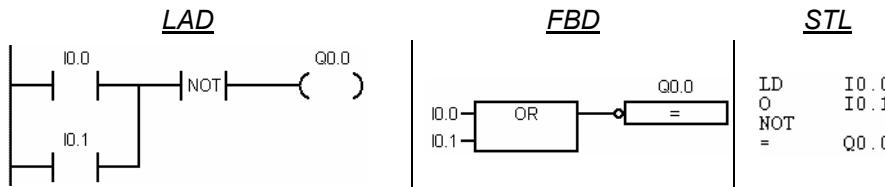
- Kết hợp lệnh NOT sau các cổng logic như OR, AND, XOR ta thu được các cổng NOR, NAND, XNOR.

Ví dụ:

- Cổng NAND với 2 ngõ vào I0.0 và I0.1 và ngõ ra Q0.0 là:



- Cổng NOR với 2 ngõ vào I0.0 và I0.1 và ngõ ra Q0.0 là:



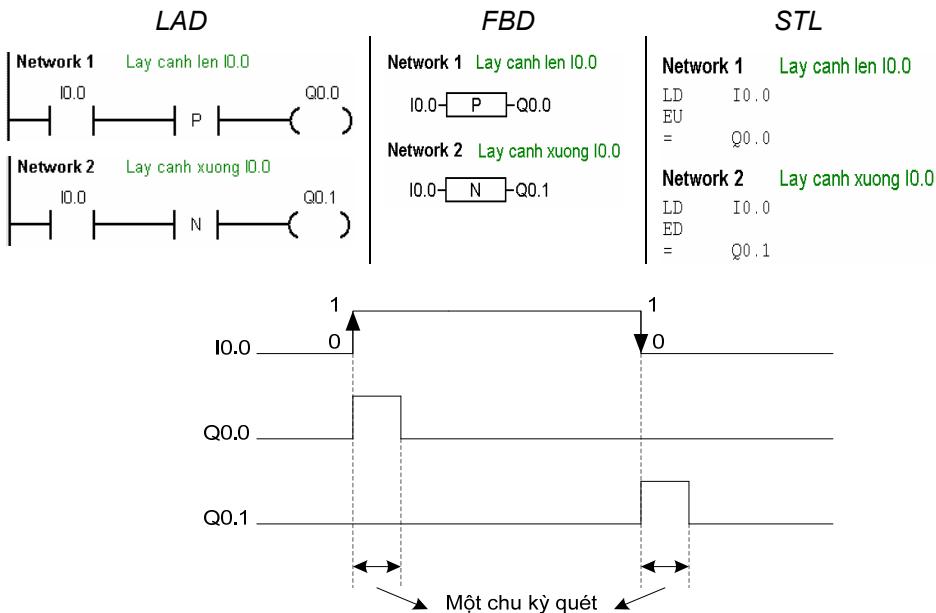
### 7.7.2 Các lệnh nhận biết cạnh tín hiệu

Hai lệnh nhận biết cạnh tín hiệu là lệnh nhận biết cạnh lên (EU) và nhận biết cạnh xuống (ED).

Lệnh nhận biết cạnh lên (EU) sẽ đặt giá trị logic “1” vào bit đầu tiên của Stack 0 trong một chu kỳ quét chương trình khi phát hiện sự chuyển trạng thái từ 0 lên 1 trong Stack 0. Còn các trường hợp khác nó sẽ đặt Stack 0 về “0”.

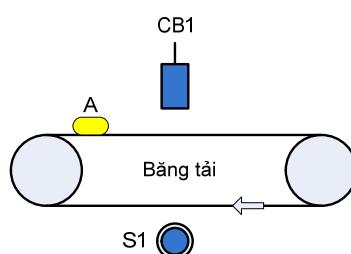
Lệnh nhận biết cạnh xuống (ED) sẽ đặt giá trị logic “1” vào bit đầu tiên của Stack 0 trong một chu kỳ quét chương trình khi phát hiện sự chuyển trạng thái từ 1 xuống 0 trong Stack 0. Còn các trường hợp khác nó sẽ đặt Stack 0 về “0”.

Ví dụ: Lấy cạnh lên của I0.0 xuất ra Q0.0, còn cạnh xuống xuất ra Q0.1.



Hình 7.20: Giản đồ thời gian của ví dụ lấy cạnh lên và xuống của tín hiệu.

**Ví dụ 7.10:** Viết chương trình điều khiển đơn giản cho băng tải sản phẩm (hình 7.21). Khi sản phẩm A được vận chuyển đến vị trí cần thao tác thì băng tải dừng lại (được phát hiện bởi cảm biến CB1).Ấn nút S1 thì băng tải tiếp tục hoạt động cho đến khi nào một sản phẩm đến đúng vị trí thì dừng lại. Quá trình cứ lặp lại như trên.



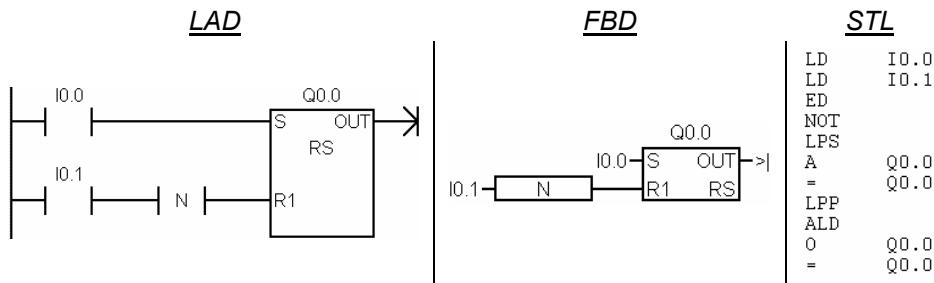
Hình 7.21: ví dụ 7.10

*Phân tích:*

- Điều kiện Set băng tải: Nút nhấn S1
- Điều kiện Reset băng tải: Cảm biến CB1.
- Sản phẩm đến cảm biến CB1 thì băng tải dừng lại, như vậy cảm biến luôn bị tác động. Nếu ta dùng ưu tiên Reset thì không thể nào khởi động lại băng tải. Còn nếu dùng ưu tiên Set thì khi nào sản phẩm qua khỏi cảm biến mới có thể buông tay thả nút nhấn S1 → Dùng lệnh nhận biết cạnh tín hiệu để khống chế. Và để chắc chắn sản phẩm đã qua cảm biến thì sử dụng lệnh nhận biết cạnh xuống.

Chương trình như sau:

	Symbol	Address	Comment
1	S1	I0.0	Nút nhấn khởi động băng tải
2	CB1	I0.1	Cảm biến sản phẩm đúng vị trí
3	K1	Q0.0	Contactor đóng điện cho động cơ kéo băng tải



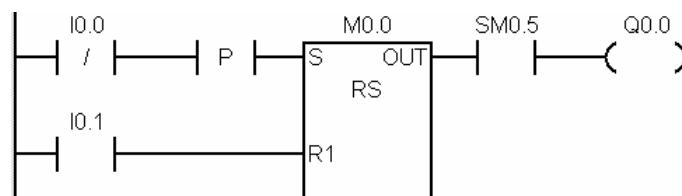
## 7.8 Các Bit nhớ đặc biệt (Special Memory bits)

Các bit nhớ SM (Special memory bits) cung cấp nhiều chức năng trạng thái và điều khiển, cũng như cung cấp thông tin truyền thông giữa S7-200 và chương trình. Các bit nhớ đặc biệt có thể được sử dụng ở dạng bits, bytes, words và double words. Trong phần này chỉ trình bày các bit trạng thái của SMB0. Còn các bit nhớ SM khác sẽ được trình bày ở mỗi chương tương ứng trong quyển sách này và ở quyển tiếp theo (tập 2).

SMB0 chứa tám bit trạng thái và được cập nhật ở mỗi chu kỳ quét của S7-200. Đây là các bit nhớ chỉ đọc.

Bit	Chức năng
SM0.0	Bit luôn luôn có trạng thái 1
SM0.1	Bit có trạng thái 1 ở vòng quét đầu tiên của chương trình
SM0.2	Bit báo dữ liệu bị thất lạc (0: dữ liệu còn đủ, 1: dữ liệu bị thất lạc).
SM0.3	Bit báo PLC được đóng nguồn. (1: ở vòng quét đầu tiên, 0: ở các vòng quét còn lại).
SM0.4	Bit tạo ra xung có chu kỳ 1 phút (0: trong 30s đầu, 1 trong 30s sau).
SM0.5	Bit tạo xung có chu kỳ 1s (tần số 1 Hz) (0: trong 0,5s đầu ; 1 trong 0,5 s sau).
SM0.6	Bit lên 1 ở một vòng quét và xuống 0 ở vòng quét tiếp theo. Nó được sử dụng để làm ngõ vào của bộ đếm vòng quét.
SM0.7	Bit báo vị trí của công tắc chọn chế độ làm việc của PLC (0: TERM, 1: RUN).

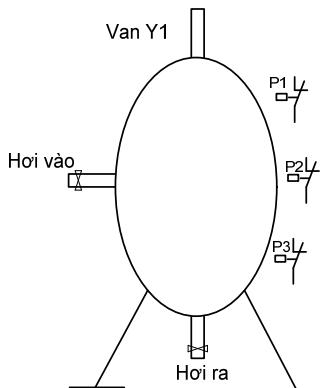
**Ví dụ:** Khi có tín hiệu sự cố (ngõ vào I0.0 (NC) xuống mức 0) thì đèn báo sự cố (Q0.0) sẽ nhấp nháy 1 Hz. Nhấn nút I0.1 để Reset.



## 7.9 Câu hỏi và bài tập

Các bài tập ứng dụng giả sử dùng CPU 224 DC/DC/DC để điều khiển.

### BT7.1 An toàn cho lò hơi



Hình 7.22 Mô hình lò hơi

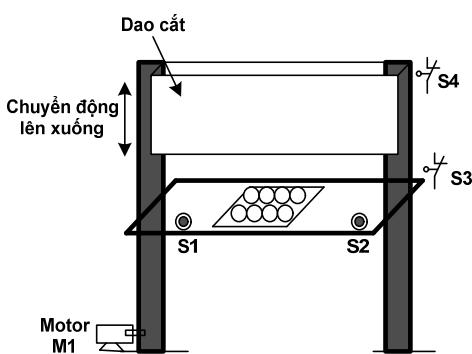
Một thiết bị lò hơi có hơi đi vào và ra khỏi lò được thực hiện tự động qua bộ điều chỉnh đặt ở bên ngoài. Lò hơi có đặt 3 bộ cảm biến áp suất P1, P2 và P3 ở các vị trí khác nhau để kiểm soát quá áp suất. Mạch an toàn sẽ hoạt động khi có sự cố, trường hợp áp suất trong lò hơi tăng quá cao thì van an toàn từ tính Y1 sẽ hoạt động xả bớt hơi ra ngoài. Cần có ít nhất bắt kỳ hai trong ba cảm biến tác động thì mạch an toàn mở van từ tính Y1. Hãy :

- Viết chương trình sao cho nếu có **bắt kỳ 2 trong 3** cảm biến tác động thì van Y1 mở.
- Vẽ sơ đồ nối dây tín hiệu phần cứng

\* Bảng ký hiệu:

	Symbol	Address	Comment
1	Sensor_P1	I0.0	Tiep diem NC
2	Sensor_P2	I0.1	Tiep diem NC
3	Sensor_P3	I0.2	Tiep diem NC
4	Van_Y1	Q0.0	
5			

### BT7.2 Điều khiển cơ cấu máy dập



Hình 7.23 Mô hình máy dập nhỏ

Một cơ cấu dập trong một máy dập nguyên liệu (ví dụ dập ra các vỏ hộp) có thể chuyển động nâng lên hay hạ xuống nhờ một động cơ điện M1 quay 2 chiều. Để đảm bảo an toàn cho tay người vận hành thì chỉ khi nào người vận hành dùng cả 2 tay nhấn đồng thời 2 nút nhấn S1 (NO) và S2 (NO) thì bàn dập mới hạ xuống. Khi hạ xuống đụng công tắc hành trình giới hạn dưới S3 (NC) thì tự chạy nâng lên cho tới khi đụng công tắc hành trình giới hạn trên S4 (NC) thì dừng lại. Chu kỳ lặp lại

khi nào người vận hành lại nhấn 2 nút nhấn S1 và S2.

\* Bảng ký hiệu:

		Symbol	Address	Comment
1		S1	I0.0	Nút nhấn NO
2		S2	I0.1	Nút nhấn NO
3		S3	I0.2	Công tắc hành trình NC
4		S4	I0.3	Công tắc hành trình NC
5		K1	Q0.0	Contactor Motor M1 quay phải
6		K2	Q0.1	Contactor Motor M1 quay trái

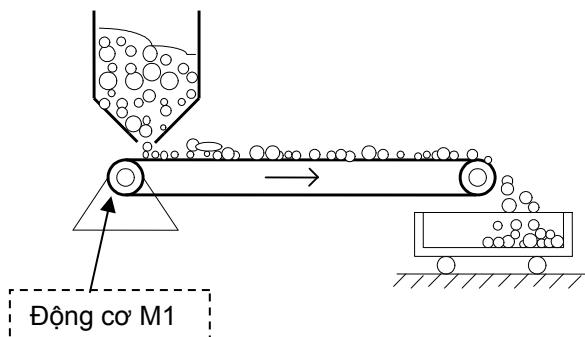
Hãy :

- Viết chương trình điều khiển
- Vẽ sơ đồ nối dây phanh cứng

### BT7.3 Băng tải chuyển vật liệu

Một thiết bị băng tải dùng để chuyển vật liệu từ thùng chứa vào xe gác. Hãy viết chương trình sao cho: Khi bật công tắc khởi động S0 (NO), thì đèn H0 sáng báo hệ thống sẵn sàng làm việc. Khi nhấn nút S1 (NO) động cơ M1 chạy kéo băng tải và nguyên liệu trong thùng chứa được vận chuyển theo băng tải. Khi nhấn nút dừng S2 (NC) thì băng tải dừng lại. Khi xảy ra sự cố quá dòng (tiếp điểm nhiệt F3 (NC) tác động) thì động cơ sẽ dừng lại.

❖ Sơ đồ công nghệ:



Hình 7.24 Băng tải chuyển vật liệu

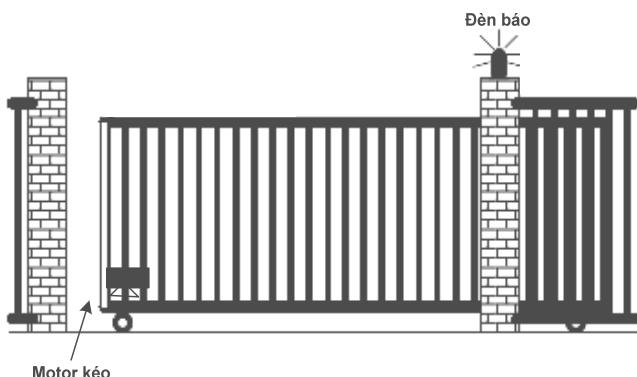
\* Bảng ký hiệu:

	Symbol	Address	Comment
1	S0	I0.0	Cong tac khai dong
2	S1	I0.1	Start
3	S2	I0.2	Stop
4	F3	I0.3	Tiep diem nhan NC
5	H1	Q0.0	Den bao san sang
6	M1	Q0.1	Contactor Motor M1

#### BT7.4 Điều khiển cổng ra vào

Một cổng ở công ty cần được điều khiển ở 2 chế độ tay và tự động nhờ một công tắc chọn S0 có 2 vị trí :

- **Ở chế độ tay:** Nhấn nút mở S1 (NO) thì động cơ M1 quay phải và cổng mở ra, nếu thả tay ra thì động cơ dừng lại. Tuy nhiên, nếu cổng mở ra đụng công tắc hành trình giới hạn mở S3 (NC) thì cũng dừng lại. Tương tự, nếu nhấn nút đóng S2 (NO) thì động cơ M1 quay trái và cổng đóng lại, nếu thả tay ra thì động cơ dừng lại. Nếu đụng công tắc hành trình giới hạn đóng S4 (NC) thì cổng cũng dừng lại.
- **Ở chế độ tự động:** Nhấn nút mở thì cửa sẽ mở cho tới khi đụng công tắc hành trình giới hạn mở S3 mới dừng lại. Khi nhấn nút đóng, cổng sẽ đóng lại cho tới khi đụng công tắc hành trình đóng S4 mới dừng lại.
- Có thể dừng quá trình đóng hoặc mở bất cứ lúc nào nếu nhấn nút dừng S5 (NC) hoặc động cơ bị quá tải (tiếp điểm nhiệt F3 (NC) tác động ).
- Trong quá trình đóng hoặc mở một đèn báo H1 sẽ sáng lên báo cổng đang hoạt động. Hãy :
  - Viết 2 chương trình con: Sub0 cho chế độ tay và Sub1 cho chế độ tự động.
  - Vẽ sơ đồ nối dây phần cứng
    - ❖ Sơ đồ công nghệ:



Hình 7.15 Điều khiển cổng

❖ Bảng ký hiệu:

		Symbol	Address	Comment
1		S0	I0.0	= 0 la che do tay, = 1 la che do tu dong
2		S1	I0.1	Mo cua
3		S2	I0.2	Dong cua
4		S3	I0.3	Cong tac hanh trinh mo
5		S4	I0.4	Cong tac hanh trinh dong
6		S5	I0.5	Stop
7		F3	I0.6	Qua tai
8		H1	Q0.0	Bao hoat dong
9		K1	Q0.1	Contactor cho M1 quay phai (mo cua)
10		K2	Q0.2	Contactor cho M1 quay trai (dong cua)

### BT7.5 Điều khiển xe rót vật liệu vào bồn chứa

Một xe kéo dùng để rót vật liệu vào bồn chứa. Khi bật công tắc khởi động S0 (NO) thì đèn H0 sáng báo hệ thống sẵn sàng làm việc. Khi nhấn nút S1 (NO), động cơ M1 có điện kéo xe di chuyển lên, đồng thời đèn H1 chớp sáng với tần số 1Hz. Khi xe lên tới vị trí trên cùng đụng phải công tắc hành trình S4 (NC) thì dừng lại. Nhấn nút S2 (NO) động cơ M1 đảo chiều và kéo xe di chuyển xe xuống, đồng thời đèn báo H2 chớp với tần số 1Hz. Khi xe đến vị trí cuối cùng đụng phải công tắc hành trình S3 (NC) thì dừng lại. Khi động cơ M1 có sự cố quá dòng (tiếp điểm nhiệt F3 (NC) tác động) thì động cơ sẽ dừng lại và đèn H0 sẽ chớp sáng với tần số 1Hz.. Quá trình mới được khởi động khi bật lại công tắc S0. Hãy:

- Viết chương trình điều khiển
- Vẽ sơ đồ nối dây phàn cứng với PLC

❖ Bảng ký hiệu:

		Symbol	Address	Comment
1		S0	I0.0	Cong tac khai dong
2		S1	I0.1	Xe chay len
3		S2	I0.2	Xe chay xuong
4		S3	I0.3	Cong tac hanh trinh duoi
5		S4	I0.4	Cong tac hanh trinh tren
6		F3	I0.5	Qua tai
7		H0	Q0.0	Bao hoat dong + Bao su co
8		H1	Q0.1	Bao xe chay len
9		H2	Q0.2	Bao xe chay xuong
10		K1	Q0.3	Contactor cho M1 quay phai (Xe len)
11		K2	Q0.4	Contactor cho M1 quay trai (Xe xuong)

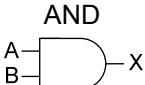
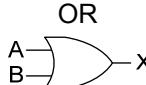
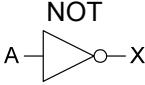
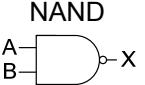
## 8 Thiết kế theo logic Bool & biểu đồ Karnaugh

### 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.

 $X = A \cdot B$	 $X = A + B$	 $X = \bar{A}$	 $X = \overline{A \cdot B}$																																																			
<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <thead> <tr> <th>A</th><th>B</th><th>X</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </tbody> </table>	A	B	X	0	0	0	0	1	0	1	0	0	1	1	1	<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <thead> <tr> <th>A</th><th>B</th><th>X</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </tbody> </table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	1	<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <thead> <tr> <th>A</th><th>X</th></tr> </thead> <tbody> <tr><td>0</td><td>1</td></tr> <tr><td>1</td><td>0</td></tr> </tbody> </table>	A	X	0	1	1	0	<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <thead> <tr> <th>A</th><th>B</th><th>X</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </tbody> </table>	A	B	X	0	0	0	0	1	0	1	0	0	1	1	1
A	B	X																																																				
0	0	0																																																				
0	1	0																																																				
1	0	0																																																				
1	1	1																																																				
A	B	X																																																				
0	0	0																																																				
0	1	1																																																				
1	0	1																																																				
1	1	1																																																				
A	X																																																					
0	1																																																					
1	0																																																					
A	B	X																																																				
0	0	0																																																				
0	1	0																																																				
1	0	0																																																				
1	1	1																																																				
 $X = \overline{A + B}$	 $X = A \oplus B$	 $X = \overline{A \oplus B}$																																																				
<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <thead> <tr> <th>A</th><th>B</th><th>X</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>1</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>0</td></tr> </tbody> </table>	A	B	X	0	0	1	0	1	0	1	0	0	1	1	0	<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <thead> <tr> <th>A</th><th>B</th><th>X</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>0</td></tr> </tbody> </table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	0	<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <thead> <tr> <th>A</th><th>B</th><th>X</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>1</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </tbody> </table>	A	B	X	0	0	1	0	1	0	1	0	0	1	1	1							
A	B	X																																																				
0	0	1																																																				
0	1	0																																																				
1	0	0																																																				
1	1	0																																																				
A	B	X																																																				
0	0	0																																																				
0	1	1																																																				
1	0	1																																																				
1	1	0																																																				
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 công 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:

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

- **Các định lý của đại số Bool**

Tiên đề: 1.  $A + A = 0$

2.  $A \cdot 1 = A$

3.  $A \cdot \bar{A} = 0$

4.  $A + \bar{A} = 1$

5.  $\bar{1} = 0$

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

9.  $\overline{(A \cdot B)} = \bar{A} + \bar{B}$

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

11.  $(A \cdot B) \cdot C = A \cdot (B \cdot C)$

12.  $A + \bar{A} \cdot B = A + B$

13.  $A \cdot (\bar{A} + B) = A \cdot B$

14.  $A + B = B + A$

15.  $A \cdot B = B \cdot A$

16.  $A + (B \cdot C) = (A + B) \cdot (A + C)$

17.  $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$

Định lý DeMorgan's

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

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

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

$$\begin{aligned} A &= \bar{B} \cdot (C \cdot (\bar{D} + E + C) + \bar{F} \cdot C) \\ A &= \bar{B} \cdot (\bar{D} \cdot C + E \cdot C + C \cdot C + \bar{F} \cdot C) \quad (1) \\ A &= \bar{B} \cdot (\bar{D} \cdot C + E \cdot C + C + \bar{F} \cdot C) \quad (2) \\ A &= \bar{B} \cdot C \cdot (\bar{D} + E + 1 + \bar{F}) \quad (3) \\ A &= \bar{B} \cdot C \cdot (1) \quad (4) \\ A &= \bar{B} \cdot C \quad (5) \end{aligned}$$

**Chú ý:** Khi đơn giản các biểu thức đại số Bool, phép toán 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:

$$\begin{aligned} X &= \overline{(A + B \cdot C)} + A \cdot (B + \bar{C}) \\ X &= \overline{(A)} + (B \cdot C) + A \cdot (B + \bar{C}) \leftarrow \begin{array}{l} \text{Các phép toán có ưu tiên cao} \\ \text{được đặt trong ngoặc} \end{array} \\ X &= \overline{(A)} \cdot \overline{(B \cdot C)} + A \cdot (B + \bar{C}) \leftarrow \text{Ứng dụng định lý DeMorgan's} \\ X &= \overline{A} \cdot \overline{(B + \bar{C})} + A \cdot (B + \bar{C}) \leftarrow \text{Ứng dụng tiếp định lý DeMorgan's} \\ X &= \overline{A} \cdot \overline{B} + \overline{A} \cdot \overline{C} + A \cdot B + A \cdot \bar{C} \leftarrow \text{Bỏ ngoặc} \\ X &= \overline{A} \cdot \overline{B} + (\overline{A} \cdot \overline{C} + A \cdot \bar{C}) + A \cdot B \leftarrow \begin{array}{l} \text{Chọn các số hạng có cùng thừa} \\ \text{số, ở đây chỉ có NOT C} \end{array} \\ X &= \overline{A} \cdot \overline{B} + \overline{C} \cdot (\overline{A} + A) + A \cdot B \leftarrow \begin{array}{l} \text{Đặt thừa số chung} \\ \text{A} \end{array} \\ X &= \overline{A} \cdot \overline{B} + \overline{C} + A \cdot B \leftarrow \text{Ứng dụng định lý để đơn giản} \end{aligned}$$

### 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 toán 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 được 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 loại 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

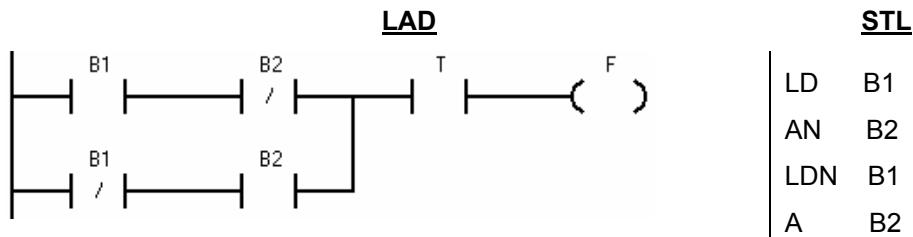
Biểu thức đại số Bool:

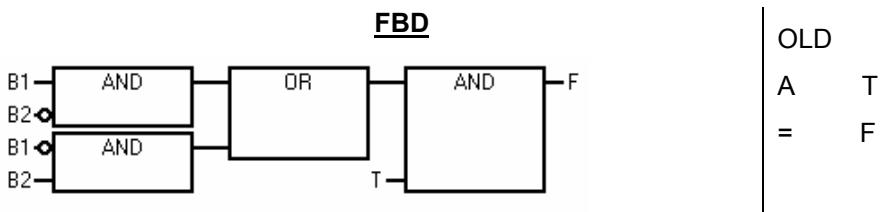
$$F = T \cdot (B_1 \oplus B_2) \quad (1)$$

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

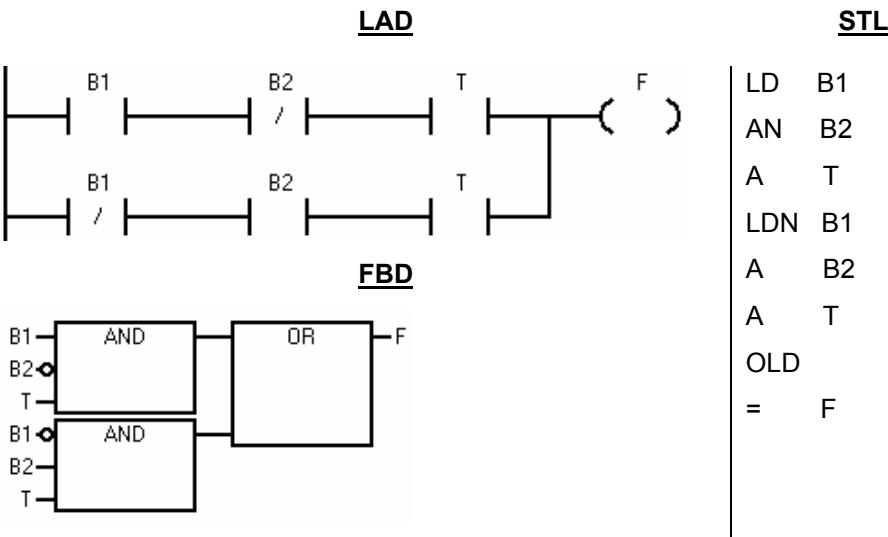
$$F = B_1 \cdot \overline{B_2} \cdot T + \overline{B_1} \cdot B_2 \cdot T \quad (3)$$

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





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):

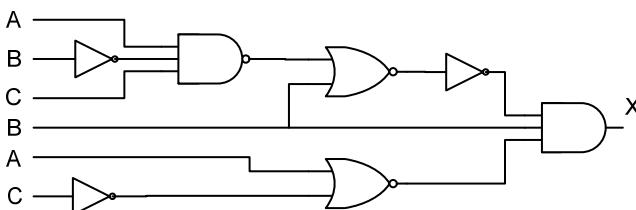


Hình 8.3: Biểu thức đại số Bool được thiết kế theo ngôn ngữ của PLC S7-200

**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 công 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.

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

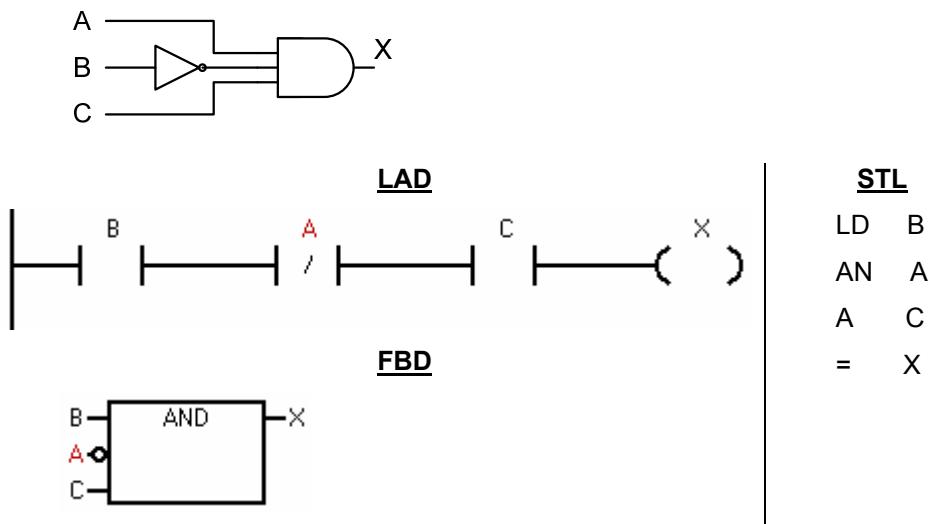
$$X = (\overline{A} + B + \overline{C} + B) \cdot B \cdot (\overline{A} \cdot C)$$

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

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

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

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



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:

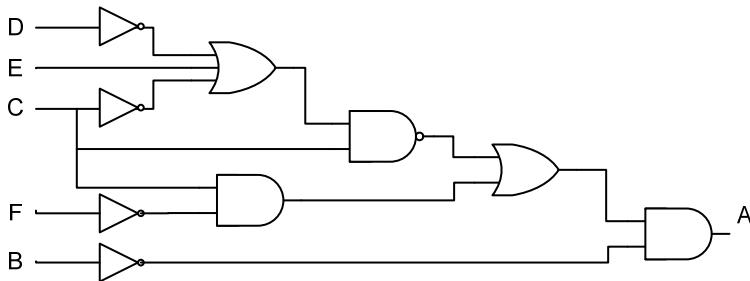
- Loại bỏ các cổng NOT không cần thiết. Thông thường có thể thực hiện bằng cách thay thế các cổng NAND và NOR bằng một biểu thức đơn giản hơn sử dụng định lý DeMorgan.
- Loại bỏ các công thức phức tạp như XOR.

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

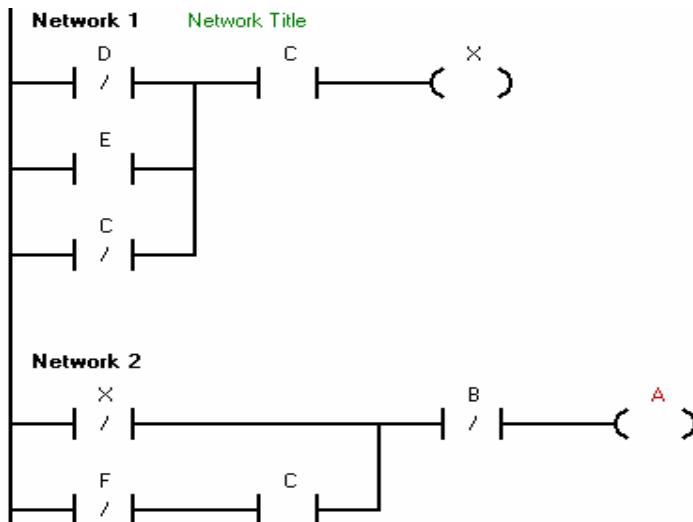
**Ví dụ 8.3:** Cho biểu thức điều khiển:

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

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 Boolean sang LAD

### 8.3.1 Các kỹ thuật đại số Boolean

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.

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

Chứng minh:  $A + C\bar{A}$

$$\begin{aligned} &\Leftrightarrow (A + C)(A + \bar{A}) \\ &\Leftrightarrow (A + C)(1) \\ &\Leftrightarrow A + C \end{aligned}$$

$$AB + A = A$$

Chứng minh:

$$AB + A$$

$$\Leftrightarrow AB + A1$$

$$\Leftrightarrow A(B + 1)$$

$$\Leftrightarrow A(1)$$

$$\Leftrightarrow A$$

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

Chứng minh:

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

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

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

$$\Leftrightarrow (\overline{AB})\overline{C}$$

$$\Leftrightarrow \overline{ABC}$$

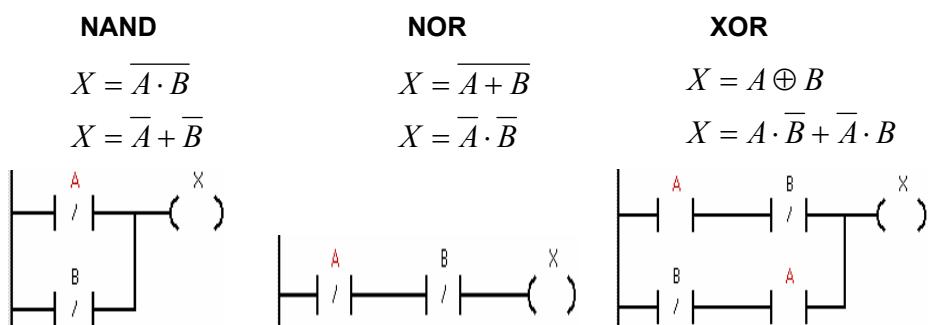
Hình 8.7: Các kỹ thuật đại số Bool

## 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.

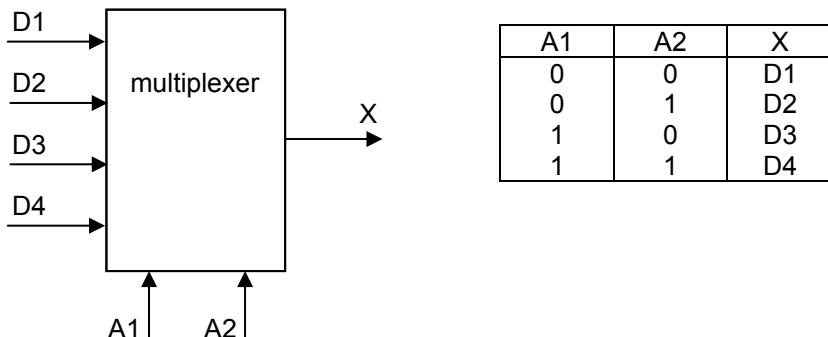


Hình 8.8: Chuyển đổi các chức năng logic phức

### 8.4.2 Multiplexers

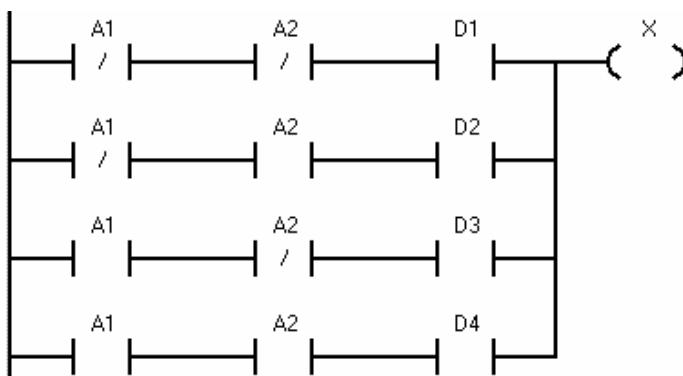
Multiplexers là sự đa hợp các thiết bị được kết nối với một thiết bị đơn. Nó rất thông dụng trong các hệ thống điện thoại. Một *chuyển mạch* điện thoại được sử dụng để xác định điện thoại nào sẽ được kết nối.

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.



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

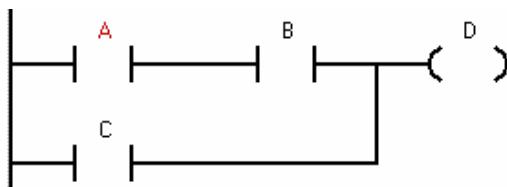
## 8.5 Một số ví dụ thiết kế đơn giản với đại số bool

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.

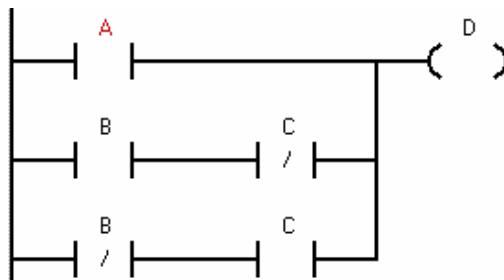
$$\text{Giải quyết: } D = (A \cdot B) + C$$



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:  $D = A + (B \oplus C)$

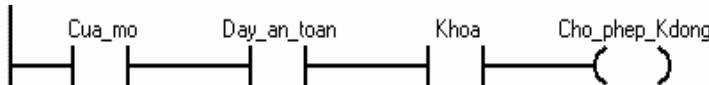


Hình 8.12: Chương trình được viết ở LAD

### 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 chốt 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:

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

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

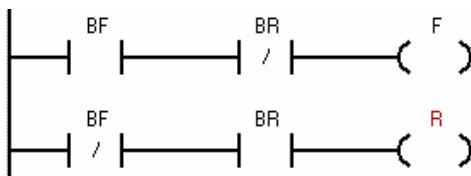
Ở đây:

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:

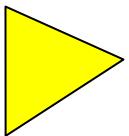
Ký hiệu	Địa chỉ	Chú thích
A	Q0.0	Đèn và cảnh báo, ON=“1”
W	I0.0	Cảm biến cửa sổ/cửa chính, thường đóng
M	I0.1	Cảm biến chuyển động, thường mở
S	I0.2	Công tắc kích hoạt cảnh báo, ON=“1”

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

- Nếu cảnh báo là “ON”, kiểm tra cảm biến.
- 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



Cảnh báo tắt

Không có kẻ trộm, tắt cảnh báo

Có kẻ trộm, Bật cảnh báo

Hình 8.15: Bảng sự thật cảnh báo trộm

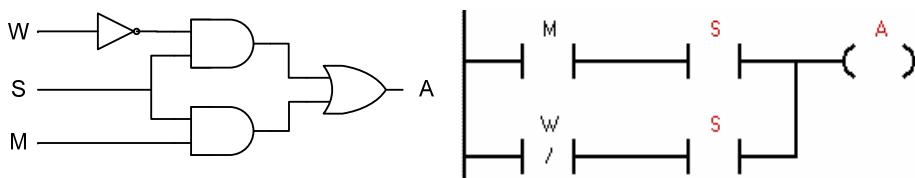
Biểu thức Boolean và đơn giản được cho ở hình 8.17 được viết từ bảng sự thật hình 8.16.

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

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

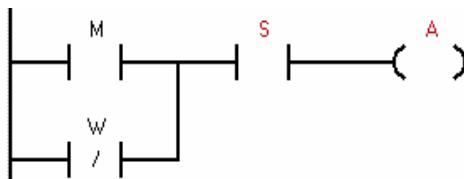
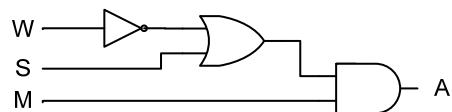
$$A = S \cdot ((\overline{M} \cdot \overline{W} + M \cdot \overline{W}) + (M \cdot \overline{W} + M \cdot W))$$

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



Hình 8.16: Biểu thức Boolean 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 Boolean 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 Boolean. 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 mở
S	I0.2	Công tắc kích hoạt cảnh báo, ON="1"

Và:

$Q = \text{Báo yên tĩnh} (0 = \text{yên tĩnh})$

#### Bước 1: Vẽ bảng sự thật

Bảng sự thật của mạch cảnh báo trộm như hình 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. Đổi 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

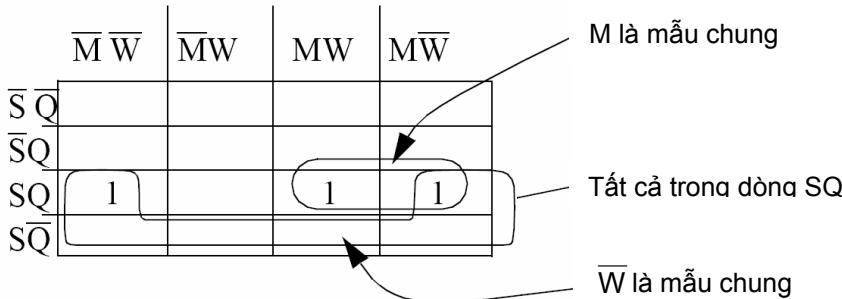
	$\bar{M}\bar{W} (= 00)$	$\bar{M}W (= 01)$	$M\bar{W} (= 11)$	$MW (= 10)$
$\bar{S}\bar{Q} (= 00)$				
$\bar{S}Q (= 01)$				
$S\bar{Q} (= 11)$	1		1	1
$SQ (= 10)$				

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  $\bar{W}$  chung. Những cái này bây giờ có thể tổ hợp thành công thức. Cuối cùng công thức được chuyển thành sơ đồ LAD.

#### Bước 4: Tìm kiếm mẫu trong bảng

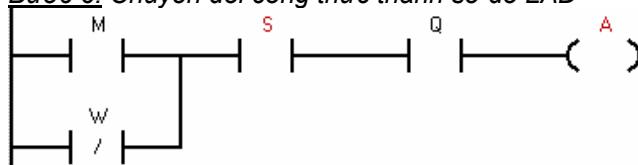


Hình 8.20: Khoanh mẫu

#### Bước 5: Viết thành công thức sử dụng các mẫu

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

#### Bước 6: Chuyển đổi công thức thành sơ đồ LAD

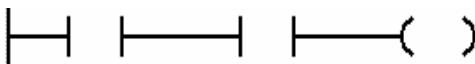


Hình 8.21: Chuyển đổi biểu thức thành sơ đồ LAD

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:** Cỗng logic được biểu diễn ở ngôn ngữ LAD cho ở dưới đây là cỗng AND hay OR?



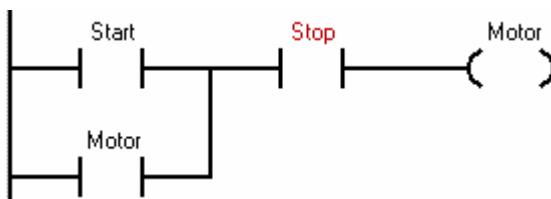
**BT 8.2:** Vẽ một sơ đồ hình thang với ngõ ra D là “1” khi công tắc A và công tắc B được đóng hoặc khi công tắc C được đóng.

**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 Boolean sau thành chương trình ở ngôn ngữ LAD đơn giản nhất có thể được.

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

**BT 8.7:** Đơn giản các biểu thức sau:

a)  $A(B + AB)$

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

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

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

**BT 8.8:** Đơn giản các biểu thức sau:

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

b)  $ABCD + \overline{A}\overline{B}CD + ABC\overline{D} + ABC\overline{D}$

**BT 8.9:** Đơn giản biểu thức Boolean sau:

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

**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.

$$Y = (\overline{ABC}D + A\overline{BC}D + \overline{ABC}\overline{D} + \overline{ABC}\overline{D}) + D$$

**BT 8.12:** Cho biểu thức đại số sau:

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

- 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

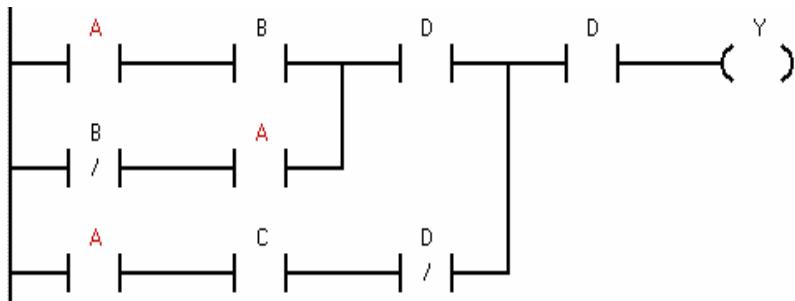
**BT 8.14:** Đơn giản biểu thức sau thành đơn giản nhất và viết thành chương trình ở ngôn ngữ LAD.

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

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

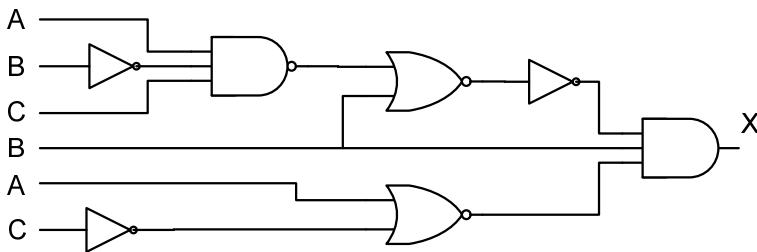
$$X = (\overline{A} + B \cdot \overline{A}) + (\overline{C} + D + E\overline{C})$$

**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

- Viết thành biểu thức ở mạch logic đã cho.
- Đơn giản biểu thức này.
- 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:

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

- Đơn giản biểu thức sử dụng đại số Bool.
- Thực hiện sơ đồ mạch số theo biểu thức ban đầu và biểu thức đã được đơn giản ở câu a).
- 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.

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

**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	$C\bar{D}$	$\bar{C}D$	$\bar{C}\bar{D}$
AB	1	0	0	1
$A\bar{B}$	0	0	0	0
$\bar{A}\bar{B}$	0	0	0	0
$\bar{A}B$	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	0	0
1	0	0	1	1	1
1	0	1	0	0	0
1	0	1	1	0	1
1	1	0	0	0	0
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	1	1	0
1	1	1	1	0	1
1	1	1	1	1	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	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	0
1	0	0	1	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	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	0
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	1
0	1	0	0	0
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	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

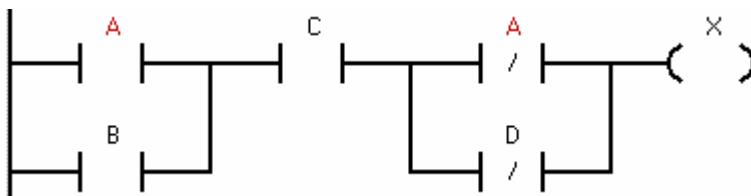
Hình BT 8.27

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

- Tìm biểu thức đại số Boolean sử dụng bảng Karnaugh.
- 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:
- Có 3 nút nhấn A, B, C.
  - Ngõ ra là “1” nếu bất kỳ hai nút nhấn nào được ấn.
  - 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.

	$\bar{A}\bar{B}$	$A\bar{B}$	$AB$	$\bar{A}B$
$CD$	1			1
$\bar{C}D$		1	1	
$\bar{C}\bar{D}$				
$\bar{C}\bar{D}$	1			1

**BT 8.31:** Xem xét biểu thức boolean sau:

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

- 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 Boolean 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*) :

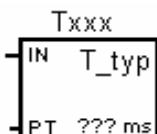
- + Timer đóng mạch chậm TON (On-delay Timer).
- + Timer đóng mạch chậm có nhớ TONR (Retentive On-delay Timer).
- + Timer ngắt mạch chậm TOF (Off-delay Timer).

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

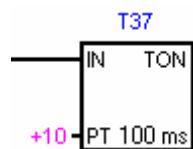
PT: Ngõ vào số Integer, hằng số thời gian.

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.

$$\text{Thời gian trì hoãn} = [\text{PT}] \times [\text{???ms}].$$

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 :  $10 \times 100\text{ms} = 1\text{s}$ .

## 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.

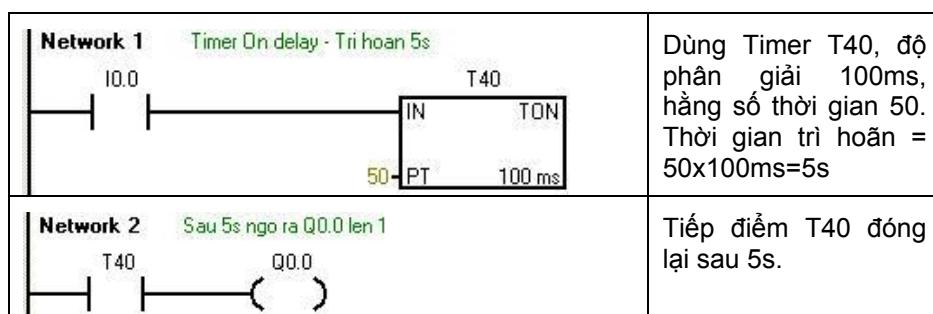
Để xóa timer, có thể sử dụng lệnh Reset (R). Lệnh Reset sẽ làm cho Timer Bit ở mức logic “0” và giá trị hiện hành của timer (Timer Current) =0.

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

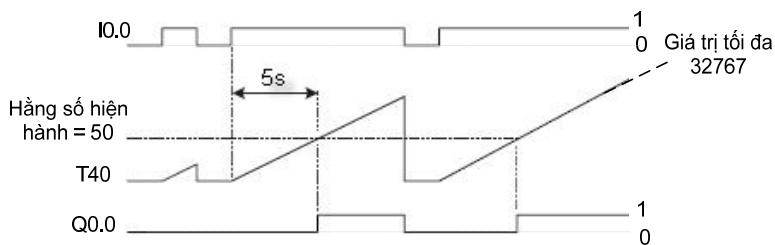
Số Timer	Độ phân giải	Thời gian trì hoãn tối đa
T32, T96	1ms	32,767s
T33 ... T36, T97 ... T100	10ms	327,67s
T37 ... T63, T101 ... T255	100ms	3276,7s

**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.



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).

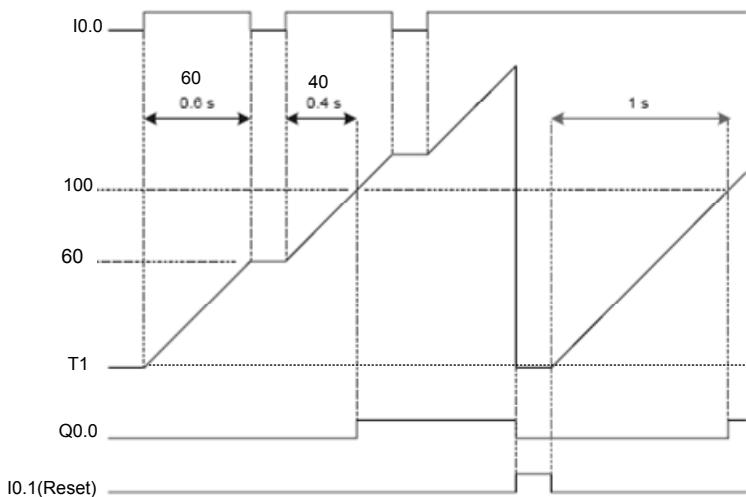
Có 64 timer TONR trong S7-200 được phân chia theo độ phân giải như ở bảng sau:

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

<b>Network 1</b> <p>Network Title</p>	Tín hiệu I0.0 kích hoạt timer TONR T1 có độ phân giải 10ms (thời gian = $100 \times 10\text{ms} = 1\text{s}$ )
<b>Network 2</b> 	Sau 1 s ngõ ra Q0.0 lên mức 1
<b>Network 3</b> 	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”.

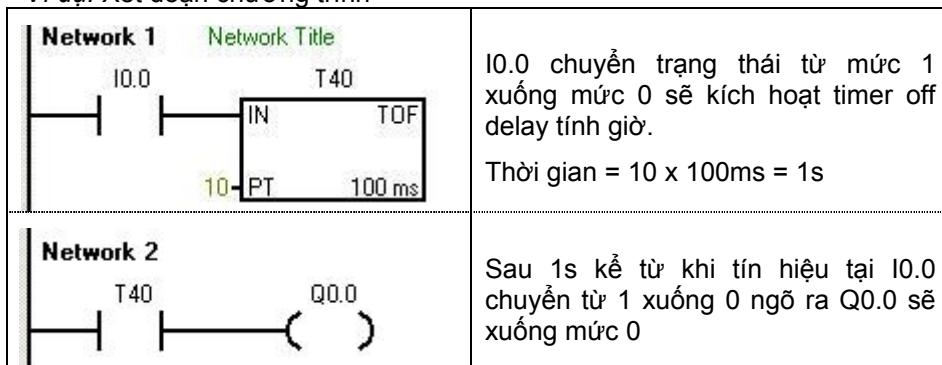
Để xóa timer, có thể sử dụng lệnh Reset (R). Lệnh Reset sẽ làm cho Timer Bit ở mức logic “0” và giá trị hiện hành của timer (Timer Current) =0.

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

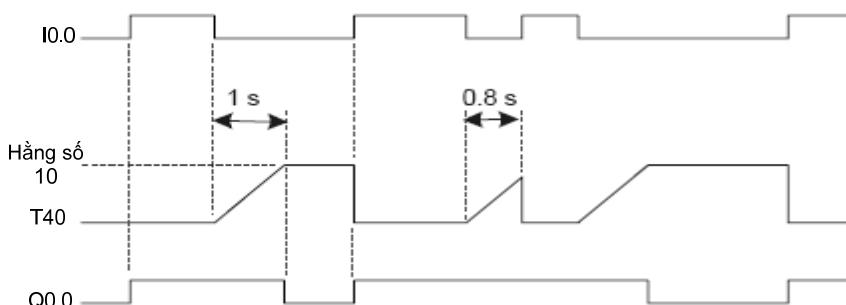
Số Timer	Độ phân giải	Thời gian trì hoãn tối đa
T32, T96	1ms	32,767s
T33 ... T36, T97 ... T100	10ms	327,67s
T37 ... T63, T101 ... T255	100ms	3276,7s

**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



Giản đồ thời gian:

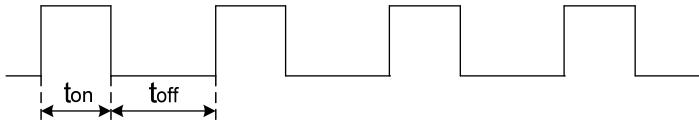


Để lấy TOF, ta nhấp chuột vào dấu (+) ở biểu tượng Timers trong cây lệnh. Sau đó trỏ chuột vào  TOF giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập số Timer cho TOF, điều kiện cho ngõ vào IN và giá trị ở PT theo mong muốn.

## 9.5 Ứng dụng Timer

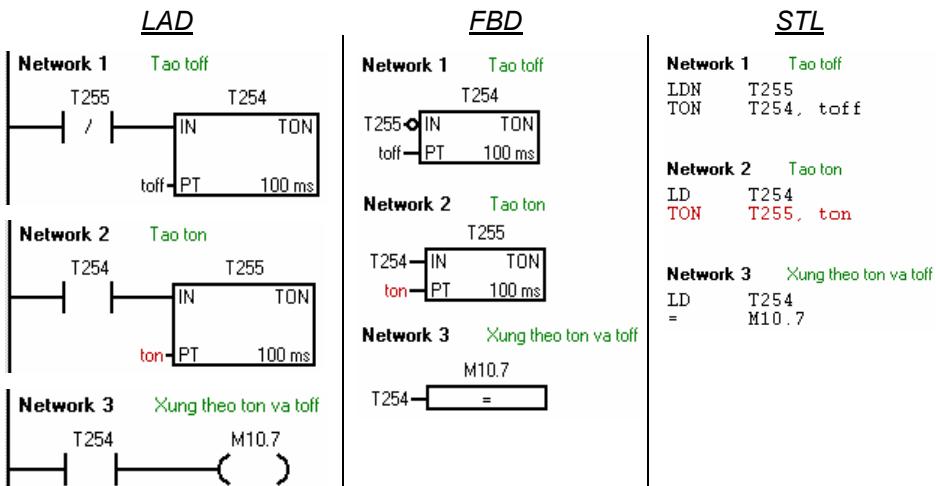
### 9.5.1 Tạo xung có tần số theo mong muốn

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.

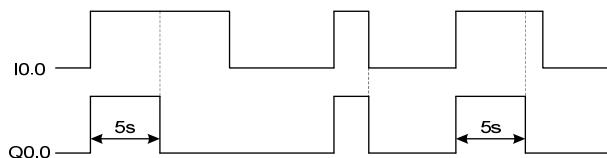
Chương trình:

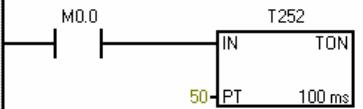
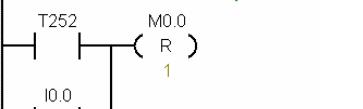


### 9.5.2 Tạo Timer xung và timer xung có nhớ

#### 9.5.2.1 Timer xung (Pulse timer)

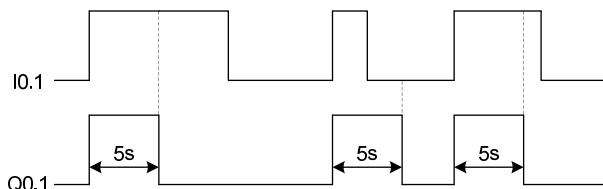
Timer xung sẽ cho ngõ ra là một xung khi tín hiệu vào ở mức logic “1” có thời gian lớn hơn hay bằng thời gian đặt ở timer xung. Để dễ hình dung xem giản đồ thời gian của chương trình tạo timer xung với ngõ ra timer là Q0.0, ngõ vào tín hiệu là I0.0, thời gian xung là 5s như sau:



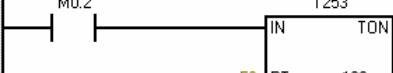
<u>LAD</u>	<u>STL</u>
<b>Network 1</b> Set Memory Bit cho Pulse Timer 	<b>Network 1</b> Set Memory Bit cho Pulse Timer LD I0.0 // load I0.0. EU // Khi co canh len S M0.0, 1 // set memory bit M0.0.
<b>Network 2</b> Khởi động Timer T252 	<b>Network 2</b> Khởi động Timer T252 LD M0.0 // Load M0.0. TON T252, 50 // khai dong timer T252 // voi PT la 50 (5s)
<b>Network 3</b> Reset Memory Bit M0.0 	<b>Network 3</b> Reset Memory Bit M0.0 LD T252 // Load timer T252. ON I0.0 // hoac neu I0.0 khong = "1" nua, R M0.0, 1 // thi reset M0.0.
<b>Network 4</b> Xuất ra Output Q0.0 	<b>Network 4</b> Xuất ra Output Q0.0 LD M0.0 // Load M0.0. = Q0.0 // set ngo ra Q0.0.

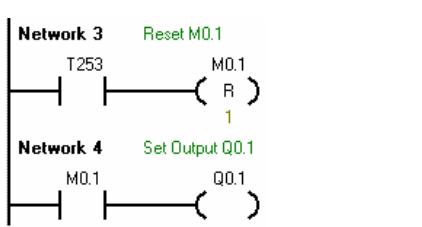
### 9.5.2.2 Timer xung có nhớ (Extended Pulse timer)

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:

<u>LAD</u>	<u>STL</u>
<b>Network 1</b> Set Memory Bit cho Extended Pulse Timer 	<b>Network 1</b> 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.
<b>Network 2</b> Khởi động Timer T253 	<b>Network 2</b> Khởi động Timer T253 LD M0.2 // Load M0.1. TON T253, +50 // thi khai dong timer T253 // voi PT=50 (5s)
<b>Network 3</b> Reset M0.1 	<b>Network 3</b> Reset M0.1 LD T253 // Load timer T253. R M0.1, 1 // reset M0.1.
<b>Network 4</b> Set Output Q0.1 	<b>Network 4</b> Set Output Q0.1 LD M0.1 // Load M0.1. = Q0.1 // set output Q0.1.



### 9.5.3 Đảo chiều quay động cơ có khống chế thời gian

#### Mô tả hoạt động

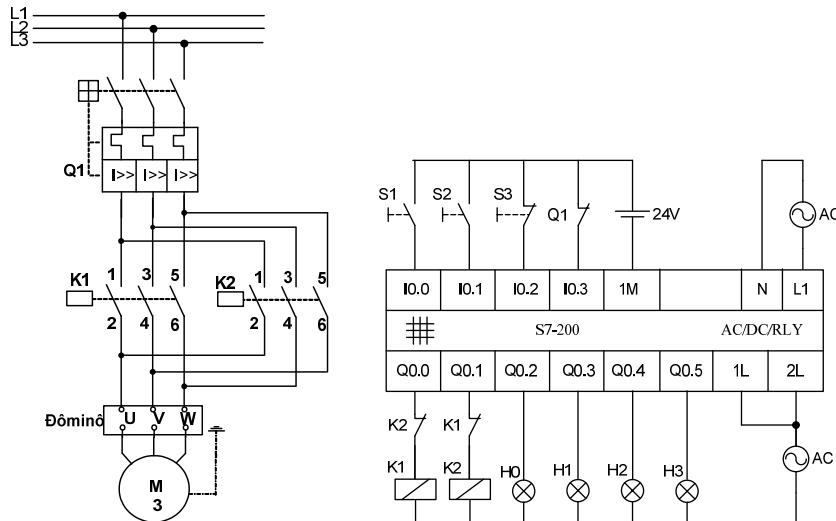
Một động cơ điện 3 pha có thể đảo chiều quay. Khi ấn nút nhấn quay phải “S1” (NO) thì động cơ quay phải, đèn “H1” sáng báo động cơ quay phải. Khi ấn nút nhấn quay trái “S2” (NO) thì động cơ quay trái, đèn “H2” sáng báo động cơ quay trái. Động cơ có thể dừng bất cứ lúc nào nếu ấn nút nhấn dừng “S3” (NC) 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)**

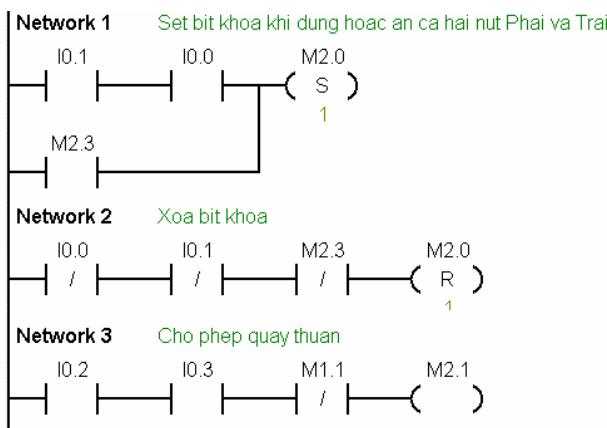
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Nút nhấn quay phải, NO
S2	I0.1	Nút nhấn quay trái, NO
S3	I0.2	Nút nhấn dừng, NC
Q1	I0.3	Tiếp điểm motor CB bảo vệ quá tải, NC
K1	Q0.0	Contactor điều khiển quay phải
K2	Q0.1	Contactor điều khiển quay trái
H0	Q0.2	Đèn báo động cơ dừng
H1	Q0.3	Đèn báo động cơ quay phải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

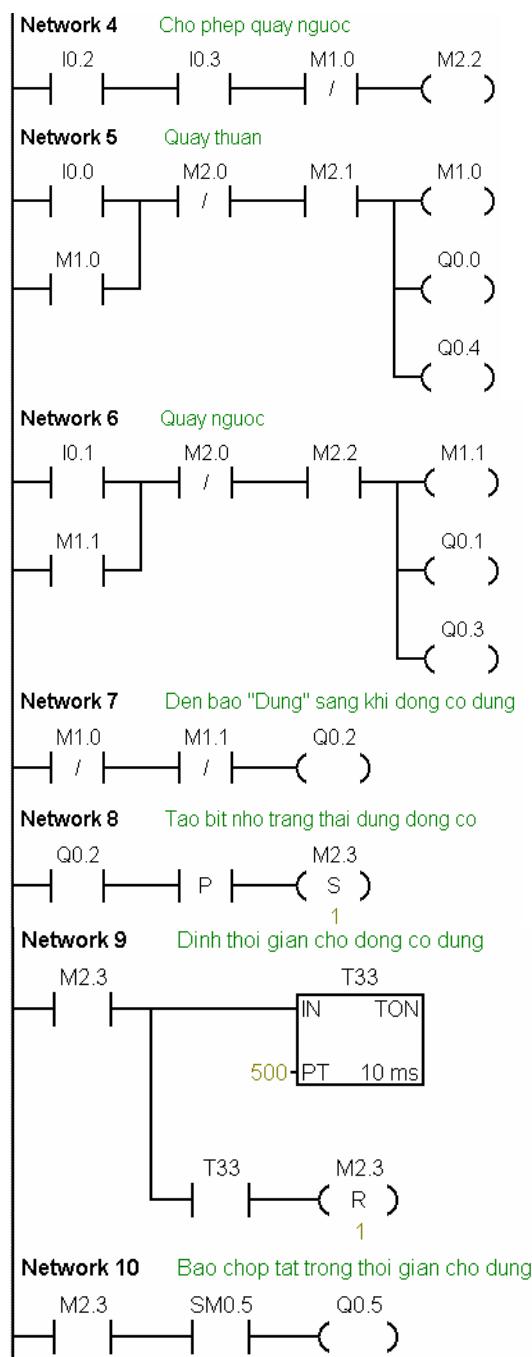
**Phân tích:**

- 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:**



### 9.5.4 Chiếu sáng Garage

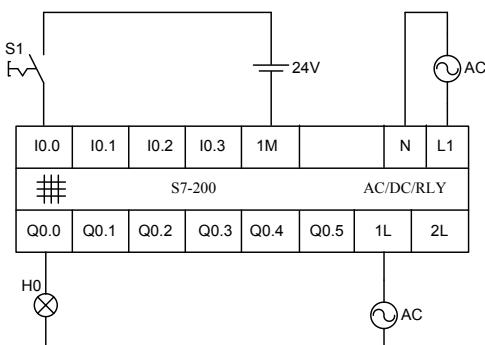
#### Mô tả hoạt động

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

#### Bảng xác định vào/ra

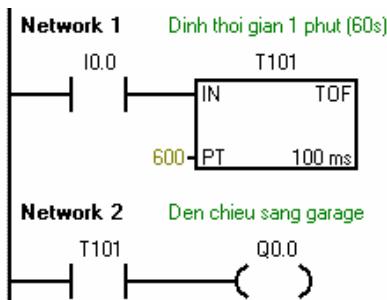
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Công tắc
H1	Q0.0	Đèn chiếu sáng Garage

#### Nối dây PLC:

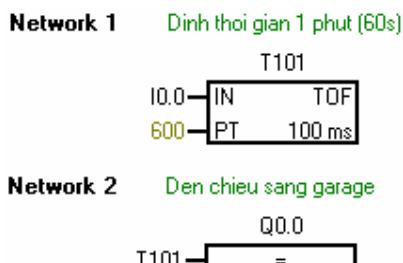


#### Chương trình

##### LAD



##### FBD



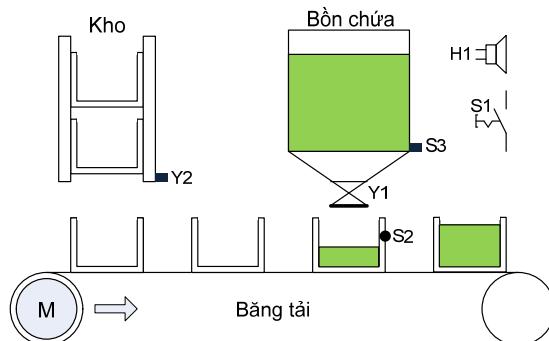
##### STL

**Network 1**      Định thời gian 1 phút (60s)  
 LD            I0.0  
 TOF          T101, 600

**Network 2**      Den chieu sang garage  
 LD            T101  
 =             Q0.0

### 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

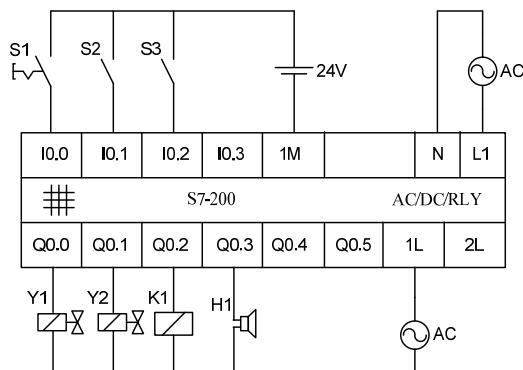
Khi bật công tắc “S1” thì thùng từ kho chứa thùng rỗng sẽ được đưa vào băng tải, và băng tải vận chuyển thùng hoạt động. Khi một thùng rỗng đến dưới bồn chứa (được nhận biết bởi cảm biến “S2”) thì băng tải dừng. Van “Y1” mở rót chất lỏng trong bồn vào thùng. Sau thời gian 5s thì thùng chứa đầy. Van “Y1” 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 rót vào băng tải chỉ cần solenoid có điện trong thời gian 100ms.

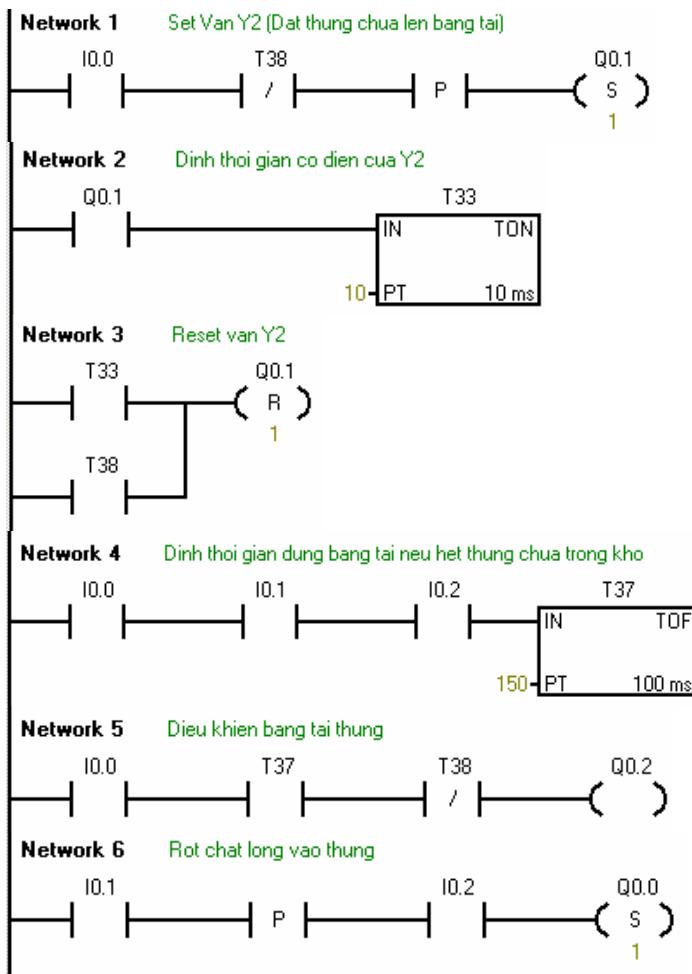
#### Bảng xác định vào/ra (Bảng ký hiệu)

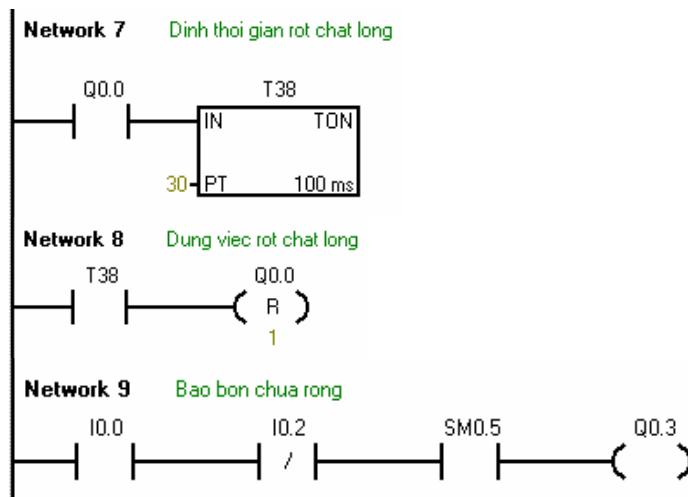
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Công tắc ON/OFF thiết bị rót
S2	I0.1	Cảm biến báo thùng đúng vị trí, (NO)
S3	I0.2	Cảm biến báo bồn rỗng, bồn rỗng =”0”
Y1	Q0.0	Van xả chất lỏng vào thùng chứa
Y2	Q0.1	Đặt thùng chứa lên băng tải
K1	Q0.2	Contactor điều khiển động cơ M kéo băng tải
H1	Q0.3	Còi báo bồn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

```

## **Network 2** Dinh thoai gian co dien cua Y2

```

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

```

**Network 3**      Reset van Y2

```

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

```

**Network 4** Dinh thoai 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 thoigian dung bang tai

```

#### **Network 5** Điều khiển bằng tay thông

```

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       // khai 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** Định thời gian rot chât long

```
LD      Q0.0      // Khi van Y1 mo
TON    T38, 30   // Định thời gian rot chât long vào thung
```

**Network 8** Dùng việc rot chât long

```
LD      T38       // Khi đủ thời gian rot
R      Q0.0, 1    // đóng van Y1
```

**Network 9** Bảo bối 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.

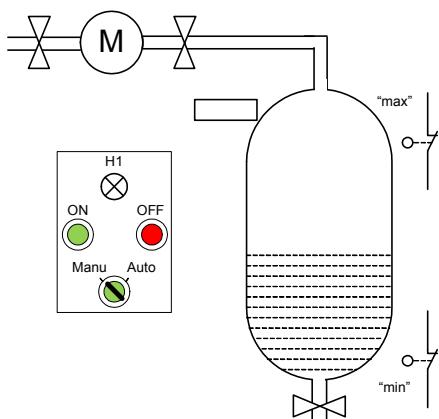
### BT9.3 Điều khiển Đèn và Quạt hút

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 mức 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:

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

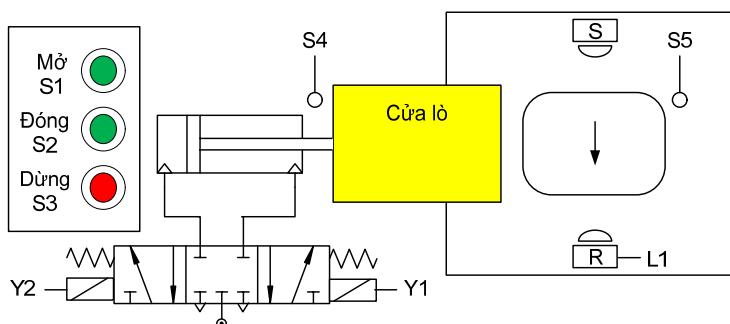
3. Vẽ sơ đồ nối dây PLC
4. Viết chương trình điều khiển và nạp vào PLC để kiểm tra.

### 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 cylinder đóng cửa
Y2	Q0.1	Van điều khiển cylinder mở cửa

Yêu cầu:

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

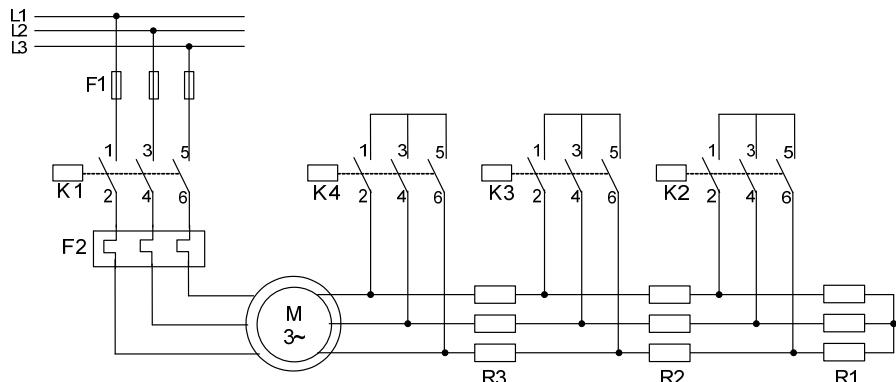
### BT9.6 Điều khiển quá trình khởi động động cơ rotor dây quấn

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.

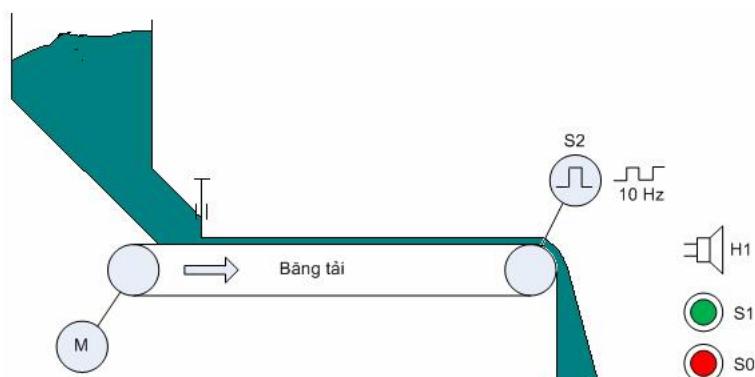
### BT9.7 Giám sát hoạt động băng tải bằng cảm biến phát xung

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.

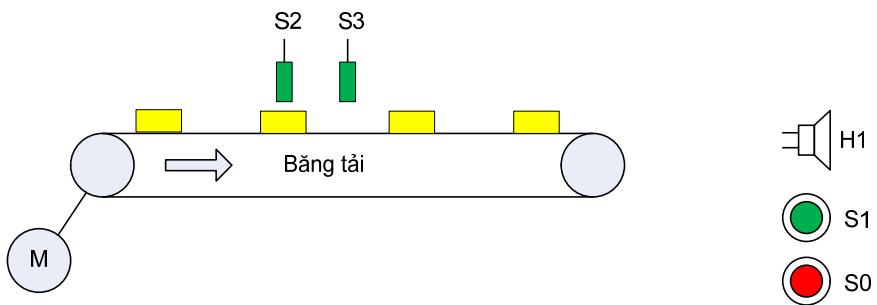
**BT9.8 Giám sát hoạt động băng tải bằng thời gian**

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.

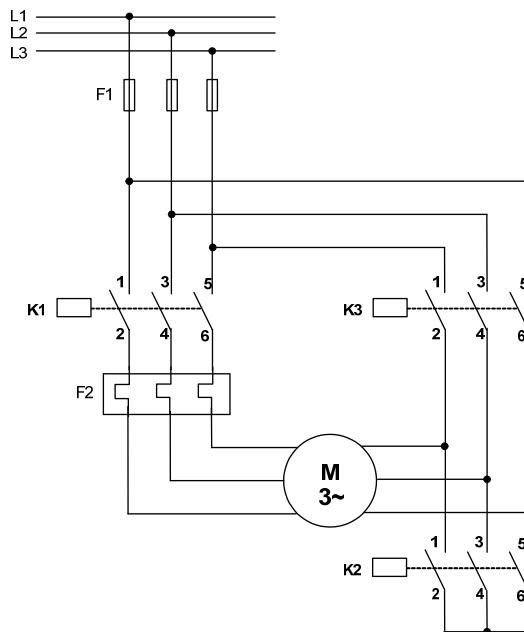
#### BT9.9 Khởi động Sao-tam giác

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:

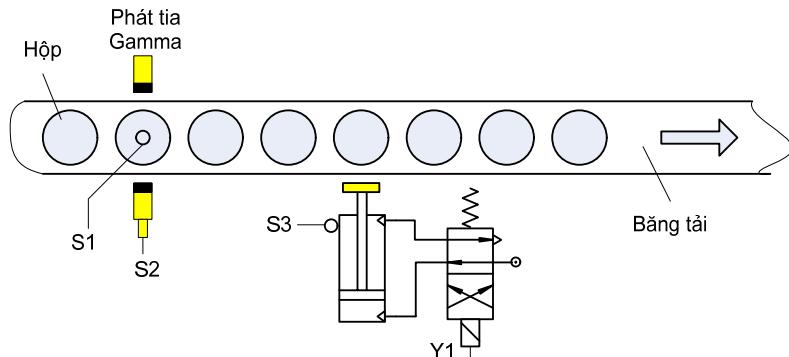
1. Vẽ sơ đồ nối dây với PLC loại AC/DC/RLY
2. Viết chương trình và nạp vào PLC để kiểm tra.

**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 đỗ 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 đỗ 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 đỗ 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 đày 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.

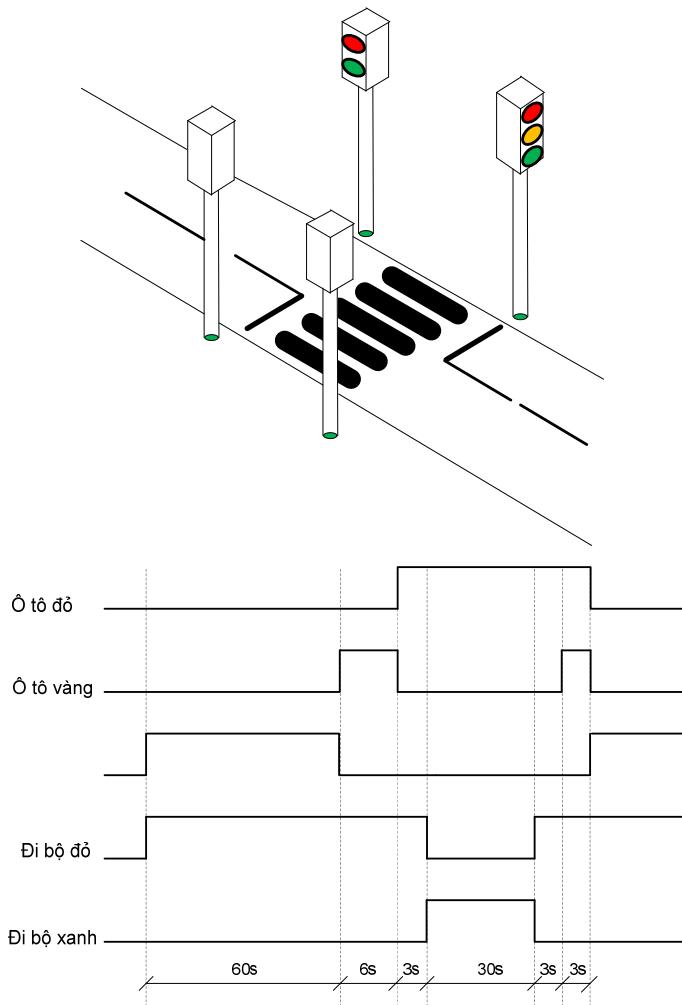
#### \* Chế độ ngày

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: Công 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:

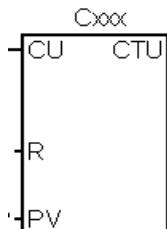
- + Bộ đếm lên CTU (Up Counter).
- + Bộ đếm xuống CTD (Down Counter).
- + Bộ đếm lên-xuống (Up/Down Counter).

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,

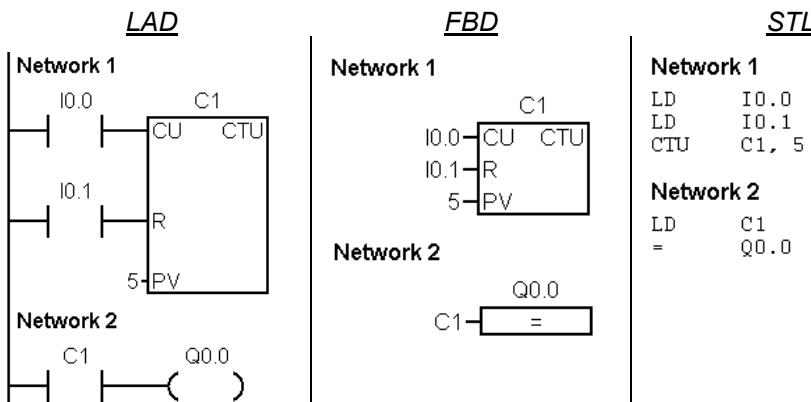
R: Xóa bộ đếm về 0. Ngõ vào bit,

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

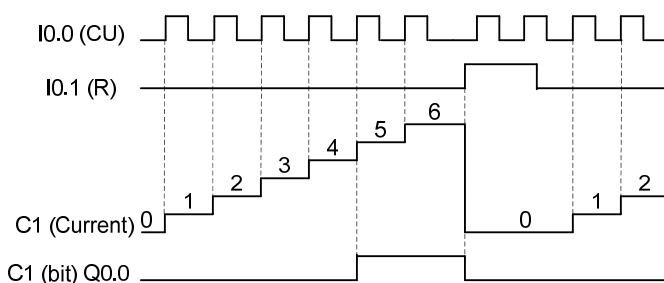
Mỗi khi tín hiệu tại CU từ mức “0” lên “1” thì bộ đếm sẽ tăng giá trị hiện hành của nó lên 1 đơn vị. Khi giá trị hiện hành của bộ đếm (Cxxx) lớn hơn hoặc bằng giá trị đặt trước tại ngõ vào PV (Preset Value) thì ngõ ra bit của counter (counter bit) sẽ lên mức “1”. Giá trị đếm lên tối đa là 32.767. Phạm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



*Giản đồ xung:*

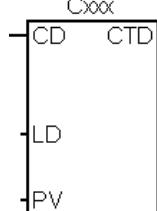


Để lấy counter CTD, trong cây lệnh bấm vào dấu (+) của biểu tượng Counters, sau đó chọn CTD, bấm và giữ chuột trái kéo thả vào vị trí mong muốn trong chương trình. Nhập các thông tin ở Cxxx, CU, R và PV.

### 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”.

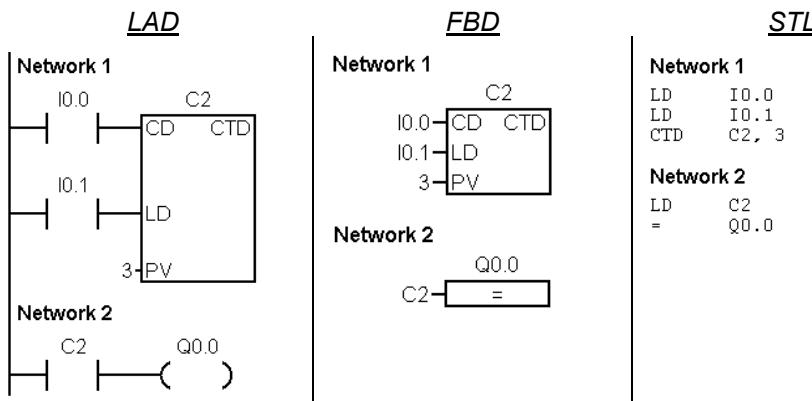
Bộ đếm sẽ dừng đếm khi giá trị hiện hành bằng 0 và counter bit Cxxx lên “1”.

Phạm vi của bộ đếm là C0 đến C255.

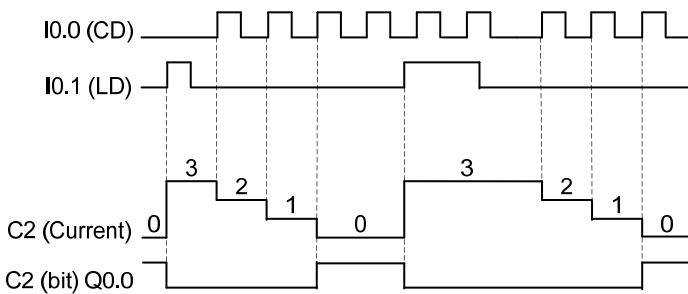
Khi xóa bộ đếm bằng lệnh Reset, counter bit bị xóa và giá trị hiện hành được đặt về 0.

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.

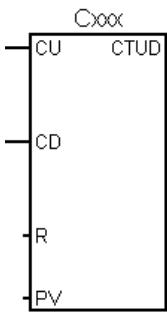


*Giản đồ xung:*



## 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 bên 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 bên kế tiếp tại ngõ vào đếm xuống bộ đếm sẽ đặt về giá trị max (32.767).

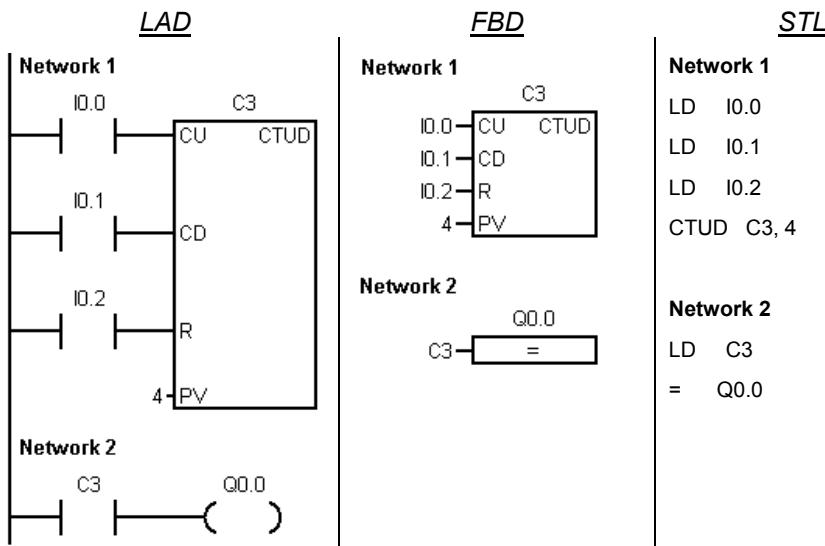
Khi giá trị hiện hành Cxxx lớn hơn hoặc bằng giá trị đặt trước PV, thì Counter Bit Cxxx lên “1”. Ngược lại Counter Bit Cxxx bằng “0”.

Phạm vi của bộ đếm là C0 đến C255.

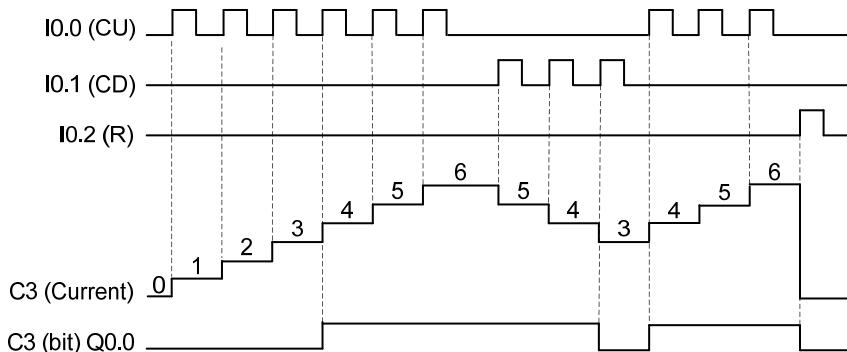
Bộ đếm sẽ bị xóa về 0 khi ngõ vào Reset (R) lên mức “1”, hoặc khi sử dụng lệnh Reset để xóa bộ đếm.

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 xuồng C3. Ngõ vào đếm lên nối với I0.0. Ngõ vào đếm xuồng nối với I0.1. Xóa bộ đếm bằng I0.2. Khi bộ đếm có giá trị hiện hành  $\geq 4$  thì ngõ ra Q0.0 lên “1”.



*Giản đồ xung:*



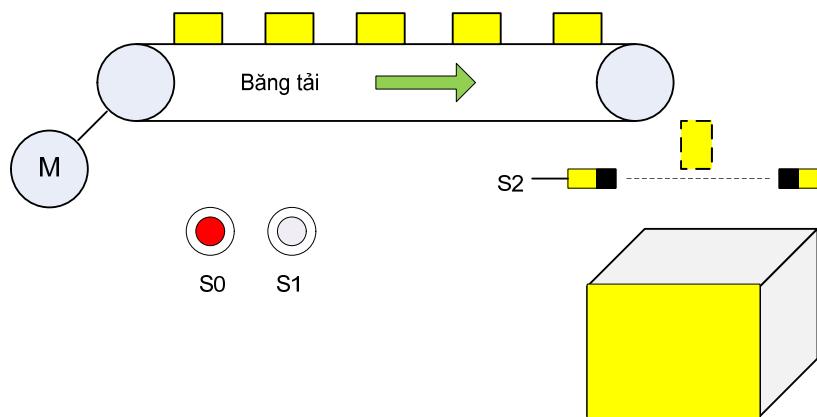
## 10.5 Ứng dụng bộ đếm

### 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 để 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).

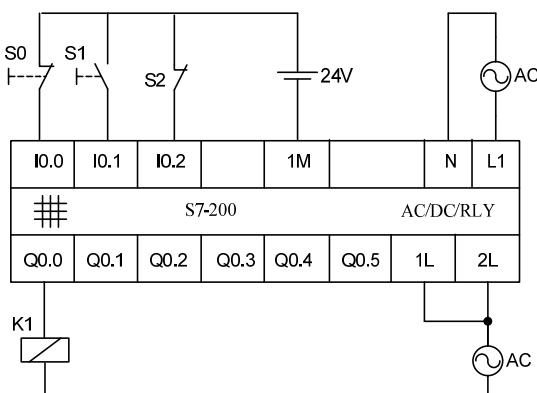
**Sơ đồ công nghệ:**



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 kiện 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.

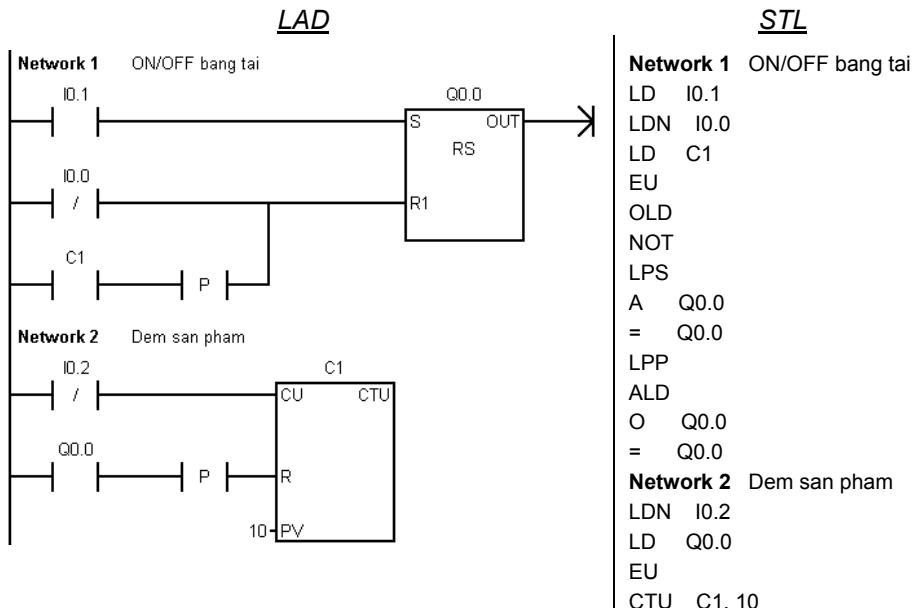
Điều kiện ngõ vào đếm lên CU: =  $\overline{S2}$

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.

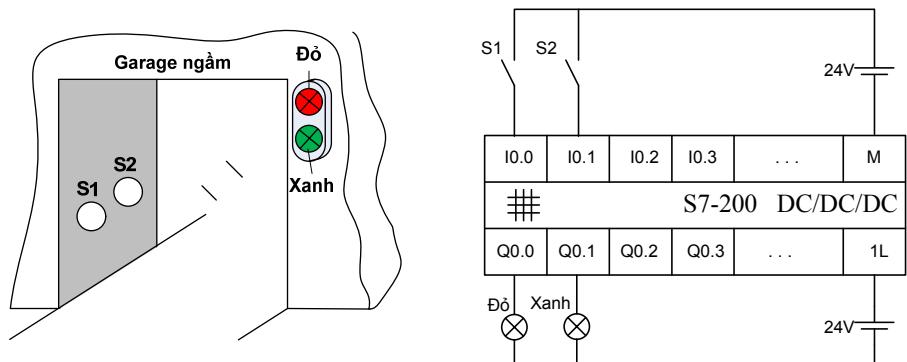
### Chương trình



### 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

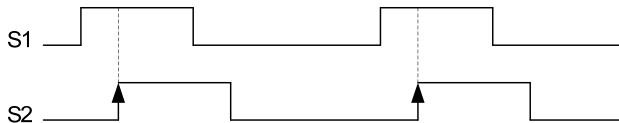
Bảng xác định vào/ra

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 xe 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 và 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

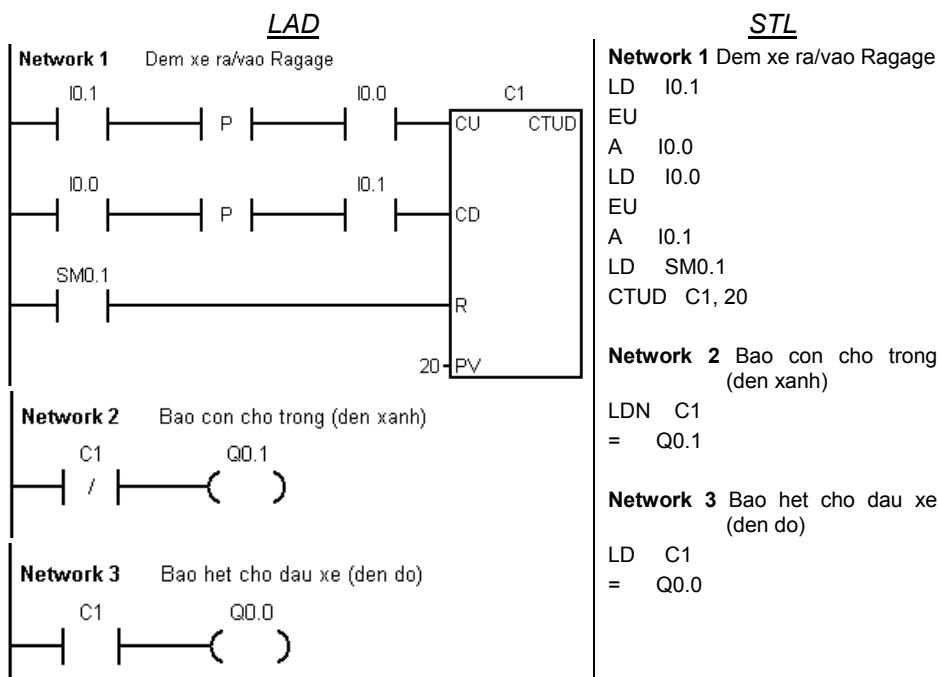
Vì số lượng xe trong Garage thay đổi khi có xe vào và ra, nên ở đây sử dụng bộ đếm lên và xuống. Ngoài ra, để đơn giản khi khởi động lại PLC thì bộ đếm xóa về 0, ta có thông tin cho các ngõ vào của bộ đếm như sau:

- 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



## 10.6 Câu hỏi và bài tập

### BT10.6.1 Điều khiển bồn sấy

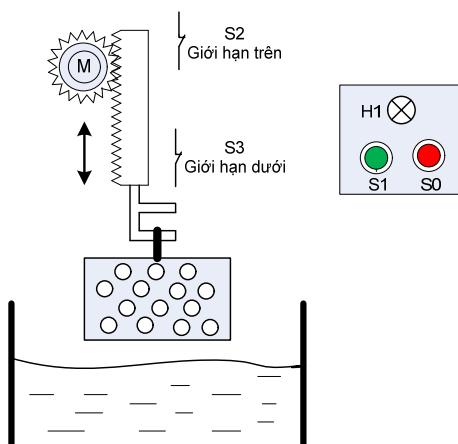
Một bồn sấy hoạt động như sau:

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. Giảm chứa các tấm mạch được treo vào một cần như hình 10.3. Khi ấn nút khởi động S1 (NO) thì cần hạ giò 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 giò 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 giò lên
K2	Q0.1	Contactor điều khiển động cơ hạ giò xuống
H1	Q0.2	Đèn báo hệ thống hoạt động

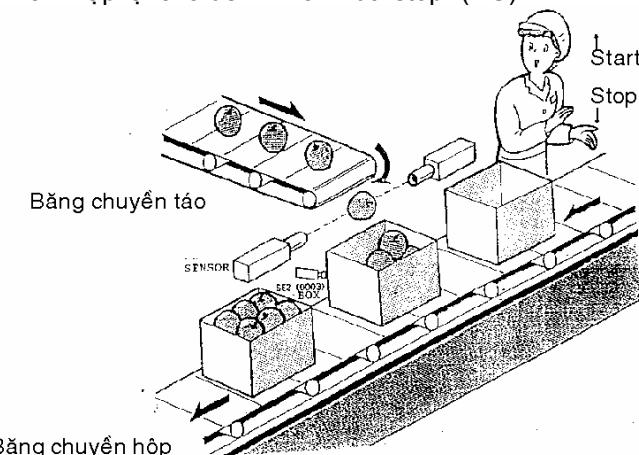
Yêu cầu:

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 đựng 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 cập 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 động**
    - *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 mở. 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 dùng 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 lẩn 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

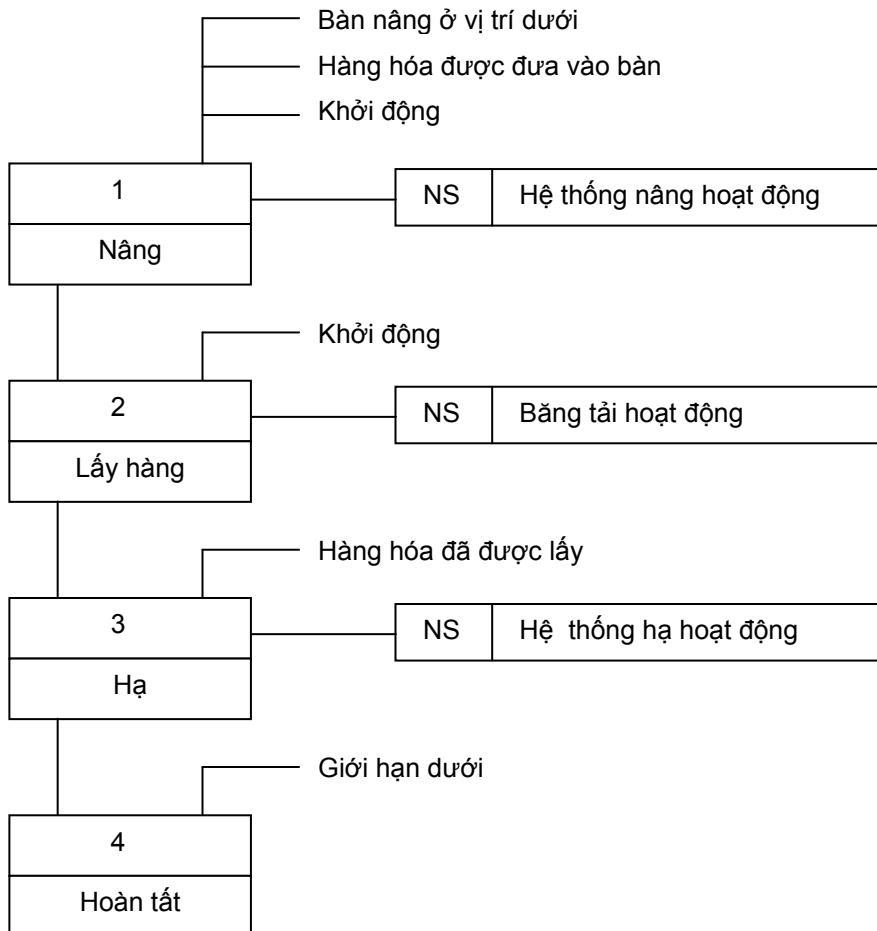
Trong công nghiệp, hầu hết các dự án điều khiển xảy ra một cách trình tự, khâu xử lý sau chậm hơn khâu xử lý trước một khoảng thời gian xác định. Ví dụ như quá trình chuyển động mới bắt đầu nếu như một quá trình khác được kết thúc.

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 nhân 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 nhòn ...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 .

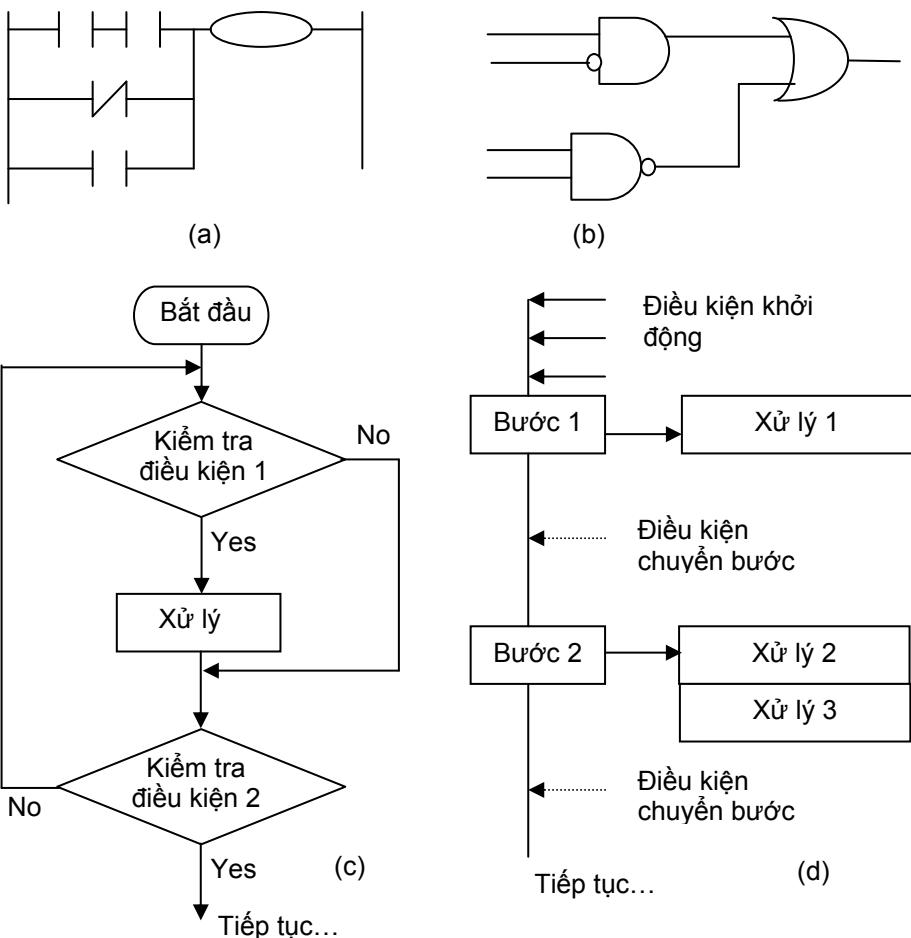
### 11.2.2 Phương pháp lập trình điều khiển trình tự

*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 cập, 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 lẩn 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 tùy 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 tùy 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) công logic; (c) lưu đồ; (d) sơ đồ chức năng

- **Phương pháp logic relay và công 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 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 đặc 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 bắt 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, nhằm lẵn 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 tương ứng 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 (function-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 .

Mỗi 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 .

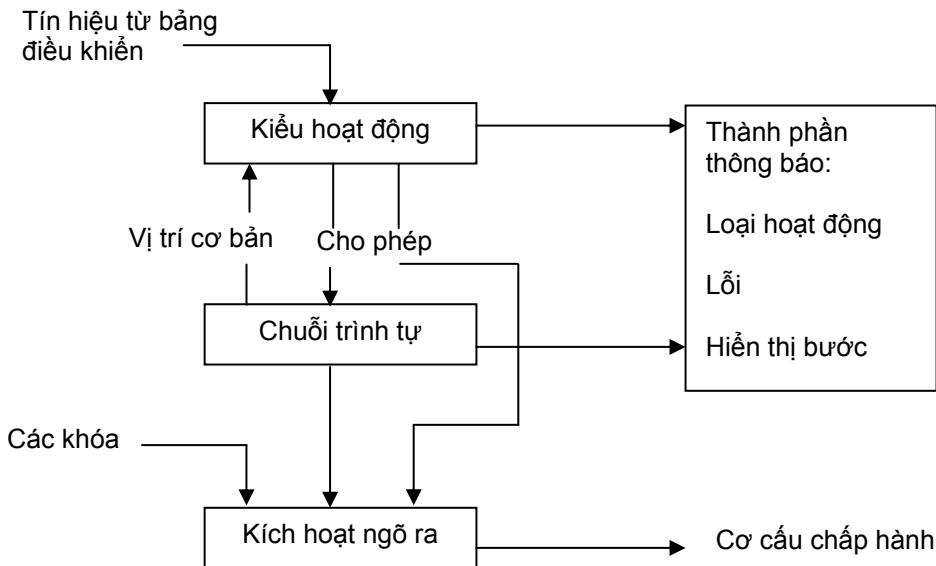
### b. Chế độ tay hay hoạt động theo bước

Trong chế độ hoạt động 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 rõ 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.

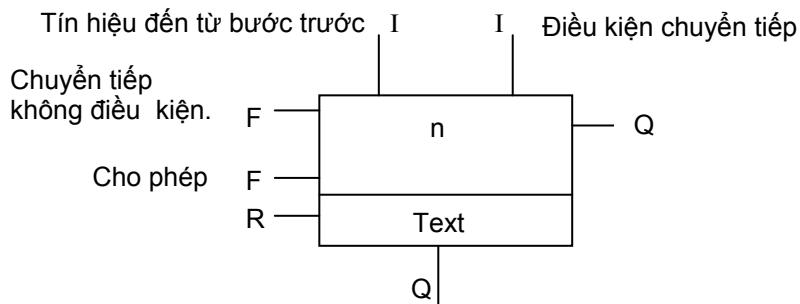
Ý nghĩa	Ký hiệu
Ký hiệu chung cho bước n : Bước thực hiện xxx: Tên bước thực hiện	
Lệnh: A : Loại lệnh. B : Tên gọi và tác dụng của các lệnh tới thiết bị được giải thích bằng chữ ( ví dụ : băng tải dừng ) C : Vị trí ngắt của lệnh.	
Đường dẫn tác dụng n : số kí hiệu của vị trí ngắt	
Tóm tắt của các đường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

Bảng 11.1: Các ký hiệu

## 11.6 Bước trình tự

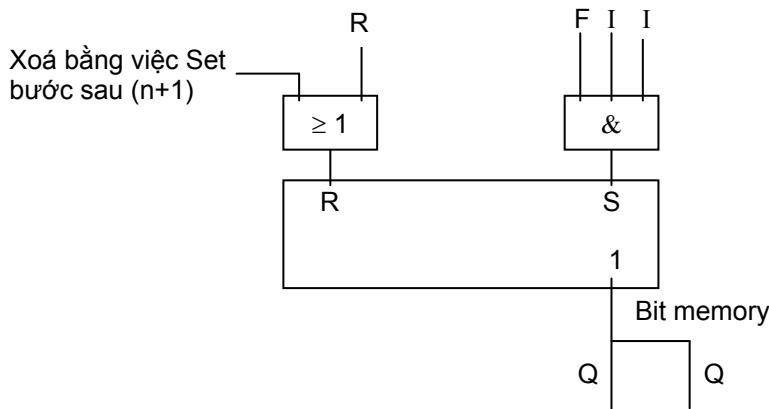
Một bước trình tự được cho như hình vẽ 11.4. Phần trên có kí hiệu “n” là số bước, phần dưới dùng để mô tả chức năng của bước. Bước “n” được

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 ”.



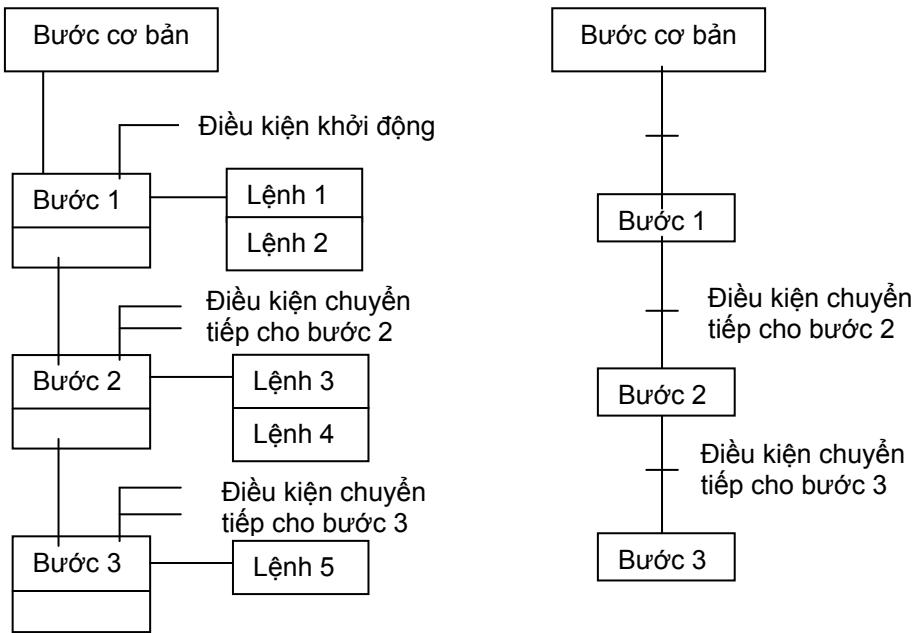
Hình 11.4: Ký hiệu của một bước với các ngõ vào và ra

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 cỗng 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 :

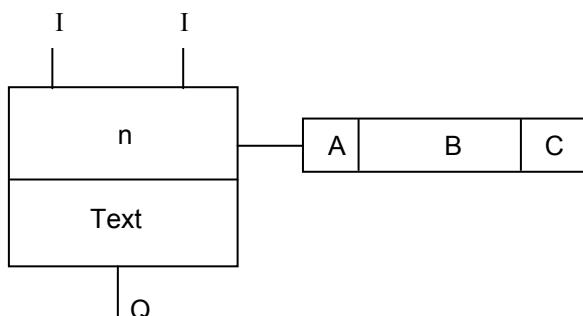


Hình 11.5: Các cách biểu diễn theo các chuẩn khác nhau

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
<p>I0.7 I0.4 M1.2 NSD Q0.5 Quạt ON, T37= 3s</p>	<p>M1.5 I0.7 I0.4 M1.2 &amp; T3s Q0.5</p>

\* **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
<p>I1.0 I0.1 M2.2 T Q1.2 Tín hiệu cảnh báo T37=10s</p>	<p>M0.6 I1.0 I0.1 M2.2 &amp; T37 &amp; Q1.2</p>

\* **Lệnh S** (duy trì)

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

Ví dụ lệnh	Biểu diễn sơ đồ logic
<p>I0.4   I1.0   F   I1.1   F   I1.2 M0.3   S   Q0.3   Motor ON</p> <p>12   I1.2   R   Q0.3   Motor STOP M1.4  </p>	<p>M1.4   I1.2   M0.3   I0.4   I1.0   I1.1 S   Q0.3   Motor ON</p> <p>R   S 1</p> <p>M2.0   &amp;</p> <p>&amp; Q0.3</p>

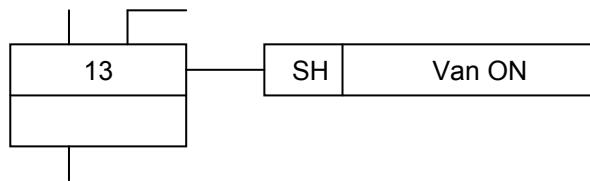
\* **Lệnh SD** (trì hoãn thời gian và được duy trì )

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
<p>I0.2   I0.7   R   I1.0   M2.1 M0.5   SD   Q0.7   Van ON. T38=5s</p> <p>12   I1.0   R   Q0.7   Van STOP M1.5  </p>	<p>M1.5   I0.7   M0.5   I0.2   I1.0   M2.1 SD   Q0.7   Van ON. T38=5s</p> <p>R   S I=0 1</p> <p>M3.0   T38   t 0</p> <p>&amp; Q0.7</p>

\* **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.

Ví dụ lệnh	Biểu diễn sơ đồ logic
 Input 5 → ST → Q1.5 Còi ON, T39= 6s	

## 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:

I1		Tự động/tay	Q4		Báo chế độ tự động				
I2		Chấp nhận chế độ							
I3		Cho phép hoạt động			Hiển thị bước				
I4		Dừng	Q3		Q2		Q1		Q0

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

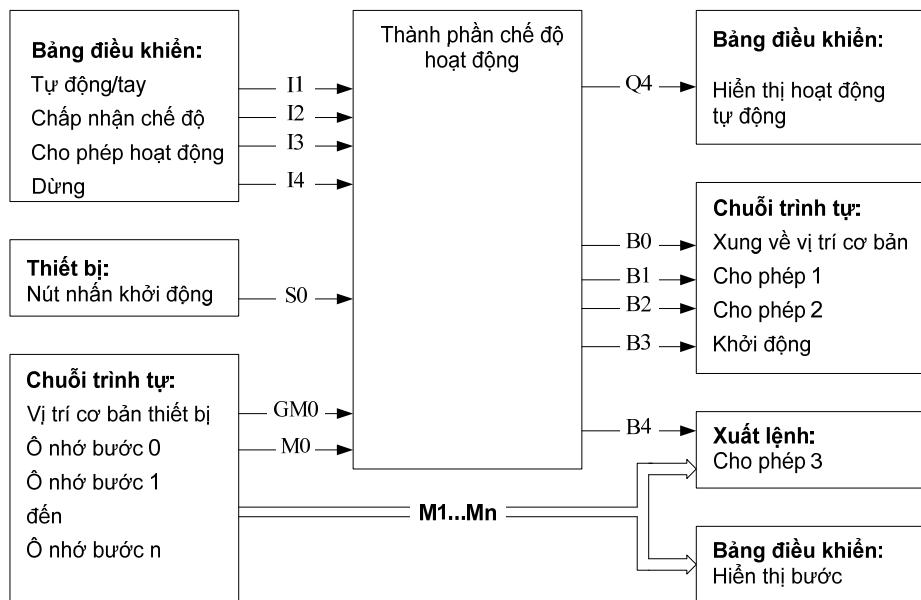
Kết thúc chế độ hoạt động tự động khi đến bước cuối cùng trong chuỗi trình tự.

### 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)

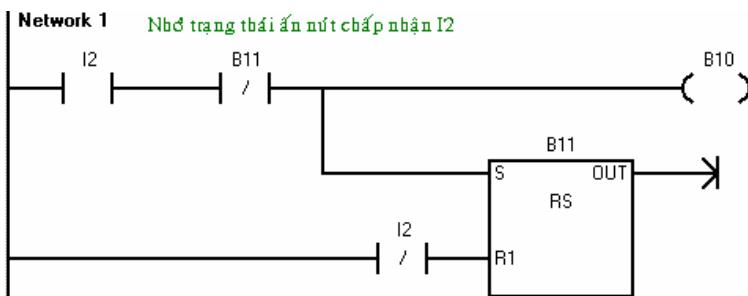
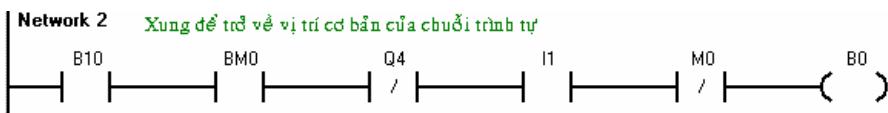
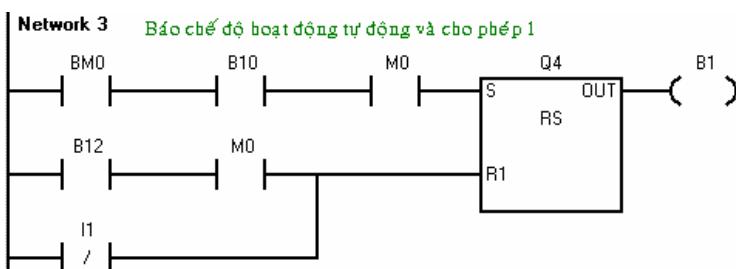
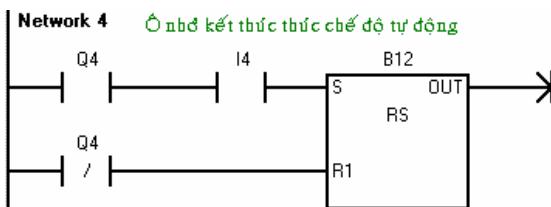
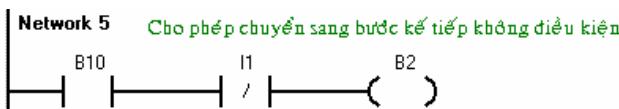
Tín hiệu cho phép 2 đối với chuyển mạch tiếp theo không điều kiện (tay)

Tín hiệu cho phép 3 đối với việc xuất lệnh

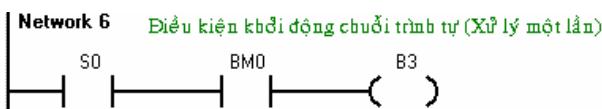
Dưới đây là các đoạn chương trình cho các khâu trong chế độ hoạt động với:

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

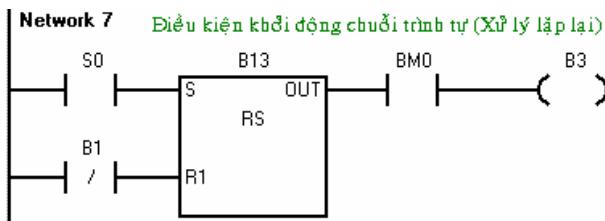
Các tín hiệu ra là Q4, B0, B1, B2, B3 và các ô nhớ phụ là B10, B11 và B12.

**Tín hiệu B0:****Tín hiệu Q4 và B1:****Tín hiệu B12:****Tín hiệu B2:** Cho phép chuyển mạch tiếp theo không điều kiện

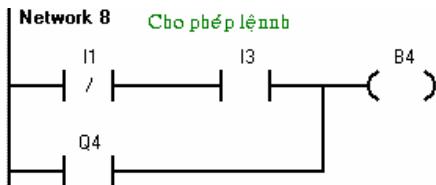
Điều kiện khởi động cho chuỗi trình tự (xử lý một lần)



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 khung 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

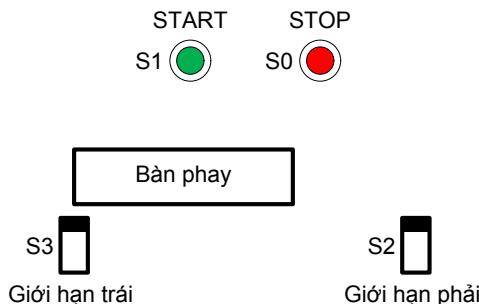
#### Mô tả hoạt động:

Khi nhấn 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 nhấn 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).

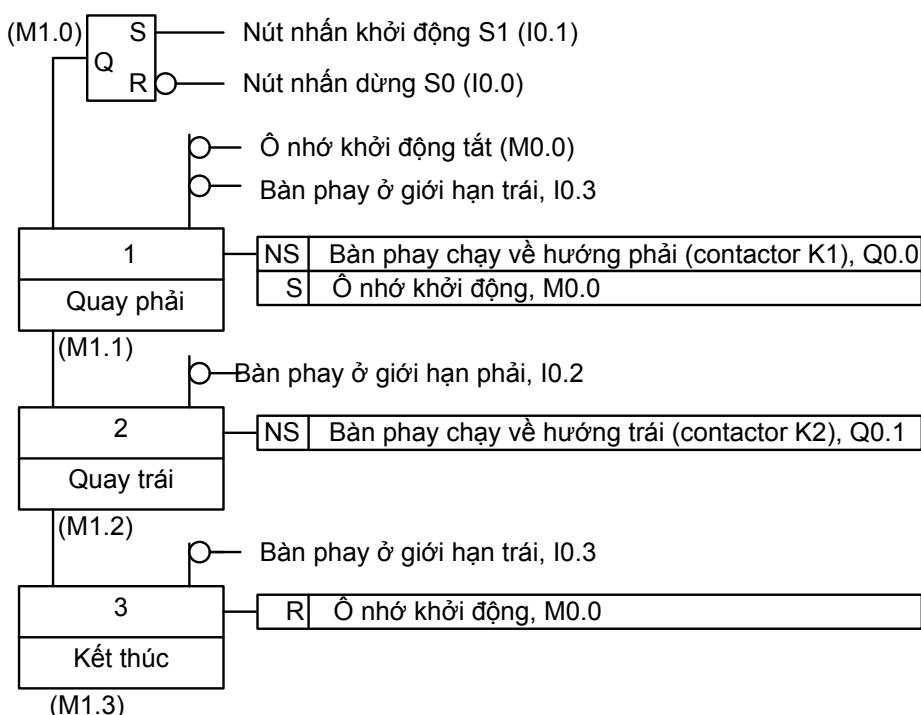
Thực hiện viết chương trình điều khiển máy phay này theo phương pháp trình tự.

### 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:

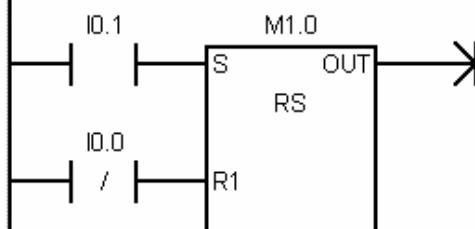
Ký hiệu	Địa chỉ	Chú thích
<b>Các biến vào</b>		
S0	I0.0	Nút nhấn dừng, NC

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
<b>Các biến ra</b>		
K1	Q0.0	Contactor điều khiển bàn phay chạy về hướng phải
K2	Q0.1	Contactor điều khiển bàn phay chạy về hướng trái

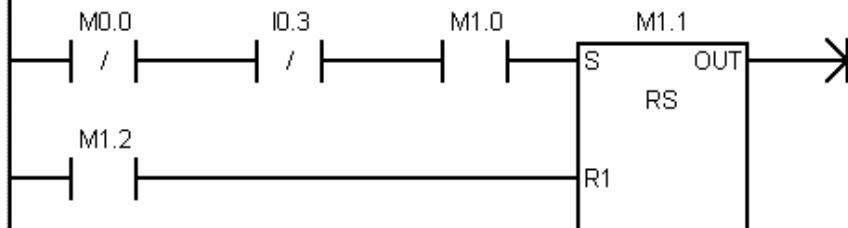
Chương trình

Biểu diễn ở LAD:

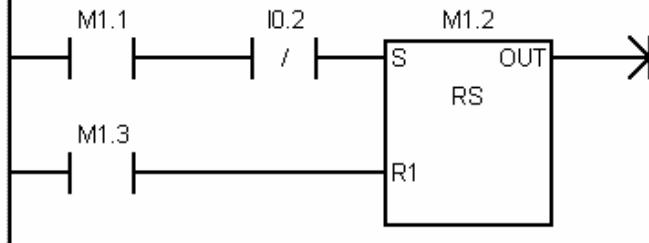
**Network 1** Bước cơ bản

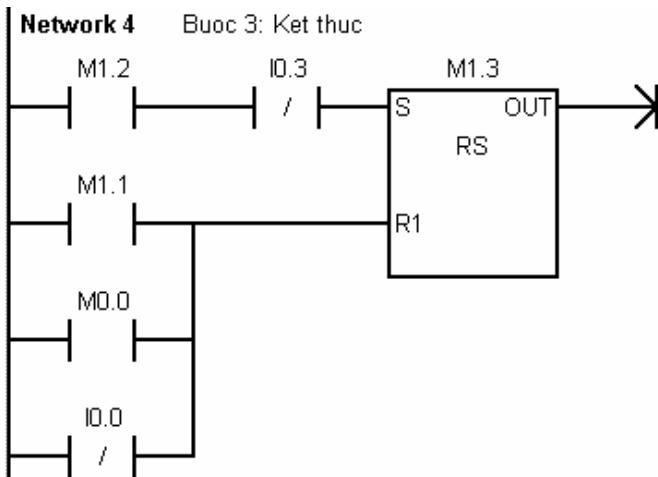


**Network 2** Bước 1: Quay phải

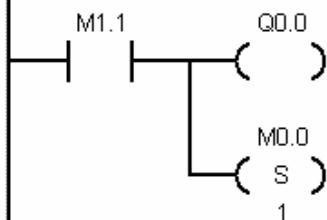


**Network 3** Bước 2: Quay trái





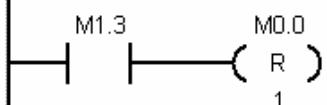
**Network 5** Thực hiện nhiệm vụ trong bước 1: Contactor K1.



**Network 6** Thực hiện nhiệm vụ trong bước 2: contactor K2



#### **Network 7**    Thuc hien nhanh yu trong buoc 3



Biểu diễn ở STI :

#### **Network 1** Bước cơ bản

# Network I

LD  
LDN

LDN  
NOT

NOI  
LBS

LPS

A

=

LPP  
ALB

ALD

M1.0

### **Network 2** Bước 1: Quay phai

Network 2

AN 10.3

A M1.0

A  
1D

ED  
NOT

NOT  
IPS

A M1.1

=

LPP

ALD

O M1.1

204

**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 nhanh vu trong buoc 1: Contactor K1

```

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

```

**Network 6** Thuc hien nhanh vu trong buoc 2: contactor K2

```

LD   M1.2
=   Q0.1

```

**Network 7** Thuc hien nhanh vu trong buoc 3

```

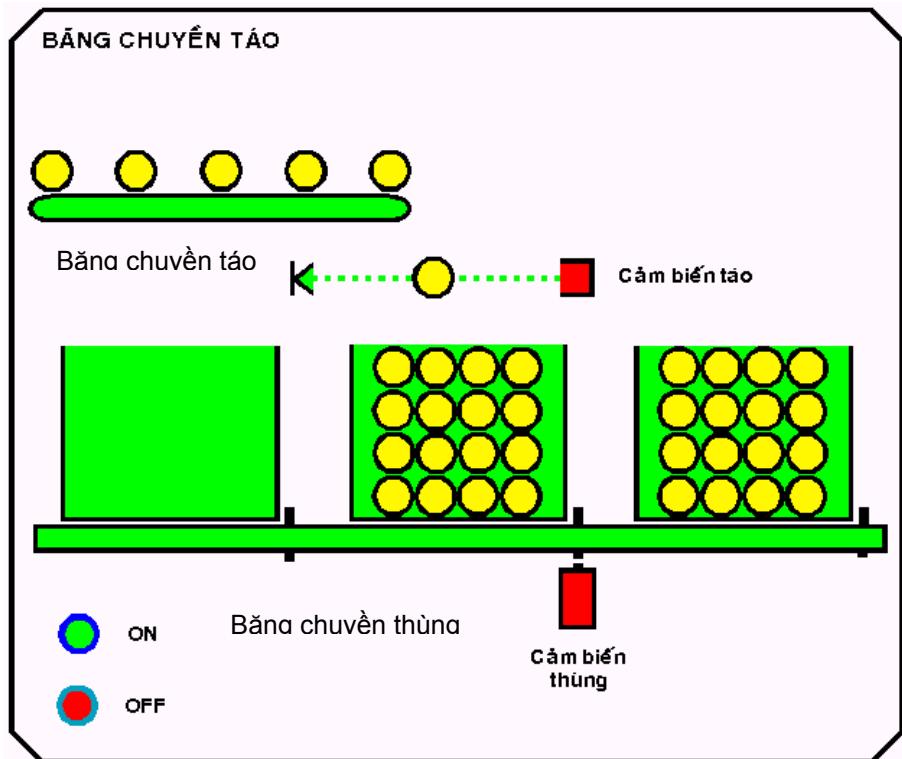
LD   M1.3
R   M0.0, 1

```

**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 băng 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:**

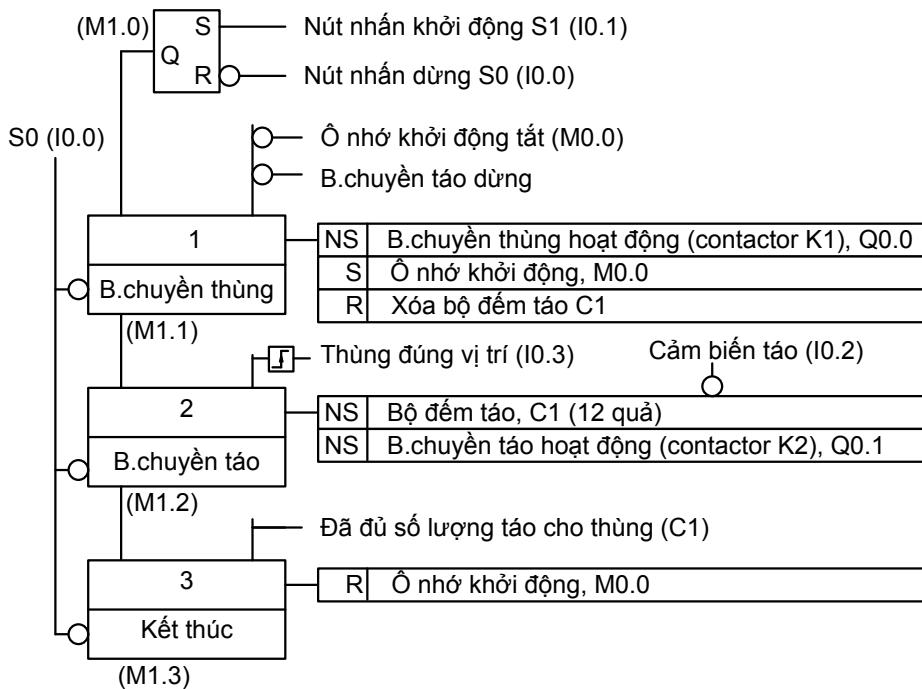


Hình 11.10: Sơ đồ công nghệ băng chuyền đếm táo

Bảng ký hiệu:

Ký hiệu	Địa chỉ	Chú thích
<b>Các biến vào</b>		
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
<b>Các biến ra</b>		
K1	Q0.0	Contactor điều khiển băng chuyền táo
K2	Q0.1	Contactor điều khiển băng chuyền thùng

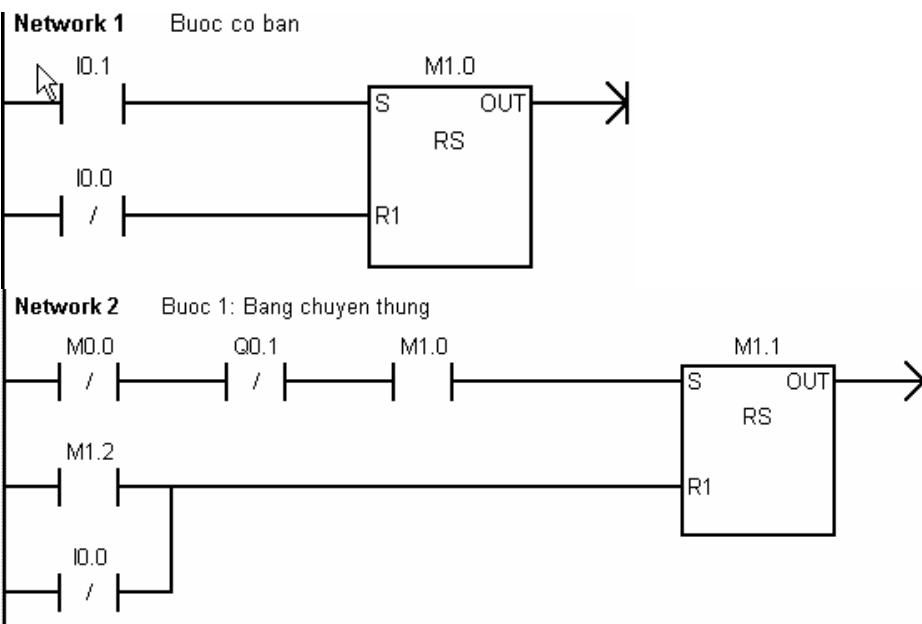
**Sơ đồ điều khiển theo trình tự:**

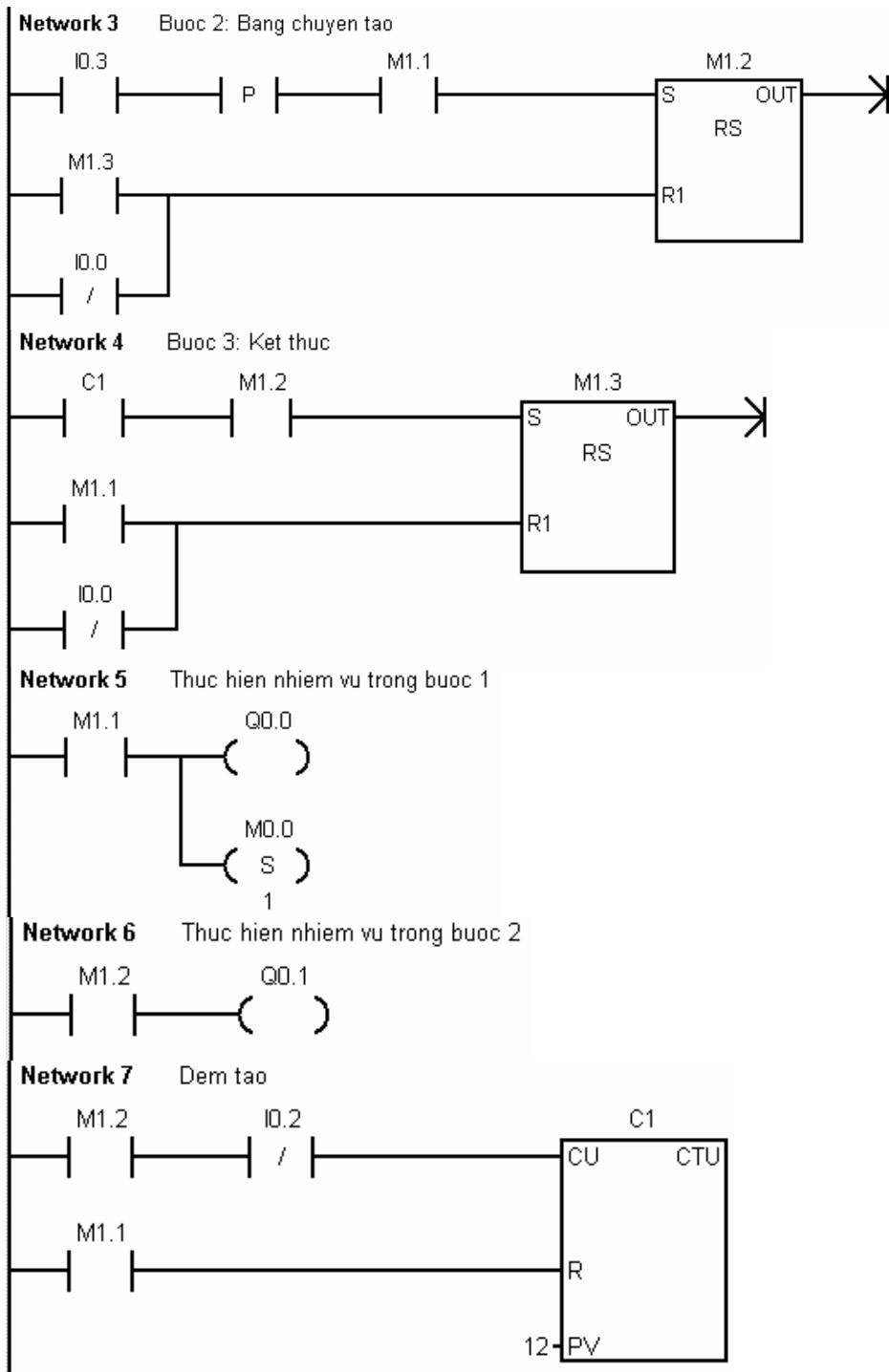


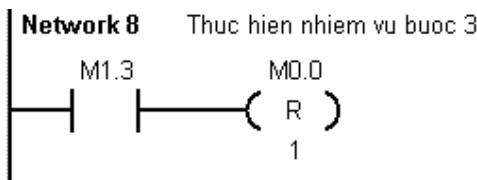
Hình 11.11: Sơ đồ điều khiển theo trình tự bằng chuyển đếm táo

Chương trình

Biểu diễn ở LAD:





**Chương trình biểu diễn ở STL:****Network 1** Bước cơ bản

```

LD I0.1
LDN I0.0
NOT
LPS
A M1.0
= M1.0
LPP
ALD
O M1.0
= M1.0

```

**Network 2** Bước 1: Bang chuyển 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** Bước 2: Bang chuyển tạo

```

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** Bước 3: Kết thúc

```

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** Thực hiện nhiệm vụ trong bước 1

```

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

```

**Network 6** Thực hiện nhiệm vụ trong bước 2

```

LD M1.2
= Q0.1

```

**Network 7** Dem tạo

```

LD M1.2
AN I0.2
LD M1.1
CTU C1, 12

```

**Network 8** Thực hiện nhiệm vụ bước 3

```

LD M1.3
R M0.0, 1

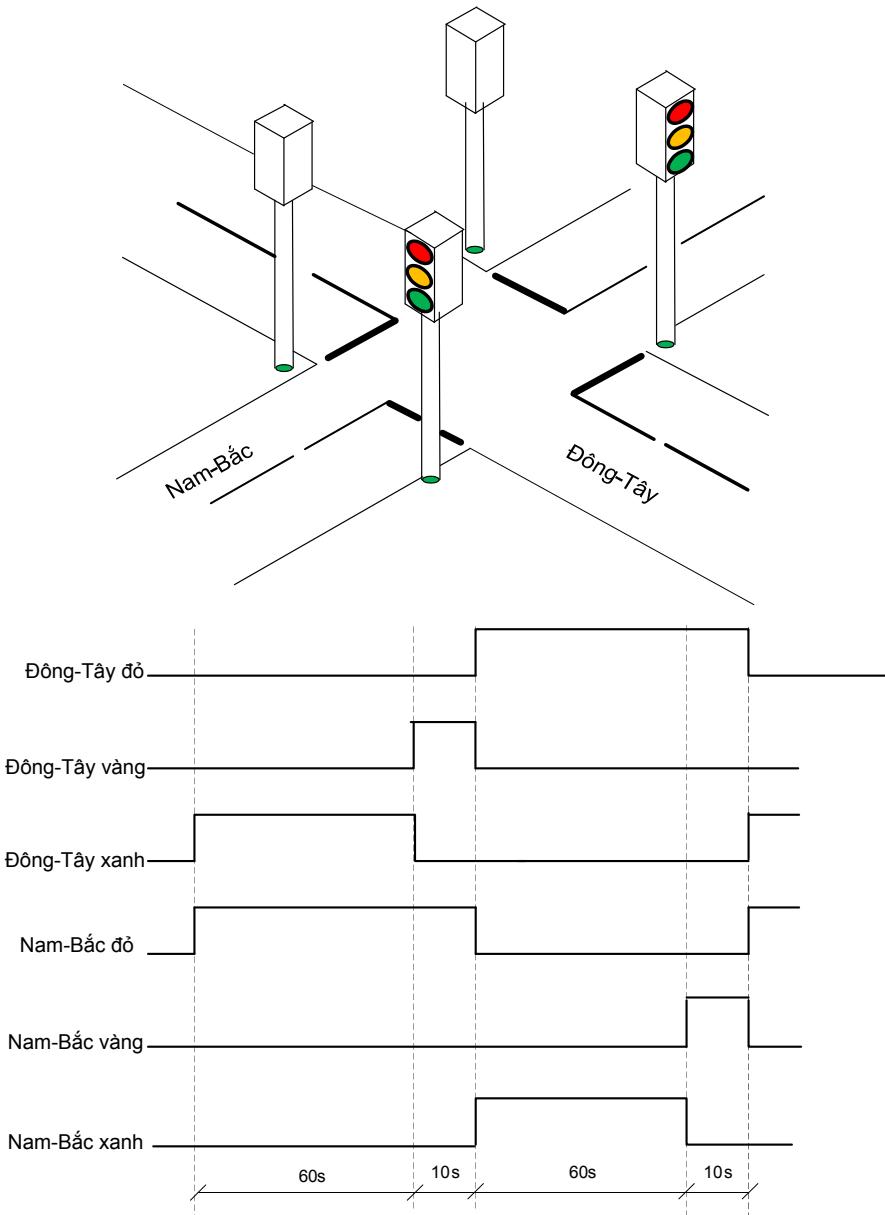
```

## 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



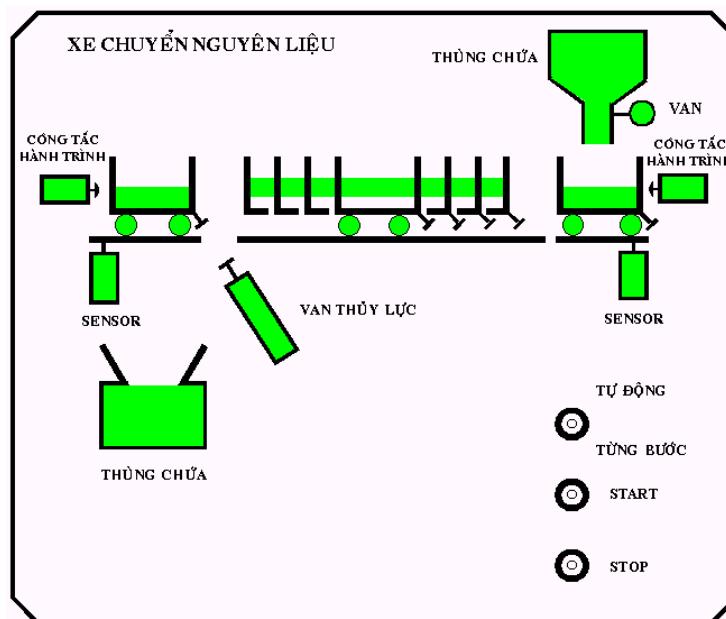
Hình 11.12: Sơ đồ công nghệ đèn giao thông và giản đồ thời gian

**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 chuyên nguyên liệu**

Hình 11.13: Sơ đồ công nghệ xe chuyên nguyên liệu

**Bảng ký hiệu**

Ký hiệu	Địa chỉ	Chú thích
Start	I0.0	Khởi động hệ thống, thường mở.
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 mở.
Stop	I0.5	Dừng, thường đóng.
Step	I0.6	Chế độ bước, thường mở.
Auto	I0.7	Chế độ tự động, thường mở.
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.

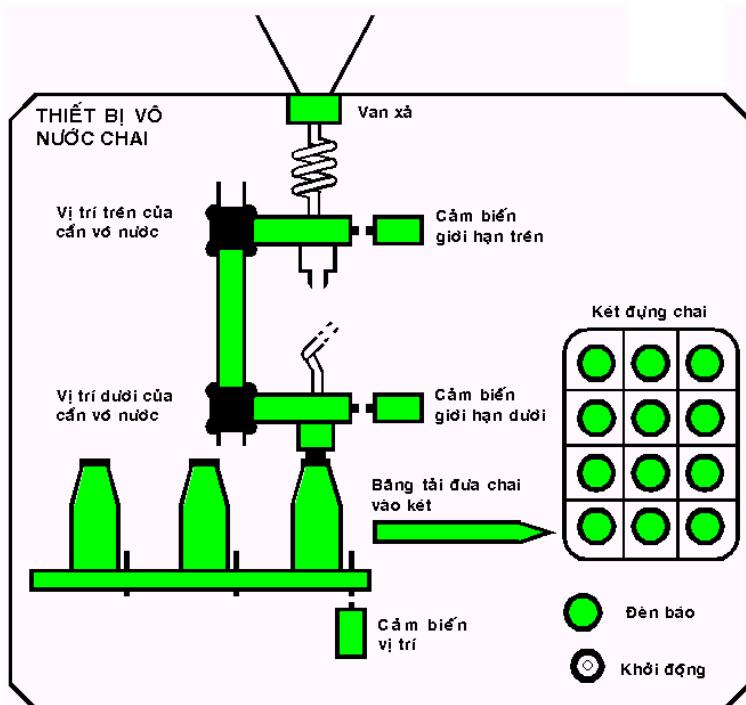
### Chế độ bước:

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 vò nước chai



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

#### Bảng ký hiệu

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

#### Mô tả

Thiết bị vò nước chai hoạt động như sau:

Trước khi vận hành thiết bị vò 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 vò 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 để 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 vò 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.

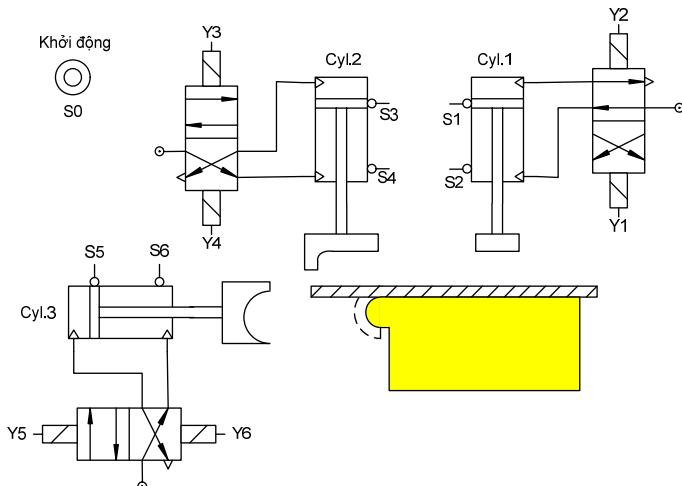
Chai đã đầy nước được đưa sang băng tải đưa chai vào két khi băng tải chai rỗng hoạt động, khi chai đúng vị trí trong két thì có một tín hiệu phát ra (I0.4).

Quá trình được lặp đi lặp 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 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
- Án nút khởi động S0 thì xy lanh Cyl.1 hạ xuống để giữ lấy thanh kim loại.

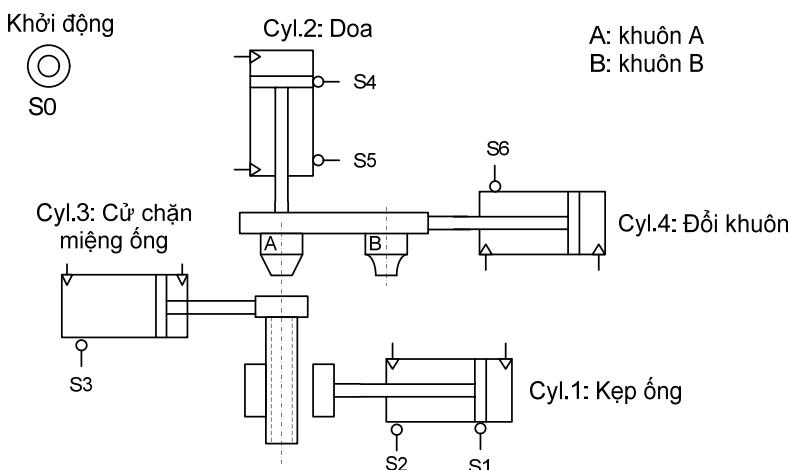
- Khi thanh kim loại được giữ chặt (nhận biết bởi công tắc hành trình S2) thì xy lanh Cyl.2 hạ xuống để uốn thanh kim loại vuông góc trước. Sau khi uốn xong thì tự động nâng lên nhờ công tắc hành trình S4.
- Khi xy lanh Cyl.2 trở về vị trí cơ bản (nhận biết bởi S3) thì xy lanh Cyl.3 được đẩy để uốn thanh 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 miệng 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 khoảng 3s. Sau đó xy lanh Cyl.2 rút về và khuôn B được 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 kẽ 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.

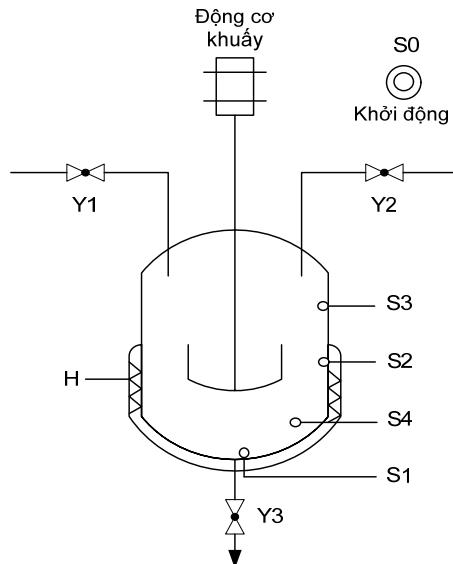
### **BT 11.6 Bồn trộn**

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:

I1 <input type="radio"/>	Tự động/tay	Q4 <input checked="" type="checkbox"/>	Báo chế độ tự động		
I2 <input type="radio"/>	Chấp nhận chế độ				
I3 <input type="radio"/>	Cho phép hoạt động		Hiển thị bước		
I4 <input type="radio"/>	Dừng	<input checked="" type="checkbox"/> Q3	<input checked="" type="checkbox"/> Q2	<input checked="" type="checkbox"/> Q1	<input checked="" type="checkbox"/> Q0

Bảng ký hiệu:

Ký hiệu	Địa chỉ	Chú thích
<b>Các biến vào</b>		
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 mục chất lỏng 1 (bồn rỗng)
S2	I0.2	Công tắc hành trình báo mục chất lỏng 2
S3	I0.3	Công tắc hành trình báo mục chất lỏng 3
S4	I0.4	Cảm biến nhiệt độ
<b>Các biến ra</b>		
Q0	Q0.6	Chỉ thị bước 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

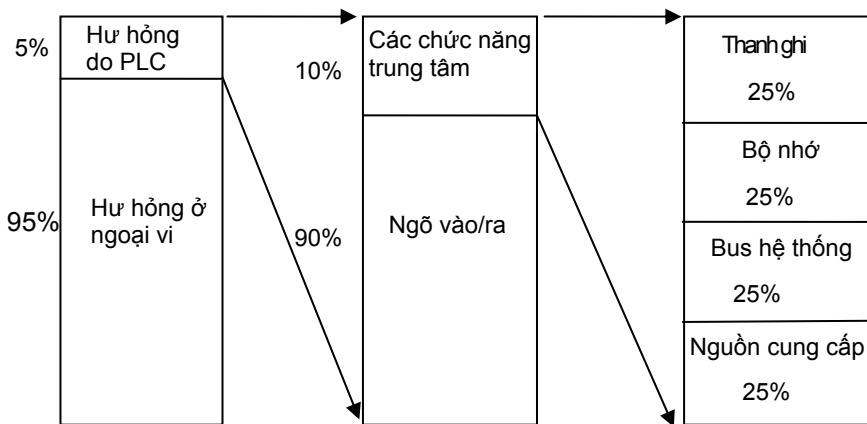
### 12.2 Hư hỏng ở PLC

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 khói vào/rơ, 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 thỏa 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:

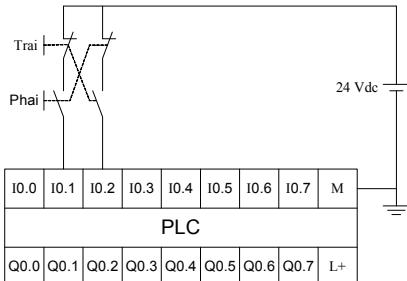
#### \* Khóa 2 ngõ vào (hình 12.10)

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.

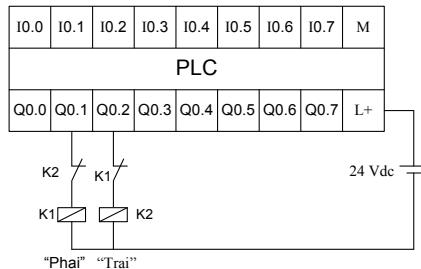
### \* Khóa ngõ ra (hình 12.11)

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 khoá được chỉ định bắt buộc, vì hiện tượng dính tiếp điểm của contactor và lỗi lập trình gây ra.

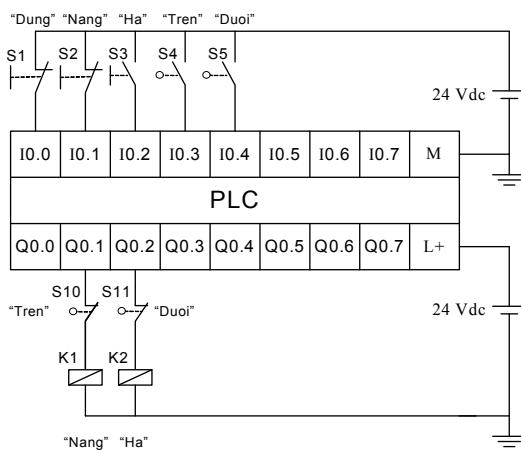


Hình 12.14: Khóa 2 ngõ vào



Hình 12.15: Khóa 2 ngõ ra

### \* Khóa do nhấn 2 tay cùng lúc



Hình 12.16: Sử dụng công tắc giới hạn an toàn

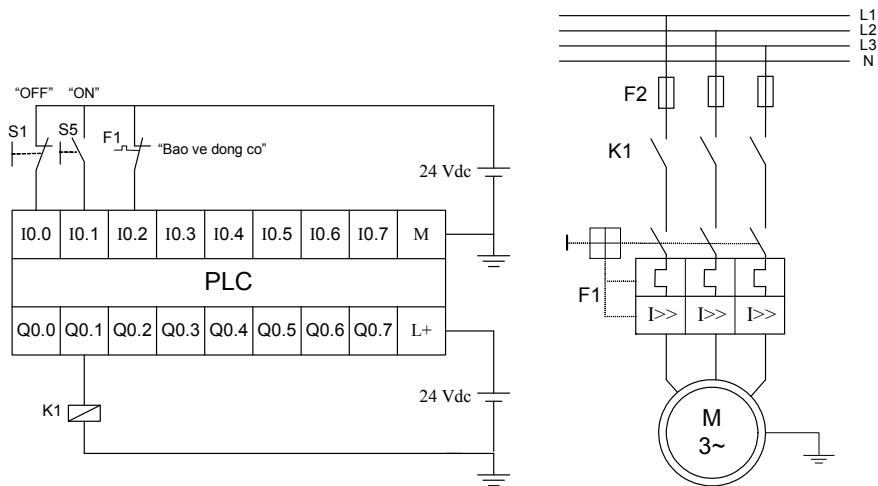
#### - Công tắc bảo vệ động cơ

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.

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.



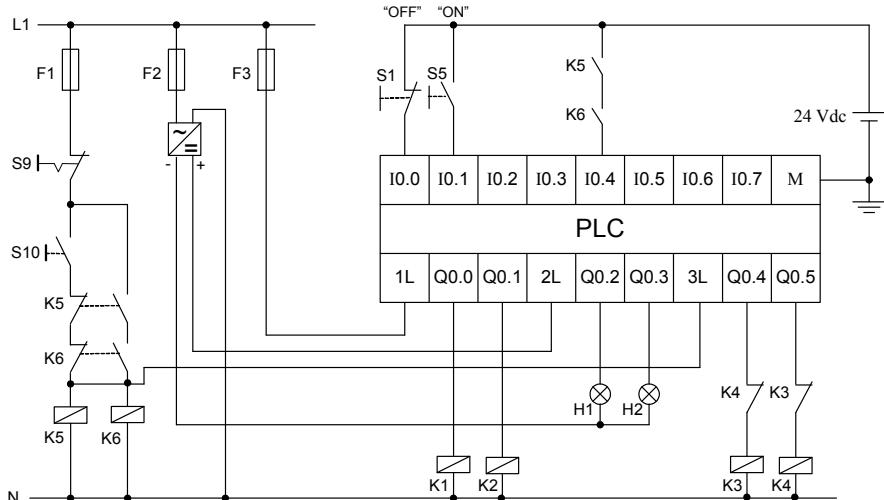
Hình 12.17: Sử dụng công tắc bảo vệ động cơ trong hệ thống điều khiển bằng PLC

- **Công tắc dừng khẩn cấp**

Công tắc dừng khẩn 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 án. 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.

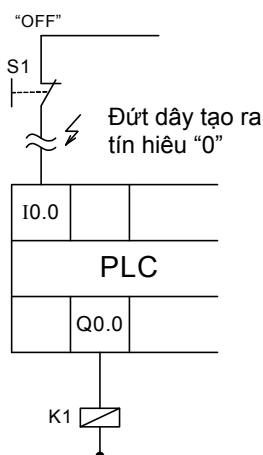
- Lập trình an toàn khi đứt dây**

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?

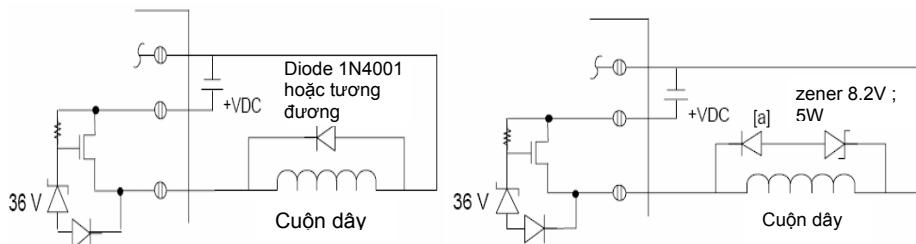
Với sự xem xét có tính nguyên tắc này cho phép thiết bị điều khiển từ chương trình thực hiện an toàn ở các bước tiếp theo. Nếu các yêu cầu an toàn được đặt cao hơn, thì lỗi nguy hiểm phải được nhận biết thông qua các biện pháp phụ và ngăn cản các tác dụng của nó.

## 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.

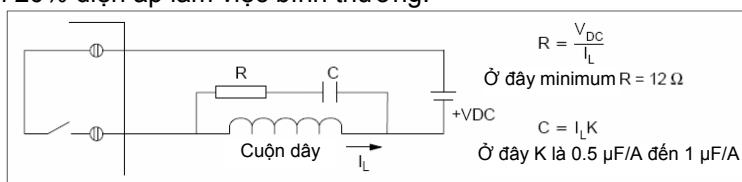


#### 12.4.2 Bảo vệ ngõ ra Röle có nguồn điều khiển DC

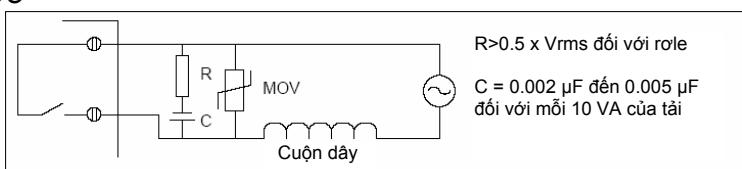
Trong trường hợp này người ta thường sử dụng mạng điện trở/tụ điện và điện áp điều khiển có thể đến 30VDC.

#### 12.4.3 Bảo vệ ngõ ra Röle và ngõ ra AC có nguồn điều khiển AC

Khi sử dụng röle 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 dùng 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 cài 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 đổi 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 nói 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 sửa 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 cắt 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.

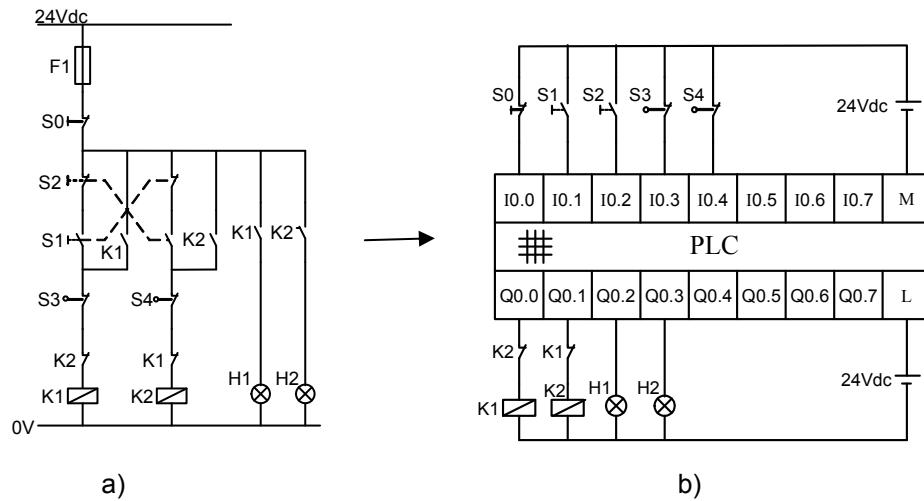
Hình 13.1 là một ví dụ về việc chuyển đổi một sơ đồ điều khiển cửa ra vào cơ quan bằng contactor thành hệ thống điều khiển với PLC (chỉ dừng lại ở việc chuyển đổi kết nối dây, còn chương trình thực hiện ở các chương sau).

Dựa vào các bước trên, ta nhận thấy các nút nhấn, contactor cần thiết được 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 mở
S2	I0.2	Nút nhấn đóng cửa, thường mở
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



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.

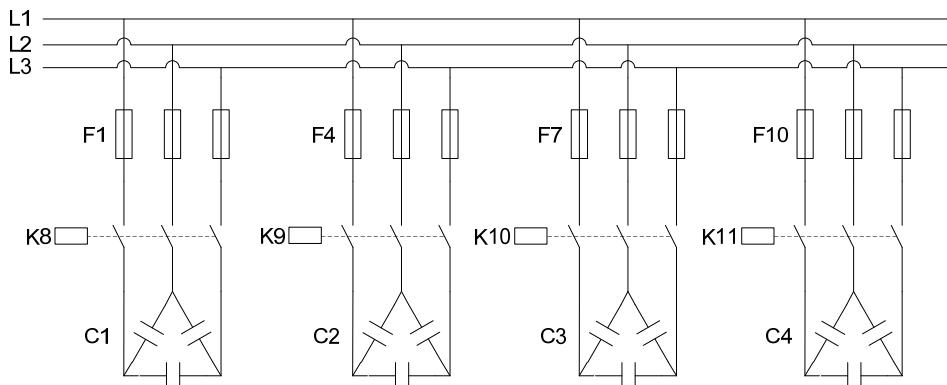
Việc thực hiện chuyển đổi từ điều khiển bằng contactor thành PLC có thể xem chương 4 (kết nối dây PLC với ngoại vi). Ngoài ra cần chú ý thêm một số điểm sau:

- 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:
  - o Các điều kiện làm cho đối tượng điều khiển ở mức tích cực (logic "1") được sử dụng với lệnh S.
  - o Các điều kiện làm cho đối tượng điều khiển ở mức không tích cực (logic "0") được sử dụng với lệnh R.
  - o 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.

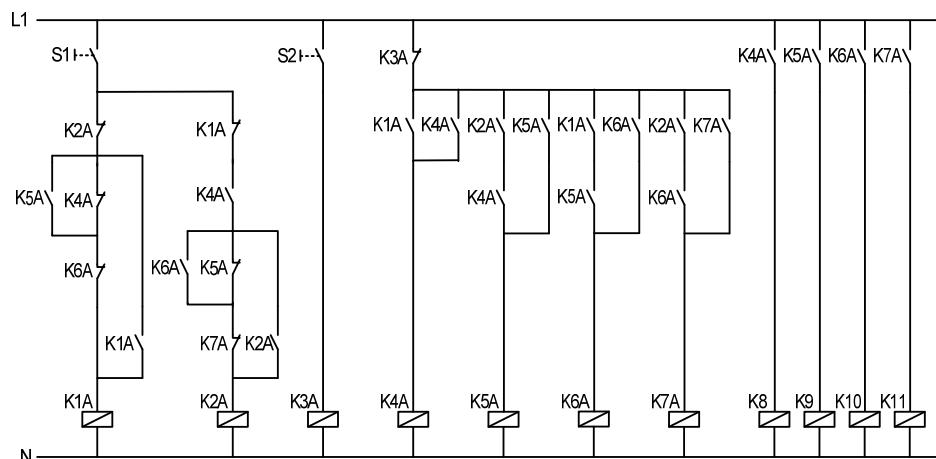
- Nếu lệnh S được viết trước lệnh R thì kết quả thu được sẽ là kết quả của lệnh R nếu cả hai điều kiện cho S và R cùng ở mức logic “1” nghĩa là đối tượng điều khiển ở mức logic “0”.
- Nếu lệnh R được viết trước lệnh S thì kết quả thu được sẽ là kết quả của lệnh S nếu cả hai điều kiện cho S và R cùng 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 nhấn nút nhấn S1 thì một bộ tụ bù được đóng vào lưới điện. Để cắt tụ bù ra khỏi lưới thì 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.

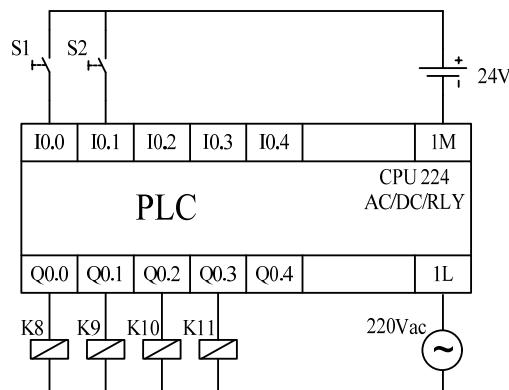
Khi lập bảng ký hiệu nên ghi chú đầy đủ thông tin để dễ dàng trong quá trình viết chương trình.

Bảng ký hiệu

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

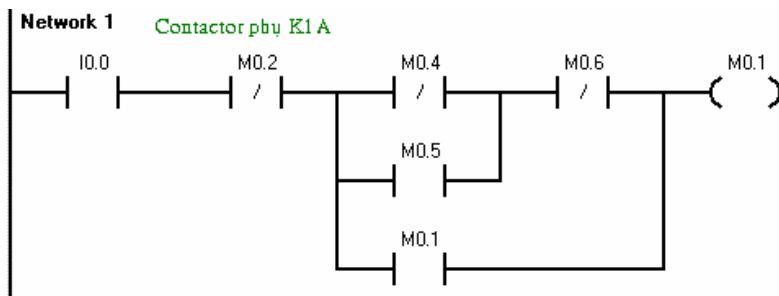
K10	Q0.2	Contactor chính K10, đóng tụ bù C3
K11	Q0.3	Contactor chính K11, đóng tụ bù C4
<b>Biến trung gian</b>		
K1A	M0.1	Contactor phụ K1A
K2A	M0.2	Contactor phụ K2A
K3A	M0.3	Contactor phụ K3A
K4A	M0.4	Contactor phụ K4A
K5A	M0.5	Contactor phụ K5A
K6A	M0.6	Contactor phụ K6A
K7A	M0.7	Contactor phụ K7A

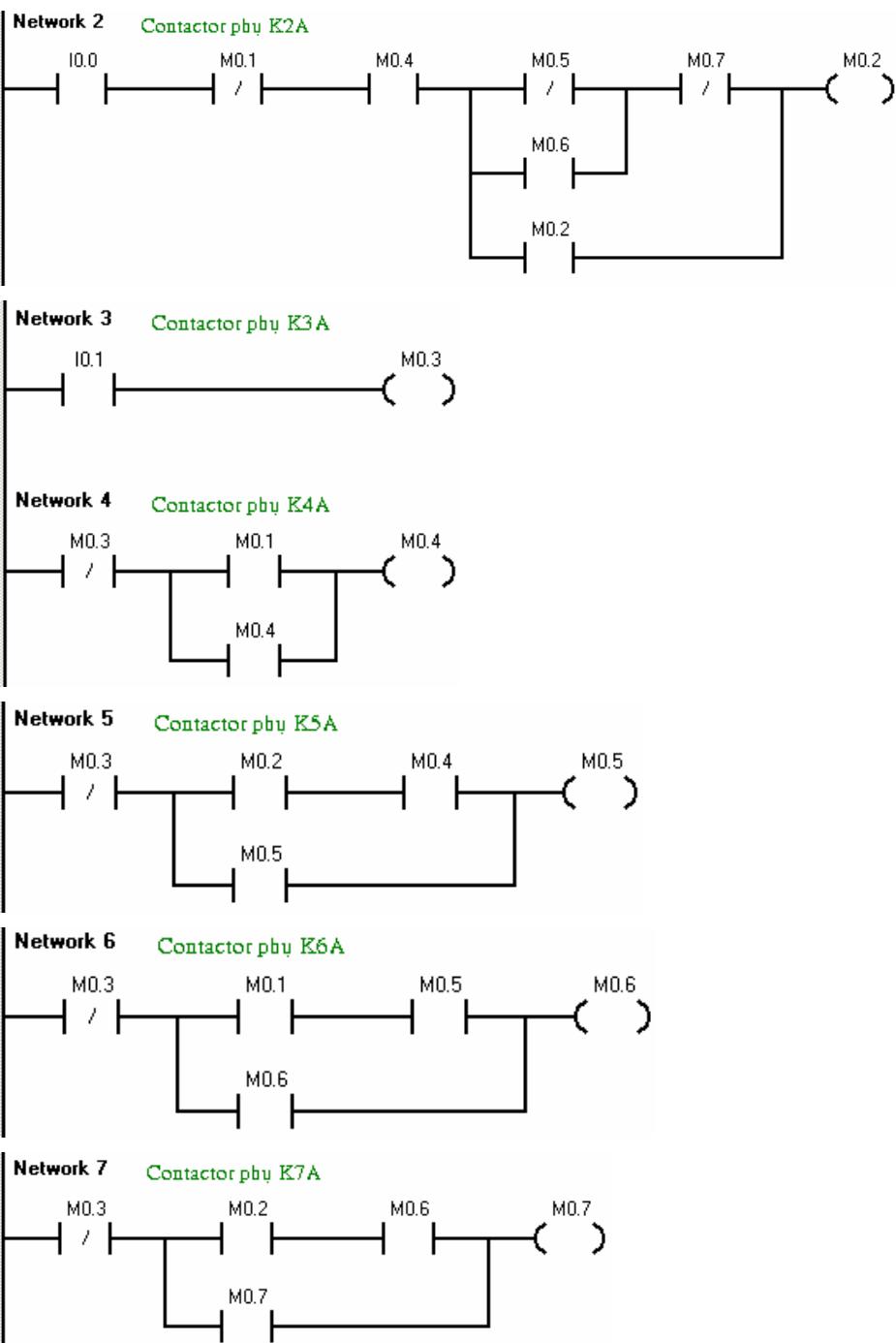
Kết nối dây với PLC:

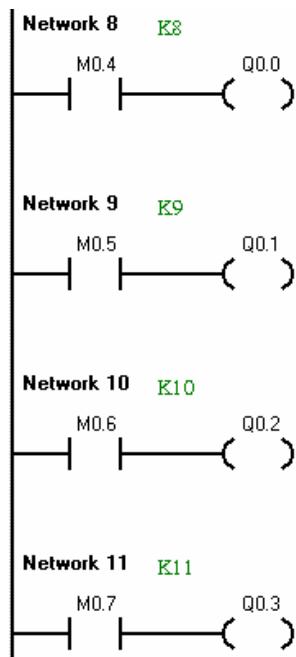


Hình 13.3: Nối dây các ngoại vi với ngõ vào ra PLC khi điều khiển bằng PLC

Chương trình PLC ở LAD:







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**

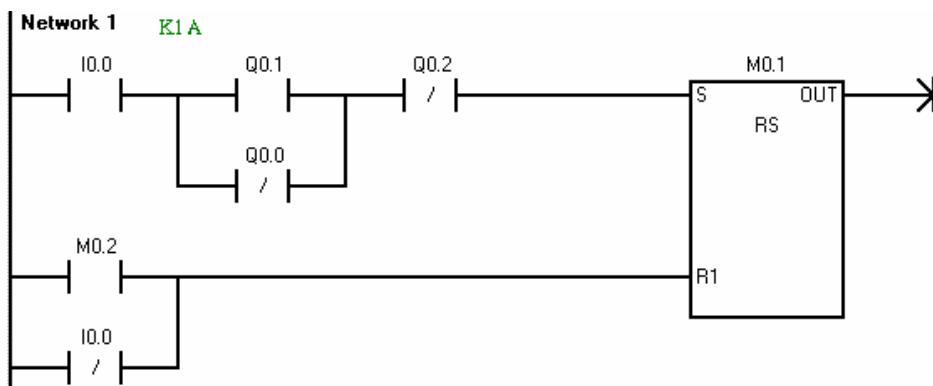
Theo cách thức điều khiển đặt ra, cứ mỗi lần tác động S1 thì một contactor chính được đóng điện, tác động S2 thì cắt điện toàn bộ.

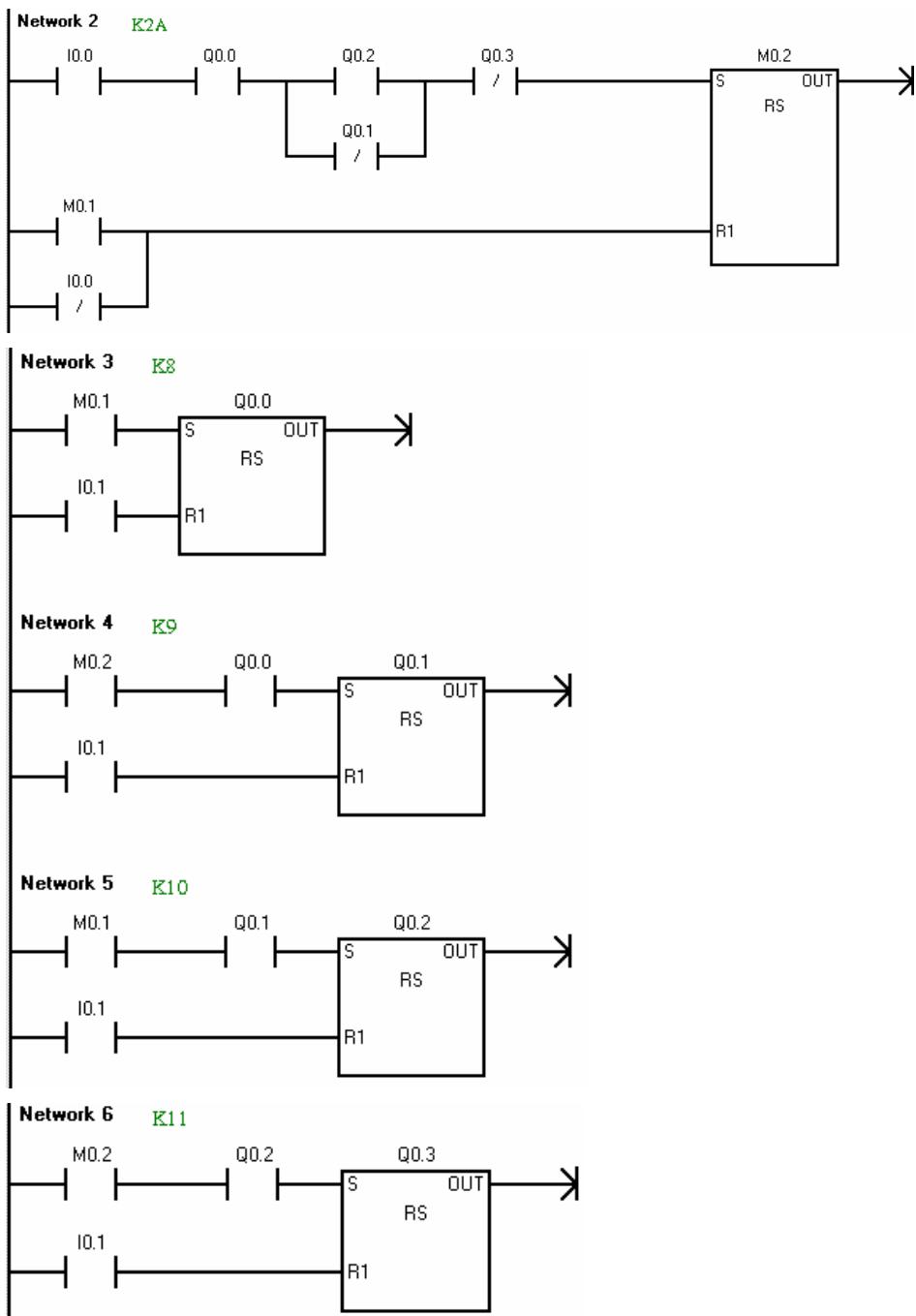
Mục đích của việc thêm các contactor phụ là để tăng thêm số lượng tiếp điểm. Nếu thực hiện bằng chương trình ta có thể đưa trực tiếp ra các ngõ ra từ Q0.0 đến Q0.3 mà không cần phải qua các ô nhớ M0.4 đến M0.7. M0.3 cũng có thể loại bỏ, thay thế trực tiếp bằng nút nhấn S2 (I0.1).

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:

<b>Network 1</b>	<b>K1A</b>	<b>Network 4</b>	<b>K9</b>
LD	I0.0	LD	M0.2
LD	Q0.1	A	Q0.0
ON	Q0.0	LD	I0.1
ALD		NOT	
AN	Q0.2	LPS	
LD	M0.2	A	Q0.1
ON	I0.0	=	Q0.1
NOT		LPP	
LPS		ALD	
A	M0.1	O	Q0.1
=	M0.1	=	Q0.1
LPP			
ALD			
O	M0.1		
=	M0.1		
<b>Network 2</b>	<b>K2A</b>	<b>Network 5</b>	<b>K10</b>
LD	I0.0	LD	M0.1
A	Q0.0	A	Q0.1
LD	Q0.2	LD	I0.1
ON	Q0.1	NOT	
ALD		LPS	
AN	Q0.3	A	Q0.2
LD	M0.1	=	Q0.2
ON	I0.0	LPP	
NOT		ALD	
LPS		O	Q0.2
A	M0.2	=	Q0.2
=	M0.2		
LPP			
ALD			
O	M0.2		
=	M0.2		
<b>Network 3</b>	<b>K8</b>	<b>Network 6</b>	<b>K11</b>
LD	M0.1	LD	M0.2
LD	I0.1	A	Q0.2
NOT		LD	I0.1
LPS		NOT	
A	Q0.0	LPS	
=	Q0.0	A	Q0.3
LPP		=	Q0.3
ALD		LPP	
O	Q0.0	ALD	
=	Q0.0	O	Q0.3
		=	Q0.3

### 13.2.2 Thiết bị nghiên

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

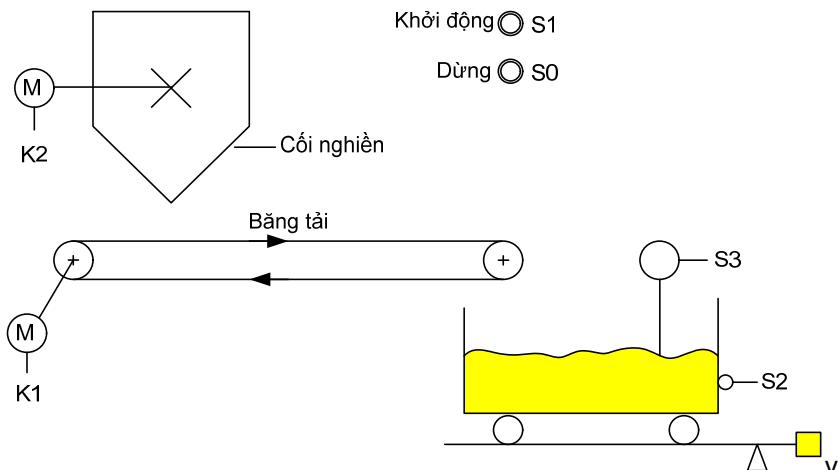
Quá trình vận chuyển vật liệu đã được nghiên được khởi động nếu xe đã vào vị trí vận chuyển và nút nhấn S1 được ấn. Để đảm bảo an toàn thì

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

Khi xe đàm (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.

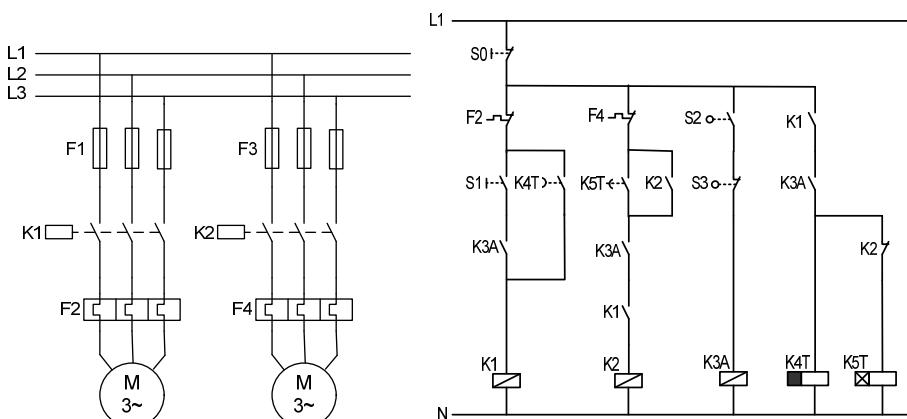
Trong quá trình hoạt động có thể dừng băng bằng nút nhấn S0.

### 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 đàm 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 contactor thể hiện và không có sự thay đổi nào với các bộ phát tín hiệu này.

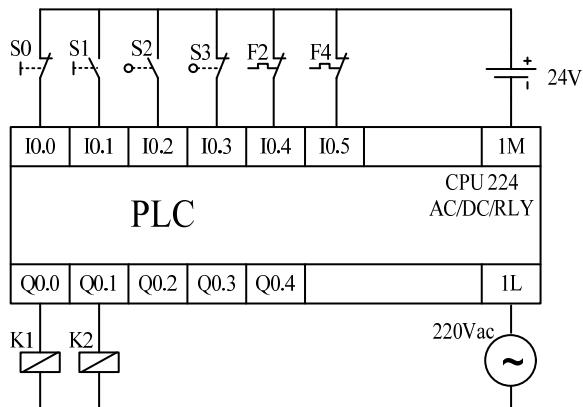
Các contactor chính K1 và K2 cần phải có 2 ngõ ra để điều khiển  
Contactor phụ K3A được thay thế bằng một ô nhớ.

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**

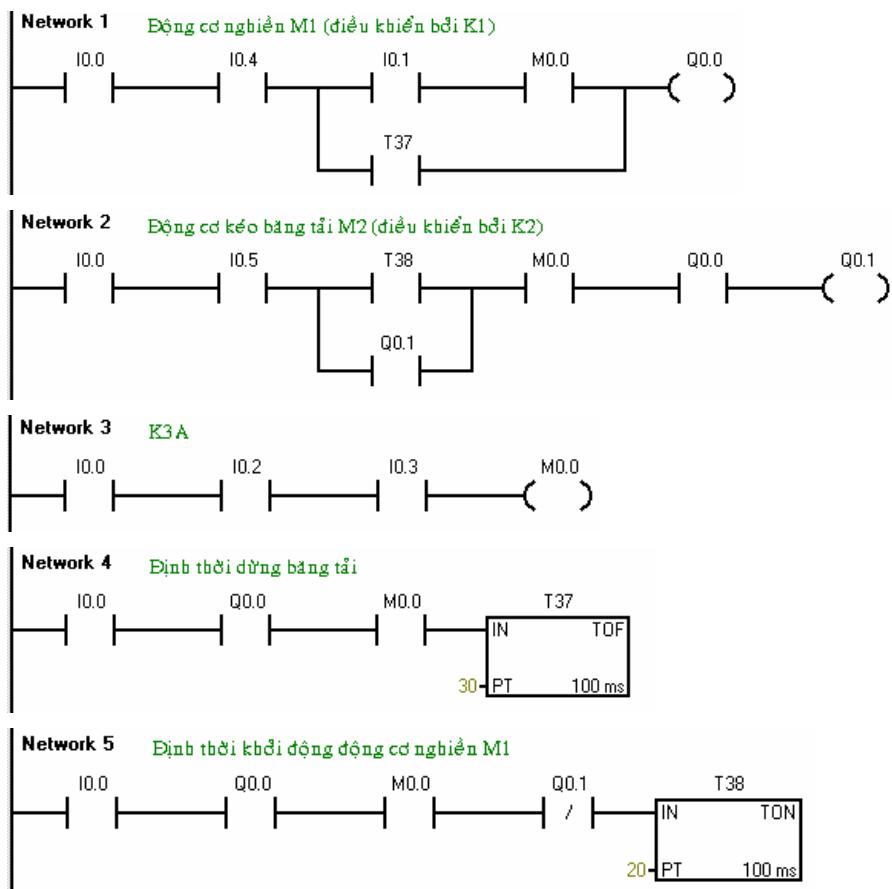
<b>Ký hiệu</b>	<b>Địa chỉ (PLC)</b>	<b>Chú thích</b>
<b>Biến ngõ vào</b>		
S0	I0.0	Nút nhấn dừng, thường đóng (NC)
S1	I0.1	Nút nhấn khởi động hệ thống, thường mở (NO)
S2	I0.2	Công tắc hành trình, báo xe đúng vị trí (NO)
S3	I0.3	Tín hiệu báo xe đàm, thường đóng (NC)
F2	I0.4	Tiếp điểm bảo vệ quá dòng M1, (NC)
F4	I0.5	Tiếp điểm bảo vệ quá dòng M2, (NC)
<b>Biến ngõ ra</b>		
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
<b>Biến trung gian</b>		
K3A	M0.0	Contactor phụ K3A
<b>Bộ định thời</b>		
K4T	T37	OFF delay timer, định thời dừng băng tải, 3s
K5T	T38	ON delay 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      **Bộng 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      **Bộng 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 tử. Dù các van xung điện tử hay van điện tử 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:

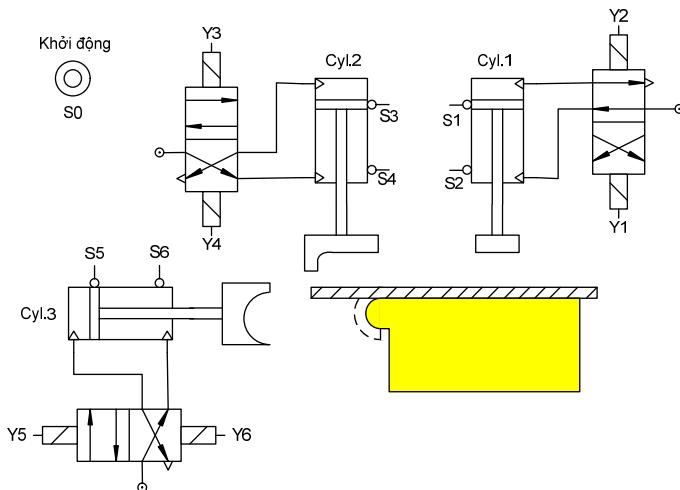
- Khâu điều chỉnh của xylanh làm việc được thay thế bằng van điện từ.
- Tất cả các van xung được thay thế bằng khâu nhớ RS.
- Xác định được tính logic của mạch.
- Chuyển đổi mạch thành chương trình PLC.

### 13.3.1 Máy uốn thanh kim loại

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

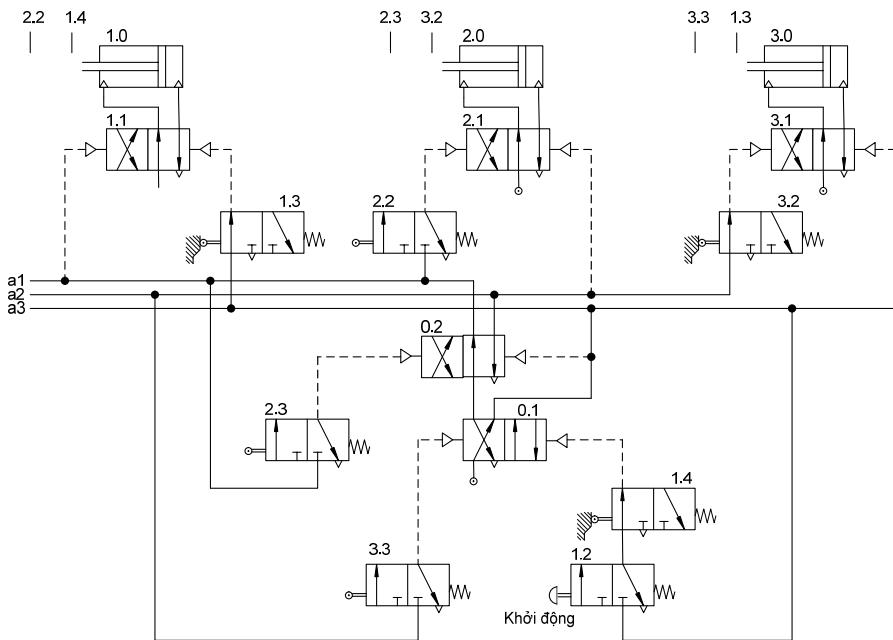
- Thanh kim loại cần uốn được đặt lên khuôn uốn
- Án nút khởi động S0 thì xy lanh Cyl.1 hạ xuống để giữ lấy thanh kim loại.
- Khi thanh kim loại được giữ chặt (nhận biết bởi công tắc hành trình S2) thì xy lanh Cyl.2 hạ xuống để uốn thanh kim loại vuông góc trước. Sau khi uốn xong thì tự động nâng lên 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:*

Tùy 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 công nghệ).

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

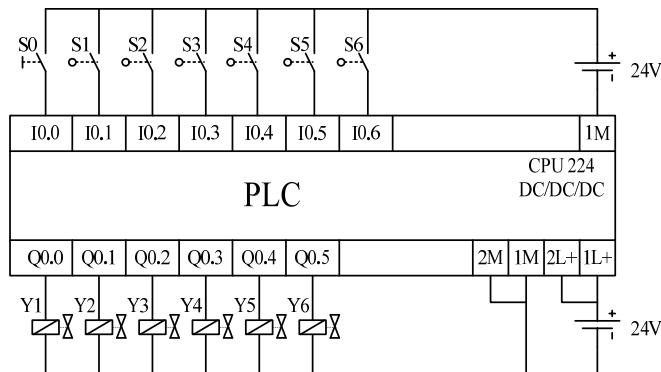
a3= M0.0

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

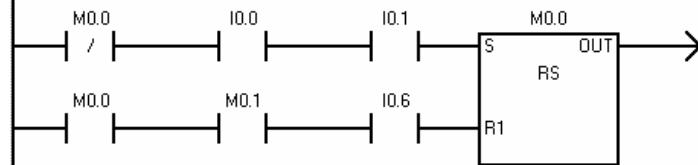
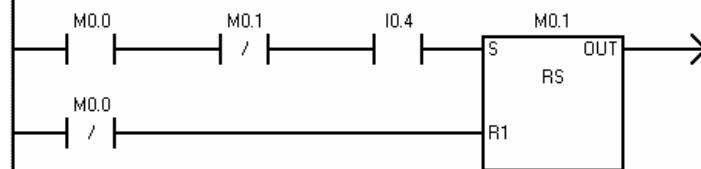
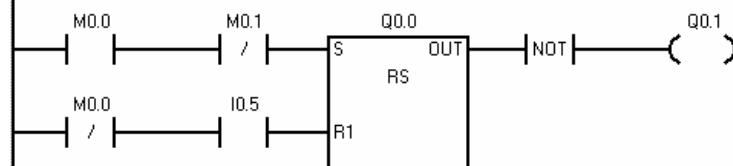
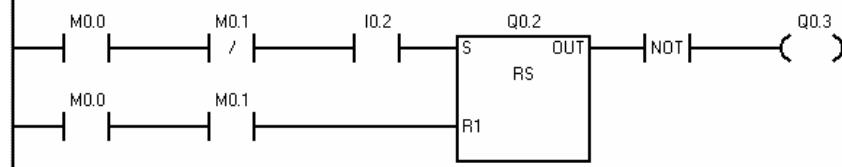
Ký hiệu	Địa chỉ (PLC)	Chú thích
<b>Biến ngõ vào</b>		
S0	I0.0	Nút nhấn khởi động, thường mở
S1	I0.1	CTHT nhận biết vị trí cơ bản xy lanh Cyl.1
S2	I0.2	CTHT nhận biết vị trí giữ thanh kim loại của xy lanh Cyl.1
S3	I0.3	CTHT nhận biết vị trí cơ bản xy lanh Cyl.2
S4	I0.4	CTHT nhận biết vị trí uốn của xy 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
<b>Biến ngõ ra</b>		
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
<b>Biến trung gian</b>		
Van 0.1	M0.0	Van 0.1
Van 0.2	M0.1	Van 0.2

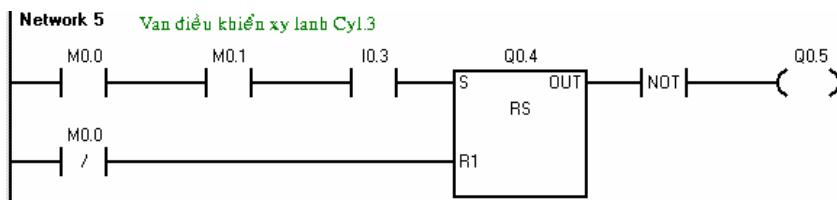
Kết nối dây với PLC:



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

Chương trình PLC ở LAD:

**Network 1 Van xung 0.1****Network 2 Van xung 0.2****Network 3 Van điều khiển xy lanh Cyl.1****Network 4 Van điều khiển xy lanh Cyl.2**



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 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 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 5 Van điều khiển xy lanh Cyl.3**

```

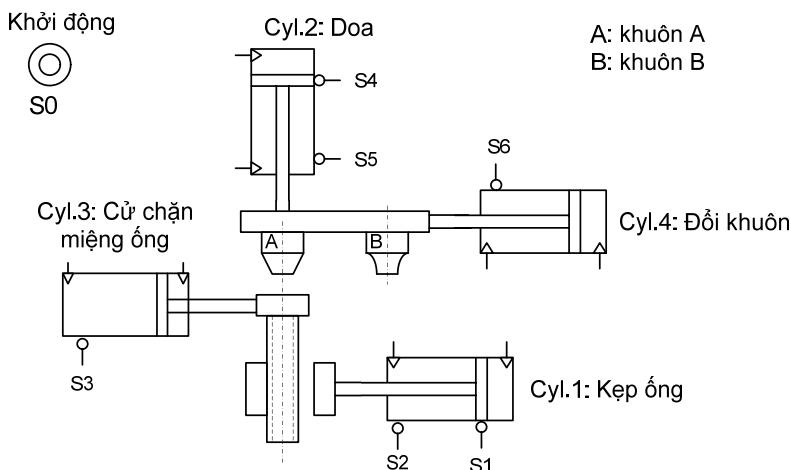
LD    M0.0
A    M0.1
A    I0.3
LDN  M0.0
NOT
LPS
A    Q0.4
=    Q0.4
LPP
ALD
O    Q0.4
=    Q0.4
NOT
=    Q0.5
  
```

### 13.3.2 Máy dòa 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:

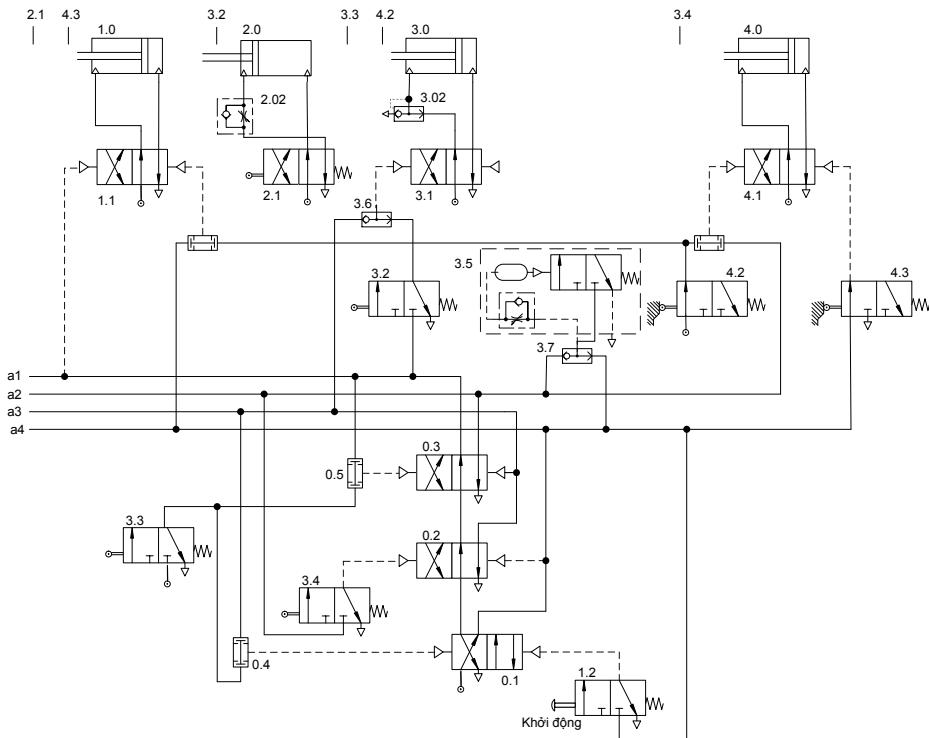
Người vận hành đặt ống kim loại cần doa miệng vào vị trí sao cho miệng ống phải chạm vào cùi chặn miệng ống. Sau đó ấn nút nhấn S0, xy lanh Cyl.1 sẽ kẹp ống lại. Khi ống đã được kẹp thì cùi chặn miệng ống tự động rút về. Xy lanh Cyl.2 sẽ hạ xuống doa miệng ống theo khuôn A. Thời gian doa khoảng 3s. Sau đó xy lanh Cyl.2 rút về và khuôn B được 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 miệng 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:

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

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

$$a3 = M0.0 \& M0.1$$

$$a4 = \overline{M0.0}$$

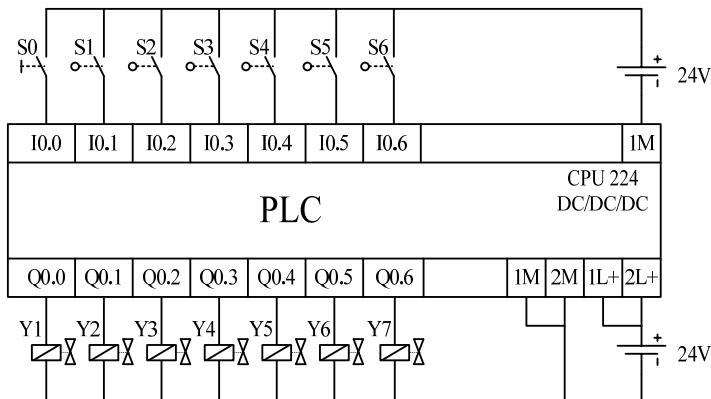
Khâu điều chỉnh trễ 3.5 được thay thế bằng một timer.

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

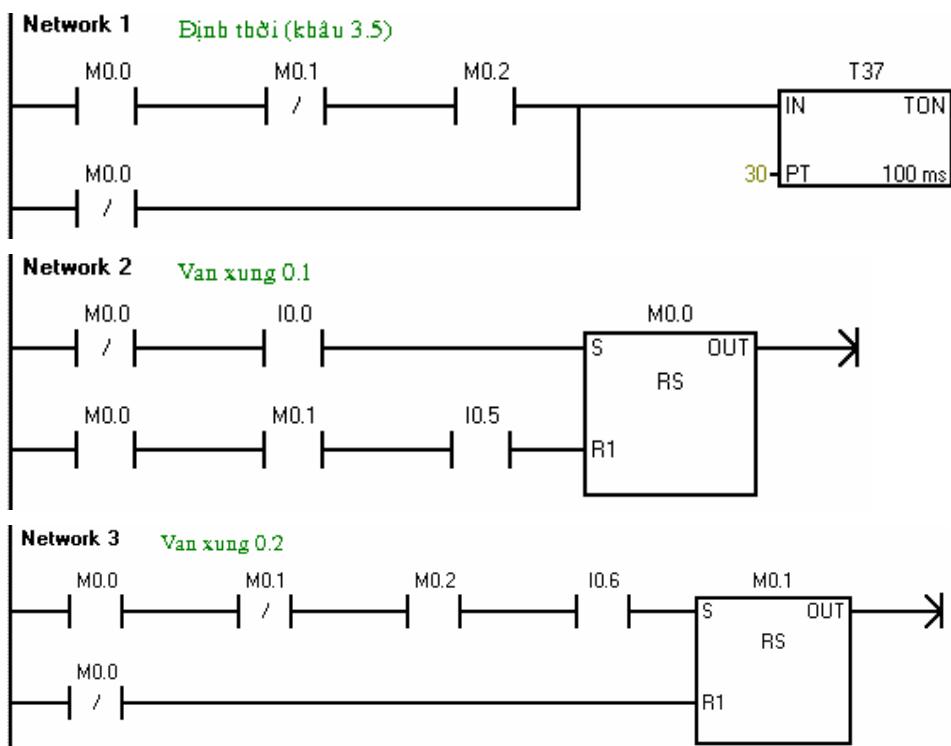
Ký hiệu	Địa chỉ (PLC)	Chú thích
<b>Biến ngõ vào</b>		
S0	I0.0	Nút nhấn khởi động, thường mở
S1	I0.1	CTHT nhận biết vị trí cơ bản xy lanh Cyl.1
S2	I0.2	CTHT nhận biết vị trí giữ ống kim loại của xy lanh Cyl.1
S3	I0.3	CTHT nhận biết vị trí rút về của xy lanh Cyl.2
S4	I0.4	CTHT nhận biết vị trí rút về của 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
<b>Biến ngõ ra</b>		
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
<b>Biến trung gian</b>		
Van 0.1	M0.0	Van 0.1
Van 0.2	M0.1	Van 0.2
Van 0.3	M0.1	Van 0.3
<b>Bộ định thời</b>		
Delay 3.5	T37	ON delay timer, định thời doa, 3s

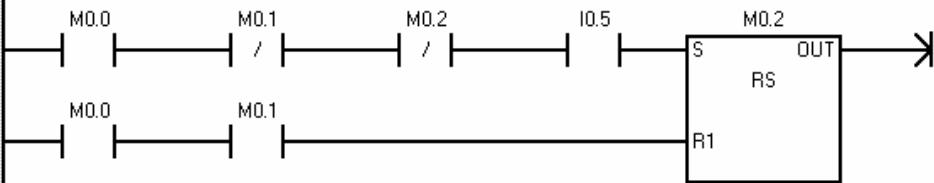
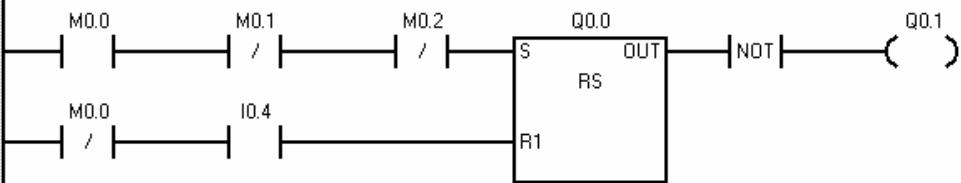
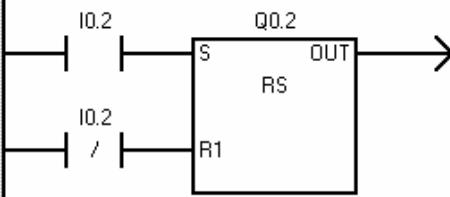
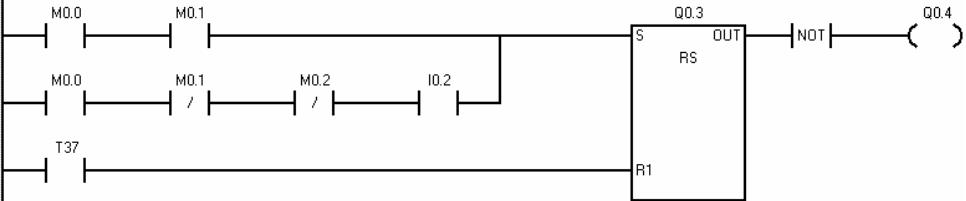
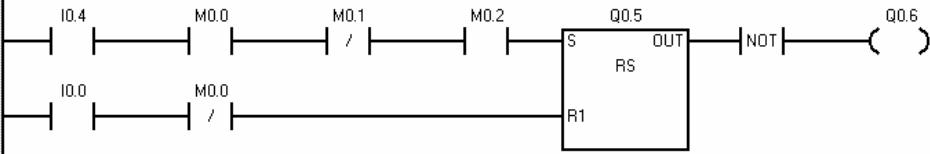
Kết nối dây với PLC:



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

Chương trình viết ở LAD:



**Network 4 Van xung 0.3****Network 5 Van xung 1.1****Network 6 Van xung 2.1****Network 7 Van xung 3.1****Network 8 Van xung 4.1**

Chương trình viết ở STL:

<b>Network 1</b>	<b>Định thời (khoảng 3.5)</b>	<b>Network 5</b>	<b>Van xung 1.1</b>
LD	M0.0	LD	M0.0
AN	M0.1	AN	M0.1
A	M0.2	AN	M0.2
ON	M0.0	LDN	M0.0
TON	T37, 30	A	I0.4
		NOT	
		LPS	
		A	Q0.0
		=	Q0.0
		LPP	
		ALD	
		O	Q0.0
		=	Q0.0
		NOT	
		=	Q0.1
<b>Network 2</b>	<b>Van xung 0.1</b>	<b>Network 6</b>	<b>Van xung 2.1</b>
LDN	M0.0	LD	I0.2
A	I0.0	LDN	I0.2
LD	M0.0	NOT	
A	M0.1	LPS	
A	I0.5	A	Q0.2
NOT		=	Q0.2
LPS		LPP	
A	M0.0	ALD	
=	M0.0	O	Q0.2
LPP		=	Q0.2
ALD		NOT	
O	M0.0	LPS	
=	M0.0	A	Q0.2
<b>Network 3</b>	<b>Van xung 0.2</b>	<b>Network 7</b>	<b>Van xung 3.1</b>
LD	M0.0	LD	M0.0
AN	M0.1	A	M0.1
A	M0.2	LD	M0.0
A	I0.6	AN	M0.1
LDN	M0.0	AN	M0.2
NOT		A	I0.2
LPS		OLD	
A	M0.1	LD	T37
=	M0.1	NOT	
LPP		LPS	
ALD		A	Q0.3
O	M0.1	=	Q0.3
=	M0.1	LPP	
<b>Network 4</b>	<b>Van xung 0.3</b>	<b>Network 7</b>	<b>Van xung 3.1</b>
LD	M0.0	ALD	
AN	M0.1	O	Q0.3
AN	M0.2	=	Q0.3
A	I0.5	NOT	
LD	M0.0	LPS	
A	M0.1	A	Q0.4
NOT		=	Q0.4
LPS			
A	M0.2		
=	M0.2		
LPP			
ALD			
O	M0.2		
=	M0.2		

Network 8 Van xung 4.1

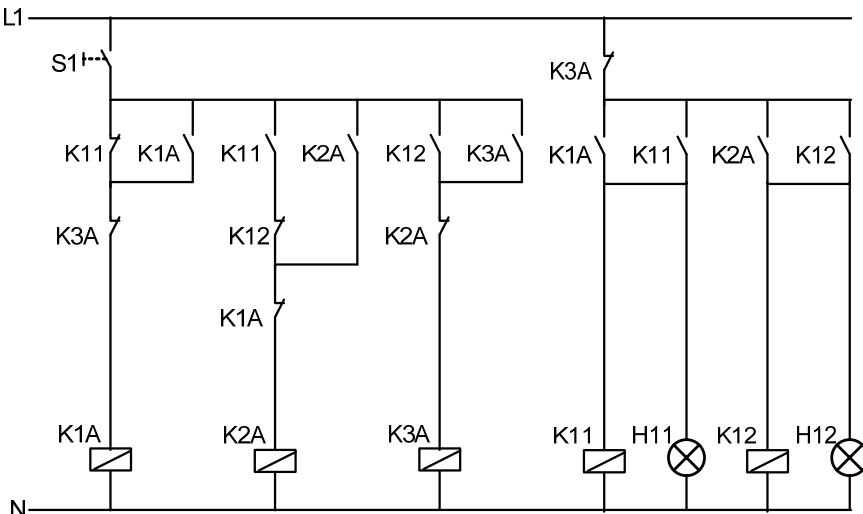
LD	I0.4
A	M0.0
AN	M0.1
A	M0.2
LD	I0.0
AN	M0.0
NOT	
LPS	
A	Q0.5
=	Q0.5
LPP	
ALD	
O	Q0.5
=	Q0.5
NOT	
=	Q0.6

## 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 nhiệt 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

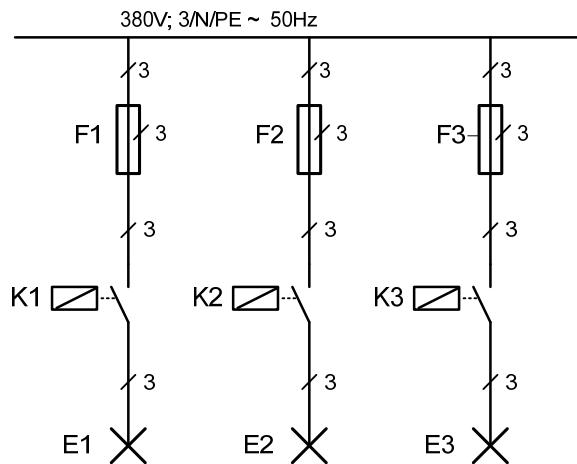
Sau 40s tắt cả các đèn đều tắt

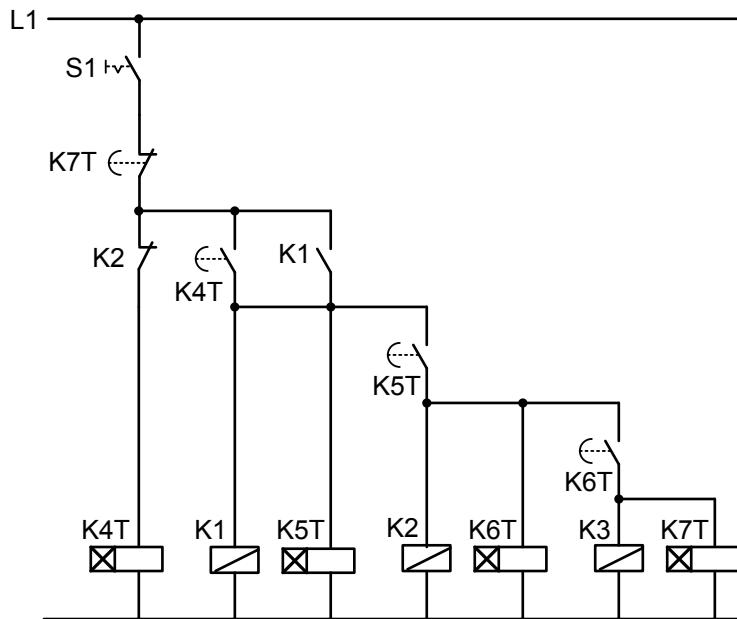
Sau đó bắt đầu tự động lại chu kỳ mới

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

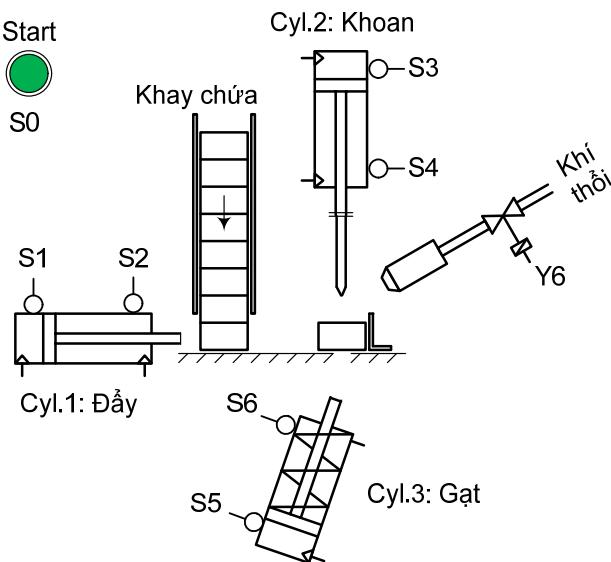
1. Thiết lập bảng ký hiệu.
2. Vẽ sơ đồ kết nối dây với PLC
3. Viết chương trình điều khiển theo hai cách:
  - a. Sơ đồ kết nối dây cứng
  - b. Theo yêu cầu công nghệ

#### **Sơ đồ mạch động lực:**

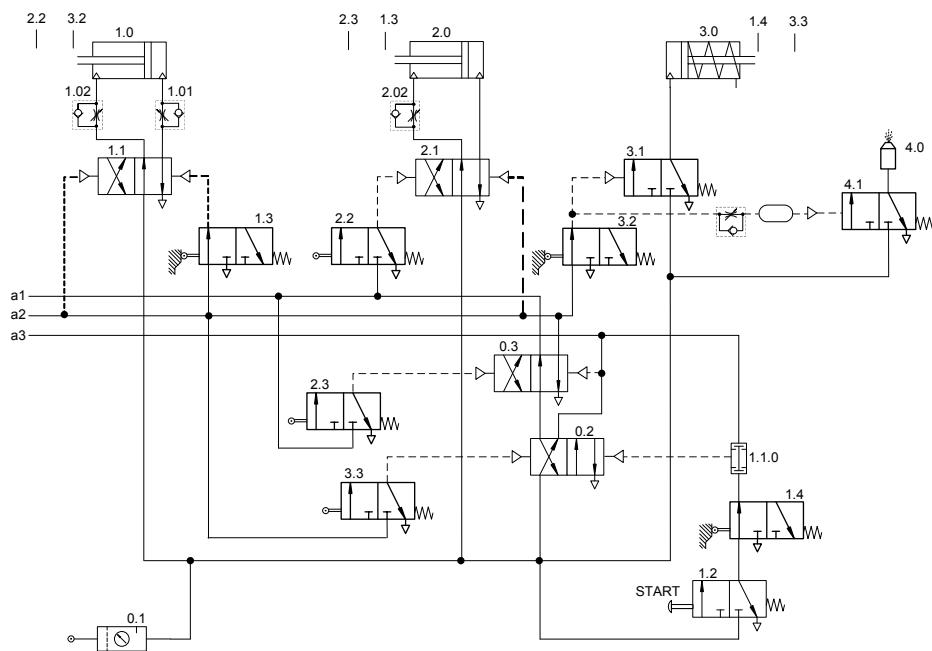


**Sơ đồ mạch điều khiển:****BT 13.3 Máy khoan**

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.

Tùy 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

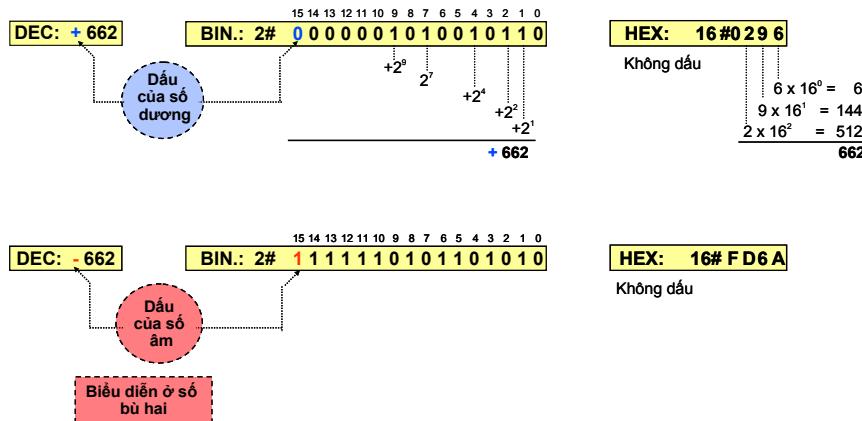
#### 14.1.1 Kiểu dữ liệu Integer (INT)

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 mă 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 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.

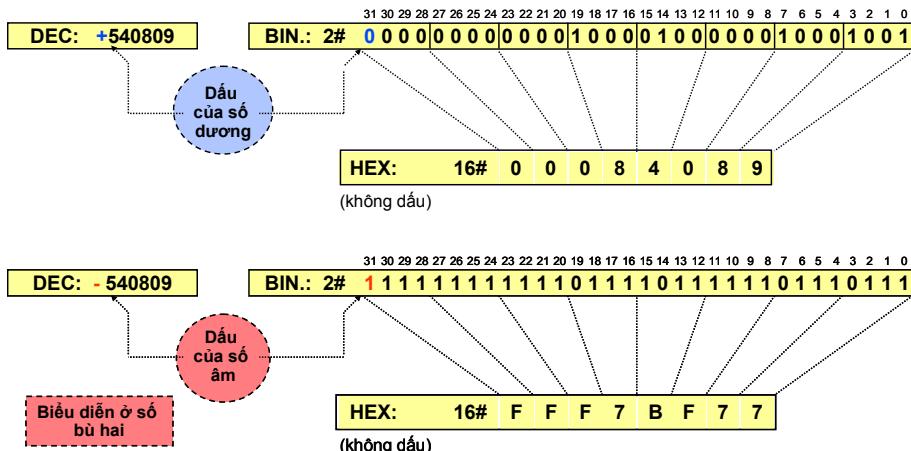
Hàng số trong dạng số Hexadecimal không được sử dụng cho các giá trị số integer.

### 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.

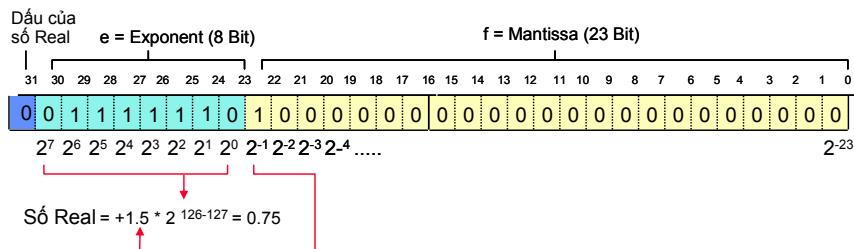
Phạm vi biểu diễn của số thực từ  $-1.175495 \cdot 10^{-38}$  đến  $3.402823 \cdot 10^{38}$

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).

Dạng 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.

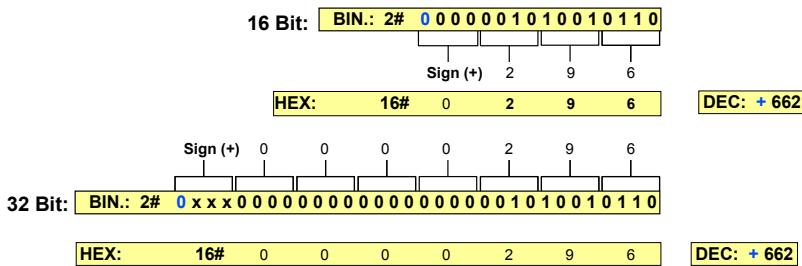
Mỗi chữ số của số decimal được mã hóa ở bốn bit. Vì chữ số cao nhất của decimal là 9 nên bốn bit được sử dụng và có mã nhị phân tương ứng cho các chữ số decimal như sau:

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

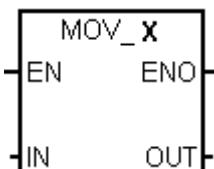
Để sao chép các dữ liệu kiểu byte, word, double word kể cả số thực (real) từ nơi này đến nơi khác ta sử dụng lệnh Move.

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.

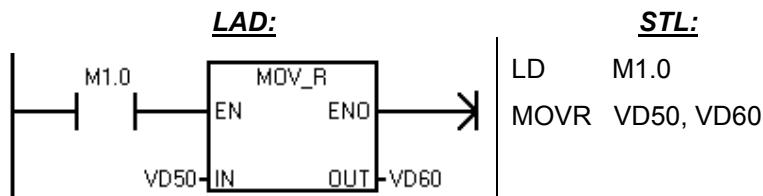
Để lấy lệnh MOV ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng Move trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là:

MOV B: sao chép Byte  
 MOV W: sao chép Word

MOV D: sao chép double Word  
 MOV R: 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:



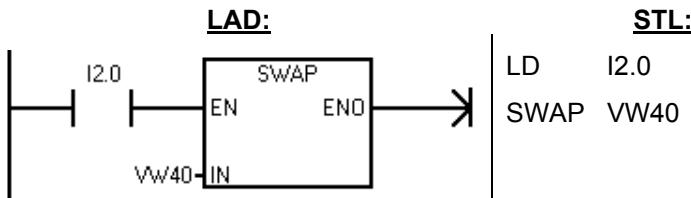
\* Cú pháp dùng lệnh SWAP trong LAD như sau:

LAD	Toán hạng
	<b>IN (Word):</b> 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:



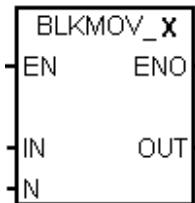
#### 14.2.2 Các lệnh sao chép một mảng lớn dữ liệu

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:

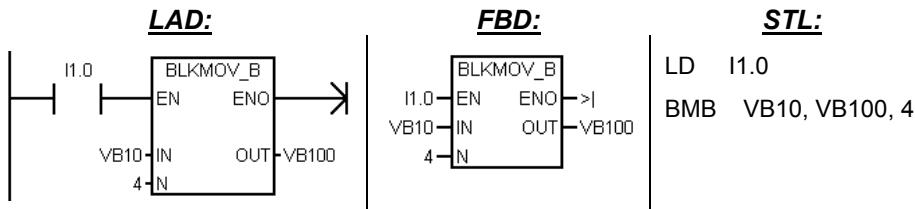


Với:

- \* **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 (saو chép mảng Byte), BLKMOV\_W (saو chép mảng Word), BLKMOV\_D (saو 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:



Giả thiết nội dung của mảng cần sao chép là:

Byte	VB10	VB11	VB12	VB13
Nội dung	20	21	22	23

Kết quả thu được 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: <i>Byte, Int, DInt, Real</i>	> : Lớn hơn >= : Lớn hơn hoặc bằng == : Bằng nhau <> : Không bằng nhau (khác nhau) <= : Bé hơn hoặc bằng < : Bé hơn	Dữ liệu có thể là: <i>Byte, Int, DInt, Real</i>

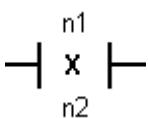
Khi so sánh giá trị Byte (B) thì không cần phải để ý đến dấu của toán hạng, ngược lại khi so sánh là các số Int (I), Dint (D), Real (R) thì phải chú ý đến dấu của toán hạng.

Cú pháp tổng quát cho phép toán 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.

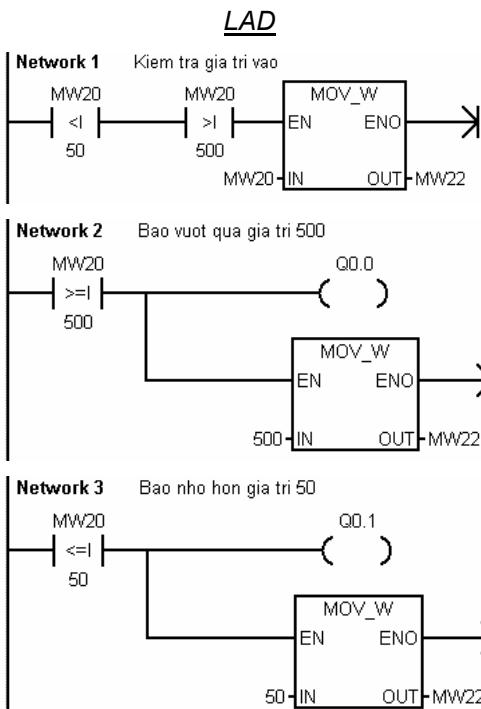
#### Ví dụ 14.2: Giới hạn giá trị.

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:**

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**

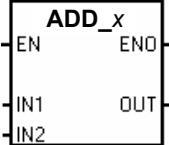
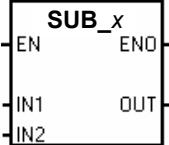
Ở nhiều nhiệm vụ đếm như đếm sản phẩm, đếm số vòng quay, đếm xung .v.v... thì kết quả đếm phải được giám sát. Bên cạnh các phép toán so sánh đã biết cần phải có thêm các phép toán số học như cộng, trừ, nhân, chia. Còn các phép toán khác như sin, cos, tan, PID .... sẽ được khảo sát ở tập 2 của bộ sách *kỹ thuật điều khiển lập trình PLC SIMATIC S7-200*.

**14.4.1. Cộng và trừ**

Các phép toán 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 toán 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:

<b>Phép toán cộng</b>	<b>Phép toán trừ</b>	<b>Chú thích</b>
Biểu diễn ở LAD:  <b>Thực hiện:</b> $IN1 + IN2 = OUT$	Biểu diễn ở LAD:  <b>Thực hiện:</b> $IN1 - IN2 = OUT$	<ul style="list-style-type: none"> <li>* <math>x</math>: có thể là I (Integer), DI (Double integer), R(Real).</li> <li>* EN = "1": cho phép cộng hoặc trừ.</li> <li>* ENO = "0": khi có lỗi.</li> <li>* IN1, IN2, OUT: các ngõ vào ra dạng số có cùng kiểu dữ liệu với <math>x</math>.</li> </ul>
Biểu diễn ở STL: $+I$ IN1, OUT $+D$ IN1, OUT $+R$ IN1, OUT  <b>Thực hiện:</b> $IN1 + OUT = OUT$	Biểu diễn ở STL: $-I$ IN1, OUT $-D$ IN1, OUT $-R$ IN1, OUT  <b>Thực hiện:</b> $OUT - IN1 = OUT$	

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.

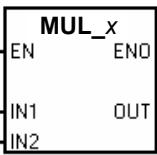
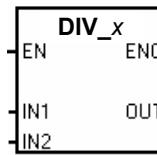
Để lấy lệnh **cộng hoặc trừ số thực (real)** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng  Floating-Point Math trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là:  ADD\_R (cộng số real),  SUB\_R (trừ số real), giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều 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 toán 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 toán 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:

<b>Phép toán nhân</b>	<b>Phép toán chia</b>	<b>Chú thích</b>
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.

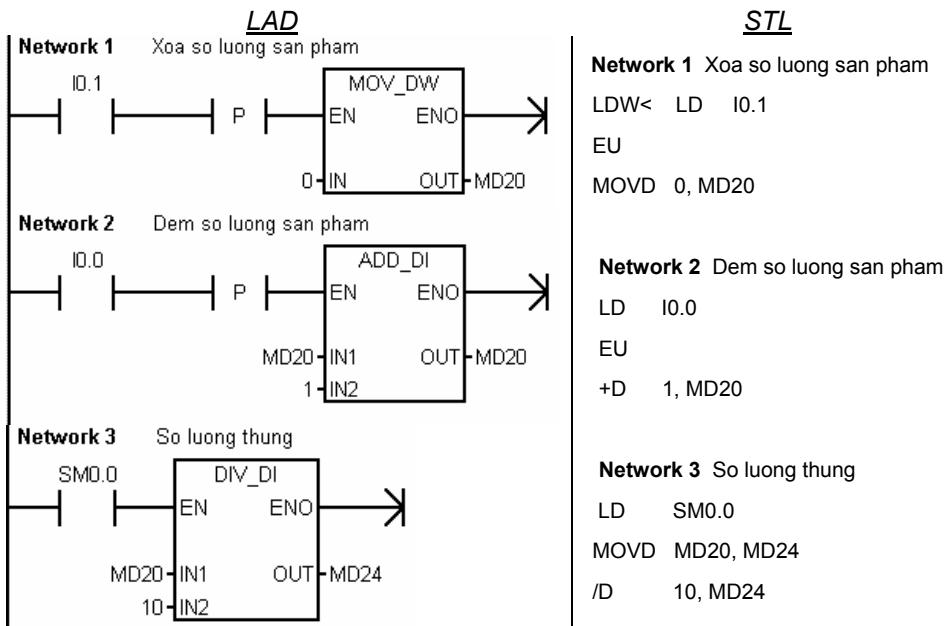
### 14.4.3. Ví dụ phép toán số học

#### 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**

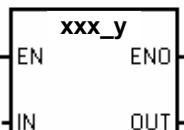
Ký hiệu	Địa chỉ	Chú thích
S1	I0.0	Cảm biến nhận biết sản phẩm
S2	I0.1	Nút nhấn xóa số lượng sản phẩm
So_SP	MD20	Giá trị sản phẩm đếm được
So_Thung	MD24	Số lượng thùng

**Chương trình:****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:

	<u>Lệnh tăng:</u>	<u>Lệnh giảm:</u>
<b>Byte:</b>	<b>INCB OUT</b>	<b>DECB OUT</b>
<b>Word:</b>	<b>INCW OUT</b>	<b>DECW OUT</b>
<b>Double word:</b>	<b>INCD OUT</b>	<b>DECD OUT</b>
<b>Ý nghĩa:</b>	<b>OUT + 1 = OUT</b>	<b>OUT -1 = OUT</b>

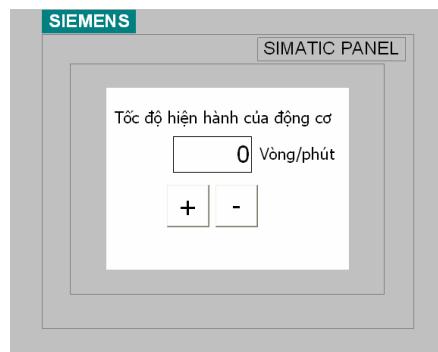
Để lấy lệnh **tăng hoặc giảm thanh ghi** ở màn hình soạn thảo LAD, ta nhấp chuột vào dấu (+) ở biểu tượng  Integer Math trong cây lệnh. Sau đó trỏ chuột vào một trong các lệnh cần lấy là:

- 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.

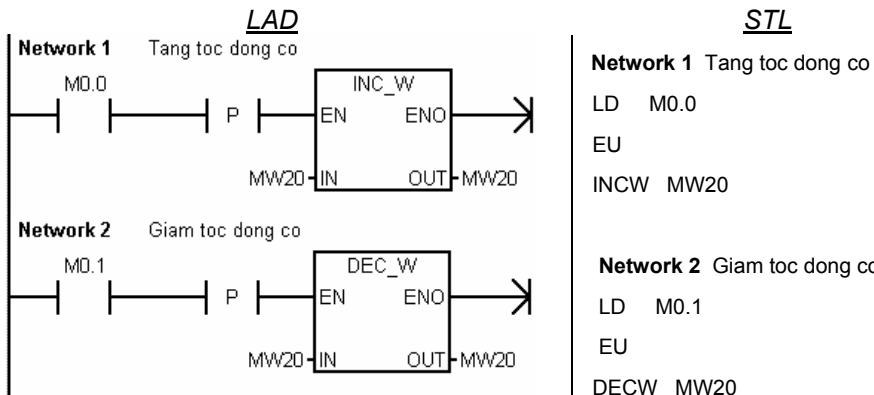


Giải

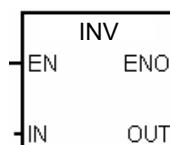
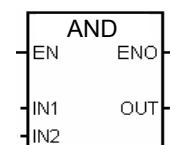
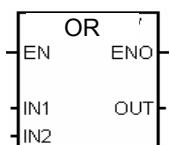
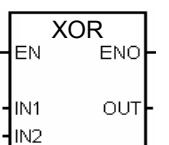
Nhằm giúp cho bạn đọc dễ hiểu, cứ mỗi lần ấn một phím (+) hoặc phím (-) thì tốc độ động cơ tăng hoặc giảm đi một vòng quay. Ở đây có thể có nhiều phương pháp nhưng chỉ giới hạn kiến thức cơ bản trong quyển sách, còn các kiến thức nâng cao xin bạn đọc tập 2.

**Bảng ký hiệu**

Ký hiệu	Địa chỉ	Chú thích
Phím +	M0.0	Tăng tốc động cơ
Phím -	M0.1	Giảm tốc động cơ
TD_Dongco	MW20	Biên tốc độ động cơ

**Chương trình:****14.6. Các phép toán logic số****14.6.1 Các logic số trong S7-200**

Phép toán logic số sẽ thực hiện theo từng bit của hai toán hạng số tương ứng hay một toán 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 (INV), 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:

<input type="checkbox"/> INV_B : đảo byte	<input type="checkbox"/> WOR_B : OR byte
<input type="checkbox"/> INV_W : đảo word	<input type="checkbox"/> WOR_W : OR word
<input type="checkbox"/> INV_DW : đảo dword	<input type="checkbox"/> WOR_DW : OR double word
<input type="checkbox"/> WAND_B : AND byte	<input type="checkbox"/> WXOR_B : XOR byte
<input type="checkbox"/> WAND_W : AND word	<input type="checkbox"/> WXOR_W : XOR word
<input type="checkbox"/> WAND_DW : AND double word	<input type="checkbox"/> WXOR_DW : XOR double word

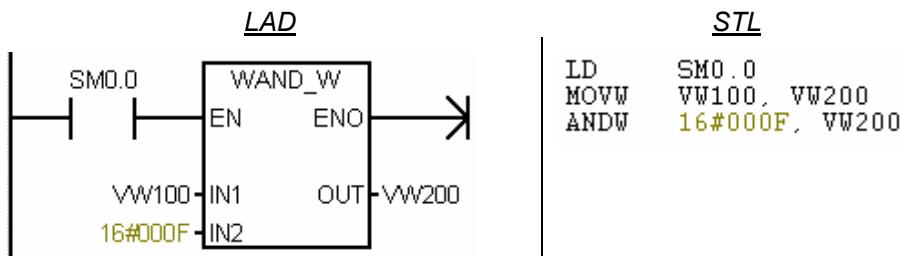
giữ chuột trái, kéo và thả vào vị trí mong muốn. Nhập điều kiện cho ngõ vào EN, nếu lúc nào cũng thực hiện thì sử dụng bit nhớ SM0.0. Nhập các biến cần tăng hoặc giảm ngõ IN. Nhập biến chứa kết quả ở ngõ OUT. (thông thường ngõ vào và ra có chung một biến).

## 14.6.2. Ứng dụng

### 14.6.2.1 Che vị trí các bit

Một ứng dụng của phép toán 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.

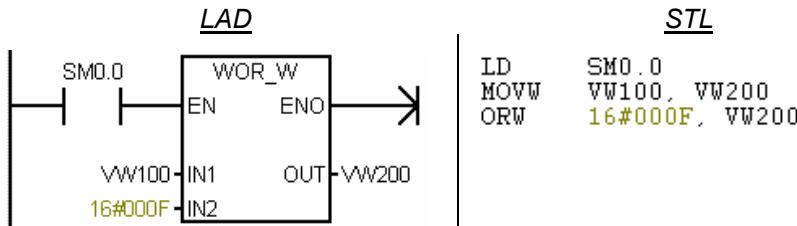
Chương trình:



### 14.6.2.2 Chèn thêm bit

Một ứng dụng của phép toán 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:



## 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.

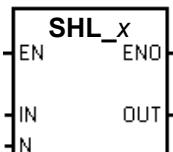
Trong PLC họ S7-200, ngoài ngôn ngữ được biểu diễn theo chuẩn IEC 1131-3, còn có ngôn ngữ được biểu diễn theo chuẩn của hãng sản xuất (Siemens). Các lệnh dịch chuyển thanh ghi được cho như sau:

#### 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:

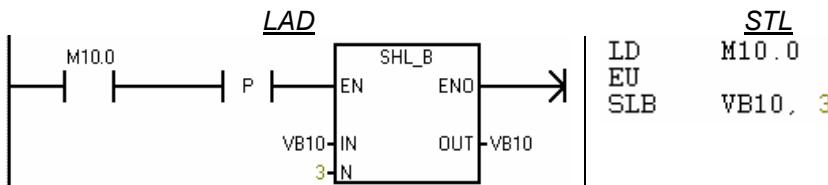
\* Dịch trái byte: SLB OUT, N

\* 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.

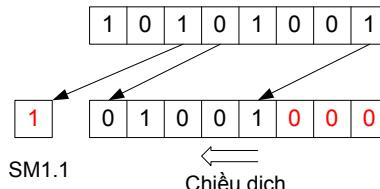
Ví dụ: Khi bit M10.0 từ “0” → “1” thì yêu cầu dịch trái byte VB10 đi 3 vị trí, kết quả chứa vào VB10.



VB10 trước khi dịch:



VB10 sau khi dịch 3 vị trí:

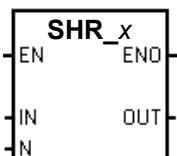


#### 14.7.1.2 Dịch phải

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

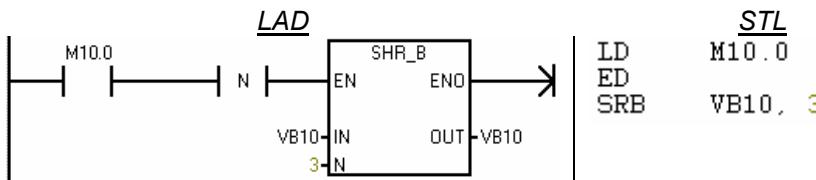
**Chú ý:** Ở STL, thì kết quả sau phép dịch phải sẽ được chứa vào chính thanh ghi cần dịch.

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:

- |                                                         |                                                         |
|---------------------------------------------------------|---------------------------------------------------------|
| <input type="checkbox"/> SHL_B : dịch trái byte         | <input type="checkbox"/> SHR_B : dịch phải byte         |
| <input type="checkbox"/> SHL_W : dịch trái word         | <input type="checkbox"/> SHR_W : dịch phải word         |
| <input type="checkbox"/> SHL_DW : dịch trái double word | <input type="checkbox"/> 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.



VB10 trước khi dịch:

1	0	1	0	1	0	0	1
---	---	---	---	---	---	---	---

VB10 sau khi dịch 3 vị trí

0	0	0	1	0	1	0	1	0
---	---	---	---	---	---	---	---	---

Chiều dịch

### 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.

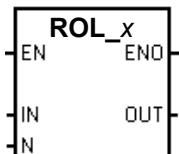
Trong PLC họ S7-200, ngoài ngôn ngữ được biểu diễn theo chuẩn IEC 1131-3, còn có ngôn ngữ được biểu diễn theo chuẩn của hãng sản xuất (Siemens). Các lệnh quay thanh ghi được cho như sau:

#### 14.7.2.1 Quay trái

Ở phép toán quay trái, cho phép quay byte, word và double word.

Cú pháp chung biểu diễn ở 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 ở STL là:

- \* Quay trái byte: RLB OUT, N
- \* Quay trái word: RLW OUT, N
- \* Quay trái double word: RLD OUT, N

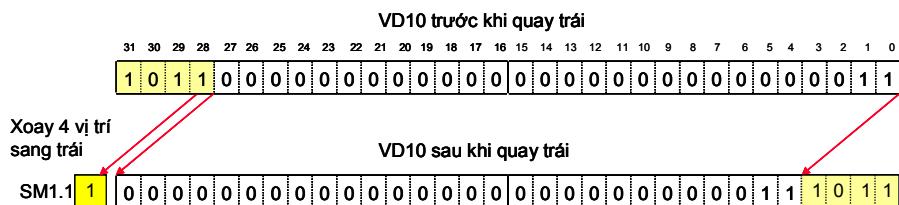
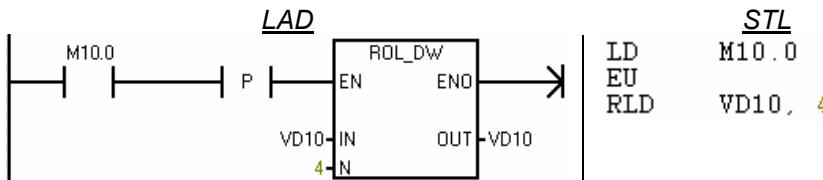
**Chú ý:** Ở STL, thì kết quả sau phép quay trái sẽ được chứa vào chính thanh ghi cần quay.

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:

- |                                                         |                                                         |
|---------------------------------------------------------|---------------------------------------------------------|
| <input type="checkbox"/> ROL_B : Quay trái byte         | <input type="checkbox"/> ROR_B : Quay phải byte         |
| <input type="checkbox"/> ROL_W : Quay trái word         | <input type="checkbox"/> ROR_W : Quay phải word         |
| <input type="checkbox"/> ROL_DW : Quay trái double word | <input type="checkbox"/> 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.

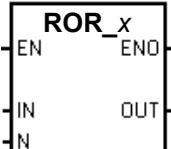


#### **14.7.2.2 Quay phải**

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:

Vój:



- \* 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 chèn vào chính thanh ghi cần quay.